VARIANCE-OPTIMAL CONTROL VARIATES FOR LEARNING WITH BLACK-BOX FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern models increasingly learn through black-box oracles such as humans, optimization solvers, and external tools that provide feedback without exposing their internal mechanisms. In such cases, gradients are often estimated using score-function (REINFORCE) estimators but can suffer from prohibitive variance. A common remedy is to learn an (action-)value function as a control variate. In this paper, we first observe that even an exact action-value function can be arbitrarily far from variance-optimal. We show that this gap arises because the value function minimizes the noise in each action's own gradient term, while an action can still affect the rest of the gradient estimator through shared parameters. A simple unbiased correction, at no extra oracle cost, can still reduce its variance by an arbitrarily large factor. Motivated by this, we then prove that the residual variance can be decomposed exactly by actions with no cross terms. This decomposition yields a closed-form *variance-minimizing correction* for neural-network parameters, which can be computed by a simple projection. Empirically, our correction consistently reduces the variance left by the value function and improves learning across all tasks. The source code for all experiments is available here.

1 INTRODUCTION

Many learning systems optimize predictions through a component that is not differentiable. In general, a predictor reads a context and outputs a vector of parameters, and then a black-box oracle evaluates the parameters and returns only a scalar loss. In decision-focused learning (DFL) (Elmachtoub & Grigas, 2022; Mandi et al., 2024), for example, the predicted parameters define an optimization problem, and the loss is the regret of the decision made by the solver. In large language model (LLM) agents, the predicted parameters can be a tool configuration, e.g., which Model Context Protocol (MCP) servers the agent allows to call (Anthropic, 2024), and the loss is whether the agent then completes the task. In all these cases, the downstream computation is opaque, so we have no access to its gradient, and each oracle call can potentially be expensive.

A standard approach is the score-function (REINFORCE) estimator (Williams, 1992; Sutton et al., 1999; Mohamed et al., 2020), which provides an unbiased gradient using only sampled black-box evaluations. Its main limitation is variance. When a single scalar outcome must train many discrete outputs, the resulting gradient can be extremely noisy, especially in long-horizon settings or high-dimensional discrete action spaces (Tokui & Sato, 2017), and provides only coarse credit across outputs. A common unbiased remedy is therefore to use an action-value function as an action-dependent control variate (Gu et al., 2016; Liu et al., 2018; Wu et al., 2018; Foerster et al., 2018). The value function replaces the shared evaluations with conditional information about actions, providing customized credit while preserving unbiasedness. It is also the object that exact local marginalization computes by brute force, at the price of one oracle call per value of every action (Titsias & Lazaro-Gredilla, 2015; Tokui & Sato, 2017; Liu et al., 2019), and it is known to be variance-optimal when every action owns its own parameters (Tokui & Sato, 2017).

However, we show that using this action-value function alone does not solve the variance-minimization problem. Even when the action-value function is known exactly, its gradient estimator can be arbitrarily far from variance-optimal. A simple unbiased correction, at no extra oracle cost, can still reduce its variance by an arbitrarily large factor. The key reason is parameter sharing. The action-value function minimizes the variance of each action's gradient contribution in isolation.

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.

Table 1: Trade-offs for optimizing discrete predictions through an expensive black-box oracle. d is the number of actions and V the number of values each action can take. †Variance-optimal within the realizable family of Theorem 4.5.

	REINFORCE	**Exact local marg.**	**Learned surrogate**	**Ours**
Black-box oracle calls	$\mathcal{O}(1)$	$\mathcal{O}(dV)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Unbiased gradient	✓	✓	×	✓
Variance of estimator	High	Low	Low	Variance-optimized†
Credit assignment	Uniform	Customized	Customized	Customized

However, the final estimator is the sum of these contributions. When actions share neural-network parameters, their gradient vectors overlap in the same parameter space, so the variance of the sum also depends on how the contributions interact. A control variate that is optimal for each term separately is therefore not necessarily optimal for their sum, since changing one term can increase or decrease its covariance with the others. The action-value function does not account for these cross-action interactions.

To characterize and remove this remaining variance, we show that the residual noise left by the action-value function admits an exact action-wise decomposition whose components have no cross terms in variance. Thus, although the original gradient contributions interact through shared parameters, the residual variance can be **optimized separately** for each action. This decomposition directly yields a closed-form variance-minimizing correction to the action-value function. We further show that the optimal correction is simply a projection onto the realizable control-variate subspace, *without* learning an additional control-variate network or solving a large global covariance system. This gives a different view of value-function baselines, in which the action-value function solves the local credit-assignment problem but is not in general the variance-optimal endpoint. When shared-parameter effects are weak, little correction is needed; when they are strong, substantial variance can remain even after the value function is exact. Table 1 summarizes the trade-offs among these approaches and our method.

Empirically, we evaluate this effect across three tasks, including both controlled settings and black-box learning with language-model predictors. We find that substantial variance can remain after applying the action-value function, and that our correction consistently removes a large fraction of this residual variance. These reductions translate into improved learning performance across all three tasks.

2 Related Works

Gradient estimation with action-independent control variates. When the loss is not differentiable, the score-function estimator (Williams, 1992) is the standard unbiased choice, and Silvestri et al. (2026) recently apply it to DFL. The same estimator drives black-box variational inference, where the log-derivative trick estimates the gradient of the ELBO (Paisley et al., 2012; Ranganath et al., 2014). Because the score function has high variance, a long line of work reduces it with baselines, including the optimal constant baselines (Weaver & Tao, 2001; Greensmith et al., 2001), learned baselines (Sutton et al., 1999; Mnih & Gregor, 2014), and leave-one-out baselines (Mnih & Rezende, 2016; Kool et al., 2019). Group-relative baselines are now widely used for LLM reinforcement learning (Shao et al., 2024; Ahmadian et al., 2024; Liu et al., 2025), and Li et al. (2026) derive the optimal token-level baseline for long-horizon LLM RL. Close to us, Cheng et al. (2020) decompose the variance along a trajectory and show that a per-state baseline leaves the terms that couple different time steps. However, these baselines do not depend on actions, so every coordinate still receives the same credit.

Gradient estimation with action-dependent control variates. Action-dependent control variate removes the credit limitation, and they have been built from learned critics with an analytic correction term (Gu et al., 2017), continuous relaxation (Gu et al., 2016; Tucker et al., 2017; Grathwohl et al., 2018), Stein's identity (Liu et al., 2018), factorized policies (Wu et al., 2018; Foerster et al.,

2018), and discrete structures (Titsias & Shi, 2022; Shi et al., 2022), although Tucker et al. (2018) show their gain over action-independent ones is small in practice.

These action-dependent baselines mainly propose new forms of control variate and fit them by regression or by minimizing the variance directly, while the jointly variance-minimizing choice remains less understood when gradient contributions across decisions are coupled. Known optimality results are for baselines, where the optimal baseline has a closed form (Weaver & Tao, 2001; Greensmith et al., 2001; Li et al., 2026), and so does the optimal action-dependent one when the scores of different coordinates are orthogonal (Wu et al., 2018). We characterize the optimum over per-decision control variates without this assumption (Theorem 4.4), which says that a learned control variate should predict how much it can gain over the value function, and when it can gain nothing.

Exact marginalization. Another line of work sums out one variable at a time instead of sampling it (Titsias & Lazaro-Gredilla, 2015; Tokui & Sato, 2017; Liu et al., 2019; Smet et al., 2023), and a related family couples several samples of the same input to the same end (Yin & Zhou, 2019; Dong et al., 2020; Dimitriev & Zhou, 2021). In our notation, these methods build Q_j exactly, by evaluating the loss at *every value of every variable*, which costs dV calls per step and is prohibitive when one call is an agent run. Tokui & Sato (2017) show that this choice is variance-optimal when each variable has its own parameters, while the shared-parameter case has been left open (Smet et al., 2023). Our results answer this question by proving that with shared parameters, Q is no longer optimal, and the optimal control variate has a closed form.

3 Problem Statement

Black-box learning setting. For a context x, a stochastic predictor parameterized by $\theta \in \mathbb{R}^n$ samples a (discrete) parameter vector $b = (b_1, \dots, b_d)$, whose entries take values in a finite set $\mathcal{V}$ of size V, according to a known categorical distribution $p(b|x,\theta) = \prod_{j=1}^{d} p(b_j|z_j, x, \theta)$. Here, z_j denotes the context available when action j is sampled.

We consider two common cases in practice. For an *autoregressive* predictor such as an LLM that outputs one action (decision/token) after another, $z_j = b_{<j}$. For a *factorized* predictor, the coordinates of b are independent given x; we fix an arbitrary order of the coordinates and again set $z_j = b_{<j}$, so that $p_\theta(b_j|z_j, x) = p_\theta(b_j|x)$. All results below are stated for $z_j = b_{<j}$ and therefore cover both cases; for a factorized predictor, the chosen order affects the variance of the estimators in Section 4 but not their unbiasedness. We write $p_j(v)$ for $p_\theta(b_j = v|z_j)$ throughout.

The sampled vector is then evaluated by a black-box oracle, and we optimize $\min_\theta \mathcal{L}(\theta) \coloneqq \mathbb{E}_{p(b|\theta)}[f(b)] = \mathbb{E}_{p(b|\theta)}[\mathrm{Eval}(\mathrm{Oracle}(b))]$. In decision-focused learning, Oracle solves a downstream problem with the predicted parameter b, and Eval evaluates the decision under the true parameters, so f is the decision regret. Our analysis only assumes that f can be evaluated but not differentiated. We also assume the oracle to be deterministic; an oracle with independent evaluation noise only adds an irreducible variance.

REINFORCE. Since the Oracle is a black-box, we can employ the score-function to express the gradient of the objective g as

$$g \coloneqq \nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_{b\sim p(\cdot|\theta)}[f(b) \sum_{j=1}^{d} s_j(b_j)], \quad s_j(v) \coloneqq \nabla_\theta \log p_j(v). \tag{1}$$

The single-sample REINFORCE estimator $\hat{g}_{\mathrm{SF}} \coloneqq f(b) \sum_j s_j(b_j)$ requires only one evaluation of f, but it suffers from both high variance and coarse credit assignment, since the same sampled loss multiplies every score term. Both worsen as d grows because if the sum has d terms, the single scalar $f(b)$ has to explain all of them.

Parameter sharing. We focus on the common case where different actions share model parameters. Then their score vectors can overlap, so reducing the variance of each action separately may not minimize variance of their sum.

In this paper, our goal is to characterize the unbiased action-dependent control variate that minimizes the variance of the full gradient estimator, i.e., find an optimal control variate that solves

$$\hat{g} \in \arg\min_{\hat{g} \in \mathcal{G}} \mathrm{Var}(\hat{g}), \quad \text{s.t. } \mathbb{E}[\hat{g}] = g,$$

where $\mathcal{G}$ denotes the family unbiased score-function estimators with action-dependent control variates. Section 4.4 restricts $\mathcal{G}$ further to the corrections a neural-network gradient can realize; all optimality statements in this paper refer to that family. This is challenging because the optimum must account for interactions across actions rather than treating them independently.

4 Variance-optimal Control Variate

We ask how much of the score-function variance can be removed without introducing bias. Our key observation is that the coordinate-wise optimality can be strictly (and even arbitrarily) suboptimal when parameters are shared across actions. Then, we characterize the remaining variance through prefix-wise residuals and derive a variance-optimal correction in closed form.

4.1 The Action-Value Control Variate

Consider the j-th term in Equation (1). Let $b_{\mathrm{rem}(j)}$ denote the randomness not contained in (z_j, b_j). Applying the law of total expectation gives

$$\begin{aligned}\mathbb{E}_b\big[f(b)s_j(b_j)\big] &= \mathbb{E}_{z_j}\mathbb{E}_{b_j|z_j}\mathbb{E}_{b_{\mathrm{rem}(j)}|z_j,b_j}\Big[f(b)s_j(b_j)\Big] \\ &= \mathbb{E}_{z_j}\mathbb{E}_{b_j|z_j}\big[Q(z_j,b_j)s_j(b_j)\big], \qquad Q(z_j,v) \coloneqq \mathbb{E}_{b_{\mathrm{rem(j)}}}\big[f(b)|z_j,b_j=v\big]. \end{aligned} \tag{2}$$

Here, Q can be regarded as the action-value function, which is widely used in RL for variance reduction. Since b_j is categorical, $g = \nabla_\theta \mathcal{L}(\theta) = \sum_{j=1}^{d} \mathbb{E}_{z_j}[\sum_{v\in\mathcal{V}} p_j(v) Q(z_j,v) s_j(v)]$.

The exact sweep for $v \in \mathcal{V}$ is usually too expensive to compute, since it costs one oracle call for each of the dV pairs (j, v), so a common way is to replace Q by a predictor $\hat{Q}(z_j, v) \approx Q(z_j, v)$ fitted from previous oracle evaluations. However, simply replacing Q by $\hat{Q}$ in Equation (2) would introduce bias due to approximation error. Instead, we use $\hat{Q}$ as a control variate (Hammersley & Handscomb, 1964). By adding and subtracting the expectation, we get the following estimator

$$\hat{g}_{\hat{Q}} \coloneqq \sum_{j=1}^{d} \Big[\underbrace{\big(f(b) - \hat{Q}(z_j,b_j)\big)s_j}_{\text{term (a)}} + \underbrace{\textstyle\sum_{v\in\mathcal{V}} p_j(v)\hat{Q}(z_j,v)s_j(v)}_{\text{term (b)}} \Big]. \tag{3}$$

Here, $\mathrm{term\ (a)}$ denotes the score-function with an action-dependent baseline, while $\mathrm{term\ (b)}$ is its correction term, given by the expectation of the baseline-weighted score over the current action. The estimator pays exactly one oracle call per step, the same as REINFORCE, and the sweep it needs is over the *fitted model* $\hat{Q}$, not over the oracle.

Lemma 4.1 (Unbiasedness and coordinate-wise optimality)**.** *Given a fitted predictor $\hat{Q}$, Equation* (3) *is unbiased for any choice of $\hat{Q}$, i.e., $\mathbb{E}[\hat{g}_{\hat{Q}}] = \nabla_\theta \mathcal{L}(\theta) = g$. Moreover, writing $\hat{g}_{\hat{Q},j}$ for the j-th summand in Equation* (3)*, the exact conditional mean Q minimizes its conditional variance, i.e., $Q(z_j,\cdot) \in \arg\min_{\hat{Q}} \mathrm{Var}(\hat{g}_{\hat{Q},j}|z_j)$.*

See Appendix A.1 for proof. The lemma explains why Q is the natural target for a learned action-dependent control variate. It removes all variance that can be removed when the index j is optimized in isolation. This is also the sense in which prior work calls it optimal (Tokui & Sato, 2017). However, previous work did not consider the total variance of the gradient estimator *sum* over j, and their analysis does not guarantee its optimality.

4.2 Why Q Is Not Variance-Optimal

Lemma 4.1 shows that Q can minimize each action's own gradient term in isolation. This does not imply that the variance of the sum of all d terms in Equation (3) is minimal, because those terms are

vectors in one shared parameter space and can be correlated. The following example shows the gap exists even in the simplest case with $d = 2$; Appendix A.2 turns it into a family in which the exact value function removes *none* of the REINFORCE variance while the optimum removes *all* of it.

Example 4.2. *Consider two binary decisions generated one after another, $b_1 \sim \text{Bernoulli}(p)$ and $b_2 \mid b_1 \sim \text{Bernoulli}(p)$, with a single shared parameter $p = \sigma(\theta)$, where $\sigma(\cdot)$ denotes the sigmoid function, and let $f(b_1, b_2) = b_1 b_2$. At $\theta = 0$ we have $p = 1/2$ and $\nabla_\theta p = 1/4$, so $\nabla_\theta \mathcal{L}(\theta) = \nabla_\theta\, p^2 = 1/4$, and the score function is $s_j(v) = \nabla_\theta \log p(v) = v - 1/2$. With prefix conditioning, $z_1 = \emptyset$ and $z_2 = b_1$, the exact conditional means are $Q_1(v) = \mathbb{E}[f \mid b_1 = v] = v/2$ and $Q_2(b_1, v) = b_1 v$: Q_1 averages out the later decision b_2, while Q_2 conditions on the earlier one. Substituting into Equation* (3) *and using $b_1(b_1 - \frac{1}{2}) = b_1/2$,*

$$\hat{g}_Q = \underbrace{\tfrac{b_1}{2}\big(b_2 - \tfrac{1}{2}\big)}_{\text{term (a)}} + \underbrace{\tfrac{1}{8} + \tfrac{b_1}{4}}_{\text{term (b)}}, \qquad \mathbb{E}[\hat{g}_Q] = \tfrac{1}{4}, \qquad \text{Var}(\hat{g}_Q) = \tfrac{3}{64} > 0.$$

Both summands remain random even though Q is exact. The first through $f - Q_1(b_1) = b_1(b_2 - \frac{1}{2})$, which depends on the later *decision, and the second through its correction term $\sum_v p_2(v) Q_2(b_1, v) s_2(v) = b_1/4$, which depends on the* earlier *one. Now perturb the exact conditional means by $\tilde{Q}_1 = Q_1 + \frac{1}{4}$ and $\tilde{Q}_2 = Q_2 + \frac{b_1}{2}$. Neither perturbation depends on v, so term (b) is unchanged (because $\sum_v p_j(v) s_j(v) = 0$), while in term (a) the perturbations cancel the remaining randomness exactly:*

$$\hat{g}_{\tilde{Q}} = \underbrace{-\tfrac{1}{4}\big(b_1 - \tfrac{1}{2}\big)}_{\text{term (a)}} + \underbrace{\tfrac{1}{8} + \tfrac{b_1}{4}}_{\text{term (b)}} = \tfrac{1}{4}, \qquad \text{Var}(\hat{g}_{\tilde{Q}}) = 0 < \text{Var}(\hat{g}_Q).$$

The issue in the $\hat{g}_{\hat{Q}}$ estimator in Equation (3) is not an inaccurate value function, because even Q is exact, its variance could still be suboptimal. Rather, Q minimizes the variance of one coordinate at a time, while the quantity we ultimately care about is the variance of their sum in a shared parameter space, which could potentially be further optimized. We next characterize exactly what remains.

4.3 A General Framework for Variance-Optimal Control Variates

In this section, we introduce a general correction term Δ to capture the remaining noise in $\hat{g}_{\hat{Q}}$ estimator and minimize the variance of the resulting gradient estimator.

General control variate. Let $\Delta_j(z_j, v) \in \mathbb{R}^n$ be any vector-valued function. We center it over the current action to obtain a zero-mean function:

$$\tilde{\Delta}_j \coloneqq \Delta_j(z_j, b_j) - \sum_v p_j(v) \Delta_j(z_j, v), \quad U_\Delta \coloneqq \sum_{j=1}^{d} \tilde{\Delta}_j. \tag{4}$$

By construction, $\mathbb{E}[\tilde{\Delta}_j | z_j] = 0$ and hence $\mathbb{E}[U_\Delta] = 0$. Therefore, we have a general estimator

$$\hat{g}_\Delta \coloneqq \hat{g}_Q - U_\Delta, \tag{5}$$

which is unbiased for any choice of Δ. Note that Δ_j is a function of the sampled action, so subtracting it keeps the per-action credit assignment of Equation (3) and costs no additional oracle call.

Decomposition of remaining noises. Let $e_Q \coloneqq \hat{g}_Q - g$ denote the remaining noise after subtracting Q. We first decompose the residual e_Q into different action time steps. For each action j and value v, define the *residual increment*

$$\xi_j(z_j, v) \coloneqq \mathbb{E}[\hat{g}_Q | z_j, b_j = v] - \mathbb{E}[\hat{g}_Q | z_j]. \tag{6}$$

Thus, $\xi_j(z_j, v)$ measures how much choosing $b_j = v$ can change the expected gradient, given everything decided before j. By construction, we have

$$\mathbb{E}_{b_j | z_j}[\xi_j(z_j, b_j)] = \sum_v p_j(v) \xi_j(z_j, v) = 0. \tag{7}$$

Lemma 4.3 (Action-wise decomposition)**.** *Let $z_j = b_{<j}$. Along a sampled b, we can write $e_Q = \hat{g}_Q - g = \sum_{j=1}^{d} \xi_j(z_j, b_j)$. Moreover, for any $i < j$, we have $\mathbb{E}[\xi_i(z_i, b_i)^\top \xi_j(z_j, b_j)] = 0$. Therefore, the variance of $\hat{g}_Q$ is $\text{Var}(\hat{g}_Q) = \sum_{j=1}^{d} \mathbb{E}\big[\|\xi_j(z_j, b_j)\|_2^2\big]$.*

Proof. Because $z_j = b_{<j}$ and $(z_j, b_j) = b_{\le j}$,

$$\sum_{j=1}^{d} \xi_j(z_j, b_j) = \sum_{j=1}^{d} \left(\mathbb{E}[\hat{g}_Q | b_{\le j}] - \mathbb{E}[\hat{g}_Q | b_{<j}]\right) = \hat{g}_Q - \mathbb{E}[\hat{g}_Q] = e_Q,$$

where the sum telescopes, using that $\hat{g}_Q$ is a function of $b = b_{\le d}$ alone, and that $\mathbb{E}[\hat{g}_Q] = g$ by Lemma 4.1. For any $i < j$, $\xi_i(z_i, b_i)$ is determined by $b_{\le i}$ and hence by $z_j = b_{<j}$. Therefore, $\mathbb{E}[\xi_i^\top \xi_j] = \mathbb{E}\left[\xi_i^\top \mathbb{E}[\xi_j | z_j]\right] = 0$, where the last equality follows from Equation (7). The variance identity then follows by expanding $\|e_Q\|_2^2$ and using the orthogonality above. □

Lemma 4.3 is the step that makes the problem tractable. The residual left by Q is a difference sequence in the action index, so its variance carries **no cross terms** even though the underlying gradient contributions are strongly coupled through θ. Then, we are ready to derive our main theorem: the variance-optimal correction in closed form.

Theorem 4.4. *Let $z_j = b_{<j}$. For any control variates $\Delta = (\Delta_1, \dots, \Delta_d)$ in Equation* (4)*,*

$$\mathrm{Var}(\hat{g}_\Delta) = \mathrm{Var}(\hat{g}_Q) + \sum_{j=1}^{d} \mathbb{E}[\|\tilde{\Delta}_j\|^2 - 2\xi_j^\top \tilde{\Delta}_j] = \sum_{j=1}^{d} \mathbb{E}\|\xi_j(z_j, b_j) - \tilde{\Delta}_j(z_j, b_j)\|^2. \tag{8}$$

Hence, among unrestricted control variates, the optimal one is $\tilde{\Delta}_j^*(z_j, v) = \xi_j(z_j, v), \forall j \in [d]$, which attains $\mathrm{Var}(\hat{g}_{\Delta^*}) = 0$ (see Remark A.3).

Proof. See Section A.4 for proof. The key idea is that Lemma 4.3 decomposes the remaining noise into orthogonal action-wise residuals ξ_j. The variance therefore separates across j and is minimized by matching each centered control variate to ξ_j. □

Theorem 4.4 shows that, after applying the conditional-mean control variate Q, the remaining variance is exactly the mismatch between the residual increment ξ_j and the additional control variate, and that these mismatches add up over decisions with no cross terms. The unrestricted minimizer is a benchmark, not an estimator (Remark A.3); all optimality claims below refer to the realizable family of Section 4.4.

Algorithm 1 Learning through a black box with an action-conditioned control variate.

Require: policy $p_\theta(b) = \prod_{j=1}^{d} p_j(b_j | z_j)$, black box f, fitted reward model $\hat{f}$, step size η

1: $\mathcal{D} \leftarrow \{(b, f(b)) : b \sim p_{\theta_0}\}$ ▷ a fixed offline dataset; $\mathcal{D} = \emptyset$ also works
2: **for** $t = 1, 2, \dots$ **do**
3: fit $\hat{f}$ to $\mathcal{D}$ ▷ any regressor, e.g., a neural network
4: sample $b \sim p_\theta$ and pay one call $f(b)$
5: $\hat{Q}(z_j, v) \leftarrow \mathbb{E}\big[\hat{f} | z_j, b_j = v\big]$ for every $j \in [d]$ and value $v \in \mathcal{V}$ ▷ under p_θ, no oracle call
6: $\hat{g}_{\hat{Q}} \leftarrow \sum_{j=1}^{d} \Big[\big(f(b) - \hat{Q}(z_j, b_j)\big) s_j(b_j) + \sum_v p_j(v) \hat{Q}(z_j, v) s_j(v)\Big]$ ▷ Equation (3)
7: $\hat{\xi}_j \leftarrow \mathbb{E}_{\hat{f}}[\hat{g}_{\hat{Q}} | z_j, b_j] - \mathbb{E}_{\hat{f}}[\hat{g}_{\hat{Q}} | z_j]$ for every $j \in [d]$ ▷ Equation (6)
8: $\hat{g} \leftarrow \hat{g}_{\hat{Q}} - \sum_{j,r} \Pi_{j,r} \hat{\xi}_{j,r}$ ▷ no matrix solve, fast
9: $\theta \leftarrow \theta - \eta \hat{g}$; $\mathcal{D} \leftarrow \mathcal{D} \cup \{(b, f(b))\}$
10: **end for**

4.4 Variance-Optimal Corrections for Neural Network Gradients

Theorem 4.4 characterizes the ideal control variate in gradient space, but does not yet specify a practical representation. For a neural network, we know the direction in which each parameter's gradient can move, and we use this to build a practical correction. Consider a linear weight matrix and one of its parameter rows r. At action j, the gradient within that row lies along the corresponding layer input. Let $h_{j,r}$ denote this local input direction, which is fixed by the prefix z_j and therefore does not depend on the candidate action v. The realizable row-wise subspace is $\mathcal{S}_{j,r} \coloneqq \mathrm{span}\{h_{j,r}\}$. Write $\xi_{j,r}$ variate for the component of ξ_j in row r. We therefore restrict the correction for row r

to the realizable subspace $\mathcal{S}_{j,r}$. Within this restricted family, Equation (8) shows that the variance-minimizing correction is a point in $\mathcal{S}_{j,r}$ closest to $\xi_{j,r}$. The remaining question is therefore how to compute this closest realizable correction, which is answered by the following theorem.

Theorem 4.5. *Define an operator* $\Pi_{j,r}(u) \coloneqq h_{j,r}h_{j,r}^\top u/\|h_{j,r}\|_2^2$, *with* $\Pi_{j,r}(u) = 0$ *when* $h_{j,r} = 0$. *Within the row-wise family, the variance minimization separates across* j *and* r*; moreover, the optimal correction is automatically centered, i.e.,*

$$\tilde{\Delta}^*_{j,r}(z_j, v) = \Pi_{j,r}\xi_{j,r}(z_j, v), \;\; \mathrm{Var}(\hat{g}_{\Delta^*}) = \mathrm{Var}(\hat{g}_Q) - \sum_{j=1}^{d} \mathbb{E}\Big[\sum_r \big\|\Pi_{j,r}\big(\xi_{j,r}(z_j, b_j)\big)\big\|_2^2\Big]. \quad (9)$$

See Appendix A.5 for proof. Theorem 4.5 gives a simple interpretation of the optimal row-wise correction. Each parameter row removes exactly the component of the residual noise $\xi_{j,r}$ that lies along a gradient direction that the row can produce. The orthogonal component cannot be removed within this row-wise family, while the removable variance is precisely the squared norm of the projection in Equation (9). Thus, the global variance-minimization problem reduces to independent one-dimensional projections, with no additional control-variate network or coupled covariance solve. Algorithm 1 summarizes the resulting procedure.

5 Experiment Settings

Section 4 shows that the action-value function Q removes decision-wise noise, while the correction Δ targets the remaining variance across decisions. We evaluate how much this additional correction helps in practice on three black-box problems: 1) a quadratic program (QP) from DFL, 2) a live LLM agent whose tools are selected by a language model, and 3) job-shop scheduling, where an LLM generates dispatching rules and a solver constructs the final schedule. Unless otherwise stated, results are averaged over multiple random seeds, and shaded regions indicate one standard deviation. See more experimental details in Section B.

Baselines. We compare our approach with different methods:

- **REINFORCE** (Williams, 1992): the vanilla score-function estimator with the same reward for all decisions.
- **State baseline** V: an action-independent prefix baseline, $V_j(z_j) = \sum_v p_j(v)Q_j(z_j, v)$.
- **RLOO** (Kool et al., 2019): uses rewards of other samples in the batch as a leave-one-out baseline.
- **GRPO** (Shao et al., 2024): normalizes each reward within the sampled group.
- **OTB** (Li et al., 2026): uses the optimal action-independent baseline at each token.
- **RELAX** (Grathwohl et al., 2018): a variance-trained control variate with a pathwise correction (LAX for the Gaussian predictor, RELAX for the discrete ones).

5.1 Portfolio Optimization (DFL)

We first consider a DFL problem in which multiple predictions share the same network and jointly determine a downstream decision. A shared network reads daily features and predicts the returns of $d = 12$ assets. Given a sampled return vector b, the *black-box oracle* solves a risk-averse portfolio optimization problem to obtain a portfolio $w^*(b)$, evaluates this portfolio under the true returns, and returns only a scalar normalized reward. The reward is the portfolio's mean-variance utility under the true returns, rescaled separately for each day. A reward of 1 corresponds to the portfolio obtained from the true returns, while 0 corresponds to the portfolio obtained from sign-flipped returns.

The predictor outputs a Gaussian distribution with fixed standard deviation, so the sum over v in Equation (3) becomes a Gaussian expectation that is available in closed form for a quadratic reward model. We use an offline dataset of predicted returns and oracle rewards to fit this quadratic model and update during policy training. The fitted model provides $\hat{Q}_j$ and the corresponding control variates without additional oracle calls. The action space is continuous and the downstream reward varies smoothly. Our method extends to continuous actions because its derivation only requires expectations under the action distribution, which are available in closed form here under the Gaussian policy and quadratic reward model.

5.2 LIVE MCP AGENT

We next consider a discrete black-box agent setting in which the reward is observed only after an autoregressive configuration has been executed. An LLM reads a task and generates the tool configuration for another black-box agent. The configuration has 4 slots, and each slot selects one of three candidate MCP servers or no server. The candidates are drawn from a pool of 12 servers (e.g., filesystem, git, sqlite, and pytest). We allow at most 2 of the 4 slots to contain servers, while the remaining slots are empty, to avoid attaching unnecessary tools. The agent then attempts the task using the selected servers, and the black box returns 1 only if the task is completed and every attached server is used. Thus, one decision corresponds to one slot, while the reward is observed only after the full configuration has been executed.

For the LLM predictor, we use the Llama-3.2-3B-Instruct model (Grattafiori et al., 2024) with rank-8 LoRA (Hu et al., 2022), and for the black-box oracle, we run through OpenAI's GPT 4.1-mini API. For each task, only 1–3 of the 256 configurations succeed. We use the four tasks that the base agent can solve and cache the reward of every configuration, allowing us to run 20 seeds and compute the exact $P(\text{success})$ throughout training.

5.3 JOB-SHOP SCHEDULING

Finally, we consider a combinatorial black-box setting where the reward is observed only after a complete dispatching plan is executed. An LLM reads a job-shop instance and generates a dispatching plan. The machines are partitioned into 4 groups, and the plan assigns one of 5 dispatch rules to each group, giving $5^4 = 625$ possible plans. Given a plan, the black-box oracle constructs the schedule by selecting ready operations according to each machine group's rule. It returns the instance lower bound divided by the resulting makespan, i.e., the total time to complete all jobs, giving a graded reward between 0 and 1. Thus, one decision selects one group's rule, while the reward is returned only after the full schedule is constructed. The four rules are generated autoregressively as a single JSON plan, so earlier choices can affect later ones.

6 DISCUSSION OF EXPERIMENTAL RESULTS

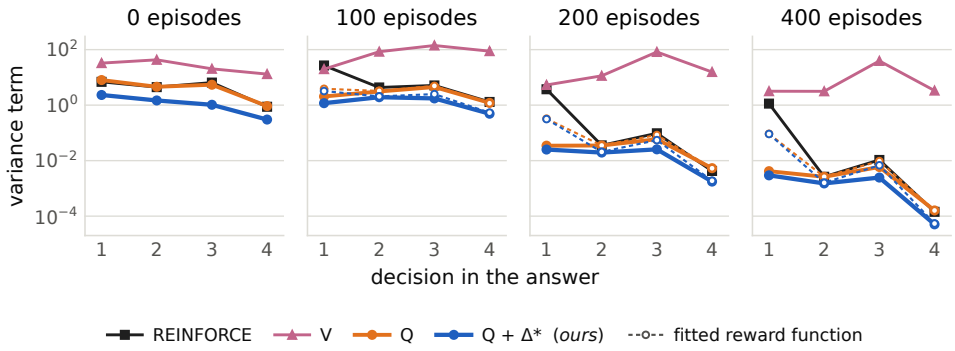


Figure 1: Variance across decisions in the LLM MCP task at different stages of training. The x-axis indexes the 4 decision slots, and the y-axis reports the corresponding variance on a log scale. Solid curves use the true reward table, while dotted curves use the fitted reward model.

When will we need an action-dependent baseline and where does the correction reduce variance for action-dependent baselines? Previously, Tucker et al. (2018) show that in practical RL settings, action-dependent baselines provide limited gains over action-independent ones, suggesting that their benefit can depend strongly on the problem structure. Here, our decomposition provides a complementary view by identifying the variance that remains after action-value baseline has solved the local credit-assignment problem.

Figure 1 directly tests the decomposition in Lemma 4.3. Each panel shows a different stage of training. It shows that the action-dependent baseline Q consistently reduces variance compared to the action-independent baseline V, indicating that local credit assignment matters in this task. Second, substantial variance can remain after conditioning on the current action, and our correction $Q + \Delta^*$ further reduces this residual, which achieves the lowest variance across decisions and training stages.

Third, the same ordering largely persists when Q and the correction are built from a fitted reward model rather than the true reward table, which suggests that the gains are not restricted to oracle reward information.

Does the variance reduction improve learning? We next ask whether the variance reductions in Figure 1 are reflected in optimization performance. Across all three tasks in Figure 2, the action-dependent baseline Q learns faster and reaches better solutions than REINFORCE and the standard baselines, while adding our correction yields a further improvement and typically reaches strong solutions with fewer black-box evaluations. The effect is particularly large on the QP and MCP tasks, where $Q + \Delta^*$ approaches the exact-gradient reference and learns substantially faster. On MCP, the corrected estimator is also more stable under aggressive optimization, whereas higher-variance estimators can collapse to policies from which successful trajectories are rarely observed. The gain is smaller on JobShop, where the graded reward provides a denser learning signal, but the same qualitative ordering remains. Overall, the reduction in residual variance observed in Figure 1 is consistently accompanied by faster and more stable learning.

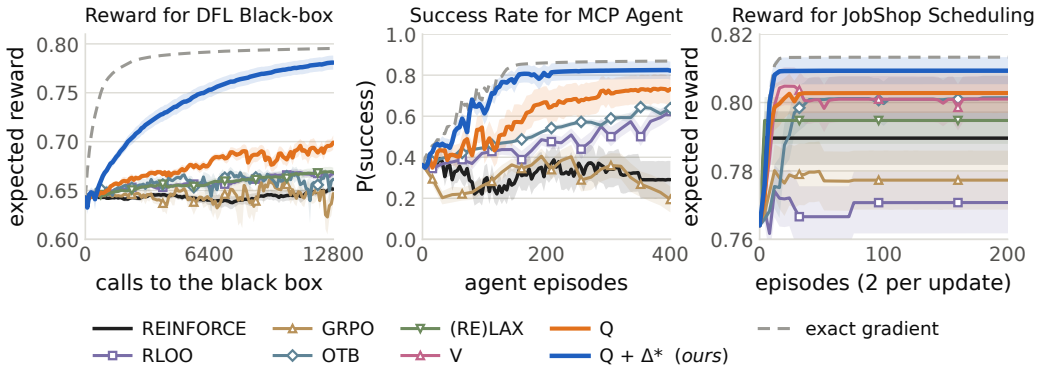


Figure 2: Training performance on three black-box tasks. We compare our methods with baselines. The exact gradient curve shows training with the exact gradient of the expected black-box objective and is included only as an oracle reference.

How much offline data is needed? Figure 3 varies the data used to fit the reward model. With too few episodes, the fitted baselines provide little benefit because successful trajectories are rarely observed. Once the budget reaches roughly one expected success per task, both Q and $Q + \Delta^*$ reduce variance substantially, well before the reward model is exact. The right panel shows that fitted Q can even outperform exact Q, since exact Q is only coordinate-wise optimal, whereas the gap for $Q + \Delta^*$ remains positive and approaches zero as the reward model improves.

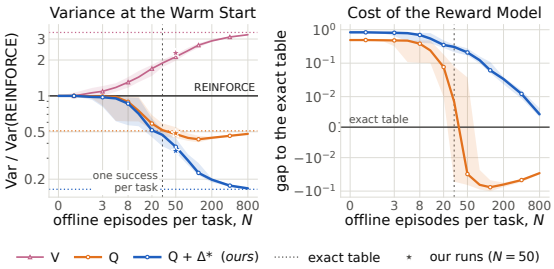


Figure 3: More offline data improves the fitted reward model and strengthens variance reduction.

7 Conclusion

We show that the action-value function can remain suboptimal for the full gradient when decisions share parameters. By decomposing the remaining variance across decisions, we derive a closed-form correction that reduces to simple row-wise projections for neural-network parameters. Across three black-box tasks, the correction consistently reduces variance beyond the action-value function and improves learning. Overall, variance reduction should target the full shared-parameter gradient rather than each decision in isolation.

AI USE STATEMENT

In this work, we used generative AI tools for assisting in the writing of proofs, designing or providing feedback on research methodology or experiments, implementing methods. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, proposing or refine hypotheses, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, interpret results. and assisting with translation are not applicable to this work. Additionally, we used generative AI tools for creating or modifying scientific figures or images, creating or editing software coding, and editing a research paper to improve readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We release an open-source repository containing all code, configuration files, and scripts needed to reproduce our results, including data generation and figure plotting. All proofs for the main paper are stated in the appendix with explanations and proper assumptions.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International conference on machine learning*, pp. 136–145. PMLR, 2017.

Anthropic. Introducing the Model Context Protocol. `https://www.anthropic.com/news/model-context-protocol`, 2024.

Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. Learning with differentiable pertubed optimizers. *Advances in neural information processing systems*, 33:9508–9519, 2020.

Ching-An Cheng, Xinyan Yan, and Byron Boots. Trajectory-wise control variates for variance reduction in policy gradient methods. In *Conference on Robot Learning*, pp. 1379–1394. PMLR, 2020.

Aleksandar Dimitriev and Mingyuan Zhou. Arms: Antithetic-reinforce-multi-sample gradient for binary variables. In *International conference on machine learning*, pp. 2717–2727. PMLR, 2021.

Zhe Dong, Andriy Mnih, and George Tucker. Disarm: An antithetic gradient estimator for binary latent variables. *Advances in neural information processing systems*, 33:18637–18647, 2020.

Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, 30, 2017.

Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management science*, 68(1): 9–26, 2022.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SyzKd1bCW`.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Evan Greensmith, Peter Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. In T. Dietterich, S. Becker, and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/584b98aac2dddf59ee2cf19ca4ccb75e-Paper.pdf.

Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop: Unbiased backpropagation for stochastic neural networks. In *International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1511.05176.

Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E. Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJ3rcZcxl.

J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods*. Monographs on Statistics and Applied Probability. Springer Dordrecht, 1964. doi: 10.1007/978-94-009-5819-7.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 REINFORCE samples, get a baseline for free!, 2019. URL https://openreview.net/forum?id=r1lgTGL5DE.

Yingru Li, Jiawei Xu, Ziniu Li, Jiacai Liu, Wei Liu, Yuxuan Tong, Longtao Zheng, Zhenghai Xue, Yaxiang Zhang, Tianle Cai, Ge Zhang, Qian Liu, and Baoxiang Wang. The optimal token baseline: Variance reduction for long-horizon LLM-RL. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=hpxI4Mndt4.

Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent control variates for policy optimization via stein identity. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1mCp-ZRZ.

Runjing Liu, Jeffrey Regier, Nilesh Tripuraneni, Michael Jordan, and Jon Mcauliffe. Rao-blackwellized stochastic gradients for discrete distributions. In *International Conference on Machine Learning*, pp. 4023–4031. PMLR, 2019.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

Jayanta Mandi, James Kotary, Senne Berden, Maxime Mulamba, Victor Bucarey, Tias Guns, and Ferdinando Fioretto. Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *Journal of Artificial Intelligence Research*, 80:1623–1701, 2024.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pp. 1791–1799. PMLR, 2014.

Andriy Mnih and Danilo Rezende. Variational inference for monte carlo objectives. In *International conference on machine learning*, pp. 2188–2196. PMLR, 2016.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

Mathias Niepert, Pasquale Minervini, and Luca Franceschi. Implicit mle: backpropagating through discrete exponential family distributions. *Advances in Neural Information Processing Systems*, 34:14567–14579, 2021.

John Paisley, David M. Blei, and Michael I. Jordan. Variational bayesian inference with stochastic search. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pp. 1363–1370, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

Marin Vlastelica Pogancic, Anselm Paulus, Vit Musil, Georg Martius, and Michal Rolinek. Differentiation of blackbox combinatorial solvers. In *ICLR*, 2020.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pp. 814–822. PMLR, 2014.

Sanket Shah, Kai Wang, Bryan Wilder, Andrew Perrault, and Milind Tambe. Decision-focused learning without decision-making: Learning locally optimized decision losses. *Advances in Neural Information Processing Systems*, 35:1320–1332, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Jiaxin Shi, Yuhao Zhou, Jessica Hwang, Michalis Titsias, and Lester Mackey. Gradient estimation with discrete stein operators. *Advances in neural information processing systems*, 35:25829–25841, 2022.

Mattia Silvestri, Senne Berden, Gaetano Signorelli, Ali İrfan Mahmutoğulları, Jayanta Mandi, Brandon Amos, Tias Guns, and Michele Lombardi. Score function gradient estimation to widen the applicability of decision-focused learning. *Journal of Artificial Intelligence Research*, 85, 2026.

Lennert De Smet, Emanuele Sansone, and Pedro Zuidberg Dos Martires. Differentiable sampling of categorical distributions using the catlog-derivative trick. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=AQyqxXctsN.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Michalis Titsias and Miguel Lazaro-Gredilla. Local expectation gradients for black box variational inference. *Advances in neural information processing systems*, 28, 2015.

Michalis Titsias and Jiaxin Shi. Double control variates for gradient estimation in discrete latent variable models. In *International Conference on Artificial Intelligence and Statistics*, pp. 6134–6151. PMLR, 2022.

Seiya Tokui and Issei Sato. Evaluating the variance of likelihood-ratio gradient estimators. In *International Conference on Machine Learning*, pp. 3414–3423. PMLR, 2017.

George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. *Advances in Neural Information Processing Systems*, 30, 2017.

George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. In *International conference on machine learning*, pp. 5015–5024. PMLR, 2018.

Lex Weaver and Nigel Tao. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, UAI'01, pp. 538–545, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558608001.

Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1658–1665, 2019.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1tSsb-AW.

Mingzhang Yin and Mingyuan Zhou. ARM: Augment-REINFORCE-merge gradient for stochastic binary networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1lg0jAcYm.

Arman Zharmagambetov, Brandon Amos, Aaron Ferber, Taoan Huang, Bistra Dilkina, and Yuandong Tian. Landscape surrogate: Learning decision losses for mathematical optimization under partial information. *Advances in Neural Information Processing Systems*, 36:27332–27350, 2023.

A Appendix

A.1 Proof for Lemma 4.1

Proof. For unbiasedness, conditioning on z_j gives

$$\mathbb{E}_{b_j|z_j}\left[\hat{Q}_j(z_j,b_j)s_j(b_j)\right] = \sum_{v\in\mathcal{V}} p_j(v)\hat{Q}_j(z_j,v)s_j(v), \tag{10}$$

so the two terms involving $\hat{Q}_j$ cancel in expectation.

For the second claim, consider any perturbation $Q_j+\delta_j$. Let $\hat{g}_{Q,j}$ denote the j-th summand obtained with the exact Q_j, and define

$$u_j := \delta_j(z_j,b_j)s_j(b_j) - \sum_v p_j(v)\delta_j(z_j,v)s_j(v). \tag{11}$$

Then

$$\hat{g}_{Q+\delta,j} = \hat{g}_{Q,j} - u_j, \qquad \mathbb{E}[u_j|z_j] = 0. \tag{12}$$

At the exact conditional mean,

$$\hat{g}_{Q,j} - \mathbb{E}[\hat{g}_{Q,j}|z_j] = \big(f(b) - Q_j(z_j,b_j)\big)s_j(b_j). \tag{13}$$

Since $\mathbb{E}[f - Q_j|z_j,b_j] = 0$, this residual is conditionally orthogonal to u_j. Hence

$$\mathrm{Var}\big(\hat{g}_{Q+\delta,j}|z_j\big) = \mathrm{Var}\big(\hat{g}_{Q,j}|z_j\big) + \mathbb{E}\big[\|u_j\|_2^2|z_j\big], \tag{14}$$

which is minimized by the exact conditional mean (up to perturbations that induce the same zero correction). □

A.2 Arbitrarily worse

Example 4.2 shows that an exact conditional value function need not be variance-optimal. This appendix strengthens the statement in the two directions that matter for the claim in the abstract. First, the suboptimality can be made arbitrarily large while both variances stay positive. Second, in the worst case, the exact value function can remove *none* of the REINFORCE variance while the optimal correction removes *all* of it. Both constructions are autoregressive, so they are instances of the setting that Lemma 4.3 and Theorem 4.5 are stated for, and in both the value function is exact rather than fitted.

Proposition A.1 (The gap is unbounded)**.** *Consider the autoregressive setting with the exact conditional value function Q, and let $\hat{g}_{\mathrm{row}}$ be the optimal row-wise estimator of Theorem 4.5.*

1. *For every $\rho \in [0, 1)$ there is an instance with $d = 2$ binary decisions and a single shared weight vector in which*

$$\mathrm{Var}(\hat{g}_Q) = \tfrac{1}{2}, \qquad \mathrm{Var}(\hat{g}_{\mathrm{row}}) = \tfrac{1}{2}\big(1 - \rho^2\big), \qquad \textit{so} \qquad \frac{\mathrm{Var}(\hat{g}_Q)}{\mathrm{Var}(\hat{g}_\Delta)} = \frac{1}{1 - \rho^2},$$

   where ρ is the cosine between the two decisions' input directions. Both variances are strictly positive and the ratio is unbounded as $\rho \to 1$.

2. *For every $d \geq 2$ there is an instance with d binary decisions and one shared parameter in which*

$$\mathrm{Var}(\hat{g}_{\mathrm{SF}}) = \mathrm{Var}(\hat{g}_Q) = \frac{d}{4}, \qquad \mathrm{Var}(\hat{g}_{\Delta^*}) = 0.$$

   The exact value function removes no variance at all, for any horizon, while the optimal correction of the same family removes all of it.

Proof. At the stated parameter values every decision has $p_j(1) = \frac{1}{2}$, so $s_j(v) = (v - \frac{1}{2})\, h_j$ in part 1 and $s_j(v) = v - \frac{1}{2}$ in part 2. We use the identity $(2v - 1)(v - \frac{1}{2}) = \frac{1}{2}$ for $v \in \{0, 1\}$ throughout.

Part 1. With prefix conditioning, the exact conditional means are

$$Q_1(v) = \mathbb{E}[f \mid b_1 = v] = (2v - 1)\, \mathbb{E}[2b_2 - 1] = 0, \qquad Q_2(b_1, v) = (2b_1 - 1)(2v - 1) = f(b_1, v).$$

In Equation (3), the correction term of $j = 1$ vanishes because $Q_1 \equiv 0$, and term (a) of $j = 2$ vanishes because $Q_2 = f$. The two surviving pieces are

$$f(b)\, s_1(b_1) = (2b_1 - 1)(2b_2 - 1)\big(b_1 - \tfrac{1}{2}\big)\, h_1 = \tfrac{1}{2}(2b_2 - 1)\, h_1,$$
$$\sum_v \tfrac{1}{2}\, Q_2(b_1, v)\, s_2(v) = \tfrac{1}{2}(2b_1 - 1) \sum_v (2v - 1)\big(v - \tfrac{1}{2}\big)\, h_2 = \tfrac{1}{2}(2b_1 - 1)\, h_2,$$

so that

$$\hat{g}_Q = \tfrac{1}{2}(2b_2 - 1)\, h_1 + \tfrac{1}{2}(2b_1 - 1)\, h_2, \qquad g = \mathbb{E}[\hat{g}_Q] = 0.$$

The residual increments of Equation (6) are therefore

$$\xi_1(b_1) = \mathbb{E}[\hat{g}_Q \mid b_1] - g = \tfrac{1}{2}(2b_1 - 1)\, h_2, \qquad \xi_2(b_1, b_2) = \hat{g}_Q - \mathbb{E}[\hat{g}_Q \mid b_1] = \tfrac{1}{2}(2b_2 - 1)\, h_1,$$

and Lemma 4.3 gives $\mathrm{Var}(\hat{g}_Q) = \mathbb{E}\|\xi_1\|^2 + \mathbb{E}\|\xi_2\|^2 = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$. Note that ξ_1 lies along h_2, the input direction of the *other* decision: the noise left at decision 1 is entirely the effect of b_1 on decision 2's gradient term.

There is a single row, so $\Pi_j = h_j h_j^\top$ and

$$\Pi_1 \xi_1 = \tfrac{\rho}{2}(2b_1 - 1)\, h_1 = \rho\, s_1(b_1), \qquad \Pi_2 \xi_2 = \tfrac{\rho}{2}(2b_2 - 1)\, h_2 = \rho\, s_2(b_2), \qquad \|\Pi_1 \xi_1\|^2 = \|\Pi_2 \xi_2\|^2 = \tfrac{\rho^2}{4}.$$

Theorem 4.5 then yields

$$\mathrm{Var}(\hat{g}_{\mathrm{row}}) = \tfrac{1}{2} - \tfrac{\rho^2}{4} - \tfrac{\rho^2}{4} = \tfrac{1}{2}\,(1 - \rho^2), \qquad \hat{g}_{\mathrm{row}} = \hat{g}_Q - \rho\, s_1(b_1) - \rho\, s_2(b_2).$$

The optimal row-wise correction is thus a baseline of value ρ on each decision on top of Q. Its value is set by the cross-decision alignment; the classical per-decision baseline criterion would set it to zero here, since $\mathbb{E}[f - Q_j \mid z_j, b_j] = 0$.

Part 2. Write $P \coloneqq \prod_{k<d}(2b_k - 1) \in \{\pm 1\}$. Because the factors $2b_k - 1$ are independent with mean zero,

$$Q_j(b_{<j}, v) = \prod_{k<j}(2b_k - 1)\,(2v - 1) \prod_{k>j} \mathbb{E}[2b_k - 1] = 0 \quad (j < d), \qquad Q_d(b_{<d}, v) = P\,(2v - 1).$$

For REINFORCE, $f^2 \equiv 1$ and $\mathbb{E}[f(b)(b_j - \frac{1}{2})] = 0$ for $d \geq 2$, so

$$\mathrm{Var}(\hat{g}_{\mathrm{SF}}) = \mathbb{E}\Big[\Big(\sum_{j=1}^{d} \big(b_j - \tfrac{1}{2}\big)\Big)^2\Big] = \frac{d}{4}.$$

For $\hat{g}_Q$, each summand $j < d$ reduces to $f(b)\, s_j(b_j)$ (both Q_j and its correction term vanish), while the summand $j = d$ reduces to its correction term $\sum_v \frac{1}{2}\, P\,(2v-1)(v-\frac{1}{2}) = \frac{P}{2}$. Hence

$$\hat{g}_Q = f(b) \sum_{j<d} \big(b_j - \tfrac{1}{2}\big) + \frac{P}{2} = P\Big[(2b_d - 1) \sum_{j<d} \big(b_j - \tfrac{1}{2}\big) + \frac{1}{2}\Big], \qquad \mathbb{E}[\hat{g}_Q] = 0 = g,$$

and, since $P^2 = (2b_d - 1)^2 = 1$ and b_d is independent of $b_{<d}$ with $\mathbb{E}[2b_d - 1] = 0$,

$$\mathbb{E}[\hat{g}_Q^2] = \mathbb{E}\Big[\Big(\sum_{j<d} \big(b_j - \tfrac{1}{2}\big)\Big)^2\Big] + \mathbb{E}\Big[(2b_d - 1) \sum_{j<d} \big(b_j - \tfrac{1}{2}\big)\Big] + \frac{1}{4} = \frac{d-1}{4} + 0 + \frac{1}{4} = \frac{d}{4}.$$

Finally, θ is a scalar, so every $h_{j,r} = 1$ and the row-wise family is unrestricted; Theorem 4.5 with **??** gives $\mathrm{Var}(\hat{g}_{\mathrm{row}}) = \mathrm{Var}(\hat{g}_Q) - \sum_j \mathbb{E}\|\xi_j\|^2 = 0$. □

A.3 How can we compute the ξ_j?

Lemma A.2. *Let $\hat{g}_{Q,>j} := \sum_{i>j} \hat{g}_{Q,i}$. For an autoregressive model, revealing b_j can affect gradient terms both before and after j, i.e.,*

$$\xi_j(z_j, v) = (Q(z_j, v) - \sum_w p_j(w) Q(z_j, w)) \sum_{i<j} s_i(b_i) + \mathbb{E}[\hat{g}_{Q,>j} | z_j, b_j = v] - \mathbb{E}[\hat{g}_{Q,>j} | z_j]. \tag{15}$$

Proof. We decompose $\hat{g}_Q$ by time step. For $i < j$, $s_i(b_i)$ and $Q_i(z_i, b_i)$ are determined by $b_{\le j}$, and the j-th correction term $\sum_w p_j(w) Q_j(z_j, w) s_j(w)$ is determined by z_j; the only remaining random object in the summands $i \le j$ is $f(b)$, whose conditional mean is $Q_j(v)$ by Equation (2). Conditioning the summands $i \le j$ of Equation (3) on $(z_j, b_j = v)$ therefore gives

$$\mathbb{E}\Big[\sum_{i\le j} \hat{g}_{Q,i} \,\Big|\, z_j, b_j = v\Big] = Q_j(v) \underbrace{\sum_{i<j} s_i(b_i)}_{S_{<j}} \underbrace{- \sum_{i<j} Q_i(z_i, b_i) s_i(b_i) + \sum_{i\le j} \sum_w p_i(w) Q_i(z_i, w) s_i(w)}_{:= R_j \text{ (does not depend on } v)}, \tag{16}$$

where term (a) of the j-th summand has zero conditional mean and was dropped. Thus, we can rewrite Equation (3) as

$$\mathbb{E}[\hat{g}_Q | z_j, b_j = v] = R_j + Q_j S_{<j} + D_j, \tag{17}$$

where $D_j(v) = \mathbb{E}[\sum_{i>j}((f(b) - Q_i(z_i, b_i)) s_i(b_i) + \sum_w p_i(w) Q_i(z_i, w) s_i(w)) | z_j, b_j = v]$ collects the remaining conditional expected gradient after choosing v. Similarly, we have

$$\mathbb{E}[\hat{g}_Q | z_j] = R_j + V_j S_{<j} + \bar{D}_j, \tag{18}$$

where $\bar{D}_j = \sum_v p_j(v) D_j(v)$. Thus, we have

$$\xi_j(z_j, v) \coloneqq \mathbb{E}[\hat{g}_Q | z_j, b_j = v] - \mathbb{E}[\hat{g}_Q | z_j] = (Q_j - V_j) S_{<j} + D_j - \bar{D}_j. \tag{19}$$

□

The first term, $(Q_j - V_j) S_{<j}$, arises because observing b_j changes the credit assigned to the earlier score terms; the second, $D_j - \bar{D}_j$, because b_j changes the distribution, and hence the expected gradient contributions, of the later decisions. Q removes the randomness local to decision j (there is no $i = j$ term), but both cross-decision effects remain. Note that both terms are linear combinations of scores $s_i(\cdot)$ with $i \neq j$, which is what Corollary A.4 uses.

A.4 Proof for Theorem 4.4

Proof. Since $\hat{g}_\Delta = \hat{g}_Q - \sum_j \tilde{\Delta}_j$ is unbiased,

$$\begin{aligned}\mathrm{Var}(\hat{g}_\Delta) &= \mathbb{E}[\|e_Q - \sum_{j=1}^{d} \tilde{\Delta}_j\|_2^2] \\ &= V_Q - 2\sum_{j=1}^{d} \mathbb{E}[e_Q^\top \tilde{\Delta}_j] + \mathbb{E}[\|\sum_{j=1}^{d} \tilde{\Delta}_j\|_2^2]. \end{aligned} \tag{20}$$

For $i < j$, $\tilde{\Delta}_i$ is determined by $z_j = b_{<j}$, while $\mathbb{E}[\tilde{\Delta}_j \mid z_j] = 0$. Hence

$$\mathbb{E}[\tilde{\Delta}_i^\top \tilde{\Delta}_j] = \mathbb{E}[\tilde{\Delta}_i^\top \mathbb{E}[\tilde{\Delta}_j \mid z_j]] = 0, \tag{21}$$

so

$$\mathbb{E}[\|\sum_{j=1}^{d} \tilde{\Delta}_j\|_2^2] = \sum_{j=1}^{d} \mathbb{E}[\|\tilde{\Delta}_j\|_2^2]. \tag{22}$$

Next, since $\tilde{\Delta}_j$ depends only on (z_j, b_j),

$$\mathbb{E}[e_Q^\top \tilde{\Delta}_j] = \mathbb{E}[\mathbb{E}[e_Q \mid z_j, b_j]^\top \tilde{\Delta}_j]. \tag{23}$$

By the definition of ξ_j,

$$\mathbb{E}[e_Q \mid z_j, b_j] = \xi_j(z_j, b_j) + \mathbb{E}[\hat{g}_Q \mid z_j] - g. \tag{24}$$

The last two terms depend only on z_j, and therefore vanish after taking the inner product with $\tilde{\Delta}_j$. Thus,

$$\mathbb{E}[e_Q^\top \tilde{\Delta}_j] = \mathbb{E}[\xi_j^\top \tilde{\Delta}_j]. \tag{25}$$

Substituting Equations (22) and (25) into Equation (20) gives

$$\mathrm{Var}(\hat{g}_\Delta) = V_Q + \sum_{j=1}^{d} \mathbb{E}[\|\tilde{\Delta}_j\|_2^2 - 2\xi_j^\top \tilde{\Delta}_j]. \tag{26}$$

Finally, using

$$V_Q = \sum_{j=1}^{d} \mathbb{E}[\|\xi_j\|_2^2], \tag{27}$$

we obtain

$$\mathrm{Var}(\hat{g}_\Delta) = \sum_{j=1}^{d} \mathbb{E}[\|\xi_j - \tilde{\Delta}_j\|_2^2]. \tag{28}$$

Therefore, the unrestricted minimum is attained at

$$\tilde{\Delta}_j^*(z_j, v) = \xi_j(z_j, v), \tag{29}$$

for each j. □

A.5 Proof for Theorem 4.5

Proof. For a fixed (j, r), Equation (8) reduces the optimization to

$$\arg\min_{u \in \mathcal{S}_{j,r}} \|u - \xi_{j,r}(z_j, v)\|_2^2,$$

whose solution is the Euclidean projection

$$\tilde{\Delta}^*_{j,r}(z_j, v) = \Pi_{j,r}\xi_{j,r}(z_j, v).$$

Since $\Pi_{j,r}$ depends on the prefix and row but not on v,

$$\sum_v p_j(v)\tilde{\Delta}^*_{j,r}(z_j, v) = \Pi_{j,r}\sum_v p_j(v)\xi_{j,r}(z_j, v) = 0,$$

where the last equality follows from Equation (7). Thus, the projected correction is already centered.

Finally, by the Pythagorean identity,

$$\|\xi_{j,r} - \Pi_{j,r}\xi_{j,r}\|_2^2 = \|\xi_{j,r}\|_2^2 - \|\Pi_{j,r}\xi_{j,r}\|_2^2.$$

Summing over rows and decisions and using **??** gives the stated variance formula. □

A.6 Additional Remarks

Remark A.3 (The unrestricted optimum is only a benchmark). Setting $\tilde{\Delta}_j = \xi_j$ removes all remaining noise in $\hat{g}_Q$ and therefore gives zero variance. However, computing ξ_j requires exact conditional expectations of the black-box reward, so this choice is not practical. We therefore use it only as an oracle benchmark.

In practice, the correction is restricted by the network gradient structure. Under this restriction, Theorem 4.4 shows that the global variance-minimization problem decomposes into simple row-wise problems, leading to Theorem 4.5. All optimality claims in this paper refer to this realizable family.

Corollary A.4 (When Q is already optimal, and a diagnostic)**.** *Suppose that, for every row r and every prefix, the input direction $h_{j,r}$ of decision j is orthogonal to the input directions $h_{i,r}$ of all other decisions $i \neq j$. Then $\Pi_{j,r}\xi_{j,r} = 0$ for all (j, r), so $\mathrm{Var}(\hat{g}_{\Delta^*}) = \mathrm{Var}(\hat{g}_Q)$. This means Q is variance-optimal within the row-wise family. This holds in particular for disjoint per-decision parameters, recovering Tokui & Sato (2017) and the orthogonal-score condition of Wu et al. (2018); conversely, in Proposition A.1(1) the removed fraction of $\mathrm{Var}(\hat{g}_Q)$ is ρ^2.*

Proof. By Lemma A.2, ξ_j is a linear combination of scores $s_i(\cdot)$ of the other decisions $i \neq j$ (averaged over continuations for $i > j$), whose row-r components lie in $\mathrm{span}\{h_{i,r}\}$; under the orthogonality assumption their projections onto $\mathcal{S}_{j,r}$ vanish, so $\Pi_{j,r}\xi_{j,r} = 0$ and Equation (9) gives $\mathrm{Var}(\hat{g}_{\Delta^*}) = \mathrm{Var}(\hat{g}_Q)$. When decisions use disjoint parameters, $h_{j,r}$ and $h_{i,r}$ are supported on disjoint coordinate blocks and are therefore orthogonal. The second claim is computed in the proof of Proposition A.1. □

Remark A.5 (Computational cost). Line 7 of Algorithm 1 requires $\mathbb{E}_{\hat{f}}[\hat{g}_{\hat{Q}} \mid z_j, b_j = v]$ for every (j, v), which by Lemma A.2 involves an expectation over the continuations $b_{>j}$ under p_θ. In our experiments the continuations are enumerated exactly (at most 625 sequences, or a closed-form Gaussian–quadratic expectation), so the correction costs no oracle call but $\mathcal{O}(d|\mathcal{V}|)$ evaluations of the fitted model per step, and the enumeration grows exponentially with d. For long horizons one can instead draw, independently of the realized b, a single continuation per (j, v) under p_θ and use its value of $\hat{g}_{\hat{Q}}$ in place of the expectation, at $\mathcal{O}(d|\mathcal{V}|)$ model rollouts per step. Because the same sampled continuations enter both terms of Equation (4), the resulting $\tilde{\Delta}_j$ still has zero mean given z_j, so $\hat{g}$ remains unbiased; only the variance reduction degrades. We leave this variant to future work.

B Additional Experimental Details

The exact gradient. The dashed line in every training figure is $\nabla_\theta \mathbb{E}[f]$ itself. On the QP the regret is a quadratic in the prediction, so the Gaussian expectation is closed-form; on the agent there are 256 configurations per task and the reward of each is cached, so we enumerate them exactly. It is the ceiling of every method, and it costs no calls. The same enumeration gives the exact $\mathrm{Var}(\hat{g})$ of every estimator at the current policy, which is what Figure 1 plots, and the exact $P(\text{success})$, which is what we report instead of a sampled success rate.

Portfolio Optimization from DFL The offline dataset has 300 calls, 50 per day. The reward model of each day is a quadratic in b fitted by least squares to the offline calls and the calls paid on that day, with the ridge picked by leave-one-out cross-validation. LAX starts its surrogate from the same dataset; the other baselines have no use for it. We then train with 12,800 calls and 10 seeds. RLOO, GRPO and OTB draw eight samples per step, and LAX refits its surrogate on the day's last 128 samples.

Job-shop task and synthetic data Each of n jobs visits all m machines once in a fixed order. Operations are non-preemptive, and each machine processes at most one operation at a time. The goal is to minimize the makespan, when all jobs finish. Synthetic instance families are summarized in Table 2. Machines are divided into four groups by $g(k) = \lfloor 4k/m \rfloor$, where k is the zero-based machine index. The planner selects one of five rules per group, yielding $5^4 = 625$ plans:

- `est_spt`: earliest start, then shortest processing time.
- `job_ready`: earliest job availability.
- `ratio`: smallest $(e+1)/(w+1)$, where e is the earliest start and w is the job's remaining processing time.
- `fifo`: smallest next-operation index.
- `completion`: earliest prospective completion.

At each step, the constructor scores each job's next unscheduled operation using its machine group's rule and selects the smallest priority key. Ties use job index; postprocessing is disabled.

The reward is the workload lower bound divided by the schedule's makespan. This bound is the larger of the maximum total processing time of any job and the maximum workload on any machine. Higher reward indicates a shorter schedule relative to this bound, without requiring the optimal makespan to be known.

Table 2: Synthetic job-shop families. Integer ranges are inclusive; draws are independent except for the shared quantities specified below.

Family	Generation rule
Uniform	Uniform random machine permutation per job; durations uniform in 1–99.
Bottleneck	As Uniform, but multiply durations on one uniformly chosen machine by 4.
Heterogeneous	Uniform routes; durations uniform in 1–25, multiplied by a shared per-job factor uniform in $\{1, 2, 4\}$ and by a further 4 independently with probability 0.15 per operation.
Correlated routes	One shared uniform random route, with $\max(1, \lfloor m/5 \rfloor)$ uniformly sampled swaps of distinct positions per job; durations uniform in 1–99.

Each split contains 128 distinct instances: four per size–family combination across eight sizes and four families. Training, validation, and test share these settings. All instances are checked for positive integer durations and valid machine routes.

C Additional Related Works

Decision-focused learning. DFL trains the predictor with the loss of the induced decision (Elmachtoub & Grigas, 2022). For more detailed introduction to DFL, we refer the readers to (Mandi et al., 2024). Most DFL methods differentiate through the optimization problem (Amos & Kolter, 2017; Donti et al., 2017; Wilder et al., 2019) or through a smoothed version of it Pogancic et al. (2020); Berthet et al. (2020); Niepert et al. (2021), which needs access to the structure of the problem. When the problem is a black-box, a common approach is to learn a differentiable surrogate of the decision loss (Shah et al., 2022; Zharmagambetov et al., 2023). In this work, we also learn a model of the loss, but we only use it to reduce the variance of an unbiased estimator, so its error never introduces a bias but only affects the variance.