REFLECTIVE GRADIENTS: FEASIBILITY–TASK TRADEOFFS IN SOLVER DISTILLATION WITH VERIFIED ACTION RELEASE

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural policies can violate explicit action constraints, while repairing every proposal with an optimization solver adds inference cost. We study whether detached solver supervision reduces repair dependence when an independent verifier remains responsible for action release. Admissible actions are finite unions of bounded polyhedra. The verifier checks proposals, requests repair when needed, and rechecks the decoded serialized action under the current state and rules. Under explicit compilation, arithmetic, state consistency, and consumer assumptions, every released action satisfies the encoded constraints independently of solver status. Feasibility verification alone gives no general approximation guarantee for branch-candidate selection; we reproduce severely suboptimal targets from a numerical repair service. A locally frozen study uses 20 paired trials per setting across disconnected and convex numerical tasks and a synthetic resource allocation application. Exact teachers, task reweighting, branch models, and fresh direct task optimization expose distinct feasibility, utility, and computational tradeoffs. Replacing numerical corrections with exact targets changes those tradeoffs without demonstrating a consistent learning advantage over task reweighting. Regret curves reveal threshold-dependent rankings, and the allocation task exposes poor learned utility at the fixed training budget. We provide conditional correctness proofs, analytic counterexamples, and auditable computation records. The findings motivate specifying teacher quality and evaluating complete decision cost under explicit task requirements rather than treating fewer repairs as sufficient evidence of useful constraint internalization.

1 INTRODUCTION

Learning a constrained action policy creates two separate questions: whether the delivered action obeys its specification, and whether learning reduces the cost of producing a useful action. A runtime checker can answer the first under an explicit execution contract. It does not answer the second. In particular, an optimization service can return a verified feasible correction whose distance or task quality is poor, and imitating that correction can change both the policy's intervention rate and its task performance.

We study detached regression to solver corrections while an independent verifier remains responsible for action release. "Reflective Gradients" names this correction-learning study; it does not denote a new gradient-reflection operator. The setting is constrained numerical prediction with observable objectives and constraints, represented as finite unions of bounded polyhedra. This observability makes a direct task optimizer a necessary comparison: an agent should not learn to imitate repairs to an approximate task solution without checking whether solving the task itself is already competitive.

Auditing a prior teacher study reveals that exact checking can discard numerical branch candidates, leaving a distant feasible target. One reproduced correction is 2.515 units from its proposal although the exact minimum is 0.00314. Equivalent convex-set representations also change whether the service supplies a target. Nominal teacher norms can therefore conceal consequential differences in the supplied actions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Our contributions are a controlled diagnosis of teacher-quality failures and their learning consequences. We construct a bounded convex example showing why verified candidate selection lacks a uniform approximation-ratio guarantee. Deterministic exact teachers and controls distinguish numerical target quality, projection rule, task-label reweighting, and branch selection. A locally frozen protocol compares these controls with freshly computed direct optimization over 20 paired trials per task: disconnected 2D, its convex control, and 16-dimensional resource allocation with four operating modes. Task-quality curves, paired effects, coverage, and complete decision timings support the assessment.

2 Related work

Independent action enforcement predates this study. Shielding corrects actions against formal specifications (Alshiekh et al., 2018), while predictive safety filters use dynamics-aware conditions (Wabersich & Zeilinger, 2021). Our contract concerns membership in the current encoded action set, including the decoded value released to a consumer. A recent wireless-scheduling audit likewise distinguishes predictive accuracy, joint feasibility, and trust in reported information (Zafari et al., 2026).

Distilling symbolic supervision into neural parameters is established. Hu et al. (2016) distill logic rules, and NeSyPr compiles symbolic planning knowledge for efficient LM inference (Choi et al., 2025). DAgger addresses learner-induced sequential distributions (Ross et al., 2011), whereas our states are independent. PACT transfers physical-constraint guidance into diffusion policies using reverse-KL training and a curriculum (Wu et al., 2026). Its manipulation setting, objective, and supervision differ from detached squared-error correction imitation. These precedents motivate controlling teacher accuracy, projection norm, mode choice, and task supervision.

Constrained output architectures offer alternatives. HardNet studies affine and convex enforcement with explicit approximation assumptions (Min & Azizan, 2025). DisjunctiveNet uses lifted convexifications; its exact DNF result requires an optimal extreme-point solution of the extended formulation (Pal & Li, 2026). PAL constructs constrained continuous distributions by symbolic integration (Kurscheidt et al., 2025). DC3 differentiates completion and correction (Donti et al., 2021). OptNet and differentiable convex layers supply precedents for optimization within neural networks (Amos & Kolter, 2017; Agrawal et al., 2019). Our branch baseline selects a supervised action head, optionally projected into its box/budget mode. It is a simple architecture control, not a reproduction of these systems.

Multimodal regression is another established connection. Bishop (1994) explains the limitations of regression averages for multivalued mappings and models their conditional distribution. Our analytic compromise instantiates this familiar issue. Homeomorphic projection requires a set homeomorphic to a ball (Liang et al., 2023), so disconnected unions require additional mode handling. PCGrad modifies conflicting parameter gradients (Yu et al., 2020); our action-space and one-step parameter diagnostics do not implement that remedy. We assess correction learning and direct optimization under stated controls rather than claim a new shielding principle or general constrained-learning algorithm.

3 Encoded feasibility and the release contract

For state x and rule version v, the admissible set is

$$F_v(x) = \bigcup_{k=1}^{K} \{u \in \mathbb{R}^d : A_k(x)u \le b_k(x),\ l(x) \le u \le h(x)\}. \tag{1}$$

The representation contains finite rational coefficients, closed inequalities, units, and a binding to a validated state. The implemented compiler consumes explicit structured rules. It does not certify a natural-language interpretation, a sensor reading, or the adequacy of the rules as a physical model. The verifier checks every represented inequality exactly and accepts when the global box and one complete branch hold. A floating-point proposal denotes its exact represented rational value; no numerical tolerance changes membership.

The release path checks the proposal, requests repair if it is invalid, independently checks the returned candidate, serializes it, and checks the decoded wire value. Immediately before release it rejects stale state/rule bindings, reparses the payload, and checks that value under the current context. Failure returns no action. Solver status, a routing prediction, and a cached acceptance flag cannot authorize release.

Proposition 1 (conditional release soundness). Assume correct compilation of the intended encoded rules, correct parsing and rational membership checking, a state/rule context valid at consumption, and a consumer that uses exactly the checked value and units without unchecked transformation. Every non-null released action belongs to $F_v(x)$, independently of the policy, repair optimality, or optimizer status.

Proof. Non-null release requires final acceptance of the decoded action. Exact acceptance identifies a branch satisfying all its rows and the global bounds. The compilation, context, and consumer assumptions identify that set and value with the rules applicable to consumption. No solver or learning property enters this implication. □

This contract proves encoded membership, not trajectory safety, availability, or a hard deadline. Withholding may itself require an application-specific fallback. The proposition is a specification-level argument; it is not a machine-checked proof of the Python implementation. Tests and independent audits check implementation behavior under the stated interface.

4 Feasible correction is not optimal correction

4.1 A reproduced numerical confound

The historical L_1 service runs a numerical linear program per branch. It tests the binary output exactly, attempts denominator-limited rational reconstruction if needed, and selects the closest surviving candidate. This certifies each returned candidate's membership. It does not certify global projection optimality when a better branch candidate is rejected.

We audited all 1,142 used targets from the five-trial teacher study. Generic rational enumeration, an independent structured objective formula, and a fast constructive exact oracle agree on every minimum. Fresh numerical replays reproduce every historical target. Table 1 reports excess correction distance, not an optimizer tolerance. The material threshold 10^{-6} is used only to summarize gaps; release checking remains exact.

Table 1: Historical used-target audit. Gaps are numerical distance minus exact minimum in dimensionless L_1 units. These are target counts, not independent trial estimates.

Geometry	Targets	Gap $> 10^{-6}$	Mean gap	Max. gap
Disconnected	881	70	0.09889	2.51168
Convex	261	72	0.21334	2.11211

In the reproduced large-gap case, $p = (-4218809, 5364421)/4194304$ and the coupled budget is $27/100$. An exact nearest action is $(27/100 - p_2, p_2)$, at distance 0.003135185. The numerical service instead retains $(1/4, 1/50)$, at distance 2.514820080: an excess of 2.511684895, or about $802\times$ the minimum. Exact checking correctly rejects the nearby infeasible numerical candidate; the defect concerns correction quality, not the checker.

For the convex control, merging the two branches into their equivalent single polyhedron turns 72 of 261 previously supplied numerical targets into withheld targets; 189 remain unchanged. Exact actions remain identical on all 261. These fractions are conditioned on targets used historically and are not coverage over all requests. Multiple optimal L_1 targets also matter: 314 disconnected and all 261 convex queries have nonunique minima. Eleven historical action-gradient conflicts admit no conflicting exact minimizer; other conflicts can depend on the optimal-face tie. Thus historical numerical-L_1 versus exact-L_2 comparisons combine projection norm, ties, and numerical candidate quality.

4.2 AN UNBOUNDED APPROXIMATION RATIO DESPITE FEASIBLE OUTPUT

Proposition 2 (verified candidate pruning). A repair service that returns the nearest exactly feasible surviving branch candidate need not have any uniform multiplicative bound on its global L_1 correction distance, even when the represented union is convex and bounded.

Proof. Let $F = [-1, 1]^2 \cap \{u_1 + u_2 \leq 0\}$, partitioned into F_- with $u_1 \leq 0$ and F_+ with $u_1 \geq 0$. For rational $0 < \epsilon < 1/2$, take $p = (-1/2, 1/2 + \epsilon)$. The sum constraint requires at least ϵ of L_1 change, attained by $(-1/2, 1/2)$. Suppose the numerical candidate and reconstruction attempts for F_- remain infeasible, so exact checking rejects them, while the candidate $(0, 0)$ from F_+ survives. The latter is branch-optimal but has distance $1 + \epsilon$ from p. The returned ratio is $1 + 1/\epsilon$, which is unbounded. □

This is behavior permitted by the candidate-selection contract, not a claim that a particular solver produces it for every ϵ. A mocked-backend regression isolates the contract, while the $802\times$ case is an actual replay. Relaxing the checker would trade away release soundness; the appropriate control is a better specified correction oracle.

4.3 EXACT TEACHERS AND DETERMINISTIC TIES

The revised L_1 teacher minimizes distance over the entire feasible union and then selects the lexicographically smallest rational action. In 2D, subdivision at the proposal's coordinate kink lines reduces the problem to affine minimization over polygon cells; exact vertex enumeration is a general reference. A faster algorithm applies to each nonempty box-and-budget branch. If $q = \mathrm{clip}(p, l, h)$ and $e = \max(0, \mathbf{1}^\mathsf{T} q - b)$, the optimum distance is

$$D_1^* = \|p - q\|_1 + e. \tag{2}$$

Reducing coordinates of q by a total of e, earliest coordinate first and without crossing l, attains this bound and the lexicographic tie. Exact L_2 references use active faces in 2D and rational water-filling for the allocation task (Appendix A). Every candidate still passes the independent verifier. Equivalent partitions and branch order preserve the exact selected action; lexicographic ties are a coordinate convention, not a rotation-invariant rule. Numerical versus exact L_1 changes both optimality and ties, so that contrast does not isolate a numerical-error-only causal effect.

5 LEARNING FROM CORRECTIONS AND SELECTING A BRANCH

For proposal $p_i = f_\theta(x_i)$, shared feasible task label y_i, frozen correction r_i, and fixed query mask m_i, flat-policy training minimizes

$$L(\theta) = \frac{1}{N} \sum_i \left[\frac{\|p_i - y_i\|^2}{d} + \lambda m_i \frac{\|p_i - \mathrm{stopgrad}(r_i)\|^2}{d} \right]. \tag{3}$$

The denominator includes all batch rows; targets and masks are detached. The task-reweighting control substitutes the existing y_i for r_i on the same mask. A soft-penalty control adds the minimum branch violation penalty, including bounds and budget. All methods learn from the same exact task labels, converted to the declared training precision. The allocation task's weighted evaluation cost and this label-regression training loss are distinct.

Action-space mechanism. For $\ell_y(p) = \|p - y\|^2 / 2$ and a repair-only step $p' = p - \eta(p - r)$,

$$\ell_y(p') - \ell_y(p) = -\eta(p - y)^\mathsf{T}(p - r) + \frac{\eta^2}{2} \|p - r\|^2. \tag{4}$$

If r is the exact Euclidean projection onto a closed convex set containing y, projection optimality gives $(p - y)^\mathsf{T}(p - r) \geq \|p - r\|^2$, so this action step does not increase ℓ_y for $0 \leq \eta \leq 2$. Disconnected sets need not satisfy this property: for $F = [-2, -1] \cup [1, 2]$, $p = -0.1$, $y = 1$, and nearest repair $r = -1$, the inner product is -0.99. The weighted-regression minimizer $(1 - \lambda)/(1 + \lambda)$ lies in the infeasible gap for every finite $\lambda > 0$. This elementary example isolates how a nearest correction can oppose the task objective.

For a neural parameter step the first-order term instead involves $(p-y)^{\mathsf{T}}JJ^{\mathsf{T}}(p-r)$, where J is the policy Jacobian. Shared parameters, batch coupling, Adam, and simultaneous task loss prevent action-space alignment from certifying a training trajectory. We therefore record parameter-gradient inner products and the task-loss change of one Adam step on the combined task plus exact-L_1 loss at weight one, with reset optimizer state, on a fixed diagnostic minibatch at the parent. This finite-step diagnostic is not a repair-only perturbation.

Branch-aware controls. A shared trunk predicts K logits and K action heads. Inference selects the head at the largest logit, with a deterministic index tie; it never averages branches. Gate cross-entropy and target-head MSE use the lowest-index branch containing the exact task label. The correction variant adds MSE to the head associated with its exact L_1 target, while the gate remains task-supervised. A constrained-output variant projects every head onto its own box-and-budget branch before task-head regression. Its active-set derivative is valid away from transitions; floating outputs still require exact release checking. These controls apply established mode-selection and projection ideas. The head budget is approximately matched to the flat MLP, but branch labels and architecture both differ, so cross-architecture effects are not an isolated intervention on capacity.

6 Frozen experimental protocol

6.1 Tasks and observable information

The 2D task exposes (o, a, b, B, g_1, g_2) and minimizes $c_x(u) = \|u-g\|^2$ subject to

$$(u_1 \le o-a \ \vee \ u_1 \ge o+a), \qquad u_1+u_2 \le b, \qquad -B \le u_j \le B. \tag{5}$$

Training excludes the large-gap/tight-budget combination; testing includes ID, held-out composition, and tighter global bounds. The convex control sets $a=0$ using the same remaining sampled parameters. Its composition split is consequently a budget-distribution control, not a novel logical formula.

The distinct synthetic allocation task assigns resources to $d=16$ jobs under $K=4$ operating modes. Mode k supplies heterogeneous minimum-service and capacity vectors l_k, h_k, with shared budget $\mathbf{1}^{\mathsf{T}}u \le b$. The observable demands g_j and positive weights w_j define $c_x(u) = d^{-1}\sum_j w_j(u_j-g_j)^2$. All 161 input features—budget, demands, weights, and mode bounds—are available to every method. ID and tighter-budget states test an independent-state application; no industrial dataset, temporal simulator, or actuator is implied.

6.2 Paired trials, teachers, and selection

The configuration and executed source were hashed locally before primary execution. The study fixes 20 fresh paired seeds per task, with 512 training, 128 validation, and 256 examples per test split. Each flat parent trains for 40 epochs. Its invalid proposals are frozen; the common mask includes only queries for which the numerical L_1 service supplies a verified target. Exact alternatives and numerical failures are logged for every invalid query. Numerical L_1, exact L_1, exact L_2, soft penalty, and task reweighting each use weights $0.25, 1, 4$. Continued task training is the reference. All 16 flat continuations start from identical parent weights and use the same 240 Adam updates, minibatches, and learning rate 0.0003. Parents use learning rate 0.003 and batch size 64. The unprojected branch task/correction pair shares its own parent and teacher mask; the projected task-only model has a separate parent and no teacher mask.

For each flat auxiliary family, validation selects the lowest intervention rate among candidates with full coverage, no released violations, and success at most two percentage points below the task continuation. The task continuation participates as a qualified fallback. Greater success, then lower regret, then the fixed name break ties. This tolerance is an experimental convention, not an application-approved requirement or proof of test noninferiority. Fixed weight-one candidates are also evaluated to expose effects hidden by fallback. All 60 trial selections and checkpoint hashes are saved before any final-test evaluation; the run count is fixed rather than extended according to significance. A descriptive precision ablation uses float64 on five predeclared seeds per task.

6.3 DIRECT OPTIMIZATION, RELEASE, AND MEASUREMENT

Every neural method uses the same exact lexicographic L_1 release repair. Numerical release is evaluated separately as a guard-sensitivity experiment. The direct task baseline freshly solves for the task-optimal action inside every decision interval and then uses the same verification, serialization, and release path. It never returns a precomputed dataset target. The 2D task includes both the generic active-face task optimizer and the faster structure-specific optimizer; allocation uses the weighted rational mode optimizer.

Native proposal feasibility is measured before the shared release repair. For unprojected networks it is raw neural feasibility; for the projected-head and direct-optimizer controls it follows their intrinsic projection or optimization. Success requires a released feasible action with exact regret $c_x(u) - c_x(y) \leq 0.05$; withheld attempts fail. We also report success across 12 regret thresholds, conditional regret quantiles, coverage, and repair frequency. Since the task scales differ, 0.05 is not claimed to express identical operational quality across applications.

All primary inference runs on CPU with one Torch thread and deterministic Torch algorithms. Timing includes input construction, neural inference or fresh optimization, guard work, serialization, and release. It excludes precompiled state/constraint construction and audit logging for every method. After all selections, three timing rounds randomize method blocks and state order; rounds are averaged within each seed. Paired intervals use 20 seed-level differences. The 45 predeclared contrasts comprise three tasks, five method pairs, and three endpoints; we show ordinary and Bonferroni simultaneous 95% Student-t intervals. They depend on seed-level distribution assumptions; repeated actions do not create additional training trials. Stage wall times and teacher-generation costs are retained. Background activity and OS scheduling prevent hard real-time claims.

7 RESULTS

Teacher quality changes the tradeoff. All primary methods have full test coverage and no observed released violations; these facts do not identify the best proposals. At fixed auxiliary weight one on the disconnected task, numerical L_1 has 75.64% native feasibility and 89.16% guarded success, compared with 71.00% and 94.38% for exact L_1. On the convex task the corresponding success rates are 91.80% and 98.96%. Replacing the numerical target with an exact, lexicographically tied target preserves more task quality here, but does not maximize native feasibility. This contrast changes optimality and tie choice together.

Selection and the reweighting control. On disconnected ID states, task-only continuation has 68.32% native feasibility and 96.41% success. The selected exact-L_1 family has 69.61% and 95.57%, while reweighting has 69.24% and 95.80%. Exact L_1 minus reweighting gives +0.371 [-0.604, +1.347] percentage points in native feasibility and -0.234 [-1.127, +0.658] in success; brackets are simultaneous 95% intervals over all 45 primary contrasts. These are selection-procedure outcomes, including task fallbacks, not effects of a fixed positive correction weight. Appendix tables expose every task, fallback counts, fixed-weight-one results, and paired intervals.

Direct task optimization is a decisive reference. Fresh direct solves attain zero regret and full success on every tested state. Table 2 includes their complete release cost and distinguishes the generic from the structure-specific 2D implementation. Direct structured optimization minus selected exact-L_1 latency is -0.139 [-0.180, -0.098] ms on disconnected 2D, -0.002 [-0.035, +0.032] ms on convex 2D, and +0.278 [-0.136, +0.693] ms on allocation, with simultaneous intervals. Direct optimization is faster in the disconnected comparison; the other two simultaneous intervals include zero and do not establish equivalent runtime. Fewer neural repairs do not demonstrate useful compute savings against this reference.

Branch choice, thresholds, and allocation. Adding exact-L_1 correction supervision to the unprojected branch model changes disconnected native feasibility by +3.32 percentage points and success by -2.13. A higher feasible fraction can still accompany worse utility. In convex 2D, projected-head success at regret 0.05 is 99.88% versus 99.02% for task continuation, but at 0.001 the ranking reverses: 44.86% versus 81.82% (Figure 2). The allocation stress test exposes poor learned utility at this budget: task continuation succeeds on 4.84% of ID states and projected heads on 4.82%. It is not a successful deployment demonstration.

Table 2: Complete decision latency in milliseconds (mean ± sample SD of 20 seed summaries). Each seed averages three randomized timing rounds. All methods use the same exact release path; direct methods freshly recompute the task optimum. Compilation and audit logging are excluded for all methods.

Method	Disconnected 2D	Convex 2D	Allocation 16D
Task	0.979 ± 0.069	0.678 ± 0.285	6.066 ± 1.703
Soft penalty	0.980 ± 0.080	0.651 ± 0.265	6.060 ± 1.770
Numerical L_1	0.980 ± 0.076	0.664 ± 0.282	6.029 ± 1.725
Exact L_1	0.975 ± 0.065	0.649 ± 0.263	6.015 ± 1.662
Exact L_2	0.980 ± 0.061	0.643 ± 0.247	6.037 ± 1.807
Task reweighting	0.991 ± 0.073	0.665 ± 0.252	6.055 ± 1.803
Branch task	1.115 ± 0.099	0.809 ± 0.312	6.335 ± 1.861
Branch + exact L_1	1.097 ± 0.079	0.816 ± 0.331	6.154 ± 1.856
Branch projection	3.732 ± 0.267	2.827 ± 1.090	7.637 ± 2.334
Direct task (structured)	0.836 ± 0.061	0.647 ± 0.250	6.293 ± 1.812
Direct task (generic)	2.609 ± 0.143	2.060 ± 0.805	—

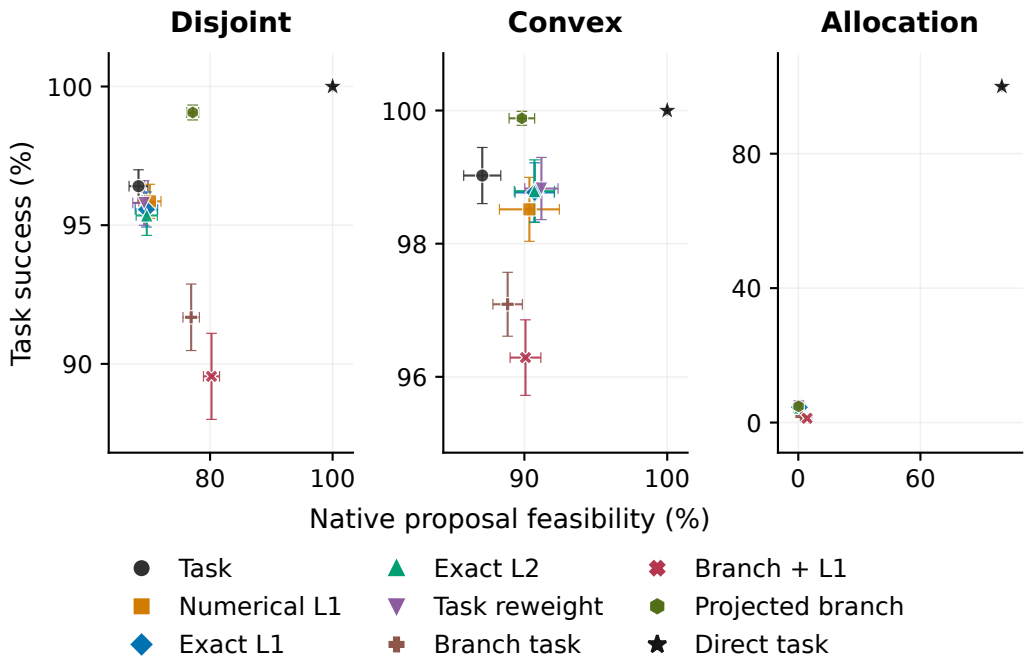


Figure 1: Measured ID native proposal feasibility and guarded success at regret 0.05. Points are means across 20 paired trials; bars are marginal, unadjusted 95% seed-level Student-t intervals. Direct optimization is a freshly computed reference, with a non-neural proposal.

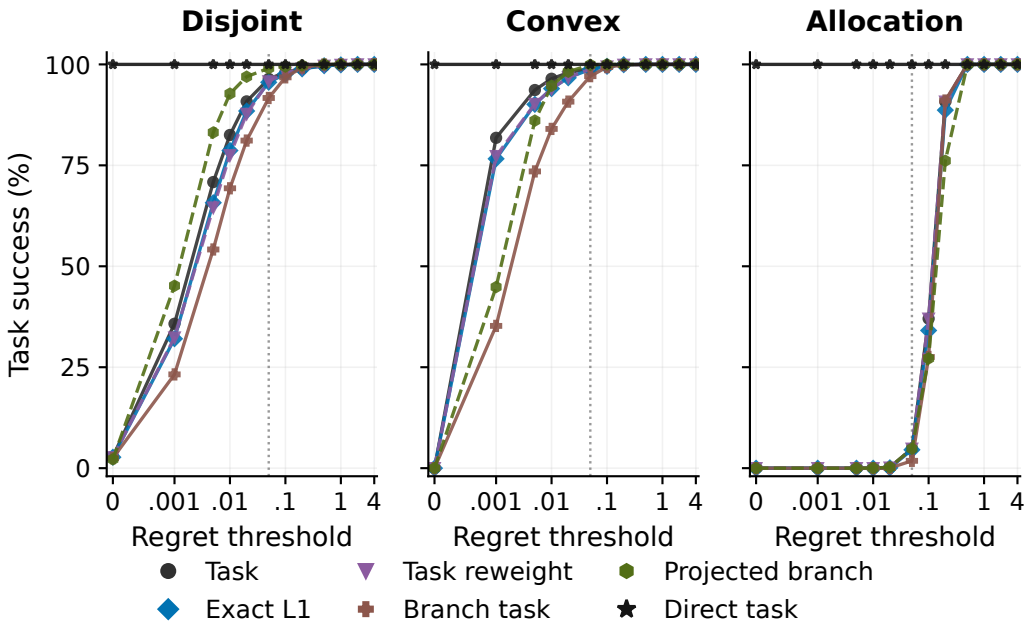


Figure 2: Measured ID success across regret thresholds. Every curve uses all attempted states, including withheld actions as failures. Curves are descriptive seed means; dotted vertical lines mark the fixed primary success threshold 0.05. The horizontal scale is linear up to 0.001 and logarithmic above it.

Secondary diagnostics. Matched teacher refresh changes disconnected ID feasibility by +0.41 percentage points and success by -0.53; these descriptive effects reuse the same parents. A separate post-hoc allocation execution control changes native feasibility from 0.08% with floating all-head projection to 100.00% with exact selected-head projection. The latter still performs one intrinsic projection per decision and has 5.04% task success. Appendix tables report paired uncertainty, projection counts, latency, and the separate non-neural dimension/mode cost profile.

The primary allocation projected model has only 0.14% exactly feasible native proposals, yet 99.96% have encoded-row violation residual at most 10^{-6} (Table 19). This low exact-feasibility fraction is sensitive to boundary arithmetic; poor guarded task utility is a separate finding. These residual diagnostics never relax the release checker.

8 Discussion and Limitations

Teacher specification changes the measured feasibility–utility tradeoff. Exact correction improves target quality without establishing a consistent advantage over task-label reweighting. Branch selection and arithmetic change the tradeoffs; direct task optimization supplies a decisive reference when objectives and constraints are observable. Fewer shared-guard repairs alone demonstrate neither useful constraint internalization nor lower total compute. Evaluations need task-quality requirements, explicit teacher ties, and complete release-cost accounting.

All tasks are synthetic, with observable objectives and constraints. Allocation broadens dimension and decision semantics but does not validate industrial control. Its label-regression loss and small training budget leave stronger learned optimizers open. Exact references exploit box/budget structure; their runtimes need not transfer to arbitrary polyhedral unions. The shared query mask excludes an exact teacher's possible coverage advantage. The historical audit motivated this revision, making the overall research program adaptive despite frozen new selections and trial counts. Twenty seeds do not guarantee power or population noninferiority. Branch controls add structured supervision; exact/numerical L_1 also differ in ties. Release soundness assumes correct compilation, arithmetic, context, and consumption; zero observed violations alone cannot certify physical safety.

Reproducibility statement

The anonymous supplement supplies current PyTorch source, exact teacher references, the frozen configuration and source snapshot, shared input files, checkpoint hashes, selected checkpoints, all selection metrics, and compact action records. Independent scripts reconcile exact constraints, regret, coverage, and reported summaries. Appendices give algorithms, proofs, selection details, and historical provenance. Running the supplied commands reproduces computation; historical timing measurements are observations of their recorded environment and cannot be authenticated by rerunning them on another machine.

AI assistance statement

Generative AI tools assisted with research ideation, literature discovery, experiment design, implementation, synthetic-data generator design, mathematical claims and proof drafting, diagnostic analysis, figure generation, and manuscript drafting and editing. The synthetic numerical examples are produced by the supplied seeded programs rather than unrecorded language-model samples. Verification includes executable tests, exact reference comparisons, and independent record audits; these checks do not substitute for author review. The authors are responsible for checking the cited sources, proofs, empirical interpretation, and all submitted content.

References

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/9ce3c52fc54362e22053399d3181c638-Abstract.html.

Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11797. URL https://ojs.aaai.org/index.php/AAAI/article/view/11797.

Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 136–145, 2017. URL https://proceedings.mlr.press/v70/amos17a.html.

Christopher M. Bishop. Mixture density networks. Technical Report NCRG/94/004, Aston University, Birmingham, UK, 1994. URL https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/bishop-ncrg-94-004.pdf.

Wonje Choi, Jooyoung Kim, and Honguk Woo. NeSyPr: Neurosymbolic proceduralization for efficient embodied reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5804. URL https://papers.nips.cc/paper_files/paper/2025/hash/fee6424a319905353c3d7ffdfcfeee67-Abstract-Conference.html.

Priya L. Donti, David Rolnick, and J. Zico Kolter. DC3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/pdf?id=V1ZHVxJ6dSS.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2410–2420. Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-1228. URL https://aclanthology.org/P16-1228/.

Leander Kurscheidt, Paolo Morettin, Roberto Sebastiani, Andrea Passerini, and Antonio Vergari. A probabilistic neuro-symbolic layer for algebraic constraint satisfaction. In *Proceedings of the Forty-first Conference on Uncertainty in Artificial Intelligence*, volume 286 of *Proceedings of Machine Learning Research*, pp. 2431–2471. PMLR, 2025. URL https://proceedings.mlr.press/v286/kurscheidt25a.html.

Enming Liang, Minghua Chen, and Steven Low. Low complexity homeomorphic projection to ensure neural-network solution feasibility for optimization over (non-)convex set. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 20623–20649, 2023. URL https://proceedings.mlr.press/v202/liang23a.html.

Youngjae Min and Navid Azizan. HardNet: Hard-constrained neural networks with universal approximation guarantees. arXiv preprint arXiv:2410.10807, 2025. URL https://arxiv.org/abs/2410.10807v4. Version 4, revised 19 October 2025; first posted in 2024.

Shraman Pal and Can Li. DisjunctiveNet: Neural symbolic learning via differentiable convexified optimization layers. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.30456v3. Technical comparison uses arXiv:2605.30456v3.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Kim Peter Wabersich and Melanie N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021. doi: 10.1016/j.automatica.2021.109597. URL https://www.sciencedirect.com/science/article/pii/S0005109821001175.

Lingxuan Wu, Zijian Zhu, Lizhong Wang, Chengyang Ying, Huayu Chen, Xiao Yang, Fangming Liu, and Jun Zhu. PACT: Self-evolving physical safety alignment for diffusion policies in embodied manipulation. arXiv preprint arXiv:2606.08414, 2026. URL https://arxiv.org/abs/2606.08414v1. Version 1.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe78a8acf5fda99de95303940a2420c-Abstract.html.

Mehdi Zafari, Iman Mohammadi, and A. Lee Swindlehurst. Feasible but not safe: Constraint violations and report-channel attacks in learned cell-free ISAC association. In *17th ACM Workshop on Wireless of the Students, by the Students, for the Students (S3)*, 2026. URL https://arxiv.org/abs/2609.03147v1. Accepted; author-deposited manuscript.

A Reference algorithms and elementary proofs

A.1 Exact box-and-budget correction

For a nonempty branch $C = \{u : l \le u \le h, \sum_j u_j \le b\}$, clipping $q = \mathrm{clip}(p, l, h)$ gives, for every box-feasible u,

$$\|p - u\|_1 = \|p - q\|_1 + \|q - u\|_1 .$$

This follows coordinatewise: a coordinate outside its interval reaches every feasible coordinate through its clipped endpoint, while an inside coordinate is unchanged. The budget implies $\|q - u\|_1 \ge e = \max(0, \sum_j q_j - b)$. If $e > 0$, the available reduction $\sum_j (q_j - l_j)$ is at least e, since nonemptiness implies $\sum_j l_j \le b$. Reduce coordinates from the first through the last until the total reduction reaches e. Equality in the lower bound forces every optimal point to satisfy $u \le q$ coordinatewise with total decrease exactly e. The constructed point is therefore lexicographically smallest: at an earlier coordinate no optimum can reduce more without exceeding the required total or its lower bound. If $e = 0$, clipping is already the unique box projection. Enumerate branches and choose minimum exact distance followed by lexicographic action. Branch labels do not enter this comparison.

The generic 2D L_1 reference augments each branch's constraint and box boundary lines with $u_1 = p_1$ and $u_2 = p_2$. These lines partition the bounded polygon into cells on which the objective is affine. A minimum and a lexicographic minimizing point occur at a cell vertex, including bounded segments and singleton cases. Feasible intersections of nonparallel boundary/kink lines therefore contain a global optimum. An empty candidate set represents an empty bounded branch. Tests cover lower-dimensional sets, redundant rows, reordered branches, and equivalent partitions. The fast reference rejects unsupported row structure rather than silently approximating a general set.

A.2 Exact Euclidean and weighted task optimization

In 2D a closest point to a bounded polygon is the goal itself, a perpendicular projection onto an edge, or a vertex. Enumerating these candidates and selecting an exactly feasible minimum squared distance gives the generic task/L_2 oracle. For a box with coupled budget, if clipping violates the budget then the optimum lies on $u_1 + u_2 = b$. Substitute $u_2 = b - u_1$, minimize the resulting quadratic, and clip $(p_1 - p_2 + b)/2$ to

$$[\max(l_1, b - h_2),\ \min(h_1, b - l_2)].$$

This yields the unique branch optimum. Comparing exact branch costs and actions solves the union.

For allocation mode k, positive weights make its objective strictly convex. The KKT form is

$$u_j(\mu) = \mathrm{clip}\left(g_j - \frac{\mu}{2w_j}, l_{kj}, h_{kj}\right), \qquad \mu \ge 0.$$

If $\mu = 0$ satisfies the budget, it is optimal. Otherwise $\sum_j u_j(\mu) = b$. This sum is continuous and piecewise affine in μ; rational breakpoints $2w_j(g_j - h_{kj})$ and $2w_j(g_j - l_{kj})$ identify an interval on which its root is solved exactly. The positive weights give a unique mode optimum. Enumerating all modes and choosing minimum exact cost, then lexicographic action, gives the global optimizer. Dividing cost by d does not change the minimizer. Unweighted L_2 correction sets all weights to one. These structured reference algorithms exploit box-and-budget geometry.

A.3 Projection alignment and regression compromises

For convex C and $r = \Pi_C(p)$, $r + t(y - r) \in C$ for $0 \le t \le 1$. The squared distance from p to this segment is minimized at $t = 0$, whose nonnegative right derivative gives $(p - r)^\mathsf{T}(y - r) \le 0$. Therefore

$$(p - y)^\mathsf{T}(p - r) = \|p - r\|^2 - (y - r)^\mathsf{T}(p - r) \ge \|p - r\|^2 .$$

Substituting into (4) bounds the task change by $(-\eta + \eta^2/2)\|p - r\|^2 \le 0$ when $0 \le \eta \le 2$. The premise is exact Euclidean projection, not merely a feasible candidate or arbitrary L_1 minimizer.

For fixed y, r and $\lambda > 0$, the Hessian of $\|p - y\|^2/2 + \lambda\|p - r\|^2/2$ is $(1 + \lambda)I$, so its unique minimizer is $(y + \lambda r)/(1 + \lambda)$. In the disconnected example this equals $(1 - \lambda)/(1 + \lambda) \in (-1, 1)$,

outside the feasible set. A shared continuous policy on a connected goal interval also cannot realize both disconnected components while remaining feasible everywhere: if its endpoint outputs lie in opposite components, the intermediate-value theorem forces an output in the gap. A discrete gate removes that continuity restriction; it does not by itself guarantee that the chosen head is feasible.

At a differentiable policy state, the squared repair-label loss has gradient $\nabla_\theta \ell_r = J^\mathsf{T}(p-r)$. A small repair-only parameter step changes the proposal by $-\eta J J^\mathsf{T}(p-r)+o(\eta)$ and the squared task-label loss by $-\eta(p-y)^\mathsf{T} J J^\mathsf{T}(p-r)+o(\eta)$. This expansion does not describe the branch cross-entropy term or a nondifferentiable gate transition. Action-gradient conflict counts and parameter updates are different diagnostics; Adam and finite minibatch updates require measuring the actual step.

A.4 Boundary precision and a margin implication

Small imitation loss does not guarantee exact raw feasibility when targets lie on the boundary: for $F = [-1, 0]$, target $r = 0$ and proposal $p = \delta > 0$, the squared error δ^2 tends to zero while every proposal is invalid. Conversely, suppose each teacher target has margin at least $\gamma > 0$ within one branch after normalizing its nonzero row normals in Euclidean norm, including the box rows. If $\mathbb{E}\,\|p-r\|^2 \le \varepsilon$, then violation of that branch implies $\|p-r\| > \gamma$. Raw infeasibility implies violation of the target's branch, so Markov's inequality bounds its probability by $\min(1, \varepsilon/\gamma^2)$. This conditional observation does not supply a margin for boundary projection targets. The precision and residual-curve diagnostics do not relax the exact acceptance criterion.

B Additional results and experiment details

B.1 Primary estimates and distribution shifts

Table 3: Disconnected 2D: ID results, mean ± sample SD across 20 paired trials. Success counts all attempted states; mean regret conditions on feasible releases.

Method	Native feasible (%)	Success (%)	Mean regret	Coverage (%)
Task	68.32 ± 3.29	96.41 ± 1.26	0.0120 ± 0.0045	100.00
Soft penalty	69.75 ± 4.15	95.70 ± 1.54	0.0135 ± 0.0044	100.00
Numerical L_1	70.18 ± 3.87	95.86 ± 1.31	0.0132 ± 0.0044	100.00
Exact L_1	69.61 ± 3.90	95.57 ± 1.35	0.0138 ± 0.0043	100.00
Exact L_2	69.67 ± 3.69	95.35 ± 1.54	0.0140 ± 0.0049	100.00
Task reweighting	69.24 ± 3.94	95.80 ± 1.72	0.0136 ± 0.0049	100.00
Branch task	76.91 ± 2.86	91.68 ± 2.56	0.0165 ± 0.0036	100.00
Branch + exact L_1	80.23 ± 2.80	89.55 ± 3.31	0.0200 ± 0.0045	100.00
Branch projection	77.17 ± 2.08	99.06 ± 0.57	0.0043 ± 0.0010	100.00
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.0000 ± 0.0000	100.00
Direct task (generic)	100.00 ± 0.00	100.00 ± 0.00	0.0000 ± 0.0000	100.00

Table 4: Convex 2D: ID results, mean ± sample SD across 20 paired trials. Success counts all attempted states; mean regret conditions on feasible releases.

Method	Native feasible (%)	Success (%)	Mean regret	Coverage (%)
Task	87.05 ± 2.78	99.02 ± 0.90	0.0022 ± 0.0010	100.00
Soft penalty	92.68 ± 2.72	98.48 ± 1.12	0.0036 ± 0.0012	100.00
Numerical L_1	90.35 ± 4.48	98.52 ± 1.03	0.0033 ± 0.0015	100.00
Exact L_1	90.72 ± 2.93	98.77 ± 0.95	0.0030 ± 0.0011	100.00
Exact L_2	90.70 ± 3.01	98.79 ± 1.01	0.0029 ± 0.0012	100.00
Task reweighting	91.19 ± 2.49	98.83 ± 1.00	0.0030 ± 0.0012	100.00
Branch task	88.83 ± 2.20	97.09 ± 1.03	0.0073 ± 0.0017	100.00
Branch + exact L_1	90.08 ± 2.30	96.29 ± 1.21	0.0083 ± 0.0020	100.00
Branch projection	89.82 ± 1.91	99.88 ± 0.22	0.0028 ± 0.0005	100.00
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.0000 ± 0.0000	100.00
Direct task (generic)	100.00 ± 0.00	100.00 ± 0.00	0.0000 ± 0.0000	100.00

Table 5: Allocation 16D: ID results, mean ± sample SD across 20 paired trials. Success counts all attempted states; mean regret conditions on feasible releases.

Method	Native feasible (%)	Success (%)	Mean regret	Coverage (%)
Task	0.27 ± 0.31	4.84 ± 1.65	0.1240 ± 0.0053	100.00
Soft penalty	1.09 ± 1.49	3.55 ± 2.21	0.1396 ± 0.0238	100.00
Numerical L_1	0.41 ± 0.35	4.57 ± 1.55	0.1286 ± 0.0103	100.00
Exact L_1	0.55 ± 0.73	4.55 ± 1.75	0.1294 ± 0.0119	100.00
Exact L_2	0.70 ± 0.95	4.34 ± 1.78	0.1301 ± 0.0104	100.00
Task reweighting	0.27 ± 0.31	4.82 ± 1.61	0.1241 ± 0.0053	100.00
Branch task	1.50 ± 0.76	1.82 ± 0.85	0.1318 ± 0.0035	100.00
Branch + exact L_1	4.32 ± 1.57	1.27 ± 0.69	0.1412 ± 0.0044	100.00
Branch projection	0.14 ± 0.29	4.82 ± 3.06	0.1524 ± 0.0176	100.00
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.0000 ± 0.0000	100.00

Table 6: Disconnected 2D: paired ID effects across 20 seeds. Simultaneous intervals adjust the complete 45-contrast family. Native-feasibility and success differences are percentage points; latency is milliseconds. A degenerate sample interval does not establish population equivalence.

Metric	Mean	95% interval	Simultaneous interval
Exact L_1 − Numerical L_1			
NF (pp)	-0.566	[-1.650, +0.517]	[-2.552, +1.419]
Success (pp)	-0.293	[-0.752, +0.166]	[-1.134, +0.548]
Latency (ms)	-0.004	[-0.026, +0.017]	[-0.043, +0.035]
Exact L_1 − Task reweighting			
NF (pp)	+0.371	[-0.161, +0.903]	[-0.604, +1.347]
Success (pp)	-0.234	[-0.721, +0.253]	[-1.127, +0.658]
Latency (ms)	-0.015	[-0.032, +0.002]	[-0.047, +0.016]
Branch + exact L_1 − Branch task			
NF (pp)	+3.320	[+2.795, +3.846]	[+2.357, +4.284]
Success (pp)	-2.129	[-2.671, -1.587]	[-3.122, -1.135]
Latency (ms)	-0.018	[-0.039, +0.003]	[-0.057, +0.021]
Direct task − Task			
NF (pp)	+31.680	[+30.141, +33.218]	[+28.859, +34.501]
Success (pp)	+3.594	[+3.002, +4.186]	[+2.508, +4.679]
Latency (ms)	-0.143	[-0.159, -0.127]	[-0.172, -0.114]
Direct task − Exact L_1			
NF (pp)	+30.391	[+28.565, +32.216]	[+27.043, +33.738]
Success (pp)	+4.434	[+3.802, +5.065]	[+3.276, +5.591]
Latency (ms)	-0.139	[-0.162, -0.117]	[-0.180, -0.098]

Table 7: Convex 2D: paired ID effects across 20 seeds. Simultaneous intervals adjust the complete 45-contrast family. Native-feasibility and success differences are percentage points; latency is milliseconds. A degenerate sample interval does not establish population equivalence.

Metric	Mean	95% interval	Simultaneous interval
Exact L_1 − Numerical L_1			
NF (pp)	+0.371	[-1.684, +2.426]	[-3.397, +4.139]
Success (pp)	+0.254	[-0.162, +0.670]	[-0.509, +1.017]
Latency (ms)	-0.016	[-0.049, +0.017]	[-0.076, +0.045]
Exact L_1 − Task reweighting			
NF (pp)	-0.469	[-1.126, +0.188]	[-1.673, +0.735]
Success (pp)	-0.059	[-0.266, +0.149]	[-0.440, +0.322]
Latency (ms)	-0.016	[-0.039, +0.006]	[-0.058, +0.025]
Branch + exact L_1 − Branch task			
NF (pp)	+1.250	[+0.828, +1.672]	[+0.476, +2.024]
Success (pp)	-0.801	[-1.026, -0.575]	[-1.215, -0.387]
Latency (ms)	+0.007	[-0.013, +0.026]	[-0.029, +0.043]
Direct task − Task			
NF (pp)	+12.949	[+11.649, +14.249]	[+10.566, +15.332]
Success (pp)	+0.977	[+0.555, +1.398]	[+0.204, +1.749]
Latency (ms)	-0.031	[-0.054, -0.007]	[-0.074, +0.012]
Direct task − Exact L_1			
NF (pp)	+9.277	[+7.904, +10.650]	[+6.760, +11.795]
Success (pp)	+1.230	[+0.786, +1.675]	[+0.415, +2.046]
Latency (ms)	-0.002	[-0.020, +0.016]	[-0.035, +0.032]

Table 8: Allocation 16D: paired ID effects across 20 seeds. Simultaneous intervals adjust the complete 45-contrast family. Native-feasibility and success differences are percentage points; latency is milliseconds. A degenerate sample interval does not establish population equivalence.

Metric	Mean	95% interval	Simultaneous interval
Exact L_1 − Numerical L_1			
NF (pp)	+0.137	[-0.143, +0.417]	[-0.377, +0.650]
Success (pp)	-0.020	[-0.342, +0.302]	[-0.610, +0.571]
Latency (ms)	-0.014	[-0.170, +0.142]	[-0.300, +0.272]
Exact L_1 − Task reweighting			
NF (pp)	+0.273	[-0.073, +0.620]	[-0.362, +0.908]
Success (pp)	-0.273	[-0.644, +0.097]	[-0.953, +0.407]
Latency (ms)	-0.040	[-0.244, +0.165]	[-0.415, +0.335]
Branch + exact L_1 − Branch task			
NF (pp)	+2.812	[+2.109, +3.516]	[+1.523, +4.102]
Success (pp)	-0.547	[-0.801, -0.292]	[-1.013, -0.080]
Latency (ms)	-0.181	[-0.378, +0.016]	[-0.543, +0.181]
Direct task − Task			
NF (pp)	+99.727	[+99.580, +99.873]	[+99.458, +99.995]
Success (pp)	+95.156	[+94.382, +95.931]	[+93.737, +96.576]
Latency (ms)	+0.227	[+0.080, +0.374]	[-0.043, +0.496]
Direct task − Exact L_1			
NF (pp)	+99.453	[+99.110, +99.796]	[+98.825, +100.082]
Success (pp)	+95.449	[+94.631, +96.267]	[+93.949, +96.949]
Latency (ms)	+0.278	[+0.052, +0.504]	[-0.136, +0.693]

Table 9: Disconnected 2D composition: 20-seed means and sample SD. These are distribution-shift outcomes, separate from the ID primary comparisons.

Method	Native feasible (%)	Success (%)	Mean regret
Task	42.34 ± 3.66	85.27 ± 3.78	0.084 ± 0.027
Soft penalty	43.09 ± 3.98	84.65 ± 4.40	0.086 ± 0.028
Numerical L_1	42.93 ± 3.90	84.69 ± 4.17	0.089 ± 0.030
Exact L_1	42.97 ± 3.80	84.04 ± 4.93	0.089 ± 0.031
Exact L_2	42.99 ± 3.83	83.75 ± 5.61	0.088 ± 0.031
Task reweighting	42.97 ± 3.73	84.34 ± 5.01	0.086 ± 0.029
Branch task	41.86 ± 3.25	89.90 ± 2.74	0.032 ± 0.009
Branch + exact L_1	43.65 ± 3.47	88.98 ± 3.50	0.034 ± 0.009
Branch projection	62.87 ± 2.82	96.66 ± 1.50	0.018 ± 0.013
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000

Table 10: Disconnected 2D tight bounds: 20-seed means and sample SD. These are distribution-shift outcomes, separate from the ID primary comparisons.

Method	Native feasible (%)	Success (%)	Mean regret
Task	34.77 ± 6.81	88.73 ± 8.96	0.025 ± 0.014
Soft penalty	35.04 ± 6.95	88.38 ± 9.22	0.027 ± 0.015
Numerical L_1	35.18 ± 6.75	88.36 ± 8.87	0.026 ± 0.014
Exact L_1	34.98 ± 6.55	87.89 ± 9.13	0.027 ± 0.014
Exact L_2	34.90 ± 6.69	87.97 ± 9.47	0.027 ± 0.015
Task reweighting	34.84 ± 6.75	88.09 ± 9.27	0.026 ± 0.014
Branch task	40.47 ± 4.95	88.83 ± 4.03	0.022 ± 0.006
Branch + exact L_1	43.12 ± 5.26	87.27 ± 4.63	0.025 ± 0.007
Branch projection	76.95 ± 2.36	97.79 ± 1.45	0.010 ± 0.004
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000

Table 11: Convex 2D composition: 20-seed means and sample SD. These are distribution-shift outcomes, separate from the ID primary comparisons.

Method	Native feasible (%)	Success (%)	Mean regret
Task	79.26 ± 4.47	98.03 ± 1.24	0.004 ± 0.001
Soft penalty	88.52 ± 4.22	96.29 ± 1.68	0.007 ± 0.002
Numerical L_1	82.73 ± 6.00	97.17 ± 1.78	0.006 ± 0.002
Exact L_1	85.12 ± 4.63	97.07 ± 1.84	0.006 ± 0.002
Exact L_2	85.16 ± 4.15	97.15 ± 1.77	0.006 ± 0.002
Task reweighting	85.61 ± 3.65	97.21 ± 1.79	0.006 ± 0.002
Branch task	78.03 ± 3.83	94.94 ± 2.19	0.011 ± 0.003
Branch + exact L_1	80.39 ± 3.88	93.36 ± 2.42	0.013 ± 0.004
Branch projection	82.34 ± 2.31	99.77 ± 0.43	0.003 ± 0.001
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000

Table 12: Convex 2D tight bounds: 20-seed means and sample SD. These are distribution-shift outcomes, separate from the ID primary comparisons.

Method	Native feasible (%)	Success (%)	Mean regret
Task	52.52 ± 8.14	88.54 ± 9.35	0.026 ± 0.022
Soft penalty	53.36 ± 8.02	87.97 ± 9.59	0.027 ± 0.023
Numerical L_1	52.97 ± 8.26	88.46 ± 9.34	0.026 ± 0.023
Exact L_1	53.16 ± 8.01	88.11 ± 9.57	0.027 ± 0.023
Exact L_2	53.03 ± 7.95	88.36 ± 9.37	0.027 ± 0.023
Task reweighting	53.14 ± 7.93	88.38 ± 9.13	0.027 ± 0.023
Branch task	49.34 ± 7.79	94.10 ± 4.22	0.015 ± 0.006
Branch + exact L_1	50.14 ± 7.96	93.55 ± 4.67	0.015 ± 0.006
Branch projection	90.06 ± 2.00	98.55 ± 1.68	0.009 ± 0.004
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000

Table 13: Allocation 16D tight budget: 20-seed means and sample SD. These are distribution-shift outcomes, separate from the ID primary comparisons.

Method	Native feasible (%)	Success (%)	Mean regret
Task	0.00 ± 0.00	7.70 ± 1.97	0.099 ± 0.005
Soft penalty	0.00 ± 0.00	8.96 ± 3.03	0.096 ± 0.005
Numerical L_1	0.00 ± 0.00	7.95 ± 1.97	0.098 ± 0.004
Exact L_1	0.00 ± 0.00	7.81 ± 2.12	0.098 ± 0.004
Exact L_2	0.00 ± 0.00	8.14 ± 2.02	0.097 ± 0.004
Task reweighting	0.00 ± 0.00	7.70 ± 1.99	0.099 ± 0.005
Branch task	0.00 ± 0.00	3.52 ± 1.18	0.127 ± 0.003
Branch + exact L_1	0.00 ± 0.00	3.32 ± 1.06	0.127 ± 0.003
Branch projection	0.43 ± 1.12	5.04 ± 1.19	0.113 ± 0.003
Direct task (structured)	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000

B.2 Teacher diagnostics and ablations

Table 14: Teacher-bank and fixed-minibatch diagnostics. L1 gaps exceed 10^{-6}; parameter conflict is a negative task/auxiliary gradient dot product. Task increase counts a positive label-loss change after a reset-state Adam step on the combined task plus exact-L_1 loss at weight one, using the first 64 dataset rows. The last two columns count parents out of 20, not individual actions.

Task	Parent	Invalid	Used	L1 gaps	Param. conflict	Task increase
Disconnected 2D	flat	3213	3213	197	7	0
Disconnected 2D	branch	2344	2344	200	10	0
Convex 2D	flat	1319	1318	292	5	20
Convex 2D	branch	1031	1030	234	10	0
Allocation 16D	flat	10227	10226	1142	4	0
Allocation 16D	branch	10071	10071	2065	5	0

Table 15: Trainable parameter counts. Both branch variants use the same trunk width. Branch supervision and architecture still differ from flat regression.

Task	Flat parameters	Branch parameters
Disconnected 2D	4738	4718
Convex 2D	4738	4718
Allocation 16D	15568	15458

Table 16: Fixed auxiliary weight one on ID: 20-seed means and sample SD. These candidates are evaluated without the validation fallback used in the main selected-family comparisons.

Method	Native feasible (%)	Success (%)	Regret	Coverage (%)
Disconnected 2D				
Exact L_1	71.00 ± 3.13	94.38 ± 1.47	0.016 ± 0.005	100.00
Exact L_2	70.80 ± 3.10	94.45 ± 1.39	0.015 ± 0.004	100.00
Numerical L_1	75.64 ± 3.13	89.16 ± 3.79	0.024 ± 0.009	100.00
Task reweighting	69.69 ± 3.97	95.64 ± 1.41	0.014 ± 0.004	100.00
Convex 2D				
Exact L_1	88.95 ± 2.91	98.96 ± 0.87	0.003 ± 0.001	100.00
Exact L_2	88.95 ± 2.83	98.93 ± 1.00	0.002 ± 0.001	100.00
Numerical L_1	96.82 ± 1.52	91.80 ± 2.85	0.012 ± 0.004	100.00
Task reweighting	89.18 ± 2.64	98.93 ± 1.00	0.003 ± 0.001	100.00
Allocation 16D				
Exact L_1	0.64 ± 0.58	3.11 ± 1.14	0.139 ± 0.006	100.00
Exact L_2	0.70 ± 0.56	3.14 ± 1.12	0.139 ± 0.006	100.00
Numerical L_1	0.59 ± 0.43	3.16 ± 1.29	0.141 ± 0.006	100.00
Task reweighting	0.27 ± 0.31	4.82 ± 1.61	0.124 ± 0.005	100.00

Table 17: Numerical release-service sensitivity on the first 128 ID states of each seed: 20-seed means and sample SD. Every proposed release still passes the exact checker. These descriptive values use fewer states than the 256-state primary tables and do not supply a matched guard-effect estimate.

Method	Native feasible (%)	Success (%)	Regret	Coverage (%)
Disconnected 2D				
Branch task	76.76 ± 4.19	88.95 ± 3.54	0.080 ± 0.043	100.00
Exact L_1	69.26 ± 4.64	93.32 ± 2.27	0.080 ± 0.054	100.00
Numerical L_1	69.69 ± 4.39	93.95 ± 1.86	0.065 ± 0.041	100.00
Task	67.97 ± 3.73	94.10 ± 1.95	0.070 ± 0.045	100.00
Task reweighting	68.91 ± 4.59	93.40 ± 2.37	0.079 ± 0.056	100.00
Convex 2D				
Branch task	88.48 ± 2.58	94.45 ± 2.24	0.045 ± 0.025	100.00
Exact L_1	90.90 ± 3.25	96.91 ± 1.85	0.030 ± 0.024	100.00
Numerical L_1	90.70 ± 4.80	96.37 ± 1.59	0.033 ± 0.022	100.00
Task	87.19 ± 3.05	96.21 ± 1.37	0.044 ± 0.018	100.00
Task reweighting	91.21 ± 3.17	96.88 ± 1.86	0.027 ± 0.021	100.00
Allocation 16D				
Branch task	1.60 ± 1.23	2.07 ± 1.35	0.147 ± 0.005	100.00
Exact L_1	0.66 ± 0.81	3.87 ± 1.90	0.141 ± 0.009	100.00
Numerical L_1	0.51 ± 0.63	4.06 ± 1.77	0.140 ± 0.009	100.00
Task	0.43 ± 0.59	4.53 ± 1.77	0.136 ± 0.005	99.96
Task reweighting	0.43 ± 0.59	4.69 ± 2.00	0.136 ± 0.006	99.96

Table 18: Descriptive precision ablation on the same five declared seeds (1101–1105): means and sample SD. Exact L_1 uses fixed auxiliary weight one. Float64 retrains its own parent and continuation with its own teacher mask; it is a training-and-inference pipeline comparison, not casting a fixed checkpoint.

Training precision	Native feasible (%)	Success (%)	Regret
Disconnected 2D			
Float32 task	68.12 ± 1.78	96.64 ± 0.98	0.013 ± 0.006
Float32 exact L_1	71.02 ± 1.47	94.30 ± 1.69	0.016 ± 0.006
Float64 task	68.12 ± 1.78	96.64 ± 0.98	0.013 ± 0.006
Float64 exact L_1	71.09 ± 1.59	94.22 ± 1.86	0.016 ± 0.006
Convex 2D			
Float32 task	86.33 ± 3.05	99.38 ± 0.90	0.002 ± 0.001
Float32 exact L_1	87.58 ± 2.46	99.45 ± 0.76	0.002 ± 0.001
Float64 task	86.33 ± 3.05	99.38 ± 0.90	0.002 ± 0.001
Float64 exact L_1	87.58 ± 2.46	99.45 ± 0.76	0.002 ± 0.001
Allocation 16D			
Float32 task	0.31 ± 0.17	5.70 ± 1.96	0.123 ± 0.004
Float32 exact L_1	0.47 ± 0.33	3.52 ± 1.32	0.138 ± 0.003
Float64 task	0.31 ± 0.17	5.70 ± 1.96	0.123 ± 0.004
Float64 exact L_1	0.39 ± 0.39	3.52 ± 1.32	0.138 ± 0.003

Table 19: ID proposals with violation residual at most the column threshold (percent; means over 20 seeds). Residual is the minimum, over branches, of the maximum positive encoded-row residual, including global box rows, without row normalization. These are descriptive diagnostics for selected methods; all curves and intervals are supplied. Release checking remains exact at zero. Near-zero residual does not imply task success.

Method	0	10^{-9}	10^{-7}	10^{-6}	10^{-4}
Disconnected 2D					
Task	68.32	68.32	68.32	68.32	68.32
Exact L_1	69.61	69.61	69.61	69.61	69.63
Branch projection	77.17	78.11	98.83	100.00	100.00
Convex 2D					
Task	87.05	87.05	87.05	87.05	87.13
Exact L_1	90.72	90.72	90.72	90.72	90.78
Branch projection	89.82	89.82	99.10	100.00	100.00
Allocation 16D					
Task	0.27	0.27	0.27	0.27	0.27
Exact L_1	0.55	0.55	0.55	0.55	0.55
Branch projection	0.14	0.20	84.94	99.96	100.00

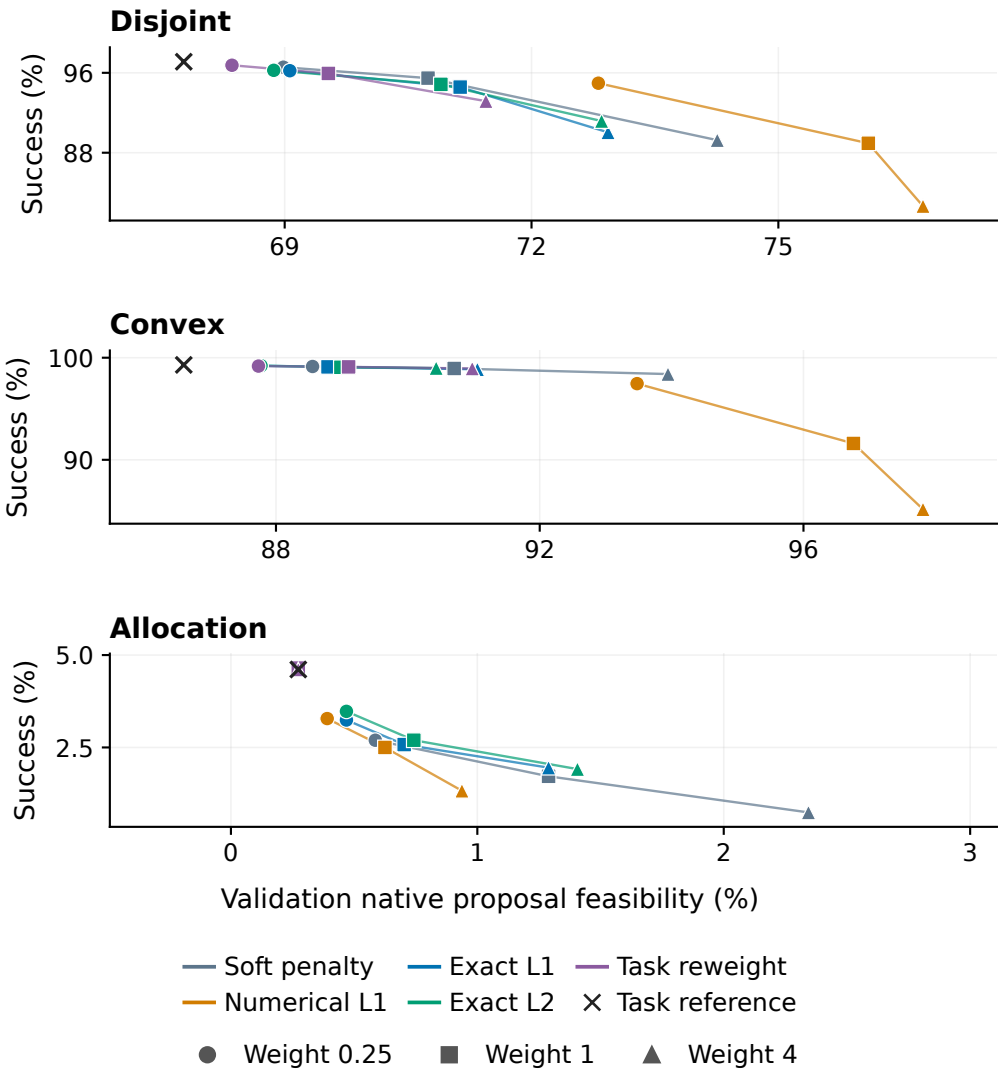


Figure 3: Validation means for every trained flat candidate weight, including unselected weights. These summarize the fixed selection search; they are not held-out test frontiers. The selected checkpoint identities were frozen before final testing.

B.3 MATCHED TEACHER REFRESH AND STRUCTURED COST PROFILES

Two secondary controls were locally frozen before the primary held-out evaluation. The refresh control reuses the 20 disconnected-task parents and the same fixed query indices as the primary study. Both arms make three exact L_1 query banks, at epochs 0, 10, and 20, and train for 30 epochs with one persistent Adam optimizer. The frozen-repeat arm queries the original parent proposals again; the refreshed arm queries its current proposals at the same indices, including identity targets if those proposals have become feasible. Both therefore make $3M$ queries, where M is the seed's common-mask size. Repeating a frozen query is intentional budget matching, not a recommended production strategy. Every frozen-repeat final tensor hash matches the primary fixed-weight-one exact-L_1 continuation. These are paired teacher-refresh controls on existing parents, not 20 additional independent primary trials or a fully adaptive query-routing method.

The structured cost profile uses dimensions 8, 16, and 32 crossed with 2, 4, and 8 modes. Each of the nine shapes uses five fresh seeds, 32 states, three randomized measured rounds, and eight warm-up attempts per method. It compares fresh weighted task optimization with the observable demand vector followed by exact L_1 repair and verified release. It does not train or time a scaled neural policy. State compilation and audit logging are excluded from both intervals; all 9,360 attempts, including 720 warm-ups, are retained. Each mode has $2d+1$ branch rows and $3d$ nonzero branch coefficients; dense storage contains $(2d+1)d$ coefficients per mode, with another $2d$ global box inequalities. The profile characterizes these structured implementations, not general polyhedral complexity.

Table 20: Refreshed minus frozen-repeat exact L_1: descriptive paired effects over the same 20 primary disconnected-task parents. Intervals are pointwise and separate from the 45 primary contrasts.

Split	Metric	Mean effect	95% interval
id	NF (pp)	+0.4102	[+0.2276, +0.5927]
id	Success (pp)	-0.5273	[-0.8255, -0.2291]
id	Regret	+0.0008	[+0.0001, +0.0014]
composition	NF (pp)	-0.0195	[-0.1450, +0.1059]
composition	Success (pp)	+0.1172	[-0.1916, +0.4260]
composition	Regret	-0.0013	[-0.0046, +0.0020]
tight bounds	NF (pp)	+0.1562	[-0.0982, +0.4107]
tight bounds	Success (pp)	-0.1758	[-0.4570, +0.1054]
tight bounds	Regret	+0.0005	[+0.0002, +0.0009]

Table 21: Matched refresh-control costs: means and sample SD over 20 parents. The first bank is included in the three-bank interval, and bank projection is nested within training and queries; these times must not be summed. The frozen-repeat arm intentionally repeats redundant queries to match query counts.

Recorded quantity	Frozen-repeat	Refreshed
Teacher queries	481.950 ± 64.463	481.950 ± 64.463
First query bank (s)	0.023 ± 0.007	0.023 ± 0.007
All three banks (s)	0.072 ± 0.019	0.070 ± 0.016
Training and queries (s)	0.420 ± 0.054	0.425 ± 0.052

Table 22: Structured cost profile: means and sample SD of five seed summaries. Each seed averages three timing rounds. The complete profile, including p95 times and pointwise intervals, is supplied. This is a non-neural comparison.

Actions	Modes	Direct task (ms)	Goal + guard (ms)	Goal + guard regret
8	2	1.602 ± 0.536	1.332 ± 0.237	0.109 ± 0.011
8	4	2.733 ± 0.137	2.931 ± 0.393	0.169 ± 0.033
8	8	5.844 ± 0.789	5.606 ± 1.044	0.185 ± 0.018
16	2	2.635 ± 0.306	2.707 ± 0.228	0.094 ± 0.010
16	4	5.324 ± 0.272	4.706 ± 0.165	0.135 ± 0.021
16	8	10.656 ± 0.812	9.405 ± 0.889	0.162 ± 0.012
32	2	5.565 ± 0.573	5.641 ± 0.550	0.079 ± 0.018
32	4	11.673 ± 1.005	10.468 ± 1.014	0.120 ± 0.008
32	8	25.516 ± 1.723	20.261 ± 2.033	0.150 ± 0.011

B.4 Post-hoc selected-branch execution control

A 32-action diagnostic of the primary allocation projected-head checkpoint motivated a separate engineering control after primary testing had begun. We froze this control before its own execution. It reuses all 20 allocation projected-head checkpoints without retraining or selection and evaluates 128 fresh ID states per checkpoint, with three randomized timing rounds and eight warm-ups per method. Its 23,520 attempts include 480 warm-ups. The same learned gate and raw heads are used in both neural routes. The floating route projects all four heads using the primary implementation and selects one; the exact route selects the raw head first and performs one rational unweighted Euclidean projection onto that selected mode. A fresh weighted task optimizer enumerating all four modes is the third route. Every route uses the same exact final release path.

This comparison changes both arithmetic and the execution algorithm, not precision alone. Native feasibility follows intrinsic projection. Zero shared-guard repairs would therefore mean that one intrinsic projection suffices for release, not solver-free inference or raw neural feasibility. The exact selected-mode projection preserves that mode and minimizes distance to the selected head; it need not minimize the task cost or select the task-optimal mode. Independent auditing replays the checkpoint logits, raw heads, gate, and floating operations on all 2,560 states, verifies rational KKT conditions for exact projections and direct optima, and reconciles serialized payloads, randomization, and per-stage times. These are descriptive post-hoc results on existing trained seeds, separate from the 45 primary contrasts. Their latencies are compared within this run only.

Table 23: Post-hoc allocation execution control: means and sample SD over 20 checkpoints with fresh states. Success uses regret 0.05. All projection and release work is inside the decision interval.

Execution route	Native feasible (%)	Success (%)	Regret	Time (ms)
Floating, all heads	0.08 ± 0.24	5.04 ± 2.66	0.154 ± 0.017	14.858 ± 3.604
Exact, selected head	100.00 ± 0.00	5.04 ± 2.66	0.154 ± 0.017	8.247 ± 1.867
Direct weighted task	100.00 ± 0.00	100.00 ± 0.00	0.000 ± 0.000	12.512 ± 2.914

Table 24: Optimization accounting per attempted action in the post-hoc execution control. Intrinsic and guard work are counted separately; an exactly projected action still incurs optimization.

Execution route	Intrinsic mode solves	Guard mode solves	Guard repair (%)	Coverage (%)
Floating, all heads	4.000 ± 0.000	3.997 ± 0.010	99.92	100.00
Exact, selected head	1.000 ± 0.000	0.000 ± 0.000	0.00	100.00
Direct weighted task	4.000 ± 0.000	0.000 ± 0.000	0.00	100.00

Table 25: Descriptive paired effects for the post-hoc execution control. These pointwise intervals are not members of the predeclared primary family.

Metric	Mean effect	95% interval
Exact, selected head − Floating, all heads		
NF (pp)	+99.9219	[+99.8093, +100.0344]
Success (pp)	+0.0000	[+0.0000, +0.0000]
Regret	-4.63e-09	[-5.71e-09, -3.56e-09]
Latency (ms)	-6.6109	[-7.4975, -5.7242]
Exact, selected head − Direct weighted task		
NF (pp)	+0.0000	[+0.0000, +0.0000]
Success (pp)	-94.9609	[-96.2078, -93.7141]
Regret	+0.1540	[+0.1462, +0.1617]
Latency (ms)	-4.2654	[-4.7906, -3.7402]

B.5 Training, search, and timing costs

The complete study includes 195 parent models and 1170 continuation candidates, including the declared precision controls. All parent, candidate, and teacher costs are retained. The three stage wall intervals are outer intervals; optimization and teacher collection are nested components and must not be added to those totals. Summing trial durations is work accounting, not elapsed calendar time. Teacher query counts and per-teacher solve times are supplied per trial. Equal update counts do not imply equal total compute, and no training-cost amortization advantage is established here.

Table 26: Recorded costs across 20 trials per task. Nested component intervals are listed for transparency and overlap the fit stage. Test-stage wall times may include session interruptions; inference claims use the separate randomized timing stage.

Task	Recorded interval	Mean ± SD (s)	Total (s)
Disconnected 2D	Fit stage	14.41 ± 1.63	288.17
	Test stage	12.46 ± 0.82	249.27
	Timing stage	15.96 ± 0.98	319.27
	Parent optimization	2.22 ± 0.26	44.40
	Continuation optimization	6.90 ± 0.64	138.10
	Teacher collection	1.55 ± 0.43	31.00
Convex 2D	Fit stage	14.31 ± 0.77	286.14
	Test stage	12.06 ± 0.90	241.10
	Timing stage	11.76 ± 4.63	235.28
	Parent optimization	2.27 ± 0.20	45.41
	Continuation optimization	7.02 ± 0.27	140.42
	Teacher collection	0.71 ± 0.24	14.21
Allocation 16D	Fit stage	77.85 ± 9.63	1557.07
	Test stage	103.48 ± 132.96	2069.57
	Timing stage	60.18 ± 17.32	1203.59
	Parent optimization	2.68 ± 0.29	53.62
	Continuation optimization	8.17 ± 0.53	163.34
	Teacher collection	32.92 ± 7.03	658.36

Table 27: Task-only fallbacks selected by validation, out of 20 trials per family. All trial selections are supplied. A fallback is included in that selected family's test estimate.

Task	Soft	Num. L_1	Exact L_1	Exact L_2	Reweight
Disconnected 2D	3	11	4	5	6
Convex 2D	0	9	0	0	0
Allocation 16D	12	13	14	12	4

B.6 Environment and interruption record

The primary manifest records Windows, Python 3.11.9, PyTorch 2.7.1+cu128, NumPy 2.4.6, SciPy 1.17.0, and an Intel64 Family 6 Model 197 processor. Execution uses CPU, one Torch thread, and deterministic Torch algorithms. CUDA package availability is not evidence of GPU execution. Source and versions were rechecked before resuming the session. All 60 selections preceded testing. Fifty-nine test trials finished before the interruption; the final allocation trial's incomplete outputs were preserved separately and that trial alone was rerun with unchanged checkpoint, source, and inputs. The successful final test and separate randomized timing records supply the reported outcomes. An unusually long earlier allocation test-stage wall interval was retained rather than treated as controlled latency. The idle-sleep request used during measurement is temporary and does not change saved power settings. OS scheduling and other numerical-library behavior remain outside Torch's determinism setting.

B.7 Synthetic allocation generation

Jobs are divided into four groups. In each mode, one group has minimum service sampled on the hundredth grid from $[0.30, 0.50]$ and capacity from $[1.00, 1.80]$; other jobs have zero minimum and capacity from $[0.10, 0.30]$. Demands range from 0.10 to 1.70 and positive weights from 0.50 to 2.00 on the same grid. Define $L = \max_k \sum_j l_{kj}$ and $H = \max_k \sum_j h_{kj}$. Budget is $L + \alpha(H - L)$, with $\alpha \in [0.30, 0.75]$ for train/validation/ID and $[0.12, 0.28]$ for the tighter-budget split. This construction keeps every mode nonempty and makes mode capacity tradeoffs observable. RNG streams include seed, dimension, mode count, and split identifier; stored rational features reconstruct the exact constraints. The implementation supports other divisible dimension/mode counts, but the primary learned-policy experiment is specifically 16 actions and four modes.

B.8 Selection, comparisons, and metric interpretation

The primary pairs, written as first minus second, are exact L_1 minus numerical L_1, exact L_1 minus task reweighting, branch exact L_1 minus branch task, direct task optimization minus task, and direct task optimization minus exact L_1. Endpoints are native proposal feasibility, success at regret 0.05, and repeated-round mean decision latency, each on ID. There are $3 \times 5 \times 3 = 45$ simultaneous comparisons. Positive feasibility/success and negative latency favor the first method. Identical observed outcomes can produce a zero-width sample interval; this is not a population-equivalence certificate.

All curve denominators are attempted states, so withholding never improves the success curve by removing failures. Conditional regret summaries are interpreted alongside coverage. Regret thresholds and metric comparisons use exact rationals; plotted values are converted only for display. Selection uses validation metrics and may fall back to continued task training. A selected family mean therefore describes the declared selection procedure, not necessarily a positive correction weight in every trial. Fixed weight-one results are separately labeled. All candidate validation metrics, order hashes, and seeds for regenerating the orders are retained, including unselected auxiliary-weight settings and checkpoint hashes. The compact supplement retains parent, selected, and fixed-weight-one checkpoint tensors; other candidate tensors were inspected by the original full audit and are reproducible from the protocol.

C Historical teacher evidence

The five-trial teacher/geometry study supplies the 1,142 targets audited in Section 4. Its numerical-L_1/exact-L_2 conflict counts cannot isolate projection norm because of the demonstrated target-quality and tie confounds. The new study supplies the policy comparisons reported here. Earlier model comparisons and records remain in the local research archive; the supplement includes the historical inputs needed to check overlap and the teacher evidence needed to reproduce the retained correction-quality claims.

D ARTIFACT, INTERFACES, AND REPRODUCTION

The supplement contains one active code tree, frozen protocols and source snapshots, regression tests, shared data, teacher targets, selected/parent/fixed-weight-one checkpoint states, all validation and decision records, and generated report tables. A manifest binds supplied file hashes. The original full audit checked all candidate states; the compact artifact declares unselected checkpoint omissions and their original hashes. Lossless trace transport restores original JSONL bytes, including number types and line endings. Storage transport changes do not alter any model state, action, or measured time. Follow the README's restoration instructions before inspecting compressed state artifacts if present.

Independent auditing recomputes rational constraints and task regret, validates teacher optimality with separate references, reconstructs validation selection, and checks timing/order inventories. The original branch audit additionally replays gate/head arithmetic and checks retained serialized payloads. A separately labeled recorded-action mode supports inspection on another floating-point platform but does not certify its neural replay. Audits validate internal evidence consistency; they do not authenticate historical clocks or replace an externally witnessed execution.

The implementation rejects malformed parser arrays and excessive nesting, enforces bounded integer/payload sizes, fails closed on invalid or stale bindings, and requires finite validation metrics before a fallback can qualify. The optional demonstration generates fresh synthetic decisions and invokes the real checkpoint and release path. It does not actuate hardware or enforce a hard deadline. Its broader timing interval is not pooled with precompiled benchmark timing. Reproduction scripts write new output directories, retain every attempt, and distinguish fresh execution from report regeneration.