NAPAC'22, August 9, 2022

Artificial Intelligence and Machine Learning for Particle Accelerators

Auralee Edelen

edelen@slac.stanford.edu

leelinska

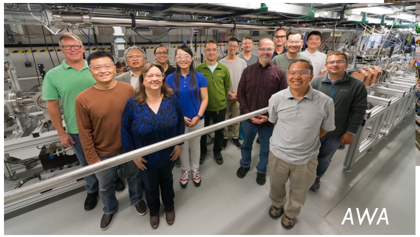
with work/examples also from many colleagues, especially: R. Roussel, C. Mayes, C. Emma, S. Miskovich, D. Ratner, J. Duris, A. Hanuka, A. Scheinker, N. Neveu, L. Gupta, A. Adelmann, Y. Huber, M. Frey, E. Cropp, P. Musumeci, A. Mishra

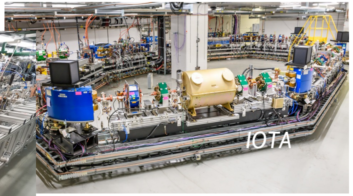
AI/ML is well-suited for cross-cutting applications → algorithm transfer between accelerator facilities is possible

Large User Facilities

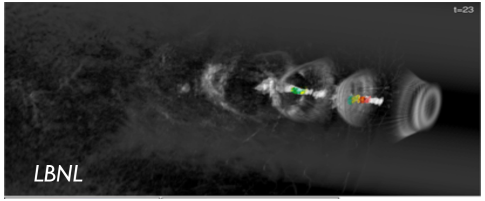
Industrial / Medical

Small Test Facilities



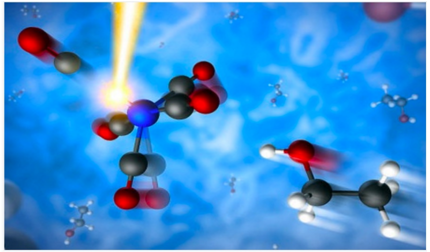


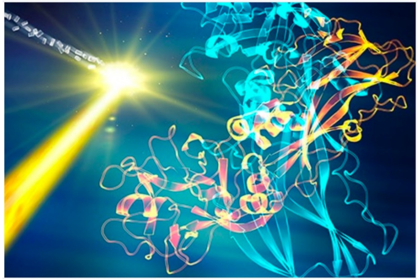
Novel Acceleration Schemes

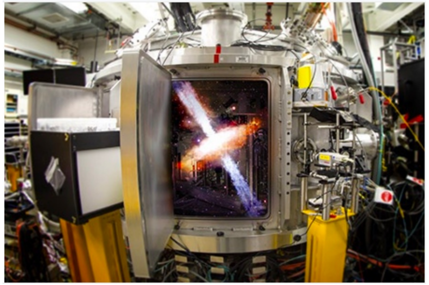


Different specific needs, but **many similar challenges** in online modeling, machine understanding, and control

Accelerators have unique challenges/characteristic → *can also contribute to AI/ML research*





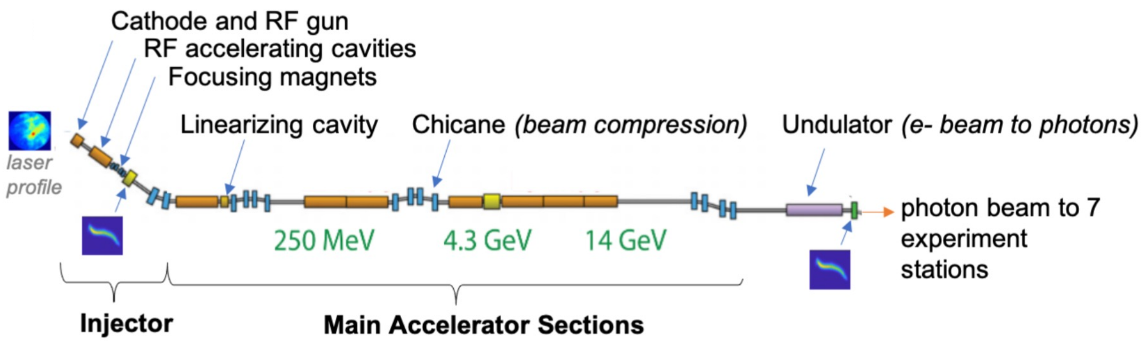


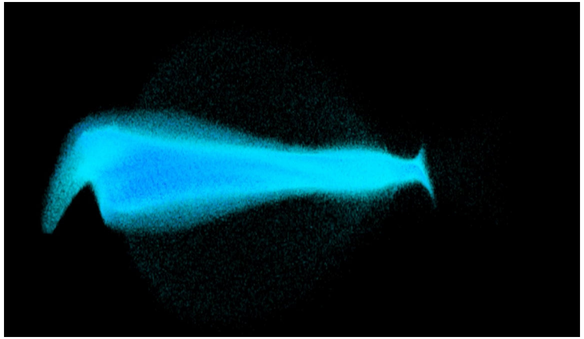
1,062 experiments in 2016

~1023 papers since 2009

Experimenters come for a few days – a week

beam duration, x-ray wavelength etc. adjusted for each experiment



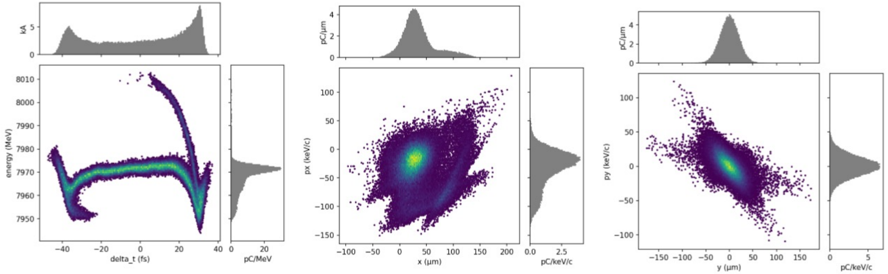


Beam exists in 6-D position-momentum phase space

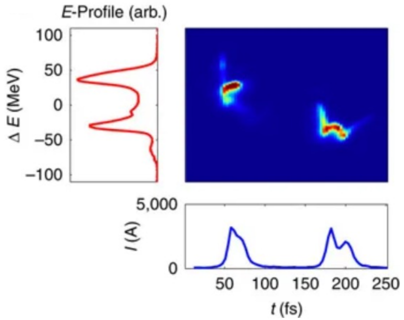
Have incomplete information: measure 2-D projections or reconstruct based on perturbations of upstream controls

Can have dozens-to-hundreds of controllable variables and hundreds-of-thousands to millions to monitor

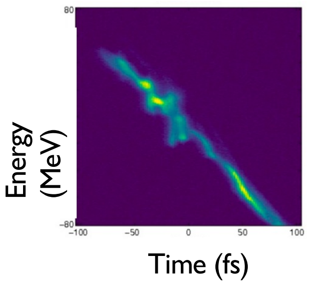
Nonlinear, high-dimensional optimization problem

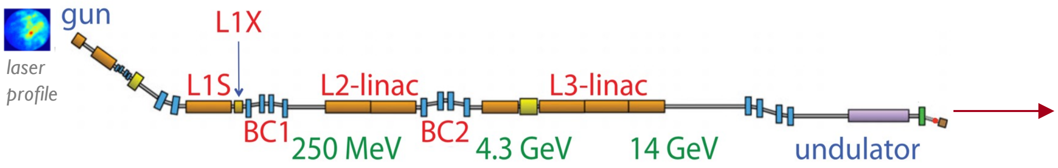


A. Marinelli, et al., Nat. Commun. **6**, 6369 (2015)



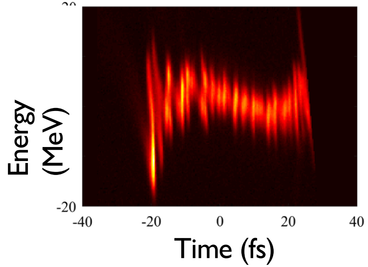
A. Marinelli, IPAC'18



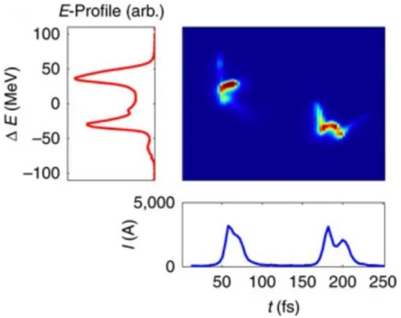


photon beam to
7 experiment
stations

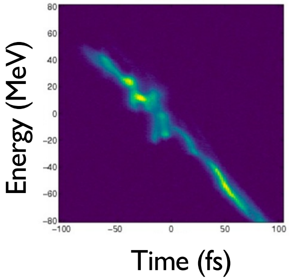
J. Qiang et al, PRAB (2017)



A. Marinelli, et al., Nat. Commun. **6**, 6369 (2015)



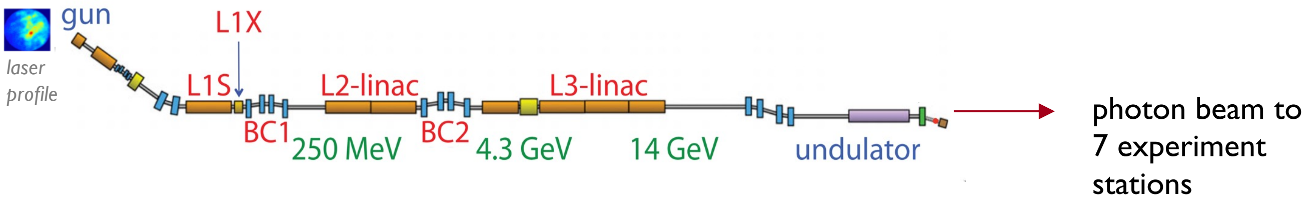
A. Marinelli, IPAC'18

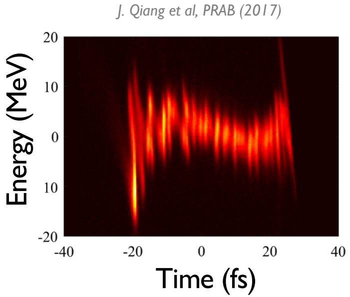


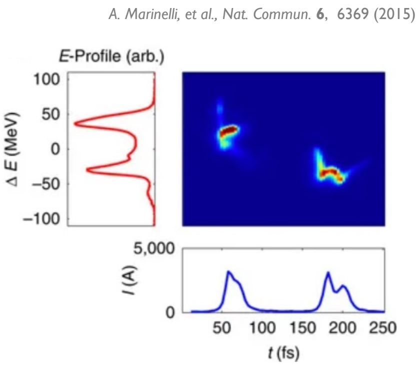
Approximate Annual Budget: $145 million
Approximate hours of experiment delivery per year: 5000
About $30k per experiment hour to run

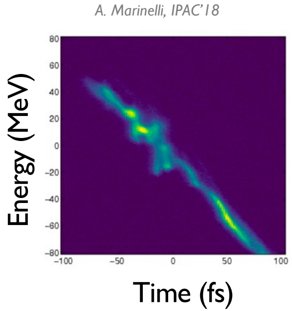
400 hours hand-tuning in a year → $12 million value
~10 additional experiments

wide spectrum of tuning needs at different accelerators







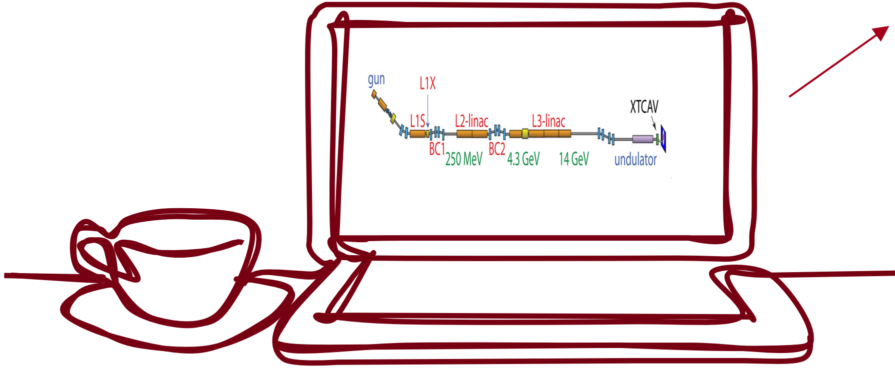


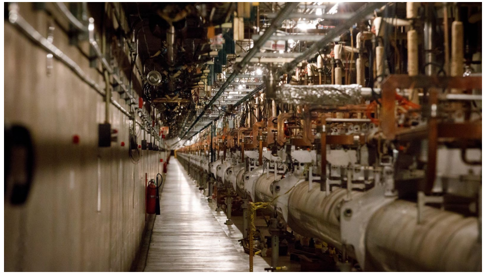
Rapid beam customization

Achieve new configurations + unprecedented beam parameters

Fine control to maintain stability within tolerances

In a perfect world…





Use a fast, accurate model …

find some knobs that give us the beam we want and apply those to the machine

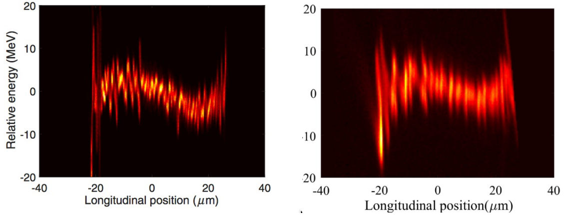
get info about unobserved parts of machine (online model / virtual diagnostic)

do offline planning and control algorithm prototyping

In reality things are much more difficult…

computationally expensive simulations

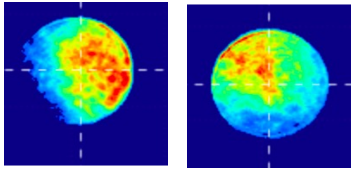
Simulation | Measurement

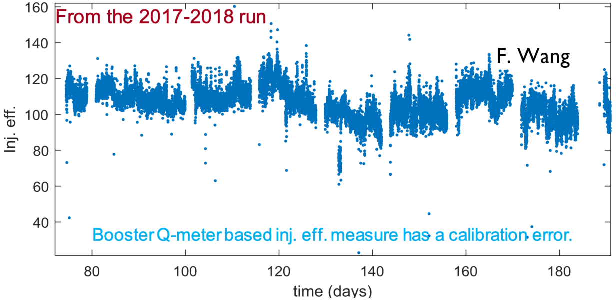


10 hours on thousands of cores at NERSC!

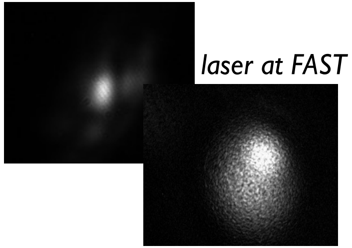
J. Qiang, et al., PRSTAB30, 054402, 2017

fluctuations/noise (e.g. laser spot)

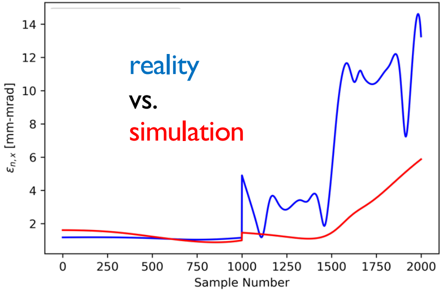




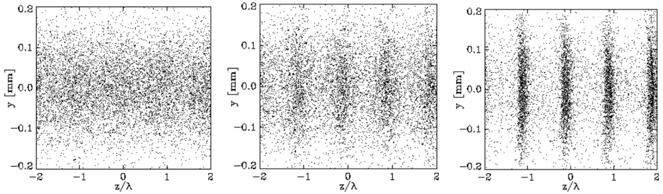
hidden variables / sensitivities



drift over time



many small, compounding sources of uncertainty



nonlinear effects / instabilities

AI/ML is well-positioned to help address these challenges

Tuning approaches can leverage different amounts of data/previous knowledge

less ← assumed knowledge of machine → more

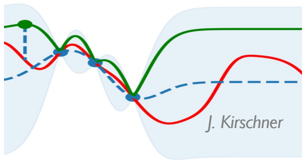
Model-Free Optimization

Observe performance change after a setting adjustment

→ estimate direction or apply heuristics toward improvement

gradient descent
simplex

Model-guided Optimization

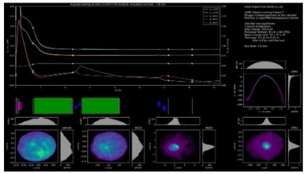


Update a model at each step

→ use model to help select the next point

Bayesian optimization
reinforcement learning

Global Modeling + Feed-forward Corrections



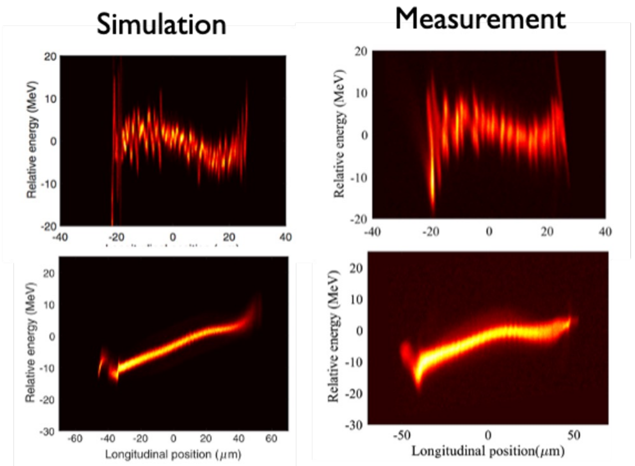
Make fast system model

→ provide initial guess (i.e. warm start) for settings or fast compensation

ML system models +
inverse models

Fast-Executing, Accurate System Models

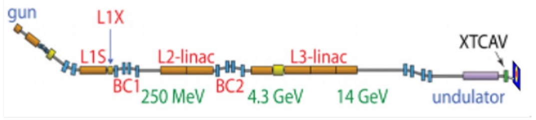
Accelerator simulations that include nonlinear and collective effects are powerful tools, but they can be computationally expensive



J. Qiang, et al., PRSTAB30, 054402, 2017

10 hours on thousands of cores at NERSC!

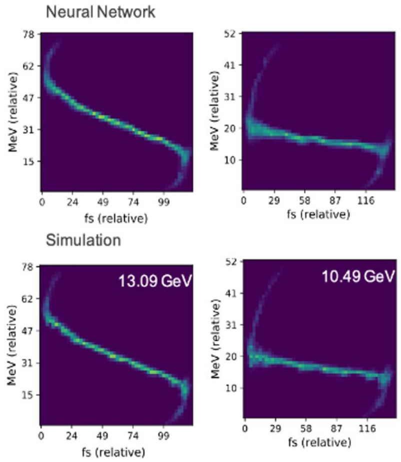
ML models can provide fast approximations to simulations ("surrogate models")



Linac sim in Bmad with collective beam effects

Scan of 6 settings in simulation

Variable	Min	Max	Nominal	Unit
L1 Phase	-40	-20	-25.1	deg
L2 Phase	-50	0	-41.4	deg
L3 Phase	-10	10	0	deg
L1 Voltage	50	110	100	percent
L2 Voltage	50	110	100	percent
L3 Voltage	50	110	100	percent



< ms execution speed

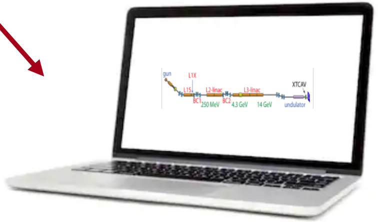
10^6 times speedup

ML modeling enables high-fidelity predictions of system responses with unprecedented speeds, opening up new avenues for high-fidelity online prediction, tracking of machine behavior, and model-based control

Fast-Executing, Accurate System Models

Bringing simulation tools from HPC systems to online/local compute

Online prediction
Model-based control



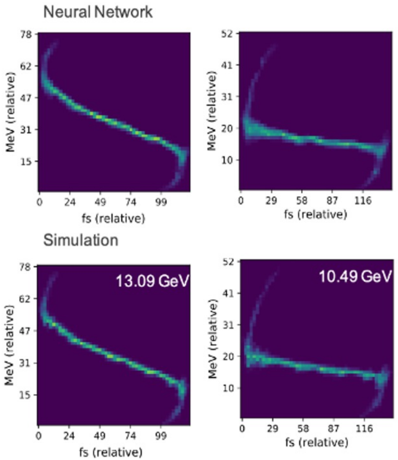
Control prototyping
Experiment planning

ML models can provide fast approximations to simulations (“surrogate models”)

Linac sim in Bmad with collective beam effects

Scan of 6 settings in simulation

Variable	Min	Max	Nominal	Unit
L1 Phase	-40	-20	-25.1	deg
L2 Phase	-50	0	-41.4	deg
L3 Phase	-10	10	0	deg
L1 Voltage	50	110	100	percent
L2 Voltage	50	110	100	percent
L3 Voltage	50	110	100	percent



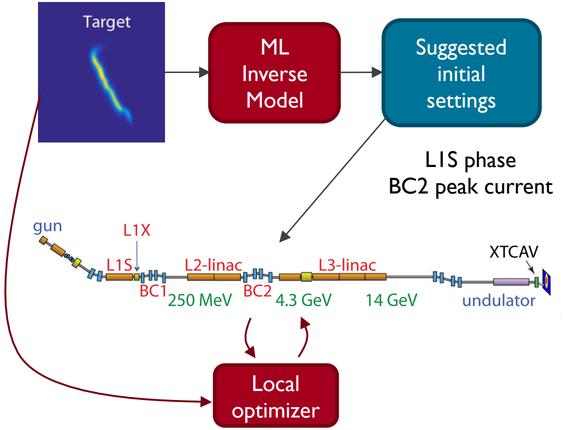
< *ms execution speed*

10^6 times speedup

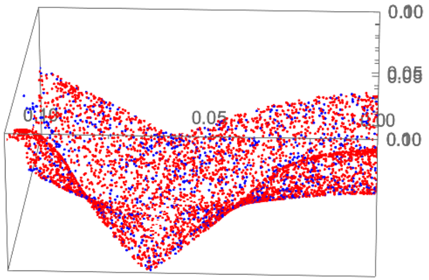
ML modeling enables high-fidelity predictions of system responses with unprecedented speeds, opening up new avenues for high-fidelity online prediction, tracking of machine behavior, and model-based control

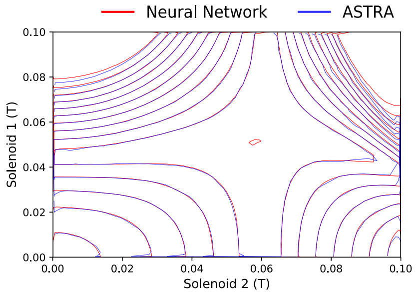
Warm starts for optimization

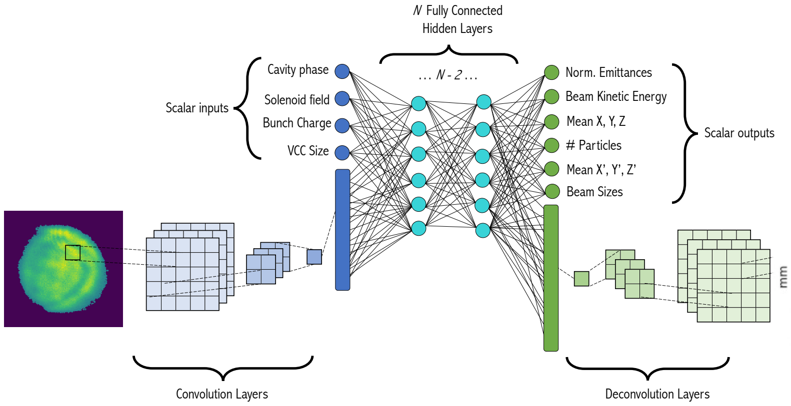
A. Scheinker, A. Edelen, et al, PRL, 2018



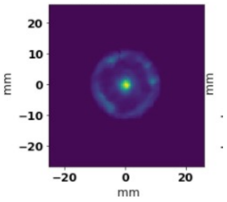
Smooth interpolation
Example σ_x surface from 2D scan, LCLS-II Injector

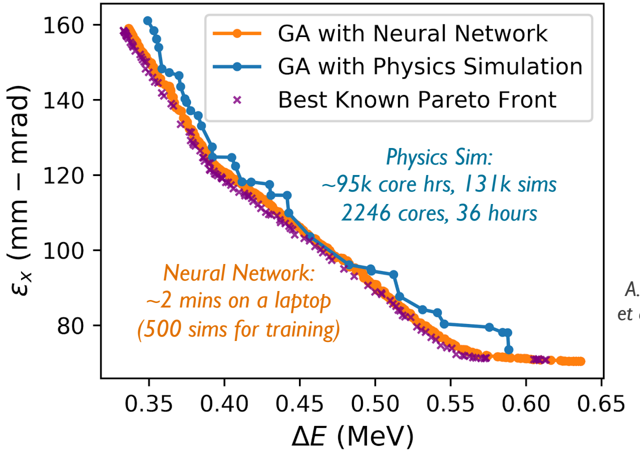






L. Gupta, et al, MLST, 2021





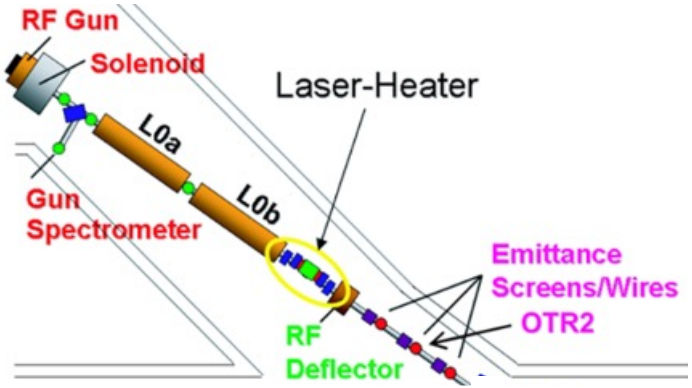
A. Edelen et al., PRAB, 2020

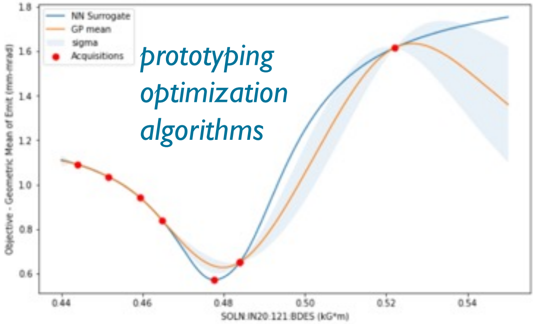
Include high-dimensional input information → better output predictions

Surrogate-boosted design optimization (example on AWA)

Example: Injector Surrogate Model at LCLS

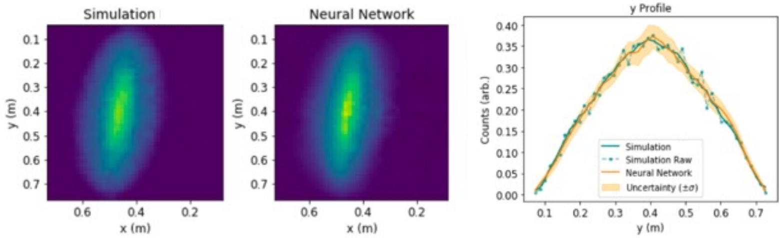
- ML models trained on physics simulations
- Inputs sampled widely across valid ranges
- **Used to develop/prototype new algorithms before testing online at FACET-II and LCLS** *e.g. new Bayesian optimization methods, adaptive emittance measurement*



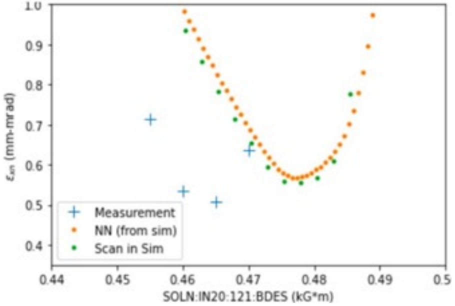


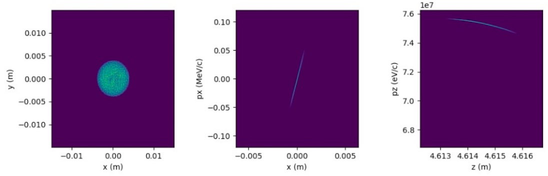
prototyping optimization algorithms

ML model provides accurate replication of simulation



Simulation and ML model trained on it are qualitatively similar to measurements





interactive model widget and visualization tools

ML models trained on simulations enable fast prototyping of new optimization algorithms → greatly reduces development time

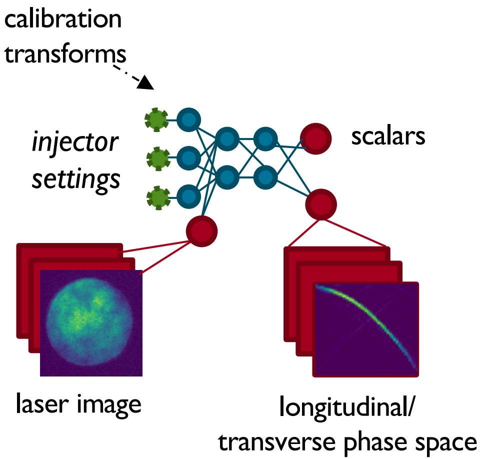
Finding Sources of Error Between Simulations and Measurement

Many non-idealities not included in physics simulations:

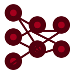
- **static error sources** (e.g. magnetic field nonlinearities, physical offsets)
- **time-varying changes** (e.g. temperature-induced phase calibrations)

*Want to identify these to get **better understanding of machine***

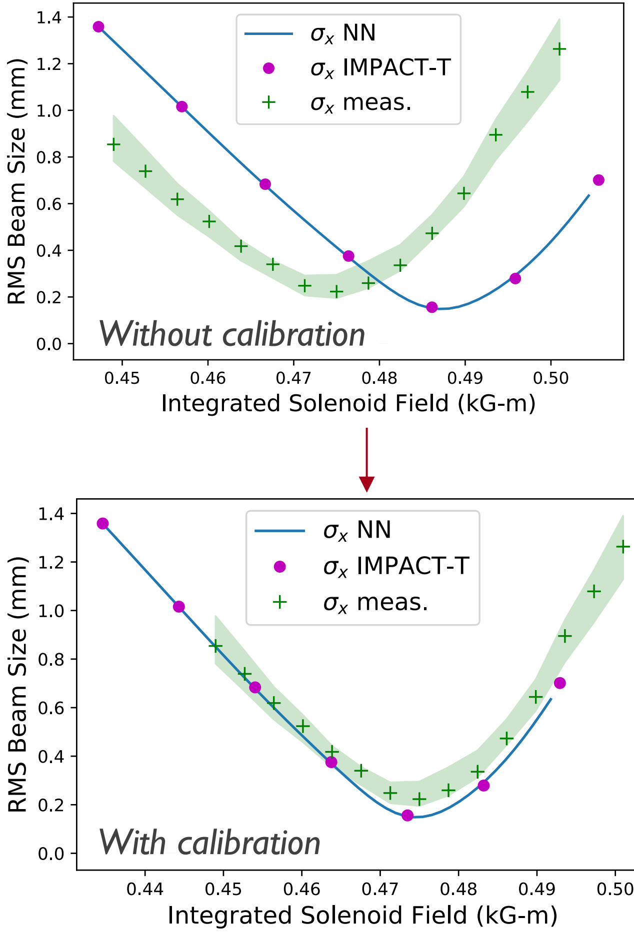
→ fast-executing ML model allows fast / automatic exploration of possible error sources



Inputs
Laser radius
Laser spot sizes
Pulse length
Charge
Solenoid
L0A phase
L0B phase
SQ quad
CQ quad
6 matching quads



Outputs
Beam size (x,y)
Emittance (x,y)
Bunch length

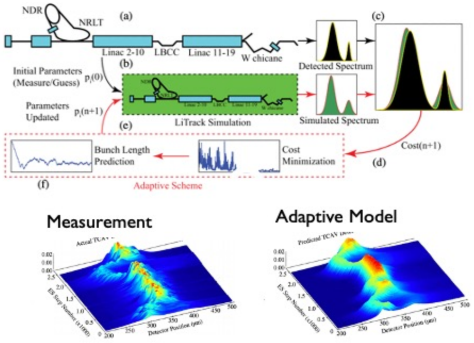


Here: calibration offset in solenoid strength found automatically with neural network model (trained first in simulation, then calibrated to machine)

Virtual Diagnostics

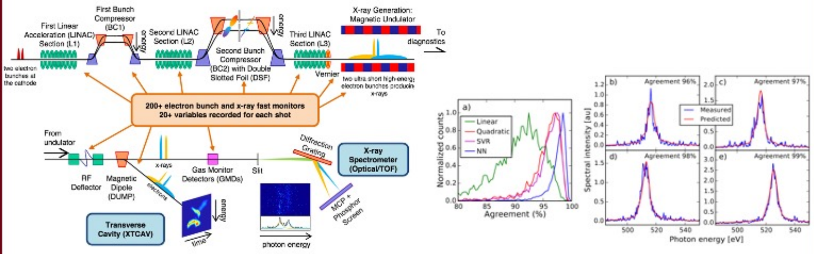
Provide information about parts of the system that are typically inaccessible (destructive, too slow, not directly measurable)

Adaptively tune a simple physics model



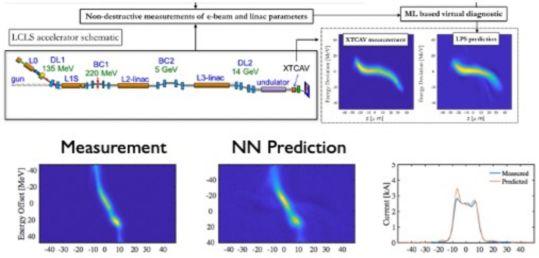
A. Scheinker, S.Gessner, PRAB 18, 102801 (2015)

Fill in shots: use archive data to learn correlation between fast and slow diagnostics



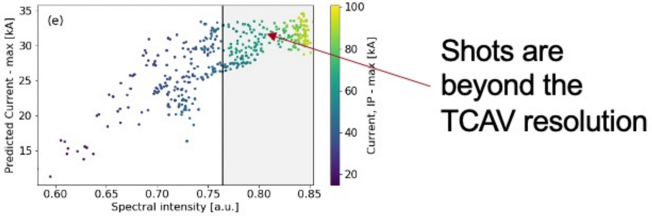
A. Sanchez-Gonzalez, et al., Nature Comms (2017)

Predict with a trained neural network



C. Emma, A. Edelen, et al., PRAB21, 112802 (2018)

Can use spectral information as input to predict beyond typical diagnostic resolution

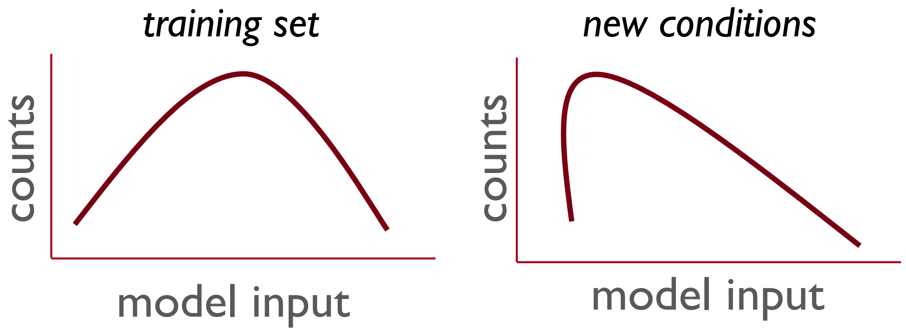


A. Hanuka, et al. 2009.12835 [accepted to Nature Scientific Reports]

Fundamental problem for using models online and for tuning: **distribution shift**

→ *accuracy is degraded on data outside of the statistical distribution of the training data*

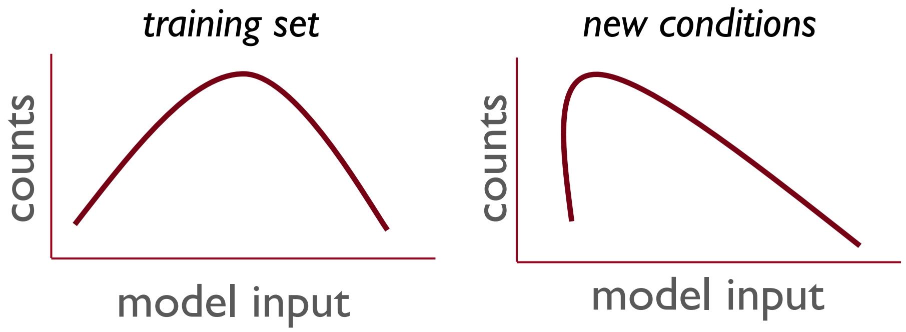
→ ***many ML approaches don't consider uncertainty estimates***

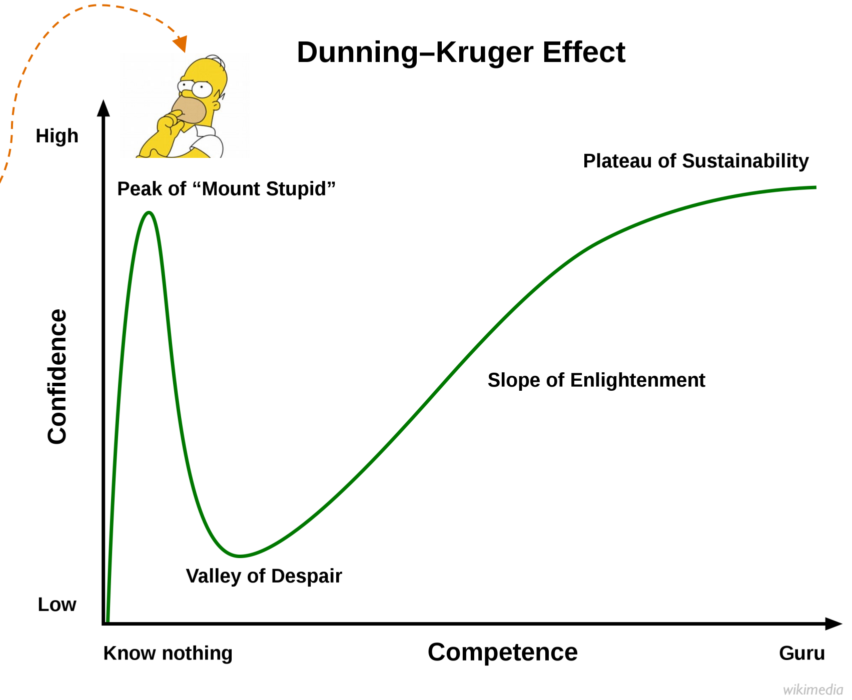


Fundamental problem for using models online and for tuning: **distribution shift**

→ *accuracy is degraded on data outside of the statistical distribution of the training data*

→ ***many ML approaches don't consider uncertainty estimates***

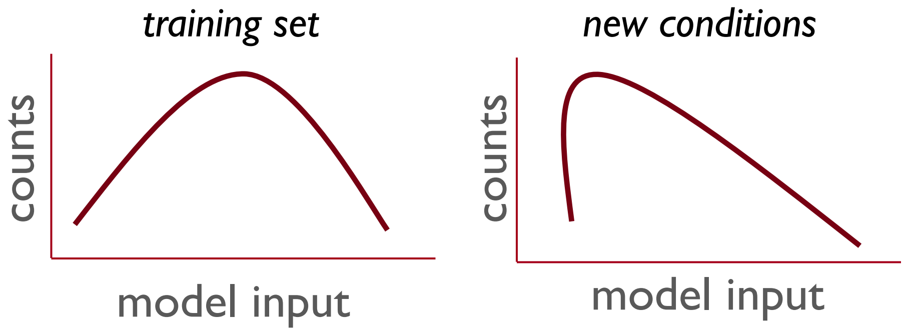


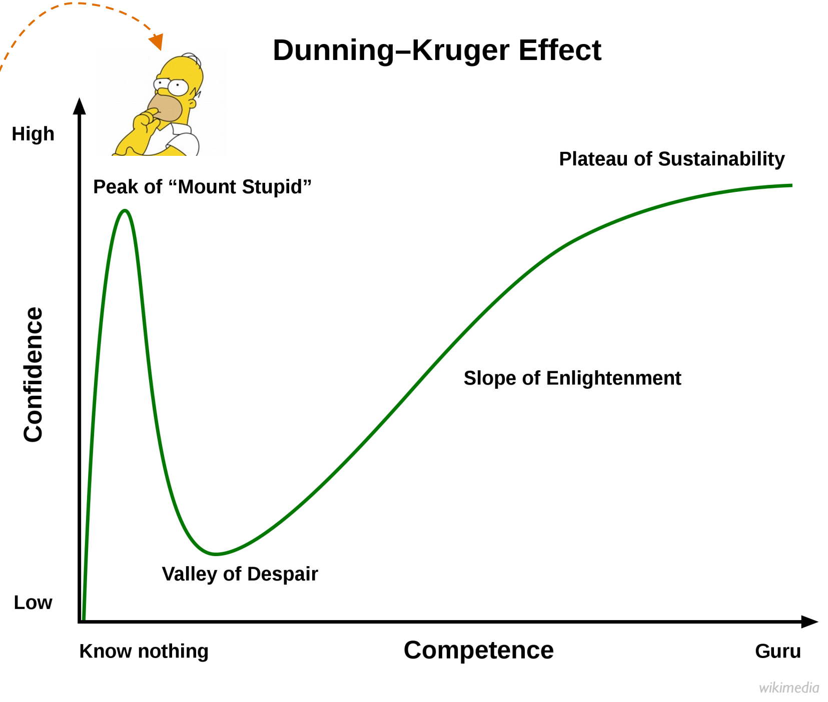


Fundamental problem for using models online and for tuning: **distribution shift**

→ *accuracy is degraded on data outside of the statistical distribution of the training data*

→ ***many ML approaches don't consider uncertainty estimates***



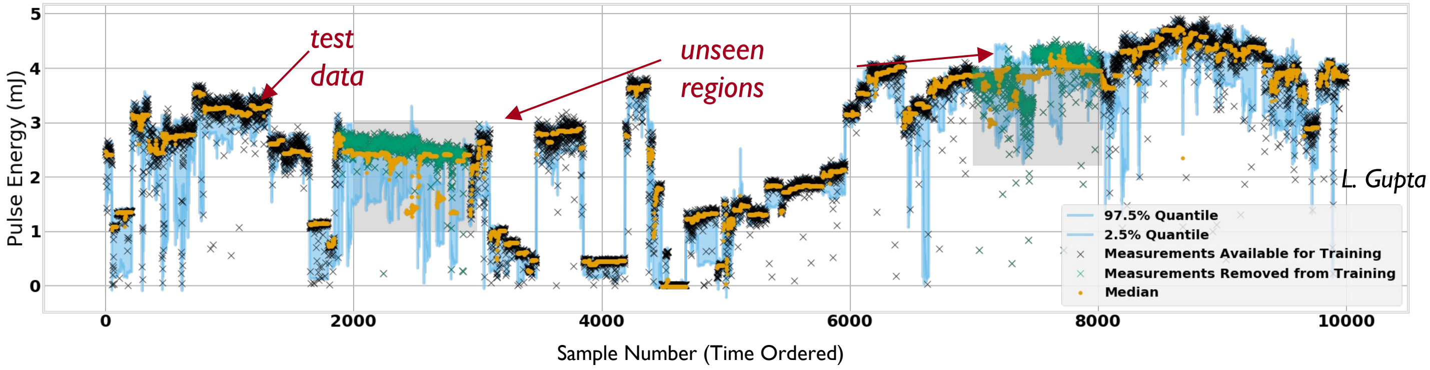


Want to have a reliable model confidence metric before using predictions

→ **need uncertainty quantification / robust modeling**

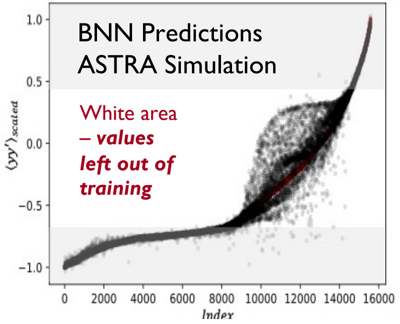
Uncertainty Quantification / Robust Modeling

Essential for decision making under uncertainty (e.g. safe opt., intelligent sampling, virtual diagnostics)



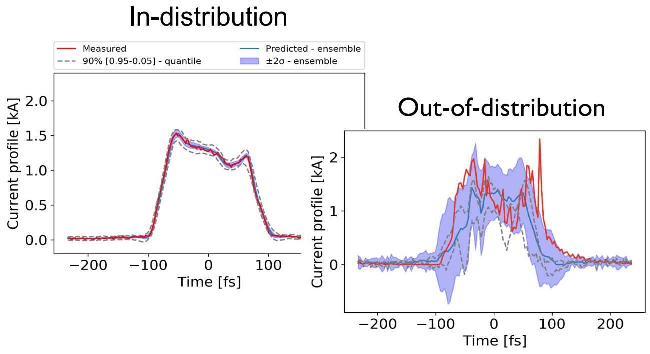
Current approaches
- Ensembles
- Gaussian Processes
- Bayesian NNs
- Quantile Regression

Neural network with quantile regression predicting FEL pulse energy at LCLS



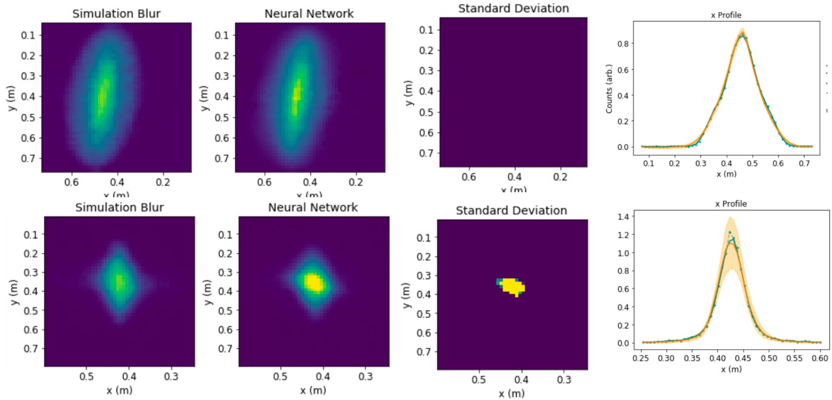
Scalar parameters for the LCLS-II injector (Bayesian neural network)

A. Mishra et. al., PRAB, 2021



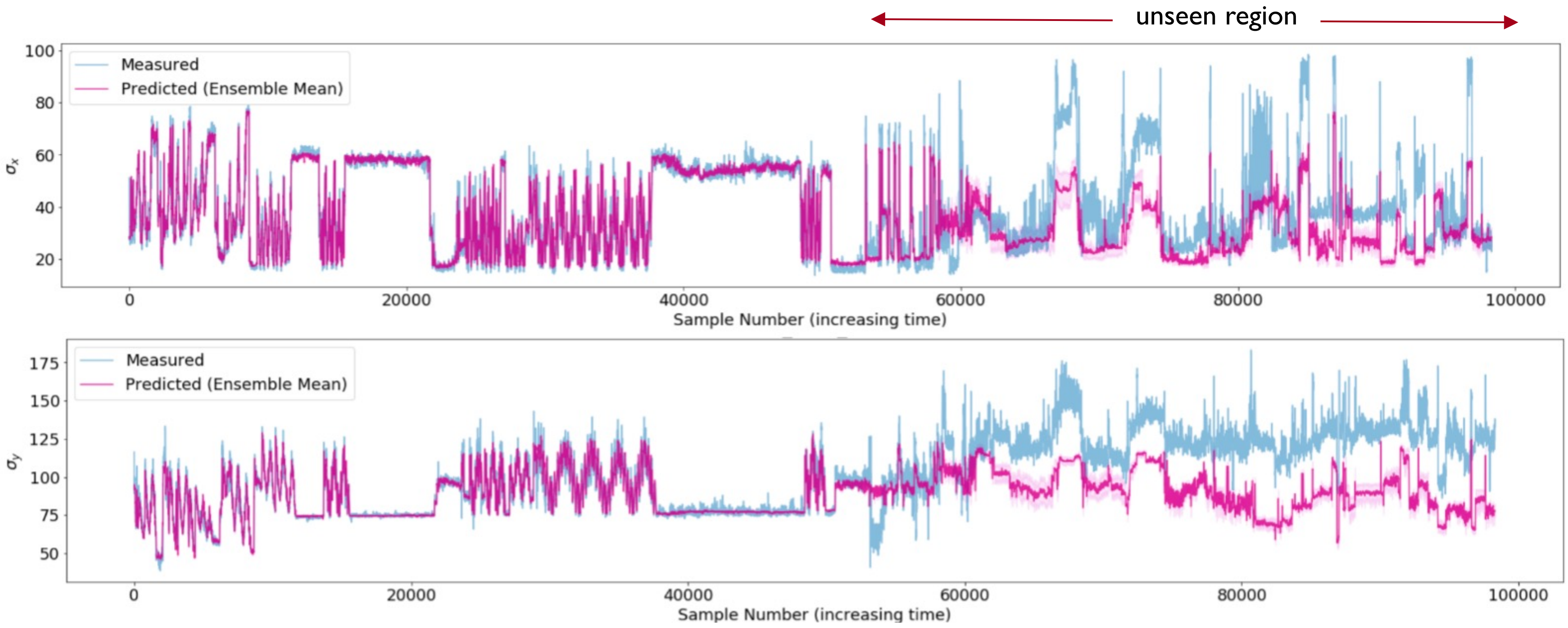
longitudinal phase space (quantile regression + ensemble)

O. Convery, et al., PRAB, 2021



LCLS injector transverse phase space (ensemble)

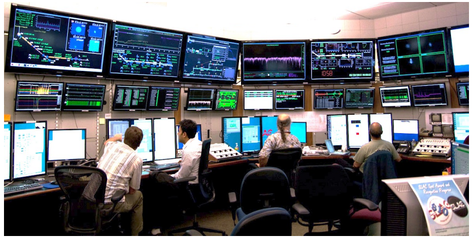
Example of beam size prediction and uncertainty estimates under drift from a neural network (@ UCLA Pegasus)



Uncertainty estimate from neural network ensemble does not cover the OOD prediction error, but it does give a qualitative metric for relative uncertainty

Data sets also present a challenge:

- Most examples above used thousands to tens-of-thousands of examples
- Not feasible to gather new data in every configuration *(from simulation or measurements)*
- Not everyone has access to large compute resources or ample beam time



→ how can we increase model generalization to new conditions and decrease data set sizes (i.e. improve sample-efficiency)?

→ inherent question: how to make ML models more readily adaptable?

"Physics-informed" modeling → *incorporate physics domain knowledge to reduce need for data, and aid interpretability + generalization*

Many approaches:

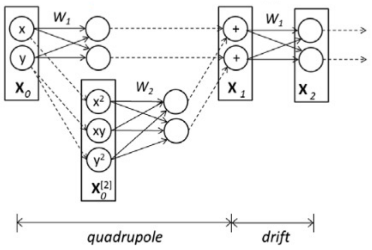
- Combine physics representations and machine learning models directly *(e.g. differentiable simulations)*

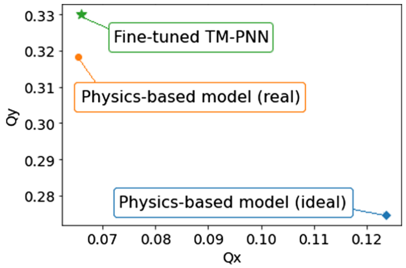
- Add physics constraints to output metrics

- Force to satisfy expected symmetries *(e.g. inductive biases in ML model)*

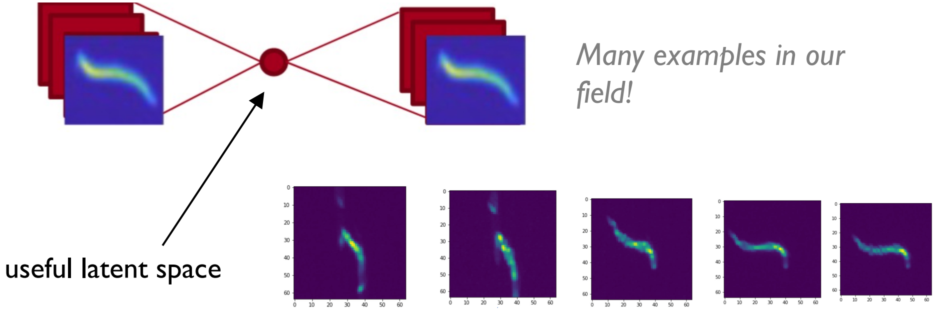
- Loose form: learn from many physics sims in a way that results in good representation of the physics *(also related to representation learning)*

Differentiable Taylor map physics model + weights → train like ML model
needed very little data to calibrate PETRA IV model
Ivanov et al, PRAB, 2020





Physics-driven representation learning
(e.g. encoder-decoder neural network models)



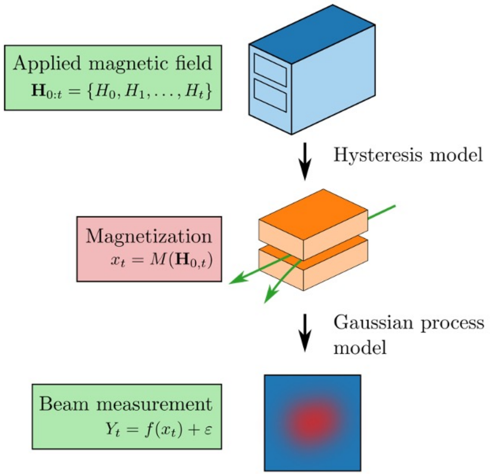
Many examples in our field!

Review paper: Karniadakis et al, *Nat Rev Phys* **3**, 422–440 (2021)
Snowmass accelerator modeling white paper: arXiv:2203.08335

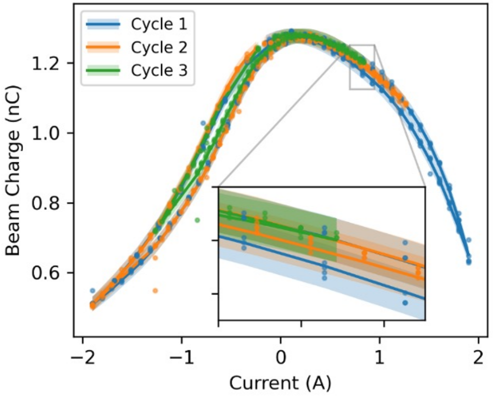
Differentiable Physics Simulations and ML

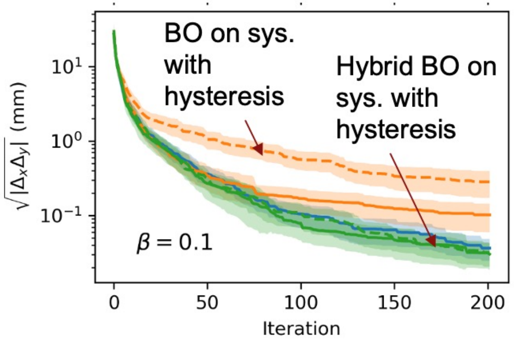
Modern ML uses gradients in learning → *differentiable physics sims enable modular combinations with ML components, analyses, etc.*

Fundamentally new approach in combining physics models, data, and ML



Differentiable physics model of hysteresis combined with ML enables in situ characterization of magnetic hysteresis in accelerator magnets and higher-precision optimization



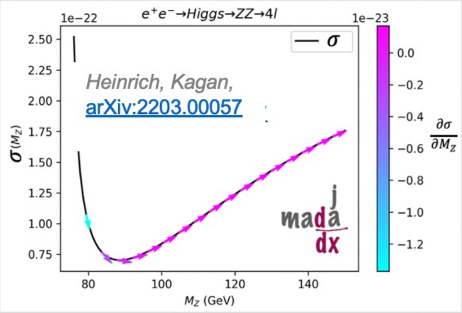


R. Roussel, et al., PRL, 2022, arXiv:2202.07747

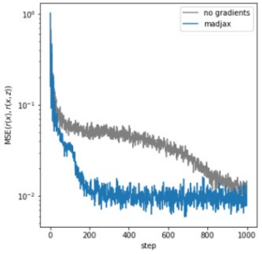
arXiv:2203.13818

Toward the End-to-End Optimization of Particle Physics Instruments with Differentiable Programming: a White Paper

Differentiable physics models can facilitate instrument-wide optimization, from accelerator to detector to physics analysis



Differentiable matrix elements of high energy scattering processes



ML-Assisted Optimization and Characterization

Large, nonlinear, and sometimes noisy search spaces for accelerators and detectors → need to find optima and examine trade-offs with limited budget *(computational resources, machine time)*

ML-assisted optimization **leverages learned representations** to improve sample efficiency. Some methods also include **uncertainty estimation** to inform where to sample next *(avoid undesirable regions, target information-rich areas).*

Similar set of tools for operation and design *(with a few differences: parallel vs. serial acquisition, need for uncertainty-aware/safe optimization)*

Bayesian optimization / active learning / reinforcement learning
→ All learn iteratively via online interaction with the system

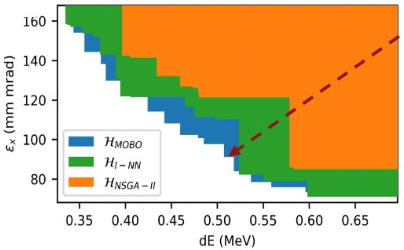
next point to search

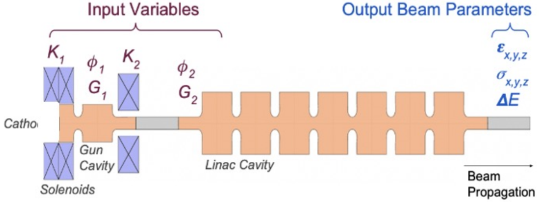
Learning Algorithm (GP model + Bayes. opt., iterative surrogate, RL agent)

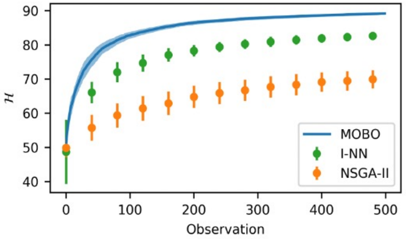
System (Simulation or Online Interaction)

data

Pareto front: optimal tradeoff between parameters of interest



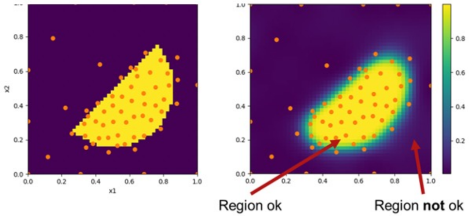




Faster multi-objective optimization with Bayesian optimization and iterated surrogate models

R. Roussel et al., arXiv:2010.09824

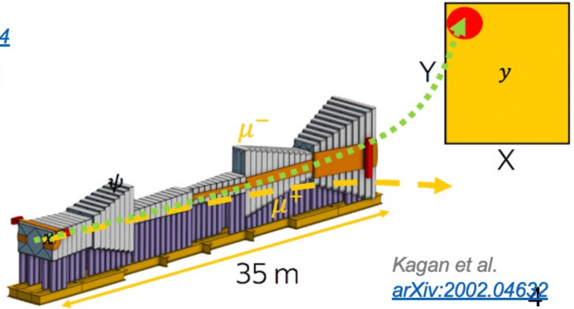
A. Edelen et al., arXiv:1903.07759



Output constraints learned on-the-fly

R. Roussel et al., arXiv:2106.09202

Local generative surrogates and gradient descent for the SHiP magnetic shield design



Kagan et al.
arXiv:2002.04632

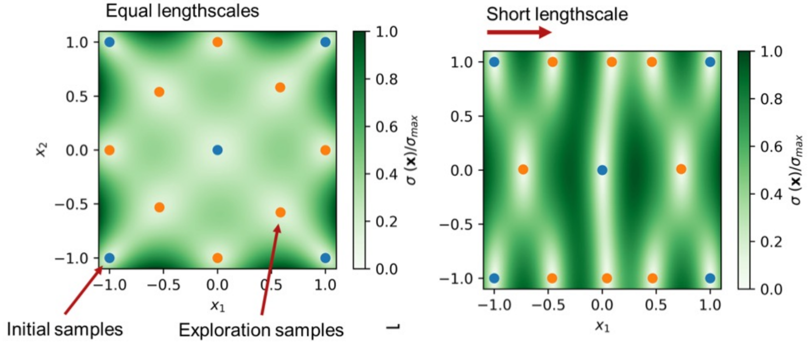
Efficient Characterization with Bayesian Exploration

$$\alpha(\boldsymbol{x}) = \sigma(\boldsymbol{x}) \prod_{i=1}^{N} p_i(g_i(\boldsymbol{x}) \geq h_i)\, \Psi(\boldsymbol{x}, \boldsymbol{x_0})$$

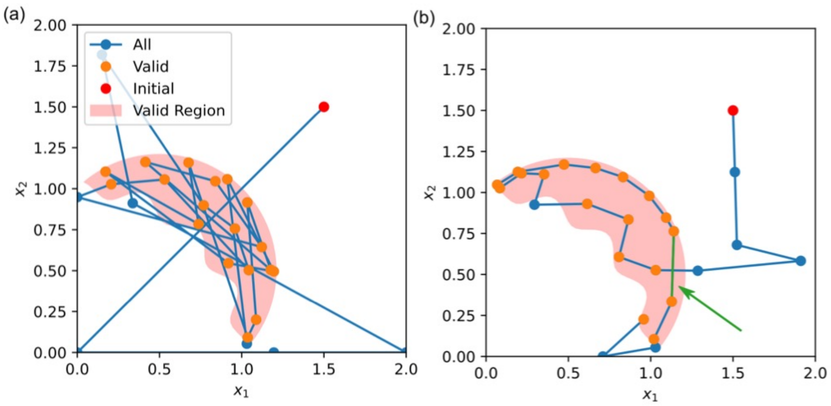
R. Roussel et. al.
Nat. Comm. **2021**

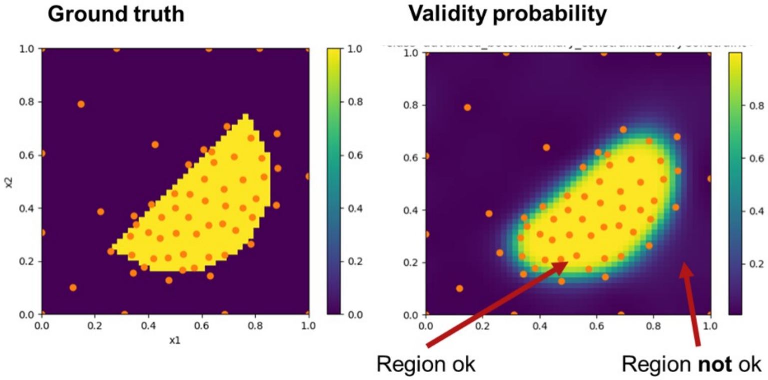
proximal biasing

adaptive sampling



learning constraints

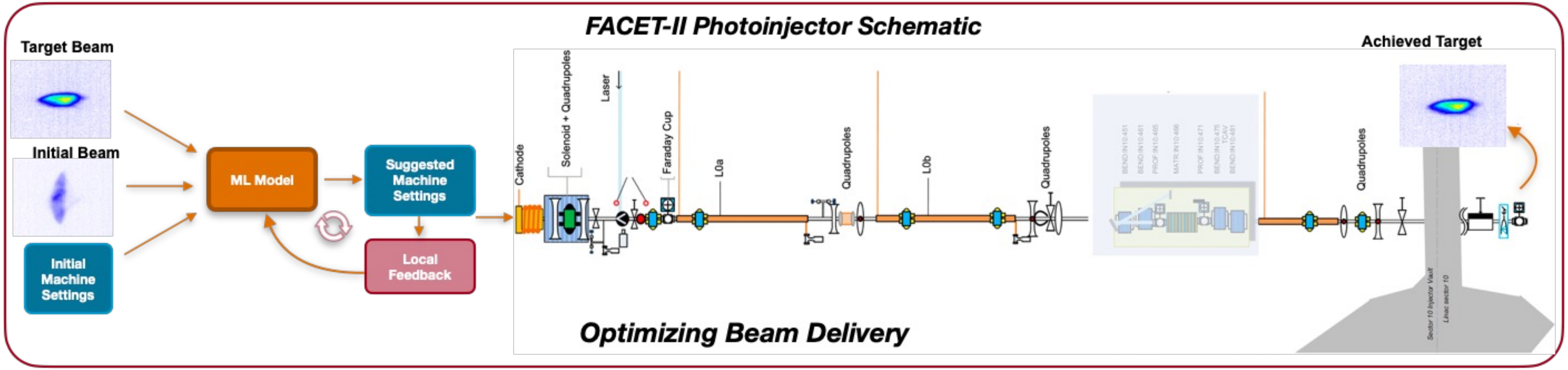




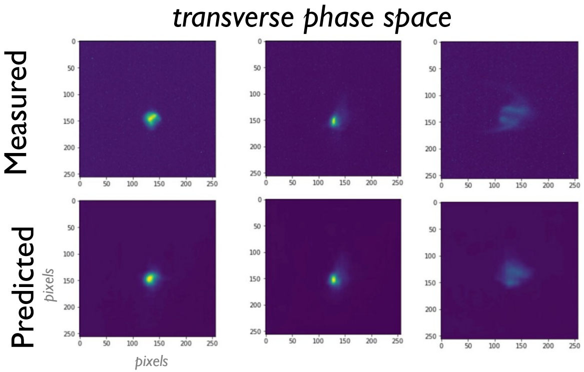
Enables sample-efficient characterization of high-dimensional spaces, while respecting both input and output constraints

See Ryan's tutorial on Friday!

Example: FACET-II Injector Characterization, Modeling, and Optimization



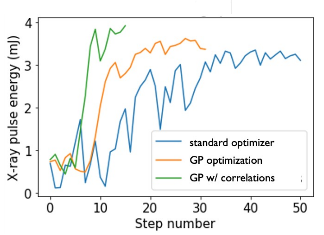
- Used Bayesian Exploration for efficient high-dimensional characterization (10 variables) at 700pC: **2 hrs for 10 variables compared to 5 hrs for 4 variables with N-D parameter scan**
- Data was used to train ML models to predict + optimize beam emittance and injector match
- **Example of integrated cycle between characterization, modeling, and optimization → now extending to larger system sections and new setups (e.g. two-bunch)**



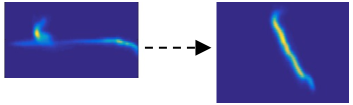
Use of Bayesian exploration to generate training data is sample-efficient, reduces burden of data cleaning, and can result in a well-balanced distribution for the training data set over the input space.

Broad Set of Areas for ML to Impact Operation

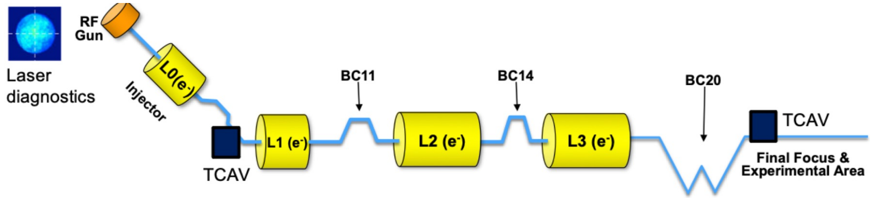
automated control + optimization



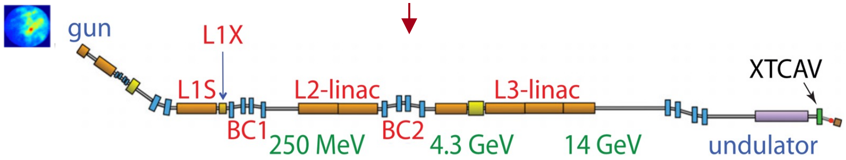
J. Duris et al., PRL, 2020



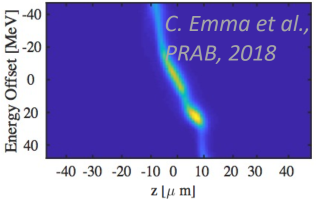
Data reduction/rejection *(kHz/MHz data streams)*
Event triggering



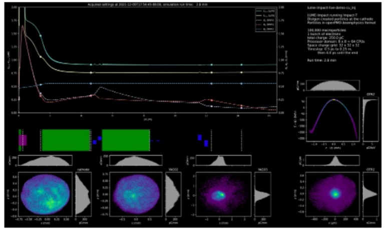
algorithm transfer between systems

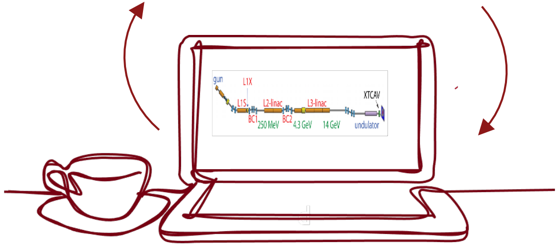


ML-enhanced diagnostics
(provide insight at faster rate, at higher resolution, non-invasively)



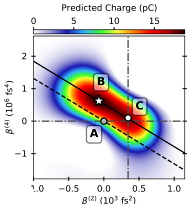
anomaly detection
failure prediction
(plan maintenance; alert to changes in machine; alert to interesting science)





digital twins + online modeling
(fast sims, differentiable sims, model calibration, model adaptation)

extract unknown relationships + correlations
(feed into future control / design)



R. Shaloo et al. arXiv:2007.14340

+ need uncertainty quantification for all
+ can incorporate physics information in all

Modular, Open-Source Software Development

Community development of **re-usable, reliable, flexible software tools** for AI/ML workflows is essential to maximize return on investment and ensure transferability between systems

Modularity is key: separating different parts of the workflow + using shared standards

Xopt

Optimizer

Simulation

Data Set

standard data format

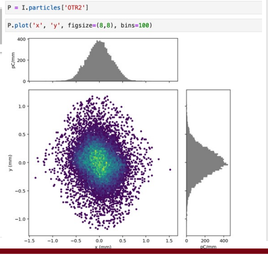
standard algorithm definition

Impact
ASTRA
GPT
Bmad
Genesis
SRW
work in progress: elegant

LUME

openPMD

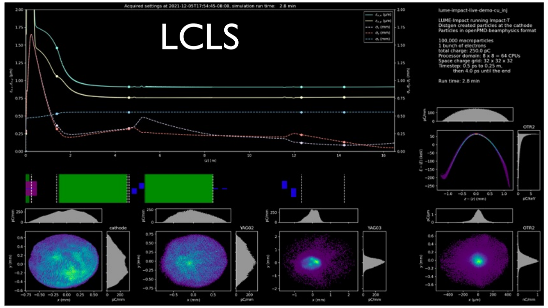
gen_1.json
▾ root:
▸ variables:
generation: 1
▸ vocs:
▸ error: [] 1241 items
▸ inputs: [] 1241 items
▸ outputs: [] 1241 items

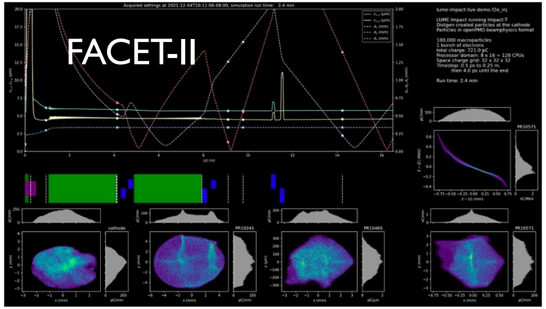



```
algorithm:
  name: bayesian_exploration
  options:
      n_initial_samples: 5
      n_steps: 25
      generator_options:
          batch_size: 1
          #sigma: [[0.01, 0.0],
          use_gpu: False


simulation:
  name: test_TNK
  evaluate: xopt.tests.evaluators

vocs:
  name: TNK_test
  variables:
    x1: [0, 3.14159]
    x2: [0, 3.14159]
  objectives: {y1: MINIMIZE}
  constraints:
    c1: [GREATER_THAN, 0]
    c2: ['LESS_THAN', 0.5]
```





Online Impact-T simulation and live display for FACET-II injector; trivial to get running using same software tools as the LCLS injector

Different software for different tasks:

Optimization algorithm driver (*e.g. Xopt*)

Visual control room interface (*e.g. Badger*)

Simulation drivers (*e.g. LUME*)

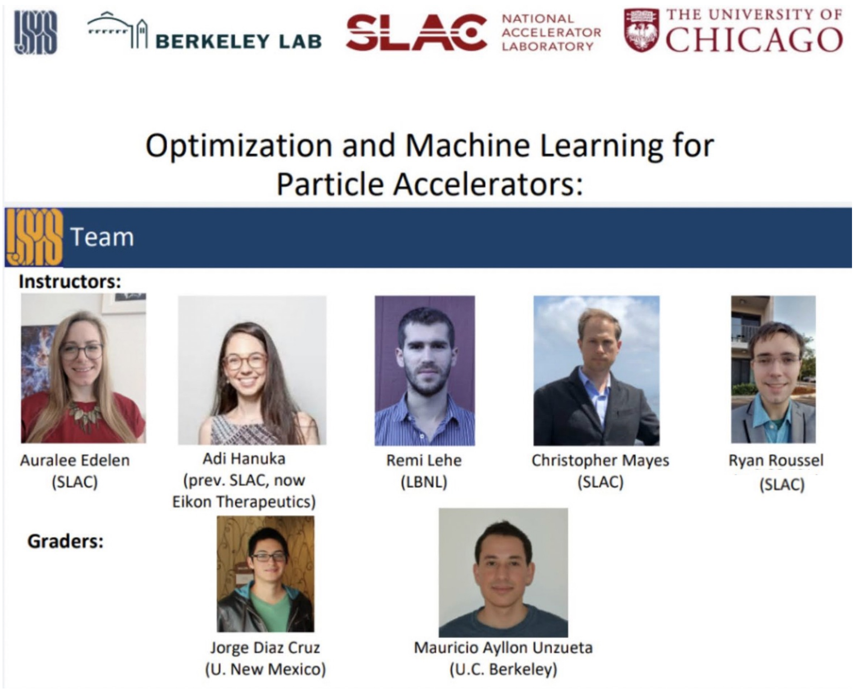
Standards model descriptions, data formats, and software interfaces (*e.g. openPMD*)

Online ML model deployment

More details at https://www.lume.science/

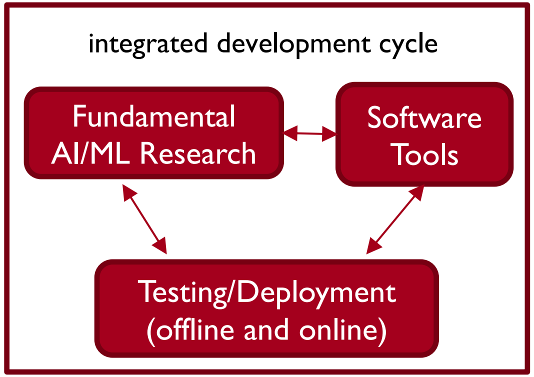
Conclusions

- Many proof-of-principle results and prototypes form a solid foundation for future work
- AI/ML tools can improve achievable beam characteristics, reduce tuning time, and aid understanding of experiments → now need integration into regular operation
- Current/future efforts focus on improving robustness, developing hybrid physics + ML methods, developing techniques to scale up to larger machine sections (requires new algorithms/workflows) and more challenging setups, and continued software development/deployment into regular operation
- Want to learn more? See the USPAS class "Optimization and Machine Learning for Particle Accelerators" https://slaclab.github.io/USPAS_ML/

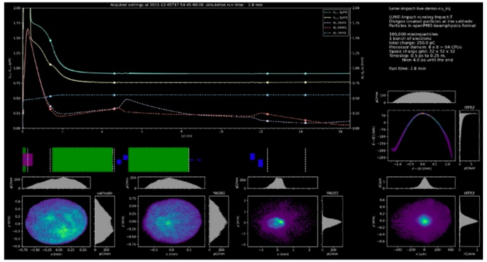


Thank you for your attention!

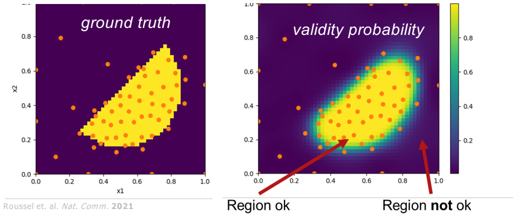
Broad Research Program in AI/ML for Accelerators



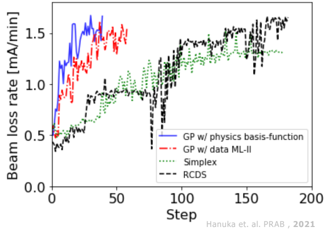
Online prediction with physics sims and **fast/accurate ML models**



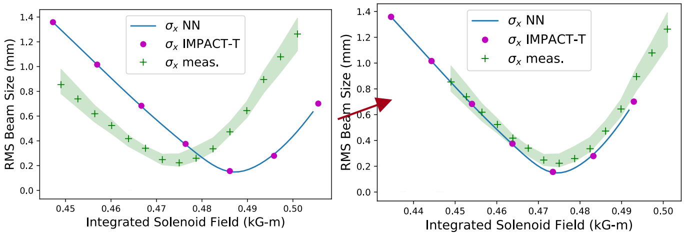
Efficient optimization and characterization *(useful also for simulation exploration/design, data generation)*



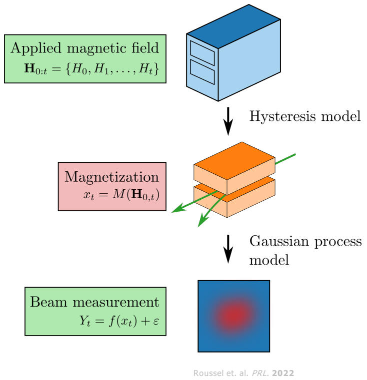
Output constraints learned on-the-fly



Adaptation of models and **identification of sources of deviation** between simulations and as-built machine

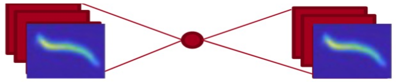


Techniques for **combining physics and ML** *(more reliable/transferrable, require less data, more interpretable)*, including **differentiable simulators**



Representation learning

(e.g. better ways of modeling beams)

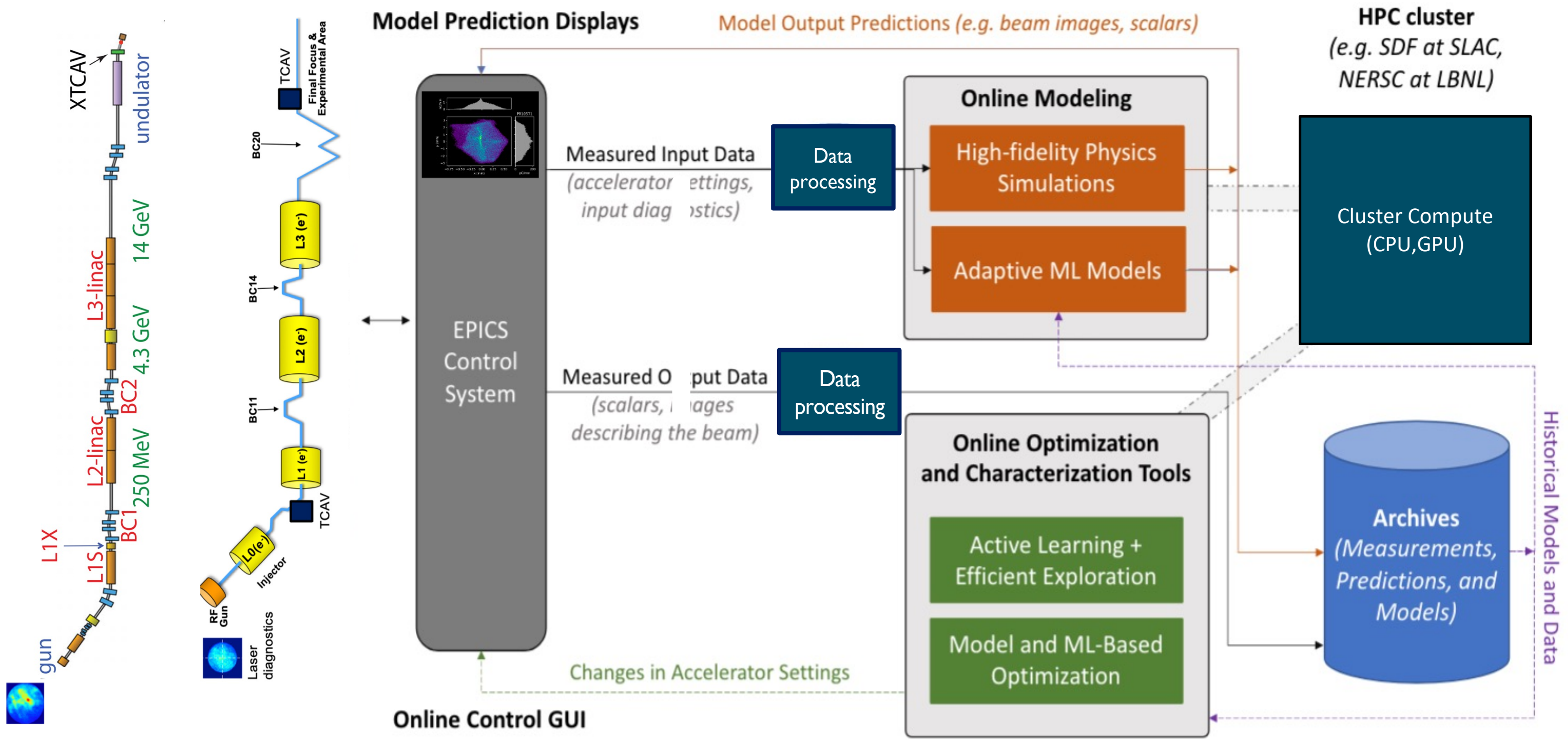


Software packages and standards for data generation, modeling, and optimization (*LUME, xopt, Badger*)

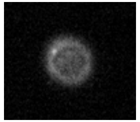
Future: Full Integration of AI/ML Optimization, Modeling, and Physics Simulations

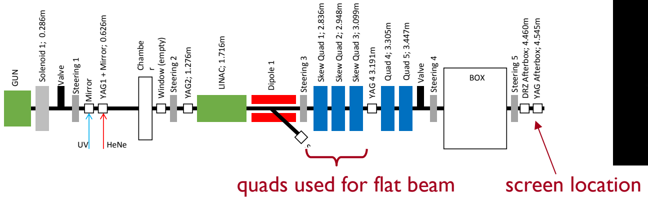
Need to integrate disparate methods and proof-of-principle results into a ***facility-agnostic*** ecosystem for online simulation, ML modeling, and AI/ML driven characterization/optimization

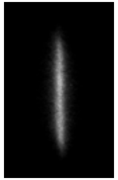
Will enable system-wide application to aid operations, and help drive AI/ML development *(e.g. higher dimensionality, robustness, combining algorithms efficiently)*



On machine: can run optimizer on a learned online model

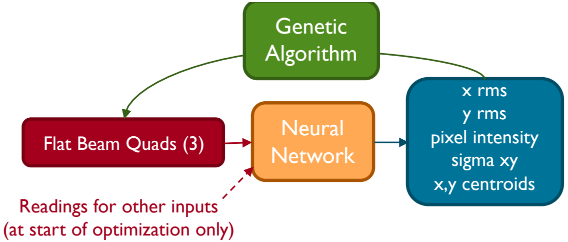


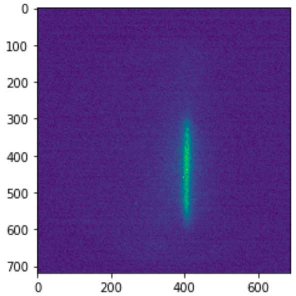




Expert hand-tuning:
10 – 20 minutes

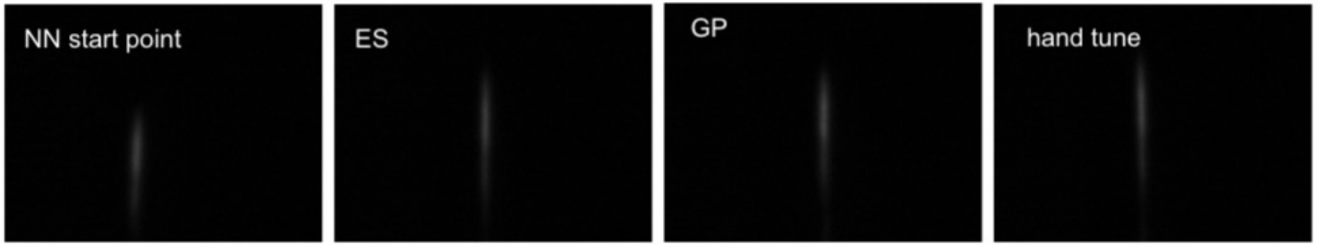
- Round to flat beam transforms are challenging to optimize
- Took measured scan data at Pegasus (UCLA)
- Trained neural network model to predict fits to beam image
- Tested online multi-objective optimization over model (3 quad settings) given present readings of other inputs





Results are for one full day after last training data

Can use neural network to provide first guess at solution, then fine tune with other methods…



Hand-tuning in seconds vs. tens of minutes

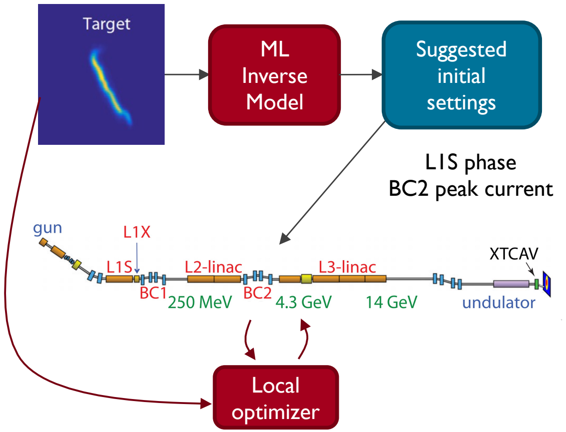
Significant boost in convergence speed for other algorithms

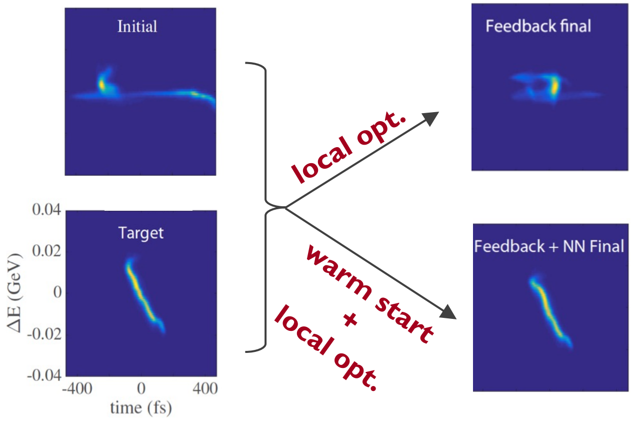
E. Cropp et al., in preparation

Inverse models: example from LCLS

A. Scheinker, A. Edelen, et al., PRL 121, 044801 (2018)

- Use global inverse model to give rough suggested settings → then fine-tune with local optimizer
- Preliminary study at LCLS:
  - Two settings scanned *(L1S phase, BC2 peak current)*
  - Compared optimization algorithm with/without warm start



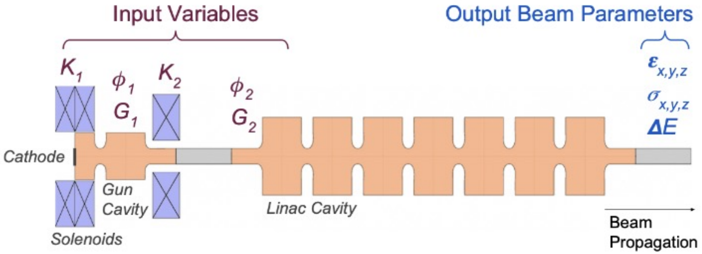


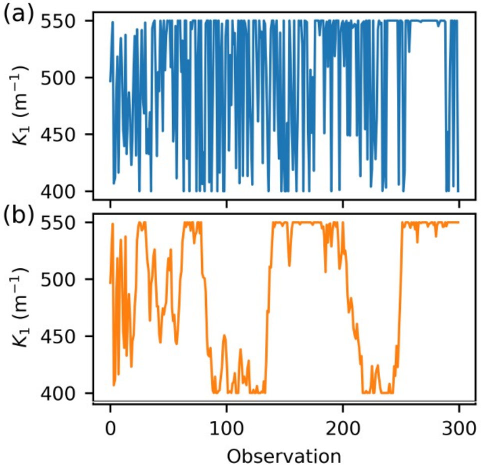
Local optimizer alone was unable to converge → **able to converge after initial settings from neural network**

Example: Multi-Objective Bayesian Optimization (MOBO)

Multi-objective optimization (MOO) in accelerators is traditionally done offline with high performance computing and simulations, or online at individual working points only

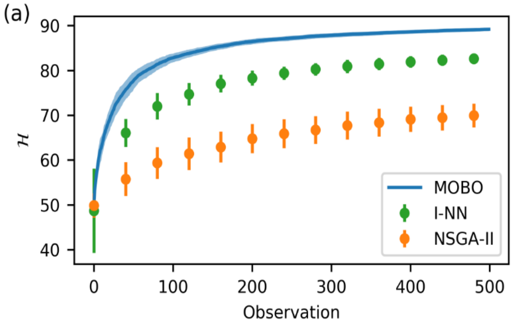
- MOBO enables full characterization of optimal beam parameter tradeoffs (i.e. the Pareto front) online with high sample-efficiency
- Has now been used experimentally at AWA, FACET-II, LCLS and SLAC UED

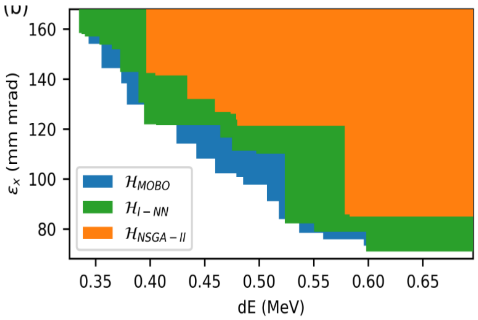




Can enforce smooth exploration

(no wild changes in ... settings)

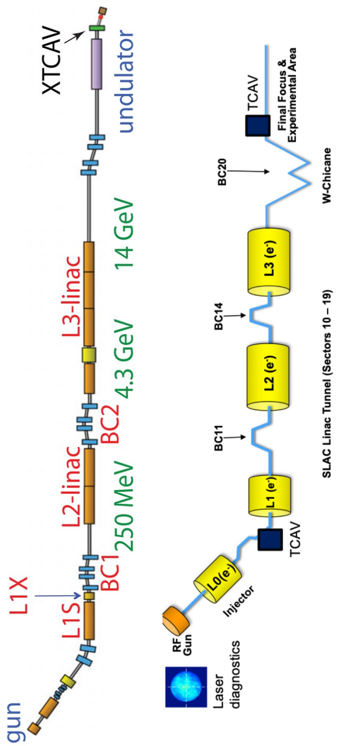


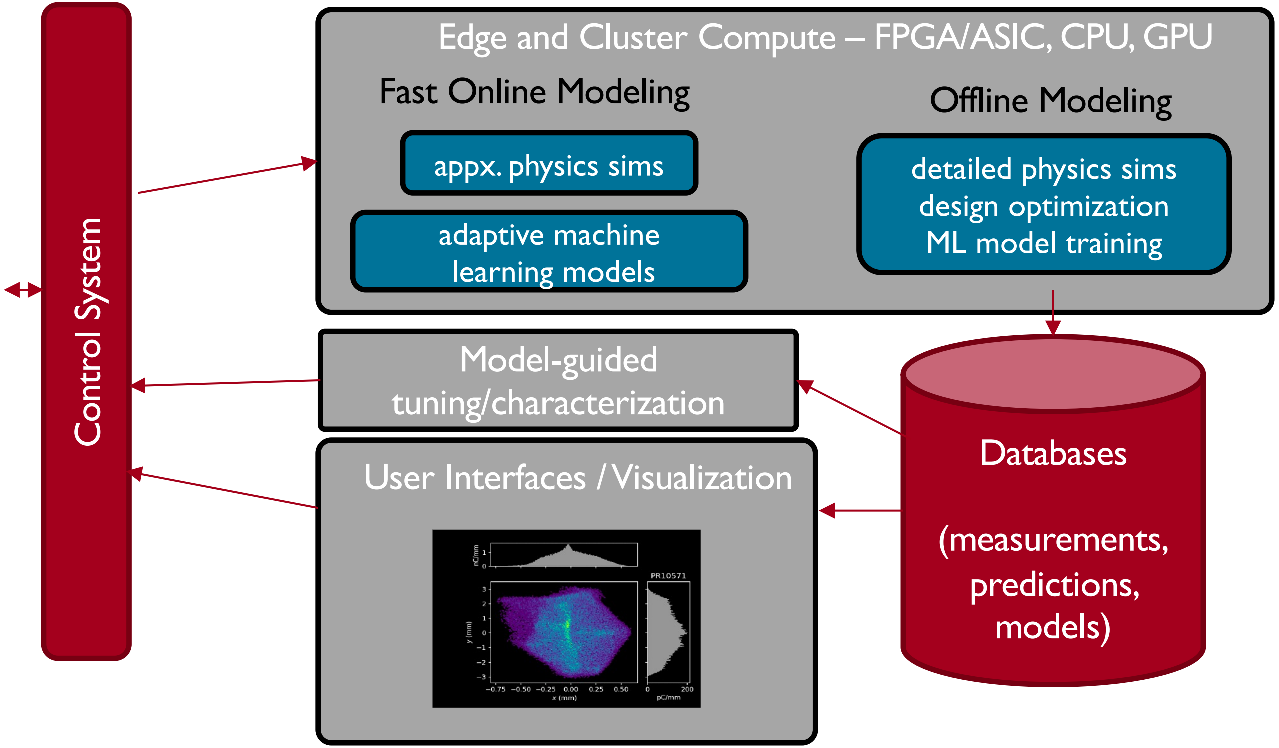


R. Roussel, et al., *PRAB (2021)*

Unprecedented ability to fully characterize tradeoffs between beam parameters in real accelerator systems.

A common dream: fully-integrated virtual accelerator





Snowmass21 Accelerator Modeling Community White Paper

by the Beam and Accelerator Modeling Interest Group (BAMIG)*

Authors (alphabetical): S. Biedron[13], L. Brouwer[1], D.L. Bruhwiler[7], N. M. Cook[7], A. L. Edelen[6], D. Filippetto[1], C.-K. Huang[9], A. Huebl[1], N. Kuklev[4], R. Lehe[1], S. Lund[12], C. Messe[1], W. Mori[10], C.-K. Ng[6], D. Perez[9], P. Piot[4,5], J. Qiang[1], R. Roussel[6], D. Sagan[2], A. Sahai[11], A. Scheinker[9], E. Stern[14], F. Tsung[10], J.-L. Vay[1], D. Winklehner[8], and H. Zhang[3]

Encourage checking out the Snowmass accelerator modeling whitepaper: arXiv:2203.08335