Journal of Ambient Intelligence and Humanized Computing (2023) 14:11263–11271
https://doi.org/10.1007/s12652-023-04640-7

ORIGINAL RESEARCH

Wasserstein enabled Bayesian optimization of composite functions

Antonio Candelieri[1] · Andrea Ponti[1] · Francesco Archetti[2]

Received: 7 October 2022 / Accepted: 8 May 2023 / Published online: 22 May 2023

Abstract

Bayesian optimization (BO) based on the Gaussian process model (GP-BO) has become the most used approach for the global optimization of black-box functions and computationally expensive optimization problems. BO has proved its sample efficiency and its versatility in a wide range of engineering and machine learning problems. A limiting factor in its applications is the difficulty of scaling over 15–20 dimensions. In order to mitigate this drawback, it has been remarked that optimization problems can have a lower intrinsic dimensionality. Several optimization strategies, built on this observation, map the original problem into a lower dimension manifold. In this paper we take a novel approach mapping the original problem into a space of discrete probability distributions endowed with a Wasserstein metric. The Wasserstein space is a non-linear manifold whose elements are discrete probability distributions. The input of the Gaussian process is given by discrete probability distributions and the acquisition function becomes a functional in the Wasserstein space. The minimizer of the acquisition functional in the Wasserstein space is then mapped back to the original space using a neural network. Computational results for three test functions with dimensionality ranging from 5 to 100, show that the exploration in the Wasserstein space is significantly more effective than that performed by plain Bayesian optimization in the Euclidean space and its advantage grows with the dimensions of the search space.

Keywords Bayesian optimization · Wasserstein distance · Gaussian processes

1 Introduction

1.1 Motivation

The challenge of Bayesian Optimization (BO) in high dimensional problems has been addressed mapping it into low-dimensional problems defined on subsets of variables. Kandasamy et al. (2015), Moriconi et al. (2020) or exploiting a lower intrinsic dimensionality. To tackle the issue of high dimensionality a different approach is proposed in the present paper mapping the original problem into a space of discrete probability distributions.

We consider the optimization of a black-box, expensive, multi-extremal function $f(x)$:

$$f(x) : x \in X \subset \mathbb{R}^d \to R \tag{1}$$

where $\mathcal{X}$ is the search space and neither gradient nor convexity information are available.

Consider the following composite function for $i = 1, \dots, n$

$$f(x) = C\left(h_1(\mathrm{x}), \dots, h_n(\mathrm{x})\right) \tag{2}$$

where $h_1(\mathrm{x}), \dots, h_n(\mathrm{x})$ is the univariate point cloud associated to x.

In the specific case of a linear scalarization of a multi-objective problem, $f(x) = \sum_{i=1}^{n} \lambda_i h_i(x)$ and each vector $h_1(\mathrm{x}), \dots, h_n(\mathrm{x})$ is the point cloud associated to x. Another class of problems which yield naturally a distributional representation of a candidate solution are simulation–optimization problems. This is the case in which the objective function is the average performance of a system where $f(x, w)$ is the value of $f(x)$ under the environmental condition w and $p(w)$ represents the "relevance" of condition w (probability of its occurrence or the fraction of time that condition w occurs). Another setting is the hyperparameter optimization of a machine learning algorithm via k-fold cross validation, with $f(x, w)$ a loss function (e.g., predictive accuracy, fairness, explainability, etc.) on

✉ Andrea Ponti
andrea.ponti@unimib.it

[1] Department of Economics, Management and Statistics, University of Milano-Bicocca, Milan, Italy

[2] Department of Computer Science, Systems and Communication, University of Milano-Bicocca, Milan, Italy

fold w using hyperparameter configuration x. The point clouds lay onto a metric space—i.e., the space of discrete probability distributions—in which a metric defines the distance between two points in that space, with the properties of positiveness, symmetry, and triangle inequality. Due to the nature of elements belonging to this space, the most appropriate distance between them is a distance between probability distributions. In this paper we focus on the Wasserstein (WST) distance and embed the original optimization problem in the metric space whose elements are discrete probability distributions, which we call Wasserstein space $\mathcal{W}$. Wasserstein distance, also known as the Optimal Transport (OT) distance, is a mathematically principled method to align probability distributions. Originated by a paper of Monge (1781), it received its linear programming formulation in Kantorovich (1942). A complete mathematical formulation is in Villani (2009) while Peyré and Cuturi (2019) offer a complete review of recent theoretical and computational advances. The Wasserstein distance has been widely applied in machine learning from shape analysis (Gangbo and McCann 2000) to image interpolation, domain adaptation (Redko et al. 2019), parameter estimation in simulation models (Öcal et al. 2019), structured data on graphs (Vayer et al. 2018), active learning (Frogner et al. 2019), and adversarial networks (Arjovsky et al. 2017). The Wasserstein distance has many important properties: its representational capability has been shown by embedding in $\mathcal{W}$ a variety of complex objects like images, networks, and words. An explanation of the interest in the Wasserstein distance is that Euclidean embeddings of data are flawed as they account for the correspondence of each feature independently of the other features. Bayesian Optimization (BO) algorithms have so far largely focused on problems where inputs are represented as numerical and categorical variables in Euclidean spaces. A significant advance is provided in Jaquier and Rozo (2020) which extends BO to Riemannian manifolds.

In this paper we extend the distributional approach to BO by encoding the geometry of the data generated in the sequential optimization process and performing the search in $\mathcal{W}$. The key advantage of BO is its well-known sample efficiency. The main question considered in our study is whether its sample efficiency can be further improved by embedding the optimization process in $\mathcal{W}$. An important result is the development of a multi-layer perceptron (MLP) to map the results obtained by BO in $\mathcal{W}$ back to original search space $\mathcal{X}$. The resulting algorithm BOWS (Bayesian Optimization in the Wasserstein Space)**,** at least for the test functions considered, outperforms "Euclidean" BO already in 10 dimensions and its competitive edge increases substantially as the dimension of the search space increases. We have only considered the case in which the probability measures are univariate discrete probability distributions (aka point clouds).

1.2 Related works

The use of Gaussian Processes with probabilistic inputs has been proposed in Candelieri et al. (2022a), but the use of the WST distance in optimization problems is still a sparsely explored field. The issue of placing optimization in the space of probability distributions has been analyzed in Zhang et al. (2018) where policy optimization in reinforcement learning is modelled using Wasserstein gradient flows and Zhang et al. (2019) where the problem of approximating the posterior distribution in Thompson sampling is solved via Wasserstein gradient flow providing also a theoretical guarantee of convergence. Since Thompson sampling (TS) is used both for sampling a Gaussian process as also as acquisition function in the Bayesian optimization framework, an optimal transport based efficient computational strategy for performing TS is directly relevant for optimization. TS is a sequential optimization process based on the following steps: updating a posterior depending on the set of observations, drawing a sample from posterior as an approximation to the function to be minimized, minimizing this sample function to identify the next candidate point and evaluating the objective function at that point. However, calculating exact posterior distributions is intractable for all but the simplest models. Therefore, the development of computationally efficient approximate methods for the posterior distributions is a crucial problem for scalable TS.

In Gong et al. (2019) and Liu and Wang (2016) it is shown how batch-BO enables to transform the optimization of the acquisition function into finding the optimal distribution in the space of all distributions. The resulting quantile variational optimization is then solved using Stein variational gradient descent. The use of gradient flows in the Wasserstein space has been proposed in Salim et al. (2020) for the identification of OT maps. Wasserstein gradient flows have been also suggested in Rout et al. (2021) and Liutkus et al. (2019) for solving the optimization problems arising in generative modelling. Another problem which has been formulated as optimization over data-generating joint probability distributions is the dataset transformation from unlabelled to labelled (Alvarez-Melis and Fusi 2021) using a particle-based method. The same approach has been proposed in Alvarez-Melis and Fusi (2020) for transfer learning via OT.

The theoretical framework of the previous papers has been reconsidered and focused on BO (Crovini et al. 2022) where the authors propose a batch sequential algorithm based on the Expected Improvement (EI) acquisition function, which is transformed into an acquisition functional defined over a space of probability measures. The key result is that this functional is concave, according to the strong factorization assumption that the probability measure of the batch points takes the form of a product measure. However, the concavity result is derived only for the batch-EI. The optimization of the acquisition functional is then based on its gradient flow over $\mathcal{W}$. Two formulations of the gradient flow on the space of probability measures are considered: the Stein gradient flow and the particle-based Wasserstein gradient flow. The estimation of batch-EI and the computation of its gradient flow are quite complex involving the solution of the non-linear Fokker–Plank equation.

Other results are related to BO over Riemannian manifolds (Jaquier and Rozo 2020) focused on Robot Learning and Jacquier et al. (2020) focused on high dimensional BO, proposing an approach that builds, on the theory of Riemannian manifolds, a representation of the objective function in a low dimensional latent space.

The issue of Distributionally Robust Optimization (DRO) is analysed in Lau and Liu (2022) who propose a *Wasserstein barycentric ambiguity set* and (Liu et al. 2022). Closer to the focus of our paper is Kandasamy et al. (2018) which use a kernel induced by the WST distance in a BO framework to search for the best neural network architecture.

Another possible approach for learning from distributions is to consider Reproducing Kernel Hilbert Spaces (RKHS). The kernels associated to probability distributions, in particular the Hilbertian kernel on probability measures have been first proposed in Hein and Bousquet (2005). A solution to the problem in the setting of Hilbert spaces has been provided in Peyré and Cuturi (2019). It must be remarked that in the case of multivariate distributions the construction of positive definite kernels on sets of probability measures is not straightforward.

1.3 Our contributions

A key contribution of this paper is to show that mapping candidate solutions from the search space into univariate discrete probability measures, specifically point clouds, associated to the components of the objective function, can be applied to obtain a BO algorithm where the Gaussian process and the acquisition function are defined over $\mathcal{W}$. The mapping back from $\mathcal{W}$ into the original search space $\mathcal{X}$ is accomplished by a neural network. An indication of the convergence is obtained from a measure of concentration around the global optimum in $\mathcal{W}$ as the ambiguity set built upon the WST distance between point clouds. Preliminary computational results on additive benchmark functions show that the relative performance of the BOWS algorithm improves over plain BO, both in terms of function evaluations and wall-clock time, as the dimension of the search space $\mathcal{X}$ increases.

1.4 Organization of the paper

The contents of the paper are organized as follows. Section 2 provides background knowledge about Wasserstein distance and the optimal transport formulations. Section 3 establishes the BOWS algorithm and proposes a neural network which maps the probability distributions from $\mathcal{W}$ into $\mathcal{X}$. Section 4 describes the experimental set-up including the algorithms considered and the parameters values for benchmarking BOWS and the computational results over the test functions. Section 5 provides conclusions and perspectives.

2 Methodological background

2.1 The Wasserstein distance between point clouds

Consider two univariate point clouds, respectively denoted with $\mathbf{H} = \left(h^{(1)}, \dots, h^{(n)}\right)$ and $\mathbf{G} = \left(g^{(1)}, \dots, g^{(m)}\right)$. Since the WST distance is originally defined between two probability measures, the two point clouds are mapped into discrete probability distributions. Given a point cloud $\mathbf{H}$, the associated probability measure is given by:

$$\alpha = \frac{1}{n}\sum_{i=1}^{n} \delta_{h^{(i)}} \tag{3}$$

Given two point clouds, their WST distance is $W_2(\mathbf{H},\mathbf{G}) = \min\limits_{P\in U(n,m)} \sqrt{\langle P, C^2\rangle}$, with $C^2 \in \mathfrak{R}^{n\times m}$ is the cost matrix between the points of the two clouds and P is the coupling matrix where $P_{i,j}$ denotes the weight of the assignment of $h^{(i)}$ to$g^{(j)}$, and U is the set of all the possible assignments, that is $U(n,m) = \left\{P \in \mathfrak{R}_{+}^{n\times m} : P1_m = \frac{1}{n}1_n, P^T 1_n = \frac{1}{m}1_m\right\}$.

The optimal coupling $P^* = \operatorname*{argmin}\limits_{P\in U(n,m)} \sqrt{\langle P, C^2\rangle}$ can be obtained through the simplex algorithm.

Since our case study considers univariate point clouds with the same cardinality (i.e., the number n of objective function's components), the computation of $W_2(\mathbf{H},\mathbf{G})$ can be simplified as:

$$W_2(\mathbf{H},\mathbf{G}) = \left(\frac{1}{n}\sum\nolimits_i^n \left|\widetilde{h}^{(i)} - \widetilde{g}^{(i)}\right|^2\right)^{\frac{1}{2}} \tag{4}$$

where $\widetilde{h}^{(i)}$ and $\widetilde{g}^{(i)}$ denote the sorted samples of the two point clouds.

2.2 The "vanilla" Bayesian optimization

A Gaussian Process (GP) is a probability distribution over functions denoted as $f(x) \sim GP(\mu(x), k(x, x'))$ where $\mu(x)$ is the mean function of the GP and $k(x, x')$ is the covariance function (aka kernel). Therefore, GP is as a collection of random variables, any finite number of which have a joint Gaussian distribution and $f(x)$ can be considered as a sample from a multivariate normal distribution (Archetti and Candelieri 2019; Frazier 2018).

Let denote with $\mathrm{X}_{1:N} = \{\mathbf{x}^{(i)}\}_{i=1,\dots,N}$ a set of N points in $\Omega \subset \mathfrak{R}^d$ and with $y_{1:N} = \{f(\mathbf{x}^{(i)}) + \varepsilon\}_{i=1,..,N}$ the associated function values, possibly noisy with ε a zero-mean Gaussian noise $\varepsilon \sim \mathcal{N}(0, \lambda_\varepsilon^2)$. Then the posterior predictive mean $\mu(\mathbf{x})$ and standard deviation $\sigma^2(\mathbf{x})$, conditioned on $\mathrm{X}_{1:N}$ and $y_{1:N}$, are given by the following equations:

$$\mu(\mathbf{x}) = k(\mathbf{x}, \mathrm{X}_{1:N})\left[\mathrm{K} + \lambda_\varepsilon^2 I\right]^{-1} y_{1:N} \tag{5}$$

$$\sigma^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - k(\mathbf{x}, \mathrm{X}_{1:N})\left[\mathrm{K} + \lambda_\varepsilon^2 I\right]^{-1} k(\mathrm{X}_{1:N}, \mathbf{x}) \tag{6}$$

where $\mathrm{k}(\mathbf{x}, \mathrm{X}_{1:N}) = \{k(\mathbf{x}, \mathbf{x}^{(i)})\}_{i=1:N}$ and $\mathrm{K} \in \mathfrak{R}^{N \times N}$ with entries $\mathrm{K}_{i,j} = k(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$.

The acquisition function manages the balance between exploration and exploitation, it is the key driver of the sample efficiency of BO and is an important concept also outside machine learning (Candelieri et al. 2021). It drives the search of the new evaluation points towards regions of the search space with potential better values of the objective function either because value of $\mu(\mathbf{x})$ is better or the uncertainty represented by $\sigma^2(\mathbf{x})$ is high (or both). A widely used acquisition function is the Confidence Bound (Lower and Upper, respectively for minimization and maximization problems):

$$UCB(\mathbf{x}) = \mu(\mathbf{x}) + \xi^{\frac{1}{2}}\sigma(\mathbf{x}) \tag{7}$$

$$LCB(\mathbf{x}) = \mu(\mathbf{x}) - \xi^{1/2}\sigma(\mathbf{x}) \tag{8}$$

where ξ is the parameter to manage the exploration/exploitation trade-off.

3 The Bayesian optimization in Wasserstein space algorithm

3.1 Preliminaries

First, we recall here, and also introduce, some useful notations:

- $\mathcal{X} \subset \mathbb{R}^d$ is the Euclidean original search space.
- $y \in \mathbb{R}$ is the co-domain of the objective function $f(\mathbf{x})$.
- $\mathcal{W} \subset \mathbb{R}^n$ is the (unknown) co-domain of the objective function's observable components, that are $h_1(\mathbf{x}), \dots, h_n(\mathbf{x})$—or compactly the point cloud $\mathbf{H} \in \mathcal{W}$.
- $\mu(\mathbf{H})$ and $\sigma^2(\mathbf{H})$ are the predictive mean and variance of a GP defined over the space of point clouds, $\mathcal{W}$, and computed according to (5) and (6) where $\mathbf{x} \in \mathcal{X}$ is replaced by$\mathbf{H} \in \mathcal{W}$.
- $\varphi : \mathcal{W} \to \mathcal{X}$ is a mapping from the space univariate probability distributions back to original search space.

3.2 The BOWS's GP

For the GP model, we have decided to adopt the (Euclidean) Squared Exponential (SE) kernel, operating on the space $\mathcal{W}$, that is the n-dimensional space whom the point clouds belong to. Specifically, the (Euclidean) SE kernel is:

$$k(\mathbf{H}, \mathbf{H}') = e^{-\frac{\|\mathbf{H}-\mathbf{H}'\|^2}{2\updownarrow^2}}$$

with $\updownarrow$ the so-called length-scale hyperparameter which is tuned via MLE. If $\updownarrow \in \mathfrak{R}$ the kernel is said *isotropic*, while it is anisotropic if $\updownarrow \in \mathfrak{R}^n$.

Although using a Euclidean-based kernel on $\mathcal{W}$ can seem a contradiction (Candelieri et al. 2022b) prove that using a Euclidean SE kernel between univariate probability measures is equivalent to using a *non-stationary anisotropic* Wasserstein-based SE kernel, that is:

$$k(\mathbf{H}, \mathbf{H}') = e^{-\frac{W_2^2(\boldsymbol{H}, \boldsymbol{H}')}{2\updownarrow^2}}$$

with $W_2^2(\boldsymbol{H}, \boldsymbol{H}')$ computed as in (4).

3.3 The BOWS's acquisition function

As the test problems considered in the paper are minimization problems, we will use LCB as acquisition function. The main difference with respect to the vanilla BO is that here LCB is defined—as well as the GP—over $\mathcal{W}$ instead of $\mathcal{X}$. Thus, minimizing LCB leads to the next point cloud $\widehat{\mathbf{H}}^{(N+1)}$ giving the best exploration–exploitation trade-off, that is:

$$\widehat{\mathbf{H}}^{(N+1)} = \underset{\mathbf{H} \in \Omega \subset \mathcal{W}}{\operatorname{argmin}} LCB(\mathbf{H}) \tag{9}$$

It is important to remark that, contrary to the search space $\mathcal{X}$ that is defined by the user and usually box-bounded, there not exists any preliminary information about $\Omega \subset \mathcal{W}$. The unknown search space problem is intractable to solve in practice. Therefore, we decided to dynamically set up the search space Ω, according to the point clouds observed so far. This kind of procedure—which is mandatory in our case—it has been anyway proposed in vanilla BO, quite recently and named "weakly specified" search space (Nguyen et al. 2017).

3.4 Mapping from $\mathcal{W}$ back to $\mathcal{X}$

We need to map $\widehat{\mathbf{H}}^{(N+1)}$ back to $\mathcal{X}$ to obtain the associated value $\mathbf{x}^{(N+1)}$ and, consequently $y^{(N+1)}$ and the actual $\mathbf{H}^{(N+1)}$. Indeed, it is important to remark that any possible mapping is anyway affected by some reconstruction error. The mapping $\varphi : \mathcal{W} \rightarrow \mathcal{X}$ is performed by a MLP trained using the sets $\mathcal{H}_{1:N} = \left\{\mathbf{H}^{(i)}\right\}_{i=1:N}$ and $\mathrm{X}_{1:N} = \left\{\mathbf{x}^{(i)}\right\}_{i=1:N}$ as input and output, respectively. The number of layers of the MLP has been set to three and in each layer the number of neurons is $\max(n, d)$. The MLP is retrained at each iteration or after a given number of iterations, according to the user's preferences and available computational budget. On the contrary, the GP model is always trained at each iteration (as usual in BO). The MLP model provides the new point $\mathbf{x}^{(N+1)}$ which yields $y^{(N+1)} = f\left(\mathbf{x}^{(N+1)}\right)$ and concurrently the actual $\mathbf{H}^{(N+1)}$.

The additional computational complexity of BOWS with respect to vanilla BO is given by the training of the MLP, mapping from $\mathcal{W}$ back to $\mathcal{X}$. A rough indication of the computational complexity is $O(m_1 \times m_2 \times m_3 \times m_4)$. Where m_1 is the number of epochs; m_2 is the number of training examples; m_3 is the number of objective function's components; m_4 is the number of neurons. The computational overhead due to working in $\mathcal{W}$ and the ensuing need to map back to $\mathcal{X}$ is substantial and explains why the wall-clock time of BOWS is poorer than vanilla BO. It is a reasonable cost to pay for the improvement in sampling efficiency as it will be shown in the computational results in the following section.

4 Computational results

4.1 Experimental setting

The algorithms have been implemented using BoTorch (Balandat et al. 2020) a Python library for Bayesian Optimization part of the PyTorch framework. BoTorch provides an easy-to-use interface for defining, managing, and running sequential experiments and a modular interface for composing Bayesian optimization primitives as probabilistic models, acquisition functions, and optimizers. The computational results reported in this section have been obtained using UCB (with $\beta = 4$) and a gradient based optimizer. Three test functions have been considered (Table 1) with dimensionality$d = 5, 10, 15, 20$. For each experiment 10 independent runs have been executed with $20d$ iterations and d initial points.

4.2 Experimental results

In this section, the computational results on the three test functions reported in Table 1 are presented, considering dimensionality $d = 5, 10, 15, 20$.

As shown in Table 2, considering Alpine01 and Vincent, BOWS has a better overall performance respect to vanilla BO; the advantage increases in higher dimensionality. In Fig. 1 is highlighted that BOWS converges faster to an optimal solution, in terms of iterations, particularly considering higher d. In the case of Michalewicz, BOWS generally performs worse than standard BO. The gap in performances decreases increasing the dimensionality.

To explain the different behaviour of BOWS with the Michalewicz test function we have to look at the MLP error. The error is defined in the Wasserstein space as

Table 2 Best seen averaged over 10 trials, with its standard deviation

		Best seen	
Function	Dimension	BO	BOWS
Alpine01	5	**0.83 ± 1.08**	2.85 ± 0.68
	10	9.02 ± 3.83	**1.88 ± 1.03**
	15	21.44 ± 2.33	**1.62 ± 0.46**
	20	30.26 ± 3.20	**1.8 ± 0.76**
Michalewicz	5	– **2.33 ± 0.34**	– 1.21 ± 0.43
	10	– **4.06 ± 0.41**	– 3.23 ± 1.01
	15	– **4.98 ± 0.43**	– 4.16 ± 0.59
	20	– **6.08 ± 0.62**	– 5.4 ± 0.62
Vincent	5	– 3.54 ± 0.63	– **3.89 ± 0.75**
	10	– 5.24 ± 0.61	– **8.11 ± 1.27**
	15	– 7.04 ± 0.85	– **11.54 ± 2.20**
	20	– 8.59 ± 1.27	– **15.52 ± 2.08**

The best result for each experiment is highlighted in bold

Table 1 Test functions

Function	Equation	Search space	Global minimum
Alpine01	$f(x) = \sum_{i=1}^{d} \left\lvert x_i \sin\left(x_i\right) + 0.1x_i \right\rvert$	$[-10, 10]^d$	$f(0) = 0$
Michalewicz	$f(x) = -\sum_{i=1}^{d} \sin\left(x_i\right)\sin^{2\mathrm{k}}\left(\frac{ix_i^2}{\pi}\right)$	$[0, \pi]^d$	$f(0) = 1.8013$
Vincent	$f(x) = \sum_{i=1}^{d} \sin\left(10\log\left(x_i\right)\right)$	$[0.25, 10]^d$	$f(7.706281) = -d$

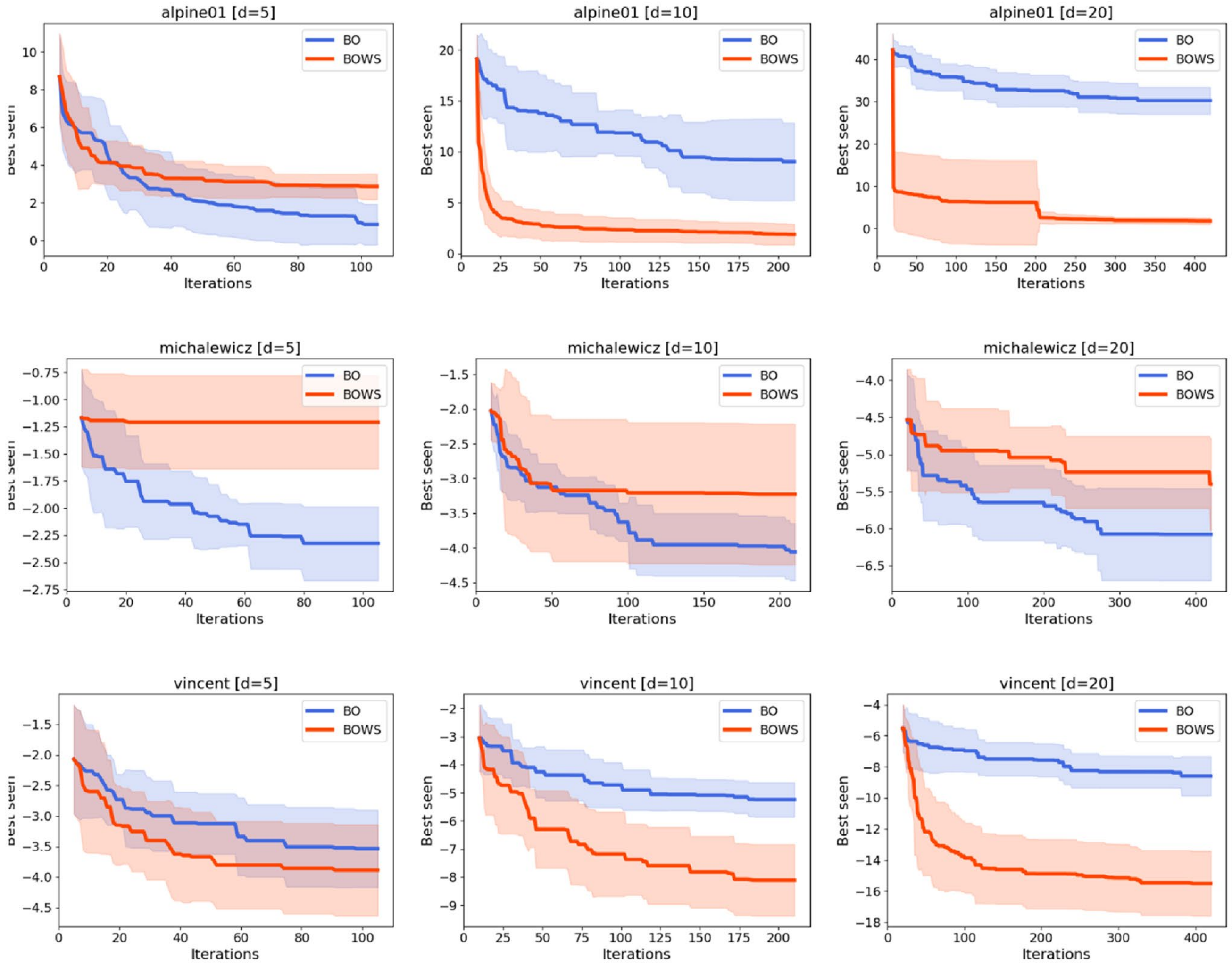

Fig. 1 Best seen over iterations for the two algorithms and the three test functions. The line represents the mean over 10 independents runs while the shaded area is the standard deviation

$\frac{1}{N}\sum_{i=1}^{N} W\left(\hat{\mathbf{H}}^{(i)}, \mathbf{H}^{(i)}\right)$. In the case of Alpine01 and Vincent the error appears to slightly decrease with the increasing of the iterations (Fig. 2). This is coherent to the fact that increasing the iterations means a higher number of training points for the neural network. In the case of Michalewicz, the error shows a completely opposite behaviour, meaning that the MLP cannot properly map the function's components from $\mathcal{W}$ back to the search space $\mathcal{X}$. This behaviour is particularly marked for higher dimensionality of the search space.

The main difference between Michalewicz and the other two test function is that the Michalewicz's components depend on the number of dimensions d, and in particular they get more complex as d increases (Fig. 3). Specifically, the number of local minima is $d!$. In the case of Alpine01 and Vincent the complexity of the components does not depend on the dimensionality d.

The difference in complexity of the functions' components can also be seen by looking at the correlation between $\mathbf{H}^{(i)}$ and $\mathbf{x}^{(i)}$ for $i = 1, \ldots, N$. As shown in Table 3, in the case of Michalewicz, the Pearson correlation is much lower than the other two test functions, meaning a higher complexity in finding a mapping function.

Since mapping the Michalewicz's components back to the search space is more complex a possible solution is to increase the number of hidden layers of the MLP. Figure 4 and Fig. 5 show that with 5 hidden layers the performance of BOWS for Michalewicz improves and the MLP error decreases.

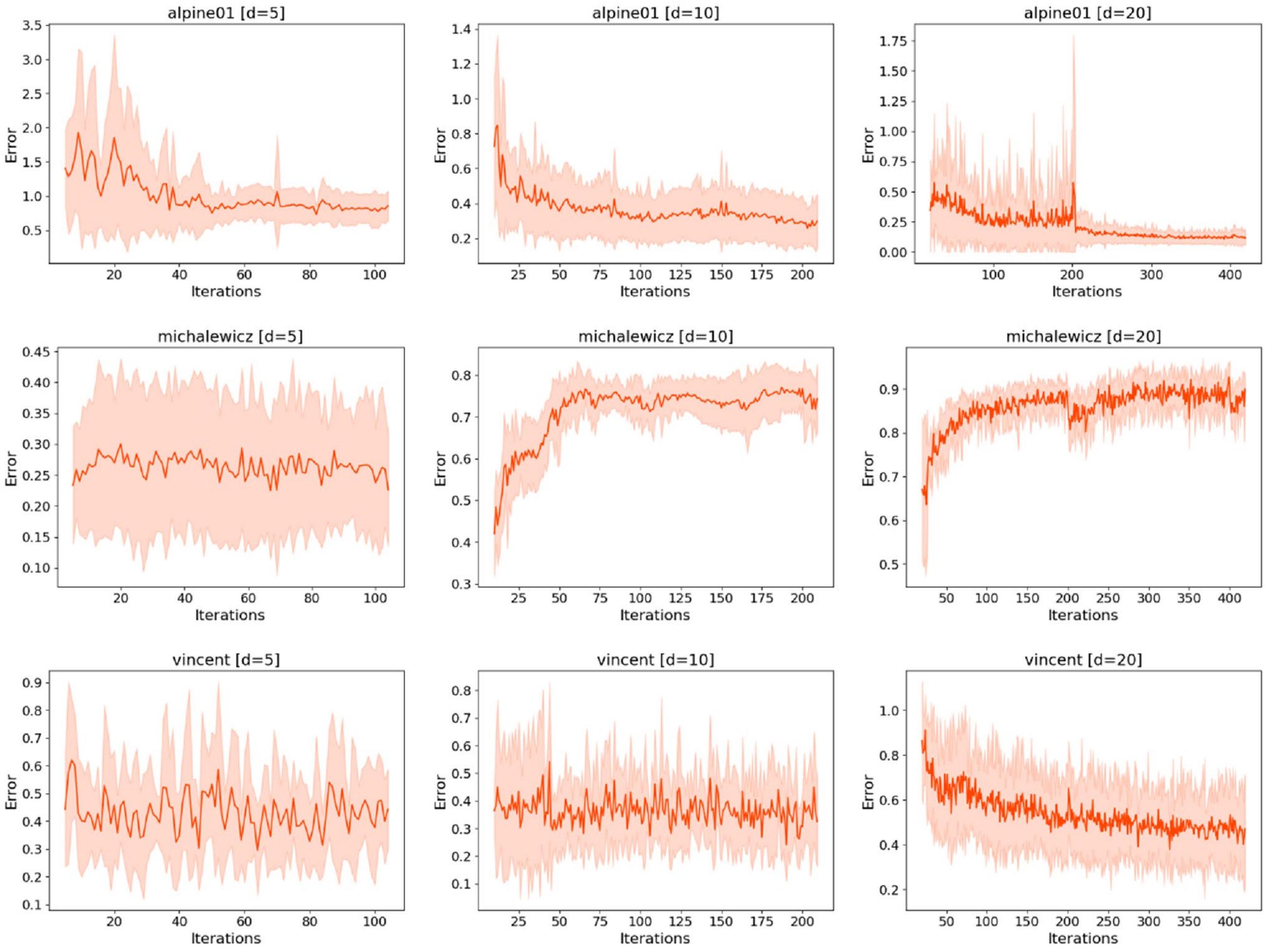

Fig. 2 MLP's error over iterations for the three test functions. The line represents the mean over 10 independents runs while the shaded area is the standard deviation. The error is computed in the Wasserstein space instead of the Euclidean space

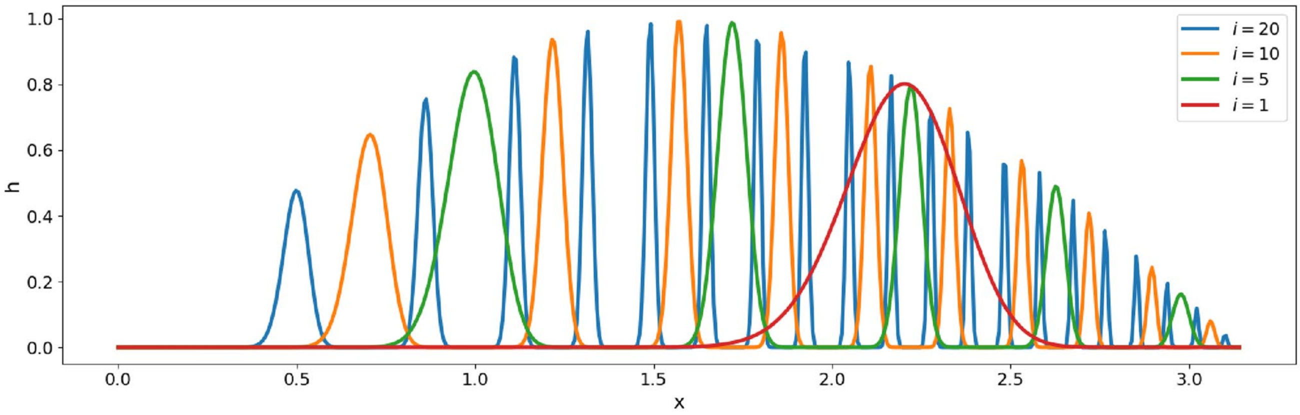

Fig. 3 Michalewicz's components $\sin(x)\sin^{2\mathrm{k}}\left(\frac{ix^2}{\pi}\right)$ with $i = 1, 5, 10, 20$

Table 3 Pearson correlation between $\mathbf{H}^{(i)}$ and $\mathbf{x}^{(i)}$ for $i = 1, \ldots, N$

d	Alpine01	Michalewicz	Vincent
5	0.4754	0.3243	0.4510
10	0.4686	0.2865	0.3808
15	0.3508	0.2102	0.3729
20	0.3924	0.1560	0.3756

5 Conclusions and future works

The main conclusion is that a distributional representation of points in the search space as point clouds can be effectively applied to Bayesian optimization. The Wasserstein distance has been chosen because it's a metric, captures complex relationships between inputs, neighbourhood sizes and connectivity and provides geometrically meaningful distances. Computational experiments show, both in terms of function evaluations and wall clock time, how the new method in two out of three benchmark functions outperforms vanilla Bayesian optimization and its advantage increases with the dimension of the search space.

Future works should address the following main issues:

- Methodological advances to improve the optimization of the acquisition function considering also, from a theoretical standpoint, both the differentiability of the WST distance and the relation between the gradient flows of the objective function and the transport map.
- A full analysis of the optimization problems which fit into the BOWS framework. The distributional approach is natural for simulation–optimization problems over discrete structures, sensor placement in physical and informational networks and stochastic vehicle routing. Also, the issue of high dimensionality and the underlying additive structure should be further analyzed.

Additional experiments are required for a more extensive numerical validation of the proposed approach.

Funding Open access funding provided by Università degli Studi di Milano - Bicocca within the CRUI-CARE Agreement.

Data availability All code and data used in this paper are available at https://github.com/andreaponti5/bows.

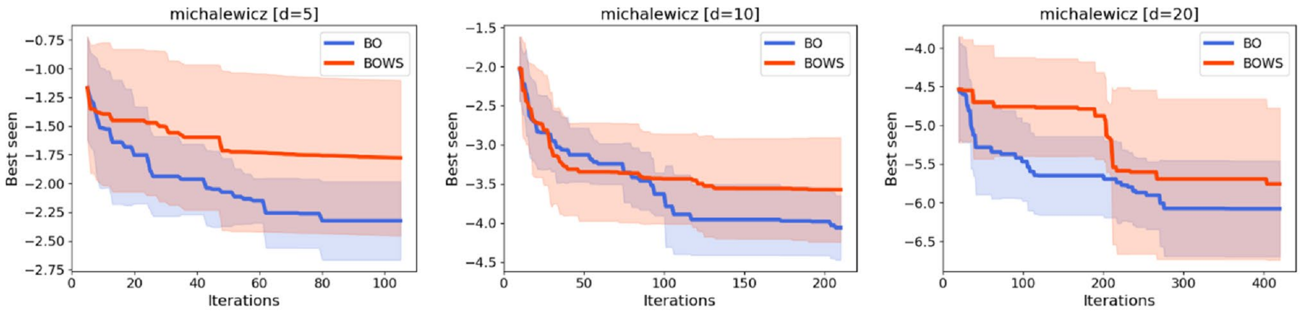

Fig. 4 Best seen over iterations of Michalewicz considering the 5 layers MLP in BOWS

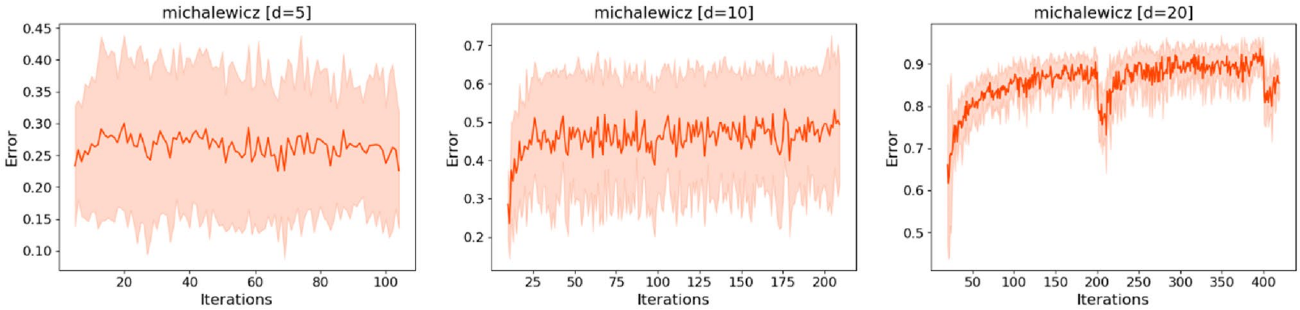

Fig. 5 MLP's error over iterations of Michalewicz considering the 5 layers MLP in BOWS

References

Alvarez-Melis D, Fusi N (2020) Geometric dataset distances via optimal transport. Adv Neural Inf Process Syst 33:21428–21439

Alvarez-Melis D, Fusi N (2021) Dataset dynamics via gradient flows in probability space. In: International Conference on machine learning. PMLR, pp 219–230

Archetti F, Candelieri A (2019) Bayesian optimization and data science. Springer International Publishing

Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks. In: International conference on machine learning. PMLR, pp 214-223

Balandat M, Karrer B, Jiang D, Daulton S, Letham B, Wilson AG, Bakshy E (2020) BoTorch: a framework for efficient Monte-Carlo Bayesian optimization. Adv Neural Inf Process Syst 33:21524–21538

Candelieri A, Ponti A, Archetti F (2021) Uncertainty quantification and exploration–exploitation trade-off in humans. J Ambient Intell Humaniz Comput, 1–34.

Candelieri A, Ponti A, Archetti F (2022a) Bayesian optimization in Wasserstein spaces. In: International Conference on Learning and Intelligent Optimization. Springer, Cham

Candelieri A, Ponti A, Archetti F (2022b) Gaussian Process regression over discrete probability measures: on the non-stationarity relation between Euclidean and Wasserstein Squared Exponential Kernels. arXiv preprint arXiv:2212.01310

Crovini E, Cotter SL, Zygalakis K, Duncan AB (2022) Batch Bayesian optimization via particle gradient Flows. arXiv preprint arXiv:2209.04722

Frazier PI (2018) Bayesian optimization. In: Recent advances in optimization and modeling of contemporary problems. INFORMS, pp 255–278

Frogner C, Mirzazadeh F, Solomon J (2019) Learning embeddings into entropic Wasserstein spaces. arXiv preprint arXiv:1905.03329

Gangbo W, McCann RJ (2000) Shape recognition via Wasserstein distance. Q Appl Math 58:705–737

Gong C, Peng J, Liu Q (2019) Quantile stein variational gradient descent for batch Bayesian optimization. In: International Conference on machine learning, pp 2347–2356. PMLR.

Hein M, Bousquet O (2005) Hilbertian metrics and positive definite kernels on probability measures. In: International Workshop on Artificial Intelligence and Statistics, pp 136–143. PMLR

Jaquier N, Rozo L (2020) High-dimensional Bayesian optimization via nested Riemannian manifolds. Adv Neural Inf Process Syst 33:20939–20951

Jaquier N, Rozo L, Calinon S, Bürger M (2020) Bayesian optimization meets Riemannian manifolds in robot learning. In: Conference on Robot Learning, pp 233–246. PMLR

Kandasamy K, Schneider J, Póczos B (2015) High dimensional Bayesian optimisation and bandits via additive models. In: International Conference on machine learning, pp 295–304. PMLR

Kandasamy K, Neiswanger W, Schneider J, Poczos B, Xing EP (2018) Neural architecture search with bayesian optimisation and optimal transport. In: Advances in neural information processing systems, p 31

Kantorovich LV (1942) On the translocation of masses. Dokl. Akad. Nauk. USSR (NS) 37:199–201

Lau TTK, Liu H (2022) Wasserstein distributionally robust optimization via Wasserstein barycenters. arXiv:2203.12136

Liu Q, Wang D (2016) Stein variational gradient descent: a general purpose Bayesian inference algorithm. In: Advances in neural information processing systems, p 29

Liu J, Wu J, Li B, Cui P (2022) Distributionally robust optimization with data geometry. In: Advances in neural information processing systems, vol 35, pp 33689–33701

Liutkus A, Simsekli U, Majewski S, Durmus A, Stöter FR (2019) Sliced-Wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions. In: International Conference on machine learning, pp 4104–4113. PMLR

Monge G (1781) Mémoire sur la théorie des déblais et des remblais. De l'Imprimerie Royale

Moriconi R, Kumar KS, Deisenroth MP (2020) High-dimensional Bayesian optimization with projections using quantile Gaussian processes. Optim Lett 14(1):51–64

Nguyen V, Gupta S, Rane S, Li C, & Venkatesh S (2017) Bayesian optimization in weakly specified search space. In: 2017 IEEE International Conference on data mining (ICDM), pp 347–356. IEEE

Öcal K, Grima R, Sanguinetti G (2019) Parameter estimation for biochemical reaction networks using Wasserstein distances. J Phys A Math Theor 53(3):034002

Peyré G, Cuturi M (2019) Computational optimal transport: with applications to data science. Found Trends® Mach Learn 11(5–6):355–607

Redko I, Courty N, Flamary R, Tuia D (2019) Optimal transport for multi-source domain adaptation under target shift. In: The 22nd International Conference on artificial intelligence and statistics, pp 849–858. PMLR

Rout L, Korotin A, Burnaev E (2021) Generative modeling with optimal transport maps. arXiv preprint arXiv:2110.02999

Salim A, Korba A, Luise G (2020) The Wasserstein proximal gradient algorithm. Adv Neural Inf Process Syst 33:12356–12366

Vayer T, Chapel L, Flamary R, Tavenard R, Courty N (2018) Optimal transport for structured data with application on graphs. arXiv preprint arXiv:1805.09114

Villani C (2009) Optimal transport: old and new, vol 338. Springer, Berlin, p 23

Zhang R, Chen C, Li C, Carin L (2018) Policy optimization as wasserstein gradient flows. In International Conference on machine learning, pp 5737–5746. PMLR.

Zhang R, Wen Z, Chen C, Carin L (2019) Scalable Thompson sampling via optimal transport. arXiv preprint arXiv:1902.07239.