DISTRIBUTIONAL BAYESIAN OPTIMAL EXPERIMENTAL DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential Bayesian optimal experimental design (BOED) chooses informative experiments adaptively based on the outcomes of prior experiments. The standard sequential BOED objective, expected information gain (EIG), scores an experimental design policy by the mean of its information gain (IG) distribution. However, the mean alone can obscure important characteristics of the distribution, such as its variance or lower-tail behavior. To address this limitation, we present DistBOED, a distributional learning method for sequential BOED that models the full distribution of IG induced by a design policy. DistBOED uses a quantile critic and a campaign-level actor update to learn policies for objectives beyond mean IG. We demonstrate DistBOED across three different scenarios with various risk-sensitive IG measures including: conditional value at risk, variance-penalized mean, and entropic risk measure. In each case we observe that accounting for higher-order aspects of the IG distribution yields more robust design policies than EIG alone.

1 INTRODUCTION

In many experimental settings, scientists are interested in gathering data to reduce their uncertainty about a quantity of interest (QoI). Since more informative data will yield stronger updates about the QoI, and since experimenters must make many choices about experimental configuration that will lead to different observations, it is natural to ask whether there are some configurations that yield more informative data than others. Bayesian optimal experimental design (BOED) is a framework that formalizes this question and allows any potential experimental configuration, called a *design*, to be scored based on how much information it provides (Lindley, 1956).

Since one experimental configuration could produce many possible observations, each of which provides a different amount of information about the QoI, a design induces a probability distribution over measured information gain (IG) values. Consequently, the typical method for scoring a design is the Expected Information Gain (EIG), which is the expectation of the IG distribution induced by this design (Rainforth et al., 2024). While the mean is a straightforward summary of the IG distribution, it is not always the most helpful statistic for the experimenter. Indeed, two designs may induce distributions with the same mean, but may differ in their IG distribution.

Limitations of the mean as a summary statistic are particularly important in sequential BOED, where experimental choices can affect the menu of later choices. For example, a high-intensity measurement may provide precise data but risk destroying the specimen, ending the campaign. An EIG objective can favor this risky design when highly informative campaigns compensate for those that end with little information due to the loss of the specimen. In this setting, the object being optimized is a *policy* that maps the history of designs and observations to the distribution over the next design. Each observation is random, and since each later design depends on previous observations, the IG over the course of the whole experimental campaign can be highly variable. In spite of this, standard sequential BOED methods learn policies by maximizing estimates of EIG (Foster et al., 2021; Shen et al., 2025). There is a shortage of methods that choose a sequential design policy for a risk-sensitive objective, such as one that rewards a policy for rarely producing an uninformative dataset rather than for its average information.

To this end, we introduce DistBOED, a framework for sequential BOED that uses methods from distributional reinforcement learning (Bellemare et al., 2017) to learn the distribution of IG induced by a design policy. DistBOED allows practitioners to specify preferences over information outcomes

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.

through a functional of this distribution, and then optimizes policies for these preferences. DistBOED stands in contrast to standard methods, which are restricted to EIG or some approximation of it. We use this framework to train policies for EIG, conditional-value-at-risk (Rockafellar et al., 2000), entropic risk (Howard & Matheson, 1972), and a variance-penalized mean (Mannor & Tsitsiklis, 2011). Each of these objectives represents different preferences about the tradeoff between how informative experimental campaigns are on average, how variable their outcomes are, and how often they yield little information. Our contributions in this work are as follows:

1. We present DistBOED, a distributional actor-critic algorithm that learns the distribution of IG over a sequence of experiments and can learn sequential design policies that are optimized for an arbitrary functions of the return distribution;
2. We demonstrate that DistBOED with risk-sensitive objectives produces policies that reduce dispersion or improve lower-tail IG relative to mean-seeking policies;
3. We evaluate DistBOED in three experimental settings with different sources of risk: source localization under risk of sensor jamming, inference under a misspecified spring-mass model, and destructive experiments that can prematurely end the experimental campaign.

2 PRELIMINARIES

2.1 BAYESIAN OPTIMAL EXPERIMENTAL DESIGN (BOED)

Let θ denote the QoI in an experiment, with initial uncertainty given by the prior: $p(\theta)$. The experimental design we denote $d \in \mathcal{D}$, where $\mathcal{D}$ is the space of all possible designs, influences the observed data y via the likelihood function $p(y \mid \theta, d)$, which describes the data-generating process. Once data are observed, the prior belief is updated to the posterior $p(\theta \mid y, d)$ using Bayes' rule. The IG is given by the Kullback-Leibler divergence of the posterior from the prior:

$$G(y, d) = D_{\mathrm{KL}}\left[p(\theta \mid y, d) || p(\theta)\right] = \mathbb{E}_{p(\theta|y,d)}\left[\log \frac{p(\theta \mid y, d)}{p(\theta)}\right]. \tag{1}$$

Because the observation is not known before the experiment is performed, BOED typically scores a design by its expected IG (Lindley, 1956):

$$\mathrm{EIG}(d) = \mathbb{E}_{p(y|d)}[G(y, d)] = \mathbb{E}_{p(y,\theta|d)}\left[\log \frac{p(\theta \mid y, d)}{p(\theta)}\right]. \tag{2}$$

The EIG is equivalent to the mutual information between θ and y under design d. One-step BOED selects the design that maximizes EIG: $d^* = \arg\max_d \mathrm{EIG}(d)$.

Sequential BOED chooses T experiments, where each design is selected after seeing the outcomes of the experiments before it. We let (d_t, y_t) be the design and observation, respectively, of experiment t, and then we let $h_t = \{d_1, y_1, \ldots, d_t, y_t\}$ be the history after t experiments, with $h_0 = \emptyset$. A *policy* π maps each history to to a distribution over the next design, from which the next design is sampled: $d_t \sim \pi(\cdot \mid h_{t-1})$, and we call a sequence of T experiments a *campaign*. The posterior after a campaign is $p(\theta \mid h_T) \propto p(\theta) \prod_{t=1}^{T} p(y_t \mid \theta, d_t, h_{t-1})$, and the IG over the campaign is the KL divergence between the prior and this posterior:

$$G(h_T) = D_{\mathrm{KL}}[p(\theta \mid h_T) \,\|\, p(\theta)]\,. \tag{3}$$

A policy induces a distribution over histories given by:

$$p_\pi(h_T) = \int p(\theta) \prod_{t=1}^{T} \pi(d_t \mid h_{t-1}) p(y_t \mid \theta, d_t, h_{t-1})\, d\theta. \tag{4}$$

Since $G(h_T)$ is a random variable, it has a distribution F_π induced by the policy π. The mean of this distribution is the sequential EIG:

$$\mathrm{EIG}(\pi) = \mathbb{E}_{p_\pi(h_T)}[G(h_T)] = \mathbb{E}_{p(\theta)p_\pi(h_T|\theta)}\left[\log \frac{p_\pi(h_T \mid \theta)}{p_\pi(h_T)}\right] \tag{5}$$

which is the mutual information between θ and h_T (Foster et al., 2021).

2.2 Risk-Sensitive Functions of the return distribution

EIG is one function of F_π, but it does not explicitly account for risk (e.g. it is "risk-neutral"). In this work we consider three utilities that account for lower-tail outcomes or dispersion, which the mean does not capture. Let $G \sim F_\pi$ and let F_π^{-1} denote its quantile function. The **conditional value at risk (CVaR)** for a level $\alpha \in (0, 1]$ is given by:

$$\mathrm{CVaR}_\alpha(G) = \frac{1}{\alpha} \int_0^\alpha F_\pi^{-1}(u)\, du, \tag{6}$$

which averages the lowest α fraction of return probability (Rockafellar & Uryasev, 2000). For a continuous distribution, these are the outcomes below the 100αth percentile. Letting $\alpha = 1$ recovers the mean, and as $\alpha \to 0$ it approaches the essential infimum of the return. The **entropic risk measure** for a coefficient $\beta > 0$ is given by:

$$\mathrm{ERM}_\beta(G) = -\frac{1}{\beta} \log \mathbb{E}\left[e^{-\beta G}\right], \tag{7}$$

which tilts the distribution toward its low outcomes (Howard & Matheson, 1972). It satisfies $\mathrm{ERM}_\beta(G) \leq \mathbb{E}[G]$, as $\beta \to 0$ then $\mathrm{ERM}_\beta(G) \to \mathbb{E}[G]$, and when the moment-generating function is finite in a neighborhood of zero, $\mathrm{ERM}_\beta(G) = \mathbb{E}[G] - (\beta/2)\mathrm{Var}(G) + O(\beta^2)$ for small β. Note that for $\beta < 0$, the entropic risk measure acts as a risk-seeking utility, but such risk-seeking utilities are outside the scope of this work. Finally, for a weight $\lambda > 0$ the **variance-penalized mean** is:

$$\mathrm{MV}_\lambda(G) = \mathbb{E}[G] - \lambda \mathrm{Var}[G]. \tag{8}$$

This objective (Mannor & Tsitsiklis, 2011) trades EIG against variability across campaigns. Unlike CVaR, it penalizes deviations both above and below the mean. This criterion was studied for one-step BOED in Shen & Huan (2026). Provided the required moments are finite, each risk measure is a function of F_π, and for each a larger value is preferred.

2.3 Distributional reinforcement learning

Standard reinforcement learning estimates the expected return from a state and action. Distributional reinforcement learning instead models the entire distribution of that return (Bellemare et al., 2023). This representation describes variability and tail behavior of the return that the expectation alone does not capture. For a fixed policy π, let $Z_\pi(s, a)$ denote the random cumulative reward obtained by taking action a in state s and following π thereafter. Its distribution satisfies the distributional Bellman equation:

$$Z_\pi(s, a) \overset{D}{=} R + \gamma Z_\pi(S', A'), \quad A' \sim \pi(\cdot \mid S'), \tag{9}$$

where R is the immediate reward, S' is the next state conditional on (s, a), γ is the discount factor, and $\overset{D}{=}$ denotes equality in distribution. Let $F_\pi(\cdot \mid s, a)$ denote the conditional return distribution of $Z_\pi(s, a)$. The right-hand side is constructed by sampling

$$(R, S') \sim p(\cdot, \cdot \mid s, a), \qquad A' \sim \pi(\cdot \mid S'), \qquad Z' \sim F_\pi(\cdot \mid S', A'), \tag{10}$$

and forming the return $R + \gamma Z'$. A distributional critic approximates these conditional return distributions and is trained using sampled Bellman updates. Common representations include categorical (Bellemare et al., 2017) and quantile (Dabney et al., 2018b;a). In the quantile representation, each prediction estimates a specified percentile of the distribution. Together, the predictions approximate the distribution and allow functions of the return to be computed or approximated.

3 DistBOED

DistBOED learns a design policy π to maximize a chosen utility $\mathcal{U}(F_\pi)$ of the campaign return distribution. Training uses a simulator to generate complete experimental campaigns and compute their terminal IG. A distributional critic predicts the terminal IG distribution conditional on the observed history and a proposed design. The actor combines these predictions with observed campaign returns to improve the chosen objective. We first describe how the critic learns conditional return distributions, then how the chosen campaign objective determines design scores and actor updates. Appendix A provides the complete training procedures and an overview of the architecture. See Appendix A for a full pseudocode listing.

3.1 CRITIC UPDATE

At experiment t, we encode the available history h_{t-1} as a state vector s_t, which contains posterior information, the experiment stage, and other application-specific features described in Appendix D. The critic is a neural network with parameters ψ that receives s_t and outputs K quantile estimates for each candidate design $d \in \mathcal{D}$. We use the fixed-quantile representation of Dabney et al. (2018b), with K equally spaced quantile levels τ_k:

$$q_{\psi,k}(s_t, d) \approx F_\pi^{-1}(\tau_k \mid s_t, d), \qquad \tau_k = \frac{2k-1}{2K}, \qquad k = 1, \dots, K. \tag{11}$$

Here $F_\pi(\cdot \mid s_t, d)$ is the conditional distribution of the terminal IG reward received after selecting d in state s_t and following π for the remaining experiments, and F_π^{-1} is its quantile function. Each prediction $q_{\psi,k}(s_t, d)$ estimates the $100\tau_k$th percentile of this distribution; for example $\tau_k = 0.1$ corresponds to the 10th percentile. Assigning equal probability to the predicted quantiles gives:

$$\widehat{F}_\psi(\cdot \mid s_t, d) = \frac{1}{K} \sum_{k=1}^{K} \delta_{q_{\psi,k}(s_t,d)}, \tag{12}$$

where δ_q denotes a point mass at q. In our BOED setting, rewards are zero before the final experiment. For a completed campaign with history h_T, the terminal reward is $g = G(h_T)$, a realization of the random campaign return $G(H_T)$. With no discounting, the distributional Bellman relation in (9) gives the target values for the critic's quantile predictions at (s_t, d_t):

$$z_j = \begin{cases} q_{\bar{\psi},j}(s_{t+1}, d_{t+1}), & t < T, \\ g, & t = T, \end{cases} \qquad j = 1, \dots, K, \tag{13}$$

where d_{t+1} is sampled from the policy being evaluated. The parameters $\bar{\psi}$ belong to a soft update of the critic, used to stabilize training targets (Lillicrap et al., 2016). At intermediate stages, the targets represent the predicted terminal return distribution at the next decision. At the final stage, all targets equal the observed campaign return. We fit the quantiles using the pinball loss: $\ell_\tau(z, q) = \big(\tau - \mathbb{1}[z < q]\big)(z - q)$ (Koenker & Bassett, 1978; Dabney et al., 2018b). A minimizer of $\mathbb{E}[\ell_\tau]$ is a τ-quantile of the target distribution. Averaging over predicted and target quantiles gives:

$$\mathcal{L}_{\text{critic}}(\psi) = \frac{1}{K^2} \sum_{k=1}^{K} \sum_{j=1}^{K} \ell_{\tau_k}\big(z_j, q_{\psi,k}(s_t, d_t)\big). \tag{14}$$

We minimize this loss over sampled transitions. All campaign objectives use the same critic representation and training loss. Appendix C gives the target-update rule and optimization settings.

3.2 FROM CAMPAIGN OBJECTIVES TO DESIGN SCORES

The utility $\mathcal{U}(F_\pi)$ describes preferences over the distribution of complete campaign IG returns, through a function such as the mean, lower CVaR, entropic risk, or a variance-penalized mean. However, the critic predicts distributions conditional on the current history and design. Applying a nonlinear risk measure separately to these conditional distributions does not necessarily yield a policy gradient for $\mathcal{U}(F_\pi)$. For example, selecting actions by the CVaR of the conditional distribution does not necessarily optimize the CVaR of the overall campaign return (Lim & Malik, 2022). To connect these conditional return distributions to the overall campaign objective, we follow Xiao et al. (2024) and define an objective-specific score $f_{\mathcal{U}}(g)$ for each terminal return, with CVaR and mean-variance scores following the construction of (Tamar et al., 2015; Castro et al., 2012). Let $\pi_{\varphi_{\text{old}}}$ be the policy used to generate the current training batch. The score is chosen so that,

$$\nabla_\varphi \mathcal{U}(F_{\pi_\varphi})\big|_{\varphi=\varphi_{\text{old}}} = \nabla_\varphi \mathbb{E}_{\pi_\varphi}[f_{\mathcal{U}}(G(H_T))]\big|_{\varphi=\varphi_{\text{old}}}. \tag{15}$$

The expected value of $f_{\mathcal{U}}$ need not equal the objective itself; we are only concerned that the gradients are equal in expectation. For the mean objective $\mathcal{U}(F_\pi) = \mathbb{E}_\pi[G]$, we use $f_{\mathcal{U}} = g$. For CVaR, mean-variance, and entropic risk, we use score functions that depend on statistics of the return distribution that are estimated using earlier sampled returns. Appendix B.2 derives the score functions.

The actor must choose a design before the campaign return is known. We therefore estimate the expected score for each candidate design over the potential future decisions and actions. The expected score is computed by averaging the score function over the critic's predicted quantiles:

$$\widehat{f}_{\mathcal{U},\psi}(s_t, d) = \frac{1}{K}\sum_{k=1}^{K} f_{\mathcal{U}}\big(q_{\psi,k}(s_t, d)\big) \approx \mathbb{E}_{\pi_{\varphi_{\text{old}}}}[f(G(H_T)) \mid s_t, d]\,. \tag{16}$$

A larger $\widehat{f}_{\mathcal{U},\psi}$ predicts a higher expected campaign score after selecting d and following π_{old}.

3.3 Actor update

The actor is a neural network with parameters φ that outputs probabilities $\pi_\varphi(\cdot \mid s_t)$ over a discrete design space $\mathcal{D}$. The actor uses the predicted expected scores to compare candidate designs. We use the observed score from each completed campaign to correct the policy-gradient estimate. This gives an action-dependent control-variate estimator (Tucker et al., 2018).

Before collecting the current campaigns, we save the critic parameters as ψ_{old} and compute the reference statistics that define $f_{\mathcal{U}}$. The saved critic supplies the expected scores $\widehat{f}_{\mathcal{U},\psi_{\text{old}}}(s_t, d)$ in (16). Using these quantities, the actor maximizes the objective

$$\mathcal{S}(\varphi) = \mathbb{E}_{\pi_{\varphi_{\text{old}}}}\left[\frac{1}{T}\sum_{t=1}^{T}\left(\sum_{d\in\mathcal{D}}\pi_\varphi(d \mid s_t)\widehat{f}_{\mathcal{U},\psi_{\text{old}}}(s_t, d) + \frac{\pi_\varphi(d_t \mid s_t)}{\pi_{\varphi_{\text{old}}}(d_t \mid s_t)}\Big(f_{\mathcal{U}}(g) - \widehat{f}_{\mathcal{U},\psi_{\text{old}}}(s_t, d_t)\Big)\right)\right]. \tag{17}$$

The expectation is over complete campaigns generated by $\pi_{\phi_{\text{old}}}$ and is estimated by a batch average. The same terminal score $f_{\mathcal{U}}(g)$ weights the policy gradient of every design selected during that campaign. We minimize the actor loss $\mathcal{L}_{\text{actor}}(\varphi) = -\mathcal{S}(\varphi)$. At $\varphi = \varphi_{\text{old}}$, the critic contributions cancel in the expected gradient, leaving the likelihood-ratio policy gradient (Williams, 1992):

$$\nabla_\varphi \mathcal{S}(\varphi)|_{\varphi=\varphi_{\text{old}}} = \frac{1}{T}\mathbb{E}_{\text{old}}\left[f(g)\sum_{t=1}^{T}\nabla_\varphi \log \pi_\varphi(d_t \mid s_t)|_{\varphi=\varphi_{\text{old}}}\right]. \tag{18}$$

Appendix B gives more details of the actor and Appendix C gives details of the training.

4 Related work

Sequential BOED. Exploration in sequential BOED largely began with the seminal work of Lindley (1956), however adoption has been slow due to the computational cost of inference. Variational (Foster et al., 2019; Dahlke et al., 2023; Dahlke & Pacheco, 2025) and Monte Carlo approximations (Callahan et al., 2026; Ryan et al., 2016) make BOED tractable for moderately-sized problems, but modern BOED methods (Rainforth et al., 2024) often learn amortized policies that eschew inference at execution time (Foster et al., 2021; Ivanova et al., 2021). Other sequential approaches use approximate dynamic programming (Huan & Marzouk, 2016) or reinforcement learning (Blau et al., 2022; Shen & Huan, 2023; Shen et al., 2025) to account for information gained from future experiments. DistBOED instead trains a critic for the conditional distribution of terminal IG and uses campaign-level risk transformations to update the design policy.

Robust BOED. A practical limitation of BOED is that it assumes the data generating process is known to the experimenter–an assumption that rarely holds in practice. Indeed, Callahan & Catanach (2026) study how errors due to model misspecification can yield misleading design policies. Work addressing these limitations includes evaluating design utilities under an alternative model (Overstall & McGree, 2022), explicitly modeling discrepancy (Feng, 2015; Tsirpitzi et al., 2023), and optimizing worst-case design criteria (Go & Isaac, 2022; Attia et al., 2025). Theoretical foundations for generalized information measures (Duersch & Catanach, 2020) also support design criteria that are robust to model misspecification (Catanach & Das, 2023). Other approaches jointly target parameter inference and misspecification detection (Forster et al., 2025). Complementary work incorporates risk preferences into goal-oriented BOED through predictive uncertainty criteria (Jakeman et al., 2026). Our sequential spring experiment shows that DistBOED reduces the frequency and severity of misleading inference under model misspecification while preserving much of the average information gain.

Policy	Mean	Variance	$\text{CVaR}_{0.15}$	$\text{MV}_{0.5}$	ERM_3	Destroyed (%)
DAD	3.253	1.031	1.711	2.738	2.024	53.6
DistBOED mean	3.230	1.024	1.738	2.717	2.163	52.2
DistBOED mean–variance	3.140	0.620	1.989	2.830	2.444	31.2
DistBOED CVaR	2.959	0.619	2.102	2.649	2.465	11.5
DistBOED entropic	2.973	0.613	2.106	2.667	2.498	13.8

Table 1: Destructive-probe campaign returns. The risk parameters are $\lambda = 0.5$, $\alpha = 0.15$, and $\beta = 3$.

Risk-sensitive sequential decision making. While the majority of BOED methods consider a risk-neutral EIG objective, Shen & Huan (2026) study a variance-penalized objective for nonlinear BOED. Iqbal et al. (2024) learn sequential design policies using policy gradients estimated through particle-based inference. Their risk-seeking objective encourages exploration during policy learning, but crucially they do not target protection against low-information campaigns, which is the central aim of DistBOED. DistBOED considers a broader set of risk measures including CVaR and entropic risk. In solving risk-sensitive Markov decision processes, several methods have been proposed for optimizing CVaR (Chow & Ghavamzadeh, 2014), mean-variance (Mannor & Tsitsiklis, 2011), and entropic risk (Granados et al., 2025; Granados & Pacheco, 2026). The closely related setting of distributional reinforcement learning (Bellemare et al., 2023) learns the full return distribution using a finite categorical representation (Bellemare et al., 2017), explicit quantile regression (Dabney et al., 2018b), or via implicit quantile networks (Dabney et al., 2018a). DistBOED adapts this distributional methodology to BOED opting for an explicit quantile representation.

5 Experimental Results

We now evaluate DistBOED under three different scenarios using three different risk-sensitive objectives, in addition to a mean-seeking objective. Additionally, we compare DistBOED to the standard Deep Adaptive Design (DAD) of Foster et al. (2021) and show that under a mean-seeking policy, DistBOED recovers similar performance.

5.1 Destructive probe

We first consider a scenario in which an experimenter probes a sample to infer a parameter $\theta \in [0, 1]^2$. Each of up to $T = 5$ measurements uses an intensity $d_t \in \{0, 0.1, \ldots, 1\}$. Higher intensity yields a more accurate observation but also increases the probability of destroying the specimen before taking a reading. Intensity 0 provides a safe, lower-precision measurement while intensity 1 provides a risky high-precision measurement.

The prior $p(\theta)$ is a discrete approximation to a Gaussian restricted to the unit square. The campaign return is the terminal posterior-to-prior KL divergence. The design problem is therefore to balance measurement precision against the chance of losing the specimen before collecting enough information. Appendix D.1.1 gives details on the prior, observation and destruction models, and state representation.

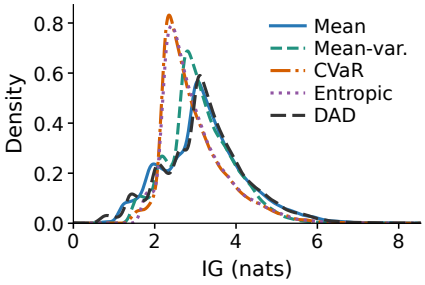

Figure 1: Estimated return densities for DAD and the three DistBOED objectives in the destructive probe experiment. We see that the left tail of the risk-sensitive distributions is not as negative as the left tail of the mean-seeking distributions. The risk-sensitive policies lead to fewer uninformative outcomes than the mean-seeking policies.

Results. Table 1 contains a comparison of the different policies. Mean-trained DistBOED and DAD achieve similar EIG. Mean-variance, CVaR, and entropic risk-trained policies reduce variability and improve the worst campaign outcomes at a cost to EIG. Mean-variance has better EIG, while CVaR and entropic risk provide stronger lower tail protection and substantially reduce specimen destruction. Fig. 1 shows the PDFs of the resulting return distributions.

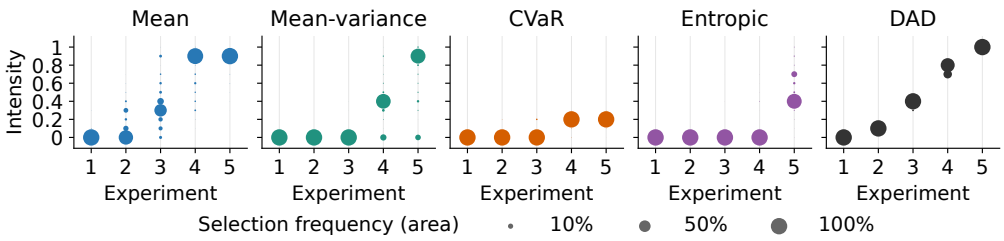

Figure 2: DistBOED intensity-selection frequencies at each stage, of the destructive probe experiment, conditional on the specimen being intact before the measurement. Intensity 0 is safe; intensity 1 is aggressive. The action that causes destruction is included, while actions after destruction are excluded. The risk-sensitive policies choose safer probe intensities, while the mean-optimized policies choose riskier intensities earlier in the campaign.

Policy	Mean	Variance	$\text{CVaR}_{0.1}$	$\text{MV}_{0.5}$	ERM_1
DAD	2.006	1.398	0.468	1.307	1.462
DistBOED mean	2.087	1.507	0.441	1.333	1.487
DistBOED mean–variance	1.797	0.664	0.496	1.465	1.482
DistBOED CVaR	1.335	0.239	0.661	1.216	1.232
DistBOED entropic	1.986	1.283	0.469	1.344	1.473

Table 2: Jammer campaign returns. The risk parameters are $\lambda = 0.5$, $\alpha = 0.1$, and $\beta = 1$.

Fig. 2 shows the policies learned by each objective. Mean-trained DistBOED and DAD begin with low intensities and then increase measurement precision at the risk of destruction. Mean–variance training collects three nearly safe readings before increasing intensity. CVaR training remains conservative throughout, while entropic training postpones its stronger measurement until the final stage. These policies secure information before risking specimen loss, with CVaR and entropic training placing greater emphasis on avoiding premature destruction.

5.2 Source-location with a signal jammer

We next consider the problem of sequentially placing a sensor to localize a signal source with unknown position $\theta \in [0, 1]^2$, similar to the setting described in Foster et al. (2021). A campaign consists of $T = 6$ measurements at locations chosen from 25 receiver sites, starting from a uniform prior over the source position. We modify the traditional source-location experiment with a signal jammer that sits at the known location $j = (0.5, 0.5)$ and activates with probability 0.6 at each measurement time. When active, it adds noise to the measurement whose variance increases with proximity to the jammer (Fig. 3). Appendix D.1.2 gives the observation model, prior distribution, receiver sites, and state representation.

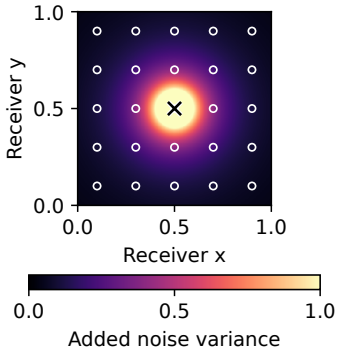

Figure 3: Added observation-noise variance during an active jammer burst. The cross marks the known jammer location; open circles mark the 25 available receiver sites. The jammer produces interference with probability 0.6 at each measurement.

Results. Table 2 compares the learned policies for the source-localization experiment. DAD and the DistBOED mean policies achieve higher EIG but have more variable campaign outcomes. Mean-variance training reduces this variability while preserving more EIG than the CVaR-trained policy. CVaR provides stronger protection against low-IG campaigns at a larger cost to EIG. The entropic risk objective produces a smaller change and does not improve the entropic score over mean training. Fig. 4 shows the resulting return distributions.

The measurement strategies shown in Fig. 5 illustrate this tradeoff. DAD and mean-trained DistBOED concentrate their first measurement at a single site, then spread subsequent measurements primarily along the boundary. The risk-sensitive policies fa-

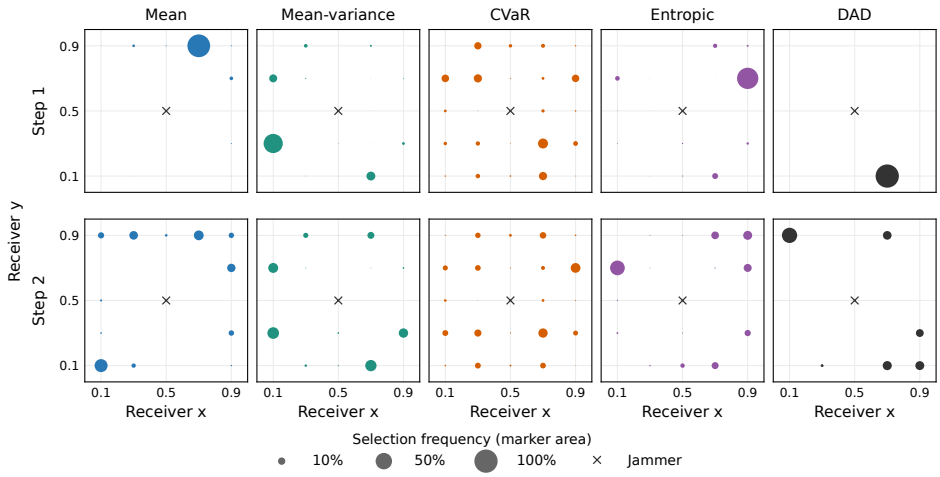

Figure 5: Sensor locations chosen for the first two steps of the source-localization problem. Rows show the steps and columns show DistBOED policies and DAD. Dot area is proportional to the policy's selection frequency, and the cross marks the location of the jammer.

vor more central locations, accepting stronger interference to hedge against uncertainty in the source position. Central placement reduces the greatest possible distance to the source, limiting the risk of receiving a weak signal from across the domain.

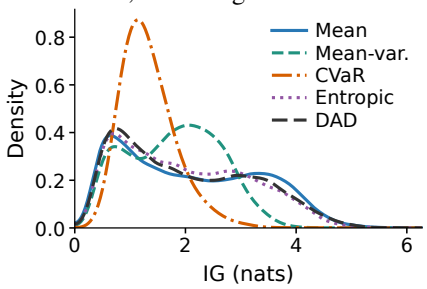

Figure 4: Estimated information-gain densities for DAD and DistBOED in the jammer experiment. Mean–variance and CVaR training reduce dispersion; CVaR training provides stronger lower-tail protection at a larger cost to average IG.

outcomes.

5.3 Misspecified Damped Spring-Mass

We study whether DistBOED can reduce misleading inference when the model used to compute posterior updates differs from the data generating process. To account for this model mismatch, we evaluate designs using generalized information gain (GIG), which measures the change in log density at the true parameter from the prior to the working posterior (Callahan & Catanach, 2026; Catanach & Das, 2023). Negative values indicate that observations reduce the posterior density at the truth. Training a policy to maximize the mean return distribution maximizes expected GIG (EGIG), while risk-sensitive training accounts for variability and low or negative gain

We use the forced spring-mass system of Callahan & Catanach (2026), depicted in Fig. 6, with true dynamics:

$$\ddot{x} + c_d|\dot{x}|\dot{x} + k_l x + k_c x^3 = \gamma \sin(\omega u), \qquad (19)$$

where x is displacement of the mass, u is time within an experiment, and $d_t = \gamma$ is the forcing amplitude. The experimenter seeks to infer damping and linear stiffness, $\theta = (c_\mathrm{d}, k_\mathrm{l})$, using a working model that omits the cubic term $k_\mathrm{c}x^3$. Stronger excitation increases the response but also increases the influence of the omitted term.

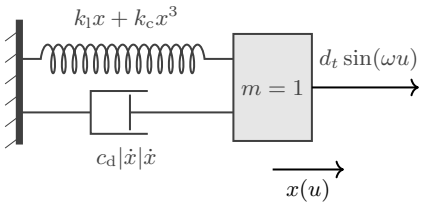

Figure 6: Forced spring-mass system with nonlinear damping and restoring force. The working model omits the cubic term.

A campaign consists of eight experiments. Each test starts with the spring from rest, applies one of six available forcing amplitudes, and records the mass displacement 16 times with independent Gaussian measurement noise. The damping and linear stiffness coefficients have independent log-uniform priors, and the cubic stiffness coefficient has an independent log-normal prior. Appendix D.1.3 gives the model parameters, numerical implementation, and return definition in full.

Policy	Mean	Variance	$\text{CVaR}_{0.1}$	$\text{MV}_{0.5}$	$G_{\text{gen}} < 0$
DAD (EGIG)	3.089	8.241	-1.378	-1.032	4.13%
DistBOED mean	3.079	9.919	-1.849	-1.881	4.64%
DistBOED mean–variance	3.025	1.841	0.239	2.105	2.37%
DistBOED CVaR	2.917	1.284	0.546	2.275	1.83%
DistBOED ERM	3.068	5.933	-1.184	0.102	4.03%

Table 3: Spring campaign returns. The final column is the negative-gain frequency. The risk parameters are $\lambda = 0.5$, $\alpha = 0.1$, and $\beta = 0.1$.

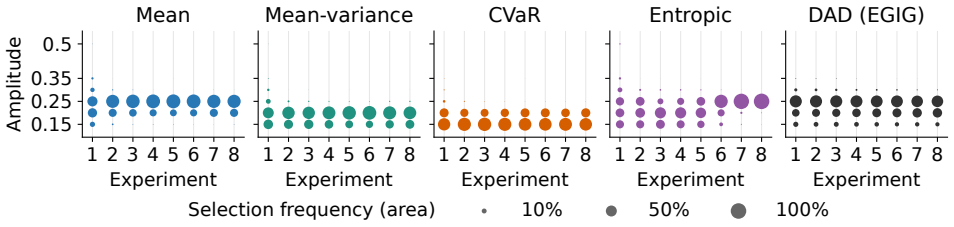

Figure 8: Learned policies for the misspecified spring-mass experiment. Dot area is proportional to selection frequency. Risk-sensitive policies favor lower amplitudes, reducing the influence of the omitted cubic term.

Results. Table 3 shows that DAD and Mean-trained DistBOED achieve the highest EGIG, but have the greatest variability and the most negative worst-decile means. Mean-variance and CVaR substantially reduce both the frequency and severity of misleading outcomes while preserving much of the average information. Mean-variance retains more average information, whereas CVaR provides stronger lower-tails. Both achieve positive worst-decile means, although some campaigns still produce negative gains. Entropic-risk training produces a smaller change. It reduces variance, but its worst-decile mean remains negative and its negative-gain frequency decreases only modestly.

Figs. 7 and 8 show the return distributions and learned designs. DAD and mean-trained DistBOED favor stronger excitation, while mean–variance and CVaR favor lower amplitudes. ERM uses lower amplitudes early, then concentrates on amplitude 0.25 late in the campaign. Weaker excitation sacrifices response strength but reduces the influence of the omitted cubic term, improving agreement with the working model.

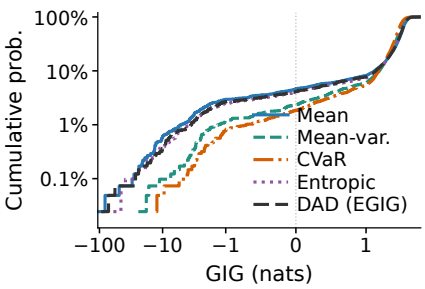

Figure 7: GIG distributions for the DistBOED policies in the misspecified spring-mass experiment shown as CDFs. Risk-sensitive policies show a lower probability of negative generalized IG outcomes and less variance than the mean policies.

6 Conclusion

We have introduced DistBOED, which learns the conditional distribution of final IG and uses it to train sequential design policies for different campaign objectives. Its quantile critic and corrected policy update support arbitrary utilities of this distribution in a common training procedure. Across three applications, mean-trained DistBOED retains strong mean-information performance relative to DAD, while risk-sensitive training changes both the chosen experiments and the distribution of campaign outcomes. Delaying destructive measurements reduces dispersion; weaker spring excitation reduces misleading inference; and different receiver policies trade average information against protection from low-gain campaigns. DistBOED makes these tradeoffs part of the optimization, so the experimental objective can reflect experimenter preference.

Limitations. Our implementation computes posterior features, adding inference cost, although a policy using raw histories could defer this computation until the terminal return is evaluated. Terminal-only rewards can make credit assignment difficult over long campaigns. Implementation complexity is increased due to several interacting components, including an actor, distributional critic, and estimates of objective-specific campaign statistics. Finally, the formulation enumerates a finite design menu; extending it to continuous design spaces requires further development.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experimental design, assist with implementing methods, and interpret results. We did not use generative AI tools to develop theoretical models or conceptual frameworks, or to propose or refine hypotheses. Formulating new mathematical claims and developing proofs, generating synthetic datasets, translation, dataset cleaning or reformatting, and qualitative or thematic analysis were not part of this work.

Additionally, we used generative AI tools to assist with manuscript preparation, including drafting, editing, and organization; finding and summarizing relevant literature and formatting references; and preparing figures.

We carefully reviewed all AI-assisted work. We checked AI-assisted code to verify that it implements the methods described in the paper and independently checked all reported results. We checked AI-suggested references against the cited papers and independently verified factual claims included in the manuscript. We assessed methodological suggestions, checked that interpretations are supported by the results, verified figures against the data and models, and reviewed manuscript revisions for technical accuracy. We take responsibility for the final content of this work, including all text, claims, code, and figures produced with generative AI assistance.

REFERENCES

Ahmed Attia, Sven Leyffer, and Todd S. Munson. Robust A-optimal experimental design for sensor placement in Bayesian linear inverse problems. *SIAM/ASA Journal on Uncertainty Quantification*, 13(2):744–774, 2025. doi: 10.1137/24M1667543.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 449–458. PMLR, 2017. URL https://proceedings.mlr.press/v70/bellemare17a.html.

Marc G Bellemare, Will Dabney, and Mark Rowland. *Distributional reinforcement learning*. MIT Press, 2023.

Tom Blau, Edwin V. Bonilla, Iadine Chades, and Amir Dezfouli. Optimizing sequential experimental design with deep reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2107–2128, 2022. URL https://proceedings.mlr.press/v162/blau22a.html.

Jake Callahan and Tommie Catanach. On the misinformation in a statistical experiment. In *Proceedings of the 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 2287–2295. PMLR, 2026. URL https://proceedings.mlr.press/v300/callahan26a.html.

Jake Callahan, Andrew Chin, Jason Pacheco, and Tommie Catanach. Reverse-annealed sequential monte carlo for efficient bayesian optimal experiment design. *Advances in Neural Information Processing Systems*, 38:64910–64932, 2026.

Dotan Di Castro, Aviv Tamar, and Shie Mannor. Policy gradients with variance related risk criteria. In John Langford and Joelle Pineau (eds.), *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pp. 935–942, New York, NY, USA, July 2012. Omnipress. ISBN 978-1-4503-1285-1.

Tommie A Catanach and Niladri Das. Metrics for bayesian optimal experiment design under model misspecification. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 7707–7714. IEEE, 2023.

Yinlam Chow and Mohammad Ghavamzadeh. Algorithms for CVaR optimization in MDPs. In *Advances in Neural Information Processing Systems*, volume 27, 2014. URL https://papers.nips.cc/paper_files/paper/2014/hash/35f1050a4381d2d216bf56ad46b0277d-Abstract.html.

Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pp. 1096–1105. PMLR, 2018a.

Will Dabney, Mark Rowland, Marc G. Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 2892–2901. AAAI Press, 2018b. doi: 10.1609/aaai.v32i1.11791. URL https://ojs.aaai.org/index.php/AAAI/article/view/11791.

Caleb Dahlke and Jason Pacheco. Flow-based variational mutual information: Fast and flexible approximations. In *International Conference on Learning Representations*, volume 2025, pp. 68771–68793, 2025.

Caleb Dahlke, Sue Zheng, and Jason Pacheco. Fast variational estimation of mutual information for implicit and explicit likelihood models. In *International Conference on Artificial Intelligence and Statistics*, pp. 10262–10278. PMLR, 2023.

Jed A Duersch and Thomas A Catanach. Generalizing information to the evolution of rational belief. *Entropy*, 22(1):108, 2020.

Chi Feng. Optimal Bayesian experimental design in the presence of model error. Master's thesis, Massachusetts Institute of Technology, 2015. URL https://hdl.handle.net/1721.1/97790.

Alex Forster, Desi R Ivanova, and Tom Rainforth. Improving robustness to model misspecification in bayesian experimental design. In *7th Symposium on Advances in Approximate Bayesian Inference-Workshop Track*, 2025.

Adam Foster, Martin Jankowiak, Elias Bingham, Paul Horsfall, Yee Whye Teh, Thomas Rainforth, and Noah Goodman. Variational bayesian optimal experimental design. *Advances in neural information processing systems*, 32, 2019.

Adam Foster, Desi R. Ivanova, Ilyas Malik, and Tom Rainforth. Deep adaptive design: Amortizing sequential Bayesian experimental design. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3384–3395. PMLR, 2021. URL https://proceedings.mlr.press/v139/foster21a.html.

Jinwoo Go and Tobin Isaac. Robust expected information gain for optimal Bayesian experimental design using ambiguity sets, 2022. URL https://arxiv.org/abs/2205.09914.

Alonso Granados and Jason Pacheco. Risk-sensitive exponential actor critic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 21343–21351, 2026.

Alonso Granados, Mohammadreza Ebrahimi, and Jason Pacheco. Risk-sensitive variational actor-critic: A model-based approach. In *International Conference on Learning Representations*, volume 2025, pp. 17112–17133, 2025.

Ernst Hairer, Syvert Paul Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics*. Springer, 2 edition, 1993. doi: 10.1007/978-3-540-78862-1.

Ronald A. Howard and James E. Matheson. Risk-sensitive Markov decision processes. *Management Science*, 18(7):356–369, 1972. doi: 10.1287/mnsc.18.7.356. URL https://pubsonline.informs.org/doi/10.1287/mnsc.18.7.356.

Xun Huan and Youssef M. Marzouk. Sequential bayesian optimal experimental design via approximate dynamic programming, 2016. URL https://arxiv.org/abs/1604.08320.

Sahel Iqbal, Adrien Corenflos, Simo Särkkä, and Hany Abdulsamad. Nesting particle filters for experimental design in dynamical systems. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 21047–21068. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/iqbal24a.html.

Desi R Ivanova, Adam Foster, Steven Kleinegesse, Michael U Gutmann, and Thomas Rainforth. Implicit deep adaptive design: Policy-based experimental design without likelihoods. *Advances in neural information processing systems*, 34:25785–25798, 2021.

John D. Jakeman, Rebekah White, Bart van Bloemen Waanders, Drew P. Kouri, and Alen Alexanderian. Risk-aware goal-oriented bayesian optimal experimental design, 2026. URL https://arxiv.org/abs/2609.08023.

Roger Koenker and Gilbert Bassett, Jr. Regression quantiles. *Econometrica*, 46(1):33–50, 1978. doi: 10.2307/1913643. URL https://www.jstor.org/stable/1913643.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1509.02971.

Shiau Hong Lim and Ilyas Malik. Distributional reinforcement learning for risk-sensitive policies. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30977–30989, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c88a2bd0e793550d0e885aa6e31ca277-Abstract-Conference.html.

Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Shie Mannor and John N Tsitsiklis. Mean-variance optimization in markov decision processes. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 177–184, 2011.

Antony M. Overstall and James M. McGree. Bayesian decision-theoretic design of experiments under an alternative model. *Bayesian Analysis*, 17(4):1021–1041, 2022. doi: 10.1214/21-BA1286.

Tom Rainforth, Adam Foster, Desi R Ivanova, and Freddie Bickford Smith. Modern bayesian experimental design. *Statistical Science*, 39(1):100–114, 2024.

R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *The Journal of Risk*, 2(3):21–41, 2000. doi: 10.21314/jor.2000.038. URL https://doi.org/10.21314/jor.2000.038.

R Tyrrell Rockafellar, Stanislav Uryasev, et al. Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42, 2000.

Elizabeth G Ryan, Christopher C Drovandi, James M McGree, and Anthony N Pettitt. A review of modern computational algorithms for bayesian optimal design. *International Statistical Review*, 84(1):128–154, 2016.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897. PMLR, 2015. URL https://proceedings.mlr.press/v37/schulman15.html.

Wanggang Shen and Xun Huan. Bayesian sequential optimal experimental design for nonlinear models using policy gradient reinforcement learning. *Computer Methods in Applied Mechanics and Engineering*, 416:116304, 2023. doi: 10.1016/j.cma.2023.116304.

Wanggang Shen and Xun Huan. Mean–variance risk-aware Bayesian optimal experimental design for nonlinear models. *arXiv preprint arXiv:2604.04315*, 2026. doi: 10.48550/arXiv.2604.04315. URL https://arxiv.org/abs/2604.04315.

Wanggang Shen, Jiayuan Dong, and Xun Huan. Variational sequential optimal experimental design using reinforcement learning. *Computer Methods in Applied Mechanics and Engineering*, 444:118068, 2025. doi: 10.1016/j.cma.2025.118068.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Aviv Tamar, Yonatan Glassner, and Shie Mannor. Optimizing the CVaR via sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, pp. 2993–2999, 2015. doi: 10.1609/aaai.v29i1.9561.

Renata Eirini Tsirpitzi, Frank Miller, and Carl-Fredrik Burman. Robust optimal designs using a model misspecification term. *Metrika*, 86(7):781–804, 2023. doi: 10.1007/s00184-023-00893-6.

George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E. Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5015–5024. PMLR, 2018. URL https://proceedings.mlr.press/v80/tucker18a.html.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

Minheng Xiao, Xian Yu, and Lei Ying. Policy gradient methods for risk-sensitive distributional reinforcement learning with provable convergence. *arXiv preprint arXiv:2405.14749*, 2024. doi: 10.48550/arXiv.2405.14749. URL https://arxiv.org/abs/2405.14749.

A Algorithm overview and training procedure

Fig. 9 summarizes the algorithm architecture. The actor selects designs using the observed state, and the simulator generates observations and computes the terminal campaign return. These campaign rollouts train the critic to predict conditional return distributions and supply the observed returns used in the actor update. Algorithm 1 gives the complete training algorithm.

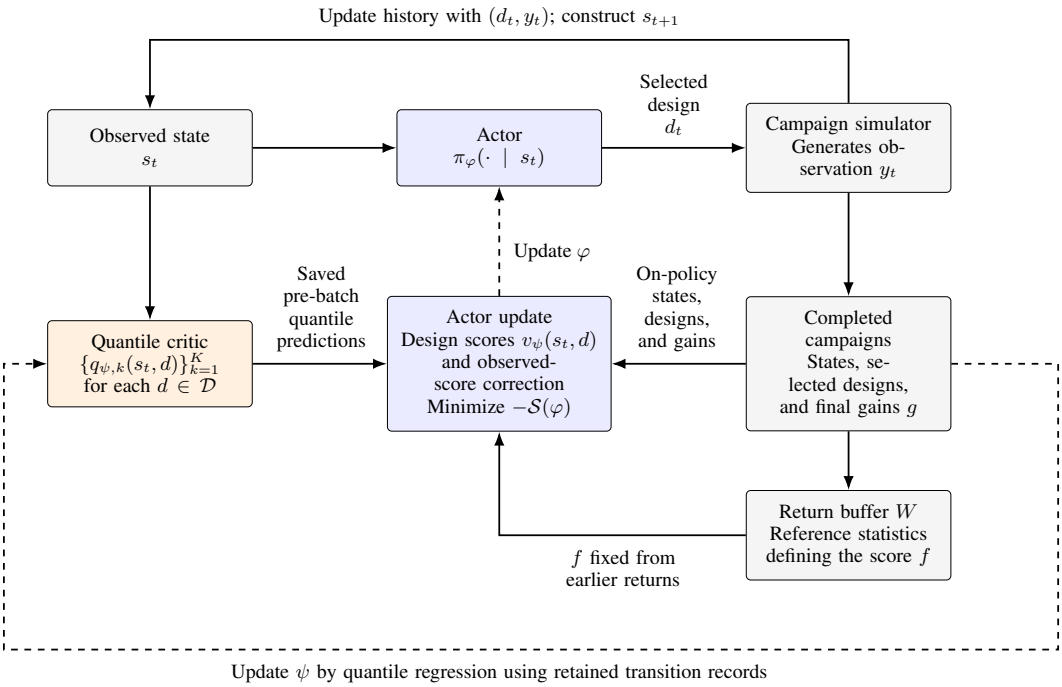

Figure 9: DistBOED architecture. The actor selects designs from the observed state. The critic predicts a final-gain distribution for each candidate design. The actor update combines scores from saved pre-batch critic predictions with a correction based on observed campaign returns. Earlier on-policy returns supply the reference statistics that define the score function f. Solid arrows indicate data flow; dashed arrows indicate parameter updates. Target-critic and optimizer details are omitted.

Campaign rollouts. Each iteration collects complete campaigns under the current policy network. We save the states, chosen designs, and transitions, and attach the terminal return to each transition after the campaign ends. These on-policy campaigns are used to train both the actor and the critic.

Inspired by the exploring-starts approach of Sutton & Barto (2018, Chapter 5), we collect additional exploratory campaigns to train the critic on designs that the actor would otherwise only rarely select. Each exploratory campaign follows the current policy except at one uniformly sampled stage, where it takes a uniformly sampled design. We retain transitions only from that stage onward, since earlier transitions would include a future forced action in their continuation. Exploratory campaigns are used to train the critic, but not the actor.

Fixed scores and critic predictions. Before collecting the on-policy campaigns used for one actor update, we estimate the reference statistics defining the campaign score (see Section B.2) from earlier on-policy returns stored in a buffer. We also save a copy of the critic. These reference statistics and saved predictions remain fixed during the actor update step. The saved critic supplies the design scores used in (16), while the new terminal IG returns from the newly collected on-policy campaigns supply the observed scores used in the correction term of (17).

Update order. Training begins with warmup iterations that fit the critic and populate the return buffer while holding the actor fixed. Each subsequent iteration holds the score function constant and saves the critic, collects campaigns, updates the critic, and then finally updates the actor using the saved critic. Finally, the new on-policy returns are appended to the buffer. Appendix B shows how the control-variate estimator of Tucker et al. (2018) applies to our campaign objectives, and Appendix C gives the network architectures, optimization settings, and hyperparameters.

The DistBOED algorithm. The algorithm is detailed in full as Algorithm 1. We write $\mathrm{SimulateCampaign}(\pi, J, a)$ for a campaign generated under policy π, with design a forced at stage J; setting $J = 0$ disables forcing. For a completed campaign C with terminal return g_C, let $R(C, J)$ be the set containing the transition records $(s_t, d_t, s_{t+1}, g_C, t)$ for $t = J, \ldots, T$. The batch $\mathcal{B}$ contains the on-policy campaigns collected in the current iteration, and $\mathcal{R}$ contains the records used to train the critic. The return buffer W stores terminal returns from earlier on policy campaigns. In the algorithm, $s \in \mathcal{B}$ denotes a state recorded in these campaigns. The operation $\mathrm{Reference}(W)$ constructs the score function f in (24) using reference statistics estimated from the return buffer W. For the mean objective, it returns $f(g) = g$ without estimating any statistics. The empirical surrogate $\widehat{\mathcal{S}}$ replaces the expectation in (17) with an average over $\mathcal{B}$.

The operation $\mathrm{AdamStep}$ takes a loss gradient and returns updated parameters and optimizer state. The operator Clip_c limits the gradient's Euclidean norm to c. The operation $\mathrm{Backtrack}$ tests the proposed actor parameters and successively halved displacements from φ_{old}, according to the method detailed in Appendix C. If no candidate passes, it returns φ_{old}. Finally, $\uplus$ appends records without removing duplicates, $\|$ concatenates return sequences, and Last_L retains the most recent L entries.

At deployment, only the actor and the state-construction procedure are needed. The actor selects a design, the resulting observation updates the history, and the next state is constructed. The critic and return buffer are used only during training.

B Campaign objectives and policy-gradient derivations

This section derives the campaign score functions and shows how the actor objective in (17) combines them with critic predictions to estimate the gradient of the chosen campaign objective. The estimator uses the action-dependent control-variate construction of Tucker et al. (2018).

B.1 Policy gradient for campaign returns

The actor seeks to improve an objective of the terminal campaign return distribution. Because every design can affect this return, the actor update must account for how the policy changes the probability of a complete campaign history $h_T = (d_1, y_1, \ldots, d_T, y_T)$. We first express this dependence mathematically and then use it to derive the score function for each objective.

Algorithm 1 DistBOED training

Require: $N_0, N, B_{\text{on}}, B_{\text{ex}}, M, b, K, L$; optimizer settings.
Initialize φ, ψ; $\bar{\psi} \leftarrow \psi$; $W \leftarrow \emptyset$; optimizer states $\omega_\varphi, \omega_\psi \leftarrow 0$.
for $n = 1, \dots, N_0 + N$ **do**
  $\varphi_{\text{old}} \leftarrow \varphi$; $\pi_{\text{old}} \leftarrow \pi_{\varphi_{\text{old}}}$.
  if $n > N_0$ **then**
    $\psi_{\text{old}} \leftarrow \psi$; $f \leftarrow \text{Reference}(W)$.
  end if
  $\mathcal{B} \leftarrow \{\text{SimulateCampaign}(\pi_{\text{old}}, 0, \varnothing)\}_{i=1}^{B_{\text{on}}}$.
  $\mathcal{R} \leftarrow \biguplus_{C \in \mathcal{B}} R(C, 1)$.
  for $i = 1, \dots, B_{\text{ex}}$ **do**
    $J \sim \text{Uniform}\{1, \dots, T\}$; $a \sim \text{Uniform}(\mathcal{D})$.
    $C \leftarrow \text{SimulateCampaign}(\pi_{\text{old}}, J, a)$.
    $\mathcal{R} \leftarrow \mathcal{R} \uplus R(C, J)$.
  end for
  if $n > N_0$ **then**
    $v(s, d) \leftarrow K^{-1} \sum_{k=1}^{K} f(q_{\psi_{\text{old}},k}(s, d))$, for $s \in \mathcal{B}$, $d \in \mathcal{D}$.
    Form $\widehat{\mathcal{S}}(\varphi)$ using $\mathcal{B}, f, v$ [Eq. (17)].
  end if
  for $m = 1, \dots, M$ **do**
    Sample $\{(s_i, d_i, s'_i, g_i, t_i)\}_{i=1}^{b}$ from $\mathcal{R}$.
    for $i = 1, \dots, b$ **do**
      if $t_i < T$ **then**
        $d'_i \sim \pi_{\text{old}}(\cdot \mid s'_i)$; $z_{ij} \leftarrow q_{\bar{\psi},j}(s'_i, d'_i)$, $j = 1, \dots, K$.
      else
        $z_{ij} \leftarrow g_i$, $j = 1, \dots, K$.
      end if
    end for
    $\widehat{\mathcal{L}} \leftarrow (bK^2)^{-1} \sum_{i,k,j} \ell_{\tau_k}(z_{ij}, q_{\psi,k}(s_i, d_i))$.
    $u_\psi \leftarrow \text{Clip}_{c_\psi}(\nabla_\psi \widehat{\mathcal{L}})$, holding z_{ij} fixed.
    $(\psi, \omega_\psi) \leftarrow \text{AdamStep}(\psi, \omega_\psi, u_\psi, \text{lr}_\psi)$.
    $\bar{\psi} \leftarrow (1 - \kappa)\bar{\psi} + \kappa\psi$.
  end for
  if $n > N_0$ **then**
    $u_\varphi \leftarrow \text{Clip}_{c_\varphi}(-\nabla_\varphi \widehat{\mathcal{S}}(\varphi)|_{\varphi=\varphi_{\text{old}}})$.
    $(\widetilde{\varphi}, \omega_\varphi) \leftarrow \text{AdamStep}(\varphi_{\text{old}}, \omega_\varphi, u_\varphi, \text{lr}_\varphi)$.
    $\varphi \leftarrow \text{Backtrack}(\varphi_{\text{old}}, \widetilde{\varphi}, \widehat{\mathcal{S}}, \mathcal{B})$.
  end if
  $W \leftarrow \text{Last}_L(W \| [g_C]_{C \in \mathcal{B}})$.
end for
return π_φ.

Let φ_{old} denote the parameters used to collect the current campaigns, and write $\mathbb{E}_\varphi$ for expectation under π_φ. The probability of a campaign history is

$$p_{\pi_\varphi}(h_T) = \prod_{t=1}^{T} \pi_\varphi(d_t \mid s_t) p(y_t \mid h_{t-1}, d_t). \tag{20}$$

Here, s_t is constructed from h_{t-1} and other application-specific state parameters, and $p(y_t \mid h_{t-1}, d_t)$ is the simulator's conditional predictive density, with θ and any other latent variables averaged out. We assume that the simulator, state encoding, and return calculation have no explicit dependence on φ once the campaign history is fixed. We also assume that $\pi_\varphi(d \mid s)$ is differentiable with respect to φ, that the set of designs assigned positive probability does not change with φ, and we assume sufficient integrability conditions such that differentiation can be interchanged with expectation. Under these assumptions, only the design probabilities contribute to the derivative of the

log probability of a campaign history:

$$\nabla_\varphi \log p_\varphi(h_t) = \sum_{t=1}^{T} \nabla_\varphi \log \pi_\varphi(d_t \mid s_t). \tag{21}$$

For a fixed function f of the terminal return, the likelihood-ratio identity gives (Williams, 1992)

$$\nabla_\varphi \mathbb{E}_\varphi[f(G)] = \mathbb{E}_\varphi[f(G)\nabla_\varphi \log p_\varphi(H_T)]. \tag{22}$$

We choose f so that this expression equals the gradient of the desired campaign objective at φ_{old}.

B.2 Objective-specific score functions

Each objective $\mathcal{U}$ is a function of the whole distribution, but the likelihood-ratio identity in (22) applies to the expectation of a function of the scalar returns. We therefore construct a score function $f(g)$ whose expected gradient matches the objective's gradient at φ_{old}. Define the population reference statistics by

$$\eta_0 \in \arg\max_{\eta\in\mathbb{R}} \left\{ \eta - \frac{1}{\alpha}\mathbb{E}_{\phi_{\text{old}}}[(\eta - G)_+] \right\}, \quad m_0 = \mathbb{E}_{\phi_{\text{old}}}[G], \quad c_0 = \mathbb{E}_{\phi_{\text{old}}}[e^{-\beta G}], \tag{23}$$

where $(x)_+ = \max\{x, 0\}$. The scores are

$$f(g) = \begin{cases} g, & \text{if } \mathcal{U} = \mathbb{E}[G], \\ -(\eta_0 - g)_+/\alpha, & \text{if } \mathcal{U} = \text{CVaR}_\alpha(G), \\ (1 + 2\lambda m_0)g - \lambda g^2, & \text{if } \mathcal{U} = \text{MV}_\lambda(G), \\ -e^{-\beta g}/(\beta c_0), & \text{if } \mathcal{U} = \text{ERM}_\beta(G). \end{cases} \tag{24}$$

We will now show that the expected gradients of these score functions match the policy gradients.

Mean. For $\mathcal{U}(F_{\pi_\phi}) = \mathbb{E}_\phi[G]$, choosing $f(g) = g$ gives the usual expected-return policy gradient directly.

Mean-variance. We obtain the mean-variance score from the policy-gradient identities of Castro et al. (2012, Lemma 4.2). The following derivation adapts these identities to our campaign objective. Write $m_\phi = \mathbb{E}_\varphi[G]$. Expanding the variance and differentiating gives

$$\begin{aligned} \nabla_\phi \text{MV}_\lambda(G)|_{\phi=\phi_{\text{old}}} &= \nabla_\phi \left(m_\phi - \lambda \mathbb{E}_\phi[G^2] + \lambda m_\phi^2 \right)\big|_{\phi=\phi_{\text{old}}} \\ &= \left[(1 + 2\lambda m_0)\nabla_\phi \mathbb{E}_\phi[G] - \lambda \nabla_\phi \mathbb{E}_\phi[G^2] \right]_{\phi=\phi_{\text{old}}} \\ &= \nabla_\phi \mathbb{E}_\phi[f(G)]|_{\phi=\phi_{\text{old}}}, \end{aligned} \tag{25}$$

where $f(g) = (1 + 2\lambda m_0)g - \lambda g^2$.

CVaR. We use the CVaR score corresponding to the policy-gradient formula of Tamar et al. (2015). We derive it here using the threshold representation of Rockafellar & Uryasev (2000), following its use in risk-sensitive policy gradients (Chow & Ghavamzadeh, 2014). The CVaR is

$$\text{CVaR}_\alpha(G) = \max_{\eta\in\mathbb{R}} \left\{ \eta - \frac{1}{\alpha}\mathbb{E}_\phi[(\eta - G)_+] \right\}. \tag{26}$$

Let $\eta_0 = F^{-1}_{\pi_{\text{old}}}(\alpha)$ be the α-quantile of the return distribution under π_{old}. The value η_0 maximizes the expression in (26). At policies where the CVaR objective is differentiable and differentiation can be interchanged with expectation, its gradient is

$$\begin{aligned} \nabla_\phi \text{CVaR}_\alpha(G)|_{\phi=\phi_{\text{old}}} &= \nabla_\phi \left(\eta_0 - \frac{1}{\alpha}\mathbb{E}_\phi[(\eta_0 - G)_+] \right)\Big|_{\phi=\phi_{\text{old}}} \\ &= -\frac{1}{\alpha} \nabla_\phi \mathbb{E}_\phi[(\eta_0 - G)_+]|_{\phi=\phi_{\text{old}}} \\ &= \nabla_\phi \mathbb{E}_\phi[f(G)]|_{\phi=\phi_{\text{old}}}, \end{aligned} \tag{27}$$

where $f(g) = -(\eta_0 - g)_+/\alpha$.

Entropic risk. We use the exponential return transformation from **?**. We express their gradient formula in our notation and show how it yields the expected-score gradient used by DistBOED. Write $c_\phi = \mathbb{E}_\phi[e^{-\beta G}]$. Assume this exponential moment is finite in a neighborhood of ϕ_{old}. Differentiating the entropic objective gives

$$\begin{aligned}\nabla_\phi \text{ERM}_\beta(G)|_{\phi=\phi_{\text{old}}} &= \nabla_\phi \left(-\frac{1}{\beta}\log c_\phi\right)\Big|_{\phi=\phi_{\text{old}}} \\ &= -\frac{1}{\beta c_0}\,\nabla_\phi \mathbb{E}_\phi[e^{-\beta G}]\big|_{\phi=\phi_{\text{old}}} \\ &= \nabla_\phi \mathbb{E}_\phi[f(G)]|_{\phi=\phi_{\text{old}}}\,,\end{aligned} \tag{28}$$

where $f(g) = -e^{-\beta g}/(\beta c_0)$.

B.3 Gradient of the actor objective

The preceding derivations connect each campaign objective to an expected score. we now show that the actor objective in (17) estimates this expected-score gradient using the construction of Tucker et al. (2018). We write $v(s,d) = v_{\pi_{\text{old}}}(s,d)$ for the design scores computed from the critic in (16). Throughout this derivation, expectations are conditional on the training data available before collecting the current campaigns. Thus, f and v do not depend on the outcomes of those campaigns. Differentiating the actor objective at ϕ_{old}, the parameters used to generate the current campaigns, gives

$$\begin{aligned}\nabla_\phi \mathcal{S}(\phi)|_{\phi=\phi_{\text{old}}} = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\phi_{\text{old}}}\Bigg[&\sum_{d\in\mathcal{D}} v(S_t,d)\,\nabla_\phi \pi_\phi(d \mid S_t)|_{\phi=\phi_{\text{old}}} \\ &+ \big(f(G) - v(S_t,D_t)\big)\,\nabla_\phi \log \pi_\phi(D_t \mid S_t)|_{\phi=\phi_{\text{old}}}\Bigg].\end{aligned} \tag{29}$$

The two contributions from the critic therefore cancel in expectation, leaving

$$\begin{aligned}\nabla_\phi \mathcal{S}(\phi)|_{\phi=\phi_{\text{old}}} &= \frac{1}{T}\mathbb{E}_{\phi_{\text{old}}}\left[f(G)\sum_{t=1}^{T}\nabla_\phi \log \pi_\phi(D_t \mid S_t)|_{\phi=\phi_{\text{old}}}\right] \\ &= \frac{1}{T}\,\nabla_\phi \mathbb{E}_\phi[f(G)]|_{\phi=\phi_{\text{old}}} \\ &= \frac{1}{T}\,\nabla_\phi \mathcal{U}(F_{\pi_\phi})\big|_{\phi=\phi_{\text{old}}}\,.\end{aligned} \tag{30}$$

B.4 Estimated reference statistics and finite updates

The implementation estimates the reference statistics from a buffer of earlier on-policy returns. These estimates have sampling error and may reflect earlier policies. Consequently, the critic cancellation remains valid for the resulting score function, but its expected-score gradient only approximates the gradient of the intended campaign objective. The gradient identity holds at ϕ_{old}. Away from these parameters, the probability ratio in (17) adjusts the probability of the selected design at a recorded history, but does not account for changes in the distribution of histories. The actor objective is therefore a local surrogate. The optimizer and backtracking procedure described in Appendix C restrict the update size, but do not guarantee improvement of the population campaign objectives.

C Implementation and hyperparameters

We train a separate actor and critic for each objective. All three environments use the architecture and training settings in Table 4. Table 5 lists the settings that vary by environment.

C.1 Network architectures

The actor and critic are separate fully connected networks. Each has two hidden layers of nodes and ReLU activations. They receive the same state vector within each environment. The actor

Setting	Value
Hidden layers per network	2
Units per hidden layer	128
Quantiles per design, K	101
Warmup iterations, N_0	200
Actor-update iterations, N	10,000
On-policy campaigns per iteration, B_{on}	64
Exploratory campaigns per iteration, B_{ex}	64
Critic updates per iteration, M	8
Critic minibatch size, b	256
Return-buffer capacity, L	2,048
Actor learning rate	10^{-4}
Critic learning rate	5×10^{-4}
Actor gradient-norm limit, c_ϕ	1
Critic gradient-norm limit, c_ψ	5
Target update coefficient, κ	0.01
Mean policy-KL limit, δ_{avg}	0.002
Maximum policy-KL limit, $\delta_{\max}$	0.02
Maximum backtracking halvings, R	16
Surrogate acceptance tolerance, ε	10^{-7}

Table 4: Shared DistBOED architecture and training settings. Actor-update iterations count attempted updates, including proposals rejected by backtracking.

Setting	Probe	Jammer	Spring
Campaign length, T	5	6	8
Number of candidate designs	11	25	6
State dimension	11	4,256	4,371
CVaR level, α	0.15	0.1	0.1
Variance penalty, λ	0.5	0.5	0.5
Entropic coefficient, β	3	1	—

Table 5: Environment-specific settings for the main DistBOED comparisons. State encodings are specified in Appendix D.

outputs one logit per candidate design, which defines a categorical policy through a softmax. The critic outputs $K = 101$ quantiles per design at the quantile levels specified in (12). Its outputs are unconstrained and are not sorted during training. To help stabilize critic training, we parameterize the predicted terminal-return quantiles as the current posterior KL plus a learned residual. Let $b(s_t)$ denote the current posterior-to-prior KL. The critic predicts

$$q_{\psi,k}(s_t, d) = b(s_t) + r_{\psi,k}(s_t, d), \tag{31}$$

where $r_{\psi,k}$ is the corresponding network output. The quantile-regression loss is applied to $q_{\psi,k}$, so the critic still estimates the full terminal-return distribution. In the spring experiment, the offset is the working-posterior KL, while the critic's prediction target remains the terminal generalized information gain.

Both networks' weights and biases are initialized uniformly on $[-1/\sqrt{n_{\text{in}}}, 1/\sqrt{n_{\text{in}}}]$, where n_{in} is input width. The target critic is initialized as a copy of the critic.

C.2 Training data and reference statistics

Each policy is trained for 200 warmup iterations followed by 10,000 iterations that update both networks. Each iteration performs eight critic updates and, after warmup, one actor update. During warmup, the actor parameters remain fixed while the critic is trained and the return buffer is populated.

Critic minibatches are sampled uniformly with replacement from the set of transitions from the current actor. For nonterminal transitions, Bellman targets use the next design recorded in the campaign. Terminal targets use the observed campaign return. After each critic update, the target parameters

are averaged according to

$$\bar{\psi} \leftarrow (1-\kappa)\bar{\psi} + \kappa\psi, \qquad \kappa = 0.01. \tag{32}$$

The objective discount is one.

The return buffer contains the most recent 2,048 on-policy terminal returns. Before each actor update, reference statistics are computed from earlier campaigns: the sample mean for mean-variance, the sample mean of $e^{-\beta g}$ for entropic risk, and the empirical α-quantile for CVaR. The quantile estimate uses linear interpolation between adjacent ordered returns. Each actor update uses all recorded decisions from the current batch of $B_{\mathrm{on}} = 64$ on-policy campaigns to estimate the objective in (17).

C.3 Optimization

The actor and critic use separate Adam optimizers with moment coefficients $(0.9, 0.999)$, numerical constant 10^{-8}, and zero weight decay. learning rates remain constant throughout training. Before each optimizer step, gradient norms are clipped:

$$\mathrm{Clip}_c(u) = \begin{cases} u \min\{1, c/\|u\|_2\}, & \|u\|_2 > 0, \\ 0, & \|u\|_2 = 0. \end{cases} \tag{33}$$

The critic minimizes the pinball loss in (14), with equal weights across predicted and target quantiles. The actor minimizes the negative empirical surrogate in (17).

C.4 Actor backtracking

We adapt the line-search procedure in Schulman et al. (2015), which reduces a proposed policy update until it satisfies a KL-divergence constraint and improves the surrogate objective. We do this to prevent large policy changes that could make the actor objective, estimated from campaigns generated by the old policy, an unreliable guide for the update. In our implementation, Adam supplies a proposed change to the parameters ϕ. We shorten the proposed step until it passes two checks. First, the actor objective in (17), estimated from the current campaigns, must not decrease beyond a set numerical tolerance. Second, we compute the KL divergence between the old and proposed policies at each state visited in the current on-policy campaigns. Both the average and the largest divergence must stay below specified limits, recorded in Table 4.

Let $\widetilde{\phi}$ denote the parameters proposed by Adam and R the maximum number of halvings. We test the full proposal and then successively halve its displacement:

$$\phi^{(r)} = \phi_{\mathrm{old}} + 2^{-r}(\widetilde{\phi} - \phi_{\mathrm{old}}), \qquad r = 0, \dots, R. \tag{34}$$

For the state s_{it} at stage t of on-policy campaign i, define

$$k_{it}^{(r)} = D_{\mathrm{KL}}\big(\pi_{\mathrm{old}}(\cdot \mid s_{it}) \,\|\, \pi_{\phi^{(r)}}(\cdot \mid s_{it})\big). \tag{35}$$

We accept the first candidate satisfying

$$\frac{1}{B_{\mathrm{on}} T} \sum_{i=1}^{B_{\mathrm{on}}} \sum_{t=1}^{T} k_{it}^{(r)} \le \delta_{\mathrm{avg}}, \quad \max_{i,t} k_{it}^{(r)} \le \delta_{\max}, \quad \widehat{\mathcal{S}}(\phi^{(r)}) \ge \widehat{\mathcal{S}}(\phi_{\mathrm{old}}) - \varepsilon. \tag{36}$$

If no candidate passes, the actor parameters remain unchanged. Adam's optimizer state advances even when the parameter proposal is rejected.

D Experiment specifications and evaluation

This section gives the state encodings, numerical settings, baseline implementation, and evaluation procedure for the experiments in Section 5. The actor and critic receive the same state representation within each experiment.

D.1 Simulator and state specifications

We represent the posterior numerically by normalized probability masses on a fixed parameter grid. After each observation, we multiply each grid point's current mass by its likelihood and normalize:

$$p_{t+1,i} = \frac{p_{t,i}\, p(y_t \mid \theta_i, d_t)}{\sum_j p_{t,j}\, p(y_t \mid \theta_j, d_t)}. \tag{37}$$

Here $p_{t,i}$ is the posterior mass at grid point θ_i before experiment t. The probe has a discrete prior supported on this grid; for the jammer and spring, the grid approximates a continuous parameter distribution. The spring update uses the working-model likelihood. We compute the posterior means μ_t, standard deviations σ_t of each component of θ, and current posterior-to-prior KL divergence $b(s_t)$ from these masses. The grid coordinates and state features for each experiment are specified below.

D.1.1 Destructive probe

Parameter and design spaces. The campaign has five stages. DistBOED selects intensities from $\mathcal{D} = \{0, 0.1, \ldots, 1\}$, while DAD selects continuous intensities in $[0, 1]$. The parameter θ remains fixed throughout the campaign and takes values at the centers of a 32×32 grid in the unit square. The prior assigns normalized Gaussian weights:

$$p(\theta_i) = \frac{\exp\left[-\|\theta_i - (0.5, 0.5)\|^2/(2(0.22)^2)\right]}{\sum_{j=1}^{1024} \exp[-\|\theta_j - (0.5, 0.5)\|^2/(2(0.22)^2)]}. \tag{38}$$

Inference uses this discrete prior directly.

Observation and destruction models. Conditional on survival, intensity d_t produces a two-dimensional observation:

$$\begin{aligned} y_t &= \theta + \sigma(d_t)\varepsilon_t, \qquad \varepsilon_t \sim \mathcal{N}(0, I), \\ \sigma(d_t) &= \left(\frac{1 - d_t}{0.12^2} + \frac{d_t}{0.05^2}\right)^{-1/2}. \end{aligned} \tag{39}$$

Destruction occurs with probability $0.3d_t$ before the reading is obtained. Conditional on the selected intensity, destruction is independent of θ, so destruction alone does not change the posterior. Intensity zero is safe and has observation standard deviation 0.12; intensity one has standard deviation 0.05 conditional on survival.

After destruction, the remaining stages supply no observations and leave the posterior unchanged. We retain these stages in the fixed-length training records. The campaign return is the terminal posterior-to-prior KL divergence in (3).

State representation. The state contains a five-dimensional one-hot encoding of the experiment stage, the posterior mean and standard deviation of each of the two parameter components, an intact-specimen indicator, and the current posterior-to-prior KL divergence. These features give 11 inputs to both the actor and critic. DAD instead receives the history of intensities, readings, and indicators of whether each reading was obtained.

D.1.2 Source localization with a jammer

Source and receiver locations. The source position θ is drawn uniformly from the unit square and remains fixed throughout the campaign. Each of six measurements selects a receiver location from $\mathcal{D} = \{0.1, 0.3, 0.5, 0.7, 0.9\}^2$. The jammer is at the known location $j = (0.5, 0.5)$. Inference uses a uniform prior on the centers of a 65×65 grid in the unit square.

Observation model. The signal mean and added interference variance are

$$\mu(\theta, d) = \frac{0.1}{\sqrt{0.01 + \|d - \theta\|^2}}, \qquad I(d) = \min\left\{1, \frac{0.02}{0.01 + \|d - j\|^2}\right\}. \tag{40}$$

The observation at step t is

$$y_t = \mu(\theta, d_t) + \sqrt{\sigma_0^2 + B_t I(d_t)}\,\varepsilon_t, \qquad \varepsilon_t \sim \mathcal{N}(0,1), \tag{41}$$

where $\sigma_0 = 0.05$ and $B_t \sim \text{Bernoulli}(0.6)$. Jammer activity and measurement noise are independent across measurements. Activation probability does not depend on receiver location; location determines the added variance when the jammer is active.

The experimenter observes y_t but not B_t. Posterior updates therefore marginalize over jammer activity:

$$\begin{aligned} p(y \mid \theta, d) = 0.4\,\mathcal{N}\big(y; \mu(\theta,d), \sigma_0^2\big) \\ + 0.6\,\mathcal{N}\big(y; \mu(\theta,d), \sigma_0^2 + I(d)\big)\,. \end{aligned} \tag{42}$$

State representation and return. The state s_t contains:

- The posterior representation: for each of the 4,225 grid points, we use $\sqrt{4225 p_{t,i}}$, the square root of its posterior mass divided by its uniform prior mass.
- The posterior mean and standard deviation of each component of θ.
- The current posterior-to-prior KL divergence.
- The experiment stage, encoded as $(t-1)/6$.
- The number of previous visits to each of the 25 receiver sites, divided by six.

Together, these give 4,256 inputs to both the actor and critic. DAD instead receives the history of receiver locations and scalar readings. The campaign return is the posterior-to-prior KL divergence after all six measurements, as defined in (3).

D.1.3 Misspecified spring-mass

Dynamics and observations. The true dynamics are given in (19), with forcing frequency fixed at $\omega = 1.8$. The working model omits the cubic restoring force:

$$\ddot{x} + c_\text{d}|\dot{x}|\dot{x} + k_\text{l} x = d_t \sin(1.8u). \tag{43}$$

Each campaign contains eight tests, with amplitudes selected from $\mathcal{D} = \{0.15, 0.20, 0.25, 0.30, 0.35, 0.50\}$. Each test starts from rest, $x(0) = \dot{x}(0) = 0$, and produces 16 displacement observations at times $u_j = 0.75j$, for $j = 1, \ldots, 16$:

$$y_{t,j} = x_t(u_j) + \varepsilon_{t,j}, \qquad \varepsilon_{t,j} \sim \mathcal{N}(0, 0.08^2). \tag{44}$$

Here x_t solves the true dynamics under amplitude d_t. Measurement errors are independent across times and tests. The physical parameters remain fixed throughout the campaign. Displacement and velocity are reset between tests, while the working posterior retains all previous observations.

Parameter distributions and numerical implementation. Damping and linear stiffness have independent continuous log-uniform priors:

$$\log c_\text{d} \sim \text{Uniform}(\log 0.1, \log 0.5), \qquad \log k_\text{l} \sim \text{Uniform}(\log 0.5, \log 2). \tag{45}$$

Cubic stiffness is independent of these parameters and follows

$$\log k_\text{c} \sim \mathcal{N}(\log 0.15, 1). \tag{46}$$

All three coefficients are sampled once per campaign. The working model uses the same priors for damping and linear stiffness but sets $k_\text{c} = 0$. We solve the dynamics using DOP853 (Hairer et al., 1993), with relative tolerance 10^{-7} and absolute tolerance 10^{-9}. Working-posterior integrals use midpoint quadrature on a 65×65 grid in log damping and log stiffness.

State representation. The state contains all 4,225 working-posterior masses, the stage fraction $(t-1)/8$, the padded design–observation history, and the current working-posterior KL. Each history entry contains the forcing frequency divided by 1.8, amplitude divided by 0.5, and 16 observations divided by the observation-noise standard deviation 0.08. The frequency feature is retained although frequency is fixed in this experiment. Unobserved history entries are zero. These features give 4,371 inputs.

Campaign return. We score each campaign using generalized information gain evaluated at the sampled true damping and stiffness $\theta^\star$ (Callahan & Catanach, 2026; Catanach & Das, 2023):

$$\begin{aligned} g = G_{\mathrm{gen}}(\theta^\star, h_T) &= \log \frac{p_{\mathrm{work}}(\theta^\star \mid h_T)}{p(\theta^\star)} \\ &= \log \ell_{\mathrm{work}}(h_T; \theta^\star) - \log \int \ell_{\mathrm{work}}(h_T; \theta) p(\theta)\, d\theta. \end{aligned} \tag{47}$$

Here ℓ_{work} is the product of working-model likelihoods across the eight tests. The observations come from the nonlinear simulator, while the working likelihood sets cubic stiffness to zero. Negative returns indicate that the working posterior assigns less density to the true parameters than the prior does. The sampled parameters are used to score the campaign but are not included in the policy state.

Averaging this return over the true joint distribution of parameters and campaign histories gives expected generalized information gain:

$$\mathrm{EGIG}(\pi) = \mathbb{E}_{p_{\mathrm{true},\pi}(\theta, h_T)} \left[G_{\mathrm{gen}}(\theta, h_T) \right]. \tag{48}$$

Mean training therefore optimizes EGIG, while risk-sensitive training uses the distribution of the realized return in (47). When the true and working models coincide, the conditional average equals the posterior-to-prior KL divergence, although individual realized returns can still be negative.

DAD adaptation Standard DAD (Foster et al., 2021) trains policies using sPCE, which compares the likelihood of a campaign under its generating parameter with an average likelihood over that parameter and independent prior draws. Using the same model to generate observations and evaluate these likelihoods gives a lower bound on EIG.

For the misspecified spring experiment, we retain the DAD policy architecture but generate campaigns under the true dynamics in (19) and evaluate likelihoods under the working model in (43). This separation allows the training objective to reflect whether working-model inference improves beliefs about the true parameters when observations follow the nonlinear dynamics.

We evaluate the denominator using the same 65×65 working-prior quadrature as DistBOED, replacing the sampled likelihood average. The numerator remains the working likelihood at the continuous sampled parameter $\theta^\star$, which is not added to the quadrature grid. With exact integration, this log ratio equals the realized generalized information gain in (47), whose expectation under the true simulator is EGIG. Our training objective therefore approximates EGIG, although it does not retain sPCE's lower-bound guarantee.

D.2 DAD BASELINE

We compare against Deep Adaptive Design (DAD) (Foster et al., 2021) in the probe and jammer experiments. Both implementations optimize sequential prior contrastive estimation with 511 prior contrasts. Each design–observation pair passes through an encoder with two width-128 ReLU hidden layers and a 32-dimensional output. The pair embeddings are summed, and a design network with one width-128 ReLU hidden layer produces the next design.

For the probe, each history entry contains the selected intensity, two observed coordinates, and an indicator of whether a reading was obtained. A sigmoid output gives a continuous intensity in $[0, 1]$. Training combines derivatives through the observations with a likelihood-ratio gradient for the destruction events. We use 10,200 updates with 128 campaigns per update and decrease the learning rate from 10^{-3} to 10^{-4}.

For the jammer, each history entry contains the receiver coordinates and scalar reading. The network outputs probabilities over the 25 receiver sites. Training evaluates a continuation for every candidate site at each decision, sharing the source, jammer activity, observation noise, and contrastive parameters across these continuations. The gradient sums the contributions of all sites. We use 10,000 updates with 64 initial campaigns per update and decrease the learning rate from 10^{-2} to 10^{-3}. Probe DAD and each DistBOED policy use 1,305,600 training campaigns. Jammer DAD uses 96,640,000 simulated campaigns compared with 1,305,600 for each DistBOED policy.

	Probe	Jammer	Spring
Evaluation campaigns	20,000	8,192	4,096
Policy posterior grid	32×32	65×65	65×65
Evaluation grid	32×32	129×129	257×257

Table 6: Initialization and evaluation settings. Policies within each setting share the evaluation parameters and simulator randomness. Spring grids use log parameter coordinates.

Training objective	Parameter	Mean	Variance	$\text{CVaR}_{0.1}$
Mean	—	2.087	1.507	0.441
Mean–variance	$\lambda = 0.25$	2.028	1.362	0.442
	$\lambda = 0.5$	1.797	0.664	0.496
	$\lambda = 1$	1.525	0.288	0.616
CVaR	$\alpha = 0.1$	1.335	0.239	0.661
	$\alpha = 0.25$	1.384	0.238	0.703
	$\alpha = 0.5$	1.728	0.620	0.529
Entropic	$\beta = 0.5$	2.027	1.398	0.442
	$\beta = 1$	1.986	1.283	0.469
	$\beta = 2$	1.692	0.580	0.579

Table 7: Sensitivity to risk parameters in the jammer experiment. All policies are evaluated on the same 8,192 campaigns using the 129×129 scoring grid. The final column reports CVaR at the common evaluation level 0.1, regardless of the training objective.

D.3 Evaluation protocol

Within each experiment, we evaluate all policies using the same sampled parameters and random draws for observation noise and other simulator events. Each policy selects its own designs, so its observations and final return can differ. Evaluation returns are computed independently of the learned critic using campaign rollouts. Table 6 gives the campaign counts, scoring grids.

Reported statistics. We compute all statistics from evaluated campaign returns, retaining negative spring gains without clipping. CVaR uses the empirical lower-tail average at the specified probability level. Where confidence intervals are reported, we use 2,000 paired bootstrap resamples of complete campaigns.

E Additional results

E.1 Receiver selections across the full campaign

Fig. 10 extends Fig. 5 to all six experiments. Each panel aggregates receiver selections across evaluation campaigns. We see similar behavior across all time steps that we do in the first two, discussed in Section 5.2.

E.2 Sensitivity to risk preferences

We vary the risk parameters in the jammer experiment while keeping the architecture, training procedure, initialization seed, and evaluation campaigns fixed. Each parameter setting produces a separately trained policy. Larger λ penalizes variance more strongly, larger β increases the entropic penalty for low returns, and smaller α focuses CVaR on a smaller fraction of low-return campaigns.

Table 7 shows that increasing either the variance penalty or the entropic coefficient produces policies with lower variance and better worst-decile outcomes, at a cost to mean IG. Thus, these parameters change the balance between average information and protection against low-information campaigns.

The CVaR results are not monotonic in the training level: the policy trained at $\alpha = 0.25$ achieves a higher evaluated $\text{CVaR}_{0.1}$ than the policy trained directly at $\alpha = 0.1$, while also retaining more

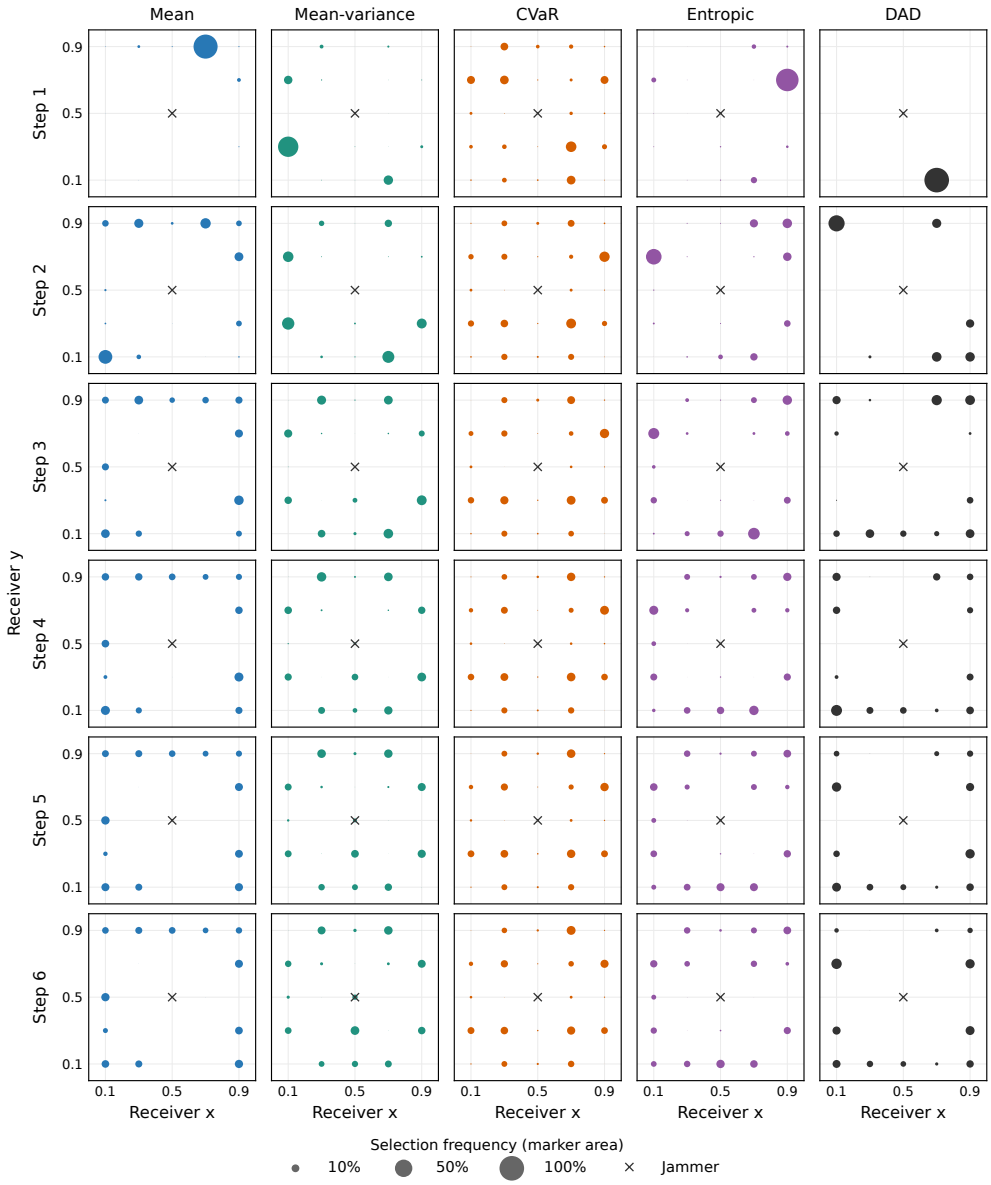

Figure 10: Receiver-site selection frequencies across the six-step source-localization campaign. Rows show experiment stages, and columns show policies. Dot area is proportional to selection frequency, with a common scale across panels. Frequencies in each panel sum to one. The cross marks the jammer location.

mean IG. A stronger emphasis on the lowest returns therefore does not necessarily produce better lower-tail performance after training.