An Integrated Multi-Physics Optimization Framework for Particle Accelerator Design

Gongxiaohui Chen[a], Tyler Chang[a], John Power[a], and Chungunag Jing[b]

[a]Argonne National Laboratory, [b]Euclid TechLabs

WSC 2023, San Antonio, TX, USA
Dec XX, 2023

Outlines

Introduction to a simplified accelerating beamline

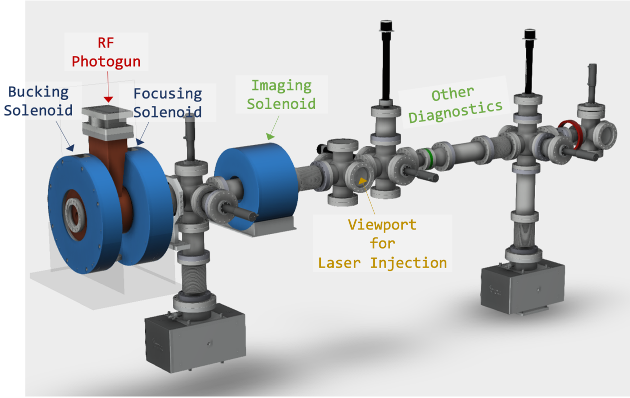
An integrated framework for global optimization

Optimization methods

Experiment setup

Conclusion and Future Work

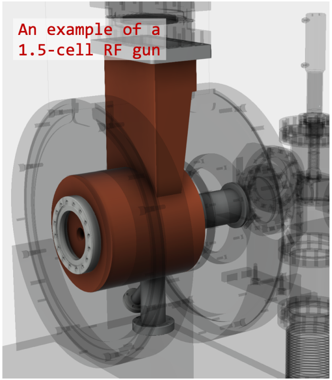
Introduction to a simplified accelerating beamline

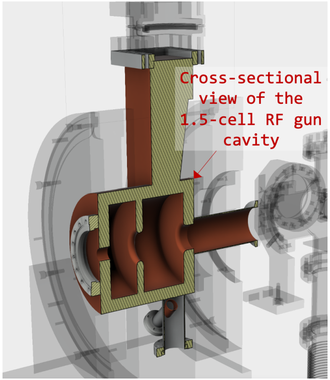


A standard accelerating beamline includes:

- RF photogun
- Bucking/focusing solenoid (or main solenoid)
- Other magnets or diagnostics

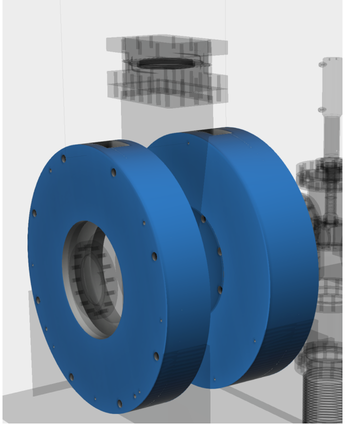
Introduction to a RF photogun





- RF gun cavity which generates the **electromagnetic field** is used to accelerate the electron beams that emitted from the cathode.
- The geometry of the gun needs to be carefully optimized in order to have the desired resonant frequency, Q factor etc.

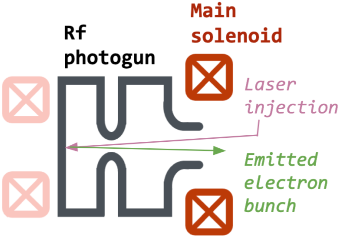
Introduction to a solenoid magnet



- A solenoid magnet(s) which generates the **magnetic field** is commonly installed around the RF gun to focus and to confine the transverse emittance (ε_{trans}) of the electron beam.

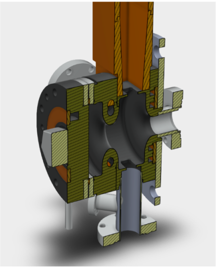
Motivation

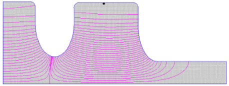
Ultimate Goal is to produce high brightness beam through the designed beamline (including a RF gun cavity and a main solenoid).
Traditional way for designing a beamline starts from individual components optimization cavities, magnets and beam dynamics are optimized independently by separate individuals using different codes with isolated targeting objectives, which may not be a direct way of achieving the goal.
Our proposed way for designing a beamline is by setting up a unified framework by integrating various modules, with the desired objectives (i.e. ε_{trans}, frequency etc.).



Introduction to the integrated framework

Electromagnetic module (EMM)

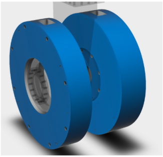




Software: SUPERFISH

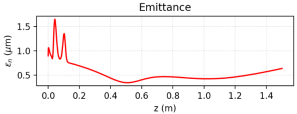
+

Magnetostatic module (MSM)



Software/Tool: POISSON, or parameterized on-axis Bz curve

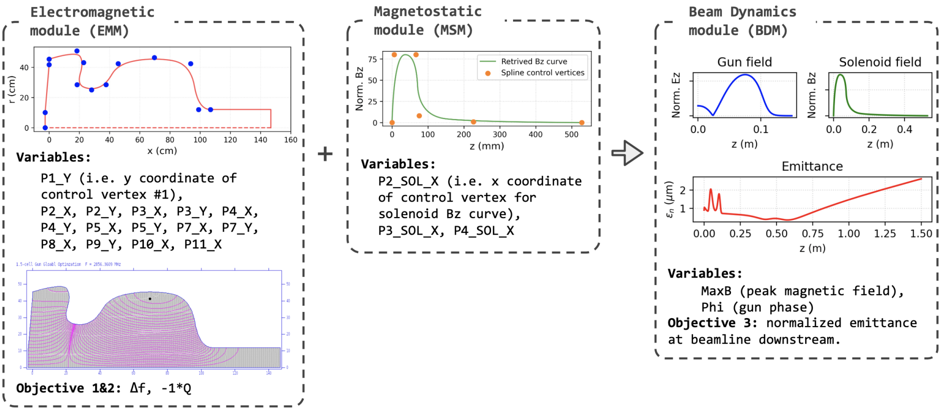
Beam Dynamics module (BDM)



Software: Astra

Workflow: the simulated electromagnetic field of the optimized gun from EMM and the parameterized on-axis B_z from MSM will be used in the BDM for beam dynamics simulations.

Diagram of Modules



Variables in EMM are coordinates of control vertices to generate the spline (aka. the gun geometry). **Variables in MMM** are coordinates of control vertices to generate the on-axis Bz. **Variables in BDM** are gun phase and solenoid peak Bz.

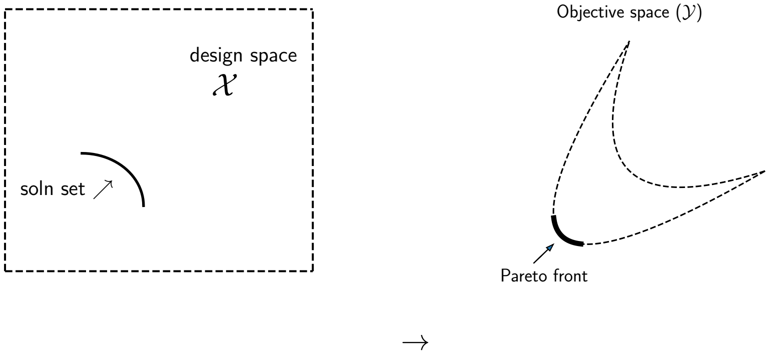
The Multiobjective Optimization Problem

min(ε_{trans} from BDM, -1$\cdot$ Q factor and Δf (defined by $|f_{target} - f_{simulate}|$) from EMM)

for all MSM shapes, EMM geometries, and BDM settings

s.t. few particles lost, beam quality is sufficient, etc.

The Multiobjective Optimization Problem



Optimization Challenges

- 23 design variables (15 control vertices from EMM + 6 from MSM + 2 from BDM)
- Multiple objectives (ε_{trans}, -1$\cdot$ Q factor and Δf)
- For almost all settings, problem is infeasible (particles lost, geometry is not physical, beam quality is poor)
- Running all simulations is expensive (10ish minutes on HPC)
- No gradients (derivative-free)

Optimization Challenges

- 23 design variables (15 control vertices from EMM + 6 from MSM + 2 from BDM)
- Multiple objectives (ε_{trans}, -1$\cdot$ Q factor and Δf)
- For almost all settings, problem is infeasible (particles lost, geometry is not physical, beam quality is poor)
- Running all simulations is expensive (10ish minutes on HPC)
- No gradients (derivative-free)

How to tune 23 variables, 3 objectives on limited budget, when almost all settings are infeasible?

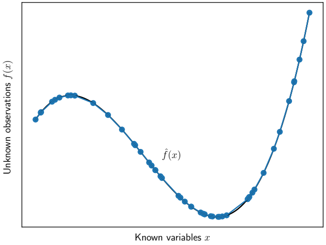
Popular Optimization Methodologies

- Multiobjective Evolutionary/Genetic Algorithms (NSGA-II)
- Multiobjective Bayesian Optimization

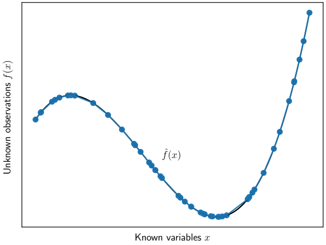
Popular Optimization Methodologies

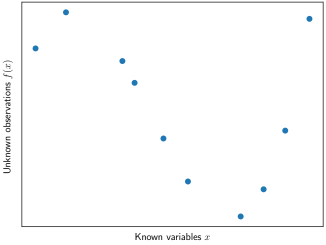
- Multiobjective Evolutionary/Genetic Algorithms (NSGA-II)
- Multiobjective Bayesian Optimization
- Multiobjective Trust-region descent?

Scalability of optimization methods

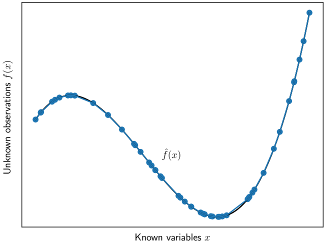


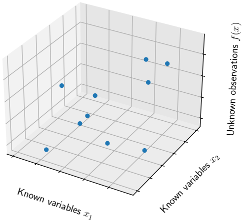
Scalability of optimization methods

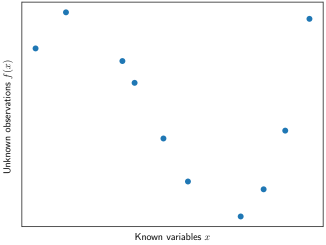




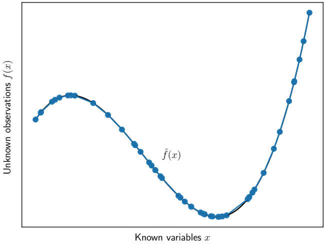
Scalability of optimization methods

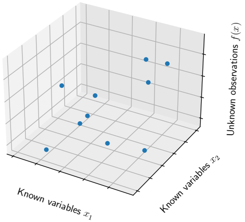


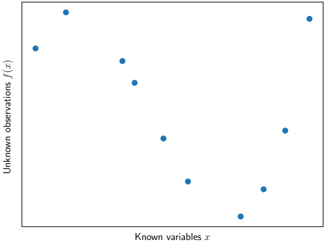


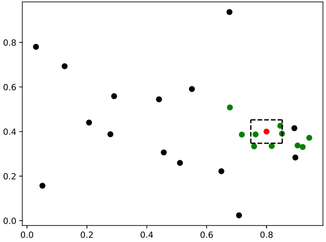


Scalability of optimization methods









Comparison on a test problem

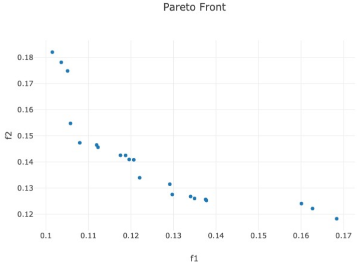
A 50D, 2 objective, convex test problem, 1000 function evals:

Gaussian process surrogate, comparing BO vs. Latin hypercube + trust-region

Comparison on a test problem

A 50D, 2 objective, convex test problem, 1000 function evals:

Gaussian process surrogate, comparing BO vs. Latin hypercube + trust-region

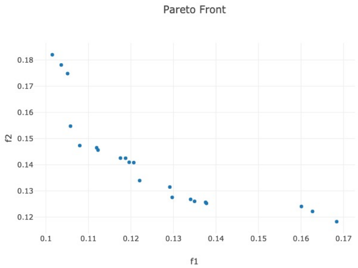


BO

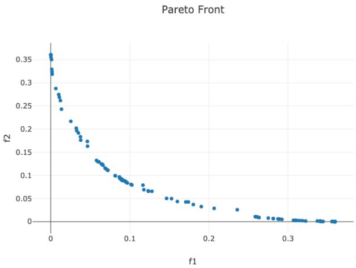
Comparison on a test problem

A 50D, 2 objective, convex test problem, 1000 function evals:

Gaussian process surrogate, comparing BO vs. Latin hypercube + trust-region



BO



TR

Software Stack

- EMM + MSM modules: `POISSON/SUPERFISH` – Fortran SW from
- BDM module: `ASTRA` – Fortran SW by K. Floetmann (DESY)
- Docker image: `github.com/hhslepicka/docker-poisson-superfish-nobin`
- Python interface: `github.com/ChristopherMyers/PySuperfish`

Software Stack

- EMM + MSM modules: `POISSON/SUPERFISH` – Fortran SW from
- BDM module: `ASTRA` – Fortran SW by K. Floetmann (DESY)
- Docker image: `github.com/hhslepicka/docker-poisson-superfish-nobin`
- Python interface: `github.com/ChristopherMyers/PySuperfish`
- Optimization SW: `ParMOO` – Chang & Wild (ANL)

Software Stack

- EMM + MSM modules: `POISSON/SUPERFISH` – Fortran SW from
- BDM module: `ASTRA` – Fortran SW by K. Floetmann (DESY)
- Docker image: `github.com/hhslepicka/docker-poisson-superfish-nobin`
- Python interface: `github.com/ChristopherMyers/PySuperfish`
- Optimization SW: `ParMOO` – Chang & Wild (ANL)
- Coordination and workflow on HPC systems: `libEnsemble` – Hudson, Larson, Navarro & Wild (ANL)

Software Stack

- EMM + MSM modules: `POISSON/SUPERFISH` – Fortran SW from
- BDM module: `ASTRA` – Fortran SW by K. Floetmann (DESY)
- Docker image: `github.com/hhslepicka/docker-poisson-superfish-nobin`
- Python interface: `github.com/ChristopherMyers/PySuperfish`
- Optimization SW: `ParMOO` – Chang & Wild (ANL)
- Coordination and workflow on HPC systems: `libEnsemble` – Hudson, Larson, Navarro & Wild (ANL)
- Ran all experiments on the HPC `Bebop` in the LCRC at Argonne

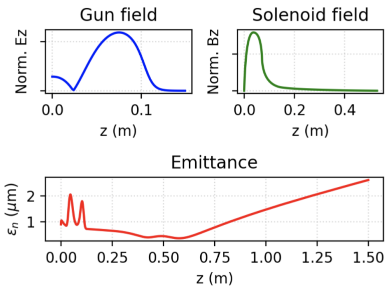
Experiment setup

ParMOO hyperparameters:

- **search budget**: 800 (all infeasible)
- **search**: LatinHypercube,
- **surrogate**: LocalGaussRBF
- **batch size**: 16, where we set 15 randomized scalarizations, and fixed 1 scalarization for emittance optimization.
- **total budget**: 5600 over all 300 iterations of localized Gaussian process modeling and trust-region descent.

Results of optimization

- Given a initial beam source with rms spot size (σ_x) of 0.5 mm, bunch length (FWHM) of 300 fs, bunch charge of 100 pC, along with a standard operation gun gradient of 150 MV/m.
- The emittance generated by the optimized beamline was found to converge to <0.3 μm, which is comparable to state-of-the-art results. An example of the optimized result is shown here.
- Solution was found within the first 2000 function evaluations.



Conclusions and future work

Conclusion of this Study:

- Our proposed platform by integrating multi-physics modules shows significant efficiency in the design of rf structures.
- With the unified platform, a new Linac cavity has been proposed and designed.
- Our optimization methods are able to tune these complex geometries for multiple obejctives on a realistic budget.

Conclusions and future work

Conclusion of this Study:

- Our proposed platform by integrating multi-physics modules shows significant efficiency in the design of rf structures.
- With the unified platform, a new Linac cavity has been proposed and designed.
- Our optimization methods are able to tune these complex geometries for multiple obejctives on a realistic budget.

Next Steps:

- Perform more rigorous experiments and comparisons with other optimization methods and previous works.
- Explore the tuning of other aspects of geometry such as cell lengths and additional concave features.

Acknowledgments

This work is supported by Laboratory Directed Research and Development (LDRD) funding from Argonne National Laboratory.

This work was also supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research and Office of High-Energy Physics, Scientific Discovery through Advanced Computing (SciDAC) Program through the FASTMath Institute and the CAMPA Project under Contract No. DE-AC02-06CH11357.

We gratefully acknowledge the computing resources provided on Bebop, a HPC cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory.

We thank Philippe Piot for the insightful discussions on the simulations. We also thank Jeff Larson, John-Luke Navarro, and Steve Hudson for their advice on libEnsemble usage.