# Bayesian optimization on non-conventional search spaces

Oh, C.

**Publication date**
2023

**Document Version**
Final published version

Bayesian Optimization on Non-conventional Search Spaces Changyong Oh

# Bayesian Optimization on Non-conventional Search Spaces

Changyong Oh

# Bayesian Optimization on Non-conventional Search Spaces

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op donderdag 15 juni 2023, te 11.00 uur

door Chang Yong Oh
geboren te Daegu

# SUMMARY

Thanks to its high sample efficiency, BO has been popular and successful in high-cost design problems. Nonetheless, the application of BO in the literature has been restricted to low-dimensional Euclidean spaces. Along with the ever-increasing complexity and diversity of design problems, the necessity of effective BO in various spaces is increasing. In response to such demand, in this thesis, we propose BO on spaces other than low-dimensional Euclidean ones to broaden the applicability of BO. Specifically, motivated by the successes of BO with the Gaussian process (GP) surrogate model on low-dimensional Euclidean spaces, we focus on BO with the GP surrogate model.

Our contributions are as follows

- We propose Bayesian optimization on high-dimensional Euclidean spaces, *BOCK* (Chapter 3). To mitigate the excessive exploration of high-dimensional BO, we use the cylindrical transformation that makes BO focus more on the promising region, i.e. the center of the search space. We demonstrate that BOCK achieves competitive performance on high-dimensional problems up to 500 dimensions without making structural assumptions on the objective.

- We propose Bayesian optimization on combinatorial spaces with ordinal and categorical variables, *COMBO* (Chapter 4). Representing a combinatorial space with a graph, called the combinatorial graph, we propose the ARD diffusion kernel that efficiently and scalably models the smoothness of functions on a combinatorial space. The ARD diffusion kernel equips BO with a variable selection mechanism, which helps improve the sample efficiency. We demonstrate that *COMBO* exhibits superior sample efficiency with scalability up to a combinatorial problem with $2^{60}$ choices.

- To model dependence between different types of variables, we propose *frequency modulation* (Chapter 5) – modulating the frequency of the ARD diffusion kernel (Chapter 4) with distances of continuous variables. We find out that BO is degraded without an additional condition – the similarity measure behavior of the kernel – and thus provide a sufficient condition to guarantee it. We show that, with the similarity measure behavior, BO with the GP surrogate model can achieve high sample efficiency on mixed-variable spaces.

- We propose a batch acquisition method applicable to permutation spaces, *LAW* (Chapter 6). To alleviate the difficulty of the joint optimization of multiple permutations, we adapt Determinantal point processes (DPP) based approach. In *LAW*, diversity captured by DPP is augmented with the acquisition weights for quality. We analyze the effect of weights on the regret bound in *LAW*. Empirically, we demonstrate that considering quality enables the batch acquisition method to scale well to large batch sizes.

- We show the potential of BO for combinatorial optimization problems in chip design. We adapt *LAW* for the macro placement of a practical scale (Chapter 7). We demonstrate that BO significantly outperforms simulated annealing and performs competitively to methods specialized to a specific objective. The experiment supports that BO can be a competitive option for combinatorial problems in chip design when the objective is expensive but close to the ultimate chip performance metric.

# SAMENVATTING – SUMMARY IN DUTCH

Dankzij zijn hoge sample-efficiëntie is BO populair en succesvol bij ontwerpproblemen met hoge kosten. Toch is de toepassing van BO in de literatuur beperkt tot laag-dimensionale Euclidische ruimtes. In samenhang met de steeds toenemende complexiteit en diversiteit van ontwerpproblemen neemt de behoefte naar effectieve BO in alternatieve ruimtes toe. Als antwoord hierop stellen we in dit proefschrift BO-methodes voor die functioneren in ruimtes die anders zijn dan laag-dimensionale Euclidische ruimtes. In het bijzonder, gemotiveerd door de successen van BO met het Gaussische proces (GP) surrogaatmodel op laag-dimensionale Euclidische ruimtes, richten we ons op BO met het GP-surrogaatmodel.

Onze bijdragen zijn als volgt:

- We introduceren BOCK (Hoofdstuk 3) voor hoog-dimensionale Euclidische ruimtes. Om de overmatige verkenning van hoog-dimensionale BO te vermijden gebruiken we de cilindrische transformatie, waardoor BO zich meer richt op de veelbelovende regio, d.w.z. het centrum van de zoekruimte. We demonstreren dat BOCK competitieve prestaties behaalt op hoog-dimensionale problemen tot wel 500 dimensies zonder enige structurele aannames te maken over het optimalisatie doel.

- We introduceren COMBO (Hoofdstuk 4) voor Bayesiaanse optimalisatie op combinatorische ruimten met ordinale en categorische variabelen. Door een combinatorische ruimte weer te geven als een graaf, de combinatorische graaf, introduceren we de ARD-diffusiekernel die efficiënt en schaalbaar de gladheid van functies in een combinatorische ruimte modelleert. De ARD-diffusiekernel rust BO uit met een selectiemechanisme voor variabelen, wat zorgt voor een betere sample-efficiëntie. We demonstreren dat COMBO een superieure sample-efficiëntie heeft met schaalbaarheid tot een combinatorisch probleem met 260 keuzes.

- Om de afhankelijkheid tussen verschillende soorten variabelen te modelleren, introduceren we frequentiemodulatie (Hoofdstuk 5), wat de frequentie van de ARD-diffusiekernel (Hoofdstuk 4) moduleert met afstanden van continue variabelen. We observeren dat de prestaties van BO worden gedegradeerd zonder een aanvullende voorwaarde – namelijk het gedrag van de gelijkenismaat van de kernel – en introduceren een voorwaarde die dit

garandeert. We laten zien dat BO met het GP-surrogaatmodel een hoge sample-efficiëntie kan bereiken op gemengde-variabele ruimten.

- We stellen een batch-acquisitiemethode voor die toepasbaar is op permutatieruimten, LAW (Hoofdstuk 6). Om de complexiteit van de gezamenlijke optimalisatie van meerdere permutaties te verminderen, passen we de methode van determinantale puntprocessen (DPP) aan. In LAW wordt diversiteit vastgelegd door DPP geschaald met de acquisitiegewichten voor kwaliteit. We analyseren het effect van gewichten op de spijtgrens in LAW. We laten empirisch zien dat het in acht nemen van de kwaliteit ervoor zorgt dat de batch-acquisitiemethode kan worden opgeschaald naar substantiële batchgroottes.

- We tonen het potentieel van BO aan voor combinatorische optimalisatieproblemen bij chipontwerp. We passen LAW aan voor de macroplaatsing op een praktische schaal (Hoofdstuk 7). We demonstreren dat BO aanzienlijk beter presteert dan simulated annealing en competitief is met methodes die gespecialiseerd zijn in een specifiek doel. Het experiment ondersteunt dat BO een competitieve optie kan zijn voor combinatorische problemen bij het ontwerpen van chips wanneer het optimalisatiedoel duur is, maar dicht bij de ultieme maatstaf voor chipprestaties ligt.

# LIST OF PUBLICATIONS

This thesis is based on the following publications:

**Changyong Oh**, Efstratios Gavves, and Max Welling. "BOCK: Bayesian optimization with cylindrical kernels." *ICML, 2018*.

**Changyong Oh**, Jakub M. Tomczak, Efstratios Gavves, and Max Welling. "Combinatorial Bayesian Optimization using the Graph Cartesian Product." *NeurIPS, 2019*.

**Changyong Oh**, Efstratios Gavves, and Max Welling. "Mixed Variable Bayesian Optimization with Frequency Modulated Kernels." *UAI, 2021*.

**Changyong Oh**, Roberto Bondesan, Efstratios Gavves, and Max Welling. "Batch Bayesian Optimization on Permutations using Acquisition Weighted Kernels." *NeurIPS, 2022*.

**Changyong Oh**, Roberto Bondesan, Dana Kianfar, Rehan Ahmed, Rishubh Khurana, Payal Agarwal, Romain Lepert, Mysore Sriram, and Max Welling. "Bayesian Optimization for Macro Placement." *ICML Adaptive Experimental Design and Active Learning in the Real World Workshop, 2022*

Ideas, text, figures, and experiments originate in majority from the first author. All other authors had important advisory roles, helped with writing a number of individual sections of the above listed papers.

The author has further contributed to the following publications:

**Changyong Oh**, Kamil Adamczewski, and Mijung Park. "Radial and Directional Posteriors for Bayesian Deep Learning." *AAAI, 2020*.

Shuai Liao, Efstratios Gavves, **Changyong Oh**, and Cees Snoek. "Quasibinary Classifier for Images with Zero and Multiple Labels." *ICPR, 2020*.

# CONTENTS

# 1 INTRODUCTION

This chapter provides an overview of the research questions we address in this thesis. To motivate our research questions, we first show the ubiquity of design problems in Sec 1.1. Then we present why Bayesian optimization (BO) has been successful in such problems and what needs to be improved to bring these successes to broader classes of design problems. In Sec. 1.2, we specify research questions. Then we discuss the motive of each research question and our strategy to deal with it. In Sec. 1.3, the thesis structure is provided.

## 1.1 DESIGN OF EXPERIMENTS AND BAYESIAN OPTIMIZATION

In many scientific and engineering procedures, influencing factors are selected to obtain the best possible output. To name a few,

- In machine learning, hyperparameters, e.g. learning rate, neural architecture, etc. have a huge impact on model performance, and they are deliberately tuned to optimize the generalization performance. (Y. Chen, A. Huang, et al., 2018; B. Zhang et al., 2021; Zoph, Vasudevan, et al., 2018).

- In chemistry, experiments are subject to many conditions, such as temperature, acidity, constants in chemical reaction models, parameters of chemical devices, etc. Such factors are deliberately configured to optimize the targets, e.g. selectivity and reproducibility. (Boelrijk et al., 2021; Hase et al., 2018).

- In chip design, components of integrated circuits (memory blocks, logic gates) interact in a complex way. These components are optimally placed and wired to produce high-performing and robust chips. (Lyu et al., 2018; Mirhoseini et al., 2021; S. Zhang et al., 2019)

- Even when developing a cookie recipe, the amount of ingredients can be optimized to find a highly-rated cookie recipe (Solnik et al., 2017).

Even though such problems of selecting an optimal setting of influencing factors occur in various disciplines and have different appearances, many of them have common characteristics below.

- A functional form of the relation between influencing factors and target output is unknown. Consequently, the gradient is not available.
- The measured value of the target output is corrupted by noise. Thus repeated evaluations for the same factor value may differ from each other.
- Evaluating a target output value for a given factor value is expensive in terms of time, compute resources, etc.

For the problems with the above characteristics, Bayesian optimization (BO) has emerged as a competitive option (Brochu et al., 2010; Shahriari et al., 2015). BO is a black-box function optimizer. It only requires evaluation at a given point, not demanding additional information such as a gradient.[1] Among various black-box function optimizers, BO is famous for its high sample efficiency. It is capable of finding near-optimum solutions with fewer number of evaluations. Thanks to the high sample efficiency, BO has successfully demonstrated its competitiveness in design problems with the above characteristics, for instance, AlphaGo (Silver et al., 2017) where BO increased the win rate from 50.0% to 66.5% (Y. Chen, A. Huang, et al., 2018).

The competitiveness of BO originates from two main components of BO, the *surrogate model* and the *acquisition function*. BO begins by choosing the search space on which it tries to find an optimum. Then the *surrogate model* is constructed to probabilistically model the objective to optimize. The surrogate model provides not only prediction of the objective on unevaluated points but also the uncertainty of its prediction. As a model of the objective, the surrogate model enables BO to have a global view of the objective. In contrast to methods using local search, the global view allows search beyond local neighborhoods and thus gives BO global optimization capability. Based on the predictive distribution provided by the surrogate model, another function on the same search space is defined. This is the *acquisition function*, which assesses how informative each point in the search space is for the optimization of the objective. The acquisition function considers not only the chance to make a prompt improvement (exploitation) but also its influence on the optimization afterward (exploration). Instead of optimizing the expensive-to-evaluate original objective, a cheap substitute (acquisition function) is optimized to find the next promising point. With the combination of the surrogate model and the acquisition function, BO globally and efficiently optimizes the objectives with the aforementioned characteristics.

1 BO does not require gradients. However, if the gradient is available, BO can utilize it to actively use all accessible information as in (Wu, Poloczek, et al., 2017).

When the objective is noisy and not differentiable, other alternatives are applicable, for instance, genetic algorithms (Mitchell, 1998), evolutionary strategy (N. Hansen and Ostermeier, 1996), etc. However BO stands out, especially when the evaluation is expensive as in the above examples, thanks to its high sample efficiency. Such high sample efficiency is attributed to the well-calibrated uncertainty of the surrogate model (Pleiss et al., 2018; Shahriari et al., 2015). Gaussian processes (GPs) are famous for their principled and well-calibrated uncertainty (Williams and Rasmussen, 2006), and the GP surrogate model has been shown to outperform other surrogate models when the GP surrogate model is applicable. (Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014)

### *Limitations of Existing approaches*

As increasingly many success stories of BO are reported, the demand for the utilization of BO in novel scenarios is growing. There are many cases where vanilla BO is either inapplicable or ineffective. The optimization may involve several objectives that are optimized simultaneously – multi-objective BO. Sometimes the optimization needs to be performed in a few-shot setting with not identical but related optimization problems – multi-task, contextual, transfer BO. In practice, the complexity of design problems keeps increasing, and thus more factors need to be optimized – high-dimensional BO. Also, many design problems consist of both continuous and combinatorial variables – combinatorial BO and mixed-variable (hybrid) BO. Among the above ones, the focus of this thesis is high-dimensional BO and combinatorial BO. We briefly discuss the difficulty of each of them.

HIGH-DIMENSIONAL BO In high-dimensional spaces, it is infeasible to collect evaluations enough to cover the search space. The surrogate model is fitted with insufficient data. Thus the point predictions are inaccurate in most areas of the search space. Moreover, the uncertainty of the prediction is so high that its difference at different areas of the search space is negligible. Therefore, in high-dimenionsional BO, it is difficult for the surrogate model to produce actionable information in most areas of the search space. In practice, BO users usually select a subset of factors to optimize with the rest of the factors fixed to manually chosen values, which leads to a suboptimal result.

COMBINATORIAL BO Despite abundant combinatorial choices in design problems, BO on combinatorial spaces is under-explored. A simple approach is to embed combinatorial variables into Euclidean spaces. However, such an approach is not only likely to be suboptimal (Garrido-Merchán and D. Hernández-Lobato, 2020) but also inapplicable to non-tabular data, e.g., molecules. Besides, existing methods developed for Euclidean spaces are not transferable to combinatorial spaces since many intuitive structures in Euclidean spaces such as distance, angle, etc., lack in combinatorial spaces.

## 1.2 RESEARCH QUESTIONS

Building on the many successes of BO in low-dimensional Euclidean spaces, researchers and practitioners wish and try to bring the high sample efficiency of BO into the problems on spaces other than low-dimensional Euclidean spaces. In response to such demand, our general goal in this thesis is to answer the question below.

***Can we develop efficient BO methods on search spaces other than low-dimensional Euclidean spaces?***

Under the general question, we focus on improving the sample efficiency of BO by utilizing the well-calibrated uncertainty of GPs. We propose BO with the GP surrogate model on various types of search spaces by tackling difficulties arising from each search space.

**Research Question 1** *What causes the excessive exploration in high-dimensional Euclidean spaces? And how can we mitigate the excessive exploration for effective high-dimensional BO?* (Chapter 3)

As the dimension increases, the volume is dominated by regions near the boundary. Moreover the point with the highest uncertainty is always on the boundary. Thus, in high-dimensional Euclidean spaces, there is a plethora of points waiting for evaluation due to their high uncertainty.

Even worse, such drive to evaluate points near the boundary conflicts with a common practice of BO. In BO, to the best of users' prior knowledge, the search space is set to contain a near-optimal point around the center of the search space. Even after many evaluations, there is still huge unprobed area near the boundary. Thus, it is unlikely to evaluate a point near the center in high-dimensional spaces.

To reconcile the conflict, we propose a geometric transformation of the search space. Our proposed transformation, the cylindrical transformation, shrinks the volume near the boundary and expands the volume near the center. Furthermore, the cylindrical transformation induces the effect of restraining high uncertainty near the boundary. With this transformation, we propose *BOCK*, BO for high-dimensional Euclidean spaces. We demonstrate that the cylindrical transformation effectively mitigates the excessive exploration, and that BOCK can achieve impressive performance on high-dimensional problems – up to 500 dimension – without relying on assumptions on the structure of the objective.

**Research Question 2** *How can we define smooth surrogate models and acquisition functions on combinatorial spaces? And can we develop a flexible GP surrogate model for BO in large combinatorial spaces?* (Chapter 4)

In order to define the smoothness of functions on combinatorial spaces, we propose a graph representation called the *combinatorial graph*. On the combinatorial graph, each vertex corresponds to a combinatorial value, and edges represent relations between combinatorial values. With the combinatorial graph, functions on a combinatorial space can be interpreted as functions on a graph, i.e. graph signal (Ortega et al., 2018). We define the smoothness of functions on combinatorial spaces as the smoothness of the graph signal.

We further equip the above notion of smoothness with computational efficiency and modeling flexibility. We set the combinatorial graph to be decomposable into small graphs i.e. the graph Cartesian product of small graphs (Hammack et al., 2011). Such decomposability enables efficient computation of the diffusion kernel (R. I. Kondor and Lafferty, 2002; Smola and R. Kondor, 2003) quantifying the smoothness of graph signals. Using the decomposability, we propose the ARD diffusion kernel to equip with a larger modeling capacity.

Compared with the existing surrogate model using Bayesian linear regression (Baptista and Poloczek, 2018), our GP surrogate model can model arbitrarily high-order interactions among variables even with faster computation. We also demonstrate that GP using the ARD diffusion kernel has an improved modeling capacity and that BO with the ARD diffusion kernel, *COMBO*, scales well to large combinatorial spaces such as one represented by the combinatorial graph with $2^{60} \approx 1.15 \times 10^{18}$ vertices.

**Research Question 3** *How can we model dependencies between different types of variables in kernels? And do we need conditions for kernels other than positive definiteness?* (Chapter 5)

In order to model dependence between different types of variables in GPs, we propose a kernel construction method called *frequency modulation*. In the frequency modulation, the frequency in the ARD diffusion kernel for combinatorial variables is modulated by the distance of continuous variables. This construction introduces coupling between different types of variables.

While developing the frequency modulation, we discover that the positive (semi-)definiteness of kernels does not provide any guarantee on the natural behavior of the kernels – the *similarity measure behavior*, i.e., the more similar two points are, the higher their kernel value is. We demonstrate that violation of the similarity measure behavior severely degrades the performance in BO and regression tasks. Therefore, we provide a sufficient condition to guarantee the similarity measure behavior of the kernels from the frequency modulation.

On many BO and GP regression tasks, we show that the similarity measure behavior is another crucial property for robust and stable performance. We demonstrate that, also in mixed-variable spaces, the well-calibrated uncertainty of GP enables BO to achieve high sample efficiency, for example, in the joint optimization of neural architecture and learning hyperparameters.

**Research Question 4** *Can we develop a batch acquisition method applicable to permutation spaces? And how can we maintain the quality of the batch acquisition method as the batch size increases?* (Chapter 6)

In super-exponentially growing[2] permutation spaces, sequential BO on a large permutation space is likely to require unaffordable wall-clock runtime. Therefore, we investigate batch acquisition on permutation spaces. For more economical use of each evaluation, we aim for a batch acquisition method that makes BO keep its performance for large batch sizes.

We choose the Determinantal point process (DPP) (Kulesza and Taskar, 2012) as a feasible framework for batch acquisition on permutation spaces. DPP is not only widely used for diversity modeling, but it also has a property called submodularity (Kulesza and Taskar, 2012) which makes the optimization of a function on multiple permutations tractable (Buchbinder et al., 2014; Nemhauser et al., 1978).

Retaining the strengths of DPPs, we augment DPPs with the acquisition value to take into account the quality of each point in the batch. Theoretically, we analyze the cumulative regret taking into account the effect of the quality. We demonstrate that the batch acquisition considering diversity and quality significantly outperforms the one considering diversity only.

**Research Question 5** *Can BO be an effective method for combinatorial optimization problems in chip design? And can it be efficient enough to scale to problems of a practical scale?* (Chapter 7)

Recently, BO has been applied to problems arising in chip design, most of which focus on continuous optimization (Lyu et al., 2018; S. Zhang et al., 2019). However, there are numerous combinatorial optimization problems in chip design (A. Kahng et al., 2011). Among others, we focus on the macro placement in which components with a large area, e.g., memory and IP blocks, are placed on a chip canvas (Shahookar and Mazumder, 1991).

While applying BO to the macro placement of a practical scale, two computational challenges arise. First, the macro placement is subject to many constraints and the computation of the feasibility with respect to such constraints takes up nonnegligible time. Second, for a large number of macros, batch acquisition can be time-consuming, which prevents the use of large batch sizes.

We adapt LAW (Chapter 6) to handle such issues. We incorporate a parallel version of the feasibility check into the acquisition function optimization and propose a parallel heuristic to accelerate the batch

2 Let $S_N$ be the symmetric group (permutation space) of length $N$. Super-exponential growth is formally expressed as $\lim_{N\to\infty} \frac{|S_N|}{c^N} = \infty$ for all $c \in (0, \infty)$.

acquisition. We test the adapted LAW on the macro placement benchmarks and compare it with other black-box optimizers and methods specialized to a specific form of the objective.

## 1.3 THESIS STRUCTURE

In this thesis, we propose BO methods for search spaces other than low-dimensional Euclidean spaces. We demonstrate that the well-calibrated uncertainty of GPs improves sample efficiency of BO on search spaces other than low dimensional Euclidean spaces and that the principled formulation of GP uncertainty opens opportunities to improve the efficiency of batch acquisition.

We cover five different types of search spaces in the thesis.

| | |
|---|---|
| Chapter 3 | High dimensional Euclidean spaces |
| Chapter 4 | Combinatorial spaces |
| Chapter 5 | Mixed-variable spaces |
| Chapter 6 | Permutation spaces (Symmetric group) |
| Chapter 7 | Constrained product spaces of two permutation spaces |

Between the two components of BO, the surrogate model and the acquisition function, in each chapter, we focus on one component. In the first three chapters, we focus on devising effective kernels for GP surrogate model on each search space. In the last two chapters, we focus on effective batch acquisition in terms of both the number of samples and the wall-clock runtime.

The rest of the thesis is structured as follows. In Chapter 2, we briefly overview BO, GPs and relevant topics. In Chapter 3, we present BOCK, BO for high dimensional Euclidean spaces. In Chapter 4, we present COMBO, BO for combinatorial spaces with categorical and ordinal variables. In Chapter 5, we present frequency modulation, a novel kernel combination method with the application to mixed-variable BO. In Chapter 6, we present LAW, a batch acquisition method applicable to permutation spaces. In Chapter 7, we extend LAW for the application of BO on a combinatorial optimization problem appearing in chip design. Then, we conclude the thesis with a discussion on the limitations of the proposed methods and the future works which we believe are promising.

# 2 BACKGROUND

This chapter explains prerequisite knowledge of the research in this thesis. In Sec. 2.1, we explain the basics of Bayesian optimization (BO), the workflow of Bayesian optimization (BO) and two key components – the surrogate model and the acquisition function. Among others, the surrogate model of our interest, Gaussian processes, is explained in more detail in Sec. 2.2. In Sec. 2.3, the central notions of general sequential decision-making – exploration and exploitation – are elaborated in the context of BO. Lastly, in Sec. 2.4, we compare and contrast BO to relevant topics to better position our research from a broader perspective.

## 2.1 BAYESIAN OPTIMIZATION

Bayesian optimization (BO) aims at finding a global minimum of the objective $f_{obj}$ on the *search space* $\mathcal{X}$

$$\mathbf{x}_{opt} = \underset{\mathbf{x}\in\mathcal{X}}{\operatorname{argmin}}\, f_{obj}(\mathbf{x}). \tag{2.1}$$

Distinctively, BO typically targets the optimization with the constraints below

- Functional form and thus gradients of the objective $f_{obj}$ are not accessible.
- Evaluating $f_{obj}$ at a given point $\mathbf{x}$ is expensive.
- The evaluation is noisy $f_{obj}(\mathbf{x}) = f_{ground\ truth}(\mathbf{x}) + \epsilon$.

Due to the inaccessibility to the gradient of the objective, BO takes a black-box function optimization approach where it relies on the evaluation of the objective $f_{obj}$ at given points.[3] Due to the high cost of evaluations, BO puts emphasis on high sample efficiency. BO tries to find $\mathbf{x}^*$ whose evaluation $f_{obj}(\mathbf{x}^*)$ is as close to the optimum $\min_{\mathbf{x}\in\mathcal{X}} f_{obj}(\mathbf{x})$ as possible with fewer evaluations.

In pursuit of a sample-efficient global optimizer, BO takes a model-based approach. In contrast with model-free approaches such as simulated annealing (Van Laarhoven and Aarts, 1987) and genetic algorithm (Mitchell, 1998), BO builds a probabilistic model approximating

3 When the gradient of the objective is available, certain BO methods leverage this information to improve the performance (Wu, Poloczek, et al., 2017)

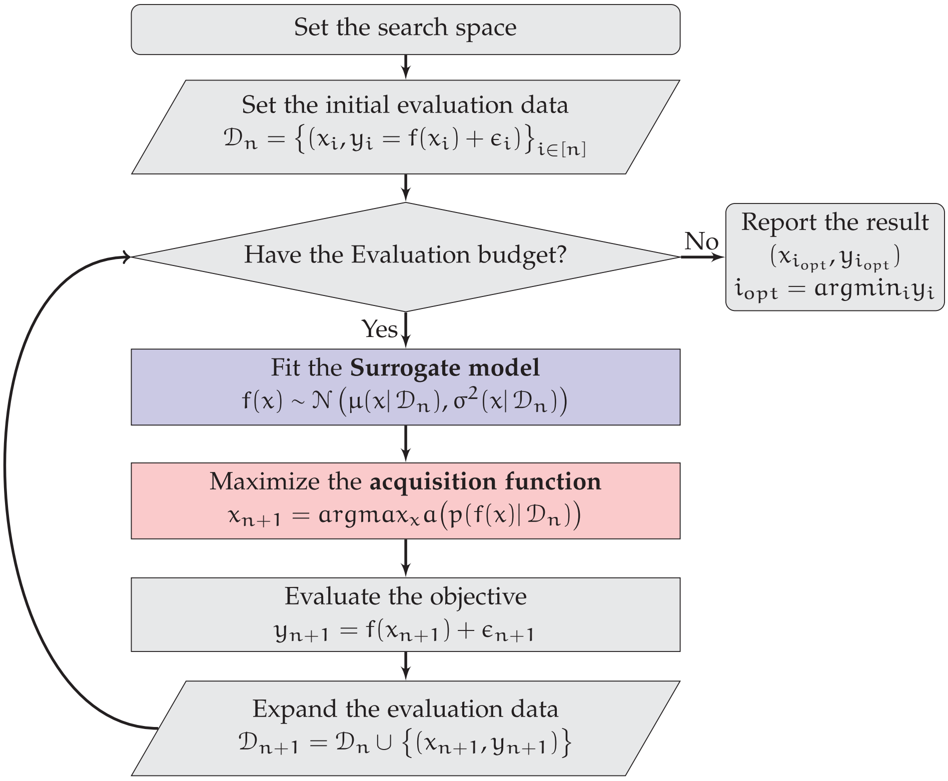

Set the search space
Set the initial evaluation data
$\mathcal{D}_n = \{(x_i, y_i = f(x_i) + \epsilon_i)\}_{i\in[n]}$
Have the Evaluation budget?
No
Report the result
$(x_{i_{opt}}, y_{i_{opt}})$
$i_{opt} = \text{argmin}_i y_i$
Yes
Fit the **Surrogate model**
$f(x) \sim \mathcal{N}(\mu(x|\mathcal{D}_n), \sigma^2(x|\mathcal{D}_n))$
Maximize the **acquisition function**
$x_{n+1} = \text{argmax}_x a(p(f(x)|\mathcal{D}_n))$
Evaluate the objective
$y_{n+1} = f(x_{n+1}) + \epsilon_{n+1}$
Expand the evaluation data
$\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{(x_{n+1}, y_{n+1})\}$

Figure 2.1: Bayesian optimization workflow

the objective, called the surrogate model $\mathcal{M}$ conditioned on the previously evaluated data $\mathcal{D}_{n-1} = (\mathbf{x}_i, y_i)_{i=1,\cdots,n-1}$, $y_i = f_{obj}(\mathbf{x}_i) + \epsilon_i$ and $\epsilon_i \sim \mathcal{N}(0, \sigma^2_{obs})$. The surrogate model is the primary channel leveraging the evaluation data $\mathcal{D}_{n-1}$ in BO and the prediction made by the surrogate model provides a global view of the objective.

In addition to the accurate prediction, another key requirement for the surrogate model is well-calibrated uncertainty. Due to the limited size of evaluation data $\mathcal{D}_{n-1}$, it is unlikely that the surrogate model makes fairly accurate predictions on the entire search space $\mathcal{X}$. Well-calibrated uncertainty complements the predictions by informing how trustworthy the predictions are, which, in turn, guides the optimization.

The way that the surrogate model steers the optimization is via balancing between two modes of information utilization, exploitation and exploration. Exploitation is to evaluate the point of low objective but not high predictive value which is conferred with uncertain predictions, expecting the probable improvement of the optimization process. Exploration is to evaluate the point of high uncertainty, which not only aids optimization later to be based on more reliable information, but also helps to probe unexplored areas. If lucky, exploration sometimes happens to evaluate the point making a huge improvement in the optimization.

Mathematically, the surrogate model takes the data of points evaluated, $\mathcal{D}$, as an input and generates predictive distributions on all $\mathbf{x} \in \mathcal{X}$, which is typically in the form of the predictive mean $\mu_{n-1}(\mathbf{x}) = \mu(\mathbf{x} \,|\, \mathcal{D}_{n-1})$ and the predictive variance $\sigma_{n-1}(\mathbf{x}) = \sigma(\mathbf{x} \,|\, \mathcal{D}_{n-1})$.

Quantitatively balancing between exploitation and exploration is the responsibility of another key component of BO, the acquisition function $\mathbf{a}(\mathbf{x})$. By using the probabilistic prediction made by the surrogate model $\mathcal{M}$, the acquisition function represents a good balancing score between exploitation and exploration as a single number. Many interesting acquisition functions have been proposed, and all reflect the intuition that points with low predictive mean (in minimization) and high predictive uncertainty have high acquisition values.

Some acquisition functions only summarize the predictive distribution at $\mathbf{x}$ to compute the acquisition value $\mathbf{x}$. Such local acquisition functions include probability of improvement (PI) (Kushner, 1964) and expected improvement (EI) (Jones et al., 1998).

$$\begin{aligned}
\mathbf{a}_{PI}(\mathbf{x}) &= \mathbb{P}_{y \sim \mathcal{N}(\mu_{n-1}(\mathbf{x}), \sigma^2_{n-1}(\mathbf{x}))}[y \leqslant \tilde{y}_{min}] \\
\mathbf{a}_{EI}(\mathbf{x}) &= \mathbb{E}_{y \sim \mathcal{N}(\mu_{n-1}(\mathbf{x}), \sigma^2_{n-1}(\mathbf{x}))}[(\tilde{y}_{min} - y)^+]
\end{aligned}$$

where $\tilde{y}_{min} = \min_{i=1,\cdots,n-1} y_i$. Others take into account predictive distributions at other locations, which are more global such as entropy search (ES) (Hennig and Schuler, 2012) and predictive entropy search (PES) (J. M. Hernández-Lobato, M. W. Hoffman, et al., 2014)

$$\begin{aligned}
\mathbf{a}_{ES}(\mathbf{x}) =& \mathbb{H}[p(\mathbf{x}^* \,|\, \mathcal{D}_{n-1})] \\
& - \mathbb{E}_{y \sim \mathcal{N}(\mu_{n-1}(\mathbf{x}), \sigma^2_{n-1}(\mathbf{x}))}[\mathbb{H}[p(\mathbf{x}^* \,|\, \mathcal{D}_{n-1} \cup \{(\mathbf{x}, y)\})]] \\
\mathbf{a}_{PES}(\mathbf{x}) =& \mathbb{H}[\mathcal{N}(\mu_{n-1}(\mathbf{x}), \sigma^2_{n-1}(\mathbf{x}))] \\
& - \mathbb{E}_{p(\mathbf{x}^* \,|\, \mathcal{D}_{n-1})}[\mathbb{H}[p(y| \,\mathcal{D}_{n-1}, \mathbf{x}, \mathbf{x}^*]]
\end{aligned}$$

where the predictive distributions at other locations are considered indirectly via $p(\mathbf{x}^* \,|-)$.

Now the acquisition function $\mathbf{a}(\mathbf{x})$ is maximized to select the most promising point with respect to a chosen acquisition function. $\mathbf{a}(\mathbf{x})$ is a function defined on the same space $\mathcal{X}$ as the search space of the original optimization task. However, $\mathbf{a}(\mathbf{x})$ is much cheaper to evaluate than the original objective $f$ and if the search space is continuous, then $\mathbf{a}(\mathbf{x})$ is differentiable.

With these two key components of BO, the surrogate model and the acquisition function, the BO algorithm proceeds as given in Fig. 2.1.

## 2.2 GAUSSIAN PROCESSES

Different probabilistic models have been used as surrogate models of BO including random forests (Hutter et al., 2011), the tree-structured Parzen estimator (J. Bergstra, Yamins, et al., 2013; J. S. Bergstra et al., 2011), Bayesian neural networks (Springenberg et al., 2016), deep ensembles (Belanger et al., 2019; K. Swersky, Rubanova, et al., 2020), Bayesian linear models (Baptista and Poloczek, 2018; Daxberger et al., 2021; Perrone, Jenatton, et al., 2018; Snoek, Rippel, et al., 2015) and Gaussian processes (Eriksson, Pearce, et al., 2019; E. Lee et al., 2020; Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014; Zi Wang, Gehring, et al., 2018; Zi Wang, C. Li, et al., 2017).

Gaussian processes (GPs) have proven their competitiveness over others in black-box function optimization tasks (Ru, Alvi, et al., 2020; Snoek, Larochelle, et al., 2012; Springenberg et al., 2016), mainly due to their well-calibrated uncertainty.

GP is a Bayesian nonparametric method to model a function (Rasmussen, 2003). As a surrogate model in BO, it probabilistically models the objective function $f_{obj}$ whose evaluation is often noisy. GP is an infinite dimensional generalization of the multivariate Gaussian distribution. The corresponding infinite dimensional mean parameter vector and infinite dimensional covariance parameter matrix are specified by the mean function $\mathbf{m}(\mathbf{x})$ and the covariance function $\mathbf{k}(\mathbf{x}, \mathbf{x}')$. In order to define an appropriate covariance, the covariance function should satisfy the following condition, which is called positive (semi-) definite.

$$\sum_{i,j} c_i \, \mathbf{k}(\mathbf{x}_i, \mathbf{x}_j) c_j \geqslant 0 \qquad \forall c_i \in \mathbb{R} \quad \forall \mathbf{x}_i \in \mathcal{X}$$

More commonly, the covariance function is called the (positive semi-definite) kernel.

For a given mean function and a kernel, conditioned on the data $\mathcal{D} = (\mathbb{X}, \mathbf{y})$, $\mathbb{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]^T$ and $\mathbf{y} = [y_1, \cdots, y_n]^T$, the predictive distribution of $f_*(\mathbf{x}_* \,|\, \mathcal{D})$ at $\mathbf{x}_*$ is given as $\mathcal{N}(\mu(\mathbf{x}_* \,|\, \mathcal{D}), \sigma^2(\mathbf{x}_* \,|\, \mathcal{D}))$ where

$$\mu(\mathbf{x}_* \,|\, \mathcal{D}) = \mathbf{m}(\mathbf{x}_*) + \mathbf{k}(\mathbf{x}_*, \mathbb{X})[\mathbf{k}(\mathbb{X}, \mathbb{X}) + \sigma^2_{obs}\, \mathbb{I}_n]^{-1}(\mathbf{y} - \mathbf{m}(\mathbb{X}))$$
$$\sigma^2(\mathbf{x}_* \,|\, \mathcal{D}) = \mathbf{k}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*, \mathbb{X})[\mathbf{k}(\mathbb{X}, \mathbb{X}) + \sigma^2_{obs}\, \mathbb{I}_n]^{-1}\, \mathbf{k}(\mathbb{X}, \mathbf{x}_*)$$

Model predictive performance is heavily affected by the hyperparameters of GPs $\theta$, including mean function parameters, kernel parameters, and noise variance. GP hyperparameters can be chosen by optimizing the (log-)marginal likelihood.

$$\begin{aligned}\log p(\mathbf{y} \,|\, \mathbb{X}, \theta) = &-\frac{1}{2}(\mathbf{y} - \mathbf{m}(\mathbb{X}))^T\, \mathbf{k}(\mathbb{X}, \mathbb{X} + \sigma^2_{obs}\, \mathbb{I})^{-1}(\mathbf{y} - \mathbf{m}(\mathbb{X})) \\ &-\frac{1}{2}\log\det \mathbf{k}(\mathbb{X}, \mathbb{X}) - \frac{n}{2}\log(2\pi)\end{aligned}$$

Alternatively, taking full Bayesian approach, the posterior of $\theta$ can be used. For a given prior $p(\theta)$, the posterior

$$p(\theta | \mathbf{y}, \mathbb{X}) = \frac{p(\mathbf{y}|\mathbb{X}, \theta)p(\theta)}{p(\mathbf{y}|\mathbb{X})}.$$

is used to construct the predictive distribution marginalized over $\theta$

$$\mathbb{E}_{p(\theta|\mathbf{y},\mathbb{X})}[f_*(\mathbf{x}_*|\mathcal{D},\theta)] = \int \mathcal{N}(\mu(\mathbf{x}_*|\mathcal{D},\theta), \sigma^2(\mathbf{x}_*|\mathcal{D},\theta))p(\theta|\mathcal{D})d\theta$$

The posterior is approximated by MCMC (Snoek, Larochelle, et al., 2012) or variational inference (M. Titsias, 2009).

## 2.3 EXPLORATION-EXPLOITATION TRADE-OFF

Among black-box function optimization methods, BO is famous for its high sample efficiency, i.e. it finds an optimum with fewer number of evaluations.[4] Its high sample efficiency makes BO a method of choice in the situation where the objective is expensive to evaluate. The high sample efficiency of BO is attributed to the capability of balancing between exploration and exploitation (Brochu et al., 2010; Shahriari et al., 2015). In the context of BO, the exploitation is to utilize the previous evaluations conservatively, which corresponds to evaluating a point near the best evaluation in hand. On the other hand, the exploration is to make an aggressive step by trying a point in the unprobed region with high uncertainty. With exploratory moves, we can expect two scenarios. With any luck, an exploratory move may find an input better than the current best input outside the neighborhood of the current best input. Even in the worst case, an exploratory move reveals a region unlikely to have a good input by supplying the evaluation to the surrogate model in future BO rounds.

In BO, the exploration-exploitation trade-off is controlled by the uncertainty from the surrogate model and the acquisition function. Compared with other surrogate models, random forests (Hutter et al., 2011) and the tree-structured Parzen estimator (J. Bergstra, Yamins, et al., 2013; J. S. Bergstra et al., 2011), Gaussian processes (GPs) (Snoek, Larochelle, et al., 2012) provide principled and well-calibrated uncertainty (Rasmussen, 2003). The well-calibrated uncertainty of GPs enables better exploration-exploitation trade-off and, in turn, better sample efficiency (Brochu et al., 2010; Shahriari et al., 2015; Snoek, Larochelle, et al., 2012).

4 Strictly speaking, optimization methods find an optimum or a point whose evaluation is close to the evaluation of an optimum unless there is a guarantee to find an optimum.

## 2.4 RELEVANT TOPICS

### 2.4.1 Hyperparameter optimization

In machine learning communities, BO has received attention as a hyperparameter optimization method (Hutter et al., 2011; Ru, Alvi, et al., 2020; Snoek, Larochelle, et al., 2012). These days, deep learning models are becoming more complex. Accordingly, the size of the hyperparameter space far exceeds the scope of human researchers' ability to search it, and the evaluation of a configuration of hyperparameters becomes increasingly costly. Thanks to its high sample-efficiency, BO has proved its competitiveness in many deep learning hyperparameter optimization tasks (J. Bergstra, Yamins, et al., 2013; J. S. Bergstra et al., 2011; Snoek, Larochelle, et al., 2012; Snoek, Rippel, et al., 2015). Among others, BO was successfully deployed to optimize the hyperparameters of AlphaGo (Silver et al., 2017) agent training with the impressive result of improving the agent's win rate from 50% to 66.5% (Y. Chen, A. Huang, et al., 2018).

Among other hyperparameters of deep learning models, the search for an optimal architecture has been extensively studied (Elsken et al., 2019; Wistuba, Rawat, et al., 2019). The search for an architecture of deep learning models, a.k.a. neural architecture search (NAS) began with reinforcement learning (Zoph and Q. Le, 2017). As a black-box function optimization problem, NAS was also tackled by BO (Kandasamy, Neiswanger, et al., 2018; Changyong Oh, Tomczak, et al., 2019; Ru, Wan, et al., 2020). Recently, the differentiable approach which is prevalent since its inception (H. Liu et al., 2018) has becomes the mainstream of NAS. Its popularity is attributable to its scalability. BO requires the training of many neural networks, on the other hand, the differentiable approach trains a single large neural network within which many sub-architectures are included (Pham et al., 2018). Even though the differentiable approach is more promising than BO in NAS, non-differentiable sample-based approaches including BO tend to obtain better and more stable results at the cost of compute time (X. Dong, L. Liu, et al., 2021). Moreover, while joint optimization of architecture and other hyperparameters can be handled in the BO framework (Falkner et al., 2018; Changyong Oh, Gavves, et al., 2021), a differentiable correspondent is still missing.

### 2.4.2 Bandits

Bayesian optimization has been studied under different names such as Gaussian process bandit optimization (Bogunovic, Scarlett, et al., 2016; Grünewälder et al., 2010; Janz et al., 2020; Krause and Ong, 2011; Srinivas et al., 2010), kernelized bandit (Bogunovic and Krause, 2021; Chowdhury and Gopalan, 2017), etc.

There are subtle differences between the BO perspective and the bandit perspective. Works from the bandit perspective tend to take more rigorous approaches stressing theoretical guarantees on the regret. Such guarantees do not come for free. They make regularity assumptions on the objective such as Lipschitz, small RKHS norm, etc. (De Freitas et al., 2012; Srinivas et al., 2010) which are oftentimes too complex to validate in practice. On the other hand, BO tends to focus more on heuristically designed methods emphasizing extensive empirical validation on complex objectives (Eriksson, Pearce, et al., 2019; Snoek, Rippel, et al., 2015).

The way they handle the surrogate model slightly differs. Methods from the bandit framework often use GPs with fixed kernel hyperparameters (Chowdhury and Gopalan, 2017). However, in BO, fitting the kernel hyperparameters is a crucial step to improve sample efficiency, which is almost always adopted.

The seminal work of (Srinivas et al., 2010) shows that the cumulative regret from GP-UCB acquisition function can be analyzed in the bandit framework. Beginning with GP-UCB, it is shown that other acquisition functions can enjoy similar theoretical analysis, for example, EI (Gupta et al., 2022; V. Nguyen, Gupta, et al., 2017) and TS (Kandasamy, Krishnamurthy, et al., 2018). Although the ultimate objective of BO is usually the simple regret, the analysis of BO usually finds a cumulative regret bound as an intermediate step (Contal et al., 2013; Desautels et al., 2014; Kandasamy, Krishnamurthy, et al., 2018) since the simple regret is bounded above by the cumulative regret.

There is no clear-cut distinction between BO and bandits, but the difference is more of a tendency of the focus under different names. From the bandit perspective, BO can be described as a heuristic structured bandit where the performance takes precedence over the theoretical guarantee.

### 2.4.3 Reinforcement Learning (RL)

In RL, the goal is to find a policy taking an optimal sequence of actions in a given environment (Sutton and Barto, 2018). Since a bandit is a simplified RL agent[5] (Sutton and Barto, 2018), on problems that can be reformulated without state transition, BO can be adopted in place of RL.

Examples include the neural architecture search (NAS) (Elsken et al., 2019; Zoph and Q. Le, 2017; Zoph, Vasudevan, et al., 2018), drug discovery (Gómez-Bombarelli et al., 2018; Korovina et al., 2020; Popova et al., 2018; Pyzer-Knapp, 2018), chip design (R. Cheng and Yan, 2021; Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022;

5 A contextual bandit can be regarded as reinforcement learning where states are given solely by the environment not affected by the agent's actions. Bandit without context can be regarded as stateless reinforcement learning.

Mirhoseini et al., 2021; Changyong Oh, Bondesan, et al., 2022), etc. In Zoph and Q. Le, 2017, NAS was initially formulated as a RL problem where the architecture specification is a sequence of actions imposing constraints on possible next actions. Later, many BO approaches were developed for NAS by reformulating architecture specification as a combinatorial choice (Kandasamy, Neiswanger, et al., 2018; Ru, Wan, et al., 2020).

In drug discovery, desirable properties of molecules are optimized. In RL (H. Chen et al., 2018; Popova et al., 2018), molecules are searched by gradually constructing valid molecules. In BO (Gómez-Bombarelli et al., 2018; Korovina et al., 2020; Pyzer-Knapp, 2018; K. K. Yang et al., 2019), the search for optimal molecules is conducted by performing a constrained optimization.

In chip design, there are many optimization problems with complex constraints. RL (R. Cheng and Yan, 2021; Mirhoseini et al., 2021) readily handles constraints by taking a sequence of actions respecting constraints. However, RL requires many evaluations and thus resorts to a cheap proxy for the original objective. On the other hand, BO (Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022; Lyu et al., 2018; Changyong Oh, Bondesan, et al., 2022) requires a well-designed representation to handle constraints, but it can optimize the objective directly.

In Markov Decision Processes, many constraints can be easily handled by masking infeasible actions in the state transition. Such flexibility is one of the reasons for the popularity of RL in many complex problems. However, it comes at the cost of fragile training stability. Compared with RL, BO is deployed with a formulation without state transition, and thus, in some cases, BO produces more stable results (X. Dong, L. Liu, et al., 2021).

### 2.4.4 Design of Experiments

The design of experiments is concerned with identifying factors that affect the response of the experiment, approximating the relation of the factors and the response, finding an optimal setting of the factors with respect to the response, etc. (Montgomery, 2017; STANDARDS and TECHNOLOGY/SE-MATECH, 2013) BO also has been actively employed to find an optimal setting for the experiment in science and engineering (Greenhill et al., 2020; Shahriari et al., 2015).

Traditional experimental design methods, such as factorial design and space-filling design, do not consider the responses of the experiments (Greenhill et al., 2020; Montgomery, 2017). A more advanced method is response surface methodology (RSM) (Box and K. B. Wilson, 1992; Greenhill et al., 2020). RSM adaptively determines which factor values to try based on the responses of the experiments.

The general procedure of RSM resembles that of BO, but there are key differences. RSM does not consider the uncertainty, but BO actively utilizes the uncertainty to balance exploitation and exploration. RSM makes incremental changes purely based on a local search on its response surface, but BO searches globally based on the global viewpoint granted by the surrogate model.

In many design problems, not only in hyperparameter optimization but also in various fields in science and engineering, BO is becoming the method of choice. (Char et al., 2019; Frazier and Jialei Wang, 2016; R.-R. Griffiths and J. M. Hernández-Lobato, 2020; Jalas et al., 2021; X. Lu et al., 2018; Pang et al., 2017; Perdikaris and Karniadakis, 2016).

# 3 BOCK : BAYESIAN OPTIMIZATION WITH CYLINDRICAL KERNELS

## 3.1 INTRODUCTION

When we talk about stars and galaxies we use *parsecs* to describe structures, yet when we discuss the world around us we use *meters*. In other words, the natural lengthscale scale with which we describe the world increases with distance away from us. We believe this same idea is useful when performing optimization in high dimensional spaces.

In Bayesian Optimization (or other forms of hyperparameter optimization) we define a cube or a ball and search for the solution inside that volume. The origin of that sphere is special in the sense that this represents the part of space with the highest probability if finding the solution. Moreover, in high dimensions, when we move outwards, the amount of volume contained in an annulus with width $\delta R$, $A(\mathbf{c}; R - \delta R, R) = \{\mathbf{x} | R - \delta R < \| \mathbf{x} - \mathbf{c} \| < R\}$, grows exponentially with distance $R$. As such, if we would spend an equal amount of time searching each volume element $\delta V$, we would spend all our time at the boundary of our search region. This effective attraction to the places with more volume is the equivalent of an "entropic force" in physics, and in the case of optimization is highly undesirable, since we expect the solution at a small radius $R$.

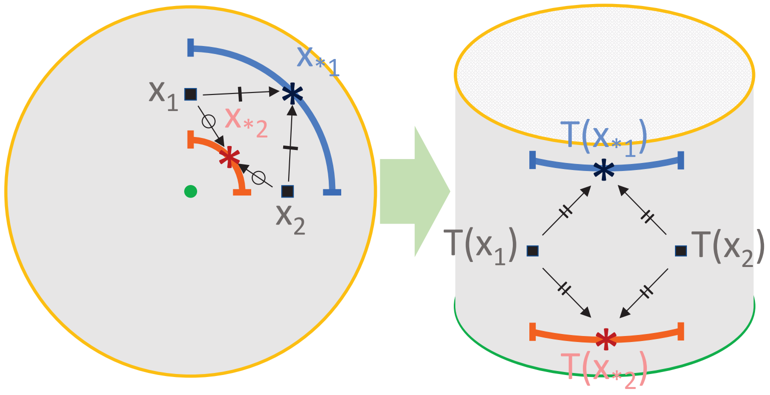


**Figure 3.1:** Cylindrical transformation

**Algorithm 2** Bayesian Optimization pipeline.

```
1: Input: surrogate model $\mathcal{M}$, acquisition function $\alpha$, search space $\mathcal{X}$, initial training data $\mathcal{D}_{init}$, function $f$
2: Output: optimum $\mathbf{x}_{opt} \in \mathcal{X}$ of $f$
3: Initialize $\mathcal{D} = \mathcal{D}_{init}$
4: while evaluation budget available do
5:     Set $\mu(\cdot|\mathcal{D}), \sigma^2(\cdot|\mathcal{D}) \leftarrow \mathcal{M}|\mathcal{D}$ // Surrogate function returns predictive mean function and predictive variance function by fitting $\mathcal{M}$ to $\mathcal{D}$
6:     Maximize $\widehat{\mathbf{x}} = \operatorname{argmax}_{\mathbf{x} \in \mathcal{X}} \alpha(\mu(\mathbf{x}|\mathcal{D}), \sigma^2(\mathbf{x}|\mathcal{D}))$
       // Acquisition function suggests next evaluation by maximization
7:     Evaluate $\hat{y} = f(\widehat{\mathbf{x}})$ // Evaluate the score of the point selected by the acquisition function
8:     Set $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\widehat{\mathbf{x}}, \hat{y})\}$ // Update the training dataset by including the newly evaluated pair $(\widehat{\mathbf{x}}, \hat{y})$
9: end while
```

In this chapter we, therefore, reformulate Bayesian Optimization in a transformed space, where a ball, $B(\mathbf{x}; R) = \{\mathbf{x} \| \| \mathbf{x} - \mathbf{c} \| \leqslant R\}$, is mapped to a cylinder, $C(p, q; \mathbf{c}, L = \{(r, \mathbf{a}) \| r \in [p, q], \| \mathbf{d} - \mathbf{c} \| = L\}$ (see Fig. 3.1). In this way, every annulus of width $\delta R$ contains an equal amount of volume for every radius $R$, and the entropic force pulling the optimizer to the boundary disappears. We call our method BOCK, for *Bayesian Optimization with Cylindrical Kernel*. BOCK is effective in handling the *boundary issue* – too much volume near the boundary (K. J. Swersky, 2017)). By mitigating the boundary issue, BOCK spends the evaluation budget in a promising region of the search space, the center, which is aligned with the *prior assumption* that the solution most likely lies close to the origin. We find that our algorithm is able to successfully handle much higher dimensional problems than standard Bayesian optimizers. As a result, we manage to not only optimize modestly sized neural network layers (up to 500 dimensions in our experiments), obtaining solutions competitive to SGD training, but also hyper-optimize stochastic depth Resnets (G. Huang et al., 2016).

## 3.2 PRELIMINARIES

### 3.2.1 Bayesian Optimization

Bayesian optimization aims at finding the global optimum of black-box functions, namely

$$\mathbf{x}_{opt} = \operatorname*{argmin}_{\mathbf{x}} f(\mathbf{x}) \tag{3.1}$$

The general pipeline of Bayesian Optimization is given in Alg. 1. Prior to starting, a search space must be defined, where the optimum $f(\mathbf{x}_{opt})$ will be searched for. Given this search space, the initial training dataset must be set, typically by naive guessing where the solution might lie or by informed expert knowledge of the problem. Having completed these two steps, Bayesian Optimization proceeds in an iterative fashion. At each round, in the absence of any other information regarding the nature of $f(\mathbf{x})$ a surrogate model attempts to approximate the behavior of $f(\mathbf{x})$ based on the so far observed points $(\mathbf{x}_i, y_i), y_i = f(\mathbf{x}_i)$. The surrogate function is then followed by an acquisition function that suggests the next most interesting point $\mathbf{x}_{i+1}$ that should be evaluated. The pair $(\mathbf{x}_i, y_i)$ is added to the training dataset, $\mathcal{D} = \mathcal{D} \cup (\mathbf{x}_i, y_i)$, and the process repeats until the optimization budget is depleted.

The first design choice of the Bayesian Optimization pipeline is the surrogate model. The task of the surrogate model is to model probabilistically the behavior of $f(\cdot)$ in the $x$-space in terms of *(a)* a predictive mean $\mu(\mathbf{x}_* \mid \mathcal{D})$ that approximates the value of $f(x)$ at any point $\mathbf{x}_*$, and *(b)* a predictive variance that represents the uncertainty of the surrogate model in this prediction. Any model that can provide a predictive mean and variance can be used as a surrogate model, including random forests (Hutter et al., 2011), tree-based models (J. S. Bergstra et al., 2011) and neural networks (Snoek, Rippel, et al., 2015; Springenberg et al., 2016). Among other things, Gaussian Processes not only provide enough flexibility it terms of kernel design but also allow for principled and tractable quantification of uncertainty (Rasmussen, 2003). Therefore, we choose Gaussian Processes as our surrogate model. The predictive mean and the predictive variance of Gaussian processes are given as below

$$\mu(\mathbf{x}_* \mid \mathcal{D}) = K_{*\mathcal{D}}(K_{\mathcal{D}\mathcal{D}} + \sigma^2 I)^{-1}\mathbf{y} \tag{3.2}$$

$$\sigma^2(\mathbf{x}_* \mid \mathcal{D}) = K_{**} - K_{*\mathcal{D}}(K_{\mathcal{D}\mathcal{D}} + \sigma^2_{obs} I)^{-1} K_{\mathcal{D}*} \tag{3.3}$$

where $K_{**} = K(\mathbf{x}_*, \mathbf{x}_*)$, $K_{*\mathcal{D}}$ is a row vector whose *i*th entry is $K(\mathbf{x}_*, \mathbf{x}_i)$, $K_{\mathcal{D}*} = (K_{*\mathcal{D}})^T$, $[K_{\mathcal{D}\mathcal{D}}]_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$, $\sigma^2_{obs}$ is the variance of observational noise and $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_i$ is the dataset of observations so far.

The second design choice of the Bayesian Optimization pipeline is the acquisition function. The predictive mean and the predictive variance from the surrogate model is input to the acquisition function that quantifies the significance of every point in $\mathbf{x}$ as a next evaluation point. While different acquisition functions have been explored in the literature (Hennig and Schuler, 2012; J. M. Hernández-Lobato, M. W. Hoffman, et al., 2014; Kushner, 1964; Močkus, 1975; Srinivas et al., 2010; Thompson, 1933), they all share the following property: they return high scores at regions of either high predictive variance (high but uncertain reward), or low predictive mean (modest but certain reward).

Last, the third design choice of the Bayesian Optimization pipeline, often overlooked, is the search space. In (Snoek, K. Swersky, et al., 2014) the kernel of the surrogate model is defined on a warped search space, thus allowing for a more flexible modeling of $f(\mathbf{x})$ by the surrogate function. As the search space defines where optimal solutions are to be sought for, the search space definition is a means of infusing prior knowledge into the Bayesian Optimization. Usually, a search space is set so that the expected optimum is close to the center.

### 3.2.2 High-dimensional Bayesian Optimization

Even with its successes in many applications, several theoretical as well as practical issues (Shahriari et al., 2015) still exist when employing Bayesian Optimization to real world problems. Among others, many Bayesian optimization algorithms are restricted in practice to problems of moderate dimensions. In high dimensional problems, one suffers from the curse of dimensionality. To overcome the curse of dimensionality, several works make structural assumptions, such as low effective dimensionality (J. Bergstra and Bengio, 2012; Ziyu Wang, Hutter, et al., 2016) or additive structure (Kandasamy, Schneider, et al., 2015).

Because of the way Gaussian Processes quantify uncertainty, the curse of dimensionality is a serious challenge for Gaussian Processes-based Bayesian Optimization in high dimensions. Since in high dimensions data points typically lie mostly on the boundary, and anyways far away from each other, the predictive variance tends to be higher in the regions near the boundary. Thus, the acquisition function is somewhat biased to choose evaluations near the boundary, hence, biasing Bayesian Optimization towards solution near the boundary and away from the center, contradicting with the prior assumption. This is the *boundary issue*(K. J. Swersky, 2017).

### 3.2.3 Contributions

Different from the majority of the Bayesian Optimization methods that rely on a Euclidean geometry of the search space implicitly or explicitly(J. S. Bergstra et al., 2011; Hutter et al., 2011; Snoek, Larochelle, et al., 2012; Snoek, Rippel, et al., 2015; Snoek, K. Swersky, et al., 2014; K. Swersky, Snoek, et al., 2013; Zi Wang, C. Li, et al., 2017), the proposed BOCK applies a cylindrical geometric transformation on it. The effect is that the volume near the center of the search space is expanded, while the volume near the boundary is shrunk. Compared to (Snoek, K. Swersky, et al., 2014), where warping functions were introduced with many kernel parameters to be learned, we do not train transformations. Also, we avoid learning many additional kernel parameters for better efficiency and scalability. Because of the transformation, the proposed BOCK solves also the issue of flat optimization surfaces of the acquisition function in high dimensional spaces (Rana et al., 2017). And compared to REMBO (Ziyu Wang, Hutter, et al., 2016), BOCK does not rely on assumptions of low dimensionality of the latent search space.

## 3.3 METHODS

### 3.3.1 Prior assumption and search space geometry

The flexibility of a function $f$ on a high-dimensional domain $X$ can be, and usually is, enormous. To control the flexibility and make the optimization feasible some reasonable assumptions are required. A standard assumption in Bayesian Optimization is the *prior assumption* K. J. Swersky, 2017, according to which the optimum of $f(\mathbf{x})$ should lie somewhere near the center of the search space $X$. Since the search space is set with the *prior assumption* in mind, it is reasonable for Bayesian Optimization to spend more evaluation budget in areas near the center of $X$.

It is interesting to study the relation of the *prior assumption* and the geometry of the search space. The ratio of the volume of two concentric balls $B(\mathbf{c}; R-\delta R)$ and $B(\mathbf{c}; R)$, with a radius difference of $\delta R$, is

$$\frac{\text{volume}(B(\mathbf{c}; R-\delta R))}{\text{volume}(B(\mathbf{c}; R))} = o((1-\delta)^D), \tag{3.4}$$

which rapidly goes to zero with increasing dimensionality $D$. This means that the volume of $B(\mathbf{c}; R)$ is mostly concentrated near the boundary, which in combination with Gaussian processes' behavior of high predictive variance at points far from data, creates the *boundary issue* K. J. Swersky, 2017.

It follows, therefore, that with a transformation of the search space we could avoid excessively biasing our search towards large values of $R$.

### 3.3.2 Cylindrical transformation of search space

The search space geometry has a direct influence on the kernel $K(\mathbf{x}, \mathbf{x}')$ of the Gaussian Process surrogate model, and, therefore, its predictive variance $\sigma^2(\mathbf{x})$, see eq. (3.3). A typical design choice for Gaussian Processes González et al., 2016; Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014 are stationary kernels, $K(\mathbf{x}, \mathbf{x}') \propto f(\mathbf{x}-\mathbf{x}')$. Unfortunately, stationary kernels are not well equipped to tackle the *boundary issue*. Specifically, while stationary kernels compute similarities only in terms of relative locations $\mathbf{x}-\mathbf{x}'$, the boundary issue dictates the use of location-aware kernels $K(\mathbf{x}, \mathbf{x}')$ to recognize whether $\mathbf{x}, \mathbf{x}'$ lie near the boundary or the center areas of the search space.

A kernel that can address this should have the following two properties. First, the kernel must define the similarity between two points $\mathbf{x}, \mathbf{x}'$ in terms of their absolute locations, namely the kernel has to be non-stationary. Second, the kernel must transform the geometry of its input (*i.e.*, the search space for the Gaussian Process surrogate model) such that regions near the center and the boundaries are equally represented. To put it otherwise, we need a geometric transformation of the search space that expands the region near the center while contracting the regions near the boundary. A transformation with these desirable properties is the cylindrical one, separating the radius and angular components of a point $\mathbf{x}$, namely

$$\begin{aligned} T(\mathbf{x}) &= \begin{cases} (\|\mathbf{x}\|_2, \mathbf{x}/\|\mathbf{x}\|_2) & \text{for } \|\mathbf{x}\|_2 \neq 0 \\ (0, \mathbf{a}_{arbitrary}) & \text{for } \|\mathbf{x}\|_2 = 0 \end{cases} \\ T^{-1}(r, \mathbf{a}) &= r\mathbf{a} \end{aligned} \tag{3.5}$$

where $\mathbf{a}_{arbitrary}$ is an arbitrarily chosen vector with unit $\ell_2$-norm [6].

After applying the geometric transformation we arrive at a new kernel $K_{cyl}(\mathbf{x}_1, \mathbf{x}_2)$, which we will refer to as the cylindrical kernel. The geodesic similarity measure (kernel) of $K_{cyl}$ on the *transformed cylinder*, $T(X)$, is defined as

$$K_{cyl}(\mathbf{x}_1, \mathbf{x}_2) = \widetilde{K}(T(\mathbf{x}_1), T(\mathbf{x}_2)) = K_r(r_1, r_2) \cdot K_a(\mathbf{a}_1, \mathbf{a}_2) \tag{3.6}$$

where the final kernel decomposes into a 1-D radius kernel $K_r$ measuring the similarity of the radii of $r_1, r_2$ and a angle kernel $K_a$.

For the angle kernel $K_a(\mathbf{a}_1, \mathbf{a}_2)$, we opt for a continuous radial kernel on the (hyper-)sphere Jayasumana et al., 2014,

$$K_d(\mathbf{a}_1, \mathbf{a}_2) = \sum_{p=0}^{P} c_p (\mathbf{a}_1^T \mathbf{a}_2)^p, \quad c_p \geqslant 0, \forall p \tag{3.7}$$

with trainable kernel parameters of $c_0, \cdots, c_P$ and $P$ user-defined. The advantages of a continuous radial kernel is two-fold. First, with increasing $P$ a continuous radial kernel can approximate any continuous positive definite kernel on the sphere with arbitrary precision Jayasumana et al., 2014. Second, the cylindrical kernel has $P + 1$ parameters, which is independent of the dimensionality of $X$. This means that while the continuous radial kernel retains enough flexibility, only few additional kernel parameters are introduced, which are independent of the dimensionality of the optimization problem and

6 Another possible geometric transformation could be from rectangular to spherical coordinates. Unfortunately, the inverse transformation from spherical to rectangular coordinate entails multiplication of many trigonometric functions, causing numerical instabilities because of large products of small numbers.

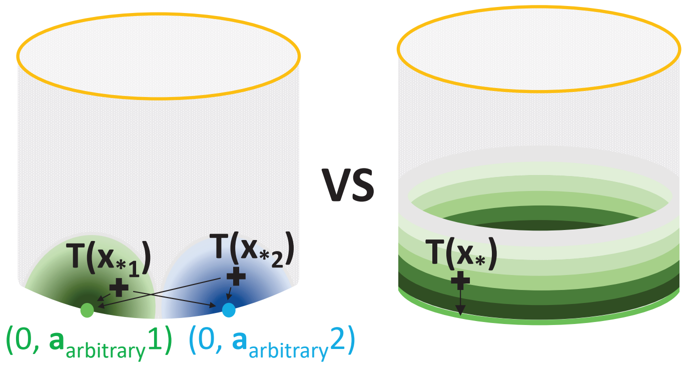


**Figure 3.2:** Similarity to the center point in the *transformed geometry*.

can, thus, easily scale to more than 50 dimensions. This compares favorably to Bayesian optimization with ARD kernels that introduce at least $d$ kernel parameters for a $d$-dimensional search space.

Although the boundary issue is mitigated by the cylindrical transformation of the search space, the prior assumption (good solutions are expected near the center) can be promoted. To this end, and to reinforce the near-center expansion of the cylindrical transformation, we consider input warping Snoek, K. Swersky, et al., 2014 on the radius kernel $K_r(r_1, r_2)$. Specifically, we use the cumulative distribution function of the Kumaraswamy distribution, $\mathrm{Kuma}(r|\alpha,\beta) = 1-(1-r^\alpha)^\beta$ (with $\alpha > 0, \beta > 0$),

$$\begin{aligned} K_r(r_1, r_2) =& K_{base}(\mathrm{Kuma}(r_1|\alpha,\beta), \mathrm{Kuma}(r_1|\alpha,\beta)) \\ =& K_{base}(1-(1-r_1^\alpha)^\beta, 1-(1-r_2^\alpha)^\beta|\alpha,\beta) \end{aligned}$$

where the non-negative $a, b$ are learned together with the kernel parameters. $K_{base}$ is the base kernel for measuring the radius-based similarity. Although any kernel is possible for $K_{base}$, in our implementations we opt for the Matern52 kernel used in Spearmint Snoek, Larochelle, et al., 2012. By making radius warping concave and non-decreasing, $K_r$ and, in turn, $K_{cyl}$ focus more on areas with small radii.

Overall, the transformation of the search space has two effects. The first effect is that the volume is redistributed, such that areas near the center are expanded, while areas near the boundaries are contracted. Bayesian optimization's attention in the search space, therefore, is also redistributed from the boundaries to the center of the search space. The second effect is that the kernel similarity changes, such that the predictive variance depends mostly on the angular difference between the existing data points and the ones to be evaluated. An example is illustrated in Fig. 3.1, where our dataset comprises of $\mathcal{D} =$

$\{\mathbf{x}_1, \mathbf{x}_2\}$ and the acquisition function must select between two points, $\mathbf{x}_{*,a}$ and $\mathbf{x}_{*,b}$. Whereas in the original Euclidean geometry (Fig. 3.1 to the left) $\mathbf{x}_{*,a}$ is further away from $\mathcal{D}$, thus having higher predictive variance, in the cylindrical geometry both $\mathbf{x}_{*,a}$ and $\mathbf{x}_{*,b}$ are equally far, thus reducing the artificial preference to near-boundary points.

### 3.3.3 Balancing center over-expansion

The transformation $T$ maps an annulus $A(\mathbf{0}; R - \delta R, R)$ of width $\delta R$ to the cylinder $C(R - \delta R, R; \mathbf{0}, 1)$, where $(\mathbf{0}, 1)$ is the center and the radius of the cylinder. For almost any point in the original ball there is a one-to-one mapping to a point on the cylinder. The only exception is the extreme case of the ball origin, which is mapped to the 0-width sphere $C(0, 0; 0, 1) = \{(0, \mathbf{a}) | \|\mathbf{a}\| = 1\}$ on the base of the cylinder (bright green circle in the Fig. 3.2 to the right). Namely, the center point $\mathbf{x}_{center}$ is overly expanded, corresponding to a set of points. Because of the one-to-many correspondence between $\mathbf{x}_{center}$ and $C(0, 0; 0, 1)$, an arbitrary point is selected in eq. (3.5).

Unfortunately, the dependency on a point that is both arbitrary *and* fixed incurs an arbitrary behavior of $K_{cyl}$ as well. For any point $\mathbf{x}_* \in X \setminus \{\mathbf{0}\}$ the kernel $K_{cyl}(\mathbf{x}_{center}, \mathbf{x}_*)$ changes arbitrarily, depending on the choice of $\mathbf{a}_{arbitrary}$, see Fig. 3.2. Having a fixed arbitrary point, therefore, is undesirable as it favors points lying closer to it. To this end, we define $\mathbf{a}_{arbitrary}$ as the angular component of a test point $\mathbf{x}_*$, $\mathbf{a}_{arbitrary} = \mathbf{x}_* / \| \mathbf{x}_* \|$, thus being not fixed anymore. Geometrically, this is equivalent to using the point in $C(0, 0; 0, 1)$ closest to $T(\mathbf{x}_*)$, see Fig. 3.2 to the right. This implies that, if the origin is in the dataset, the Gram matrix needed for computing the predictive density now depends on the angular location of the test point under consideration. This is somewhat unconventional but still well behaved (the kernel is still positive definite and the predictive mean and variance change smoothly). More details can be found in Appx. A.1.

## 3.4 EXPERIMENTS

In Bayesian optimization experiments, we need to define *(a)* how to train the surrogate model, *(b)* how to optimize the acquisition function and *(c)* how to set the search space. For BOCK we use Gaussian Process surrogate models, where following (Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014) we train parameters of BOCK with MCMC (slice sampling (Murray and Adams, 2010; Neal, 2003)) . For the acquisition function, we use the Adam (Kingma and Ba, 2015) optimizer, instead of L-BFGS-B (Zhu et al., 1997). To begin the optimization we feed 20 initial points to Adam. To select the 20 initial points, a sobol sequence (Bratley and B. L. Fox, 1988) of 20,000 points

Table 3.1: Performance on benchmarks of 20 and 200 dimensions

| Benchmark | Repeated Branin | | Repeated Hartmann6 | | Rosenbrock | | Levy | |
|---|---|---|---|---|---|---|---|---|
| Dimensions | 20 | 100 | 20 | 100 | 20 | 100 | 20 | 100 |
| Minimum | 0.3979 | 0.3979 | -3.3223 | -3.3223 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |
| SMAC | 15.95±3.71 | 20.03±0.85 | -1.61±0.12 | -1.16±0.19 | 8579.13± 58.45 | 8593.09± 18.80 | 2.35±0.00 | 9.60±0.04 |
| TPE | 7.59±1.20 | 23.55±0.73 | -1.74±0.10 | -1.01±0.10 | 8608.36± 0.00 | 8608.36± 0.00 | 2.35±0.00 | 9.62±0.00 |
| Spearmint | 5.07±3.01 | 2.78±1.06 | -2.60±0.42 | -2.55±0.19 | 7970.05± 1276.62 | 8608.36± 0.00 | 1.88±0.59 | 4.87±0.35 |
| Spearmint+ | 6.83±0.32 | - | -2.91±0.25 | - | 5909.63± 2725.76 | - | 2.35±0.00 | - |
| Additive BO* | 5.75±0.93 | 14.07±0.84 | -3.03±0.13 | -1.69±0.22 | 3632.25± 1642.71 | 7378.27± 305.24 | 2.32±0.02 | 9.59±0.04 |
| Elastic BO | 6.77±4.85 | - | -2.85±0.57 | - | 5346.96± 2494.89 | - | 1.35±0.34 | - |
| Matern | 0.41±0.00 | 0.54±0.06 | -3.29±0.04 | -2.91±0.26 | 230.25± 187.41 | 231.42± 28.94 | 0.38±0.13 | 2.17±0.18 |
| BOCK | 0.50±0.12 | 1.03±0.17 | -3.30±0.02 | -3.16±0.10 | 47.87± 33.94 | 128.69± 52.84 | 0.54±0.13 | 6.78±2.16 |

+ Spearmint+ (Snoek, K. Swersky, et al., 2014) and Elastic BO (Rana et al., 2017) are evaluated only on the 20-dimensional cases because of prohibitive execution times. * Additive BO (Kandasamy, Schneider, et al., 2015) requires a user-specified "maximum group size" to define the additive structure. In each experiment we tried 5 different values and reported the best result.

is generated on the cube (we used the cube for fair comparison with others). The acquisition function is evaluated on these points and the largest 20 points are chosen as the initial ones. Instead of using a static sobol sequence in the entire course of Bayesian optimization (Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014), we generate different sobol sequences for different evaluations, as fixed grid point impose too strong constraints in high dimensional problems. In the d-dimensional space, our search space is a ball $B(\mathbf{0}, \sqrt{d})$ circumscribing a cube $[-1, 1]^d$, which is the scaled and translated version of the typical search region, unit cube $[0, 1]^d$. Our search space is much larger than a cube. By generating sobol sequence on the cube, the reduction of the *boundary issue* mostly happens at corners of the cube $[-1, 1]^d$.

### 3.4.1 Benchmarks

First, we compare different Bayesian Optimization methods and BOCK on four benchmark functions. Specifically, following (Eggensperger et al., 2013; Laguna and Marti, 2005) we use the repeated Branin, repeated Hartmann6 and Levy to assess Bayesian Optimization in high dimensions. To test the ability of Bayesian Optimization methods to optimize functions with more complex structure and stronger intra-class dependencies, we additionally include the Rosenbrock benchmark, typically used as benchmark for gradient-based optimization (Laguna and Marti, 2005). The precise formulas for the four benchmark functions are added to Appx. A.2. We solve the benchmark functions in 20 and 100 dimensions[7], using 200 and 600 function evaluations respectively for all Bayesian Optimization methods. We compare the proposed BOCK with the following Bayesian Optimization methods using publicly available software: SMAC (Hutter et al., 2011), TPE (J. S. Bergstra et al., 2011), Spearmint (Snoek,

7 We also solve the 50-dimensional cases. As conclusions are similar, we report these results in Appx. A.4.

Larochelle, et al., 2012), Spearmint+ (Snoek, K. Swersky, et al., 2014), additive BO (Kandasamy, Schneider, et al., 2015), elastic BO (Rana et al., 2017). We also report an in-house improved Spearmint implementation, which we refer to as Matern.[8]

We focus on four aspects: *(a)* accuracy, *(b)* efficiency (wall clock time) vs accuracy, *(c)* scalability (number of dimensions) vs efficiency, and *(d)* robustness of BOCK to hyperpararameters and other design choices. We study (a) in all four benchmark functions. For brevity, we report (b)-(d) on the Rosenbrock benchmark only, the hardest of the four benchmark functions for all Bayesian Optimization methods in terms of accuracy, and report results the rest of the benchmark functions in Appx. A.4.

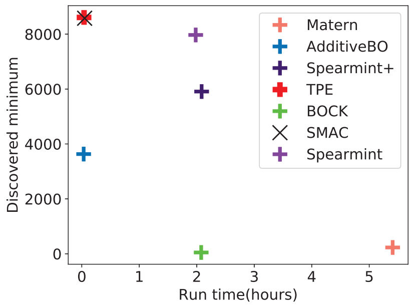


**Figure 3.3:** Accuracy vs wall clock time for 20 dim. Rosenbrock

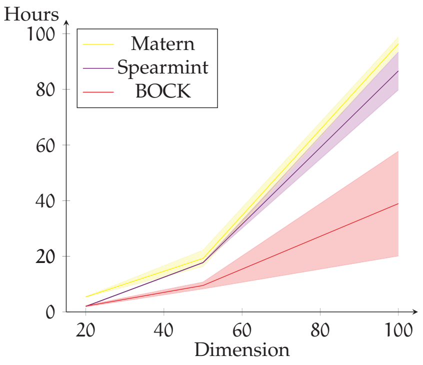


**Figure 3.4:** Wall clock time on Rosenbrock of 20, 50 and 100 dim.

**Accuracy.** We first present the results regarding the accuracy of BOCK and the Bayesian Optimization baselines in Tab. 3.1. BOCK and Matern outperform others with large margin in discovering near optimal solutions. For benchmark functions with complicated dependen-

8 Differences with standard Spearmint: (a) a non-ARD, Matern52 kernel for the surrogate model, (b) dynamic search grid generation per evaluation, (c) Adam (Kingma and Ba, 2015) instead of L-BFGS-B (Zhu et al., 1997), (d) more steps for optimizer.

cies between variables, such as the repeated Hartmann6 and Rosenbrock, BOCK consistently discovers smaller values compared to other baselines, while not being affected by an increasing number of dimensions. What is more, BOCK is on par even with methods that are designed to exploit the specific geometric structures, if the same geometric structures can be found in the the evaluated functions. For instance, the repeated Branin and Levy have an additive structure, where the same low dimensional structure is repeated. The non-ARD kernel of Matern can exploit such special, additive structures. BOCK is able to reach a similar near-optimum solution without being explicitly designed to exploit such structures.

We conclude that BOCK is accurate, especially when we have no knowledge of the geometric landscape of the evaluated functions. In the remaining of the experiments we focus on the Bayesian Optimization methods with competitive performance, namely BOCK, Spearmint and Matern.

**Efficiency vs accuracy.** Next, we compare in Fig. 3.3 the accuracy of the different Bayesian Optimization methods as a function of their wall clock times for the 20-dimensional case for Rosenbrock. As the function minimum is $f(\mathbf{x}_{\text{opt}}) = 0$, the optimal operating point is at $(0, 0)$. BOCK is the closest to the optimal point. Matern is the second most accurate, while being considerably slower to run. SMAC (Hutter et al., 2011) and AdditiveBO (Kandasamy, Schneider, et al., 2015) are faster than BOCK, however, they are also considerably less accurate.

**Scalability.** In Fig. 3.4 we evaluate the most accurate Bayesian Optimization methods from Tab. 3.1 (Spearmint, Matern and BOCK.) with respect to how scalable they are, namely measuring the wall clock time for an increasing number of dimensions. We test on Rosenbrock of 20, 50 and 100 dimensions with 200, 400 and 600 function evaluations respectively for all methods. We report mean and standard error of 5 runs. Compared to Spearmint, BOCK is less affected by the increasing number of dimensions. Not only the BOCK surrogate kernel requires fewer parameters, but also the number of surrogate kernel parameters is independent of the number of input dimensions, thus making the surrogate model fitting faster. BOCK is also faster than Matern, although the latter uses a non-ARD kernel that is also independent of the number of input dimensions. Presumably, this is due to a better, or smoother, optimization landscape after the cylindrical transformation of geometry of the input space, affecting positively the search dynamics. We conclude that BOCK is less affected by the increasing number of dimensions, thus scaling better.

**Robustness.** To study the robustness of BOCK to design choices, we compare three BOCK variants. The first is the standard BOCK as described in Sec. 3.3. The second variant, BOCK-W, removes the input warping on the radius component. The third variant, BOCK+B, includes an additional boundary treatment to study whether further reduction of the predictive variance is beneficial. Specifically, we reduce the predictive variance by adding "fake" data. [9] We report the result in Tab. 3.2.

Table 3.2: Ablation study on BOCK variants

| Dimensions | 20 | 50 | 100 |
|---|---|---|---|
| BOCK | 47.87± 33.94 | 29.65±11.56 | 128.69± 52.84 |
| BOCK-W | 1314.03± 1619.73 | 51.14±58.18 | 157.89± 161.92 |
| BOCK+B | 48.87± 18.33 | 33.90±21.69 | 87.00± 36.88 |

Removing the input warping on the radius is hurting the robustness, as BOCK-W tends to reach slightly worse minima than BOCK. However, introducing further boundary treatments has a marginal effect.

Further, we assess the sensitivity of BOCK with respect to the hyperparameter P in eq.(3.5). For $P = 3, 5, 7, 9$, we observe that higher P tends to give slightly better minima, while increasing the computational cost.

For clarity of presentation, as well as to maintain the experimental efficiency, in the rest of the experiments we focus on BOCK with $P = 3$.

### 3.4.2 Optimizing a neural network layer

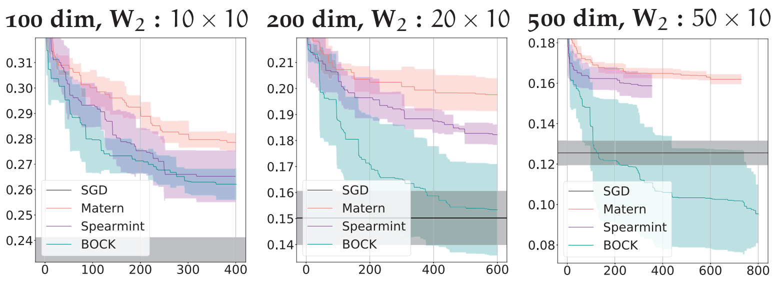


Figure 3.5: Neural network parameter optimization (100, 200, 500) dim.

As BOCK allows for accurate and efficient Bayesian Optimization for high-dimensional problems, we next perform a stress test, attempting to optimize neural network layers of 100, 200 and 500 di-

9 Predictive variance depends only on the inputs $\mathbf{x}$, not the evaluations $y = f(\mathbf{x})$. Thus we can manipulate the predictive variance only with input data. BOCK+B uses one additional "fake data", which does not have output value(evaluation), in its predictive variance. BOCK's predictive variance $\sigma^2(\mathbf{x}_* \,|\, \mathcal{D})$ becomes $\sigma^2(\mathbf{x}_* \,|\, \mathcal{D} \cup \{(R\,\mathbf{x}_* / \|\,\mathbf{x}_*\,\|, \sim)\})$ in BOCK+B on the search space of the ball $B(\mathbf{0}; R)$, where $(R\,\mathbf{x}_* / \|\,\mathbf{x}_*\,\|, \sim)$ is the fake data.

mensions. Specifically, we define a two-layered neural network with architecture: $784\,\mathbf{x} \xrightarrow{\mathbf{W}_1,\mathbf{b}_1} N_{hidden}\,\mathbf{x} \xrightarrow{\mathbf{W}_2,\mathbf{b}_2} 10$, using ReLU as the intermediate non-linearity.

In this experiment we are *only* interested in the optimization ability of BOCK of the parameters of a neural network, not in its ability to find solutions that generalize well. Thus, we intentionally follow a procedure that tests if BOCK is able to even overfit to the test set. Specifically, for all Bayesian optimization experiments $\mathbf{W}_1$, $\mathbf{b}_1$ and $\mathbf{b}_2$ are optimized with Adam (Kingma and Ba, 2015) and $\mathbf{W}_2$ with Bayesian Optimization. The training proceeds as follows. First, Bayesian Optimization suggests a $\mathbf{W}_2$ based on evaluations on the **test** set. Given this $\mathbf{W}_2$ we train on the **train+validation** sets the $\mathbf{W}_1, b_1, b_2$ with Adam, then repeat. We show results in Fig. 3.5, where we report mean and standard deviation over 5 runs for all methods. We compare BOCK with the competitive Spearmint and Matern. As baseline, we train a network with Adam (Kingma and Ba, 2015) on the training set and report the test loss. To the best of our knowledge we are the first to apply Gaussian Process-based Bayesian Optimization in so high-dimensional and complex, representation learning spaces. [10].

We observe that BOCK clearly outperforms Spearmint and Matern, with the gap increasing for higher $\mathbf{W}_2$ dimensions. What is more surprising, however, is that BOCK is able to match and even outperform the Adam-based SGD in the 200 and 500-dimensional experiments for all 5 runs. There are two reasons for this. First, in this experiment, all Bayesian optimization algorithms directly optimize the test loss. Second, in its sophistication Adam (Kingma and Ba, 2015) probably overfits to the training set.

In the end, the final neural network is obviously **not optimal** in terms of generalization, as to optimize $\mathbf{W}_2$ BOCK has access to the test set. However, even the fact that it is possible to optimize such high-dimensional and complex (representation learning) functions with Bayesian Optimization is noteworthy. We conclude that BOCK is able to optimize complex, multiple-optima functions, such as neural network layers.

### 3.4.3 Hyper-optimizing stochastic depth ResNets

As BOCK allows for accurate and efficient Bayesian Optimization, in our last experiment we turn our attention to a practical hyperparameter optimization application. Stochastic Depth ResNet (SDResNet) (G. Huang et al., 2016) was shown to obtain better accuracy and faster training by introducing a stochastic mechanism that randomly suppresses ResNet blocks (ResBlock) (He et al., 2016a). The stochastic

10 To our knowledge, running Bayesian Optimization on 200 or 500 dimensional problems has only been tried with methods assuming low effective dimensionality (B. Chen et al., 2012; Ziyu Wang, Hutter, et al., 2016)

Table 3.3: Optimization of the "death rates" of a Stochastic Depth ResNet-110

| Method | Test Acc. | Val. Acc. | Exp. Depth |
|---|---|---|---|
| ResNet-110 | 72.98±0.43 | 73.03±0.36 | 110.00 |
| SDResNet-110+Linear | 74.90±0.15 | 75.06±0.04 | 82.50 |
| SDResNet-110+BOCK | 75.06±0.19 | 75.21±0.05 | 74.51±1.22 |

mechanism for dropping ResBlocks is controlled by a vector $p \in [0,1]^t$ of probabilities for $t$ ResBlocks, called "death rate". In (G. Huang et al., 2016) a linearly increasing (from input to output) death rate was shown to improve accuracies.

Instead of pre-defined death rates, we employ BOCK to find the optimal death date vector for SDRes-110 on CIFAR100 (Krizhevsky and Hinton, 2009). We first train an SD-ResNet for 250 epochs and linear death rates with exactly the same configuration in (G. Huang et al., 2016) up to 250 epochs. In this experiment BOCK has access to the **training and validation set only**. Then, per iteration BOCK first proposes the next candidate $p$ based on evaluation on the validation set. Given the candidate $p$ we run 100 epochs of SGD on the training set and repeat with an annealed learning rate (0.01 for 50 epochs, then 0.001 for 50 more). We initialize the death rate vector to $p = [0.5, 0.5, ..., 0.5]$. We report the final accuracies computed in the **unseen test set** in Tab. 3.3, using only 50 evaluations.

We observe that BOCK learns a p-value that results in an improved validation accuracy compared to SDResNet, all the while allowing for a lower expected depth. The improved validation accuracy materializes to an only slightly better test accuracy, however. One reason is that optimization is not directly equivalent to learning, as also explained in Sec. 3.4.2. What is more, it is likely that the accuracy of SDResNet-110 on CIFAR-100 is maxed out, especially considering that only 50 evaluations were made. We conclude that BOCK allows for successful and efficient Bayesian Optimization even for practical, large-scale learning problems.

## 3.5 CONCLUSION

We propose BOCK, Bayesian Optimization with Cylindrical Kernels. Many of the problems in Bayesian Optimization relate to the *boundary issue* (too much value near the boundary), and the *prior assumption* (optimal solution probably near the center). Because of the boundary issue, not only much of the evaluation budget is unevenly spent to the boundaries, but also the *prior assumption* is violated. The basic idea behind BOCK is to transform the ball geometry of the search space with a cylindrical transformation, expanding the volume near the center while contracting it near the boundaries. As such, the Bayesian optimization focuses less on the boundaries and more on the center.

We test BOCK extensively in various settings. On standard benchmark functions BOCK is not only more accurate, but also more efficient and scalable compared to state-of-the-art Bayesian Optimization alternatives. Surprisingly, optimizing a neural network (on the test set) up to 500 dimensions with BOCK allows for even better (albeit overfitting) parameters than SGD with Adam Kingma and Ba, 2015. And hyper-optimizing the "death rate" of stochastic depth ResNet G. Huang et al., 2016 results in smaller ResNets while maintaining accuracy.

We conclude that BOCK allows for accurate, efficient and scalable Gaussian Process-based Bayesian Optimization. We plan to make the code public upon acceptance.

# 4 COMBINATORIAL BAYESIAN OPTIMIZATION USING THE GRAPH CARTESIAN PRODUCT

## 4.1 INTRODUCTION

This chapter focuses on Bayesian Optimization (BO) for objectives on combinatorial search spaces consisting of ordinal or categorical variables. Combinatorial BO (Jones et al., 1998) has many applications, including finding optimal chipset configurations, discovering the optimal architecture of a deep neural network or the optimization of compilers to embed software on hardware optimally. All these applications, where Combinatorial BO is potentially useful, feature the following properties. They *(i)* have black-box objectives for which gradient-based optimizers (Aaron Wilson et al., 2014) are not amenable, *(ii)* have expensive evaluation procedures for which methods with low sample efficiency such as, evolution (Freitas, 2009) or genetic (Davis, 1991) algorithms are unsuitable, and *(iii)* have noisy evaluations and highly non-linear objectives for which simple and exact solutions are inaccurate (Brochu et al., 2010; Frazier, 2018; Shahriari et al., 2015).

Interestingly, most BO methods in the literature have focused on continuous (Močkus, 1975) rather than combinatorial search spaces. One of the reasons is that the most successful BO methods are built on top of Gaussian Processes (GPs) (Kandasamy, Dasarathy, J. B. Oliva, et al., 2016; ChangYong Oh et al., 2018; Snoek, Larochelle, et al., 2012). As GPs rely on the smoothness defined by a kernel to model uncertainty (Rasmussen, 2003), they are originally proposed for, and mostly used in, continuous input spaces. In spite of the presence of kernels proposed on combinatorial structures (Haussler, 1999; R. I. Kondor and Lafferty, 2002; Smola and R. Kondor, 2003), to date the relation between the smoothness of graph signals and the smoothness of functions defined on combinatorial structures has been overlooked and not been exploited for BO on combinatorial structures. A simple solution is to use continuous kernels and round them up. This rounding, however, is not incorporated when computing the covariances at the next BO iteration (Garrido-Merchán and D. Hernández-Lobato, 2020), leading to unwanted artifacts. Furthermore, when considering combinatorial search spaces the number of possible configurations quickly explodes: for $M$ categorical variables with $k$ categories the number of possible combinations scales with $\mathcal{O}(k^M)$. Applying BO with GPs on combinatorial spaces is, therefore, not straightforward.

We propose COMBO, a novel Combinatorial BO designed to tackle the aforementioned problems of lack of smoothness and computational complexity on combinatorial structures. To introduce smoothness of a function on combinatorial structures, we propose the *combinatorial graph*. The combinatorial graph comprises sub-graphs –one per categorical (or ordinal) variable– later combined by the graph Cartesian product. The combinatorial graph contains as vertices all possible combinatorial choices. We define then smoothness of functions on combinatorial structures to be the smoothness of graph signals using the Graph Fourier Transform (GFT) (Ortega et al., 2018). Specifically, we propose as our GP kernel on the graph a variant of the diffusion kernel, the automatic relevance determination(ARD) diffusion kernel, for which computing the GFT is computationally tractable via a decomposition of the eigensystem. With a GP on a graph COMBO accounts for arbitrarily high order interactions between variables. Moreover, using the sparsity-inducing Horseshoe prior (Carvalho et al., 2009) on the ARD parameters COMBO performs variable selection and scales up to high-dimensional. COMBO allows for accurate, efficient and large-scale BO on combinatorial search spaces.

In this work, we make the following contributions. First, we show how to introduce smoothness on combinatorial search spaces by introducing combinatorial graphs. On top of a combinatorial graph we define a kernel using the GFT. Second, we present an algorithm for Combinatorial BO that is computationally scalable to high dimensional problems. Third, we introduce individual scale parameters for each variable making the diffusion kernel more flexible. When adopting a sparsity inducing Horseshoe prior (Carvalho et al., 2009, 2010), COMBO performs variable selection which makes it scalable to high dimensional problems. We validate COMBO extensively on *(i)* four numerical benchmarks, as well as two realistic test cases: *(ii)* the weighted maximum satisfiability problem (P. Hansen and Jaumard, 1990; Resende et al., 1997), where one must find boolean values that maximize the combined weights of satisfied clauses, that can be made true by turning on and off the variables in the formula, *(iii)* neural architecture search (Elsken et al., 2019; Wistuba, Rawat, et al., 2019). Results show that COMBO consistently outperforms all competitors.

## 4.2 METHODS

### 4.2.1 Bayesian optimization with Gaussian processes

Bayesian optimization (BO) aims at finding the global optimum of a black-box function $f$ over a search space $\mathcal{X}$, namely, $\mathbf{x}_{opt} = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. At each round, a surrogate model attempts to approximate $f(\mathbf{x})$ based on the evaluations so far, $\mathcal{D} = \{(\mathbf{x}_i, y_i = f(\mathbf{x}_i))\}$. Then an acquisition function suggests the most promising point $\mathbf{x}_{i+1}$ that should be evaluated. The $\mathcal{D}$ is appended by the new evaluation, $\mathcal{D} = \mathcal{D} \cup (\{\mathbf{x}_{i+1}, y_{i+1})\}$. The process repeats until the evaluation budget is depleted.

The crucial design choice in BO is the surrogate model that models $f(\cdot)$ in terms of *(i)* a predictive mean to predict $f(\cdot)$, and *(ii)* a predictive variance to quantify the prediction uncertainty. With a GP surrogate model, we have the predictive mean $\mu(\mathbf{x}_* \,|\, \mathcal{D}) = K_{*\mathcal{D}}(K_{\mathcal{D}\mathcal{D}} + \sigma_n^2 I)^{-1}\,\mathbf{y}$ and variance $\sigma^2(\mathbf{x}_* \,|\, \mathcal{D}) = K_{**} - K_{*\mathcal{D}}(K_{\mathcal{D}\mathcal{D}} + \sigma_n^2 I)^{-1} K_{\mathcal{D}*}$ where $K_{**} = K(\mathbf{x}_*, \mathbf{x}_*)$, $[K_{*\mathcal{D}}]_{1,i} = K(\mathbf{x}_*, \mathbf{x}_i)$, $K_{\mathcal{D}*} = (K_{*\mathcal{D}})^T$, $[K_{\mathcal{D}\mathcal{D}}]_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$ and $\sigma_n^2$ is the noise variance.

### 4.2.2 Combinatorial graphs and kernels

In BO on continuous search spaces the most popular surrogate models rely on GPs (Kandasamy, Dasarathy, J. B. Oliva, et al., 2016; ChangYong Oh et al., 2018; Snoek, Larochelle, et al., 2012). Their popularity does not extend to combinatorial spaces, although kernels on combinatorial structures have also been proposed (Haussler, 1999; R. I. Kondor and Lafferty, 2002; Smola and R. Kondor, 2003). To design an effective GP-based BO algorithm on combinatorial structures, a space of smooth functions –defined by the GP– is needed. We circumvent this requirement by the notion of the combinatorial graph defined as a graph, which contains all possible combinatorial choices as its vertices for a given combinatorial problem. That is, each vertex corresponds to a different joint assignment of categorical or ordinal variables. If two vertices are connected by an edge, then their respective set of combinatorial choices differ only by a single combinatorial choice. As a consequence, we can now revisit the notion of smoothness on combinatorial structures as smoothness of a graph signal (Chung, 1997; Ortega et al., 2018) defined on the combinatorial graph. On a combinatorial graph, the shortest path is closely related to the Hamming distance.

THE COMBINATORIAL GRAPH To construct the combinatorial graph, we first define one sub-graph per combinatorial variable $C_i$, $\mathcal{G}(C_i)$. For a categorical variable $C_i$, the sub-graph $\mathcal{G}(C_i)$ is chosen to be a complete graph while for an ordinal variable we have a path graph. We aim at building a search space for combinatorial choices, i.e. a combinatorial graph, by combining sub-graphs $G(C_i)$ in such way that a distance between two adjacent vertices corresponds to a change of a value of a single combinatorial variable. It turns out that the graph Cartesian product (Hammack et al., 2011) ensures this property. Then, the graph Cartesian product of subgraphs $\mathcal{G}(C_j) = (\mathcal{V}_j, \mathcal{E}_j)$ is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}) = \square_i\, \mathcal{G}(C_i)$, where $\mathcal{V} = \times_i\, \mathcal{V}_i$ and $(v_1 =$

$(c_1^{(1)}, \cdots, c_N^{(1)}), v_2 = (c_1^{(2)}, \cdots, c_N^{(2)})) \in \mathcal{E}$ if and only if $\exists j$ such that $\forall i \neq j \; c_i^{(1)} = c_i^{(2)}$ and $(c_j^{(1)}, c_j^{(2)}) \in \mathcal{E}_j$.

As an example, let us consider a simplistic hyperparameter optimization problem for learning a neural network with three combinatorial variables: *(i)* the batch size, $c_1 \in C_1 = \{16, 32, 64\}$, *(ii)* the optimizer $c_2 \in C_2 = \{\mathtt{AdaDelta}, \mathtt{RMSProp}, \mathtt{Adam}\}$ and *(iii)* the learning rate annealing $c_3 \in C_3 = \{\mathtt{Constant}, \mathtt{Annealing}\}$. The sub-graphs $\{\mathcal{G}(C_i)\}_{i=1,2,3}$ for each of the combinatorial variables, as well as the final combinatorial graph after the graph Cartesian product, are illustrated in Fig. 4.1. For the ordinal batch size variable we have a path graph, whereas for the categorical optimizer and learning rate annealing variables we have complete graphs. The final combinatorial graph contains all possible combinations for batch size, optimizer and learning rate annealing.

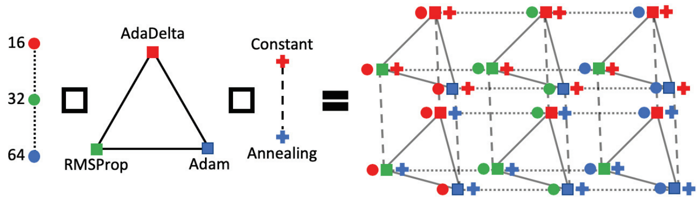


**Figure 4.1:** Combinatorial Graph: graph Cartesian product $\mathcal{G}(C_1) \square \mathcal{G}(C_2) \square \mathcal{G}(C_3)$

CARTESIAN PRODUCT AND HAMMING DISTANCE The Hamming distance is a natural choice of distance on categorical variables. With all complete sub-graphs, the shortest path between two vertices in the combinatorial graph is exactly equivalent to the Hamming distance between the respective categorical choices.

**Theorem 4.1.** *Assume a combinatorial graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ constructed from categorical variables, $C_1, \ldots, C_N$, that is, $\mathcal{G}$ is a graph Cartesian product $\square_i \, \mathcal{G}(C_i)$ of complete sub-graphs $\{\mathcal{G}(C_i)\}_i$. Then the shortest path $s(v_1, v_2; \mathcal{G})$ between vertices $v_1 = (c_1^{(1)}, \cdots, c_N^{(1)}), v_2 = (c_1^{(2)}, \cdots, c_N^{(2)}) \in \mathcal{V}$ on $\mathcal{G}$ is equal to the Hamming distance between $(c_1^{(1)}, \cdots, c_N^{(1)})$ and $(c_1^{(2)}, \cdots, c_N^{(2)})$.*

*Proof.* See Appx. Thm. B.1 □

When there is a sub-graph which is not complete, the below result follows from the Thm. 4.1:

**Corollary 4.2.** *If a sub-graph is not a complete graph, then the shortest path is equal to or bigger than the Hamming distance.*

The combinatorial graph using the graph Cartesian product is a natural search space for combinatorial variables that can encode a widely used metric on combinatorial variables like Hamming distance.

KERNELS ON COMBINATORIAL GRAPHS. In order to define the GP surrogate model for a combinatorial problem, we need to specify a a proper kernel on a combinatorial graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The role of the surrogate model is to smoothly interpolate and extrapolate neighboring data. To define a smooth function on a graph, i.e. a smooth graph signal $f : \mathcal{V} \mapsto \mathbb{R}$, we adopt Graph Fourier Transforms (GFT) from graph signal processing (Ortega et al., 2018). Similar to Fourier analysis on Euclidean spaces, GFT can represent any graph signal as a linear combination of graph Fourier bases. Suppressing the high frequency modes of the eigendecomposition approximates the signal with a smooth function on the graph. We adopt the diffusion kernel which penalizes basis-functions in accordance with the magnitude of the frequency (R. I. Kondor and Lafferty, 2002; Smola and R. Kondor, 2003).

To compute the diffusion kernel on the combinatorial graph $\mathcal{G}$, we need the eigensystem of graph Laplacian $L(\mathcal{G}) = \mathbf{D}_{\mathcal{G}} - \mathbf{A}_{\mathcal{G}}$, where $\mathbf{A}_{\mathcal{G}}$ is the adjacency matrix and $\mathbf{D}_{\mathcal{G}}$ is the degree matrix of the graph $\mathcal{G}$. The eigenvalues $\{\lambda_1, \lambda_2, \cdots, \lambda_{|\mathcal{V}|}\}$ and eigenvectors $\{u_1, u_2, \cdots, u_{|\mathcal{V}|}\}$ of the graph Laplacian $L(\mathcal{G})$ are the graph Fourier frequencies and bases, respectively. Eigenvectors paired with large eigenvalues correspond to high-frequency Fourier bases. The diffusion kernel is defined as

$$k([p],[q]|\beta) = \sum_{i=1}^{n} e^{-\beta\lambda_i} u_i([p]) u_i([q]), \tag{4.1}$$

from which it is clear that higher frequencies, $\lambda_i \gg 1$, are penalized more. In a matrix form, with $\Lambda = diag(\lambda_1, \cdots, \lambda_{|\mathcal{V}|})$ and $\mathbf{U} = [u_1, \cdots, u_{|\mathcal{V}|}]$, the kernel takes the following form:

$$K(\mathcal{V}, \mathcal{V}) = \mathbf{U} \exp(-\beta\Lambda) \mathbf{U}^T, \tag{4.2}$$

which is the Gram matrix on all vertices whose submatrix is the Gram matrix for a subset of vertices.

### 4.2.3 Scalable combinatorial Bayesian optimization with the graph Cartesian product

The direct computation of the diffusion kernel is infeasible because it involves the eigendecomposition of the Laplacian $L(\mathcal{G})$, an operation with cubic complexity with respect to the number of vertices $|\mathcal{V}|$. As we rely on the graph Cartesian product $\square_i \, \mathcal{G}_i$ to construct our combinatorial graph, we can take advantage of its properties and dramatically increase the efficiency of the eigendecomposition of the Laplacian $L(\mathcal{G})$. Further, due to the construction of the combinatorial graph, we can propose a variant of the diffusion kernel: automatic relevance determination (ARD) diffusion kernel. The ARD diffusion kernel has more flexibility in its modeling capacity. Moreover, in combination with the sparsity-inducing Horseshoe prior (Carvalho et al.,

2009) the ARD diffusion kernel performs variable selection automatically that allows to scale to high dimensional problems.

SPEEDING UP THE EIGENDECOMPOSITION WITH GRAPH CARTESIAN PRODUCTS. Direct computation of the eigensystem of the Laplacian $L(\mathcal{G})$ naively is infeasible, even for problems of moderate size. For instance, for 15 binary variables, eigendecomposition complexity is $O(|\mathcal{V}|^3) = (2^{15})^3$.

The graph Cartesian product allows to improve the scalability of the eigendecomposition. The Laplacian of the Cartesian product of two sub-graphs $\mathcal{G}_1$ and $\mathcal{G}_2$, $\mathcal{G}_1 \square \mathcal{G}_2$, can be algebraically expressed using the Kronecker product $\otimes$ and the Kronecker sum $\oplus$ (Hammack et al., 2011):

$$L(\mathcal{G}_1 \square \mathcal{G}_2) = L(\mathcal{G}_1) \oplus L(\mathcal{G}_2) = L(\mathcal{G}_1) \otimes \mathbf{I}_1 + \mathbf{I}_2 \otimes L(\mathcal{G}_2), \tag{4.3}$$

where $\mathbf{I}$ denotes the identity matrix. Considering the eigensystems $\{(\lambda_i^{(1)}, u_i^{(1)})\}$ and $\{(\lambda_j^{(2)}, u_j^{(2)})\}$ of $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively, the eigensystem of $\mathcal{G}_1 \square \mathcal{G}_2$ is $\{(\lambda_i^{(1)} + \lambda_j^{(2)}, u_i^{(1)} \otimes u_j^{(2)})\}$. Given Eq. (4.3) and matrix exponentiation, for the diffusion kernel of $m$ categorical (or ordinal) variables we have

$$\mathbf{K} = \exp\big(-\beta \bigoplus_{i=1}^{m} L(\mathcal{G}_i)\big) = \bigotimes_{i=1}^{m} \exp\big(-\beta\, L(\mathcal{G}_i)\big). \tag{4.4}$$

This means we can compute the kernel matrix by calculating the Kronecker product per sub-graph kernel. Specifically, we obtain the kernel for the $i$-th sub-graph from the eigendecomposition of its Laplacian as per eq. (4.2).

Importantly, the decomposition of the final kernel into the Kronecker product of individual kernels in Eq. (4.4) leads to the following proposition.

**Proposition 4.3.** *Assume a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the graph cartesian product of sub-graphs $\mathcal{G} = \square_i, \mathcal{G}_i$. The graph Fourier Transform of $\mathcal{G}$ can be computed in $O(\sum_{i=1}^{m} |\mathcal{V}_i|^3)$ while the direct computation takes $O(\prod_{i=1}^{m} |\mathcal{V}_i|^3)$.*

*Proof.* See Appx. Prop. B.2. $\square$

VARIABLE-WISE EDGE SCALING. We can make the kernel more flexible by considering individual scaling factors $\{\beta_i\}$, a single $\beta_i$ for each variable. The diffusion kernel then becomes:

$$\mathbf{K} = \exp\big(\bigoplus_{i=1}^{m} -\beta_i\, L(\mathcal{G}_i)\big) = \bigotimes_{i=1}^{m} \exp\big(-\beta_i\, L(\mathcal{G}_i)\big), \tag{4.5}$$

where $\beta_i \geqslant 0$ for $i = 1, \ldots, m$. Since the diffusion kernel is a discrete version of the exponential kernel, the application of the individual $\beta_i$ for each variable is equivalent to the ARD kernel (MacKay, 1994; Neal, 1995). Hence, we can perform variable (sub-graph) selection automatically. We refer to this kernel as the *ARD diffusion kernel*.

**Algorithm 3** COMBO: Combinatorial Bayesian Optimization on the combinatorial graph

```
1: Input: N combinatorial variables $\{C_i\}_{i=1,\cdots,N}$
2: Set a search space and compute Fourier frequencies and bases: # See Subsec. 4.2.2
3: ▷ Set sub-graphs $\mathcal{G}(C_i)$ for each variables $C_i$.
4: ▷ Compute eigensystem $\{(\lambda_k^{(i)}, u_k^{(i)})\}_{i,k}$ for each sub-graph $\mathcal{G}(C_i)$
5: ▷ Construct the combinatorial graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}) = \square_i \mathcal{G}(C_i)$ using graph Cartesian product.
6: Initialize $\mathcal{D}$.
7: repeat
8:     Fit GP using ARD diffusion kernel to $\mathcal{D}$ with slice sampling : $\mu(v_*|\mathcal{D}), \sigma^2(v_*|\mathcal{D})$
9:     Maximize acquisition function : $v_{next} = \text{argmax}_{v_* \in \mathcal{V}}\, a(\mu(v_*|\mathcal{D}), \sigma^2(v_*|\mathcal{D}))$
10:    Evaluate $f(v_{next})$, append to $\mathcal{D} = \mathcal{D} \cup \{(v_{next}, f(v_{next}))\}$
11: until stopping criterion
```

PRIOR ON $\beta_i$. To determine $\beta_i$, and to prevent GP with ARD kernel from overfitting, we apply posterior sampling with a Horseshoe prior (Carvalho et al., 2009) on the $\{\beta_i\}$. The Horseshoe prior encourages sparsity, and, thus, enables variable selection, which, in turn, makes COMBO statistically scalable to high dimensional problems. For instance, if $\beta_i$ is set to zero, then $L(\mathcal{G}_i)$ does not contribute in Eq (4.5).

### 4.2.4 COMBO algorithm

We present the COMBO approach in Algorithm 3. More details about COMBO could be found in the Appx. B.2.

We start the algorithm with defining all sub-graphs. Then, we calculate GFT (line 4 of Alg. 3), whose result is needed to compute the ARD diffusion kernel, which could be sped up due to the application of the graph Cartesian product. Next, we fit the surrogate model parameters using slice sampling (Murray and Adams, 2010; Neal, 2003) (line 8). Sampling begins with 100 steps of the burn-in phase. With the updated $\mathcal{D}$ of evaluated data, 10 points are sampled without thinning. More details on the surrogate model fitting are given in Appx. B.2.

Last, we maximize the acquisition function to find the next point for evaluation (line 9). For this purpose, we begin with evaluating 20,000 randomly selected vertices. Twenty vertices with highest acquisition values are used as initial points for acquisition function optimization. We use the breadth-first local search (BFLS), where at a given vertex we compare acquisition values on adjacent vertices. We then move to the vertex with the highest acquisition value and repeat until no acquisition value on adjacent vertices are higher than the acquisition value at the current vertex. BFLS is a local search, however, the initial random search and multi-starts help to escape from local minima. In experiments (Appx. B.2) we found that BFLS performs on par or better than non-local search, while being more efficient.

In our framework we can use any acquisition function like GP-UBC, the Expected Improvement (EI) (Jones et al., 1998), the predictive entropy search (J. M. Hernández-Lobato, M. W. Hoffman, et al., 2014) or knowledge gradient (Wu, Poloczek, et al., 2017). We opt for EI that generally works well in practice (Shahriari et al., 2015).

## 4.3 RELATED WORKS

While for continuous inputs, $\mathcal{X} \subseteq \mathbb{R}^D$, there exist efficient algorithms to cope with high-dimensional search spaces using Gaussian processes(GPs) (ChangYong Oh et al., 2018) or neural networks (Snoek, Rippel, et al., 2015), few Bayesian Optimization(BO) algorithms have been proposed for combinatorial search spaces (Baptista and Poloczek, 2018; J. Bergstra, Yamins, et al., 2013; Hutter et al., 2011).

A basic BO approach to combinatorial inputs is to represent all combinatorial variables using one-hot encoding and treating all integer-valued variables as values on a real line. Further, for the integer-valued variables an acquisition function considers the closest integer for the chosen real value. This approach is used in Spearmint (Snoek, Larochelle, et al., 2012). However, applying this method naively may result in severe problems, namely, the acquisition function could repeatedly evaluate the same points due to rounding real values to an integer and the one-hot representation of categorical variables. As pointed out in (Garrido-Merchán and D. Hernández-Lobato, 2020), this issue could be fixed by making the objective constant over regions of input variables for which the actual objective has to be evaluated. The method was presented on a synthetic problem with two integer-valued variables, and a problem with one categorical variable and one integer-valued variable. Unfortunately, it remains unclear whether this approach is suitable for high-dimensional problems. Additionally, the proposed transformation of the covariance function seems to be better suited for ordinal-valued variables rather than categorical variables, further restricting the utility of this approach. In contrast, we propose a method that can deal with high-dimensional combinatorial (categorical and/or ordinal) spaces.

Another approach to combinatorial optimization was proposed in BOCS (Baptista and Poloczek, 2018) where the sparse Bayesian linear regression was used instead of GPs. The acquisition function was optimized by a semi-definite programming or simulated annealing that allowed to speed up the procedure of picking new points for next evaluations. However, BOCS has certain limitations which restrict its application mostly to problems with low order interactions between variables. BOCS requires users to specify the highest order of interactions among categorical variables, which inevitably ignores interaction terms of orders higher than the user-specified order. Moreover, due to its parametric nature, the surrogate model of BOCS has excessively large number of parameters even for moderately high order (e.g. up to the 4th or 5th order). Nevertheless, this approach achieved state-of-the-art results on four high-dimensional binary optimization problems. Different from (Baptista and Poloczek, 2018), we use a non-parametric regression, i.e. GPs and perform variable selection both of which give more statistical efficiency.

## 4.4 EXPERIMENTS

We evaluate COMBO on two binary variable benchmarks, one ordinal and one multi-categorical variable benchmarks, as well as in two realistic problems: weighted Maximum Satisfiability and Neural Architecture Search. We convert all into minimization problems. We compare SMAC (Hutter et al., 2011), TPE (J. Bergstra, Yamins, et al., 2013), Simulated Annealing (SA) (Spears, 1993), as well as with BOCS (BOCS-SDP and BOCS-SA3)[11] (Baptista and Poloczek, 2018). All details regarding experiments, baselines and results are in Appx. B.2 and B.4.

### 4.4.1 BO with binary variables

**Table 4.1**: Results on the binary benchmarks (25 runs)

| | Contamination control | | | Ising sparsification | | |
|---|---|---|---|---|---|---|
| Method | $\lambda = 0$ | $\lambda = 10^{-4}$ | $\lambda = 10^{-2}$ | $\lambda = 0$ | $\lambda = 10^{-4}$ | $\lambda = 10^{-2}$ |
| SMAC | 21.61±0.04 | 21.50±0.03 | 21.68±0.04 | 0.152±0.040 | 0.219±0.052 | 0.350±0.045 |
| TPE | 21.64±0.04 | 21.69±0.04 | 21.84±0.04 | 0.404±0.109 | 0.444±0.095 | 0.609±0.107 |
| SA | 21.47±0.04 | 21.49±0.04 | 21.61±0.04 | **0.095**±0.033 | 0.117±0.035 | 0.334±0.064 |
| BOCS-SDP | 21.37±0.03 | 21.38±0.03 | 21.52±0.03 | 0.105±0.031 | **0.059**±0.013 | **0.300**±0.039 |
| COMBO | **21.28**±0.03 | **21.28**±0.03 | **21.44**±0.03 | 0.103±0.035 | 0.081±0.028 | 0.317±0.042 |

CONTAMINATION CONTROL The contamination control in food supply chain is a binary optimization problem with 21 binary variables ($\approx 2.10 \times 10^6$ configurations) (Hu et al., 2010), where one can intervene at each stage of the supply chain to quarantine uncontaminated food with a cost. The goal is to minimize food contamination while minimizing the prevention cost. We set the budget to 270 evaluations

11 We exclude BOCS from ordinal/multi-categorical experiments, because at the time of the submission of this chapter to NeurIPS 2019, the open source implementation provided by the authors did not support ordinal/multi-categorical variables. For the explanation on how to use BOCS for ordinal/multi-categorical variables, please refer to the supplementary material of (Baptista and Poloczek, 2018).

including 20 random initial points. We report results in Tab. 4.1 and figures in Appx. B.4. COMBO outperforms all competing methods. Although the optimizing variables are binary, there exist higher order interactions among the variables due to the sequential nature of the problem, showcasing the importance of the modelling flexibility of COMBO.

ISING SPARSIFICATION A probability mass function(p.m.f) $p(x)$ can be defined by an Ising model $I_p$. In Ising sparsification, we approximate the p.m.f $p(z)$ of $I_p$ with a p.m.f $q(z)$ of $I_q$. The objective is the KL-divergence between p and q with a λ-parameterized regularizer: $\mathcal{L}(x) = D_{KL}(p||q) + \lambda\|x\|_1$. We consider 24 binary variable Ising models on $4 \times 4$ spin grid ($\approx 1.68 \times 10^7$ configurations) with a budget of 170 evaluations, including 20 random initial points. We report results in Tab. 4.1 and figures in Appx. B.4.1. We observe that COMBO is competitive, obtaining slightly worse results, probably because in Ising sparsification there exist no complex interactions between variables.

### 4.4.2 BO with ordinal and multi-categorical variables

ORDINAL VARIABLES The Branin benchmark is an optimization problem of a non-linear function over a 2D search space (Jones et al., 1998). We discretize the search space, namely, we consider a grid of points that leads to an optimization problem with ordinal variables. We set the budget to 100 evaluations and report results in Appx. B.4.2. COMBO converges to a better solution faster and with better stability.

MULTI-CATEGORICAL VARIABLES The Pest control is a modified version of the contamination control with more complex, higher-order variable interactions, as detailed in Appx. B.4.2. We consider 21 pest control stations, each having 5 choices ($\approx 4.77 \times 10^{14}$ combinatorial choices). We set the budget to 320 including 20 random initial points. Results are in in the Appx. B.4.2 COMBO outperforms all methods and converges faster.

**Table 4.2:** Results on the non-binary benchmarks (25 runs)

| Method | Branin (Ordinal) | Pest Control (Multi-categorical) |
|---|---|---|
| SMAC | 0.6962±0.0705 | 14.2614±0.0753 |
| TPE | 0.7578±0.0844 | 14.9776±0.0446 |
| SA | 0.4659±0.0211 | 12.7154±0.0918 |
| COMBO | **0.4113**±0.0012 | **12.0012**±0.0033 |

[*] We exclude BOCS, as the open source implementation provided by the authors does not support ordinal/multi-categorical variables.

### 4.4.3 Weighted maximum satisfiability

The satisfiability (SAT) problem is an important combinatorial optimization problem, where one decides how to set variables of a Boolean formula to make the formula true. Many other optimization problems can be reformulated as SAT/MaxSAT problems. Although highly successful, specialized MaxSAT solvers (Bacchus et al., 2018) exist, we use MaxSAT as a testbed for BO evaluation. We run tests on three benchmarks from the Maximum Satisfiability Competition 2018.[12] The wMaxSAT weights are unit normalized. All evaluations are negated to obtain a minimization problem. We set the budget to 270 evaluations including 20 random initial points. We report results in Tab. 4.3 (Mean $\pm$ Std.Err. over 25 runs) and figures in Appx. B.4.3, and runtimes on wMaxSAT43 in Fig. 4.2 on wMaxSAT28 (Appx. Fig. B.12)[13]

**Table 4.3:** Negated wMaxSAT Minimum (25 runs)

| Method | wMaxSAT28 | wMaxSAT43 | wMaxSAT60 |
|---|---|---|---|
| SMAC | -20.05$\pm$0.67 | -57.42$\pm$1.76 | -148.60$\pm$1.01 |
| TPE | -25.20$\pm$0.88 | -52.39$\pm$1.99 | -137.21$\pm$2.83 |
| SA | -31.81$\pm$1.19 | -75.76$\pm$2.30 | -187.55$\pm$1.50 |
| BOCS-SDP | -29.49$\pm$0.53 | -51.13$\pm$1.69 | -153.67$\pm$2.01 |
| BOCS-SA3 | -34.79$\pm$0.78 | -61.02$\pm$2.28[a] | N.A[b] |
| COMBO | **-37.80**$\pm$0.27 | **-85.02**$\pm$2.14 | **-195.65**$\pm$0.00 |

[a] 270 evaluations were not finished after 168 hours.
[b] Not tried due to the computation time longer than wMaxSAT43.

COMBO performs best in all cases. BOCS benefits from third-order interactions on wMaxSAT28 and wMaxSAT43. However, this comes at the cost of large number of parameters (Baptista and Poloczek, 2018), incurring expensive computations. When considering higher-order terms BOCS suffers severely from inefficient training. This is due to a bad ratio between the number of parameters and number of training samples (e.g. for the 43 binary variables case BOCS-SA3/SA4/SA5 with, respectively, 3rd/4th/5th order interactions, has 13288/136698/1099296 parameters to train). In contrast, COMBO models arbitrarily high order interactions thanks to GP's nonparametric nature in a statistically efficient way.

Focusing on the largest problem, wMaxSAT60 with $\approx 1.15 \times 10^{18}$ configurations, COMBO maintains superior performance. We attribute this to the sparsity-inducing properties of the Horseshoe prior, after

12 https://maxsat-evaluations.github.io/2018/benchmarks.html
13 The all runtimes were measured on Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz with python codes.

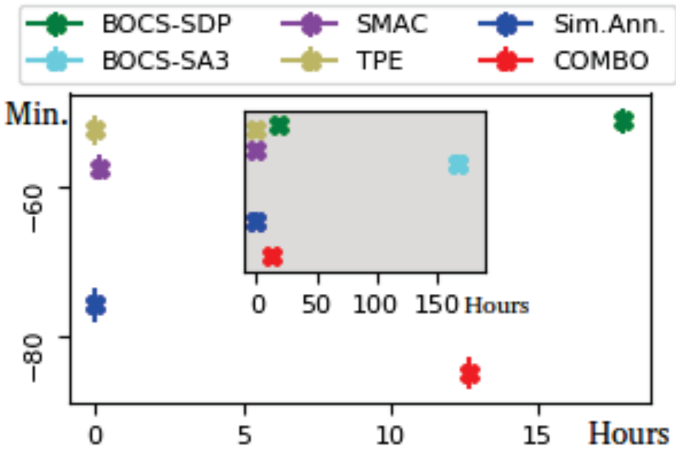


Figure 4.2: Runtime vs Minimum on wMaxSAT43

examining non sparsity-inducing priors (Fig. B.11). The Horseshoe prior helps COMBO attain further statistical efficiency. We can interpret this reductionist behavior as the combinatorial version of methods exploiting low-effective dimensionality (J. Bergstra and Bengio, 2012) on continuous search spaces (Ziyu Wang, Hutter, et al., 2016).

The runtime – including evaluation time – was measured on a dual 8-core 2.4 GHz (Intel Haswell E5-2630-v3) CPU with 64 GB memory using Python implementations. SA, SMAC and TPE are faster but inaccurate compared to BOCS. COMBO is faster than BOCS-SA3, which needed 168 hours to collect around 200 evaluations. COMBO – modelling arbitrarily high-order interactions – is also faster than BOCS-SDP constrained up-to second-order interactions only.

We conclude that in the realistic maximum satisfiablity problem COMBO yields accurate solutions in reasonable runtimes, easily scaling up to high dimensional combinatorial search problems.

### 4.4.4 Neural architecture search

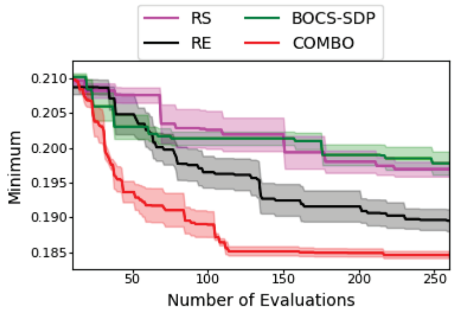


Figure 4.3: Neural Architecture Search (4 runs)

Last, we compare BO methods on a neural architecture search (NAS) problem, a typical combinatorial optimization problem (Wistuba, Rawat, et al., 2019). We compare COMBO with BOCS, as well as Regularized Evolution (RE) (Real et al., 2019), one of the most successful evolutionary search algorithm for NAS (Wistuba, Rawat, et al., 2019). We include Random Search (RS) which can be competitive in well-designed search spaces (Wistuba, Rawat, et al., 2019). We do not compare with the BO-based NASBOT (Kandasamy, Neiswanger, et al., 2018). NASBOT focuses exclusively on NAS problems and optimizes over a different search space than ours using an optimal transport-based metric between architectures, which is out of the scope for this work.

**Table 4.4:** Neural Architecture search space

Connectivity
(O - connected, X - disconnected)

| | **IN** | **H1** | **H2** | **H3** | **H4** | **H5** | **OUT** |
|---|---|---|---|---|---|---|---|
| **IN** | - | O | X | X | X | O | X |
| **H1** | - | - | X | O | X | X | O |
| **H2** | - | - | - | X | O | X | X |
| **H3** | - | - | - | - | X | O | X |
| **H4** | - | - | - | - | - | O | O |
| **H5** | - | - | - | - | - | - | X |
| **OUT** | - | - | - | - | - | - | - |

Computation type

| | **MaxPool** | **Conv** |
|---|---|---|
| **Small** | Id ≡ MaxPool(1×1) | Conv(3×3) |
| **Large** | MaxPool(3×3) | Conv(5×5) |

For the considered NAS problem we aim at finding the optimal cell comprising of one input node (**IN**), one output node (**OUT**) and five possible hidden nodes (**H1–H5**). We allow connections from **IN** to all other nodes, from **H1** to all other nodes and so one. We exclude connections that could cause loops. An example of connections within a cell can be found in Tab. 4.4 on the left, where the input state **IN** connects to **H1**, **H1** connects to **H3** and OUT, and so on. The input state and output state have identity computation types, whereas the computation types for the hidden states are determined by combination of 2 binary choices from the table on the right of Tab. 4.4. In total, the search space consists of 31 binary variables, 21 for the connectivities and 2 for 5 computation types.

The objective is to minimize the classification error on validation set of CIFAR10 (Krizhevsky and Hinton, 2009) with a penalty on the amount of FLOPs of a neural network constructed with a given cell. We search for an architecture that balances accuracy and computational efficiency. In each evaluation, we construct a cell, and stack three cells to build a final neural network. More details are given in the Appx. B.3.

In Fig. 4.3 we can notice that COMBO outperforms other methods significantly. BOCS-SDP and RS exhibit similar performance, confirming that for NAS modeling high-order interactions between variables is crucial. Furthermore, COMBO outperforms the specialized RE, one of the most successful evolutionary search (ES) algorithms shown to perform better on NAS than reinforcement learning (RL) algorithms (Real et al., 2019; Wistuba, Rawat, et al., 2019). When increasing the number of evaluations to 500, RE still cannot reach the performance of COMBO with 260 evaluations, see Appx. Fig. B.15. A possible explanation for such behavior is the high sensitivity to choices of hyperparameters of RE, and ES requires far more evaluations in general. Details about RE hyperparameters can be found in the Appx. B.4.4.

Due to the difficulty of using BO on combinatoral structures, BOs have not been widely used for NAS with few exceptions (Kandasamy, Neiswanger, et al., 2018). COMBO's performance suggests that a well-designed general combinatorial BO can be competitive or even better in NAS than ES and RL, especially when computational resources are constrained. Since COMBO is applicable to any set of combinatorial variables, its use in NAS is not restricted to the typical NASNet search space. Interestingly, COMBO can approximately optimize continuous variables by discretization, as shown in the ordinal variable experiment, thus, jointly optimizing the architecture and hyperparameter learning.

## 4.5 CONCLUSION

In this work, we propose COMBO, a Bayesian Optimization method for combinatorial search spaces. To the best of our knowledge, COMBO is the first Bayesian Optimization algorithm using Gaussian Processes as a surrogate model suitable for problems with complex high order interactions between variables. To efficiently tackle the exponentially increasing complexity of combinatorial search spaces, we rest upon the following ideas: *(i)* we represent the search space as the combinatorial graph, which combines sub-graphs given to all combinatorial variables using the graph Cartesian product. Moreover, the combinatorial graph reflects a natural metric on categorical choices (Hamming distance) when all combinatorial variables are categorical. *(ii)* we adopt the GFT to define the "smoothness" of functions on combinatorial structures. *(iii)* we propose a flexible ARD diffusion kernel for GPs on the combinatorial graph with a Horseshoe prior on scale parameters, which makes COMBO scale up to high dimensional problems by performing variable selection. All above features together make that COMBO outperforms competitors consistently on a wide range of problems. COMBO is a statistically and computationally scalable Bayesian Optimization tool for combinatorial spaces, which is a field that has not been extensively explored.

# 5 MIXED VARIABLE BAYESIAN OPTIMIZATION WITH FREQUENCY MODULATED KERNELS

## 5.1 INTRODUCTION

Bayesian optimization has found many applications ranging from daily routine level tasks of finding a tasty cookie recipe (Solnik et al., 2017) to sophisticated hyperparameter optimization tasks of machine learning algorithms (e.g. Alpha-Go (Y. Chen, A. Huang, et al., 2018)). Much of this success is attributed to the flexibility and the quality of uncertainty quantification of Gaussian Process (GP)-based surrogate models (ChangYong Oh et al., 2018; Snoek, Larochelle, et al., 2012; K. Swersky, Snoek, et al., 2013).

Despite the superiority of GP surrogate models, as compared to non-GP ones, their use on spaces with discrete structures (*e.g.*, chemical spaces (Reymond and Awale, 2012), graphs and even mixtures of different types of spaces) is still application-specific (Kandasamy, Neiswanger, et al., 2018; Korovina et al., 2020). The main reason is the difficulty of defining kernels flexible enough to model dependencies across different types of variables. On mixed variable spaces which consist of different types of variables including continuous, ordinal and nominal variables, current BO approaches resort to non-GP surrogate models, such as simple linear models or linear models with manually chosen basis functions (Daxberger et al., 2021). However, such linear approaches are limited because they may lack the necessary model capacity.

There is much progress on BO using GP surrogate models (GP BO) for continuous, as well as for discrete variables. However, for mixed variables it is not straightforward how to define kernels ,which can model dependencies across different types of variables. To bridge the gap, we propose *frequency modulation* which uses distances on continuous variables to modulate the frequencies of the graph spectrum (Ortega et al., 2018) where the graph represents the discrete part of the search space (Changyong Oh, Tomczak, et al., 2019).

A potential problem in the frequency modulation is that it does not always define a kernel with the similarity measure behavior (Vert et al., 2004). That is, the frequency modulation does not necessarily define a kernel that returns higher values for pairs of more similar points. Formally, for a stationary kernel $k(x, y) = s(x - y)$, $s$ should be decreasing (Remes et al., 2017). In order to guarantee the similarity measure behavior of kernels constructed by frequency modulation, we stipulate a condition, the *frequency modulation principle*. Theoretical analysis results in proofs of the positive definiteness as well as the effect of the frequency modulation principle. We coin frequency modulated (FM) kernels as the kernels constructed by frequency modulation and respecting the frequency modulation principle.

Different to methods that construct kernels on mixed variables by kernel addition and kernel multiplication, for example, FM kernels do not impose an independence assumption among different types of variables. In FM kernels, quantities in the two domains, that is the distances in a spatial domain and the frequencies in a Fourier domain, interact. Therefore, the restrictive independence assumption is circumvented, and thus flexible modeling of mixed variable functions is enabled.

In this chapter, *(i)* we propose frequency modulation, a new way to construct kernels on mixed variables, *(ii)* we provide the condition to guarantee the similarity measure behavior of FM kernels together with a theoretical analysis, and *(iii)* we extend frequency modulation so that it can model complex dependencies between arbitrary types of variables. In experiments, we validate the benefit of the increased modeling capacity of FM kernels and the importance of the frequency modulation principle for improved sample efficiency on different mixed variable BO tasks. We also test BO with GP using FM kernels (BO-FM) on a challenging joint optimization of the neural architecture and the hyperparameters with two strong baselines, Regularized Evolution (RE) (Real et al., 2019) and BOHB (Falkner et al., 2018). BO-FM outperforms both baselines which have proven their competence in neural architecture search (X. Dong, L. Liu, et al., 2021). Remarkably, BO-FM outperforms RE with three times evaluations.

## 5.2 PRELIMINARIES

### 5.2.1 Bayesian Optimization with Gaussian Processes

Bayesian optimization (BO) aims at finding the global optimum of a black-box function $g$ over a search space $\mathcal{X}$. At each round BO performs an evaluation $y_i$ on a new point $\mathbf{x}_i \in \mathcal{X}$, collecting the set of evaluations $\mathcal{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1,\cdots,t}$ at the t-th round. Then, a surrogate model approximates the function $g$ given $\mathcal{D}_t$ using the predictive mean $\mu(\mathbf{x}_* \,|\, \mathcal{D}_t)$ and the predictive variance $\sigma^2(\mathbf{x}_* \,|\, \mathcal{D}_t)$. Now, an acquisition function $r(\mathbf{x}_*) = r(\mu(\mathbf{x}_* \,|\, \mathcal{D}_t), \sigma^2(\mathbf{x}_* \,|\, \mathcal{D}_t))$ quantifies how informative input $\mathbf{x} \in \mathcal{X}$ is for the purpose of finding the global optimum. $g$ is then evaluated at $\mathbf{x}_{t+1} = \operatorname{argmax}_{\mathbf{x} \in \mathcal{X}} r(\mathbf{x})$, $y_{t+1} = g(\mathbf{x}_{t+1})$. With the updated set of evaluations, $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(\mathbf{x}_{t+1}, y_{t+1})\}$, the process is repeated.

A crucial component in BO is thus the surrogate model. Specifically, the quality of the predictive distribution of the surrogate model is critical for balancing the exploration-exploitation trade-off (Shahriari et al., 2015). Compared with other surrogate models (such as Random Forest (Hutter et al., 2011) and a tree-structured density estimator (J. S. Bergstra et al., 2011)), Gaussian Processes (GPs) tend to yield better results (ChangYong Oh et al., 2018; Snoek, Larochelle, et al., 2012).

For a given kernel $k$ and data $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ where $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]^T$ and $\mathbf{y} = [y_1, \cdots, y_n]^T$, a GP has a predictive mean $\mu(\mathbf{x}_* | \mathbf{X}, \mathbf{y}) = k_{*\mathbf{X}}(k_{\mathbf{X}\mathbf{X}} + \sigma^2 I)^{-1}\mathbf{y}$ and predictive variance $\sigma^2(\mathbf{x}_* | \mathbf{X}, \mathbf{y}) = k_{**} - k_{*\mathbf{X}} \cdot (k_{\mathbf{X}\mathbf{X}} + \sigma^2 I)^{-1} k_{\mathbf{X}*}$ where $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$, $[k_{*\mathbf{X}}]_{1,i} = k(\mathbf{x}_*, \mathbf{x}_i)$, $k_{\mathbf{X}*} = (k_{*\mathbf{X}})^T$ and $[k_{\mathbf{X}\mathbf{X}}]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$.

### 5.2.2 Kernels on discrete variables

We first review some kernel terminology (Scholkopf and Smola, 2001) that is needed in the rest of the chapter.

**Definition 5.1** (Gram Matrix)**.** Given a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and data $x_1, \cdots, x_n \in \mathcal{X}$, the $n \times n$ matrix $K$ with elements $[K]_{ij} = k(x_i, x_j)$ is called the Gram matrix of $k$ with respect to $x_1, \cdots, x_n$.

**Definition 5.2** (Positive Definite Matrix)**.** A real $n \times n$ matrix $K$ satisfying $\sum_{i,j} a_i [K]_{ij} a_j \geqslant 0$ for all $a_i \in \mathbb{R}$ is called positive definite (PD)[14].

**Definition 5.3** (Positive Definite Kernel)**.** A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which gives rise to a positive definite Gram matrix for all $n \in \mathbb{N}$ and all $x_1, \cdots, x_n \in \mathcal{X}$ is called a positive definite (PD) kernel, or simply a kernel.

A search space which consists of discrete variables, including both nominal and ordinal variables, can be represented as a graph (R. I. Kondor and Lafferty, 2002; Changyong Oh, Tomczak, et al., 2019). In this graph each vertex represents one state of exponentially many joint states of the discrete variables. The edges represent relations between these states (e.g. if they are similar) (Changyong Oh, Tomczak, et al., 2019). With a graph representing a search space of discrete variables, kernels on a graph can be used for BO. In (Smola and R. Kondor, 2003), for a positive decreasing function $f$ and a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose graph Laplacian $L(\mathcal{G})$[15] has the eigendecomposition $U\Lambda U^T$, it is shown that a kernel can be defined as

$$k_{disc}(v, v'|\beta) = [U f(\Lambda|\beta) U^T]_{v,v'} \tag{5.1}$$

14 Sometimes, different terms are used, semi-positive definite for $\sum_{i,j} a_i [K]_{ij} a_j \geqslant 0$ and positive definite for $\sum_{i,j} a_i [K]_{ij} a_j > 0$. Here, we stick to the definition in (Scholkopf and Smola, 2001).

15 In this chapter, we use a (unnormalized) graph Laplacian $L(\mathcal{G}) = D - A$ while, in (Smola and R. Kondor, 2003), symmetric normalized graph Laplacian, $L^{sym}(\mathcal{G}) = D^{-1/2}(D - A)D^{-1/2}$. ($A$ : adj. mat. / $D$ : deg. mat.) Kernels are defined for both.

where $\beta \geqslant 0$ is a kernel parameter and f is a positive decreasing function. It is the reciprocal of a regularization operator (Smola and R. Kondor, 2003) which penalizes high frequency components in the spectrum.

## 5.3 METHODS

With the goal of obtaining flexible kernels on mixed variables which can model complex dependencies across different types of variables, we propose the frequency modulated (FM) kernel. Our objective is to enhance the modelling capacity of GP surrogate models and, thereby improve the sample efficiency of mixed-variable BO. FM kernels use the continuous variables to modulate the frequencies of the kernel of discrete variables defined on the graph. As a consequence, FM kernels can model complex dependencies between continuous and discrete variables. Specifically, let us start with continuous variables of dimension $D_{\mathcal{C}}$, and discrete variables represented by the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose graph Laplacian $L(\mathcal{G})$ has eigendecompostion $U\Lambda U^T$. To define a frequency modulated kernel we consider the function $k : (\mathbb{R}^{D_{\mathcal{C}}} \times \mathcal{V}) \times (\mathbb{R}^{D_{\mathcal{C}}} \times \mathcal{V}) \Rightarrow \mathbb{R}$ of the following form

$$k((\mathbf{c}, v), (\mathbf{c}', v') | \beta, \boldsymbol{\theta}) = \sum_{i=1}^{|\mathcal{V}|} [U]_{v,i} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta) [U]_{v',i} \tag{5.2}$$

where $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2 = \sum_{d=1}^{D_{\mathcal{C}}} (c_d - c'_d)^2 / \theta_d^2$ and $(\boldsymbol{\theta}, \beta)$ are tunable parameters. f is the frequency modulating function defined below in Def. 5.4.

The function f in Eq. (5.2) takes frequency $\lambda_i$ and distance $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2$ as arguments, and its output is combined with the basis $[U]_{v,i}$. That is, the function f processes the information in each eigencomponent separately while Eq. (5.2) then sums up the information processed by f. Note that unlike kernel addition and kernel product,[16], the distance $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2$ influences each eigencomponent separately as illustrated in Fig. 5.1. Unfortunately, Eq. (5.2) with an arbitrary function f does not always define a positive definite kernel. Moreover, Eq. (5.2) with an arbitrary function f may return higher kernel values for less similar points, which is not expected from a proper similarity measure (Vert et al., 2004). To this end, we first specify three properties of functions f such that Eq. (5.2) guaranteed to be a positive definite kernel and a proper similarity measure at the same time. Then, we motivate the necessity of each of the properties in the following subsections.

16 e.g $k_{add}((\mathbf{c}, v), (\mathbf{c}', v')) = e^{-\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2} + k_{disc}(v, v')$ and $k_{prod}((\mathbf{c}, v), (\mathbf{c}', v')) = e^{-\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2} \cdot k_{disc}(v, v')$

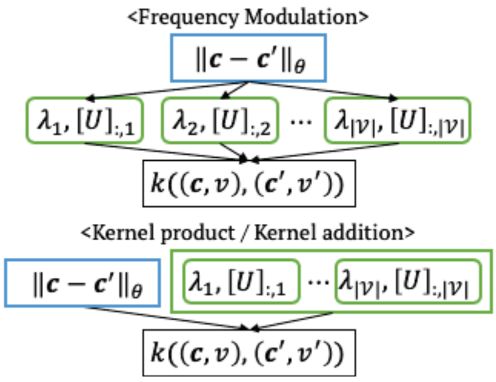


Figure 5.1: Influence on eigencomponents

**Definition 5.4** (Frequency modulating function)**.** A frequency modulating function is a function $f : \mathbb{R}^+ \times \mathbb{R} \to \mathbb{R}$ satisfying the three properties below.

**FM-P1** For a fixed $t \in \mathbb{R}$, $f(s, t)$ is a positive and decreasing function with respect to $s$ on $[0, \infty)$.

**FM-P2** For a fixed $s \in \mathbb{R}^+$, $f(s, \| \mathbf{c} - \mathbf{c}' \|_\theta)$ is a positive definite kernel on $(\mathbf{c}, \mathbf{c}') \in \mathbb{R}^{D_e} \times \mathbb{R}^{D_e}$.

**FM-P3** For $t_1 < t_2$, $h_{t_1,t_2}(s) = f(s, t_1) - f(s, t_2)$ is positive, strictly decreasing and convex w.r.t $s \in \mathbb{R}^+$.

**Definition 5.5** (FM kernel)**.** A FM kernel is a function on $(\mathbb{R}^{D_e} \times \mathcal{V}) \times (\mathbb{R}^{D_e} \times \mathcal{V})$ of the form in Eq. (5.2), where $f$ is a frequency modulating function on $\mathbb{R}^+ \times \mathbb{R}$.

### 5.3.1 Frequency Regularization of FM kernels

In (Smola and R. Kondor, 2003), it is shown that Eq. (5.1) defines a kernel that regularizes the eigenfunctions with high frequencies when $f$ is positive and decreasing. It is also shown that the reciprocal of $f$ in Eq. (5.1) is a corresponding regularization operator. For example, the diffusion kernel defined with $f(\lambda) = \exp(-\beta\lambda)$ corresponds to the regularization operator $r(\lambda) = \exp(\beta\lambda)$. The regularized Laplacian kernel defined with $f(\lambda) = 1/(1 + \beta\lambda)$ corresponds to the regularization operator $r(\lambda) = 1 + \beta\lambda$. Both regularization operators put more penalty on higher frequencies $\lambda$.

Therefore, the property **FM-P1** forces FM kernels to have the same regularization effect of promoting a smoother function by penalizing the eigenfunctions with high frequencies.

### 5.3.2 Positive Definiteness of FM kernels

Determining whether Eq.5.2 defines a positive definite kernel is not trivial. The reason is that the gram matrix $[k((\mathbf{c}_i, \nu_i), (\mathbf{c}_j, \nu_j))]_{i,j}$ is not determined only by the entries $\nu_i$ and $\nu_j$, but these entries are additionally affected by different distance terms $\| \mathbf{c}_i - \mathbf{c}_j \|_{\boldsymbol{\theta}}$. To show that FM kernels are positive definite, it is sufficient to show that $f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta)$ is positive definite on $(\mathbf{c}, \mathbf{c}') \in \mathbb{R}^{D_e} \times \mathbb{R}^{D_e}$.

**Theorem 5.1.** *If* $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta)$ *defines a positive definite kernel with respect to* $\mathbf{c}$ *and* $\mathbf{c}'$*, then the FM kernel with such* $f$ *is positive definite* ***jointly*** *on* $\mathbf{c}$ *and* $\nu$*. That is, the positive definiteness of* $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta)$ *on* $\mathbb{R}^{D_e}$ *implies the positive definiteness of the FM kernel on* $\mathbb{R}^{D_e} \times \mathcal{V}$*.*

*Proof.* See Appx. Thm. C.1. □

Note that Thm. 5.1 shows that the property **FM-P2** guarantees that FMs kernels are positive definite jointly on $\mathbf{c}$ and $\nu$.

In the current form of Thm. 5.1, the frequency modulating functions depend on the distance $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$. However, the proof does not change for the more general form of $f(\lambda, \mathbf{c}, \mathbf{c}' | \alpha, \beta)$, where $f$ does not depend on $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$. Hence, Thm. 5.1 can be extended to the more general case that $f(\lambda, \mathbf{c}, \mathbf{c}' | \alpha, \beta)$ is positive definite on $(\mathbf{c}, \mathbf{c}') \in \mathbb{R}^{D_e} \times \mathbb{R}^{D_e}$.

### 5.3.3 Frequency Modulation Principle

A kernel, as a similarity measure, is expected to return higher values for pairs of more similar points and vice versa (Vert et al., 2004). We call such behavior the *similarity measure behavior*.

In Eq. (5.2), the distance $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$ represents a quantity in the "spatial" domain interacting with quantities $\lambda_i$s in the "frequency" domain. Due to the interplay between the two different domains, the kernels of the form Eq. (5.2) do not exhibit the similarity measure behavior *for an arbitrary function* $f$. Next, we derive a sufficient condition on $f$ for the similarity measure behavior to hold for FM kernels.

Formally, the similarity measure behavior is stated as

$$\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \leqslant \|\tilde{\mathbf{c}} - \tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}} \quad \Rightarrow \quad k((\mathbf{c}, \nu), (\mathbf{c}', \nu')) \geqslant k((\tilde{\mathbf{c}}, \nu), (\tilde{\mathbf{c}}', \nu')) \quad (5.3)$$

or equivalently,

$$\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \leqslant \|\tilde{\mathbf{c}} - \tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}} \quad \Rightarrow \quad \sum_{i=1}^{|\mathcal{V}|} [U]_{\nu,i} h_{t_1,t_2}(\lambda_i|\beta) [U]_{\nu',i} \geqslant 0 \qquad (5.4)$$

where $h_{t_1,t_2}(\lambda|\beta) = f(\lambda, t_1|\beta) - f(\lambda, t_2|\beta)$, $t_1 = \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$ and $t_2 = \|\tilde{\mathbf{c}} - \tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}}$.

**Theorem 5.2.** *For a connected and weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with non-negative weights on edges, define a similarity (or kernel) $a(v, v') = [Uh(\Lambda)U^T]_{v,v'}$, where $U$ and $\Lambda$ are eigenvectors and eigenvalues of the graph Laplacian $L(\mathcal{G}) = U\Lambda U^T$. If $h$ is any non-negative and strictly decreasing convex function on $[0, \infty)$, then $a(v, v') \geqslant 0$ for all $v, v' \in \mathcal{V}$.*

Therefore, these conditions on $h(\Lambda)$ result in a similarity measure $a$ with only positive entries, which in turn proves property Eq. (5.4). Here, we provide a proof of the theorem for a simpler case with an unweighted complete graph, where Eq. (5.4) holds without the convexity condition on $h$.

*Proof.* For a unweighted complete graph with $n$ vertices, we have eigenvalues $\lambda_1 = 0$, $\lambda_2 = \cdots = \lambda_n = n$ and eigenvectors such that $[U]_{\cdot 1} = 1/\sqrt{n}$ and $\sum_{i=1}^{n} [U]_{v,i}[U]_{v',i} = \delta_{vv'}$. For $v \neq v'$, the conclusion in Eq. (5.4), $\sum_{i=1}^{n} h(\lambda_i)[U]_{v,i}[U]_{v',i}$ becomes $h(0)/n + h(n) \cdot \sum_{i=2}^{n} [U]_{v,i}[U]_{v',i} = (h(0) - h(n))/n$ in which non-negativity follows with decreasing $h$. For the complete proof, see Appx. Thm. C.2. □

Thm. 5.2 thus shows that the property **FM-P3** is sufficient for Eq. (5.4) to hold. We call the property **FM-P3** the *frequency modulation principle*. Thm. 5.2 also implies the non-negativity of many kernels derived from graph Laplacian.

**Corollary 5.3.** *The random walk kernel derived from the symmetric normalized Laplacian (Smola and R. Kondor, 2003), the diffusion kernels (R. I. Kondor and Lafferty, 2002; Changyong Oh, Tomczak, et al., 2019) and the regularized Laplacian kernel (Smola and R. Kondor, 2003) derived from symmetric normalized or unnormalized Laplacian, are all non-negatived valued.*

*Proof.* See Appx. Coro. C.3. □

### 5.3.4 FM kernels in practice

SCALABILITY Since the (graph Fourier) frequencies and basis functions are computed by the eigendecomposition of cubic computational complexity, a plain application of frequency modulation makes the computation of FM kernels prohibitive for a large number of discrete variables. Given $P$ discrete variables where each variable can be individually represented by a graph $\mathcal{G}_p$, the discrete part of the search space can be represented as a product space, $\mathcal{V} = \mathcal{V}_1 \times \cdots \times \mathcal{V}_P$.

In this case, we define FM kernels on $\mathbb{R}^{D_e} \times \mathcal{V} = \mathbb{R}^{D_e} \times (\mathcal{V}_1 \times \cdots \times \mathcal{V}_P)$ as

$$\begin{aligned} k((\mathbf{c}, \mathbf{v}), (\mathbf{c}', \mathbf{v}') | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\theta}) &= \prod_{p=1}^{P} k_p((\mathbf{c}, v_p), (\mathbf{c}', v'_p) | \beta_p, \boldsymbol{\theta}) \\ &= \prod_{p=1}^{P} \sum_{i=1}^{|\mathcal{V}_p|} [U^p]_{v_p,i} f(\lambda_i^p, \alpha_p \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta_p) [U^p]_{v'_p,i} \quad (5.5) \end{aligned}$$

where $\mathbf{v} = (v_1, \cdots, v_P)$, $\mathbf{v}' = (v'_1, \cdots, v'_P)$, $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_P)$ $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_P)$ and the graph Laplacian is given as $L(\mathcal{G}_p)$ with the eigendecomposition $U_p \mathrm{diag}[\lambda^p_1, \cdots, \lambda^p_{\| \mathcal{V}_p \|}] U_p^T$.

Eq.5.5 should not be confused with the kernel product of kernels on each $\mathcal{V}_p$. Note that the distance $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$ is shared, which introduces the coupling among discrete variables and thus allows more modeling freedom than a product kernel. In addition to the coupling, the kernel parameter $\alpha_p$s lets us individually determine the strength of the frequency modulation.

EXAMPLES Defining a FM kernel amounts to constructing a frequency modulating function. We introduce examples of flexible families of frequency modulating functions.

**Proposition 5.4.** *For $S \in (0, \infty)$, a finite measure $\mu$ on $[0, S]$, $\mu$-measurable $\tau : [0, S] \Rightarrow [0, 2]$ and $\mu$-measurable $\rho : [0, S] \Rightarrow \mathbb{N}$, the function of the form below is a frequency modulating function.*

$$f(\lambda, \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \beta) = \int_0^S \frac{1}{(1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^{\tau(s)})^{\rho(s)}} \mu(ds) \tag{5.6}$$

*Proof.* See Appx. Prop. C.8. □

Assuming $S = 1$ and $\tau(s) = 2$, Prop. 5.4 gives $(1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2)^{-1}$ with $\rho(s) = 1$ and $\mu(ds) = ds$, and $\sum_{n=1}^N a_n (1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2)^{-n}$ with $\rho(s) = \lfloor Ns \rfloor$ and $\mu(\{n/N\}) = a_n \geqslant 0$ and $\mu([\{n/N\}^c_{n=1,\cdots,N}) = 0$.

### 5.3.5 Extension of the Frequency Modulation

Frequency modulation is not restricted to distances on Euclidean spaces but it is applicable to any arbitrary space with a kernel defined on it. As a concrete example of frequency modulation by kernels, we show a non-stationary extension where $f$ does not depend on $\| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}$ but on the neural network kernel $k_{NN}$ (Rasmussen, 2003). Consider Eq. (5.2) with $f = f_{NN}$ as follows.

$$f_{NN}(\lambda, k_{NN}(\mathbf{c}, \mathbf{c}' | \Sigma) | \beta) = \frac{1}{2 + \beta\lambda - k_{NN}(\mathbf{c}, \mathbf{c}' | \Sigma)} \tag{5.7}$$

where $k_{NN}(\mathbf{c}, \mathbf{c}' | \Sigma) = \frac{2}{\pi} \arcsin\left( \frac{2\, \mathbf{c}^T \Sigma\, \mathbf{c}'}{(1 + \mathbf{c}^T \Sigma\, \mathbf{c})(1 + \mathbf{c}'^T \Sigma\, \mathbf{c}')} \right)$ is the neural network kernel (Rasmussen, 2003).

Since the range of $k_{NN}$ is $[-1, 1]$, $f_{NN}$ is positive and thus satisfies **FM-P1**. Through Eq. (5.7), Eq. (5.2) is positive definite (Sec. C.1, Prop.C.9) and thus property **FM-P2** is satisfied. If the premise $t_1 < t_2$ of the property **FM-P3** is replaced by $t_1 > t_2$, then **FM-P3** is also satisfied. In contrast to the frequency modulation principle with distances in Eq. (5.3), the frequency modulation principle with a kernel is formalized as

$$k_{NN}(\mathbf{c}, \mathbf{c}' | \Sigma) \geqslant k_{NN}(\tilde{\mathbf{c}}, \tilde{\mathbf{c}}' | \Sigma) \Rightarrow k((\mathbf{c}, v), (\mathbf{c}', v')) \geqslant k((\tilde{\mathbf{c}}, v), (\tilde{\mathbf{c}}', v'))$$

Note that $k_{NN}(\mathbf{c}, \mathbf{c}'|\Sigma)$ is a similarity measure and thus the inequality is not reversed unlike Eq. (5.3).

All above arguments on the extension of the frequency modulation using a nonstationary kernel hold also when the $k_{NN}$ is replaced by an arbitrary positive definite kernel. The only required condition is that a kernel has to be upper bounded, *i.e.*, $k_{NN}(\mathbf{c}, \mathbf{c}') \leqslant C$, needed for **FM-P1** and **FM-P2**.

## 5.4 RELATED WORKS

On continuous variables, many sophisticated kernels have been proposed (ChangYong Oh et al., 2018; Remes et al., 2017; Samo and S. Roberts, 2015; Andrew Wilson and Nickisch, 2015). In contrast, kernels on discrete variables have been studied less (Haussler, 1999; R. I. Kondor and Lafferty, 2002; Smola and R. Kondor, 2003). To our best knowledge, most of existing kernels on mixed variables are constructed by a kernel product (S. Li et al., 2016; K. Swersky, Snoek, et al., 2013) with some exceptions (Fiducioso et al., 2019; Krause and Ong, 2011; K. Swersky, Snoek, et al., 2013), which rely on kernel addition.

In mixed variable BO, non-GP surrogate models are more prevalent, including SMAC (Hutter et al., 2011) using random forest and TPE (J. S. Bergstra et al., 2011) using a tree structured density estimator. Recently, by extending the approach of using Bayesian linear regression for discrete variables (Baptista and Poloczek, 2018), (Daxberger et al., 2021) proposes Bayesian linear regression with manually chosen basis functions on mixed variables, providing a regret analysis using Thompson sampling as an acquisition function. Another family of approaches utilizes a bandit framework to handle the acquisition function optimization on mixed variables with theoretical analysis (Gopakumar et al., 2018; D. Nguyen et al., 2019; Ru, Alvi, et al., 2020). (D. Nguyen et al., 2019) use GP in combination with multi-armed bandit to model category-specific continuous variables and provide regret analysis using GP-UCB. Among these approaches, (Ru, Alvi, et al., 2020) also utilize information across different categorical values, which –in combination with the bandit framework– makes itself the most competitive method in the family.

Our focus is to extend the modelling prowess and flexibility of pure GPs for surrogate models on problems with mixed variables. We propose frequency modulated kernels, which are kernels that are specifically designed to model the complex interactions between continuous and discrete variables.

In architecture search, approaches using weight sharing such as DARTS (H. Liu et al., 2018) and ENAS (Pham et al., 2018) are gaining popularity. In spite of their efficiency, methods training neural networks from scratch for given architectures outperform approaches based on weight sharing (X. Dong, L. Liu, et al., 2021). Moreover, the joint optimization of learning hyperparameters and architectures is under-explored with a few exceptions such as BOHB (Falkner et al., 2018) and autoHAS (X. Dong, Tan, et al., 2020). Our approach proposes a competitive option to this challenging optimization of mixed variable functions with expensive evaluation cost.

## 5.5 EXPERIMENTS

To demonstrate the improved sample efficiency of GP BO using FM kernels (BO-FM) we study various mixed variable black-box function optimization tasks, including 3 synthetic problems from (Ru, Alvi, et al., 2020), 2 hyperparameter optimization problems (SVM (Smola and R. Kondor, 2003) and XGBoost (T. Chen and Guestrin, 2016)) and the joint optimization of neural architecture and SGD hyperparameters.

As per our method, we consider ModLap which is of the form Eq. 5.5 with the following frequency modulating function.

$$f_{\mathtt{Lap}}(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \frac{1}{1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2} \tag{5.8}$$

Moreover, to empirically demonstrate the importance of the similarity measure behavior, we consider another kernel following the form of Eq. 5.5 but disrespecting the frequency modulation principle with the function

$$f_{\mathtt{Dif}}(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \exp\left(-(1 + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^2) \beta\lambda\right) \tag{5.9}$$

We call the kernel constructed with this function ModDif.

In each round, after updating with an evaluation, we fit a GP surrogate model using marginal likelihood maximization with 10 random initialization until convergence (Williams and Rasmussen, 2006). We use the expected improvement (EI) acquisition function (Donald, 1998) and optimize it by repeated alternation of L-BFGS-B (Zhu et al., 1997) and hill climbing (Skiena, 1998) until convergence. More details on the experiments are provided in Appx. C.2.

BASELINES For synthetic problems and hyperparameter optimization problems below, baselines we consider[17] are SMAC[18] (Hutter et al., 2011), TPE[19] (J. S. Bergstra et al., 2011), and CoCaBO[20] (Ru, Alvi, et al., 2020) which consistently outperforms One-hot BO (GPyOpt, 2016)

17 The methods (Daxberger et al., 2021; D. Nguyen et al., 2019) whose code has not been released are excluded.
18 `https://github.com/automl/SMAC3`
19 `http://hyperopt.github.io/hyperopt/`
20 `https://github.com/rubinxin/CoCaBO_code`

and EXP3BO (Gopakumar et al., 2018). For CoCaBO, we consider 3 variants using different mixture weights.[21]

### 5.5.1 Synthetic problems

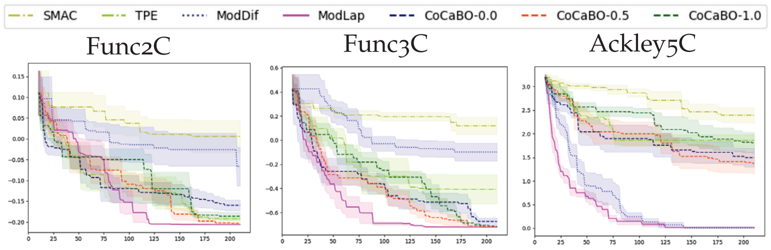


**Figure 5.2:** Synthetic functions on mixed-variable spaces (5 runs)

**Table 5.1:** Synthetic functions on mixed-variable spaces (5 runs)

| | Func2C | Func3C | Ackley5C |
|---|---|---|---|
| SMAC | $+0.006 \pm 0.039$ | $+0.119 \pm 0.072$ | $+2.381 \pm 0.165$ |
| TPE | $-0.192 \pm 0.005$ | $-0.407 \pm 0.120$ | $+1.860 \pm 0.125$ |
| ModDif | $-0.066 \pm 0.046$ | $-0.098 \pm 0.074$ | $+\mathbf{0.001} \pm \mathbf{0.000}$ |
| ModLap | $-\mathbf{0.206} \pm \mathbf{0.000}$ | $-\mathbf{0.722} \pm \mathbf{0.000}$ | $+\mathit{0.019} \pm \mathit{0.006}$ |
| CoCaBO-0.0 | $-0.159 \pm 0.013$ | $-0.673 \pm 0.027$ | $+1.499 \pm 0.201$ |
| CoCaBO-0.5 | $-\mathit{0.202} \pm \mathit{0.002}$ | $-\mathit{0.720} \pm \mathit{0.002}$ | $+1.372 \pm 0.211$ |
| CoCaBO-1.0 | $-0.186 \pm 0.009$ | $-0.714 \pm 0.005$ | $+1.811 \pm 0.217$ |

We test on 3 synthetic problems proposed in (Ru, Alvi, et al., 2020)[22]. Each of the synthetic problems has the search space as in Tab. 5.2. Details of synthetic problems can be found in (Ru, Alvi, et al., 2020).

**Table 5.2:** Synthetic Problem Search Spaces

| | Conti. Space | Num. of Cats. |
|---|---|---|
| Func2C | $[-1, 1]^2$ | 3, 5 |
| Func3C | $[-1, 1]^2$ | 3, 5,4 |
| Ackley5C | $[-1, 1]$ | 17, 17, 17, 17, 17 |

On all 3 synthetic benchmarks, ModLap shows competitive performance (Fig. 5.2). On Func2C and Func3C, ModLap performs the best, while on Ackley5C ModLap is at the second place, marginally further from the first. Notably, even on Func2C and Func3C, where

21 Learning the mixture weight is not supported in the implementation, we did not include it. Moreover, as shown in (Ru, Alvi, et al., 2020), at least one of 3 variants usually performs better than learning the mixture weight.

22 In the implementation provided by the authors, only Func2C and Func3C are supported. We implemented Ackley5C.

ModDif underperforms significantly, ModLap exhibits its competitiveness, which empirically supports that the similarity measure behavior plays an important role in the surrogate modeling in Bayesian optimization.

### 5.5.2 Hyperparameter optimization problems

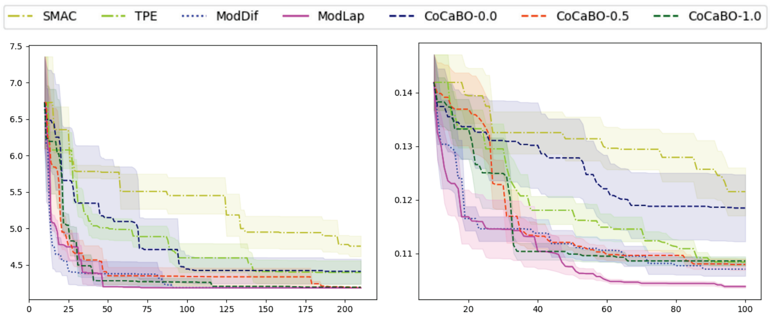


**Figure 5.3:** Hyperparameter optimization of SVM and XGBoost (5 runs)

**Table 5.3:** Hyperparameter optimization of SVM and XGBoost (5 runs)

| SVM | Method | XGBoost |
|---|---|---|
| $4.759 \pm .141$ | SMAC | $.1215 \pm .0045$ |
| $4.399 \pm .163$ | TPE | $.1084 \pm .0007$ |
| *4.188 ± .001* | ModDif | *.1071 ± .0013* |
| **4.186 ± .002** | ModLap | **.1038 ± .0003** |
| $4.412 \pm .170$ | CoCaBO-0.0 | $.1184 \pm .0062$ |
| $4.196 \pm .004$ | CoCaBO-0.5 | $.1079 \pm .0010$ |
| $4.196 \pm .004$ | CoCaBO-1.0 | $.1086 \pm .0008$ |

Now we consider a practical application of Bayesian optimization over mixed variables. We take two machine learning algorithms, SVM (Smola and R. Kondor, 2003) and XGBoost (T. Chen and Guestrin, 2016) and optimize their hyperparameters.

SVM We optimize hyperparameters of $\nu$-SVR in scikit-learn (Pedregosa et al., 2011). We consider 3 categorical hyperparameters and 3 continuous hyperparameters (Tab. 5.4) and for continuous hyperparameters we search over $\log_{10}$ transformed space of the range.
For each of 5 split of Boston housing dataset with train:test(7:3) ratio, $\nu$-SVR is fitted on the train set and RMSE on the test set is computed. The average of 5 test RMSE is the objective.

23 https://scikit-learn.org/stable/modules/generated/sklearn.svm.$\nu$-SVR .html

**Table 5.4:** $\nu$-SVR hyperparameters

| $\nu$-SVR param.[23] | Range |
|---|---|
| kernel | {linear, poly, RBF, sigmoid} |
| gamma | {scale, auto } |
| shrinking | {on, off } |
| C | $[10^{-4}, 10]$ |
| tol | $[10^{-6}, 1]$ |
| nu | $[10^{-6}, 1]$ |

XGBOOST We consider 1 ordinal, 3 categorical and 4 continuous hyperparameters (Tab. 5.5).

**Table 5.5:** XGBoost hyperparameters

| XGBoost param.[24] | Range |
|---|---|
| max_depth | $\{1, \cdots, 10\}$ |
| booster | {gbtree, dart} |
| grow_policy | {depthwise, lossguide} |
| objective | {multi:softmax, multi:softprob} |
| eta | $[10^{-6}, 1]$ |
| gamma | $[10^{-4}, 10]$ |
| subsample | $[10^{-3}, 1]$ |
| lambda | $[0, 5]$ |

For 3 continuous hyperparameters, eta, gamma and subsample, we search over the $\log_{10}$ transformed space of the range. With a stratified train:test(7:3) split, the model is trained with 50 rounds and the best test error over 50 rounds is the objective of SVM hyperparameter optimization.

In Fig. 5.3 and Tab. 5.3, ModLap performs the best. On XGBoost hyperparameter optimization, ModLap exhibits clear benefit compared to the baselines. Here, ModDif wins the second place in both problems.

COMPARISON TO DIFFERENT KERNEL COMBINATIONS In Appx. C.3, we also report the comparison with different kernel combinations on all 3 synthetic problems and 2 hyperparameter parameter optimization problems. We make two observations. First, ModDif, which does not respect the similarity measure behavior, sometimes severely degrades BO performance. Second, ModLap obtains equally good final results and consistently finds the better solutions faster than the kernel product. This can be clearly shown by comparing the area above the mean curve of BO runs using different kernels. The area above the mean curve of BO using ModLap is larger than the are above the mean curve of BO using the kernel product. Moreover, the gap between the area from ModLap and the area from kernel product

24 `https://xgboost.readthedocs.io/en/latest/parameter.html`

increases in problems with larger search spaces. Even on the smallest search space, Func2C, ModLap lags behind the kernel product up to around 90th evaluation and outperforms after it. The benefit of ModLap modeling complex dependency among mixed variables is more prominent in higher dimension problems.

### 5.5.3 Joint optimization of neural architecture and SGD hyperparameters

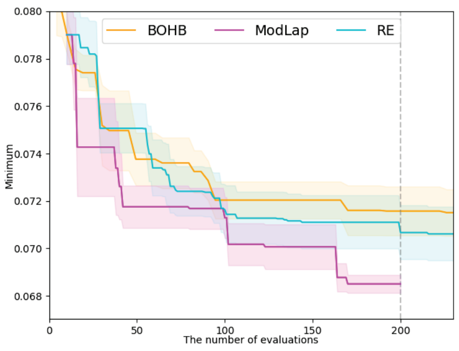


**Figure 5.4:** Joint optimization of the architecture and SGD hyperparameters (4 runs)

**Table 5.6:** Joint optimization of the architecture and SGD hyperparameters

| Method | Num.Evaluations | Mean ± Std.Err. (4 runs) |
|---|---|---|
| BOHB | 200 | $7.158 \times 10^{-2} \pm 1.0303 \times 10^{-3}$ |
| BOHB | 230 | $7.151 \times 10^{-2} \pm 9.8367 \times 10^{-4}$ |
| BOHB | 350 | $7.061 \times 10^{-2} \pm 5.9322 \times 10^{-4}$ |
| RE | 200 | $7.067 \times 10^{-2} \pm 1.1417 \times 10^{-3}$ |
| RE | 230 | $7.061 \times 10^{-2} \pm 1.1329 \times 10^{-3}$ |
| RE | 400 | $6.929 \times 10^{-2} \pm 6.4804 \times 10^{-4}$ |
| RE | 600 | $6.879 \times 10^{-2} \pm 1.0039 \times 10^{-3}$ |
| ModLap | 200 | $6.850 \times 10^{-2} \pm 3.7914 \times 10^{-4}$ |

* For the figure with all numbers above, see Appx. C.3.

Next, we experiment with BO on mixed variables by optimizing continuous and discrete hyperparameters of neural networks. The space of discrete hyperparameters $\mathcal{A}$ is modified from the NASNet search space (Zoph and Q. Le, 2017), which consists of 8,153,726,976 choices. The space of continuous hyperparameters $\mathcal{H}$ comprises 6 continuous hyperparameters of the SGD with a learning rate scheduler: learning rate, momentum, weight decay, learning rate reduction factor, 1st reduction point ratio and 2nd reduction point ratio.

A good neural architecture should both achieve low errors and be computationally modest. Thus, we optimize the objective $f(a, h) = err_{valid}(a, h) + 0.02 \times FLOP(a)/\max_{a' \in \mathcal{A}} FLOP(a')$. To increase the separability among smaller values, we use $\log f(a, h)$ transformed values whenever model fitting is performed on evaluation data. The reported results are still the original non-transformed $f(a, h)$.

We compare with two strong baselines. One is BOHB (Falkner et al., 2018) which is an evaluation-cost-aware algorithm augmenting unstructured bandit approach (L. Li et al., 2017) with model-based guidance. Another is RE (Real et al., 2019) based on a genetic algorithm with a novel population selection strategy. In (X. Dong, L. Liu, et al., 2021), on discrete-only spaces, these two outperform competitors including weight sharing approaches such as DARTS (H. Liu et al., 2018), SETN (X. Dong and Yi Yang, 2019), ENAS (Pham et al., 2018) and etc. In the experiment, for BOHB, we use the public implementation[25] and for RE, we use our own implementation.

For a given set of hyperparameters, with ModLap or RE, the neural network is trained on FashionMNIST for 25 epochs while BOHB adaptively chooses the number of epochs. For further details on the setup and the baselines we refer the reader to Appx. C.2 and C.3.

We present the results in Fig. 5.4. Since BOHB adaptively chooses the budget (the number of epochs), BOHB is plotted according to the budget consumption. For example, the y-axis value of BOHB on 100-th evaluation is the result of BOHB having consumed 2,500 epochs (25 epochs $\times$ 100).

We observe that ModLap finds the best architecture in terms of accuracy and computational cost. What is more, we observe that ModLap reaches the better solutions faster in terms of numbers of evaluations. Even though the time to evaluate a new hyperparameter is dominant, the time to suggest a new hyperparameter in ModLap is not negligible in this case. Therefore, we also provide the comparison with respect the wall-clock time. It is estimated that RE and BOHB evaluate 230 hyperparameters while ModLap evaluate 200 hyperparameters (Appx. C.2). For the same estimated wall-clock time, ModLap(200) outperforms competitors(RE(230), BOHB(230)).

In order to see how beneficial the sample efficiency of BO-FM is in comparison to the baselines, we perform a stress test in which more evaluations are allowed for RE and BOHB. We leave RE and BOHB for 600 evaluations and 350 evaluations, respectively. Notably, the RE with 600 evaluations almost reaches ModLap with 200 evaluations (Fig. 5.4 and Appx. C.3). We conclude that ModLap exhibits higher sample efficiency than the baselines.

25 `https://github.com/automl/HpBandSter`

## 5.6 CONCLUSION

We propose FM kernels to improve the sample efficiency of mixed variable Bayesian optimization.

On the theoretical side, we provide and prove conditions for FM kernels to be positive definite and to satisfy the similarity measure behavior. Both conditions are not trivial due to the interactions between quantities on two disparate domains, the spatial domain and the frequency domain.

On the empirical side, we validate the effect of the conditions for FM kernels on multiple synthetic problems and realistic hyperparameter optimization problems. Further, we successfully demonstrate the benefits of FM kernels compared to non-GP based Bayesian Optimization on a challenging joint optimization of neural architectures and SGD hyperparameters. BO-FM outperforms its competitors, including Regularized evolution, which requires three times as many evaluations.

We conclude that an effective modeling of dependencies between different types of variables improves the sample efficiency of BO. We believe the generality of the approach can have a wider impact on modeling dependencies between discrete variables and variables of arbitrary other types, including continuous variables.

# 6 BATCH BAYESIAN OPTIMIZATION ON PERMUTATIONS USING THE ACQUISITION WEIGHTED KERNELS

## 6.1 INTRODUCTION

From the celebrated traveling salesman problem(Gutin and Punnen, 2006) to flowshop and jobshop scheduling problems(Garey et al., 1976), permutations are ubiquitous representations in combinatorial optimization. Such combinatorial problems on permutations arise in highly impactful application areas. For example, in chip design, permutations specify relative placements of memories and logical gates on a chip(Alpert et al., 2008). As another example, in 3D printing, scheduling is an important factor to determine the production time(Chergui et al., 2018; V. Griffiths et al., 2019; Xu Song et al., 2020). In both cases, as well as in many others, evaluating the cost associated to a given permutation is expensive.

In situations where the evaluation is expensive, Bayesian optimization (BO) has shown good performance in many problems(Snoek, Larochelle, et al., 2012; Snoek, Rippel, et al., 2015). Recently, BO on combinatorial spaces has made significant progress for categorical variables (Baptista and Poloczek, 2018; Dadkhahi et al., 2020; Deshwal, Belakaria, J. R. Doppa, and Fern, 2020; Changyong Oh, Tomczak, et al., 2019; K. Swersky, Rubanova, et al., 2020). However, BO on permutations is yet under-explored with a few exceptions (Bachoc et al., 2020; Zaefferer et al., 2014; J. Zhang et al., 2019).

In this work we present a framework to deal with BO on permutations where the evaluation of the objective is expensive. We extend batch Bayesian optimization, which allows one to speed up the optimization by acquiring a batch of multiple points and evaluating the batch in parallel(Azimi, Fern, et al., 2010; González et al., 2016), to the case of permutations. Then, motivated by the observation that both the diversity of the points in the batches and the informativeness of the individual points in the batch improve the performance(Gong et al., 2019), we propose a new batch acquisition method which is applicable to the search space of permutations and takes into account both the diversity of the batch and the informativeness of each point. This method is based on determinantal point processes (DPPs), which have been widely used to model sets of diverse points(Kulesza and Taskar, 2012), and can be conveniently incorporated into the Gaussian Process framework since DPPs are specified by a kernel. To overcome the lack of informativeness of DDPs(Kathuria et al., 2016) (more specifically, the selection of points in batches relies solely on the predictive variance of the surrogate model), we enhance DPPs by using a kernel weighted by acquisition values. Therefore, we propose a new batch acquisition method using the so-called DPP **L**-ensemble(Borodin and Rains, 2005; Kulesza and Taskar, 2012) augmented with the **A**cquisition **W**eight, dubbed LAW. The whole procedure to find the optimal ordering (permutation) through LAW is thus dubbed LAW2ORDER. We compare LAW2ORDER and other competitors, firstly, on three combinatorial optimization benchmarks on permutations such as quadratic assignment problem, the flowshop scheduling problem, and the traveling salesman problem. We also make comparisons on the structure learning problem. In the structure learing problem, LAW2ORDER performs the best and the performance gap is more significant for larger permutation spaces. Moreover, LAW2ORDER still outperforms significantly genetic algorithms which use twice as many evaluations.

## 6.2 PRELIMINARIES

In this section, we briefly discuss some prerequisites for our proposed method and introduce notation. Below we will denote a function $f$ with one input as $f(\cdot)$, and function $K$ with two inputs as $K(\cdot,\cdot)$. For $B \in \mathbb{N}$, $[B] = \{1, \cdots, B\}$ while for a set $\mathcal{X}$, $|\mathcal{X}|$ is the number of elements in $\mathcal{X}$.

### 6.2.1 Batch Bayesian Optimization

Bayesian Optimization (BO) aims at finding the global optimum of a black-box function $f$ over a search space $\mathcal{X}$, namely, $\mathbf{x}_{opt} = \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. Two main components are the probabilistic modeling of the objective $f(\mathbf{x})$ and the acquisition of new points to evaluate. Probabilistic modeling is performed by the surrogate model. At the t-th round, the surrogate model attempts to approximate $f(\mathbf{x})$ based on the evaluation data $\mathcal{D}_{t-1}$, producing the predictive mean $\mu_{t-1}(\mathbf{x}) = \mu(\mathbf{x}|\mathcal{D}_{t-1})$ and the predictive variance $\sigma^2_{t-1}(\mathbf{x}) = \sigma^2(\mathbf{x}|\mathcal{D}_{t-1})$. In the acquisition of a new point, the acquisition function is specified $a_t(\mathbf{x}) = a_{seq}(\mathbf{x}|\mu_{t-1}(\cdot), \sigma^2_{t-1}(\cdot))$, which is based on the predictive mean $\mu_{t-1}(\cdot)$ and the predictive variance $\sigma^2_{t-1}(\cdot)$ to score how informative points are for the optimization. Next, the point that maximizes the acquisition function is obtained, $\mathbf{x}_t = \operatorname{argmax}_{\mathbf{x}} a_t(\mathbf{x})$, and the objective evaluated, $y_t = f(\mathbf{x}_t)$. Then, the new evaluation point is appended to the old dataset, $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$ and the process repeats by fitting the surrogate model with $\mathcal{D}_t$. The process continues until the evaluation budget is depleted. To contrast with the proposed method, we call this basic BO as sequential BO. For a more extensive overview of Bayesian optimization, please refer to (Frazier, 2018; Shahriari et al., 2015).

With more computational resources, such as more GPUs and CPUs, we can speed up Bayesian optimization by allowing multiple evaluations in parallel. For this, we acquire a batch of multiple points, a method known as Batch Bayesian Optimization (BBO) (Azimi, Fern, et al., 2010; González et al., 2016). In BBO, we need an acquisition function $a_{batch}$ scoring the quality of batches of B points $\{\mathbf{x}_b\}_{b\in[B]}$ instead of individual points. At time t, a batch of B points is acquired, $\{\mathbf{x}_{t,b}\}_{b\in[B]} = \text{argmax}_{\{\mathbf{x}_b\}_{b\in[B]}} a_t(\{\mathbf{x}_b\}_{b\in[B]})$, where $a_t(\{\mathbf{x}_b\}_{b\in[B]})$ is given as $a_{batch}(\{\mathbf{x}_b\}_{b\in[B]}|\mu_{t-1}(\cdot), \sigma^2_{t-1}(\cdot))$. Then the points in the acquired batch are evaluated in parallel and the evaluation data is updated by $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_{t,b}, y_{t,b})\}_{b\in[B]}$.

### 6.2.2 Determinantal Point Processes

Determinantal point processes (DPPs) are stochastic point processes well-suited to model sets of diverse points (Kulesza and Taskar, 2012). Let us assume that we want to sample a set of diverse points from a finite set $\mathcal{X}$. One way to define DPP is to use the so-called L-ensemble (Kulesza and Taskar, 2012). For a given kernel $L(\cdot,\cdot)$ on $\mathcal{X}$, the L-ensemble is defined as the random point process with density $P^{DPP}_L(X) = \frac{\det([L(\mathbf{x},\mathbf{y})]_{\mathbf{x},\mathbf{y}\in X})}{\det(I+L)}$ where $X \subset \mathcal{X}$ and $[L(\mathbf{x},\mathbf{y})]_{\mathbf{x},\mathbf{y}\in X}$ is a submatrix of L restricted to X (Borodin and Rains, 2005).

For a batch of just two points, $X = \{\mathbf{x},\mathbf{y}\}$ it is easy to observe that DPP encourages diversity — $P^{DPP}_L(\{\mathbf{x},\mathbf{y}\}) \propto L(\mathbf{x},\mathbf{x})L(\mathbf{y},\mathbf{y}) - L(\mathbf{x},\mathbf{y})^2$. Indeed, for more similar points the value of $L(\mathbf{x},\mathbf{y})$ is higher, resulting in a lower density. In DPPs there is no cardinality constraint on X. We, therefore, define k-DPP, which is a DPP with the restriction that sampled sets have precisely k points. Denoting the set of subsets of $\mathcal{X}$ with k points by $\mathcal{X}_k$, the k-DPP density is defined for $X \in \mathcal{X}_k$ by $P^{k\text{-}DPP}_L(X) = \frac{\det([L(\mathbf{x},\mathbf{y})]_{\mathbf{x},\mathbf{y}\in X})}{\sum_{X'\in\mathcal{X}_k}\det([L(\mathbf{x},\mathbf{y})]_{\mathbf{x},\mathbf{y}\in X'})}$. Therefore, $X^* = \text{argmax}_{X\in\mathcal{X}_k} P^{k\text{-}DPP}_L(X)$ is the most diverse set of k points with respect to the similarity encoded by the kernel $L(\cdot,\cdot)$.

In our algorithm, we use that log of k-DPP density is submodular (Kulesza and Taskar, 2012; Srinivas et al., 2010) and can be greedily maximized with approximation guarantees (Nemhauser et al., 1978; Sakaue, 2020) (See Appx. D.3.1 for a brief discussion).

## 6.3 METHODS

Batch acquisition on a combinatorial space poses two difficulties. First, the batch acquisition objectives of existing batch Bayesian optimization are designed based on the properties and intuition applicable to continuous spaces (Gong et al., 2019; González et al., 2016). This may not always be suitable for discrete spaces. For instance, the method in (González et al., 2016) is defined by using Euclidean distance. Also, the difficulties of combinatorial optimization are exacerbated when optimizing a batch jointly. This is in stark contrast to the continuous case where gradient based optimization is easily extended to batch optimization of multiple points in parallel (Jialei Wang et al., 2020).

To cope with these challenges, we introduce a new batch acquisition method for Bayesian optimization, the maximization of the determinantal point process (DPP) density defined by an **L**-ensemble with **A**cquisition **W**eights, dubbed **LAW**. We describe LAW in Subsec. 6.3.1 and its regret analysis is provided in Subsec. 6.3.2 and 6.3.3.

### 6.3.1 Batch Acquisition using LAW

We start to define the main components of LAW.

**Definition 6.1** (Weight function)**.** We call a function $w : \mathbb{R} \to \mathbb{R}$ a weight function if it is positive ($r \in \mathbb{R}, w(r) > 0$), increasing ($r_1 \leqslant r_2 \implies w(r_1) \leqslant w(r_2)$), and bounded below and above by a positive number ($w_- = \inf_{r \in \mathbb{R}} w(r) > 0$ and $w_+ = \sup_{r \in \mathbb{R}} w(r) < \infty$).

**Definition 6.2** (Posterior covariance function)**.** Given a (prior) kernel $K(\mathbf{x}_1, \mathbf{x}_2)$, data $\mathcal{D}$ and noise variance $\sigma^2$, the posterior (predictive) covariance function $K_{post}(\mathbf{x}_1, \mathbf{x}_2 \mid \mathcal{D}, \sigma^2)$ is defined as $K(\mathbf{x}_1, \mathbf{x}_2) - K(\mathbf{x}_1, \mathcal{D})\ (K(\mathcal{D}, \mathcal{D}) + \sigma^2 I)^{-1} K(\mathcal{D}, \mathbf{x}_2)$.

Let us assume that we are running batch Bayesian optimization with Gaussian process surrogate model using the kernel $K(\cdot, \cdot)$ and the acquisition function $a(\cdot)$, and that we acquire a batch of $B$ points in each round. At the t-th round, we have the evaluation data $\mathcal{D}_{t-1}$, the posterior covariance function $K_t(\cdot, \cdot) = K_{post}(\cdot, \cdot | K, \mathcal{D}_{t-1}, \sigma^2_{obs})$ and the acquisition function $a_t(\{\mathbf{x}_b\}_{b \in [B]}) = a(\{\mathbf{x}_b\}_{b \in [B]} | \mu_{t-1}(\cdot), \sigma^2_{t-1}(\cdot))$ where $\mu_{t-1}(\cdot)$ and $\sigma^2_{t-1}(\cdot)$ are the predictive mean and the predictive variance conditioned on $\mathcal{D}_{t-1}$.

In the existing work (Kathuria et al., 2016) on batch Bayesian optimization using DPP, the posterior covariance function is used as the kernel defining DPP. Even though the use of DPP in (Kathuria et al., 2016) encourages diversity among points in batches, it essentially chooses points of high predictive variance.

However, the predictive mean also provides valuable information in Bayesian optimization. It is the acquisition function which harmo-

**Algorithm 4** Batch Acquisition by LAW

```
1: Input: weight function $w(\cdot)$, diversity gauge $L(\cdot,\cdot)$, acquisition
   function $a(\cdot)$, batch size $B$
2: Output: batch of $B$ points $\{\mathbf{x}_{t,1}, \cdots, \mathbf{x}_{t,B}\}$
3: $\mathbf{x}_{t,1} = \operatorname{argmax}_{\mathbf{x}\in\mathcal{X}} a(\mathbf{x}) = \operatorname{argmax}_{\mathbf{x}\in\mathcal{X}} w(a(\mathbf{x}))$
4: for $b = 2, \cdots, B$ do
5:   $\mathbf{x}_{t,b} = \operatorname{argmax}_{\mathbf{x}} \log(L(\mathbf{x}, \mathbf{x}|\{\mathbf{x}_{t,i}\}_{i\in[b-1]}))w(a(\mathbf{x}))^2$
6: end for
```

nizes the predictive mean and the predictive variance to quantify how useful each point is. Therefore, we propose a new batch acquisition method which actively uses the acquisition function while retaining the strength of DPP encouraging diversity in each batch.

We define the *Acquisition Weighted* kernel $L^{AW}$ as follows

$$L^{AW}(\mathbf{x}_1, \mathbf{x}_2) = w(a(\mathbf{x}_1)) \cdot L(\mathbf{x}_1, \mathbf{x}_2) \cdot w(a(\mathbf{x}_2)).$$

Here $w$ is a positive weight function. We call the kernel $L$ in $L^{AW}$ the *diversity gauge* of $L^{AW}$.

With the posterior covariance function as the diversity gauge $L = K_t$ and the acquisition function $a = a_t$, the acquisition weighted kernel becomes

$$L_t^{AW}(\mathbf{x}_1, \mathbf{x}_2) = w(a_t(\mathbf{x}_1)) \cdot K_t(\mathbf{x}_1, \mathbf{x}_2) \cdot w(a_t(\mathbf{x}_2)) \tag{6.1}$$

Due to the dependency of $L = K_t$ and $a_t$ to the round index $t$, we subscript $L^{AW}$ and $L$ with $t$.

We optimize the density of DPP using the acquisition weighted kernel to choose the points in the batch. Before introducing how this optimization is actually performed, we provide an interpretation by rewriting the numerator of the density of the k-DPP defined by $L_t^{AW}$ as

$$\det([L_t^{AW}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j\in[B]}) = \det([K_t(\mathbf{x}_i, \mathbf{x}_j)]_{i,j\in[B]}) \prod_{i=1}^{B} w(a_t(\mathbf{x}_i))^2 \tag{6.2}$$

This shows that the maximization of eq. (6.2) can be obtained by increasing both $\det([L_t(\mathbf{x}_i, \mathbf{x}_j)]_{i,j\in[B]})$ and $\prod_{i=1}^{B} w(a_t(\mathbf{x}_i))^2$ in a balanced way. Increasing the determinant term and increasing the product term promote diversity and acquisition values, respectively.

Now, we provide details on how the optimization is implemented. In combinatorial spaces where the optimization of a single point is challenging, the joint optimization of multiple points is daunting. Thanks to the submodularity of the log of the determinant (Kulesza and Taskar, 2012), the joint optimization of multiple points can be approximated by a sequence of single point optimizations with an approximation guarantee (See Appx. D.3.1 for submodularity).

The first point is obtained as in sequential Bayesian optimization by optimizing an acquisition function (line 3 in Alg. 4). The rest of the

$B-1$ points are obtained by maximizing the k-DPP density defined by the acquisition weighted kernel, which we approximately perform with a greedy method (line 4 in Alg. 4). Having chosen $b-1$ points $\{\mathbf{x}_{t,i}\}_{i\in[b-1]}$, the greedy maximization selects $\mathbf{x}_b$ as follows:

$$\begin{aligned}\mathbf{x}_b &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log\det([L_t^{AW}(\cdot,\cdot)]_{\{x_i\}_{i\in[b-1]}\cup\{x\}}) \\ &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log(L_t(\mathbf{x},\mathbf{x}|\{\mathbf{x}_{t,i}\}_{i\in[b-1]})\cdot w(a_t(\mathbf{x}))^2)\end{aligned}$$

where $L_t(\mathbf{x},\mathbf{x}|\{\mathbf{x}_{t,i}\}_{i\in[b-1]})$ is the posterior variance of the kernel $L_t$ conditioned on $\{\mathbf{x}_{t,i}\}_{i\in[b-1]}$.

### 6.3.2 Regret Analysis

In this subsection, we provide a theoretical analysis on the performance of LAW with two acquisition functions, GP-UBC (Srinivas et al., 2010) and EST (Zi Wang, B. Zhou, et al., 2016).

We begin with definitions needed in the analysis.

**Definition 6.3.** In the minimization of $f$ using batch acquisition, where $x^* = \operatorname{argmin}_{\mathbf{x}} f(\mathbf{x})$, $r_{t,b} = f(\mathbf{x}_{t,b}) - f(\mathbf{x}^*)$ is called instantaneous regret and $r_t^{(B)} = \min_{b\in[B]} r_{t,b} = \min_{b\in[B]}(f(\mathbf{x}_{t,b}) - f(\mathbf{x}^*))$ is called batch instantaneous regret. Simple regret is defined as the minimum of batch instantaneous regrets $R_T^{(B)}$.

$$S_T^{(B)} = \min_{t=1,\cdots,T} r_t^{(B)} = \min_{t=1,\cdots,T}\min_{b\in[B]} r_{t,b}$$

Batch cumulative regret $R_T^{(B)}$ is defined as the sum of batch instantaneous regrets

$$R_T^{(B)} = \sum_{t=1}^{T} r_t^{(B)} = \sum_{t=1}^{T}\min_{b\in[B]} r_{t,b}.$$

*Remark* 6.1. Note that $S_T^{(B)} \leqslant \frac{1}{T}R_T^{(B)}$. Vanishing simple regret is proved by showing $\frac{1}{T}R_T^{(B)} \to 0$.

**Definition 6.4.** For Gaussian processes with the kernel $K$ and the variance of observation noise $\sigma^2$, the maximum information gain $\gamma_T$ is defined as

$$\gamma_T = \gamma(T;\mathcal{X},K,\sigma^2) = \max_{X\subset\mathcal{X},|X|=T}\frac{1}{2}\log\det(I+\sigma^{-2}K(X,X)).$$

For UCB and EST, we have the following regret bound.

**Theorem 6.1.** *Assume a kernel such that* $K(\cdot,\cdot)\leqslant 1$, $|\mathcal{X}|<\infty$ *and* $f : \mathcal{X}\to\mathbb{R}$ *is sampled from* $\mathcal{GP}(\mathbf{0},K)$*. In each round* $t\in[T]$ *of batch Bayesian optimization, LAW acquires a batch using the evaluation data* $\mathcal{D}_{t-1}$*, the* *diversity measure* $L_t(\cdot,\cdot) = K(\cdot,\cdot|\mathcal{D}_{t-1})$*, an acquisition function* $a_t(\cdot)$ *and a weight function* $w(\cdot)$ *(Def. 6.1).*

*Let* $C_1 = \frac{36}{\log(1+\sigma^{-2})}$ *where* $\sigma^2$ *is the variance of the observation noise and* $\delta \in (0,1)$.

*At round* $t$*, define* $\beta_{t,1}^{(B)UCB} = 2\log\left(\frac{|\mathcal{X}|\pi^2((t-1)B+1)^2}{6\delta}\right)$ *and* $\nu_t = \min_{\mathbf{x}}\left(\frac{\mu_{t-1}(\mathbf{x})-\hat{m}_t}{\sigma_{t-1,1}(\mathbf{x})}\right)$ *where* $\hat{m}_t$ *is the estimate of the optimum (Zi Wang, B. Zhou, et al., 2016),* $\zeta_t = \left(2\log\left(\frac{\pi_t^2}{2\delta}\right)\right)^{1/2}$, $\pi_t > 0$ *such that* $\sum_{t=1}^{\infty}\pi_t^{-1} \leqslant 1$.

*Then batch cumulative regret satisfies the following bound*

$$P\left(\left\{\frac{R_T^{(B)}}{T} \leqslant \frac{\eta_T^{(B)}}{T} + \eta_T^{(B)}\frac{w_+}{w_-}\sqrt{C_1\frac{\gamma_{TB}}{TB}}\right\}\right) \geqslant 1-\delta \tag{6.3}$$

*where for EST,* $\eta_t^{(B)} = \nu_{t^*} + \zeta_t$ *and for UBC,* $\eta_t^{(B)} = 2(\beta_{t,1}^{(B)UCB})^{1/2}$*, and* $t^* = \operatorname*{argmax}_{s\in[t]} \nu_s$.

*Proof.* See. Appx. D.1. □

*Remark* 6.2. This theorem shows that, for the same kernel, the regret bound of LAW also enjoys the same asymptotic behavior as the regret bound of existing works (Contal et al., 2013; Desautels et al., 2014; Kathuria et al., 2016).

*Remark* 6.3. Note that Thm. 6.1 is about a bound on $\frac{1}{T}R_T^{(B)}$ while the analysis in (Desautels et al., 2014; Kandasamy, Krishnamurthy, et al., 2018) is to bound $\frac{1}{TB}R_{T,B}$ where $R_{T,B} = \sum_{t,b} r_{t,b}$. Since $\frac{1}{T}R_T^{(B)} \leqslant \frac{1}{TB}R_{T,B}$, bounding $\frac{1}{TB}R_{T,B}$ implies bounding $\frac{1}{T}R_T^{(B)}$. For the purpose of showing vanishing simple regret, both approaches are viable. Technically, two approaches require different treatments. See Appx. D.1.3 for the discussion on the differences between two approaches.

*Remark* 6.4. The ratio $\frac{w_+}{w_-}$ in Thm. 6.1 determines how LAW balances between the quality and the diversity. If the ratio is large, then the acquisition value is more influential in Eq. 6.2. Otherwise, Eq. 6.2 is dominated by the determinant of the diversity gauge, and the diversity of the batch is more emphasized. The bound in Eq. 6.3 reveals the necessity of the upper bound of $\frac{w_+}{w_-}$. Without the upper bound, i.e. virtually considering the acquisition value only, the batch acquisition may result in non-vanishing regret. However, the bound is not tight enough considering the extreme case $\frac{w_+}{w_-} = 1$. Nonetheless, the necessity of the upper bound of $\frac{w_+}{w_-}$ guides how the weight function $w(\cdot)$ is set (See for details). Moreover, the benefit of considering the acquisition weight is supported by the experimental results (Sec. 6.5)

Note that $\eta_T^{(B)} = \mathcal{O}(\sqrt{\log(TB)})$ (See Appx. D.1.4 for details). In Thm. 6.1, we need $\eta_T^{(B)} \cdot \sqrt{\frac{\gamma_{TB}}{TB}} \to 0$ to prove vanishing simple regret.

We provide a bound for the maximum information gain $\gamma_T$ of a kernel on a finite space, which we use later to show the vanishing simple regret.

**Theorem 6.2.** $K$ *is a kernel on a finite set* $\mathcal{X}$ *(*$|\mathcal{X}| < \infty$*),* $\sigma^2$ *is the variance of the observation noise and* $\Lambda = \{\lambda_n\}_{1,\cdots,|\mathcal{X}|}$ $(\lambda_n \geqslant \lambda_{n+1} \geqslant 0)$ *is the set of eigenvalues of the gram matrix* $K(\mathcal{X},\mathcal{X})$*. Then*

$$\gamma_T \leqslant \frac{1}{2}\min\{T \cdot \log\det(1+\sigma^{-2}\max_{x\in\mathcal{X}} K(x,x)), \\ |\mathcal{X}| \cdot \log(1+\sigma^{-2}\lambda_{max}T)\} \tag{6.4}$$

*where* $\lambda_{max}$ *is the largest eigenvalue of* $K(\mathcal{X},\mathcal{X})$*.*

*Proof.* See. Appx. D.2.1 □

### 6.3.3 Position Kernel

Based on the comparative experiments in (Zaefferer et al., 2014) showing that the Position kernel outperforms others consistently,[26], we use the position kernel in our BBO on permutations

$$K(\pi_1,\pi_2|\tau) = \exp\Big(-\tau \cdot \sum_i |\pi_1^{-1}(i) - \pi_2^{-1}(i)|\Big).$$

The positive definiteness of the position kernel was empirically tested via simulation (Zaefferer et al., 2014) and has not been shown rigorously. Therefore, we show the positive definiteness of the position kernel and bound its eigenvalues.

**Theorem 6.3.** *The position kernel* $K(\cdot,\cdot|\tau)$ *defined on* $S_N$ *is positive definite and the eigenvalues of* $K(X,X)$ *where* $X \subset \mathcal{X}$ *lie between* $\left(\frac{1-\rho}{1+\rho}\right)^N$ *and* $\left(\frac{1+\rho}{1-\rho}\right)^N$ *where* $\rho = \exp(-\tau)$*.*

*Proof.* See Appx. D.2.2 □

By utilizing the property of the position kernel, we provide a bound on the maximal information gain which is tighter than the one obtained in Thm. 6.2.

**Theorem 6.4.** $K(\cdot,\cdot|\tau)$ *is the position kernel defined on* $S_N$*,* $\sigma^2$ *is the variance of the observation noise,* $\rho = \exp(-\tau)$ *and,* $D_{max} = (N^2 - (N \textit{ mod } 2))/2$*.*

26 We also compared different kernels on regression tasks, including Kendall, Mallow (Jiao and Vert, 2015), Hamming, Manhattan, Position (Zaefferer et al., 2014) and Neural Kernel Network (NKN) (S. Sun, G. Zhang, C. Wang, et al., 2018) using mentioned kernels as building blocks. The position kernel and NKN performs similarly the best. NKN uses the position kernel as a building block kernel which is attributed to the position kernel in the performance of NKN.

*Then*

$$\gamma_T \leqslant \frac{1}{2}\min\{A(T), |\mathcal{X}| \cdot \log(1+\sigma^{-2}\lambda_{max}T)\}$$

*where* $\lambda_{max}$ *is the largest eigenvalue of* $K(\mathcal{X}, \mathcal{X})$ *and*

$$\begin{aligned} A(T) = &\log(1+\sigma^{-2}(1+(T-1)\rho^{D_{max}})) \\ &+ (T-1)\log(1+\sigma^{-2}(1-\rho^{D_{max}})) \end{aligned}$$

*which is smaller than* $T \cdot \log(1+\sigma^{-2}\max_{x\in\mathcal{X}} K(x,x))$.

*Proof.* See Appx. D.2.3 □

*Remark* 6.5. When $\rho \in (0,1)$ is close to one, i.e. $\log(1+\sigma^{-2}(1-\rho^{D_{max}})) \approx 0$, we can observe that even in the finite-time regime, the regret is almost sublinear since it is dominated by $\log(1+\sigma^{-2}(1+(T-1)\rho^{D_{max}}))$. In this case, the theorem provides a bound which is significantly tighter than the bound in Thm. 6.2 even in the finite-time regime.

*Remark* 6.6. If $\lambda_{max}$ is bounded, Thm. 6.2 can show the vanishing simple regret. For a kernel $K$ on a finite space $\mathcal{X}$, $\lambda_{max} \leqslant \mathtt{trace}(K(\mathcal{X}, \mathcal{X})) < \infty$. Therefore, $\gamma_T = \mathcal{O}(\mathtt{log}(T))$ for any kernel. However, considering the magnitude of $|\mathcal{X}|$ and $\lambda_{max}$ for large spaces, Eq. 6.4 is quite loose. $\lambda_{max}$ in Eq. 6.4 reflects kernel-dependent behavior of $\gamma_T$. Therefore, in Thm. 6.4 we bound $\lambda_{max}$ for a specific kernel and analyze further kernel-dependent non-asymptotic behavior.

The regret bounds of LAW are most informative in the asymptotic regime of large $T$. However, in Bayesian optimization where, typically, only a small number of evaluations can be afforded, the asymptotic bound may not be informative in terms of practical performance. In Sec. 6.5, we show that, in practice, LAW significantly outperforms other methods.

## 6.4 RELATED WORKS

Most existing batch Bayesian optimization methods using Gaussian process surrogate models focus on continuous search spaces. Many of them are not applicable to combinatoral spaces because the algorithms use specific properties of Euclidean spaces, e.g, Euclidean distance (Azimi, Fern, et al., 2010; González et al., 2016; Kathuria et al., 2016; Lyu et al., 2018; Zi Wang, B. Zhou, et al., 2016; Wu and Frazier, 2016), grid partitioning (Zi Wang, Gehring, et al., 2018; Zi Wang, C. Li, et al., 2017), projection using Euclidean geometry (Jialei Wang et al., 2020). The methods (Gong et al., 2019; Kandasamy, Krishnamurthy, et al., 2018; Shah and Ghahramani, 2015) using Thompson sampling (TS) (Thompson, 1933; J. Wilson et al., 2020), random feature (Rahimi, Recht, et al., 2007) or entropy search (Hennig and

Schuler, 2012; J. M. Hernández-Lobato, M. W. Hoffman, et al., 2014) require either closed-form expression of eigenfunctions or Choleksy decomposition of the gram matrix on all points in the search space. In general, a closed-form expression of eigenfunctions (RBF) is not available. For large combinatorial spaces, Choleksy decomposition is infeasible. LAW is a batch acquisition method applicable to general spaces including permutation spaces.

Recently, BO on combinatorial spaces has made significant progress for categorical variables (Baptista and Poloczek, 2018; Dadkhahi et al., 2020; Deshwal, Belakaria, J. R. Doppa, and Fern, 2020; Changyong Oh, Tomczak, et al., 2019; K. Swersky, Rubanova, et al., 2020). However, relatively few works in Bayesian optimization have focused on permutations (Bachoc et al., 2020; Zaefferer et al., 2014; J. Zhang et al., 2019). While existing works focus on the effect of the kernel on performance, our focus is to scale up Bayesian optimization on permutations via batch acquisition, which has not been studied in previous works.

The application of determinantal point processes (DPPs) to Bayesian optimization is not new. The use of DPP and the regret analysis on continuous search spaces (Kathuria et al., 2016) is closely related to our work. We focus on optimization problems on permutations rather than continuous spaces and use acquisition weighted kernels in our DPP. We provide a regret bound, which includes the unweighted case as a special case. Moreover, we show the behavior of the information gain of the position kernel, which, in turn, helps to understand the behavior of BO on permutations.

The idea of using weighted kernels was investigated in DPP (Kulesza and Taskar, 2010, 2012), also recently in the context of active learning (Bıyık et al., 2019) and more recently in architecture search (V. Nguyen, T. Le, et al., 2021). In addition to the use of the acquisition weights, we provide a regret analysis and the bound on the information gain of the position kernel for BO on permutations.

In existing works on regret analysis of batch Bayesian optimization, the cumulative regret is analyzed as an end goal (Desautels et al., 2014) and as a medium to show vanishing simple regret (Kandasamy, Krishnamurthy, et al., 2018). On the other hand, we analyze the batch cumulative regret not the cumulative regret (see Def. 6.3 and remarks thereafter). The batch cumulative regret is analyzed in (Contal et al., 2013) but without the acquisition weight. More detailed discussion on the difference among all these analyses is provided in Appx. D.1.3.

## 6.5 EXPERIMENTS

We empirically demonstrate the benefit of LAW on many optimization problems on permutations.[27]

In all Gaussian process (GP) based BO including baselines, we use the position kernel (see. Subsec. 6.3.3). At each round, evaluation outputs are normalized. GP surrogate models are trained with output normalized evaluation data by optimizing the marginal likelihood until convergence with 10 different random initializations. We use the Adam optimizer (Kingma and Ba, 2015) with default PyTorch (Paszke et al., 2017) settings except for the learning rate of 0.1.

When the optimization is performed on a single permutation variable, for example in greedy optimization, hill climbing is used until convergence and the neighbors are defined as the set of permutations obtainable by swapping two locations.

### 6.5.1 Weight function

The motivation of the acquisition weight is to promote the quality of the queries in the batch by using acquisition weights. In order to reflect this motivation, the weight function should be monotonically increasing. The better the quality (acquisition value) is, the larger the batch acquisition objective is.

In Eq. 6.2, the batch acquisition objective is factorized into the DPP with the diversity gauge and the product of weights, thus the weight function should be positive to prevent the product of an even number of large negative values becomes a large positive value. Also, in Eq. 6.2, zero weight value nullifies the diversity component, thus the weights function is required to be nonzero.

In Thm. 6.1, for vanishing regret, the ratio $\frac{w_+}{w_-}$ should be upper-bounded. Not only it facilitates the proof, but the upper bound is also intuitively appealing because we do not want to overly emphasize the quality of the queries. We want to balance quality and diversity. Weight functions with an unbounded ratio may erase the diversity consideration.

Combining the rationale behind LAW and its regret analysis, we set the weight function to be, monotonically increasing, positive valued, bounded below, bounded above.

### 6.5.2 Combinatorial Optimization Benchmarks

We consider LAW with two acquisition functions[28], EST (Zi Wang, B. Zhou, et al., 2016) and EI (Jones et al., 1998), LAW-EST and LAW-EI.

27 The code is available at `https://github.com/ChangYong-Oh/LAW2ORDER`

28 The $\beta_t$ in UCB balancing between exploitation and exploration increases as the size of the search space increases in the finite search space case (Srinivas et al., 2010). In

Even though the regret bound of LAW-EI is not provided in Thm. 6.1, we include LAW-EI because EI is the most popular acquisition function and this reveals the effect of the acquisition weights with different acquisition functions. We use the sigmoid $w(a) = 0.01 + 0.99(1 + \exp(-0.2 \cdot a))^{-1}$ for LAW-EST and $w(a) = 0.01 + a$ for LAW-EI.[29]

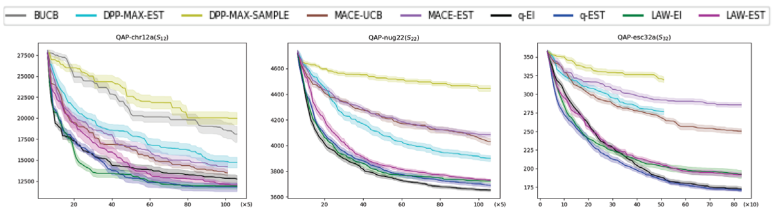


**Figure 6.1:** Quadratic Assignment Problems (15 runs)

**Table 6.1:** Quadratic Assignment Problems (15 runs)

| Benchmarks | QAP-chr12a | QAP-nug22 | QAP-esc32a |
|---|---|---|---|
| Batch | 5 | 5 | 10 |
| BUCB | $18105 \pm 955$ (8) | - | - |
| DPP-MAX-EST | $14732 \pm 634$ (7) | $3900 \pm 23$ (5) | $276.5 \pm 3.9$ (6) |
| DPP-SMP-EST | $19970 \pm 719$ (9) | $4446 \pm 22$ (8) | $319.6 \pm 3.8$ (8) |
| MACE-UCB | $13440 \pm 348$ (5) | $4031 \pm 26$ (6) | $250.3 \pm 3.5$ (5) |
| MACE-EST | $14126 \pm 596$ (6) | $4086 \pm 20$ (7) | $285.6 \pm 3.1$ (7) |
| q-EI | $12769 \pm 457$ (4) | $3653 \pm 10$ (1) | $172.7 \pm 3.2$ (2) |
| q-EST | $11790 \pm 498$ (1) | $3690 \pm 15$ (2) | $171.2 \pm 1.8$ (1) |
| LAW-EI | $11914 \pm 345$ (2) | $3724 \pm 13$ (3) | $192.5 \pm 5.3$ (4) |
| LAW-EST | $12067 \pm 238$ (3) | $3731 \pm 9$ (4) | $191.7 \pm 2.9$ (3) |

The baselines are BUCB (Desautels et al., 2014), DPP-MAX-EST, DPP-SAMPLE-EST (Kathuria et al., 2016)[30] MACE-UCB, MACE-EST (Lyu et al., 2018)[31] and q-EI, q-EST (Ginsbourger et al., 2008). Even though the original names of the baselines are used to emphasize their batch acquisition strategy, all baselines use the position kernel. Hence, the batch acquisition strategy is the only differentiating factor among baselines and LAW(ours). Note that DPP-MAX-EST (Kathuria

the experiments, due to the size of the search space, GP-UCB virtually becomes the predictive variance. Thus we exclude LAW-UCB.

29 LAW-EI is included to check the influence of different acquisition functions despite the lack of regret analysis. Therefore, the weight function is chosen to prevent zero values from numerical truncation.

30 PE (Contal et al., 2013) is equivalent to DPP-MAX-UCB (Kathuria et al., 2016). Since on continuous problems DPP-MAX-EST outperforms DPP-MAX-UCB (Kathuria et al., 2016), we exclude PE.

31 The MACE requires multi-objective optimization on permutations. We use NSGA-II (Deb et al., 2002) in Pymoo (Blank and Deb, 2020). MACE-UCB uses the original set of acquisition functions: PI, EI and UCB (Lyu et al., 2018), while MACE-EST replaces UCB with EST.

et al., 2016) corresponds to LAW-EST with $w(\cdot) \equiv$ const., i.e. no acquisition weight.

Note that, due to the reasons discussed in Sec. 6.4, existing works based on Thompson sampling or the properties of Euclidean space are excluded from the baselines.

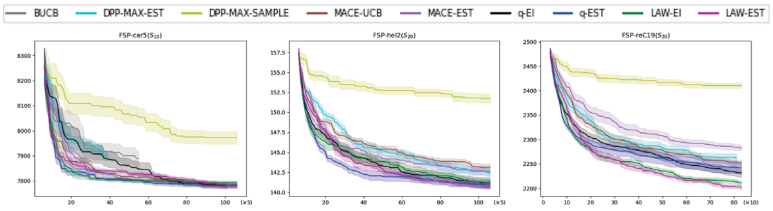


**Figure 6.2:** Permutations Benchmarks (15 runs)

**Table 6.2:** Flowshop Scheduling Problems (15 runs)

| Benchmarks<br>Batch | FSP-car5<br>5 | FSP-hel2<br>5 | FSP-reC19<br>10 |
|---|---|---|---|
| BUCB | $7887 \pm 32$ (8) | - | - |
| DPP-MAX-EST | $7796 \pm 11$ (7) | $142.5 \pm 0.48$ (5) | $2262 \pm 7.7$ (6) |
| DPP-SMP-EST | $7973 \pm 26$ (9) | $151.7 \pm 0.58$ (8) | $2410 \pm 6.1$ (8) |
| MACE-UCB | $7776 \pm 10$ (1) | $143.1 \pm 0.42$ (7) | $2252 \pm 5.8$ (5) |
| MACE-EST | $7791 \pm 9$ (5) | $142.5 \pm 0.45$ (5) | $2282 \pm 5.9$ (7) |
| q-EI | $7783 \pm 11$ (4) | $141.2 \pm 0.66$ (3) | $2231 \pm 8.4$ (3) |
| q-EST | $7782 \pm 9$ (3) | $141.0 \pm 0.49$ (2) | $2242 \pm 12.1$ (4) |
| LAW-EI | $7794 \pm 8$ (6) | $141.2 \pm 0.45$ (3) | $2211 \pm 4.5$ (2) |
| LAW-EST | $7780 \pm 7$ (2) | $140.7 \pm 0.31$ (1) | $2202 \pm 4.2$ (1) |

We consider three types of combinatorial optimization on permutations, Quadratic Assignment Problems (QAP), Flowshop Scheduling Problems (FSP) and Traveling Salesman Problems (TSP) (See Appx. D.3.3 for data source).

For each benchmark, all methods share 5 randomly generated initial evaluation data sets of 20 points and for each initial evaluation data set, each method is run three times — 15 runs in total.

DPP-MAX-EST uses the position kernel as LAW-EST, this is equivalent to LAW-EST without the acquisition weight, i.e. $w(a) = 1$. By comparing LAW-EST with DPP-MAX-EST, we can directly evaluate the benefits of using the acquisition weight.

As shown in Tab. 6.1,6.2,6.3 and Fig. 6.1,6.2,6.3, LAW-EI, LAW-EST, q-EI and q-EST are in top four except for FSP-car5 and TSP-att48. LAW-EST performs the best on FSP and TSP while q-EI or q-EST perform the best on QAP. Along with the experiment on the structure learning (Subsec. 6.5.3), we conjecture that QAP has a certain structure more friendly to q-EI and q-EST. Also LAW-EI exhibits comparable performance with q-EI and q-EST while outperforming other

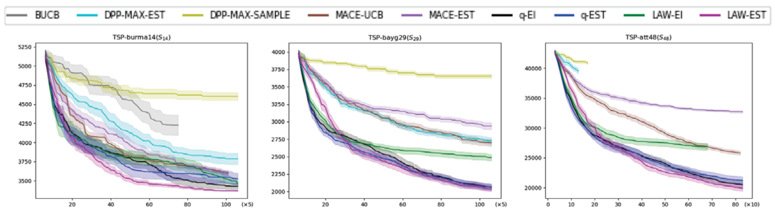


**Figure 6.3:** Traveling Salesman Problems (15 runs)

**Table 6.3:** Traveling Salesman Problems (15 runs)

| Benchmarks<br>Batch | TSP-burma14<br>5 | TSP-bayg29<br>5 | TSP-att48<br>10 |
|---|---|---|---|
| BUCB | $4184 \pm 132$ (8) | - | - |
| DPP-MAX-EST | $3786 \pm 74$ (7) | $2727 \pm 50$ (6) | $39539 \pm 487$ (7) |
| DPP-SMP-EST | $4603 \pm 52$ (8) | $3653 \pm 29$ (8) | $40893 \pm 265$ (8) |
| MACE-UCB | $3583 \pm 21$ (6) | $2698 \pm 50$ (5) | $25773 \pm 371$ (4) |
| MACE-EST | $3576 \pm 25$ (5) | $2940 \pm 49$ (7) | $32711 \pm 212$ (6) |
| q-EI | $3427 \pm 40$ (3) | $2065 \pm 36$ (3) | $20472 \pm 502$ (2) |
| q-EST | $3527 \pm 75$ (4) | $2060 \pm 48$ (2) | $21199 \pm 620$ (3) |
| LAW-EI | $3466 \pm 26$ (2) | $2487 \pm 47$ (4) | $26864 \pm 589$ (5) |
| LAW-EST | $3369 \pm 7$ (1) | $2038 \pm 36$ (1) | $19846 \pm 485$ (1) |

baselines, which supports the benefit of the acquisition weight. In terms of the average rank over all benchmarks, LAW-EST performs the best with the average rank of 1.89 against q-EST (2.44) and q-EI (2.78). Overall, among the baselines, LAW-EST exhibits stable and competitive performance across different benchmarks.

*Comparison to the local penalization (LP)*

Two additional variants of LAW, LAW-PRIOR-EST and LAW-PRIOR-EI, are also compared (Appx. D.4), which use the prior covariance function as the diversity gauge, $L = K$, of $L_t^{AW}$. These variants do not use evaluation data in the diversity gauge.

Interestingly, LAW-PRIOR-EST and LAW-PRIOR-EI resemble the local penalization (LP) (González et al., 2016) (Appx. D.3.2), and thus this allows an indirect comparison to LP which is not applicable to combinatorial spaces.[32]

These variants using the prior covariance function performs worse than LAW using the posterior covariance function, which is natural since using more data for the diversity gauge enhances the performance. More importantly, LAW-PRIOR-EST and LAW-PRIOR-EI outperform DPP-MAX-EST which uses the posterior covariance function

32 LAW variants use the kernel of the GP surrogate model as the diversity gauge which is more guided by data while LP uses the local penalizer which is heuristically designed. We expect that this distinction will still make a difference on the performance.

without the acquisition weight, which supports that the acquisition weight is key in the performance improvement.

Based on the empirical analysis above, we choose LAW-EST as our final recommendation, which we call LAW2ORDER.

### 6.5.3 Structure Learning

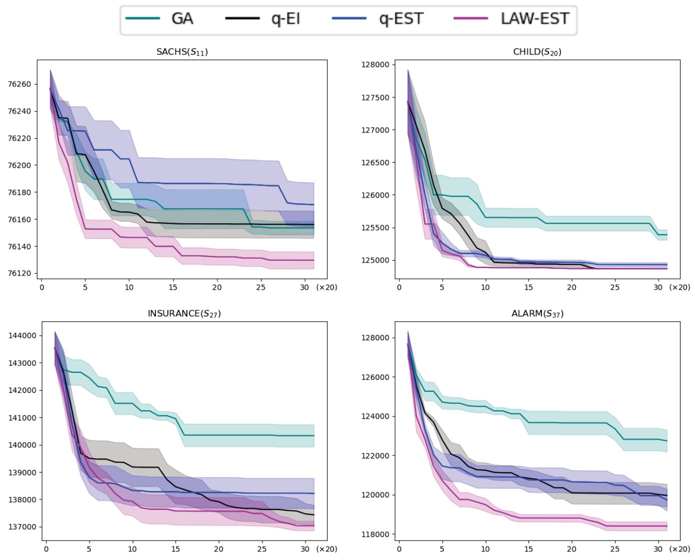


**Figure 6.4:** Neg. log NML minimization for the structure learning (5 runs)

**Table 6.4:** Neg. log NML minimization for the structure learning (5 runs)

| BN(#Node) | | Sachs(11) | Child(20) | Insurance(27) | Alarm(37) |
|---|---|---|---|---|---|
| Sec. to eval. | | $60 \sim 80$ | $120 \sim 140$ | $150 \sim 170$ | $200 \sim 220$ |
| Method | #Eval | $C = 76100$ | $C = 124000$ | $C = 135000$ | $C = 117000$ |
| GA | 620 | $53.46 \pm 4.99$ | $1387.12 \pm 79.26$ | $3330.60 \pm 406.92$ | $4825.19 \pm 570.55$ |
| GA | 1240 | $31.90 \pm 5.86$ | $1368.07 \pm 92.26$ | $2814.04 \pm 418.49$ | $4114.97 \pm 449.93$ |
| q-EI | 620 | $55.98 \pm 10.11$ | **864.85** $\pm 0.16$ | $433.23 \pm 357.18$ | $2969.00 \pm 518.67$ |
| q-EST | 620 | $70.67 \pm 16.31$ | $928.83 \pm 32.97$ | $1215.75 \pm 556.36$ | $2739.77 \pm 554.12$ |
| LAW-EST | 620 | **29.58** $\pm 6.36$ | *866.64* $\pm 0.39$ | **33.95** $\pm 174.04$ | **1409.27** $\pm 227.57$ |

We apply LAW2ORDER to the score-based structure learning problem (Drton and Maathuis, 2017). Existing score-based methods assume a computationally amenable structure of the score to be optimized (decomposability) (Koller and Friedman, 2009; Scutari et al., 2019). Distinctively our approach does not necessitate the decomposability of the score to be optimized.

We consider the NML score as below

$$S_{NML}(\mathcal{G}, \mathcal{D}) = -\log p_{BN}(\mathcal{D} \mid \mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D})) + REG_{NML}(\mathcal{G}, |\mathcal{D}|)$$

where $p_{BN}(\cdot \mid \mathcal{G}, \theta)$ is the density of Bayesian Network (BN) with DAG $\mathcal{G}$ and the parameter $\theta$, and $REG_{NML}$ is the normalized marginal likelihood (NML) which is a complexity measure from the minimum description length principle (P. D. Grünwald and Grunwald, 2007). NML is not decomposable, and thus the methods assuming a decomposable score are not applicable. Since it is infeasible to compute NML exactly and we resort on MC estimate, the NML score evaluation is noisy. In addition to not being decomposable, noisy evaluation also makes existing methods inapplicable to the NML score objective. For more details of NML and its MC estimate, see Appx. D.3.4.

Similarly to (Raskutti and Uhler, 2018; Solus et al., 2021) we search over permutations specifying topological order of DAGs and the existence of edges is determined by the conditional independence test. In addition to q-EI and q-EST performing well on the benchmarks, we compare LAW2ORDER with the genetic algorithm (GA), which is one of the most popular choices for optimization problems on permutations including TSP (Potvin, 1996).

We generated 5 sets of 20 random initial points. LAW2ORDER, q-EI and q-EST is run on each of these 5 sets using a batch size 20. Assuming the same resource constraint (at most 20 evaluations in parallel), GA generates 20 off-springs in each generation. GA is also run 5 times with a population size of 100 points using Pymoo (Blank and Deb, 2020). The first 20 points of each initial population in the 5 runs are equal to the 20 initial points used in LAW2ORDER, q-EI and q-EST. Even though the real deployment of Bayesian optimization assumes that the cost of evaluation is expensive enough to render the time to acquire new batches negligible, as a stress test, we allowed twice the evaluation budget for GA.

On data generated from 4 real-world BNs (Scutari, 2010; Scutari et al., 2019), the results are reported in Fig. 6.4 and Tab. 6.4. LAW2ORDER outperforms q-EI and q-EST with a significant margin except for Child where all three find the putative optimum quickly. Still, in Child, LAW2ORDER finds a point of negligible differences with the putative optimum the most quickly. Also, except for Sachs, we observe that the performance gap increases as the permutation size (the size of the search space) increases. On this realistic problem, our argument that LAW2ORDER is stable and efficient batch acquisition method on permutations is reinforced.

In comparison with GA, we consider GA(620) with the same evaluation budget and GA(1240) with twice large evaluation budget. LAW2ORDER dominates GA(620) in all problems with a significant margin. Even compared with GA(1240), LAW2ORDER significantly outperforms except for Sachs which has one of the smallest search spaces (See Tab. 6.4 and Appx. D.4). Contrary to our expectation that BO has the sample efficiency higher than GA, GA(620) outperforms q-EI and q-EST on Sachs. LAW2ORDER shows robust performance even in the problem where the performance of q-EI and q-EST is degraded.

Together with the experiments on the benchmarks, LAW2ORDER is shown to be a robust batch acquisition method on permutations. Moreover, promoting the diversity in batches while taking into account the acquisition weight appears more beneficial with larger batch size (20) as shown in the structure learning experiment.

## 6.6 CONCLUSION

In this chapter we have focused on combinatorial optimization problems over permutations where each evaluation is assumed to be expensive. This class of problems has many interesting applications, ranging from chip design (where we wish to place cells while minimizing area and wire-length), warehouse optimization (where we need to order the retrieval of items from a warehouse using a robot), neural architecture search and so on. In spite of its practical significance, BO on permutations is under-explored in contrast to the recent progress on combinatorial BO with categorical variables.

In response to this, we have proposed a batch Bayesian optimization algorithm on permutations, LAW2ORDER, which uses an extension of the determinantal point processes with the acquisition weighted kernel. This allows the search process over the surrogate function to make optimal use of all parallel available computational resources and be guided by both the expected objective value and its posterior uncertainty.

On the theory side we offer a regret analysis, which shows that the regret bound of LAW enjoys the same asymptotic behavior as existing methods. On the empirical side, we show that LAW variants consistently exhibit competitive performance on a wide range of combinatorial optimization tasks, including a challenging structure learning problem.

From these we conclude that the acquisition weights are indeed a key factor in the success of the proposed method, and that the performance gains increase for large batch sizes.

LAW is applicable to general search spaces for which a kernel can be defined. We leave the exploration of our method to applications outside searching over permutations for future work.

LAW achieves improved sample efficiency in the sense that the quality of batches from LAW is maintained for large batch sizes. However, due to the sequential nature of the greedy maximization of LAW objective, its computational complexity is linear with respect to the batch size. Especially for large permutation spaces, this may be a nonnegligible cost. We hope that the sample efficiency of LAW is complemented by the computational efficiency allowing massive parallelization.

Even though the regret bound in Thm. 6.1 describes the effect of the acquisition weight, as mentioned in Rmk. 6.4, it only sheds light on the demerit of excessive emphasis on the acquisition weights but it does not detect the demerit of considering the diversity only. We hope our work inspires the regret bound for acquisition methods taking into account properties other than diversity.

# 7 BAYESIAN OPTIMIZATION FOR MACRO PLACEMENT

Macro placement is the problem of placing memory blocks on a chip canvas. It can be formulated as a combinatorial optimization problem over sequence pairs, a representation which describes the relative positions of macros. Solving this problem is particularly challenging since the objective function is expensive to evaluate. In this chapter, we develop a novel approach to macro placement using Bayesian optimization (BO) over sequence pairs. BO is a machine learning technique that uses a probabilistic surrogate model and an acquisition function that balances exploration and exploitation to efficiently optimize a black-box objective function. BO is more sample-efficient than reinforcement learning and therefore can be used with more realistic objectives. Additionally, the ability to learn from data and adapt the algorithm to the objective function makes BO an appealing alternative to other black-box optimization methods such as simulated annealing, which relies on problem-dependent heuristics and parameter-tuning. We benchmark our algorithm on the fixed-outline macro placement problem with the half-perimeter wire length objective and demonstrate competitive performance.

## 7.1 INTRODUCTION

In chip placement two different types of objects are placed on a chip canvas: macros, which are large memory blocks, and standard cells, which are small gates performing logical operations. Compared to macros, standard cells are typically thousands of times smaller but tens or hundreds of thousands of times more numerous. While standard cell placement can be efficiently solved using continuous optimization, e.g. (C.-K. Cheng et al., 2019), macro placement is typically framed as a combinatorial optimization problem due to their larger physical size. This involves searching over the discrete set of relative positions between pairs of macros, e.g. whether macro $i$ is to the left or right of macro $j$, which act as constraints against overlapping macros. The most popular combinatorial representation of relative positions is called the sequence pair which is composed of a pair of permutations, one per spatial dimension (Murata et al., 1996).

The goal of macro placement is to place macros in such a way that the power, performance and area metrics are jointly optimized. The combinatorial nature and varying sizes of macros and standard cells, together with the cost of evaluating the objective function (several

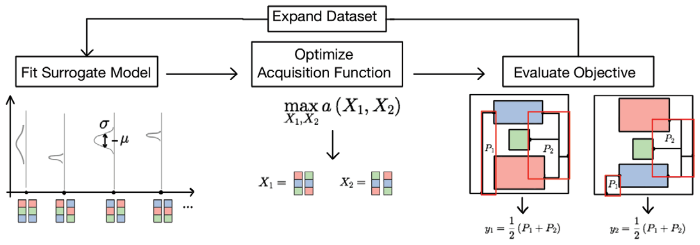

Expand Dataset
Fit Surrogate Model
Optimize Acquisition Function
Evaluate Objective
$\sigma$
$\mu$
$\max_{X_1,X_2} a(X_1, X_2)$
$X_1 =$
$X_2 =$
$P_1$
$P_2$
$y_1 = \frac{1}{2}(P_1 + P_2)$
$y_2 = \frac{1}{2}(P_1 + P_2)$

**Figure 7.1:** Bayesian optimization for a macro placement workflow for $N = 3$ macros and batch size $B = 2$. First we fit a surrogate model (Gaussian process) to the data, updating the mean $\mu$ and standard deviation $\sigma$ of the cost function estimate for each sequence pair, here represented as a pair of N dimensional arrays where permutations correspond to different colors patterns. Then we optimize an acquisition function $a$ conditioned on the data observed so far to find new sequence pairs $x_1, \ldots, x_B$. Next we evaluate $x_1, \ldots, x_B$ by placing macros to minimize HPWL while respecting the sequence pair constraints, and compute the corresponding objective values $y_1, \ldots, y_B$. Here $P_i$ is the perimeter of the $i$-th bounding box of the net between macros and I/O pads. Finally we add these new points to the dataset and repeat the procedure until the computational budget is exhausted.

days for complex designs), make macro placement a notoriously challenging step in physical design. Macro placement is also related to floorplanning, where standard cells are clustered in soft rectangles that are jointly placed with hard rectangles that represent macros (A. Kahng et al., 2011). In practice, designers manually place macros based on their intuition which is likely sub-optimal.

Machine learning algorithms offer an advantage over traditional optimization algorithms for macro placement since they can learn from past designs and improve over time in an automated fashion, adapting the algorithms to specific use cases. Applying machine learning to physical design has therefore recently emerged as a main research effort in electronic design automation (A. B. Kahng, 2018). In particular, reinforcement learning (RL) provides a natural framework for automating design decisions, where an agent plays the role of a designer in carefully selecting parameter configurations to evaluate next while searching for optimal solutions. However, in practice applying RL is very costly because of the large number of samples required for learning a good policy, due in part to the very large design space and costly evaluation as remarked above. For this reason, to the best of our knowledge, applications of RL in the literature are either limited to a handful of parameters (Agnesina et al., 2020) or require the use of cheap proxies instead of the real objective (Mirhoseini et al., 2021), which changes the focus towards designing good proxies.

Bayesian optimization (BO) is a technique that is well-known for its sample-efficiency, whereby it carefully explores the optimization landscape through selecting a candidate based on previous evaluations (Shahriari et al., 2015). Compared with other black-box function optimization methods such as RL, genetic algorithms, or simulated annealing (SA), the sample efficiency of BO allows flexibility for the macro placement. Especially when it is desirable to perform optimization close to the real objective, not a proxy, inevitable evaluation cost leaves only BO as a viable option. For the application to macro placement, more relevant is BO on combinatorial structures (Baptista and Poloczek, 2018; Deshwal, Belakaria, and J. R. Doppa, 2021; Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022; Deshwal and J. Doppa, 2021; Changyong Oh, Tomczak, et al., 2019). For the details on BO on combinatorial structures, please refer to the references.

**Contributions** In this chapter we introduce BO on sequence pairs for macro placement as a replacement for other black-box optimization methods such as SA which are routinely applied in the literature (S. N. Adya and Igor L Markov, 2002; Xu et al., 2017). We use batch BO for parallel evaluation of a batch of data points to accelerate the optimization. Fig. 7.1 summarizes our workflow.

Our main contributions are:

- We extend batch BO on permutations in Ch. 6 to batch BO on sequence pairs and devise an efficient algorithm for parallel batch acquisition function optimisation.
- We benchmark our algorithm on the MCNC dataset and obtain superior performance in terms of wire length metric as compared to SA.

## 7.2 BACKGROUND

### 7.2.1 Sequence pair

Sequence pairs (SPs) were introduced in (Murata et al., 1996) as a combinatorial representation for macro packing problems. We recall that a macro is a rectangle with distinguished points called pins which may connect wires. For macros $\{m_1, \cdots, m_N\}$, an SP is a pair of permutations of length $N$, one per spatial dimension, which specifies the relative location of each pair of macros. The relationship between the four possible relative locations of macros $m_i$ and $m_j$ and SPs are explained in Tab. 7.1.

Traditionally, the SP representation has been used in macro placement for optimizing area and half perimeter wire length (HPWL) (Murata et al., 1996). HPWL is the half perimeter of the bounding box around a net (e.g. the red boxes in Fig. 7.1). To convert the SP to a packed placement, an algorithm called the Longest Common Subsequence (LCS) is used. For a given SP, i.e. given relative locations

Table 7.1: Relative location specified by sequence pair $(\pi, \pi')$

| $\pi$ | $\pi'$ | Relative location of $i$ and $j$ |
|---|---|---|
| $(\cdots, i, \cdots, j, \cdots)$ | $(\cdots, i, \cdots, j, \cdots)$ | $i$ is on the left of $j$ |
| $(\cdots, j, \cdots, i, \cdots)$ | $(\cdots, j, \cdots, i, \cdots)$ | $i$ is on the right of $j$ |
| $(\cdots, i, \cdots, j, \cdots)$ | $(\cdots, j, \cdots, i, \cdots)$ | $i$ is below $j$ |
| $(\cdots, j, \cdots, i, \cdots)$ | $(\cdots, i, \cdots, j, \cdots)$ | $i$ is above $j$ |

among macros, it ensures minimal area placement, where no further vertical or horizontal adjustment of any macro is possible (Murata et al., 1996).

Simulated annealing (SA) is commonly used to search over the space of SPs by carefully-designed stochastic moves (S. Adya and I. Markov, 2001). The optimization objective is typically a linear combination of area and HPWL. The conversion from an SP to a placement is the main computational bottleneck. Since SA requires several thousands of evaluations to find a good solution, a cheap proxy for the objective that relies on LCS is used in practice (S. Adya and I. Markov, 2001). Another direction of work focused on the efficient LCS implementations to handle this computational bottleneck (Tang, Tian, et al., 2001; Tang and D. Wong, 2001). In contrast, our work aims to optimize a complex and expensive objective through a BO routine, while using LCS to assert whether an SP can be converted to a placement which fits within the fixed placement region.

### 7.2.2 Bayesian optimization

BO has been widely successful in optimizing expensive-to-evaluate objectives such as hyperparameter optimization (Snoek, Rippel, et al., 2015), neural architecture search (V. Nguyen, T. Le, et al., 2021) and optimization of the tokamak control for nuclear fusion (Char et al., 2019). The superior sample efficiency of BO is attributed to two components, namely the surrogate model and the acquisition function. The surrogate model is a probabilistic model that approximates the objective while measuring the uncertainty of its approximation. This uncertainty plays a crucial role in the exploration-exploitation tradeoff. For this reason, Gaussian processes (GPs) are widely used due to their superior uncertainty quantification (Snoek, Larochelle, et al., 2012; Williams and Rasmussen, 2006). Given a point $x$ in the search space, at the $t$-th iteration of BO, the predictive mean $\mu_t(x)$ and the predictive covariance $K^{(t)}(x, x')$ of the GP surrogate model are defined as

$$\begin{aligned} \mu_t(x) &= m_x + K_{x,\mathbf{X}^{(t)}}(K_{\mathbf{X}^{(t)},\mathbf{X}^{(t)}} + \sigma^2 I)^{-1}(\mathbf{y}^{(t)} - m_{\mathbf{X}^{(t)}}) \\ K^{(t)}(x, x') &= K_{x,x'} - K_{x,\mathbf{X}^{(t)}}(K_{\mathbf{X}^{(t)},\mathbf{X}^{(t)}} + \sigma^2 I)^{-1} K_{\mathbf{X}^{(t)},x'} \end{aligned} \tag{7.1}$$

where $m_\cdot$ is the mean function, $K_{\cdot,\cdot}$ is the kernel (i.e. prior covariance function), $\sigma^2$ is the variance of the observational noise and $\mathbf{X}^{(t)}$ is the set of points evaluated so far. The predictive variance is $\sigma_t^2(x) = K^{(t)}(x, x)$.

Using the GP predictive distribution conditioned on the evaluation dataset $(\mathbf{X}^{(t)}, \mathbf{y}^{(t)})$, the acquisition function $a(\cdot, \cdot)$ quantifies the chance that the evaluation of a point improves the GP optimization. An acquisition function is based on the intuition that the predictive mean and the predictive variance can be used to make an informed guess about the usefulness of a point in the input space (Shahriari et al., 2015):

$$a^{(t)}(x) = a(\mu_t(x), \sigma_t^2(x)) \,. \tag{7.2}$$

In general, the acquisition function value is higher at points where the predictive mean and the predictive variance are relatively high. The argument of the maximum of the acquisition function $x_{opt}^{(t)}$ is evaluated under the true objective $y^{(t)} = f(x_{opt}^{(t)})$. This new datapoint is then added to the evaluation dataset and the BO process is repeated.

$$\mathbf{X}^{(t+1)} = [\mathbf{X}^{(t)}; x_{opt}^{(t)}], \quad \mathbf{y}^{(t+1)} = [\mathbf{y}^{(t)}; y^{(t)}]$$

BO can be accelerated when computational resources permit parallel evaluation of the objective. In this case, the acquisition function is defined over multiple points so that its optimization yields multiple points whose evaluation can be parallelized.

$$\{x_{opt,b}^{(t)}\}_{b=1}^{B} = \operatorname*{argmax}_{x_1, \cdots, x_B} a_{batch}^{(t)}(x_1, \cdots, x_B)$$

This is called batch BO. Several works have proposed methods which use different batch acquisition functions(Gong et al., 2019; González et al., 2016; Lyu et al., 2018; Wu and Frazier, 2016). For a detailed overview of BO, the reader is referred to (Brochu et al., 2010; Frazier, 2018; Shahriari et al., 2015).

## 7.3 METHODOLOGY

In contrast to the traditional macro placement approaches, we consider an expensive-to-evaluate objective with which we perform macro placement. We also consider the fixed-outline constraint addressed in many existing works (S. Adya and I. Markov, 2001).

In order to efficiently tackle macro placement, we employ batch BO on the space of sequence pairs (SPs) – a pair of two permutations – a compact representation for the relative positions of macros. Sequence pairs describe non-overlapping placements of macros. Intuitively, if we imagine the macros to be placed on a line, the space of non-overlapping placements can be indexed by permutations of the macros, and a macro optimization problem with non-overlapping constraints can be solved by searching over the space of permutations. In the two-dimensional setting of this chapter we need a pair of permutation to describe non-overlapping macro placements. See Appx. Subsec. 7.2.1 and (Murata et al., 1996) for detailed explanation. To this end, we introduce *1)* a kernel on the space of SPs, *2)* an efficient heuristic to optimize the batch acquisition function *3)* an efficient vectorized Python implementation of the least common subsequence (LCS) algorithm of $O(N \log N)$ run-time complexity where $N$ is the number of macros.

For our Gaussian process surrogate model in BO, the proposed kernel is based on the position kernel on permutation due to its superior performance in (batch) BO on permutation spaces (Zaefferer et al., 2014). Denoted $\pi_1, \pi_2, \pi_1', \pi_2'$ four permutations of $N$ elements, our kernel is:

$$K_{sp}((\pi_1, \pi_1'), (\pi_2, \pi_2') | \mathbf{W}, \mathbf{W}') = K_{perm}(\pi_1, \pi_2 | \mathbf{W}) \cdot K_{perm}(\pi_1', \pi_2' | \mathbf{W}')$$

where $\mathbf{W} = [w_1, \cdots, w_N]$, $\mathbf{W}' = [w_1', \cdots, w_N']$, and

$$K_{perm}(\pi_1, \pi_2 | w_1, \cdots, w_N) = \exp\Big( \sum_{n=1}^{N} w_n \cdot |\pi_1^{-1}(n) - \pi_2^{-1}(n)| \Big)$$

$K_{perm}(\pi_1, \pi_2 | 1, \cdots, 1)$ is the position kernel (Zaefferer et al., 2014). In addition to the mentioned works, in our position kernel we introduce parameters $\mathbf{W}, \mathbf{W}'$ that account for widths and heights of different macros. We optimize those parameters by maximizing the marginal likelihood of the training data by gradient descent.

In the batch acquisition function of BO, we adopt the method in Ch. 6 which uses determinantal point processes (DPPs) with a weighted kernel to obtain a batch of diverse points each of which is likely to speed up BO progress. DPPs quantify the diversity of points using determinant of the gram matrix. Since the determinant of a matrix is the volume of a hyper-parallelepiped whose vertices are columns of the matrix, the more diverse the points are, the larger the determinant is (Kulesza and Taskar, 2010, 2012). The batch acquisition function is defined as

$$\begin{aligned} & a_{batch}^{(t)}(x_1, \cdots, x_B) \\ & \qquad = \det \left[ \rho\big(a^{(t)}(x_b)\big)\, K^{(t)}(x_b, x_c)\, \rho\big(a^{(t)}(x_c)\big) \right]_{b,c=1,\cdots,B} \end{aligned}$$

where $K^{(t)}$ the covariance function conditioned on the evaluation dataset $\mathbf{X}^{(t)}$ as in Eq. (7.1), $a^{(t)}$ is the acquisition function for a single point as in Eq. (7.2) (e.g. EI, UCB, EST(Zi Wang, B. Zhou, et al., 2016)), and $\rho(\cdot)$ is a positive and strictly increasing function. This batch acquisition function balances the quality of each point (i.e. the likelihood of improving the objective) through the acquisition weight $\rho\big(a^{(t)}(\cdot)\big)$, and the diversity among points (i.e. avoiding information redundancy in parallel evaluations) through $K^{(t)}$. This function has been demonstrated to perform well for BO on permutation spaces in Ch. 6.

While $a^{(t)}_{batch}$ effectively fulfills the quality and diversity requirements, its optimization is computationally demanding. In Ch. 6, a greedy approach was employed with certain optimality guarantees. However, that method optimizes $a^{(t)}_{batch}$ sequentially over the batch index and limits the scalability of batch BO. Therefore, we propose a new heuristic for parallel optimization of the batch acquisition function (Alg. 5).

**Algorithm 5** Parallel heuristic

```
Require: $a$ : an acquisition function for a single point
    $a_{batch}$ : a batch acquisition function
    $u_{local}(g(\cdot), x)$ : a local update function
        such that $g(u_{local}(g(\cdot), x)) \geqslant g(x)$(maximization)
    $\mathbf{X}_{feasible}$ : a feasible set
 1: $x_{opt,1} = argmax_{x \in \mathbf{X}_{feasible}} a(x)$
 2: Randomly choose $x_{0,b}$ for $b = 2, \cdots, B$ from $\mathbf{X}_{feasible}$
 3: repeat
 4:   $\mathbf{X}_{feasible,s} = \emptyset$
 5:   for $b \in \{2, \cdots, B\}$ do {Parallel}
 6:     Update $x_{s,b}$
        $a_{s,b}(x) = a_{batch}(x_{opt,1}, x_{s,2}, \cdots, x_{s,b-1}, x, x_{s,b+1}, \cdots)$
        $x_{s+1,b}, \mathbf{X}_{feasible,s,b} = u_{local}(a_{s,b}(\cdot), x_{s,b})$
 7:     Collect feasible sets
        $\mathbf{X}_{feasible,s} = \mathbf{X}_{feasible,s} \cup \mathbf{X}_{feasible,s,b}$
 8:   end for
 9:   Expand the feasible set
      $\mathbf{X}_{feasible} = \mathbf{X}_{feasible} \cup \mathbf{X}_{feasible,s}$
10:   Update step count $s = s + 1$
11: until Convergence or other stopping criteria
12: return $(x_{opt,1}, x_{\cdot,2}, \cdots, x_{\cdot,B}), \mathbf{X}_{feasible}$
```

The main idea is to perform small local updates in parallel for each element of the batch. Specifically, we first compute $x_{opt,1}$, the optimum of the single point acquisition function (See line no. 1 of Alg. 5). Then we optimize the function $a_{s,b}$ defined by fixing all but the $b$-th element of the batch, for $b = 2, \ldots, B$ (See line no. 6 of Alg. 5). This step can be parallelized over the batch. Here $s$ denotes the iteration time over which this procedure is repeated. When a single point is updated (line no. 1 of Alg. 5), we apply a small local update instead of running until convergence to minimize the deviation of our individual updates from the simultaneous update method. Intuitively, if any single point is significantly altered while the rest is fixed, the end result of the individual updates will drastically differ from that of the simultaneous update.

The parallel heuristic (Alg. 5) takes as input a local update function. The local update function (Alg. 6) checks the constraint of fixed outline of the placement region. We call feasible SPs those SPs that fit into the placement region.

**Algorithm 6** Local update with feasibility check

```
Require: $g(x)$ : an objective function
    $x_{old}$ : an initial point
    $c_{feasible}(x)$ : a function checking the feasibility
    $\mathcal{N}(x)$ : a function listing neighbors of $x$
1: Find neighbors of $x_{old}$, $\mathcal{N}(x_{old})$
2: Compute feasibility
   $\mathcal{N}_{feasible}(x_{old}) = \{x \in \mathcal{N}(x_{old}) \mid c_{feasible}(x) \text{ is true}\}$
3: Move toward the best feasible neighbor
   $x_{new} = \mathrm{opt}_{x \in \mathcal{N}_{feasible}(x_{old})} g(x)$
4: Expand the feasible set
   $X_{feasible} = X_{feasible} \cup \mathcal{N}_{feasible}(x_{old})$
5: return $x_{new}, X_{feasible}$
```

By using Alg. 6 as the local update function for the parallel heuristic (Alg. 5), the latter collects feasible points by accumulating the feasible sets generated by the former. When the local update function (Alg. 6) is invoked in the parallel heuristic (Alg. 5), $c_{feasible}(\cdot)$ is the function which asserts the fixed-outline constraint using the LCS algorithm, and $\mathcal{N}(x)$ is the set of neighbors of the sequence pair $x$ obtained by swapping adjacent elements in each permutation.

Given the kernel $K_{sp}$, the batch acquisition function, and the parallel heuristic for its optimization, we present the complete Bayesian optimization for macro placement workflow in Alg. 7. See also Fig. 7.1 for a graphical illustration.

## 7.4 RELATED WORK

Sequential macro placers (S. N. Adya and Igor L Markov, 2002; Igor L. Markov et al., 2015; Xu et al., 2017) produce overlap-free placements for macros in four steps:

1. cluster standard cells into soft rectangles,
2. run a floorplanner on the original (hard) macros and new soft rectangles,
3. remove the soft rectangles,
4. place standard cells with fixed macros.

The floorplanner of choice is typically based on SA over sequence pairs with the most popular implementation being Parquet (S. Adya

**Algorithm 7** Batch Bayesian optimization macro placement

```
Require: $f$ : the optimization objective
    $\mathbf{X}^{(0)}_{feasible}$ : an initial feasible set
    $\mathbf{X}^{(0)}, \mathbf{y}^{(0)}$ : an initial evaluation dataset
 1: repeat
 2:     Fit the surrogate model on the data ($\mathbf{X}^{(t)}$, $\mathbf{y}^{(t)}$)
        $\mu_t(x)$, $\sigma_t^2(x)$        [Used in the acquisition function]
 3:     Optimize the acquisition function
        by calling Alg. 5 with
        - Local update fn.: Alg. 6
        - Feasible set: $\mathbf{X}^{(t)}_{feasible}$
        $(x_1^{(t)}, \cdots, x_B^{(t)})$, $\mathbf{X}^{new}_{feasible} \leftarrow$ Alg. 5
 4:     Evaluate the objective at $(x_1^{(t)}, \cdots, x_B^{(t)})$ in parallel
        $y_1^{(t)} = f(x_1^{(t)}), \cdots, y_B^{(t)} = f(x_B^{(t)})$
 5:     Expand the evaluation dataset
        $\mathbf{X}^{(t+1)} = [\mathbf{X}^{(t)}; x_1^{(t)}; \cdots; x_B^{(t)}]$
        $\mathbf{y}^{(t+1)} = [\mathbf{y}^{(t)}; y_1^{(t)}; \cdots; y_B^{(t)}]$
 6:     Expand the feasible set
        $\mathbf{X}^{(t+1)}_{feasible} = \mathbf{X}^{(t)}_{feasible} \cup \mathbf{X}^{new}_{feasible}$
 7:     Update BO round count $t = t + 1$
 8: until Computational budget is exhausted
 9: return $\mathbf{X}^{(\cdot)}$, $\mathbf{y}^{(\cdot)}$
```

and I. Markov, 2001) which incorporates several heuristics to select new configurations. Modern sequential workflows such as the Triton macro placer included in the OpenRoad project (*The-OpenROAD-Project* 2021) use RePlace (C.-K. Cheng et al., 2019) for standard cell placement. RePlace is a state-of-the-art academic analytical placer that uses an electrostatic analogy whereby cells and macros are modelled as charged objects with charges proportional to their areas, and their electrostatic equilibrium leads to a uniformly spread placement. Performing joint macro and standard cell placement using RePlace produces overlaps that must be later removed by a legalization step, as done in (J. Lu et al., 2015) that also uses SA.

In all the aforementioned workflows we can replace SA with our BO algorithm. SA requires many iterations to converge and does not scale when using realistic cost functions, which limits the choice of cost functions that designers can feasibly use for SA and thus may lead to important aspects of the problem being ignored. Furthermore, SA requires the designer to carefully adjust parameters such as the temperature schedule and the acceptance probability to obtain good results – though (Vashisht et al., 2020) proposes an algorithm that learns to propose good initial values. In contrast, in BO the kernel hyperparameters can be tuned automatically by fitting the training data with gradient-based optimization. Nevertheless, acquisition function maximization in our combinatorial setting requires some tuning, see Sec. 7.3.

Various techniques other than SA have been proposed for floorplanning. In (Funke et al., 2016) an exact enumeration algorithm is applied to larger problems using a divide-and-conquer strategy. However, this method can only be applied to the half-perimeter wire length objective and not more realistic cost functions. Similarly, (W. Liu and Nannarelli, 2008; Samaranayake et al., 2009) also use wire length proxy functions.

A closely-related work to ours uses RL for macro placement (Mirhoseini et al., 2021). RL requires many training iterations to converge to a good policy, while BO is more data-efficient and is therefore more appealing when evaluating an expensive reward function. In contrast to RL, BO does not learn to act in multiple situations, meaning that each new design requires optimization from scratch. BO can be seen as a simplified instance of RL where feedback – evaluations for BO and rewards for RL – is given after taking a single action (instead of a sequence of actions) with a single state – no state transition – as in bandits (Chowdhury and Gopalan, 2017; Srinivas et al., 2010; Sutton and Barto, 2018). Other practical differences of our work and (Mirhoseini et al., 2021) are: 1) their RL agent places macros sequentially while we jointly place all macros as done in SA; 2) they discretize the macro positions on a fictitious grid while we work in the exact continuum optimization formulation with no overlap constraints. In Sec.7.5 we compared our results against our SA implementation and previously reported methods on the same benchmark dataset. We leave benchmarking against (Mirhoseini et al., 2021) as future work.

Recently, BO was tested on similar but much smaller cases in (Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022). Their focus is on proposing a new kernel on permutations and it is orthogonal to our focus on the batch acquisition to tackle super-exponential growth. In contrast to (Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022), our experiments were conducted on much larger spaces i.e., 3~4 times more macros – in terms of the size of search space, this makes a huge difference due to the super-exponential growth of permutation spaces – and demonstrated the effectiveness of the batch acquisition. We leave the search for the optimal combination of the kernel and the batch acquisition method for macro placement as a future work.

Finally, we note that in (Xu et al., 2017) a bandit-based approach similar to BO has been applied to optimizing parameters of FPGA compilation. This work does not tackle the challenges of large combinatorial spaces in macro placement.

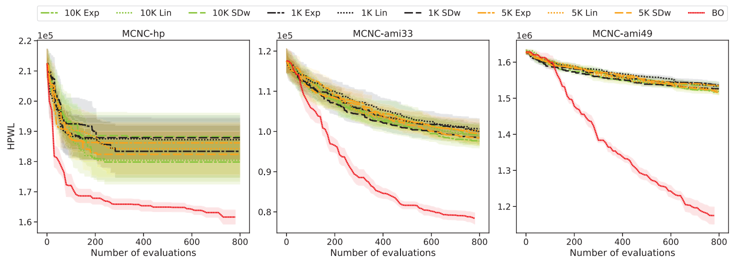


**Figure 7.2:** BO vs SA on MCNC (*hp*, *ami33*, *ami49*)

## 7.5 EXPERIMENTS

As a demonstration of the potential of BO, we test it on the MCNC benchmark (Kozminski, 1991)[33] and present the results in Tab. 7.2. The optimization objective is to minimize HPWL which connects macros pins to I/O pads under the fixed-outline constraint. Note that I/O pads are fixed on the boundary of the placement region. Under the relative location constraints specified by a sequence pair, we perform the linear constrained programming to minimize HPWL. Note that this objective is simpler and cheaper-to-evaluate than the ones where BO can possibly show its full strengths. Nonetheless, this experiment does indicate the potential of BO in macro placement.

**Table 7.2:** HPWL results on the MCNC benchmark.

| - ()* | *apte* (9) | *xerox* (10) | *hp* (11) | *ami33* (33) | *ami49* (49) |
|---|---|---|---|---|---|
| eWL[&] | 513,061 | 370,993 | 153,328 | 58,627 | 640,509 |
| ELS[#] | 614,602 | 404,278 | 253,366 | 96,205 | 1,070,010 |
| FD[%,a] | 545,136 | 755,410 | 155,463 | 63,125 | 871,128 |
| SA[b,d] | 515,570 ± 525 *1e3, Exp* | 431,108 ±15,312 *1e3, Lin* | 179,826 ±6,550 *1e4, Exp* | 97,691 ±1,592 *1e4, Exp* | 1,517,051 ± 5,095 *1e4, Lin* |
| BO[c,d] | 514,138 ± 264 | 388,936 ± 3,700 | 161,620 ±2,113 | 78,359 ±1,271 | 1,174,972 ±21,396 |

[*] The number of macros in parentheses.
[&] eWL(Funke et al., 2016)
[#] ELS (W. Liu and Nannarelli, 2008)
[%] FD (Samaranayake et al., 2009)
[a] Packings for *hp*, *ami33*, and *ami49* have overlaps.
[b] Result from the best temperature scheduling
[c] BO uses the batch size $B = 10$.
[d] Mean and standard error of 5 runs are reported.

33 `http://vlsicad.eecs.umich.edu/BK/MCNCbench/`

In Tab. 7.2, we compare the black-box function optimizers (BO, SA) with other methods in the literature on all 5 MCNC problems. In Fig 7.2, we compare BO and SA with different temperature scheduling for the 3 largest problems – temperatures (1K, 5K, 10K) and schedules (linear, exponential, stepdown). For all problems, BO has superior performance for the same number of evaluations, with the gap growing larger for larger number of macros. In comparison with the other methods of Tab. 7.2, we can see that BO performs competitively with only 520 evaluations of the objective on *apte* which is the smallest problem. We acknowledge that on the designs with a larger number of macros, *ami33* and *ami49*, there is a non-negligible gap between BO and eWL (Funke et al., 2016). However, we expect that this gap does not translate to the real world applications that we envision since eWL cannot optimize macro placements with standard cells, while BO and SA can. This is because eWL relies on efficient HPWL evaluation and uses a far higher number of evaluations. Moreover, for *apte*, *xerox*, *hp*, eWL performed an exhaustive search. On the other hand, due to super-exponential size of the space of sequence pairs on *ami33* and *ami49*, evaluations are not performed exhaustively but nevertheless are many orders of magnitude larger than the number of BO evaluations. In comparison with ELS (W. Liu and Nannarelli, 2008), BO outperforms in all but *ami49*. However, ELS is a SA tuned for a specific proxy cost function and we expect it not to be transferable to optimize more general and expensive cost functions. Further, in spite of extremely small number of evaluations compared with the size of the search space in *ami49*, BO demonstrates its potential for more general and more realistic objectives on such large number of macros. FD (Samaranayake et al., 2009) outperforms BO on *hp*, *ami33*, *ami49* but the macro locations that FD outputs have overlaps, while our method does not.

## 7.6 CONCLUSIONS

In this chapter, we demonstrated the effectiveness of Bayesian optimization for macro placement, and have shown that it performs competitively with exhaustive search techniques on small benchmarks and performs reasonably well within compute constraints for large benchmarks. In comparison to simulated annealing, we have shown our BO framework to outperform across benchmarks with fewer evaluations. As mentioned above, realistic macro placement quality evaluation requires an expensive global placement loop. Our optimization objective in the experiments was to minimize HPWL of macro to I/O pads connections which helped us evaluate macro placement quality without standard cell placement in the loop. In the future, we plan *1)* to extend this framework with an objective that considers standard cell placement for HPWL computation and congestion estimation *2)* to utilize the current work's output as initial solution to macro placement with subsequent standard cell placement *3)* to extend our constraints with memory stacking requirements, dataflow constraints, channel and snapping constraints, which are typical in industry standard IPs. On the machine learning front, a future challenge is to develop a BO framework that transfers across designs.

# 8 CONCLUSION

The main contribution of this thesis is the extension of the applicability and efficiency of BO beyond low-dimensional Euclidean spaces. Motivated by the empirical successes of BO with the GP surrogate model on low-dimensional Euclidean spaces, we propose GP surrogate models on various search spaces in Chapters 3, 4 and 5. In Chapter 6, focusing on the reduction of BO runtime, we study batch acquisition efficiency utilizing information from a GP. In Chapter 7, we investigate further speed-up of the proposed batch acquisition in application to chip design.

## 8.1 CONCLUSIONS

Based on our proposed methods and their empirical assessments, we provide our answers to the research questions.

**Research Question 1** *What causes the excessive exploration in high-dimensional Euclidean spaces? And how can we mitigate the excessive exploration for effective high-dimensional BO?* (Chapter 3)

We propose the cylindrical kernel using the cylindrical transformation. The cylindrical transformation shrinks the volume near the boundary whereas it expands the volume near the center. This makes BOCK probe the region near the center with higher resolution. Second, if two points are near the boundary, then the cylindrical kernel considers them more similar than when two points are near the center. Thus, for points near the boundary, the cylindrical kernel assigns lower uncertainty than existing kernels (RBF, Matérn) by using the information from distant points more actively.

BOCK shows impressive performance in high dimensional problems up to 500 dimensions – optimization of neural network parameters. To our surprise, compared with SGD, BOCK achieve better generalization even with larger gaps on higher dimensions.

Based on the rationale of the design and experimental results, we conclude that the cylindrical transformation is effective in mitigating the excessive exploration. When the search space can be chosen to have an optimum near its center with high confidence, BOCK is an effective high-dimensional BO.

The assumption that there is an optimum near the center can be too restrictive. In (Eriksson, Pearce, et al., 2019), the region that the surrogate model should focus on is adaptively chosen on the fly. Another strong assumption – the radial symmetry in the cylindrical transformation – can be weakened by employing the elliptical transformation, which can have the ARD effect.

**Research Question 2** *How can we define smooth surrogate models and acquisition functions on combinatorial spaces? And can we develop a flexible GP surrogate model for BO in large combinatorial spaces?* (Chapter 4)

We propose the combinatorial graph to represent a combinatorial search space and the ARD-diffusion kernel to model functions on the combinatorial graph. In combination with a sparsity-inducing prior, COMBO has a variable selection capability while having flexible modeling capacity and computational efficiency.

We demonstrate that COMBO significantly and consistently outperforms BO with a linear model and other competitors on wide range of combinatorial problems.

We conclude that as long as a reasonable definition of smoothness is provided, combinatorial BO can also benefit from the well-calibrated uncertainty of a GP. The combinatorial graph is one way to define smoothness resulting in effective combinatorial BO.

In (Khan et al., 2022; Zela et al., 2022), COMBO tends to degrade when there are many choices for each combinatorial variable. We conjecture that this is due to too the strong assumption posed by complete graphs. There exists rigidity originating from given and fixed graph structures. Especially when there are many choices for each variable, learning the underlying graph structure appears necessary.

**Research Question 3** *How can we model dependence between different types of variables in kernels? And do we need conditions for kernels other than the positive definiteness?* (Chapter 5)

We propose to modulate the frequency of the ARD diffusion kernel on combinatorial variables in Chapter 4 with the distance of continuous variables. During its development, we observe that frequency modulation requires an additional condition – similarity measure behavior – for the stability of methods using kernels with frequency modulation.

In a wide range of BO and GP regression tasks, we conclude that the similarity measure behavior is an important condition to consider when developing new kernels. Moreover, we hardly observe that dependence modeling with (revised) frequency modulation degrades performance. From this, we expect that the similarity measure behavior guaranteed frequency modulation is not distracted by non-existing dependencies.

Another difficulty in mixed-variable BO is the optimization of the acquisition function. Although we also adopt widely used variable-typewise ascent (Daxberger et al., 2021; Wan et al., 2021), this leaves many nontrivial design choices, e.g. how many updates for the continuous variables are needed for one update for the discrete variables. We may utilize recent advances in mixed integer programming (Gurobi Optimization, Inc., 2012). Also, a data-driven approach demonstrates its competitiveness in the optimization of acquisition functions on combinatorial spaces (K. Swersky, Rubanova, et al., 2020). Combination of both seems a promising way forward.

**Research Question 4** *Can we develop a batch acquisition method applicable to permutation spaces? And how can we maintain the quality of the batch acquisition method as the batch size increases?* (Chapter 6)

We propose a DPP-based batch acquisition, LAW, applicable to permutation spaces. While using weights as a channel to promote properties of batches other than diversity, we also analyze the effect of weights to the regret bound.

We demonstrate that batch acquisition considering both diversity and quality improves the efficiency of BO, more notably when a large batch size is used.

We conclude that DPPs provide a viable formulation for batch acquisition on permutation spaces and the incorporation of the quality of queries can maintain the performance of the batch acquisition for large batch sizes.

Even though the greedy method works well in practice, the approximation guarantee does not hold without the positivity and monotonicity of a submodular function. There could be room for improvement by adopting an approximation method for general submodular functions. A more practical concern is the sequential nature of the greedy approximation. LAW objective optimization can be time-consuming for a large batch size, which limits the batch size used. We see two directions promising to overcome this issue, *1)* efficient GP Thompson sampling on large combinatorial spaces *2)* parallel approximation of submodular maximization with an approximation guarantee.

**Research Question 5** *Can BO be an effective method for combinatorial optimization problems in chip design? And can it be efficient enough to scale to problems of a practical scale?* (Chapter 7)

We demonstrate the competitiveness of BO on the macro placement – a combinatorial optimization problem in chip design. To cope with problems at a practical scale, we propose modifications to LAW in Chapter 6 to speed up the optimization of the batch acquisition objective. The parallel heuristic and the efficient implementation of constraint checking enables $10\times \sim 100\times$ speed up.

We demonstrate that LAW with parallel heuristics is shown to significantly outperforms another black-box function optimizer – simulated annealing – and to perform competitively to methods tuned for specific objectives. From this partial but promising result, we forecast that BO can be a valuable tool to speed up the chip design cycle, hopefully in the near future.

Compared to a RL method (Mirhoseini et al., 2021), BO has an advantage for expensive and realistic objectives. However, BO applied to chip design (Deshwal, Belakaria, J. R. Doppa, and D. H. Kim, 2022;

Grosnit, Malherbe, et al., 2022) optimizes from scratch whereas the RL method demonstrates that it can use knowledge obtained from relevant tasks. BO utilizing knowledge from relevant tasks – warm-start BO (Perrone, Shen, et al., 2019), tranfer BO (Tighineanu et al., 2022) – is a must to handle today's chips of ever increasing complexity. The experiments are mostly a proof of concept. More expensive but realistic objectives taking into account realistic macro placement following place-and-route should be tested to attract chip design engineers in the field.

## 8.2 FUTURE WORKS

In this section, we discuss important research directions not covered in this thesis. We envision that an ideal BO will be able to utilize heterogeneous evaluation data collected from various problems and to optimize new tasks efficiently in a few-shot sense. Therefore, we specifically discuss scalable BO and transfer BO, which is not covered.

SCALABLE GAUSSIAN PROCESSES The development of the surrogate model scalable to a large number of evaluations is approached from two angles. One angle is to improve the quality of the uncertainty estimates from scalable probabilistic models such as deep ensembles, Bayesian neural networks (BNN) (Springenberg et al., 2016), etc. Another angle is to scale up GPs to large data. Recently, there have been notable progress in scalable GPs (Gardner et al., 2018; K. Wang et al., 2019) and scalable GPs show better efficiency – in sample and computation – than BNN on BO tasks (Eriksson, Pearce, et al., 2019). Moreover, the variational method for sparse GPs from (Hensman, Fusi, et al., 2013; Hensman, Matthews, et al., 2015) enables inference using mini-batches in scalable GPs (A. G. Wilson et al., 2016). In spite of such progress, inference and prediction of GPs require the access to full data. And, the uncertainty of the variational method from (Hensman, Fusi, et al., 2013) is shown to exhibit unwanted behavior – namely it is dominated by the observational noise (Jankowiak et al., 2020). We believe that scalable GPs – trainable with mini-batches, producing well-calibrated uncertainty and not accessing full data in inference and prediction – has a potential to broaden the application of BO.

THOMPSON SAMPLING OF GAUSSIAN PROCESSES ON COMBINATORIAL SPACES As long as batch acquisition is not severely degraded compared with (sequential) acquisition, it allows significant speed-up of BO in terms of wall-clock runtime (Azimi, Fern, et al., 2010; Azimi, Jalali, et al., 2012). For scalable BO, efficient batch acquisition is indispensable. While determinantal point process (DPP) based approaches are popular (Kathuria et al., 2016; Nava et al., 2022), Thompson sampling (TS) (Russo et al., 2018) is an attractive alternative (J. M. Hernández-Lobato, Requeima, et al., 2017; Kandasamy, Krishnamurthy, et al., 2018). In addition to its firm theoretical ground (Russo et al., 2018), TS is intrinsically amenable to various parallel settings (J. M. Hernández-Lobato, Requeima, et al., 2017; Kandasamy, Krishnamurthy, et al., 2018). If efficient TS of GP posterior on large combinatorial spaces becomes available, massively parallelizable batch acquisition will immediately follow. Even though there is recent progress on efficient TS on Euclidean spaces (J. Wilson et al., 2020), such efficiency is not easily transported to non-Euclidean spaces since such methods rely on the existence of closed-form eigenfunctions of specific kernels (e.g. RBF) (Shah and Ghahramani, 2015; J. Wilson et al., 2020). We believe that TS is the most natural direction toward massively parallelizable batch acquisition in large combinatorial spaces.

TRANSFERRING KNOWLEDGE FROM OTHER TASKS Utilizing information from relevant tasks is a natural way to improve sample efficiency. BO using knowledge from other tasks has been studied under different names, multi-task BO (K. Swersky, Snoek, et al., 2013), contextual BO (Krause and Ong, 2011), warm-starting BO (Poloczek et al., 2016), etc. In general transfer learning, a parameter is a widely used vessel for carrying extracted information from other tasks. For BO with the GP surrogate model, deep kernel models can be a good compromise between a model with well-calibrated uncertainty and a model with many parameters. In (Wenlin Chen et al., 2022), it is shown that deep model parameters meta-learned across tasks provide an environment where GP kernel hyperparameters can be efficiently tuned for BO. Some methods use other components as a channel to convey knowledge from other tasks. In (Volpp et al., 2019), the acquisition functions are approximated by a parametric model to share knowledge across tasks. In (Perrone, Shen, et al., 2019), a novel approach is taken by using the search space to capture the knowledge across tasks. Despite such advances, we expect that an ideal BO may be able to transfer knowledge across tasks on different search spaces. Moreover, we hope that improved sample efficiency of transfer BO eventually enables the optimization of a new task with very few evaluations, owing to knowledge learned from other relevant tasks.

# BIBLIOGRAPHY

Adya, S.N. and I.L. Markov

2001 "Fixed-outline floorplanning through better local search", in *Proceedings 2001 IEEE International Conference on Computer Design: VLSI in Computers and Processors. ICCD 2001*, pp. 328-334, DOI: `10.1109/ICCD.2001.955047`. (Cit. on pp. 94, 95, 98.)

Adya, Saurabh N and Igor L Markov

2002 "Consistent placement of macro-blocks using floorplanning and standard-cell placement", in *Proceedings of the 2002 International Symposium on Physical design*, pp. 12-17. (Cit. on pp. 93, 98.)

Agnesina, Anthony, Kyungwook Chang, and Sung Kyu Lim

2020 "VLSI Placement Parameter Optimization using Deep Reinforcement Learning", in *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1-9. (Cit. on p. 92.)

Alpert, C.J., D.P. Mehta, and S.S. Sapatnekar

2008 *Handbook of Algorithms for Physical Design Automation*, CRC Press. (Cit. on p. 71.)

Anari, Nima, Shayan Oveis Gharan, and Alireza Rezaei

2016 "Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes", in *Conference on Learning Theory*, PMLR, pp. 103-115.

Andrieu, Christophe and Gareth O Roberts

2009 "The pseudo-marginal approach for efficient Monte Carlo computations", *The Annals of Statistics*, 37, 2, pp. 697-725.

Assael, John-Alexander M, Ziyu Wang, Bobak Shahriari, and Nando de Freitas

2014 "Heteroscedastic treed bayesian optimisation", *arXiv preprint arXiv:1410.7172*. (Cit. on p. 151.)

Audibert, Jean-Yves, Sébastien Bubeck, and Gábor Lugosi

2011 "Minimax policies for combinatorial prediction games", in *Proceedings of the 24th Annual Conference on Learning Theory*, pp. 107-132.

Azimi, Javad, Alan Fern, and Xiaoli Z Fern

2010 "Batch bayesian optimization via simulation matching", in *Advances in Neural Information Processing Systems*, Citeseer, pp. 109-117. (Cit. on pp. 71, 73, 79, 108.)

Azimi, Javad, Ali Jalali, and Xiaoli Z Fern

2012 "Hybrid batch Bayesian optimization", in *Proceedings of the 29th International Conference on Machine Learning*, pp. 315-322. (Cit. on p. 108.)

Bacchus, Fahiem, Matti Jarvisalo, and Ruben Martins

2018 (eds.), *MaxSAT Evaluation 2018. Solver and Benchmark Descriptions*, vol. B-2018-2, University of Helsinki. (Cit. on p. 47.)

Bach, Francis R and Michael I Jordan

2002 "Kernel independent component analysis", *Journal of Machine Learning Research*, 3, Jul, pp. 1-48.

Bachoc, François, Baptiste Broto, Fabrice Gamboa, and Jean-Michel Loubes

2020 "Gaussian field on the symmetric group: Prediction and learning", *Electronic Journal of Statistics*, 14, 1, pp. 503-546. (Cit. on pp. 71, 80.)

Bal, Henri, Dick Epema, Cees de Laat, Rob van Nieuwpoort, John Romein, Frank Seinstra, Cees Snoek, and Harry Wijshoff

2016 "A medium-scale distributed system for computer science research: Infrastructure for the long term", *Computer*, 49, 5, pp. 54-63.

Balandat, Maximilian, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy

2020 "BoTorch: a framework for efficient Monte-Carlo Bayesian optimization", *Advances in neural information processing systems*, 33, pp. 21524-21538.

Baptista, Ricardo and Matthias Poloczek

2018 "Bayesian Optimization of Combinatorial Structures", in *International conference on machine learning*, pp. 462-471. (Cit. on pp. 7, 14, 44, 45, 47, 61, 71, 80, 93, 163, 165.)

Belanger, David, Suhani Vora, Zelda Mariet, Ramya Deshpande, David Dohan, Christof Angermueller, Kevin Murphy, Olivier Chapelle, and Lucy Colwell

2019 *Biological Sequences Design using Batched Bayesian Optimization*. (Cit. on p. 14.)

Berg, Christian, Jens Peter Reus Christensen, and Paul Ressel

1984 *Harmonic analysis on semigroups: theory of positive definite and related functions*, Springer, vol. 100. (Cit. on pp. 178, 179.)

Bergstra, James and Yoshua Bengio
2012 "Random search for hyper-parameter optimization", *Journal of Machine Learning Research*, 13, Feb, pp. 281-305. (Cit. on pp. 24, 48.)

Bergstra, James, Daniel Yamins, and David Cox
2013 "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures", in *International conference on machine learning*, PMLR, pp. 115-123. (Cit. on pp. 14-16, 44, 45.)

Bergstra, James S, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl
2011 "Algorithms for hyper-parameter optimization", in *Advances in neural information processing systems*, pp. 2546-2554. (Cit. on pp. 14-16, 23, 24, 29, 55, 61, 62.)

Bian, Andrew An, Joachim M Buhmann, Andreas Krause, and Sebastian Tschiatschek
2017 "Guarantees for greedy maximization of non-submodular functions with applications", in *International conference on machine learning*, PMLR, pp. 498-507. (Cit. on p. 206.)

Bingham, Eli, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman
2018 "Pyro: Deep universal probabilistic programming.(2018)", *arXiv preprint arXiv:1810.09538*.

Bıyık, Erdem, Kenneth Wang, Nima Anari, and Dorsa Sadigh
2019 "Batch active learning using determinantal point processes", *arXiv preprint arXiv:1906.07975*. (Cit. on p. 80.)

Blank, Julian and Kalyanmoy Deb
2020 "pymoo: Multi-objective optimization in python", *IEEE Access*, 8, pp. 89497-89509. (Cit. on pp. 82, 86.)

Boelrijk, Jim, Bob Pirok, Bernd Ensing, and Patrick Forré
2021 "Bayesian optimization of comprehensive two-dimensional liquid chromatography separations", *Journal of Chromatography A*, 1659, p. 462628. (Cit. on p. 3.)

Bogunovic, Ilija and Andreas Krause
2021 "Misspecified Gaussian process bandit optimization", *Advances in Neural Information Processing Systems*, 34, pp. 3004-3015. (Cit. on p. 16.)

Bogunovic, Ilija, Jonathan Scarlett, and Volkan Cevher
2016 "Time-varying Gaussian process bandit optimization", in *International Conference on Artificial Intelligence and Statistics*, 314-323. (Cit. on p. 16.)

Borodin, Alexei and Eric M Rains

2005 "Eynard–Mehta theorem, Schur process, and their Pfaffian analogs", *Journal of statistical physics*, 121, 3, pp. 291-317. (Cit. on pp. 72, 73.)

Boros, Endre and Peter L Hammer

2002 "Pseudo-boolean optimization", *Discrete applied mathematics*, 123, 1-3, pp. 155-225.

Bottou, Léon

2010 "Large-scale machine learning with stochastic gradient descent", in *Proceedings of COMPSTAT'2010*, Springer, pp. 177-186.

Box, George EP and Kenneth B Wilson

1992 "On the experimental attainment of optimum conditions", in *Breakthroughs in statistics*, Springer, pp. 270-310. (Cit. on p. 18.)

Bratley, Paul and Bennett L Fox

1988 "Algorithm 659: Implementing Sobol's quasirandom sequence generator", *ACM Transactions on Mathematical Software (TOMS)*, 14, 1, pp. 88-100. (Cit. on p. 28.)

Brochu, Eric, Vlad M Cora, and Nando de Freitas

2010 *A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning*. (Cit. on pp. 4, 15, 37, 95.)

Buchbinder, Niv, Moran Feldman, Joseph Naor, and Roy Schwartz

2014 "Submodular maximization with cardinality constraints", in *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, SIAM, pp. 1433-1452. (Cit. on p. 8.)

Carlier, Jacques

1978 "Ordonnancements a contraintes disjonctives", *RAIRO-Operations Research*, 12, 4, pp. 333-350. (Cit. on p. 207.)

Carpenter, Bob, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell

2017 "Stan: A probabilistic programming language", *Journal of statistical software*, 76, 1.

Carvalho, Carlos M, Nicholas G Polson, and James G Scott

2009 "Handling sparsity via the horseshoe", in *International Conference on Artificial Intelligence and Statistics*, pp. 73-80. (Cit. on pp. 38, 41, 43.)

2010 "The horseshoe estimator for sparse signals", *Biometrika*, 97, 2, pp. 465-480. (Cit. on pp. 38, 160.)

Char, Ian, Youngseog Chung, Willie Neiswanger, Kirthevasan Kandasamy, Andrew Oakleigh Nelson, Mark Boyer, Egemen Kolemen, and Jeff Schneider

2019 "Offline contextual bayesian optimization", in *Advances in neural information processing systems*, pp. 4629-4640. (Cit. on pp. 19, 94.)

Chen, Bo, Rui Castro, and Andreas Krause

2012 "Joint optimization and variable selection of high-dimensional Gaussian processes", *arXiv preprint arXiv:1206.6396*. (Cit. on p. 33.)

Chen, Eunice Yuh-Jie, Arthur Choi, and Adnan Darwiche

2015 "Learning Bayesian networks with non-decomposable scores", in *International Workshop on Graph Structures for Knowledge Representation and Reasoning*, Springer, pp. 50-71.

Chen, Hongming, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke

2018 "The rise of deep learning in drug discovery", *Drug discovery today*, 23, 6, pp. 1241-1250. (Cit. on p. 18.)

Chen, Lixing, Jie Xu, and Zhuo Lu

2018 "Contextual combinatorial multi-armed bandits with volatile arms and submodular reward", in *Advances in neural information processing systems*, pp. 3247-3256.

Chen, Tianqi and Carlos Guestrin

2016 "Xgboost: A scalable tree boosting system", in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785-794. (Cit. on pp. 62, 64.)

Chen, Wei, Yajun Wang, and Yang Yuan

2013 "Combinatorial multi-armed bandit: General framework and applications", in *International conference on machine learning*, pp. 151-159.

Chen, Wenlin, Austin Tripp, and José Miguel Hernández-Lobato

2022 "Meta-learning Feature Representations for Adaptive Gaussian Processes via Implicit Differentiation", *arXiv preprint arXiv:2205.02708*. (Cit. on p. 109.)

Chen, Yuh-Jie

2016 *Learning Bayesian Network Structures with Non-Decomposable Scores*, PhD thesis, UCLA.

Chen, Yutian, Aja Huang, Ziyu Wang, Ioannis Antonoglou, Julian Schrittwieser, David Silver, and Nando de Freitas

2018 "Bayesian optimization in alphago", *arXiv preprint arXiv:1812.06855*. (Cit. on pp. 3, 4, 16, 53.)

Chen, Yutian, Xingyou Song, Chansoo Lee, Zi Wang, Qiuyi Zhang, David Dohan, Kazuya Kawakami, Greg Kochanski, Arnaud Doucet, Marc'aurelio Ranzato, et al.

2022 "Towards Learning Universal Hyperparameter Optimizers with Transformers", *arXiv preprint arXiv:2205.13320*.

Cheng, Chung-Kuan, Andrew B. Kahng, Ilgweon Kang, and Lutong Wang

2019 "RePlAce: Advancing Solution Quality and Routability Validation in Global Placement", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 38, 9, pp. 1717-1730, DOI: `10.1109/TCAD.2018.2859220`. (Cit. on pp. 91, 99.)

Cheng, Ruoyu and Junchi Yan

2021 "On Joint Learning for Solving Placement and Routing in Chip Design", *Advances in Neural Information Processing Systems*, 34, pp. 16508-16519. (Cit. on pp. 17, 18.)

Chergui, Akram, Khaled Hadj-Hamou, and Frédéric Vignat

2018 "Production scheduling and nesting in additive manufacturing", *Computers & Industrial Engineering*, 126, pp. 292-301. (Cit. on p. 71.)

Chickering, David Maxwell

2002 "Optimal structure identification with greedy search", *Journal of machine learning research*, 3, Nov, pp. 507-554.

Cho, Youngmin and Lawrence K Saul

2009 "Kernel methods for deep learning", in *Advances in neural information processing systems*, pp. 342-350.

Chowdhury, Sayak Ray and Aditya Gopalan

2017 "On kernelized multi-armed bandits", in *International Conference on Machine Learning*, PMLR, pp. 844-853. (Cit. on pp. 16, 17, 100.)

Christofides, Nicos and Enrique Benavent

1989 "An exact algorithm for the quadratic assignment problem on a tree", *Operations Research*, 37, 5, pp. 760-768. (Cit. on p. 207.)

Chung, Fan RK

1997 *Spectral graph theory*, American Mathematical Soc., vol. 92. (Cit. on p. 39.)

Contal, Emile, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis

2013 "Parallel Gaussian process optimization with upper confidence bound and pure exploration", in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 225-240. (Cit. on pp. 17, 77, 80, 82, 189, 192-198.)

Dadkhahi, Hamid, Karthikeyan Shanmugam, Jesus Rios, Payel Das, Samuel C Hoffman, Troy David Loeffler, and Subramanian Sankaranarayanan

2020 "Combinatorial Black-Box Optimization with Expert Advice", in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1918-1927. (Cit. on pp. 71, 80.)

Davis, Lawrence

1991 *Handbook of genetic algorithms*. (Cit. on p. 37.)

Daxberger, Erik, Anastasia Makarova, Matteo Turchetta, and Andreas Krause

2021 "Mixed-variable Bayesian optimization", in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 2633-2639. (Cit. on pp. 14, 53, 61, 62, 106, 181.)

De Freitas, Nando, Alex J Smola, and Masrour Zoghi

2012 "Exponential regret bounds for Gaussian process bandits with deterministic observations", in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 955-962. (Cit. on p. 17.)

Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan

2002 "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE transactions on evolutionary computation*, 6, 2, pp. 182-197. (Cit. on p. 82.)

Desautels, Thomas, Andreas Krause, and Joel W Burdick

2014 "Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization", *Journal of Machine Learning Research*, 15, pp. 3873-3923. (Cit. on pp. 17, 77, 80, 82, 189, 191, 196-198.)

Deshmukh, Aniket Anand, Urun Dogan, and Clay Scott

2017 "Multi-task learning for contextual bandits", in *Advances in neural information processing systems*, pp. 4848-4856.

Deshwal, Aryan, Syrine Belakaria, and Janardhan Rao Doppa

2021 "Mercer features for efficient combinatorial Bayesian optimization", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 7210-7218. (Cit. on p. 93.)

Deshwal, Aryan, Syrine Belakaria, Janardhan Rao Doppa, and Alan Fern

2020 "Optimizing discrete spaces via expensive evaluations: A learning to search framework", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 3773-3780. (Cit. on pp. 71, 80.)

Deshwal, Aryan, Syrine Belakaria, Janardhan Rao Doppa, and Dae Hyun Kim

2022 "Bayesian optimization over permutation spaces", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 6515-6523. (Cit. on pp. 17, 18, 93, 100, 107.)

Deshwal, Aryan and Jana Doppa

2021 "Combining latent space and structured kernels for bayesian optimization over combinatorial spaces", *Advances in Neural Information Processing Systems*, 34, pp. 8185-8200. (Cit. on p. 93.)

Donald, R Jones

1998 "Efficient global optimization of expensive black-box function", *J. Global Optim.*, 13, pp. 455-492. (Cit. on pp. 62, 181.)

Dong, Kun, David Eriksson, Hannes Nickisch, David Bindel, and Andrew G Wilson

2017 "Scalable log determinants for Gaussian process kernel learning", *Advances in Neural Information Processing Systems*, 30.

Dong, Xuanyi, Lu Liu, Katarzyna Musial, and Bogdan Gabrys

2021 "NATS-Bench: Benchmarking nas algorithms for architecture topology and size", *IEEE Transactions on Pattern Analysis and Machine Intelligence*. (Cit. on pp. 16, 18, 54, 62, 67.)

Dong, Xuanyi, Mingxing Tan, Adams Wei Yu, Daiyi Peng, Bogdan Gabrys, and Quoc V Le

2020 "AutoHAS: Efficient hyperparameter and architecture search", *arXiv preprint arXiv:2006.03656*. (Cit. on p. 62.)

Dong, Xuanyi and Yi Yang

2019 "One-shot neural architecture search via self-evaluated template network", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3681-3690. (Cit. on p. 67.)

Drton, Mathias and Marloes H Maathuis

2017 “Structure learning in graphical modeling”, *Annual Review of Statistics and Its Application*, 4, pp. 365-393. (Cit. on p. 85.)

Duvenaud, David K, Hannes Nickisch, and Carl E Rasmussen

2011 “Additive gaussian processes”, in *Advances in neural information processing systems*, pp. 226-234.

Eggensp-erger, Katharina, Matthias Feurer, Frank Hutter, James Bergstra, Jasper Snoek, Holger Hoos, and Kevin Leyton-Brown

2013 “Towards an empirical foundation for assessing bayesian optimization of hyperparameters”, in *NIPS workshop on Bayesian Optimization in Theory and Practice*, vol. 10. (Cit. on p. 29.)

Elsken, Thomas, Jan Hendrik Metzen, and Frank Hutter

2019 “Neural architecture search: A survey”, *The Journal of Machine Learning Research*, 20, 1, pp. 1997-2017. (Cit. on pp. 16, 17, 38.)

Eriksson, David and Martin Jankowiak

2021 “High-dimensional Bayesian optimization with sparse axis-aligned subspaces”, in *Uncertainty in Artificial Intelligence* , PMLR, pp. 493-503.

Eriksson, David, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek

2019 “Scalable Global Optimization via Local Bayesian Optimization”, in *Advances in Neural Information Processing Systems*, vol. 32. (Cit. on pp. 14, 17, 105, 108.)

Eschermann, Bernhard and Hans-Joachim Wunderlich

1990 “Optimized synthesis of self-testable finite state machines”. (Cit. on p. 207.)

Falkner, Stefan, Aaron Klein, and Frank Hutter

2018 “BOHB: Robust and efficient hyperparameter optimization at scale”, in *International Conference on Machine Learning*, PMLR, pp. 1437-1446. (Cit. on pp. 16, 54, 62, 67.)

Feige, Uriel, Vahab S Mirrokni, and Jan Vondrák

2011 “Maximizing non-monotone submodular functions”, *SIAM Journal on Computing*, 40, 4, pp. 1133-1153. (Cit. on p. 206.)

Fiducioso, Marcello, Sebastian Curi, Benedikt Schumacher, Markus Gwerder, and Andreas Krause

2019 “Safe contextual Bayesian optimization for sustainable room temperature PID control tuning”, in *IJCAI*, AAAI Press , pp. 5850-5856. (Cit. on p. 61.)

Folland, Gerald B

1999 *Real analysis: modern techniques and their applications*, Wiley. (Cit. on p. 180.)

Fortuin, Vincent, Gideon Dresdner, Heiko Strathmann, and Gunnar Rätsch

2021 "Sparse Gaussian processes on discrete domains", *IEEE Access*, 9, pp. 76750-76758.

Fox, Emily B and David B Dunson

2015 "Bayesian nonparametric covariance regression", *Journal of Machine Learning Research*, 16, 1, pp. 2501-2542.

Frazier, Peter I

2018 "A tutorial on Bayesian optimization", *arXiv preprint arXiv:1807.02811*. (Cit. on pp. 37, 73, 95.)

Frazier, Peter I and Jialei Wang

2016 "Bayesian optimization for materials design", in *Information science for materials discovery and design*, Springer, pp. 45-75. (Cit. on p. 19.)

Freitas, Alex A

2009 "A review of evolutionary algorithms for data mining", in *Data Mining and Knowledge Discovery Handbook*, Springer , pp. 371-400. (Cit. on p. 37.)

Fujishige, Satoru

2005 *Submodular functions and optimization*, Elsevier. (Cit. on p. 205.)

Fukumizu, Kenji

2010 "Kernel method: Data analysis with positive definite kernels", *Graduate University of Advanced Studies*. (Cit. on p. 180.)

Funke, J., S. Hougardy, and J. Schneider

2016 "An exact algorithm for wirelength optimal placements in VLSI design", *Integration*, 52, pp. 355-366. (Cit. on pp. 100-102.)

Gardner, Jacob, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson

2018 "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration", in *Advances in neural information processing systems*, pp. 7576-7586. (Cit. on p. 108.)

Garey, Michael R, David S Johnson, and Ravi Sethi

1976 "The complexity of flowshop and jobshop scheduling", *Mathematics of operations research*, 1, 2, pp. 117-129. (Cit. on p. 71.)

Garnett, Roman, Michael A Osborne, and Stephen J Roberts
2010 "Bayesian optimization for sensor set selection", in *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, pp. 209-219. (Cit. on pp. 161, 181.)

Garrido-Merchán, Eduardo C and Daniel Hernández-Lobato
2020 "Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes", *Neurocomputing*, 380, pp. 20-35. (Cit. on pp. 5, 37, 44.)

Ginsbourger, David, Rodolphe Le Riche, and Laurent Carraro
2008 *A multi-points criterion for deterministic parallel global optimization based on Gaussian processes*. (Cit. on p. 82.)

Gómez-Bombarelli, Rafael, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik
2018 "Automatic chemical design using a data-driven continuous representation of molecules", *ACS central science*, 4, 2, pp. 268-276. (Cit. on pp. 17, 18.)

Gong, Chengyue, Jian Peng, and Qiang Liu
2019 "Quantile stein variational gradient descent for batch bayesian optimization", in *International Conference on Machine Learning*, PMLR, pp. 2347-2356. (Cit. on pp. 71, 74, 79, 95.)

González, Javier, Zhenwen Dai, Philipp Hennig, and Neil Lawrence
2016 "Batch Bayesian optimization via local penalization", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 648-657. (Cit. on pp. 25, 71, 73, 74, 79, 84, 95, 206, 209.)

Gopakumar, Shivapratap, Sunil Gupta, Santu Rana, Vu Nguyen, and Svetha Venkatesh
2018 "Algorithmic assurance: An active approach to algorithmic testing using bayesian optimisation", in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 5470-5478. (Cit. on pp. 61, 63.)

GPyOpt, authors
2016 *GPyOpt: A Bayesian Optimization framework in Python*, `http://github.com/SheffieldML/GPyOpt`. (Cit. on p. 62.)

Greenhill, Stewart, Santu Rana, Sunil Gupta, Pratibha Vellanki, and Svetha Venkatesh
2020 "Bayesian optimization for adaptive experimental design: a review", *IEEE access*, 8, pp. 13937-13948. (Cit. on p. 18.)

Grenander, Ulf and Gabor Szegö

1958 *Toeplitz forms and their applications*, Univ of California Press. (Cit. on p. 202.)

Griffiths, Ryan-Rhys and José Miguel Hernández-Lobato

2020 "Constrained Bayesian optimization for automatic chemical design using variational autoencoders", *Chemical science*, 11, 2, pp. 577-586. (Cit. on p. 19.)

Griffiths, Valeriya, James P Scanlan, Murat H Eres, Antonio Martinez-Sykora, and Phani Chinchapatnam

2019 "Cost-driven build orientation and bin packing of parts in Selective Laser Melting (SLM)", *European Journal of Operational Research*, 273, 1, pp. 334-352. (Cit. on p. 71.)

Grosnit, Antoine, Cedric Malherbe, Rasul Tutunov, Xingchen Wan, Jun Wang, and Haitham Bou Ammar

2022 "BOiLS: bayesian optimisation for logic synthesis", in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, pp. 1193-1196. (Cit. on p. 107.)

Grosnit, Antoine, Rasul Tutunov, Alexandre Max Maraval, Ryan-Rhys Griffiths, Alexander I Cowen-Rivers, Lin Yang, Lin Zhu, Wenlong Lyu, Zhitang Chen, Jun Wang, Jan Peters, and Haitham Bou Ammar

2021 "High-dimensional Bayesian optimisation with variational autoencoders and deep metric learning", *arXiv preprint arXiv:2106.03609*.

Grünewälder, Steffen, Jean–Yves Audibert, Manfred Opper, and John Shawe–Taylor

2010 "Regret bounds for Gaussian process bandit problems", in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, pp. 273-280. (Cit. on p. 16.)

Grünwald, Peter and Teemu Roos

2019 "Minimum description length revisited", *International journal of mathematics for industry*, 11, 01, p. 1930001. (Cit. on p. 207.)

Grünwald, Peter D and Abhijit Grunwald

2007 *The minimum description length principle*, MIT press. (Cit. on pp. 86, 207.)

Gupta, Sunil, Santu Rana, Svetha Venkatesh, et al.

2022 "Regret Bounds for Expected Improvement Algorithms in Gaussian Process Bandit Optimization", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 8715-8737. (Cit. on p. 17.)

Gurobi Optimization, Inc.

2012 *Gurobi optimizer reference manual*, `http://www.gurobi.com`. (Cit. on p. 106.)

Gutin, Gregory and Abraham P Punnen

2006 *The traveling salesman problem and its variations*, Springer Science & Business Media, vol. 12. (Cit. on p. 71.)

Gutmann, Michael U and Jukka Corander

2016 "Bayesian optimization for likelihood-free inference of simulator-based statistical models", *Journal of Machine Learning Research*.

Hammack, Richard, Wilfried Imrich, and Sandi Klavžar

2011 *Handbook of product graphs*, CRC press. (Cit. on pp. 7, 39, 42, 157.)

Hammer, Peter L and Sergiu Rudeanu

2012 *Boolean methods in operations research and related areas*, Springer Science & Business Media, vol. 7.

Han, Insu, Prabhanjan Kambadur, Kyoungsoo Park, and Jinwoo Shin

2017 "Faster greedy MAP inference for determinantal point processes", in *International Conference on Machine Learning*, PMLR, pp. 1384-1393.

Hansen, Nikolaus and Andreas Ostermeier

1996 "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation", in *Proceedings of IEEE international conference on evolutionary computation*, IEEE, pp. 312-317. (Cit. on p. 5.)

Hansen, Pierre and Brigitte Jaumard

1990 "Algorithms for the maximum satisfiability problem", *Computing*, 44, 4, pp. 279-303. (Cit. on p. 38.)

Hase, Florian, Loïc M Roch, Christoph Kreisbeck, and Alán Aspuru-Guzik

2018 "Phoenics: a Bayesian optimizer for chemistry", *ACS central science*, 4, 9, pp. 1134-1145. (Cit. on p. 3.)

Haussler, David

1999 *Convolution kernels on discrete structures*, tech. rep., Technical report, Department of Computer Science, University of California ... (Cit. on pp. 37, 39, 61.)

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun

2016a "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778. (Cit. on p. 33.)

2016b "Identity mappings in deep residual networks", in *European conference on computer vision*, Springer, pp. 630-645.

Heller, Jack

1960 "Some numerical experiments for an M$\times$ J flow shop and its decision-theoretical aspects", *Operations Research*, 8, 2, pp. 178-184. (Cit. on p. 207.)

Hennig, Philipp and Christian J Schuler

2012 "Entropy search for information-efficient global optimization", *Journal of Machine Learning Research*, 13, Jun, pp. 1809-1837. (Cit. on pp. 13, 23, 79.)

Hensman, James, Nicolò Fusi, and Neil D. Lawrence

2013 "Gaussian Processes for Big Data", in *Uncertainty in Artificial Intelligence*, AUAI Press, vol. 29. (Cit. on p. 108.)

Hensman, James, Alexander Matthews, and Zoubin Ghahramani

2015 "Scalable variational Gaussian process classification", in *International Conference on Artificial Intelligence and Statistics* , PMLR, pp. 351-360. (Cit. on p. 108.)

Hernández-Lobato, José Miguel, Matthew W Hoffman, and Zoubin Ghahramani

2014 "Predictive entropy search for efficient global optimization of black-box functions", in *Advances in neural information processing systems*, pp. 918-926. (Cit. on pp. 13, 23, 44, 80.)

Hernández-Lobato, José Miguel, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik

2017 "Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space", in *International conference on machine learning*, PMLR, pp. 1470-1479. (Cit. on p. 109.)

Hoffman, Matthew D and Andrew Gelman

2014 "The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research*, 15, 1, pp. 1593-1623.

Hu, Yingjie, JianQiang Hu, Yifan Xu, Fengchun Wang, and Rong Zeng Cao

2010 "Contamination control in food supply chain", in *Simulation Conference (WSC), Proceedings of the 2010 Winter*, IEEE, pp. 2678-2681. (Cit. on pp. 45, 164.)

Huang, Gao, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger

2016 "Deep networks with stochastic depth", in *European conference on computer vision*, Springer, pp. 646-661. (Cit. on pp. 22, 33-35.)

Hutter, Frank, Holger H Hoos, and Kevin Leyton-Brown

2011 "Sequential model-based optimization for general algorithm configuration", in *International conference on learning and intelligent optimization*, Springer, pp. 507-523. (Cit. on pp. 14-16, 23, 24, 29, 31, 44, 45, 55, 61, 62.)

Ishwaran, Hemant and J Sunil Rao

2005 "Spike and slab variable selection: frequentist and Bayesian strategies", *The Annals of Statistics*, 33, 2, pp. 730-773.

Jacot, Arthur, Franck Gabriel, and Clément Hongler

2018 "Neural tangent kernel: Convergence and generalization in neural networks", in *Advances in neural information processing systems*, pp. 8571-8580.

Jalas, Sören, Manuel Kirchen, Philipp Messner, Paul Winkler, Lars Hübner, Julian Dirkwinkel, Matthias Schnepp, Remi Lehe, and Andreas R Maier

2021 "Bayesian optimization of a laser-plasma accelerator", *Physical review letters*, 126, 10, p. 104801. (Cit. on p. 19.)

Jankowiak, Martin, Geoff Pleiss, and Jacob Gardner

2020 "Parametric gaussian process regressors", in *International Conference on Machine Learning*, PMLR, pp. 4702-4712. (Cit. on p. 108.)

Janz, David, David Burt, and Javier González

2020 "Bandit optimisation of functions in the Matérn kernel RKHS", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 2486-2495. (Cit. on p. 16.)

Jayasumana, Sadeep, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi

2014 "Optimizing over radial kernels on compact manifolds", in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3802-3809. (Cit. on pp. 26, 152.)

Jiao, Yunlong and Jean-Philippe Vert

2015 "The Kendall and Mallows kernels for permutations", in *International Conference on Machine Learning*, PMLR, pp. 1935-1944. (Cit. on p. 78.)

Jones, Donald R

2001 "A taxonomy of global optimization methods based on response surfaces", *Journal of global optimization*, 21, 4, pp. 345-383.

Jones, Donald R, Matthias Schonlau, and William J Welch

1998 "Efficient global optimization of expensive black-box functions", *Journal of Global optimization*, 13, 4, pp. 455-492. (Cit. on pp. 13, 37, 44, 46, 81.)

Kahng, A.B., J. Lienig, I.L. Markov, and J. Hu

2011 *VLSI Physical Design: From Graph Partitioning to Timing Closure*, Springer Netherlands. (Cit. on pp. 8, 92.)

Kahng, Andrew B

2018 "Machine learning applications in physical design: Recent results and directions", in *Proceedings of the 2018 International Symposium on Physical Design*, pp. 68-73. (Cit. on p. 92.)

Kandasamy, Kirthevasan, Gautam Dasarathy, Junier Oliva, Jeff Schneider, and Barnabas Poczos

2019 "Multi-fidelity gaussian process bandit optimisation", *Journal of Artificial Intelligence Research*, 66, pp. 151-196.

Kandasamy, Kirthevasan, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos

2016 "Gaussian process bandit optimisation with multi-fidelity evaluations", in *Advances in Neural Information Processing Systems*, pp. 992-1000. (Cit. on pp. 37, 39.)

Kandasamy, Kirthevasan, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos

2018 "Parallelised Bayesian optimisation via Thompson sampling", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 133-142. (Cit. on pp. 17, 77, 79, 80, 109, 189-191, 197, 198.)

Kandasamy, Kirthevasan, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric P Xing

2018 "Neural architecture search with bayesian optimisation and optimal transport", in *Advances in Neural Information Processing Systems*, pp. 2016-2025. (Cit. on pp. 16, 18, 49, 50, 53.)

Kandasamy, Kirthevasan, Jeff Schneider, and Barnabás Póczos

2015 "High dimensional Bayesian optimisation and bandits via additive models", in *International conference on machine learning*, pp. 295-304. (Cit. on pp. 24, 29-31.)

Kathuria, Tarun, Amit Deshpande, and Pushmeet Kohli

2016 "Batched gaussian process bandit optimization via determinantal point processes", *Advances in Neural Information Processing Systems*, 29. (Cit. on pp. 72, 74, 77, 79, 80, 82, 109, 192-194, 196.)

Kauffman, Stuart and Simon Levin

1987 "Towards a general theory of adaptive walks on rugged landscapes", *Journal of theoretical Biology*, 128, 1, pp. 11-45.

Kauffman, Stuart A and Edward D Weinberger

1989 "The NK model of rugged fitness landscapes and its application to maturation of the immune response", *Journal of theoretical biology*, 141, 2, pp. 211-245.

Khan, Asif, Alexander I Cowen-Rivers, Derrick-Goh-Xin Deik, Antoine Grosnit, Kamil Dreczkowski, Philippe A Robert, Victor Greiff, Rasul Tutunov, Dany Bou-Ammar, Jun Wang, et al.

2022 "AntBO: Towards Real-World Automated Antibody Design with Combinatorial Bayesian Optimisation", *arXiv preprint arXiv:2201.12570*. (Cit. on p. 106.)

Khanna, Sanjeev, Rajeev Motwani, Madhu Sudan, and Umesh Vazirani

1998 "On syntactic versus computational views of approximability", *SIAM Journal on Computing*, 28, 1, pp. 164-191.

Kim, Jungtaek and Seungjin Choi

2020 "On local optimizers of acquisition functions in bayesian optimization", in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 675-690.

Kingma, Diederik P and Jimmy Ba

2015 "Adam: A Method for Stochastic Optimization", in *International Conference on Learning Representations*. (Cit. on pp. 28, 30, 33, 35, 81, 162.)

Koller, Daphne and Nir Friedman

2009 *Probabilistic graphical models: principles and techniques* , MIT press. (Cit. on p. 85.)

Kondor, Risi Imre and John Lafferty

2002 "Diffusion kernels on graphs and other discrete structures", in *International conference on machine learning*. (Cit. on pp. 7, 37, 39, 41, 55, 59, 61, 178.)

Kontkanen, Petri and Petri Myllymäki

2007 "A linear-time algorithm for computing the multinomial stochastic complexity", *Information Processing Letters*, 103, 6 , pp. 227-233. (Cit. on p. 208.)

Koopmans, Tjalling C and Martin Beckmann

1957 "Assignment problems and the location of economic activities", *Econometrica: journal of the Econometric Society*, pp. 53-76. (Cit. on p. 206.)

Korovina, Ksenia, Sailun Xu, Kirthevasan Kandasamy, Willie Neiswanger, Barnabas Poczos, Jeff Schneider, and Eric Xing

2020 "Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 3393-3403. (Cit. on pp. 17, 18, 53.)

Kozminski, K.

1991 "Benchmarks for layout synthesis - evolution and current status", in *28th ACM/IEEE Design Automation Conference*, pp. 265-270. (Cit. on p. 101.)

Krause, Andreas and Cheng S Ong

2011 "Contextual gaussian process bandit optimization", in *Advances in neural information processing systems*, pp. 2447-2455. (Cit. on pp. 16, 61, 109.)

Krause, Andreas, Ajit Singh, and Carlos Guestrin

2008 "Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies." *Journal of Machine Learning Research*, 9, 2. (Cit. on p. 192.)

Krizhevsky, Alex and Geoffrey Hinton

2009 *Learning multiple layers of features from tiny images*. (Cit. on pp. 34, 49, 162.)

Kulesza, Alex and Ben Taskar

2010 "Structured determinantal point processes", *Advances in neural information processing systems*, 23, pp. 1171-1179. (Cit. on pp. 80, 96.)

2012 "Determinantal Point Processes for Machine Learning", *Machine Learning*, 5, 2-3, pp. 123-286. (Cit. on pp. 8, 71-73, 75, 80, 96.)

Kushner, Harold J

1964 "A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise", *Journal of Basic Engineering*, 86, 1, pp. 97-106. (Cit. on pp. 13, 23.)

Kveton, Branislav, Zheng Wen, Azin Ashkan, and Csaba Szepesvari

2015 "Tight regret bounds for stochastic combinatorial semi-bandits", in *International Conference on Artificial Intelligence and Statistics*, pp. 535-543.

Laguna, Manuel and Rafael Marti

2005 "Experimental testing of advanced scatter search designs for global optimization of multimodal functions", *Journal of Global Optimization*, 33, 2, pp. 235-255. (Cit. on pp. 29, 153.)

Lam, Rémi, Matthias Poloczek, Peter Frazier, and Karen E Willcox

2018 "Advances in bayesian optimization with applications in aerospace engineering", in *2018 AIAA Non-Deterministic Approaches Conference*, p. 1656.

Lan, Shiwei

2019 "Learning Temporal Evolution of Spatial Dependence with Generalized Spatiotemporal Gaussian Process Models", *arXiv preprint arXiv:1901.04030*.

Lan, Shiwei, Andrew Holbrook, Gabriel A Elias, Norbert J Fortin, Hernando Ombao, and Babak Shahbaba

2020 "Flexible bayesian dynamic modeling of correlation and covariance matrices", *Bayesian analysis*, 15, 4, p. 1199.

Lattimore, Tor and Csaba Szepesvári

2020 *Bandit algorithms*, Cambridge University Press. (Cit. on p. 190.)

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner

1998 "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, 86, 11, pp. 2278-2324.

Lee, Eric, David Eriksson, David Bindel, Bolong Cheng, and Mike Mccourt

2020 "Efficient Rollout Strategies for Bayesian Optimization", in *Conference on Uncertainty in Artificial Intelligence*, PMLR , pp. 260-269. (Cit. on p. 14.)

Li, Lisha, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar

2017 "Hyperband: A novel bandit-based approach to hyperparameter optimization", *The Journal of Machine Learning Research*, 18, 1, pp. 6765-6816. (Cit. on p. 67.)

Li, Rui, Michael TM Emmerich, Jeroen Eggermont, Ernst GP Bovenkamp, Thomas Bäck, Jouke Dijkstra, and Johan HC Reiber

2006 "Mixed-integer NK landscapes", in *Parallel Problem Solving from Nature-PPSN IX*, Springer, pp. 42-51.

Li, Shuai, Baoxiang Wang, Shengyu Zhang, and Wei Chen

2016 "Contextual Combinatorial Cascading Bandits.", in *International conference on machine learning*, vol. 16, pp. 1245-1253. (Cit. on p. 61.)

Linting, Mariëlle and Anita van der Kooij

2012 "Nonlinear principal components analysis with CATPCA: a tutorial", *Journal of personality assessment*, 94, 1, pp. 12-25.

Liu, Hanxiao, Karen Simonyan, and Yiming Yang

2018 "DARTS: Differentiable Architecture Search", in *International Conference on Learning Representations*. (Cit. on pp. 16, 61, 67.)

Liu, Wei and Alberto Nannarelli

2008 "Net balanced floorplanning based on elastic energy model", in *2008 NORCHIP*, IEEE, pp. 258-263. (Cit. on pp. 100-102.)

Lovász, László

1983 "Submodular functions and convexity", in *Mathematical programming the state of the art*, Springer, pp. 235-257. (Cit. on p. 205.)

Lu, Jingwei, Hao Zhuang, Pengwen Chen, Hongliang Chang, Chin-Chih Chang, Yiu-Chung Wong, Lu Sha, Dennis Huang, Yufeng Luo, Chin-Chi Teng, and Chung-Kuan Cheng

2015 "ePlace-MS: Electrostatics-Based Placement for Mixed-Size Circuits", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34, 5, pp. 685-698, DOI: 10.1109/TCAD.2015.2391263. (Cit. on p. 99.)

Lu, Xiaoyu, Javier Gonzalez, Zhenwen Dai, and Neil D Lawrence

2018 "Structured variationally auto-encoded optimization", in *International conference on machine learning*, PMLR, pp. 3267-3275. (Cit. on p. 19.)

Lyu, Wenlong, Fan Yang, Changhao Yan, Dian Zhou, and Xuan Zeng

2018 "Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design", in *International conference on machine learning*, PMLR, pp. 3306-3314. (Cit. on pp. 3, 8, 18, 79, 82, 95.)

MacKay, David JC

1994 "Bayesian nonlinear modeling for the prediction competition", *ASHRAE transactions*, 100, 2, pp. 1053-1062. (Cit. on p. 42.)

Malkomes, Gustavo, Charles Schaff, and Roman Garnett

2016 "Bayesian optimization for automated model selection", in *Advances in Neural Information Processing Systems*, pp. 2900-2908. (Cit. on p. 161.)

Marcus, Marvin and Henryk Minc
1992 *A survey of matrix theory and matrix inequalities*, Courier Corporation, vol. 14.

Markov, Igor L., Jin Hu, and Myung-Chul Kim
2015 "Progress and Challenges in VLSI Placement Research", *Proceedings of the IEEE*, 103, 11, pp. 1985-2003, DOI: `10.1109/JPROC.2015.2478963`. (Cit. on p. 98.)

Mirhoseini, Azalia, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, et al.
2021 "A graph placement methodology for fast chip design", *Nature*, 594, 7862, pp. 207-212. (Cit. on pp. 3, 17, 18, 92, 100, 107.)

Mitchell, Melanie
1998 *An introduction to genetic algorithms*, MIT press. (Cit. on pp. 5, 11.)

Močkus, Jonas
1975 "On Bayesian methods for seeking the extremum", in *Optimization Techniques IFIP Technical Conference*, Springer, pp. 400-404. (Cit. on pp. 23, 37.)

Mononen, Tommi and Petri Myllymäki
2007 "Fast NML computation for naive Bayes models", in *International Conference on Discovery Science*, Springer, pp. 151-160. (Cit. on p. 208.)

Montgomery, Douglas C
2017 *Design and analysis of experiments*, John wiley & sons. (Cit. on p. 18.)

Moriconi, Riccardo, Marc Peter Deisenroth, and KS Sesh Kumar
2020 "High-dimensional Bayesian optimization using low-dimensional feature spaces", *Machine Learning*, 109, 9, pp. 1925-1943.

Murata, Hiroshi, Kunihiro Fujiyoshi, Shigetoshi Nakatake, and Yoji Kajitani
1996 "VLSI module placement based on rectangle-packing by the sequence-pair", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15, 12, pp. 1518-1524. (Cit. on pp. 91, 93, 94, 96.)

Murray, Iain and Ryan P Adams
2010 "Slice sampling covariance hyperparameters of latent Gaussian models", in *Advances in neural information processing systems*, pp. 1732-1740. (Cit. on pp. 28, 43, 158.)

Murray, Iain and Matthew Graham

2016 "Pseudo-marginal slice sampling", in *International Conference on Artificial Intelligence and Statistics*, pp. 911-919.

Nava, Elvis, Mojmir Mutny, and Andreas Krause

2022 "Diversified Sampling for Batched Bayesian Optimization with Determinantal Point Processes", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 7031-7054. (Cit. on p. 109.)

Neal, Radford M

1995 *Bayesian learning for neural networks*, PhD thesis, University of Toronto. (Cit. on p. 42.)

2003 "Slice sampling", *Annals of statistics*, pp. 705-741. (Cit. on pp. 28, 43, 158-160.)

Nemhauser, George L, Laurence A Wolsey, and Marshall L Fisher

1978 "An analysis of approximations for maximizing submodular set functions—I", *Mathematical programming*, 14, 1, pp. 265-294. (Cit. on pp. 8, 73, 205.)

Nguyen, Dang, Sunil Gupta, Santu Rana, Alistair Shilton, and Svetha Venkatesh

2019 "Bayesian Optimization for Categorical and Category-Specific Continuous Inputs", *arXiv preprint arXiv:1911.12473*. (Cit. on pp. 61, 62.)

Nguyen, Vu, Sunil Gupta, Santu Rana, Cheng Li, and Svetha Venkatesh

2017 "Regret for expected improvement over the best-observed value and stopping condition", in *Asian Conference on Machine Learning*, PMLR, pp. 279-294. (Cit. on p. 17.)

Nguyen, Vu, Tam Le, Makoto Yamada, and Michael A Osborne

2021 "Optimal Transport Kernels for Sequential and Parallel Neural Architecture Search", in *International Conference on Machine Learning*, PMLR. (Cit. on pp. 80, 94.)

Niu, Mu, Zhenwen Dai, Neil Lawrence, and Kolja Becker

2016 "Spatio-temporal Gaussian processes modeling of dynamical systems in systems biology", *arXiv preprint arXiv:1610.05163*.

Nugent, Christopher E, Thomas E Vollmann, and John Ruml

1968 "An experimental comparison of techniques for the assignment of facilities to locations", *Operations research*, 16, 1, pp. 150-173. (Cit. on p. 207.)

Nyikosa, Favour M, Michael A Osborne, and Stephen J Roberts

2018 "Bayesian Optimization for Dynamic Problems", *arXiv preprint arXiv:1803.03432*.

2019 “Adaptive Configuration Oracle for Online Portfolio Selection Methods”, *arXiv preprint arXiv:1908.08258*.

O’Donnell, Ryan

2014 *Analysis of boolean functions*, Cambridge University Press.

Oh, Changyong, Roberto Bondesan, Dana Kianfar, Rehan Ahmed, Rishubh Khurana, Payal Agarwal, Romain Lepert, Mysore Sriram, and Max Welling

2022 “Bayesian Optimization for Macro Placement”, *arXiv preprint arXiv:2207.08398*. (Cit. on p. 18.)

Oh, ChangYong, Efstratios Gavves, and Max Welling

2018 “BOCK: Bayesian optimization with cylindrical kernels”, in *International Conference on Machine Learning*, PMLR, pp. 3868-3877. (Cit. on pp. 37, 39, 44, 53, 55, 61, 181.)

Oh, Changyong, Efstratios Gavves, and Max Welling

2021 “Mixed variable Bayesian optimization with frequency modulated kernels”, in *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, ed. by Cassio de Campos and Marloes H. Maathuis, Proceedings of Machine Learning Research, PMLR, vol. 161, pp. 950-960. (Cit. on p. 16.)

Oh, Changyong, Jakub M Tomczak, Efstratios Gavves, and Max Welling

2019 “Combinatorial Bayesian optimization using the graph cartesian product”, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 2914-2924. (Cit. on pp. 16, 53, 55, 59, 71, 80, 93, 178, 184.)

Ortega, Antonio, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst

2018 “Graph signal processing: Overview, challenges, and applications”, *Proceedings of the IEEE*, 106, 5, pp. 808-828. (Cit. on pp. 7, 38, 39, 41, 53.)

Osborne, Michael A, Roman Garnett, and Stephen J Roberts

2009 “Gaussian processes for global optimization”, in *3rd International Conference on Learning and Intelligent Optimization*, pp. 1-15.

Pang, Guofei, Paris Perdikaris, Wei Cai, and George Em Karniadakis

2017 “Discovering variable fractional orders of advection–dispersion equations from field data using multi-fidelity Bayesian optimization”, *Journal of Computational Physics*, 348, pp. 694-714. (Cit. on p. 19.)

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer

2017 *Automatic differentiation in pytorch*. (Cit. on pp. 81, 162.)

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al.

2011 "Scikit-learn: Machine learning in Python", *the Journal of machine Learning research*, 12, pp. 2825-2830. (Cit. on p. 64.)

Perdikaris, Paris and George Em Karniadakis

2016 "Model inversion via multi-fidelity Bayesian optimization: a new paradigm for parameter estimation in haemodynamics, and beyond", *Journal of The Royal Society Interface*, 13, 118, p. 20151107. (Cit. on p. 19.)

Perrone, Valerio, Rodolphe Jenatton, Matthias Seeger, and Cédric Archambeau

2018 "Scalable hyperparameter transfer learning", in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 6846-6856. (Cit. on p. 14.)

Perrone, Valerio, Huibin Shen, Matthias W Seeger, Cedric Archambeau, and Rodolphe Jenatton

2019 "Learning search spaces for bayesian optimization: Another view of hyperparameter transfer learning", *Advances in Neural Information Processing Systems*, 32. (Cit. on pp. 108, 109.)

Pham, Hieu, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean

2018 "Efficient neural architecture search via parameters sharing", in *International Conference on Machine Learning*, PMLR , pp. 4095-4104. (Cit. on pp. 16, 61, 67.)

Phan, Du, Neeraj Pradhan, and Martin Jankowiak

2019 "Composable Effects for Flexible and Accelerated Probabilistic Programming in NumPyro", *arXiv preprint arXiv:1912.11554*.

Pleiss, Geoff, Jacob Gardner, Kilian Weinberger, and Andrew Gordon Wilson

2018 "Constant-time predictive distributions for Gaussian processes", in *International Conference on Machine Learning*, PMLR, pp. 4114-4123. (Cit. on p. 5.)

Poloczek, Matthias, Jialei Wang, and Peter I Frazier

2016 "Warm starting Bayesian optimization", in *2016 Winter Simulation Conference (WSC)*, IEEE, pp. 770-781. (Cit. on p. 109.)

Popova, Mariya, Olexandr Isayev, and Alexander Tropsha

2018 "Deep reinforcement learning for de novo drug design", *Science advances*, 4, 7, eaap7885. (Cit. on pp. 17, 18.)

Potvin, Jean-Yves

1996 "Genetic algorithms for the traveling salesman problem", *Annals of Operations Research*, 63, pp. 337-370. (Cit. on p. 86.)

Pyzer-Knapp, Edward O

2018 "Bayesian optimization for accelerated drug discovery", *IBM Journal of Research and Development*, 62, 6, pp. 2-1. (Cit. on pp. 17, 18.)

Qin, Lijing, Shouyuan Chen, and Xiaoyan Zhu

2014 "Contextual combinatorial bandit and its application on diversified online recommendation", in *Proceedings of the 2014 SIAM International Conference on Data Mining*, SIAM, pp. 461-469.

Rahimi, Ali, Benjamin Recht, et al.

2007 "Random Features for Large-Scale Kernel Machines.", in *NIPS*, Citeseer, vol. 3, p. 5. (Cit. on p. 79.)

Rana, Santu, Cheng Li, Sunil Gupta, Vu Nguyen, and Svetha Venkatesh

2017 "High dimensional Bayesian optimization with elastic gaussian process", in *International Conference on Machine Learning*, pp. 2883-2891. (Cit. on pp. 25, 29, 30.)

Raskutti, Garvesh and Caroline Uhler

2018 "Learning directed acyclic graph models based on sparsest permutations", *Stat*, 7, 1, e183. (Cit. on p. 86.)

Rasmussen, Carl Edward

2003 "Gaussian processes in machine learning", in *Summer School on Machine Learning*, Springer, pp. 63-71. (Cit. on pp. 14, 15, 23, 37, 60.)

Real, Esteban, Alok Aggarwal, Yanping Huang, and Quoc V Le

2019 "Regularized evolution for image classifier architecture search", in *Proceedings of the aaai conference on artificial intelligence*, vol. 33, pp. 4780-4789. (Cit. on pp. 49, 50, 54, 67, 163, 171, 184.)

Reeves, Colin R

1995 "A genetic algorithm for flowshop sequencing", *Computers & operations research*, 22, 1, pp. 5-13. (Cit. on p. 207.)

Remes, Sami, Markus Heinonen, and Samuel Kaski

2017 "Non-stationary spectral kernels", in *Advances in neural information processing systems*, pp. 4642-4651. (Cit. on pp. 53, 61.)

Resende, Mauricio GC, LS Pitsoulis, and PM Pardalos

1997 "Approximate solution of weighted MAX-SAT problems using GRASP", *Satisfiability problems*, 35, pp. 393-405. (Cit. on p. 38.)

Reymond, Jean-Louis and Mahendra Awale

2012 "Exploring chemical space for drug discovery using the chemical universe database", *ACS chemical neuroscience*, 3, 9, pp. 649-657. (Cit. on p. 53.)

Riquelme, Carlos, George Tucker, and Jasper Snoek

2018 "Deep Bayesian Bandits Showdown: An Empirical Comparison of Bayesian Deep Networks for Thompson Sampling", in *International Conference on Learning Representations*.

Roos, Teemu, Tomi Silander, Petri Kontkanen, and Petri Myllymaki

2008 "Bayesian network structure learning using factorized NML universal models", in *2008 Information Theory and Applications Workshop*, IEEE, pp. 272-276. (Cit. on p. 208.)

Ru, Binxin, Ahsan Alvi, Vu Nguyen, Michael A Osborne, and Stephen Roberts

2020 "Bayesian optimisation over multiple continuous and categorical inputs", in *International Conference on Machine Learning*, PMLR, pp. 8276-8285. (Cit. on pp. 14, 16, 61-63.)

Ru, Binxin, Xingchen Wan, Xiaowen Dong, and Michael Osborne

2020 "Interpretable Neural Architecture Search via Bayesian Optimisation with Weisfeiler-Lehman Kernels", in *International Conference on Learning Representations*. (Cit. on pp. 16, 18.)

Russo, Daniel J, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen

2018 "A Tutorial on Thompson Sampling", *Foundations and Trends® in Machine Learning*, 11, 1, pp. 1-96. (Cit. on p. 109.)

Sakaue, Shinsaku

2020 "Guarantees of Stochastic Greedy Algorithms for Non-monotone Submodular Maximization with Cardinality Constraint", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 11-21. (Cit. on pp. 73, 206.)

Samaranayake, Meththa, Helen Ji, and John Ainscough

2009 "Development of a force directed module placement tool", in *2009 Ph. D. Research in Microelectronics and Electronics*, IEEE, pp. 152-155. (Cit. on pp. 100-102.)

Samo, Yves-Laurent Kom and Stephen Roberts

2015 "Generalized spectral kernels", *arXiv preprint arXiv:1506.02236*. (Cit. on p. 61.)

Scanagatta, Mauro, Antonio Salmerón, and Fabio Stella

2019 "A survey on Bayesian network structure learning from data", *Progress in Artificial Intelligence*, 8, 4, pp. 425-439.

Schäfer, Christian

2013 "Particle algorithms for optimization on binary spaces", *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 23, 1, p. 8.

Scholkopf, Bernhard and Alexander J Smola

2001 *Learning with kernels: support vector machines, regularization, optimization, and beyond*, MIT press. (Cit. on p. 55.)

Schölkopf, Bernhard, Alexander Smola, and Klaus-Robert Müller

1998 "Nonlinear component analysis as a kernel eigenvalue problem", *Neural computation*, 10, 5, pp. 1299-1319.

Scutari, Marco

2010 "Learning Bayesian networks with the bnlearn R package", *Journal of Statistical Software*. (Cit. on p. 86.)

Scutari, Marco, Catharina Elisabeth Graafland, and José Manuel Gutiérrez

2019 "Who learns better Bayesian network structures: Accuracy and speed of structure learning algorithms", *International Journal of Approximate Reasoning*, 115, pp. 235-253. (Cit. on pp. 85, 86.)

Shah, Amar and Zoubin Ghahramani

2015 "Parallel predictive entropy search for batch global optimization of expensive objective functions", in *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pp. 3330-3338. (Cit. on pp. 79, 109.)

Shahookar, Khushro and Pinaki Mazumder

1991 "VLSI cell placement techniques", *ACM Computing Surveys (CSUR)*, 23, 2, pp. 143-220. (Cit. on p. 8.)

Shahriari, Bobak, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas

2015 "Taking the human out of the loop: A review of Bayesian optimization", *Proceedings of the IEEE*, 104, 1, pp. 148-175. (Cit. on pp. 4, 5, 15, 18, 24, 37, 44, 55, 73, 93, 95.)

Sharma, Dravyansh, Ashish Kapoor, and Amit Deshpande

2015 "On greedy maximization of entropy", in *International Conference on Machine Learning*, PMLR, pp. 1330-1338. (Cit. on p. 206.)

Silander, Tomi, Janne Leppä-Aho, Elias Jääsaari, and Teemu Roos

2018 "Quotient normalized maximum likelihood criterion for learning Bayesian network structures", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 948-957. (Cit. on p. 208.)

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al.

2017 "Mastering the game of go without human knowledge", *nature*, 550, 7676, pp. 354-359. (Cit. on pp. 4, 16.)

Skiena, Steven S

1998 *The algorithm design manual: Text*, Springer Science & Business Media, vol. 1. (Cit. on pp. 62, 181.)

Smola, Alexander J and Risi Kondor

2003 "Kernels and regularization on graphs", in *Learning theory and kernel machines*, Springer, pp. 144-158. (Cit. on pp. 7, 37, 39, 41, 55-57, 59, 61, 62, 64, 178.)

Snoek, Jasper, Hugo Larochelle, and Ryan Prescott Adams

2012 "Practical Bayesian Optimization of Machine Learning Algorithms", *Advances in Neural Information Processing Systems*. (Cit. on pp. 5, 14-16, 24, 25, 27-29, 37, 39, 44, 53, 55, 71, 94, 158, 161, 181.)

Snoek, Jasper, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams

2015 "Scalable bayesian optimization using deep neural networks", in *International conference on machine learning*, PMLR, pp. 2171-2180. (Cit. on pp. 14, 16, 17, 23, 24, 44, 71, 94.)

Snoek, Jasper, Kevin Swersky, Rich Zemel, and Ryan Adams

2014 "Input warping for Bayesian optimization of non-stationary functions", in *International conference on machine learning*, 1674-1682. (Cit. on pp. 5, 14, 24, 25, 27-30, 158.)

Solnik, Benjamin, Daniel Golovin, Greg Kochanski, John Elliot Karro, Subhodeep Moitra, and D. Sculley

2017 "Bayesian Optimization for a Better Dessert", in *Proceedings of the 2017 NIPS Workshop on Bayesian Optimization*, The workshop is BayesOpt 2017 NIPS Workshop on Bayesian Optimization December 9, 2017, Long Beach, USA, December 9, 2017, Long Beach, USA. (Cit. on pp. 3, 53.)

Solus, Liam, Yuhao Wang, and Caroline Uhler

2021 "Consistency guarantees for greedy permutation-based causal inference algorithms", *Biometrika*, 108, 4, pp. 795-814. (Cit. on p. 86.)

Song, Xu, Stefanie Feih, Wei Zhai, Chen-Nan Sun, Feng Li, Raj Maiti, Jun Wei, Yangzhan Yang, Victor Oancea, Leon Romano Brandt, et al.

2020 "Advances in additive manufacturing process simulation: Residual stresses and distortion predictions in complex metallic components", *Materials & Design*, 193, p. 108779. (Cit. on p. 71.)

Spears, William M

1993 "Simulated annealing for hard satisfiability problems.", in *Cliques, Coloring, and Satisfiability*, Citeseer, pp. 533-558. (Cit. on p. 45.)

Springenberg, Jost Tobias, Aaron Klein, Stefan Falkner, and Frank Hutter

2016 "Bayesian optimization with robust Bayesian neural networks", in *Advances in Neural Information Processing Systems* , pp. 4134-4142. (Cit. on pp. 14, 23, 108.)

Srinivas, Niranjan, Andreas Krause, Sham Kakade, and Matthias Seeger

2010 "Gaussian process optimization in the bandit setting: no regret and experimental design", in *International Conference on Machine Learning*, pp. 1015-1022. (Cit. on pp. 16, 17, 23, 73, 76, 81, 100, 194, 195, 199, 200.)

STANDARDS, NATIONAL INSTITUTE OF and TECHNOLOGY/SEMATECH

2013 *e-Handbook of statistical methods*. (Cit. on p. 18.)

Stanton, Samuel, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson

2022 "Accelerating Bayesian Optimization for Biological Sequence Design with Denoising Autoencoders", in *International Conference on Machine Learning*.

Strang, Gilbert

2006 *Linear algebra and its applications*, Thomson, Brooks/Cole, Belmont, CA.

Sun, Shengyang, Guodong Zhang, Jiaxin Shi, and Roger Grosse

2018 "FUNCTIONAL VARIATIONAL BAYESIAN NEURAL NETWORKS", in *International Conference on Learning Representations*.

Sun, Shengyang, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger Grosse

2018 "Differentiable compositional kernel learning for Gaussian processes", in *International Conference on Machine Learning*, PMLR, pp. 4828-4837. (Cit. on p. 78.)

Sutton, Richard S and Andrew G Barto

2018 *Reinforcement learning: An introduction*, MIT press. (Cit. on pp. 17, 100.)

Swersky, Kevin, Yulia Rubanova, David Dohan, and Kevin Murphy

2020 "Amortized bayesian optimization over discrete spaces", in *Conference on Uncertainty in Artificial Intelligence*, PMLR , pp. 769-778. (Cit. on pp. 14, 71, 80, 107.)

Swersky, Kevin, Jasper Snoek, and Ryan P Adams

2013 "Multi-task bayesian optimization", in *Advances in neural information processing systems*, pp. 2004-2012. (Cit. on pp. 24, 53, 61, 109.)

Swersky, Kevin Jordan

2017 *Improving Bayesian Optimization for Machine Learning using Expert Priors*, PhD thesis, University of Toronto. (Cit. on pp. 22, 24, 25.)

Syslo, Maciej, Narsingh Deo, and Janusz S Kowalik

2006 *Discrete optimization algorithms: with Pascal programs*, Dover Publications.

Tang, Xiaoping, Ruiqi Tian, and DF Wong

2001 "Fast evaluation of sequence pair in block placement by longest common subsequence computation", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20, 12, pp. 1406-1413. (Cit. on p. 94.)

Tang, Xiaoping and DF Wong

2001 "FAST-SP: A fast algorithm for block placement based on sequence pair", in *Proceedings of the 2001 Asia and South Pacific design automation conference*, pp. 521-526. (Cit. on p. 94.)

*The-OpenROAD-Project*

2021 `https://github.com/The-OpenROAD-Project/OpenROAD`. (Cit. on p. 99.)

Thompson, William R

1933 "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples", *Biometrika*, 25, 3/4, pp. 285-294. (Cit. on pp. 23, 79.)

Tighineanu, Petru, Kathrin Skubch, Paul Baireuther, Attila Reiss, Felix Berkenkamp, and Julia Vinogradska

2022 "Transfer Learning with Gaussian Processes for Bayesian Optimization", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 6152-6181. (Cit. on p. 108.)

Titsias, Michalis

2009 "Variational learning of inducing variables in sparse Gaussian processes", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 567-574. (Cit. on p. 15.)

Titsias, Michalis K, Neil Lawrence, and Magnus Rattray

2008 "Markov chain Monte Carlo algorithms for Gaussian processes", *Inference and Estimation in Probabilistic Time-Series Models*, 9.

Trench, William F

1999 "Asymptotic distribution of the spectra of a class of generalized Kac–Murdock–Szegö matrices", *Linear algebra and its applications*, 294, 1-3, pp. 181-192. (Cit. on p. 202.)

Van Laarhoven, Peter JM and Emile HL Aarts

1987 "Simulated annealing", in *Simulated annealing: Theory and applications*, Springer, pp. 7-15. (Cit. on p. 11.)

Vashisht, Dhruv, Harshit Rampal, Haiguang Liao, Yang Lu, Devika Shanbhag, Elias Fallon, and Levent Burak Kara

2020 "Placement in Integrated Circuits using Cyclic Reinforcement Learning and Simulated Annealing", *arXiv preprint arXiv:2011.07577*. (Cit. on p. 99.)

Venkitaraman, Arun, Saikat Chatterjee, and Peter Handel

2020 "Gaussian processes over graphs", in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, pp. 5640-5644.

Vert, Jean-Philippe, Koji Tsuda, and Bernhard Schölkopf

2004 "A primer on kernel methods", *Kernel methods in computational biology*, 47, pp. 35-70. (Cit. on pp. 53, 56, 58.)

Vishwanathan, S Vichy N, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt

2010 "Graph kernels", *Journal of Machine Learning Research*, 11, Apr, pp. 1201-1242.

Volpp, Michael, Lukas P Fröhlich, Kirsten Fischer, Andreas Doerr, Stefan Falkner, Frank Hutter, and Christian Daniel

2019 "Meta-Learning Acquisition Functions for Transfer Learning in Bayesian Optimization", in *International Conference on Learning Representations*. (Cit. on p. 109.)

Vondrák, Jan

2007 "Submodularity in combinatorial optimization
".

Wan, Xingchen, Vu Nguyen, Huong Ha, Binxin Ru, Cong Lu, and Michael A Osborne

2021 "Think Global and Act Local: Bayesian Optimisation over High-Dimensional Categorical and Mixed Search Spaces", in *International Conference on Machine Learning*, PMLR, pp. 10663-10674. (Cit. on p. 106.)

Wang, Jialei, Scott C Clark, Eric Liu, and Peter I Frazier

2020 "Parallel Bayesian global optimization of expensive functions", *Operations Research*, 68, 6, pp. 1850-1865. (Cit. on pp. 74, 79.)

Wang, Ke, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson

2019 "Exact Gaussian processes on a million data points", *Advances in Neural Information Processing Systems*, 32. (Cit. on p. 108.)

Wang, Zi, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka

2018 "Batched large-scale bayesian optimization in high-dimensional spaces", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 745-754. (Cit. on pp. 14, 79.)

Wang, Zi, Chengtao Li, Stefanie Jegelka, and Pushmeet Kohli

2017 "Batched high-dimensional Bayesian optimization via structural kernel learning", in *International Conference on Machine Learning*, PMLR, pp. 3656-3664. (Cit. on pp. 14, 24, 79.)

Wang, Zi, Bolei Zhou, and Stefanie Jegelka

2016 "Optimization as estimation with Gaussian processes in bandit settings", in *Artificial Intelligence and Statistics*, PMLR
, pp. 1022-1031. (Cit. on pp. 76, 77, 79, 81, 96, 194-196, 199.)

Wang, Ziyu and Nando de Freitas

2014 "Theoretical analysis of Bayesian optimisation with unknown Gaussian process hyper-parameters", *arXiv preprint arXiv:1406.7758*.

Wang, Ziyu, Frank Hutter, Masrour Zoghi, David Matheson, and Nando de Feitas

2016 "Bayesian optimization in a billion dimensions via random embeddings", *Journal of Artificial Intelligence Research*, 55 , pp. 361-387. (Cit. on pp. 24, 25, 33, 48.)

Weinberger, ED

1996 "NP completeness of Kauffman's NK model, a tuneable rugged fitness landscape (1996)", *Santa Fe Institute working paper*.

White, Colin, Willie Neiswanger, and Yash Savani

2021 "Bananas: Bayesian optimization with neural architectures for neural architecture search", in *Proceedings of the AAAI Conference on Artificial Intelligence*, 12, vol. 35, pp. 10293-10301.

Wikipedia contributors

2020 *Flow shop scheduling — Wikipedia, The Free Encyclopedia*, [Online; accessed 3-February-2021], `https://en.wikipedia.org/w/index.php?title=Flow_shop_scheduling&oldid=992871042`. (Cit. on p. 207.)

Williams, Christopher KI and Carl Edward Rasmussen

2006 *Gaussian processes for machine learning*, MIT press Cambridge, MA, vol. 2. (Cit. on pp. 5, 62, 94, 151.)

Williams, Christopher KI and Matthias Seeger

2001 "Using the Nyström method to speed up kernel machines", in *Advances in neural information processing systems*, pp. 682-688.

Wilson, Aaron, Alan Fern, and Prasad Tadepalli

2014 "Using trajectory data to improve Bayesian optimization for reinforcement learning", *The Journal of Machine Learning Research*, 15, 1, pp. 253-282. (Cit. on p. 37.)

Wilson, Andrew and Hannes Nickisch

2015 "Kernel interpolation for scalable structured Gaussian processes (KISS-GP)", in *International Conference on Machine Learning*, pp. 1775-1784. (Cit. on p. 61.)

Wilson, Andrew G, Zhiting Hu, Russ R Salakhutdinov, and Eric P Xing

2016 "Stochastic variational deep kernel learning", *Advances in Neural Information Processing Systems*, 29. (Cit. on p. 108.)

Wilson, James, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth

2020 "Efficiently sampling functions from Gaussian process posteriors", in *International Conference on Machine Learning*, PMLR, pp. 10292-10302. (Cit. on pp. 79, 109.)

Wistuba, Martin and Josif Grabocka

2020 "Few-Shot Bayesian Optimization with Deep Kernel Surrogates", in *International Conference on Learning Representations*.

Wistuba, Martin, Ambrish Rawat, and Tejaswini Pedapati

2019 "A survey on neural architecture search", *arXiv preprint arXiv:1905.01392*. (Cit. on pp. 16, 38, 49, 50.)

Wolpert, David H and William G Macready

1997 "No free lunch theorems for optimization", *IEEE transactions on evolutionary computation*, 1, 1, pp. 67-82.

Wu, Jian and Peter I Frazier

2016 "The parallel knowledge gradient method for batch Bayesian optimization", in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 3134-3142. (Cit. on pp. 79, 95.)

Wu, Jian, Matthias Poloczek, Andrew G Wilson, and Peter Frazier

2017 "Bayesian optimization with gradients", in *Advances in Neural Information Processing Systems*, pp. 5267-5278. (Cit. on pp. 4, 11, 44.)

Xiao, Han, Kashif Rasul, and Roland Vollgraf

2017 "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms", *arXiv preprint arXiv:1708.07747*. (Cit. on p. 183.)

Xiaoping Tang, DF and Ruiqi Tian Wong

2000 "Fast evaluation of sequence pair in block placement by longest common subsequence computation date", in *Design, Automation and Test in Europe*, p. 106.

Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He

2017 "Aggregated residual transformations for deep neural networks", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492-1500.

Xie, Saining, Alexander Kirillov, Ross Girshick, and Kaiming He

2019 "Exploring randomly wired neural networks for image recognition", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1284-1293.

Xu, Chang, Gai Liu, Ritchie Zhao, Stephen Yang, Guojie Luo, and Zhiru Zhang

2017 "A Parallel Bandit-Based Approach for Autotuning FPGA Compilation", in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, FPGA '17, Association for Computing Machinery, pp. 157-166, ISBN: 9781450343541. (Cit. on pp. 93, 98, 100.)

Yang, Antoine, Pedro M Esperança, and Fabio M Carlucci

2019 "NAS evaluation is frustratingly hard", in *International Conference on Learning Representations*.

Yang, Kevin K, Yuxin Chen, Alycia Lee, and Yisong Yue

2019 "Batched stochastic Bayesian optimization via combinatorial constraints design", in *The 22nd International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 3410-3419. (Cit. on p. 18.)

Yuan, Ya-xiang

2000 "A review of trust region algorithms for optimization", in *Iciam*, 1, vol. 99, pp. 271-282.

Zaefferer, Martin, Jörg Stork, and Thomas Bartz-Beielstein

2014 "Distance measures for permutations in combinatorial efficient global optimization", in *International Conference on Parallel Problem Solving from Nature*, Springer, pp. 373-383. (Cit. on pp. 71, 78, 80, 96.)

Zagoruyko, Sergey and Nikos Komodakis

2016 "Wide Residual Networks", in *British Machine Vision Conference 2016*, British Machine Vision Association.

Zela, Arber, Julien Niklas Siems, Lucas Zimmer, Jovita Lukasik, Margret Keuper, and Frank Hutter

2022 "Surrogate NAS benchmarks: Going beyond the limited search spaces of tabular NAS benchmarks", in *Tenth International Conference on Learning Representations*, OpenReview. net, pp. 1-36. (Cit. on p. 106.)

Zhang, Baohe, Raghu Rajan, Luis Pineda, Nathan Lambert, André Biedenkapp, Kurtland Chua, Frank Hutter, and Roberto Calandra

2021 "On the importance of hyperparameter optimization for model-based reinforcement learning", in *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 4015-4023. (Cit. on p. 3.)

Zhang, Jianming, Xifan Yao, Min Liu, and Yan Wang

2019 "A Bayesian Discrete Optimization Algorithm for Permutation Based Combinatorial Problems", in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, pp. 874-881. (Cit. on pp. 71, 80.)

Zhang, K, J Peters, D Janzing, and B Schölkopf

2011 "Kernel-based Conditional Independence Test and Application in Causal Discovery", in *27th Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, AUAI Press, pp. 804-813.

Zhang, Shuhan, Wenlong Lyu, Fan Yang, Changhao Yan, Dian Zhou, and Xuan Zeng

2019 "Bayesian optimization approach for analog circuit synthesis using neural network", in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, pp. 1463-1468. (Cit. on pp. 3, 8.)

Zhou, Zhenpeng, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley

2019 "Optimization of molecules via deep reinforcement learning", *Scientific reports*, 9, 1, pp. 1-10.

Zhu, Ciyou, Richard H Byrd, Peihuang Lu, and Jorge Nocedal

1997 "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization", *ACM Transactions on Mathematical Software (TOMS)*, 23, 4, pp. 550-560. (Cit. on pp. 28, 30, 62, 181.)

Zoph, Barret and Quoc Le

2017 "Neural Architecture Search with Reinforcement Learning", in *International Conference on Learning Representations*. (Cit. on pp. 16-18, 66, 182.)

Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le

2018 "Learning transferable architectures for scalable image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697-8710. (Cit. on pp. 3, 17.)

# ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my supervisor, *Max Welling*, for providing me with the opportunity to pursue a PhD and for his guidance and advice. The feedback and insight Max gave were the staple sources of learning during my PhD. I also want to express my appreciation to my co-supervisor, *Efstratios Gavves*, for his advice and help. Stratis spared no efforts to provide help on various aspects of research skills, without which my life would have been much tougher. I am also grateful to my other co-supervisor, *Roberto Bondesan*. Roberto provided excellent guidance in interdisciplinary projects, from which I learned how to manage interdisciplinary projects.

I was also fortunate to have excellent collaborators. Via discussions with *Jakub Tomczak*, I received excellent feedback and learned the importance of a positive attitude in research and collaboration. Thankfully, *Mijung Park* gave me an opportunity to work on a new topic in a new environment. Mijung also shared many beneficial experiences and tips helpful to become a good researcher. Thanks to Qualcomm Combinatorial optimization team, I learned a lot about the real-world application of machine learning while doing an internship.

I cherish my memory with *QUvA/AMLab* members, from their warm welcome to all discussions and fun activities we shared. All of them boosted my energy for my PhD progress. Discussions with QUvA/AMLab members taught me various subfields of machine learning, which enriched my research and broadened my view. During the pandemic, my limited social interaction managed to survive, thanks to *Shuai Liao*, *Mert Kilickaya*, and *Noureldien Hussein*, which helped me to stay healthy socially and mentally.

Thanks should also go to *Jim Boelrijk*, *David Ruhe*, and *Wouter Kool* for the Dutch translation of the Summary. Thank you to *Felice Arends*, *Virginie Mes*, and *Dennis Koelma* for their help with administration and IT.

Lastly, I want to thank my family and friends for their support and interaction. Without them, I could not have decided to do PhD and would not be greeting this moment.

# A BOCK : BAYESIAN OPTIMIZATION WITH CYLINDRICAL KERNELS

## A.1 SPECIAL TREATMENT FOR THE CENTER POINT

In Subsec. 3.3.3, we propose a special treatment for the center point to correct the problem resulting from the over-expansion for the center point. We also show that the cylindrical kernel is positive semi-definiteness with the special treatment for the center point.

The cylindrical kernel $K_{cyl}$ is a tensor product of the kernel $K_r$ from the radius component, and the kernel $K_a$ from the angular component. If we can show that both $K_r$ and $K_d$ are proper kernels, i.e. positive semi-definite, then we can conclude that $K_{cyl}$ is also positive semi-definite (Williams and Rasmussen, 2006).

Let us denote with $T : B(\mathbf{o}, R) \to C(0, R; \mathbf{o}, 1)$ the transformation from a ball to a cylinder, and with $\pi_a$ the projection to angle component in a cylinder. For a given set $\widetilde{\mathcal{D}} = \mathcal{D} \cup \{\mathbf{o}\}$, we denote the angle component $\pi_a(T(\mathcal{D}))$ as $\mathcal{D}_a$. Then the gram matrix $K_a$ on $\widetilde{\mathcal{D}}$ can be represented by

$$\begin{bmatrix} K_a(\mathcal{D}_a, \mathcal{D}_a) & K_a(\mathcal{D}_a, \mathbf{a}_{arbitrary}) \\ K_a(\mathbf{a}_{arbitrary}, \mathcal{D}_a) & K_a(\mathbf{a}_{arbitrary}, \mathbf{a}_{arbitrary}) \end{bmatrix}$$

In the special treatment, we set $\mathbf{a}_{arbitrary} = \mathbf{a}_* = \mathbf{x}_* / \| \mathbf{x}_* \|$. This is nothing but the gram matrix of $K_a$ on the dataset $\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_N, \mathbf{a}_*$ As long as $K_a$ is a proper kernel, using the special treatment does not violate the positive semi-definiteness.

The special treatment assumes that the points where the prediction is made are not the center point, $\mathbf{x}_* \neq \mathbf{o}$. A single point is of measure zero under any non-atomic measure, and thus the assumption holds. In our experiments, we start with data including $\mathbf{o}$ as an initial data point, and thus the acquisition function does not need to go over $\mathbf{x}_* = 0$ anymore.

Interestingly, the special treatment bears similarity to Bayesian Optimization using treed Gaussian Processes (Assael et al., 2014). When there is $\mathbf{o}$ in our training data set, at each prediction, we have a Gaussian Process on the same data except for one point. Namely, one can view this as having different Gaussian Processes at different prediction points, in the sense that the data conditioning the Gaussian Process change (not the kernel parameters). As the treed Bayesian Optimization guarantees continuity between the regions having the different Gaussian Processes, the cylindrical kernel with the special treatment also has continuity since the Gram matrix is a continuous function of $\mathbf{d}_{\mathrm{arbitrary}}$.

However, at different prediction points, we have different gram matrices. Hence, a naive implementation of the above idea makes the maximization of the acquisition function infeasible. In Gaussian process prediction, the main computation bottle is to calculate a quadratic form as below.

$$\begin{bmatrix}\mathbf{p}^T & p_0\end{bmatrix}\left(\begin{bmatrix}K_{cyl}(\mathcal{D},\mathcal{D}) & K_{cyl}(\mathcal{D},0)\\ K_{cyl}(0,\mathcal{D}) & K_{cyl}(0,0)\end{bmatrix}+\sigma^2_{obs}I\right)^{-1}\begin{bmatrix}\mathbf{q}^T\\ q_0\end{bmatrix} \quad (A.1)$$

Fortunately, we can calculate the quadratic form Eq. (A.1) efficiently by using block matrix inversion. Once we calculate $K_{cyl}(\mathcal{D},\mathcal{D})^{-1}$, by using pre-calculated, $K_{cyl}(\mathcal{D},\mathcal{D})^{-1}$, calculating eq (A.1) for different $\mathbf{x}_*$ requires marginal computation.

## POSITIVE SEMI-DEFINITENESS OF THE CYLINDRICAL KERNEL

**Theorem A.1.** *If* $K_a(\mathbf{a},\mathbf{a}) = \eta > 0$, $\forall\,\mathbf{a} \in \mathcal{S}^{d-1}$, *then cylindrical kernels are positive semi-definite with the special treatment for the centre point.*

*Proof.* We need to show that $\forall n \in \mathbf{N}$, $\forall\,\mathbf{a}_i \in \mathbb{R}^n$, $\mathbf{c} \in \mathbb{R}^n$

$$\mathbf{c}^T K_a(\mathcal{D}_a,\mathcal{D}_a)\,\mathbf{c} \geqslant 0$$

where $\mathcal{D}_a = \pi_a(T(\mathcal{D}))$ and $\mathcal{D} = \{\mathbf{a}_i\}_{i=1,\cdots,n}$.

Case 1. $\mathbf{0} \notin \mathcal{D}$. In this case, this is positive semi-definite as shown in (Jayasumana et al., 2014).

Case 2. $\mathbf{0} \in \mathcal{D}$. Let us denote $\mathcal{D} = \mathcal{D}_{>0} \cup \{\mathbf{0}_a\}$, where $\mathbf{0}_a = \pi_a(T(\mathbf{0}))$ and $[\mathbf{c}^T \quad c]^T \in \mathbb{R}^n$. Then we need to show following

$$\begin{aligned}&\begin{bmatrix}\mathbf{c}^T & c\end{bmatrix}\begin{bmatrix}K(\mathcal{D}_{>0,a},\mathcal{D}_{>0,a}) & \eta\mathbf{1}\\ \eta\mathbf{1}^T & \eta\end{bmatrix}\begin{bmatrix}\mathbf{c}\\ c\end{bmatrix}\\ &= \mathbf{c}^T K_{>0,a}\,\mathbf{c} - 2\eta c\mathbf{1}^T\mathbf{c} + \eta c^2 \geqslant 0 \end{aligned}\quad (A.2)$$

where $\mathcal{D}_{>0,a} = \pi_a(\mathcal{D}_{>0})$, $\mathbf{1} \in \mathbb{R}^{n-1}$ and $K_{>0,a} = K(\mathcal{D}_{>0,a},\mathcal{D}_{>0,a})$.

After differentiation, we have

$$\begin{aligned}\frac{\partial}{\partial\,\mathbf{c}}\mathbf{c}^T K_{>0,a}\,\mathbf{c} - 2\eta c\mathbf{1}^T\mathbf{c} + \eta c^2 &= 2K_{>0,a}\,\mathbf{c} - 2\eta c\mathbf{1}\\ \frac{\partial}{\partial c}\mathbf{c}^T K_{>0,a}\,\mathbf{c} - 2\eta c\mathbf{1}^T\mathbf{c} + \eta c^2 &= -2\eta\mathbf{1}^T\mathbf{c} + 2\eta c\end{aligned}$$

Thus the minimum of the quadratic form Eq. A.2 satisfies

$$K_{>0,a}\,\mathbf{c} = \eta c\mathbf{1} \quad \text{and} \quad \eta\mathbf{1}^T\mathbf{c} = \eta c$$

By substituting this into Eq. A.2, we get

$$\eta\,\mathbf{c}^T\mathbf{1}\mathbf{1}^T\mathbf{c} - 2\eta\,\mathbf{c}^T\mathbf{1}\mathbf{1}^T\mathbf{c} + \eta(\mathbf{1}^T\mathbf{c})^2 = \eta\cdot 0 \geqslant 0$$

The minimum of Eq. A.2 is zero. We have shown that the special treatment for the center point guarantees that cylindrical kernels are positive semi-definite.

□

## A.2 BENCHMARKS

## A.3 BENCHMARK FUNCTIONS

The suggested search space for below benchmark functions are adjusted to be $[-1, 1]^D$ in our experiments.

### REPEATED BRANIN

$$f_{rep-branin}(x_1, x_2, \cdots, x_D) = 1/\lfloor \frac{D}{2} \rfloor \sum_{i=1}^{\lfloor D/2 \rfloor} f_{branin}(x_{2i-1}, x_{2i})$$

where $f_{branin}$ is branin function whose formula can be found in (Laguna and Marti, 2005). The original search space of branin function is $[-5, 10] \times [0, 15]$.

### REPEATED HARTMANN6

$$\begin{aligned} & f_{rep-hartmann6}(x_1, x_2, \cdots, x_D) \\ & = 1/\lfloor \frac{D}{6} \rfloor \sum_{i=1}^{\lfloor D/6 \rfloor} f_{hartmann6}(x_{6i-5}, x_{6i-4}, x_{6i-3}, x_{6i-2}, x_{6i-1}, x_{6i}) \end{aligned}$$

where $f_{hartmann6}$ is hartmann6 function whose formula can be found in (Laguna and Marti, 2005). The original search space of hartmann6 function is $[0, 1]^6$.

### ROSENBROCK (Laguna and Marti, 2005)

$$f_{rosenbrock}(x_1, x_2, \cdots, x_D) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$$

The original search space is $[-5, 10]^D$.

### LEVY (Laguna and Marti, 2005)

$$\begin{aligned} f_{levy}(x_1, x_2, \cdots, x_D) = \sin^2(\pi w_1) \sum_{i=1}^{D-1} \Big\{ & (w_i - 1)^2 \left[1 + 100 \sin^2(\pi w_i + 1)\right] \\ & + (w_D - 1)^2 \left[1 + \sin^2(2\pi w_D)\right] \Big\} \\ \text{where} \quad w_i = 1 + & \frac{x_i - 1}{4} \end{aligned}$$

The original search space is $[-10, 10]^D$.

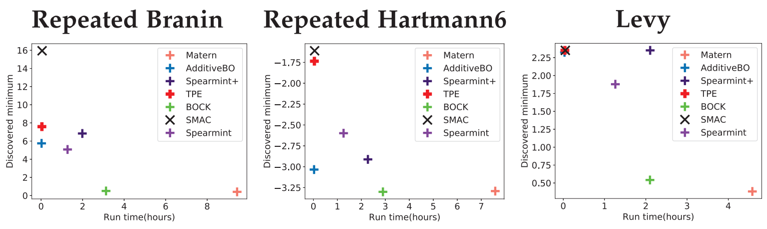


**Figure A.1:** Accuracy vs wall clock time on 20 dim. benchmarks

## A.4 ADDITIONAL EXPERIMENTS

### A.4.1 Efficiency vs Accuracy

We conduct the same analysis for efficiency vs accuracy with other benchmark functions on 20 dimensional case. In all cases, BOCK is the closest to the optimum operating point $(0, \cdots, 0)$. Matern is also accurate enough, although considerably slower, while SMAC and additive BO are faster but considerably less accurate.

### A.4.2 Scalability

We also experiment to check the scalability of algorithms with other benchmark functions on 20 and 100 dim. with 200 and 600 function evaluations respectively for all methods. In Fig. A.2, The solid lines and colored regions represent the mean wall clock time and one standard deviation over these 5 runs. As obtaining the evaluation score $y = f(\mathbf{x}_*)$ on these benchmark functions is instantaneous, the wall clock time is directly related to the computational efficiency of algorithms. In Fig. A.2, we compare BOCK and BOs with relative high accuracy in all benchmark functions, such as Spearmint and Matern. BOCK is clearly more efficient, all the while being less affected by the increasing number of dimensions. We make the same observation that BOCK is more efficient and less affected by the increasing dimensionality.

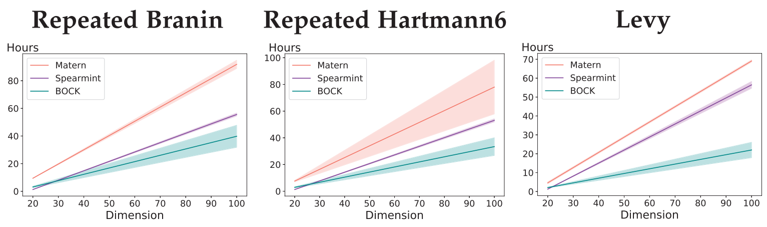


**Figure A.2:** Wall clock time on benchmarks of 20 and 100 dim.

# B COMBINATORIAL BAYESIAN OPTIMIZATION USING THE GRAPH CARTESIAN PRODUCT

## B.1 GRAPH CARTESIAN PRODUCT

### B.1.1 Graph Cartesian product and Hamming distance

**Theorem B.1.** *Assume a combinatorial graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ constructed from categorical variables, $C_1, \cdots, C_N$, that is, $\mathcal{G}$ is the graph Cartesian product $\square_i \mathcal{G}(C_i)$ of complete sub-graphs $\{\mathcal{G}(C_i)\}_i$. Then the shortest path $s(v_1, v_2; \mathcal{G})$ between vertices $v_1 = (c_1^{(1)}, \cdots, c_N^{(1)}), v_2 = (c_1^{(2)}, \cdots, c_N^{(2)}) \in \mathcal{V}$ on $\mathcal{G}$ is equal to the Hamming distance between $(c_1^{(1)}, \cdots, c_N^{(1)})$ and $(c_1^{(2)}, \cdots, c_N^{(2)})$.*

*Proof.* From the definition of the graph Cartesian product, we have that the shortest path between $v_1$ and $v_2$ consists of edges that change a value in one categorical variable at a time. As a result, an edge between $c_i^{(1)}$ and $c_i^{(2)}$, i.e. a difference in the $i$-$th$ categorical variable, and all other edges fixed contributes one error to the Hamming distance. Therefore, we can define the shortest path between $v_1$ and $v_2$ as the sum over all edges for which $c_i^{(1)}$ and $c_i^{(2)}$ are different, $s(v_1, v_2; \mathcal{G}) = \sum_i \mathbb{1}[c_i^{(1)} \neq c_i^{(2)}]$. It is equivalent to the definition of the Hamming distance between two sets of categorical choices. $\square$

### B.1.2 Graph Fourier transform with the graph Cartesian product

The graph Cartesian products can help us improve the scalability of the eigendecomposition (Hammack et al., 2011). The Laplacian of the graph Cartesian product $\mathcal{G}_1 \square \mathcal{G}_2$ of two sub-graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ can be algebraically expressed using the Kronecker product $\otimes$ and the Kronecker sum $\oplus$ (Hammack et al., 2011):

$$L(\mathcal{G}_1 \square \mathcal{G}_2) = L(\mathcal{G}_1) \oplus L(\mathcal{G}_2) = L(\mathcal{G}_1) \otimes \mathbf{I}_1 + \mathbf{I}_2 \otimes L(\mathcal{G}_2), \tag{B.1}$$

where $\mathbf{I}$ denotes the identity matrix. As a consequence, considering the eigensystems $\{(\lambda_i^{(1)}, u_i^{(1)})\}$ and $\{(\lambda_j^{(2)}, u_j^{(2)})\}$ of $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively, the eigensystem of $\mathcal{G}_1 \square \mathcal{G}_2$ is $\{(\lambda_i^{(1)} + \lambda_j^{(2)}, u_i^{(1)} \otimes u_j^{(2)})\}$.

**Proposition B.2.** *Assume a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the graph Cartesian product of sub-graphs $\mathcal{G} = \square_i, \mathcal{G}_i$. Then the graph Fourier Transform of $\mathcal{G}$ can be computed in $O(\sum_{i=1}^{m} |\mathcal{V}_i|^3)$ while the direct computation takes $O(\prod_{i=1}^{m} |\mathcal{V}_i|^3)$.*

*Proof.* The graph Fourier Transform is the eigendecomposition of the graph Laplacian $L(\mathcal{G})$ where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The eigendecomposition is of cubic complexity with respect to the number of rows(= the number of columns), which is the number of vertices $|\mathcal{V}|$ for the graph Laplacian $L(\mathcal{G})$. If we directly compute the eigendecomposition of $L(\mathcal{G})$, it costs $O(\prod_i |\mathcal{V}|^3)$. If we utilize the graph Cartesian product, then we compute the eigendecomposition for each sub-graphs and combine those to obtain the eigendecomposition of the original full graph $\mathcal{G}$. The cost for the eigendecomposition of each subgraphs is $O(|\mathcal{V}_i|^3)$ and in total, it is summed to $O(\sum_i |\mathcal{V}|^3)$. For the graph Cartesian product, the graph Fourier Transform can be computed in $O(\sum_i |\mathcal{V}|^3)$. □

**REMARK** In the computation of gram matrices, eigenvalues from sub-graphs are summed and entries of eigenvectors are multiplied. Compared to the cost of $O(\prod_i |\mathcal{V}|^3)$, this overhead is marginal. Thus with the graph Cartesian product, the ARD diffusion kernel can be computed efficiently with a pre-computed eigensystem for each sub-graphs. This pre-computation is performed efficiently by Prop. B.2.

## B.2 IMPLEMENTATION DETAILS

### B.2.1 GP-parameter posterior sampling

In the surrogate model fitting step of COMBO, GP-parameters are sampled from the posterior using slice sampling (Murray and Adams, 2010; Neal, 2003) as in Spearmint (Snoek, Larochelle, et al., 2012; Snoek, K. Swersky, et al., 2014).

For a nonzero mean function, the marginal likelihood of $\mathcal{D} = (V, \mathbf{y})$ is

$$-\frac{1}{2}(\mathbf{y} - m)^T(\sigma_f^2 K_{VV} + \sigma_n^2 I)^{-1}(\mathbf{y} - m)$$
$$-\frac{1}{2}\log\det(\sigma_f^2 K_{VV} + \sigma_n^2 I) - \frac{n}{2}\log 2\pi$$

where $m$ is the value of constant mean function. With ARD diffusion kernel, the gram matrix is given by

$$\sigma_f^2 K_{VV} + \sigma_n^2 I = \sigma_f^2 \bigotimes_i U_i \exp^{-\beta_i \Lambda_i} U_i^T + \sigma_n^2 I \tag{B.2}$$

where $\Lambda_i$ is a diagonal matrix whose diagonal entries are eigenvalues of a sub-graph given to a combinatorial variable $L(\mathcal{G}(C_i))$, $U_i$ is a orthogonal matrix whose columns are corresponding eigenvalues of $L(\mathcal{G}(C_i))$, signal variance $\sigma_f^2$ and noise variance $\sigma_n^2$.

**REMARK** In the implementation of Eq. (B.2), a normalized version $\exp^{-\beta_i \Lambda_i}/\Psi_i$ where $\Psi_i = 1/|\mathcal{V}_i| \sum_{j=1,\cdots|\mathcal{V}_i|} \exp^{-\beta_i \lambda_j^{(i)}}$ is used for numerical stability instead of $\exp^{-\beta_i \Lambda_i}$.

In the surrogate model fitting step of COMBO, all GP-parameters are sampled from the posterior which is proportional to the product of above marginal likelihood and priors on all GP-parameters such as $\beta_i$'s, signal variance $\sigma_f^2$, noise variance $\sigma_n^2$ and constant mean function value $m$. We use the slice sampling (Neal, 2003).

A single step of the slice sampling in COMBO consists of multiple univariate slice sampling steps:

1. $m$ : constant mean function value $m$
2. $\sigma_f^2$ : signal variance
3. $\sigma_n^2$ : noise variance
4. $\{\beta_i\}_i$ with a randomly shuffled order

In COMBO, the slice sampling does warm-up with 100 burn-in steps and at every new evaluation, 10 more samples are generated to approximate the posterior.

*Priors*

Especially in BO where data is scarce, priors used in the posterior sampling play an extremely important role. The Horseshoe priors are specified for $\beta_i$'s with the design goal of variable selection as stated in the main text. Here, we provide details about other GP-parameters including constant mean function value $m$, signal variance $\sigma_f^2$ and noise variance $\sigma_n^2$.

PRIOR ON CONSTANT MEAN FUNCTION VALUE $m$ Given $\mathcal{D} = (V, \mathbf{y})$ the prior over the mean function is the following:

$$p(m) \propto \begin{cases} \mathcal{N}(\mu, \sigma^2) & \text{if} \quad y_{min} \leqslant m \leqslant y_{max} \\ 0 & \text{otherwise} \end{cases}$$

where $\mu = mean(\mathbf{y})$, $\sigma = (y_{max} - y_{min})/4$, $y_{min} = \min(\mathbf{y})$ and $y_{max} = \max(\mathbf{y})$.

This is the truncated Gaussian distribution between $y_{min}$ and $y_{max}$ with a mean at the sample mean of $\mathbf{y}$. The truncation bound is set so that untruncated version can sample in truncation bound with the probability of around 0.95.

PRIOR ON SIGNAL VARIANCE $\sigma_f^2$ Given $\mathcal{D} = (V, \mathbf{y})$ the prior over the log-variance is the following:

$$p(\log(\sigma_f^2)) \propto \begin{cases} \mathcal{N}(\mu, \sigma^2) & \text{if} \quad \frac{\sigma_y^2}{K_{VV max}} \leqslant \sigma_f^2 \leqslant \frac{\sigma_y^2}{K_{VV min}} \\ 0 & \text{otherwise} \end{cases}$$

where $\sigma_y^2 = variance(\mathbf{y})$, $\mu = \frac{1}{2}(\frac{\sigma_y^2}{K_{VVmin}} + \frac{\sigma_y^2}{K_{VVmax}})$, $\sigma = \frac{1}{4}(\frac{\sigma_y^2}{K_{VVmin}} + \frac{\sigma_y^2}{K_{VVmax}})$, $K_{VVmin} = \min(K_{VV})$, $K_{VVmax} = \max(K_{VV})$ and $K_{VV} = K(V,V)$.

This is the truncated Log-Normal distribution. The intuition behind this choice of prior is that in GP prior, $\sigma_f^2 K_{VV}$ is the covariance matrix of $\mathbf{y}$ with the assumption of very small noise variance $\sigma_n^2$. The truncation bound is set so that the untruncated version can sample in truncation bound with the probability of around 0.95. Since for larger $\sigma_f^2$, the magnitude of the change of $\sigma_f^2$ has less significant effect than for smaller $\sigma_f^2$. In order to take into account the relative amount of change in $\sigma_f^2$, the Log-Normal distribution is used rather than the Normal distribution.

PRIORS ON SCALING FACTOR $\beta_i$ AND NOISE VARIANCE $\sigma_n^2$ We use the Horseshoe prior for $\beta_i$ and $\sigma_n^2$ in order to encourage sparsity. Since the probability density of the Horseshoe is intractable, its closed-form bound is used as a proxy (Carvalho et al., 2010):

$$\frac{K}{2}\log(1+\frac{4\tau^2}{x^2}) < p(x) < K\log(1+\frac{2\tau^2}{x^2})$$

where $x = \beta_i$ or $x = \sigma_n^2$, $\tau$ is a global shrinkage parameter and $K = (2\pi^3)^{-1/2}$ (Carvalho et al., 2010). Typically, the upper bound is used to approximate Horseshoe density. For $\beta_i$, we use $\tau = 5$ to avoid excessive sparsity. For $\sigma_n^2$, we use $\tau = \sqrt{0.05}$ that prefers very small noise similarly to the Spearmint implementation.[34]

### *Slice Sampling*

At every new evaluation in COMBO, we draw samples of $\beta_i$. For each $\beta_i$ the sampling procedure is the following:

SS-1 Set $t = 0$ and choose a starting $\beta_i^{(t)}$ for which the probability is non-zero.

SS-2 Sample a value q uniformly from $[0, p(\beta_i^{(t)}|\mathcal{D}, \beta_{-i}^{(t)}, m^{(t)}, (\sigma_f^2)^{(t)}, (\sigma_n^2))^{(t)}]$.

SS-3 Draw a sample $\beta_i$ uniformly from regions, $p(\beta_i|\mathcal{D}, \beta_{-i}^{(t)}, m^{(t)}, (\sigma_f^2)^{(t)}, (\sigma_n^2)^{(t)}) > q$.

SS-4 Set $\beta_i^{(t+1)} = \beta_i$ and repeat from SS-2 using $\beta_i^{(t+1)}$.

In SS-2, we step out using doubling and shrink to draw a new value. For detailed explanation about slice sampling, please refer to (Neal, 2003). For other GP-parameters, the same univariate slice sampling is applied.

34 https://github.com/JasperSnoek/spearmint

### B.2.2 Acquisition Function Optimization

In the acquisition function maximization step, we begin with candidate vertices chosen to balance between exploration and exploitation. $20,000$ vertices are randomly selected for exploration. To balance exploitation, we use 20 spray vertices similar to spray points[35] in (Snoek, Larochelle, et al., 2012). Spray vertices are randomly chosen in the neighborhood of a vertex with the best evaluation (e.g, $nbd(v_{best}, 2) = \{v | d(v, v_{best}) \leqslant 2\}$). Out of $20,020$ initial vertices, 20 vertices with the highest acquisition values are used as initial points for further optimization. This type of combination of heuristics for exploration and exploitation has shown improved performances (Garnett et al., 2010; Malkomes et al., 2016).

We use a breadth-first local search (BFLS) to further optimize the acquisition function. At a given vertex, we compare acquisition values on adjacent vertices. We then move to the vertex with the highest acquisition value and repeat until no acquisition value on an adjacent vertex is higher than the acquisition value at the current vertex.

NON-LOCAL SEARCH FOR ACQUISITION FUNCTION OPTIMIZATION We tried simulated annealing as a non-local search in different ways, namely:

- Randomly split 20 initial points into 2 sets of 10 points and optimize from 10 points in one set with BFLS and optimize from 10 points in another set with simulated annealing.
- For given 20 initial points, optimize from 20 points with BFLS and optimize from the same 20 points with simulated annealing.
- For given 20 initial points, firstly optimize from 20 points with BFLS and use 20 points optimized by BFLS as initial points for optimization using simulated annealing.

The optimum of all 3 methods is hardly better than the optimum discovered solely by BFLS. Therefore, we decided to stick to the simpler procedure without SA.

## B.3 ARCHITECTURE SEARCH DETAILS

SEARCH SPACE In our architecture search problem, the cell consists of one input state(**IN**), one output state(**OUT**) and five hidden states(**H1**~**H5**). The connectivity between 7 states are specified as in the left of Tab. 4.4 where it can be read that (**IN**→**H1**) and (**IN**→**H5**)

35 `https://github.com/JasperSnoek/spearmint/blob/b37a541be1ea035f82c7c82bbd93f5b4320e7d91/spearmint/spearmint/chooser/GPEIOptChooser.py#L235`

**Table B.1**: Comparison to NASNet search space

| Binary | | NASNet |
|---|---|---|
| Yes | Invalid Architecture | No |
| Not fixed | The number of inputs to each state | 2 |
| 4 | The number of computation type of states | 13 |

from the first row. Input and output states are identity maps. The computation type of each of 5 hidden states are determined by combination of 2 binary choices as in the right of Tab. 4.4.

In total, our search space consists of 31 binary variables. Note that we design a binary search space for NAS so that to also compare with BOCS. COMBO is not restricted to binary choices for NAS, however. We compare our search space with the NASNET search space.

EVALUATION For a given 31 binary choices, we construct a cell and stack 3 cells as follows

Input
Conv($3 \times 16 \times 3 \times 3$)-BN-ReLU
Cell with 16 channels
MaxPool($2 \times 2$)-Conv($16 \times 32 \times 1 \times 1$)
Cell with 32 channels
MaxPool($2 \times 2$)-Conv($32 \times 64 \times 1 \times 1$)
Cell with 64 channels
MaxPool($2 \times 2$)-FC($1024 \times 10$)
Output

At each MaxPool, the height and the width of features are halved.

The network is trained for 20 epochs with Adam (Kingma and Ba, 2015) with the default settings in pytorch (Paszke et al., 2017) except for the weight decay of $5 \times 10^{-5}$. CIFAR10 (Krizhevsky and Hinton, 2009) training data is randomly shuffled with random seed 0 in the command "numpy.RandomState(0).shuffle(indices)". In the shuffled training data, the first 30000 is used for training and the last 10000 is used for evaluations. The batch size is 100. Early stopping is applied when the current validation accuracy is worse than the validation accuracy 10 epochs ago.

Due to the small number of epochs, evaluations are a bit noisy. In order to stabilize evaluations, we run 4 times of training for a given architecture. On GTX 1080 Ti with 11GB, 4 runs can be run in parallel. Depending on a given architecture training took approximately 5~30 minutes.

Since the some binary choices result in invalid architectures, in such case, validation accuracy is given as 10%, which is the expected accuracy of constant prediction.

The final evaluation is given as

$$\text{Error}_{validation} + 0.02 \cdot \frac{\text{FLOPs of a given architecture}}{\text{Maximim FLOPs in the search space}}$$

where "Maximim FLOPs in the search space" is computed from the cell with all connectivity among states and Conv($5 \times 5$) for all **H1~H5**. 0.02 is set with the assumption that we can afford 1% of increased error with 50% reduction in FLOPs.

REGULARIZED EVOLUTION HYPERPARAMETERS In evolutionary search algorithms, the choice of mutation is critical to the performance. Since our binary search space is different from the NASNet search space where Regularized Evolution (RE) was originally applied, we modify mutation steps. All possible mutations proposed in (Real et al., 2019) can be represented as simple binary flipping in binary search space. In the binary search space, some binary variables are about computation type and others are about connectivity. Thus uniform-randomly choosing binary variables to flip can mutate computation type or connectivity. Since the binary search space is redundant we did not explicitly include identity mutation (not mutating). Since evolutionary search algorithms are believed to be less sample efficient than BO, we gave an advantage to RE by only allowing valid architectures in mutation steps.

On other crucial hyperparameters, population size $P$ and sample size $S$, motivated by the best value used in (Real et al., 2019), $P = 100$, $S = 25$.

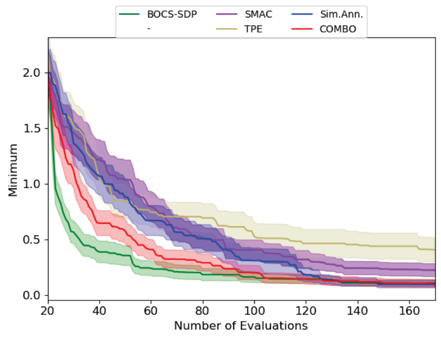
## B.4 EXPERIMENTAL RESULTS

### B.4.1 Bayesian optimization with binary variables

ISING SPARSIFICATION Ising sparsification is about approximating a zero-field Ising model expressed by $p(z) = \frac{1}{Z_p}\exp\{z^\top J^p z\}$, where $z \in \{-1, 1\}^n$, $J^p \in \mathbb{R}^{n \times n}$ is an interaction symmetric matrix, and $Z_p = \sum_z \exp\{z^\top J^p z\}$ is the partition function, using a model $q(z)$ with $J^q_{ij} = x_{ij} J^p_{ij}$ where $x_{ij} \in \{0, 1\}$ are the decision variables. The objective function is the regularized Kullback-Leibler divergence between $p$ and $q$

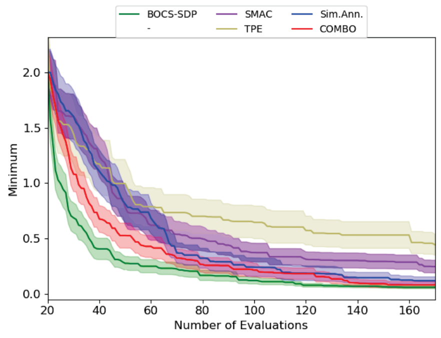
$$\mathcal{L}(x) = D_{KL}(p\|q) + \lambda\|x\|_1$$

where $\lambda > 0$ is the regularization coefficient $D_{KL}$ could be calculated analytically (Baptista and Poloczek, 2018). We follow the same setup as presented in (Baptista and Poloczek, 2018), namely, we consider $4 \times 4$ grid of spins, and interactions are sampled randomly from a uniform distribution over $[0.05, 5]$. The exhaustive search requires enumerating all $2^{24}$ configurations of $x$ that is infeasible. We consider $\lambda \in \{0, 10^{-4}, 10^{-2}\}$. We set the budget to 170 evaluations.



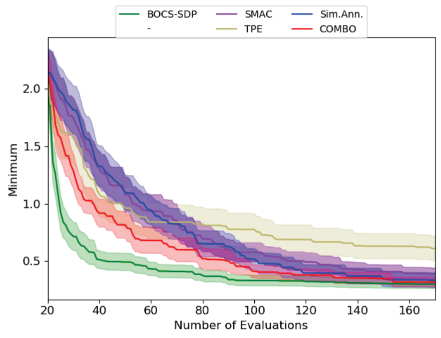
| Method | $\lambda = 0.0$ |
|---|---|
| SMAC | 0.1516±0.0404 |
| TPE | 0.4039±0.1087 |
| SA | **0.0953**±0.0331 |
| BOCS – SDP | 0.1049±0.0308 |
| COMBO | 0.1030±0.0351 |

**Figure B.1**: Ising sparsification ($\lambda = 0.0$)



| Method | $\lambda = 0.0001$ |
|---|---|
| SMAC | 0.2192±0.0522 |
| TPE | 0.4437±0.0952 |
| SA | 0.1166±0.0353 |
| BOCS – SDP | **0.0586**±0.0125 |
| COMBO | 0.0812±0.0279 |

**Figure B.2**: Ising sparsification ($\lambda = 0.0001$)



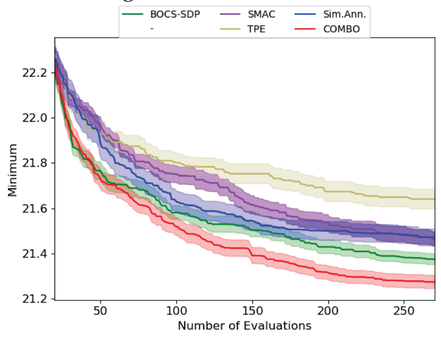
| Method | $\lambda = 0.01$ |
|---|---|
| SMAC | 0.3501±0.0447 |
| TPE | 0.6091±0.1071 |
| SA | 0.3342±0.0636 |
| BOCS – SDP | **0.3001**±0.0389 |
| COMBO | 0.3166±0.0420 |

**Figure B.3**: Ising sparsification ($\lambda = 0.01$)

CONTAMINATION CONTROL The contamination control in food supply chain is a binary optimization problem (Hu et al., 2010). The problem is about minimizing the contamination of food where at each stage a prevention effort can be made to decrease a possible contamination. Applying the prevention effort results in an additional cost $c_i$. However, if the food chain is contaminated at stage $i$, the contamination spreads at rate $\alpha_i$. The contamination at the $i$-th stage is represented by a random variable $\Gamma_i$. A random variable $z_i$ denotes a fraction of contaminated food at the $i$-th stage, and it could be expressed in an recursive manner, namely, $z_i = \alpha_i(1-x_i)(1-z_{i-1}) + (1-\Gamma_i x_i)z_{i-1}$, where $x_i \in \{0,1\}$ is the decision variable representing the preventing effort at stage $i$. Hence, the optimization problem is to make a decision at each stage whether the prevention effort should be applied so that to minimize the general cost while also ensuring that the upper limit of contamination is $u_i$ with probability at least $1 - \varepsilon$. The initial contamination and other random variables follow beta distributions that results in the following objective function

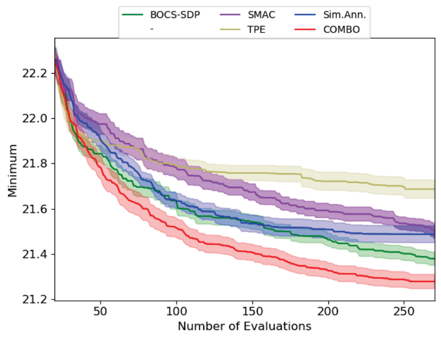
$$\mathcal{L}(x) = \sum_{i=1}^{d} \left[ c_i x_i + \frac{\rho}{T} \sum_{k=1}^{T} 1_{\{z_k > u_i\}} \right] + \lambda \|x\|_1$$

where $\lambda$ is a regularization coefficient, $\rho$ is a penalty coefficient (we use $\rho = 1$) and we set $T = 100$. Following (Baptista and Poloczek, 2018), we assume $u_i = 0.1$, $\varepsilon = 0.05$, and $\lambda \in \{0, 10^{-4}, 10^{-2}\}$. We set the budget to 270 evaluations.



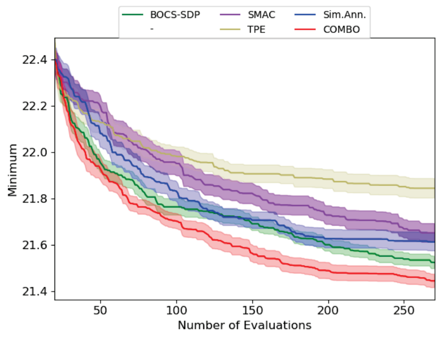
| Method | $\lambda = 0.0$ |
|---|---|
| SMAC | 21.4644±0.0312 |
| TPE | 21.6408±0.0437 |
| SA | 21.4704±0.0366 |
| BOCS – SDP | 21.3748±0.0246 |
| COMBO | **21.2752**±0.0292 |

Figure B.4: Contamination control ($\lambda = 0.0$)



| Method | $\lambda = 0.0001$ |
|---|---|
| SMAC | 21.5011±0.0329 |
| TPE | 21.6868±0.0406 |
| SA | 21.4871±0.0372 |
| BOCS – SDP | 21.3792±0.0296 |
| COMBO | **21.2784**±0.0314 |

Figure B.5: Contamination control ($\lambda = 0.0001$)



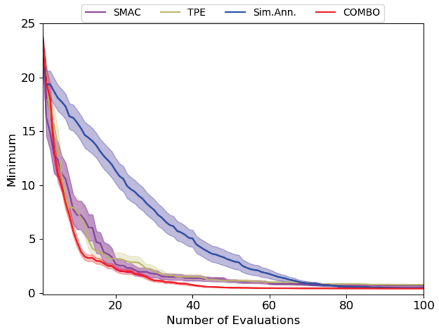
| Method | $\lambda = 0.01$ |
|---|---|
| SMAC | 21.6512±0.0403 |
| TPE | 21.8440±0.0422 |
| SA | 21.6120±0.0385 |
| BOCS – SDP | 21.5232±0.0269 |
| COMBO | **21.4436**±0.0293 |

Figure B.6: Contamination control ($\lambda = 0.01$)

### B.4.2 Bayesian optimization with ordinal and multi-categorical variables

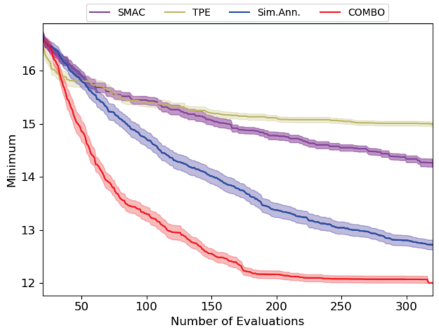
ORIDINAL VARIABLES : DISCRETIZED BRANIN In order to test COMBO on ordinal variables. We adopt widely used continuous benchmark branin function. Branin is defined on $[0,1]^2$, we discretize each dimension with 51 equally space points so that center point can be chosen in the discretized space. Therefore, the search space is comprised of 2 ordinal variables with 51 values(choices) for each.

COMBO outperforms all of its competitors. In Fig. B.7, SMAC and TPE exhibit similar search progress as COMBO, but in term of the final value at 100 evaluations, those two are overtaken by SA. COMBO maintains its better performance over all range of evaluations up to 100.



| Method | Branin |
|---|---|
| SMAC | 0.6962±0.0705 |
| TPE | 0.7578±0.0844 |
| SA | 0.4659±0.0211 |
| COMBO | **0.4113**±0.0012 |

Figure B.7: Ordinal variables : discretized branin



| Method | Pest |
|---|---|
| SMAC | 14.2614±0.0753 |
| TPE | 14.9776±0.0446 |
| SA | 12.7154±0.0918 |
| COMBO | **12.0012**±0.0033 |

Figure B.8: Multi-categorical variables : pest control

MULTI-CATEGORICAL VARIABLES : PEST CONTROL In the chain of stations, pest is spread in one direction, at each pest control station, the pest control officer can choose to use a pesticide from 4 different companies which differ in their price and effectiveness.

For N pest control stations, the search space for this problem is $5^N$, 4 choices of a pesticide and the choice of not using any of it.

The price and effectiveness reflect following dynamics.

- If you have purchased a pesticide a lot, then in your next purchase of the same pesticide, you will get discounted proportional to the amount you have purchased.
- If you have used a pesticide a lot, then pests will acquire strong tolerance to that specific product, which decrease effectiveness of that pesticide.

Formally, there are four variables: at $i$-th pest control $Z_i$ is the portion of the product having pest, $A_i$ is the action taken, $C_i^{(l)}$ is the adjusted cost of pesticide of type $l$, $T_i^{(l)}$ is the beta parameter of the Beta distribution for the effectiveness of pesticide of type $l$. It starts with initial $Z_0$ and follows the same evolution as in the contamination control, but after each choice of pesticide type whenever the taken action is to use one out of 4 pesticides or no action. $\{C_i^{(l)}\}_{1,\cdots,4}$ are adjusted in the manner that the pesticide which has been purchased most often will get a discount for the price. $\{T_i^{(l)}\}_{1,\cdots,4}$ are adjusted in the fashion that the pesticide which has been frequently used in previous control point cannot be as effective as before since the insects have developed tolerance to that.

The portion of the product having pest follows the dynamics below

$$z_i = \alpha_i(1 - x_i)(1 - z_{i-1}) + (1 - \Gamma_i x_i) z_{i-1} \tag{B.3}$$

when the pesticide is used, the effectiveness $x_i$ of pesticide follows beta distribution with the parameters, which has been adjusted according to the sequence of actions taken in previous control points.

Under this setting, our goal is to minimize the expense for pesticide control and the portion of products having pest while going through the chain of pest control stations. The objective is similar to the contamination control problem

$$\mathcal{L}(x) = \sum_{i=1}^{d} \left[ c_i x_i + \frac{\rho}{T} \sum_{k=1}^{T} 1_{\{z_k > u_i\}} \right]$$

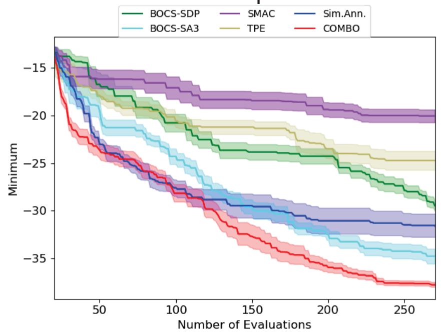
However, we want to stress out that the dynamics of this problem is far more complex than the one in the contamination control case. First, it has 25 variables and each variable has 5 categories. More importantly, the price and effectiveness of pesticides are dynamically adjusted depending on the previously made choice.

### B.4.3 Weighted maximum satisfiability(wMaxSAT)

Satisfiability problem is the one of the most important and general form of combinatorial optimization problems. SAT solver competition is held in Satisfiability conference every year.[36] Due to the resemblance between combinatorial optimizations and weighted Maximum

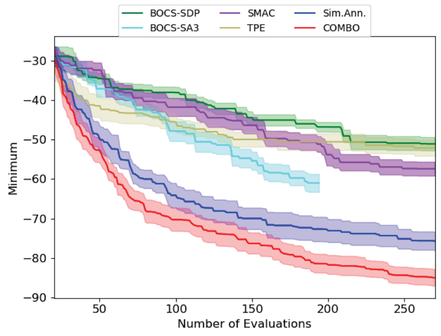
36 http://satisfiability.org/, http://sat2018.azurewebsites.net/competitions/

satisfiability problems, in which the goal is to find boolean values that maximize the combined weights of satisfied clauses, we optimize certain benchmarks taken from Maximum atisfiability(MaxSAT) Competition 2018. We took randomly 3 benchmarks of weighted maximum satisfiability problems with no hard clause with the number of variables not exceeding 100.[37] The weights are normalized by mean subtraction and standard deviation division and evaluations are negated to be minimization problems.



| Method | 28 |
|---|---|
| SMAC | -20.0530±0.6735 |
| TPE | -25.2010±0.8750 |
| SA | -31.8060±1.1929 |
| BOCS-SDP | -29.4865±0.5348 |
| BOCS-SA3 | -34.7915±0.7814 |
| COMBO | **-37.7960**±0.2662 |

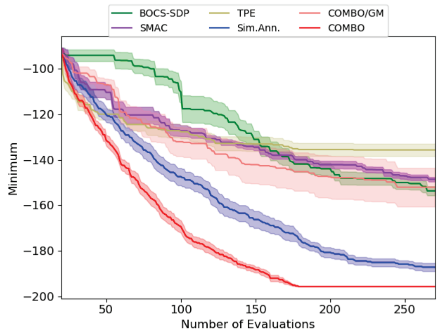
Figure B.9: wMaxSAT28



| Method | 43 |
|---|---|
| SMAC | -57.4217±1.7614 |
| TPE | -52.3856±1.9898 |
| SA | -75.7582±2.3048 |
| BOCS-SDP | -51.1265±1.6903 |
| BOCS-SA3* | -61.0186±2.2812 |
| COMBO | **-85.0155**±2.1390 |

*BOCS-SA3 was run for 168 hours but could not finish 270 evaulations.

Figure B.10: wMaxSAT43



| Method | 60 |
|---|---|
| SMAC | -148.6020±1.0135 |
| TPE | -137.2104±2.8296 |
| SA | -187.5506±1.4962 |
| BOCS-SDP | -153.6722±2.0096 |
| COMBO/GM | -152.0745±8.5167 |
| COMBO | **-195.6527**±0.0000 |

Figure B.11: wMaxSAT60

37 https://maxsat-evaluations.github.io/2018/benchmarks.html maxcut-johnson8-2-4.clq.wcnf (28 variables), maxcut-hamming8-2.clq.wcnf (43 variables), frb-frb10-6-4.wcnf (60 variables)

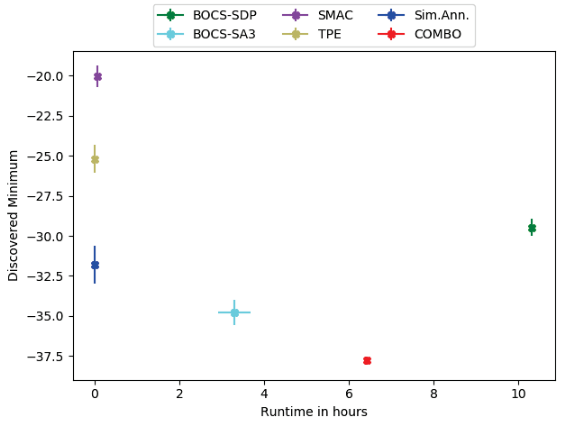
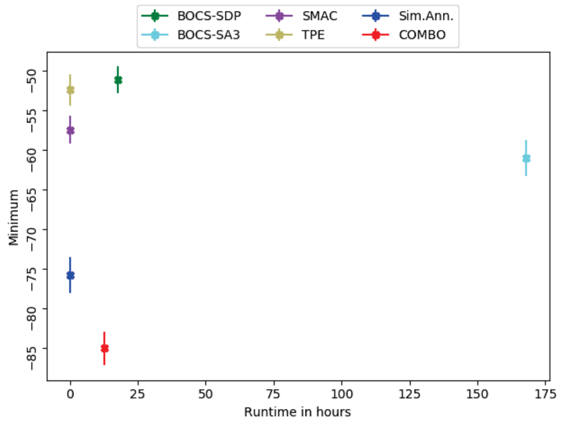


Figure B.12: Runtime VS. Minimum on wMaxSAT28



*BOCS-SA3 was run for 168 hours but could not finish 270 evaulations.

Figure B.13: Runtime VS. Minimum on wMaxSAT4

### B.4.4 Neural architecture search(NAS)

SEARCH SPACE In our architecture search problem, the cell consists of one input state(**IN**), one output state(**OUT**) and five hidden states(**H1**~**H5**). The connectivity between 7 states are specified as in the left of Tab. B.2 where it can be read that (**IN**→**H1**) and (**IN**→**H5**) from the first row. Input and output states are identity maps. The computation type of each of 5 hidden states are determined by combination of 2 binary choices as in the right of Tab. B.2.

In total, our search space consists of 31 binary variables.[38]

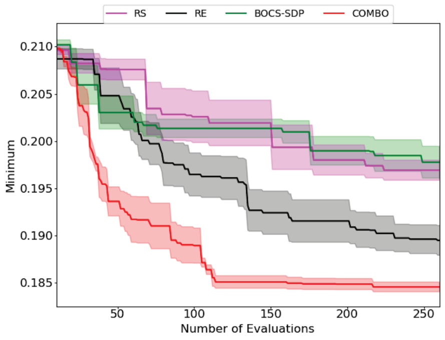
COMPARISON TO NASNET SEARCH SPACE

38 We design a binary search space for NAS so that to also compare with BOCS. COMBO is not restricted to binary choices for NAS, however.

Table B.2: Connectivity and Computation type.

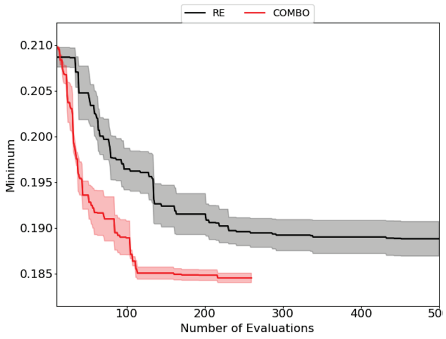
| | **IN** | **H1** | **H2** | **H3** | **H4** | **H5** | **OUT** |
|---|---|---|---|---|---|---|---|
| **IN** | - | O | X | X | X | O | X |
| **H1** | - | - | X | O | X | X | O |
| **H2** | - | - | - | X | O | X | X |
| **H3** | - | - | - | - | X | O | X |
| **H4** | - | - | - | - | - | O | O |
| **H5** | - | - | - | - | - | - | X |
| **OUT** | - | - | - | - | - | - | - |

| | **MaxPool** | **Conv** |
|---|---|---|
| **Small** | Id ≡ MaxPool(1×1) | Conv(3×3) |
| **Large** | MaxPool(3×3) | Conv(5×5) |



| Method | NAS |
|---|---|
| RS | 0.1969±0.0011 |
| BOCS – SDP | 0.1978±0.0017 |
| RE | 0.1895±0.0016 |
| COMBO | **0.1846**±0.0005 |

Figure B.14: Neural architecture search experiment.



| Method(#eval) | NAS |
|---|---|
| RE(260) | 0.1895±0.0016 |
| RE(500) | 0.1888±0.0019 |
| COMBO(260) | **0.1846**±0.0005 |

Figure B.15: Neural architecture search experiment with additional evaluations for RE (up to 500 evaluations).

| Binary | | NASNet |
|---|---|---|
| Yes | Invalid Architecture | No |
| Not fixed | The number of inputs to each state | 2 |
| 4 | The number of computation type of states | 13 |

REGULARIZED EVOLUTION HYPERPARAMETERS In evolutionary search algorithms, the choice of mutation is critical to the performance. Since our binary search space is different from NASNet search space where Regulairzed Evolution(RE) was originally applied, we modify mutation steps. All possible mutations proposed in (Real et al., 2019) can be represented as simple binary flipping in binary search space. In binary search space, some binary variables are about computation type and others are about connectivity. Thus uniform-randomly choosing binary variable to flip can mutate computation type or connectivity. Since binary search space is redundant we did not explicitly include identity mutation (not mutating). Since evolutionary search algorithms are believed to be less sample efficient than BO, we gave an advantage to RE by only allowing valid architectures in mutation steps.

On other crucial hyperparameters, population size $P$ and sample size $S$, motivated by the best value used in (Real et al., 2019), $P = 100$, $S = 25$. We set our $P$ and $S$ to have similar ratio as the one originally proposed. Since, we assumed less number of evaluations(260, 500) compared to 20000 in (Real et al., 2019), we reduced $P$ and $S$. In NAS on binary search space, we used $P = 50$ and $S = 15$.

# C MIXED VARIABLE BAYESIAN OPTIMIZATION WITH FREQUENCY MODULATED KERNELS

## C.1 PROOFS

### C.1.1 Positive Definiteness of FM kernels

For a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with graph Laplacian $L(\mathcal{G}) = U\Lambda U^T$. The frequency modulating kernel is defined as

$$k((\mathbf{c}, v), (\mathbf{c}', v') \| \boldsymbol{\theta}, \beta) = [\sum_{i=1}^{\|\mathcal{V}\|} [U]_{:,i} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta) [U]_{:,i}]_{v,v'} \quad \text{(C.1)}$$

where $[U]_{:,i}$ are eigenvectors of $L(\mathcal{G})$ which are columns of $U$ and $\lambda_i = [\Lambda]_{ii}$ are corresponding eigenvalues. $\mathbf{c}$ and $\mathbf{c}'$ are continuous variables in $\mathbb{R}^{D_c}$, $\boldsymbol{\theta} \in \mathbb{R}^{D_c}$ is a kernel parameter similar to the lengthscales in the RBF kernel. $\beta \in \mathbb{R}$ is a kernel parameter from kernels derived from the graph Laplacian.

**Theorem C.1.** *If $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta)$ defines a positive definite kernel on $(\mathbf{c}, \mathbf{c}') \in \mathbb{R}^{D_c} \times \mathbb{R}^{D_c}$, then a FreMod kernel defined with such $f$ is positive definite jointly on $(\mathbf{c}, v)$.*

*Proof.*

$$\begin{aligned} k((\mathbf{c}, v), (\mathbf{c}', v') \| \boldsymbol{\theta}, \beta) &= \Big[ \sum_{i=1}^{\|\mathcal{V}\|} [U]_{:,i} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta) [U]_{:,i} \Big]_{v,v'} \\ &= \sum_{i=1}^{\|\mathcal{V}\|} [U]_{v,i} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta) [U]_{v',i} \end{aligned}$$

Since a sum of positive definite (PD) kernels is PD, we prove PD of frequency modulating kernels by showing that $k_i((\mathbf{c}, v), (\mathbf{c}', v') \| \boldsymbol{\theta}, \beta) = [U]_{v,i} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta) [U]_{v',i}$ is PD.

Let us consider $\mathbf{a} \in \mathbb{R}^S$, $\mathcal{D} = \{(\mathbf{c}_1, v_1), \cdots, (\mathbf{c}_S, v_S)\}$, then

$$\begin{aligned} &\mathbf{a}^T \begin{bmatrix} [U]_{v_1,i} f(\lambda_i, \| \mathbf{c}_1 - \mathbf{c}_1 \|_{\boldsymbol{\theta}} | \beta) [U]_{v_1,i} & \cdots & [U]_{v_1,i} f(\lambda_i, \| \mathbf{c}_1 - \mathbf{c}_S \|_{\boldsymbol{\theta}} | \beta) [U]_{v_S,i} \\ \vdots & \cdots & \vdots \\ [U]_{v_S,i} f(\lambda_i, \| \mathbf{c}_S - \mathbf{c}_1 \|_{\boldsymbol{\theta}} | \beta) [U]_{v_1,i} & \cdots & [U]_{v_S,i} f(\lambda_i, \| \mathbf{c}_S - \mathbf{c}_S \|_{\boldsymbol{\theta}} | \beta) [U]_{v_S,i} \end{bmatrix} \mathbf{a} \\ &= (\mathbf{a} \circ [U]_{:,i})^T \begin{bmatrix} f(\lambda_i, \| \mathbf{c}_1 - \mathbf{c}_1 \|_{\boldsymbol{\theta}} \| \beta) & \cdots & f(\beta\lambda_i, \| \mathbf{c}_1 - \mathbf{c}_S \|_{\boldsymbol{\theta}} | \beta) \\ \vdots & \cdots & \vdots \\ f(\lambda_i, \| \mathbf{c}_S - \mathbf{c}_1 \|_{\boldsymbol{\theta}} \| \beta) & \cdots & f(\beta\lambda_i, \| \mathbf{c}_S - \mathbf{c}_S \|_{\boldsymbol{\theta}} | \beta) \end{bmatrix} (\mathbf{a} \circ [U]_{:,i}) \end{aligned} \quad \text{(C.2)}$$

where $\circ$ is elementwise product and $[U]_{:,i} = [[U]_{\nu_1,i}, \cdots, [U]_{\nu_S,i}]^T$.

By letting $\mathbf{a}' = \mathbf{a} \circ [U_i]_{\pi_i(\nu_:),n}$, since $f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta)$ is PD, we show that $k_i((\mathbf{c}, \nu), (\mathbf{c}', \nu') \| \boldsymbol{\theta}, \beta) = u_{i,\nu} f(\lambda_i, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} \| \beta) u_{i,\nu'}$ is PD. □

### C.1.2 Frequency Modulation Principle

**Theorem C.2.** *For a connected and undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with non-negative weights on edges, define a kernel $k(\nu, \nu') = [Uf(\Lambda)U^T]_{\nu,\nu'}$ where $U$ and $\Lambda$ are eigenvectors and eigenvalues of the graph Laplacian $L(\mathcal{G}) = U\Lambda U^T$. If $f$ is any non-negative and strictly decreasing convex function on $[0, \infty)$, then $K(\nu, \nu') \geqslant 0$ for all $\nu, \nu' \in \mathcal{V}$.*

*Proof.* We show that, for a connected weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$,

$$\min_{\nu,\nu'} k_{\mathcal{G}}(\nu, \nu') = \min_{p,q=1,\cdots,D} [Uf(\Lambda)U^T]_{p,q} \geqslant 0 \quad \text{where} \quad L(\mathcal{G}) = U\Lambda U^T.$$

For a connected weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the graph Laplacian $L(\mathcal{G}) = U\Lambda U^T$ has exactly one zero eigenvalue $0 = \lambda_1 < \lambda_2 \leqslant \cdots \leqslant \lambda_D$ where $\lambda_i = [\Lambda]_{i,i}$ and the corresponding eigenvector is $[U]_{1,\cdot} = \frac{1}{\sqrt{D}} \mathbb{1}$ and $|\mathcal{V}| = D$. Note that $U^T U = UU^T = I$. Importantly, from the definition of the graph Laplacian, $[U\Lambda U^T]_{p,q} \leqslant 0$ when $p \neq q$.

For given eigenvalues $0 = \lambda_1 < \lambda_2 \leqslant \cdots \leqslant \lambda_D$, we solve the following minimization problem

$$\min_{[U]_{p,i},[U]_{q,i}} \frac{f(0)}{D} + \sum_{i=2}^{D} f(\lambda_i)[U]_{p,i}[U]_{q,i} \tag{C.3}$$

with the constraints

$$\sum_{i=2}^{D} \lambda_i [U]_{p,i}[U]_{q,i} \leqslant 0 (p \neq q)$$
$$\sum_{i=2}^{D} [U]_{p,i}^2 = \sum_{i=2}^{D} [U]_{q,i}^2 = 1 - \frac{1}{D}$$
$$\sum_{i=2}^{D} [U]_{p,i}[U]_{q,i} = -\frac{1}{D} (p \neq q)$$

When $p = q$, Eq. (C.3) is nonnegative since $f(\cdot) \geqslant 0$. From now on, we assume $p \neq q$.

With $\eta \geqslant 0$, Lagrange multiplier is given as

$$\begin{aligned}
L_{KKT}([U]_{p,i},[U]_{q,i},\eta,a,b,c) &= \frac{f(0)}{D} + \sum_{i=2}^{D} f(\lambda_i)[U]_{p,i}[U]_{q,i} \\
&+ \eta\Big(\sum_{i=2}^{D} \lambda_i [U]_{p,i}[U]_{q,i}\Big) + a\Big(\sum_{i=2}^{D}[U]_{p,i}^2 - (1-\frac{1}{D})\Big) \\
&+ b\Big(\sum_{i=2}^{D}[U]_{q,i}^2 - (1-\frac{1}{D})\Big) + c\Big(\sum_{i=2}^{D}[U]_{p,i}[U]_{q,i} + \frac{1}{D}\Big).
\end{aligned}$$

From the stationary conditions $\frac{\partial L_{KKT}}{\partial [U]_{p,i}} = f(\lambda_i)[U]_{q,i} + \eta\lambda_i[U]_{q,i} + c[U]_{q,i} + 2a[U]_{p,i} = 0$ and $\frac{\partial L_{KKT}}{\partial [U]_{q,i}} = f(\lambda_i)[U]_{p,i} + \eta\lambda_i[U]_{p,i} + c[U]_{p,i} + 2b[U]_{q,i} = 0$, we have

$$\begin{aligned}
(f(\lambda_i) + \eta\lambda_i + c)[U]_{q,i} &= -2a[U]_{p,i} \\
(f(\lambda_i) + \eta\lambda_i + c)[U]_{p,i} &= -2b[U]_{q,i}
\end{aligned} \tag{C.4}$$

and

$$\begin{aligned}
((f(\lambda_i) + \eta\lambda_i + c)^2 - 4ab)[U]_{q,i} &= 0 \\
((f(\lambda_i) + \eta\lambda_i + c)^2 - 4ab)[U]_{p,i} &= 0
\end{aligned}$$

using $a = b$ that is derived from $\sum_{i=2}^{D} \frac{\partial L_{KKT}}{\partial [U]_{p,i}} \cdot [U]_{p,i} = \sum_{i=2}^{D} \frac{\partial L_{KKT}}{\partial [U]_{q,i}} \cdot [U]_{q,i} = 0$.

If $i \in \{i | (f(\lambda_i) + \eta\lambda_i + c)^2 - 4ab \neq 0\}$, we have $[U]_{p,i} = [U]_{q,i} = 0$. On the other hand, if $(f(\lambda_i) + \eta\lambda_i + c)^2 - 4ab = 0$, then we have $f(\lambda_i) + \eta\lambda_i + c = -2a$ or $f(\lambda_i) + \eta\lambda_i + c = 2a$ because $a = b$.

With the three index sets defined below and Eq. (C.1.2), we have

$$\begin{aligned}
I_0 &= \{i | (f(\lambda_i) + \eta\lambda_i + c)^2 - 4a^2 \neq 0\} & i \in I_0 &\Rightarrow [U]_{p,i} = [U]_{q,i} = 0 \\
I_+ &= \{i | f(\lambda_i) + \eta\lambda_i + c + 2a = 0\} - \{1\} & i \in I_+ &\Rightarrow [U]_{p,i} = [U]_{q,i} \\
I_- &= \{i | f(\lambda_i) + \eta\lambda_i + c - 2a = 0\} & i \in I_- &\Rightarrow [U]_{p,i} = -[U]_{q,i}
\end{aligned}$$

With these conditions, the constraints can be expressed as

$$\begin{gathered}
\sum_{i_+ \in I_+} \lambda_{i_+}[U]_{p,i}^2 - \sum_{i_- \in I_-} \lambda_{i_-}[U]_{p,i}^2 \leqslant 0, \\
\sum_{i_+ \in I_+} [U]_{p,i_+}^2 = \frac{1}{2} - \frac{1}{D}, \quad \sum_{i_- \in I_-} [U]_{p,i_-}^2 = \frac{1}{2}
\end{gathered}$$

We divide cases according to the number of solutions $g(\lambda) = f(\lambda) + \eta\lambda$ can have. *i)* $f(\lambda) + \eta\lambda$ can have at most one solution, *ii)* $f(\lambda) + \eta\lambda$ may have two solutions. Note that $g(\lambda)$ is convex as sum of two convex functions. Since a convex function can have at most two zeros unless it is constantly zero, these two cases are exhaustive. When $\eta = 0$, $f(\lambda)$ is strictly decreasing function and, thus $g(\lambda)$ has at most one solution. Also, when $\eta \geqslant -f'(0) = \max_\lambda -f'(\lambda)$, $f'(\lambda) + \eta$ is positive except for $\lambda = 0$ and $g(\lambda)$ has at most one solution.

CASE *I)* $f(\lambda) + \eta\lambda$ can have at most one solution. ($\eta = 0$ or $\eta \geqslant -f'(0) = \max_\lambda -f'(\lambda)$)

Let us denote $\lambda^E$ the unique solution of $f(\lambda_i) + \eta\lambda_i + c + 2a = 0$ and $\lambda^N$ the unique of $f(\lambda_i) + \eta\lambda_i + c - 2a = 0$.

Therefore $\lambda_{i_+} = \lambda^E$, $\forall i_+ \in I_+$ and $\lambda_{i_-} = \lambda^N$, $\forall i_- \in I_-$. The minimization objective becomes

$$
\begin{aligned}
&\frac{f(0)}{D} + \sum_{i=2}^{D} f(\lambda_i)[U]_{p,i}[U]_{q,i} \\
&= \frac{f(0)}{D} + f(\lambda^E) \sum_{i_+ \in I_+} [U]_{p,i}^2 - f(\lambda_N) \sum_{i_- \in I_-} [U]_{p,i}^2 \\
&= \frac{f(0)}{D} + \Big(\frac{1}{2} - \frac{1}{D}\Big) f(\lambda^E) - \frac{1}{2} f(\lambda^N) =: g_1(\lambda^E, \lambda^N) \qquad \text{(C.5)}
\end{aligned}
$$

The inequality constraint becomes

$$
\begin{aligned}
\sum_{i=2}^{D} \lambda_i [U]_{p,i}[U]_{q,i} &= \frac{f(0)}{D} + \lambda^E \sum_{i_+ \in I_+} [U]_{p,i_+}^2 - \lambda_N \sum_{i_- \in I_-} [U]_{p,i_-}^2 \\
&= \Big(\frac{1}{2} - \frac{1}{D}\Big)\lambda^E - \frac{1}{2}\lambda^N \leqslant 0.
\end{aligned}
$$

Since $\lambda^E, \lambda^N \in \{\lambda_2, \cdots, \lambda_D\}$, there is maximum value with respect to the choice of $\lambda^E$ and $\lambda^N$. We consider continuous relaxation of the minimization problem with respect to $\lambda^E$ and $\lambda^N$. By showing that the objective is nonnegative when $\lambda^E \geqslant 0$ and $\lambda^N \geqslant 0$, we prove our claim. When we consider continuous optimization over $\lambda^E$ and $\lambda^N$, the minimum is obtained when the inequality constraint becomes equality constraints. If $\left(\frac{1}{2} - \frac{1}{D}\right)\lambda^E - \frac{1}{2}\lambda^N < 0$, then $\left(\frac{1}{2} - \frac{1}{D}\right)(\lambda^E + \delta) - \frac{1}{2}\lambda^N = 0$ for some $\delta > 0$. Since $f$ is decreasing, $f(\lambda^E)$ is decreased to $f(\lambda^E + \delta)$ and thus Eq. C.5 further decreases by replacing $\lambda^E$ with $\lambda^E + \delta$. Thus, the minimum is obtained when the inequality constraint is equality. When $\eta > 0$, the inequality constraint automatically becomes an equality constraint by the slackness condition of the Karush-Kuhn-Tucker conditions.

The minimum is obtained when $\left(\frac{1}{2} - \frac{1}{D}\right)\lambda^E - \frac{1}{2}\lambda^N = 0$. $h_1(\lambda^E) = g_1(\lambda^E, (1 - 2/D)\lambda^E)$ is increasing with respect to $\lambda^E$ from the convexity of $f$ and $\lim_{\lambda_E \to 0} h_1(\lambda^E) = 0$. Thus, the minimum of Eq. C.5 is bounded below by 0.

CASE *II)* $f(\lambda) + \eta\lambda$ may have two solutions. ($0 < \eta < -f'(0) = \max_\lambda -f'(\lambda)$)

By the slackness condition, the inequality constraint becomes an equality constraint. Since $f(\lambda) + \eta\lambda$ is convex, it has at most two solutions. Let us denote $\lambda_1^E < \lambda_2^E$ two solutions of $f(\lambda) + \eta\lambda + c + 2a = 0$ and $\lambda_1^N < \lambda_2^N$ two solutions of $f(\lambda) + \eta\lambda + c - 2a = 0$. Then

$$\begin{aligned} f(\lambda_1^E) + \eta\lambda_1^E + c + 2a &= 0 \qquad & f(\lambda_2^E) + \eta\lambda_2^E + c + 2a &= 0 \\ f(\lambda_1^N) + \eta\lambda_1^N + c - 2a &= 0 \qquad & f(\lambda_2^N) + \eta\lambda_2^N + c - 2a &= 0 \end{aligned}$$

The objective becomes

$$\begin{aligned} &\frac{f(0)}{D} + f(\lambda_1^E) \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_1^E} [U]_{p,i_+}^2 + f(\lambda_2^E) \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_2^E} [U]_{p,i_+}^2 \\ &\quad - f(\lambda_1^N) \sum_{i_- \in I_- : \lambda_{i_-} = \lambda_1^N} [U]_{p,i_-}^2 - f(\lambda_2^N) \sum_{i_- \in I_- : \lambda_{i_-} = \lambda_2^N} [U]_{p,i_-}^2 \end{aligned}$$

with the constraints

$$\begin{aligned} \text{C1:} \quad & \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_1^E} [U]_{p,i_+}^2 + \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_2^E} [U]_{p,i_+}^2 = \frac{1}{2} - \frac{1}{D} \\ \text{C2:} \quad & \sum_{i_- \in I_- : \lambda_i = \lambda_1^N} [U]_{p,i_-}^2 + \sum_{i_- \in I_- : \lambda_i = \lambda_2^N} [U]_{p,i_-}^2 = \frac{1}{2} \\ \text{C3:} \quad & \lambda_1^E \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_1^E} [U]_{p,i_+}^2 + \lambda_2^E \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_2^E} [U]_{p,i_+}^2 \\ & -\lambda_1^N \sum_{i_- \in I_- : \lambda_{i_-} = \lambda_1^N} [U]_{p,i_-}^2 - \lambda_2^N \sum_{i_- \in I_- : \lambda_{i_-} = \lambda_2^N} [U]_{p,i_-}^2 = 0. \end{aligned}$$

Let

$$\begin{aligned} A^E &= \sum_{i_+ \in I_+ : \lambda_{i_+} = \lambda_1^E} [U]_{p,i_+}^2 \in [0, \frac{1}{2} - \frac{1}{D}] \\ A^N &= \sum_{i_- \in I_- : \lambda_{i_-} = \lambda_1^N} [U]_{p,i_-}^2 \in [0, \frac{1}{2}]. \end{aligned}$$

Then the objective becomes

$$\frac{f(0)}{D} + f(\lambda_1^E)A^E + f(\lambda_2^E)(\frac{1}{2} - \frac{1}{D} - A^E) - f(\lambda_1^N)A^N - f(\lambda_2^N)(\frac{1}{2} - A^N).$$

Taking derivatives

$$\frac{\partial}{\partial A^E} \Rightarrow f(\lambda_1^E) - f(\lambda_2^E) > 0 \quad \text{and} \quad \frac{\partial}{\partial A^N} \Rightarrow -f(\lambda_1^N) + f(\lambda_2^N) < 0.$$

Thus the minimum is obtained at the boundary point where $A^E = 0$ and $A^N = \frac{1}{2}$ which falls back to Case *i)* whose minimum is bounded below by zero. □

*Remark* C.1. Thm. C.2 holds for weighted undirected graphs, that is, for any arbitrary graph with arbitrary symmetric nonnegative edge weights.

*Remark* C.2. Note that in numerical simulations, you may observe small negative values ($\approx 10^{-7}$) due to numerical instability.

*Remark* C.3. In numerical simulations, the convexity condition does not appear to be necessary for complete graphs where $\max_{p \neq q} [L(\mathcal{G})]_{p,q} < -\epsilon$ for some $\epsilon > 0$. For complete graphs, the convexity condition may be relaxed, at least, in a stochastic sense.

**Corollary C.3.** *The random walk kernel derived from normalized Laplacian Smola and R. Kondor, 2003 and the diffusion kernels R. I. Kondor and Lafferty, 2002, the ARD diffusion kernel Changyong Oh, Tomczak, et al., 2019 and the regularized Laplacian kernel Smola and R. Kondor, 2003 derived from normalized and unnormalized Laplacian are all positive valued kernels.*

*Proof.* The condition that off-diagonal entries are nonpositive holds for both normalized and unnormalized graph Laplacian. Therefore for normalized graph Laplacian, the proof in the above theorem can be applied without modification. The positivity of kernel value also holds for kernels derived from normalized Laplacian as long as it satisfies the conditions in Thm.C.2. $\square$

*Remark* C.4. In numerical simulations with nonconvex functions and arbitrary connected and weighted undirected graphs, negative values easily occur. For example, the inverse cosine kernel (Smola and R. Kondor, 2003) does not satisfies the convexity condition and has negative values.

### C.1.3 Frequency Modulating functions

In this section, we first review the definition of conditionally negative definite (CND) and relations between positive definite (PD). Utilizing relations between PD and CND and the properties of PD and CND, we provide an example of a flexible family of frequency modulating functions.

**Definition C.1** (3.1.1 (Berg et al., 1984))**.** A symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a conditionally negative definite (CND) kernel if $\forall n \in \mathbb{N}, x_1, \cdots, x_n \in \mathcal{X}$ $a_1, \cdots, a_n \in \mathbb{R}$ such that $\sum_{i=1}^{n} a_i = 0$

$$\sum_{i,j=1}^{n} a_i k(x_i, x_j) a_j \leqslant 0$$

Please note that CND requires the condition $\sum_{i=1}^{n} a_i = 0$.

**Theorem C.4** (3.2.2 (Berg et al., 1984))**.** $K(x, x')$ *is conditionally negative definite if and only if* $e^{-tK(x,x')}$ *is positive definite for all* $t > 0$.

As mentioned in p.75 Berg et al., 1984, from Thm. C.4, we have

**Theorem C.5.** $K(x,x')$ *is conditionally negative definite and* $K(x,x')) \geqslant 0$ *if and only if* $(t+K(x,x'))^{-1}$ *is positive definite for all* $t > 0$.

**Theorem C.6** (3.2.10 (Berg et al., 1984))**.** *If* $K(x,x')$ *is conditionally negative definite and* $K(x,x) \geqslant 0$*, then* $(K(x,x'))^{a}$ *for* $0 < a < 1$ *and* $\log K(x,x')$ *are conditionally negative definite.*

**Theorem C.7** (3.2.13 (Berg et al., 1984))**.** $K(x,x') = \|x-x'\|^{p}$ *is conditionally negative definite for all* $0 < p \leqslant 2$.

Using above theorems, we provide a quite flexible family of frequency modulating functions

**Proposition C.8.** *For* $S \in (0,\infty)$*, a finite measure* $\mu$ *on* $[0,S]$ *and* $\mu$*-measurable* $\tau : [0,S] \to [0,2]$ *and* $\rho : [0,S] \to \mathbb{N}$,

$$f(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \int_0^S \frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{\tau(s)})^{\rho(s)}} \mu(ds)$$

*is a frequency modulating function.*

*Proof.* First we show that

$$f^{p,t}(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{t})^{p}}$$

is a frequency modulating function for $t \in (0,2]$ and $p \in \mathbb{N}$.

Property **FM-P1** on $f^{p,t}$) $f^{p,t}(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ is positive valued and decreasing with respect to $\lambda$.

Property **FM-P2** on $f^{p,t}$) $\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}$ is conditionally negative definite by Thm.C.7 Then by Thm.C.5, $\frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{t})}$ is positive definite with respect to $\mathbf{c}$ and $\mathbf{c}'$. Since the product of positive definite kernels is positive definite, $f^{p,t}(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ is positive definite.

Property **FM-P3** on $f^{p,t}$) Let $h^{p,t} = f^{p,t}(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) - f^{p,t}(\lambda, \|\tilde{\mathbf{c}}-\tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}} | \alpha, \beta)$, then

$$\begin{aligned}
h^{p,t}_{\lambda} &= \frac{\partial h^{p,t}}{\partial \lambda} \\
&= -p\beta\Big(\frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{t})^{p+1}} - \frac{1}{(1+\beta\lambda+\alpha\|\tilde{\mathbf{c}}-\tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}}^{t})^{p+1}}\Big) \\
h^{p,t}_{\lambda\lambda} &= \frac{\partial^2 h^{p,t}}{\partial \lambda^2} \\
&= p(p+1)\beta^2\Big(\frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{t})^{p+2}} - \frac{1}{(1+\beta\lambda+\alpha\|\tilde{\mathbf{c}}-\tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}}^{t})^{p+2}}\Big)
\end{aligned}$$

For $\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} < \|\tilde{\mathbf{c}}-\tilde{\mathbf{c}}'\|_{\boldsymbol{\theta}}$, $h > 0$, $h_{\lambda} < 0$ and $h_{\lambda\lambda} > 0$, therefore this satisfies the frequency modulation principle.

Now we show that

$$f(\lambda, \| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \int_0^S \frac{1}{(1+\beta\lambda+\alpha\| \mathbf{c}-\mathbf{c}' \|_{\boldsymbol{\theta}}^{\tau(s)})^{\rho(s)}} \mu(ds)$$

satisfies all 3 conditions.

Property **FM-P1**) Trivial from the definition.

Property **FM-P2**) Since a measurable function can be approximated by simple functions (Folland, 1999), we approximate $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ with following increasing sequence

$$f_n(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) = \sum_{i=1}^{2^n} \sum_{j=1}^{n} \frac{\mu(A_{i,j})}{(1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^{\frac{i-1}{2^n}2})^j}$$
$$\text{where} \quad A_{i,j} = \left\{ s | \frac{i-1}{2^n} 2 < \rho(s) \leqslant \frac{i}{2^n} 2, \tau(s) = j \right\}$$

Each summand $\mu(A_{i,j})/(1 + \beta\lambda + \alpha \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}}^{\frac{i-1}{2^n}2})^j$ is positive definite as shown above and sum of positive definite kernels is positive definite. Therefore, $f_n(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ is positive definite. Since the pointwise limit of positive definite kernels is a kernel (Fukumizu, 2010), we show that $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ is positive definite.

Property **FM-P3**) If we show that $\frac{\partial}{\partial \lambda}$ and $\int \mu(ds)$ are interchangeable, from the Condition #3 on $f_{p,t}$, we show that $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ satisfies the frequency modulating principle.

Let $h = f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta) - f(\lambda, \| \tilde{\mathbf{c}} - \tilde{\mathbf{c}}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$. There is a constant $A > 0$ such that

$$\left| \frac{h^{\tau(s),\rho(s)}(\lambda + \delta) - h^{\tau(s),\rho(s)}(\lambda)}{\delta} \right| < \left| \frac{\partial h^{\tau(s),\rho(s)}}{\partial \lambda} \right| + A < \left| \frac{\partial h^{0,1}}{\partial \lambda} \right| + A$$

For a finite measure, $\left| \frac{\partial h^{0,1}}{\partial \lambda} \right| + A$ is integrable. Therefore, $\frac{\partial}{\partial \lambda}$ and $\int \mu(ds)$ are interchangeable by dominated convergence theorem (Folland, 1999). With the same argument, $\frac{\partial^2}{\partial \lambda^2}$ and $\int \mu(ds)$ are interchangeable.

Now, we have

$$h_\lambda = \frac{\partial h}{\partial \lambda} = \int_0^S \frac{\partial h^{\tau(s),\rho(s)}}{\partial \lambda} \mu(ds)$$
$$h_{\lambda\lambda} = \frac{\partial^2 h}{\partial \lambda^2} = \int_0^S \frac{\partial^2 h^{\tau(s),\rho(s)}}{\partial \lambda^2} \mu(ds)$$

From the Condition #3 on $f^{p,t}$, $h_\lambda < 0$ and $h_{\lambda\lambda} > 0$ follow and thus we show that $f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ satisfies the frequency modulating principle.

$f(\lambda, \| \mathbf{c} - \mathbf{c}' \|_{\boldsymbol{\theta}} | \alpha, \beta)$ is a frequency modulating function. □

**Proposition C.9.** *If* $k_{\mathcal{H}} : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ *on a RKHS* $\mathcal{H}$ *is bounded above by* $u > 0$*, then for any* $\delta > 0$

$$f(\lambda, k_{\mathcal{H}}(h, h') | \alpha, \beta) = \frac{1}{\delta + u + \beta\lambda - k_{\mathcal{H}}(h, h')}$$

*is positive definite on* $(h, h') \in \mathcal{H} \times \mathcal{H}$.

*Proof.* The negation of a positive definite kernel is conditionally negative definite by Appx. Def. C.1. Also, by definition, a constant plus a conditionally negative definite kernel is conditionally negative definite. Therefore, $u - k_{\mathcal{H}}(h, h')$ is conditionally negative definite.

Using Appx. Thm.C.5, we show that $1/(\delta + u + \beta\lambda - k_{\mathcal{H}}(h, h'))$ is positive definite on $(h, h') \in \mathcal{H} \times \mathcal{H}$. □

## C.2 IMPLEMENTATION DETAILS

In this section, we provide the details of each component of BO pipeline, the surrogate model and how it is fitted to evaluation data, the acquisition function and how it is optimized. We also provide each experiment specific details including the search spaces, evaluation detail, run time analysis and etc. The code used for the experiments will be released upon acceptance.

### C.2.1 Acquisition Function Optimization

We use Expected Improvement (EI) acquisition function (Donald, 1998). Since, in mixed variable BO, acquisition function optimization is another mixed variable optimization task, we need a procedure to perform an optimization of acquisition functions on mixed variables.

ACQUISITION FUNCTION OPTIMIZATION Similar to (Daxberger et al., 2021), we alternatively call continuous optimizer and discrete optimizer, which is similar to coordinate-wise ascent, and, in this case, it is so-called type-wise ascent. For continuous variables, we use L-BFGS-B (Zhu et al., 1997) and for discrete variables, we use hill climbing (Skiena, 1998). Since the discrete part of the search space is represented by graphs, hill climbing is amount to greedy ascent in neighborhood. We alternate one discrete update using hill climbing call and one continuous update by calling *scipy.optimize.minimize(method= "L-BFGS-B", maxiter=1)*.

SPRAY POINTS Acquisition functions are highly multi-modal and thus initial points with which the optimization of acquisition functions starts have an impact on exploration-exploitation trade-off. In order to encourage exploitation, spray points (Garnett et al., 2010; ChangYong Oh et al., 2018; Snoek, Larochelle, et al., 2012), which are points in the neighborhood of the current optimum (e.g, optimum among the collected evaluations), has been widely used.

INITIAL POINTS FOR ACQUISITION FUNCTION OPTIMIZATION On 50 spray points and 100000 randomly sampled points, acquisition values are computed, and the highest 40 are used as initial points to start acquisition function optimization.

### C.2.2 Joint optimization of neural architecture and SGD hyperparameter

DISCRETE PART OF THE SEARCH SPACE The discrete part of the search space, $\mathcal{A}$, is modified from the NASNet search space (Zoph and Q. Le, 2017). Each block consists of 4 states $S_1, S_2, S_3, S_4$ and takes two inputs $S_{-1}, S_0$ from a previous block. For each state, two inputs are chosen from the previous states, Then two operations are chosen and the state finishes its process by summing up two results of the chosen operation For example, if two inputs $S_{-1}$, $S_2$ and two operations $OP_3^{(1)}$, $OP_3^{(2)}$ are chosen for $S_3$, we have $(S_{-1}, S_2) \xrightarrow{S_3} OP_3^{(1)}(S_{-1}) + OP_3^{(2)}(S_2)$.

Operations are chosen from 8 types below

- ID
- Conv$1 \times 1$
- Conv$3 \times 3$
- Conv$5 \times 5$
- Sep. Conv$3 \times 3$
- Sep. Conv$5 \times 5$
- Max Pool$3 \times 3$
- Max Pool$5 \times 5$

Two inputs for each state are chosen from states with smaller subscript(e.g $S_i$ is allowed to have $S_j$ as an input if $j < i$). By choosing $S_4$ and one of $S_1, S_2, S_3$ as outputs of the block, the configuration of a block is completed.

In ModLap, it is required to specify graphs for discrete variables. For graphs representing operation types, we use complete graphs. For graphs representing inputs of each states, we use graphs which reflect the ordering structure. In a graph representing inputs of each state, each vertex is represented by a tuple, for the graph representing inputs of $S_3$, its vertex set is $\{(-1,0), (-1,1), (-1,2), (0,1), (0,2), (1,2)\}$. For example, choosing $(-1,0)$ means $S_3$ takes $S_{-1}$(input 1 of the block) and $S_0$(input 2 of the block) as inputs of the cell and choosing $(0,2)$ means $S_3$ takes $S_0$(input 2 of the block) and $S_2$(cell 2) as inputs. There exists an edge between vertices as long as one input is shared and two distinct inputs differ by one. For example, there is an edge between $(-1,0)$ and $(-1,1)$ because $-1$ is shared and $|0-1| = 1$ and there is no edge between $(-1,0)$ and $(-1,2)$ because $|0-1| \neq 1$ even though $-1$ is shared. Note that in the graph representing inputs for $S_4$, we exclude the vertex $(-1,0)$ to avoid the identity block. For graphs representing outputs of the block, we use the path graph with 3 vertices since we restrict the output is one of $(1,4), (2,4), (3,4)$. By defining graphs corresponding variables in this way, a prior knowledge about the search space can be infused and be of help to Bayesian optimization.

CONTINUOUS PART OF THE SEARCH SPACE The space of continuous hyperparameters $\mathcal{H}$ comprises 6 continuous hyperparameters of the SGD with a learning rate scheduler: learning rate, momentum, weight decay, learning rate reduction factor, 1st reduction point ratio1 and 2nd reduction point ratio. The ranges for each hyperparameter are given in Tab. C.1.

**Table C.1**: SGD Hyperparameter Range

| SGD hyperparameter | Trans. | Range |
|---|---|---|
| Learning Rate | log | $[\log(0.001), \log(0.1)]$ |
| Momentum | · | $[0.8, 1.0]$ |
| Weight Decay | log | $[\log(10^{-6}), \log(10^{-2})]$ |
| Learning Rate Reduction Factor | · | $[0.1, 0.9]$ |
| 1st Reduction Point Ratio | · | $[0, 1]$ |
| 2nd Reduction Point Ratio | · | $[0, 1]$ |

For a given learning rate $l$, learning rate reduction factor $\gamma$, 1st reduction point ratio $r_1$ and 2nd reduction point ratio $r_2$, the scheduling detail is shown below.

**Table C.2**: Learning Rate Scheduling. In the experiment, the number of epochs $E$ is set to 25.

| Begin Epoch(<) | (⩽)End Epoch | Learning Rate |
|---|---|---|
| 0 | $E \times r_1$ | $l$ |
| $E \times r_1$ | $E \times (r_1 + (1 - r_1)r_2)$ | $l \cdot \gamma$ |
| $E \times (r_1 + (1 - r_1)r_2)$ | $E$ | $l \cdot \gamma^2$ |

EVALUATION For a given block configuration $a \in \mathcal{A}$, the model is built by stacking 3 blocks with downsampling between blocks. Note that there are two inputs and two outputs of the blocks. Therefore, the downsampling is applied separately to each output. The two outputs of the last block are concatenated after max pooling and then fed to the fully connected layer.

The model is trained with the hyperparameter $h \in \mathcal{H}$ on a half of FashionMNIST (Xiao et al., 2017) training data for 25 epochs and the validation error is computed on the rest half of training data. To reduce the high noise in validation error, the validation error is averaged over 4 validation errors from models trained with different random initialization. With the batch size of 32, each evaluation takes 12~21 minutes on a single GTX 1080 Ti depending on architectures

REGULARIZED EVOLUTION HYPERPARAMETERS RE has hyperparameters, the population size and the sample size. We set to 50 and 15, respectively, to make those similar to the optimal choice in (Changyong Oh, Tomczak, et al., 2019; Real et al., 2019). Accordingly, RE starts with a population with 50 random initial points. In each run of 4 runs, the first 10 initial points of 50 random initial points are shared with 10 initial points used in GP-BO.

Another hyperparameter is the mutation rule. In addition to the mutation of architectures used in (Real et al., 2019), for continuous variables, a randomly chosen single continuous variable is mutated by Gaussian noise with small variance. In each round, one continuous variable and one discrete variable are altered.

WALL-CLOCK RUN TIME The total run time of ModLap(200), $61.44 \pm 4.09$ hours, is sum of $9.27 \pm 2.60$ hours for BO suggestions and $52.16 \pm 1.79$ hours for evaluations. BO suggestions were run on Intel Xeon Processor E5-2630 v3 and evaluations were run on GTX 1080 Ti.

In the actual execution of RE, two different types of GPUs were used, GTX 1080 Ti(fast) and GTX 980(slow). Therefore, the evaluation time for RE is estimated by assuming that RE were also run on GTX 1080 Ti(fast) only. During the total run time of ModLap(200), $61.44 \pm 4.09$ hours, RE is estimated to collects 230 evaluations. $230 \approx 61.44/52.16 \times (200 - 10) + 10$ where 10 is adjusted because the evaluation time for 10 random initial points was not measured.

Since in both RE and BOHB, we assume zero seconds to acquire new hyperparameters and only consider times spent for evaluations, the wall-clock runtime of BOHB is estimated to be equal to wall-clock runtime of RE.

## C.3 ADDITIONAL EXPERIMENTAL RESULTS

In this section, in addition to the results reported in Sec. 5.5, we provide additional results.

On 3 synthetic problems and 2 hyperparameter optimization problems, along with the frequency modulation, we also compare other kernel combinations such as the kernel addition and the kernel product as follows.

We make following observations with this additional comparison. Firstly, ModDif which does not respect the similarity measure behavior, sometimes severely degrades BO performance. Secondly, the kernel product often performs better than the kernel addition. Thirdly, ModLap shows the equally good final results as the kernel product and finds the better solution faster than the kernel product consistently. This can be clearly shown by comparing the area above the mean curve of BO runs using different kernels. The area above the

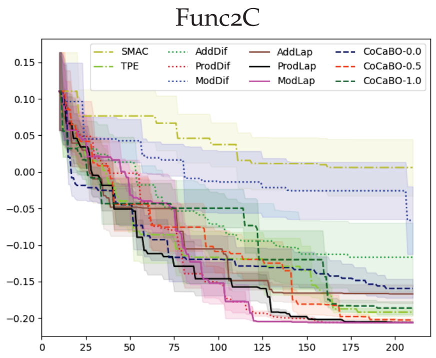
| | | |
|---|---|---|
| ProdLap : $k_{RBF} \times k_{Lap}$ | AddLap : $k_{RBF} + k_{Lap}$ | ModLap : Eq.5.5 with $f = f_{Lap}$ |
| ProdDif : $k_{RBF} \times k_{Dif}$ | AddDif : $k_{RBF} + k_{Dif}$ | ModDif : Eq.5.5 with $f = f_{Dif}$ |

where $k_{RBF}$ is the RBF kernel and

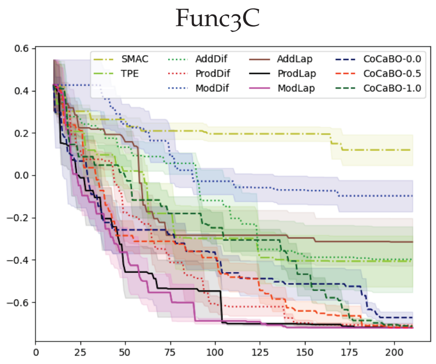
$$\begin{aligned} k_{Lap}(\mathbf{v}, \mathbf{v}') &= \prod_{p=1}^{P} \sum_{i=1}^{|\mathcal{V}_p|} [U^p]_{v_p,i} \frac{1}{1+\beta_p \lambda_i^p} [U^p]_{v'_p,i} \\ k_{Dif}(\mathbf{v}, \mathbf{v}') &= \prod_{p=1}^{P} \sum_{i=1}^{|\mathcal{V}_p|} [U^p]_{v_p,i} \exp(-\beta_p \lambda_i^p) [U^p]_{v'_p,i} \end{aligned} \tag{C.6}$$

mean curve of BO using ModLap is larger than the are above the mean curve of BO using the kernel product. Moreover, the gap between the area from ModLap and the area from kernel product increases in problems with larger search spaces. Even on the smallest search space, Func2C, ModLap lags behind the kernel product up to around 90th evaluation and outperforms after it. The benefit of ModLap modeling complex dependency among mixed variables is more prominent in higher dimension problems.

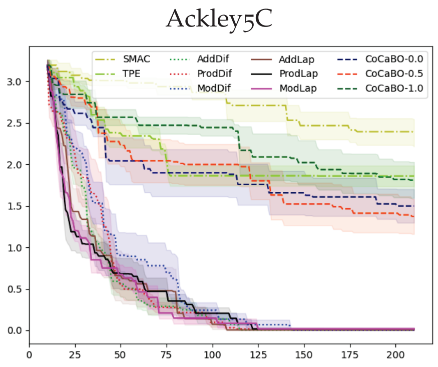
On the joint optimization of SGD hyperparameters and architecture, we show the additional result where RE and BOHB are continued 600 evaluations and 350 evaluations, respectively.



| Method | Mean±Std.Err. |
|---|---|
| SMAC | $+0.0060 \pm 0.0387$ |
| TPE | $-0.1917 \pm 0.0053$ |
| AddDif | $-0.1167 \pm 0.0472$ |
| ProdDif | $-0.2060 \pm 0.0002$ |
| ModDif | $-0.0662 \pm 0.0463$ |
| AddLap | $-0.1669 \pm 0.0127$ |
| ProdLap | $-0.2060 \pm 0.0001$ |
| ModLap | $-0.2063 \pm 0.0000$ |
| CoCaBO-0.0 | $-0.1594 \pm 0.0130$ |
| CoCaBO-0.5 | $-0.2025 \pm 0.0018$ |
| CoCaBO-1.0 | $-0.1861 \pm 0.0090$ |

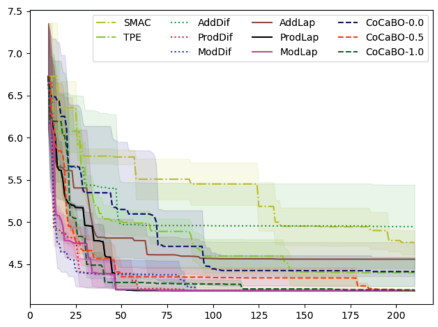


| Method | Mean±Std.Err. |
|---|---|
| SMAC | $+0.1194 \pm 0.0723$ |
| TPE | $-0.4068 \pm 0.1204$ |
| AddDif | $-0.3979 \pm 0.1555$ |
| ProdDif | $-0.7100 \pm 0.0106$ |
| ModDif | $-0.0977 \pm 0.0742$ |
| AddLap | $-0.3156 \pm 0.1125$ |
| ProdLap | $-0.7213 \pm 0.0005$ |
| ModLap | $-0.7215 \pm 0.0004$ |
| CoCaBO-0.0 | $-0.6730 \pm 0.0274$ |
| CoCaBO-0.5 | $-0.7202 \pm 0.0016$ |
| CoCaBO-1.0 | $-0.7139 \pm 0.0051$ |



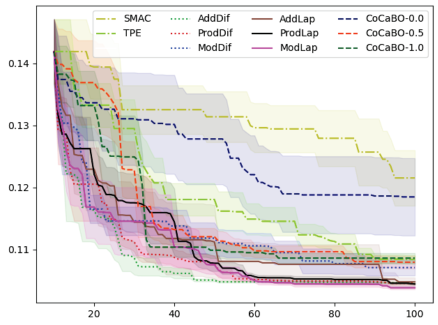
| Method | Mean±Std.Err. |
|---|---|
| SMAC | $+2.3809 \pm 0.1648$ |
| TPE | $+1.8601 \pm 0.1248$ |
| AddDif | $+0.0040 \pm 0.0015$ |
| ProdDif | $+0.0152 \pm 0.0044$ |
| ModDif | $+0.0008 \pm 0.0003$ |
| AddLap | $+0.0042 \pm 0.0018$ |
| ProdLap | $+0.0177 \pm 0.0038$ |
| ModLap | $+0.0186 \pm 0.0057$ |
| CoCaBO-0.0 | $+1.4986 \pm 0.2012$ |
| CoCaBO-0.5 | $+1.3720 \pm 0.2110$ |
| CoCaBO-1.0 | $+1.8114 \pm 0.2168$ |

SVM Hyperparameter



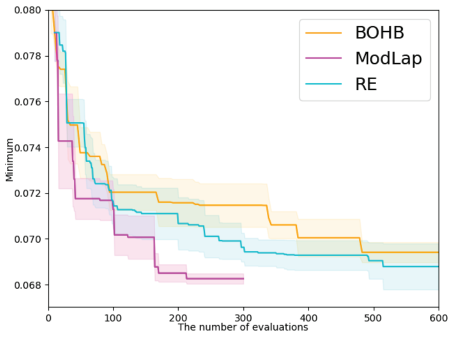
| Method | Mean±Std.Err. |
|---|---|
| SMAC | $+4.7588 \pm 0.1414$ |
| TPE | $+4.3986 \pm 0.1632$ |
| AddDif | $+4.9463 \pm 0.4960$ |
| ProdDif | $+4.1857 \pm 0.0017$ |
| ModDif | $+4.1876 \pm 0.0012$ |
| AddLap | $+4.5600 \pm 0.2014$ |
| ProdLap | $+4.1856 \pm 0.0012$ |
| ModLap | $+4.1864 \pm 0.0015$ |
| CoCaBO-0.0 | $+4.4122 \pm 0.1703$ |
| CoCaBO-0.5 | $+4.1957 \pm 0.0040$ |
| CoCaBO-1.0 | $+4.1958 \pm 0.0037$ |

XGBoost Hyperparameter



| Method | Mean±Std.Err. |
|---|---|
| SMAC | $+0.1215 \pm 0.0045$ |
| TPE | $+0.1084 \pm 0.0007$ |
| AddDif | $+0.1046 \pm 0.0001$ |
| ProdDif | $+0.1045 \pm 0.0003$ |
| ModDif | $+0.1071 \pm 0.0013$ |
| AddLap | $+0.1048 \pm 0.0007$ |
| ProdLap | $+0.1044 \pm 0.0001$ |
| ModLap | $+0.1038 \pm 0.0003$ |
| CoCaBO-0.0 | $+0.1184 \pm 0.0062$ |
| CoCaBO-0.5 | $+0.1079 \pm 0.0010$ |
| CoCaBO-1.0 | $+0.1086 \pm 0.0008$ |

Joint Optimization of SGD hyperparameters & neural architecture



| Method(#Eval.) | Mean±Std.Err. |
|---|---|
| BOHB(200) | $7.158 \times 10^{-2} \pm 1.0303 \times 10^{-3}$ |
| BOHB(230) | $7.151 \times 10^{-2} \pm 9.8367 \times 10^{-4}$ |
| BOHB(350) | $7.061 \times 10^{-2} \pm 5.9322 \times 10^{-4}$ |
| RE(200) | $7.067 \times 10^{-2} \pm 1.1417 \times 10^{-3}$ |
| RE(230) | $7.061 \times 10^{-2} \pm 1.1329 \times 10^{-3}$ |
| RE(400) | $6.929 \times 10^{-2} \pm 6.4804 \times 10^{-4}$ |
| RE(600) | $6.879 \times 10^{-2} \pm 1.0039 \times 10^{-3}$ |
| ModLap(200) | $6.850 \times 10^{-2} \pm 3.7914 \times 10^{-4}$ |
| ModLap(230) | $6.826 \times 10^{-2} \pm 2.2317 \times 10^{-4}$ |
| ModLap(290) | $6.826 \times 10^{-2} \pm 2.2317 \times 10^{-4}$ |

# D BATCH BAYESIAN OPTIMIZATION ON PERMUTATIONS USING THE ACQUISITION WEIGHTED KERNELS

## D.1 LAW REGRET BOUND

### D.1.1 Outline

We show that LAW with GP-UCB or EST has the vanishing simple regret with high probability.

In Bayesian optimization (BO), the goal is to find a minimum for a given objective $f$

$$\mathbf{x}^\star = \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x})$$

First, we introduce different types of regret. Our analysis on the vanishing simple regret of LAW only requires batch version of all regrets below. Therefore, the proof for the vanishing simple regret can be read without referring to sequential version of regrets below. The sequential version definitions are used when we contrast our regret analysis with the regret analysis in existing works (Desautels et al., 2014; Kandasamy, Krishnamurthy, et al., 2018).

We begin with two equivalent round indexing in *the batch setting*, the sequential indexing, an 1-tuple and the batch indexing, an ordered 2-tuple which are related via following mappings.

$$\begin{aligned}
&\mathfrak{T}^{(B)}_{bat} : \mathbb{N} \to \mathbb{N} \times [B] && t \mapsto ([(t-1) \ \mathbf{mod}\ B] + 1, [(t-1)\ \mathbf{rem}\ B] + 1) \\
&\mathfrak{T}^{(B)}_{seq} : \mathbb{N} \times [B] \to \mathbb{N} && (t, b) \mapsto (t-1) \times B + b
\end{aligned}$$

The batch indexing is primarily used and the sequential indexing is expressed via $\mathfrak{T}^{(B)}_{seq}$.

With two indexing, we have batch and sequential versions of regret definitions with the instantaneous regret $r_{t,b} = f(\mathbf{x}^\star) - f(\mathbf{x}_{t,b})$ at a query point $\mathbf{x}_{t,b}$. In the case of a noisy objective, $y_{t,b} = f(\mathbf{x}_{t,b}) + \epsilon_{t,b}$ is a corresponding evaluation with a noise $\epsilon_{t,b}$.

Note that we use the definition of sequential simple/cumulative regret in the context of batch BO. Since sequential simple regret is equal to batch simple regret, we call both simple regret without prefixes. In (Contal et al., 2013), sequential cumulative regret is termed full cumulative regret to contrast with batch cumulative regret.

| Type | Batch | Sequential |
|---|---|---|
| Instantaneous | $r_t^{(B)} = \min_{b=1,\cdots,B} r_{t,b}$ | $r_{\mathfrak{T}_{seq}^{(B)}(t,b)} = r_{t,b}$ |
| Simple | $S_T^{(B)} = \min_{t=1,\cdots,T} r_t^{(B)}$ | $S_{\mathfrak{T}_{seq}^{(B)}(T,b)} = \min_{\mathfrak{T}_{seq}^{(B)}(t,b') \leqslant \mathfrak{T}_{seq}^{(B)}(T,b)} r_{T_{seq}^{(B)}(t,b')}$ |
| Cumulative | $R_T^{(B)} = \sum_{t=1}^{T} r_t^{(B)}$ | $R_{\mathfrak{T}_{seq}^{(B)}(T,b)} = \sum_{\mathfrak{T}_{seq}^{(B)}(t,b') \leqslant \mathfrak{T}_{seq}^{(B)}(T,b)} r_{T_{seq}^{(B)}(t,b')}$ |
| Simple & Cumulative | $S_T^{(B)} \leqslant \frac{1}{T} R_T^{(B)}$ | $S_{\mathfrak{T}_{seq}^{(B)}(T,b)} \leqslant \frac{1}{\mathfrak{T}_{seq}^{(B)}(T,b)} R_{\mathfrak{T}_{seq}^{(B)}(T,b)}$ |
| Between Simple | $S_T^{(B)} = S_{\mathfrak{T}_{seq}^{(B)}(T,B)}$ | |
| Between Cumulative | $R_T^{(B)} \leqslant \frac{1}{B} R_{\mathfrak{T}_{seq}^{(B)}(T,B)}$ | |

**Table D.1**: Types of regrets

The simple regret is in accord with the goal of BO (Kandasamy, Krishnamurthy, et al., 2018) while the cumulative regret is prevalent in bandit (Lattimore and Szepesvári, 2020).

When an algorithm exhibits that cumulative regret averaged over rounds converges to zero, then the algorithm is called no regret. As stated in Tab. D.1, simple regret is bounded above by cumulative regret averaged over rounds, therefore, vanishing simple regret is often proved by showing that the algorithm is no regret (Kandasamy, Krishnamurthy, et al., 2018).

In batch BO, there are two types of query depending on the accessible information. *Non-delayed* query point uses all previous query points with all corresponding evaluations, e.g. $\{\mathbf{x}_{t,1}\}_{t\in[T]}$ in round $T$ of LAW while *delayed* query point uses all previous query points but evaluations corresponding to the query points $\{\mathbf{x}_{T,b}\}_{b=2,\cdots,B}$ are not used in round $T$ of LAW. For evaluation, instantaneous regret and posterior variance, we can say *non-delayed* and *delayed* according to the query point with which it is defined.

The our analysis consists of steps below

1. Bound batch cumulative regret with the sum of non-delayed instantaneous regrets, i.e, the regrets from the first points in each batch

$$R_T^{(B)} = \sum_{t=1}^{T} r_t^{(B)} \leqslant \sum_{t=1}^{T} r_{t,1}$$

2. Bound non-delayed instantaneous regrets with non-delayed posterior variance, i.e, posterior variance conditioned on all previous query points with their evaluations.

$$\sum_{t=1}^{T} r_{t,1} \leqslant \sum_{t=1}^{T} \eta_t \sigma_{t-1,1}(\mathbf{x}_{t,1})$$

$\eta_t$ depends on the acquisition function and the details are given in Thm. D.8.

3. Bound non-delayed posterior variance with all posterior variance (Lem. D.2)

$$\sum_{t=1}^{T} \sigma_{t-1,1}(\mathbf{x}_{t,1}) \leqslant 1 + \frac{w_+}{w_-}\frac{1}{B}\sum_{t=1}^{T}\sum_{b=1}^{B}\sigma_{t-1,b}(\mathbf{x}_{t,b})$$

While the regret analysis on sequential cumulative regret (Desautels et al., 2014; Kandasamy, Krishnamurthy, et al., 2018)[39] requires high probability confidence interval for $r_{t,b}$ for all $t \in [T]$ and $b \in [B]$, our analysis on batch cumulative regret requires high probability confidence interval for $r_{t,1}$ for all $t \in [T]$. More detailed discussion on the differences between two approaches is given after the proof (see Appx. D.1.3).

### D.1.2 Regret Bound of LAW

In Bayesian optimization using LAW, the surrogate model is Gaussian processes with a kernel $K$. At t-th round of batch Bayesian optimization with the batch size of $B$, we have $L_t^{AW}$ which defines L-ensembles of k-DPP. $L_t^{AW}$ is the product of the predictive covariance function of $K$ conditioned on $\mathcal{D}_{t-1} = \{(\mathbf{x}_{s,b}, y_{s,b})\}_{s\in[t-1],b\in[B]}$ and the acquisition function $a_t$ using the evaluation data $\mathcal{D}_{t-1}$ as follows

$$L_t^{AW}(\mathbf{x},\mathbf{x}') = w(a_t(\mathbf{x})) \cdot L_t(\mathbf{x},\mathbf{x}') \cdot w(a_t(\mathbf{x}')). \tag{D.1}$$

where $L_t(\mathbf{x},\mathbf{x}') = K(\mathbf{x},\mathbf{x}' \,|\, \mathcal{D}_{t-1})$ is the *diversity gauge* and $w : \mathbb{R} \to \mathbb{R}$ is positive increasing, $w_- = \inf_{x\in\mathbb{R}} w(x) > 0$ and $w_+ = \sup_{x\in\mathbb{R}} w(x) < \infty$, which we call the *weight function*.

The batch with $B$ points $\mathbf{x}_{t,1}, \cdots, \mathbf{x}_{t,B}$ are acquired by

$$\begin{aligned}
\mathbf{x}_{t,1} &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} a_t(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} w(a_t(\mathbf{x})) = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log w(a_t(\mathbf{x}))^2 \\
\mathbf{x}_{t,b} &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log\det([L_t^{AW}(\mathbf{x},\mathbf{x})]_{\{\mathbf{x}_i\}_{i\in[b-1]}\cup\{\mathbf{x}\}}) \\
&= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log(L_t(\mathbf{x},\mathbf{x}\,|\{\mathbf{x}_{t,i}\}_{i\in[b-1]}) \cdot w(a_t(\mathbf{x}))^2)
\end{aligned} \tag{D.2}$$

where $L_t(\mathbf{x},\mathbf{x}\,|\{\mathbf{x}_{t,i}\}_{i\in[b-1]})$ is the posterior variance of the kernel $L_t$ conditioned on $\{\mathbf{x}_{t,i}\}_{i\in[b-1]}$.

Note that the posterior variance of the *posterior covariance function* $K_t$ conditioned on $\{\mathbf{x}_{t,i}\}_{i\in[b-1]}$ is equal to the posterior variance of the *prior covariance function* $K$ conditioned on $\mathcal{D}_{t-1} \cup \{\mathbf{x}_{t,i}\}_{i\in[b-1]}$.

In the rest of the section, we use below simpler notation

$$\sigma^2_{t-1,b}(\mathbf{x}) = L_t(\mathbf{x},\mathbf{x}\,|\{\mathbf{x}_{t,i}\}_{i\in[b-1]}) = K(\mathbf{x},\mathbf{x}\,|\,\mathcal{D}_{t-1}\cup\{\mathbf{x}_{t,i}\}_{i\in[b-1]}) \tag{D.3}$$

39 In (Kandasamy, Krishnamurthy, et al., 2018), vanishing simple regret is proved by showing that a bound with sequential cumulative regret averaged over rounds converges to zero.

with $[0] = \emptyset$, and $\mu_t(\mathbf{x})$ for the predictive mean conditioned on $\mathcal{D}_{t-1}$.

Note that $\sigma^2_{t-1,b}$ is well defined for $b = 2, \cdots, B$ since the posterior variance does not depend on output values while the predictive mean is defined only when $b = 1$ where evaluated output $y_{s,b}$ exists for each $\mathbf{x}_{s,b}$ in conditioning data.

We start with lemmas used in the regret bound analysis.

**Lemma D.1.** *Assume a kernel such that* $K(\cdot,\cdot) \leqslant 1$. *For each* $t \in [T]$, *LAW acquires a batch using the evaluation data* $\mathcal{D}_{t-1}$, *the diversity measure* $L_t(\cdot,\cdot) = K(\cdot,\cdot|\mathcal{D}_{t-1})$, *an acquisition function* $a_t(\cdot)$ *and a weight function* $w(\cdot)$ *(as defined below Eq. D.1). The posterior variance defined as Eq. D.3. has the following relation*

$$\sigma_{t,1}(\mathbf{x}_{t+1,1}) \leqslant \frac{w_+}{w_-}\sigma_{t-1,b}(\mathbf{x}_{t,b}) \quad 1 \leqslant t \leqslant T, 2 \leqslant b \leqslant B$$

*Proof.* By the definition of $\mathbf{x}_{t,b}$

$$\begin{aligned}\mathbf{x}_{t,b} &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log(L_t(\mathbf{x}|\{\mathbf{x}_{t,i}\}_{i\in[b-1]}) \cdot w(a_t(\mathbf{x}))^2) \\ &= \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \log(\sigma^2_{t-1,b}(\mathbf{x}) \cdot w(a_t(\mathbf{x})))\end{aligned}$$

we have

$$w(a_t(\mathbf{x}))\sigma_{t-1,b}(\mathbf{x}) \leqslant w(a_t(\mathbf{x}_{t,b}))\sigma_{t-1,b}(\mathbf{x}_{t,b}) \quad \forall \mathbf{x} \in \mathcal{X} \tag{D.4}$$

thus $\forall \mathbf{x} \in \mathcal{X}$

$$\sigma_{t-1,b}(\mathbf{x}) \leqslant \frac{w(a_t(\mathbf{x}))}{w(a_t(\mathbf{x}_{t,b}))}\sigma_{t-1,b}(\mathbf{x}_{t,b}) \leqslant \frac{w_+}{w_-}\sigma_{t-1,b}(\mathbf{x}_{t,b}). \tag{D.5}$$

By the "Information never hurts" principle (Krause, Singh, et al., 2008), i.e. the posterior variance decreases as the conditioning set increases, we have

$$\sigma_{t,1}(\mathbf{x}) \leqslant \sigma_{t-1,b}(\mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{X}$$

since $\sigma_t$ is conditioned by $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{\mathbf{x}_{t,i}\}_{i\in[B]}$ while $\sigma_{t,b}$ is conditioned by $\mathcal{D}_{t-1} \cup \{\mathbf{x}_{t,i}\}_{i\in[b-1]}$. Combining these two inequalities, we have

$$\sigma_{t,1}(\mathbf{x}) \leqslant \sigma_{t-1,b}(\mathbf{x}) \leqslant \frac{w_+}{w_-}\sigma_{t-1,b}(\mathbf{x}_{t,b}) \quad \forall \mathbf{x} \in \mathcal{X}$$

which also applies when $\mathbf{x} = \mathbf{x}_{t+1,1}$.

Q.E.D. □

*Remark* D.1. LAW does not use the heuristic called the relevant region (Contal et al., 2013; Kathuria et al., 2016), which makes the proof simpler compared with the Lemma 6.5 in (Kathuria et al., 2016).

*Remark* D.2. The Lemma 6.5 in (Kathuria et al., 2016) claims that the inequality similar to Eq. D.4 and Eq. D.5 holds for sampling (DPP-SAMPLE). However, such inequality relies on fact that $\mathbf{x}_{t,b}$ is the maximum of an objective which is not guaranteed to hold for sampling (DPP-SAMPLE). The regret analysis of DPP-SAMPLE in (Kathuria et al., 2016) appears to need a revision. In our version, we do not make any claim for the case of sampling.

**Lemma D.2.** *Assume a kernel such that* $K(\cdot,\cdot) \leqslant 1$. *For each* $t \in [T]$, *LAW acquires a batch using the evaluation data* $\mathcal{D}_{t-1}$, *the diversity measure* $L_t(\cdot,\cdot) = K(\cdot,\cdot|\mathcal{D}_{t-1})$, *an acquisition function* $a_t(\cdot)$ *and a weight function* $w(\cdot)$ *(as defined below Eq. D.1). The posterior variance defined as Eq. D.3. has the following relation*

$$\sum_{t=1}^{T} \sigma_{t-1,1}(\mathbf{x}_{t,1}) \leqslant 1 + \frac{w_+}{w_-}\frac{1}{B}\sum_{t=1}^{T}\sum_{b=1}^{B} \sigma_{t-1,b}(\mathbf{x}_{t,b}). \tag{D.6}$$

*Proof.* From Lem. D.1, for $b = 2, \cdots, B$, we have

$$\sigma_{t,1}(\mathbf{x}_{t+1,1}) = \sigma_t(\mathbf{x}_{t+1,1}) \leqslant \frac{w_+}{w_-}\sigma_{t-1,b}(\mathbf{x}_{t,b})$$

Summing these for $b = 2, \cdots, B$ and $\sigma_{t-1,1}(\mathbf{x}_{t,b})$

$$\sigma_{t-1,1}(\mathbf{x}_{t,b}) + (B-1)\sigma_{t,1}(\mathbf{x}_{t+1,1}) \leqslant \frac{w_+}{w_-}\sum_{b=1}^{B} \sigma_{t-1,b}(\mathbf{x}_{t,b})$$

since $w_- \leqslant w_+$. Summing this with respect to t, we have

$$\sum_{t=1}^{T} \sigma_{t-1,1}(\mathbf{x}_{t,b}) + (B-1)\sum_{t=1}^{T} \sigma_{t,1}(\mathbf{x}_{t+1,1}) \leqslant \frac{w_+}{w_-}\sum_{t=1}^{T}\sum_{b=1}^{B} \sigma_{t-1,b}(\mathbf{x}_{t,b})$$

The term on the left hand side can be rewritten

$$B\sum_{t=1}^{T} \sigma_{t-1,1}(\mathbf{x}_{t,b}) + (B-1)(\sigma_{T,1}(\mathbf{x}_{T+1,1}) - \sigma_{0,1}(\mathbf{x}_{1,1}))$$

Since $(B-1)(\sigma_{0,1}(\mathbf{x}_{1,1}) - \sigma_{T,1}(\mathbf{x}_{T+1,1})) \leqslant B\sigma_{0,1}(\mathbf{x}_{1,1})$

$$\sum_{t=1}^{T} \sigma_{t-1,1}(\mathbf{x}_{t,b}) \leqslant \sigma_{0,1}(\mathbf{x}_{1,1}) + \frac{w_+}{w_-}\frac{1}{B}\sum_{t=1}^{T}\sum_{b=1}^{B} \sigma_{t-1,b}(\mathbf{x}_{t,b})$$

Q.E.D. □

*Remark* D.3. In Lemma 3 in (Contal et al., 2013) and Lemma 6.5 in (Kathuria et al., 2016), the second term in Eq. D.7 is ignored. However, $\sigma_{T,1}(\mathbf{x}_{T+1,1}) - \sigma_{0,1}(\mathbf{x}_{1,1})$ can be negative, which should not be ignored. Nevertheless, this error does not change the regret analysis in (Contal et al., 2013) because constant terms divided by T vanishes. Our version has the additional constant 1 on the right hand side of Eq. D.6.

**Definition D.1.** The maximum information gain $\gamma_T$ is defined as below

$$\gamma_T = \gamma(T;\mathcal{X}) = \max_{X\subset\mathcal{X},|X|=T} \mathbf{I}(Y_X;\mathbf{f}_X) = \max_{X\subset\mathcal{X},|X|=T} H(Y_X) - H(Y_X|\mathbf{f}_X)$$

where Y is the observation at X and H is the differential entropy.

For Gaussian processes with the kernel K and the variance of observation noise $\sigma^2$

$$\gamma_T = \gamma(T;\mathcal{X},K,\sigma^2) = \max_{X\subset\mathcal{X},|X|=T} \frac{1}{2}\log\det(I + \sigma^{-2}K(X,X))$$

We rephrase lemmas from previous works with the batch indexing for notational ease and discuss the noteworthy points in the rephrased version compared with the original ones.

**Lemma D.3** (Lemma 3 (Srinivas et al., 2010), Lemma 4 (Contal et al., 2013), Theorem 3.1 (Zi Wang, B. Zhou, et al., 2016))**.** *Assume a kernel such that* $K(\cdot,\cdot) \leqslant 1$*. For each* $t \in [T]$*, LAW acquires a batch using the evaluation data* $\mathcal{D}_{t-1}$*, the diversity measure* $L_t(\cdot,\cdot) = K(\cdot,\cdot|\mathcal{D}_{t-1})$*, an acquisition function* $a_t(\cdot)$ *and a weight function* $w(\cdot)$ *(as defined below Eq. D.1). The posterior variance defined as Eq. D.3. has the following relation*

$$\sum_{t=1}^{T}\sum_{b=1}^{B} \sigma^2_{t-1,b}(\mathbf{x}_{t,b}) \leqslant C_1 \gamma_{TB}$$

*where* $C_1 = \frac{2}{\log(1+\sigma^{-2})}$ *and* $\gamma_{TB}$ *is the maximum information gain at* $TB$

*Proof.* Following the trick used in the proof of Lemma 5.4 in (Srinivas et al., 2010),

$$\sigma^2_{t-1,b}(\mathbf{x}) = \sigma^2\sigma^{-2}\sigma^2_{t-1,b}(\mathbf{x}) \leqslant \frac{1}{\log(1+\sigma^{-2})}\log(1+\sigma^{-2}\sigma^2_{t-1,b}(\mathbf{x})).$$

In LAW, $\mathbf{x}_{t,1}$ and $\mathbf{x}_{t,b}$ deterministic conditioned respectively on $\mathcal{D}_{t-1} = \{(\mathbf{x}_{s,b}, y_{s,b})\}_{s\in[t-1],b\in[B]}$ and $\mathcal{D}_{t-1} \cup \{\mathbf{x}_{t,c}\}_{c=2,\cdots,b-1}$ for $b = 2,\cdots,B$. Also, $\mathbf{x}_{t,b}$ does not depend on $\{y_{s,c}\}_{\mathfrak{T}^{(B)}_{seq}(s,c)<\mathfrak{T}^{(B)}_{seq}(t,b)}$ as long as $\{\mathbf{x}_{s,c}\}_{\mathfrak{T}^{(B)}_{seq}(s,c)\leqslant\mathfrak{T}^{(B)}_{seq}(t,b)}$. Therefore, the proof of Lemma 5.3 in (Srinivas et al., 2010) can be applied

$$\sum_{t=1}^{T}\sum_{b=1}^{B} \sigma^2_{t-1,b}(\mathbf{x}_{t,b}) \leqslant \frac{1}{\log(1+\sigma^{-2})}\sum_{t=1}^{T}\sum_{b=1}^{B}\log(1+\sigma^{-2}\sigma^2_{t-1,b}(\mathbf{x}_{t,b}))$$
$$= \frac{2}{\log(1+\sigma^{-2})}\mathbf{I}(Y_{\{\mathbf{x}_{t,b}\}_{t\in[T],b\in[B]}}; \mathbf{f}_{\{\mathbf{x}_{t,b}\}_{t\in[T],b\in[B]}}) \leqslant \frac{2}{\log(1+\sigma^{-2})}\gamma_{TB}$$

□

*Remark* D.4. In contrast to Lemma 5.4 in (Srinivas et al., 2010) which bounds the sum of square of regrets, Lem. D.3 bounds the sum of the posterior variances. The delayed evaluation $\{y_{t,b}\}_{t\in[T],b\in[B]}$ does not cause any impediment in the proof.

**Lemma D.4** (Lemma 6.1 (Kathuria et al., 2016), Lemma 3.2 (Zi Wang, B. Zhou, et al., 2016))**.** *For* $\zeta_t = \left(2\log\left(\frac{\pi_t^2}{2\delta}\right)\right)^{1/2}$ *with* $\delta \in (0,1)$ *and* $\pi_t > 0$ *such that* $\sum_{t=1}^{\infty}\pi_t \leqslant 1$*, an arbitrary sequence of actions* $\mathbf{x}_{1,1},\cdots,\mathbf{x}_{T,1} \in \mathcal{X}$

$$P\left(\bigcap_{t\in[T]}\left\{f \,\middle|\, |f(\mathbf{x}_{t,1}) - \mu_{t-1}(\mathbf{x}_{t,1})| \leqslant \zeta_t \cdot \sigma_{t-1,1}(\mathbf{x}_{t,1})\right\}\right) \geqslant 1-\delta.$$

*Remark* D.5. $\zeta_t$ only depends on the number of batch round $t$ and is independent with the batch size $B$. Therefore, $\zeta_t$ in batch BO is the same as one in the sequential BO.

**Lemma D.5** (Lemma 3.3 (Zi Wang, B. Zhou, et al., 2016))**.** *If* $|f(\mathbf{x}_{t,1}) - \mu_{t-1}(\mathbf{x}_{t,1})| \leqslant \zeta_t \sigma_{t-1,1}(\mathbf{x}_{t,1})$

$$r_{t,1} = f(\mathbf{x}_{t,1}) - f(\mathbf{x}^\star) \leqslant (\nu_t + \zeta_t)\sigma_{t-1,1}(\mathbf{x}_{t,1})$$

*where* $\nu_t = \left(\min_{\mathbf{x}\in\mathcal{X}} \frac{\mu_{t-1}(\mathbf{x}) - \hat{m}}{\sigma_{t-1,1}(\mathbf{x})}\right)$*,* $\hat{m}$ *is the estimate of the optimum(Zi Wang, B. Zhou, et al., 2016) and* $\zeta_t = \left(2\log\left(\frac{\pi_t^2}{2\delta}\right)\right)^{1/2}$ *with* $\delta \in (0,1)$ *and* $\pi_t > 0$ *such that* $\sum_{t=1}^{\infty} \pi_t^{-1} \leqslant 1$.

*Remark* D.6. In Lem. D.5, we only bound regrets in $(t,1)$-th round where there is no delayed evaluation.

*Remark* D.7. In contrast to the original condition $\sum_{t=1}^{T} \pi_t \leqslant 1$, we use $\sum_{t=1}^{\infty} \pi_t \leqslant 1$ so that $\pi_t$s are $T$ independent as the recommendation of the choice $\pi_t = \frac{1}{6}\pi^2 t^2$ in (Zi Wang, B. Zhou, et al., 2016). When the number of rounds $T$ is known in advance, $T$ dependent $\pi_t$ is possible, e.g, $\pi_t = T$ (Zi Wang, B. Zhou, et al., 2016). By making $\pi_t$ independent with $T$, EST becomes anytime, i.e. not requiring that the number of rounds is known in advance.

**Lemma D.6** (Lemma 5.1 (Srinivas et al., 2010))**.** *For* $\delta \in (0,1)$ *and* $\beta_{t,1}^{(B)UCB} = 2\log\left(\frac{|\mathcal{X}|\pi^2(\mathfrak{T}_{seq}^{(B)}(t,1))^2}{6\delta}\right)$*,*

$$P\left(\bigcap_{\mathbf{x}\in\mathcal{X}} \left\{ f \;\middle|\; |f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leqslant (\beta_{t,1}^{(B)UCB})^{1/2} \cdot \sigma_{t-1,1}(\mathbf{x}) \right\}\right) \geqslant 1 - \delta.$$

*Remark* D.8. Note that $\beta_{t,1}^{(B)UCB} = \beta_{\mathfrak{T}_{seq}^{(B)}(t,1)}^{UCB}$. in batch BO with the batch size of $B$, $\beta$ is set as if there is $B$ times more rounds.

**Lemma D.7** (Lemma 5.2 (Srinivas et al., 2010), Lemma 1 (Contal et al., 2013))**.** *If* $|f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leqslant (\beta_{t,1}^{(B)UCB})^{1/2}\sigma_{t-1,1}(\mathbf{x})$ *for all* $\mathbf{x} \in \mathcal{X}$*, then*

$$r_{t,1} = f(\mathbf{x}_{t,1}) - f(\mathbf{x}^\star) \leqslant 2(\beta_{t,1}^{(B)UCB})^{1/2}\sigma_{t-1,1}(\mathbf{x}_{t,1}).$$

*Remark* D.9. In Lem. D.7, we only bound regrets in $(t,1)$-th round where there is no delayed evaluation.

**Theorem D.8.** *Assume a kernel such that* $K(\cdot,\cdot) \leqslant 1$*,* $|\mathcal{X}| < \infty$ *and* $f : \mathcal{X} \to \mathbb{R}$ *is sampled from* $\mathcal{GP}(\mathbf{0}, K)$*. In each round* $t \in [T]$ *of batch Bayesian optimization, LAW acquires a batch using the evaluation data* $\mathcal{D}_{t-1}$*, the diversity measure* $L_t(\cdot,\cdot) = K(\cdot,\cdot|\mathcal{D}_{t-1})$*, an acquisition function* $a_t(\cdot)$ *and a weight function* $w(\cdot)$ *(as defined below Eq. D.1).*

*Let* $C_1 = \frac{36}{\log(1+\sigma^{-2})}$ *where* $\sigma^2$ *is the variance of the observation noise and* $\delta \in (0,1)$*.*

*For GP-UCB, define* $\beta_{t,1}^{(B)UCB} = 2\log\left(\frac{|\mathcal{X}|\pi^2(\mathfrak{T}_{seq}^{(B)}(t,1))^2}{6\delta}\right)$ *and let*

$$\eta_t^{(B)} = 2(\beta_{t,1}^{(B)UCB})^{1/2}$$

*For EST, define* $\nu_t = \min_{\mathbf{x}}\left(\frac{\mu_{t-1}(\mathbf{x})-\hat{m}_t}{\sigma_{t-1,1}(\mathbf{x})}\right)$ *where* $\hat{m}_t$ *is the estimate of the optimum (Zi Wang, B. Zhou, et al., 2016),* $\zeta_t = \left(2\log\left(\frac{\pi_t^2}{2\delta}\right)\right)^{1/2}$, $\pi_t > 0$ *such that* $\sum_{t=1}^{\infty}\pi_t^{-1} \leqslant 1$ *and let*

$$\eta_t^{(B)} = \nu_{t^*} + \zeta_t$$

*where* $t^* = \underset{s\in[t]}{\operatorname{argmax}}\,\nu_s$.

*Then batch cumulative regret satisfies the following bound*

$$P\left(\left\{\frac{R_T^{(B)}}{T} \leqslant \frac{\eta_T^{(B)}}{T} + \eta_T\frac{w_+}{w_-}\sqrt{C_1\frac{\gamma_{TB}}{TB}}\right\}\right) \geqslant 1-\delta.$$

*Proof.* Let $\eta_t^{(B)} = \begin{cases}\nu_{t^*}+\zeta_t & \text{EST}\\ 2(\beta_{t,1}^{(B)UCB})^{1/2} & \text{UCB}\end{cases}$.

For the batch cumulative regret case, we use Lem. D.1.

$$\begin{aligned}
R_T^{(B)} &= \sum_{t=1}^{T} r_t^{(B)} = \sum_{t=1}^{T}\min_{b=1,\cdots,B} r_{t,b}\\
&\leqslant \sum_{t=1}^{T} r_{t,1} \qquad\qquad (D.7)\\
&\leqslant \eta_T^{(B)}\cdot\sum_{t=1}^{T}\sigma_{t-1,1}(\mathbf{x}_{t,1}) \quad\text{by}\quad \begin{cases}\text{Lem. D.5} & \text{EST}\\ \text{Lem. D.7} & \text{UCB}\end{cases}\\
&\leqslant \eta_T^{(B)}\cdot\left(1+\frac{w_+}{w_-}\frac{1}{B}\sum_{t=1}^{T}\sum_{b=1}^{B}\sigma_{t-1,b}(\mathbf{x}_{t,b})\right) \quad\text{by Lem. D.1}\\
&\leqslant \eta_T^{(B)}\cdot\left(1+\frac{w_+}{w_-}\sqrt{\frac{T}{B}\sum_{t=1}^{T}\sum_{b=1}^{B}\sigma_{t-1,b}^2(\mathbf{x}_{t,b})}\right) \quad\text{by Cauchy-Schwarz}\\
&\leqslant \eta_T^{(B)}\cdot\left(1+\frac{w_+}{w_-}\sqrt{\frac{T}{B}C_1\gamma_{TB}}\right) \quad\text{by Lem. D.3}
\end{aligned}$$

By Lem. D.4 for EST and Lem. D.6 for UCB, above two inequalities hold with the probability at least $1-\delta$. □

*Remark* D.10. Due to the difference of the statements in Lem. D.1, the batch cumulative regret bound additionally has the term $\frac{w_+}{w_-}$. Even with this additional term, it shows that the bound of the batch cumulative regret of LAW enjoys the same asymptotic behavior as existing methods (Contal et al., 2013; Desautels et al., 2014; Kathuria et al., 2016).

*Remark* D.11. This theorem provides a rough guideline how to choose a weight function, that is, bounded below by a positive value and bounded above, which is the condition we specify for the weight function. Even though this shows that simple regret vanishes, this regret bound for LAW is loose because not much specific structure of the weight function other than the bound is used. We expect that, using other properties of the weight function along with the boundedness, the bound can be improved.

### D.1.3 Difference to the analysis of sequential cumulative regret

In our regret analysis, we analyze batch cumulative regret. In existing works, sequential cumulative regret is analyzed as an end goal (Desautels et al., 2014) and as a medium to show vanishing simple regret in (Kandasamy, Krishnamurthy, et al., 2018). In (Contal et al., 2013),[40] both batch cumulative regret and sequential cumulative regret are analyzed.[41] We discuss the differences between these two approaches and the technical details in their proofs.

By definition, the analysis of sequential cumulative regret takes into account all instantaneous regrets incurred while batch cumulative regret considers minimum instantaneous regrets in each batch. Therefore, bounds on sequential cumulative regret are stronger than ones on batch cumulative regret in this sense (as shown in Tab. D.1 Relation between two Cumulative). However, each has its own more appropriate scenario to use. The sequential cumulative regret is often appropriate in the situation where the optimization objective represents the cost of evaluations. For example, in multi-armed bandit, each instantaneous regret represents the cost of evaluation (playing arm-pulling) and the goal is to minimize the incurred cost in finding the best bandit machine. On the other hand, batch cumulative regret is often reasonable when the optimization objective is different to the cost of evaluations. For example, in hyperparameter optimization, the cost of evaluations can be wall-clock time and the objective is the cross-validation error. In this case, we want to find a good hyperparameter no matter how bad hyperparameters are evaluated, which possibly acts as exploratory query points.

In proofs, each analysis takes a slightly different route. As argued in (Desautels et al., 2014), to bound all instantaneous regret, a wider confidence bound is needed to bound instantaneous regret with the corresponding posterior variance. While the posterior mean is not update in the batch acquisition until all query points are evaluated, the posterior variance is updated whenever a new query point is

40 Sequential cumulative regret is termed full cumulative regret in (Contal et al., 2013).

41 The analysis of sequential cumulative regret in (Contal et al., 2013) may need modification and not be correct, see following paragraph for a brief explanation and for more elaborated explanation, refer to (Desautels et al., 2014).

given no matter whether it is evaluated or not. To guarantee high probability bound for all instantaneous regrets, an additional kernel-dependent constant is introduced and the constant is controlled with an initialization scheme(Desautels et al., 2014). In (Kandasamy, Krishnamurthy, et al., 2018), the analysis relies on such kernel-dependent constant and the initialization scheme but it is empirically shown that the algorithm performs well without the initialization scheme. The necessity of the kernel-dependent constant suggests that the analysis of sequential cumulative regret in (Contal et al., 2013) requires a revision.

For the purpose to show vanishing simple regret, batch cumulative regret can be used circumventing the additional constant and the initialization scheme proposed in (Desautels et al., 2014). In the analysis using batch cumulative regret, only non-delayed regret is considered and bounded by non-delayed posterior variance (Eq. D.7). Then non-delayed posterior variance is bounded by the average of non-delayed posterior variance and delayed posterior variances in the same batch (Lem. D.2). Therefore, the effect of the batch size influences the bound in this posterior variance bounding step. However, in the analysis using sequential cumulative regret(Desautels et al., 2014; Kandasamy, Krishnamurthy, et al., 2018), both non-delayed and delayed instantaneous regrets need to be bounded. The bound is the corresponding posterior variance multiplied by a specially design number to handle delayed cases. In response to this, the additional kernel-dependent constant and the initialization scheme are introduced in (Desautels et al., 2014).

Batch cumulative regret is enough in showing vanishing simple regret. The proof only considers non-delayed instantaneous regrets in batches. Therefore, the analysis of batch cumulative regret reveals how delayed query points in a batch explore effectively and help to reduce future non-delayed instantaneous regrets. We admit that some may argue that a tighter bound is possible by taking into account delayed evaluations with smaller instantaneous regrets. Still, this is aligned with the intuition of many batch acquisition methods promoting diversity in batches. In practice, it is not unlikely to observe a delayed evaluation is better than the non-delayed evaluation in the same batch.

### D.1.4 Growth Rate of UCB/EST hyperparameter

$$\eta_t^{(B)} = \begin{cases} \nu_{t^*} + \zeta_t & \text{EST} \\ 2(\beta_{t,1}^{(B)UCB})^{1/2} & \text{UCB} \end{cases}$$

where $\beta_{t,1}^{(B)UCB} = 2\log\left(\frac{|\mathcal{X}|\pi^2((t-1)B+1)^2}{6\delta}\right)$, $t^* = \operatorname{argmax}_{s\in[t]} \nu_s$, $\nu_t = \min_{\mathbf{x}}\left(\frac{\mu_{t-1}(\mathbf{x})-\hat{m}_t}{\sigma_{t-1,1}(\mathbf{x})}\right)$ where $\hat{m}_t$ is the estimate of the optimum (Zi Wang, B. Zhou, et al., 2016), $\zeta_t = \left(2\log\left(\frac{\pi_t^2}{2\delta}\right)\right)^{1/2}$, $\pi_t > 0$ such that $\sum_{t=1}^{\infty}\pi_t^{-1} \leqslant 1$.

For UCB, it is clear that $2(\beta_{t,1}^{(B)UCB})^{1/2} = \mathcal{O}((\log(tB))^{1/2})$.

For EST, we first look into $\zeta_t$. If we choose $\pi_t = \frac{\pi^2 t^2}{6}$ as suggested in (Zi Wang, B. Zhou, et al., 2016), then $\zeta_t = \mathcal{O}((\log(tB))^{1/2})$. Since $\hat{m}_t = \mathbb{E}_{f\sim GP(\mu_{t-1}(\cdot),\sigma_{t-1}^2(\cdot))}[\inf_{\mathbf{x}} f(\mathbf{x})]$ (Zi Wang, B. Zhou, et al., 2016), from Lemma 5.1 in (Srinivas et al., 2010), we have

$$|f(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leqslant \tau_t\sigma_{t-1}(\cdot) \quad \forall \mathbf{x} \in \mathcal{X}$$

where $\tau_t^{1/2} = 2\log\left(\frac{|\mathcal{X}|\pi^2 t^2}{6\delta}\right)$.

Then

$$\hat{m}_t \geqslant \mu_{t-1}(\mathbf{x}_{lb}) - \tau_t\sigma_{t-1}(\mathbf{x}_{lb})$$

with $\mathbf{x}_{lb} = \operatorname{argmin}_{\mathbf{x}} \mu_{t-1}(\mathbf{x}) + \tau_t\sigma_{t-1}(\mathbf{x})$,

$$\min_{\mathbf{x}}\left(\frac{\mu_{t-1}(\mathbf{x}) - \hat{m}_t}{\sigma_{t-1,1}(\mathbf{x})}\right) \leqslant \frac{\mu_{t-1}(\mathbf{x}_{lb}) - \mu_{t-1}(\mathbf{x}_{lb}) + \tau_t\sigma_{t-1}(\mathbf{x}_{lb})}{\sigma_{t-1,1}(\mathbf{x}_{lb})} = \tau_t$$

Since, $\tau_t$ is increasing, we have $\nu_{t^*} \leqslant \tau_t$ and $\nu_{t^*} = \mathcal{O}((\log(tB))^{1/2})$.

Therefore, for EST, $\eta_t^{(B)} = \nu_{t^*} + \zeta_t = \mathcal{O}((\log(tB))^{1/2})$.

## D.2 POSITION KERNEL

### D.2.1 Information gain of kernels on a finite space

In this subsection, we show a bound of the information gain of kernels defined on a finite set.

**Theorem D.9.** *$K$ is a kernel on a finite set $\mathcal{X}$ ($|\mathcal{X}| < \infty$), $\sigma^2$ is the variance of the observation noise and $\Lambda = \{\lambda_n\}_{1,\cdots,|\mathcal{X}|}$ ($\lambda_n \geqslant \lambda_{n+1} \geqslant 0$) is the set of eigenvalues of the gram matrix $K(\mathcal{X},\mathcal{X})$.*

*The number of elements in a set $A$ is denoted by $N_A$, so $N_{\mathcal{X}}$ is the number of elements of $\mathcal{X}$ and is equal to the number of eigenvalues of $K(\mathcal{X},\mathcal{X})$.*

*Then*

$$\gamma(T; K, \mathcal{X}, \sigma^2) \leqslant \frac{1}{2}\min\{T \cdot \log\det(1 + \sigma^{-2}\max_{x\in\mathcal{X}} K(x,x)), \\ N_{\mathcal{X}} \cdot \log(1 + \sigma^{-2}\lambda_{max} \cdot T)\}$$

*Proof.* Let us consider the eigenvalues and the eigenvectors of the gram matrix $K(\mathcal{X},\mathcal{X})$.

$$K(\mathcal{X},\mathcal{X}) = U\Lambda U^T$$

with $\Lambda = diag(\lambda_1, \cdots, \lambda_{N_\mathcal{X}})$, $U = [u_1, \cdots, u_{N_\mathcal{X}}] \in \mathbb{R}^{N_\mathcal{X} \times N_\mathcal{X}}$ where $\lambda_i$ is an eigenvalue and $u_i$ is the corresponding eigenvector.

Since

$$K(x, x') = \sum_{i=1}^{N_\mathcal{X}} \lambda_i [u_i]_x [u_i]_{x'},$$

the map

$$\phi(x) = [\sqrt{\lambda_1}[u_1]_x, \cdots, \sqrt{\lambda_{N_\mathcal{X}}}[u_{N_\mathcal{X}}]_x]^T$$

is a $N_\mathcal{X}$ dimensional feature map

$$K(x, x') = \phi(x)^T \cdot \phi(x').$$

For a sequence $A = \{a_1, \cdots, a_{N_A}\}$ of $a_i \in \mathcal{X} = x_i{}_{i=1}^{N_\mathcal{X}}$, the gram matrix $K(A, A)$ can be expressed with the projection matrix $P_A^\mathcal{X} \in \{0,1\}^{N_A \times N_\mathcal{X}}$ from $X$ to $A$ such that $[P_A^\mathcal{X}]_{ij} = 1$ if $a_i = x_j$

$$K(A, A) = P_A^\mathcal{X} U \Lambda U^T (P_A^\mathcal{X})^T.$$

*Remark* D.12. Note that $(P_A^\mathcal{X})^T \cdot P_A^\mathcal{X}$ is $N_\mathcal{X} \times N_\mathcal{X}$ diagonal matrix and $[(P_A^\mathcal{X})^T \cdot P_A^\mathcal{X}]_{ii}$ is how many times $x_i$ appears in the sequence $A$.

We obtain two bounds. The first one is

$$\log\det(I + \sigma^{-2} K(A, A)) \leqslant \sum_{a \in A} \log\det(1 + \sigma^{-2} K(a, a))$$

using Hadamard's inequality.[42]

Adopting the proof for the information gain of the linear kernel from (Srinivas et al., 2010), the second one is

$$\begin{aligned}
&\log\det(I + \sigma^{-2} K(A, A)) \\
&= \log\det(I + \sigma^{-2} P_A^\mathcal{X} K(X, X) (P_A^\mathcal{X})^T) \\
&= \log\det(I + \sigma^{-2} P_A^\mathcal{X} U \Lambda U^T (P_A^\mathcal{X})^T) && \text{by Eq. D.8} \\
&= \log\det(I + \sigma^{-2} \Lambda^{\frac{1}{2}} U^T (P_A^\mathcal{X})^T P_A^\mathcal{X} U \Lambda^{\frac{1}{2}}) && \text{by Weinstein-Aronszajn identity} \\
&\leqslant \sum_{i=1}^{N_\mathcal{X}} \log\det(1 + \sigma^{-2} \lambda_i [U^T (P_A^\mathcal{X})^T P_A^\mathcal{X} U]_{ii}) && \text{by Hadamard's inequality} \\
&\leqslant \sum_{i=1}^{N_\mathcal{X}} \log\det(1 + \sigma^{-2} \lambda_i T) && \text{by Eq. D.9} \qquad \text{(D.8)}
\end{aligned}$$

42 https://en.wikipedia.org/wiki/Hadamard%27s_inequality

using Weinstein-Aronszajn identity[43], Hadamard's inequality[44] and below

$$
\begin{aligned}
&[U^T (P^{\mathcal{X}}_A)^T P^{\mathcal{X}}_A U]_{ii} \\
&= \sum_{k=1}^{N_{\mathcal{X}}} \sum_{l=1}^{N_{\mathcal{X}}} ([U]_{ki}) [(P^{\mathcal{X}}_A)^T P^{\mathcal{X}}_A]_{kl} ([U]_{li}) \\
&= \sum_{k=1}^{N_{\mathcal{X}}} [(P^{\mathcal{X}}_A)^T P^{\mathcal{X}}_A]_{kk} ([U]_{ki})^2 \qquad \text{by Rmk. D.12} \\
&\leqslant \underbrace{\sum_{k=1}^{N_{\mathcal{X}}} [(P^{\mathcal{X}}_A)^T P^{\mathcal{X}}_A]_{kk}}_{=N_A=T} \cdot \underbrace{\sum_{k=1}^{N_{\mathcal{X}}} ([U]_{ki})^2}_{=1} \qquad \sum_i a_i b_i \leqslant (\sum_i a_i)(\sum_i b_i) \quad \text{if} \quad a_i, b_i \geqslant 0
\end{aligned} \tag{D.9}
$$

where the second equality comes from the fact that $(P^{\mathcal{X}}_A)^T \cdot P^{\mathcal{X}}_A$ is $N_{\mathcal{X}} \times N_{\mathcal{X}}$ diagonal matrix and the last inequality is possible because every numbers are non-negative since $[(P^{\mathcal{X}}_A)^T \cdot P^{\mathcal{X}}_A]_{ii}$ is how many times $x_i$ appears in the sequence $A$.

Putting Eq. D.8 and Eq. D.8 together,

$$
\begin{aligned}
\log\det(I + \sigma^{-2} K(A, A)) \leqslant \min \Big\{ & T \cdot \log\det(1 + \sigma^{-2} \max_{x \in \mathcal{X}} K(x, x)), \\
& N_{\mathcal{X}} \cdot \log\det(1 + \sigma^{-2} \lambda_{max} T) \Big\}
\end{aligned}
$$

Q.E.D. □

## D.2.2 Positive definiteness of the Position kernel

In this subsection, we show that the positive definiteness of the position kernel and the bound of its eigenvalues.

**Theorem D.10.** *The position kernel* $K(\cdot, \cdot|\tau)$ *defined on* $S_N$ *is positive definite and the eigenvalues of the* $K(A, A)$ *where* $A \subset \mathcal{X}$ *lie between* $\left(\frac{1-\rho}{1+\rho}\right)^N$ *and* $\left(\frac{1+\rho}{1-\rho}\right)^N$ *where* $\rho = \exp(-\tau)$.

*Proof.* We show that the kernel is positive definite on a larger set

$$\mathcal{X} = \prod_{i=1}^{N} \{1, \cdots, N\}.$$

Since $S_N \subset \mathcal{X}$, $K(S_N, S_N)$ is a principal submatrix of $K(\mathcal{X}, \mathcal{X})$ With Poincaré seperation theorem (or Cauchy interlacing theorem), we show that the position kernel is positive definite and that the eigenvalues of $K(S_N, S_N)$ lie between the smallest eigenvalue and the largest eigenvalue of $K(\mathcal{X}, \mathcal{X})$.

43 https://en.wikipedia.org/wiki/Weinstein%E2%80%93Aronszajn_identity
44 https://en.wikipedia.org/wiki/Hadamard%27s_inequality

On $\mathcal{X}$, the position kernel is a product kernel of N kernels defined $\{1, \cdots, N\}$ as below

$$K(\pi_1, \pi_2|\tau) = \exp\Big( -\tau \cdot \sum_i |\pi_1^{-1}(i) - \pi_2^{-1}(i)|\Big).$$

and its gram matrix on each component has following form

$$[\rho^{|i-j|}]_{ij} = \begin{bmatrix} 1 & \rho & \rho^2 & \cdots & \rho^{N-2} & \rho^{N-1} \\ \rho & 1 & \rho & \cdots & \rho^{N-3} & \rho^{N-2} \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ \rho^{N-1} & \rho^{N-2} & \rho^{N-3} & \cdots & \rho & 1 \end{bmatrix}$$

where $\rho = \exp(-\tau)$.

This form of matrix is known as Kac-Murdock-Szegö (KMS) matrix (Grenander and Szegö, 1958; Trench, 1999), which we denote by $KMS(\rho)$ $(0 < \rho < 1)$.

Their eigenvalues $\lambda_n$ are bounded as below (Grenander and Szegö, 1958; Trench, 1999)

$$\lambda_n = \frac{1-\rho^2}{1+\rho^2 - 2\rho\cos(\theta_n)}$$

where

$$\frac{n-1}{N+1}\pi < \theta_n < \frac{n}{N+1}\pi$$

Therefore

$$\frac{1-\rho}{1+\rho} < \frac{1-\rho^2}{1+\rho^2 - 2\rho\cos(\frac{n}{N+1}\pi)} < \lambda_n < \frac{1-\rho^2}{1+\rho^2 - 2\rho\cos(\frac{n-1}{N+1}\pi)} < \frac{1+\rho}{1-\rho}$$

We observe that the each component kernel is positive definite with above bounds on the eigenvalues.

Since

$$K(\mathcal{X}, \mathcal{X}) = \bigotimes_{i=1}^{N} K(\{1, \cdots, N\}, \{1, \cdots, N\})$$

where $\bigotimes$ is the Kronecker product, the lower bound and the upper bound of the eigenvalues of $K(\mathcal{X}, \mathcal{X})$ are $\left(\frac{1-\rho}{1+\rho}\right)^N$ and $\left(\frac{1+\rho}{1-\rho}\right)^N$, respectively.

For $A \in S_N \in \mathcal{X}$, these bounds also apply to the eigenvalues of $K(A, A)$ by Poincaré seperation theorem (or Cauchy interlacing theorem).

Q.E.D. $\square$

### D.2.3 Information Gain of the Position kernel

**Theorem D.11.** *$K(\cdot, \cdot|\tau)$ is the position kernel defined on $S_N$, $\sigma^2_{obs}$ is the variance of the observation noise, $\rho = \exp(-\tau)$ and*

$$D_{max} = \begin{cases} \frac{N^2}{2} & N \textit{ mod } 2 = 0 \\ \frac{N^2-1}{2} & N \textit{ mod } 2 = 1 \end{cases}$$

*Then*

$$\gamma_T \leqslant \frac{1}{2}\min\{A(T), N_{\mathcal{X}} \cdot \log(1+\sigma_{obs}^{-2}\lambda_{max} \cdot T)\}$$

*where*

$$\begin{aligned} A(T) =& \log(1+\sigma_{obs}^{-2}(1+(T-1)\rho^{D_{max}})) \\ &+ (T-1)\log(1+\sigma_{obs}^{-2}(1-\rho^{D_{max}})) \end{aligned}$$

*which is smaller than* $T \cdot \log\det(1+\sigma_{obs}^{-2}\max_{x\in\mathcal{X}} K(x,x))$.

*Proof.* $\gamma_T$ is defined as

$$\frac{1}{2}\max_{A\subset\mathcal{X},|A|=T} \log\det(I+\sigma_{obs}^{-2}K(A,A))$$

By Appx. Lem. D.12, for $i,,j = 1,\cdots,T$, $\rho^{D_{max}} \leqslant [K(A,A)]_{ij} \leqslant 1$.

By Perron-Frobenius theorem, the largest eigenvalue of $K(A,A)$ is bounded below by

$$1+(T-1)\rho^{D_{max}}$$

When $\lambda_i^{(A)}$ is the i-th eigenvalue of $K(A,A)$, with the constraint $\lambda_1^{(A)} \geqslant 1+(T-1)\rho^{D_{max}}$,

$$\prod_{i=1}^{T}(1+\sigma_{obs}^{-2}\lambda_i^{(A)})$$

is bounded above by

$$\begin{aligned} &(1+\sigma_{obs}^{-2}(1+(T-1)\rho^{D_{max}}))\prod_{i=2}^{T}\left(1+\sigma_{obs}^{-2}\frac{T-(1+(T-1)\rho^{D_{max}})}{T-1}\right) \\ &= (1+\sigma_{obs}^{-2}(1+(T-1)\rho^{D_{max}}))\prod_{i=2}^{T}(1+\sigma_{obs}^{-2}(1-\rho^{D_{max}})) \end{aligned}$$

Here, we use the fact that for $\sum_i x_i = C$, $x_i > 0$ if there are $p$ and $q$ such that $x_p < x_q$, then for $x_i'$ defined as $x_i' = x_i$ for $i \neq p, 1$ and $x_p' = x_p + d$, $x_q' = x_q - d$ where $d \leqslant (x_q - x_p)/2$

$$\prod_i x_i \leqslant \prod_i x_i'.$$

Note that without the constraint on the lower bound of the largest eigenvalue,

$$\prod_{i=1}^{T}(1+\sigma_{obs}^{-2}\lambda_i^{(A)}) \leqslant \left(1+\sigma_{obs}^{-2}\frac{trace(K(A,A))}{T}\right)^T$$

where $trace(K(A,A)) = T$ for kernels such that $K(x,x)$ is a constant independent of $x \in \mathcal{X}$ as is for the position kernel.

This shows that the bound in this theorem is tighter than that of Thm. D.9.

Q.E.D. □

*Remark* D.13. specially when $\sigma^2_{obs}$ and/or $\rho$ is large, i.e. $\log(1+\sigma^{-2}_{obs}(1-\rho^{D_{max}})) \approx 0$, we can observe that even in the finite-time regime, the regret is almost sublinear since it is dominated by $\log(1+\sigma^{-2}_{obs}(1+(T-1)\rho^{D_{max}}))$. In this case, the theorem provide a bound which is significantly tighter than the bound in Thm. 6.2 even in the finite-time regime. Even though both are the same in the asymptotic regime, they may differ significantly in the finite-time regime.

**Lemma D.12.** *For* $\pi_1, \pi_2$ *in* $S_N$,

$$d_{pos}(\pi_1, \pi_2) = \sum_i |\pi_1^{-1}(i) - \pi_2^{-1}(i)| \geqslant D_{max}$$

*where*

$$D_{max} = \begin{cases} \frac{N^2}{2} & N \textit{ mod } 2 = 0 \\ \frac{N^2-1}{2} & N \textit{ mod } 2 = 1 \end{cases}$$

*Proof.* Note that $d_{pos}$ is left-invariant, that is,

$$d_{pos}(\pi_1, \pi_2) = d_{pos}(\pi \circ \pi_1, \pi \circ \pi_2)$$

for $\pi \in S_N$, and thus

$$d_{pos}(\pi_1, \pi_2) = d_{pos}(\pi_{id}, (\pi_1)^{-1} \circ \pi_2)$$

where $\pi_{id} = (1, \cdots, N)$.

By induction on N, we show that

$$\max_{\pi \in S_N} d_{pos}(\pi_{id}, \pi) = d_{pos}((1, 2, 3, \cdots, N), (N, N-1, \cdots, 2, 1))$$

BASE CASE$(N = 2)$ This is trivial.

INDUCTION STEP As the induction hypothesis, assume that above is true for $N = k$. When $N = k+1$, let us consider $\pi = (-, -, \cdots, , a) \in S_{k+1}$ an arbitrary permutation whose last element is $a \neq 1$,

$$\begin{aligned} d_{pos}(\pi_{id}, \pi) = & \sum_{i:\pi^{-1}(i)<a, i<k+1} |i - \pi^{-1}(i)| \\ & + \sum_{i:\pi^{-1}(i)>a, i<k+1} |i - \pi^{-1}(i)| + |k+1-a| \end{aligned}$$

where $a \neq N$.

Then

$$\begin{aligned}
&\sum_{i:\pi^{-1}(i)<a, i<k+1} |i-\pi^{-1}(i)| + \sum_{i:\pi^{-1}(i)>a, i<k+1} |i-\pi^{-1}(i)+1-1| \\
&\leqslant \sum_{i:\pi^{-1}(i)<a, i<k+1} |i-\pi^{-1}(i)| \\
&\quad + \sum_{i:\pi^{-1}(i)>a, i<k+1} |i-(\pi^{-1}(i)-1) + \sum_{i:\pi^{-1}(i)>a, i<k+1} 1 \\
&\leqslant d_{pos}((1,\cdots,k),(k,\cdots,1)) + (k-a) \\
&\leqslant d_{pos}((1,\cdots,k),(k,\cdots,1)) + k + (k \bmod 2) \\
&= d_{pos}((1,\cdots,k+1),(k+1,\cdots,1))
\end{aligned}$$

where

$$\begin{aligned}
\sum_{i:\pi^{-1}(i)<a, i<k+1} |i-\pi^{-1}(i)| + \sum_{i:\pi^{-1}(i)>a, i<k+1} |i-(\pi^{-1}(i)-1)| \\
\leqslant d_{pos}((1,\cdots,k),(k,\cdots,1))
\end{aligned}$$

is from the induction hypothesis.

Therefore

$$\max_{\pi\in S_N} d_{pos}(\pi_{id},\pi) = D_{max} = \begin{cases} \frac{N^2}{2} & N \bmod 2 = 0 \\ \frac{N^2-1}{2} & N \bmod 2 = 1 \end{cases}$$

Q.E.D. □

## D.3 IMPLEMENTATION & EXPERIMENT DETAILS

### D.3.1 Submodular Maximization

A set function $g : 2^{\Omega} \to \mathbb{R}$, where $2^{\Omega}$ is the power set of $\Omega$, is submodular when it has the diminishing returns property, that is, for all $P \subset Q \subset \Omega$ and $\mathbf{p} \in \Omega \setminus Q$

$$g(P \cup \{\mathbf{p}\}) - g(P) \geqslant g(Q \cup \{\mathbf{p}\}) - g(Q)$$

As a combinatorial version of convexity (Lovász, 1983), submodularity has been playing a critical role in combinatorial optimization (Fujishige, 2005).

One important property of the submodular function is that when it is positive ($g(\cdot) \geqslant 0$) and monotone ($P \subset Q \implies g(P) \leqslant g(Q)$), its maximization can be performed greedily with an approximation guarantee (Nemhauser et al., 1978) as given below. In the maximization of a positive monotone submodular function with the cardinality constraints, $g(P^*) = \max_{|P|=M} g(P)$, the solution $P^*_{greedy} = \{\mathbf{p}^*_1, \cdots, \mathbf{p}^*_M\}$ from the greedy strategy which sequentially solves $\mathbf{p}^*_m =$

$\mathrm{argmax}_{\mathbf{p}\in\Omega}\, g(\{\mathbf{p}_1^*, \cdots, \mathbf{p}_{m-1}^*, \mathbf{p}\})$ has the following approximation guarantee

$$(1-e^{-1})g(P^*) \leqslant g(P^*_{greedy}) \leqslant g(P^*)$$

In practice, this greedy method often provides almost optimum solutions (Sharma et al., 2015). Moreover, it is possible to relax the conditions (positivity, monotonicity, and even submodularity) (Bian et al., 2017; Feige et al., 2011; Sakaue, 2020).

### D.3.2 Resemblance to Local Penalization

Taken from (González et al., 2016), the local penalization strategy selects b-th point in a batch as follows

$$\mathbf{x}_{t,b} = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} \left\{ g(a_t(\mathbf{x})) \prod_{i=1}^{b-1} \phi(\mathbf{x}, \mathbf{x}_{t,i}) \right\} \tag{D.10}$$

where $\phi(\mathbf{x}, \mathbf{x}_{t,i})$ is a local penalizer which is non-decreasing function of Euclidean distance $\| \mathbf{x} - \mathbf{x}_{t,i} \|_2$ and $g(\cdot)$ is a positive increasing function similar to our weight function.

If we use the prior covariance function $K(\cdot,\cdot)$, which is the kernel of the GP surrogate model in place of the posterior covariance function $K_t(\cdot,\cdot)$ as the diversity gauge of $L_t^{AW}$, the greedy maximization objective becomes

$$\mathbf{x}_b = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} w(a_t(\mathbf{x}))^2 \cdot K(\mathbf{x}, \mathbf{x} | \{\mathbf{x}_i\}_{|b-1|}) \tag{D.11}$$

where $K(\mathbf{x}, \mathbf{x} | \{\mathbf{x}_i\}_{|b-1|}) = K(\mathbf{x}, \mathbf{x}) - K(\mathbf{x}, \{\mathbf{x}_i\}_{|b-1|})(K(\{\mathbf{x}_i\}_{|b-1|}, \{\mathbf{x}_i\}_{|b-1|}) + \sigma^2 I)^{-1}(K(\{\mathbf{x}_i\}_{|b-1|}, \mathbf{x})$. We call this LAW variant as LAW-prior-EST and LAW-prior-EI according to the acquisition function each uses.

Since the closer to the conditioning data it is, the smaller the predictive variance is, the predictive variance behaves exactly as what the local penalizer aims at. Another key difference is that, while the local penalizer Eq. D.10 is heuristically designed, LAW-prior-EST\EI use the kernel whose hyperparameters are fitted in the surrogate model fitting step. Therefore, the diversity measured in LAW-prior-EST\EI is more guided by the collected evaluation data.

Additional comparison to these variants (Appx. D.4) reveals the contribution of the acquisition weights and thus further confirms the benefit of using acquisition weights in the optimization performance

### D.3.3 Benchmarks of optimization on permutations

QUADRATIC ASSIGNMENT PROBLEMS (Koopmans and Beckmann, 1957) Given N facilities $\mathcal{P}$ and N locations $\mathcal{L}$, a distance $d(\cdot,\cdot)$ is given for each pair of locations and a weight $k(\cdot,\cdot)$ is given for each pair of facilities, for example, the cost of delivery between facilities. Then the goal is to find an assignment represented by a permutation $\pi^*$ minimizing $f(\pi) = \sum_{a,b \in \mathcal{P}} k(a,b) \cdot d(\pi(a), \pi(b))$.

Data source (`https://www.opt.math.tugraz.at/qaplib/inst.html`): char12a (Christofides and Benavent, 1989), nug22 (Nugent et al., 1968), esc32a (Eschermann and Wunderlich, 1990)

FLOWSHOP SCHEDULING PROBLEMS (Wikipedia contributors, 2020) There are $N$ machines and $M$ jobs. Each job requires $N$ operations to complete. The $n$-th operation of the job must be executed on the $m$-th machine. Each machine can process at most one operation at a time. Each operation in each job has its own execution specified. Even though jobs can be executed in any order, operations in each job should obey the given order. The problem is to find an optimal order of jobs to minimize execution time. For a formal description, please refer to (Reeves, 1995).

Data source (`http://people.brunel.ac.uk/~mastjjb/jeb/orlib/flowshopinfo.html`): car5 (Carlier, 1978), hel2 (Heller, 1960), reC19 (Reeves, 1995)

TRAVELING SALESMAN PROBLEMS For given cities, a salesman visits each city exactly once while minimizing a given cost incurred in travelling. TSP is the most widely known example of combinatorial optimization on permutations.

Data source (`http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`)

### D.3.4 Normalized Maximum Likelihood

*Model Selection with Minimum Description Length*

In minimum description length (MDL) principle (P. D. Grünwald and Grunwald, 2007), a distribution called a universal distribution is associated with each model class, for example, $\bar{p}_{\mathcal{G}}(\cdot)$ is associated with the $\mathcal{M}^{\mathcal{G}}$, BNs with a given DAG $\mathcal{G}$. For a given data $\mathcal{D}$, model selection can be performed by comparing the universal distribution relative to a model class

$$\bar{p}_{\mathcal{G}_1}(\mathcal{D}) \quad \text{VS} \quad \bar{p}_{\mathcal{G}_2}(\mathcal{D})$$

*Normalized Maximum Likelihood*

Normalized Maximum Likelihood (NML) is regarded as the most fundamental universal distribution (P. Grünwald and Roos, 2019). For the discrete BNs with a DAG $\mathcal{G}$ with the data $\mathcal{D}$, NML is defined as

$$\bar{p}_{\mathcal{G}}(\mathcal{D}) = \frac{p_{BN}(\mathcal{D} \mid \mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D}))}{\sum_{|\mathcal{D}|=|\mathcal{D}'|} p_{BN}(\mathcal{D}' \mid \mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D}'))}$$

where $\hat{\theta}_{ML}(\mathcal{G}, \mathcal{D})$ is the maximum likelihood estimator of the parameters of the BN with the DAG $\mathcal{G}$ on the data $\mathcal{D}$. The summation over all possible data with the same cardinality is the computational bottleneck. The log of the denominator $\mathrm{REG}_{NML}(\mathcal{G}, N) = \log(\sum_{|\mathcal{D}'|=N} p_{BN}(\mathcal{D}'|\mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D}')))$ is called NML regret.[45]

*NML regret estimation*

Even though it is strongly principled, NML computation is restricted to certain classes of models, e.g, multinomial distribution (Kontkanen and Myllymäki, 2007), naive Bayes (Mononen and Myllymäki, 2007), which prevents its use in score-based structure learning. In Bayesian networks, efficient approximations were proposed and shown to perform better in model selection (Roos et al., 2008; Silander et al., 2018).

Even though $\mathrm{REG}_{NML}(\mathcal{G}, N)$ cannot be exactly computed, the summation can be estimated using Monte carlo with proper scaling when BN is discrete.

$$\begin{aligned}
&\log\Big(\sum_{|\mathcal{D}|=|\mathcal{D}'|} p_{BN}(\mathcal{D}'|\mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D}'))\Big) \\
&\approx \log\Big(\frac{\sum_{|\mathcal{D}|=|\mathcal{D}'|} 1}{|\mathcal{S}|}\Big) + \log\Big(\mathit{LSE}_{\mathcal{D}' \in \mathcal{S}} \log(p_{BN}(\mathcal{D}'|\mathcal{G}, \hat{\theta}_{ML}(\mathcal{G}, \mathcal{D}')))\Big)
\end{aligned}$$

where *LSE* is the logsumexp whose implementation increases the numerical stability significantly.[46]

In our scaled MC estimate of $\mathrm{REG}_{NML}(\cdot, \cdot)$, we observed that smaller samples tend to marginally underestimate the value. However, the estimation seems quickly saturated with respect to the sample size. We observed that a MC-estimate of NML regret using $|\mathcal{S}| = 10,000$ is a good compromise between the stability of the estimation and the time needed for the evaluation. With 10,000 samples, the estimation is stable and the difference made by using more samples is marginal to the difference made by the choice of different DAGs. On machines with Intel(R) Xeon(R) CPU E5-2630 v3 2.40GHz, the evaluation time of the objectives (Tab. 6.4) ranges from one minute to four minutes.

## D.4 ADDITIONAL EXPERIMENTAL RESULTS

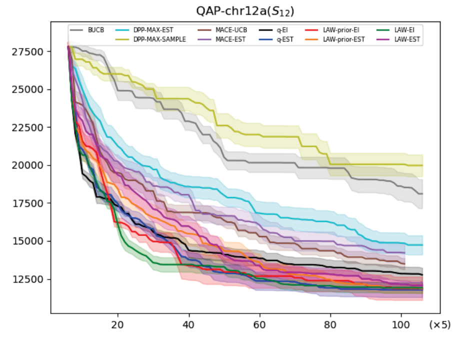
In this section, we provide the additional experimental results which we cannot present in the main text due to the page limit. Following results are presented from the next page

45 Originally, it is called regret but not to confuse with bandit regret, we prefix it with NML.

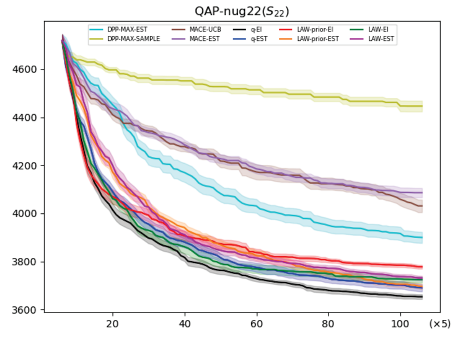
46 `https://pytorch.org/docs/stable/generated/torch.logsumexp.html`

- Comparison with other LAW variants as combinatorial versions of the local penalization(González et al., 2016) (Subsec. 6.5.2)
- Figures of the structure learning experiments (Subsec. 6.5.3)

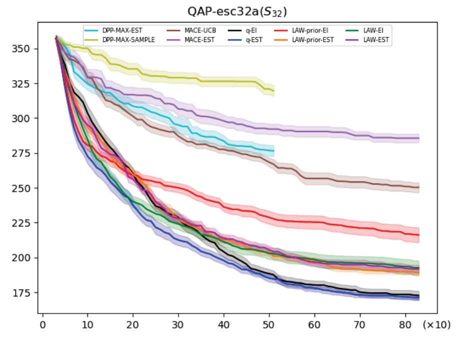
## Quadratic Assignment Problems



| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| BUCB | $+18104.80 \pm 955.15$ | 530 |
| DPP-MAX-EST | $+14731.60 \pm 633.79$ | 530 |
| DPP-SAMPLE-EST | $+19969.60 \pm 718.90$ | 530 |
| MACE-EST | $+14126.13 \pm 596.29$ | 530 |
| MACE-UCB | $+13440.13 \pm 347.78$ | 530 |
| q-EI | $+12769.20 \pm 457.11$ | 530 |
| q-EST | $+11790.13 \pm 497.59$ | 530 |
| LAW-EI | $+11914.40 \pm 345.21$ | 530 |
| LAW-EST | $+12067.07 \pm 237.50$ | 530 |
| LAW-prior-EI | $+11875.67 \pm 771.34$ | 530 |
| LAW-prior-EST | $+11842.53 \pm 301.49$ | 530 |

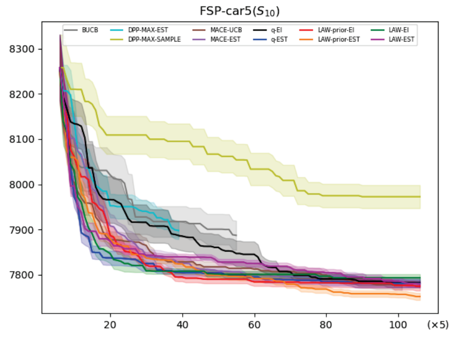


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | $+3899.60 \pm 23.04$ | 530 |
| DPP-SAMPLE-EST | $+4446.13 \pm 22.45$ | 530 |
| MACE-EST | $+4085.87 \pm 19.65$ | 530 |
| MACE-UCB | $+4030.80 \pm 26.37$ | 530 |
| q-EI | $+3653.07 \pm 10.06$ | 530 |
| q-EST | $+3690.00 \pm 15.16$ | 530 |
| LAW-EI | $+3724.00 \pm 12.71$ | 530 |
| LAW-EST | $+3730.93 \pm 9.03$ | 530 |
| LAW-prior-EI | $+3777.47 \pm 7.90$ | 530 |
| LAW-prior-EST | $+3695.47 \pm 11.49$ | 530 |

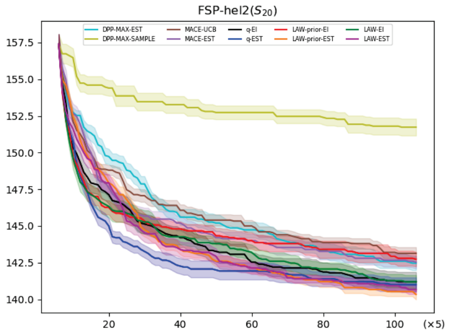


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | $+276.53 \pm 3.87$ | 510 |
| DPP-SAMPLE-EST | $+319.60 \pm 3.78$ | 510 |
| MACE-EST | $+285.60 \pm 3.13$ | 830 |
| MACE-UCB | $+250.27 \pm 3.51$ | 830 |
| q-EI | $+172.67 \pm 3.23$ | 830 |
| q-EST | $+171.20 \pm 1.84$ | 830 |
| LAW-EI | $+192.53 \pm 5.26$ | 830 |
| LAW-EST | $+191.73 \pm 2.89$ | 830 |
| LAW-prior-EI | $+216.13 \pm 5.30$ | 830 |
| LAW-prior-EST | $+188.93 \pm 1.91$ | 830 |

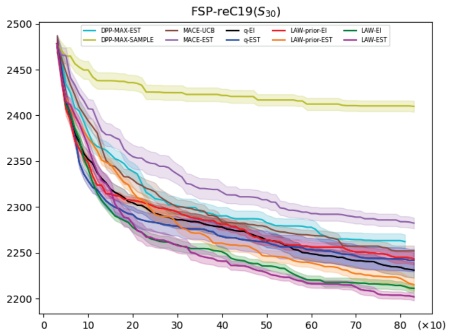
### Flow-shop Scheduling Problems



| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| BUCB | +7887.20 ± 32.37 | 275 |
| DPP-MAX-EST | +7795.67 ± 11.11 | 530 |
| DPP-SAMPLE-EST | +7972.73 ± 25.60 | 530 |
| MACE-EST | +7791.27 ± 9.34 | 530 |
| MACE-UCB | +7775.87 ± 9.73 | 530 |
| q-EI | +7782.67 ± 10.76 | 530 |
| q-EST | +7781.94 ± 9.25 | 530 |
| LAW-EI | +7793.53 ± 7.89 | 530 |
| LAW-EST | +7779.87 ± 7.29 | 530 |
| LAW-prior-EI | +7774.93 ± 9.97 | 530 |
| LAW-prior-EST | +7751.94 ± 7.80 | 530 |

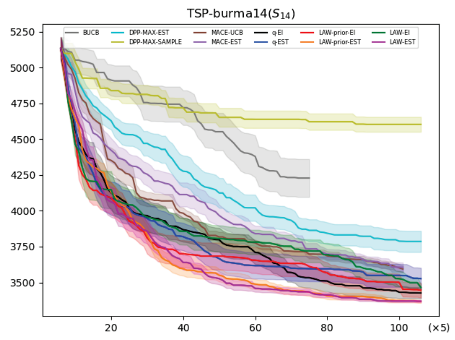


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | +142.47 ± 0.48 | 530 |
| DPP-SAMPLE-EST | +151.73 ± 0.58 | 530 |
| MACE-EST | +142.53 ± 0.45 | 530 |
| MACE-UCB | +143.13 ± 0.42 | 530 |
| q-EI | +141.20 ± 0.66 | 530 |
| q-EST | +141.00 ± 0.49 | 530 |
| LAW-EI | +141.20 ± 0.45 | 530 |
| LAW-EST | +140.67 ± 0.31 | 530 |
| LAW-prior-EI | +142.73 ± 0.49 | 530 |
| LAW-prior-EST | +140.33 ± 0.35 | 530 |

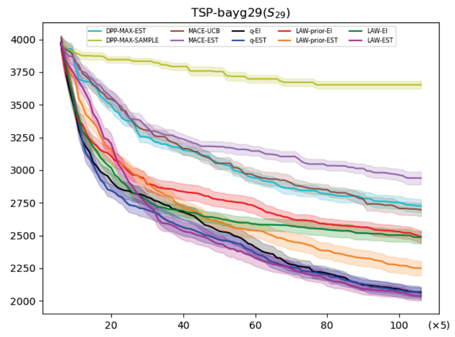


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | +2262.13 ± 7.66 | 810 |
| DPP-SAMPLE-EST | +2409.87 ± 6.09 | 830 |
| MACE-EST | +2282.40 ± 5.86 | 830 |
| MACE-UCB | +2252.00 ± 5.79 | 830 |
| q-EI | +2231.07 ± 8.39 | 830 |
| q-EST | +2241.87 ± 12.06 | 830 |
| LAW-EI | +2211.20 ± 4.47 | 830 |
| LAW-EST | +2202.00 ± 4.17 | 830 |
| LAW-prior-EI | +2243.60 ± 6.56 | 830 |
| LAW-prior-EST | +2215.27 ± 7.20 | 830 |

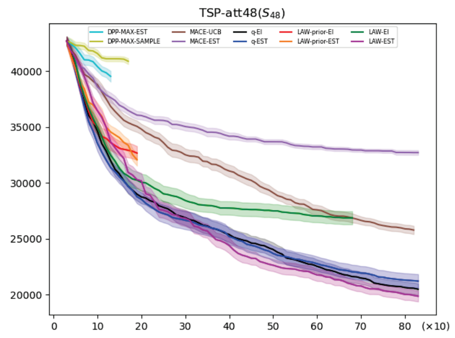
## Traveling Salesman Problems



| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| BUCB | +4184.20 ± 132.13 | 405 |
| DPP-MAX-EST | +3786.00 ± 73.76 | 530 |
| DPP-SAMPLE-EST | +4602.93 ± 52.15 | 530 |
| MACE-EST | +3575.53 ± 25.04 | 530 |
| MACE-UCB | +3582.93 ± 20.93 | 530 |
| q-EI | +3426.53 ± 39.93 | 530 |
| q-EST | +3526.80 ± 75.02 | 530 |
| LAW-EI | +3465.87 ± 25.69 | 530 |
| LAW-EST | +3369.27 ± 7.20 | 530 |
| LAW-prior-EI | +3445.53 ± 51.35 | 530 |
| LAW-prior-EST | +3367.40 ± 10.66 | 530 |

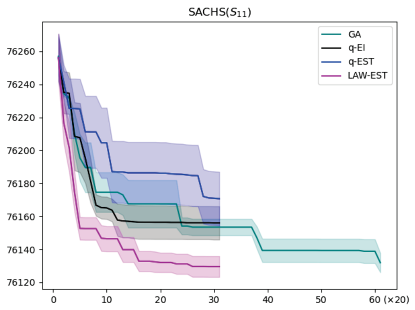


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | +2726.93 ± 50.37 | 530 |
| DPP-SAMPLE-EST | +3652.87 ± 29.48 | 530 |
| MACE-EST | +2939.67 ± 49.04 | 530 |
| MACE-UCB | +2697.93 ± 50.18 | 530 |
| q-EI | +2065.13 ± 36.48 | 530 |
| q-EST | +2059.73 ± 47.93 | 530 |
| LAW-EI | +2486.87 ± 47.36 | 530 |
| LAW-EST | +2038.40 ± 36.28 | 530 |
| LAW-prior-EI | +2491.27 ± 46.49 | 530 |
| LAW-prior-EST | +2250.00 ± 56.72 | 530 |

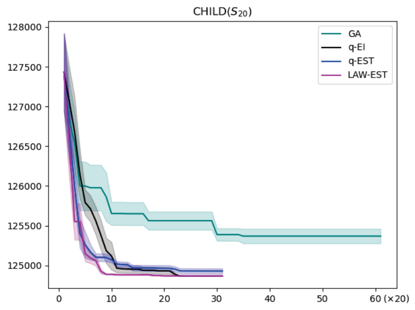


| Method | Mean±Std.Err. | #Eval |
|---|---|---|
| DPP-MAX-EST | +39539.47 ± 486.85 | 130 |
| DPP-SAMPLE-EST | +40893.30 ± 265.03 | 170 |
| MACE-EST | +32710.55 ± 212.19 | 830 |
| MACE-UCB | +25772.51 ± 370.62 | 820 |
| q-EI | +20472.44 ± 502.39 | 830 |
| q-EST | +21199.09 ± 619.65 | 830 |
| LAW-EI | +26864.42 ± 589.32 | 680 |
| LAW-EST | +19846.04 ± 484.86 | 830 |
| LAW-prior-EI | +32670.35 ± 614.68 | 190 |
| LAW-prior-EST | +32072.59 ± 544.12 | 190 |

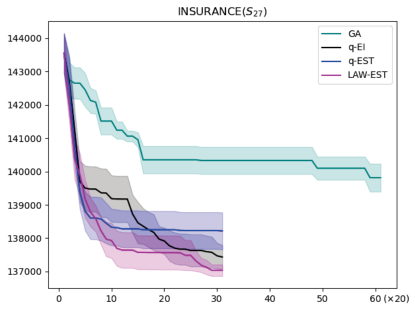
### Structure Learning



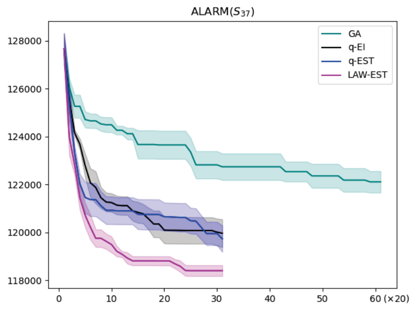
| | | C = 76100 |
|---|---|---|
| Method | #Eval | Mean±Std.Err. |
| GA | 620 | $(C + 53.46) \pm 4.99$ |
| GA | 1240 | $(C + 31.90) \pm 5.86$ |
| q-EI | 620 | $(C + 55.98) \pm 10.11$ |
| q-EST | 620 | $(C + 70.67) \pm 16.31$ |
| LAW-EST | 620 | $(C + 29.58) \pm 6.36$ |



| | | C = 124000 |
|---|---|---|
| Method | #Eval | Mean±Std.Err. |
| GA | 620 | $(C + 1387.12) \pm 79.26$ |
| GA | 1240 | $(C + 1368.07) \pm 92.22$ |
| q-EI | 620 | $(C + 864.85) \pm 0.16$ |
| q-EST | 620 | $(C + 928.83) \pm 32.97$ |
| LAW-EST | 620 | $(C + 866.64) \pm 0.39$ |



| | | C = 135000 |
|---|---|---|
| Method | #Eval | Mean±Std.Err. |
| GA | 620 | $(C + 5330.60) \pm 406.92$ |
| GA | 1240 | $(C + 4814.04) \pm 418.49$ |
| q-EI | 620 | $(C + 2433.23) \pm 357.18$ |
| q-EST | 620 | $(C + 3215.75) \pm 556.36$ |
| LAW-EST | 620 | $(C + 2033.95) \pm 174.04$ |



| | | C = 117000 |
|---|---|---|
| Method | #Eval | Mean±Std.Err. |
| GA | 620 | $(C + 5825.19) \pm 570.55$ |
| GA | 1240 | $(C + 5114.97) \pm 449.93$ |
| q-EI | 620 | $(C + 2969.00) \pm 581.67$ |
| q-EST | 620 | $(C + 2739.77) \pm 554.12$ |
| LAW-EST | 620 | $(C + 1409.27) \pm 227.57$ |