REVISITING VANILLA BAYESIAN OPTIMIZATION IN HIGH-DIMENSIONAL PERMUTATION SPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

High-dimensional Bayesian optimization (BO) has recently witnessed a paradigm shift. Accumulating evidence suggests that "Vanilla BO", i.e., standard Gaussian processes with robust kernel initialization and length-scale scaling, can achieve state-of-the-art performance in high-dimensional continuous tasks, casting doubt on the necessity of complex dimensionality reduction techniques. However, this narrative largely overlooks permutation spaces, a distinct and ubiquitous class of discrete combinatorial problems critical to industrial applications and scientific discovery. In this work, we revisit the efficacy of vanilla BO in high-dimensional permutation spaces under limited evaluation budgets. We propose VaPBO, a framework that performs variable selection over permutation elements, fits a Merge-kernel surrogate on active sub-permutations, and reconstructs full permutations through a structure-aware fill-in strategy. Across classical combinatorial problems, chip placement, and biodiversity history reconstruction, VaPBO with greedy local search achieves the best average rank among the compared methods. Further experiments show that surrogate guidance improves candidate selection and that fill-in preserves useful ordering from good solutions. Our findings call for a reevaluation of vanilla BO's generality and highlight the enduring value of dimensionality reduction for high-dimensional permutation optimization.

1 INTRODUCTION

Bayesian optimization (BO) has emerged as a principled framework for optimizing expensive black-box functions, particularly in scenarios where gradients are unavailable and function evaluations are costly (Shahriari et al., 2016; Frazier, 2018; Garnett, 2023). Classical BO with Gaussian processes (GPs) (Rasmussen & Williams, 2006) as surrogate models often struggles in high-dimensional spaces (Binois & Wycoff, 2022). To address this, the community has traditionally relied on structural assumptions, such as additive models (Kandasamy et al., 2015; Rolland et al., 2018; Mutný & Krause, 2018), low-dimensional embeddings (Wang et al., 2016; Letham et al., 2020; Papenmeier et al., 2022), and axis-aligned variable selection (Li et al., 2017; Eriksson & Jankowiak, 2021; Song et al., 2022), to mitigate the curse of dimensionality.

However, high-dimensional BO has recently witnessed a paradigm shift. Accumulating evidence suggests that "Vanilla BO", specifically standard GPs equipped with robust kernel initialization and dimension-aware length-scale scaling, can achieve state-of-the-art performance in high-dimensional continuous tasks (Hvarfner et al., 2024; Xu et al., 2025; Papenmeier et al., 2025b). This trend has cast doubt on the necessity of dimensionality reduction techniques, suggesting that the perceived failure of standard GPs can result from poor configuration rather than inherent modeling constraints.

While this narrative is compelling for continuous optimization, it largely overlooks permutation spaces, which represent a distinct and ubiquitous class of discrete combinatorial problems critical to industrial applications and scientific discovery (Jiao & Vert, 2015; Deshwal et al., 2022; Oh et al., 2022; Xie & Chen, 2025). Given n items, the size of a permutation space scales factorially as $n!$, and kernel similarity depends on discrete order structure. In this work, we revisit the efficacy of vanilla BO in high-dimensional permutation spaces under limited evaluation budgets. Our empirical analysis examines distance concentration and the difficulty of predicting high-quality permutations with full-space kernels. These observations motivate subspace modeling as a way to improve optimization in this setting.

Our findings highlight the value of dimensionality reduction for high-dimensional permutations. Specifically, we propose VaPBO, a framework that performs **va**riable selection over **p**ermutation

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

elements with **BO**. VaPBO randomly selects active items, fits a Merge-kernel GP on their induced sub-permutation, and optimizes the acquisition function over their positions. A structure-aware Best-K fill-in strategy reconstructs a full permutation by preserving the relative order of inactive items from a good historical solution. We validate the framework through an extensive empirical suite ranging from classical combinatorial problems, specifically the traveling salesman (Reinelt, 1991), quadratic assignment (Burkard et al., 1997), and traveling thief problems (Bonyadi et al., 2013), to challenging real-world applications, including industrial chip placement from electronic design automation (EDA) (Mirhoseini et al., 2021; Xue et al., 2025) and biodiversity history reconstruction from AI-for-Science (Fan et al., 2020; Tang et al., 2024). VaPBO with greedy local search achieves the best average rank on the high-dimensional tasks. Additional experiments show that GP guidance selects useful candidates and Best-K fill-in improves candidate quality, while the optimization gain from GP guidance varies across tasks.

Our contributions are summarized as follows:

- **Diagnosis**: We investigate distance concentration and predictive difficulties for full-space permutation kernels in high-dimensional spaces under limited budgets.
- **Methodology**: We propose VaPBO, a simple subspace BO framework that combines stochastic active-item selection, subspace Merge-kernel modeling, acquisition optimization, and Best-K completion.
- **Evaluation**: We introduce a high-dimensional permutation BO benchmark spanning standard combinatorial problems and two new real-world EDA / AI-for-Science tasks, together with low-dimensional controls, task-wise comparisons, and OTSD analyses. Matched-budget ablations examine the roles of local search, variable selection, and surrogate guidance.

2 Background

This section formalizes Bayesian optimization in permutation spaces, reviews permutation kernels, and positions our study relative to continuous high-dimensional BO and discrete BO baselines.

2.1 Bayesian Optimization in Permutation Spaces

Let $[n] = \{1, \dots, n\}$ denote n items. A permutation can be represented as a ranking vector $\pi \in \mathbb{S}_n$, where $\pi(i)$ is the position of item i. The search space is the symmetric group $\mathbb{S}_n$ with cardinality $|\mathbb{S}_n| = n!$. We consider expensive black-box functions over permutations $f : \mathbb{S}_n \to \mathbb{R}$.

Bayesian Optimization on $\mathbb{S}_n$. BO over permutations follows the standard BO loop, but replaces Euclidean kernels with permutation-aware similarities. Given observations $\mathcal{D}_t = \{(\pi_i, y_i)\}_{i=1}^t$ with $y_i = f(\pi_i) + \epsilon_i$, a GP surrogate $\mathcal{GP}(\mu_0, K(\pi, \pi'))$ is fitted to obtain a posterior mean $\mu_t(\pi)$ and uncertainty $\sigma_t(\pi)$. An acquisition function $\alpha_t(\pi)$ such as expected improvement (EI) (Mockus, 1994; Jones et al., 1998) is then maximized over $\mathbb{S}_n$ to propose the next evaluation point. Continuous relaxation (CR) provides a simple alternative by optimizing in a continuous box and mapping continuous vectors to permutations via sorting, but this representation can discard algebraic order structure.

- **Mallows kernel.** Mallows kernel adapts the RBF framework to the symmetric group via the Kendall-tau distance d_{KT} (Jiao & Vert, 2015; Deshwal et al., 2022). This distance counts discordant pairs between two rankings. It can be written through an implicit pairwise-comparison feature map $\Phi_{Mal}(\pi) \in \{0,1\}^{n(n-1)/2}$:

$$\begin{aligned} K_{\text{Mallows}}(\pi, \pi') &= \exp\big(-\lambda d_{\text{KT}}(\pi, \pi')\big) \\ &= \exp\left(-\frac{\|\Phi_{\text{Mal}}(\pi) - \Phi_{\text{Mal}}(\pi')\|_2^2}{2\ell^2}\right). \end{aligned}$$

  While mathematically robust, the Mallows kernel has $O(n^2)$ feature dimensionality, which becomes costly in high-dimensional regimes.

- **Merge kernel.** The Merge kernel (Xie & Chen, 2025) uses the fixed comparison path of merge sort to produce a structured representation $\Phi_{\mathrm{Mer}}(\pi) \in \{0,1\}^{\Theta(n \log n)}$. This compact representation is attractive for high-dimensional permutation BO and is the kernel backbone used in this work. Details are provided in the supplementary material.

2.2 Related Work

Continuous high-dimensional BO and the vanilla paradigm. High-dimensional BO traditionally relies on structural assumptions: additive models decompose the objective into lower-dimensional components (Kandasamy et al., 2015; Rolland et al., 2018; Mutný & Krause, 2018), embeddings restrict the search to a low-dimensional space (Wang et al., 2016; Letham et al., 2020; Papenmeier et al., 2022), and variable selection focuses on a subset of input dimensions (Li et al., 2017; Eriksson & Jankowiak, 2021; Song et al., 2022).

Recent continuous-domain results challenge the need for such reductions: standard GPs can perform strongly when their length scales and optimization are configured appropriately (Hvarfner et al., 2024; Xu et al., 2025). In particular, scaling length scales with $\sqrt{D}$, where D is the input dimension, helps avoid vanishing gradients during GP fitting. MSR (Papenmeier et al., 2025b) combines scaled initialization, maximum likelihood estimation, and random axis-aligned subspace perturbations, highlighting the role of local search in high-dimensional BO.

These findings motivate a further question: do the same improvements suffice in permutation spaces, where kernel similarity depends on discrete order and candidates must satisfy permutation constraints? We examine this question by comparing full-space models with subspace modeling and by separating surrogate guidance from local search.

Discrete and permutation BO. Discrete BO covers a broad range of spaces, including binary, categorical, sequence, graph, and permutation domains (González-Duque et al., 2024). Methods such as ComBO (Oh et al., 2019) exploit graph Cartesian products, while BOPS (Deshwal et al., 2022) and Merge-BO (Xie & Chen, 2025) specialize in permutation spaces. TuRBO (Eriksson et al., 2019) is a strong continuous local BO method and can be combined with continuous relaxation for permutation tasks. We include these discrete BO methods and a continuous-relaxation (CR) version of TuRBO in our comparisons. We focus on the geometry of high-dimensional permutations, subspace surrogate modeling, and attribution between local search and BO guidance in this setting.

3 Principle-Driven Approach for High-Dimensional Permutation Spaces

We first present an empirical diagnosis in Section 3.1, identifying two fundamental pathologies, i.e., geometric concentration of measure and topological ruggedness, which cause standard GP surrogates to fail as dimensionality increases. Motivated by these failures, Section 3.2 introduces VaPBO, a principle-driven framework that restores kernel informativeness. We detail how VaPBO leverages sub-permutation variable selection to restore variance and employs a structure-aware fill-in strategy to navigate the permutation spaces effectively in high dimensions.

3.1 Analyzing the Properties of High-Dimensional Permutation Spaces

Recent studies in continuous domains suggest that Vanilla BO can achieve state-of-the-art performance in high dimensions without subspace techniques. In this section, we investigate whether this "Vanilla Hypothesis" holds for the permutation space $\mathbb{S}_n$. Through the empirical analysis, we identify two fundamental pathologies, i.e., geometric concentration and topological ruggedness, that render global permutation kernels uninformative as n increases.

The Curse of Dimensionality: Geometric Collapse of the Permutation Space. Unlike continuous search spaces where the volume grows exponentially as c^d, the cardinality of the symmetric group $\mathbb{S}_n$ scales factorially ($n! \gg c^d$), creating a fundamentally more sparse and complex optimization landscape. In high-dimensional regimes, this complexity is further compounded by the

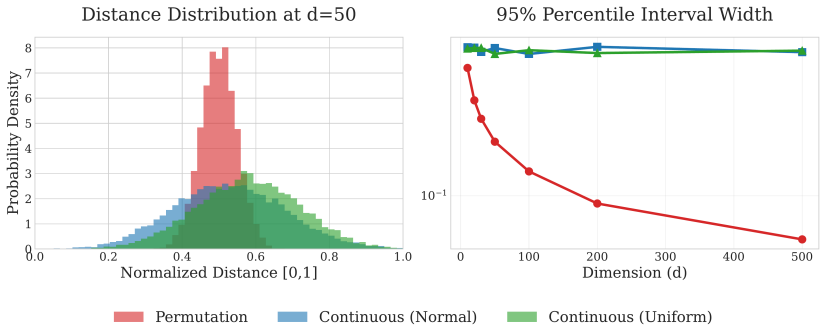

Figure 1: Empirical diagnosis of geometric collapse and concentration of measure in permutation spaces. (**Left**) At a representative dimension of $d = 50$, the normalized distance distribution between random pairs in $\mathbb{S}_n$ (red) is more concentrated around the mean than in Gaussian (blue) or Uniform (green) continuous spaces. This concentration phenomenon renders global kernels uninformative, as most candidates appear nearly equidistant to the observed data. (**Right**) The 95% percentile interval width of normalized distances exhibits a sharp, factorial-rate decay as dimensionality n increases. This "Geometric Collapse" indicates that standard kernels fail to maintain discriminative power in high dimensions, providing the fundamental motivation for the structural dimensionality reduction.

Concentration of Measure phenomenon, which dictates that the distances between randomly sampled permutations converge tightly around a central mean with vanishing variance as n increases. This phenomenon is empirically diagnosed in Figure 1. At a relatively low dimension of $d = 50$ (Left), the normalized distance distribution for permutations is more peaked than its continuous counterparts (Uniform or Gaussian), indicating that the vast majority of the space resides at a nearly constant distance from any reference point. More critically, as shown in the right sub-figure, the 95% percentile interval width of these distances exhibits a sharp, factorial-rate decay, which is a pathology we define as geometric collapse. This collapse implies that in high dimensions, any unobserved candidate π^* appears nearly equidistant to all previously observed points in the dataset.

Impact on GP Modeling From a modeling perspective, geometric collapse poses a catastrophic challenge for global permutation kernels. When distances concentrate, the resulting Gram matrix tends toward an identity or a constant matrix, effectively stripping the GP of its ability to distinguish between near and far regions. Consequently, the surrogate model fails to propagate information from the training data to unobserved regions, causing the posterior mean to remain flat and the acquisition function to become uninformative.

Topological Ruggedness and Surrogate Fragility. The issues of geometric collapse induce topological ruggedness, thereby posing a formidable challenge for GP regression when applying BO to high-dimensional permutation tasks. To evaluate model performance, we utilize Negative Log-Likelihood (NLL) as the primary metric, as it accounts for both predictive accuracy and uncertainty quantification (Deshwal et al., 2022; Papenmeier et al., 2025b), which is a crucial balance for effective exploration and exploitation in BO. To further comprehensively assess the surrogate models' capacity to capture the underlying structure of high-dimensional permutation spaces, we evaluate their performance on two distinct test sets: an In-Distribution (ID) set, mirroring the training data distribution, and an Out-of-Distribution (OOD) set comprising points with superior objective quality. This design, widely used in offline model-based optimization (Kim et al., 2025; Trabucco et al., 2022), measures predictive quality on unseen, high-performing candidates.

Table 1 shows that CR has the lowest ID NLL but the highest OOD NLL on both application tasks. Mallows and Merge reduce, but do not eliminate, this OOD error. The low ID-NLL of CR is a deceptive indicator of performance, as it originates from overfitting to structured noise rather than capturing the underlying combinatorial topology. In contrast, permutation-aware Mallows and Merge kernels exhibit superior OOD robustness. They provide a compact and structure-aware representation, effectively avoiding the structural inconsistencies of continuous relaxation and achieving superior optimization performance. Despite these improvements, the high NLL values across all methods indicate the difficulty of fitting accurate surrogates for high-dimensional permutations.

Metric	CR	Mallows	Merge
`Chip Placement` (ID)	0.72	1.52	1.51
`Biodiversity` (ID)	0.54	1.51	1.51
`Chip Placement` (OOD)	1015.7	221.78	840.08
`Biodiversity` (OOD)	1251.32	854.67	1035.9

Table 1: Comparison of Negative Log-Likelihood (NLL) results for different methods after 200 iterations. Lower values indicate better model calibration and uncertainty estimation. ID denotes the results for the in-distribution test set, while OOD represents the results for out-of-distribution points with better actual quality.

Kernel Selection in High-Dimensional Permutation Spaces. In high-dimensional permutation spaces, the choice of kernel is governed by a critical trade-off between the ability to capture global structural invariance and the search efficiency driven by the compactness of the feature space. While the Mallows (Jiao & Vert, 2015; Deshwal et al., 2022) kernels offer robust distance metrics, their $O(n^2)$ feature dimensionality leads to computational and statistical redundancy as the number of objects n grows, severely limiting their sample efficiency in high-dimensional regimes. In this work, the recently proposed Merge kernel (Xie & Chen, 2025) is chosen because it leverages the divide-and-conquer logic of merge sort to produce a structured representation that matches the information-theoretic lower bound of $\Theta(n \log n)$ for lossless permutation encoding (Knuth, 1998). The construction of the Merge kernel provides a scalable approach and outperforms both Mallows kernels and continuous relaxation methods in both final regret and convergence speed.

Right-Invariance of Permutation Kernels. A kernel K on $\mathbb{S}_n$ is *right-invariant* if $K(\pi, \pi') = K(\pi \circ \sigma, \pi' \circ \sigma)$ for any $\sigma \in \mathbb{S}_n$. This property preserves kernel similarity when the same item relabeling is applied to both rankings. The Mallows kernel satisfies this property (Jiao & Vert, 2015), whereas the Merge kernel uses a fixed merge-sort comparison path and is not right-invariant (Xie & Chen, 2025). We use Merge for its compact representation of relative-order comparisons. In VaPBO, this representation models the active items, while Best-K fill-in constructs full permutations from the selected active-item positions.

3.2 Restoring Informative GP via Dimensionality Reduction

As illustrated in the right sub-figure of Figure 1, the changing rate of concentration in permutation spaces is steeper than that in continuous spaces, suggesting that dimensionality reduction is potentially more effective for permutations. To mitigate the pathologies identified in Section 3.1, i.e., the concentration of distance measures and topological ruggedness, we propose VaPBO (Variable Selection of Permutation with Bayesian Optimization). We argue that applying recent vanilla techniques directly to the full permutation space yields limited improvements, as they cannot fundamentally counteract the intrinsic concentration of measure. Therefore, to restore the discriminative power of the GP, it is imperative to actively project the high-dimensional $\mathbb{S}_n$ onto a lower-dimensional subspace where the kernel's distance distribution regains sufficient variance.

However, performing such a reduction on permutation structures is non-trivial and necessitates specialized architectural designs. Unlike continuous domains where axes are independent, dimensionality reduction in permutation spaces is non-trivial due to the strict combinatorial constraints (i.e., the bijectivity of the permutation). VaPBO addresses this via a dropout-style variable selection approach (Li et al., 2017) tailored for $\mathbb{S}_n$, integrated with a Structure-Aware Fill-in strategy to handle inactive dimensions. Detailed algorithm flow of VaPBO is provided in the supplementary material.

Permutation Subspace Definition. We define the variable selection mechanism on the ranking representation. Let $\pi \in \mathbb{S}_n$ be the permutation vector where $\pi(i)$ denotes the position of item i. In each BO iteration, we randomly select a subset of active items $\mathcal{I} \subset \{1, \ldots, n\}$ with cardinality $|\mathcal{I}| = n_a < n$. The optimization is then performed on the subspace configuration $\mathbf{z} \in \{1, \ldots, n\}^{n_a}$, which represents the positions of the active items:

$$\mathbf{z} = \{\pi(i) \mid i \in \mathcal{I}\}.$$

Crucially, while standard variable selection in Euclidean space fixes inactive dimensions to a seen value, fixing variables in $\mathbb{S}_n$ is constrained by the mutual exclusion of ranks. The positions assigned

to $\mathcal{I}$ dictate the available slots for the inactive set $\mathcal{I}^c$, creating a dependency that must be managed by the fill-in strategy.

Subspace Modeling with Merge Kernel. To enable sample-efficient learning, we construct the surrogate model strictly over the active subspace. For any two permutations π, π', their similarity is computed based solely on the active items $\mathcal{I}$. We use the Merge kernel on the active items:

$$K_{sub}(\pi, \pi'; \mathcal{I}) = K_{\text{Merge}}(\Phi(\pi)_{\mathcal{I}}, \Phi(\pi')_{\mathcal{I}}),$$

where $\Phi(\cdot)_{\mathcal{I}}$ extracts the merge-sort comparison features relevant only to the items in $\mathcal{I}$. By restricting the kernel to n_a dimensions, we artificially inject variance into the distance distribution, preventing the Gram matrix from collapsing to the identity, which is a necessary condition for effective posterior inference in high-dimensional settings.

Structure-Aware Fill-in Strategies. A core challenge in dimensionality reduction-based permutation BO is determining the configuration of the *inactive items* $\mathcal{I}^c$. Since the objective function $f(\pi)$ requires a full permutation for evaluation, we employ a fill-in strategy Ψ that maps a subspace configuration $\mathbf{z}$ to a valid full permutation $\pi = \Psi(\mathbf{z}; \mathcal{H}_t)$, potentially utilizing history $\mathcal{H}_t$. A naive approach is to assign the items in $\mathcal{I}^c$ to the remaining available positions uniformly at random. While this ensures unbiased exploration, it disregards the high degree of conditional dependence between variables. To combat landscape ruggedness and exploit known high-quality substructures, we introduce a Best-k Position fill-in strategy. We maintain a set of the top-k best solutions found so far, denoted as $\mathcal{P}_k^* = \{\pi_1^*, \ldots, \pi_k^*\}$. For a candidate subspace configuration $\mathbf{z}$ (defining positions for items $\mathcal{I}$), we select a reference template $\pi_{ref} \in \mathcal{P}_k^*$ that minimizes active-item positional distance:

$$\pi_{ref} = \arg\min_{\pi \in \mathcal{P}_k^*} \sum_{i \in \mathcal{I}} (\pi(i) - \mathbf{z}_i)^2.$$

The full permutation is constructed by fixing the active items according to $\mathbf{z}$ and filling the remaining positions using the relative ordering of $\mathcal{I}^c$ found in π_{ref}. This acts as a data-driven soft trust region, stabilizing the optimization by ensuring that unoptimized dimensions adhere to structures known to yield low objective values.

Acquisition Optimization. Optimizing $\alpha(\pi)$ over a reduced but still combinatorial subspace remains challenging. We compare acquisition and search operators under matched evaluation settings.

- **Local Search (LS)** explores the neighborhood of a candidate through swap operations. In VaPBO-LS, each neighbor is evaluated by projecting it onto the active subspace $\mathcal{I}$ and querying the subspace GP acquisition function.
- **Greedy Local Search (GLS)** allocates more neighborhood-search effort around the current incumbent rather than relying mainly on repeated random restarts. This stronger exploitative operator is especially useful in rugged permutation landscapes. Our main variant, VaPBO-GLS, combines GLS with the VaPBO subspace surrogate.
- **Evolutionary Algorithm (EA)** maintains a population of sub-permutations and uses structured recombination to explore disjoint acquisition regions.
- **Gradient-Based Optimization (Grad)** uses gradients in the Merge feature space as a proxy search direction. We also compare EI with LogEI in the supplementary material.

For attribution, we distinguish **VaP** variants, which use variable selection and fill-in without GP acquisition guidance, from **VaPBO** variants, which rank candidates using the subspace GP acquisition function. This separates the effects of variable selection and local search from GP guidance.

4 Experiments

We evaluate whether subspace modeling improves high-dimensional permutation optimization and examine the contributions of variable selection, surrogate guidance, and local search.

4.1 SETTINGS AND TASKS

Compared methods. The baselines include Random Search, which samples permutations uniformly; Vanilla BO (CR) and TuRBO (CR) (Eriksson et al., 2019), which use continuous relaxation; and the permutation methods BOPS (Deshwal et al., 2022), Merge-BO (Xie & Chen, 2025), and Vanilla Merge-BO (Xu et al., 2025; Papenmeier et al., 2025b).

LS and GLS use the same neighborhood-search families as our acquisition optimizers. VaP-LS and VaP-GLS add variable selection and Best-K fill-in but omit GP guidance. VaPBO-LS, VaPBO-GLS, VaPBO-EA, and VaPBO-Grad combine these components with a GP and differ in acquisition optimization. Unless otherwise stated, $n_a/n = 50\%$ and $K = 3$ throughout the experiments.

Evaluation settings. All methods use matched initialization and black-box evaluation budgets within each task. Initial designs contain five random candidates for combinatorial tasks and ten for the applications. We report comparisons at 50 evaluations for TSP, QAP, and TTP, and 80 for chip placement and biodiversity reconstruction; Figure 3 compares all high-dimensional tasks at 50 evaluations. Results are mean $\pm$ standard deviation over five independent seeds.

Implementation. The Merge and Vanilla Merge baselines build on the released implementations of Xie & Chen (2025) and Xu et al. (2025); code links are provided in Appendix B. Our VaPBO implementation uses BoTorch (Balandat et al., 2020). GP hyperparameters are fitted through the marginal log-likelihood objective, with the specified priors retained for Vanilla Merge. VaPBO uses EI, and Vanilla Merge uses LogEI. Merge-BO uses EI on TSP, QAP, and the applications; its TTP implementation uses the upper confidence bound (UCB) acquisition function. Appendix B.4 compares EI and LogEI for VaPBO-Grad.

Tasks. We evaluate on combinatorial benchmarks (Deshwal et al., 2022; Xie & Chen, 2025) and two new permutation BO applications, with dimensions up to 318.

Combinatorial optimization. The traveling salesperson problem (TSP) minimizes the length of a tour visiting every city once (Reinelt, 1991). The quadratic assignment problem (QAP) assigns facilities to locations to minimize the sum of flow–distance products (Burkard et al., 1997). The traveling thief problem (TTP) couples a city tour with item selection: collected items increase the carried weight and reduce travel speed, and the objective maximizes profit minus travel-time cost (Bonyadi et al., 2013). These tasks test ordering decisions with different forms of interaction.

Chip placement. We use chip placement from EDA (Mirhoseini et al., 2021; Xue et al., 2025), represented by a Sequence Pair (Murata et al., 1996): two permutations of the macros determine their horizontal and vertical ordering. A decoder produces macro coordinates, and the objective is half-perimeter wirelength (HPWL). This gives 256 permutation variables for 128 macros.

Biodiversity reconstruction. We order species' first and last appearance events to reconstruct a global timeline from incomplete local fossil records (Fan et al., 2020; Tang et al., 2024). The CONOP evaluator repairs ordering and coexistence constraints, then scores disagreement with the observed geological sections. We minimize this penalty on tasks with 124 and 278 events.

4.2 MAIN RESULTS

Is full-space vanilla BO sufficient? Table 2 and Figure 2 show that VaPBO-GLS achieves the best average rank, ahead of full-space vanilla BO and TuRBO (CR). Vanilla Merge gives only a small average-rank improvement over Merge: applying the vanilla recipe to the Merge kernel does not close the gap to the strongest methods. TuRBO is a stronger baseline, but its local search still trails VaPBO-GLS in aggregate. These results favor subspace search in the high-dimensional, limited-budget setting studied here. Figure 3 tracks ranks over a common 50-evaluation budget; Appendix B.1 provides the task-wise convergence curves (Figure 6).

Where do the gains come from? Local search alone performs poorly in the high-dimensional comparison. Adding variable selection and Best-K fill-in improves GLS's average rank from 10.62 to 3.50, placing VaP-GLS ahead of every full-space BO baseline without GP guidance. The same components bring little average-rank improvement from LS to VaP-LS, indicating that their effec-

Method	TSP127	TSP318	QAP256	TTP280	CP1	CP2	Bio1	Bio2	Avg. rank
Random	8.36±2.50	4.88±1.30	632.43±51.01	291.09±0.00	55.80±7.99	37.58±10.72	1.27±0.15	0.18±0.02	9.88
LS	9.36±3.27	6.71±2.45	677.75±35.28	290.95±5.95	79.02±17.29	44.49±11.49	1.34±0.20	0.19±0.03	12.62
GLS	6.27±3.89	7.47±2.11	672.70±46.58	284.80±6.29	63.65±24.31	29.12±9.74	1.35±0.35	0.18±0.06	10.62
Vanilla BO (CR)	8.53±2.05	4.89±1.79	671.49±43.25	108.22±12.58	**23.04±13.44**	18.74±8.89	1.04±0.29	0.15±0.03	6.62
TuRBO (CR)	5.73±2.93	3.00±1.11	584.88±48.84	286.44±4.20	23.22±10.65	9.29±4.71	1.02±0.21	0.15±0.03	4.62
BOPS	7.39±2.90	4.69±0.00	634.93±51.81	289.65±6.21	59.79±16.29	28.75±6.78	0.83±0.12	0.22±0.01	8.88
Merge	7.72±2.67	6.27±1.00	666.78±38.21	247.33±4.77	56.74±10.50	26.95±4.58	1.22±0.17	0.17±0.03	9.38
Vanilla Merge	7.19±2.25	5.22±1.88	678.21±34.49	174.35±141.78	52.74±7.72	25.43±3.79	1.13±0.27	0.19±0.02	9.12
VaP-LS	7.30±2.16	5.93±1.07	710.54±33.62	292.58±4.51	85.80±15.30	47.49±9.35	1.43±0.18	0.17±0.02	12.25
VaP-GLS	4.75±3.11	3.46±0.96	582.57±34.42	276.71±5.21	26.42±15.70	22.34±11.21	0.54±0.23	**0.07±0.05**	3.50
VaPBO-Grad	5.25±1.69	**1.08±0.92**	646.96±17.00	127.25±8.74	54.22±10.57	31.26±6.02	1.10±0.18	0.14±0.05	5.75
VaPBO-EA	5.46±2.64	3.99±1.26	675.94±20.47	86.63±34.54	34.83±10.48	18.20±11.98	1.00±0.08	0.16±0.01	5.38
VaPBO-LS	6.53±1.94	3.93±1.38	647.01±55.04	99.77±12.30	41.89±21.02	17.69±8.07	0.75±0.14	0.13±0.03	4.50
VaPBO-GLS	**3.90±1.87**	1.64±1.31	**579.68±24.42**	**64.40±7.57**	50.64±28.78	**7.30±6.86**	**0.43±0.36**	0.11±0.07	**1.88**

Table 2: High-dimensional regret results. Each entry is mean ± standard deviation over five independent runs. The best mean in each column is bold.

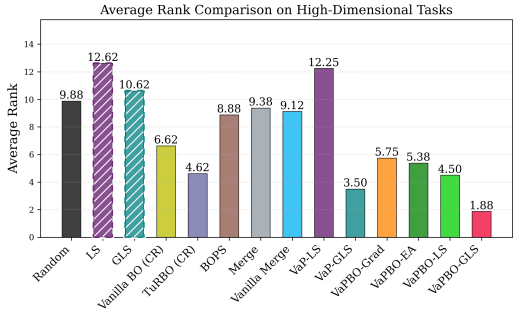

Figure 2: Average rank comparison on high-dimensional tasks, including full-space BO baselines and local-search variants. Lower is better. VaPBO-GLS ranks first, followed by the variant without GP guidance, VaP-GLS.

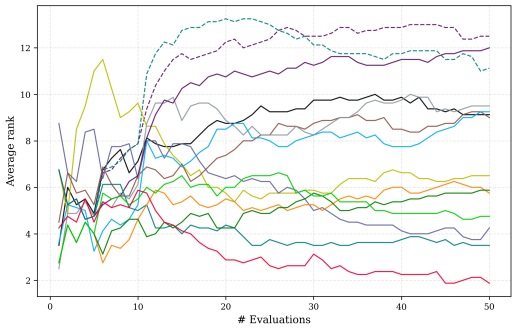

Figure 3: Average rank across the high-dimensional tasks as the evaluation budget increases from 1 to 50. Lower is better. Colors match Figure 2; dashed lines denote LS and GLS.

tiveness depends on the search operator. GP guidance further improves the average rank to 1.88; among VaPBO optimizers, GLS also ranks ahead of LS, EA, and Grad.

Performance across tasks. VaPBO-GLS obtains the best mean on five of the eight tasks. Other methods remain stronger on individual tasks: VaPBO-Grad on TSP318, Vanilla BO (CR) on CP1, and VaP-GLS on the second biodiversity task. To examine the GP benefit further, we compare VaPBO-GLS, VaP-GLS with Best-K or random fill-in, and simulated annealing on TSP318 and QAP256 using ten additional paired seeds and 50 evaluations, including five shared initial permutations. GP guidance gives a consistent benefit on TSP318, but a smaller and less consistent gain on QAP256. Final-cost differences between Best-K and random fill-in are small relative to variation across runs. The settings and curves are in Appendix B.1 (Figure 7).

Surrogate calibration. Table 3 shows that VaPBO substantially lowers OOD NLL on both application tasks, while its ID NLL remains close to that of Merge. Compared with the full-space results in Table 1, subspace modeling improves prediction on high-quality permutations.

Low-dimensional comparisons. We compare methods on TSP10, QAP15, Floor Planning30, and Cell Placement30. VaP-GLS has the best average rank, while BOPS and VaPBO-GLS tie, showing that full-space BOPS remains competitive and GP guidance adds little here. Appendix B.2 provides the settings and detailed results (Table 4 and Figure 8).

Sensitivity analysis. We vary $n_a/n \in \{25\%, 50\%, 75\%\}$ on Chip Placement 2 and Biodiversity 2 over 80 evaluations. The first task favors 50% and the second favors 25%; both outperform 75% (Figure 5). We also analyze Best-K fill-in with $K \in \{1, 3, 5\}$ across the combinatorial and application tasks. Convergence trends are similar on most tasks, with $K = 3$ providing a useful default. Appendix B.3 gives the settings and Best-K curves (Figure 9).

Metric	Merge	Vanilla Merge	VaPBO
`Chip Placement` (ID)	1.51	52.45	**1.30**
`Biodiversity` (ID)	1.51	77.21	**1.30**
`Chip Placement` (OOD)	840.08	40.08	**5.06**
`Biodiversity` (OOD)	1035.9	52.87	**11.03**

Table 3: Reduced-space NLL results on the application tasks. Lower values indicate better calibration. VaPBO improves OOD calibration compared with the full-space models in Table 1.

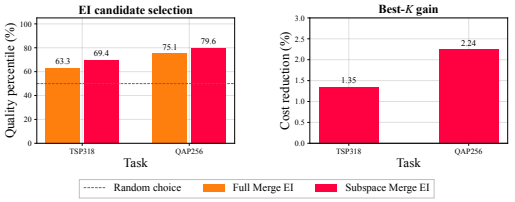

Figure 4: Mean EI selection quality (left) and Best-K completion gain (right) over ten histories on TSP318 and QAP256.

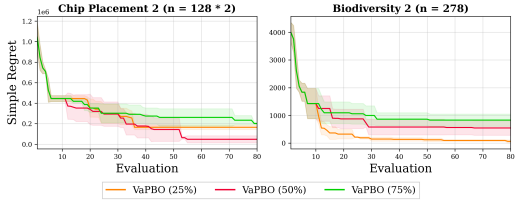

Figure 5: Active-ratio sensitivity on Chip Placement 2 and Biodiversity 2. Curves compare three active ratios under the same evaluation budget.

Acquisition optimization and search behavior. We compare normalized OTSD across the VaPBO optimizers and VaP-GLS, using Random Search as a reference. VaP-GLS searches most locally, followed by VaPBO-GLS; the other variants remain closer to Random Search. We also replace EI with LogEI in VaPBO-Grad across low- and high-dimensional tasks. LogEI improves final regret on TSP127 and QAP15, whereas EI is better on TSP318 and QAP256. Appendix B.4 provides the OTSD definition and full trajectories (Figure 10), together with the EI/LogEI curves (Figure 11).

Surrogate guidance. On TSP318 and QAP256, we fit full-space and subspace Merge GPs using ten histories of 20 observations each, then let EI select from the same pool of 64 candidates. Both models select better candidates than uniform random choice, with subspace EI attaining a higher mean quality percentile (Figure 4, left). This measures candidate-selection quality; details and numerical results are in Appendix B.5.

Structure-aware fill-in. We use ten fresh histories on the same tasks and fix 64 active-item assignments per history. Completing each assignment with Best-K ($K = 3$) reduces mean candidate cost relative to random inactive-item ordering by 1.35% on TSP318 and 2.24% on QAP256 (Figure 4, right). Preserving the inactive-item order of good historical solutions thus improves the quality of completed candidates. Appendix B.5 provides the construction and numerical results of this paired comparison.

5 Conclusion

We revisited vanilla BO in high-dimensional permutation spaces and found that conclusions from continuous domains do not transfer automatically. VaPBO combines stochastic subspaces, Merge-kernel modeling, and structure-aware completion. Across fourteen baselines and ablations, VaPBO-GLS obtains the best average rank on the high-dimensional tasks. Ablations show that variable selection and fill-in strengthen GLS, while GP guidance further improves the average rank and selects higher-quality candidates. Overall, high-dimensional permutation BO benefits from combining subspace variable selection, surrogate-guided candidate ranking, and strong local search. Future work may explore dependence-aware active-set selection. We also plan broader evaluations across permutation tasks and budgets.

AI USE STATEMENT

We used generative AI tools to assist with code implementation and improve the readability of the manuscript. We reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, and code.

REPRODUCIBILITY STATEMENT

The source code is provided in the supplementary materials.

REFERENCES

Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. In *Advances in Neural Information Processing Systems 33*, 2020.

Mickaël Binois and Nathan Wycoff. A survey on high-dimensional Gaussian process modeling with application to Bayesian optimization. *ACM Transactions on Evolutionary Learning and Optimization*, 2(2):1–26, 2022.

Mohammad Reza Bonyadi, Zbigniew Michalewicz, and Luigi Barone. The travelling thief problem: The first step in the transition from theoretical problems to realistic problems. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, 2013.

Rainer E. Burkard, Stefan E. Karisch, and Franz Rendl. QAPLIB - A quadratic assignment problem library. *Journal of Global Optimization*, 10(4):391–403, 1997.

Aryan Deshwal, Syrine Belakaria, Janardhan Rao Doppa, and Dae Hyun Kim. Bayesian optimization over permutation spaces. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 2022.

David Eriksson and Martin Jankowiak. High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, 2021.

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

Jun-xuan Fan, Shu-zhong Shen, Douglas H Erwin, Peter M Sadler, Norman MacLeod, Qiu-ming Cheng, Xu-dong Hou, Jiao Yang, Xiang-dong Wang, Yue Wang, et al. A high-resolution summary of cambrian to early triassic marine invertebrate biodiversity. *Science*, 367(6475):272–277, 2020.

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

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

Miguel González-Duque, Richard Michael, Simon Bartels, Yevgen Zainchkovskyy, Søren Hauberg, and Wouter Boomsma. A survey and benchmark of high-dimensional Bayesian optimization of discrete sequences. In *Advances in Neural Information Processing Systems 37*, 2024.

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*, 2024.

Yunlong Jiao and Jean-Philippe Vert. The kendall and mallows kernels for permutations. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

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

Kirthevasan Kandasamy, Jeff G. Schneider, and Barnabás Póczos. High dimensional Bayesian optimisation and bandits via additive models. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

Minsu Kim, Jiayao Gu, Ye Yuan, Taeyoung Yun, Zixuan Liu, Yoshua Bengio, and Can Chen. Offline model-based optimization: Comprehensive review. *arXiv:2503.17286*, 2025.

Donald E. Knuth. *The Art of Computer Programming: Sorting and Searching, Volume 3*. Addison-Wesley Professional, 1998.

Benjamin Letham, Roberto Calandra, Akshara Rai, and Eytan Bakshy. Re-examining linear embeddings for high-dimensional Bayesian optimization. In *Advances in Neural Information Processing Systems 33*, 2020.

Cheng Li, Sunil Gupta, Santu Rana, Vu Nguyen, Svetha Venkatesh, and Alistair Shilton. High dimensional Bayesian optimization using dropout. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017.

Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim M. Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, et al. A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212, 2021.

Jonas Mockus. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 4(4):347–365, 1994.

Hiroshi Murata, Kunihiro Fujiyoshi, Shigetoshi Nakatake, and Yoji Kajitani. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(12):1518–1524, 1996.

Mojmír Mutný and Andreas Krause. Efficient high dimensional Bayesian optimization with additivity and quadrature Fourier features. In *Advances in Neural Information Processing Systems 31*, 2018.

ChangYong Oh, Jakub M. Tomczak, Efstratios Gavves, and Max Welling. Combinatorial Bayesian optimization using the graph cartesian product. In *Advances in Neural Information Processing Systems 32*, 2019.

ChangYong Oh, Roberto Bondesan, Efstratios Gavves, and Max Welling. Batch Bayesian optimization on permutations using the acquisition weighted kernel. In *Advances in Neural Information Processing Systems 35*, 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 35*, 2022.

Leonard Papenmeier, Nuojin Cheng, Stephen Becker, and Luigi Nardi. Exploring exploration in Bayesian optimization. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2025a.

Leonard Papenmeier, Matthias Poloczek, and Luigi Nardi. Understanding high-dimensional Bayesian optimization. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025b.

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

Gerhard Reinelt. TSPLIB - A traveling salesman problem library. *ORSA Journal on Computing*, 3 (4):376–384, 1991.

Paul Rolland, Jonathan Scarlett, Ilija Bogunovic, and Volkan Cevher. High-dimensional Bayesian optimization via additive models with overlapping groups. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, 2018.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2016.

Lei Song, Ke Xue, Xiaobin Huang, and Chao Qian. Monte Carlo tree search based variable selection for high dimensional Bayesian optimization. In *Advances in Neural Information Processing Systems 35*, 2022.

Qing Tang, Wentao Zheng, Shuhan Zhang, Junxuan Fan, Leigh Anne Riedman, Xudong Hou, A. D. Muscente, Natalia Bykova, Peter M Sadler, Xiangdong Wang, et al. Quantifying the global biodiversity of proterozoic eukaryotes. *Science*, 386(6728):eadm9137, 2024.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-Bench: Benchmarks for data-driven offline model-based optimization. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.

Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando de Freitas. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55(1):361–387, 2016.

Zikai Xie and Linjiang Chen. From sorting algorithms to scalable kernels: Bayesian optimization in high-dimensional permutation spaces. *arXiv:2507.13263*, 2025.

Zhitong Xu, Haitao Wang, Jeff M. Phillips, and Shandian Zhe. Standard Gaussian process is all you need for high-dimensional Bayesian optimization. In *Proceedings of the 13th International Conference on Learning Representations*, 2025.

Ke Xue, Ruo-Tong Chen, Rong-Xi Tan, Xi Lin, Yunqi Shi, Siyuan Xu, Mingxuan Yuan, and Chao Qian. Bboplace-bench: Benchmarking black-box optimization for chip placement. *arXiv:2510.23472*, 2025.

Martin Zaefferer, Jörg Stork, and Thomas Bartz-Beielstein. Distance measures for permutations in combinatorial efficient global optimization. In *Proceedings of the International Conference on Parallel Problem Solving from Nature*, 2014.

A METHOD DETAILS

A.1 DISTANCES ON $\mathbb{S}_n$

Modeling permutations requires a suitable distance or similarity measure. Following standard practice in permutation BO (Deshwal et al., 2022; Oh et al., 2022; Xie & Chen, 2025), we consider the following two common distances:

(i) Kendall distance counts the number of discordant pairs

$$d_{\mathrm{KT}}(\pi, \pi') = n_d(\pi, \pi') = \sum_{1 \le i < j \le n} \mathbb{I}\left[\begin{matrix}(\pi(i) - \pi(j))(\pi'(i) - \pi'(j)) \\ < 0\end{matrix}\right].$$

(ii) Position-based distances measure absolute position discrepancies, e.g., the Spearman footrule

$$d_{\mathrm{pos}}(\pi, \pi') = \sum_{i=1}^{n} \big|\pi(i) - \pi'(i)\big|.$$

These distances preserve different invariances and impose different inductive biases; see Jiao & Vert (2015); Zaefferer et al. (2014) for a systematic discussion.

A.2 MERGE KERNEL DETAILS (XIE & CHEN, 2025)

VaPBO uses the Merge kernel for GP modeling. We summarize the merge-sort feature map $\Phi_{\mathrm{Mer}}(\cdot)$ and the fixed-length construction used for kernel evaluation in Algorithm 1.

Fixed-length comparison features. A comparison-based sorting procedure maps a permutation $\pi \in \mathbb{S}_n$ to a sequence of binary comparison outcomes. These outcomes encode the relative-order information used during sorting. For kernel evaluation, each feature coordinate must have the same meaning for every input, and all feature vectors must have the same length.

The Merge kernel stabilizes merge sort by recording $L + R - 1$ outcomes when merging sublists of lengths L and R. After one sublist is exhausted, redundant outcomes fill the remaining coordinates. Recursively concatenating these outcomes gives the deterministic fixed-length map $\Phi_{\mathrm{Mer}}(\pi)$ with $\Theta(n \log n)$ coordinates, matching the asymptotic lower bound for lossless encoding of permutations.

Algorithm 1 Merge Feature Mapping with Fixed Comparison Path

Function MERGEFEATURE(π, n):
Input: Permutation $\pi \in \mathbb{S}_n$, dimension n
Output: Binary feature vector V
if $n = 1$ **then**
  Return: [] // Base case: single element
end if
if $n = 2$ **then**
  Return: $[\mathbb{I}(\pi[0] > \pi[1])]$ // Base case: single comparison
end if
$mid \leftarrow \lfloor n/2 \rfloor$
$V_L \leftarrow$ MergeFeature($\pi[0 : mid], mid$) // Recursive split
$V_R \leftarrow$ MergeFeature($\pi[mid : n], n - mid$)
// Track merge path using sorted sub-sequences
$\hat{\pi}_l \leftarrow$ Sort($\pi[0 : mid]$), $\hat{\pi}_r \leftarrow$ Sort($\pi[mid : n]$)
$V_{\text{Merge}} \leftarrow []$, $i \leftarrow 0, j \leftarrow 0$
while $i <$ length($\hat{\pi}_l$) **and** $j <$ length($\hat{\pi}_r$) **do**
  if $\hat{\pi}_l[i] > \hat{\pi}_r[j]$ **then**
    V_{Merge}.append(1); $j \leftarrow j + 1$
  else
    V_{Merge}.append(0); $i \leftarrow i + 1$
  end if
end while
Append redundant outcomes to V_{Merge} until its length is $n - 1$
Return: $V_L \oplus V_R \oplus V_{\text{Merge}}$ // Concatenate features

A.3 VaPBO Algorithm Details

Algorithm 2 gives the VaPBO procedure used in the main paper.

Stage 1: Random active-item selection. At each BO iteration, VaPBO samples an active set $\mathcal{I}_t \subset [n]$ uniformly at random, with $|\mathcal{I}_t| = n_a < n$. The active set is a stochastic projection that reduces the dimension seen by the surrogate; it does not assume that permutation items are independent.

Stage 2: Subspace projection and modeling. Write $\mathcal{I}_t = \{i_{t,1}, \ldots, i_{t,n_a}\}$ in a fixed order and define $\mathbf{z}_{\mathcal{I}_t}(\pi) = (\pi(i_{t,1}), \ldots, \pi(i_{t,n_a}))$. Valid candidates belong to $\mathcal{Z}_{\mathcal{I}_t} = \{\mathbf{z} \in [n]^{n_a} : z_j \neq z_k \text{ for } j \neq k\}$. The observations in $\mathcal{D}_{t-1}$ are projected to this ordered representation, and the GP is fitted with the Merge kernel on the active items.

Stage 3: Acquisition optimization. VaPBO maximizes α_t over $\mathcal{Z}_{\mathcal{I}_t}$. The experiments use LS, GLS, EA, and Grad for this step; VaPBO-GLS is the main variant.

Stage 4: Best-K fill-in. Let $\mathcal{P}_K^*$ contain the K best full permutations observed so far. For a proposed active configuration $\mathbf{z}$, VaPBO selects the member of $\mathcal{P}_K^*$ with the smallest squared positional distance on the active items. It fixes the active items at the proposed positions and fills the remaining positions according to the inactive-item order in this reference permutation.

Algorithm 2 VaPBO with Random Active-Item Selection and Best-K Fill-in

Input: Objective $f : \mathbb{S}_n \to \mathbb{R}$, active dimension n_a, fill-in parameter K
Initialize: $\mathcal{D}_0 \leftarrow$ InitialDesign(), $\mathcal{H}_0 \leftarrow \mathcal{D}_0$
for $t = 1, \ldots, T$ **do**
  Sample $\mathcal{I}_t \subset [n]$ uniformly at random with $|\mathcal{I}_t| = n_a$
  Define $\mathcal{Z}_{\mathcal{I}_t} \leftarrow \{\mathbf{z} \in [n]^{n_a} : z_j \neq z_k \text{ for } j \neq k\}$
  $\widetilde{\mathcal{D}}_{t-1} \leftarrow \{(\mathbf{z}_{\mathcal{I}_t}(\pi), y) : (\pi, y) \in \mathcal{D}_{t-1}\}$
  Fit $\mathcal{M}_t$ to $\widetilde{\mathcal{D}}_{t-1}$ using the Merge kernel
  $\mathbf{z}_t^* \leftarrow \arg\max_{\mathbf{z} \in \mathcal{Z}_{\mathcal{I}_t}} \alpha_t(\mathbf{z} \mid \mathcal{M}_t)$
  $\pi_t \leftarrow \Psi(\mathbf{z}_t^*, \mathcal{I}_t; \mathcal{P}_K^*(\mathcal{H}_{t-1}))$
  $y_t \leftarrow f(\pi_t)$; $\mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(\pi_t, y_t)\}$
  Update $\mathcal{H}_t$ and the top-K set $\mathcal{P}_K^*$
end for
Return: Best observed permutation π_{best}

B ADDITIONAL RESULTS

We first report high-dimensional convergence and additional runs, followed by low-dimensional comparisons. We then examine sensitivity to the active dimension and fill-in parameter, acquisition optimization and search behavior, and the effects of surrogate guidance and completion.

Implementation sources. We use the released Merge and Vanilla Merge code.[1]

B.1 HIGH-DIMENSIONAL OPTIMIZATION

Convergence across tasks. Figure 6 reports task-wise convergence curves. The middle and bottom rows show the eight high-dimensional tasks in Table 2; the top row provides four low-dimensional controls. The high-dimensional curves complement the final ranks in Figure 2 and the rank trajectories in Figure 3. VaPBO-GLS performs well across these tasks, while VaP-GLS is also competitive, particularly on the biodiversity tasks.

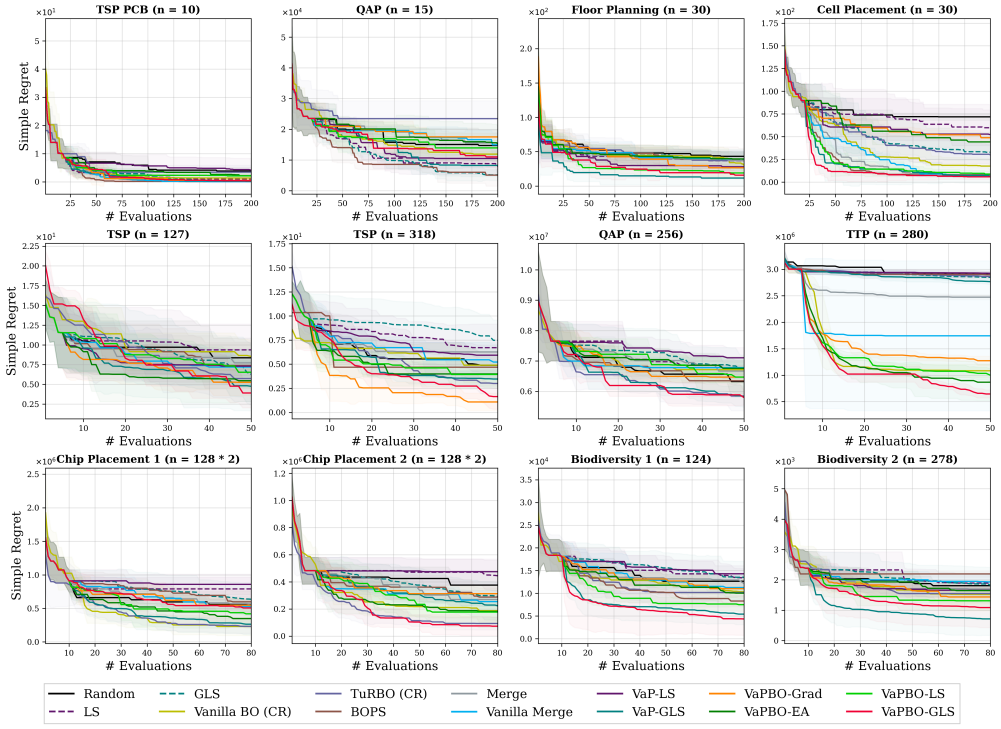

Figure 6: Best-so-far regret curves for the compared methods. The top row shows four low-dimensional tasks; the remaining rows show high-dimensional combinatorial and application tasks.

[1] https://github.com/XieZikai/MergeBO; https://github.com/XZT008/Standard-GP-is-all-you-need-for-HDBO.

Validation with additional seeds. We ran ten additional paired seeds (10–19) with a budget of 50 objective evaluations on TSP318 and QAP256, including five shared initial permutations for each task and seed. We used an active ratio of $1/2$, Best-K fill-in with $K = 3$, and EI for VaPBO-GLS. Figure 7 compares VaPBO-GLS, VaP-GLS, VaP-GLS with random fill-in, and simulated annealing. We report mean $\pm$ sample standard deviation across the ten runs. The metric is best-so-far cost, with TSP tour cost divided by 10^4 and QAP cost unscaled.

On TSP318, VaPBO-GLS attained a final cost of 106.94 ± 1.89, versus 110.51 ± 1.51 for VaP-GLS, and improved in all ten paired runs. The mean paired relative cost reduction was 3.22%. On QAP256, the corresponding costs were $50{,}442{,}475 \pm 654{,}699$ and $50{,}733{,}012 \pm 468{,}906$. VaPBO-GLS improved in seven of ten runs, with a mean paired relative reduction of 0.57%. The final-cost differences between Best-K and random fill-in within VaP-GLS were small relative to their variation across runs on both tasks. Thus, the additional runs support an optimization benefit on TSP318, while the benefit on QAP256 is smaller and less consistent. Candidate-search effort differs between the GP-guided and no-GP strategies; Appendix B.5 controls the available candidates.

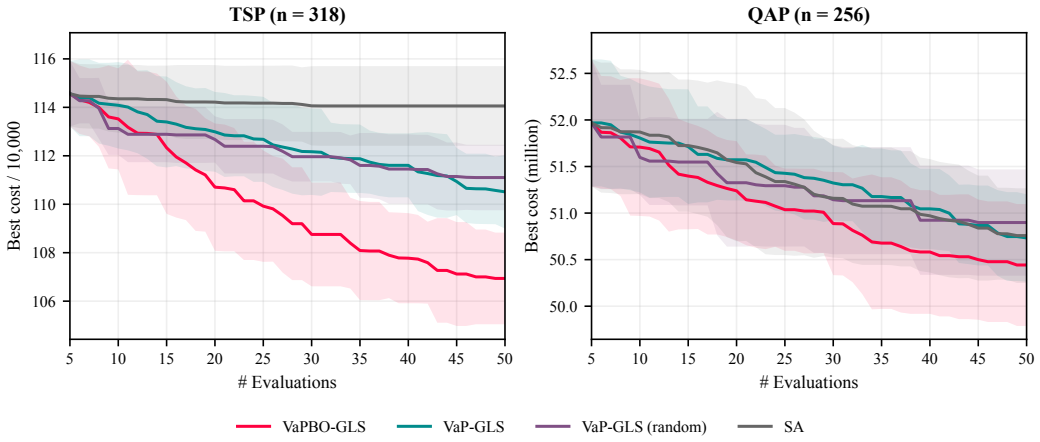

Figure 7: Independent 50-evaluation comparison on TSP318 and QAP256. Curves show mean best-so-far cost over ten seeds; shading denotes standard deviation. Lower is better.

B.2 LOW-DIMENSIONAL COMPARISONS

We also compare methods on smaller permutation problems. Among the 14 methods in Table 4, VaP-GLS has the best average rank (2.75), while BOPS and VaPBO-GLS tie at 3.75. Thus, full-space BOPS remains competitive and GP guidance adds little to VaP-GLS in these smaller problems. Figure 8 adds ComBO to the average-rank comparison. BOPS and VaPBO-GLS both obtain an average rank of 3.75 in this comparison.

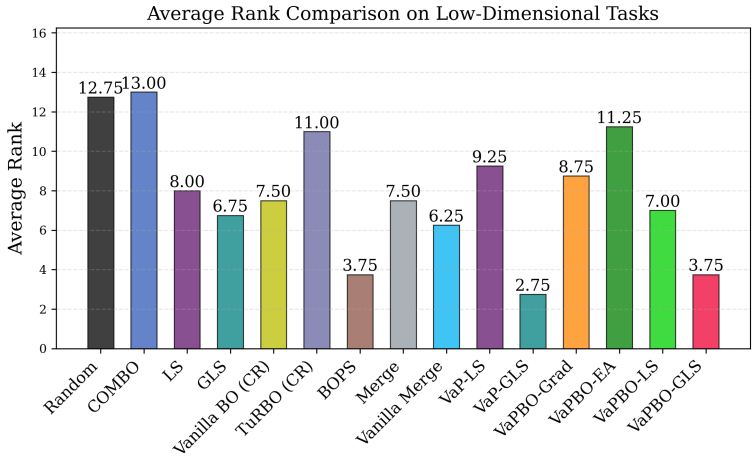

Figure 8: Average ranks on low-dimensional tasks. BOPS and VaPBO-GLS perform comparably, showing that full-space permutation BO remains competitive on smaller problems.

Method	TSP10	QAP15	Floor Planning30	Cell Placement30	Avg. rank
Random	3.82±1.28	14707.60±5128.01	43.20±5.78	71.80±3.71	12.50
LS	1.01±1.33	8040.80±2078.41	36.80±11.79	59.80±19.64	7.75
GLS	0.28±0.25	5136.00±1780.90	38.00±15.18	32.80±8.73	6.75
Vanilla BO (CR)	1.34±1.13	11897.60±908.88	30.20±10.48	17.60±8.36	7.25
TuRBO (CR)	3.40±2.49	23452.00±5848.59	37.00±15.07	29.80±13.17	10.50
BOPS	0.05±0.10	5053.60±3575.96	37.00±17.89	8.00±0.89	3.75
Merge	0.13±0.20	15536.00±4144.56	37.20±10.01	6.80±2.14	7.50
Vanilla Merge	0.05±0.06	14934.00±3937.79	39.40±11.09	6.40±1.20	6.25
VaP-LS	4.20±4.04	10492.40±5213.00	28.00±3.03	52.40±32.89	9.00
VaP-GLS	0.13±0.00	8210.40±3611.51	11.60±11.76	6.40±7.74	2.75
VaPBO-Grad	0.52±0.34	17500.80±1189.08	26.60±13.97	48.80±4.35	8.75
VaPBO-EA	2.17±1.03	15248.80±5784.04	39.40±9.58	44.20±10.55	11.00
VaPBO-LS	2.09±1.85	13924.40±2607.40	19.00±6.87	8.80±3.25	6.75
VaPBO-GLS	0.36±0.25	10990.40±3278.00	15.80±5.53	5.80±3.12	3.75

Table 4: Low-dimensional regret results: mean ± standard deviation over five independent runs.

B.3 SENSITIVITY ANALYSIS

Active dimension ratio. Figure 5 compares active ratios $n_a/n \in \{25\%, 50\%, 75\%\}$ on Chip Placement 2 and Biodiversity 2. The 50% setting gives the lowest final regret on Chip Placement 2, while 25% is better on Biodiversity 2. Both reduced settings outperform 75% at the final evaluation. Thus, the preferred active ratio varies by task, and moderate reduction provides a useful default.

Best-K fill-in sensitivity. Figure 9 compares $K \in \{1, 3, 5\}$ for VaPBO-LS. Convergence trends are similar on most tasks, and the default $K = 3$ performs consistently well.

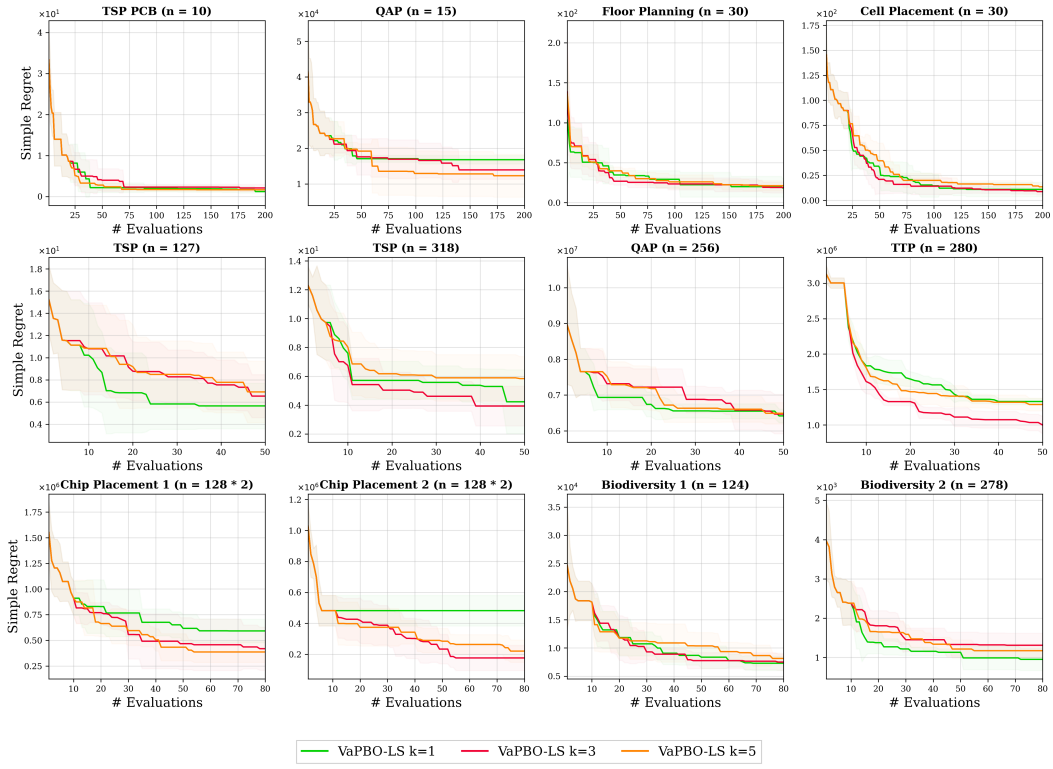

Figure 9: Best-so-far regret for $K \in \{1, 3, 5\}$ in the Best-K fill-in rule.

B.4 Acquisition optimization and search behavior

Search behavior through OTSD. To examine how acquisition optimization affects search behavior, we use Observation Traveling Salesperson Distance (OTSD) (Papenmeier et al., 2025a), which measures the length of a shortest tour through evaluated points. Figure 10 shows normalized OTSD and its difference from Random Search for VaPBO variants and VaP-GLS. VaP-GLS has the lowest OTSD, followed by VaPBO-GLS, whereas the other VaPBO variants remain closer to Random Search. This ordering indicates more concentrated search for the GLS variants. VaPBO-GLS also has the best final average rank in Table 2; Appendix B.5 directly examines the quality of candidates selected by the surrogate.

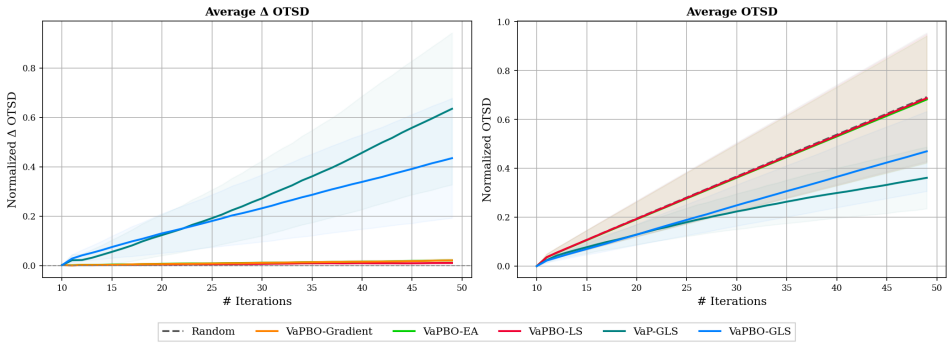

Figure 10: Full OTSD comparison, including the difference from Random Search shown on the left.

EI and LogEI for VaPBO-Grad. Figure 11 compares EI and LogEI for VaPBO-Grad on low- and high-dimensional tasks. LogEI gives lower final regret on TSP127 and QAP15, whereas EI is better on TSP318 and QAP256. The effect of the acquisition function therefore depends on the task.

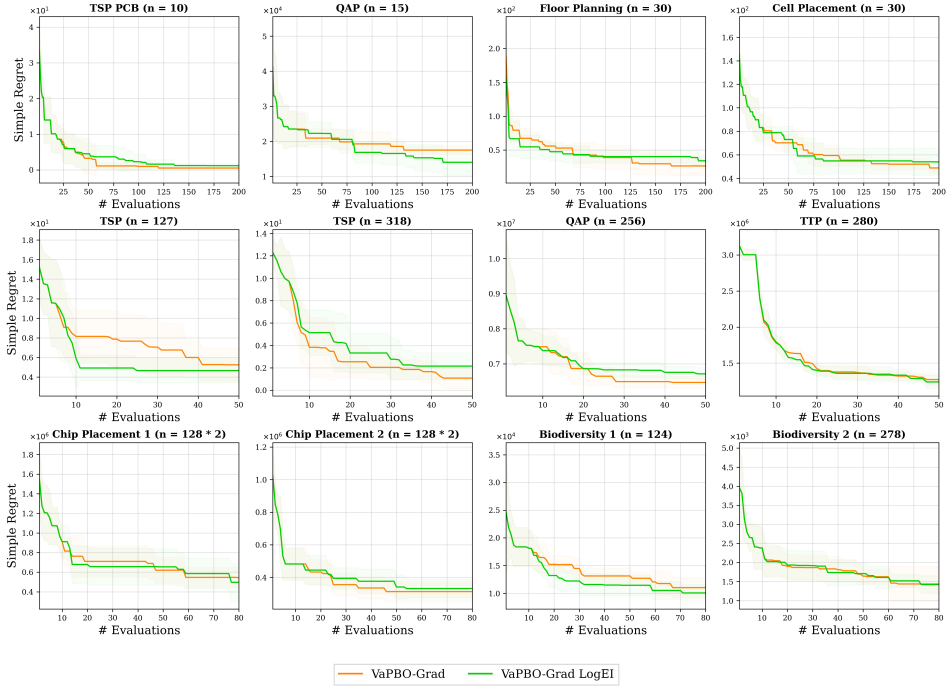

Figure 11: Best-so-far regret with EI and LogEI for VaPBO-Grad.

B.5 CANDIDATE SELECTION AND STRUCTURE-AWARE COMPLETION

Candidate selection with a fixed history and candidate pool. For each task, we took the first 20 observations from each of the ten VaP-GLS runs and constructed 64 unique feasible candidates using a fixed active set and Best-K fill-in. Full-space and subspace Merge GPs used the same history and candidate pool; candidate objective values were evaluated after fitting the models and selecting by EI. We define quality percentile as the fraction of candidates with higher cost than the selected candidate, counting ties with weight $1/2$. Higher values indicate better choices, and uniform random selection has an expected percentile of 50%.

Subspace EI selected candidates at quality percentiles of $69.4\% \pm 21.9\%$ on TSP318 and $79.6\% \pm 16.7\%$ on QAP256 (Figure 4, left). Full-space EI also selected useful candidates, with percentiles of $63.3\% \pm 24.6\%$ and $75.1\% \pm 17.2\%$, respectively. These results show that subspace EI contains useful information for choosing among the available candidates. The mean and standard deviation are computed across ten independent histories per task.

Best-K fill-in with fixed active-item positions. In an additional exploratory experiment, we used fresh seeds 20–29 to generate ten VaP-GLS histories of 20 evaluations per task. For each history, we fixed an active set and 64 candidate assignments of active items to positions. Each assignment was completed using either Best-K ($K = 3$) or a random ordering of inactive items. Thus, each pair differed only in how the inactive items filled the remaining positions. We first averaged the relative cost reduction within each history, then computed the mean and standard deviation across histories.

Best-K reduced candidate costs by $1.35\% \pm 1.46\%$ on TSP318 and $2.24\% \pm 1.16\%$ on QAP256 (Figure 4, right). This comparison directly supports the benefit of retaining inactive-item order from elite solutions when completing a candidate. The 50-evaluation optimization results are reported in Appendix B.1. Objective evaluations for both candidate diagnostics are additional to the online optimization budgets.