SolarPACES 2022, 28th International Conference on Concentrating Solar Power and Chemical Energy Systems
Analysis and Simulation of CSP and Hybridized Systems
https://doi.org/10.52825/solarpaces.v1i.636

Published: 13 Dec. 2023

Sample-Efficient Hyperparameter Optimization of an Aim Point Controller for Solar Tower Power Plants by Bayesian Optimization

David Zanger[1,2][https://orcid.org/0000-0002-6111-7531], Barbara Lenz[1][https://orcid.org/0009-0007-3764-3068], Daniel Maldonado Quinto[1,2][https://orcid.org/0000-0003-2929-8667], and Robert Pitz-Paal[1,2][https://orcid.org/ 0000-0002-3542-3391]

[1]Institute of Solar Research, German Aerospace Center (DLR), Germany

[2]Chair of Solar Technology, RWTH Aachen University, Germany

Abstract. This work introduces a sample-efficient algorithm to optimize the control parameters of an aim point controller for solar power tower plants. Optimizing the control parameters increases the performance of the aim point controller, and thus the efficiency of the plant. However, optimizing the parameters in simulation will not yield the true optimal parameters at the real plant due to mismatches between simulation and reality. Thus, optimization must be done at the real tower to find a true optimum. As this can be time consuming and costly, the optimizer should require a minimum number of steps. Hence, a sample-efficient optimization strategy is needed. This work introduces a new algorithm based on Bayesian Optimization (BO), which leverages multiple sets of simulation data to accelerate the optimization. The algorithm is tested on a six-dimensional test function representing an arbitrary aim point controller. The proposed algorithm outperformed standard Bayesian Optimization by reaching near optimal parameter configurations of 95% accuracy within 33% less optimization steps. In a second test, the proposed algorithm is used to optimize a simulated Vant-Hull aim point controller with two hyperparameters. Here, the algorithm also needs 33% less optimization iterations than the standard BO.

Keywords: Aim Point Control, Solar Tower, Bayesian Optimization

1. Introduction

To increase the efficiency of solar tower power plants, aim point control is used to maximize the power on the receiver and simultaneously meet allowable flux conditions to prevent damages through overheating. While an optimal aim point control algorithm is still a research question, it is well known that controllers come with a hyperparameter optimization problem when there are complex dependencies between the controller's tuning parameters and its performance. Usually, simulations can be used to find an optimal parameter configuration. However, deviations from expected performance occur when there is a mismatch between simulation and reality due to modeling errors. To ensure optimal control behavior, hyperparameters have to be readjusted on the real plant which takes numerous trials and thus is a costly procedure. A more sophisticated approach is using a sample efficient optimization algorithm. Bayesian Optimization (BO) outperforms other common optimization algorithms in terms of sample efficiency [1], which can be further increased using simulation data [2]. However, former approaches employ only one set of simulation data to the optimization. Since some simulation variables of solar tower plants, like mirror errors, rely on possibly inaccurate estimations, it would be advantageous to generate multiple simulation data sets for different values of the

simulation variable in question. This paper introduces a hyperparameter optimization approach to an arbitrary aim point controller based on BO that uses multiple sets of simulation data to enhance sample efficiency while still reaching near optimal parameter configurations in case of simulation to reality mismatches.

2. Sample-Efficient Hyperparameter Optimization

2.1 Bayesian Optimization

Bayesian Optimization is an iterative algorithm that optimizes an unknown objective function f with respect to its input $\boldsymbol{x}$. The objective function f can be defined as any function indicating the performance of an aim point controller, such as a weighted sum of separate performance metrics like power and violation of allowable flux conditions. $\boldsymbol{x}$ denotes the decision variable which is the vector of relevant controller parameters. In BO, Gaussian Process Regression is used to fit a surrogate model of the objective function f, based on previous function evaluations. A Gaussian Process (GP) is a distribution over functions, specified by its mean function m and covariance function k.

$$f(\boldsymbol{x}) \sim GP\left(m(\boldsymbol{x}),\, k(\boldsymbol{x}, \boldsymbol{x}')\right) \tag{1}$$

The mean function is usually chosen to be 0 [2]. However, within this work the mean function is set to a constant c, which is fitted on previously gathered observation data, because it may enhance the GP Regression [3]. A common kernel function is the squared exponential (SE) kernel, which is defined as

$$k_{SE}(\boldsymbol{x},\boldsymbol{x}')=\sigma_k^2 exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}')^2}{2l^2}\right), \tag{2}$$

where σ_k^2 denotes the output variance and l the length-scale parameter, which determines the smoothness of the function. The function in Eq. (2) is called the prior distribution. Given some observation data with inputs X and outputs $\boldsymbol{y}$ a posterior distribution can be derived

$$f(\boldsymbol{x}_*)|\boldsymbol{y},\, X \sim N\left(\mu_n(\boldsymbol{x}_*),\sigma_n^2(\boldsymbol{x}_*)\right), \tag{3}$$

where $\boldsymbol{x}_*$ is an unevaluated parameter vector. This posterior distribution acts like a regression.

In every optimization step, an acquisition function is used to select the next parameter configuration $\boldsymbol{x_i}$ to evaluate, based on the posterior distribution. A common acquisition function is the Expected Improvement (EI) acquisition function a_{EI}, which estimates the expected magnitude of improvement. The next parameter configuration to evaluate is then chosen by finding the maximum of the acquisition function

$$\boldsymbol{x}_{n+1}=\underset{\boldsymbol{x}\in A}{argmax}\; a_{EI}(\boldsymbol{x}) \tag{4}$$

where A denotes the set of possible parameter configurations. The maximum is usually located where either the expected value and/or uncertainty of the posterior distribution is high. Hence, the acquisition function tries to find a trade-off between exploration and exploitation. The parameter configuration is then applied to the actual system and the observed target value $\mathrm{y}(\mathbf{x_i})$ is used to update the posterior distribution. During the update of the posterior distribution, the hyperparameters $\boldsymbol{\theta}$ of the Gaussian Process e.g. c, l, σ_k are estimated by maximizing the marginal likelihood $p_{ML}(\boldsymbol{y}|X,\boldsymbol{\theta})$. Then, the next iteration starts by finding the maximum of a_{EI} again.

2.2 Enhancing Sample Efficiency in Bayesian Optimization

Bayesian Optimization is already considered as a sample efficient algorithm. In this context, sample efficiency denotes the amount of information an algorithm can use from previous observed samples. Ideally, an increase in sample efficiency decreases the number of iterations

to find the optimum of the objective function. The sample efficiency can further be enhanced by incorporating simulation data into the BO framework. In literature, prior information is either incorporated into the mean or the kernel function. Regarding the mean function there exist different approaches. For example, Cully et al. [4] include a metric based on preselected simulation points into the mean function. Other approaches are described in [5] and [1]. The other possibility is to define a custom kernel function. Here, also different approaches in literature exist. For example, Marco et al. [6] use a kernel composed of the addition of two kernels. There, one kernel is optimized for simulation data and the other models the difference between the simulation and the real system. Wilson et al. [5] define a new kernel function by using the so-called Kullback-Leibler divergence. Another approach is introduced by Antonova et al. [7], who deploy neural networks (NN) into the kernel. Furthermore, Rai et al. [8] introduce an extension to a kernel function by deploying an additional GP which models the mismatch between the simulation and the real system. Preliminary considerations and tests have shown that the approach using NNs by Antonova et. al in combination with the mismatch correction by Rai et al. is the most suitable approach in the context of using simulation data for optimizing an aim point controller. Thus, these approaches are further explained.

2.3 Deploying Neural Networks into the Kernel Function

The kernel function with NN is based on the SE kernel. But, instead of using the distance in parameter space $\left|\left|\mathbf{x}\text{-}\mathbf{x}'\right|\right|_2$ they use the distance in the space of objective function values $\left|\left|y(\mathbf{x})\text{-}y(\mathbf{x}')\right|\right|_2$. However, as the true objective function is not known, they approximate the objective function by an NN, i.e.

$$\hat{y}(\boldsymbol{x})=f_{NN}(\boldsymbol{x}) \tag{5}$$

The NN is trained on one simulation data set. By determining the correlation between two parameter configurations from the objective values, the true objective function may be better approximated. Furthermore, it biases the BO towards promising regions within the simulation data. Using the NN function yields the following kernel function

$$k_{NN}(\boldsymbol{x},\boldsymbol{x}')=\sigma_k^2 exp\left(-\frac{\left(f_{NN}(\boldsymbol{x})\text{-}f_{NN}(\boldsymbol{x}')\right)^2}{2l^2}\right) \tag{6}$$

2.4 Mismatch Correction

The mismatch correction by Rai et al. introduces an additional GP, which models the deviation between simulation $\hat{y}$ and reality y. This mismatch is defined as

$$d=\hat{y}\text{-}y=f_{NN}\text{-}y \tag{7}$$

Here, the simulation data is again interpolated by an NN. The GP to model the deviation is chosen to have a zero mean function and an SE kernel, and thus results to

$$d(\boldsymbol{x})\sim GP\left(0,k_{SE}(\boldsymbol{x},\boldsymbol{x}')\right) \tag{8}$$

Based on the previous observed deviations the predictive posterior mean μ_{mis} of the GP can be calculated. This predicted mismatch is then incorporated into the kernel function by extending the kernel with an additional dimension. This results in the following kernel function

$$\begin{gathered}\boldsymbol{g}(\boldsymbol{x})=\begin{bmatrix}f_{NN}(\boldsymbol{x})\\ \mu_{mis}(\boldsymbol{x})\end{bmatrix}\\ k_{NN,mis}(\boldsymbol{x},\boldsymbol{x}')=\sigma_k^2 exp\left(-\frac{1}{2}\left[\boldsymbol{g}(\boldsymbol{x})\text{-}\boldsymbol{g}(\boldsymbol{x}')\right]^T diag\left(\begin{bmatrix}l_1\\ l_2\end{bmatrix}\right)^{-2}\left[\boldsymbol{g}(\boldsymbol{x})\text{-}\boldsymbol{g}(\boldsymbol{x}')\right]\right)\end{gathered} \tag{9}$$

with two independent length scale parameters l_1 and l_2. The mismatch correction has the effect that two parameter configurations are considered only strongly correlated if they have a similar

simulated objective function value and a similar predicted mismatch. The influence of the mismatch correction on the BO can be illustrated by an example. Assume a parameter configuration which yielded bad performance. Due to the fact that the objective function values are used for correlation, every parameter configuration which yields a similar simulated objective function value will be considered to yield bad performance. This also holds true for parameter configurations which lie in completely different areas of the parameter space. However, assuming mismatches between simulation data and reality, these parameter configurations could still yield promising results on the real system. The mismatch term enables these parameter configurations to be tested as well, as usually the mismatch between parameter configurations which are far away from each other in parameter space are not strongly correlated due to the properties of the mismatch kernel.

2.5 Using Multiple Sets of Simulation Data

So far, the approaches to enhance the sample-efficiency only considered one set of simulation data. However, as explained in the introduction the algorithm should be able to incorporate multiple i.e. n sets of simulation data. This can be achieved by either using a composed kernel function or by combining different GPs within the acquisition function. A composed kernel could be constructed by adding multiple kernels for different simulation data sets. The addition would act like an OR operation as described in [9]. Therefore, two parameter configurations would yield a high covariance if at least one kernel yields a high covariance. Because it also may not reasonable to use the information from every kernel as they may have a too strong mismatch, they can further be weighted individually. To consider multiple GPs within the acquisition function, there exist a few approaches in literature. Namely, the Most Likely Expected Improvement (MLEI) [10] and the Weighted Mixture Expected Improvement (WMEI) [11]. The MLEI determines the EI score and its respective parameter configuration for every GP model and weights the EI score by its marginal likelihood. Then, it chooses the next parameter configuration from the GP with the highest weighted EI score. In contrast, the WMEI acquisition function determines not an individual parameter configuration with its EI score for each individual GP. Instead, it directly determines one parameter configuration by maximizing the weighted EI acquisition functions of the considered GP's. This can be expressed mathematically by

$$\boldsymbol{x}_{n+1}=\underset{\boldsymbol{x}\in A}{\operatorname{argmax}}\ a_{WMEI}\left(\boldsymbol{x}/GP_{1,\dots,n}\right)=\underset{\boldsymbol{x}\in A}{\operatorname{argmax}}\ \left(\frac{1}{n}\sum_{i=1}^{n} w_i a_{EI}(\boldsymbol{x}/GP_i)\right) \tag{10}$$

$$\text{with } w_i = \frac{log\ p_{ML}(\boldsymbol{y}|X,\mathcal{GP}_i)}{\sum_{j=1}^{n} log\ p_{ML}(\boldsymbol{y}|X,\mathcal{GP}_j)}. \tag{11}$$

Again, preliminary tests showed that the WMEI acquisition function is the best performing approach to consider multiple sets of simulation data and is thus used within this work.

2.6 Further Modifications

In order to further enhance the sample-efficiency of the algorithm some additional modifications to the algorithm are introduced. Firstly, the criterion used to weight the models in the WMEI acquisition function is changed to a Monte Carlo (MC) Cross Validation (CV) criterion with the predictive posterior probability as a scoring function. This showed to be more suitable for model selection, as it considers the posterior probability distribution and not the prior probability distribution as the marginal likelihood criterion does. Thus, the calculation of the weights changes to

$$w_i=\frac{CV_{MC}(GP_i)}{\sum_{j=1}^{k} CV_{MC}(GP_j)} \tag{12}$$

More information about the Monte Carlo CV can be found in [12].

Furthermore, the length scale might be chosen unreasonably large when optimizing the hyperparameters of the GP. This may also influence the performance of the BO regarding the

number of evaluations. Therefore, the maximum length scale is bounded by the maximum distance of objective function values within the simulation data space. This is a suitable choice, since in general it is not possible to extrapolate more than l units away from a data point [13].

Combining all previous mentioned modifications results in a novel BO algorithm leveraging prior simulation data to enhance sample efficiency. In summary, it extends the standard BO algorithm by using $n+1$ GP's. Each GP compromises of a constant mean function and an SE kernel, which uses a neural network trained on one simulation data set with an added mismatch correction. The only exception is the last GP, which is just a standard GP with constant mean and SE kernel. After fitting the hyperparameters of the GP's with a bounded length scale, the next promising point is chosen by the WMEI acquisition function. This acquisition function weights the GP's by a Monte Carlo CV.

Lastly, for each simulation data set a different GP model is trained and used in the WMEI acquisition function. However, it might evolve the case that no data set approximates the actual objective function well. To consider this case, an extra GP is introduced with constant mean function and a standard SE kernel. Thus, the algorithm can fall back to a normal BO if no data fits well with the real system. Additionally, to save computational complexity and to facilitate the interpretation of the results, the best scoring GP of the n GP's regarding the MC CV is preselected for the WMEI acquisition function. Thus, only one prior-informed GP and the standard GP is used in the acquisition function.

3. Results

In a first test, the proposed algorithm is tested on the six-dimensional Hartmann function also known as Hartmann6. The function is taken to show the general applicability of the algorithm and can be considered to represent the objective function for an arbitrary aim point controller. In a second test, an actual aim point control, namely the Vant-Hull algorithm, optimized by the proposed algorithm.

3.1 Evaluation Criteria

To evaluate the performance of the algorithm, the number of function evaluation to reach a specified accuracy is evaluated. The accuracy is defined as

$$acc = \left(1 - \frac{|f_{opt} - f^+|}{d_{max}}\right) \cdot 100\%, \tag{13}$$

where f_{opt} is the true optimal value and f^+ the best observed value of the BO. d_{max} denotes the maximum distance between two values in the range of objective function values.

As a baseline to compare the proposed BO algorithm, a standard BO algorithm using a GP with constant mean function and SE kernel, as well as the EI acquisition is chosen. No other baseline is considered because BO already outperforms other commonly used hyperparameter optimizations with respect to sample-efficiency and incorporating prior assumptions about the true objective function [1].

3.2 Hartmann6

The Hartmann6 function is a common test function for optimization problems, usually evaluated on the hypercube $x_i \in [0, 1] \; \forall \, i=1,..., 6$. On this hypercube, the function possesses two local minima. Four simulation data sets are created for this function. The first data set is created from the Hartmann6 function with an additive noise. The second is created with small shifts on two dimensions and the third with a large shift on one dimension. The last data set is generated from a six-dimensional polynomial function. While the first two data sets still resemble the true Hartmann6 function, the third set approximates rather badly and the last set not at all the true

Hartmann6 function. To use the simulations within the BO framework each set is approximated by a neural network.

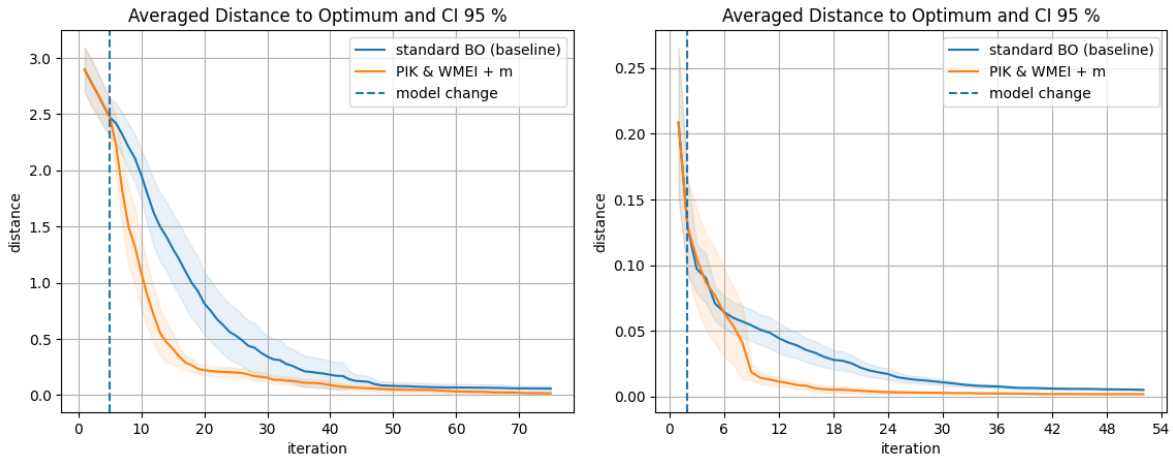
Usually, the BO algorithm is fed with some observation data from random samples before it starts. In this test, each algorithm was fed with observation data from five different random samples. As the performance of the algorithm is also dependent of the initial observation data, 30 optimization runs are performed for the algorithm as well as for the baseline. The results are averaged over all runs. To ensure comparability, the initial observation data in each run is equal for the proposed algorithm and the baseline.

In Tab. 1 the results are shown regarding the number of function evaluations averaged over 30 optimization runs. Each number is rounded to an integer. It can be seen that the proposed algorithm outperforms the baseline for each shown accuracy by at least 33%. To reach 90% the performance goes up and reaches its maximum of 45% less function evaluations than the baseline. However, after that the proposed algorithm slows down but still manages to reach an accuracy of 95% with 14 less function evaluations in average.

Table 1. Number of function evaluations for Hartmann6 function.

Algorithm	**50 % acc.**	**80 % acc.**	**90 % acc.**	**95 % acc.**
BO (baseline)	12 eval.	23 eval.	33 eval.	43 eval.
Proposed BO	8 (-33 %)	13 (-43 %)	17 (-45 %)	29 (-33 %)

In Fig. 1a the distance to the global optimum as well as the confidence interval of 95% is shown. The dotted line denotes the model change, where the actual BO algorithm starts after the random initial samples. It can be observed that the proposed algorithm also stays in tighter bounds, thus its performance depends less on the initial observations compared to the baseline.



(a) Hartmann6 test function (b) Vant-Hull algorithm

Figure 1. Mean distance and 95% confidence interval to the global optimum.

3.3 Vant-Hull Aim Point Controller

The Vant-Hull controller is a simple open-loop control method to prevent violation of the allowable flux density (AFD). The controller locates the aim points with a certain distance from the upper or lower edge of the receiver. This distance is determined by the approximate beam radius and a parameter k for each heliostat. This algorithm was extended by Collado et al. [14] to use three different k factors. Each factor is allocated to one sector of the field, which are divided in radial direction. These factors are optimized in this test. However, the last factor is kept constant as it showed to have no significant influence. The simulation data was created

for the solar tower in Jülich with 2153 heliostats and a norther-field layout. The algorithm was actually designed for an all-round field, however using the solar field in Jülich leaves the possibility to verify the simulation at the real plant as this plant can be used by the author. 14 different data sets were created with varying mirror error (1.5-2 mrad), mirror reflectivity (0.72-0.9) and AFD (750-800 kW/m²). These parameters were chosen as they are usually not exactly known during simulation trials. The objective function was designed to increase with the power on the receiver while being penalized by causing overflux conditions. Again, these data sets are approximated by an NN. One of the data sets is taken to resemble to the true plant and the others are used within the BO algorithm. The algorithms are averaged over 30 runs but this time with only two initial observations due to the smaller number of parameters. The results are listed in Table 2. As the algorithm converges fast for the small problem, values of 90% accuracy and above are shown. This time the proposed algorithm is equally fast to reach 90% accuracy. However, 95% accuracy is achieved faster by requiring 33% less steps and for 98% accuracy 61% less steps. In Fig. 1b the distance and standard deviation is depicted for 52 iterations. One can recognize that the baseline is outperformed by the proposed BO after six steps.

Table 2. Number of function evaluations for Vant-Hull algorithm.

Algorithm	**90 % acc.**	**95 % acc.**	**98 % acc.**	**99 % acc.**
BO (baseline)	4 eval.	12 eval.	23 eval.	32 eval.
Proposed BO	4 (+0 %)	8 (-33 %)	9 (-61 %)	14 (-56 %)

4. Conclusion and Outlook

In this work, a novel approach was proposed for sample-efficient hyperparameter optimization of an arbitrary aim point controller for solar power tower plants. The approach is based on the Bayesian Optimization and was extended to efficiently make use of simulation data. The proposed algorithm was tested on the six-dimensional Hartmann function to evaluate the general applicability on an arbitrary aim point controller. As benchmark, a standard BO approach was used. In this test, the proposed algorithm outperformed the benchmark e.g. the algorithm yielded an accuracy of 95% regarding the optimal value within 33% less function evaluations. In a second test, the algorithms were tested on the modified Vant-Hull aim point controller with two hyperparameters. Here, the improvement was similar. To reach 95% accuracy, 33% less steps are needed. In conclusion, the objective of using multiple simulation data sets to speed up the finding of near optimal controller parameters was achieved. Thus, the algorithm can enable the optimization of aim point controllers at a real plant. However, if the reduction of optimization steps is sufficient depends on the objective function as well as the time to execute one test run at a real plant. In future work, the algorithms shall be evaluated on other aim point control strategies.

Data availability statement

The data is not published due to legal issues.

Author contributions

Conceptualization: D.Z. methodology, formal analysis, software, investigation: B.L., writing - original draft: D.Z., visualization: B.L. and D.Z.; writing - review and editing: B.L., D.M.Q.; supervision: D.M.Q and R.P.-P.

Competing interests

The authors declare no competing interests.

Funding

This work was carried out in the project “SolarFuelNow” (Grant No.: 03EE5042A) with the financial support from the Ministry of Economic Affairs and Climate Action of the State of North Rhine-Westphalia.

References

1. K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, and J.-B. Mouret, “A survey on policy search algorithms for learning robot controllers in a handful of trials,” Jul. 2018. [Online]. Available: https://arxiv.org/pdf/1807.02303
2. C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge Mass.: MIT Press, 2006.
3. G. de Ath, J. E. Fieldsend, and R. M. Everson, “What do you mean?,” 2020. [Online]. Available: https://arxiv.org/pdf/2004.08349
4. A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, “Robots that can adapt like animals,” 7553, Jul. 2014. [Online]. Available: https://arxiv.org/pdf/1407.3501
5. Aaron Wilson, Alan Fern, and Prasad Tadepalli, “Using Trajectory Data to Improve Bayesian Optimization for Reinforcement Learning,” *Journal of Machine Learning Research*, vol. 15, no. 8, pp. 253–282, 2014.
6. A. Marco *et al.,* “Virtual vs. real: Trading off simulations and physical experiments in reinforcement learning with Bayesian optimization,” in *ICRA2017*, Singapore, Singapore, op. 2017, pp. 1557–1563.
7. R. Antonova, A. Rai, and C. G. Atkeson, “Deep Kernels for Optimizing Locomotion Controllers,” *PMLR 78:47-56*, 2017.
8. A. Rai, R. Antonova, F. Meier, and C. G. Atkeson, “Using Simulation to Improve Sample-Efficiency of Bayesian Optimization for Bipedal Robots,” *Journal of Machine Learning Research (JMLR)*.
9. D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani, “Structure Discovery in Nonparametric Regression through Compositional Kernel Search,” Feb. 2013. [Online]. Available: https://arxiv.org/pdf/1302.4922
10. R. Pautrat, K. Chatzilygeroudis, and J.-B. Mouret, “Bayesian Optimization with Automatic Prior Selection for Data-Efficient Direct Policy Search,” Sep. 2017. [Online]. Available: https://arxiv.org/pdf/1709.06919
11. I. Roman, R. Santana, A. Mendiburu, and J. A. Lozano, “An Experimental Study in Adaptive Kernel Selection for Bayesian Optimization,” *IEEE Access*, vol. 7, pp. 184294–184302, 2019, doi: https://doi.org/10.1109/ACCESS.2019.2960498.
12. E. Fong and C. C. Holmes, “On the marginal likelihood and cross-validation,” *Biometrika*, vol. 107, no. 2, pp. 489–496, 2020, doi: https://doi.org/10.1093/biomet/asz077.
13. D. Duvenaud, “Automatic model construction with Gaussian processes,” Apollo - University of Cambridge Repository, 2014.
14. F. J. Collado and J. Guallar, “A two-parameter aiming strategy to reduce and flatten the flux map in solar power tower plants,” *Solar Energy*, vol. 188, pp. 185–189, 2019, doi: 10.1016/j.solener.2019.06.001.