



Equivalent Optima, Different Policies: Certified Distribution-Preserving Reformulations for AI Decision Systems

Craig Steven Wright 

University of Exeter Business School, Streatham Campus, Rennes Drive, Exeter EX4 4PU, UK; cw881@exeter.ac.uk

How To Cite: Wright, C.S. Equivalent Optima, Different Policies: Certified Distribution-Preserving Reformulations for AI Decision Systems. *AI Engineering* 2026, 2(2), 20. <https://doi.org/10.53941/aieng.2026.100020>

Received: 23 September 2026

Revised: 8 October 2026

Accepted: 13 October 2026

Published: 21 October 2026

Abstract: A representation change can leave an optimization problem's feasible decisions and costs intact while altering how a stochastic decision layer uses it. We develop separate tests for the decoded probability law and the complete-policy relative-entropy objective of finite log-linear policies with deterministic decoding and compatible positive reference laws. Equality of the weighted residual-feature spectra across decoder fibres characterizes preservation of the decoded law for every natural parameter. Given that equality, a common spectrum concentrated at one point also preserves the relative-entropy objective; a spectrum with several atoms adds a nonnegative conditional-divergence penalty. The penalties add across compatible reformulation stages, and an exact correction recovers the physical objective. For compact convex policy classes, preserving optimal values for every cost is stronger than preserving optimal physical policies. An exact rational checker and a bounded-integer compiler implement the certificates. In a designed six-period scheduling study with 20 cost vectors, four tied auxiliary bits per period give a median physical-policy total variation of 0.245 at unit regularization (95% percentile bootstrap interval, 0.224–0.254). The corrected and untied controls recover the physical implementation to floating-point accuracy. Checks with two optimizers and three initializations expose four unresolved gradient-tolerance failures. Analytically solved controls separate distortion caused by the representation from numerical optimization error.

Keywords: AI decision systems; optimization reformulation; probabilistic compilation; relative entropy; certificate checking; constrained learning

1. Introduction

In AI-enabled engineering, an optimization model often forms an interface within a larger decision system. Before its decisions are used, a dispatch, routing, allocation, or control model may undergo solver-specific transformation, become part of a learned decision layer, or be generated automatically. Feasibility and the best objective value remain necessary checks, but they need not exhaust the requirements imposed by that system. We ask whether a change in representation leaves the *consumer* with the same physical decision problem.

A two-choice example makes the distinction explicit. Let the physical decisions be $x = 0, 1$, with costs 0 and 1, respectively. The original representation lists each decision once. An equivalent reformulation lists $x = 0$ once and $x = 1$ twice; both copies retain cost 1 and decode to $x = 1$. Both deterministic problems have the unique optimum $x = 0$. With unit-temperature Gibbs sampling and a uniform reference over the listed states, however, the original physical probability is $p(1) = 1/(1 + e)$, whereas the reformulation gives $Dq(1) = 2/(2 + e)$. Feasibility, physical costs, and the deterministic optimum are unchanged, but the output laws differ. Giving each copy of $x = 1$ weight $1/2$, relative to weight 1 for $x = 0$, restores the original law after normalization. A second example keeps the decoded law unchanged at every parameter. With one auxiliary bit scored by the same parameter as x , and physical cost coefficient $a = \tau = 1$, the complete-policy objective has optimum $p(1) = \text{sigmoid}(-1/2) \simeq 0.378$, while the physical objective has optimum $\text{sigmoid}(-1) \simeq 0.269$ (Example 1).



Consider integer production decisions x_t . If each x_t is replaced by nonnegative integer variables y_t, z_t with $x_t = y_t + z_t$, both the physical schedule and any cost depending only on x are preserved. Yet each physical decision x_t has $x_t + 1$ complete descriptions under the replacement. This multiplicity is immaterial to a deterministic optimizer. When a stochastic consumer assigns weight to complete descriptions, it can alter preferences without any change to the physical objective. Projected counting and sampling already separate retained decisions from auxiliary descriptions [1–3]; certified projected compilation can also verify the logical projection [4]. What remains to be specified is the certification obligation when the downstream AI consumer imposes its own parameterization and regularizer.

Inference and learning place different demands on a representation. If two implementations give the same physical policy p_θ at every θ , any loss that depends only on that policy is the same for both. A difference appears when the encoded implementation optimizes another quantity, such as relative entropy of the complete policy including auxiliary variables. With shared parameters, the auxiliary term varies with the parameters that determine the physical policy even though decoding discards the auxiliaries.

This paper makes four contributions. First, it gives an exact residual-spectrum characterization of all-parameter preservation of the physical law for finite log-linear consumers. Second, conditional on a passing distribution certificate, it identifies when a complete-policy relative-entropy objective is also preserved, proves composition without cancellation, and provides an exact corrective objective. Third, it separates this parameterized result from a complementary compact-convex policy-class result, where a constant auxiliary penalty can shift values without changing any optimal physical policy. Fourth, it implements these contracts in a rational-arithmetic checker and a bounded-integer compiler, supported by exhaustive and structured tests and a controlled learning study with analytic global controls.

Section 4 tests these mechanisms on designed scheduling instances. Small cases are checked exhaustively, larger ones by dynamic programming, and matched learning runs test whether corrected and untied implementations recover the intended policy. A representation must be certified for the consumer that will use it and checked again whenever its support, reference, score features or encoded restrictions change.

Engineering requirement: certify the representation for its consumer.

1. **Specify the interface:** record the supports X, Z , decoder d , physical cost c , positive references μ, ν , score features S, T , and parameter domain. Check $D\nu = \mu$.
2. **Establish inference preservation:** compare the full weighted residual spectra; selected moments and sampled parameters do not suffice (Theorem 1).
3. **Establish or restore learning equivalence:** for complete-policy KL, require a point-mass common spectrum or apply the correction $J_{\text{corr}} = J_X$ (Theorem 2; Corollary 1). Losses determined solely by physical outputs require no further joint-KL certificate.
4. **Revalidate after restrictions:** check the surviving support, refresh the reference when valid, and test both contracts again. The restricted case requires its own certificate (Section 3.4.1).

2. Related Work

2.1. Optimization Reformulation and Learned Optimization

Mathematical-programming reformulation provides the first layer of the problem. Liberti distinguishes exact reformulations, narrowings, relaxations, and approximations, and allows multiple reformulated optima to represent one original optimum [5]. Verified reductions add constructive proof obligations for feasibility and objective relations [6], while FLARE develops machine-checkable MILP reformulation proofs with forward and backward mappings [7]. Rewriting systems and model interfaces operationalize these ideas in optimization software [8,9]. These contracts define the transformed optimization problem. They do not, by themselves, specify a reference probability measure over the transformed representation.

A second literature embeds optimization inside learning. OptNet differentiates through quadratic-programming layers; differentiable convex layers organize canonicalization and solution differentiation; black-box combinatorial differentiation and DiffOpt provide alternative interfaces between optimization and learned parameters [10–13]. Decision-focused learning trains predictions according to the quality of the decisions obtained from an optimization problem, as reviewed by Mandi et al. [14]. This learning-augmented optimization setting connects a learned component to a downstream solver. Our certificates concern another interface within such a pipeline: whether changing the optimization representation preserves a declared stochastic output law and, when the learner uses complete-policy KL, its regularized objective. They complement decision-focused training criteria by checking

representation semantics. Automated model generation creates the same need to separate a proposed representation from the guarantees subsequently checked on it [15].

2.2. Projection, Counting, and Probability-Preserving Compilation

Counting and probabilistic inference already account explicitly for auxiliary descriptions. Knowledge compilation identifies the operations that remain tractable under a change of representation [16]. Algebraic model counting extends weighted evaluation [17], and compiled structures also support gradient calculations [18]. ProCount separates additive and existential projection; WAPS samples weighted projected assignments; distribution-aware SAT sampling specifies the target distribution directly [1–3]. Bryant et al. go further by certifying projected knowledge compilation, including the elimination of quantified variables and the properties needed for counting [4]. These works already treat multiplicity and projection explicitly.

Probabilistic compilers address an even stronger semantic target. Dice compiles discrete probabilistic programs to weighted formulas with source-semantics guarantees [19]; verified density compilers and probabilistic-program semantics extend such reasoning to richer settings [20–22]. The distinction studied here arises after that probability layer: A representation can induce the correct decoded physical law at every parameter while a complete-policy learning objective remains different. The relevant correctness target therefore depends on the consumer, not only on the projected measure.

Exact construction of projected masses can itself be difficult: counting complexity and treewidth-based analyses place structural limits on what any general finite implementation can promise [23,24]. We therefore separate the cost of checking an explicit certificate from the potentially much larger cost of constructing the represented support.

2.3. AI Engineering Assurance and Model Interfaces

In Ying et al., model transformation and learned warm starts coexist with feasibility and dispatch constraints enforced by the optimization procedure [25]. Rabbanian et al. couple multi-agent reinforcement learning to an operational charging model. Their comparison with an exact small-scale formulation retains explicit engineering objectives and constraints [26]. Zhang treats syntactic synthesis, executable behavior, and formal verification separately because their guarantees can fail independently [27]. Kumar and Bist assemble drift detection, adversarial testing, governance auditing, and experiment tracking into a combined assurance procedure [28]. Yuan et al. make target-sample admissibility an explicit requirement of open-set adaptation through a three-criterion gate [29]. Pandey’s enterprise architecture assigns data quality, provenance, access control, and downstream AI augmentation to distinct interfaces [30].

Our question is when an optimization representation preserves the stochastic semantics required by an AI decision layer. For the systems considered here, feasibility, the decoded law and the complete-policy learning objective each require an appropriate check. The definitions below separate these obligations. Table 1 summarizes the correctness targets.

Table 1. Correctness targets and their relation to the present consumer-specific contract. The table compares guarantees, not runtime performance.

Target	Primary Object Preserved	Additional Question Addressed Here
Optimization reduction [6,7]	Feasibility and a declared objective relation	Does the transformed representation preserve the physical stochastic law?
Weighted projection [1,2,4]	Weighted mass on retained assignments	Does the learner’s complete-policy objective remain the same after projection?
Probabilistic compilation [19–21]	Source probability semantics	Does a downstream consumer impose an additional representation-dependent regularizer?
AI engineering assurance [25–30]	Application-specific constraints, verification, monitoring, or admissibility	Which semantic obligation belongs specifically to the representation–consumer interface?
Present implementation	Finite declared semantics and a bounded rewrite fragment	Separate distribution, joint-KL, and policy-class decisions with explicit failure outcomes.

2.4. Regularization, Multiple Descriptions, and Auxiliary Variables

Regularized MDPs and relative-entropy policy search make the reference policy explicit in stochastic optimization [31,32]. Work on action redundancy demonstrates how nominally distinct actions with the same effect can distort entropy-based exploration [33]. GFlowNets make the distinction between probabilities assigned to construction paths and those assigned to terminal objects [34]. Perturb-and-MAP and stochastic-softmax methods further require the randomization mechanism to be specified when characterizing an optimization-based sampler [35–37].

Conditional divergence gaps occur in auxiliary-variable variational models [38]. SurVAE Flows uses stochastic right inverses supported on fibres and permits composition of surjective transformations [39]. Exponential-family log-partition identities and missing-information decompositions are also standard tools [40,41]. We use these identities to obtain finite, checkable tests for a declared optimization representation and the policy class that consumes it.

3. Materials and Methods

3.1. Physical Decisions, Encoded Policies, and Consumer Contracts

3.1.1. Finite Supports and Declared Semantics

Let X be a nonempty finite set of physical decisions and Z a nonempty finite set of complete encoded assignments. A deterministic decoder $d : Z \rightarrow X$ identifies the physical decision implemented by each assignment. Write $Z_x = d^{-1}(x)$ and let D be the linear pushforward operator,

$$(Dq)_x = \sum_{z \in Z_x} q_z. \quad (1)$$

The physical meaning of d must be specified independently of equality of objective values. Two allocations to different machines are not automatically the same physical decision merely because their costs coincide.

The physical optimization objective is $c : X \rightarrow \mathbb{R}$. Its encoded counterpart is $c \circ d$. Surjectivity then preserves physical feasibility and the minimum physical cost. This objective is distinct from the score functions that parameterize a learned policy. Let μ and ν be strictly positive probability distributions on X and Z , respectively. Every theorem involving the reference conditional kernel assumes

$$D\nu = \mu, \quad K_x(z) = \nu_z / \mu_x \quad (z \in Z_x). \quad (2)$$

Zero reference mass is handled by support restriction, not by division by zero. Let $X_+ = \{x : \mu_x > 0\}$ and $Z_+ = \{z : \nu_z > 0\}$. When $D\nu = \mu$, the restricted decoder maps Z_+ onto X_+ , and the theorem applies there. A target state with positive physical reference mass but no positive-mass encoding makes preservation impossible. This restriction concerns the stochastic model: removing a physically feasible zero-reference decision still requires a separate check of the original deterministic feasible set.

A reference measure is part of the model. Uniformity over encoded assignments is not interchangeable with uniformity over physical decisions. A certificate must therefore carry X, Z, d, μ, ν in addition to feasibility and objective information.

Table 2 summarizes the core notation.

Standing assumptions:

- (A1) X and Z are nonempty finite supports.
- (A2) The reference laws μ and ν are normalized and strictly positive on these supports. References with zeros are first restricted as described above; all stochastic conclusions then concern X_+ and Z_+ .
- (A3) The decoder d is deterministic and $D\nu = \mu$. With $\mu > 0$, this forces d to be onto X . If x has positive physical reference mass but its decoder fibre is empty, every encoded policy assigns decoded probability zero to x , whereas the physical log-linear policy assigns positive probability at each finite parameter; the distribution contract therefore fails.
- (A4) Repeated feature and residual vectors are allowed; conditional reference weights at identical residual vectors are summed.
- (A5) The parameterization may be nonminimal or non-identifiable. The certificate uses the induced residual spectra.
- (A6) The universal characterization uses $\mathbb{R}^k$ or a nonempty open subset. Under a linear restriction $\theta = B\omega$, it uses the projected features on an open domain of ω ; arbitrary nonlinear parameter manifolds require a separate analysis.

Table 2. Core notation. The physical cost and policy score are different objects.

Symbol	Meaning	Symbol	Meaning
X, Z	Physical and encoded supports	d, D	Decoder and linear pushforward
μ, ν	Positive reference laws	K_x	Reference conditional law on fibre x
c	Physical operational cost	S, T	Physical and encoded score features
θ	Shared natural parameter	τ	Relative-entropy coefficient
R, σ_x	Residual feature and its fibre law	ψ, g	Residual log partition and conditional KL
L	Reference conditional lifting operator	ρ_Q	Best feasible conditional KL on a policy fibre

3.1.2. The Baseline Gibbs Contract

For inverse temperature $\beta > 0$, the intended policy is

$$p_c(x) = \frac{\mu_x e^{-\beta c(x)}}{\sum_y \mu_y e^{-\beta c(y)}}. \quad (3)$$

Equivalently, it uniquely minimizes $\mathbb{E}_p c + \beta^{-1} \text{KL}(p \| \mu)$ over the physical simplex. We use the convention $0 \log 0 = 0$.

Proposition 1 (Reference mass test). *Let $w_z > 0$ be arbitrary encoded weights and let $q_c(z) \propto w_z e^{-\beta c(dz)}$. Define $W_x = \sum_{z \in Z_x} w_z$. Then $Dq_c = p_c$ if and only if W_x / μ_x is constant over X .*

Proof. Summing q_c on a fibre gives a mass proportional to $W_x e^{-\beta c(x)}$. Since every physical mass is positive, equality of the normalized probabilities is equivalent to equality of all ratios W_x / μ_x . This argument applies even to a single fixed finite cost vector. $\square$

This proposition is a standard aggregation identity. A normalized conditional kernel K_x gives one exact correction, $w_z = \mu_{d(z)} K_{d(z)}(z)$. Choosing a uniform kernel yields inverse-multiplicity weights. Nonuniform kernels provide equally valid corrections.

Scaling the objective requires a separate check. If an encoded consumer uses $c' = ac + b$ with $a > 0$, preservation of the physical exponential weights requires the inverse-temperature relation $\beta' a = \beta$. A general monotone transformation retains the minimizers without necessarily retaining the finite-temperature law.

3.1.3. Two Contracts for Parameterized Consumers

Write $S : X \rightarrow \mathbb{R}^k$ and $T : Z \rightarrow \mathbb{R}^k$ for the physical and encoded score features, respectively, and define

$$p_\theta(x) = \mu_x \exp\{\theta^\top S(x) - A_X(\theta)\}, \quad (4)$$

$$q_\theta(z) = \nu_z \exp\{\theta^\top T(z) - A_Z(\theta)\}, \quad (5)$$

Here the A terms denote log partition functions and $\theta \in \mathbb{R}^k$. The features specify the policy architecture. They do not replace the physical engineering objective c .

The *distribution contract* requires $Dq_\theta = p_\theta$ at every point of the declared parameter domain. The *parameter-wise joint-KL contract* additionally requires

$$\text{KL}(q_\theta \| \nu) = \text{KL}(p_\theta \| \mu). \quad (6)$$

When both contracts hold, the same physical cost and coefficient $\tau > 0$ produce identical objectives at each parameter. The second contract is deliberately stronger than equality of the optimal physical policy for a particular cost.

For clarity, the objectives compared throughout the learning analysis are

$$\begin{aligned} J_X(\theta) &= \mathbb{E}_{p_\theta} c + \tau \text{KL}(p_\theta \| \mu), \\ J_Z(\theta) &= \mathbb{E}_{q_\theta} [c \circ d] + \tau \text{KL}(q_\theta \| \nu), \end{aligned} \quad \tau > 0.$$

The coefficient τ is fixed in each comparison. Relative entropy is directed from the policy to its declared reference: $\text{KL}(r \| s) = \sum_i r_i \log(r_i / s_i)$, with natural logarithms and the convention $0 \log(0 / s_i) = 0$ for $s_i > 0$. Both references are normalized probability distributions, fixing the normalization convention. For unrestricted

Gibbs minimization this convention corresponds to inverse temperature $\beta = 1/\tau$; a restricted parameterized policy family need not attain that unrestricted optimum.

3.2. Exact Residual-Spectrum Certificates

3.2.1. A Complete Finite Test for Output Distributions

Define the residual feature

$$R(z) = T(z) - S(dz), \quad \sigma_x = \sum_{z \in Z_x} K_x(z) \delta_{R(z)}. \quad (7)$$

The finite probability measure σ_x is the weighted residual spectrum of fibre x . It aggregates equal residual vectors, not merely distinct encoded states.

Definition 1 (Meaning of the declared objects). A declared decoder is the fixed map $d : Z \rightarrow X$ supplied with the model; its physical interpretation is an input to certification, not a conclusion inferred from equal costs. The decoder fibre Z_x contains every encoded assignment implementing x . In this finite setting, a reference measure is a normalized baseline probability law, μ on X or ν on Z ; these are the corresponding policies at $\theta = 0$. The residual feature $R(z)$ records the encoded score feature in excess of the decoded physical feature. The weighted residual-feature spectrum σ_x is its probability distribution under the conditional reference K_x , with weights summed whenever residual vectors coincide. A common spectrum is one probability measure σ satisfying $\sigma_x = \sigma$ for every physical decision. By physical implementation we mean direct evaluation of p_θ on X using the declared μ and S , without auxiliary assignments.

Theorem 1 (Residual-spectrum characterization). Under (A1)–(A6), with d onto the positive physical support, for the families (4)–(5), the following are equivalent: (i) $Dq_\theta = p_\theta$ for all θ in a nonempty open subset of $\mathbb{R}^k$; (ii) $Dq_\theta = p_\theta$ for every $\theta \in \mathbb{R}^k$; (iii) σ_x is the same finite measure for every $x \in X$.

Under a linear parameterization $\theta = B\omega$, apply the characterization to $B^\top S, B^\top T$ on an open domain of ω . No ambient-spectrum characterization is asserted for an arbitrary nonlinear parameter manifold.

Proof. Put $H_x(\theta) = \int e^{\theta^\top r} \sigma_x(dr)$. Fibrewise summation in (5) gives

$$(Dq_\theta)_x = \frac{\mu_x e^{\theta^\top S(x)} H_x(\theta)}{\sum_y \mu_y e^{\theta^\top S(y)} H_y(\theta)}. \quad (8)$$

All factors are positive. Equality with (4) is therefore equivalent to $H_x(\theta) = H_y(\theta)$ for every pair of fibres.

If (iii) holds, their transforms agree everywhere, proving (ii), which implies (i). Conversely, suppose (i). For a pair x, y , combine their distinct residual vectors as $r_1, \dots, r_m$ and their weight differences as $a_1, \dots, a_m$. Then $\sum_j a_j e^{\theta^\top r_j} = 0$ on an open set. Choose an interior point θ_0 of that open set.

Choose a direction v outside the finite union of hyperplanes $v^\top(r_i - r_j) = 0$. The numbers $\lambda_j = v^\top r_j$ are distinct. For sufficiently small t , the point $\theta_0 + tv$ remains in the open set. Differentiating $\sum_j a_j e^{\theta_0^\top r_j} e^{t\lambda_j} = 0$ at $t = 0$ for orders $0, \dots, m-1$ gives a Vandermonde system with nonzero determinant. Thus every coefficient $a_j e^{\theta_0^\top r_j}$ vanishes, and therefore every $a_j = 0$. The two finite measures are equal, and the pair was arbitrary. $\square$

Remark 1 (Parameter domains). Agreement at finitely many parameters is insufficient for a universal certificate. A linear restriction may eliminate residual directions; its certificate must therefore use the projected features. Failure in the larger ambient space does not settle the restricted problem. Agreement along a nonlinear curve likewise need not imply equality of the ambient spectra.

A bounded parameter region with nonempty interior provides the same certificate because it contains an open subset. A finite grid or a set confined to a lower-dimensional surface does not supply that implication. For a linear family $\theta = B\omega$, the relevant certificate compares projected residual vectors $B^\top R$, as stated in the theorem.

For rational weights and features, the theorem gives an exact rational certificate. Construct one dictionary per fibre, using $T(z) - S(dz)$ as the key and accumulating ν_z/μ_x at each key. Dictionary equality is exactly condition (iii). If the dictionaries differ, the checker returns two physical decisions, a residual vector, and the unequal rational weights. This witness disproves the universal contract. It need not show a discrepancy at a numerical parameter selected in advance.

The certificate requires the full weighted spectrum; low-order moments can agree when those spectra differ. For instance,

$$\sigma_0 = \frac{1}{2}\delta_{-1} + \frac{1}{2}\delta_1, \quad \sigma_1 = \frac{1}{8}\delta_{-2} + \frac{3}{4}\delta_0 + \frac{1}{8}\delta_2 \quad (9)$$

agree in mean and variance while their exponential transforms differ. A regression test in the separate archive rejects this pair, as required.

3.2.2. The Learning Penalty behind a Passing Distribution Certificate

Suppose the conditions of Theorem 1 hold, with common spectrum σ . Define

$$\psi(\theta) = \log \int e^{\theta^\top r} \sigma(dr), \quad g(\theta) = \theta^\top \nabla \psi(\theta) - \psi(\theta). \quad (10)$$

Since σ is a probability measure, $\psi(0) = 0$.

Theorem 2 (Distribution and joint-KL contracts separate). *Under (A1)–(A6) and a passing distribution contract, for every finite physical cost c ,*

$$\mathbb{E}_{q_\theta}[c \circ d] + \tau \text{KL}(q_\theta \| \nu) = \mathbb{E}_{p_\theta} c + \tau \text{KL}(p_\theta \| \mu) + \tau g(\theta), \quad (11)$$

$$\nabla g(\theta) = \text{Cov}_{\sigma_\theta}(R) \theta, \quad (12)$$

where $\sigma_\theta(dr) = e^{\theta^\top r - \psi(\theta)} \sigma(dr)$. On an open neighborhood of zero, the parameterwise joint-KL contract holds precisely when σ is a point mass. This condition also gives the contract throughout $\mathbb{R}^k$. Under $\theta = B\omega$, use the common law of $B^\top R$: the joint-KL contract holds on an open neighborhood of $\omega = 0$ if and only if this projected law is a point mass. The same condition then gives the identity throughout the linear family.

Equation (11) identifies the objective at each parameter. A nonzero penalty does not imply that every optimal physical policy changes; see Remark 2 and Theorem 4.

Proof. The conditional encoded law on Z_x is

$$q_\theta(z | x) = K_x(z) \exp\{\theta^\top R(z) - \psi(\theta)\}. \quad (13)$$

Taking relative entropy against K_x gives $\theta^\top \nabla \psi - \psi$. Because the common spectrum does not depend on x , neither does this conditional divergence. Apply the finite KL chain rule with $Dq_\theta = p_\theta$ to obtain (11). The derivative of the penalty is $\nabla g = (\nabla^2 \psi) \theta$. For the finite exponential sum defining ψ , the Hessian $\nabla^2 \psi$ is the covariance matrix in (12).

A spectrum $\sigma = \delta_{r_0}$ gives the linear log partition function $\psi(\theta) = \theta^\top r_0$, hence $g = 0$. For the reverse implication, assume $g = 0$ near the origin and expand along an arbitrary direction v : $g(tv) = \frac{1}{2}t^2 \text{Var}_\sigma(v^\top R) + O(t^3)$. The quadratic coefficient is therefore zero in every direction. Choosing the coordinate directions shows that each coordinate of the residual is constant on the positive-weight support. Thus the spectrum has only one atom. $\square$

A nonzero residual contributes only a common score shift if it is the *same vector on every encoded assignment*. Such a residual is harmless. When the common spectrum has several atoms, cancellation of a common partition factor still preserves the output law, but a parameter-dependent conditional KL term remains.

Under a passing distribution certificate, the nonnegative term has an explicit conditional-divergence interpretation:

$$g(\theta) = \sum_x p_\theta(x) \text{KL}(q_\theta(\cdot | x) \| K_x) = \text{KL}(\sigma_\theta \| \sigma) \geq 0.$$

The common-spectrum assumption makes the conditional divergence identical across fibres. Normalization of the common spectrum gives $g(0) = 0$ at the reference parameter. A point-mass common spectrum need not be located at the zero residual vector: any single common residual produces only a common score shift. With several atoms, the penalty may still vanish at parameters for which $\theta^\top R$ is constant on the spectrum. Thus failure of the universal joint-KL certificate does not imply a strictly positive penalty at every parameter.

Example 1 (Exact temperature inflation). *Take $X = \{0, 1\}$ and $Z = \{0, 1\}^{k+1}$, with the first bit as the decoded decision. Use uniform references on both supports and features $S(x) = x$ and $T(z) = \sum_{j=0}^k z_j$. The encoded law*

then factors into $k + 1$ Bernoulli distributions, each with success probability $p = \text{sigmoid}(\theta)$. Decoding recovers the original Bernoulli policy at every θ .

For physical cost $c(x) = ax$, minimizing the physical regularized objective gives $p^* = \text{sigmoid}(-a/\tau)$. Minimizing the complete-policy objective gives

$$\tilde{p}^* = \text{sigmoid}\left(-\frac{a}{(k+1)\tau}\right). \quad (14)$$

The physical feasible set and cost are unchanged; the extra regularizer alone moves the optimum.

Figure 1 evaluates the two global optima in Example 1 over physical cost coefficient a and regularization temperature τ . The surfaces show probabilities indexed by those parameters. They separate the optima of the physical and complete-policy objectives, with inference agreeing at each natural parameter.

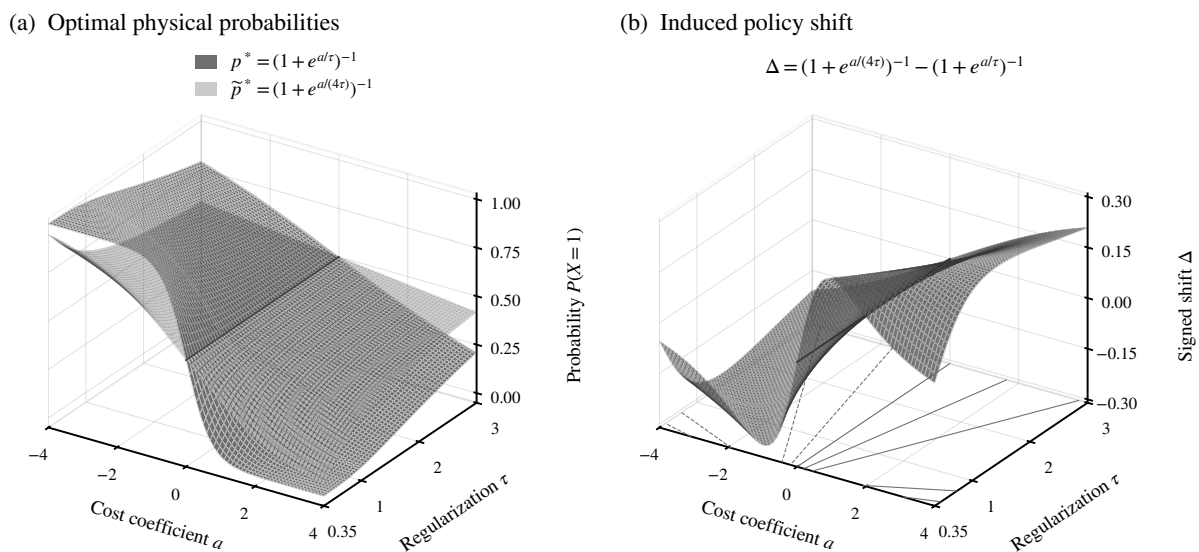


Figure 1. Closed-form policy surfaces from (14) for $k = 3$ tied auxiliary bits. (a) The physical optimum $p^* = 1/(1 + \exp(a/\tau))$ and decoded complete-policy optimum $\tilde{p}^* = 1/(1 + \exp(a/(4\tau)))$. (b) Their signed difference $\Delta = \tilde{p}^* - p^*$. The domain is $a \in [-4, 4]$, $\tau \in [0.35, 3]$; τ is a regularization coefficient in the same units as a , not a physical temperature measurement. Both probabilities equal $1/2$ at $a = 0$. The plotted coordinates are evaluated from the formulas; rendering code and coordinates are retained in the separate archive. This is an analytic architectural example, not field evidence.

Proposition 2 (Constrained replicas have an analytic global optimum). *Let X be any nonempty finite constrained decision set and $\mu > 0$. On $Z = X^{k+1}$, decode the first component, take $\nu = \mu^{\otimes(k+1)}$, and define $q_\theta = p_\theta^{\otimes(k+1)}$. Physical inference is preserved for every θ . If the feature family represents the Gibbs policies for the cost c , the global optimum of the joint-regularized objective has physical law*

$$\tilde{p}_k^*(x) = \frac{\mu_x \exp\{-c(x)/[(k+1)\tau]\}}{\sum_y \mu_y \exp\{-c(y)/[(k+1)\tau]\}}. \quad (15)$$

This statement also holds when optimization is over the entire physical simplex before applying the product lift.

Proof. Product additivity gives $\text{KL}(q_\theta \parallel \nu) = (k+1) \text{KL}(p_\theta \parallel \mu)$, whereas the expected physical cost is $\mathbb{E}_{p_\theta} c$. For arbitrary p , subtracting the objective at (15) leaves $(k+1)\tau \text{KL}(p \parallel \tilde{p}_k^*) \geq 0$. Equality holds only at the stated law. Representability transfers this global minimizer to the parameterized family. No unconstrained factorization of X is assumed. $\square$

This construction supplies an exactly solvable multivariate control with nonuniform physical references and scheduling constraints. The standard Gibbs calculation gives an exact multivariate optimum, allowing the learning distortion in Section 4 to be checked independently of optimizer convergence.

Remark 2 (What a failed learning certificate does not imply). *Failure of (6) does not mean every physical optimum*

changes. Let $S(x) = (x, 0)$ and $R(u) = (0, u)$ with independent uniform bits. Parameters (a, b) govern the physical and auxiliary bits separately. The additional penalty depends only on b and can be minimized at $b = 0$ without changing the physical policy. Theorem 2 certifies a parameterwise identity, not an if-and-only-if statement about all optimal physical policies of arbitrary nonconvex parameterized classes.

3.2.3. Compositions and the Impossibility of Penalty Cancellation

Consider finite supports $X_0, \dots, X_L$, decoders $d_i : X_i \rightarrow X_{i-1}$, and positive references satisfying $D_i \mu_i = \mu_{i-1}$. All levels use the same natural parameter θ and feature maps S_i . Suppose each stage passes the distribution contract, with a common stage spectrum σ_i for $S_i(z) - S_{i-1}(d_i z)$.

Theorem 3 (Compositional spectrum and no cancellation). *Assume (A1)–(A6) at every stage. At each intermediate interface, the support, reference μ_i and feature map S_i coincide with those used by the next stage; every decoder is deterministic and every stage identity is exact. If parameters are tied through one linear map, apply that map to all feature declarations before checking the stages. In the resulting common effective parameter space, suppose every stage passes its distribution contract. Define the conditional reference kernel by $K_i(x_i | x_{i-1}) = \mathbf{1}_{\{d_i(x_i)=x_{i-1}\}} \mu_i(x_i) / \mu_{i-1}(x_{i-1})$ and the final reference law on X_L by $\nu_L = \mu_0 K_1 \cdots K_L = \mu_L$. Then the composite decoder passes the distribution contract with respect to μ_0 and ν_L . Its residual spectrum is the convolution $\sigma_1 * \cdots * \sigma_L$. Writing ψ_i and g_i as in (10),*

$$\psi_{1:L} = \sum_i \psi_i, \quad g_{1:L} = \sum_i g_i. \quad (16)$$

The composite parameterwise joint-KL contract holds on an open neighborhood of zero if and only if every stage spectrum is a point mass.

Proof. Generate the reference chain from $X_0 \sim \mu_0$ using the kernels K_i ; the terminal law is ν_L . At stage i , conditioning on the preceding history fixes X_{i-1} . The stage certificate then assigns the residual the law σ_i , regardless of that history. Successive conditioning factors the exponential transform into $\prod_i \exp \psi_i$. Meanwhile, the sum of the stage residuals telescopes to $S_L(X_L) - S_0(X_0)$. Its law is therefore the stated convolution, which satisfies the distribution criterion in Theorem 1. Taking logarithms and differentiating gives (16).

No stage can offset another's penalty: g_i is a nonnegative KL divergence. A zero total throughout a neighborhood forces every g_i to be zero there. Theorem 2 then requires a point mass at each stage. The reverse implication follows by applying the same theorem to those point masses. In covariance terms, the matrices at zero sum to zero only when each is zero, since every summand is positive semidefinite. $\square$

Parameter sharing is included in the theorem when the effective features agree at each interface. Restricting a valid identity to a nonlinear parameter map preserves it on that image, but completeness for a new nonlinear family requires a separate argument. Global normalization is absorbed by the partition functions; state-dependent reweighting changes the declared reference or features. Approximate equalities, stochastic decoders and encoded restrictions require additional checks, with the last case treated in Section 3.4.1.

Corollary 1 (Certified objective correction). *Consider the compatible passing layers in Theorem 3, and write J_Z for physical expected cost plus final joint KL. Then*

$$J_{\text{corr}}(\theta) = J_Z(\theta) - \tau \sum_{i=1}^L g_i(\theta) = J_X(\theta), \quad \nabla J_{\text{corr}} = \nabla J_Z - \tau \sum_i \text{Cov}_{\sigma_i, \theta}(R_i) \theta. \quad (17)$$

Exact deterministic optimization therefore follows the physical objective's trajectory when parameters, initialization, and update rule are identical.

Proof. Insert the additive penalty from (16) into (11) and remove its contribution to the objective. Differentiating the resulting identity is valid because the sums are finite. Starting at the same parameter, a deterministic update receives the same objective information in both implementations and therefore produces the same next parameter. Induction gives equality at every iteration. $\square$

Numerical evaluation should follow algebraic simplification of this identity, avoiding subtraction of large, nearly equal floating-point values. Equality of the mathematical gradients alone does not guarantee agreement between

independent stochastic gradient samples, differently preconditioned updates, or calls to an opaque optimizer. The reference consumer implements the reduced physical objective and separately reports the encoded objective and penalty.

The correction requires the declared physical decoder, target reference and physical score features as well as the encoded model. Once a spectrum's M distinct atoms are stored, log-sum-exp and tilted moments give ψ , g and its gradient using $O(Mk)$ scalar arithmetic, M exponentials and one logarithm if the reference normalizer is cached (two otherwise). Layer costs add. The covariance-vector product is $\mathbb{E}[R(R^\top \theta)] - \mathbb{E}[R](\mathbb{E}[R]^\top \theta)$, so a dense covariance matrix is unnecessary. Building a spectrum from an implicit formulation can dominate these evaluation costs. Since $J_{\text{corr}} = J_X$ identically, their Hessians agree in the same parameterization: the corrected problem has the physical problem's conditioning. Evaluating the difference $J_Z - \tau \sum_i g_i$ separately can introduce cancellation; algebraic reduction to J_X avoids that subtraction. Independent stochastic gradient samples and different update rules remain separate numerical considerations.

3.2.4. A Parameter-Bounded Learning Budget

The additive penalty can be bounded without solving the encoded learning problem. Let B_i be the Euclidean diameter of $\text{supp } \sigma_i$.

Proposition 3 (Penalty and physical-policy bounds). *For a certified chain,*

$$0 \leq g_{1:L}(\theta) \leq \frac{\|\theta\|^2}{8} \sum_i B_i^2, \quad \|\nabla g_{1:L}(\theta)\| \leq \frac{\|\theta\|}{4} \sum_i B_i^2. \quad (18)$$

Suppose the unconstrained physical Gibbs optimum for c and τ is attainable as p_{θ^*} . If $\tilde{\theta}$ has encoded objective at most ϵ above the global infimum, then

$$\text{KL}(p_{\tilde{\theta}} \| p_{\theta^*}) \leq g_{1:L}(\theta^*) + \epsilon/\tau. \quad (19)$$

Proof. Every scalar projection of a random vector supported on a set of diameter B_i has range at most B_i in a unit direction. For a scalar $Y \in [a, b]$, the inequality $\mathbb{E}[(Y - a)(b - Y)] \geq 0$ gives $\text{Var}(Y) \leq (\mathbb{E}Y - a)(b - \mathbb{E}Y) \leq (b - a)^2/4$. Consequently $\|\nabla^2 \psi_i\|_{\text{op}} \leq B_i^2/4$. Differentiating $g_i(t\theta)$ and integrating from zero to one gives $g_i(\theta) = \int_0^1 t \theta^\top \nabla^2 \psi_i(t\theta) \theta dt$. This proves both bounds after summing over stages.

Let $J_X(\theta) = \mathbb{E}_{p_\theta} c + \tau \text{KL}(p_\theta \| \mu)$. Approximate global optimality and Theorem 3 imply $J_X(\tilde{\theta}) + \tau g(\tilde{\theta}) \leq J_X(\theta^*) + \tau g(\theta^*) + \epsilon$. Discard the nonnegative left-hand penalty. The Gibbs identity $J_X(\tilde{\theta}) - J_X(\theta^*) = \tau \text{KL}(p_{\tilde{\theta}} \| p_{\theta^*})$ then yields (19). $\square$

A numerical optimizer terminating is not evidence that its global gap is ϵ . The bound is conditional on that assumption. The scalar and constrained-replica constructions have analytic global optima; the separate multidimensional tied-bit study reports numerical solutions without claiming global certification.

3.3. Policy-Class Compatibility beyond a Fixed Parameterization

3.3.1. The Induced Regularizer

The preceding section concerns a specified log-linear parameterization. We now consider optimization directly over a compact convex encoded class $\mathcal{Q} \subseteq \Delta(Z)$, with nonempty compact convex image $\mathcal{P} = D\mathcal{Q}$. Define the reference lift Lp by $(Lp)_z = p_{d(z)} K_{d(z)}(z)$. For $p \in \mathcal{P}$, define

$$\rho_{\mathcal{Q}}(p) = \min_{q \in \mathcal{Q}: Dq=p} \sum_x p_x \text{KL}(q(\cdot | x) \| K_x). \quad (20)$$

Conditionals on a zero-probability fibre are arbitrary, with zero contribution. Since the conditional term equals the difference of two continuous finite KL expressions, the minimum is attained on the compact fibre.

Proposition 4 (Induced objective). *Optimizing the complete-policy objective over $\mathcal{Q}$ is equivalent to*

$$\min_{p \in \mathcal{P}} \{ \langle c, p \rangle + \tau \text{KL}(p \| \mu) + \tau \rho_{\mathcal{Q}}(p) \}. \quad (21)$$

Moreover, $\rho_{\mathcal{Q}}(p) = 0$ if and only if $Lp \in \mathcal{Q}$.

Proof. Disintegrating $q_z = p_x q(z | x)$ gives

$$\text{KL}(q \| \nu) = \text{KL}(p \| \mu) + \sum_x p_x \text{KL}(q(\cdot | x) \| K_x). \quad (22)$$

Minimize first over the feasible fibre, then over p . The conditional divergence is zero exactly when $q(z | x) = K_x(z)$ on positive- p fibres. That fixes $q = Lp$, proving the second statement by attainment. $\square$

3.3.2. Preserving Values Is Stronger than Preserving Policies

Let $V_X(c)$ minimize $\langle c, p \rangle + \tau \text{KL}(p \| \mu)$ on $\mathcal{P}$, and let $V_Z(c)$ minimize the complete-policy objective on $\mathcal{Q}$.

Theorem 4 (Universal value and policy compatibility). *Let X, Z, d, μ, ν satisfy the finite-support and positive-reference conditions in (2), let $\tau > 0$, let $\mathcal{Q} \subseteq \Delta(Z)$ be nonempty, compact and convex, and set $\mathcal{P} = D\mathcal{Q}$. Both value functions compare on the same physical class $\mathcal{P} = D\mathcal{Q}$. If $D\mathcal{Q} \neq \Delta(X)$, they refer to this narrowed class; comparison with optimization over the full physical simplex additionally requires $D\mathcal{Q} = \Delta(X)$ (Corollary 2 provides a vertex test). With the two value functions defined above:*

- (1) $V_Z(c) = V_X(c)$ for every $c \in \mathbb{R}^{|X|}$ if and only if $\rho_{\mathcal{Q}} \equiv 0$, equivalently $L(\mathcal{P}) \subseteq \mathcal{Q}$.
- (2) The optimal physical policy is the same in the two problems for every c if and only if $\rho_{\mathcal{Q}}$ is constant on $\mathcal{P}$. If that constant is a , then $V_Z(c) = V_X(c) + \tau a$ for every c .

Proof. Extend $F(p) = \text{KL}(p \| \mu)$ by $+\infty$ outside $\mathcal{P}$, and put $G(p) = \min_{q \in \mathcal{Q}: Dq=p} \text{KL}(q \| \nu)$ with the same convention. Both are proper closed convex functions. Closedness of G follows from compactness of $\mathcal{Q}$ and continuity of KL with positive ν . For different feasible p_1, p_2 , their unique minimizing lifts differ. Strict convexity of KL and feasibility of their mixture show that G is strictly convex on its domain; F is also strictly convex. Equation (22) gives $G = F + \rho_{\mathcal{Q}}$ on $\mathcal{P}$.

The value functions are $-\tau F^*(-c/\tau)$ and $-\tau G^*(-c/\tau)$. Equality for every c implies equality of the conjugates and, by biconjugation of closed convex functions, $G = F$. The converse and the equivalence to the lift condition follow from the preceding proposition.

For the second claim, uniqueness of the physical minimizer and compactness allow the finite-dimensional envelope theorem to be applied to each problem. Differentiation with respect to c returns that problem's optimal physical policy. Agreement of the policies for every c thus makes the gradient of $V_Z - V_X$ zero everywhere. Since $\mathbb{R}^{|X|}$ is connected, this value difference is a constant, written τa . The conjugate representations above then imply $G = F + a$, or $\rho_{\mathcal{Q}} = a$. Conversely, a constant penalty in (21) shifts the objective by the same amount at every feasible policy and leaves its minimizer unchanged. $\square$

Remark 3 (Ties and fixed-cost equality). *The comparison is on $\mathcal{P} = D\mathcal{Q}$. Under the standing assumptions, $\tau > 0$ and positive references make the physical and induced encoded objectives strictly convex on their common physical domain, so each has a unique optimal physical policy even when the operational cost c has ties. Moreover, $V_Z(c) = V_X(c)$ for a single fixed cost already implies equality of these physical optimizers: if q^* is encoded-optimal and $p^* = Dq^*$, then*

$$V_X(c) \leq \langle c, p^* \rangle + \tau \text{KL}(p^* \| \mu) \leq V_Z(c) = V_X(c).$$

Thus both inequalities are equalities and uniqueness identifies p^* . Equal optimal values with different decoded optimum sets cannot occur under this theorem's assumptions. Such examples require changing the feasible physical class, dropping strict regularization, or otherwise leaving the stated comparison. The converse fails: a constant positive conditional penalty preserves every optimal physical policy while shifting all optimal values, as the harmless-penalty example below shows.

At $\tau = 0$, the regularizer vanishes and the theorem's strict-convexity argument no longer applies. With $c(0) = c(1) = 0$, every policy in $\Delta(\{0, 1\})$ has value zero. Narrowing the decoded class to $\{\delta_0\}$ retains that value but leaves only δ_0 optimal. This example changes the feasible physical class and illustrates the need to specify the comparison domain.

Example 2 (A harmless nonzero penalty). *Let each fibre contain two states with uniform K_x , and restrict $\mathcal{Q}$ to policies supported on one fixed representative per fibre. Every distribution in the physical simplex remains attainable, although $\rho_{\mathcal{Q}}(p) = \log 2$ for each p . The resulting value shift is $\tau \log 2$; the optimal physical policies are unchanged. A positive conditional penalty consequently does not, by itself, certify a change of optimum.*

Corollary 2 (A finite vertex test). *Let $\mathcal{Q} = \{q \in \Delta(Z) : Aq \leq b, Bq = e\}$. The lift of every physical distribution belongs to $\mathcal{Q}$ if and only if $L\delta_x$ satisfies those constraints for every $x \in X$.*

Proof. Necessity follows by choosing a simplex vertex. Sufficiency follows because $Lp = \sum_x p_x L\delta_x$, and $\mathcal{Q}$ is convex. In particular the condition also gives $D\mathcal{Q} = \Delta(X)$, since DL is the identity. $\square$

For a proper polytope $\mathcal{P}$, the same argument uses its vertices. For a nonconvex neural policy class, vertex feasibility alone is not sufficient. The tied-parameter example is not a convex class of joint distributions, so it is not being passed through this corollary.

Example 3 (Unequal fibres change the optimal policy). *Let $X = \{1, 2\}$, $\mu = (1/2, 1/2)$, and let the decoder fibres have sizes $m = (1, 2)$, with uniform conditional references in each fibre. Thus $Z = \{z_{11}, z_{21}, z_{22}\}$, $d(z_{ij}) = i$, and $\nu = (1/2, 1/4, 1/4)$ satisfies $D\nu = \mu$. Restrict $\mathcal{Q}$ to policies supported on the fixed representatives z_{11}, z_{21} , retaining this original reference rather than conditioning it on the restriction. Every physical policy remains attainable, so $D\mathcal{Q} = \Delta(X)$, but*

$$\rho_{\mathcal{Q}}(p) = p_2 \log 2.$$

With $c = 0$ and $\tau = 1$, the physical objective $\text{KL}(p||\mu)$ is minimized at $p_X^* = (1/2, 1/2)$. The encoded objective induces $\text{KL}(p||\mu) + p_2 \log 2$, whose minimizer is

$$p_Z^* = \frac{\mu/m}{\sum_i \mu_i/m_i} = (2/3, 1/3), \quad \text{TV}(p_Z^*, p_X^*) = \frac{1}{2}(|2/3 - 1/2| + |1/3 - 1/2|) = 1/6.$$

The optimal values are $V_X(0) = 0$ and $V_Z(0) = \log(4/3)$. The reference lift $L\delta_2$ splits its mass between z_{21} and z_{22} , so it is excluded from $\mathcal{Q}$. This is a failure of Corollary 2's lift condition, with a nonconstant penalty and different optimal values; it is not an equal-value counterexample to Theorem 4.

If a fixed alternative conditional H_x is enforced, then $\rho(p) = \sum_x p_x \text{KL}(H_x||K_x)$. This is an explicit physical cost shift. A canonical representative in a fibre with $m(x)$ uniform alternatives has penalty $\log m(x)$, which is constant only when the relevant multiplicities are equal.

3.4. Certificate Maintenance and Implementation

3.4.1. Later Constraints Require a New Check

Let $A \subseteq Z$ be an encoded restriction and $s_x = \sum_{z \in Z_x \cap A} K_x(z) > 0$. Global conditioning of the reference gives physical marginal proportional to $\mu_x s_x$. It preserves the reference physical law if and only if s_x is constant. Feasibility can therefore remain unchanged while probabilities change.

A fibrewise reference refresh is

$$\nu'_z = \mu_{d(z)} \frac{K_{d(z)}(z) \mathbf{1}_A(z)}{s_{d(z)}}. \quad (23)$$

It restores $D\nu' = \mu$ on the retained support. It does not automatically restore the residual-spectrum contract for a nontrivial $T - S \circ d$. Those spectra must be checked again, as must policy-class compatibility. If $s_x = 0$ for any physical decision, the physical support has changed. The previous certificate then no longer applies.

3.4.2. Exact Finite Checking

The checker performs exact rational arithmetic in Python. Input validation excludes floating-point certificate data, reference weights that are unnormalized or nonpositive, out-of-range decoder values, and incompatible feature dimensions. The checker then verifies reference transport, compares residual spectra, and tests whether the common spectrum is a point mass. For a linear policy class, each reference lift is tested against the given rational constraints. Composition is rejected when intermediate references or score features do not match.

For $N = |Z|$ explicit assignments and feature dimension k , residual-key construction uses $O(Nk)$ rational arithmetic operations. The Python reference validation checks reference transport by scanning the assignments for each physical state, adding $O(N|X|)$ work; the full current implementation therefore uses $O(N|X| + Nk)$ rational arithmetic operations, with expected linear dictionary access under ordinary hashing. Sorting an output spectrum or failure witness needs at most $O(N \log N)$ lexicographic key comparisons, each examining up to k rational components. Rational bit lengths affect all these costs.

Proposition 5 (Correctness of the explicit checker and product verifier). *For exact rational input satisfying (A1)–(A6), the explicit checker’s distribution flag is true if and only if the spectra in Theorem 1 agree. When they agree, its joint-KL flag is true if and only if the common spectrum is a point mass. If spectra differ, the returned physical pair, residual and unequal conditional weights witness failure of the universal distribution contract. If the common spectrum has several atoms, the returned spectrum itself witnesses failure of the universal joint-KL contract; two distinct positive-weight atoms suffice.*

For the bounded source-bound product verifier, valid structure and passing local spectra imply the global distribution contract under the decoded-only feasibility hypotheses of Theorem 5, with nonemptiness established separately. Its joint-KL pass additionally requires each common local spectrum to be a point mass. A local-spectrum mismatch is inconclusive for arbitrary globally restricted supports. Invalid structure or incompatible declarations reject the certificate and make no assertion about an unspecified consumer’s behaviour.

Proof. The explicit algorithm visits each declared assignment and accumulates $\nu_z/\mu_d(z)$ under the exact key $T(z) - S(d(z))$. Positive weights prevent cancellation of atoms. Its dictionary is therefore exactly σ_x ; dictionary equality is condition (iii) of Theorem 1. A differing entry proves the returned mismatch. Equal dictionaries with one key give a point mass. If two keys $r \neq s$ occur with positive mass, the projection in direction $v = r - s$ has positive variance; Theorem 2 then rules out the universal joint-KL identity, including its Taylor criterion at zero. Projected features give the same argument for a declared linear restriction.

The product verifier reconstructs the permitted local supports and checks their rational weights, residuals and source-derived linear declarations. Under decoded-only feasibility, its passing tables satisfy Theorem 5, which proves soundness. An offending local coordinate value can be excluded by global physical constraints, so a failed sufficient local test is not a necessary global failure. Structural rejection certifies only that the supplied object has failed verification. These are algorithmic correctness arguments; they do not assert proof-assistant verification of the Python implementation. $\square$

Support completeness is an input to the explicit checker. A separately written enumerator checks this interface on small cases. The decoder-substitution routine is shared by compiler and verifier and remains within the trusted software boundary. Companion Table 2 identifies the validation groups.

3.4.3. A Bounded-Integer Compiler with Product Certificates

The supported source comprises a bounded integer vector $x \in \prod_{i=1}^n \{0, \dots, U_i\}$, rational linear constraints, and a rational linear physical objective. A coordinate may be represented natively, as a split sum, by standard binary bits, by a unary bit vector, or by a slack pair. Optional auxiliary bits contribute to the encoded policy score while leaving the physical objective unchanged. The precise local rules appear in Table 3, exposing both the optimization rewrite and the policy architecture to inspection.

Table 3. Implemented coordinate rules before optional auxiliary bits. Every encoding has the stated bounds.

Rule	Local Encoded Variables	Decoder and Local Guard	Fibre Size
Native	$z \in \{0, \dots, U\}$	$x = z$	1
Split	$y, z \in \{0, \dots, U\}$	$x = y + z \leq U$	$x + 1$
Binary	$b_j \in \{0, 1\}$	$x = \sum_{j=0}^{w-1} 2^j b_j \leq U$	1
Unary	$b_1, \dots, b_U \in \{0, 1\}$	$x = \sum_j b_j$	$\binom{U}{x}$
Slack	$z, s \in \{0, \dots, U\}$	$x = z, z + s = U$	1

Binary width is $w = \max(1, \lceil \log_2(U + 1) \rceil)$. At $U = 0$, the unary representation has one empty encoding. All original linear constraints and costs are pulled back through the emitted decoder matrix.

Compiler output includes bounded integer variables, decoder and score matrices, transformed linear constraints, the pulled-back objective, and rational local conditional tables. With k_i optional bits, the default table gives each local representation mass $1/[m_i(x_i)2^{k_i}]$. The score declarations are $T_i = x_i + \sum_j u_{ij}$ and $S_i = x_i$. The physical reference μ can be any positive law on the constrained support; coordinatewise factorization is not required.

Theorem 5 (Product-certificate soundness). *Let X be a nonempty finite subset of the physical Cartesian bounds. Assume decoded-only feasibility: for each $x \in X$, the encoded support contains the complete Cartesian product of its declared local representations, with no additional encoded-only restrictions. Nonemptiness of X must be*

established separately. The following local test is sufficient, not a complete test for arbitrary global encodings. Let

$$\nu(z) = \mu(dz) \prod_i K_{i,x_i}(z_i), \quad T(z) - S(dz) = \sum_i R_i(z_i), \quad (24)$$

where each K_{i,x_i} is normalized. If the law of R_i under K_{i,x_i} is a common finite measure σ_i independent of x_i , then the global residual spectrum is $\sigma_1 * \dots * \sigma_n$ on every physical fibre. The global distribution contract holds, and the parameterwise joint-KL contract holds if and only if all σ_i are point masses.

Proof. Fix a feasible physical vector x . By decoded-only feasibility, its complete representations range over the declared Cartesian product. The normalized conditional factors in (24) sum to one, so their total reference mass is precisely $\mu(x)$. Under this conditional product, the residual components are independent and have the fixed laws σ_i . Their sum consequently has the same convolution law on every physical fibre. Apply Theorems 1 and 2 to obtain the two contract tests. For independent finite vectors, covariance matrices add; their positive semidefiniteness means the sum is zero exactly when each component has zero covariance. This proves the point-mass equivalence. Constraints coupling the physical coordinates select feasible values of x but do not change the conditional product for a fixed feasible value. $\square$

Without scored auxiliaries, all five default rules have local spectrum δ_0 . Adding k_i scored bits gives a binomial distribution on residual coordinate i . Consequently, encodings of the same source can pass both distribution and KL checks or pass the distribution check alone. The verifier records which contract passes.

The source-bound verifier checks the certificate against an expected source specification. It reconstructs every permitted local encoding set, verifies weights and residuals exactly, and derives the emitted linear model again. Structural equality includes type equality: numerical equality in Python does not permit a Boolean or floating-point value to replace an integer model field. Changes to an objective or matrix, an omitted encoding, or a stale source hash invalidate the certificate. A varying local spectrum instead produces an *inconclusive* result. Global coupling may exclude the offending coordinate value, so failure of this sufficient local test is not necessarily a universal counterexample. Nonemptiness of the physical support requires separate verification. The checker explicitly reports that global feasibility has not been solved; for the benchmark instances, a constant feasible schedule can be checked directly.

The implementation imposes explicit bounds on the source fragment: $0 \leq U_i \leq 16$ and $0 \leq k_i \leq 4$. The table size before auxiliary bits is $U_i + 1$ for native, binary, and slack rules, $(U_i + 1)(U_i + 2)/2$ for split, and 2^{U_i} for unary. It is multiplied by 2^{k_i} afterwards. Verification does not enumerate the Cartesian product over coordinates. Nevertheless, unary tables grow exponentially with local capacity, and dense emitted matrices may require quadratic storage in the number of coordinates and constraints. These bounds do not imply polynomial complexity in the size of an unrestricted binary-encoded input.

The reference runtime enumerates physical decisions only. It evaluates the physical log-linear law, computes the certified auxiliary penalty and gradient, and samples local encoded conditionals. Scheduling instances also have a separate dynamic-programming backend. The emitted linear constraints do not compel a solver to sample with certified weights; a stochastic consumer must use the reference tables explicitly. Integration with CVXPY, MathOptInterface, a commercial solver, or a general-purpose LLM frontend is not claimed.

3.4.4. Approximate Physical-Law Errors

For an approximate normalization with log distortion h_x , the decoded law has the form $q_x \propto p_x e^{h_x}$. The following bound isolates this representation error from optimizer or sampler error.

Proposition 6 (Sharp oscillation bound). *If p has positive finite support and $q_x \propto p_x e^{h_x}$, then*

$$\text{TV}(p, q) \leq \tanh\left(\frac{\text{osc } h}{4}\right). \quad (25)$$

The bound is attained by suitable two-point distributions. If an implemented sampler differs by at most η in TV from its intended encoded law, add η to the bound, capped at one.

Proof. Let $L_x = q_x/p_x$, $a = \min_x L_x$, and $b = \max_x L_x$. Then $\mathbb{E}_p L = 1$ and $b/a \leq r = e^{\text{osc } h}$. For $a < 1 < b$, convexity of $(L - 1)_+$ and the chord between a and b give $\text{TV}(p, q) = \mathbb{E}_p (L - 1)_+ \leq (1 - a)(b - 1)/(b - a)$. Maximizing this expression under $b/a \leq r$ gives $b = ra$ and $a = r^{-1/2}$, with value $(\sqrt{r} - 1)/(\sqrt{r} + 1)$. At a degenerate endpoint the likelihood ratio is constant and the TV is zero. To attain the nondegenerate bound, put the

likelihood ratios $r^{-1/2}, r^{1/2}$ on two points and choose their probabilities to make the mean ratio one. Finally, decoding cannot increase TV. The triangle inequality therefore combines the representation bound with the sampler error. $\square$

This supporting change-of-measure calculation requires a uniform guarantee. A log-normalizer error bounded everywhere by ε gives $\text{osc } h \leq 2\varepsilon$; an average error bound is insufficient. Similarly, a uniform oscillation bound $\text{osc } \rho \leq \varepsilon$ in the full-simplex class setting gives $\text{KL}(p_Q \| p^*) \leq \varepsilon$ by comparing the induced objective at p_Q and p^* . Measured runtime and justified global optimization gaps remain separate requirements; these bounds cannot substitute for either.

3.5. Independent Finite-Support Comparison of Certificates and Direct Checks

A separately written JavaScript reference implementation compares exact residual-spectrum checking with direct numerical evaluation of the physical and encoded distributions. It does not import or time the project's Python rational checker. In twelve passing instances, $n = |X| \in \{16, 64, 256, 1024\}$ and each fibre has $M \in \{2, 8, 32\}$ distinct residuals, giving $|Z| = nM$. For $x = 0, \dots, n-1$, the physical reference weights are $b_x = 1 + (x \bmod 7)$ and $S(x) = (x \bmod 17) - 8$. Residual $r_j = j - \lfloor M/2 \rfloor$ has positive weight $w_j = 1 + (j \bmod 5)$, $j = 0, \dots, M-1$, independently of the fibre. The encoded reference weights are $b_x w_j$ and $T(x, j) = S(x) + r_j$. Both references are normalized. The exact path uses integer-key dictionaries, BigInt weight accumulation and cross-products to check reference transport and equality of conditional residual masses. It rejects empty supports, nonpositive weights, missing fibres, and unsafe integer features or computed residuals. The generator additionally rejects unsafe encoded-feature sums and numeric weight products; sixteen named rejection regressions check these boundaries. It scans every fibre without early rejection. The numerical path separately normalizes the two exponential families and aggregates the encoded law through the decoder.

The direct checks use either $\theta = 0$ alone or $G \in \{9, 33\}$ equally spaced parameters in $[-0.5, 0.5]$. A grid passes numerically when its maximum TV is at most 10^{-10} . The revised timings were measured on 8 October 2026 in Node.js 24.16.0 (V8 13.6.233.17-node.49), Windows x64 build 10.0.26200, on an Intel Xeon Gold 6430 host reporting 64 logical processors. The monotonic `performance.now()` clock records milliseconds at sub-millisecond resolution. Three warm-up calls precede doubling of batch repetitions until a batch reaches 25 ms or 65,536 repetitions, followed by seven measured batches. The reported value is median batch duration divided by repetitions, rounded to two significant figures. Input generation and row projection are excluded; per-call dictionaries, typed arrays, result objects and arithmetic are included. Objective timing consumes both value and gradient. Execution order is fixed. The separate archive retains the unchanged check algorithms, timer adaptation, runtime record and all raw batches; S1 describes their use. The earlier millisecond-clock measurements remain archived.

Five additional controls examine detectability: a common point mass at residual 3; a common five-atom spectrum; and three unequal pairs of positive rational spectra. Two unequal pairs agree in raw moments through degree 3 and 7, respectively. The third agrees through degree 3 and is constructed to evade the selected grid $\{0, \log 2, \log 3\}$. All controls have compatible references and integer features. Separately, nine of the passing models, with $n \in \{16, 64, 256\}$ and $M \in \{2, 8, 32\}$, evaluate corrected objective values and scalar gradients at five parameters $\{-0.3, -0.1, 0, 0.1, 0.3\}$, using $\tau = 1.7$ and $c(x) = 0.15S(x)^2 + 0.3S(x)$. These 45 evaluations test the correction identity without running an optimizer.

3.6. Independent Optimizer and Initialization Checks

A separate JavaScript reference implementation evaluates the tied-bit scheduling objective using explicit physical-support sums and an analytic covariance gradient. It neither imports the Python implementation nor reproduces its random-number stream. The principal design fixes $T = 6, U = 4, D = 12, R = 2, x_0 = 2$ and compares full-memory BFGS with Polak–Ribière+ nonlinear conjugate gradient. Five seeded cost vectors, five coefficients $\tau \in \{0.25, 0.5, 1, 2, 4\}$, three auxiliary counts $k \in \{0, 1, 4\}$, and three starts give 450 optimizer executions. The starts are zero and two seeded Gaussian parameter vectors with standard deviation 0.35. Both algorithms use the same analytic gradient and an Armijo line search, with initial step one, sufficient-decrease coefficient 10^{-4} , and at most 45 step halvings. Conjugate gradient restarts its direction every T iterations. The target gradient infinity norm is 10^{-7} ; the initial budget is 1200 iterations. Every primary run missing that target receives a recorded restart from its endpoint with a further budget of 10,000 iterations, resetting the optimizer's internal state. No initial miss or final endpoint is discarded.

Costs use Mulberry32 pseudorandom uniforms and a Box–Muller transform, followed by centering of Gaussian coefficients with standard deviation 0.65. The cost seed is $272900 + 100T + s$; the nonzero-start seed is $273100 + 1000T + 100s + i$, where s is the cost index and $i = 1, 2$ is the start index. A further cross-scale check

uses $T = 4, 6, 8$, three cost vectors per horizon, $\tau = 1$, $k = 0, 1, 2, 4, 8$, zero starts, and BFGS with the original 1,200-iteration budget. Nine of its 45 cells reuse primary endpoints; the other 36 require new executions. Costs change with the horizon. The archived optimizer outputs use IEEE-754 double precision and were generated on 8 October 2026 by standalone JavaScript execution whose engine build and processor were not recorded. The source, parameter vectors, statuses and outputs are retained. Subsequent endpoint audits in the documented Node environment above validate stored calculations but do not identify the original execution environment.

4. Results

4.1. Protocol and Independence of Checks

The evaluation uses constructed integer production schedules, with every row traceable to the specified finite design. No observations are missing. There is no multiple null-hypothesis-testing procedure. Exact certificate checks use Python integer and rational arithmetic. Numerical error maxima are rounded upward to the displayed precision and are therefore upper bounds. Over a horizon of T periods, the physical decisions satisfy

$$x_t \in \{0, \dots, U\}, \quad \sum_{t=1}^T x_t = D, \quad |x_t - x_{t-1}| \leq R, \quad (26)$$

where x_0 is fixed. The cost is $c(x) = \sum_t a_t x_t + 0.08x_t^2$. All quantities use dimensionless design units and have not been calibrated to factory costs.

For the bookkeeping rewrite $x_t = y_t + z_t$, the components are nonnegative integers and the multiplicity is $m(x) = \prod_t (x_t + 1)$. Decoding preserves both (26) and $c(x)$. The comparison uses the following three consumers:

$$p(x) \propto e^{-\beta c(x)}, \quad (27)$$

$$q_+(x) \propto m(x) e^{-\beta c(x)}, \quad (28)$$

$$q_-(x) \propto m(x)^{-1} e^{-\beta c(x)}. \quad (29)$$

The normalized lift implements the first law, and uniform weighting of encoded assignments gives the second. For the third, the learner is confined to a canonical representative while the original inverse-multiplicity reference is retained. Rebasing that reference correctly would recover p in a unique-representation implementation. This case supplies an analytic negative control; we do not treat it as an inferior competitor.

For small instances, we enumerate physical schedules to check the results. Independently implemented forward-backward dynamic programming supplies partitions and one-period marginals. For two still smaller support designs, complete (y, z) assignments are also enumerated independently; this checks the multiplicity aggregation rather than only comparing two uses of the same formula. Complete joint TV is reported only where the physical joint is enumerated. Large-horizon KL and expected costs are computed from exact factor identities and dynamic-programming marginals.

The archived Python calculations use Python 3.13.5, NumPy 2.3.5, SciPy 1.17.0, PyTorch 2.10.0+cpu and Matplotlib 3.10.8 on Linux x86_64. The shared host reported an AMD EPYC 9V74 processor; float64 and one PyTorch thread were used. The separate archive retains seeds and raw rows. The median-interval calculation added during revision uses the archived CSV with NumPy 2.4.0; its resampling method and seed are specified below.

4.2. Representation-Induced Physical-Policy Distortion

The small study uses $T \in \{4, 6, 8\}$, $U = 4$, $D = 2T$, $R = 2$, and $x_0 = 2$. Five price vectors per horizon are generated uniformly on $[0.3, 2.0]^T$ with specified seeds. Each is evaluated at five inverse temperatures 0.25, 0.5, 1, 2, 4. Thus there are 15 underlying scheduling/cost instances and 75 instance-temperature cases, with 225 method evaluations. The cases are matched; they are not 225 independent experiments.

Table 4 and Figure 2 show errors caused by representation alone. The representation-distortion TV in Table 4 is not interchangeable with the learning-distortion TV in Section 4.6: the former compares weighting conventions at fixed costs, whereas the latter compares policies after optimizing different regularized objectives. The normalized formula has zero discrepancy by algebraic identity. Independent enumeration of complete assignments gives a maximum probability discrepancy of 5.00×10^{-16} across 18 checks. The independently implemented dynamic program differs from joint enumeration by at most 1.42×10^{-14} in partitions or marginals.

Table 4. Joint physical-distribution discrepancy in 75 matched instance–temperature cases. KL is $KL(q||p)$; TV is dimensionless.

Consumer	Median TV	Maximum TV	Maximum KL
Uniform encoded weighting	0.10848	0.13059	0.05280
Normalized lift	0	0	0
Canonical, reference not rebased	0.11193	0.13710	0.06169

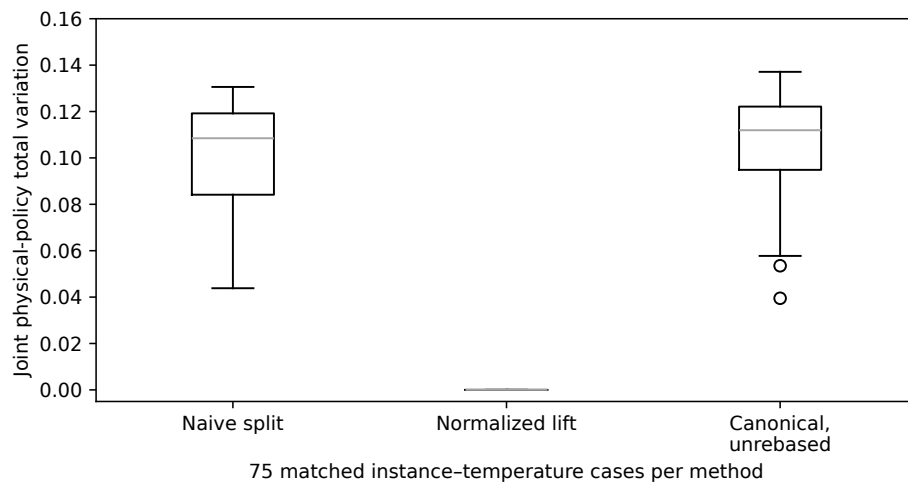


Figure 2. Physical-policy TV for the three consumers. Each box displays the median and interquartile range. Whiskers reach the most extreme observation within 1.5 interquartile ranges; observations beyond them are shown individually. The 75 observations are matched designed cases rather than a field sample.

4.3. Structured Scaling without Enumerating All Schedules

For $T \in \{8, 16, 32, 64\}$, set $U = 8$, $D = 4T$, $R = 2$, $x_0 = 4$, and $\beta = 1$. Each horizon receives a separately generated seeded price vector. Dynamic-programming states (t, s, u) record time, accumulated output, and previous production. The recurrence combines admissible ramp transitions with local weights. Normalization requires only the local correction $\log(x_t + 1)$. This implementation uses $O(TDU(2R + 1))$ arithmetic operations and stores $O(TDU)$ forward/backward states.

Table 5 reports the structured-inference counts and timings. The dynamic program uses additive costs and ramp constraints. Both the support and prices vary across the four horizons. Figure 3 shows KL for both distorted consumers under those joint changes.

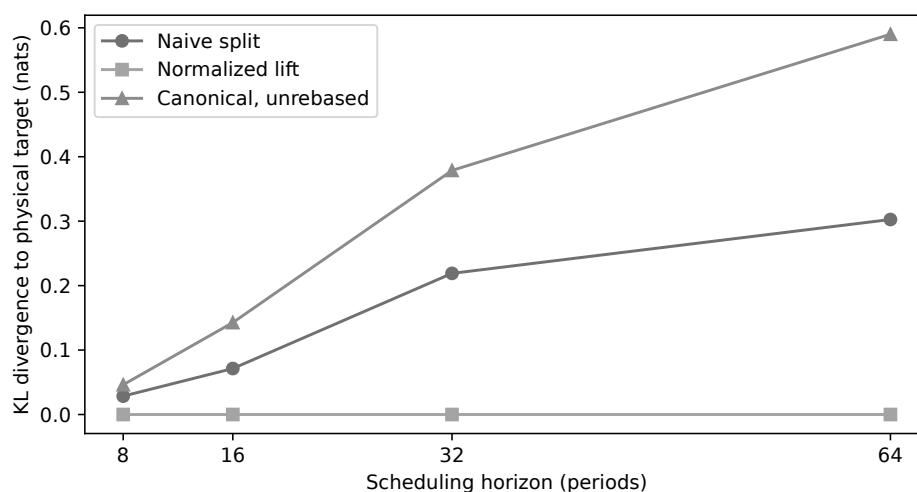


Figure 3. KL divergence on four structured horizon cases. No full-joint TV is inferred from one-period marginals on these large supports.

Table 5. Scaling of structured inference subject to an initial ramp constraint. Counts are rounded here; exact integers are archived. Runtime is the median of three normalized-lift DP calls.

T	Physical Schedules	Encodings	KL, Naive	DP Seconds
8	7383	2.11×10^9	0.02851	0.01402
16	9.14×10^8	5.65×10^{19}	0.07134	0.03116
32	2.30×10^{19}	6.29×10^{40}	0.21899	0.06815
64	2.11×10^{40}	1.13×10^{83}	0.30259	0.17758

4.4. Compiling and Checking Multiple Representations

The bounded frontend is evaluated on all five rules in 15 small cases, including scored auxiliary bits. The full finite checker independently expands each small encoding and agrees with the local certificate. Across 120 sampled natural parameters, explicit encoded probabilities decode to the physical law within 3.34×10^{-16} . A second test exhaustively traverses 2523 raw integer assignments across 50 heterogeneous-rule cases; all emitted constraints and physical costs agree with the original specification, with 252 feasible encodings covering the expected physical support.

For the larger certificate study, $U = 8$, demand is $4T$, and adjacent ramp limits are two, with $T = 8, 16, 32, 64$. Unlike the preceding DP study, this source has no initial-ramp constraint. Its counts must therefore not be equated with Table 5. The constant schedule $x_t = 4$ proves nonemptiness. Table 6 reports the 64-period results for four transformed representations. An independent integer-state dynamic program counts approximately 3.018×10^{40} physical schedules; the certificate examines local tables rather than enumerating that global support.

Table 6. Certificates for the 64-period bounded-source design, comprising approximately 3.018×10^{40} physical schedules without an initial ramp constraint. Every case passes both contracts without global schedule enumeration.

Rule	Local Entries	JSON kB	Compile s	Check s
Split	2880	344.2	0.01494	0.04843
Binary	576	327.4	0.01630	0.02891
Unary	16384	1746.9	0.06071	0.17870
Slack	576	220.2	0.01278	0.03073

A kilobyte is 1000 bytes. Reported JSON sizes include the transformed linear matrices. Times are medians from three local executions, excluding solving and sampling.

Figure 4 shows how verification time varies with the number of coordinates. The timings cover compilation and checking; each median summarizes three local repeats. Certificate sizes use compact JSON. Readable archived examples occupy more space. Expanded validation rejects 11 malformed-table/source cases and six further matrix/source-binding mutations. When a valid example fails only the local sufficient condition, the result is inconclusive. Runtime validation also covers 80 complete-encoding probability/KL cases and 800 sampled decodings. The largest joint-KL discrepancy is 8.89×10^{-16} .

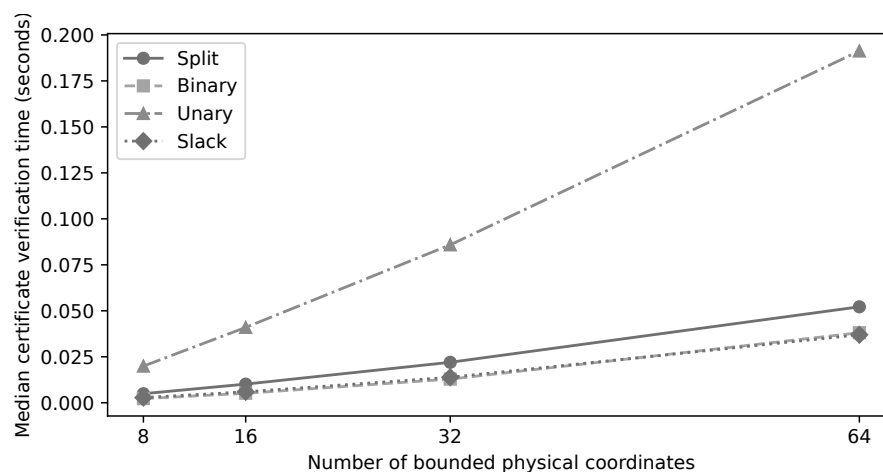


Figure 4. Certificate-verification times from local execution, including re-derivation of the linear matrices. Capacity is fixed at eight. Points report medians of three repeats, with each rule distinguished by its marker and line style.

4.5. Contextual Learning through a Physical Decision Layer

Use the six-period support $U = 4, D = 12, R = 2, x_0 = 2$. The experiment uses five seeds. Each seed determines a centered 3×6 coefficient matrix and a set of 192 Gaussian context vectors, each with three components. Target-policy scores are

$$-a^\top Wx - 0.08 \sum_t x_t^2, \quad (30)$$

with context a . Construction makes the full target distribution available, so the task is distribution matching without industrial labels. Of the contexts, 128 are used for training and the remaining 64 for testing. The optimizer receives no teacher coefficient matrix.

The four implementations use physical scores, physical scores plus $\log m(x)$, normalized-lift scores, and physical scores minus $\log m(x)$. All are fitted by full-distribution cross entropy with deterministic L-BFGS, sharing the same initialization within each seed. The two invariant implementations have the same mathematical objective and consequently the same parameter path under these deterministic calls. Their agreement is the expected negative-control result and supplies no improvement over the physical implementation.

Table 7 summarizes the held-out results. The declared linear contextual scores cannot generally absorb the nonlinear, schedule-dependent term $\log m(x)$. The observed errors therefore concern this hypothesis class.

Table 7. Mean results over five contextual-learning seeds, each evaluated on 64 held-out contexts. KL denotes $\text{KL}(p_{\text{target}} \| q)$.

Implementation	Mean Held-Out KL	Mean Held-Out TV
Physical implementation	4.25×10^{-14}	9.56×10^{-8}
Uniform encoded weighting	0.02258	0.08441
Normalized lift	4.25×10^{-14}	9.56×10^{-8}
Canonical, reference not rebased	0.02521	0.09108

4.6. Inference-Preserving Encodings that Change Joint-KL Learning

Table 8 and Figure 5 distinguish inference preservation from learning preservation. At each fixed parameter, the encoding preserves the physical law. Its uncorrected complete-policy regularizer nevertheless changes the physical policy obtained by numerical optimization. Corrected and untied controls distinguish this effect from optimization error.

Table 8. Post-training TV relative to the intended physical optimum at $\tau = 1$. At a fixed physical parameter, each of the five encoded output laws preserves physical inference.

Implementation	Median TV	Mean TV	95% Interval Mean	95% Interval Median
Physical objective	2.23×10^{-8}	2.42×10^{-8}	$[1.90, 2.97] \times 10^{-8}$	$[1.59, 3.08] \times 10^{-8}$
Joint KL, 1 tied bit	0.10982	0.10518	[0.09363, 0.11583]	[0.09704, 0.11361]
Joint KL, 4 tied bits	0.24468	0.24145	[0.21831, 0.26367]	[0.22350, 0.25406]
Corrected joint KL, 4 bits	2.23×10^{-8}	2.42×10^{-8}	$[1.89, 2.97] \times 10^{-8}$	$[1.59, 3.08] \times 10^{-8}$
Joint KL, 4 untied bits	2.23×10^{-8}	2.42×10^{-8}	$[1.89, 2.98] \times 10^{-8}$	$[1.59, 3.08] \times 10^{-8}$

Approximate nominal 95% percentile bootstrap intervals use 10,000 resamples of the 20 cost vectors at $\tau = 1$, with seed 272221 and linear percentile interpolation. Mean and median use the same draws within each method. Exact limits are retained in JSON.

To examine Theorem 2, we retain the six-period support and use $p_\theta(x) \propto e^{\theta^\top x}$. Each period receives k independent auxiliary bits with Bernoulli success probability $\text{sigmoid}(\theta_t)$. At every θ , summing over the bits recovers p_θ exactly. The common residual spectrum therefore passes the distribution test.

The learning objective combines physical expected cost with joint KL. In this experiment the physical cost is purely linear, $c(x) = \sum_t a_t x_t$, with the centered Gaussian coefficients specified below. The quadratic term used in the representation-distortion and contextual designs is not included here. For $\tau = 1$, the auxiliary contribution is

$$k \sum_t \text{KL}(\text{Bern}(\text{sigmoid } \theta_t) \| \text{Bern}(1/2)). \quad (31)$$

Twenty Gaussian price vectors with standard deviation 0.65 are generated using seeds $272700 + s$, $s = 0, \dots, 19$, and centered to define the matched cost instances. For each $\tau \in \{0.5, 1, 2\}$, we compare five imple-

mentations: the physical objective, joint-KL with one tied bit, joint-KL with four tied bits, the certified corrected objective with four bits, and an untied four-bit architecture. The design yields 300 numerical optimization runs in 60 matched cost–temperature cases. Every run initializes the physical parameters at zero. The untied architecture additionally initializes its six auxiliary parameters at their reference value of zero; these impose no constraint on the physical parameters. Optimization uses PyTorch L-BFGS with strong-Wolfe line search, a maximum of 250 iterations, gradient tolerance 10^{-10} , and change tolerance 10^{-14} . This matched design varies costs, temperatures, and implementations, not random initializations or optimization algorithms. For the corrected objective, we apply the exact symbolic reduction of Corollary 1. The physical Gibbs optimum is available analytically. We make no global-optimality claim for the numerical tied-bit solutions and archive the stationarity residual from every run.

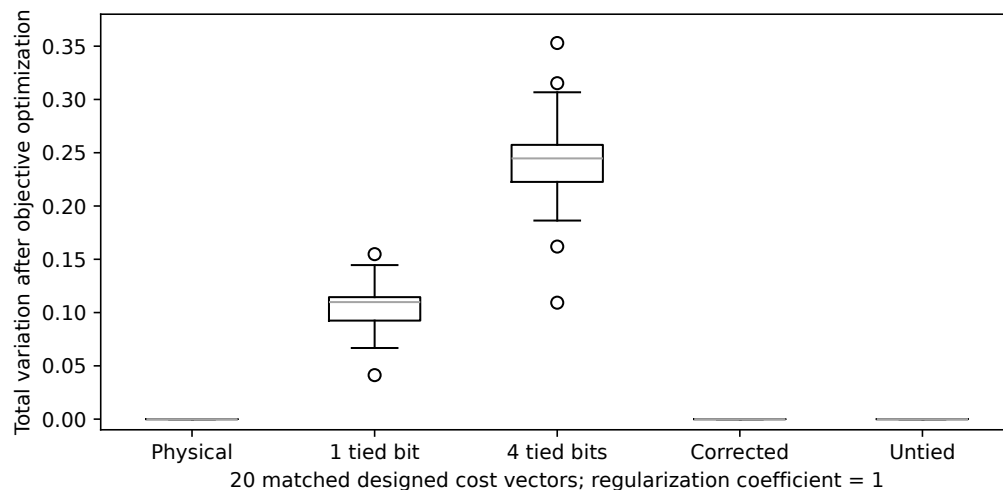


Figure 5. Joint-KL distortion with and without the certified correction at $\tau = 1$. All five implementations use the same 20 designed cost instances. Boxes and whiskers follow the conventions of Figure 2. The physical, corrected, and untied boxes lie close to zero and overlap the axis at the displayed scale.

At $\tau = 1$, four tied bits give mean TV 0.24145 (95% percentile bootstrap interval [0.21831, 0.26367]) and median TV 0.24468 (interval [0.22350, 0.25406]). The intervals use 10,000 resamples of the 20 cost vectors, seed 272221. Within each implementation, the same resamples give its mean and median intervals; percentiles use linear interpolation. The median interval is an approximate nominal 95% nonparametric bootstrap interval for the cost-generator design, with the cost vector as the resampling unit. Over all 60 cost–temperature pairs, the corrected and untied implementations’ TV-to-optimum errors differ from the physical run’s error by less than 1.71×10^{-16} .

Table 9 summarizes the archived 300 runs across all three regularization coefficients. At each coefficient, every one of the 20 cost vectors has larger TV with four tied bits than with one. The smallest paired increases are 0.11639, 0.06798 and 0.03563 at $\tau = 0.5, 1, 2$, respectively.

Table 9. Descriptive reanalysis of saved tied-bit learning runs. Each row contains 20 matched cost vectors; TV is measured against the analytic physical Gibbs optimum. No additional optimizer execution is included.

τ	Bits/Period	Mean TV	Median TV	Interquartile Range	Full Range
0.5	1	0.18300	0.18778	[0.16836, 0.19166]	[0.08257, 0.27613]
0.5	4	0.38705	0.39286	[0.38001, 0.41329]	[0.19896, 0.53368]
1	1	0.10518	0.10982	[0.09246, 0.11447]	[0.04129, 0.15483]
1	4	0.24145	0.24468	[0.22262, 0.25738]	[0.10928, 0.35297]
2	1	0.05436	0.05515	[0.04670, 0.06108]	[0.02056, 0.08382]
2	4	0.13200	0.13394	[0.11604, 0.14345]	[0.05619, 0.20206]

Quartiles use linear interpolation at index $(n - 1)p$ in the sorted observations. Temperatures reuse the same cost vectors and are not independent samples. Original bootstrap intervals at $\tau = 1$ are retained in Table 8; the ranges here are descriptive, not confidence intervals.

The largest archived gradient infinity norm over the 300 runs is 2.3238×10^{-7} , exceeding the configured 10^{-10} tolerance. Across the 60 cost–temperature pairs, the largest TV between the trained physical policy and its analytic Gibbs optimum is 8.7997×10^{-8} . The corrected and untied controls agree with the matched physical run’s

TV-to-optimum error to less than 1.71×10^{-16} . Objective equality follows from the symbolic correction; these scalar-error comparisons describe its numerical implementation.

When joint-KL preservation is not certified, apply Corollary 1's correction in Section 3.2.3 if the distribution certificate passes and the required decoder and reference data are available; an inconclusive check alone does not prove inequivalence.

4.7. Optimizer, Regularization, and Cross-Scale Sensitivity

The independent reference study checks the tied-bit effect across the declared algorithms and starts. All 225 primary BFGS executions attain the requested 10^{-7} gradient threshold within 35 iterations. Of the 225 primary conjugate-gradient executions, 210 attain it within the first 1,200 iterations. Fifteen receive the specified restart continuation, after which twelve attain it and three remain above threshold at a total of 11,200 iterations. Their final gradient infinity norms are 1.0327×10^{-7} , 1.3996×10^{-7} , and 2.0063×10^{-7} . Thus 447 of 450 primary endpoints meet the target; the three misses are retained.

Across the six algorithm–start endpoints in each of the 75 cost–temperature–auxiliary cells, the largest pairwise physical-policy TV is 1.0916×10^{-6} and the largest objective-value range is 1.9603×10^{-12} . These comparisons include the three tolerance misses. For the physical-only controls, the largest TV to the analytic Gibbs optimum is 1.0058×10^{-6} . A central-difference check of the implemented gradient gives maximum absolute error 2.10×10^{-10} on the recorded validation points. The cross-start endpoints are numerically consistent in this finite design; their unequal iteration requirements and stopping failures remain part of the result.

Table 10 extends the coefficient range beyond the archived Python design, using five independently generated cost vectors. Four tied bits give greater TV than one in every cost-vector comparison.

Table 10. Independent reference study across five regularization coefficients. Each row summarizes the same five cost vectors, using zero-start BFGS endpoints. Ranges are descriptive; repeated algorithms and starts are not additional cost observations. TV is relative to the analytic physical optimum.

τ	Mean TV, $k = 1$	Range, $k = 1$	Mean TV, $k = 4$	Range, $k = 4$
0.25	0.24700	[0.22255, 0.27517]	0.49611	[0.45331, 0.55638]
0.5	0.18300	[0.14468, 0.23407]	0.38229	[0.30956, 0.48129]
1	0.10380	[0.07337, 0.14627]	0.23634	[0.17089, 0.32495]
2	0.05406	[0.03765, 0.07864]	0.12797	[0.08728, 0.18451]
4	0.02715	[0.01846, 0.03988]	0.06541	[0.04359, 0.09566]

The cross-scale supports contain 31, 387, and 5,231 schedules. For each of the nine cost–horizon instances, TV increases over the tested counts $k = 0, 1, 2, 4, 8$. Forty-four of the 45 cells attain the gradient target. The remaining eight-period, eight-bit cell stops at the 1,200-iteration budget with gradient infinity norm 1.2889×10^{-7} and TV 0.41983; its value is included in Table 11.

Table 11. Mean TV across three seeded cost vectors at each horizon in the reference cross-scale check, with $\tau = 1$ and zero-start BFGS. Costs change with the horizon, precluding an isolated size-effect interpretation.

T	$ X $	$k = 1$	$k = 2$	$k = 4$	$k = 8$
4	31	0.06374	0.10044	0.14503	0.19532
6	387	0.12178	0.19094	0.27299	0.35621
8	5,231	0.11998	0.19064	0.27728	0.36911

4.8. An Analytic Global Control with a Nonuniform Reference

We evaluate Proposition 2 on the same constrained six-period support using $\mu_x \propto \exp\{-0.03 \sum_t x_t^2\}$. The combination of twenty cost vectors, three temperatures, and $k \in \{0, 1, 4\}$ produces 180 cases with analytic solutions. At $\tau = 1$, the median TV relative to the intended physical optimum is 0, 0.26429, and 0.42594 for zero, one, and four replicas, respectively. Independently implemented simplex-logit optimization over 60 cases differs from the analytic global law by at most 5.91×10^{-6} TV. Global optimality follows from Proposition 2.

Table 12 summarizes the analytic results at $\tau = 1$. The replica experiment changes the auxiliary architecture and reference together. It supplies a constrained multivariate global solution that independently separates inference preservation from joint-objective preservation.

Table 12. Analytic global optima for constrained-replica policies at $\tau = 1$ under a nonuniform reference. Each row uses the same twenty cost vectors.

Replicas k	Median TV	Maximum TV	Effective KL Multiplier
0	0	0	1
1	0.26429	0.34111	2
4	0.42594	0.58058	5

4.9. What Direct Finite-Parameter Checks Detect and Cost

Every compatible-reference control agrees at $\theta = 0$, which is automatic from $D\nu = \mu$ and supplies no further evidence for the universal distribution contract. The point-mass control passes both certificates. The common five-atom control, with weights $(1, 2, 3, 4, 5)/15$ on residuals $(-2, -1, 0, 1, 2)$, passes the distribution certificate but fails the universal joint-KL certificate. Its maximum numerical TV on $\{-1, -0.5, 0, 0.5, 1\}$ is 2.31×10^{-16} , while its complete-minus-physical KL at $\theta = 0.5$ is 0.14108888.

Exact spectrum comparison rejects all three unequal-spectrum controls. All nevertheless pass the near-zero numerical grid $\{-10^{-3}, -5 \times 10^{-4}, 0, 5 \times 10^{-4}, 10^{-3}\}$ at tolerance 10^{-10} . The stronger, exact grid-miss control uses

$$\sigma_0 = \frac{6\delta_0 + 57\delta_2 + 32\delta_4 + \delta_6}{96}, \quad \sigma_1 = \frac{29\delta_1 + 58\delta_3 + 9\delta_5}{96}.$$

With $X = \{0, 1\}$, uniform physical reference, and $S(x) = x$, its transform difference is

$$H_0(\theta) - H_1(\theta) = \frac{(t-1)^4(t-2)(t-3)}{96}, \quad t = e^\theta.$$

Thus agreement at $\theta = 0$, $\log 2$, $\log 3$ and moment agreement through degree 3 are exact, despite a failed universal certificate. The measured maximum TV on that three-point grid is 2.78×10^{-17} , versus 0.00955185 on $\{-1, -0.5, 0, 0.5, 1\}$. Numerical agreement on a selected grid therefore cannot replace the declared exact spectral comparison.

Table 13 reports the $M = 32$ timing cells. Across the twelve timing cells, the reference-point check costs 0.068–0.25 times the exact certificate's median time; the nine- and 33-parameter grids cost 1.1–2.4 and 4.1–8.5 times that median. These ratios describe the complete check paths. Support construction, compilation, sampling and training are excluded. The exact path scans the encoded support once, while the direct path evaluates $G(|X| + |Z|)$ exponential weights.

Across the 45 correction evaluations, the largest absolute residuals in $J_Z - \tau g - J_X$ and its scalar derivative are 7.11×10^{-14} and 1.35×10^{-12} . Separate timing at $\theta = 0.15$ measures physical evaluation, encoded evaluation, common-spectrum penalty evaluation and explicit encoded-minus-penalty evaluation. The penalty costs 7.8–9.0%, 2.1–3.5% and 0.51–0.90% of the encoded value-and-gradient median for $n = 16, 64, 256$, respectively. Two combined-path medians are below their uncorrected counterparts, consistent with noise at small increments. The penalty evaluates M exponentials and two logarithms, or one logarithm with a cached reference normalizer. Symbolic reduction calls the physical evaluator directly.

Table 13. Remeasured JavaScript reference timings for $M = 32$ residual atoms per fibre, using Node.js 24.16.0 and `performance.now()` on the documented Xeon host. Median milliseconds per complete call over seven batches in one session, rounded to two significant figures. $G = 1$ checks $\theta = 0$. Every row passes the exact distribution certificate and numerical grids.

$ X $	$ Z $	Exact Spectrum	Direct $G = 1$	Direct $G = 9$	Direct $G = 33$
16	512	0.05	0.0041	0.064	0.24
64	2048	0.21	0.016	0.29	1.1
256	8192	0.98	0.071	1.2	4.2
1024	32768	4.5	0.32	5.1	19

4.10. Verification Coverage

The primary suite passes 8532 assertions covering exact reference transport, spectrum separation, mask refresh, linear lifts, randomized compositions, numerical KL identities, and scalar optimizer comparisons. Further tests

independently enumerate 819 distinct complete assignments over two support designs and 18 cost/temperature checks. The additional checks comprise 200 calculations of the covariance gradient and 200 feature-aware layer compositions. All eight malformed-input cases are rejected.

The maximum numerical KL-chain residual is 1.11×10^{-15} . Central finite-difference calculations check the covariance-gradient formula to 2.22×10^{-10} . Relative to (14), the scalar optimizer's probability discrepancy is 7.17×10^{-9} . Failures abort the scripts rather than silently writing a passing status. The release also tests the parameter-domain and consumer boundaries explicitly: linear-projection passes, a nonlinear-curve non-certificate, harmless constant penalties, product-test inconclusiveness, and strict rejection of numeric-type substitutions in emitted model fields. The separate archive retains exact counts and outputs.

5. Discussion and Limitations

A formulation can be equivalent for deterministic optimization while changing the semantics required by a stochastic consumer. The residual-spectrum test identifies the distribution obligation, and the joint-KL test identifies the additional obligation when complete encoded policies enter learning. A failed universal certificate calls for correction, rejection or further analysis of the declared consumer before use.

The negative controls explain when the extra checks matter. A correctly projected sampler removes naive multiplicity; losses depending only on the physical law are identical when that law is preserved. Within a compact convex class, a constant conditional penalty shifts values while preserving every optimal physical policy. Parameterized architectures require their own analysis, because their attainable policies and parameter coupling can differ.

The interpretation is subject to the following limits.

- (1) The experiments test specified mechanisms on designed scheduling costs and supports. Their bootstrap intervals summarize the cost-generator design; repeated temperatures, starts and algorithms reuse cases. They provide no estimate of prevalence in deployed systems and no general monotonicity theorem.
- (2) The tied-bit optimizer endpoints are numerical solutions. Configured tolerances are reported alongside actual stationarity residuals; four final threshold misses remain in the separate reference study. Global-optimality conclusions are confined to the proved scalar and replica controls.
- (3) The explicit-support characterization is complete for the declared finite log-linear family. A linear restriction uses projected features; arbitrary nonlinear parameter families, approximate stages and stochastic decoders need separate arguments. The stronger joint-KL contract is relevant when encoded variables enter the objective or restrict the policy class.
- (4) The product certificate is sufficient for the bounded rewrite fragment under decoded-only feasibility. It neither solves physical nonemptiness nor supplies a complete test for arbitrary global encoded restrictions. Physical interpretation, intended references and support completeness remain declared inputs. Unary local tables grow exponentially in capacity; construction can cost more than checking.
- (5) Correctness proofs accompany the algorithmic statements. The Python implementation is tested, but has no proof-assistant verification. Compiler and verifier share decoder-substitution code within the trusted boundary. The finite guarantees do not extend automatically to arbitrary neural losses, independently sampled gradients, stateful reinforcement learning or opaque samplers.
- (6) Timings are single-session measurements of the specified paths. Original Python timings use the recorded shared Linux host. Revised JavaScript timings use the documented Windows Node process; the earlier millisecond-clock timings and optimizer outputs came from an execution environment whose engine build and processor were not recorded. Endpoint checks do not recover that environment, and batch repeats are not independent operational samples.
- (7) WAPS, ProCount, Dice and FLARE supply different contracts and have not been compared as executables. A systems benchmark would require matched encodings and separate costs for compilation, checking, sampling and training. The isolated penalty timings provide no end-to-end training-speedup or negative-overhead measurement.

A practical representation review records the downstream consumer, verifies the relevant declarations and renews the certificate after changes. Extending these finite tests to richer continuous or stateful consumers remains an open problem.

6. Conclusions

Correctness of an optimization reformulation depends on its intended use. For finite log-linear decision systems, equality of feasible physical decisions and costs does not ensure equality of stochastic output laws. Nor does equality of those laws ensure equality of complete-policy learning objectives. Identical weighted residual

spectra characterize the distribution requirement; a point-mass common spectrum is the stronger joint-KL criterion. The composition and policy-class results specify how the obligations accumulate, when exact correction is available, and when a nonzero auxiliary penalty leaves the physical optimum unchanged.

Implementation begins with a declared physical decoder, reference measure and downstream consumer. The bounded frontend and reproduction software provide the corresponding checks for the stated finite fragment. The designed experiments show that the distinction can be numerically material; analytic and negative controls explain when correction is required. These checks sit between construction of an optimization model and its stochastic use, with the guarantees specified by the finite theorems and bounded compiler.

Supplementary Materials

The following supporting information can be downloaded at: https://media.sciltp.com/articles/others/2610191531246193/AIEng-26090251_S1.pdf. Supplementary Material S1 documents implementation traceability, validation boundaries, equation-derived figure checks and the reproduction procedure. Reproduction requires the separate code-and-results archive described in S1. The replication material is available from Zenodo at <https://doi.org/10.5281/zenodo.23253837>. It is not included with the manuscript files.

Author Contributions

C.S.W.: conceptualization, methodology, formal analysis, software, validation, investigation, data curation, visualization, writing—original draft preparation, and writing—review and editing. The author has read and agreed to the submitted version of the manuscript.

Funding

This research received no external funding.

Institutional Review Board Statement

Not applicable. The study uses finite mathematical models and constructed computational inputs only and does not involve humans or animals.

Informed Consent Statement

Not applicable. The study does not involve human participants or identifiable patient information.

Data Availability and Statistical Reporting

The constructed inputs, raw results, source specifications and software are retained in the separate code-and-results archive described in S1. The replication material is available from Zenodo at <https://doi.org/10.5281/zenodo.23253837>. It is not included with the manuscript files. All observations from the designed experiments are retained. There is no multiple null-hypothesis-testing procedure; bootstrap intervals describe the specified cost generator. Materials and Methods and S1 report the computational environments. S1 gives the archive contents, software requirements, literal command flags and the expected consumer-specific rejection of the uncorrected joint-KL example.

Conflict of Interest

The author declares no conflict of interest.

Use of AI-assisted technologies

During preparation of this work, AI tools assisted with literature search, mathematical exposition, calculation-code development, vector-figure preparation and submission materials. ChatGPT and OpenAI Codex were used during revision to assist with wording, source editing and computational checks. The author reviews and edits the AI-assisted content and takes full responsibility for the article.

References

1. Dudek, J.M.; Phan, V.H.N.; Vardi, M.Y. ProCount: Weighted Projected Model Counting with Graded Project-Join Trees. In *Theory and Applications of Satisfiability Testing—SAT 2021*; Springer International Publishing: Cham, Switzerland, 2021; Volume 12831, pp. 152–170. https://doi.org/10.1007/978-3-030-80223-3_11.
2. Gupta, R.; Sharma, S.; Roy, S.; et al. WAPS: Weighted and Projected Sampling. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2019)*; Vojnar, T., Zhang, L., Eds.; Springer International Publishing: Cham,

- Switzerland, 2019; Volume 11427, pp. 59–76. https://doi.org/10.1007/978-3-030-17462-0_4. Correction in 2019, p. C1. https://doi.org/10.1007/978-3-030-17462-0_29.
3. Chakraborty, S.; Fremont, D.; Meel, K.; et al. Distribution-Aware Sampling and Weighted Model Counting for SAT. In Proceedings of the AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; Volume 28, pp. 1722–1730. <https://doi.org/10.1609/aaai.v28i1.8990>.
4. Bryant, R.E.; Tan, Y.K.; Heule, M.J.H. Certifying Projected Knowledge Compilation. In Proceedings of the 28th International Conference on Theory and Applications of Satisfiability Testing (SAT 2025), Glasgow, UK, 12–15 August 2025; Volume 341, pp. 8:1–8:22. <https://doi.org/10.4230/lipics.sat.2025.8>.
5. Liberti, L. Reformulations in Mathematical Programming: Definitions and Systematics. *RAIRO-Oper. Res.* **2009**, *43*, 55–85. <https://doi.org/10.1051/ro/2009005>.
6. Bentkamp, A.; Fernández Mir, R.; Avigad, J. Verified Reductions for Optimization. In *Tools and Algorithms for the Construction and Analysis of Systems*; Springer Nature Switzerland: Cham, Switzerland, 2023; Volume 13994, pp. 74–92. https://doi.org/10.1007/978-3-031-30820-8_8.
7. Robbins, H.; Lawless, C.; Udell, M.; et al. FLARE: Verifying MILP Reformulations with LLM-Based Theorem Proving. *arXiv* **2026**, arXiv:2608.25220.
8. Agrawal, A.; Verschuere, R.; Diamond, S.; et al. A Rewriting System for Convex Optimization Problems. *J. Control. Decis.* **2018**, *5*, 42–60. <https://doi.org/10.1080/23307706.2017.1397554>.
9. Legat, B.; Dowson, O.; Dias Garcia, J.; et al. MathOptInterface: A Data Structure for Mathematical Optimization Problems. *INFORMS J. Comput.* **2022**, *34*, 672–689. <https://doi.org/10.1287/ijoc.2021.1067>.
10. Amos, B.; Kolter, J.Z. OptNet: Differentiable Optimization as a Layer in Neural Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; Volume 70, pp. 136–145. <http://proceedings.mlr.press/v70/amos17a/amos17a.pdf>.
11. Agrawal, A.; Amos, B.; Barratt, S.; et al. Differentiable Convex Optimization Layers. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Volume 32, pp. 9562–9574.
12. Vlastelica, M.; Paulus, A.; Musil, V.; et al. Differentiation of Blackbox Combinatorial Solvers. In Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020), Online, 26–30 April 2020.
13. Besançon, M.; Dias Garcia, J.; Legat, B.; et al. Flexible Differentiable Optimization via Model Transformations. *INFORMS J. Comput.* **2024**, *36*, 456–478. <https://doi.org/10.1287/ijoc.2022.0283>.
14. Mandi, J.; Kotary, J.; Berden, S.; et al. Decision-Focused Learning: Foundations, State of the Art, Benchmark and Future Opportunities. *J. Artif. Intell. Res.* **2024**, *80*, 1623–1701. <https://doi.org/10.1613/jair.1.15320>.
15. Ahmaditeshnizi, A.; Gao, W.; Udell, M. OptiMUS: Scalable Optimization Modeling with (MI)LP Solvers and Large Language Models. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 21–27 July 2024; Volume 235, pp. 577–596. <https://proceedings.mlr.press/v235/ahmaditeshnizi24a.html>.
16. Darwiche, A.; Marquis, P. A Knowledge Compilation Map. *J. Artif. Intell. Res.* **2002**, *17*, 229–264. <https://doi.org/10.1613/jair.989>.
17. Kimmig, A.; Van den Broeck, G.; De Raedt, L. Algebraic Model Counting. *J. Appl. Log.* **2017**, *22*, 46–62. <https://doi.org/10.1016/j.jal.2016.11.031>.
18. Maene, J.; De Raedt, L. The Gradient of Algebraic Model Counting. In Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence, Philadelphia, PA, USA, 25 February–4 March 2025; Volume 39, pp. 19367–19377. <https://doi.org/10.1609/aaai.v39i18.34132>.
19. Holtzen, S.; Van den Broeck, G.; Millstein, T. Scaling Exact Inference for Discrete Probabilistic Programs. *Proc. ACM Program. Lang.* **2020**, *4*, 1–31. <https://doi.org/10.1145/3428208>.
20. Eberl, M.; Hölzl, J.; Nipkow, T. A Verified Compiler for Probability Density Functions. In *Programming Languages and Systems*; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9032, pp. 80–104. https://doi.org/10.1007/978-3-662-46669-8_4.
21. Tassarotti, J.; Tristan, J.B. Verified Density Compilation for a Probabilistic Programming Language. *Proc. ACM Program. Lang.* **2023**, *7*, 615–637. <https://doi.org/10.1145/3591245>.
22. Staton, S.; Yang, H.; Wood, F.; et al. Semantics for Probabilistic Programming: Higher-Order Functions, Continuous Distributions, and Soft Constraints. In Proceedings of the LICS’16: 31st Annual ACM/IEEE Symposium on Logic in Computer Science, New York, NY, USA, 5–8 July 2016; pp. 525–534. <https://doi.org/10.1145/2933575.2935313>.
23. Valiant, L.G. The Complexity of Enumeration and Reliability Problems. *SIAM J. Comput.* **1979**, *8*, 410–421. <https://doi.org/10.1137/0208032>.
24. Fichte, J.K.; Hecher, M.; Morak, M.; et al. Solving Projected Model Counting by Utilizing Treewidth and Its Limits. *Artif. Intell.* **2023**, *314*, 103810. <https://doi.org/10.1016/j.artint.2022.103810>.
25. Ying, Y.; Zhai, Q.; Zhou, Y.; et al. An LSTM-Driven Efficient Solution Algorithm of Economic Dispatch with Integral Constraints for Energy-Intensive Enterprise Microgrids. *AI Eng.* **2025**, *1*, 5. <https://doi.org/10.53941/aieng.2025.100005>.
26. Rabbanian, S.; Wang, H.; Wu, L. Adaptive Multi-Agent Reinforcement Learning for Electric Vehicle On-Road Charging Optimization under Dynamic Traffic Conditions. *AI Eng.* **2026**, *2*, 2. <https://doi.org/10.53941/aieng.2026.100002>.

27. Zhang, B. NeSyWikiCompiler: A Neural-Symbolic Compiler for Verifiable AI Knowledge Engineering. *AI Eng.* **2026**, *2*, 11. <https://doi.org/10.53941/aieng.2026.100011>.
28. Kumar, S.; Bist, S.H. A Comprehensive AI Security Pipeline: Drift Detection, Adversarial Robustness and Automated ML Testing. *AI Eng.* **2026**, *2*, 13. <https://doi.org/10.53941/aieng.2026.100013>.
29. Yuan, X.; Chen, S.; Hu, B.; et al. Open-Set Semiconductor Defect Classification in SEM Images via Uncertainty-Aware Local Prototype Alignment. *AI Eng.* **2026**, *2*, 12. <https://doi.org/10.53941/aieng.2026.100012>.
30. Pandey, A. A Five-Layer Reference Architecture for First-Party Enterprise Knowledge Graphs with GraphRAG Integration and Governance Controls. *AI Eng.* **2026**, *2*, 14. <https://doi.org/10.53941/aieng.2026.100014>.
31. Geist, M.; Scherrer, B.; Pietquin, O. A Theory of Regularized Markov Decision Processes. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Volume 97, pp. 2160–2169. <https://proceedings.mlr.press/v97/geist19a.html>.
32. Peters, J.; Mülling, K.; Altun, Y. Relative Entropy Policy Search. In Proceedings of the AAAI Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; Volume 24, pp. 1607–1612. <https://doi.org/10.1609/aaai.v24i1.7727>.
33. Baram, N.; Tennenholtz, G.; Mannor, S. Action Redundancy in Reinforcement Learning. In Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, Online, 27–30 July 2021; Volume 161, pp. 376–385. <https://proceedings.mlr.press/v161/baram21a.html>.
34. Bengio, Y.; Lahlou, S.; Deleu, T.; et al. GFlowNet Foundations. *J. Mach. Learn. Res.* **2023**, *24*, 1–55. <https://jmlr.org/papers/v24/22-0364.html>.
35. Papandreou, G.; Yuille, A.L. Perturb-and-MAP Random Fields: Using Discrete Optimization to Learn and Sample from Energy Models. In Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 193–200. <https://doi.org/10.1109/iccv.2011.6126242>.
36. Hazan, T.; Jaakkola, T. On the Partition Function and Random Maximum a-Posteriori Perturbations. In Proceedings of the 29th International Conference on Machine Learning, Edinburgh, Scotland, 26 June–1 July 2012; pp. 1667–1674.
37. Paulus, M.B.; Choi, D.; Tarlow, D.; et al. Gradient Estimation with Stochastic Softmax Tricks. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Online, 6–12 December 2020; Volume 33, pp. 5691–5704.
38. Ranganath, R.; Tran, D.; Blei, D. Hierarchical Variational Models. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; Volume 48, pp. 324–333.
39. Nielsen, D.; Jaini, P.; Hoogeboom, E.; et al. SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Online, 6–12 December 2020; Volume 33, pp. 12685–12696.
40. Nielsen, F.; Garcia, V. Statistical Exponential Families: A Digest with Flash Cards. *arXiv* **2009**, arXiv:0911.4863v2.
41. Louis, T.A. Finding the Observed Information Matrix when Using the EM Algorithm. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **1982**, *44*, 226–233. <https://doi.org/10.1111/j.2517-6161.1982.tb01203.x>.