Reconsider Multi-objective Black Box Bayesian Optimization Algorithms

Applied Mathematics and Statistics
Johns Hopkins University

Abstract

Briefly summarized some classical and cutting-edge Multi-objective Black-box Bayesian optimization methods. Then discussed the performance of MESMO, qNEHVI, qEHVI, ParEGO and random search algorithms on synthetic and real-world multi-objective problems. We utilized the Pareto Hypervolume as the measurement of the quality. We summarize the proposed algorithms for different application scenarios based on the experimental results.

1. Introduction

Black-box optimization (BBO) problems usually occur in many situations, where one can obtain zeroth-order evaluations of a function (black-box) that must be optimized over a specified domain. However, the evaluations of this function are computationally expensive and thus the goal is to reduce the number of evaluations.

There is a generic method called Bayesian Optimization (BO) method for optimizing a black-box function. BO method operates sequentially, using past observations to determine the next point. Bayesian Optimization has two main components. The first component is the Probabilistic Model. The probabilistic model is also known as the surrogate function or the posterior distribution. The posterior can capture the updated belief about the unknown objective function. Gaussian Process (GP) is the commonly used Surrogate function to model.

Another component of Bayesian Optimization is the Acquisition Function. The acquisition function can guide the search for the optimum and determine what areas in the domain of f(x) are worth exploiting and what areas are worth exploring. Maximizing the acquisition function can be used to select the next best point at which to evaluate the function. And two typical acquisition functions are the Probability of Improvement (PI) and the Expected Improvement (EI).

2. Related Work and Background

Black-box optimization is meant to address the approach for optimizing unknown functions with high evaluation costs. Bayesian optimization is an efficient method to solve Black-box optimization problems. Using cheap surrogate models, Bayesian optimization method searches for a globally optimal solution by steps of learning probability models and adding points selectively based on a specific acquisition function.

2.1. Gaussian Process

When probability models are defined in the Bayesian Optimization, a typical model for building the unknown function $f(\mathbf{x})$ is Gaussian process [8]. Then the point which maximizes the acquisition function is added in the learned probability model according to Bayes Theory.

The Gaussian Process is a stochastic process of function values $f(\mathbf{x})$ that follows a prior distribution with mean λ and covariance between two points $f(\mathbf{x})$ and $f(\mathbf{x}^{'})$ is defined by a positive definite kernel function $\mathcal{K}(\mathbf{x}, \mathbf{x}^{'})$. In this project, we utilize an effective kernel function proposed by Rasmussen [8], which with the Kernel parameters determined by evidence maximization. We denote the covariance matrix $\mathbb{K}_n$ having elements $[\mathbb{K}_n]_{i,j} = \mathcal{K}(\mathbf{x_i}, \mathbf{x_j})$.

For the given observed dataset $\mathcal{D}_n$, the posterior distribution of $f(x)$ is defined as follows:

$$p(f(\mathbf{x})|\mathcal{D}_\mathbf{n}) = \mathcal{N}(\mu_\mathbf{n}(\mathbf{x}), \sigma^\mathbf{2}_\mathbf{n}(\mathbf{x})), \tag{1}$$

$$\mu_n(\mathbf{x}) = \lambda + \mathcal{K}_\mathbf{n}(\mathbf{x})^\mathbf{T}\mathbb{K}_\mathbf{n}^\mathbf{-1}(\mathbf{f} - \lambda\mathbf{1}) \tag{2}$$

$$\sigma^2_n(\mathbf{x}) = \mathcal{K}(\mathbf{x}, \mathbf{x}) - \mathcal{K}_\mathbf{n}(\mathbf{x})^\mathbf{T}\mathbb{K}_\mathbf{n}^\mathbf{-1}\mathcal{K}_\mathbf{n}(\mathbf{x}) \tag{3}$$

where $\mathcal{K}_n(\mathbf{x}) = [\mathcal{K}(\mathbf{x}, \mathbf{x_1}), \ldots, \mathcal{K}(\mathbf{x}, \mathbf{x_n})]^\mathbf{T}$, and $\mathbf{f} = [\mathbf{f}(\mathbf{x_1}), \ldots, \mathbf{f}(\mathbf{x_n})] \ldots ^\mathbf{T}$, $\mathbf{1}$ is a vector with all ones. In the Bayesian Optimization frame, the posterior distribution $p(f(\mathbf{x})|\mathcal{D}_\mathbf{n})$ is updated when a new observation is added [5].

2.2. Multi-objective Bayesian Optimization

Multi-objective optimization focuses on the problem of minimizing multiple objective functions $f_1(\lambda), \ldots, f_n(\lambda)$. In general, there is no single λ that minimizes all objectives at the same time since the objectives are often in competition with each other. Rather, multiple *Pareto-optimal* solutions can be optimal in that one cannot reduce any f_i without increasing at least one other $f_j (i \neq j)$.

Now, we denote the Bayesian Optimization problem for multi-objective black-box optimization. The unknown function is denoted as $f(\mathbf{x})$ which represents the relation between the objective variable and the explanatory variable $\mathbf{x} \in \mathbb{X}$ where $\mathbb{X}$ is the feasible region of $\mathbf{x}$. Considering the multi-objective optimization problem with d_f objectives. Thus, the optimization target can be collectively represented as $F(\mathbf{x}) = [\mathbf{f^1}(\mathbf{x}), \ldots, \mathbf{f^{d_f}}(\mathbf{x})]$. Let $\mathcal{X}^* \subset \mathcal{X}$ is the Pareto solution set. And, a plane on the range of object variables formed by a set of Pareto solutions is referred to as the Pareto front. The Multi-objective Bayesian Optimization aims to search for a finite number of solutions that closely approximate the Pareto front. This optimization goal can be expressed as below [6]:

$$\begin{aligned}\forall \mathbf{x}, (\mathbf{x}, F(\mathbf{x})) \in \mathcal{D}_\mathbf{n}^* \subset \mathcal{D}_\mathbf{n}, \forall \mathbf{x}', (\mathbf{x}', F(\mathbf{x}')) \in \mathcal{D}_\mathbf{n} \\ \exists k \in \{1, \ldots, d_f\} \text{s.t. } F^k(\mathbf{x}) \leq F^\mathbf{k}(\mathbf{x}')\end{aligned}$$

where $\mathcal{D}_n = (\mathbf{x_1}, F(\mathbf{x_1})), \ldots, (\mathbf{x_n}, F(\mathbf{x_n}))$. The Pareto front is expected to be found by evaluating the maximum improvement when a new point is added to the observation dataset $\mathcal{D}_n$ based on specific criteria function.

2.3. EHVI based acquisition function

A natural acquisition function for Multi-objective Bayesian Optimization is Expected Hypervolume Improvement (EHVI). Given a reference point $\mathbf{r} \in \mathbb{R}^\mathbf{M}$, the hypervolume of a finite approximate Pareto set $\mathcal{P}$ is the M-dimensional Lebesgue measure λ_M of the space dominated by $\mathcal{P}$ and bounded from below by $\mathbf{r}$: $HV(\mathcal{P}, \mathbf{r}) = \lambda_M(\cup_{i=1}^{|\mathcal{P}|}[\mathbf{r}, \mathbf{y_i}])$, where $[\mathbf{r}, \mathbf{y_i}]$ denotes the hyper-rectangle bounded by vertices $\mathbf{r}$ and $\mathbf{y_i}$. Then the hypervolume improvement of a set of points $\mathcal{Y}$ is the hypervolume increased when these points are added into Pareto set $\mathcal{P}$. EHVI is the expectation of HVI over the posterior $P(\mathbf{f}, \mathcal{D})$: $\text{EHVI}(\mathcal{X}_{cand}) = E[\text{HVI}(\mathbf{f}(\mathcal{X}_\mathbf{cand}))]$. EHVI can be approximately computed with Mote Carlo integration based on the approximate expression. Maximizing the hypervolume (HV) has been shown to produce Pareto fronts with good coverage.

2.4. Information-based acquisition function

It's intuitive to compute the entropy $H[p(x^*)]$ of the random variable (the minimizer of an optimization problem) with associated probability density function $p(x^*)$. By minimizing the entropy, we can measure the amount of information that we have about the minimizer x^*. PAL and PESMO [7] are principled algorithms based on information theory. PAL tries to classify the input points based on the learned models into three categories: Pareto optimal, non-Pareto optimal, and uncertain. In each iteration, it selects the candidate input for evaluation with the goal of minimizing the size of the uncertain set. PAL provides theoretical guarantees, but it is only applicable for input space X with a finite set of discrete points. PESMO relies on input space entropy-based acquisition function and iteratively selects the input that maximizes the information gained about the optimal Pareto set. The acquisition function is written as:

$$\alpha(\mathbf{x}) = \mathbf{H}[\mathbf{p}(\mathbf{y}|\mathcal{D})] - \mathbb{E}_{\mathbf{p}(\mathbf{y}|\mathcal{D},\mathbf{x})}[\mathbf{H}(\mathbf{p}(\mathcal{X}^*|\mathcal{D}, \mathbf{x}, \mathcal{X}^*))] \quad (4)$$

where $\mathcal{X}^*$ is the Pareto set, $p(\mathbf{y}|\mathcal{D})$ is the GP predictive distribution for each objective.

3. Possible Approaches

In this section, we briefly introduce several effective state-of-the-art algorithms. Then we will utilize these in this project.

3.1. Parallel Multi-Objective Bayesian Optimization

Since maximizing the hypervolume has been shown to produce Pareto fronts with good coverage, there is a vast body of literature has focused on efficient EHVI computation. However, the time complexity for computing EHVI is still exponential in the number of objectives. Here we introduce the algorithm called parallel q-Expected Hypervolume Improvement (qEHVI) [3]. As introduced before, for each subspace A_i is bounded when calculating HVI, according to the inclusion-exclusion principle. We may compute $\lambda_M(\cup_{i=1}^q)A_i$ with the equation as below:

$$\begin{aligned}\text{HVI}(\mathbf{f}(\mathbf{x_i})_{i=1}^q) &= \lambda_M(\cup_{i=1}^q A_i) \\ &= \sum_{j=1}^q (-1)^{j+1} \sum_{1 \leq i_1 \leq \cdots \leq i_j \leq q} \lambda_M(A_{i_1} \cap \cdots \cap A_{i_j})\end{aligned}$$

Considering box composition, denote $S_k{}_{k=1}^K$ is a disjoint partition, $\lambda_M(A_{i_1} \cap \cdots \cap A_{i_j}) = \sum_{k=1}^K \lambda_M(S_k \cup A_{i_1} \cup \cdots \cup A_{i_j})$. Then we can compute $\lambda_M(S_k \cup A_{i_1} \cup \cdots \cup A_{i_j})$ in a piece-wise fashion across the K hyper-rectangles $S_k{}_{k=1}^K$ as the HV of the intersection of $A_{i_1} \cup \cdots \cup A_{i_j})$ with each hyper-rectangle S_k. With the help of this expression, the vertices in $S_k \cup A_{i_1} \cup \cdots \cup A_{i_j}$ are easily derived resulting in advantages in computation. Moreover, the intersections of subsets $A_{i_1} \cup \cdots \cup A_{i_j}$ for $1 \leq i_j \leq, \ldots, \leq i_j \leq q$ and across all K hyper-rectangles can be performed in parallel. Moreover, qEHVI also completed optimization via sample average approximation. They used the sample average approximation (SAA) approach, which allows the employment of deterministic, high-order optimizers to achieve faster convergence rates. All in all, the qEHVI is expected to be fast and efficient by avoiding determining the Pareto set for each sample by using the inclusion-exclusion principle to compute the joint HVI over the pending points

$x_1, \ldots, x_{i-1}$ and the new candidate and computing HVI in a parallel way.

ParEGO is also a generally utilized EHVI-based algorithm, which uses a MC-based Expected Improvement [38] Acquisition function, where the objectives are modeled independently and the augmented Chebyshev scalarization is applied to the posterior samples as a composite objective. qParEGO allows the use of sequential greedy optimization of q candidates. Compared with the general ParEGO, the speed will be greatly improved.

Besides, qNEHVI [4] is also considered in this project. Although qEHVI is optimized using sequential greedy batch selection, the inclusion-exclusion principle is used over all candidates x1, ..., xi when selecting candidate i. Although the inclusion-exclusion principle could similarly be used to compute qNEHVI, qNEHVI instead leverage CBD, which yields a sequential greedy approximation of the joint (noisy) EHVI that is mathematically equivalent to the inclusion-exclusion principle formulation, but significantly reduces computational overhead.

3.2. Max-value Entropy Search For Multi-objective Bayesian Optimization

As introduced before, information-based acquisition functions are proved effective to search for the optimal solution in multi-objective optimization problems. Most of these algorithms are based on input space entropy, for example, PESMO. However, this process requires a series of approximations, which can be potentially suboptimal, and it is computationally expensive to approximate the information-based acquisition function in input space. MEMSO [2] (Max-value Entropy Search for Multi-objective Optimization) is designed to overcome these drawbacks. It employs an output space entropy-based acquisition function to select the candidate inputs for evaluation. The acquisition function that maximizes the information gain between the next candidate input x and Pareto front $\mathcal{Y}^*$ is given as:

$$\begin{aligned}\alpha((\mathbf{x})) &= I(\{\mathbf{x},\mathbf{y}\},\mathcal{Y}^*|\mathcal{D}) \\ &= H(\mathcal{Y}|\mathcal{D}) - \mathbb{E}_y[H(\mathcal{Y}^*|\mathcal{D}\cup\{\mathbf{x},\mathbf{y}\})] \\ &= H(\mathbf{y}|\mathcal{D},\mathbf{x}) - \mathbb{E}_{\mathcal{Y}^*}[\mathbf{H}(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{Y}^*)]\end{aligned}$$

Then MEMSO can iteratively compute Pareto front samples with a cheap multiple-objective optimization solver and then compute entropy with a sample Pareto front in an approximative way. This algorithm is expected to be computationally efficient and robust to the number of samples.

4. Experiments and Results

In this section, we launch a series of experiments and present different results from different algorithms on diverse synthetic and real-world benchmarks.

4.1. Experimental Setup

Multi-objective BO algorithms We compare the algorithms we introduced before: MESMO, qNEHVI, qEHVI, ParEGO with the random search as a baseline. We use a GP-based statistical model with the same kernel introduced before in all our experiments. We initialize the GP models for all functions by sampling initial points at random from a Sobol grid. These algorithms are completed based on public code sources and Botorch framework [1].

Synthetic problems Here we utilize several synthetic multi-objective benchmark problems to design experiments.

1. **low input dim** (<3) **+ low function dim** (<3). We evaluate the function MCMcCourt02 defined in botorch. The dimension of input space and function are all 2.

2. **high input dim** (≥ 3) **+ low function dim** (<3). We evaluate the function MCMcCourt08 defined in botorch. The dimension of the input space is 12 and the dimension of the function is 2.

3. **low input dim** (<3) **+ high function dim** (≥ 3) We evaluate the function SRDS defined by ourselves. The dimension of the input space is 2 and the dimension of the function is 4. SRDS function is a combination of 4 commonly-used objective functions: Sphere,Rastrigin,DropWave, and sum squares.

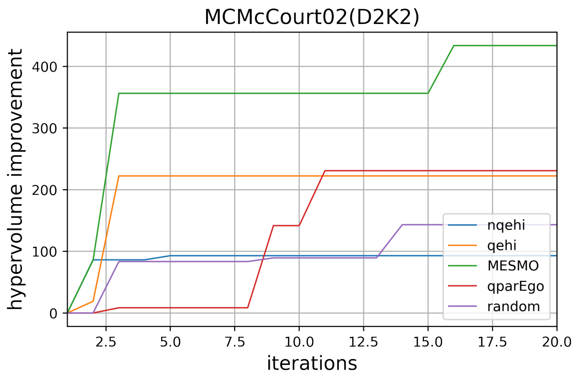
4. **high input dim** (≥ 3) **+ high function dim** (≥ 3). We evaluate the function DTLZ3 function defined in botorch. The dimension of the input space is 5 and the dimension of the function is 4.

Real-world benchmarks Here we consider the Quota related problem, where the data is from a cloud computing platform. The design space of this dataset consists of $d = 18$ variables of users; we optimize the quota based on 2 application-specific criteria metrics. We also have a reduced space with this problem, which has an input dimension equal to 2 and a function dimension equal to 2.

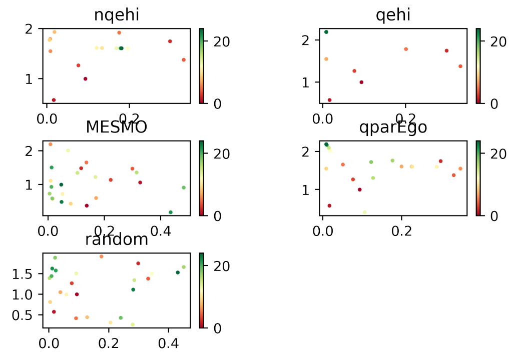
Evaluation metrics We utilize the Pareto Hypervolume as the measurement of the quality of a given Pareto front. After each iteration t, we report the hypervolume improvement of the estimated Pareto front ($\mathcal{Y}_t$) by a given algorithm. Also, for the low-dimension problem, we can plot the distribution of the Pareto front, which will help us clearly see the point scatter.

4.2. Results and Discussion

low input dim (<3) **+ low function dim** (<3). We run all the algorithms 5 trials. For each algorithm, we have the same hyperparameter setting. The hypervolume improvement is reported as a function of the number of iterations.



(a) Hypervolume Improvement.



(b) Pareto Front.

Figure 1. Results of MCMcCourt02 function (all algorithms).

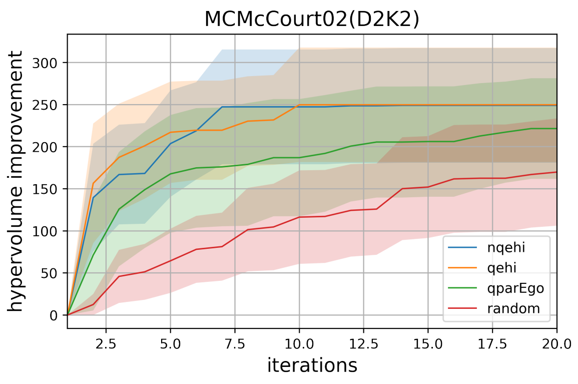
From the Figure 1, we can easily see that the MESMO method outperforms others with faster convergence and greater hypervolume improvement and clearer Pareto front edge. However, the computational time of MESMO is much longer than other methods, and its convergence depends on the sampling of MCMC, but too many sampling points will take longer. An inappropriate number of sampling points as well as the number of iterations can even lead to convergence failure. To better evaluate other algorithms, we increase the number of batches and trials of the algorithm and increase the number of sampling points, even so, the computation time of the algorithm is much lower. Then we got a more stable result of nqEHI,qEHI, qParEgo, and random, as shown in Figure 2. The nqEHI has almost the same result as qEHI, but with faster convergence. This coincides with the mathematical equivalence of the two methods. Also, all methods significantly outperformed random search, confirming the effectiveness of these methods in this scenario. However, it is worth noting that these methods with adjusted parameters still do not reach the result of MESMO.Even though the computational time cost is significant, MESMO is very powerful.

high input dim (≥ 3) **+ low function dim** (<3)**.** We run all the algorithms 5 trials. For each algorithm, we have the same hyperparameter setting. The hypervolume improvement is reported as a function of the number of iterations. In Figure 3, we can notice that the MESMO method is not effective as it is in the last problem. We suspect that this is due to the dependence on MCMC sampling points. Then, since the other algorithms are much faster, we continue to increase the number of iterations and the number of sampling points obtained by the other algorithms. The results are shown in Figure 4. It can be seen that the poor performance of the other algorithms in Figure 3 may be due to the fact that the desired convergence state has not been reached yet. The nqEHI still has good performance with fast convergence and high hypervolume improvement.

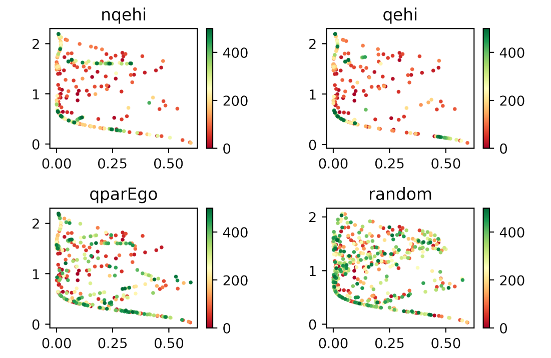
low input dim (<3) **+ high function dim** (≥ 3) After launching experiments, we found that the EHVI based methods and random search methods failed in this case. They have almost difficulty converging, and it is evident that they are having difficulty with the higher dimensional function objectives. The results are shown in Figure 5. In order to evaluate the results with more certainty, we increased the iteration parameters and MCMC sampling points. The final results are shown in Figure 5. We can see the information-based methods(MESMO and ParEgo) can still work. MESMO is still advantaged, while the ParEgo has a much faster computational speed and easy parameters adjustment.

high input dim (≥ 3) **+ high function dim** (≥ 3)**.** Since EHVI-based methods don't work with the high function dimension, we only utilize the ParEgo and MESMO algorithms to do the comparison. The result is shown in Figure 6. When the dimension of input and function is high, it is hard to run the MESMO, while the ParEgo converges fast and has a fairly good result.

Real-world benchmarks. According to the results we got before, we use nqEHI here rather than qEHI because it is generally more efficient than qEHI and mathematically equivalent in the noiseless setting. Then we compare the nqEHI, MESMO and ParEgo in both the full space and reduced space. The result is shown in Figure 7. The results almost align with our expectations. In the reduced space, ParEgo, nqEVI and MESMO all have relatively good results, while MESMO have more expensive computational cost and tricky hyperparameter adjustment. In the full space, MESMO have advantages compared with other algorithms. Since this real-world problem has high input dimension but low function dimension, the nqEHI still works well with large iterations. Overall, the nqEHI has the fastest convergence speed and stable parameter setting.

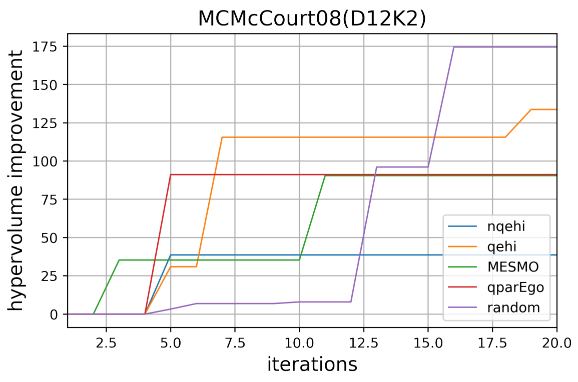


(a) Hypervolume Improvement.

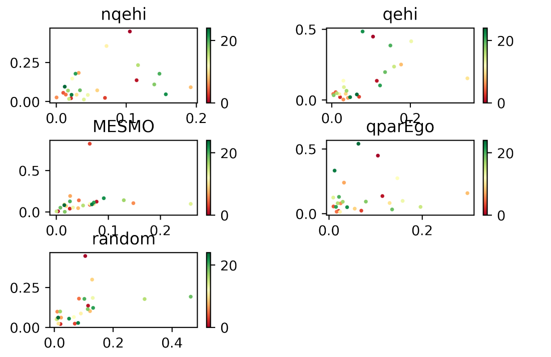


(b) Pareto Front.

Figure 2. Results of MCMcCourt02 function.



(a) Hypervolume Improvement.



(b) Pareto Front.

Figure 3. Results of MCMcCourt08 function (all algorithms).

5. Conclusion and Open discussion

Based on the above discussion and experiments, we discuss and have a simple summary here. nqEVI is highly recommended for low dimensional cases, especially for low dimensional functions, it has the same mathematical principle as nEVI, but it is optimized for better speed and performance. In some high-dimensional complex cases and with sufficient computational power, MESMO is recommended even though it is computationally expensive. And the hyperparameters can be adjusted by methods such as grid search to obtain stable results. However, these methods are still difficult to guarantee good performance for some black-box problems, and it may be a good open direction to consider some integrated method design, but this may reduce the interpretability of the method. In conclusion, better method design and framework integration as well as more comprehensive open experiments are needed in this area. And we are so glad to have a basic understanding of this problem via this course.

References

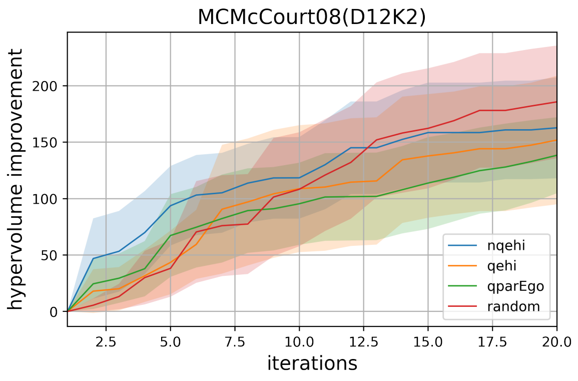
[1] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020. 3

[2] Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Max-value entropy search for multi-objective bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019. 3

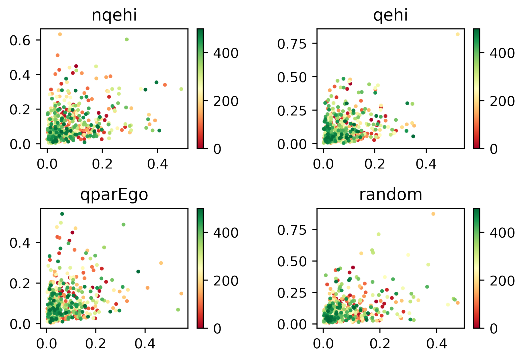
[3] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. *Advances in Neural Information Processing Systems*, 33:9851–9864, 2020. 2

[4] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement. *Advances in Neural Information Processing Systems*, 34:2187–2200, 2021. 3

[5] Eduardo C Garrido-Merchán and Daniel Hernández-Lobato. Parallel predictive entropy search for multi-objective

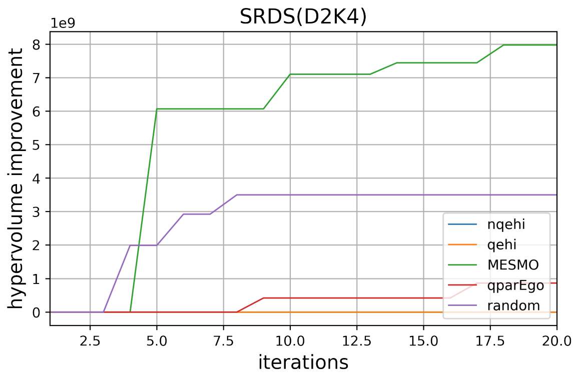


(a) Hypervolume Improvement.

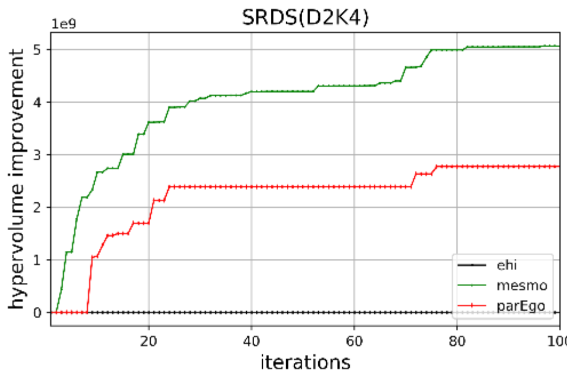


(b) Pareto Front.

Figure 4. Results of MCMcCourt08 function.



(a) Hypervolume Improvement.



(b) Hypervolume with lager iterations.

Figure 5. Results of SRDS function.

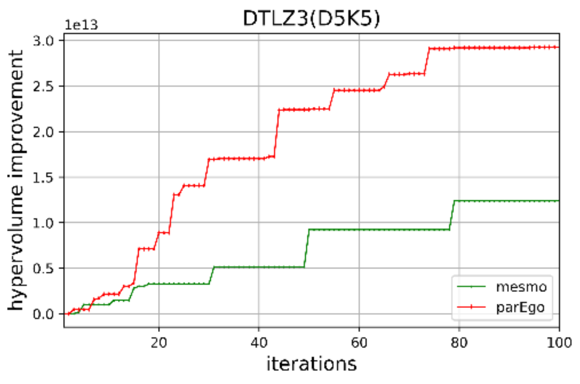


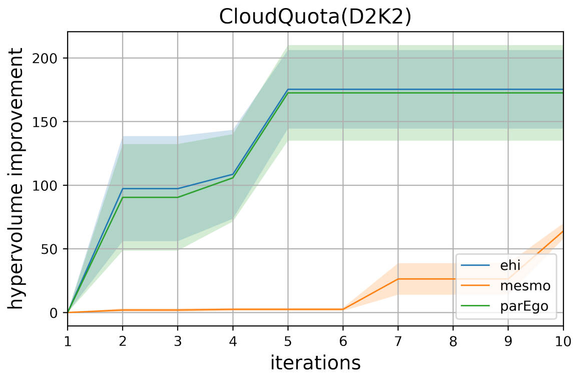
Figure 6. Result of DTLZ function

bayesian optimization with constraints. *arXiv preprint arXiv:2004.00601*, 2020. 1

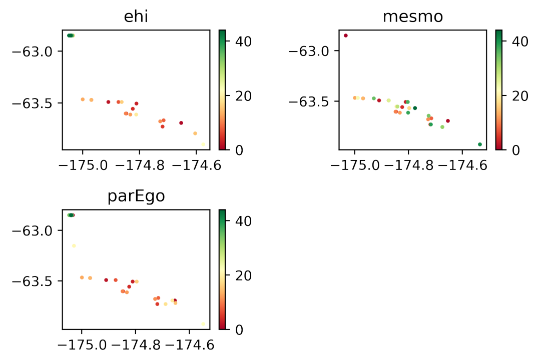
[6] Julia Guerrero-Viu, Sven Hauns, Sergio Izquierdo, Guilherme Miotto, Simon Schrodi, Andre Biedenkapp, Thomas Elsken, Difan Deng, Marius Lindauer, and Frank Hutter. Bag of baselines for multi-objective joint neural architecture search and hyperparameter optimization. *arXiv preprint arXiv:2105.01015*, 2021. 2

[7] Daniel Hernández-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective bayesian optimization. In *International conference on machine learning*, pages 1492–1501. PMLR, 2016. 2

[8] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006. 1

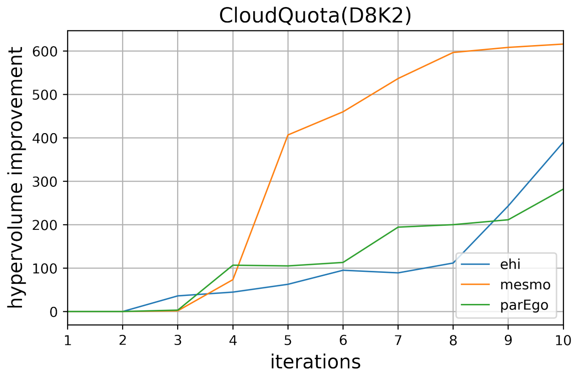


(a) Hypervolume Improvement.

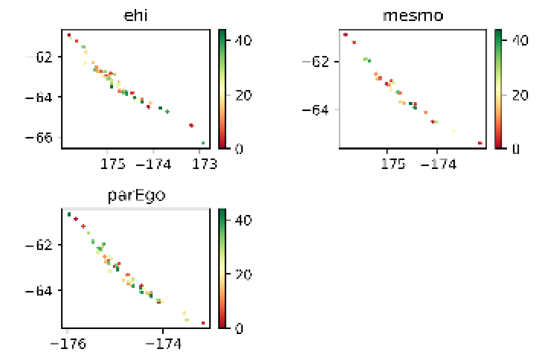


(b) Pareto Front

Figure 7. Results of Real-world dataset in reduced space.



(a) Hypervolume Improvement.



(b) Pareto Front

Figure 8. Results of Real-world dataset in full space.