BAYESIAN OPTIMISATION FOR FAST AND SAFE PARAMETER TUNING OF SWISSFEL

J. Kirschner, M. Nonnenmacher, M. Mutný, A. Krause, Dept. of Computer Science, ETH Zurich
N. Hiller, R. Ischebeck, A. Adelmann, Paul Scherrer Institute

Abstract

Parameter tuning is a notoriously time-consuming task in accelerator facilities. As tool for global optimization with noisy evaluations, Bayesian optimization was recently shown to outperform alternative methods. By learning a model of the underlying function using all available data, the next evaluation can be chosen carefully to find the optimum with as few steps as possible and without violating any safety constraints. However, the per-step computation time increases significantly with the number of parameters and the generality of the approach can lead to slow convergence on functions that are easier to optimize. To overcome these limitations, we divide the global problem into sequential subproblems that can be solved efficiently using safe Bayesian optimization. This allows us to trade off local and global convergence and to adapt to additional structure in the objective function. Further, we provide slice-plots of the function as user feedback during the optimization. We showcase how we use our algorithm to tune up the FEL output of SwissFEL with up to 40 parameters simultaneously, and reach convergence within reasonable tuning times in the order of 30 minutes (< 2000 steps).

INTRODUCTION

Empirical parameter tuning is an important and reoccurring task of the FEL setup. Common tuning objectives include the pulse energy or spectrum of the FEL signal and loss minimization, or a combination of multiple criteria. A recent effort started to use optimization methods to automate the tuning process [1–6], and significant speedups compared to manual operator tuning have been obtained. As only point evaluations of the objective function are available, one has to rely on *zero-order or blackbox optimization methods*. Due to their simplicity, the Nelder-Mead (Simplex) [7] algorithm and random walk optimizers have become popular choices to assist operator tuning in some facilities. As an alternative, Bayesian optimization (BO) [8, 9] has recently gained interest also in the accelerator community [1, 2]. Bayesian optimization deals with observation noise in a principled way, allows to leverage prior data and comes with theoretical convergence guarantees in some cases [10]. A variant (SafeOPT) has been proposed [11], that takes additional safety constraints into account. This is of importance for tuning FEL parameters, for instance to avoid electron losses or to maintain a minimum level of the FEL signal during optimization. However, these benefits come at the expense of increased computational requirements as well as additional complexity in the choice and sensitivity of BO hyperparameters. We believe that these issues constitute the main reasons that prevent a more wide-spread use of Bayesian optimization in this context beyond an academic setup. Moving forward with this, we present the following contributions.

- We implement the LineBO [12] algorithm at SwissFEL. The algorithm is designed to be computationally efficient when used with a large number of parameters and allows to take safety critical constraints into account.
- On SwissFEL we compare our methods to a simple parameter scan and the Nelder-Mead method, and obtain promising results when tuning up to 40 parameters.

Mathematical Setting

Let d denote the number of tuning parameters. The parameter space is $\mathcal{X} = [a_1; b_1] \times \cdots \times [a_d; b_d] \subset \mathbb{R}^d$, where $a_i \leq x_i \leq b_i$ defines the allowed range for each parameter $i = 1, \ldots, d$. The objective function (e.g. the FEL pulse energy) is an unknown function $f : \mathcal{X} \to \mathbb{R}$. The constraints are modelled by some unknown $g : \mathcal{X} \to \mathbb{R}$. A parameter $x \in \mathcal{X}$ is said to satisfy the safety constraint if $g(x) \leq \tau$ for a user-specified threshold level $\tau \in \mathbb{R}$. For simplicity, we use the formulation with a single constraint but multiple constraints are possible by choosing g vector valued. The only way to access f and g is to measure at any $x \in \mathcal{X}$, which yields noisy point observations of the objective and the constraint, $y = f(x) + \epsilon$ and $s = g(x) + \eta$. We refer to this process as *evaluating at* x, which in our case includes averaging the FEL signal over multiple shots. Formally, the tuning objective is

$$x^* = \underset{x \in \mathcal{X}; g(x) \leq \tau}{\arg\max} f(x) \,. \tag{1}$$

An optimization algorithm iteratively chooses a sequence of parameters $x_1, \ldots, x_T$ and obtains evaluations of f and g for each point. After T steps (when the user stops the optimization), a final solution x_T is returned. A tuning method is called *safe* if all the iterates $(x_t)_{t=1}^T$ satisfy $g(x_t) \leq \tau$. In the unconstrained case the requirement $g(x) \leq \tau$ is dropped.

BAYESIAN OPTIMIZATION

Bayesian optimization is a method for global optimization. The idea is to learn a model of the objective function based on the data collected during the optimization (and possibly any available prior data). The decision where to place the next evaluation is then based on the model's predictions of plausible locations of the optimum x^* and also takes the model uncertainty into account. More formally, in the unconstrained case, Bayesian optimization uses the following steps.

Step 0) Initialize step counter $t = 0$, data $\mathcal{D}_0 = \{\}$.

Step 1) Compute a Gaussian Process (GP) estimate $\hat{f}_t$ of f using data $\mathcal{D}_t$. Gaussian processes are probabilistic regression models that provide confidence intervals $\hat{f}_t \pm \beta_t \sigma_t(x)$. Here $\sigma_t(x)$ quantifies the model uncertainty at x, and β_t is a scaling factor that controls the coverage probability.

Step 2) Evaluate at $x_t = \arg\max_{x \in \mathcal{X}} \alpha_t(x)$. Here, $\alpha_t(x)$ is a so-called acquisition function based on $\hat{f}_t$ and the model uncertainty, which captures the utility of choosing a point x_t. One of the most common acquisition functions is the upper confidence bound (UCB) $\alpha_t(x) = f_t(x) + \beta_t \sigma_t(x)$. In this case the point with the largest plausible value according to the confidence intervals is chosen.

Step 3) The point x_t is evaluated and the outcome $y_t = f(x) + \epsilon$ is added to the data set $\mathcal{D}_{t+1}$. Increase $t := t + 1$, go to *Step 1)*.

The interested reader may consult the excellent tutorials [8,9] for a more detailed introduction to Gaussian processes and Bayesian optimization. For the constrained case, a safe BO method has been proposed (SafeOPT) [11]. The above outline is then modified to construct estimates $\hat{f}_t$ and $\hat{g}_t$ of f and the constraint function g in *Step 1)*, and the acquisition function in *Step 2)* is re-defined to ensure that a) all evaluations are safe b) the set of safe parameter settings is explored sufficiently and c) the optimum within the set of safe parameters is identified. We describe this approach in more detail below.

Our Approach: SafeLineBO

Bayesian optimization as defined above has been successfully used in many applications, and was shown to outperform alternative approaches on some domains [13]. When applied to high-dimensional settings, the major challenge is to scale *Step 2)*, which itself defines an optimization problem over the parameter space $\mathcal{X}$. The reasoning behind replacing the main objective (1) by the acquisition *Step 2* is that α_t can be optimized offline (without taking evaluations on the machine) and with enough computational power a solution can be obtained in reasonable time. Still, *Step 2* becomes prohibitive as the number of parameters increases. For the case of safe optimization already $d \geq 5$ parameters are problematic, because the SafeOPT method requires a discretization of the domain, hence requires to iterate over exponentially (in d) many points in each step.

To address this short-coming, we use Bayesian optimization combined with a linesearch technique as we previously proposed in [12]. Below we give the algorithmic outline.

Step 0) Initialize $t = 0$, $\mathcal{D}_0 = \{\}$, $\epsilon > 0$, safe starting point x_0.

Step 1) Query f and g at $\sim d$ local perturbations of x_t, and add the evaluations to the data set $\mathcal{D}_t$.

Step 2) Update the Gaussian Process (GP) estimates $\hat{f}_t(x) \pm \beta_t \sigma_t(x)$, $\hat{g}_t(x) \pm \beta_t \rho_t(x)$ of f and g with $\mathcal{D}_t$.

Step 3) Compute the estimated gradient $l_t := \nabla \hat{f}_t(x_t)$.

Step 4) Define 1d subspace $\mathcal{L}_t = \{\lambda \cdot l_t + x_t : \lambda \in \mathbb{R}\} \cap \mathcal{X}$.

Step 5) Use SafeOPT to solve the objective (1) on $\mathcal{L}_t$:

i) Define safe subset of $\mathcal{L}_t$, $\mathcal{L}_s = \{x \in \mathcal{L}_t : \hat{g}_t(x) + \beta_t \rho_t(x) \leq \tau\}$.

ii) Compute a candidate expander point $x_e \in \mathcal{X}_s$ that is predicted to enlarge the estimated safe set.

iii) Compute a candidate optimizer $x_o \in \mathcal{X}_s$ that reduces uncertainty about the optimum on $\mathcal{L}_t$.

iv) Among $\{x_e, x_o\}$ choose the point where the prediction of $\hat{f}$ is more uncertain, ie $x = \arg\max_{\tilde{x} \in \{x_e, x_o\}} \sigma_t(\tilde{x})$.

v) Evaluate at x, update data $\mathcal{D}_t$ and estimates $\hat{f}_t$, $\hat{g}_t$.

vi) As long as error is larger than ϵ, go to *i)*.

Step 6) Until user stops optimization, increase $t := t + 1$, go to *Step 1)*.

A fully formal description as well as a more extensive empirical evaluation of this approach is given in [12]. Importantly, *Step 4-5)* circumvent the expensive maximization of the acquisition function by restricting the global problem to iteratively chosen 1-dimensional subsets of the domain, while keeping all data in a global model. Nevertheless, this approach satisfies strong convergence guarantees [12]. We present the evaluation of this approach on SwissFEL in the next section.

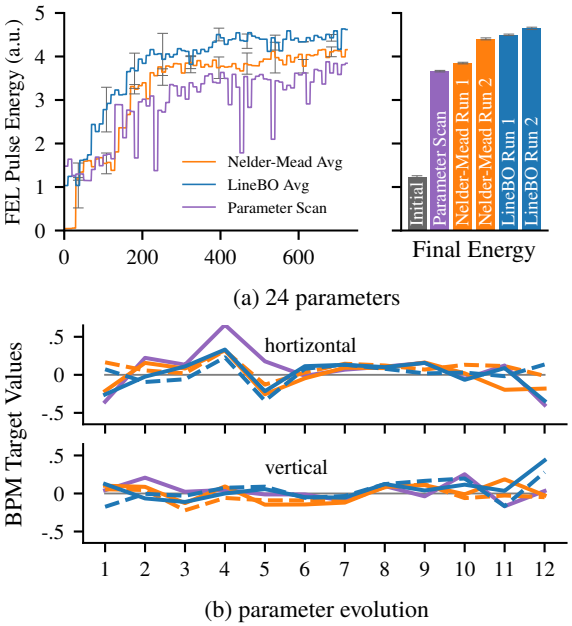
PARAMETER TUNING AT SWISSFEL

We implemented a python interface for optimizers based on pyepics. The optimization framework runs on a server an can be controlled via an application interface. We have beam checks in place, that pause the optimization in case the signal is lost. Further, we have a configurable settling time and we check for feedback error signal to fall below a threshold before we evaluate a point.

Objective: Per shot pulse energy measured with a gas detector [14], averaged over 10 measurements at 25-50 Hz. With the settling and computation time, we are able to run 1-2 optimization steps per second.

Tuning parameters:

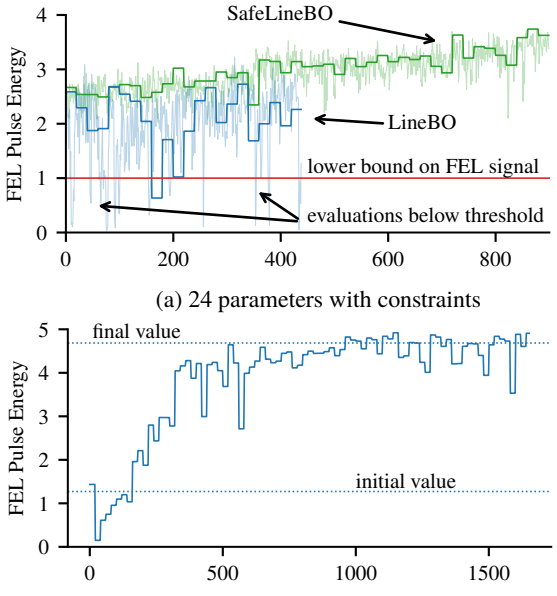
- 24 (horizontal & vertical) beam position monitor (BPM) target values of the orbit feedback through the undulators (maintaining 2 fixed points in each plane).
- 5 matching quadrupoles before the undulators.
- 11 gap settings of the undulators.



(a) 24 parameters

(b) parameter evolution

Figure 1: *Top:* Performance of parameter scan, Nelder-Mead and LineBO without safety constraints in terms of number of evaluations on the machine, using the 24 BPM target values. *Bottom:* Corresponding final BPM target values relative to the starting parameter in the allowed range (±80 µm). Note that the algorithms converge to different solutions, which likely is due to local optima or insensitive parameters.



(a) 24 parameters with constraints

(b) 40 parameters

Figure 2: *Top:* Comparing LineBO and SafeLineBO (with and without safety constraint). In this case SafeLineBO performs significantly better because the search space is more constrained: parameter that yield very low values (<1) are never explored whereas LineBO also evaluates points below this threshold. *Bottom:* Tuning with 40 parameters where we obtain good values with 800 steps (15 minutes).

Constraints: As a constraint we used a lower bound on the FEL signal (Fig. 2a). A possible use-case is parasitical tuning during user operation, where a minimum signal is required. Losses were not a major concern with these parameters, but could be included for other parameters or for optimization on other accelerators (e.g. proton accelerators).

EMPIRICAL RESULTS

We did multiple test runs comparing LineBO and SafeLineBO (with the constraint) to Nelder-Mead and a simple parameter scan baseline, using the 24 BPM target values as tuning parameters. For the parameter scan baseline, we consecutively optimized each parameter with a single scan. For each test run, we manually detuned the machine to a fixed starting point with a low signal value. Additionally, we conducted an experiment with 40 parameters, including BPM target values, matching quadrupoles and gap settings of the undulators. Our findings are presented in Fig. 1 and 2. Note that the figures show results from different tuning sessions. We found that the Bayesian optimization method was able to consistently outperform our baselines, however our method required setting GP hyperparameters such as lengthscales, which we have not fully automatized yet.

OUTLOOK

The ultimate goal is to establish a well-performing optimization method to assist operators with tuning FEL parameters, in particular those that need regular adjustments. For such a method to be practical, the method itself should have at most 1-2 hyper-parameters that need adjustment, where ideally the best setting is relatively robust or obtained with a guiding principle. In our current implementation we explore learning some of the GP hyper-parameters such as lengthscales from initial scans (a process that needs to be done only once); however the results are not yet fully satisfying. An additional advantage of the LineBO method is that one can obtain slice plots of the model predictions on the current line as feedback of the current optimization progress, which can also guide the setting of GP hyper-parameters.

We also think that a more systematic comparison to other methods should be conducted. Another candidate that to the best of our knowledge has not yet been tested substantially for FEL tuning is CMA-ES [15], which however does not directly deal with safety constraints.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the support of the entire SwissFEL team. This research was supported by SNSF grant 200020_159557 and 407540_167212 through the NRP 75 Big Data program. Further, this project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme grant agreement No 815943.

REFERENCES

[1] M. W. McIntire, T. M. Cope, D. F. Ratner, and S. Ermon, "Bayesian Optimization of FEL Performance at LCLS", in *Proc. 7th Int. Particle Accelerator Conf. (IPAC'16)*, Busan, Korea, May 2016, pp. 2972–2975. doi:10.18429/JACoW-IPAC2016-WEPOW055

[2] S. I. Tomin, G. Geloni, I. V. Agapov, W. Decking, M. Scholz, and I. Zagorodnov, "On-line Optimization of European XFEL with OCELOT", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 1038–1042. doi:10.18429/JACoW-ICALEPCS2017-WEAPL07

[3] E. Schuster, C. K. Allen, and M. Krstic, "Optimized Beam Matching Using Extremum Seeking", in *Proc. 21st Particle Accelerator Conf. (PAC'05)*, Knoxville, TN, USA, May 2005, paper FPAT092, pp. 4269–4271.

[4] A. Scheinker, R. W. Garnett, D. Rees, D. K. Bohler, A. L. Edelen, and S. V. Milton, "Applying Artificial Intelligence to Accelerators", in *Proc. 9th Int. Particle Accelerator Conf. (IPAC'18)*, Vancouver, Canada, Apr.-May 2018, pp. 2925–2928. doi:10.18429/JACoW-IPAC2018-THYGBE1

[5] X. Huang, "Robust simplex algorithm for online optimization," *Phys. Rev. Accel. Beams*, vol. 21, p. 104601, Oct 2018.

[6] X. Huang and J. Safranek, "Online optimization of storage ring nonlinear beam dynamics," *Phys. Rev. ST Accel. Beams*, vol. 18, p. 084001, Aug 2015.

[7] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[8] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[9] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[10] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *International Conference on Machine Learning*, 2010.

[11] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with gaussian processes," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 997–1005, PMLR, 2015.

[12] J. Kirschner, M. Mutny, N. Hiller, R. Ischebeck, and A. Krause, "Adaptive and safe Bayesian optimization in high dimensions via one-dimensional subspaces," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 3429–3438, PMLR, 2019.

[13] P.-I. Schneider, X. Garcia Santiago, V. Soltwisch, M. Hammerschmidt, S. Burger, and C. Rockstuhl, "Benchmarking five global optimization approaches for nano-optical shape optimization and parameter reconstruction," *arXiv preprint arXiv:1809.06674*, 2018.

[14] P. Juranić, J. Rehanek, C. A. Arrell, C. Pradervand, R. Ischebeck, C. Erny, P. Heimgartner, I. Gorgisyan, V. Thominet, K. Tiedtke, A. Sorokin, R. Follath, M. Makita, G. Seniutinas, C. David, C. J. Milne, H. Lemke, M. Radovic, C. P. Hauri, and L. Patthey, "SwissFEL Aramis beamline photon diagnostics," *Journal of Synchrotron Radiation*, vol. 25, pp. 1238–1248, Jul 2018.

[15] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.