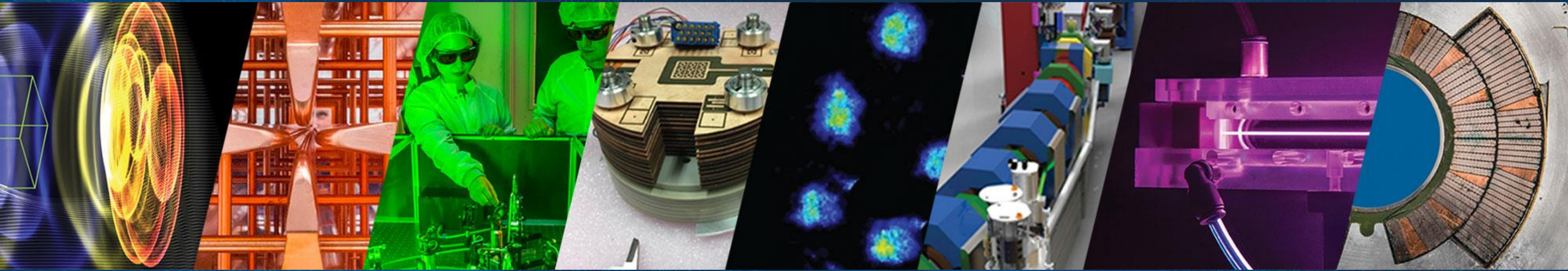
AI/ML for Accelerator Design/Control at LBNL

Remi Lehe
Lawrence Berkeley National Laboratory



June 13, 2024

BERKELEY LAB
ACCELERATOR TECHNOLOGY & APPLIED PHYSICS DIVISION ATAP
U.S. DEPARTMENT OF ENERGY | Office of Science

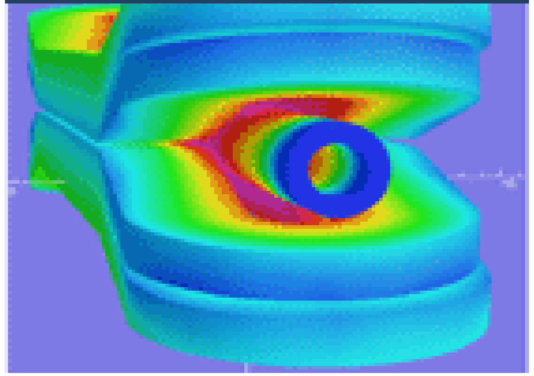
AI/ML for accelerators is used across several groups in the ATAP division

ATAP
ACCELERATOR TECHNOLOGY &
APPLIED PHYSICS DIVISION

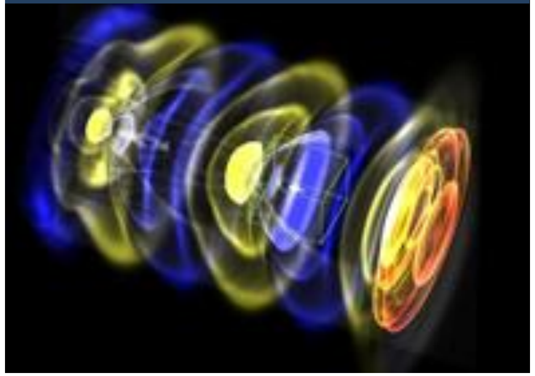
Advanced Light Source Accelerator Physics

Berkeley Lab Laser Accelerator Center

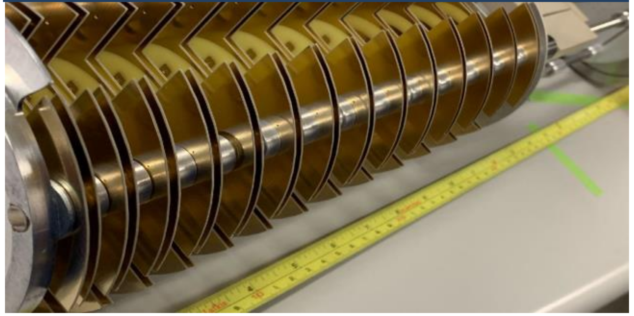
Superconducting Magnet Program



Advanced Modeling Program



Fusion Science and Ion Beam Technology Program



Berkeley Accelerator Controls & Instrumentation (BACI)

Outline

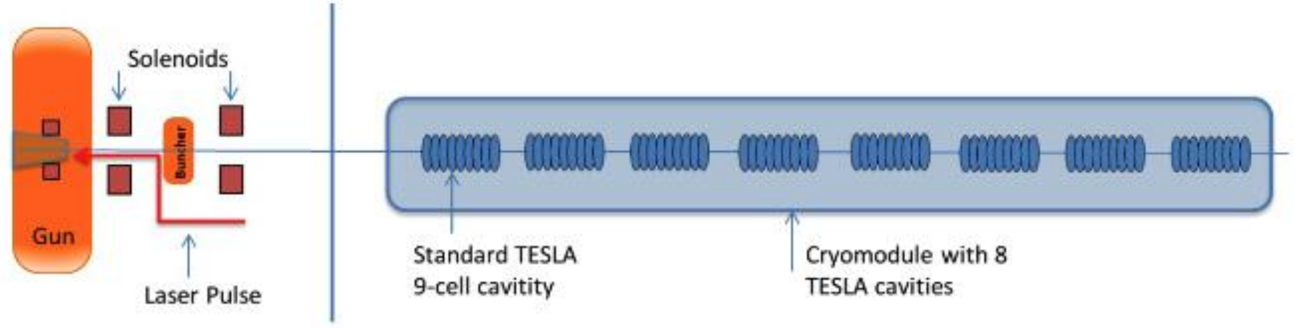
- AI/ML for design optimization of accelerators
- AI/ML for accelerator operation
- Adapting simulation tools for integration with AI/ML

We use ML to accelerate simulation-based design of particle accelerators.

- Designing accelerators often involve **simultaneously tuning** many parameters (focusing, accelerating cavities, etc.) to reach the design with **optimal performance,**

  i.e. maximize $f(\boldsymbol{x})$

  $\boldsymbol{x}$: vector of accelerator parameters
  f: function to maximize ("objective function")



$$f = \epsilon_{\perp}$$

$$\boldsymbol{x} = \begin{pmatrix} B_{solenoid} \\ E_{cavity} \end{pmatrix}$$

- Each combination of parameters needs to be evaluated with **expensive simulations.**

We use ML to accelerate simulation-based design of particle accelerators.

“Conventional” optimization algorithms:

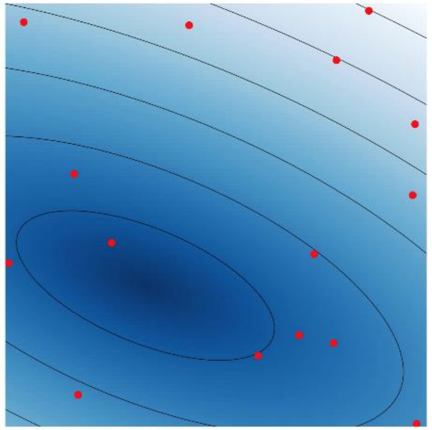
e.g.
- Gradient descent
- Genetic algorithms
- Nelder-Mead algorithm (a.k.a. simplex)
- …

The next evaluations are based on simple rules that the depend on the **last few evaluations**.

Typically require **many** evaluations of *f*.

Optimization algorithms based on machine learning:

Progressively learn a **global model** of the objective function $f(\boldsymbol{x})$ over the parameter space.
Use this model to **only evaluate** the most promising $\boldsymbol{x}$.



Model of f

Typically require **fewer** evaluations of *f*.

A. Ferran Pousa et al., Bayesian optimization of laser-plasma accelerators assisted by reduced physical models, PRAB (2023)
Y. Lu et al., Demonstration of machine learning-enhanced multi-objective optimization of ultrahigh-brightness lattices for 4th-generation synchrotron light sources, NIMA (2023)

We are fostering interoperability across open-source optimization software.

- Several **open-source optimization frameworks** are being used in the accelerator community (each with their respective strengths)

Xopt
github.com/ChristopherMayes/Xopt
Optimization algorithms

optimas
github.com/optimas-org/optimas
Optimization algorithms

rsopt
github.com/radiasoft/rsopt
Optimization algorithms

powered by **libensemble**
github.com/Libensemble/libensemble

- Ongoing efforts by the developers to **standardize optimizers** and **foster interoperability.**

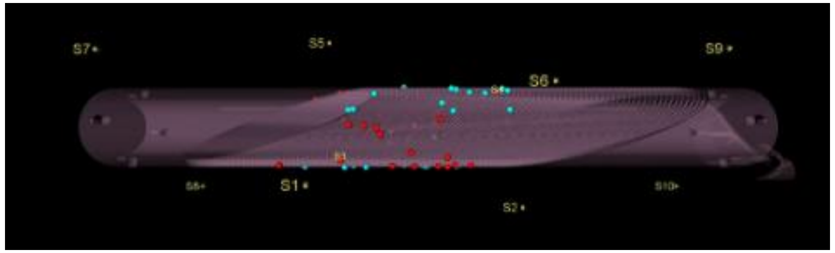
Outline

- AI/ML for design optimization of accelerators
- AI/ML for accelerator operation
- Adapting simulation tools for integration with AI/ML

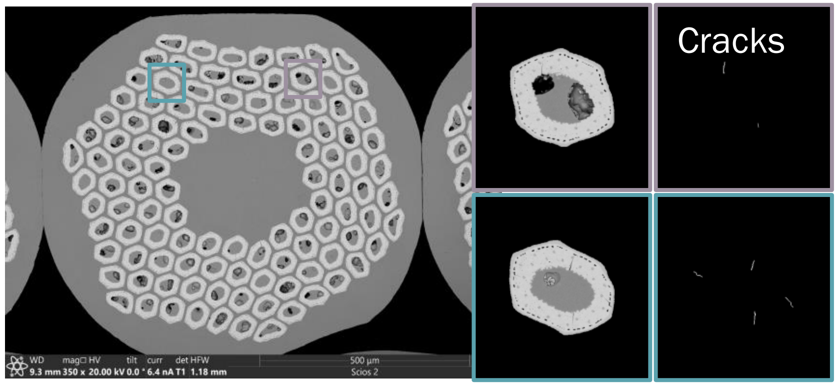
AI/ML for accelerator operations covers a broad range of topics.

Superconducting magnets verification/protection:

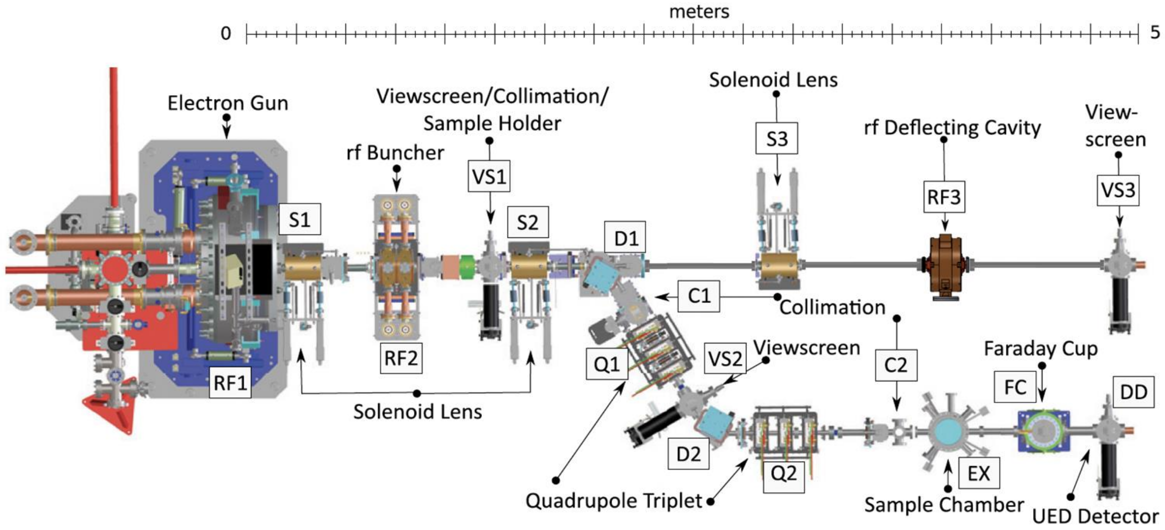
- Detection/classification of **quench precursors** from acoustic emission.
  M. Marchevsky, arXiv:2203.08871 (2022)



- Check for **cracks/defects** in superconducting cables with automated image analysis



Control and autotuning at compact LBNL accelerators (e.g., HiRES & NDCX-II):



A. Scheinker, et. al., Scientific Reports (2021)
A. Scheinker, et. al, Phys. Rev. E 107 (2023)
F. Cropp, et. al., Phys. Rev. Accel. Beams 26 (2023)

ML is used in operation at the Advanced Light Source, to stabilize beam size.

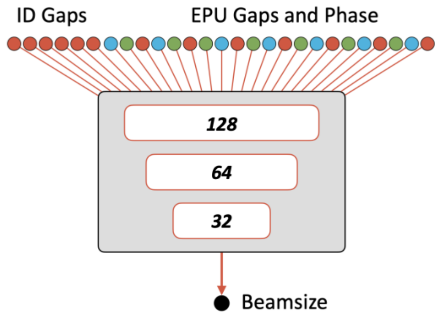
- The ALS is a storage ring light source.

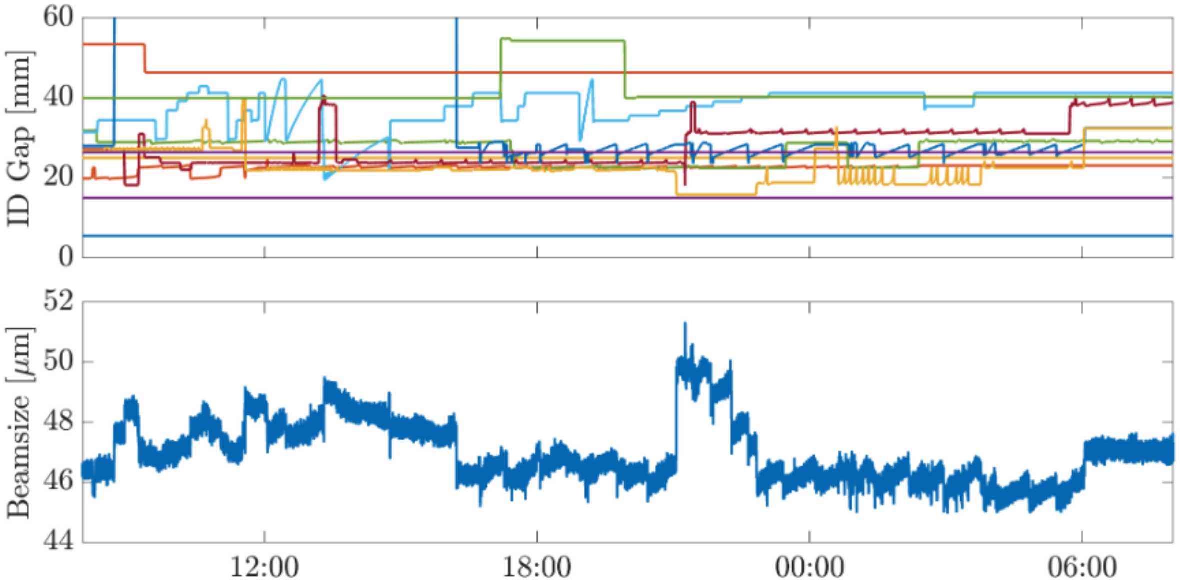
- The parameters of **insertion devices** (e.g. undulators) are frequently changing, due to changing modes of operation.

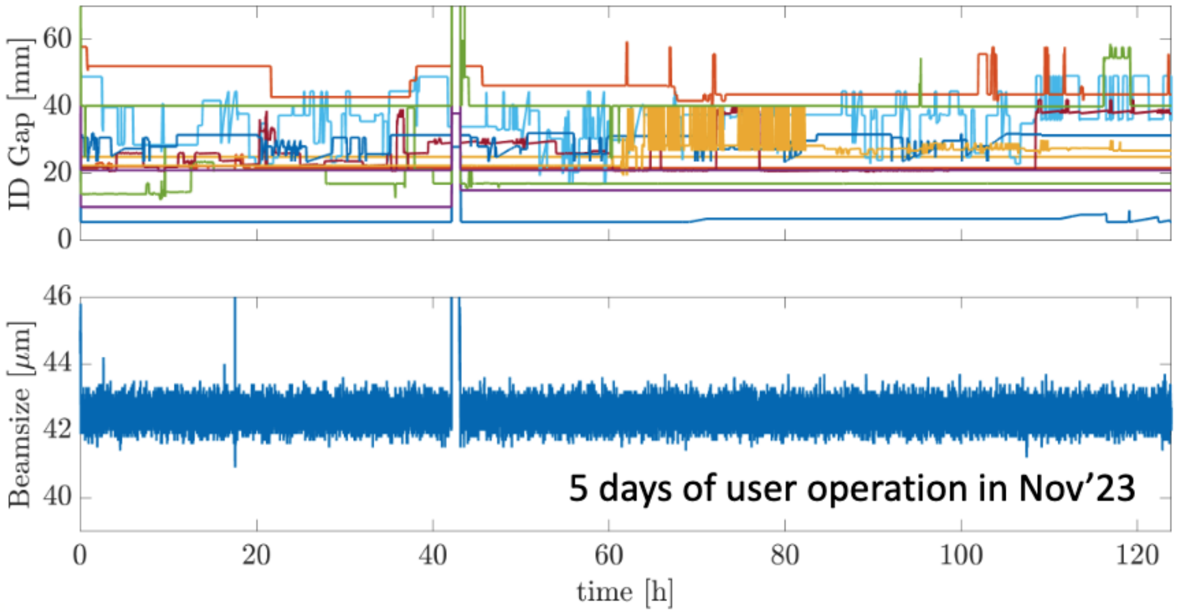
- These changes affect the **beam size**.

- The ALS now uses a neural network to **predict** changes in beam size and **correct** them.
  Leeman et al., PRL 123, 194801 (2019)
  Hellert et al., accepted in PRAB (2024)

- Used in routine operation







ML is used in operation at the Advanced Light Source, to stabilize beam size.

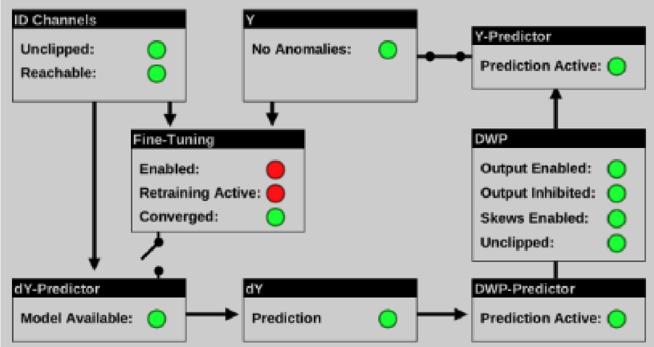
- The machine configuration **drifts** over time.
  To always **remain accurate**, the neural network needs to be periodically **retrained**.

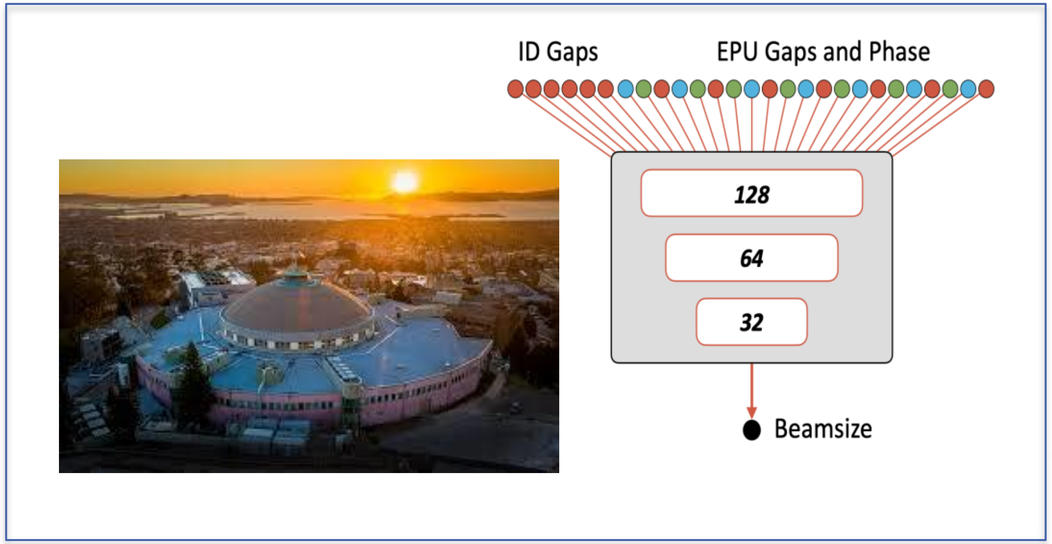
- The ALS team developed an extensive framework to automatically **monitor, retrain, archive and deploy** the neural network.

CS-Studio control system studio Phoebus

PyTorch

EPICS





Online model

Collect data from current user run

Fine-tune NN model (retraining)

~ seconds

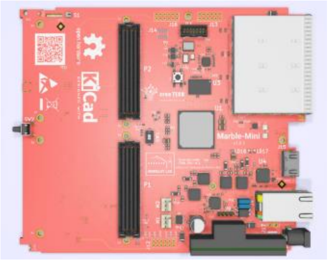
Replace NN model

ALS ADVANCED LIGHT SOURCE

The BACI group is developing tools to run AI/ML on FPGA.

- Some accelerator control applications require **very low latency**, necessitating FPGA hardware.

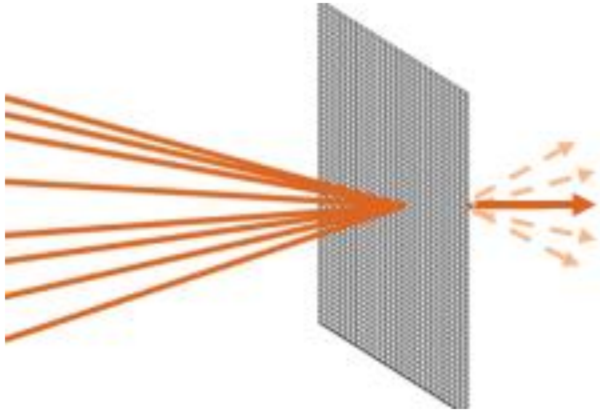
- The **Marble-Mini** is an FPGA-based board that is widely used in low-level RF control for accelerators.



github.com/BerkeleyLab/Marble-Mini

- The BACI group at LBNL is developing a framework to run neutral networks on Marble-Mini.
  L.Doolittle, Q. Du and D. Wang, Software Disclosure 2024-008: Generic Multi-Layer Perceptron Inference Accelerator on FPGA (vneuron) v1.0
  \+ LBNL LDRD led by Dan Wang

- **Example use case:** neural network that controls the phases of 9 coherently combined laser beams



- 3-layer neural network with ~1600 weights
  Inference time: 131 cycles (1046 ns)
  Uses only 16% of the chip resources

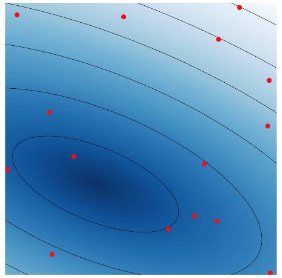
We are working towards digital twins to guide accelerator tuning in real time.

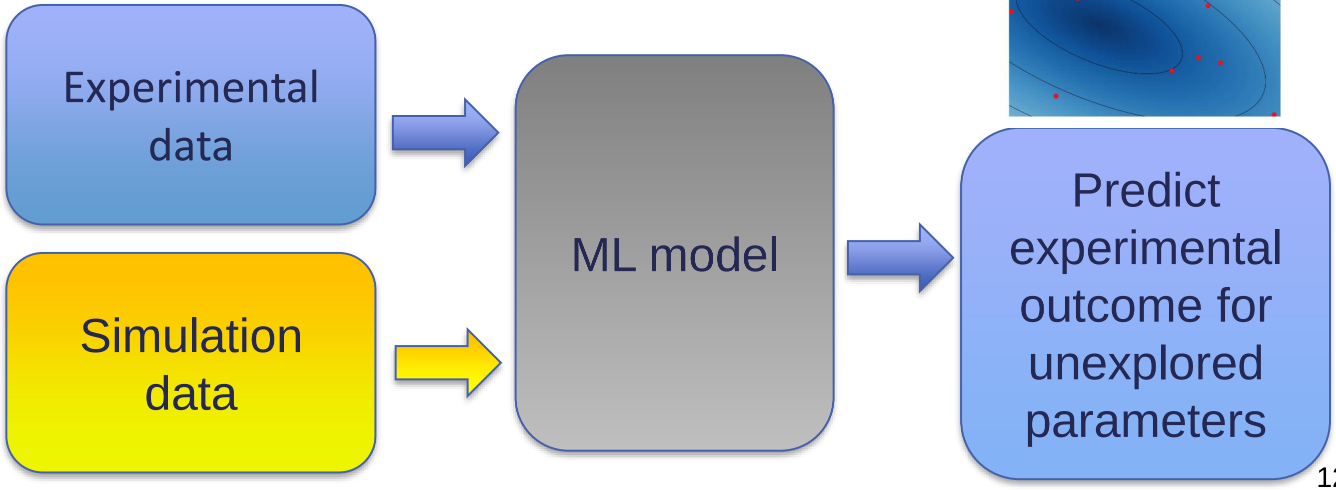
- **Real-time accelerator tuning** requires to simultaneously adjust multiple parameters (e.g. focusing elements, accelerator cavities) to achieve **optimal operation**.

- Corresponds to optimization in high-dimensional space, **time-consuming** if the number of parameters is large.

- Leveraging information from **numerical simulations** can significantly speed-up real-time tuning.

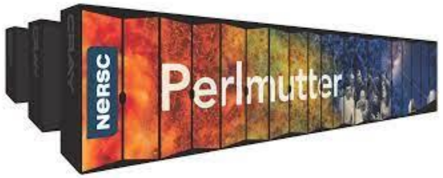
  e.g. Hanuka et al., Physics model-informed Gaussian process for online optimization of particle accelerators (2021)





We are working towards digital twins to guide accelerator tuning in real time.

Framework for deployment (”Superfacility”)



Experimental facility:

- Real-time tuning
- Data collection

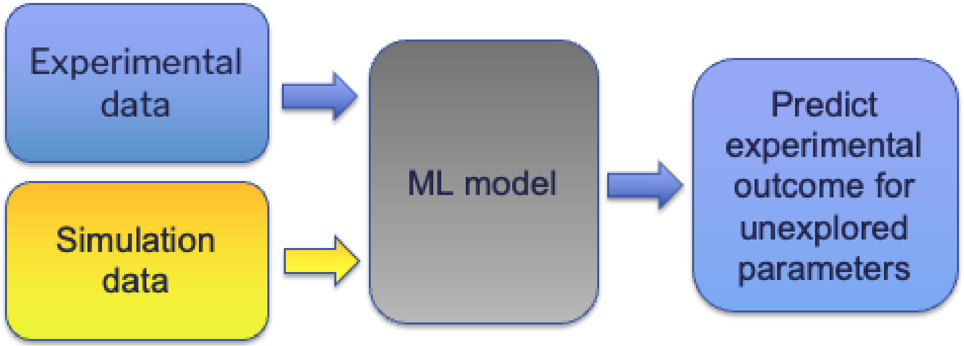
Supercomputer:

- Updates ML model
- Launch new simulations

- Ongoing collaboration with SLAC to develop corresponding software framework. (funding: LDRD + NESAP program)

- Will leverage existing software for parts of this workflow e.g., github.com/slaclab/lume-services

ML challenges



- Experimental and simulation data **oftentimes don’t match exactly**. The ML model needs to **handle them differently**.

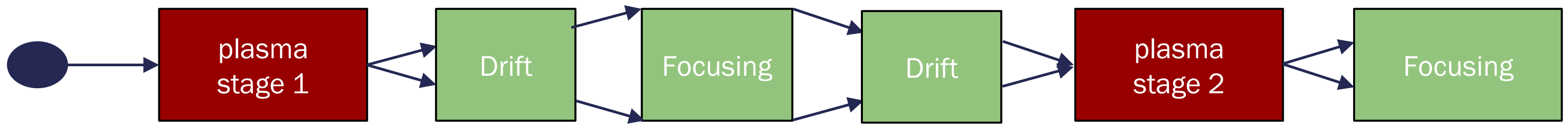
- Several possible techniques:
  - Multi-fidelity Gaussian process *A. Ferran Pousa, PRAB (2023)*
  - Calibration of neural networks *T. Boltz, arxiv 2403.0322 (2024)*

Outline

- AI/ML for design optimization of accelerators
- AI/ML for accelerator operation
- Adapting simulation tools for integration with AI/ML

ML surrogate models can speed-up simulations.

- In a given lattice, some elements can be more **computationally-expensive** to model.
  Extreme example: laser-plasma acceleration stages

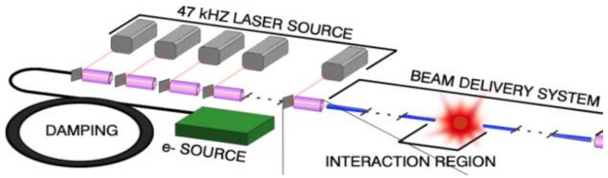


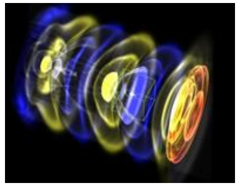
few pC
e- beam

WarpX
(full EM-PIC)

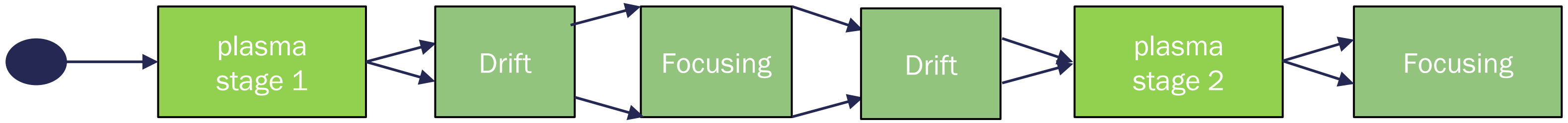
ImpactX
(tracking code)

WarpX
(full EM-PIC)





- <u>Under certain conditions</u> (here: negligible collective effects, specific range of parameters), computationally-expensive elements can be replaced by **ML models**, trained over **past simulations.**



WarpX-trained
neural network
PyTorch

ImpactX
(tracking code)

WarpX-trained
neural network
PyTorch

Simulation time: (with full geometry/physics)

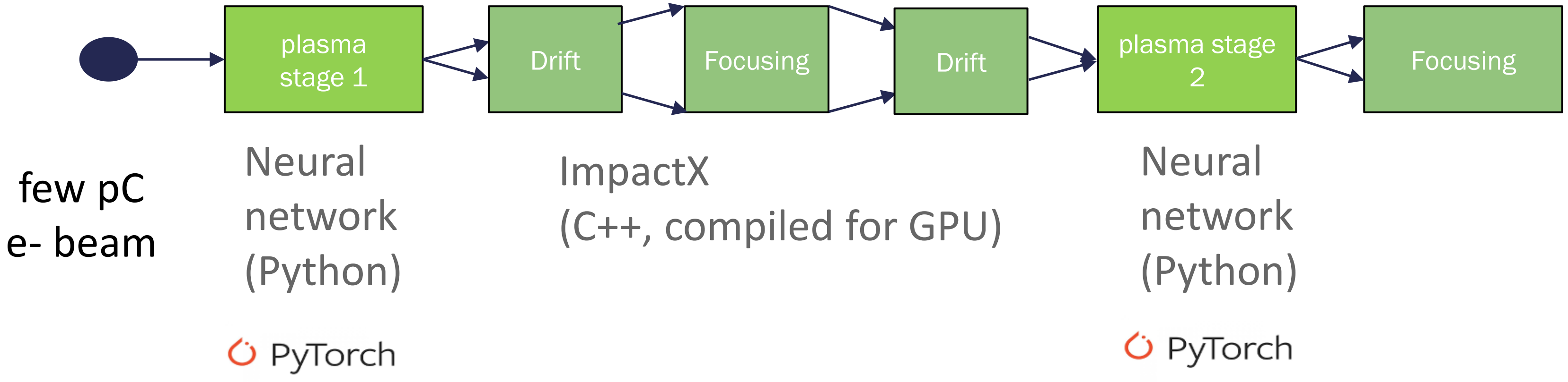
hrs	<sec
on several GPUs	on 1 GPU

LDRD
BERKELEY LAB
SciDAC Scientific Discovery through Advanced Computing
DOE GARD

RT Sandberg et al., IPAC23, DOI:10.18429/JACoW-IPAC2023-WEPA101 (2023)
*RT Sandberg et al., **PASC24 Best Paper** (2024)*

We adapt our accelerator simulation codes for seamless integration with ML surrogates.

- This workflow requires to pass **beam data** across elements (position/momentum for each particle)



- Our C++ simulation codes expose beam data (in GPU memory) through a Python interface.
  → Seamless, **GPU-Accelerated Coupling** of **accelerator simulation & ML Frameworks.**

github.com/AMReX-Codes/pyamrex
github.com/ECP-WarpX/WarpX
github.com/ECP-WarpX/ImpactX

(based on pybind11 and python standards for GPU arrays)

LDRD

DOE GARD

A Huebl et al. *pyAMReX: GPU-Enabled, Zero-Copy AMReX Python Bindings including AI/ML. software*, 2023. DOI:10.5281/zenodo.8408733 github.com/AMReX-Codes/pyamrex ;
A. Myers et al., AMReX and pyAMReX: Looking Beyond ECP. arXiv:2403.12179

Even tighter ML integration can be achieved with differentiable simulations

Regular simulation code

Input

Output

$\boldsymbol{X} \longrightarrow$ Simulation code $\longrightarrow f(\boldsymbol{X})$

Differentiable simulation code

Input

Output

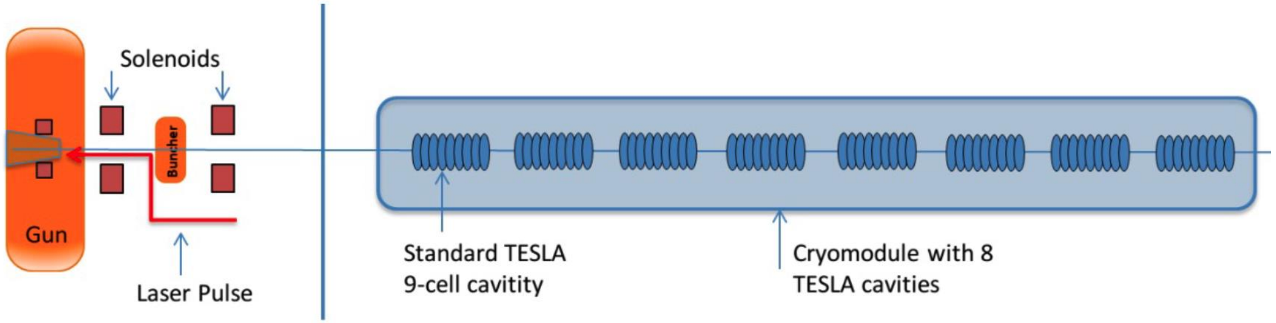
$\boldsymbol{X} \longrightarrow$ Simulation code $\longrightarrow f(\boldsymbol{X}),\ \frac{\partial f}{\partial \boldsymbol{X}}(\boldsymbol{X})$

Example:

Input:
accelerator parameters

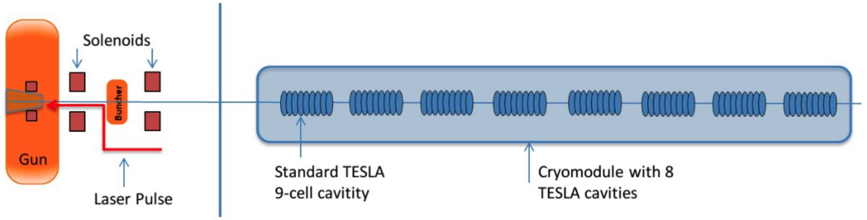
$$f = \epsilon_\perp$$

$$\boldsymbol{X} = \begin{pmatrix} B_{solenoid} \\ \varphi_{RF\ cavity} \\ E_{RF\ cavity} \\ \sigma_{beam,i} \end{pmatrix}$$



$$\frac{\partial f}{\partial \boldsymbol{X}} = \begin{pmatrix} \frac{\partial \epsilon_\perp}{\partial B_{solenoid}} \\ \frac{\partial \epsilon_\perp}{\partial \varphi_{RF\ cavity}} \\ \frac{\partial \epsilon_\perp}{\partial E_{RF\ cavity}} \\ \frac{\partial \epsilon_\perp}{\partial \sigma_{beam,i}} \end{pmatrix}$$

Differentiable codes have several advantages.

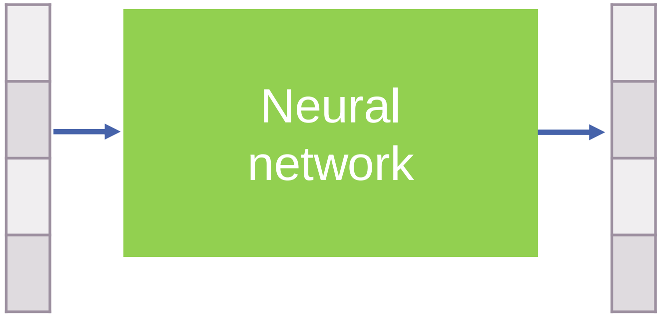


- **Sensitivity studies**
  $\frac{\partial f}{\partial x}$ quantifies how sensitive the output is to the input.

- **Optimization in high-dimensional space** (e.g. of accelerator designs)
  $\frac{\partial f}{\partial x}$ can be used in **gradient-based** optimizers, which often converge faster

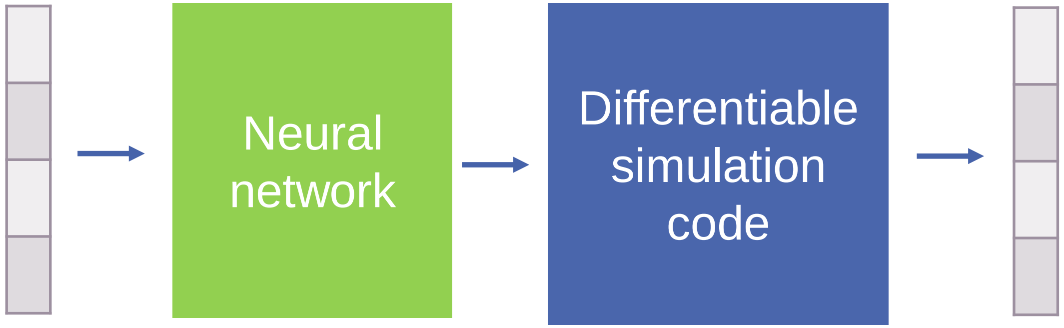
- **Allows <u>training</u> of a neural network that is <u>combined</u> with a differentiable code**

Traditional training of neural network



Input/output pairs, from a data set

Training of neutral network <u>combined</u> with a code



Example: *R. Roussel et al., Phase Space Reconstruction from Accelerator Beam Measurements Using Neural Networks and Differentiable Simulations, PRL (2023)*

We are exploring frameworks for differentiable codes.

- Several **algorithms** are available to make a code differentiable. e.g.
  J. Qiang, Differentiable self-consistent space-charge simulation for accelerator design, PRAB (2023)

- Several efforts to build differentiable **accelerator simulation codes**,
  based on **auto-differentiation frameworks.**

  - `pytorch` PyTorch
    Cheetah: accelerator code based on `pytorch`
    github.com/desy-ml/cheetah

    Karlsruher Institut für Technologie · DESY · SLAC · BERKELEY LAB

  - `Julia` julia
    Bmad-Julia: proposal to implement Bmad algorithms in Julia
    github.com/bmad-sim

    Brookhaven National Laboratory · BERKELEY LAB · radiasoft · SLAC · Jefferson Lab

  - Enzyme AD
    Takes existing C++ code and makes it auto-differentiable at compile time.
    Could be leveraged to make BLAST codes (ImpactX, WarpX, …) differentiable.

Conclusion

- AI/ML is used in **many accelerator-relevant applications**, in the ATAP division.
- In the process, we are building **software frameworks** for robust deployment of AI/ML.
- Many of these frameworks are **open-source** and can be leveraged at **other accelerator facilities**. If interested, feel free to reach out!

USPAS course on AI/ML for accelerators

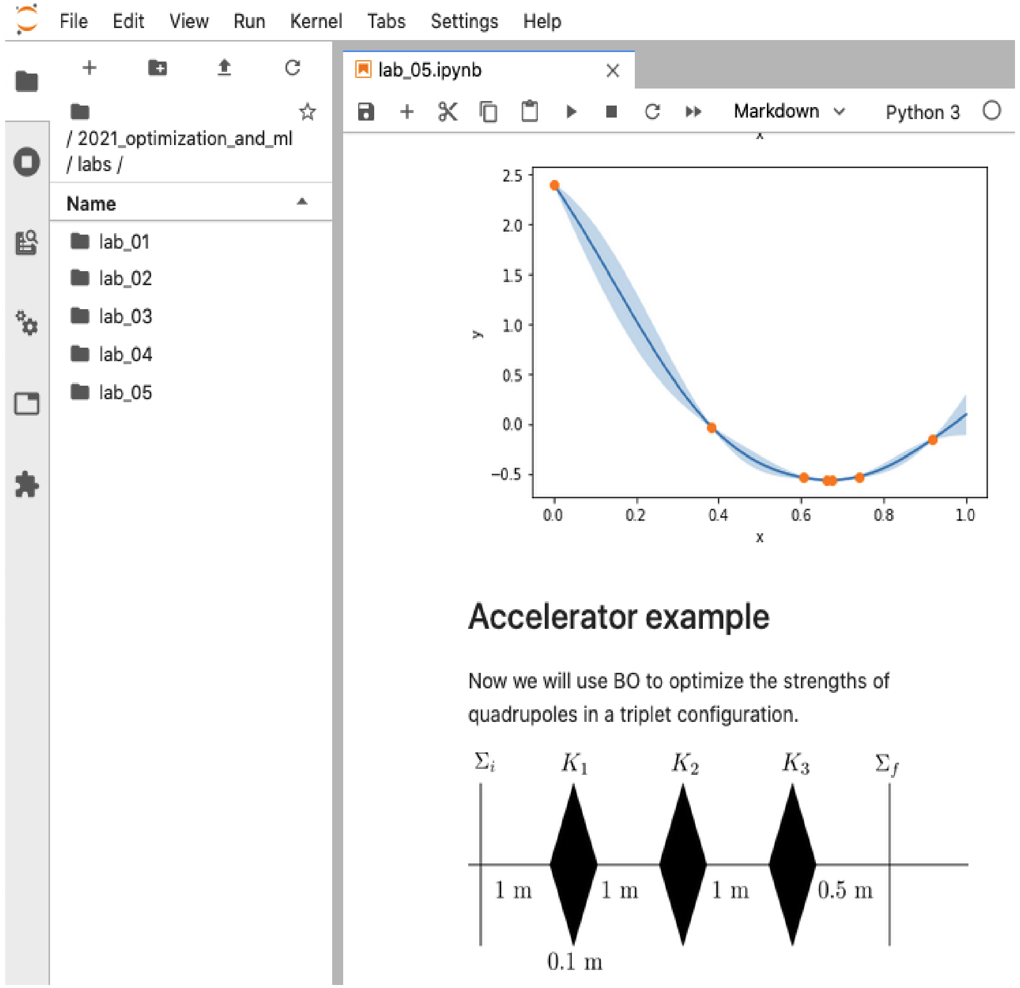
Course on **Optimization and Machine Learning for Accelerators** at the **U.S. Particle Accelerator School**, since 2021

Some of the Instructors:

Remi Lehe (LBNL)

Auralee Edelen (SLAC)

Ryan Roussel (SLAC)



Next USPAS session:

Held in: Knoxville, Tennessee

Dates: Jan 27 - Feb 7, 2025

May include optimization and ML (not confirmed yet)

Thank you