DEVELOPMENT AND APPLICATION OF ONLINE OPTIMIZATION ALGORITHMS *

Xiaobiao Huang†, SLAC, Menlo Park, CA 94025, USA

Abstract

Automated tuning is an online optimization process. It can be faster and more efficient than manual tuning and can lead to better performance. It may also substitute or improve upon model based methods. Noise tolerance is a fundamental challenge to online optimization algorithms. We discuss our experience in developing a high efficiency, noise-tolerant optimization algorithm, the RCDS method, and the successful application of the algorithm to various real-life accelerator problems. Experience with a few other online optimization algorithms are also discussed. A performance stabilizer and an interactive optimization GUI are presented.

BEAM BASED CORRECTION AND BEAM BASED OPTIMIZATION

Modern accelerators are complex systems that consists of many components. The optimal performance of the machine can be achieved only when all of the essential components are working at the proper settings. A major challenge to the accelerator community is to ensure the machines deliver the best possible performance that meets or exceeds the design requirements.

Accelerators are almost always built and operated according to a design model. Ideally, the machine should perform as the model predicts. However, in reality, all kinds of errors come in, causing deviations in operating conditions from the ideal scenario. For example, in a magnet, there are mechanic errors in the machining of the magnet pole pieces; the magnetization curve of the actual magnetic material may differ from the design; and, the current regulation may fluctuate with temperature and humidity. Magnet alignment errors are another major source of magnetic field errors experienced by the beams. In addition, many electromagnetic fields are typically not included in the models, such as beam induced wakefields, insertion devices, and magnet fringe fields.

Differences between the model and the actual machine can be reduced through precise measurements and correction or compensation of errors of each individual component, improved alignment precision, and including as many physical phenomena in the model as accurately as possible. However, despite our best effort and ever-improving precision in accelerator technology, there will always be differences between the model and the real machine. Beam based methods have to used to mitigate the performance deficiency caused by such differences.

Beam based methods may be divided into two categories - beam based correction (BBC) and beam based optimization (BBO). Beam based correction refers to methods that use beam based measurements to detect deviations of the operating condition of an accelerator sub-system from the ideal setting and use a deterministic, pre-determined procedure to set the operating condition toward the ideal setting. BBC requires beam diagnostics to monitor the beam conditions which provide sufficient information in order to work out the required adjustments of machine setting with a deterministic method. The correction target, or the ideal setting the beam monitors would indicate at optimal performance, is known a priori.

Take orbit correction as an example, beam position monitors (BPMs) are the diagnostics; the method of inverting an orbit response matrix is the correction calculation method; and, the ideal orbit is determined through beam-based alignment measurements or other requirements. In the orbit correction case, an accelerator model is not required. However, for many cases, BBC requires a model as a representation of the ideal target and to be used in the correction calculation. For example, in storage ring optics correction, the ideal target may be represented by the orbit response matrix, or beta functions and phase advances, calculated with the lattice model; the Jacobian matrix of these representing parameters with respect to the quadrupole correctors is also calculated with the model. BBC typically targets a sub-system because the deterioration of the main performance indicators, such as reduced injection efficiency or beam lifetime, usually does not by itself contain enough information that can lead to a deterministic correction.

When any of the required elements, the diagnostics, the deterministic method, or the ideal target, is absent, BBC cannot be done. However, if the machine performance can be measured and the operating conditions can be adjusted, then beam based optimization can be used to improve the performance. The machine performance may be characterized by one or more performance parameters that are basically functions of the adjustable operating parameters (i.e., knobs). The BBO process is to optimize these functions with the available knobs within the proper parameter space. The functions are evaluated through the system (i.e., the machine); but knowledge of the interior of the system is unnecessary (see Fig. 1).

BBO is frequently performed in machine operation in the form of manual tuning. In this case the parameters are changed by literally turning knobs or manually typing in parameter values through a computer. Data processing and implementation of the optimization algorithm are done with a human brain. Alternatively, BBO can be conducted automatically with a computer. In this sense it may be referred to as automated tuning. Compared to manual tuning, automated tuning has the advantages of being fast, independent

* Work supported by DOE Contract No. DE-AC02-76SF00515
† xiahuang@slac.stanford.edu

This material is based upon work supported by the U.S. Department of Energy, Office of Science, under Contract No. DE-AC02-76SF00515.

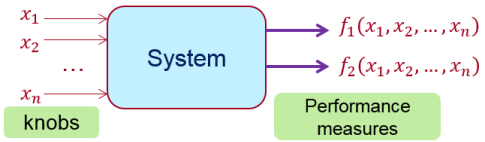

Figure 1: Function evaluation through the machine in beam based optimization.

of human operators, and scalable to large problems. Automated tuning for accelerators has been done before [1]; but until recently it had not become very popular. This is probably because of the lack of reliable, effective online optimization algorithms.

The biggest challenge to online optimization algorithms is that functions evaluated on a machine are noisy. Most of the traditional optimization algorithms are designed for smooth functions. The optimum search strategies for those algorithms often involve comparison of function values between data points. The comparison result could be altered by the noise, causing the algorithms to fail to converge. Online optimization algorithms need to be efficient, i.e., being able to converge to the optimum in as few function evaluations as possible. Furthermore, they should be safe, reliable, and robust. For example, the algorithms should survive occasional outliers in evaluated data and should behave properly in case of machine failures.

The robust conjugate direction search (RCDS) method was proposed specifically for online optimization [2]. It has been shown to be reliable and efficient in many online optimization applications. In the next few sections we will first discuss the RCDS algorithm and its applications, followed by comments on a few other algorithms that may be used for online optimization. In the end we present a performance stabilizer and an interactive optimization graphical user interface (GUI).

DEVELOPMENT AND USAGE OF THE RCDS ALGORITHM

The RCDS algorithm

The RCDS algorithm is a single-objective, direct search method for function optimization. The algorithm iteratively searches along a set of directions in the parameter space for the minimum. Ideally, the initial direction set consists of mutually conjugate directions, i.e., any pair of directions, $\mathbf{u}$ and $\mathbf{v}$, satisfy $\mathbf{u}^T \cdot \mathbf{H} \cdot \mathbf{v} = 0$, where $\mathbf{H}$ is the Hessian matrix of the objective function $f(\mathbf{x})$, with $H_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. The algorithm implements Powell's method to automatically build up a conjugate direction set by replacing the original directions according to search result. However, for online applications, one usually does not run the algorithm for enough iterations to benefit from this mechanism. An approximate conjugate direction set can be calculated if a machine model is available.

The effectiveness of the RCDS method in optimizing noisy functions come from the robust line optimizer that it uses. The line optimizer optimizes the 1-D function $g(\alpha) = f(\mathbf{x}_0 + \alpha\mathbf{u})$ for the direction defined by vector $\mathbf{u}$. The robust line optimizer takes two steps to locate the nearby local minimum. First, it brackets the local minimum by searching both ways until either the boundary of the parameter space is reached, or a point α_b is found that satisfies $g(\alpha_b) - g(\alpha_{min}) > 3\sigma_f$, where σ_f is the noise sigma of the objective function and α_{min} represents the minimum point of the present search. Point α_b or the parameter boundary then defines one limit of the bracket. The step size of the bracketing search increases by each step to avoid inefficiency. Second, the algorithm fills in additional points within the bracket, if necessary, and then fits the data points to a parabola from which the location of the local minimum is derived. The noise sigma is used to detect potential outliers in the fitted data. If an outlier is found, it is discarded and the curve is re-fitted. Because of the robust line optimizer takes noise into consideration in both bracketing and fitting steps, it is robust against noise and outliers.

The parameter space for RCDS is bounded through user defined parameter ranges, as is appropriate for an algorithm that operates over real machines. In an RCDS implementation, the parameter space should be normalized, e.g., with each parameter ranging within [0, 1]. Parameter normalization essentially decouples the algorithm from the actual applications it is applied to.

Simulation and experiments have demonstrated the effectiveness of the RCDS algorithm on real-life accelerator problems, some of which with high random noise. Interested readers may refer to Ref. [2] for more information.

Usage of the RCDS code

A Matlab implementation of the RCDS algorithm has been developed and is available from the author. The usage of the code is very simple and straightforward. The user only needs to define the objective function in a Matlab function and set up a launching script, both of which can be modified from the example included in the package.

The objective function takes a vector of normalized parameters, which represent the solution to be evaluated, as the input argument and returns the evaluated (measured) function value. Inside the function, the normalized parameters are first converted to the actual machine parameters. Any sanity check can be done here, too, if necessary. The parameters are then set to the machine. A waiting period may be inserted to ensure the machine settles to the new operating condition. This could also be done by repeatedly checking the read-back parameters. Then the machine performance parameter(s) are measured, from which the objective function value is evaluated. The operating condition and performance parameters, and any other parameters worth recording, are all saved in an entry of a global data array. At the end of this function, it is advisable to print out some vital information such as the total number of evaluations,

the machine parameters, and the objective function value so that the user can monitor the progress.

In the launching script, the user defines the number of operation parameters, the parameter ranges, the initial solution, the noise level of the objective function, and the initial direction set. The noise level can be measured by evaluating the objective function multiple (say, 20) times. The initial direction set can be represented by the identity matrix, if a conjugate set is not available. The global data array is initialized. The algorithm main function is then launched. Termination conditions can be passed in terms of the number of total iterations and the number of total number of function evaluations. But often times the user can manually terminate the program after a few iterations and when no more gains is being made. After that, the user can sort all evaluated solutions and apply the best solution to the machine.

A Python implementation is being developed and will also be made available to readers of interest.

APPLICATIONS OF THE RCDS ALGORITHM

In addition to applications on SPEAR3 as reported in Ref. [2], the RCDS method has found more applications on SPEAR or at other facilities. Some of these applications are worth reporting here in order to give readers an idea over the potential of the method.

- SPEAR3 dynamic aperture optimization [3]: the RCDS method was used to optimize the dynamic aperture of the SPEAR3 storage ring using sextupole knobs. There are 10 sextupole power supplies for SPEAR3, 8 combinations of which that do not change the chromaticities were derived using the chromaticity response matrix. The injection efficiency was used as the objective function. The injection kicker bump size was decreased to reduce the initial injection efficiency. The RCDS algorithm was then applied to improve injection efficiency with the 8 sextupole knobs by enlarging the dynamic aperture of the ring. The dynamic aperture was increased from 15.1 mm to 20.6 mm.

- LCLS undulator taper profile optimization [4]: the LCLS photon beam power can be improved by adjusting the undulator gaps along the electron beam path (i.e., tapering) to optimally match the changing electron beam energy and bunching condition. The undulator tapering profile was assumed to be a linear plus quadratic curve with four control parameters. Two phase shifters were also included as optimization parameters. The measured photon beam power was the objective function. After initially tuning the parameters away from the optimal condition, RCDS was able to restore the beam power within ∼30 min (about 150 evaluations).

- BEPC-II luminosity optimization [5]: The measured specific luminosity of the BEPC-II collider was used as the objective function. In the tests beam orbit steering at the interaction point (IP) and the horizontal-vertical linear coupling at the IP were used as optimization knobs, respectively. In both cases RCDS was able to quickly recover the luminosity to previously established optimal value.

  In another test, three combinations of 8 quadrupoles around the IP were created to change the beta functions and dispersion function at the IP while keeping the betatron tunes and α_x, α_y, and D'_x at the IP fixed. After these knobs were optimized for both of the electron and positron rings, respectively, the luminosity was increased by a total of ∼15%.

- ESRF beam lifetime optimization [6]: Sextupoles knobs, either using sextupole correctors or the main sextupole families, were used as optimization parameters. The objective function was measured lifetime normalized by beam current, calculated bunch length, and measured average vertical beam size. Beam lifetime was substantially improved within ∼ 300 evaluations.

  At ESRF the RCDS method was also successfully used in coupling correction and injection beam steering experiments.

OTHER OPTIMIZATION ALGORITHMS

Other algorithms may also be used for online optimization. The applicability of an algorithm to a particular problem depends on the nature of the problem, such as the noise level and the complexity of the functional dependence over the parameters. Generally speaking, the RCDS algorithm is an ideal choice considering robustness and efficiency. In the following we make comparisons of RCDS and some other commonly used algorithms.

Iterative parameter scan is an intuitive method and is widely used [1]. It can be very effective for simple problems. However, it is not very efficient compared to RCDS. First, it does not take advantage of conjugate directions. For problems with highly coupled parameters (such as linear coupling correction with skew quadrupoles) it may take many iterations to do what one iteration of conjugate direction search does. Second, parameter scan usually cover the whole parameter range with fixed number of steps or fixed step size. This is not as efficient and accurate compared to the RCDS line optimizer, which uses bracketing with variable step size and quadratic fitting.

The downhill simplex method is a popular choice which is known for fast convergence. But it could fail for some problems [7]. It also has additional disadvantages for online optimization. First, the downhill simplex method assumes an unbounded parameter space, which may be problematic for online applications. Second, as was shown in Ref. [2], when measurement noise starts to affect the vertex comparison results, the simplex method stops to converge, preventing reaching the optimum, or at least limiting its accuracy.

Genetic algorithms are capable of locating the global optimum in a complex terrain. However, generally speaking,

they are not efficient as is required for online optimization. An additional disadvantage of the genetic algorithms, as revealed in Ref. [2], is that solutions biased favorably by noise tend to enter the next generation in the selection operation which defeats the evolution strategy and may prevent convergence to the optimum. An experiment of using a genetic algorithm for coupling correction has been done on SPEAR3 [8]. The beam loss monitor signal was used as the objective function, which has much lower noise level than the case in Ref. [2]. It took the genetic algorithm 20,000 evaluations to reach the same level of coupling correction as what the RCDS algorithm achieved in 300 evaluations.

Particle swam optimization (PSO) is another type of stochastic optimization method that is capable of finding the global optimum. It was shown that PSO is more efficient than genetic algorithms for some accelerator applications [9, 10], due to improved diversity in the new solutions evaluated. The PSO has been experimentally tried on SPEAR3 for the coupling correction problem using the same setup as in Ref. [8]. It took less than 3000 evaluations to reach the same coupling correction result (Fig. 2). The PSO method has also been used in SPEAR3 dynamic aperture optimization [3].

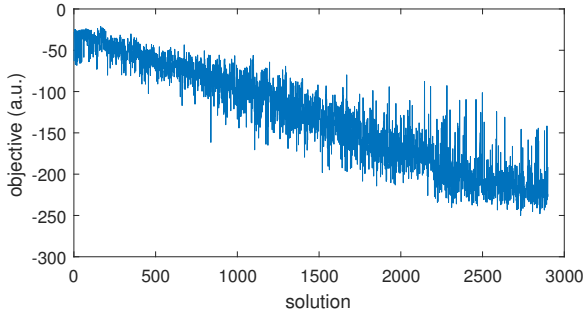

Figure 2: History of objective function in the SPEAR3 coupling correction optimization with PSO experiment.

The extremum seeking (ES) method was first proposed as a feedback algorithm for system stabilization [11]. Its use in online optimization was recently tested on SPEAR3 with the kicker bump matching problem [12]. Its ability of tracking a time-varying system was demonstrated in this test. In this method the optimization parameters are periodically rotated, each with different frequency and a phase modulation related to the objective function. In the high frequency limit, the optimizer's behavior approaches that of a gradient descent method. The disadvantages of the ES method as an online optimization method include (1) the optimization parameters are not bounded; (2) the search for optimum is not direct and thus not the most efficient; (3) the algorithm control parameters such as the amplitudes and frequencies of parameter rotation need to be tuned for each application.

VARIANTS OF RCDS

Two useful variants of the RCDS were developed for different practical purposes. One is a performance stabilizer which is designed to stabilize system performance during operation. The other is an interactive, automatic tuner that gives operator easier access and better control over the optimizer.

The performance stabilizer

Earlier on in the development of RCDS we realized the need for an algorithm that stabilizes the performances of systems in normal operation in response to potential drifting. For such an purpose unnecessary probe and frequent, large deviations should be avoided. We developed an performance stabilizer whose algorithm is similar to RCDS but with significant differences. For example, a target performance is set. When performance is above this level, no action is taken. There is no bracketing of minimum or fitting of data points. There is also no update of direction set. The algorithm simply probes in each direction and decides if the working point should be moved. The working point is moved if the objective function value of a new point is better than the present working point by more than $1.5\sigma_f/\sqrt{N}$, where N is the number of evaluations at the new point.

The performance stabilizer was tested with injection beam steering for SPEAR3. In a 2013 test, the stabilizer tuned four steering magnets at the end of the Booster-to-SPEAR3 (BTS) transport line. As is shown in Fig. 3, when two upstream magnets (BTS-COR7 and BTS-COR6) were manually changed, injection rate was immediately reduced. In both cases the stabilizer successfully compensated the steering changes and restored the injection rate.

AutoTuner - an interactive optimization GUI

A graphical-user-interface (GUI) is significantly more user friendly than a script. A GUI could be used by operators with little training while modifying and running a script always come with some risks and usually should be done only by experts. It would be ideal to have an online optimization GUI that is set up for the routine control room tasks to be used by experts and non-experts alike. We recently developed such a tool called the AutoTuner.

The AutoTuner optimization algorithm is based on RCDS. However, unlike the usual RCDS algorithm which is executed sequentially and the only way to interrupt is to terminate it, the AutoTuner is implemented to allow user interaction. The progress of the optimization run is plotted in real time. Users can pause, resume, and stop the execution of the optimization session.

The knob and the objective function can be simple PVs, or Matlab functions. Multi-knob is supported. The default optimization setups, including knob PV or function name, objective PV or function name, the parameter range, and noise level are done through a table. The default values can be changed on the GUI.

The AutoTuner has been tested on SPEAR3 and its injector with many knobs, such as Booster injection septum, gun phase, linac klystron phases, and SPEAR3 injection kicker parameters. Fig. 4 shows a snapshot of the GUI and an example when it was used to optimize the injection kicker bump with a multi-knob.

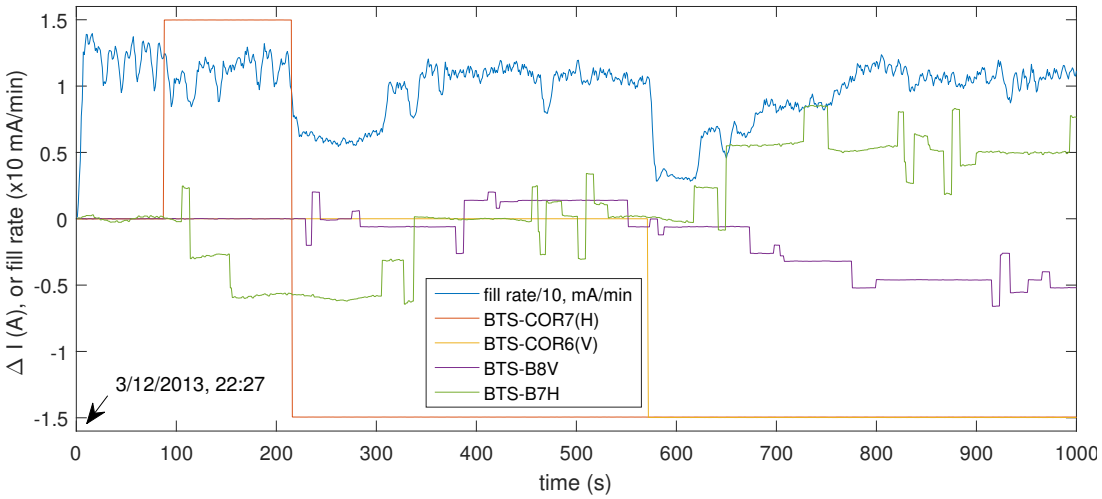

Figure 3: In this 2013 test, the stabilizer responded to upstream trajectory changes by tuning downstream steering magnets (BTS-B7H, BTS-B8V and two others not shown) to restore the fill rate.

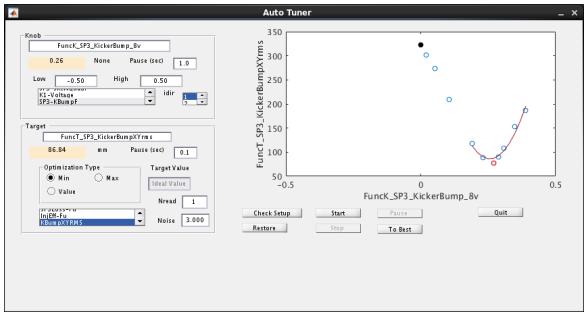

Figure 4: The AutoTuner GUI.

CONCLUSION

We advocate that automated tuning, or online optimization should be more broadly used to improve accelerator performance. The RCDS algorithm is a powerful tool developed specifically for the purpose of online optimization. Since it was proposed in 2013, it has found real-life applications at many facilities through which its effectiveness in online optimization has been demonstrated.

In this report we discussed the importance of beam based optimization, the features and selected applications of the RCDS algorithm, and commented on a few other online optimization algorithms. We presented a performance stabilizer algorithm which could be used during operation as a performance feedback. A recent development, an interactive online optimization GUI, is also presented.

ACKNOWLEDGMENT

We thank James Safranek for many helpful discussions. We also thank users of RCDS that help demonstrated its usefulness, especially Juhao Wu (SLAC), Yi Jiao and Hongfei Ji (IHEP), and S. M. Liuzzo (ESRF).

REFERENCES

[1] L. Emery, H. Shang, M. Borland, PAC'03, Portland, Oregon (2003).

[2] X. Huang, J. Corbett, J. Safranek, J. Wu, Nucl. Instr. Meth. A, 726 (2013) 77-83.

[3] X. Huang, J. Safranek, Phys. Rev. ST Accel. Beams 18, 084001 (2015).

[4] J. Wu, K. Fang, X. Huang, unpublished (April 2014).

[5] H.-F. Ji, Y. Jiao, S. Wang, D.-H. Ji, C.-H. Yu, Y. Zhang, X. Huang, Chinese Physics C 39 (12), 127006 (2015).

[6] S. M. Liuzzo, N. Carmignani, L. Farvacque, B. Nash, T. Perron, P. Raimondi, R. Versteegen, S. M. White, Proc. of IPAC'2016, THPMR015, (2016).

[7] R. M. Lewis, V. Torczon, M. W. Trosset, J. Comp. and Appl. Math. 124 (2000), 191.

[8] K. Tian, J. Safranek, Y. Yan, Phys. Rev. ST Accel. Beams 17, 020703 (2014).

[9] X. Pang, L. Rybarcyk, Nucl. Instr. Meth. A 741, 124 (2014).

[10] X. Huang, J. Safranek, Nucl. Instr. Meth. A 757, 48 (2014).

[11] A. Scheinker, M. Krstic, IEEE Trans. Automatic Control, 58, 1107 (2013).

[12] A. Scheinker, X. Huang, J. Wu, SLAC-PUB-16508 (2016).