6th ICML Workshop on Automated Machine Learning (2019)

Bayesian Optimization over Sets

Jungtaek Kim JTKIM@POSTECH.AC.KR
Pohang University of Science and Technology, Pohang, Republic of Korea

Michael McCourt MCCOURT@SIGOPT.COM
SigOpt, San Francisco, USA

Tackgeun You TACKGEUN.YOU@POSTECH.AC.KR
Pohang University of Science and Technology, Pohang, Republic of Korea

Saehoon Kim SHKIM@AITRICS.COM
AITRICS, Seoul, Republic of Korea

Seungjin Choi SEUNGJIN@POSTECH.AC.KR
Pohang University of Science and Technology, Pohang, Republic of Korea

Abstract

We propose a Bayesian optimization method over sets, to minimize a black-box function that can take a set as single input. Because set inputs are permutation-invariant and variable-length, traditional Gaussian process-based Bayesian optimization strategies which assume vector inputs can fall short. To address this, we develop a Bayesian optimization method with *set kernel* that is used to build surrogate functions. This kernel accumulates similarity over set elements to enforce permutation-invariance and permit sets of variable size, but this comes at a greater computational cost. To reduce this burden, we propose a more efficient probabilistic approximation which we prove is still positive definite and is an unbiased estimator of the true set kernel. Finally, we present several numerical experiments which demonstrate that our method outperforms other methods in various applications.

1. Introduction

Bayesian optimization (BO) is an effective method to optimize a black-box function which is expensive to evaluate. It has proven useful in several applications, including hyperparameter optimization (Snoek et al., 2012; Hutter et al., 2011), material design (Frazier and Wang, 2016), and synthetic gene design (González et al., 2014). Classic BO assumes that a search region $\mathcal{X} \subset \mathbb{R}^d$ is defined and that the black-box function f can only produce scalar output in the presence of additive noise ϵ, i.e., $y = f(\mathbf{x}) + \epsilon$ for $\mathbf{x} \in \mathcal{X}$.

Unlike this standard BO formulation, in this article we assume that our search region is $\mathcal{X}_{\text{set}} = \{\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \mid \mathbf{x}_i \in \mathbb{R}^d\}$ for a fixed positive integer m. Thus, for $\mathbf{X} \in \mathcal{X}_{\text{set}}$, f would take in a *set* containing m elements, all of length d, and return a noisy function value y:

$$y = f(\mathbf{X}) + \epsilon. \tag{1}$$

Our motivating example comes from the soft k-means clustering algorithm over a dataset $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$; in particular, we want to find the optimal initialization of such an algorithm. The objective function for this problem is a squared loss function which takes in the cluster initialization points $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ and returns the weighted distance between the points in $\mathcal{P}$ and the converged cluster centers $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$.

Some previous research has tried to build Gaussian process (GP) models on set data. (Garnett et al., 2010) proposes a method over discrete sets using stationary kernels over the first Wasserstein distance between two sets, though the power set of fixed discrete sets as domain space is not our interest. However, this method needs the complexity $\mathcal{O}(n^2m^3d)$, to compute a covariance matrix with respect to n sets. Moreover, because it only considers stationary kernels, GP regression is restricted to the form that cannot express non-stationary models (Paciorek and Schervish, 2004).

Therefore, we instead adapt and augment a strategy proposed by (Gätner et al., 2002) involving the creation of a specific *set kernel*. This set kernel uses a kernel defined on the elements $\mathbf{x} \in \mathbb{R}^d$ of the sets to build up its own sense of covariance between sets. In turn, then, it can be directly used to build surrogate functions through GP regression, which then can power BO, by Lemma 1.

A key contribution of this article is the development of a computationally efficient approximation to this set kernel. Given n total observed function values, the cost of constructing the matrix required for fitting the GP is $\mathcal{O}(n^2m^2d)$ where $m \geq n$ approximately. We propose the use of random subsampling to reduce the computational cost to $\mathcal{O}(n^2L^2d)$ for $L < m$ while still producing an unbiased estimate of the expected value of the true kernel.

2. Background

2.1 Bayesian Optimization

BO seeks to minimize an unknown function f which is expensive to evaluate:

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{2}$$

where $\mathcal{X} \subset \mathbb{R}^d$ is a compact space. It is a sequential optimization strategy which, at each iteration, balances exploration and exploitation with surrogate model and acquisition function. After exhausting a predefined observation budget T, BO returns the best point $\mathbf{x}^\dagger$ that has the minimum observation. The benefit of this process is that the optimization of the expensive function f has been replaced by the optimization of much cheaper and more well-understood acquisition functions a_n.

In this paper, we use GP regression (Rasmussen and Williams, 2006) to produce the surrogate function s_n; from s_n, we use the standard acquisition function expected improvement (Moćkus et al., 1978): $a_n(\mathbf{x}) = \mathbb{E}[(y_n^\dagger - s_n(\mathbf{x}))_+]$, where $y_n^\dagger = \min_{1\leq i\leq n} y_i$. See (Brochu et al., 2010; Shahriari et al., 2016; Frazier, 2018) for further details on BO.

2.2 Set Kernel

We start by introducing notation which is required for performing kernel approximation of functions on set data. A set of m vectors is denoted $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$. In a collection of n such sets (as will occur in the BO setting), the kth set would be denoted $\mathbf{X}^{(k)} = \{\mathbf{x}_1^{(k)}, \ldots, \mathbf{x}_m^{(k)}\}$. Note that we are restricting all sets to be of the same size $|\mathbf{X}^{(k)}| = m$ here.

To build a GP surrogate, we require a prior belief of the covariance between elements in $\mathcal{X}_{\text{set}} = \{\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \mid \mathbf{x}_i \in \mathbb{R}^d\}$. This belief is imposed in the form of a positive-definite covariance kernel $k_{\text{set}} : \mathcal{X}_{\text{set}} \times \mathcal{X}_{\text{set}} \to \mathbb{R}$; see (Schölkopf and Smola, 2002; Fasshauer and

McCourt, 2015) for more discussion on approximation with kernels. In addition to the symmetry $k_{\text{set}}(\mathbf{X}^{(i)}, \mathbf{X}^{(j)}) = k_{\text{set}}(\mathbf{X}^{(j)}, \mathbf{X}^{(i)})$ required in standard kernel settings, kernels on sets require an additional property. The ordering of elements in $\mathbf{X}$ should be immaterial (since sets have no inherent ordering).

Given an empirical approximation of the kernel mean $\boldsymbol{\mu}_{\mathbf{X}} \approx |\mathbf{X}|^{-1} \sum_{i=1}^{|\mathbf{X}|} \phi(\mathbf{x}_i)$ where $\phi(\cdot)$ is a basis function $\mathbb{R}^d \to \mathbb{R}^{d'}$ and d' is a dimension of projected space by ϕ, a set kernel (Gätner et al., 2002; Muandet et al., 2017) is defined as

$$k_{\text{set}}\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\right) = \langle \boldsymbol{\mu}_{\mathbf{X}^{(1)}}, \boldsymbol{\mu}_{\mathbf{X}^{(2)}} \rangle = \frac{1}{|\mathbf{X}^{(1)}||\mathbf{X}^{(2)}|} \sum_{i=1}^{|\mathbf{X}^{(1)}|} \sum_{j=1}^{|\mathbf{X}^{(2)}|} k\left(\mathbf{x}_i^{(1)}, \mathbf{x}_j^{(2)}\right) \tag{3}$$

since $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$. Here, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive-definite kernel defined to measure the covariance between the d-dimensional elements of the sets.

3. Proposed Method

In order for (3) to be a viable covariance kernel of a GP regression, it must be positive-definite. To discuss this topic, we denote a list of n sets with the notation $\mathfrak{X} = [\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}] \in {\mathcal{X}_{\text{set}}}^n$; in this notation, the order of the entries matters.

Lemma 1 *Suppose we have a list $\mathfrak{X}$ which contains distinct sets $\mathbf{X}^{(i)}$ for $1 \leq i \leq n$. We define the matrix $\mathsf{K} \in \mathbb{R}^{n \times n}$ as*

$$(\mathsf{K})_{ij} = k_{set}\left(\mathbf{X}^{(i)}, \mathbf{X}^{(j)}\right), \tag{4}$$

for k_{set} defined with a chosen inner kernel k as in (3). Then, K is a symmetric positive-semidefinite matrix if k is a symmetric positive-definite kernel.

This proof appears in (Haussler, 1999), and is discussed in (Gätner et al., 2002).

3.1 Approximation of the Set Kernel

Computing (4) requires pairwise comparisons between all sets present in $\mathfrak{X}$, which has a complexity $\mathcal{O}(n^2 m^2 d)$. To alleviate this cost, we propose to approximate (3) with

$$\tilde{k}_{\text{set}}\left(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}; \pi, \mathbf{w}, L\right) = k_{\text{set}}\left(\tilde{\mathbf{X}}^{(1)}, \tilde{\mathbf{X}}^{(2)}\right) \tag{5}$$

where $\pi : [1, \ldots, m] \to [1, \ldots, m]$, $\mathbf{w} \in \mathbb{R}^d$ and $L \in \mathbb{Z}_+$, and $\tilde{\mathbf{X}}^{(i)}$ is a subset of $\mathbf{X}^{(i)}$ which is defined by those three quantities (we omit explicitly listing them in $\tilde{\mathbf{X}}^{(i)}$ to ease the notation).

The goal of the approximation strategy is to convert from $\mathbf{X}^{(i)}$ (of size m) to $\tilde{\mathbf{X}}^{(i)}$ (of size L) in a consistent fashion during all the $\tilde{k}_{\text{set}}$ computations comprising K. We accomplish this in two steps: (i) use a randomly generated vector $\mathbf{w}$ to impose an (arbitrary) ordering of the elements of all sets $\mathbf{X}^{(i)}$, and (ii) randomly permute the indices $[1, \ldots, m]$ via a function π. These random strategies are defined once before computing the K matrix, and then used consistently throughout the entire computation.

To impose an ordering of the elements, we use a *random scalar projection* $\mathbf{w} \in \mathbb{R}^d$ such that the elements of $\mathbf{w}$ are drawn from the standard normal distribution. If the scalar projections of each $\mathbf{x}_i$ are computed, this produces the set of scalar values $\{\mathbf{w}^\top \mathbf{x}_1, \ldots, \mathbf{w}^\top \mathbf{x}_m\}$, which can be sorted to generate an ordered list of $[\ell_1, \ldots, \ell_m]$, $\mathbf{w}^\top \mathbf{x}_{\ell_1} \leq \cdots \leq \mathbf{w}^\top \mathbf{x}_{\ell_m}$, for an ordering of distinct indices $\ell_1, \ldots, \ell_m \in [1, \ldots, m]$. Ties between $\mathbf{w}^\top \mathbf{x}_i$ values can be dealt with arbitrarily. The function π then is simply a random bijection of the integers $[1, \ldots, m]$ onto themselves. Using this, we can sample L vectors from $\mathbf{X}^{(i)}$:

$$\tilde{\mathbf{X}}^{(i)} = \{\mathbf{x}_{\ell_j} \mid \ell_j = \pi(j) \text{ for } 1 \leq j \leq L\}. \tag{6}$$

This process, given $\mathbf{w}$, π and L is sufficient for computing k_{set}.

3.2 Properties of the approximation

The covariance matrix for this approximation of the set kernel, which we denote by $(\tilde{\mathsf{K}})_{ij} = \tilde{k}_{\text{set}}(\mathbf{X}^{(i)}, \mathbf{X}^{(j)}; \mathbf{w}, \pi, L)$, should approximate the full version of covariance matrix, K from (4). Because of the random structure introduced in Section 3.1, the matrix $\tilde{\mathsf{K}}$ will be random. This will be addressed in Theorem 3, but for now, $\tilde{\mathsf{K}}$ represents a single realization of that random variable, not the random variable itself. To be viable, this approximation must satisfy the following requirements: (i) pairwise symmetry, (ii) permutation invariance, (iii) positive definiteness, and (iv) reduced computational cost, which are directly satisfied.

Missing from this list is a statement regarding the quality of the approximation. We address this in Theorem 3, though we first start by stating Lemma 2. Moreover, using Theorem 3, we can also obtain the variance of our estimate, as shown in Theorem 4.

Lemma 2 *Suppose there are two sets $\mathbf{X}, \mathbf{Y} \in \mathcal{X}_{set}$. Without loss of generality, let $\mathbf{X}^{(i)}$ and $\mathbf{Y}^{(j)}$ denote the ith and jth of $\binom{m}{L}$ possible subsets containing L elements of $\mathbf{X}$ and $\mathbf{Y}$, respectively, in an arbitrary ordering. For $1 \leq L \leq m$,*

$$\sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}) = \frac{L^2 \binom{m}{L}^2}{m^2} \sum_{c=1}^{m} \sum_{d=1}^{m} k(\mathbf{x}_c, \mathbf{y}_d), \tag{7}$$

where $\bar{\mathbf{x}}_a^{(i)}$ and $\bar{\mathbf{y}}_b^{(j)}$ are the ath and bth elements of $\mathbf{X}^{(i)}$ and $\mathbf{Y}^{(j)}$, respectively, in an arbitrary ordering.

Theorem 3 *Suppose that we are given two sets $\mathbf{X}, \mathbf{Y} \in \mathcal{X}_{set}$ and $L \in \mathbb{Z}_+$. Suppose, furthermore, that $\mathbf{w}$ and π can be generated randomly as defined in Section 3.1 to form subsets $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$. The value of $\tilde{k}_{set}(\mathbf{X}, \mathbf{Y}; \mathbf{w}, \pi, L)$ is an unbiased estimator of the value of $k_{set}(\mathbf{X}, \mathbf{Y})$.*

Theorem 4 *Suppose the same conditions as in Theorem 3. Suppose, furthermore, that $k(\mathbf{x}, \mathbf{x}') \geq 0$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. The variance of $\tilde{k}_{set}(\mathbf{X}, \mathbf{Y}; \mathbf{w}, \pi, L)$ is bounded by a function of m, L and $k_{set}(\mathbf{X}, \mathbf{Y})$:*

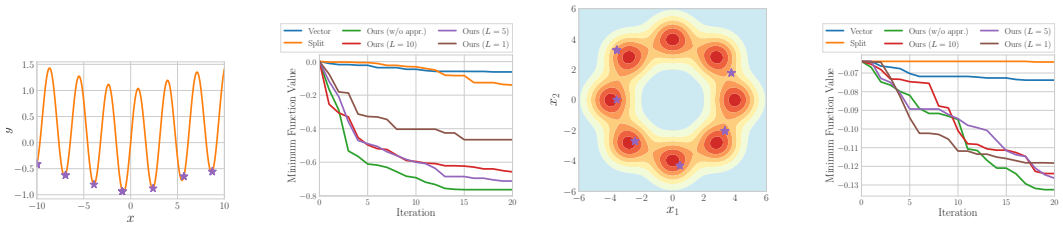
$$Var\left[\tilde{k}_{set}(\mathbf{X}, \mathbf{Y}; \mathbf{w}, \pi, L)\right] \leq \left(\frac{m^4}{L^4} - 1\right) k_{set}(\mathbf{X}, \mathbf{Y})^2. \tag{8}$$

Proof To accommodate the page limit, the proofs of Lemma 2, Theorem 3 and Theorem 4 are provided in the appendix. ■

Algorithm 1 Bayesian Optimization over Sets

Input: A domain $\mathcal{X}_{\text{set}}$, a function $f : \mathcal{X}_{\text{set}} \to \mathbb{R}$, a budget $T \in \mathbb{Z}_+$.
Output: Best acquired set $\mathbf{X}^\dagger$

1: Choose an initial point $\mathbf{X}^{(1)}$ randomly from $\mathcal{X}_{\text{set}}$ and evaluate $y_1 = f(\mathbf{X}^{(1)}) + \epsilon_1$.
2: **for** k from 1 to $T - 1$ **do**
3: Fit the surrogate model s_k to all available data $\{(\mathbf{X}^{(i)}, y_i)\}_{i=1}^k$.
4: Compute the acquisition function a_k from s_k.
5: Identify $\mathbf{X}^{(k+1)} = \arg\max_{\mathbf{X} \in \mathcal{X}_{\text{set}}} a_k(\mathbf{X})$.
6: Evaluate $y_{k+1} = f(\mathbf{X}^{(k+1)}) + \epsilon_k$.
7: **end for**
8: **return** $\mathbf{X}^\dagger = \mathbf{X}^{(i)}$ if $y_i = \max_{1 \le j \le T} y_j$.



(a) Syn. 1: Acquired set (b) Syn. 1: BO (c) Syn. 2: Acquired set (d) Syn. 2: BO

Figure 1: Results on two synthetic functions. All experiments are repeated ten times.

3.3 Bayesian Optimization over Sets

For BO over $\mathcal{X}_{\text{set}}$, the goal is to identify the set $\mathbf{X} \in \mathcal{X}_{\text{set}}$ such that a given function $f : \mathcal{X}_{\text{set}} \to \mathbb{R}$ is minimized. As shown in Algorithm 1, BO over sets follows similar steps as laid out in Section 2.1, except that it involves the space of set inputs and requires a surrogate function on $\mathcal{X}_{\text{set}}$. As we have indicated, we plan to use a GP surrogate function, with prior covariance defined either with (3) or (5) and a Matérn 5/2 inner kernel k.

Using a GP model requires computation on the order of $\mathcal{O}(n^3)$ at the nth step of the BO because the K matrix must be inverted. Compared to the complexity for computing a full version of the set kernel $\mathcal{O}(n^2 m^2 d)$, the complexity of computing the inverse is smaller if roughly $m \geq n$ (that is, computing the matrix can be as costly or more costly than inverting it). Because BO is efficient sampling-based global optimization, n is small and the situation $m \geq n$ is reasonable. Therefore, the reduction proposed by the approximation in Section 3.1 can be effective in reducing complexity of all steps for BO over sets.

4. Experiments

In the appendix we describe the baseline methods against which we compare our method. All codes are implemented using `bayeso` (Kim and Choi, 2017). The free parameters for GP regression are obtained by maximizing a marginal likelihood.

4.1 Synthetic Functions

We test two synthetic circumstances to show BO over sets is a valid approach to find an optimal set that minimizes an objective function $f : \mathcal{X}_{\text{set}} \to \mathbb{R}$. In each setting, there is

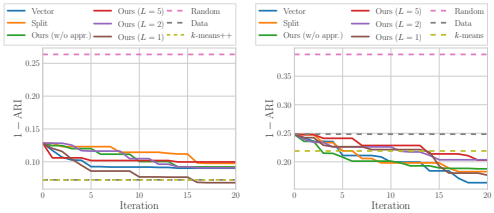


Figure 2: Results on initializing k-means clustering and GMM are averaged over 10 trials. To compare with the baselines fairly, Random, Data, k-means++ (Arthur and Vassilvitskii, 2007), k-means are run 1,000 times.

an auxiliary function $g : \mathcal{X} \to \mathbb{R}$, and the function f is defined as $f(\mathbf{X}) = \frac{1}{m}\sum_{i=1}^{m} g(\mathbf{x}_i)$. The g functions (see the appendix) are designed to be multimodal, giving the opportunity for the set $\mathbf{X}$ to contain $\mathbf{x}_i$ values from each of the modes in the domain. Additionally, as is expected, f is permutation invariant (any ordering imposed on the elements of $\mathbf{x}$ is immaterial).

Synthetic 1 We consider $d = 1$, $m = 20$ and choose g to be a simple periodic function.

Synthetic 2 g is the summation of probability density functions, where $d = 2$, $m = 20$.

As shown in Figure 1, both of these circumstances have a clear multimodal structure, allowing for optimal sets to contain points which are clustered in a single local minima or to be spread out through the domain in several local minima. The first three columns of Figure 1, show that the "Vector" and "Split" strategies have difficulty optimizing functions in both circumstances. On the other hand, our proposed method finds optimal outcomes more effectively.

4.2 Initialization of Clustering Methods

We initialize two clustering methods for dataset $\mathcal{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]$ with BO over sets: (i) k-means clustering, and (ii) Gaussian mixture model (GMM). For these experiments, we add four additional baselines for clustering algorithms (see the appendix for the additional baselines). To fit a dataset with those four baselines, we use the whole dataset without splitting. For two clustering algorithms, we generate a dataset where $N = 500$, $d = 5$, and $k = 10$. We split the dataset to training (70%) and test (30%) datasets. In BO settings, after finding the converged cluster centers $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ with training dataset, the adjusted Rand index (ARI) is computed by test dataset. The algorithms are optimized over $1 - \text{ARI}$. All clustering models are implemented using `scikit-learn` (Pedregosa et al., 2011). Due to the page limit, the detailed explanations of this experiment and Figure 2 are described in the appendix.

5. Conclusion

In this paper, we propose the BO method over sets, which takes a set as an input and produces a scalar output. Our method based on GP regression models a surrogate function using set-taking covariance functions, referred to as set kernel. We approximate the set kernel to the efficient positive-definite kernel that is an unbiased estimator of the original set kernel. Our experimental results demonstrate our method can be used in some novel applications for BO.

References

D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1027–1035, New Orleans, Louisiana, USA, 2007.

E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv e-prints*, arXiv:1012.2599, 2010.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

H. Edwards and A. Storkey. Towards a neural statistician. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

G. E. Fasshauer and M. M. McCourt. *Kernel-based Approximation Methods Using Matlab*. World Scientific, 2015.

C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1126–1135, Sydney, Australia, 2017.

P. I. Frazier. A tutorial on Bayesian optimization. *arXiv e-prints*, arXiv:1807.02811, 2018.

P. I. Frazier and J. Wang. Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pages 45–75. Springer, 2016.

M. Garnelo, D. Rosenbaum, C. J. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. J. Rezende, and S. M. A. Eslami. Conditional neural processes. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1690–1699, Stockholm, Sweden, 2018.

R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for sensor set selection. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 209–219, Stockholm, Sweden, 2010.

T. Gätner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 179–186, Sydney, Australia, 2002.

M. G. Genton. Classes of kernels for machine learning: a statistics perspective. *Journal of Machine Learning Research*, 2:299–312, 2001.

J. González, J. Longworth, D. C. James, and N. D. Lawrence. Bayesian optimization for synthetic gene design. In *Neural Information Processing Systems Workshop on Bayesian Optimization (BayesOpt)*, Montreal, Quebec, Canada, 2014.

D. Haussler. Convolution kernels on discrete structures. Technical report, Department of Computer Science, University of California at Santa Cruz, 1999.

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the International Conference on Learning and Intelligent Optimization*, pages 507–523, Rome, Italy, 2011.

J. Kim and S. Choi. bayeso: A Bayesian optimization framework in Python. `https://github.com/jungtaekkim/bayeso`, 2017.

S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

J. Moćkus, V. Tiesis, and A. Źilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

C. J. Paciorek and M. J. Schervish. Nonstationary covariance functions for Gaussian process regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 17, pages 273–280, Vancouver, British Columbia, Canada, 2004.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 25, pages 2951–2959, Lake Tahoe, Nevada, USA, 2012.

M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 3391–3401, Long Beach, California, USA, 2017.

Appendix A. Related Works

Although it has been raised in different interests, meta-learning approaches dealt with set inputs are promising in machine learning community, because they can generalize distinct tasks with meta-learners (Edwards and Storkey, 2017; Zaheer et al., 2017; Finn et al., 2017; Garnelo et al., 2018). In particular, (Edwards and Storkey, 2017; Zaheer et al., 2017; Garnelo et al., 2018) propose feed-forward neural networks which take permutation-invariant and variable-length inputs: they have the goal of obtaining features derived from the sets with which to input to a standard (meta-)learning routine. Because they consider modeling of set data, they are related to our work, but they are interested in their own specific examples such as point cloud classification, few-shot learning, and image completion.

In BO, (Garnett et al., 2010) suggests a method to find a set that produces a global minimum with respect to discrete sets, each of which is an element of power set of entire set. Because the time complexity of the first Wasserstein distanace is $\mathcal{O}(n^2m^3d)$, they assume a small cardinality of sets and discrete searching space for global optimization method. Furthermore, their method restricts the number of iterations for optimizing an acquisition function, since by the curse of dimensionality the number of iterations should be increased exponentially. However, it implies that the global solution of acquisition function is hard to be found.

Compared to (Garnett et al., 2010), we consider continuous domain space which implies an acquired set can be composed of any instances in a compact space $\mathcal{X}$. We thus freely use off-the-shelf global optimization method or multi-started local optimization method (Shahriari et al., 2016) with relatively large number of instances in sets. In addition, its structure of kernel is $k_{\text{st}}(d(\mathbf{X}^{(1)}, \mathbf{X}^{(2)}))$ where $k_{\text{st}}(\cdot)$ is a stationary kernel (Genton, 2001) and $d(\cdot,\cdot)$ is some distance function over two sets (e.g., in (Garnett et al., 2010) the first Wasserstein distance). Using the method proposed in the subsequent section, non-stationary kernels might be considered in modeling a surrogate function.

Appendix B. Proof of Lemma 2

Proof We can rewrite the original summation in a slightly more convoluted form, as

$$\begin{aligned}\sum_{i=1}^{\binom{m}{L}}\sum_{j=1}^{\binom{m}{L}}\sum_{a=1}^{L}\sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}) &= \sum_{i=1}^{\binom{m}{L}}\sum_{j=1}^{\binom{m}{L}}\sum_{a=1}^{L}\sum_{b=1}^{L}\sum_{c=1}^{m}\sum_{d=1}^{m} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}) I_{\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}}(\mathbf{x}_c, \mathbf{y}_d), \\ &= \sum_{c=1}^{m}\sum_{d=1}^{m}\left[\sum_{i=1}^{\binom{m}{L}}\sum_{j=1}^{\binom{m}{L}}\sum_{a=1}^{L}\sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}) I_{\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}}(\mathbf{x}_c, \mathbf{y}_d)\right], \quad (9)\end{aligned}$$

where $I_{\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}}(\mathbf{x}_c, \mathbf{x}_d) = 1$ if $\bar{\mathbf{x}}_a^{(i)} = \mathbf{x}_c$ and $\bar{\mathbf{y}}_b^{(j)} = \mathbf{y}_d$, and 0 otherwise. As these are finite summations, they can be safely reordered.

The symmetry in the structure and evaluation of the summation implies that as each $\mathbf{x}_c$ quantity will be paired with each $\mathbf{y}_d$ quantity the same number of times. Therefore, we need only consider the number of times that these quantities appear.

We recognize that this summation follows a pattern related to Pascal's triangle. Among the $\binom{m}{L}$ possible subsets $\bar{\mathbf{x}}$ of $\mathbf{X}$, only the fraction L/m of those contain the quantity $\mathbf{x}_c$ for

all $1 \leq c \leq m$ (irrespective of how that entry may be denoted in $\bar{\mathbf{x}}_a^{(i)}$ terminology). Because of the symmetry mentioned above, each of those $\mathbf{x}_c$ quantities is paired with each of the $\mathbf{y}_d$ quantities the same $\frac{L}{m}\binom{m}{L}$ number of times. This result implies that

$$\sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}) I_{\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}}(\mathbf{x}_c, \mathbf{y}_d) = \frac{L^2 \binom{m}{L}^2}{m^2} k(\mathbf{x}_c, \mathbf{y}_d), \tag{10}$$

where $I_{\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}}(\mathbf{x}_c, \mathbf{y}_d) = 1$ if $\bar{\mathbf{x}}_a^{(i)} = \mathbf{x}_c$ and $\bar{\mathbf{y}}_b^{(j)} = \mathbf{y}_d$, and 0 otherwise. Substituting (10) into the bracketed quantity in (9) above completes the proof. ■

Appendix C. Proofs of Set Kernel Estimator Properties

We start by introducing the notation W and Π to be random variables such that $W \sim \mathcal{N}(0, \mathsf{I}_d)$ and Π is a uniformly random permutation of the integers between 1 and m. These are the distributions defining the $\mathbf{w}$ and π quantities described above. With this, we note that $\tilde{k}_{\text{set}}(\mathbf{X}, \mathbf{Y}; W, \Pi, L)$ is a random variable.

We also introduce the notation $\sigma_L(\mathbf{X})$ to be the distribution of random subsets of $\mathbf{X}$ with L elements selected without replacement, the outcome of the subset selection from Section 3.1. This notation allows us to write the quantities

$$\mathbb{E}_{W,\Pi}[\tilde{k}_{\text{set}}(\mathbf{X}, \mathbf{Y}; W, \Pi, L)] = \mathbb{E}_{\bar{\mathbf{X}}, \bar{\mathbf{Y}}}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] \equiv \mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})], \tag{11}$$

$$\text{Var}_{W,\Pi}[\tilde{k}_{\text{set}}(\mathbf{X}, \mathbf{Y}; W, \Pi, L)] = \text{Var}_{\bar{\mathbf{X}}, \bar{\mathbf{Y}}}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] \equiv \text{Var}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})], \tag{12}$$

for $\bar{\mathbf{X}} \sim \sigma_L(\mathbf{X}), \bar{\mathbf{Y}} \sim \sigma_L(\mathbf{Y})$. We have dropped the random variables from the expectation and variance definitions for ease of notation.

C.1 Proof of Theorem 3

Proof Our goal is to show that $\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] = k_{\text{set}}(\mathbf{X}, \mathbf{Y})$, where $\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})]$ is defined in (11).

We first introduce an extreme case: $L = m$. If $L = m$, the subsets we are constructing are the full sets, i.e., $\sigma_m(\mathbf{X})$ contains only one element, $\mathbf{X}$. Thus, $\tilde{k}_{\text{set}}(\mathbf{X}, \mathbf{Y}; W, \Pi, m) = k_{\text{set}}(\mathbf{X}, \mathbf{Y})$ is not a random variable.

For $1 \leq L < m$, we compute this expected value from the definition (with some abuse of notation):

$$\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] = \sum_{\bar{\mathbf{X}}, \bar{\mathbf{Y}}} k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}}) p(\bar{\mathbf{X}}, \bar{\mathbf{Y}}). \tag{13}$$

There are $\binom{m}{L}$ subsets, all of which could be indexed (arbitrarily) as $\bar{\mathbf{X}}^{(i)}$ for $1 \leq i \leq \binom{m}{L}$. The probability mass function is uniform across all subsets, meaning that $p(\bar{\mathbf{X}} = \bar{\mathbf{X}}^{(i)}, \bar{\mathbf{Y}} = \bar{\mathbf{Y}}^{(j)}) = 1/\binom{m}{L}^2$. Using this, we know

$$\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] = \sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} k_{\text{set}}(\bar{\mathbf{X}}^{(i)}, \bar{\mathbf{Y}}^{(j)}) \frac{1}{\binom{m}{L}^2}. \tag{14}$$

We apply (3) to see that

$$k_{\text{set}}(\bar{\mathbf{X}}^{(i)}, \bar{\mathbf{Y}}^{(j)}) = \frac{1}{L^2} \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}), \tag{15}$$

following the notational conventions used above. The expectation involves four nested summations,

$$\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] = \frac{1}{L^2 \binom{m}{L}^2} \sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{y}}_b^{(j)}). \tag{16}$$

We utilize Lemma 2 to rewrite this as

$$\mathbb{E}[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})] = \frac{1}{L^2 \binom{m}{L}^2} \frac{L^2 \binom{m}{L}^2}{m^2} \sum_{c=1}^{m} \sum_{d=1}^{m} k(\mathbf{x}_c, \mathbf{y}_d) = \frac{1}{m^2} \sum_{c=1}^{m} \sum_{d=1}^{m} k(\mathbf{x}_c, \mathbf{y}_d). \tag{17}$$

■

C.2 Proof of Theorem 4

Proof The variance of $k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})$, defined in (12), is computed as

$$\begin{aligned} \text{Var}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})\right] &= \mathbb{E}\left[\left(k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}}) - \mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})\right]\right)^2\right] \\ &= \mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})^2\right] + k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2 - 2k_{\text{set}}(\mathbf{X}, \mathbf{Y})\mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})\right] \\ &= \mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})^2\right] - k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2, \end{aligned} \tag{18}$$

where Theorem 3 is invoked to produce the final line. Using (14) and (15), we can express the first term of (18) as

$$\mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})^2\right] = \sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \left(\frac{1}{L^2} \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{x}}_b^{(j)}) \right)^2 \frac{1}{\binom{m}{L}^2}. \tag{19}$$

At this point, we invoke the fact that $k(\mathbf{x}, \mathbf{x}') \geq 0$ to state

$$0 \leq \sum_{a=1}^{L} \sum_{b=1}^{L} k(\bar{\mathbf{x}}_a^{(i)}, \bar{\mathbf{x}}_b^{(j)}) \leq \sum_{a=1}^{m} \sum_{b=1}^{m} k(\mathbf{x}_a, \mathbf{x}_b), \tag{20}$$

which is true because the summation to m terms contains all of the elements in the summation to L terms, as well as other (nonnegative) elements. Using this, we can bound (19)

by

$$\begin{aligned}
\mathbb{E}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})^2\right] &\leq \sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \left(\frac{1}{L^2} \sum_{a=1}^{m} \sum_{b=1}^{m} k(\mathbf{x}_a, \mathbf{x}_b) \right)^2 \frac{1}{\binom{m}{L}^2} \\
&= \frac{m^4}{\binom{m}{L}^2 L^4} \sum_{i=1}^{\binom{m}{L}} \sum_{j=1}^{\binom{m}{L}} \left(\frac{1}{m^2} \sum_{a=1}^{m} \sum_{b=1}^{m} k(\mathbf{x}_a, \mathbf{x}_b) \right)^2 \\
&= \frac{m^4}{\binom{m}{L}^2 L^4} \binom{m}{L}^2 k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2 \\
&= \frac{m^4}{L^4} k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2. \qquad (21)
\end{aligned}$$

Therefore, with (21), (18) can be written as

$$\text{Var}\left[k_{\text{set}}(\bar{\mathbf{X}}, \bar{\mathbf{Y}})\right] \leq \frac{m^4}{L^4} k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2 - k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2 = \left(\frac{m^4}{L^4} - 1 \right) k_{\text{set}}(\mathbf{X}, \mathbf{Y})^2 \qquad (22)$$

which concludes this proof. ■

The restriction $k(\mathbf{x}, \mathbf{x}') \geq 0$ is satisfied by many standard covariance kernels (such as the Gaussian, the Matérn family and the multiquadric) as well as some more interesting choices (such as the Wendland or Wu families of compactly supported kernels). It does, however, exclude some oscillatory kernels such as the Poisson kernel as well as kernels defined implicitly which may have an oscillatory behavior. More discussion on different types of kernels and their properties can be found in the kernel literature Fasshauer and McCourt (2015).

Appendix D. Experiments

D.1 Baselines for All Experiments

Vector A standard BO is performed over a md-dimensional space where, at the nth step, the available data $\mathfrak{X}_n \in {\mathcal{X}_{\text{set}}}^n$ is vectorized to $[\mathbf{x}_1, \ldots, \mathbf{x}_n]$ for $\mathbf{x}_i \in \mathbb{R}^{md}$ with associated function values. At each step, the vectorized next location $\mathbf{x}_{n+1}$ is converted into a set $\mathbf{X}_{n+1}$.

Split Individual BO strategies are executed on the m components comprising $\mathcal{X}$. At the nth step, the available data $\mathfrak{X}_n \in {\mathcal{X}_{\text{set}}}^n$ is decomposed into m sets of data, the ith of which consists of $[\mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_n^{(i)}]$ with associated data. The m vectors produced during each step of the optimization are then collected to form $\mathbf{X}_{n+1}$ at which to evaluate f.

D.2 Baselines for Initialization of Clustering Methods

Random This baseline randomly draws k points from a compact space $\subset \mathbb{R}^d$.

Data This baseline randomly samples k points from a dataset $\mathcal{P}$. It is widely used in initializing a clustering algorithm.

***(k-means only)* k-means++** This is a method for k-means clustering with the intuition that spreading out initial cluster centers is better than the "Data" baseline (see (Arthur and Vassilvitskii, 2007) for the details).

***(GMM only)* k-means** This baseline sets initial cluster centers as the results of k-means clustering.

D.3 Synthetic Functions

We define our synthetic functions as

Synthetic 1 We consider $d = 1$, $m = 20$ and a g function which is a simple periodic function

$$g(\mathbf{x}) = \sin(2\|\mathbf{x}\|_2) + |0.05\|\mathbf{x}\|_2|. \tag{23}$$

Synthetic 2 We consider $d = 2$, $m = 20$ and a g function which is the summation of probability density functions

$$g(\mathbf{x}) = -\sum_{i=1}^{8} p(\mathbf{x}; \mu_i, \Sigma_i) \tag{24}$$

where p is the normal density function with μ_i depicted in Figure 1 and $\Sigma_i = \mathsf{I}_2$.

D.4 Initialization of Clustering Methods

The function of interest in the k-means clustering setting is the *converged* clustering residual $k\text{-means}(\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}) = \sum_{i=1}^{N} \sum_{j=1}^{k} w_{ij} \|\mathbf{p}_i - \mathbf{c}_j\|_2^2$, where $\{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ is the set of proposed initial cluster centers, $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ is the set of converged cluster centers (Lloyd, 1982), and w_{ij} are softmax values from the pairwise distances. Here, the fact that $\mathbf{c}_j$ is a function of $\mathbf{X}$ and $\mathcal{P}$ is omitted for notational simplicity. The set of converged cluster centers is determined through an iterative strategy which is highly dependent on the initial points $\mathbf{X}$ to converge to effective centers. As shown in Figure 2, our method with $L = 1$ shows the best performance compared to other baselines.

In contrast to k-means clustering, the GMM estimates parameters of Gaussian distributions and mixing parameters between the distributions. Because it is difficult to minimize negative log-likelihood of the observed data, we fit the GMM using expectation-maximization (EM) algorithm (Dempster et al., 1977). Similarly to k-means clustering, this requires initial guesses $\mathbf{X}$ to converge to cluster centers $\{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$.

As shown in Figure 2, we conduct the experiments to initialize the mixture of Gaussians. Our method shows better performance than other baselines except the "Vector" baseline. Because the "Vector" baseline finds an optimal set under the assumption that we know the structure of sets, it does not directly match to the problem interested in this work, though it can be considered as the baseline.