Dynamic Kernel Selection Criteria for Bayesian Optimization

Ibai Roman
Intelligent Systems Group
University of the Basque Country (UPV/EHU)
ibai.roman@ehu.es

Roberto Santana
Intelligent Systems Group
University of the Basque Country (UPV/EHU)
roberto.santana@ehu.es

Alexander Mendiburu
Intelligent Systems Group
University of the Basque Country(UPV/EHU)
alexander.mendiburu@ehu.es

Jose A. Lozano
Intelligent Systems Group
University of the Basque Country (UPV/EHU)
ja.lozano@ehu.es

Abstract

In Bayesian Optimization, when using a Gaussian Process prior, some kernels adapt better than others to the objective function. This research evaluates the possibility of dynamically changing the kernel function based on the probability of improvement. Five kernel selection strategies are proposed and tested in well known synthetic functions. According to our preliminary experiments, these methods can improve the efficiency of the search when the best kernel for the problem is unknown.

1 Introduction

In order to find the global optimum in expensive black-box functions [6] each evaluation point must be carefully selected. In this scenario two possible strategies are: minimizing the number of evaluations while maximizing the information gain, or maximizing the cumulative reward by finding a balance between exploration and exploitation. However, it has been proved that both strategies are closely related [13]. Bayesian Optimization (BO) [10] has been widely used to face this optimization problem regardless of the strategy, giving promising results [4] [3].

When there is some previous knowledge about the characteristics of the objective function, such as smoothness, Gaussian Process (GP) priors are a popular choice to take advantage of this knowledge [9] [15]. Thus, depending on the kernel, the GP assumes some characteristics to fit the objective function.

Nevertheless, the shape of the objective function might change depending on the area of the search space. Although kernel selection has been extensively studied in GP research field [11], this decision is not a trivial one and it clearly influences the algorithm's performance. Moreover, GP priors are known to be expensive to compute. The covariance matrix increases every step and sampling the GP becomes more and more expensive.

In our context, when the evaluation of the objective function is considered to be expensive, this computational cost is less relevant. It is even possible to carry out an optimization computing several

GPs in parallel without increasing significantly the overall cost. The selection of the kernel can be done during the optimization process to take advantage of the knowledge acquired, improving the search results.

2 Kernel Functions and Probability of Improvement

In this research, stationary kernels will be considered. These kernels only take into account the distance between the points, ignoring their absolute positions ($r = ||x' - x||$).

Table 1 shows six well known stationary kernel functions selected for the experiments, where l is the lengthscale hyper-parameter and σ_n refers to the noise.

Kernel Functions	
$k_{SE}(r) = exp\left(-r^2/2l^2\right) + \sigma_n$	$k_{M32}(r) = \left(1 + \sqrt{3}r/l\right) exp\left(-\sqrt{3}r/l\right) + \sigma_n$
$k_{M52}(r) = \left(1 + \sqrt{5}r/l + 5r^2/3l^2\right) exp\left(-\sqrt{5}r/l\right) + \sigma_n$	$k_E(r) = exp\left(-r/l\right) + \sigma_n$
$k_{E15}(r) = exp\left(-\left(r/l\right)^{1.5}\right) + \sigma_n$	$k_{RQ2}(r) = \left(1 + r^2/4l^2\right)^{-2} + \sigma_n$

Table 1: Kernel Functions. k_{SE}: Squared Exponential. k_{M32}: Matern 32. k_{M52}: Matern 52. k_E: Exponential. k_{E15}: γ-exponential. k_{RQ2}: Rational Quadratic.

To have a good criterion to select the next point, BO relies on the acquisition function ($u(\cdot)$). It suggests a measure of utility for every point in the search space given the data. In addition, this function regulates the exploration versus exploitation trade-off.

Similarly, in this experiment, the acquisition function will be used to guide the search. However, having several GPs implies having several acquisition functions and they must be comparable. Thus, *Probability of Improvement*[7] has been selected as the acquisition function, where the probability of the observation to be better than the best observed point is measured.

$$PI(x) = P(f(x) \geq f(x^+) + \xi) = \Phi\left(\frac{\mu(x) - f(x^+) - \xi}{\sigma(x)}\right)$$

Where μ and σ are the mean and the variance given by the GP, and x^+ is the best point so far. Φ denotes the cumulative distribution function (CDF) of a normally distributed variable. Finally, as recommended in previous studies, $\xi = 0.01$ will be used to guarantee the exploration[8].

3 Dynamic Kernel Selection Criteria

Our proposal resembles a typical BO algorithm [1] with two slight differences. Fist, during the updating process, all GPs get the latest state of the data set ($D_{1:t}$). Second, when selecting the next point several acquisition functions are optimized. Then, to select the next point a chooser function is called.

Algorithm 1: **Dynamic Kernel Selection algorithm**

Sample a random point: $y_1 = f(x_1)$ and Initialize the data $D_1 = \{(x_1, y_1)\}$
Initialize the GPs $\leftarrow D_1$
for $t = 2$ **to** ... **do** $\{$
    Each GP (k) suggests a point to sample by optimizing the acquisition function: $x_{t,k} = argmax_x u_k(x|D_{1:t-1})$
    Choose a point $x_t \leftarrow choose\ (x_{t,k})$
    Sample the objective function: $y_t = f(x_t)$
    Update the data $D_{1:t} = \{D_{1:t-1}, (x_t, y_t)\}$
    Update the GPs $\leftarrow D_{1:t}$
$\}$

Based on Algorithm 1 different $choose$ functions have been defined:

DynamicRandom All GPs will be updated at every function evaluation, and they will be randomly selected during the optimization process.

BestUtility All GPs optimize their acquisition function and suggest a point to evaluate. The point with the highest probability of improvement is selected for the next evaluation.

WeightedBest Each acquisition function is weighted ($u_k^*(x) = u_k(x)w_{k,t}$). Initially all weights are 0.5. After the evaluation, the improvement is measured and the selected GP updates its weight ($w_{k,t+1} = w_{k,t}(\Phi(i_{t-1}) + 0.5)$). Other GPs remain the same ($w_{k,t+1} = w_{k,t}$).

ParallelTest At first step, all GPs optimize their acquisition function and they suggest a point to evaluate. All these points are evaluated and their improvement is measured. The GP with the highest improvement is selected for the next N steps. This process is repeated until any stop criteria is met. In our experiments $N = 20$ was used.

UtilityMean Instead of optimizing each acquisition function on its own, the next point is selected by maximizing the mean of these functions ($u(x) = \sum_{k=1}^{n} \frac{u_k(x)}{n}$). Thus, $choose$ function includes the optimization step.

4 Experimental results

The proposed $choose$ functions were implemented as part of a software framework. To set the hyper-parameters, and for the sake of simplicity, each kernel was tested 25 times with 100 hyper-parameter values and the combination with minimum regret was selected. Other approaches to set the hyper-parameters [12][14] could be considered in the future. On the other hand, in order to optimize the acquisition function, DIRECT was used [5], a global-derivative free algorithm previously applied in [1].

Six well known test functions were selected: Branin-Hoo, Camelback, Hartmann 6d, Schwefels, Rosenbrock and Rastrigin [2]. These functions have been optimized using all previously described selection strategies. Each optimization process was carried out reaching 100 function evaluations and the minimum error at each step is recorded. Due to the stochastic nature of the algorithm, the experiments were repeated 25 times and the arithmetic mean was calculated.

Figure 1 shows the results of the experimentation. Only Hartmann 6d, Rosenbrock and Rastrigin functions are shown, where different behaviors can be observed: for Hartmann 6d and Rosenbrock, our proposals shows a better performance than single kernels, while for Rastrigin, our proposal is not able to outperform the single kernels. The rest of the functions can be found in the website of the project [1].

The upper plots correspond to the average minimum error of the 25 runs. Kernels described in section 2 are compared to the Selection Criteria in section 3. Knowing the kernel selection frequency is interesting to understand the behavior of the $choose$ functions. Hence, the lower graphs show the percentage of times each kernel is selected throughout the optimization process.

Hartmann 6d is a function with 6 dimensions, 6 local optima and a global optimum. Here, $RationalQuadratic2$ and $SquaredExponential$ compete side by side along with $GammaExponential15$ to get the best reward. Regarding to the choosers, $WeightedBest$ outperforms the rest of the choosers, getting the best overall reward. In the first steps, this chooser tries to use a $Matern32$ kernel, but soon changes to $SquaredExponential$ as it is the best performing kernel, and finally, it relies on $Exponential$ kernel for the rest of the experiment. The strategy proposed by $WeightedBest$ combines these kernels, improving their individual results. $UtilityMean$ also gets a nice reward, even improving $WeightedBest$'s results in the last evaluation.

[1] http://ibaidev.bitbucket.org/dynamickernelselection/

	Exponential	GammaExponential15	SquaredExponential	Matern32	Matern52	RationalQuadratic2
Hartmann	138.59	69.54	57.59	145.08	139.59	61.75
Rosenbrock	2030808.97	1852830.13	1759822.40	1877182.86	1900499.22	1837902.24
Rastrigin	6011.16	4919.25	5757.98	6827.07	6810.02	5129.48

	DynamicRandom	BestUtility	WeightedBest	ParallelTest	UtiltyMean
Hartmann	63.73	88.76	49.57	71.15	53.36
Rosenbrock	1850373.80	2030808.97	2016131.36	1986641.55	1338827.99
Rastrigin	5619.27	5589.32	5567.54	5840.98	5708.85

Table 2: Regret.

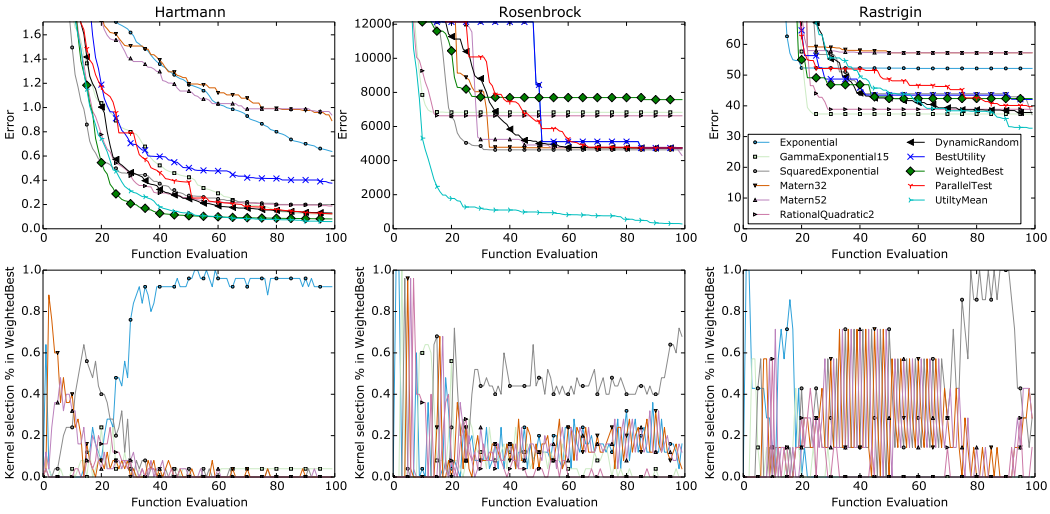

Figure 1: Minimum error per function evaluations

In the other experiment, Rosenbrock has been selected to contrast the results in Hartmann 6d. This function is used with 4 dimensions and a search interval between -10 and 10. $Matern32$, $Matern32$ and $SquaredExponential$ have the lowest regret but they are not able to improve in the second half of the optimization. $Exponential$ kernel also improves in the middle part of the process. This time, $UtilityMean$ clearly gets the best reward, being able to combine the goodnesses of each kernel. However, $WeightedBest$ is not able to improve the local minimum, even correctly selecting $SquaredExponential$ to do it.

Similarly, Rastrigin is used with 4 dimensions and a search interval between -10 and 10. In this case, most of the kernels get locked in a local minimum. $GammaExponential15$ and $RationalQuadratic2$ have the lowest regret but they are not able to improve in the second half of the optimization. On the other hand, none of the choosers is able to beat the best kernel. As shown in the Figure, $WeightedBest$ is not able select a good kernel, cyclically choosing the worst ones. Although, $UtilityMean$ improves in the last evaluations, its overall reward is lower than mentioned kernels.

5 Discussion

This research focuses on finding new ways to combine the knowledge given by each kernel. A test framework has been developed to experiment with model changes during the optimization process. 6 kernels and 5 selection criteria were tested in 6 synthetic functions. According to the results, it has been shown that combining kernels can be a useful technique to improve the performance of GPs. However, the results vary depending on the objective function. When optimizing each acquisition function, it was difficult select the best depending on the acquisition function. On the contrary, the mean of the utility functions has shown to be a promising strategy. Further research is suggested, including new functions and strategies.

References

[1] E. Brochu, V. M. Cora, and N. d. Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-23, Department of Computer Science, University of British Columbia, Nov. 2009.

[2] K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, and K. Leyton-Brown. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *NIPS Workshop on Bayesian Optimization in Theory and Practice*, 2013.

[3] R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for sensor set selection. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2010.

[4] A. Jalali, J. Azimi, X. Fern, and R. Zhang. A lipschitz exploration-exploitation scheme for bayesian optimization. In *Machine Learning and Knowledge Discovery in Databases*, pages 210–224. Springer, 2013.

[5] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

[6] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, Dec. 1998.

[7] H. J. Kushner and G. G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer New York, New York, NY, 1997.

[8] D. J. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Edmonton, Alta., Canada, 2008. AAINR46365.

[9] N. Mahendran, Z. Wang, F. Hamze, and N. D. Freitas. Adaptive MCMC with bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 751–760, 2012.

[10] J. Mockus. *Bayesian approach to global optimization*. Springer, 1989.

[11] C. E. Rasmussen and C. K. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[12] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2951–2959. Curran Associates, Inc., 2012.

[13] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 1015–1022, 2010.

[14] Z. Wang and N. de Freitas. Theoretical analysis of bayesian optimisation with unknown gaussian process hyper-parameters. *arXiv:1406.7758 [cs, stat]*, June 2014. arXiv: 1406.7758.

[15] A. Wilson, A. Fern, and P. Tadepalli. Using trajectory data to improve bayesian optimization for reinforcement learning. *J. Mach. Learn. Res.*, 15(1):253–282, Jan. 2014.