Adaptive Candidate Point Thompson Sampling for High-Dimensional Bayesian Optimization

Donney Fan
University of British Columbia
Vector Institute
donneyf@cs.ubc.ca

Geoff Pleiss
University of British Columbia
Vector Institute
geoff.pleiss@stat.ubc.ca

Abstract

In Bayesian optimization, Thompson sampling selects the evaluation point by sampling from the posterior distribution over the objective function maximizer. Because this sampling problem is intractable for Gaussian process (GP) surrogates, the posterior distribution is typically restricted to fixed discretizations (i.e., candidate points) that become exponentially sparse as dimensionality increases. While previous works aim to increase candidate point density through scalable GP approximations, our orthogonal approach increases density by *adaptively* reducing the search space during sampling. Specifically, we introduce Adaptive Candidate Thompson Sampling (ACTS), which generates candidate points in subspaces guided by the gradient of a surrogate model sample. ACTS is a simple drop-in replacement for existing TS methods—including those that use trust regions or other local approximations—producing better samples of maxima and improved optimization across synthetic and real-world benchmarks.

1 INTRODUCTION

Bayesian optimization (BO) is a popular framework for sample-efficient optimization, with prominent applications in machine learning (Snoek et al., 2012; Swersky et al., 2014), scientific discovery (Maus et al., 2022; Chitturi et al., 2024; Slattery et al., 2024), and robotics (Berkenkamp et al., 2023; Deisenroth and Rasmussen, 2011; Deneault et al., 2021), where a probabilistic surrogate model of a black-box function is refined iteratively through adaptively chosen function evaluations. Historically, BO has been confined to low-dimensional problems (typically ≤ 10 dimensions), but recent advances have scaled this paradigm to problems with hundreds or thousands of dimensions (Eriksson et al., 2019; Papenmeier et al., 2022; Hvarfner et al., 2024).

Many of these new methods—including TuRBO (Eriksson et al., 2019), BAxUS (Papenmeier et al., 2022), and their variants (e.g. Eriksson and Poloczek, 2021; Daulton et al., 2022; Maus et al., 2022; Rashidi et al., 2024)—rely on Thompson sampling (TS) (Thompson, 1933) to determine which data points to acquire. Unique among other acquisition functions, Thompson sampling uses randomness to balance exploration and exploitation by maximizing a *sample* of the posterior surrogate model rather than maximizing some deterministic function of its marginal moments. This randomized strategy affords strong theoretical convergence guarantees (Russo and Van Roy, 2014; Kandasamy et al., 2018).

Applying TS with a Gaussian process (GP) surrogate is hampered by the curse of dimensionality. As sampling a continuous path is intractable, the GP is often sampled on a discretized *candidate set* of points, whose density is crucial for performance (Pleiss et al., 2020). However, the number of points needed to adequately fill a space grows exponentially with its dimension, a scaling that quickly overwhelms the computational budget of GP sampling (typically $\approx 10,000$ points (Garnett, 2023)). Although scalable approximations can increase its efficacy (e.g. Pleiss et al., 2018), they do not overcome this exponential dependence.

While existing Thompson sampling methods use sparsity (Eriksson et al., 2019; Regis and Shoemaker, 2013) or locality (Rashidi et al., 2024) to mitigate the curse of dimensionality, this paper introduces a novel and complementary strategy. Our approach, Adaptive Candidate Thompson Sampling (ACTS), *adaptively* con-

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

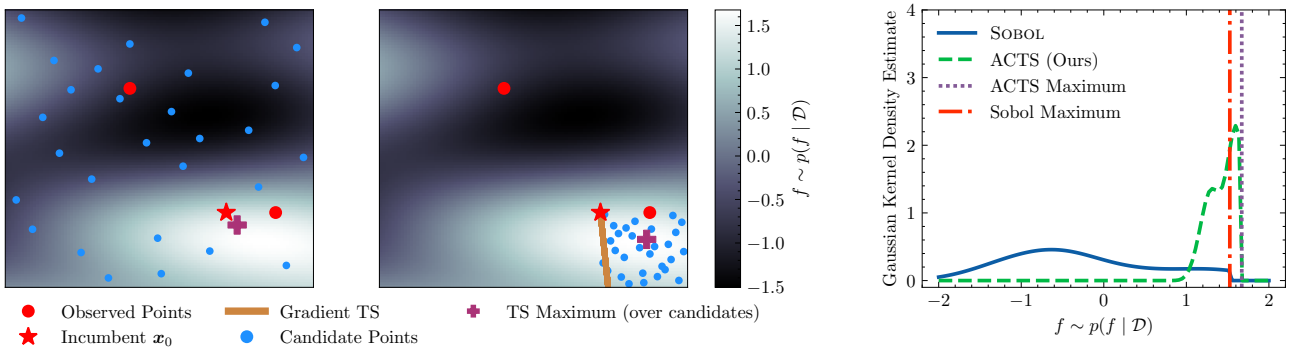

Figure 1: **Illustration of Adaptive Candidate Thompson Sampling.** A GP posterior sample (background colour) can only be evaluated at a finite set of candidate points. (Left.) Standard candidate point methods produce a poor discretization of the input domain. (Center.) ACTS only produces candidate points in a subspace that is aligned with the gradient of the posterior sample, increasing density in a region where a (local) function sample maximum is likely to be found. (Right.) Thus, ACTS candidate points yield values closer to the true optimum of the GP posterior sample.

structs candidate sets in regions where the GP sample path is likely to attain its maximum. The key insight is that candidate sets need not be independent of the sample path. ACTS first samples the GP's gradient at the incumbent to identify an ascent direction, then populates a candidate set within a small, gradient-aligned region that is significantly smaller than the input space, enabling a far denser discretization of candidate points than naïve approaches. This initial sample is extended to a joint sample over the new candidates, yielding a valid GP realization evaluated at promising points. Despite using local gradient information to define the candidate set, we empirically show that ACTS is no more local than other TS variants and theoretically show global consistency (i.e., it eventually queries arbitrarily close to the global maximizer).

We evaluate ACTS across several high-dimensional benchmarks. By coupling candidate point selection with posterior sampling, ACTS finds higher values of posterior sample paths than other TS methods. When paired with local or sparse BO strategies, ACTS can offer significant improvements to optimization efficacy, matching or exceeding alternative methods.

2 BACKGROUND

We seek to maximize an expensive black-box function $F(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ over a compact domain $\mathcal{X} \subset \mathbb{R}^d$ (often $[0,1]^d$). F lacks a known analytical form (e.g., no gradient information) and is accessed only through noisy point-wise evaluations of the form $y = F(\boldsymbol{x}) + \sigma_n \epsilon$, where $\epsilon \sim \mathcal{N}(0,1)$ and σ_n is inferred from data.

Bayesian Optimization maximizes F by building a probabilistic surrogate model f, most commonly a Gaussian process (GP), which is updated as new observations are gathered. The GP prior is $p(f) = \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$, where we assume a zero mean function and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive definite kernel function whose hyperparameters are typically estimated with type-II maximum likelihood (Rasmussen and Williams, 2006, Ch. 5). At each iteration t, given n_t total observation pairs $\mathcal{D}_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_t}$ (i.e., no gradient observations), the posterior over f is also a GP, $p(f \mid \mathcal{D}_t) = \mathcal{GP}(\mu_t(\boldsymbol{x}), k_t(\boldsymbol{x}, \boldsymbol{x}'))$, with mean and covariance

$$\mu_t(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{X}_t)\widetilde{\boldsymbol{K}}^{-1}\boldsymbol{y}, \tag{1}$$

$$k_t(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{X}_t)\widetilde{\boldsymbol{K}}^{-1}k(\boldsymbol{X}_t, \boldsymbol{x}'). \tag{2}$$

Here, $\boldsymbol{X}_t \in \mathbb{R}^{n_t \times d}$ is the row-wise concatenation of $\{\boldsymbol{x}_i\}_{i=1}^{n_t}$, $k(\boldsymbol{x}, \boldsymbol{X}_t) = [k(\boldsymbol{x}, \boldsymbol{x}_i)]_{i=1}^{n_t} \in \mathbb{R}^{n_t}$, $\widetilde{\boldsymbol{K}} = [k(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{ij} + \sigma^2 \boldsymbol{I} \in \mathbb{R}^{n_t \times n_t}$ is the Gram matrix plus observational noise, and $\boldsymbol{y}_t = [y_i]_{i=1}^{n_t}$. Next, an *acquisition function* $\alpha_{f|\mathcal{D}_t}$ is maximized to select the next point: $\boldsymbol{x}_{t+1} = \arg\max_{\boldsymbol{x} \in \mathcal{X}} \alpha_{f|\mathcal{D}_t}(\boldsymbol{x})$. All acquisition functions balance exploration—searching over regions where the model is uncertain—with exploitation—searching near the best-known point, or *incumbent* $\boldsymbol{x}_0$:

$$\boldsymbol{x}_0 := \arg\max_{\boldsymbol{x} \in \mathcal{D}_t} \mathbb{E}[f(\boldsymbol{x}) \mid \mathcal{D}_t].$$

Common choices of $\alpha_{f|\mathcal{D}_t}$ include Expected Improvement (EI) (Močkus, 1975; Jones et al., 1998), the Upper Confidence Bound (UCB) (Srinivas et al., 2010), and Thompson Sampling (Thompson, 1933); the latter of which is the focus of this work.

Thompson Sampling is a randomized acquisition function, popular in high-dimensional BO (e.g. Daulton et al., 2022; Eriksson et al., 2019; Papenmeier et al., 2022), that maximizes a single function f drawn from

the posterior, $f \sim p(f \mid \mathcal{D}_t)$. The next point is thus a sample of the posterior maximizer's location: $\boldsymbol{x}_{t+1} \sim p(\arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) \mid \mathcal{D}_t)$, corresponding to the policy $\alpha_{f|\mathcal{D}_t}(\boldsymbol{x}) = f(\boldsymbol{x})$. This approach provides a probabilistic balance between exploration and exploitation. In regions of high posterior variance, function samples f exhibit significant variability, driving the search towards unexplored areas. At the same time, these samples will nearly interpolate the incumbent $\boldsymbol{x}_0$, and so it is also likely that search is concentrated around a known good point. TS is supported by theoretical guarantees of convergence to the global optimum under mild conditions (Kandasamy et al., 2018; Russo and Roy, 2016).

Practical Implementations of TS with Candidate Points. Since sampling the infinite-dimensional function $f \sim p(f \mid \mathcal{D}_t)$ is intractable, practical implementations sample over a smaller finite set of M candidate points $\tilde{\mathcal{X}} = \{\boldsymbol{x}_1^*, \dots, \boldsymbol{x}_M^*\}$ via reparameterization:

$$\boldsymbol{f}_{\tilde{\mathcal{X}}} =: \big[f(\boldsymbol{x}_1^*) \dots f(\boldsymbol{x}_M^*)\big] = \mu_t(\tilde{\mathcal{X}}) + k_t(\tilde{\mathcal{X}}, \tilde{\mathcal{X}})^{\frac{1}{2}} \boldsymbol{z}, \quad (3)$$

where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, $\mu_t(\tilde{\mathcal{X}}) \in \mathbb{R}^M$ and $k_t(\tilde{\mathcal{X}}, \tilde{\mathcal{X}}) \in \mathbb{R}^{M\times M}$ are the posterior mean and covariance (Eqs. 1, 2) evaluated at the candidate points.[1] Maximization is then restricted to this discretized domain:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}^*_{i_{\max}}, \qquad i_{\max} := \arg\max_{i=1,\dots,M} [\boldsymbol{f}_{\tilde{\mathcal{X}}}]_i.$$

Computing $k_t(\tilde{\mathcal{X}}, \tilde{\mathcal{X}})^{\frac{1}{2}}$ in Eq. 3 naïvely scales with $O(M^3)$. This cubic scaling limits M to 10^4 or less in practice, even when using scalable sampling approximations (e.g. Pleiss et al., 2018, 2020; Rahimi and Recht, 2007; Renganathan and Carlson, 2025).

Policies for Generating Candidate Points. Strategies for generating candidate points $\tilde{\mathcal{X}} \subset \mathcal{X}$ can be formally described as random (or quasi-random) policies π_0 that place candidates within a subregion $C \subseteq \mathcal{X}$ of the domain:

$$\tilde{\mathcal{X}} \sim \pi_0(\tilde{\mathcal{X}}; C), \qquad C \subset \mathcal{X}, \quad (4)$$

Simple strategies include space-filling methods like *Sobol sequences* or uniform sampling. Policies such as *Cylindrical Thompson Sampling (CTS)* (Rashidi et al., 2024) sample from a spherical distribution around $\boldsymbol{x}_0$, aiming to improve local exploration. The most prominent policy for high-dimensional BO is *Random Axis-Aligned Subspace Perturbations (RAASP)* (Daulton et al., 2022; Eriksson et al., 2019; Papenmeier et al., 2022, 2025), which generates candidates by perturbing $\boldsymbol{x}_0$ in only a few coordinates. This policy can be implemented by sampling a perturbation vector $\boldsymbol{u}_i$ which is multiplied by a sparse binary mask $\boldsymbol{b}_i$ to limit the number of perturbed dimensions:

$$\begin{aligned} \tilde{\boldsymbol{x}}_i &= \boldsymbol{x}_0 + \boldsymbol{u}_i \odot \boldsymbol{b}_i, \\ (\boldsymbol{x}_0 + \boldsymbol{u}_i) &\sim \text{Uniform}(\mathcal{X}), \\ [\boldsymbol{b}_i]_j &\overset{\text{iid}}{\sim} \text{Bernoulli}(20/d), \end{aligned} \quad (5)$$

where $\odot$ denotes element-wise multiplication (Hadamard product). Each candidate $\tilde{\boldsymbol{x}}_i$ is formed by applying a sparse binary mask $\boldsymbol{b}_i$ to a perturbation vector $\boldsymbol{u}_i$, which is typically derived from a Sobol sequence. This effectively restricts perturbations of $\boldsymbol{x}_0$ to a low-dimensional subspace.

While discretization makes TS feasible, it is hindered by the curse of dimensionality, as the number of points needed for adequate coverage grows exponentially with d. For instance, Appendix E shows that even 10^6 Sobol points can fail to sample near the optimum in simple settings. The sparsity imposed by RAASP alleviates this issue to some degree, but—as our experiments will show—RAASP can be improved upon through denser sampling afforded by our method.

Jacobian GP Model. As GPs are closed under linear operations (e.g. Rasmussen and Williams, 2006) they induce a joint Gaussian distribution over function and derivative values with any once-differentiable kernel (a condition that is satisfied by the popular RBF and Matérn kernels). This allows us to obtain a joint prior over the observations $\boldsymbol{y}_t = [y_1, \dots, y_{n_t}]$, function values at the candidate points $f_{\tilde{\mathcal{X}}} = [f(\boldsymbol{x}_1^*), \dots, f(\boldsymbol{x}_M^*)]$, and the gradient $\nabla f(\boldsymbol{x}_0)$:

$$\begin{bmatrix} \boldsymbol{y}_t \\ \boldsymbol{f}_{\tilde{\mathcal{X}}} \\ \nabla f(\boldsymbol{x}_0) \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} k(\boldsymbol{X},\boldsymbol{X}) + \sigma_n^2 \boldsymbol{I} & k(\boldsymbol{X}, \tilde{\mathcal{X}}) & k(\boldsymbol{X}, \boldsymbol{x}_0)\nabla^\top \\ k(\tilde{\mathcal{X}}, \boldsymbol{X}) & k(\tilde{\mathcal{X}}, \tilde{\mathcal{X}}) & k(\tilde{\mathcal{X}}, \boldsymbol{x}_0)\nabla^\top \\ \nabla k(\boldsymbol{x}_0, \boldsymbol{X}) & \nabla k(\boldsymbol{x}_0, \tilde{\mathcal{X}}) & \nabla k(\boldsymbol{x}_0, \boldsymbol{x}_0)\nabla^\top \end{bmatrix}\right), \quad (6)$$

where ∇ and $\nabla^\top$ are the gradient operators with respect to the first and second arguments of k, respectively. Applying standard Gaussian conditioning rules to Eq. 6 yields the joint posterior $p(\boldsymbol{f}_{\tilde{\mathcal{X}}}, \nabla f(\boldsymbol{x}_0) \,|\, \mathcal{D}_t)$. This model has been used in BO with observed gradients (Wu et al., 2017; Shekhar and Javidi, 2021) or for local exploration (Müller et al., 2021; Nguyen et al., 2022; Wu et al., 2023). To the best of our knowledge, our work is the first to leverage it within the context of TS and candidate point placement.

Other Thompson Sampling Approaches. Several alternative TS algorithms avoid constructing explicit candidate-point sets. Wilson et al. (2020, 2021) propose a *pathwise* sampling strategy, where a GP is approximated by a finite-basis Bayesian linear regression model. Sampling the model's finite-dimensional coefficients yields a differentiable function realization from

[1] $k_t(\tilde{\mathcal{X}}, \tilde{\mathcal{X}})^{\frac{1}{2}}$ denotes any $\boldsymbol{L}$ such that $\boldsymbol{L}\boldsymbol{L}^\top = k_t(\tilde{\mathcal{X}}, \tilde{\mathcal{X}})$. The most common choice is the Cholesky factor.

the approximate posterior, which can be optimized using standard gradient-based methods. However, this approach introduces an inexact GP approximation and its own curse of dimensionality, as the number of required basis functions can scale poorly with d. Another strategy uses Markov-Chain Monte-Carlo (MCMC) to sample directly from the distribution of the maximizer, $p(\arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) \mid \mathcal{D}_t)$ (Bijl et al., 2017; Sweet, 2024; Yi et al., 2024). The efficiency of this direct approach is contingent on the MCMC algorithm, as many samplers have convergence rates that scale poorly in high-dimensional spaces.

3 ADAPTIVE CANDIDATE THOMPSON SAMPLING

We introduce Adaptive Candidate Thompson Sampling, a method that mitigates the curse of dimensionality in candidate-based Thompson sampling. ACTS samples the gradient of the posterior $f \sim p(f \mid \mathcal{D}_t)$ to concentrate the candidate set into a small region likely to contain a local maximum of f. The full procedure is detailed in Algorithm 1.

Intuition. A GP posterior sample over a candidate set, $\boldsymbol{f}_{\tilde{\mathcal{X}}} \sim p(\boldsymbol{f}_{\tilde{\mathcal{X}}} \mid \mathcal{D}_t)$, can be drawn as the marginal of a joint sample that also includes an auxiliary variable, such as the gradient at the incumbent, $\nabla f(\boldsymbol{x}_0)$:

$$\begin{bmatrix}\boldsymbol{f}_{\tilde{\mathcal{X}}}^\top & \nabla f(\boldsymbol{x}_0)\end{bmatrix} \sim p\left(\begin{bmatrix}\boldsymbol{f}_{\tilde{\mathcal{X}}}^\top & \nabla f(\boldsymbol{x}_0)\end{bmatrix} \mid \mathcal{D}_t\right).$$

Since the joint posterior is Gaussian (Eq. 6), we can factorize the distribution and sample sequentially using standard conditioning rules:

$$\begin{aligned}\boldsymbol{f}_{\tilde{\mathcal{X}}} &\sim p(\boldsymbol{f}_{\tilde{\mathcal{X}}} \mid \nabla f(\boldsymbol{x}_0), \mathcal{D}_t),\\ \nabla f(\boldsymbol{x}_0) &\sim p(\nabla f(\boldsymbol{x}_0) \mid \mathcal{D}_t).\end{aligned} \tag{7}$$

The key insight of our method is that the candidate set $\tilde{\mathcal{X}}$ need not be fixed beforehand, but can be *adaptively* constructed based on the sampled gradient:

$$\begin{aligned}\boldsymbol{f}_{\tilde{\mathcal{X}}_t} &\sim p(\boldsymbol{f}_{\tilde{\mathcal{X}}_t} \mid \tilde{\mathcal{X}}_t, \mathcal{D}_t),\\ \tilde{\mathcal{X}}_t &\sim \pi(\tilde{\mathcal{X}} \mid \nabla f(\boldsymbol{x}_0)),\\ \nabla f(\boldsymbol{x}_0) &\sim p(\nabla f(\boldsymbol{x}_0) \mid \mathcal{D}_t).\end{aligned} \tag{8}$$

Here, π is a policy that uses the sampled gradient $\nabla f(\boldsymbol{x}_0)$, a direction of ascent for the function sample f. Consequently, π can densely place candidate points in this promising region, increasing the likelihood of a large Thompson sample $\max_{\boldsymbol{x}\in\tilde{\mathcal{X}}} f(\boldsymbol{x})$. We emphasize that the resulting $\boldsymbol{f}_{\tilde{\mathcal{X}}_t}$ in Eq. 8 is "exact"—i.e., it is a valid realization of $p(f \mid \mathcal{D}_t)$ on some finite subset—even though $\tilde{\mathcal{X}}$ is adaptively chosen.

Algorithm 1 Adaptive Candidate Thompson Sampling (ACTS)

Hyperparameters: Number of candidate points M.

$\boldsymbol{x}_0 \leftarrow \boldsymbol{x}_j,\ j = \arg\max_i y_i$ ▷ Obtain incumbent

$\nabla f(\boldsymbol{x}_0) \sim \nabla f(\boldsymbol{x}_0) \mid \mathcal{D}_t$ ▷ Sample via Eq. 6.

$[\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_M^*] =: \tilde{\mathcal{X}}_t \sim \pi(\tilde{\mathcal{X}} \mid \nabla f(\boldsymbol{x}_0))$ ▷ Generate candidate points

$\boldsymbol{f}_{\tilde{\mathcal{X}}_t} \sim p(\boldsymbol{f}_{\tilde{\mathcal{X}}_t} \mid \nabla f(x_0), \mathcal{D}_t)$ ▷ Sample via Eq. 6

$\boldsymbol{x}_{t+1} = \boldsymbol{x}^*_{i_{\max}}$, where $i_{\max} = \arg\max_i [\boldsymbol{f}_{\tilde{\mathcal{X}}_t}]_i$

ACTS Search Spaces. Our adaptive candidate policy, $\pi(\tilde{\mathcal{X}} \mid \nabla f(\boldsymbol{x}_0))$, considers a general recipe that applies a base non-adaptive policy π_0 (see Eq. 4) to a smaller search space $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)} \subset \mathcal{X}$ that is aligned with the sampled gradient:

$$\pi(\tilde{\mathcal{X}} \mid \nabla f(\boldsymbol{x}_0)) := \pi_0(\tilde{\mathcal{X}}; \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}). \tag{9}$$

Specifically, ACTS defines $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$ as an axis-aligned cone rooted at the incumbent $\boldsymbol{x}_0$ and extending along the coordinate-wise direction of the gradient:

$$\mathcal{T}_{\nabla f(\boldsymbol{x}_0)} = \left\{\boldsymbol{x}_0 + \boldsymbol{v} \odot \nabla f(\boldsymbol{x}_0) \mid \boldsymbol{0} \preceq \boldsymbol{v} \in \mathbb{R}^d\right\} \cap \mathcal{X}, \tag{10}$$

where $\odot$ denotes element-wise product. This construction, illustrated in Fig. 1, defines a d-dimensional search space (rectangular when $\mathcal{X}$ is rectangular) that is significantly smaller than the original domain. For instance, if $\boldsymbol{x}_0$ is centered in a $d = 100$ dimensional space, $\mathrm{vol}\left(\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}\right)$ is reduced by a factor of $2^{100} \approx 10^{30}$, as we remove a half-space for each dimension. Importantly, by aligning with the gradient, $\tilde{\mathcal{X}}$ is guaranteed to contain points with higher function values than the incumbent $\boldsymbol{x}_0$, as any $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$ with non-zero hypervolume contains some $\boldsymbol{x}'$ with $f(\boldsymbol{x}') > f(\boldsymbol{x}_0)$. While the global $\arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$ may not lie in $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$, it is likely that $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$ contains some local maximum of f. Thus, limiting the search to $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$ increases the likelihood that some $\boldsymbol{x}^* \in \tilde{\mathcal{X}}$ is close to this maximizer.

Although this strategy confines the candidate set to a region based on local gradient information, it avoids getting stuck in local optima because this gradient is a random draw. We prove in Appendix D that ACTS maintains the global consistency of standard TS.

Theorem 1 (Informal)**.** *Under mild assumptions, running Bayesian optimization with ACTS is guaranteed to eventually query a point arbitrarily close to a global maximizer.*

The proof of Theorem 1 shows that since we do not observe gradients, the posterior covariance never fully collapses, and thus a sampled gradient is non-zero almost surely, thus the ACTS search space remains well-defined, even when $\nabla f(\boldsymbol{x}_0) \approx \boldsymbol{0}$.

Alternative search spaces. The axis-aligned cone is one of several possible geometries for $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$. We explore alternatives in Appendix H, including a one-dimensional line search along the gradient. We find that the construction in Eq. 10 provides a robust balance between reducing the search space volume and yielding effective acquisitions.

ACTS with RAASP. Having defined the search space $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$, ACTS can enhance any base policy π_0, such as Sobol sampling (e.g. Balandat et al., 2020) or CTS (Rashidi et al., 2024), by applying it within this restricted region (Eq. 9). Here, we focus on integrating ACTS with RAASP, which is arguably among the most widely used policies in high-dimensional BO (Eriksson et al., 2019; Papenmeier et al., 2022).

While the standard RAASP policy (Eq. 5) can be directly plugged into Eq. 9, we provide a simple modification that additionally leverages information from $\nabla f(\boldsymbol{x}_0)$. Recall that RAASP produces candidates by randomly perturbing the incumbent $\boldsymbol{x}_0$ in ≈ 20 randomly chosen dimensions. With the gradient information from ACTS, we can favour perturbations in dimensions aligned with $\nabla f(\boldsymbol{x}_0)$ (i.e., the direction of steepest ascent). Mathematically, we modify the masking vector $\boldsymbol{b}_i \in \{0,1\}^d$ in Eq. 5—which selects the dimensions to perturb—from i.i.d. Bernoulli entries to non-i.i.d. entries:

$$[\boldsymbol{b}_i]_j \overset{\text{iid}}{\sim} \text{Bernoulli}\left(\min\left\{20\frac{[\nabla f(\boldsymbol{x}_0)]_j^2}{||\nabla f(\boldsymbol{x}_0)||_2^2}, 1\right\}\right). \quad (11)$$

While RAASP may produce perturbations in dimensions with minimal gradient (i.e., flatter directions with low likelihood of a posterior sample maximizer), our formulation ensures that dimensions with greater contributions to the gradient vector are more likely to be perturbed.

Compatibility with Local BO Methods. ACTS is readily compatible with local BO methods that employ trust regions, such as TuRBO (Eriksson et al., 2019). Letting $\mathcal{R}_t$ denote the trust region at time t, ACTS simply replaces the non-adaptive candidate policy $\pi_0(\tilde{\mathcal{X}}; \mathcal{R}_t)$ with $\pi_0(\tilde{\mathcal{X}}; \mathcal{R}_t \cap \mathcal{T}_{\nabla f(\boldsymbol{x}_0)})$. The resulting search space is smaller than the original trust region $\mathcal{R}_t$. While both methods restrict the search space, they are different and complementary. Trust region strategies enhance regression accuracy within a confined area to rapidly converge to a local optimum. In contrast, ACTS improves the fidelity of the Thompson sampling acquisition by more densely discretizing the posterior sample f in a region likely to contain its maximum. Again, the global consistency guarantee (Theorem 1) ensures that this focus on a subregion of $\mathcal{X}$ does not yield convergence to a local optimum. This makes ACTS a purposeful search policy that is orthogonal to (local) trust region approaches.

Compatibility with Batch BO. Thompson sampling naturally extends to batch BO, where q points are acquired by maximizing q independent posterior samples, $f^{(1)}, \ldots, f^{(q)} \sim p(f \mid \mathcal{D}_t)$. ACTS integrates seamlessly into this framework. For each sample $f^{(i)}$, we independently perform the ACTS procedure: draw a gradient $\nabla f^{(i)}(\boldsymbol{x}_0)$, construct a candidate set $\tilde{\mathcal{X}}_t^{(i)}$ within the corresponding cone $\mathcal{T}_{\nabla f^{(i)}(\boldsymbol{x}_0)}$, and find the maximizer of $f^{(i)}$ over this unique set. This process naturally promotes diversity. Since the initial gradient samples are random, the adaptive cones and resulting candidate sets typically differ, directing the parallel searches to distinct promising regions.

Computational Complexity. While ACTS requires additional computation over standard TS methods, this overhead is negligible under a fixed candidate point budget. All methods take $O(M^3)$ time to sample the GP posterior over $\tilde{\mathcal{X}}$. Given $|\mathcal{D}_t| = n_t$ observations, ACTS uses an additional $O(n_t^2 d)$ computation to sample from $p(\nabla f(\boldsymbol{x}_0) \mid \mathcal{D}_t)$ (i.e., via standard conditioning of Eq. 6) and $O(Md^2)$ to condition $p(\boldsymbol{f}_{\tilde{\mathcal{X}}_t} \mid \mathcal{D}_t)$ on $\nabla f(\boldsymbol{x}_0)$ to sample $\boldsymbol{f}_{\tilde{\mathcal{X}}_t}$. Assuming $d \ll M$ and $n_t \ll M$, as is often the case, ACTS retains the $O(M^3)$ asymptotic complexity of standard TS methods. It also permits scalable sampling extensions as mentioned in Section 2.

4 EXPERIMENTAL RESULTS

Baselines. We compare ACTS against three leading TS baselines: RAASP (Eriksson et al., 2019) and Cylindrical TS (Rashidi et al., 2024) candidate policies, and the candidate-free Pathwise TS (Wilson et al., 2020). Each method is evaluated both in a global space and with TuRBO trust regions. We omit the naive Sobol policy from main comparisons due to its poor performance (Appendix E), though we retain it for ablation studies. On large-scale benchmarks, we add comparisons to three state-of-the-art methods: the sparse, fully-Bayesian SAASBO (Eriksson and Jankowiak, 2021); the progressive latent-space method BAxUS (Papenmeier et al., 2022); and the improved LogEI (Ament et al., 2023).

Implementation Details. We use GP surrogates with a constant mean and a squared exponential kernel, adopting the dimensional-scaled priors from Hvarfner et al. (2024). All candidate-based TS methods use $M = 10^4$ points. For the GuacaMol benchmarks (Brown et al., 2019), which involve thousands of evaluations, we use conjugate gradients (Gardner et al.,

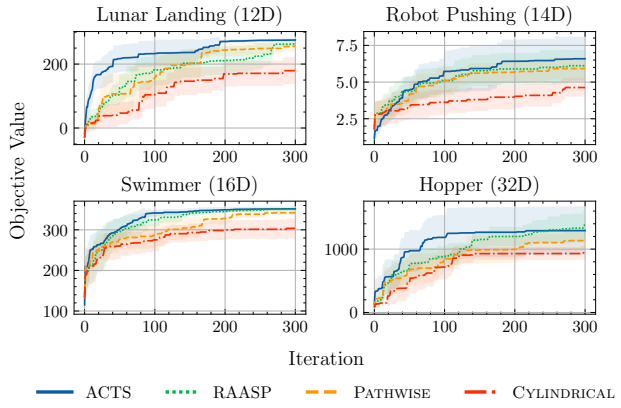

Figure 2: **Optimization performance of medium-dimensional real-world optimization problems.** ACTS consistently exhibits top performance (within two standard errors) and achieves high objective values earlier in optimization than other methods.

2018) to efficiently compute the marginal log-likelihood. Each run is warm-started with 30 initial points from a Sobol sequence. Results are averaged over 10 runs with plots showing mean performance shaded by ± 2 standard errors. Further details are in Appendix A and an implementation of ACTS is available at `https://github.com/DonneyF/ACTS`.

4.1 Optimization Performance

Medium-Dimensional Problems. We first evaluate ACTS on medium-dimensional ($d \leq 32$) benchmarks from robotics and control: Lunar Lander (12D), Robot Pushing (14D) (Wang et al., 2018), Swimmer (16D), and Hopper (32D) (Todorov et al., 2012). As shown in Fig. 2, after 300 iterations of optimization, ACTS achieves the highest reward on most tasks, frequently outperforming both RAASP and CTS.

High-Dimensional Optimization Performance. We evaluate our method on several high-dimensional problems ($d \leq 1000$) common in the literature. These include Rover (60D) (Wang et al., 2018), MOPTA08 (124D) (Jones, 2008), an SVM hyperparameter tuning task (388D) (Papenmeier et al., 2022), and tasks from the LassoBench suite (180–1000D) (Šehić et al., 2022). Finally, we consider challenging molecule design tasks from the GuacaMol benchmarks (Brown et al., 2019), optimizing within a 256D continuous latent space defined by a pretrained SELFIES-VAE (Maus et al., 2022).

Figure 3 compares the four TS approaches, both in a global setting and within TuRBO trust regions (Eriksson et al., 2019). While no single baseline is consistently best, ACTS is a top performer across nearly all benchmarks, with or without TuRBO. (In Appendix B, we perform a ranking analysis to confirm that ACTS outperforms all other methods in aggregate across benchmarks.) Pathwise and Cylindrical TS have varying levels of efficacy and TuRBO is sometimes less efficient than using the entire space. While RAASP offers competitive performance on some tasks, ACTS demonstrates a clear advantage on MOPTA08, SVM, and Median Molecules 1.

We further compare ACTS against SAASBO, BAxUS, and LogEI.[2] For brevity, Table 1 reports the final objective value achieved by each method (see Appendix B for full optimization plots). ACTS consistently matches or exceeds the performance of these state-of-the-art methods, demonstrating robust performance where other baselines are less consistent.

Batch Optimization. We evaluate ACTS in a parallel optimization setting, a common application for Thompson sampling. Figure 4 compares the four TS methods (without TuRBO) on a subset of benchmarks using a batch size of $q = 100$ for 200 iterations. We observe that ACTS maintains its strong performance ranking relative to the other TS methods, demonstrating its effectiveness extends from the sequential to the batch setting. Additional results for smaller batch sizes ($q = 10, 50$) are in Appendix C.

4.2 Analysis and Ablations

Analysis of Search Space Volume. We hypothesize that ACTS improves performance by increasing candidate point density within a drastically smaller search space. We verify this in Fig. 5, which plots the search space volume induced by ACTS on the 60D Rover problem over $\mathcal{X} = [0, 1]^d$. The volume is orders of magnitude smaller than the full domain, confirming a massive increase in candidate density under a fixed budget. In this case, we have that $0.5^{60} \approx 10^{-18}$; thus ACTS reduces on average each dimension by more than half. When combined with TuRBO, the intersection of the two regions shrinks the volume even further (to as low as 10^{-200}), confirming that the methods provide complementary benefits.

Analysis of Candidate Points Function Values. We hypothesize that the ACTS discretization will better cover regions of high function values, and will thus produce candidates closer to the true posterior sample maximum. To this end, we compare the $f_{\max}$ distributions produced by ACTS versus other candidate point methods on the 60-dimensional Rover task. Specifically,

[2]We omit SAASBO on the GuacaMol benchmarks due to its prohibitive runtime.

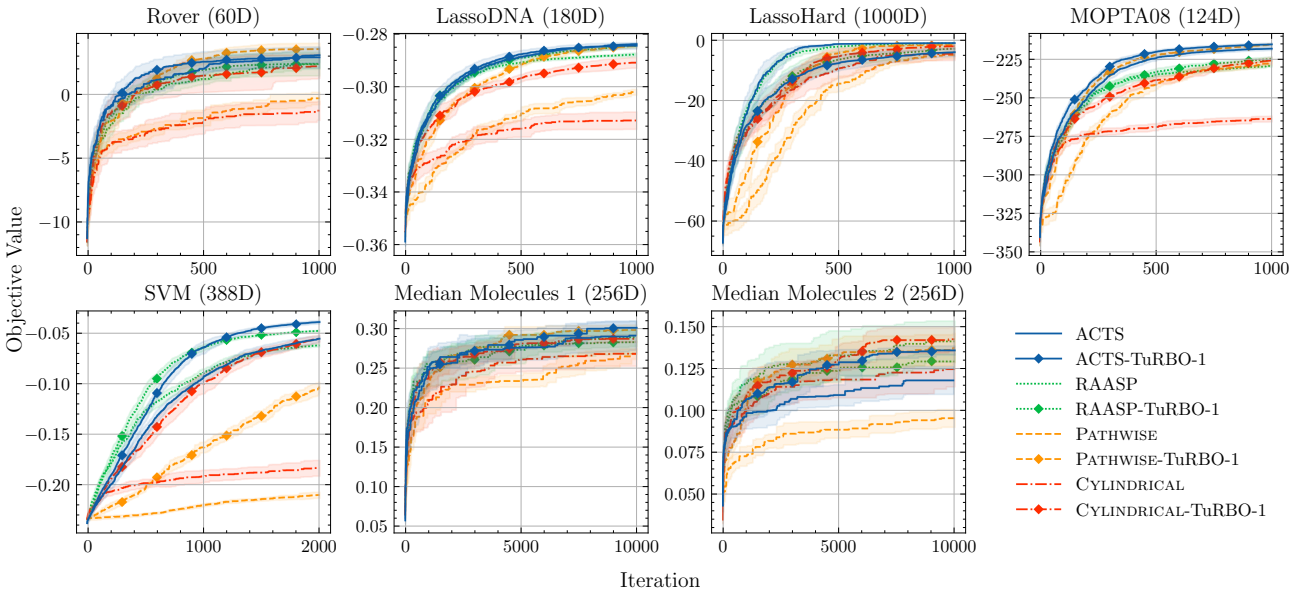

Figure 3: **Optimization performance on several high-dimensional problems.** ACTS ranks highly in all benchmarks, achieving top performance (within two standard errors) on all benchmarks, and outperforming RAASP (with significance) on MOPTA08, SVM, and Median Molecules 1. (Note that TuRBO does not consistently improve performance on all benchmarks, but the performance boost/regression it induces is consistent across all methods on a given benchmark.)

	ACTS-TuRBO (Ours)	RAASP-TuRBO	Pathwise	LogEI	SAASBO	BAxUS
Rover (60D)	$\mathbf{2.87 \pm 0.85}$	$\mathbf{2.36 \pm 0.94}$	-0.33 ± 0.49	$\mathbf{3.09 \pm 0.44}$	$\mathbf{2.42 \pm 1.01}$	0.67 ± 1.29
LassoDNA (180D)	$\mathbf{-0.28 \pm 0.00}$	$\mathbf{-0.28 \pm 0.00}$	-0.30 ± 0.00	-0.29 ± 0.00	-0.29 ± 0.00	-0.29 ± 0.00
MOPTA08 (124D)	$\mathbf{-215.81 \pm 1.07}$	-225.12 ± 1.46	-227.79 ± 1.51	-217.94 ± 0.67	$\mathbf{-219.16 \pm 2.34}$	-240.85 ± 2.30
SVM (388D)	$\mathbf{-0.04 \pm 0.00}$	-0.05 ± 0.00	-0.21 ± 0.00	-0.05 ± 0.00	-0.15 ± 0.02	-0.10 ± 0.00
M. Mol. 1 (256D)	$\mathbf{0.30 \pm 0.01}$	$\mathbf{0.28 \pm 0.02}$	0.27 ± 0.01	$\mathbf{0.30 \pm 0.01}$	—	$\mathbf{0.29 \pm 0.01}$
M. Mol. 2 (256D)	0.14 ± 0.00	0.13 ± 0.00	0.10 ± 0.01	$\mathbf{0.19 \pm 0.01}$	—	$\mathbf{0.20 \pm 0.01}$

Table 1: Final objective values from ACTS vs. state-of-the-art methods on high-dim. benchmarks.

we seed each method with the same $|\mathcal{D}_t| = 200$ observations and GP hyperparameters,[3] generate 10^4 candidate points, and draw 10^4 Thompson samples from the GP posterior. In Fig. 6 we plot the histogram of $\max_{\boldsymbol{x} \in \tilde{\mathcal{X}}} f(\boldsymbol{x})$ values produced by each method over 500 seeds of this procedure. As we hypothesized, ACTS produces larger values of $\max_{\boldsymbol{x} \in \tilde{\mathcal{X}}} f(\boldsymbol{x})$ than other methods. Moreover, these higher sample function values are correlated with higher objective function values. The bottom row displays the true objective function values for $\arg\max_{\boldsymbol{x} \in \tilde{\mathcal{X}}} f(\boldsymbol{x})$ points of each method, and ACTS achieves the highest objective values.

Ablation Study of Base Policy. To test our hypothesis that ACTS improves any underlying candidate strategy, we apply it to a Sobol base policy on the same benchmark tasks. As shown in Figure 7 (which includes RAASP variants for reference), ACTS provides a significant performance uplift over standard Sobol candidates. This improvement is not replicated by restricting the search space with TuRBO, demonstrating that the benefit stems from ACTS's specific candidate allocation rather than any search space reduction. While ACTS-RAASP remains the top performer, ACTS substantially closes the performance gap between the sophisticated RAASP and naïve Sobol policies.

Ablation Study of Search Space Geometry. In Appendix H, we ablate our choice of search space by testing a ACTS variant that restricts candidates to a one-dimensional line search along the sampled gradient. This 1D alternative performs slightly worse than our proposed axis-aligned cone, suggesting the cone construction strikes a better balance between aggressively reducing the search space and avoiding an overly myopic search direction.

[3]We generate this dataset and hyperparameters from 200 iterations of RAASP-based TS.

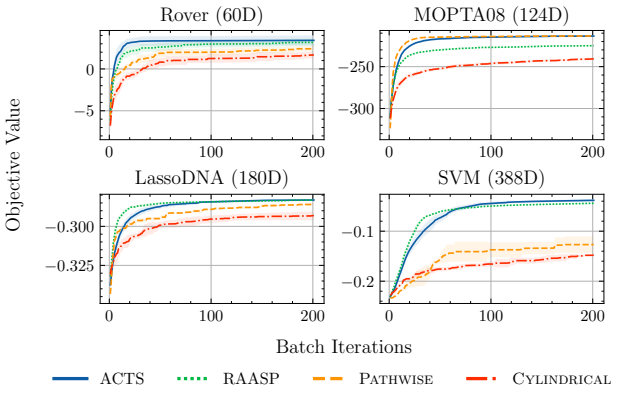

Figure 4: **Batch optimization performance ($q = 100$) of selected high-dimensional problems.** ACTS achieves top results (within two standard errors) on all benchmarks.

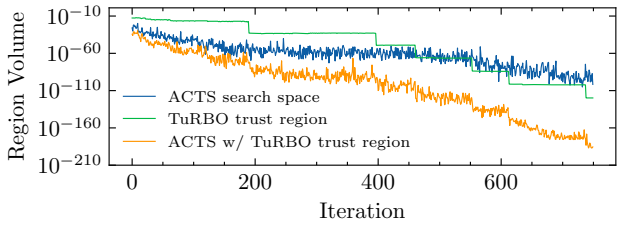

Figure 5: **ACTS increases candidate point density by shrinking search spaces.** ACTS exhibits volumes similar to TuRBO without the direct use of trust regions in two BO runs of Rover, with and without TuRBO, up to the first restart.

Search Behaviour. ACTS substantially reduces the search space, which enables a high density of candidate points in promising regions. While this might suggest a tendency toward local search, our analysis shows this is not the case; queries from ACTS are often less local than TuRBO (with RAASP) or LogEI. In Appendix F, we compute the Traveling Salesman Problem tour for the sequence of queries, a common metric for search locality (Papenmeier et al., 2025), and find that ACTS often produces more exploratory trajectories than other methods. We hypothesize the performance uplift of ACTS stems from a higher-fidelity discretization of the Thompson sample.

5 DISCUSSION

ACTS provides a principled approach to mitigate the curse of dimensionality in candidate-based TS. By adaptively concentrating candidate points in a region aligned with a posterior gradient sample, our method achieves a denser, more effective discretization of promising areas. Our experiments confirm that this strategy yields robust performance across a wide range of high-dimensional benchmarks in both sequential and batch settings, without biasing the search towards being any more local than existing TS methods.

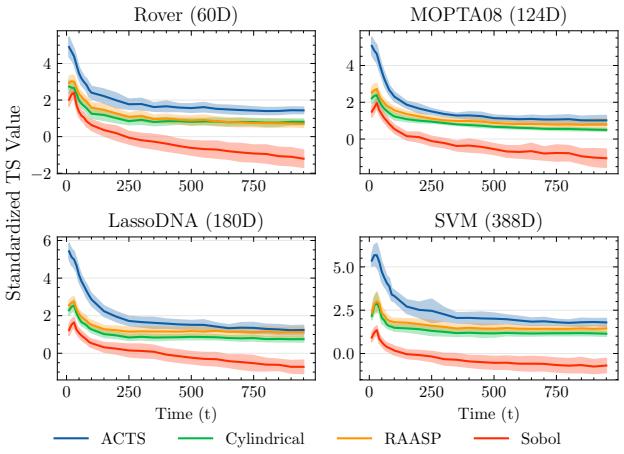

Figure 6: ACTS produces greater TS maxima than other candidate policies on several benchmarks. We generate candidates from each policy, sample from the posterior 100 times, and aggregate over 10 models with standardized outputs.

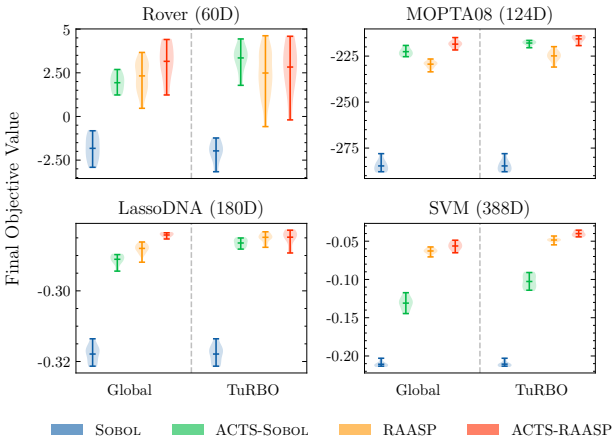

Figure 7: **Optimization performance of Sobol and RAASP policies with and without ACTS on high-dimensional objectives.** RAASP consistently outperforms the poorly-performing Sobol policy. ACTS search spaces provide a significant performance boost to all methods, especially Sobol. This indicates the ACTS search spaces provide better fidelity for TS maximization than TuRBO trust regions.

Limitations and Extensions. As ACTS is a drop-in replacement for standard TS approaches, it does not have many limitations beyond existing TS methods. The primary assumption of ACTS is a once-

differentiable kernel, and thus is suited towards objectives that are also once-differentiable. We also note that reducing the search space based on a gradient is a myopic approximation of the maximizing direction, which, despite global consistency, may be inappropriate in problem settings with low intrinsic lengthscales. However, these situations may nevertheless benefit from other adaptive candidate strategies, of which we have only begun to explore in this work. For example, repeating the autoregression in Eq. 8 could mimic the behavior of gradient descent, yielding an exact local maximizer of the GP sample path. This autoregression would scale poorly and require significant sequential computation, which may offer little benefit as our single-step adaptation performs well in practice. Nevertheless, this extension and others are worth exploring, especially in unique high-dimensional application domains.

Acknowledgements

This research was enabled in part by support provided by the Vector Institute, Advanced Research Computing at the University of British Columbia, and the Digital Research Alliance of Canada. These resources were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute. We acknowledge the support of the Natural Sciences and Engineering Research Council and the Social Sciences and Humanities Research Council of Canada (NSERC: RGPIN-2024-06405, NFRFE-2024-00830). GP is supported by the Canada CIFAR AI Chairs program.

References

Sebastian Ament, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 36, pages 20577–20612. Curran Associates, Inc., 2023.

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient Monte-Carlo Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 33, pages 21524–21538. Curran Associates, Inc., 2020.

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

Hildo Bijl, Thomas B. Schön, Jan-Willem van Wingerden, and Michel Verhaegen. A sequential Monte Carlo approach to Thompson sampling for Bayesian optimization, 2017.

Nathan Brown, Marco Fiscato, Marwin H.S. Segler, and Alain C. Vaucher. Guacamol: Benchmarking models for de novo molecular design. *Journal of Chemical Information and Modeling*, 59(3):1096–1108, Mar 2019. ISSN 1549-9596.

Sathya R Chitturi, Akash Ramdas, Yue Wu, Brian Rohr, Stefano Ermon, Jennifer Dionne, Felipe H da Jornada, Mike Dunne, Christopher Tassone, Willie Neiswanger, et al. Targeted materials discovery using Bayesian algorithm execution. *npj Computational Materials*, 10(1): 156, 2024.

Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Multi-objective Bayesian optimization over high-dimensional search spaces. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 507–517. PMLR, 01–05 Aug 2022.

Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning*, pages 465–472, 2011.

James R Deneault, Jorge Chang, Jay Myung, Daylond Hooper, Andrew Armstrong, Mark Pitt, and Benji Maruyama. Toward autonomous additive manufacturing: Bayesian optimization on a 3D printer. *MRS Bulletin*, 46:566–575, 2021.

David Eriksson and Martin Jankowiak. High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In *Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 493–503. PMLR, 27–30 Jul 2021.

David Eriksson and Matthias Poloczek. Scalable constrained Bayesian optimization. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 730–738. PMLR, 13–15 Apr 2021.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. Gpytorch: Blackbox matrix-matrix Gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

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

Carl Hvarfner, Erik Orm Hellsten, and Luigi Nardi. Vanilla Bayesian optimization performs great in high dimensions. In *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.

Donald R Jones. Large-scale multi-disciplinary mass optimization in the auto industry. In *MOPTA 2008 Conference (20 August 2008)*, volume 64, 2008.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492, 1998.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised Bayesian optimisation via Thompson sampling. In *Artificial Intelligence and Statistics*, 2018.

Natalie Maus, Haydn Jones, Juston Moore, Matt J Kusner, John Bradshaw, and Jacob Gardner. Local latent space Bayesian optimization over structured inputs. In

Advances in Neural Information Processing Systems, volume 35, pages 34505–34518. Curran Associates, Inc., 2022.

Jonas Močkus. On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference Novosibirsk*, pages 400–404, 1975.

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

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Scionce & Business Media, 2012.

Quan Nguyen, Kaiwen Wu, Jacob Gardner, and Roman Garnett. Local Bayesian optimization via maximizing probability of descent. *Advances in neural information processing systems*, 35:13190–13202, 2022.

Leonard Papenmeier, Luigi Nardi, and Matthias Poloczek. Increasing the scope as you learn: Adaptive Bayesian optimization in nested subspaces. In *Advances in Neural Information Processing Systems*, volume 35, pages 11586–11601. Curran Associates, Inc., 2022.

Leonard Papenmeier, Nuojin Cheng, Stephen Becker, and Luigi Nardi. Exploring exploration in bayesian optimization. In *Proceedings of the Forty-First Conference on Uncertainty in Artificial Intelligence*, UAI '25. JMLR.org, 2025.

Geoff Pleiss, Jacob Gardner, Kilian Weinberger, and Andrew Gordon Wilson. Constant-time predictive distributions for Gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4114–4123. PMLR, 10–15 Jul 2018.

Geoff Pleiss, Martin Jankowiak, David Eriksson, Anil Damle, and Jacob Gardner. Fast matrix square roots with applications to Gaussian processes and Bayesian optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

Bahador Rashidi, Kerrick Johnstonbaugh, and Chao Gao. Cylindrical Thompson sampling for high-dimensional Bayesian optimization. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 3502–3510. PMLR, 02–04 May 2024.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Rommel G. Regis and Christine A. Shoemaker. Combining radial basis function surrogates and dynamic coordinate search in high-dimensional expensive black-box optimization. *Engineering Optimization*, 45(5):529–555, 2013.

Ashwin Renganathan and Kade Carlson. qpots: Efficient batch multiobjective Bayesian optimization via pareto optimal thompson sampling. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pages 4051–4059. PMLR, 03–05 May 2025.

Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of Thompson sampling. *Journal of Machine Learning Research*, 17(68):1–30, 2016.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Shubhanshu Shekhar and Tara Javidi. Significance of gradient information in bayesian optimization. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 2836–2844. PMLR, 13–15 Apr 2021.

Aidan Slattery, Zhenghui Wen, Pauline Tenblad, Jesús Sanjosé-Orduna, Diego Pintossi, Tim den Hartog, and Timothy Noël. Automated self-optimization, intensification, and scale-up of photocatalysis in flow. *Science*, 383 (6681):eadj1817, 2024.

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

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *International Conference on Machine Learning*, pages 1015–1022, 2010.

David Sweet. Fast, precise Thompson sampling for Bayesian optimization, 2024.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw bayesian optimization. *arXiv preprint arXiv:1406.3896*, 2014.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. ISSN 00063444.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

Kenan Šehić, Alexandre Gramfort, Joseph Salmon, and Luigi Nardi. Lassobench: A high-dimensional hyperparameter optimization benchmark suite for lasso. In *Proceedings of the First International Conference on Automated Machine Learning*, volume 188 of *Proceedings of Machine Learning Research*, pages 2/1–24. PMLR, 25–27 Jul 2022.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*. Cambridge University Press, 2019.

Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale bayesian optimization in high-dimensional spaces. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 745–754. PMLR, 09–11 Apr 2018.

James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. Efficiently sampling functions from Gaussian process posteriors. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine*

Learning Research, pages 10292–10302. PMLR, 13–18 Jul 2020.

James T. Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Peter Deisenroth. Pathwise conditioning of Gaussian processes. *Journal of Machine Learning Research*, 22(105):1–47, 2021.

Jian Wu, Matthias Poloczek, Andrew G Wilson, and Peter Frazier. Bayesian optimization with gradients. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Kaiwen Wu, Kyurae Kim, Roman Garnett, and Jacob Gardner. The behavior and convergence of local Bayesian optimization. *Advances in neural information processing systems*, 36:73497–73523, 2023.

Zeji Yi, Yunyue Wei, Chu Xin Cheng, Kaibo He, and Yanan Sui. Improving sample efficiency of high dimensional Bayesian optimization with MCMC. In *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*, volume 242 of *Proceedings of Machine Learning Research*, pages 813–824. PMLR, 15–17 Jul 2024.

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. [Yes]
   (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]

	Rover (60D)	MOPTA08 (124D)	LassoDNA (180D)	SVM (388D)
ACTS	0.53 ± 0.14	0.52 ± 0.10	0.53 ± 0.07	0.65 ± 0.12
RAASP	0.44 ± 0.06	0.45 ± 0.06	0.47 ± 0.05	0.53 ± 0.06
Cylindrical	0.75 ± 0.06	0.77 ± 0.06	0.80 ± 0.07	0.92 ± 0.07
Pathwise	1.57 ± 0.53	4.61 ± 2.24	6.09 ± 2.33	11.07 ± 2.93

Table 2: Wallclock time (in seconds ± 2 std. err.) to propose a single candidate of selected TS methods.

A EXPERIMENTAL DETAILS

All our models and experiments were executed using the BoTorch library Balandat et al. (2020). For the GP surrogate, we use a zero-mean function with the squared-exponential kernel, whose lengthscales are inferred from the data using Automatic Relevance Determination Neal (2012). Optimization of the lengthscale leverages a "dimensional-scaled prior" (DSP), where the lengthscale prior scales with the square-root of the dimensionality Hvarfner et al. (2024). Following use of the DSP, the outputscale of this model is fixed at 1. We also standardize the data during training to have zero mean and unit variance. On benchmarks with ≥ 10000 iterations we make use of GPyTorch's conjugate gradient-based inference (Gardner et al., 2018); otherwise we perform hyperparameter optimization and posterior inference without approximation.

The Log Expected Improvement, TuRBO, Pathwise, and SAASBO approaches use the default BoTorch implementations. TuRBO hyperparameters include a default length of 0.8, a minimum side-length of 0.5^7, a max length of 1.6, and an adaptive failure tolerance of $\left\lceil \max\left\{\frac{4}{q}, \frac{d}{q}\right\}\right\rceil$ for batch size q. When a restart is triggered for TuRBO, we include the 30 Sobol initial points in the evaluation budget. For Pathwise, by default BoTorch uses 1024 features. SAASBO uses the qLogNoisyExpectedImprovement acquisition function and a noise variance of 10^{-6}. Training the fully-Bayesian SAAS prior uses the NUTS sampler with 512 warmup steps, 256 samples, and a thinning factor of 16. For BAxUS, we use the original author implementation found at `https://github.com/LeoIV/BAxUS`. For Cylindrical TS, we use the original author implementation found at `https://github.com/HW-AI-Research/CTS-HDBO`, using the same hyperparameters. Specifically, we set $\sigma_{\text{init}} = 0.125$ and when not using TuRBO, we set $R = 2\sqrt{d}$. We note that Cylindrical TS leverages local modelling by only performing hyperparameter optimization using observed points that live inside the spherical trust region, whereas the original TuRBO implementation always fits using all points possible (within each restart).

We run our experiments on a server with two Intel Xeon Silver 4116 CPUs with 192 GB of RAM and four NVIDIA Tesla V100 32GB GPUs. However, when using ACTS with $M = 10^4$ points, we typically only use a single core and 8 GB of RAM. A single optimization run on SVM with 1000 iterations typically requires 2 hours of wall time to complete for our method, RAASP, and BAxUS. However, LogEI and Pathwise uses significantly more time due to gradient optimization steps, typically 6 or more hours.

A.1 Wallclock Time

In Table 2, we report the wallclock time of mean acqusition time over 10 runs for selected TS methods. The fastest method is RAASP, followed by ACTS, then Cylindrical TS, and finally Pathwise. ACTS incurs a minor overhead from sampling the gradient, where as Cylindrical TS requires additional time to sample from a truncated Gaussian distribution. All three approaches are limited by the Cholesky decomposition, which is performed on 10^4 points. Lastly, Pathwise requires significantly more time due to the need to optimize the acquisition function using gradient-based optimization.

A.1.1 Memory Usage

In Table 3, we measure the peak memory usage across 10 runs for selected TS methods as reported by PyTorch's `cuda.max_memory_allocated` in GiB. Similarly to the wall-clock time, the chief space usage comes from the Cholesky decomposition, which is performed on 10^4 points for all methods except Pathwise, which only evaluates the posterior for random restarts on 512 points for BoTorch's `sample_around_best` optimization initializer.

	Rover (60D)	MOPTA08 (124D)	LassoDNA (180D)	SVM (388D)
ACTS	4.64 ± 0.30	4.58 ± 0.18	4.57 ± 0.06	4.73 ± 0.23
RAASP	3.93 ± 0.10	3.95 ± 0.10	3.96 ± 0.10	4.01 ± 0.10
Cylindrical	4.77 ± 0.10	4.79 ± 0.10	4.82 ± 0.11	4.93 ± 0.13
Pathwise	0.08 ± 0.04	0.12 ± 0.06	0.17 ± 0.08	0.36 ± 0.14

Table 3: Mean peak memory usage (in GiB ± 2 std. err.) of selected TS methods.

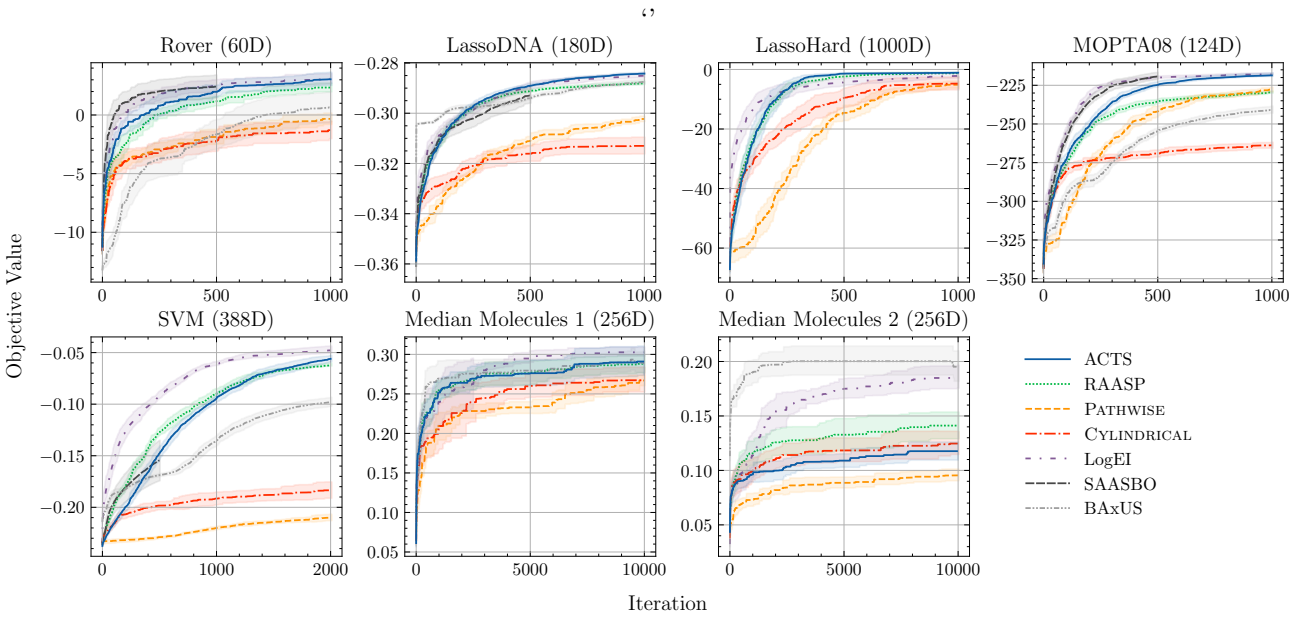

Figure 8: **Optimization performance of high-dimensional optimization problems without TuRBO**. The TS methods exhibit strong performance compared to several non-TS benchmarks, though notably the LogEI and BAxUS baselines obtain significantly better performance on Median Molecules 2.

B EXTENDED OPTIMIZATION RESULTS

In Figs. 8 and 9, we provide the full results of the high-dimensional optimization problems. We compare to SAASBO Eriksson and Jankowiak (2021), which uses a fully-Bayesian surrogate model to identify sparse subspaces for optimization, BAxUS Papenmeier et al. (2022), a latent-space approach that progressively increases effective search dimension, and the Log Expected Improvement in conjunction with the dimensional-scaled prior Ament et al. (2023); Hvarfner et al. (2024). We observe that ACTS exhibits strong optimization performance in several objectives, matching or exceeding performance on many benchmarks. The notable exceptions are the SVM and Median Molecules 2 benchmarks, where ACTS is outperformed by LogEI and BAxUS. We do note however that ACTS achieves much higher objective values than BAxUS on all other benchmarks.

We additionally perform a statistical analysis using the Friedman and Nemenyi tests and aggregate rankings across all sequential optimization benchmarks, summarized in Figure 10. ACTS achieves the highest mean rank among all TS strategies. It is statistically distinguishable ($p < 0.05$) from Pathwise and Cylindrical TS in the global setting. Even within the competitive TuRBO setting, ACTS retains the top rank.

C EXTENDED BATCH OPTIMIZATION RESULTS

Here, we complement our earlier $q = 100$ batch optimization setting with $q = 10$ (Figure 11) and $q = 50$ (Figure 12). The strong optimization performance of ACTS is maintained across the batch settings, only losing to RAASP in the short term (we see in $q = 50$ ACTS eventually overtakes RAASP). Pathwise exhibits varying performance while Cylindrical TS ranks poorly.

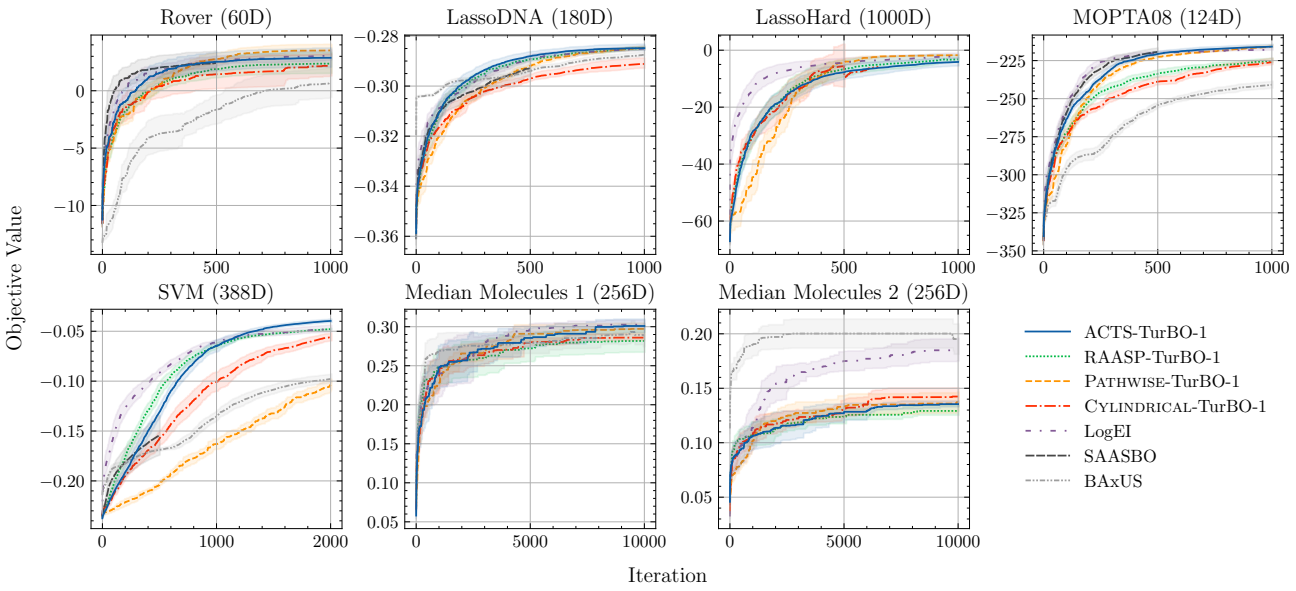

Figure 9: **Optimization performance of high-dimensional optimization problems with TuRBO**. When using TuRBO trust regions, the Thompson sampling methods' optimization performance can be enhanced.

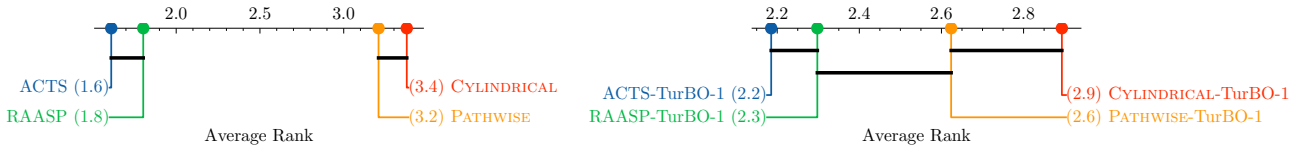

Figure 10: Ranking significance of selected TS methods. **Left:** Global setting. **Right:** TuRBO setting.

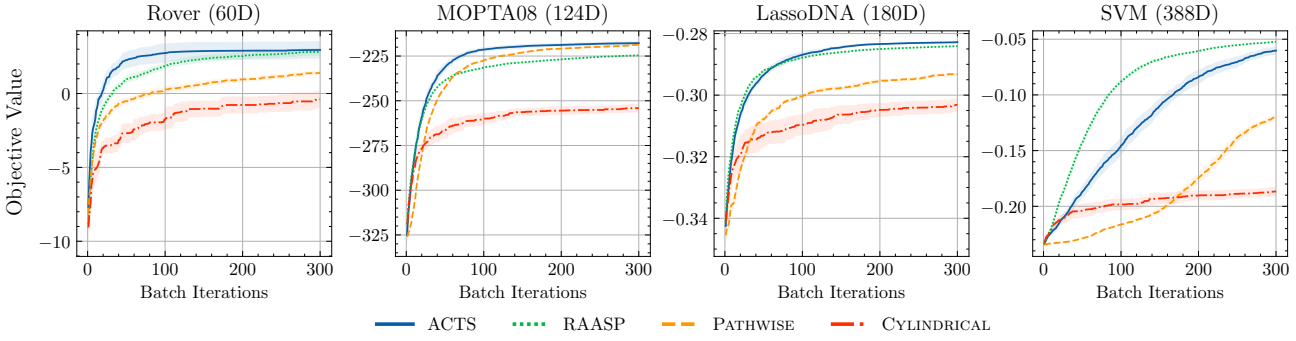

Figure 11: **Optimization performance of selected high-dimensional benchmarks of small batches ($q = 10$).** ACTS achieves strong performance in all benchmarks compared to other TS methods, though potentially requiring a larger evaluation budget to overcome RAASP in SVM.

Batch Diversity. When constructing a batch, it is typically important to have a diverse set of points so that posterior uncertainty is reduced in many locations. That is, no two points in the batch should be too close nor all points in the batch be too far apart. Our approach naturally balances this tradeoff. At the start of optimization, when few points are known, $\{\nabla f^{(k)}\}_{k=1}^{q}$ are likely to be dissimilar, inducing different search spaces for each point in the batch. Uncertainty in our gradient samples decrease as more points near $\boldsymbol{x}_0$ are explored, and thus points in the batch will be more similar. However, randomness in our candidate points and through sampling inherently affords us some diversity even when exploiting near local maxima.

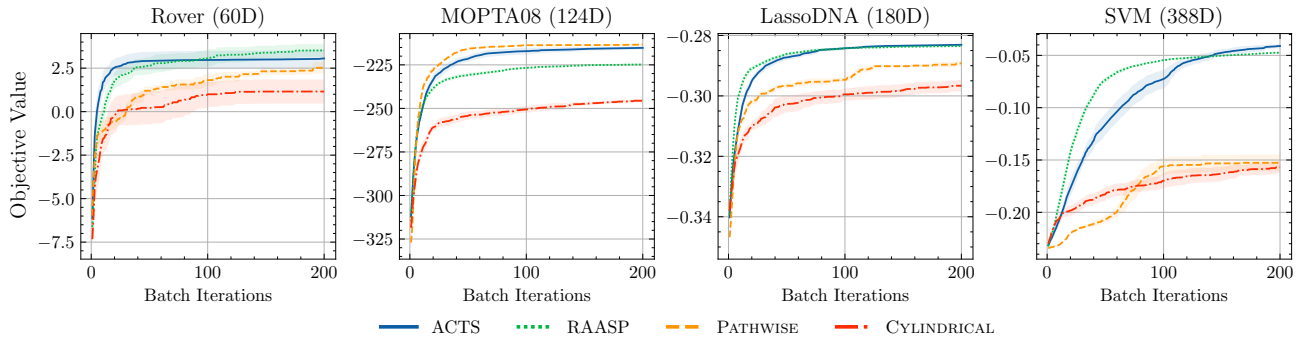

Figure 12: **Optimization performance of selected high-dimensional benchmarks of medium-sized batches ($q = 50$).** ACTS ranks high among TS methods, with Pathwise finding the best performance in MOPTA08.

D GLOBAL CONSISTENCY OF ACTS

We show that ACTS will query the global maximizer as the number of evaluations tends to infinity.

Theorem 1. *Choose any $\epsilon > 0$ and assume the following:*

- *A1. k is a non-degenerate kernel with $k(\boldsymbol{x}, \boldsymbol{x}) = 1$ for all $\boldsymbol{x} \in \mathcal{X}$ (standard stationary kernel with no outputscale)*
- *A2. k, ∇k, and $k\nabla^\top$ are bounded.*
- *A3. $|F(\boldsymbol{x})| \leq B$, $\forall \boldsymbol{x} \in \mathcal{X}$ (bounded objective function)*
- *A4. $y_t = F(\boldsymbol{x}_t) + \eta$, $|\eta| \leq b$ (bounded observation noise)*
- *A5. $\mathcal{X} \subset \mathbb{R}^d$ is compact with $d < \infty$*
- *A6. The candidate set $\widetilde{\mathcal{X}}_t$ is constructed by sampling M points independently from some fully-supported distribution over $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)} \cap \mathcal{X}_\epsilon$, where $\mathcal{X}_\epsilon$ is some ϵ-covering of $\mathcal{X}$ (with respect to $\|\cdot\|_2$).*

Then running Bayesian optimization with ACTS will, in finite time, almost surely query some point $\boldsymbol{x}$ such that $\|\boldsymbol{x} - \boldsymbol{x}^*\|_2 \leq \epsilon$, where $\boldsymbol{x}^* = \arg\max_{x \in \mathcal{X}} F(\boldsymbol{x})$ is the global maximizer of F.

We decompose this proof into several lemmas and delay presentation to the end. The general outline of our proof is to use contradiction to define an event where some point $\boldsymbol{x}_u$ has not been observed, and show that the event that the ACTS search space will occur infinitely often. We note that, by A6, ACTS will only ever sample candidate points from $\mathcal{X}_\epsilon$, and thus all observations $\{\boldsymbol{x}_i\}_{i=1}^\infty$ will be from the finite set $\mathcal{X}_\epsilon$.

The first lemma is a generalized concentration inequality that we will use to upper bound the posterior maximizer over subsets of $\mathcal{X}_\epsilon - \{\boldsymbol{x}_u\}$.

Lemma 1. *Fix $p \in \mathbb{N}$. For any constant $\mu_0 \in \mathbb{R}$, any (strictly) positive definite matrix $\boldsymbol{S}_0 \in \mathbb{R}^{p \times p}$, and any $\alpha > \sqrt{p}\|\boldsymbol{S}_0\|_2 + \mu_0$, there exists some constant $C_0 > 0$ that depends only on p, μ_0, $\boldsymbol{S}_0$, and α such that*

$$\Pr_{\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})} (\|\boldsymbol{z}\|_\infty < \alpha) > C_0$$

for all $\boldsymbol{m} \in \mathbb{R}^p$ with $\|\boldsymbol{m}\|_2 \leq \mu_0$ and all $\boldsymbol{S}$ with $\boldsymbol{0} \prec \boldsymbol{S} \preceq \boldsymbol{S}_0$.

Proof. We have that:

$$\begin{aligned}
\Pr_{\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})} (\|\boldsymbol{z}\|_\infty < \alpha) &= 1 \; - \Pr_{\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{S})} (\|\boldsymbol{z}\|_\infty \geq \alpha) \\
&= 1 \; - \Pr_{\boldsymbol{z}' \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})} \left(\|\boldsymbol{S}^{1/2}\boldsymbol{z}' + \boldsymbol{m}\|_\infty \geq \alpha \right)
\end{aligned}$$

We have that the function $g(\boldsymbol{z}') := \|\boldsymbol{S}^{1/2}\boldsymbol{z}' + \boldsymbol{m}\|_\infty$ is Lipschitz continuous with respect to the Euclidean norm (as it is the composition of Lipschitz continuous functions) with Lipschitz constant $\sqrt{\|\boldsymbol{S}\|_2}$. By concentration of

measure for Lipschitz functions of Gaussian random variables (Wainwright, 2019, Thm. 2.26), for any $t > 0$ we have that

$$\Pr_{\boldsymbol{z}\sim\mathcal{N}(\boldsymbol{m},\boldsymbol{S})}\left(\|\boldsymbol{z}\|_\infty - \mathbb{E}[\|\boldsymbol{z}\|_\infty] \geq t\right) \leq \exp\left(-\frac{t^2}{2\|\boldsymbol{S}\|_2}\right) \leq \underbrace{\exp\left(-\frac{t^2}{2\|\boldsymbol{S}_0\|_2}\right)}_{:=C_0} < 1.,$$

where the last inequality follows from the fact that $\|\boldsymbol{S}\|_2 \leq \|\boldsymbol{S}_0\|_2$. Now note that:

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{z}\|_\infty] \leq \mathbb{E}[\|\boldsymbol{z}\|_2] &\leq \mathbb{E}[\|\boldsymbol{z}-\boldsymbol{m}\|_2] + \|\boldsymbol{m}\|_2 \\
&\leq \mathbb{E}_{z'\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})}[\|\boldsymbol{S}\boldsymbol{z}'\|_2] + \|\boldsymbol{m}\|_2 \\
&\leq \|\boldsymbol{S}\|_2 \mathbb{E}_{z'\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})}[\|\boldsymbol{z}'\|_2] + \|\boldsymbol{m}\|_2 \\
&\leq \|\boldsymbol{S}\|_2 \sqrt{\mathbb{E}_{z'\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})}[\|\boldsymbol{z}'\|_2^2]} + \|\boldsymbol{m}\|_2 && \text{(Jensen's inequality)} \\
&\leq \sqrt{p}\|\boldsymbol{S}\|_2 + \|\boldsymbol{m}\|_2 \leq \sqrt{p}\|\boldsymbol{S}_0\|_2 + \mu_0.
\end{aligned}$$

Setting $t = \alpha - \left(\sqrt{p}\|\boldsymbol{S}_0\|_2 + \mu_0\right)$ gives the desired result. □

We now use this lemma to bound the posterior maximizer over any finite subset of $\mathcal{X}_\epsilon$.

Lemma 2. *Assume A1-A6. For any $\mathcal{S} \subseteq \mathcal{X}_\epsilon$, there exists some constants $C_\mathcal{S} > 0$, $U_S < \infty$ such that, for all $t > 0$ and $\mathcal{D}_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^t \in (\mathcal{X}_\epsilon \times \mathbb{R})^t$,*

$$\Pr\left(\|\boldsymbol{f}_\mathcal{S}\|_\infty < U_S \mid \mathcal{D}_t\right) > C_\mathcal{S}.$$

Proof of Lemma 2. Define $\boldsymbol{f}_\mathcal{S} := [f(\boldsymbol{x})]_{\boldsymbol{x}\in\mathcal{S}}$ as the vector of (noiseless and unobserved) objective function values at all points in $\mathcal{S}$. The posterior distribution $f_\mathcal{S} \mid \mathcal{D}_t$ is a multivariate normal random variable, with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ given by the standard GP posterior formulas:

$$\begin{gathered}
\boldsymbol{f}_\mathcal{S} \mid \mathcal{D}_t \;\sim\; \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right) \\
\boldsymbol{\mu} := \boldsymbol{K}_{\mathcal{S},\mathcal{O}}\left(\boldsymbol{K}_\mathcal{O} + \boldsymbol{D}\right)^{-1}\bar{\boldsymbol{y}}_\mathcal{O}, \qquad \boldsymbol{\Sigma} := \boldsymbol{K}_\mathcal{S} - \boldsymbol{K}_{\mathcal{S},\mathcal{O}}\left(\boldsymbol{K}_\mathcal{O} + \boldsymbol{D}\right)^{-1}\boldsymbol{K}_{\mathcal{S},\mathcal{O}}^\top,
\end{gathered}$$

where $\mathcal{O} := \cup_{i=1}^t \{\boldsymbol{x}_i\} \subseteq \mathcal{X}_\epsilon$ is the set of all inputs that have been observed at least once; $\boldsymbol{K}_\mathcal{O} \in \mathbb{R}^{|\mathcal{O}|\times|\mathcal{O}|}$ and $\boldsymbol{K}_\mathcal{O} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ are the Gram matrices of all points in $\mathcal{O}$ and $\mathcal{S}$ respectively; $\boldsymbol{K}_{\mathcal{S},\mathcal{O}} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{O}|}$ is the cross-Gram matrix between $\mathcal{S}$ and $\mathcal{O}$; $\boldsymbol{D} \in \mathbb{R}^{|\mathcal{O}|\times|\mathcal{O}|}$ is a diagonal matrix with $D_{ii} = \sigma^2/n_i$ where n_i is the number of times $\boldsymbol{x}_i$ has been observed; and $\bar{\boldsymbol{y}}_\mathcal{O} \in \mathbb{R}^{|\mathcal{O}|}$ is the vector of average observed values at each input in $\mathcal{O}$.

Letting $\mu(\cdot) : \mathcal{X}_\epsilon \to \mathbb{R}$ be the posterior mean function of f and letting $\mathcal{H}$ be the reproducing kernel Hilbert space defined by $k(\cdot,\cdot)$, we first note that

$$\begin{aligned}
[\boldsymbol{\mu}]_i &= \langle k(\boldsymbol{x}_{\mathcal{S}_i}, \cdot), \mu(\cdot)\rangle_\mathcal{H} \\
&\leq \underbrace{\|k(\boldsymbol{x}_{\mathcal{S}_i}, \cdot)\|_\mathcal{H}}_{1} \quad \underbrace{\|\mu(\cdot)\|_\mathcal{H}}_{\sqrt{\boldsymbol{y}_\mathcal{O}^\top(\boldsymbol{K}_\mathcal{O}+\boldsymbol{D})^{-1}\boldsymbol{K}_\mathcal{O}(\boldsymbol{K}_\mathcal{O}+\boldsymbol{D})^{-1}\boldsymbol{y}_\mathcal{O}}} \\
&\leq \sqrt{\boldsymbol{y}_\mathcal{O}^\top\left(\boldsymbol{K}_\mathcal{O}\right)^{-1}\boldsymbol{y}_\mathcal{O}} \\
&\leq \sqrt{\tfrac{1}{\lambda_{\min}(\boldsymbol{K}_\mathcal{O})}}\|\bar{\boldsymbol{y}}_\mathcal{O}\|_2 \\
&\leq \sqrt{\tfrac{1}{\lambda_{\min}(\boldsymbol{K}_\mathcal{O})}}(B+b)\sqrt{|\mathcal{O}|} \\
&\leq \underbrace{\min_{\mathcal{C}\in 2_\epsilon^\mathcal{X}} \sqrt{\tfrac{1}{\lambda_{\min}(\boldsymbol{K}_\mathcal{C})}}}_{=:C_0<\infty}(B+b)\sqrt{|\mathcal{X}_\epsilon|},
\end{aligned}$$

where $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of a matrix. The finiteness of C_0 follows from the fact that $\mathcal{X}_\epsilon$ is finite (and thus all $\boldsymbol{K}_\mathcal{C}$ are finite matrices) and k is a non-degenerate kernel. Therefore, $\|\boldsymbol{\mu}\|_2 \leq C_0(B+b)\sqrt{|\mathcal{X}_\epsilon|}\sqrt{|\mathcal{S}|} < C_0(B+b)|\mathcal{X}_\epsilon|$. Moreover, we have that

$$\boldsymbol{0} \;\prec\; \boldsymbol{\Sigma} \;\preceq\; \underbrace{\boldsymbol{K}_\mathcal{S}}_{=:\boldsymbol{\Sigma}_0}$$

Because $\mathcal{S}$ is finite and k is bounded on $\mathcal{X} \times \mathcal{X}$, we have that $\|\boldsymbol{\Sigma}_0\|_2 < \infty$. Defining $U_\mathcal{S}$ to be any constant such that

$$C_0(B+b)|\mathcal{X}_\epsilon| \;<\; U_\mathcal{S} \;<\; \infty,$$

by Lemma 1, there exists some constant $C_\mathcal{S} > 0$ that depends only on $|\mathcal{S}|$, B, b, $U_\mathcal{S}$, and $\boldsymbol{\Sigma}_0$ such that

$$\Pr\left(\|\boldsymbol{f}_\mathcal{S}\|_\infty < U_\mathcal{S} \;\mid\; \mathcal{D}_t\right) \;=\; \Pr_{\mathbf{z}\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left(\|\boldsymbol{z}\|_\infty < U_\mathcal{S}\right) \;>\; C_\mathcal{S}.$$

Note that this bound does not depend on t or $\mathcal{D}_t$, and thus the result follows. □

The next two lemmas establish a lower bound for posterior samples at any unobserved point cointained in the ACTS search space.

Lemma 3. *For any $\boldsymbol{x}_0, \boldsymbol{x}_u \in \mathcal{X}$, let $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u) = \{\nabla f(\boldsymbol{x}_0) \in \mathbb{R}^d \mid \boldsymbol{x}_u \in \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}\}$ be the set of gradients for $\boldsymbol{x}_0$ that generate ACTS search spaces containing $\boldsymbol{x}_u$. Then $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$ has strictly positive Lebesgue measure.*

Proof. We can re-write $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$ as the following:

$$\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u) = \left\{\boldsymbol{v} \in \mathbb{R}^d \mid v_i > 0 \text{ if } [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i > 0, v_i < 0 \text{ if } [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i < 0, v_i \in \mathbb{R} \text{ if } [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i = 0\right\}.$$

Consider the indicator vector $\boldsymbol{w} \in \mathbb{R}^d$ where

$$w_i = \begin{cases} 1 & [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i > 0 \\ -1 & [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i < 0 \\ 1 & [\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i = 0, \end{cases}$$

where in the last case we have arbitrarily chosen $w_i = 1$. We note that $\boldsymbol{w}$ trivially satisfies the conditions to be an element of $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$. Now consider any $\boldsymbol{v}' \in \mathcal{B}(\boldsymbol{w}, 1/2)$, where $\mathcal{B}(\boldsymbol{w}, 1/2)$ denotes the open ball of radius $1/2$ centered at $\boldsymbol{w}$, such that $|v'_i - w_i| \leq \|\boldsymbol{v}' - \boldsymbol{w}\| < 1/2$ for all $i \in [1, d]$. If $[\boldsymbol{x}_u]_i - [\boldsymbol{x}_0]_i > 0$, then $w_i = 1$ and thus $v'_i \in (0.5, 1.5)$, thus satisfying the condition $v'_i > 0$ that is required for $\boldsymbol{v}' \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$. The other cases can be verified similarly. Because this holds for all $i \in [1, d]$, we have that all $\boldsymbol{v}' \in \mathcal{B}(\boldsymbol{w}, 1/2)$ are elements of $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$. Thus $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$ contains a non-empty open-set and so it has strictly positive Lebesgue measure. □

Lemma 4. *Assume A1-A6. Let $\boldsymbol{x}_u \in \mathcal{X}_\epsilon$ be a fixed point that ACTS has not observed, and define $\mathcal{S}$ as in Lemma 2. Given the incumbent $\boldsymbol{x}_0$, define $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u) = \{\nabla f(\boldsymbol{x}_0) \in \mathbb{R}^d \mid \boldsymbol{x}_u \in \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}\}$ as in Lemma 3. Given some $U < \infty$, there exists some constant C_U such that*

$$\Pr\left(\nabla f(\boldsymbol{x}_0) \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u),\; f(\boldsymbol{x}_u) > U \;\mid\; \boldsymbol{f}_\mathcal{S}\right) \geq C_U > 0$$

for all $\|\boldsymbol{f}_\mathcal{S}\|_\infty \leq U$.

Proof. From the Jacobian GP model, we have the following posterior distributions:

$$\begin{bmatrix} f(\boldsymbol{x}_u) \\ \nabla f(\boldsymbol{x}_0) \end{bmatrix} \;\mid\; \boldsymbol{f}_\mathcal{S} \;\sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

$$\boldsymbol{\mu} = \begin{bmatrix} k(\boldsymbol{x}_u, \mathcal{S}) \\ \nabla k(\boldsymbol{x}_0, \mathcal{S}) \end{bmatrix} k(\mathcal{S}, \mathcal{S})^{-1} \boldsymbol{f}_\mathcal{S}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} k(\boldsymbol{x}_u, \boldsymbol{x}_u) & k(\boldsymbol{x}_u, \boldsymbol{x}_0)\nabla^\top \\ \nabla k(\boldsymbol{x}_0, \boldsymbol{x}_u) & \nabla k(\boldsymbol{x}_0, \boldsymbol{x}_0)\nabla^\top \end{bmatrix} - \begin{bmatrix} k(\boldsymbol{x}_u, \mathcal{S}) \\ \nabla k(\boldsymbol{x}_0, \mathcal{S}) \end{bmatrix} k(\mathcal{S}, \mathcal{S})^{-1} \begin{bmatrix} k(\boldsymbol{x}_u, \mathcal{S}) \\ \nabla k(\boldsymbol{x}_0, \mathcal{S}) \end{bmatrix}^\top$$

Note that

$$||k(\mathcal{S}, \mathcal{S})^{-1}|| = \frac{1}{\lambda_{\min}(K_\mathcal{S})} < \infty,$$

where $\lambda_{\min}$ denotes the minimum eigenvalue. (Finiteness follows from the fact that $\mathcal{S}$ is finite and k is a non-degenerate kernel.) Thus, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are all bounded quantities due to the boundedness of k and ∇k. We write the desired probability as the integral

$$\int_{\boldsymbol{g}\in\mathcal{C}} \int_{y\in(U,\infty)} p\left(\nabla f(\boldsymbol{x}_0) = \boldsymbol{g}, f(\boldsymbol{x}_u) = y \mid \boldsymbol{f}_\mathcal{S}\right) d\boldsymbol{g}\, dy$$

Given any $\boldsymbol{g}$, y, and $\boldsymbol{f}_\mathcal{S}$, the density on the inside of the integral is Gaussian and thus strictly positive. By Lemma 3, the integration sets have strictly positive measure, and thus the integral is strictly positive for any choice of U and $\boldsymbol{f}_\mathcal{S}$. Finally, for any fixed U, we note that this integral is continuous with respect to $\boldsymbol{f}_\mathcal{S}$, and thus by the extreme value theorem, it attains its minimum C_u over the compact set $\{\boldsymbol{f}_\mathcal{S} \mid \|\boldsymbol{f}_\mathcal{S}\|_\infty \leq U\}$. Therefore, this integral is bounded below by $C_u > 0$. □

Now we are ready to prove Theorem 1.

Proof of Theorem 1. Assume to the contrary that there exists some $\boldsymbol{x}_u$ that will never be observed (i.e., $F(\boldsymbol{x}_u)$ is never evaluated). Fix $\mathcal{S} = \mathcal{X}_\epsilon - \{\boldsymbol{x}_u\}$. Consider the event E_t defined as:

$$E_t = (f(\boldsymbol{x}_u) > U_\mathcal{S}, \boldsymbol{x}_u \in \tilde{\mathcal{X}}_t, ||\boldsymbol{f}_\mathcal{S}||_\infty \leq U_\mathcal{S}) \mid \mathcal{D}_t.$$

where $\boldsymbol{x}_0$ is the incumbent at iteration t and $U_\mathcal{S} < \infty$ is a constant to be determined later. Intuitively, this is an event where $\boldsymbol{x}_u$ is a candidate point and the posterior maximizer at iteration t. We will express the probability of E_t as an integral over the joint density

$$p\left(\mathbf{1}[\boldsymbol{x}_u \in \tilde{\mathcal{X}}_t], \nabla f(\boldsymbol{x}_0), f(\boldsymbol{x}_u), \boldsymbol{f}_\mathcal{S} \mid \mathcal{D}_t\right) = p\left(\mathbf{1}[\boldsymbol{x}_u \in \tilde{\mathcal{X}}_t] \mid \nabla f(\boldsymbol{x}_0)\right) p\left(\nabla f(\boldsymbol{x}_0), f(\boldsymbol{x}_u) \mid \boldsymbol{f}_\mathcal{S}, \mathcal{D}_t\right) p\left(\boldsymbol{f}_\mathcal{S} \mid \mathcal{D}_t\right)$$

where the first factor is conditionally independent of $\mathcal{D}_t$ and $f(\boldsymbol{x}_u)$ (as $\tilde{\mathcal{X}}_t$ is constructed by sampling M points from $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$), and the second factor is conditionally independent of $\mathcal{D}_t$ given that $\{x_i\}_{i=1}^t \subseteq \mathcal{S}$. The probability of E_t can be thus be decomposed as

$$P(E_t) = \int_{\boldsymbol{g} \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)} P\left(\boldsymbol{x}_u \in \tilde{\mathcal{X}}_t \mid \nabla f(\boldsymbol{x}_0) = \boldsymbol{g}\right) \int_{\boldsymbol{y}' \in \mathcal{U}} p\left(\boldsymbol{f}_\mathcal{S} = \boldsymbol{y}' \mid \mathcal{D}_t\right) \int_{y \in (U, \infty)} p\left(\nabla f(\boldsymbol{x}_0) = \boldsymbol{g}, f(\boldsymbol{x}_u) = y \mid \boldsymbol{f}_\mathcal{S} = \boldsymbol{y}'\right) \times d\boldsymbol{g}\, d\boldsymbol{y}'\, dy,$$

where $\mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u) = \{\nabla f(\boldsymbol{x}_0) \in \mathbb{R}^d \mid \boldsymbol{x}_u \in \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}\}$ as in Lemma 3 and $\mathcal{U} := \{\boldsymbol{y} \mid ||\boldsymbol{y}||_\infty < U\}$. First, as $\boldsymbol{g} \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)$ implies $\boldsymbol{x}_u \in \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$, we note that

$$P\left(\boldsymbol{x}_u \in \tilde{\mathcal{X}}_t \mid \nabla f(\boldsymbol{x}_0) = \boldsymbol{g}\right) \geq \underbrace{\min_{\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}} \min_{\boldsymbol{x}_u \in \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}} P\left(\boldsymbol{x}_u \in \tilde{\mathcal{X}}_t \mid \mathcal{T}_{\nabla f(\boldsymbol{x}_0)}\right)}_{=:C_1} > 0 \qquad \forall \boldsymbol{g} \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u),$$

where the first inequality uses the fact that the set of all possible $\mathcal{T}_{\nabla f(\boldsymbol{x}_0)}$ is finite (as $\boldsymbol{x}_0 \in \mathcal{X}_\epsilon$, $\mathcal{X}_\epsilon$ is finite, and there are finitely-many axis-aligned cones for each $\boldsymbol{x}_0$) and the last inequality is due to A6. Thus, after applying Fubini's theorem we have that:

$$P(E_t) \geq C_1 \int_{\boldsymbol{y}' \in \mathcal{U}} p\left(\boldsymbol{f}_\mathcal{S} = \boldsymbol{y}' \mid \mathcal{D}_t\right) \int_{\boldsymbol{g} \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)} \int_{y \in (U, \infty)} p\left(\nabla f(\boldsymbol{x}_0) = \boldsymbol{g}, f(\boldsymbol{x}_u) = y \mid \boldsymbol{f}_\mathcal{S} = \boldsymbol{y}'\right) \times dy\, d\boldsymbol{g}\, d\boldsymbol{y}'.$$

Next, the innermost double integral is lower bounded by some constant $C_2 > 0$ from Lemma 4. Finally, we have

$$\begin{aligned} P(E_t) &> C_1 C_2 \int_{\boldsymbol{g} \in \mathcal{C}(\boldsymbol{x}_0, \boldsymbol{x}_u)} \int_{\boldsymbol{y}' \in \mathcal{U}} p(\boldsymbol{f}_\mathcal{S} = \boldsymbol{y}' \mid \mathcal{D}_t) d\boldsymbol{y}' d\boldsymbol{g} \\ &= C_1 C_2 P(\|\boldsymbol{f}_\mathcal{S}\|_\infty < U \mid \mathcal{D}_t) \\ &= C_1 C_2 C_3, \end{aligned}$$

where $C_3 > 0$ is the constant denoted as $C_\mathcal{S}$ in Lemma 2 (where we now set $U_\mathcal{S}$ to be the corresponding $U_\mathcal{S}$ from that lemma). We note that these constants are all strictly positive and do not depend on t. Thus, by the counterpart of the Borel-Cantelli lemma, we have that E_t occurs infinitely often, and thus $\boldsymbol{x}_u$ will be selected as the Thompson sample maximizer with probability 1 as $t \to \infty$, contradicting our previous assumption. □

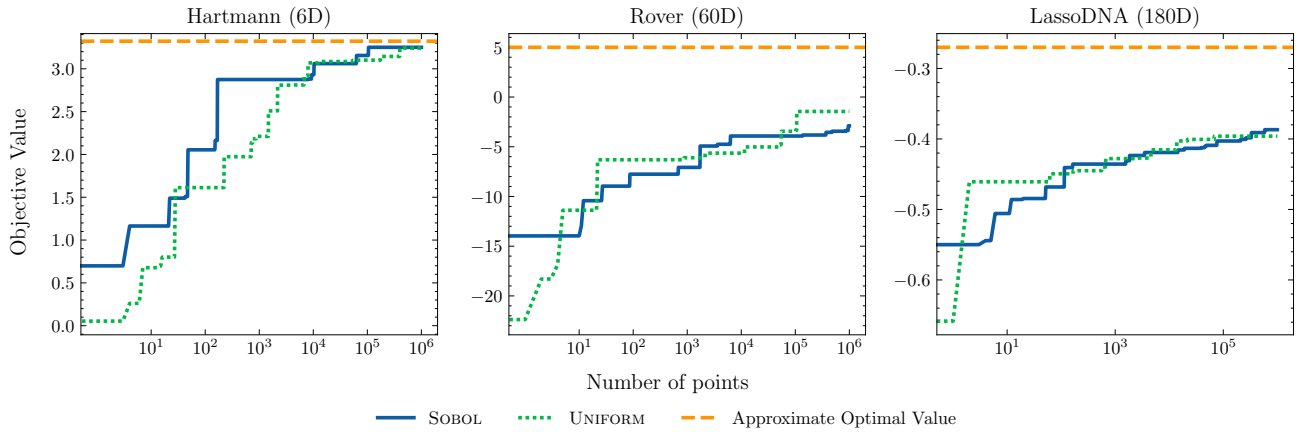

Figure 13: **Global space filling sequence fail to fill the space of high-dimensional problems**. We track the best observed objective value as the number of space filling points increases, up to a budget of 10^6 points. Aside from low-dimensional problems, the best found point remains far from the (approximate) optimal.

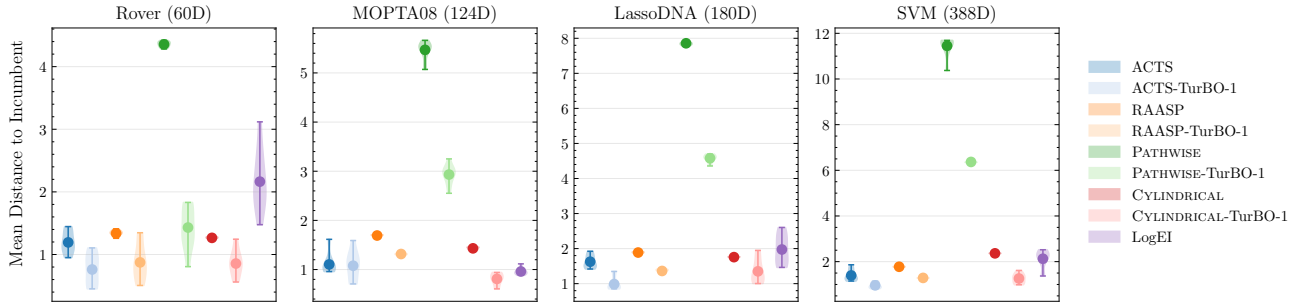

Figure 14: Mean euclidean distance ($\pm$ one standard error) from $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t-1}$ averaged over t and 10 runs (arbitrary units).

E CURSE OF DIMENSIONALITY OF STANDARD THOMPSON SAMPLING

We illustrate the poor approach of "Standard Thompson Sampling" in high-dimensional problems, where such an approach involves using a space-filling sequence to construct the candidate points $\tilde{\mathcal{X}}$. Then $\boldsymbol{x}_{t+1} = \arg\max_{\boldsymbol{x}\in\tilde{\mathcal{X}}} f(x)$, where $f \sim p(f \mid \mathcal{D}_t)$. We recall 10^4 is a typical value for $|\tilde{\mathcal{X}}|$.

We highlight this by giving an evaluation budget of 10^6 points. In Figure 13, we ask: if we are restricted to only the points given by the π_{Sobol} or π_{Uniform} candidate policies, how far from the optimal will our Thompson sampling be restricted to? For small dimensional problems like Hartmann (6D), 10^4 sufficiently provides enough points to have a small amount of regret. However, for higher-dimensional problems, even 10^6 points are insufficient. In other words, given a candidate point budget of 10^4 and an identical batch budget, even after 100 iterations, π_{Sobol} or π_{Uniform} fail to find good solutions. As many standard Thompson sampling approaches use one of these approaches, they are inherently limited in high-dimensional problems.

F LOCALITY

We attempt to characterize the locality of chosen methods in Figures 14 and 15. For Fig. 15, we solve the Travelling Salesman Problem using a greedy approximate heuristic. In both cases, a lower value may be correlated with a more local method. For example, perhaps queries are close to the incumbent, indicating exploitation, or the points queried over time are not too far from each other. We ultimately find that ACTS does not exhibit any more local behavior than other popular methods and find that Pathwise appears to be the most exploratory.

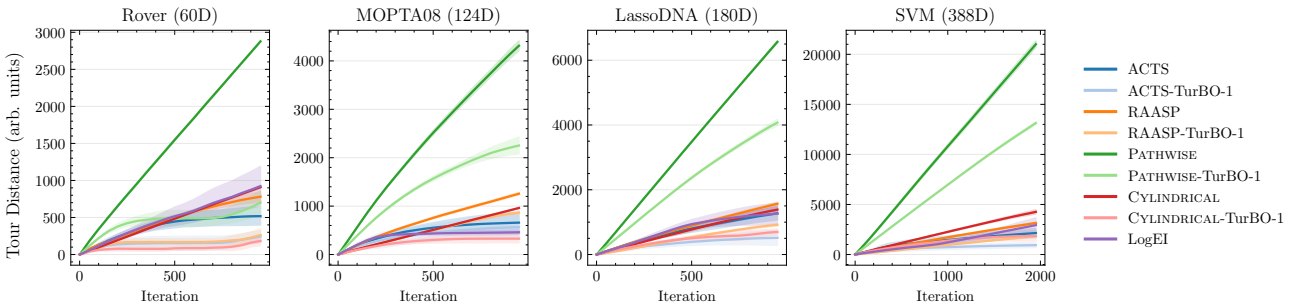

Figure 15: Travelling Salesman Problem tour (± one standard error) for several selected objectives and acquisitions (arbitrary units), averaged over 10 runs.

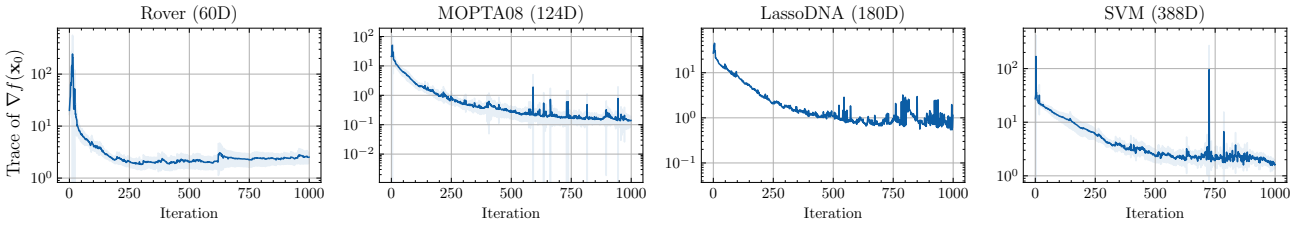

Figure 16: **ACTS reduces the uncertainty appreciably with new observations**. We aggregate over 10 runs, where the uncertainty in the gradient is measured by the trace of the covariance of the gradient distribution.

G REDUCTION IN GRADIENT UNCERTAINTY

As we acquire data over time, especially if we've been stuck at $\boldsymbol{x}_0$ for some time, our estimation of the gradient will improve, even without explicit gradient observations. We reinforce this in Figure 16 and Table 4. Figure 16 shows that the magnitude of the variance of the gradient decreases over time. This is intuitive: as queries fail to provide a new incumbent, this information is used to improve the fit of the GP, thus leading to better gradient estimates. However, the ACTS search space depends only on the direction, and thus the randomness of the sign of the gradient is also important. In Table 4, we compute the fraction of times each gradient sample dimension has a positive/negative sign, average the result over all dimensions, and compute the minimum fraction. A smaller number indicates less disagreement in the sign. In both cases, we find that ACTS is successful in leveraging the GP to reduce the uncertainty in the Thompson sample gradients.

H ABLATION STUDY OF ACTS SEARCH SPACE

We recall our chosen subspace, a cone centered at the incumbent $\boldsymbol{x}_0$ whose rays point in the positive scaled directions of the gradient $\nabla f(\boldsymbol{x}_0)$:

$$\mathcal{T}_{\nabla f(\boldsymbol{x}_0)} = \{\boldsymbol{x}_0 + \boldsymbol{v} \odot \nabla f(\boldsymbol{x}_0) \mid \mathbf{0} \preceq \boldsymbol{v} \in \mathbb{R}^d\} \cap \mathcal{X}.$$

It is natural to consider the case when $\boldsymbol{v} = v\mathbf{1}$ for some $v \geq 0$. That is, we search in the 1D space induced solely by the direction of gradient. Further, we also consider some sparsity in this 1D search: $[\boldsymbol{v}]_j = v[\boldsymbol{b}]_j$, where $[\boldsymbol{b}]_j$ is sampled via Eq. 11 and $v \geq 0$ for $j = 1, \ldots, d$. That is, we apply an adaptive RAASP-style mask to the gradient to induce search sparsity. In Figure 17, we consider these methods as "Cone", "Line" and "Line w/ Random Subspace", respectively and their use with TuRBO trust regions. We allow the maximum value of v to take the half-hypotenuse of the hypercube of the global space or trust region. The 1D line subspaces are not as performant as when using the cone subspace. In Figure 18 we observe that when compared to Figure 6, the 1D line search space is able to find strong TS maximizers, even compared to ACTS however the objective values are highly concentrated. Thus these 1D approaches may suffer from overexploitation.

	$t = 50$	$t = 100$	$t = 200$	$t = 500$	$t = 750$	$t = 1000$
Rover (60D)	0.323	0.345	0.281	0.195	0.198	0.207
MOPTA08 (124D)	0.415	0.383	0.272	0.169	0.128	0.110
SVM (388D)	0.470	0.439	0.384	0.334	0.271	0.246
LassoDNA (180D)	0.446	0.407	0.376	0.213	0.233	0.240

Table 4: **ACTS search space reduces uncertainty while also permitting exploration.** We compute the average minimum fraction of gradient samples that point in the minority direction. A number p in this table can be interpreted to say that, on average, p fraction of gradient samples point in the minority direction for any given dimension.

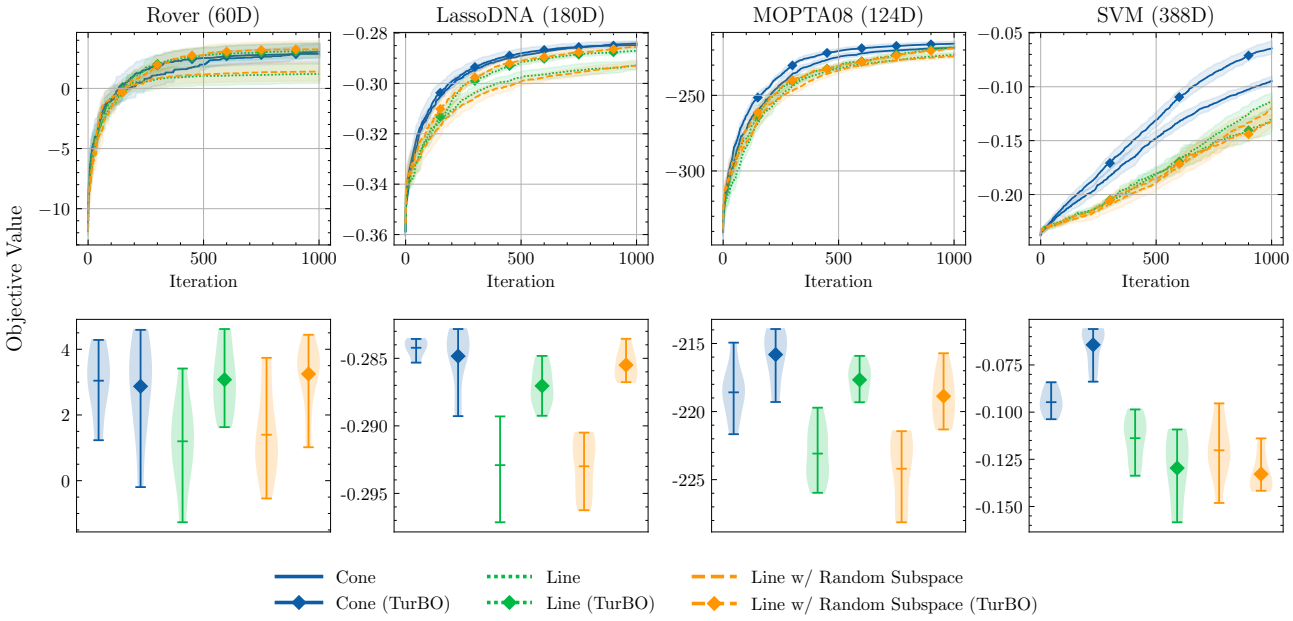

Figure 17: **Optimization performance of ACTS when ablating on different search spaces**. While the 1D line search spaces provide a comparable level of performance in some cases, it fails to be as performant as the proposed cone search space.

I ABLATION STUDY OF ACTS WITH RAASP

We ablate on different parameters of the adaptive RAASP-style policy, defined in Eq. 9, where we recall one of the probabilities as $P = c\gamma$, where $c = 20$ and $\gamma = |\nabla f(\boldsymbol{x}_0)|_j^2 / |\nabla f(\boldsymbol{x}_0)|_2^2$. We first consider the degree to which we favour large elements of the gradient norm, where we settled on the "$L2$" or squared normalized formulation. We consider different "Lp"-style formulations, where $\gamma = |\nabla f(\boldsymbol{x}_0)|_j^p / |\nabla f(\boldsymbol{x}_0)|_p^p$ for $p \in \{1, 2, 3\}$. We further consider "Top-K", where the top $K = \min\{20, d\}$ dimensions are always perturbed, ranked by magnitude. Lastly, we consider using the softmax probabilities on the element-wise magnitude of the gradient, $\gamma = \text{softmax}(|\nabla f(\boldsymbol{x}_0)|)$ where $|\nabla f(\boldsymbol{x}_0)| = [|\nabla f(\boldsymbol{x}_0)|_j]_{j=1}^d$.

In Figs. 19 and 20, we observe mild sensitivity to the exact choice of γ, except when using the "Top-K" approach. "$L3$" may be more performant in some cases. Using TuRBO appears to further reduce the sensitivity.

Finally, we ablate on the choice c, which controls the average number of perturbed dimensions, to scale with the dimensionality in Figs. 21 and 22. We observe that when perturbing in every dimension, the performance varies more relative to $c = 20$, potentially indicating that too many dimensions are perturbed. On the other hand, setting $c = d/10$ has no appreciable effect. Similarly, we observe when using TuRBO trust regions that sensitivity to this parameter is less emphasized.

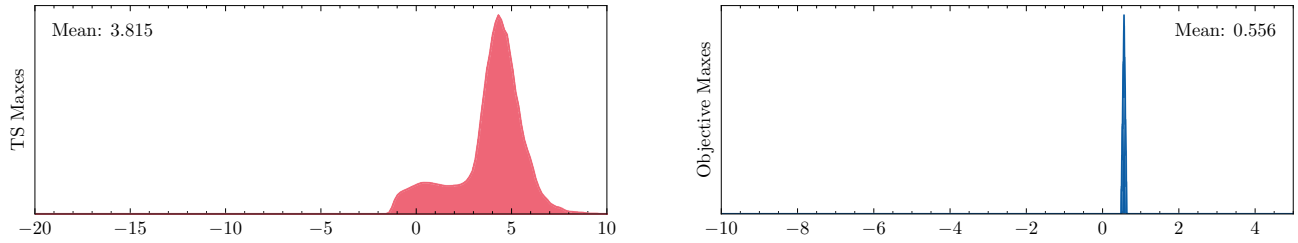

Figure 18: Candidate point quality of 1D line subspace. (See Figure 6 to compare). While the observed TS maxes are high, the objective function values at these maxima are highly concentrated.

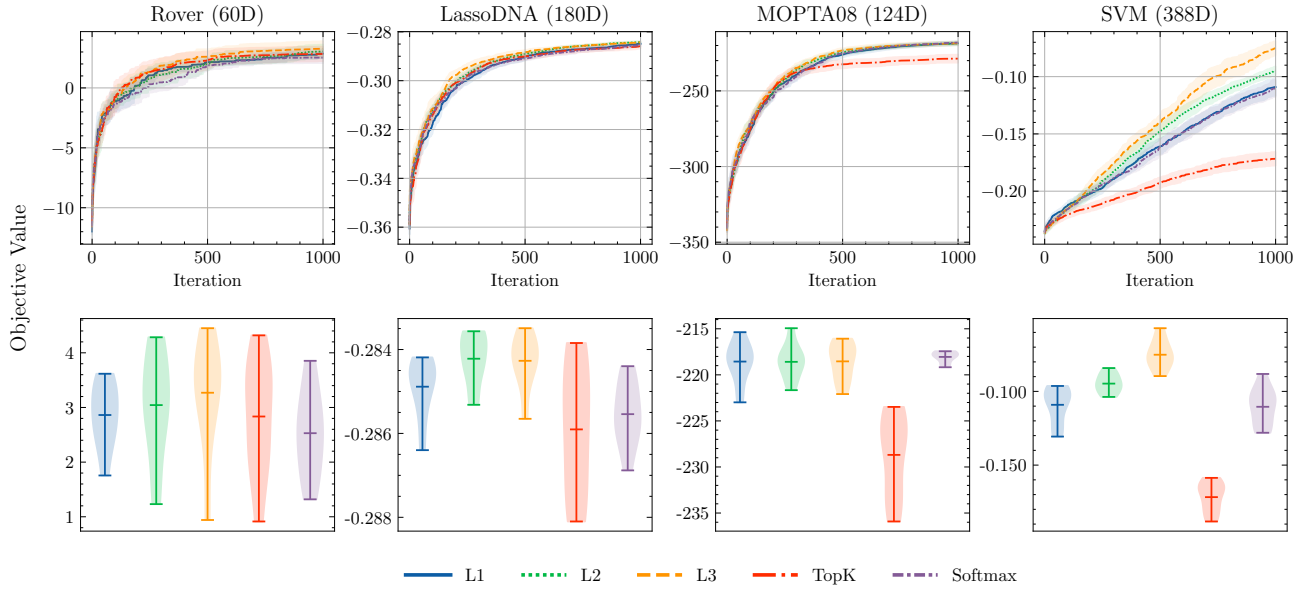

Figure 19: **Increased preference to large-magnitude gradient dimensions improves ACTS**. Except when using "TopK", the final performance remains competitive.

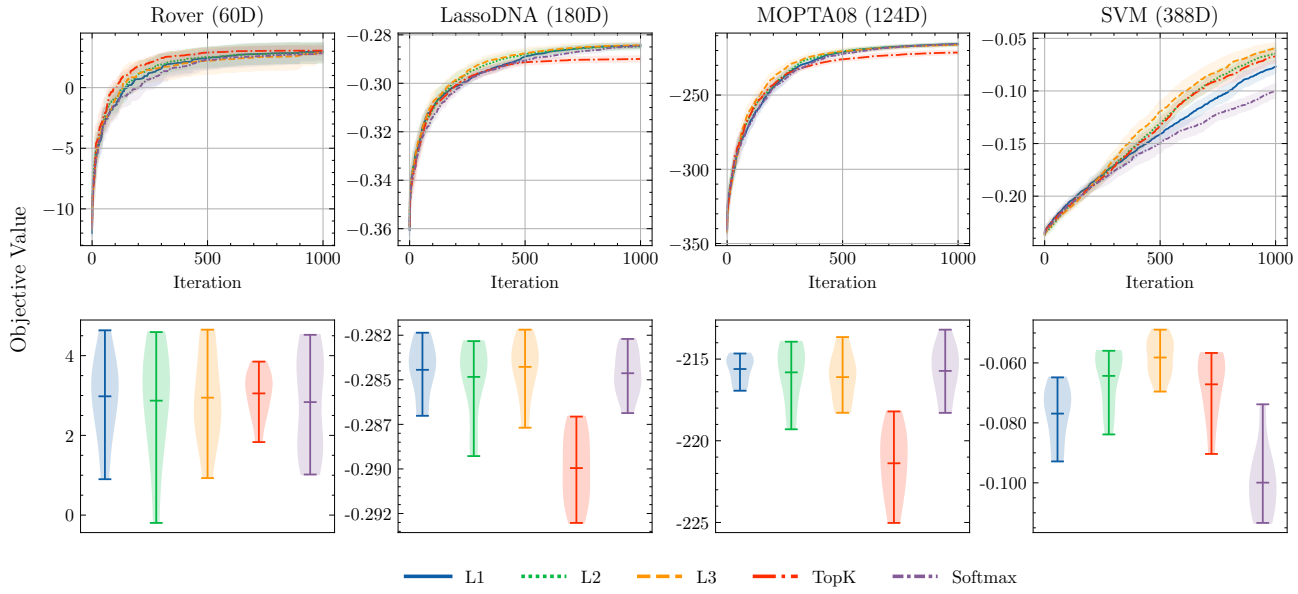

Figure 20: **Increased preference to large-magnitude gradient dimensions improves ACTS**. However, the effect is less pronounced when using TuRBO.

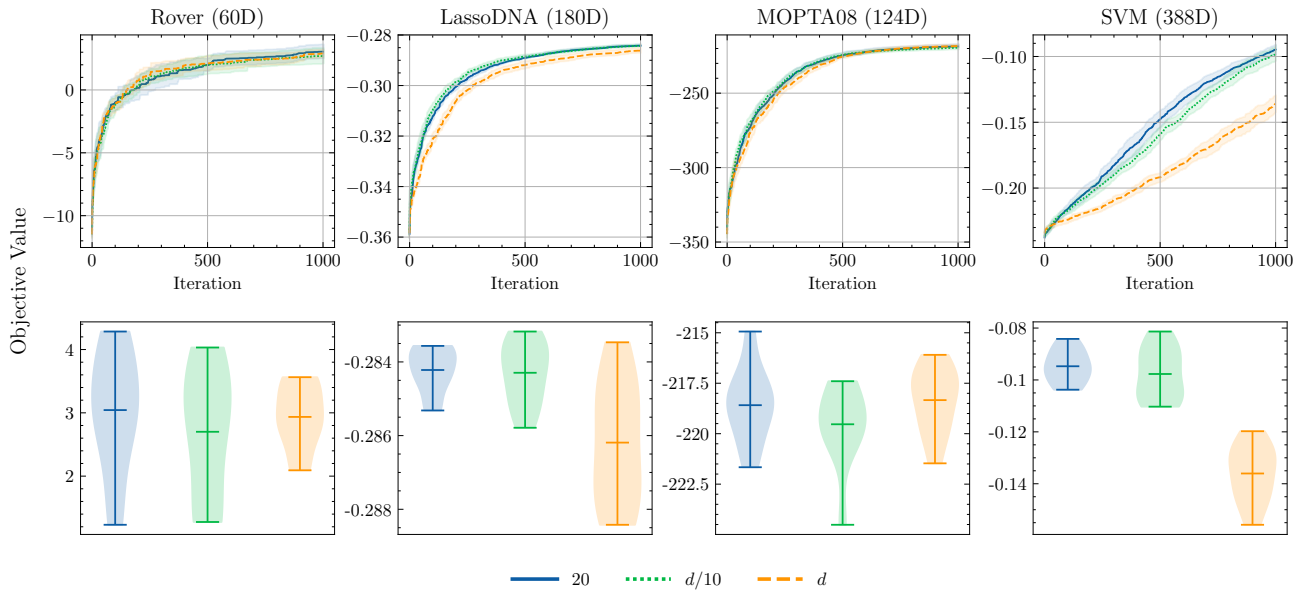

Figure 21: **Optimization performance when ablating on the effective number of dimensions perturbed by RAASP.**

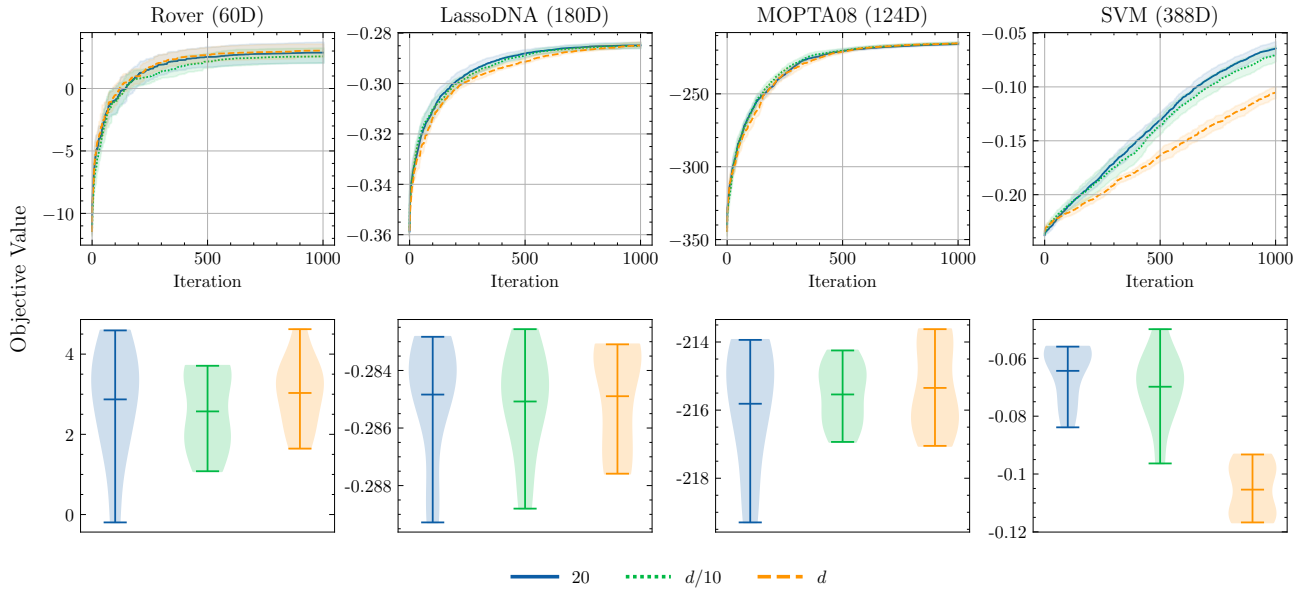

Figure 22: **Optimization performance when ablating on the effective number of dimensions perturbed by RAASP when using TuRBO.**