OPTIMIZATION ACROSS DISCIPLINES

A Transdisciplinary Field Guide to
Better Choices, Designs, Systems, and Works

OPTIMIZATION ACROSS DISCIPLINES

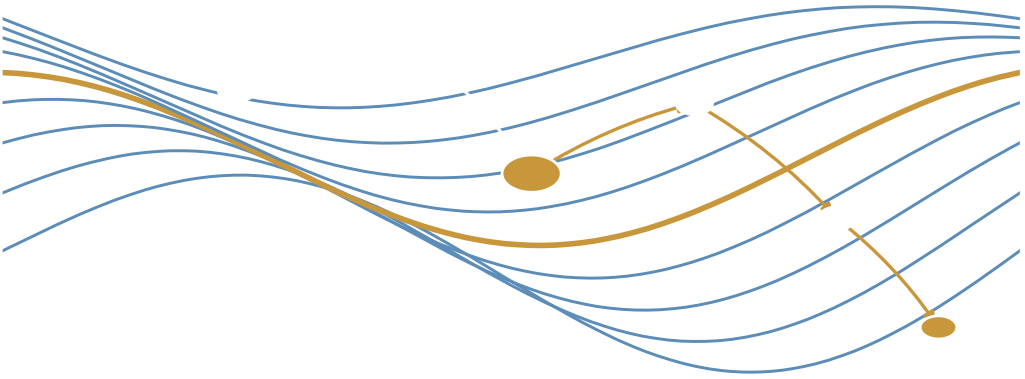
A Transdisciplinary Field Guide to Better Choices, Designs, Systems, and Works

Idea: Dmitry Vostokov

Author: GPT-5.6 Sol Max

Referee: Fable 5.1 Extra

Version 39 · 2026



Copyright

ISBN-13: 978-1-919135-00-7

Version 39 · 2026

The book provides general educational discussion. It is not a substitute for professional engineering, medical, legal, financial, safety, or regulatory judgment. Models, standards, legislation, guidance, and software practices must be checked in their current operational context.

About this book

We can consider optimization as a branch of mathematics and as a process of choosing among alternatives. In a mathematical formulation, we select variables, define an objective function and constraints, and seek a minimum or maximum. Similar comparisons occur in other disciplines. Engineers compare mass and strength, physicians compare benefit and harm under uncertainty, and computer scientists compare time and memory requirements. Resource allocation also requires a choice of fairness criteria. In creative work, constraints affect form, playability, and expression. A writer may select, revise, compress, expand, or reorder a text without using a single score or a fixed endpoint.

In this book, we develop a common vocabulary for these activities and examine the differences between their applications. We begin with mathematical foundations and proceed to computing, engineering, medicine, biology, economics, social science, the humanities, and creative work. We also consider the limits of optimization. Some values resist measurement; some goals change when measured; objectives can conflict; and participants may adapt to a metric. In such cases, restraint, satisficing, care, exploration, or open-ended creation may be more appropriate than maximization.

The domain chapters show how a recurring optimization structure appears in different contexts. The practical sections explain how we can formulate a problem, select a method, test the result, and revise the model or its use. Equations make mathematical relationships explicit. The accompanying explanations and illustrations describe their meaning and limits.

Scope and intellectual stance

We cannot enumerate every possible discipline in one volume. Instead, we use a framework that can be applied to an existing field or extended to a new one. The domain chapters use the following seven-part schema:

1. the decision or design variables;
2. the space of feasible possibilities;
3. the objective, preference relation, or evaluation process;
4. the constraints and invariants;
5. uncertainty, dynamics, and other agents;
6. evidence, computation, and validation; and
7. values, stakeholders, governance, and revision.

An objective function belongs to a model. To interpret an optimization result, we also need to understand how that model represents the system and its environment.

Reference policy

The bibliography favors primary papers, standard textbooks, standards bodies, regulators, and authoritative institutional publications. Appendix D records the scope of the earlier audit and the targeted primary-source review for Version 34, dated 11 September 2026. Structural checks cover citation navigation and bibliography coverage. A verified identifier establishes the identity of a work; its use still requires interpretation of the source and its assumptions.

A note on equations

Each displayed equation is introduced and interpreted in the surrounding text. Symbols are local unless a chapter explicitly carries them forward. Readers can follow the conceptual explanations without deriving every formula. The equations also provide compact specifications for readers who wish to examine the mathematics.

Preface: Optimization as a shared language

We call a candidate *optimal* when, among the specified possibilities and under the stated conditions, no other candidate is better according to the chosen criterion. This definition leaves several questions to examine. We need to identify the alternatives, the criterion and its owner, the available information, the time horizon, the cost of search, and the risks excluded from the model.

In mathematical optimization, the omissions are made explicit. A decision vector x is chosen from a feasible set $\mathcal{X}$ to minimize a function f:

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x). \tag{1}$$

We use this formulation throughout the book. However, it does not specify who chose f, how $\mathcal{X}$ was constructed, whether measurements are reliable, whether other people respond strategically, or whether the problem changes while it is being solved. Those questions connect optimization to statistics, control, economics, ethics, politics, design, and interpretation.

Category theory provides another way to examine these connections. For example, programs and transformations, or narratives and revisions, can be considered objects and morphisms. We can investigate functors into a common category of decision problems, provided that the proposed mappings preserve identities and composition. Such functors are not necessarily equivalences. A cost in one discipline may correspond only approximately to a cost in another. Also, a transformation may preserve a mathematical relationship while changing an object's meaning. We therefore need to describe the information lost during translation. The categorical terminology follows Mac Lane (1998) and Spivak (2014).

We therefore consider optimization as the explicit use of preferences under constraints. Before solving a mathematical program, we need to decide whether the problem is appropriate for formalization, identify what the model omits, and select a method suited to its structure. The resulting decision process should remain open to correction.

How to read the book

For a **general introduction**, read Chapters 1–3, 17, and 20, followed by selected examples from Chapters 18–19. For **mathematical foundations**, begin with Chapters 1–7. **Technical systems** are covered in Chapters 1–3, 5, 7–11, and 17–20; **medicine and society** in Chapters 1–3, 6, 12–14, and 17–20; and **creative work** in Chapters 1–3, 6–7, 14–16, and 17–20. For **category theory**, read Chapters 1–3, 7, 10, 14, and 18–20. Appendix F provides a **mathematics-first route** through Units 1–7, 8–22, 23–35, and 36–42.

Each chapter ends with a *translation lens* and a field checklist. These identify relationships, assumptions, and conditions to examine when applying a method in another discipline.

Contents

List of figures

List of tables

Bird’s-eye view

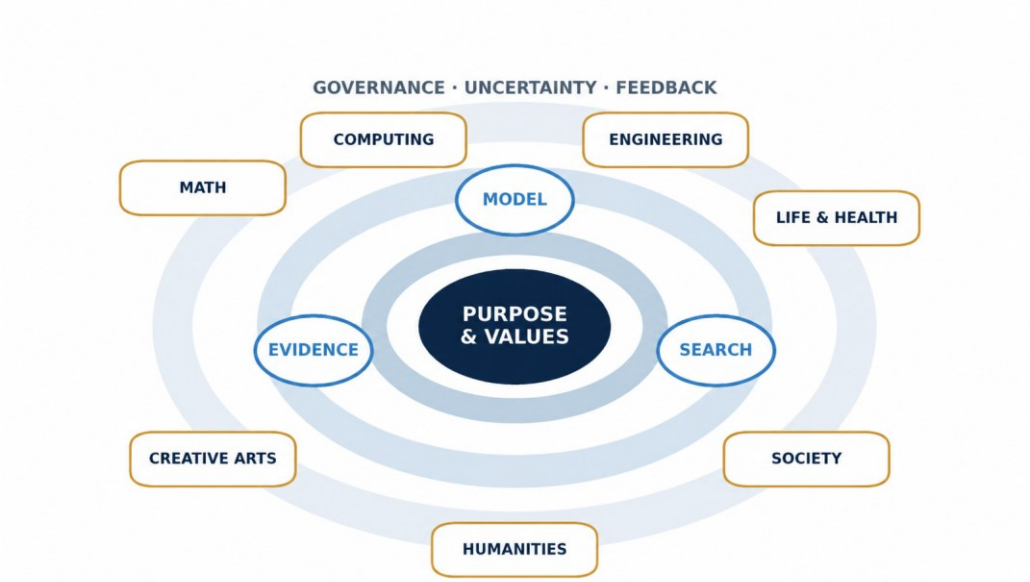


Figure 1. Bird’s-eye view: purpose and values at center; model, evidence, search, and seven domains around them; governance, uncertainty, and feedback spanning the map.

In Figure 1, purpose and values occupy the center. The surrounding domain examples use the same general relationships between models, evidence, search, governance, and feedback, although their objects and requirements differ.

We can consider optimization as a cycle. First, we interpret a situation as a decision problem and identify variables, possible actions, objectives, constraints, uncertainties, affected parties, and a time horizon. Search produces candidate solutions, and a selection rule chooses or recommends one. Implementation then changes the system and produces new evidence. It may also change the preferences or constraints used in the original formulation. Evaluation connects these observations to the next decision.

The full cycle can be written schematically as

$$W_t \xrightarrow{M} P_t \xrightarrow{A} S_t \xrightarrow{I} W_{t+1} \xrightarrow{E} \widehat{W}_{t+1}, \tag{2}$$

where W_t is the world at time t, M is modeling, P_t is the formalized problem, A is an algorithm or deliberative procedure, S_t is a selected solution, I is implementation, and E is observation or evaluation. The next problem is not necessarily the old problem with new data: implementation may change incentives, institutions, technologies, populations, meanings, and available actions.

The seven questions

We can use the following seven questions to examine optimization in a particular discipline.

1. What can change?

These are the decision variables. They may be continuous quantities such as pressure, dosage, or portfolio weights; discrete choices such as routes, treatments, words, or circuit placements; functions such as a control policy; distributions such as a transport plan; or structured artifacts such as programs, buildings, melodies, and institutions.

2. What counts as admissible?

The feasible set represents physical laws, budgets, safety margins, logical consistency, contracts, rights, style rules, manufacturability, deadlines, and other invariants. These constraints determine which candidates can be considered. In creative work, they also help define a recognizable form.

3. What is better?

A scalar objective assigns comparable numerical values to alternatives. To use it for different concerns, we first need a rule for combining those concerns. Sometimes we cannot justify such a rule. We can then use multiple objectives, partial orders, lexicographic priorities, thresholds, vetoes, or deliberation. For example, we may look for nondominated alternatives, an option acceptable to all participants, or a design that preserves useful future possibilities.

4. What is unknown or changing?

Uncertainty may concern parameters, observations, models, other agents, future conditions, or the objective itself. Robust, stochastic, Bayesian, adaptive, and distributionally robust optimization answer different versions of this question. Dynamics add state and path dependence; feedback makes the optimizer part of the system being optimized.

5. How will candidates be generated and compared?

Candidate generation may use proof, calculus, enumeration, relaxation, decomposition, simulation, heuristics, human judgment, evolutionary variation, negotiation, or hybrid methods. We also need to include the computational cost. An exact answer may be unusable if it arrives after the decision deadline.

6. How will success be tested?

We test a candidate using evidence appropriate to the claim. This may include held-out data, sensitivity analysis, optimality certificates, simulation, experiments, stress tests, safety cases, audits, replication, qualitative interpretation, and monitoring after deployment.

7. Who defines, bears, and may revise the objective?

Here we consider who defines the objective and who can change it. The people who receive a benefit may differ from those who bear a cost, and participants may change their behavior in response to a metric. Rights can also restrict the trade-offs available to the optimizer. Some values therefore require hard constraints or explicit decision rights. We also need a procedure for appeals, conditions for stopping, and a way to revise the formulation.

The master form

A broad mathematical template is

$$\min_{x,\pi} \rho(F(x,\pi,\xi);\, \theta) \tag{3}$$

subject to

$$g_i(x,\pi,\xi) \le 0, \qquad i = 1,\dots,m, \tag{4}$$

$$h_j(x,\pi,\xi) = 0, \qquad j = 1,\dots,p, \tag{5}$$

$$x \in \mathcal{X}, \qquad \pi \in \Pi, \qquad R_k(x,\pi) \ge r_k. \tag{6}$$

Here x is a design or allocation, π is a policy, ξ represents uncertainty, F is a vector of consequences, ρ aggregates or orders those consequences using parameters θ, g_i and h_j encode inequality and equality constraints, and $R_k \ge r_k$ represents rights, resilience, reliability, or other non-negotiable requirements, indexed by $k = 1,\dots,q$. The count q is local to these requirements, distinct from the dual function in Equation (23). Different disciplines instantiate the same form differently; some reject the aggregation ρ and preserve a set of incomparable alternatives.

Domain map

We group the disciplines into seven families. Their boundaries overlap, since a discipline can share methods and problems with several others. The six parts of the book specify the reading order, whereas the seven families provide a classification of the domains.

Table 1. Seven domain families and their characteristic optimization objects.

Family	Typical variables	Typical objectives	Characteristic difficulty
Mathematics	vectors, sets, measures, graphs, functions	optimum, bound, convergence, proof	existence, geometry, complexity
Computing	representations, programs, schedules, policies	correctness, runtime, memory, service	scale, discrete structure, feedback

Family	Typical variables	Typical objectives	Characteristic difficulty
Engineering	geometry, materials, controls, processes	performance, cost, safety, energy	coupled physics and manufacturability
Life and health	interventions, allocations, pathways, protocols	health, survival, function, discovery	heterogeneity, uncertainty, ethics
Economy and society	prices, rules, matches, budgets, policies	welfare, stability, equity, growth	strategic response and contested values
Humanities	corpora, classifications, narratives, arguments	coherence, insight, preservation, reach	meaning resists scalarization
Creative arts	form, sequence, material, gesture, style	expression, novelty, fit, experience	objectives emerge during making

Table 1 is a translation aid: similar mathematical forms sit beside different variables, objectives, and characteristic difficulties.

We can use a method in several disciplines without assuming that their objects have the same meaning. For example, a Pareto front in aircraft design can be compared with a set of editorial alternatives only after their orderings and constraints have been specified. In this analogy, we seek a morphism that preserves specified structure; we do not assume an isomorphism between the disciplines.

The optimization lifecycle

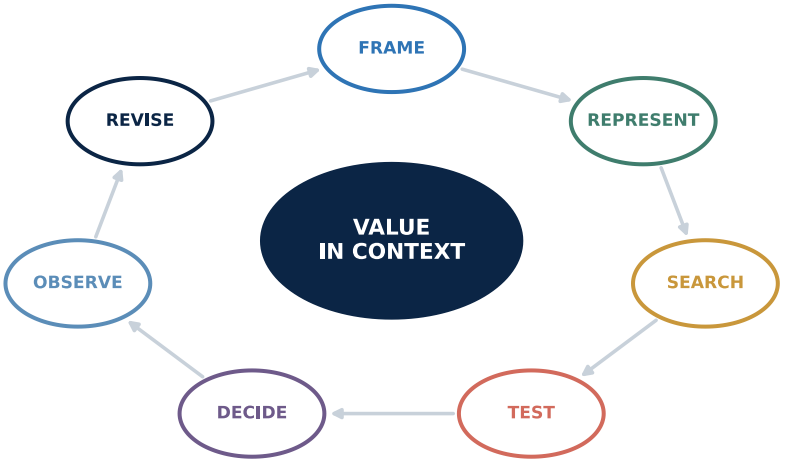


Figure 2. Seven-stage optimization lifecycle—frame, represent, search, test, decide, observe, revise—around value in context.

Figure 2 illustrates a control loop. Observations can lead us to revise the original formulation, and decisions about authority and responsibility apply at every stage.

The lifecycle repeats as new observations become available. For decisions with serious consequences, we need records that can be inspected at each stage. These include the problem statement, data sheet, objective specification, constraint register, solver configuration, evidence of optimality or approximation, sensitivity report, reasons for the decision, implementation plan, monitoring metrics, and conditions for revision. Such records allow us to reconstruct an earlier optimization process.

Ten recurring tensions

Across disciplines, ten tensions recur: **single objective versus plural values; local improvement versus global performance; short versus long horizon; efficiency versus resilience; average outcome versus tail risk; precision versus robustness; central coordination versus distributed autonomy; exploitation versus exploration; measurability versus meaning;** and **optimization power versus accountability.**

The book develops tools for navigating these tensions rather than declaring one side universally correct.

Part I · The Grammar of Optimization

Chapter 1 · Purpose, alternatives, and the grammar of optimization

Purpose, alternatives, and decision systems

Optimization is disciplined comparison among alternatives. We can begin with a set of candidates $\mathcal{X}$ and a rule for preferring one candidate to another. If this rule is represented by a real-valued function f, minimization means finding the following set:

$$\arg\min_{x\in\mathcal{X}} f(x) = \{x \in \mathcal{X} : f(x) \le f(y) \text{ for all } y \in \mathcal{X}\}. \tag{7}$$

The notation distinguishes the *optimal value* $f^* = \inf_{x\in\mathcal{X}} f(x)$ from the *optimizer* x^*. The distinction matters: a best value may be approached but never attained; many different candidates may attain it; or the infimum may be $-\infty$. Existence, uniqueness, and computability are separate questions.

In practice, optimization has at least five layers.

1. **Ontological layer.** What entities and actions exist in the model?
2. **Evaluative layer.** What counts as better, and for whom?
3. **Mathematical layer.** What variables, objectives, constraints, distributions, and dynamics represent the situation?
4. **Computational layer.** What algorithm can produce a useful answer with available time, data, and hardware?
5. **Institutional layer.** Who authorizes the decision, bears its effects, monitors it, and may appeal or revise it?

Mathematical optimization concentrates on the middle layers. Its application depends on all five. An optimality certificate establishes a result for the stated mathematical problem; it does not establish that this problem represents the intended decision.

Improvement, optimum, and satisficing

Improvement and optimality describe different comparisons. When we claim an improvement, we compare a candidate with a reference candidate. Global optimality requires comparison with every feasible candidate, whereas local optimality concerns a neighborhood. Therefore, an intervention can improve the current situation without being the best available intervention. Also, convergence of a solver does not by itself show that the resulting design is globally optimal.

Exact optimization is not always the rational target. Search has a cost. Let $c(a,t)$ be the cost of running algorithm a for time t, and let $L(x)$ be the loss of the returned decision. The operational problem is closer to

$$\min_{a,t} \mathbb{E}\big[L\big(x_{a,t}\big)\big] + c(a,t), \tag{8}$$

than to minimizing L alone. Approximation algorithms, anytime methods, heuristics, and satisficing can therefore be principled responses to limited information and computation. Simon's account of bounded rationality considers decision procedures under these limitations, rather than treating them as defective copies of omniscient maximization (Simon 1955, 1956).

Optimization as compression

We can consider an objective function as a form of compression. It reduces observations and values to a comparison, retaining some information and discarding the rest. Weighted sums, scores, rankings, utility functions, and dashboards are examples of such lossy representations. We need to identify what each representation omits. Multiobjective methods retain trade-offs through a Pareto set, robust methods retain uncertainty sets, and distributional methods retain probability information. Qualitative review can preserve meanings that a numerical encoding does not express.

Optimization as search and proof

We can distinguish methods that search for candidates from methods that establish a guarantee. Enumeration can prove optimality by examining every feasible candidate. Convex duality can supply a lower bound equal to an attained objective value. Branch-and-bound combines candidate search with bounds. Metaheuristics can search irregular objective functions, but they usually do not prove global optimality. We therefore describe a result according to the evidence available: *best found*, *locally stationary*, *ε-optimal*, *globally optimal under the model*, or *preferred after deliberation*.

The compact expression $x^* \in \arg\min_{x \in \mathcal{X}} f(x)$ specifies an optimization problem, but further information is needed for a decision. It omits the origin of the alternatives, the authority to act, the reliability of observations, the cost of implementation, and the rule for reopening the problem. We can therefore describe an optimization project as a tuple of related artifacts: a purpose statement, a decision space, a consequence model, an ordering or deliberative procedure, a search method, an evidence plan, an owner, and a revision rule. In the category-theoretic analogy, the mathematical program participates in a larger system of objects and transformations.

The distinction can be seen in a simple operational example. Suppose a service team must choose an alert threshold τ. Lowering τ detects more incidents but sends more false alarms; raising it reduces workload but misses some failures. The candidate set is not merely the real line. Operationally admissible thresholds depend on sensor calibration, staffing, escalation time, user impact, and the authority of an on-call engineer to suppress alerts. The consequence of τ is a vector containing detection probability, false-alert rate, response delay, interruption cost, and residual risk. A scalar score may be useful for search, yet the final decision may impose a hard miss-rate limit and ask engineers to choose among the remaining Pareto-efficient thresholds.

We can distinguish four comparisons in this example:

1. **Technical comparison:** which threshold performs better on representative data?
2. **Operational comparison:** which threshold is sustainable under actual staffing and incident load?
3. **Risk comparison:** which threshold avoids intolerable tail outcomes?
4. **Institutional comparison:** which threshold is authorized, contestable, and reversible?

Combining these comparisons too early can conceal their differences. Keeping them separate without a selection procedure can also prevent a decision. One possible sequence is to preserve the consequence vector, eliminate inadmissible options, apply an explicit preference rule, and record the judgment used to select among near-equivalent candidates.

Different projects should deliberately choose different output types.

- **A point** is appropriate when implementation needs one setting and uncertainty is small enough.
- **A set** is appropriate when several alternatives are nondominated or evidence cannot distinguish them.
- **A policy** is appropriate when future observations should change action.
- **A ranking** is appropriate when a human process allocates scarce review, provided the ranking is not silently converted into entitlement.
- **A certificate** is appropriate when the claim is feasibility, a bound, or a verified invariant.
- **A counterexample** is appropriate when search reveals that a proposed requirement set is inconsistent.
- **A set of alternatives for discussion** is appropriate when the main uncertainty concerns values or authority rather than consequences.

The output type also limits its authorized use. For example, a sortable score from a screening model may support a review queue. It does not by itself authorize an automatic decision. We need to specify which actions each output permits.

Objectives, constraints, values, and authority

Before writing equations, we choose a system boundary. Suppose a hospital scheduling model includes operating-room utilization but excludes clinician fatigue and downstream bed congestion. Its selected schedule may then improve utilization while worsening the omitted outcomes. Similar boundary choices arise when a compiler reduces execution time but increases energy or binary size, or when a recommender increases engagement but narrows discovery. The boundary determines which consequences the model represents.

From consequences to objectives

Suppose a decision x produces a consequence vector

$$F(x) = \big(f_1(x), f_2(x), \dots, f_k(x)\big). \tag{9}$$

A scalar objective $J(x) = \rho\big(F(x)\big)$ requires an aggregation rule ρ. The familiar weighted sum

$$J(x) = \sum_{i=1}^{k} w_i\, f_i(x), \qquad w_i \geq 0, \tag{10}$$

requires us to specify units and weights. The weights express trade-offs, and aggregation may omit nonlinear effects or thresholds. Also, a weighted sum cannot recover every Pareto-efficient point when the attainable objective set is nonconvex. Other approaches include lexicographic priority, goal programming, reference-point methods, outranking, constrained optimization, and deliberative selection from a Pareto set.

Values can occupy different formal roles.

- A **target** invites maximization or minimization.
- A **constraint** establishes an admissibility boundary.
- A **penalty** permits violation at a price.
- A **veto** removes options regardless of other gains.
- A **right** may restrict which trade-offs are legitimate at all.
- A **process requirement** specifies participation, explanation, or review rather than an outcome score.

Treating every value as a commensurable objective is a modeling choice, not a mathematical necessity. Rawlsian priority, Sen's capability approach, and social-choice theory show why aggregation can conflict with plurality, liberty, and interpersonal incomparability (Rawls 1971; Sen 1985; Arrow 1951).

Why value types are not merely large weights

One possible objection is that these distinctions can be represented by a sufficiently general preference ordering. If a right outranks ordinary welfare, we can give it lexicographic priority. If a violation is instead treated as a sufficiently serious cost, we can attach a large penalty. These are different assumptions about what may be exchanged. Targets, constraints, vetoes, rights, and process requirements can be reduced to one ordering only when that ordering preserves their substantive and procedural conditions.

That reduction is not automatic. Let $v(x) \geq 0$ measure violation and M be finite. Without a proved bound or an exact-penalty theorem, the penalized and constrained problems need not select the same decision:

$$\arg\min_{x \in \mathcal{X}} f(x) + Mv(x) \not\equiv \arg\min_{x \in \mathcal{X}, v(x)=0} f(x) \quad \text{(10a)}$$

For any finite M, the optimizer may accept a violation if the modeled improvement in f exceeds the penalty. Increasing M can introduce ill-conditioning, and we still need to justify the chosen value. Exact-penalty results provide conditions under which a penalty recovers a constrained solution. We cannot assume that these conditions hold whenever we replace a constraint with a weight (Nocedal and Wright 2006).

A penalty assigns a cost to violation, which the optimizer can compare with other gains. A hard constraint excludes inadmissible options. A veto identifies an actor or rule that can block an option, while a right restricts the exchanges that may be offered. A process requirement

specifies how the decision becomes legitimate. These roles remain distinct even when their numerical encodings produce the same ranking on one dataset.

The type of a requirement also determines who can change it and what evidence is needed. For example, a modeling team may adjust a performance weight. Changing a legal duty, a safety case, a patient's consent, or a community's jurisdiction requires a different procedure. Two representations may produce the same ordering on current data while remaining different in this respect. Their revision may involve different people, evidence, notice, and appeal rights.

We can use the reduction when the same authority controls the ordering, the relevant scales are bounded or exactness conditions are proved, the lexicographic and tie-breaking rules are explicit, and the encoding preserves procedural protections. Otherwise, the interface should retain the value type. This allows us to distinguish admissibility, substitution, authority, and revision.

Constraints create form

Constraints also determine form. A sonnet, a bridge, a chip floorplan, and a clinical protocol each have characteristic limits. Equality constraints can express manifolds or conservation laws. Inequalities can express resource limits and safety margins. Logical constraints describe compatibility, chance constraints limit violation probability, and temporal constraints specify order and deadlines.

For a differentiable program

$$\min_x f(x) \quad \text{subject to} \quad g_i(x) \le 0,\ h_j(x) = 0, \tag{11}$$

the Lagrangian is

$$\mathcal{L}(x, \lambda, \nu) = f(x) + \sum_{i=1}^{m} \lambda_i\, g_i(x) + \sum_{j=1}^{p} \nu_j\, h_j(x). \tag{12}$$

Under regularity conditions, the multipliers λ_i and ν_j describe local sensitivity to constraints. Economists often call them shadow prices, and engineers may interpret them as the marginal value or burden of a constraint. This mathematical interpretation does not establish that the corresponding quantity can legitimately be bought or sold.

The feasible set is socially produced

In a mathematical problem, $\mathcal{X}$ is often given. In an institution, the feasible set depends on laws, standards, infrastructure, procurement rules, professional norms, accessibility requirements, and political power. These conditions can change. For example, a new material, diagnostic test, public process, or legal rule may introduce alternatives that were absent from the original formulation. In such cases, changing the feasible set may be more useful than improving the solver for the old set.

Revising the formulation

We can record the formulation in a *framing register* containing the system boundary, stakeholders, excluded effects, time horizon, baseline, counterfactual, and reasons for choosing each objective and constraint. If evidence contradicts an assumption, we may need to change the formulation. In the category-theoretic analogy, reframing is a morphism between problem objects that preserves some structures and changes others.

A metric represents selected aspects of a purpose. For example, utilization may represent part of health-service performance, test results part of learning, and citation counts part of scholarly activity. Engagement and predicted attention provide similarly partial representations of friendship and art. Distinct situations can receive the same score. We can describe these indistinguishable situations by analogy with the kernel of a measurement morphism. Optimization may exploit the differences that the score fails to represent.

Practical reasoning therefore precedes technical reasoning. Aristotle distinguishes deliberation about means from the formation and cultivation of ends; contemporary capability and justice traditions similarly resist treating every good as a commensurable quantity (Aristotle 2002; Sen 1985; Rawls 1971; Nussbaum 2011). We therefore ask whether optimization is appropriate at all. It may be appropriate to optimize the scheduling of a care service while refusing to rank the intrinsic worth of its patients. It may be appropriate to optimize acoustic treatment while refusing to optimize a composition to an engagement model. Refusing a proposed optimization can therefore express a reasoned boundary on the transformations we consider legitimate.

We can describe responsibility using a table. Its rows represent decisions, and its columns represent the actions of proposing, modeling, validating, approving, executing, overriding, investigating, appealing, and retiring. An empty assignment shows that responsibility has not been specified. If one person or group performs every action, we need to examine whether independent review is possible.

Authority should match response time. An operator can stop an unsafe action immediately, while permanent objective changes require broader review. A user may correct personal data without owning the global model. A community body may hold consent or veto rights that are not reducible to advisory preference.

The responsibility map should be available at a level of detail appropriate to the decision. It identifies the human and institutional actions behind a system output and allows escalation procedures to be tested before an incident.

Search cost, baselines, and levels of intervention

Optimization also consumes resources. For example, modeling can delay action, data collection can burden participants, and simulation requires energy. Solver tuning uses specialist time, and exact search may continue beyond the decision deadline. We can include these costs in a more complete operational objective:

$$\min_{a,t} \ \mathbb{E}_{\xi}\left[L\left(x_{a,t};\xi\right)\right] + C_{\text{search}}(a,t) + C_{\text{delay}}(t) + C_{\text{change}}\left(x_{a,t},x_0\right) \tag{13}$$

where a is a method, t is the effort or stopping time, x_0 is the current practice, and ξ represents uncertain consequences. The returned decision $x_{a,t}$ depends on the method and its stopping time; the expectation also covers any randomization in the method. This formulation extends the simpler search-cost model in Equation (8) by adding delay and change costs. It explains why the most sophisticated solver is not automatically the best intervention. A fast transparent approximation can dominate an exact but delayed answer; an existing safe practice can dominate a fragile improvement whose transition cost is ignored. We can interpret this as an explicit model of bounded rationality (Simon 1955, 1956).

The consequences of the decision help determine the evidence required. For frequent decisions with limited consequences, measured improvement may be sufficient. An irreversible or safety-critical design may also require feasibility proofs, worst-case bounds, independent verification, and conservative margins. We need to specify these requirements before choosing the method. Finding a candidate with a heuristic does not establish the guarantee needed for its use.

The existing practice provides a baseline with both known defects and accumulated experience. We need to record its performance, workarounds, social meaning, and recovery procedures. Comparison with a proposed solution should also include the transition between them. This may require migration, simultaneous operation of both systems, training, temporary risk, or loss of tacit knowledge. We also need to consider whether the change can be reversed.

We can use a transparent baseline to examine whether a more complex method adds value. If the complex method does not improve forward validation, its additional complexity does not justify deployment. If the difference is smaller than measurement uncertainty or implementation cost, we may retain the existing rule and improve the evidence. The candidate set should therefore include leaving the current practice unchanged.

We can distinguish three levels of optimization. Operational optimization selects an action under an accepted rule, such as a vehicle route, controller setting, or shift assignment. Design optimization changes the artifact or process that makes these actions possible, such as fleet composition, sensor layout, or workflow. Constitutional optimization concerns who can decide, which values can enter the formulation, and how a rule can be challenged. Optimization supports the constitutional process; it does not replace that process. Identifying the level helps clarify disagreements about the proposed variables.

These levels depend on each other. An operational model may show that a design permits no safe action. A different design may make an institutional requirement feasible, while a constitutional restriction may exclude an otherwise efficient design. However, authority at one level does not automatically extend to another. For example, a dispatcher may reroute a vehicle without being authorized to redefine the right to service. Similarly, tuning a controller

does not authorize an engineer to waive a safety duty, and approval by a board does not validate a measurement.

Write the decision level beside every variable. If a model silently promotes a fixed policy parameter into a decision variable, identify who has authority to change it. If it treats a political or ethical choice as data, restore it to deliberation. If it freezes a design that dominates operational performance, widen the feasible set.

The levels may also be revised at different intervals. For example, operations may change each minute, design each quarter, and constitutional requirements each year or after an incident. A faster process should refer changes in a slower commitment to the appropriate authority. The slower process also needs observations from the faster one, including overrides and demand that could not be served.

The hierarchy helps specify which layer can change, which conditions remain fixed, and which observations would justify changing the boundary.

An example of a decision charter

We can now return to the alert-threshold example. The objective of minimizing false alarms and missed incidents leaves four choices implicit: the definition of an incident, whose interruption is counted as a cost, the weighting of severe events, and who accepts residual risk. A decision charter makes these choices explicit.

The purpose is to detect service degradation early enough to limit user harm while keeping the on-call workload manageable. We choose the threshold together with the persistence window, routing, suppression, escalation, and review interval. The system boundary includes the sensor, detection logic, paging service, engineer attention, escalation, user impact, remediation, and learning after the event. Reliability leadership approves operational policy; security and safety owners may require separate alarms. On-call engineers can suppress a faulty signal, and affected product teams can challenge repeated burden. We use historical replay, injected incidents, calibration, workload observation, and staged deployment as evidence. Rollback conditions cover severe misses, unmanageable workload, and degraded response. Changes in architecture, workload, staffing, or sensor behavior require us to reconsider the formulation.

With this description, we can identify additional variables. Persistence and grouping may reduce nuisance alarms without increasing the signal threshold. Better sensor calibration may improve detection and reduce workload. Staffing and automated remediation may also change which policies are feasible. The original threshold problem therefore becomes a co-design problem involving several related components.

We also need to reconsider validation. A random split of historical alerts may place related incidents in both training and test data, or retain conditions that no longer apply. Replay should therefore hold out later periods, entire incident types, and major architecture changes. Injected incidents test whether faults are detected and lead to the expected pages. Shadow

operation shows how often the candidate policy would have paged an engineer. Interviews can then help determine whether those pages are understandable and actionable.

The final record can make a conditional claim: policy π met a hard severe-event detection requirement, reduced median nonactionable load against baseline π_0, remained feasible under staffing stress, and was chosen over a slightly quieter policy because independent review judged its tail miss risk unacceptable. Such a record supports the decision while retaining uncertainty about preferences.

Field checklist

- Name the decision maker, affected parties, and those with veto or appeal rights.
- Distinguish candidate generation, candidate comparison, and authorization to act.
- State whether the output is a point, a set, a policy, a ranking, a certificate, a counterexample, or a set of alternatives for discussion.
- Price the costs of data, modeling, search, delay, transition, and monitoring.
- Match the claimed guarantee to the evidence actually produced.
- Record what the objective compresses away and which consequences remain non-negotiable.
- Define a stopping condition for search and a reopening condition for the formulation.

Translation lens. Across disciplines, we begin with alternatives and a rule for comparison. When transferring a formulation, we also need to examine its purpose, authority, evidence, and revision procedure. These determine whether the relation between a choice and its intended effects remains valid.

Chapter 2 · Problem structure, taxonomy, and historical foundations

Taxonomy and structural fingerprint

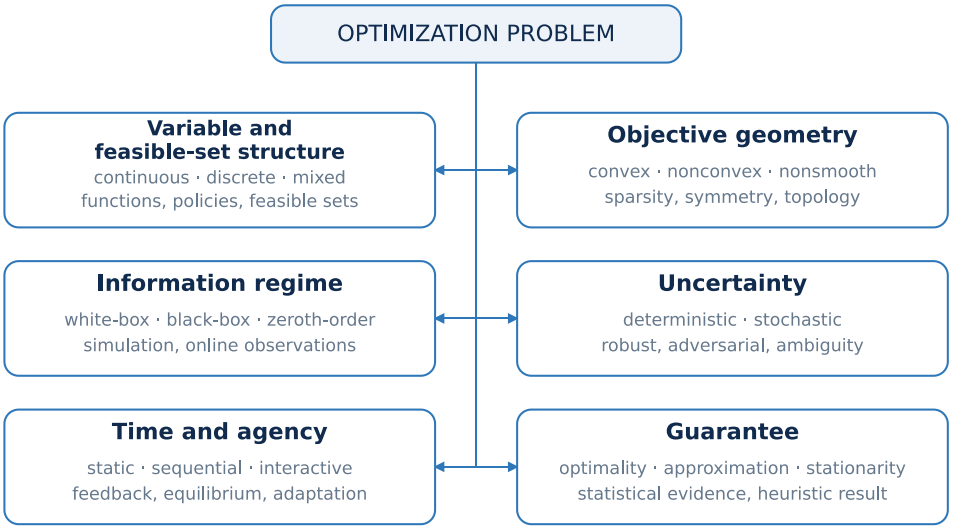


Figure 3. Structural fingerprint across six axes: variable and feasible-set structure, objective geometry, information regime, uncertainty, time and agency, and guarantee.

Figure 3 shows properties that help us select a method. Two problems with the same objective may require different algorithms because the available variables, information, uncertainty, or required guarantees differ.

Optimization problems can be classified along independent axes. The classification matters because algorithm choice should follow structure.

Variable and feasible-set structure

Variables may be continuous, integer, binary, categorical, graph-valued, sequence-valued, function-valued, or mixed. A continuous relaxation may provide bounds for a discrete problem. A topology or architecture search may require variable-dimensional representations. Function-space optimization leads to the calculus of variations and optimal control.

Objective geometry

A convex objective over a convex feasible set has no non-global local minima. Strong convexity adds uniqueness and quantitative curvature. Smoothness enables gradients and Hessians; nonsmoothness calls for subgradients, proximal operators, bundle methods, or smoothing. Nonconvexity can arise from physics, discrete structure, neural networks, equilibrium constraints, or composition.

Information regime

In white-box optimization, we can inspect useful algebraic or derivative structure. A black-box interface provides responses to evaluations without exposing the internal calculation needed by the optimizer. Zeroth-order methods may estimate directions from these responses. Simulation optimization may involve deterministic or noisy evaluations, often at substantial cost. Online optimization receives losses sequentially. These classifications describe different properties: some derivative-free methods have convergence theory, and some uses of gradient methods in nonconvex problems provide only empirical evidence. Chapter 5 develops the black-box variants.

Uncertainty

- **Deterministic:** parameters are treated as known.
- **Stochastic:** a probability model supports expectations or risk measures.
- **Robust:** decisions are protected against an uncertainty set.
- **Distributionally robust:** the distribution belongs to an ambiguity set.
- **Adaptive:** information arrives and decisions may change.
- **Adversarial:** another process selects difficult inputs.

Time and agency

In a static problem, we make one choice. In a dynamic problem, we choose a trajectory or policy, and in a control problem, an action affects the evolving state. Games include agents that respond strategically. A bilevel problem includes the solution of one optimization problem within another. Mechanism design concerns rules and the strategic responses to them. Distributed optimization divides data, variables, or authority among participants.

Guarantee

An algorithm may provide an exact optimum, a certified bound, an approximation ratio, asymptotic convergence, regret bounds, probabilistic guarantees, empirical superiority, or only a plausible candidate. For NP-hard problems, no polynomial-time exact algorithm is known, and none exists unless $P = NP$. Approximation and problem-specific structure therefore become central (Garey and Johnson 1979; Papadimitriou and Steiglitz 1982; Goemans and Williamson 1995). No-free-lunch results formalize that performance advantages depend on assumptions about the problem class (Wolpert and Macready 1997).

A selection rule

The practical algorithm-selection sequence is:

1. expose variable and constraint types;
2. test convexity, smoothness, separability, sparsity, and symmetry;
3. identify uncertainty and observation cost;
4. state the required guarantee and deadline;
5. select a method family;

6. compare multiple implementations on representative instances; and
7. verify sensitivity to modeling and solver choices.

Before choosing a method, we can construct a structural fingerprint of the problem. The fingerprint is a compact inventory of decision type, feasible-set geometry, objective regularity, information access, uncertainty, temporal structure, agency, scale, and required guarantee. Two problems described with the same domain noun may have different fingerprints: "routing" can mean a deterministic shortest path, a capacitated vehicle-routing MILP, an online dispatch policy, or a robust multi-agent game. Conversely, a chip floorplanner and a hospital bed allocator may share a decomposable assignment structure even though their domain meanings differ.

A useful fingerprint answers the following questions in order.

Variable and feasible-set structure. Are the variables continuous, integer, categorical, graph-valued, sequence-valued, function-valued, or mixed? Does the representation contain symmetries that duplicate the same physical or semantic object?

Objective geometry. Is the feasible set convex, disconnected, manifold-valued, or combinatorial? Are the objectives smooth, Lipschitz, separable, sparse, submodular, piecewise linear, or discontinuous? Which claims are mathematical facts, and which describe observations in a limited region?

Information regime. Can the method query exact or noisy values, derivatives, adjoints, constraints, samples, or only rankings? How expensive is a query? Can evaluations fail or run in parallel?

Uncertainty. Are inputs treated as known, sampled from a probability law, constrained to an uncertainty or ambiguity set, or chosen adversarially? Which assumptions connect the observations to future conditions?

Time and agency. Is the choice one-shot, repeated, sequential, or closed-loop? Do other agents adapt strategically? Does the action change the data-generating process or future feasible set?

Guarantee. Does the decision require a feasible candidate, a local stationary point, a certified bound, an approximation ratio, a regret guarantee, a calibrated predictive distribution, or alternatives that people can review and justify?

The fingerprint helps us choose a method from the problem structure. If we begin with a familiar solver and force the problem into its representation, we may lose useful information. A mixed-integer program, differentiable objective, or reinforcement-learning environment should be selected because it preserves the relevant structure.

Before accepting an aggregation or surrogate, list invariants that must survive. Common invariants include conservation of material, nonnegative inventory, temporal precedence, legal eligibility, connectedness, mass or energy balance, probability normalization, anonymity, and an accessibility guarantee. Then test the transformation.

We can examine this loss in several examples. Aggregating hourly demand into a daily total preserves quantity but can hide peak capacity requirements. Euclidean distance omits network features such as bridges, slopes, borders, and accessibility. An integrality relaxation can retain a bound while producing a nonimplementable solution. A text embedding represents selected statistical neighborhoods; exact citations, chronology, and negation may need separate representations.

We can also express this check in categorical terms. A proposed morphism is suitable for the task only if the diagrams describing the required invariants commute within the stated error. Otherwise, we need to add information to the representation, restrict the claim, or reject the transformation.

History as accumulated structure

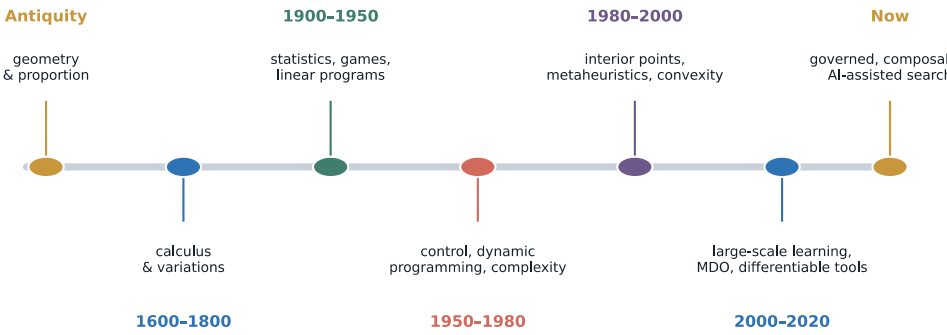


Figure 4. A timeline from ancient reasoning about the good, through calculus, variational principles, industrial operations research, convexity and complexity, to data-driven, compositional, and governed optimization.

Figure 4 shows developments in representation, mathematical guarantees, and computation. These developments follow several interacting histories.

The history of optimization includes contributions from ethics, geometry, mechanics, economics, computation, statistics, and administration. We can follow their connections without treating them as one sequence of methods.

Ancient ethics considered what constitutes a good life and how practical judgment selects an action. Aristotle's account of deliberation also provides a way to examine the choice of ends, although it is not a formulation of mathematical programming (Aristotle 2002). Early geometric extremum problems and later differential calculus allowed maxima and minima to be studied mathematically. Variational principles in mechanics extended such questions to

curves and functions. Economic theories formalized utility, exchange, efficiency, and equilibrium, and raised further questions about aggregation and welfare.

During the twentieth century, optimization developed through practical problems in production, transportation, military logistics, and resource allocation. Dantzig's simplex method, Kantorovich's allocation work, and Koopmans's activity analysis connected mathematical formulations with these applications (Dantzig 1963; Kantorovich 1960; Koopmans 1951). Dynamic programming described sequential decisions using a principle of optimality (Bellman 1957). Optimal control connected variational reasoning with feedback and engineering (Pontryagin et al. 1962). The Karush-Kuhn-Tucker conditions provided a general framework for constrained nonlinear optimization (Kuhn and Tucker 1951).

Computational complexity changed the question from "Does an optimum exist?" to "What resources are required to find or approximate it?" Khachiyan's ellipsoid method established polynomial-time solvability of linear programming with rational data (Khachiyan 1979). Karmarkar introduced a different polynomial-time approach that stimulated the development of interior-point methods (Karmarkar 1984). Combinatorial optimization connected polyhedral geometry, graphs, approximation, and integer programming (Schrijver 2003).

Uncertainty produced stochastic programming, robust optimization, Bayesian decision methods, and risk measures. Multiobjective optimization replaced the single best point with a frontier of trade-offs (Miettinen 1998). Evolutionary, swarm, annealing, and surrogate-based methods expanded the repertoire for irregular or expensive landscapes (Kirkpatrick, Gelatt, and Vecchi 1983; Kennedy and Eberhart 1995; Jones, Schonlau, and Welch 1998).

Machine learning introduced many large optimization problems. Stochastic gradients became widely used for their solution (Robbins and Monro 1951; Bottou, Curtis, and Nocedal 2018). Learned models have also been used to select architectures, acquire data, choose policies, design chip layouts, and investigate scientific hypotheses. Rankings, recommendation systems, and automated allocation provide examples in which the chosen objective can change the behavior being measured (Espeland and Sauder 2007; Sculley et al. 2015).

The contemporary phase adds three themes. First, optimization is increasingly compositional: large systems are assembled from interacting subsystems, motivating category-theoretic, distributed, and multidisciplinary approaches (Fong and Spivak 2019; Martins and Lambe 2013). Second, optimization is increasingly governed: safety, fairness, privacy, sustainability, and rights enter formulation and lifecycle management (Tabassi 2023; European Parliament and Council of the European Union 2024). Third, optimization is increasingly reflexive: the system observes and changes itself, while users and institutions adapt in response.

We can often recognize older ideas in a new method, even when its computational setting has changed. Questions about purposes, authority, and judgment also recur when an objective is implemented in software.

We can describe this history in terms of the objects that became available for modeling. Calculus provided local differential information, and variational methods included functions

and trajectories. Linear programming allowed the computation of resource allocations and dual prices. Dynamic programming made explicit the information that a state must retain for subsequent decisions. Complexity theory distinguished the existence of a solution from the resources needed to construct it. Robust and stochastic optimization represented different forms of uncertainty, while multiobjective methods retained partial orderings. Compositional and category-theoretic approaches examine which guarantees remain valid when systems are connected.

A new method does not necessarily replace an earlier one. Simplex, interior-point methods, dynamic programming, decomposition, gradient methods, heuristics, simulation, and deliberation apply to different problem structures. An older method may become useful in another setting when hardware, automatic differentiation, data, or modeling languages change. Historical comparison can also help us recognize a rediscovered idea and examine the conditions under which it previously failed (Dantzig 1963; Bellman 1957; Karmarkar 1984; Schrijver 2003).

Representation, complexity, and equivalent decisions

Consider locating p service facilities to cover demand points. A direct binary formulation can use $y_j \in \{0,1\}$ to indicate whether facility j opens and $x_{ij} \in \{0,1\}$ to assign demand point i to facility j. Let $\mathcal{F}_p$ contain the binary pairs (x, y) satisfying $\sum_{j=1}^{m} x_{ij} = 1$ for every demand point i, $x_{ij} \le y_j$ for every pair i, j, and $\sum_{j=1}^{m} y_j = p$. Here d_i is the demand weight at point i, c_{ij} is the assignment cost, and f_j is the fixed cost of opening facility j. There are m candidate sites and n demand points. The model is then

$$\min_{(x,y)\in\mathcal{F}_p} \left(\sum_{i=1}^{n}\sum_{j=1}^{m} d_i\, c_{ij} x_{ij} + \sum_{j=1}^{m} f_j\, y_j\right). \tag{14}$$

The formulation includes fixed costs, assignment costs, and their dependencies. We still need to define what distance c_{ij} means in the application. It may represent travel time, accessibility, cultural acceptability, network latency, or ecological disruption. Equation (14) can therefore be used for emergency stations, content replicas, warehouses, or clinics, but each application requires its own interpretation and validation evidence.

Representation can also change computational difficulty. Aggregating demand may shrink the model but hide vulnerable groups. Relaxing integrality gives a lower bound but may create impossible fractional facilities. Decomposing by geography can accelerate solution but mishandle cross-boundary demand. Symmetry-breaking can remove redundant facility labels without changing physical alternatives. Each transformation should state what it preserves: feasible real decisions, objective bounds, uncertainty, distributional information, or merely average behavior.

Worst-case complexity is indispensable: it prevents claims that an expressive formulation is automatically tractable (Garey and Johnson 1979; Papadimitriou and Steiglitz 1982). Operational choice also depends on instance distribution, horizon, warm starts, and

repetition. A difficult combinatorial problem can be easy on a stable sparse network and hard after one business rule creates symmetry or dense coupling. Conversely, a polynomial method can be unusable if each evaluation invokes a high-fidelity simulator.

We therefore record both formal complexity and empirical scaling. By varying problem size, density, conditioning, uncertainty, and constraint tightness separately, we can identify failure regimes that a median runtime conceals. For a learned or heuristic method, shifted instance families can test its inductive bias. These observations help identify when the method ceases to fit the problem.

An optimization can contain many numerical points representing the same substantive choice. Facilities may be unlabeled, neural units permutable, rotations physically equivalent, and schedules identical up to workers with the same qualification. These symmetries multiply search without adding alternatives. Symmetry-breaking constraints or quotient representations can improve computation if they preserve every real equivalence class.

The reverse problem is identifiability: distinct world mechanisms can produce the same observations. A calibrated simulation may fit parameters that compensate for one another. A causal model may match outcomes under current policy and disagree under intervention. An inverse design may have many geometries with indistinguishable measured performance and different manufacturability.

We can distinguish two questions before search. Which numerical differences represent the same object? Which different mechanisms produce observations that the model treats as identical? The first question concerns quotient representations and symmetry breaking. The second concerns experiments and uncertainty.

For every claimed parameter estimate, report whether the decision depends on the parameter itself or only on a prediction. If two parameter sets lead to the same admissible action across relevant scenarios, identifiability may not be decision-critical. If they lead to different actions under a foreseeable condition, collect discriminating evidence or choose a policy robust to both.

Symmetry and identifiability meet in translation. A musical chord can be considered equivalent under octave or permutation for one analysis and distinct in voicing and instrument for performance. A transport model can identify people by demand class for capacity planning and must distinguish access needs for service design. We therefore specify the purpose when declaring two objects equivalent.

Decompose when components have sparse coupling and can exchange enough information to coordinate. Geographic, temporal, scenario, resource, or functional decomposition can shrink computation and align organizational ownership. It fails when the interface omits a strong global constraint, shared uncertainty, or common-cause failure.

Test decomposition against a centralized model on smaller instances. Compare feasibility and objective, inspect interface residuals, and vary coupling strength. If local solutions oscillate,

expose richer response functions or use coordination penalties. If organizations strategically report local information, the decomposition is also a mechanism and needs incentive analysis.

Benchmark representations, not only solvers. For a manageable set of real instances, implement two formulations that encode the same declared decision. Compare model size, relaxation bound, solve time, numerical behavior, ease of adding domain constraints, feasibility after export, and explanation to reviewers.

We can then vary the domain by adding a fairness floor, correlated failure, a new resource, or a different horizon. A representation that solved the original problem quickly may be difficult to maintain after such changes. This maintenance cost also affects its suitability.

Check semantic equivalence with generated and hand-built examples. Map each solution back to the domain and compare. Exact algebraic reformulations should preserve feasible decisions and values; approximations should satisfy declared error; aggregations should preserve selected invariants. A discrepancy is either a bug or evidence that the two models never represented the same object.

Representation benchmarks provide evidence for formulation choices. Their examples can also be reused when a model is refactored, decomposed, or moved to another solver.

An example of method selection

Suppose a regional blood service needs routes from collection sites to laboratories. We need to account for time limits, uncertain volume, road disruption, temperature control, and vehicle failure. Calling this a routing problem does not yet tell us which method to use. We first examine its structural properties.

Routes and assignments are discrete, whereas departure times and quantities may be continuous. The static problem includes capacity constraints and time windows. Vehicle reuse and the coupling between pickup and delivery prevent us from representing it as a simple shortest-path problem. Travel times and volumes are forecast, temperature conditions are monitored, and some disruptions become known during operation. The plan is repeated and may require immediate repair. Drivers and hospitals respond to operational changes, but we do not model them as strategic opponents. Every dispatched route must be feasible and traceable. Rapid recovery and low risk of extreme delay are more useful here than a global optimality proof.

We can use these properties to combine several methods. A mixed-integer model can plan the fleet and nominal routes. Scenario or robust constraints can represent critical time windows, and an insertion-and-repair heuristic can respond during operation. A separate safety rule can exclude actions that violate validated temperature and handling requirements. A traveling-salesperson formulation based only on distance would omit these requirements. A reinforcement-learning policy would need a suitable simulator and safety architecture, whose construction might cost more than the expected improvement.

Several representations are possible. A time-expanded network expresses time windows explicitly, but its size can grow rapidly. A route-column formulation represents complete feasible routes and permits column generation. A vehicle-indexed MILP is direct, although equivalent vehicle labels may introduce symmetry. An event-based constraint model may handle time and resource conflicts through propagation. We can compare at least two representations using instances that retain realistic disruptions and route structure.

When translating blood logistics to parcel routing, we can retain capacity, time, flow, and recovery relationships. We need to change the interpretation of consequences: a late parcel and a compromised medical product have different effects. Temperature and traceability requirements therefore remain explicit constraints in the blood-logistics model.

Field checklist

- Build the structural fingerprint before selecting software or an algorithm family.
- Separate facts about the mathematical object from observations about sampled instances.
- Identify symmetries, decompositions, conservation laws, sparsity, and natural interfaces.
- Record every relaxation, aggregation, surrogate, and discretization with its preservation claim.
- Test at least one representation different from the first formulation.
- Specify the evidence required for the decision and check whether the solver can provide it.
- Revisit the fingerprint when the boundary, data regime, or implementation cadence changes.

Translation lens. We can transfer a formulation when the relevant structural correspondences are preserved. For example, logistics and computing may share assignment and capacity relationships. Distance, harm, and validation evidence still require definitions for the new domain. A translation should identify what it preserves and what it omits.

Chapter 3 · Measurement, evidence, governance, and the optimization lifecycle

Measurement and the evidence chain

To compare candidates, we need evidence about their consequences. A metric represents only part of this evidence. Its measurements depend on instruments, definitions, sampling processes, and institutions. We can therefore encounter a reliable but irrelevant metric, a relevant but manipulable one, or a measure that is informative at one scale and misleading at another.

The measurement model

Let z be a latent property of interest and let y be the measured proxy:

$$y = m(z, c) + \varepsilon, \tag{15}$$

where m is a measurement process that depends on context c, and ε represents error. Optimizing y changes incentives around m and may change the relationship between y and z. Selection pressure can favor the gap between proxy and purpose: an algorithm may exploit favorable measurement error or omitted consequences, and participants may adapt to the target. The measured result can improve without improving the intended property. These are distinct mechanisms of Goodhart-like failure (Goodhart 1975; Campbell 1976; Muller 2018).

Evidence for solutions

Evidence should answer several distinct questions.

- **Mathematical validity:** Were the objective and constraints solved as claimed?
- **Numerical validity:** Are tolerances, conditioning, discretization, and stochastic error controlled?
- **Model validity:** Do assumptions and parameters represent the relevant system?
- **Comparative validity:** Is the baseline fair, and were alternatives tuned equally?
- **External validity:** Does performance transfer across settings, populations, and time?
- **Causal validity:** Will intervention produce the predicted consequence, rather than merely correlate with it?
- **Normative validity:** Do stakeholders accept the objectives, constraints, and distribution of effects?

Different methods answer different questions about a result. Sensitivity analysis shows how it changes when parameters change, whereas scenario analysis compares possible future conditions. Stress tests look for failures under extreme conditions. Ablation studies examine the contribution of individual components. Where ethical and practical, randomized trials can estimate causal effects. Qualitative inquiry can identify mechanisms and harms that are absent from numerical records.

Governance as control over the formulation

Governance specifies rights and responsibilities throughout the optimization lifecycle. We need to identify who may set the objective, add a constraint, inspect data and assumptions, stop deployment, hear appeals, and respond when the model and the observed system diverge.

A governed optimization system should have:

1. a named owner for the decision and a distinct reviewer;
2. a documented objective and constraint register;
3. traceable data and model versions;
4. thresholds for abstention or human escalation;
5. monitoring for drift, distributional effects, and strategic response;
6. incident and rollback procedures; and
7. scheduled reframing, not only parameter retraining.

Public frameworks increasingly express this lifecycle view. NIST's AI RMF organizes risk work around Govern, Map, Measure, and Manage, while health guidance emphasizes autonomy, safety, transparency, accountability, inclusiveness, and sustainability (Tabassi 2023; World Health Organization 2021). Standards such as ISO/IEC 25010:2023 provide quality models that keep software optimization from collapsing into one performance metric (International Organization for Standardization 2023).

Audit the objective, not only the algorithm

An algorithm audit may examine bias, robustness, or accuracy while leaving the objective unchanged. We also need to ask why the objective was chosen, how it was measured, which other purposes it omits, and what behavior it encourages. For constraints, we ask which interests are protected by a hard limit and which can be traded against other outcomes. Affected people should be able to question these choices.

An optimization result inherits uncertainty from observation, data cleaning, parameter estimation, causal assumptions, model structure, numerical approximation, solver termination, implementation, and behavioral response. Reporting only the final objective value hides these dependencies. We can instead trace the claims along this chain and associate each claim with an appropriate test.

Suppose a city optimizes school-support funding using a predicted "risk of underperformance." The prediction may be calibrated, yet the allocation question is causal: will a particular support package improve learning for a particular school under actual delivery constraints? Historical labels may reflect unequal resources; measured attainment may omit well-being, inclusion, and long-term opportunity; schools may change behavior when the score controls funding. A technically accurate model can therefore support an invalid intervention.

The seven evidence categories above identify the claims we want to support. The following eight checks examine how those claims can fail. Construct and statistical checks refine model validity; operational, distributional, and temporal checks refine model and external validity. Causal, numerical, and normative checks appear in both views. Mathematical and comparative validity still require their separate solution and baseline checks.

1. **Construct validity:** does the measurement represent the intended property?
2. **Statistical validity:** are sampling, dependence, missingness, and uncertainty handled?
3. **Causal validity:** does the proposed action change the outcome as predicted?
4. **Numerical validity:** are discretization, tolerances, and conditioning controlled?
5. **Operational validity:** can the intervention be implemented with actual people and infrastructure?
6. **Distributional validity:** who benefits, who bears errors, and which subgroups are hidden by averages?
7. **Temporal validity:** what drifts, adapts, or becomes obsolete after deployment?
8. **Normative validity:** is the objective legitimate and contestable?

No single method answers all eight. Randomized trials can estimate some causal effects but may have limited transportability. Simulation can expose dynamics but only within its assumptions. Formal verification can prove conformance to a model but not adequacy of the model. Qualitative work can reveal meanings, workarounds, and harms absent from recorded variables. We therefore combine different forms of evidence according to the claims they can support.

Every consequential optimization should produce a dossier that another competent reader can inspect without reconstructing the project from code. The dossier includes:

- decision charter and system boundary;
- stakeholder and rights analysis;
- data provenance and measurement definitions;
- model equations, assumptions, and units;
- solver, version, tolerances, random seeds, and computational budget;
- baseline and alternative formulations;
- feasibility, optimality, approximation, or empirical evidence;
- sensitivity, scenarios, and stress tests;
- distributional and accessibility analysis;
- implementation, monitoring, override, and rollback plan; and
- unresolved disagreements and known unknowns.

The dossier records the state needed to continue the optimization work across teams and over time. Without it, a later user may have the output but lack the assumptions, conditions, and responsibilities required to interpret it.

A decision dossier becomes more useful when its claims are arranged as rows and evidence sources as columns. For a predictive-maintenance policy, rows might include “the sensor

measures the intended condition," "the model is calibrated before failure," "the threshold reduces unplanned downtime," "maintenance capacity can absorb alerts," "the policy does not systematically defer difficult assets," and "the fallback is safe." Columns might include calibration bench tests, historical data, designed inspections, simulation, shadow deployment, operator review, and prospective outcomes.

The amount of data in a row does not establish its claim. For example, historical data cannot establish a safe response to a sensor fault that never occurred in the data. A successful bench test does not establish a maintenance benefit. Operator acceptance does not prove calibration. However, repeated disagreement from operators may indicate an omitted state or a workflow that cannot be followed.

For each cell, we record whether the evidence is direct, indirect, contradictory, absent, or not applicable. We also record the population, conditions, version, uncertainty, and responsible person. This can show that an important claim depends on one indirect study or on a model tested only near the current policy. A reference to a general method should be distinguished from evidence that validates the local implementation.

Post-deployment incidents are unusually informative because they expose a complete chain. Reconstruct the world state, available data, model version, recommendation, interface, human response, system action, and consequence. Then run counterfactual replays with the prior policy, candidate revision, and simple baseline. Ask which differences are supported and which depend on unobserved assumptions.

During replay, we need to preserve the information available at the time. Hindsight can make a rare signal appear obvious while hiding the false alarms that a more sensitive rule would have produced. If the mechanism is uncertain, we retain several plausible causal accounts. The replay then tests whether sensing, representation, authority, and recovery operated as intended.

We can examine near misses and overrides in the same way. If a human correction prevents harm, the safeguard has worked, but the model that produced the recommendation may still require revision. Staff may stop recording overrides if the records are used to penalize them. Records intended for learning should therefore be kept separate from punitive performance measures.

Distribution shift has several types. Covariate shift changes the frequency of observed conditions. Label shift changes outcome prevalence. Concept drift changes the relation between observations and outcome. Policy shift changes data because the decision system acts. Boundary shift changes who or what is included. Normative shift changes the institution's values or legal duties.

These changes require different responses. Reweighting may help with some covariate shifts, but it cannot repair a changed causal mechanism. Recalibration can correct probabilities without justifying a new use of the model. Similarly, additional data may reduce estimation

uncertainty without supplying missing consent. We therefore monitor inputs, residuals, calibration, decisions, outcomes, population, and governance conditions separately.

We need to specify which changes can be made automatically and which require a decision by an authorized person or body. A temperature-forecast parameter may be updated routinely. A threshold controlling access to care should not change automatically when service capacity changes. If the resource constraint makes the threshold infeasible, we need to report the policy conflict and review it.

We can assess reviewer independence through access and authority. A reviewer should be able to inspect relevant evidence, use another method or model, and report disagreement without being rewarded for deployment. Possible checks include reimplementation, alternative data cleaning, sensitivity to the boundary and metric, adversarial cases, affected-party review, and comparison with a simple baseline.

For the highest-stakes systems, divide evidence production and approval. The model team can explain assumptions; it should not be the sole judge of their adequacy. Publish unresolved disagreement proportionally to the decision. A decision can proceed under disagreement, but the disagreement must not disappear from the record.

Every modeled quantity should have a measurement protocol. Name the construct, operational definition, instrument or data source, unit, sampling frame, timing, aggregation, missing-data rule, uncertainty, and owner. State how the measure can be manipulated and which groups or conditions it represents poorly.

Consider “wait time” in a public service. It can mean time from need to first request, request to appointment, arrival to service, or cumulative time across referrals. Each definition answers a different question. Administrative timestamps may omit people who abandon before registration. Averaging completed cases can improve when the longest-waiting cases remain unresolved. The optimization must choose a clock and include censoring, abandonment, and repeated contact.

Construct validation combines quantitative and qualitative evidence. Compare the measure with related and distinct constructs, inspect known groups and edge cases, and ask participants whether the operational definition captures their experience. A high correlation with another proxy is not enough. The protocol should specify when the construct changes—for example, when virtual service changes what “arrival” means.

Measurement error propagates through optimization nonlinearly. If a decision selects extremes, even unbiased error can create biased selected values. Use repeat measurement, reliability models, shrinkage, and out-of-sample confirmation where appropriate. For thresholds, study classification near the boundary; for constraints, include uncertainty margins rather than pretend the measured value is exact.

A baseline includes the policy and its operating context. We need to record who received each action, how resources were rationed, and which adaptations occurred. A single historical

average cannot describe these conditions. If an optimizer changes eligibility, workload, or reporting, the comparison must account for the changed population and mechanism.

Use several baselines: current practice, simple transparent rule, no-action or minimum-service case, and a domain-expert policy. A complex method should show incremental value against the strongest reasonable simple alternative. When randomization is not appropriate, use prospective shadow decisions, natural experiments, matched comparisons, interrupted time series, or simulation with explicit causal limits.

Evidence may depend on the site, population, time, and implementation. Before using it elsewhere, we need to describe which causal mechanisms are shared, which effect modifiers differ, and how resources and workflows compare. We can then identify local observations that would test these assumptions.

For example, a trial estimates an effect for its participating centers and protocol. Another setting may have different staffing, adherence, language, baseline risk, or treatment alternatives. Similarly, a software benchmark describes performance on particular hardware and workloads. Production introduces concurrency, failures, and operator actions. A material test on a coupon also differs from a manufactured assembly with joints and defects.

Separate three claims. **Internal validity** concerns the original estimate. **Transportability** concerns whether relevant mechanisms and modifiers support a new setting. **Implementation validity** concerns whether the intervention delivered there matches the one studied. New local calibration cannot repair a different causal mechanism.

We can investigate the transfer with a small prospective study, shadow deployment, local measurement of effect modifiers, and checks of the proposed mechanism. We also specify which discrepancies require further investigation. The external source and the local evidence should remain identifiable, since combining them into one confidence score may hide their different roles.

Missingness can arise from failure, burden, access, discretion, avoidance, or because an outcome has not yet occurred. The missing-data model should describe this process. Imputation under a convenient assumption is not neutral.

Map missingness by group, time, state, and decision. Ask whether the optimization changes who is observed. A triage model can reduce follow-up for low-ranked cases, making their outcomes missing and confirming the model. A maintenance policy can inspect only high-risk equipment and bias failure labels.

Use sensitivity analyses for plausible missing-not-at-random mechanisms, collect audit samples, and include missingness indicators only with causal care. Sometimes the correct intervention is to improve measurement access rather than optimize over incomplete records.

For each claim, we specify a review date or a condition that requires review. Software, law, populations, equipment, climate, and practice change at different rates. An earlier study can

remain correct for its original setting even when the assumptions needed to apply it elsewhere no longer hold.

Triggers include model or interface change, new population, policy response, incident, material supplier change, drift, or revised scientific guidance. Review should ask whether old evidence still covers the current system, not merely whether the citation remains accessible.

Governance as feedback and memory

We can model governance as a control layer over the optimization lifecycle. It observes behavior and consequences, compares them with requirements, and changes permissions, constraints, or the formulation. This analogy helps explain why review must continue after deployment. It has limits: stakeholders cannot be treated simply as disturbances, rights as tunable penalties, or legitimacy as a stability margin.

A governed project has at least four loops.

Fast operational loop. Monitors feasibility, service limits, safety alarms, and data quality; can pause or fall back immediately.

Model loop. Re-estimates parameters, checks drift, recalibrates uncertainty, and retests sensitivity on a scheduled cadence.

Decision loop. Reviews whether recommendations improve real outcomes, whether operators override them, and whether affected parties can obtain correction or recourse.

Purpose loop. Reopens the boundary, objective, and institutional mandate. It asks whether the system should continue to exist in its present form.

The loops require different authority. The team benefiting from a favorable metric should not be its sole interpreter. A named owner remains accountable, while independent review, incident reporting, and external challenge provide ways to question and revise the decision.

A metric should be tested not only on historical data but under anticipated optimization pressure. If admissions, funding, compensation, or ranking depends on a score, ask how a rational participant could raise the score without improving purpose. Red-team the measurement process: change documentation, shift difficult cases, alter denominators, delay events, relabel outcomes, or move cost outside the boundary. These tests examine differences that the metric fails to distinguish before incentives encourage their exploitation in production (Campbell 1976; Goodhart 1975; Espeland and Sauder 2007; Muller 2018).

We can compare the target metric with other measurements and qualitative observations. For example, a throughput objective can be examined together with rework, the distribution of delays, staff workload, and user-reported quality. A model-accuracy objective can be examined together with calibration, abstention, distribution shift, energy use, and the benefit of subsequent decisions. These checks do not eliminate manipulation, but they help us identify improvements in one measure that conceal deterioration elsewhere.

Monitoring needs a response rule. For each metric, we specify the threshold, trend, review interval, owner, action, and escalation procedure. We also include slow outcomes and leading indicators, since a system may remain within monthly limits while a rare hazard accumulates or trust declines.

We need to retain versions of decisions, incidents, appeals, overrides, and model changes. When staff change, these records explain why an existing constraint was introduced. Linking each change to its evidence and retaining rejected alternatives allows a later reviewer to reconstruct the decision and the reasons given for it.

Field checklist

- Write the evidence chain and associate each link with a distinct validation method.
- Test causal claims as causal claims rather than substituting prediction accuracy.
- Define fast operational, model, decision, and purpose-review loops.
- Separate ownership from independent review and provide a real path for contestation.
- Red-team the metric under selection pressure before deployment.
- Preserve baselines, failed alternatives, overrides, incidents, and dissent in the dossier.
- Schedule reframing; do not wait for parameter drift to reveal a purpose failure.

Translation lens. Evidence connects modeled consequences with observed consequences across problem versions. We can compare this relationship with a natural transformation when the relevant objects, maps, and compatibility conditions have been defined. Governance determines how the comparison is inspected, challenged, and used to revise the optimization problem.

Part II · Mathematical Foundations

Chapter 4 · Continuous optimization: calculus, convexity, duality, and certificates

Calculus and local numerical geometry

Continuous optimization begins with local change. If $f: \mathbb{R}^n \to \mathbb{R}$ is differentiable, its gradient $\nabla f(x)$ is the direction of steepest increase under the Euclidean norm. The negative gradient therefore supplies a local descent direction:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \tag{16}$$

where the step size α_k may be fixed, varied according to a schedule, or selected by line search. The geometry affects the resulting sequence. For example, poorly scaled variables can produce a narrow valley in which the iterates oscillate. Saddle points may slow or redirect the search. With noisy evaluations, the sequence becomes stochastic. Constraints may require projection or a different choice of geometry.

First- and second-order conditions

For an interior local minimizer x^* of a differentiable function, $\nabla f(x^*) = 0$ is necessary. If f is twice differentiable, the Hessian $H(x) = \nabla^2 f(x)$ describes local curvature. A positive-definite Hessian at a stationary point is sufficient for a strict local minimum. Newton's method uses a local quadratic model:

$$x_{k+1} = x_k - H(x_k)^{-1} \nabla f(x_k). \tag{17}$$

The inverse in Equation (17) requires a nonsingular Hessian. Positive definiteness gives a descent direction when the gradient is nonzero. Near a well-behaved solution it can converge rapidly, but forming and solving with the Hessian can be expensive. Quasi-Newton methods such as BFGS update an approximation using gradient differences. Conjugate-gradient and Krylov methods exploit large sparse structure. Trust-region methods restrict each local model to a region where it is credible (Nocedal and Wright 2006).

Geometry other than Euclidean geometry

The gradient depends on the inner product used to represent the differential. Preconditioning changes this metric to account for different scales in the problem. Mirror descent uses a Bregman divergence in place of Euclidean distance, which can be useful for probability vectors, simplex constraints, or sparse representations. A natural gradient uses a metric from information geometry. On a manifold, we compute a Riemannian gradient in the tangent space and use a retraction to return the step to the manifold.

Nonscalar and complex variables

Matrix variables appear in control, covariance estimation, semidefinite programming, and quantum information. Function variables appear when the decision is a curve, field, or policy. The calculus of variations considers a functional such as

$$J[y] = \int_a^b L\left(t, y(t), y'(t)\right) dt. \tag{18}$$

Under regularity and fixed endpoint conditions, an extremal satisfies the Euler-Lagrange equation

$$\frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial y'}\right) = 0. \tag{19}$$

We have changed the decision object from a point to a path. This type of formulation occurs in the study of geodesics, mechanics, shape design, image processing, and optimal control.

Discretize then optimize, or optimize then discretize?

Many scientific and engineering problems are formulated in an infinite-dimensional space and then discretized for computation. A coarse discretization may allow the optimizer to improve a numerical artifact instead of the intended response. Also, differentiating the discretized problem may give a different result from discretizing a continuous adjoint. We therefore need mesh refinement, consistency checks, and adjoint verification during the optimization process.

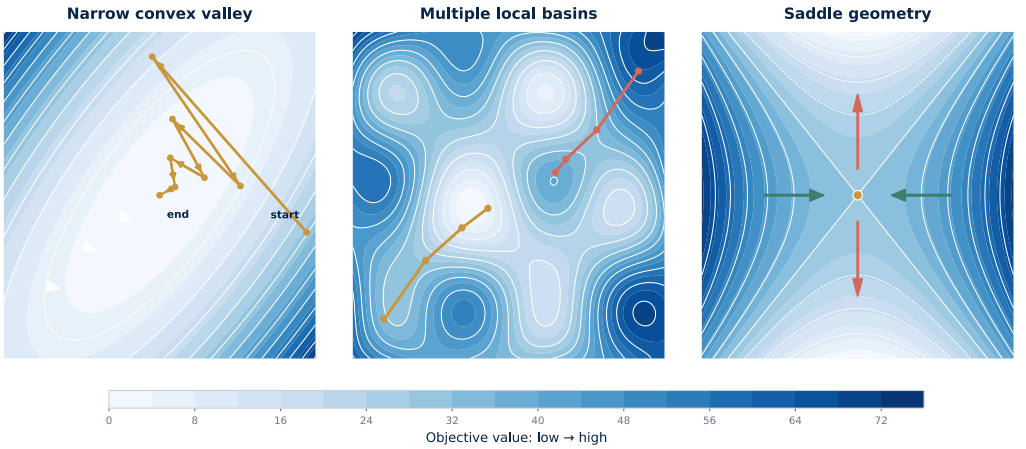


Figure 5. Narrow-valley, multiple-basin, and saddle contour landscapes with distinct search directions.

In Figure 5, the arrows in the narrow valley show a sequence of decreasing zig-zag steps from the darker starting point toward the lighter minimum. The other panels illustrate multiple basins and directions near a saddle point. All panels use the same low-to-high scale.

We need to distinguish the local quantities used by continuous optimization. The differential is a covector; an inner product identifies it with a gradient vector. A Hessian describes local curvature, while convexity allows supporting hyperplanes to provide global bounds. These quantities support different conclusions. A zero gradient may occur at a minimum, maximum,

saddle, or flat point. A positive semidefinite Hessian at one point does not establish global convexity. A small step may indicate convergence, poor scaling, or a stalled algorithm.

Constraints replace the unconstrained zero-gradient condition with a normal-cone balance. Geometrically, KKT stationarity expresses the negative objective gradient as a nonnegative combination of active inequality normals together with an unrestricted combination of equality normals.

Constraint qualifications ensure that the local geometry of the constraint representation matches the geometry of the feasible set closely enough for multipliers to exist. A convex problem does not automatically make every KKT necessity claim valid; Slater's condition is a standard sufficient hypothesis for convex inequalities, while other settings require other regularity conditions (Boyd and Vandenberghe 2004; Nocedal and Wright 2006). We therefore state these conditions together with the certificate.

Real solvers work with floating-point arithmetic, scaled residuals, tolerances, and stopping tests. A status of "optimal" commonly means that primal infeasibility, dual infeasibility, and a scaled gap fall below configured thresholds. Ill-conditioning can make small residuals coexist with large errors in variables or sensitivities. Units that differ by twelve orders of magnitude can distort a method even when the algebra is correct.

A numerical certificate should report at least the primal residual, dual residual, complementarity or gap, scaling, iteration limit, and termination reason. For nonconvex problems it should distinguish local stationarity from global evidence. Multiple starts, lower bounds, interval methods, convex relaxations, or problem-specific arguments can strengthen a claim, but none should be implied by the word "converged."

Derivative checking should be routine. Automatic differentiation computes derivatives of the executed program, which may differ from the intended mathematical model. Compare analytic, adjoint, or automatic derivatives with directional finite differences at representative and boundary points. Complex-step checks avoid subtraction error when the code path admits the required analytic extension. Discontinuous branches, clipping, stochastic sampling, and iterative inner solvers require separate checks. Verify units and signs, and test more than one point.

First-order methods are useful when dimensions are large, gradients are available, and moderate accuracy is sufficient. Accelerated and variance-reduced variants require additional assumptions. Newton and quasi-Newton methods use curvature information to improve local convergence near a well-behaved solution. Trust-region methods explicitly limit where a local model is used. Interior-point methods provide primal-dual information for structured constrained problems, while proximal and splitting methods exploit separable nonsmooth terms. When only function evaluations are available, we can consider derivative-free methods.

We need to consider the full computation required by a method. An iteration that appears inexpensive may include a PDE solution or synchronization between machines. Fewer

iterations may still require more energy or elapsed time. The cost model should therefore include algorithmic complexity, memory transfers, parallel execution, and evaluation fidelity.

The mathematical problem and its numerical representation can have very different geometry. If one variable is measured in meters and another in micrometers, or one constraint residual is near 10^6 while another is near 10^{-4}, termination tests and search directions can be dominated by units. Nondimensionalize where possible, choose physically meaningful scales, and report residuals in both scaled and original units.

Conditioning describes how a solution responds to small changes in the problem. An ill-conditioned optimum may be computed accurately for the recorded data while changing substantially under measurement error. Examples include nearly collinear regressors, nearly active constraints, flat valleys, and alternatives with almost equal values. Regularization, bounds, or another parameterization may stabilize computation. However, these changes also modify the problem and need a justification.

Many practical objectives are composite: a smooth loss plus a norm, indicator, hinge, maximum, or piecewise tariff. For

$$\min_x F(x) = f(x) + g(x), \tag{20}$$

with smooth f and convex possibly nonsmooth g, proximal methods use the map

$$\mathrm{prox}_{\alpha g}(v) = \arg\min_x \left(g(x) + \frac{1}{2\alpha} \| x - v \|_2^2 \right). \tag{21}$$

The proximal step preserves structure that smoothing can blur: sparsity, constraints, or a sharp penalty. Equation (21) appears in regularized estimation, imaging, signal recovery, and resource allocation (Parikh and Boyd 2014). When applying this formulation, we need a domain reason for the regularizer, such as a prior or cost. The availability of a sparsity-producing algorithm does not provide that reason.

For genuinely nonsmooth nonconvex models, stationarity has several definitions and solver messages can be easy to overinterpret. Report the notion used, residual, and local perturbation tests. If discontinuity comes from a design rule or simulation switch, ask whether the discontinuity is real or a modeling artifact.

A numerical trust-region method limits a step to a region in which a local approximation is considered reliable (Nocedal and Wright 2006). We can use a similar idea when applying a model. Here the region is described by input distributions, states, actions, environments, and consequences. Observations may support interpolation inside this region. Near its boundary, we need additional checks; outside it, we may need to abstain, use a fallback, or collect more evidence.

Because search often approaches boundaries, we can record distance from the region supported by evidence as a diagnostic quantity. A candidate may satisfy mathematical constraints while lying outside that region. Tests should extend the region progressively as observations support the predictions.

For an irreversible decision, we may need to define this region using consequences as well as variables. Familiar inputs can combine to produce a rare outcome that has not been validated. We therefore test interactions and common-cause conditions in addition to varying each parameter separately.

Convex relaxations can supply lower bounds, feasible guidance, or tractable surrogates. Report which role is used. If a rank, integrality, or physics constraint is relaxed, inspect the relaxed solution's violation and whether rounding preserves hard requirements. A tight objective bound does not ensure an easily recoverable design.

Successive convex approximation solves local convex models of a nonconvex problem. It needs a feasible or recoverable initialization, conservative constraint treatment, and monitoring of actual versus predicted improvement. Report dependence on starts. The method produces a local claim unless global structure supplies more.

We assign units to variables, objectives, gradients, multipliers, and residuals. For example, a gradient component has objective units divided by the units of its variable. A multiplier has objective units divided by constraint units. Checking these relationships can reveal modeling and implementation errors before the optimizer is run.

For dimensionless learned scores, state their scale and transformation. A one-unit change in a logit, embedding distance, or standardized feature has no direct operational meaning. Translate sensitivity through calibration to consequences before presenting it as importance.

Tests of data transformations and exported decisions should include timestamps, currencies, coordinate systems, and scale factors. Errors in these representations can invalidate an otherwise correct mathematical model.

Convexity, duality, and certificates

Convexity is the central tractable structure of continuous optimization. A set C is convex when every line segment between two of its points remains inside it. A function f is convex on C when

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta)f(y), \qquad 0 \le \theta \le 1. \tag{22}$$

The inequality says that the graph lies below its chords. Convex problems have the following local-to-global property: every local minimum is global. When f is strictly convex, the minimizer is unique if it exists. Strong convexity and smoothness give quantitative convergence rates and condition numbers (Boyd and Vandenberghe 2004; Bubeck 2015).

Convexity as model knowledge

We can often identify a suitable method by first checking convexity. Affine functions are both convex and concave, and norms are convex. Pointwise maxima, nonnegative sums, and affine compositions preserve convexity under the corresponding composition rules. Disciplined convex modeling systems use these rules to check whether a formulation belongs to the supported problem class.

Lagrange duality

For the constrained problem of Chapter 1, the dual function is

$$q(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu), \qquad \lambda \geq 0. \tag{23}$$

Every dual-feasible pair produces a lower bound on a minimization problem. The dual problem maximizes this bound:

$$\max_{\lambda \geq 0, \nu} q(\lambda, \nu). \tag{24}$$

Let p^* denote the primal minimum value and d^* the dual maximum value; these starred value symbols differ from p, the number of equality constraints. Weak duality always gives $d^* \leq p^*$. For a convex problem satisfying Slater's condition, strong duality gives $d^* = p^*$. More generally, strong duality requires hypotheses that must be stated rather than inferred from convexity alone. A matching primal solution and dual bound form an optimality certificate.

Karush-Kuhn-Tucker conditions

For differentiable convex problems, the KKT conditions are

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \nu_j^* \nabla h_j(x^*) = 0, \tag{25}$$

$$g_i(x^*) \leq 0, \qquad h_j(x^*) = 0, \tag{26}$$

$$\lambda_i^* \geq 0, \qquad \lambda_i^* g_i(x^*) = 0. \tag{27}$$

The last condition is complementary slackness: either an inequality is inactive and its multiplier is zero, or it is active and may carry a nonzero shadow value. Even for convex problems, necessity of the KKT conditions requires an appropriate constraint qualification, such as Slater's condition in the standard differentiable inequality-constrained setting. Under convexity, KKT conditions are sufficient for global optimality when a feasible primal-dual point satisfies them; outside convexity they are generally only local necessary conditions under regularity and do not certify a global optimum (Kuhn and Tucker 1951; Rockafellar 1970; Boyd and Vandenberghe 2004).

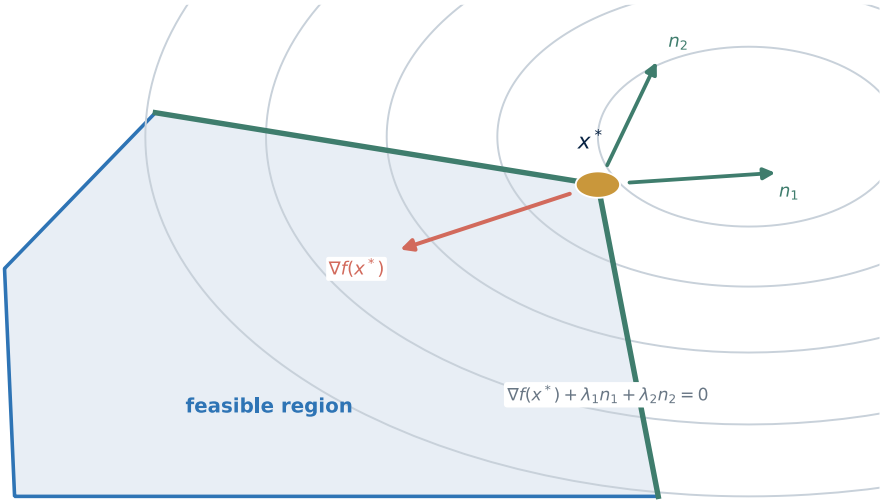


Figure 6. Feasible region, objective contours, active constraints, normal vectors, and the KKT balance at an optimum.

Figure 6 interprets KKT stationarity geometrically: at the constrained optimum, the objective gradient is balanced by normals to the active constraints.

Fenchel duality and proximal structure

For convex f, the convex conjugate, written f^* locally rather than denoting the optimal value used in Chapter 1,

$$f^*(y) = \sup_x \big(\langle y, x\rangle - f(x)\big) \tag{28}$$

translates between primal and dual representations. Nonsmooth regularizers often have simple proximal maps:

$$\operatorname{prox}_{\gamma f}(v) = \arg\min_x \left(f(x) + \frac{1}{2\gamma} \| x - v \|^2\right). \tag{29}$$

This restates the proximal map of Equation (21), using f for the generic nonsmooth term and γ for the proximal parameter. This local use of f differs from the smooth term in Equation (20). Splitting methods decompose objectives into parts whose individual proximal operations are easy (Parikh and Boyd 2014).

Suppose a workshop produces two products, with contributions 3 and 5, using two limited resources. The continuous relaxation is

$$\max_{x_1, x_2 \geq 0} \; 3x_1 + 5x_2 \quad \text{subject to} \quad x_1 + 2x_2 \leq 8,\; 3x_1 + 2x_2 \leq 12. \tag{30}$$

Its dual assigns nonnegative shadow values y_1, y_2 to the two resources:

$$\min_{y_1, y_2 \geq 0} \; 8y_1 + 12y_2 \quad \text{subject to} \quad y_1 + 3y_2 \geq 3,\; 2y_1 + 2y_2 \geq 5. \tag{31}$$

Weak duality says every dual-feasible resource valuation upper-bounds every primal-feasible production value. When primal and dual values coincide, the pair certifies global optimality. The multipliers also give local sensitivity: if a resource limit increases slightly and the active set remains stable, the corresponding dual value estimates marginal benefit. That interpretation is conditional. Large changes can alter the basis; discrete production invalidates marginal divisibility; market, environmental, or labor consequences may not be present in the objective.

We can find corresponding interpretations in other disciplines. In networks, dual variables become potentials or congestion prices. In mechanics, they become reactions or sensitivities. In control, adjoints propagate the value of state perturbations backward. In machine learning, Lagrange multipliers enforce fairness, privacy, or resource constraints. The mathematical role transfers, but the legitimacy of interpreting a multiplier as a price does not transfer automatically.

We can also use duality diagnostically. A dual certificate of primal infeasibility can identify an impossible target set. A large multiplier may indicate a bottleneck or a scaling problem. A duality gap may reflect nonconvexity, numerical difficulty, or incomplete convergence. In decomposition, dual messages coordinate component decisions.

Interpretation must remain conditional. A multiplier is a marginal value for a small relaxation under the model and active set. It may not remain valid for a discrete policy change, large expansion, strategic response, or moral trade. Before calling it a “price,” test perturbations, units, sign, stability, and whether the constraint owner recognizes the proposed trade.

We can classify the available evidence by the claim it supports: a sampled candidate, a feasible point, a stationary point, a local minimum under stated regularity, a global optimum under convexity or global search, an approximation guarantee, a verified invariant, or an implementation-level safety case. These claims concern different aspects of the result. For example, a global optimum based on invalid data can provide less useful evidence than a carefully validated feasible baseline.

We report mathematical and empirical uncertainty together. An optimality gap of 0.2% and outcome uncertainty of ±15% indicate different limits on the result. Further numerical search may be unnecessary when its remaining gap is immaterial to the decision.

A derivative describes an infinitesimal change with the active structure held fixed. A practical decision may involve a finite change, uncertainty, or a transition to another regime. We can examine these effects by varying parameters over plausible ranges and solving the model again. For each solution, we record the objective, decision, active constraints, feasibility margin, and operating regime.

Plot where the selected alternative changes. A stable objective value can hide a discontinuous decision; a changing objective can coexist with a stable action. For discrete or nonconvex problems, one-sided and scenario sensitivities are often more informative than a derivative.

For learned models, include retraining or policy response where relevant rather than perturbing one prediction in isolation.

Dual multipliers can help us choose which perturbations to investigate first. A large, stable multiplier suggests that the constraint has a substantial local effect. A large, unstable multiplier may indicate degeneracy or nearly equivalent alternatives. If active constraints can exchange multiplier values, we report the range of dual solutions instead of interpreting one value as a unique price.

Field checklist

- State the exact claim and its convexity or regularity hypotheses.
- Verify active-constraint signs and normal geometry.
- Report scaled primal, dual, and complementarity residuals.
- Cross-check derivatives independently at representative points.
- Test finite-range sensitivity before interpreting shadow values.
- Separate a mathematical multiplier from a legitimate social price.

Translation lens. Continuous optimization uses tangent directions, normal cones, curvature, adjoints, and dual bounds across disciplines. In each application, we need to specify the meaning of a perturbation, the units of sensitivity, and who may authorize a relaxation of a constraint.

Chapter 5 · Structured search: linear, conic, discrete, global, and derivative-free methods

Linear and conic forms as compilation targets

Linear programming chooses $x \in \mathbb{R}^n$ under linear objectives and constraints:

$$\min_x c^T x \quad \text{subject to} \quad Ax = b,\ x \geq 0. \tag{32}$$

The feasible set is a polyhedron whose extreme points correspond to basic feasible solutions. The simplex method moves between basic feasible solutions, whereas interior-point methods follow points in the interior. Linear programming has been applied to production, transport, diet, blending, scheduling relaxations, network flow, and resource allocation (Dantzig 1963; Karmarkar 1984).

The corresponding dual is

$$\max_y b^T y \quad \text{subject to} \quad A^T y \leq c. \tag{33}$$

Primal and dual variables often express two views of one system: quantities versus marginal values, flows versus potentials, or plans versus certificates.

Quadratic programming

A quadratic program has objective

$$\frac{1}{2} x^T Q x + c^T x \tag{34}$$

with linear constraints. If $Q \succcurlyeq 0$, the problem is convex. Quadratic programs appear in least squares, portfolio optimization, support-vector machines, model predictive control, and sequential quadratic programming. Equality-constrained QPs reduce to structured linear systems; active-set and interior-point methods address inequalities.

Conic form

Conic optimization unifies several families:

$$\min_x c^T x \quad \text{subject to} \quad Ax = b,\ x \in K, \tag{35}$$

where K is a convex cone. Choosing the nonnegative orthant yields linear programming. Second-order cones express norm constraints such as $\| Ax + b \|_2 \leq c^T x + d$. The positive-semidefinite cone yields semidefinite programming (SDP), where a symmetric matrix variable X must satisfy $X \succcurlyeq 0$.

SDP supports control synthesis, moment relaxations, covariance problems, geometry, and combinatorial approximation. The Goemans-Williamson Max-Cut algorithm solves an SDP relaxation and rounds its solution to obtain a provable approximation (Goemans and Williamson 1995). We can therefore use a common conic form to represent different feasible geometries.

Modeling with relaxations

A relaxation enlarges the feasible set or weakens a constraint to obtain a tractable bound. A tight relaxation guides exact search and can be operationally useful by itself. Relaxations are also diagnostics: a large gap between the relaxed and feasible solution reveals where discrete or nonlinear structure matters. Poorly chosen “big-M” constants, however, can destroy numerical conditioning and bound quality.

Numerical scale and sparsity

Large structured programs are solved by exploiting sparsity, block structure, low rank, separability, and warm starts. Rescaling variables and constraints affects conditioning, error amplification, and iteration behavior. Solver status must be read with tolerances in mind. “Optimal” means optimal within a numerical model and stopping criterion.

Several transformations occur before a solver can process a domain model. We first represent domain concepts by variables and constraints. Nonlinear relations may be expressed in conic form or approximated by piecewise functions. Discrete logic may become binary variables, clauses, or propagators, and uncertainty may become scenarios or sets. The resulting equations are stored as sparse matrices and expression graphs. Presolve reduces the instance before the algorithm produces iterates, nodes, cuts, or samples. Their execution also requires memory transfers and communication.

Each compilation stage can preserve, relax, or distort structure. An exact reformulation preserves feasible decisions and objective values, although it may introduce auxiliary variables. A relaxation preserves a bounding relation (see “Modeling with relaxations”). An approximation changes values within a stated error. A surrogate predicts an expensive response and carries statistical uncertainty. A heuristic transformation may preserve neither feasibility nor bounds but improve search empirically. In the category-theoretic analogy, these are different types of morphisms. The model record should identify which type each transformation represents.

Discrete search as proof construction

Some decisions cannot be divided into arbitrary fractions. For example, we may build a facility, assign a worker to a shift, select a route edge, place a block on a grid, include a feature, or choose a word. Integer optimization represents these choices using integer variables. A mixed-integer linear program (MILP) has the following form:

$$\min_x c^T x \quad \text{subject to} \quad Ax \le b, \quad x_i \in \mathbb{Z} \text{ for } i \in I. \tag{36}$$

The integrality requirement changes the feasible set. Its continuous relaxation may have a good fractional solution that cannot be implemented. Exact methods address this difference by combining the following operations.

- **Branching** divides the feasible set.
- **Bounding** uses relaxations to exclude subtrees.

- **Cutting planes** add valid inequalities that remove fractional solutions.
- **Presolve** simplifies variables and constraints.
- **Primal heuristics** seek feasible incumbents early.
- **Decomposition** separates complicating variables or scenarios.

We can interpret branch-and-bound as constructing a proof tree. A feasible incumbent supplies a witness; valid relaxation bounds exclude subtrees, while cuts and presolve simplify the remaining search. For minimization, the gap compares the incumbent upper bound with the best global lower bound, obtained from the minimum bound over open nodes. When no unresolved region can improve the incumbent within tolerance, the tree supports an optimality certificate relative to the formulation and arithmetic.

Graph and network problems

Shortest paths, flows, matchings, spanning trees, cuts, and colorings reveal how combinatorial structure creates specialized algorithms. Some problems admit integral polyhedra, so a linear relaxation already returns integer solutions. Others are NP-hard and motivate approximation, parameterized algorithms, or problem-specific heuristics (Schrijver 2003).

The stable matching problem illustrates that the objective need not be a scalar. Gale-Shapley deferred acceptance finds a matching with no blocking pair under stated preferences (Gale and Shapley 1962). This existence result establishes stability; it does not by itself establish fairness, strategy-proofness, or welfare optimality.

Submodularity

A set function $f\colon 2^V \to \mathbb{R}$ is submodular when it has diminishing returns:

$$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B) \tag{37}$$

whenever $A \subseteq B$ and $e \notin B$. Normalized, nonnegative, monotone submodular maximization under a cardinality constraint admits the classical greedy $1 - 1/e$ guarantee. Coverage, sensor placement, influence, and summarization require their respective modeling assumptions before this guarantee applies (Nemhauser, Wolsey, and Fisher 1978; Nemhauser and Wolsey 1988). It is a discrete analogue of convexity in some respects, though the analogy must be used carefully.

Constraint programming and satisfiability

Constraint programming propagates consequences through variable domains and global constraints. SAT and SMT solvers use clauses, conflict analysis, and supported theories to establish logical satisfaction; MaxSAT and pseudo-Boolean optimization add objectives. Planning, verification, configuration, and scheduling often benefit from these representations more than from a generic MILP.

Symmetry and representation

Equivalent representations create redundant search. Symmetry-breaking constraints, canonical forms, and quotient spaces reduce the domain without changing distinct solutions.

Representation can alter difficulty dramatically even when the underlying problem is unchanged.

The solver log records this evolving proof. Rapid incumbent improvement with a slowly improving bound may indicate a weak relaxation. A strong bound without a feasible solution may indicate difficult primal structure. A growing node count may reflect symmetry, poor branching, or a weak formulation. Numerical warnings require separate examination because they can undermine the apparent certificate. We can use these trace observations to guide investigation and parameter changes.

Different representations allow different forms of inference. Dynamic programming can use a small state that retains the information required for future decisions. Network algorithms use graph structure and properties such as total unimodularity. Submodular optimization uses diminishing returns. Although a MILP can express many of these problems, another representation may make the relevant structure easier to use (Papadimitriou and Steiglitz 1982; Schrijver 2003; Nemhauser and Wolsey 1988).

In discrete optimization, an incumbent and a bound answer different questions. The incumbent shows an implementable value if it is truly feasible in the domain. The bound shows how much the formulation could still improve. A small gap can support stopping; a large gap does not mean the incumbent is poor relative to current practice.

Report four comparisons: incumbent versus operational baseline, incumbent versus best known alternative, bound versus incumbent, and real-world sensitivity versus the gap. An incumbent that reduces patient delay materially and is stable under duration uncertainty may justify use despite a loose proof bound. Conversely, a certified optimum that exploits unrealistic no-show estimates should be rejected.

Feasibility deserves independent repair and checking. Solver tolerances can permit small algebraic violations that become meaningful after rounding or unit conversion. Recompute every hard constraint from exported decisions in original units. For combinatorial designs, run a domain checker that does not share indexing code with the model.

Global, black-box, and surrogate search

An objective may contain kinks, discontinuities, multiple basins, simulator failures, or discrete transitions. Its derivatives may also be unknown. We need to identify which of these properties affects the problem before selecting a search method.

Nonsmooth structure

Convex nonsmooth functions retain global tractability but require generalized derivatives. A subgradient $g \in \partial f(x)$ is a vector, unlike the constraint functions denoted by g_i in Chapter 4, and satisfies

$$f(y) \geq f(x) + g^T(y - x) \quad \text{for all } y. \tag{38}$$

Subgradient methods are robust but can be slow. Proximal-gradient methods are effective for composite objectives $f(x) + r(x)$ with smooth f and simple nonsmooth r:

$$x_{k+1} = \mathrm{prox}_{\alpha r}\big(x_k - \alpha \nabla f(x_k)\big). \tag{39}$$

Here r denotes the nonsmooth term called g in Equation (20). This pattern underlies sparsity-inducing regularization, constrained projections, and operator splitting.

Nonconvex landscapes

Nonconvexity removes the local-to-global guarantee. Multiple starts, continuation, homotopy, deterministic global methods, relaxations, and problem-specific initialization become important. In large machine-learning problems, the practical target is often a solution with good validation behavior rather than a certified global optimum; training loss and decision utility must not be confused.

Deterministic global methods use interval bounds, convex envelopes, spatial branch-and-bound, or algebraic moment relaxations to establish global bounds. Their computational cost may limit the size of the problems they can certify. When evaluations are expensive, we can instead approximate the objective with a surrogate and use an acquisition function to select the next evaluation. This selection balances investigation of uncertain regions with improvement near promising candidates (Jones, Schonlau, and Welch 1998; Shahriari et al. 2016).

The black-box interface

Suppose we can submit a design to a simulator and obtain a response, but cannot differentiate the internal calculation. We can treat this interface as a black box. The same situation occurs when tuning a service through load tests, selecting a laboratory experiment, or comparing candidate designs through human judgments. The black-box description concerns the information available to the optimizer. It does not imply that the underlying system has no structure.

We use three terms for different properties of a problem or method. By black-box, we mean that the optimizer obtains responses through an evaluation interface. By derivative-free, we mean that the method does not require supplied derivatives. Global describes the scope of the solution being sought. A derivative-free method may therefore perform only local search. Also, a black-box search may still use known bounds, conservation rules, or a partially known model. If more internal information becomes available, we can include it in a gray-box or hybrid formulation to help construct feasible candidates and check results.

Before choosing a method, we record what an evaluation returns: a deterministic value, a noisy observation, several objectives, a constraint measurement, a preference comparison, or a failure status. We also record cost, runtime, variable types, reproducibility, and whether another evaluation can run concurrently. These properties can occur together. For example, a constrained experiment may have mixed variables, noisy outputs, and several fidelity levels.

Random and space-filling search

Random search provides a useful baseline. We specify distributions over the variables and sample candidate configurations. Unlike a full grid, it can examine many different values of an influential variable without repeating the same small set at every combination of less influential variables. Bergstra and Bengio (2012) demonstrated the importance of this distinction in hyperparameter optimization. A baseline still requires a meaningful domain: uniform sampling of a scale parameter and uniform sampling of its logarithm represent different search priorities.

Space-filling initial designs distribute evaluations across a specified domain before adaptation begins. They can help reveal broad response variation and provide initial data for a surrogate. Coverage in the coordinates does not ensure coverage of important behaviors, especially when most feasible designs occupy a small region. We therefore inspect feasibility and response diversity as well as the geometric arrangement of the samples.

Derivative-free and zeroth-order methods

Derivative-free methods use function evaluations without requiring supplied derivatives. Direct search compares selected points, local model-based methods fit approximations, and randomized zeroth-order methods estimate derivatives or directions from perturbed evaluations. These approaches are useful when derivatives are unavailable or unreliable. Their assumptions still matter: a method developed for smooth responses does not automatically handle arbitrary discontinuities or noise (Larson, Menickelly, and Wild 2019).

Direct search and local interpolation models

The Nelder-Mead method maintains a simplex of evaluated points and changes it through reflection, expansion, contraction, and shrinkage (Nelder and Mead 1965). Its simplex is unrelated to the basis used by the simplex method for linear programming. The method is convenient for small continuous problems, but convergence of its simplex does not generally establish stationarity. The review by Larson, Menickelly, and Wild (2019) discusses counterexamples and distinguishes this behavior from methods with explicit stationarity results.

Mesh adaptive direct search, or MADS, uses search and poll steps on an adapting mesh. Its polling directions become sufficiently rich to support constrained stationarity results under stated regularity conditions (Audet and Dennis 2006; Audet, Custódio, and Dennis 2008). Here, a convergence theorem concerns the specified limiting stationarity property. It does not provide a finite-budget certificate that all other basins have been excluded. Constraint handling and the regularity hypotheses remain part of the claim.

Local model-based methods approximate the response around evaluated points and limit the next step to a trust region. BOBYQA uses quadratic interpolation for continuous problems with bound constraints (Powell 2009). The model can provide gradient-like information even though the original evaluator supplies no derivatives. We need informative sample geometry

and a response that the local approximation can describe. Simulator noise, abrupt switches, or repeated invalid evaluations may require a different method or a specialized extension.

Randomized zeroth-order estimation

Simultaneous perturbation stochastic approximation, or SPSA, estimates a gradient by perturbing all variables together. Its basic two-sided estimate uses two objective measurements regardless of the number of variables (Spall 1992). This describes the cost of one estimate, not a dimension-independent total optimization cost. Estimation variance, perturbation scale, step schedule, and any averaging still affect the number of evaluations needed.

Suppose a noisy simulation has many continuous controls. Changing each coordinate separately may be expensive, while a pair of simultaneous perturbations is affordable. SPSA is then a candidate method. If small perturbations are hidden by quantization or simulator noise, the estimated direction may carry little information. We can compare repeated perturbations and change their scale, while retaining separate evaluations for final confirmation.

Metaheuristics

Simulated annealing accepts some uphill moves with a temperature-dependent probability, echoing statistical mechanics (Kirkpatrick, Gelatt, and Vecchi 1983). Evolutionary algorithms vary and select populations (Holland 1975). Particle swarm methods use social and individual memory (Kennedy and Eberhart 1995). Ant-colony optimization reinforces promising paths (Dorigo and Stützle 2004). These methods are flexible, parallelizable, and often effective on mixed or black-box spaces. Their guarantees depend on the particular method and assumptions. Practical performance also depends on budget, representation, and tuning.

Covariance adaptation and differential evolution

CMA-ES uses the rankings of sampled candidates to adjust the mean, scale, and covariance of a search distribution. Covariance adaptation allows it to represent dependencies between continuous variables and directions with different scales (Hansen 2016). We can consider this as learning a geometry for subsequent search. However, this covariance describes the search distribution. It does not by itself describe calibrated posterior uncertainty about the objective.

Differential evolution constructs candidate displacements from differences between population members, combines them with a target candidate, and uses selection to retain improvements (Storn and Price 1997). It provides another population-based approach for continuous black-box problems. Population size, mutation scale, crossover, initialization, and boundary handling affect its behavior. Restarts and hybrid local refinement can help exploration, but a best sampled candidate remains distinct from a certified global optimum.

For either method, we need to examine the variable representation. Rounding a continuous proposal into categories can produce many duplicate evaluations and an artificial neighborhood. Permutations, graph structures, and conditional choices may require

specialized variation operators. When evaluation is very expensive, a large population can consume the budget before adaptation becomes useful.

Bayesian optimization and acquisition rules

Bayesian optimization maintains a probabilistic model of an expensive response and uses an acquisition rule to select the next evaluation. A Gaussian-process surrogate is a common choice. Efficient global optimization uses expected improvement to combine predicted improvement and uncertainty (Jones, Schonlau, and Welch 1998). After evaluating the proposed point, we update the model and repeat the selection. Optimizing the acquisition is an inner problem whose cost must also be counted.

Acquisition rules answer different questions. Probability of improvement concerns the probability of beating a reference level. Expected improvement also accounts for the size of improvement. Confidence-bound rules combine a predicted response with an uncertainty term. Thompson sampling optimizes a sampled response function, while information-based rules seek observations that reduce uncertainty relevant to the optimum. Their exploration behavior depends on model assumptions, parameters, and budget (Shahriari et al. 2016).

We can differentiate a smooth acquisition function even when the original evaluator is a black box. This does not supply derivatives of the real experiment. It supplies derivatives of the model-based selection criterion. Numerical behavior also matters: expected-improvement values can become too small for reliable floating-point optimization. LogEI reformulations address this difficulty, including related batch and multiobjective criteria (Ament et al. 2023). A poorly optimized acquisition can make an otherwise useful surrogate appear ineffective.

Mixed categorical and conditional variables

A configuration space may combine continuous values, integers, categories, and variables that exist only after another choice. Suppose a model family determines whether a kernel parameter or a tree-depth parameter is active. An inactive parameter is not an ordinary numeric value. We need the search representation to preserve this condition, rather than assign an arbitrary distance between incompatible configurations.

Tree-structured Parzen Estimator, or TPE, models distributions of configurations associated with better and worse observed responses and uses their density relationship to guide proposals (Bergstra et al. 2011). This differs from fitting a Gaussian-process posterior directly to the response function. SMAC3 provides sequential model-based optimization with support for complex configuration spaces, including random-forest surrogates used for algorithm configuration (Lindauer et al. 2022). The encoding, conditional structure, and treatment of inactive values remain part of the model.

Noisy evaluations and repeated measurements

Repeated evaluations of the same candidate may return different values because of random seeds, sampling, workload variation, measurement error, or an evolving environment. We distinguish the underlying response of interest from one observation. The lowest observed

value can be a favorable noise realization. Selecting it and reporting the same observation as confirmation introduces selection bias (Smith and Winkler 2006).

We therefore allocate evaluations between exploring new candidates and repeating existing ones. A noise model should distinguish variation that is roughly constant from variation that changes across the domain. Shared seeds or workloads can make paired comparisons more informative, but they also create dependence that the analysis must retain. Replication reduces sampling uncertainty; it does not remove a systematic simulator bias. Final candidates need confirmation using the declared estimand, such as expected response or a tail criterion, under representative conditions.

Constraints, hidden failures, and safe exploration

Known algebraic constraints can often be checked before an expensive evaluation. Unknown constraints may require the experiment itself. Constrained Bayesian optimization models objective and constraint responses and selects candidates using improvement and feasibility information (Gardner et al. 2014). A small predicted probability of violation remains a probabilistic model output. It is not a proof that the next experiment is safe.

We also distinguish infeasibility from evaluator failure. A candidate may violate a design requirement, cause a simulator to fail numerically, exceed a time limit, or lose its output because of infrastructure failure. Replacing every case with one large objective penalty merges different mechanisms. We retain the status and any partial measurements. A timeout can be a censored runtime observation if the experiment has that interpretation; an infrastructure outage may require a controlled retry.

Safe optimization restricts the evaluations themselves. SafeOpt expands exploration from an initially safe set using confidence bounds and regularity assumptions, with guarantees concerning a safely reachable optimum (Sui et al. 2015). Such guarantees are conditional on those assumptions and the stated confidence construction. An isolated better region may remain unreachable under the safety rule. For a physical experiment, independent operating limits and stop procedures remain necessary even when the statistical model predicts safety.

Multiple fidelities and adaptive resource allocation

A fidelity is an evaluation setting that changes cost and accuracy: mesh resolution, simulation duration, dataset size, or training budget. In multi-fidelity optimization, we choose both a candidate and how to evaluate it. BOCA models a continuum of approximations and their relationship to the target response (Kandasamy et al. 2017). The target fidelity remains explicit. A good result from a coarse model is useful only to the extent that it informs that target.

Hyperband allocates resources among sampled configurations through successive halving and several initial budget arrangements (Li et al. 2018). It can stop unpromising trials before full evaluation without maintaining a Bayesian response model. BOHB combines this allocation approach with model-based configuration proposals (Falkner, Klein, and Hutter 2018). Thus,

early stopping, multi-fidelity modeling, and Bayesian search can be combined, but they describe different operations.

We need to test whether low-budget performance predicts full-budget performance. A slowly improving candidate can be eliminated too early; a coarse physical model can reverse the ranking of designs. The evaluation record should retain learning curves or fidelity settings, promotion and stopping decisions, and the cost of full-fidelity confirmation. Repeating a biased approximation does not make it equivalent to the target process.

Batch, asynchronous, and cost-aware search

When several evaluations can run together, batch selection should account for the information shared by proposed candidates. Taking the largest values of a one-point acquisition can select a redundant group. Pending evaluations can instead be represented through joint acquisition rules or possible outcomes, sometimes called fantasies. Cost-aware Bayesian optimization also accounts for variation in evaluation duration (Snoek, Larochelle, and Adams 2012).

Synchronous search waits for a batch to finish. Asynchronous search proposes a new candidate when a worker becomes available. The latter can improve resource use when runtimes differ, but its decisions are made with an incomplete evaluation history. We record pending, completed, failed, and cancelled runs separately. A policy that favors short evaluations may improve progress per hour while giving insufficient attention to expensive regions. This trade-off should be measured against the actual deadline and target quality.

Multiple objectives and preferences

For several expensive objectives, we may seek a set of trade-offs rather than one scalar optimum. Multiobjective Bayesian optimization uses surrogate models to select evaluations that improve the current approximation to the Pareto frontier. Noisy expected hypervolume improvement integrates uncertainty in that frontier; its parallel variant, qNEHVI, selects batches (Daulton, Balandat, and Bakshy 2021). Hypervolume depends on objective scales and a reference point, so these choices need to be reported. The frontier still requires a separate selection decision.

Sometimes the evaluator returns a comparison instead of a numerical score. A musician may prefer one response over another, or a designer may compare two forms without assigning stable ratings. Preferential Bayesian optimization learns a latent preference model from pairwise comparisons and selects further comparisons (González et al. 2017). We should retain the participant, context, uncertainty, and any disagreement. A fitted latent utility does not establish that preferences are stable, transitive, or shared by all participants.

High-dimensional search and transfer

A large number of variables makes global surrogate fitting and acquisition search difficult. Useful structure may still be local or involve only a few effective dimensions. TuRBO maintains local probabilistic models in trust regions and allocates evaluations across them (Eriksson et

al. 2019). SAASBO uses a sparsity prior to identify a small set of influential coordinate directions (Eriksson and Jankowiak 2021). These are different assumptions. A method suited to sparse coordinate effects may fail when the relevant structure is dense or poorly aligned with the chosen coordinates.

Multi-task Bayesian optimization transfers information between related optimization tasks through a joint model (Swersky, Snoek, and Adams 2013). For example, earlier workloads may inform service tuning on a new workload. We need to record the task relationship and check for negative transfer. A contextual variable describes conditions observed for a decision, such as workload or ambient temperature; it should not be treated as a controllable design variable unless it can actually be changed. When conditions drift, older evaluations may require a different model or reduced relevance.

Recent work also moves part of the search computation into prior training. PABBO learns both surrogate and acquisition behavior for preference-based optimization, reducing the inference work required during an interaction (Zhang et al. 2025). Its transfer to another application depends on the training-task distribution, the evaluator, and the available constraints. We count training and adaptation costs when comparing it with methods fitted only to the current task.

A service-tuning example

Suppose we tune worker count, cache capacity, and a categorical batching policy to reduce tail latency under a memory limit. Each load test returns a latency estimate, memory use, runtime, and completion status. Repeated tests use declared workload samples. We begin with current settings and a small feasible initial design, then compare random search with a method that represents the mixed configuration space. The objective is the chosen tail-latency statistic; uncertainty in its estimate is recorded separately.

Short tests can screen candidates only if their relationship to the full test is checked. Memory failures are retained as constraint observations, while a lost test worker is recorded as an evaluation failure. Pending runs remain visible to an asynchronous selector. Promising configurations receive repeated full tests, including held-out workloads. If improvement disappears under confirmation, we investigate workload variation, selection bias, or an inaccurate low-fidelity model before accepting the candidate.

We can use a similar sequence for materials experiments, manufacturing settings, and creative comparisons. In each case, we propose a candidate, evaluate it, retain the observations, and update the search. We need to change the evaluator and the rules for admissibility when moving to another domain. The response, the permitted experiment, and the conclusion supported by confirmation must also be defined for that domain.

Reading the optimization trace

We can analyze the evaluation history as a trace. Repeated identical proposals may indicate quantization or a collapsing search distribution. A favorable surrogate prediction followed by a

poor observation may indicate model error, noise, or changed conditions. Frequent timeouts may identify an expensive region or an infrastructure problem. A long plateau can result from a small feasible region, an ineffective acquisition optimizer, exhausted local variation, or an insufficient budget. These observations suggest hypotheses; none identifies a cause without further evidence.

The final report distinguishes the best observed value, the estimated response of the recommended candidate, verified feasibility, and any mathematical bound. It includes evaluation and wall-time budgets, model-fitting overhead, failed runs, repeats, fidelity costs, seeds, and untouched confirmation results. We can then state whether the result is a useful candidate, a statistically supported improvement, a local stationarity result, or a certified global bound. These conclusions should not be inferred from an algorithm name alone.

Hybrid search

We can combine methods when their operations address different parts of the problem. A heuristic can generate incumbents for branch-and-bound, a relaxation can guide a population method, and a local method can refine Bayesian proposals. Evolutionary outer loops can use gradient-based inner loops. A learned policy can propose branches while a conventional solver retains valid bounds and exhaustive search, preserving its correctness guarantees. Constraint propagation or an exact verifier can reject inadmissible moves; a human can select among feasible generated alternatives. For every interface, we state which feasibility and bounding guarantees survive the composition.

No-free-lunch results depend on specific averaging assumptions. Wolpert and Macready consider uniform averaging over all objective functions between finite sets (Wolpert and Macready 1997). The sharpening to function classes closed under permutations of the search domain is due to Schumacher, Vose, and Whitley (2001). In that setting, non-revisiting methods have equal aggregate performance. Practical problems can instead share smoothness, locality, sparsity, invariance, low effective dimension, generative regularity, or a characteristic distribution of instances. We need to state which structure a method assumes and test whether the target problems have it.

A surrogate has its own domain of validity. Record its training design, representation, calibration domain, uncertainty assumptions, and failure handling. Search can select candidates with favorable surrogate error, so use trust regions, uncertainty penalties, and adversarial or space-filling checks where appropriate. A proposal still needs confirmation by the target evaluator. Repeated low-fidelity measurements do not remove systematic approximation bias.

We can sometimes design a heuristic so that each move preserves a hard constraint. For example, a routing move can preserve capacity and continuity, and a scheduling move can preserve qualifications and precedence. A geometric mutation can remain within a manufacturable parameterization. Such moves reduce the need for repair and the number of evaluations spent on impossible candidates.

When feasibility cannot be preserved, separate violation by meaning. Safety, law, and physical impossibility should not share a generic penalty with preferences. Use lexicographic repair or a feasibility restoration phase. Report the rate and severity of violations and verify exported candidates independently.

Solver portfolios, comparison, and infeasibility

For consequential problems, we can compare a small portfolio containing a transparent baseline, a method that uses exact or convex structure, and an exploratory heuristic. Equalize problem definitions, stopping rules, preprocessing, tuning budgets, and implementation attention. Report hardware, wall time, energy, memory, feasibility rate, bounds, variability across seeds, and author effort separately from machine time. Preserve configurations and solver logs, and include representative and shifted instances.

We also need to allocate the evaluation budget among seeds, starting points, fidelities, and problem instances. A single long run shows one search trajectory. Multiple starts provide information about different basins, and repeated runs show stochastic variation. Different instances test whether performance transfers. Final evaluation at the target fidelity checks whether the proposed improvement remains valid.

We specify this allocation before choosing the preferred result and reserve a separate budget for final confirmation. Performance profiles or empirical cumulative distributions can describe differences across instances more clearly than one average. We report the time to the first feasible solution, time to a target value, best value, and failure rate.

Energy and memory can matter as much as runtime. A slightly better solution can be dominated by its search burden and delayed implementation.

Suppose a clinic schedules imaging appointments with different durations, preparation requirements, equipment, staff, urgency, and no-show risk. If the objective includes only unused capacity, the optimizer may overbook appointments or reduce recovery time. Patients and staff would then bear consequences that the objective does not represent.

A stronger formulation uses discrete assignment and start variables, precedence and equipment constraints, staff skills, cleaning and safety buffers, accessibility needs, and reserved urgent capacity. Objectives distinguish patient delay, overtime, schedule disruption, and unused capacity. Some requirements—clinical readiness, safe staffing, equipment compatibility, and urgent access—are constraints. The model retains slack because variability creates queues.

We can examine the scheduling problem using several representations. A time-indexed MILP accounts directly for capacity, although it may be large. Constraint programming represents intervals and resource conflicts. A two-stage stochastic model chooses a schedule and then repairs it after attendance or duration becomes known. Simulation can evaluate the policy under realistic arrival and recovery conditions. Comparing these representations helps us identify useful structure before further solver tuning.

The formulation must include the human workflow. A mathematically available ten-minute slot can be unusable if transport, consent, preparation, or interpretation is omitted. Staff review of rejected and extreme schedules is a validation method. If the optimizer persistently uses a “legal” pattern that staff refuse, the pattern is either an unmodeled burden or a governance conflict.

Benchmarking can itself overfit. Keep a final untouched instance set and include structurally shifted cases. A portfolio chosen after repeated testing on the same benchmark inherits the optimizer’s curse from Chapter 17. Report selection procedure and final evaluation separately.

Infeasibility can indicate conflicting requirements, incorrect data, unit errors, or an impossible target. We can investigate using irreducible inconsistent subsets, feasibility relaxation, dual certificates, and domain review. Constraint relaxation should follow this investigation and the relevant authority decision.

Classify constraints by authority and meaning, then ask which change is legitimate. A unit or data correction differs from relaxing a service promise; a design-variable expansion differs from purchasing away a right. The report should show the smallest conflicting sets and alternative ways to restore feasibility.

Sometimes we find that the requested requirements cannot all be satisfied. A proof of infeasibility can then support a review of priorities and capacity. We should report the conflict explicitly before replacing any requirement with a penalty.

Field checklist

- Map domain objects to machine operations; label every transformation.
- Use solver traces to assess formulation, feasibility, proof, and representation fit.
- Compare portfolios under equal budgets, seeds, and implementation effort; report bounds and guarantees.
- Challenge inductive bias with shifted, adversarial, and structurally different cases.

Translation lens. We can view structured optimization as a sequence of model transformations. For each step, we record which domain meanings, feasible decisions, bounds, and guarantees are preserved.

Chapter 6 · Uncertainty, multiple objectives, games, and hierarchy

Uncertainty, risk, and four attitudes to action

Under uncertainty, we need to specify how possible outcomes are compared. One approach is to minimize expected loss:

$$\min_x \ \mathbb{E}_\xi[f(x,\xi)] \tag{40}$$

is appropriate only when a probability model is credible and average performance captures the decision maker's concern. Rare harms, ambiguity, nonstationarity, and irreversibility require other criteria.

Stochastic programming

Two-stage stochastic programming separates a here-and-now decision x from recourse $y(\xi)$ after uncertainty is observed:

$$\min_x \ c^T x + \mathbb{E}_\xi[Q(x,\xi)], \tag{41}$$

where

$$Q(x,\xi) = \min_y\{q(\xi)^T y : W(\xi)y = h(\xi) - T(\xi)x,\ y \geq 0\}. \tag{42}$$

Scenario trees generalize to multiple stages but grow rapidly. Decomposition, sampling, and approximate dynamic programming manage this complexity (Shapiro, Dentcheva, and Ruszczyński 2021).

Chance constraints and risk measures

A chance constraint

$$\Pr(g(x,\xi) \leq 0) \geq 1 - \varepsilon \tag{43}$$

limits violation probability, but can be hard to compute and may hide the severity of violations. Value-at-Risk is a quantile; Conditional Value-at-Risk (CVaR) averages losses beyond that quantile. For loss $L(x,\xi)$ and confidence α,

$$\mathrm{CVaR}_\alpha(L) = \min_\eta \left[\eta + \frac{1}{1-\alpha}\mathbb{E}[(L-\eta)_+]\right]. \tag{44}$$

Here $(z)_+ = \max(z,0)$ is the positive part. The confidence level and the units of the loss must be specified; tail cost, mortality, and service delay do not become interchangeable merely because each admits a CVaR. This representation makes CVaR optimization tractable in many sampled problems (Rockafellar and Uryasev 2000).

Robust optimization

Robust optimization protects against an uncertainty set $\mathcal{U}$:

$$\min_x \ \max_{\xi\in\mathcal{U}} f(x,\xi). \tag{45}$$

We need to justify the uncertainty set. A small set may exclude relevant errors, whereas a very large set may produce a solution that is too conservative to use. Budgeted uncertainty allows us to vary the degree of protection and examine its cost (Ben-Tal, El Ghaoui, and Nemirovski 2009; Bertsimas and Sim 2004). Distributionally robust optimization instead uses a set $\mathcal{P}$ of plausible probability distributions:

$$\min_{x} \sup_{P \in \mathcal{P}} \mathbb{E}_P[f(x, \xi)]. \tag{46}$$

Ambiguity sets may be defined by moments, divergences, or optimal-transport distances.

Bayesian decision and value of information

Bayesian decision theory combines posterior uncertainty with utility. The expected value of perfect information compares the best expected consequence when uncertainty is revealed before action with the best expected consequence under present information. For sample information Y, first find the best expected consequence conditional on each possible Y, and then average over its predictive distribution. Subtract the best expected consequence without Y. Study cost is accounted for separately. A more precise estimate can have zero decision value when it leaves the preferred action unchanged.

Reliability and resilience

Reliability concerns the probability of performing as required under specified conditions. Resilience also includes recovery, adaptation, and continued operation with reduced performance when nominal assumptions fail. Maximizing efficiency may remove the spare capacity or diversity needed for these responses. We therefore often need to consider nominal performance together with tail risk, recovery, and the value of retaining future options.

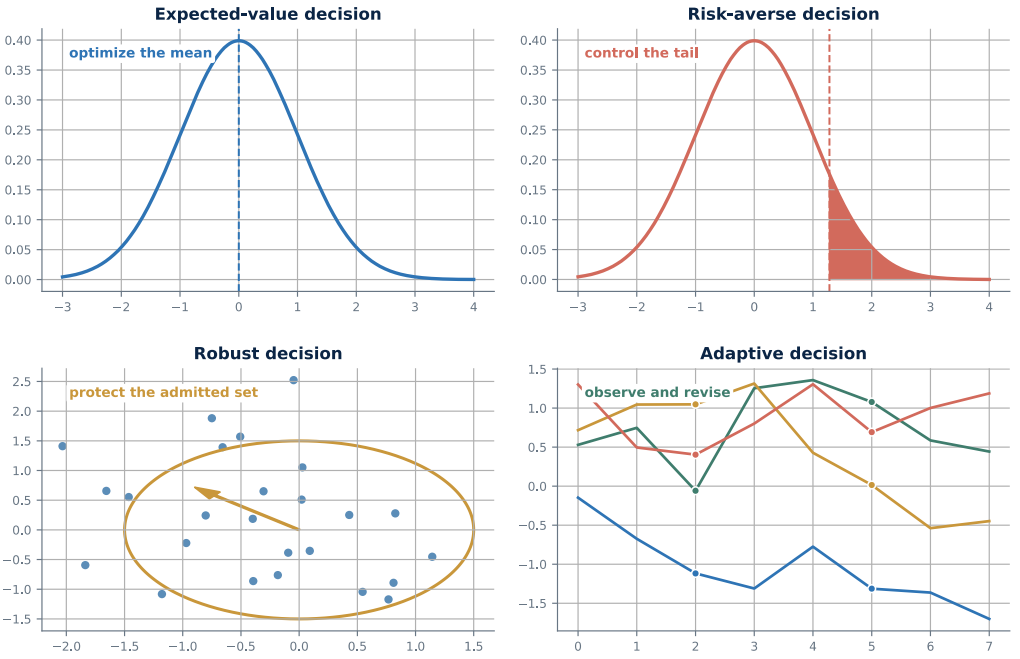


Figure 7. A 2 × 2 comparison of expected-value, risk-averse, robust, and adaptive decisions under uncertainty.

Figure 7 distinguishes four questions often blurred together: what is best on average, what limits tail loss, what survives an uncertainty set, and what can adapt after observation.

We can represent uncertainty by a probability distribution, a set of plausible parameters, an ambiguity set of distributions, or observations that will arrive later. These representations express different information. Their selection therefore precedes algorithm selection.

An expected-value formulation is appropriate when repeated outcomes, calibrated probabilities, and approximately linear consequence aggregation make the mean decision-relevant. A risk measure such as CVaR emphasizes a tail while retaining probabilistic structure. Robust optimization protects against every point in an admitted set, which is valuable for feasibility but can be overconservative. Distributionally robust optimization is a variant of the robust attitude: it protects against a family of distributions instead of a set of parameter realizations. Adaptive optimization reserves recourse so that actions can change after information arrives.

These formulations may select different actions from the same observations. For example, an expected-cost model of flood defense may favor a barrier suited to the central part of a climate distribution. A risk-averse model may provide more protection against severe outcomes. A robust model considers the full specified scenario set. An adaptive plan begins with one measure while retaining land, permits, and rules for later expansion. Its evaluation includes the possibility of changing the action when more information becomes available.

We need to calibrate uncertainty and ambiguity sets against the available evidence. Choosing a set only because it simplifies the equations does not establish protection from relevant hazards. Historical samples may omit structural changes, and expert scenarios may omit

correlated failures. We also test events outside the chosen set to examine the consequences of this modeling assumption.

Begin by asking what can be observed before action, what can be observed afterward, and what can never be known in time. This divides decisions into here-and-now commitments and recourse. A hospital may choose base staffing before demand, call additional staff after a forecast, and divert patients only after local capacity is observed. A static “robust schedule” can be needlessly conservative if recourse is available and unsafe if the recourse is infeasible.

Scenario trees represent staged revelation. Their branches should reflect conditional information, not merely sampled futures. If weather is observed before a transport decision, policies may depend on it. If a supply failure is discovered only after dispatch, they may not. Nonanticipativity constraints enforce this information structure.

A probability model can be used when frequencies or calibrated beliefs support it. A robust set can represent bounded errors against which feasibility is required, and an ambiguity set can represent uncertainty about an estimated distribution. Narrative scenarios can describe plausible mechanisms for which a probability is difficult to justify. We may use these representations together, for example, by combining expected operating cost, a tail-risk limit, robust safety constraints, and narrative stress tests.

Robustness should be tested by group and location. A policy with stable total performance can transfer variability to a small population. For each scenario, report group outcomes and constraint violations; then inspect which scenarios and groups drive the robust solution. An uncertainty set built from aggregate error can underrepresent sparse groups.

Resilience includes detection, fallback, mutual aid, recovery, compensation, and learning when conditions exceed the uncertainty model. Robustness concerns protection within that model. We distinguish these properties because a safety margin alone does not provide a recovery procedure.

Study design should target uncertainty near decision boundaries, active constraints, or scenario failures. Compare the expected value of information with delay, burden, and ethical cost. In urgent settings, act with a safe provisional policy while collecting information for revision.

We can also reduce dependence on current information by retaining options for later decisions. Examples include a modular design, reserved capacity, a reversible contract, a staged project, or a feedback policy. Their value comes from the ability to respond to future observations. A static formulation that omits these responses may underestimate flexibility and overestimate the value of a precise forecast.

Test whether the risk measure hides severity or frequency. A chance constraint limits violation probability and can ignore magnitude. Worst-case loss protects magnitude and can be driven by a dubious extreme. Entropic or variance penalties respond differently to distributions. Use several views and explain which consequence they protect.

For group harms, compute tails by group as well as population. An aggregate tail can be dominated by a large group and conceal systematic severe outcomes in a smaller one. If a harm is unacceptable, use a constraint or veto rather than relying on a risk weight.

We use the term deep uncertainty when participants cannot agree on a probability model, the consequences, or the future system. This often occurs in long-lived infrastructure and policy decisions. Assigning a precise distribution in such a case may conceal the disagreement in assumptions that readers cannot see.

Use exploratory modeling. Generate many plausible futures across physical, economic, behavioral, and institutional dimensions. Evaluate policies, identify vulnerability regions, and discover which assumptions cause failure. Seek robust or low-regret policies and adaptive triggers rather than one scenario-optimal plan.

Scenario diversity should include mechanisms, not only parameter grids. Correlated shocks, political delay, technology change, migration, and strategic response can alter structure. Narrative scenarios can expose conditions no smooth distribution represents.

A policy may perform acceptably across the modeled futures while remaining contested on distributional or rights grounds. We therefore present its vulnerabilities and trade-offs to the authorized decision process. Monitoring, resources, and funding need to be linked to adaptation triggers so that the planned response can be carried out.

Failure models often assume independence because it simplifies calculation. However, components may share a supplier, software, climate exposure, institution, dataset, or incentive. These dependencies can produce a common cause of failure, and their effect may increase during stress.

Map dependency explicitly. Use copulas or joint scenarios where data supports them, and stress structural common causes where it does not. Differently labeled assets may still share an unrecorded exposure. Redundancy should vary technology, location, control, and organization where appropriate.

After an event, update the dependency map rather than only marginal failure rates. A new common cause changes system architecture.

Multiple objectives and uncertain preferences

When consequences conflict, there may be no single best solution. For objectives $F(x) = \big(f_1(x), \dots, f_k(x)\big)$, a feasible point x dominates y for minimization when $f_i(x) \leq f_i(y)$ for all i and the inequality is strict for at least one component. A point is Pareto efficient if no feasible point dominates it.

The Pareto front makes trade-offs visible. It does not choose among them. Selection requires preferences, priorities, negotiation, or governance (Miettinen 1998; Deb 2001).

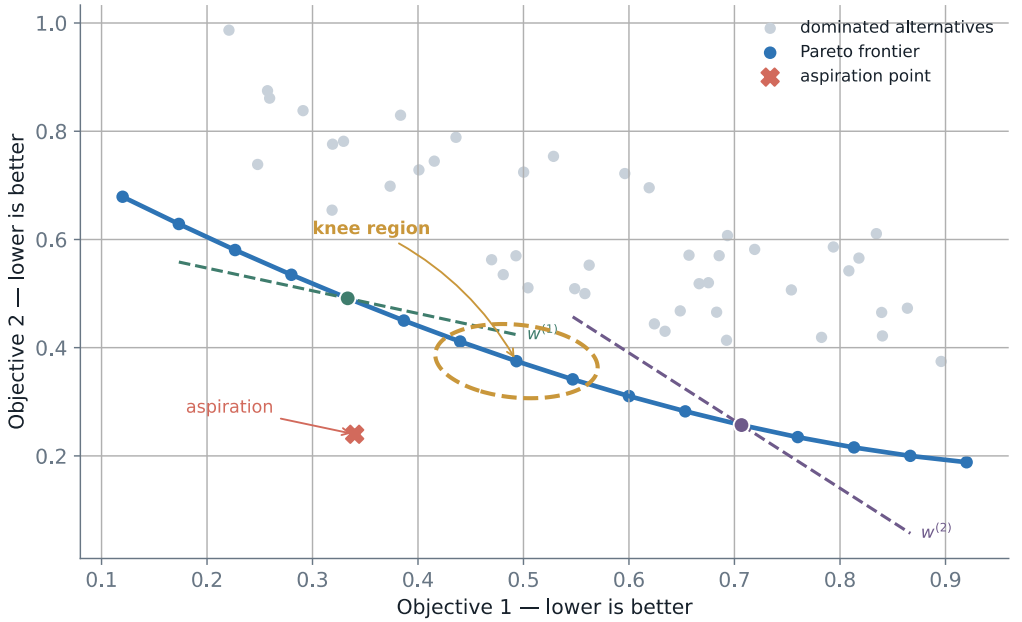


Figure 8. A Pareto front showing dominated designs, nondominated alternatives, a knee region, aspiration point, and the effect of changing scalarization weights.

Figure 8 shows that a Pareto frontier exposes trade-offs but does not choose for the decision maker; knees, aspirations, and scalarization slopes encode additional judgments.

Scalarization and preference articulation

Weighted sums are simple but scale-sensitive and incomplete for nonconvex fronts. The ε-constraint method optimizes one objective while bounding others. Reference-point methods minimize distance to aspirations. Lexicographic optimization protects priority order. Outranking methods can retain vetoes and incomparability. Interactive methods alternate optimization with preference feedback as people learn about the alternatives. Evolutionary multiobjective algorithms approximate diverse fronts when gradients or convexity are unavailable (Deb et al. 2002).

Games

In a game, each agent's result depends on other agents' choices. A Nash equilibrium x^* satisfies

$$u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*) \quad \text{for every player } i \text{ and action } x_i. \tag{47}$$

No player can profit by unilateral deviation. For every finite game, Nash's existence theorem guarantees at least one equilibrium in mixed strategies; a pure-strategy equilibrium need not exist. Equilibrium is not the same as collective optimality, fairness, or stability under learning. Mechanism design reverses the problem: choose the rules so that strategic behavior produces desirable outcomes (Nash 1950; von Neumann and Morgenstern 1944; Myerson 1981).

Bilevel optimization

Bilevel problems contain an optimization problem inside a constraint. The optimistic formulation lets the leader select a favorable follower optimum when several exist:

$$\min_{x,y} F(x,y) \quad \text{subject to} \quad y \in \arg\min_{z}\{f(x,z): g(x,z) \leq 0\}. \tag{48}$$

They model leader-follower games, pricing, hyperparameter tuning, adversarial learning, infrastructure planning, and policy response. A pessimistic formulation instead evaluates the follower optima least favorable to the leader. The tie-breaking assumption therefore belongs to the model. Even linear bilevel problems can be hard because the lower-level solution map is discontinuous or set-valued (Dempe 2002).

Social choice and incomplete order

Individual rankings cannot always be combined into a scalar social preference while satisfying the desired axioms. Arrow's theorem establishes an incompatibility among a particular set of such requirements (Arrow 1951). We therefore need to specify the collective decision procedure as well as the optimization problem. In some cases, deliberation or several criteria may produce an acceptable set of options without selecting a unique mathematical winner.

The frontier itself is uncertain. Estimated objectives, finite simulation, and model error can reorder candidates. Report confidence regions or dominance probabilities where possible. Avoid showing a thin deterministic curve when the evidence supports a band. A robustly nondominated option may be more useful than a nominal knee that disappears under small perturbations.

Preferences can be uncertain because stakeholders disagree, consequences are hard to compare, or alternatives reveal new values. Do not represent every form as a probability distribution over weights. Preserve intervals, vetoes, ordinal comparisons, or incomparability when they match the evidence.

Robust preference analysis asks which alternatives remain nondominated over a set of plausible weights or aspiration levels. A candidate chosen only under a narrow weight range needs a stronger justification than one that performs acceptably across many. Value-of-information analysis can ask whether another study would change the chosen region of the frontier rather than merely narrow estimates.

Deliberation should show the effect of moving a criterion from objective to constraint. This change is often more important than tuning a weight. A minimum service level, carbon budget, or rights condition changes which alternatives are admissible and expresses a different ethical structure.

Games, mechanisms, and strategic stress tests

Mechanism design treats rules as variables. For example, a matching system, auction, congestion charge, admissions policy, or platform ranking changes incentives and information

revelation. For a risk-neutral buyer and seller with independent private valuations and positive densities on intervals whose interiors overlap, ex post efficiency, Bayesian incentive compatibility, interim individual rationality, and budget balance cannot all be achieved (Myerson and Satterthwaite 1983). Privacy, simplicity, and distributional fairness add further design questions. Simulation with strategic agents can reveal exploits but does not prove their absence.

For a policy or platform, list actors, information, actions, payoffs, and adaptation speed. Then ask how each actor could improve their outcome under the rule. Test honest participation, gaming, collusion, withdrawal, and creation of new intermediaries. Include administrators and model owners, not only end users; institutions optimize reports too.

Equilibrium analysis can identify stable responses under strong assumptions. Agent-based or behavioral simulation can explore heterogeneous adaptation. Historical and qualitative evidence can reveal tactics the model omitted. None proves that the mechanism is strategy-proof in the world.

An undesirable equilibrium can indicate a need to revise the mechanism. If several equilibria exist, implementation history and coordination affect which one is reached. A rule that is efficient at one equilibrium may settle at another. We therefore record the equilibrium-selection assumptions.

A single interaction may be an inadequate model of an institution that operates repeatedly. Participants can learn, develop reputations, coordinate, retaliate, or invest in influence. The rule may also change who enters or leaves. We therefore include repeated incentives, information feedback, and participation costs when they affect the decision.

Robust mechanisms should be simple enough to understand, monitor, and appeal. Complexity can create strategic advantage for well-resourced actors even if formal rules are symmetric. Test not only equilibrium allocation but administrative error, communication, enforcement, and contestability.

When a mechanism fails, distinguish bad equilibrium, manipulation, infeasible enforcement, and illegitimate objective. Each calls for a different redesign.

An example of flood adaptation

We can organize a flood-adaptation example in three stages. First, generate a portfolio of infrastructure, nature-based measures, building codes, warning systems, insurance changes, and retreat options. Second, simulate them across sea-level, rainfall, economic, ecological, and demographic scenarios. Third, present a frontier of expected cost, severe loss, ecological effect, displacement, and option value to the accountable decision makers.

Legal rights, minimum safety, protected habitats, and procedural requirements remain hard constraints in this example. Other criteria are represented as uncertain objectives. The model identifies dominated portfolios and important sensitivities, and the authorized decision process selects an adaptive plan. The policy also specifies monitoring conditions. Observed

sea level, maintenance condition, population exposure, or distributional harm may trigger the next stage of action.

Field checklist

- Identify whether uncertainty is probabilistic, set-valued, distributional, adversarial, or sequential.
- Calibrate uncertainty objects and stress events outside their boundaries.
- Preserve a frontier long enough for preferences and disagreement to remain visible.
- Show uncertainty around the frontier, not only uncertainty inside each simulation.
- State whether strategic agents, equilibrium selection, and tie-breaking are modeled.
- Treat recourse, reversibility, and option value as design variables.
- Keep rights and vetoes out of compensatory scalarizations unless their status is explicitly changed.

Translation lens. Stochastic and robust models, multiobjective orderings, games, and bilevel problems describe different decision structures. When combining them, we preserve uncertainty type, incomparable preferences, strategic responses, and decision authority.

Chapter 7 · Dynamics, transport, category theory, algorithms, and verification

Decisions through time

In dynamic optimization, actions change a state and thereby affect later consequences. Let

$$s_{t+1} = T(s_t, a_t, w_t) \tag{49}$$

describe a system with state s_t, action a_t, and disturbance w_t. A policy π maps observations to actions. The objective may discount future costs:

$$J^{\pi}(s_0) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \, c(s_t, a_t)], 0 \leq \gamma < 1, |c(s,a)| \leq C < \infty. \tag{50}$$

Bellman's principle

Under the assumptions stated with Equation (50), the optimal value function satisfies

$$V^*(s) = \min_a [c(s,a) + \gamma \, \mathbb{E}[V^*(s') \mid s, a]]. \tag{51}$$

This recurrence is used in dynamic programming, Markov decision processes, and much reinforcement learning (Bellman 1957; Puterman 1994; Sutton and Barto 2018). Its direct computation can become difficult because the number of states may grow exponentially. This is commonly called the curse of dimensionality.

Optimal control

For continuous dynamics $\dot{x} = f(x, u, t)$ with terminal cost Φ and running cost L, define the objective

$$J[u] = \Phi\big(x(T)\big) + \int_0^T L\,(x(t), u(t), t)\, dt. \tag{52}$$

Pontryagin's maximum principle introduces a costate λ and Hamiltonian

$$H(x, u, \lambda, t) = L(x, u, t) + \lambda^T f(x, u, t). \tag{53}$$

With the cost convention in Equation (53), the optimal control minimizes the Hamiltonian pointwise; equivalently, it maximizes its negative. Necessary conditions couple state, costate, and control. The Hamilton-Jacobi-Bellman equation gives the dynamic-programming description. Linear-quadratic control supplies analytic structure; nonlinear constrained control usually requires numerical methods (Pontryagin et al. 1962; Bryson and Ho 1975).

Model predictive control

Model predictive control repeatedly solves a finite-horizon problem, implements its first action, observes the new state, and solves again. This replanning supplies feedback and handles constraints. Stability and continued feasibility require suitable terminal assumptions, feasible fallback behavior, and computational deadlines (Rawlings, Mayne, and Diehl 2017).

Reinforcement learning

Reinforcement learning estimates values or policies from interaction. A model-free method does not construct an explicit transition model, but its result still depends on exploration, reward design, observability, stationarity, and data coverage. If the reward represents the wrong behavior, successful optimization may increase that behavior. In offline reinforcement learning, the policy may also choose actions for which historical data provide little support.

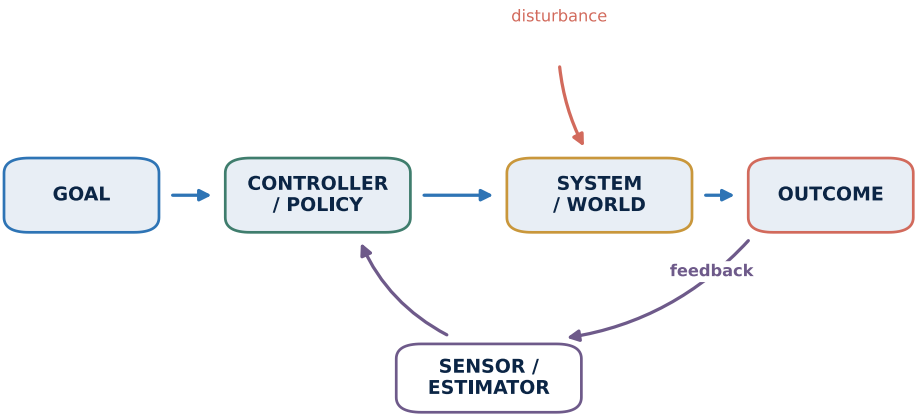


Figure 9. Closed loop from goal through controller or policy, system or world, and outcome, with disturbance and sensor or estimator feedback.

Figure 9 shows optimization within a feedback system. The policy selects an action for the plant, and an estimator uses observations to reconstruct the state. The observed consequences can also lead to changes in the model and objective.

Partial observability and belief states

When the state is hidden, observations update a belief distribution. The belief is sufficient in theory but often high-dimensional. Filtering, state estimation, memory, and active sensing become coupled optimization problems. Kalman filtering provides an optimal linear-Gaussian estimator under its assumptions (Kalman 1960).

Dynamic optimization replaces a point with a trajectory or policy. The state must contain enough information to evaluate future consequences; the action changes both immediate cost and later opportunity. Bellman's principle expresses this recursively, but the practical challenge is constructing a valid state. Omitting relevant history can invalidate the Markov assumption. Retaining too much history can make computation intractable.

Using ξ_t for the disturbance denoted by w_t in Equation (49), for a controlled system $s_{t+1} = T(s_t, a_t, \xi_t)$, a finite-horizon value function satisfies

$$V_t(s) = \min_{a \in \mathcal{A}(s)} \mathbb{E}\big[c_t(s,a,\xi_t) + V_{t+1}\big(T(s,a,\xi_t)\big)\big]. \tag{54}$$

Equation (54) also specifies an interface between the current decision and future value: the future is summarized by V_{t+1}. Dynamic programming, model predictive control, and reinforcement learning differ in how they represent or learn that interface. Exact tables are possible for small states; approximate value functions, policy gradients, tree search, and fitted models trade bias, variance, and computation at larger scales (Bellman 1957; Puterman 1994; Sutton and Barto 2018).

An open-loop plan must survive uncertainty without observing it. A feedback policy can adapt but depends on sensors, estimators, communication, and actuation. Delay, saturation, partial observability, and human override can destabilize a policy that is optimal in an ideal simulator (Åström and Murray 2008).

A state is sufficient only relative to a prediction and decision horizon. In maintenance, current sensor values may predict immediate failure while maintenance history and environment are needed for long-term planning. In education, a score history may predict a test and omit the relationships and opportunities an intervention changes. State construction is therefore a claim that histories mapped to the same state are interchangeable for the future consequences being optimized.

Test the claim by searching for histories with the same encoded state and different outcomes or optimal actions. Include delayed effects, path dependence, accumulated exposure, and strategic memory. If the representation fails, enrich the state, shorten the claim horizon, use a partially observed model, or retain human context at the interface.

Approximate state can still be useful. The report should state the information loss and how the policy detects when it matters. A compact representation with a safe fallback may be preferable to a high-dimensional state that cannot be validated.

Consider a building-energy system. A weather and occupancy forecaster sends distributions to a scheduler; the scheduler sends temperature and power trajectories to local controllers; controllers act on equipment and return measurements.

The forecast interface specifies units, spatial and temporal resolution, quantiles or scenarios, calibration window, issue time, missing-data behavior, and conditions outside support. The scheduling interface specifies targets, comfort and safety envelopes, flexibility, price or carbon signals, and priority when constraints conflict. The control interface specifies admissible set points, ramp limits, timing, acknowledgment, local safety override, and fallback.

Each interface can omit information needed by the next component. For example, a point forecast omits uncertainty. A fixed trajectory may prevent a controller from adjusting to a local disturbance, even when such an adjustment would improve the overall result. If override events are not reported, the scheduler may incorrectly assume that all its requested actions were executed. Scenario sets, value functions, feasible response ranges, and status information can help retain the information required for composition.

We therefore label the arrows in a component diagram with their interface types. The component name identifies an operation; the arrow specifies the contract required for composition.

Transport and geometry as semantics

Optimization is shaped by the geometry of its space. Euclidean vectors are only one case. Probabilities, rotations, graphs, shapes, strings, and equivalence classes require metrics and topologies that respect their structure.

Optimal transport

Given probability measures μ and ν and a cost $c(x, y)$, Kantorovich's formulation seeks a coupling γ:

$$\min_{\gamma \in \Pi(\mu,\nu)} \int c(x, y)\, d\gamma(x, y). \tag{55}$$

For $p \geq 1$ and probability measures with finite pth moments, with $c(x, y) = d(x, y)^p$, the optimum defines the p-Wasserstein distance after taking the pth root. Unlike pointwise divergences, transport respects the ground geometry: nearby mass is cheaper to move than distant mass (Villani 2003; Peyré and Cuturi 2019).

Entropic regularization adds

$$\varepsilon \, \mathrm{KL}(\gamma \parallel \mu \otimes \nu) \tag{56}$$

and enables fast Sinkhorn iterations (Cuturi 2013). The regularization smooths and accelerates but also changes the solution; ε is a modeling as well as numerical parameter.

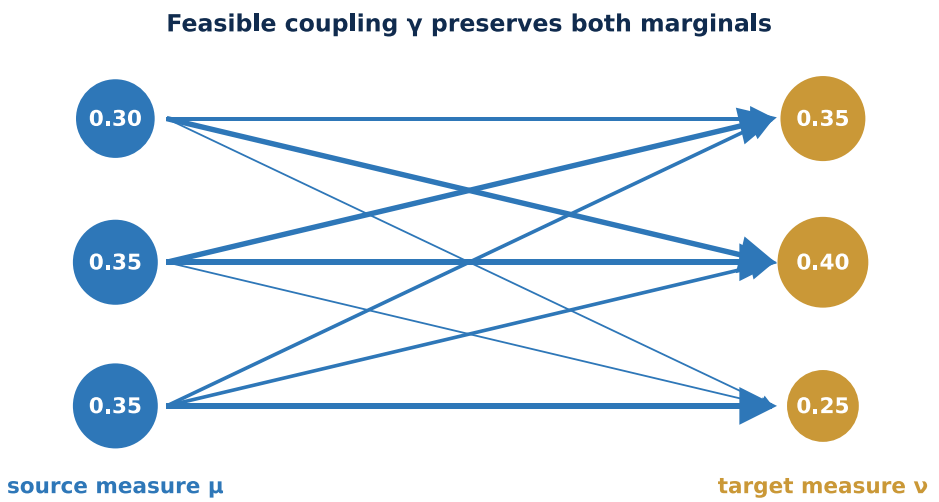


Figure 10. Transport coupling between three source and three target masses; arrow thickness shows moved mass.

Figure 10 illustrates transport relative to a chosen ground cost. Feasible couplings move the same source mass to the required targets but may have different total costs. The

unregularized optimum minimizes that cost; entropic regularization can spread the transport across more connections.

Manifolds and quotient spaces

Rotations lie on $SO(3)$, covariance matrices form a positive-semidefinite cone, whose positive-definite interior admits smooth manifold descriptions, and low-rank factorizations have redundant representations. Optimization on manifolds respects constraints intrinsically rather than repeatedly projecting from an ambient space. Quotient geometry identifies parameter values that encode the same object, reducing artificial symmetries.

Topology and shape

Topology distinguishes connectivity, holes, and continuity properties that survive deformation. Topology optimization in engineering asks where material should exist, but topological data analysis also turns persistent features into objectives or constraints. Shape optimization varies boundaries; level-set methods represent them implicitly. These problems often require regularization because unconstrained fine-scale structure exploits the discretization.

Metrics as semantics

A metric says which changes are small. Edit distance, perceptual distance, geodesic distance, Wasserstein distance, and task loss induce different neighborhoods and therefore different local optima. Metric learning and representation learning make the geometry itself a learned object. If the metric omits a distinction relevant to stakeholders, the resulting search will not account for it.

Transport applications include distribution comparison in machine learning, physical flow in logistics, and shape alignment in imaging. The ground cost specifies which transformations are small or large and needs a justification in each application.

More generally, we can consider geometry as a description of variation. A quotient space groups equivalent representations, a manifold expresses constrained coordinates, and a graph metric measures paths through relations. Information divergences express statistical differences. These choices affect both the algorithm and the interpretation of its result. For example, a default Euclidean norm may assign equal distances to changes that have very different consequences.

In optimal transport, the ground cost determines which movement is expensive. For images, Euclidean pixel distance may be physically meaningful; for language, embedding distance may merge antonyms or historical meanings; for social policy, geographic distance can ignore borders, disability, safety, or time. The transport plan is optimal relative to this cost.

Before interpreting a transport distance, test invariance and counterexamples. Which transformations should have zero or small cost? Which distinctions must remain large? Does

the cost obey symmetry or triangle inequality, and does the domain require those properties? A nonmetric cost may be correct if direction or irreversible harm matters.

Entropic regularization improves computation and smooths couplings. It also spreads mass, which may be a useful uncertainty model or an artifact. Report the regularization level and test whether substantive conclusions persist as it changes.

Category theory and composition

Category theory asks how structures and transformations compose. This is especially valuable when optimization concerns systems built from subsystems, each with its own variables, constraints, objectives, and interfaces.

A category of open optimization problems

Consider an open component with input interface X, output interface Y, internal decision z, feasibility relation $R \subseteq X \times Z \times Y$, and cost $c: X \times Z \times Y \to \overline{\mathbb{R}}$. It can be represented schematically as a morphism

$$P: X \to Y. \tag{57}$$

Sequential composition connects the output of one component to the input of another. Parallel composition uses a monoidal product:

$$(P_1 \otimes P_2): (X_1 \otimes X_2) \to (Y_1 \otimes Y_2). \tag{58}$$

The total cost might add, take a maximum, or combine in another monoidal way. The choice of monoid is part of the semantics. Decorated cospans and related constructions provide formal languages for open networks whose internal decorations may be circuits, dynamical systems, resource requirements, or optimization data (Fong 2015; Fong and Spivak 2019).

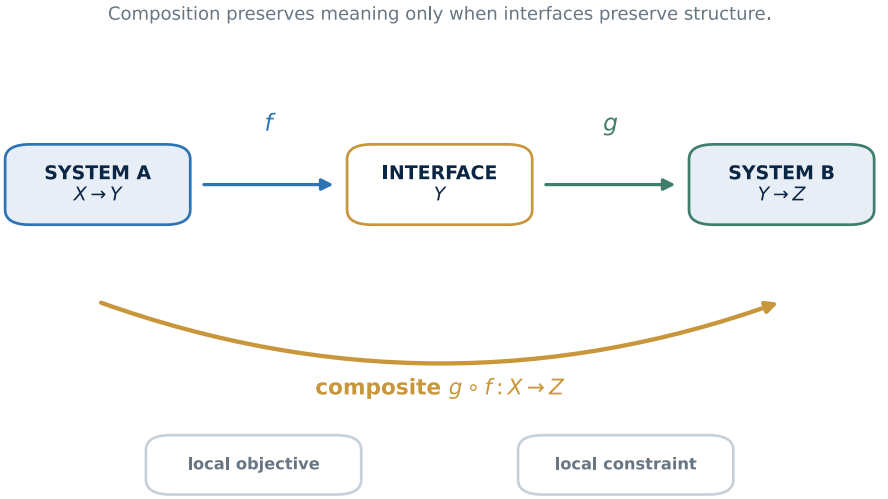


Figure 11. Sequential composition of SYSTEM A and SYSTEM B through an interface, with f, g, g ∘ f, a local objective, and a constraint.

Figure 11 uses f and g as generic component morphisms; they play the roles denoted by P_1 and P_2 here, rather than objective and constraint functions. It illustrates sequential composition. Compatible interface types are necessary for connecting components; system guarantees additionally depend on the treatment of hidden states, costs, and constraints. The same interface checks are needed when arranging components in parallel.

Why argmin is not generally a functor

Mapping each problem to a selected solution need not preserve composition. A small perturbation can change a selected minimizer abruptly. More fundamentally, a locally suboptimal interface value can permit a larger improvement downstream. Tied local selections can also be incompatible. A set-valued correspondence retains tied alternatives:

$$\operatorname{Argmin}(P) \subseteq \mathcal{X}_P, \tag{59}$$

possibly equipped with order, topology, probability, or approximation structure. Changing the codomain to relations, ordered sets, or probabilized sets may support a functor under additional assumptions. A set of local minimizers alone does not preserve all alternatives needed downstream.

Enrichment and cost

Lawvere described generalized metric spaces as categories enriched over nonnegative extended reals (Lawvere 1973). Composition records the triangle inequality, and the monoidal operation combines costs. This technical meaning of enrichment retains quantitative magnitude in categorical descriptions of shortest paths, resource accounting, and semantics.

Adjunctions and duality

Adjunctions formalize optimal translations between descriptions. Galois connections link abstraction and concretization; convex conjugacy links primal and dual functions; free and forgetful constructions trade structure against universality. These are not all the same adjunction, but they share a pattern: one transformation is the best approximation compatible with another.

Compositional games and co-design

Open games make strategic interactions composable as morphisms in a symmetric monoidal category (Ghani et al. 2018). Co-design problems represent components by feasibility or resource relations and compose them along interfaces. String diagrams expose information flow, feedback, and parallelism. This allows us to examine whether separately optimized modules have compatible objectives and feasible interfaces.

Sheaves and local-to-global consistency

In distributed systems, local decisions may agree on overlaps yet fail to form a global solution. Sheaf-like structures organize local data and compatibility. This is useful for sensor fusion, network constraints, collaborative design, and multi-view evidence. We can therefore examine global feasibility as a gluing problem that includes compatibility on shared interfaces.

A practical categorical checklist

1. Identify objects as typed interfaces, not only internal models.
2. Specify sequential and parallel composition.
3. State how costs and constraints compose.
4. Preserve provenance through transformations.
5. Treat feedback explicitly.
6. Check whether local solutions glue to a global one.
7. Record what structure each translation forgets.

Local argmin sets do not generally compose, so we pass feasible relations and value functions before selecting a solution. Dual messages or contracts can compress this information when their assumptions hold. Passing the entire local argmin set still discards locally nonoptimal choices that a downstream component may need.

This observation helps determine what a component should expose. If subsystem A exports only its selected point, subsystem B cannot compare alternatives. A response curve or value function retains that comparison. When privacy or organizational boundaries prevent sharing internal models, dual prices or contracts may provide a compressed interface, provided their assumptions are recorded.

Suppose we combine a calibrated predictor, an optimal scheduler, and a stable controller. We still need to examine the assembled system. The scheduler may change the input distribution on which calibration was established. Its target may lie outside the controller's stability

region. Communication delay may violate the assumptions of both components, and human overrides may introduce another mode of operation. None of these effects is excluded by the individual component guarantees.

Write assurance as assume-guarantee contracts. Each component lists assumptions about inputs and environment and guarantees under them. Composition is valid when upstream guarantees satisfy downstream assumptions and feedback does not invalidate upstream conditions. Then test common-cause failures and modes outside every contract.

Category theory describes composition, engineering supplies quantitative contracts, and governance assigns authority when a contract fails. The system needs these contributions together.

An example of predictor, scheduler, and controller composition

A predictor maps a history $h \in H$ to a forecast distribution $\mu \in \mathcal{D}$. A scheduler needs both the forecast and a capacity state $s \in S$ to select a target $u \in U$. A controller uses this target and the plant state $q \in Q$ to select an action $a \in A$. The predictor interface must therefore pass the capacity state through unchanged. Units, distributional information, and observation freshness are also parts of these interface types.

Represent each component by a feasible relation and a value function, assigning cost $+\infty$ to infeasible boundary pairs. For $P_1 : X \to Y$ and $P_2 : Y \to Z$ with additive costs, sequential composition eliminates the shared interface y. Here Z is an output interface, distinct from the internal-decision space used in the earlier schematic relation:

$$V_{P_2 \circ P_1}(x, z) = \inf_y \left[V_{P_1}(x, y) + V_{P_2}(y, z) \right] \quad \text{(59a)}$$

For the predictor and scheduler, $x = (h, s)$, $y = (\mu, s)$, and $z = u$. The feasible relation requires the same capacity state on both sides of the predictor. When adding the controller, we similarly retain the plant state q until its use. Equation (59a) combines compatible costs before eliminating an interface. It describes total downstream value, rather than separate local selection.

Consider two candidate forecasts μ_1 and μ_2 with local predictor losses 0 and 1. Suppose their best scheduler-controller costs are 10 and 0, respectively. The predictor alone selects μ_1, giving total cost 10. If we include the other components before selecting, we choose μ_2, giving total cost 1. This example shows why composition of feasible relations and value functions can give a different result from composition of separately selected solutions.

Composition and selection. With matching interface types, exact feasibility relations, and all relevant additive costs included, we compose value functions by Equation (59a). We then derive the composite argmin from the composed value function. An attained infimum permits an actual minimizing choice. Unique minimizers or compatible tie-

breaking permit consistent recovery of a minimizing tuple. None of these conditions licenses composition of independently computed local argmin sets.

Proof sketch. Nested infima over the forecast and control interfaces equal the joint infimum over their feasible pairs. This establishes associativity of value-function composition. If the infimum is attained, a minimizing tuple can be recovered after composition. Independent local minimization can remove this tuple before the downstream costs are considered, as the two-forecast example shows.

These conditions give us a way to check the design. If composition fails, we investigate whether an interface omitted a state, cost, or authority requirement. At the system boundary, we also test distribution shift, stability regions, latency, uncertainty, and feedback.

We can use these conditions to design component interfaces. A component may hide implementation while exporting the structures its neighbors need: types, feasibility, value, uncertainty, provenance, and assume-guarantee conditions. If an interface omits one of these structures, the decision to omit it requires justification (Fong and Spivak 2019; Ghani et al. 2018).

Algorithms and verification across the stack

An optimization method is an algorithm only when its inputs, outputs, stopping conditions, resource model, and failure states are specified. The same mathematical iteration can behave differently under finite precision, parallel execution, noisy evaluation, and imperfect derivatives.

Convergence and rates

Convergence asks whether iterates approach an appropriate solution concept. Rates ask how fast. For an L-smooth convex function, gradient descent with suitable step size obtains an objective gap of order $O(1/k)$. Strong convexity yields linear convergence. Accelerated first-order methods can achieve $O(1/k^2)$ for smooth convex problems. Stochastic gradients often yield rates in expectation that trade bias, variance, batch size, and step schedule (Bubeck 2015; Robbins and Monro 1951).

Worst-case rates are not runtime predictions. Constants, conditioning, memory traffic, parallelism, and stopping accuracy matter. Benchmarking should report wall time, evaluations, energy where relevant, and quality as a function of budget.

Complexity and approximation

Complexity classifies scaling with input size and representation. A pseudo-polynomial algorithm is not polynomial in encoded input length. Continuous problems require an accuracy parameter ε. Oracle complexity counts function, gradient, or Hessian queries. Approximation algorithms guarantee a ratio or additive error; randomized algorithms state

probabilities. Practical tractability can come from sparsity, low treewidth, convexity, fixed parameters, or benign data distributions even when the general problem is hard.

Sensitivity and conditioning

Sensitivity asks how the solution changes when data change. If $x^*(\theta)$ solves a parameterized problem, derivatives such as $dx^*/d\theta$ support design, inference, and bilevel learning. The envelope theorem often differentiates the optimal value without differentiating the optimizer. Ill-conditioning means small perturbations can create large changes; an unstable optimum may be mathematically correct and operationally unusable.

Automatic differentiation and adjoints

Forward-mode automatic differentiation propagates directional derivatives; reverse mode propagates sensitivities from outputs to many inputs. Reverse mode is the computational basis of backpropagation and adjoint methods. Differentiating through an optimizer can use unrolled iterations or implicit differentiation of optimality conditions. Each choice has memory, accuracy, and stability trade-offs.

Verification

The following six categories describe the evidence to collect.

- **Implementation:** derivative checks, unit tests, invariant checks, deterministic seeds.
- **Numerics:** residuals, feasibility tolerances, conditioning, precision sensitivity.
- **Optimality:** primal-dual gaps, KKT residuals, bounds, approximation guarantees.
- **Robustness:** perturbation, scenario, adversarial, and out-of-distribution tests.
- **Model:** calibration, causal evidence, physical conservation, domain review.
- **Decision:** stakeholder review, alternatives, distributional impact, reversibility.

The verification checks can also be organized by five system levels. At the mathematical layer, check feasibility, residuals, bounds, and convergence. At the implementation layer, test gradients, sparse indexing, units, and solver configuration. At the component interface, test contracts, ranges, latency, and failure behavior. At the system layer, simulate closed-loop scenarios, strategic response, and common-cause failures. At the decision layer, validate consequences and governance.

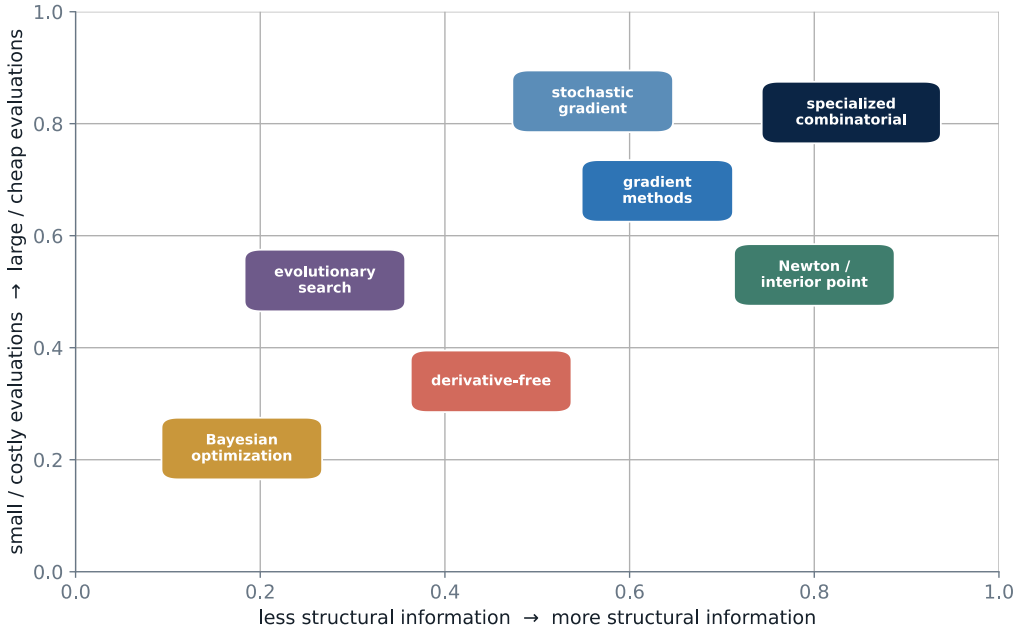


Figure 12. Algorithm families positioned by available structural information and by evaluation scale and cost.

Figure 12 relates algorithm families to available structural information and evaluation cost. After selecting a method, we still need to examine the candidate, any certificate it produces, and the evidence supporting its application.

Reproducibility

A reproducible optimization result includes data and preprocessing versions, mathematical formulation, solver and version, parameter settings, hardware, random seeds, stopping criteria, logs, and enough intermediate evidence to reconstruct the claim. For commercial or sensitive systems, reproducibility may occur in a controlled audit environment rather than through public release.

Reproducing a computation can also reproduce an error. Independent checks of the model, implementation, and interpretation remain necessary. A certificate should contain enough information to establish its stated claim and remain practical to check.

Field checklist

- Define a state that is sufficient for prediction and decision, then test that assumption.
- Distinguish open-loop plans from feedback policies and model delay, saturation, and fallback.
- Treat the metric or ground cost as semantics, not as a neutral default.
- At every interface, state variables, units, admissible ranges, uncertainty, and guarantees.
- Delay local selection when downstream components need alternatives or value functions.
- Verify mathematical, implementation, interface, system, and decision layers separately.
- Reproduce the pipeline and independently challenge the formulation.

Translation lens. Dynamic methods select trajectories or policies, transport methods compare transformation costs, and category theory describes composition through typed interfaces. In each case, we need to preserve information required by subsequent decisions. Passing only a locally selected answer may discard that information.

Part III · Computing and Engineered Systems

Chapter 8 · Algorithms, computing systems, software, and cloud platforms

Algorithms, representations, and resource models

We can implement an optimization method as a sequence of operations subject to time, memory, communication, energy, precision, and development budgets. These costs depend on the representation and execution environment. A short mathematical description may require an expensive implementation, while an asymptotically efficient method may be slower at the sizes used in practice. Fragility and difficulty of inspection can also limit an otherwise fast method.

Complexity as a resource model

For input size n, asymptotic analysis studies growth such as $T(n) = O(n\log n)$. It deliberately suppresses constants and machine details, which makes it portable but incomplete. A fuller cost model might be

$$C(A,n) = w_t T(A,n) + w_m M(A,n) + w_i I(A,n) + w_e E(A,n), \tag{60}$$

where T is time, M memory, I input/output, and E energy. The weights depend on the deployment environment. External-memory algorithms optimize transfers between storage levels; cache-oblivious algorithms seek good locality without hard-coding cache parameters (Frigo et al. 1999); parallel algorithms trade total work against span, synchronization, and communication (Cormen et al. 2022; Knuth 1997).

We can analyze an algorithm in the worst case, on average, over a sequence of operations, under small perturbations, or against an offline comparator. These analyses answer different questions. For example, expected constant-time lookup in a hash table does not exclude adversarial collisions. An occasional expensive resize of a dynamic array can coexist with low amortized cost. An online algorithm may be compared with an offline optimum that has all the input in advance. We choose the analysis according to the claim we need to examine.

Data structures and assumptions about operations

A data structure supports a particular distribution of operation costs. Balanced trees favor ordered updates and queries; hash tables favor equality lookup; heaps favor repeated extremum extraction; union-find favors incremental connectivity; sketches exchange exactness for compactness. Columnar layouts favor scans and compression; indexes accelerate reads while adding storage and write work. The right structure depends on the operation distribution, not merely the data type.

The representation also affects search. Dynamic programming stores results for repeated subproblems, although the required state space may grow exponentially. A greedy method commits to a choice at each step and needs an appropriate exchange property or similar

structure for an exactness guarantee. Divide-and-conquer can expose parallel work, while branch-and-bound combines search with bounds. Randomization can simplify a method or reduce its sensitivity to particular inputs (Kleinberg and Tardos 2005).

Approximation, streaming, and learning-augmented methods

When exact computation is too costly, one can optimize the guarantee. For a nonnegative objective, an approximation algorithm for minimization may promise

$$f(\hat{x}) \leq \rho f(x^*) \tag{61}$$

for ratio $\rho \geq 1$. Streaming algorithms control memory and passes. Fixed-parameter algorithms isolate a small structural parameter. Learned indexes and learned heuristics exploit observed distributions, but need fallback guarantees when the distribution shifts.

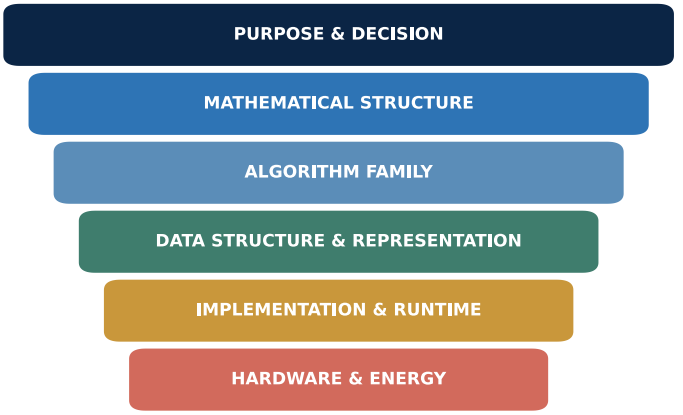


Figure 13. Six-layer stack from purpose and decision to hardware and energy.

Figure 13 connects an abstract algorithm with its implementation and physical resource use. Memory, communication, parallelism, and energy help explain why an asymptotic complexity result does not directly predict execution time on a machine.

Engineering the crossover point

Algorithm choice should be benchmarked across the expected input distribution. A theoretically inferior method may win below a crossover size because of vectorization, locality, or setup cost. A microbenchmark can hide allocation, cold caches, data skew, or tail latency. The measurement protocol below makes these conditions explicit.

We can examine computing optimization at five scales: the abstract algorithm, the implementation, the machine, the service, and the user journey. Each scale has a different cost model. Big-O notation suppresses constants and hardware details to describe growth; a

profiler measures one implementation and workload; a hardware counter exposes cache misses, branches, or vector utilization; a service trace exposes queues, retries, fan-out, tail latency, and failure. A user journey includes client behavior, retries, and human waiting across services. When describing an algorithm as faster, we therefore specify the scale and resource model.

An improvement at one level may increase cost at another. For example, fewer arithmetic operations can still require more memory transfers and a longer runtime. Compression saves bandwidth but uses CPU time. Parallel execution may shorten average completion time while increasing coordination and variation in tail latency. Caching improves common requests but introduces invalidation work. We therefore consider several resources together:

$$R(x) = \big(T_{50}(x), T_{99}(x), M(x), B(x), E(x), C(x), A(x)\big), \tag{62}$$

where the components may represent median and tail latency, memory, bandwidth, energy, monetary cost, and availability. Equation (62) invites Pareto analysis and explicit service constraints. It also prevents a local microbenchmark from silently standing in for user experience.

Systems, queues, and cloud capacity

A computer system is a queueing network governed by feedback. Requests compete for processors, memory, storage, accelerators, network links, locks, and human operators. Local utilization gains can worsen global latency, and average performance can conceal catastrophic tails.

Throughput, latency, and queues

Little's law relates average number in a stable system L, arrival rate λ, and average time W:

$$L = \lambda W. \tag{63}$$

It is a conservation law, not a scheduling policy (Little 1961). Near capacity, variability, bursts, and dependencies can produce sharply increasing delay. High average utilization can coexist with long tails and slow recovery. Batching improves throughput but adds waiting. Replication reduces read latency and increases availability while creating consistency and coordination costs. Backpressure protects downstream components but can propagate upstream.

Tail latency matters because a fan-out request waits for slow children. If n independent components each meet a latency target with probability p, the probability that all do so is p^n. Hedged requests, timeouts, load balancing, admission control, and redundancy trade extra work for more predictable service (Dean and Barroso 2013).

Parallelism and the memory wall

Amdahl's law estimates speedup when fraction s is serial and $1 - s$ is accelerated on p processors; here p is a processor count, whereas it denotes a probability in the preceding fan-out example:

$$S(p) = \frac{1}{s+(1-s)/p}. \tag{64}$$

Gustafson's law instead asks how problem size can scale with available processors (Amdahl 1967; Gustafson 1988). Both are idealizations. Real scaling is limited by memory bandwidth, communication, load imbalance, synchronization, and energy. Roofline-style reasoning compares arithmetic intensity with compute and bandwidth ceilings.

Databases and distributed state

Database optimization chooses physical plans for a declarative query. Join order, access paths, cardinality estimates, partitioning, caching, and materialization interact combinatorially. A wrong selectivity estimate can dominate an otherwise sophisticated cost model. Adaptive execution treats the plan as a policy that can change after observing runtime evidence.

Distributed data introduces consistency choices. Stronger ordering simplifies application reasoning but may increase coordination and reduce availability under partition. Eventual consistency reduces coordination but shifts reconciliation to data types and applications. We choose the required consistency properties according to workload and the consequences of inconsistency.

The same holds for distributed state. Strong consistency, causal consistency, eventual convergence, partition tolerance, availability, durability, and latency form a design space. A database schema embeds anticipated joins and invariants. A partition key embeds an expected access pattern. When the workload changes, a previously good representation becomes technical debt. Optimization should therefore include migration and observability, not only initial performance.

Cloud scheduling and reliability

Cloud platforms place workloads across heterogeneous machines while controlling cost, carbon, locality, deadlines, and failure domains. A simplified scheduler may solve

$$\min_{x} \; C_{\text{money}}(x) + \lambda C_{\text{latency}}(x) + \mu C_{\text{carbon}}(x) \tag{65}$$

subject to capacity, affinity, security, and service-level constraints. Here λ and μ are objective weights, distinct from queueing arrival and service rates. Autoscaling is a feedback controller; if delayed or overly aggressive, it oscillates. Reliability engineering uses redundancy, graceful degradation, fault containment, and recovery objectives rather than assuming failure can be optimized away.

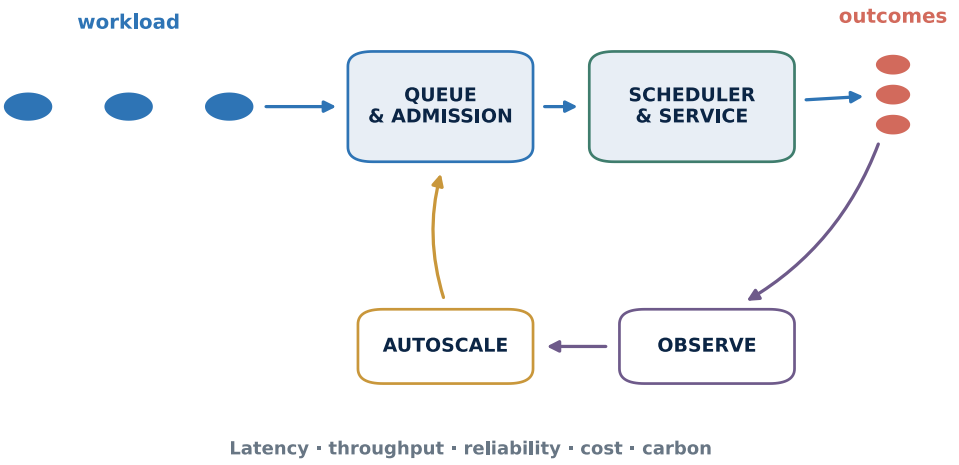


Figure 14. Service flow from workload to outcomes, with observation and autoscaling feedback.

Figure 14 includes the scheduler within a service system. Workload, queues, hardware, feedback, and service objectives interact, so we need to examine their combined behavior.

These queue interventions also move cost: replication adds write amplification; hedging issues extra work; autoscaling acts after measurement and provisioning delay. The model must include these feedback loops and recovery workloads (Barroso, Hölzle, and Ranganathan 2018).

A useful performance experiment therefore specifies the workload distribution, concurrency, warm-up, cache state, failure conditions, request dependency graph, and measurement interval. It reports distributions, not only averages. It separates service time from queueing time and client-perceived time. It checks coordinated omission, in which a load generator stops sending work while a service is slow and thereby under-samples the worst delays.

Suppose a service allocates replicas r_{sz} by service s and zone z. Let $\mathcal{R}$ be the set of nonnegative integer allocations satisfying $Ar \le b$, the latency-risk bound $\Pr(L_s(r,\xi) > \ell_s) \le \varepsilon_s$, and the survivability requirement $\sum_z r_{sz}\,\mathbf{1}\{z \notin F\} \ge k_s$ for every service s and admitted failure domain F. Here L_s is service latency, distinct from the average queue population L in Equation (63). A resource model can then minimize cost and energy compactly:

$$\min_{r\in\mathcal{R}} \left(\sum_{s,z} c_{sz}\, r_{sz} + \lambda_E E(r)\right). \tag{66}$$

The feasible-set definition paired with Equation (66) makes three structures visible: integer capacity, chance or risk constraints for latency, and survivability under failure sets. The model should include startup delay, state replication, data locality, deployment surge, and recovery workload. A capacity plan that is feasible only in steady state can fail during the change that implements it.

Failure domains must be physical and organizational. Zones can share power, control planes, software versions, credentials, or operators. Correlated software failure can defeat geographic redundancy. Stress tests should remove common dependencies, delay telemetry, and corrupt coordination—not only simulate independent host loss.

Software quality, architecture, and delivery

We can examine software optimization at two related levels. As a product, software needs to remain understandable and changeable. During execution, it consumes resources and provides a service. Runtime improvements therefore need to be considered together with maintainability, user requirements, cost, and physical limits.

Quality is a vector

Software quality includes functional suitability, performance efficiency, compatibility, interaction capability, reliability, security, maintainability, flexibility, and safety-related properties. ISO/IEC 25010 provides a quality model, while a project needs to define the thresholds applicable to its use (International Organization for Standardization 2023). Keeping the dimensions separate helps us identify which properties each metric measures.

We can consider technical debt as a liability with an uncertain future cost of change. An expedient shortcut can be rational if the option value of speed exceeds expected remediation cost. It becomes dangerous when the debt is hidden, compounding, or coupled to safety-critical behavior (Cunningham 1992; Sculley et al. 2015).

Staffing decisions also change coordination and communication costs. Adding people to a late software project can delay it further when training and coordination consume the capacity expected from the new staff (Brooks 1995).

Profile before transforming

Performance work follows a causal loop: define a user-visible target, measure representative workloads, locate a bottleneck, change one mechanism, and remeasure. CPU profiles, allocation traces, lock contention, I/O traces, database plans, and distributed traces observe different layers. Optimizing the hottest function may not improve wall-clock time if the critical path lies elsewhere.

Compiler optimization preserves a chosen semantics while changing representation. Constant folding, common-subexpression elimination, dead-code elimination, inlining, loop transformation, vectorization, instruction scheduling, register allocation, and code generation solve interacting problems (Aho et al. 2006; Muchnick 1997). Whole-program and profile-guided optimization use more context but raise build time, reproducibility, and debugging costs.

Delivery as flow optimization

Build and deployment pipelines optimize lead time subject to correctness and recovery constraints. Small batches shorten feedback; automated tests reduce verification cost; canary

releases and feature flags expose a controlled fraction of users; rollback and observability limit downside (Humble and Farley 2010). Common metrics—deployment frequency, lead time, change failure rate, and recovery time—should be used diagnostically, not converted into isolated quotas (Forsgren, Humble, and Kim 2018).

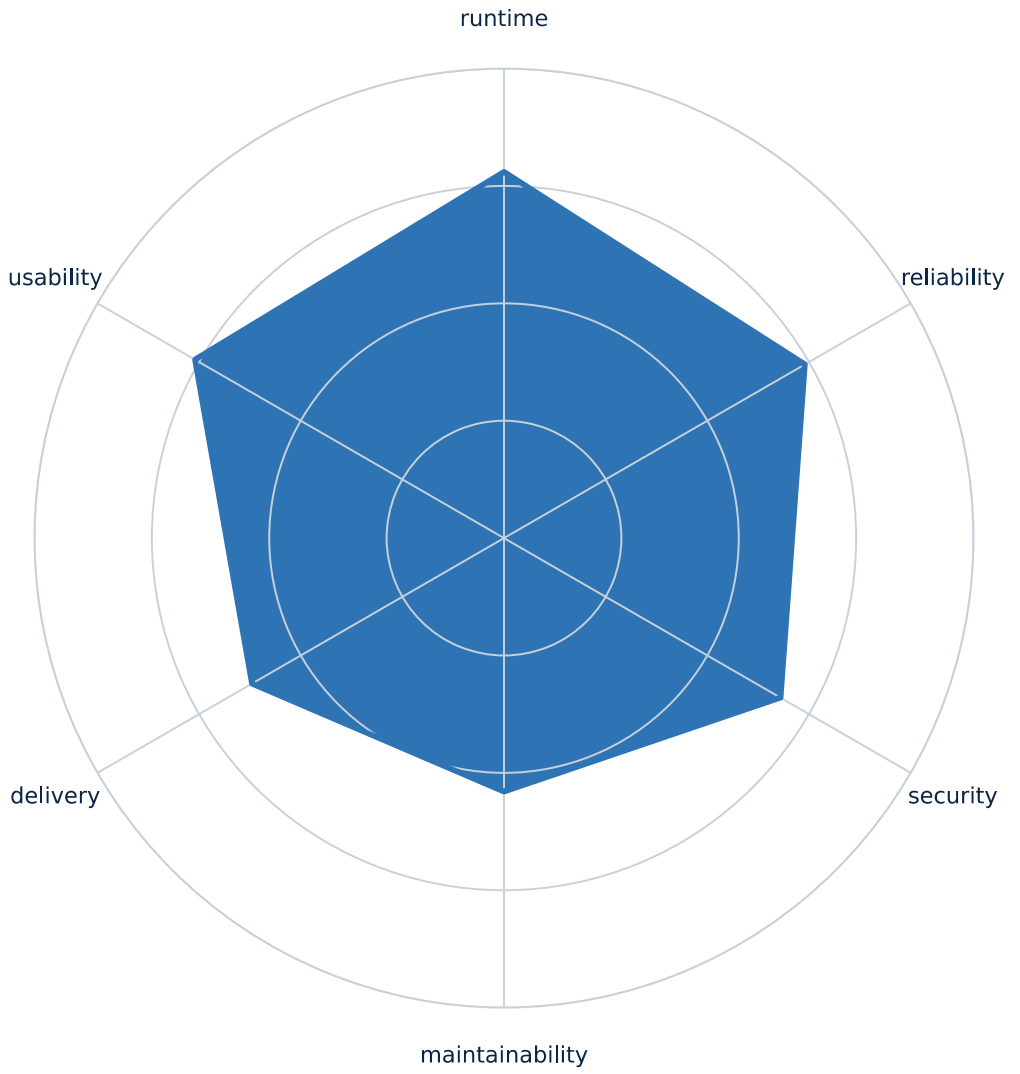


Figure 15. A radar chart comparing runtime, reliability, security, maintainability, delivery, and usability.

Figure 15 uses illustrative project attributes, including usability and delivery; its axes are not a reproduction of ISO/IEC 25010:2023. A runtime improvement may also change reliability, security, maintainability, delivery speed, or the effort needed to understand the software.

The optimization boundary

Some concerns belong in code, some in architecture, some in operations, and some in product policy. Caching can reduce latency but create staleness. Retries can improve success rates but

amplify overload. Compression saves bandwidth but costs compute. A locally reasonable library choice may fragment the platform. Architecture makes these couplings explicit.

Secure software development integrates threat modeling, dependency control, least privilege, code review, reproducible builds, vulnerability response, and provenance throughout the lifecycle (Souppaya, Scarfone, and Dodson 2022). If these controls are introduced only after performance tuning, their interactions with the existing design may require substantial revision.

We can also optimize the transformations used in software engineering. A compiler changes code to improve runtime while preserving semantics. A refactoring changes structure while preserving behavior. A deployment pipeline changes delivery cadence while preserving safety. A scheduler changes execution order while preserving dependencies. In each case, we verify the preservation claim before accepting the local gain.

Changes can move cost between quality dimensions. Aggressive inlining increases binary size and instruction-cache pressure; abstraction may require additional allocation; a fast release process without observability can increase recovery time; controls that are difficult to use can encourage bypasses.

Delivery also depends on work in progress, review latency, and coordination between teams. Very small changes can create excessive review overhead. Automated tests shorten feedback only when they are trustworthy and maintained.

Organizational boundaries act like software interfaces. Work waits for review, environments, security, release, or another specialization. Optimizing each team for utilization can worsen global flow by increasing these queues. Limit work in progress, reduce batch size, assign responsibility, and measure lead time and recovery across the complete change (Forsgren, Humble, and Kim 2018).

Measurement protocol

Performance measurements depend on the state of the computing system. Caches warm, compilers change generated code, and garbage collectors move work between intervals. Networks share capacity, cloud hosts vary, and measurement itself consumes resources. We therefore treat a benchmark result as an observation under specified conditions and test a proposed optimization experimentally.

We begin with the service consequence to be improved. Users may care about tail response, throughput under a fixed cost, energy per completed task, or time to recovery. CPU utilization, allocation rate, cache misses, and request concurrency are explanatory measurements. Optimizing an internal counter can move work elsewhere or improve a microbenchmark while degrading the end-to-end path.

A reproducible performance experiment records hardware and topology, firmware, operating system, runtime, compiler and flags, container limits, power state, dataset, request mix,

warm-up, affinity, concurrency, sample count, clock source, and background load. Separate initialization, steady state, and recovery. Report distributions and tails, not only a mean.

For an optimization with parameters x, measure paired alternatives under randomized or interleaved order where feasible. Use common workloads to reduce comparison variance. Hold out workloads and machines for final evaluation. If search repeatedly probes the same benchmark, the reported best inherits selection bias; confirm it on an untouched run.

Architecture and change

Architectural optimization chooses boundaries, interfaces, consistency, caching, batching, and deployment structure. These decisions determine which later resource allocations are possible. A stateful synchronous dependency creates latency coupling; an asynchronous queue creates buffering and eventual consistency; a cache changes data freshness and invalidation risk. If the architecture is fixed before we examine cloud cost, the largest sources of cost may be excluded from the decision variables.

Automated architecture checks can enforce dependency rules, latency budgets, schema compatibility, security controls, and deployment invariants. However, an old rule can also preserve a poor design boundary. We need to review whether the rule still serves its purpose and investigate workarounds created by teams.

Record the evidence behind technical-debt estimates: dependencies, unsupported versions, missing tests, incident history, manual work, cognitive load, and change lead time. A single score can conceal this evidence and obscure which liabilities are deliberate choices and which hazards lack an owner.

A release system should optimize for small reversible changes within safety constraints. Feature flags, canaries, compatibility, observability, and automated rollback create recourse. They can also increase state space and operational complexity. We therefore include ownership and removal of obsolete flags in the maintenance plan.

Field checklist

- Name the scale of every performance claim: algorithm, implementation, machine, service, or user journey.
- Use a resource vector and explicit service constraints rather than a single benchmark score.
- Model queues, variability, bursts, retries, and feedback near capacity.
- Treat data structures, schemas, partitioning, and indexes as compiled workload assumptions.
- Require a preservation claim for compiler, refactoring, scheduling, and deployment transformations.
- Benchmark distributions under representative and failure workloads; guard against measurement artifacts.
- Include migration, observability, rollback, and organizational flow in the optimization boundary.

Translation lens. Computing connects abstract operations to physical resource use through algorithms and implementations. We need to preserve program semantics and service obligations while measuring changes in time, memory, bandwidth, energy, and human work.

Chapter 9 · Machine learning, artificial intelligence, security, privacy, and assurance

Learning as nested, decision-focused optimization

Machine learning fits models from data. Its training loss represents a particular criterion that may only approximate the intended outcome. We can examine the full process through problem selection, data generation, labeling, representation, optimization, evaluation, deployment, monitoring, and governance. Each transition can introduce an error or change the interpretation of the result.

Empirical risk and generalization

Given data $(x_i, y_i)_{i=1}^n$, empirical risk minimization chooses

$$\hat{\theta} \in \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell\left(f_\theta(x_i), y_i\right) + \lambda R(\theta). \tag{67}$$

The loss ℓ encodes an error model; the regularizer R encodes preference or capacity control. Training error is not deployment error. Generalization depends on hypothesis class, data dependence, sampling, optimization, and shift (Hastie, Tibshirani, and Friedman 2009; Goodfellow, Bengio, and Courville 2016). Cross-validation estimates out-of-sample performance only when splits respect time, groups, leakage paths, and intended use.

Stochastic gradient methods estimate a full gradient from minibatches. Momentum, adaptive methods, schedules, normalization, and preconditioning reshape the trajectory (Robbins and Monro 1951; Duchi, Hazan, and Singer 2011; Kingma and Ba 2015; Bottou, Curtis, and Nocedal 2018). In deep learning, reaching zero training loss does not identify a unique solution; initialization and implicit bias help select among many interpolating models.

Metrics, calibration, and decisions

Accuracy alone may be misleading when classes are imbalanced or errors have different costs. Precision, recall, specificity, ranking loss, calibration, time-to-event error, robustness, latency, and abstention describe different aspects of performance. We still need a decision rule after estimating probabilities. With action a, state y, loss $L(a, y)$, and features x, Bayes decision theory selects

$$a^*(x) \in \arg\min_{a} \mathbb{E}[L(a, Y) \mid X = x]. \tag{68}$$

Thresholds therefore depend on consequences and capacity, not merely model discrimination.

Scaling and search

Architecture search, hyperparameter tuning, data selection, and prompt or policy optimization form outer loops around training (Elsken, Metzen, and Hutter 2019). Bayesian

optimization is useful when evaluations are expensive. Distributed training trades elapsed time against communication, synchronization, reproducibility, and energy. Larger models can improve broad capability while increasing cost, concentration of access, and evaluation difficulty (Kaplan et al. 2020).

Deployment as a controlled intervention

Deployment can change the process that generates future data. For example, ranking affects clicks, moderation affects language, pricing affects demand, and clinical advice affects treatment. We need to distinguish the quality of a model from the effect of the system that uses it. Off-policy evaluation, randomized trials where ethical, staged deployment, monitoring, and causal analysis provide different forms of evidence for this distinction.

Federated learning keeps some computation near data holders. Privacy requires separate guarantees. A mechanism M is (ε, δ)-differentially private if, for all adjacent datasets D, D' and all measurable output events S,

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \Pr[M(D') \in S] + \delta \tag{69}$$

the inequality holds for the stated adjacency relation (Dwork and Roth 2014). The privacy budget composes across queries and depends on the mechanism and implementation. Accuracy and participation remain coupled. Access control, security, and purpose limitation address additional requirements.

Federated learning moves some computation to data holders but does not automatically provide privacy or fairness. Updates can leak information; client participation can be unequal; communication and device energy can dominate; non-identical data complicates convergence (Kairouz et al. 2021). Secure aggregation, differential privacy, robust aggregation, and governance address different failure modes.

Machine learning contains several nested loops. An inner loop fits parameters; an outer loop chooses architecture, data, regularization, and hyperparameters; a product loop chooses thresholds and actions; an operational loop monitors consequences and retraining; a governance loop may change the objective or stop the system. We need to include these outer decisions when examining the effects of optimization.

Empirical risk minimization assumes that an average over training observations is informative about a target distribution. Generalization theory and validation methods qualify this step, but deployment adds further shifts: users adapt, upstream systems change, selection policies alter who receives labels, and interventions change outcomes. A model can continue minimizing its training loss while its decision value falls.

Equation (68) uses loss minimization. With utility U, the corresponding decision rule uses estimated probabilities $P_\theta(y \mid x)$ and permits a larger action set:

$$a^*(x) \in \arg\max_{a \in \mathcal{A}} \sum_{y \in \mathcal{Y}} P_\theta(y \mid x)\, U(a, y; x). \tag{70}$$

Equation (70) separates probabilistic estimation from utility and permits abstention, referral, or information gathering. Calibration matters because predicted probabilities enter the consequence calculation. Class balance and a confusion matrix are insufficient when harms differ, capacity is constrained, or the action changes future labels.

Data selection determines which distinctions a model can learn. Sampling, labeling, augmentation, deduplication, and curriculum have distributional consequences. More data can reinforce biased measurement, leak evaluation cases, duplicate common examples, or add computation without decision value. Filtering can remove dialects, rare events, disability access patterns, or contested cases; a web corpus can contain private, false, or harmful material.

We assess data quality through provenance, coverage, the labeling process, dependence, recency, and consent. These axes describe the evidence available to the model; a separate record of data rules (below) states which forms of collection and use are permitted.

Hyperparameter search can overfit the validation set. Repeated evaluation selects favorable noise just as any optimizer does. Nested validation, sealed test sets, preregistered metrics, and limited access reduce this optimizer's curse. When teams compare many architectures, datasets, and seeds, they should report the selection process and variability rather than only the winning run.

A scaling decision is multiobjective. Compute-optimal training under a narrow loss may not be deployment-optimal once serving, maintenance, evaluation, and governance are included (Hoffmann et al. 2022).

Evaluation of the complete system

A model maps inputs to predictions or generated outputs. An AI system also chooses data, prompts or context, retrieval, tools, thresholds, abstention, user interface, logging, escalation, and update. Evaluation must follow this full decision path. A better benchmark model can produce a worse system if latency removes review time, confidence is miscommunicated, retrieval leaks data, or automation changes who receives service.

We first describe the intended use. This includes the users and affected nonusers, the decision and its authority, unacceptable actions, required evidence, operating conditions, and a safe alternative. These requirements then guide data and model selection. For example, a document summarizer needs source fidelity and appropriate citations. A ranking system needs analysis of exposure and appeals. A control agent needs defined permissions and recovery procedures.

Organize evaluation by claim and condition. Rows can include task performance, calibration, robustness, safety, security, privacy, fairness, human usability, resource cost, and recovery. Columns can include ordinary data, temporal shift, subgroup or language, adversarial input, missing context, tool failure, stale retrieval, overload, and operator disagreement.

We select a test for the claim in each cell. Accuracy does not establish calibration, and finding an adversarial failure does not estimate its frequency. A fairness measure also depends on whether the relevant groups have been defined adequately. A privacy mechanism requires a specified budget and implementation boundary. Similarly, preference for an interface does not show that users rely on its recommendations appropriately.

Generative evaluation needs reference-grounded and open-ended tasks. Test factual consistency, source entailment, uncertainty expression, refusal, instruction hierarchy, harmful transformation, and recovery after error. Preserve examples and rationales, because aggregate scores can improve while a severe capability regresses. For open-ended quality, use multiple trained reviewers, calibrate disagreement, and avoid presenting a preference model as universal taste.

Data rules and provenance

We can record the data rules explicitly: permitted sources and uses, consent or legal basis, provenance, exclusions, retention, correction, deletion, access, and audit. Document whose labor labels and moderates data and which ambiguities the labeling guidelines remove. When model outputs create new training data, mark synthetic provenance and avoid feedback that narrows diversity or amplifies errors.

Decision-focused learning

Prediction error and decision loss can differ. An error near an action threshold or within a constrained subgroup may matter more than a larger error that leaves the action unchanged. Decision-focused training includes a differentiable or approximate representation of subsequent optimization so that the learned model can support the decision. This may improve overall performance. However, evaluation under one fixed policy may conceal defects that become relevant under another policy.

Retain independent predictive and causal evaluation. If the downstream objective is misspecified, decision-focused learning will exploit it more efficiently. If constraints or prices change, a model specialized to yesterday's decision may transfer poorly. Compare modular and end-to-end approaches under policy change.

We propagate model uncertainty into the decision procedure. This includes checking calibration where actions change, evaluating abstention, and examining how uncertainty affects constraints. For consequential decisions, conservative limits and professional review may be needed in addition to posterior confidence.

Resources and access

Model selection consumes compute, energy, hardware, data labor, and time. Report training and inference resources at the service boundary, including repeated experiments and utilization. A smaller model can be dominated if its poor accuracy causes costly downstream action; a larger model can be dominated if marginal benefit is negligible.

Optimize total service consequence: hardware manufacture, electricity, latency, throughput, retrieval, networking, cooling, and human review. Test demand rebound. A faster or cheaper model can increase use enough to raise absolute burden.

Resource access also shapes who can reproduce research or compete. Use end-to-end baselines and disclose budgets. Automated architecture or hyperparameter search should count all trials, not only the final run (Hutter, Kotthoff, and Vanschoren 2019; Elsken, Metzen, and Hutter 2019).

Security, privacy, and adversarial optimization

We can model some security decisions as optimization against an adaptive opponent. A defender allocates limited resources to prevention, detection, response, and recovery. An attacker searches for weaknesses that can be exploited. Historical incident rates are endogenous: they depend on these earlier choices. Successful defenses may prevent attacks from appearing as incidents, while recorded attacks reflect the conditions the attacker encountered.

Threat models and attack surfaces

A threat model identifies assets, adversaries, capabilities, entry points, trust boundaries, and harms. A stylized defense problem is

$$\min_{x \in X} \mathbb{E}_{a \sim P(a|x)}[H(x,a)] + C(x), \tag{71}$$

where investment x changes both harm H and the distribution of attacks. Static risk scores miss strategic adaptation. Game theory, robust optimization, and red-team exercises supply complementary views.

Attackers can exploit people, suppliers, credentials, dependencies, and operational exceptions. A threat model must therefore describe the actual system, including adversary knowledge, access, incentives, and acceptable residual risk.

Least privilege, segmentation, secure defaults, diversity, rate limits, and multiple layers of controls can limit the consequences of compromise. These controls also add work and complexity. We need to test both adversarial behavior and ordinary workflows. A convenient shortcut may be exploitable, while a restriction that prevents necessary work may encourage users to bypass it.

Privacy and information flow

Privacy concerns appropriate information flow, autonomy, inference, and power. Encryption protects data in transit or storage but not necessarily data in use. Access control limits authorized operations; minimization reduces retained attack surface; differential privacy bounds contribution leakage; secure multiparty computation and trusted execution change who can observe intermediate state. These mechanisms solve different problems.

Privacy requirements include consent, contextual integrity, data minimization, access control, retention, and protection against inference. Their logical status must be explicit. The differential-privacy guarantee in Equation (69) addresses changes between adjacent datasets; it does not settle all these requirements.

Prevention and resilience

Expected loss alone can underweight rare systemic failure. Resilience asks whether the system can continue essential functions, degrade safely, restore state, and learn. Objectives may combine expected loss with tail risk:

$$\min_{x} \ \mathbb{E}[L(x,\xi)] + \lambda \mathrm{CVaR}_{\alpha}\big(L(x,\xi)\big). \tag{72}$$

Backups must be isolated and restored in exercises; incident response must be rehearsed; recovery objectives must reflect dependency chains. Diversity can reduce common-mode failure but increases maintenance burden.

Formal methods and certificates

Testing samples behaviors. Formal verification reasons over a model of all admitted behaviors. Model checking explores state transitions; theorem proving derives claims from axioms; abstract interpretation computes safe approximations; type systems exclude classes of programs. Optimization enters synthesis, counterexample search, invariant discovery, solver-backed verification, and minimal repair.

We interpret a certificate together with its assumptions. A proof about source semantics says little if the compiler, hardware, configuration, or specification is wrong. Assurance therefore links each claim backward through its evidence, model, and assumptions.

A formal result still requires a specification that represents the intended policy, deployment of the verified artifact, and examination of attacks outside the modeled interface. Each transition from specification to operation needs a preservation check.

Adversarial machine learning makes the nested structure explicit. A defender chooses model parameters while an attacker chooses a perturbation:

$$\min_{\theta} \ \mathbb{E}_{(x,y)} \left[\max_{\delta \in \Delta(x)} \ell(f_\theta(x+\delta), y) \right]. \tag{73}$$

The set $\Delta(x)$ describes the assumed threat. A small norm ball may simplify computation while omitting semantic manipulation, data poisoning, model extraction, prompt injection, or supply-chain compromise. A robustness result within Δ therefore applies to the stated perturbations and does not establish general security.

Treating privacy as a hard design constraint can stimulate better architecture: local computation, aggregation, limited retention, synthetic or sampled data, and purpose-specific models. A penalty formulation that trades privacy against marginal accuracy may silently change the normative status of the requirement.

For AI systems, examine retrieval manipulation, tool abuse, credential compromise, and dependency attacks as well as perturbations to model inputs. These are extensions of the threat model, with distinct access assumptions and consequences.

Robust training solves a minimax approximation under a threat set. The threat set is a governance artifact: it says what the defender anticipates and what perturbations are considered equivalent. Robustness inside it can trade against ordinary performance and does not protect against a different mechanism (Goodfellow, Bengio, and Courville 2016). Red teams should therefore challenge the boundary, permissions, and recovery as well as model inputs.

Agents require least privilege, scoped credentials, confirmation for consequential actions, rate and spend limits, sandboxing, idempotence where possible, transaction logs, and a tested stop. Tool output is untrusted input. A planning system should not be able to grant itself broader authority because the objective is difficult.

Assurance for adaptive systems and change

Adaptive systems require assurance cases rather than a single metric. The case links claims, evidence, assumptions, monitors, and response. It should cover data quality, performance by relevant group and condition, calibration, robustness, security, privacy, energy, human factors, fallback behavior, and impact. NIST's AI RMF organizes risk work around Govern, Map, Measure, and Manage; the EU AI Act adds legal duties differentiated by system role and risk (Tabassi 2023; European Parliament and Council of the European Union 2024). These frameworks do not replace domain regulation or professional judgment.

For each monitored quantity, we specify a threshold, owner, investigation path, and rollback rule. Human oversight also needs sufficient time, information, and authority to delay or reverse a recommendation. Override records provide evidence about the system and its use; they should not automatically be classified as operator failures.

An adaptive system can change through data, weights, configuration, retrieval, prompts, tools, interfaces, or users. Each change requires appropriate evidence. For example, a retrieval update may introduce factual or legal errors; a threshold change may alter actions; and a model update may reduce calibration despite better average performance.

Maintain a system card with versions, intended use, evidence, limitations, incidents, and owners. Monitor input and outcome drift, calibration, abstention, overrides, subgroup effects, resource use, security events, and appeal. These records support the assurance case; they do not certify the system by themselves.

An assurance case is a structured argument from claims to evidence, assumptions, and defeaters. A safety case applies this structure to safety claims. It may state that a system remains within an operational envelope, detects specified faults, refuses unsupported cases, and hands off safely. Evidence can include analysis, tests, formal checks, simulation, human-factors work, and monitored use.

Learning systems require runtime evidence because behavior can be difficult to exhaustively enumerate. Monitor distribution, confidence, constraint residuals, tool actions, and outcome. Runtime monitors should be independent enough that a common model error does not defeat both actor and guard.

The case should identify claims for which evidence is insufficient. “No harmful output exists” is usually indefensible; “the system blocks these enumerated actions, was challenged under these conditions, and routes uncertain cases through this process” is testable. Update the case with every material system change.

Field checklist

- Map the parameter, hyperparameter, data, decision, operational, and governance loops separately.
- Convert predictive scores into actions through calibrated consequences, capacity, and abstention.
- Protect the test process from repeated selection and report seeds, search budget, and variability.
- State the adversary and justify the perturbation or attack set semantically.
- Treat privacy, security, rights, and safety according to their logical status, not as default penalties.
- Build an assurance case connecting claims to heterogeneous evidence and response actions.
- Verify that human oversight has time, information, authority, and a safe fallback.

Translation lens. Learning, decision-making, adversarial search, and governance form interacting optimization processes. When transferring this structure, we specify their data, agents, authority, constraints, threats, and consequences. A predictive guarantee alone does not establish the quality of the resulting decision.

Chapter 10 · Systems engineering, structures, signals, control, hardware, and co-design

Systems engineering, co-design, and exchanged envelopes

Engineering optimization connects a purpose with an artifact and the system in which it will operate. We need to consider requirements, architecture, physical models, uncertainty, manufacturing, verification, operation, maintenance, and retirement. A component selected in isolation may be unsuitable for the system because its requirements and effects depend on other components.

From needs to a design space

Requirements convert stakeholder needs into testable constraints and preferences. Good requirements state the operational context, units, tolerance, and verification method. Premature numerical targets can freeze one architecture; vague aspirations cannot support trade studies. A design structure matrix or dependency graph helps identify coupled variables, analyses, and teams.

Multidisciplinary design optimization (MDO) can be written schematically as

$$\min_{x,y_1,\dots,y_m} f(x, y_1, \dots, y_m) \tag{74}$$

subject to discipline equations

$$r_i(x, y_1, \dots, y_m) = 0, \qquad i = 1, \dots, m, \tag{75}$$

and system constraints. Here m is the number of disciplines. The shared design x might contain geometry and material choices; y_i might represent aerodynamic, structural, thermal, control, or economic states. Architectures differ in whether they solve discipline consistency inside each evaluation, distribute targets, or optimize all states simultaneously (Martins and Lambe 2013).

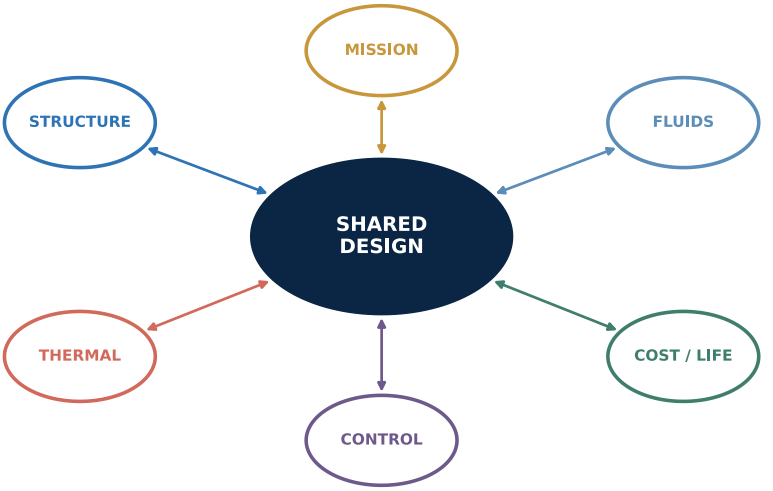


Figure 16. Coupled engineering disciplines exchanging shared design variables, states, sensitivities, and requirements.

Figure 16 visualizes co-design as an exchange of shared variables, states, sensitivities, and requirements rather than a sequence of isolated component optima.

Trade studies and value models

Weighted scoring tables are useful for organizing evidence, but their arithmetic can suggest false precision. Scores should expose dominance, sensitivity to weights, and uncertainty. A total score does not justify replacing a hard safety constraint with a small penalty.

Surrogate and reduced-order models can reduce the cost of analysis. We need to identify the region in which each approximation is supported, since search may select a candidate in a poorly modeled region. Adaptive sampling can use both predicted performance and uncertainty to choose further evaluations. Verification checks the solution of the equations, whereas validation checks their representation of the intended system.

Lifecycle optimization

Lifecycle cost includes more than the acquisition price. Reliability, maintainability, spares, training, energy, upgrades, disposal, and opportunity cost matter. Design for modularity can raise initial mass or price while lowering future adaptation cost. Real options value the ability to defer, expand, switch, or abandon.

Robustness and resilience are distinct. A robust design performs acceptably across anticipated variation. A resilient system can detect, absorb, recover, and adapt after disruption. Both require margins whose apparent “inefficiency” is functional.

A monolithic MDO formulation exposes all coupling but can be organizationally and computationally unwieldy. Decomposition coordinates subsystem problems through

consistency constraints, targets, or prices. The architecture must match ownership, confidentiality, model cadence, and the ability to exchange sensitivities.

Exchanged envelopes and margin

An interface contract should specify variables, units, coordinate frames, valid ranges, uncertainty, fidelity, update cadence, and responsibility. A component that exports only a nominal value conceals sensitivity and may leave downstream robustness analysis without the information it needs. Exporting a response surface, bounds, or an adjoint may preserve more useful structure without revealing every internal detail.

Suppose each engineering team supplies one nominal value: a mass, a power requirement, a workload, or a trajectory. This interface hides flexibility and may lead to repeated redesign. We can instead exchange feasible or performance envelopes that describe how these quantities change together.

For example, a thermal team can describe allowable heat as a function of ambient conditions and flow. A compute team can describe throughput for different power, precision, and latency settings. A battery team can provide relationships among energy, peak power, temperature, degradation, and reserve. A controller can describe reachable performance under actuator and timing limits. We can combine these envelopes before choosing one setting for each component.

Each envelope has a version, uncertainty, and a region of validity. An excessively conservative range may make the combined problem infeasible. An optimistic range may allow a design that later fails integration. Joint tests can help revise these ranges. Where full models cannot be shared, response surfaces, dual sensitivities, or bounds may provide the information needed by the other teams.

Engineering margin absorbs model error, manufacturing tolerance, degradation, load growth, and rare events. We need to record its allocation across components because stacked conservative margins can produce excess mass or cost, while assumed shared margin can leave a gap.

Create a margin register with source, owner, uncertainty covered, confidence, consumption, and release condition. Separate deterministic allowance, statistical variation, and contingency. Allocate system margin where it most effectively protects consequence, then verify that components do not spend the same reserve twice.

Optimization can estimate the local value of relaxing a margin, but this estimate depends on the model. A small objective improvement may leave too little thermal or schedule reserve. We therefore report the distance to constraints and test what happens when a component fails to meet its requirement. Robustness concerns the margin available in the specified scenarios; resilience also requires detection and recovery when that margin is exhausted.

Structures and physical feasibility

Mechanical and civil optimization shape matter to carry loads, manage motion and heat, endure fatigue, resist hazards, and remain buildable. The governing equations are physical; the design variables may be dimensions, topology, materials, controls, or construction sequences.

Size, shape, and topology

A structural sizing problem may minimize mass using a vector of fixed element mass coefficients m

$$\min_x m^T x \tag{76}$$

Here m is a vector of element mass coefficients, distinct from the discipline count in Equation (75), and f below is an applied load rather than an objective. The design is subject to equilibrium $K(x)u = f$, stress limits, displacement limits, buckling, vibration, fatigue, and manufacturability. Shape optimization moves boundaries. Topology optimization decides where material exists, often through a density field $0 \leq \rho_e \leq 1$. A common compliance formulation is

$$\min_\rho f^T u(\rho) \quad \text{subject to} \quad K(\rho)u = f, \quad \sum_e v_e \rho_e \leq V. \tag{77}$$

In the usual density implementation, a positive minimum density or ersatz stiffness prevents a singular stiffness matrix. Penalization encourages near-binary densities. Filters and length-scale constraints address checkerboards and mesh dependence, which are distinct discretization problems (Bendsøe and Sigmund 2004; Sigmund and Maute 2013). The mathematically optimal geometry may still need support removal, minimum radii, draw directions, standardized stock, repair and inspection access, and tolerances.

Mechanics across scales

Material optimization spans microstructure, component, assembly, and infrastructure. Homogenization links microstructure to effective properties; multiscale methods trade fidelity against cost. Metamaterials invert the usual process by designing geometry to obtain stiffness, wave, thermal, or acoustic behavior.

Contact, fracture, plasticity, turbulence, and large deformation create nonsmoothness or expensive simulation. Adjoint methods compute gradients of a scalar objective with cost weakly dependent on the number of design variables, but deriving and verifying them is substantial work. Finite differences remain useful as a gradient check, not as a universal method.

Civil systems and codes

Buildings, bridges, dams, and geotechnical works face uncertain loads, long lifetimes, changing climate, and public consequences. Code compliance supplies minimum requirements within

the design problem. Reliability-based design uses failure probability or reliability index, while performance-based design evaluates multiple hazard levels and consequences.

Construction scheduling couples design with temporary states. A final structure may be safe although an erection stage is not. Whole-life carbon can conflict with cost and schedule; reuse and adaptability can dominate material efficiency over decades. Equity enters infrastructure through who receives service, who bears disruption, and who is exposed to residual risk.

Robust design protects against specified uncertainty sets; reliability-based design uses probabilities; safety factors compress several uncertainties into a rule. Combining these approaches without examining their assumptions can double-count conservatism or leave gaps.

Whole-life comparisons can reverse a local material optimum. Low mass can coexist with high embodied carbon. A low-cost route can divide a community, and protection for one location can transfer flood risk to another. Include the affected geography in the model.

Validation across scales

Validate material coupon, component, subsystem, integrated system, and operation at their relevant scales. Model correlation across levels rather than assuming a coupon property transfers unchanged to a manufactured assembly. Tolerances, joints, interfaces, software timing, operators, and environment enter progressively.

For control, prove or test invariant regions under actuator limits and delay, then challenge estimator error and mode transitions. For structures, combine stress and deformation with fatigue, stability, damage tolerance, and manufacturability. For hardware, verify logical function, timing, power integrity, thermal behavior, packaging, and workload performance. The system certificate must show how these conditional component claims compose.

Signals, estimation, sensing, and control

Electrical systems optimize energy, information, dynamics, and uncertainty. Circuit design, estimation, filtering, communication, power conversion, and control share mathematical objects—quadratic forms, spectra, state spaces, norms, and feedback—while differing in physical interpretation.

Signals and estimators

Least squares estimates x from measurements $y = Ax + \varepsilon$ by minimizing

$$\| Ax - y \|_2^2. \tag{78}$$

Weighted least squares reflects unequal noise; regularization stabilizes ill-posed inversion; sparsity penalties support compressed representations. Fourier and wavelet transforms choose bases in which signals or operators become simple. Wiener and Kalman filters optimize mean-square estimation under stated stochastic models (Oppenheim and Schafer

2010; Kalman 1960). Model mismatch, non-Gaussian tails, and sensor bias require robust or nonlinear alternatives.

Information and communication

For a discrete source X, entropy is

$$H(X) = -\sum_x p\,(x)\log_2 p(x). \tag{79}$$

Channel capacity is the maximum mutual information over input distributions,

$$C = \max_{p(x)} I(X;Y), \tag{80}$$

under the channel model and constraints (Shannon 1948). Coding approaches these limits with delay, complexity, and energy costs. Network design adds routing, congestion, spectrum allocation, interference, fairness, and resilience.

Feedback control

For a linear system

$$x_{t+1} = Ax_t + Bu_t, \qquad y_t = Cx_t, \tag{81}$$

where C is the output matrix, distinct from channel capacity in Equation (80), linear-quadratic regulation minimizes

$$J = \sum_{t=0}^{\infty}(x_t^T Q x_t + u_t^T R u_t). \tag{82}$$

Optimality relies on stabilizability, detectability, and the model. Robust control bounds uncertainty; adaptive control updates parameters; model predictive control repeatedly solves a finite-horizon constrained problem using new state estimates (Åström and Murray 2008; Rawlings, Mayne, and Diehl 2017). Delay and saturation can destabilize a controller that looks optimal in an instantaneous model.

Circuit and power design

Analog and digital circuits balance gain, bandwidth, noise, linearity, power, area, yield, and temperature. Power electronics balance switching loss, passive size, electromagnetic compatibility, and control. Electric grids coordinate generation, storage, transmission, demand response, and reserves under network physics. Security-constrained optimal power flow must survive credible contingencies, not only minimize nominal generation cost.

Signal processing selects representations and estimators under limits on noise, bandwidth, latency, and computation. The loss depends on how the signal will be used. For example, a filter minimizing mean-square error may suppress a rare transient needed for diagnosis. Compression that preserves perceived quality may also remove information needed by a subsequent diagnostic method.

Control actions affect both the plant and subsequent observations. State estimation and sensing therefore belong within the feedback design. Safety filters can provide additional

guarantees under their assumptions, including conditions on delay, saturation, and estimator error.

Sensing and human control

Human operators are part of the control architecture. Automation changes skill, attention, mode awareness, and workload. An intervention should specify when the human receives information, what action remains possible, how rapidly control can be transferred, and whether the interface supports diagnosis under stress.

Sensors are often chosen before controllers and models. Yet sensing determines observability, fault detection, wiring, energy, latency, maintenance, privacy, and cost. Co-design sensor placement, precision, sampling, redundancy, and calibration with the decisions the system must make.

Value-of-information analysis can show which measurement changes action. Observability analysis shows which states can be reconstructed under a model. Neither captures maintenance access, cyberattack, consent, or the trust people place in a measurement. Add those constraints explicitly.

Redundancy should consider common cause. Two identical sensors sharing power, location, calibration software, or environmental exposure are not independent. Diverse sensing and analytical consistency checks can improve diagnosis. When sensors disagree, preserve the disagreement and enter a safe mode rather than averaging away a fault.

Hardware beyond PPA

Hardware optimization involves discrete geometric choices, expensive fabrication, substantial verification, and decisions that may persist for years. Power, performance, and area, commonly abbreviated PPA, provide familiar criteria. We also need to consider yield, reliability, security, thermal behavior, verification time, and software compatibility.

Architecture and the energy cost of movement

Processor design allocates transistors and power across execution units, caches, predictors, interconnects, accelerators, and memory controllers. Dennard scaling once allowed smaller transistors to switch faster at roughly constant power density; its breakdown made energy and parallelism central (Dennard et al. 1974; Hennessy and Patterson 2019). Data movement often costs more energy than arithmetic, motivating locality, compression, near-memory computing, and specialized accelerators (Horowitz 2014).

Speedup from an accelerator is limited by the fraction it accelerates, conversion overhead, and utilization. A tensor unit with high peak throughput may be idle because of memory bandwidth, sparsity, small batches, or control flow. Hardware-software co-design shapes algorithms, numerical formats, compilers, and silicon together (Jouppi et al. 2017).

From logic to layout

Electronic design automation transforms a specification through synthesis, technology mapping, floorplanning, placement, clock-tree synthesis, routing, timing closure, power integrity, and sign-off. Each stage changes the feasible region of the next. Placement may minimize a surrogate such as weighted wirelength,

$$\min_{(x_i,y_i)} \sum_{e\in E} w_e \ \mathrm{HPWL}(e), \tag{83}$$

where HPWL denotes half-perimeter wirelength, subject to density and block constraints. True performance depends on routed parasitics, congestion, timing, voltage drop, and thermal coupling.

Each EDA stage uses approximations to quantities evaluated later. Wirelength guides placement but does not equal routed delay. Static timing abstracts from physical waveforms, and design-rule compliance does not establish yield. An improvement at one stage can worsen a later result, so repeated analysis and revision are needed.

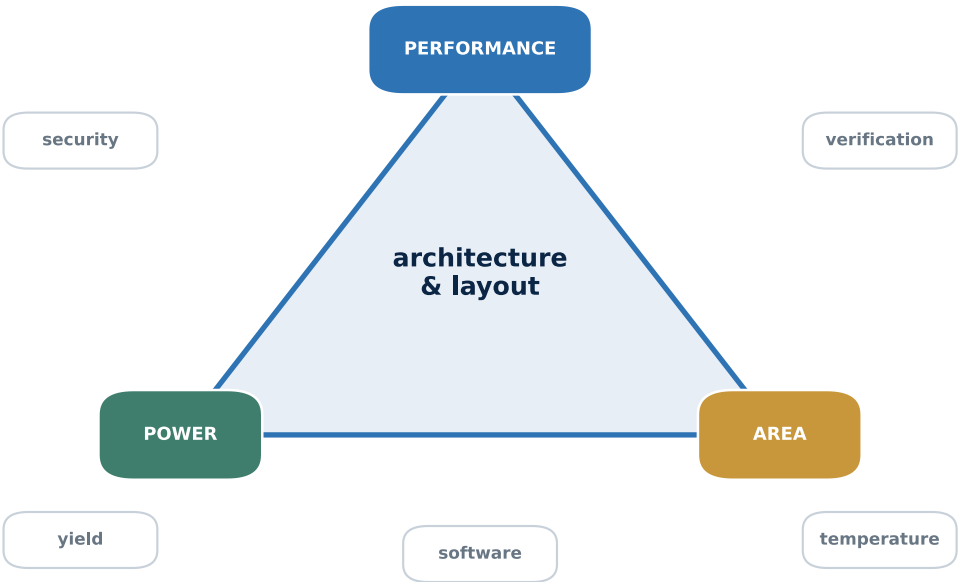


Figure 17. Hardware trade space around performance, power, area, and architecture or layout, with security, verification, yield, software, and temperature.

Figure 17 adds temperature, yield, and verification to power, performance, and area. These additional properties may determine whether a proposed hardware design can be manufactured and used reliably.

Learned placement can search large design spaces, but comparisons require fixed technology files, constraints, downstream tools, and sign-off metrics (Mirhoseini et al. 2021). Human floorplanners encode tacit constraints that a nominal objective may omit.

Learned guidance can fail on a new process, hierarchy, or macro configuration. Warm starts and proposals remain conditional on conventional feasibility analysis and sign-off; the learned reward does not replace these checks.

Variation, yield, and verification

Manufacturing variation makes timing and power distributions rather than constants. Statistical static timing, corner analysis, guard bands, redundancy, and error correction manage tails. Over-margining wastes area and energy; under-margining harms yield and field reliability.

Functional verification commonly dominates project effort. Simulation, emulation, formal equivalence, property checking, coverage, and post-silicon validation are complementary. Verification time can therefore determine whether a faster design is usable within the project schedule.

Development schedule, software ecosystem, and embodied cost can also determine the preferred design. A specialized accelerator may improve selected workloads while narrowing future flexibility.

An edge-accelerator example

Consider an edge accelerator for a medical imaging device. Variables include model architecture, arithmetic precision, memory size, dataflow, clock, voltage, floorplan, thermal path, and software batching. Objectives include inference accuracy, worst-case latency, energy per study, cost, reliability, and updateability. Hard constraints include diagnostic performance, maximum temperature, regulatory configuration control, memory, timing, and safe fallback.

Optimizing the neural model alone may select operators inefficient on the target hardware. Optimizing hardware against a frozen model can create a brittle specialization. Co-design alternates or jointly searches model and architecture, but evaluation must include compilation, calibration, corner timing, thermal transients, and clinical workflow. Quantization error interacts with subgroup performance; batching interacts with latency; thermal throttling interacts with sustained throughput. Verification must cross all interfaces.

The design review should present a Pareto set and a traceable reason for selecting one point. It should identify the margin budget retained after selection. It should specify which future model updates remain feasible. We can include option value by preserving capacity for new algorithms, security patches, or sensor changes.

Software depends on assumptions about instruction behavior, memory, timing, precision, concurrency, and faults. Hardware design depends on assumptions about workload, locality,

duty cycle, thermal conditions, and acceptable approximation. We need to exchange distributions and feasible envelopes for these quantities. One benchmark result does not describe all the conditions needed for co-design.

Approximate arithmetic can improve energy or throughput and alter numerical stability, fairness, or safety. Validate at application and consequence level. A small average error can concentrate near a threshold. Compiler transformations and quantization can change models differently across hardware; regression tests should span targets.

The interface contract includes update. A hardware accelerator can freeze assumptions for years while software and threats evolve. Preserve a fallback path, programmability where justified, and a versioned compatibility matrix.

Field checklist

- Define interface contracts with units, ranges, uncertainty, fidelity, and ownership.
- Expose coupling variables and decide deliberately between monolithic and decomposed coordination.
- Distinguish mathematical, physical, manufacturing, code, and lifecycle feasibility.
- Model state estimation, control, delay, saturation, fallback, and human transfer together.
- Expand PPA to temperature, yield, reliability, verification, security, software, and option value.
- Validate every EDA or design surrogate against the downstream sign-off quantity it represents.
- Preserve margins and future feasible updates rather than optimizing all slack away.

Translation lens. Engineering applications share conservation laws, interfaces, sensitivities, and margins. We can examine co-design through composition: each subsystem needs explicit boundary types and a statement of the properties that remain valid when it is connected to the others.

Chapter 11 · Processes, materials, manufacturing, energy, climate, robotics, and transport

Processes, manufacturing, and safe intensification

Chemical and manufacturing optimization concerns transformations of matter under conservation laws, kinetics, thermodynamics, equipment limits, uncertainty, and safety. We need to examine yield together with selectivity, purity, energy, throughput, operability, emissions, and hazard because these quantities interact.

Process design and operation

A steady-state process model may satisfy

$$F(z, x) = 0, \tag{84}$$

where z contains temperatures, pressures, flows, and compositions, and x contains design or control choices. The optimizer minimizes cost or environmental impact subject to these equations and inequalities. Heat integration, reactor-separator-recycle structure, utilities, and equipment sizing create nonconvex mixed-integer nonlinear programs (Biegler 2010; Edgar, Himmelblau, and Lasdon 2001).

Phase equilibrium and reaction kinetics can produce several steady states. A mathematically feasible state may be dynamically unstable or difficult to reach during startup. We therefore examine operability, controllability, hazards, and relief requirements together with economic performance. Startup, shutdown, cleaning, and abnormal operation belong to process feasibility. Process intensification may remove unit operations and buffers while increasing coupling, fault propagation, and the consequences of failure.

Manufacturing systems

Manufacturing optimization includes product geometry, process parameters, toolpaths, nesting, line balance, inventory, inspection, maintenance, and supply networks. Additive manufacturing expands feasible geometry while imposing anisotropy, support, distortion, surface, and qualification constraints. Subtractive manufacturing rewards accessible features and standard tooling. Design for manufacture and assembly brings these realities upstream.

Quality control distinguishes common-cause from special-cause variation. Tightening tolerances everywhere can increase scrap and cost without improving function. Robust parameter design seeks settings that reduce sensitivity to noise. Predictive maintenance should optimize intervention decisions, not merely predict failure scores.

Large batches improve setup efficiency while increasing delay, work in progress, and defect exposure. High utilization can lengthen queues under variable arrivals and service times, while reducing maintenance windows. Integrate process control with scheduling because tool condition and product mix change the data-generating process.

The manufacturing route also includes orientation, scan strategy, residual-stress control, post-processing, tool access, and fixturing. Low material use does not establish low production cost.

A flexible batch-plant example

Consider a plant with batch processes, storage, renewable electricity, time-varying prices, and delivery commitments. Decisions choose batch start, recipe, inventory, energy purchase, and flexible load. A mixed-integer model can exploit low-price or low-carbon periods, but thermal and material states carry across time. Storage converts temporal variability into capacity and loss.

Include startup, shutdown, cleaning, off-specification risk, minimum run, operator coverage, demand charges, and emergency states. Distinguish average grid carbon from marginal response. A plan that moves every batch to the same “green” hour can create a new peak and change the marginal generator.

Use robust constraints for critical delivery and process safety, stochastic scenarios for price and renewable variation, and receding-horizon updates for operation. Reserve a simple feasible schedule for forecast or solver failure. Compare absolute energy and material throughput so efficiency gains do not hide rebound.

Inherently safer design removes or reduces hazard rather than adding control layers. Prefer less hazardous material, lower inventory, moderated conditions, or simpler pathways where feasible. Optimization can compare alternatives but should not price a catastrophic hazard as an ordinary expected cost.

Lifecycle sustainability includes feedstock, land, water, energy, labor, emissions, toxicity, durability, repair, and end-of-life. Report burden shifting by stage and region. A low operational footprint can depend on scarce or harmful upstream material.

Materials by inverse design and evidence

The composition-processing-microstructure relationship is often many-to-one. Models span scales, and data can be sparse or biased toward successful experiments. Uncertainty must therefore control extrapolation in inverse design.

High-dimensional composition spaces motivate design of experiments, surrogate models, and Bayesian optimization. Physics-based models use mechanistic structure to support extrapolation when their assumptions hold; data-driven models exploit accumulated measurements. Active learning chooses the next experiment by balancing information and potential improvement.

A closed-loop materials campaign selects a candidate, synthesizes it, measures properties, updates the model, and selects again. Failed synthesis may provide information about feasibility; we record its cause and distinguish it from an equipment or measurement failure. Multi-fidelity models combine inexpensive calculations, high-fidelity simulation, and

laboratory evidence. The acquisition objective can include novelty, uncertainty reduction, cost, hazard, and expected improvement. Human scientific judgment remains essential for mechanisms and anomalous results.

Engineering models move between electronic, material, component, process, plant, network, and lifecycle scales. Fine-scale behavior becomes effective parameters at the next scale. The relation between scales is a model that requires validation. Microstructure can create anisotropy; rare defects can dominate fatigue; local congestion can create network instability.

We test relationships between scales in the regions selected by optimization. For example, a material approximation calibrated on conventional geometry may fail for thin members produced by topology optimization. An aggregated grid model may fail when many controllers respond to the same price. We retain uncertainty in these relationships and use higher-fidelity analysis when an extreme candidate requires it.

In multi-fidelity optimization, we specify when a candidate proceeds to a more detailed evaluation. A cheap model can screen many candidates, an intermediate model can refine the comparison, and the target experiment can confirm it. We also retain cases in which fidelities disagree, since they may reveal limitations of the approximations.

The feasible material technology requires reproducible synthesis, characterization, scale-up, joining, certification, durability, toxicity and supply assessment, and an appropriate end-of-life route. A predicted property alone is insufficient.

Use active learning to choose experiments that improve decision-relevant knowledge, preserve negative and failed results, and challenge surrogate extrapolation. Diversity among candidates protects against correlated model error and supply risk. Confirm performance with independent methods and conditions.

The transfer from computation to manufacturing includes a recipe, tolerances, provenance, and evidence about valid operating conditions. A formula alone does not supply this information.

Energy, climate, conservation, and circularity

Energy and environmental optimization spans seconds of grid balancing, decades of infrastructure, and centuries of climate consequence. It is marked by uncertainty, irreversibility, spatial inequity, and multiple decision makers.

Energy-system planning

Capacity expansion chooses generation, storage, transmission, and demand-side resources. A stylized model minimizes discounted system cost

$$\min_{x} \sum_{t=0}^{T} \frac{C_t(x_t)}{(1+r)^t} \tag{85}$$

subject to demand balance, reliability, build rates, network limits, emissions, and technology constraints. Hourly or subhourly operations must be represented well enough that a capacity

plan remains feasible. Representative periods reduce computation but can erase rare stress events and long storage cycles.

Unit commitment and dispatch coordinate discrete starts, ramping, reserves, network flow, and uncertain renewable output. Storage arbitrage alone understates capacity, flexibility, congestion, and resilience value. Demand is partly a design variable shaped by efficiency, tariffs, automation, and social practice.

Weather and demand are correlated; technology costs and policy change; extreme events can determine adequacy. Representing one typical year can misprice storage and reliability.

Linking planning and dispatch requires chronology, transmission, reserves, maintenance, and rare events. Validate a plan in an independent chronological simulation and stress compound events that scenario reduction might erase.

Carbon and ecological boundaries

Lifecycle assessment accounts for impacts across extraction, production, use, and end of life. Results depend on system boundary, allocation rule, counterfactual, time horizon, and data quality. Optimizing one indicator can shift burden to land, water, toxicity, biodiversity, or another region.

Climate mitigation pathways use integrated models to explore technology, land, economy, and policy under carbon budgets (Intergovernmental Panel on Climate Change 2022b). They are scenarios conditioned on assumptions, not forecasts. Planetary-boundary research emphasizes interacting Earth-system limits (Rockström et al. 2009). A safe operating space is more naturally represented by multiple constraints than by a single monetized objective.

Adaptation, resilience, and justice

Flood defenses, heat plans, water systems, ecosystems, and emergency capacity must perform across uncertain futures. Robust decision making seeks actions that are acceptable in many plausible worlds (Lempert, Popper, and Bankes 2003); adaptive pathways stage decisions and preserve options. Discounting strongly affects intergenerational comparisons. Distribution matters because aggregate benefit can conceal concentrated displacement, pollution, or energy poverty.

Circular-economy design optimizes loops of durability, repair, reuse, remanufacture, and recycling. Yet circularity is not automatically low impact: extra transport, processing, or rebound can offset gains. Absolute resource and emission outcomes remain the test.

Balances across boundaries

We can identify balance equations in chemical processes, manufacturing lines, energy systems, transport networks, and robots. Depending on the domain, these concern mass, atoms, charge, energy, momentum, inventory, vehicles, or tasks. A smaller model boundary still needs to account for transfers across it. Conservation equations exclude impossible candidates and provide residuals that we can inspect diagnostically.

For a process network with flow f_{ij} from node i to node j, a steady balance has the form

$$\sum_{j:(j,i)\in E} f_{ji} + s_i = \sum_{j:(i,j)\in E} f_{ij} + d_i, \qquad i \in V, \tag{86}$$

where s_i and d_i represent supply and demand. Equation (86) can describe molecules, electricity under an approximation, vehicles, data, or money, but each domain adds different physics and loss. The equation is a structural correspondence, not an assertion that the entities are interchangeable.

We can check a balance by computing input minus output minus accumulation. This applies to quantities such as material, energy, charge, inventory, people, or money. We compare the residual with the expected measurement and numerical error. A persistent discrepancy may indicate an omitted flow, inconsistent units, sensor bias, or a solver defect.

Balances should be checked at nested boundaries. A factory can close mass onsite while transferring waste upstream; a grid model can balance electrical energy and omit fuel or embodied infrastructure; a transport model can conserve vehicles and omit driver time. The balance therefore applies to the entities and flows included in the model.

Electrification can move emissions upstream; efficiency can induce rebound; mineral extraction and land use can create new burdens. Examine who pays, who receives reliable service, whose land is used, and who is exposed during transition (Intergovernmental Panel on Climate Change 2022b; Raworth 2017).

Repair and recovery networks

Whole-life optimization extends the lifecycle to maintenance, repair, reuse, disassembly, recycling, and disposal. A product optimized for manufacturing cost can be difficult to repair; a lightweight composite can be hard to recycle; a low-operating-energy device can have high embodied impact. Design variables should include modularity, standard fasteners, material separation, diagnostic access, software support, and residual value.

A recycling percentage represents only one aspect of circularity. Material quality degrades, reverse logistics consumes energy, markets fluctuate, and informal labor may bear risk. Mass-balance and provenance make claims auditable. The preferred intervention may be longer life or reduced demand rather than more efficient throughput.

Material choice, joining, modularity, information, and ownership affect what can happen to a product at the end of its use. We can model collection and recovery as a reverse logistics network. The timing, quality, and demand of returned items are uncertain, and decisions include inspection, repair, remanufacture, reuse, recycling, and disposal.

The objective should respect a hierarchy: prevent unnecessary material, extend useful life, preserve high-value function, recover material where appropriate, and manage residual harm. A recycling-rate target can reward heavy low-value material and ignore toxicity or downcycling. Track mass, function, quality, energy, and displacement of virgin production.

Design for disassembly can conflict with durability or safety; modularity can add material; local repair can require training and parts. Pareto analysis should expose these trades. Policy, warranty, right-to-repair, and producer responsibility change the feasible network. We therefore consider circularity through both material design and the institutions that organize reuse and recovery.

Robotics and transportation in open environments

Aerospace, robotics, and transportation combine geometry, dynamics, control, perception, networks, human behavior, and safety. Optimization must bridge design-time models and real-time decisions.

Trajectory and vehicle design

An optimal-control problem chooses state $x(t)$ and control $u(t)$, with terminal cost Φ and running cost L:

$$J = \Phi\big(x(T)\big) + \int_0^T L\,(x(t), u(t), t)\,dt \tag{87}$$

subject to dynamics $\dot{x} = f(x, u, t)$, boundary conditions, and path constraints. Direct transcription discretizes the trajectory into a nonlinear program; indirect methods use necessary conditions from optimal control (Pontryagin et al. 1962; Bryson and Ho 1975). Fuel, time, heating, noise, comfort, risk, and control effort may conflict.

Aircraft and spacecraft design couple aerodynamics, structures, propulsion, thermal protection, guidance, manufacturability, and mission operations. A mass reduction in one subsystem may demand greater cooling or certification effort elsewhere. Margins protect against model error and late growth.

Robotics

Robot optimization spans mechanism design, calibration, state estimation, motion planning, manipulation, and policy learning. A motion planner searches configuration space while avoiding obstacles; trajectory optimization refines a smooth feasible path; feedback control rejects disturbances. The planning procedure needs to account for uncertainty in perception.

In shared spaces, robot objectives need legibility, comfort, recoverability, and conservative behavior under uncertainty. People must be able to anticipate movement. A geometrically shortest path can therefore be behaviorally unsuitable even when collision-free in the nominal model.

Hierarchical robot architectures combine global planning with fast reactive safety. Localization error, moving agents, delay, and actuator saturation qualify a collision-free plan. Formal safety filters remain conditional on their modeled dynamics and sensing.

Simulation can omit contact, lighting, rare obstacles, social norms, or human improvisation. Examine transfer to operation through these mechanisms as well as aggregate success.

Define safe reachable sets, speed and force limits, stop and recovery, and human communication. Operators need mode awareness and authority.

We also include wear, energy, maintenance, and the division of work between robots and people. If a robot increases nominal throughput but creates unplanned recovery tasks for operators, those tasks need to be included in the cost comparison.

Transportation networks

Wardrop's first principle describes user equilibrium: used routes have equal and no greater travel cost than unused alternatives (Wardrop 1952). A system optimum minimizes total travel time. These differ because each traveler ignores congestion imposed on others. Pricing, information, capacity, and service design can move the equilibrium, but demand adapts.

Traffic assignment, transit scheduling, fleet routing, crew planning, charging, and last-mile logistics operate at different levels (Sheffi 1985). Accessibility—what opportunities people can reach—is often a better goal than vehicle speed. Safety, emissions, affordability, disability access, land use, and community severance are core outcomes.

Autonomous and human-driven participants form a mixed system. A policy optimized in a compliant simulator may provoke or confuse real drivers. Field trials should expand gradually, preserve a safe operating domain, and record near misses and interventions. Learning from rare events requires careful exposure denominators and counterfactual analysis.

Infrastructure expansion can induce demand and reshape development. Evaluate short- and long-horizon response, mode shift, access, emissions, safety, and land use. A travel-time improvement is not automatically an accessibility improvement if destinations or affordable housing move.

Scheduling and routing should include transfer reliability, walking and waiting, disability access, freight, maintenance, and disruption. A network with high average performance can isolate a community after one bridge or line fails. Critical connectivity and recovery deserve constraints and stress tests.

A congestion price can represent an external cost under the model. It may also burden people who have no practical alternative. We therefore examine distributional effects, revenue use, and service investment before adopting it. The shadow price alone does not determine whether a toll is justified.

Field checklist

- Write conservation and inventory balances before adding economic objectives.
- Include startup, shutdown, cleaning, maintenance, hazard, and abnormal operation.
- Compile designs through the actual manufacturing and inspection route.
- Record failed experiments and distinguish infeasibility from evaluation failure in inverse design.
- Validate energy plans with independent chronology and compound-event stress tests.
- Distinguish user equilibrium, system optimum, and legitimate transport policy.
- Extend lifecycle boundaries through repair, reuse, disassembly, and disposal.

Translation lens. Physical and infrastructure problems share balances, flows, dynamics, and lifecycle relationships. The conserved quantity may differ. In each application, we still account for sources, sinks, losses, accumulation, and changes to the system boundary.

Part IV · Living Systems, Institutions, and Society

Chapter 12 · Biology, clinical medicine, public health, and biomedical discovery

Evolution and contextual biological optimality

Fitness landscapes and frequency dependence

A genotype or phenotype x may be assigned fitness $w(x)$ in a fixed environment, producing a landscape. Mutation and recombination determine which moves are accessible; drift matters in finite populations; environments change. If fitness depends on population composition p, then

$$w_i = w_i(p), \tag{88}$$

so fitness changes with population composition. Replicator dynamics describe changes in frequency:

$$\dot{p}_i = p_i\big(w_i(p) - \bar{w}(p)\big), \qquad \bar{w}(p) = \textstyle\sum_{j=1}^{m} p_j\, w_j(p). \tag{89}$$

An evolutionarily stable strategy resists invasion under the specified game (Maynard Smith 1982). This property does not express moral or social desirability. Also, a trait may be a by-product, an exaptation, or a consequence of constraints. An adaptation-based explanation therefore needs comparison with alternatives and supporting evidence (Gould and Lewontin 1979).

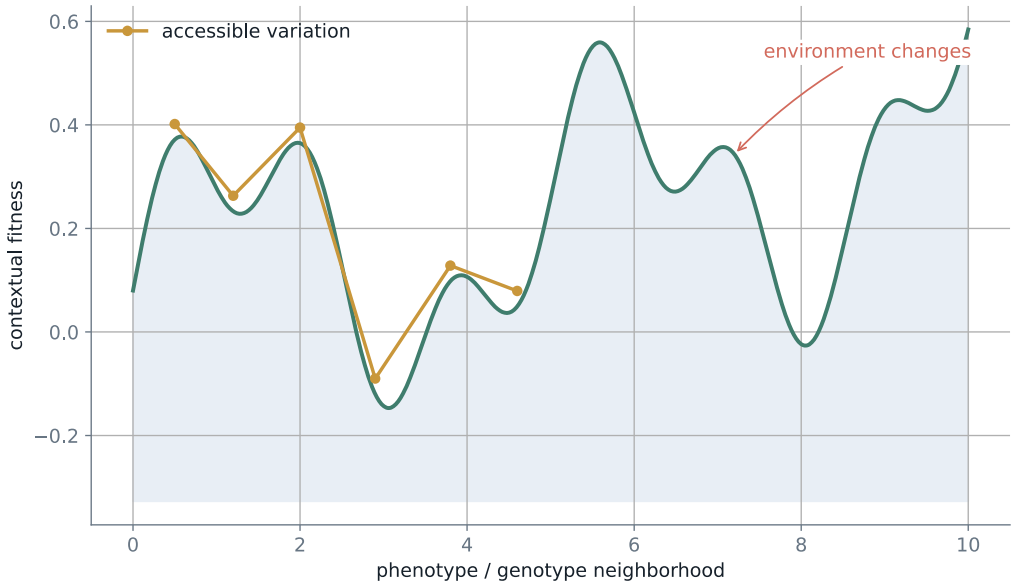


Figure 18. Contextual-fitness landscape with accessible variation and environmental change.

Figure 18 represents a changing adaptive landscape. Mutation, drift, environmental change, and changing niches affect both the available moves and their fitness consequences.

Metabolism and cellular allocation

Cells allocate limited energy, ribosomes, membrane, and elements across maintenance, growth, defense, and reproduction. Flux balance analysis represents steady-state metabolism with stoichiometric matrix S:

$$Sv = 0, \qquad l \le v \le u, \tag{90}$$

where l and u are lower and upper reaction-flux bounds, and then optimizes a proxy such as biomass production $c^T v$ (Orth, Thiele, and Palsson 2010). Conservation limits the possible flows, but we still need to check the objective and steady-state assumptions experimentally. Multiobjective and regulatory extensions can represent relationships among growth, yield, stress, and by-products.

For the flux-balance model in Equation (90), a biomass objective may work in one laboratory condition and fail under stress, regulation, or community interaction.

Ecology and conservation

Ecological optimization concerns populations, food webs, spatial connectivity, harvesting, reserves, invasive control, and restoration. The system may have tipping points, delayed feedback, and alternative stable states (Levin 1999). Maximum sustainable yield can fail when recruitment is uncertain or ecosystem interactions are omitted. Precautionary harvest control and protected-area design manage uncertainty and option value.

Conservation planning often seeks a minimum-cost set of sites meeting representation and connectivity targets. Species counts represent only part of ecological value: evolutionary history, function, cultural relationships, and future adaptation matter. Long-term persistence requires more than static species coverage. Community consent and land tenure also constrain which conservation actions are legitimate.

Biological design versus human design

In synthetic biology and bioengineering, we intentionally design genetic circuits, metabolic pathways, proteins, or ecosystems. The resulting organisms can still mutate, compete, and move. We therefore need to include containment, reversibility, monitoring, and evolutionary stability when evaluating a design.

Natural selection has no external designer, fixed global objective, or guaranteed progress. Historical contingency and inherited material constrain the accessible paths (Darwin 1859; Fisher 1930). Optimization language in biology therefore describes a conditional explanatory model.

Foraging models predict behavior under energy, risk, and time assumptions (Stephens and Krebs 1986). Agreement supports these assumptions without showing that an organism solves

an explicit program. Disagreement can reveal a missing constraint or the wrong scale of selection.

Compare an optimality model with neutral, developmental, phylogenetic, and historical explanations. It should produce predictions that distinguish mechanisms capable of generating the same phenotype (Parker and Maynard Smith 1990).

Clinical decisions and treatment policies

Clinical optimization supports choices for a particular person under uncertainty about diagnosis, prognosis, treatment response, side effects, preferences, time, and resources. We interpret the result within the patient's values and the clinician's responsibility.

From probabilities to actions

Suppose disease state $D \in \{0,1\}$ with state set $\mathcal{D} = \{0,1\}$, and action a is test, treat, watch, or refer. Expected utility is

$$EU(a \mid x) = \sum_{d \in \mathcal{D}} P\,(D = d \mid x)\, U(a, d; x). \tag{91}$$

The preferred action maximizes expected utility, subject to consent and clinical constraints. If treatment benefit increases with disease probability while treatment harm does not, a threshold probability can separate "do not treat" from "treat" (Pauker and Kassirer 1980). A threshold is not universal: it changes with disease severity, treatment burden, comorbidity, alternatives, and the person's preferences.

Bayes' theorem updates odds with evidence:

$$\frac{P(D|T)}{1-P(D|T)} = \frac{P(T|D)}{P(T|\neg D)} \frac{P(D)}{1-P(D)}. \tag{92}$$

Likelihood ratios characterize a test, while pretest odds place it in context. Testing has value only if results can change an action or meaningfully inform the person.

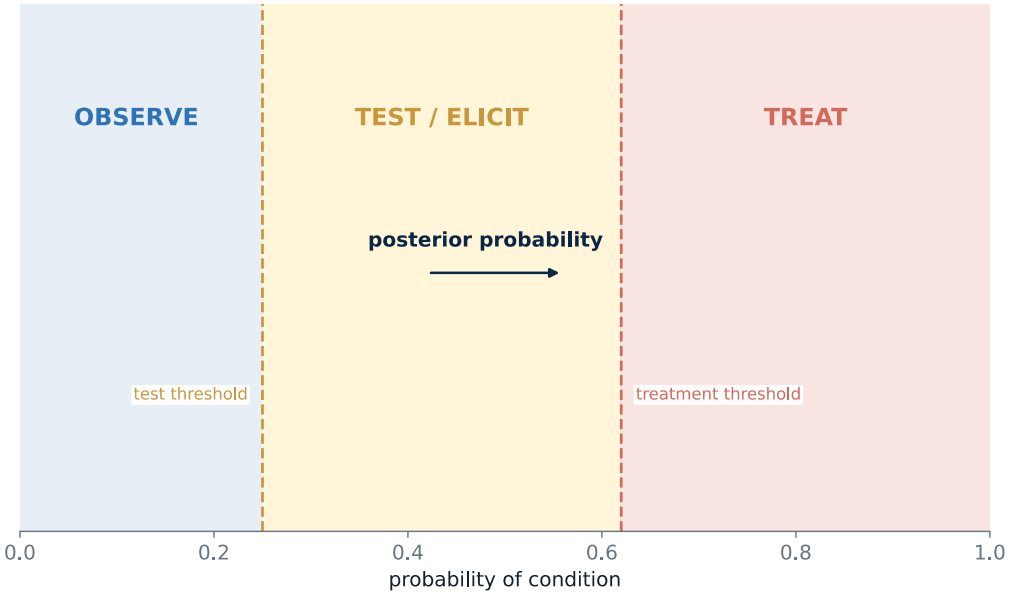


Figure 19. Two clinical thresholds divide posterior probability into OBSERVE, TEST / ELICIT, and TREAT regions.

Figure 19 illustrates how a probability estimate can be used in a decision policy. The thresholds separate observation, further testing or preference elicitation, and treatment. Their use depends on uncertainty and the patient's preferences.

Diagnostic and predictive models

Sensitivity and specificity condition on disease status; positive and negative predictive values depend on prevalence. Discrimination measures ranking. Calibration requires predicted probabilities to match outcome frequencies in the intended population, period, and decision range: among cases assigned probability p, the outcome frequency should be approximately p. Decision-curve analysis evaluates net benefit across plausible thresholds and can show that a model adds no value over treat-all or treat-none strategies (Vickers and Elkin 2006). Good discrimination alone cannot justify an action threshold.

Dataset shift, missingness, verification bias, missing follow-up, treatment-confounder feedback, and unequal measurement threaten validity. Subgroup performance, workflow effects, and alert fatigue also determine clinical value. The evidence chain must extend through workflow to monitored outcomes (World Health Organization 2021).

An AI system may optimize an intermediate endpoint while worsening care through delay, automation bias, or fragmented responsibility. Prospective evaluation should compare workflows, not only algorithms (Rajkomar, Dean, and Kohane 2019; Topol 2019). Human override is not a safety mechanism unless users have time, information, authority, and feedback.

Treatment choice and preference sensitivity

Randomized trials estimate average causal effects under eligibility and protocol conditions. Individual choices may depend on heterogeneous effects, absolute baseline risk, interactions, competing risks, and treatment burden. Shared decision-making is especially important when options have similar survival but different quality-of-life profiles. Framing can change choices even when outcomes are numerically equivalent (McNeil et al. 1982).

Shared decision-making elicits values that affect the utility function and sometimes the feasible set. For example, one patient may prioritize survival probability, while another prioritizes cognitive function, independence, fertility, pain, or avoiding hospitalization. We therefore need to check whether a population-level recommendation is compatible with the individual's preferences.

Quality-adjusted life years combine duration and health-state utility,

$$Q = \int_0^T u\,(t)\,dt, \tag{93}$$

but utilities vary across people and contexts. Disability, age, and chronic illness raise ethical questions that a scalar cannot settle. Clinical constraints, rights, and individual preference should remain visible.

Sequential care

Care is a partially observed sequence: investigate, act, observe, and revise. Dynamic treatment regimes and adaptive trials formalize this sequence. Real care also includes adherence, access, caregiver burden, and changing goals. A recommendation must also be accessible, affordable, understandable, and sustainable if it is to provide the intended benefit.

Treatment changes both the patient's state and the information available for the next decision. A response can reveal biology; an adverse effect can exclude a later option. A treatment policy therefore needs the clinically relevant history or belief state as well as the latest measurement.

In an insulin-delivery system, the controller balances time in target range, hypoglycemia, hyperglycemia, variability, burden, and device constraints. Meals, exercise, illness, sensor delay, and physiology create uncertainty. Safety layers, alarms, manual override, and fallback basal delivery are part of the controller. An objective optimized in simulation must be validated across representative people and exceptional conditions.

We also include stopping among the possible clinical actions. A policy should specify when to withhold, refer, obtain information, or defer to a human. Models trained only on cases where treatment proceeded can be weakest exactly at abstention boundaries.

Validate the action policy's follow-up, capacity, subgroup performance, patient understanding, and appeal. Responsibility remains with the authorized people and care process.

Public health and allocation

Public health allocates attention and resources across prevention, surveillance, treatment, workforce, infrastructure, communication, and social determinants. Outcomes emerge through populations and networks, so individual and collective optima may diverge.

Allocation under scarcity

A health system might maximize total expected health gain

$$\max_{x} \sum_{i=1}^{n} B_i(x_i) \tag{94}$$

subject to budget, staff, beds, geography, eligibility, and minimum-service constraints. The objective immediately raises questions: Are benefits measured as deaths averted, quality-adjusted life years, unmet need, financial protection, or capability? Are gains weighted by severity, disadvantage, or reciprocity? Cost-effectiveness supplies evidence, not a complete allocation rule (Weinstein et al. 1996; Hunink et al. 2014).

Population benefit can conflict with equity, geographic access, and priority for the worst off. A QALY can support comparison but remains ethically incomplete as a sole allocation rule (Neumann et al. 2016).

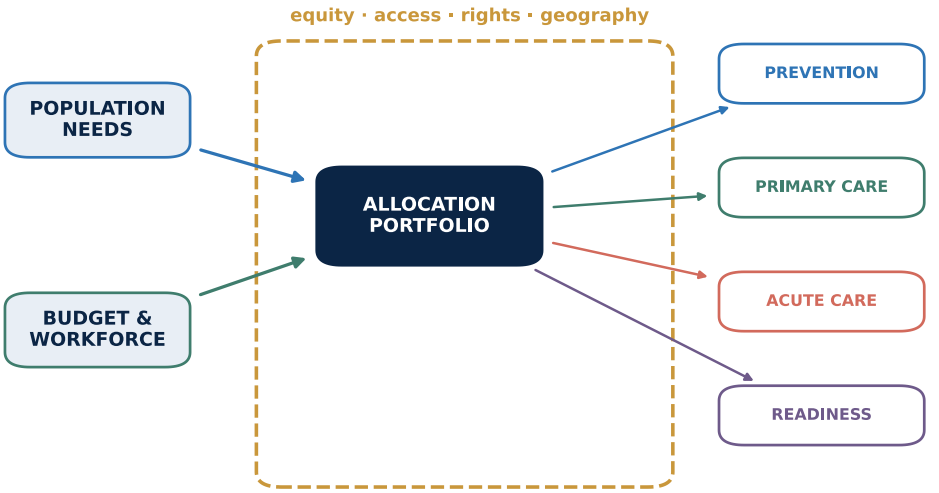


Figure 20. Health-system allocation from needs, budget, and workforce to prevention, primary care, acute care, and readiness, governed by equity, access, rights, and geography.

Figure 20 shows related uses of health-system resources. Prevention, primary care, acute care, workforce, and readiness contribute jointly to outcomes. Geography, equity, access, and rights also constrain their allocation.

Queueing models help size clinics, operating rooms, emergency departments, and laboratories. Minimizing average wait may worsen the longest waits or displace complex

cases. Triage rules must be auditable and clinically grounded. Capacity buffers that appear idle are insurance against outbreaks, seasonal peaks, and correlated failures.

Screening and prevention

Screening can advance diagnosis without improving outcomes because of lead-time, length, and overdiagnosis biases. Program evaluation must include downstream confirmation, treatment, false positives, missed disease, participation, and unequal access. A test optimized in a case-control sample may perform poorly in population use.

Vaccination and infection control create externalities: one person's action changes others' risk. Compartmental epidemic models approximate transitions among states, while contact networks capture heterogeneity. Optimal intervention timing depends on growth, delay, behavior, health capacity, and uncertainty. Robust policies and transparent communication can be more valuable than a brittle point forecast.

Operations and policy

Hospital optimization covers rostering, bed assignment, theatre scheduling, inventory, referral, and discharge. These interact: maximizing theatre utilization can create recovery-bed congestion; minimizing length of stay can increase readmission or caregiver burden. System-wide objectives and causal evaluation are essential.

Health policy should be monitored for distributional effect. A digital-first service may lower average transaction cost while excluding people without devices, language support, stable housing, or trust. WHO guidance for AI in health emphasizes autonomy, well-being, transparency, accountability, inclusiveness, and sustainability (World Health Organization 2021). Regulation and post-market surveillance add lifecycle obligations (U.S. Food and Drug Administration, Health Canada, and Medicines and Healthcare products Regulatory Agency 2021).

Preparedness

Preparedness preserves response options through stockpiles, flexible contracts, interoperable data, surge staffing, mutual aid, redundant laboratories, adaptable facilities, and exercises. These resources may be unused during quiet years. Assess essential functions, safe degradation, recovery, and learning rather than classifying all low utilization as waste.

A screening program contains several dependent decisions. The test threshold affects referrals, capacity affects waiting, and waiting may affect outcomes. Outreach changes who is observed. False positives may lead to procedures and anxiety, while false negatives may delay care. Increasing sensitivity without sufficient capacity may therefore worsen the care pathway. We follow the patient through confirmation, treatment, and follow-up when defining the model boundary.

Biomedical discovery and translation

Biomedical optimization connects molecules, devices, data, trials, regulation, manufacturing, and care. Its search spaces are vast and its measurements expensive. Biological variability and high consequence make validation central.

Drug discovery and development

A candidate drug must balance potency, selectivity, exposure, toxicity, stability, formulation, manufacturability, and clinical differentiation. Multi-parameter optimization often uses desirability functions or Pareto analysis, but correlated assay error and selection bias can produce the optimizer's curse. Active learning should choose experiments that reduce decision uncertainty, not merely maximize a predicted score.

Surrogate and generative models can propose drug candidates, but uncertainty and synthesizability limit their value. Experiments should reduce uncertainty about the decision, including reasons to stop development.

Dose selection can be modeled through exposure-response and toxicity. A simplistic utility is

$$U(d) = B(d) - \lambda H(d), \tag{95}$$

where benefit B and harm H are uncertain and vary between people. At a development stage, new evidence may support continuation, redesign, partnership, or stopping. We can consider these as real-option decisions. Portfolio evaluation also needs to retain negative information that incentives might otherwise encourage participants to withhold.

Adaptive trial designs may alter allocation, sample size, arms, or population under prespecified rules. They can improve information and participant benefit but require careful control of error, operational bias, and interpretation. Surrogate endpoints shorten trials only when they reliably predict outcomes that matter.

Medical devices and prostheses

Devices optimize physical performance together with biocompatibility, usability, sterilization, battery life, cybersecurity, alarm behavior, and service. A prosthesis or implant is part of a person's body and routine; comfort, identity, maintenance, and learning can dominate laboratory efficiency. Human factors validation must include realistic environments and foreseeable misuse.

Closed-loop devices—pacemakers, insulin delivery, neurostimulation—combine sensing, estimation, control, and failsafe behavior. Personalization can improve control but creates verification challenges. Safe fallback modes and bounded adaptation are design variables.

Bioinformatics and systems biology

Sequence alignment, assembly, phylogenetics, variant calling, protein structure, network inference, and single-cell analysis optimize likelihoods, scores, or posterior probabilities. The objective embeds a biological error model. For sequence alignment, gap penalties express

hypotheses about insertion and deletion; for phylogeny, substitution models and priors shape the tree.

High-dimensional molecular data invite multiple testing and leakage. Family-wise error, false discovery rate, replication, batch correction, and preregistered analysis separate signal from selection. Mechanistic and statistical models can be composed: physics-informed learning constrains prediction with differential equations (Raissi, Perdikaris, and Karniadakis 2019).

An optimal alignment remains an inference under its scoring scheme. Report ambiguity and alternative solutions when downstream conclusions depend on them.

Translational evidence

Performance often falls between bench, animal model, trial, and routine care because populations, endpoints, operators, and incentives change. Moves between development phases should be treated as a sequence of domain shifts with explicit go/no-go criteria. Recording negative results and uncertainty allows them to inform later decisions.

Repeated selection can favor candidates that perform well in particular assays or animal models because of measurement error or model-specific conditions. Before clinical use, we need evidence about mechanism, exposure, endpoints, population, and care context. Prospective validation, preregistration, independent replication, and reporting of failed candidates help us examine the optimism introduced by selection.

Biomedical discovery proceeds through target identification, assays, hit selection, optimization, preclinical studies, trials, review, manufacturing, and monitored use. Selecting on an early proxy may remove useful candidate diversity or favor assay error. We therefore retain independent types of assays, mechanistic checks, negative results, and replication evidence across these stages.

Portfolio optimization should account for correlation among candidates, platform dependencies, stage-specific cost, and value of information. Stopping work on a candidate can be a useful decision if it prevents later harm or releases capacity. Publication and incentive systems should not reward only positive progression.

Translation to populations requires external validation, causal evidence, benefit-harm assessment, manufacturability, access, and monitoring. Statistical normality is not health (Canguilhem 1991). Patient goals and capability remain part of the objective system.

Field checklist

- Treat biological fitness as contextual and historical, not as moral worth or global purpose.
- Separate prediction, utility, capacity, and action in clinical models.
- Calibrate probabilities in the intended population and decision-relevant range.
- Include abstention, referral, override, and safe fallback in sequential policies.
- Follow screening and allocation through the full care pathway and distribution of effects.
- Value preparedness options that nominal utilization metrics call idle.
- Require prospective and translational evidence after computational candidate selection.

Translation lens. Biological and health applications share adaptation, conservation, thresholds, policies, and allocation structures. Each application still requires evidence about its biological mechanisms, patient preferences, clinical effects, consent, and distribution of benefits and harms.

Chapter 13 · Economics, operations research, public policy, law, and education

Markets, finance, and endogenous organizations

Economics studies allocation under scarcity. We can use its optimization models to describe choices or design institutions, while retaining the assumptions behind those uses. Preferences may be incomplete, context-dependent, or socially formed, and purchasing power differs. Institutions also determine which choices enter the model. A result under these assumptions does not by itself establish the best social arrangement.

Consumers, firms, and equilibrium

A textbook consumer chooses bundle x to maximize utility under a budget, with price vector p and income m:

$$\max_{x \geq 0} u(x) \quad \text{subject to} \quad p^T x \leq m. \tag{96}$$

A firm chooses inputs and outputs to maximize profit. Prices coordinate these decisions in competitive models. General equilibrium asks whether individually optimizing choices and market clearing can coexist. Welfare theorems require strong assumptions and do not select a distribution of endowments. Arrow's result shows that no social-welfare aggregation rule can satisfy a set of plausible conditions simultaneously when there are at least three alternatives (Arrow 1951).

Pareto efficiency means no one can be made better off without making someone worse off (Pareto 1896). It is silent about inequality: both egalitarian and extremely unequal allocations can be Pareto efficient. Distribution, rights, public goods, externalities, information, and market power require additional institutions.

Strategic behavior and mechanism design

Chapter 6 defines Nash equilibrium and its mixed-strategy existence guarantee. Equilibrium is a consistency condition under specified rules, not necessarily an efficient, fair, or dynamically reachable outcome (Nash 1950). Mechanism design works backward from desired properties to rules under strategic reporting. Myerson's auction analysis concerns revenue-optimal design (Myerson 1981). In bilateral trade with independent private values and overlapping supports, the Myerson-Satterthwaite result rules out simultaneously achieving efficiency, Bayesian incentive compatibility, interim individual rationality, and budget balance (Myerson and Satterthwaite 1983). Privacy, simplicity, and access add further design questions. Stable matching establishes another property: absence of a blocking pair under the model (Gale and Shapley 1962).

Behavioral and institutional realities

Expected-utility models are useful baselines, while prospect theory models reference dependence, loss aversion, and nonlinear probability weighting (Kahneman and Tversky 1979). Bounded rationality emphasizes limited information and computation; satisficing may be more realistic than global maximization (Simon 1955). Rules, norms, common-property arrangements, and trust can govern resources without a simple state-versus-market dichotomy (Ostrom 1990).

Finance and risk

Mean-variance portfolio theory chooses weights w by trading expected return $\mu^T w$ against variance $w^T \Sigma w$:

$$\min_{w} \; 1/2\, w^T \Sigma w - \lambda \mu^T w, \qquad \mathbf{1}^T w = 1. \tag{97}$$

It established diversification as an optimization problem (Markowitz 1952), but estimated returns are unstable and covariance changes during stress. Robust allocation, scenario analysis, liquidity limits, transaction costs, leverage, taxes, and tail risk matter. Conditional value at risk averages losses beyond a quantile and supports convex formulations (Rockafellar and Uryasev 2000).

Preferences, endowments, information, institutions, and power can be endogenous: the intervention changes them. Social evaluation therefore needs criteria beyond consistency of exchange or Pareto efficiency (Sen 1970).

Finance makes estimation risk visible. Mean-variance optimization is highly sensitive to expected returns; the optimizer can amplify small errors into extreme portfolios (Michaud 1989). Constraints, shrinkage, robust uncertainty, transaction costs, liquidity, and stress scenarios often improve out-of-sample decisions. The optimizer's curse arises because selecting the best estimated strategy also favors estimation error (Smith and Winkler 2006).

A rule can change how participants report information and act. We therefore evaluate auctions, matching procedures, platform rankings, taxes, and service eligibility under the responses they may produce. The criteria include incentives, participation, budget, privacy, simplicity, fairness, and adaptation. A mechanism designed for truthful reports may fail if truthfulness is not an equilibrium or if participants discover workarounds.

A formally strategy-proof mechanism can remain inaccessible or politically illegitimate. Evaluate the reporting and participation demands of a rule as well as its equilibrium properties.

Prices reflect scarcity under particular conditions of ownership, income, information, and power. They provide useful information but do not measure all social value. For example, a benefit-cost calculation based on willingness to pay can give greater weight to wealthier people. We also report distribution and capability, and retain rights and public duties outside monetary substitution (Sen 1985; Rawls 1971).

Shadow prices from an allocation model are likewise conditional. They show marginal value under the objective, constraints, and data. Additional reasoning is needed before using them to set a fee, wage, compensation, or moral priority. Translating a multiplier into policy requires legal and institutional reasoning.

Use market mechanisms where participants can understand and act, externalities are governed, and concentration or strategic power is addressed. Otherwise the mechanism may optimize exchange within a distorted boundary.

Operations research as institutional engineering

Operations research turns recurring organizational decisions into models: what to produce, stock, route, schedule, staff, locate, or maintain. Its distinctive strength is connecting mathematical structure to operational evidence and implementation.

Flow, inventory, and service

Network-flow models represent supply, demand, capacity, and conservation. Facility-location models trade fixed opening costs against transport and service. Vehicle-routing models add sequence, capacity, time windows, driver rules, and uncertain travel. Inventory control balances holding, ordering, shortage, spoilage, and service; multi-echelon systems coordinate stock across a network.

A simple newsvendor chooses quantity q before random demand D:

$$\min_{q} c_o \, \mathbb{E}[(q - D)_+] + c_u \, \mathbb{E}[(D - q)_+]. \tag{98}$$

Under continuity, an optimal order quantity satisfies the critical-fractile condition

$$F_D(q^*) = \frac{c_u}{c_u + c_o}. \tag{99}$$

The result relates the service level to the specified costs. If these costs omit or misrepresent a consequence, the selected stock policy may be inappropriate even when the calculation is correct.

Inventory provides another example of a model boundary. A classical formulation balances holding cost against ordering or shortage cost. Stock also affects resilience, obsolescence, working capital, service differences, supplier risk, and future options. A just-in-time policy may perform well under ordinary variation and fail under a common-cause disruption. We therefore distinguish items by the consequences of shortage and uncertainty in replenishment time.

Scheduling and projects

Scheduling allocates machines, people, rooms, vehicles, or computing resources over time. Objectives include makespan, lateness, throughput, fairness, setup, and robustness. Exact models may be MILP or constraint programs; dispatching rules and metaheuristics offer

speed. A schedule should include recovery mechanisms because processing times, absences, and arrivals change.

Scheduling illustrates hidden human constraints. A mathematically feasible roster may violate preferences, recovery, skill development, predictability, or informal coordination. Some constraints are legal or safety-critical; others should be elicited and prioritized. Publishing a schedule changes swapping and absence behavior. Pilot implementation and operator review are part of the solution.

Project optimization uses precedence networks, critical paths, resource constraints, and optional crashing. Optimizing the baseline plan is less important than managing uncertainty and feedback. Excessively tight utilization removes the slack needed to resolve surprises.

Project optimization similarly requires managing uncertainty. A deterministic critical path can create false precision; correlated delays, rework, resource contention, and learning matter. Buffers should be located according to risk and coordination, not simply removed as waste. A late change can alter scope and the precedence graph itself.

From model to decision

An operations model has a data pipeline, ownership structure, and cadence. Master data errors can dominate solver gaps. Planners may hold tacit constraints absent from the model. Good deployment captures overrides and reasons, updates forecasts, and makes infeasibility informative.

Scenario optimization and simulation address uncertainty. Digital experiments can compare policies, but validation must cover arrival patterns, service times, correlations, behavioral responses, and rare disruptions. A solution should be stress-tested for parameter sensitivity, not merely replayed on average history.

Organizational incentives

Performance indicators affect which work receives attention. For example, minimizing purchase price may increase defects, lead time, or working capital. Minimizing call-handling time may increase repeat calls. We need measurements of the whole process and constraints on omitted consequences. We also examine how participants may change their behavior in response to the indicators.

Policy, law, delivery, and accountability

Social optimization changes systems whose participants interpret, resist, learn from, and modify the intervention. We need to include their rights and agency in the formulation. Predictions and rankings can themselves change subsequent behavior.

Causal policy questions

Policy asks what would happen under an action. In potential-outcome notation, the average treatment effect is

$$\mathrm{ATE} = \mathbb{E}[Y(1) - Y(0)]. \tag{100}$$

Only one potential outcome is observed for each unit. Randomization, natural experiments, adjustment, and structural models identify effects under different assumptions. A model that accurately predicts unemployment does not necessarily reveal which intervention reduces it.

Optimization on observational data can exploit confounding. Policy should state the estimand, intervention, population, time horizon, spillovers, and identification assumptions. Distributional effects and qualitative evidence matter alongside averages.

Predictive targeting estimates who is likely to experience an outcome under historical policy. Allocation needs the effect of an intervention and its opportunity cost. A high-risk person may be hard to help; a moderate-risk person may benefit greatly. These cases show why a ranking by risk can differ from a ranking by treatment effect.

Policy also changes behavior. Schools teach to measures, firms change reporting, landlords adjust, and agencies ration informally. Campbell's and Goodhart's warnings concern this endogenous response (Campbell 1976; Goodhart 1975). Build mechanisms and enforcement into scenarios. Use pilots and qualitative observation to find adaptations before scaling.

Historical policy and access shape the observed data. A correlation-based allocation can withhold help from people whose outcomes appear good precisely because they already received support. This endogeneity makes the causal estimand and counterfactual essential.

Policy design and administration

Policy portfolios combine taxes, subsidies, standards, information, public provision, and institutional reform. Compliance cost, enforcement, administrative capacity, legitimacy, and political durability affect results. Adaptive policy uses monitoring and predetermined review points without surrendering long-term commitments.

Policy models often assume implementation capacity. Staff, procurement, data, local partners, legal authority, communication, maintenance, and trust determine whether an intervention exists in practice. Include capacity constraints and investments that expand them.

A portfolio that funds programs without administration can be infeasible. A program that relies on local discretion can need training and audit. A policy with a rapid benefit can be dominated by a slower one if the first cannot be delivered equitably.

Implementation capacity can change over time. Emergency work may consume resources used for routine service, repeated reforms may burden staff, and outsourcing may reduce knowledge retained by the institution. We therefore include slack and learning, and measure delivery quality and staff workload as well as allocated funds.

Rankings change organizations. When professional schools and universities are compared on a public ranking, they redirect effort, manage reported categories, and alter admissions and organizational practice (Espeland and Sauder 2007). Campbell's and Goodhart's laws warn

that a measure used as a target becomes vulnerable to corruption (Campbell 1976; Goodhart 1975). Auditing must look for displaced work and excluded populations.

Law as constraint and objective

Law defines rights, duties, procedures, and remedies; it is not reducible to aggregate efficiency. Proportionality, due process, equality, privacy, and reason-giving constrain administrative optimization. Automated systems used in benefits, policing, hiring, credit, or migration must support notice, contestation, records, and accountable authority.

The European Union's AI Act takes a risk-based lifecycle approach to regulated AI uses (European Parliament and Council of the European Union 2024). Compliance is not identical to ethical adequacy, but legal requirements are hard constraints. A policy cannot purchase a right violation with sufficient predicted benefit.

A legal threshold can be categorical even when evidence is probabilistic. A model may support consistent application while remaining subordinate to interpretation and appeal.

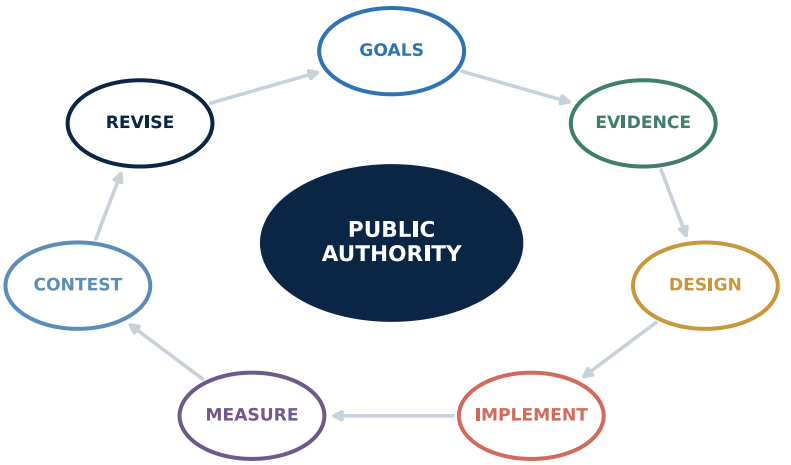


Figure 21. A governance loop linking public goals, causal evidence, policy design, implementation, measurement, contestation, and revision.

Figure 21 illustrates feedback in public policy. Public goals and causal evidence inform the design. Implementation produces new observations, and challenges to the decision may identify a need for revision.

Notice, explanation, participation, hearing, and review require time. These procedures also support legitimacy, error correction, and information gathering. We therefore include their functions when optimizing an administrative process. Reducing processing time by removing a review procedure may also remove a safeguard.

Delivery and contestability

A policy is implemented through legislation, budgets, guidance, procurement, organizations, frontline discretion, communication, take-up, and enforcement. Each stage can change the intervention. We therefore need to model delivery as well as the statutory rule.

For a proposed benefit, regulation, placement system, or public service, map the chain and assign evidence. Eligibility rules require administrative data and legal review. Take-up requires access, trust, time, language, and documentation. Provider response requires capacity and incentives. Outcomes require causal and distributional analysis. Appeal and correction require operations of their own.

Optimization can allocate budgets or choose parameters after this chain is modeled. It should include administrative burden, waiting, non-take-up, error, discretion, and enforcement—not merely program spending. When a policy shifts paperwork and uncertainty to vulnerable people, those costs need to remain inside the social evaluation boundary.

Legal requirements also affect the model's interfaces. Eligibility, nondiscrimination, notice, reason-giving, hearing, privacy, proportionality, and review may need explicit constraints, processes, and records. A weighted ethics term cannot be assumed to preserve these requirements.

Contestability needs capacity: people must know a decision occurred, understand enough to challenge it, obtain relevant data, reach a human with authority, and receive timely remedy. An explanation alone does not provide this correction procedure. Measure appeal accessibility, resolution, reversal, delay, and repeated error.

Automation can make inconsistent decisions easier to detect and can standardize their treatment. It can also apply a rule too rigidly when context permits an exception. We retain discretion where law or justice requires it and monitor how it is used. Eliminating visible discretion may otherwise leave the same policy choices hidden in the model.

Ranking and triage

A ranking converts several dimensions of need or merit into an order. It may help allocate review, but nearby ranks can appear more different than the evidence supports, especially around a cutoff. We therefore state the criteria and uncertainty and inspect near-ties. A rank describes the result of that comparison, not a person's intrinsic worth.

Triage should include urgency, potential benefit, alternatives, waiting harm, and equity under a clinically or institutionally valid policy. It must support reassessment as state changes. The position in the queue should therefore be open to reassessment.

People need notice and correction where ranking affects material opportunity. Audit missing data, representation, subgroup error, and strategic behavior. When capacity shortfall drives harm, do not let the ranking model absorb responsibility for the shortage.

Education and measurement response

Education involves learners, teachers, families, institutions, and long horizons. Its aims include knowledge, skill, capability, curiosity, belonging, creativity, citizenship, and opportunity. Test scores sample selected outcomes and can support diagnosis. As high-stakes targets, they can change teaching, enrollment, classification, and exclusion (Campbell 1976).

School assignment and admissions also combine capacity, preferences, proximity, diversity, siblings, access, and strategic behavior. Stable matching can improve specified process properties while leaving the policy choices contested. Teachers and learners need intelligible recommendations and a way to challenge data.

Adaptive learning systems can sequence material using estimated mastery, while accounting for changes in the learner and context. Motivation, identity, social interaction, fatigue, and misconception matter. A short-term answer-rate objective can prefer easier content and reduce durable learning. Evaluate both measured attainment and the conditions that produce it, including transfer, retention, subgroup experience, teacher workload, and the possibility of independent learning.

We need to distinguish the need for support from the predicted gain after support. Maximizing total measured gain may favor students who are easier to help and leave less provision for those who need more assistance. Equity requirements may therefore take the form of minimum service levels, priority weights, or rights to provision.

An allocation model should include service capacity, student and family burden, peer and neighborhood effects, accessibility, continuity, and uncertainty. Randomized or quasi-experimental evidence can estimate some interventions; implementation and local meaning affect transportability. Avoid ranking students or schools as if predicted outcome were intrinsic worth.

We can use optimization to construct feasible resource alternatives and examine trade-offs while retaining public authority over educational purpose. Measured gains represent only part of a school’s contribution.

Worked policy formulation: heat-health response

A city heat-health program chooses cooling-center locations, opening hours, transport, outreach, home visits, tree and shade investment, energy assistance, and emergency staffing. The objective system includes mortality and morbidity, access, cost, emissions, trust, and long-term resilience. Constraints include staff, buildings, electricity, transport, disability access, data protection, and legal duties.

Exposure and vulnerability are uncertain and unequally measured. Mobile-phone data can estimate movement while excluding or surveilling vulnerable people. Historical emergency calls may underrepresent those unable to seek help. Participatory mapping and local organizations provide evidence that remote sensing cannot.

The operational model can optimize daily activation under forecasts; the planning model can optimize permanent infrastructure; the governance process determines priorities and data use. Monitoring triggers should include indoor temperature, service uptake, response time, neighborhood coverage, and adverse events. A post-event review updates both parameters and the intervention portfolio.

Field checklist

- Treat prices, preferences, rules, and measured behavior as potentially endogenous.
- Include estimation uncertainty and selection bias in financial or policy optimization.
- Evaluate mechanisms under strategic response, participation, and distribution.
- Elicit human scheduling, workload, predictability, and recovery constraints.
- State the causal estimand and implementation pathway for every policy claim.
- Preserve due process, explanation, and appeal as safeguards, not default delay penalties.
- In education, distinguish measured gain from capability, inclusion, and durable learning.

Translation lens. Economic, organizational, and policy applications share allocation, flow, incentive, and causal-intervention structures. We need to redefine the relevant institutions, rights, behavioral responses, and procedures when transferring a formulation.

Chapter 14 · Psychology, ethics, sustainability, philosophy, history, and language

Human factors and function allocation

Human performance depends on the task and its context. People distribute attention, switch strategies, learn, become fatigued, and coordinate with others. Human reliability analysis estimates error under task, environment, and organizational conditions. We therefore test a design across these conditions. A design fitted to an average user may fail under high workload or when errors have serious consequences.

Bounded cognition

Humans often use heuristics because exhaustive search is costly and the world is uncertain. Some heuristics are biases in one environment and efficient adaptations in another (Gigerenzer and Selten 2002). Choice architecture influences decisions through defaults, ordering, framing, and friction. Ethical design makes important consequences visible and preserves meaningful choice.

A simple speed-accuracy trade-off treats decision threshold a as a variable: higher a gathers more evidence and usually improves accuracy while increasing response time. In safety-critical work, the optimal threshold changes with urgency and loss. Training should develop flexible calibration rather than one fixed response style.

Bounded rationality includes the cost of information and search (Simon 1955). A heuristic can be ecologically rational when it uses environmental regularities (Gigerenzer and Selten 2002). Compare it with complex models under realistic data and attention costs. Additional precision has little value when it cannot change the action.

A system should not compare human action with an omniscient optimum and label the difference error. Design information, defaults, timing, and choice sets to support actual decision strategies without manipulation (Payne, Bettman, and Johnson 1993).

Human-machine systems

Automation changes work instead of merely removing it. It can reduce routine load while leaving rare, ambiguous, high-stakes exceptions. Poor function allocation creates distinct failure mechanisms. Mode confusion concerns hidden or misunderstood automation states (Sarter and Woods 1995); out-of-the-loop performance combines loss of situation awareness and skill under passive control (Endsley and Kiris 1995); overreliance and alarm-driven disuse require separate analysis of automation use (Parasuraman and Riley 1997). Displays should support situation awareness—perception of relevant state, comprehension, and projection—without treating that model as evidence for every automation failure (Endsley 1995).

“Human error” prompts analysis of interfaces, procedures, staffing, incentives, fatigue, and recovery opportunities. Rasmussen’s skill-rule-knowledge taxonomy distinguishes cognitive

control modes (Rasmussen 1983). His later account of migration toward performance boundaries examines adaptation across operating, managerial, and regulatory levels (Rasmussen 1997). These accounts describe related but distinct mechanisms.

Ergonomics and inclusive design

We can distinguish physical, cognitive, and organizational ergonomics. Physical ergonomics concerns posture, force, repetition, reach, vibration, lighting, and the environment. Cognitive ergonomics concerns memory, attention, mental models, alarms, and coordination. Organizational ergonomics includes schedules, teamwork, and culture. A change at one level may transfer effort or strain to another, so we examine their interaction.

Inclusive design treats variation in bodies, senses, language, cognition, and technology as a design input. Accessibility features such as captions, contrast, keyboard control, clear language, and adjustable pacing often help broad populations. Optimization should consider the tails of capability and context.

Interfaces and interaction cost

Keystroke-level and pointing models estimate the time required for routine actions (Card, Moran, and Newell 1983). The user may also need time to resolve uncertainty, recover from interruption, or check a result. For example, a one-click action may be quick but easy to trigger accidentally. A confirmation at the appropriate point may prevent an error. Usability testing helps us observe the strategies people actually use (Norman 2013).

Human factors optimization studies a joint cognitive, social, and technical system. A design that assumes perfect attention, memory, reaction, or compliance omits relevant human constraints. People adapt, develop expertise, create workarounds, coordinate informally, and reinterpret goals. These capacities can stabilize a brittle system and can also conceal its risk until conditions change.

Interaction cost includes time to detect, understand, decide, act, and recover. Accessibility changes feasibility: test sensory, language, dexterity, cognitive, and assistive-technology conditions rather than treating them as an average-user penalty.

We also need to examine the human task left after automation. If routine cases are removed, the remaining cases may be more difficult and less frequent, while operators receive less practice. Approval after a recommendation may be affected by anchoring or workload. If control is returned only when the system becomes uncertain, the person may receive it when the situation is hardest to understand. Task speed alone does not describe these effects.

A function-allocation design should specify authority, information, timing, skill maintenance, override, responsibility, and fallback. It should simulate degradation and disagreement. Operators should be able to inspect why a constraint is binding, which data are uncertain, and what alternatives exist. We evaluate explanations by whether they support the required action.

Participatory design can extend the formulation. Users and affected groups may identify hidden work, values, and failure modes. Their input can change variables, the system boundary, ownership, or the decision to build the system. Collecting preference weights after the architecture is fixed captures only part of this contribution.

Measure cognitive work at the point of use. How many signals must be integrated, how often, under what interruption and consequence? A dashboard that exposes every variable can reduce understanding. Progressive disclosure, stable terminology, external memory, and rehearsal can improve control. Automation should reduce low-value load while preserving situation awareness and skill for degraded modes (Endsley 1995).

Ethics, fairness, and moral uncertainty

Ethical analysis examines the objectives and constraints, who has standing, which procedures are legitimate, and which values may be traded. These questions affect the formulation and its use. We cannot assume that one adjustable weight represents all of them.

Consequences, duties, virtues, and capabilities

Consequentialist reasoning compares outcomes; deontological reasoning emphasizes duties and persons; virtue ethics emphasizes character and practical judgment; capability approaches ask what people are substantively able to do and be (Mill 1863; Kant 1785; Aristotle 2002; Nussbaum 2011). Care ethics additionally examines relationships and dependence. These perspectives reveal different failure modes. A policy with high aggregate benefit may violate a duty; a formally equal rule may leave capabilities unequal.

Rawlsian reasoning considers basic liberties, fair opportunity, and the position of the least advantaged (Rawls 1971). A maximin objective,

$$\max_{x} \min_{i} u_i(x), \tag{101}$$

captures one protective intuition but not the full theory. Ethical frameworks should guide questions and institutions rather than be caricatured as formulas.

A single weighted sum cannot preserve all distinctions between outcomes, duties, rights, character, capabilities, care, and legitimate procedures. We need to state which relationships the formulation retains and which require a separate decision process.

Fairness in algorithms

Statistical fairness criteria include demographic parity, equalized odds, equal opportunity, calibration, and individual or causal notions. They condition on different variables and express different social views. When base rates differ, calibration and equalized error rates can be mutually incompatible except in special cases (Chouldechova 2017; Hardt, Price, and Srebro 2016). No metric resolves whether labels, groups, interventions, or institutions are just.

Selecting the smallest disparity after trying many definitions does not establish fairness. The chosen metric requires a theory of harm, a decision context, and institutional accountability.

Abstraction can hide the social system in which a model acts (Selbst et al. 2019). A lending model may equalize error while the credit product remains harmful. A hiring model may remove a protected attribute while proxies and past exclusion remain. Fairness work therefore includes problem formulation, data history, stakeholder participation, recourse, monitoring, and institutional alternatives (Barocas, Hardt, and Narayanan 2023).

Expanding the boundary can reveal interventions upstream of prediction: changing documentation, offering assistance, redesigning eligibility, or removing the decision altogether.

Define who is affected, which outcome and error matter, whether the model allocates opportunity or burden, and what alternative exists. Examine label quality, selection, measurement, exposure, and downstream response. Aggregate parity can hide intersectional or local harm; very small groups create uncertain estimates.

Access, notice, explanation, appeal, representation in design, and distribution of benefit remain part of fairness analysis. Some uses are unacceptable even when parity is achieved.

Procedural safeguards

Responsible optimization documents purpose, affected groups, evidence, uncertainty, alternatives, constraints, residual risk, ownership, appeal, and retirement. Independent review is important when builders benefit from deployment. High-impact decisions need understandable reasons and a route for correction.

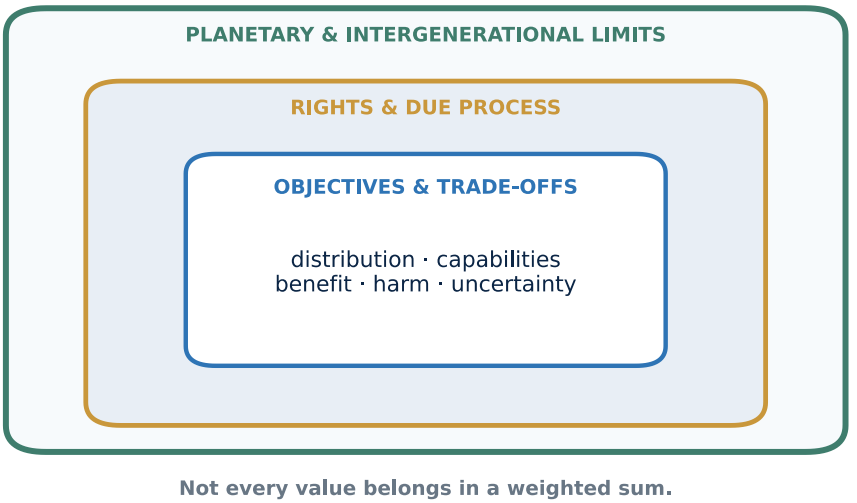


Figure 22. A responsible-optimization frame with rights as boundaries, stakeholder capabilities, distributional outcomes, and planetary constraints.

Figure 22 places rights and planetary limits outside ordinary compensation, while stakeholder capabilities and distributional outcomes remain visible inside the admissible region.

Participation and moral uncertainty

Participation can influence the boundary, alternatives, harms, evidence, and governance. It therefore includes more than a survey for fitting weights. Interviews, workshops, community research, deliberative panels, accessibility testing, and representative institutions provide different forms of input and support different claims.

Document who participated, who could not, compensation, language and access, information supplied, disagreement, and how input changed the model. Do not claim community consent from a small advisory process. Some groups have legal or sovereign authority beyond stakeholder preference.

Model outputs can support participation by making constraints and trade-offs visible. Use understandable scenarios and concrete alternatives. Preserve minority reports and allow participants to challenge the categories themselves.

Parameter uncertainty concerns quantities within a model. Moral uncertainty concerns the principles or reasons that should guide the decision. We can use probabilities over ethical theories as an analytical exercise, but such a representation does not determine which decision procedure is legitimate.

Use robust ethical reasoning where possible: identify actions unacceptable across several relevant frameworks, duties that remain regardless of aggregate benefit, and alternatives that preserve capabilities and reversibility. Where principles conflict, expose the conflict to accountable deliberation.

Ethics review should trace who benefits, who bears risk, whose agency changes, what rights apply, and whether repair is possible. It should examine process as well as outcome. A fair distribution produced through coercion or deception is not therefore justified.

Sustainability across generations

Environmental boundaries, social foundations, and intergenerational justice belong in sustainability analysis (Raworth 2017). Efficiency can increase total demand through rebound. Absolute budgets and sufficiency may therefore be needed alongside per-unit improvement.

Sustainability introduces long horizons, irreversibility, thresholds, distribution, and deep uncertainty. Discounting converts future effects to present values, but the rate combines empirical opportunity cost with ethical judgment. A single rate can conceal intergenerational disagreement. Report results across rates, physical thresholds, and noncompensatory constraints.

Planetary boundaries and safety limits should not be treated as ordinary weighted objectives when crossing them creates unacceptable or poorly reversible harm (Rockström et al. 2009; Intergovernmental Panel on Climate Change 2022b). Robustness, precaution, adaptive

pathways, and option preservation may dominate expected-value efficiency. Lifecycle analysis should include rebound, leakage, embodied effects, and distribution across places and workers.

Future people cannot participate directly in the present decision. Institutions may represent their interests through lasting standards, independent review, conservation duties, and the preservation of options. We need to state the limits of this representation.

Humanities, interpretation, and historical path dependence

The humanities examine meaning, value, interpretation, memory, and form. Humanistic inquiry selects archives, categories, narratives, and arguments without always seeking an extremum. Optimization can help identify patterns and compare candidate readings. We need to preserve ambiguities that belong to the work or its context, rather than automatically classify them as noise or reduce cultural significance to a score.

Philosophy of optimization

Aristotle distinguished productive making, theoretical understanding, and practical judgment. Modern optimization is strongest in production under stable ends; practical reason is needed when ends are contested. Means can also transform ends: a metric, platform, or institution changes what people notice and desire.

The concept of the normal has both descriptive and normative force. Medicine and society can mistake statistical typicality for desirable life (Canguilhem 1991). Technological arrangements distribute authority and possibilities for action (Winner 1980). Philosophy therefore asks who authored the objective, what ontology it presumes, and which counterfactuals it excludes.

Historical explanation

Historical actors optimize under beliefs, institutions, technologies, and moral worlds unlike ours. Retrospective claims that an outcome was “inevitable” confuse realized paths with feasible alternatives. Path dependence, contingency, archival silence, and changing categories limit simple objective functions.

Quantitative history can analyze prices, networks, texts, demography, and spatial patterns. Its models should be triangulated with sources and historiography. An optimized fit to surviving records may reproduce preservation bias. The absence of a surviving document is not by itself evidence that an event or practice did not occur.

Language and translation

Language technologies often maximize likelihood:

$$\max_{\theta} \sum_{t=1}^{T} \log p_\theta(w_t \mid w_{<t}), \tag{102}$$

but probable continuation is not the same as truth, relevance, or literary value. Translation trades semantic content, tone, rhythm, cultural reference, and audience. No single alignment score captures all of them.

Lexicography, corpus linguistics, authorship analysis, and distant reading reveal large-scale patterns (Moretti 2005). Close reading detects ambiguity, irony, voice, and form. The methods are complementary at different scales. Corpus construction and feature selection determine which patterns a model can represent. A topic model identifies distributions without establishing a theme's meaning; a network omits unrecorded influence. State what the representation retains and omits, then return to close reading and historical context to interpret the result.

Interpretation as constrained search

An interpretation must attend to the work, context, evidence, and alternative readings. It is neither arbitrary nor generally unique. Umberto Eco distinguishes openness from unlimited interpretive license (Eco 1990). One can model interpretation loosely as abductive search for an explanation with coherence and evidential fit, while preserving the fact that new contexts generate new questions. Disagreement among defensible readings can itself identify further questions and distinctions. Preserve excerpts, provenance, excluded material, and counterreadings rather than forcing convergence when ambiguity is constitutive.

Category-theoretic caution

We can borrow concepts from category theory to describe a translation between disciplines. First, we specify the objects, morphisms, and composition rules. We can then ask whether a proposed mapping preserves the required structure. For example, mapping fitness, utility, beauty, and loss to a scalar may remove distinctions needed in their original domains. In such a case, we describe the limited correspondence instead of assuming a functorial relationship. A language translation may preserve factual reference while changing connotation, rhythm, register, or social force. The proposed invariants might be chronological order, citation, syntax, or narrative function. A commutative diagram is useful when the corresponding morphisms identify a compatibility condition that can be checked.

Inherited choices and repair

The alternatives available today depend partly on earlier investment, exclusion, standards, and institutions. Optimization within that set can reproduce historical injustice while appearing neutral. We therefore also consider actions that repair or expand capacity, instead of limiting the decision to allocation of existing resources.

Historical data encodes previous policies. A model can accurately predict outcomes generated by unequal access and then justify the same allocation. Causal and historical inquiry asks how the pattern was made and which intervention can change it.

Path dependence also creates option value and lock-in. Standards, infrastructure, and categories become difficult to reverse. Long-horizon optimization should price transition, preserve interoperability, and ask whose knowledge or livelihood is displaced.

Field checklist

- Model human attention, expertise, adaptation, workload, and coordination as system variables.
- Test automation handover, disagreement, skill retention, and degraded modes.
- Include accessibility in feasibility and involve affected people before the objective is fixed.
- Distinguish consequences, rights, duties, capabilities, care, and procedural requirements.
- Justify fairness metrics through a harm model and institutional context.
- Test long-horizon decisions across discount rates, thresholds, and adaptive pathways.
- In humanistic work, preserve ambiguity and make corpus, category, and interpretation choices inspectable.

Translation lens. Human and interpretive disciplines use constrained search, evidence, and revision while sometimes retaining a partial ordering or several defensible alternatives. People remain participants in defining, interpreting, challenging, and changing the optimization process.

Part V · Design, Art, Music, Literature, and Play

Chapter 15 · Architecture, design, visual art, music, and performance

Creative practice: form, criteria, critique, and accessibility

Creative work can have a purpose while its criteria continue to develop. Artists and designers discover requirements through making and reviewing alternatives. We can use optimization to expand exploration, satisfy technical constraints, and examine trade-offs. A preference score remains a model of particular judgments and should be interpreted in that context.

Design as reflection and search

Herbert Simon described design as changing existing situations into preferred ones (Simon 1996). The meaning of preferred can change as the maker works with sketches, prototypes, users, sites, and materials. Reflective practice alternates framing and action rather than executing a complete specification (Schön 1983).

We can describe a design using several concerns, such as function, cost, carbon, enjoyment, access, and risk. Some can be measured more reliably than others. The vector helps organize their comparison without requiring every component to be a precise number. Pareto alternatives show trade-offs, and a new design concept may change which combinations are achievable.

Architecture

Architecture coordinates site, program, structure, envelope, energy, light, acoustics, circulation, code, cost, construction, and civic presence. Parametric design connects geometry to rules; performance simulation evaluates daylight, heat, airflow, views, structure, and egress; generative design searches variants (Woodbury 2010).

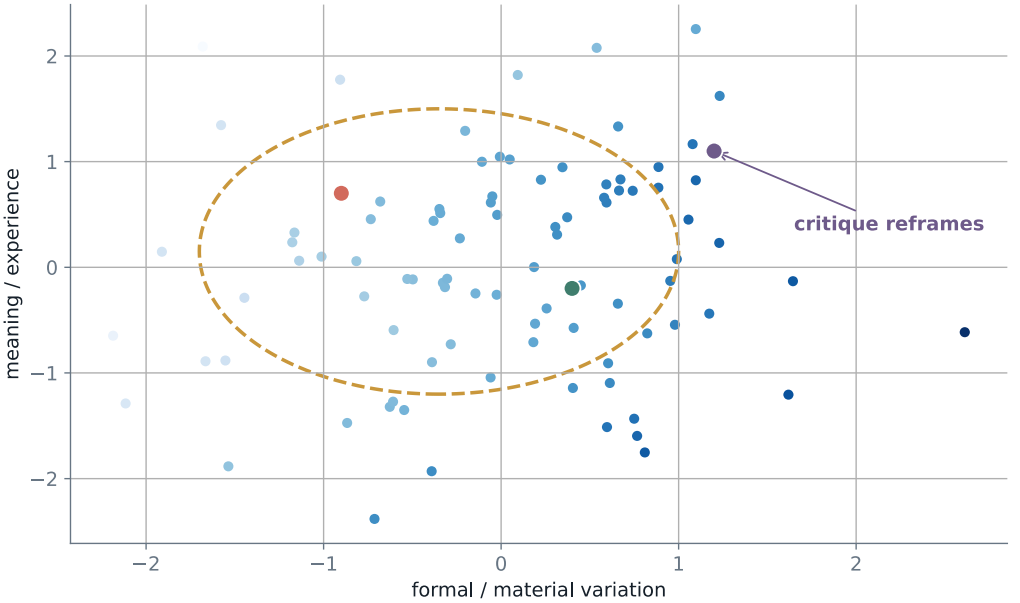


Figure 23. A creative design space in which constraints shape a field of possibilities and critique redirects the search.

Figure 23 shows how creative work can change both a candidate and its evaluation criteria. Constraints define the current possibilities, while critique can redirect the work and its formulation.

We interpret architectural objectives through the use and experience of the building. For example, increasing rentable area may reduce daylight, adaptability, or public space. Lower operational energy may require more embodied carbon. Shorter circulation routes may remove opportunities for encounter or ceremony. Building codes supply required constraints, while culture and place also need interpretation.

Christopher Alexander's early work treated design as decomposing interacting misfits (Alexander 1964). Decomposition helps computation but interfaces matter: structure shapes space, envelope shapes comfort, and circulation shapes social relation. The envelope cannot be optimized independently of structure and environmental systems. Integrated design sessions restore those couplings.

Visual art and computational creativity

Constraint has long structured art: perspective, proportion, palette, serial procedures, chance operations, and material limits. Evolutionary art and generative systems search image spaces through mutation, rules, or learned models (Sims 1991; Elgammal et al. 2017). Aesthetic measures can filter or provoke, but they reflect training data and taste communities.

Computational creativity may be evaluated by novelty, value, surprise, intentionality, and process (Boden 2004; Wiggins 2006; Colton 2008). These criteria are relational. A novel image

may be trivial; a derivative form may be culturally transformative. Authorship, provenance, consent, labor, and environmental cost belong in the system boundary.

A generative system can vary composition, palette, texture, camera, iconography, or style. A ranking model based on past popularity may favor what the archive made visible. Embedding distance does not establish cultural originality. If aesthetic quality is learned from anonymous preferences without their community and purpose, the model erases this context and reproduces the sample's visibility structure. Retain that structure and its limits when interpreting the score.

We therefore include provenance in the design requirements. A responsible creative system should disclose source permissions, transformations, attribution requirements, human contributions, and material or energy costs. Stylistic distance does not establish consent. A work can be statistically unlike a training example and still depend on an exploitative acquisition process. Likewise, an image can be legally usable and ethically inappropriate in a given cultural setting.

We also record objectives that the project declines to optimize. A museum may refuse to optimize visitor flow if the proposed route destroys contemplative time. An artist may refuse an engagement objective that rewards immediate recognizability. A community archive may refuse automated similarity labels for sacred or contested material. These decisions preserve values that the available objective cannot represent.

Critique as an optimizer

Studio critique samples how a work is perceived, identifies latent constraints, and proposes transformations. Different responses require interpretation before they can guide a revision. The maker decides which feedback reveals the work's internal necessity and which belongs to another project. This resembles interactive multiobjective optimization with a preference model that is being learned and revised.

Making an artifact may change the problem we are trying to solve. For example, an architect investigating circulation may begin to focus on how people encounter each other. A painter may retain an edge that was initially considered a defect. A composer may decide that a passage needs resistance instead of smoothness. Work with sketches, materials, performers, sites, and audiences can therefore change the concepts used to describe the objective.

Formal methods can support this process by generating contrasting alternatives, enforcing hard constraints, and showing dependencies or trade-offs. A provisional measure of taste needs to remain open to revision. Where comparisons are meaningful, the Pareto methods in Chapter 6 can show alternatives without fixing all preference weights in advance. The revision process in Chapter 3 is also relevant: the maker may change the criteria after examining the work they produce.

Three interacting creative loops

We can distinguish three interacting loops in the creative process.

1. The **artifact loop** changes a drawing, building, performance, text, image, or sound.
2. The **evaluation loop** changes how alternatives are compared: a criterion is added, dropped, reinterpreted, or moved from objective to constraint.
3. The **framing loop** changes what kind of project is being attempted, who it is for, and what evidence can count.

An ordinary optimizer changes the artifact under fixed criteria. Interactive multiobjective methods can also support changes in how alternatives are compared. Changes to the project itself involve authorship, dialogue, institutional authority, and ethical responsibility. For example, a proposal for a faster transit interchange may become a proposal for a civic meeting place. A prosthetic controller may become a musical instrument, or a visualization may become an argument about missing data. These changes affect the type of project being considered.

We can describe the distinction using category-theoretic language. Within a fixed framing, transformations between design states can be represented as morphisms. Reframing may change the category when the objects, admissible transformations, or invariants change. There may be a translation between the old and new descriptions, but it will forget information. Chapter 7's warning that argmin is not generally functorial becomes practical: the best component under one framing need not remain best after composition, and a minimizer need not survive a change in what counts as equivalent.

Architecture: objectives with different legal status

Consider an urban infill library. The team can simulate daylight, annual energy, structural demand, egress time, acoustic separation, embodied carbon, construction cost, and spatial visibility. These quantities should not all enter a weighted sum. Code compliance and structural safety are feasibility conditions. An accessibility commitment is a right and process requirement, not a benefit that high rentable area can purchase away. Carbon may be governed by a hard project budget and then improved within it. Cost is both a limit and a value requiring lifecycle interpretation. Civic presence, belonging, and the dignity of ordinary use require situated judgment.

The formulation can therefore be layered:

- first exclude illegal, unsafe, inaccessible, or materially infeasible schemes;
- then identify nondominated alternatives on operational energy, embodied carbon, cost, adaptability, and selected measures of comfort;
- then review those alternatives through spatial experience, community use, cultural context, maintenance, and future conversion;
- finally record why one alternative was chosen and what evidence would justify revision.

This hierarchy is lexicographic in part and deliberative in part. It is not reducible to a single scalar without losing the distinction among targets, constraints, penalties, vetoes, rights, and process requirements introduced in Chapter 1.

A parametric model is useful when its parameters correspond to decisions that can be controlled and built. We also need to represent construction methods, tolerances, maintenance access, and procurement. Otherwise, the model may generate a geometry that cannot be realized as intended. As in the co-design problems of Chapter 10, the building envelope cannot be optimized independently of structure, environmental systems, controls, and occupant behavior.

Preference learning and critique

Interactive optimization often asks a user to compare alternatives, then infers a latent utility function. Studio critique can look similar: proposals are shown, responses gathered, and the next move chosen. We can identify three corresponding structures.

- **Sequential information.** Each artifact is both a candidate and an experiment that reveals a preference or problem.
- **Active sampling.** A deliberately extreme proposal can be valuable because it clarifies a boundary, not because it is likely to be selected.
- **Model revision.** Contradictory feedback can reveal missing criteria, context dependence, or multiple audiences.

The analogy with preference learning has limits. A critic can introduce vocabulary, question premises, interpret historical references, and redirect the maker's attention. Feedback may concern the process or ethical position as well as the artifact. Its effect also depends on who gives it. The same statement from a teacher, patron, community member, or algorithm may be received differently. A computational model can summarize comparisons, but responsibility for critique and authorship remains with the participants.

We therefore retain disagreement with its context. The record identifies the evaluator, the alternative, the concern, and the confidence of the judgment. More accessible, historically false, and personally uninteresting are different kinds of observations. Combining them into one average score can conceal both their meaning and the response they require.

A protocol for creative optimization

Before introducing a generator, score, or search algorithm, write four short statements.

The preservation statement names what must survive translation: a rhythmic identity, an accessible route, a material practice, a community's authority, a performer's agency, or the ambiguity on which a work depends.

The exploration statement names what the system is allowed to vary and why variation is valuable. Without this purpose, generating more alternatives may only increase the work of selection.

The critique statement names who can reject or reframe a proposal, what evidence they will see, and how disagreement will be recorded.

The provenance statement records sources, permissions, transformations, labor, and environmental cost.

We then run the optimization with an explicit scope and stopping condition. Retain diverse alternatives, including a manually made or deliberately nonoptimal baseline that challenges the system's own vocabulary. Ask what the strongest candidate exploits, what it makes newly visible, and what it systematically omits. If the answer changes the project's purpose, return to framing rather than adjusting another weight.

When a creative system proposes alternatives, review them in four passes. The **constraint pass** checks safety, access, provenance, buildability, and other nonnegotiable conditions. The **difference pass** asks what each alternative makes possible that the baseline does not. The **meaning pass** asks how form, material, reference, audience, and context change interpretation. The **process pass** asks what accepting the proposal does to authorship, labor, skill, and future exploration.

Retain rejected candidates and reasons. A pattern of rejection can reveal a missing constraint or an inappropriate representation.

Critique may show that none of the proposed alternatives addresses the intended problem. We record this as a change to the formulation instead of assigning every candidate a low score. The next step may be to generate a different family of alternatives or to return to direct making.

Accessibility should enter before candidate generation. In architecture it changes routes, thresholds, acoustics, signage, sensory conditions, and participation. In visual art it changes description, scale, contrast, touch, sound, and display. In performance it changes instruments, pacing, seating, communication, and rehearsal.

Introducing access requirements only at the final compliance check can exclude useful creative alternatives. Co-design with disabled participants can produce forms valuable to wider audiences without claiming that all needs are identical. Preserve conflicts and provide alternatives where one solution cannot serve every mode.

Music: geometry, composition, performance, and listening

Music relates pitch, rhythm, timbre, dynamics, space, expectation, gesture, and social practice. We can use optimization to generate material, solve voice-leading problems, tune instruments, design rooms, or personalize tools. The resulting calculation addresses a particular musical task. Minimizing its rule violations does not by itself establish the value of the work.

Pitch spaces and tuning

Twelve-tone equal temperament divides an octave into twelve equal logarithmic steps. A pitch n semitones above frequency f_0 is

$$f_n = f_0\, 2^{n/12}. \tag{103}$$

This makes transposition uniform while approximating just intervals. Other temperaments optimize different compromises among interval purity, key color, modulation, and instrument construction. We can therefore examine tuning as a historically developed multiobjective design.

Tymoczko represents chords and voice leadings geometrically, revealing equivalence under octave, permutation, and transposition (Tymoczko 2006). A simple voice-leading cost between ordered pitch vectors p and q is

$$C(p,q) = \min_{\pi,k} \sum_{i=1}^{m} w_i \left| p_i - q_{\pi(i)} - 12k_i \right|, \tag{104}$$

where m is the number of voices, $w_i \geq 0$ weights the relative cost of moving voice i, π is a permutation matching voices, and each integer k_i represents an octave shift. The cost captures motion, not expressive function by itself.

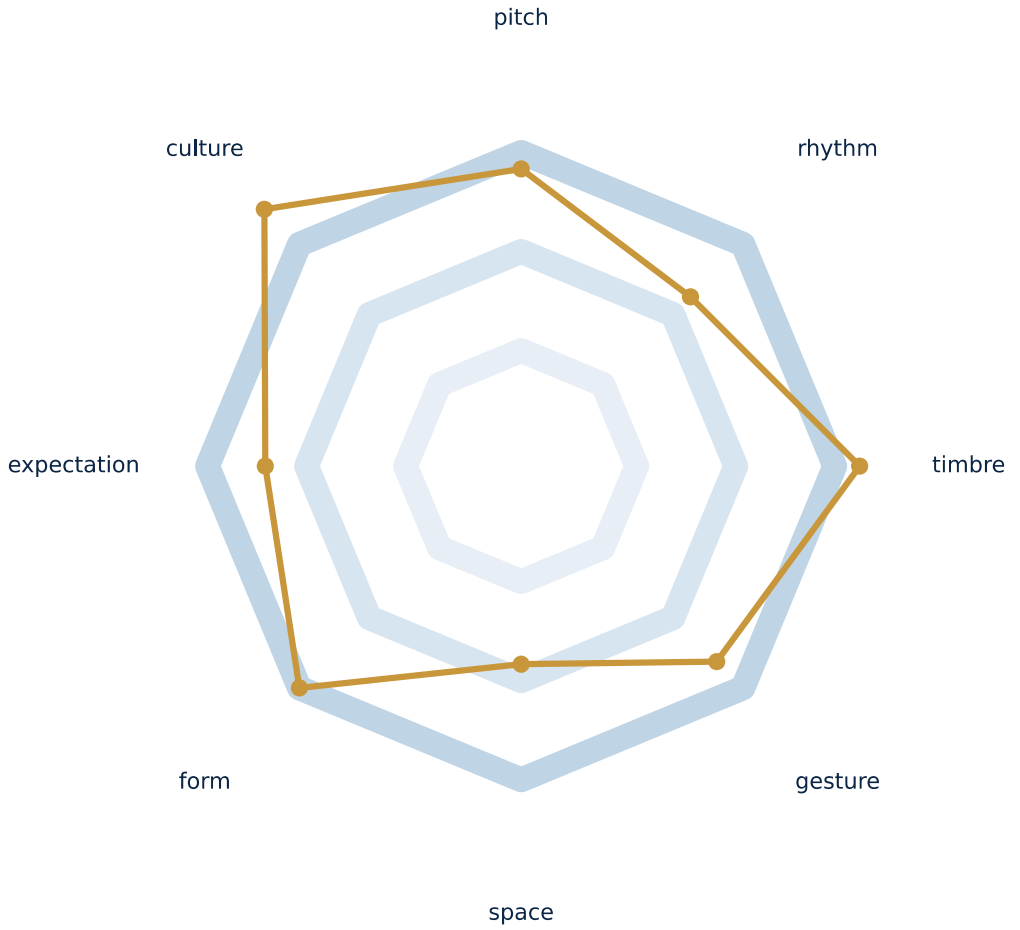


Figure 24. An eight-axis musical design radar spanning pitch, rhythm, timbre, gesture, space, form, expectation, and culture.

Composition and constraint

Counterpoint, harmonic syntax, rhythmic cycles, serialism, stochastic music, and algorithmic composition define different feasible spaces. Xenakis develops stochastic, Markov, game-theoretic, and sieve-based composition (Xenakis 1992). Constraint satisfaction can avoid parallels, respect ranges, or realize a rhythmic pattern. Dynamic programming and shortest paths can optimize voice leading, while probabilistic models generate sequences under a specified musical model.

Lerdahl and Jackendoff model aspects of tonal cognition through grouping, meter, time-span, and prolongational structures (Lerdahl and Jackendoff 1983). Such theories explain expectations for particular listening traditions, not universal musical law.

Music: local costs and global reasons

Voice-leading cost, tuning error, fingering difficulty, rhythmic alignment, spectral masking, and room response admit meaningful formal models. They are local descriptions of musical problems. We need to check whether improving the local cost supports the purpose of the complete work.

Suppose a harmonization system minimizes voice movement, forbidden parallels, range violations, and tonal surprise. It may produce an idiomatic passage that contributes little to the intended dramatic effect. Increasing surprise may change individual events without improving the form. We therefore evaluate event-to-event movement, phrases, sections, and the complete work. A useful rule at one scale may be inappropriate at another. For example, repetition can be redundant locally and necessary to the form.

Performance takes place through a performer and an acoustic environment. Interpreting a symbolic score involves communication through timing, touch, breath, balance, and risk. A technically optimal fingering may not support the desired articulation. A click-aligned ensemble may lose collective elasticity. A loudness-normalized master may satisfy a platform specification while losing transient shape. Each translation from score to gesture, gesture to sound, sound to recording, and recording to listener preserves some relations and changes others.

Figure 24 presents eight interacting musical attributes. The radar does not depict mappings between spaces or establish that acoustics determines meaning. Pitch geometry preserves interval relations under chosen equivalences; it need not preserve historical function, embodied difficulty, or emotional association.

Performance

A score underdetermines timing, articulation, balance, phrasing, and physical gesture. Performers communicate intention through these choices within the limits of instrument, room, ensemble, fatigue, and rehearsal time. Expressive microtiming deliberately departs from a strict grid; minimizing timing deviation can remove the intended phrase.

Practice itself is a resource-allocation problem. Slow practice, chunking, interleaving, mental rehearsal, recording, and performance simulation target different weaknesses. The shortest route to a technically correct passage may not build resilient recall on stage.

Performance and rehearsal as adaptive control

Rehearsal is a closed loop with delayed and noisy observation. The performer chooses a practice action, observes errors and sensations, updates an internal model, and selects the next task. Chapter 17's experimental discipline applies: isolate a difficulty when possible, vary conditions to test robustness, and distinguish a temporary performance gain from learning that persists.

The Banister fitness-fatigue model in Chapter 16 is a useful caution. Load can create both adaptation and fatigue on different time constants. The same is true of cognitive and motor practice. More repetitions can improve the current run while degrading attention or increasing injury risk. A robust rehearsal plan alternates consolidation, contextual variation, rest, and simulated performance. It also defines a stopping rule: continuing after reliable success may yield less than switching to another weakness.

Ensemble rehearsal also requires coordination. A player concentrating on local accuracy may become less responsive to others. We can therefore consider the ability to respond as a property to retain across performances. Conductors and directors allocate limited attention when deciding which errors to interrupt and discuss. Stopping after every local error may prevent the ensemble from developing an understanding of the whole work.

Acoustics and production

Room design balances reverberation, clarity, envelopment, intimacy, bass response, isolation, and variability across seats. Sound reinforcement and mixing manage headroom, masking, loudness, localization, and translation across playback systems. An optimizer can flatten a frequency response while degrading phase, transient behavior, or musical intent.

Listening and recommendation

Recommendation systems optimize engagement and can narrow exposure or reward homogeneity. Diversity, creator sustainability, user agency, and serendipity deserve explicit controls.

Field checklist

- Separate hard obligations, trade-offs, and interpretation; use alternatives for critique, not outsourced authorship.
- Evaluate across spatial, temporal, and social scales; record provenance, consent, labor, and environmental burden.
- Preserve disagreement across value types; test buildability, performance, maintenance, and accessibility.
- Name who may reframe or refuse the optimization.

Translation lens. Creative practice transfers constrained exploration, preference learning, feedback, and revision across representations. People can also revise the criteria and the framing itself. They retain authorship and responsibility for the work.

Chapter 16 · Literature, narrative, media, games, sport, and competition

Narrative systems, language models, and media

Literary works use different criteria. A work may seek clarity, ambiguity, suspense, density, voice, estrangement, persuasion, memory, or play, and its aims may change during writing. Formal models can help us analyze or generate structure. Their results still require interpretation through reading.

Plot and narrative structure

Narratives arrange events through order, duration, frequency, perspective, and voice (Genette 1980). A story is not identical to its discourse: the same events can be told in different sequences and from different positions. Propp's morphology identifies recurring functions in a corpus of folktales (Propp 1968); it is a descriptive grammar, not a recipe for all stories.

A planning model may define states s, actions a, transitions $P(s' \mid s, a)$, and narrative rewards such as causal coherence or character goal progress. The difficulty is that an action's value depends on interpretation and later revelation. Surprise requires balancing improbability with retrospective intelligibility.

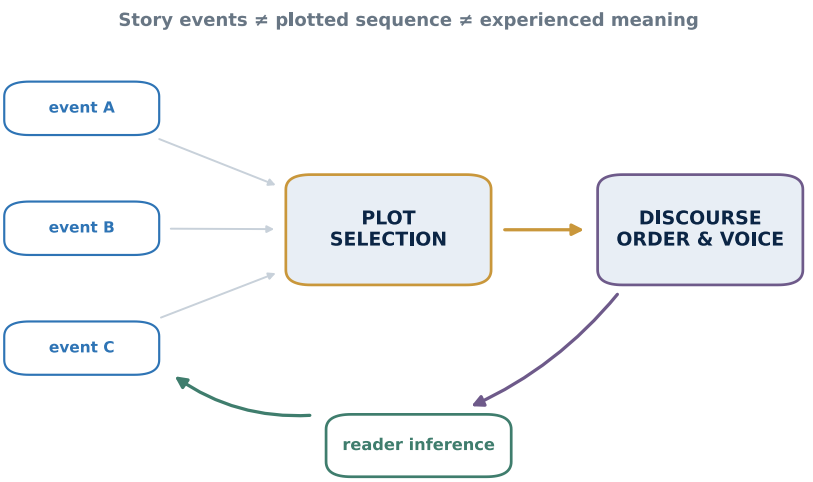


Figure 25. A narrative search diagram connecting world events, plot selection, discourse order, voice, reader inference, and revision.

Style and language models

Stylometry measures word frequencies, syntax, rhythm, and semantic patterns. Compression and probabilistic models estimate regularity. These descriptions can assist stylistic analysis without determining what a particular work should say.

Editing involves several related criteria. Shortening may improve pace while removing necessary detail. Simplifying may improve access while changing the author's vocabulary or sentence structure. We can use a partly lexicographic procedure: preserve meaning and ethical commitments, then revise structure, clarity, rhythm, and correctness within those limits.

Media systems

Headlines, feeds, search engines, advertising, and streaming platforms optimize attention and conversion. These metrics are behavioral responses shaped by the platform. Click-through can reward outrage, deception, or repetition. Long-term trust, informed agency, diversity, creator welfare, and civic effect require slower measurements and governance.

Communication campaigns optimize reach, comprehension, recall, and action across audiences. Testing variants can improve wording, but effects are contextual and can decay. Persuasion becomes manipulation when it exploits vulnerability, hides intent, or removes meaningful choice.

Interactive narrative

Games and hypertext create branching paths. Aarseth's ergodic literature requires nontrivial reader action (Aarseth 1997). Branching multiplies content, while procedural systems recombine rules. Optimization can allocate authoring effort to high-impact nodes, but rare paths may carry disproportionate artistic or ethical meaning.

We can compare narrative and play through the relation between sequence and participation. A finished novel offers a controlled sequence whose effects depend on interpretation; a competitive game offers explicit rules whose outcomes depend on action. Between them lie interactive fiction, recommendation feeds, live performance, sport, role-playing, and procedural media. Each can use optimization, although a score describes only part of the activity.

The common structure is temporal. Decisions alter what can happen next, and their value depends on later consequences. Chapter 7's Bellman recursion therefore transfers naturally, but only after the state is defined. In chess the state is largely specified by the board and rules. In a novel, "state" includes what the reader believes, remembers, suspects, and values—quantities that cannot be observed directly and need not be shared across readers. In a live sport, fatigue, injury, morale, weather, officiating, and opponent adaptation make the state partially observed. We therefore need to redefine the state and its meaning when transferring the mathematical structure.

Narrative and the order of information

Discourse controls when information becomes available. Scheduling clues under consistency constraints represents part of a suspense problem, but suspense is not a scalar quantity: it depends on what a reader considers possible, fears, desires, and later finds retrospectively fair.

A planning representation can help an author inspect gaps in causal progress or character goals. It becomes prescriptive when corpus frequency is mistaken for narrative necessity.

We can begin an editorial revision by identifying factual commitments, ethical limits, and ambiguities essential to the work. We then examine causal clarity, pacing, continuity, rhythm, and sentence-level precision. A final reading of the complete text checks interactions between local changes. For example, removing a repeated image may weaken a later motif, and changing a late scene may alter the meaning of an early one. The process resembles coordinate descent, but the variables remain coupled: a local revision changes the context in which other passages are read.

Figure 25 distinguishes world events, plot selection, discourse order, voice, reader inference, and revision. The figure also helps identify what a compressed representation omits. A plot graph preserves causal or temporal relations selected by the model. It does not preserve prose texture, unreliable narration, cultural reference, or the experience of rereading.

Language models and decoding objectives

A generative language model assigns conditional probabilities to token sequences. Decoding then turns a distribution into a particular text. Greedy selection, beam search, temperature, nucleus sampling, repetition penalties, and reranking embody different trade-offs among likelihood, variation, consistency, and computational effort. More exhaustive likelihood search can produce dull, repetitive, or even empty output because model score, adequacy, and open-ended literary quality are not monotonically related (Holtzman et al. 2020; Stahlberg and Byrne 2019).

We evaluate a writing system at several levels. Token likelihood measures statistical fit, while factual claims require sources and checks of what those sources support. Coherence concerns entities, causal relations, themes, and discourse. Style includes word choice, sentence structure, rhythm, voice, register, and cliché. Ethical review concerns representation, privacy, attribution, and foreseeable use. We also examine whether the writer can understand, revise, and reject the contribution. One automatic metric cannot validate all these properties.

The use of a writing tool can also change the work of authors. If a platform rewards speed and volume, it may encourage repeated templates even when the tool was introduced to assist writing. We record human and machine transformations so that they can be inspected. Authorship cannot be inferred from the percentage of retained machine text, since selection, rejection, relocation, rewriting, and context also affect a sentence’s role.

Media platforms as endogenous optimization

Media measures are part of an intervention, not passive observations. A recommendation changes which items people encounter. This affects subsequent learning, creator incentives, and preferences, as well as the data available for training. Click-through, watch time, completion, subscription, and retention therefore depend partly on the platform’s earlier actions.

The Goodhart mechanism introduced in Chapter 3 and distinguished from selection bias in Chapter 17 appears in at least three forms. Creators adapt thumbnails and pacing to the metric. Users learn habits shaped by the interface. The platform's training data increasingly reflects its own past policy. A model can therefore become accurate about a world it helped narrow. Randomized exploration provides causal information, but experimentation on attention needs limits: vulnerable users, political content, grief, health information, and compulsive patterns cannot be treated as ordinary engagement opportunities.

A governed media objective should include long-horizon and distributional measures: user control, diversity of exposure, creator concentration, satisfaction after use, trust, misinformation burden, and the ability to leave. Some should be constraints or audit triggers rather than weighted rewards. The counterfactual baseline matters. "More time on platform" can mean displaced sleep, work, care, or offline community.

Recommendation evidence

A media or cultural recommender observes outcomes only for items it exposed. Naively comparing clicked and unclicked items confounds relevance with policy. Randomized or structured exploration can estimate counterfactual response, but exploration has ethical and experiential cost. Use conservative domains, user control, and clear exclusions.

Evaluate beyond immediate behavior. Follow satisfaction after use, diversity across a session and over time, ability to find a known item, novelty judged useful, creator concentration, and whether users can reset or escape the profile. Compare with chronological, curated, search-based, and user-configured baselines.

An engagement improvement can reflect compulsion or a difficult exit. Pair behavioral measures with survey, interview, complaint, and well-being evidence. Preserve a nonpersonalized path. The recommendation should remain open to the person's own judgment.

Evidence for writing assistance

An automatic story metric may measure textual overlap, predictability, or a classifier's judgment. Search can then favor repeated formulas, keywords, or apparent coherence that improves that metric. We also use human reading, source checks, inspection of causal relationships, and adversarial examples. Passages on which reviewers disagree should remain available for interpretation.

A benchmark corpus represents genres, languages, and historical conventions unevenly. Report coverage and avoid universal claims. Generated narrative should be evaluated for attribution, stereotype, privacy, and the effects of deployment, not only text quality.

For writing assistance, measure whether the user can maintain intention, identify unsupported content, revise efficiently, and understand provenance. Fluency alone does not show that the continuation supports the writer's authorship.

Games, sport, rules, and training

Games define goals within rule systems. Sport adds physical performance, institutions, spectators, and culture. These explicit goals make both useful for studying optimization. We can also observe cases in which improving a score undermines another purpose of the activity.

Strategy and equilibrium

In a finite two-player zero-sum game with payoff matrix A, the row player chooses mixed strategy p and the column player q. The minimax theorem gives

$$\max_{p}\min_{q} p^T A q = \min_{q}\max_{p} p^T A q. \tag{105}$$

This supports equilibrium strategies in games such as poker subproblems. General games may have multiple equilibria and require beliefs about opponents. Exploitative play can outperform equilibrium against a known weakness while becoming vulnerable to counteradaptation.

Search methods evaluate game trees with minimax, alpha-beta pruning, Monte Carlo tree search, learned value functions, and policy improvement. AlphaGo combined deep networks with tree search and reinforcement learning (Silver et al. 2016). The achievement depended on a clear rule set and simulator—conditions absent from many real institutions.

A strategy can change the opponent population by provoking counters. The value landscape therefore changes with policy use. Equilibrium analysis describes strategic consistency; it does not replace evaluation of the rules themselves.

Game design

Designers tune challenge, pacing, economy, progression, information, matchmaking, social interaction, and accessibility. Character balance concerns viable counterplay and meaningful choice; identical statistics are not required. Live-service telemetry reveals behavior, yet optimizing retention or spending can produce compulsion and distrust.

Procedural content generation searches levels, maps, quests, or items under feasibility and novelty constraints. Human playtesting detects legibility, emotion, and emergent strategies that static metrics miss. Rules should be evaluated for dominant strategies, exploits, accessibility, and community norms.

Balance can mean viable strategic diversity, counterplay, accessibility, and recoverability. Equal aggregate win rates can conceal a frustrating strategy or dominance confined to expert levels.

Telemetry should be stratified. Aggregate success rates can hide platform, region, disability, skill, or party-composition effects. Patch evaluation should distinguish immediate novelty from stable adaptation. A balance change is a sequential experiment with interference:

players share information and change opponents' experiences. Confidence intervals derived as if matches were independent can be falsely narrow.

Monetization creates a second objective system. Retention and spending can conflict with player welfare, parental expectations, and the integrity of progression. The governance process therefore needs to address manipulative design patterns. Constraints should cover disclosure, age suitability, purchase friction, self-exclusion, and the separation of competitive advantage from spending where appropriate.

Access and difficulty

We can separate difficulty into motor speed, precision, perception, memory, planning, language, ambiguity, time pressure, and tolerance for failure. Reducing enemy strength in an easy mode may leave a player's actual access barrier unchanged. Adjustable dimensions allow us to address that barrier while preserving the intended experience where possible.

Accessibility options can be part of the rule system: remapping, timing windows, visual and auditory alternatives, automation, pause, information, and cooperative support. Evaluate with players whose access needs differ. We can examine competitive integrity through the design of categories, disclosure, and meaningful comparisons that accommodate access needs.

Optimization can tune adaptive difficulty, but hidden adaptation may undermine trust or erase accomplishment. Give players control or clear communication. Session length should be interpreted in relation to sustained meaningful play.

Sports analytics

Sport optimization covers tactics, roster construction, scheduling, training, nutrition, equipment, and injury prevention. Expected-points or win-probability models translate situations into decision support. Their validity depends on opponent, era, officiating, selection, and strategic reaction.

Prediction and prescription differ. Success under current selection practices does not establish that those practices are fair or causally effective. Coaches, injuries, contracts, and earlier models select the observed data.

A tactical action depends on the surrounding state and the opponent's response. For example, a shot's value depends on the defender, clock, score, fatigue, lineup, and what the opponent learns from the attempt. Repeatedly choosing the same action changes later opportunities. We therefore compare immediate gains with the value of variation over a sequence of contests.

Training balances stimulus and recovery. The Banister fitness-fatigue model represents performance as the difference between slower positive adaptation and faster negative fatigue responses (Banister et al. 1975):

$$R(t) = R_0 + k_1 \sum_{i=1}^{t-1} u_i \, e^{-(t-i)/\tau_1} - k_2 \sum_{i=1}^{t-1} u_i \, e^{-(t-i)/\tau_2}. \tag{106}$$

$R(t)$ is modeled readiness at time t and R_0 is baseline readiness; u_i is the training impulse at time i; k_1 and k_2 are positive adaptation and fatigue gains; and τ_1 and τ_2 are their decay time constants. The described slower adaptation and faster, initially larger fatigue response require $\tau_1 > \tau_2$ and $k_2 > k_1$. These parameters are individual and uncertain. Maximizing short-term load can increase injury and burnout. Athlete consent, education, and data privacy constrain monitoring.

The readiness model is conceptual and supplies no universal training prescription. Individual parameters vary, observations are noisy, and injury mechanisms exceed one performance state. Medical professionals and athletes determine whether a proposed load is acceptable.

Athlete data raises special governance issues. Location, sleep, menstrual cycles, biometrics, mental health, and injury history can improve planning and enable surveillance or coercion. Consent inside an employment or scholarship relationship may not be freely given. Collect only data tied to a defined decision, limit access and retention, provide correction and appeal, and prohibit secondary uses that were not agreed. An athlete must be able to challenge both the measurement and the inference.

Monitoring should support discussion of the proposed action. Distinguish population evidence from an individual pattern and preserve the uncertainty. Pain, illness, psychological state, or a sudden change can invalidate a forecast.

Use a safe workload envelope, planned recovery, and review triggers. Separate monitoring from unrelated selection or contract decisions. Performance optimization remains subject to bodily autonomy.

Integrity and spectacle

Anti-doping, equipment rules, salary caps, draft systems, and officiating seek a credible contest. Rule changes optimize a bundle: safety, fairness, pace, tradition, broadcast appeal, and participation. Participants adapt, so every rule is a mechanism-design experiment.

Rules distribute opportunity and shape style. Mechanism design asks how they transform private incentives into collective outcomes; legitimacy asks who had standing to choose those outcomes.

Every rule change should name its theory of harm or benefit, anticipated strategic response, distributional effects, evidence window, and reversal condition. A faster game can be less safe. A more “objective” judging system can reward conformity to measurable technique. A cost cap can create enforcement complexity that advantages organizations with legal and engineering resources. Optimization of the competition requires an audit of the enforcement system.

We need to retain the purposes that define the activity. In sport, these may include excellence, uncertainty of outcome, bodily expression, community, tradition, and credible comparison of performances. In games, they may include mastery, discovery, social

interaction, fiction, or playful disorder. If improving a metric undermines an intended purpose, we reconsider the metric or stop using it.

When a league or game changes rules, the treatment reaches all participants and they adapt together. Evaluation should use interrupted time series, comparable competitions, simulation, and qualitative observation, while acknowledging interference. Separate transient novelty from stable strategy.

Define desired mechanisms before outcomes: reduce dangerous contact, increase viable strategies, shorten dead time, or improve access. Then measure unintended response, enforcement burden, distribution by role or level, and whether participants find the contest more legitimate. A rule can meet a numerical target and undermine play.

A translation protocol for narrative and play

First identify the level: artifact, participant strategy, platform policy, or rule system. Objectives at one level can be constraints at another. Second identify the state variables the model cannot observe: interpretation, trust, tacit skill, injury, informal norms, or future cultural value. Third identify adaptation: readers learn genres, players learn exploits, athletes change tactics, creators change output. Fourth compare short- and long-horizon effects. Fifth preserve a route for authors, participants, and affected communities to contest the model.

Field checklist

- Distinguish events, discourse, and reader inference; evaluate generated language beyond likelihood.
- Treat recommendation as intervention; separate player strategy, game design, platform policy, and institutional rules.
- Test tactics and patches under adaptation; protect athlete consent, privacy, health, and appeal rights.
- Audit whether the score serves the activity’s constitutive purpose.

Translation lens. Narrative and competition share dynamics, search, equilibrium, and feedback structures. A transfer can preserve temporal relationships while changing meaning, legitimacy, or purpose. We therefore examine those changes explicitly.

Part VI · Practice, Translation, and Frontiers

Chapter 17 · Experimentation, digital twins, human judgment, and failure modes

Evidence-producing optimization

We can distinguish several sources of evidence for an optimization decision. Experiments estimate causal response, simulations explore models, digital twins maintain selected model states in relation to an operating asset, and human-in-the-loop methods elicit judgments that are difficult to specify in advance. Each source supports a different kind of claim.

Design of experiments

One-factor-at-a-time testing cannot reveal interactions efficiently. Factorial and fractional-factorial designs vary inputs together; response-surface methods fit local curvature; blocking controls nuisance variation; randomization protects against unmeasured trends. The response-surface model is

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \sum_{i=1}^{p} \beta_{ii} x_i^2 + \sum_{i=1}^{p-1} \sum_{j=i+1}^{p} \beta_{ij} x_i x_j + e. \tag{107}$$

In Equation (107), the pure quadratic terms represent local curvature, the interaction terms show why the best setting of one factor can depend on another, and e is regression noise (National Institute of Standards and Technology 2013). Replication estimates noise. Sequential designs use early results to choose later experiments for information or improvement.

Online A/B tests estimate treatment effects in live systems. Guardrail metrics protect reliability, safety, and vulnerable groups. Network spillovers, novelty, interference, multiple testing, peeking, and long-term adaptation can invalidate simple comparisons. Stopping rules and analysis plans should be specified before results are inspected.

Simulation optimization

When the objective is an expectation available only through simulation,

$$\min_x J(x), \qquad J(x) = \mathbb{E}[g(x, \xi)]. \tag{108}$$

Here $g(x, \xi)$ is the realized simulation cost. Each evaluation is noisy. Common random numbers can reduce comparison variance; ranking-and-selection allocates samples among alternatives. Surrogates and Bayesian optimization can reduce expensive runs (Jones, Schonlau, and Welch 1998; Shahriari et al. 2016). Stochastic approximation supplies another search approach. Validation must compare simulated and observed behavior at the level needed for the decision.

Digital twins

A digital twin maintains a relationship among an asset, data, models, and decisions. Its state estimate should record uncertainty and data freshness. Calibration can reproduce one output through compensating parameter errors, so we also need to examine identifiability and performance under multiple conditions.

A twin can support condition monitoring, what-if analysis, predictive control, maintenance, and training. Its governance includes model version, sensor provenance, cybersecurity, operator override, and retirement. The required fidelity depends on the decisions the twin supports.

Human-in-the-loop search

Interactive optimization asks people to compare alternatives, set aspiration levels, adjust constraints, or critique proposals. Pairwise preference data can fit a latent utility model, but preferences may be unstable and constructed during interaction. The interface should show trade-offs and uncertainty without forcing false precision.

Figure 26. Seven-stage evidence loop—question, experiment, simulate, calibrate, decide, operate, and learn—around a central EVIDENCE LOOP.

Figure 26 shows how several sources of evidence enter the process. Experiments support causal comparisons, simulations support conditional claims about models, and digital twins relate model states to a system in operation. Operation provides observations of consequences. Judgment and revision determine how these observations change the next decision.

Participatory modeling broadens whose knowledge enters. Domain experts reveal tacit constraints; affected communities reveal harms and meanings; operators reveal workarounds.

The record should retain disagreements when averaging would conceal different constraints or interpretations.

Practice note. Close every optimization loop with a measurement plan, a model-update rule, an accountable decision maker, and a condition for stopping or rollback.

Operational telemetry describes a system already affected by earlier decisions. When combining it with experiments, models, and judgments, retain the source and scope of every claim.

For each reported result, we therefore record its provenance and the scope of its claim. If alternative A has a lower expected cost, we specify the model, calibration period, uncertainties, exclusions, and decision horizon. The decision dossier in Chapter 3 provides a place for this information.

Experiments for improvement and information

When experiments are expensive, we may choose the next trial to improve a candidate or to reduce uncertainty about the response. These aims can suggest different trials. Sampling near a promising candidate may improve the result, while sampling elsewhere may distinguish models or reveal an interaction. We specify which information could change the decision before trying to reduce uncertainty across the whole domain.

Blocking, randomization, replication, and predeclared stopping help prevent production drift, operator learning, calendar patterns, and selective continuation from being mistaken for treatment effects.

Online experiments add interference and adaptation. A pricing change affects competitors; a feed treatment affects what creators produce; a clinical workflow affects staff behavior across arms. We therefore need evidence for the assumption that one unit's outcome is unaffected by another unit's treatment. If spillovers are material, randomize at a cluster or system level, model the network, or treat the experiment as a policy intervention rather than a collection of independent observations.

Guardrails need explicit decision rules. A primary metric can improve while a rare safety outcome deteriorates. Because rare harms produce wide intervals, "not statistically significant" is not evidence of safety. Use prior evidence, severity-weighted thresholds, sequential monitoring, and conservative deployment stages. Define who may stop the experiment and how quickly the system can be restored.

Simulation and extrapolation

Simulation permits controlled comparisons within a model. Search can also select regions where the model has received little validation or where its approximation error favors a candidate. We need to examine optimized candidates as well as the ordinary operating conditions used to calibrate the simulator.

We validate quantities that can affect the decision, including rankings, sensitivities, constraint margins, failure modes, and the responses of selected candidates. Matching an average output is insufficient for these purposes. For example, a simulator may reproduce average throughput while ranking schedules incorrectly. A building model may reproduce annual energy while misrepresenting peak load. A physiological model may reproduce ordinary meal responses while failing under exercise.

Use several layers of challenge:

- **verification:** does the implementation solve the stated equations or rules;
- **calibration:** do parameters reproduce selected observations;
- **validation:** does the model support the intended class of claims;
- **stress testing:** does it behave plausibly outside calibration conditions;
- **red-teaming:** can the optimizer exploit numerical, physical, or semantic gaps;
- **reconciliation:** when model and world differ, is the difference explained and recorded?

A simple policy evaluated in the same simulator and in the field provides a useful baseline. Efficient simulation search still needs an independently justified evaluation boundary.

Digital twins as governed relationships

A twin suitable for thermal control may provide insufficient evidence for fatigue prognosis. Associate each fidelity claim with the decision it supports.

An operational twin should display the current state of its evidence. This includes data timestamps, missing observations, sensor health, model version, parameter uncertainty, prediction intervals, and distance from validated conditions. A status indicator that omits stale data may imply more confidence than the available observations support.

Recalibration may change alarms, maintenance schedules, or controller actions. We therefore treat a model update as a configuration change that requires testing, approval, rollback, and versioned evidence. Corrupted sensor data or an unauthorized update may also invalidate the represented state. The assurance case in Chapter 9 and interface specifications in Chapter 10 help us trace these dependencies.

Human judgment and authority

Operators, patients, engineers, editors, communities, and decision boards know things the model does not. They also have biases, incentives, limited attention, and uneven authority. The effectiveness of human review depends on its operating conditions. The human needs time, information, competence, an actionable alternative, and protection from retaliation. Overrides must be observable and learnable without becoming a performance target that discourages justified dissent.

We assign work according to what each participant can contribute. Machines can check consistency, search many combinations, and identify anomalies. People can reinterpret the purpose, recognize a new context, discuss values, and accept responsibility. Some decisions

require two independent judgments or an appeal body outside the operating process. The interface needs to present uncertainty and causal assumptions as well as the recommendation.

Preference elicitation deserves special caution. A person asked to rank complex alternatives may construct a preference during the process. Pairwise comparisons can reduce cognitive load, but the inferred utility is conditional on framing, information, and fatigue. Preserve the raw comparison, allow “incomparable” and “need more information,” and show how sensitive the recommendation is to the fitted preference model.

Stop, rollback, and learn

Stop search when further computation is unlikely to change the decision or measurement uncertainty dominates solver error. Stop or roll back deployment when specified safety, equity, reliability, or trust limits are crossed. Stop the project when its representation cannot support an authorized and justified decision. Chapter 20 develops these distinct rules.

Rollback requires preparation. Preserve the previous policy, compatible data, operator knowledge, spare capacity, and authority to restore service. A recommendation that cannot be reversed needs stronger evidence and broader consent. Irreversibility may arise from physical construction, ecological thresholds, institutional lock-in, or loss of skill even when the software itself is reversible.

Post-deployment review should compare predicted and observed outcomes, including distribution and mechanism. Record overrides, near misses, complaints, workaround behavior, and unanticipated externalities. A failed prediction can improve a model; an unexplained organizational workaround may reveal that the modeled constraint was fictitious or that an omitted burden made the recommendation unusable.

The minimum evidence packet

For each optimized intervention, preserve:

- the decision and rejected baselines;
- model, data, code, solver, versions, and random seeds where relevant;
- objective and constraint definitions with owners;
- calibration, validation, sensitivity, and stress-test results;
- evidence gaps and extrapolation flags;
- affected groups, approval, appeal, and override routes;
- monitoring metrics, thresholds, response owners, and review dates;
- rollback assets and the condition for retirement.

This evidence packet allows another person to reconstruct the claim, examine its assumptions, and decide whether it still applies. Its completeness alone does not establish that the claim is correct.

Failure rehearsal and model risk

Before deployment, ask the team to imagine that the optimization caused a serious failure one year later. Each participant writes a mechanism independently: wrong boundary, proxy gaming, corrupted sensor, strategic response, inaccessible interface, common-cause failure, silent update, lost skill, or unavailable fallback. Group the mechanisms by detection and recovery, then add tests and owners.

Run a rollback drill in the actual environment. Can the prior model, policy, data schema, credentials, configuration, and operating instructions be restored? Is old hardware or manual capacity still available? Does rollback preserve records created under the new system? How long does it take, and who can authorize it outside business hours?

Stopping the software may not restore the earlier service. Other systems may now depend on its output, staff may have lost familiarity with the old process, or a data migration may be irreversible. We therefore test recovery of the complete service as well as the off switch.

List material model assumptions and assign each an evidence strength, consequence if wrong, detectability, and mitigation. We use this record to direct further examination toward assumptions with substantial consequences and weak supporting evidence.

Use scenario diversity and independent evidence where common assumptions could fail. Two simulations sharing the same physical law, data, or code are not independent. Use alternate formulations, analytical bounds, experiments, and domain judgment. Track unresolved assumptions through deployment.

When the optimizer selects a region near a weak assumption, raise the risk class and require confirmation. The assessment of model risk then changes with the region selected by the search.

Failure modes, review, and purpose

An optimization failure can occur even when the solver correctly solves the stated problem. The formulation may contain an inappropriate boundary, proxy, data model, dynamic assumption, incentive, or assignment of authority. The following cases help distinguish these defects.

Proxy gaming and Goodhart effects

Let $M(x) = V(x) + \varepsilon(x)$ be a noisy proxy for true value. For a simple strict-bias example, take at least two candidates with equal true value and independent, identically distributed, nondegenerate zero-mean errors with finite expectation. Selecting the largest measured value then selects favorable error:

$$\mathbb{E}[\varepsilon(\hat{x})] > 0, \qquad \hat{x} \in \arg\max_{x \in \mathcal{X}} M(x). \tag{109}$$

This example isolates statistical selection bias. With unequal true values, the magnitude and strictness of the bias depend on the joint value-and-error model. Smith and Winkler analyze the optimizer's curse and its implications for decision analysis (Smith and Winkler 2006). Goodhart adaptation instead changes behavior and the measurement process after a metric becomes a target (Goodhart 1975; Muller 2018).

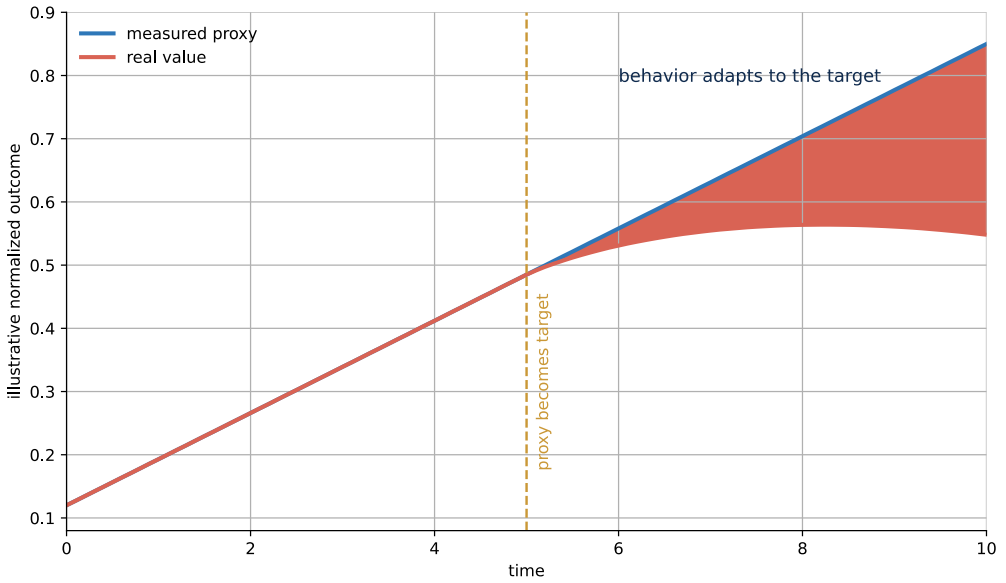


Figure 27. Illustrative divergence of measured proxy and real value after a proxy becomes a target and participants adapt to it.

Figure 27 illustrates the adaptive form, not the optimizer's curse. The separation matters: a rotating metric may frustrate gaming but does not remove selection bias, while shrinkage or out-of-sample evaluation may reduce selection bias without changing strategic response.

Countermeasures include multiple measures, qualitative review, random audits, causal evaluation, rotating indicators, uncertainty penalties, and explicit anti-gaming tests. We also need to review whether the goal remains appropriate.

Boundary errors

A boundary error occurs when the model omits a relevant consequence. For example, a factory may reduce on-site emissions by outsourcing a polluting process. A hospital may shorten stays while increasing family care work. A model may reduce false negatives while producing more alerts than staff can handle. We examine the lifecycle, supply chain, feedback, affected groups, and externalities before selecting metrics.

Model exploitation

An optimizer searches farther than ordinary use and finds model weaknesses: nonphysical geometry, adversarial inputs, unsafe controller actions, unrealistic financial leverage, or policy loopholes. Validation data sampled near past behavior may not cover the optimized region.

Trust regions, feasibility restoration, adversarial testing, and staged deployment limit extrapolation.

Brittleness and hidden coupling

Tight optimization can consume slack and remove capacity needed during disruption. High utilization can lengthen queues, just-in-time inventory can increase exposure to supply failures, and weight reduction can reduce damage tolerance. Specialization may limit adaptation. Robustness requires scenario diversity and attention to common causes, since correlated failures can defeat nominal redundancy.

Technical systems also carry organizational coupling. A dashboard metric can reshape budgets; an automated rule can dissolve expertise; a recommendation can become mandatory through workload. Hidden technical debt in machine-learning systems often resides in data dependencies, feedback, entanglement, and undeclared consumers (Sculley et al. 2015).

Optimization theater

A precise model can legitimate a decision already made, conceal value choices, or overwhelm challenge. Warning signs include undocumented weights, no baseline, no sensitivity analysis, selective scenarios, unexplained exclusions, and no accountable owner. A small transparent model may support better judgment than a large unchallengeable one.

Success that erases purpose

A procedure can improve its metric while contributing less to the institution’s purpose. For example, higher test scores may accompany a narrower education, and more citations may accompany less useful scholarship. Engagement may fail to represent artistic value, and throughput may omit important aspects of care. Periodic review asks whether the activity would still be valued if the metric were removed.

Practice note. Use extreme solutions as diagnostic examples. Check what the formulation permits, then examine the candidate through independent domain review, stress tests, and observations from operation.

Review and retirement

At a decision meeting, assign one reviewer to argue from the baseline, one to challenge the boundary, one to inspect uncertainty, one to represent implementation and recovery, and one to trace affected-party consequences. Give them access to model and data assumptions before the presentation.

Ask the project team to search for a candidate that performs well under the objective but fails domain review. If such a candidate is found, record the missing or incorrect assumption. If none is found, record the scope of the search; this result alone does not establish that the formulation captures every relevant requirement.

The meeting record should distinguish empirical, inferred, monetary, preferential, ethical, aesthetic, and procedural judgments, together with unresolved disagreements. A frontier chart should retain these distinctions in its interpretation.

When a project stops, preserve the reason. Categories include no legitimate authority, insufficient measurement, unacceptable harm, infeasible delivery, no safe fallback, or no improvement over baseline. Record what narrower use remains acceptable and what evidence could reopen the question.

Recording the reason for stopping helps another team avoid repeating unsupported assumptions. It can also direct work toward measurement, service capacity, process repair, or a decision aid with a narrower scope. A documented decision to stop is therefore a useful project outcome.

Operational dashboards usually emphasize frequent observations. Changes in care quality, learning, trust, ecological function, artistic activity, or institutional capability may be slower and harder to measure. We include qualitative and long-term evidence in the review of the system's purpose.

We ask whether the activity remains valuable without the metric and look for workarounds, lost capabilities, or burdens transferred elsewhere. Affected parties and independent experts contribute to this review.

Purpose monitoring can lead to reframing or retirement while service metrics remain within their limits. The review concerns whether the activity still serves its purpose, which may require evidence beyond those metrics.

Field checklist

- Match evidence to claims; predeclare decision-relevant experiments and stopping rules.
- Validate simulations on rankings, sensitivities, margins, and candidates; version twins with uncertainty and rollback.
- Give reviewers time, authority, alternatives, and appeal; distinguish optimizer's curse, Goodhart adaptation, model exploitation, and boundary error.
- Specify when to stop search or deployment, or refuse the project.

Translation lens. Experimentation and digital twins use feedback, state estimation, and sequential design. When transferring their results, preserve the source and scope of the evidence. Participation, decision authority, simulation results, and observations from operation remain distinct.

Chapter 18 · Worked translations I: infrastructure, health, and adaptive control

How to read the worked translations

The seven cases in Chapters 18 and 19 examine transfers between disciplines. Each begins with a domain, its vocabulary, evidence, and decision authority. We then identify a structure imported from another discipline, the relations it preserves, and the meanings that require separate interpretation. The cases show how these choices affect a proposed decision.

We use a repeated sequence to make the cases comparable. Each begins with the decision, purpose, boundary, affected parties, variables, and alternatives. We then describe objectives, uncertainty, methods, evidence, and responsibility for failures. A transfer ledger records the relationships borrowed from other disciplines and their limitations. Each case also includes a more detailed domain discussion and a seven-point checklist. Most cases add a section on domain-specific structure between the objectives and the method.

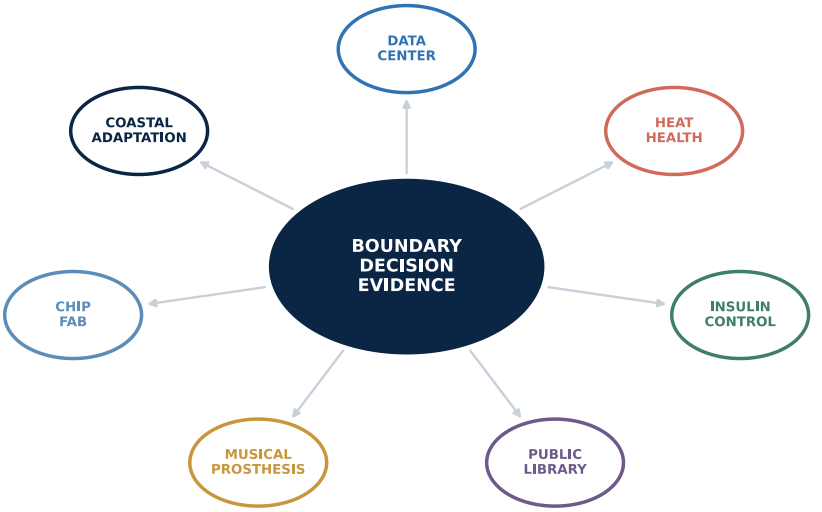


Figure 28. Seven worked translations around shared boundary, decision, and evidence.

Figure 28 groups the seven cases around shared questions of boundary, decision, and evidence. Purpose guides the boundary and alternatives; governance assigns decision authority and response. The transfer ledgers state which properties survive each mapping and which need separate treatment.

Case A · A low-carbon, water-aware, resilient data center

Decision and purpose

Suppose a cloud operator needs to place and schedule work across a regional fleet while replacing hardware and cooling systems. Reducing carbon is one aim, but the operator must also meet service commitments. We therefore consider lifecycle greenhouse-gas emissions and water use together with reliability, security, and data-location requirements. The assessment must prevent a nominal improvement from transferring environmental burdens to communities with less power to resist them.

The decision has several time scales. Every few seconds, schedulers route requests. Every few minutes, controllers adjust cooling and power states. Day-ahead systems defer batch work or purchase electricity. Annual planning chooses capacity, renewable contracts, cooling retrofits, and hardware replacement. Site selection and grid infrastructure create commitments lasting decades. A model that treats these decisions as one static allocation will conceal their different reversibility and evidence requirements.

Comparing a new schedule with an unconstrained optimum exaggerates benefits. The relevant baselines include current production policy, a simple carbon-threshold policy, a price-only scheduler, and a reliability-first policy with existing capacity. Benefits should be reported against all of them under the same workload traces and grid scenarios.

Boundary and affected parties

The narrow boundary contains servers, network, storage, cooling, and the purchased electricity reported by the operator. The decision-relevant boundary also includes upstream hardware manufacture, construction, refrigerants, backup generation, grid marginal response, water withdrawal and consumption, workload-induced network traffic, hardware disposal, and the user consequences of delay or failure.

Affected parties include service users; tenants with latency, fairness, and data-residency needs; operators and maintenance workers; electricity and water utilities; communities near generating plants and data centers; suppliers; and future populations exposed to emissions. Carbon reduction and distributional effects need separate assessment. A workload shifted away from a high-carbon region can increase water stress elsewhere. A site with low annual average emissions can depend on a marginal generator during the hour when work moves to it. A hardware refresh can lower operating energy while increasing embodied emissions and electronic waste.

Draw the lifecycle before defining the metric. The data flow should distinguish measured site electricity, modeled facility overhead, contractual renewable accounting, estimated marginal emissions, supplier declarations, and scenario-based embodied impacts. These evidence types should not be added without showing their provenance.

Variables and alternatives

At operational scale, let x_{jst} be the fraction or binary assignment of job j to site s and time interval t. Other decisions include the powered-capacity indicator or fraction z_{st}, cooling set points T_{st}, battery charge and discharge b_{st}, and reserve capacity r_{st}. Planning variables can select cooling technology, renewable contracts, network upgrades, spare capacity, and hardware type and replacement date.

Feasibility depends on technical, legal, and organizational conditions. Software architecture determines whether work is movable. Data sovereignty limits sites. User-facing requests may not be deferrable. Checkpointing, replication, accelerator availability, data locality, and dependency graphs make migration costly or impossible. Organizational contracts can be harder constraints than hardware capacity.

Representation should therefore begin with workload classes: interactive latency-critical, deadline-constrained batch, interruptible batch, stateful services, regulated data, and safety-relevant services. Each class carries different admissible transformations. For example, moving an interruptible training job in time preserves different requirements from moving a clinical decision service across jurisdictions.

Objective architecture

A first operational objective might combine electricity cost, marginal emissions, water consumption, service penalties, and wear:

$$\min_{x,z,T,b} \sum_{s,t}[p_{st}E_{st} + \lambda_c c_{st}E_{st} + \lambda_w \omega_{st}W_{st} + \lambda_d D_{st} + \lambda_h H_{st}]. \tag{110}$$

Here p_{st} is electricity price, E_{st} is electricity use, c_{st} is an operational emissions factor, W_{st} is water consumption, ω_{st} is a scarcity or local-impact factor, D_{st} is service degradation, and H_{st} is hardware or battery wear. The weights in Equation (110) permit substitution between these quantities. For example, a sufficiently large latency benefit may offset greater water use, or a cost saving may offset additional carbon. Equation (110) is useful for computation and inadequate as governance. We need to establish whether the organization permits these trade-offs before using the objective.

A safer architecture is layered. Reliability, security, data residency, and emergency-service commitments are hard constraints. Carbon and water receive annual budgets and site-level limits. Tenant fairness receives minimum service guarantees. Cost and residual impact are improved within that feasible region. Planning then compares nondominated portfolios across lifecycle emissions, water, land, cost, adaptability, and recovery time.

Operational and embodied carbon must not be collapsed without time. Manufacturing emissions occur now; operational savings accrue over a device's remaining life under uncertain grid decarbonization. The replacement decision should compare discounted or physically time-indexed emissions and include reuse. A "more efficient" accelerator may increase total demand through rebound. Report both per-service efficiency and absolute fleet impact.

Constraints, rights, and interfaces

Capacity constraints couple jobs assigned to a site:

$$\sum_j a_{jk}\, x_{jst} \le C_{kst} z_{st} \qquad \text{for each resource } k\text{, site } s\text{, and time } t. \tag{111}$$

Here a_{jk} is job j's demand for resource k, C_{kst} is installed capacity at site s in period t, and $z_{st} \in [0,1]$ is the powered fraction; a binary version switches this capacity on or off. Deadline, precedence, locality, and replication constraints define workload feasibility. Thermal dynamics constrain temperatures and ramp rates. Power contracts constrain peak draw and battery behavior. Reliability requires reserve, failure-domain diversity, and recovery. Security requires authenticated control, least privilege, and separation of optimization from protection.

Different institutions are responsible for these constraints. A scheduler is not authorized to replace a data-residency requirement with a penalty when the problem becomes infeasible. Some service targets, however, may be open to revision if their environmental effects were not considered when they were adopted. We therefore record the source and owner of each constraint and the procedure for changing a nonessential commitment.

The locally optimized components also need compatible interfaces. For example, a job scheduler sends a load profile to a cooling controller, which observes thermal state rather than business priority. A battery optimizer depends on tariffs and emergency reserve requirements. Procurement changes the hardware mix used in capacity forecasts. As in Chapter 7, differences in signals, time scales, and failure assumptions can prevent the individual guarantees from holding for the combined system.

Uncertainty and dynamics

Workload arrivals, job durations, renewable output, marginal emissions, weather, water conditions, equipment failures, and prices are uncertain. Forecast errors are correlated: a heat wave can raise cooling load, increase grid carbon intensity, reduce water availability, and raise service demand simultaneously. Independent scenarios understate tail risk.

Expected-value scheduling is appropriate for frequent, recoverable variation. Risk measures such as CVaR from Chapter 6 can protect tail service delay or cost. Robust sets can protect thermal and capacity feasibility against bounded forecast error. Resilience requires more: detecting model failure, shedding noncritical load, transferring state safely, operating locally during network partition, and recovering without data loss.

Use receding-horizon control for operational decisions. At each interval, update workload, thermal state, grid forecasts, and equipment availability; optimize a horizon; implement only the near-term action; and repeat. But constrain policy churn. Excessive migrations, set-point oscillation, or battery cycling can consume the savings. Stability and wear belong in the model.

Long-term planning should use adaptive pathways. Rather than commit today to one capacity forecast, specify triggers: retrofit cooling if water stress or temperature crosses a threshold; add storage if price volatility and carbon correlation reach a range; defer hardware

replacement if grid decarbonization reduces its lifecycle benefit. The option to wait has value when choices are irreversible.

Method and computation

The operational problem combines mixed-integer assignment, queueing, thermal dynamics, and uncertainty. A single monolithic solver may be too slow or brittle. Decompose by time scale and interface.

1. A planning model chooses capacity, technology, and carbon-water budgets under scenarios.
2. A day-ahead mixed-integer or convex approximation schedules movable batch work and energy assets.
3. A fast online policy routes requests and corrects forecast error.
4. Local safety controllers maintain thermal and electrical envelopes regardless of economic instructions.

Queueing relations provide checks on a proposed schedule. Little's law relates average work in the system, throughput, and time under stable conditions (Little 1961). With variable arrivals and service times, delay can increase rapidly as utilization approaches capacity. A schedule based on persistent near-total utilization therefore needs an explicit queueing assessment. Tail latency matters when a request waits for responses from several services (Dean and Barroso 2013).

Lagrange multipliers can coordinate decomposed sites: a carbon or network-capacity shadow price communicates scarcity without revealing every local variable. That is a transfer of duality from Chapter 4. It preserves marginal trade-off information under the model. It does not confer moral legitimacy on the price or ensure that communities can be compensated for harm.

Evidence and validation

Data-center demand and efficiency must be measured rather than inferred from headline growth, while water accounting must distinguish withdrawal from consumption and direct cooling from electricity-supply effects (Masanet et al. 2020; Ristic, Madani, and Makuch 2015). Carbon claims likewise depend on the emission factor appropriate to the intervention; average and marginal factors are not interchangeable (Elenes et al. 2022).

The evidence plan begins before deployment. Reconstruct at least a year of workload, weather, electricity, water, failures, and maintenance with known data quality. Back-test forecasts without leakage. Validate energy and thermal models across seasons and load regimes. Compare modeled job completion and tail latency with production. Reconcile facility meters with server telemetry and power-usage-effectiveness estimates.

Carbon accounting records emissions within a stated system boundary and time period. Carbon validation distinguishes two questions. Average and marginal factors answer different questions about attributed emissions and response to a change. Location-based and market-

based reporting use different electricity-accounting conventions. State both choices and their purpose. Water accounting distinguishes withdrawal from consumption and includes local scarcity. Embodied impacts need supplier data, allocation assumptions, uncertainty ranges, and treatment of reuse.

Run the optimizer in shadow mode. It produces actions without controlling the system; operators inspect feasibility, predicted benefit, and surprising behavior. Then use staged deployment by workload class and site, with holdbacks where interference is manageable. Stress tests should include heat waves, grid emergencies, network partition, sensor drift, stale carbon data, correlated hardware failure, and malicious input.

Measure absolute impact as well as modeled savings. Did total energy, emissions, and water change after workload growth and rebound? Did latency burdens fall disproportionately on low-paying tenants? Did operators create manual workarounds? A reduction claimed by the optimizer should be reconciled against bills, meters, service outcomes, and independent lifecycle review.

Governance, monitoring, and failure response

Create a decision board with operations, reliability, security, sustainability, finance, tenant representation, and local environmental expertise. Each role owns distinct constraints and has defined veto or escalation authority. Weight changes, boundary changes, and new workload classes require review.

The operational dashboard should show service, carbon, water, cost, reserve margin, forecast error, and distance from validated conditions. Alert when data is stale or a policy relies on an extrapolated model. Preserve a manual safe policy and the capacity to restore it. Security incidents, major model error, thermal instability, inequitable service, or environmental-budget breach trigger rollback.

Publish an impact report that distinguishes avoided modeled impact, measured operational change, contractual accounting, and lifecycle estimates. Communities should receive information relevant to local water, noise, backup generation, and grid consequences, not only a global carbon total. Complaints and local observations are evidence and must enter review.

Transfer ledger

Queueing theory relates arrivals, work in the system, service capacity, and delay. When applying it to computing workloads, we also need to describe the value of different services and the contractual history of their targets.

Optimal control can represent cooling and battery dynamics, state constraints, forecasts, and receding-horizon correction. Operator practices and rules governing energy use require additional representation.

Dual decomposition can coordinate the fleet through marginal scarcity signals under the required separability assumptions. A shadow price does not establish compensation, consent, or a fair distribution of effects.

Lifecycle assessment follows material and energy flows across stages under specified allocation rules. We also need to examine unmeasured local harms, rebound, and disagreements about valuation.

Reliability engineering supplies methods for examining failure modes, redundancy, detection, recovery, and common causes. The decision about essential services and who may receive reduced service remains specific to the institution and affected communities.

The resulting policy has separate claims about service, climate, water, security, and equity. We can assess the transfer by checking these claims and the properties omitted from each contributing model.

Implementation economics and tenant contracts

A schedule may reduce modeled environmental impact while conflicting with existing contracts. Tenant service targets, reserved capacity, internal prices, renewable claims, and ownership of savings affect whether the change can be adopted. We can therefore include contracts and charging rules among the design variables. For example, a tenant charged for delay but not carbon has little incentive to defer work even when deferral benefits the fleet.

Pilot a tariff or service class that exposes a small menu: immediate, deadline-flexible, location-flexible, or carbon-water budgeted service, with truthful disclosure and no coercive degradation of the baseline. Estimate actual flexibility rather than assuming every batch job can move. Protect small tenants from a market in which large customers purchase all low-impact capacity.

Capital planning must reconcile organizational and physical depreciation. Hardware may be financially depreciated while still physically useful, or operationally inefficient while embodied impact makes immediate replacement unattractive. Compare keep, repurpose, resell, and replace pathways. Record rebound assumptions and who receives avoided cost.

Also test whether optimization displaces operator attention. A policy that requires constant exception handling can transfer environmental benefit into hidden labor and reliability risk. Staff workload, explainability, and incident recovery belong in the measured implementation cost.

Case checklist

- Define operational, planning, and infrastructure time scales separately.
- Distinguish average, marginal, contractual, and lifecycle carbon claims.
- Weight water by local context and report withdrawal and consumption.
- Preserve reliability, security, residency, and essential-service constraints.
- Validate optimized candidates, not only ordinary operating points.

- Monitor absolute impact, rebound, workload fairness, and operator burden.
- Maintain a safe fallback policy and explicit rollback triggers.

Case B · An equitable heat-health program

Decision and purpose

A metropolitan authority expects more frequent and severe heat. It must choose a portfolio of near-term services and long-term changes: heat alerts, outreach, transport, cooling centers, home cooling and weatherization, tree canopy, shade, cool roofs and pavements, worker protections, clinical surveillance, utility support, and emergency staffing. The purpose is to prevent illness and death, reduce unequal exposure and vulnerability, preserve dignity and access, and adapt the urban fabric without creating new burdens.

The interventions act through different mechanisms and at different times. An alert helps only if people receive it, trust it, and can act. A cooling center may be unreachable, stigmatizing, inaccessible, or unsafe. Trees provide shade but require time, water, and maintenance; greening can also contribute to displacement. Air conditioning reduces indoor exposure while increasing electricity demand. Housing retrofit can reduce heat and energy costs, subject to landlord and tenant incentives.

We can formulate this as a portfolio and pathway problem. The authority should decide what to do now, what capability to build, which irreversible commitments to avoid, and which observations will trigger later action.

Boundary and affected parties

The physical boundary includes weather, urban heat islands, buildings, vegetation, energy and water systems, transport, and health services. The social boundary includes housing quality, disability, age, chronic illness, medication, outdoor work, income, language, social isolation, carceral settings, homelessness, immigration status, caregiving, and trust in institutions. Exposure depends on infrastructure and institutions as well as weather.

Affected parties include residents, workers, students, patients, caregivers, landlords and tenants, employers, utilities, emergency services, community organizations, transit agencies, and neighboring jurisdictions. The groups used in the model must not be treated as homogeneous. “Older adults” can include independent residents with strong networks and isolated residents without cooling. Neighborhood averages can hide dangerous apartments, street segments, or work sites.

The boundary must include displacement and rebound. A greening project that raises property values can displace the people whose risk justified it. A home-cooling subsidy can lower exposure and raise energy debt if utility support is absent. A center that concentrates services may reduce travel for the agency and increase travel for residents. These distributional effects belong in the assessment of the intervention.

Variables and alternatives

Let x_i denote the scale or selection of intervention i, and $y_{i,g,t}$ the allocation of service i to neighborhood or group g in period t. In Equations (112)–(113), the portfolio vector x subsumes both the intervention choices and these allocations. Continuous, binary, spatial, and pathway variables represent outreach, facility opening, location, and later triggers. The index g denotes a group in this case, rather than a constraint function.

Alternatives should be constructed with communities and operators before optimization. A model can only choose among represented actions. Residents may identify evening transport, trusted-language outreach, pharmacy check-ins, pet accommodation, drinking-water access, or tenant enforcement as decisive options absent from an engineering inventory. Clinicians may identify medications or conditions requiring special protocols. Workers may reveal that a formal rest rule is unenforceable without wage protection.

Include “maintain existing services,” “repair reliability first,” and “do not build” alternatives. A new cooling center should compete with keeping libraries, clinics, and community facilities open longer. A sensor network should compete with funding outreach and enforcement. The optimizer should not inherit a capital-project bias.

Objective architecture

For group g, scenario ξ, and portfolio x, let $R_g(x,\xi)$ be residual risk of a defined adverse health outcome over a stated period. Let N_g be group population. A pure utilitarian objective would minimize population-weighted expected risk:

$$\min_x \ \sum_g N_g \ \mathbb{E}\big[R_g(x,\xi)\big]. \tag{112}$$

This objective can prefer many small improvements for already protected populations over urgent protection for a smaller high-risk group. Define total population harm $L(x,\xi) = \sum_g N_g\, R_g(x,\xi)$, portfolio cost $C(x)$, and budget B. Let X_H contain accessible portfolios satisfying $C(x) \le B$, $\mathbb{E}\big[R_g(x,\xi)\big] \le \tau_g$, and $\Pr\big(R_g(x,\xi) > r_g\big) \le \varepsilon_g$ for every designated priority group. The group limits and tail term then give

$$\min_{x \in X_H} \Big(\sum_g N_g \ \mathbb{E}\big[R_g(x,\xi)\big] + \lambda\, \mathrm{CVaR}_\alpha\big(L(x,\xi)\big) \Big). \tag{113}$$

Equation (113) combines average benefit, tail protection, group-specific limits, budget, and accessibility. The thresholds are not generated by the solver. They come from public-health evidence, legal duties, capability judgments, and deliberation. Sen’s and Nussbaum’s capability approaches help explain why access to bodily health, mobility, affiliation, and control cannot be reduced to aggregate welfare or revealed preference alone; their use requires deliberative judgment (Sen 1985; Nussbaum 2011).

Some obligations should remain outside probabilistic trade. An accessible evacuation route, nondiscriminatory service, safe worker practice, and emergency duty cannot be waived

because the model assigns low event probability. Similarly, cultural and neighborhood continuity may require procedural protections that a risk score cannot encode.

Constraints, mechanisms, and interaction

The portfolio faces budgets, workforce, land, water, electrical capacity, procurement, seasonal timing, and maintenance constraints. It also faces behavioral and institutional mechanisms. Outreach requires a trusted messenger and current contact path. Cooling requires reliable power and an affordable bill. Shade requires a reachable route at the time of exposure. Worker protection requires enforcement and protection against lost income. Clinical surveillance requires privacy and a plan for response.

Interventions interact. Weatherization can improve cooling efficiency; tree canopy can reduce building load; transit can make centers accessible; utility shutoff protection can make equipment usable; clinical outreach can target people identified through nonintrusive community networks. Other interactions are adverse. Trees can conflict with water constraints or power lines. Reflective surfaces can increase glare or shift radiant load. Added cooling demand can stress the grid during the event.

Represent these mechanisms explicitly where evidence permits. When it does not, preserve them in scenario narratives and feasibility review. An unsupported coefficient can conceal uncertainty that should remain explicit.

Uncertainty and adaptive pathways

Uncertainty includes climate trajectories, event timing, humidity, smoke, power failures, exposure-response relationships, intervention uptake, migration, land-use change, and future budgets. Historical averages are a weak guide under a changing climate (Intergovernmental Panel on Climate Change 2021). Yet maximizing against an unconstrained worst case can direct all resources to an implausible scenario. Use a plural strategy: stochastic scenarios for frequency, robust constraints for critical service under forecast error, and adaptive pathways for long-lived investments.

An adaptive pathway might fund immediate outreach, facility-hours, worker rules, home audits, and grid-resilience measures; begin canopy and retrofit in priority areas; and reserve land or design options for later centers. Trigger additional cooling capacity when indoor-temperature surveillance, outage frequency, or health outcomes cross predeclared thresholds. Review triggers annually because observation systems and populations change.

We test compound events, such as heat with wildfire smoke, a power outage, a transport disruption, or repeated hot nights. The assessment also needs to show how the authority detects failure, reaches isolated people, maintains safe spaces, restores service, and learns from the event. Low expected risk alone does not establish these capabilities.

Method and computation

The planning problem is a mixed-integer, spatial, multiobjective portfolio under uncertainty. Use geospatial preprocessing to estimate hazard and accessibility; epidemiologic models to estimate conditional health response; building and energy models for indoor exposure and grid demand; network models for travel and outreach; and optimization for portfolio comparison. Keep the components separate enough that their assumptions can be challenged.

Start with dominance screening and budget scenarios. Generate a Pareto set across total expected risk, priority-group risk, tail loss, cost, time to benefit, and maintenance burden. Use robust or distributionally robust analysis where parameter uncertainty is material (Ben-Tal, El Ghaoui, and Nemirovski 2009; Shapiro, Dentcheva, and Ruszczyński 2021). Then take a small set of diverse portfolios into deliberation. Do not ask a community board to choose among thousands of solver outputs.

Shadow prices can show where budget, staff, or grid capacity binds. We can use them to identify questions for further investigation before allocating funds. A high shadow value on outreach capacity may justify training and partnerships. A high value on a group-risk threshold may reveal either an expensive commitment or a misspecified intervention set. Extreme values are model diagnostics.

Evidence and validation

The intervention portfolio follows the current WHO heat-health action-plan framework: governance, warning, identification of populations at increased risk, communication, health-system resilience, exposure reduction, surveillance, and monitoring and learning (World Health Organization Regional Office for Europe 2026). Local thresholds and delivery mechanisms still require local epidemiology, service data, and community validation.

Build an evidence chain for each mechanism. Weather and land-surface data support hazard estimates. Housing audits and indoor sensors support exposure estimates, with privacy and sampling protections. Health records support outcome models but may undercount people with poor access to care. Surveys and community organizations support access and trust claims. Facility logs support capacity and use. Utility data support outage and energy-burden analysis.

We validate at the spatial and temporal scale of the decision. A model calibrated on citywide mortality may not predict neighborhood ambulance demand. People with different mobility, sensory, cognitive, language, and caregiving needs can test actual routes to cooling centers. An alert needs to be understood and acted on as well as delivered. Travel estimates also need the time of day, heat exposure along the route, and possible service disruption.

Use pilot deployments where feasible, but do not withhold established protection merely to create a clean experiment. Phased implementation, matched comparison, interrupted time

series, and process evidence may be more ethical than individual randomization. Predefine outcomes and analyze distribution, not just average attendance or temperature.

After every event, reconcile forecast, operations, and outcomes. Investigate who did not receive service, which facilities failed, where staff improvised, and which data arrived too late. Near misses and community testimony should update the model even when statistical power is limited.

Governance and accountability

Governance should join public health, emergency management, housing, planning, labor, disability access, utilities, clinicians, community organizations, and residents from priority areas. The authority must publish how priority groups, thresholds, and budgets were chosen. Residents need a route to correct data, challenge facility plans, and report inaccessible or unsafe services.

Create role-specific authority. Public-health officials can trigger emergency actions; access officers can reject inaccessible plans; labor regulators enforce protections; utilities own restoration protocols; community partners can activate outreach and report loss of trust. No weighted vote should permit one role to erase another's statutory duty.

Monitor service reach, indoor exposure, health outcomes, outages, energy burden, worker compliance, center conditions, and displacement indicators. Roll back or redesign interventions that create heat, glare, surveillance, policing, unaffordable bills, or exclusion. Long-term projects need maintenance covenants and anti-displacement measures; the expected shade depends on the tree's survival and continued maintenance.

Transfer ledger

Climate hazard models describe heat intensity and frequency under specified scenarios. Public-health planning also needs information about individual exposure, social protection, and trust in institutions.

Reliability engineering can describe emergency-service capacity, redundancy, failure domains, and recovery. We also investigate why some people cannot or will not use a service that is nominally available.

Portfolio optimization represents shared budgets, interactions, timing, and trade-offs between interventions. It cannot select actions absent from the model or establish that the chosen priorities are legitimate.

Control and adaptive pathways connect observations with triggers, staged commitments, and revision. In a public institution, an observed trigger leads to action only if the necessary authority and funding are available.

Capability ethics distinguishes formal availability from the actual ability to remain safe. Using this distinction in feasibility resists reduction to preference satisfaction and requires deliberative judgments about access and circumstances.

Including equity in the formulation changes feasibility, introduces group constraints, and can change the value assigned to access interventions. It can therefore change the preferred portfolio before the final decision review.

A distributional audit in space and time

For each candidate portfolio, produce maps and tables of baseline exposure, intervention reach, residual risk, travel or paperwork burden, energy cost, outage exposure, and time to benefit. Show uncertainty and population counts. Do not publish fine-grained health or vulnerability data that can identify individuals; use privacy-preserving aggregation and community review.

Compare at least four lenses: total benefit, worst residual risk, change for priority groups, and people made worse off. A citywide improvement can coexist with one neighborhood receiving heat, glare, policing, construction burden, or displacement. Report these mechanisms separately instead of subtracting them from an aggregate benefit.

We also compare the timing of benefits. Emergency services can protect this season; canopy and housing take years. A portfolio that is equitable at maturity can leave a decade-long protection gap. Add interim thresholds and budgets, and monitor whether long-term projects are maintained. If a priority community waits longer for benefit, compensate with immediate measures and decision authority.

The audit should be discussed with people who can verify whether mapped access is real. A route that crosses a dangerous road, a center that excludes pets, or an outreach list that omits undocumented residents can invalidate a nominal allocation.

Case checklist

- Model exposure, vulnerability, and ability to act—not temperature alone.
- Construct interventions with communities and frontline operators.
- Use priority-group and tail-risk constraints alongside total benefit.
- Test compound events, outages, smoke, access, and displacement.
- Validate alerts for action and facilities for real accessibility.
- Publish thresholds, authority, maintenance, and appeal routes.
- Treat after-action testimony and near misses as evidence.

Case C · Closed-loop insulin delivery as adaptive care

Decision and purpose

A closed-loop insulin system repeatedly estimates metabolic state and recommends or delivers insulin. We consider its purpose for a particular person: maintaining a clinically appropriate range, sharply limiting dangerous low glucose, and avoiding excessive dosing or oscillation. The system also needs to support sleep and daily activities and reduce cognitive work. The person and clinical team need to understand its behavior and be able to interrupt or adapt it.

The decision is safety-critical and asymmetric. Too little insulin can lead to sustained high glucose and acute or long-term harm; too much can produce a rapid emergency. The same measured glucose can require different action depending on trend, recent dose, meal absorption, exercise, illness, stress, alcohol, hormones, sensor quality, and individual physiology. The controller acts under delay: subcutaneous insulin takes time, and continuous glucose sensing measures an interstitial signal with lag and noise.

This case illustrates an optimization structure and is not a dosing design. Device development and care require current clinical, engineering, cybersecurity, human-factors, and regulatory expertise. We use the formulation to identify the model's assumptions and the responsibilities surrounding its use.

Boundary and affected parties

The system boundary contains the person, sensor, infusion set and pump, insulin, controller, user interface, mobile or embedded software, alarms, charging and consumables, clinician configuration, data services, update process, caregiver access where authorized, and emergency fallback. It also contains daily life: meals are incompletely announced, exercise can be spontaneous, sleep limits attention, and connectivity or supplies can fail.

Affected parties include the user, caregivers selected by the user, clinicians, device and software teams, pharmacists and suppliers, emergency personnel, regulators, and security responders. Their authority is not symmetric. The person living with the system owns goals and burden trade-offs within clinical safety. Clinicians define and review therapeutic parameters. Manufacturers are responsible for device behavior and evidence. Remote service providers should not acquire decision authority merely because they process data.

We need to specify the conditions under which human override can provide protection. An override is useful only when the person is awake, informed, physically able, has supplies, understands the alarm, and can act before harm. Automatic fallback must remain safe under realistic human absence.

State, variables, and alternatives

Let G_t be glucose-related state at time t, u_t insulin delivery, d_t known or estimated disturbance such as meal or exercise, and I_t insulin-on-board state. A model may include compartments for glucose appearance and insulin action. The controller estimates hidden state from sensor history, commands, and contextual inputs.

Decision variables include basal modulation, correction micro-boluses where supported, temporary targets, alarm timing, and adaptation parameters. Design variables include sampling, horizon, model structure, constraints, fault detection, interface, and fallback mode. Clinical variables include individualized target range, insulin sensitivity, carbohydrate ratio, active-insulin time, and conditions requiring manual operation.

Alternatives include standard pump therapy, sensor-augmented therapy, predictive low-glucose suspend, hybrid closed loop with meal announcements, and more automated control

under specified conditions. Comparison must include the additional interactions, alarms, training, supplies, and failure modes alongside simulated performance.

Objective architecture

At time t, the controller selects doses $u_{t:t+H-1}$ over horizon H. Define $\phi(G) = w_h\big(G - G_{\text{high}}\big)_+^2 + w_l(G_{\text{low}} - G)_+^2$, where $(v)_+ = \max(v, 0)$. The set $\mathcal{U}_t$ enforces dose bounds $0 \le u_{t+k} \le \bar{u}_t$ for $k = 0, \dots, H-1$, insulin-on-board bounds $I_{t+k} \le \bar{I}_t$ for $k = 1, \dots, H$, and state transitions $(G_{t+k}, I_{t+k}) = F_{\theta_t}(G_{t+k-1}, I_{t+k-1}, u_{t+k-1}, d_{t+k-1})$ for $k = 1, \dots, H$. With smoothing weight $r \ge 0$ and last applied dose u_{t-1} fixed, a stylized objective is

$$\min_{u_{t:t+H-1} \in \mathcal{U}_t} \sum_{k=1}^{H} [\phi(G_{t+k}) + r\,(u_{t+k-1} - u_{t+k-2})^2]. \tag{114}$$

The low-glucose weight w_l is much larger than the high-glucose weight over the near horizon because harms are asymmetric. The positive part ensures that only violations beyond the range are penalized in these terms. Dose and insulin-on-board bounds supply safety constraints. Equation (114) illustrates receding-horizon structure (Rawlings, Mayne, and Diehl 2017); a clinical implementation requires richer models, fault handling, verification, and evidence.

A scalar cost is insufficient. Safety requirements should specify maximum delivery, rate of change, response to missing or implausible sensor data, behavior after manual dose, and safe degradation. Human factors add constraints on alarm burden, comprehensibility, and mode visibility. Privacy and security add collection, authentication, update, and access controls. Performance is then evaluated across time in clinically meaningful ranges, events, variability, burden, and subgroup or individual response.

Uncertainty and adaptation

Different sources of uncertainty have different mechanisms. Parameters may be uncertain, state estimates may be inaccurate, and sensors may have bias or delay. Infusion can fail, meals may be unannounced, and exercise or illness can change physiology. Behavioral variation is a further source: routines, announcements, and responses change. An unannounced meal is also an observability problem, because the controller lacks information about a material input. A blocked infusion set changes the controlled system, whereas compression can produce a structured sensor error. We therefore cannot represent all these conditions as identically distributed noise.

Robust design should separate ordinary variability from faults. The controller can adapt within validated parameter bounds while a fault monitor tests sensor consistency, delivery response, and communication. If confidence falls, it uses a validated conservative mode and informs the user. Conservative behavior need not mean uniform inactivity: the appropriate safe response depends on the fault, physiology, and current mode. Adaptation must not silently drift outside the evidence envelope.

Personalization is valuable and potentially confounded. If the algorithm changes at the same time as the person's routine, a fitted improvement can be spurious. Use bounded updates, minimum data requirements, conservative priors, change detection, and clinical review. Preserve a versioned history of parameters, triggers, and outcomes so a deterioration can be diagnosed and reversed.

Method and computation

The formal problem combines nonlinear dynamics, partial observation, constraints, and hybrid modes. Model predictive control provides a disciplined architecture because it forecasts, handles constraints, and repeats after new measurement (Rawlings, Mayne, and Diehl 2017). State estimation filters sensor data and propagates uncertainty. Robust or stochastic variants can test parameter and disturbance scenarios. A safety supervisor can enforce invariants independent of the performance optimizer.

Computation must meet deterministic timing and resource limits. Solver failure, numerical ill-conditioning, overflow, time synchronization, battery depletion, and update interruption are system hazards. Verification should cover not only mathematical source but generated code, units, data types, configuration, and hardware integration. Every control output should be traceable to valid inputs, current mode, and a tested software version.

The development process uses simulation, hardware-in-the-loop, controlled studies, and monitored real-world evidence. A high-fidelity simulator can generate rare scenarios, but an optimizer can exploit physiological-model error. Include diverse virtual subjects and challenge cases, then verify behavior with independent models and domain review.

Evidence and validation

Clinical evidence must compare the complete closed-loop system with an appropriate control and report both target-range and safety outcomes. A six-month randomized multicenter trial found improved time in range and illustrates why infusion-set failure, software configuration, and adverse-event monitoring belong in the evidence plan (Brown et al. 2019).

Validation proceeds from component to integrated claim. Test sensor behavior under lag, drift, missing data, compression, and interfering conditions. Test pump delivery, occlusion detection, and command limits. Verify controller code against the formal design. Use hardware-in-the-loop to test timing, communications, battery, and faults. Challenge the complete system with meals, exercise, sleep, illness, sensor replacement, manual dosing, and connectivity loss.

Clinical evaluation should report more than a mean. Show time in relevant ranges, frequency and duration of low and high events, variability, rescue interventions, alarm burden, discontinuation, sleep, quality of life, and differential performance. Decision-curve thinking is useful: the clinical value of a threshold depends on the relative consequence of missing and triggering action, not discrimination alone (Vickers and Elkin 2006).

External validity requires diversity in age, physiology, schedule, language, disability, device experience, access to supplies, and care setting. Average benefit can coexist with an unsafe subgroup or an unusable interface. Missingness and discontinuation can provide evidence about system use and should be analyzed.

Post-deployment monitoring should distinguish device fault, model mismatch, use difficulty, supply interruption, and clinical change. Near misses, overrides, manual-mode episodes, complaints, and cybersecurity signals enter the evidence chain. Updates require regression testing and a reasoned statement of whether existing evidence transfers to the changed system.

Governance and failure response

The person using the device must know current mode, recent delivery, data quality, active alerts, and how to obtain help. Interfaces should avoid silent automation and alarm floods. Training includes ordinary use, sick-day or exercise guidance as clinically directed, sensor or infusion failure, manual fallback, and emergency response.

Clinicians need interpretable summaries and the ability to review parameter history, not an unmanageable stream of raw data. Manufacturers need safety monitoring, secure disclosure, update control, and prompt communication. Regulators require evidence matched to the intended population and claims; adaptive software must remain under lifecycle control (U.S. Food and Drug Administration, Health Canada, and Medicines and Healthcare products Regulatory Agency 2021). Data access should be purpose-limited, encrypted, logged, and revocable where possible.

Rollback operates at several levels: revert a model or software update; restore prior parameters; enter a validated safe mode; or discontinue automation with a clinically established alternative. A failure response must not depend on cloud connectivity. Identifiable organizations and professionals remain accountable for the system.

Transfer ledger

Model predictive control provides constrained actions over a receding horizon, with correction after each observation. In insulin delivery, we also need physiological mechanisms omitted from the model and the person’s experience of alarms, attention, and trust.

Robust optimization provides protection against the specified parameter and disturbance uncertainty. Novel faults may lie outside that set, so detection, recovery, and fallback remain necessary.

Fault-tolerant control represents detection, mode changes, invariants, and recovery. Safe degradation also requires evidence that the person can understand and manage the resulting mode.

Clinical decision analysis relates thresholds to asymmetric benefits and harms (Pauker and Kassirer 1980). We need to revisit this relation when the person's preferences or circumstances change.

Software assurance traces requirements through code and tests. It does not establish clinical benefit. We can combine engineering and clinical evidence only when their assumptions about the interface are compatible.

We can transfer dynamic equations from control engineering to insulin delivery while retaining the person's goals, clinical responsibilities, and conditions of daily use. The controller operates within this larger care system.

Human-factors and alarm workload

Alarm design is a constrained communication problem. The system must convey urgency, cause where known, immediate action, and confidence without overwhelming the person. Alarm frequency affects response; missed or repeated nonactionable alarms change behavior. Evaluate comprehension and action under sleep, exercise, stress, sensory difference, and competing tasks.

Modes should be perceptible. The interface must distinguish automated, limited, fallback, suspended, and manual operation, and explain what delivery continues. A transition should not rely on a transient notification. Confirm critical actions and avoid terminology that implies certainty the system lacks.

Caregiver or clinician sharing should be chosen and revocable by the user within safety and legal requirements. Escalation delays, duplicate alarms, and unclear responsibility can create diffusion rather than protection. Test the complete route from device signal to effective assistance.

Usability assessment includes actions taken incorrectly, required actions omitted, recovery time, retention of training, and workload over several weeks. Success during one supervised session does not establish performance in ordinary life. A system may work when users watch it closely and fail when automation appropriately reduces the attention they must devote to routine control. We therefore test the human reliability assumptions under realistic conditions.

The final design record should connect every safety claim to a test and every user-facing mode to training and fallback. It should preserve alternatives rejected for burden as well as those rejected for control performance. When a person's goals or circumstances change, clinicians and the user need enough history to distinguish physiological change from software, parameter, sensor, or behavior change. That traceability is part of continuity of care.

Review the counterfactual "automation unavailable for one week." Supplies, instructions, clinical contacts, and retained skills must support safe continuity. Dependence on a remote service, proprietary account, or scarce consumable belongs in the system claim and procurement assessment.

The review also covers travel, hospitalization, school or work, and emergency handoff across organizations. Test the handoff with realistic records, time pressure, and an unfamiliar responder. Record missing information and revise the emergency interface before wider use.

Case checklist

- Define safety asymmetry, individualized purpose, and validated operating conditions.
- Model delays, insulin on board, faults, unannounced disturbances, and fallback.
- Keep safety invariants separate from performance optimization.
- Validate component, code, hardware, integrated behavior, and clinical outcomes.
- Report range, events, variability, burden, discontinuation, and differential effects.
- Version adaptations and preserve a path to prior software and parameters.
- Ensure the user has mode visibility, authority, supplies, and an offline fallback.

Comparative synthesis: what feedback means in three domains

The three cases use feedback under uncertainty. We can compare them by asking what the state represents, which actions are possible, and what constitutes a failure.

In the data center, state is largely instrumented equipment and workload; action is scheduling, set point, or capacity; many mistakes are economically recoverable, while outages, security breaches, and environmental limits create harder boundaries. Feedback can run rapidly, and aggregate data is often meaningful.

In heat-health policy, state is distributed across weather, buildings, institutions, and people. Observation is incomplete and shaped by access to care. Action includes public service, infrastructure, law, and trusted communication. Feedback is slow and political: a measured uptake rate reflects both intervention and institution. Aggregate uptake should therefore be accompanied by an examination of people the intervention did not reach.

In insulin delivery, state is an individual physiological estimate; action directly changes the body; delay and asymmetry are acute. Feedback is fast but noisy, and safe fallback must remain personal and immediate. Population evidence informs but does not replace individual adaptation.

The common loop of observation, estimation, action, and measurement operates under different requirements. In a data center, these concern service, security, resource limits, and communities. In public health, they concern capability, access, and public duty. In care, they concern bodily safety, autonomy, and clinical responsibility.

The cases also show three uses of robustness. Fleet scheduling uses uncertainty sets and reserve to maintain feasibility. Heat adaptation uses scenarios and pathways because distributions and institutions change. Insulin control uses bounded adaptation plus fault detection because novel faults cannot be treated as ordinary noise. We retain these design differences when describing each use of robustness.

Their evidence loops also have different stopping rules. A cloud policy can expand after shadow operation and staged load. A public-health intervention may need action before randomized evidence, with protective baselines and after-event learning. A medical controller requires progressively stronger component, integrated, clinical, and lifecycle evidence. The common formulation therefore requires different supporting evidence in each application.

Boundary perturbation exercise

We can examine boundary choices by adding or removing a layer and checking whether the recommended policy changes for a defensible reason.

For the data center, first optimize site electricity alone. Then add marginal grid response, water scarcity, embodied hardware, tenant fairness, and community exposure one layer at a time. A schedule that initially moves work toward a low-price site may reverse after water or grid correlation enters. A replacement plan may reverse after embodied impact and reuse enter. Record which decision changes and which layer caused it. This analysis concerns changes to the objects represented in the model as well as changes to their parameter values.

For heat-health, begin with outdoor temperature and citywide health risk. Add indoor exposure, power failure, mobility, language, work, housing tenure, and displacement. The preferred portfolio should move from broad temperature reduction toward trusted delivery, home protection, worker rules, and group thresholds when those mechanisms are material. If it does not, inspect whether the new variables affect feasibility or merely sit in an unused report.

For the insulin example, we begin with a nominal physiological model and mean tracking. We then add asymmetric low-glucose harm, delays, insulin on board, sensor faults, exercise, alarm burden, user-selected modes, and offline fallback. The formulation needs to distinguish these conditions. Replacing every safeguard with a penalty may permit unsafe trade-offs, while treating every uncertainty as the worst possible case may make the controller unusable. Explicit constraint types and mode-transition rules help express the required responses.

The exercise helps determine whether the boundary is sufficient for the decision. A model that includes everything becomes untestable. Retain an element when it can change action, validate a constraint, explain a failure, or assign authority. Keep other consequences in monitoring and deliberation. Document excluded layers with the condition that would bring them inside.

Comparing evidence without ranking disciplines

The cases use bench measurement, telemetry, epidemiology, simulation, trials, interviews, participatory mapping, human-factors tests, and post-event review. Each source supports particular claim types, which we need to identify before comparing evidence strength.

Bench and hardware tests support component behavior under controlled conditions. Telemetry supports operational distributions and can be endogenous to policy. Epidemiology supports population associations and causal claims under specified designs. Simulation

supports conditional counterfactuals. Clinical study supports benefit and harm for a defined intervention and population. Interviews and participatory work support mechanism, meaning, access, and acceptability claims often absent from recorded variables. After-action review supports reconstruction of a coupled failure.

Combining evidence requires compatible interface assumptions. A heat model plus an exposure-response curve plus a facility-access model supports a conditional risk estimate only if populations, spatial resolution, behavior, and uncertainty align. A verified controller plus tested pump plus clinical model supports safety only if timing, units, failure modes, and intended use align. Evidence strength is lost at interfaces when bridge assumptions are untested.

We can record these relationships in a claim-evidence graph. Its nodes identify claims, its edges identify inferences, and its sources carry provenance. Defeaters describe observations that would invalidate an inference. This representation allows us to combine contributions while retaining the validation requirements of each discipline.

Field checklist

- Distinguish average, marginal, contractual, and lifecycle accounting before comparing infrastructure alternatives.
- Link heat warnings to funded actions, accountable roles, accessible communication, and distributional monitoring.
- Treat closed-loop medical control as a clinical service with alarms, fallback, training, cybersecurity, and adverse-event review.
- Validate feedback at the time scale on which each domain can safely observe and act.
- Preserve a baseline, rollback route, and authority to suspend automation.
- Record what each transfer preserves and what it forgets.

Translation lens for Chapter 18. Infrastructure and health both use feedback, robustness, and staged decisions, but their invariants differ. A data center preserves service and environmental commitments; a heat program preserves capabilities and public duties; an insulin system preserves safety and personal agency. The shared grammar coordinates evidence without making these purposes interchangeable.

Chapter 19 · Worked translations II: knowledge, creativity, fabrication, and climate

Case D · A public library collection and discovery system

Decision and purpose

A public library must allocate a constrained acquisition, licensing, preservation, space, cataloging, programming, and outreach budget. It also wants to improve discovery across shelves, catalogs, and recommendations. Circulation measures one aspect of use. The public purpose includes several aims: enable informed agency, learning, research, recreation, encounter, cultural memory, local authorship, minority-language use, and access for people underserved by commercial markets.

The library chooses what to acquire, license, retain, relocate, digitize, preserve, display, recommend, or remove. It also assigns work to metadata creation and physical or digital access. These decisions have different time horizons. A display may affect discovery this month, a license may expire, and an archive may be lost permanently if discarded. Cataloging can support uses that become apparent much later.

Recorded demand depends partly on earlier acquisitions, language coverage, opening hours, interfaces, education, marketing, transport, accessibility, and recommendations. If we train only on recorded use, the resulting policy may repeat the exclusions that produced those records.

Boundary and affected parties

The boundary contains patrons and nonusers, physical branches, mobile and outreach services, staff, collections, publishers and vendors, digital platforms, schools and universities, local archives, community organizations, privacy law, storage and preservation, and the urban or rural transport context. It includes the time needed to browse and the expertise needed to catalog, curate, mediate, and teach.

Affected groups include children, students, researchers, job seekers, migrants, linguistic minorities, disabled patrons, older adults, incarcerated or institutionalized people where served, local creators, people without reliable internet or private space, and future residents who cannot express current demand. Staff capacity has several components: subject knowledge, language, community trust, preservation skill, and accessibility expertise are distinct resources.

Digital borrowing introduces additional boundary conditions. License terms can limit simultaneous use, preservation, privacy, accessibility, and transfer between institutions. A commercial recommendation service can observe reading interests that the library would otherwise protect. Vendor concentration can make short-term convenience a long-term lock-in decision.

Variables and alternatives

Let $x_i \in \{0,1,2,\dots\}$ be copies or licenses of item i, r_i a retention decision, s_{ib} allocation to branch b, and h_{ikt} exposure of item i through interface or display position k at time t. Additional decisions allocate cataloging, conservation, digitization, programming, and outreach hours.

An individual item is not always the appropriate unit of analysis. A collection may provide coverage of a language, subject, period, format, reading level, or community record. Series and reference works may be useful together. An archive also has provenance and order that would be lost if its documents were treated independently. A license may bundle wanted and unwanted material. We therefore choose a representation suited to the intellectual and material object.

Generate alternatives with librarians and communities. Options include cooperative purchasing, interlibrary lending, rotating deposits, shared preservation, demand-driven acquisition with safeguards, targeted retrospective cataloging, multilingual metadata, sensory-accessible formats, outreach collections, and repair of opening-hour or transport barriers. Including these actions expands the feasible set beyond individual purchasing decisions.

Objective architecture

For collection alternative x, define measures for expected use $U(x)$, subject breadth $B(x)$, linguistic and cultural representation $L(x)$, educational and research support $Q(x)$, preservation value $P(x)$, accessibility $A(x)$, serendipitous exposure $S(x)$, and lifecycle cost $C(x)$. We can use this vector to organize deliberation while keeping the meanings and scales of its components distinct.

A portfolio model can maximize a declared scalarization while enforcing floors. Let X_L contain licensed portfolios satisfying $L_g(x) \geq \ell_g$ for protected language or community areas, $A_f(x) \geq a_f$ for required accessible formats, $P_j(x) = 1$ for preservation commitments, and $\sum_i c_i\, x_i + C_{\text{staff}}(x) \leq B_{max}$. Here c_i is the unit acquisition or licensing cost, $B_{\max}$ is the monetary budget, and $B(x)$ remains subject breadth. The subscript in X_L means licensed; it differs from the representation measure $L(x)$. Then

$$\max_{x \in X_L} \big[U(x) + \lambda_B B(x) + \lambda_Q Q(x) + \lambda_S S(x) - \lambda_C C(x)\big]. \tag{115}$$

Equation (115) places representation, access, preservation, licensing, and staff cost into feasibility. It leaves several choices political and professional: which floors are adequate, how preservation is designated, and how breadth is described. A Pareto set across use, breadth, representation, preservation, cost, and staff burden should accompany any chosen scalarization.

Circulation should be decomposed. A renewal, reference consultation, in-library use, digital access, program use, and archival visit mean different things. Rare use can have high value.

Nonuse can reflect invisibility or inaccessibility rather than irrelevance. A book that supports one life-changing inquiry is not well described by a low count; neither is a heavily borrowed but replaceable bestseller automatically a preservation priority.

Discovery as intervention

Search and recommendation determine which collection is encountered. A ranking model may estimate relevance from query, metadata, availability, popularity, and user context. Privacy and intellectual freedom argue against collecting more history than necessary. The system should support anonymous or minimally identified exploration, clear control of personalization, deletion, and a nonpersonalized route.

Recommendations can create a feedback loop. A high rank increases exposure, which increases recorded use and may increase rank again. Exploration can interrupt this loop, although novelty alone does not establish a useful discovery. Curated sequences, related works, local context, and staff knowledge can help readers encounter relevant material. The interface can also explain why it presents material with little previous exposure.

The interpretation of fairness depends on the collection's purpose. It may concern language access, representation of community publishing, protection from commercial concentration, or opportunity for a relevant but low-history item to be seen. Evaluate exposure conditional on relevance and collection purpose. Audit whether new or minority-language items are systematically ranked below items with richer metadata.

Uncertainty and dynamics

Demand forecasts are uncertain and endogenous. Curricula change, communities migrate, authors emerge, crises alter information needs, and licenses change. Long-term value is especially uncertain for archives and scholarship. A robust collection retains option value: adaptable space, interoperable metadata, preservation copies, vendor diversity, and cooperative networks.

Model acquisitions over a rolling horizon but protect long-lived commitments from short-term oscillation. If yearly circulation drives automatic weeding, the system can erase a subject before a community program has time to build use. Use minimum review periods, professional assessment, and reversible relocation before irreversible disposal. Scenario analysis should cover budget cuts, branch closure, platform loss, sudden demand, and disaster.

The discovery policy should reserve controlled exploration and measure long-term diversity, not only immediate clicks or loans. Yet experiments must protect privacy and avoid manipulating sensitive inquiry. Treat query logs as potentially sensitive records and restrict secondary use.

Method and computation

The collection problem combines knapsack and set-coverage structure, submodular benefits, facility allocation, and multiobjective choice. Greedy approximation can be useful when coverage exhibits diminishing returns; mixed-integer programming handles bundles, floors, and branch constraints; robust scenarios protect uncertain cost or demand; qualitative review handles significance not captured by data.

The discovery system combines information retrieval, exposure allocation, and human curation. Use offline relevance judgments, privacy-preserving aggregate logs using a specified differential-privacy mechanism where appropriate (Dwork and Roth 2014), accessibility tests, and controlled pilots. Maintain independent channels: catalog search, curated lists, browsing maps, staff assistance, and recommendation. These channels provide different representations through which a patron can discover material.

Shadow prices can reveal scarce cataloging skill or the cost of a representation floor. They do not tell the library whether the commitment is worthwhile. A high cost can justify capacity investment rather than relaxation. Chapter 4's dual variables remain model-relative.

Evidence and validation

The evidence base includes the public library's institutional mission, not only circulation data. The IFLA–UNESCO manifesto treats the library as infrastructure for education, culture, information, inclusion, and participation, while the Library Bill of Rights makes viewpoint diversity and resistance to censorship explicit constraints (International Federation of Library Associations and Institutions and UNESCO 2022; American Library Association 2019).

Use multiple evidence streams: circulation and hold queues; in-library and reference use; program participation; qualitative inquiry; gap analysis against curricula and community needs; language and format coverage; preservation condition; accessibility testing; and surveys of users and nonusers. Separate absence of demand from absence of access.

Validate the collection model through counterfactual review. Would it have retained materials later judged indispensable? Which communities and subjects lose under alternative weights? Does it recommend packages whose staff or license burden was omitted? Ask subject and community reviewers to inspect both selected and rejected items.

Validate discovery with task-based studies. Can patrons find a known item, explore an unfamiliar subject, distinguish available formats, protect privacy, understand rankings, and recover from spelling or vocabulary differences? Measure exposure and satisfaction across language, accessibility mode, branch, and query type. Automatic relevance metrics should be interpreted alongside these observations of use.

Monitor effects after implementation: concentration of loans, diversity of exposure, use of new and local materials, privacy incidents, staff workload, vendor dependence, complaints, and patterns of failed searches. A failed search can provide evidence about metadata or collection gaps.

Governance and refusal

Collection policy should be public, with defined roles for librarians, preservation specialists, community advisory groups, governing boards, and legal counsel. Staff must be able to override an algorithm with a recorded reason, and repeated overrides should trigger model review rather than staff discipline. Patrons need routes to request, challenge, correct, and appeal.

Collection policy can specify objectives and practices the institution declines to adopt. The library may refuse engagement optimization, sale of reading data, automatic removal based only on use, opaque vendor ranking, or a recommendation objective that narrows intellectual exposure. It may preserve controversial material while governing display and access according to law and professional ethics. Optimization cannot settle a challenge to intellectual freedom; it can make consequences and consistency visible.

Transfer ledger

Portfolio optimization represents budgets, complementarities, minimum provision, and opportunity costs in collection development. Librarians and communities need to supply judgments about intellectual significance and future value.

Set coverage shows whether selected items represent specified topics, languages, or formats. It does not by itself describe depth, quality, relations between works, or the politics and reasons behind the taxonomy.

A recommendation model represents relevance and exposure under its chosen features. It may omit private reasons for inquiry. If its features and acquisition rules retain inherited exclusions, it will reproduce those exclusions rather than remedy them.

Preservation engineering describes condition, redundancy, media risks, and recovery. Decisions about cultural memory also require reasons for retaining an object and the authority governing sensitive material.

Capability theory distinguishes a resource being available from a person being able to use it. Applying this distinction to library access requires information about disability, language, time, transport, safety, and trust.

We can assess the system through its contribution to informed agency, together with its effects on privacy, intellectual freedom, representation, and preservation.

Irreversibility and the archival exception

Routine collection items, local ephemera, special collections, and community archives have different reversibility. A common replacement-cost field is misleading when an object is unique, its market disappears, or its provenance depends on an intact collection. Create retention classes and require specialist or community authority before irreversible action.

Digitization may improve access while losing material features, scale, marginalia, binding, or context. Donors and communities may also restrict its use. A digital file requires format

migration, storage integrity, metadata, rights, and funding. We therefore record which properties the digital representation preserves and which access rules continue to apply.

Disaster planning should coordinate environmental control, fire and water response, priority salvage, offsite copies, vendor recovery, and staff training. The optimization objective concerns recoverable cultural function, which may differ from the number of items saved. Rarely used local records may have priority because loss is permanent.

When algorithms propose weeding or relocation, sample rejected cases and near-threshold items, audit by subject and community, and preserve an appeal period. Reversibility supplies time for evidence and public challenge.

A yearly decision record should show additions, removals, representation and accessibility floors, preservation actions, vendor concentration, staff capacity, privacy incidents, failed searches, and community requests. Publish enough to support accountability without exposing reading histories. The record lets future librarians reconstruct why a low-use item was retained or why a popular package was refused. Preserving the decision record supports later review of collection policy.

Evaluate the record against the people absent from current data: future residents, nonusers, people whose language metadata is poor, and researchers whose topic is not fashionable. Their interests cannot be estimated precisely. Representation floors, professional stewardship, public requests, and preserved option value provide institutional proxies without pretending to speak completely for them.

Case checklist

- Model collections at the level of works, series, archives, and coverage where appropriate.
- Put representation, accessibility, preservation, privacy, and licenses into feasibility.
- Treat circulation as one evidence stream, not total value.
- Audit feedback between ranking, exposure, use, and future acquisition.
- Validate with nonusers, failed searches, qualitative inquiry, and professional review.
- Preserve reversible review before discard and multiple routes to discovery.
- Publish policy, override, request, challenge, and refusal mechanisms.

Case E · A musical prosthesis co-designed as an instrument

Decision and purpose

Suppose a musician with a limb difference, injury, progressive condition, or changing mobility needs a device for performance. The team selects geometry, attachment, materials, joints, sensing, control mapping, feedback, appearance, maintenance, and fabrication. Replacing an average anatomical function may produce a mechanically adequate device that does not support the musician’s practice. We therefore begin with this person’s technique, repertoire, comfort, expressive range, stage identity, and ability to adapt the device later.

We can describe the device as a prosthesis or as an instrument interface. These descriptions overlap but emphasize different requirements. The first may emphasize clinical function and bodily safety, while the second emphasizes expressive use learned through practice. The project can also change during development. A device initially intended to imitate wrist motion may become a controller that enables different gestures. The musician determines whether that change is useful or unwanted.

We define the baseline through the musician's existing activities and alternatives. Compare alternatives with the musician's present technique, existing device if any, adapted instrument, environmental change, assistance, and a range of newly designed interfaces. Sometimes modifying the instrument or performance setup yields more benefit than adding complexity to the body-worn device.

Boundary and participants

The boundary contains the musician's body and sensation, device, instrument, repertoire, posture, stage movement, clothing, room, amplification, rehearsal and travel, fabrication, charging or consumables, repair, clinician or therapist where involved, ensemble collaborators, and audience context. It includes setup time and the consequences of failure during performance.

Participants include the musician as decision owner, prosthetist or orthotist where applicable, clinicians and therapists, mechanical and electronics engineers, instrument makers, software and control designers, materials and fabrication specialists, teachers or ensemble partners selected by the musician, and accessibility or performance staff. Each contributes evidence; none can substitute for the musician's account of embodiment and expression.

Privacy includes motion and physiological data. A stage device can also be a visible identity object. Appearance is one of the design requirements. The musician may want continuity, concealment, deliberate visibility, cultural reference, or a form that belongs to the instrument. The design process must not optimize normalization against the user's wishes.

Variables and alternatives

Mechanical variables include length, joint axes, stiffness, damping, mass distribution, socket or attachment geometry, contact pressure, range, release mechanism, and material. Sensing variables include modality, placement, sampling, filtering, calibration, and redundancy. Control variables map sensed gesture q and context c to musical or mechanical command y.

Alternatives span passive mechanical adapters, body-powered mechanisms, myoelectric or inertial sensing, switches, pressure and touch surfaces, gaze or foot control, hybrid mappings, instrument modification, and software transformation. A modular interface can support different terminal devices or mappings for practice, studio, and stage.

The feasible space must include donning, removal, cleaning, repair, transport, and operation when power or a sensor fails. A technically expressive system that requires unavailable

specialist maintenance is not feasible. Nor is a beautiful prototype whose attachment causes pain after a rehearsal rather than a laboratory trial.

Objective architecture

Let x describe hardware and π a control mapping. Evaluation can include task accuracy A, expressive coverage E, comfort K, learning burden L, fatigue or load F, reliability R_{rel}, setup time S, maintainability M, mass m, and cost C. These measures have different status.

Safety, tissue protection, structural integrity, electrical safety, secure release, and clinically required limits are constraints. The musician may define essential musical gestures or repertoire passages as coverage constraints. Within feasibility, a Pareto search can expose alternatives across expression, learning, fatigue, mass, and cost:

$$\begin{aligned} &\min_{x,\pi} \big(-E(x,\pi),\ L(x,\pi),\ F(x,\pi),\ m(x),\ C(x,\pi)\big) \\ &\text{subject to} \quad g_{\text{safety}}(x) \le 0,\ A_r(x,\pi) \ge \underline{a}_r\ \forall r \in \mathcal{R}. \end{aligned} \tag{116}$$

The repertoire set $\mathcal{R}$ is distinct from the reliability measure R_{rel} and is selected with the musician and should include demanding, ordinary, and personally important tasks. Equation (116) does not define expressive quality. E may combine reachable dynamics, timing, articulation, continuous control, gesture distinction, and qualitative evaluation. It remains an evolving model of possibility.

We retain different mechanisms during search. Combining all criteria into one score too early may select a compromise that serves no task particularly well. For example, the musician may prefer a light passive device for one repertoire and a powered interface for another. Several tools may therefore be more useful than one general-purpose device.

Control mapping and learnability

The mapping π: $(q, c) \mapsto y$ needs evaluation for controllability, observability, consistency, latency, resolution, and expressive affordance. Anatomical imitation is not automatically learnable. A nonlinear mapping can become embodied through practice when it remains predictable and supplies usable feedback.

Calibration should be quick, transparent, and recoverable. The system should show when a signal is outside its trained range or a sensor has shifted. Adaptive mapping can compensate for fatigue or placement, but it can also change a learned relationship between gesture and response. Bound adaptation, expose it, and allow the musician to freeze or restore a mapping.

Chapter 10's control theory transfers through feedback and constraints. Chapter 15's music theory adds structure: timing deviation, timbre, phrasing, and tension can be meaningful rather than error. A controller that minimizes deviation from a score may suppress precisely the gesture the device should enable.

Uncertainty and robustness

Body state, skin condition, sweat, clothing, electrode or sensor placement, fatigue, venue temperature, wireless interference, battery, and instrument setup vary. The system needs robustness across expected variation and a graceful response to faults. Design experiments should distinguish device variability from learning and from day-to-day bodily change.

Worst-case design can become heavy and restrictive. Use risk-based layers: hard limits for injury and unsafe release; robust margins for ordinary placement or load variation; detection and fallback for rare faults; and modular replacement for damage. A passive safe state may preserve limited playing rather than end a performance abruptly.

Long-term uncertainty includes changing repertoire, condition, technique, and technology. Adjustable geometry, open interfaces, replaceable sensors, documented calibration, and local fabrication preserve option value. A tightly integrated proprietary device can optimize initial performance and create dependency.

Co-design method

Begin with contextual observation and interviews, then map meaningful activities rather than generic motions. Ask the musician to identify passages, gestures, situations, discomforts, and ambitions. Create low-fidelity mechanical and interface prototypes before expensive integration. Use deliberately contrasting alternatives to reveal preferences: light versus capable, anatomical versus novel, continuous versus discrete control, visible versus discreet form.

We repeat fitting, short tasks, rehearsal, reflection, and redesign. The studio critique discussed in Chapter 15 helps identify preferences and changes to the formulation. The musician may prefer one alternative or may explain that the comparison does not address the intended activity. We record the latter response as a change to the model.

Use engineering optimization inside the cycle for stress, geometry, topology, component selection, or controller tuning. Keep interfaces parameterized so the result can return to physical prototype quickly. A finite-element optimum that changes after real contact pressure or gesture should be updated using the observed contact and gesture as evidence for the next model.

Evidence and validation

Accessible digital-instrument research emphasizes adaptability, user participation, iterative prototyping, and interdisciplinary development; musical-interface research adds low and stable latency, learnability, and long-term capacity for virtuosity (Frid 2019; Wessel and Wright 2001). These are design claims to test with the performer, not universal weights to import.

Evidence combines engineering tests, clinical or ergonomic assessment where required, task performance, learning curves, rehearsal and stage trials, maintenance trials, and first-person

report. Bench tests cover load, fatigue, impact, release, ingress, battery, electromagnetic conditions, latency, and fault response. Fit tests cover pressure, skin, range, discomfort, and duration.

Musical evaluation uses a representative task battery and open performance. Quantitative measures can include timing, error, dynamic range, repeatability, setup time, and endurance. Qualitative evidence covers agency, embodiment, sound, expressive sufficiency, attention, confidence, appearance, and interaction with collaborators. Record trade-offs rather than converting every response to a mean score.

Validation must extend beyond a demonstration. Test an entire rehearsal, travel and setup, rapid recovery from miscalibration, changing venue, and repair by the intended support network. Novelty can temporarily improve attention or mask burden. Longitudinal review distinguishes learning from fatigue and identifies changing needs.

Validation should establish support for the musician's declared activities, safety, and authorship across realistic contexts. Resemblance to a normative movement is relevant only when it serves those requirements.

Governance, ownership, and repair

The musician controls goals, acceptable appearance, data sharing, and adoption. Clinicians own clinical judgments; engineers own safety and technical claims; fabricators own material and process control. Responsibilities and warranties should be explicit at interfaces. If research is involved, participation and withdrawal must not jeopardize access to a working device.

Document parts, software, calibration, maintenance, known limits, data flow, and emergency procedures in accessible language. Define who can update firmware or mappings and how a prior version is restored. Prefer interoperable formats and locally repairable components where feasible. Secure remote features without making essential operation cloud-dependent.

Success may create a design others want. Reuse requires consent and revalidation: bodies, techniques, and purposes differ. Do not present one musician's solution as a universal optimization. We can transfer the co-design method and interface architecture; the fitted geometry and weights require a new assessment for each musician.

Transfer ledger

Biomechanical optimization describes loads, motion, contact, mass, and material behavior under modeled conditions. Device geometry also needs evaluation through sensation, identity, expressive intention, and learning.

Control theory represents state, feedback, latency, stability, constraints, and fault response in a gesture mapping. Musical use also permits intentional deviation and changes to what the performer wants to control.

Musical geometry describes selected chord and voice-leading relations (Tymoczko 2006). Repertoire coverage also depends on timbre, historical style, bodily technique, and personal meaning.

Human-centered design uses repeated evaluation in use and allows the user to revise the framing (Norman 2013; Schön 1983). We still need to examine power differences and access to fabrication and care when describing the work as co-authorship.

Modular systems engineering provides interfaces, replaceable parts, and options for later change. Repair and development also depend on the practical knowledge needed to make the device comfortable and responsive.

We can describe the transfer from prosthesis to instrument through controllable action and expanded musical possibilities. The mapping remains limited by the musician's purposes, bodily experience, and authority over the design.

Failure-mode and effects review

Walk through loss of power, sensor drift, broken attachment, software crash, wireless interference, accidental release, stuck actuator, moisture, impact, skin change, and unavailable replacement parts. For each mode, record effect on body, instrument, performance, data, and recovery; detection; safe state; and owner.

The appropriate fault response depends on the context. Releasing immediately may prevent injury but allow an instrument to fall. Holding position may protect the instrument while applying an unsafe load to the body. The musician, with the relevant clinicians and engineers, needs to define priorities for rehearsal, stage, travel, and ordinary use.

Practice fault recovery deliberately. The performer should be able to recognize the mode, switch mapping or device, continue in a limited way where safe, or stop without confusion. Ensemble collaborators can know a communication cue without receiving private health data.

Track failures and repairs longitudinally. A design that requires frequent expert adjustment may be unsuitable even if each failure is minor. Maintenance evidence should update geometry, materials, spares, documentation, and local support. Reliability is part of expressive freedom because fear of failure changes performance.

At handoff, create a performer-owned dossier: fitted geometry, safe limits, components, source and build files where permitted, mapping versions, calibration, spares, maintenance, fault actions, data permissions, and contacts. Record the musician's own evaluation and unresolved trade-offs. The dossier supports repair by a future team without converting intimate data into an unrestricted technical asset.

If the project produces a general platform, separate common modules from personal fitting and mapping. Validate the platform's interface and failure behavior, then repeat co-design for every new musician. A reusable connector can preserve mechanical compatibility; it cannot certify comfort, repertoire, identity, or control preference for another body.

Authorship and credit should record the musician's contributions to concept, testing, mapping, and performance.

Any public demonstration should use language and imagery approved by the musician.

Case checklist

- Frame the purpose with the musician, not against a normative body.
- Compare body-worn, instrument, software, environmental, and hybrid alternatives.
- Keep tissue, structural, electrical, and release safety as constraints.
- Evaluate expressive range, embodiment, learning, maintenance, and appearance.
- Bound and expose adaptive control; preserve calibration and version rollback.
- Test full rehearsals, stage faults, travel, setup, and local repair.
- Transfer the process and interfaces, not one person's fitted optimum.

Case F · A yield-aware, energy-aware semiconductor fabrication system

Decision and purpose

A semiconductor manufacturer must plan product mix, wafer starts, routes, batches, equipment qualification, preventive maintenance, dispatching, sampling, rework, and energy use across a fabrication facility. Throughput and cost per wafer measure parts of performance. The purpose is to deliver conforming devices on time with high and stable yield, protect workers and environment, maintain traceability, preserve learning, and use scarce equipment, materials, energy, and water responsibly.

A wafer can return to the same equipment class at different process layers, so the fabrication system is re-entrant. Routes can contain hundreds or thousands of operations, qualifications, queue-time limits, batch constraints, setup families, and inspections. Processing times vary, equipment fails, and recipes drift. Engineering experiments also use production tools. A dispatch rule that improves immediate utilization may therefore increase cycle time, contamination risk, or queues at a later operation.

The decisions occur over different intervals. Dispatching selects the next lot, while maintenance and qualification are planned over days. Wafer starts and product mix may be planned over weeks, and equipment purchases or facility design over years. We coordinate these decisions while retaining their different horizons and revision procedures.

Boundary and affected parties

The operational boundary includes wafer fabrication, metrology, material handling, utilities, maintenance, process engineering, quality, planning, and information systems. A lifecycle boundary includes ultrapure water, chemicals and gases, electricity, abatement, upstream materials, masks, packaging and test, equipment manufacture, waste, logistics, and the downstream energy or capability of the chips.

Affected parties include operators and technicians, process and equipment engineers, customers, suppliers, nearby communities, utility and water systems, emergency services, regulators, and future users of the technology. Safety and environmental compliance impose requirements on production. Exposure limits, incompatible materials, exhaust and abatement, emergency states, and waste handling define feasibility.

Data boundaries matter. A measurement can reveal proprietary process information; supplier and customer data may be contractually restricted. Yet hiding interfaces can prevent root-cause analysis. Governance should enable traceability with controlled access rather than create disconnected local optimizers.

Variables and alternatives

Let x_{lomt} indicate whether lot l begins operation o on tool m at time t. Planning variables s_{pt} set wafer starts for product p; q_{mt} select qualification or maintenance; e_{mt} set operating or idle-energy modes; and a_{kt} allocate engineering sampling or metrology capacity for activity class k. In Equation (117), x denotes the complete decision vector and subsumes starts, qualification, energy modes, and metrology allocation as well as lot assignments.

Alternatives include dispatch rules, batch formation, setup sequencing, qualification calendars, preventive or condition-based maintenance, sampling strategies, rework policies, buffer limits, cross-training, spare strategy, and capacity expansion. Changes to processes and equipment recipes require the experimental and approval procedures that govern their use.

The representation should preserve lot identity, route stage, recipe and reticle compatibility, contamination class, priority, due date, queue-time window, hold state, and genealogy. Aggregating all work in an equipment family can erase layer-specific constraints. Conversely, a lot-level exact model may be computationally useless for long-horizon planning. Use multiple resolutions with explicit reconciliation.

Objective architecture

Operational objectives include cycle time, on-time delivery, work in process (WIP), yield risk, setup and qualification loss, energy, water, maintenance, and schedule stability. A stylized rolling-horizon objective is

$$\min_{x} \left[\sum_l w_l\, T_l(x) + \lambda_W \sum_t W_t\,(x) + \lambda_E \sum_t E_t\,(x) + \lambda_Q \sum_l Q_l\,(x) + \lambda_\Delta \parallel x - x^{\text{prev}} \parallel_1\right] \quad (117)$$

subject to route precedence, tool and operator capacity, qualification, batching, contamination, queue-time, maintenance, safety, and due-date commitments. T_l is tardiness, W_t work in process or congestion, E_t energy, Q_l a validated yield-risk indicator, and the last term discourages schedule churn. Equation (117) coordinates operational choices; its use remains subject to critical yield limits and escalation gates described in the following paragraph.

Critical yield and safety rules belong in constraints or escalation gates. Environmental budgets can be hard annual and event-level limits. Within feasibility, Pareto comparison should show

cost, cycle time, delivery risk, energy-water burden, and learning capacity. Engineering experiments consume short-term capacity and create information that can improve future yield; their value is sequential.

Do not optimize reported yield without defining denominator and disposition. Scrap, rework, test limits, product mix, sampling, and deferred failure can change the metric. Track physical material flows and final customer quality alongside accounting yield.

Coupled physics, queues, and learning

Fabrication joins three models. The **process model** links recipes, equipment state, environment, and material response. The **flow model** links starts, routes, capacity, queues, and completion. The **learning model** links measurement and experiments to changes in process belief. Optimizing one while holding the others fixed can omit changes that affect the result.

Variable tool availability at a bottleneck amplifies queues and fabrication cycle time, especially under high utilization. Releasing more work can increase apparent activity while delaying completed output. Work-in-process limits, workload control, and bottleneck dispatching address this mechanism. Little's law supplies an aggregate consistency check; discrete-event simulation represents route and failure details.

Yield measurements are affected by the sampling policy. Engineers may send suspected problem lots for additional measurement, so those lots are not a random sample. A predictor trained on them may confuse the sampling rule with process risk. If the predictor later selects which lots to measure, it changes its own future data. We therefore retain random or designed audit samples and represent the selection process.

Uncertainty and robustness

Uncertainty includes arrivals, process time, tool failure and repair, yield drift, material variation, measurement error, demand, supplier interruption, and utility conditions. Some uncertainty is aleatory; some signals incomplete knowledge or a new failure mode. A robust schedule should retain recovery room without protecting equally against every imagined case.

Use scenario and risk analysis for correlated bottleneck failures, utility curtailment, contamination events, and demand changes. Maintain slack, qualified alternative tools, spare parts, cross-trained staff, and safe inventory for critical materials. These are resilience assets whose value is invisible in steady-state throughput.

Maintenance creates an exploration-exploitation problem. Deferring work increases capacity today and failure risk tomorrow. Fixed intervals can waste useful life; purely predictive maintenance can miss novel failures or become dependent on a fragile sensor. Combine condition evidence with engineering limits, mandatory tasks, and opportunistic maintenance during known idle periods.

Method and computation

Long-horizon planning can use linear or mixed-integer models with aggregated capacity and scenario demand. Detailed scheduling uses constraint programming, decomposition, rolling-horizon mixed-integer methods, dispatch rules, or simulation-based search. Real-time execution needs fast policies and exception handling rather than fragile exact schedules.

Digital twins combine equipment, process, and flow models, with validation for each claim. Predicting cycle time does not establish prediction of defects. Surrogates can accelerate recipe or layout search subject to uncertainty and physical constraints. Learned chip placement provides a specific example of search at a different design level: Mirhoseini and colleagues use deep reinforcement learning for macro placement (Mirhoseini et al. 2021). This result does not establish a fabrication or manufacturability guarantee.

Energy-aware scheduling can move noncritical utilities or batch steps when grid conditions improve, subject to material queue time, thermal stability, and production risk. Energy per operation and data movement inform hardware comparisons (Horowitz 2014; Hennessy and Patterson 2019). Facility energy, embodied impact, and absolute demand require separate lifecycle and usage evidence. A faster accelerator can reduce time per computation while increasing total use.

Evidence and validation

Semiconductor process control couples statistical monitoring, experimental design, yield modeling, equipment diagnosis, and run-to-run adaptation (May and Spanos 2006; Del Castillo and Montgomery 1995). Equipment availability and utilization should use a stable state vocabulary such as SEMI E10 so that suppliers, operators, and capacity models count time consistently (SEMI 2022).

Trace every model input to manufacturing execution, equipment, metrology, maintenance, utility, or laboratory records with time, units, version, and quality flags. Align clocks and lot genealogy. Missing or manually corrected records should remain visible. Recipe and software version are part of experimental identity.

Validate flow models on queue distributions, cycle-time tails, bottleneck shifts, tool utilization, holds, and recovery from downtime—not only monthly average output. Validate yield models across tools, products, layers, time, and sampling policies. Use holdout periods and prospective shadow evaluation. Investigate calibration where the decision changes, such as sampling or maintenance thresholds.

Before a scheduling policy controls production, replay historical disruptions, run discrete-event simulations, operate in advisory mode, and pilot in a bounded area with safe override. Compare against simple bottleneck and due-date rules. Monitor schedule churn and operator workarounds; they reveal constraints the model omitted.

Process optimization requires designed experiments, engineering review, material characterization, reliability testing, and change control. A simulated optimum cannot bypass

qualification. Extreme recipes or geometries are model-exploitation candidates until independent physical evidence supports them.

Governance and failure response

Separate production authority, process-change authority, safety/environmental authority, and model ownership. A scheduler can sequence approved work; it cannot approve a recipe or relax an exposure control. Operators must be able to place a lot or tool on hold. Repeated manual holds should trigger root-cause and model review.

Monitor delivery, cycle time distribution, WIP, yield and escapes, sampling bias, maintenance deferral, energy, water, chemicals, alarms, schedule churn, and distance from validated conditions. Roll back a dispatch or model update if it increases unsafe work, hidden queues, quality escapes, instability, or environmental breach. Preserve the previous rule set and compatible data.

The organization needs to retain the evidence behind model changes. Record why a model changed, what evidence supported it, which products and tools are covered, and what observations would falsify it. Avoid making a single proprietary model an unchallengeable control point.

Transfer ledger

Job-shop scheduling represents resource capacity, precedence, setup, batching, and due dates in fabrication flow. We also need process-layer requirements and contamination constraints for each route.

Queueing models relate work in process, throughput, variability, and delay. Lot history and rare holds require additional information beyond these aggregate relationships.

Statistical process control compares observations with baseline variation to detect changes. Yield analysis also needs causal investigation and checks for changes introduced by adaptive sampling.

Reliability optimization represents failure probability, downtime, spares, and lifecycle cost. Maintenance decisions also use technician observations and evidence about new failure mechanisms.

Hardware-software co-design represents dependencies among architecture, mapping, and physical implementation. Applying it to production planning also requires supply-chain, labor, and environmental consequences within the system boundary.

The fabrication system combines production decisions with continued measurement and learning. We therefore evaluate reliable output, learning capacity, and recovery alongside tool utilization.

Rare defects and controlled change

Average yield can improve while a rare reliability mechanism worsens. Sampling plans should combine risk-based metrology with designed audit samples that estimate escapes. Track defect families, spatial patterns, tool and chamber history, material lots, and downstream test. A classifier that merges rare defects into “other” may erase the signal that matters most.

Process and dispatch changes should have a change hypothesis, covered products and tools, preconditions, validation, monitoring window, and rollback. Avoid changing multiple coupled controls without a design capable of separating effects. Emergency fixes need retrospective review.

Statistical limits should account for adaptive recipes and sampling. A stable chart can reflect a controller hiding variation; an alarm can reflect a product-mix change. Link process control to physical mechanism and genealogy. When causality is uncertain, quarantine and investigate rather than optimize disposition labels.

Customer and field evidence extends the observations available from final testing, which covers only selected conditions and durations. Returns, reliability tests, and downstream integration can reveal failure modes absent from wafer metrics. Governance should prevent delivery pressure from redefining a quality escape as acceptable without proper authority.

The change dossier should link requirement, experiment, model, recipe, equipment and software versions, affected genealogy, acceptance limits, review, release, and monitoring. Preserve negative and inconclusive results so a later optimization does not repeat unsafe regions. If a schedule, maintenance policy, or yield model changes the sampling process, mark that change in the data lineage before retraining or trend comparison.

Close the economic loop as well. Expedite, scrap, rework, energy, water, maintenance, and customer escape costs can be recorded by different systems and owners. Reconcile them before declaring improvement. Otherwise an optimizer can lower the planning cost by moving burden into engineering holds, utilities, field support, or later reliability loss.

Supplier changes require the same lineage: material identity, qualification, affected products, substitution logic, and observation window. A nominally equivalent input can interact with a process margin the scheduling model never sees.

Assign an owner to audit emergency substitutions after normal supply returns. Feed the findings into supplier qualification, process limits, inventory policy, and future disruption scenarios, and retire undocumented workarounds.

Case checklist

- Separate planning, dispatch, process change, and infrastructure horizons.
- Preserve route stage, qualification, contamination, queue-time, and genealogy.
- Model process, flow, and learning loops together at their interfaces.
- Keep safety, environmental, and critical-quality rules outside ordinary trade-offs.
- Audit adaptive sampling and optimization-induced data feedback.

- Validate tails, disruptions, optimized policies, and operator workarounds.
- Maintain holds, safe rules, prior versions, and recovery capacity.

Case G · A coastal adaptation pathway under deep uncertainty

Decision and purpose

Suppose a coastal region needs to reduce flood, erosion, salinity, and storm risk. The decision also affects homes, livelihoods, ecosystems, infrastructure, public access, and cultural relationships to place. Alternatives include protection, accommodation, restoration, warning, recovery, development changes, and voluntary or regulated relocation. A sea-level estimate alone cannot determine the choice. The region needs to act under uncertainty while retaining possibilities for later change.

Maximizing protected property value privileges wealth already capitalized in land, can direct protection toward the most expensive assets, and can erase renters, informal use, heritage, ecosystems, and future mobility. The decision must protect life and critical services, distribute burdens justly, avoid preventable ecological and fiscal lock-in, and maintain legitimate choices for communities.

Different actions have different reversibility. Updating building codes and preserving setback space can keep options open. A major barrier or levee creates physical, ecological, financial, and political commitment. Retreat changes property, identity, tax base, and social networks. Waiting also acts: it permits new development, rising exposure, and loss of land needed for future adaptation.

Boundary and affected parties

The physical system includes coastline, rivers, groundwater, sediment, wetlands, dunes, drainage, waves and surge, rainfall, subsidence, buildings, roads, ports, water and wastewater, power, communications, and evacuation routes. The social system includes ownership and tenancy, insurance and finance, local government revenue, employment, schools, health, heritage, indigenous or customary rights, public access, and receiving communities.

The relevant boundary includes hydrological and institutional relationships across municipalities. An upstream defense can redirect risk; a seawall can change erosion alongshore; pumping can affect groundwater and subsidence. Protecting one tax base can shift regional infrastructure cost. Relocation creates obligations in receiving areas. A benefit-cost study confined to the project footprint is structurally incomplete.

Affected parties include residents and renters, indigenous and historically rooted communities, businesses and workers, fishers and farmers, port users, utilities, emergency services, insurers and lenders, tourists, ecosystems represented through law and stewardship, neighboring jurisdictions, and future residents. Participation must occur before alternatives and objectives are fixed.

Variables and pathways

Let a_t be the adaptation action selected at decision period t, z_t the built and ecological system state, and ξ_t observed climate and socioeconomic conditions. Actions can build or raise defenses, restore wetlands, elevate structures, upgrade drainage, protect critical facilities, change land use, buy property, support relocation, or invest in warning and recovery.

We represent the decision as a policy π mapping observations to actions over time. A pathway might restore wetlands and restrict new exposure now, elevate a critical road when nuisance flooding passes a threshold, begin voluntary acquisition under specified conditions, and construct a surge barrier only if sea level and asset configuration enter a range where it remains effective and acceptable.

Trigger design is itself a decision. A physical threshold may be observed too late because planning and construction take years. Use lead indicators and decision lead time. Triggers can include event frequency, drainage failure, insurance availability, maintenance cost, salinity, ecosystem condition, population exposure, or loss of evacuation reliability. Define measurement, review interval, responsible authority, funding source, and action lead time for each.

Objective architecture

For policy π and scenario trajectory ξ, let $L(\pi, \xi)$ include mortality and health risk, displacement, service interruption, property and infrastructure loss, ecosystem damage, and recovery burden. Let $C(\pi, \xi)$ be public and private lifecycle cost, and let $I_g(\pi, \xi)$ be residual capability for group g.

Let Ξ be the admitted scenario trajectories and P_{ref} a stated reference ensemble with explicit weights. The expectation in Equation (118) averages over this ensemble; it is not a claim to know the true joint law. Set $J(\pi, \xi) = L(\pi, \xi) + C(\pi, \xi)$ under the declared valuation and $J^*(\xi) = \min_{\pi' \in \Pi_{\mathrm{safe}}} J(\pi', \xi)$. The feasible policy set Π_{safe} requires capability floors $I_g(\pi, \xi) \geq \underline{I}_g$ for every protected group and admitted scenario, stated critical-service risk limits, state transitions $z_{t+1} = F(z_t, a_t, \xi_t)$, and actions $a_t \in A(z_t, \xi_{0:t})$. The action set enforces legal, ecological, engineering, financial, and consent requirements. Then

$$\min_{\pi \in \Pi_{\mathrm{safe}}} \left(\mathbb{E}_{\xi}[L(\pi, \xi) + C(\pi, \xi)] + \lambda \max_{\xi \in \Xi} [J(\pi, \xi) - J^*(\xi)] \right). \tag{118}$$

The regret term compares each pathway with the best policy chosen with hindsight for the same scenario and feasible policy class. Its worst-case comparison is distribution-free over the admitted scenarios. The reference-ensemble mean remains sensitive to its weights, which should be varied. The set of admissible actions is one part of the complete feasible policy set.

Not every value belongs in L. Human life, cultural continuity, public access, and ecosystem integrity should not be monetized without showing the ethical and legal assumptions. Use a dashboard and Pareto comparison. Where rights or irreversible thresholds apply, use

constraints and vetoes. Planetary and ecological boundaries require separate consideration from ordinary preferences (Rockström et al. 2009).

Deep uncertainty and option value

Uncertainty spans sea-level pathways, ice-sheet response, storms, rainfall, subsidence, erosion, ecosystems, population, development, technology, insurance, migration, and political capacity (Intergovernmental Panel on Climate Change 2021; Intergovernmental Panel on Climate Change 2022a). A precise joint distribution may imply more knowledge than the evidence supports. Scenario discovery identifies combinations that cause a policy to fail (Bryant and Lempert 2010); robust decision making compares policies across plausible futures (Lempert, Popper, and Bankes 2003).

Option value comes from actions that preserve future alternatives: setback land, modular defenses, adaptable buildings, transferable finance, open evacuation routes, ecosystem migration corridors, and institutions capable of staged relocation. These assets can look inefficient in a most-likely scenario. Their value appears when the future or public preference changes.

The evidence required depends partly on the reversibility of the action. A reversible pilot can proceed with uncertainty and monitoring. A barrier that transforms an estuary or a relocation program that dissolves a community requires deeper evidence, consent, compensation, and alternatives. Waiting for greater certainty also has consequences when exposure and commitments continue to grow.

Coupled engineering and social mechanisms

Hydrodynamic and geomorphic models estimate inundation, waves, erosion, and sediment response. Structural and geotechnical models assess defenses and foundations. Network models assess evacuation and critical-service access. Ecological models assess habitat and protective function. Economic models estimate direct and indirect losses. Social research examines mobility, trust, heritage, tenure, and the ability to use programs.

We need to examine the mappings between these models. A flood-depth map becomes a damage estimate through inventories and vulnerability curves. Its effect on a household decision also depends on finance, information, tenure, health, and attachment to place. Each mapping may omit information. We retain intermediate results and uncertainty so that the final optimization is not based solely on an expected monetary loss.

Nature-based measures illustrate coupling. Wetland or dune restoration can reduce waves, create habitat, and provide recreation, but performance depends on sediment, space, maintenance, and event conditions. A model should not treat “one hectare restored” as a fixed risk reduction. Nor should uncertainty be used to dismiss ecological co-benefits while engineered protection is modeled with false precision.

Method and computation

Use exploratory modeling over a wide scenario ensemble rather than one forecast. Simulate candidate policies through time, including triggers and construction lead. Compute expected loss, tail loss, regret, critical-service failure, distributional outcomes, ecosystem indicators, fiscal exposure, and stranded investment. Identify scenarios where each policy violates requirements.

Dynamic programming can solve stylized staged choices; mixed-integer models handle project timing and network investment; robust optimization handles bounded uncertainty; real-options analysis clarifies the value of waiting and flexibility; multiobjective visualization supports deliberation. The decision process then reviews these computational results. A small number of distinct pathways should be brought to communities and decision bodies with assumptions and failure regions visible.

Stress tests should include compound flooding, defense failure, evacuation during power loss, repeated events before recovery, insurance withdrawal, funding delay, population change, and neighboring adaptation. Physical robustness therefore needs to be assessed together with institutional feasibility.

Evidence and validation

Dynamic adaptive policy pathways formalize staged actions, monitoring signposts, and transfer points across many plausible futures (Haasnoot et al. 2013; Kwakkel, Haasnoot, and Walker 2015). Coastal portfolios must also confront the distributional and institutional reality of protection, accommodation, and retreat; managed retreat is a risk response with governance and justice requirements, not merely a location variable (Hino, Field, and Mach 2017; Intergovernmental Panel on Climate Change 2022a).

Validate hazard models against tide gauges, storm records, high-water marks, topography, subsidence observations, and event hindcasts, while acknowledging nonstationarity. Validate drainage, building, infrastructure, and ecosystem components at the scale of the decision. Compare model ensembles and explain structural differences instead of averaging them away.

Validate social and distributional claims with property and tenancy data, accessibility analysis, interviews, participatory mapping, historical records, and after-action evidence. Official property data can omit informal residence, cultural use, or people without secure tenure. Community knowledge can identify routes, gathering places, and failure mechanisms missing from maps.

Pilot warnings, shelter routes, retrofit delivery, and voluntary programs. Monitor uptake and reasons for nonparticipation. A low uptake rate may reveal distrust, paperwork, inaccessible finance, landlord control, or an unacceptable offer. These mechanisms should be examined before interpreting the response.

After events, reconcile forecast extent, infrastructure performance, communication, evacuation, service loss, recovery time, displacement, and ecological effects. Update the

pathway only through a documented review that distinguishes new evidence from changed values or budgets.

Governance, justice, and failure response

Create a regional body with municipal, infrastructure, emergency, housing, environmental, indigenous or customary, community, labor, and scientific representation. Define decision rights, not merely consultation. Some actions require local consent or statutory process. Publish scenarios, maps, model limits, property and distributional effects, and the basis for thresholds.

The pathway requires funding for the actions associated with its triggers. Create mechanisms for maintenance, emergency action, compensation, relocation support, and receiving-community investment. Protect renters and people without clear title. Avoid buyout designs that reward only property value or produce patchwork abandonment.

Monitoring includes hazard, defense condition, ecosystem state, critical services, exposure, insurance and housing cost, displacement, participation, public access, and trust. Trigger review if impacts shift to another area, maintenance fails, ecological thresholds are crossed, or protected groups fall below the capability floor. Preserve emergency alternatives if the preferred long-term project is delayed.

Transfer ledger

Robust control contributes state observation, constraints, feedback, and policy revision to an adaptation pathway. Authority, finance, and consent determine whether an institution can take the action associated with a trigger.

Real-options analysis describes the value of waiting, staging, and reversibility in infrastructure decisions. We also need to consider attachments that have no market price and differences in who can safely wait.

Reliability engineering describes failure domains, redundancy, critical services, recovery, and common causes. Regional resilience also depends on histories of neglect and unequal recovery capacity.

Ecosystem models describe material flows and protective functions under specified conditions. Decisions about nature-based protection also require cultural relationships, intrinsic value, and rules governing land and water.

Welfare analysis compares stated consequences under a valuation rule. Rights, procedural justice, and meanings that do not permit substitution require separate treatment.

A pathway includes continuing observation, deliberation, action, and revision. Each review should distinguish uncertainty in the evidence from decisions about acceptable risk and the timing of action.

Relocation as a complete service system

Relocation involves information, appraisal, legal help, finance, housing search, employment, school, health and social care, cultural continuity, moving, site restoration, and integration in receiving communities. Offers must be timely enough that people can choose before repeated loss removes choice.

Eligibility rules should protect renters, informal occupants, shared and inherited ownership, and people with accessibility needs. Compensation based only on market value can be insufficient to obtain equivalent safe housing and can ignore business, network, and cultural loss. Monitor who accepts, who cannot, and who remains in a fragmented service area.

Voluntary programs can become coercive if services, insurance, or maintenance are withdrawn before a viable alternative exists. Publish service-transition rules and preserve emergency response. Community-scale or group relocation options may retain networks better than isolated buyouts, but require land and receiving-area participation.

After acquisition, govern demolition, contamination, land stewardship, public access, and ecological restoration. A checkerboard of empty parcels can increase hazard and maintenance burden. The optimization should include the future landscape and fiscal system, not end at transaction count.

Publish a decision register at each pathway review, recording new observations, trigger status, model and scenario changes, completed and delayed projects, maintenance, group and ecological outcomes, available funds, and disagreement. It also identifies the next reversible and irreversible actions. This allows us to check whether the pathway still has the assumptions, authority, and resources needed for implementation.

Case checklist

- Model a policy with triggers, lead times, and funding—not one terminal project.
- Expand the boundary along water, ecosystems, infrastructure, and displacement.
- Keep safety, rights, cultural authority, and ecological thresholds visible.
- Test expected loss, tail loss, regret, critical services, and group capabilities.
- Preserve option value through setbacks, modularity, corridors, and institutions.
- Validate physical and social mappings separately and retain intermediate uncertainty.
- Monitor distribution, maintenance, ecosystem condition, trust, and receiving areas.

Comparative synthesis: portfolios, identity, and irreversibility

The four cases share a portfolio structure. A library allocates money, space, attention, and exposure. A musician selects a family of mechanical and control possibilities. A fab allocates capacity, maintenance, experiments, and environmental resources. A coast selects staged interventions and preserved options. Portfolio mathematics reveals coupling and opportunity cost in each case, while the purpose comes from the domain and its decision process.

Their objects differ in identity. A book can be replaceable as a commodity and unique as part of an archive. A fitted device is inseparable from one musician's body and practice. A wafer lot has genealogy through repeated operations. A coastal place contains ecological and social relations not captured by parcels. The representation needs to distinguish changes that preserve the relevant identity from changes that remove it.

Irreversibility also takes four forms. A library can irreversibly discard cultural evidence. A device can create bodily harm or dependency but is often physically modifiable. A process change can propagate latent defects into many products. Coastal construction or relocation can transform ecosystems and communities for generations. Evidence and participation should become stronger as reversibility decreases.

Each case contains an endogenous measurement loop. Library exposure creates circulation. Device adaptation changes technique and sensor data. Fab sampling follows suspected risk. Coastal investment changes development and insurance. Historical data therefore records effects of prior policy as well as demand. Exploration, audit samples, and qualitative inquiry prevent the loop from confirming itself.

We can interpret the transfer ledgers using a category-theoretic analogy. A specified mapping may preserve coverage, feedback, flow, interface, or option relationships while omitting cultural meaning, embodiment, craft knowledge, or institutional history. A formal claim about preservation requires defined objects and morphisms. In the broader examples, the ledger identifies the omissions that require domain evidence and review.

Four interface patterns

Across the cases, four interface patterns recur.

The coverage interface appears in the library and the coast. A collection covers subjects, languages, formats, and memories; an adaptation portfolio covers hazards, places, groups, and time. Set coverage can show gaps, but the definition of an element and adequate coverage is local. One copy does not provide access if format or discovery fails; one project does not provide protection if maintenance or service transition fails.

The embodiment interface appears in the prosthesis and, less visibly, the library. A control mapping becomes usable through a musician's learning and sensation. A discovery system becomes useful through a patron's language, attention, privacy, and ability to reach material. An abstract input-output model preserves command or relevance and forgets bodily and social conditions. Co-design restores some of that context.

The genealogy interface appears in fabrication and archives; both require an object's history. For a wafer, this includes routes, tools, recipes, materials, and measurements. For an archival object, it includes provenance, order, custody, and relations to other objects. Aggregation may simplify planning while removing information needed for quality or interpretation. We therefore retain this history in the evidence even when the optimizer uses a coarser representation.

The pathway interface appears in coastal adaptation and device evolution. Present choices change future feasible sets. A barrier, discarded archive, proprietary component, or frozen interface can close options. Staged commitment, modularity, reversibility, and trigger rules preserve option value. These design qualities preserve capacity to learn and change, and affect expected cost.

These patterns are candidates for reuse in a domain not covered here. Before importing a metric, we ask: what constitutes coverage, embodiment, genealogy, and pathway in the new setting, and who can validate the answer?

Negative transfer tests

We can test a cross-disciplinary analogy by constructing cases in which the imported model may give an inappropriate recommendation.

Library as inventory. A stock-turn rule may recommend removing rarely used local history. This indicates that the retail model omitted preservation and future users. Retention categories, professional review, and archival provenance provide additional requirements for the library application.

Prosthesis as robot manipulator. A trajectory-accuracy model may recommend a device that the musician finds painful or unfamiliar in an unwanted way. We then need to include bodily experience and authorship through safety constraints, co-design, and expressive evaluation.

Fab as generic job shop. A schedule may improve utilization or due dates while sending an incompatible layer through a tool or deferring measurements needed for yield. We need to add route types, qualification, and controlled sampling to the generic scheduling model.

Coast as asset portfolio. Minimizing expected property loss may direct protection toward expensive assets and represent relocation only as a transaction. We need additional constraints and evidence about rights, communities, ecosystems, and receiving places. Capability floors, consent, and complete service arrangements change the feasible alternatives.

The test generalizes. Construct a candidate optimum under the imported model. Ask a domain expert and an affected participant to find a case that is numerically good and substantively wrong. Identify the missing relation, then decide whether it belongs in representation, feasibility, evidence, governance, or refusal. Repeat until remaining failures are understood and monitored.

The resulting transfer ledger records a counterexample, the omitted relation, and the design change made in response. Remaining limitations need an explicit scope and a monitoring plan.

Field checklist

- Protect rights, mission, cultural memory, and expressive agency before optimizing use or throughput.

- Co-design accessible instruments for long-term musical control, repair, authorship, and changing capability.
- Separate semiconductor yield learning from uncontrolled process change and use stable equipment-state definitions.
- Build coastal pathways around signposts, option value, finance, consent, and complete relocation services.
- Test irreversible choices against distribution, identity, maintenance, and future feasible alternatives.
- Preserve the differences among knowledge, creativity, fabrication, and adaptation when translating methods.

Translation lens for Chapter 19. Libraries, musical prostheses, fabrication systems, and coastal regions use portfolios, feedback, interfaces, and option value. Each transfer requires additional interpretation of discovery, embodiment, yield, or place. These meanings determine whether the imported model supports the domain's purpose.

Chapter 20 · Future frontiers, stopping rules, refusal, and synthesis

Computational and institutional frontiers

Optimization methods can participate in the design of models, experiments, machines, institutions, and explanations. We can examine several areas where these activities interact with the search procedure.

Automated modeling and scientific discovery

Machine learning can propose hypotheses, approximate expensive simulators, infer differential equations, select experiments, and search for molecules or materials. Physics-informed neural networks include differential-equation residuals in the learning objective (Raissi, Perdikaris, and Karniadakis 2019). We also need to examine identifiability, extrapolation, uncertainty, and the evidence for a proposed mechanism.

Automated machine learning and neural architecture search optimize pipelines that themselves optimize models (Hutter, Kotthoff, and Vanschoren 2019; Elsken, Metzen, and Hutter 2019). These nested loops amplify data leakage, benchmark overfitting, and compute inequality. Reproducible evaluation and resource accounting must scale with automation.

Differentiable systems and inverse design

Automatic differentiation makes gradients available through simulators, renderers, physics solvers, and programs. Inverse design can optimize shapes, materials, optics, controls, and manufacturing jointly. Discrete choices and discontinuities remain difficult; surrogate gradients can be useful without being faithful. The resulting gradient describes the computational graph; its relevance to a physical intervention depends on the model.

A derivative needs a provenance path: which variables were fixed, which discontinuities were smoothed, which constraints were relaxed, and which measurements calibrate the model. Local sensitivity becomes a system claim only after the interfaces are validated.

Composable and distributed optimization

A large system may need to divide computation across organizations or data-access boundaries. Distributed optimization, federated learning, market mechanisms, and category-theoretic interfaces provide different ways to coordinate local work. We then need to establish which guarantees about the components remain valid for the combined system.

Open games model strategic components with explicit inputs, outputs, observations, and continuations (Ghani et al. 2018). Decorated cospans model systems assembled along interfaces (Fong 2015). These approaches suggest software-like libraries of optimization components, while context dependence limits naive reuse.

Quantum and hybrid optimization

Quantum algorithms investigate new representations of search, sampling, linear algebra, and energy minimization. The quantum approximate optimization algorithm alternates problem and mixing operators in a parameterized circuit (Farhi, Goldstone, and Gutmann 2014). Near-term devices face noise, limited depth, and optimization landscapes with barren plateaus (Preskill 2018; McClean et al. 2018; Cerezo et al. 2021). Claims of advantage require end-to-end comparisons including encoding, error mitigation, classical optimization, and hardware cost.

The same end-to-end comparison applies to specialized accelerators, distributed computing, and learned surrogates. A speedup for an inner kernel need not improve the complete decision process.

AI agents and institutional optimization

Agents that plan, call tools, write code, negotiate, or operate infrastructure create long-horizon optimization with partial observability. Capability must be paired with permissions, sandboxes, audit trails, reversibility, and human authority. Evaluations should include deception, specification gaming, correlated failure, and recovery—not just task completion.

At institutional scale, algorithmic systems may allocate work, credit, care, or attention. Democratic oversight and legal contestability must evolve as quickly as technical capability. These applications therefore raise questions about who can define, review, and revise the objectives being optimized.

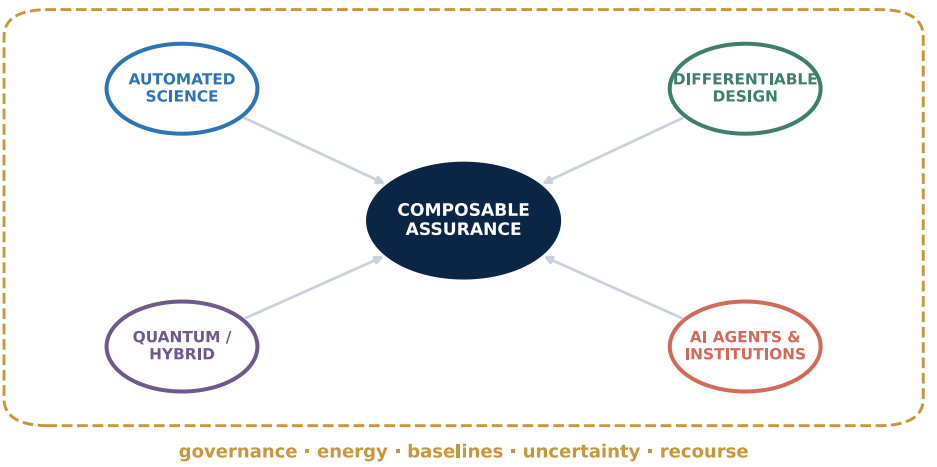


Figure 29. Automated science, differentiable design, quantum or hybrid methods, and AI agents and institutions feed composable assurance.

Practice note. Compare the complete decision process with a baseline, including data, interfaces, energy, validation, human work, and residual risk.

Figure 29 connects development with its validation requirements. Automated science needs independent experiments; differentiable design needs model provenance; compositional systems need interface contracts; quantum methods need end-to-end baselines; institutional optimization needs legitimacy and repair.

Larger search spaces, richer derivative information, distributed decisions, and faster updates can extend optimization in science, design, infrastructure, and care. They also create more dependencies between a local objective and its effects. The surrounding institution needs ways to specify those dependencies, observe their consequences, challenge the formulation, and stop the process when required.

An automated modeling system can choose features, architectures, experiments, tools, and evaluation criteria. These choices affect the evidence later used to assess it. For example, experiments proposed by a model may become its next training data. An agent that writes both code and tests may repeat the same omission in each. We therefore retain independent baselines and holdout evidence outside these selection loops.

Compositional assurance

Compositional assurance asks when local reasoning supports a guarantee for the assembled system. The categorical constructions above specify interfaces and composition; engineering evidence must establish their application conditions.

Connecting components introduces conditions absent from their separate evaluations. Local feasible sets may be incompatible. Two stable controllers may interact unstably, and individually fair allocations may have an unfair combined effect. Repeated use changes a privacy budget. A policy may also change another component's input distribution. We therefore state the composition rule together with the conditions required for its guarantee.

For each interface, record:

- semantic type and units of inputs and outputs;
- timing, ordering, and freshness assumptions;
- uncertainty and error contract;
- resource and privacy budget consumed;
- safety and security invariants preserved;
- authority to reject, override, or degrade;
- observations available for detecting contract failure.

A system-level guarantee requires the relevant interface assumptions to hold together. Category-theoretic models can help state which compositions are defined and which properties they preserve. Establishing a guarantee still requires the corresponding mathematical result and evidence that the application satisfies its assumptions. A failed compatibility check identifies a condition to investigate.

Optimization charters and constitutions

Before building a model, write a one-page optimization charter with eight fields.

Purpose. What change is sought, and why?

Decision. Who can choose what, and on what cadence?

Boundary. Which lifecycle, feedback, and affected groups are included?

Values. Which outcomes are objectives, constraints, safeguards, or deliberative questions?

Evidence. What data and causal assumptions support the model?

Uncertainty. Which unknowns, shifts, and adversaries matter?

Authority. Who approves, operates, contests, audits, and retires it?

Exit. What triggers rollback, redesign, or abandonment?

Unanswered questions in the charter identify work needed before the model can support the proposed decision.

For an autonomous system, the requirements recorded in the charter form an optimization constitution. It distinguishes alternatives the optimizer may search from actions the institution may authorize. A useful constitution contains seven clauses.

1. **Purpose.** Name the human, social, scientific, or ecological capability the system serves and the baseline it seeks to improve.
2. **Jurisdiction.** Define the people, assets, places, time horizons, and externalities inside the decision boundary.
3. **Rights and limits.** State legal, safety, ethical, environmental, and procedural requirements that cannot be traded by a weight.
4. **Evidence.** Define which claims require simulation, experiment, professional review, participation, or monitored deployment.
5. **Authority.** Assign who can approve objectives, change constraints, deploy, override, investigate, and stop.
6. **Contestability.** Give affected parties understandable notice, correction, explanation, and appeal appropriate to the stakes.
7. **Sunset and repair.** Specify review dates, rollback assets, incident response, retirement, and restoration of capabilities the system may displace.

These clauses specify the responsibilities at institutional interfaces. A regulator, operator, model owner, community body, or user can hold different authority. Oversight is described through named acts and response times.

Stopping and refusal

Stopping search

Stop computational search when the expected value of another iteration is less than its resource and delay cost, or when solver uncertainty is already small relative to model and preference uncertainty. A simple decision rule is

$$\text{continue search only if} \quad \mathbb{E}[\Delta V_{n+1} \mid \mathcal{I}_n] > C_{\text{compute}} + C_{\text{delay}} + C_{\text{risk}}, \tag{119}$$

where $\mathcal{I}_n$ is current information and ΔV_{n+1} is improvement relevant to the decision. A lower numerical objective need not provide such an improvement. Equation (119) identifies quantities to compare, but it remains qualitative until those quantities are estimated.

Practical stopping indicators include stable decisions across plausible models, a certified bound adequate for the choice, repeated alternatives within measurement error, exhausted evaluation budget, or a finding that new computation cannot resolve the contested value. Report the best candidate, bound or gap, sensitivity, and reason for stopping.

Stopping deployment

Stop or roll back deployment when a safety invariant fails, operation leaves the validated region, data or models are compromised, a protected group crosses its threshold, operators cannot understand or safely manage recommendations, strategic responses invalidate the assumed mechanism, or monitoring is insufficient to support continued operation. Each condition needs an owner and a response procedure.

Triggers combine quantitative thresholds and authorized qualitative judgment. A rare serious incident can require action before its rate is estimated precisely. Repeated justified overrides or community complaints can reveal failure before the primary metric changes. Test restoration time and the service available after an emergency stop. An off switch without a functioning safe service is not a safe fallback.

Updates require proportional revalidation. A display change may alter human action even if the model is unchanged. A new data source may change privacy and selection. A solver or hardware change can affect numerics. A new population or jurisdiction can invalidate evidence. Version the complete decision system, including model weights and these dependencies.

Stopping the project

Refusal is appropriate when the institution lacks a legitimate objective, the necessary decision is outside its authority, harms cannot be made acceptable, evidence cannot support the claim, affected people cannot meaningfully contest the system, measurement would create unacceptable surveillance, the project would destroy a capability essential for safe fallback, or irreversibility and uncertainty make preservation of future options paramount. Some decisions should remain deliberative, professional, personal, or political even when analysis informs them.

A decision to refuse the proposed optimization may redirect work toward measurement, service design, capacity, rights protection, interface repair, or a less centralized tool. A hospital may refuse automatic discharge optimization and build better care-transition evidence. A school may refuse a single teacher score and improve observation and support. A cultural institution may refuse personalized surveillance and invest in contextual discovery.

Satisficing is often the responsible alternative (Simon 1955, 1956). Find a feasible action that meets safety, rights, service, and evidence thresholds, then preserve slack and learning capacity. Further numerical improvement should be compared with its effects on these capacities.

Refusal can also protect activities whose value depends on freedom from ranking or instrumental control. Care, friendship, play, mourning, citizenship, scholarship, and art can be supported by limited optimization while being damaged by its unrestricted extension.

Maturity and operational authority

Assess maturity across formulation, evidence, computation, integration, governance, and repair. A system may be advanced in one and underdeveloped in another.

Exploration generates hypotheses or alternatives; its outputs are not operational authority. **Advisory use** supplies validated outputs to a named human process with baselines and recorded overrides. **Bounded automation** permits action within verified constraints, with monitoring and fallback. **Adaptive operation** adds change control and lifecycle evidence for updates. **Institutional maturity** includes purpose, distributional effects, appeals, incident learning, and retirement.

Do not promote a system because one metric improves. Require the evidence and organizational capability of the next level. High-stakes projects can deliberately remain advisory. The appropriate degree of automation is a design decision.

Maturity can fall when staff, data, suppliers, law, or context changes. Review the operating institution as well as the model.

Further translation · Archaeological fieldwork

Archaeological fieldwork is a useful stress test because search can destroy its own evidence. A team must choose survey area, excavation units, depth and sequence, sampling, conservation, documentation, laboratory analysis, and allocation of limited time and expertise. The objective is to produce defensible knowledge while preserving context, respecting law and community authority, protecting people and material, and leaving future researchers meaningful options.

The boundary includes landscape, site, stratigraphy, artifacts and ecofacts, remote sensing, archives, laboratory capacity, conservation, land ownership, descendant and indigenous communities, permitting, funding, weather, security, curation, and publication. A find is valuable partly through relation: layer, association, orientation, material, absence, and excavation history. Removing an object without context can increase count and destroy information.

Decisions are sequential and partially observed. Noninvasive survey and test pits update beliefs about site structure. Excavation reveals and irreversibly changes the state. A policy can choose the next action based on observations, but the action determines what future

evidence remains. This resembles Bayesian experimental design and adaptive control, with an unusual state constraint: some information-bearing structures cannot be restored after observation.

A model could compare expected information gain, research coverage, preservation, community priorities, and cost. Its constraints would include permits, sensitive material, human remains, safety, conservation capacity, recording standards, and agreed authority. Information gain depends on a particular set of hypotheses and observations. Reducing posterior entropy distinguishes among those hypotheses; it does not fully describe interpretive value or community meaning.

Use a portfolio of methods: archival research, pedestrian and geophysical survey, environmental sampling, limited excavation, conservation, and analysis. Optimize the sequence so results from less destructive methods inform later action. Reserve unexcavated areas and samples for future techniques. Value of information and option value from Chapter 6 become material stewardship.

Evidence validation includes calibration and ground truth for survey instruments, recording consistency, chain of custody, stratigraphic review, independent interpretation, laboratory controls, and reconciliation with historical, linguistic, and community knowledge. Negative evidence must be interpreted against sampling and preservation. Publication should include methods, uncertainty, data and access rules, and competing readings.

Communities may have legal, cultural, or sovereign authority over sites, access, interpretation, display, repatriation, and data. These requirements enter the decision before excavation. A technically promising proposal may still be refused, and safety or conservation teams may need to stop work. We record why material was removed, what remains, and who can authorize later use.

Transfer ledger

Experimental design represents sequential information and sampling efficiency, while the archaeological application also needs the destructive and culturally governed nature of observation. Optimal stopping compares further information with cost, but duties to leave material undisturbed require separate treatment. Information theory distinguishes modeled hypotheses without representing all possible meanings. Portfolio methods describe budgets and complementary methods, but excavation may also change authority and relationships to place.

In this example, the formulation can support a smaller search, better documentation, preservation of future options, or a decision against excavation. The preferred action depends on the relation between information gained and context lost.

Further translation · Language revitalization

A language community may allocate resources among teaching, family transmission, speaker support, documentation, publishing, media, terminology, events, digital tools, archives, and institutional recognition. A platform might propose maximizing enrolled learners, daily active users, test scores, or corpus size. Those measures can support operations and cannot define revitalization.

The purpose belongs to the community and may include intergenerational use, cultural continuity, sovereignty, expressive range, public presence, speaker well-being, and domains in which the language can live. The boundary includes fluent and heritage speakers, learners, families, teachers, elders, artists, schools, government, technology platforms, archives, dialects, orthographies, land and ceremony, and historical suppression. Data collection and publication can expose knowledge the community considers restricted.

The interventions can change later conditions. Training teachers increases teaching capacity, media may encourage use, and terminology may enable new activities. A dictionary preserves forms but can also establish one standard at the expense of others. Speech tools can improve access and can appropriate voices without consent. Recommendation may increase exposure while favoring short, standardized content. We therefore examine how each intervention affects future language use and data.

A portfolio model can track reach, proficiency, speaker support, domain coverage, accessibility, cost, and resilience. Hard constraints should encode community data governance, consent, restricted knowledge, fair compensation, dialect inclusion, and the authority to approve uses. Minimum investment in fluent speakers or family transmission may protect the source of learning rather than maximize learner count.

Uncertainty concerns participation, institutional continuity, migration, technology, funding, and the effect of interventions. Adaptive pathways can build immediate speaker and teacher capacity, pilot programs, monitor real use, and expand only with community approval. Preserve multiple delivery modes so dependence on one platform does not create lock-in.

Evidence should combine participation and learning data with observed use, speaker and family accounts, domain presence, material quality, teacher burden, and cultural assessment. A standardized test can measure selected competence and miss confidence, identity, interaction, or use with elders. Corpus size can grow through duplicated or synthetic material without increasing living language. Qualitative evidence helps evaluate these aspects of language use.

Governance assigns the community decision rights over objectives, data, models, publication, access, and commercial use. The contributions of technologists and funders do not transfer these decision rights. Speakers should be compensated and able to withdraw particular recordings where agreements permit. Systems need provenance down to recording, speaker permission, transformation, and downstream model.

Transfer ledger

Educational optimization represents resources, sequence, and learning support while omitting some relationships and identity involved in language transmission. Information retrieval finds documented material under cultural protocols and oral context. Machine learning represents statistical patterns but forgets authority and can standardize away variation even when that is not the stated objective. Ecological resilience contributes diversity, redundancy, and institutional-dependence concepts; biological fitness supplies no ranking of people or languages.

The recommended policy can therefore prioritize speaker time, teacher formation, community-controlled archives, family contexts, creative production, and open but governed technical infrastructure over the metric with the largest user count. The formulation identifies resource relationships that optimization can coordinate while retaining community authority over the problem's purpose and scope.

Retrospective stress tests: running the ledger backward

The seven worked translations are prospective. These two cases instead run the seven questions backward from an observed failure or accumulated production evidence, recording which questions would and would not have warned. Because their sources already helped shape the grammar, they are consistency checks rather than out-of-sample validation.

Retrospective case R1 · Rankings and reactive institutions

Espeland and Sauder studied how published law-school rankings changed the institutions being ranked. Schools redirected attention, managed reported categories, and changed admissions and organizational practices in response to a common public measure. The study concerns professional schools and their rankings. Commensuration makes heterogeneous institutions comparable; reactivity describes their response when an external measure becomes an internal target (Espeland and Sauder 2007).

Run the seven questions backward. Q3, what is better, flags the conversion of plural educational purposes into an ordinal score. Q4 flags endogenous adaptation and self-fulfilling feedback. Q6 flags the need to validate institutional consequences, not only ranking reproducibility. Q7 flags authority, burden, and the absence of recourse against a privately produced ordering. Q1 and Q5 warn only if the boundary includes schools' strategic responses rather than treating rank as passive information. Q2 warns only if admissibility encodes mission, rights, and protected exclusions rather than merely data eligibility.

Backward transfer ledger. Commensuration makes heterogeneous schools comparable under a common ordering. The measure may omit institutional mission, unmeasured educational quality, local context, and the effects of publication on behavior. Repair requires plural evidence and institutional recourse rather than a more elaborate single score. We retain several evidence sources, examine manipulation and redistribution, state uncertainty and category choices, and assign authority to challenge or discontinue the measure.

The counterexample changes the design rule. A ranking should be evaluated as an intervention in an institution, with predeclared reactivity scenarios and a stopping condition if adaptation degrades the underlying purpose. The field guide would not have predicted every response; it would have required the response channel to exist in the model and governance plan.

Retrospective case R2 · Hidden technical debt in machine-learning systems

Sculley and colleagues synthesized production experience across machine-learning systems. Their boundary-erosion mechanisms include entanglement, correction cascades, and undeclared consumers. Data dependencies, feedback loops, configuration debt, and external changes are additional technical-debt categories. Local model improvements can increase maintenance risk at system level (Sculley et al. 2015).

Run the questions backward. Q1 flags that the decision object is the production system, not model parameters alone. Q4 flags feedback and changing data dependencies. Q6 flags system-level validation, monitoring, and recovery. Q7 flags ownership of dependencies and changes. Q3 warns if maintenance, failure, and operator burden are part of what counts as better; without them it can bless a locally improved model. Q2 and Q5 alone are insufficient: a feasible training run and a strong search procedure do not expose hidden consumers or coupled failure modes.

Backward transfer ledger. Component optimization represents local loss, resources, and visible interface assumptions. It may omit dependencies, data history, other consumers, feedback, and the cost of safe changes. Interface contracts, data and service specifications, provenance, dependency records, shadow deployment, rollback, and comparisons of the complete system help address these omissions. Later Google production practice applied related checks across several dozen teams through tests of data, models, pipelines, canaries, monitoring, and rollback (Breck et al. 2017).

The retrospective cases also show a limitation of the seven questions. Several identify a failure only when the boundary includes the relevant dependency and the formulation recognizes maintenance as a value. The questions support inquiry; the choice of representation still requires domain knowledge and evidence.

Contributions from humanities and arts

Humanities and arts can contribute methods to optimization as well as application examples. Their procedures help identify changes in the object, frame, or evaluation criterion during interpretation and use. We can examine these contributions as transfers in the reverse direction.

We can use critique to examine a proposed preference model. A counter-reading asks which assumptions support the selected candidate, whose purposes are taken for granted, and which alternative challenges the ranking. In retrospective case R1, interpreting a ranking as an intervention reveals feedback omitted from a passive measurement model. Such readings can

identify missing variables, constraints, and authority. Reflective practice uses the unexpected result to reconsider the formulation (Schön 1983; Eco 1990).

Framing-loop analysis. This method asks whether search helps produce the objective it later claims to satisfy. A ranking creates attention; a recommendation system creates exposure; a design prototype teaches users what to want; a public narrative changes which futures appear possible. This is endogeneity of the objective. In such systems the map from preference to outcome loops back into preference formation. We may therefore need to monitor the objective as a changing state of the system.

Plural reading. Several interpretations may remain supported by the evidence and purpose. We retain their sources, differences, and limits, together with the procedure for choosing when an action is required. This resembles a set-valued argmin. However, the analogy needs to retain the reasons and authority associated with each interpretation, as well as membership in the set.

Narrative and historical analysis examine sequence, voice, genre, and memory (Genette 1980; Moretti 2005). We infer a modeling consequence: two systems with the same encoded current state can carry different obligations because their histories involve dispossession, repair, consent, or promise. That state representation is inadequate when it omits history constitutive of the decision. A richer state or a different model is then needed.

Reverse transfer ledger. Critical interpretation can expose omitted premises and support challenge, but it needs a connection to operational decisions. Framing-loop analysis requires observations that could contradict it. Plural reading can remain useful, provided the institution specifies how disagreement affects action. Narrative history retains sequence and voice, while alternative readings and archives help test dependence on one account.

The practical implication is a richer optimization workflow. Before fixing an objective, commission a counter-reading. During pilots, observe whether the intervention changes the frame. At evaluation, preserve justified disagreement and the evidence supporting it. At governance review, ask which histories and voices authorize revision. These procedures can reveal errors in framing or representation that parameter sensitivity analysis would leave unchanged.

Decision and synthesis

Optimization is appropriate when decisions are consequential, alternatives exist, evidence can discriminate among them, and the model can be governed. It is not automatically appropriate because a quantity can be measured.

Match method to consequence

We select the simplest method that supplies the evidence needed for the decision. A checklist may suffice when there are few choices and qualitative criteria. A spreadsheet may support a transparent comparison. Linear or convex optimization is useful when the structure fits and a

certificate is required. We use simulation, mixed-integer, stochastic, bilevel, or learning methods when their additional capabilities are needed.

We report more than x^*. The recommendation includes the selected action, objective value, feasibility evidence, optimality or approximation gap, sensitivity, assumptions, validation evidence, and responsible owner. These records explain the result and support its proposed use.

A final bird's-eye view

Across calculus, chips, clinics, cities, markets, poems, and music, the recurring pattern is a loop: frame, represent, search, test, decide, observe, and revise.

The transitions between these activities determine whether a result remains valid in use. Compositional assurance connects these conditional guarantees. Category theory provides language for examining composition and preservation of structure across interfaces. The application also needs to preserve domain meaning, supporting evidence, and accountable authority.

We therefore need to identify which properties should remain invariant as the world, model, and values change, and which changes require a new formulation.

At each step, we record which structure was borrowed, which relationships it preserves, and which information it omits. We also identify the local evidence and authority needed to address those omissions. This transfer ledger makes the analogy and its limits available for review.

Record three independent decisions: stop computing, authorize or refuse action, and schedule reopening. The mathematically best candidate can be refused. An approximate candidate can be authorized with monitoring. A successful deployment can be retired when purpose or context changes.

A defensible stopping decision records why further search is unnecessary, unsupported, or too costly for the purpose. The resulting improvement remains subject to observation, review, and revision.

Final field checklist

- State the purpose and baseline before naming the objective.
- Type every value as target, constraint, penalty, veto, right, or process requirement.
- Keep model uncertainty separate from contested preference and absent authority.
- Validate optimized candidates at the boundary where they will act.
- Name what each disciplinary translation preserves and forgets.
- Assign approval, override, incident, appeal, rollback, and retirement authority.
- Stop search when model or value uncertainty dominates numerical improvement.
- Stop deployment when invariants, evidence, or controllability fail.
- Refuse systems whose purpose, authority, evidence, or repair cannot be made legitimate.

Translation lens. We can connect optimization models from different disciplines by specifying which relationships their composition preserves. Each application retains its own requirements for evidence, harm, agency, and meaning. A shared objective is not required for every discipline.

Acknowledgements

The author thanks the independent referee whose detailed reading sharpened the book's structure, mathematics, references, navigation, and illustrations. Responsibility for the synthesis and any remaining errors belongs to the author.

About the idea's author

Dmitry Vostokov is a software diagnostics researcher, engineer, and author whose work connects mathematical structure, computing systems, evidence, and practical problem solving. His broader writing develops pattern-oriented and transdisciplinary languages for diagnosing and improving complex technical systems.

Appendices

Appendix A · Mathematical toolkit

This appendix collects the notation and results most often needed to read optimization work. It is a map, not a substitute for a course in analysis, probability, numerical linear algebra, or algorithms. For the full teaching treatment, see Appendix F, Units 8–22 for the mathematical foundations, Units 23–30 for optimization, and Unit 41 for guarantees; this appendix is the summary card.

Sets, vectors, and matrices

- $x \in \mathcal{X}$ means that x belongs to feasible or ambient set $\mathcal{X}$.
- $\mathbb{R}^n$ is n-dimensional real space.
- $\| x \|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$ for $p \geq 1$; $\| x \|_\infty = \max_{1 \leq i \leq n} |x_i|$.
- $x^T y$ is an inner product in Euclidean coordinates.
- $A \succcurlyeq 0$ means symmetric matrix A is positive semidefinite: $x^T A x \geq 0$ for all x.
- $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ are extreme eigenvalues for a symmetric matrix.

Norm choice expresses geometry. An ℓ_1 ball has corners that encourage sparse solutions; an ℓ_2 ball is rotationally symmetric; an ℓ_∞ ball bounds each coordinate.

Differential quantities

For $f: \mathbb{R}^n \to \mathbb{R}$, the gradient is

$$\nabla f(x) = [\partial f / \partial x_1 \quad \cdots \quad \partial f / \partial x_n]^T. \tag{120}$$

The Hessian is $\nabla^2 f(x)$ with entries $\partial^2 f / (\partial x_i \, \partial x_j)$. For vector function $r: \mathbb{R}^n \to \mathbb{R}^m$, the Jacobian J_r collects first derivatives. The directional derivative in direction d is

$$Df(x)[d] = \lim_{t \downarrow 0} \frac{f(x+td) - f(x)}{t}. \tag{121}$$

Automatic differentiation evaluates derivatives of a program by repeatedly applying the chain rule. Forward mode is efficient for few inputs; reverse mode is efficient for few scalar outputs, subject to storage and control-flow costs.

Probability and statistics

Expectation, variance, and covariance are

$$\mathbb{E}[X], \qquad \mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}X)^2], \qquad \mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)]. \tag{122}$$

Bayes' rule, for $P(B) > 0$, is

$$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}. \tag{123}$$

A confidence interval is a procedure with repeated-sampling coverage under assumptions; it is not generally a posterior probability that the parameter lies in the realized interval. A

Bayesian credible interval is a posterior probability statement conditional on model and prior. Prediction intervals address future observations and are usually wider.

Optimization statements

The expression

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x) \tag{124}$$

returns minimizers; $\min_{x \in \mathcal{X}} f(x)$ returns the minimum value. An infimum may exist without being attained. A local minimum has no better feasible point in a neighborhood. A global minimum has no better feasible point anywhere in $\mathcal{X}$.

For smooth unconstrained optimization, stationarity $\nabla f(x^*) = 0$ is necessary at an interior local optimum. It is sufficient for global optimality only with additional structure such as convexity. For constrained problems, use feasible directions, normal cones, or KKT conditions.

Convergence vocabulary

If errors $e_k \to 0$:

- linear convergence means roughly $e_{k+1} \leq c e_k$ for $0 < c < 1$;
- superlinear means $e_{k+1}/e_k \to 0$;
- quadratic means $e_{k+1} \leq c e_k^2$ near the solution;
- sublinear rates include $O(1/k)$ and $O\left(1/\sqrt{k}\right)$.

Iteration count is only one cost. Derivative evaluation, factorization, memory, communication, precision, and parallel efficiency determine wall-clock behavior.

Approximation and error

Distinguish four errors:

1. **Representation error:** the model omits or simplifies reality.
2. **Estimation error:** parameters are inferred from finite or biased data.
3. **Discretization error:** continuous objects are approximated on a mesh or basis.
4. **Optimization error:** the algorithm stops short of the model optimum.

Driving optimization error below the other three may waste computation and create false confidence.

Appendix B · Method cards

Each card states when a method is a plausible first choice and what to verify. Hybrid methods are common.

Linear programming

Use when: objectives and constraints are linear and decisions continuous.

Strengths: mature solvers, dual values, certificates, scalability with sparsity.

Watch: wrong linearization, units, ill-conditioning, degeneracy, hidden integrality.

Report: primal/dual feasibility, objective, gap, tolerances, sensitivity.

Convex optimization

Use when: the objective and feasible set are convex or can be represented by disciplined transformations.

Strengths: global optimality, duality, composable modeling.

Watch: domain errors, scaling, solver tolerances, converting hard constraints into penalties.

Report: formulation class, constraint qualification, residuals, dual certificate.

Mixed-integer optimization

Use when: choices are discrete and linear or convex relaxations are informative.

Strengths: exactness certificates, rich logical modeling, strong commercial and open solvers.

Watch: weak big-M constants, symmetry, huge time-indexed formulations, false precision.

Report: incumbent, best bound, gap, runtime, node count, formulation choices.

Constraint programming and SAT/SMT

Use when: logical, temporal, combinatorial, or global constraints dominate.

Strengths: propagation, expressive discrete structure, proof of infeasibility.

Watch: weak propagation, encoding size, missing optimization bounds.

Report: solver theory, encoding, solution/proof status, limits.

Dynamic programming

Use when: optimal substructure and a manageable state summarize the past.

Strengths: exact recursion, policy outputs, interpretable value functions.

Watch: curse of dimensionality, discretization, incorrect Markov state.

Report: state/action definitions, boundary conditions, approximation error.

Gradient and Newton-type methods

Use when: variables are continuous and derivatives are available or estimable.

Strengths: high-dimensional scaling; rapid local convergence for second-order methods.

Watch: saddle points, conditioning, noisy gradients, line-search failure, nonphysical regions.

Report: initialization, scaling, derivative checks, stopping criterion, trajectory variability.

Proximal and splitting methods

Use when: objectives decompose into smooth and structured nonsmooth terms.

Strengths: sparsity, constraints via projections, distributed decomposition.

Watch: step-size rules, slow convergence to high accuracy, primal-dual residual interpretation.

Report: operator/prox definitions, residuals, convergence assumptions.

Derivative-free and black-box optimization

Use when: the evaluator supplies values or observations but usable derivatives are unavailable. Match the method to continuous, discrete, mixed, or conditional variables; evaluation cost; noise; and the available budget.

Strengths: direct search and interpolation models support local search; SPSA provides inexpensive randomized gradient estimates; population methods explore through candidate distributions; Bayesian methods use probabilistic models to allocate costly evaluations.

Watch: geometry, dimensionality, noise, hidden constraints, evaluator failures, nonstationarity, acquisition cost, and assumptions behind any convergence or safety claim. A local stationarity result does not certify global optimality.

Report: representation, baseline, initial design, evaluation and wall-time budgets, repeats, failures, feasibility, stopping rule, and independent confirmation. Distinguish best observed response, estimated candidate quality, and certified bounds. See the variant descriptions in Chapter 5.

Bayesian optimization variants

Use when: evaluations are costly enough to justify model fitting and acquisition search. Choose surrogate and acquisition assumptions that match the observed response and the configuration space.

Strengths: explicit allocation of evaluations using predicted response and uncertainty; support for noisy, constrained, mixed, batch, multi-fidelity, multiobjective, preferential, and related-task settings.

Watch: miscalibrated uncertainty, biased low-fidelity rankings, duplicate batches, negative transfer, changing preferences, and unsafe extrapolation. Safe exploration requires additional assumptions and an appropriate initial safe set.

Report: surrogate, acquisition, noise and constraint models, treatment of pending runs, fidelity relation, training data from other tasks, fitting overhead, and target-condition confirmation.

Evolutionary and population methods

Use when: the representation supports suitable variation operators and the evaluation budget permits population search. CMA-ES and differential evolution address continuous variables; other representations may need specialized operators.

Strengths: parallel exploration, flexible representation, Pareto sets.

Watch: large evaluation budgets, representation bias, weak baselines, premature convergence.

Report: initialization, population size, operators or covariance adaptation, restarts, boundary handling, seeds, evaluation and wall-time budgets, and independently confirmed results.

Robust optimization

Use when: uncertainty sets are defensible and constraint failure matters.

Strengths: explicit worst-case protection; tractable counterparts for some sets.

Watch: overconservatism, arbitrary uncertainty sets, correlated errors.

Report: uncertainty construction, protection level, price of robustness.

Stochastic and chance-constrained optimization

Use when: probability distributions or scenarios are credible.

Strengths: expected-value, tail, recourse, service-level formulations.

Watch: rare events, scenario bias, dependence, sample size, time inconsistency.

Report: distribution/scenarios, seed, sampling error, out-of-sample stress tests.

Multiobjective optimization

Use when: trade-offs should remain visible.

Strengths: Pareto frontier, preference elicitation, avoids premature scalarization.

Watch: too many objectives, incomparable scales, dominated displays, hidden veto constraints.

Report: objective definitions, normalization, dominance rule, selected point, rationale.

Simulation optimization

Use when: a validated simulator is the evaluable model.

Strengths: complex queues, networks, agent behavior, uncertainty.

Watch: simulation bias, correlated noise, long transients, optimizer exploiting simulator artifacts.

Report: calibration, validation, warm-up, replications, variance reduction, uncertainty.

Reinforcement learning and optimal control

Use when: actions affect future states and repeated feedback is available.

Strengths: policies rather than one-time decisions; explicit dynamics or learned value.

Watch: unsafe exploration, reward misspecification, partial observation, distribution shift.

Report: state/action/reward, horizon, simulator fidelity, off-policy evaluation, safety limits.

Human-in-the-loop and participatory methods

Use when: preferences, tacit knowledge, or legitimacy cannot be fixed in advance.

Strengths: reveals constraints and meanings, supports negotiation, builds ownership.

Watch: participation without influence on the decision, fatigue, power differences, and unstable preference models.

Report: who participated, what authority they had, disagreements, revisions, recourse.

Critical interpretation and counter-reading

Use when: a metric, category, model, or proposed optimum may suppress meanings, alternatives, or power relations.

Strengths: reveals hidden premises; generates adversarial cases; distinguishes disagreement about facts, frames, and authority.

Watch: critique without a decision link, unbounded suspicion, tokenized participation, treating one counter-reading as final.

Report: focal text or artifact, situated readers, primary and counter-readings, evidence, suppressed alternatives, resulting model change.

Framing-loop analysis

Use when: search, measurement, ranking, recommendation, or design changes the preferences, demand, categories, or evidence used to evaluate it.

Strengths: detects endogenous objectives, performative metrics, feedback-created data, shifts in the feasible imagination.

Watch: causal stories without observation, confusing ordinary adaptation with corruption, monitoring only the original metric.

Report: initial frame, intervention, feedback channel, observed frame change, affected groups, stopping trigger, reframing decision.

Plural interpretation and comparative reading

Use when: evidence supports several non-equivalent readings and premature aggregation would erase purpose, voice, or context.

Strengths: preserves warranted disagreement, provenance, scope conditions, consequences of alternative interpretations.

Watch: false balance, interpretations detached from evidence, hidden selection among readings, indecision without a governance rule.

Report: interpretation set, supporting evidence, conflicts and shared ground, who may select for which decision, what remains unresolved.

Appendix C · Domain translation matrix

The matrix is a bird's-eye index. It shows recurring objects without asserting that the disciplines are interchangeable.

Table 2. Domain translation matrix: decisions, objectives, invariants, and evidence.

Domain	Typical decisions	Common objectives	Hard constraints and invariants	Evidence and validation
Mathematics	point, function, measure, proof path	value, bound, convergence	topology, algebra, feasibility	theorem, counterexample, numerical residual
Algorithms	representation, operation sequence	time, memory, approximation	correctness, resource model	proof, benchmark, adversarial cases
Software	architecture, code, deployment	quality vector, lead time	semantics, security, compatibility	tests, profiles, incidents, users
Systems/cloud	placement, scheduling, replication	latency, throughput, cost, carbon	capacity, service-level objectives, consistency	traces, load tests, failure drills
AI/data	model, data, threshold, policy	loss, calibration, utility	privacy, safety, compute	holdout, causal test, monitoring
Hardware	architecture, placement, routing	PPA, yield, reliability	physics, timing, fabrication rules	sign-off, simulation, silicon
Mechanical/civil	shape, size, material, sequence	mass, stiffness, lifecycle value	equilibrium, codes, buildability	analysis, test, inspection, field data
Process/materials	flows, conditions, composition	yield, purity, energy, hazard	balances, thermodynamics, safety	experiment, pilot, plant history
Energy/environment	capacity, dispatch, protection	cost, emissions, resilience	carbon, reliability, ecological limits	scenarios, lifecycle data, monitoring
Transport/robotics	route, trajectory, fleet, control	time, energy, access, safety	dynamics, collision, service	simulation, trials, operations
Biology/ecology	intervention, reserve, synthetic design	persistence, function, restoration	evolution, conservation, consent	field/lab data, replication
Clinical medicine	test, treat, monitor, refer	benefit, harm, preference fit	consent, contraindication, standard of care	trials, calibration, shared decision
Public health	program and resource portfolio	population health, equity, resilience	rights, capacity, access	surveillance, causal evaluation, deliberation
Economics/finance	allocation, mechanism, portfolio	welfare, profit, risk	budgets, incentives, regulation	market data, stress tests, causal evidence
Policy/law	intervention, rule, eligibility	public outcomes, legitimacy	rights, due process, authority	impact evaluation, audit, challenge
Human factors	interface, function allocation, workload	performance, safety, inclusion	capability, fatigue, accessibility	observation, usability test, incident
Architecture/design	form, material, program	function, place, delight, carbon	site, code, craft, access	prototype, simulation, critique, use

Domain	Typical decisions	Common objectives	Hard constraints and invariants	Evidence and validation
Art/music/literature	rule, form, gesture, revision	expressive intention, surprise	medium, provenance, internal coherence	critique, performance, interpretation
Games/sport	strategy, rules, training	win, balance, spectacle, health	rules, integrity, bodies	match data, playtest, athlete report

Table 2 should be read across rows before it is read down columns: the shared grammar is useful only when each domain's admissibility rules and evidence remain intact.

Structural correspondences

Some patterns transfer reliably:

- **Conservation:** flow balance, mass balance, probability normalization, accounting identities.
- **Duality:** quantities and prices, flows and potentials, primal plans and certificates.
- **Feedback:** controllers, markets, learning systems, policy response, critique and revision.
- **Interfaces:** software APIs, physical ports, organizational handoffs, categorical boundaries.
- **Regularization:** mathematical penalties, engineering margins, institutional safeguards.
- **Pareto frontiers:** visible trade-offs when no single ordering is legitimate.

Other correspondences are dangerous. Biological fitness is not moral worth; market price is not social value; engagement is not well-being; statistical normality is not health; compression is not interpretation; equilibrium is not justice.

Appendix D · Reference validation and audit trail

Scope and selection standard

The main bibliography contains 229 cited records, prioritizing foundational monographs, peer-reviewed research, official standards, legislation, and public guidance. Version 34 added 23 records for black-box optimization. Version 35 revised word choice and sentence structure. Version 36 corrected attributions and added twelve supporting references. Version 37 adds the original 1978 submodularity paper and corrects the published Parikh–Boyd page range. Appendix F retains its separately numbered source list.

A DOI, ISBN, arXiv identifier, standard number, or official catalog page establishes bibliographic identity; it does not certify the argument made from the source. Citation coverage and record structure were checked throughout. Link liveness and the current applicability of every standard, law, guideline, and software practice remain time-dependent.

Reproducible checks

1. **Structure.** Citation keys are unique; every cited key resolves; each record has a title, year, and accountable author or institution; container fields are checked by type.
2. **Coverage.** Every in-text citation appears in the generated bibliography, and every generated record is cited.
3. **Identifiers.** The new black-box records were checked against primary papers, publisher records, proceedings, author resources, or repository records. Earlier corrected records retain the identifier and edition notes documented in the correction ledger.
4. **Rendering.** ISBNs that would otherwise be hidden by the citation style are repeated in an explicit note. A later-edition identifier is not substituted for an original edition merely to fill a field.
5. **Link hygiene.** Known dead, TLS-failing, stale, or edition-mismatched links are omitted or replaced only when an authoritative record identifies the same work or a clearly labeled authorized reprint.

Correction ledger

- The NIST SSDF and AI RMF records identify the report authors, institutional provenance, and report numbers. In-text institutional references point to the corresponding reports.
- Schrijver and Spivak carry visible ISBN notes alongside official catalog pages. Other otherwise-unlinked books expose their recorded ISBNs in notes; pre-ISBN originals are labeled rather than assigned identifiers from later editions.
- Bendsøe and Sigmund is aligned to the second corrected printing dated 2004, with the Springer DOI and hardcover ISBN stated together.
- Campbell's 1976 working paper links to the authorized 2011 reprint and says so; the DOI is not misrepresented as an identifier for an unaltered 1976 digital issue.
- Farhi, Goldstone, and Gutmann is explicitly identified as arXiv preprint **arXiv:1411.4028**.

- Publisher ampersands and particles in personal and institutional names are protected from author-list parsing. The ledger describes only records that remain in the generated bibliography.
- Version 26 added four cited records—Sen (1985), Kantorovich (1960), Kaplan et al. (2020), and Hoffmann et al. (2022). Version 27 replaced an uncited 2017 conference record with Breck et al. (2017), removed the city from the Sen (1985) imprint, and left 193 cited records.

Version 34 added 23 black-box optimization references, extending the bibliography from 193 to 216 records. Version 35 retained the equation sequence and figure assets while revising prose. Earlier citation-navigation repairs restored missing bibliography bookmarks and separated the Holland and Hoffmann targets. The claimed separation of script variables from punctuation in Equations (7) and (73) was not present in the Version 35 baseline; Version 36 applies that repair throughout the native equations and their atlas copies.

Version 36 repairs the Sen (1985) targets, adds the separate MADS erratum, cites PABBO's ICLR 2025 publication, and distinguishes physical climate science from mitigation evidence. It corrects the selection-composition criterion, local equation definitions, and missing hypotheses; consolidates repeated chapter treatments; restores technical terms and stopping conditions; rebuilds equation-specific atlas readings; and aligns figures, glossaries, and navigation. The formerly unnumbered displays become (10a) and (59a), retaining the original (1)–(124) sequence.

Version 37 restores the eight-field charter and omitted content, consolidates the remaining restatements, and repairs sentence links and figure-caption agreement. The atlas now distinguishes continuous and discrete study routes, completes expectation and weight definitions, and uses the local notation consistently. The primary records for Parikh–Boyd (2014) and Nemhauser–Wolsey–Fisher (1978) were checked on 12 September 2026. Equation numbering and exact atlas copies are retained.

Version 38 completes the generic feasible-set notation while retaining the case-specific set names. It aligns Figure 3 with its caption and description, repairs the remaining sentence links and atlas readings, leads the Parikh–Boyd record with its publication DOI, and records the dedicated stopping section in the subject index.

Version 39 uses the existing figure descriptions as alternative text and makes the feasible-set glossary label visibly bold.

Limits and date

The earlier audit snapshot was 22 August 2026, followed by the targeted black-box source review on 11 September. Version 36 adds a primary-source check of the referee's attribution and edition findings dated 12 September 2026. This review does not establish current operational applicability of every standard, law, clinical guideline, or software practice. Their applicable official editions remain the source for use in a particular setting.

The cross-disciplinary synthesis remains interpretation by the author. Citations identify evidence and intellectual lineage; they do not make the cited authors responsible for the synthesis or resolve disagreements among fields.

Appendix E · Extended figure descriptions

These descriptions supplement the concise captions and embedded alternative text. They identify layout, reading order, and the relation each figure is intended to clarify.

Figure 1

Caption. Bird's-eye view: purpose and values at center; model, evidence, search, and seven domains around them; governance, uncertainty, and feedback spanning the map.

Description. A concentric map puts PURPOSE & VALUES centrally; MODEL, EVIDENCE, and SEARCH around them; seven domain boxes outside; and GOVERNANCE · UNCERTAINTY · FEEDBACK above.

Figure 2

Caption. Seven-stage optimization lifecycle—frame, represent, search, test, decide, observe, revise—around value in context.

Description. Seven nodes circle VALUE IN CONTEXT clockwise: FRAME, REPRESENT, SEARCH, TEST, DECIDE, OBSERVE, and REVISE; REVISE returns to FRAME.

Figure 3

Caption. Structural fingerprint across six axes: variable and feasible-set structure, objective geometry, information regime, uncertainty, time and agency, and guarantee.

Description. Six branches run from OPTIMIZATION PROBLEM to variable and feasible-set structure, objective geometry, information regime, uncertainty, time and agency, and guarantee. Variable and feasible-set structure distinguishes variable types and admissible decisions; objective geometry identifies convexity, smoothness, and structure; information regime describes evaluator access; uncertainty describes random, robust, or adversarial variation; time and agency describe sequential and interactive decisions; guarantee identifies the claim sought from computation.

Figure 4

Caption. A timeline from ancient reasoning about the good, through calculus, variational principles, industrial operations research, convexity and complexity, to data-driven, compositional, and governed optimization.

Description. Six stages on a horizontal timeline connect ancient reasoning about the good; calculus; variational principles; industrial operations research; convexity and complexity; and data-driven, compositional, governed optimization.

Figure 5

Caption. Narrow-valley, multiple-basin, and saddle contour landscapes with distinct search directions.

Description. Three contour panels show a narrow valley with gold paths, multiple basins with gold/red trajectories, and a saddle with red/green directional arrows.

Figure 6

Caption. Feasible region, objective contours, active constraints, normal vectors, and the KKT balance at an optimum.

Description. A polygonal feasible region is crossed by objective contours. The optimum lies where two constraints are active; one objective gradient and two constraint normals are drawn with their vector-balance equation.

Figure 7

Caption. A 2 × 2 comparison of expected-value, risk-averse, robust, and adaptive decisions under uncertainty.

Description. A 2 × 2 grid compares four cells labeled “optimize the mean,” “control the tail,” “protect the admitted set,” and “observe and revise.” These correspond to expected-value, risk-averse, robust, and adaptive decisions.

Figure 8

Caption. A Pareto front showing dominated designs, nondominated alternatives, a knee region, aspiration point, and the effect of changing scalarization weights.

Description. A trade-off curve separates dominated points from nondominated alternatives. A knee, aspiration point, and two tangent scalarizations show different selection logics.

Figure 9

Caption. Closed loop from goal through controller or policy, system or world, and outcome, with disturbance and sensor or estimator feedback.

Description. GOAL → CONTROLLER / POLICY → SYSTEM / WORLD → OUTCOME. Disturbance enters the system; outcome returns through SENSOR / ESTIMATOR. The footer names model, delay, saturation, noise, and override.

Figure 10

Caption. Transport coupling between three source and three target masses; arrow thickness shows moved mass.

Description. Three source masses connect to three target masses through a feasible coupling γ. Source masses are 0.30, 0.35, and 0.35; target masses are 0.35, 0.40, and 0.25. Arrow width

is proportional to transported mass. The coupling preserves both marginals; selecting an optimal coupling additionally requires the ground cost.

Figure 11

Caption. Sequential composition of SYSTEM A and SYSTEM B through an interface, with f, g, g∘f, a local objective, and a constraint.

Description. SYSTEM A (X→Y) passes through INTERFACE Y to SYSTEM B (Y→Z) by f then g; a curved arrow shows g∘f. Lower boxes name objective and constraint.

Figure 12

Caption. Algorithm families positioned by available structural information and by evaluation scale and cost.

Description. A two-axis chart positions seven method families by structural information (horizontal) and evaluation scale and cost (vertical): Bayesian, evolutionary, derivative-free, stochastic-gradient, gradient, Newton/interior-point, and specialized combinatorial methods.

Figure 13

Caption. Six-layer stack from purpose and decision to hardware and energy.

Description. Six narrowing layers run from PURPOSE & DECISION through mathematical structure, algorithm family, representation, and runtime to HARDWARE & ENERGY.

Figure 14

Caption. Service flow from workload to outcomes, with observation and autoscaling feedback.

Description. Workload flows through QUEUE & ADMISSION and SCHEDULER & SERVICE to outcomes; OBSERVE and AUTOSCALE return feedback. Footer metrics include latency, throughput, reliability, cost, and carbon.

Figure 15

Caption. A radar chart comparing runtime, reliability, security, maintainability, delivery, and usability.

Description. A filled blue radar profile spans runtime, reliability, security, maintainability, delivery, and usability.

Figure 16

Caption. Coupled engineering disciplines exchanging shared design variables, states, sensitivities, and requirements.

Description. Six outer boxes—MISSION, STRUCTURE, FLUIDS, THERMAL, CONTROL, and COST / LIFE—exchange arrows with SHARED DESIGN. The arrows represent coupled requirements, shared variables, states, and sensitivities.

Figure 17

Caption. Hardware trade space around performance, power, area, and architecture or layout, with security, verification, yield, software, and temperature.

Description. PERFORMANCE, POWER, and AREA surround ARCHITECTURE & LAYOUT; security, verification, yield, software, and temperature extend the hardware trade space.

Figure 18

Caption. Contextual-fitness landscape with accessible variation and environmental change.

Description. Contextual fitness is plotted over a phenotype/genotype neighborhood; a gold accessible-variation path crosses a green landscape, and a red marker shows environmental change.

Figure 19

Caption. Two clinical thresholds divide posterior probability into OBSERVE, TEST / ELICIT, and TREAT regions.

Description. Test and treatment thresholds divide a 0-to-1 probability axis into OBSERVE, TEST / ELICIT, and TREAT; a central arrow shows increasing posterior probability.

Figure 20

Caption. Health-system allocation from needs, budget, and workforce to prevention, primary care, acute care, and readiness, governed by equity, access, rights, and geography.

Description. POPULATION NEEDS and BUDGET & WORKFORCE feed ALLOCATION PORTFOLIO, which branches to PREVENTION, PRIMARY CARE, ACUTE CARE, and READINESS under equity, access, rights, and geography.

Figure 21

Caption. A governance loop linking public goals, causal evidence, policy design, implementation, measurement, contestation, and revision.

Description. A circular governance process moves from public goals and causal evidence through policy design, implementation, measurement, contestation, and revision.

Figure 22

Caption. A responsible-optimization frame with rights as boundaries, stakeholder capabilities, distributional outcomes, and planetary constraints.

Description. A bounded responsible-optimization frame places rights at the border, planetary constraints outside ordinary trade-offs, and stakeholder capabilities and distributional outcomes inside.

Figure 23

Caption. A creative design space in which constraints shape a field of possibilities and critique redirects the search.

Description. A cloud of creative alternatives is shaped by constraint boundaries. A critique arrow redirects the search toward a revised region rather than a fixed optimum.

Figure 24

Caption. An eight-axis musical design radar spanning pitch, rhythm, timbre, gesture, space, form, expectation, and culture.

Description. A gold radar profile spans eight labeled axes: pitch, rhythm, timbre, gesture, space, form, expectation, and culture.

Figure 25

Caption. A narrative search diagram connecting world events, plot selection, discourse order, voice, reader inference, and revision.

Description. A directed narrative pipeline moves from world events to plot selection, discourse order, voice, reader inference, and revision, with feedback from interpretation to later choices.

Figure 26

Caption. Seven-stage evidence loop—question, experiment, simulate, calibrate, decide, operate, and learn—around a central EVIDENCE LOOP.

Description. Seven nodes circle EVIDENCE LOOP clockwise: QUESTION, EXPERIMENT, SIMULATE, CALIBRATE, DECIDE, OPERATE, and LEARN; LEARN returns to QUESTION.

Figure 27

Caption. Illustrative divergence of measured proxy and real value after a proxy becomes a target and participants adapt to it.

Description. Blue proxy and coral value curves share an initial trajectory and separate after the vertical “proxy becomes target” marker. Shading shows the widening difference caused by the illustrated adaptive response. The curves are schematic, not observations from a study.

Figure 28

Caption. Seven worked translations around shared boundary, decision, and evidence.

Description. BOUNDARY, DECISION, EVIDENCE sends arrows to DATA CENTER, HEAT HEALTH, INSULIN CONTROL, PUBLIC LIBRARY, MUSICAL PROSTHESIS, CHIP FAB, and COASTAL ADAPTATION.

Figure 29

Caption. Automated science, differentiable design, quantum or hybrid methods, and AI agents and institutions feed composable assurance.

Description. AUTOMATED SCIENCE, DIFFERENTIABLE DESIGN, QUANTUM / HYBRID, and AI AGENTS & INSTITUTIONS point inward to COMPOSABLE ASSURANCE, inside a frame whose footer lists governance, energy, baselines, uncertainty, and recourse.

Appendix F · Mathematics for Optimization

A mathematics companion from school foundations to advanced topics

Appendix Version 18 · Independent mathematics companion included in Optimization Across Disciplines, Version 39

We begin with arithmetic, fractions, algebra, graphs, functions, sets, logic, and proof. These provide the prerequisites for linear algebra, calculus, probability, statistics, optimization, dynamics, transport, information theory, topology, functional ideas, complexity, and category theory. The appendix helps readers interpret the mathematical statements in the volume and identify the earlier concepts on which they depend. It does not replace a full course in each subject.

The teaching order follows dependencies between concepts. For example, a Bellman equation uses functions, probability, sums, and optimization. A KKT system uses algebra, derivatives, geometry, and logic. A categorical description uses functions and composition to specify interfaces. Readers can begin at any unit and return to its prerequisites when a symbol or operation is unfamiliar.

Coverage. The atlas includes all 126 numbered equations in the chapters, front matter, and Appendix A. The original sequence (1)–(124) is retained; the two formerly unnumbered displays are labeled (10a) and (59a). Each atlas entry supplies its location, prerequisites, a reading in words, local symbol meanings, and structural tags. The teaching units derive and illustrate selected ideas.

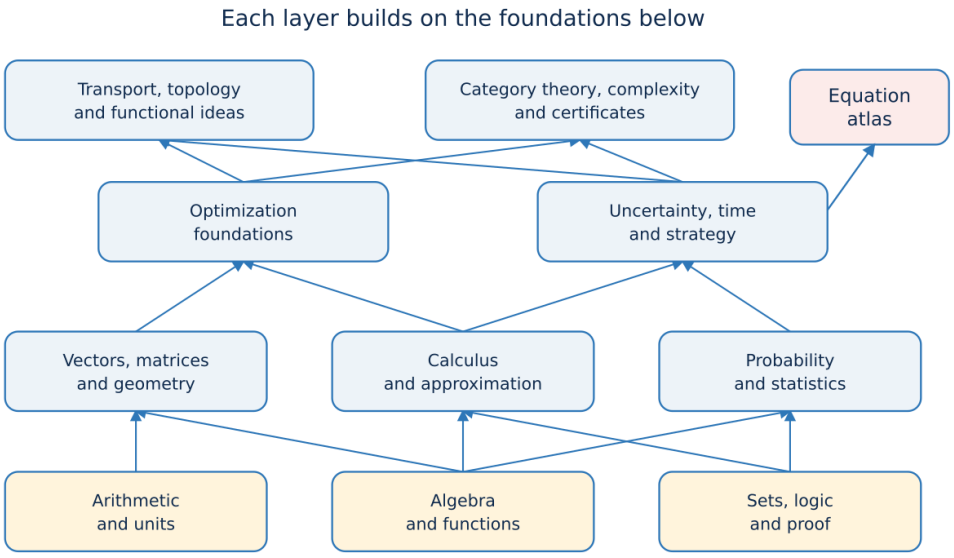


Figure F.1. Dependency map from arithmetic and logic through calculus, probability, optimization, dynamics, transport, category theory, and the 126-equation atlas.

The atlas on the right connects the volume's equations with the teaching units. Each entry points back to the concepts used in that equation, so it can be consulted alongside the dependency map.

How to use this appendix

Read Units 1–7 if school mathematics feels rusty; use Units 8–22 to rebuild the standard undergraduate foundations; use Units 23–35 for the core mathematics of optimization; and use Units 36–41 as bridges to the advanced language employed in the volume. Unit 42 demonstrates cross-disciplinary translation. The atlas then supports equation-by-equation lookup.

We read a displayed equation by first identifying its objects and their types. We then distinguish variables from fixed quantities and describe the relation or optimization operation in words. Finally, we check assumptions, units, and the conclusion supported by the expression. These steps connect the notation with its meaning in the application.

Appendix F figure map and descriptions

1. Figure F.1: dependency map. Three foundation blocks - arithmetic and units; algebra and functions; sets, logic, and proof - feed vectors and geometry, calculus and approximation, and probability and statistics. Higher arrows connect these to optimization, uncertainty, transport, topology, category theory, complexity, certificates, and the equation atlas.
2. Figure F.2: numbers, ratios, and units. A number line orders values from -2 to 4. A ratio of 3:5 becomes the fraction 3/5 and 60 percent, while 72 km/h becomes 20 m/s after multiplication by 1000/3600; both paths preserve the represented quantity.
3. Figure F.3: functions, derivatives, and integrals. Three aligned panels show the same U-shaped function: its input-output graph, a tangent line marking local slope at a point, and a shaded area marking accumulated value over an interval. The panels distinguish value, rate of change, and accumulation.
4. Figure F.4: vectors and matrices. Vectors u and v share an origin and combine to u + v. A matrix A then maps a rectangle to a rotated, scaled, or sheared parallelogram, presenting the matrix as a composable linear transformation rather than merely a rectangular array of numbers.
5. Figure F.5: gradients and curvature. Nested elliptical contours surround a minimum. At a sample point, the red gradient crosses contours toward steepest increase, while the green negative-gradient direction points through lower contours toward lower values; the contour geometry depicts local curvature.
6. Figure F.6: Bayesian updating. Prior probability branches into likelihood and evidence, which combine at the posterior. The posterior informs a decision threshold; a decision can produce new data or revision that feeds back into the next update, separating inference, decision, and learning.

7. Figure F.7: convexity and KKT geometry. Curved objective contours meet a polygonal convex feasible set at a boundary optimum. The objective gradient and active-constraint normal point in opposing directions there, visualizing the first-order KKT balance created by an active constraint. A marked unconstrained minimum outside the set shows why the descent direction is blocked at the boundary.
8. Figure F.8: numerical iteration and evidence. The main path runs from scaling and initialization through model evaluation, step computation, acceptance or rejection, and stopping or iteration. Residuals, optimality gap, units, and sensitivity inform the step, while a feedback arc returns a revised iterate, trust region, or fidelity.
9. Figure F.9: Pareto reasoning. Gray points are dominated and gold points form the nondominated front for two objectives that are both minimized. A circled knee identifies a region where further improvement in one objective requires a comparatively large sacrifice in the other, without declaring a unique best choice.
10. Figure F.10: risk and CVaR. A loss distribution marks expected loss near its center and a value-at-risk threshold in the right tail. The tail beyond that quantile is shaded; conditional value at risk averages losses within this shaded region, distinguishing frequency from tail severity.
11. Figure F.11: Bellman and control feedback. State enters a policy, which selects an action applied to the system or world. The resulting reward returns to policy evaluation and the next state loops into the following decision, so future value becomes part of the present comparison.
12. Figure F.12: Markov chains and queues. The upper row shows neighboring states connected by forward and backward transition arrows. The lower row follows arrivals into a waiting line, through service, and to departure.
13. Figure F.13: optimal transport. Three source masses connect to three target masses through labeled transport links. Splitting a source across destinations illustrates a coupling rather than one-to-one matching, and the optimization box states the governing objective: minimize transported mass multiplied by distance.
14. Figure F.14: categorical composition. The upper chain maps input object X through model f, decision g, and consequence h to Y, with a single arc showing the composite morphism. The lower pair shows parallel components joined by a shared interface or contract; composition is valid only when meanings and constraints align.
15. Figure F.15: complexity and certificates. The upper sequence connects problem size to algorithmic cost, approximation or residual, and a claim with explicit scope. Below, feasible witnesses, bound or dual witnesses, and validation evidence are linked as complementary forms of assurance rather than interchangeable proof.
16. Figure F.16: method families. Four structural problem classes - continuous smooth, convex structured, discrete or logical, and black-box or global - map to representative algorithm families. All paths converge on certificates and validation, emphasizing that problem structure determines admissible claims as well as solver choice.

Part F.1 · School mathematics rebuilt

Unit 1 · Numbers, arithmetic, and units

Prerequisites. none beyond counting.

Why it matters in this volume. Optimization compares quantities whose meanings and units differ. For example, cost, mass, latency, probability, utility, carbon, dose, and musical distance require separate interpretations even when they are all stored numerically.

A number represents a magnitude and may have a unit. We add like quantities, multiply to form products, and divide to form rates or ratios. A negative number may represent direction or deficit. Zero may indicate an actual quantity, a boundary, or an unavailable measurement. Before calculating, we estimate the expected order of magnitude so that we can recognize an implausible result.

Dimensional analysis treats units algebraically. The dimensions on both sides of a valid equation must be compatible. Scaling can replace a dimensional variable with a dimensionless one and may improve numerical conditioning. Significant figures indicate measurement precision. Scientific notation separates the order of magnitude from the leading digits.

$$72\,\mathrm{km/h} = 72\,\frac{1000\,\mathrm{m}}{3600\,\mathrm{s}} = 20\,\mathrm{m/s} \tag{F.1}$$

Unit factors equal one, so the represented speed is unchanged.

$$x_{\mathrm{scaled}} = \frac{x - x_0}{s} \tag{F.2}$$

Subtract a reference value and divide by a meaningful scale.

Worked reading. If a server processes 2,400 requests in 60 seconds, its observed throughput is 40 requests per second. That rate does not by itself state latency, burst tolerance, or maximum sustainable load. The mathematical value and the operational claim must remain distinct.

Common traps.

- Adding meters to seconds
- Treating a rounded display value as exact
- Ignoring whether zero means none, unknown, or below detection

Where it reappears. Chapters 3, 4, 8, 10, 11, 12, and the worked translations repeatedly use units, rates, scaling, and residuals.

Sources. [F1; F2]

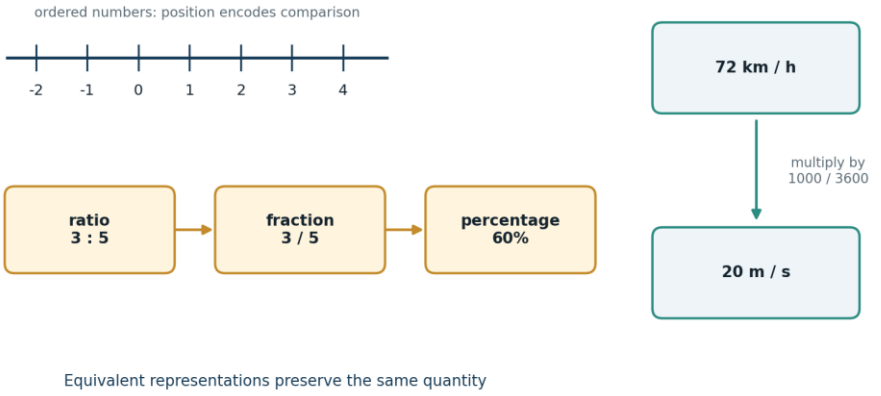


Figure F.2. Equivalent representations of numbers, ratios, percentages, and unit conversion.

Unit 2 · Fractions, ratios, proportions, and percentages

Prerequisites. Unit 1.

Why it matters in this volume. Mixtures, allocations, probabilities, utilization, error rates, fairness metrics, and portfolio weights are all forms of part-to-whole reasoning.

A fraction is a number, a ratio is a comparison, and a percentage is a ratio per hundred. The same symbol can play different roles: 3/5 may denote a quotient, a probability, a slope, or a share. A proportion asserts equality between two ratios; cross-multiplication is valid because both denominators are nonzero and the same equality-preserving multiplication is applied to both sides.

Weighted averages use nonnegative shares that sum to one. Relative change divides a change by a baseline, so a percentage change is undefined when the baseline is zero and unstable when it is very small. Percentage points and percent change are different: moving from 20% to 25% is a rise of five percentage points and a relative increase of 25%.

$$\sum_{i=1}^{n} w_i = 1, \qquad \bar{x}_w = \sum_{i=1}^{n} w_i\, x_i, \qquad w_i \geq 0 \tag{F.3}$$

Nonnegative normalized weights form a convex combination and therefore a weighted average.

$$\text{relative change} = \frac{x_{\text{new}} - x_{\text{old}}}{x_{\text{old}}} \tag{F.4}$$

Change is measured relative to a nonzero baseline.

Worked reading. A hospital allocating 40%, 35%, and 25% of a flexible budget has weights summing to one. If the outcomes being averaged use different denominators or populations, however, the weighted average can be meaningless even though the arithmetic is correct.

Common traps.

- Confusing percentage points with percent change
- Averaging ratios with incompatible denominators
- Normalizing weights that should instead represent absolute capacity

Where it reappears. Allocation, mixture, portfolio, probability, utilization, and multiobjective examples throughout Chapters 6, 8, 11–13, and 18–19.

Sources. [F1; F2]

Unit 3 · Powers, roots, exponentials, and logarithms

Prerequisites. Units 1–2; letter notation is introduced as needed and developed in Unit 4.

Why it matters in this volume. Growth, decay, discounting, learning curves, entropy, likelihood, numerical conditioning, and algorithmic complexity all use powers or logarithms.

A power repeats multiplication. A root reverses a power on an appropriate domain. An exponential has the variable in the exponent and models proportional growth or decay. A logarithm asks which exponent produces a number. Logarithms turn multiplication into addition, which is why log-likelihoods and decibels are computationally convenient.

The natural exponential and logarithm use base e. For positive a and b, log(ab)=log a+log b. Exponential growth is not merely ‘fast’; its defining property is a constant relative rate. For a>0, the equation $x^2=a$ has the two real solutions $\pm\sqrt{a}$; for a=0 it has one real solution, and for a<0 it has none. The symbol $\sqrt{a}$ conventionally denotes the nonnegative principal root when a≥0.

$$a^m a^n = a^{m+n}, \qquad \log(ab) = \log a + \log b \quad (a, b > 0). \tag{F.5}$$

Exponent and logarithm rules express inverse algebraic structures.

$$x(t) = x_0 e^{rt} \tag{F.6}$$

A constant relative rate r produces exponential change.

Worked reading. If a failure probability is the product of many small conditional factors, computing the logarithm of the product as a sum can avoid underflow. The transformation preserves ordering because the logarithm is strictly increasing on positive numbers.

Common traps.

- Using logarithms of nonpositive real numbers
- Assuming every increase is exponential
- Dropping units from an exponent, which must be dimensionless

Where it reappears. Complexity, probability, information, finance, reliability, and learning models in Chapters 4, 8, 9, 10, 12, 13, and 20.

Sources. [F2; F3]

Unit 4 · Algebra: expressions, equations, and inequalities

Prerequisites. Units 1–2; Unit 3 supplies supporting examples with powers and logarithms.

Why it matters in this volume. Optimization models are algebraic descriptions of what may change, what must remain true, and how consequences are computed.

An expression names a quantity; an equation states that two expressions are equal; an inequality states an order. A variable may be an unknown to solve for, a decision under control, a measured input, or a random quantity. The role matters. Solving an equation applies reversible operations that preserve the solution set. Solving an inequality requires extra care: multiplying by a negative number reverses the order.

A system collects several simultaneous requirements. Substitution eliminates variables; elimination combines equations; factorization exposes roots or structure. An identity is true for all admissible values, whereas an equation to solve is true only for particular values. Constraints may be strict, non-strict, equality, logical, or set-membership statements.

$$ax + b = c \quad \Rightarrow \quad x = \frac{c-b}{a}, \qquad a \neq 0 \tag{F.7}$$

Undo addition and multiplication while recording the nonzero condition.

$$-2x \leq 6 \quad \Rightarrow \quad x \geq -3 \tag{F.8}$$

Division by a negative reverses the inequality.

Worked reading. For a budget $p_1x_1+p_2x_2 \leq B$, the coefficients are unit prices, the variables are quantities, and the right-hand side is available money. The inequality encodes admissibility, not a preference to spend the entire budget.

Common traps.

- Cancelling a quantity that might be zero
- Changing an equation on only one side
- Treating every variable as a decision variable

Where it reappears. The master form and every mathematical chapter, especially Chapters 1, 4–7, 10–13, and 18–19.

Sources. [F2]

Unit 5 · Coordinates, graphs, and functions

Prerequisites. Units 1–4.

Why it matters in this volume. Objectives, constraints, response surfaces, calibration curves, trajectories, and Pareto fronts are functions viewed through graphs.

A function assigns exactly one output to each input in its domain. The codomain specifies the possible type or set of outputs, and the image contains the outputs actually attained. A graph represents this assignment geometrically. Its slope describes output change per unit of input change. Intercepts, turning points, discontinuities, and asymptotes help us examine the function's structure.

Composition applies one function after another. An inverse reverses a one-to-one function on its image. Piecewise functions describe rules that change by region. Level sets collect inputs with the same output and become contour lines or surfaces in higher dimensions.

$$(g \circ f)(x) = g\big(f(x)\big) \tag{F.9}$$

Composition passes the output of f into g.

$$m = \frac{f(x_2) - f(x_1)}{x_2 - x_1} \tag{F.10}$$

A secant slope is average change over an interval.

Worked reading. A latency penalty may be flat below a service objective and rise sharply above it. Modeling it as one global straight line erases the threshold; a piecewise or smooth-threshold function represents the intended semantics more faithfully.

Common traps.

- Forgetting to state the domain
- Assuming a plotted curve proves behavior outside the plotted range
- Confusing f^{-1} with 1/f

Where it reappears. Response surfaces, calibration, thresholds, objectives, and transformations throughout the volume.

Sources. [F2; F3]

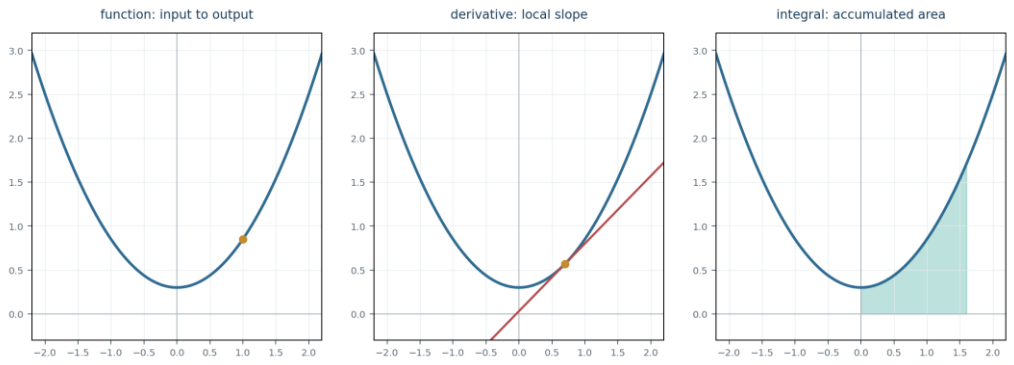


Figure F.3. A function, its local derivative, and its accumulated integral.

Unit 6 · Sequences, series, and summation notation

Prerequisites. Units 1–5.

Why it matters in this volume. Iterations, time steps, discounted returns, accumulated costs, sample averages, and finite resource totals are indexed collections.

A sequence is an ordered list generated by a rule. Sigma notation compresses a finite or infinite sum and makes the index range explicit. Reindexing changes labels, not values, when done consistently. A series is the sum of sequence terms; infinite series require a convergence question before algebraic manipulation is justified.

Geometric sequences multiply by a constant ratio. Arithmetic sequences add a constant difference. Recurrences define later terms from earlier ones and are the simplest form of dynamics. Discount factors reduce the weight of distant terms but do not automatically guarantee convergence if rewards grow too quickly.

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2} \tag{F.11}$$

A closed form replaces a linear number of additions.

$$\sum_{t=0}^{\infty} \gamma^t = \frac{1}{1-\gamma}, \qquad |\gamma| < 1 \tag{F.12}$$

The infinite geometric series converges only under the stated condition.

Worked reading. A ten-step cost $\sum_t c_t$ is finite regardless of discounting. An infinite-horizon cost needs assumptions about γ and c_t. Writing the summation symbol does not discharge those assumptions.

Common traps.

- Omitting index limits
- Manipulating divergent series as if finite
- Confusing an iteration counter with physical time

Where it reappears. Algorithms, dynamic programming, control, reinforcement learning, finance, and lifecycle accounting in Chapters 4, 7–13, and 18–19.

Sources. [F3]

Unit 7 · Sets, logic, quantifiers, and proof

Prerequisites. Units 1–6.

Why it matters in this volume. Feasible sets, events, state spaces, categories, guarantees, and certificates are all built from sets and logical statements.

A set is determined by membership, not order. Union means ‘in either’; intersection means ‘in both’; complement means ‘not in the set’ relative to a stated universe. Logic distinguishes implication from equivalence and universal claims from existential claims. Negating ‘for every’ produces ‘there exists a counterexample’; negating ‘there exists’ produces ‘for every, not.’

A proof is a reproducible chain from assumptions to conclusion. Direct proof, contrapositive, contradiction, induction, and construction answer different logical shapes. A counterexample

refutes a universal claim but does not establish a competing universal claim. Necessary and sufficient conditions point in opposite implication directions.

$$A \subseteq B \quad \Leftrightarrow \quad \forall x\,(x \in A \Rightarrow x \in B) \tag{F.13}$$

Subset inclusion is a quantified implication.

$$\neg\big(\forall x\,P(x)\big) \quad \Leftrightarrow \quad \exists x\,\neg P(x) \tag{F.14}$$

One counterexample negates a universal statement.

Worked reading. To claim that every local minimum of a convex function is global, one must state the convexity and domain assumptions. A single nonconvex counterexample shows why the qualifier matters.

Common traps.

- Reversing an implication
- Treating absence of evidence as proof of impossibility
- Hiding assumptions inside notation

Where it reappears. Feasibility, optimality conditions, verification, complexity, rights constraints, and category theory throughout Chapters 1–7 and 20.

Sources. [F4]

Part F.2 · Linear structures and geometry

Unit 8 · Vectors and linear combinations

Prerequisites. Part F.1.

Why it matters in this volume. The principal decision object in continuous optimization is often a vector: a jointly chosen collection whose components interact.

A vector can represent displacement, features, allocations, states, gradients, or coefficients. Addition combines compatible vectors; scalar multiplication changes magnitude and possibly direction. A linear combination forms a weighted sum. Span describes everything reachable by linear combinations; independence means no listed direction is redundant.

Coordinates describe a vector relative to a basis. Changing basis changes coordinates but not the underlying vector. This distinction is essential when variables are rescaled or represented in different units. Linear structure lets local behavior be summarized by matrices and covectors.

$$v = \sum_{i=1}^{n} \alpha_i \, b_i \tag{F.15}$$

Coordinates α_i express v in basis vectors b_i.

$$\alpha u + \beta v \in V \tag{F.16}$$

A vector space is closed under linear combinations.

Worked reading. A portfolio weight vector and a physical-force vector obey the same algebra but different semantics. The analogy preserves linear combination; it does not identify risk with force.

Common traps.

- Adding vectors with incompatible meanings or units
- Confusing a vector with its coordinate column
- Assuming a long list of features is linearly independent

Where it reappears. Chapters 4–7, machine learning, signal processing, control, structural design, finance, and the worked translations.

Sources. [F5]

Unit 9 · Matrices, linear maps, and systems of equations

Prerequisites. Unit 8.

Why it matters in this volume. Matrices encode coupled linear relationships, transformations, constraints, dynamics, covariance, and curvature approximations.

We can consider a matrix as a linear map. Multiplication by a vector forms a weighted combination of its columns. The equation Ax=b asks which input maps to b. Rank gives the dimension of the image. A square singular matrix maps some distinct inputs to the same output and therefore has no ordinary inverse.

Matrix multiplication is composition: AB applies B first and A second. Order therefore matters. Transpose exchanges input and output roles under the Euclidean inner product. Eigenvectors preserve direction under a transformation; singular values measure stretching without requiring a square or symmetric matrix.

$$Ax = b \tag{F.17}$$

The columns of A combine with coefficients x to produce b.

$$A(Bx) = (AB)x \tag{F.18}$$

Matrix multiplication represents composition of linear maps.

$$Av = \lambda v \text{ for some } v \neq 0 \tag{F.19}$$

An eigenvector is a nonzero vector that keeps its direction and is scaled by λ.

Worked reading. If constraints are nearly linearly dependent, the system may be solvable yet sensitive. An inverse printed by software is not evidence of a well-conditioned model; residuals and condition estimates must be examined.

Common traps.

- Multiplying matrices in the wrong order
- Assuming every square matrix is invertible
- Using explicit inversion when solving a system is safer and cheaper

Where it reappears. Linear, quadratic, conic and semidefinite optimization; Kalman filtering; control; machine learning; and hardware/system models.

Sources. [F5]

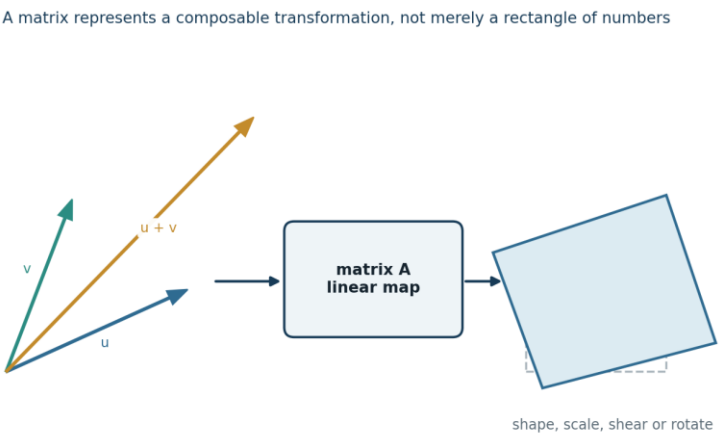


Figure F.4. Vectors, addition, and a matrix viewed as a composable linear transformation.

Unit 10 · Inner products, norms, distance, and projection

Prerequisites. Units 8–9.

Why it matters in this volume. Optimization needs a notion of size, similarity, direction, and nearest feasible point. Those notions are chosen mathematical structure, not automatic facts.

An inner product measures alignment and induces a norm. A norm measures magnitude and obeys positivity, homogeneity, and the triangle inequality. A metric measures distance. Different norms encode different geometries: ℓ_1 favors axis-aligned sparsity, ℓ_2 is rotationally symmetric, and ℓ_∞ measures the largest absolute component.

Projection selects a nearest point in a set under a chosen norm. In Euclidean space, projection onto a nonempty closed convex set exists and is unique. Least squares provides an example: the residual is orthogonal to the column space of the model matrix. This connects approximation with geometry.

$$\langle x, y \rangle = x^{\mathsf{T}} y, \qquad \| x \|_2 = \sqrt{x^{\mathsf{T}} x} \tag{F.20}$$

The Euclidean norm is induced by the standard inner product.

$$\Pi_C(v) = \arg\min_{x \in C} \| x - v \|_2 \tag{F.21}$$

Projection chooses the nearest feasible point.

Worked reading. Standardizing one feature but not another changes Euclidean distance and therefore nearest neighbors, gradients, and regularization. Geometry must be justified by the meaning and scale of the variables.

Common traps.

- Calling every dissimilarity a metric
- Forgetting which norm defines ‘steepest’
- Assuming projection onto a nonconvex set is unique

Where it reappears. Gradient methods, regularization, signal and image recovery, clustering, transport, trust regions, and category-aware interfaces.

Sources. [F5; F10]

Unit 11 · Graphs, networks, and combinatorics

Prerequisites. Part F.1 and Unit 8.

Why it matters in this volume. Schedules, routes, dependencies, causal structures, circuits, social interactions, narratives, and category diagrams can be represented as nodes and relations.

A graph consists of vertices and edges. An edge may have a direction to represent an asymmetric relation, or a weight to represent cost, capacity, probability, or strength. A path follows connected edges. A tree is a connected graph without cycles. Directed acyclic graphs can represent partial orders, as in scheduling and causal models.

Combinatorics counts structured possibilities. The multiplication principle counts independent stages; combinations ignore order; permutations retain it. Exponential growth in combinations explains why discrete optimization cannot normally enumerate every candidate.

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{F.22}$$

The binomial coefficient counts k-element subsets.

$$\text{cost}(P) = \sum_{e \in P} c_e \tag{F.23}$$

A path cost is the sum of its edge costs when costs are additive.

Worked reading. Choosing 20 items from 100 already yields more than 5×10^{20} subsets. A compact problem statement can therefore describe a search space far beyond exhaustive inspection.

Common traps.

- Confusing a graph drawing with the abstract graph
- Treating correlation networks as causal graphs
- Assuming edge costs remain additive when congestion couples paths

Where it reappears. Chapters 2, 5, 7–11, 13, 16, and the library and semiconductor cases and the routing and narrative illustrations.

Sources. [F6; F7]

Part F.3 · Change, approximation, and continuous models

Unit 12 · Limits and continuity

Prerequisites. Functions, algebra, and sequences.

Why it matters in this volume. Derivatives, integrals, convergence, and numerical error all depend on controlled limiting behavior.

A limit describes what a function approaches as its input approaches a point; it need not equal the function's value there. Continuity means small input changes produce small output changes locally. In optimization, continuity helps establish existence of extrema on compact feasible sets, but does not ensure uniqueness or easy computation.

Convergence specifies a mode: variables, function values, distributions, or residuals may converge differently. Numerical sequences can appear stable while converging to the wrong result because of discretization or modeling error.

$$\lim_{x \to a} f(x) = L \tag{F.24}$$

Values of f(x) approach L as x approaches a.

$$x_k \to x^* \quad \Leftrightarrow \quad \forall \varepsilon > 0 \; \exists K \; \forall k \geq K \colon \| x_k - x^* \| < \varepsilon \tag{F.25}$$

Convergence eventually enters every requested tolerance.

Worked reading. A step-function tariff is discontinuous at its threshold. A smooth approximation may help a solver but changes the economics near the threshold. The approximation error must be treated as part of the model.

Common traps.

- Substituting before checking that a limit is well-defined
- Equating a flat numerical trace with mathematical convergence
- Assuming continuity implies differentiability

Where it reappears. Chapters 4, 7, 9–11, 17, and any discussion of convergence, discretization, or trust regions.

Sources. [F3]

Unit 13 · Derivatives and one-dimensional optimization

Prerequisites. Unit 12.

Why it matters in this volume. A derivative translates local change into a number and supplies the first tool for finding stationary candidates.

The derivative is the limit of a difference quotient. Its sign indicates local increase or decrease; its magnitude measures sensitivity in output units per input unit. At an interior differentiable local optimum, the derivative must be zero, but zero derivative also occurs at maxima, saddles in higher dimensions, and flat inflection points.

The chain rule differentiates compositions and therefore mirrors the structure of computational graphs. Product and quotient rules describe how sensitivities combine. Nondifferentiable points require one-sided derivatives or generalized derivatives.

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{F.26}$$

The derivative is a limiting local slope.

$$\frac{d}{dx} g\big(f(x)\big) = g'\big(f(x)\big) f'(x) \tag{F.27}$$

The chain rule propagates local sensitivity through composition.

Worked reading. For f(x)=(x−3)²+2, f′(x)=2(x−3), so x=3 is stationary. Since f″(3)=2>0, it is a strict local minimum; the quadratic form also shows it is global.

Common traps.

- Declaring every stationary point a minimum
- Ignoring derivative units
- Using finite differences with a step so small that subtraction error dominates

Where it reappears. Chapter 4, gradient verification, response surfaces, engineering sensitivities, learning, and control.

Sources. [F3; F9]

Unit 14 · Gradients, Jacobians, Hessians, and covectors

Prerequisites. Units 8–13.

Why it matters in this volume. Multivariable optimization needs derivatives that distinguish scalar outputs, vector outputs, and curvature.

The differential of a scalar function is a covector: it consumes a direction and returns the first-order change. An inner product identifies that covector with a gradient vector. A Jacobian collects derivatives of a vector-valued map. A Hessian collects second derivatives of a scalar function and approximates curvature.

The gradient depends on geometry. Under the Euclidean inner product, the negative gradient is steepest descent per unit ℓ_2 length. Changing variables or preconditioning changes the represented direction. Positive-definite Hessian at a stationary point is sufficient for a strict local minimum; semidefinite curvature at one point is inconclusive.

$$f(x + d) = f(x) + \nabla f(x)^{\mathsf{T}} d + o(\| d \|) \tag{F.28}$$

The gradient represents the first-order differential in Euclidean coordinates.

$$H_f(x) = \nabla^2 f(x) = \left[\frac{\partial^2 f}{\partial x_i \, \partial x_j} \right] \tag{F.29}$$

The Hessian records local second-order interaction.

Worked reading. For a two-variable objective, contour lines show equal values. The gradient is normal to a smooth contour. A direction tangent to the contour has zero first-order change even though second-order change may remain.

Common traps.

- Treating the gradient as coordinate-free
- Confusing a Jacobian with a Hessian
- Inferring global convexity from one positive-semidefinite Hessian

Where it reappears. Chapter 4, KKT geometry, automatic differentiation, machine learning, structures, control, and physical inverse design.

Sources. [F3; F5; F9]

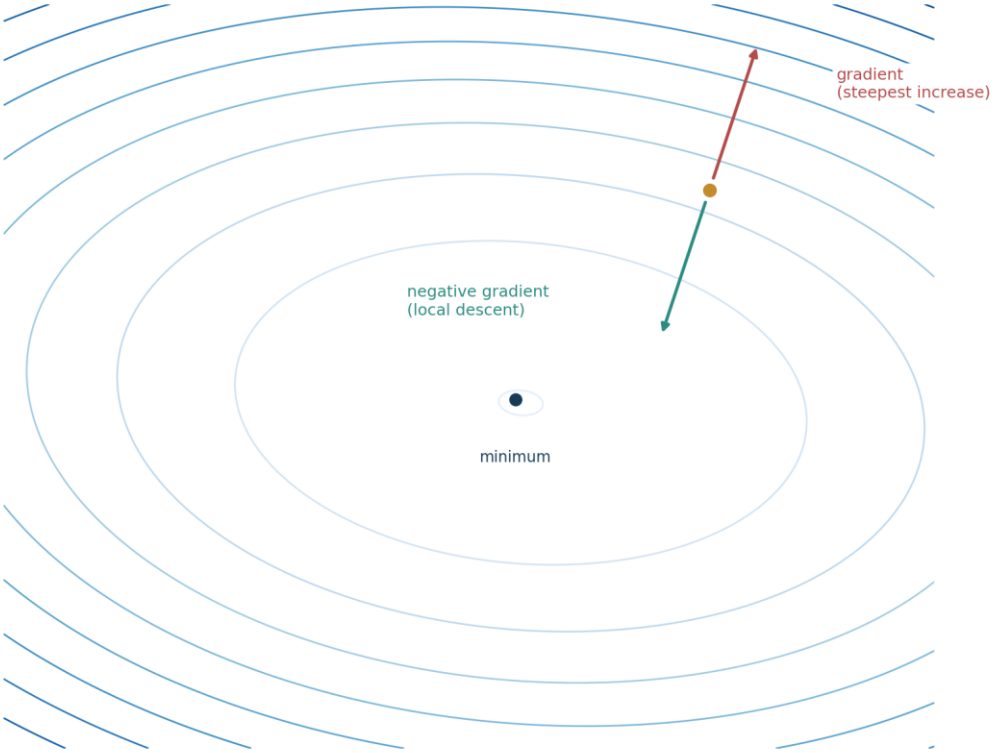


Figure F.5. Objective contours with a gradient, a descent direction, and the minimum.

Unit 15 · Taylor approximation, curvature, and conditioning

Prerequisites. Unit 14.

Why it matters in this volume. Solvers predict improvement from local models, and sensitivity analysis predicts how solutions move when data change.

Taylor expansion approximates a smooth function by polynomial information at a reference point. First order gives a tangent model; second order adds curvature. The remainder states

what has been omitted. A trust region restricts steps to a neighborhood where the approximation has evidence.

Conditioning belongs to the problem: it measures response to perturbations. Stability belongs to the algorithm: it measures whether the computation amplifies error beyond the problem's inherent sensitivity. Scaling can improve numerical behavior without changing the underlying decision, provided transformations are tracked correctly.

$$f(x+d) \approx f(x) + \nabla f(x)^{\mathsf{T}} d + \frac{1}{2} d^{\mathsf{T}} H_f(x) d \tag{F.30}$$

A local quadratic model combines slope and curvature.

$$\kappa(A) = \| A \| \, \| A^{-1} \| \tag{F.31}$$

For invertible A and an induced matrix norm, this condition number bounds the worst-case relative amplification of perturbations.

Worked reading. A narrow curved valley can have a modest objective value error but large variable uncertainty along its flat direction. Reporting only the objective hides practical instability.

Common traps.

- Treating an approximation sign as equality
- Calling an ill-conditioned answer inaccurate without separating data sensitivity from solver error
- Changing scale without transforming tolerances and units

Where it reappears. Trust regions, scaling, residuals, digital twins, sensitivity, inverse design, and deployment envelopes throughout Chapters 4, 10, 11, 17–19.

Sources. [F5; F9]

Unit 16 · Integration and differential equations

Prerequisites. Units 6 and 12–14.

Why it matters in this volume. Accumulated cost, energy, exposure, probability, work, inventory, and state evolution are expressed through integrals and differential equations.

A definite integral accumulates infinitesimal contributions and can be understood as a limit of sums. The fundamental theorem connects accumulated area with differentiation. An ordinary differential equation specifies a rate of change; an initial condition selects a trajectory among possible solutions.

Numerical integration and time stepping introduce discretization error. Smaller steps usually reduce truncation error but increase computation and can amplify roundoff effects. Stiffness can force very small explicit steps or motivate implicit methods. Conservation and units provide independent checks.

$$\int_a^b f\,(t)\,dt = \lim_{n\to\infty} \sum_{i=1}^{n} f\,(t_i^*)\Delta t \tag{F.32}$$

Integration is the limit of increasingly fine accumulated sums.

$$\dot{x}(t) = F(x(t), u(t), t), \qquad x(0) = x_0 \tag{F.33}$$

A controlled differential equation defines state evolution from an initial condition.

Worked reading. If power P(t) is measured in watts, energy over an interval is ∫P(t)dt in joules. Summing power readings without multiplying by the sample interval produces the wrong unit and usually the wrong answer.

Common traps.

- Forgetting the constant of integration in indefinite integration
- Assuming every differential equation has a unique global solution
- Ignoring numerical step-size and stiffness effects

Where it reappears. Energy, dose and exposure, control, climate, process systems, robotics, and dynamic worked translations.

Sources. [F3]

Unit 17 · Functionals and the calculus of variations

Prerequisites. Units 14–16.

Why it matters in this volume. When the decision is an entire curve, field, shape, signal, or policy, the objective is a function of a function—a functional.

In the calculus of variations, we perturb a candidate function by a small admissible function and examine the first-order change. Under the required regularity and endpoint conditions, stationarity gives the Euler–Lagrange equation. We need to specify the boundary conditions and admissible function space as part of the problem.

Discretize-then-optimize turns a functional problem into a finite vector problem. Optimize-then-discretize derives continuous optimality conditions first. The two routes need not agree at finite resolution; mesh refinement and adjoint checks compare them.

$$J[y] = \int_a^b L\left(t, y(t), y'(t)\right) dt \tag{F.34}$$

A functional assigns a number to an entire function y.

$$\frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial y'}\right) = 0 \tag{F.35}$$

The Euler–Lagrange equation is a stationarity condition for admissible path variations.

Worked reading. The shortest path in a curved geometry is not found by optimizing each point independently. The path must be varied as a whole while respecting endpoints and geometry.

Common traps.

- Varying a path outside the admissible class
- Omitting boundary conditions

- Assuming discretization preserves every continuous constraint

Where it reappears. Chapter 4, optimal control, geodesics, shape and topology optimization, signal processing, and continuous-time policy design.

Sources. [F14]

Part F.4 · Probability, statistics, and evidence

Unit 18 · Probability, events, and random variables

Prerequisites. Sets, functions, fractions, and sums.

Why it matters in this volume. Uncertainty in observations, parameters, outcomes, environments, and other agents is represented probabilistically in many—but not all—optimization problems.

A probability model is a probability space (Ω, $\mathcal{F}$, P): Ω is the sample space, $\mathcal{F}$ is a σ-algebra of events, and P maps each event to [0, 1], with P(Ω)=1 and countable additivity for every countable pairwise-disjoint family of events. The complement rule $P(A^c)=1-P(A)$ follows from these axioms. A real-valued random variable is a measurable function $X:(\Omega,\mathcal{F})\to(\mathbb{R},\mathcal{B}(\mathbb{R}))$. Its distribution describes probability over values, not a single uncertain value hidden in a box.

Discrete distributions use probability masses; continuous distributions use densities whose integrals give probabilities. A density value can exceed one because it is not itself an event probability. Independence is a structural factorization claim, stronger than zero correlation.

$$\begin{gathered} P(\Omega)=1, \quad P(A^c)=1-P(A), \\ P(\bigcup_{i=1}^{\infty} A_i)=\sum_{i=1}^{\infty} P(A_i) \quad \text{if } A_i \cap A_j=\varnothing \ (i \neq j). \end{gathered} \tag{F.36}$$

Normalization and countable additivity imply the complement rule for every event A.

$$P(X \in A)=\int_A p_X(x)\,dx \tag{F.37}$$

A continuous probability is area under the density over an event.

Worked reading. A 1% component failure probability does not imply a 1% system failure probability. Dependence, common causes, redundancy, and system logic determine how component events compose.

Common traps.

- Treating a density as a probability
- Assuming uncorrelated means independent
- Assigning probabilities without stating the reference class or time horizon

Where it reappears. Chapters 3, 6, 9–13, 17–19: uncertainty, learning, medicine, reliability, finance, climate, and evidence.

Sources. [F8]

Unit 19 · Expectation, variance, covariance, and correlation

Prerequisites. Unit 18 and algebra.

Why it matters in this volume. Objectives under uncertainty often compare expected consequences, dispersion, covariance, or tail behavior.

Expectation is a probability-weighted average. Variance measures expected squared deviation from the mean. Covariance measures joint linear variation; correlation standardizes covariance to a dimensionless number between −1 and 1 when variances are positive.

Expectation is linear even without independence. Variance of a sum includes covariance terms. Mean and variance do not determine every distribution, and two distributions with the same moments can have different tails and decision implications.

$$\mathbb{E}[X] = \sum_x x\ P(X = x) \quad \text{or} \quad \int x p_X(x)\, dx \tag{F.38}$$

Expectation averages values using their probability law.

$$\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}X)^2] \tag{F.39}$$

Variance is mean squared deviation from the mean.

$$\mathrm{Var}(X + Y) = \mathrm{Var}(X) + \mathrm{Var}(Y) + 2\mathrm{Cov}(X, Y) \tag{F.40}$$

Dependence contributes a covariance term.

Worked reading. Diversification works when components do not move perfectly together. Simply adding individual variances ignores covariance and can understate or overstate system risk.

Common traps.

- Interpreting expectation as the outcome that will occur
- Using correlation as evidence of causation
- Comparing variances of quantities with different units without normalization

Where it reappears. Risk, portfolios, learning, reliability, public health, and stochastic optimization in Chapters 6, 9, 12, 13, 18, and 19.

Sources. [F8; F12]

Unit 20 · Conditional probability, Bayes' rule, and likelihood

Prerequisites. Units 18–19.

Why it matters in this volume. Diagnosis, filtering, calibration, learning, and adaptive decisions update beliefs after evidence arrives.

Conditional probability restricts attention to cases where the condition holds. Bayes' rule reverses the direction of conditioning by combining a prior probability with a likelihood and normalizing by the evidence. A likelihood is the observed-data probability viewed as a function of a parameter; it is not a probability distribution over the parameter until combined with a prior and normalized.

Base rates matter. A highly sensitive test can still have a low positive predictive value when the condition is rare. Sequential updating is coherent when the data model and conditional-independence assumptions are correct.

$$P(H \mid D) = \frac{P(D|H)P(H)}{P(D)} \tag{F.41}$$

Posterior equals likelihood times prior divided by evidence.

$$P(D) = \sum_h P\,(D \mid H = h)P(H = h) \tag{F.42}$$

The evidence averages the likelihood over hypotheses.

Worked reading. With prevalence 1%, sensitivity 90%, and false-positive rate 5%, a positive test does not imply 90% probability of disease. Bayes' rule combines all three quantities and yields about 15.4% under the simplified assumptions.

Common traps.

- Ignoring the base rate
- Swapping P(H|D) and P(D|H)
- Calling a likelihood a posterior probability

Where it reappears. Clinical thresholds, Bayesian optimization, state estimation, machine learning, security, and adaptive control.

Sources. [F8]

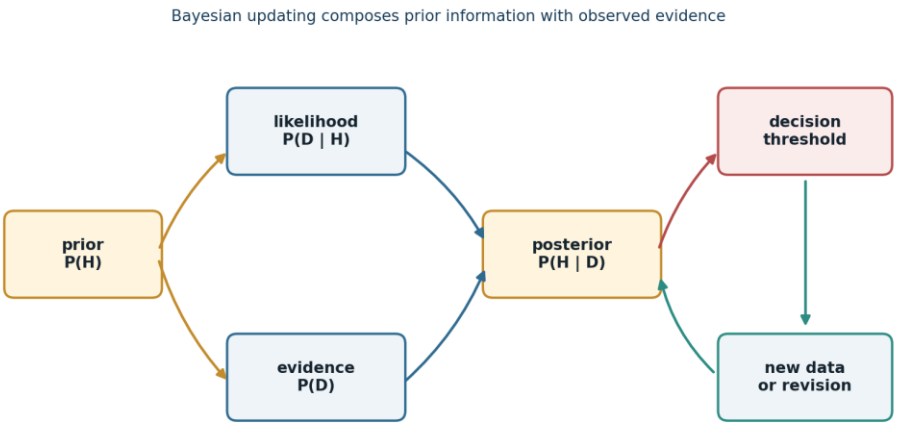


Figure F.6. Bayesian updating from prior and likelihood to posterior and revised decisions.

Unit 21 · Sampling, estimation, uncertainty intervals, and causality

Uncertainty quantification describes what observations and assumptions permit us to say about estimation error and plausible parameter values. The intervals in this unit are examples; their interpretation depends on the statistical model.

Prerequisites. Units 18–20.

Why it matters in this volume. Optimization consumes estimates. If evidence is biased, confounded, shifted, or selectively measured, a precisely solved model can optimize the wrong world.

The population is the collection about which we want to make a claim, and the sample contains the observations available to us. An estimator maps sample data to an estimate. Bias, variance, and consistency describe its behavior under repeated sampling. A confidence interval concerns the coverage of a procedure under assumptions. It is not generally a posterior probability statement about a fixed parameter.

Causal inference asks what would change under intervention. Randomization makes assignment independent of potential outcomes by design; in a finite sample, covariates are balanced only in expectation. Observational identification requires explicit assumptions about confounding, selection, measurement, interference, and positivity. Prediction alone does not answer an intervention question.

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{F.43}$$

The sample mean estimates a population mean under a sampling model.

$$\mathrm{MSE}(\hat{\theta}) = \mathrm{Var}(\hat{\theta}) + \mathrm{Bias}(\hat{\theta})^2 \tag{F.44}$$

Mean squared error decomposes into variance and squared bias.

Worked reading. A hospital risk model may predict readmission accurately because treatment is encoded in the data. Using its association as a causal treatment effect can recommend the opposite intervention. The estimand must be specified before the estimator.

Common traps.

- Treating a convenience sample as representative
- Interpreting statistical significance as practical importance
- Optimizing a predictor as if it were a causal response surface

Where it reappears. Measurement and evidence in Chapter 3; ML in Chapter 9; medicine and policy in Chapters 12–14; experiments and digital twins in Chapter 17.

Sources. [F8; F11; F12]

Unit 22 · Stochastic processes and Markov models

Prerequisites. Sequences, probability, and matrices.

Why it matters in this volume. Queues, reliability, control, finance, disease progression, ecology, and reinforcement learning evolve through uncertain time.

A stochastic process is an indexed family of random variables. A Markov process makes the next-state distribution depend on the present state rather than the full recorded history, conditional on that state. The assumption is about the chosen state representation; omitted memory can make a process appear non-Markovian.

A transition matrix contains conditional probabilities and maps a state distribution forward. Stationary distributions are unchanged by the transition map, but stationarity need not imply rapid mixing or independence over time.

$$P(X_{t+1} = j \mid X_t = i, X_{t-1}, \dots) = P(X_{t+1} = j \mid X_t = i) \tag{F.45}$$

The chosen present state screens off earlier history.

$$\mu_{t+1}^{\mathsf{T}} = \mu_t^{\mathsf{T}} P \tag{F.46}$$

A row distribution advances through the transition matrix.

Worked reading. If machine degradation depends on cumulative thermal exposure, ‘current temperature’ alone is not a Markov state. Adding a sufficient damage variable may restore the property.

Common traps.

- Assuming every time series is Markov
- Confusing a stationary distribution with a static process
- Ignoring uncertainty dependence across time

Where it reappears. Chapters 6–13, particularly queues, filtering, reliability, control, reinforcement learning, health pathways, and finance.

Sources. [F13]

Part F.5 · Optimization foundations and methods

Unit 23 · Formulating an optimization problem

Prerequisites. Parts F.1–F.4.

Why it matters in this volume. The mathematical grammar of the volume begins by separating choices, admissibility, evaluation, uncertainty, and authority.

Decision variables specify what the procedure can choose, and the feasible set specifies the admissible choices. An objective or preference relation compares their consequences. Parameters and data remain fixed for that decision unless the system is authorized to change them. Argmin denotes the set of minimizers. When a minimum is attained, min denotes its value.

A formulation is a model, not the world. Equivalent algebraic forms may differ numerically. Multiple objectives may be preserved rather than prematurely aggregated. Rights, safety, and logical consistency may belong as hard constraints or decision rights rather than soft penalties.

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x) \tag{F.47}$$

x* is one minimizing choice in the feasible set.

$$p^* = \min_{x \in \mathcal{X}} f(x) \tag{F.48}$$

p* is the best attained objective value.

Worked reading. For staffing, x may be integer shift assignments, X the legal and coverage constraints, and f a cost-plus-service criterion. Historical staffing is a baseline, not automatically a constraint or objective.

Common traps.

- Optimizing variables the decision maker cannot control
- Conflating argmin with minimum value
- Hiding stakeholder trade-offs in an unexplained scalar score

Where it reappears. Bird’s-eye view, Chapters 1–3, all mathematical chapters, and every worked translation.

Sources. [F9; F15]

Unit 24 · Convexity and supporting geometry

Prerequisites. Functions, vectors, inequalities, and derivatives.

Why it matters in this volume. Convexity is the principal local-to-global structure used in the volume.

A convex set contains every line segment between its points. A convex function lies below its chords. Minimizing a convex function over a convex set has no non-global local minima. Strict convexity can imply uniqueness; strong convexity provides quantitative curvature.

Convexity is preserved by nonnegative sums, affine composition, pointwise maxima, and other composition rules. Recognizing structure is often more valuable than choosing a fashionable algorithm. Convexity of an objective alone is insufficient if the feasible set is nonconvex.

$$\theta x + (1-\theta)y \in C, \qquad 0 \le \theta \le 1 \tag{F.49}$$

Every segment between feasible points stays feasible.

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta)f(y), \quad 0 \le \theta \le 1. \tag{F.50}$$

The function lies below the chord joining two graph points.

Worked reading. The squared Euclidean distance to a fixed point is convex. A binary restriction on the variables makes the feasible set discrete and therefore changes the global structure even if the formula is unchanged.

Common traps.

- Calling a problem convex because the objective looks bowl-shaped
- Assuming strict convexity guarantees existence
- Inferring convexity from a small plot

Where it reappears. Chapter 4, relaxations in Chapter 5, risk and multiobjective formulations in Chapter 6, and numerous domain models.

Sources. [F9; F15]

Unit 25 · Lagrange multipliers and KKT conditions

Prerequisites. Units 14, 23, and 24.

Why it matters in this volume. Constraints change stationarity from ‘zero gradient’ to a balance between objective and active-constraint normals.

For minimization with inequality constraints $g(x) \le 0$ and equalities $h(x) = 0$, inequality multipliers satisfy $\lambda \ge 0$ and equality multipliers are unrestricted. The Lagrangian combines the objective with weighted constraint functions. Multipliers measure the local objective value of relaxing appropriately scaled constraints. KKT conditions combine primal feasibility, dual feasibility, stationarity, and complementary slackness.

Constraint qualifications are required for necessity claims. Under convexity, a feasible primal-dual point satisfying KKT conditions certifies global optimality. Outside convexity, KKT generally provides local stationarity evidence, not a global certificate.

$$\mathcal{L}(x, \lambda, \nu) = f(x) + \lambda^{\mathsf{T}} g(x) + \nu^{\mathsf{T}} h(x) \tag{F.51}$$

The Lagrangian attaches multipliers to inequality and equality constraints.

$$\nabla_x \mathcal{L}(x^*, \lambda^*, \nu^*) = 0, \quad \lambda_i^* g_i(x^*) = 0 \tag{F.52}$$

Stationarity and complementary slackness form part of KKT.

Worked reading. An inactive capacity constraint has slack and therefore zero multiplier under complementary slackness. A positive multiplier indicates local value of additional capacity only within the model and scaling used.

Common traps.

- Reading multipliers as universal prices
- Ignoring constraint qualifications
- Treating KKT satisfaction as a global certificate for a nonconvex model

Where it reappears. Chapter 4, constrained engineering, allocation, policy, control, and worked translations.

Sources. [F9; F15; F16]

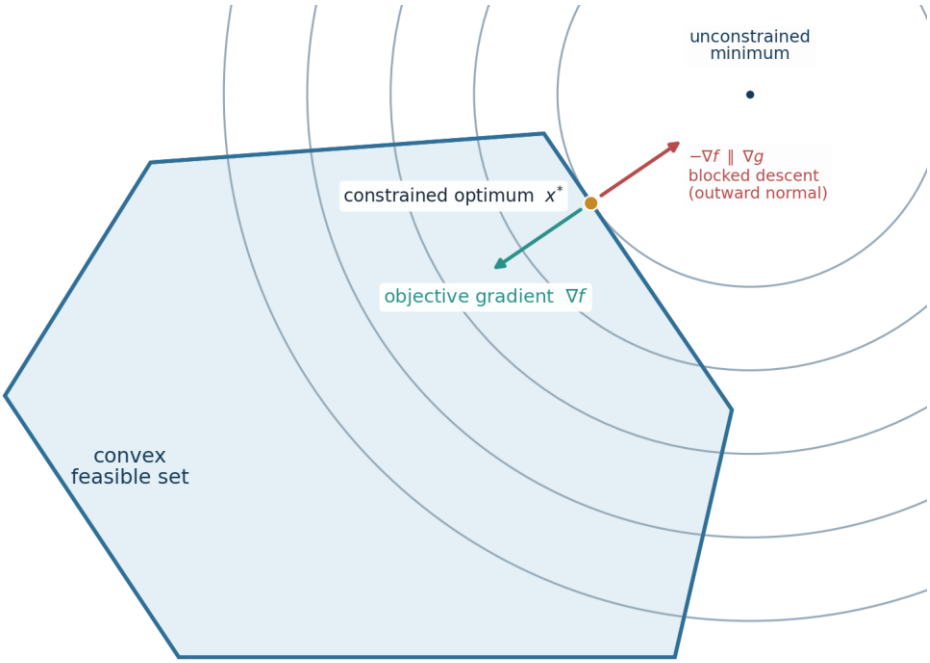


Figure F.7. Convex feasible geometry and the KKT balance at an active boundary.

Unit 26 · Duality, shadow values, and sensitivity

Prerequisites. Unit 25.

Why it matters in this volume. Duality supplies bounds, decompositions, interpretations, and certificates.

With the sign convention $g(x) \leq 0$, $h(x) = 0$, and $\lambda \geq 0$ from Unit 25, the dual function is the infimum of the Lagrangian over primal variables and therefore gives a lower bound for a minimization problem. Maximizing that bound forms the dual problem. Weak duality always orders dual and primal values. Strong duality requires hypotheses such as convexity plus an appropriate regularity condition.

Sensitivity connects small parameter changes to optimal value changes. Dual multipliers often provide first-order sensitivities, but only locally and under regularity. Near degeneracy, active-set changes can make sensitivity nonsmooth or set-valued.

$$q(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu), \qquad \lambda \geq 0 \tag{F.53}$$

The dual function is a lower bound indexed by feasible multipliers.

$$d^* \leq p^* \tag{F.54}$$

Weak duality places the best dual bound below the primal optimum.

Worked reading. If a resource multiplier is 12 objective units per resource unit, one additional unit is locally worth about 12—provided the active set, model, and units remain stable. It is not a forecast for arbitrarily large expansion.

Common traps.

- Assuming zero duality gap without checking assumptions
- Extrapolating a local shadow value
- Interpreting a multiplier without its units and constraint scaling

Where it reappears. Chapter 4; linear and conic optimization; markets; capacity planning; resource allocation; and certificates.

Sources. [F9; F15; F16]

Unit 27 · Linear, quadratic, conic, and semidefinite forms

Prerequisites. Algebra, matrices, convexity, and duality.

Why it matters in this volume. Many domain models compile into standard structures with mature algorithms and interpretable certificates.

Linear programming uses affine objectives and constraints. Quadratic programming adds a quadratic objective; positive-semidefinite curvature preserves convexity. Conic optimization constrains affine expressions to a cone, unifying nonnegative, second-order, and semidefinite forms. Semidefinite programming optimizes over symmetric positive-semidefinite matrices.

Standard form is a translation target, not a claim that the world is linear. Piecewise-linear approximation, lifting, and relaxation may expose structure while adding variables or approximation error.

$$\min_x c^{\mathrm{T}} x \quad \text{s.t.} \quad Ax = b,\ x \geq 0 \tag{F.55}$$

A linear program has an affine objective and affine feasibility.

$$\min_x \frac{1}{2} x^\top Q x + c^\top x \quad \text{s.t.} \quad Ax \le b \tag{F.56}$$

A quadratic program is convex when Q is positive semidefinite.

$$X \succcurlyeq 0 \tag{F.57}$$

A symmetric matrix X is positive semidefinite.

Worked reading. A covariance matrix must be positive semidefinite because every linear combination has nonnegative variance. That same cone appears in control, relaxations, and experimental design.

Common traps.

- Assuming every quadratic is convex
- Using a relaxation without measuring recovery error
- Treating standard-form coefficients as unitless

Where it reappears. Chapter 5 and applications in control, covariance, hardware, structures, energy, manufacturing, finance, and learning.

Sources. [F9; F15]

Unit 28 · Discrete, integer, and combinatorial optimization

Prerequisites. Unit 11 and formulation.

Why it matters in this volume. Assignments, schedules, routes, layouts, logic, selections, and program transformations involve indivisible choices.

Integer variables encode counts or categories; binary variables encode yes/no choices. Relaxing integrality creates a continuous bound but may produce an unusable fractional decision. Branch-and-bound partitions the search space while using bounds to prune regions. Cutting planes remove fractional solutions without removing integer-feasible ones.

Constraint programming emphasizes propagation over domains; SAT/SMT emphasizes logical satisfiability; dynamic programming exploits repeated substructure. Representation can change practical difficulty without changing the underlying decision.

$$x_i \in \{0,1\} \tag{F.58}$$

A binary variable selects or excludes item i.

$$\sum_i w_i \, x_i \le B \tag{F.59}$$

Selected items must respect a capacity or budget.

Worked reading. A fractional solution selecting 0.4 of a bridge project may be a useful lower bound but is not an implementable plan. Rounding can violate budget, safety, or coupling constraints, so recovery must be validated.

Common traps.

- Rounding each variable independently

- Calling a heuristic solution optimal without a bound
- Using an integer variable when a continuous proportion is physically realizable

Where it reappears. Chapter 5; cloud scheduling; compiler and circuit design; manufacturing; transport; public allocation; libraries; narrative and game structures.

Sources. [F6; F7]

Unit 29 · Gradient, Newton, proximal, and trust-region algorithms

Prerequisites. Units 13–17 and 23–27.

Why it matters in this volume. Algorithms turn mathematical structure and information access into candidate-generating procedures.

Gradient descent follows first-order local information. Newton's method solves a local quadratic model using curvature. Quasi-Newton methods estimate curvature from gradient changes. Proximal methods preserve nonsmooth structure such as norms and indicator functions. Trust-region methods restrict steps to where a model is credible.

A stopping rule should report residuals, gap, iteration or evaluation limits, and the kind of claim supported. Automatic differentiation differentiates implemented code, not intended mathematics; directional checks compare them.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) \tag{F.60}$$

Gradient descent moves opposite the local Euclidean gradient.

$$x_{k+1} = x_k - H(x_k)^{-1} \nabla f(x_k) \tag{F.61}$$

The inverse requires $H(x_k)$ to be nonsingular. A positive-definite Hessian gives a descent direction when the gradient is nonzero; a line search or trust region controls the accepted step.

Newton's step solves the local quadratic stationarity equation.

$$\operatorname{prox}_{\alpha g}(v) = \arg\min_x \left(g(x) + \frac{\|x - v\|_2^2}{2\alpha} \right) \tag{F.62}$$

A proximal map balances structure g with distance from v.

Worked reading. A method that needs fewer iterations may still use more wall time or energy if each iteration solves a large physical simulation. Evaluation cost, memory traffic, parallelism, and synchronization belong to the algorithmic cost model.

Common traps.

- Comparing algorithms only by iteration count
- Accepting a solver status without residuals
- Differentiating through discontinuities without specifying the derivative notion

Where it reappears. Chapter 4, algorithm stacks in Chapters 7–10, digital twins, inverse design, and worked translations.

Sources. [F9; F10]

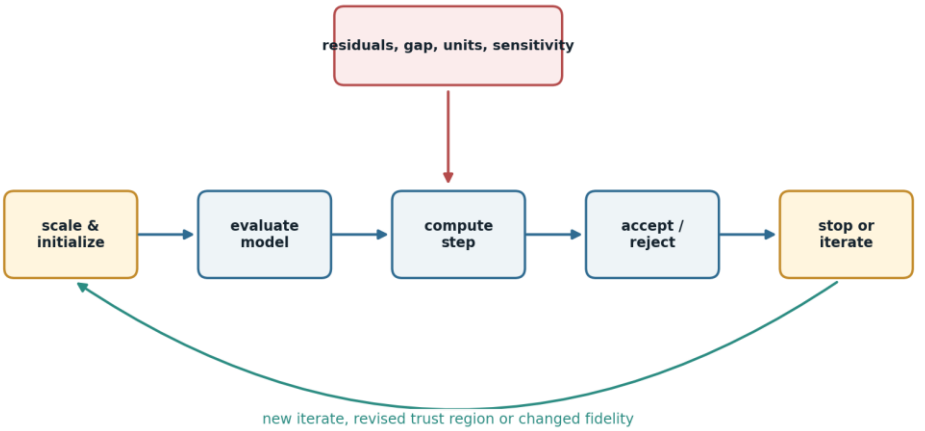


Figure F.8. A numerical optimization loop with scaling, model evaluation, steps, acceptance, and explicit evidence.

Unit 30 · Global, derivative-free, and surrogate optimization

Prerequisites. Functions, probability, and formulation.

Why it matters in this volume. A simulator, experiment, benchmark, or human comparison may return observations without usable derivatives. The search method must fit this information interface. Local convergence, statistical improvement, and a certified global bound are different possible conclusions.

Direct-search and interpolation methods use evaluated values to choose local steps. Randomized zeroth-order methods estimate directions, and population methods generate and adapt groups of candidates. Bayesian optimization selects evaluations using a probabilistic surrogate and an acquisition rule. One problem may combine expensive evaluations, mixed variables, constraints, noise, several fidelities, pending runs, and preferences. Chapter 5 describes the corresponding variants and assumptions.

A surrogate describes a modeled response. Its uncertainty requires appropriate assumptions and checking. Global optimization methods may instead use valid lower and upper bounds to exclude regions. Best-observed response, surrogate confidence, stationarity, and a global optimality gap should therefore be reported separately.

$$x_{n+1} \in \arg\max_{x \in \mathcal{X}} a_n(x) \tag{F.63}$$

An acquisition function chooses the next evaluation using the current surrogate.

$$L_k \le f^* \le U_k \tag{F.64}$$

Global methods narrow lower and upper bounds on the optimum.

Worked reading. Suppose each laboratory evaluation takes a day. A surrogate may help select which candidate to test next, while a lower-cost approximation may provide an additional fidelity. We still reserve resources to confirm the candidate at the target fidelity and distinguish failed evaluations from infeasible designs.

Common traps.

- Calling the best sampled point globally optimal without a valid bound or an applicable theorem.
- Trusting surrogate uncertainty or a low-fidelity ranking outside the conditions supporting the model.
- Reporting the most favorable noisy observation as confirmed performance, or omitting failed, repeated, and cancelled runs from the budget.

Where it reappears. Chapter 5, biomedical discovery, materials, manufacturing, creative search, experiments, and digital twins.

Sources. [F10]; Larson, Menickelly, and Wild 2019; Shahriari et al. 2016; Audet and Dennis 2006; Audet, Custódio, and Dennis 2008. Chapter 5 supplies the variant-specific references and service-tuning example.

Unit 31 · Multiple objectives and Pareto reasoning

Prerequisites. Functions, sets, and formulation.

Why it matters in this volume. Real decisions often preserve several values that cannot be reduced without a normative choice.

A point dominates another when it is no worse in every objective and better in at least one, with directions stated consistently. Nondominated points form a Pareto set; their objective images form a Pareto front. Scalarization converts a vector objective into one comparison but introduces weights, reference points, or constraints.

Weighted sums can miss nonconvex parts of a front. Normalization changes the meaning of weights. A knee is a region where small improvement in one objective requires large sacrifice in another, but knee identification itself depends on scale and geometry.

$$\min_{x \in \mathcal{X}} \big(f_1(x), \ldots, f_m(x)\big) \tag{F.65}$$

A vector objective preserves several criteria rather than imposing a total order.

$$\min_{x \in \mathcal{X}} \sum_{i=1}^{m} w_i f_i(x), \qquad w_i \geq 0 \tag{F.66}$$

A weighted sum is one explicit scalarization.

Worked reading. Power, performance, area, temperature, yield, and verification effort use different units. We can display their trade-offs on a Pareto front and use stakeholder discussion to select an alternative. A transparent Pareto front is often more honest than a hidden composite score because it keeps the reasons for selection available for examination.

Common traps.

- Calling a nondominated point uniquely optimal
- Comparing weights before normalizing objectives
- Treating preference elicitation as purely technical

Where it reappears. Chapter 6 and nearly every domain chapter, especially hardware, health allocation, policy, sustainability, art, music, and literature.

Sources. [F18]

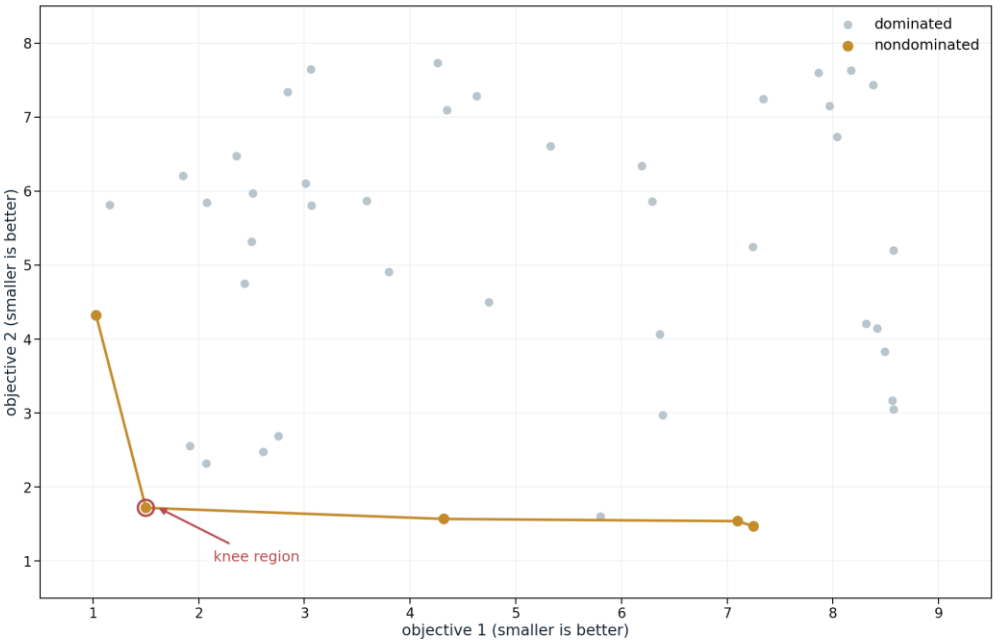


Figure F.9. Dominated and nondominated alternatives on a two-objective Pareto front.

Part F.6 · Uncertainty, time, and strategic interaction

Unit 32 · Expected value, risk, robustness, and distributional ambiguity

Prerequisites. Probability, optimization, and convexity.

Why it matters in this volume. Decision makers may care about average performance, tails, worst cases, adaptability, or ambiguity about the probability law.

Stochastic optimization minimizes an expectation or another distributional functional. Risk measures such as CVaR emphasize poor outcomes. Robust optimization protects against every parameter in an uncertainty set. Distributionally robust optimization protects against every probability law in an ambiguity set. These are different attitudes and should not be conflated.

Worst-case protection can be conservative when the set is too broad and unsafe when it omits plausible joint extremes. Scenario methods approximate distributions or uncertainty sets and require sampling and confidence statements.

$$\min_{x} \mathbb{E}[f(x,\xi)] \tag{F.67}$$

Expected-value optimization averages over uncertain ξ.

$$\mathrm{CVaR}_{\alpha}(L) = \min_{\eta}\left(\eta + \frac{1}{1-\alpha}\mathbb{E}[(L-\eta)_{+}]\right) \tag{F.68}$$

CVaR averages losses beyond an optimized threshold.

$$\min_{x}\max_{\xi\in\mathcal{U}} f(x,\xi) \tag{F.69}$$

Robust optimization protects against every modeled uncertainty in U.

Worked reading. Two designs can have equal expected cost while one has a small probability of catastrophic loss. Tail-sensitive or hard-safety constraints expose the difference that the mean suppresses.

Common traps.

- Calling a robust solution risk-free
- Choosing an uncertainty set without calibration
- Treating CVaR as the same as maximum loss

Where it reappears. Chapter 6; safety, security, medicine, finance, energy, climate, manufacturing, and every worked translation.

Sources. [F17; F19; F20]

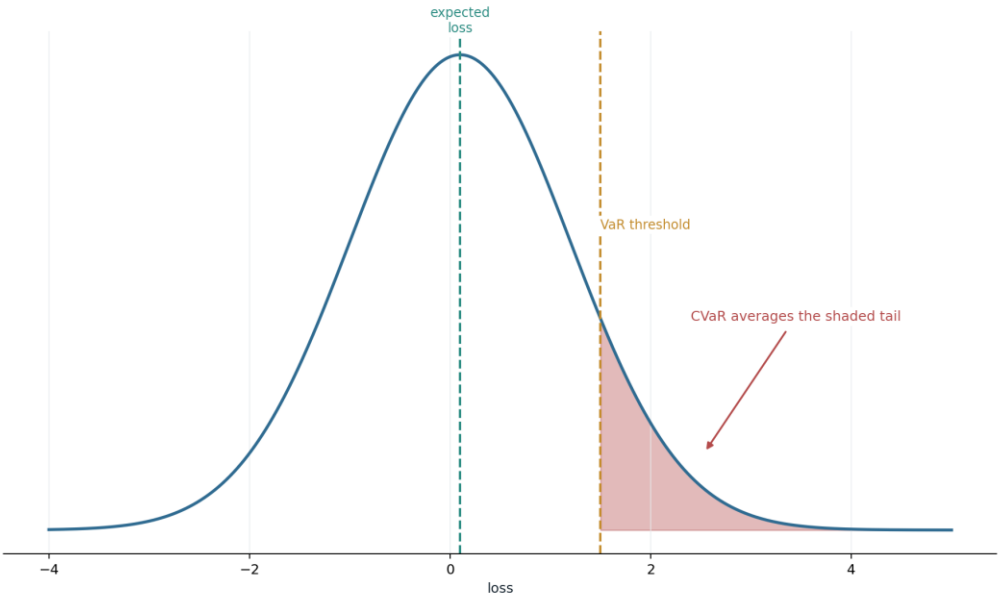


Figure F.10. Expected loss, a value-at-risk threshold, and the tail averaged by conditional value at risk.

Unit 33 · Dynamic programming, optimal control, MPC, and reinforcement learning

Prerequisites. Sequences, probability, differential equations, and optimization.

Why it matters in this volume. When actions change future states, present choices must include downstream consequences and information.

Dynamic programming uses the principle of optimality: after the first action, the remaining policy must be optimal for the resulting state under the same model. Bellman equations express value recursively. Optimal control applies this reasoning to dynamical systems. Model predictive control repeatedly solves a finite-horizon problem, applies the first action, observes, and replans.

Reinforcement learning estimates value functions, models, or policies from interaction. Exploration changes the data-generating process and may carry real risk. State choice, reward design, discounting, off-policy evaluation, and distribution shift determine what is learned.

$$V_t(s) = \min_a (c_t(s,a) + \mathbb{E}[V_{t+1}(S') \mid s,a]) \tag{F.70}$$

Bellman recursion combines immediate cost with optimal continuation value.

$$J(\pi) = \mathbb{E}_\pi[\textstyle\sum_{t=0}^{\infty} \gamma^t r_t], 0 \le \gamma < 1, |r_t| \le R < \infty. \tag{F.71}$$

A policy is evaluated by expected discounted return.

Worked reading. A controller with a perfect short-horizon model can still create long-horizon damage if the terminal cost and constraints omit slow state variables. Receding-horizon feedback does not repair a misspecified objective automatically.

Common traps.

- Treating reward as the same as purpose
- Assuming the Markov state is sufficient
- Deploying exploration without safety constraints and oversight

Where it reappears. Chapter 7, cloud scheduling, robotics, control, adaptive care, climate pathways, games, and sport training.

Sources. [F21; F22; F23; F24]

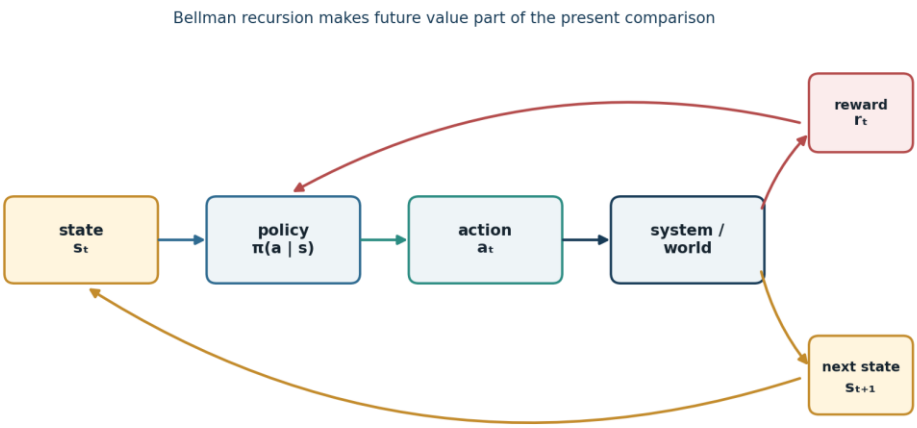


Figure F.11. State, policy, action, system, reward, and next-state feedback in Bellman recursion.

Unit 34 · Games, Nash equilibrium, mechanisms, and bilevel problems

Prerequisites. Probability, optimization, and logic.

Why it matters in this volume. Other agents adapt. Policies, markets, security controls, recommendations, and competitions can change the behavior that generates outcomes.

A game specifies players, strategies, information, and payoffs. A Nash equilibrium is a strategy profile where no player benefits from unilateral deviation. It predicts mutual best response under the model, not fairness, efficiency, uniqueness, or social desirability.

Mechanism design chooses rules so strategic responses produce desired properties. Bilevel optimization represents a leader choosing while anticipating a follower's optimization. The follower's problem may be set-valued, discontinuous, or misspecified.

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*) \quad \forall s_i \tag{F.72}$$

No player improves by deviating alone from a Nash equilibrium.

$$\min_x F\big(x, y^*(x)\big) \quad \text{where} \quad y^*(x) \in \arg\min_y f(x, y) \tag{F.73}$$

A leader anticipates a follower's optimized response.

Worked reading. A congestion-pricing policy changes route choices; evaluating it with fixed historical traffic ignores the response it is designed to induce. Equilibrium modeling helps, but behavioral and institutional evidence remains necessary.

Common traps.

- Equating equilibrium with optimum
- Assuming players know and optimize the modeled payoff
- Ignoring multiple follower optima in a bilevel model

Where it reappears. Chapter 6; security; markets and finance; policy and law; media; games and sport; Goodhart effects.

Sources. [F25]

Unit 35 · Queueing, reliability, and resource systems

Prerequisites. Probability, stochastic processes, and rates.

Why it matters in this volume. Latency, congestion, availability, maintenance, hospital flow, transport, and service capacity depend on arrival, service, and failure processes.

A queue couples random arrivals with finite service. Utilization near one can produce rapidly growing delay even when average capacity exceeds average demand. Little's law relates long-run average number in system, throughput, and average time under broad stability and conservation conditions.

We assess system reliability using component behavior and the way the components are connected. A series system fails when any required component fails. For redundancy, we also need dependencies and common causes. Availability includes repair and restoration as well as failure probability.

$$L = \lambda W \tag{F.74}$$

Average number equals throughput times average time under Little's-law conditions.

$$A = \frac{\mathrm{MTBF}}{\mathrm{MTBF+MTTR}} \tag{F.75}$$

A simple steady-state availability ratio combines mean up and repair times.

Worked reading. At 95% utilization, a small demand forecast error can create a large delay error. Capacity planning should compare distributions and tail service objectives, not only average work per average capacity.

Common traps.

- Using Little's law without a stable flow boundary
- Adding independent failure probabilities when failures share causes
- Equating utilization with user-visible service quality

Where it reappears. Chapters 8, 10–13, 18, and 19: cloud systems, hardware, manufacturing, transport, health systems, and public services.

Sources. [F26; F27]

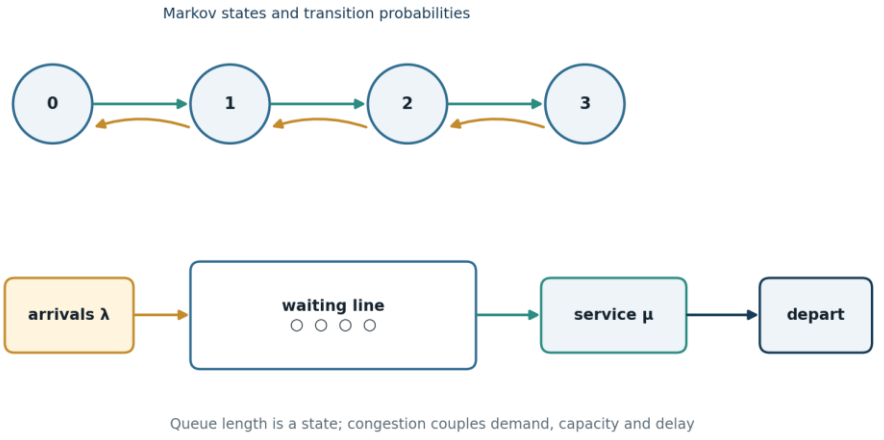


Figure F.12. Markov transitions and a queue with arrivals, waiting, service, and departure.

Part F.7 · Advanced bridges used in the volume

Unit 36 · Metric spaces and topology

Prerequisites. Sets, functions, and distance.

Why it matters in this volume. Optimization in curves, distributions, shapes, manifolds, and infinite-dimensional spaces needs language beyond ordinary coordinates.

A metric space supplies a distance and therefore notions of convergence, openness, and continuity. Topology retains neighborhood and continuity structure even when no particular distance is privileged. Compactness supports existence results by preventing minimizing sequences from escaping without a convergent subsequence.

Connectedness distinguishes one-piece feasible regions from separated components. Homeomorphism preserves topological structure, not lengths or angles. Topology optimization in engineering uses the word in a related but application-specific sense: material connectivity may change during design.

$$d(x,z) \le d(x,y) + d(y,z) \tag{F.76}$$

A metric obeys the triangle inequality.

$$x_n \to x \quad \Leftrightarrow \quad d(x_n, x) \to 0 \tag{F.77}$$

Metric convergence reduces to distance tending to zero.

Worked reading. A feasible set can be bounded in a coordinate plot yet fail to include its limiting boundary. Then an infimum may exist without an attainable minimum. Closedness and compactness clarify the distinction.

Common traps.

- Assuming every topology comes from the pictured Euclidean distance
- Equating boundedness with compactness in arbitrary spaces
- Using ‘topology’ without distinguishing mathematical from engineering usage

Where it reappears. Chapters 4, 7, 11, 15, and 20: manifolds, transport, topology optimization, creative spaces, and frontiers.

Sources. [F28]

Unit 37 · Function spaces, dual spaces, and infinite-dimensional optimization

Prerequisites. Vectors, norms, calculus, and topology.

Why it matters in this volume. Policies, signals, trajectories, fields, probability densities, and shapes are functions, so their optimization lives in a space of functions.

A normed function space treats functions as vectors. Completeness ensures Cauchy sequences converge within the space. A Hilbert space has an inner product and supports orthogonal projection. A dual space consists of continuous linear functionals; gradients are naturally dual objects before geometry identifies them with vectors.

Linear operators map functions to functions or numbers. Adjoints transfer a map across inner products and support efficient sensitivities. In a reflexive Banach space, every bounded sequence has a weakly convergent subsequence. Together with weak lower semicontinuity and a weakly closed feasible set, this can establish existence of a minimizer from a bounded minimizing sequence.

$$\langle f, g\rangle_{L^2} = \int f(t)g(t)\,dt \tag{F.78}$$

The L² inner product generalizes Euclidean dot products to functions.

$$\langle Ax, y\rangle = \langle x, A^*y\rangle \tag{F.79}$$

The adjoint transfers an operator across an inner product.

Worked reading. In PDE-constrained design, one adjoint solve can produce the gradient with respect to many parameters. The result remains a derivative of the discretized or continuous model chosen, so adjoint verification is still required.

Common traps.

- Treating all function spaces as interchangeable
- Calling a covector a vector without stating the pairing
- Ignoring boundary conditions and regularity

Where it reappears. Chapter 4, signals and control, inverse design, PDE models, optimal transport, and future differentiable systems.

Sources. [F14]

Unit 38 · Optimal transport and geometry of distributions

Prerequisites. Probability, optimization, metrics, and linear programming.

Why it matters in this volume. Optimal transport compares distributions by the cost of moving mass rather than comparing values point by point.

A coupling is a joint distribution with prescribed marginals. The transport problem chooses a coupling minimizing expected movement cost. Wasserstein distance arises from particular costs and metric assumptions. Entropic regularization makes computation smoother and often faster while spreading mass and changing the exact problem.

Optimal transport is used in domain adaptation, matching, imaging, generative models, allocation, and geometric comparison. We need to interpret the cost in each application. Moving probability mass over physical distance has a different meaning from matching representations using distance in an embedding.

$$\min_{\gamma\in\Pi(\mu,\nu)} \int c(x,y)\,d\gamma(x,y) \tag{F.80}$$

Choose a coupling with marginals μ and ν that minimizes movement cost.

$$W_p(\mu,\nu) = \left(\inf_{\gamma\in\Pi(\mu,\nu)} \int d(x,y)^p\,d\gamma\right)^{1/p} \tag{F.81}$$

Wasserstein distance is optimal p-power transport cost converted back to distance units.

Worked reading. Two histograms with nearby shifted peaks can be far under pointwise difference but close under transport. That usefulness depends on the ground distance representing meaningful movement.

Common traps.

- Using an arbitrary embedding distance as ground truth
- Forgetting marginal constraints
- Reporting an entropically regularized value as the exact unregularized optimum

Where it reappears. Chapter 7 and applications to data science, imaging, matching, distribution shift, and cross-domain translation.

Sources. [F29; F30]

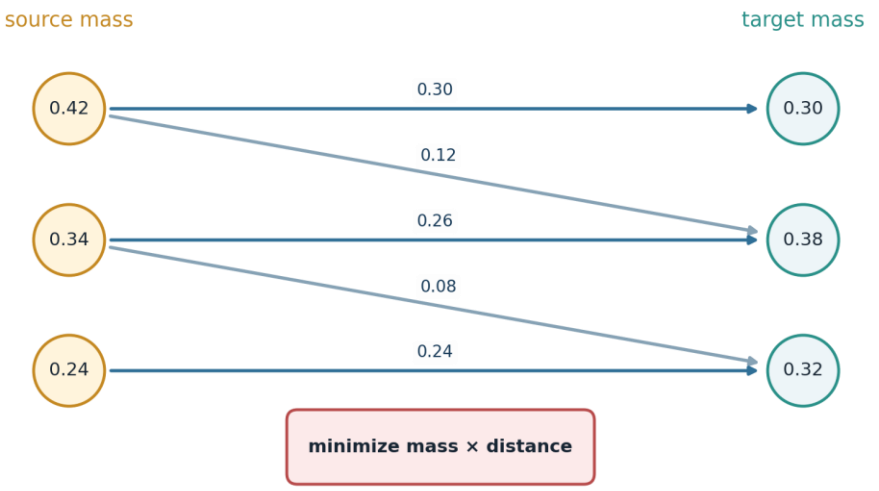


Figure F.13. A transport coupling moving source mass to target mass while minimizing mass-weighted distance.

Unit 39 · Information, entropy, and divergence

Prerequisites. Probability and logarithms.

Why it matters in this volume. Compression, uncertainty, learning, privacy, exploration, communication, and model comparison use information-theoretic quantities.

Shannon entropy is expected self-information. It is largest for a uniform distribution over a finite set and zero for a certain outcome. Cross-entropy scores predictions against observed distributions. Kullback–Leibler divergence measures directed information discrepancy and is not a metric: it is asymmetric and can be infinite.

Mutual information measures dependence by comparing a joint distribution with the product of marginals. Information measures depend on representation, discretization, and reference distribution; they do not automatically measure meaning or social value.

$$H(X) = -\sum_x p\,(x)\log p(x) \tag{F.82}$$

Entropy averages the information content of outcomes.

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_x P\,(x)\log \frac{P(x)}{Q(x)} \tag{F.83}$$

KL divergence penalizes coding P with a model Q.

Worked reading. A message stream can have high entropy because it contains random noise or because its signals vary substantially. Entropy measures unpredictability under the specified distribution. We need an additional model to interpret the messages' meaning or usefulness.

Common traps.

- Calling KL divergence a distance
- Comparing entropy values computed with different log bases without converting units
- Equating statistical information with meaning

Where it reappears. Chapters 7–9, signals and communication, privacy, active learning, Bayesian optimization, and creative/narrative domains.

Sources. [F31; F32]

Unit 40 · Category theory and compositional optimization

Prerequisites. Sets, functions, logic, and linear maps.

Why it matters in this volume. Category theory provides language for assembling optimization components while keeping interfaces and transformations explicit.

A category has objects, morphisms between objects, identity morphisms, and associative composition. Functions form a category of sets; linear maps form a category of vector spaces; processes can form categories when interfaces compose. A functor maps objects and morphisms while preserving identity and composition. A natural transformation compares functors coherently across every object.

A monoidal product models parallel combination; when the monoidal structure is cartesian, it is the categorical product. Composition models sequential connection. An optimization component can expose inputs, outputs, constraints, costs, states, and certificates. Composition is sound only when interfaces and semantics align; categorical notation does not make an invalid translation valid.

$$h \circ (g \circ f) = (h \circ g) \circ f \tag{F.84}$$

Associativity makes multi-stage composition unambiguous.

$$F(g \circ f) = F(g) \circ F(f), \qquad F(1_X) = 1_{F(X)} \tag{F.85}$$

A functor preserves composition and identities.

$$\eta_Y \circ F(f) = G(f) \circ \eta_X \tag{F.86}$$

Naturality states coherent comparison across a morphism f.

Worked reading. A model-to-solver pipeline is not automatically a functor: changing representation must preserve the relevant feasible decisions and objective meaning. If a compilation drops a constraint, composition has failed semantically even when types match.

Common traps.

- Treating analogy as isomorphism
- Assuming composability from matching data types alone
- Using categorical vocabulary without specifying objects and morphisms

Where it reappears. Chapter 7, compositional systems, software/hardware co-design, workflows, evidence chains, and future automated science.

Sources. [F33; F34; F35]

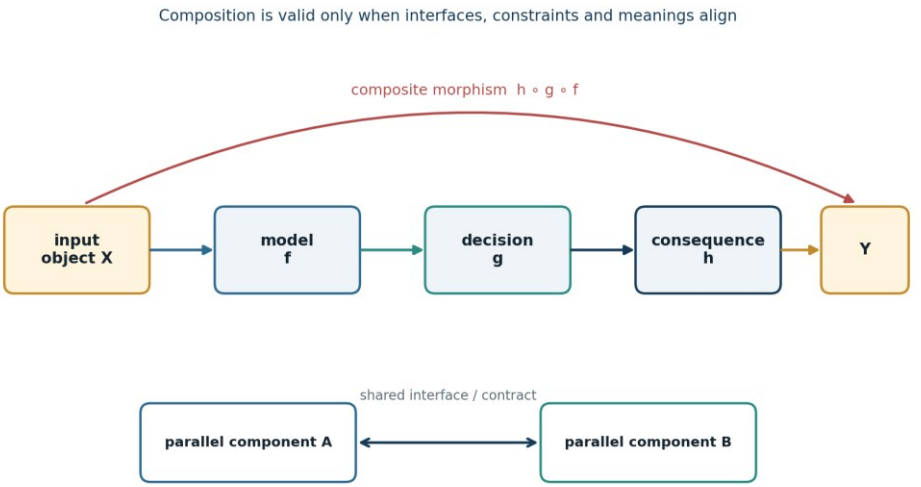


Figure F.14. Sequential and parallel composition of optimization components with explicit interfaces.

Unit 41 · Complexity, approximation, and certificates

Prerequisites. Logic, combinatorics, algorithms, and optimization.

Why it matters in this volume. A mathematical answer is operational only when it can be computed, checked, and delivered within resource and decision limits.

Time complexity counts operations as input size grows; space complexity counts memory. Big-O is an asymptotic upper bound, not a stopwatch prediction. Complexity classes describe

families of decision problems and reductions. NP-hardness concerns worst-case scalable exact solution, not impossibility of solving every useful instance.

An approximation algorithm supplies a quality ratio or additive error bound; a randomized algorithm may supply a probability statement. A feasible witness supports a feasibility claim. A certificate may supply a dual bound, residual bound, optimality gap, proof log, or verified enclosure. Checking such evidence is distinct from validating the model and data for their intended use.

$$T(n) \in O(n^2) \tag{F.87}$$

Runtime grows no faster than a constant multiple of n² beyond some threshold.

$$f(\hat{x}) \leq \alpha f(x^*), \qquad \alpha \geq 1 \tag{F.88}$$

For nonnegative minimization, an approximation ratio α≥1 bounds the achieved objective relative to the optimum.

$$\text{gap} = U - L \tag{F.89}$$

Upper and lower bounds quantify remaining objective uncertainty.

Worked reading. A solver can produce a tiny primal-dual gap for a model whose data are obsolete. Conversely, a valuable policy may have strong field evidence without a formal optimality certificate. Trustworthy practice records both layers.

Common traps.

- Reading asymptotic notation as an exact runtime
- Calling NP-hardness a reason not to model structure
- Confusing optimality evidence with real-world validity

Where it reappears. Chapters 2, 4, 5, 7–10, 17, and 20; approximation, verification, stopping rules, and computational frontiers.

Sources. [F6; F7]

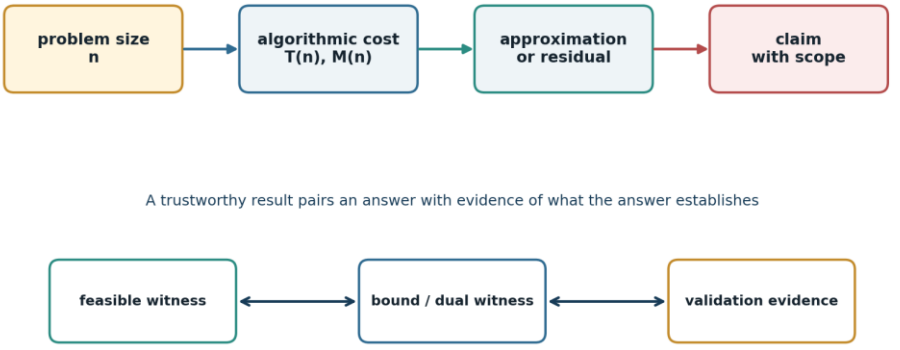


Figure F.15. Problem size, computational cost, approximation, scoped claims, and complementary certificate types.

Part F.8 · Cross-disciplinary mathematical bridges

Unit 42 · Reading mathematics across domains

Prerequisites. Units 1–41.

Why it matters in this volume. The same formal pattern may travel across disciplines, but its variables, evidence, units, and ethical meaning do not travel automatically.

In cloud capacity, Little's law connects throughput, time, and concurrent work, while a tail constraint protects service quality. In machine learning, a regularized loss balances fit with an inductive bias whose operational meaning must be defended. In control, state equations and Bellman recursion connect present action to future consequence. In medicine, Bayes' rule updates risk while causal assumptions determine treatment effects.

In finance, covariance and tail risk distinguish diversification from average return. In structural and semiconductor design, matrices, PDEs, constraints, and integer choices couple physics with manufacturability. In music, distances and constraints can describe voice leading without reducing artistic value to a single score. In literature, graphs and orderings can illuminate narrative alternatives without asserting that interpretation is a numeric optimum.

$$\text{formal similarity} \neq \text{semantic identity} \tag{F.90}$$

A shared equation is a translation aid, not proof that domains are the same.

Worked reading. For each translation, we identify the objects, morphisms, preserved and omitted structures, evidence, and decision authority. We can then distinguish a general analogy from a correspondence that satisfies specific preservation conditions.

Common traps.

- Copying a method without its assumptions
- Suppressing units and stakeholders during translation
- Confusing a useful model with a complete account of the domain

Where it reappears. All domain chapters and the seven worked translations in Chapters 18 and 19.

Sources. [F6; F7; F9; F10; F11; F12; F13; F14; F15; F16; F17; F18; F19; F20; F21; F22; F23; F24; F25; F26; F27; F28; F29; F30; F31; F32; F33; F34; F35]

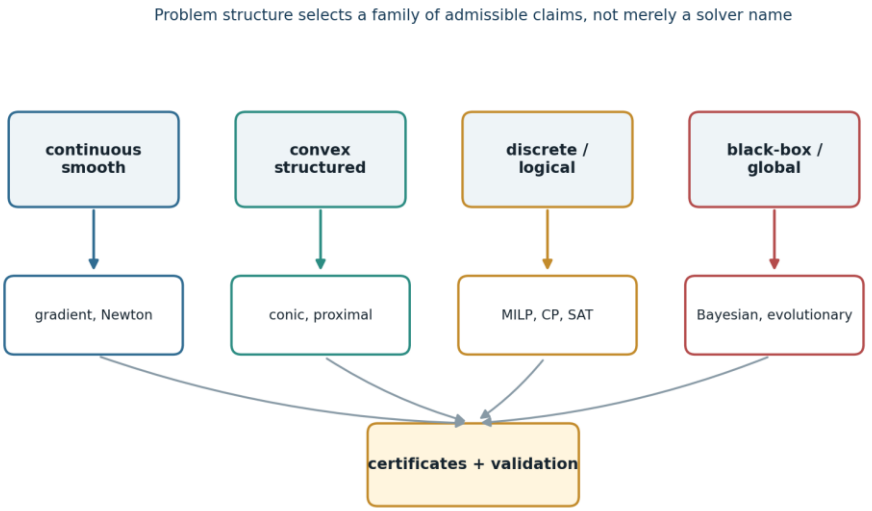


Figure F.16. Problem structures mapped to algorithm families, certificates, and validation.

Part F.9 · Equation atlas for the volume

The atlas covers the numbered equations in the front matter, chapters, and Appendix A. Labels V.1–V.124, V.10a, and V.59a distinguish repeated volume equations from the teaching equations F.1–F.90. Each reading explains the particular statement. Local symbols override general conventions in the notation concordance.

Volume equation 1 · Preface: Optimization as a shared language

Study. Units 5, 7, 23. **Structure.** optimization.

Reading. A selected feasible point belongs to the set of global minimizers; the expression does not claim uniqueness.

Local symbols. x*: selected decision; $\mathcal{X}$: feasible set; f: objective; argmin: set of minimizing arguments.

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x). \tag{V.1}$$

Volume equation 2 · Bird's-eye view

Study. Units 5, 7, 40, 42. **Structure.** composition; decision process.

Reading. Model the world, solve or deliberate, implement a choice, and evaluate the changed world. The arrows describe operations, not a numerical state recurrence.

Local symbols. W_t: world at stage t; P_t: formalized problem; S_t: selected solution; W_{t+1}: changed world; $\widehat{W}_{t+1}$: its observed or evaluated representation; M: modeling; A: algorithm or deliberation; I: implementation; E: evaluation, distinct from expectation.

$$W_t \xrightarrow{M} P_t \xrightarrow{A} S_t \xrightarrow{I} W_{t+1} \xrightarrow{E} \widehat{W}_{t+1}, \tag{V.2}$$

Volume equation 3 · The master form

Study. Units 23, 31, 32. **Structure.** formulation; uncertainty.

Reading. Choose a decision and policy to minimize a declared aggregation of uncertain consequences.

Local symbols. x: decision; π: policy; F: consequence vector; ξ: uncertainty; ρ: aggregation or risk functional; θ: preference or model parameter.

$$\min_{x,\pi} \rho(F(x, \pi, \xi);\ \theta) \tag{V.3}$$

Volume equation 4 · The master form

Study. Units 4, 7, 23, 32. **Structure.** feasibility; uncertainty.

Reading. Each of the m inequality requirements must hold under the stated treatment of uncertainty.

Local symbols. g_i: inequality function; x, π: decision and policy; ξ: uncertainty; i: constraint index.

$$g_i(x, \pi, \xi) \leq 0, \qquad i = 1, \ldots, m, \tag{V.4}$$

Volume equation 5 · The master form

Study. Units 4, 7, 23, 32. **Structure.** feasibility; equality.

Reading. Each of the p equality relations must hold under the stated treatment of uncertainty.

Local symbols. h_j: equality function; x, π: decision and policy; ξ: uncertainty; j: constraint index.

$$h_j(x, \pi, \xi) = 0, \qquad j = 1, \ldots, p, \tag{V.5}$$

Volume equation 6 · The master form

Study. Units 5, 7, 23. **Structure.** domains; requirement floors.

Reading. The decision and policy must belong to their admissible sets, and every requirement measure must meet its floor.

Local symbols. $\mathcal{X}$, Π: admissible sets; R_k: requirement measure; r_k: floor; k=1,…,q: requirement index.

$$x \in \mathcal{X}, \qquad \pi \in \Pi, \qquad R_k(x, \pi) \geq r_k. \tag{V.6}$$

Volume equation 7 · Purpose, alternatives, and decision systems

Study. Units 5, 7, 23. **Structure.** set-valued definition.

Reading. Argmin is defined as all feasible arguments whose objective is no greater than that of every feasible alternative.

Local symbols. x, y: feasible arguments; $\mathcal{X}$: feasible set; f: objective. This is a set definition, not a selected point or the minimum value f*.

$$\arg\min_{x \in \mathcal{X}} f(x) = \{x \in \mathcal{X} : f(x) \leq f(y) \text{ for all } y \in \mathcal{X}\}. \tag{V.7}$$

Volume equation 8 · Purpose, alternatives, and decision systems

Study. Units 19, 23, 32. **Structure.** bounded rationality; search cost.

Reading. Choose a method and effort level by comparing the expected loss of the method's output with the cost of obtaining it.

Local symbols. a: method; t: effort; $x_{(a,t)}$: resulting decision; L: loss; c: method and effort cost; $\mathbb{E}$: expectation over uncertainty in the outcome.

$$\min_{a,t} \mathbb{E}\big[L\big(x_{a,t}\big)\big] + c(a,t), \tag{V.8}$$

Volume equation 9 · Objectives, constraints, values, and authority

Study. Units 5, 8, 31. **Structure.** multiple objectives.

Reading. Collect k objective values in a vector without imposing a total ranking.

Local symbols. F: vector objective; $f_1,\ldots,f_k$: component criteria; x: decision.

$$F(x) = \big(f_1(x), f_2(x), \ldots, f_k(x)\big). \tag{V.9}$$

Volume equation 10 · Objectives, constraints, values, and authority

Study. Units 6, 8, 31. **Structure.** scalarization.

Reading. Form one objective by summing the component criteria with nonnegative weights.

Local symbols. J: scalarized objective; f_i: criterion; w_i: weight; k: number of criteria. Units and permitted substitutions require justification.

$$J(x) = \sum_{i=1}^{k} w_i\, f_i(x), \qquad w_i \geq 0, \tag{V.10}$$

Volume equation 10a · Chapter 1 · Why value types are not merely large weights

Study. Units 4, 7, 23, 25. **Structure.** penalties; rights.

Reading. Finite penalization need not have the same minimizers as the corresponding hard-constrained problem.

Local symbols. M: finite penalty coefficient; $v \geq 0$: violation measure; f: ordinary objective; $\mathcal{X}$: decision set. Equivalence needs a proved bound relative to violation or an applicable exact-penalty theorem.

$$\arg\min_{x\in\mathcal{X}} f(x) + Mv(x) \not\equiv \arg\min_{x\in\mathcal{X}, v(x)=0} f(x) \tag{V.10a}$$

Volume equation 11 · Objectives, constraints, values, and authority

Study. Units 23, 25. **Structure.** constrained optimization.

Reading. Minimize the objective subject to inequality and equality constraints.

Local symbols. f: objective; $g_i \leq 0$: inequality convention; $h_j = 0$: equality convention; x: decision.

$$\min_{x} f(x) \quad \text{subject to} \quad g_i(x) \leq 0,\ h_j(x) = 0, \tag{V.11}$$

Volume equation 12 · Objectives, constraints, values, and authority

Study. Units 6, 25, 26. **Structure.** Lagrangian definition.

Reading. Add multiplier-weighted constraint functions to the objective to define the Lagrangian.

Local symbols. λ_i: inequality multiplier; ν_j: equality multiplier; g_i, h_j: constraint functions; L: Lagrangian, not the original loss alone.

$$\mathcal{L}(x, \lambda, \nu) = f(x) + \sum_{i=1}^{m} \lambda_i \, g_i(x) + \sum_{j=1}^{p} \nu_j \, h_j(x). \tag{V.12}$$

Volume equation 13 · Search cost, baselines, and levels of intervention

Study. Units 19, 23, 32. **Structure.** search; delay; transition cost.

Reading. Choose method and effort by combining expected loss of their output with search, delay, and change costs.

Local symbols. $x_{(a,t)}$: method output; x_0: current practice; ξ: uncertain consequence; Csearch, Cdelay, Cchange: distinct costs. The decision is (a,t), not an independently chosen x. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{a,t} \; \mathbb{E}_{\xi}\big[L\big(x_{a,t}; \xi\big)\big] + C_{\text{search}}(a, t) + C_{\text{delay}}(t) + C_{\text{change}}\big(x_{a,t}, x_0\big) \tag{V.13}$$

Volume equation 14 · Representation, complexity, and equivalent decisions

Study. Units 6, 23, 28. **Structure.** facility location.

Reading. Choose binary assignments and site openings to minimize demand-weighted service cost plus opening cost.

Local symbols. x_{ij}: assignment; y_j: site opening; d_i: demand; c_{ij}: service cost; f_j: opening cost; n: demand points; m: sites; F_p: feasible pairs (x,y).

$$\min_{(x,y)\in\mathcal{F}_p} \big(\sum_{i=1}^{n} \sum_{j=1}^{m} d_i \, c_{ij} x_{ij} + \sum_{j=1}^{m} f_j \, y_j\big). \tag{V.14}$$

Volume equation 15 · Measurement and the evidence chain

Study. Units 19, 21. **Structure.** measurement.

Reading. An observed measure combines a context-dependent measurement map and error.

Local symbols. y: observation; z: intended construct; c: context; m: measurement map; ε: error. Making y a target can change this relationship.

$$y = m(z, c) + \varepsilon, \tag{V.15}$$

Volume equation 16 · Calculus and local numerical geometry

Study. Units 13, 14, 29. **Structure.** gradient descent.

Reading. Subtract a scaled gradient from the current iterate.

Local symbols. x_k: iterate; α_k: step size; ∇f: gradient. Descent and convergence require appropriate smoothness and step selection.

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \tag{V.16}$$

Volume equation 17 · Calculus and local numerical geometry

Study. Units 9, 14, 15, 29. **Structure.** Newton step.

Reading. Use the inverse Hessian to scale the gradient and compute a local Newton step.

Local symbols. H: Hessian; x_k: iterate; ∇f: gradient. The inverse needs nonsingularity; positive definiteness gives a descent direction for a nonzero gradient.

$$x_{k+1} = x_k - H(x_k)^{-1} \nabla f(x_k). \tag{V.17}$$

Volume equation 18 · Calculus and local numerical geometry

Study. Units 16, 17. **Structure.** functional.

Reading. Accumulate an integrand along a candidate curve to define its cost.

Local symbols. y(t): curve; y′: derivative; L: integrand; a,b: interval endpoints; J: functional.

$$J[y] = \int_a^b L\left(t, y(t), y'(t)\right) dt. \tag{V.18}$$

Volume equation 19 · Calculus and local numerical geometry

Study. Units 13, 17. **Structure.** Euler-Lagrange condition.

Reading. A stationary curve balances the integrand's dependence on position against the derivative of its dependence on slope.

Local symbols. L: integrand; y: curve; y′: slope; t: independent variable. The condition is necessary under the stated smoothness and boundary assumptions.

$$\frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial y'}\right) = 0. \tag{V.19}$$

Volume equation 20 · Calculus and local numerical geometry

Study. Units 23, 29. **Structure.** composite objective.

Reading. Minimize the sum of a smooth term and a structured nonsmooth term.

Local symbols. F: total objective; f: smooth term; g: nonsmooth term, distinct from the constraint functions elsewhere.

$$\min_x F(x) = f(x) + g(x), \tag{V.20}$$

Volume equation 21 · Calculus and local numerical geometry

Study. Units 10, 24, 29. **Structure.** proximal map.

Reading. Choose a point balancing the nonsmooth cost against squared distance from the supplied point.

Local symbols. g: proper convex term under the proximal assumptions; α>0: scale; v: supplied point. The squared-distance term makes the minimizer unique under the usual closed convex assumptions.

$$\mathrm{prox}_{\alpha g}(v) = \arg\min_x \left(g(x) + \frac{1}{2\alpha} \| x - v \|_2^2 \right). \tag{V.21}$$

Volume equation 22 · Convexity, duality, and certificates

Study. Units 5, 24. **Structure.** convexity.

Reading. The function value on a line segment is no greater than the corresponding mixture of endpoint values.

Local symbols. θ∈[0,1]: mixture weight; x,y: domain points; f: convex function.

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta) f(y), \qquad 0 \le \theta \le 1. \tag{V.22}$$

Volume equation 23 · Convexity, duality, and certificates

Study. Units 25, 26. **Structure.** dual function.

Reading. Define the dual function as the infimum of the Lagrangian over primal variables.

Local symbols. q: dual function; L: Lagrangian; λ≥0: inequality multipliers for g≤0; v: unrestricted equality multipliers.

$$q(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu), \qquad \lambda \ge 0. \tag{V.23}$$

Volume equation 24 · Convexity, duality, and certificates

Study. Units 25, 26. **Structure.** dual optimization.

Reading. Choose admissible multipliers to maximize the lower bound supplied by the dual function.

Local symbols. q: dual function; λ≥0: inequality multipliers; v: equality multipliers. The best dual value need not equal the primal optimum.

$$\max_{\lambda \geq 0, \nu} q(\lambda, \nu). \tag{V.24}$$

Volume equation 25 · Convexity, duality, and certificates

Study. Units 14, 25. **Structure.** KKT stationarity.

Reading. At a KKT point, the objective gradient and weighted active-constraint gradients balance.

Local symbols. x*: candidate; λ*,ν*: multipliers; m,p: inequality and equality counts; f,g_i,h_j: objective and constraints.

$$\nabla f(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(x^*) + \sum_{j=1}^{p} \nu_j^* \nabla h_j(x^*) = 0, \tag{V.25}$$

Volume equation 26 · Convexity, duality, and certificates

Study. Units 23, 25. **Structure.** primal feasibility.

Reading. The candidate satisfies all original inequality and equality constraints.

Local symbols. x*: candidate; $g_i \leq 0$ and $h_j = 0$: primal requirements.

$$g_i(x^*) \leq 0, \qquad h_j(x^*) = 0, \tag{V.26}$$

Volume equation 27 · Convexity, duality, and certificates

Study. Unit 25. **Structure.** dual feasibility; complementarity.

Reading. Inequality multipliers are nonnegative, and each multiplier times its constraint value is zero.

Local symbols. λ_i*: multiplier; $g_i(x^*)$: constraint value. An inactive inequality has zero multiplier.

$$\lambda_i^* \geq 0, \qquad \lambda_i^* g_i(x^*) = 0. \tag{V.27}$$

Volume equation 28 · Convexity, duality, and certificates

Study. Units 10, 24, 26, 37. **Structure.** convex conjugacy.

Reading. Define the convex conjugate by maximizing a linear pairing minus the original function.

Local symbols. f*(y): conjugate function, not an optimal scalar value; x: primal argument; y: dual argument; ⟨y,x⟩: pairing.

$$f^*(y) = \sup_x \big(\langle y, x \rangle - f(x) \big) \tag{V.28}$$

Volume equation 29 · Convexity, duality, and certificates

Study. Units 10, 24, 29. **Structure.** proximal map.

Reading. Define the proximal point through function value plus a quadratic distance penalty.

Local symbols. f: nonsmooth function in this local notation; γ>0: scale; v: input. This is the same construction as Equation (21) with renamed quantities.

$$\operatorname{prox}_{\gamma f}(v) = \arg\min_{x} \left(f(x) + \frac{1}{2\gamma} \| x - v \|^2 \right). \tag{V.29}$$

Volume equation 30 · Convexity, duality, and certificates

Study. Units 23, 27. **Structure.** linear program.

Reading. Maximize production benefit subject to two capacity constraints and nonnegative decisions.

Local symbols. x_1,x_2: production quantities; 3,5: unit benefits; 8,12: capacities. Coefficients specify the resource use in the example.

$$\max_{x_1,x_2 \geq 0} 3x_1 + 5x_2 \quad \text{subject to} \quad x_1 + 2x_2 \leq 8,\ 3x_1 + 2x_2 \leq 12. \tag{V.30}$$

Volume equation 31 · Convexity, duality, and certificates

Study. Units 26, 27. **Structure.** linear dual.

Reading. Minimize capacity value over nonnegative prices that cover each activity's benefit.

Local symbols. y_1,y_2: capacity multipliers. Feasible dual prices bound the primal benefit in Equation (30).

$$\min_{y_1,y_2 \geq 0} 8y_1 + 12y_2 \quad \text{subject to} \quad y_1 + 3y_2 \geq 3,\ 2y_1 + 2y_2 \geq 5. \tag{V.31}$$

Volume equation 32 · Linear and conic forms as compilation targets

Study. Units 9, 23, 27. **Structure.** linear programming.

Reading. Minimize a linear cost subject to linear balances and nonnegativity.

Local symbols. c: cost vector; A: constraint matrix; b: balance vector; x: decision.

$$\min_{x} c^T x \quad \text{subject to} \quad Ax = b,\ x \geq 0. \tag{V.32}$$

Volume equation 33 · Linear and conic forms as compilation targets

Study. Units 9, 26, 27. **Structure.** linear duality.

Reading. Maximize the balance-weighted dual value subject to reduced-cost inequalities.

Local symbols. y: unrestricted multipliers for Ax=b; A^T: transpose; b,c: primal data. This is the dual of Equation (32).

$$\max_y b^T y \quad \text{subject to} \quad A^T y \le c. \tag{V.33}$$

Volume equation 34 · Linear and conic forms as compilation targets

Study. Units 9, 15, 24, 27. **Structure.** quadratic objective.

Reading. Add a quadratic form and a linear term.

Local symbols. Q: quadratic matrix; c: linear coefficients; x: vector. Positive semidefiniteness of the symmetric Q gives convexity.

$$\frac{1}{2} x^T Q x + c^T x \tag{V.34}$$

Volume equation 35 · Linear and conic forms as compilation targets

Study. Units 9, 23, 27. **Structure.** conic optimization.

Reading. Minimize a linear objective over an affine slice of a cone.

Local symbols. K: admissible cone; A,b: linear constraints; c: cost. The cone determines the standard problem family.

$$\min_x c^T x \quad \text{subject to} \quad Ax = b,\ x \in K, \tag{V.35}$$

Volume equation 36 · Discrete search as proof construction

Study. Units 23, 28, 41. **Structure.** mixed-integer optimization.

Reading. Minimize linear cost with linear inequalities and integrality on the indexed variables.

Local symbols. I: integer-variable index set; A,b,c: model data; x: decision vector. Bounds and proof status qualify a solver result.

$$\min_x c^T x \quad \text{subject to} \quad Ax \le b, \quad x_i \in \mathbb{Z} \text{ for } i \in I. \tag{V.36}$$

Volume equation 37 · Discrete search as proof construction

Study. Units 7, 28. **Structure.** submodularity.

Reading. Adding an element has at least as much marginal benefit for the smaller set as for the larger set.

Local symbols. $A \subseteq B$ and $e \notin B$; f: set function. The classical cardinality-constrained greedy guarantee additionally requires normalization, nonnegativity, and monotonicity.

$$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B) \tag{V.37}$$

Volume equation 38 · Global, black-box, and surrogate search

Study. Units 10, 24, 29. **Structure.** subgradient.

Reading. A subgradient defines an affine lower support to a convex function at x.

Local symbols. g: subgradient vector, not a constraint function; x: support point; y: arbitrary domain point.

$$f(y) \geq f(x) + g^T(y - x) \quad \text{for all } y. \tag{V.38}$$

Volume equation 39 · Global, black-box, and surrogate search

Study. Units 14, 24, 29. **Structure.** proximal gradient.

Reading. Take a gradient step on the smooth term, then apply the proximal map of the nonsmooth term.

Local symbols. f: smooth term; r: nonsmooth term; α: step size; x_k: iterate.

$$x_{k+1} = \operatorname{prox}_{\alpha r}\big(x_k - \alpha \nabla f(x_k)\big). \tag{V.39}$$

Volume equation 40 · Uncertainty, risk, and four attitudes to action

Study. Units 19, 23, 32. **Structure.** stochastic objective.

Reading. Choose a decision to minimize expected loss under a stated distribution.

Local symbols. f(x,ξ): scenario loss; ξ: random input; $\mathbb{E}$: expectation. The distribution is part of the model.

$$\min_x \ \mathbb{E}_\xi[f(x, \xi)] \tag{V.40}$$

Volume equation 41 · Uncertainty, risk, and four attitudes to action

Study. Units 23, 32. **Structure.** two-stage recourse.

Reading. Minimize the first-stage cost plus expected optimal recourse cost.

Local symbols. x: initial decision; c: initial cost; Q(x,ξ): value of the second-stage problem after ξ is observed. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_x \ c^T x + \mathbb{E}_\xi[Q(x, \xi)], \tag{V.41}$$

Volume equation 42 · Uncertainty, risk, and four attitudes to action

Study. Units 9, 23, 32. **Structure.** recourse value.

Reading. Define second-stage value by minimizing adjustment cost subject to the residual scenario requirements.

Local symbols. y: recourse decision; q,W,h,T: scenario-dependent data; x: fixed first-stage choice. T is a technology matrix here.

$$Q(x,\xi) = \min_{y}\{q(\xi)^T y : W(\xi)y = h(\xi) - T(\xi)x,\ y \geq 0\}. \tag{V.42}$$

Volume equation 43 · Uncertainty, risk, and four attitudes to action

Study. Units 18, 23, 32. **Structure.** chance constraint.

Reading. Require the constraint to hold with probability at least one minus the allowed violation probability.

Local symbols. g: constraint function; ξ: random input; ε: violation allowance; x: decision.

$$\Pr(g(x,\xi) \leq 0) \geq 1 - \varepsilon \tag{V.43}$$

Volume equation 44 · Uncertainty, risk, and four attitudes to action

Study. Units 19, 24, 32. **Structure.** CVaR definition.

Reading. Minimize over an auxiliary threshold to obtain upper-tail conditional value at risk.

Local symbols. L: loss; η: threshold in loss units; α∈[0,1): tail level; positive part: max(L−η,0). This definition handles atoms through the threshold formulation. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\mathrm{CVaR}_{\alpha}(L) = \min_{\eta}\left[\eta + \frac{1}{1-\alpha}\mathbb{E}[(L-\eta)_+]\right]. \tag{V.44}$$

Volume equation 45 · Uncertainty, risk, and four attitudes to action

Study. Units 23, 32. **Structure.** robust optimization.

Reading. Choose a decision with the smallest worst loss over the admitted uncertainty set.

Local symbols. U: uncertainty set; ξ: uncertain realization; f: loss. Protection extends only to this set.

$$\min_{x} \max_{\xi \in \mathcal{U}} f(x,\xi). \tag{V.45}$$

Volume equation 46 · Uncertainty, risk, and four attitudes to action

Study. Units 19, 23, 32. **Structure.** distributional robustness.

Reading. Choose a decision against the largest expected loss over plausible probability laws.

Local symbols. P: one probability law; script P: ambiguity set of laws; $\mathbb{E}_P$: expectation under P; ξ: uncertain input.

$$\min_{x} \sup_{P \in \mathcal{P}} \mathbb{E}_P[f(x, \xi)]. \tag{V.46}$$

Volume equation 47 · Multiple objectives and uncertain preferences

Study. Units 5, 34. **Structure.** Nash equilibrium.

Reading. At the equilibrium profile, no player improves its payoff by changing only its own action.

Local symbols. u_i: player i's payoff; x_i^*: equilibrium action; x_{-i}^*: other players' fixed actions. This is not a welfare-optimality claim.

$$u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*) \quad \text{for every player } i \text{ and action } x_i. \tag{V.47}$$

Volume equation 48 · Multiple objectives and uncertain preferences

Study. Units 23, 34. **Structure.** optimistic bilevel optimization.

Reading. Choose an upper-level decision and a lower-level minimizing response that is favorable to the upper objective among ties.

Local symbols. F: upper objective; f,g: lower objective and constraints; x: leader choice; y: selected follower solution; z: dummy follower variable.

$$\min_{x,y} F(x, y) \quad \text{subject to} \quad y \in \arg\min_{z} \{f(x, z) : g(x, z) \leq 0\}. \tag{V.48}$$

Volume equation 49 · Decisions through time

Study. Units 6, 22, 33. **Structure.** state dynamics.

Reading. Update the state from current state, action, and disturbance.

Local symbols. s_t: state; a_t: action; w_t: disturbance; T: transition map. State sufficiency requires a separate modeling argument.

$$s_{t+1} = T(s_t, a_t, w_t) \tag{V.49}$$

Volume equation 50 · Decisions through time

Study. Units 3, 6, 19, 33. **Structure.** discounted policy value.

Reading. Sum the policy's future costs with geometric discounting and take expectation.

Local symbols. J^π: discounted policy cost; π: policy; s_0: initial state; s_t, a_t: state and action at time t; c: stage cost; $0 \leq \gamma < 1$: discount; $C < \infty$: uniform bound on $|c(s, a)|$. The bound and discount make the sum finite. $\mathbb{E}$: expectation under the stated law and conditioning.

$$J^{\pi}(s_0) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \, c(s_t, a_t)], 0 \leq \gamma < 1, |c(s,a)| \leq C < \infty. \tag{V.50}$$

Volume equation 51 · Decisions through time

Study. Units 19, 20, 23, 33. **Structure.** Bellman recursion.

Reading. Optimal value equals the least immediate cost plus discounted expected optimal continuation value.

Local symbols. V*: optimal value function; s: state; a: action; γ: discount; s′: next state. The recurrence assumes a sufficient Markov state and applicable attainment conditions. $\mathbb{E}$: expectation under the stated law and conditioning.

$$V^*(s) = \min_a [c(s,a) + \gamma \, \mathbb{E}[V^*(s') \mid s, a]]. \tag{V.51}$$

Volume equation 52 · Decisions through time

Study. Units 16, 17, 33. **Structure.** optimal-control objective.

Reading. Add terminal cost to accumulated running cost along the controlled trajectory.

Local symbols. u: control; x(t): state; T: terminal time; Φ: terminal cost; L: running cost.

$$J[u] = \Phi\big(x(T)\big) + \int_0^T L\,(x(t), u(t), t)\, dt. \tag{V.52}$$

Volume equation 53 · Decisions through time

Study. Units 10, 17, 33. **Structure.** Hamiltonian.

Reading. Define the Hamiltonian as running cost plus the costate paired with dynamics.

Local symbols. H: Hamiltonian; L: running cost; λ: costate; f: dynamics. With this minimization convention, minimize H pointwise, or maximize −H.

$$H(x, u, \lambda, t) = L(x, u, t) + \lambda^T f(x, u, t). \tag{V.53}$$

Volume equation 54 · Decisions through time

Study. Units 6, 19, 20, 33. **Structure.** finite-horizon Bellman recursion.

Reading. Current optimal value is the least expected stage cost plus next-period optimal value.

Local symbols. V_t,V_{t+1}: value functions; A(s): feasible actions; T: transition; ξ_t: disturbance, denoted w_t in Equation (49). $\mathbb{E}$: expectation under the stated law and conditioning.

$$V_t(s) = \min_{a \in \mathcal{A}(s)} \mathbb{E}\big[c_t(s, a, \xi_t) + V_{t+1}\big(T(s, a, \xi_t)\big)\big]. \tag{V.54}$$

Volume equation 55 · Transport and geometry as semantics

Study. Units 18, 36, 38. **Structure.** optimal transport.

Reading. Select a coupling with the required marginals to minimize total ground cost.

Local symbols. μ,ν: measures; γ: coupling; Π(μ,ν): coupling set; c: ground cost. A p-Wasserstein distance needs p≥1, finite pth moments, and the pth root of the distance-power cost optimum.

$$\min_{\gamma \in \Pi(\mu,\nu)} \int c(x,y)\, d\gamma(x,y). \tag{V.55}$$

Volume equation 56 · Transport and geometry as semantics

Study. Units 18, 38, 39. **Structure.** entropic transport.

Reading. Add a scaled relative-entropy penalty against the product measure to the transport objective.

Local symbols. γ: coupling; μ⊗ν: independent product measure; KL: relative entropy; ε: regularization scale. This changes both computation and the chosen coupling.

$$\varepsilon\, \mathrm{KL}(\gamma \parallel \mu \otimes \nu) \tag{V.56}$$

Volume equation 57 · Category theory and composition

Study. Units 5, 40. **Structure.** open component.

Reading. Represent a component as a morphism from its typed input interface to its typed output interface.

Local symbols. P: open optimization component; X,Y: boundary types. Internal decisions, feasibility, and costs are additional component data.

$$P : X \to Y. \tag{V.57}$$

Volume equation 58 · Category theory and composition

Study. Units 5, 40. **Structure.** parallel composition.

Reading. Place two components side by side, combining their input and output interfaces through the monoidal product.

Local symbols. P_1,P_2: components; X_1,X_2,Y_1,Y_2: interfaces; ⊗: chosen parallel product.

$$(P_1 \otimes P_2) : (X_1 \otimes X_2) \to (Y_1 \otimes Y_2). \tag{V.58}$$

Volume equation 59 · Category theory and composition

Study. Units 7, 23, 40. **Structure.** set inclusion.

Reading. The minimizers of a problem form a subset of its decision space.

Local symbols. Argmin(P): set of minimizing decisions; X_P: decision space. Retaining this set alone does not preserve locally nonoptimal alternatives needed downstream.

$$\operatorname{Argmin}(P) \subseteq \mathcal{X}_P, \tag{V.59}$$

Volume equation 59a · Chapter 7 · An example of predictor, scheduler, and controller composition

Study. Units 23, 26, 40. **Structure.** min-plus composition.

Reading. Compose boundary value functions by taking the infimum of their sum over the shared interface.

Local symbols. V_P: boundary cost, $+\infty$ if infeasible; y: eliminated interface; x,z: retained boundaries. Recover the composite argmin after composition, not from independent local argmins.

$$V_{P_2 \circ P_1}(x, z) = \inf_y \left[V_{P_1}(x, y) + V_{P_2}(y, z) \right] \tag{V.59a}$$

Volume equation 60 · Algorithms, representations, and resource models

Study. Units 1, 8, 23, 41. **Structure.** algorithm resources.

Reading. Form a declared weighted cost of running an algorithm at a given input size.

Local symbols. A: algorithm; n: input size; T,M,I,E: time, memory, input/output, energy; w: weights. This A is not a matrix.

$$C(A, n) = w_t T(A, n) + w_m M(A, n) + w_i I(A, n) + w_e E(A, n), \tag{V.60}$$

Volume equation 61 · Algorithms, representations, and resource models

Study. Units 4, 23, 41. **Structure.** approximation guarantee.

Reading. The selected point's nonnegative objective is at most ρ times the optimal value.

Local symbols. ρ≥1: approximation ratio; x-hat: returned point; x*: optimum; f≥0: objective. The ratio is a guarantee, not a risk functional.

$$f(\hat{x}) \leq \rho f(x^*) \tag{V.61}$$

Volume equation 62 · Algorithms, representations, and resource models

Study. Units 8, 31. **Structure.** resource vector.

Reading. Keep several service outcomes as separate vector components.

Local symbols. T_{50},T_{99}: latency percentiles; M: memory; B: bandwidth; E: energy; C: monetary cost; A: availability. Directions of preference differ.

$$R(x) = \big(T_{50}(x), T_{99}(x), M(x), B(x), E(x), C(x), A(x)\big), \tag{V.62}$$

Volume equation 63 · Systems, queues, and cloud capacity

Study. Units 1, 2, 35. **Structure.** Little's law.

Reading. Long-run average population equals throughput times average time in a stable system.

Local symbols. L: average number in the system; λ: long-run arrival rate, equal to throughput under the stated stability and flow-conservation conditions; W: average time in the system. The identity is neither a scheduling policy nor a utilization-delay formula.

$$L = \lambda W. \tag{V.63}$$

Volume equation 64 · Systems, queues, and cloud capacity

Study. Units 2, 4, 41. **Structure.** parallel speedup.

Reading. A serial fraction limits ideal speedup from increasing the processor count.

Local symbols. s: serial fraction; p: number of processors; S(p): ideal speedup. Communication and imbalance are outside this idealization.

$$S(p) = \frac{1}{s + (1-s)/p}. \tag{V.64}$$

Volume equation 65 · Systems, queues, and cloud capacity

Study. Units 23, 31. **Structure.** cloud scalarization.

Reading. Choose a deployment by a weighted sum of money, latency, and carbon costs.

Local symbols. x: placement or schedule; λ,μ: objective weights, not queueing rates. Capacity and service obligations remain constraints.

$$\min_{x} \; C_{\text{money}}(x) + \lambda C_{\text{latency}}(x) + \mu C_{\text{carbon}}(x) \tag{V.65}$$

Volume equation 66 · Systems, queues, and cloud capacity

Study. Units 23, 28, 32, 35. **Structure.** capacity; risk; survivability.

Reading. Choose feasible replica counts to minimize allocation cost plus energy cost.

Local symbols. r_{sz}: integer replicas; c_{sz}: unit cost; R: allocations satisfying capacity, latency-risk, and failure-domain conditions; λE: energy weight.

$$\min_{r \in \mathcal{R}} \Big(\textstyle\sum_{s,z} c_{sz}\, r_{sz} + \lambda_E E(r) \Big). \tag{V.66}$$

Volume equation 67 · Learning as nested, decision-focused optimization

Study. Units 6, 14, 21, 23, 29. **Structure.** empirical risk.

Reading. Fit parameters by minimizing average training loss plus regularization.

Local symbols. θ-hat: selected parameters; fθ: predictor; ℓ: loss; R: regularizer; λ: strength; n: sample count. Membership permits tied minimizing parameters. Validation must respect deployment dependence.

$$\hat{\theta} \in \arg\min_{\theta} \frac{1}{n}\sum_{i=1}^{n} \ell\left(f_\theta(x_i), y_i\right) + \lambda R(\theta). \tag{V.67}$$

Volume equation 68 · Learning as nested, decision-focused optimization

Study. Units 19, 20, 23. **Structure.** Bayes decision.

Reading. Select an action that minimizes posterior expected loss for the observed features.

Local symbols. a*: selected action; L(a,Y): action loss; x: features; Y: uncertain outcome. Membership allows tied minimizing actions. $\mathbb{E}$: expectation under the stated law and conditioning.

$$a^*(x) \in \arg\min_{a} \mathbb{E}[L(a, Y) \mid X = x]. \tag{V.68}$$

Volume equation 69 · Learning as nested, decision-focused optimization

Study. Units 3, 7, 18. **Structure.** differential privacy.

Reading. For every adjacent dataset pair and measurable output event, bound how much the mechanism's output probabilities can differ.

Local symbols. M: randomized mechanism; D,D′: adjacent under the declared relation; S: measurable event; ε,δ: privacy parameters. The claim is not for arbitrary dataset pairs.

$$\Pr[M(D) \in S] \leq e^{\varepsilon}\Pr[M(D') \in S] + \delta \tag{V.69}$$

Volume equation 70 · Learning as nested, decision-focused optimization

Study. Units 19, 20, 23. **Structure.** expected utility.

Reading. Choose an admissible action maximizing utility averaged under estimated conditional probabilities.

Local symbols. Pθ: estimated law; U: utility; A: action set; x: features; y: outcome. Abstention, referral, and information gathering may be included.

$$a^*(x) \in \arg\max_{a \in \mathcal{A}} \sum_{y \in \mathcal{Y}} P_\theta\left(y \mid x\right) U(a, y; x). \tag{V.70}$$

Volume equation 71 · Security, privacy, and adversarial optimization

Study. Units 19, 23, 34. **Structure.** adaptive adversary.

Reading. Choose defensive investment while accounting for its effect on both harm and the attack distribution.

Local symbols. x: investment; $\mathcal{X}$: admissible investment set; a: attack; P(a|x): response distribution; H: harm; C: cost. Historical attack frequencies are endogenous to defense. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{x\in\mathcal{X}} \mathbb{E}_{a\sim P(a|x)}[H(x,a)] + C(x), \tag{V.71}$$

Volume equation 72 · Security, privacy, and adversarial optimization

Study. Units 19, 23, 32. **Structure.** mean and tail loss.

Reading. Minimize expected loss plus a weighted tail-risk measure.

Local symbols. L(x,ξ): loss; λ: risk weight; α: CVaR level. Safety and recovery requirements may remain separate constraints. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{x}\ \mathbb{E}[L(x,\xi)] + \lambda \mathrm{CVaR}_{\alpha}\big(L(x,\xi)\big). \tag{V.72}$$

Volume equation 73 · Security, privacy, and adversarial optimization

Study. Units 19, 23, 32, 34. **Structure.** adversarial training.

Reading. Fit parameters against the worst admitted perturbation for each sampled input.

Local symbols. θ: model parameters; fθ: model; ℓ: loss; δ∈Δ(x): perturbation. The specified threat set limits the robustness claim. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{\theta}\ \mathbb{E}_{(x,y)}\left[\max_{\delta\in\Delta(x)} \ell(f_\theta(x+\delta), y)\right]. \tag{V.73}$$

Volume equation 74 · Systems engineering, co-design, and exchanged envelopes

Study. Units 9, 23. **Structure.** multidisciplinary objective.

Reading. Optimize a shared design jointly with discipline-specific states.

Local symbols. x: shared design; $y_1,\dots,y_m$: discipline states; m: discipline count; f: system objective.

$$\min_{x,y_1,\dots,y_m} f(x,y_1,\dots,y_m) \tag{V.74}$$

Volume equation 75 · Systems engineering, co-design, and exchanged envelopes

Study. Units 9, 23. **Structure.** discipline consistency.

Reading. Require each coupled discipline residual to be zero.

Local symbols. r_i: discipline equation residual; x: shared variables; y: coupled states; m: number of disciplines.

$$r_i(x, y_1, \dots, y_m) = 0, \qquad i = 1, \dots, m, \tag{V.75}$$

Volume equation 76 · Structures and physical feasibility

Study. Units 9, 23. **Structure.** structural sizing.

Reading. Minimize total mass for the fixed element mass coefficients and sizing variables.

Local symbols. m: mass coefficient vector, not the discipline count; x: sizing vector. Equilibrium, stress, displacement, and manufacturing conditions supply feasibility.

$$\min_x m^T x \tag{V.76}$$

Volume equation 77 · Structures and physical feasibility

Study. Units 9, 23, 27. **Structure.** topology; compliance.

Reading. Minimize load-displacement compliance subject to equilibrium and a material-volume limit.

Local symbols. ρ_e: element density; v_e: element volume; K: stiffness; f: applied load; u: displacement; V: volume allowance. A positive density floor or ersatz stiffness avoids singularity.

$$\min_{\rho} f^T u(\rho) \quad \text{subject to} \quad K(\rho)u = f, \quad \sum_e v_e \rho_e \le V. \tag{V.77}$$

Volume equation 78 · Signals, estimation, sensing, and control

Study. Units 9, 10, 27. **Structure.** least squares.

Reading. Measure fit by the squared Euclidean residual between predicted and observed data.

Local symbols. A: measurement operator; x: unknown vector; y: observation. Noise assumptions determine whether this is the appropriate loss.

$$\| Ax - y \|_2^2. \tag{V.78}$$

Volume equation 79 · Signals, estimation, sensing, and control

Study. Units 3, 18, 39. **Structure.** entropy.

Reading. Compute average self-information of a discrete source in bits.

Local symbols. p(x): source probability; H(X): entropy; $\log_2$: base-two logarithm; zero-probability terms use the continuous-limit convention.

$$H(X) = -\sum_x p\,(x)\log_2 p(x). \tag{V.79}$$

Volume equation 80 · Signals, estimation, sensing, and control

Study. Units 18, 23, 39. **Structure.** channel capacity.

Reading. Maximize input-output mutual information over admissible input distributions for the specified channel.

Local symbols. C: capacity; p(x): input law; I(X;Y): mutual information. Channel and resource assumptions define the admissible laws.

$$C = \max_{p(x)} I(X;Y), \tag{V.80}$$

Volume equation 81 · Signals, estimation, sensing, and control

Study. Units 6, 9, 33. **Structure.** linear state space.

Reading. Update the state linearly and map it to an observed output.

Local symbols. A: state matrix; B: input matrix; C: output matrix, not channel capacity; x_t: state; u_t: control; y_t: output.

$$x_{t+1} = Ax_t + Bu_t, \qquad y_t = Cx_t, \tag{V.81}$$

Volume equation 82 · Signals, estimation, sensing, and control

Study. Units 6, 9, 33. **Structure.** linear-quadratic control.

Reading. Accumulate quadratic penalties on state and control over an infinite horizon.

Local symbols. Q,R: weighting matrices; x_t,u_t: state and control. Stability, detectability, and weight assumptions qualify the control result.

$$J = \sum_{t=0}^{\infty} (x_t^T Q x_t + u_t^T R u_t). \tag{V.82}$$

Volume equation 83 · Hardware beyond PPA

Study. Units 6, 23, 28, 30. **Structure.** placement surrogate.

Reading. Choose placement coordinates to minimize weighted half-perimeter wirelength.

Local symbols. x_i,y_i: coordinates; E: set of nets; w_e: net weight; HPWL: half-perimeter wirelength. Density, block, routing, and sign-off requirements remain essential.

$$\min_{(x_i,y_i)} \sum_{e\in E} w_e\ \mathrm{HPWL}(e), \tag{V.83}$$

Volume equation 84 · Processes, manufacturing, and safe intensification

Study. Units 4, 9, 23. **Structure.** process equations.

Reading. Set process-model residuals to zero for the design and operating state.

Local symbols. F: residual map; z: temperatures, pressures, flows, compositions; x: design or control choices. A steady-state solution need not be dynamically operable.

$$F(z, x) = 0, \tag{V.84}$$

Volume equation 85 · Energy, climate, conservation, and circularity

Study. Units 3, 6, 23. **Structure.** capacity planning.

Reading. Minimize discounted system cost over the planning horizon.

Local symbols. x_t: period decisions; C_t: period cost; r: discount rate; T: horizon. Operational adequacy requires separate chronological constraints and validation.

$$\min_{x} \sum_{t=0}^{T} \frac{C_t(x_t)}{(1+r)^t} \tag{V.85}$$

Volume equation 86 · Energy, climate, conservation, and circularity

Study. Units 6, 11, 23. **Structure.** network balance.

Reading. At each node, incoming flow plus supply equals outgoing flow plus demand.

Local symbols. f_{ij}: arc flow; s_i,d_i: supply and demand; E,V: edges and nodes. Domain-specific physics and losses supplement this balance.

$$\sum_{j:(j,i)\in E} f_{ji} + s_i = \sum_{j:(i,j)\in E} f_{ij} + d_i, \qquad i \in V, \tag{V.86}$$

Volume equation 87 · Robotics and transportation in open environments

Study. Units 16, 17, 33. **Structure.** trajectory cost.

Reading. Add terminal consequence to running cost along a trajectory.

Local symbols. x(t): state; u(t): control; Φ: terminal cost; L: integrand; T: terminal time. Dynamics and path constraints define the admitted trajectories.

$$J = \Phi\big(x(T)\big) + \int_0^T L\,(x(t), u(t), t)\,dt \tag{V.87}$$

Volume equation 88 · Evolution and contextual biological optimality

Study. Units 5, 34. **Structure.** frequency-dependent fitness.

Reading. Fitness of a type depends on the population composition.

Local symbols. w_i: fitness of type i; p: vector of population frequencies. The landscape changes as the population changes.

$$w_i = w_i(p), \tag{V.88}$$

Volume equation 89 · Evolution and contextual biological optimality

Study. Units 6, 16, 34. **Structure.** replicator dynamics.

Reading. A type's frequency grows when its fitness exceeds mean population fitness.

Local symbols. p_i: frequency; $w_i(p)$: fitness; mean w: frequency-weighted mean; m: type count. This is a dynamic update law, not an objective to minimize.

$$\dot{p}_i = p_i\big(w_i(p) - \bar{w}(p)\big), \qquad \bar{w}(p) = \textstyle\sum_{j=1}^{m} p_j\, w_j(p). \tag{V.89}$$

Volume equation 90 · Evolution and contextual biological optimality

Study. Units 9, 23, 27. **Structure.** flux balance.

Reading. Require steady-state reaction balance and lower and upper flux bounds.

Local symbols. S: stoichiometric matrix; v: reaction fluxes; l,u: bounds. A separate objective, such as biomass, chooses among feasible fluxes.

$$Sv = 0, \qquad l \le v \le u, \tag{V.90}$$

Volume equation 91 · Clinical decisions and treatment policies

Study. Units 19, 20, 23. **Structure.** clinical utility.

Reading. Average the utility of an action over the two disease states using conditional probabilities.

Local symbols. D: binary disease variable; script D={0,1}: state set; a: action; x: clinical information; U: consequence utility. Consent and clinical limits constrain choice.

$$EU(a \mid x) = \textstyle\sum_{d \in \mathcal{D}} P\,(D = d \mid x)\, U(a, d; x). \tag{V.91}$$

Volume equation 92 · Clinical decisions and treatment policies

Study. Units 2, 20. **Structure.** Bayesian odds.

Reading. Posterior disease odds equal the likelihood ratio times prior disease odds.

Local symbols. D: disease event; T: test evidence; ¬D: absence of disease. Denominators must be positive for the displayed odds form.

$$\frac{P(D|T)}{1-P(D|T)} = \frac{P(T|D)}{P(T|\neg D)} \frac{P(D)}{1-P(D)}. \tag{V.92}$$

Volume equation 93 · Clinical decisions and treatment policies

Study. Units 1, 16. **Structure.** health-state accumulation.

Reading. Integrate health-state utility over time to obtain quality-adjusted duration.

Local symbols. u(t): health-state utility; T: duration; Q: QALYs under this valuation. The scalar does not settle ethical allocation questions.

$$Q = \int_0^T u\,(t)\,dt, \tag{V.93}$$

Volume equation 94 · Public health and allocation

Study. Units 6, 23, 31. **Structure.** health allocation.

Reading. Allocate resources to maximize total modeled health gain.

Local symbols. x_i: allocation; B_i: benefit function. Budget, workforce, geography, eligibility, and service floors constrain the decision.

$$\max_x \sum_{i=1}^{n} B_i\,(x_i) \tag{V.94}$$

Volume equation 95 · Biomedical discovery and translation

Study. Units 5, 23, 31. **Structure.** dose benefit and harm.

Reading. Represent dose utility as benefit minus weighted harm.

Local symbols. d: dose; B,H: uncertain benefit and harm; λ: trade-off weight. This schematic function is not a dosing recommendation.

$$U(d) = B(d) - \lambda H(d), \tag{V.95}$$

Volume equation 96 · Markets, finance, and endogenous organizations

Study. Units 9, 23. **Structure.** consumer choice.

Reading. Choose a nonnegative bundle maximizing utility while expenditure stays within income.

Local symbols. x: bundle; p: prices; m: income; u: utility. The budget reflects endowments and institutions as well as scarcity.

$$\max_{x \geq 0} u(x) \quad \text{subject to} \quad p^T x \leq m. \tag{V.96}$$

Volume equation 97 · Markets, finance, and endogenous organizations

Study. Units 9, 19, 23, 27. **Structure.** mean-variance portfolio.

Reading. Choose fully invested weights to trade portfolio variance against expected return.

Local symbols. w: portfolio weights; Σ: covariance; μ: expected returns; λ: trade-off weight; $1^T w=1$: full investment. Estimation and stress risk require separate checks.

$$\min_{w} \; 1/2\, w^T \Sigma w - \lambda \mu^T w, \qquad \mathbf{1}^T w = 1. \tag{V.97}$$

Volume equation 98 · Operations research as institutional engineering

Study. Units 19, 23, 32. **Structure.** newsvendor.

Reading. Choose an order quantity minimizing expected overage and shortage costs.

Local symbols. q: order; D: demand; c_o,c_u: unit overage and underage costs; positive part: max(value,0). $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{q} \; c_o\, \mathbb{E}[(q-D)_+] + c_u\, \mathbb{E}[(D-q)_+]. \tag{V.98}$$

Volume equation 99 · Operations research as institutional engineering

Study. Units 18, 23, 32. **Structure.** critical fractile.

Reading. Under the continuity assumptions, an optimal order quantity lies at the stated demand-distribution fractile.

Local symbols. F_D: demand CDF; q*: optimal quantity; $c_u/(c_u+c_o)$: critical fractile. The equality describes q*, not a decision to choose the probability itself.

$$F_D(q^*) = \frac{c_u}{c_u + c_o}. \tag{V.99}$$

Volume equation 100 · Policy, law, delivery, and accountability

Study. Units 19, 21. **Structure.** causal estimand.

Reading. Define average treatment effect as the mean difference between potential outcomes.

Local symbols. Y(1),Y(0): outcomes under treatment and control. Only one is observed for a given unit; identification requires additional assumptions or design. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\mathrm{ATE} = \mathbb{E}[Y(1) - Y(0)]. \tag{V.100}$$

Volume equation 101 · Ethics, fairness, and moral uncertainty

Study. Units 23, 31. **Structure.** maximin.

Reading. Choose an alternative maximizing the lowest individual utility.

Local symbols. u_i: individual utility; x: allocation or policy. This captures one protective intuition, not the full Rawlsian theory.

$$\max_{x} \min_{i} u_i(x), \tag{V.101}$$

Volume equation 102 · Humanities, interpretation, and historical path dependence

Study. Units 3, 6, 20, 23. **Structure.** conditional likelihood.

Reading. Fit parameters to maximize the sum of conditional log probabilities of observed tokens.

Local symbols. θ: parameters; w_t: token; $w_{<t}$: prior tokens; T: sequence length. Likelihood does not establish truth, literary value, or legitimate use.

$$\max_{\theta} \sum_{t=1}^{T} \log p_\theta(w_t \mid w_{<t}), \tag{V.102}$$

Volume equation 103 · Music: geometry, composition, performance, and listening

Study. Units 3, 5, 6. **Structure.** twelve-tone equal temperament.

Reading. Multiply a reference frequency by a factor of two for each twelve semitones.

Local symbols. f_0: reference frequency; n: semitone displacement; f_n: resulting frequency. Other temperaments use different relations.

$$f_n = f_0\, 2^{n/12}. \tag{V.103}$$

Volume equation 104 · Music: geometry, composition, performance, and listening

Study. Units 7, 10, 28, 38. **Structure.** voice-leading geometry.

Reading. Minimize weighted pitch movement over voice matching and octave shifts.

Local symbols. p,q: pitch vectors; π: permutation, not policy; k_i: integer octave shifts; $w_i \geq 0$: voice weights; m: number of voices. The cost omits expressive function.

$$C(p,q) = \min_{\pi,k} \sum_{i=1}^{m} w_i \left| p_i - q_{\pi(i)} - 12k_i \right|, \tag{V.104}$$

Volume equation 105 · Games, sport, rules, and training

Study. Units 9, 23, 34. **Structure.** minimax theorem.

Reading. In a finite two-player zero-sum game, maximizing the guaranteed payoff equals minimizing the opponent's best payoff.

Local symbols. A: payoff matrix; p,q: mixed strategies in probability simplexes. The equality does not extend to arbitrary multi-player games.

$$\max_{p} \min_{q} p^T A q = \min_{q} \max_{p} p^T A q. \tag{V.105}$$

Volume equation 106 · Games, sport, rules, and training

Study. Units 3, 6. **Structure.** fitness-fatigue model.

Reading. Add decaying positive adaptation and subtract decaying fatigue responses to prior training impulses.

Local symbols. $R(t)$: modeled readiness at time t; R_0: baseline; u_i: training impulse at time i; $k_1, k_2 > 0$: gains; $\tau_1, \tau_2 > 0$: decay time constants for adaptation and fatigue. The example uses $\tau_1 > \tau_2$ for slower adaptation and $k_2 > k_1$ for initially larger fatigue. Parameters require individual evidence.

$$R(t) = R_0 + k_1 \sum_{i=1}^{t-1} u_i \, e^{-(t-i)/\tau_1} - k_2 \sum_{i=1}^{t-1} u_i \, e^{-(t-i)/\tau_2}. \tag{V.106}$$

Volume equation 107 · Evidence-producing optimization

Study. Units 9, 14, 15, 21. **Structure.** response surface.

Reading. Approximate response by intercept, linear terms, pure quadratics, pairwise interactions, and regression noise.

Local symbols. y: response; x_i: factor; β: coefficients; p: number of factors; e: regression error. Experimental design determines which effects can be estimated.

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i \, x_i + \sum_{i=1}^{p} \beta_{ii} \, x_i^2 + \sum_{i=1}^{p-1} \sum_{j=i+1}^{p} \beta_{ij} \, x_i x_j + e. \tag{V.107}$$

Volume equation 108 · Evidence-producing optimization

Study. Units 19, 23, 30. **Structure.** simulation optimization.

Reading. Define J as expected simulated cost, then minimize it over the decision.

Local symbols. x: decision; ξ: random simulation input; g(x,ξ): realized cost; $\mathbb{E}$: expectation. The definition of J and its minimization are distinct operations.

$$\min_x J(x), \qquad J(x) = \mathbb{E}[g(x, \xi)]. \tag{V.108}$$

Volume equation 109 · Failure modes, review, and purpose

Study. Units 19, 21, 23. **Structure.** optimizer's curse.

Reading. The selected estimate has positive expected error under the example's equal-value, independent, identically distributed, nondegenerate unbiased-noise assumptions.

Local symbols. M: measured proxy; V: true value; ε: error; x-hat: selected maximizer; $\mathcal{X}$: candidate set. At least two candidates and finite expectations are required; unequal values require a further joint-model analysis. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\mathbb{E}[\varepsilon(\hat{x})] > 0, \qquad \hat{x} \in \arg\max_{x \in \mathcal{X}} M(x). \tag{V.109}$$

Volume equation 110 · Case A · A low-carbon, water-aware, resilient data center

Study. Units 1, 6, 23, 31. **Structure.** data-center objectives.

Reading. Minimize a weighted sum of electricity cost, operational emissions, scarcity-weighted water use, service degradation, and wear.

Local symbols. p, E, c, W, ω, D, H: price, electricity use, emissions factor, water use, scarcity factor, service degradation, and wear; $\lambda_c, \lambda_w, \lambda_d, \lambda_h$: weights for emissions, water, degradation, and wear; x, z, T, b: assignment, powered-fraction, cooling, and battery decisions; s, t: site and period. Here E is electricity use.

$$\min_{x,z,T,b} \sum_{s,t} [p_{st} E_{st} + \lambda_c c_{st} E_{st} + \lambda_w \omega_{st} W_{st} + \lambda_d D_{st} + \lambda_h H_{st}]. \tag{V.110}$$

Volume equation 111 · Case A · A low-carbon, water-aware, resilient data center

Study. Units 9, 23, 28. **Structure.** powered capacity.

Reading. For every resource, site, and period, assigned demand cannot exceed the powered portion of installed capacity.

Local symbols. a_{jk}: job-resource demand; x_{jst}: assignment; C_{kst}: installed capacity; $z_{st} \in [0,1]$: powered fraction, binary for on/off use. This is a feasibility inequality.

$$\sum_j a_{jk} x_{jst} \le C_{kst} z_{st} \qquad \text{for each resource } k, \text{ site } s, \text{ and time } t. \tag{V.111}$$

Volume equation 112 · Case B · An equitable heat-health program

Study. Units 19, 23, 31. **Structure.** population risk.

Reading. Choose a heat-health portfolio minimizing population-weighted expected residual risk.

Local symbols. g: group index; N_g: population; R_g: residual risk; ξ: scenario; x: full portfolio including service allocations. The mean alone does not enforce group protection. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_x \sum_g N_g \ \mathbb{E}\big[R_g(x, \xi)\big]. \tag{V.112}$$

Volume equation 113 · Case B · An equitable heat-health program

Study. Units 19, 23, 31, 32. **Structure.** heat-health risk floors.

Reading. Minimize total expected harm plus tail harm among accessible portfolios satisfying budget and priority-group limits.

Local symbols. X_H: feasible portfolios; L=Σ_g N_g R_g: total population harm; C≤B: cost budget; λ,α: CVaR weight and level. Both risk limits apply to the designated priority groups. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\min_{x\in X_H}\Big(\sum_g N_g\ \mathbb{E}\big[R_g(x,\xi)\big]+\lambda\,\mathrm{CVaR}_\alpha\big(L(x,\xi)\big)\Big). \tag{V.113}$$

Volume equation 114 · Case C · Closed-loop insulin delivery as adaptive care

Study. Units 6, 23, 32, 33. **Structure.** receding-horizon care.

Reading. Choose a feasible dose sequence balancing predicted range violations against dose changes.

Local symbols. ϕ(G)=w_h(G−Ghigh)$_+^2$+w_l(Glow−G)$_+^2$; r≥0: smoothing weight; H: horizon; u_{t-1}: last applied dose. Dose bounds use k=0,…,H−1; state and insulin-on-board conditions use k=1,…,H. This is an illustrative architecture.

$$\min_{u_{t:t+H-1}\in\mathcal{U}_t}\sum_{k=1}^{H}[\phi(G_{t+k})+r\,(u_{t+k-1}-u_{t+k-2})^2]. \tag{V.114}$$

Volume equation 115 · Case D · A public library collection and discovery system

Study. Units 23, 28, 31. **Structure.** collection portfolio.

Reading. Maximize weighted use, breadth, research support, and serendipity minus cost inside the protected collection set.

Local symbols. x: collection portfolio; X_L: licensed feasible portfolios satisfying representation, access, preservation, and budget requirements; U, B, Q, S, C: use, breadth, research support, serendipity, and cost; $\lambda_B, \lambda_Q, \lambda_S, \lambda_C$: their displayed weights. $B(x)$ denotes breadth, distinct from the monetary budget $B_{\max}$.

$$\max_{x\in X_L}\big[U(x)+\lambda_B B(x)+\lambda_Q Q(x)+\lambda_S S(x)-\lambda_C C(x)\big]. \tag{V.115}$$

Volume equation 116 · Case E · A musical prosthesis co-designed as an instrument

Study. Units 23, 31, 33. **Structure.** constrained vector objective; control policy; repertoire coverage.

Reading. Retain a vector of expression, learning, fatigue, mass, and cost criteria subject to safety and repertoire coverage.

Local symbols. x: hardware; π:(q,c)↦y: control mapping; E,L,F,m,C: criteria; script R: repertoire; A_r≥a_r: coverage. Rrel denotes reliability elsewhere. Vector minimization does not impose one best score.

$$\begin{aligned} &\min_{x,\pi} \big(-E(x,\pi),\ L(x,\pi),\ F(x,\pi),\ m(x),\ C(x,\pi)\big) \\ &\text{subject to} \quad g_{\text{safety}}(x) \le 0,\ A_r(x,\pi) \ge \underline{a}_r\ \forall r \in \mathcal{R}. \end{aligned} \tag{V.116}$$

Volume equation 117 · Case F · A yield-aware, energy-aware semiconductor fabrication system

Study. Units 6, 23, 28, 31, 35. **Structure.** fabrication scheduling.

Reading. Minimize tardiness, work in process, energy, validated yield risk, and schedule churn within production constraints.

Local symbols. x: full production decision vector; l, t: lot and period; T_l: tardiness; w_l: lot tardiness weight; W_t: work in process; E_t: energy; Q_l: validated yield-risk indicator; $\lambda_W, \lambda_E, \lambda_Q, \lambda_\Delta$: weights for work in process, energy, yield risk, and schedule change; x^{prev}: previous schedule. The ℓ_1 term measures schedule change; critical yield and safety limits remain constraints or escalation gates.

$$\min_x \left[\sum_l w_l\, T_l(x) + \lambda_W \sum_t W_t\,(x) + \lambda_E \sum_t E_t\,(x) + \lambda_Q \sum_l Q_l\,(x) + \lambda_\Delta \parallel x - x^{\text{prev}} \parallel_1\right] \tag{V.117}$$

Volume equation 118 · Case G · A coastal adaptation pathway under deep uncertainty

Study. Units 19, 23, 31, 32, 33. **Structure.** adaptive pathways; regret.

Reading. Minimize a stated reference-ensemble average plus worst regret over admitted futures, within the complete feasible policy set.

Local symbols. π: policy; Π_{safe}: policies meeting capability floors, service-risk limits, action requirements, and dynamics; $\xi \in \Xi$: admitted scenario trajectory; L: consequence loss; C: lifecycle cost; $J = L + C$: valued scenario cost; J^*: best feasible cost with scenario hindsight; regret weight λ. $\mathbb{E}_\xi$ is the mean under the declared reference ensemble, not a known true joint law.

$$\min_{\pi \in \Pi_{\text{safe}}} \Big(\mathbb{E}_\xi[L(\pi,\xi) + C(\pi,\xi)] + \lambda \max_{\xi \in \Xi}[J(\pi,\xi) - J^*(\xi)]\Big). \tag{V.118}$$

Volume equation 119 · Chapter 20 · Stopping and refusal

Study. Units 19, 23, 32. **Structure.** optimal stopping.

Reading. Continue search only when its conditional expected decision value exceeds computation, delay, and risk costs.

Local symbols. I_n: current information; ΔV: decision-relevant improvement; C: distinct costs. A numerical improvement alone need not change the decision. $\mathbb{E}$: expectation under the stated law and conditioning.

$$\text{continue search only if} \qquad \mathbb{E}[\Delta V_{n+1} \mid \mathcal{I}_n] > C_{\text{compute}} + C_{\text{delay}} + C_{\text{risk}}, \qquad \text{(V.119)}$$

Volume equation 120 · Differential quantities

Study. Units 9, 14. **Structure.** gradient definition.

Reading. Collect coordinate partial derivatives into the gradient vector.

Local symbols. f: scalar function; $x_1,\ldots,x_n$: coordinates; superscript T: transpose. This defines ∇f; it does not assert that the gradient is zero.

$$\nabla f(x) = [\partial f / \partial x_1 \quad \cdots \quad \partial f / \partial x_n]^T. \qquad \text{(V.120)}$$

Volume equation 121 · Differential quantities

Study. Units 12, 13, 14. **Structure.** directional derivative.

Reading. Take the one-sided limit of the difference quotient along direction d.

Local symbols. x: base point; d: direction; t↓0: positive step tending to zero. Existence depends on the function and direction.

$$Df(x)[d] = \lim_{t \downarrow 0} \frac{f(x+td)-f(x)}{t}. \qquad \text{(V.121)}$$

Volume equation 122 · Probability and statistics

Study. Unit 19. **Structure.** moments.

Reading. Expectation is the probability-weighted mean; variance is mean squared deviation from the mean; covariance is the mean product of centered variables.

Local symbols. X,Y: random variables; 𝔼: expectation; Var,Cov: variance and covariance. The relevant moments must exist.

$$\mathbb{E}[X], \qquad \operatorname{Var}(X) = \mathbb{E}[(X - \mathbb{E}X)^2], \qquad \operatorname{Cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)]. \qquad \text{(V.122)}$$

Volume equation 123 · Probability and statistics

Study. Units 18, 20. **Structure.** Bayes' rule.

Reading. Update prior probability by likelihood and divide by the evidence probability.

Local symbols. A: hypothesis event; B: evidence event with P(B)>0. This is Bayes' rule, not the definition of conditional probability.

$$P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}. \qquad \text{(V.123)}$$

Volume equation 124 · Optimization statements

Study. Units 5, 7, 23. **Structure.** optimization notation.

Reading. A chosen solution belongs to the set of feasible arguments attaining the smallest objective value.

Local symbols. x^*: selected minimizer; $\mathcal{X}$: feasible set; f: objective. Argmin returns a set; min returns its attained value.

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x) \tag{V.124}$$

Part F.10 · Notation concordance

Notation is typed: a symbol's role is determined by its local definition and mathematical object. The entries below record recurring conventions in the volume and appendix. When a local definition differs, the local definition has priority.

x. a decision, state, or generic variable; local text determines the role.

y. an output, response, path, or second variable.

z. an auxiliary or third variable.

t. time or a continuous index.

i. an item, component, sample, or constraint index.

j. a second index, commonly for equality constraints or states.

k. an iteration, time step, or generic index.

m. a count, dimension, or slope when defined locally.

n. sample size, dimension, horizon, or problem-size parameter.

p. probability, dimension, exponent, density, or primal value according to context.

q. a dual function, probability, requirement count, or auxiliary quantity; the local role in the text accompanying Equation (6) differs from the dual function in Equation (23).

f. an objective or generic function.

g. an inequality constraint or generic function.

h. an equality constraint or generic function.

F. a vector-valued map, aggregate objective, transition model, or functor.

G. a functor or generic map.

J. an objective functional, Jacobian, or performance criterion.

T. a transition map, technology matrix, time horizon, runtime, latency percentile, or tardiness; a superscript T denotes transpose. Each equation fixes its local role.

L. a Lagrangian function or integrand, loss, queue length, or bound.

M. a finite penalty coefficient or maintainability measure, according to local context.

A. a matrix, event, set, scalar availability, or linear operator.

B. a matrix, set, or budget.

C. a set, cost, covariance, or constant.

H. a hypothesis, Hessian, entropy, or Hamiltonian.

P. probability, transition matrix, distribution, or an open component.

Q. a quadratic matrix, action-value function, or distribution.

R. a reward, requirement, relation, or real-number set.

U. utility, an uncertainty set, a control set, or an upper bound; local definitions distinguish these uses.

V. a vector space or value function.

X. a random variable, matrix, state space, or set; distinguish it from the calligraphic feasible set below.

$\mathcal{X}$**.** a feasible or decision set; local definitions specify the admissible alternatives.

α. a step size, confidence/tail level, weight, or scalar.

β. a coefficient, weight, or scalar.

γ. a discount factor, coupling, or scalar.

ε. measurement or estimation error, a tolerance, an accuracy or privacy parameter, or a regularization weight; regression noise is denoted by e in Equation (107).

η. a threshold, auxiliary optimization variable, or natural transformation.

θ. a parameter, mixture weight, or preference parameter.

λ. a Lagrange multiplier, arrival rate, eigenvalue, or scalar.

μ. a probability measure, service rate, mean, or distribution.

ν. an equality multiplier or probability measure.

ξ. an uncertain quantity or scenario.

π. a policy, a probability, a permutation in voice matching, or the circle constant; local definitions distinguish these uses.

Π. a policy set or a set of couplings.

ρ. a risk functional, material density, utilization, correlation, or approximation ratio; local definitions distinguish these uses.

σ. standard deviation, singular value, noise scale, or strategy.

Φ. a terminal cost.

Ω. a sample space or domain.

E. expectation under a stated probability law.

∇. gradient or derivative operator.

∂. a partial-derivative symbol or subdifferential operator, according to context.

∑. summation over the displayed index range.

∫. integration over the displayed domain.

Part F.11 · Mathematics glossary

Adjoint. An operator that transfers a linear map across an inner product; central to efficient sensitivity calculations.

Affine. A linear expression plus a constant offset.

Ambiguity set. A set of probability distributions considered plausible in distributionally robust optimization.

Argmin / argmax. The set of decision values attaining the minimum or maximum. This differs from the attained objective value.

Bayes' rule. The identity that updates a prior using a likelihood and normalizing evidence.

Bellman equation. A recursive relation expressing the value of a state as immediate reward or cost plus the optimized value of successor states.

Binary variable. A decision variable restricted to zero or one.

Certificate. Checkable evidence for a claim such as optimality, a bound, feasibility, infeasibility, or program correctness.

Compact set. A set with the finite-cover property; in Euclidean space, equivalently closed and bounded.

Complementary slackness. The KKT relation stating that an inequality's slack and multiplier cannot both be nonzero.

Condition number. A measure of sensitivity of a problem or computation to perturbations; large values indicate error amplification.

Conditional probability. Probability evaluated within the event specified after the conditioning bar.

Convex function. A function whose value at a mixture is no greater than the corresponding mixture of values. Local minima are global on convex feasible sets.

Convex set. A set containing every line segment between any two of its points.

Correlation. Standardized covariance; a measure of linear association, not causation.

Covariance. Expected product of centered random variables, measuring joint linear variation.

CVaR (conditional value at risk). The mean loss in a specified upper tail, with conventions varying at discontinuities.

Derivative. A limiting local rate of change.

Differential. The linear map giving first-order change; for scalar functions it is naturally a covector.

Dual problem. A related optimization problem that provides bounds, sensitivities, decomposition, or alternative interpretation of the primal.

Entropy. Expected self-information of a probability distribution.

Expectation. Probability-weighted average of a random variable.

Feasible set. All decisions satisfying the modeled constraints.

Functional. A mapping from a function space to numbers or another scalar-like codomain.

Functor. A mapping between categories that maps objects and morphisms while preserving identities and composition.

Gradient. The vector representing the differential of a scalar function under a chosen inner product; in Euclidean space it points toward steepest local increase.

Hessian. The matrix or bilinear form of second derivatives of a scalar function.

Inner product. A pairing that measures alignment and induces a norm.

Jacobian. The derivative matrix of a vector-valued function.

KKT conditions. Stationarity, primal feasibility, dual feasibility, and complementary slackness conditions for constrained optimization.

KL divergence. A directed information discrepancy between probability distributions; not a metric.

Lagrangian. The objective augmented by constraint functions weighted by multipliers.

Likelihood. Observed-data probability or density viewed as a function of model parameters.

Limit. The value approached by a function or sequence under a specified mode of approach.

Markov property. Conditional independence of the future from earlier history given the present state.

Metric. A distance satisfying nonnegativity, identity, symmetry, and the triangle inequality.

Morphism. An arrow or structure-preserving process between objects in a category.

Natural transformation. A coherent family of morphisms comparing two functors.

Norm. A magnitude function obeying positivity, homogeneity, and the triangle inequality.

Objective function. A mapping used to rank or compare feasible decisions. It is an operational representation of purpose, not purpose itself.

Pareto frontier (also Pareto front). The set of nondominated feasible outcomes in a multiobjective problem.

Positive semidefinite. Of a symmetric matrix: having nonnegative quadratic form in every direction.

Posterior. A probability distribution after incorporating observed evidence.

Prior. A probability distribution before the current evidence is incorporated.

Projection. A nearest-point map under a specified geometry.

Random variable. A measurable numerical function of an uncertain outcome.

Residual. A discrepancy remaining after fitting, solving, or enforcing an equation.

Robust optimization. Optimization against all values in a specified uncertainty set, often emphasizing worst-case feasibility or performance.

Scalarization. Conversion of multiple objectives into one scalar, for example by weighted sum. It embeds preferences and may miss parts of a nonconvex frontier.

Shadow price. A dual multiplier interpreted as the local marginal value of relaxing a constraint, under regularity and stated units.

Stochastic optimization. Optimization involving random variables, samples, scenarios, or noisy observations.

Surrogate model. A cheaper approximation to an expensive experiment, simulator, or objective.

Topology. Neighborhood and continuity structure independent of a particular coordinate distance.

Trust region. A neighborhood in which a local model is considered reliable; steps are accepted based on predicted versus realized improvement.

Uncertainty set. A collection of plausible parameter values used in robust optimization.

Variance. Expected squared deviation from a random variable’s mean.

Wasserstein distance. A distance between probability measures induced by optimal transport cost.

Part F.12 · Mathematical concept index

This index uses stable appendix unit locators rather than page numbers, so it remains valid under later repagination.

Algebra. Units 3–7; Equation atlas.

Algorithms. Units 28–30 and 41.

Bayes' rule. Unit 20.

Calculus. Units 12–17.

Category theory. Unit 40.

Causality. Unit 21.

Certificates. Units 25–27, 29, and 41.

Combinatorics. Units 11 and 28.

Complexity. Units 11, 28, and 41.

Conditioning. Units 9 and 15.

Convexity. Units 24–27.

Derivatives. Units 13–15.

Differential equations. Units 16–17 and 33.

Discrete optimization. Units 11 and 28.

Duality. Units 25–27 and 37.

Dynamic programming. Units 22 and 33.

Entropy. Unit 39.

Equations in the volume. Equation atlas, Volume equations 1–124.

Evidence. Units 18–22 and 41.

Formulation. Unit 23.

Functions. Units 5, 12–17, and 37.

Games. Unit 34.

Geometry. Units 8–10, 14–15, 24–25, and 38.

Gradients. Units 14–15 and 29.

Graphs and networks. Unit 11.

Information theory. Unit 39.

Integration. Units 16–17.

KKT conditions. Unit 25.

Part F.13 · Appendix references and validation ledger

Reference validation dates: the original appendix review was 16 August 2026; the OpenStax, Luenberger, and Munkres edition details were rechecked on 12 September 2026. The 2023 Munkres eTextbook ISBN is retained with its format made explicit. Entries reused from the main bibliography follow its audit trail.

New pedagogical entries are checked against official publisher, author, DOI, or approved-open-textbook records. Identifiers are included only when they resolve to the cited work; a missing DOI is not replaced by a guessed identifier.

[F1] Lynn Marecek, MaryAnne Anthony-Smith, and Andrea Honeycutt Mathis. Prealgebra 2e. OpenStax, 2020. ISBN 978-1-951693-19-0. https://openstax.org/details/books/prealgebra-2e *Validation: official publisher page.*

[F2] Jay Abramson. Algebra and Trigonometry 2e. OpenStax, 2021. https://openstax.org/books/algebra-and-trigonometry-2e/pages/1-introduction-to-prerequisites *Validation: official publisher text and citation metadata.*

[F3] Gilbert Strang and Edwin “Jed” Herman. Calculus, Volumes 1–3. OpenStax, 2016. Volume 1 ISBN 978-1-947172-13-5; Volume 2 ISBN 978-1-947172-14-2; Volume 3 ISBN 978-1-947172-16-6. Volume 1; Volume 2; Volume 3. *Validation: official OpenStax book records and Volume 2 publication metadata.*

[F4] Richard Hammack. Book of Proof, 3rd ed. Virginia Commonwealth University, 2018. https://richardhammack.github.io/BookOfProof/ *Validation: author/AIM approved-textbook record.*

[F5] Gilbert Strang. Introduction to Linear Algebra, 6th ed. Wellesley-Cambridge Press, 2023. ISBN 978-1-7331466-7-8. https://epubs.siam.org/doi/abs/10.1137/1.9781733146678 *Validation: SIAM bibliographic record.*

[F6] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, 4th ed. MIT Press, 2022. ISBN 978-0-262-04630-5. https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/ *Validation: official MIT Press catalog record and ISBN checksum.*

[F7] Michael R. Garey and David S. Johnson. Computers and Intractability. W. H. Freeman, 1979. ISBN 978-0-7167-1045-5. *Validation: ISBN checksum and bibliographic identity.*

[F8] Barbara Illowsky and Susan Dean. Introductory Statistics 2e. OpenStax, 2023. ISBN 978-1-961584-32-7. https://openstax.org/details/books/introductory-statistics-2e *Validation: official publisher page.*

[F9] Stephen Boyd and Lieven Vandenberghe. Convex Optimization. Cambridge University Press, 2004. doi:10.1017/CBO9780511804441. *Validation: DOI resolution and Cambridge University Press metadata.*

[F10] Jorge Nocedal and Stephen J. Wright. Numerical Optimization, 2nd ed. Springer, 2006. doi:10.1007/978-0-387-40065-5. *Validation: DOI resolution and Springer metadata.*

[F11] Miguel A. Hernán and James M. Robins. Causal Inference: What If. Chapman & Hall/CRC, 2020. https://miguelhernan.org/whatifbook *Validation: author's official book page.*

[F12] Patrick Billingsley. Probability and Measure, 3rd ed. Wiley, 1995. ISBN 978-0-471-00710-4. *Validation: publisher bibliographic identity.*

[F13] Martin L. Puterman. Markov Decision Processes. Wiley, 1994. doi:10.1002/9780470316887. *Validation: DOI resolution and Wiley metadata.*

[F14] David G. Luenberger. Optimization by Vector Space Methods. Wiley, 1969; paperback reprint, 1997. ISBN 978-0-471-18117-0. Wiley record. *Validation: the publisher dates this ISBN to January 1997.*

[F15] R. Tyrrell Rockafellar. Convex Analysis. Princeton University Press, 1970. ISBN 978-0-691-01586-6. The ISBN identifies the 1997 first paperback edition. *Validation: Princeton University Press catalog record and ISBN checksum.*

[F16] Sébastien Bubeck. Convex Optimization: Algorithms and Complexity. Foundations and Trends in Machine Learning 8(3–4), 2015. doi:10.1561/2200000050. *Validation: DOI resolution and journal metadata.*

[F17] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. Lectures on Stochastic Programming, 3rd ed. SIAM, 2021. doi:10.1137/1.9781611976595. *Validation: DOI resolution and SIAM metadata; used for stochastic, distributional, and risk foundations.*

[F18] Kaisa Miettinen. Nonlinear Multiobjective Optimization. Kluwer, 1998. doi:10.1007/978-1-4615-5563-6. *Validation: DOI resolution and Springer metadata.*

[F19] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. Robust Optimization. Princeton University Press, 2009. doi:10.1515/9781400831050. *Validation: DOI resolution and publisher metadata.*

[F20] R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of Conditional Value-at-Risk. Journal of Risk 2(3), 2000. doi:10.21314/JOR.2000.038. *Validation: DOI resolution and journal metadata.*

[F21] Richard Bellman. Dynamic Programming. Princeton University Press, 1957. ISBN 978-0-691-14668-3. The ISBN identifies the 2010 Princeton Landmarks paperback edition. *Validation: Princeton University Press catalog record and ISBN checksum.*

[F22] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction, 2nd ed. MIT Press, 2018. ISBN 978-0-262-03924-6. *Validation: MIT Press catalog record and ISBN checksum.*

[F23] Lev S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. The Mathematical Theory of Optimal Processes. Interscience, 1962. *Validation: publisher and library catalog metadata.*

[F24] Rudolf E. Kalman. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering 82(1), 1960, 35–45. doi:10.1115/1.3662552. *Validation: DOI resolution and ASME journal metadata.*

[F25] John F. Nash. Equilibrium Points in n-Person Games. PNAS 36(1), 1950, 48–49. doi:10.1073/pnas.36.1.48. *Validation: DOI resolution and PNAS metadata.*

[F26] John D. C. Little. A Proof for the Queuing Formula: L = λW. Operations Research 9(3), 1961, 383–387. doi:10.1287/opre.9.3.383. *Validation: DOI resolution and INFORMS journal metadata.*

[F27] John F. Shortle, James M. Thompson, Donald Gross, and Carl M. Harris. Fundamentals of Queueing Theory, 5th ed. Wiley, 2018. doi:10.1002/9781119453765. *Validation: official Wiley DOI record.*

[F28] James R. Munkres. Topology, 2nd ed., Classic Version, eTextbook update. Pearson, 2023. ISBN 978-0-13-784866-9. Pearson eTextbook record. *Validation: this ISBN identifies the 2023 eTextbook update (copyright 2024); the 2017 print Classic Version has a different ISBN.*

[F29] Cédric Villani. Topics in Optimal Transportation. American Mathematical Society, 2003. doi:10.1090/gsm/058. *Validation: DOI resolution and American Mathematical Society metadata.*

[F30] Gabriel Peyré and Marco Cuturi. Computational Optimal Transport with Applications to Data Sciences. Foundations and Trends in Machine Learning 11(5–6), 2019. doi:10.1561/2200000073. *Validation: DOI resolution and journal metadata.*

[F31] Claude E. Shannon. A Mathematical Theory of Communication. Bell System Technical Journal 27, 1948, 379–423 and 623–656. doi:10.1002/j.1538-7305.1948.tb01338.x. *Validation: DOI resolution and Wiley journal metadata.*

[F32] Thomas M. Cover and Joy A. Thomas. Elements of Information Theory, 2nd ed. Wiley, 2006. doi:10.1002/047174882X. *Validation: official Wiley DOI record.*

[F33] Saunders Mac Lane. Categories for the Working Mathematician, 2nd ed. Springer, 1998. ISBN 978-0-387-98403-2. *Validation: Springer catalog record and ISBN checksum.*

[F34] Emily Riehl. Category Theory in Context. Dover, 2016. ISBN 978-0-486-80903-8. *Validation: publisher/author bibliographic record.*

[F35] Brendan Fong and David I. Spivak. An Invitation to Applied Category Theory. Cambridge University Press, 2019. doi:10.1017/9781108668804. *Validation: DOI resolution and Cambridge University Press metadata.*

Glossary

Abstraction. A representation that preserves selected structure while suppressing other detail. Its usefulness depends on what the decision requires.

Acquisition function. A criterion computed from a surrogate and the evaluation history to select the next experiment or batch. It can target improvement, uncertainty reduction, feasibility, or a cost-adjusted combination.

Active constraint. An inequality constraint satisfied at equality at a candidate solution. Its multiplier may indicate marginal value.

Active learning. Sequential selection of data or experiments intended to improve a model efficiently.

Adjoint method. A technique for computing the derivative of a scalar output with respect to many inputs by solving a backward or dual system.

Admissible decision. A choice permitted by all hard constraints, rights, and governance requirements; often synonymous with feasible when those requirements are modeled.

Algorithm. A finite operational procedure transforming inputs into outputs. Optimization algorithms search, update, bound, or certify.

Ambiguity set. A set of probability distributions considered plausible in distributionally robust optimization.

Approximation algorithm. An algorithm with a proved bound comparing its solution value with the optimum, typically for a difficult discrete problem.

Argmin / argmax. The set of decision values attaining the minimum or maximum. This differs from the attained objective value.

Asynchronous optimization. Search that proposes a new evaluation when a resource becomes available while other evaluations remain pending.

Bayesian optimization. Sequential optimization that uses a probabilistic model and an acquisition rule to choose evaluations. Its variants address constraints, noise, mixed variables, multiple fidelities, batches, objectives, and preferences. A surrogate alone does not certify global optimality.

Bellman equation. A recursive relation expressing the value of a state as immediate reward or cost plus the optimized value of successor states.

Bilevel optimization. An optimization problem whose constraints include the solution of another optimization problem, often representing a follower or training process.

Black-box optimization. Optimization through an evaluator whose usable interface provides responses or observations without exposing the full function structure. Derivatives may be unavailable even when some constraints or variable relationships are known.

Bounded rationality. Decision-making under limited information, time, attention, and computation. It treats the decision procedure and its costs as part of the problem rather than assuming omniscient maximization.

Branch-and-bound. Exact search that partitions a feasible set and uses bounds to discard regions that cannot improve the incumbent.

Bregman divergence. A generally asymmetric discrepancy generated by a differentiable strictly convex function, used in mirror descent and information geometry.

Calibration. Agreement between predicted probabilities and observed frequencies in relevant groups and conditions.

Category. In category theory, a collection of objects and morphisms with associative composition and identity morphisms.

Certificate. Checkable evidence for a claim such as optimality, a bound, feasibility, infeasibility, or program correctness.

Chance constraint. A requirement that a constraint hold with at least a specified probability.

CMA-ES. Covariance matrix adaptation evolution strategy. A method that adapts the mean, scale, and covariance of a continuous search distribution from sampled candidate rankings.

Co-design. Joint optimization of coupled layers such as hardware and software, product and process, or policy and implementation.

Combinatorial optimization. Optimization over discrete structures such as sets, graphs, assignments, sequences, or partitions.

Compositionality. The property that the behavior or guarantee of a whole can be constructed from components and their interfaces.

Condition number. A measure of sensitivity of a problem or computation to perturbations; large values indicate error amplification.

Constraint. A condition a decision must satisfy. Treating a requirement as a penalty changes its logical status.

Constraint qualification. A regularity condition under which multiplier or duality results such as KKT conditions hold.

Convex function. A function whose value at a mixture is no greater than the corresponding mixture of values. Local minima are global on convex feasible sets.

Convex set. A set containing every line segment between any two of its points.

Counterfactual. A claim about what would occur under an alternative action or condition, central to causal reasoning.

Critical path. A longest-duration precedence path determining a project's earliest completion in a deterministic network model.

Cross-validation. Repeated splitting of data for model selection and performance estimation; valid only when splits respect the deployment dependency structure.

CVaR (conditional value at risk). The mean loss in a specified upper tail, with conventions varying at discontinuities.

Decision variable. A quantity, structure, policy, or artifact the decision maker is permitted to choose.

Derivative-free optimization. Search that does not require analytic gradients, useful for black-box, discontinuous, or noisy objectives.

Differential evolution. Population search that forms trial vectors using scaled differences between candidate vectors, crossover, and selection.

Digital twin. A maintained relationship among a physical or operational entity, data, models, and decisions; not merely a static simulation.

Direct search. Derivative-free search based on comparisons of evaluated points without constructing an explicit gradient estimate. Convergence properties depend on the polling, step, and constraint rules.

Discount rate. A parameter converting future costs or benefits into present value. It can encode opportunity cost and ethical assumptions about time.

Distributionally robust optimization. Optimization against the worst expected value over a specified ambiguity set of probability distributions.

Dominance. In multiobjective minimization, solution x dominates y if it is no worse in every objective and better in at least one.

Dual problem. A related optimization problem that provides bounds, sensitivities, decomposition, or alternative interpretation of the primal.

Duality gap. The difference between primal and dual objective values. A zero or small gap can certify optimality under suitable conditions.

Dynamic programming. Optimization by recursively solving overlapping subproblems indexed by a sufficient state.

Endogeneity. Dependence created within the system, such as behavior changing in response to a policy or metric.

Equilibrium. A state consistent with the decisions or dynamics of interacting agents. Equilibrium need not be efficient, fair, stable, or unique.

Estimand. The precise quantity an analysis intends to estimate, such as an average causal effect for a population.

Expected value of information. The expected improvement in decision quality made possible by obtaining additional information, before subtracting its acquisition cost or burden.

Externality. A cost or benefit imposed on parties not fully represented in the decision or transaction.

Feasible set. All decisions satisfying the modeled constraints.

Feedback. A loop in which outcomes affect later inputs or decisions. Feedback can stabilize, amplify, oscillate, or change the objective's meaning.

Fidelity. The resolution, resource budget, duration, or other evaluation setting that changes cost and the relationship of an observation to the target response.

Fitness. In evolutionary biology, reproductive contribution relative to an environment and population context; not a general measure of worth or perfection.

Functor. A mapping between categories that maps objects and morphisms while preserving identities and composition.

Generalization. Performance of a learned model beyond its training observations under a specified target distribution or range of shift.

Global optimum. A feasible solution no worse than every other feasible solution under the objective ordering.

Goodhart's law. The recurring failure in which a measure loses reliability as a measure when it becomes a target.

Gradient. The vector representing the differential of a scalar function under a chosen inner product; in Euclidean space it points toward steepest local increase.

Hard constraint. A condition that may not be violated. It differs from a soft constraint or penalty, which permits violation at a cost.

Heuristic. A search or decision rule useful in practice without a complete guarantee of optimality.

Hyperparameter. A model or training choice set outside the immediate parameter-fitting problem, often selected in an outer optimization loop.

Identification. Conditions under which a causal or statistical quantity is uniquely determined by observed data and assumptions.

Incumbent. The best feasible solution currently known during a search.

Invariant. A property intended to remain unchanged under a transformation, composition, or optimization step.

KKT conditions. Stationarity, primal feasibility, dual feasibility, and complementary slackness conditions for constrained optimization.

Lagrangian. The objective augmented by constraint functions weighted by multipliers.

Latency. Time between initiating and completing an operation; averages and high percentiles answer different service questions.

Learning rate. The step-size parameter in gradient-based learning or stochastic approximation.

Lexicographic optimization. Optimization in strict priority order: improve a lower-priority objective only without worsening higher-priority objectives.

Lifecycle boundary. The stages and actors included in evaluation, such as extraction, manufacture, use, maintenance, and disposal.

Local optimum. A feasible solution no worse than nearby feasible solutions. It need not be globally best.

Loss function. A numerical representation of error or harm used in estimation or decision-making. It is a designed surrogate, not an objective fact.

MADS. Mesh adaptive direct search. A method using search and poll steps on an adapting mesh, with stationarity results under specified regularity and direction conditions.

Mechanism design. Design of rules or institutions to achieve properties when participants act strategically and hold private information.

Metric. A measure of performance; mathematically, also a distance satisfying specific axioms. The two senses should not be confused.

Mixed-integer program. An optimization model containing both continuous and integer-constrained variables.

Model predictive control. Feedback control that repeatedly solves a constrained finite-horizon problem, applies an initial action, observes, and replans.

Multi-fidelity optimization. Joint selection of a candidate and an evaluation fidelity to improve performance at an explicitly defined target fidelity.

Multiobjective optimization. Optimization with multiple criteria whose trade-offs may be represented by dominance, scalarization, goals, or preference interaction.

Nash equilibrium. A strategy profile in which no player can improve its payoff by changing strategy unilaterally. Every finite game has an equilibrium in mixed strategies, although a pure-strategy equilibrium need not exist.

No-free-lunch result. Uniform averaging over all objective functions between finite sets makes non-revisiting black-box optimizers equivalent under the specified performance measure (Wolpert and Macready 1997). The refinement to uniformly averaged classes closed under permutations is due to Schumacher, Vose, and Whitley (2001). Useful search privileges

some structure or distribution of problems; the result does not rank practical methods on such a distribution.

Normal cone. A set of vectors supporting a feasible set at a point, used to express constrained stationarity.

Objective function. A mapping used to rank or compare feasible decisions. It is an operational representation of purpose, not purpose itself.

Online optimization. Sequential decision-making in which data or losses arrive over time and future information is unavailable.

Optimal control. Optimization of trajectories and controls subject to dynamical equations and boundary or path constraints.

Optimal transport. Optimization over couplings that move one distribution to another under a cost.

Optimality gap. A bound on how far a feasible solution's objective may be from the global optimum.

Optimizer's curse. The systematic upward bias in the estimated value of an option selected as best from noisy estimates; stronger selection tends to select favorable estimation error as well as genuine value.

Overfitting. Exploiting accidental structure in training or evaluation data so that apparent performance fails to transfer.

Overoptimization. Search pressure so strong that it exploits proxy, model, or boundary defects and worsens true outcomes.

Pareto efficiency. A property of an allocation or decision for which no feasible change can make at least one party better off without making another worse off. It says nothing by itself about equality, rights, or legitimacy.

Pareto frontier (also Pareto front). The set of nondominated feasible outcomes in a multiobjective problem.

Penalty method. A method that adds a cost for constraint violation. Finite penalties generally do not preserve the logical force of every hard constraint.

Policy. A rule mapping observed information or states to actions; also an institutional course of action. Context should disambiguate.

Preconditioning. A transformation that improves numerical geometry or conditioning, often accelerating iterative methods.

Preferential optimization. Search using comparisons or rankings in place of, or alongside, numerical response values. A latent preference model remains conditional on the evaluator and context.

Primal problem. The original or primary formulation paired with a dual problem.

Proxy. A measurable quantity used in place of a harder-to-observe value. Proxies require validation under optimization and adaptation.

Quality-adjusted life year (QALY). A measure combining time and health-state utility. Useful for analysis but ethically incomplete as a sole allocation rule.

Queueing discipline. The rule choosing which waiting job, patient, packet, or request is served next.

Randomized algorithm. An algorithm using randomness to sample, break symmetry, or obtain probabilistic guarantees.

Recourse. Actions available after uncertainty is observed; socially, also a person's ability to challenge and correct a decision.

Regret. The cumulative difference between a sequential decision rule's loss and the loss of a specified comparator, such as the best fixed action in hindsight. A regret guarantee is meaningful only relative to its comparator and information model.

Regularization. Structure added to control complexity, instability, or undesirable solutions, commonly through a penalty or constraint.

Relaxation. A tractable problem formed by enlarging the feasible set or weakening conditions, often providing a bound.

Resilience. Capacity to sustain essential function, degrade safely, recover, and adapt after disruption.

Robust optimization. Optimization against all values in a specified uncertainty set, often emphasizing worst-case feasibility or performance.

Safe optimization. Optimization that restricts the evaluations themselves to an admissible safe set under specified assumptions. Model-based safety claims require an appropriate initial set and a valid uncertainty construction.

Safety margin. Deliberate distance from a predicted limit to protect against variation, uncertainty, degradation, and error.

Satisficing. Selecting an option that meets an aspiration level or is good enough given the cost of further search, information, and delay. It can be a rational policy under bounded resources.

Scalarization. Conversion of multiple objectives into one scalar, for example by weighted sum. It embeds preferences and may miss parts of a nonconvex frontier.

Sensitivity analysis. Study of how outputs, solutions, or recommendations change when inputs and assumptions vary.

Shadow price. A dual multiplier interpreted as the local marginal value of relaxing a constraint, under regularity and stated units.

Simulation. Computational execution of a model through scenarios or time. Fidelity should be judged by decision-relevant behavior.

Slack. Unused capacity or distance from a constraint. Slack can be waste in one model and resilience in a broader model.

Slater's condition. A constraint qualification for convex programs requiring a strictly feasible point for inequality constraints (with the usual relative-interior refinement). It is a standard sufficient condition for strong duality and KKT necessity.

Soft constraint. A preferred condition whose violation is permitted at a modeled cost or priority.

SPSA. Simultaneous perturbation stochastic approximation. A method whose basic two-sided gradient estimate uses two objective measurements while perturbing all coordinates together.

Stochastic gradient. A noisy estimate of an objective gradient, commonly computed from a sample or minibatch.

Stochastic optimization. Optimization involving random variables, samples, scenarios, or noisy observations.

Subgradient. A generalized slope for a convex nonsmooth function. The set of all subgradients is the subdifferential.

Submodularity. A diminishing-returns property for set functions that supports strong discrete algorithms and approximations.

Sunk cost. A cost already incurred that should not affect a forward-looking decision except through resulting constraints or information.

Surrogate model. A cheaper approximation to an expensive experiment, simulator, or objective.

Sustainability. Maintenance of ecological and social conditions across time, including distribution, resilience, and intergenerational consequence.

System boundary. The chosen extent of entities, processes, time, and effects included in a model.

Throughput. Completed work per unit time. Increasing throughput can worsen delay or quality if queues and rework are ignored.

Trade-off. A relation in which improving one valued outcome worsens another within the current feasible set.

Trust region. A neighborhood in which a local model is considered reliable; steps are accepted based on predicted versus realized improvement.

Uncertainty set. A collection of plausible parameter values used in robust optimization.

Utility. A representation of preference or value used for comparison. Utility is meaningful only relative to its axioms, scale, and decision context.

Validation. Evaluation of whether a model or system is adequate for its intended real-world use. It differs from verification, which checks conformance to specification.

Value of an option. Benefit of retaining the ability to choose later after more information arrives.

Verification. Evidence that an implementation, calculation, or artifact satisfies its specified model or requirements.

Worst-case analysis. A guarantee over the most adverse admitted instance or uncertainty realization. Its relevance depends on what the set admits.

Zeroth-order optimization. Optimization using function-value information, including methods that estimate derivative information from perturbed evaluations.

Subject index

Page references identify substantive or representative occurrences; “passim” marks concepts used throughout the field guide.

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W

References

Aarseth, Espen J. 1997. *Cybertext: Perspectives on Ergodic Literature*. Johns Hopkins University Press. https://www.press.jhu.edu/books/title/2048/cybertext. ISBN 9780801855795.

Aho, Alfred V., Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. 2006. *Compilers: Principles, Techniques, and Tools*. 2nd ed. Addison-Wesley. ISBN 9780321486813.

Alexander, Christopher. 1964. *Notes on the Synthesis of Form*. Harvard University Press. https://www.hup.harvard.edu/books/9780674627512. ISBN 9780674627512.

Amdahl, Gene M. 1967. "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities." In *Proceedings of the AFIPS Spring Joint Computer Conference*, 483–85. https://doi.org/10.1145/1465482.1465560.

Ament, Sebastian, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. 2023. "Unexpected Improvements to Expected Improvement for Bayesian Optimization." Advances in Neural Information Processing Systems 36. Author manuscript, arXiv:2310.20708. https://arxiv.org/abs/2310.20708.

American Library Association. 2019. "Library Bill of Rights." https://www.ala.org/advocacy/intfreedom/librarybill.

Aristotle. 2002. *Nicomachean Ethics*. Edited by Sarah Broadie. Translated by Christopher Rowe. Oxford University Press. https://global.oup.com/academic/product/nicomachean-ethics-9780198752714. ISBN 9780198752714.

Arrow, Kenneth J. 1951. *Social Choice and Individual Values*. Wiley. Original edition; published before an ISBN was assigned to this edition; later-edition identifiers are not substituted.

Audet, Charles, A. L. Custódio, and J. E. Dennis Jr. 2008. "Erratum: Mesh Adaptive Direct Search Algorithms for Constrained Optimization." SIAM Journal on Optimization 18 (4): 1501–3. https://doi.org/10.1137/060671267.

Audet, Charles, and J. E. Dennis Jr. 2006. "Mesh Adaptive Direct Search Algorithms for Constrained Optimization." SIAM Journal on Optimization 17 (1): 188–217. See the separate 2008 erratum by Audet, Custódio, and Dennis. https://doi.org/10.1137/040603371.

Åström, Karl J., and Richard M. Murray. 2008. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press. https://fbsbook.org/. ISBN 9780691135762.

Banister, Eric W., T. W. Calvert, M. V. Savage, and T. M. Bach. 1975. "A Systems Model of Training for Athletic Performance." *Australian Journal of Sports Medicine* 7: 57–61.

Barocas, Solon, Moritz Hardt, and Arvind Narayanan. 2023. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press. https://fairmlbook.org/. ISBN 9780262048613.

Barroso, Luiz André, Urs Hölzle, and Parthasarathy Ranganathan. 2018. *The Datacenter as a Computer*. 3rd ed. Morgan & Claypool. https://doi.org/10.2200/S00874ED3V01Y201809CAC046. ISBN 9781681734330.

Bellman, Richard. 1957. *Dynamic Programming*. Princeton University Press. https://press.princeton.edu/books/paperback/9780691146683/dynamic-programming. ISBN 9780691146683. Original edition cited; identifier and link are for the 2010 Princeton Landmarks paperback edition.

Ben-Tal, Aharon, Laurent El Ghaoui, and Arkadi Nemirovski. 2009. *Robust Optimization*. Princeton University Press. https://doi.org/10.1515/9781400831050. ISBN 9780691143682.

Bendsøe, Martin P., and Ole Sigmund. 2004. *Topology Optimization: Theory, Methods, and Applications*. 2nd ed., corrected printing. Springer. https://doi.org/10.1007/978-3-662-05086-6. ISBN 9783540429920.

Bergstra, James S., Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. “Algorithms for Hyper-Parameter Optimization.” Advances in Neural Information Processing Systems 24. https://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.

Bergstra, James, and Yoshua Bengio. 2012. “Random Search for Hyper-Parameter Optimization.” Journal of Machine Learning Research 13 (10): 281–305. https://jmlr.org/papers/v13/bergstra12a.html.

Bertsimas, Dimitris, and Melvyn Sim. 2004. “The Price of Robustness.” *Operations Research* 52 (1): 35–53. https://doi.org/10.1287/opre.1030.0065.

Biegler, Lorenz T. 2010. *Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes*. SIAM. https://doi.org/10.1137/1.9780898719383. ISBN 9780898717020.

Boden, Margaret A. 2004. *The Creative Mind: Myths and Mechanisms*. 2nd ed. Routledge. https://www.routledge.com/The-Creative-Mind-Myths-and-Mechanisms/Boden/p/book/9780415314534. ISBN 9780415314534.

Bottou, Léon, Frank E. Curtis, and Jorge Nocedal. 2018. “Optimization Methods for Large-Scale Machine Learning.” *SIAM Review* 60 (2): 223–311. https://doi.org/10.1137/16M1080173.

Boyd, Stephen, and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press. https://doi.org/10.1017/CBO9780511804441. ISBN 9780521833783.

Breck, Eric, Shanqing Cai, Eric Nielsen, Michael Salib, and D. Sculley. 2017. “The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction.” In *Proceedings of IEEE Big Data*, 1123–32. https://doi.org/10.1109/BigData.2017.8258038.

Brooks, Frederick P., Jr. 1995. *The Mythical Man-Month: Essays on Software Engineering*. Anniversary. Addison-Wesley. ISBN 9780201835953.

Brown, Sue A., Boris P. Kovatchev, Dan Raghinaru, John W. Lum, Bruce A. Buckingham, et al. 2019. “Six-Month Randomized, Multicenter Trial of Closed-Loop Control in Type 1 Diabetes.” *New England Journal of Medicine* 381: 1707–17. https://doi.org/10.1056/NEJMoa1907863.

Bryant, Benjamin P., and Robert J. Lempert. 2010. “Thinking Inside the Box: A Participatory, Computer-Assisted Approach to Scenario Discovery.” Technological Forecasting and Social Change 77 (1): 34–49. https://doi.org/10.1016/j.techfore.2009.08.002.

Bryson, Arthur E., Jr., and Yu-Chi Ho. 1975. *Applied Optimal Control: Optimization, Estimation, and Control*. Taylor & Francis. ISBN 9780891162285.

Bubeck, Sébastien. 2015. “Convex Optimization: Algorithms and Complexity.” *Foundations and Trends in Machine Learning* 8 (3–4): 231–357. https://doi.org/10.1561/2200000050.

Campbell, Donald T. 1976. *Assessing the Impact of Planned Social Change*. Dartmouth College Public Affairs Center. https://doi.org/10.56645/jmde.v7i15.297. DOI and link identify the authorized 2011 reprint of the original 1976 working paper.

Canguilhem, Georges. 1991. *The Normal and the Pathological*. Zone Books. https://press.princeton.edu/books/paperback/9780942299595/the-normal-and-the-pathological. ISBN 9780942299595.

Card, Stuart K., Thomas P. Moran, and Allen Newell. 1983. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates. ISBN 9780898592436.

Cerezo, Marco, Andrew Arrasmith, Ryan Babbush, et al. 2021. “Variational Quantum Algorithms.” *Nature Reviews Physics* 3: 625–44. https://doi.org/10.1038/s42254-021-00348-9.

Chouldechova, Alexandra. 2017. “Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments.” *Big Data* 5 (2): 153–63. https://doi.org/10.1089/big.2016.0047.

Colton, Simon. 2008. “Creativity Versus the Perception of Creativity in Computational Systems.” In Creative Intelligent Systems: Papers from the AAAI Spring Symposium, 14–20. Technical Report SS-08-03.

https://research.monash.edu/en/publications/creativity-versus-the-perception-of-creativity-in-computational-s/.

Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2022. *Introduction to Algorithms*. 4th ed. MIT Press. https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/. ISBN 9780262046305.

Cunningham, Ward. 1992. "The WyCash Portfolio Management System." In Addendum to the Proceedings on Object-Oriented Programming Systems, Languages, and Applications, 29–30. ACM. https://doi.org/10.1145/157710.157715.

Cuturi, Marco. 2013. "Sinkhorn Distances: Lightspeed Computation of Optimal Transport." In *Advances in Neural Information Processing Systems 26*, 2292–2300. https://papers.nips.cc/paper/4927-sinkhorn-distances-lightspeed-computation-of-optimal-transport.

Dantzig, George B. 1963. *Linear Programming and Extensions*. Princeton University Press. https://press.princeton.edu/books/paperback/9780691059136/linear-programming-and-extensions. ISBN 9780691059136. Original edition cited; identifier and link are for the 1998 paperback edition.

Darwin, Charles. 1859. *On the Origin of Species*. John Murray. https://darwin-online.org.uk/converted/published/1859_Origin_F373.html.

Daulton, Samuel, Maximilian Balandat, and Eytan Bakshy. 2021. "Parallel Bayesian Optimization of Multiple Noisy Objectives with Expected Hypervolume Improvement." Advances in Neural Information Processing Systems 34. Author manuscript, arXiv:2105.08195. https://arxiv.org/abs/2105.08195.

Dean, Jeffrey, and Luiz André Barroso. 2013. "The Tail at Scale." *Communications of the ACM* 56 (2): 74–80. https://doi.org/10.1145/2408776.2408794.

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II." *IEEE Transactions on Evolutionary Computation* 6 (2): 182–97. https://doi.org/10.1109/4235.996017.

Deb, Kalyanmoy. 2001. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley. https://www.wiley.com/en-us/Multi+Objective+Optimization+using+Evolutionary+Algorithms-p-9780471873396. ISBN 9780471873396.

Del Castillo, Enrique, and Douglas C. Montgomery. 1995. "A Kalman Filtering Process Control Scheme with an Application in Semiconductor Short Run Manufacturing." *Quality and Reliability Engineering International* 11 (2): 101–5. https://doi.org/10.1002/qre.4680110205.

Dempe, Stephan. 2002. *Foundations of Bilevel Programming*. Kluwer Academic Publishers. https://doi.org/10.1007/b101970. ISBN 9781402006319.

Dennard, Robert H., Fritz H. Gaensslen, Hwa-Nien Yu, V. Leo Rideout, Ernest Bassous, and Andre R. LeBlanc. 1974. "Design of Ion-Implanted MOSFET's with Very Small Physical Dimensions." *IEEE Journal of Solid-State Circuits* 9 (5): 256–68. https://doi.org/10.1109/JSSC.1974.1050511.

Dorigo, Marco, and Thomas Stützle. 2004. *Ant Colony Optimization*. MIT Press. https://mitpress.mit.edu/9780262042192/ant-colony-optimization/. ISBN 9780262042192.

Duchi, John, Elad Hazan, and Yoram Singer. 2011. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization." *Journal of Machine Learning Research* 12: 2121–59. https://jmlr.org/papers/v12/duchi11a.html.

Dwork, Cynthia, and Aaron Roth. 2014. *The Algorithmic Foundations of Differential Privacy*. Now Publishers. https://doi.org/10.1561/0400000042. ISBN 9781601988188.

Eco, Umberto. 1990. *The Limits of Interpretation*. Indiana University Press. ISBN 9780253318527.

Edgar, Thomas F., David M. Himmelblau, and Leon S. Lasdon. 2001. *Optimization of Chemical Processes*. 2nd ed. McGraw-Hill. ISBN 9780070393592.

Elenes, Alejandro G. N., Eric Williams, Eric Hittinger, and Naga Srujana Goteti. 2022. “How Well Do Emission Factors Approximate Emission Changes from Electricity System Models?” *Environmental Science & Technology* 56 (20): 14701–12. https://doi.org/10.1021/acs.est.2c02344.

Elgammal, Ahmed, Bingchen Liu, Mohamed Elhoseiny, and Marian Mazzone. 2017. “CAN: Creative Adversarial Networks, Generating Art by Learning about Styles and Deviating from Style Norms.” In *International Conference on Computational Creativity*. https://arxiv.org/abs/1706.07068.

Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter. 2019. “Neural Architecture Search: A Survey.” *Journal of Machine Learning Research* 20 (55): 1–21. https://jmlr.org/papers/v20/18-598.html.

Endsley, Mica R. 1995. “Toward a Theory of Situation Awareness in Dynamic Systems.” *Human Factors* 37 (1): 32–64. https://doi.org/10.1518/001872095779049543.

Endsley, Mica R., and Esin O. Kiris. 1995. “The Out-of-the-Loop Performance Problem and Level of Control in Automation.” *Human Factors* 37 (2): 381–94. https://doi.org/10.1518/001872095779064555.

Eriksson, David, and Martin Jankowiak. 2021. “High-Dimensional Bayesian Optimization with Sparse Axis-Aligned Subspaces.” Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, PMLR 161: 493–503. https://proceedings.mlr.press/v161/eriksson21a.html.

Eriksson, David, Michael Pearce, Jacob R. Gardner, Ryan Turner, and Matthias Poloczek. 2019. “Scalable Global Optimization via Local Bayesian Optimization.” Advances in Neural Information Processing Systems 32: 5497–5508. Author manuscript, arXiv:1910.01739. https://arxiv.org/abs/1910.01739.

Espeland, Wendy Nelson, and Michael Sauder. 2007. “Rankings and Reactivity: How Public Measures Recreate Social Worlds.” *American Journal of Sociology* 113 (1): 1–40. https://doi.org/10.1086/517897.

European Parliament and Council of the European Union. 2024. *Regulation (EU) 2024/1689 Laying down Harmonised Rules on Artificial Intelligence*. https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng.

Falkner, Stefan, Aaron Klein, and Frank Hutter. 2018. “BOHB: Robust and Efficient Hyperparameter Optimization at Scale.” Proceedings of the 35th International Conference on Machine Learning, PMLR 80: 1437–46. https://proceedings.mlr.press/v80/falkner18a.html.

Farhi, Edward, Jeffrey Goldstone, and Sam Gutmann. 2014. “A Quantum Approximate Optimization Algorithm.” https://arxiv.org/abs/1411.4028. arXiv preprint arXiv:1411.4028.

Fisher, R. A. 1930. *The Genetical Theory of Natural Selection*. Clarendon Press. https://archive.org/details/geneticaltheoryo031631mbp.

Fong, Brendan, and David I. Spivak. 2019. *An Invitation to Applied Category Theory: Seven Sketches in Compositionality*. Cambridge University Press. https://doi.org/10.1017/9781108668804. ISBN 9781108482295.

Fong, Brendan. 2015. “Decorated Cospans.” *Theory and Applications of Categories* 30 (33): 1096–1120. https://www.tac.mta.ca/tac/volumes/30/33/30-33abs.html.

Forsgren, Nicole, Jez Humble, and Gene Kim. 2018. *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press. https://itrevolution.com/product/accelerate/. ISBN 9781942788331.

Frid, Emma. 2019. “Accessible Digital Musical Instruments—a Review of Musical Interfaces in Inclusive Music Practice.” *Multimodal Technologies and Interaction* 3 (3): 57. https://doi.org/10.3390/mti3030057.

Frigo, Matteo, Charles E. Leiserson, Harald Prokop, and Sridhar Ramachandran. 1999. “Cache-Oblivious Algorithms.” In Proceedings of the 40th Annual Symposium on Foundations of Computer Science, 285–97. IEEE. https://doi.org/10.1109/SFFCS.1999.814600.

Gale, David, and Lloyd S. Shapley. 1962. “College Admissions and the Stability of Marriage.” *American Mathematical Monthly* 69 (1): 9–15. https://doi.org/10.1080/00029890.1962.11989827.

Gardner, Jacob R., Matt J. Kusner, Zhixiang (Eddie) Xu, Kilian Q. Weinberger, and John P. Cunningham. 2014. “Bayesian Optimization with Inequality Constraints.” Proceedings of the 31st International Conference

on Machine Learning, PMLR 32 (2): 937–45. Author names follow the paper PDF. https://proceedings.mlr.press/v32/gardner14.html.

Garey, Michael R., and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman. ISBN 9780716710455.

Genette, Gérard. 1980. *Narrative Discourse: An Essay in Method*. Cornell University Press. https://www.cornellpress.cornell.edu/book/9780801492594/narrative-discourse/. ISBN 9780801492594.

Ghani, Neil, Jules Hedges, Viktor Winschel, and Philipp Zahn. 2018. "Compositional Game Theory." In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, 472–81. https://doi.org/10.1145/3209108.3209165.

Gigerenzer, Gerd, and Reinhard Selten, eds. 2002. *Bounded Rationality: The Adaptive Toolbox*. MIT Press. https://mitpress.mit.edu/9780262571647/bounded-rationality/. ISBN 9780262571647.

Goemans, Michel X., and David P. Williamson. 1995. "Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming." *Journal of the ACM* 42 (6): 1115–45. https://doi.org/10.1145/227683.227684.

González, Javier, Zhenwen Dai, Andreas Damianou, and Neil D. Lawrence. 2017. "Preferential Bayesian Optimization." Proceedings of the 34th International Conference on Machine Learning, PMLR 70: 1282–91. https://proceedings.mlr.press/v70/gonzalez17a.html.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. https://www.deeplearningbook.org/. ISBN 9780262035613.

Goodhart, Charles A. E. 1975. "Problems of Monetary Management: The u.k. Experience." In *Papers in Monetary Economics*. Reserve Bank of Australia. https://www.rba.gov.au/publications/confs/1975/goodhart.pdf.

Gould, Stephen Jay, and Richard C. Lewontin. 1979. "The Spandrels of San Marco and the Panglossian Paradigm: A Critique of the Adaptationist Programme." *Proceedings of the Royal Society of London B* 205 (1161): 581–98. https://doi.org/10.1098/rspb.1979.0086.

Gustafson, John L. 1988. "Reevaluating Amdahl's Law." *Communications of the ACM* 31 (5): 532–33. https://doi.org/10.1145/42411.42415.

Haasnoot, Marjolijn, Jan H. Kwakkel, Warren E. Walker, and Judith ter Maat. 2013. "Dynamic Adaptive Policy Pathways: A Method for Crafting Robust Decisions for a Deeply Uncertain World." *Global Environmental Change* 23 (2): 485–98. https://doi.org/10.1016/j.gloenvcha.2012.12.006.

Hansen, Nikolaus. 2016. "The CMA Evolution Strategy: A Tutorial." arXiv preprint arXiv:1604.00772. Revised version 2, 2023. https://arxiv.org/abs/1604.00772.

Hardt, Moritz, Eric Price, and Nati Srebro. 2016. "Equality of Opportunity in Supervised Learning." In *Advances in Neural Information Processing Systems 29*, 3315–23. https://arxiv.org/abs/1610.02413.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. 2nd ed. Springer. https://doi.org/10.1007/978-0-387-84858-7. ISBN 9780387848570.

Hennessy, John L., and David A. Patterson. 2019. "A New Golden Age for Computer Architecture." *Communications of the ACM* 62 (2): 48–60. https://doi.org/10.1145/3282307.

Hino, Miyuki, Christopher B. Field, and Katharine J. Mach. 2017. "Managed Retreat as a Response to Natural Hazard Risk." *Nature Climate Change* 7: 364–70. https://doi.org/10.1038/nclimate3252.

Hoffmann, Jordan, Sebastian Borgeaud, Arthur Mensch, et al. 2022. "Training Compute-Optimal Large Language Models." arXiv:2203.15556. https://arxiv.org/abs/2203.15556.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press. Original edition; published before an ISBN was assigned to this edition; later-edition identifiers are not substituted.

Holtzman, Ari, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. "The Curious Case of Neural Text Degeneration." In *International Conference on Learning Representations*. https://iclr.cc/virtual_2020/poster_rygGQyrFvH.html.

Horowitz, Mark. 2014. "1.1 Computing's Energy Problem (and What We Can Do about It)." *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 10–14. https://doi.org/10.1109/ISSCC.2014.6757323.

Humble, Jez, and David Farley. 2010. *Continuous Delivery*. Addison-Wesley. ISBN 9780321601919.

Hunink, M. G. Myriam, Milton C. Weinstein, Eve Wittenberg, et al. 2014. *Decision Making in Health and Medicine: Integrating Evidence and Values*. 2nd ed. Cambridge University Press. https://doi.org/10.1017/CBO9781139506779. ISBN 9781107690479.

Hutter, Frank, Lars Kotthoff, and Joaquin Vanschoren, eds. 2019. *Automated Machine Learning: Methods, Systems, Challenges*. Springer. https://doi.org/10.1007/978-3-030-05318-5. ISBN 9783030053178.

Intergovernmental Panel on Climate Change. 2021. *Climate Change 2021: The Physical Science Basis*. Contribution of Working Group I to the Sixth Assessment Report. Edited by Valérie Masson-Delmotte and colleagues. Cambridge University Press. Chapters 9 and 11 address ocean, cryosphere, sea level, and weather and climate extremes. https://doi.org/10.1017/9781009157896.

Intergovernmental Panel on Climate Change. 2022a. "Cities and Settlements by the Sea." Cambridge University Press. https://www.ipcc.ch/report/ar6/wg2/chapter/ccp2/.

Intergovernmental Panel on Climate Change. 2022b. "Climate Change 2022: Mitigation of Climate Change." Cambridge University Press. https://doi.org/10.1017/9781009157926.

International Federation of Library Associations and Institutions, and UNESCO. 2022. "The IFLA–UNESCO Public Library Manifesto 2022." International Federation of Library Associations and Institutions. https://www.ifla.org/public-library-manifesto/.

International Organization for Standardization. 2023. *ISO/IEC 25010:2023 Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Product Quality Model*. https://www.iso.org/standard/78176.html.

Jones, Donald R., Matthias Schonlau, and William J. Welch. 1998. "Efficient Global Optimization of Expensive Black-Box Functions." *Journal of Global Optimization* 13: 455–92. https://doi.org/10.1023/A:1008306431147.

Jouppi, Norman P., Cliff Young, Nishant Patil, et al. 2017. "In-Datacenter Performance Analysis of a Tensor Processing Unit." In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 1–12. https://doi.org/10.1145/3079856.3080246.

Kahneman, Daniel, and Amos Tversky. 1979. "Prospect Theory: An Analysis of Decision Under Risk." *Econometrica* 47 (2): 263–91. https://doi.org/10.2307/1914185.

Kairouz, Peter, H. Brendan McMahan, Brendan Avent, et al. 2021. "Advances and Open Problems in Federated Learning." *Foundations and Trends in Machine Learning* 14 (1–2): 1–210. https://doi.org/10.1561/2200000083.

Kalman, Rudolf E. 1960. "A New Approach to Linear Filtering and Prediction Problems." *Journal of Basic Engineering* 82 (1): 35–45. https://doi.org/10.1115/1.3662552.

Kandasamy, Kirthevasan, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. 2017. "Multi-Fidelity Bayesian Optimisation with Continuous Approximations." Proceedings of the 34th International Conference on Machine Learning, PMLR 70: 1799–1808. https://proceedings.mlr.press/v70/kandasamy17a.html.

Kant, Immanuel. 1785. *Groundwork of the Metaphysics of Morals*. Johann Friedrich Hartknoch.

Kantorovich, L. V. 1960. “Mathematical Methods of Organizing and Planning Production.” *Management Science* 6 (4): 366–422. https://doi.org/10.1287/mnsc.6.4.366. Originally published in Russian in 1939.

Kaplan, Jared, Sam McCandlish, Tom Henighan, et al. 2020. “Scaling Laws for Neural Language Models.” arXiv:2001.08361. https://arxiv.org/abs/2001.08361.

Karmarkar, Narendra. 1984. “A New Polynomial-Time Algorithm for Linear Programming.” *Combinatorica* 4 (4): 373–95. https://doi.org/10.1007/BF02579150.

Kennedy, James, and Russell Eberhart. 1995. “Particle Swarm Optimization.” In *Proceedings of ICNN’95 - International Conference on Neural Networks*, 4:1942–48. https://doi.org/10.1109/ICNN.1995.488968.

Khachiyan, Leonid G. 1979. “A Polynomial Algorithm in Linear Programming.” Doklady Akademii Nauk SSSR 244 (5): 1093–96. In Russian. https://www.mathnet.ru/eng/dan42319.

Kingma, Diederik P., and Jimmy Ba. 2015. “Adam: A Method for Stochastic Optimization.” In *International Conference on Learning Representations*. https://arxiv.org/abs/1412.6980.

Kirkpatrick, Scott, C. Daniel Gelatt, and Mario P. Vecchi. 1983. “Optimization by Simulated Annealing.” *Science* 220 (4598): 671–80. https://doi.org/10.1126/science.220.4598.671.

Kleinberg, Jon, and Éva Tardos. 2005. *Algorithm Design*. Pearson. ISBN 9780321295354.

Knuth, Donald E. 1997. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. 3rd ed. Addison-Wesley. https://www-cs-faculty.stanford.edu/~knuth/taocp.html. ISBN 9780201896831.

Koopmans, Tjalling C., ed. 1951. *Activity Analysis of Production and Allocation*. John Wiley & Sons. https://cowles.yale.edu/sites/default/files/2022-09/m13-all.pdf.

Kuhn, Harold W., and Albert W. Tucker. 1951. “Nonlinear Programming.” In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, 481–92. University of California Press. https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-Second-Berkeley-Symposium-on-Mathematical-Statistics-and/chapter/Nonlinear-Programming/bsmsp/1200500249.

Kwakkel, Jan H., Marjolijn Haasnoot, and Warren E. Walker. 2015. “Developing Dynamic Adaptive Policy Pathways: A Computer-Assisted Approach for Developing Adaptive Strategies for a Deeply Uncertain World.” *Climatic Change* 132: 373–86. https://doi.org/10.1007/s10584-014-1210-4.

Larson, Jeffrey, Matt Menickelly, and Stefan M. Wild. 2019. “Derivative-Free Optimization Methods.” Acta Numerica 28: 287–404. https://doi.org/10.1017/S0962492919000060.

Lawvere, F. William. 1973. “Metric Spaces, Generalized Logic, and Closed Categories.” *Rendiconti Del Seminario Matematico e Fisico Di Milano* 43: 135–66. https://www.tac.mta.ca/tac/reprints/articles/1/tr1abs.html.

Lempert, Robert J., Steven W. Popper, and Steven C. Bankes. 2003. Shaping the Next One Hundred Years: New Methods for Quantitative, Long-Term Policy Analysis. RAND, MR-1626-RPC. https://doi.org/10.7249/MR1626.

Lerdahl, Fred, and Ray Jackendoff. 1983. *A Generative Theory of Tonal Music*. MIT Press. https://mitpress.mit.edu/9780262621076/a-generative-theory-of-tonal-music/. ISBN 9780262621076.

Levin, Simon A. 1999. *Fragile Dominion: Complexity and the Commons*. Perseus Books. ISBN 9780738201115.

Li, Lisha, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. “Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization.” Journal of Machine Learning Research 18 (185): 1–52. Year follows the journal’s article record. https://jmlr.org/papers/v18/16-558.html.

Lindauer, Marius, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. 2022. “SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization.” Journal of Machine Learning Research 23 (54): 1–9. https://jmlr.org/papers/v23/21-0888.html.

Little, John D. C. 1961. “A Proof for the Queuing Formula: L = λW.” *Operations Research* 9 (3): 383–87. https://doi.org/10.1287/opre.9.3.383.

Mac Lane, Saunders. 1998. *Categories for the Working Mathematician*. 2nd ed. Springer. ISBN 9780387984032.

Markowitz, Harry. 1952. “Portfolio Selection.” *Journal of Finance* 7 (1): 77–91. https://doi.org/10.1111/j.1540-6261.1952.tb01525.x.

Martins, Joaquim R. R. A., and Andrew B. Lambe. 2013. “Multidisciplinary Design Optimization: A Survey of Architectures.” *AIAA Journal* 51 (9): 2049–75. https://doi.org/10.2514/1.J051895.

Masanet, Eric, Arman Shehabi, Nuoa Lei, Sarah Smith, and Jonathan Koomey. 2020. “Recalibrating Global Data Center Energy-Use Estimates.” *Science* 367 (6481): 984–86. https://doi.org/10.1126/science.aba3758.

May, Gary S., and Costas J. Spanos. 2006. *Fundamentals of Semiconductor Manufacturing and Process Control*. Wiley-IEEE Press. https://doi.org/10.1002/0471790281. ISBN 9780471784067.

Maynard Smith, John. 1982. *Evolution and the Theory of Games*. Cambridge University Press. https://doi.org/10.1017/CBO9780511806292. ISBN 9780521288842.

McClean, Jarrod R., Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. 2018. “Barren Plateaus in Quantum Neural Network Training Landscapes.” *Nature Communications* 9: 4812. https://doi.org/10.1038/s41467-018-07090-4.

McNeil, Barbara J., Stephen G. Pauker, Harold C. Sox, and Amos Tversky. 1982. “On the Elicitation of Preferences for Alternative Therapies.” *New England Journal of Medicine* 306 (21): 1259–62. https://doi.org/10.1056/NEJM198205273062103.

Michaud, Richard O. 1989. “The Markowitz Optimization Enigma: Is ‘Optimized’ Optimal?” Financial Analysts Journal 45 (1): 31–42. https://doi.org/10.2469/faj.v45.n1.31.

Miettinen, Kaisa. 1998. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers. https://doi.org/10.1007/978-1-4615-5563-6. ISBN 9780792382782.

Mill, John Stuart. 1863. *Utilitarianism*. Parker, Son, and Bourn. https://www.gutenberg.org/ebooks/11224.

Mirhoseini, Azalia, Anna Goldie, Mustafa Yazgan, et al. 2021. “A Graph Placement Methodology for Fast Chip Design.” *Nature* 594: 207–12. https://doi.org/10.1038/s41586-021-03544-w.

Moretti, Franco. 2005. *Graphs, Maps, Trees: Abstract Models for Literary History*. Verso. ISBN 9781844670260.

Muchnick, Steven S. 1997. *Advanced Compiler Design and Implementation*. Morgan Kaufmann. ISBN 9781558603202.

Muller, Jerry Z. 2018. *The Tyranny of Metrics*. Princeton University Press. https://press.princeton.edu/books/hardcover/9780691174952/the-tyranny-of-metrics. ISBN 9780691174952.

Myerson, Roger B. 1981. “Optimal Auction Design.” *Mathematics of Operations Research* 6 (1): 58–73. https://doi.org/10.1287/moor.6.1.58.

Myerson, Roger B., and Mark A. Satterthwaite. 1983. “Efficient Mechanisms for Bilateral Trading.” Journal of Economic Theory 29 (2): 265–81. https://doi.org/10.1016/0022-0531(83)90048-0.

Nash, John F. 1950. “Equilibrium Points in n-Person Games.” *Proceedings of the National Academy of Sciences* 36 (1): 48–49. https://doi.org/10.1073/pnas.36.1.48.

National Institute of Standards and Technology. 2013. “Response Surface Designs.” https://www.itl.nist.gov/div898/handbook/pri/section3/pri336.htm.

Nelder, J. A., and R. Mead. 1965. “A Simplex Method for Function Minimization.” The Computer Journal 7 (4): 308–13. https://doi.org/10.1093/comjnl/7.4.308.

Nemhauser, G. L., L. A. Wolsey, and M. L. Fisher. 1978. “An Analysis of Approximations for Maximizing Submodular Set Functions—I.” *Mathematical Programming* 14: 265–294. https://doi.org/10.1007/BF01588971.

Nemhauser, George L., and Laurence A. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley. https://www.wiley.com/en-us/Integer+and+Combinatorial+Optimization-p-9780471359432. ISBN 9780471359432.

Neumann, Peter J., Gillian D. Sanders, Louise B. Russell, Joanna E. Siegel, and Theodore G. Ganiats, eds. 2016. *Cost-Effectiveness in Health and Medicine*. 2nd ed. Oxford University Press. https://global.oup.com/academic/product/cost-effectiveness-in-health-and-medicine-9780190492939. ISBN 9780190492939.

Nocedal, Jorge, and Stephen J. Wright. 2006. *Numerical Optimization*. 2nd ed. Springer. https://doi.org/10.1007/978-0-387-40065-5. ISBN 9780387303031.

Norman, Donald A. 2013. *The Design of Everyday Things*. Revised and expanded. Basic Books. https://mitpressbookstore.mit.edu/book/9780465050659. ISBN 9780465050659.

Nussbaum, Martha C. 2011. *Creating Capabilities: The Human Development Approach*. Harvard University Press. https://www.hup.harvard.edu/books/9780674072350. ISBN 9780674072350.

Oppenheim, Alan V., and Ronald W. Schafer. 2010. *Discrete-Time Signal Processing*. 3rd ed. Pearson. ISBN 9780131988422.

Orth, Jeffrey D., Ines Thiele, and Bernhard Ø. Palsson. 2010. “What Is Flux Balance Analysis?” *Nature Biotechnology* 28: 245–48. https://doi.org/10.1038/nbt.1614.

Ostrom, Elinor. 1990. *Governing the Commons*. Cambridge University Press. https://doi.org/10.1017/CBO9780511807763. ISBN 9780521405997.

Papadimitriou, Christos H., and Kenneth Steiglitz. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall. ISBN 9780131524620.

Parasuraman, Raja, and Victor Riley. 1997. “Humans and Automation: Use, Misuse, Disuse, Abuse.” *Human Factors* 39 (2): 230–53. https://doi.org/10.1518/001872097778543886.

Pareto, Vilfredo. 1896. *Cours d’économie Politique*. F. Rouge. https://archive.org/details/coursdconomiep01pare.

Parikh, Neal, and Stephen Boyd. 2014. “Proximal Algorithms.” *Foundations and Trends in Optimization* 1 (3): 127–239. https://doi.org/10.1561/2400000003.

Parker, Geoffrey A., and John Maynard Smith. 1990. “Optimality Theory in Evolutionary Biology.” *Nature* 348: 27–33. https://doi.org/10.1038/348027a0.

Pauker, Stephen G., and Jerome P. Kassirer. 1980. “The Threshold Approach to Clinical Decision Making.” *New England Journal of Medicine* 302 (20): 1109–17. https://doi.org/10.1056/NEJM198005153022003.

Payne, John W., James R. Bettman, and Eric J. Johnson. 1993. *The Adaptive Decision Maker*. Cambridge University Press. https://doi.org/10.1017/CBO9781139173933. ISBN 9780521425261.

Peyré, Gabriel, and Marco Cuturi. 2019. “Computational Optimal Transport with Applications to Data Sciences.” *Foundations and Trends in Machine Learning* 11 (5–6): 355–607. https://doi.org/10.1561/2200000073.

Pontryagin, Lev S., Vladimir G. Boltyanskii, Revaz V. Gamkrelidze, and Evgenii F. Mishchenko. 1962. *The Mathematical Theory of Optimal Processes*. Interscience.

Powell, M. J. D. 2009. “The BOBYQA Algorithm for Bound Constrained Optimization without Derivatives.” Technical report DAMTP 2009/NA06, University of Cambridge. https://www.damtp.cam.ac.uk/user/na/NA_papers/NA2009_06.pdf.

Preskill, John. 2018. “Quantum Computing in the NISQ Era and Beyond.” *Quantum* 2: 79. https://doi.org/10.22331/q-2018-08-06-79.

Propp, Vladimir. 1968. *Morphology of the Folktale*. 2nd ed. University of Texas Press. https://utpress.utexas.edu/9780292783768/morphology-of-the-folktale/. ISBN 9780292783768.

Puterman, Martin L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley. https://doi.org/10.1002/9780470316887. ISBN 9780471727828.

Raissi, Maziar, Paris Perdikaris, and George Em Karniadakis. 2019. “Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations.” *Journal of Computational Physics* 378: 686–707. https://doi.org/10.1016/j.jcp.2018.10.045.

Rajkomar, Alvin, Jeffrey Dean, and Isaac Kohane. 2019. “Machine Learning in Medicine.” *New England Journal of Medicine* 380 (14): 1347–58. https://doi.org/10.1056/NEJMra1814259.

Rasmussen, Jens. 1983. “Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models.” *IEEE Transactions on Systems, Man, and Cybernetics* SMC-13 (3): 257–66. https://doi.org/10.1109/TSMC.1983.6313160.

Rasmussen, Jens. 1997. “Risk Management in a Dynamic Society: A Modelling Problem.” *Safety Science* 27 (2–3): 183–213. https://doi.org/10.1016/S0925-7535(97)00052-0.

Rawlings, James B., David Q. Mayne, and Moritz M. Diehl. 2017. *Model Predictive Control: Theory, Computation, and Design*. 2nd ed. Nob Hill Publishing. https://sites.engineering.ucsb.edu/~jbraw/mpc/.

Rawls, John. 1971. *A Theory of Justice*. Harvard University Press. https://www.hup.harvard.edu/books/9780674000780. ISBN 9780674000780. Original edition cited; identifier and link are for the 1999 revised edition.

Raworth, Kate. 2017. *Doughnut Economics*. Random House. https://www.kateraworth.com/doughnut/. ISBN 9781847941398.

Ristic, Bora, Kaveh Madani, and Zen Makuch. 2015. “The Water Footprint of Data Centers.” *Sustainability* 7 (8): 11260–84. https://doi.org/10.3390/su70811260.

Robbins, Herbert, and Sutton Monro. 1951. “A Stochastic Approximation Method.” *The Annals of Mathematical Statistics* 22 (3): 400–407. https://doi.org/10.1214/aoms/1177729586.

Rockafellar, R. Tyrrell, and Stanislav Uryasev. 2000. “Optimization of Conditional Value-at-Risk.” *Journal of Risk* 2 (3): 21–41. https://doi.org/10.21314/JOR.2000.038.

Rockafellar, R. Tyrrell. 1970. *Convex Analysis*. Princeton University Press. https://press.princeton.edu/books/paperback/9780691015866/convex-analysis. ISBN 9780691015866. Original edition cited; identifier and link are for the 1997 first paperback edition.

Rockström, Johan, Will Steffen, Kevin Noone, et al. 2009. “A Safe Operating Space for Humanity.” *Nature* 461: 472–75. https://doi.org/10.1038/461472a.

Sarter, Nadine B., and David D. Woods. 1995. “How in the World Did We Ever Get into That Mode? Mode Error and Awareness in Supervisory Control.” *Human Factors* 37 (1): 5–19. https://doi.org/10.1518/001872095779049516.

Schön, Donald A. 1983. *The Reflective Practitioner*. Basic Books. https://www.hachettebookgroup.com/titles/donald-a-schon/the-reflective-practitioner/9780465068784/. ISBN 9780465068784.

Schrijver, Alexander. 2003. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer. https://link.springer.com/book/9783540443896. ISBN 9783540443896.

Schumacher, C., Michael D. Vose, and L. Darrell Whitley. 2001. “The No Free Lunch and Problem Description Length.” In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001), 565–70. Morgan Kaufmann. https://dl.acm.org/doi/10.5555/2955239.2955325.

Sculley, D., Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, and Dan Dennison. 2015. "Hidden Technical Debt in Machine Learning Systems." In *Advances in Neural Information Processing Systems 28*, 2503–11. https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.

Selbst, Andrew D., danah boyd, Sorelle A. Friedler, Suresh Venkatasubramanian, and Janet Vertesi. 2019. "Fairness and Abstraction in Sociotechnical Systems." In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 59–68. https://doi.org/10.1145/3287560.3287598.

SEMI. 2022. *SEMI E10: Specification for Definition and Measurement of Equipment Reliability, Availability, and Maintainability (RAM) and Utilization*. SEMI. https://www.semi.org/en/products-services/standards/step/equipment-performance-metrics.

Sen, Amartya. 1970. *Collective Choice and Social Welfare*. Holden-Day. ISBN 0816277656.

Sen, Amartya. 1985. *Commodities and Capabilities*. North-Holland. https://sen.scholars.harvard.edu/publications/commodities-and-capabilities. ISBN 0444877304.

Shahriari, Bobak, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. "Taking the Human Out of the Loop: A Review of Bayesian Optimization." *Proceedings of the IEEE* 104 (1): 148–75. https://doi.org/10.1109/JPROC.2015.2494218.

Shannon, Claude E. 1948. "A Mathematical Theory of Communication." *Bell System Technical Journal* 27 (3–4): 379–423, 623–56. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x.

Shapiro, Alexander, Darinka Dentcheva, and Andrzej Ruszczyński. 2021. *Lectures on Stochastic Programming: Modeling and Theory*. 3rd ed. SIAM. https://doi.org/10.1137/1.9781611976595. ISBN 9781611976588.

Sheffi, Yosef. 1985. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall. https://sheffi.mit.edu/book/urban-transportation-networks. ISBN 9780139397295.

Sigmund, Ole, and Kurt Maute. 2013. "Topology Optimization Approaches: A Comparative Review." *Structural and Multidisciplinary Optimization* 48 (6): 1031–55. https://doi.org/10.1007/s00158-013-0978-6.

Silver, David, Aja Huang, Chris J. Maddison, et al. 2016. "Mastering the Game of Go with Deep Neural Networks and Tree Search." *Nature* 529: 484–89. https://doi.org/10.1038/nature16961.

Simon, Herbert A. 1955. "A Behavioral Model of Rational Choice." *Quarterly Journal of Economics* 69 (1): 99–118. https://doi.org/10.2307/1884852.

Simon, Herbert A. 1956. "Rational Choice and the Structure of the Environment." *Psychological Review* 63 (2): 129–38. https://doi.org/10.1037/h0042769.

Simon, Herbert A. 1996. *The Sciences of the Artificial*. 3rd ed. MIT Press. https://mitpress.mit.edu/9780262691918/the-sciences-of-the-artificial/. ISBN 9780262691918.

Sims, Karl. 1991. "Artificial Evolution for Computer Graphics." In *Proceedings of SIGGRAPH '91*, 319–28. https://doi.org/10.1145/122718.122752.

Smith, James E., and Robert L. Winkler. 2006. "The Optimizer's Curse: Skepticism and Postdecision Surprise in Decision Analysis." *Management Science* 52 (3): 311–22. https://doi.org/10.1287/mnsc.1050.0451.

Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams. 2012. "Practical Bayesian Optimization of Machine Learning Algorithms." Advances in Neural Information Processing Systems 25. https://papers.nips.cc/paper/4522-practical-bayesian-optimization-of-machine-learning-algorithms.

Souppaya, Murugiah, Karen Scarfone, and Donna Dodson. 2022. "Secure Software Development Framework (SSDF) Version 1.1." NIST SP 800-218. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-218.

Spall, James C. 1992. “Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation.” IEEE Transactions on Automatic Control 37 (3): 332–41. https://doi.org/10.1109/9.119632.

Spivak, David I. 2014. *Category Theory for the Sciences*. MIT Press. https://mitpress.mit.edu/9780262028134/category-theory-for-the-sciences/. ISBN 9780262028134.

Stahlberg, Felix, and Bill Byrne. 2019. “On NMT Search Errors and Model Errors: Cat Got Your Tongue?” In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 3356–62. https://doi.org/10.18653/v1/D19-1331.

Stephens, David W., and John R. Krebs. 1986. *Foraging Theory*. Princeton University Press. https://press.princeton.edu/books/paperback/9780691084428/foraging-theory. ISBN 9780691084428.

Storn, Rainer, and Kenneth Price. 1997. “Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces.” Journal of Global Optimization 11: 341–59. https://doi.org/10.1023/A:1008202821328.

Sui, Yanan, Alkis Gotovos, Joel Burdick, and Andreas Krause. 2015. “Safe Exploration for Optimization with Gaussian Processes.” Proceedings of the 32nd International Conference on Machine Learning, PMLR 37: 997–1005. https://proceedings.mlr.press/v37/sui15.html.

Sutton, Richard S., and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. 2nd ed. MIT Press. https://mitpress.mit.edu/9780262039246/reinforcement-learning/. ISBN 9780262039246.

Swersky, Kevin, Jasper Snoek, and Ryan P. Adams. 2013. “Multi-Task Bayesian Optimization.” Advances in Neural Information Processing Systems 26. https://papers.nips.cc/paper/5086-multi-task-bayesian-optimization.

Tabassi, Elham. 2023. “Artificial Intelligence Risk Management Framework (AI RMF 1.0).” NIST AI 100-1. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1.

Topol, Eric J. 2019. “High-Performance Medicine: The Convergence of Human and Artificial Intelligence.” *Nature Medicine* 25: 44–56. https://doi.org/10.1038/s41591-018-0300-7.

Tymoczko, Dmitri. 2006. “The Geometry of Musical Chords.” *Science* 313 (5783): 72–74. https://doi.org/10.1126/science.1126287.

U.S. Food and Drug Administration, Health Canada, and Medicines and Healthcare products Regulatory Agency. 2021. “Good Machine Learning Practice for Medical Device Development: Guiding Principles.” https://www.fda.gov/medical-devices/software-medical-device-samd/good-machine-learning-practice-medical-device-development-guiding-principles.

Vickers, Andrew J., and Elena B. Elkin. 2006. “Decision Curve Analysis: A Novel Method for Evaluating Prediction Models.” *Medical Decision Making* 26 (6): 565–74. https://doi.org/10.1177/0272989X06295361.

Villani, Cédric. 2003. *Topics in Optimal Transportation*. American Mathematical Society. https://doi.org/10.1090/gsm/058. ISBN 9780821833124.

von Neumann, John, and Oskar Morgenstern. 1944. *Theory of Games and Economic Behavior*. Princeton University Press. https://press.princeton.edu/books/paperback/9780691130613/theory-of-games-and-economic-behavior. ISBN 9780691130613. Original edition cited; identifier and link are for the 2007 60th Anniversary Commemorative Edition.

Wardrop, John Glen. 1952. “Some Theoretical Aspects of Road Traffic Research.” *Proceedings of the Institution of Civil Engineers* 1 (3): 325–62. https://doi.org/10.1680/ipeds.1952.11259.

Weinstein, Milton C., Joanna E. Siegel, Marthe R. Gold, Mark S. Kamlet, and Louise B. Russell. 1996. “Recommendations of the Panel on Cost-Effectiveness in Health and Medicine.” *JAMA* 276 (15): 1253–58. https://doi.org/10.1001/jama.276.15.1253.

Wessel, David, and Matthew Wright. 2001. “Problems and Prospects for Intimate Musical Control of Computers.” In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 11–14. https://doi.org/10.5281/zenodo.1176382.

Wiggins, Geraint A. 2006. “A Preliminary Framework for Description, Analysis and Comparison of Creative Systems.” *Knowledge-Based Systems* 19 (7): 449–58. https://doi.org/10.1016/j.knosys.2006.04.009.

Winner, Langdon. 1980. “Do Artifacts Have Politics?” *Daedalus* 109 (1): 121–36. https://www.jstor.org/stable/20024652.

Wolpert, David H., and William G. Macready. 1997. “No Free Lunch Theorems for Optimization.” *IEEE Transactions on Evolutionary Computation* 1 (1): 67–82. https://doi.org/10.1109/4235.585893.

Woodbury, Robert. 2010. *Elements of Parametric Design*. Routledge. https://www.routledge.com/Elements-of-Parametric-Design/Woodbury/p/book/9780415779876. ISBN 9780415779876.

World Health Organization Regional Office for Europe. 2026. “Heat–Health Action Plans: Guidance.” 2nd ed. World Health Organization Regional Office for Europe. https://www.who.int/kazakhstan/publications/i/item/9789289062930. ISBN 9789289062930.

World Health Organization. 2021. “Ethics and Governance of Artificial Intelligence for Health.” World Health Organization. https://www.who.int/publications/i/item/9789240029200. ISBN 9789240029200.

Xenakis, Iannis. 1992. *Formalized Music: Thought and Mathematics in Composition*. Revised. Pendragon Press. ISBN 9781576470794.

Zhang, Xinyu, Daolang Huang, Samuel Kaski, and Julien Martinelli. 2025. “PABBO: Preferential Amortized Black-Box Optimization.” Proceedings of the International Conference on Learning Representations (ICLR 2025). https://proceedings.iclr.cc/paper_files/paper/2025/hash/881259965dacb9f42967aae84a157283-Abstract-Conference.html.

Optimization in Software Diagnostics, Observability, Memory Dump Analysis, Trace and Log Analysis, and Debugging

An Evidence-Governed Framework for Better Diagnostic Decisions

SOFTWARE DIAGNOSTICS | STANDALONE ARTICLE

Optimization in Software Diagnostics, Observability, Memory Dump Analysis, Trace and Log Analysis, and Debugging

An Evidence-Governed Framework for Better Diagnostic Decisions

Concept and editorial direction: Dmitry Vostokov, DumpAnalysis.org
Earlier drafting and revision assistance: GPT-6 Astra Max
Independent editorial review of earlier versions: Fable 5.1 Extra

Standalone article | Version 14 | September 2026

Abstract

When analyzing a software problem, we select artifacts, debugger commands, trace filters, hypotheses, and possible changes. Faster commands and smaller traces may help, depending on the diagnostic question. Acquiring additional evidence requires time, computation, storage, and analysis effort, and may expose private data or affect the execution being analyzed. Here we consider diagnostic optimization as a sequential decision process that takes into account analysis costs and the consequences of incorrect decisions.

We consider value of information, multiobjective optimization, pattern-oriented analysis, and controlled intervention in software diagnostics, observability, memory dump analysis, trace and log analysis, and debugging. The presentation follows the analysis pattern sections Problem, Context, Forces, Solution, Resulting Context, Known Uses, and Related Patterns. We distinguish between optimizing software behavior and optimizing its analysis, considering complementary artifacts, sampling and acquisition limits, mechanisms, and reduced failure reproductions. A software latency example illustrates how selected evidence can guide a reversible change. We call this process evidence-governed: each recommendation includes its supporting evidence, assumptions, uncertainty, preservation requirements, authorization, stopping rule, and conditions for revision.

Keywords: software diagnostics; observability; memory dump analysis; trace analysis; log analysis; debugging; optimization; value of information; analysis patterns; software narratology; root cause analysis

Attribution note: Dmitry Vostokov retains editorial responsibility. Versions 12 and 13 underwent independent editorial review dated September 12, 2026. Version 14 incorporates the resulting corrections.

1. Problem

Suppose we have a service that became slow after a software release. There are various artifacts available for analysis, such as metrics, traces, logs, profiles, memory dumps, configuration histories, and deployment records. Although each artifact may contain useful diagnostic information, analyzing all of them may take more time than we have. Similar situations occur when a process crashes, memory consumption increases, or a distributed transaction disappears. Before requesting another artifact, we need to consider what it may help us find out and what its collection will cost.

The problem is to select diagnostic actions and their order so that the expected benefit of the resulting decision justifies the cost and risk of collecting the necessary evidence. During diagnostic analysis, we may need to answer the following questions:

- Which observation corresponds to the failure experienced by users, and which one only measures some internal software activity?
- Which artifact is sufficient to answer the current question, and what missing evidence could change our conclusion?
- Do we need another query, a targeted trace, a memory dump, a controlled reproduction, a comparison with a working system, or an immediate rollback?
- Which hypothesis should we investigate first, given the limited analysis time and production resources?
- When does additional investigation have less value than its cost?
- What properties must a debugging change preserve?
- How can we check that the change solved the original problem without introducing another one?

An improvement in one part of the diagnostic process may introduce a problem in another part. For example, a faster analysis may finish before an alternative mechanism is considered, or additional telemetry may make a Significant Event harder to find. A software change may improve performance while reducing reliability, security, or

maintainability (distinct product-quality characteristics in ISO/IEC 25010), with debuggability addressed indirectly through analysability (International Organization for Standardization 2023). Similarly, restarting a process may reduce recovery time but erase the state needed to explain the failure. We therefore consider each such improvement in relation to the original diagnostic problem.

The result of diagnostic analysis may be a set of hypotheses arranged by plausibility, a conclusion with explicitly stated limits, a safe workaround, or a request for an artifact that helps distinguish between possible mechanisms. In some cases, we may find that a suspected mechanism is impossible or that the available evidence is insufficient to justify a change. Finding a single root cause is one possible result, provided that we have the corresponding evidence.

2. Context

This pattern is applicable to software construction and post-construction problems where we have incomplete evidence of software execution. The analysis may involve a live system, postmortem artifacts, one process, or several interacting services. Typical problems include functional failures, performance degradation, security issues, resource retention, concurrency defects, and configuration problems.

We distinguish the following activities in the diagnostic process:

- **Software diagnostics** studies signs of software structure, behavior, and abnormality in software execution artifacts and related evidence.
- **Observability** concerns the design and operation of mappings from hidden software state to outputs that can be inspected.
- **Memory dump analysis** reconstructs a dynamic system from one or more memory snapshots, which may be incomplete or acquired while the system is changing.
- **Trace and log analysis** reconstructs software behavior from ordered, partially ordered, or weakly ordered message sequences, which we also consider as software narratives.
- **Debugging** changes software execution, code, configuration, or environment to test an explanation and remove or contain a defect.

These activities form a loop where observability produces artifacts, diagnostic analysis uses them to construct explanations, and debugging uses such explanations to select

changes. The changes affect the system and its subsequent artifacts. For example, a restart erases state, a retry policy changes traffic, and an additional log statement may affect contention. When comparing artifacts collected before and after a change, we need to take these effects into account. Execution message sequences can also be considered as software narratives; the application of narratological concepts to software behavior is called software narratology.

We consider this loop in the context of Systematic Software Diagnostics, where software artifacts are treated as traces, logs, texts, and narratives to which reusable analysis patterns can be applied (Vostokov 2024). A memory dump gives us a projection of execution state, whereas a trace gives us a projection of an execution sequence. Debugger output is itself a log, and collections of stack traces can also be analyzed as structured traces. Which projection is useful depends on the diagnostic question.

Optimization can help us select such projections and the operations to be applied to them. First, we need to identify the possible decisions, objectives, constraints, uncertainty, available evidence, people affected by the decision, and conditions for revising it. This is done before selecting a solver or a debugger command. Otherwise, we may execute the whole command sequence correctly and still answer a different question.

3. Forces

Diagnostic optimization involves several objectives that may conflict with each other.

3.1 Information versus overhead

Detailed telemetry may show additional relationships, but collecting and analyzing it requires CPU time, memory, bandwidth, storage, and analyst attention. Changes in scheduling due to instrumentation may also hide or introduce concurrency behavior. We therefore need instrumentation that helps distinguish between the relevant hypotheses without exceeding the acceptable overhead.

3.2 Speed versus evidential strength

Fast triage may help protect a service while leaving the underlying mechanism unclear. Further analysis can strengthen the explanation, although the resulting delay may prolong the problem. The choice depends on the consequences of the proposed action and

whether it can be reversed. For example, changing a low-risk feature flag may be justified by provisional evidence, whereas an irreversible data migration requires stronger evidence.

3.3 Local visibility versus end-to-end consequence

We use CPU utilization, allocation rate, cache misses, queue length, and error counts to explain software behavior. However, users are concerned with completed transactions, tail latency, correctness, availability, and recovery. Long-tail behavior may dominate the performance experienced by users even when the averages are acceptable (Dean and Barroso 2013). An optimization that improves an internal counter may simply move work to another layer, or improve a microbenchmark while making the complete service path slower.

3.4 Snapshot detail versus temporal explanation

A memory dump may show object state, stack traces, locks, heaps, modules, and memory corruption at a particular time, whereas a trace may show event order, duration, correlation, repetition, and missing activity over an interval. Neither projection is preferable in every case. We select one according to whether we need to investigate state, execution history, a mechanism, or an interaction.

3.5 Standardization versus case sensitivity

Checklists and analysis patterns help us avoid omissions and compare diagnostic reports, but their applicability depends on the problem context. Repeating every step of a familiar checklist may not help with the current problem, and an unfamiliar mechanism may require a different sequence of analysis steps. We can still use the same pattern repertoire while changing the selection and order of patterns.

3.6 Automation versus analyst judgment

Enumeration, filtering, correlation, clustering, and repeated queries can be automated. However, we still need to assess artifact quality, reconsider the analysis objective, and recognize when the current representation does not contain enough information. For this reason, automated results should include their selection rules and uncertainty so that we can revise the analysis when necessary.

3.7 Recovery versus evidence preservation

A restart, failover, scale-out, or rollback may restore a service quickly and, at the same time, destroy the state needed for diagnosis. Keeping the system in a failing state for too long may also cause additional harm. Artifact acquisition procedures should therefore be authorized in advance, fit within recovery deadlines, and specify when service restoration takes priority.

3.8 Precision versus robustness

A detailed model may explain one incident but fail to explain the same software under another workload. A simpler model may be easier to test across different builds and environments. In either case, we state whether the conclusion applies to one process, release, or workload, or to a more general mechanism.

3.9 Search depth versus stopping

There is almost always another query to run or another artifact to request. We stop when the evidence supports an acceptable decision, when further analysis is unlikely to justify its cost, or when the available representation cannot distinguish between the relevant mechanisms. Such stopping is an example of satisficing under bounded rationality (Simon 1955), and does not mean that we have found a global optimum.

4. A General Model of Diagnostic Optimization

4.1 Diagnostics as a sequential policy

Let H be a set of possible explanations of the system, with $h \in H$ denoting the true explanation. In this probabilistic model, the explanations are mutually exclusive and together cover the possibilities under consideration. One explanation may include several interacting faults, and we include a residual alternative for mechanisms outside the current catalog. At time t, we have evidence E_t and can select an action a from the admissible set A_t. A policy π uses the available evidence and context to select actions, eventually producing a diagnostic or corrective decision d. Such an action may involve acquiring an artifact, running a query, comparing a baseline, reproducing a workload, or modifying the system. The resulting observation changes the evidence available for the next decision.

We therefore optimize the policy that selects and orders actions:

$$\pi^* = \arg\min_{\pi} \mathbb{E}[C_{\text{acq}}(\pi) + C_{\text{analysis}}(\pi) + C_{\text{delay}}(\pi) + C_{\text{perturb}}(\pi) + C_{\text{intervention}}(\pi) + L(d_\pi, h)], \quad \text{subject to } g_k(\pi) \le 0. \qquad (1)$$

The expectation in Equation (1) takes into account uncertainty about the true explanation h and future observations under policy π. The cost terms correspond to acquisition, analysis, delay, perturbation, and intervention, while the loss $L(d_\pi, h)$ evaluates the resulting decision against the true explanation. All these terms are expressed on a common loss scale. We also avoid counting the same harm twice (for example, as both delay cost and decision loss). Each constraint g_k specifies a condition that an admissible policy must satisfy, with k indexing the constraints. Such conditions may concern service limits, access rights, privacy, safety, or evidence preservation. This formulation uses the standard notions of feasibility and constrained optimization (Boyd and Vandenberghe 2004; Nocedal and Wright 2006) and is consistent with the broader treatment in Vostokov (2026).

If a command runs faster, one cost component is reduced. However, to find out whether this improves the diagnostic process, we also need to consider the total loss and check that the required constraints are still satisfied.

4.2 Value of information

The value of an observation depends on which decision it may change. Here we use the decision-theoretic value of information discussed by Howard (1966), whereas Shannon entropy measures uncertainty without taking decision consequences into account (Shannon 1948). Let y denote a possible observation produced by action a, and let d range over the admissible terminal decisions. We express the expected value of information (EVI) as the reduction in minimum expected decision loss after considering the possible observations:

$$\text{EVI}(a \mid E_t) = \min_d \mathbb{E}_{h|E_t}[L(d,h)] - \mathbb{E}_{y|a,E_t}\left[\min_d \mathbb{E}_{h|E_t,y,a}[L(d,h)]\right]. \qquad (2)$$

Equation (2) applies when acquiring information does not change the decision problem itself. We assume a coherent probability model, an integrable loss, and the possibility of ignoring an observation. Under these assumptions, the gross expected value is nonnegative, although the value after subtracting costs may be negative. If an action also changes the system (for example, a restart), we need to consider the resulting state and the consequences of the intervention using Equation (1). Evidence collected after such a

change is conditioned on that intervention, so the observation-only value in Equation (2) does not represent its total benefit.

For admissible observation actions, we can use a one-step selection rule that compares information value with the cost of collection and analysis:

$$a_t^* = \arg\max_{a \in A_t} \{\mathrm{EVI}(a \mid E_t) - \lambda C(a)\}. \qquad (3)$$

Here $C(a)$ combines time, computation, storage, overhead, privacy exposure, and analysis effort into a scalar estimate on a declared common cost scale. The positive multiplier λ converts this scale into decision-loss units. Actions that violate hard constraints have already been excluded from the admissible set A_t. Equation (3) is applicable only when such a scalar comparison adequately represents the remaining consequences; asymmetric or catastrophic risks may require the vector approach described in Section 4.4. If no candidate has positive net value, or there are no admissible candidates, no further observation is selected. This rule is a one-step approximation to the policy problem in Equation (1).

In practice, we can assess EVI qualitatively by asking which possible result would change the next intervention. If no plausible result from an artifact would affect that decision, its immediate decision value is low. For example, a small configuration file may be more useful than another large execution log.

In some cases, artifacts complement each other. A trace may identify an object involved in a delayed request, while a memory dump contains the state reachable from that object. Each artifact on its own may leave the next action unchanged, but together they may help us identify the mechanism. The joint information value need not be the sum of the individual values (Howard 1966). Before deciding that no further evidence is useful, we therefore consider whether collecting a few artifacts together, or one after another depending on earlier results, may help.

Consider a simple example where the lowest expected decision loss is 20 units before a query and 8 units after averaging over its possible results. In this case, EVI is 12 units. If the query costs 3 units on the same scale, we have a net value of 9 units. These numbers represent assumed decision losses, not measured incident probabilities. Different plausible losses or observation probabilities may change the ranking, so we also need to check how sensitive the estimates are to such changes.

4.3 Stopping rules

We continue while an admissible observation, or a short sequence of evidence collection actions, has positive expected net value and can be completed before the deadline. We stop or switch to containment in the following cases:

- **Sufficient evidence:** the current evidence supports an action with acceptable residual risk.
- **Insufficient value of further analysis:** none of the available diagnostic actions is expected to change the decision enough to justify its cost.
- **Need for immediate containment:** ongoing harm makes containment or rollback more valuable than collecting further evidence.
- **Insufficient representation:** the available observations cannot distinguish between relevant mechanisms, so we specify additional instrumentation or explicitly limit the conclusion.

Stopping the analysis does not imply certainty. It means that, under the current assumptions, further search is not required or does not justify its cost.

4.4 Multiobjective outputs and admissibility

Before assigning weights to the consequences of a diagnostic action, it is often useful to consider them separately. We can represent them by the following vector:

$$\mathbf{z}(a) = \left(\begin{array}{c} \text{diagnostic delay},\ \text{acquisition cost},\ \text{perturbation}, \\ \text{privacy exposure},\ \text{decision risk},\ \text{evidential deficit} \end{array} \right). \qquad (4)$$

All six components in Equation (4) are to be minimized. By evidential deficit, we mean the shortfall from the quality of evidence required for a decision. After excluding inadmissible actions, we can compare the remaining ones using thresholds, lexicographic priorities, Pareto comparison, or a weighted model with explicit weights. For example, one incident policy may give priority to data integrity, followed by service restoration, evidence preservation, and cost. For another incident class, evidence preservation may come before restoration. The selected priorities and their reasons are recorded with the decision.

A constraint should not be replaced by a very large weight without justification. For example, a privacy boundary, a requirement not to restart, or a limit on tracing overhead specifies which actions are allowed and who may authorize them. Treating such a condition as a preference changes the problem.

4.5 A compositional view

We can also consider transformations of evidence in terms of category theory. Software execution state, a memory dump, debugger output, a normalized table, and a diagnostic report can be viewed as objects, with acquisition, decoding, filtering, symbolization, and interpretation as morphisms between them. Here we use a conceptual model. Defining a formal category would additionally require the objects, morphisms, identities, and composition law to be specified.

For each transformation, we specify which information is preserved. For example, a filter may keep every message with a particular correlation identifier while removing information about other requests. A stack summary may preserve the top frames but omit recursion depth, and a quotient trace may preserve message classes while removing individual values. If a later transformation needs the information removed by an earlier one, the resulting analysis may be incomplete.

Even if every transformation succeeds, the resulting diagnosis may be incorrect. We also need to check that the inputs, assumptions, and preservation claims are compatible. Incomparable clocks, incorrect symbols, inconsistent acquisition, biased sampling, or changed message semantics may invalidate the interpretation despite successful completion of every tool.

5. Solution: Evidence-Governed Diagnostic Optimization

The following process is iterative, with each stage producing artifacts that can be inspected again. Figure 1 shows the main cycle, including a return from diagnostic analysis to artifact acquisition. Depending on the result, other stages may also require us to revisit an earlier decision.

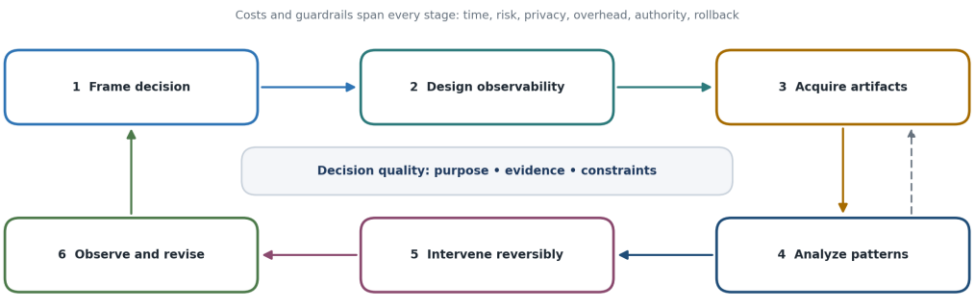


Figure 1. Diagnostic optimization as an iterative control loop. Observation can reopen framing, and analysis can require further artifact acquisition. Constraints apply throughout the process.

5.1 Frame the diagnostic decision

We begin with the decision to be made and describe the observed consequences, available time, affected components and people, and cost of delay. Symptoms and observations are recorded separately from their interpretation and proposed causes. We can then select tools according to the evidence needed for the decision.

The initial diagnostic record should include:

- the incident or question and the decision deadline;
- the current baseline and the population used for comparison;
- the hypotheses that would require substantially different actions;
- safety, privacy, access, performance, and recovery constraints;
- who may authorize evidence collection, intervention, stopping, and rollback;
- acceptable analysis results, including inconclusive results and conclusions with stated limits;
- conditions for monitoring and revising the decision after the action.

Such a record helps us notice when a convenient metric has replaced the original purpose of the investigation. For example, lower CPU utilization does not necessarily mean that a checkout service has recovered. We may need to restore the required latency while preserving order correctness and keeping the failure rate within acceptable limits.

5.2 Optimize observability for discriminating power

When designing observability, we consider which differences in software behavior may need to be recovered during an incident. For example, we may need to distinguish a

request waiting for a connection from a request executing a database query. If both are logged simply as a slow database operation, we lose a distinction that may determine the next intervention.

We can consider the available observability resources in terms of four dimensions:

- **Coverage:** which components, transitions, states, and failure paths are represented?
- **Resolution:** what temporal, spatial, and semantic detail is preserved?
- **Correlation:** can evidence be joined across requests, threads, processes, hosts, and versions?
- **Trust:** are clocks, schemas, symbol files, sampling rules, provenance, and integrity known?

Before adding large amounts of descriptive logging, we record stable identifiers, boundary crossings, significant state changes, causal parents, resource ownership, and failure results. Structured fields with explicit units and schemas are preferable. We also need enough information to distinguish between absent activity and absent instrumentation.

We can start with observations that have low collection overhead, add more detail when a trigger fires, and return to the original level after a specified interval. The trigger and return conditions also need to be observable; otherwise, detailed recording may start after the event of interest or continue after the incident. Dapper is an example of limiting instrumentation overhead through instrumentation in common libraries and sampling (Sigelman et al. 2010).

Different sampling policies preserve different parts of the observed population. For example, uniform sampling may miss rare failures, whereas tail sampling can retain traces according to latency or errors. The latter requires buffering and sufficient processing capacity (OpenTelemetry Authors n.d.), and cannot recover spans already discarded upstream. We therefore record the sampling policy and collection losses together with the acquired evidence.

A collection containing mostly failures may help us investigate their mechanisms. However, its raw error fraction and p99 do not estimate the corresponding values for all requests. For such estimates, we retain independent population counters or a reference sample with known inclusion rules. When comparing releases, we use the same request

classes and selection rules. Late or dropped spans are recorded as missing evidence, since their absence does not establish that a request was short or successful.

5.3 Optimize artifact acquisition

We consider related artifacts together. A configuration file, deployment difference, or health record may answer a question that cannot be answered from a large memory dump. Small DA+TA illustrates such a case, where a configuration setting explains unavailable functionality despite a lengthy execution log (Vostokov 2023, 276–277). Here the optimization concerns selecting evidence for the diagnostic question, rather than simply reducing artifact size.

For each possible artifact, we consider the following questions:

- Which hypotheses can we distinguish using this artifact?
- What state or history will it omit?
- How much will acquisition perturb the system?
- Can we collect it before the relevant state disappears?
- What is needed to decode and interpret it, such as symbols, schemas, keys, or clocks?
- Can we reproduce, compare, and share the artifact under the applicable policy?

Where possible, we first collect a small artifact that helps distinguish between hypotheses and use its result to decide what to collect next. Acquisition procedures for crashes, hangs, leaks, and latency problems are prepared before an incident. Related artifacts need a coordinated collection interval and a record of their acquisition order, since a dump and a trace collected several minutes apart may describe different states or workloads.

This also corresponds to Adjoint Space and Paratext, where a trace may guide a memory lookup, or a trace may be needed to explain earlier changes to the state seen in a dump (Vostokov 2024). We record the relationship between such artifacts, including process lifetime, build, object identity, and capture interval. The same address or identifier may be reused, so its presence in two artifacts does not necessarily refer to the same object.

5.4 Optimize the analysis path

During diagnostic analysis, we move between representations and consider alternative explanations. Pattern names describe recurring analysis operations, and the recorded patterns and results allow another analyst to follow the abstractions used to reach the conclusion.

We can alternate an overview of the artifacts with tests that distinguish between hypotheses:

1. Check artifact validity, origin, version, time range, completeness, and the information needed for decoding.
2. Extract Basic Facts and construct a baseline from a working system or an earlier execution.
3. Find Significant Events, state boundaries, discontinuities, dominant activities, and missing expected evidence.
4. Consider alternative mechanisms that may explain the observed behavior.
5. Select the least costly test among those expected to provide useful evidence for distinguishing the hypotheses.
6. Revise the hypothesis ranking, record contradictions, and stop when the decision rule is satisfied.

A familiar problem may suggest a hypothesis, but familiarity alone does not determine its place in the analysis. We also consider its prior plausibility, supporting and contradictory evidence, the consequences of missing the mechanism, the cost of a test that distinguishes it from alternatives, and whether the corresponding action can be reversed. Even an unlikely mechanism may need early attention if its consequences are severe.

Diagnostic analysis can itself be considered as another trace (Vostokov 2024). For example, we may find large Time Deltas between useful results, repeated queries, abandoned analysis branches, or a pattern that was found only near the end of the analysis. Such observations may point to improvements in tools or checklists. We can reduce repeated work while still allowing the diagnostic question to change.

5.5 Optimize interventions, not only explanations

In debugging, a diagnostic conclusion is used to select an experiment or a change. Possible changes include configuration settings, feature flags, rollbacks, traffic routing, resource limits, instrumentation, code patches, data repair, and architectural redesign.

A change may restore a service and also test an explanation, but these are separate results. For example, a restart changes many state variables, so it may tell us little about which variable was responsible for the problem. A smaller code or configuration change may test the proposed mechanism more directly, although preparing such a change may take longer than the incident deadline allows.

For every transformation, we need a preservation claim specifying which properties should remain unchanged. A compiler optimization should preserve the chosen program semantics, and a refactoring should preserve behavior. For a rollback, we need compatible data and interfaces; for a cache change, acceptable data freshness; for a logging change, timing within a specified tolerance. The tests cover both the intended improvement and the corresponding preservation claim.

Where applicable, we use canaries, shadow execution, fault injection, replay, and controlled workload comparison (Humble and Farley 2010). If possible, each experiment changes one mechanism. When several changes have to be applied together, we explicitly state the resulting limits on causal attribution.

5.6 Close the loop with monitoring and revision

After the intervention, we have a changed system producing new evidence. We monitor the intended result, required limits, effects across the observed population, workarounds, and recurrence. We compare the predicted consequences with the observed ones, in addition to comparing measurements before and after the change. This view of the complete process complements delivery-performance measures relating lead time, reliability, and organizational outcomes (Forsgren, Humble, and Kim 2018).

We preserve the following minimum set of evidence and associated records:

- the decision and rejected baselines;
- artifact, data, code, configuration, symbol, tool, and model versions;
- objective and constraint definitions, with the people responsible for them;

- hypothesis set, decisive evidence, contradictions, and uncertainty;
- sensitivity, failure, and rollback tests;
- authorization for the intervention and a record of its implementation;
- monitoring thresholds, the people responsible for responding, and a review date.

These artifacts and records allow another analyst to reconstruct the decision and check whether its assumptions still hold after changes to software, workload, infrastructure, organization, or purpose.

6. Applications to Diagnostic Domains

The same formulation can be applied to all five domains. Table 1 summarizes their different decisions, objectives, and preservation requirements.

Table 1. Diagnostic optimization across the five domains.

Domain	Primary decisions	Optimization focus	Constraints and preservation
Software diagnostics	Artifact requests; hypothesis order; analysis patterns; expert escalation	Decision utility; time to discrimination; reproducibility; bounded uncertainty	Evidence validity; deadline; access; service consequence; authority
Observability	Event placement; schemas; correlation; sampling; retention; alerts	Discriminating power; detection and localization delay; operator workload	Runtime overhead; privacy; storage; telemetry integrity and failure
Memory dump analysis	Dump scope; trigger; capture timing; snapshot sequence; debugger queries	State coverage; consistency; acquisition speed; analysis value	Pause time; size; privacy; symbols; capture semantics; missing pages
Trace and log analysis	Statements; fields; temporal resolution; sampling; filtering; aggregation	Temporal and causal coverage; comparability; signal-to-noise; search cost	Perturbation; clocks; schemas; provenance; cost; false absence
Debugging	Experiment; workaround; patch; rollout; rollback; architectural change	Restoration; mechanism discrimination; low residual risk; supportability	Preservation claim; reversibility; safety; compatibility; decision rights

6.1 Software diagnostics

In software diagnostics, we select artifacts, analysis patterns, the order of hypotheses to investigate, the experts to involve, and the level of detail in the report. These choices depend on the next admissible decision and what remains unknown. We also take into account incident deadlines, access, evidence quality, the effect on the service, and organizational responsibility.

Optimization improves software diagnostics when it reduces the total cost of search without prematurely excluding possible hypotheses. An analysis pattern describes a

recurring diagnostic operation in a particular context. We can use a pattern catalog to select possible operations and record their application as a pattern sequence. Such a sequence may need to be revised when the evidence points to a different mechanism or a new artifact allows us to distinguish cases that previously appeared similar.

We also need to distinguish between a problem pattern and its mechanism. For example, Dynamic Memory Corruption may result from an overwrite, an invalid parameter, or a double release, so recognizing the pattern does not tell us which of these caused it. This distinction is explicit in the later Artifacts–Patterns–Mechanisms (A.P.M.) methodology (Vostokov 2024, "Patterns-Based Root Cause Analysis Methodology"). Here optimization concerns the selection of the next artifact or test that can distinguish between such mechanisms. Knowing only how often the pattern occurs is insufficient.

6.2 Observability

For observability, we can vary event placement, metric definitions, trace context, log schemas, sampling, retention, aggregation, alert thresholds, and escalation rules. The selection depends on which hidden software structure we need to recover and the cost of recovering it.

We can express the objective as a vector comprising the ability to distinguish hypotheses, detection delay, localization time, causal coverage, overhead, storage, privacy exposure, operator workload, and resilience to telemetry failure. Each component has its own direction of improvement (for example, increasing causal coverage and reducing detection delay). We also need to test the telemetry pipeline under overload, when the evidence it produces may be most important.

An alert should result in an action whose expected benefit justifies interrupting the recipient. We therefore consider precision, recall, delay, severity, whether the alert can be acted upon, and the additional work it creates. Suppressing repeated alerts may reduce noise, but the suppression rule must preserve rare cases that require a different response.

6.3 Memory dump analysis

When analyzing a memory dump, we model a dynamic computer system from a memory slice (Vostokov 2024). The selected dump type determines which questions we can answer. Full, kernel, process, heap, thread, and selective captures contain different parts

of the state and have different acquisition costs. We need the parts that provide evidence for the hypotheses under consideration.

If memory regions are copied over an interval while the system is changing, the resulting snapshot may be temporally inconsistent. We therefore record the capture mechanism, duration, freeze semantics, missing pages, truncation, compression, and available symbols with the evidence. Several snapshots can be considered as a short trace, which may show monotonic growth, repeated blocking, ownership transfer, or a state transition.

Consider, for example, the Windows minidump format. MiniDumpWithFullMemory includes accessible process memory, whereas MiniDumpWithFullMemoryInfo includes information about memory regions. The latter does not by itself include their contents (Microsoft n.d.). We may therefore see a region in the memory map but be unable to inspect its bytes. Before interpreting an unsuccessful lookup as evidence that an object was absent, we check the capture options and the streams present in the dump.

We start by checking dump integrity, platform and build identity, symbols, the list of processes and threads, and the exception or stop reason. The subsequent analysis steps depend on the suspected crash, hang, leak, corruption, deadlock, or performance mechanism. Reducing the number of debugger commands is useful only if we still obtain the decisive observations. Macro-commands and Elementary Analysis Patterns can reduce routine work while preserving such observations for further inspection.

In many cases, comparing snapshots is more useful than examining one snapshot in greater detail. We may compare working and non-working systems, early and late snapshots, snapshots before and after a change, or repeated captures under a controlled workload. Such comparisons help distinguish persistent structure from a temporary coincidence, provided that the relevant conditions remain the same or their differences are recorded.

Execution Residue may contain evidence of earlier activity in stack or heap memory (Vostokov 2024). However, finding an address or a value does not automatically tell us where it belongs in the execution history. Memory reuse, overwritten bytes, and missing acquisition data limit what can be reconstructed. Symbols may help interpret captured addresses, but cannot restore bytes that were not captured. Depending on these limitations, another snapshot, a trace, or a controlled reproduction may be more useful than further analysis of the same dump.

6.4 Trace and log analysis

For traces and logs, optimization applies both to the construction of message sequences and to their subsequent analysis. A smaller trace may be easier to navigate, although the reduction may also remove a relevant interaction. We therefore consider semantic coverage, time resolution, correlation, and analysis cost together.

The following established analysis patterns are applicable here:

- **Focus of Tracing** changes the semantic region we examine, which is different from simply selecting a time interval (Vostokov 2023, 141).
- **Message Essence** provides a useful form of a message for searching, often by removing variable data and keeping the invariants and constants needed to distinguish messages (195).
- **Minimal Trace** is the baseline obtained when software runs with its default configuration and no interaction or input data (206).
- **Measurement** covers message-based quantities such as time delta, density, current, field values, and distance (183–184). The related **Time Delta** pattern concerns intervals between Significant Events (298).
- **Relative Density** compares semantically related message types, with the total trace size canceled from the ratio (253).
- **Silent Messages** allows us to analyze the absence of messages at a selected time resolution (272–273).
- **Small DA+TA** directs us to a small artifact that may explain the problem more readily than a large execution narrative (276–277).

For a trace containing N recorded messages over a duration Δt, let N_m and N_n count messages of types m and n. Statement Density, Statement Current, and Relative Density can be expressed as follows (Vostokov 2023, 253, 286):

$$\rho_m = \frac{N_m}{N}, \qquad I = \frac{N}{\Delta t}, \qquad r_{m,n} = \frac{N_m}{N_n}. \qquad (5)$$

Here ρ_m measures the prominence of messages of type m, I is the current of all recorded messages, and $r_{m,n}$ compares the two selected message types. The denominators require $N > 0$, $\Delta t > 0$, and $N_n > 0$, respectively. If a denominator is zero, the corresponding measure is undefined; in this case, we report the counts and explain why it cannot be calculated. These quantities are diagnostic features. For example, a high error density

may be due to a narrow filter, whereas a low density may hide a serious event among many unrelated messages.

Event ordering also needs to be distinguished from sorting by timestamp. Lamport's happened-before relation defines a partial order from local execution and message exchange (Lamport 1978), whereas sorting wall-clock timestamps may impose an order unsupported by the evidence. Correlation identifiers associate events, but this association alone does not prove a causal mechanism. When calculating Time Deltas across hosts, we take clock uncertainty into account or use compatible measurements of duration and causal links.

We can express Trace Porosity using the Silent Messages construction (Vostokov 2023, 272–273):

$$\phi_{\text{porosity}} = \frac{N_s}{N_s + N_r}, \qquad (6)$$

where N_s is the number of Silent Messages inserted at the selected time resolution and N_r is the number of recorded messages. The denominator must be positive. We count recorded messages individually, even when several have the same timestamp. Thus, the value is the fraction of Silent Messages in the expanded trace. It is also the fraction of empty time bins only in the special case where each occupied bin contains exactly one recorded message. For comparisons, we specify the resolution, activity scope, and rule for identifying gaps. Here we do not add Silent Messages before the first or after the last recorded message.

Suppose we have eight recorded messages and two inserted Silent Messages. The resulting porosity is $2/(2 + 8) = 0.2$. If we add more recorded messages at already occupied timestamps, this message-count ratio changes even though the empty time intervals remain the same. We therefore compare traces with corresponding instrumentation and activity scopes. The porosity value alone does not tell us whether the silence is due to normal waiting, missing instrumentation, or lost telemetry.

6.5 Debugging

In debugging, we select experiments and changes that test a mechanism or restore the intended functionality. A small patch is useful when it also preserves the required invariants and allows continued support of the system. The size of the patch alone does not tell us whether the change is appropriate.

We can begin a debugger query sequence with a few inexpensive checks and extend it when the result helps distinguish between hypotheses. Intermediate results can be reused while their assumptions remain valid. For live queries, we also need to consider how breakpoints, watchpoints, and tracing affect execution. Such effects are recorded together with the observations used to support the conclusion.

Finding a root cause and repairing the software are different results. One mechanism may have several possible repairs, and one repair may hide several mechanisms. A workaround may be appropriate while containing an incident but unsuitable as a permanent solution. We therefore record how long it is intended to be used and what further work is needed to correct the problem.

Suppose a long input or sequence of operations reproduces a failure. Delta debugging removes parts of that input or sequence and tests whether the same failure can still be reproduced (Zeller and Hildebrandt 2002). The test distinguishes between reproduction of the target failure, its absence, and an unresolved result (for example, invalid input). Under its assumptions, ddmin produces a 1-minimal example, where deleting any one remaining element no longer reproduces the failure. This is a local guarantee; the example is not necessarily the smallest possible one. For intermittent failures, we need repeated trials and an explicit reproduction criterion before using an observed pass as evidence.

Such reduction also requires us to specify what is preserved. A filtered trace contains selected evidence from an execution, but deleting messages from the trace does not show that the corresponding execution steps were unnecessary. We keep the original artifact and record the reduction rule. For a reduced executable reproduction, we need a test that checks for the same target failure. This differs from the Minimal Trace pattern in Section 6.4, which describes a baseline with default configuration and no interaction.

For performance changes, we can also consider causal profiling. Coz uses controlled delays to estimate how virtually speeding up selected code affects program progress (Curtsinger and Berger 2015). It helps distinguish the code where time is spent from the code whose optimization may improve throughput or latency. The prediction depends on the observed workload and experiment, so an implemented change still needs to be checked for the intended result and the required preservation properties.

7. Worked Example: Intermittent Tail Latency After a Release

Consider a distributed API where, after a release, median latency remains stable but p99 latency and the timeout rate increase. Average CPU utilization is still normal. We can restart instances, enable debug logging, collect profiles or dumps, roll back, or change retry behavior. This is an illustrative example of the analysis process, not a measured evaluation of the proposed framework.

7.1 Problem description

We need to restore successful request completion within the service objective, preserve order correctness, and avoid retry amplification. We have thirty minutes for a containment decision and one business day for a causal explanation supported by evidence. Tracing overhead must remain below the agreed production limit, payload data must not be captured, and rollback must preserve schema compatibility.

The initial hypotheses are:

1. a new lock or queue creates intermittent serialization;
2. database waits increased for one request class;
3. retry behavior amplifies a downstream slowdown;
4. garbage collection pauses grew because of retention;
5. a telemetry or clock defect exaggerates the observed tail.

7.2 Select evidence by decision value

Full debug logging produces large volumes of data and perturbs execution, while still not guaranteeing that we can recover causality across services. A targeted trace of slow and failed requests, with the corresponding deployment version and retry count, may help distinguish between several hypotheses. A CPU and allocation profile collected over a limited interval can show compute and allocation behavior, but garbage-collection events or heap evidence are needed to examine pauses and retention more directly. Coordinated thread snapshots triggered by latency can show waiting relationships, although finding a lock owner may require additional runtime or dump data.

We initially select targeted tail sampling, a small reference sample of ordinary requests, retry and queue fields, a profile collected over a limited interval, and triggered thread

snapshots. We also collect garbage-collection event records and client-observed latency for the same intervals. Independent request and failure counters are also kept for comparisons across the request population. We defer a heap dump unless allocation or retention evidence increases its expected decision value enough to justify capture. Rollback is prepared in case containment becomes necessary.

7.3 Analyze patterns and competing mechanisms

When applying the Time Delta pattern, we find the longest delays between a downstream timeout and local completion. Relative Density shows more retry messages relative to successful downstream responses within the compared request classes and sampling strata, with the same inclusion rules used throughout the comparison. Statement Current increases in these activity windows, although independent completion counters show reduced throughput. We also find Silent Messages in one activity strand while retries continue elsewhere. Thread snapshots show many requests waiting on the bounded executor, and queue measurements confirm that it is saturated.

These diagnostic indicators point to retry amplification with queue saturation. The garbage-collection events recorded during the captured intervals show no corresponding increase in pause duration. This weakens the garbage-collection hypothesis, although an unobserved retention problem is still possible. Database spans for the affected request class remain close to the baseline, without excluding an interaction under another workload. Service-side monotonic durations, client-observed latency, causal trace relationships, and waiting threads all indicate the slowdown. A clock or telemetry defect is therefore unlikely to be the primary explanation for this interval, but this does not establish that every measurement is correct. Comparing deployments also shows that the new release shortened the retry interval.

7.4 Choose a reversible intervention

We have three admissible actions: roll back the release, restore the previous retry interval through configuration, or scale the executor. Scaling may hide the mechanism and increase downstream pressure. A rollback affects more of the system but may be particularly useful for restoring service. Restoring the retry interval is the smallest change directed at the proposed mechanism, and we can first apply it to a canary.

We restore the interval for a small portion of traffic and compare it with traffic using the new interval at the same time. We monitor successful completion, queue depth,

downstream request rate, and error distribution, also checking for differences in the traffic mix. In this example, tail latency and queue depth recover on the canary, with no observed increase in failed orders. This supports applying the change more widely while keeping rollback available. It also strengthens the proposed explanation, although a small canary does not prove that the change is safe under every workload.

7.5 Close and revise

We limit the final conclusion to the observed workload and release: shortened retry intervals amplified a downstream delay and saturated a bounded executor. An interaction specific to the database remains a possible explanation outside the observed workload. Further work includes retry-budget instrumentation, a saturation alert with an associated response, and a load test with injected, correlated downstream delay, where results are grouped by database wait class. A configuration constraint keeps the retry policy within the declared downstream capacity limits.

The selected evidence allowed us to change the decision before the incident deadline. The resulting intervention addressed the proposed mechanism while preserving the stated privacy and service constraints. We retain the unresolved alternatives in the report for further investigation.

8. Failure Modes and Countermeasures

8.1 Optimizing the proxy

A smaller log, fewer debugger commands or alerts, or a shorter time to close an incident may improve the corresponding metric while making diagnosis less effective. When a measure becomes a target, optimizing it may expose a difference between the measure and the original purpose (Goodhart 1975). We therefore examine speed and volume together with decision accuracy, reopened cases, incident recurrence, violations of required limits, and qualitative review.

8.2 Over-instrumentation

Instrumentation may introduce contention, consume storage, and require additional analysis. When many signals are examined without accounting for multiple comparisons, false alarms may also increase. Telemetry whose ownership is unclear or whose dependencies are unused can create continuing maintenance work, by analogy with

hidden technical debt in machine learning systems (Sculley et al. 2015). We therefore relate each costly signal to a decision or a class of hypotheses and review instrumentation that is no longer used.

8.3 Under-instrumentation and false absence

When we do not see an expected message, we first consider whether it could have been recorded. The event may not have occurred, or its absence may be due to missing coverage, sampling, truncation, a failed emitter, or a broken telemetry pipeline. Missing Component, Missing Data, and Missing Message distinguish several such cases (Vostokov 2023, 207–209). Information about instrumentation health and coverage is therefore part of the diagnostic evidence.

8.4 Premature scalarization

If we combine everything into one incident or hypothesis score too early, asymmetric risks and incomparable objectives may be hidden. We keep the vector while examining safety, service, evidence, privacy, and reversibility separately. Conditions that must not be violated are expressed as hard constraints.

8.5 Local optimization

One team may reduce its analysis time by leaving another team to reconstruct missing evidence. We can draw an analogy with Amdahl's law, where accelerating one part leaves the rest unchanged, although the law concerns a different setting (Amdahl 1967). Here we measure the time needed to reach a decision across team transfers, repeated work, and recovery. The utilization of an individual team does not by itself show progress in resolving the incident.

8.6 Tool-induced certainty

A debugger extension, correlation engine, or solver may produce precise output even when symbols are incorrect, data is incomplete, clocks are incomparable, or the model is unsuitable. Successful tool execution therefore needs to be accompanied by checking the assumptions used to interpret its output. The report includes the relevant versions, tolerances, and coverage.

8.7 Automation bias

We may select a highly ranked hypothesis without looking at its supporting evidence. To avoid this, we keep the evidence for and against it, the tests or artifacts still needed to distinguish it from alternatives, the remaining uncertainty, and the consequences of an error. The analysis also allows alternatives that cannot yet be compared and requests for further evidence.

8.8 Destructive debugging

A restart, patch, or data correction may erase evidence of the mechanism or make the incident worse. We therefore prepare acquisition before changes, canaries, the means to roll back, and comparisons after changes. An irreversible intervention requires stronger evidence and authorization at the appropriate wider level.

8.9 Search without a stopping rule

An analysis without a stopping rule may consume the available time and produce an explanation fitted too closely to the collected narrative. We define in advance when the evidence is sufficient, when escalation is needed, and when a conclusion should be withheld. Unresolved alternatives remain part of the result.

8.10 Optimization theater

A formal score may be misleading if we cannot inspect its weights, baselines, exclusions, and assumptions. We check whether a different plausible weighting or a contradictory observation would change the recommendation. A small model with explicit reasoning may be more useful than a complex score whose result cannot be questioned.

9. Resulting Context

The resulting diagnostic report relates the original problem to the collected artifacts, their transformations, the considered hypotheses, and the recommended action. It also states what was not established and under which conditions the decision would need to be reconsidered.

We expect the following benefits:

- faster distinction between hypotheses without collecting data indiscriminately;
- clearer trade-offs between recovery, evidence, risk, privacy, and cost;
- reproducible analysis sequences expressed through patterns and records of artifact origin and processing;
- safer debugging through explicit statements of what is preserved and how a change can be rolled back;
- observability that supports diagnostic decisions instead of simply adding dashboards;
- learning from the trace of the diagnostic analysis itself;
- conclusions with stated limits that remain useful when certainty is unavailable.

This process requires us to maintain pattern catalogs, acquisition rules, evidence metadata, and checks of instrumentation health. Information value estimates may be wrong, and formal notation may hide assumptions that are not well supported. We therefore need to check whether the additional work reduces repeated analysis and makes the resulting decisions easier to reconstruct.

The result remains provisional because software, workloads, tools, and diagnostic questions may change. If an assumption no longer holds, the preserved evidence and analysis history allow us to continue the process from the corresponding point.

10. Known Uses

The following are typical application scenarios. Dapper and delta debugging provide published examples of some constituent techniques (Sigelman et al. 2010; Zeller and Hildebrandt 2002). Such results do not constitute an empirical validation of the whole framework proposed here.

- **Crash triage:** selecting the dump type, preparing symbols, and choosing the first debugger commands according to the crash class and its consequences.
- **Hang and deadlock analysis:** coordinating thread snapshots, wait-chain evidence, and live inspection within the time available before a restart.

- **Memory leak diagnosis:** scheduling repeated snapshots, comparing growing memory regions, and requesting more heap detail after the initial evidence localizes the mechanism.
- **Performance regression:** profiling representative workloads along the complete service path, including tail behavior and failures, while avoiding bias in benchmark selection.
- **Distributed incident response:** collecting traces of slow or failed requests, correlating retries and queues, and recording how the traces were sampled.
- **Observability design:** allocating telemetry resources to coverage, resolution, correlation, and trust, instead of simply recording more events.
- **Alert engineering:** selecting thresholds and aggregation rules according to the expected value of the response, the cost of interruption, and the remaining risk.
- **Release debugging:** using canaries, feature flags, rollback, and evidence with recorded versions to construct reversible experiments.
- **Diagnostic automation:** expressing repeated debugger and log-analysis operations as pattern sequences that can be inspected and changed by the analyst.
- **Post-incident learning:** analyzing the diagnostic narrative to improve tools, acquisition procedures, training, and interactions between teams.

11. Related Patterns and Concepts

This article is related to the following pattern-oriented software diagnostics concepts and methodologies:

- A.C.P. Root Cause Analysis Methodology (Artifacts, Checklists, and Patterns) and its later A.P.M. formulation (Artifacts, Patterns, and Mechanisms);
- Adjoint Space and Paratext;
- Basic Facts;
- Dynamic Memory Corruption;
- Elementary Analysis Patterns;
- Execution Residue;
- Focus of Tracing;
- Measurement;
- Message Essence;

- Minimal Trace;
- Missing Component, Missing Data, and Missing Message;
- Pattern-Oriented Diagnostic Analysis Process;
- Rapid Software Diagnostics Process;
- Relative Density;
- Right First Time Software Diagnosis;
- Significant Event;
- Silent Messages and Trace Porosity;
- Small DA+TA;
- Software Diagnostics Canvas;
- Statement Density and Current;
- Systematic Software Diagnostics;
- Time Delta.

Related optimization concepts include multiobjective and constrained optimization, value of information, sequential experimental design, optimal stopping, causal inference, control, and reversible design. Delta debugging and causal profiling give us concrete examples of connections to optimizing reproductions and interventions.

12. Conclusion

Here we considered optimization as the selection of diagnostic and corrective actions and their order, taking into account the stated objectives, constraints, and uncertainty. A smaller trace, a faster command, or a shorter recovery time is useful according to its effect on the decision supported by the evidence.

Observability determines which differences in software state and behavior can be recovered, and artifact acquisition provides the corresponding evidence of state and history. Analysis patterns help us recognize software structure and behavior, while mechanisms relate the resulting observations to possible causes. Debugging tests or changes the proposed mechanism. After the change, we use monitoring to check whether the intended improvement was achieved and the required invariants were preserved. New evidence may require us to reconsider how the original problem was formulated.

Another analyst should be able to reconstruct why the artifacts were collected, which patterns were found, how alternative mechanisms were considered, and why the intervention was selected. The same record shows what remains uncertain and which observations would require further analysis. Diagnostic optimization therefore also produces artifacts that can be used to improve the diagnostic process itself.

References

Amdahl, Gene M. 1967. “Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities.” In *Proceedings of the AFIPS Spring Joint Computer Conference*, 483–485. https://doi.org/10.1145/1465482.1465560.

Boyd, Stephen, and Lieven Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press. https://doi.org/10.1017/CBO9780511804441.

Curtsinger, Charlie, and Emery D. Berger. 2015. “Coz: Finding Code That Counts with Causal Profiling.” In *Proceedings of the 25th Symposium on Operating Systems Principles*, 184–197. https://doi.org/10.1145/2815400.2815409.

Dean, Jeffrey, and Luiz André Barroso. 2013. “The Tail at Scale.” *Communications of the ACM* 56 (2): 74–80. https://doi.org/10.1145/2408776.2408794.

Forsgren, Nicole, Jez Humble, and Gene Kim. 2018. *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press.

Goodhart, Charles A. E. 1975. “Problems of Monetary Management: The U.K. Experience.” In *Papers in Monetary Economics*. Reserve Bank of Australia.

Howard, Ronald A. 1966. “Information Value Theory.” *IEEE Transactions on Systems Science and Cybernetics* 2 (1): 22–26. https://doi.org/10.1109/TSSC.1966.300074.

Humble, Jez, and David Farley. 2010. *Continuous Delivery*. Addison-Wesley.

International Organization for Standardization. 2023. *ISO/IEC 25010:2023 Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — Product Quality Model*. https://www.iso.org/standard/78176.html.

Lamport, Leslie. 1978. “Time, Clocks, and the Ordering of Events in a Distributed System.” *Communications of the ACM* 21 (7): 558–565. https://doi.org/10.1145/359545.359563.

Microsoft. n.d. “MINIDUMP_TYPE (minidumpapiset.h).” *Microsoft Learn*. Accessed September 11, 2026. https://learn.microsoft.com/en-us/windows/win32/api/minidumpapiset/ne-minidumpapiset-minidump_type.

Nocedal, Jorge, and Stephen J. Wright. 2006. *Numerical Optimization*. 2nd ed. Springer. https://doi.org/10.1007/978-0-387-40065-5.

OpenTelemetry Authors. n.d. “Sampling.” *OpenTelemetry Documentation*. Accessed September 11, 2026. https://opentelemetry.io/docs/concepts/sampling/.

Sculley, D., Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, and Dan Dennison. 2015. “Hidden Technical Debt in Machine Learning Systems.” In *Advances in Neural Information Processing Systems 28*, 2503–2511.

Shannon, Claude E. 1948. “A Mathematical Theory of Communication.” *Bell System Technical Journal* 27 (3–4): 379–423, 623–656. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x.

Sigelman, Benjamin H., Luiz André Barroso, Mike Burrows, Pat Stephenson, Manoj Plakal, Donald Beaver, Saul Jaspan, and Chandan Shanbhag. 2010. *Dapper, a Large-Scale Distributed Systems Tracing Infrastructure*. Technical report. Google. https://research.google/pubs/dapper-a-large-scale-distributed-systems-tracing-infrastructure/.

Simon, Herbert A. 1955. “A Behavioral Model of Rational Choice.” *Quarterly Journal of Economics* 69 (1): 99–118. https://doi.org/10.2307/1884852.

Vostokov, Dmitry. 2023. *Trace, Log, Text, Narrative, Data: An Analysis Pattern Reference for Information Mining, Diagnostics, Anomaly Detection*. 5th ed., rev. 5.04. OpenTask. ISBN 978-1-912636-58-7.

Vostokov, Dmitry. 2024. *Theoretical Software Diagnostics: Collected Articles*. 4th ed., rev. 4.00. OpenTask. ISBN 978-1-912636-91-4.

Vostokov, Dmitry. 2026. *Optimization Across Disciplines: A Transdisciplinary Field Guide to Better Choices, Designs, Systems, and Works*. Version 33. OpenTask. ISBN 978-1-919135-00-7.

Zeller, Andreas, and Ralf Hildebrandt. 2002. “Simplifying and Isolating Failure-Inducing Input.” *IEEE Transactions on Software Engineering* 28 (2): 183–200. https://doi.org/10.1109/32.988498.