NeST-BO: Fast Local Bayesian Optimization via Newton-Step Targeting of Gradient and Hessian Information

Wei-Ting Tang
University of Wisconsin-Madison

Akshay Kudva
The Ohio State University

Joel A. Paulson
University of Wisconsin-Madison

Abstract

Bayesian optimization (BO) is effective for expensive black-box problems but remains challenging in high dimensions. We propose NeST-BO, a curvature-aware local BO method that targets a (modified) Newton step by jointly learning gradient and Hessian information with Gaussian process (GP) surrogates, and selecting evaluations via a one-step lookahead bound on the Newton-step error. We show that this bound contracts with batch size, so NeST-BO drives the step error to zero; in well-behaved neighborhoods it recovers the fast local convergence behavior of inexact/modified Newton methods, while standard safeguards support global convergence to stationary points. To improve scaling with problem dimension, we optimize the acquisition in low-dimensional embedded subspaces (random or learned), reducing the dominant cost of learning curvature from $O(d^2)$ to $O(m^2)$ with $m \ll d$ while preserving step targeting. Across high-dimensional synthetic and real-world problems, including cases with thousands of variables and unknown active subspaces, NeST-BO consistently yields faster convergence and better final values than state-of-the-art local and high-dimensional BO baselines.

1 INTRODUCTION

Bayesian optimization (BO) is a popular framework for optimizing expensive black-box functions because it often needs far fewer evaluations than alternative derivative-free methods (Jones et al., 1998; Frazier, 2018; Garnett, 2023). BO has been applied successfully in automated machine learning (Snoek et al., 2012; Lindauer et al., 2022), prompt optimization for LLMs (Sabbatella et al., 2024), robotics and control (Berkenkamp et al., 2023; Paulson et al., 2023), process optimization (Kudva et al., 2025), materials design (Frazier and Wang, 2015; Tang et al., 2024), and more. However, as dimensionality grows, performance often deteriorates, with recent studies attributing this decline to degeneracies such as vanishing or uninformative gradients in the Gaussian process (GP) surrogate (Williams and Rasmussen, 2006) that make acquisition optimization brittle when length scales are poorly chosen (Papenmeier et al., 2025).

This work develops a curvature-aware *local* BO approach and a practical way to make it scale to very high dimensions. We introduce **NeST-BO** (**Ne**wton-**S**tep-**T**argeted **BO**), which uses the GP surrogate model to *jointly* learn gradient and Hessian information, and chooses new evaluations to shrink a one-step lookahead bound on the Newton-step error. Conceptually, NeST-BO targets the step rather than the derivatives themselves – an approach that we find can learn the Newton direction with fewer samples than, e.g., finite difference methods would require.

NeST-BO is not motivated by the idea that a vanilla Newton methods is a robust *global* solver for nonconvex objectives. Rather, it is designed as a local refinement routine: once we are in a neighborhood where the objective is reasonably smooth and the curvature is informative, modified Newton schemes can provide much faster local convergence than purely first-order updates. In practice, this "hybrid" behavior is typically enforced by standard safeguards (e.g., damping, line search), so that the method behaves like a gradient step when curvature is unreliable, while retaining rapid local convergence when it is helpful (Nocedal and Wright, 2006). Because second-order (Newton-type) methods can be highly effective as *local* accelerators even on large, nonconvex objectives when suitably modified, e.g., (Martens, 2010), we conjecture targeting the step can substantially accelerate local BO.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

An important obstacle is the cost of Hessian-based terms, which grows as $O(d^2)$ with input dimension d. To address this, we instantiate NeST-BO inside lower-dimensional subspaces, using a nested subspace expansion strategy similar to BAxUS (Papenmeier et al., 2022). This collapses the dominant cost to $O(m^2)$ for subspace dimension $m \ll d$ while preserving the benefits of Newton-step targeting. We find that the local Newton step is also naturally robust to the non-stationarity in the mapping from the subspace to the objective function that can be introduced by subspace embeddings, which helps explain why our approach continues to perform well for some problems even as the ambient dimension reaches thousands or more.

Finally, we find that our acquisition includes a scale factor that balances gradient and Hessian learning whose optimal value must be estimated by, e.g., Monte Carlo sampling; our empirical results show that performance is robust to the precise choice of this factor, and we use a simple default that avoids this extra computation. We prove a "vanishing power-function condition" showing that NeST-BO drives the Newton-step error to zero as batch size increases. As a result, the method fits squarely within the classical inexact/modified Newton viewpoint: once the step errors are sufficiently small, one recovers the standard fast local convergence behavior of Newton-type methods, while globalization via, e.g., dampling/line search safeguards yields the usual notion of global convergence to stationary points (Nocedal and Wright, 2006).

In summary, our key contributions are:

- A curvature-aware local BO algorithm that explicitly targets the Newton step via a tractable and theoretically-sound acquisition function.
- A scalable instantiation that runs NeST-BO in enlarging subspaces, reducing computation from $O(d^2)$ to $O(m^2)$, where d and m are the ambient and maximum subspace dimensions.
- Theoretical guarantees showing that NeST-BO drives Newton-step error to zero and thus inherits the convergence behavior of modified Newton methods.
- Extensive empirical results on more than 12 synthetic and real-world problems (ranging from 20d to >7000d), where NeST-BO variants yield large performance improvements over six state-of-the-art high-dimensional BO baselines.

2 RELATED WORK

Linear subspaces and embeddings. A common strategy for high-dimensional BO is to assume the objective varies mainly in a lower-dimensional subspace and reduce model complexity accordingly. REMBO projects the search into a random linear subspace and optimizes there, with guarantees when the effective dimension is small (Wang et al., 2016). ALEBO improves robustness by using a Mahalanobis kernel and linear constraints on the acquisition to respect the original box (Letham et al., 2020). HeSBO replaces dense projections with count-sketch-style *sparse* embeddings that preserve structure with negligible overhead. BAxUS introduces *nested* random subspaces that expand during the run and a mechanism to carry observations across enlargements, providing improved success probabilities and practical robustness (Papenmeier et al., 2022). In this paper, to improve the scalability of our method, we adopt the BAxUS embedding and enlargement schedule but *replace* its trust-region-based optimizer with NeST-BO.

Learning sparse structure. Another interesting strategy for tackling high-dimensional problems is to adaptively learn space substructure. SAASBO places a sparsity-promoting prior on inverse GP length-scales (Eriksson and Jankowiak, 2021). This can be very effective when the active set is small and axis-aligned, but the fully Bayesian inference of the kernel hyperparameters makes the inference cost scale cubically with the number of evaluations, which greatly limit its ability to scale beyond fairly small sampling budgets.

Local BO. Local BO restricts search to neighborhoods around the incumbent to mitigate the curse of dimensionality. TuRBO, a trust-region variant, maintains multiple local regions with adaptive sizes (Eriksson et al., 2019). Another line of work is *directional* local BO, which uses a GP to define a local step rule. GIBO reduces gradient posterior uncertainty and moves along the mean gradient (Müller et al., 2021), while MPD chooses the direction maximizing the posterior probability of descent (Nguyen et al., 2022). MinUCB forgoes direct gradient inference and instead minimizes the upper confidence bound (UCB) objective as a local step. Our approach differs by explicitly *targeting the Newton step*, which uses both gradient and Hessian predictions from the GP.

"Vanilla BO works" in high dimensions. Several recent works show that standard BO can be competitive in high dimensions when properly designed. Hvarfner et al. scale a log-normal length-scale prior with dimension, yielding a strong "D-scaled" LogEI baseline (Hvarfner et al., 2024). Xu et al. argue that poor length-scale initialization induces vanishing gradients in GPs with squared exponential (SE) kernels and show Matérn kernels or robust initialization can

avoid this pathology (Xu et al., 2024). Papenmeier et al. analyze why such settings succeed, attributing gains to low effective dimensionality or benign benchmark structure rather than a general cure for high-dimensional problems (Papenmeier et al., 2025). We include such strong "vanilla" baselines in our experiments to reflect these practices.

3 PRELIMINARIES

3.1 Problem Setup & Bayesian Optimization

We consider the following zeroth-order optimization problem over a d-dimensional space:

$$\boldsymbol{x}^\star \in \operatorname*{argmin}_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}), \qquad \mathcal{X} \subseteq \mathbb{R}^d, \tag{1}$$

where the expensive function f can only be accessed through noisy queries $y = f(\boldsymbol{x})+\epsilon$ with i.i.d. Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$. Bayesian optimization (BO) tackles (1) by fitting a probabilistic surrogate $p(f|\mathcal{D})$ over available data $\mathcal{D}$ and uses it to select new evaluations by maximizing an *acquisition function* $\alpha(\boldsymbol{x}|\mathcal{D})$ that trades off exploration and exploitation. After exhausting the budget, the recommended solution is either the best observed point or the minimizer of the surrogate mean. Many acquisitions have been proposed including expected improvement (EI) (Jones et al., 1998), upper confidence bound (UCB) (Srinivas et al., 2010), knowledge gradient (KG) (Frazier et al., 2008), and entropy-based methods (Hennig and Schuler, 2012). We refer readers to Garnett (2023) for a full tutorial.

3.2 Gaussian Processes and their Derivatives

We use Gaussian processes (GPs) (Williams and Rasmussen, 2006) as surrogates, which is the most popular surrogate model class in BO. A GP prior $f \sim \mathcal{GP}(\mu, k)$ induces a joint Gaussian belief over any finite set of inputs; conditioning on the dataset $\mathcal{D}$ yields a posterior GP $f|\mathcal{D} \sim \mathcal{GP}(\mu_\mathcal{D}, k_\mathcal{D})$ with closed-form posterior mean $\mu_\mathcal{D}$ and covariance (or kernel) function $k_\mathcal{D}$. A key property we repeatedly use in this work is that *derivatives of a GP remain GPs* because differentiation is a linear operator (De Roos et al., 2021). Thus, the gradient $\boldsymbol{g}(\boldsymbol{x}) = \nabla f(\boldsymbol{x})$ and Hessian $\boldsymbol{H}(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x})$ have analytic posterior means and covariances obtained by differentiating the kernel in the appropriate arguments. We collect the explicit formulas for f, $\boldsymbol{g}$, and $\boldsymbol{H}$ posteriors in Appendix A.

3.3 Local BO using Gradients

Learning a globally accurate surrogate becomes data-hungry as d grows because regret bounds for global BO (e.g., GP-UCB) scale exponentially with dimension unless strong structural assumptions hold (Srinivas et al., 2010). Local BO addresses this by focusing search near the incumbent and updating the model with more locally collected data. Gradient-informed methods refine this idea by explicitly selecting evaluations that reduce uncertainty about the local descent direction. A prominent example is the Gradient Information (GI) acquisition that underlies GIBO (Müller et al., 2021). Let $\boldsymbol{x}_t$ be the current iterate and $\boldsymbol{Z} \in \mathbb{R}^{b_t\times d}$ a batch of candidates. GI selects $\boldsymbol{Z}$ to maximally reduce the expected posterior covariance of the gradient at $\boldsymbol{x}_t$ given current data $\mathcal{D}$ after observing a new batch of points $(\boldsymbol{Z}, \boldsymbol{y})$:

$$\begin{aligned}&\alpha_{\mathrm{GI}}(\boldsymbol{Z}|\boldsymbol{x}_t, \mathcal{D}) \\ &= \mathbb{E}_{\boldsymbol{y}|\mathcal{D},\boldsymbol{Z}}\big\{ \operatorname{tr}\big(\Sigma^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}_t)\big) - \operatorname{tr}\big(\Sigma^{\boldsymbol{g}}_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\big)\big\}.\end{aligned} \tag{2}$$

The key observation is, since posterior covariances do not depend on the observed targets $\boldsymbol{y}$, this simplifies to minimizing a (squared) "power function" for the gradient defined as the trace of the posterior covariance:

$$\tilde{\alpha}_{\mathrm{GI}}(\boldsymbol{Z}|\boldsymbol{x}_t, \mathcal{D}) = \operatorname{tr}\big(\Sigma^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t)\big) = \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t). \tag{3}$$

This criterion encourages sampling along directions that most reduce gradient uncertainty, after which the algorithm steps along the GP posterior mean gradient. Theoretical analysis of GIBO showed that the convergence rate to a stationary point scales linearly in d, which is better than rates at which global optimization can find the global optimum (Wu et al., 2023).

Nguyen et al. (Nguyen et al., 2022) revisit the GIBO template and show that the GP posterior mean gradient is not, in general, the direction that maximizes the *posterior probability of descent*, which motivates MPD: an acquisition that targets an upper bound on the one-step maximum descent probability together with updates along the most-probable descent direction. We include MPD as a strong baseline; NeST-BO builds on the same local gradient-learning viewpoint as GIBO but targets a *second-order* Newton step rather than a first-order descent direction.

3.4 Newton's Method for Optimization

Newton's method (NM) is a classical algorithm for (unconstrained) local minimization of twice continuously differentiable objectives. Starting from $\boldsymbol{x}_0 \in \mathbb{R}^d$, it makes the following updates:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma_t \boldsymbol{H}(\boldsymbol{x}_t)^{-1}\boldsymbol{g}(\boldsymbol{x}_t), \qquad t = 0, 1, \ldots, \tag{4}$$

where $\gamma_t > 0$ denotes the step size at iteration t. NM re-scales and rotates the gradient using local curvature. On well-behaved landscapes (e.g., in a neighborhood where $\boldsymbol{H}$ is positive definite), this makes progress

far less sensitive to conditioning than first-order methods and yields *local quadratic* convergence near a (non-degenerate) minimizer, in contrast to the linear or sub-linear rates typical of gradient schemes under comparable assumptions (Nocedal and Wright, 2006).

In the BO context, we do not get to directly observe $\boldsymbol{g}$ or $\boldsymbol{H}$, but their GP posteriors (implicitly) define a distribution over the Newton step $\boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{x})^{-1}\boldsymbol{g}(\boldsymbol{x})$. NeST-BO is built around actively reducing the posterior uncertainty of the step $\boldsymbol{d}(\boldsymbol{x})$, rather than estimating $\boldsymbol{g}(\boldsymbol{x})$ and $\boldsymbol{H}(\boldsymbol{x})$ separately. The thought is that, when the Newton step gets close to a local solution, the iteration advantage of NM can outweigh the per-step cost of learning curvature, yielding stronger sample efficiency than gradient-only methods like GIBO.

4 NEWTON-STEP-TARGETED BAYESIAN OPTIMIZATION

4.1 An Acquisition for the Newton Step

Let $\boldsymbol{x}_t$ be the current iterate and $\boldsymbol{Z} \in \mathbb{R}^{b_t \times d}$ a batch of candidates at which we might evaluate f. Our goal is to learn the Newton step $\boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{x})^{-1}\boldsymbol{g}(\boldsymbol{x})$ at $\boldsymbol{x}_t$ under the GP posterior. Because $\boldsymbol{d}(\boldsymbol{x})$ is non-Gaussian, its posterior covariance is not available in closed form; instead we derive a Reproducing Kernel Hilbert Space (RKHS) error bound (inspired by the results derived in (Wu et al., 2023)). Let the gradient and (vectorized) Hessian (squared) *power functions* at $\boldsymbol{x}$ be denoted by $\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{tr}\left(\Sigma^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\right)$ and $\pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{tr}\left(\Sigma^{\operatorname{vec}(\boldsymbol{H})}_{\mathcal{D}}(\boldsymbol{x})\right)$. Also, let $\mathcal{H}$ be the RKHS on $\mathcal{X}$ equipped with a reproducing kernel $k(\cdot,\cdot)$ and RKHS norm $\|\cdot\|_{\mathcal{H}}$. We can establish the following bound (see Appendix B for the proof):

Theorem 1 (Newton-step error bound)**.** *Let $\varepsilon_{\mathcal{D}}(\boldsymbol{x}) = \|\boldsymbol{d}(\boldsymbol{x}) - \widehat{\boldsymbol{d}}_{\mathcal{D}}(\boldsymbol{x})\|$ denote the Newton-step error at $\boldsymbol{x}$ given data $\mathcal{D}$ with $\widehat{\boldsymbol{d}}_{\mathcal{D}}(\boldsymbol{x}) = \widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})^{-1}\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})$ where $\|\cdot\|$ is the standard Euclidean norm. Assume the kernel k is stationary and four-times differentiable and that $\|\boldsymbol{H}(\boldsymbol{x})^{-1}\|$ exists and is finite. For any $f \in \mathcal{H}$ (from the RKHS with reproducing kernel k) with $\|f\|_{\mathcal{H}} \leq B$, $\boldsymbol{x} \in \mathcal{X}$, and $\mathcal{D}$, the following bound holds:*

$$\varepsilon_{\mathcal{D}}(\boldsymbol{x}) \leq C_{\boldsymbol{x}} \sqrt{\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) + s_{\mathcal{D}}(\boldsymbol{x})\, \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})}, \tag{5}$$

where $s_{\mathcal{D}}(\boldsymbol{x}) = \|\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})^{-1}\|^2\, \|\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\|^2$ is a scale factor that trades off gradient and Hessian information and $C_{\boldsymbol{x}} = \sqrt{2}B\, \|\boldsymbol{H}(\boldsymbol{x})^{-1}\|$ is independent of $\mathcal{D}$.

Motivated by the Newton-step error bound in Theorem 1, our design goal at iteration t is simple: choose a batch $\boldsymbol{Z}$ that will make the *post-batch* bound on the error as small as possible (in expectation) over the unobserved outcomes at $\boldsymbol{Z}$. Concretely, let $B_{\mathcal{D}}(\boldsymbol{x}_t) = \pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}_t) + s_{\mathcal{D}}(\boldsymbol{x}_t)\, \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}_t)$, so that (5) is $\varepsilon_{\mathcal{D}}(\boldsymbol{x}_t) \leq C_{\boldsymbol{x}_t}\sqrt{B_{\mathcal{D}}(\boldsymbol{x}_t)}$. Since $C_{\boldsymbol{x}_t}$ is independent of the data, minimizing an upper bound on the expected step error is naturally achieved by:

$$\boldsymbol{Z}_t \in \operatorname*{argmin}_{\boldsymbol{Z}}\ \mathbb{E}_{\boldsymbol{y}|\mathcal{D},\boldsymbol{Z}}\{B_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\}. \tag{6}$$

This is the direct analogue of GI, which chooses points to minimize a lookahead bound on the gradient error, as shown in (Wu et al., 2023). The objective in (6) is doing the same thing but on a lookahead bound for the error in the Newton step.

The remaining step is to simplify (6). Under a GP with Gaussian observation noise, the posterior *covariances* of derivatives at $\boldsymbol{x}_t$ after conditioning on $\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})$ do not depend on the realized values $\boldsymbol{y}$; they depend only on the locations $\boldsymbol{Z}$, so the only term in (6) that requires a lookahead expectation is $s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)$, which depends on posterior *means* through $\widehat{\boldsymbol{g}}$ and $\widehat{\boldsymbol{H}}$. Using this fact, (6) reduces to our following proposed Newton Step Targeting (NeST) acquisition function:

$$\begin{aligned}&\tilde{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z}|\boldsymbol{x}_t,\mathcal{D}) \\ &= \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t) + \mathbb{E}_{\boldsymbol{y}|\mathcal{D},\boldsymbol{Z}}\left\{s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\right\}\pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t).\end{aligned} \tag{7}$$

NeST is composed of three main terms: (i) $\pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t)$ is the trace of the covariance of the one-step lookahead gradient at $\boldsymbol{x}_t$, (ii) $\pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t)$ is the trace of the covariance of the one-step lookahead vectorized Hessian, and (iii) $\mathbb{E}_{\boldsymbol{y}|\mathcal{D},\boldsymbol{Z}}\left\{s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\right\}$ is a scale factor that weights these two terms. We can interpret (i) and (ii) as measures of the information content about quantities of interest (mainly gradient and Hessian at our current point) and (iii) as a term that adaptively balances gradient and Hessian information. Note that, since $\|\boldsymbol{H}^{-1}(\boldsymbol{x})\boldsymbol{g}(\boldsymbol{x})\| \leq \|\boldsymbol{H}^{-1}(\boldsymbol{x})\|\|\boldsymbol{g}(\boldsymbol{x})\|$, we can also think of this term as a conservative proxy for the squared step length and thus upweights Hessian information when NM may take large steps.

A practical family. The expectation over the scale factor in (7) does not have a simple closed-form expression and thus would need to be estimated using Monte Carlo (MC) sampling in practice (see Appendix C.1 for details). This is mainly due to the fact that $s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}$ is a nonlinear function of $\boldsymbol{y}$ and thus non-Gaussian in general. Through some empirical testing shown in Appendix C, we observed relatively low sensitivity in performance to the precise scale factor, which motivates the following computationally cheaper acquisition:

$$\widehat{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z}|\boldsymbol{x}_t,\mathcal{D},\widehat{s}_t) = \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t), \tag{8}$$

with a pre-determined $\widehat{s}_t > 0$ that is independent of $\boldsymbol{Z}$. This recovers GI when $\widehat{s}_t = 0$ and focuses more on

Algorithm 1 NeST-BO

Inputs: initial iterate $\boldsymbol{x}_0 \in \mathbb{R}^d$; initial data $\mathcal{D}_0$; batch sizes $\{b_t\}$; step sizes $\{\gamma_t\}$; scale factors $\{\widehat{s}_t\}$; GP hyperparameters; and total number of iterations T.

Fit GP surrogate on $\mathcal{D}_0$.
for $t = 0, \ldots, T-1$ **do**
  $\boldsymbol{X}_t \in \operatorname{argmin}_{\boldsymbol{Z} \in \mathbb{R}^{b_t \times d}} \widehat{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z} | \boldsymbol{x}_t, \mathcal{D}_t, \widehat{s}_t)$.
  Evaluate f at $\boldsymbol{X}_t$ to obtain $\boldsymbol{y}_t$.
  Augment dataset $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup (\boldsymbol{X}_t, \boldsymbol{y}_t)$.
  Update GP posterior with $\mathcal{D}_{t+1}$.
  Compute $\widehat{\boldsymbol{g}} \leftarrow \widehat{\boldsymbol{g}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ and $\widehat{\boldsymbol{H}} \leftarrow \widehat{\boldsymbol{H}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$.
  Solve $\widehat{\boldsymbol{H}}\boldsymbol{v} = \widehat{\boldsymbol{g}}$ for $\widehat{\boldsymbol{d}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t) \leftarrow \boldsymbol{v}$.
  Update iterate: $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma_t \widehat{\boldsymbol{d}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$.
end for

curvature as $\widehat{s}_t$ increases. In Section 4.4, we show the NeST samples can drive the bound in (5) to zero as the batch size increases for any choice of scale factor.

4.2 The NeST-BO Algorithm

Algorithm 1 summarizes the simple version of the loop. At iterate $\boldsymbol{x}_t$, choose a batch $\boldsymbol{X}_t$ by minimizing (8) at $\boldsymbol{x}_t$; update the GP with the new observations; then move along the predicted Newton step with some step size. Note there are a number of practical implementation details, which we discuss more in the next section.

Since we are attempting to learn both the gradient and Hessian at $\boldsymbol{x}_t$, one can in fact easily adapt this algorithm to use a step direction from any form of, e.g., damped NM (Hanzely et al., 2022) or NM with line search (Shea and Schmidt, 2025). We do not attempt to systematically compare different options in this work; instead we go with a standard "hybrid" implementation that uses Newton steps when they well-behaved and otherwise revert to gradient steps.

We provide an illustration and visual comparison of NeST-BO (top) to GIBO (bottom) in Figure 1, which demonstrates the advantages of simultaneously learning gradient and Hessian (curvature) information.

4.3 Implementation Details

Algorithm 1 describes an idealized NeST-BO loop used for exposition. In practice, just as vanilla NM can be brittle without safeguards, we found it helpful to use a slightly "hardened" variant for all experiments. The key point is that NeST-BO's value is in *where* it samples at each iteration (i.e., producing high-quality local gradient/Hessian information), so we can borrow standard Newton-style safeguards without changing the core idea. For completeness, the full pseudocode for our practical implementation is given in Appendix D. We summarize the main choices here.

GP hyperparameters. In most applications, kernel hyperparameters are not known *a priori*. We thus need to fit the GP by maximizing the marginal log likelihood, which is standard in the BO literature. Our method does not rely on this specific estimator; any hyperparameter learning procedure can be used.

Moving direction. NeST-BO is meant to behave like a local Newton-type method once it enters a well-behaved neighborhood. When $\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}_t)$ is positive definite (and not too ill-conditioned), this is a descent direction for the GP mean and typically behaves as expected. If $\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}_t)$ is indefinite or numerically unstable to invert, we interpret this as a signal that we are not yet in a regime where Newton steps are reliable, and we revert to a length-scale-normalized gradient step (Müller et al., 2021, Appendix A.4). This is a simple and robust fallback; many other Newton safeguards (e.g., damping/regularization or trust-region updates) could also be used, but we did not attempt to exhaustively test (or internally tune) these variants, which would be valuable future work.

Batch versus sequential selection. Although (8) supports joint batch optimization, we select points *greedily*. After choosing each point, we update the posterior covariance deterministically (note that the power functions do not depend on the realized outcomes), and then re-optimize to choose the next point. This replaces a single $b_t d$-dimensional search with b_t separate d-dimensional searches and naturally discourages near-duplicate selections, since conditioning on a chosen location reduces posterior variance around it. In our preliminary tests, greedy selection closely matched joint optimization.

Step size. We use a standard Armijo backtracking line search on the GP mean $\mu_{\mathcal{D}}$ along the chosen direction. This gives a lightweight safeguard against overly aggressive steps without requiring additional function evaluations, and we cap the number of backtracking iterations using standard defaults (Bertsekas, 2016, Chapter 1). There has been recent work on greedy NM (Shea and Schmidt, 2025), which suggests it can be advantageous to allow for step sizes larger than 1, but we do not consider that here.

Scale factor. The scale $\widehat{s}_t$ in (8) controls how strongly (approximate) NeST emphasizes curvature (Hessian) uncertainty relative to gradient uncertainty. For the main experiments, we use a simple deterministic proxy $\widehat{s}_t = 1$, which performed reliably across problems and dimensions. Appendix C.2 provides a

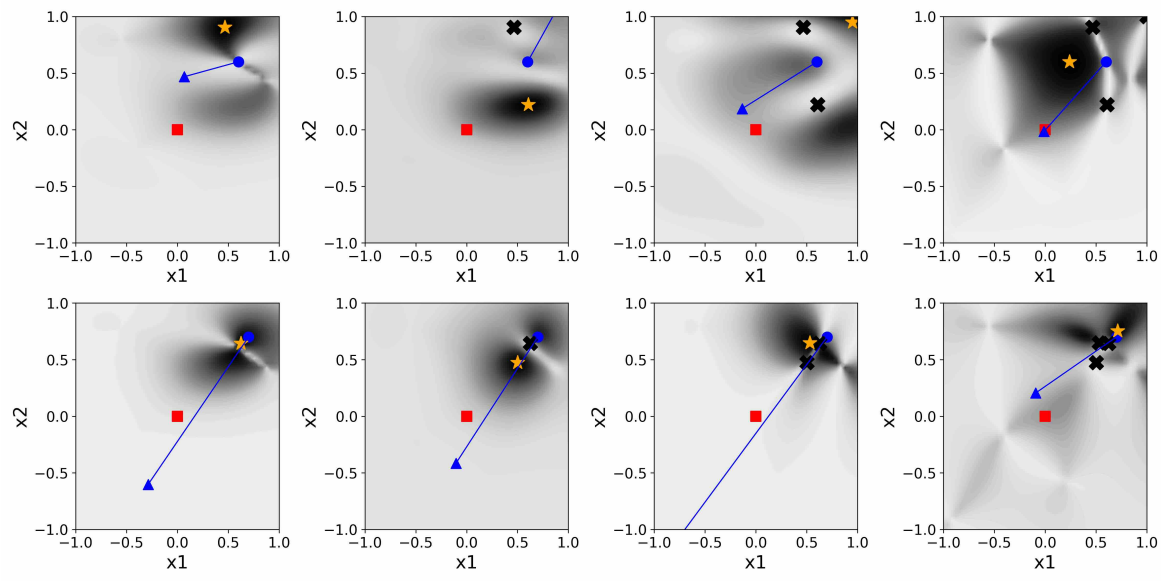

Figure 1: **Top:** NeST-BO's acquisition $\tilde{\alpha}_{\mathrm{NeST}}$; **Bottom:** GIBO's acquisition $\tilde{\alpha}_{\mathrm{GI}}$ on a 2D test function at iterate $\boldsymbol{x}_t$ (blue circle). Darker background indicates larger acquisition value. Red square: location of the true Newton step. Blue triangle: GP-predicted Newton step. Orange star: acquisition minimizer. Black crosses: Past evaluation points. NeST-BO places samples away from $\boldsymbol{x}_t$ along directions informative for curvature, rapidly shrinking the Newton-step error bound; GI tends to oversample near $\boldsymbol{x}_t$, slowing curvature identification.

broader comparison over $\widehat{s}_t$ choices. We also tested a "plug-in" variant $\widehat{s}_t = s_{\mathcal{D}}(\boldsymbol{x}_t)$ that neglects the lookahead dependence on $(\boldsymbol{Z}, \boldsymbol{y})$ (Appendix C.3); performance was very similar between these options, so we default to $\widehat{s}_t = 1$ for simplicity.

4.4 Theoretical Analysis

Let $\varepsilon_t = \varepsilon_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ denote the Newton-step error after iteration t. By Theorem 1, ε_t is controlled by posterior uncertainty in the *local derivatives* at $\boldsymbol{x}_t$; specifically, the power functions appearing in (5). This motivates a simple designability condition:

Vanishing power-function condition (VPC). The NeST design objective $\pi^{\boldsymbol{g}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ can be made arbitrarily small as b_t increases.

VPC is not an abstract assumption on ε_t; it ties the step error to *concrete, design-controllable* posterior covariances. A key subtlety is that NeST-BO cannot directly minimize the right-hand side of (5) because the scale term $s_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ is only revealed *after* the batch is observed. Nevertheless, the bound applies to the realized step error under *any* sampling rule, and VPC can be verified constructively for NeST. We state the main idea informally below; the complete statement and proof are in Appendix E.1.

Theorem 2 (VPC under NeST sampling; informal)**.** *Fix an iteration t and the current iterate $\boldsymbol{x}_t$. For a batch $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{b_t}) \in \mathcal{X}^{b_t}$, let $\Phi_t(\boldsymbol{Z}) = \widehat{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z}|\boldsymbol{x}_t, \mathcal{D}, \widehat{s}_t)$ be short for the approximate NeST acquisition function with $\widehat{s}_t > 0$. Under mild domain and kernel regularity conditions, the following hold.*

Let $b^\star = d^2 + d + 1$. Then, in the noiseless setting, the optimal NeST design value can be driven to zero:

$$\inf_{\boldsymbol{Z} \in \mathcal{X}^{b_t}} \Phi_t(\boldsymbol{Z}) = 0 \text{ (as an infimum) for all } b_t \geq b^\star.$$

Equivalently, for every $\epsilon > 0$ and every $b_t \geq b^\star$, there exists a batch $\boldsymbol{Z}_\epsilon$ such that $\Phi_t(\boldsymbol{Z}_\epsilon) \leq \epsilon$. Moreover, one explicit candidate design achieving this is a symmetric $b^\star$-point centered-difference stencil at radius h, for which $\Phi_t(\boldsymbol{Z}_h) \lesssim h^4$ as $h \to 0$.

With i.i.d. Gaussian noise of variance σ^2, using m replicates per location is equivalent to observing averaged responses with variance σ^2/m. Thus, for every $\epsilon > 0$ there exist h and m such that the corresponding replicated design satisfies $\Phi_t(\boldsymbol{Z}_h^{(m)}) \leq \epsilon$.

An important consequence is that, once NeST-BO is operating in a neighborhood where NM is well-conditioned, VPC implies $\varepsilon_t \to 0$ as the batch size increases, and NeST-BO asymptotically inherits the *local quadratic* convergence rate of NM (full statement and proof in Appendix E.2). Finally, the VPC statement itself does not depend on the particular choice of scale factor beyond requiring $\widehat{s}_t > 0$. Thus, the simplified acquisition in (8) retains the same asymp-

totic guarantee, with the caveat that the argument is inherently for large enough batches.

4.5 Scaling NeST-BO via Subspaces

The main computational bottleneck for NeST-BO in high ambient dimension d is the Hessian term $\pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t)$, which scales quadratically in d. We propose to address this by instantiating NeST-BO *inside* embedded subspaces $\boldsymbol{v} \in \mathbb{R}^m$ of size $m \ll d$. We adopt the nested, sparse random-embedding scheme of BAxUS – bins of input coordinates are hashed into m target coordinates with random signs, and the target dimension is increased over time by *splitting* bins while retaining observations across embeddings (Papenmeier et al., 2022). This approach preserves past data under splits and increases the probability that the subspace contains an optimizer as m grows.

We run NeST-BO in the subspace $\boldsymbol{v}$, map the candidates back to the ambient dimension $\boldsymbol{x} = \boldsymbol{S}^\top \boldsymbol{v}$ (where $\boldsymbol{S}^\top \in \mathbb{R}^{d\times m}$ is the projection matrix) for evaluation, and update the GP in the embedded subspace coordinates. This reduces the per-candidate curvature cost from $O(d^2)$ to $O(m^2)$ while preserving the step-targeting principle. Compared to the original BAxUS algorithm, which couples its embedding with a variant of TuRBO, our version replaces TuRBO with NeST-BO; in problems where curvature matters, we find that this can can substantially accelerate optimization progress (see results in Section 5).

Lastly, note that our theoretical results related to VPC are proved only for NeST-BO in the *original* d-dimensional space, and they do not directly extend to the subspace variant. Intuitively, the VPC proof relies on constructing d-dimensional designs that can drive the full gradient and Hessian power functions at $\boldsymbol{x}_t$ to zero, whereas a fixed low-dimensional embedding only controls derivative information within the embedded coordinates and may not capture curvature directions relevant in $\mathbb{R}^d$. The BAxUS-style embedding should thus be viewed as a practical scalability heuristic that leverages the NeST sampling principle.

5 RESULTS

We now benchmark NeST-BO and its subspace variant, labeled NeST-BO-sub, against strong local and global BO baselines: TuRBO (Eriksson et al., 2019), GIBO (Müller et al., 2021), MPD (Nguyen et al., 2022), MinUCB (Fan et al., 2024), BAxUS (Papenmeier et al., 2022), and a "vanilla" GP-BO configured with dimensionally calibrated priors and LogEI (we label this *D-scaled LogEI*) (Hvarfner et al., 2024). We also include Sobol sampling as a non-model baseline.

Unless a global minimizer is known, we report the *minimum observed value*; otherwise we show *simple regret* (in log scale). Curves display the median across 10 independent replicates with ± one standard error as the shaded band. Implementation details (e.g., models, hyperparameter updates, acquisition optimization, and the precise evaluation budgets for each task) are provided in Appendix F. In short, we used a common SE kernel across methods and standard GP training and acquisition optimizers from `BoTorch` (Balandat et al., 2020); hyperparameters and method-specific settings follow prior work and are held consistent across tasks to keep comparisons fair. Due to space limitations, additional ablations and diagnostics appear in Appendix G. The code is available on GitHub: `https://github.com/PaulsonLab/NeST-BO`.

5.1 Synthetic Test Functions

We consider two regimes relevant to our method. *Moderate dimension* ($d = 20$): Sphere, Griewank, and Ackley. *High dimension with sparse structure* ($d = 1000$ with $d_{\text{eff}} = 30$ relevant variables): Griewank, Ackley, and Rosenbrock; the remaining coordinates are dummies and the algorithms are not told which ones are active. These are commonly chosen test problems in the BO literature; formal definitions are given in Appendix F.3. Figure 2 (first six panels in the top two rows) summarizes the results.

On the $d = 20$ problems, NeST-BO consistently matches or outperforms all non-subspace baselines. A common pattern is an initial period with slower progress when far from a minimizer (before curvature is accurately estimated) followed by a steep drop once the iterate enters a well-behaved neighborhood. As several steps accumulate, the estimated Newton step better aligns with the true local geometry, further accelerating progress – consistent with our Newton-step error bound, which tightens as the gradient and Hessian power functions shrink. Subspace methods start from stronger initial values by design (they restrict the initial design and acquisitions), but NeST-BO-sub ultimately achieves the lowest regret and, notably, improves over BAxUS across all three problems. These results indicate that *how* one moves inside a subspace matters, i.e., curvature-aware Newton updates can be more effective than trust-region moves, even when both operate in the same embedding space.

On the $d = 1000$ sparse suite, subspaces are essential. Methods that attempt to learn gradients and/or Hessians in the *ambient* space require $O(d)$ queries per iteration and cannot meaningfully progress under our fixed budgets (e.g., 200 evaluations), so we do not include them here. NeST-BO-sub clearly dominates BAxUS, D-scaled LogEI, and TuRBO, achiev-

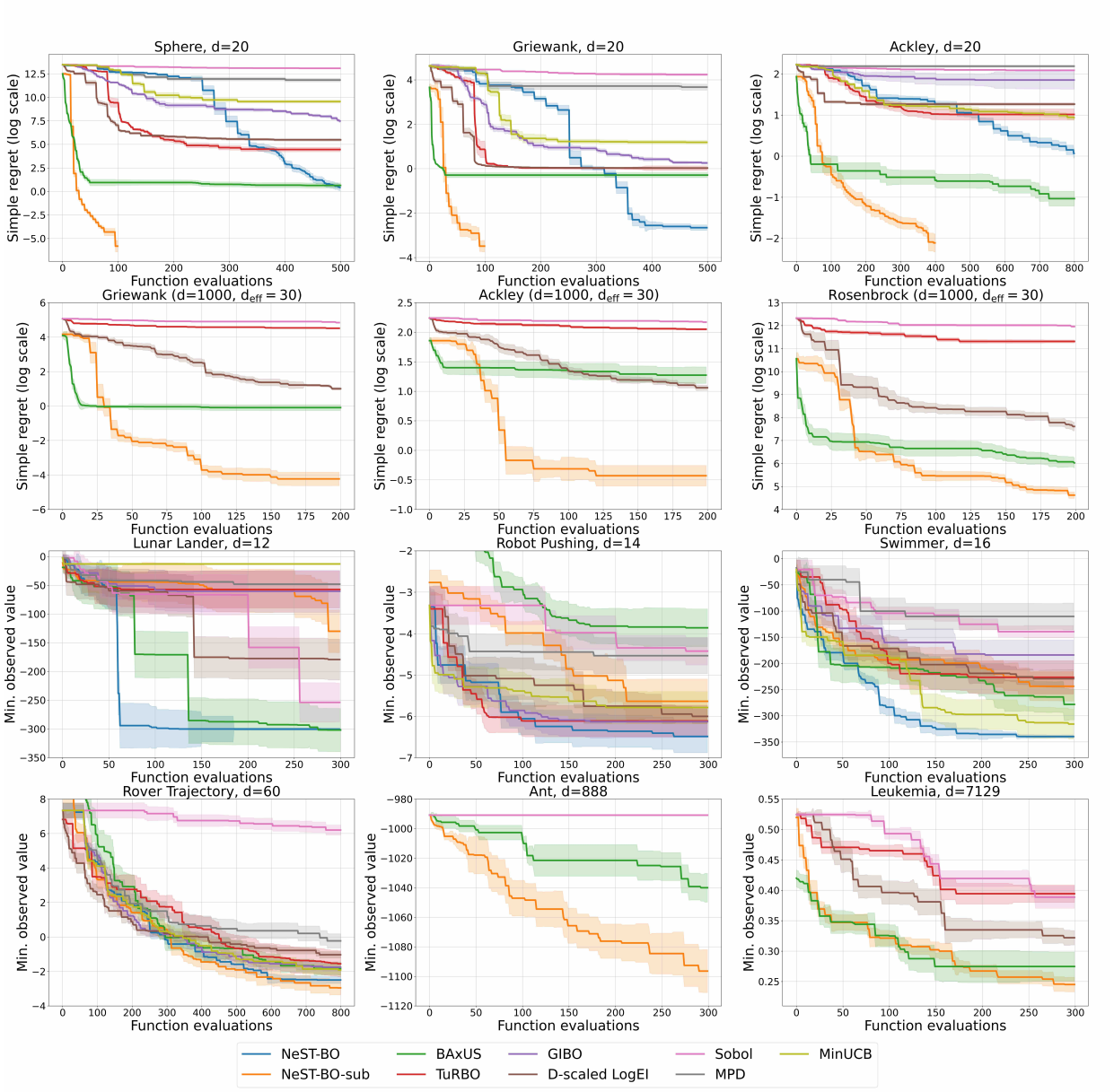

Figure 2: Summary of performance versus evaluations for all synthetic and real-world problems and all methods. Each panel shows either simple regret (log scale; when the global minimizer is known) or the minimum observed value (otherwise). Curves are medians across 10 runs; shading is ± one standard error. Top two rows: synthetic problems (20d and 1000d with 30 active variables). Bottom two rows: real-world tasks (control, planning, and high-dimensional model selection). See Appendix F for the full protocol and Appendix G for extended studies.

ing substantially lower regret on both Ackley and Griewank. TuRBO's trust-region strategy is disadvantaged in very high dimensions, where large local diameters push pairwise distances into regimes that degrade GP fit and acquisition gradients; in contrast, NeST-BO-sub converts a handful of targeted samples near the iterate into accurate Newton steps inside the selected subspace (that can expand as iterations proceed), which drives fast local improvement.

5.2 Mid- to High-Dim. Real-World Tasks

We evaluate six real-world benchmarks spanning reinforcement learning (RL) control, robotic planning, and large-scale hyperparameter tuning. **Control:** Lunar Lander ($d = 12$) and Swimmer ($d = 16$) from OpenAI Gymnasium; the objective is the negative episodic return (reward sign flipped) (Towers et al., 2024). **Planning:** Robot Pushing ($d = 14$) and Rover

Trajectory ($d = 60$) from Wang et al. (2018); Eriksson et al. (2019), both optimized as negative reward. **Very high-dimension:** Ant ($d = 888$) – a MuJoCo quadruped with 8-dimensional action space and 111-dimensional observations, yielding a linear state-feedback policy with 888 parameters – and Leukemia ($d = 7129$), a weighted Lasso problem with 7129 hyperparameters from LassoBench (Šehić et al., 2022). Task definitions, bounds, and initialization protocols are summarized in Appendix F.4. Figure 2 (last six panels in bottom two rows) reports median performance with ± one standard error bands.

In the mid-dimensional group ($d \leq 60$), NeST-BO is consistently state of the art or competitive, achieving the lowest average value in the fewest iterations for Lunar Lander, Robot Pushing, and Swimmer. On Rover Trajectory, NeST-BO-sub edges out NeST-BO, aligning with the intuition that embeddings can capture useful structure as dimensionality and coupling grow. On Lunar Lander and Swimmer, however, subspaces can slightly hurt performance, suggesting most coordinates contribute and the ambient space is preferable.

In the very high-dimensional group, NeST-BO-sub is again the best-performing method. On Ant ($d = 888$), D-scaled LogEI and TuRBO show little-to-no improvement over their starting values, while BAxUS improves but plateaus well above NeST-BO-sub. The Ant landscape is both non-stationary and ill-conditioned; length-scale calibration alone (as in D-scaled LogEI) can over-smooth such objectives, and trust-region steps struggle to adapt their geometry. On Leukemia ($d = 7129$), NeST-BO-sub continues to improve throughout the budget and achieves the best final objective, whereas other methods stagnate early.

5.3 Other Methods in Same Subspace

A natural question raised by our subspace results in Figure 2 is whether NeST-BO's gains are primarily due to the embedding, or whether targeting the local Newton step continues to matter once we restrict the search to lower-dimensional subspaces. To isolate these effects, we compare NeST-BO-sub to two strong baselines that use *the same* subspace machinery: (i) GIBO-sub, i.e., GIBO run in the subspace using its length-scale-normalized gradient step (Müller et al., 2021) and (ii) D-scaled LogEI-sub, i.e., standard LogEI run in the subspace using the GP prior from (Hvarfner et al., 2024). For context, we also include BAxUS. Figure 3 shows the embedding is *not* the whole story. NeST-BO-sub drives regret down by *several orders of magnitude* relative to GIBO-sub and D-scaled LogEI-sub (the latter two plateau near $\sim 10^0$ on the log scale), while NeST-BO-sub continues improving throughout the budget. On Ackley, the same pattern holds: NeST-BO-sub reaches substantially lower regret and converges faster, while the other subspace methods level off earlier.

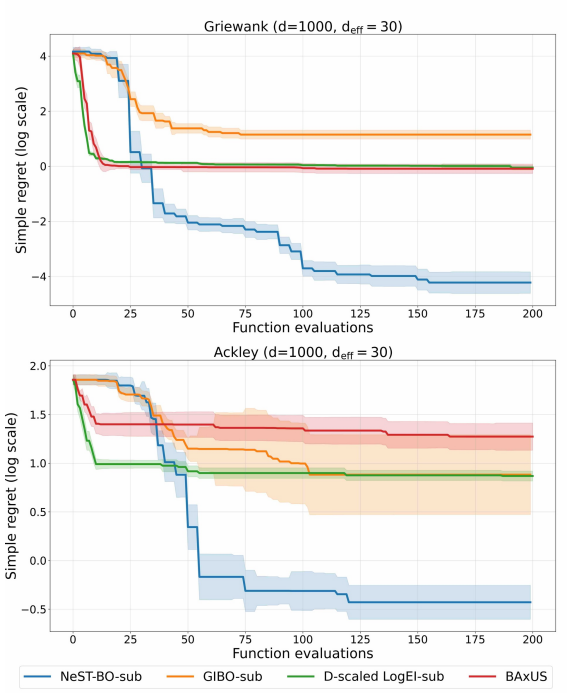

Figure 3: Optimization of 1000-dimensional Griewank and Ackley with 30 active variables. All -sub variants operate using the same BAxUS-style embedding approach. Median simple regret (log scale) with ± one standard error shading across 10 runs.

6 CONCLUSIONS

This work presents NeST-BO, a curvature-aware local Bayesian optimization (BO) method that selects samples to shrink a one-step lookahead bound on Newton-step error and then moves with a damped Newton update. Theoretical analysis establishes a vanishing power-function condition that holds under NeST-BO sampling, implying the algorithm inherits (inexact) Newton guarantees while our experiments show consistent gains over state-of-the-art local and high-dimensional BO baselines on synthetic and real-world problems, including tasks with several thousands of variables (when combined with a subspace variant to enhance scalability). Looking ahead, two promising directions for future research include investigating improved subspace embedding strategies and improving numerical efficiency of the acquisition optimization by better exploiting kernel structure and sparsity for derivative-aware GPs.

References

S. Ament, S. Daulton, D. Eriksson, M. Balandat, and E. Bakshy. Unexpected improvements to expected improvement for bayesian optimization. *Advances in Neural Information Processing Systems*, 36:20577–20612, 2023.

M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

F. Berkenkamp, A. Krause, and A. P. Schoellig. Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics. *Machine Learning*, 112(10):3713–3747, 2023.

D. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 3rd edition, 2016.

F. De Roos, A. Gessner, and P. Hennig. High-dimensional Gaussian process inference with derivatives. In *International Conference on Machine Learning*, pages 2535–2545. PMLR, 2021.

D. Eriksson and M. Jankowiak. High-dimensional bayesian optimization with sparse axis-aligned subspaces. In *Uncertainty in Artificial Intelligence*, pages 493–503. PMLR, 2021.

D. Eriksson, M. Pearce, J. Gardner, R. D. Turner, and M. Poloczek. Scalable global optimization via local Bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Z. Fan, W. Wang, S. H. Ng, and Q. Hu. Minimizing ucb: a better local search strategy in local Bayesian optimization. *Advances in Neural Information Processing Systems*, 37:130602–130634, 2024.

W. J. Firey. Remainder formulae in Taylor's theorem. *The American Mathematical Monthly*, 67(9): 903–905, 1960.

P. I. Frazier. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

P. I. Frazier and J. Wang. Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pages 45–75. Springer, 2015.

P. I. Frazier, W. B. Powell, and S. Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31, 2018.

R. Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

S. Hanzely, D. Kamzolov, D. Pasechnyuk, A. Gasnikov, P. Richtárik, and M. Takác. A damped newton method achieves global $\mathcal{O}(1/k^2)$ and local quadratic convergence rate. *Advances in Neural Information Processing Systems*, 35:25320–25334, 2022.

P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research*, 13(1):1809–1837, 2012.

C. Hvarfner, E. O. Hellsten, and L. Nardi. Vanilla Bayesian optimization performs great in high dimensions. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20793–20817. PMLR, 21–27 Jul 2024.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4): 455–492, 1998.

M. Kanagawa, P. Hennig, D. Sejdinovic, and B. K. Sriperumbudur. Gaussian processes and reproducing kernels: Connections and equivalences. *arXiv preprint arXiv:2506.17366*, 2025.

A. Kudva, W.-T. Tang, and J. A. Paulson. Multi-objective bayesian optimization for networked black-box systems: A path to greener profits and smarter designs. *arXiv preprint arXiv:2502.14121*, 2025.

B. Letham, R. Calandra, A. Rai, and E. Bakshy. Re-examining linear embeddings for high-dimensional Bayesian optimization. *Advances in Neural Information Processing Systems*, 33:1546–1558, 2020.

M. Lindauer, K. Eggensperger, M. Feurer, A. Biedenkapp, D. Deng, C. Benjamins, T. Ruhkopf, R. Sass, and F. Hutter. SMAC3: A versatile Bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research*, 23(54):1–9, 2022.

H. Mania, A. Guy, and B. Recht. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

J. Martens. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 735–742, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

S. Müller, A. von Rohr, and S. Trimpe. Local policy search with Bayesian optimization. *Advances in Neural Information Processing Systems*, 34:20708–20720, 2021.

Y. Nesterov and B. T. Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

Q. Nguyen, K. Wu, J. Gardner, and R. Garnett. Local Bayesian optimization via maximizing probability of descent. *Advances in neural information processing systems*, 35:13190–13202, 2022.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

L. Papenmeier, L. Nardi, and M. Poloczek. Increasing the scope as you learn: Adaptive Bayesian optimization in nested subspaces. *Advances in Neural Information Processing Systems*, 35:11586–11601, 2022.

L. Papenmeier, M. Poloczek, and L. Nardi. Understanding high-dimensional Bayesian optimization. *arXiv preprint arXiv:2502.09198*, 2025.

J. A. Paulson, F. Sorourifar, and A. Mesbah. A tutorial on derivative-free policy learning methods for interpretable controller representations. In *Proceedings of the American Control Conference*. IEEE, 2023.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems*, 20, 2007.

A. Sabbatella, A. Ponti, I. Giordani, and F. Archetti. A Bayesian approach for prompt optimization in LLMs. In *International Conference on Learning and Intelligent Optimization*, pages 348–360. Springer, 2024.

K. Šehić, A. Gramfort, J. Salmon, and L. Nardi. Lassobench: A high-dimensional hyperparameter optimization benchmark suite for lasso. In *International Conference on Automated Machine Learning*, pages 2–1. PMLR, 2022.

B. Shea and M. Schmidt. Greedy Newton: Newton's method with exact line search. *Optimization Letters*, pages 1–19, 2025.

A. E. Siemenn, Z. Ren, Q. Li, and T. Buonassisi. Fast Bayesian optimization of needle-in-a-haystack problems using zooming memory-based initialization (ZoMBI). *npj Computational Materials*, 9(1): 79, 2023.

J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1015–1022. Omnipress, 2010.

W.-T. Tang, A. Chakrabarty, and J. A. Paulson. BEACON: A Bayesian optimization strategy for novelty search in expensive black-box systems. *arXiv preprint arXiv:2406.03616*, 2024.

M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis. Gymnasium: A standard interface for reinforcement learning environments, 2024. URL `https://arxiv.org/abs/2407.17032`.

Z. Wang, F. Hutter, M. Zoghi, D. Matheson, and N. De Feitas. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, 2016.

Z. Wang, C. Gehring, P. Kohli, and S. Jegelka. Batched large-scale Bayesian optimization in high-dimensional spaces. In *International Conference on Artificial Intelligence and Statistics*, pages 745–754. PMLR, 2018.

H. Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

C. K. Williams and C. E. Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT Press, Cambridge, MA, 2006.

R. F. Woolson. Wilcoxon signed-rank test. *Wiley Encyclopedia of Clinical Trials*, pages 1–3, 2007.

K. Wu, K. Kim, R. Garnett, and J. Gardner. The behavior and convergence of local Bayesian optimization. *Advances in Neural Information Processing Systems*, 36:73497–73523, 2023.

Z. Xu, H. Wang, J. M. Phillips, and S. Zhe. Standard Gaussian process is all you need for high-dimensional Bayesian optimization. In *The Thirteenth International Conference on Learning Representations*, 2024.

C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on mathematical software (TOMS)*, 23 (4):550–560, 1997.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. [Yes]

(b) Complete proofs of all theoretical results. [Yes]

(c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. [Yes]

(b) The license information of the assets, if applicable. [Not Applicable]

(c) New assets either in the supplemental material or as a URL, if applicable. [Yes]

(d) Information about consent from data providers/curators. [Not Applicable]

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. [Not Applicable]

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Material

A GAUSSIAN PROCESS DERIVATIVE EXPRESSIONS

A.1 Notation Summary

Let $\mathcal{X} \subset \mathbb{R}^d$ be a convex, compact domain and let $\mathcal{H}$ be a reproducing kernel Hilbert space (RKHS) on $\mathcal{X}$ with reproducing kernel $k(\cdot,\cdot)$, inner product $\langle \cdot,\cdot \rangle_{\mathcal{H}}$, and norm $\| \cdot \|_{\mathcal{H}}$.

For the objective $f : \mathcal{X} \to \mathbb{R}$, denote the gradient and Hessian at $\boldsymbol{x} \in \mathcal{X}$ by

$$\boldsymbol{g}(\boldsymbol{x}) = \nabla f(\boldsymbol{x}) \in \mathbb{R}^d, \qquad \boldsymbol{H}(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x}) \in \mathbb{R}^{d \times d}.$$

A Gaussian process (GP) prior $\mathcal{GP}(\mu, k)$ is specified by mean μ and covariance k functions. Conditioning on data $\mathcal{D} = \{(\boldsymbol{X}, \boldsymbol{y})\}$ for $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{y} \in \mathbb{R}^n$ yields a posterior GP $f|\mathcal{D} \sim \mathcal{GP}(\mu_{\mathcal{D}}, k_{\mathcal{D}})$. Since differentiation is linear, ∇f and $\nabla^2 f$ are also GPs under the same conditioning. We use the shorthand

$$\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) = \mathbb{E}\{\boldsymbol{g}(\boldsymbol{x}) \mid \mathcal{D}\} \in \mathbb{R}^d, \quad \Sigma^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{cov}\{\boldsymbol{g}(\boldsymbol{x}) \mid \mathcal{D}\} \in \mathbb{R}^{d \times d},$$

$$\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}) = \mathbb{E}\{\boldsymbol{H}(\boldsymbol{x}) \mid \mathcal{D}\} \in \mathbb{R}^{d \times d}, \quad \Sigma^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{cov}\{\operatorname{vec}(\boldsymbol{H}(\boldsymbol{x})) \mid \mathcal{D}\} \in \mathbb{R}^{d^2 \times d^2}.$$

We write $\| \cdot \|$ for the Euclidean norm (vectors) and the induced operator 2-norm (matrices).

A.2 Posterior GP under Linear Operators

GPs $\mathcal{GP}(\mu, k)$ are closed under linear operators. For linear operators $\mathcal{L}, \mathcal{M}$ and $f \sim \mathcal{GP}(\mu, k)$,

$$\begin{bmatrix} \mathcal{L}f \\ \mathcal{M}f \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} \mathcal{L}\mu \\ \mathcal{M}\mu \end{bmatrix}, \begin{bmatrix} \mathcal{L}k\,\mathcal{L}' & \mathcal{L}k\,\mathcal{M}' \\ \mathcal{M}k\,\mathcal{L}' & \mathcal{M}k\,\mathcal{M}' \end{bmatrix} \right),$$

where $\mathcal{L}'$ and $\mathcal{M}'$ act on the second kernel argument as adjoints of $\mathcal{L}$ and $\mathcal{M}$. We condition on noisy function observations at $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, $\boldsymbol{y} = f(\boldsymbol{X}) + \boldsymbol{\varepsilon}$, $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$, take $\mathcal{M} = \mathrm{Id}$ at $\boldsymbol{X}$, and evaluate $\mathcal{L} \in \{\mathrm{Id}, \nabla, \nabla^2\}$ at a test point $\boldsymbol{x}$. Then, for any $\mathcal{L}$, the posterior $\mathcal{L}f(\boldsymbol{x}) \mid \mathcal{D} \sim \mathcal{N}(\mu^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}),\, \Sigma^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}))$ with

$$\mu^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}) = \mathcal{L}\mu(\boldsymbol{x}) + \mathcal{L}k(\boldsymbol{x}, \boldsymbol{X}) \left(k(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}\right)^{-1} \big(\boldsymbol{y} - \mu(\boldsymbol{X})\big), \tag{A.1a}$$

$$k^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}') = \mathcal{L}k(\boldsymbol{x}, \boldsymbol{x}')\, \mathcal{L}' - \mathcal{L}k(\boldsymbol{x}, \boldsymbol{X}) \left(k(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}\right)^{-1} k(\boldsymbol{X}, \boldsymbol{x}')\, \mathcal{L}', \tag{A.1b}$$

$$\Sigma^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}) = k^{\mathcal{L}}_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}')\big|_{\boldsymbol{x}'=\boldsymbol{x}}. \tag{A.1c}$$

Let $\boldsymbol{K}_{\boldsymbol{X}\boldsymbol{X}} = k(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}$ and $\boldsymbol{\alpha} = \boldsymbol{K}^{-1}_{\boldsymbol{X}\boldsymbol{X}} \big(\boldsymbol{y} - \mu(\boldsymbol{X})\big)$; these can be precomputed from the prior.

A.3 Power Functions for Gradient and Hessian

We quantify posterior uncertainty via the (squared) *power functions* (traces of derivative covariances) at $\boldsymbol{x}$:

$$\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{tr}\left(\Sigma^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\right) = \sum_{i=1}^{d} \left. \frac{\partial^2 k_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i\, \partial x'_i} \right|_{\boldsymbol{x}'=\boldsymbol{x}}, \qquad \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}) = \operatorname{tr}\left(\Sigma^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})\right) = \sum_{i=1}^{d} \sum_{j=1}^{d} \left. \frac{\partial^4 k_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i\, \partial x_j\, \partial x'_i\, \partial x'_j} \right|_{\boldsymbol{x}'=\boldsymbol{x}}.$$

Using (A.1), these admit closed forms:

$$\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) = \sum_{i=1}^{d} \left[\frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i \partial x'_i} - \frac{\partial k(\boldsymbol{x}, \boldsymbol{X})}{\partial x_i}\, \boldsymbol{K}^{-1}_{\boldsymbol{X}\boldsymbol{X}}\, \frac{\partial k(\boldsymbol{X}, \boldsymbol{x}')}{\partial x'_i} \right]_{\boldsymbol{x}'=\boldsymbol{x}}, \tag{A.2a}$$

$$\pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}) = \sum_{i=1}^{d} \sum_{j=1}^{d} \left[\frac{\partial^4 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i \partial x_j \partial x'_i \partial x'_j} - \frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{X})}{\partial x_i \partial x_j}\, \boldsymbol{K}^{-1}_{\boldsymbol{X}\boldsymbol{X}}\, \frac{\partial^2 k(\boldsymbol{X}, \boldsymbol{x}')}{\partial x'_i \partial x'_j} \right]_{\boldsymbol{x}'=\boldsymbol{x}}. \tag{A.2b}$$

A.4 Derivatives for the Squared Exponential (SE) Kernel

We use the SE kernel with automatic relevance determination (ARD) (i.e., independent length-scales ℓ_i are included for each dimension to control their importance)

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \, \exp\!\big(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^\top \boldsymbol{L}\,(\boldsymbol{x}-\boldsymbol{x}')\big)\,,$$

with scale hyperparameter σ_f^2 that controls the expected variance of f and $\boldsymbol{L} = \mathrm{diag}(\ell_1^{-2}, \ldots, \ell_d^{-2})$. Let $L_{ii} = \ell_i^{-2}$ and $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}'$ with r_i denoting its i-th component.

First derivatives.

$$\frac{\partial k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i} = -L_{ii}\, r_i\, k(\boldsymbol{x}, \boldsymbol{x}'), \qquad \frac{\partial k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x'_j} = +L_{jj}\, r_j\, k(\boldsymbol{x}, \boldsymbol{x}').$$

Second derivatives (mixed across arguments).

$$\frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i\, \partial x'_j} = \Big(L_{ii}\, \delta_{ij} - L_{ii} L_{jj}\, r_i r_j\Big)\, k(\boldsymbol{x}, \boldsymbol{x}'), \qquad \left.\frac{\partial^2 k}{\partial x_i\, \partial x'_j}\right|_{\boldsymbol{x}'=\boldsymbol{x}} = L_{ii}\, \delta_{ij}\, \sigma_f^2.$$

Second derivatives (same argument).

$$\frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i\, \partial x_j} = \frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x'_i\, \partial x'_j} = \Big(-L_{ii}\, \delta_{ij} + L_{ii} L_{jj}\, r_i r_j\Big)\, k(\boldsymbol{x}, \boldsymbol{x}').$$

Fourth derivatives (two in each argument). For the Hessian trace terms we require

$$\frac{\partial^4 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i\, \partial x_j\, \partial x'_i\, \partial x'_j} = \begin{cases} L_{ii}^2\big(L_{ii}^2 r_i^4 - 6 L_{ii} r_i^2 + 3\big)\, k(\boldsymbol{x}, \boldsymbol{x}') & \text{if } i = j, \\ L_{ii} L_{jj}\big(L_{ii} L_{jj} r_i^2 r_j^2 - L_{ii} r_i^2 - L_{jj} r_j^2 + 1\big)\, k(\boldsymbol{x}, \boldsymbol{x}') & \text{if } i \neq j, \end{cases}$$

and at coincidence $\boldsymbol{x}' = \boldsymbol{x}$ these reduce to

$$\left.\frac{\partial^2 k}{\partial x_i\, \partial x_j}\right|_{\boldsymbol{x}'=\boldsymbol{x}} = -L_{ii}\, \delta_{ij}\, \sigma_f^2, \qquad \left.\frac{\partial^4 k}{\partial x_i\, \partial x_j\, \partial x'_i\, \partial x'_j}\right|_{\boldsymbol{x}'=\boldsymbol{x}} = \sigma_f^2 \times \begin{cases} 3\, L_{ii}^2, & i = j, \\ L_{ii} L_{jj}, & i \neq j. \end{cases}$$

B PROOF OF DATA-DEPENDENT NEWTON-STEP ERROR BOUND

We prove Theorem 1 from the main text in this appendix. Notation for the GP posterior and derivative processes (including $\widehat{\boldsymbol{g}}_{\mathcal{D}}$, $\widehat{\boldsymbol{H}}_{\mathcal{D}}$, and the gradient/Hessian power functions $\pi^{\boldsymbol{g}}_{\mathcal{D}}$, $\pi^{\boldsymbol{H}}_{\mathcal{D}}$) is summarized in Appendix A.

Throughout this work, we make the following standard assumptions about the regularity of the kernel and boundedness of the ground-truth function.

Assumption 1 (Kernel regularity)**.** *The kernel k is stationary and four times continuously differentiable.*

Assumption 2 (Function class)**.** *The ground-truth f is in $\mathcal{H}$ and satisfies $\|f\|_{\mathcal{H}} \leq B$ for some $B < \infty$.*

These two assumptions directly imply pointwise boundedness of all the first- and second-order derivatives of f in terms of the RKHS norm.

B.1 Auxiliary Bounds via Power Functions

The following are minor extensions of standard RKHS "power function" bounds for derivative estimates under GP posteriors, which are a consequence of (Wendland, 2004, Theorem 11.4). The first was presented in (Wu et al., 2023, Lemma 1) and the second we prove here.

Lemma 1 (Gradient posterior error)**.** *For any $\boldsymbol{x} \in \mathcal{X}$ and dataset $\mathcal{D}$,*

$$\big\|\boldsymbol{g}(\boldsymbol{x}) - \widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\big\|^2 \;\leq\; \pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\, \|f\|^2_{\mathcal{H}}.$$

Lemma 2 (Hessian posterior error)**.** *For any $\boldsymbol{x} \in \mathcal{X}$ and dataset $\mathcal{D}$,*

$$\left\|\boldsymbol{H}(\boldsymbol{x}) - \widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})\right\|^2 \;\le\; \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})\, \|f\|^2_{\mathcal{H}}.$$

Proof. Let $\lambda : \mathcal{H} \to \mathbb{R}$ be the composition of the evaluation operator and a differential operator. (Wendland, 2004, Theorem 11.4) provides a bound on the squared error between the operator λ applied to the true function f and the posterior mean function $\mu_{\mathcal{D}}$, i.e.,

$$(\lambda f(\boldsymbol{x}) - \lambda \mu_{\mathcal{D}}(\boldsymbol{x}))^2 \le \lambda^{(1)}\lambda^{(2)} k_{\mathcal{D}}(\boldsymbol{x},\boldsymbol{x}) \|f\|^2_{\mathcal{H}},$$

where $\lambda^{(1)}$ and $\lambda^{(2)}$ are applied to the first and second argument of $k_{\mathcal{D}}(\cdot,\cdot)$, respectively. Select the linear functional to be the second partial derivative $\lambda : f \mapsto \frac{\partial^2}{\partial x_i \partial x_j} f$. The left hand side of the inequality then becomes $\left(\frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x}) - \frac{\partial^2}{\partial x_i \partial x_j} \mu_{\mathcal{D}}(\boldsymbol{x})\right)^2$, which is the error in the n-th element of the vectorized Hessian matrix where $n = (i-1)d + j$. The right hand side is exactly the n-th diagonal entry of $\Sigma^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})$. We can use this inequality for every element n and sum over $n = 1, \ldots, d^2$ to arrive at the Frobenius norm of the error in the Hessian matrix: $\|\boldsymbol{H}(\boldsymbol{x}) - \widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})\|^2_F = \sum_{i=1}^d \sum_{j=1}^d \left(\frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x}) - \frac{\partial^2}{\partial x_i \partial x_j} \mu_{\mathcal{D}}(\boldsymbol{x})\right)^2$. We can then use the standard inequality $\|\boldsymbol{A}\| \le \|\boldsymbol{A}\|_F$ for any square matrix $\boldsymbol{A}$ to complete the proof. □

B.2 Proof of Theorem 1 – Bounding the Newton-Step Error

Recall from the definitions provided in the theorem statement that $\varepsilon_{\mathcal{D}}(\boldsymbol{x}) = \|\boldsymbol{d}(\boldsymbol{x}) - \widehat{\boldsymbol{d}}_{\mathcal{D}}(\boldsymbol{x})\|$, where $\boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{H}(\boldsymbol{x})^{-1}\boldsymbol{g}(\boldsymbol{x})$ and $\widehat{\boldsymbol{d}}_{\mathcal{D}}(\boldsymbol{x}) = \widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})^{-1}\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})$. Suppressing the explicit dependence on $\boldsymbol{x}$ for readability:

$$\boldsymbol{d} - \widehat{\boldsymbol{d}}_{\mathcal{D}} = \boldsymbol{H}^{-1}\boldsymbol{g} - \widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1}\widehat{\boldsymbol{g}}_{\mathcal{D}} = \underbrace{\boldsymbol{H}^{-1}\big(\boldsymbol{g} - \widehat{\boldsymbol{g}}_{\mathcal{D}}\big)}_{(\boldsymbol{a})} + \underbrace{\big(\boldsymbol{H}^{-1} - \widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1}\big)\widehat{\boldsymbol{g}}_{\mathcal{D}}}_{(\boldsymbol{b})}.$$

For any $\boldsymbol{a}, \boldsymbol{b}$ in an inner-product space,

$$\|\boldsymbol{a}+\boldsymbol{b}\|^2 = \|\boldsymbol{a}\|^2 + \|\boldsymbol{b}\|^2 + 2\langle \boldsymbol{a}, \boldsymbol{b}\rangle \le \|\boldsymbol{a}\|^2 + \|\boldsymbol{b}\|^2 + 2\|\boldsymbol{a}\|\,\|\boldsymbol{b}\| \le 2\|\boldsymbol{a}\|^2 + 2\|\boldsymbol{b}\|^2,$$

where the first is the Cauchy-Schwarz inequality and the second uses Young's inequality (i.e., $2uv \le u^2 + v^2$ or equivalently $(u-v)^2 \ge 0$). Combining this with submultiplicativity, we get:

$$\varepsilon^2_{\mathcal{D}} \le 2\|\boldsymbol{H}^{-1}\|^2\, \|\boldsymbol{g} - \widehat{\boldsymbol{g}}_{\mathcal{D}}\|^2 + 2\|\boldsymbol{H}^{-1} - \widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1}\|^2\, \|\widehat{\boldsymbol{g}}_{\mathcal{D}}\|^2.$$

Use the resolvent identity $\boldsymbol{H}^{-1} - \widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1} = \boldsymbol{H}^{-1}\big(\widehat{\boldsymbol{H}}_{\mathcal{D}} - \boldsymbol{H}\big)\widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1}$ to bound the second term by $\|\boldsymbol{H}^{-1}\|^2\, \|\widehat{\boldsymbol{H}}_{\mathcal{D}} - \boldsymbol{H}\|^2\, \|\widehat{\boldsymbol{H}}_{\mathcal{D}}^{-1}\|^2\, \|\widehat{\boldsymbol{g}}_{\mathcal{D}}\|^2$. Applying Lemmas 1–2 and $\|f\|_{\mathcal{H}} \le B$ gives

$$\varepsilon^2_{\mathcal{D}} \le 2\,B^2\, \|\boldsymbol{H}^{-1}\|^2 \left[\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) + \underbrace{\|\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})^{-1}\|^2\, \|\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\|^2}_{s_{\mathcal{D}}(\boldsymbol{x})}\, \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})\right].$$

Equivalently, $\varepsilon_{\mathcal{D}}(\boldsymbol{x}) \le C_{\boldsymbol{x}} \sqrt{\pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) + s_{\mathcal{D}}(\boldsymbol{x})\, \pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})}$ with $C_{\boldsymbol{x}} = \sqrt{2}B\|\boldsymbol{H}(\boldsymbol{x})^{-1}\|$, as stated. □

What the bound says. The error in the Newton step decomposes into (i) gradient uncertainty and (ii) Hessian uncertainty at $\boldsymbol{x}$ *scaled* by a factor $s_{\mathcal{D}}(\boldsymbol{x}) = \|\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})^{-1}\|^2\, \|\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x})\|^2$. This scale is large precisely when the local problem is ill-conditioned and/or when the gradient is sizeable, so the bound quantitatively formalizes when learning curvature is disproportionately valuable.

C EMPIRICAL STUDY OF THE SCALE FACTOR'S IMPACT

C.1 Details on Monte Carlo estimation of NeST acquisition

For completeness, we restate the complete NeST acquisition shown in (7) here:

$$\tilde{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z} \mid \boldsymbol{x}_t, \mathcal{D}) = \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t) + \mathbb{E}_{\boldsymbol{y}|\mathcal{D},\boldsymbol{Z}}\big[\, s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\,\big]\; \pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t), \tag{C.1}$$

where $\boldsymbol{y}$ denotes the (noisy) batch observations at the candidate locations $\boldsymbol{Z}$, i.e., $\boldsymbol{y} = f(\boldsymbol{Z}) + \boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 I)$. The two power-function terms in (C.1) depend only on the design $\boldsymbol{Z}$ (and the GP hyperparameters), whereas the scale factor depends on the *realized* post-batch posterior mean of the gradient and Hessian at $\boldsymbol{x}_t$, i.e., $s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}) = \left\|\widehat{\boldsymbol{H}}_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x})^{-1}\right\|^2 \left\|\widehat{\boldsymbol{g}}_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x})\right\|^2$.

Under a GP with Gaussian observation noise, the random vector $\boldsymbol{y} \mid \mathcal{D}, \boldsymbol{Z}$ is multivariate Gaussian,

$$\boldsymbol{y}|\mathcal{D}, \boldsymbol{Z} \sim \mathcal{N}\big(\boldsymbol{\mu}_{\mathcal{D}}(\boldsymbol{Z}),\, \Sigma_{\mathcal{D}}(\boldsymbol{Z}) + \sigma^2 I\big), \tag{C.2}$$

where $\boldsymbol{\mu}_{\mathcal{D}}(\boldsymbol{Z})$ and $\Sigma_{\mathcal{D}}(\boldsymbol{Z})$ are the GP posterior mean and covariance of $f(\boldsymbol{Z})|\mathcal{D}$. However, the mapping $\boldsymbol{y} \mapsto s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)$ is nonlinear and thus in general is non-Gaussian; this makes the expectation in (C.1) analytically intractable, motivating the need for Monte Carlo (MC) approximation.

Let $\Sigma_y = \Sigma_{\mathcal{D}}(\boldsymbol{Z}) + \sigma^2 I$ and $\boldsymbol{\mu}_y = \boldsymbol{\mu}_{\mathcal{D}}(\boldsymbol{Z})$. We approximate the expectation in (C.1) by drawing S samples

$$\boldsymbol{y}^{(s)} \sim \mathcal{N}(\boldsymbol{\mu}_y, \Sigma_y), \qquad s = 1, \ldots, S, \tag{C.3}$$

and computing

$$\mathbb{E}\big[s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y})}(\boldsymbol{x}_t)\big] \approx \frac{1}{S}\sum_{s=1}^{S} s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y}^{(s)})}(\boldsymbol{x}_t). \tag{C.4}$$

Substituting (C.4) into (C.1) yields the MC-approximated acquisition

$$\tilde{\alpha}^{\mathrm{MC}}_{\mathrm{NeST}}(\boldsymbol{Z}|\boldsymbol{x}_t, \mathcal{D}) = \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t) + \left(\frac{1}{S}\sum_{s=1}^{S} s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y}^{(s)})}(\boldsymbol{x}_t)\right) \pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t), \tag{C.5}$$

where $s_{\mathcal{D}\cup(\boldsymbol{Z},\boldsymbol{y}^{(s)})}(\boldsymbol{x}_t)$ can be evaluated using standard GP conditioning rules described in Appendix A.

C.2 Comparison of scale factor selection methods on synthetic problems

The practical NeST acquisition (8) derived in the main text is

$$\widehat{\alpha}_{\mathrm{NeST}}(\boldsymbol{Z}|\boldsymbol{x}_t, \mathcal{D}, \widehat{s}_t) = \pi^{\boldsymbol{g}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}\cup\boldsymbol{Z}}(\boldsymbol{x}_t),$$

i.e., a weighted sum of the local gradient and Hessian power functions. While the one-step lookahead form (7) includes an expectation over future observations, estimating that expectation by Monte Carlo can be costly. This appendix asks: *how sensitive is performance to the choice of the scale factor $\widehat{s}_t$?*

Setup. We compare four acquisition functions for *sequential* design over $b_t = d$ points:

- $\widehat{\alpha}_{\mathrm{NeST}}(s{=}1)$ with $\widehat{s}_t = 1$ (our default);
- $\widehat{\alpha}_{\mathrm{NeST}}(s{=}\text{plugin})$ with $\widehat{s}_t = s_{\mathcal{D}}(\boldsymbol{x}_t) = \|\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}_t)^{-1}\|^2 \|\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}_t)\|^2$;
- $\tilde{\alpha}_{\mathrm{NeST}}(\mathrm{MC})$, which uses a MC estimate of the expectation (with 32 samples) in (7), i.e., (C.5) with $S = 32$;
- the GIBO gradient–information rule $\tilde{\alpha}_{\mathrm{GI}}$.

We also include a random sampling (RS) baseline. At each iteration, we *minimize* the chosen acquisition over the domain using L-BFGS (Zhu et al., 1997) with 20 random multistarts. We study the Griewank, Ackley, Rosenbrock, and Sphere functions in dimensions $d \in \{2, 3, 4, 5\}$ on $[0, 1]^d$. The mathematical expressions for each function can be found in Appendix F.3. For each d, we evaluate the Newton-step error $\varepsilon_{\mathcal{D}}(\boldsymbol{x}) = \|\boldsymbol{d}(\boldsymbol{x}) - \widehat{\boldsymbol{d}}_{\mathcal{D}}(\boldsymbol{x})\|$ at 10 randomly chosen test locations $\boldsymbol{x}$ and report the per-iteration median across 10 replicates. The GP hyperparameters are fixed across methods: they are fit once (on a separate set of samples) and then held constant. For the Griewank function, initial designs use 5/10/20/30 points for d=2/3/4/5. For all the other synthetic functions, initial designs use 5/5/10/10 points for d=2/3/4/5.

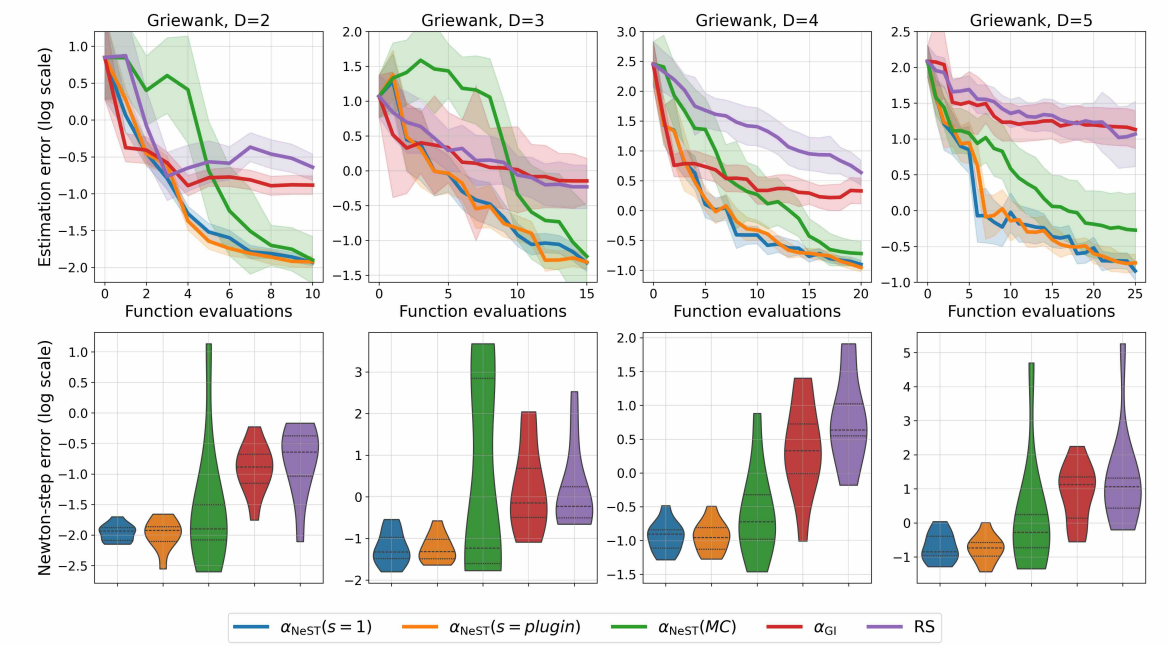

Figure C.1: Empirical study of scale factor sensitivity (Griewank). **Top:** Median Newton-step error (log scale) versus number of function evaluations. **Bottom:** Distribution of Newton-step errors (log scale) at the final iteration. NeST with a fixed $s = 1$ closely tracks the plug-in and Monte Carlo (MC) sampling variants and consistently beats α_{GI}, the core acquisition underpinning the GIBO method (Müller et al., 2021), and random sampling (RS). All experiments were replicated 10 times and the shaded regions show ± one standard error.

Results. Figure C.1–C.4 illustrate two main messages: (i) $\alpha_{\text{NeST}}(s{=}1)$ and the plug-in/MC variants reduce Newton-step error at similar rates across all d and consistently outperform α_{GI} and RS (though the MC has more variability due to its inherent randomness); and (ii) this advantage translates into lower error, as seen in the final error distributions across the different runs (bottom row). The gap widens with dimension, where curvature information becomes more important. A fixed, data-agnostic weight $s_t{=}1$ appears to be a robust and inexpensive choice: it matches the plug-in and MC versions while avoiding extra computation and hyper-sensitivity. This supports our the default selected in all experiments in the main text. It would be interesting to more carefully study, either theoretically or empirically, how the tuning of $\widehat{s}_t$ impacts optimization performance; this type of analysis was outside the scope of this initial contribution. Our results further suggest that NeST-BO's gains might come from targeting curvature at all compared to delicate tuning of $\widehat{s}_t$ – but this yet to be rigorously formalized.

C.3 Impact of scale factor on overall optimization performance

In the main experiments we default to a fixed scale $\widehat{s}_t = 1$, mainly to avoid introducing an additional moving part. As an additional sanity check, we compare this default to the "plug-in" alternative that performed similarly in our previous tests (Appendix C.2 and can be more easily justified in that it replaces the expected lookahead version with its current estimate. This plug-in choice is also appealing because it is "free" to compute once $\widehat{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}_t)$ and $\widehat{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x}_t)$ are available.

Figures C.5 and C.6, respectively, show full optimization runs on a synthetic 20-D Griewank function and a 60-D rover trajectory benchmark (each over 10 random seeds, with the same evaluation budget and experimental setup as in the main text). Overall, the plug-in and $\widehat{s}_t = 1$ variants behave very similarly across both problems, and both substantially outperform the Sobol baseline. In particular, these results support the choice $\widehat{s}_t = 1$ as a simple default that does not appear to sacrifice performance in the regimes we tested.

Figure C.2: Empirical study of scale factor sensitivity (Ackley). **Top:** Median Newton-step error (log scale) versus number of function evaluations. **Bottom:** Distribution of Newton-step errors (log scale) at the final iteration. NeST with a fixed $s = 1$ closely tracks the plug-in and Monte Carlo (MC) sampling variants and consistently beats α_{GI}, the core acquisition underpinning the GIBO method (Müller et al., 2021), and random sampling (RS). All experiments were replicated 10 times and the shaded regions show $\pm$ one standard error.

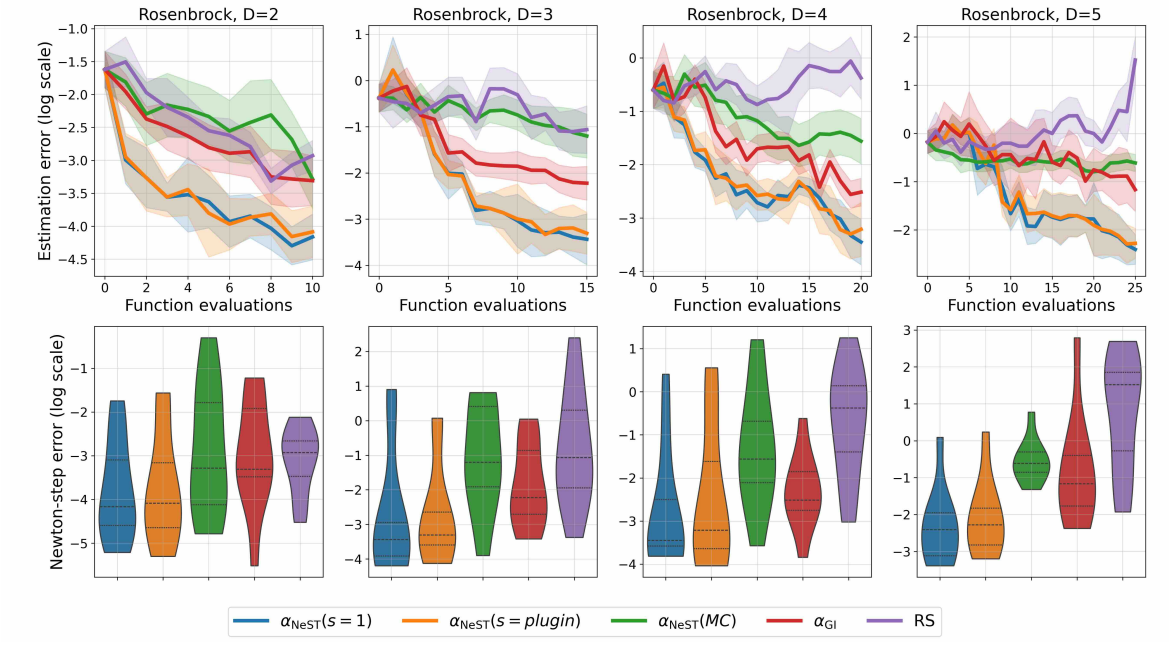

Figure C.3: Empirical study of scale factor sensitivity (Rosenbrock). **Top:** Median Newton-step error (log scale) versus number of function evaluations. **Bottom:** Distribution of Newton-step errors (log scale) at the final iteration. NeST with a fixed $s = 1$ closely tracks the plug-in and Monte Carlo (MC) sampling variants and consistently beats α_{GI}, the core acquisition underpinning the GIBO method (Müller et al., 2021), and random sampling (RS). All experiments were replicated 10 times and the shaded regions show $\pm$ one standard error.

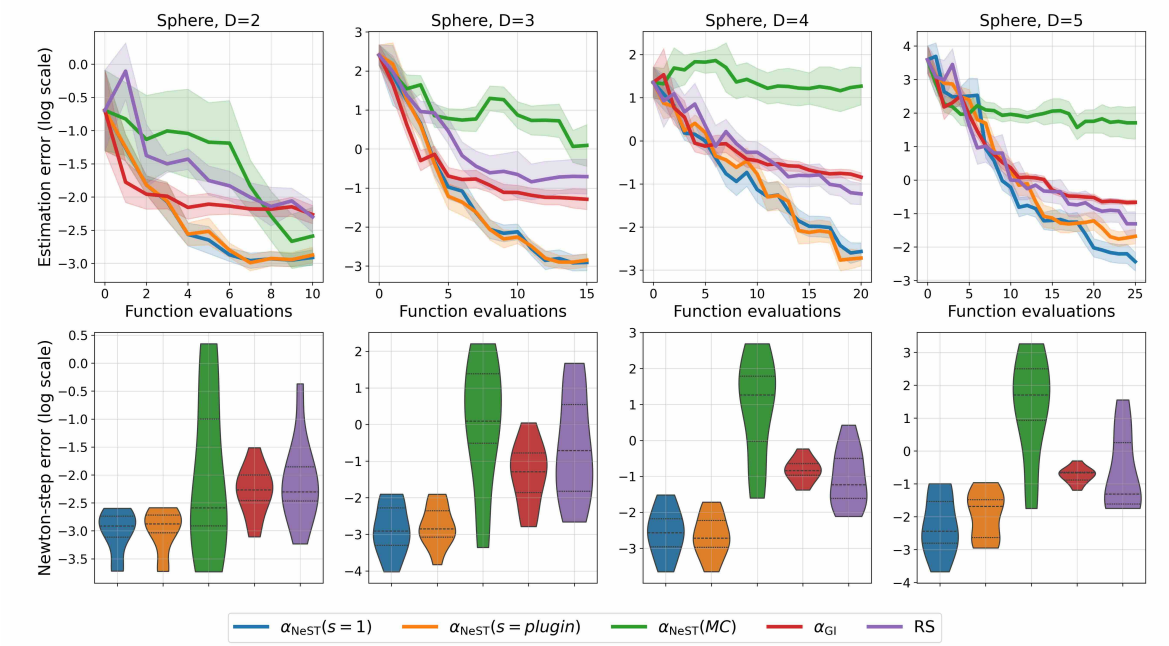

Figure C.4: Empirical study of scale factor sensitivity (Sphere). **Top:** Median Newton-step error (log scale) versus number of function evaluations. **Bottom:** Distribution of Newton-step errors (log scale) at the final iteration. NeST with a fixed $s = 1$ closely tracks the plug-in and Monte Carlo (MC) sampling variants and consistently beats α_{GI}, the core acquisition underpinning the GIBO method (Müller et al., 2021), and random sampling (RS). All experiments were replicated 10 times and the shaded regions show ± one standard error.

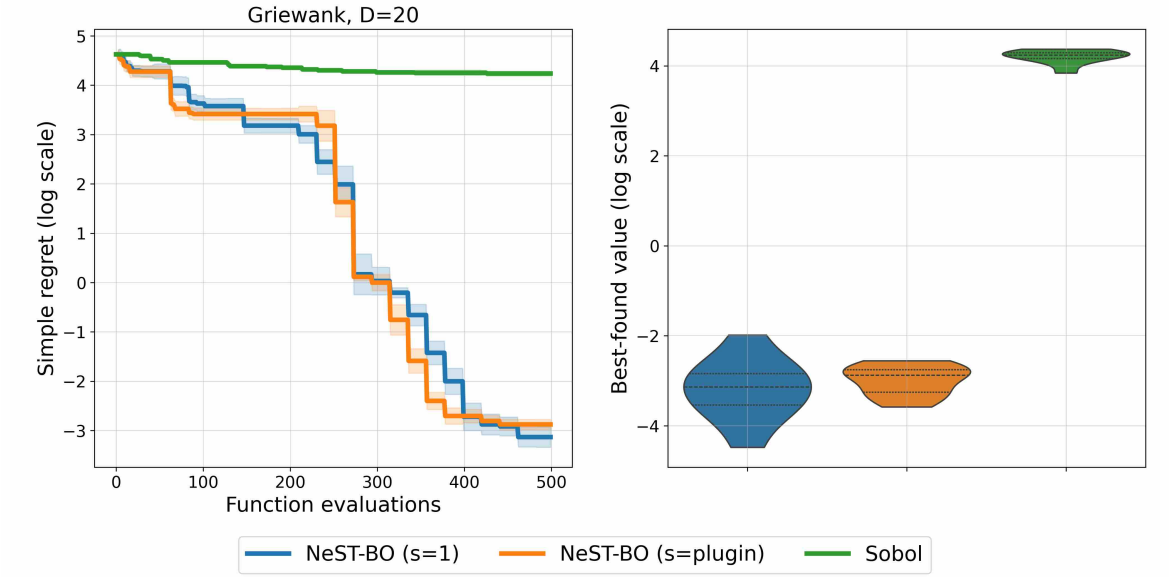

Figure C.5: Scale-factor sensitivity on the 20-D Griewank function. **Left:** Median simple regret (log scale) over 10 runs, with shaded bands showing variability across runs. **Right:** Distribution (violin plot) of the simple regret achieved by each method at the end of the budget, across the same runs. Lower is better.

D PRACTICAL IMPLEMENTATION OF NEST-BO

This appendix describes the NeST-BO variant used in our experiments (Algorithm 2). The overall structure matches Algorithm 1 in the main body, with two implementation choices that are worth making explicit: (i) we

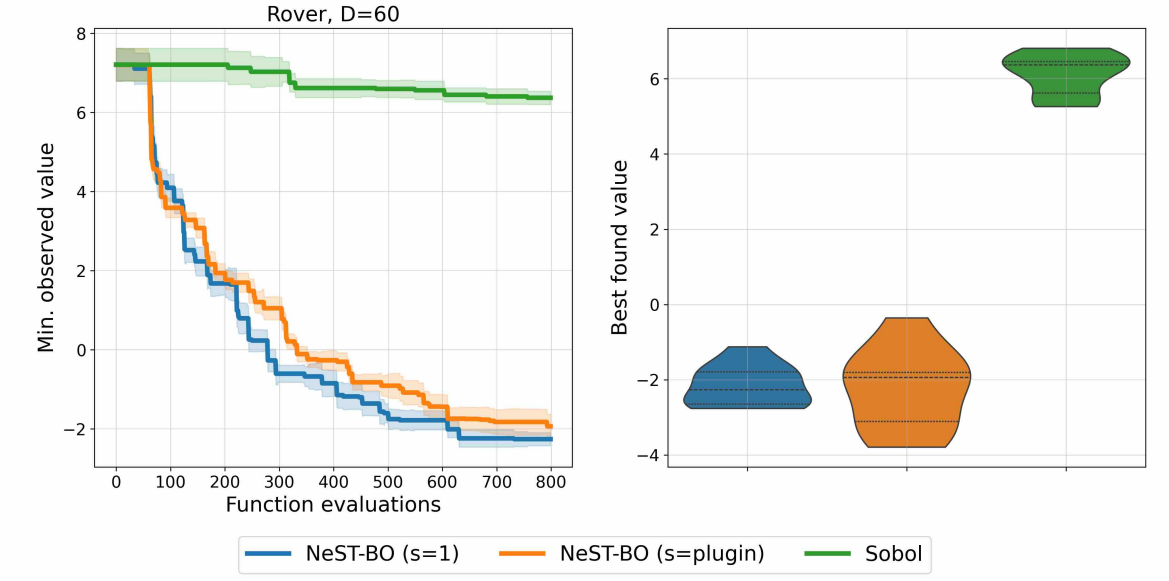

Figure C.6: Scale-factor sensitivity on the 60-D rover trajectory problem. **Left:** Median best-found objective value over 10 runs, with shaded bands showing variability across runs. **Right:** Distribution (violin plot) of the final best objective value achieved by each method at end of budget, across the same runs. Lower is better.

perform greedy (sequential) selection of M points instead of solving a joint Md-dimensional batch problem, and (ii) we only take a Newton step when the posterior-mean Hessian at the current iterate is *numerically* positive definite; otherwise we fall back to a length-scale-normalized gradient step following the GIBO procedure (Müller et al., 2021, Appendix A.4).

Furthermore, at the start of each outer iteration, we fit GP hyperparameters to the current dataset via a user-specified routine FitGP. In our experiments this is the standard marginal likelihood estimator (MLE) with restarts; when hyperparameter priors are available, a MAP variant can be used with no change to the algorithm. During the subsequent greedy inner loop, we keep these hyperparameters fixed and update the posterior as new observations arrive. If we take a Newton step, we use an Armijo backtracking line search on the GP posterior mean $\mu_{\mathcal{D}}$ (standard defaults; see, e.g., (Bertsekas, 2016, Chapter 1)). If we fall back to a gradient step, we use the length-scale-normalized update and step-size rule from GIBO (Müller et al., 2021); we do not reproduce the full derivation here, but Algorithm 2 makes clear where it enters. Lastly, $\Pi_{\mathcal{X}}$ denotes the projection operator onto the feasible set $\mathcal{X}$, so that we perform simple projection to ensure feasibility at every iteration (again this could be replaced with more sophisticated feasibility-preserving steps in the future).

E THEORETICAL CONVERGENCE PROOFS

We continue to use the notation summarized in Appendix A and let Assumptions 1–2 hold from Appendix B. We first provide a complete formal statement and proof of Theorem 2 and then show how NeST-BO inherits local quadratic convergence guarantees of inexact Newton's method.

E.1 Complete statement and proof of Theorem 2

This section provides a complete statement and proof of Theorem 2, showing that the vanishing power-function condition (VPC) holds under NeST sampling in both noiseless and noisy settings. The argument combines the following steps.

1. First, we show that NeST acquisition at $\boldsymbol{x}_t$ is upper bounded by an origin-centered error function depending only on the candidate design via monotonicity under conditioning and stationarity (Lemma 3). This reduces the problem to constructing designs that make this error arbitrarily small.

Algorithm 2 Practical implementation of NeST-BO

Inputs: initial iterate $\boldsymbol{x}_0 \in \mathcal{X} \subseteq \mathbb{R}^d$; initial dataset $\mathcal{D}_0$; outer iterations T; greedy inner selections per iteration M; scale rule $\widehat{s}_t > 0$ (we use $\widehat{s}_t = 1$ unless stated otherwise); GP hyperparameter-fitting routine FitGP; Armijo line-search routine Armijo; and GIBO gradient-normalization routine GIBOstep.

for $t = 0, \dots, T-1$ **do**
  Fit GP: $(\theta_t, \text{posterior}) \leftarrow \text{FitGP}(\mathcal{D}_t)$ ▷ hyperparameters θ_t via MLE/MAP
  Set $\mathcal{D}_t^{(0)} \leftarrow \mathcal{D}_t$
  for $m = 1, \dots, M$ **do** ▷ greedy (sequential) NeST selection
    $\boldsymbol{z}_{t,m} \in \operatorname{argmin}_{\boldsymbol{z} \in \mathcal{X}} \ \widehat{\alpha}_{\text{NeST}}\Big(\boldsymbol{z} \mid \boldsymbol{x}_t, \mathcal{D}_t^{(m-1)}, \widehat{s}_t; \theta_t\Big)$
    Observe $y_{t,m} = f(\boldsymbol{z}_{t,m}) + \varepsilon_{t,m}$ and set $\mathcal{D}_t^{(m)} \leftarrow \mathcal{D}_t^{(m-1)} \cup (\boldsymbol{z}_{t,m}, y_{t,m})$
    Update GP posterior conditioned on $\mathcal{D}_t^{(m)}$ (keeping θ_t fixed)
  end for
  Set $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t^{(M)}$
  Compute posterior-mean derivatives at $\boldsymbol{x}_t$:
    $\widehat{\boldsymbol{g}} \leftarrow \widehat{\boldsymbol{g}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t), \quad \widehat{\boldsymbol{H}} \leftarrow \widehat{\boldsymbol{H}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$
  Symmetrize: $\widehat{\boldsymbol{H}} \leftarrow (\widehat{\boldsymbol{H}} + \widehat{\boldsymbol{H}}^\top)/2$
  if $\text{IsPD}(\widehat{\boldsymbol{H}}; \tau)$ **then** ▷ e.g., Cholesky succeeds with tolerance $\tau > 0$
    Newton direction: solve $\widehat{\boldsymbol{H}}\, \boldsymbol{d}_t = \widehat{\boldsymbol{g}}$ for $\boldsymbol{d}_t$
    Line search: $\gamma_t \leftarrow \text{Armijo}\big(\mu_{\mathcal{D}_{t+1}}, \boldsymbol{x}_t, -\boldsymbol{d}_t\big)$
    $\boldsymbol{x}_{t+1} \leftarrow \Pi_{\mathcal{X}}(\boldsymbol{x}_t - \gamma_t\, \boldsymbol{d}_t)$
  else
    Gradient fallback: $(\boldsymbol{d}_t, \gamma_t) \leftarrow \text{GIBOstep}(\widehat{\boldsymbol{g}}, \theta_t)$
      ▷ θ_t provides GP lengthscales used to normalize gradient step (Müller et al., 2021)
    $\boldsymbol{x}_{t+1} \leftarrow \Pi_{\mathcal{X}}(\boldsymbol{x}_t - \gamma_t\, \boldsymbol{d}_t)$
  end if
end for

2. Second, we establish that derivative evaluation is a bounded linear functional on $\mathcal{H}$ with an explicit Riesz representer (Lemma 4), and that the GP posterior standard deviation for such a functional equals the optimal worst-case error among all linear rules based on function evaluations at the design points (Lemmas 5–6).

3. Third, we construct explicit centered-difference linear rules supported on a symmetric stencil of radius h, and use Taylor expansions with remainder plus RKHS derivative bounds to show the associated worst-case functional errors are $O(h^2)$. By the optimal worst-case-error identity, this yields $O(h^4)$ bounds on the posterior variances (power functions) for all gradient and Hessian components (Lemma 7).

4. Lastly, we combine the reduction and the $O(h^4)$ bounds to prove the noiseless VPC result. For noisy observations, we use m replicates per stencil location and show that conditioning on all replicates is equivalent to conditioning on the averaged observations with noise variance σ^2/m, which adds an explicit $O(\sigma^2/m)$ contribution that can be driven to zero by increasing m.

We proceed by establishing the lemmas in this order and then formally stating and proving Theorem 2.

Lemma 3. *By Assumption 1, the kernel is stationary $k(\boldsymbol{x}, \boldsymbol{x}') = \varphi(\boldsymbol{x} - \boldsymbol{x}')$. Fix any dataset $\mathcal{D}_t$ and any $\widehat{s}_t > 0$. Define the error function*

$$E_{d,k,s,\sigma}(b) = \inf_{\boldsymbol{Z} \in \mathcal{X}^b} \Big(\pi^{\boldsymbol{g}}_{\boldsymbol{Z}}(\boldsymbol{0}) + s\, \pi^{\boldsymbol{H}}_{\boldsymbol{Z}}(\boldsymbol{0})\Big), \tag{E.1}$$

where the power functions are evaluated at $\boldsymbol{0}$ and conditioned only on potentially noisy observations (with noise variance σ^2) at $\boldsymbol{Z}$. Then, for any batch size b_t,

$$\inf_{\boldsymbol{Z} \in \mathbb{R}^{b_t \times d}} \Phi_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) \le E_{d,k,\widehat{s}_t,\sigma}(b_t),$$

where $\Phi_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) = \pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t)$ is short for the approximate NeST acquisition function (8).

Proof. This result builds directly on two observations from (Wu et al., 2023). The first is monotonicity under conditioning, i.e., for any GP, conditioning on a *larger* dataset cannot increase posterior variances. Here, since $\mathcal{D}_t \cup \boldsymbol{Z}$ contains $\boldsymbol{Z}$, we have for every $\boldsymbol{x} \in \mathcal{X}$

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}) \leq \pi^{\boldsymbol{g}}_{\boldsymbol{Z}}(\boldsymbol{x}), \qquad \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}) \leq \pi^{\boldsymbol{H}}_{\boldsymbol{Z}}(\boldsymbol{x}),$$

hence $\Phi_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}) \leq \pi^{\boldsymbol{g}}_{\boldsymbol{Z}}(\boldsymbol{x}) + \widehat{s}_t \pi^{\boldsymbol{H}}_{\boldsymbol{Z}}(\boldsymbol{x})$.

The second is that, due to stationarity of the kernel, we can always translate our data to the origin. Specifically, let $\boldsymbol{u}_t = -\boldsymbol{x}_t$ and define the translated design $\widetilde{\boldsymbol{Z}} = \boldsymbol{Z} + \mathbf{1}\boldsymbol{u}_t^\top$ (shift every candidate location by $-\boldsymbol{x}_t$ where $\mathbf{1}$ is the all ones vector). Stationarity implies that the joint law of $\{f(\boldsymbol{x}_t), f(\boldsymbol{Z})\}$ is identical to that of $\{f(\mathbf{0}), f(\widetilde{\boldsymbol{Z}})\}$ after translation; consequently,

$$\pi^{\boldsymbol{g}}_{\boldsymbol{Z}}(\boldsymbol{x}_t) = \pi^{\boldsymbol{g}}_{\widetilde{\boldsymbol{Z}}}(\mathbf{0}), \qquad \pi^{\boldsymbol{H}}_{\boldsymbol{Z}}(\boldsymbol{x}_t) = \pi^{\boldsymbol{H}}_{\widetilde{\boldsymbol{Z}}}(\mathbf{0}).$$

Therefore,

$$\inf_{\boldsymbol{Z}} \Phi_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) \leq \inf_{\widetilde{\boldsymbol{Z}}} \left(\pi^{\boldsymbol{g}}_{\widetilde{\boldsymbol{Z}}}(\mathbf{0}) + \widehat{s}_t \pi^{\boldsymbol{H}}_{\widetilde{\boldsymbol{Z}}}(\mathbf{0})\right) = E_{d,k,\widehat{s}_t,\sigma}(b_t),$$

which is the claim and thus concludes the proof. □

Lemma 4. *Assume $\mathcal{X} \subset \mathbb{R}^d$ is open and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is m-times continuously differentiable, and let α be a multi-index with $|\alpha| \leq m$. Define $L_{\alpha,\boldsymbol{x}}(f) = \partial^\alpha f(\boldsymbol{x})$ for all $f \in \mathcal{H}_k$ and $\boldsymbol{x} \in \mathcal{X}$. Then $L_{\alpha,\boldsymbol{x}}$ is a bounded linear functional on $\mathcal{H}$, and its Riesz representer is*

$$r_{\alpha,\boldsymbol{x}} = \partial^{0,\alpha} k(\cdot, \boldsymbol{x}) \in \mathcal{H}_k,$$

such that for all $f \in \mathcal{H}$,

$$\partial^\alpha f(\boldsymbol{x}) = \langle f, \partial^{0,\alpha} k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}.$$

Moreover, $\|r_{\alpha,\boldsymbol{x}}\|^2_{\mathcal{H}_k} = \partial^{\alpha,\alpha} k(\boldsymbol{x}, \boldsymbol{x})$ and $|L_{\alpha,\boldsymbol{x}}(f)| \leq \|f\|_{\mathcal{H}} \, \|r_{\alpha,\boldsymbol{x}}\|_{\mathcal{H}}$.

Proof. Linearity of $L_{\alpha,\boldsymbol{x}}$ is immediate from linearity of partial derivatives.

By the differentiability assumption on k, the derivative evaluation functional $A(f) = \partial^\alpha f(\boldsymbol{x})$ is bounded on $\mathcal{H}_k$ and its Riesz representer is given by the kernel derivative $f_A = \partial^{0,\alpha} k(\cdot, \boldsymbol{x}) \in \mathcal{H}$ (see, e.g., Example 4.5 in (Kanagawa et al., 2025)). Therefore, for all $f \in \mathcal{H}$,

$$\partial^\alpha f(\boldsymbol{x}) = A(f) = \langle f, f_A \rangle_{\mathcal{H}} = \langle f, \partial^{0,\alpha} k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}.$$

Boundedness follows from the Cauchy-Schwarz inequality:

$$|\partial^\alpha f(\boldsymbol{x})| = |\langle f, r_{\alpha,\boldsymbol{x}} \rangle_{\mathcal{H}}| \leq \|f\|_{\mathcal{H}} \, \|r_{\alpha,\boldsymbol{x}}\|_{\mathcal{H}}.$$

Finally, the same reference gives $\|r_{\alpha,\boldsymbol{x}}\|^2_{\mathcal{H}} = \partial^{\alpha,\alpha} k(\boldsymbol{x}, \boldsymbol{x})$. □

Lemma 5. *Fix a candidate design $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_b) \in \mathbb{R}^{b \times d}$ with distinct points and consider any weights $\boldsymbol{w} = (w_1, \ldots, w_b) \in \mathbb{R}^b$. Define the linear estimator:*

$$\widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f) = \sum_{j=1}^{b} w_j f(\boldsymbol{z}_j).$$

Let $e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(\cdot) = r_{\alpha,\boldsymbol{x}}(\cdot) - \sum_{j=1}^{b} w_j k(\cdot, \boldsymbol{z}_j) \in \mathcal{H}$. Then, for every $f \in \mathcal{H}$, we have:

$$L_{\alpha,\boldsymbol{x}}(f) - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f) = \langle f, e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}} \rangle_{\mathcal{H}},$$

and hence the worst-case error over the unit ball is:

$$\sup_{f \in \mathcal{H}: \|f\|_H \leq 1} \left| L_{\alpha,\boldsymbol{x}}(f) - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f) \right| = \|e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\|_{\mathcal{H}}$$

Proof. We start by using the reproducing property to derive:

$$\widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f) = \sum_{j=1}^{b} w_j f(\boldsymbol{z}_j) = \sum_{j=1}^{b} w_j \langle f, k(\cdot, \boldsymbol{z}_j)\rangle_{\mathcal{H}} = \left\langle f, \sum_{j=1}^{b} w_j k(\cdot, \boldsymbol{z}_j) \right\rangle.$$

Also, by Lemma 4, we have $L_{\alpha,\boldsymbol{x}}(f) = \langle f, r_{\alpha,\boldsymbol{x}}\rangle$. Subtracting these two gives:

$$L_{\alpha,\boldsymbol{x}}(f) - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f) = \left\langle f, r_{\alpha,\boldsymbol{x}} - \sum_{j=1}^{b} w_j k(\cdot, \boldsymbol{z}_j) \right\rangle = \langle f, e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\rangle.$$

Taking the supremum over $\|f\|_{\mathcal{H}} \leq 1$ yields the RKHS norm by duality (the Cauchy-Schwarz inequality gives the $\leq$ and equality is achieved by $f = e/\|e\|$ when $e \neq 0$). □

Lemma 6. *Consider noiseless GP interpolation with prior $f \sim \mathcal{GP}(0, k)$ on $\mathcal{X} \subset \mathbb{R}^d$, and observations $\{f(\boldsymbol{z}_j)\}_{j=1}^b$ at a design $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_b) \in \mathbb{R}^{b\times d}$. Let $K = k(\boldsymbol{Z}, \boldsymbol{Z})$ and assume K is invertible (or interpret K^{-1} as the Moore–Penrose pseudoinverse). Let $k_{\boldsymbol{Z}}$ denote the posterior covariance kernel under noiseless conditioning:*

$$k_{\boldsymbol{Z}}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{Z}) K^{-1} k(\boldsymbol{Z}, \boldsymbol{x}').$$

Fix a multi-index α such that the derivative functional $L_{\alpha,\boldsymbol{x}}(f) = \partial^\alpha f(\boldsymbol{x})$ is bounded on $\mathcal{H}_k$.

1. *The posterior variance of the functional $L_{\alpha,\boldsymbol{x}}(f)$ is*

$$\mathrm{Var}(L_{\alpha,\boldsymbol{x}}(f) \mid f(\boldsymbol{Z})) = L^{(1)}_{\alpha,\boldsymbol{x}} L^{(2)}_{\alpha,\boldsymbol{x}} k_{\boldsymbol{Z}}(\boldsymbol{x}, \boldsymbol{x}),$$

   where $L^{(1)}$ applies L to the first argument of the kernel and $L^{(2)}$ applies it to the second.

2. *The posterior standard deviation equals the optimal worst-case error among linear rules based on $\{f(\boldsymbol{z}_j)\}_{j=1}^b$:*

$$\sqrt{\mathrm{Var}(L_{\alpha,\boldsymbol{x}}(f) \mid f(\boldsymbol{Z}))} = \inf_{\boldsymbol{w}\in\mathbb{R}^b} \sup_{f\in\mathcal{H}:\|f\|_H\leq 1} \left| L_{\alpha,\boldsymbol{x}}(f) - \sum_{j=1}^{b} w_j f(\boldsymbol{z}_j) \right|.$$

Proof. Let $u = L_{\alpha,\boldsymbol{x}}(f)$. Since $(f(\boldsymbol{Z}), u)$ is jointly Gaussian, we have the standard conditional-variance formula

$$\mathrm{Var}(u \mid f(\boldsymbol{Z})) = \mathrm{Var}(u) - \boldsymbol{c}^\top K^{-1} \boldsymbol{c},$$

where $\boldsymbol{c} \in \mathbb{R}^b$ has entries $c_j = \mathrm{Cov}(f(\boldsymbol{z}_j), u)$. By linearity of covariance for GPs and differentiability of k,

$$\mathrm{Var}(u) = L^{(1)}_{\alpha,\boldsymbol{x}} L^{(2)}_{\alpha,\boldsymbol{x}} k(\boldsymbol{x}, \boldsymbol{x}), \qquad c_j = L^{(2)}_{\alpha,\boldsymbol{x}} k(\boldsymbol{z}_j, \boldsymbol{x}).$$

Expanding $L^{(1)}_{\alpha,\boldsymbol{x}} L^{(2)}_{\alpha,\boldsymbol{x}} k_{\boldsymbol{Z}}(\boldsymbol{x}, \boldsymbol{x})$ using the definition of $k_{\boldsymbol{Z}}$ yields exactly $\mathrm{Var}(u) - \boldsymbol{c}^\top K^{-1} \boldsymbol{c}$, proving point 1.

For point 2, applying Lemma 5, we see that the right-hand side in the expression equals $\inf_{\boldsymbol{w}} \|e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\|_{\mathcal{H}}$.

It remains to identify $\mathrm{Var}(u \mid f(\boldsymbol{Z}))$ as $\inf_{\boldsymbol{w}} \|e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\|^2_{\mathcal{H}}$. For any $\boldsymbol{w}$, the random variable $u - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f)$ is Gaussian and has variance

$$\mathrm{Var}\Big(u - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f)\Big) = \left\| r_{\alpha,\boldsymbol{x}} - \sum_{j=1}^{b} w_j k(\cdot, \boldsymbol{z}_j) \right\|^2_{\mathcal{H}} = \|e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\|^2_{\mathcal{H}},$$

by expanding the squared RKHS norm and using $\langle k(\cdot, \boldsymbol{z}_i), k(\cdot, \boldsymbol{z}_j)\rangle_{\mathcal{H}} = k(\boldsymbol{z}_i, \boldsymbol{z}_j)$, $\langle r_{\alpha,\boldsymbol{x}}, k(\cdot, \boldsymbol{z}_j)\rangle_{\mathcal{H}} = L^{(2)}_{\alpha,\boldsymbol{x}} k(\boldsymbol{z}_j, \boldsymbol{x})$, and $\langle r_{\alpha,\boldsymbol{x}}, r_{\alpha,\boldsymbol{x}}\rangle_{\mathcal{H}} = L^{(1)}_{\alpha,\boldsymbol{x}} L^{(2)}_{\alpha,\boldsymbol{x}} k(\boldsymbol{x}, \boldsymbol{x})$.

Finally, for jointly Gaussian variables, the conditional mean $\mathbb{E}[u \mid f(\boldsymbol{Z})]$ is the minimum mean squared error predictor of u measurable with respect to $f(\boldsymbol{Z})$, and it is linear in $f(\boldsymbol{Z})$. Therefore,

$$\mathrm{Var}(u \mid f(\boldsymbol{Z})) = \min_{\boldsymbol{w}\in\mathbb{R}^b} \mathrm{Var}\Big(u - \widehat{L}^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}(f)\Big) = \min_{\boldsymbol{w}\in\mathbb{R}^b} \|e^{\boldsymbol{w}}_{\alpha,\boldsymbol{x}}\|^2_{\mathcal{H}_k}.$$

Taking square roots yields the expression in point 2, which ends the proof. □

Lemma 7. *Assume $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ satisfies Assumption 1 and is sufficiently smooth so that, for every multi-index α with $|\alpha| \le 4$, the derivative evaluation functional $L_{\alpha,\boldsymbol{x}}(f) = \partial^\alpha f(\boldsymbol{x})$ is bounded on $\mathcal{H}$ with Riesz representer $r_{\alpha,\boldsymbol{x}} = \partial^\alpha_{\boldsymbol{x}} k(\cdot,\boldsymbol{x}) \in \mathcal{H}$ (Lemma 4). Define the uniform derivative-evaluation constants*

$$\kappa_3 = \max_{1\le i\le d} \sup_{\boldsymbol{x}\in\mathcal{X}} \left\|\partial^3_{x_i} k(\cdot,\boldsymbol{x})\right\|_{\mathcal{H}}, \tag{E.2}$$

$$\kappa_4 = \max_{|\alpha|=4} \sup_{\boldsymbol{x}\in\mathcal{X}} \left\|\partial^\alpha_{\boldsymbol{x}} k(\cdot,\boldsymbol{x})\right\|_{\mathcal{H}}. \tag{E.3}$$

Fix $\boldsymbol{x} \in \mathcal{X}$ and a radius $h > 0$ small enough that all stencil points below lie in $\mathcal{X}$, and let

$$\mathcal{Z}_h(\boldsymbol{x}) = \{\boldsymbol{x}\} \cup \{\boldsymbol{x} \pm h\boldsymbol{e}_i : 1 \le i \le d\} \cup \{\boldsymbol{x} \pm h(\boldsymbol{e}_i + \boldsymbol{e}_j) : 1 \le i < j \le d\}, \tag{E.4}$$

where $\{\boldsymbol{e}_i\}_{i=1}^d$ are the standard basis vectors. Consider noiseless GP regression with prior $f \sim \mathcal{GP}(0, k)$ and observations $f(\mathcal{Z}_h(\boldsymbol{x}))$. Then, there exist finite constants $C_{g,k}$ and $C_{H,k}$ (depending only on k and d through κ_3, κ_4) such that, for all sufficiently small h,

$$\pi^{\boldsymbol{g}}_{\mathcal{Z}_h(\boldsymbol{x})}(\boldsymbol{x}) \le C_{g,k}\, h^4, \qquad \pi^{\boldsymbol{H}}_{\mathcal{Z}_h(\boldsymbol{x})}(\boldsymbol{x}) \le C_{H,k}\, h^4. \tag{E.5}$$

Proof. The proof proceeds in four steps: (i) construction of Taylor expansions under centered differences with explicit remainders, (ii) showing how we can build centered difference rules using only $\mathcal{Z}_h(\boldsymbol{x})$, (iii) bounding the remainder terms via RKHS derivative-evaluation constants, (iv) taking a worst-case supremum over $\{f \in \mathcal{H} : \|f\|_{\mathcal{H}} \le 1\}$ and invoking the optimal worst-case error identity shown in Lemma 6.

Step (i): Taylor series remainders under centered differences. Fix any direction $\boldsymbol{u} \in \mathbb{R}^d$ and define the 1D restriction $\varphi(t) = f(\boldsymbol{x} + t\boldsymbol{u})$. Applying Taylor's theorem, e.g., (Firey, 1960) with remainder to φ at $t = 0$ gives (for some $\xi_+ \in (0, h)$ and $\xi_- \in (-h, 0)$) the standard centered-difference remainder identities are:

$$\frac{\varphi(h) - \varphi(-h)}{2h} - \varphi'(0) = \frac{h^2}{12}\Big(\varphi^{(3)}(\xi_+) + \varphi^{(3)}(\xi_-)\Big), \tag{E.6}$$

$$\frac{\varphi(h) - 2\varphi(0) + \varphi(-h)}{h^2} - \varphi''(0) = \frac{h^2}{24}\Big(\varphi^{(4)}(\xi_+) + \varphi^{(4)}(\xi_-)\Big). \tag{E.7}$$

From (E.6) and (E.7), we obtain the bounds

$$\left|\frac{\varphi(h) - \varphi(-h)}{2h} - \varphi'(0)\right| \le \frac{h^2}{6} \sup_{|t|\le h} |\varphi^{(3)}(t)|, \tag{E.8}$$

$$\left|\frac{\varphi(h) - 2\varphi(0) + \varphi(-h)}{h^2} - \varphi''(0)\right| \le \frac{h^2}{12} \sup_{|t|\le h} |\varphi^{(4)}(t)|. \tag{E.9}$$

Step (ii): centered difference rules for gradient and Hessian. We now specialize $\boldsymbol{u}$ and map the one-dimensional quantities back to multivariate partial derivatives.

(Gradient components). For each $i \in \{1, \dots, d\}$, take $\boldsymbol{u} = \boldsymbol{e}_i$ and note that $\varphi'(0) = \partial_i f(\boldsymbol{x})$ and

$$\frac{\varphi(h) - \varphi(-h)}{2h} = \frac{f(\boldsymbol{x} + h\boldsymbol{e}_i) - f(\boldsymbol{x} - h\boldsymbol{e}_i)}{2h} = \widehat{\partial_i f}(\boldsymbol{x}; h),$$

which is a linear rule supported on $\{\boldsymbol{x} \pm h\boldsymbol{e}_i\} \subset \mathcal{Z}_h(\boldsymbol{x})$.

(Diagonal Hessian components). For each i, take $\boldsymbol{u} = \boldsymbol{e}_i$ again and note that $\varphi''(0) = \partial^2_{ii} f(\boldsymbol{x})$ and

$$\frac{\varphi(h) - 2\varphi(0) + \varphi(-h)}{h^2} = \frac{f(\boldsymbol{x} + h\boldsymbol{e}_i) - 2f(\boldsymbol{x}) + f(\boldsymbol{x} - h\boldsymbol{e}_i)}{h^2} = \widehat{\partial^2_{ii} f}(\boldsymbol{x}; h),$$

a linear rule supported on $\{\boldsymbol{x}, \boldsymbol{x} \pm h\boldsymbol{e}_i\} \subset \mathcal{Z}_h(\boldsymbol{x})$.

(Off-diagonal Hessian components). Fix $i < j$ and let $\boldsymbol{u} = \boldsymbol{e}_i + \boldsymbol{e}_j$. The directional second derivative satisfies

$$D^2_{\boldsymbol{u}} f(\boldsymbol{x}) = \frac{d^2}{dt^2} f(\boldsymbol{x} + t\boldsymbol{u})\bigg|_{t=0} = \partial^2_{ii} f(\boldsymbol{x}) + 2\partial^2_{ij} f(\boldsymbol{x}) + \partial^2_{jj} f(\boldsymbol{x}), \tag{E.10}$$

such that

$$\partial^2_{ij} f(\boldsymbol{x}) = \frac{1}{2}\Big(D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f(\boldsymbol{x}) - \partial^2_{ii} f(\boldsymbol{x}) - \partial^2_{jj} f(\boldsymbol{x})\Big). \tag{E.11}$$

Define the directional centered second-difference estimator

$$\widehat{D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j}} f(\boldsymbol{x}; h) = \frac{f(\boldsymbol{x} + h(\boldsymbol{e}_i + \boldsymbol{e}_j)) - 2f(\boldsymbol{x}) + f(\boldsymbol{x} - h(\boldsymbol{e}_i + \boldsymbol{e}_j))}{h^2},$$

which uses $\{\boldsymbol{x}, \boldsymbol{x} \pm h(\boldsymbol{e}_i + \boldsymbol{e}_j)\} \subset \mathcal{Z}_h(\boldsymbol{x})$ by construction. Then define the mixed-partial estimator by plugging centered estimators into (E.11):

$$\widehat{\partial^2_{ij} f}(\boldsymbol{x}; h) = \frac{1}{2}\Big(\widehat{D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j}} f(\boldsymbol{x}; h) - \widehat{\partial^2_{ii} f}(\boldsymbol{x}; h) - \widehat{\partial^2_{jj} f}(\boldsymbol{x}; h)\Big). \tag{E.12}$$

The right-hand side is a linear rule supported on the union of $\{\boldsymbol{x} \pm h(\boldsymbol{e}_i + \boldsymbol{e}_j)\}$, $\{\boldsymbol{x} \pm h\boldsymbol{e}_i\}$, $\{\boldsymbol{x} \pm h\boldsymbol{e}_j\}$, and $\{\boldsymbol{x}\}$, all contained in $\mathcal{Z}_h(\boldsymbol{x})$.

Step (iii): bounding Taylor remainders via RKHS derivative constants. We now bound the derivatives appearing in the remainder terms in (E.8)–(E.9) by RKHS norms. By Lemma 4 and the Cauchy-Schwarz inequality, for any multi-index α,

$$|\partial^\alpha f(\boldsymbol{y})| = |\langle f, \partial^\alpha_{\boldsymbol{y}} k(\cdot, \boldsymbol{y})\rangle_{\mathcal{H}}| \le \|f\|_{\mathcal{H}} \, \|\partial^\alpha_{\boldsymbol{y}} k(\cdot, \boldsymbol{y})\|_{\mathcal{H}}, \qquad \forall \boldsymbol{y} \in \mathcal{X}. \tag{E.13}$$

In particular, if $\|f\|_{\mathcal{H}} \le 1$, then $|\partial^\alpha f(\boldsymbol{y})| \le \kappa_4$ for all $|\alpha| = 4$ and $|\partial^3_{x_i} f(\boldsymbol{y})| \le \kappa_3$ for each i and all $\boldsymbol{y}$.

Gradient remainder. With $\boldsymbol{u} = \boldsymbol{e}_i$, we have $\varphi^{(3)}(t) = \partial^3_{iii} f(\boldsymbol{x} + t\boldsymbol{e}_i)$. Thus, for $\|f\|_{\mathcal{H}} \le 1$,

$$\sup_{|t| \le h} |\varphi^{(3)}(t)| \le \kappa_3.$$

Plugging into (E.8) yields

$$\sup_{\|f\|_{\mathcal{H}} \le 1} \Big|\partial_i f(\boldsymbol{x}) - \widehat{\partial_i f}(\boldsymbol{x}; h)\Big| \le \frac{h^2}{6}\kappa_3. \tag{E.14}$$

Diagonal Hessian remainder. With $\boldsymbol{u} = \boldsymbol{e}_i$, we have $\varphi^{(4)}(t) = \partial^4_{iiii} f(\boldsymbol{x} + t\boldsymbol{e}_i)$, so for $\|f\|_{\mathcal{H}} \le 1$, $\sup_{|t| \le h} |\varphi^{(4)}(t)| \le \kappa_4$. Plugging into (E.9) yields

$$\sup_{\|f\|_{\mathcal{H}} \le 1} \Big|\partial^2_{ii} f(\boldsymbol{x}) - \widehat{\partial^2_{ii} f}(\boldsymbol{x}; h)\Big| \le \frac{h^2}{12}\kappa_4. \tag{E.15}$$

Mixed Hessian remainder. With $\boldsymbol{u} = \boldsymbol{e}_i + \boldsymbol{e}_j$, we have $\varphi^{(4)}(t) = D^4_{\boldsymbol{u}} f(\boldsymbol{x} + t\boldsymbol{u})$. Expanding $D^4_{\boldsymbol{u}}$ in coordinate derivatives yields

$$D^4_{\boldsymbol{e}_i+\boldsymbol{e}_j} f = \partial^4_{iiii} f + 4\partial^4_{iiij} f + 6\partial^4_{iijj} f + 4\partial^4_{ijjj} f + \partial^4_{jjjj} f, \tag{E.16}$$

so by the triangle inequality and (E.13),

$$\sup_{\|f\|_{\mathcal{H}} \le 1} \sup_{|t| \le h} |D^4_{\boldsymbol{e}_i+\boldsymbol{e}_j} f(\boldsymbol{x} + t(\boldsymbol{e}_i + \boldsymbol{e}_j))| \le (1 + 4 + 6 + 4 + 1)\kappa_4 = 16\kappa_4. \tag{E.17}$$

Applying (E.9) with this $\boldsymbol{u}$ gives

$$\sup_{\|f\|_{\mathcal{H}} \le 1} \Big|D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f(\boldsymbol{x}) - \widehat{D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j}} f(\boldsymbol{x}; h)\Big| \le \frac{h^2}{12} \cdot 16\kappa_4 = \frac{4}{3}\kappa_4 \, h^2. \tag{E.18}$$

Combining (E.11)–(E.12) with (E.15) and (E.18), then taking the supremum over $\|f\|_{\mathcal{H}} \le 1$, yields

$$\sup_{\|f\|_{\mathcal{H}} \le 1} \Big|\partial^2_{ij} f(\boldsymbol{x}) - \widehat{\partial^2_{ij} f}(\boldsymbol{x}; h)\Big| \le \frac{1}{2}\left(\frac{4}{3}\kappa_4 h^2 + \frac{h^2}{12}\kappa_4 + \frac{h^2}{12}\kappa_4\right) = \frac{3}{4}\kappa_4 \, h^2. \tag{E.19}$$

Specifically, this follows from

$$\begin{aligned}\left|\partial^2_{ij} f(\boldsymbol{x}) - \widehat{\partial^2_{ij} f}(\boldsymbol{x}; h)\right| &= \frac{1}{2}\left(\left|D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f(\boldsymbol{x}) - \widehat{D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f}(\boldsymbol{x}; h) + \partial^2_{ii} f(\boldsymbol{x}) - \widehat{\partial^2_{ii} f}(\boldsymbol{x}; h) + \partial^2_{jj} f(\boldsymbol{x}) - \widehat{\partial^2_{jj} f}(\boldsymbol{x}; h)\right|\right), \\ &\leq \frac{1}{2}\left(\left|D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f(\boldsymbol{x}) - \widehat{D^2_{\boldsymbol{e}_i+\boldsymbol{e}_j} f}(\boldsymbol{x}; h)\right| + \left|\partial^2_{ii} f(\boldsymbol{x}) - \widehat{\partial^2_{ii} f}(\boldsymbol{x}; h)\right| + \left|\partial^2_{jj} f(\boldsymbol{x}) - \widehat{\partial^2_{jj} f}(\boldsymbol{x}; h)\right|\right),\end{aligned}$$

where the second line is based on the triangle inequality and we can then substitute in the previous bounds after taking the supremum over $\|f\|_{\mathcal{H}} \leq 1$.

Step (iv): from worst-case errors to the power functions. By Lemma 6, for each bounded linear functional L (here, each ∂_i and ∂^2_{ij}), we have:

$$\sqrt{\mathrm{Var}(L(f) \mid f(\mathcal{Z}_h(\boldsymbol{x})))} = \inf_{\boldsymbol{w} \in \mathbb{R}^{|\mathcal{Z}_h(\boldsymbol{x})|}} \sup_{f \in \mathcal{H}: \|f\|_H \leq 1} \left| L(f) - \sum_j w_j f(\boldsymbol{z}_j) \right|. \tag{E.20}$$

Therefore, the infimum is upper bounded by the worst-case error of any specific choice of weights. Choosing the central-difference weights constructed in Step (ii) and using (E.14), (E.15), and (E.19), we obtain the following component-wise variance bounds:

$$\mathrm{Var}(\partial_i f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))) \leq \left(\frac{\kappa_3}{6} h^2\right)^2 = \frac{\kappa_3^2}{36} h^4, \tag{E.21}$$

$$\mathrm{Var}\big(\partial^2_{ii} f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))\big) \leq \left(\frac{\kappa_4}{12} h^2\right)^2 = \frac{\kappa_4^2}{144} h^4, \tag{E.22}$$

$$\mathrm{Var}\big(\partial^2_{ij} f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))\big) \leq \left(\frac{3\kappa_4}{4} h^2\right)^2 = \frac{9\kappa_4^2}{16} h^4, \qquad i \neq j. \tag{E.23}$$

Finally, since $\mathrm{tr}(\Sigma)$ is the sum of marginal variances of the corresponding coordinates,

$$\pi^{\boldsymbol{g}}_{\mathcal{Z}_h(\boldsymbol{x})}(\boldsymbol{x}) = \sum_{i=1}^{d} \mathrm{Var}(\partial_i f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))) \leq d \cdot \frac{\kappa_3^2}{36} h^4 = \frac{d\kappa_3^2}{36} h^4, \tag{E.24}$$

proving the gradient bound with $C_{g,k} = d\kappa_3^2/36$. Similarly,

$$\pi^{\boldsymbol{H}}_{\mathcal{Z}_h(\boldsymbol{x})}(\boldsymbol{x}) = \sum_{i=1}^{d} \mathrm{Var}\big(\partial^2_{ii} f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))\big) + \sum_{i \neq j} \mathrm{Var}\big(\partial^2_{ij} f(\boldsymbol{x}) \mid f(\mathcal{Z}_h(\boldsymbol{x}))\big) \tag{E.25}$$

$$\leq d \cdot \frac{\kappa_4^2}{144} h^4 + d(d-1) \cdot \frac{9\kappa_4^2}{16} h^4 = \left(\frac{d\kappa_4^2}{144} + \frac{9d(d-1)\kappa_4^2}{16}\right) h^4, \tag{E.26}$$

so $C_{H,k}$ can be chosen as stated above, which completes the proof. □

Theorem 2 (VPC under NeST sampling; formal)**.** *Assume the following:*

1. *(*Domain*)* $\mathcal{X} \subset \mathbb{R}^d$ *and for each NeST iteration* t*, there exists* $h_0 > 0$ *such that the stencil proposed in Lemma 7, denoted by* $\boldsymbol{Z}_h(\boldsymbol{x}_t) \subset \mathcal{X}$*, for all* $h \in (0, h_0]$*.*
2. *(*Kernel regularity*) The kernel* k *satisfies Assumption 1 strong enough for Lemma 4 to hold for all* $|\alpha| \leq 4$*, and for all stencil radii* $h \in (0, h_0]$ *the corresponding kernel matrices are invertible (e.g.,* k *is strictly positive definite and the stencil points are distinct).*

Fix any sequence $\{\widehat{s}_t\}_{t \geq 0}$ *with* $\widehat{s}_t > 0$*.*

(Noiseless case, $\sigma^2 = 0$). *Let* $b^\star = d^2 + d + 1$*. For any iteration* t *and any batch size* $b_t \geq b^\star$*,*

$$0 \leq \inf_{\boldsymbol{Z} \in \mathcal{X}^{b_t}} \left(\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t)\right) \leq E_{d,k,\widehat{s}_t,0}(b_t) = 0 \quad \textit{(as an infimum).} \tag{E.27}$$

In particular, for the constructive stencil $\boldsymbol{Z} = \boldsymbol{Z}_h(\boldsymbol{x}_t)$ with any $h \in (0, h_0]$, we have

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) \le (1 + \widehat{s}_t) \, C_k \, h^4, \tag{E.28}$$

for some constant $C_k \in (0, \infty)$ depending only on k and d. Hence the NeST objective can be made arbitrarily small by shrinking $h \to 0$.

(Noisy case with replication, $\sigma^2 > 0$). *For any $\epsilon > 0$ and any iteration t, there exist $h \in (0, h_0]$ and an integer $m \ge 1$ such that, using the replicated stencil with m replicates per stencil location (thus batch size $b_t = mb^\star$),*

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h^{(m)}(\boldsymbol{x}_t)}(\boldsymbol{x}_t) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h^{(m)}(\boldsymbol{x}_t)}(\boldsymbol{x}_t) \le \epsilon. \tag{E.29}$$

Consequently, VPC holds in both noiseless and noisy settings.

Proof. We break the proof into two parts; we first prove the claimed result for noiseless function evaluations and then use this result to prove the one claimed for noisy evaluations.

Noiseless case. Fix iteration t and batch size $b_t \ge b^\star$. By definition (see Lemma 3), $E_{d,k,\widehat{s}_t,0}(b_t)$ is an infimum of a sum of posterior variances, hence $E_{d,k,\widehat{s}_t,0}(b_t) \ge 0$. To show the infimum equals 0, fix any $h \in (0, h_0]$ and consider the (origin-centered) central-difference stencil $\boldsymbol{Z}_h(\boldsymbol{0}) \subset \mathcal{X}$ from Lemma 7, which contains exactly $b^\star$ distinct points. If $b_t = b^\star$, set $\boldsymbol{Z}^{(b_t)} = \boldsymbol{Z}_h(\boldsymbol{0})$. If $b_t > b^\star$, augment $\boldsymbol{Z}_h(\boldsymbol{0})$ with any additional $(b_t - b^\star)$ distinct points in $\mathcal{X}$ and denote the resulting b_t-point design by $\boldsymbol{Z}^{(b_t)}$. By monotonicity under conditioning of posterior variances (conditioning on more points cannot increase variance),

$$\pi^{\boldsymbol{g}}_{\boldsymbol{Z}^{(b_t)}}(\boldsymbol{0}) \le \pi^{\boldsymbol{g}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}), \qquad \pi^{\boldsymbol{H}}_{\boldsymbol{Z}^{(b_t)}}(\boldsymbol{0}) \le \pi^{\boldsymbol{H}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}).$$

Therefore, using that error function E is an infimum over all b_t-point designs,

$$E_{d,k,\widehat{s}_t,0}(b_t) \le \pi^{\boldsymbol{g}}_{\boldsymbol{Z}^{(b_t)}}(\boldsymbol{0}) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\boldsymbol{Z}^{(b_t)}}(\boldsymbol{0}) \le \pi^{\boldsymbol{g}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}).$$

Applying Lemma 7 at $\boldsymbol{x} = \boldsymbol{0}$ yields

$$\pi^{\boldsymbol{g}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}) \le C_{g,k} h^4, \qquad \pi^{\boldsymbol{H}}_{\boldsymbol{Z}_h(\boldsymbol{0})}(\boldsymbol{0}) \le C_{H,k} h^4.$$

Hence, $E_{d,k,\widehat{s}_t,0}(b_t) \le \left(C_{g,k} + \widehat{s}_t C_{H,k}\right) h^4$. Since $h \in (0, h_0]$ was arbitrary, letting $h \to 0$ implies $E_{d,k,\widehat{s}_t,0}(b_t) = 0$ as an infimum. Non-negativity of the left-hand side in (E.27) is immediate because power functions are posterior variances. The upper bound

$$\inf_{\boldsymbol{Z} \in \mathcal{X}^{b_t}} \left(\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) \right) \le E_{d,k,\widehat{s}_t,0}(b_t)$$

is exactly Lemma 3 specialized to $\sigma^2 = 0$ and $s = \widehat{s}_t$. Combining with the infimum result gives (E.27), and in particular shows the NeST design-objective infimum equals 0.

To show the next bound in (E.28), fix any $h \in (0, h_0]$ and set $\boldsymbol{Z} = \boldsymbol{Z}_h(\boldsymbol{x}_t)$. By monotonicity under conditioning (i.e., conditioning on $\mathcal{D}_t$ can only reduce variances), we have

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) \le \pi^{\boldsymbol{g}}_{\boldsymbol{Z}}(\boldsymbol{x}_t), \qquad \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}}(\boldsymbol{x}_t) \le \pi^{\boldsymbol{H}}_{\boldsymbol{Z}}(\boldsymbol{x}_t).$$

By stationarity of the kernel (Assumption 1), translating the entire configuration by $-\boldsymbol{x}_t$ and then applying Lemma 7 at $\boldsymbol{x} = \boldsymbol{0}$ gives

$$\pi^{\boldsymbol{g}}_{\boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) \le C_{g,k} h^4, \qquad \pi^{\boldsymbol{H}}_{\boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) \le C_{H,k} h^4.$$

Combining these and setting $C_k = \max\{C_{g,k}, C_{H,k}\}$ yields (E.28):

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) + \widehat{s}_t \, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}_h(\boldsymbol{x}_t)}(\boldsymbol{x}_t) \le (1 + \widehat{s}_t) \, C_k \, h^4.$$

This concludes the noiseless part of the theorem.

Noisy case with replication. Let $\boldsymbol{Z} = (\boldsymbol{z}_1, \dots, \boldsymbol{z}_n)$ denote the $n = b^\star$ *distinct* central-difference stencil locations constructed around $\boldsymbol{x}_t$ with stencil radius $h > 0$. At each $\boldsymbol{z}_j$, we collect $m \geq 1$ replicate observations

$$y_{j,\ell} \;=\; f(\boldsymbol{z}_j) + \varepsilon_{j,\ell}, \qquad \varepsilon_{j,\ell} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \qquad j = 1, \dots, n,\ \ell = 1, \dots, m,$$

independent across (j, ℓ) and independent of the prior $f \sim \mathcal{GP}(0, k)$. Define the sample mean at each location

$$\bar{y}_j \;=\; \frac{1}{m} \sum_{\ell=1}^{m} y_{j,\ell} \;=\; f(\boldsymbol{z}_j) + \bar{\varepsilon}_j, \qquad \bar{\varepsilon}_j = \frac{1}{m} \sum_{\ell=1}^{m} \varepsilon_{j,\ell}.$$

Then $\bar{\varepsilon}_j \sim \mathcal{N}(0, \sigma^2/m)$ and the $\bar{\varepsilon}_j$ are independent across j.

We now justify that conditioning on all replicates $(y_{j,\ell})$ is equivalent (for the posterior of f and of any linear functional of f) to conditioning only on the averaged data $(\bar{y}_j)$.

For each j, define the replicate vector $\boldsymbol{y}_j = (y_{j,1}, \dots, y_{j,m})^\top \in \mathbb{R}^m$ and $\boldsymbol{\varepsilon}_j = (\varepsilon_{j,1}, \dots, \varepsilon_{j,m})^\top \in \mathbb{R}^m$, so that $\boldsymbol{y}_j = f(\boldsymbol{z}_j)\mathbf{1}_m + \boldsymbol{\varepsilon}_j$, where $\mathbf{1}_m$ is the m-vector of ones. Let $T \in \mathbb{R}^{m \times m}$ be any orthonormal matrix whose first row is $\mathbf{1}_m^\top / \sqrt{m}$. Define the orthogonal transform $\tilde{\boldsymbol{y}}_j = T\boldsymbol{y}_j$ and $\tilde{\boldsymbol{\varepsilon}}_j = T\boldsymbol{\varepsilon}_j$. Since T is orthonormal and $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(0, \sigma^2 I_m)$, we have $\tilde{\boldsymbol{\varepsilon}}_j \sim \mathcal{N}(0, \sigma^2 I_m)$. Moreover

$$\tilde{y}_{j,1} \;=\; \frac{1}{\sqrt{m}} \mathbf{1}_m^\top \boldsymbol{y}_j \;=\; \sqrt{m}\, f(\boldsymbol{z}_j) + \tilde{\varepsilon}_{j,1}, \qquad \tilde{y}_{j,r} \;=\; \tilde{\varepsilon}_{j,r} \quad (r = 2, \dots, m).$$

Thus, for each j, the $(m-1)$ coordinates $\tilde{y}_{j,2}, \dots, \tilde{y}_{j,m}$ are *pure noise* (independent of $f(\boldsymbol{z}_j)$), and hence carry no information about f. Therefore, for the posterior over f (and any functionals of f), conditioning on $\{\boldsymbol{y}_j\}_{j=1}^n$ is equivalent to conditioning on $\{\tilde{y}_{j,1}\}_{j=1}^n$, i.e., on $\{\bar{y}_j\}_{j=1}^n$. Concretely, we can treat the replicate experiment as the reduced observation model

$$\bar{\boldsymbol{y}} \;=\; \boldsymbol{f}_{\boldsymbol{Z}} + \bar{\boldsymbol{\varepsilon}}, \qquad \bar{\boldsymbol{\varepsilon}} \sim \mathcal{N}\Big(0, \tfrac{\sigma^2}{m} I_n\Big),$$

where $\boldsymbol{f}_{\boldsymbol{Z}} = (f(\boldsymbol{z}_1), \dots, f(\boldsymbol{z}_n))^\top$.

Now, fix any multi-index α and define the derivative functional $L_{\alpha, \boldsymbol{x}_t}(f) = \partial^\alpha f(\boldsymbol{x}_t)$. By Lemma 4, $L_{\alpha, \boldsymbol{x}_t}$ is bounded and linear on $\mathcal{H}$, and the relevant cross-covariance vector with the (distinct) sites $\boldsymbol{Z}$ is

$$\boldsymbol{k}_\alpha = \big(\partial^\alpha_{\boldsymbol{x}} k(\boldsymbol{x}_t, \boldsymbol{z}_1), \dots, \partial^\alpha_{\boldsymbol{x}} k(\boldsymbol{x}_t, \boldsymbol{z}_n) \big)^\top \in \mathbb{R}^n,$$

and the Gram matrix is $K = k(\boldsymbol{Z}, \boldsymbol{Z}) \in \mathbb{R}^{n \times n}$. Under the reduced model $\bar{\boldsymbol{y}} = \boldsymbol{f}_{\boldsymbol{Z}} + \bar{\boldsymbol{\varepsilon}}$ with $\bar{\boldsymbol{\varepsilon}} \sim \mathcal{N}(0, (\sigma^2/m) I_n)$, standard Gaussian conditioning gives the posterior variance

$$\operatorname{Var}(L_{\alpha, \boldsymbol{x}_t}(f) \,|\, \bar{\boldsymbol{y}}) \;=\; L^{(1)}_{\alpha, \boldsymbol{x}_t} L^{(2)}_{\alpha, \boldsymbol{x}_t} k(\boldsymbol{x}_t, \boldsymbol{x}_t) \;-\; \boldsymbol{k}_\alpha^\top \Big(K + \tfrac{\sigma^2}{m} I_n \Big)^{-1} \boldsymbol{k}_\alpha, \tag{E.30}$$

where $L^{(1)}$ and $L^{(2)}$ indicate that the derivative operator is applied to the first and second argument, respectively.

Define the *noisy* power function for this derivative as

$$\pi_{\boldsymbol{Z}, \sigma^2/m}(\alpha, \boldsymbol{x}_t) = \operatorname{Var}(L_{\alpha, \boldsymbol{x}_t}(f) \,|\, \bar{\boldsymbol{y}}),$$

and similarly let $\pi_{\boldsymbol{Z}, 0}(\alpha, \boldsymbol{x}_t)$ denote the noiseless version (i.e., with $\sigma^2/m = 0$).

Let $\tau^2 = \sigma^2/m$. For fixed distinct $\boldsymbol{Z}$, we have $K \succ 0$ (strictly positive definite), and therefore $(K + \tau^2 I_n)^{-1}$ is well-defined for all $\tau^2 \geq 0$. Moreover,

$$0 \leq \pi_{\boldsymbol{Z}, \tau^2}(\alpha, \boldsymbol{x}_t) - \pi_{\boldsymbol{Z}, 0}(\alpha, \boldsymbol{x}_t) = \boldsymbol{k}_\alpha^\top \Big(K^{-1} - (K + \tau^2 I_n)^{-1} \Big) \boldsymbol{k}_\alpha.$$

Using the difference of inverse identity

$$K^{-1} - (K + \tau^2 I_n)^{-1} = K^{-1} (\tau^2 I_n) (K + \tau^2 I_n)^{-1},$$

we obtain the bound

$$0 \leq \pi_{\boldsymbol{Z}, \tau^2}(\alpha, \boldsymbol{x}_t) - \pi_{\boldsymbol{Z}, 0}(\alpha, \boldsymbol{x}_t) \leq \tau^2 \|K^{-1}\|_2 \|(K + \tau^2 I_n)^{-1}\|_2 \|\boldsymbol{k}_\alpha\|_2^2 \leq \tau^2 \|K^{-1}\|_2^2 \|\boldsymbol{k}_\alpha\|_2^2. \tag{E.31}$$

In particular, for fixed $\boldsymbol{Z}$, this implies $\pi_{\boldsymbol{Z},\sigma^2/m}(\alpha, \boldsymbol{x}_t) \to \pi_{\boldsymbol{Z},0}(\alpha, \boldsymbol{x}_t)$ as $m \to \infty$.

Now choose $\boldsymbol{Z} = \boldsymbol{Z}_h(\boldsymbol{x}_t)$; we can combine this with our noiseless result to see that the increase in error for each derivative-component variance is $O(\sigma^2/m)$ for fixed h. Since there are finitely many gradient and Hessian components, we can sum (E.31) over all components and absorb the resulting finite sums into a constant $C'_k(h)$ (finite for each fixed h), to obtain:

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}(h),\sigma^2/m}(\boldsymbol{x}_t) \le C_{g,k} h^4 + C'_{g,k}(h)\,\frac{\sigma^2}{m}, \qquad \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}(h),\sigma^2/m}(\boldsymbol{x}_t) \le C_{H,k} h^4 + C'_{H,k}(h)\frac{\sigma^2}{m}.$$

Therefore,

$$\pi^{\boldsymbol{g}}_{\mathcal{D}_t \cup \boldsymbol{Z}(h),\sigma^2/m}(\boldsymbol{x}_t) + \widehat{s}_t\, \pi^{\boldsymbol{H}}_{\mathcal{D}_t \cup \boldsymbol{Z}(h),\sigma^2/m}(\boldsymbol{x}_t) \le (1+\widehat{s}_t)\, C_k h^4 + (1+\widehat{s}_t)\, C'_k(h)\frac{\sigma^2}{m}.$$

for some $C_k, C'_k(h) \in (0,\infty)$. Given any $\varepsilon > 0$, first choose h small enough so that $(1+\widehat{s}_t)C_k h^4 \le \varepsilon/2$, and then choose m large enough so that $(1+\widehat{s}_t)C'_k(h)\sigma^2/m \le \varepsilon/2$. This proves that the NeST objective can be made arbitrarily small in the noisy case, which completes the proof. □

E.2 Local Quadratic Convergence

We now analyze the local convergence behavior of NeST-BO, which follows from relatively standard results for (inexact) Newton's method. The main difference here is that we have a data-driven Newton-step error.

Theorem 3 (Local quadratic convergence with NeST)**.** *Assume f is twice differentiable, $\boldsymbol{H}$ is β-Lipschitz on a neighborhood $\mathcal{N}$ of a local minimizer $\boldsymbol{x}^\star$, and $\lambda_{\min}(\boldsymbol{H}(\boldsymbol{x})) \ge \lambda_{\min} > 0$ on $\mathcal{N}$. Consider the full-step NeST update $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \widehat{\boldsymbol{d}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ with Newton-step error $\varepsilon_t = \|\boldsymbol{d}(\boldsymbol{x}_t) - \widehat{\boldsymbol{d}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)\|$. Then, for any $\boldsymbol{x}_t \in \mathcal{N}$, we have*

$$\|\boldsymbol{x}_{t+1} - \boldsymbol{x}^\star\| \le \frac{\beta}{2\,\lambda_{\min}}\,\|\boldsymbol{x}_t - \boldsymbol{x}^\star\|^2 + \varepsilon_t, \qquad \varepsilon_t^2 \le 2B^2 \left\|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\right\|^2 \Phi_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t),$$

where $\Phi_{\mathcal{D}}(\boldsymbol{x}) = \pi^{\boldsymbol{g}}_{\mathcal{D}}(\boldsymbol{x}) + s_{\mathcal{D}}(\boldsymbol{x})\,\pi^{\boldsymbol{H}}_{\mathcal{D}}(\boldsymbol{x})$. In particular, if $\varepsilon_t \le \kappa\|\boldsymbol{x}_t - \boldsymbol{x}^\star\|^2$ eventually for $\kappa > 0$ (e.g., under VPC), the iterates enter the quadratic regime.

Proof. Write $\widehat{\boldsymbol{d}}_t = \widehat{\boldsymbol{d}}_{\mathcal{D}_{t+1}}(\boldsymbol{x}_t)$ and decompose the updated iterate as follows

$$\boldsymbol{x}_{t+1} - \boldsymbol{x}^\star = \left[\boldsymbol{x}_t - \boldsymbol{x}^\star - \boldsymbol{H}(\boldsymbol{x}_t)^{-1}\boldsymbol{g}(\boldsymbol{x}_t)\right] + \left[\boldsymbol{d}(\boldsymbol{x}_t) - \widehat{\boldsymbol{d}}_t\right].$$

The second bracket is ε_t by definition. For the first bracket, inexact-Newton analysis with β-Lipschitz Hessian gives the following sequence of equalities/inequalities

$$\begin{aligned}
\|\boldsymbol{x}_t - \boldsymbol{x}^\star - \boldsymbol{H}(\boldsymbol{x}_t)^{-1}\boldsymbol{g}(\boldsymbol{x}_t)\| &= \|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\left(\boldsymbol{H}(\boldsymbol{x}_t)(\boldsymbol{x}_t - \boldsymbol{x}^\star) - \boldsymbol{g}(\boldsymbol{x}_t)\right)\| \\
&= \|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\left(\boldsymbol{g}(\boldsymbol{x}^\star) - \boldsymbol{g}(\boldsymbol{x}_t) - \boldsymbol{H}(\boldsymbol{x}_t)(\boldsymbol{x}_t - \boldsymbol{x}^\star)\right)\|, \\
&\le \|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\| \cdot \|\boldsymbol{g}(\boldsymbol{x}^\star) - \boldsymbol{g}(\boldsymbol{x}_t) - \boldsymbol{H}(\boldsymbol{x}_t)(\boldsymbol{x}_t - \boldsymbol{x}^\star)\|, \\
&\le \|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\| \cdot \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{x}^\star\|^2, \\
&\le \frac{\beta}{2\lambda_{\min}}\|\boldsymbol{x}_t - \boldsymbol{x}^\star\|^2,
\end{aligned}$$

where the first line follows from simple rearrangement, the second line follows from $\boldsymbol{g}(\boldsymbol{x}^\star) = \boldsymbol{0}$ since $\boldsymbol{x}^\star$ is a local minimizer, the third line follows from standard norm inequalities, the fourth line follows from β-Lipschitz condition on the Hessian (Nesterov and Polyak, 2006, Lemma 1), and the final line follows from $\|\boldsymbol{H}(\boldsymbol{x}_t)^{-1}\| \le 1/\lambda_{\min}$. The bound on ε_t follows from Theorem 1; the stated result follows from these two bounds. □

F EXPERIMENT DETAILS

F.1 Implementation

Reproducibility. The code to reproduce the main components of the numerical experiments is avialable on GitHub: `https://github.com/PaulsonLab/NeST-BO`.

Software packages and shared settings. Unless otherwise stated, all BO baselines are implemented using the `BoTorch` (version 0.15)[1] (Balandat et al., 2020) and `GPyTorch`[2] (version 1.14) (Gardner et al., 2018) packages. We use squared exponential (SE) kernels with automatic relevance determination (ARD) throughout for a controlled comparison across methods. Acquisition optimization in `BoTorch` is performed with the `optimize_acqf` function using `num_restarts` $= 5$ and `raw_samples` $= 20$; for the very high-dimensional problems with $d \geq 1000$ (e.g., Ant, Leukemia), we add a 2 second timeout and reduce `num_restarts` to 3 (only on the Leukemia problem) to limit wall-clock cost. Hyperparameters are refit at different frequencies by task class: every move for directional local methods, every d iterations on our 20d synthetic tasks and the Lunar Lander, Swimmer, Robot Pusher, and Rover Trajectory benchmarks; every iteration for 1000d synthetic tasks; and every 10 iterations for Ant and Leukemia (see Appendix F.3–F.4 for task definitions).

Initialization and starting location. Initial designs use Sobol sequences over the full domain and always include the starting point (for local BO methods). Following prior local BO work, we start directional methods from the domain center on the real-world problems and from a random point on synthetic tasks, since the center can coincide with the global solution on some synthetic functions. Note that Sobol sampling uses the standard `torch.quasirandom.SobolEngine`.

Local optimization of NeST. Because the NeST acquisition (8) targets the Newton step at $\boldsymbol{x}_t$ and our kernels are stationary, informative experimental designs concentrate fairly close to the current iterate. We therefore optimize $\widehat{\alpha}_{\mathrm{NeST}}$ within a small box centered at $\boldsymbol{x}_t$ with radius δ_t, i.e., search domain $[\boldsymbol{x}_t - \delta_t, \boldsymbol{x}_t + \delta_t]$. In principle, δ_t can be adapted using standard model-agreement tests from trust-region methods; in all experiments we use a fixed radius for simplicity and speed. We set $\delta = 0.2$ in most tasks and $\delta = 0.01$ on Ant due to strong non-stationarity.

Batch sizes for directional methods. For NeST-BO, GIBO, MPD, and MinUCB we query $b_t = d$ points per iteration to learn the local step/direction (or $b_t = m$ in subspace dimension m for the subspace variant). This choice is supported by the ablations in Appendix G.3.

Method-specific details. Below, we summarize specific implementation details for each method tested in our comparisons throughout this work:

- *NeST-BO:* We implement the practical version of NeST-BO (Algorithm 2) by extending the public GIBO codebase to reuse its BO loop, GP wrappers, and acquisition optimizer, replacing GIBO's GI acquisition with our weighted power function objective in (8) and adding the Newton step update with line search. The acquisition is optimized in the local box as described above; backtracking line search is applied on the GP mean.

- *NeST-BO (subspace variant):* To study compatibility with learned embeddings, we integrate NeST-BO with the BAxUS (Papenmeier et al., 2022) subspace machinery from the implementation provided at `https://botorch.org/docs/tutorials/baxus/`. As opposed to keeping it fixed in all cases, we treat the initial subspace dimension as a tunable hyperparameter. We use 4 for most problems but increased it some for the real-world problems based on some preliminary experimentation. We adopt a similar subspace expansion heuristic of the original BAxUS implementation: if no improvement is found after some number of consecutive iterations, the subspace dimension is expanded. Note that the original implementation expands the subspace dimension once the trust region side length is smaller than a pre-specified threshold value. We set the number of consective iterations to 50 for all synthetic problems, 10 for the Lunar Lander, Swimmer, and Robot Pusher problems (as they are lower dimensional), and 20 for the Rover Trajectory, Ant, and the Leukemia problems.

- *D-scaled LogEI:* We follow the "vanilla BO works" recommendation from (Hvarfner et al., 2024) to use LogEI (Ament et al., 2023) with dimension-aware length-scale priors and standardized outputs; in BoTorch this corresponds to `LogExpectedImprovement` on a `SingleTaskGP` with appropriate priors.

[1] BoTorch: `https://botorch.org/`
[2] GPyTorch: `https://gpytorch.ai/`

- *TuRBO:* We use the (single-trust-region) TuRBO (Eriksson et al., 2019) implementation from the BoTorch tutorial at `https://botorch.org/docs/tutorials/turbo_1/` with LogEI for consistency with the global baseline. TuRBO adaptively shrinks/expands a local box based on success/failure counters, providing strong anytime performance in higher dimensions.
- *GIBO:* The GIBO method (Müller et al., 2021) selects samples that maximally reduce the posterior gradient covariance (GI acquisition), then takes a length-scale-normalized gradient step. We use the original public implementation available at `https://github.com/sarmueller/gibo` and its early-stopping rule for gradient learning to avoid oversampling near the iterate.
- *MPD:* Maximum Probability of Descent (MPD) (Nguyen et al., 2022) chooses directions maximizing the posterior probability that the (normalized) gradient is a descent direction; we use the reference implementation available at `https://github.com/kayween/local-bo-mpd` with step size $\delta = 0.01$ and probability threshold $p^\star = 0.65$.
- *MinUCB:* The MinUCB method (Fan et al., 2024) minimizes a UCB-style surrogate of gradient magnitude along candidate directions; we use the reference implementation available at `https://github.com/chinafzy1/Minimizing-UCB` and default settings from the original paper.
- *BAxUS:* For the standalone BAxUS baseline, we use the BoTorch tutorial code based on (Papenmeier et al., 2022) (same as subspace variant of NeST-BO) with LogEI to match our other baselines; BAxUS initializes a small subspace and enlarges it on stagnation. The tutorial uses a `SingleTaskGP` with log-normal lengthscale priors. We follow the original heuristic to set the subspace dimension, but increase it to 4 in the synthetic problems to avoid initial projections getting an unfair advantage of landing near the global optimum at $\mathbf{0}$.
- *Sobol:* Non-adaptive Sobol sampling uses `torch.quasirandom.SobolEngine` with scrambling enabled, which is a standard baseline method considered in the BO literature.

F.2 Computing Resources

All experiments were executed on the Ohio Supercomputing Center (OSC) cluster (`https://www.osc.edu`) using CPU nodes equipped with Intel Xeon CPU Max 9470 processors and 512 GB RAM.

F.3 Definition of Synthetic Functions

We use four standard test function that jointly probe conditioning, non-convexity, and multi-modality properties that stress local learning of curvature.

Sphere. The d-dimensional Sphere function is a convex quadratic function expressed as:

$$f(\boldsymbol{x}) = \textstyle\sum_{i=1}^{d} x_i^2,$$

which was used as a check for local methods, as it has well-behaved curvature and no local minima. We optimize over $\mathcal{X} = [-d^2, d^2]^d$. The global minimum is $f(\boldsymbol{x}^\star) = 0$ at $\boldsymbol{x}^\star = (0, 0, \ldots, 0)$.

Rosenbrock. The d-dimensional Rosenbrock function is a bowl-shaped function with a narrow, curved valley, which can be expressed as:

$$f(\boldsymbol{x}) = \textstyle\sum_{i=1}^{d-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right].$$

It is a classical "ill-conditioned" test function that is likely to reward updates that incorporate curvature information. We optimize within the bounds $\mathcal{X} = [-5, 5]^d$. The global minimum is $f(\boldsymbol{x}^\star) = 0$ at $\boldsymbol{x}^\star = (1, 1, \ldots, 1)$. This is a common benchmark in the BO literature; see, e.g., (Xu et al., 2024) for example.

Griewank. The d-dimensional Griewank function is a separable quadratic modulated by a product of cosines:

$$f(\boldsymbol{x}) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$

which features many regularly spaced local minima. We optimize within the bounds $[-300, 300]^d$. The global minimum is $f(\boldsymbol{x}^\star) = 0$ at $\boldsymbol{x}^\star = (0, 0, \ldots, 0)$. This has been recently used as a benchmark problem when analyzing high-dimensional BO algorithms; see, e.g., (Papenmeier et al., 2025).

Ackley. The d-dimensional Ackley function is a highly multi-modal landscape with a flat outer region and a steep basin near the global optimum:

$$f(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + \exp(1).$$

We optimize within the bounds $[-5, 5]^d$. The global minimum is $f(\boldsymbol{x}^\star) = 0$ at $\boldsymbol{x}^\star = (0, 0, \ldots, 0)$. The Ackley function is another popular benchmark in both low- and high-dimensional BO works; see, e.g., (Siemenn et al., 2023; Ament et al., 2023).

We highlight that these choices follow common practice in the BO literature and are meant to cover complementary problem aspects: Rosenbrock isolates ill-conditioning (benefiting Newton steps), Griewank/Ackley add dense local structure (testing how fast local surrogates learn gradients *and* curvature), while Sphere confirms that added second-order machinery can still add value on easy, well-conditioned cases.

F.4 Real-World Benchmark Problems

We include six problems spanning reinforcement learning (RL) control, robotic planning, and large-scale hyperparameter tuning. Together they cover medium to very high dimensionality, varying degrees of non-stationarity, and different noise profiles – settings where local curvature can accelerate progress and subspaces can be useful.

Lunar Lander (12d). A classic control task in the OpenAI Gymnasium (`LunarLander-v3`), where a controller with 12 parameters maps the measured state to four discrete actions. Following prior BO studies from, e.g., (Eriksson et al., 2019), we minimize the negative episodic return (reward sign flipped). Episodes are run for 1000 steps; we initialize the GP from 10 Sobol points.

Swimmer (16d). This is a MuJoCo locomotion task in the OpenAI Gymnasium (`Swimmer-v5`) with a linear policy (16 parameters). This probelm has been considered in prior BO studies, e.g., (Müller et al., 2021); we minimize negative reward and run episodes for 1000 steps. We again initialize the GP from 10 Sobol points.

Robot Pushing (14d). This is a planar manipulation benchmark where 14 controller parameters must be tuned to reduce distances to targets. We adopt bounds and setup from prior BO work, reported in (Eriksson et al., 2019), and run it in a small-noise regime to isolate optimization behavior.

Rover Trajectory (60d). A trajectory-planning task – originally introduced in (Wang et al., 2018) – in which 30 two-dimensional waypoints are optimized to maximize a reward that penalizes rough terrain and constraint violations. The resulting 60-dimensional design is structured and non-stationary, which stresses local surrogates and benefits from curvature information. We minimize negative reward and follow the large-domain setting (using 200 Sobol initial points) suggested in the literature.

Ant (888d). This is a MuJoCo quadruped with an 8-dimensional action space and 111-dimensional observations; we optimize a linear state-feedback policy (888 parameters) and minimize negative reward. This benchmark has recently been used to probe high-dimensional BO with subspaces (Hvarfner et al., 2024). In contrast to previous work that neglects contact forces and uses the `Ant-v2` environment, we use `Ant-v4` (in OpenAI Gymnasium) with contact forces enabled, which increases complexity. For NeST-BO-sub and BAxUS, we initialize at the center point of the subspace, which yields an initial objective (negative reward) of approximately -990.

Leukemia (7129d). A weighted Lasso regression task with one weight per feature (7129 hyperparameters) on the Leukemia dataset from LassoBench (Šehić et al., 2022). We follow the standard least-squares objective with weighted ℓ_1 regularization and evaluate test error under the LassoBench protocol. This problem exemplifies extremely high-dimensional, sparse settings where subspace methods are essential.

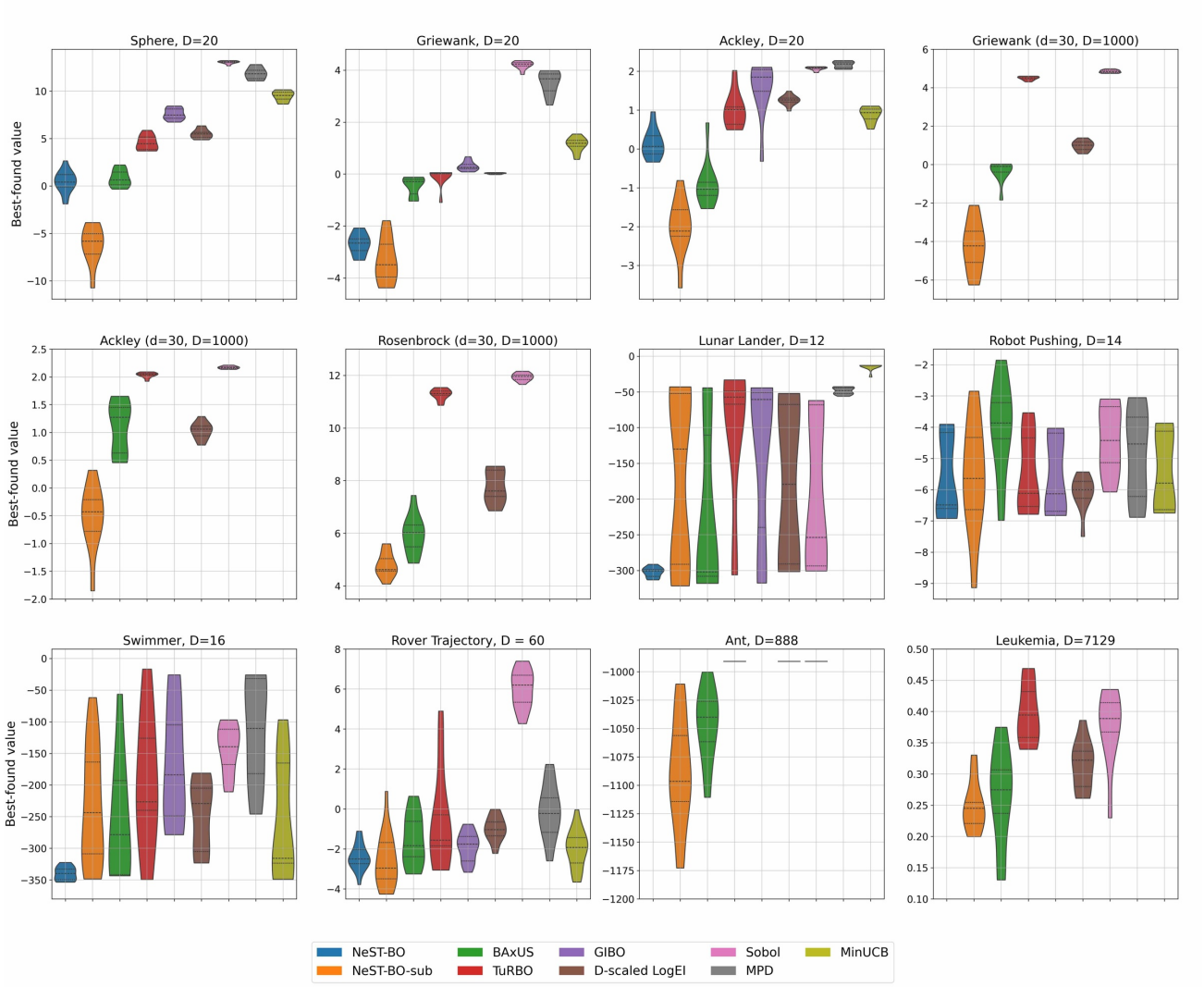

Figure F.1: Final best-found values across tasks. For each test problem, violins show the distribution of the final best-found objective over repeated runs for all methods. Dashed lines mark quartiles (median centered). Lower is better in every panel. The plot provides a compact view of both central tendency and spread at termination, complementing the iteration-wise trajectories in the main text.

The RL tasks (Lunar Lander, Swimmer, Ant) expose NeST-BO to non-stationary, stochastic objectives where local Newton steps and line search stabilize progress; Robot Pushing and Rover emphasize structured geometry and curvature; Leukemia provides a sparse, ultra-high-dimensional regime. This mix lets us isolate when curvature helps (ill-conditioned valleys, non-stationary responses) and when subspaces are essential, and it explains the large empirical gains we report over purely gradient-based local BO and global BO baselines.

F.5 Final Performance Distributions and Summary Statistics

To complement the performance-versus-iteration plots in Figure 2 in the main text, Figure F.1 summarizes, for each benchmark, the empirical distribution of the *final* best-found objective across replicates and methods. Each panel corresponds to one task. Within a panel, one violin plot per method shows the distribution of final outcomes; interior dashed lines denote empirical quartiles (median in the middle). All y-axes are in the native objective scale (negative is better for all tasks). To provide a compact numerical summary consistent with the main text, Tables F.1–F.4 report the median with interquartile range (IQR) (difference between upper and lower quartile) in parentheses for the same final outcomes (10 replicates per method).

Table F.1: Final best-found objective on 20D synthetic benchmarks. Entries are median (IQR) across 10 replicates; lower is better. Best median per column is in **bold**.

Method	Griewank (d=20)	Sphere (d=20)	Ackley (d=20)
NeST-BO	0.0704 (0.0298)	1.37 (1.90)	1.07 (0.53)
NeST-BO-sub	**0.0308 (0.0518)**	**0.0032 (0.0059)**	**0.12 (0.10)**
BAxUS	0.75 (0.33)	1.92 (3.32)	0.36 (0.12)
TuRBO	1.02 (0.01)	86.89 (127.89)	2.76 (1.06)
GIBO	1.30 (0.22)	1,791.79 (2,142.81)	6.38 (3.28)
D-scaled LogEI	1.03 (0.04)	243.39 (120.28)	3.55 (0.36)
Sobol	69.21 (8.78)	485,044.69 (62,421.03)	8.08 (0.32)
MPD	39.53 (22.38)	140,223.66 (117,641.69)	8.93 (1.28)
MinUCB	3.28 (0.74)	14,273.04 (10,468.39)	2.56 (0.63)

Table F.2: Final best-found objective on high-dimensional synthetic benchmarks (ambient D=1000; intrinsic d=30). Entries are median (IQR) across 10 replicates; lower is better. Best median per column is in **bold**.

Method	Griewank (d=30; D=1000)	Ackley (d=30; D=1000)	Rosenbrock (d=30; D=1000)
NeST-BO-sub	**0.02 (0.05)**	**0.65 (0.35)**	**48.82 (22.14)**
BAxUS	0.92 (0.33)	3.57 (2.40)	278.91 (107.26)
TuRBO	91.17 (13.20)	7.77 (0.28)	76,095.20 (19,735.29)
D-scaled LogEI	2.74 (1.03)	2.89 (0.49)	1,437.85 (774.60)
Sobol	126.71 (16.07)	8.75 (0.28)	138,662.22 (12,104.50)

Table F.3: Final best-found objective on simulator benchmarks. Entries are median (IQR) across 10 replicates; lower is better. Best median per column is in **bold**.

Method	Lunar Lander	Robot Pushing	Swimmer	Rover Trajectory
NeST-BO	-300.49 (9.88)	**-6.49 (2.43)**	**-339.62 (16.53)**	-2.26 (0.87)
NeST-BO-sub	-290.62 (18.51)	-5.64 (2.31)	-239.83 (177.65)	**-3.08 (0.89)**
BAxUS	**-301.89 (197.58)**	-3.86 (1.15)	-278.26 (148.83)	-1.83 (1.77)
TuRBO	-57.25 (18.17)	-6.11 (2.19)	-226.57 (114.12)	-1.55 (1.55)
GIBO	-60.47 (189.10)	-6.13 (2.50)	-183.91 (144.05)	-1.91 (0.98)
D-scaled LogEI	-179.18 (223.20)	-6.00 (0.55)	-229.15 (100.17)	-1.12 (0.62)
Sobol	-253.77 (225.42)	-4.42 (1.79)	-139.29 (55.79)	6.36 (0.84)
MPD	-48.05 (7.87)	-4.53 (2.54)	-110.25 (150.48)	-0.25 (1.34)
MinUCB	-12.82 (0)	-5.79 (2.51)	-315.82 (158.33)	-1.95 (1.42)

Table F.4: Final best-found objective on additional benchmarks. Entries are median (IQR) across 10 replicates; lower is better. Best median per column is in **bold**.

Method	Ant	Leukemia
NeST-BO-sub	**-1099.46 (42.28)**	**0.25 (0.03)**
BAxUS	-1044.03 (35.66)	0.27 (0.07)
TuRBO	-990.83 (0)	0.39 (0.07)
D-scaled LogEI	-990.83 (0)	0.32 (0.06)
Sobol	-990.83 (0)	0.39 (0.05)

To complement these descriptive statistics with a paired significance check, we additionally report one-sided paired Wilcoxon signed-rank tests, e.g., (Woolson, 2007) comparing NeST-BO (and NeST-BO-sub) to each baseline on the per-seed final best objective (Tables F.5–F.7). For each task and comparator, the test is applied to the matched differences in terminal outcomes (NeST variant minus baseline) and evaluates whether these paired differences are systematically negative, without assuming normality of the differences. Overall, the Wilcoxon results largely agree with the separation seen in the violin plots (Figure F.1) and the gaps in the median (IQR) tables (Tables F.1–F.4): most comparisons indicate statistically significant improvements for a NeST variant, with a small number of marginal or non-significant cases where methods are not statistically different at termination

Table F.5: Paired one-sided Wilcoxon signed-rank tests on 20D synthetic benchmarks at termination. Each entry reports significance for the hypothesis that the NeST variant attains lower final objective than the comparator, with p-value in parentheses (truncated at 0.001). Symbols: ✓ ($p < 0.05$), ≈ ($0.05 \leq p \leq 0.10$), ✗ ($p > 0.10$).

	Griewank (d=20)		Sphere (d=20)		Ackley (d=20)	
Comparator	NeST-BO	NeST-BO-sub	NeST-BO	NeST-BO-sub	NeST-BO	NeST-BO-sub
NeST-BO	–	≈ (0.08)	–	✓ (0.001)	–	✓ (0.001)
NeST-BO-sub	✗ (0.93)	–	✗ (1.0)	–	✗ (1.0)	–
BAxUS	✓ (0.001)	✓ (0.001)	✗ (0.25)	✓ (0.001)	✗ (1.0)	✓ (0.005)
TuRBO	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)
GIBO	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.002)	✓ (0.001)
D-scaled LogEI	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)
Sobol	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)
MPD	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)
MinUCB	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)

Table F.6: Paired one-sided Wilcoxon signed-rank tests on simulator benchmarks at termination. Each entry tests whether the NeST variant achieves lower final objective than the comparator, with p-value in parentheses (truncated at 0.001). Symbols: ✓ ($p < 0.05$), ≈ ($0.05 \leq p \leq 0.10$), ✗ ($p > 0.10$).

	Lunar Lander		Robot Pushing		Swimmer		Rover Trajectory	
Comparator	NeST-BO	NeST-BO-sub	NeST-BO	NeST-BO-sub	NeST-BO	NeST-BO-sub	NeST-BO	NeST-BO-sub
NeST-BO	–	✗ (0.98)	–	✗ (0.54)	–	✗ (1.0)	–	✗ (0.14)
NeST-BO-sub	✓ (0.03)	–	✗ (0.50)	–	✓ (0.001)	–	✗ (0.88)	–
BAxUS	✗ (0.12)	✗ (0.62)	✓ (0.01)	✓ (0.02)	✓ (0.01)	✗ (0.54)	≈ (0.065)	≈ (0.05)
TuRBO	✓ (0.005)	✓ (0.01)	✗ (0.46)	✗ (0.38)	✓ (0.003)	✗ (0.28)	✓ (0.01)	✓ (0.002)
GIBO	✓ (0.003)	✓ (0.02)	✗ (0.69)	✗ (0.54)	✓ (0.001)	✗ (0.16)	✗ (0.35)	✗ (0.19)
D-scaled LogEI	✓ (0.002)	≈ (0.10)	✗ (0.88)	✗ (0.78)	✓ (0.001)	✗ (0.78)	✓ (0.005)	✓ (0.007)
Sobol	✓ (0.005)	✗ (0.14)	✓ (0.001)	✓ (0.04)	✓ (0.001)	✓ (0.01)	✓ (0.001)	✓ (0.001)
MPD	✓ (0.001)	✓ (0.001)	✓ (0.01)	✗ (0.14)	✓ (0.002)	✓ (0.01)	✓ (0.001)	✓ (0.007)
MinUCB	✓ (0.001)	✓ (0.001)	✗ (0.28)	✗ (0.50)	✓ (0.002)	✗ (0.72)	✗ (0.62)	✗ (0.25)

Table F.7: Paired one-sided Wilcoxon signed-rank tests for NeST-BO-sub at termination on high-dimensional synthetic benchmarks (D=1000, d=30) and additional tasks. Each entry tests whether NeST-BO-sub achieves lower final objective than the comparator, with p-value in parentheses (truncated at 0.001). Symbols: ✓ ($p < 0.05$), ≈ ($0.05 \leq p \leq 0.10$), ✗ ($p > 0.10$).

	Griewank	Ackley	Rosenbrock	Ant	Leukemia
BAxUS	✓ (0.001)	✓ (0.001)	✓ (0.003)	✓ (0.002)	✗ (0.25)
TuRBO	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)
D-scaled LogEI	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.005)
Sobol	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)	✓ (0.001)

(beyond a standard p-value of 0.05).

G ADDITIONAL EXPERIMENTS AND ABLATIONS

G.1 GP Prior Realizations

We consider a similar study to that in (Müller et al., 2021) wherein we optimize samples drawn from a GP prior. We take a GP prior with zero mean and the squared exponential (SE) kernel with unit variance. To vary difficulty with dimension d, we draw the kernel length-scale $\ell(d)$ uniformly over a narrow interval centered at the heuristic used by (Müller et al., 2021, Appendix A.5), and keep $\ell(d)$ fixed within each realization.

Rather than fitting a surrogate to finite prior samples, we *directly* sample functions from the prior using random

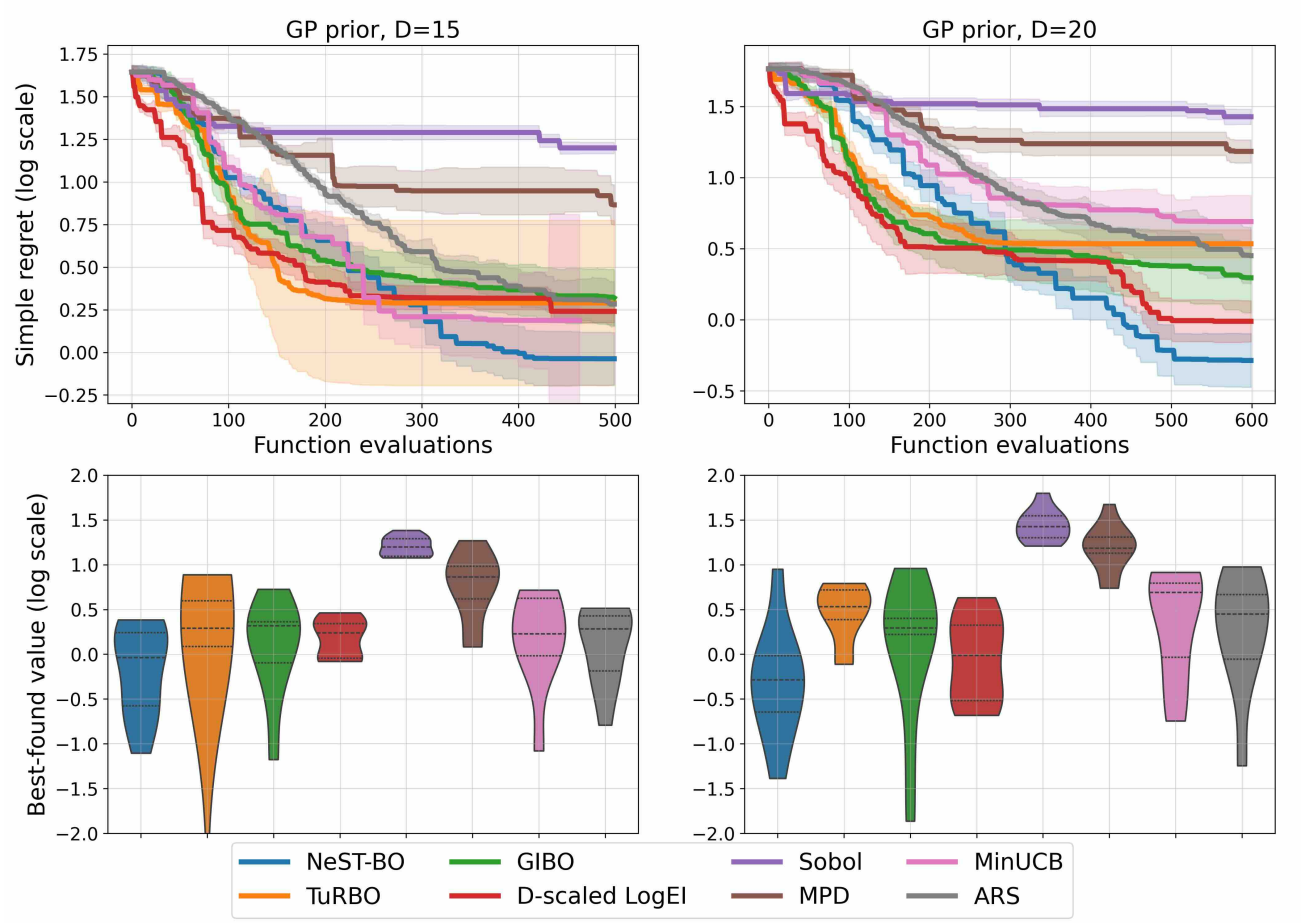

Figure G.1: Optimization of GP-prior draws in $d \in \{15, 20\}$ without subspaces. **Top:** Median simple regret versus number of function evaluations (shaded region corresponds to $\pm$ one standard error). **Bottom:** Distribution of best-found values across 10 realizations. NeST-BO converges faster and to lower regret than strong baselines on these medium-dimensional tasks.

Fourier features (RFF) (Rahimi and Recht, 2007). Concretely, with $n_b = 1024$ features,

$$f(\boldsymbol{x}) \approx \sum_{i=1}^{n_b} w_i \, \phi_i(\boldsymbol{x}), \qquad \phi_i(\boldsymbol{x}) = \sqrt{\tfrac{2}{n_b}} \cos(\boldsymbol{\theta}_i^\top \boldsymbol{x} + \tau_i),$$

where $w_i \sim \mathcal{N}(0, 1)$, $\tau_i \sim \mathcal{U}(0, 2\pi)$, and for the SE kernel we sample $\boldsymbol{\theta}_i \sim \mathcal{N}(\mathbf{0}, \ell(d)^{-2}\boldsymbol{I})$ by Bochner's theorem (Rahimi and Recht, 2007). We treat the resulting f as the ground-truth objective.

To compute simple regret, we approximate the global optimum via L-BFGS (Zhu et al., 1997) with 100 multistarts on each GP prior realization. We consider $d \in \{15, 20\}$, generate 10 independent realizations per d, and report the median across runs. Since the goal here is to stress NeST-BO itself as a high-performance algorithm in the "medium-dimensional" regime (10 to 50 dimensions), no subspace mechanisms are used (and BAxUS is omitted to reduce confounding factors). We also include Augmented Random Search (ARS) (Mania et al., 2018) as an additional baseline. Figure G.1 shows that NeST-BO consistently achieves the lowest simple regret across both dimensions. GIBO, D-scaled LogEI, and TuRBO are competitive early on, but lag in later iterations, suggesting a benefit from explicitly targeting curvature in this regime. We also observe tighter best-value distributions for NeST-BO. Note that the absolute regret depends on the RFF approximation and the length-scale draw; all methods use the same realizations to ensure a fair comparison.

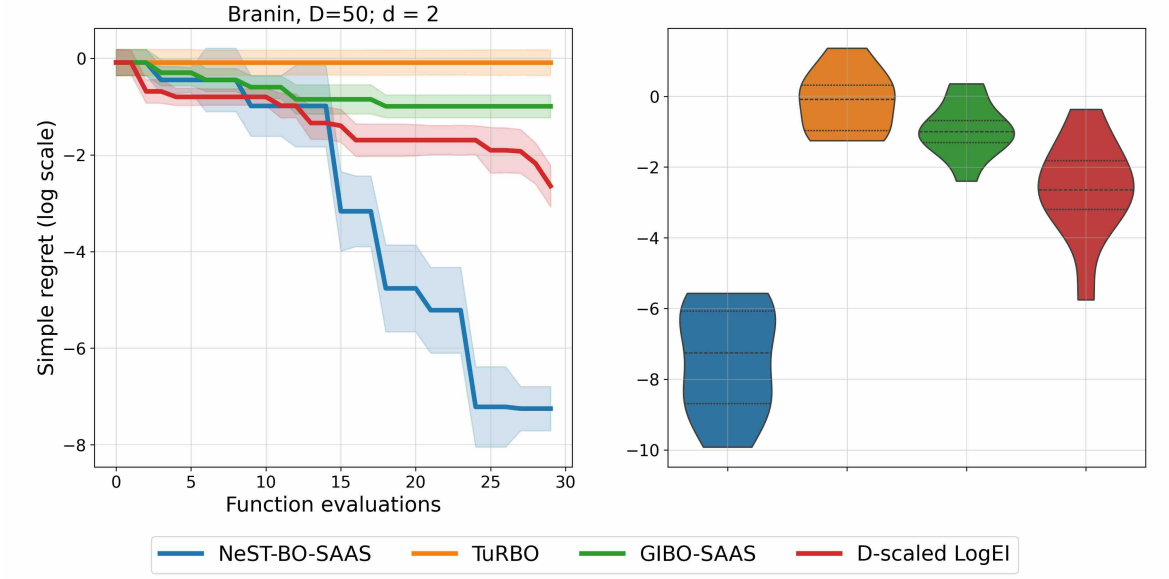

Figure G.2: Branin embedded in $D = 50$ dimensions with a learned SAAS subspace (d=2 active dimensions); shading = $\pm$ one standard error. **Left:** Median simple regret (log scale) over 10 runs. **Right:** Distribution of the best simple regret across runs. NeST-BO-SAAS uses a one-time SAAS-GP to select active coordinates, then runs NeST-BO only in that subspace; GIBO-SAAS uses the identical subspace but uses gradient-only steps.

G.2 Alternative Ways to Learn Embeddings

Our main results show that subspaces can be a powerful vehicle for scaling NeST-BO to high-dimensional spaces (e.g., using BAxUS-style nested subspaces). In this section, we ask a complementary question: *is NeST-BO also compatible with other ways of constructing subspaces?* To answer this, we consider an adaptively learned embedding obtained using the Sparse Axis-Aligned Subspace GP (SAAS-GP) (Eriksson and Jankowiak, 2021). In particular, we fit a SAAS-GP once at the start to select an active set of coordinates – those whose posterior mean length-scales fall below a threshold γ – and then run NeST-BO *only* in this learned subspace while holding the remaining coordinates fixed at their incumbent values. We denote this variant **NeST-BO-SAAS**. To assess the importance of the Newton step itself, we also run **GIBO-SAAS**, which uses the identical SAAS subspace but follows a gradient-based step rule.

We evaluate on a two-dimensional Branin function[3] embedded in 50 dimensions; we use a threshold $\gamma = 10$, initialize with 30 Sobol points, and report results over 10 independent runs. We intentionally exclude BAxUS here to avoid confounding the question of *which* subspace to use with *how* the local step is computed inside that subspace. As shown in Figure G.2, **NeST-BO-SAAS** delivers a sharp reduction in simple regret and substantially outperforms both **GIBO-SAAS** and the non-subspace baselines on this benchmark. This suggests that once a reasonably informative subspace is available, even from a simple axis-aligned selector, the curvature-aware Newton step provides a potentially large advantage over gradient-only updates.

G.3 Impact of Batch Size on NeST-BO

A defining feature of NeST-BO is its explicit use of gradient and Hessian information to construct a local Newton step. This creates a natural trade-off: in each iteration we can either devote more samples to accurately estimating the step, or spend fewer samples per step and move on more quickly. In other words, the batch size b controls how well the Newton direction is learned relative to how frequently it can be updated. Intuitively, very small b risks moving along a poorly estimated (heavily biased or noisy) direction, which can slow convergence or even push the search off course; large b improves the estimation quality but consumes budget so quickly that only a few iterations of actual movement occur.

[3]See `https://www.sfu.ca/~ssurjano/branin.html`

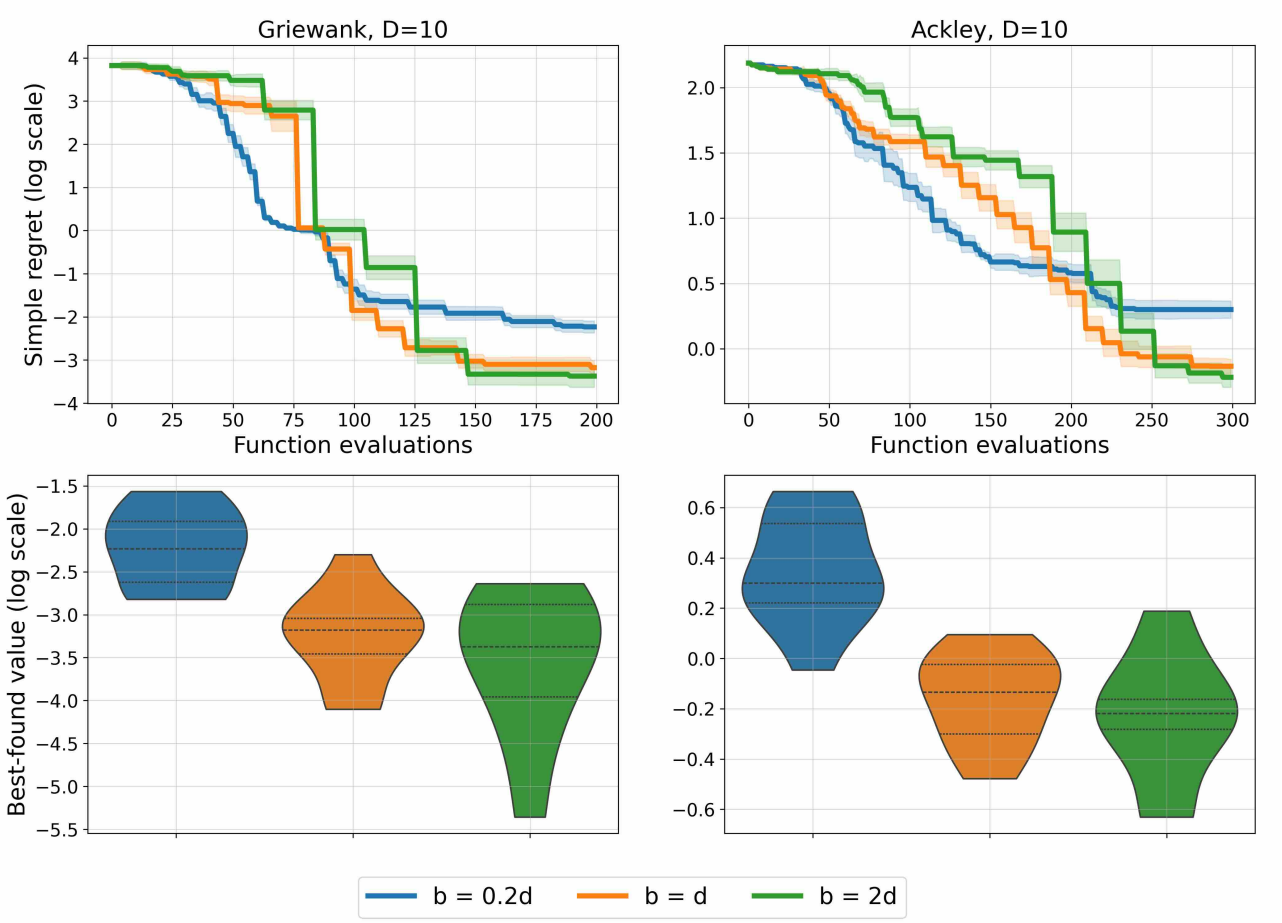

Figure G.3: Effect of sampling budget on Newton-step learning. NeST-BO with $b \in \{0.2d, d, 2d\}$ on Griewank and Ackley ($d = 10$). **Top:** Median simple regret (log scale) across 10 runs, shading = $\pm$ one standard error. **Bottom:** Distribution of best-seen values. Small b underestimates the step and slows convergence; $b = d$ and $b = 2d$ give comparable performance, indicating diminishing returns beyond $b \approx d$.

To examine this tradeoff, we ran NeST-BO on two standard $d = 10$ benchmarks – Griewank[4] and Ackley[5] – using three batch sizes: $b \in \{0.2d, d, 2d\}$. Each run started from 10 Sobol points and was repeated 10 times. This setup lets us isolate how the per-iteration sampling budget influences the quality of the learned Newton step and the resulting optimization trajectory. The results in Figure G.3 reveal a clear pattern. When the batch size is too small ($b = 0.2d$), the algorithm learns an imprecise Newton direction: simple regret decreases slowly and often plateaus at higher values. In contrast, moving from $b = d$ to $b = 2d$ produces only marginal gains in early-iteration slope but nearly identical final performance, indicating diminishing returns once the local power functions for $\boldsymbol{g}$ and $\boldsymbol{H}$ are already reasonably small. In practice, this suggests that, beyond a moderate batch size, further increasing b yields only a slight benefit in direction accuracy while substantially reducing the number of outer iterations.

Overall, these experiments show that NeST-BO benefits from a sufficiently large batch size to accurately estimate the Newton step but does not require very large batches to converge effectively. This supports our default choice of $b = d$ in the main experiments as a balanced setting between step-accuracy and iteration budget.

[4]See `https://www.sfu.ca/~ssurjano/griewank.html`
[5]See `https://www.sfu.ca/~ssurjano/ackley.html`

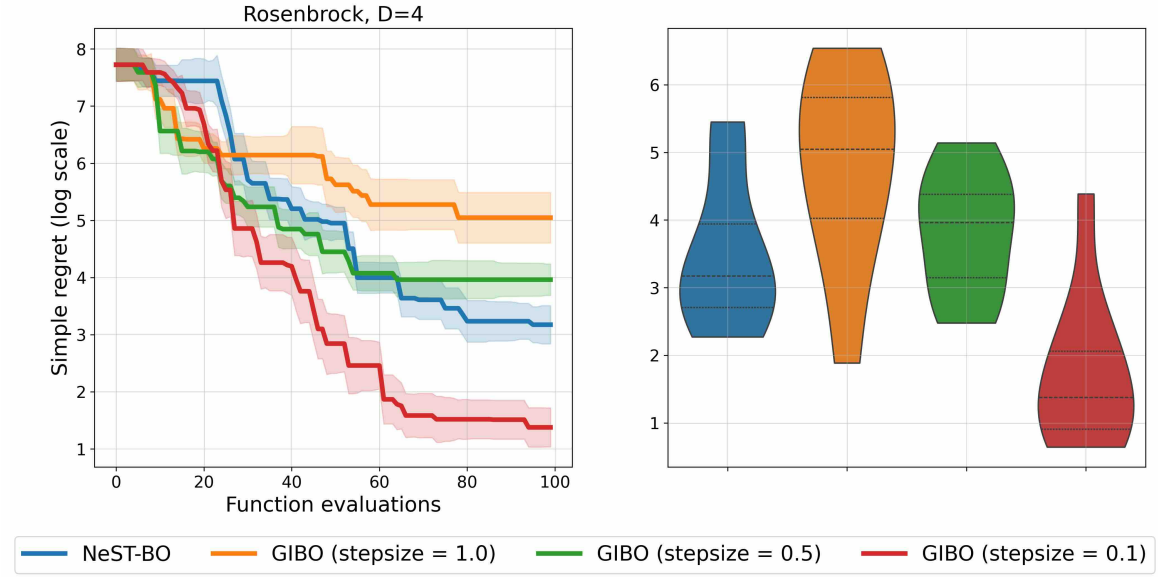

Figure G.4: Step-size sensitivity of GIBO on Rosenbrock ($d = 4$). **Left:** Median simple regret (log scale) across 10 runs; shading = $\pm$ one standard error. **Right:** Distribution of best-seen values. GIBO requires careful step-size tuning ($\eta = 0.1$ works best here); larger η harms stability. NeST-BO's line search removes this sensitivity while leveraging curvature.

G.4 Impact of Step Size on GIBO

First-order local Bayesian optimization methods (such as GIBO) build steps using only gradient information. This raises an important question: *how sensitive is their performance to the choice of step size?* A step that is too aggressive can overshoot narrow valleys or oscillate around the optimum, while a step that is too conservative may crawl slowly toward the solution. In contrast, NeST-BO augments gradient information with curvature and employs an automatic backtracking line search, which potentially makes it less sensitive to such manual tuning.

To examine this issue, we compared NeST-BO with GIBO on the four-dimensional Rosenbrock function[6], a classical ill-conditioned test problem. GIBO used their length-scale-normalized gradient update with fixed step sizes $\eta \in \{1.0, 0.5, 0.1\}$. NeST-BO used its default line search. All methods started from 10 Sobol points, and we averaged results over 10 independent runs to reduce sensitivity to the initial data.

The results in Figure G.4 highlight a striking difference. GIBO's performance is highly step-size dependent: with $\eta = 0.1$, it converges relatively well, but with larger steps ($\eta = 0.5$ or 1.0), the algorithm's performance deteriorates, often stalling or oscillating near Rosenbrock's curved valley. This behavior is consistent with overshooting under ill-conditioned curvature. NeST-BO, by contrast, maintains steady progress without any step-size tuning, leveraging its line search and curvature scaling to automatically adjust the step length. Even on this challenging landscape, NeST-BO achieves competitive regret compared with the best-tuned GIBO setting. Overall, this study underscores the practical advantage of NeST-BO's Newton-based update: by removing the need for manual step-size selection, it improves robustness and reduces the burden of hyperparameter tuning relative to first-order local BO methods like GIBO.

G.5 Runtime Comparison

In this section, we analyze the runtime of NeST-BO compared to GIBO and D-scaled LogEI under controlled conditions. To perform a fair comparison, we run each method on the same shared CPU cluster (see Appendix F.2) limited to 4 cores and evaluate the cumulative time required to complete 200 iterations, which is reported in Table G.1. Each method is initialized with 10 Sobol points, and we report average CPU time across 10 independent replicates on both 12- and 20-dimensional test functions.

[6]See `https://www.sfu.ca/~ssurjano/rosen.html`

Table G.1: Average cumulative CPU time (in seconds) to complete 200 BO iterations across 10 replicates for various algorithms. Standard deviation across replicates is shown in parentheses.

Method	**12-dim function**	**20-dim function**
NeST-BO	167.4 (14.2)	260.1 (16.3)
GIBO	138.2 (12.5)	183.6 (15.7)
D-scaled LogEI	150.7 (10.2)	189.6 (15.7)

Empirically, we find that NeST-BO is modestly more expensive than the other methods, with runtime increases of roughly 15–40% compared to GIBO depending on the dimension. This difference is expected: NeST-BO must construct and invert GP posteriors over gradient and Hessian quantities during each update, and evaluating the posterior variance of the Newton step is the most costly step. In contrast, GIBO uses only first-order information and LogEI computes acquisition values from scalar posteriors.

Despite these additional costs, we argue that the tradeoff is often worthwhile. First, NeST-BO consistently provides lower regret than the alternatives across synthetic and real-world benchmarks, as shown throughout the main paper and Appendix. Second, the marginal CPU time increase is negligible in most practical BO applications, where each black-box evaluation may take minutes to hours (or longer). Finally, the current NeST-BO implementation uses exact kernel derivatives with dense covariance matrices and no real numerical acceleration. We believe this leaves ample room for improvement, especially if recent advances in scaling GPs with derivatives (for certain common kernel classes) are leveraged, e.g., (De Roos et al., 2021)