HOP: HARNESS OPTIMIZATION WITH PROVABLE GUARANTEES

Anonymous authors
Paper under double-blind review

ABSTRACT

A language-model agent's harness governs its reasoning and tool use, but its effects on performance depend on complex task interactions. This complexity often leads harness optimization to be treated as a black-box problem. We take a novel view of harness optimization by distinguishing structured harness configurations from unstructured task contexts. Building on this distinction, we introduce **HOP**, a semiparametric framework that models outcomes parametrically in configurations and nonparametrically in task contexts. This structure allows HOP to estimate performance across an exponential configuration space while testing only a linear number of configurations, substantially reducing the data collection burden. Under suitable conditions, HOP's outcome estimators are consistent, asymptotically normal, and asymptotically efficient, and HOP yields consistently optimal harness selection. On τ^3-Bench (Shi et al., 2026), HOP achieves lower regret and fewer constraint violations than leading harness optimization methods under a common evaluation protocol.

1 INTRODUCTION

Language-model agents can perform complex tasks through reasoning and tool use (Yao et al., 2023), yet their capabilities do not consistently translate into reliable execution (Yao et al., 2024). The agent harness—the system that configures models, tools, retrieval, and validation to govern task execution—plays a key role in improving this reliability (Young, 2025).

Reliable harness selection is difficult because the relationship we need to optimize is itself hard to model. Agents process complex, unstructured inputs and interact with diverse environments, making it difficult to characterize how harness choices translate into task outcomes. A natural way forward is to learn this relationship from data: execute agents under different configurations, observe their outcomes, and fit a predictor to guide optimization. Yet this introduces a tension between model flexibility and data availability. Capturing the complexity of agent behavior often calls for flexible, nonparametric predictors, while collecting the data needed to train them is expensive: each agent run may involve multiple model calls and tool interactions. With limited data, flexible models can suffer from high estimation variance, leading to suboptimal or infeasible configuration solutions. This raises a central question: *How can we best balance model flexibility and estimation variance for reliable harness optimization?*

We introduce **HOP**, a semiparametric framework that turns this tension into a modeling principle. Rather than treating the relationship between harness configurations and task outcomes as an unstructured black box, HOP separates structured variation across configurations from complex heterogeneity across task contexts: it models harness effects parametrically while allowing these effects to vary nonparametrically across tasks (see Figure 1 for illustration). This decomposition shares information across configurations, enabling HOP to evaluate configurations with few or no direct observations while retaining flexibility where it matters. Combined with orthogonal estimation and constrained optimization, HOP delivers statistically efficient outcome estimates and reliable configuration selection under reward and resource constraints. Our main contributions are as follows:

- *A novel view of harness optimization*: HOP enables a new way of harness optimization with semiparametric modeling: it captures agent outcomes parametrically in harness configurations and nonparametrically in task contexts. This semiparametric structure preserves

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.

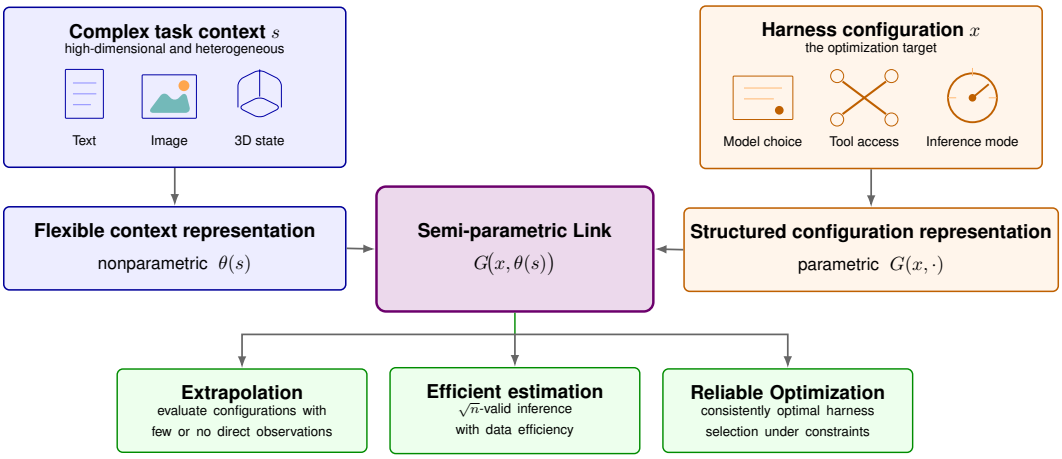


Figure 1: HOP combines a flexible representation of complex task contexts with a structured representation of harness configurations. This semiparametric decomposition retains expressive power across heterogeneous tasks while sharing information across configurations, supporting statistically efficient estimation and reliable constrained configuration selection.

flexibility across complex contexts while controlling estimation variance and simplifying optimization over harness configurations.

- *Provable guarantees*: HOP's semiparametric view also enables provable performance guarantees for constrained harness optimization in complex task environments. Under suitable regularity and feasibility conditions, HOP provides consistent and asymptotically normal outcome estimates and consistently optimal harness selection. Beyond establishing consistency and asymptotic normality, we characterize the semiparametric efficiency bound for expected harness outcomes and prove that HOP attains this bound under conditional homoskedasticity and the stated regularity conditions.
- *Empirical effectiveness*: On τ^3-Bench (Shi et al., 2026), HOP outperforms competing optimization methods, including OPRO (Yang et al., 2024), GEPA (Agrawal et al., 2026), and Meta-Harness (Lee et al., 2026), under a common evaluation protocol, achieving lower objective regret and constraint violation rates. Relative to the strongest overall baseline, pure deep learning, HOP reduces feasibility-conditional regret by 20.3% and lowers the violation rate from 21.25% to 13.44%. These gains require only 800 agent runs covering 10 of 32 configurations, approximately 74% fewer runs than exhaustive evaluation on the full task panel.

2 Related Work

Harness optimization. Early approaches optimize prompts or modular LM programs through LLM-generated proposals, textual feedback, or evolutionary search, including OPRO (Yang et al., 2024), DSPy and MIPRO (Khattab et al., 2023; Opsahl-Ong et al., 2024), TextGrad (Yuksekgonul et al., 2024), Promptbreeder (Fernando et al., 2023), and GEPA (Agrawal et al., 2026). Recent work expands the search space to system prompts (Zhang et al., 2026a), agent programs and architectures (Hu et al., 2025; Wang et al., 2025), evolving contexts and skills (Zhang et al., 2026b; Ye et al., 2026), and executable harness code (Lee et al., 2026; Pan et al., 2026). These approaches focus on generating and improving candidate harnesses using execution feedback. HOP addresses a complementary question: how can limited rollout data be used to reliably estimate harness performance and select configurations under resource constraints? Its semiparametric model shares information across configurations while capturing differences across task contexts. HOP combines this model with statistically efficient estimation to guide constrained harness selection, with guarantees for both outcome estimation and selection consistency under suitable conditions.

Black-box optimization. Harness optimization is widely regarded as a black-box problem: the functional forms relating configurations to expected reward and resource consumption are unknown

and can only be observed through expensive agent runs. Related problems have been studied in algorithm configuration (Hutter et al., 2011) and derivative-free optimization (Rios & Sahinidis, 2013), with Bayesian optimization providing a representative surrogate-based approach (Snoek et al., 2012; Shahriari et al., 2015) and extensions for unknown constraints (Gelbart et al., 2014). Black-box optimization theory provides convergence and regret guarantees under various structural and sequential-sampling assumptions, but these guarantees generally concern the performance of the query policy as new evaluations are acquired. Our setting instead focuses on statistical evaluation and constrained selection from rollout data, providing asymptotically valid inference for configuration outcomes and consistent selection over the finite configuration space under the regularity conditions.

Semiparametric estimation and inference. Our work builds on semiparametric estimation and inference with flexible machine-learning methods (Chernozhukov et al., 2018; Foster & Syrgkanis, 2023; Farrell et al., 2020; 2021; Ye et al., 2025). HOP differs from this literature in three ways. First, to our knowledge, it is the first to bring semiparametric modeling to harness optimization, modeling task-level harness outcomes in a way that accommodates heterogeneous task contexts while preserving structure across configurations. Second, we derive the semiparametric efficiency bound for our model and show that HOP attains it under suitable conditions, providing an efficiency guarantee not previously established in this setting. Third, we connect semiparametric outcome estimation to constrained harness selection and establish consistency guarantees when optimizing expected reward subject to multiple resource constraints.

3 Method and Theory

3.1 Problem Setting

We consider an agent configured by a vector of binary harness variables $x = (x_1, \dots, x_d) \in \mathcal{X}$, where $\mathcal{X} \subseteq \{0,1\}^d$, encoding harness choices such as model selection, tool access, and validation rules. The configuration space $\mathcal{X}$ incorporates logical constraints on these indicators, such as one-hot constraints for mutually exclusive options. The agent is deployed to perform tasks, each characterized by an initial context $S_0 \in \mathcal{S}$, such as a user request and the relevant environment information. Upon observing this context, the agent executes the task under configuration x and produces task-level outcomes $Y_0, \dots, Y_K$, representing performance or resource-use metrics, such as task reward, execution cost, and latency.

For each metric $k = 0, \dots, K$, define the conditional mean

$$m_k(x, s) := \mathbb{E}[Y_k \mid X = x, S_0 = s].$$

The observed outcome can then be decomposed as

$$Y_k = m_k(X, S_0) + U_k, \qquad \mathbb{E}[U_k \mid X, S_0] = 0,$$

where U_k captures outcome variation conditional on the configuration and initial task state. Let P_0 denote the target distribution of initial task states. The expected outcome of configuration x on metric k is

$$\mu_k(x) := \mathbb{E}_{S_0 \sim P_0}[m_k(x, S_0)].$$

Harness optimization seeks a configuration that maximizes an expected objective while satisfying K expected-outcome constraints:

$$\begin{aligned} \max_{x \in \mathcal{X}} \quad & \mu_0(x) \\ \text{subject to} \quad & \mu_k(x) \le b_k, \qquad k = 1, \dots, K, \end{aligned} \tag{1}$$

where b_k is the threshold for constraint k. For example, this formulation allows us to optimize task reward while limiting execution cost and latency, as illustrated in Example 3.1.

Example 3.1 (Reward maximization under resource constraints)**.** *Taking discounted reward as the objective and imposing discounted cost and latency budgets gives the following instance of equation 1:*

$$\max_{x \in \mathcal{X}} \mu_{\text{reward}}(x) \quad \textit{s.t.} \quad \mu_{\text{cost}}(x) \le B_C, \quad \mu_{\text{latency}}(x) \le B_L. \tag{2}$$

We consider this setting in our experiments (Section 4).

3.2 CHALLENGES

The harness optimization problem formulated in Section 3.1 faces three closely related challenges.

1. *Complex task contexts.* For agent tasks, the initial context S_0 can be high-dimensional and unstructured, encompassing text, webpages, images, or representations of three-dimensional environments. The conditional response $m_k(x, s)$ must capture how harness choices interact with these heterogeneous contexts, a relationship that simple parametric models may fail to represent adequately. This complexity also complicates estimation of the population outcome $\mathbb{E}_{S_0 \sim P_0}[m_k(x, S_0)]$, particularly for configurations with limited observations.
2. *Costly evaluation.* Learning these relationships from data can require substantial coverage of both task contexts and harness configurations. Yet each outcome observation requires an agent execution, potentially involving multiple model calls, tool interactions, and environment operations. Standard deep learning and other nonparametric approaches may require many such observations to achieve accurate estimation, limiting their sample efficiency in this setting.
3. *Evaluation error propagation in optimization.* Estimation errors propagate into configuration selection. Errors in the estimated objective can reverse the ranking of candidate configurations, while errors in the estimated constraints can cause infeasible configurations to appear feasible or exclude promising feasible ones. Reliable harness optimization therefore requires estimates that support accurate comparisons and constraint assessment.

These challenges help explain why many recent methods treat the mapping from agent configurations to performance as a black-box objective and guide search using observed scores or execution traces (Yang et al., 2024; Fernando et al., 2023; Agrawal et al., 2026; Hu et al., 2025; Wang et al., 2025; Lee et al., 2026). Although empirically effective, these methods generally do not provide statistical guarantees for outcome estimation, optimal feasible configuration selection, or satisfaction of multiple resource constraints.

3.3 A SEMIPARAMETRIC FRAMEWORK FOR HARNESS OPTIMIZATION

To address these challenges, we propose a semiparametric framework for harness evaluation and optimization. The framework combines flexible modeling of task contexts with a structured representation of harness effects to efficiently estimate the expected outcome of each candidate configuration.

Representation. Our key modeling insight is to distinguish structured harness configurations from unstructured task contexts. We encode this distinction through the semiparametric representation

$$m_k(x, s) = G_k(x, \theta_k^*(s)), \qquad k = 0, \ldots, K, \tag{3}$$

where G_k is a prespecified link function and $\theta_k^* : \mathcal{S} \to \mathbb{R}^{p_k}$ is an unknown function modeled nonparametrically, for example using a neural network. This representation models configuration effects parametrically while allowing them to vary flexibly with task context.

The link function can take various forms, including linear, exponential, multiplicative, and sigmoid specifications. Appropriate choices can capture both individual harness effects and flexible interactions among harness components. For the exponential and sigmoid link functions used in Section 4, Proposition A.1 in Appendix A.1 establishes that the structured representation enables extrapolation from a linear number of observed configurations to an exponentially large configuration space, substantially reducing the data collection burden.

Under equation 3, the expected outcomes in problem equation 1 take the form

$$\mu_k(x) = \mathbb{E}_{S_0 \sim P_0}\left[G_k(x, \theta_k^*(S_0))\right], \qquad k = 0, \ldots, K.$$

We next describe how to estimate these quantities from a finite sample of agent runs.

Evaluation. Fix any $k \in \{0, \ldots, K\}$. We omit the subscript k throughout this part. We consider a dataset of n i.i.d. agent runs, $\mathcal{D} = \{(S_{i,0}, X_i, Y_i)\}_{i=1}^n$, collected from observation or experiments. In each run i, the task context $S_{i,0}$ is drawn from the target distribution P_0, and the harness configuration X_i is selected according to a potentially unknown conditional distribution $\nu(x|s) = \mathbb{P}(X = x \mid S_0 = s)$, which captures how configurations are assigned across task contexts. The agent then executes the task under X_i and produces the outcome Y_i for the selected metric.

Algorithm 1 Cross-Fitted Harness Evaluation

Require: Data $\mathcal{D}$, configuration space $\mathcal{X}$, link function G, function class Θ, and fold count $F \geq 2$.

1: Randomly split the observation indices into folds $I_1, \ldots, I_F$, independently of the data.
2: **for** $f = 1, \ldots, F$ **do**
3: Let $I_{-f} = \{1, \ldots, n\} \setminus I_f$ and fit $\hat{\theta}^{(-f)} \in \arg\min_{\theta \in \Theta} \sum_{i \in I_{-f}} \ell(Y_i, X_i, \theta(S_{i,0}))$.
4: Using only observations in I_{-f}, estimate

$$\hat{\Lambda}^{(-f)}(s) = \widehat{\mathbb{E}}_{-f}\Big[\ell_{\theta\theta}\Big(Y, X, \hat{\theta}^{(-f)}(s)\Big) \mid S_0 = s\Big], \qquad \hat{\eta}^{(-f)} = \Big(\hat{\theta}^{(-f)}, \hat{\Lambda}^{(-f)}\Big).$$

5: For each $i \in I_f$ and $x \in \mathcal{X}$, compute $\hat{\psi}_i(x) = \psi\big(W_i; \hat{\eta}^{(-f)}, x\big)$.
6: **end for**
7: **return** $\hat{\mu}(x) = \frac{1}{n}\sum_{i=1}^{n} \hat{\psi}_i(x), \qquad \text{for} \quad x \in \mathcal{X}.$

Given $\mathcal{D}$, we aim to estimate $\mu(x)$ for each configuration $x \in \mathcal{X}$ and construct valid confidence intervals. Let $W = (S_0, X, Y)$. For a target configuration $\tilde{x}$, define the score function $\varphi(W; \mu, \eta, \tilde{x}) = \psi(W; \eta, \tilde{x}) - \mu(\tilde{x})$, where

$$\psi(W; \eta, \tilde{x}) = G(\tilde{x}, \theta(S_0)) - G_\theta(\tilde{x}, \theta(S_0))^\top \Lambda(S_0)^{-1} \ell_\theta(Y, X, \theta(S_0)), \tag{4}$$

$\eta = (\theta, \Lambda)$ collects the nuisance parameters, $\ell(Y, X, \theta) = \{Y - G(X, \theta)\}^2$ is a squared loss function, $\ell_\theta(Y, X, \theta) = -2\{Y - G(X, \theta)\}G_\theta(X, \theta)$ is its gradient with respect to θ, and $\Lambda(s) = \mathbb{E}[\ell_{\theta\theta}(Y, X, \theta(s)) \mid S_0 = s]$. At the truth, we have $\Lambda^*(s) = \mathbb{E}[\ell_{\theta\theta}(Y, X, \theta^*(s)) \mid S_0 = s] = 2\mathbb{E}\big[G_\theta(X, \theta^*(s))G_\theta(X, \theta^*(s))^\top \mid S_0 = s\big]$. Algorithm 1 returns the estimated outcome $\hat{\mu}(\tilde{x})$ for every candidate configuration.

Optimization. Given the estimated outcomes $\hat{\mu}_0(x), \ldots, \hat{\mu}_K(x)$, we solve

$$\begin{aligned} \max_{x \in \mathcal{X}} \quad & \hat{\mu}_0(x) \\ \text{subject to} \quad & \hat{\mu}_k(x) \leq b_k, \qquad k = 1, \ldots, K. \end{aligned} \tag{5}$$

The computational approach depends on the size of $\mathcal{X}$ and the functional form of the estimates. When $\mathcal{X}$ is finite, Equation 5 can be solved exactly by enumeration. For larger structured configuration spaces, the problem may be solved using mixed-integer optimization. The appropriate formulation depends jointly on the link function and the orthogonal correction: if the resulting $\hat{\mu}_k(x)$ are affine in binary decision variables, the problem is a mixed-integer linear program; otherwise, it generally requires a mixed-integer nonlinear or conic formulation when compatible structure is available (Wolsey, 1998; Belotti et al., 2013; Lubin et al., 2016). These formulations encode the fitted functions directly, without requiring $\hat{\mu}_k(x)$ to be precomputed for every configuration.

3.4 Theoretical Performance Guarantees

In this section, we establish asymptotically valid inference for expected harness outcomes and consistency of the resulting harness optimization. We fix an outcome and suppress its index.

Assumption 3.2 (Regularity Conditions)**.** *Assume the following conditions:*

(A1) ***Sampling and specification.*** *The observations $W_i = (S_{i,0}, X_i, Y_i)$ are i.i.d., $S_0 \sim P_0$, and Equation 3 holds.*

(A2) ***Nuisance Space.*** *There exists a compact and convex set $\mathcal{B} \subset \mathbb{R}^p$ and constants $0 < c < C < \infty$ such that $(\theta^*, \Lambda^*) \in T$, where*

$$T := \left\{ (\theta, \Lambda) \in L^2(P_0; \mathbb{R}^p) \times L^2(P_0; \mathbb{S}^p) : \begin{array}{l} \theta(s) \in \mathcal{B}, \quad cI_p \preceq \Lambda(s) \preceq CI_p \\ \text{for } P_0\text{-almost every } s \end{array} \right\},$$

$\mathbb{S}^p$ denotes the space of symmetric $p \times p$ matrices, and the two $L^2(P_0)$ norms use the Euclidean and Frobenius norms, respectively.

(A3) ***Smoothness.*** *The function $G(x, \cdot)$ is three times continuously differentiable on a neighborhood of $\mathcal{B}$, with G and its first three derivatives uniformly bounded over $\mathcal{X} \times \mathcal{B}$.*

(A4) ***Cross-fitting and nuisance rates.*** *There exist deterministic sequences $\Delta_n > 0$ and $\epsilon_n > 0$, with $\Delta_n \to 0$ and $\epsilon_n = o(n^{-1/4})$, such that $\Pr\left(\hat{\eta}^{(-f)} \in \mathcal{T}_n \text{ for every } f = 1, \ldots, F\right) \geq 1 - \Delta_n$, where*

$$\mathcal{T}_n := \left\{ (\theta, \Lambda) \in T : \begin{array}{l} \|\theta - \theta^*\|_{L^2(P_0)} \leq \epsilon_n, \\ \|\Lambda - \Lambda^*\|_{L^2(P_0)} \leq \epsilon_n \end{array} \right\}.$$

(A5) ***Moments.*** *For some $q > 2$ and $M < \infty$, $\mathbb{E}[|U|^q \mid S_0 = s] \leq M$ for P_0-almost every s. For each fixed $x \in \mathcal{X}$, $\sigma_x^2 := \mathrm{Var}(\psi(W; \eta^*, x)) > 0$, where $\eta^* = (\theta^*, \Lambda^*)$.*

These assumptions impose mild regularity conditions commonly used in double machine learning literature (Chernozhukov et al., 2018). Under these regularity conditions, the following theorem establishes the asymptotic normality of HOP's outcome estimator.

Theorem 3.3 (Asymptotic Normality of Harness Evaluation)**.** *Under Assumption 3.2, for every fixed $x \in \mathcal{X}$,*

$$\sqrt{n}\{\hat{\mu}(x) - \mu(x)\} \xrightarrow{d} N(0, \sigma_x^2),$$

where

$$\sigma_x^2 = \mathrm{Var}\big(G(x, \theta^*(S_0)) + 2G_\theta(x, \theta^*(S_0))^\top [\Lambda^*(S_0)]^{-1} G_\theta(X, \theta^*(S_0))U\big). \tag{6}$$

Furthermore, the cross-fitted variance estimator

$$\hat{\sigma}_x^2 = \frac{1}{n} \sum_{i=1}^{n} \{\hat{\psi}_i(x) - \hat{\mu}(x)\}^2 \xrightarrow{p} \sigma_x^2.$$

Theorem 3.3 implies that $\hat{\mu}(x) \pm z_{1-\alpha/2} \frac{\hat{\sigma}_x}{\sqrt{n}}$ is an asymptotically valid $(1-\alpha)$ confidence interval for each fixed configuration x, where $z_{1-\alpha/2}$ is the standard normal quantile. In Appendix A.3, we prove that, under suitable conditions, σ_x^2 attains the semiparametric efficiency bound, the minimum asymptotic variance among regular estimators. The experiments in Appendix C.8 demonstrate the practical benefits of HOP's efficient inference.

Next, we establish the consistency of harness optimization.

Corollary 3.4 (Consistency of Harness Optimization)**.** *Suppose Assumption 3.2 holds for every $x \in \mathcal{X}$ and $k = 0, \ldots, K$. Let $\mathcal{X}^\star$ be the optimal solution set of equation 1. Assume some $x^\star \in \mathcal{X}^\star$ is strictly feasible: $\mu_k(x^\star) < b_k$ for all $k = 1, \ldots, K$. Then equation 5 is feasible with probability tending to one*[1]*, and any exact optimizer $\hat{x}_n$ satisfies $\Pr(\hat{x}_n \in \mathcal{X}^\star) \to 1$ and $\mu_0(\hat{x}_n) \xrightarrow{p} \mu_0(x^\star)$.*

Remark 3.5 (Inexact Optimization)**.** *When exact optimization is computationally infeasible, let $\mathcal{A}$ be a fixed deterministic solver and $x_{\mathcal{A}} = \mathcal{A}(\mu)$ its output given the true objective and constraint values. If its output is unchanged under sufficiently small perturbations of these values, uniform consistency implies $\Pr\{\mathcal{A}(\hat{\mu}) = x_{\mathcal{A}}\} \to 1$.*

Remark 3.6 (Binding Constraints)**.** *Corollary 3.4 assumes that at least one optimal solution is strictly feasible. If optimal solutions exist but may have binding constraints, we can relax the estimated constraints to $\hat{\mu}_k(x) \leq b_k + \tau_n$, where $\tau_n \to 0$, $\sqrt{n}\,\tau_n \to \infty$. Under the remaining conditions, the same consistency conclusion holds for the relaxed problem; see Appendix A.5.*

4 Experiments

4.1 Experimental Setup

Benchmark and models. We evaluate agents on the `banking_knowledge` domain of τ^3-Bench v1.0.1 (Shi et al., 2026). The benchmark contains 97 tasks, 698 policy documents, and 21 banking product categories. To complete a task, an agent must retrieve the applicable policies, interact with a simulated customer, and execute the required banking operations. Performance is evaluated using the benchmark's task-specific grader. We use DeepSeek-V4-Flash and GLM-5.3 as agent models, both with high reasoning effort and a pinned provider route with no fallback. Following Shi et al. (2026), we fix the customer simulator to `gpt-5.2-2025-12-11` with low reasoning effort. The simulator prompt, environment, task set, and grader are held fixed across all experiments.

[1] When the estimated problem is infeasible, $\hat{x}_n$ may be assigned any value in $\mathcal{X}$.

Harness configurations and outcomes. Each harness configuration is represented by $x = (A, C, R, P, V) \in \{0, 1\}^5$. The factors specify the agent model (A: DeepSeek-V4-Flash or GLM-5.3), tool access (C: progressive discovery or direct access), retrieval (R: BM25 or BM25 plus dense retrieval with reciprocal-rank fusion), pre-action evidence validation (P), and post-action state verification (V). These choices yield 32 configurations, with $x = (0, 0, 0, 0, 0)$ as the reference.

We measure task success, token cost, and estimated generation latency. Task success is the benchmark grader's binary reward. Token cost applies fixed prices to all input and output tokens and includes retrieval-embedding costs. Latency is estimated by dividing total output tokens by the model's median generation rate: 89.005 tokens/s for DeepSeek-V4-Flash and 41.998 tokens/s for GLM-5.3. This proxy captures configuration-dependent generation demand while excluding network, queueing, and provider-load variation (Ding et al., 2024; Srivatsa et al., 2025).

4.2 Estimation and Optimization Protocol

Evaluation design. We maximize discounted reward subject to discounted cost and latency constraints using $\gamma = 0.95$, corresponding to an effective horizon of $1/(1-\gamma) = 20$ decisions and closely matching the observed trajectory lengths (mean 23.0; median 21) (Amit et al., 2020; Zhang & Ross, 2021). We use each configuration's average performance across all 97 benchmark tasks as an empirical approximation to its true expected performance, details of which can be found in Appendix C.1. In each Monte Carlo replicate, the estimator observes 10 prespecified configurations—five per agent model—on a simple random sample of 80 tasks, yielding 800 trajectories. This requires 74% fewer trajectories per replicate than exhaustive evaluation of all 32 configurations on all 97 tasks (3,104 trajectories), and reduces expensive GLM-5.3 evaluations from 1,552 to 400. This design is a task-clustered extension of the sampling model in Section 3.3 and represents settings in which some configurations are unavailable or prohibitively expensive to evaluate. Appendix B provides the corresponding theoretical extension. Because benchmark conclusions can be sensitive to the particular tasks selected (Dehghani et al., 2021), we repeat the complete sampling, estimation, and optimization procedure over 20 independently drawn task subsets and report results averaged across replicates.

Estimation and optimization. We set up HOP with 768-dimensional `gemini-embedding-2` representations of the initial task states (Shanbhogue et al., 2026). The outcome models contain task-dependent main effects for the five configuration factors, allowing estimation of all 32 configurations from the 10 observed configurations. Nuisance functions are estimated using five-fold cross-fitting at the task level (Chernozhukov et al., 2018). For each of the 80 evaluated cost–latency budget pairs, HOP selects the configuration with the highest estimated reward subject to the estimated constraints. Appendix C.2 specifies the estimation and optimization details.

4.3 Baselines

We compare our method with six baselines using the same 800 observed trajectories and evaluate them on the same 80 feasible budget pairs. Linear regression (LR) predicts each outcome from the concatenation of the state embedding and the five binary configuration factors. Pure deep learning (PDL) replaces LR with an unstructured neural network whose architecture and hyperparameters match those of our nuisance model. Neither baseline uses our state-varying coefficient structure or orthogonal correction. Bayesian optimization (BO) fits independent Matérn-$5/2$ Gaussian processes for reward, cost, and latency, and maximizes posterior-mean reward subject to predicted feasibility (Snoek et al., 2012). Finally, OPRO (Yang et al., 2024), GEPA (Agrawal et al., 2026), and Meta-Harness (MH) (Lee et al., 2026) provide prompt-based, reflective evolutionary, and end-to-end harness optimization baselines, respectively. To ensure a fair comparison under the limited rollout budget, we evaluate all methods using the same observed trajectories. We therefore adapt the LLM-based baselines to select from the 32 configurations in a single call, using DeepSeek-V4-Flash as the optimizer LLM and providing the same per-task outcomes and fixed examples. This fixed-data protocol is used only for experimental comparability; in principle, HOP can likewise be deployed iteratively, updating its estimates and configuration choices as additional rollout data become available. Full specifications are provided in Appendix C.5.

Table 1: Constrained-optimization performance. HOP delivers the strongest overall performance, combining lower regret, more reliable feasible selection (higher PCS and PGS), fewer cost and latency overruns, and incurs no optimizer-LLM cost.

Method	Regret ($\times 100$)	PCS	PGS	Violation rate	Cost overrun (%)	Latency overrun (%)	Optimizer cost (USD/draw)
HOP	1.096	45.1%	64.1%	13.44%	0.523%	0.074%	–
LR	1.500	19.9%	57.9%	14.00%†	1.149%	0.078%†	–
PDL	1.375	28.9%	49.2%	21.25%	1.084%	0.114%	–
BO	1.467	4.1%	49.8%	24.06%	0.815%	0.129%†	–
OPRO	3.003	1.2%	11.9%	26.50%	1.753%	0.498%	0.3272
GEPA	2.757	0.4%	10.3%	23.31%	1.874%	0.110%†	0.3276
MH	2.752	1.4%	11.3%	24.00%	2.071%	0.224%	0.3279

Regret is reported in reward percentage points. Overruns are $[C(\widehat{x}) - B_C]_+/B_C$ and $[L(\widehat{x}) - B_L]_+/B_L$, averaged over all selections. For the first six metrics, † denotes no significant paired difference from HOP across the 20 draws ($p \geq 0.05$); every unmarked baseline entry is significantly worse than HOP ($p < 0.05$).

Table 2: Component-ablation results.

Method	Regret ($\times 100$)	PCS	PGS	Violation rate	Cost overrun (%)	Latency overrun (%)
HOP	1.096	45.1%	64.1%	13.44%	0.523%	0.074%
No orthogonal correction	1.471*	14.1%*	53.4%*	20.44%*	0.739%	0.182%*
No cross-fitting	1.067	49.9%	65.9%	12.81%	0.482%	0.094%*
Identity links	1.097	40.0%	62.2%	19.00%*	1.586%*	0.085%

Each variant changes one component of HOP: it removes the orthogonal correction, removes cross-fitting, or replaces the outcome-specific links with identity links. Results are averaged over 80 budget pairs and 20 task draws. * denotes a significant paired difference from HOP ($p < 0.05$, two-sided t-test, $df = 19$); p-values are not adjusted for multiple comparisons.

4.4 Results

4.4.1 Optimization Performance

Table 1 summarizes performance over 80 budget pairs and 20 task draws. For each budget pair, let $r(x)$ denote the full-panel mean reward of configuration x, let $x^{\star}$ denote the reward-maximizing configuration among those satisfying the corresponding cost and latency constraints, and let $\widehat{x}$ denote the configuration selected by the estimation-and-optimization procedure. Regret is computed as $\{r(x^{\star}) - r(\widehat{x})\}$ conditional on $\widehat{x}$ being feasible. The probability of correct selection (PCS) is the probability that $\widehat{x} = x^{\star}$, while the probability of good selection (PGS) is the probability that $\widehat{x}$ is feasible and $r(\widehat{x}) \geq r(x^{\star}) - \epsilon$, with $\epsilon = 0.01$. We also report the frequency of constraint violations, normalized cost and latency overruns, and optimizer-LLM API cost.

As shown in Table 1, HOP achieves the lowest regret and the highest PCS and PGS. Relative to PDL, the strongest overall baseline, HOP reduces regret by 20.3% and improves PCS and PGS by 16.2 and 14.9 percentage points, respectively. All three improvements are statistically significant. HOP also attains the lowest violation rate and cost overrun. Its violation rate is significantly lower than those of PDL, BO, OPRO, GEPA, and MH, but statistically indistinguishable from that of LR. Its cost overrun is significantly lower than that of every baseline. HOP's latency overrun is significantly lower than those of PDL, OPRO, and MH, but statistically indistinguishable from those of LR, BO, and GEPA. Finally, the LLM-based baselines incur \$0.3272–\$0.3279 token cost per draw in optimizer-LLM costs, whereas HOP performs selection locally after model fitting. Complete statistical comparisons appear in Appendix C.6.

4.4.2 Ablation Study

Component ablation. Table 2 isolates three estimator components while holding the design, task draws, and budget grid fixed. The variants remove the orthogonal correction or cross-fitting, or replace the outcome-specific link functions with identity links.

Removing the orthogonal correction significantly worsens regret, PCS, PGS, the violation rate, and latency overrun, producing the largest deterioration in decision quality. Replacing the outcome-specific links with identity links significantly increases the violation rate and cost overrun, although its effects on regret, PCS, and PGS are not significant. Removing cross-fitting has little effect on point-estimate optimization; only the increase in latency overrun is significant. Its primary role is to support valid inference by limiting nuisance-model overfitting (Chernozhukov et al., 2018).

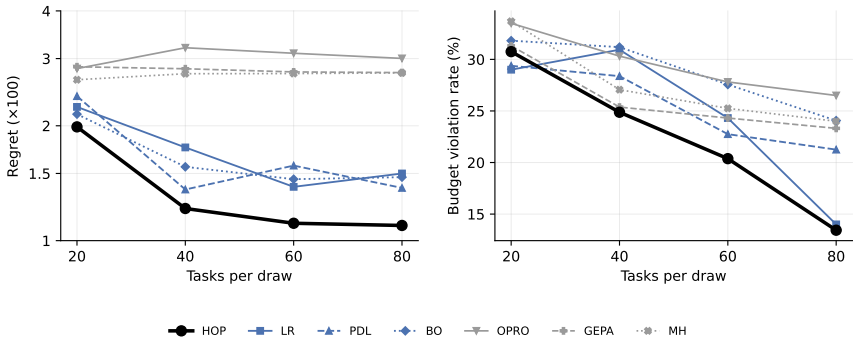


Figure 2: Effect of increasing the number of sampled tasks. HOP uses additional task samples more effectively than the baselines to reduce regret and constraint violations.

Task-sample ablation. Figure 2 varies the number of tasks per draw from 20 to 80 while holding the 10 observed configurations and budget grid fixed. HOP achieves the lowest regret at every sample size and is the only method that improves monotonically on both metrics. Its regret decreases from approximately 1.99 to 1.10 percentage points, while its violation rate falls from 30.8% to 13.4%. In contrast, the baselines' regret generally plateaus or fluctuates as more tasks are added, with the LLM-based selectors showing particularly limited improvement.

At 20 tasks, LR and PDL attain slightly lower violation rates than HOP but substantially higher regret. From 40 tasks onward, HOP achieves the lowest violation rate as well as the lowest regret, and its advantage generally widens with the task sample size. These results indicate that HOP uses additional task-level observations more effectively to improve both objective quality and constraint satisfaction.

5 Conclusion

In this paper, we introduced HOP, a semiparametric framework for harness optimization under limited rollout budgets. HOP combines a compositional outcome model with orthogonal estimation and cross-fitting to share information across configurations, quantify estimation uncertainty, and select configurations under reward, cost, and latency constraints. Under the stated identification and regularity conditions, we establish asymptotically valid outcome inference and consistent constrained selection. Experiments on τ^3-Bench show that HOP uses additional task-level data effectively, reducing regret and constraint violations relative to existing baselines, including when selecting among configurations that were not directly evaluated. These results demonstrate the value of treating harness optimization as both a statistical estimation problem and a constrained decision problem.

AI Use Statement

In this work, we used generative AI tools to assist with mathematical derivations and proofs, improve the clarity and presentation of the manuscript, and create or refine scientific figures. We also used generative AI tools to assist with code debugging and editing. All AI-assisted mathematical arguments, code, experimental results, and written content were independently reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Reproducibility Statement

We provide complete details needed to reproduce our theoretical and empirical results. Appendix A states the assumptions and provides proofs of all theoretical results. Appendix C describes the benchmark, harness configurations, outcome construction, sampling design, model architecture, training procedure, cross-fitting, and optimization protocol. All experimental code and results will be included in the supplementary materials to facilitate full reproduction of our empirical findings.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8479–8565, 2026.

Ron Amit, Ron Meir, and Kamil Ciosek. Discount factor as a regularizer in reinforcement learning. In *International conference on machine learning*, pp. 269–278. PMLR, 2020.

Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, James Luedtke, and Ashutosh Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, 21(1):C1–C68, 2018. doi: 10.1111/ectj.12097.

Harold D Chiang, Kengo Kato, Yukun Ma, and Yuya Sasaki. Multiway cluster robust double/debiased machine learning. *Journal of Business & Economic Statistics*, 40(3):1046–1056, 2022.

Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery, 2021. URL https://arxiv.org/abs/2107.07002.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks Lakshmanan, and Ahmed H Awadallah. Hybrid llm: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations*, volume 2024, pp. 41348–41366, 2024.

Max H Farrell, Tengyuan Liang, and Sanjog Misra. Deep learning for individual heterogeneity. *arXiv preprint arXiv:2010.14694*, 2020.

Max H Farrell, Tengyuan Liang, and Sanjog Misra. Deep neural networks for estimation and inference. *Econometrica*, 89(1):181–213, 2021.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

Dylan J Foster and Vasilis Syrgkanis. Orthogonal statistical learning. *The Annals of Statistics*, 51 (3):879–908, 2023.

Michael A Gelbart, Jasper Snoek, and Ryan P Adams. Bayesian optimization with unknown constraints. *arXiv preprint arXiv:1403.5607*, 2014.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pp. 507–523. Springer, 2011.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T Joshi, Hanna Moazam, et al. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*, 2023.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Miles Lubin, Emre Yamangil, Russell Bent, and Juan Pablo Vielma. Extended formulations in mixed-integer convex programming. In *International Conference on Integer Programming and Combinatorial Optimization*, pp. 102–113. Springer, 2016.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9340–9366, 2024.

Wenbo Pan, Shujie Liu, Chin-Yew Lin, Jingying Zeng, Xianfeng Tang, Xiangyang Zhou, Yan Lu, and Xiaohua Jia. Evolving agents in the dark: Retrospective harness optimization via self-preference. *arXiv preprint arXiv:2606.05922*, 2026.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Luis Miguel Rios and Nikolaos V Sahinidis. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293, 2013.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

Madhuri Shanbhogue, Zhe Li, Shanfeng Zhang, Gustavo Hernández Ábrego, Shih-Cheng Huang, Aashi Jain, Daniel Salz, Sonam Goenka, Chaitra Hegde, Ji Ma, et al. Gemini embedding 2: A native multimodal embedding model from gemini. *arXiv preprint arXiv:2605.27295*, 2026.

Quan Shi, Alexandra Zytek, Pedram Razavi, Karthik Narasimhan, and Victor Barres. tau-knowledge: Evaluating conversational agents over unstructured knowledge. *arXiv preprint arXiv:2603.04370*, 2026.

Jasper Snoek, Hugo Larochelle, and Ryan Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Vikranth Srivatsa, Zijian He, Reyna Abhyankar, Dongming Li, and Yiying Zhang. Preble: Efficient distributed prompt scheduling for llm serving. In *International conference on learning representations*, volume 2025, pp. 37057–37082, 2025.

Wenxiao Wang, Priyatham Kattakinda, and Soheil Feizi. Maestro: Joint graph & config optimization for reliable ai agents. *arXiv preprint arXiv:2509.04642*, 2025.

Laurence A Wolsey. *Integer programming*, volume 42. Wiley Online Library, 1998.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pp. 12028–12068, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. URL https://arxiv.org/abs/2406.12045.

Haoran Ye, Xuning He, Vincent Arak, Haonan Dong, and Guojie Song. Meta context engineering via agentic skill evolution. *arXiv preprint arXiv:2601.21557*, 2026.

Zikun Ye, Zhiqi Zhang, Dennis J. Zhang, Heng Zhang, and Renyu Zhang. Deep learning-based causal inference for large-scale combinatorial experiments: Theory and empirical evidence. *Management Science*, 2025. doi: 10.1287/mnsc.2024.04625. Published online October 15, 2025.

Justin Young. Effective harnesses for long-running agents. Anthropic Engineering, 2025. URL https://www.anthropic.com/engineering/effective-harnesses-for-long-running-agents.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024.

Lechen Zhang, Tolga Ergen, Lajanugen Logeswaran, Moontae Lee, and David Jurgens. Sprig: Improving large language model performance by system prompt optimization. In *International Conference on Learning Representations*, volume 2026, pp. 26649–26672, 2026a.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, volume 2026, pp. 86069–86100, 2026b.

Yiming Zhang and Keith W Ross. On-policy deep reinforcement learning for the average-reward criterion. In *International Conference on Machine Learning*, pp. 12535–12545. PMLR, 2021.

A Supplementary Materials for Section 3.4

A.1 Proposition A.1 and the Proof

Proposition A.1 (Extrapolation to Unobserved Configurations)**.** *Suppose the harness configuration is $x = (x_1, \ldots, x_d) \in \{0,1\}^d$ and the outcome model takes the form*

$$m(x, s) = G\left(\theta_0(s) + \sum_{j=1}^{d} \theta_j(s) x_j\right),$$

where G is an invertible link function. If HOP observes $d+1$ configurations whose vector $(1, x_1, \ldots, x_d)$ are linearly independent, then $\theta(s) = (\theta_0(s), \ldots, \theta_d(s))$ is uniquely determined. Consequently, HOP can extrapolate $m(x, s)$ to every $x \in \{0,1\}^d$, including configurations that are never directly observed. Thus, $d+1$ observed configurations are sufficient to extrapolate outcomes over a configuration space containing up to 2^d configurations.

Proof. Let $x^{(1)}, \ldots, x^{(d+1)}$ denote the observed configurations and define

$$\Phi = \begin{pmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \vdots & & \vdots \\ 1 & x_1^{(d+1)} & \cdots & x_d^{(d+1)} \end{pmatrix}.$$

Because G is invertible, the observed conditional outcomes satisfy

$$\begin{pmatrix} G^{-1}(m(x^{(1)}, s)) \\ \vdots \\ G^{-1}(m(x^{(d+1)}, s)) \end{pmatrix} = \Phi\, \theta(s).$$

By assumption, Φ has full rank and is therefore invertible, so

$$\theta(s) = \Phi^{-1} \begin{pmatrix} G^{-1}(m(x^{(1)}, s)) \\ \vdots \\ G^{-1}(m(x^{(d+1)}, s)) \end{pmatrix}.$$

Hence $\theta(s)$ is uniquely recovered from the $d+1$ observed configurations. For any unobserved configuration x,

$$m(x, s) = G\left(\theta_0(s) + \sum_{j=1}^{d} \theta_j(s) x_j\right),$$

so its outcome can be extrapolated directly. □

Remark A.2. *The proposition applies directly to the link functions used in our experiments: $G(u) = \exp(u)$ for cost and latency, and $G(u) = (1 + \exp(-u))^{-1}$ for reward.*

A.2 Proof of Theorem 3.3

Proof. Fix $\tilde{x} \in \mathcal{X}$, write $m^* = \mu(\tilde{x})$, and define $\varphi(W; m, \eta) = \psi(W; \eta, \tilde{x}) - m$. We aim to apply Theorems 3.1 and 3.2 in Chernozhukov et al. (2018). Thus, we verify Assumptions 3.1(a)–(e) and 3.2(a)–(d) of Chernozhukov et al. (2018) under our Assumption 3.2. All numbered conditions below refer to their paper. We use n for the total sample size and F for the number of folds, corresponding to their N and K. All expectations are under the distribution P. Throughout this proof, C_0 denotes a single fixed positive constant used in the bounds below. Each such bound can first be established with its own finite constant. Then we choose C_0 sufficiently large to dominate all these constants and keep its value unchanged throughout the proof. By convexity of $\mathcal{B}$ and the matrix constraints, T is a convex subset of a normed vector space. Define $B(X, S_0; \eta) := 2G_\theta(\tilde{x}, \theta(S_0))^\top \Lambda(S_0)^{-1} G_\theta(X, \theta(S_0))$, we have

$$\psi(W; \eta, \tilde{x}) = G(\tilde{x}, \theta(S_0)) + B(X, S_0; \eta)\{Y - G(X, \theta(S_0))\}.$$

Assumption 3.1(a): Moment condition. Since $\mathbb{E}[U \mid X, S_0] = 0$,

$$\mathbb{E}[\varphi(W; m^*, \eta^*)] = \mathbb{E}[G(\tilde{x}, \theta^*(S_0))] - m^* + \mathbb{E}[B(X, S_0; \eta^*)U] = 0.$$

Assumption 3.1(b): Linear score function. The score has the required representation $\varphi(W; m, \eta) = \varphi^a(W; \eta)m + \varphi^b(W; \eta)$, where

$$\varphi^a(W; \eta) = -1, \qquad \varphi^b(W; \eta) = \psi(W; \eta, \tilde{x}).$$

Assumption 3.1(c): Twice continuous Gâteaux differentiability. Fix $m \in \mathbb{R}$ and any $\eta, \bar{\eta} \in T$. Consider the path

$$\eta_t = (1-t)\eta + t\bar{\eta} = (\theta_t, \Lambda_t), \qquad t \in [0, 1].$$

By convexity of $\mathcal{B}$ and the matrix constraints defining T, this path remains in T. Its direction $\bar{\eta} - \eta$ is essentially bounded because $\mathcal{B}$ is compact by (A2) and the candidate matrices are uniformly bounded by the definition of T. Condition (A3) ensures that G is three times continuously differentiable in θ, with uniformly bounded derivatives up to order three. Since the score involves G and G_θ, its dependence on θ is twice continuously differentiable. Moreover, the definition of T gives $\|\Lambda_t(s)^{-1}\|_2 \leq c^{-1}$, P_0-almost everywhere, uniformly in t. Smoothness of matrix inversion therefore implies that the score is twice continuously differentiable along the path.

By (A1), $Y = G(X, \theta^*(S_0)) + U$. Combining this representation with the preceding bounds yields

$$\left| \frac{d^j}{dt^j} \varphi(W; m, \eta_t) \right| \leq C_0(1 + |U|), \qquad j = 1, 2,$$

almost surely, uniformly in $t \in [0, 1]$, for a finite constant C_0. By (A5), $\mathbb{E}|U| \leq M^{1/q} < \infty$. The common bound $C_0(1 + |U|)$ therefore has finite expectation and controls both derivatives for every t. Thus, it holds that

$$\frac{d^j}{dt^j} \mathbb{E}[\varphi(W; m, \eta_t)] = \mathbb{E}\left[\frac{d^j}{dt^j} \varphi(W; m, \eta_t) \right], \qquad j = 1, 2.$$

The expectations are continuous by the dominated convergence theorem.

Assumption 3.1(d): Neyman orthogonality. For $\eta = (\theta, \Lambda) \in T$, let

$$h = \theta - \theta^*, \qquad H = \Lambda - \Lambda^*, \qquad \eta_r = (\theta^* + rh, \Lambda^* + rH).$$

The path lies in T for $r \in [0, 1]$ by convexity. Interchanging differentiation and expectation, as justified in the verification of Assumption 3.1(c), we obtain

$$\begin{aligned} \frac{d}{dr} \mathbb{E}[\varphi(W; m^*, \eta_r)] \Big|_{r=0} &= \mathbb{E}\Big[G_\theta(\tilde{x}, \theta^*(S_0))^\top h(S_0) - B(X, S_0; \eta^*) G_\theta(X, \theta^*(S_0))^\top h(S_0) \Big] \\ &= \mathbb{E}\Big[G_\theta(\tilde{x}, \theta^*(S_0))^\top h(S_0) - G_\theta(\tilde{x}, \theta^*(S_0))^\top [\Lambda^*(S_0)]^{-1} \Lambda^*(S_0) h(S_0) \Big] \\ &= 0. \end{aligned}$$

The derivative of $B(X, S_0; \eta_r)$ at $r = 0$ multiplies U and has zero expectation by $\mathbb{E}[U \mid X, S_0] = 0$ in (A1). The second equality follows by conditioning on S_0 and using

$$\Lambda^*(S_0) = 2\mathbb{E}\big[G_\theta(X, \theta^*(S_0)) G_\theta(X, \theta^*(S_0))^\top \mid S_0 \big].$$

Thus exact Neyman orthogonality holds, and $\lambda_n = 0$.

Assumption 3.1(e): Identification. This assumption holds directly by $J_0 = \mathbb{E}[\varphi^a(W; \eta^*)] = -1$.

Assumption 3.2(a): Nuisance realization set. Clearly, $\eta^* \in \mathcal{T}_n$. This assumption follows directly by (A4).

Assumption 3.2(b): Uniform moment bounds. Uniform boundedness of G, G_θ (A3), and Λ^{-1} (A2) gives

$$|\varphi(W; m^*, \eta)| \le C_0(1 + |U|), \qquad \eta \in \mathcal{T}_n,$$

where C_0 is independent of n and η. Since $\mathbb{E}[|U|^q \mid S_0] \le M$ for some $q > 2$,

$$m_n := \sup_{\eta \in \mathcal{T}_n} \|\varphi(\cdot; m^*, \eta)\|_{L^q(P)} \le C_0(1 + M^{1/q}),$$
$$m'_n := \sup_{\eta \in \mathcal{T}_n} \|\varphi^a(\cdot; \eta)\|_{L^q(P)} = 1.$$

Both quantities are bounded by a common finite constant. Taking $c_1 = \max\{1, C_0(1 + M^{1/q})\}$, we obtain $m_n \le c_1$ and $m'_n \le c_1$. Thus Assumption 3.2(b) holds.

Assumption 3.2(c): Statistical rates. We verify its three rate requirements separately.

The rate r_n. Because $\varphi^a(W; \eta) = -1$ for every η, it follows directly that $r_n = 0$.

The rate r'_n. The identity $\Lambda^{-1} - (\Lambda^*)^{-1} = -\Lambda^{-1}(\Lambda - \Lambda^*)(\Lambda^*)^{-1}$ and the bounded derivatives of G imply $|\varphi(W; m^*, \eta) - \varphi(W; m^*, \eta^*)| \le C_0(1 + |U|)\{\|h(S_0)\| + \|H(S_0)\|_{\mathrm{F}}\}$. Under (A5), we have $\mathbb{E}[U^2 \mid S_0] \le M^{2/q}$ almost surely. Then we obtain $r'_n := \sup_{\eta \in \mathcal{T}_n} \|\varphi(\cdot; m^*, \eta) - \varphi(\cdot; m^*, \eta^*)\|_{L^2(P)} \le C_0 \epsilon_n$.

The rate λ'_n. For $\eta \in \mathcal{T}_n$, use the path η_r defined above and write

$$f_\eta(r) = \mathbb{E}[\varphi(W; m^*, \eta_r)],$$

and

$$D_r = G(X, \theta^*(S_0)) - G(X, \theta^*(S_0) + rh(S_0)).$$

Since $\mathbb{E}[U \mid X, S_0] = 0$,

$$f_\eta(r) = \mathbb{E}[G(\tilde{x}, \theta^*(S_0) + rh(S_0)) - m^* + B_{\eta_r}(X, S_0) D_r]\,.$$

Let

$$a = \|h(S_0)\|, \qquad b = \|H(S_0)\|_{\mathrm{F}}, \qquad A_r = (\Lambda^*(S_0) + rH(S_0))^{-1}.$$

Matrix differentiation gives

$$A'_r = -A_r H A_r, \qquad A''_r = 2A_r H A_r H A_r.$$

Together with the bounded derivatives of G, these identities give the following bounds, uniformly in $r \in [0, 1]$:

$$|B_{\eta_r}| \le C_0, \qquad |\partial_r B_{\eta_r}| \le C_0(a + b), \qquad |\partial_r^2 B_{\eta_r}| \le C_0(a + b)^2,$$
$$|D_r| \le C_0, \qquad |\partial_r D_r| \le C_0 a, \qquad |\partial_r^2 D_r| \le C_0 a^2.$$

The second derivative of the first term in $f_\eta(r)$ is bounded by $C_0 a^2$. Applying the product rule to $B_{\eta_r} D_r$ therefore yields

$$\sup_{r \in [0,1]} |f''_\eta(r)| \le C_0 \mathbb{E}[(a + b)^2]$$
$$\le C_0 \left(\|h\|_{L^2(P_0)} + \|H\|_{L^2(P_0)}\right)^2.$$

Consequently,

$$\lambda'_n := \sup_{\eta \in \mathcal{T}_n,\ r \in (0,1)} |f''_\eta(r)| \le C_0 \epsilon_n^2.$$

Choose

$$\delta_n = C_1\left(n^{-1/2} + \epsilon_n + \sqrt{n}\,\epsilon_n^2\right)$$

for a sufficiently large fixed constant C_1. Then $\delta_n \ge n^{-1/2}$, $\delta_n \to 0$, and

$$r_n \le \delta_n, \qquad r'_n \le \delta_n, \qquad \lambda'_n \le \delta_n / \sqrt{n}.$$

Thus all three requirements of Assumption 3.2(c) hold.

Assumption 3.2(d): Nondegenerate score variance. Since $\mathbb{E}[\varphi(W; m^*, \eta^*)] = 0$,

$$\mathbb{E}[\varphi(W; m^*, \eta^*)^2] = \mathrm{Var}(\psi(W; \eta^*, \tilde{x})) = \sigma_{\tilde{x}}^2 > 0.$$

For the fixed distribution P and configuration $\tilde{x}$, taking $c_0 = \min\{1, \sigma_{\tilde{x}}^2\} > 0$ gives $\mathbb{E}[\varphi(W; m^*, \eta^*)^2] \geq c_0$. Thus Assumption 3.2(d) holds.

All conditions of Chernozhukov et al. (2018) are therefore satisfied, taking their model class to be the singleton $\{P\}$. □

A.3 Semiparametric Efficiency of HOP

Fix an outcome Y and a configuration $\tilde{x} \in \mathcal{X}$. Let $\mathcal{M}$ be the collection of joint distributions Q of (S_0, X, Y) that have densities with respect to a common reference measure and satisfy $\mathbb{E}_Q[Y \mid X, S_0] = G(X, \theta_Q(S_0))$, where θ_Q denotes the parameter function associated with Q. Apart from this conditional mean restriction, the distribution of (S_0, X) and the conditional distribution of Y are unrestricted. For each $Q \in \mathcal{M}$, define $\mu_Q(\tilde{x}) = \mathbb{E}_Q[G(\tilde{x}, \theta_Q(S_0))]$. The true distribution P belongs to $\mathcal{M}$, with $\theta_P = \theta^*$. Write $U = Y - G(X, \theta^*(S_0))$.

Assumption A.3 (Local model flexibility)**.** *The tangent space of $\mathcal{M}$ at P is the $L^2(P)$-closure of all mean-zero, square-integrable directions satisfying the linearized conditional mean restriction. As Q varies locally around P within $\mathcal{M}$, local directions for θ_Q are unrestricted subject to square integrability. Along every regular submodel $\{Q_t\} \subset \mathcal{M}$ with $Q_0 = P$, the map $t \mapsto \theta_{Q_t}$ is differentiable at zero in $L^2(P_0; \mathbb{R}^p)$, and differentiation may be interchanged with the expectations defining the conditional mean and the target $\mu_{Q_t}(\tilde{x})$.*

Assumption A.4 (Equal noise variance across configurations)**.** *Conditional on S_0, the noise variance does not depend on the configuration X:*

$$\mathbb{E}_P[U^2 \mid X, S_0] = v(S_0), \qquad 0 < \underline{v} \leq v(S_0) \leq \overline{v} < \infty \quad \textit{almost surely}.$$

Theorem A.5 (Semiparametric Efficiency under Conditional Homoskedasticity)**.** *Suppose Assumptions 3.2, A.3, and A.4 hold. Then the cross-fitted estimator $\hat{\mu}(\tilde{x})$ is semiparametrically efficient for $\mu_P(\tilde{x})$ at P relative to $\mathcal{M}$. Its efficient influence function is*

$$\varphi_{\mathrm{eff}}(W; \tilde{x}) = G(\tilde{x}, \theta^*(S_0)) - \mu_P(\tilde{x}) + 2G_\theta(\tilde{x}, \theta^*(S_0))^\top [\Lambda^*(S_0)]^{-1} G_\theta(X, \theta^*(S_0))U,$$

and the efficiency bound is

$$V_{\mathrm{eff}}(\tilde{x}) = \mathrm{Var}_P\big(G(\tilde{x}, \theta^*(S_0))\big) + 2\mathbb{E}_P\big[v(S_0)G_\theta(\tilde{x}, \theta^*(S_0))^\top [\Lambda^*(S_0)]^{-1} G_\theta(\tilde{x}, \theta^*(S_0))\big].$$

In particular,

$$\sqrt{n}\{\hat{\mu}(\tilde{x}) - \mu_P(\tilde{x})\} \xrightarrow{d} \mathcal{N}(0, V_{\mathrm{eff}}(\tilde{x})).$$

Proof. All unindexed expectations are under the true distribution P. For brevity, write

$$g_x(s) = G_\theta(x, \theta^*(s)), \qquad a(s) = G(\tilde{x}, \theta^*(s)) - \mu_P(\tilde{x}).$$

Here $g_x(s)$ is the gradient of the conditional mean, and $a(s)$ is the centered conditional mean at the target configuration. Recall that

$$\Lambda^*(s) = 2\mathbb{E}[g_X(s)g_X(s)^\top \mid S_0 = s].$$

Define the residual correction and the candidate influence function by

$$r(W) = 2g_{\tilde{x}}(S_0)^\top [\Lambda^*(S_0)]^{-1} g_X(S_0)U, \qquad \varphi(W) = a(S_0) + r(W).$$

By the definition of $\mu_P(\tilde{x})$, $\mathbb{E}[a(S_0)] = 0$. Also, $\mathbb{E}[U \mid X, S_0] = 0$ implies $\mathbb{E}[r(W) \mid X, S_0] = 0$. Thus $\mathbb{E}[\varphi(W)] = 0$, and the assumed bounds give $\mathbb{E}[\varphi(W)^2] < \infty$.

Step 1: Verify the influence-function identity. We show that

$$\frac{d}{dt}\mu_{Q_t}(\tilde{x})\bigg|_{t=0} = \mathbb{E}[\varphi(W)b(W)]$$

for every regular submodel $\{Q_t\} \subset \mathcal{M}$ with $Q_0 = P$, where b is its score.

Specifically, let $q_t = dQ_t/d\nu$. For differentiable density paths, the score is

$$b(w) = \frac{\partial}{\partial t} \log q_t(w)\Big|_{t=0} = \frac{\partial_t q_t(w)|_{t=0}}{q_0(w)} \qquad P\text{-almost surely.}$$

More generally, it is defined through differentiability in quadratic mean.[2] It satisfies $\mathbb{E}[b(W)] = 0$ and $\mathbb{E}[b(W)^2] < \infty$. Write

$$h(s) = \frac{d}{dt}\theta_{Q_t}(s)\Big|_{t=0}$$

for the corresponding derivative of the parameter function.

Starting from $\mu_{Q_t}(\tilde{x}) = \mathbb{E}_{Q_t}[G(\tilde{x}, \theta_{Q_t}(S_0))]$, Assumption A.3 allows us to differentiate both the distribution and the integrand:

$$\begin{aligned}\frac{d}{dt}\mu_{Q_t}(\tilde{x})\Big|_{t=0} &= \mathbb{E}[G(\tilde{x}, \theta^*(S_0))b(W)] + \mathbb{E}[g_{\tilde{x}}(S_0)^\top h(S_0)] \\ &= \mathbb{E}[a(S_0)b(W)] + \mathbb{E}[g_{\tilde{x}}(S_0)^\top h(S_0)].\end{aligned}$$

The last equality uses $\mathbb{E}[b(W)] = 0$. Since $\varphi(W) = a(S_0) + r(W)$, it remains to show that the second term equals $\mathbb{E}[r(W)b(W)]$.

To do so, differentiate the conditional mean restriction $\mathbb{E}_{Q_t}[Y \mid X, S_0] = G(X, \theta_{Q_t}(S_0))$. Writing the conditional density as the joint density divided by its (X, S_0) marginal density and applying the quotient rule gives

$$\begin{aligned}g_X(S_0)^\top h(S_0) = \frac{d}{dt}\mathbb{E}_{Q_t}[Y \mid X, S_0]\Big|_{t=0} \\ = \mathbb{E}[Yb(W) \mid X, S_0] - \mathbb{E}[Y \mid X, S_0]\mathbb{E}[b(W) \mid X, S_0] \\ = \mathbb{E}[Ub(W) \mid X, S_0],\end{aligned} \tag{7}$$

where the last equality uses $U = Y - G(X, \theta^*(S_0))$.

Multiplying by $g_X(S_0)$ and conditioning on S_0 yields

$$\mathbb{E}[g_X(S_0)Ub(W) \mid S_0] = \mathbb{E}[g_X(S_0)g_X(S_0)^\top \mid S_0]h(S_0) = \frac{\Lambda^*(S_0)}{2}h(S_0).$$

Hence, by the definition of r,

$$\begin{aligned}\mathbb{E}[r(W)b(W)] &= 2\mathbb{E}\big[g_{\tilde{x}}(S_0)^\top[\Lambda^*(S_0)]^{-1}\mathbb{E}[g_X(S_0)Ub(W) \mid S_0]\big] \\ &= \mathbb{E}[g_{\tilde{x}}(S_0)^\top h(S_0)].\end{aligned}$$

Substituting this into the target derivative gives

$$\frac{d}{dt}\mu_{Q_t}(\tilde{x})\Big|_{t=0} = \mathbb{E}[a(S_0)b(W)] + \mathbb{E}[r(W)b(W)] = \mathbb{E}[\varphi(W)b(W)],$$

as required.

Step 2: Establish efficiency. We show that φ also belongs to the tangent space. Since $\mathbb{E}[U \mid X, S_0] = 0$, Assumption A.4 gives

$$\begin{aligned}\mathbb{E}[U\varphi(W) \mid X, S_0] &= 2v(S_0)g_{\tilde{x}}(S_0)^\top[\Lambda^*(S_0)]^{-1}g_X(S_0) \\ &= g_X(S_0)^\top h_\varphi(S_0),\end{aligned}$$

where

$$h_\varphi(s) = 2v(s)[\Lambda^*(s)]^{-1}g_{\tilde{x}}(s).$$

The assumed bounds make h_φ square-integrable. Hence φ is a mean-zero, square-integrable direction satisfying the linearized conditional mean restriction. Assumption A.3 places φ in the tangent space.

[2]The score b satisfies $\|\sqrt{q_t} - \sqrt{q_0} - \frac{t}{2}b\sqrt{q_0}\|_{L^2(\nu)} = o(|t|)$.

Together with Step 1, this establishes that φ is the efficient influence function. Indeed, any other influence function differs from φ by a function orthogonal to the tangent space, and therefore has variance at least $\mathbb{E}[\varphi(W)^2]$.

Because $\mathbb{E}[r(W) \mid S_0] = 0$, the two terms $a(S_0)$ and $r(W)$ are uncorrelated. Using the conditional variance assumption and the identity for Λ^*, the efficiency bound is

$$V_{\text{eff}}(\tilde{x}) = \mathbb{E}[\varphi(W)^2] = \operatorname{Var}\big(G(\tilde{x}, \theta^*(S_0))\big) + 2\mathbb{E}\big[v(S_0) g_{\tilde{x}}(S_0)^\top [\Lambda^*(S_0)]^{-1} g_{\tilde{x}}(S_0)\big] .$$

Steps 1–2 show that φ is the efficient influence function. By the standard characterization of efficient estimators, this expansion implies that $\hat{\mu}(\tilde{x})$ is regular at P and attains the semiparametric efficiency bound. □

A.4 Proof of Corollary 3.4

Proof. By Theorem 3.3, each $\hat{\mu}_k(x)$ is consistent. As a result,

$$\max_{0 \le k \le K} \max_{x \in \mathcal{X}} |\hat{\mu}_k(x) - \mu_k(x)| \xrightarrow{p} 0.$$

Strict feasibility implies that $x^\star$ is feasible for the estimated problem with probability tending to one. Every truly infeasible configuration violates at least one constraint by a positive amount, while every feasible nonoptimal configuration has a strictly smaller objective value than $x^\star$. Because $\mathcal{X}$ is finite, the uniform convergence above implies that, with probability tending to one, all infeasible configurations are excluded and all feasible nonoptimal configurations have estimated objective values below that of $x^\star$. On this event, the estimated problem is feasible and every exact optimizer belongs to $\mathcal{X}^\star$. Hence $\Pr(\hat{x}_n \in \mathcal{X}^\star) \to 1$. The conclusion follows. □

A.5 Optimization with Vanishing Constraint Relaxation

To accommodate optimal configurations with binding constraints, consider

$$\max_{x \in \mathcal{X}} \hat{\mu}_0(x) \quad \text{subject to} \quad \hat{\mu}_k(x) \le b_k + \tau_n, \qquad k = 1, \dots, K, \tag{8}$$

where $\tau_n > 0$ is a deterministic tolerance tending to zero. Let $\hat{x}_n^\tau$ denote any exact optimizer, with an arbitrary value in $\mathcal{X}$ assigned when the estimated problem is infeasible. Suppose the true feasible set $\mathcal{F} = \{x \in \mathcal{X} : \mu_k(x) \le b_k,\ k = 1, \dots, K\}$ is nonempty and define

$$\Delta_n = \max_{0 \le k \le K} \max_{x \in \mathcal{X}} |\hat{\mu}_k(x) - \mu_k(x)|.$$

Theorem A.6 (Consistency under Constraint Relaxation)**.** *If $\tau_n \to 0$ and $\Delta_n = o_p(\tau_n)$, then the relaxed feasible set*

$$\hat{\mathcal{F}}_n^\tau = \{x \in \mathcal{X} : \hat{\mu}_k(x) \le b_k + \tau_n,\ k = 1, \dots, K\}$$

satisfies

$$\Pr(\hat{\mathcal{F}}_n^\tau = \mathcal{F}) \longrightarrow 1.$$

Moreover, we have

$$\Pr(\hat{x}_n^\tau \in \mathcal{X}^\star) \longrightarrow 1, \text{ and} \qquad \mu_0(\hat{x}_n^\tau) \xrightarrow{p} \max_{x \in \mathcal{F}} \mu_0(x).$$

Proof. Since $\Delta_n = o_p(\tau_n)$, the event $\Delta_n \le \tau_n$ has probability tending to one. On this event, every $x \in \mathcal{F}$ satisfies $\hat{\mu}_k(x) \le \mu_k(x) + \Delta_n \le b_k + \tau_n$, so $\mathcal{F} \subseteq \hat{\mathcal{F}}_n^\tau$.

If $\mathcal{X} \setminus \mathcal{F}$ is nonempty, finiteness implies a positive minimum violation

$$\rho = \min_{x \in \mathcal{X} \setminus \mathcal{F}} \max_{1 \le k \le K} \{\mu_k(x) - b_k\} > 0.$$

Whenever $\Delta_n + \tau_n < \rho$, every infeasible configuration violates at least one relaxed constraint, thereby $\hat{\mathcal{F}}_n^\tau \subseteq \mathcal{F}$. Because $\Delta_n + \tau_n \xrightarrow{p} 0$, this event also has probability tending to one. If $\mathcal{X} \setminus \mathcal{F} = \varnothing$, then $\mathcal{F} = \mathcal{X}$, and hence $\hat{\mathcal{F}}_n^\tau \subseteq \mathcal{X} = \mathcal{F}$ by the definition of $\hat{\mathcal{F}}_n^\tau$. Thus, $\Pr(\hat{\mathcal{F}}_n^\tau = \mathcal{F}) \to 1$.

If $\mathcal{F} \setminus \mathcal{X}^\star$ is nonempty, the objective gap

$$\gamma = \min_{x \in \mathcal{F} \setminus \mathcal{X}^\star} \left\{ \max_{z \in \mathcal{F}} \mu_0(z) - \mu_0(x) \right\}$$

is strictly positive. On the event $\hat{\mathcal{F}}_n^\tau = \mathcal{F}$ and $2\Delta_n < \gamma$, no suboptimal configuration can maximize the estimated objective. Hence $\hat{x}_n^\tau \in \mathcal{X}^\star$ with probability tending to one. If every feasible configuration is optimal, the same conclusion follows directly from feasible-set recovery. The remaining claims follow. □

Under pointwise root-n consistency for every metric and configuration, finiteness implies $\Delta_n = O_p(n^{-1/2})$. It therefore suffices to choose $\tau_n \to 0$ with $\sqrt{n}\tau_n \to \infty$, for example $\tau_n = n^{-1/4}$.

B Experiment Theory

The main theory considers one configuration per task draw. Our experiment observes a fixed set $\mathcal{A}$ of M configurations for each task. Let

$$W_q = (S_q, \{Y_{q,a} : a \in \mathcal{A}\}), \qquad q = 1, \ldots, Q,$$

where tasks are independent, while outcomes may be arbitrarily dependent within a task. Define

$$\bar{\ell}_{\theta,q}(\theta) = \frac{1}{M} \sum_{a \in \mathcal{A}} \ell_\theta(Y_{q,a}, a, \theta)$$

and

$$\Lambda_{\mathcal{A}}(s) = \mathbb{E}\left[\frac{1}{M} \sum_{a \in \mathcal{A}} \ell_{\theta\theta}(Y_a, a, \theta(s)) \,\middle|\, S = s \right].$$

The task-level orthogonal score is

$$\psi_c(W_q; \eta, x) = G(x, \theta(S_q)) - G_\theta(x, \theta(S_q))^\top \Lambda_{\mathcal{A}}(S_q)^{-1} \bar{\ell}_{\theta,q}(\theta(S_q)).$$

Cross-fitting is performed by task: all trajectories from the same task are kept in the same fold. The estimator and variance estimator are

$$\widehat{\mu}(x) = \frac{1}{Q} \sum_{q=1}^{Q} \widehat{\psi}_{c,q}(x), \qquad \widehat{\sigma}_x^2 = \frac{1}{Q} \sum_{q=1}^{Q} \{\widehat{\psi}_{c,q}(x) - \widehat{\mu}(x)\}^2.$$

Under the conditions of Theorem 3.3, with Q replacing n, fixed M, and uniformly nonsingular $\Lambda_{\mathcal{A}}(s)$,

$$\sqrt{Q}\{\widehat{\mu}(x) - \mu(x)\} \rightsquigarrow N(0, \sigma_x^2).$$

Thus, within-task dependence is captured by the task-level score and its variance. The result follows from the same orthogonality and cross-fitting argument as in the main theorem, applied to independent task clusters; see Chiang et al. (2022) for cluster-robust cross-fitting and variance estimation. Identification of unevaluated configurations additionally requires the observed configurations to provide sufficient variation for $\Lambda_{\mathcal{A}}(s)$ to be nonsingular.

C Experiment Details

C.1 Complete Factorial Results

Table 3 reports the complete factorial results. Panel A gives the undiscounted task outcomes, while Panel B gives the corresponding discounted outcomes with $\gamma = 0.95$, which define the ground-truth values used in our estimation and optimization experiments. Each model-configuration cell averages one trajectory for each of the 97 tasks. Because the agent model is identified by the column group, configurations are represented by four bits in the order (C, R, P, V).

Table 3: Complete factorial results for the two agent models.

Configuration	DeepSeek-V4-Flash			GLM-5.3		
	Reward	Cost (USD)	Latency (s)	Reward	Cost (USD)	Latency (s)
Panel A: Undiscounted outcomes						
1111	0.3608	0.176282	178.11	0.4742	1.504098	240.38
1110	0.3711	0.190382	185.81	0.5258	1.475298	222.37
1101	0.3402	0.112807	148.94	0.4227	1.030644	201.20
1100	0.3505	0.103375	138.06	0.5052	1.020856	202.84
1011	0.3608	0.212147	190.26	0.5052	1.666828	225.92
1010	0.3814	0.223530	187.66	0.4330	1.747570	226.68
1001	0.3608	0.132311	148.33	0.4227	1.071253	194.98
1000	0.3608	0.120194	138.49	0.4330	1.185591	197.85
0111	0.4124	0.261264	210.54	0.3196	1.751618	300.71
0110	0.3299	0.272225	220.42	0.3402	1.830935	292.60
0101	0.3505	0.137130	155.55	0.4330	1.143802	209.65
0100	0.3402	0.135650	150.80	0.4433	1.141761	207.82
0011	0.4227	0.285302	209.81	0.3918	2.148705	308.71
0010	0.3505	0.305503	213.64	0.3196	2.169431	318.31
0001	0.3814	0.162320	156.82	0.4536	1.403710	219.22
0000	0.2887	0.161196	154.56	0.4536	1.342187	204.26
Panel B: Discounted outcomes ($\gamma = 0.95$)						
1111	0.1693	0.083826	104.02	0.1953	0.678529	117.57
1110	0.1545	0.088171	104.99	0.2054	0.660472	107.86
1101	0.1753	0.062381	91.15	0.2068	0.548980	109.30
1100	0.1790	0.058137	84.98	0.2447	0.542724	110.23
1011	0.1585	0.095948	105.48	0.1970	0.718233	108.47
1010	0.1642	0.098061	102.75	0.1693	0.730933	107.11
1001	0.1839	0.070238	89.96	0.2148	0.555136	102.70
1000	0.1691	0.064201	83.71	0.2063	0.590235	102.77
0111	0.1444	0.092990	99.96	0.1271	0.643060	128.54
0110	0.1333	0.090706	102.22	0.1398	0.653064	124.35
0101	0.1723	0.064264	88.37	0.1921	0.523609	103.82
0100	0.1639	0.062506	84.61	0.1837	0.515967	101.86
0011	0.1485	0.099900	98.13	0.1357	0.716278	124.11
0010	0.1373	0.099082	98.81	0.1231	0.737915	129.43
0001	0.1677	0.073630	86.46	0.1935	0.606231	104.12
0000	0.1391	0.070896	83.66	0.1879	0.585446	98.51

Each cell is the equal-task mean over 97 trajectories. Costs include agent-LLM and dense-retrieval embedding usage and are measured in USD. Token-based latency divides agent output tokens by the empirical median generation rate: 89.005 tokens/s for DeepSeek-V4-Flash and 41.998 tokens/s for GLM-5.3. Panel B discounts decision-level reward, cost, and latency by γ^t; the official binary reward is realized at the final agent decision. Configuration bits follow the order (C, R, P, V).

C.2 IMPLEMENTS

Configuration encoding. The combined panel represents a harness configuration as $x = (A, C, R, P, V) \in \{0, 1\}^5$. The leading bit selects the agent model, while the remaining four bits encode the harness factors listed in Table 4. The full factorial space therefore contains 32 configurations, with 00000 serving as the conceptual reference configuration.

Discounted outcomes and resource accounting. Let $\mathcal{O} = \{R, C, L\}$ denote reward, token cost, and token-based latency. For task q, configuration x, and agent decision t, we construct the discounted return

$$Y_{q,o}(x) = \sum_{t=0}^{T_q(x)-1} \gamma^t z_{qt,o}(x), \qquad \gamma = 0.95.$$

The reward increment is zero except at the final agent decision, where it equals the official binary task reward. Consequently, the reported reward accounts for both task success and the number of decisions required to complete the task.

Table 4: Configuration factors in bit order (A, C, R, P, V).

Bit	Stage	Level 0	Level 1
A	Agent model	DeepSeek-V4-Flash	GLM-5.3
C	Tool access	Official progressive discovery through unlock and broker tools	Direct exposure of the fixed public specialized banking-tool schemas, compiled to the unchanged official protocol before grading
R	Retrieval	BM25 behind `KB_search`	Fixed-budget reciprocal-rank fusion of BM25 and Qwen3-Embedding-8B dense retrieval behind the same `KB_search` schema
P	Pre-action	No additional transaction gate	Before each consequential transaction, require an accepted evidence record containing retrieved document IDs, the exact tool name, and exact JSON arguments
V	Post-action	Standard completion behavior	After each successful state-changing decision batch, insert an audit reminder observable at the next agent decision

Table 5: Resource schedule used to construct the outcome panel.

Model and role	Input price	Output price	Generation rate v_g
DeepSeek-V4-Flash (agent)	\$0.14	\$0.28	89.005
GLM-5.3 (agent)	\$1.40	\$4.40	41.998
Qwen3-Embedding-8B (retrieval)	\$0.01	–	–

Let $\mathcal{K}_{qt}^{A}$ denote the agent-model calls associated with decision t, and let N_{qtk}^{in} and N_{qtk}^{out} be the corresponding provider-reported input and output tokens. If N_{qt}^{Q} denotes input tokens used by the Qwen retrieval embedder, the standardized token-cost increment is

$$z_{qt,C}(x) = \frac{1}{10^6}\left[\sum_{k\in\mathcal{K}_{qt}^{A}}\left(p_{g(x)}^{\text{in}}N_{qtk}^{\text{in}} + p_{g(x)}^{\text{out}}N_{qtk}^{\text{out}}\right) + p_{Q}^{\text{in}}N_{qt}^{Q}\right],$$

where $g(x)$ is the agent model selected by A. This standardized measure charges all input tokens at the cache-miss rate and therefore does not apply a provider-specific cache discount. It includes agent-model and retrieval costs, but excludes the cost of the simulated user.

Token-based latency is

$$z_{qt,L}(x) = \frac{\sum_{k\in\mathcal{K}_{qt}^{A}} N_{qtk}^{\text{out}}}{v_{g(x)}},$$

where v_g is the median call-level generation rate, N^{out}/generation seconds, over valid calls in that model's complete 1,552-trajectory collection. Table 5 reports the resource schedule used to construct the panel.

For each configuration, the evaluation reference is the equal-task mean over the complete 97-task panel:

$$\mu_o^{\text{panel}}(x) = \frac{1}{97}\sum_{q=1}^{97} Y_{q,o}(x).$$

This quantity is an empirical finite-panel reference rather than the value under an infinite task population.

Partial-panel design. Every estimation draw observes the fixed design

$$\mathcal{X}_{\text{obs}} = \{\texttt{00010}, \texttt{00011}, \texttt{01000}, \texttt{01010}, \texttt{01011}, \texttt{10001}, \texttt{11000}, \texttt{11010}, \texttt{11011}, \texttt{11101}\}.$$

The design contains five configurations for each agent model. Its model matrix with columns $(1, A, C, R, P, V)$ has full column rank, so all six main-effect coefficients are identified. The reference configuration 00000 is not observed; it is estimated, together with the other 21 unobserved configurations, by extrapolation through the structured outcome model.

Each draw samples 80 distinct tasks without replacement and observes all ten design configurations for every sampled task. Thus, an estimation run contains 800 trajectories. Configurations outside $\mathcal{X}_{\text{obs}}$ are not made available to any estimator. Appendix C.4 presents a sensitivity analysis of the configuration designs.

Budget grid and configuration selection. Let $C_{\min}$ and $C_{\max}$ be the minimum and maximum full-panel cost means over the 32 configurations, and define $L_{\min}$ and $L_{\max}$ analogously. The initial grid contains 51 evenly spaced cost budgets and 49 evenly spaced latency budgets. We retain every sixth value beginning with the minimum:

$$\mathcal{B}_C = \left\{ C_{\min} + \frac{j}{50}(C_{\max} - C_{\min}) : j = 0, 6, \ldots, 48 \right\},$$

$$\mathcal{B}_L = \left\{ L_{\min} + \frac{\ell}{48}(L_{\max} - L_{\min}) : \ell = 0, 6, \ldots, 48 \right\}.$$

Their Cartesian product produces a 9×9 grid. Optimization performance is evaluated on the 80 budget pairs for which at least one configuration is feasible according to the full-panel means.

For each budget pair, a method selects the configuration with the highest predicted reward among those satisfying both predicted constraints. If its predicted feasible set is empty, it selects the configuration minimizing

$$\max\left\{ \frac{\widehat{\mu}_C(x)}{B_C}, \frac{\widehat{\mu}_L(x)}{B_L} \right\},$$

with ties broken in favor of higher predicted reward.

State representation and outcome model. For task q and observed arm $a \in \mathcal{X}_{\text{obs}}$, let S_{qa} denote the reconstructed state immediately before the first agent decision. It contains the benchmark-common system prompt and tool schemas together with the visible dialogue, including the greeting and customer opening request. The assigned configuration, configuration-specific prompt additions, treatment-specific tool interfaces, retrieval implementation, model arguments, hidden environment information, evaluation criteria, and grader information are excluded.

The reconstruction prevents the assigned configuration from being encoded directly in the state. However, it does not require the realized initial states of different trajectories for the same benchmark task to be identical; the estimator therefore retains the trajectory-level notation S_{qa}.

Each state is serialized deterministically and embedded using `gemini-embedding-2` with the classification-task prefix and a 768-dimensional output (Shanbhogue et al., 2026). If the prefixed input exceeds the 8,192-token limit, it is divided into nonoverlapping chronological chunks. The normalized chunk embeddings are averaged using their state-token counts and the resulting vector is renormalized.

Let

$$\varphi(x) = (1, A, C, R, P, V)^\top.$$

For each outcome o, HOP models the conditional mean as

$$m_o(x, s) = G_o(x, \theta_o(s)) = h_o\{\varphi(x)^\top \theta_o(s)\},$$

where $\theta_o(s) \in \mathbb{R}^6$ is state dependent. The reported specification contains main effects only, with

$$h_R(u) = \{1 + \exp(-u)\}^{-1}, \qquad h_C(u) = h_L(u) = \exp(u).$$

For numerical stability, the linear predictor is clipped to $[-40, 40]$ before applying a nonlinear link. The reported estimator imposes neither bounds nor shrinkage on the state-dependent factor effects.

We parameterize $\theta_o(s)$ by a neural network with layer widths 768–16–8–6, SiLU activations (Ramachandran et al., 2017), and dropout 0.2 after each hidden layer. The model is trained using full-batch AdamW (Loshchilov & Hutter, 2017), with learning rate 3×10^{-3}, weight decay 0.1, and squared loss. Row weights assign equal total weight to every task.

The current experiment performs no hyperparameter search or validation-based early stopping. The training lengths are fixed at 53 epochs for reward, 215 for cost, and 70 for latency. These values are the rounded-up medians of the stopping epochs selected by earlier held-out-loss rotations on observed trajectories. Neither their selection nor the current fitting procedure uses the 97-task full-panel outcomes. Appendix C.3 presents a sensitivity analysis of the hyperparameter choices.

Cross-fitting and orthogonal correction. We use five randomly constructed task-level folds. Each fold contains 16 held-out tasks; its nuisance model is fitted using the remaining 64 tasks. All ten trajectories belonging to a task remain in the same fold.

For a task q in held-out fold f, define

$$\dot{\widehat{m}}_{qa,o}(x) = G_{o,\theta}\left(x, \widehat{\theta}_o^{(-f)}(S_{qa})\right).$$

For the observed trajectory assigned to arm a, the squared-loss gradient is

$$\widehat{\ell}_{qa,o} = 2\dot{\widehat{m}}_{qa,o}(a)\left\{\widehat{m}_{qa,o}(a) - Y_{qa,o}\right\},$$

where

$$\widehat{m}_{qa,o}(x) = G_o\left(x, \widehat{\theta}_o^{(-f)}(S_{qa})\right).$$

At the same state, the conditional information matrix is estimated as

$$\widehat{\Lambda}_{qa,o} = \frac{2}{|\mathcal{X}_{\text{obs}}|}\sum_{b\in\mathcal{X}_{\text{obs}}} \dot{\widehat{m}}_{qa,o}(b)\dot{\widehat{m}}_{qa,o}(b)^\top.$$

The cross-fitted task score for target configuration x is

$$\widehat{\psi}_{q,o}(x) = \frac{1}{|\mathcal{X}_{\text{obs}}|}\sum_{a\in\mathcal{X}_{\text{obs}}}\left[\widehat{m}_{qa,o}(x) - \dot{\widehat{m}}_{qa,o}(x)^\top \widehat{\Lambda}^\dagger_{qa,o}\widehat{\ell}_{qa,o}\right].$$

Thus, trajectory-level corrected predictions are first averaged within a task, after which the value estimate is

$$\widehat{\mu}_o(x) = \frac{1}{n}\sum_{q=1}^{n}\widehat{\psi}_{q,o}(x), \qquad n = 80.$$

Treating tasks as the independent sampling units, the reported standard error is

$$\widehat{\text{se}}_o(x) = \left[\frac{1}{n(n-1)}\sum_{q=1}^{n}\left\{\widehat{\psi}_{q,o}(x) - \widehat{\mu}_o(x)\right\}^2\right]^{1/2}.$$

Numerical inversion. The current implementation does not regularize the orthogonal correction using a condition-number gate, ridge penalty, leverage threshold, or substantive eigenvalue truncation. Instead, $\widehat{\Lambda}_{qa,o}$ is inverted through an eigendecomposition,

$$\widehat{\Lambda}_{qa,o} = \sum_{j=1}^{6}\lambda_j v_j v_j^\top,$$

with

$$\widehat{\Lambda}^\dagger_{qa,o} = \sum_{j:\lambda_j > 10^{-12}\lambda_{\max}}\lambda_j^{-1} v_j v_j^\top.$$

Hence, the ordinary inverse is used whenever the matrix is numerically nonsingular. The 10^{-12} cutoff only drops directions that are singular at floating-point precision, and the number of affected rows is recorded as a diagnostic. The extrapolation leverage

$$\dot{\widehat{m}}_{qa,o}(x)^\top \widehat{\Lambda}^\dagger_{qa,o}\dot{\widehat{m}}_{qa,o}(x)$$

is also recorded but does not alter the correction or the selected configuration.

Repeated task draws. For draw $d = 0, \ldots, 19$, we set

$$\text{seed}_d = 20260824 + 1009d$$

and select the first 80 entries of the resulting random permutation of the sorted 97 task identifiers. The same seed determines the task-level cross-fitting folds. Task sampling, estimation, and constrained optimization are therefore repeated for 20 draws, and reported results are averaged across them.

The repetitions reduce dependence on any particular 80-task subset. Because the evaluation reference is itself the finite 97-task panel mean, they measure sampling sensitivity relative to that empirical benchmark rather than estimation error with respect to an infinite task population.

Table 6: Hyperparameter sensitivity of HOP and PDL.

	Regret ($\times 100$)		PCS		PGS	
Setting	HOP	PDL	HOP	PDL	HOP	PDL
Base	1.096	1.375	45.1%	28.9%	64.1%	49.2%
Epochs $0.5\times$	1.065	2.161	46.4%	9.3%	63.0%	44.9%
Epochs $2\times$	0.976	1.040	53.9%	35.9%	64.3%	60.6%
Hidden $(8, 4)$	1.057	1.699	48.3%	16.6%	64.9%	48.1%
Hidden $(32, 16)$	0.991	1.096	51.9%	31.0%	66.3%	54.5%
Learning rate 10^{-3}	1.011	2.098	48.1%	11.6%	63.3%	45.6%
Learning rate 10^{-2}	0.965	0.999	53.6%	33.2%	64.2%	63.4%
Dropout 0	1.137	1.395	45.1%	27.3%	59.2%	48.4%
Dropout 0.4	1.049	1.413	49.1%	30.9%	66.1%	49.3%
Weight decay 0.01	1.096	1.364	46.1%	28.8%	65.1%	49.1%
Weight decay 1	1.114	1.064	46.9%	40.3%	64.5%	57.6%

Each non-base row changes one hyperparameter from the reported specification and applies the resulting protocol identically to HOP and PDL. Entries are averages over the same 20 task draws used in the main experiment. Regret is reported in reward percentage points.

Table 7: Constrained-optimization performance averaged over 10 designs.

Method	Regret ($\times 100$)	PCS	PGS	Violation rate	Cost overrun (%)	Latency overrun (%)	Optimizer cost (USD/draw)
HOP	0.957	29.5%	61.9%	17.38%	0.630%	0.290%	–
LR	1.230	22.3%	60.5%†	21.22%	0.957%	0.348%	–
PDL	1.216	17.1%	46.6%	27.53%	1.343%	0.678%	–
BO	1.092	27.6%†	52.7%	24.19%	1.139%	0.657%	–
OPRO	2.533	9.5%	26.9%	24.38%	1.844%	0.774%	0.3283
GEPA	2.266	12.7%	30.5%	21.98%	1.673%	0.627%	0.3284
MH	2.172	12.9%	31.0%	23.53%	1.843%	0.688%	0.3288

Each design observes ten configurations, five per agent model. Entries are means over 20 task draws of 80 tasks of the average over the 10 designs. Regret is reported in reward percentage points. Overruns are $[C(\widehat{x}) - B_C]_+/B_C$ and $[L(\widehat{x}) - B_L]_+/B_L$, averaged over all selections. † denotes no significant paired difference from HOP across the 20 draws ($p \geq 0.05$); every unmarked baseline entry is significantly worse than HOP ($p < 0.05$). Optimizer cost is the optimizer LLM's input tokens times the input price plus its output tokens times the output price.

C.3 Hyperparameter Sensitivity

We examine whether the optimization results are sensitive to the neural-network training specification. We compare HOP with PDL because PDL uses the same hidden-layer specification, optimizer, and training hyperparameters as HOP while omitting HOP's structured outcome model and orthogonal correction. Starting from the reported specification, we vary one hyperparameter at a time: the outcome-specific training lengths are multiplied by 0.5 or 2, the hidden widths are changed from $(16, 8)$ to $(8, 4)$ or $(32, 16)$, the learning rate from 3×10^{-3} to 10^{-3} or 10^{-2}, dropout from 0.2 to 0 or 0.4, and weight decay from 0.1 to 0.01 or 1. The same perturbed protocol is applied to both methods, while the observed configurations, 20 task draws, task-level folds, and budget grid are held fixed. The sensitivity grid was fixed before examining its results.

HOP attains higher mean PCS and PGS than PDL under all 11 specifications and lower mean regret under 10 of the 11. The sole regret exception is weight decay 1, where PDL has slightly lower mean regret (1.064 versus 1.114), but the paired difference across task draws is not statistically significant ($p = 0.430$). Thus, the main decision-quality comparison is not specific to the reported neural-network hyperparameters, although the magnitude of HOP's advantage varies across settings.

C.4 Design Sensitivity

We conduct a design sensitivity analysis by considering 10 different configuration choices, with all 10 configurations observed by the model, and report the average performance of HOP and the baselines across these designs in Table 7. The results show that HOP maintains robust performance across different design choices.

C.5 Baseline Implementations

Common protocol. In each draw, every baseline uses only the outcomes of the same 10 observed configurations on the same 80 tasks and selects from all 32 configurations. All methods make selections on the common 9×9 cost–latency grid; reported optimization metrics exclude the single budget cell for which no configuration is feasible according to the full-panel means. No baseline may request additional agent rollouts.

If a point-estimation method predicts a nonempty feasible set, it selects the configuration with the largest predicted reward. Otherwise, HOP, LR, PDL, and BO use the common least-violation fallback

$$\arg\min_{x} \max \left\{ \frac{\widehat{C}(x)}{B_C}, \frac{\widehat{L}(x)}{B_L} \right\},$$

with ties broken in favor of higher predicted reward. The LLM selectors are instead instructed to name the configuration expected to overrun the budgets least when they predict that no candidate is feasible.

Linear regression (LR). For each outcome $o \in \{R, C, L\}$, LR fits the task-weighted ordinary least-squares model

$$Y_{qa}^{(o)} = \beta_{o0} + S_{qa}^{\top} \beta_{oS} + a^{\top} \beta_{oX} + \varepsilon_{qa,o},$$

where $S_{qa} \in \mathbb{R}^{768}$ is the reconstructed state for the trajectory of task q under observed configuration a, and $a = (A, C, R, P, V)^{\top} \in \{0, 1\}^5$. Training weights assign equal total weight to each task.

LR is fitted once using all 800 observed trajectories and is not cross-fitted. A training split from the five-fold procedure would contain only $64 \times 10 = 640$ rows for 773 predictors—the 768 state coordinates and five configuration indicators—making the fold-specific regression underdetermined. For each target configuration x, LR retains each observed state S_{qa} and replaces the observed configuration indicators a by x. Predictions are averaged first across trajectories within task and then equally across the 80 tasks. LR uses global additive configuration effects, an identity link, and no orthogonal correction.

Pure deep learning (PDL). PDL replaces LR with a separate direct-outcome network for each outcome,

$$\widehat{m}_o(x, S_{qa}) = f_o([S_{qa}; x]; \eta_o).$$

It uses the same five task-level outer folds as HOP: each network is trained on 64 tasks and predicts the 16 held-out tasks. Predictions are averaged first within task and then equally across all 80 tasks.

The network has hidden widths 16 and 8, SiLU activations, dropout 0.2, and a scalar identity output for every outcome. It is trained using task-balanced squared loss and full-batch AdamW, with learning rate 3×10^{-3} and weight decay 0.1. The current implementation does not use validation-based early stopping. Training lengths are fixed at 53 epochs for reward, 215 for cost, and 70 for latency, matching HOP's nuisance-network protocol. PDL shares information across configurations through the common network but imposes neither HOP's state-dependent coefficient representation and outcome-specific links nor its orthogonal correction.

Bayesian optimization (BO). The one-shot BO baseline fits an independent Gaussian process for each outcome using the equal-task means of the 10 observed configurations. The observation variance is the squared paired standard error obtained after removing the task effect shared across configurations. Each outcome is centered and scaled before fitting a constant-times-Matérn-$5/2$ kernel on the five binary configuration factors. The reported fits use 20 optimizer restarts.

The three Gaussian-process surrogates are frozen before any budget is considered. Their posterior means provide predictions for all 32 configurations, and their posterior standard deviations provide BO's uncertainty estimates. BO then applies the common constrained posterior-mean selection rule. Unlike sequential Bayesian optimization, it acquires no additional configuration outcomes (Snoek et al., 2012).

LLM-based selectors. We evaluate a one-step OPRO (Yang et al., 2024), a one-step GEPA (Agrawal et al., 2026), and a one-shot Meta-Harness (Lee et al., 2026). All use DeepSeek-V4-Flash with temperature zero and reasoning disabled. For each method and draw, one independent call is made for each of the 81 budget pairs.

Table 8: Statistical significance of optimization results.

Metric	LR	PDL	BO	OPRO	GEPA	MH
Regret	+.0040 (<.0001)	+.0028 (.0101)	+.0037 (.0001)	+.0191 (<.0001)	+.0166 (<.0001)	+.0166 (<.0001)
PCS	+25.2 pp (<.0001)	+16.2 pp (.0012)	+41.1 pp (<.0001)	+43.9 pp (<.0001)	+44.8 pp (<.0001)	+43.8 pp (<.0001)
PGS	+6.2 pp (.0421)	+14.8 pp (<.0001)	+14.3 pp (.0002)	+52.2 pp (<.0001)	+53.8 pp (<.0001)	+52.8 pp (<.0001)
Violation	+0.6 pp (.7393)	+7.8 pp (.0002)	+10.6 pp (<.0001)	+13.1 pp (<.0001)	+9.9 pp (<.0001)	+10.6 pp (<.0001)
Cost overrun	+0.627 pp (.0008)	+0.562 pp (.0005)	+0.292 pp (.0227)	+1.230 pp (<.0001)	+1.351 pp (<.0001)	+1.548 pp (<.0001)
Latency overrun	+0.005 pp (.7661)	+0.040 pp (.0244)	+0.055 pp (.0753)	+0.425 pp (<.0001)	+0.036 pp (.2800)	+0.151 pp (.0069)

Each cell reports the mean paired difference across the 20 task draws followed by the two-sided p-value from a one-sample t-test ($df = 19$). Differences are oriented so that a positive value favors our method. Regret differences are in reward units. PCS, PGS, and violation differences are in percentage points, and cost and latency overrun differences are in percentage points of the budget. No multiple-comparison correction is applied.

Every call receives the factor definitions, all 32 candidate codes, and the 800 anonymized task-level outcome records for the 10 observed configurations. It also receives up to 20 redacted representative trajectories selected from that draw: one success and one failure per observed configuration when both are available. Within each outcome class, the representative trajectory is the observation nearest the class center in standardized cost and latency. The prompt contains no outcomes or trajectories from unobserved configurations. It requires the model to report whether it expects any candidate to be feasible and to name a configuration either way.

OPRO lists the observed-configuration blocks in ascending order of empirical mean reward and asks the model to infer factor effects and interactions before proposing a candidate. GEPA contrasts successful and unsuccessful traces and uses the resulting reflection to mutate or extrapolate beyond an observed configuration. Meta-Harness derives at most four ordered selection rules and applies them to the full candidate set. These are deliberately one-call, offline adaptations rather than the full iterative algorithms: they cannot test a proposal, observe a new outcome, or refine it.

C.6 Optimization Significance Tests

We compare our method with each baseline using two-sided one-sample t-tests on the paired differences across the 20 task draws. For metric j and baseline b,

$$D_{d,j}^{(b)} = \begin{cases} M_{d,j}^{(\text{ours})} - M_{d,j}^{(b)}, & \text{if higher is better,} \\ M_{d,j}^{(b)} - M_{d,j}^{(\text{ours})}, & \text{if lower is better,} \end{cases}$$

so a positive difference favors our method. We test $H_0 : \mathbb{E}[D_{d,j}^{(b)}] = 0$ with the task draw as the unit of inference, which is equivalent to a paired t-test because all methods are evaluated on the same draws. Table 8 gives the mean oriented difference alongside each p-value, so direction is readable from the table itself. Optimizer-LLM cost is excluded because it is a directly computed accounting quantity rather than a task-sampling outcome.

C.7 Component Ablation Significance Tests

Table 9 reports paired tests using the same 20 task draws for HOP and each ablated variant. For each metric, we test whether the mean draw-level difference is zero. Removing the orthogonal correction significantly worsens regret, PCS, PGS, violation rate, and latency overrun. Replacing the outcome-specific links significantly increases violation rate and cost overrun, while removing cross-fitting significantly affects only latency overrun. These results use nominal two-sided p-values without multiple-comparison adjustment.

C.8 Inference

Point-estimate optimization identifies a promising harness configuration, but deployment may also require inference about a particular configuration's expected benefits and resource requirements. HOP provides such uncertainty quantification with the validity guarantee established in Section 3.4. We evaluate nominal 95% confidence intervals for all 32 configurations and separately for the 22 configurations excluded from the experimental sample. Coverage is measured against the empirical full-panel mean. Because an interval can appear artificially precise when its standard error is underestimated, raw interval lengths are not comparable across methods with different coverage. We

Table 9: Statistical significance of component-ablation results.

Metric	No orthogonal correction	No cross-fitting	Identity links
Regret	+.0037 (.0445)	-.0003 (.6499)	+.0000 (.9880)
PCS	+31.1 pp (<.0001)	-4.8 pp (.4269)	+5.1 pp (.3058)
PGS	+10.7 pp (.0467)	-1.9 pp (.4941)	+1.8 pp (.4971)
Violation	+7.0 pp (.0005)	-0.6 pp (.7246)	+5.6 pp (.0007)
Cost overrun	+0.217 pp (.0761)	-0.041 pp (.7900)	+1.063 pp (<.0001)
Latency overrun	+0.108 pp (.0059)	+0.021 pp (.0137)	+0.011 pp (.3307)

Each cell reports the mean paired difference between HOP and the ablated variant across the 20 task draws, followed by the two-sided p-value from a one-sample t-test ($df = 19$). Differences are oriented so that a positive value favors HOP. Regret differences are in reward units. PCS, PGS, and violation differences are in percentage points, and cost and latency overrun differences are in percentage points of the budget. No multiple-comparison correction is applied.

Table 10: Coverage and mean length of nominal 95% intervals.

		LR		PDL		BO		HOP	
Configurations	Outcome	Coverage	Length	Coverage	Length	Coverage	Length	Coverage	Length
All (32)	Reward	93%	0.128	78%	0.118	79%	0.093	99%	0.116
	Cost (USD)	51%	0.138	73%	0.143	99%	0.116	96%	0.111
	Latency (s)	75%	26.8	15%	25.7	70%	18.4	85%	22.5
Unevaluated (22)	Reward	90%	0.139	72%	0.121	76%	0.091	99%	0.125
	Cost (USD)	37%	0.145	64%	0.150	100%	0.125	95%	0.117
	Latency (s)	66%	29.2	13%	26.0	57%	18.1	79%	24.0

Coverage is computed against the 97-task mean and pooled over configurations and 20 task draws. Calibrated length is $2(1.96)s(x)/\sqrt{1-n/N}$, where $s(x)$ is the across-draw standard deviation, $n = 80$, and $N = 97$, averaged over configurations on the outcome's original scale. Across outcomes, HOP generally combines high coverage with comparatively short calibrated lengths.

therefore report an empirically calibrated length, computed using each estimator's observed variation across the 20 task draws rather than its reported standard error. Coverage is evaluated separately using each method's own standard errors: HOP uses cross-fitted task-level scores, LR and PDL use cross-task variation in their predictions, and BO uses posterior standard deviations.

As shown in Table 10, HOP attains coverage within five percentage points of 95% for both reward and cost, over both all and unevaluated configurations. Among methods meeting this criterion, HOP has the shortest calibrated length in all four cases. Relative to LR, its reward lengths are 9% shorter overall and 10% shorter on unevaluated configurations; relative to BO, its cost lengths are 4% and 6% shorter, respectively. For latency, HOP attains 100% coverage on evaluated configurations but 79% on unevaluated ones. This undercoverage reflects extrapolation bias because the retrieval factor appears in only one experimental arm, providing limited support for estimating its effect outside the observed configurations. All four methods are affected by this design limitation, but HOP achieves the highest latency coverage, both overall (85%) and on unevaluated configurations (79%).