Bayesian Experimental Design via Score Matching

Angus Phillips[1] **Gavin Kerrigan**[1] **Tom Rainforth**[1]

[1]Department of Statistics, University of Oxford

Abstract

Policy-based approaches to Bayesian experimental design (BED) allow the learning of deep policy networks that adaptively make intelligent design decisions based on previously collected data. However, the training of such policies is often held back by a fundamental challenge: the double intractability of the expected information gain (EIG). This necessitates expensive or complex approximations that restrict the effort one can invest in optimising the policy itself. To address this, we show that the double intractability of the EIG can be isolated from the policy learning by first solving a score matching problem that is independent of the policy used, then using the learned score approximation to train the policy in a singly intractable manner. This turns the key multiplicative cost into an additive one and reduces the computational burden on the policy training itself, making it far cheaper to train the policy multiple times when needed, e.g. for architecture search, hyperparameter tuning, or avoiding local optima. In our experiments we train multiple competitive policies without inducing a multiplicative cost in likelihood evaluations, which can increase performance by allowing us to select the best policy even without performing hyperparameter or architecture searches.

1 INTRODUCTION

The optimal design of experiments is critical for conducting informative and cost-effective scientific experiments in any field. Bayesian Experimental Design (BED) (Chaloner and Verdinelli, 1995; Huan et al., 2024; Lindley, 1956; Rainforth et al., 2024) offers a theoretically principled approach to selecting experimental designs grounded in Bayesian decision theory that has found successful application in fields as broad as physics (Dushenko et al., 2020; McMichael et al., 2021), psychology (Watson, 2017), quantum computing (Sarra and Marquardt, 2023), machine learning (Gal et al., 2017) and beyond. Typically, BED is performed by maximising the expected information gain (EIG) of the experiment(s) (Lindley, 1956), where the information gain is given by the reduction in Shannon entropy (Shannon, 1948) from our prior to posterior beliefs in some target quantities of interest, and the expectation is over possible data we might see.

Of particular importance in many applications is the ability to propose the next design in real time, such as conducting a live survey with human participants (Pasek and Krosnick, 2010; Vincent and Rainforth, 2017) or learning via control of a live dynamical system (Iollo et al., 2025). Unfortunately, this can be difficult in practice, as the EIG is a doubly intractable quantity that is very challenging to optimise (Rainforth et al., 2018, 2024), while the need to repeatedly do inference between iterations can also be problematic.

Policy-based BED (Blau et al., 2022; Bracher et al., 2025; Foster et al., 2021; Hedman et al., 2025; Iqbal et al., 2024; Ivanova et al., 2021; Shen and Huan, 2023) offers the most viable approach to overcoming this, by amortising adaptive design decisions into a pre-trained policy which can be deployed in a single forward pass. However, the problem of optimising a doubly intractable objective is now transferred to the policy training itself, where the challenges can actually be compounded due to the complexity of policy learning, which often forms a significantly harder optimisation than just optimising designs (Blau et al., 2022). Furthermore, variational approaches to side-stepping the double intractability (Foster et al., 2019, 2020; Kleinegesse and Gutmann, 2019) are often challenging to apply due to the difficulty of learning accurate variational distributions in high-dimensional settings. Existing approaches often thus simply resort to nested or contrastive sampling of the EIG gradient, accepting the substantial computational costs this induces.

To address this, we show that it is possible to disentangle the double intractability of the EIG from the training of a

Proceedings of the 42nd Conference on Uncertainty in Artificial Intelligence (UAI 2026), PMLR 337:5393–5426.

policy, utilising the fact that the information gain itself depends on the policy only through the chosen designs and not the policy parameters themselves. Specifically, we derive a reparameterised form of the EIG gradient where the double intractability is isolated in two terms with no direct dependency on the policy: the Stein score and Fisher score of the marginal likelihood given the full design rollout. We then use this to introduce a score matching approach to BED, which we call SCOREBED. SCOREBED first learns a single network which approximates both of these scores using an objective we call marginal score matching, then substitutes the learned score network into our reparameterised EIG gradient, yielding a singly intractable estimator that can be used to train the policy via stochastic gradient descent.

A key benefit of our two-stage setup is that it transforms the typically multiplicative cost of nested sampling procedures into an additive cost of two singly intractable problems, allowing gradient-based policy training without expensive nested sampling. Critically, the approximations we learn are entirely independent of any policy and thus the same upfront approximation can be reused repeatedly to allow sample-efficient and scalable training for any number of policies. This provides substantial benefits to policy training by allowing cheaper hyperparameter and architecture searches, longer policy training, or multiple retrainings to avoid local optima, all of which can be important in practice due to the difficulty of policy training (Ivanova et al., 2021).

Experimentally, we find that, when matching total budgets, SCOREBED provides competitive performance compared to existing approaches that simultaneously deal with double intractability of the EIG and policy training. Moreover, as only a small proportion of the budget in SCOREBED is on the policy training itself, it can cheaply train multiple such competitive policies, thereby paving the way for more nuanced and computationally intensive policy training schemes.

2 BACKGROUND

Bayesian Experimental Design. In BED, one assumes a Bayesian model for an experiment consisting of prior $p(\theta)$ and likelihood $p(y \mid \theta, \xi)$ where ξ denotes the design, y is the experiment outcome, and θ are the target variables we wish to learn about. The goal is to select ξ to maximise some expected utility function under the data generating process. The standard choice of utility function in BED literature is the information gain on model parameters, defined by Lindley (1956) as the reduction in Shannon's entropy from the prior to the posterior, $\mathrm{IG}_\theta(y, \xi) = \mathbb{H}[p(\theta)] - \mathbb{H}[p(\theta \mid y, \xi)]$. Taking expectation under the data generating process yields the EIG objective $\mathcal{I}(\xi) = \mathbb{E}_{p(y|\xi)}\{\mathrm{IG}_\theta(y, \xi)\}$.

We consider a sequential setting with T rounds of experiments, therefore let $h_t = \{y_s, \xi_s\}_{s=1}^t$ denote the experimental history to time t. Following Foster et al. (2021), we learn an adaptive policy network $\pi_\phi(\xi_t \mid h_{t-1})$ which is trained to propose the optimal next design given h_{t-1} on the fly. There are several advantages of such a policy-based approach (Foster et al., 2021; Rainforth et al., 2024), including significantly faster deployment times versus greedy approaches, avoiding the need for potentially inaccurate inference at each time step, and making non-myopic decisions by optimising up front for the total information gain over the entire experiment. In this policy-based setting, we will write the data generating process under the policy π_ϕ[1] as $p_\phi(y_{1:T}, \xi_{1:T}) = \int p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta) \mathrm{d}\theta$ where

$$p_\phi(y_{1:T}, \xi_{1:T} \mid \theta) = \prod_{t=1}^{T} \pi_\phi(\xi_t \mid h_{t-1}) p(y_t \mid \theta, \xi_t, h_{t-1}).$$

Foster et al. (2021) propose optimising the total expected information gain over the whole experiment, therefore giving the policy-based EIG objective:

$$\mathcal{I}_T(\phi) = \mathbb{E}_{p_\phi(y_{1:T}, \xi_{1:T})}\{\mathrm{IG}_\theta(y_{1:T}, \xi_{1:T})\}, \tag{1}$$

where $\mathrm{IG}_\theta(y_{1:T}, \xi_{1:T}) = \mathbb{H}[p(\theta)] - \mathbb{H}[p(\theta \mid y_{1:T}, \xi_{1:T})]$.

Policy-based BED is therefore concerned with finding the optimal policy $\phi^* = \arg\max_\phi \mathcal{I}_T(\phi)$. We take a stochastic gradient-based approach to this problem, although previous works have also searched for designs using Bayesian optimisation (Snoek et al., 2012), such as Foster et al. (2019); Kleinegesse and Gutmann (2019).

Approximating the EIG. A key challenge in BED comes from the double intractability of the objective. The EIG can be written either as an expected KL divergence between the posterior and prior:

$$\mathbb{E}_{p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta)}\{\log p(\theta \mid y_{1:T}, \xi_{1:T}) - \log p(\theta)\},$$

or as the mutual information between $y_{1:T}$ and θ given $\xi_{1:T}$:

$$\mathbb{E}_{p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta)} \left\{ \log \frac{p(y_{1:T} \mid \theta, \xi_{1:T})}{p(y_{1:T} \mid \xi_{1:T})} \right\}.$$

In both cases it is clear the objective is doubly intractable as it involves evaluating the posterior or marginal distributions which appear non-linearly within the outer expectation.

Foster et al. (2020) introduce the Prior Contrastive Estimation (PCE) bound which lower bounds the EIG. This bound uses M contrastive samples from the prior to approximate the intractable marginal distribution:

$$\mathcal{I}_T(\xi_{1:T}) \geq \mathbb{E} \left\{ \log \frac{p(y_{1:T} \mid \theta_0, \xi_{1:T})}{\frac{1}{M+1} \sum_{m=0}^{M} p(y_{1:T} \mid \theta_m, \xi_{1:T})} \right\}.$$

The inclusion of θ_0 in the estimate of the intractable marginal reduces the variance of the estimate compared with ordinary nested Monte Carlo (NMC). The PCE estimator can

[1] When $\pi_\phi(h_{t-1})$ is deterministic, this distribution becomes degenerate. We ignore the measure-theoretic considerations in this case and continue to write $p(y \mid \xi)$ even when ξ is deterministic.

be differentiated using a reparameterisation trick (Kingma and Welling, 2013) or a REINFORCE gradient estimator (Williams, 1992), allowing gradient-based optimisation.

An alternative approach is to construct a functional or variational approximation of the intractability in the EIG. Foster et al. (2019) propose replacing the intractable posterior with a variational approximation $q(\theta \mid y_{1:T}, \xi_{1:T})$, which they show results in a lower bound of the EIG due to the Barber-Agakov bound (Barber and Agakov, 2003). Alternatively, they show that using a variational approximation of the intractable marginal $q(y_{1:T} \mid \xi_{1:T})$ leads to an upper bound on the EIG. Foster et al. (2020) further propose a unified procedure which co-optimises the variational approximation and the designs, meaning the approximation is specialised to the particular policy network.

Score Matching. Score matching (Hyvärinen, 2005) refers to techniques used to approximate the Stein score $\nabla_z \log p(z)$ of a distribution with density $p(z)$ by minimising the explicit score matching objective $\frac{1}{2}\mathbb{E}_z\{\|s_\psi(z) - \nabla \log p(z)\|^2\}$. Being typically intractable, the target score is unavailable to us, therefore equivalent tractable objectives have been proposed (Hyvärinen, 2005; Song et al., 2019; Vincent, 2011) such as the sliced score matching objective:

$$\mathcal{J}_{\text{SSM}}(\psi) = \mathbb{E}_\nu \mathbb{E}_z\big\{v^T \nabla_z s_\psi(z) v + \frac{1}{2}(v^T s_\psi(z))^2\big\} \quad (2)$$

which only requires samples of z.

Score matching has achieved particular success in denoising diffusion generative models (Ho et al., 2020; Song and Ermon, 2019), where a variant known as denoising score matching (Vincent, 2011) is used to learn the noise-conditional score function of the generative process. This has proved successful even in high-dimensional and conditional settings (Dhariwal and Nichol, 2021; Rombach et al., 2022). Denoising score matching considers a noisy observation $\tilde{z}$ of z, according to the Gaussian convolution $p(\tilde{z}) = \int p(z)p(\tilde{z} \mid z)\mathrm{d}z$ where $p(\tilde{z} \mid z)$ is a noising kernel. The score of the noised distribution is shown to satisfy $\nabla_z \log p(\tilde{z}) = \int \nabla \log p(\tilde{z} \mid z)p(z \mid \tilde{z})\mathrm{d}z$ which is the unique minimiser of the regression objective $\mathcal{J}_{\text{DSM}}(\psi) = \mathbb{E}_{z,\tilde{z}}\big\{\|s_\psi(\tilde{z}) - \nabla \log p(\tilde{z} \mid z)\|^2\big\}$.

3 METHOD

We contribute three significant observations and results in the following section, which culminate in our score-based method for EIG optimisation of policies. Firstly, we observe that the information gain is fundamentally independent of how the observed data was collected, so approximations to deal with the double intractability do not necessarily have to depend on the policy. Secondly, we derive a precise EIG gradient expression that shows the exact source of the double intractability which we aim to resolve in a way that is detached from the policy. Thirdly, we show how these intractabilities can be naturally approximated with a score network trained on a specific score matching loss. Combining these facts leads to our SCOREBED approach, where we first learn an accurate and computationally favourable approximation of the EIG gradient upfront, then use this learned approximation for training policies. This transforms the computationally challenging, doubly intractable EIG objective into two sequential singly intractable problems.

The importance of this separation is that it can offer significant computational benefits by replacing a multiplicative cost (number of training steps times number of inner samples) with an additive cost (cost of training the score network plus cost of the resulting *singly intractable* policy training). By substantially reducing the burden on the policy training part of the overall procedure, this gives scope to perform architecture adjustments, hyperparameter tuning, and even multiple training restarts to obtain a good policy network (Rainforth et al., 2024). Historically this required all the computation of the full procedure to be repeated again; for our score matching approach it only requires the now singly intractable policy training to be repeated, providing substantial gains. This is particularly important if likelihood evaluations are expensive, and is expected to be increasingly beneficial as the problems we work with require more complex policies and training mechanisms (Blau et al., 2022).

Separating intractabilities from policies. We begin with an important observation that, to the best of our knowledge, has not previously been noted in the context of policy-based BED: the information gain $\text{IG}_\theta(y_{1:T}, \xi_{1:T}) = \mathbb{H}[p(\theta)] - \mathbb{H}[p(\theta \mid y_{1:T}, \xi_{1:T})]$ itself has no direct dependence on the design policy but is simply a function of the data $(y_{1:T}, \xi_{1:T})$, i.e. for given data it is independent of how that data was collected (see Appendix A.1 for proof). The policy dependence of the EIG only enters through the outer expectation over data, thus the double intractability of the information gain is completely separable from the policy. The upshot of this is that any approximation of the intractabilities in the information gain has a global optimum that is independent of the policy (assuming our approximator has infinite capacity). We can thus learn approximation schemes upfront that are policy-independent, then apply them as fixed approximations for training any policy.

EIG gradient expression. The critical quantity required for gradient-based training of policies is the gradient of the EIG objective with respect to policy parameters, $\nabla_\phi \mathcal{I}_T(\phi)$. Existing approaches typically only consider estimating this quantity indirectly, by differentiating through an estimate or bound of the EIG itself. However, variational bounds may be uninformative if the variational approximation is not accurate and differentiating such an approximation does not guarantee an accurate approximation of its gradient. We therefore sidestep these potential challenges by choosing to approximate the gradient of the EIG directly.

To that end, we present an exact EIG gradient expression which clearly highlights the exact nested intractabilities that are present, as well as how the policy parameters ϕ enter the expression *independently* of those intractabilities.

Theorem 1 (Reparameterised EIG Gradient Expression)**.** *The following expression for the gradient of the EIG holds:*

$$\nabla_\phi \mathcal{I}_T(\phi) = \mathbb{E}_{p(\theta)q(\epsilon)}\Big\{ \frac{\mathrm{d}}{\mathrm{d}\phi} \log p(y_{1:T} \mid \xi_{1:T}, \theta) - \frac{\partial}{\partial y_{1:T}} \log p(y_{1:T} \mid \xi_{1:T}) \frac{\partial y_{1:T}}{\partial \phi} - \frac{\partial}{\partial \xi_{1:T}} \log p(y_{1:T} \mid \xi_{1:T}) \frac{\partial \xi_{1:T}}{\partial \phi} \Big\} \tag{3}$$

where the data and designs are obtained by reparameterisation, $(y_{1:T}, \xi_{1:T}) = g(\epsilon, \theta, \phi)$, with g the reparameterisation map of the data-generating process and $\epsilon \sim q(\epsilon)$ a set of noise variables independent of ϕ, so that $(y_{1:T}, \xi_{1:T}) \sim p_\phi(\cdot, \cdot \mid \theta)$. The explicit form of g is deferred to Appendix A.2 for brevity. In the specific case of static (batch) designs, the expression simplifies to:

$$\nabla_{\xi_{1:T}} \mathcal{I}_T(\xi_{1:T}) = \mathbb{E}_{p(\theta)q(\epsilon)}\Big\{ \frac{\mathrm{d}}{\mathrm{d}\xi_{1:T}} \log p(y_{1:T} \mid \theta, \xi_{1:T}) - \frac{\partial}{\partial y_{1:T}} \log p(y_{1:T} \mid \xi_{1:T}) \frac{\partial y_{1:T}}{\partial \xi_{1:T}} \Big\} \tag{4}$$

where again $y_{1:T} = g(\epsilon, \theta, \xi_{1:T})$ via reparameterisation.

Proof. See Appendix A.2. □

The expression in Equation (3) now reveals the inner structure of the nested intractabilities and dependence on policy parameters ϕ in the EIG gradient. In particular, the intractable terms are the Stein score function $\frac{\partial}{\partial y_{1:T}} \log p(y_{1:T} \mid \xi_{1:T})$ and the Fisher score function $\frac{\partial}{\partial \xi_{1:T}} \log p(y_{1:T} \mid \xi_{1:T})$. Importantly, these terms are *evaluated* at data and designs $(y_{1:T}, \xi_{1:T})$ which are functions of the policy parameter ϕ due to reparameterisation, but the terms themselves are entirely conditionally independent of ϕ *given* the designs.

A key intuition here that has not been highlighted by previous work is that the marginal likelihood term $p(y_{1:T} \mid \xi_{1:T})$ that appears in the EIG and its gradient is not the same as the marginal distribution on data for a given policy $p(y_{1:T}; \pi_\phi)$. Thus, even though the data distribution obviously depends on our policy, the actual intractable term we are dealing with is given by $\mathbb{E}_{p(\theta)}[p(y_{1:T} \mid \xi_{1:T}, \theta)]$, which does not depend on the policy given the designs.

Additionally, Theorem 1 shows analytically that the Fisher score term disappears in the case of static designs. This result reduces bias and variance of our gradient approximation in this case, and could also be used to improve EIG gradient estimation when not using our score-based approach.

Algorithm 1 SCOREBED

Input: prior $p(\theta)$, likelihood $p(y_{1:T} \mid \xi_{1:T}, \theta)$, reparameterisation map $(y_{1:T}, \xi_{1:T}) = g(\epsilon, \theta, \phi)$, design sampler $q(\xi_{1:T})$, policies $\{\pi_{\phi_1}, \ldots, \pi_{\phi_P}\}$
Output: trained policies $\{\pi_{\phi_1}, \ldots, \pi_{\phi_P}\}$

***Stage 1:** Learn intractable score (policy-independent).*
for score budget **do**
  Sample batch $\{\xi^b_{1:T} \sim q(\xi_{1:T}),\ \theta^b \sim p(\theta),\ y^b_{1:T} \sim p(y_{1:T} \mid \xi^b_{1:T}, \theta^b)\}_{b=1}^B$
  $\mathcal{J}_{\text{MSM}}(\psi) \leftarrow \frac{1}{B}\sum_{b=1}^B \| s_\psi(y^b_{1:T}, \xi^b_{1:T}) - \nabla_{y,\xi} \log p(y^b_{1:T} \mid \xi^b_{1:T}, \theta^b) \|^2$
  $\psi \leftarrow \text{UPDATE}(\psi, \nabla_\psi \mathcal{J}_{\text{MSM}}(\psi))$
end for

***Stage 2:** Train each policy with the fixed score s_ψ.*
for $p = 1, \ldots, P$ **do**
  for policy budget **do**
    Sample batch $\{\theta^n \sim p(\theta),\ \epsilon^n \sim q(\epsilon)\}_{n=1}^N$
    $(y^n_{1:T}, \xi^n_{1:T}) \leftarrow g(\epsilon^n, \theta^n, \phi_p)$
    $\widehat{\nabla_\phi \mathcal{I}_T}(\phi_p) \leftarrow \sum_{n=1}^N \Big[\mathrm{d}_{\phi_p} \log p(y^n_{1:T} \mid \xi^n_{1:T}, \theta^n) - s_\psi(y^n_{1:T}, \xi^n_{1:T})\, \partial_{\phi_p}(y^n_{1:T}, \xi^n_{1:T}) \Big] / N$ (eq. 3)
    $\phi_p \leftarrow \text{UPDATE}(\phi_p, \widehat{\nabla_\phi \mathcal{I}_T}(\phi_p))$
  end for
end for
return $\{\pi_{\phi_1}, \ldots, \pi_{\phi_P}\}$

BED via score-matching. Theorem 1 makes clear that we need to estimate or approximate $\frac{\partial}{\partial y_{1:T}} \log p(y_{1:T} \mid \xi_{1:T})$ and $\frac{\partial}{\partial \xi_{1:T}} \log p(y_{1:T} \mid \xi_{1:T})$ (whether this is done implicitly or directly) as part of an overall scheme for estimating the EIG gradient. Previous work has either done this through estimates of the marginal likelihood that are recalculated for each new $(y_{1:T}, \xi_{1:T})$, or using a variational approximation of either the marginal likelihood or posterior, with gradients then taken of the approximation itself.

We propose to instead learn these terms *directly* using score matching. We argue that score matching is a natural choice in this scenario for a number of reasons. Firstly, it explicitly targets accuracy on the intractable score, which as we have shown is the key quantity in the EIG gradient. While one could instead take any density estimator or variational approximation of the intractable marginal and differentiate its log density to approximate the score, such an approach may not be accurate in practice. Indeed, the score function can be expressed as $\nabla_z \log p(z) = \nabla_z p(z)/p(z)$ and density estimation or variational approximations do not control gradient values. In other words, $p \to \nabla \log p$ is not continuous in the metrics used for density estimation or variational approximation. Secondly, the score function is independent of the normalisation constant and therefore the expressivity

of a score network is not limited by the restrictions imposed by normalisation which are found in variational approaches. Finally, score matching is simply a supervised regression problem which scales well with dimension and is not complicated by gradient variance in the same way as variational methods (Kingma and Welling, 2013).

To guarantee the accuracy of our EIG gradient approximation in practice, we show in Appendix B that if the reparameterisation map $\phi \mapsto (y_{1:T}, \xi_{1:T})$ is L-Lipschitz, the ℓ_2 error of our gradient estimator is bounded above by L times the mean score error. Achieving accurate EIG gradient estimates is therefore simply a question of score network capacity, training budget, and avoiding local optima, noting we have access to unlimited model samples during training.

Marginal score matching. We propose a score matching objective which directly recovers the intractable score functions from the likelihood of the model. Removing dependence on t for ease of notation, we are required to learn $\nabla_{y,\xi} \log p(y \mid \xi)$ where $p(y \mid \xi) = \int p(y \mid \xi, \theta) p(\theta) \mathrm{d}\theta$. A simple score identity is as follows:

$$\begin{aligned} \nabla_{y,\xi} \log p(y \mid \xi) &= \nabla_{y,\xi} \log \left(\int p(y \mid \xi, \theta) p(\theta) \mathrm{d}\theta \right) \\ &= \int \nabla_{y,\xi} \left[\log p(y \mid \xi, \theta) \right] p(\theta \mid \xi, y) \mathrm{d}\theta \end{aligned} \tag{5}$$

This identity has been considered by Ao and Li (2024); Brehmer et al. (2020) as the basis for Monte Carlo score approximations, but has not been used for training score networks. This inspires the following regression objective, which we refer to as marginal score matching (MSM):

$$\mathcal{J}_{\text{MSM}}(\psi) = \mathbb{E}_{\xi,y,\theta}\{\| s_\psi(y, \xi) - \nabla_{y,\xi} \log p(y \mid \xi, \theta) \|^2\}, \tag{6}$$

where the expectation is taken over the joint distribution of the designs ξ, observations y and parameters θ, and $s_\psi(y, \xi)$ is a deep neural network with parameters ψ which we will use as an approximation to the intractable score functions. It is easy to show that the parameters ψ^* minimising eq. (6) are such that $s_{\psi^*}(y, \xi) = \nabla_{y,\xi} \log p(y \mid \xi)\ \forall (y, \xi); q(y, \xi) > 0$ where q is the distribution used to sample (y, ξ). Critically, the objective requires no direct sampling from the posterior, only the joint on ξ, y, θ. Consequently, our approach is suitable in models with high-dimensional parameter spaces where nested sampling approaches suffer exponential sample complexity with parameter space dimension. Our regression targets live in the design and output spaces which are unaffected in complexity by parameter dimension.

Moreover, assuming a sufficiently powerful score network architecture, we are free to take the expectation over ξ with respect to *any* distribution that has support over all possible designs, because we are simply regressing from (y, ξ) pairs to scores and $s_{\psi^*}(y, \xi)$ for any given ξ is independent of our distribution on ξ. Thus, we do not need to sample according to the policy and can instead sample from the joint $\tilde{q}(\xi, y, \theta) = q(\xi) p(\theta) p(y \mid \xi, \theta)$, where $q(\xi)$ is simply chosen to reflect where we most want the score network to be accurate. In our experiments we found that sampling $\xi_{1:T}$ from a simple design distribution (e.g. uniform over the design space) was usually sufficient and maintained independence of the approximation from the policy. We experiment with different choices of $q(\xi)$ in Appendix E.3.

We also point out that we could alternatively train the score network by minimising an implicit score matching loss such as the sliced score matching (SSM) loss (Song et al., 2019), which does not rely on the model likelihood. We found the MSM loss performed better, however the SSM loss is useful for settings with implicit or expensive likelihoods.

Score network architecture. We provide full details of our score network architecture in Appendix D.4, however we remark on two important points in the following. Firstly, in learning $\nabla_{y,\xi} \log p(y \mid \xi)$ we simultaneously recover each of the required score terms $\frac{\partial}{\partial y} \log p(y \mid \xi)$ and $\frac{\partial}{\partial \xi} \log p(y \mid \xi)$ from the same score network. We found that a shared backbone with separate prediction heads for each score component was the most efficient architecture. Secondly, when observations y_t are conditionally i.i.d. given (θ, ξ), the marginal likelihood $p(y_{1:T} \mid \xi_{1:T})$ is exchangeable with respect to pairs (y_t, ξ_t). We encode this permutation equivariance as an inductive bias by using a TNP-style architecture with self-attention (Nguyen and Grover, 2022; Vaswani et al., 2017) and add positional encodings to (y_t, ξ_t) in non-exchangeable cases.

Applicability. SCOREBED relies on explicit likelihoods with differentiable reparameterisations, however these are typical assumptions in alternative approaches (Ao and Li, 2024; Foster et al., 2021; Goda et al., 2022; Iollo et al., 2025). Our method also requires continuous design and observation spaces in order for the Stein and Fisher score functions to exist. We highlight that a large number of BED problems satisfy these assumptions, meaning that our method is still a practical BED approach for a wide class of problems.

4 RELATED WORK

The main body of BED literature has revolved around estimating or bounding the EIG itself, for instance contrastive estimation (Foster et al., 2021) and variational approaches (see e.g. Rainforth et al. (2024) and the references therein). Some recent works have further explored direct EIG gradient estimation. Goda et al. (2022) introduce an unbiased estimate of the EIG gradient based on multi-level Monte Carlo (Rhee and Glynn, 2015), which acts as a de-biasing scheme in finite samples, at the cost of increased variance. Ao and Li (2024) also directly target the EIG gradient using expressions similar to eq. (3) and eq. (5). Contrary to our amortisation approach, they use MCMC to sample the posterior and estimate eq. (5) via Monte Carlo, leading to a high cost per gradient update when training policies and poor

sample complexity in high parameter dimensions. A final direct EIG gradient approach is Iollo et al. (2025), who use sampling as optimisation to roll the nested sampling problem into a single loop, although they do not train policies.

A closely related family of approaches is the variational methods introduced by Foster et al. (2019, 2020) and further adopted by Bracher et al. (2025); Dong et al. (2025); Huang et al. (2026b). The original work of Foster et al. (2019) applied a two-stage procedure in the case of static designs, learning the variational approximation up front and using the resulting bound on the EIG to optimise designs. However, subsequent policy-based variational approaches (Bracher et al., 2025; Dong et al., 2025; Huang et al., 2026b) co-train the policy and the variational approximation rather than recognising the variational approximation as a global approximation independent of the policy. Furthermore, the BED literature typically considers a policy-dependent data distribution $p(y_{1:T}; \pi_\phi)$ and thus considers the posterior to be dependent on the policy π_ϕ. We instead emphasise—as shown in Appendix A.1—that the posterior never depends on the policy, nor more generally on any sampling or observation mechanism (Rubin, 1976) under appropriate assumptions. As such, variational methods can be applied in the same two-stage procedure which we adopt, and we demonstrate this in our experiments in Section 5.

Concurrent work by Huang et al. (2026a) also reduces the cost of policy training, leveraging belief representations from a foundation model pre-trained independently of any policy, thereby relieving policy training from learning representations. They approximate the EIG using PCE, so their cost saving is complementary to ours. Finally, Ivanova et al. (2021); Kleinegesse and Gutmann (2020) take an amortised approach for implicit models using critics to build functional approximations which bound the mutual information, while Hedman et al. (2025) consider periodic retraining of the policy during the experiment. We also note connections to gradient estimators for implicit models, particularly those leveraging score matching as proposed by Song et al. (2019). One such work is Lim et al. (2020), who approximate the gradient of the entropy of an implicit distribution using denoising autoencoders (Vincent, 2011). Alternative entropy gradient estimation techniques can also be derived from Stein's identity (Li and Turner, 2018; Shi et al., 2018; Wen et al., 2021).

5 EXPERIMENTS

We empirically validate SCOREBED on a wide selection of common experimental design tasks against several competitive baseline approaches. In each task we train adaptive design policies using SCOREBED and baselines and report upper and lower bounds on the attained EIG of the final policy. Our experiments span tasks with varying properties, including non-Markovian likelihoods and high-dimensional parameter spaces. We present four BED tasks in the following sections and include an additional task in Appendix E.2. Full experimental details are included in Appendix D.

Procedure. In order to ensure a fair comparison, each task has the same fixed computational budget across methods, specified through the allowed *number of likelihood evaluations* (NLE). For each approach, we obtain a final policy network through two distinct procedures. First, we train a singular policy network and report its achieved EIG bounds with standard errors reporting variation over re-runs of a single policy training. Second, we instead consider training P restarts of the policy network and report the EIG bounds of the best-performing policy of the P candidates (estimated on a held-out test set), reporting standard errors across re-runs of the best-of-P procedure. The configuration of each baseline is adjusted in order to train P policies with the same total NLE budget. Our two-stage approach means that training multiple policies can be done cheaply, therefore the score training budget is only modestly reduced to accommodate increased P. On the other hand, approaches which do not amortise the intractabilities of the EIG must reduce their budget per policy proportionally to P. In practice, training P policies allows for architecture searches and hyperparameter tuning to increase performance, but we considered a fixed policy configuration to isolate the quality of each method on a consistent policy training task.

Baselines. We compare SCOREBED to several established BED baselines which cover both contrastive sampling and variational approaches. First, we compare against PCE (Foster et al., 2021). In some cases we also include PCE*, which uses a $10\times$ larger-budget variant of PCE since related works sometimes consider larger M than our original budgets. Second, we compare against two pre-trained policy-based extensions of the variational approaches of Foster et al. (2019), the variational marginal (VarMarg) and variational posterior (VarPost), where we pre-train variational approximations of the intractable marginal and posterior distributions, and then train the policy using these fixed approximations (resulting in upper and lower bounds on the EIG respectively). We emphasise that these approaches do not currently appear in the literature as variational approaches are typically co-trained alongside the policy network. Finally, we compare to the co-trained variational posterior approach of Foster et al. (2020) (JointVarPost), in which the policy is co-trained with the variational approximation by maximising the Barber-Agakov lower bound on the EIG. On the dynamical systems tasks, we also include IO-SMC^2 (Iqbal et al., 2024). VarMarg and VarPost represent two-stage approaches which share computation across multiple policies similarly to SCOREBED, while PCE, JointVarPost and IO-SMC^2 must repeat all computations for each policy. Details of each baseline are included in Appendix D.5. We also tested MLMC (Goda et al., 2022) but found that the wall-clock cost was prohibitive and the results were not competitive, see Appendix E.1.

Table 1: Results for the source location finding task. Every method used a total NLE budget of 6.144×10^9, except for PCE* which used $10\times$ the budget. Results show mean ± one standard error over 4 policy repeats, bold indicates statistically indistinguishable from the best lower bound at 95% confidence.

	$d = 2, K = 2$		$d = 3, K = 10$	
Method	Lower Bound	Upper Bound	Lower Bound	Upper Bound
SCOREBED (P=1)	10.81 ± 0.06	10.95 ± 0.07	$\mathbf{9.45 \pm 0.03}$	9.74 ± 0.05
SCOREBED (P=5)	10.98 ± 0.04	11.12 ± 0.04	$\mathbf{9.37 \pm 0.04}$	9.65 ± 0.07
PCE (P=1)	10.12 ± 0.11	10.16 ± 0.11	8.82 ± 0.02	8.97 ± 0.03
PCE (P=5)	9.61 ± 0.06	9.64 ± 0.06	8.49 ± 0.04	8.61 ± 0.04
PCE* (P=1)	10.83 ± 0.03	10.92 ± 0.04	9.14 ± 0.08	9.33 ± 0.09
JointVarPost (P=1)	$\mathbf{11.52 \pm 0.05}$	11.78 ± 0.06	8.18 ± 0.04	8.44 ± 0.05
JointVarPost (P=5)	11.25 ± 0.06	11.48 ± 0.08	7.86 ± 0.04	8.00 ± 0.04
VarMarg (P=1)	4.51 ± 0.08	4.51 ± 0.08	3.31 ± 0.06	3.31 ± 0.06
VarMarg (P=5)	5.02 ± 0.08	5.03 ± 0.08	3.57 ± 0.02	3.57 ± 0.02
VarPost (P=1)	11.11 ± 0.04	11.23 ± 0.05	7.08 ± 0.21	7.14 ± 0.22
VarPost (P=5)	11.18 ± 0.02	11.31 ± 0.03	7.37 ± 0.10	7.46 ± 0.11

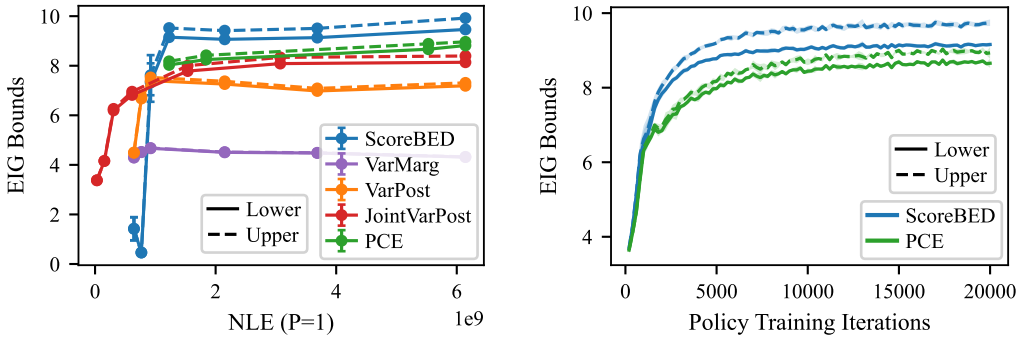

Figure 1: Left: EIG vs NLE ($P = 1$) curves for each method on the $d = 3, K = 10$ location finding setting. Right: Comparing policy training curves for SCOREBED and PCE on the same problem.

The exact NLE cost of each method is calculated in Appendix D.7 and the configurations of each baseline to respect the exact budget are detailed in Appendix D.8. In order to accommodate different numbers of policies P while respecting the same total budget, we varied the upfront training budget for SCOREBED, VarMarg and VarPost, the total training budget for JointVarPost and the number of inner samples and particles for PCE and IO-SMC^2 respectively.

5.1 SOURCE LOCATION FINDING

The first experiment is a long-standing BED benchmark problem (Foster et al., 2021; Sheng and Hu, 2005) where one places designs in a d-dimensional space and makes a noisy observation of the superimposed signals propagating from K unknown sources. We consider the typical $d = 2, K = 2$ setting and a more challenging $d = 3, K = 10$ setting. In each case we perform $T = 30$ experiments and train a permutation-invariant DAD network (Foster et al., 2021). We highlight that in each setting, a single pre-trained score network was used to train *all* the score-based policies reported. Similarly, VarMarg and VarPost use a single pre-trained variational approximation while JointVarPost trains entirely from scratch for each policy.

Results are reported in Table 1. For the $d = 2, K = 2$ setting we observe that each of the two-stage approaches exhibits marginally improved performance at the $P = 5$ protocol, while PCE and the co-trained variational posterior see better performance under $P = 1$. This highlights that pre-training of the EIG intractabilities enables cheap training of multiple policies *without* sacrificing the performance of each individual policy. We emphasise that in practical applications, we expect more significant gains by using these P policies to search over architectures and hyperparameters. Here the overall best-performing approach was JointVarPost, closely followed by VarPost and SCOREBED.

For the $d = 3, K = 10$ setting, Table 1 clearly shows that SCOREBED outperforms all competing methods at both the $P = 1$ and $P = 5$ protocols. The parameter space in this task is 30-dimensional, which we believe particularly suits SCOREBED (and challenges nested or variational posterior approaches) due to the behaviour of the score-matching objective in high parameter dimensions, as discussed in

Table 2: Results for dynamical systems. All tasks have a common NLE budget of 1.024×10^{10}. SCOREBED selects policies across three score network seeds each receiving one third of the budget. The NLE cost of IO-SMC^2 is random due to extra steps following degeneracy criteria; exact values are presented in Appendix Table 11. Results show mean ± one standard error over 4 policy repeats, bold indicates statistically indistinguishable from the best lower bound at 95% confidence.

	Stochastic pendulum		Cart-pole		Double link	
Method	Lower Bound	Upper Bound	Lower Bound	Upper Bound	Lower Bound	Upper Bound
SCOREBED (P=3)	**3.91 ± 0.02**	3.91 ± 0.02	6.94 ± 0.01	6.96 ± 0.00	11.01 ± 0.07	11.59 ± 0.12
SCOREBED (P=50)	**3.95 ± 0.02**	3.95 ± 0.02	**7.17 ± 0.02**	7.19 ± 0.02	11.78 ± 0.01	13.04 ± 0.05
PCE (P=1)	**3.91 ± 0.01**	3.91 ± 0.01	**7.22 ± 0.01**	7.23 ± 0.01	11.57 ± 0.06	12.49 ± 0.15
PCE (P=50)	3.90 ± 0.01	3.90 ± 0.01	7.13 ± 0.03	7.15 ± 0.03	11.15 ± 0.06	11.86 ± 0.10
JointVarPost (P=1)	**3.70 ± 0.16**	3.70 ± 0.16	6.10 ± 0.25	6.10 ± 0.25	11.76 ± 0.08	12.99 ± 0.14
JointVarPost (P=50)	**3.95 ± 0.01**	3.95 ± 0.01	**7.20 ± 0.01**	7.22 ± 0.01	**11.99 ± 0.03**	13.37 ± 0.15
VarMarg (P=1)	3.63 ± 0.02	3.64 ± 0.02	5.50 ± 0.02	5.50 ± 0.02	11.44 ± 0.04	12.33 ± 0.10
VarMarg (P=50)	3.63 ± 0.01	3.63 ± 0.01	5.54 ± 0.00	5.55 ± 0.00	11.44 ± 0.01	12.35 ± 0.04
VarPost (P=1)	3.87 ± 0.01	3.87 ± 0.01	**7.21 ± 0.01**	7.22 ± 0.01	11.79 ± 0.02	12.92 ± 0.06
VarPost (P=50)	**3.92 ± 0.02**	3.92 ± 0.02	**7.19 ± 0.02**	7.20 ± 0.02	11.84 ± 0.02	13.32 ± 0.21
IO-SMC^2 (P=1)	3.49 ± 0.04	3.49 ± 0.04	5.81 ± 0.02	5.81 ± 0.02	11.31 ± 0.04	12.09 ± 0.04

Section 3. Interestingly, SCOREBED performs marginally better when $P = 1$, which indicates that the score network has not yet plateaued at this budget, as corroborated by the EIG vs NLE curves (see below). Surprisingly, we observe that the variational marginal approach does not perform well on either of the location finding settings, despite being theoretically more suited to the high-dimensional setting for the same reasons as SCOREBED.

In Figure 1 we present a curve of EIG bounds versus NLE budget for each method under the $P = 1$ protocol for the higher dimensional setting, which shows SCOREBED achieving consistently higher EIG over a range of NLE budgets. We also visualise a comparison between the EIG curves during policy training for SCOREBED and PCE, where SCOREBED appears to benefit from slightly faster training *per iteration* than PCE, likely due to reducing the variance of the gradient estimates. This translates to significantly faster policy training per NLE since the per-iteration cost of SCOREBED is significantly cheaper in NLE given the pre-trained score network allowing us to avoid the need to consider M contrastive samples.

5.2 DYNAMICAL SYSTEMS

Our second set of experiments is a series of dynamical systems models, introduced as BED benchmarks by Iqbal et al. (2024). Each problem consists of a dynamical system whose behaviour is governed by an SDE depending on a set of unknown parameters and controlled by an external force ξ_t. The design problem is to control the system via ξ_t in order to learn about the unknown parameters. The models are the stochastic pendulum, cart-pole and double link systems, which increase in complexity based on dimensionality and trajectory dynamics. In all cases, each design decision directly impacts the state of the system at future iterations, creating a non-Markovian structure where early design decisions can dramatically change the regime of later observations. We simulate each system for $T = 50$ time steps using an Euler-Maruyama discretisation scheme with step size $\delta = 0.05$. We train an adaptive policy using the stochastic LSTM architecture from Iqbal et al. (2024).

We note for these problems we experienced some instability in the training of the score network. To mitigate this, we trained three score networks each with a third of the allowed budget. We then split the P policy restarts between the three networks and reported the performance of the best policy as per our established best-of-P protocol. We therefore report $P = 3$ and $P = 50$ variants only for our approach. We also only present IO-SMC^2 in the $P = 1$ variant because the wall-clock time for the $P = 50$ protocol was prohibitive.

Stochastic pendulum. This system consists of a pendulum of unknown mass and length controlled by a torque $\xi_t \in [-1, 1]$. Results in Table 2 show statistically indistinguishable performance between SCOREBED and the two variational posterior approaches under the $P = 50$ protocol, while PCE performs similarly as well. We found policy training to be very cheap here—only requiring 500 gradient steps to converge—therefore we were able to cheaply train 50 policy restarts but found only modest improvements in doing so. We expect larger gains could be realised by allowing search over architectures amongst the $P = 50$ policies. Both VarMarg and IO-SMC^2 performed poorly compared to the other approaches. The lower performance of IO-SMC^2 than presented in Iqbal et al. (2024) throughout the experiments is due to our problem setup being slightly different, with a higher observation noise level to avoid EIG saturation and thus more reliable evaluation. We were still able to replicate their results when run on the same noise setting.

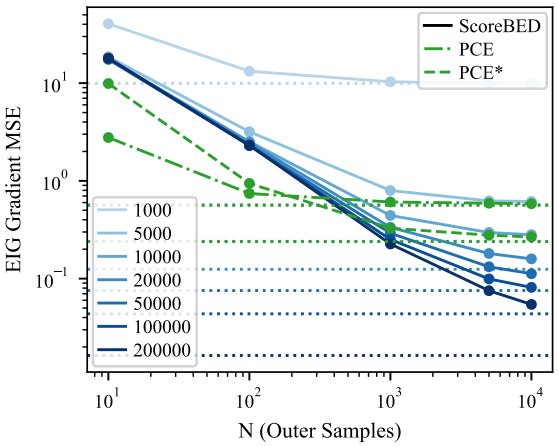

Figure 2: EIG gradient MSE versus N for PCE and SCOREBED. Horizontal lines mark squared-bias asymptotes. PCE* uses $10\times$ the budget of PCE by increasing the number of inner samples by 10, as per Table 1.

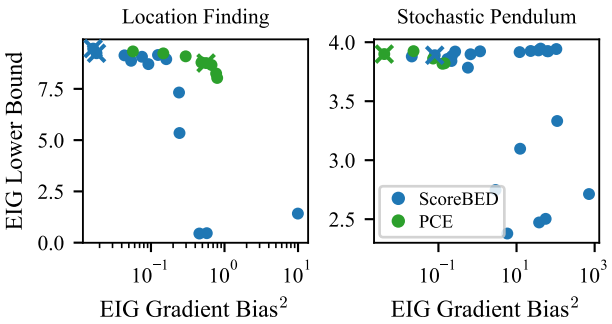

Figure 3: EIG lower bound versus EIG gradient squared-bias. SCOREBED points are generated from different seeds and training checkpoints. PCE points cover a range of M inner samples. 'x' marks the budgets used in our main results.

Cart-pole. The cart-pole system consists of a cart of unknown mass and a freely pivoting pole of unknown mass and length. The design $\xi_t \in [-5, 5]$ is the force applied to the cart, and observations consist of the velocity and acceleration of the cart and the angular position and angular velocity of the pole. The results presented in Table 2 draw similar conclusions to the stochastic pendulum variant with very minor differences between approaches, except for VarMarg and IO-SMC^2, which performed poorly. We note the joint variational posterior approach also experienced some instability on this task and thus benefitted from selecting the best policy from the $P = 50$ protocol.

Double link. The double link system is made up of two links, each of unknown mass and length. The system is controlled by a two-dimensional design with $\xi_t^{(1)} \in [-4, 4]$ being the torque on the first link and $\xi_t^{(2)} \in [-2, 2]$ the torque on the second. The results in Table 2 show that the co-trained variational posterior works best on this task, with the variational posterior and SCOREBED performing similarly and nearly catching up with the co-trained variant when $P = 50$. The best PCE variant here occurs with $P = 1$ and lags slightly behind the aforementioned approaches, but is slightly better than VarMarg or IO-SMC^2.

5.3 POLICY GRADIENT BIAS

To investigate the key factors behind the quality of policy training, we explored the error of the SCOREBED EIG gradient estimates as a function of N and the number of score training iterations on the high-dimensional source location finding task in Figure 2. At low N, we observe the expected $1/N$ convergence rate, which gives way to a bias plateau as N increases. We derive a bias-variance decomposition in Appendix C which explains this behaviour as variance terms in the decomposition are scaled by $1/N$. Increasing N in the gradient estimator eventually leads to the bias dominating the error, and even at modest N—for example we used $N = 1024$ in our experiments—the implicit averaging effect of stochastic gradient descent still reveals the bias as the limiting source of error over policy training. Indeed, we overlaid the error curve for PCE and found significantly higher gradient biases, which matches the inferior performance of PCE on this task (whereas we found this bias to be lower for PCE on tasks where it performed better, cf. Figure 3).

Furthermore, we find a clear correlation between the gradient bias and the downstream EIG achieved across all our experiments, which we illustrate in Figure 3. Higher gradient biases can throw off the policy training and lead to poor performance of the final policy. This reveals the gradient bias as a key determining factor to the success of any policy training methodology. For SCOREBED, bias corresponds to approximation and estimation errors of the score network, which are typically reducible by better training and network capacity, while PCE and nested sampling approaches admit bias through finite sample sizes. This confirms each of these approaches as distinct methods with different properties which may each be advantageous in their own scenarios.

6 CONCLUSION

We turned the doubly intractable EIG gradient in policy-based BED problems into two singly intractable problems which can be solved sequentially: a score matching problem for the intractable marginal followed by policy-training using the learned score network. This SCOREBED approach allows significantly cheaper gradient-based policy training by solving the double intractability upfront, such that we can easily repeatedly train multiple policies on a given problem. This benefits policy training through, for example, allowing architecture and hyperparameter searches and avoiding local optima. Our results illustrate that SCOREBED achieves competitive performance with existing approaches that do not allow cheap policy retraining and with newly proposed alternatives that train a fixed amortised posterior network upfront.

Acknowledgements

AP is supported by the EPSRC CDT in Modern Statistics and Statistical Machine Learning (EP/S023151/1). GK and TR are supported by EPSRC grant EP/Y037200/1.

References

Ziqiao Ao and Jinglai Li. On Estimating the Gradient of the Expected Information Gain in Bayesian Experimental Design. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(18):20311–20319, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i18.30012. Number: 18.

David Barber and Felix Agakov. Information Maximization in Noisy Channels : A Variational Approach. In *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.

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*, pages 2107–2128. PMLR, June 2022.

Niels Bracher, Lars Kühmichel, Desi R. Ivanova, Xavier Intes, Paul-Christian Bürkner, and Stefan T. Radev. JADAI: Jointly Amortizing Adaptive Design and Bayesian Inference, December 2025. arXiv:2512.22999 [stat].

James Bradbury, Roy Frostig, Peter Hawkins, Matthew Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

Johann Brehmer, Gilles Louppe, Juan Pavez, and Kyle Cranmer. Mining gold from implicit models to improve likelihood-free inference. *Proceedings of the National Academy of Sciences*, 117(10):5242–5249, March 2020. doi: 10.1073/pnas.1915980117.

Kathryn Chaloner and Isabella Verdinelli. Bayesian Experimental Design: A Review. *Statistical Science*, 10(3): 273–304, August 1995. ISSN 0883-4237, 2168-8745. doi: 10.1214/ss/1177009939.

Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.

Jiayuan Dong, Christian Jacobsen, Mehdi Khalloufi, Maryam Akram, Wanjiao Liu, Karthik Duraisamy, and Xun Huan. Variational Bayesian optimal experimental design with normalizing flows. *Computer Methods in Applied Mechanics and Engineering*, 433:117457, January 2025. ISSN 00457825. doi: 10.1016/j.cma.2024.117457.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural Spline Flows. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Sergey Dushenko, Kapildeb Ambal, and Robert D. McMichael. Sequential Bayesian Experiment Design for Optically Detected Magnetic Resonance of Nitrogen-Vacancy Centers. *Physical Review Applied*, 14(5):054036, November 2020. doi: 10.1103/PhysRevApplied.14.054036.

Lawrence C Evans. *Measure theory and fine properties of functions*. Chapman and Hall/CRC, 2025.

Adam Foster, Martin Jankowiak, Elias Bingham, Paul Horsfall, Yee Whye Teh, Thomas Rainforth, and Noah Goodman. Variational Bayesian Optimal Experimental Design. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Adam Foster, Martin Jankowiak, Matthew O'Meara, Yee Whye Teh, and Tom Rainforth. A unified stochastic gradient approach to designing bayesian-optimal experiments. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2959–2969. PMLR, 26–28 Aug 2020.

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*, pages 3384–3395. PMLR, July 2021.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1183–1192, Sydney, NSW, Australia, August 2017. JMLR.org.

Takashi Goda, Tomohiko Hironaka, Wataru Kitade, and Adam Foster. Unbiased MLMC Stochastic Gradient-Based Optimization of Bayesian Experimental Designs. *SIAM Journal on Scientific Computing*, 44(1):A286–A311, February 2022. ISSN 1064-8275. doi: 10.1137/20M1338848.

Marcel Hedman, Desi R. Ivanova, Cong Guan, and Tom Rainforth. Step-DAD: Semi-Amortized Policy-Based Bayesian Experimental Design. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 22904–22923. PMLR, October 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, pages 6840–6851, Red Hook, NY, USA, December 2020. Curran Associates Inc. ISBN 978-1-7138-2954-6.

Xun Huan, Jayanth Jagalur, and Youssef Marzouk. Optimal experimental design: Formulations and computations. *Acta Numerica*, 33:715–840, July 2024. ISSN 0962-4929, 1474-0508. doi: 10.1017/S0962492924000023.

Daolang Huang, Zhuoyue Huang, Conor Hassan, Luigi Acerbi, Samuel Kaski, and Tom Rainforth. Efficient adaptive data acquisition via pretrained belief representations. *arXiv preprint arXiv:2606.25197*, 2026a.

Daolang Huang, Xinyi Wen, Ayush Bharti, Samuel Kaski, and Luigi Acerbi. Aline: Joint amortization for bayesian inference and active data acquisition. *Advances in Neural Information Processing Systems*, 38:54068–54102, 2026b.

Aapo Hyvärinen. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. ISSN 1533-7928.

Jacopo Iollo, Christophe Heinkelé, Pierre Alliez, and Florence Forbes. Bayesian Experimental Design via Contrastive Diffusions, March 2025. arXiv:2410.11826 [stat] version: 2.

Sahel Iqbal, Adrien Corenflos, Simo Särkkä, and Hany Abdulsamad. Nesting Particle Filters for Experimental Design in Dynamical Systems. In *Proceedings of the 41st International Conference on Machine Learning*, pages 21047–21068. PMLR, July 2024.

Desi R Ivanova, Adam Foster, Steven Kleinegesse, Michael U. Gutmann, and Thomas Rainforth. Implicit Deep Adaptive Design: Policy-Based Experimental Design without Likelihoods. In *Advances in Neural Information Processing Systems*, volume 34, pages 25785–25798. Curran Associates, Inc., 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. arXiv:1412.6980 [cs].

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2013. arXiv:1312.6114 [stat].

Steven Kleinegesse and Michael U. Gutmann. Efficient Bayesian Experimental Design for Implicit Models. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 476–485. PMLR, April 2019.

Steven Kleinegesse and Michael U. Gutmann. Bayesian Experimental Design for Implicit Models by Mutual Information Neural Estimation. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5316–5326. PMLR, November 2020.

Yingzhen Li and Richard E Turner. Gradient estimators for implicit models. In *International Conference on Learning Representations*, 2018.

Jae Hyun Lim, Aaron Courville, Christopher Pal, and Chin-Wei Huang. AR-DAE: Towards Unbiased Neural Entropy Gradient Estimation. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6061–6071. PMLR, November 2020.

D. V. Lindley. On a Measure of the Information Provided by an Experiment. *The Annals of Mathematical Statistics*, 27 (4):986–1005, December 1956. ISSN 0003-4851, 2168-8990. doi: 10.1214/aoms/1177728069.

Dennis Victor Lindley. *Bayesian statistics: A review*. SIAM, 1972.

Robert D. McMichael, Sergey Dushenko, and Sean M. Blakley. Sequential Bayesian experiment design for adaptive Ramsey sequence measurements. *Journal of Applied Physics*, 130(14):144401, October 2021. ISSN 0021-8979. doi: 10.1063/5.0055630.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16569–16594. PMLR, 17–23 Jul 2022.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Josh Pasek and Jon A. Krosnick. Optimizing Survey Questionnaire Design in Political Science: Insights from Psychology. In Jan E. Leighley, editor, *The Oxford Handbook of American Elections and Political Behavior*, page 0. Oxford University Press, February 2010. ISBN 978-0-19-923547-6. doi: 10.1093/oxfordhb/9780199235476.003.0003.

Tom Rainforth, Rob Cornish, Hongseok Yang, Andrew Warrington, and Frank Wood. On Nesting Monte Carlo Estimators. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4267–4276. PMLR, July 2018.

Tom Rainforth, Adam Foster, Desi R. Ivanova, and Freddie Bickford Smith. Modern Bayesian Experimental Design. *Statistical Science*, 39(1):100–114, February 2024. ISSN 0883-4237, 2168-8745. doi: 10.1214/23-STS915.

Chang-Han Rhee and Peter W. Glynn. Unbiased Estimation with Square Root Convergence for SDE Models. *Operations Research*, 63(5):1026–1043, 2015. ISSN 0030-364X.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, June 2022. doi: 10.1109/CVPR52688.2022.01042.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

Leopoldo Sarra and Florian Marquardt. Deep Bayesian experimental design for quantum many-body systems. *Machine Learning: Science and Technology*, 4(4):045022, October 2023. ISSN 2632-2153. doi: 10.1088/2632-2153/ad020d.

C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27 (3):379–423, 1948. ISSN 1538-7305. doi: 10.1002/j.1538-7305.1948.tb01338.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1538-7305.1948.tb01338.x.

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, November 2023. ISSN 00457825. doi: 10.1016/j.cma.2023.116304. arXiv:2110.15335 [cs].

Xiaohong Sheng and Yu-Hen Hu. Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks. *IEEE Transactions on Signal Processing*, 53(1):44–53, January 2005. ISSN 1941-0476. doi: 10.1109/TSP.2004.838930. Conference Name: IEEE Transactions on Signal Processing.

Jiaxin Shi, Shengyang Sun, and Jun Zhu. A spectral approach to gradient estimation for implicit distributions. In *International Conference on Machine Learning*, pages 4644–4653. PMLR, 2018.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms, August 2012. arXiv:1206.2944 [stat].

Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced Score Matching: A Scalable Approach to Density and Score Estimation, June 2019. arXiv:1905.07088.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. In *Advances in Neural Information Processing Systems*, volume 33, pages 7537–7547. Curran Associates, Inc., 2020.

W. M. Telford, L. P. Geldart, and R. E. Sheriff. *Gravity Methods*, page 6–61. Cambridge University Press, 1990.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Benjamin Thomas Vincent and Tom Rainforth. The DARC Toolbox: automated, flexible, and efficient delayed and risky choice experiments using Bayesian adaptive design, October 2017.

Pascal Vincent. A Connection Between Score Matching and Denoising Autoencoders. *Neural Computation*, 23(7):1661–1674, July 2011. ISSN 0899-7667. doi: 10.1162/NECO_a_00142. Conference Name: Neural Computation.

Andrew B. Watson. QUEST+: A general multidimensional Bayesian adaptive psychometric method. *Journal of Vision*, 17(3):10, April 2017. ISSN 1534-7362. doi: 10.1167/17.3.10.

Liangjian Wen, Haoli Bai, Lirong He, Yiji Zhou, Mingyuan Zhou, and Zenglin Xu. Gradient estimation of information measures in deep learning. *Knowledge-Based Systems*, 224:107046, July 2021. ISSN 09507051. doi: 10.1016/j.knosys.2021.107046.

Ronald J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.*, 8(3-4):229–256, May 1992. ISSN 0885-6125. doi: 10.1007/BF00992696.

Bayesian Experimental Design via Score Matching (Supplementary Material)

Angus Phillips[1] **Gavin Kerrigan**[1] **Tom Rainforth**[1]

[1]Department of Statistics, University of Oxford

INTRODUCTION

The Appendix is structured as follows. In Appendix A we derive the gradient expression of Theorem 1. In Appendix B we prove an error bound on the SCOREBED EIG gradient approximation. In Appendix C we provide the bias–variance decomposition discussed in Section 5.3. In Appendix D we provide full details of our experimental procedures. In Appendix E we provide additional results, ablation studies and further empirical investigations.

A EIG GRADIENT EXPRESSION

A.1 POLICY-BASED TOTAL EIG

Recall the information gain as a measure of the utility (Lindley, 1956) of a set of designs $\xi_{1:T}$ for which you observe data $y_{1:T}$, defined as $\mathrm{IG}_\theta(y_{1:T}, \xi_{1:T}) = \mathbb{H}[p(\theta)] - \mathbb{H}[p(\theta \mid y_{1:T}, \xi_{1:T})]$. Since we wish to optimise the next designs prior to observing any data, we consider the expectation of the information gain with respect to the data generating process.

In a policy-based BED setting, we plan to draw designs from an adaptive proposal distribution $\pi_\phi(\xi_t \mid h_{t-1})$ which we assume to be stochastic in general although we note that optimal policies are always deterministic (Lindley, 1972). In the stochastic case, $\pi_\phi(\xi_t \mid h_{t-1})$ represents a distribution with density of the same notation defined with respect to the Lebesgue measure. If the policy network is instead deterministic, π_ϕ no longer has a density with respect to the Lebesgue measure and instead collapses to a degenerate distribution. In the deterministic case our notation is no longer formally correct but an alternative formulation of the likelihood $p(y_t \mid \theta; \xi_t)$ could be considered where ξ_t is no longer a random variable but a parameter of the likelihood. With that distinction, all of the same definitions and results hold. A more in-depth discussion can be found in the Appendix of Ivanova et al. (2021).

Given the (stochastic) policy network π_ϕ and conditional on the underlying parameter θ, the data generating process can be written as $p_\phi(y_{1:T}, \xi_{1:T} \mid \theta) = \prod_{t=1}^T \pi_\phi(\xi_t \mid h_{t-1}) p(y_t \mid \theta, \xi_t, h_{t-1})$. We integrate this over the prior $p(\theta)$ to obtain the joint $p_\phi(y_{1:T}, \xi_{1:T})$. The posterior distribution is $p(\theta \mid y_{1:T}, \xi_{1:T}) \propto p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta)$ which is easily seen to be independent of how the designs $\xi_{1:T}$ were proposed, as follows:

$$p(\theta \mid y_{1:T}, \xi_{1:T}) = \frac{p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta)}{\int p(\theta) p_\phi(y_{1:T}, \xi_{1:T} \mid \theta) \mathrm{d}\theta} \tag{7}$$

$$= \frac{p(\theta) \prod_{t=1}^T \pi_\phi(\xi_t \mid h_{t-1}) p(y_t \mid \xi_t, h_{t-1}, \theta)}{\int p(\theta) \prod_{t=1}^T \pi_\phi(\xi_t \mid h_{t-1}) p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta} \tag{8}$$

$$= \frac{p(\theta) \prod_{t=1}^T p(y_t \mid \xi_t, h_{t-1}, \theta)}{\int p(\theta) \prod_{t=1}^T p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta} \tag{9}$$

where the design proposals cancelled. This assumes that observations do not depend on the policy in any way beyond the realised designs, that is we have $p(y_t|\xi_t, h_{t-1}, \theta)$ rather than $p(y_t|\xi_t, h_{t-1}, \theta, \phi)$. The ignorability of the observation

mechanism in posterior inference and the conditions required have previously been studied in related literature on missing data, see Rubin (1976).

We also define the marginal distribution of the data given a set of designs, $p(y_{1:T} \mid \xi_{1:T}) = \int p(\theta) \prod_{t=1}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta = \prod_{t=1}^{T} p(y_t \mid \xi_t, h_{t-1})$ where $p(y_t \mid \xi_t, h_{t-1}) = \int p(\theta \mid h_{t-1}) p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta$. This decomposition is axiomatically true by stacking nested conditional densities but we also show below that $p(y_t \mid \xi_t, h_{t-1})$ has a particular form:

$$\int p(\theta) \prod_{t=1}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta \tag{10}$$

$$= \int p(\theta) p(y_1 \mid \xi_1, h_0, \theta) \prod_{t=2}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta \tag{11}$$

$$= \underbrace{\int p(\theta) p(y_1 \mid \xi_1, h_0, \theta) \mathrm{d}\theta}_{p(y_1 \mid \xi_1, h_0)} \int \frac{p(\theta) p(y_1 \mid \xi_1, h_0, \theta)}{\int p(\theta) p(y_1 \mid \xi_1, h_0, \theta) \mathrm{d}\theta} \prod_{t=2}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta \tag{12}$$

$$= p(y_1 \mid \xi_1, h_0) \int p(\theta \mid h_1) \prod_{t=2}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta \tag{13}$$

$$= p(y_1 \mid \xi_1, h_0) \underbrace{\int p(\theta \mid h_1) p(y_2 \mid \xi_2, h_1, \theta) \mathrm{d}\theta}_{p(y_2 \mid \xi_2, h_1)} \int \frac{p(\theta \mid h_1) p(y_2 \mid \xi_2, h_1, \theta)}{\int p(\theta \mid h_1) p(y_2 \mid \xi_2, h_1, \theta) \mathrm{d}\theta} \prod_{t=3}^{T} p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta \tag{14}$$

$$= \cdots \tag{15}$$

$$= \prod_{t=1}^{T} p(y_t \mid \xi_t, h_{t-1}) \tag{16}$$

where $p(y_t \mid \xi_t, h_{t-1}) = \int p(\theta \mid h_{t-1}) p(y_t \mid \xi_t, h_{t-1}, \theta) \mathrm{d}\theta$ and the intermediate steps were completed by induction.

With the above distributions carefully defined, we consider the expected information gain as $\mathcal{I}_T(\phi) = \mathbb{E}_{p_\phi(y_{1:T}, \xi_{1:T})}\{\mathrm{IG}_\theta(y_{1:T}, \xi_{1:T})\}$. Note that the information gain itself does not depend on the policy network, but only on the specific set of designs that are seen. The dependency on the policy network is only introduced in the outer expectation of the EIG.

A.2 DERIVATION OF EIG GRADIENT

We restate our theorem for the gradient of the EIG in full notation below.

Theorem 2 (Gradient of EIG)**.** *The following expression for the gradient of the EIG holds:*

$$\nabla_\phi \mathcal{I}_T(\phi) = \mathbb{E}_{p(\theta) q(\delta_{1:T}, \epsilon_{1:T})} \left\{ \frac{\mathrm{d}}{\mathrm{d}\phi} \log p(y_{1:T} | \xi_{1:T}, \theta) - \frac{\partial}{\partial y_{1:T}} \log p(y_{1:T} | \xi_{1:T}) \frac{\partial y_{1:T}}{\partial \phi} - \frac{\partial}{\partial \xi_{1:T}} \log p(y_{1:T} | \xi_{1:T}) \frac{\partial \xi_{1:T}}{\partial \phi} \right\} \tag{17}$$

where $y_{1:T} = \{g_t^1(\delta_{1:t}, \epsilon_{1:t}, \theta, \phi)\}_{t=1}^T$, $\xi_{1:T} = \{g_t^2(\delta_{1:t-1}, \epsilon_{1:t}, \theta, \phi)\}_{t=1}^T \sim p_\phi(\cdot, \cdot \mid \theta)$ *for* $\delta_{1:T}, \epsilon_{1:T} \sim q(\cdot, \cdot)$ *are obtained by reparameterisation. In the specific case of static (non-adaptive) designs, the expression simplifies to:*

$$\nabla_{\xi_{1:T}} \mathcal{I}_T(\xi_{1:T}) = \mathbb{E}_{p(\theta) q(\delta_{1:T})} \left\{ \frac{\mathrm{d}}{\mathrm{d}\xi_{1:T}} \log p(y_{1:T} | \theta; \xi_{1:T}) - \frac{\partial}{\partial y_{1:T}} \log p(y_{1:T}; \xi_{1:T}) \frac{\partial y_{1:T}}{\partial \xi_{1:T}} \right\} \tag{18}$$

where again we use the reparameterisation trick such that $y_{1:T} = \{g_t(\delta_{1:t}, \theta, \xi_{1:T})\}_{t=1}^T \sim p(\cdot; \xi_{1:T})$ *for* $\delta_{1:T} \sim q(\cdot)$.

Proof. To derive this expression we start by considering the mutual information form of the EIG as follows:

$$\mathcal{I}_T(\phi) = \mathbb{E}_{p_\phi(y_{1:T},\xi_{1:T})}\{\mathrm{IG}_\theta(y_{1:T},\xi_{1:T})\} \tag{19}$$
$$= \mathbb{E}_{p_\phi(y_{1:T},\xi_{1:T})}\{\mathbb{E}_{p(\theta|y_{1:T},\xi_{1:T})}\log p(\theta \mid y_{1:T},\xi_{1:T}) - \mathbb{E}_{p(\theta)}\log p(\theta)\} \tag{20}$$
$$= \mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\{\log p(\theta \mid y_{1:T},\xi_{1:T})\} + C \tag{21}$$
$$= \mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\{\log p(y_{1:T} \mid \xi_{1:T},\theta) - \log p(y_{1:T} \mid \xi_{1:T})\} + C \tag{22}$$

where C denotes terms constant in ϕ. Firstly, we consider the full gradient expression in the general case of stochastic proposals, given in eq. (3). This expression follows by reparameterising, differentiating under the expectation, and applying the chain rule. Explicitly, we define the reparameterisation recursively by $y_t = g_t^1(\delta_t, \xi_t, \theta, h_{t-1})$ (recall $y_t \sim p(\cdot \mid \xi_t, \theta, h_{t-1})$) and $\xi_t = g_t^2(\epsilon_t, h_{t-1}, \phi)$ (recall $\xi_t \sim \pi_\phi(\cdot \mid h_{t-1})$). Rolling out the recursive reparameterisation we obtain[1] $y_{1:T} = g_{1:T}^1(\delta_{1:T}, \epsilon_{1:T}, \theta, \phi)$, $\xi_{1:T} = g_{1:T}^2(\delta_{1:T-1}, \epsilon_{1:T}, \theta, \phi) \sim p_\phi(\cdot, \cdot \mid \theta)$ for $\epsilon_{1:T}, \delta_{1:T} \sim q(\cdot, \cdot)$, as stated in the theorem. Then the gradient of the objective can be written as:

$$\nabla_\phi \mathcal{I}_T(\phi) = \mathbb{E}_{p(\theta)q(\epsilon_{1:T},\delta_{1:T})}\left\{\frac{\mathrm{d}}{\mathrm{d}\phi}\log p(g_{1:T}^1(\dots) \mid g_{1:T}^2(\dots),\theta) - \frac{\mathrm{d}}{\mathrm{d}\phi}\log p(g_{1:T}^1(\dots) \mid g_{1:T}^2(\dots))\right\} \tag{23}$$

$$= \mathbb{E}_{p(\theta)q(\epsilon_{1:T},\delta_{1:T})}\Bigg\{\frac{\mathrm{d}}{\mathrm{d}\phi}\log p(g_{1:T}^1(\dots) \mid g_{1:T}^2(\dots),\theta)$$
$$- \frac{\partial}{\partial y_{1:T}}\log p(y_{1:T} \mid \xi_{1:T})\Big|_{\substack{y_{1:T}=g_{1:T}^1(\delta_{1:T},\epsilon_{1:T},\theta,\phi)\\ \xi_{1:T}=g_{1:T}^2(\delta_{1:T-1},\epsilon_{1:T},\theta,\phi)}} \frac{\partial g_{1:T}^1(\delta_{1:T},\epsilon_{1:T},\theta,\phi)}{\partial\phi}$$
$$- \frac{\partial}{\partial \xi_{1:T}}\log p(y_{1:T} \mid \xi_{1:T})\Big|_{\substack{y_{1:T}=g_{1:T}^1(\delta_{1:T},\epsilon_{1:T},\theta,\phi)\\ \xi_{1:T}=g_{1:T}^2(\delta_{1:T-1},\epsilon_{1:T},\theta,\phi)}} \frac{\partial g_{1:T}^2(\delta_{1:T-1},\epsilon_{1:T},\theta,\phi)}{\partial\phi}\Bigg\}. \tag{24}$$

This is the main gradient expression of the theorem as required.

As stated in the beginning, when the designs are static, they are uniquely determined up front and have no dependence on the data or unknown parameters of the model, and so we can simplify the above expression. Starting again from eq. (22):

$$\mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\{\log p(y_{1:T} \mid \xi_{1:T},\theta) - \log p(y_{1:T} \mid \xi_{1:T})\} + C \tag{25}$$
$$= \mathbb{E}_{p(\theta)p(y_{1:T}|\theta;\xi_{1:T})}\{\log p(y_{1:T} \mid \theta;\xi_{1:T}) - \log p(y_{1:T};\xi_{1:T})\} + C \tag{26}$$

where we have rewritten the marginal distribution of the data given the designs as $p(y_{1:T} \mid \xi_{1:T}) = p(y_{1:T};\xi_{1:T})$ in the static case (to emphasise the deterministic nature of the designs). This is now easy to reparameterise as $y_t = g_t(\delta_t, \theta, h_{t-1})$ which rolls out to $y_{1:T} = \{g_t(\delta_{1:t}, \theta, \xi_{1:t})\}_{t=1}^T$ with $\delta_{1:T} \sim q(\cdot)$. Now differentiating under the expectation and applying the chain rule we obtain:

$$\nabla_{\xi_{1:T}}\mathcal{I}_T(\xi_{1:T}) = \mathbb{E}_{p(\theta)q(\delta_{1:T})}\Bigg\{\frac{\mathrm{d}}{\mathrm{d}\xi_{1:T}}\log p(g_{1:T}(\delta_{1:T},\theta,\xi_{1:T}) \mid \theta;\xi_{1:T})$$
$$- \frac{\partial}{\partial y_{1:T}}\log p(y_{1:T};\xi_{1:T})\Big|_{y_{1:T}=g_{1:T}(\delta_{1:T},\theta,\xi_{1:T})} \frac{\partial g_{1:T}(\delta_{1:T},\theta,\xi_{1:T})}{\partial \xi_{1:T}}$$
$$- \frac{\partial}{\partial \xi_{1:T}}\log p(y_{1:T};\xi_{1:T})\Big|_{y_{1:T}=g_{1:T}(\delta_{1:T},\theta,\xi_{1:T})}\Bigg\}. \tag{27}$$

Now we show that the final term is zero in expectation. To do so we undo the original reparameterisation by writing $\delta_{1:T} = h_{1:T}(y_{1:T},\theta,\xi_{1:T}) \sim q(\cdot)$ for $y_{1:T} \sim p(\cdot;\xi_{1:T})$, where h is the inverse of g, i.e. $y_{1:T} = g_{1:T}(h_{1:T}(y_{1:T},\theta,\xi_{1:T}),\theta,\xi_{1:T})$.

[1] We use shorthand notation $y_{1:T} = g_{1:T}^1(\delta_{1:T},\epsilon_{1:T},\theta,\phi)$ to mean $y_{1:T} = \{g_t^1(\delta_{1:t},\epsilon_{1:t},\theta,\phi)\}_{t=1}^T$.

Then we have:

$$\mathbb{E}_{p(\theta)q(\delta_{1:T})}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T};\xi_{1:T})\Big|_{y_{1:T}=g_{1:T}(\delta_{1:T},\theta,\xi_{1:T})}\right\} \tag{28}$$

$$=\mathbb{E}_{p(\theta)p(y_{1:T}|\theta;\xi_{1:T})}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T};\xi_{1:T})\Big|_{y_{1:T}=g_{1:T}(h_{1:T}(y_{1:T},\theta,\xi_{1:T}),\theta,\xi_{1:T})}\right\} \tag{29}$$

$$=\mathbb{E}_{p(\theta)p(y_{1:T}|\theta;\xi_{1:T})}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T};\xi_{1:T})\right\} \tag{30}$$

$$=\iint\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T};\xi_{1:T})p(\theta)p(y_{1:T}\mid\theta;\xi_{1:T})\mathrm{d}y_{1:T}\mathrm{d}\theta \tag{31}$$

$$=\iint\frac{\frac{\partial}{\partial\xi_{1:T}}p(y_{1:T};\xi_{1:T})}{p(y_{1:T};\xi_{1:T})}p(y_{1:T};\xi_{1:T})p(\theta\mid y_{1:T})\mathrm{d}y_{1:T}\mathrm{d}\theta \tag{32}$$

$$=\iint\frac{\partial}{\partial\xi_{1:T}}p(y_{1:T};\xi_{1:T})p(\theta\mid y_{1:T})\mathrm{d}y_{1:T}\mathrm{d}\theta \tag{33}$$

$$=\int\frac{\partial}{\partial\xi_{1:T}}p(y_{1:T};\xi_{1:T})\int p(\theta\mid y_{1:T})\mathrm{d}\theta\mathrm{d}y_{1:T} \tag{34}$$

$$=\frac{\partial}{\partial\xi_{1:T}}\int p(y_{1:T};\xi_{1:T})\mathrm{d}y_{1:T}=0. \tag{35}$$

So we have removed one intractable term from the gradient expression in the static design case, providing the second part of the result from the theorem. □

One might be tempted to try a similar trick to eliminate the last term in the general case. However, when using policies it turns out that this term is no longer zero in expectation. At a high level, this is because it is necessary to reparameterise the designs, which introduces a dependence on θ which cannot be integrated away as in the penultimate line of the proof above. For completeness, we show this below. Starting with the last term and applying the inverse reparameterisation trick, analogous to the deterministic case with $\delta_{1:T}=h^1_{1:T}(y_{1:T},\xi_{1:T},\theta,\phi),\epsilon_{1:T}=h^2_{1:T}(y_{1:T-1},\xi_{1:T},\theta,\phi)$, we have:

$$\mathbb{E}_{p(\theta)q(\delta_{1:T},\epsilon_{1:T})}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T}\mid\xi_{1:T})\Big|_{\substack{y_{1:T}=g^1_{1:T}(\delta_{1:T},\epsilon_{1:T},\theta,\phi)\\ \xi_{1:T}=g^2_{1:T}(\delta_{1:T-1},\epsilon_{1:T},\theta,\phi)}}\frac{\partial g^2_{1:T}(\delta_{1:T-1},\epsilon_{1:T},\theta,\phi)}{\partial\phi}\right\} \tag{36}$$

$$=\mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_{1:T}\mid\xi_{1:T})\times \underbrace{\frac{\partial g^2_{1:T}(h^1_{1:T}(y_{1:T-1},\xi_{1:T},\theta,\phi),h^2_{1:T}(y_{1:T-1},\xi_{1:T-1},\theta,\phi),\theta,\phi)}{\partial\phi}}_{J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)}\right\} \tag{37}$$

$$=\mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\left\{\frac{\partial}{\partial\xi_{1:T}}\log\prod_{t=1}^{T}p(y_t\mid\xi_t,h_{t-1})J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)\right\} \tag{38}$$

$$=\mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\left\{\sum_{t=1}^{T-1}\frac{\partial}{\partial\xi_{1:T}}\log p(y_t\mid\xi_t,h_{t-1})J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)+ \frac{\partial}{\partial\xi_{1:T}}\log p(y_T\mid\xi_T,h_{T-1})J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)\right\} \tag{39}$$

Looking at the last term:

$$\mathbb{E}_{p(\theta)p_\phi(y_{1:T},\xi_{1:T}|\theta)}\left\{\frac{\partial}{\partial\xi_{1:T}}\log p(y_T\mid\xi_T,h_{T-1})J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)\right\} \tag{40}$$

$$=\iiint\frac{\frac{\partial}{\partial\xi_{1:T}}p(y_T\mid\xi_T,h_{T-1})}{p(y_T\mid\xi_T,h_{T-1})}J_\phi(y_{1:T-1},\xi_{1:T},\theta,\phi)p(\theta\mid y_{1:T},\xi_{1:T}) \prod_{t=1}^{T}p(y_t\mid\xi_t,h_{t-1})\pi_\phi(\xi_t\mid h_{t-1})\mathrm{d}y_{1:T}\mathrm{d}\xi_{1:T}\mathrm{d}\theta \tag{41}$$

$$= \iiint \frac{\partial}{\partial \xi_{1:T}} p(y_T \mid \xi_T, h_{T-1}) J_\phi(y_{1:T-1}, \xi_{1:T}, \theta, \phi) p(\theta \mid y_{1:T}, \xi_{1:T}) \pi_\phi(\xi_T \mid h_{T-1}) \prod_{t=1}^{T-1} p(y_t \mid \xi_t, h_{t-1}) \pi_\phi(\xi_t \mid h_{t-1}) \mathrm{d}y_{1:T} \mathrm{d}\xi_{1:T} \mathrm{d}\theta. \tag{42}$$

The problem now is that we cannot eliminate the dependence on y_T from the posterior by integrating over θ since the Jacobian term, which arose from the reparameterisation of the designs, depends on θ.

A.3 CONSTANT ADDITIVE NOISE MODELS

Our gradient expression can be further simplified when the model is known to have constant additive noise, i.e. $y = f(\xi, \theta) + \varepsilon$ where ε is a realisation of an independent noise variable. In this case, $p(y \mid \theta, \xi) = p_\varepsilon(y - f(\xi, \theta))$ so:

$$\begin{aligned}
\mathbb{H}[p(y \mid \theta, \xi)] &= \mathbb{E}_{p(y|\theta,\xi)} \left\{ \log p(y \mid \theta, \xi) \right\} \\
&= \int \log p\left(y \mid \theta, \xi\right) p\left(y \mid \theta, \xi\right) \mathrm{d}y \\
&= \int \log p_\varepsilon\left(y - f(\xi, \theta)\right) p_\varepsilon\left(y - f(\xi, \theta)\right) \mathrm{d}y \\
&= \int \log p_\varepsilon(z) p_\varepsilon(z) \mathrm{d}z \\
&= \mathbb{H}[p_\varepsilon(\varepsilon)] \equiv \text{constant}.
\end{aligned}$$

Therefore, in such cases the first term of our EIG gradient expression (eq. (3)) is also identically zero and the gradient of the EIG only depends on the intractable score terms which we propose to approximate.

B ERROR BOUNDS

In this section we show that the accuracy of our score-based EIG gradient estimator is directly controlled by the accuracy of the score approximation, in the sense that the ℓ_2 error in the gradient estimator is bounded above by a constant times the mean score error. The constant depends only on the Lipschitz properties of the map from policy parameters to rolled-out trajectories $(y_{1:T}, \xi_{1:T})$.

Set-up. Let $z_\phi := (y_{1:T}, \xi_{1:T}) \in \mathbb{R}^{T(d_y + d_\xi)}$ denote the rolled-out trajectory under the policy with parameters ϕ, viewed as a function of the reparameterisation variables $(\delta_{1:T}, \epsilon_{1:T})$, the latent θ, and ϕ:

$$z_\phi(\delta_{1:T}, \epsilon_{1:T}, \theta) = \left(g^1_{1:T}(\delta_{1:T}, \epsilon_{1:T}, \theta, \phi),\ g^2_{1:T}(\delta_{1:T-1}, \epsilon_{1:T}, \theta, \phi)\right). \tag{43}$$

Let $s_\psi : \mathbb{R}^{T(d_y+d_\xi)} \to \mathbb{R}^{T(d_y+d_\xi)}$ be a score approximation and define the corresponding score-based gradient estimator by replacing the marginal score in Equation (17) with s_ψ:

$$\widehat{G}_\psi(\phi) = \mathbb{E}_{p(\theta)q(\delta_{1:T}, \epsilon_{1:T})} \left[\frac{\mathrm{d}}{\mathrm{d}\phi} \log p(y_{1:T} \mid \xi_{1:T}, \theta) - s_\psi(y_{1:T}, \xi_{1:T}) \frac{\partial z_\phi}{\partial \phi} \right]. \tag{44}$$

Theorem 3. *Fix a tolerance $\eta > 0$. Assume that, for all $(\delta_{1:T}, \epsilon_{1:T}, \theta)$ in the support of $p(\theta)q(\delta_{1:T}, \epsilon_{1:T})$, the reparameterisation map $\phi \mapsto z_\phi(\delta_{1:T}, \epsilon_{1:T}, \theta)$ is uniformly L-Lipschitz, i.e. there exists $L > 0$ with*

$$\left\| z_\phi(\delta_{1:T}, \epsilon_{1:T}, \theta) - z_{\phi'}(\delta_{1:T}, \epsilon_{1:T}, \theta) \right\|_2 \leq L \|\phi - \phi'\|_2 \quad \textit{for all } \phi, \phi'. \tag{45}$$

Suppose further that the score approximation satisfies the mean error bound

$$\mathbb{E}_{p,q}\left[\left\| s_\psi(y_{1:T}, \xi_{1:T}) - \nabla_{(y_{1:T}, \xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T}) \right\|_2 \right] \leq \eta / L. \tag{46}$$

Then the ℓ_2 error of the score-based gradient estimator is bounded by η for every ϕ:

$$\left\| \nabla_\phi \mathcal{I}_T(\phi) - \widehat{G}_\psi(\phi) \right\|_2 \leq \eta. \tag{47}$$

Proof. Subtracting the approximate gradient $\widehat{G}_\psi$ in Equation (44) from the EIG gradient expression Equation (17) gives

$$\nabla_\phi \mathcal{I}_T(\phi) - \widehat{G}_\psi(\phi) = \mathbb{E}_{p,q}\left[\left(s_\psi(y_{1:T}, \xi_{1:T}) - \nabla_{(y_{1:T},\xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T})\right) \frac{\partial z_\phi}{\partial \phi}\right]. \tag{48}$$

Taking ℓ_2 norms, by Jensen's inequality and submultiplicativity of the operator norm,

$$\left\|\nabla_\phi \mathcal{I}_T(\phi) - \widehat{G}_\psi(\phi)\right\|_2 \leq \mathbb{E}_{p,q}\left[\left\|\left(s_\psi(y_{1:T}, \xi_{1:T}) - \nabla_{(y_{1:T},\xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T})\right) \frac{\partial z_\phi}{\partial \phi}\right\|_2\right] \tag{49}$$

$$\leq \mathbb{E}_{p,q}\left[\left\|s_\psi(y_{1:T}, \xi_{1:T}) - \nabla_{(y_{1:T},\xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T})\right\|_2 \cdot \left\|\frac{\partial z_\phi}{\partial \phi}\right\|_{\mathrm{op}}\right]. \tag{50}$$

Since $\phi \mapsto z_\phi$ is L-Lipschitz almost surely, Rademacher's theorem (Evans, 2025) gives $\|\partial z_\phi / \partial \phi\|_{\mathrm{op}} \leq L$ almost surely. Pulling this constant out of the expectation gives

$$\left\|\nabla_\phi \mathcal{I}_T(\phi) - \widehat{G}_\psi(\phi)\right\|_2 \leq L \cdot \mathbb{E}_{p,q}\left[\left\|s_\psi(y_{1:T}, \xi_{1:T}) - \nabla_{(y_{1:T},\xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T})\right\|_2\right] \leq \eta, \tag{51}$$

where the final inequality uses the assumed score-error bound Equation (46). □

Theorem 3 confirms the intuition motivating our approach: the EIG-gradient error is directly controlled by the score-matching error. Reducing the mean score error by a factor k reduces the gradient error by the same factor (up to the Lipschitz constant L of the reparameterisation, which is independent of the score network). Since score-matching training has access to unlimited samples from the model, achieving small mean score error is in principle only a question of network capacity and training budget.

Remark (component-separated bound). A slightly sharper bound is available when the y- and ξ-reparameterisation maps have different Lipschitz constants. If $\phi \mapsto y_{1:T}$ and $\phi \mapsto \xi_{1:T}$ are uniformly Lipschitz with constants L_y and L_ξ respectively, then by the same argument applied to each score component separately,

$$\left\|\nabla_\phi \mathcal{I}_T(\phi) - \widehat{G}_\psi(\phi)\right\|_2 \leq L_y \cdot \mathbb{E}_{p,q}\|e_y\|_2 + L_\xi \cdot \mathbb{E}_{p,q}\|e_\xi\|_2, \tag{52}$$

where e_y and e_ξ are the y- and ξ-component errors of the score network. This refinement is useful when one of the two reparameterisation maps is significantly more regular than the other.

C BIAS-VARIANCE DECOMPOSITIONS

We now decompose errors in the score network and resulting EIG gradient approximation into interpretable bias and variance components, revealing the behaviour of our EIG gradient approximation.

Notation. For notational simplicity we drop the t component of our data and design variables and we write $s(\xi, y)$ for the score network, $\mu(\xi, y) = \nabla_{y,\xi} \log p(y|\xi)$ for the true marginal score. We define $\Sigma_\nu(\xi, y) := \mathrm{Cov}_{\theta|\xi,y}(\nabla \log p(y|\xi, \theta))$ as the posterior covariance of the conditional score.

C.1 SCORE ERROR

Firstly we consider a bias-variance decomposition of the score matching loss. For the MSM loss (Equation (6)), adding and subtracting μ inside the loss gives

$$\mathcal{L} = \mathbb{E}_{\xi,y}\mathbb{E}_{\theta|\xi,y}\|s(\xi, y) - \nabla \log p(y|\xi, \theta)\|^2 \tag{53}$$

$$= \mathbb{E}_{\xi,y}\|s(\xi, y) - \mu(\xi, y)\|^2 + \mathbb{E}_{\xi,y} \operatorname{Tr} \Sigma_\nu(\xi, y), \tag{54}$$

where we used $\mathbb{E}_{\theta|\xi,y}[\nabla \log p(y|\xi, \theta)] = \mu$ to eliminate the cross term. In this expression the second term isolates the irreducible target variance $\mathbb{E}_{\xi,y} \operatorname{Tr} \Sigma_\nu(\xi, y)$ from the learnable score network error in the first term. Applying the standard bias–variance identity $\mathbb{E}\|e\|^2 = \|\mathbb{E}e\|^2 + \operatorname{Tr} \mathrm{Cov}(e)$ to the first term with error $e = s - \mu$ gives

$$\mathcal{L} = \underbrace{\|B\|^2}_{\text{bias}} + \underbrace{V_e}_{\text{error variance}} + \underbrace{V_\nu}_{\text{target noise}}, \tag{55}$$

with $B := \mathbb{E}_{\xi,y}[s-\mu]$, $V_e := \operatorname{Tr}\operatorname{Cov}_{\xi,y}(s-\mu)$ and $V_\nu := \mathbb{E}_{\xi,y}\operatorname{Tr}\Sigma_\nu(\xi,y)$. Empirically, in our experiments we found the dominant term in this decomposition to be the error variance V_e while the irreducible target variance V_ν was insignificant in the overall loss.

C.2 EIG GRADIENT ERROR

We now consider analogous decompositions for the EIG gradient error of SCOREBED. For ease of notation we consider gradients with respect to a single static design ξ^*, with the policy gradient versions coming from a single pushback through the policy network Jacobian.

Recalling the form of the SCOREBED EIG gradient approximation in Equation (3), the estimate is formed by an expectation over $y \sim p(\cdot|\xi^*)$ of the per-sample integrand which we denote by:

$$g_s(\xi^*, y) = s^\xi(\xi^*, y) + J(\xi^*, y)^\top s^y(\xi^*, y), \tag{56}$$

where s^ξ, s^y are the design- and data-score outputs of the network and $J(\xi^*, y) = \partial y/\partial \xi^*$ is the reparameterisation Jacobian. We write g_μ for the same integrand built with the true score μ. The SCOREBED gradient approximation is then realised by an N-sample Monte Carlo estimate of the expectation which we denote

$$\hat{\mathcal{G}}_N(\xi^*) := \frac{1}{N}\sum_{i=1}^{N} g_s(\xi^*, y_i), \tag{57}$$

while the true gradient is denoted $\mathcal{G}(\xi^*) = \mathbb{E}_{y|\xi^*}[g_\mu(\xi^*, y)]$.

We now apply the standard bias–variance decomposition to the MSE of the SCOREBED EIG gradient approximation. Since the target $\mathcal{G}(\xi^*)$ is deterministic, the identity $\mathbb{E}\|Z-c\|^2 = \|\mathbb{E}Z - c\|^2 + \operatorname{Tr}\operatorname{Cov}(Z)$ applied to $Z = \hat{\mathcal{G}}_N(\xi^*)$ gives

$$\mathrm{MSE}_N(\xi^*) := \mathbb{E}_{y_{1:N}|\xi^*}\left\|\hat{\mathcal{G}}_N(\xi^*) - \mathcal{G}(\xi^*)\right\|^2 = \underbrace{\left\|\mathbb{E}_{y_{1:N}|\xi^*}[\hat{\mathcal{G}}_N(\xi^*)] - \mathcal{G}(\xi^*)\right\|^2}_{\text{squared bias}} + \underbrace{\operatorname{Tr}\operatorname{Cov}_{y_{1:N}|\xi^*}(\hat{\mathcal{G}}_N(\xi^*))}_{\text{variance}}. \tag{58}$$

For the bias, linearity of the sample average gives $\mathbb{E}_{y_{1:N}|\xi^*}[\hat{\mathcal{G}}_N(\xi^*)] = \mathbb{E}_{y|\xi^*}[g_s]$, so

$$\mathbb{E}_{y_{1:N}|\xi^*}[\hat{\mathcal{G}}_N(\xi^*)] - \mathcal{G}(\xi^*) = \mathbb{E}_{y|\xi^*}[g_s - g_\mu] =: B(\xi^*), \tag{59}$$

which is independent of N. For the variance, the outer samples $y_i \sim p(\cdot|\xi^*)$ are i.i.d., so the covariance of their average is $1/N$ times the single-sample covariance,

$$\operatorname{Tr}\operatorname{Cov}_{y_{1:N}|\xi^*}\left(\tfrac{1}{N}\textstyle\sum_{i=1}^{N} g_s(\xi^*, y_i)\right) = \tfrac{1}{N}\operatorname{Tr}\operatorname{Cov}_{y|\xi^*}(g_s) =: \tfrac{1}{N}V_s(\xi^*), \tag{60}$$

where $V_s(\xi^*)$ is the per-sample variance of the score integrand. Combining the two terms,

$$\boxed{\mathrm{MSE}_N(\xi^*) = \|B(\xi^*)\|^2 + \tfrac{1}{N}V_s(\xi^*).} \tag{61}$$

Overall, we observe the variance contributions to the EIG gradient error are scaled down by $\frac{1}{N}$, leaving the *gradient bias as the dominant term* which determines the quality of the policy training.

Relating B to $B(\xi^*)$. It remains to show how the bias $B = \mathbb{E}_{\xi,y}[s-\mu]$ in the score matching loss decomposition relates to the EIG gradient bias $B(\xi^*) = \mathbb{E}_{y|\xi^*}[g_s - g_\mu]$, and indeed whether reducing B corresponds to a meaningful reduction in $B(\xi^*)$.

To do so, we observe that B is an *unconditional* mean of the score error over the joint distribution of (ξ, y), whereas $B(\xi^*)$ is a *conditional* mean of the integrand error at a fixed design. We therefore introduce the design-conditional score bias:

$$b(\xi) := \mathbb{E}_{y|\xi}[s(\xi, y) - \mu(\xi, y)], \tag{62}$$

which recovers $B = \mathbb{E}_\xi[b(\xi)]$ by the tower property. This design-conditional bias offers a conditional version of the bias–variance decomposition of the MSM loss:

$$\mathbb{E}_{\xi,y}\|s(\xi,y) - \mu(\xi,y)\|^2 = \mathbb{E}_\xi \mathbb{E}_{y|\xi}\|s(\xi,y) - \mu(\xi,y)\|^2 \tag{63}$$

$$= \mathbb{E}_\xi[\|b(\xi)\|^2 + \operatorname{Tr}\operatorname{Cov}_{y|\xi}(s(\xi,y) - \mu(\xi,y))] \tag{64}$$

$$= \underbrace{\mathbb{E}_\xi\|b(\xi)\|^2}_{\text{design-averaged squared bias}} + V_e^{\text{within}}, \tag{65}$$

where $V_e^{\text{within}} = \mathbb{E}_\xi[\operatorname{Tr}\operatorname{Cov}_{y|\xi}(s(\xi,y) - \mu(\xi,y))]$, which gives the MSM loss decomposition:

$$\mathcal{L} = \underbrace{\mathbb{E}_\xi\|b(\xi)\|^2}_{\text{design-averaged squared bias}} + V_e^{\text{within}} + V_\nu. \tag{66}$$

This confirms that the loss penalises $\mathbb{E}_\xi\|b(\xi)\|^2$, the mean *squared* conditional bias — not merely $\|B\|^2 = \|\mathbb{E}_\xi b(\xi)\|^2$. In particular, conditional biases of opposite sign at different designs cancel in B but not in $\mathbb{E}_\xi\|b(\xi)\|^2$.

It remains to connect the conditional score bias $b(\xi)$ to the integrand bias $B(\xi^*)$. Splitting the integrand error into its design and data components,

$$g_s - g_\mu = (s^\xi - \mu^\xi) + J(\xi^*, y)^\top (s^y - \mu^y), \tag{67}$$

the design component of $B(\xi^*)$ is exactly the design block of the conditional bias $b(\xi^*)$, so the argument above transfers directly: reducing the loss drives down the design-averaged squared design bias. The data component is instead the *Jacobian-weighted* conditional mean $\mathbb{E}_{y|\xi^*}[J^\top(s^y - \mu^y)]$, which the (unweighted, J-agnostic) loss controls only through a Cauchy–Schwarz bound

$$\left\|\mathbb{E}_{y|\xi^*}[J^\top(s^y - \mu^y)]\right\|^2 \le \left(\mathbb{E}_{y|\xi^*}\|J\|_{\text{op}}^2\right)\left(\mathbb{E}_{y|\xi^*}\|s^y - \mu^y\|^2\right), \tag{68}$$

with a constant set by the reparameterisation geometry. This directly bounds the data-component contribution to $\|B(\xi^*)\|^2$ by the per-design score MSE $\mathbb{E}_{y|\xi^*}\|s^y - \mu^y\|^2$ which the loss controls. This is the same mechanism which is identified in our error bound proof of Appendix B.

In summary, we confirm through the illustrative bias-variance decompositions presented above that the score matching loss penalises the score network in such a way that reduces the bias in the error of the EIG gradient estimate, either directly in the case of the design score term or indirectly via the Cauchy–Schwarz inequality for the data score term.

Penalising design-conditional bias. We briefly highlight two score identities which allow one to directly penalise the design-conditional bias of the score network during training. For $y \in \mathbb{R}^d$, assuming boundary terms vanish, the divergence theorem gives

$$\mathbb{E}_{y|\xi}[\nabla_y \log p(y|\xi)] = \int \nabla_y p(y|\xi)\,\mathrm{d}y = 0, \tag{69}$$

and for the design component

$$\mathbb{E}_{y|\xi}[\nabla_\xi \log p(y|\xi)] = \nabla_\xi \int p(y|\xi)\,\mathrm{d}y = 0. \tag{70}$$

Thus the true marginal score integrates to zero over $y|\xi$, and at fixed ξ the bias derives from the network alone: $\|b(\xi)\|^2 = \|\mathbb{E}_{y|\xi}s(\xi,y)\|^2$. Given samples $\{y_i\} \sim p(\cdot|\xi)$, one could estimate and penalise this bias as an addition to the MSM loss. In particular, the naive plug-in $\|\frac{1}{M}\sum_i s(\xi,y_i)\|^2$ is itself biased upward by $\frac{1}{M}\operatorname{Tr}\operatorname{Cov}_{y|\xi}(s)$ but an unbiased estimate could be obtained via the U-statistic

$$U(\xi) = \frac{1}{M(M-1)}\sum_{i\neq j} s(\xi,y_i)^\top s(\xi,y_j). \tag{71}$$

This could be included in the score network training objective to provide additional signal on the critical design-conditional bias, while leaving the optimum unchanged.

D EXPERIMENTAL DETAILS

Here we detail the experimental set-up used to produce the results in Section 5. Our approach is implemented in Jax (Bradbury et al., 2018) and all training was performed on a single NVIDIA Hopper H100 GPU. We benchmark against

several baselines: PCE, an unbiased Monte Carlo approach MLMC (Goda et al., 2022), pre-trained variational marginal and posterior approaches (Foster et al., 2019), a joint variational posterior (Foster et al., 2020) trained on the Barber-Agakov lower bound, and finally IO-SMC^2 (Iqbal et al., 2024). We implemented the PCE, MLMC, and all variational baselines ourselves and used the authors' own implementation (Iqbal et al., 2024) of IO-SMC^2. Our results tables report NMC/PCE bounds on EIG performance of the final policies, with error bounds representing standard error with respect to randomness in the selected policies. We also explore the stability of the score network training in Appendix E.4.

This section is organised as follows. In Appendix D.1 we describe the generative model for each of our experimental tasks. In Appendix D.3 we detail the policy networks used and their training parameters. In Appendix D.4 we give details of the score matching approach followed by details of each baseline in Appendix D.5. In Appendix D.6 we provide details of the PCE/NMC bounds and the sample sizes used for evaluation. Finally, in Appendix D.8 we calculate the NLE cost of each method and provide the exact parameters used to obtain the compute-normalised comparisons in our main results.

Our code is available here: `https://github.com/angusphillips/Score_Matching_BED`

D.1 BED TASKS

D.1.1 Source location finding

Model definition. For the source location finding task we use the exact set-up of Foster et al. (2021). The model consists of a Gaussian prior on K hidden sources in $\mathbb{R}^d$, $\theta_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_d)$ for $k = 1, \dots, K$, and a log-normal observation model

$$\log y \mid \theta, \xi \sim \mathcal{N}\big(\log \mu(\theta, \xi),\, \sigma^2\big), \tag{72}$$

where the mean function is a superposition of decaying signals from each source,

$$\mu(\theta, \xi) = b + \sum_{k=1}^{K} \frac{\alpha_k}{m + \|\theta_k - \xi\|^2}. \tag{73}$$

We use $b = 10^{-1}$, $m = 10^{-4}$, $\alpha_k = 1$, $\sigma = 0.5$. In the standard location finding task we consider $d = 2$, $K = 2$ and $T = 30$, while the high-dimensional setting considers $d = 3$, $K = 10$ and $T = 30$. We therefore have $y_t \in \mathbb{R}$ and $\xi_t \in \mathbb{R}^2$ or $\xi_t \in \mathbb{R}^3$ respectively.

D.1.2 Dynamical systems

The three dynamical-systems tasks follow the experimental set-up of Iqbal et al. (2024) but, for completeness, we restate the generative models in full here. In each case the latent parameters are physical constants with a log-normal prior $\log \theta \sim \mathcal{N}(0,\, 0.01\, I_{d_\theta})$ and the data $y_t \in \mathbb{R}^{d_y}$ is generated by an Euler–Maruyama discretisation of the corresponding controlled SDE with step size $\delta t = 0.05$,

$$y_t = y_{t-1} + \delta t \cdot f(y_{t-1}, \xi_t, \theta) + \sqrt{\delta t}\, L\, \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I_{d_y}), \tag{74}$$

where f is the system-specific drift and L is a diagonal diffusion matrix. We use $T = 50$ for all three systems and initialise all systems with $y_0 = 0$. The likelihood $p(y_t \mid y_{t-1}, \xi_t, \theta) = \mathcal{N}(y_t;\, y_{t-1} + \delta t \cdot f(y_{t-1}, \xi_t, \theta),\, \delta t\, LL^\top)$ is Markovian (i.e. *not* i.i.d. across t).

Stochastic pendulum. The state $y = (q, \dot{q}) \in \mathbb{R}^2$ contains the angle q and angular velocity $\dot{q}$ of a damped pendulum driven by a scalar control torque $\xi \in [-1, 1]$. The latent parameters $\theta = (m, l)$ are the mass and length, both with log-normal priors. The drift in Equation (74) is

$$f(y, \xi, \theta) = \begin{pmatrix} \dot{q} \\ -\dfrac{3\,g}{2\,l}\sin(q) + \dfrac{3\,(\xi - b\,\dot{q})}{m\,l^2} \end{pmatrix}, \tag{75}$$

where $g = 9.81$ is gravitational acceleration and $b = 0.1$ is a damping coefficient.[2] We use the diffusion matrix $L = \mathrm{diag}(0.1, 0.1)$. Designs are constrained to $[-1, 1]$ by clipping, which is done via a tanh output activation on the final

[2]There is an inconsistency between the equations stated in Iqbal et al. (2024) and their public implementation: the stated drift omits the factor of 3 on the input/damping term, while their implementation includes it. To remain consistent with the results in their paper we follow the implementation.

stochastic output of the policy network.

Cart-pole. The state $y = (s, q, \dot{s}, \dot{q}) \in \mathbb{R}^4$ contains the cart position s, pole angle q and their velocities. The latent parameters $\theta = (l, m_p, m_c)$ are the pole length, pole mass and cart mass, all with log-normal priors. A scalar control force $\xi \in [-5, 5]$ is applied to the cart. The drift in Equation (74) is

$$f(y, \xi, \theta) = (\dot{s}, \dot{q}, \ddot{s}, \ddot{q})^T, \tag{76}$$

$$\ddot{s} = \frac{\xi + m_p \sin(q)\big(l\dot{q}^2 + g\cos(q)\big) - (k_1 s + d_1 \dot{s}) - (k_2 q + d_2 \dot{q})\,\cos(q)/l}{m_c + m_p \sin^2(q)}, \tag{77}$$

$$\ddot{q} = \frac{-\xi\cos(q) - m_p\, l\, \dot{q}^2 \cos(q)\sin(q) - (m_c + m_p)\, g\, \sin(q) - (k_1 s + d_1 \dot{s})\cos(q) - (k_2 q + d_2 \dot{q})\cos^2(q)/l}{l\,(m_c + m_p \sin^2(q))}, \tag{78}$$

with stiffness/damping constants $(k_1, k_2, d_1, d_2) = (0.01, 0.01, 0.01, 0.01)$ and $g = 9.81$. We use the diffusion matrix $L = \mathrm{diag}(0.1, 0.1, 0.1, 0.1)$. Again designs are constrained to $[-5, 5]$ with a tanh output activation on the policy network.

Double link. The state $y = (q_1, q_2, \dot{q}_1, \dot{q}_2) \in \mathbb{R}^4$ contains the angles and angular velocities of the two links. The latent parameters $\theta = (m_1, m_2, l_1, l_2)$ are the masses and lengths of the two links, all with log-normal priors. A 2-dimensional control torque $\xi \in [-4, 4] \times [-2, 2]$ is applied. The Euler-Lagrange equations give the constrained dynamics

$$M(q_2; \theta)\begin{pmatrix}\ddot{q}_1 \\ \ddot{q}_2\end{pmatrix} + C(q_2, \dot{q}_1, \dot{q}_2; \theta)\begin{pmatrix}\dot{q}_1 \\ \dot{q}_2\end{pmatrix} + \tau(q_1, q_2; \theta) = \xi, \tag{79}$$

with mass matrix

$$M(q_2; \theta) = \begin{pmatrix}(m_1 + m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 l_2 \cos(q_2) & m_2 l_2^2 + m_2 l_1 l_2 \cos(q_2) \\ m_2 l_2^2 + m_2 l_1 l_2 \cos(q_2) & m_2 l_2^2\end{pmatrix}, \tag{80}$$

coriolis/centrifugal matrix

$$C(q_2, \dot{q}_1, \dot{q}_2; \theta) = \begin{pmatrix}0 & -m_2 l_1 l_2 \sin(q_2)\,(2\dot{q}_1 + \dot{q}_2) \\ \frac{1}{2} m_2 l_1 l_2 (2\dot{q}_1 + \dot{q}_2)\sin(q_2) & -\frac{1}{2} m_2 l_1 l_2 \dot{q}_1 \sin(q_2)\end{pmatrix}, \tag{81}$$

and gravity term

$$\tau(q_1, q_2; \theta) = -g\begin{pmatrix}(m_1 + m_2) l_1 \sin(q_1) + m_2 l_2 \sin(q_1 + q_2) \\ m_2 l_2 \sin(q_1 + q_2)\end{pmatrix}. \tag{82}$$

The drift in Equation (74) is then $f(y, \xi, \theta) = (\dot{q}_1,\, \dot{q}_2,\, M^{-1}(\xi - C\dot{q} - \tau))$. We use the diffusion matrix $L = \mathrm{diag}(0.1, 0.1, 0.1, 0.1)$ and $g = 9.81$. Again designs are constrained with a tanh output activation on the final output of the policy network.

D.2 GRAVIMETRY

Model definition. The gravimetry experiments are inspired by the gravity methods of Telford et al. (1990) for detecting an underground void or well from surface measurements of the local gravitational anomaly. A buried spherical mass anomaly of source strength κ at horizontal position x_0 and depth z produces, at a surface measurement location ξ_t, a vertical gravity perturbation (in μGal, with lengths in metres)

$$g(\xi_t; \theta) = \kappa\, \frac{z}{\big((\xi_t - x_0)^2 + z^2\big)^{3/2}}. \tag{83}$$

The unknown parameter is $\theta = (x_0, z, \kappa) \in \mathbb{R} \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\leq 0}$. The source strength κ (in $\mu\mathrm{Gal} \cdot \mathrm{m}^2$) is negative for a mass deficit such as a void, and relates to the physical radius R and density contrast $\Delta\rho$ of the anomaly through

$$\kappa = \frac{G \cdot \frac{4}{3}\pi R^3\, \Delta\rho}{10^{-8}}, \tag{84}$$

i.e. the mass-anomaly term $G\,\Delta M$ rescaled from $\mathrm{m\,s^{-2}}$ to μGal with $1\ \mu\mathrm{Gal} = 10^{-8}\ \mathrm{m\,s^{-2}}$. We fix $G = 6.674 \times 10^{-11}\ \mathrm{m^3\,kg^{-1}\,s^{-2}}$ and a density contrast $\Delta\rho = -2200\ \mathrm{kg\,m^{-3}}$ (typical of an air-filled void in rock). Only the product

$R^3\Delta\rho$ (equivalently κ) is identifiable, so we fix $\Delta\rho$ and never infer R and $\Delta\rho$ jointly. The observations are i.i.d. additive Gaussian around this nonlinear mean,

$$y_i \mid \theta, \xi_i \sim \mathcal{N}\big(g(\xi_i;\theta),\, \sigma^2\big), \qquad p(y_{1:T} \mid \theta, \xi) = \prod_{i=1}^{N} \mathcal{N}\big(y_i;\, g(\xi_i;\theta),\, \sigma^2\big), \tag{85}$$

with observation noise scale $\sigma = 1$ μGal.

The prior on the latent parameters $\theta = (x_0, z, \kappa)$ is

$$x_0 \sim \mathcal{N}(0,\, \tau^2), \qquad \log z \sim \mathcal{N}(\ln 10,\, 0.5^2)\ \ (z \geq 3), \qquad R \sim \mathcal{U}(1.5,\, 4)\ \mathrm{m}, \tag{86}$$

where $\tau = 30$ m and the source strength κ is obtained generatively from the sampled radius R through the mapping above with $\Delta\rho$ fixed, giving $\kappa \in [-3940, -210]$ $\mu\mathrm{Gal}\cdot\mathrm{m}^2$. This places the horizontal source position within roughly ± 60 m of the origin, the depth at a median of 10 m (with 95% mass in $[3.7, 27]$ m and a floor at 3 m), and the source strength at typical void magnitudes.

D.3 POLICY NETWORKS

For each task we use the same policy network architecture across all methods, so the resulting policies differ only through the gradient estimator or alternative method that trains them. In total we use three different types of policy network across the experimental tasks. Firstly, in both settings of location finding we used the DAD policy network (Foster et al., 2021). For each of the dynamical systems models we used a stochastic LSTM with GRU cells as per Iqbal et al. (2024). Finally, in the gravimetry experiment we used the iDAD attention network introduced in Ivanova et al. (2021). Optimiser settings for the policy networks are also shared across all methods and are summarised in Table 4; these settings are largely borrowed from their original implementations.

DAD policy network (location finding). We use the deep adaptive design (DAD) policy of Foster et al. (2021). Each (y_t, ξ_t) pair is concatenated and passed through a shared encoder MLP with hidden layers $[256]$ and ReLU activations, producing a 16-dimensional encoding per time step. The history encoding is formed by summing the per-step encodings, which is permutation-invariant by construction, and is passed through a linear emitter to produce the next design $\xi_t \in \mathbb{R}^2$ (we reduce default initialisation of the weights in the final emitter layer from $\mathcal{U}(-\frac{1}{\sqrt{f_{\text{in}}}}, \frac{1}{\sqrt{f_{\text{in}}}})$ to $\mathcal{U}(-0.05, 0.05)$ to control the dynamic range of designs at initialisation).

Iqbal GRU policy network (dynamical systems). For all three dynamical-systems tasks we use the stochastic LSTM policy network of Iqbal et al. (2024). Each (y_t, ξ_t) pair is concatenated and passed through a shared encoder MLP with hidden layers $[256, 256]$ and ReLU activations, producing a 64-dimensional encoding per time step. The sequence of encodings is processed by a 2-layer GRU of hidden size 64, with the top-layer hidden state mapped through an emitter MLP with hidden layers $[256, 256]$ to the design dimensionality. Policies in this case are stochastic, following the original implementation by Iqbal et al. (2024). We therefore let the emitter output mean μ_t and we parameterise a per-component learnable standard deviation in log space, so that $\xi_t \sim \mathcal{N}(\mu_t,\, \mathrm{diag}(\sigma^2))$.

Attention policy network (gravimetry). For the gravimetry tasks we use a transformer-based policy inspired by the iDAD policy network (Ivanova et al., 2021). Each design ξ_t and outcome y_t is first embedded separately by design- and outcome-encoder MLPs, each with hidden layers $[64]$ and GELU activations producing 32-dimensional embeddings; these are concatenated and passed through a fusion MLP (hidden layers $[64]$, GELU) to give a 32-dimensional per-step pair representation r_t. The set of history representations $\{r_1, \ldots, r_t\}$, each augmented with a learned history-type embedding, is prepended with a learned decision token and processed by a single self-attention transformer encoder layer with 4 heads, model dimension 32 and a position-wise feed-forward network of hidden size 64 (using GELU activation, residual connections and layer normalisation throughout). A causal mask ensures that when emitting the design at step t the decision token attends only to itself and the t available history tokens, so the aggregation is invariant to the presentation order of the observations. The design is emitted from the decision-token output using an MLP with hidden layers $[64]$ and GELU activations. As with the DAD policy, we reduce the default initialisation of the final emitter layer weights to $\mathcal{U}(-0.05, 0.05)$ to control the dynamic range of designs at initialisation.

Policy output activations. In order to respect any scaling or bound constraints on designs ξ_t, we apply output activations to the raw outputs of the policy network which are detailed in Table 3.

Task	output activation
Location finding	$\xi_t = u_t$
Stochastic pendulum	$\xi_t = \tanh(u_t)$
Cart-pole	$\xi_t = 5\tanh(u_t)$
Double pendulum	$\xi_t = (4, 2) \odot \tanh(u_t)$
Gravimetry	$\xi_t = 40u_t$

Table 3: Policy output activations by task. $\odot$ represents element-wise multiplication and $\tanh$ is applied elementwise.

Optimiser settings. Each policy is trained with the Adam optimiser, gradient clipping at $\|\nabla\|_2 \leq 1$, and an exponential learning-rate decay schedule. The full hyperparameters are given in Table 4.

Task	# iters	batch size	schedule	init LR	decay rate	decay every	(β_1, β_2)
Location finding	20,000	1024	exp decay	5×10^{-5}	0.98	1000	$(0.8, 0.998)$
Stochastic pendulum	500	1024	constant	1×10^{-4}	N/A	N/A	$(0.9, 0.999)$
Cart-pole	500	1024	constant	1×10^{-4}	N/A	N/A	$(0.9, 0.999)$
Double link	500	1024	constant	1×10^{-4}	N/A	N/A	$(0.9, 0.999)$
Gravimetry	10000	1024	exp decay	5×10^{-5}	0.98	1000	$(0.8, 0.998)$

Table 4: Policy training hyperparameters. All optimisers are Adam with gradient clipping at $\|\nabla\|_2 \leq 1$.

D.4 SCORE MATCHING APPROACH

Score network architecture. For all tasks we use the same backbone architecture, differing only in input/output dimensionalities, the presence of positional encodings, and the choice of inputs to the final output head (detailed below). The backbone is a stack of pre-norm transformer blocks (Vaswani et al., 2017) with two prediction heads, one for $\nabla_{y_t} \log p(y_{1:T} \mid \xi_{1:T})$ and one for $\nabla_{\xi_t} \log p(y_{1:T} \mid \xi_{1:T})$. The exact configuration is given in Table 5 and each component is detailed below.

Embedding. Each y_t and ξ_t is first standardised then separately embedded with a random Fourier features layer (Tancik et al., 2020) (with $\sigma = 10$) into 64-dimensional features each. These are concatenated and passed through a 2-layer MLP with $[192, 128]$ hidden units and GeLU activations. A linear projection maps these to the 256-dim model space, giving a sequence of T per-timestep embeddings.

Positional encodings. When positional encodings are required (see below), a sinusoidal encoding (Vaswani et al., 2017) is added to the per-timestep embeddings.

Backbone. A learnable global token is joined to the sequence, making its length $T + 1$. The sequence is then processed by 4 transformer blocks, each consisting of layer norm, multi-head self-attention over the time dimension (8 heads), residual, layer norm, a 2-layer MLP with hidden width $2 \cdot d_{\text{model}}$ and GeLU activations, and a final residual.

Prediction heads. Following the main transformer backbone, two independent prediction heads produce the y-score and ξ-score components. These heads consist of a layer norm followed by a 2-layer MLP with d_{model} and GeLU activations. The prediction heads receive different inputs depending on the task, and use a separate "head embedding" — an independent RFF + MLP embedding of the raw (y_t, ξ_t) pair. The inputs are constructed as follows:

- *standard inputs* (location finding): the head input is the concatenation of [global token, head embedding at t, transformer backbone output at t].
- *markov window inputs* (dynamical systems): the head input additionally concatenates the head embeddings at $t - 1$ and $t + 1$ (zero-padded at the boundaries). This Markov window is motivated by the Markovian likelihoods of the dynamical systems since the conditional score at time t has explicit dependence on its one-step neighbours. This setup saves the backbone output from having to reconstruct this information through the attention layers.

Conservative wrapper. The two-headed transformer described above produces a $d_y + d_\xi$-dimensional vector at each time step. We project this to a scalar potential $\Phi_\psi(y_{1:T}, \xi_{1:T})$ via a final linear layer, and define the score network as $s_\psi(y_{1:T}, \xi_{1:T}) = \nabla_{(y_{1:T}, \xi_{1:T})} \Phi_\psi(y_{1:T}, \xi_{1:T})$, obtained by automatic differentiation. This ensures the learned score is

conservative (i.e. it is the gradient of a scalar function), a structural property that the true score $\nabla_{(y_{1:T},\xi_{1:T})} \log p(y_{1:T} \mid \xi_{1:T})$ satisfies. This is a principled design choice which was found to greatly stabilise policy training on the location finding task and therefore was adopted across all tasks. However, we note that computationally this choice considerably increases the burden of evaluating the score network during policy training.

Permutation equivariance. For tasks with an exchangeable likelihood, $p(y_{1:T} \mid \xi_{1:T})$ is exchangeable with respect to the ordering of (y_t, ξ_t) pairs, so the score is permutation-equivariant in t. We bake this into the network by omitting the positional encoding. For the dynamical systems the likelihood is Markovian and the score is not equivariant, therefore we add sinusoidal positional encodings to the per-timestep embeddings.

Table 5: Score network architecture, shared across all tasks (excluding positional encodings for non-iid tasks and task specific inputs to the output heads). The full network has approximately 3 million parameters.

Component	Hyperparameter	Value
RFF embedding	dimension	64 (i.e. 128 after concat)
	σ	10.0
Embedding MLP	hidden widths	$[192, 128]$
	activation function	GeLU
Transformer	model dim d_{model}	256
	# transformer blocks	4
	# attention heads	8
	MLP hidden width	$2 \cdot d_{\text{model}} = 512$
	activation function	GeLU
	global summary token	yes (learnable)
Head embedding	RFF dim	64
	MLP hidden widths	$[192, 128]$
Head MLP	hidden width	$d_{\text{model}} = 256$
	activation	GeLU
Potential proj.	final linear to scalar Φ_ψ	yes
Score	$s_\psi = \nabla_{(y_{1:T},\xi_{1:T})} \Phi_\psi$ (conservative)	yes

Score training. The score network is trained on the marginal score-matching loss Equation (6) with the Adam optimiser (Kingma and Ba, 2017) and a warmup-cosine learning-rate schedule that linearly warms the learning rate up over the first 0.1% of training to a peak value, then cosine-decays to a small floor of 10^{-5}. The standard score-matching loss is augmented with a per-coordinate weighting on the y-score component (denoted λ_y in Table 6) chosen so that the y-score loss is comparable in magnitude to the ξ-score loss; this is task-dependent because the relative scales of the y- and ξ-score components differ. Designs for the score-training data distribution are sampled from simple distributions spanning the relevant region of ξ-space, discussed in greater detail below. All training hyperparameters are summarised in Table 6. The number of training iterations is dictated for each experiment by the allowed NLE budget, which we describe in Appendix D.8.

Task	batch size	peak LR	grad-norm clip	λ_y
Location finding	1024	2×10^{-4}	600	30.0
Stochastic pendulum	1024	2×10^{-4}	5,000	0.0125
Cart-pole	1024	2×10^{-4}	600	0.004
Double link	1024	2×10^{-4}	10,000	0.02
Gravimetry	1024	2×10^{-4}	5,000	10000

Table 6: Score network training hyperparameters. Optimiser is Adam with default $(\beta_1, \beta_2) = (0.9, 0.999)$. Learning rate uses linear warmup over 0.1% of training iterations to the peak value, then cosine decay to 10^{-5}.

Score-based policy training. Once the score network is trained, the policy is optimised using the EIG gradient estimator of Equation (3), with the intractable score terms replaced by s_ψ. The policy network and its optimiser settings are shared with the baselines and described in Section D.3.

D.4.1 Design sampling distributions

As described in Section 3, our marginal score matching approach samples designs from a distribution $q(\xi)$ which we are free to choose. In the following we describe the distributions used in each experiment, while in Appendix E.3 we ablate a wider selection of sampling distributions to illustrate the influence of this choice. In each case the chosen samplers reflect our a-priori beliefs about the regions of design space which may be likely under an optimal policy.

Location finding. Here we use a temporally correlated Gaussian distribution with uniform hyperpriors on temporal correlation and random scale. Specifically, for each sampled trajectory we first draw a scale $\sigma \sim \mathcal{U}(0.2, 5.0)$ and a temporal correlation coefficient $\rho \sim \mathcal{U}(0.7, 1.0)$, and then sample the full design sequence $\xi = (\xi_1, \dots, \xi_T)$, with each $\xi_t \in \mathbb{R}^d$, from the zero-mean Gaussian

$$\xi \sim \mathcal{N}\big(0,\ \sigma^2\, K_\rho \otimes I_d\big)\,, \qquad [K_\rho]_{s,t} = \rho^{|s-t|}, \tag{87}$$

where $K_\rho \in \mathbb{R}^{T\times T}$ is an AR(1) correlation matrix over the T experiment steps and the d spatial coordinates of each design are sampled independently. This construction gives designs that are marginally $\mathcal{N}(0, \sigma^2 I_d)$ but whose successive locations are smoothly correlated in time: $\rho \to 1$ yields near-constant trajectories while smaller ρ produces more rapidly varying ones.

Dynamical systems. Here we draw design trajectories from an equal mixture of two distributions. Firstly, a simple i.i.d. uniform distribution on the design space $[-B, B]$. Secondly, a 'pseudo-random' design distribution which concentrates designs on the boundary by randomly switching between each boundary according to a Poisson process. Specifically, for each trajectory and each design dimension we sample a sign process $s_t \in \{-1, +1\}$: the initial sign s_1 is drawn uniformly, and at each subsequent step the sign flips with probability $1 - e^{-\lambda}$, where the switch rate is drawn per trajectory as $\lambda \sim \mathcal{U}(0.02, 0.1)$. This is a discrete-time analogue of a Poisson switching process, so on average the design remains pinned to one boundary for a geometrically distributed number of steps before flipping to the other. In practice we apply a small jitter to sit just inside the boundary by adding standard Gaussian noise around a nominal boundary level u_B for which $\tanh(u_B) \approx 1$. In other words, we sample $\xi_t = B\tanh(s_t u_B + \varepsilon)$ where ε is Gaussian noise and s_t is the sign process.

Gravimetry. Here we sample designs independently from a Gaussian with trajectory-level random scale $\sigma \sim \mathcal{U}[0.3, 1.2]$. Designs are then mapped to the physical space using the output activation $\xi_t = 40u_t$, similarly to the policy output activation detailed in Appendix D.3.

D.5 BASELINES

PCE. PCE (Foster et al., 2020) uses the same policy architecture and training schedule as our score-based approach (see Table 4) but replaces the score-based gradient estimator with the sequential PCE estimator (Foster et al., 2021). The number of contrastive samples for each task is dictated by the allowed NLE budget which we describe in Appendix D.8.

Pre-trained variational marginal (VarMarg). The variational marginal approach (Foster et al., 2019) replaces the intractable $\log p(y_{1:T} \mid \xi_{1:T})$ in the MI form of the EIG with a learned approximation $\log q_\eta(y_{1:T} \mid \xi_{1:T})$, providing an upper-bound surrogate. The variational approximation q_η is a Masked Autoregressive Flow (Papamakarios et al., 2017) with rational-quadratic-spline transforms (Durkan et al., 2019): 5 MAF layers, conditioner networks of depth 2 and width 80, and 8 spline bins on an interval of $[-4, 4]$. The base distribution is a standard Normal of dimension $T(d_y + d_\xi)$, and the flow is trained by maximum likelihood on samples from the model. We train q_η using a warmup-cosine-decay learning-rate schedule with peak 10^{-3} and floor 10^{-6}. We sample designs from the same design sampling distributions as the score network (Appendix D.4.1) and we model the joint $p(y_{1:T}, \xi_{1:T})$ then subtract the analytic marginal design score to obtain the required conditional. The policy is then trained with $\log q_\eta(y_{1:T} \mid \xi_{1:T})$ providing an approximation of the intractable marginal. The policy training hyperparameters are identical to those in Table 4 and the number of training iterations is dictated by the allowed NLE budget as per Appendix D.8.

Pre-trained variational posterior (VarPost). In this approach, we pre-train a variational approximation of the posterior distribution $p(\theta|h_T)$ and use the resulting approximation to train the policy by maximising the Barber-Agakov lower bound on $\mathcal{I}_T$ (Barber and Agakov, 2003). Again we sample designs from the same design sampling distributions as the score network (Appendix D.4.1).

Joint variational posterior (JointVarPost). The joint variational posterior approach (Foster et al., 2020) co-optimises the variational posterior $q_\eta(\theta \mid y_{1:T}, \xi_{1:T})$ and the policy by maximising the Barber-Agakov lower bound on the EIG.

The variational posterior $q_\eta(z \mid y_{1:T}, \xi_{1:T})$ is shared between each of the variational approaches. It is a transformer-encoded conditional mixture of Gaussians where the transformer backbone mimics the architecture of the score networks as closely as possible to provide a fair comparison. Specifically, it is identical to the score transformer network presented in Appendix D.4 up until the prediction heads, where here we use per-component heads to predict the means and diagonal log-scales of the K mixture components and the mixing logits. We use $K = 10$ components. The network is specialised further in the location finding task where we canonicalise the hidden source ordering by sorting on their first coordinate to maintain identifiability. Furthermore, in the dynamical systems tasks we use positional encodings to capture the permutation-aware structure of the conditioning observations.

In each variational approach, we train the posterior with batch size 1024 and one posterior sample per data point per step. The posterior optimiser is Adam with a warmup-cosine schedule (peak 10^{-3}, floor 10^{-6}, warmup over 0.1% of training). In the joint approach, the policy optimiser is Adam with a cosine-decay schedule from 10^{-5} to 5×10^{-6}.

IO-SMC^2. For IO-SMC^2 we use the authors' public Jax implementation (Iqbal et al., 2024). We only modified the diffusion matrices from their set-up in order to avoid the extremely high EIG settings where policy performance is hard to reliably validate. In order to compensate for the lower reward signal in this regime, we experimented with increasing the tempering parameter and reducing the strength of the slew rate penalty on designs, however neither of these provided any noticeable improvements. The hyperparameters used are therefore identical to the original implementation and we varied the number of particles used in order to respect the fixed NLE budget, which is described in Appendix D.8.

D.6 EVALUATION BOUNDS

For each trained policy we report upper and lower bounds on the true total EIG $\mathcal{I}_T(\phi)$. These bounds are introduced by Foster et al. (2021) and are computed as follows. Let $\theta_0 \sim p(\theta)$ and $y_{1:T} \sim p_\phi(y_{1:T} \mid \theta_0)$ denote an outer sample from the model under the policy, and let $\theta_{1:M} \overset{\text{i.i.d.}}{\sim} p(\theta)$ denote M independent contrastive samples from the prior.

sNMC upper bound. The sequential NMC estimator of Foster et al. (2021); Rainforth et al. (2018) gives the upper bound

$$\mathcal{I}_{T,M}^{\mathrm{sNMC}}(\phi) = \mathbb{E}\left\{\log \frac{p(y_{1:T} \mid \theta_0, \xi_{1:T})}{\frac{1}{M}\sum_{m=1}^{M} p(y_{1:T} \mid \theta_m, \xi_{1:T})}\right\} \geq \mathcal{I}_T(\phi), \tag{88}$$

which is asymptotically tight as $M \to \infty$.

sPCE lower bound. The sequential prior contrastive estimator of Foster et al. (2021) gives the lower bound

$$\mathcal{I}_{T,M}^{\mathrm{sPCE}}(\phi) = \mathbb{E}\left\{\log \frac{p(y_{1:T} \mid \theta_0, \xi_{1:T})}{\frac{1}{M+1}\sum_{m=0}^{M} p(y_{1:T} \mid \theta_m, \xi_{1:T})}\right\} \leq \mathcal{I}_T(\phi), \tag{89}$$

which is asymptotically tight as $M \to \infty$ and bounded above by $\log(M + 1)$.

Sample sizes. Both bounds are estimated with $M = 1 \times 10^6$ inner (contrastive) samples and $N = 2048$ outer samples for every (policy, task) pair, with the exception of the standard location finding task and the gravimetry tasks, where $M = 5 \times 10^6$ contrastive samples were used. With $N = 2048$ samples, the standard error of the bound estimates was found to be below the standard error over randomness in the choice of final policy.

D.7 NLE CALCULATIONS

To ensure a fair comparison and highlight the benefit of pre-training the score network in our approach, we compare all methods under a fixed budget of *number of likelihood evaluations* (NLE) consumed during training. We define one NLE as a single evaluation of the conditional likelihood $p(y_t \mid \theta, \xi_t, h_{t-1})$, so an evaluation of the joint log-likelihood over the full experiment costs T NLE. We use NLE as a natural cost metric because it provides a comparable measure of how much information each method consumes from the model. For implicit methods (the variational posterior approaches), these methods do not evaluate the model likelihood but do require samples from it, therefore we use the number of samples as an equivalent proxy.

Method	Upfront	Per-policy
Score-based	$K_U B_U T$	$K_P N T$
PCE	—	$K_P N(M+1)T$
VarMarg	$K_U B_U T$	$K_P N T$
VarPost	$K_U B_U T$	$K_P N T$
JointVarPost	—	$K_J B_J T$
IO-SMC^2 (lower)	—	$KS(NMT + (N-1)MT) + NMT$
IO-SMC^2 (upper)	—	$KS(NMT + (N-1)(MT + MRT(T+1)/2)) + NMT + NMRT(T+1)/2$

Table 7: Per-policy and total-budget NLE formulas for each method. "Upfront" is paid once and amortises over P policies. IO-SMC^2 costs are stated as lower/upper bounds spanning the IBIS resampling regime; the realised cost depends on the rate at which the degeneracy criterion triggers.

We use the following common notation: K_P is the number of policy gradient steps, N the policy training outer batch size, T the experiment length, P the number of policies trained, K_U, B_U the number of iterations and batch size for upfront training (applies to score, VarMarg and VarPost methods), M the number of contrastive samples per outer sample for PCE, and K_J, B_J the joint iterations and batch size for the joint variational posterior training. The total cost across P policies is upfront $+ P \cdot$ per-policy. We summarise the calculations in Table 7.

Score-based (and pre-trained variational). Our score-based approach incurs NLE in two phases. The upfront variational approaches share an identical cost structure. The upfront score-training phase performs K_U iterations with batch size B_U; each sample requires one evaluation of the per-time-step grad-log-likelihood in the marginal score matching objective (Equation (6)), which we count as T NLE per sample. The upfront cost is therefore $K_U B_U T$. The policy-training phase performs K_P iterations with batch size N; each sample requires one evaluation of the joint log-likelihood to compute the conditional-likelihood gradient term of Equation (3), costing T NLE. The per-policy cost is therefore $K_P N T$, and total cost across P policies is

$$C_{\text{score}}(P) = K_U B_U T + P \cdot K_P N T. \tag{90}$$

The upfront score-training cost is paid once and amortises across all P policies, giving a key computational advantage when P grows.

PCE. Using the PCE (Foster et al., 2021), the policy is trained for K_P iterations with batch size N. Each sample requires one evaluation of the joint log-likelihood under the realised θ (T NLE), plus an inner Monte-Carlo estimate of $p(y_{1:T} \mid \xi_{1:T})$ using M samples from the prior (MT NLE). The per-policy cost is therefore $K_P N(M+1)T$ and the total is

$$C_{\text{PCE}}(P) = P \cdot K_P N(M+1)T. \tag{91}$$

Co-trained variational posterior. The variational-posterior approach (Foster et al., 2020) co-trains a posterior $q_\eta(\theta \mid y_{1:T}, \xi_{1:T})$ and the policy on the Barber-Agakov lower bound. We charge T NLE per outer sample so that joint training for K iterations with batch size B costs

$$C_{\text{var-post}}(P) = P \cdot K_J B_J T. \tag{92}$$

The cost is per-policy because the posterior is specialised to the current policy and is not re-used across restarts.

IO-SMC^2. IO-SMC^2 (Iqbal et al., 2024) implements a particle filter (IBIS) within a particle filter (CSMC) within Markovian score climbing. Working from the outermost loop inwards: score climbing performs K gradient-based maximisation iterations on the policy parameters, each of which calls IO-SMC^2 once to sample N trajectories from a reward-tilted distribution. Note that the policy-parameter gradient is taken through the policy density $\nabla_\phi \log \prod_{t=1}^{T} \pi_\phi(\xi_t \mid z_{0:t-1})$, not through the dynamics likelihood, so the outer optimisation itself contributes no likelihood evaluations.

The inner IO-SMC^2 algorithm uses IBIS to approximate the posterior over θ at each step with M particles. Each IBIS step costs at minimum M NLE. If a degeneracy criterion is met at step t, R move-resample steps cost an additional MRt NLE. The per-step IBIS cost is therefore between M and $M(1+Rt)$. Particle re-weighting in the CSMC kernel itself costs M NLE per trajectory per step. Summing over T steps and N trajectories (IBIS is run for $N-1$ trajectories since the reference trajectory is inherited), and accounting for the one-off cost of initialising the reference trajectory via unconditional SMC, the

Table 8: Location Finding budget configurations. All methods share a fixed NLE budget of exactly 6.144×10^9; PCE* uses $10\times$ this budget (6.144×10^{10}) reflecting more typical values required for this method. Shared fixed values (across all rows): experiment length $T = 30$, policy-training iterations $K_P = 10^4$, policy batch size $N = 1024$, and upfront/joint batch size $B_U = B_J = 1024$. We tabulate only the quantities varied to meet the budget: the number of policies P, the upfront/joint training iterations K_U/K_J, and the number of PCE contrastive samples M.

Method	P	K_U / K_J	M
Score-based (plus VarMarg, VarPost)	1	190,000	—
Score-based (plus VarMarg, VarPost)	5	150,000	—
PCE	1	—	19
PCE	5	—	3
PCE*	1	—	199
JointVarPost	1	200,000	—
JointVarPost	5	40,000	—

total cost of IO-SMC^2 with S CSMC kernel applications per gradient step satisfies

$$C^{\text{lower}}_{\text{IO-SMC}^2}(P) = P \cdot \big[KS(NMT + (N{-}1)MT) + NMT\big], \tag{93}$$

$$C^{\text{upper}}_{\text{IO-SMC}^2}(P) = P \cdot \Big[KS\big(NMT + (N{-}1)(MT + MRT(T{+}1)/2)\big) + NMT + NMRT(T{+}1)/2\Big]. \tag{94}$$

The realised cost falls between these bounds. In our experiments we track the actual number of evaluations and report these additionally. The original IO-SMC^2 paper uses $S = 1$ and $R = 3$.

Table 9: Dynamical Systems configurations. All methods share a fixed NLE budget of exactly 1.024×10^{10}. Shared fixed values (across all rows): experiment length $T = 50$, policy-training iterations $K_P = 500$, policy batch size $N = 1024$, and upfront/joint batch size $B_U = B_J = 1024$. We tabulate only the varied quantities: the number of policies P, the upfront/joint training iterations K_U/K_J, and the number of PCE contrastive samples M. Realised IO-SMC^2 configurations are reported separately in Table 11.

Method	P	K_U / K_J	M
Score-based (plus VarMarg, VarPost)	1	199,500	—
Score-based (plus VarMarg, VarPost)	50	175,000	—
PCE	1	—	399
PCE	50	—	7
JointVarPost	1	200,000	—
JointVarPost	50	4,000	—

D.8 NLE VALUES

In the previous subsection, we detailed the NLE cost structure of each method. In our experimental protocol, all methods were assigned a fixed NLE budget. We presented the results for each method by training either one or multiple policies P. For $P > 1$ we select the best performing policy based on an initial estimate of its PCE lower bound, and report an independent re-estimate of the EIG bounds for that selected policy. Depending on P, we adjusted the number of upfront/joint training iterations K_U, K_J and the number of contrastive samples M for each method (similarly the number of particles M and trajectories N in IO-SMC^2) in order to respect the fixed NLE budget available. This is representative of the procedure one might undertake when selecting a policy network by testing multiple architectures/hyperparameters or by training multiple policies to avoid local optima. We provide full details of the NLE budget and the configurations used to achieve these budgets in Tables 8 to 11.

Table 10: Gravimetry configurations. All methods share a fixed NLE budget of 4.096×10^9; PCE* uses $10\times$ this budget (4.096×10^{10}). Shared fixed values (across all rows): experiment length $T = 20$, policy-training iterations $K_P = 10^4$, policy batch size $N = 1024$, and upfront/joint batch size $B_U = B_J = 1024$. We tabulate only the varied quantities: the number of policies P, the upfront/joint training iterations K_U/K_J, and the number of PCE contrastive samples M.

Method	P	K_U / K_J	M
Score-based (plus VarMarg, VarPost)	1	190,000	—
Score-based (plus VarMarg, VarPost)	5	150,000	—
PCE	1	—	19
PCE	5	—	3
PCE*	1	—	199
JointVarPost	1	200,000	—
JointVarPost	5	40,000	—

Table 11: IO-SMC^2 NLE bounds on Dynamical Systems experiments. Here we report the configuration used and the upper and lower bounds on NLE cost. The configuration was chosen to capture the target NLE value of 1.024×10^{10} allowed to each other method. We then report the realised NLE value on each experiment. We did not use a $P = 50$ variant for IO-SMC^2 as the wall-clock time required to train and evaluate the required number of policies using the IO-SMC^2 implementation was prohibitive.

(K, N, M, R, S)	lower	upper
$(25, 1024, 512, 3, 1)$	1.34×10^9	5.34×10^{10}

	Stochastic Pendulum	Cart-Pole	Double Pendulum
Realised NLE	$9.9 \times 10^9 \pm 5.7 \times 10^7$	$1.5 \times 10^{10} \pm 3.6 \times 10^7$	$1.8 \times 10^{10} \pm 7.8 \times 10^7$

E ABLATION STUDIES, FURTHER RESULTS AND INSIGHTS

E.1 MLMC RESULTS

We implemented the MLMC approach of Goda et al. (2022) and tested this on the standard location finding experiment. MLMC uses a debiasing scheme which randomises the number of inner samples to provide an unbiased estimate of the EIG gradient. In our experiments we found that the resulting gradients are high variance, therefore we increased the number of outer samples per gradient estimate and trained the policy network for longer to allow proper convergence. We obtained a **lower bound of** 7.21 ± 0.40 **and an upper bound of** 7.27 ± 0.40 **on the** $d = 2, K = 2$ **location finding task**. Unfortunately, the implementation took > 24 hours to train a single policy, therefore we were unable to provide results across all our experimental settings. We provide the experimental settings and approximate NLE cost used for these results in Table 12. Note that the configuration we tested far exceeds the NLE cost of the budget for the main results, yet the MLMC performance was still significantly worse.

NLE cost of MLMC. Each outer sample $n = 1, \ldots, N$ is assigned a random level L_n drawn from a geometric distribution,

$$\Pr(L = l) = (1 - 2^{-\tau})\,(2^{-\tau})^l, \qquad l = 0, 1, 2, \ldots, \tag{95}$$

and the number of inner (nested) samples at level l is $M_0 2^l$. The expected number of inner samples per outer sample is therefore

$$\mathbb{E}\big[M_0 2^L\big] = M_0(1 - 2^{-\tau}) \sum_{l=0}^{\infty} \big(2^{1-\tau}\big)^l = M_0 \, \frac{1 - 2^{-\tau}}{1 - 2^{1-\tau}}, \tag{96}$$

which is finite only for $\tau > 1$. We used $\tau = 1.1$, for which this is $\approx 7.96 M_0$. This gives a total theoretical expected NLE cost of $7.96 T K_P N M_0$.

To compile and parallelise the loop we require a static number of inner samples. We therefore cap the level L_n at $l_{\max} = 12$,

Table 12: Configuration and NLE values for MLMC on the standard location finding experiment. Theoretical NLE gives an expected NLE based on the expected number of inner samples per outer sample in the randomised scheme. Realised NLE reports the actual NLE required due to padding to allow parallelisation.

N	2000
M_0	1
K_P	50000
τ	1.1
Theoretical NLE	2.4×10^{10}
Realised NLE	2.0×10^{13}

Method	Lower Bound	Upper Bound
SCOREBED (P=1)	8.09 ± 0.09	9.31 ± 0.11
SCOREBED (P=5)	8.42 ± 0.16	9.49 ± 0.27
PCE (P=1)	8.43 ± 0.06	8.78 ± 0.09
PCE (P=5)	8.37 ± 0.06	8.85 ± 0.12
JointVarPost (P=1)	$\mathbf{8.86 \pm 0.04}$	10.20 ± 0.15
JointVarPost (P=5)	$\mathbf{8.80 \pm 0.04}$	10.54 ± 0.48
VarMarg (P=1)	7.51 ± 0.13	8.19 ± 0.15
VarMarg (P=5)	7.65 ± 0.06	8.55 ± 0.14
VarPost (P=1)	$\mathbf{8.85 \pm 0.05}$	9.72 ± 0.07
VarPost (P=5)	$\mathbf{8.87 \pm 0.04}$	9.91 ± 0.15

Table 13: Results on the 1D gravimetry task. Each approach is allowed an NLE budget of 4.096×10^9. Results are mean ± one standard error over policy repeats with bold indicating statistically significant best lower bound at 95% confidence level.

which clips a total mass of at most 0.01% of the tail. The actual realised compute cost is therefore $TNK_PM_02^{l_{\max}}$, in other words equivalent to PCE with $2^{12}M_0 = 4096M_0$ inner samples.

E.2 GRAVIMETRY RESULTS

We consider an additional experimental design task inspired by the gravity methods described in Telford et al. (1990) with which one infers the unknown location, depth and strength of an underground void based on measurements of local gravitational anomalies. We consider a one-dimensional search space, where designs $\xi_t \in \mathbb{R}$ are placed on a line and a noisy measurement of the gravitational deviation is observed. This task can be seen as a more challenging development of the source location finding task. In particular, the signal strength—a function of the size and relative density of the underground void—is now unknown and the depth and signal strength can only be independently identified by measurements adjacent to but not directly above the source. We place $T = 20$ designs and train an adaptive TNP-style policy network (Nguyen and Grover, 2022). Results are presented in Table 13 where again we observe that SCOREBED performs on par with PCE and slightly worse than the variational approaches.

E.3 CHOICE OF DESIGN SAMPLING DISTRIBUTION

Here we explore the importance of the design sampling distribution $q(\xi_{1:T})$ used during score network training. We provided details of the sampling distributions used in our main results (see Section 5) in Appendix D.4.1. Here we present additional results using different choices of design sampling distribution.

Location finding. Here we consider the influence of the design sampling distribution used during score network training for the $d = 2, K = 2$ location finding task. We consider four different choices of design sampler. Firstly, a standard isotropic Gaussian sampler which simply samples i.i.d. designs $\xi_t \sim \mathcal{N}(0, 1), t = 1, ...T$, which represents the simplest possible choice for this experiment. Secondly, we augment the standard Gaussian with a random scale sampled uniformly per trajectory from $\mathcal{U}[0.5, 2]$. Thirdly, we consider a handcrafted design distribution which was specifically created to mimic the behaviour of an optimal DAD policy, which typically exhibits a spiralling behaviour. This was achieved by sampling

Sampler	Lower Bound	Upper Bound
Gaussian	2.86 ± 0.26	2.86 ± 0.26
Random scale Gaussian	$\mathbf{10.84 \pm 0.08}$	10.97 ± 0.09
Temporal correlation	10.81 ± 0.06	10.95 ± 0.07
Low rank projection	$\mathbf{11.01 \pm 0.02}$	11.18 ± 0.02

Table 14: Ablation results for design sampler choice on the $d = 2, K = 2$ location finding task. Final policies are obtained using the $P = 1$ protocol. Results present mean ± one standard error with bold indicating statistically best lower bound at the 95% confidence level.

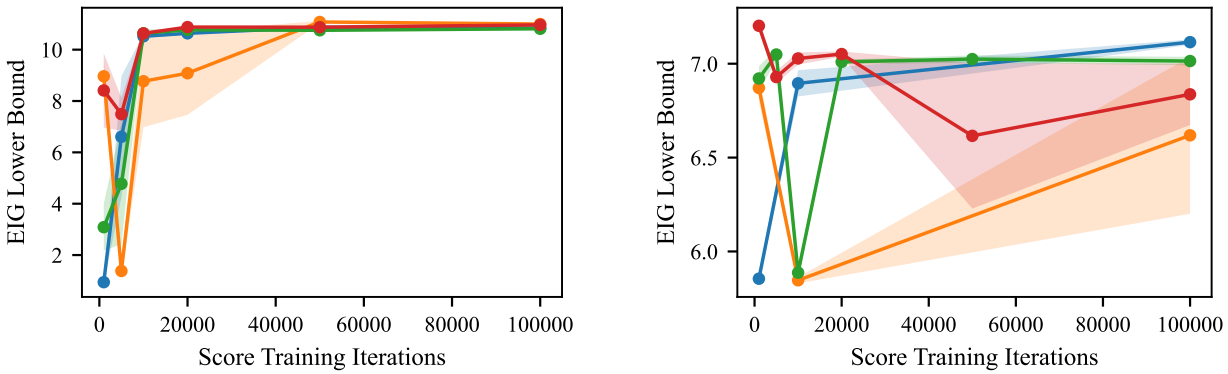

Figure 4: Stability of SCOREBED with respect to score training. Each curve presents the final policy EIG under the $P = 1$ protocol against number of score training iterations. Left: $d = 2, K = 2$ location finding. Right: cart-pole.

trajectories as low-rank projections of Gaussian noise, $\xi = \Phi z$, where $\Phi \in \mathbb{R}^{T \times r}$ is a time-basis matrix and $z \in \mathbb{R}^{r \times d}$ are Gaussian coefficients, $z_{ij} \sim \mathcal{N}(0, s^2)$ with an overall scale drawn randomly as $s \sim \mathcal{U}[0.5, 2]$. For each sample the basis Φ is drawn uniformly from a family of temporal bases — sinusoidal, Fourier, or spirals — and the projection rank r is drawn uniformly in $[2, 8]$, so that designs are smooth, low-dimensional curves in time whose complexity varies across samples. Finally, for reference we include the temporally correlated random scale Gaussian which was used in our main results.

We present the results of the ablation in Table 14. We find that when the score network is trained on the plain isotropic Gaussian, it fails to train successful policies. The other design sampler choices perform similarly, with marginally best performance from the low rank projection sampler. From our experiments, we found that the critical ingredient for this task is the hyperprior on the sampler scale. This is critical because at initialisation, the policy network emits designs on a scale much smaller than a standard Gaussian and the scale then grows as the policy trains. Including a hyperprior on design scales better covers the design space in the regions that the policy visits during training. The custom low rank projection sampler gives the best performance although it required substantial customisation to do so.

E.4 SCORE TRAINING STABILITY

We investigated the stability of SCOREBED with respect to score training seed. Figure 4 plots the final policy EIG under the $P = 1$ protocol for multiple score network training seeds on the standard location finding and cart-pole tasks. While SCOREBED is clearly very stable on the location finding task, we observe instability early in the score network training, as noted in Section 5. In order to compensate for this, we split the allowed score budget between multiple score networks, therefore guarding against this instability. We believe that this instability is more precisely attributed to unstable policy training due to bias in the SCOREBED EIG gradient approximation, which we explore in the following section.

E.5 ADDITIONAL BIAS-VARIANCE RESULTS

We now extend the discussion from Section 5.3 by exploring the bias of the EIG gradients of SCOREBED and PCE on a larger set of tasks. Figure 7 presents error curves of SCOREBED against N for increasing score training iterations on both location finding tasks and two dynamical systems tasks. It is immediately clear that SCOREBED achieves considerably

lower bias than PCE on both location finding tasks, while SCOREBED carries greater bias on both the stochastic pendulum and cart-pole tasks compared with PCE. As per our bias–variance decomposition and discussions in Section 5.3, we believe that gradient bias is a key factor in the quality of policy training as large bias can throw off policy training by pushing the policy in suboptimal directions. Indeed, the scatter plots in Figure 5 show the correlation between EIG gradient bias and final policy performance, illustrating that lower bias is associated with stable policy performance. We see that the bias of SCOREBED on the dynamical systems tasks may be on the verge of stable performance, but by selecting policies between multiple score training seeds in the $P > 1$ protocol, we are still able to avoid this instability.

In order to investigate the cause of this bias on the dynamical systems tasks, we explored correlation between score MSE and the conditioning of the Jacobian $J_\xi(y) = \frac{\partial y}{\partial \xi}$ appearing in the gradient bias (cf. eq. (68)), see Figure 6. We found that in the trained (final) policy regime, trajectories have significantly worse Jacobian conditioning and there is apparent correlation between score MSE and Jacobian conditioning. This is unsurprising since the non-Markovian property of the dynamical systems models makes them particularly sensitive to perturbations early in the trajectory, and it suggests the bound on EIG gradient error in eq. (68)— equivalently the constant L_y in eq. (52)—may be such that the score networks are required to be extremely accurate on this task to provide sufficiently low EIG gradient bias.

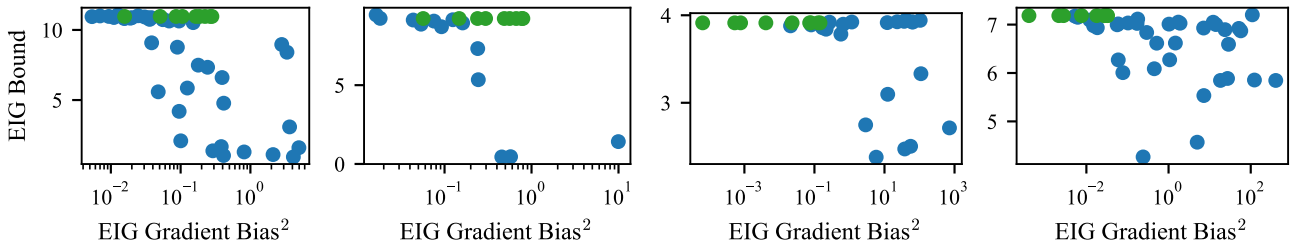

Figure 5: Scatter plots showing correlation between EIG gradient bias and final policy performance, with points representing different score training budgets/seeds (blue) and different numbers of PCE inner samples (green). From left to right: location finding ($d = 2, K = 2$), location finding ($d = 3, K = 10$), stochastic pendulum, cart-pole.

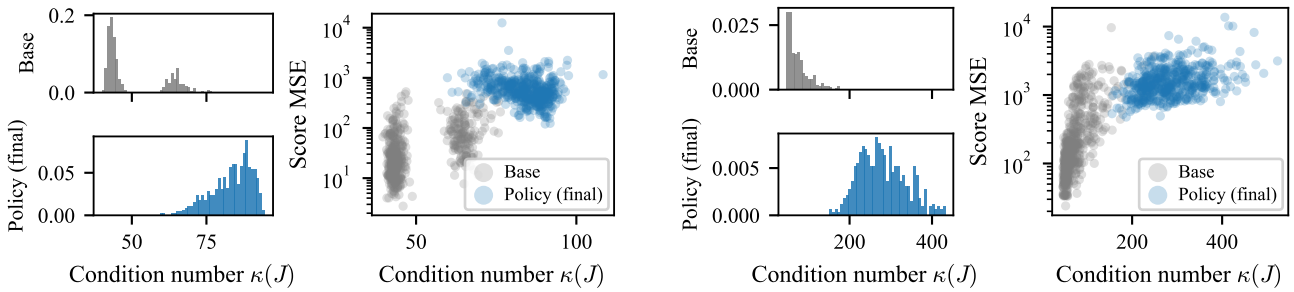

Figure 6: Histograms showing the distribution of Jacobian condition number between the score training distribution and the distribution induced by the final policy network. Scatter plots show correlation between score MSE and Jacobian condition number. Left: stochastic pendulum. Right: cart-pole.

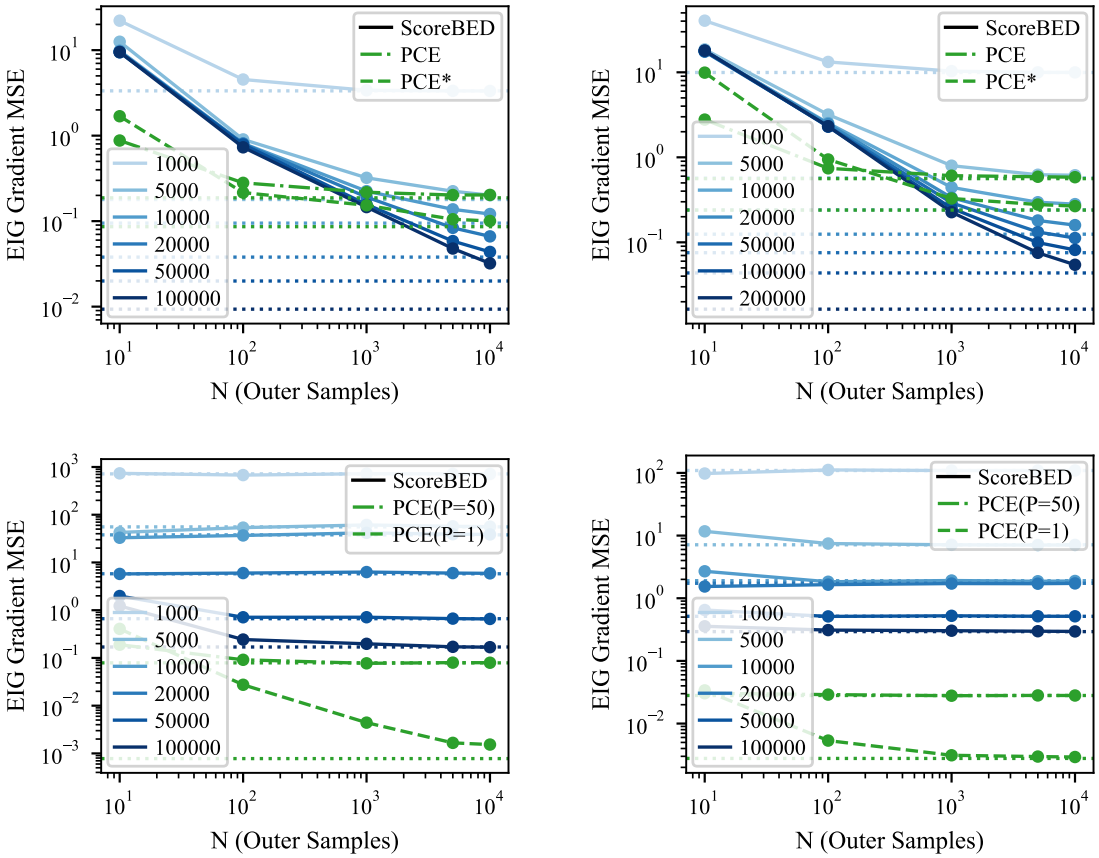

Figure 7: EIG gradient MSE curves for SCOREBED and PCE. PCE uses $M = 19$ and PCE* uses $M = 199$ inner samples on the location finding tasks, while PCE ($P = 50$) uses $M = 7$ and PCE ($P = 1$) uses $M = 399$ on the dynamical systems tasks. These values are chosen according to the fixed NLE budgets for each task as per Appendix D.8. From top left to bottom right: location finding ($d = 2, K = 2$), location finding ($d = 3, K = 10$), stochastic pendulum, cart-pole.