ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB221

MULTI-OBJECTIVE OPTIMIZATION WITH ACE3P AND IMPACT*

D. A. Bizzozero†, J. Qiang, Lawrence Berkeley National Laboratory, Berkeley, USA
L. Ge, Z. Li, C.-K. Ng, L. Xiao, SLAC National Accelerator Laboratory, Menlo Park, USA

Abstract

Photo injector design is an important consideration in the construction of next generation accelerators. In current injector optimization, accelerator components (e.g. RF cavities) are individually shape-optimized for performance subject to general requirements such as peak surface field, shunt impedance, and resonant frequency. Once these components' shapes are determined, beam dynamics simulations optimize the injector lattice by adjusting parameters such as the amplitude and phase of the driving fields. However, this form of beam dynamics optimization is restricted by the fixed geometrical shape and field profile of the components.

For a more general and unrestricted accelerator design optimization, a coupled optimization of the cavity shape and beam parameters is required. For this coupled optimization problem, we have created an integrated ACE3P-IMPACT workflow. Within this workflow, the geometries for a set of components are adjusted, the field modes are then computed with Omega3P, a module in the ACE3P suite, and imported into IMPACT-T for beam dynamics simulation. This workflow is encapsulated into a multi-objective optimization algorithm using the DEAP [1] and libEnsemble [2] Python libraries to yield a pareto-optimal set of solutions for a simplified injector design model.

INTRODUCTION

In previous studies, optimization of accelerator components and lattices have been done separately in a set of isolated optimizations. For example, an RF photocathode cavity can be designed around a target frequency, acceleration gradient, and peak surface field. Once this cavity is optimized, it's design is held fixed when optimizing the next component or the phases and amplitudes for an accelerating lattice around other output parameters such as minimum bunch length and transverse emittance.

While this form of sequential optimization may yield reasonable results, it will optimize locally only on a subset of the input parameter space at a time. By contrast, in end-to-end global optimization, all input parameters ranging from the photocathode shape parameters to the lattice phases and amplitudes can be adjusted simultaneously. The global optimization of input parameters will always yield a solution set of inputs at least as good as those obtained by sequential optimization provided enough iterations.

We have developed a systematic integrated workflow management system, titled A3PI (ACE3P with Impact-T), written in Python which interfaces various codes with genetic algorithms and built-in parallelism to perform end-to-end global multi-objective optimization on high performance computing (HPC) systems.

WORKFLOW OVERVIEW

Here we provide an overview of A3PI which manages the workflow of tasks for optimization and interfaces the component codes: Cubit [3], ACE3P [4, 5], and Impact-T [6–8]. For example, A3PI can be used to automate a chain of tasks such as: (1) run Cubit to generate the geometry of an accelerator cavity, (2) run acdtool (a subprogram of ACE3P) to convert the output mesh from Cubit to a format suitable for ACE3P, (3) run Omega3P (a module of ACE3P) to compute eigenmodes of the meshed geometry, (4) run acdtool again to extract the modal fields on a Cartesian grid for use in Impact-T, and (5) run Impact-T with the external fields provided. This example of a single run is shown in Fig. 1.

Thus, A3PI can encapsulate the workflow as a single function evaluation where various parameters are the inputs and the Impact-T particle data are the outputs. Depending on the complexity of the desired workflow, the setup file of A3PI can be quite long as it contains all necessary information to run each code individually. However, A3PI is modular in that if a given component code (e.g. Cubit) isn't necessary, it can be omitted in the setup file. The advantage to this approach is that A3PI can use a parsing utility in Python to replace necessary values for each input file automatically. For example, if the length of a cavity is to change, then A3PI can automatically replace the appropriate values in the Cubit input file as well as the lattice section for Impact-T.

The A3PI tools also include post-processing MATLAB [9] scripts to assist in visualizing the optimization process including a interactive plot sliders and custom data tooltips to quickly view the set of input parameters for any selected individual. We also included visualization tools to further investigate individual evaluations by importing field data, mesh data, and/or particle data used in the varying components of the A3PI workflow. These tools are interactive and can be used to generate high-quality videos of optimizations or Impact-T simulations. An example particle plot created using an A3PI plotting tool is shown in Fig. 2.

MULTI-OBJECTIVE OPTIMIZATION

In the previous section, we discussed how to set up A3PI to run a single chain of tasks for a given set of parameters. The next step is to use this workflow as a black-box function to optimize a set of input parameters with respect to desired output quantities of interest. For the multi-objective optimization, we use the DEAP [1] Python library to set up a genetic algorithm; we opt to use NSGA-II but there are other algorithms to choose from for various types of problems.

* Work supported by U.S. Department of Energy
† dbizzozero@lbl.gov

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB221

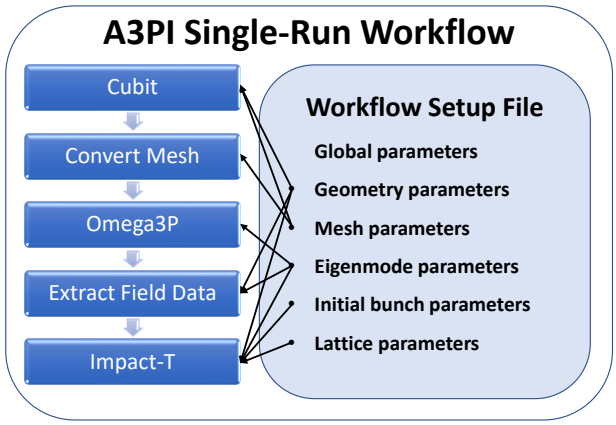

Figure 1: Example layout of a setup of A3PI to run a chain of tasks given a set of run parameters. The A3PI code will automatically generate input files for the component codes and run them sequentially.

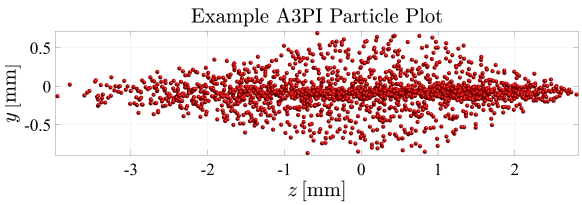

Figure 2: In this demo from an injector simulation with 10^6 macro particles, 2000 particles are shown using the A3PI particle plotting MATLAB script.

While we could simply wrap a multi-objective optimizer around the A3PI workflow, if the individual workflow evaluations are costly, convergence to a Pareto front may take a very long time. To further parallelize A3PI, we use the libEnsemble [2] Python library to assist with the task distribution to multiple nodes in an HPC environment. One particular challenge is that the tasks within the A3PI workflow may use varying amounts of resources: Omega3P in particular has a large memory footprint to accurately solve for eigenmode fields. Thus, by using libEnsemble, A3PI can efficiently distribute tasks to computing resources for the optimization cycle.

To set up A3PI for use with DEAP and libEnsemble, an additional section is added to the A3PI setup file which contains optimization parameters such as population size, mutation parameters, etc. in addition to the libEnsemble-specific options desired. When A3PI is called with these additional settings, a manager process distributes a single-run workflow, as in Fig. 1, to different "worker" folders for parallel evaluation. The choice of input parameters from the manager process is passed to each worker and the workflow is evaluated; the output quantities of interest from the workers are then returned to the manager process for mutation in choosing the next population generation. When the desired convergence is achieved or a maximum number of evaluations is performed, the manager process returns the results of the optimization. An overview diagram of the optimization hierarchy is shown in Fig. 3.

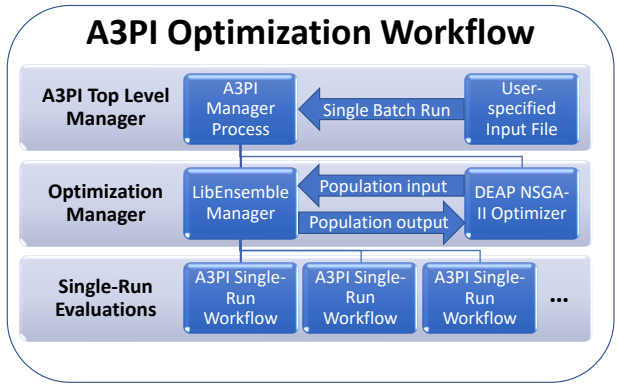

Figure 3: Example layout of a setup of A3PI to run a chain of tasks given a set of run parameters. The A3PI code will automatically generate input files for the component codes. Each single-run evaluation is performed in parallel while the tasks within each run are sequential.

INJECTOR OPTIMIZATION EXAMPLE

To showcase the capability of A3PI, we use a toy-model of an injector lattice consisting of a photocathode gun with a movable cathode stalk (Fig. 4 (top)), a focusing solenoid, and four standard 9-cell accelerating cavities (Fig. 4 (bottom)). The input parameter space we optimize over is given in Table 1. Other necessary simulation parameters such as bunch charge or cavity RF power are either held fixed or determined from the input parameters; some of these additional fixed parameters are given in Table 2. The ranges for phase parameters were chosen after using A3PI in single-run mode to determine a suitable range.

For the multi-objective optimization test, we set A3PI to minimize the final transverse RMS emittance and bunch length while maintaining the constraint that the beam energy must be greater than 60 MeV. Such constraints are enforced via a penalty scaling term to objectives.

Additionally, since all parameters except for d_{cathode} are only used by Impact-T, we can simplify the A3PI workflow by using field interpolation. For this procedure, the *Cubit → Convert Mesh → Omega3P → Extract Field Data* task chain is initially performed only 6 times to obtain field data using $d_{\text{cathode}} = \{0, 2, 4, 6, 8, 10\,\text{mm}\}$. These field maps are then used to construct approximate the fields at other values of d_{cathode} within the defined range. Thus a shortened A3PI workflow can be set up for this model by replacing the aforementioned task chain with a very fast field lookup and interpolation routine.

While this interpolation shortcut introduces a small error in the optimization cycle, the Pareto-optimal parame-

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB221

ters from the shortened workflow can be validated and re-optimized by using the complete workflow for a few additional population generations.

To test the optimization model, we set the population size to 128 individuals and ran an A3PI workflow for a total of 50 generations with the NSGA-II optimizer. Each individual evaluation using the shorter interpolation-based workflow takes approximately 1 minute on a KNL node on Cori@NERSC, thus using 128 KNL nodes, this full optimization can be completed in less than an hour. However, when using the full task chain workflow, Omega3P generally requires more CPU and memory resources than other tasks and such that a single-run evaluation uses approximately 10 minutes with 10 KNL nodes. Therefore, it is not recommended to use the full workflow until the population is close to the Pareto-front; in our example, after 40~50 generations.

As shown in Fig. 5, the individuals approach an approximate Pareto-optimal boundary after a few dozen generations. Each individual reflects a choice of the 9 parameters given in Table 1 which are then evaluated with the A3PI single-run workflow; the output final transverse emittance and bunch length are plotted.

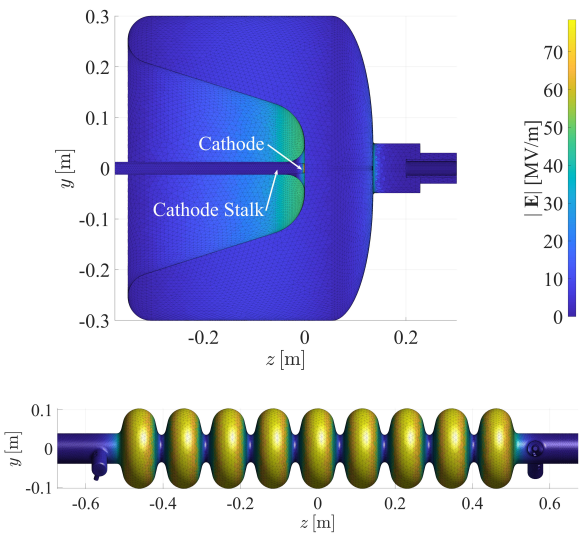

Figure 4: (Top) Slice-view of model RF photocathode with movable stalk. As the cathode depth d_{cathode} increases, the cathode can move inwards which changes the E_z on-axis field. (Bottom) Mesh geometry of one of the 9-cell 1.3 GHz accelerating cavities using the A3PI mesh plotting utility.

CONCLUSION

Multi-objective optimization of an end-to-end accelerator structure is an important aspect in the design of next-generation particle accelerators. For this task, we created A3PI, our novel integrated workflow manager written in Python, to interface existing field and particle codes. We have demonstrated this end-to-end optimization capability with a test model injector but aim to use A3PI for a realistic photoinjector model soon.

Table 1: Model Injector Input Parameters

Parameter	Range	Description
d_{cathode}	0 -10 mm	Cathode stalk position
θ_{cathode}	150 -170 deg	Cathode driving phase
$\sigma_{x,y}$	400 -800 µm	Laser trans. spot size
σ_z	25 -40 µm	Laser pulse length
B_{solenoid}	100 -200 mT	Focusing sol. strength
θ_{accel1}	280 -310 deg	Accel. cavity 1 phase
θ_{accel2}	280 -310 deg	Accel. cavity 2 phase
θ_{accel3}	280 -310 deg	Accel. cavity 3 phase
θ_{accel4}	280 -310 deg	Accel. cavity 4 phase

Table 2: Model Injector Additional Parameters

Parameter	Value	Description
q_{bunch}	200 pC	Electron bunch charge
ω_{cathode}	200 MHz	Cathode mode freq.
W_{total}	8.85 µJ	Cath. cav. total energy*
ω_{accel}	1.3 GHz	Accel. cavity freq.
$E_{z,\text{accel}}$	30 MV/m	Accel. cavity peak field

*: Cathode E_z field varies from 25~45 MV/m with stalk position.

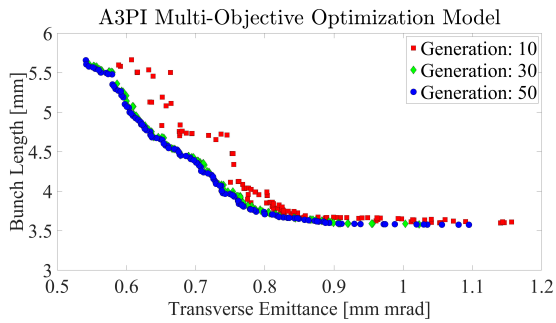

Figure 5: Best individuals at varying generations from the NSGA-II multi-objective optimization of the A3PI injector model.

Our future goals include adding compatibility of other codes to A3PI, for example to enable optimization of storage rings, and further improvements to the A3PI utilities in more versatile computing environments.

ACKNOWLEDGMENTS

This work is supported by the Director of the Office of Science of the US Department of Energy under contracts DE-AC02-05-CH11231 and DE-AC02-76-SF00515.

REFERENCES

[1] F.-A. Fortin *et al.*, "DEAP: Evolutionary Algorithms Made Easy", *J. Mach. Learn. Res.*, vol. 13, pp. 2171–2175, Jul. 2012.

[2] Stephen Hudson *et al.*, *libEnsemble User Manual*, Argonne National Laboratory, USA, 2020, https://libensemble.readthedocs.io/en/main/

ISBN: 978-3-95450-214-1 ISSN: 2673-5490 doi:10.18429/JACoW-IPAC2021-THPAB221

[3] R. W. Quadros *et al.*, *The CUBIT Geometry & Meshing Toolkit*, Sandia National Laboratories, USA, 2021; https://cubit.sandia.gov

[4] ACE3P (Advanced Computational Electromagnetics 3P) Code Suite: https://portal.slac.stanford.edu/sites/ard_public/acd/

[5] L.-Q. Lee "A Parallel Finite-Element Eigenmode Analysis Code for Accelerator Cavities", SLAC, Menlo Park, USA, Rep. SLAC-PUB-13529, Mar. 2009.

[6] J. Qiang, S. Lidia, R. Ryne, and C. Limborg-Deprey, "Three-dimensional quasistatic model for high brightness bream dynamics simulation", *Phys. Rev. ST Accel. Beams*, vol. 9, p. 044204, Apr. 2006. doi:10.1103/physrevstab.9.044204

[7] J. Qiang *et al.*, "High resolution simulation of beam dynamics in electron linacs for x-ray free electron lasers", *Phys. Rev. ST Accel. Beams*, vol. 12, p. 100702, Oct. 2009. doi:10.1103/physrevstab.12.100702

[8] J. Qiang, R. Ryne, S. Habib, and V. Decyk "An Object-Oriented Particle-In-Cell Code for Beam Dynamics Simulation in Linear Accelerators", *J. Comp. Phys.*, vol. 163, p. 434, 1999. doi:10.1145/331532.331587

[9] MATLAB version 8.5.0.197613 (R2018a). The MathWorks Inc., Natick, Massachusetts, United States.