Locally-Biased Bayesian Optimization using Nonstationary Gaussian Processes

Ruben Martinez-Cantin
Centro Universitario de la Defensa
Zaragoza, 50090, Spain
rmcantin@unizar.es

Abstract

Bayesian optimization is becoming a fundamental global optimization algorithm in many applications where sample efficiency is needed, ranging from automatic machine learning, robotics, reinforcement learning, experimental design, simulations, etc. The most popular and effective Bayesian optimization method relies on a stationary Gaussian process as surrogate. In this paper, we present a novel nonstationary strategy for Bayesian optimization that is able to outperform the state of the art in Bayesian optimization both in stationary and nonstationary problems, such as automatic hyperparameter tuning and reinforcement learning.

1 Introduction

Bayesian optimization, although being a classic method [17, 18], has become quite popular recently for being a sample efficient method of global optimization [12]. Recent works have found similarities with Bayesian optimization and the way biological systems adapt and search, such as human active search [2] or animal adaptation to injuries [5]. In machine learning, it has been applied for automatic algorithm tuning by [22] and reinforcement learning by [16]. The main contribution of the paper is an algorithm for improved Bayesian optimization using a combination of local and global kernels to achieve a nonstationary behavior, called *Spartan Bayesian Optimization*. Although it reaches its best performance in problems that are clearly nonstationary, our evaluation shows that it can improve the results of Bayesian optimization in most scenarios, similar to other locally-biased global optimization algorithms [7].

Consider the problem of finding the minimum x^* of an unknown real valued function $f : \mathbb{X} \to \mathbb{R}$, where $\mathbb{X}$ is a compact space, $\mathbb{X} \subset \mathbb{R}^d, d \geq 1$. In order to find the minimum, we assume has a maximum budget of N evaluations of the target function f. The purpose of Bayesian optimization is to find optimal decisions for searching the minimum using a probabilistic surrogate model $P(f)$. For the remainder of the paper, we are going to assume that $P(f)$ is a Gaussian process $\xi(\mathbf{x})$ with inputs $\mathbf{x} \in \mathbb{X}$ and a kernel or covariance function $k(\cdot, \cdot)$ with hyperparameters Θ. We assume that hyperparameters are estimated using MCMC. In order to avoid bias and guarantee global optimality, we rely on an initial design of p points based on *Latin Hypercube Sampling* (LHS) following the recommendation of [12, 4, 13]. Finally, we use the *expected improvement* criterion from [17] to select the next point each iteration. However, it is important to note that the ideas presented also work with other popular models such as Student-t processes [18, 29, 21] or other criteria such as *upper confidence bound* by [24] or *relative entropy* [10, 9], among others.

2 Nonstationarity in Gaussian processes

Many applications of Gaussian process regression, including Bayesian optimization, are based on the assumption that the process is stationary and often isotropic. For example, the use of the isotropic squared exponential kernel in GPs is quite frequent: $k_{SE}(\mathbf{x}, \mathbf{x}') = \exp(-1/2r^2)$, being $r^2 =$

$(\mathbf{x}-\mathbf{x}')^T\Lambda(\mathbf{x}-\mathbf{x}'))$, where $\Lambda = \theta_l^{-1}\mathbf{I}$ and θ_l represents the length-scale hyperparameter that captures the smoothness or variability of the function. That is, small values of θ_l will be more suitable to capture signals with high frequency components; while large values of θ_l result in model for low frequency signals or flat functions. This property also holds for other popular kernels like the anisotropic kernels with automatic relevance determination (ARD) [19] where in this case, $\Lambda = diag(\Theta)$ becomes a diagonal matrix with a length-scale parameter per dimension.

This length-scale estimation results in an interesting behavior in Bayesian optimization. For the same distance between points, a kernel with smaller length-scale will result in higher predictive variance, therefore the exploration will be more aggressive. This idea has been explored previously in [27] by forcing smaller scale parameters to improve the exploration. More formally, in order to achieve no-regret convergence to the minimum, the target function must be an element of the reproducing kernel Hilbert space (RKHS) characterized by the kernel $k(\cdot,\cdot)$ [4, 24]. For a set of kernels like the SE or Matérn, it can be shown that, given two kernels k_l and k_s with large and small length scale hyperparameters respectively, *any function f in the RKHS characterized by a kernel k_l is also an element of the RKHS characterized by k_s* [27]. Thus, using k_s instead of k_l is safer in terms of guaranteeing convergence. However, if the small kernel is used everywhere, it might result in unnecessary sampling of smooth areas.

There have been several attempts to model nonstationary functions with Gaussian processes. For example, the use of specific nonstationary kernels [19], Bayesian Treed GP models by [8] or projecting the input space to a stationary latent space by [20]. Recently, a version of the latter idea has been applied to Bayesian optimization by [23], with a further work building Treed GPs on top of the warping model by [1]. Treed GPs were previously used in BO by [26]. A related approach of additive GPs is used in [13] for Bayesian optimization of high dimensional functions under the assumption that the actual function is a combination of lower dimensional functions.

Our approach to nonstationarity, the *Spartan Bayesian Optimization* algorithm, is based on the model presented in [14] where the input space is partitioned in different regions such as the resulting GP is the linear combination of local GPs: $\xi(\mathbf{x}) = \sum_i \lambda_i(\mathbf{x})\xi_i(\mathbf{x})$. Each local GP has its own specific hyperparameters, making the final GP nonstationary even when the local GPs are stationary. In order to achieve smooth interpolation between regions, [14] suggest the use of a weighting function $\nu_i(\mathbf{x})$ for each region, having the maximum in region i and decreasing its value with distance to region i. Then, we can set $\lambda_i(\mathbf{x}) = \sqrt{\frac{\nu_i(\mathbf{x})}{\sum_j \nu_j(\mathbf{x})}}$.

For Bayesian optimization, we suggest the combination of a local and a global kernel with multivariate normal distributions as weighting functions. Assuming a normalized input space $[0,1]^d$, we consider that for each dimension:

$$\nu_{global}^{(k)} = \mathcal{N}(0.5, \sigma_{global}); \qquad \nu_{local}^{(k)} = \mathcal{N}(\theta_{pos}^{(k)}, \sigma_{local}) \qquad \forall \;\; k = 1 \ldots d \tag{1}$$

where $\{\theta_{pos}^{(k)}\}_1^d$ is considered to be part of the set of hyperparameters of the surrogate model that are learned accordingly when new data is available $\Theta = \{\theta_{pos}, \theta_{local}, \theta_{global}\}$. In that way, the position of the local kernel is adapting towards to the area near the minimum or other important area. Besides, we set in advance the value of the variances $(\sigma_{global}, \sigma_{local})$ as a regularization term to avoid overfitting. In our tests and experiments, we found that a robust value for the variances was $\sigma_{global} = 10$ and $\sigma_{local} = 0.05$. Thus, the resulting kernel function is defined as:

$$k(\mathbf{x},\mathbf{x}'|\theta_i) \leftarrow \lambda_l(\mathbf{x}|\theta_i^{pos})\lambda_l(\mathbf{x}'|\theta_i^{pos})k_l(\mathbf{x},\mathbf{x}'|\theta_i^l) + \lambda_g(\mathbf{x})\lambda_g(\mathbf{x}')k_g(\mathbf{x},\mathbf{x}'|\theta_i^g) \tag{2}$$

The intuition behind this setup is the same of many acquisition functions in Bayesian optimization: *the aim of the surrogate model is not to approximate the target function precisely in every point, but to provide information about the location of the minimum*. For example, the resulting model can *flatten* most of the search space, as soon as the region near the minimum has the correct variability. Many optimization problems are difficult due to the fact that the region near the minimum has higher variability than the rest of the space. However, it is important to note that the kernel hyperparameters are initialized with the same prior for the local and global kernel. Thus, there is no guarantee that the local kernel becomes the kernel with smaller length-scale. Depending on the data captured, it could learn a model where the local kernel has larger length-scale (i.e.: smoother) than the global kernel, which may also improves the convergence in plateau-like functions.

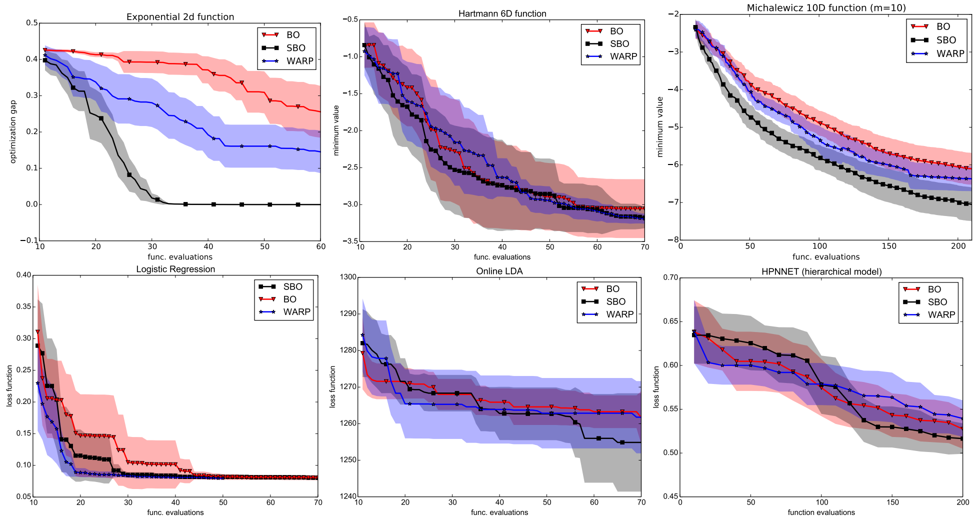

Figure 1: Results on classic optimization and machine learning benchmarks [6].

3 Evaluation and results

We have implemented[1] Spartan Bayesian Optimization (SBO) in the BayesOpt library [15]. For comparison, we also implemented the input warping method (WARP) [23]. We also evaluated other nonstatioary Bayesian optimization models like random forests [11], but did not include in the results as its performance was in general worse than *vanilla* BO.

The results presented in this section are based on the standard convention in Bayesian optimization literature, that is, a simple zero-mean Gaussian process, a Matérn kernel $\nu = 5/2$ with automatic relevance determination for continuous variables $k_{M5/2}(\mathbf{x}, \mathbf{x}') = \exp(-\sqrt{5}r)(1 + \sqrt{5}r + \frac{5}{3}r^2)$, a Hamming kernel as presented in [27] for categorical variables and slice sampling for learning the model hyperparameters (length-scale, warping, position, etc.). However, using BayesOpt, the suggested method has also been tested with other models such as Student-t processes, other kernels, etc. Due to the computational burden of MCMC for the hyperparameters, we have used a small number of samples (10), while trying to decorrelate every resample with large burn-in periods (100 samples) following the convention in [22]. All experiments were repeated 20 times using common random number to reduce the sampling error between algorithms. The number of function evaluations in each plot includes a initial design of 10 points from LHS.

Top row of Figure 1 shows the results of optimizing the exponential 2D function $f(\mathbf{x}) = x_1 \exp(-x_1^2 - x_2^2)$ for $x_1, x_2 \in [-2, 18]^2$ from [8]. The use of classical stationary models (BO) results in a poor convergence because of the high nonstationarity of the function, while nonstationary methods, such as [23] (WARP) and the proposed method (SBO) result in an improved convergence. For the Hardmann 6D function, the differences are barely significant, which might imply that the function is stationary. However, even in that case, we can see that nonstationary methods can easily avoid getting stuck in local minima, being more robust. The Michalewicz function is known to be one of the hardest benchmarks in global optimization due to its many local minima. In this case, all the methods have slower convergence, due to the complexity of the problem

Bottom row of Figure 1 is based on well known benchmarks for automatic tunning of machine learning algorithms [6]. Among all the available benchmarks we have selected the Gradient Boosting as it provides the lowest RMSE with respect to the actual problem. The logistic regression problem (4D continuous) is easy for Bayesian optimization. Even the *vanilla* BO can reach the minimum in less than 50 function evaluations. In this case, the WARP method is the fastest one, with almost 20 evaluations. However, the proposed method has only slightly worse performance by a small fraction of the total cost. For the onlineLDA problem (3D continuous), both the standard BO and the WARP method get stuck while our method is able to achieve a 50% extra gain. Finally, for the HP-NNET problem using the MRBI dataset (7D continuous, 7D categorical), SBO fails to converge at an early

[1] The code will be freely available once the paper gets published.

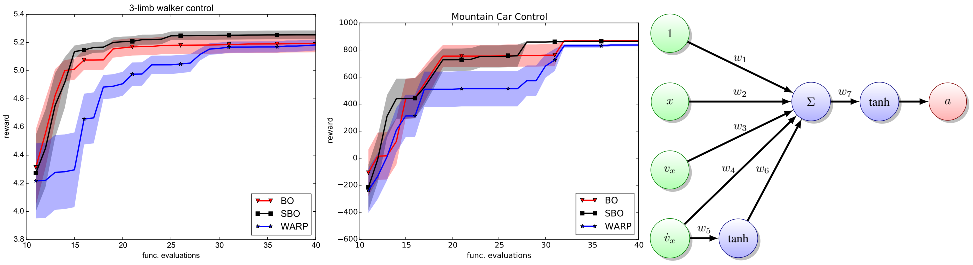

Figure 2: RL problems. Left: Walker. Center: Mountain Car Right: Mountain Car policy.

stage, due to the complexity of the problem. However, as more data is gathered, the local kernel jumps to a good spot and the convergence is faster.

We also evaluate SBO with several classic reinforcement learning problems. We use the *active policy search* model [16], which has the advantage being model free and without access to the dynamics, state or instantaneous reward of the system. However, in this setup there are usually *failure* states or scenarios which result in flat or plateau regions due to large penalties. This is opposed to the behavior of the reward near the optimal policy where small variations on a suboptimal policy can considerably change the performance achieved. Therefore, the resulting reward function presents a nonstationary behavior with respect to the policy parameters.

Figure 2 shows the performance of learning the controller of a three limb walker to allow fast upright walking on the model presented in [28]. It has been already used for Bayesian optimization benchmark in [10]. The reward was based on a walking speed with a penalty for not maintaining the upright position. The dynamic controller has 8 continuous parameters. We also present the classical mountain car problem by [25], but dealing directly with continuous states and actions. The policy is a simple perceptron model inspired by [3]. The potentially unbounded policy parameters $\mathbf{w} = \{w_i\}_{i=1}^{7}$ are computed as $\mathbf{w} = \tan\left((\pi - \epsilon)\,\mathbf{w}_{01} - \frac{\pi}{2}\right)$ where $\mathbf{w}_{01}$ are the policy parameters bounded in the $[0,1]^7$ space and ϵ is a small number to avoid $w_i \to \infty$.

Time (s)	Exp [8]	Hart6	Micha10	LogReg	OnLDA	HPNNET	Walker	MCar
#evals	60	70	210	50	70	200	40	40
BO	120	460	8360	28	112	20	47	38
SBO	2481	10415	225313	730	2131	146	440	797
WARP	13929	188942	4445854	9149	21299	2853	20271	18972

One of the main advantages of SBO is its reduced computational cost with respect to WARP as can be seen above.The main different between the three algorithms is the kernel function $k(\cdot,\cdot)$, which becomes an important factor as the kernel function is called millions or billions of times in Bayesian optimization. However, the same computational tricks and parallelization can be applied to all the methods. In the case of WARP, the kernel includes the Beta CDF which is much more involved than the evaluation of the Matérn kernel or the Gaussian weights of SBO. Besides, we found that in all benchmarks, learning the position of the local kernel is easy for slice sampling. In contrast, the narrow likelihood of Beta parameters implies that many samples are rejected during MCMC[2]. It is important to note that, although Bayesian optimization is intended for expensive functions and the cost per iteration is negligible, the difference between methods could mean hours of CPU-time for a single iteration, changing the range of potential applications.

4 Conclusions

We have presented a new algorithm called Spartan Bayesian Optimization (SBO) which combines a local and a global kernel to deal with nonstationarity in Bayesian optimization. Besides, we have shown that the model can increase convergence speed even in stationary problems by improving local refinement while retaining global exploration capabilities. We have validated the performance of the algorithm in standard optimization benchmarks, machine learning applications such as hyperparameter tuning problems and classic reinforcement learning scenarios.

[2] For all algorithms we use slice sampling as recommended by [23]

Acknowledgments

This work has been supported by project SIRENA (CUD05-2013)

References

[1] John-Alexander M. Assael, Ziyu Wang, and Nando de Freitas. Heteroscedastic treed bayesian optimisation. Technical report, arXiv, 2014.

[2] Ali Borji and Laurent Itti. Bayesian optimization explains human active search. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 55–63. Curran Associates, Inc., 2013.

[3] E. Brochu, V.M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. eprint arXiv:1012.2599, arXiv.org, December 2010.

[4] Adam D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

[5] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521:503507, 2015.

[6] K. Eggensperger, F. Hutter, H.H. Hoos, and K. Leyton-Brown. Efficient benchmarking of hyperparameter optimizers via surrogates. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 2015.

[7] Joerg M Gablonsky and C Tim Kelley. A locally-biased form of the DIRECT algorithm. *Journal of Global Optimization*, 21(1):27–37, 2001.

[8] Robert B Gramacy. *Bayesian treed Gaussian process models*. PhD thesis, University of California, Santa Clara, 2005.

[9] Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13:1809–1837, 2012.

[10] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 918–926. Curran Associates, Inc., 2014.

[11] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *LION-5*, page 507523, 2011.

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

[13] Kirthevasan Kandasamy, Jeff Schneider, and Barnabas Poczos. High dimensional bayesian optimisation and bandits via additive models. In *International Conference on Machine Learning (ICML)*, 2015.

[14] Andreas Krause and Carlos Guestrin. Nonmyopic active learning of Gaussian processes: an exploration-exploitation approach. In *International Conference on Machine Learning (ICML)*, Corvallis, Oregon, June 2007.

[15] Ruben Martinez-Cantin. BayesOpt: A Bayesian optimization library for nonlinear optimization, experimental design and bandits. *Journal of Machine Learning Research*, 15:3735–3739, 2014.

[16] Ruben Martinez-Cantin, Nando de Freitas, Eric Brochu, Jose Castellanos, and Arnoud Doucet. A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27(3):93–103, 2009.

[17] Jonas Mockus. *Bayesian Approach to Global Optimization*, volume 37 of *Mathematics and Its Applications*. Kluwer Academic Publishers, 1989.

[18] Anthony O'Hagan. Some Bayesian numerical analysis. *Bayesian Statistics*, 4:345–363, 1992.

[19] Carl E. Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, Massachusetts, 2006.

[20] Paul D Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417):108–119, 1992.

[21] Amar Shah, Andrew Gordon Wilson, and Zoubin Ghahramani. Student-t processes as alternatives to Gaussian processes. In *AISTATS, JMLR Proceedings. JMLR.org*, 2014.

[22] Jasper Snoek, Hugo Larochelle, and Ryan Adams. Practical Bayesian optimization of machine learning algorithms. In *NIPS*, pages 2960–2968, 2012.

[23] Jasper Snoek, Kevin Swersky, Richard S. Zemel, and Ryan Prescott Adams. Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning*, 2014.

[24] Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. International Conference on Machine Learning (ICML)*, 2010.

[25] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[26] Matthew A Taddy, Herbert KH Lee, Genetha A Gray, and Joshua D Griffin. Bayesian guided pattern search for robust local optimization. *Technometrics*, 51(4):389–401, 2009.

[27] Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando de Freitas. Bayesian optimization in high dimensions via random embeddings. In *International Joint Conferences on Artificial Intelligence (IJCAI) - Extended version: http://arxiv.org/abs/1301.1942*, 2013.

[28] Eric R Westervelt, Jessy W Grizzle, Christine Chevallereau, Jun Ho Choi, and Benjamin Morris. *Feedback control of dynamic bipedal robot locomotion*, volume 28. CRC press, 2007.

[29] Brian J. Williams, Thomas J. Santner, and William I. Notz. Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica*, 10(4):1133–1152, 2000.