Bayesian Optimization and Reinforcement Learning

SLAC Summer Institute
Aug 11, 2023

Auralee Edelen, Ryan Roussel

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)*

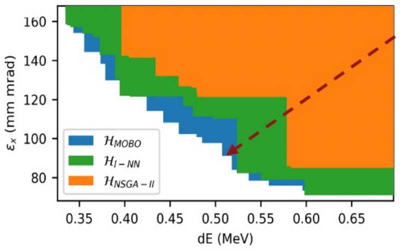
Bayesian optimization / active learning / reinforcement learning
→ All learn iteratively via online interaction with the system

next point to search

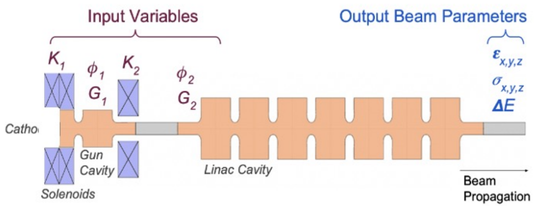
Learning Algorithm
(GP model + Bayes. opt., iterative surrogate, RL agent)

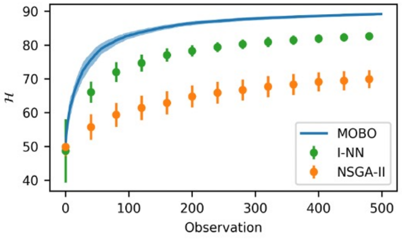
System (Simulation or Online Interaction)

data



Pareto front: optimal tradeoff between parameters of interest

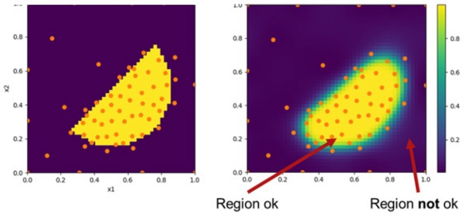




Faster multi-objective optimization with Bayesian optimization and iterated surrogate models

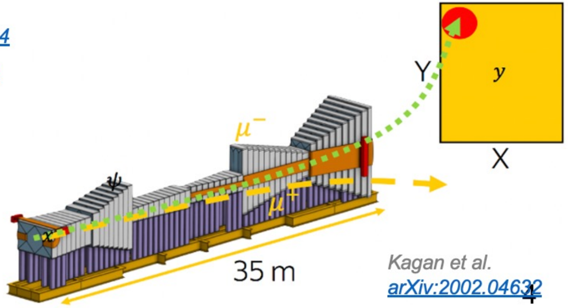
R. Roussel et al., [arXiv:2010.09824]

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]

Examples in Xopt: Flexible Optimization of Arbitrary Problems

Will have links to algorithm examples in Xopt

```python
# create Xopt object.
X = Xopt(YAML)

# take 10 steps and view data
for _ in range(10):
    X.step()

X.data
```

Python interface

```yaml
xopt:
    max_evaluations: 6400

generator:
    name: cnsga
    population_size: 64
    population_file: test.csv
    output_path: .

evaluator:
    function: xopt.resources.test_functions.tnk.evaluate_TNK
    function_kwargs:
      raise_probability: 0.1

vocs:
    variables:
        x1: [0, 3.14159]
        x2: [0, 3.14159]
    objectives: {y1: MINIMIZE, y2: MINIMIZE}
    constraints:
        c1: [GREATER_THAN, 0]
        c2: [LESS_THAN, 0.5]
    linked_variables: {x9: x1}
    constants: {a: dummy_constant}
```

Text file for problem setup

Badger GUI interface

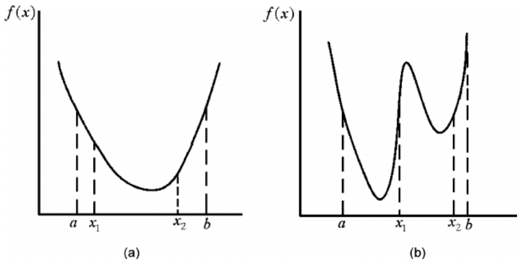
Many optimization algorithms
- Genetic algorithms (NSGA-II, etc.)
- Nelder-Mead Simplex
- Bayesian optimization

https://christophermayes.github.io/Xopt/

https://christophermayes.github.io/Xopt/algorithms/

Optimization Considerations

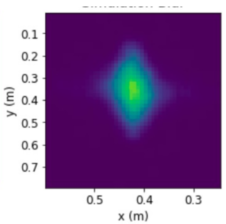
Problem complexity
how difficult is the problem to solve?

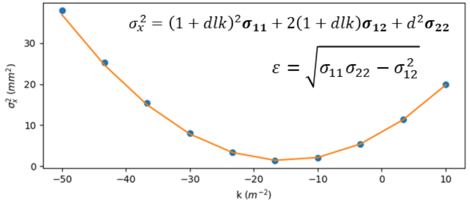


Overhead
how expensive is it to prepare for optimization?

Optimizer cost
how expensive is it to make decisions?

Evaluation cost
how expensive is it to evaluate objectives/constraints?



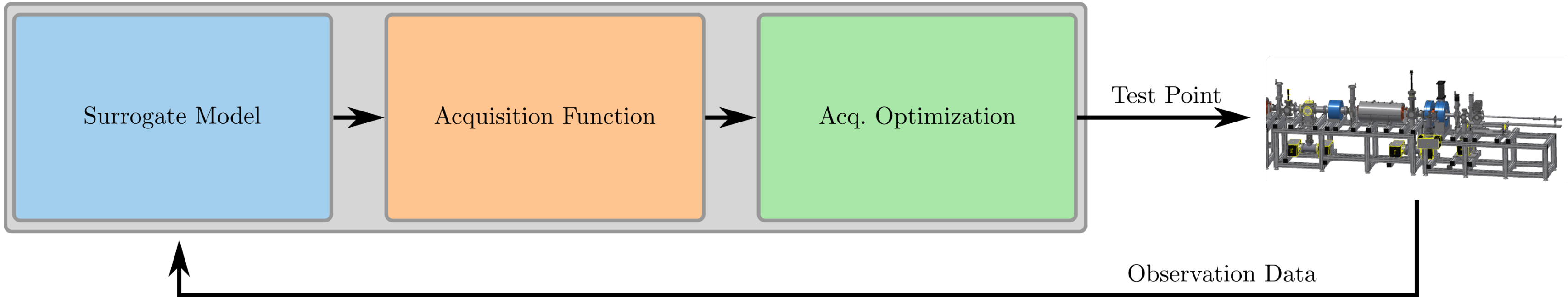


Bayesian Optimization

Create a computational model of the system

Determine the value of potential future measurements

Pick the point that maximizes value

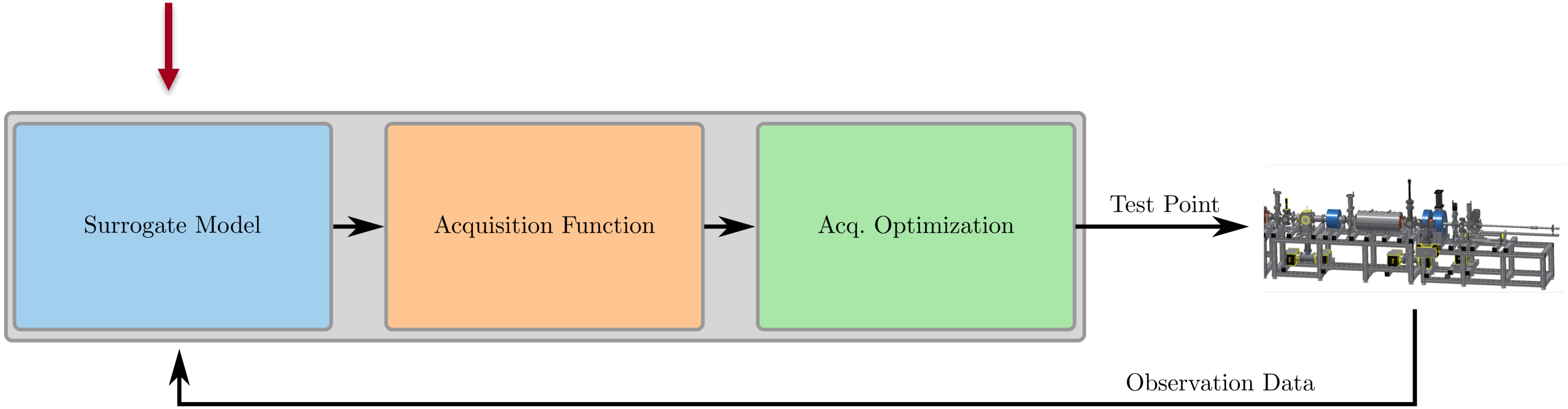


Used broadly in black-box optimization of unknown, noisy functions

[Xopt example]

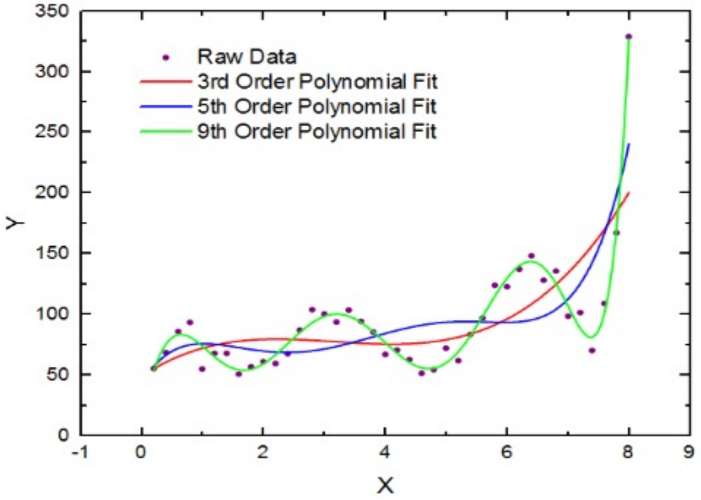
Gaussian Process Modeling

Gaussian processes (GPs)



- Standard model of choice for basic Bayesian Optimization
- Gains information from a small number of data points → sample-efficient
- Accounts for noise and uncertainty → ideal for accelerators + global optimization

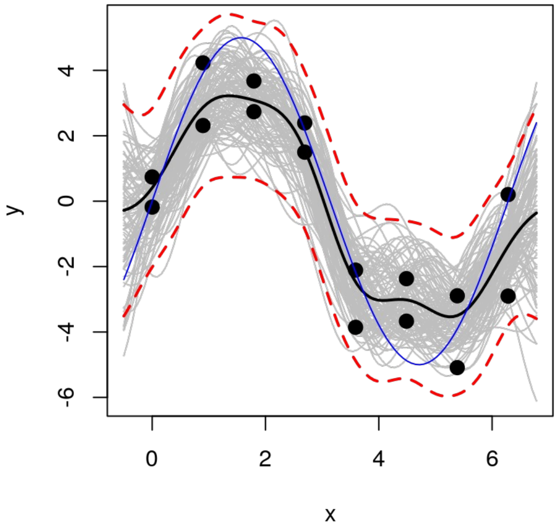
Parametric modeling



$$f(x) = f(x;\theta)$$

e.g. Neural Networks

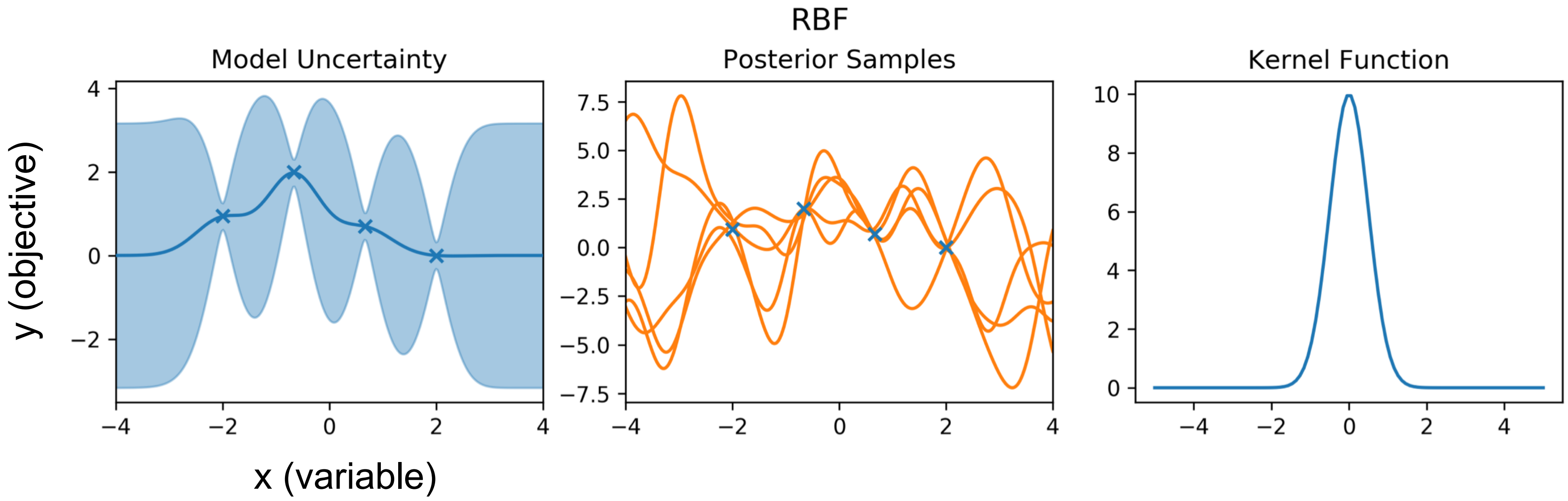
Non-Parametric modeling



$$f = \{\theta_0(x_0), \theta_1(x_1), \ldots, \theta_N(x_N)\}$$

e.g. Gaussian Processes

Gaussian Processes



Courtesy Johannes Kirschner (ETH Zurich)

The **kernel** specifies function value covariances at two points x, x'

→ controls the function behavior

→ parameterized by **hyperparameters** that are automatically fit to the data

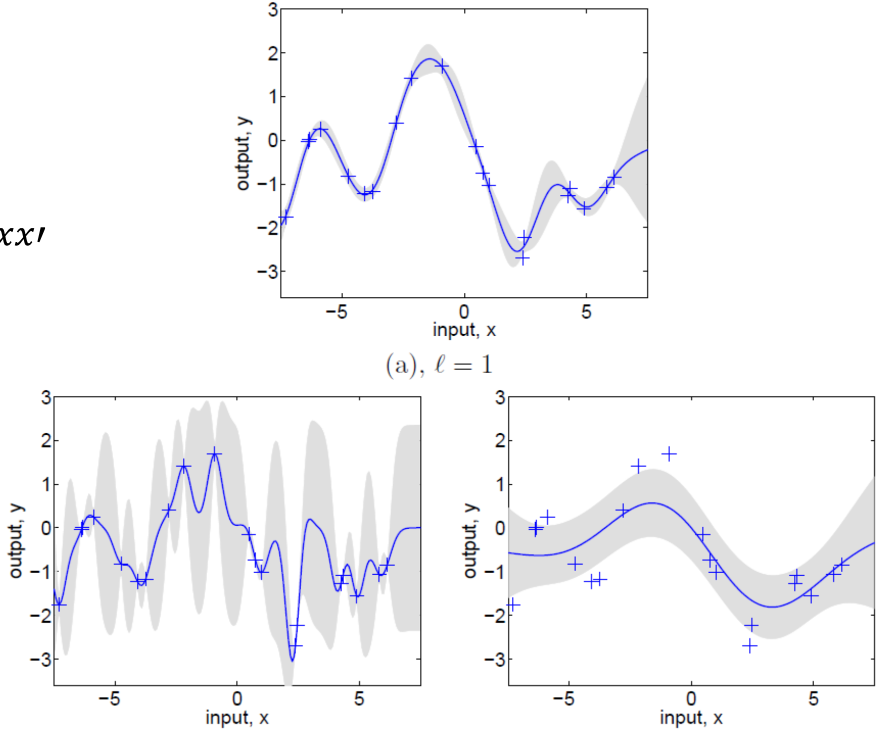
Radial Basis Function:

$$k(x, x') = \boldsymbol{\sigma_f^2} \exp\left(-\frac{1}{2\boldsymbol{l}^2}(x - x')^2\right) + \boldsymbol{\sigma_n^2} \delta_{xx'}$$

Kernel amplitude

Kernel length scale

Noise



(a), $\ell = 1$

(b), $\ell = 0.3$

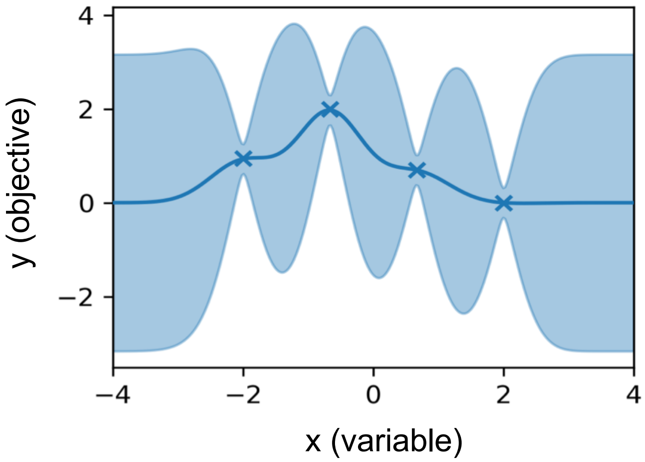
(c), $\ell = 3$

Rasmussen and Williams

Process of Bayesian Optimization

Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points



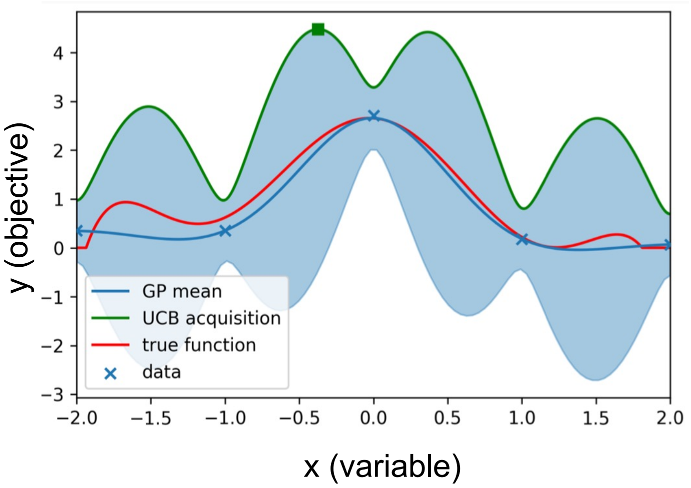
Courtesy Johannes Kirschner (ETH Zurich)

Process of Bayesian Optimization

Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points

Use model predictions and uncertainty to guide search for optimum while sampling



Courtesy Johannes Kirschner (ETH Zurich)

Process of Bayesian Optimization

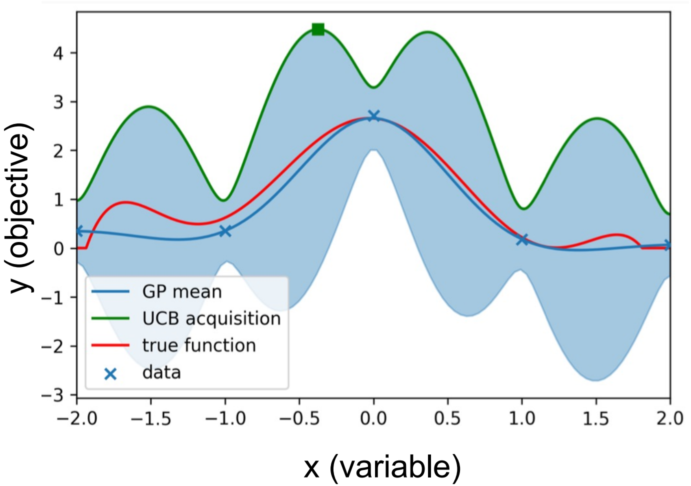
Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points

Use model predictions and uncertainty to guide search for optimum while sampling

Decide tradeoff between exploring new areas and exploiting learned info to reach optimum

→ *“exploration, exploitation tradeoff”*
→ codified in the ***acquisition function***



Courtesy Johannes Kirschner (ETH Zurich)

Process of Bayesian Optimization

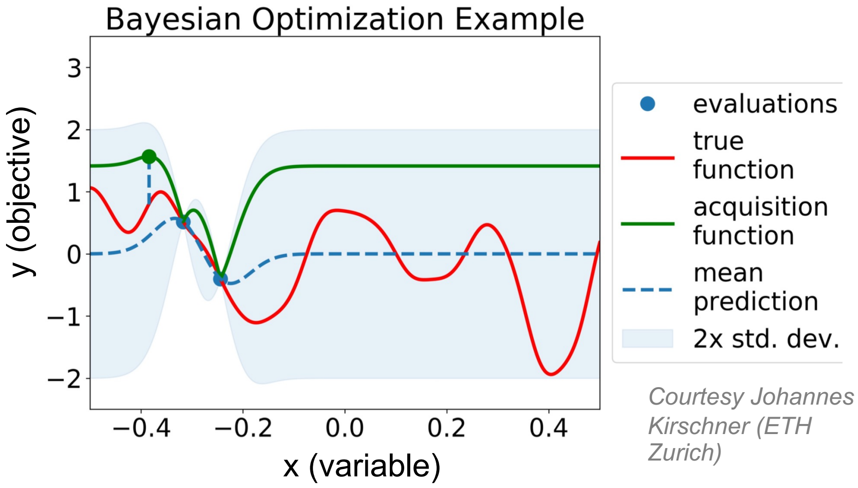
Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points

Use model predictions and uncertainty to guide search for optimum while sampling

Decide tradeoff between exploring new areas and exploiting learned info to reach optimum

→ *“exploration, exploitation tradeoff”*
→ codified in the ***acquisition function***



Courtesy Johannes Kirschner (ETH Zurich)

Process of Bayesian Optimization

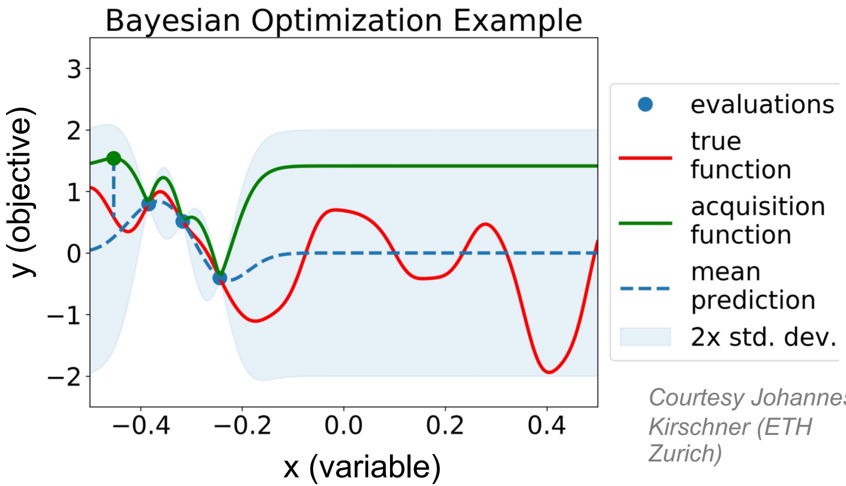
Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points

Use model predictions and uncertainty to guide search for optimum while sampling

Decide tradeoff between exploring new areas and exploiting learned info to reach optimum

→ *"exploration, exploitation tradeoff"*
→ codified in the ***acquisition function***



Courtesy Johannes Kirschner (ETH Zurich)

Process of Bayesian Optimization

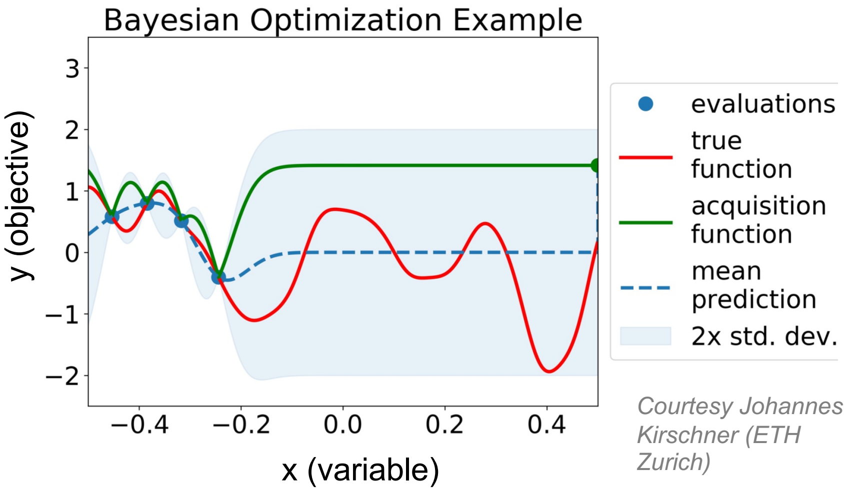
Build probabilistic model
→ e.g. Gaussian Process model

Iteratively refit model while sampling new points

Use model predictions and uncertainty to guide search for optimum while sampling

Decide tradeoff between exploring new areas and exploiting learned info to reach optimum

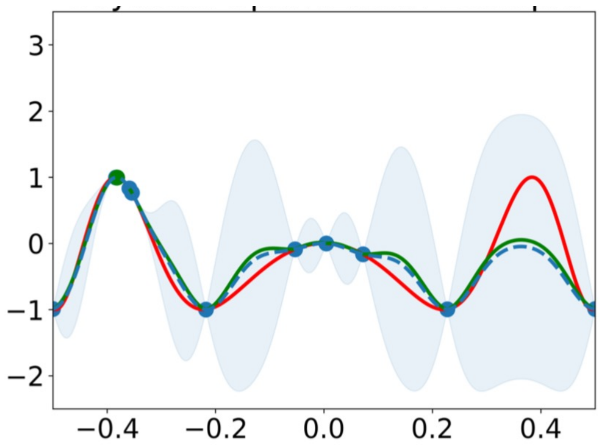
→ *“exploration, exploitation tradeoff”*
→ codified in the ***acquisition function***



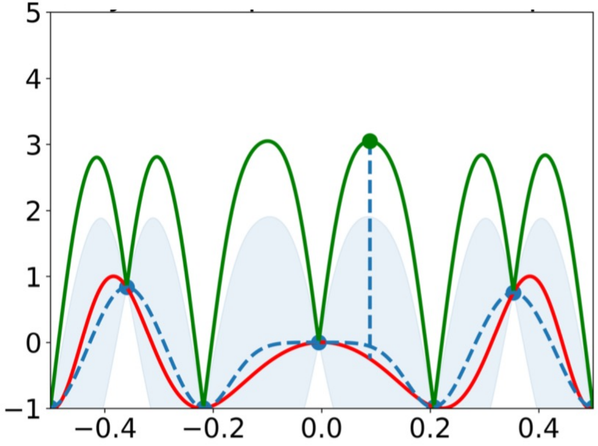
Courtesy Johannes Kirschner (ETH Zurich)

exploration-exploitation tradeoff at the extremes

Figs courtesy Johannes Kirschner (ETH Zurich)



exploit → *may miss better optima*



explore → *uniform sampling*

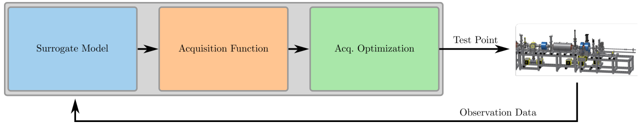
Acquisition function hyperparameter called *beta* or *kappa* controls this tradeoff

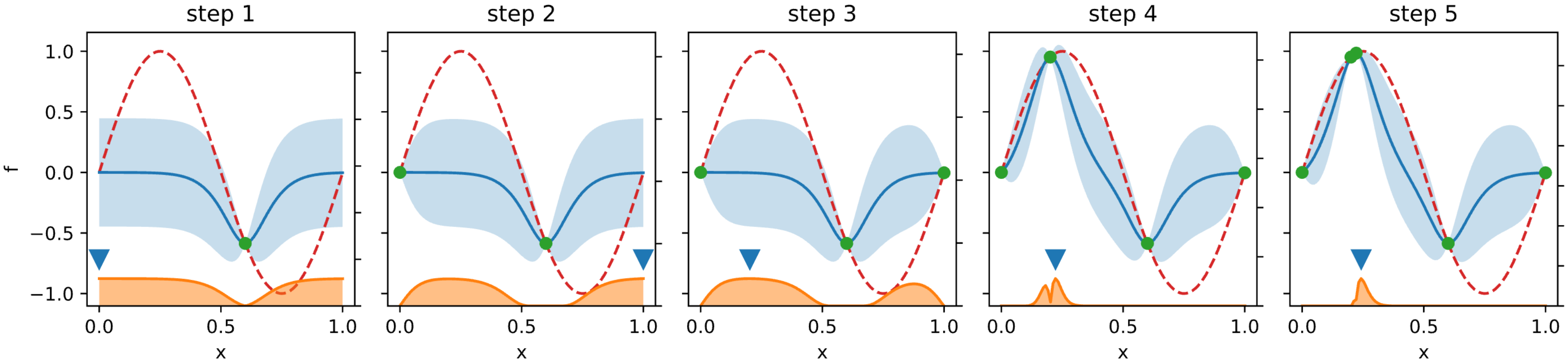
Single Objective Optimization

$$\mathrm{EI}(\boldsymbol{x}) = \mathbb{E}[\max(f(\boldsymbol{x}) - f^*)]$$

$$\boldsymbol{x}_{t+1} = \mathrm{argmax}_{\boldsymbol{x}} \mathrm{EI}(\boldsymbol{x})$$

(Assumes maximization)

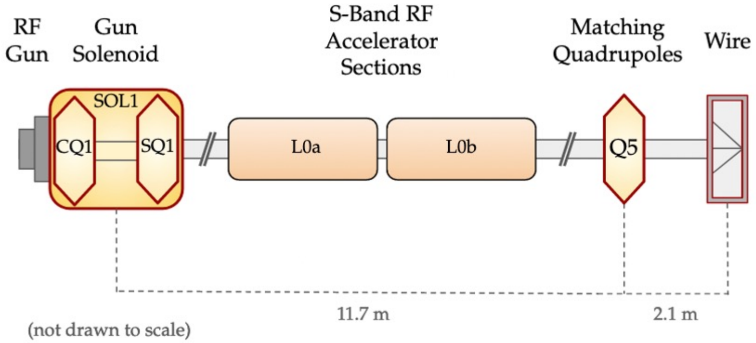


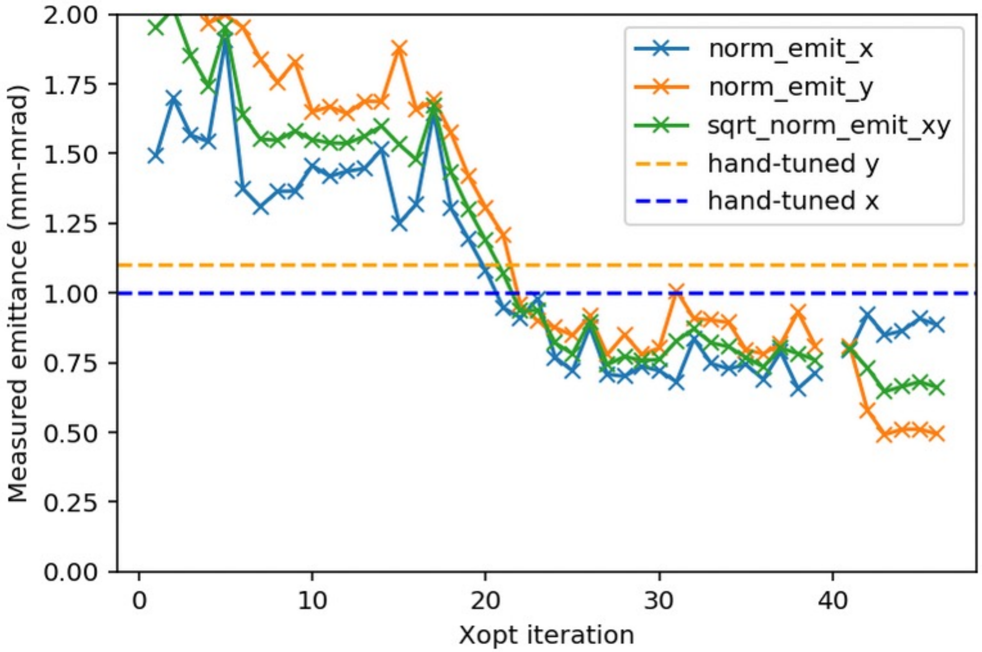


- The model accuracy improves in the region of interest
- Initially model uncertainty is high at domain boundaries, BO likes to sample those
- Helpful if the acquisition function is differentiable → use gradient descent to optimize

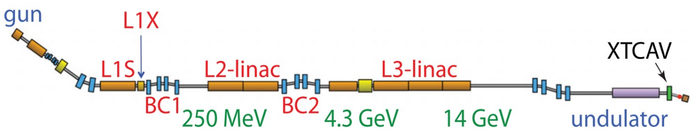
Example: LCLS-II Injector Emittance

- BO to optimize several magnets (SOL1, SOL2, SQ1, SQ2, CQ1, CQ2)
- Upper confidence bound acquisition function





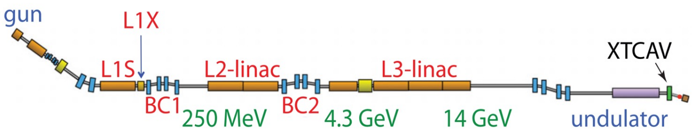
Incorporating Physics-Informed Priors



Can use a prior based on expected physics

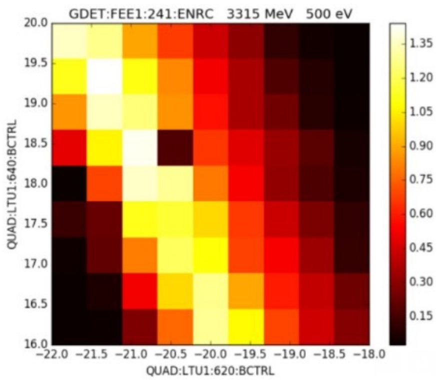
- BO at LCLS→ tune quadrupoles maximize FEL pulse energy
- Make GP kernel informed by how quads correlate with FEL

Incorporating Physics-Informed Priors



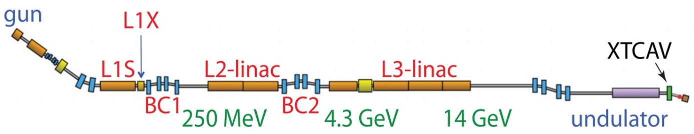
Can use a prior based on expected physics

- BO at LCLS→ tune quadrupoles maximize FEL pulse energy
- Make GP kernel informed by how quads correlate with FEL



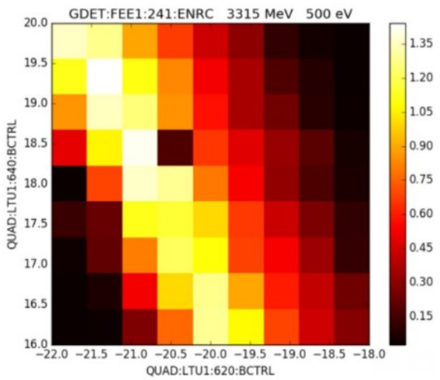
Measured FEL: quads 620 and 640 are adjacent so must be anti-correlated

Incorporating Physics-Informed Priors

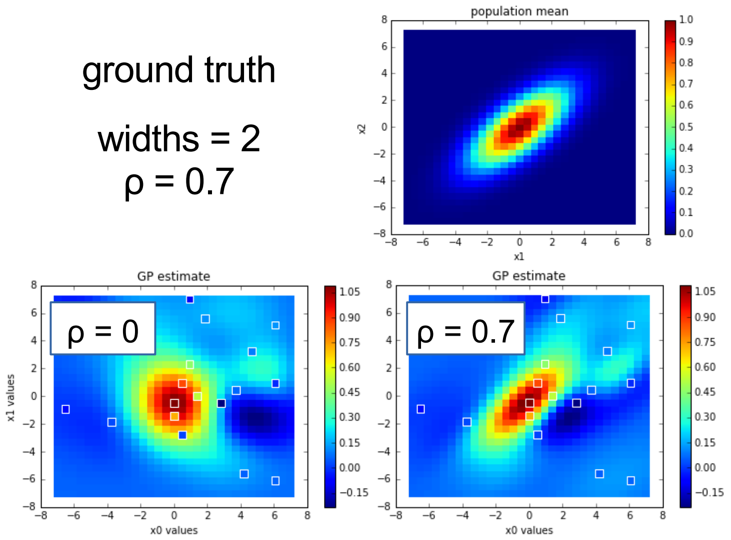


Can use a prior based on expected physics

- BO at LCLS→ tune quadrupoles maximize FEL pulse energy
- Make GP kernel informed by how quads correlate with FEL

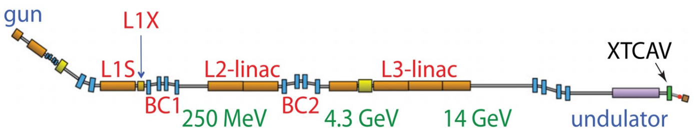


Measured FEL: quads 620 and 640 are adjacent so must be anti-correlated



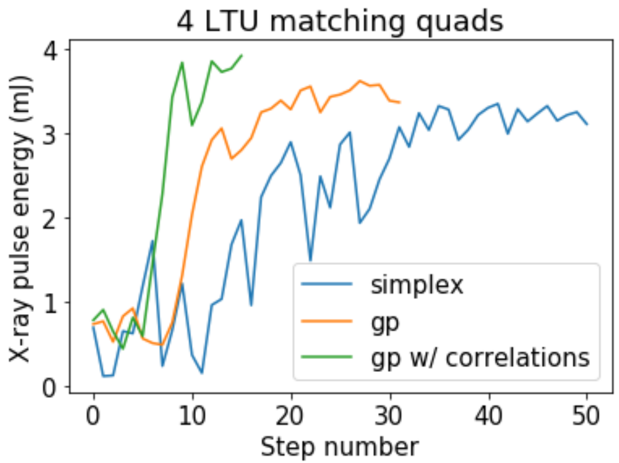
regression on the ***same samples***

Incorporating Physics-Informed Priors



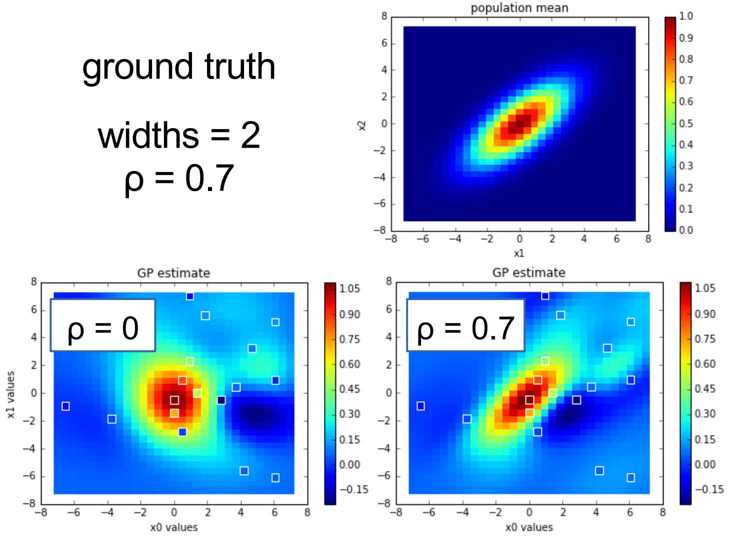
Can use a prior based on expected physics

- BO at LCLS→ tune quadrupoles maximize FEL pulse energy
- Make GP kernel informed by how quads correlate with FEL



ground truth

widths = 2
ρ = 0.7

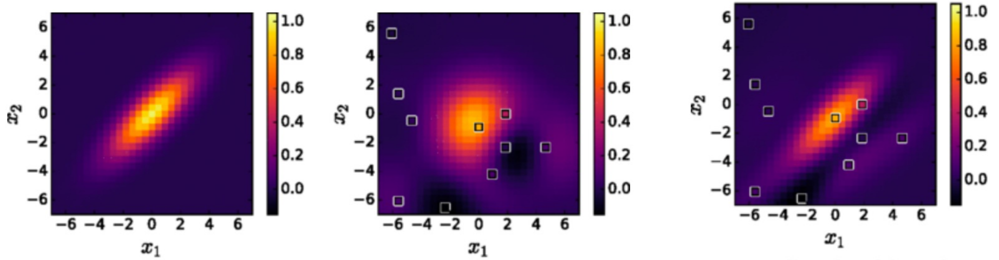


regression on the ***same samples***

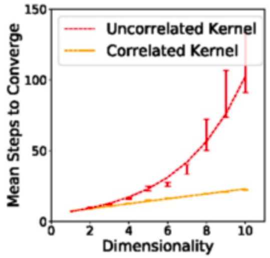
Incorporating Physics-Informed Priors

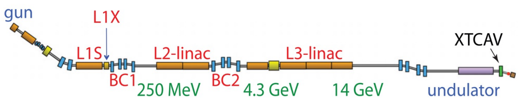
J. Duris et al., PRL, 2020
A. Hanuka, et al., PRAB, 2021

→ design Gaussian Process kernel from expected correlations between inputs (e.g. quads)



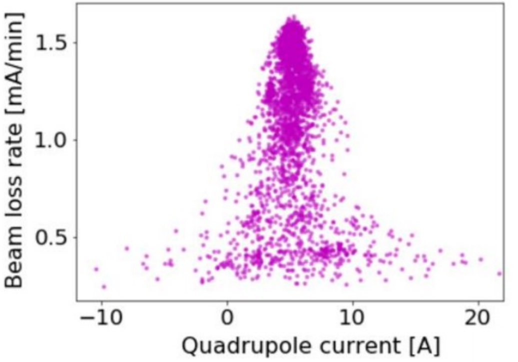
(a) Ground truth (b) Isotropic kernel (c) Correlated kernel

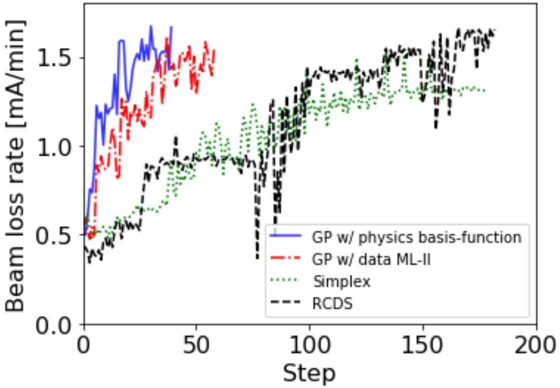


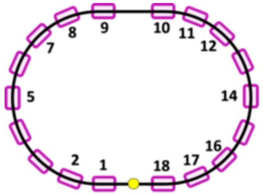


FEL tuning @LCLS

→ take the Hessian of model at expected optimum to get the correlations







vertical emittance tuning @SPEAR3

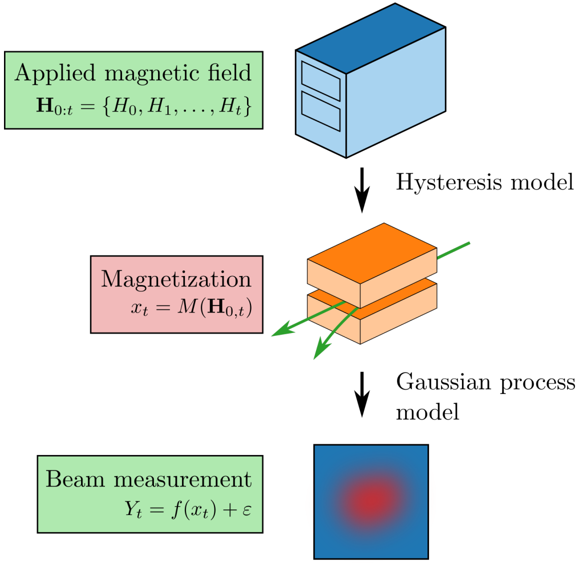
No measured data needed ahead of time, just a physics model

Including correlation between inputs enables increased sample-efficiency and results in faster optimization
→ kernel-from-Hessian enables easy computation of correlations even in high dimension

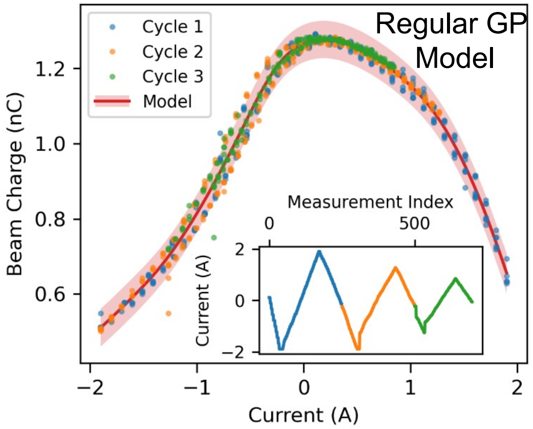
Differentiable Physics + GP Modeling

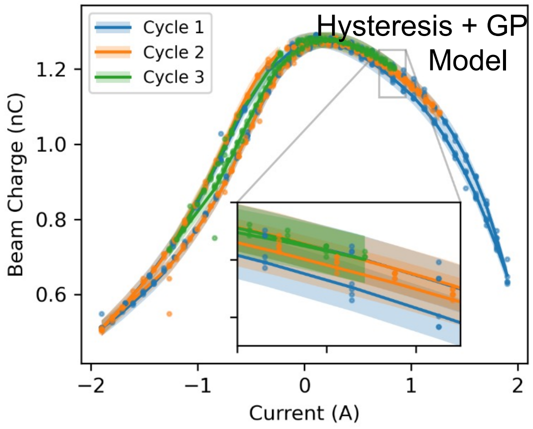
Magnetic hysteresis has been a major impediment to high-precision tuning → *historically required standardization of magnets*

New modeling approach combining classical Preisach model and a Gaussian Process

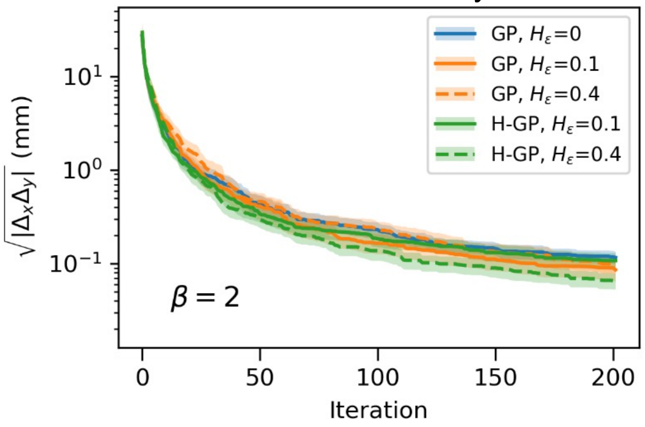


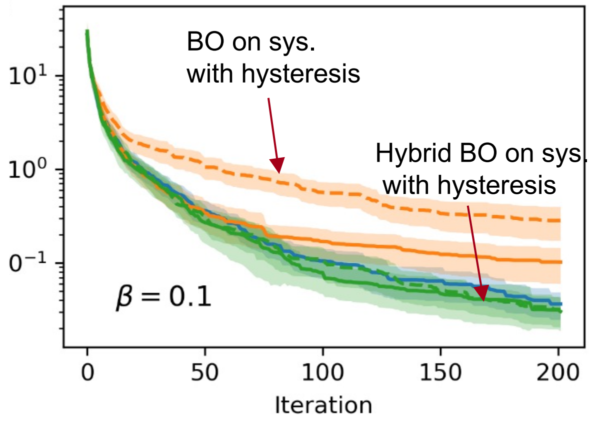
R. Roussel, et al., PRL, 2022





Joint modeling of hysteresis and beam propagation is more accurate and enables in-situ hysteresis characterization



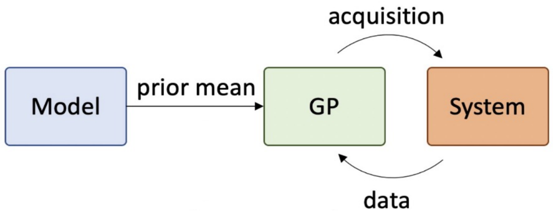


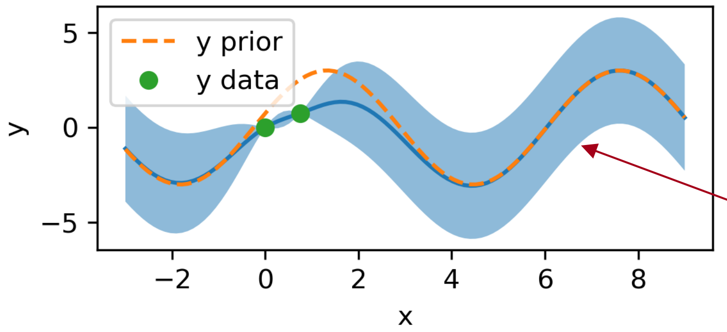
Higher-precision optimization possible when including hysteresis effects in model

Promising example showing the power of differentiable physics and ML models to enable high-precision characterization and control with minimal data.

Combining GP Modeling with Neural Networks

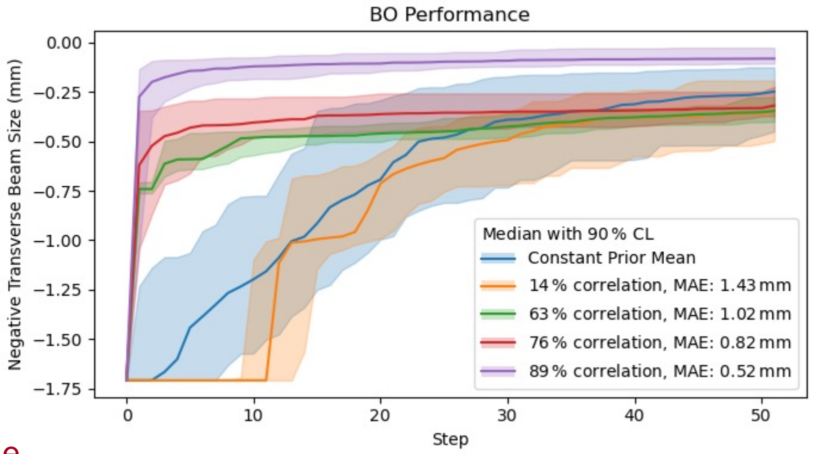
We can specify a **prior mean** function to bias the model where data does not exist





NN prior improves optimization performance even with limited model accuracy → *don't need a perfect model*

LCLS injector surrogate

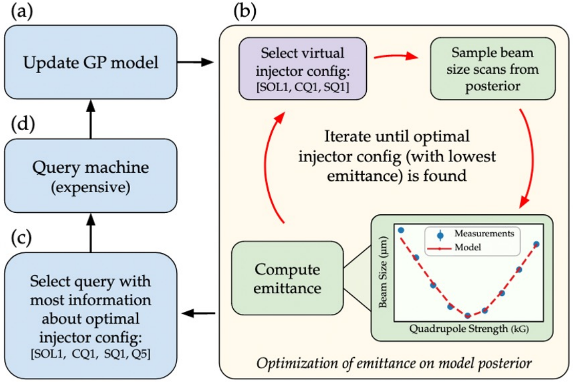


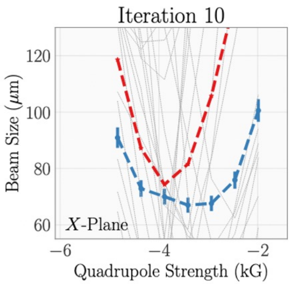
Xopt example

NeurIPS proceeding: https://arxiv.org/abs/2211.09028

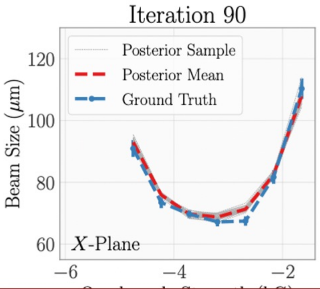
Efficient Emittance Optimization with Partial Measurements

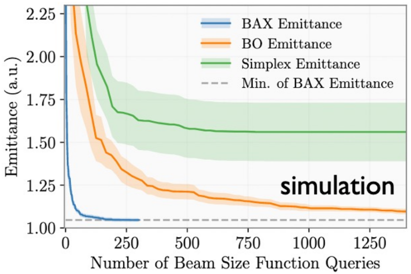
- **Instead of tuning on costly emittance measurements directly: learn a fast-executing model online for beam size while optimizing** → *learn on direct observables (e.g. beam size); do inferred "measurements" (e.g. emittance)*
- **New algorithmic paradigm leveraging "Bayesian Algorithm Execution" (BAX) for 20x speedup in tuning**

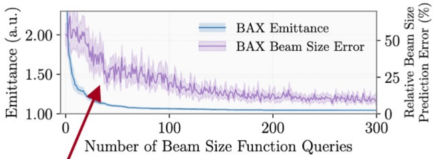




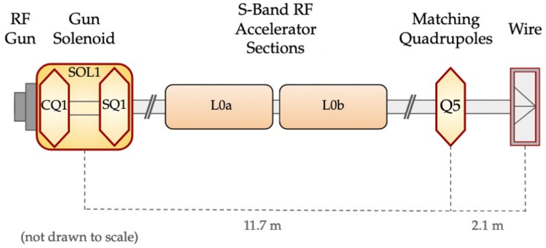
model is learned on-the-fly

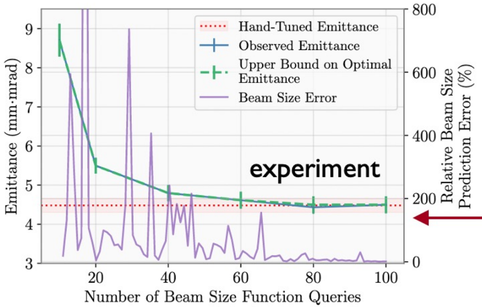






Convergence of beam size prediction error gives practical indicator of optimization convergence *(no need to do direct emittance measurement until the end)*





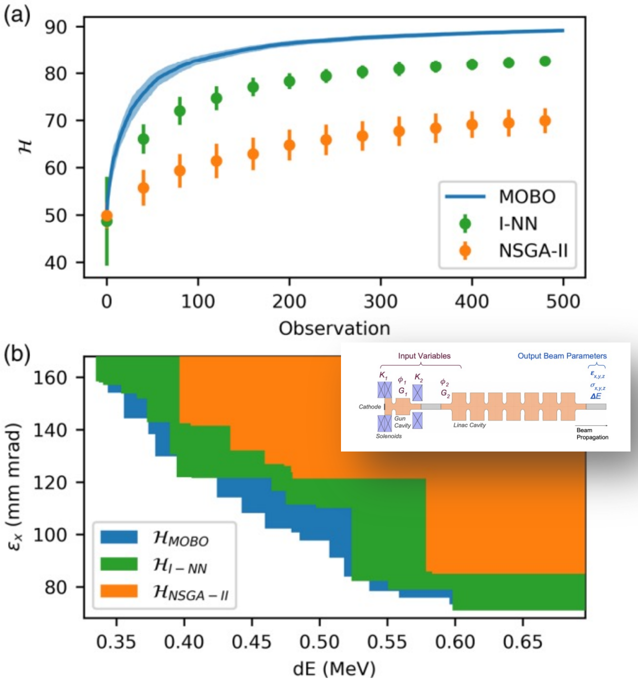
Found equivalent quality to hand-tuning in about 70 iterations *(estimate this would take a few minutes with computationally optimized routine)*

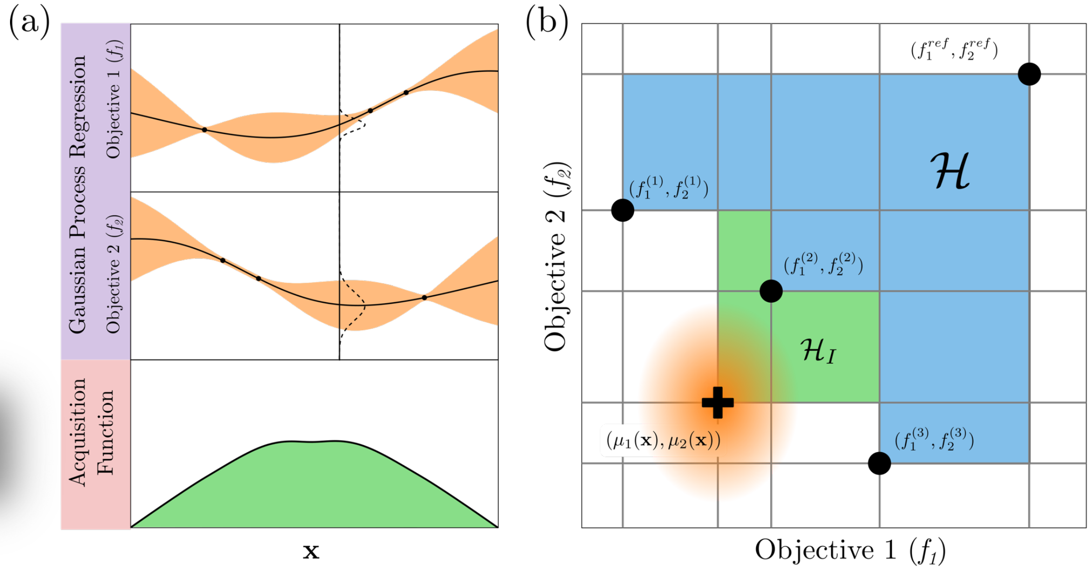
https://arxiv.org/abs/2209.04587

Paradigm shift in how tuning on indirectly computed beam measurements (such as emittance) is done, with 20x improvement over standard method for emittance tuning. → ***Now working to integrate into operations.***
→ ***Also now working to incorporate more informative global models /priors rather than learning the model from scratch each time.***

Multi-Objective Optimization

Determine the optimal trade-off between objectives: the Pareto front





$$\alpha_{EHVI}(\boldsymbol{\mu},\boldsymbol{\sigma},\mathcal{P},\boldsymbol{r}) := \int_{\mathbb{R}^P} \mathrm{HVI}(\mathcal{P},\boldsymbol{y},\boldsymbol{r})\cdot\boldsymbol{\xi}_{\mu,\boldsymbol{\sigma}}(\boldsymbol{y})d\boldsymbol{y}$$

Xopt example

Roussel et. al. PRAB **2021**

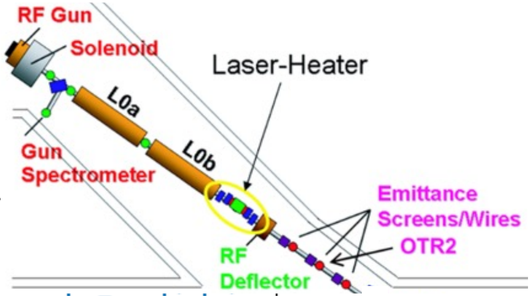
Example: Ideal Tradeoffs for LCLS Injector

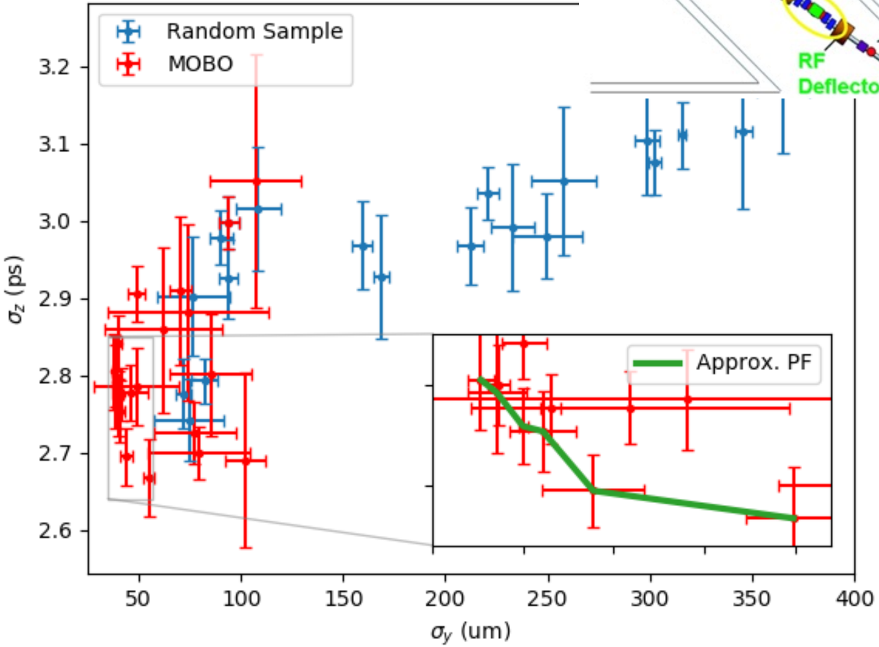
Objectives:
- Minimize longitudinal bunch length
- Minimize vertical bunch size

Tuning variables:
- Solenoid strength
- Skew quad strength
- Normal quad strength

Started with random sampling of input space, then ran Multi-Objective Bayesian Optimization for 25 iterations

LCLS Injector

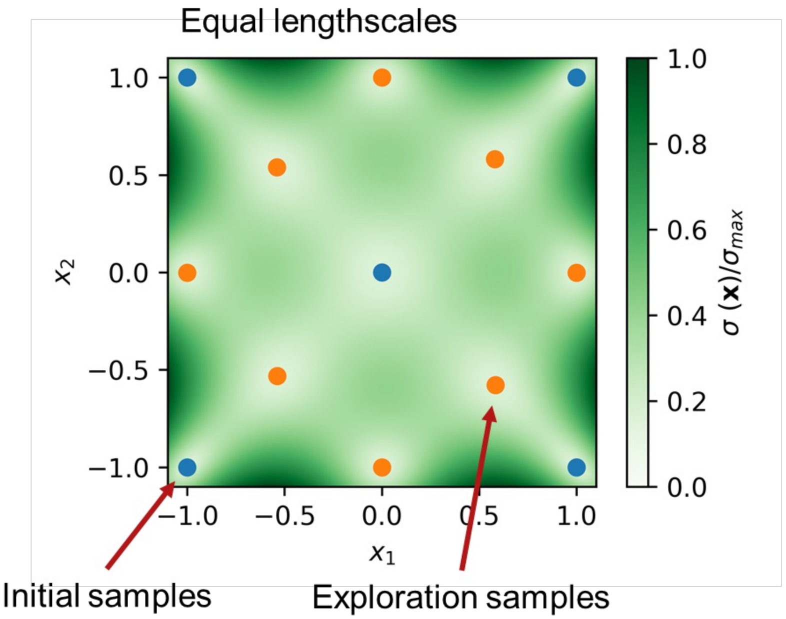


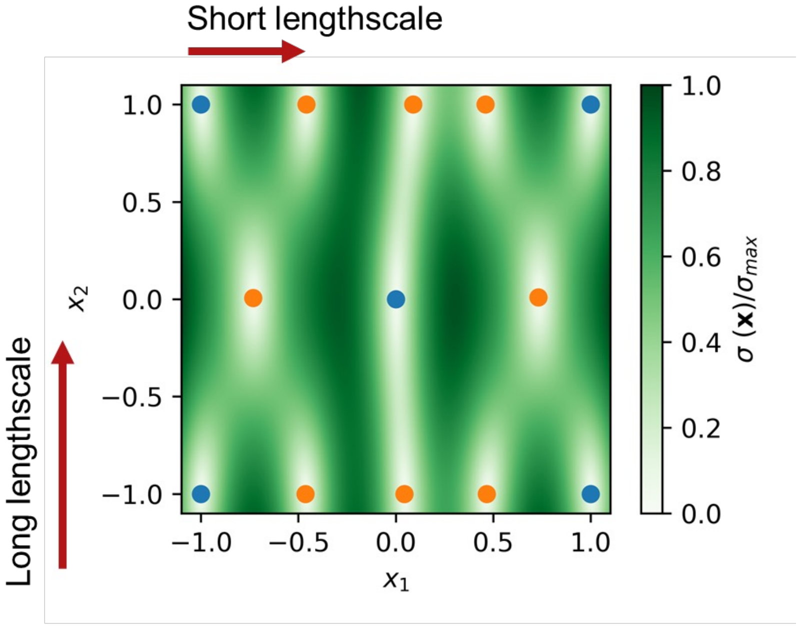


Autonomous Characterization – Bayesian Exploration

If the function changes more rapidly along one axis, sample more points along that axis!

$$\alpha(\boldsymbol{x}) = \sigma(\boldsymbol{x})$$





Xopt example

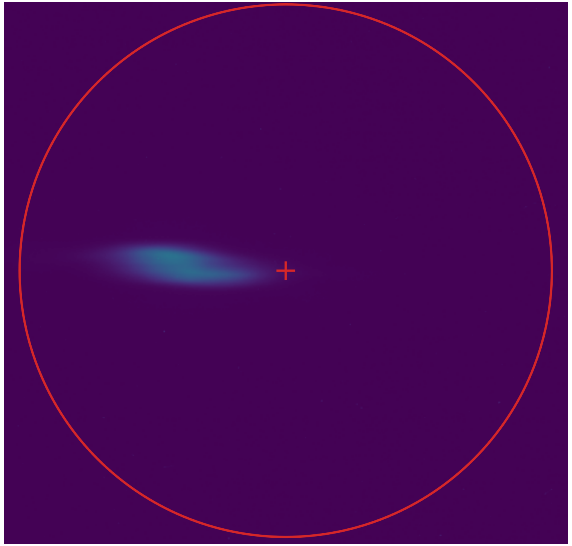
Roussel et. Al. *Nat. Comm.* **2021**

Incorporating Constraints

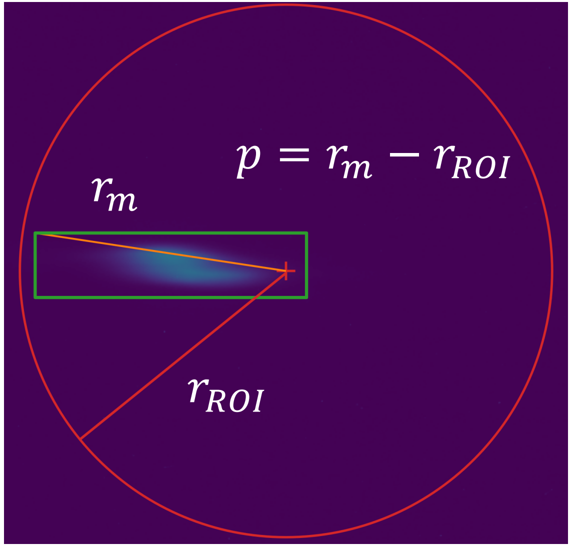
Example: We want to ensure during measurements that the beam stays within a ROI.

- Define a **smoothly varying** penalty function to act as a constraint

Define a circular ROI



Measure maximum distance from the ROI center to bounding box corners.



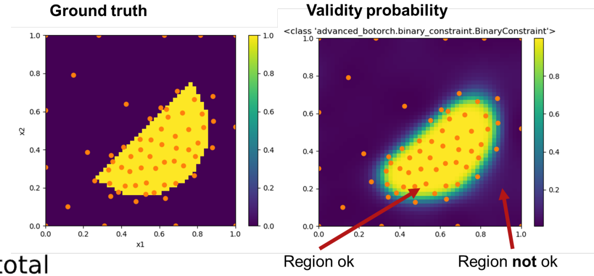
Constraint:
$p \leq 0$

Other examples: Beam losses, dark current production, emittance, etc.

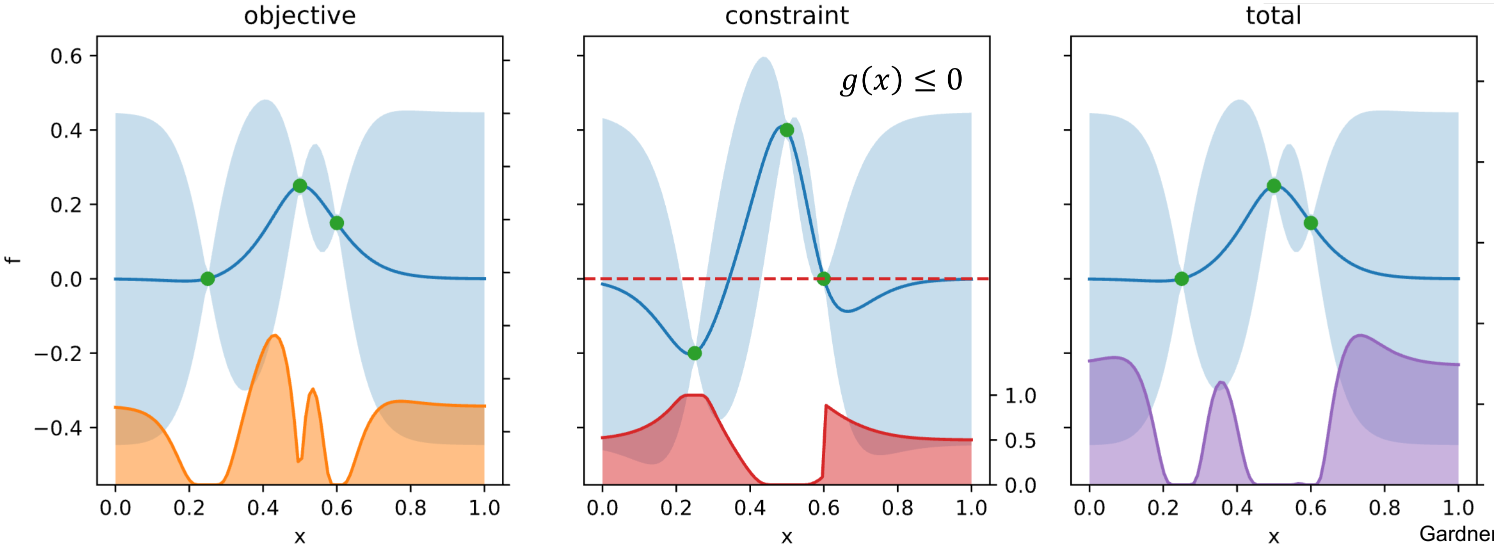
Incorporating Constraints

Weight the acquisition function by the probability that constraints are satisfied

$$\hat{\alpha}(x) \rightarrow \alpha(x) \prod_i p[g_i(x) \leq h_i]$$



2D Example



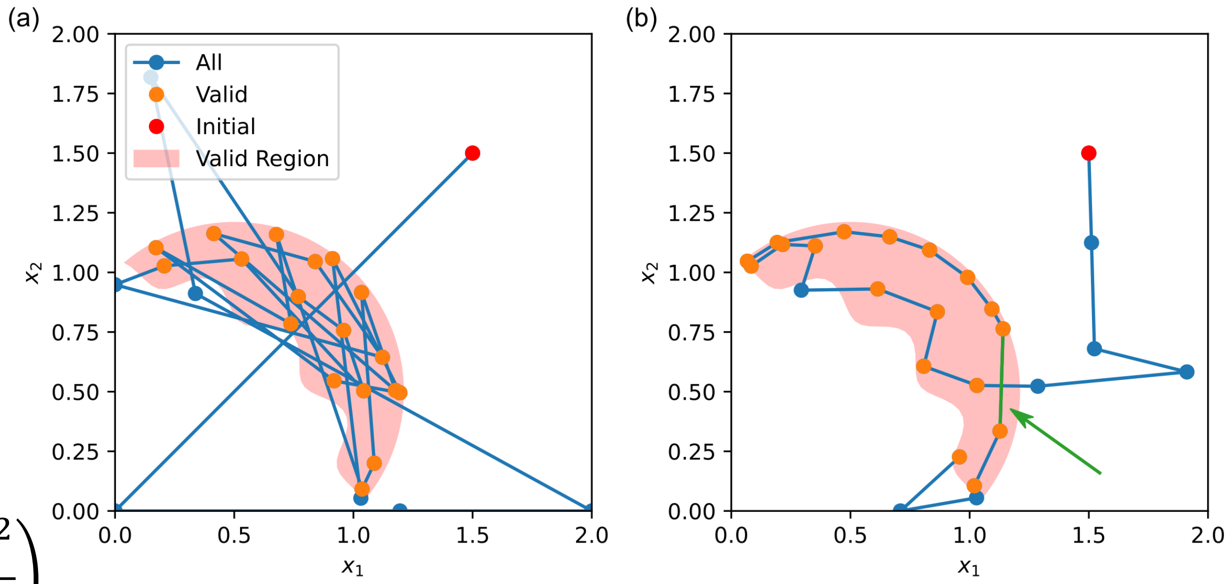
Gardner et. al. ICML 2014

Proximal Biasing

Poor optimization behavior for experimental beamlines

Weight the acquisition function by travel distance → better than hard limits

$$\hat{\alpha}(x) \to \alpha(x) \exp\left(-\frac{(x - x_0)^2}{2\sigma^2}\right)$$



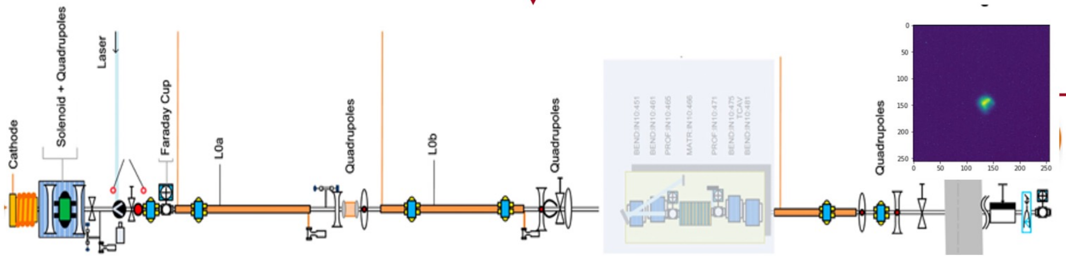
Roussel et. Al. *Nat. Comm.* **2021**

Efficient Characterization of FACET-II Injector

Automatic Exploration
(constrained to useful values of emittance and match)

Comprehensive ML Models of Injector

Setting changes on 10 variables (solenoid, bucking coil, corrector quads and matching quads)

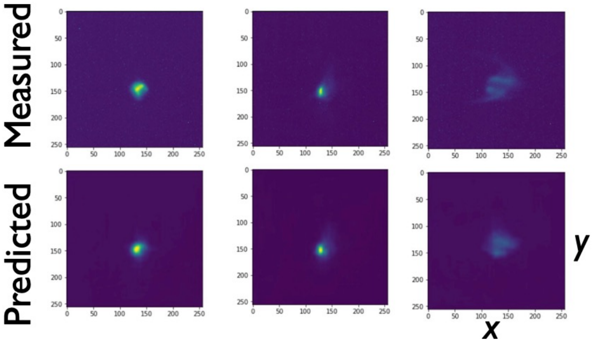


x-y emit, match, and beam images

FACET-II Injector

- Used Bayesian Exploration for efficient high-dimensional characterization (10 variables) of emittance and match at 700pC: **2 hrs for 10 variables compared to 5 hrs for 4 variables with N-D parameter scan**
- Data was used to train neural network model of injector response predicting x-y beam images. GP ML model from exploration predicts emittance and match.
- **Example of integrated cycle between characterization, modeling, and optimization → now want to extend to larger system sections and new setups**

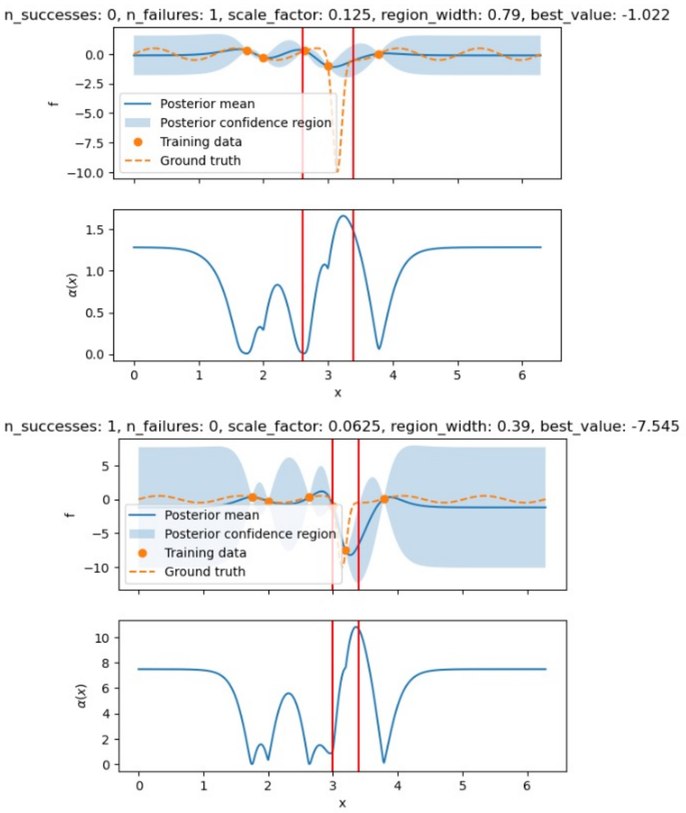
transverse phase space



Use of Bayesian exploration to generate training data was sample-efficient, reduced burden of data cleaning, and resulted in a well-balanced distribution for the training data set over the input space. ML models were immediately useful for optimization.

Trust Region Bayesian Optimization (TuRBO)

- Bayesian optimization tends to **prioritize exploration** in order to find global optima
- **Restrict search region** to local area around best observation
- Expand / contract **“trust” region** based on algorithm successes / failures on-the-fly
- Helps find local extrema in high dimensional optimization problems

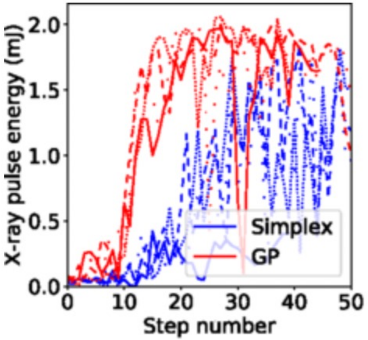


Xopt example

Many successes with Bayesian Optimization in accelerators

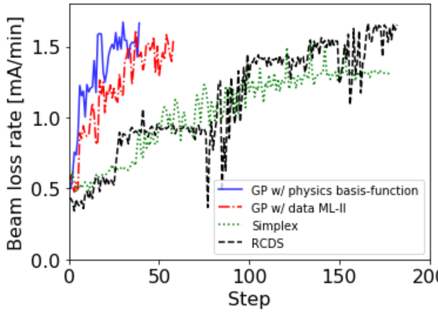
(+ improvements)

FEL pulse energy tuning at LCLS



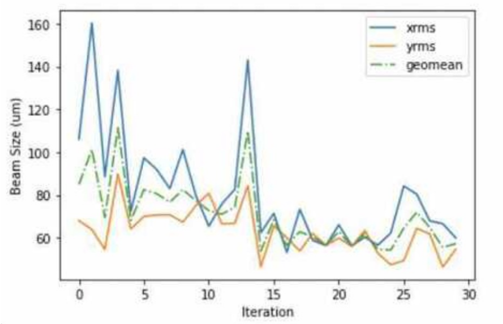
Duris et. al. PRL , **2020**

Loss rate tuning at SPEAR3

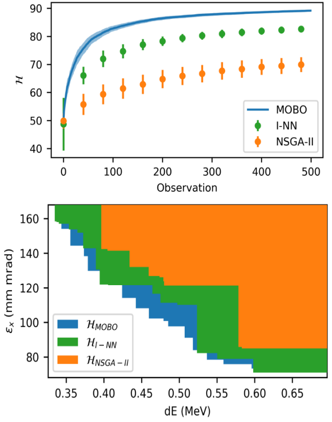


Hanuka et. al. PRAB , **2021**

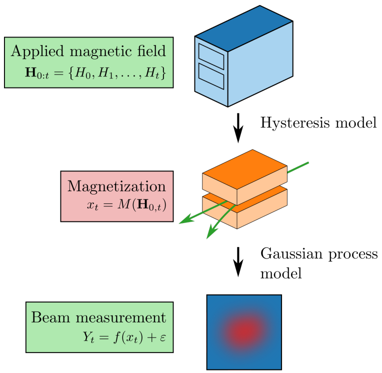
Sextupole tuning for IP at FACET-II



Multi-objective Bayesian Optimization

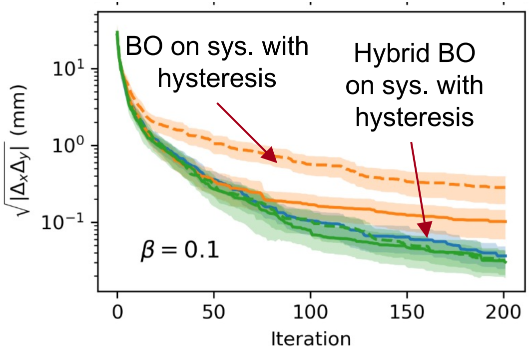


Roussel et. al. PRAB , **2021**

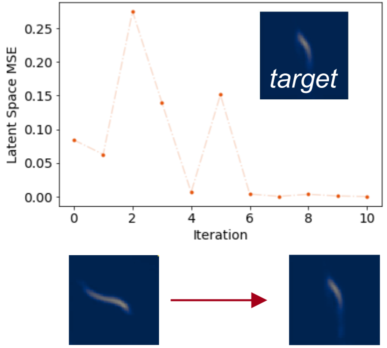


Roussel et. al. PRL , **2022**

Higher-precision optimization possible when including hysteresis effects in model

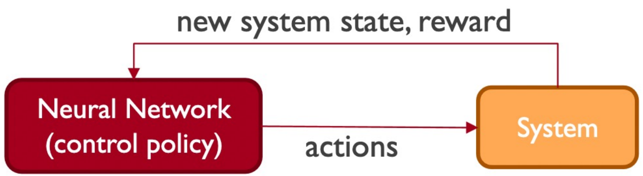


Longitudinal phase space tuning on LCLS



Algorithms being implemented/distributed in Xopt: https://github.com/ChristopherMayes/Xopt

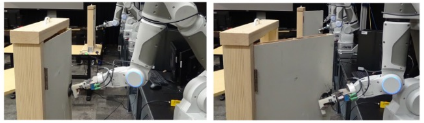
Deep Reinforcement Learning



- Control policy maps states to actions
- Policy is learned over time based on performance *(quantified by the "reward")*
- Neural network enables use of diverse signal types *(e.g. scalars, images, time series)*
- Often learns a system model simultaneously *(map states + actions to expected reward)*

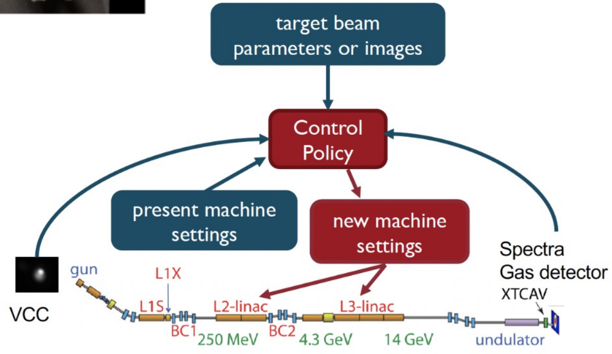
Nagabandi, et al., 2019

Gu, et al., 2016



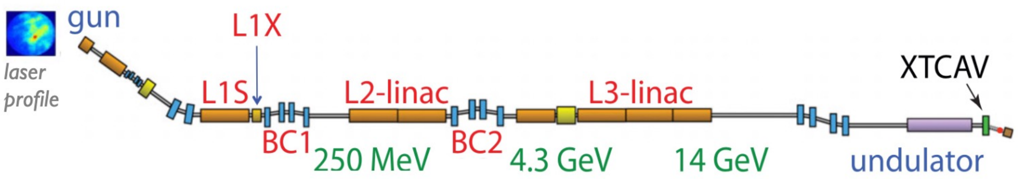
Appeal for accelerator control:

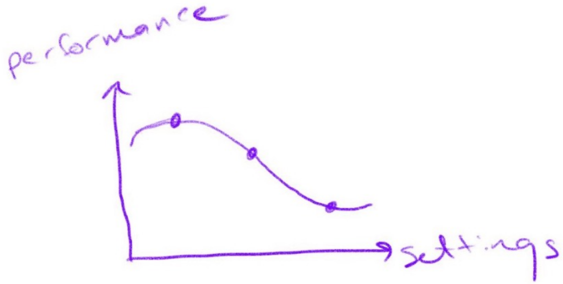
- Suitable for large, nonlinear systems
- Exploit machine-wide sensitivities + directly use complicated diagnostic information
- Leverage information from past observations
- Transfer between similar designs
- Well-established in other fields (e.g. robotic control)
  → but accelerators have unique challenges



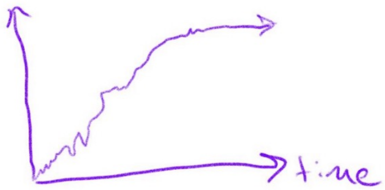
Deep RL is well-suited to accelerator control, but dedicated R&D is needed to bring it to full fruition

Can treat many high-level accelerator tuning problems as either time-dependent or time-independent…





"search for optimal settings"



"game to take actions that maximize performance over time"

as machine drifts over time → reoptimize, or keep playing

Some problems need to be treated as time-dependent…

RF electron gun at the Fermilab Accelerator Science and Technology (FAST) facility

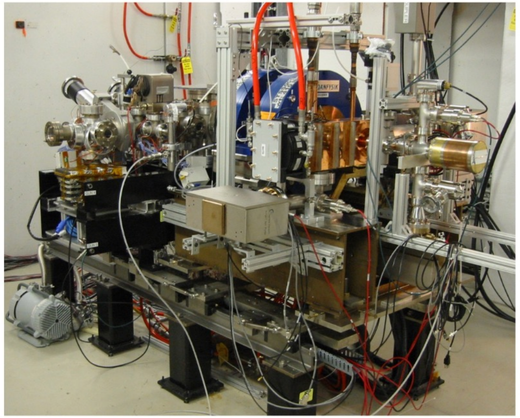
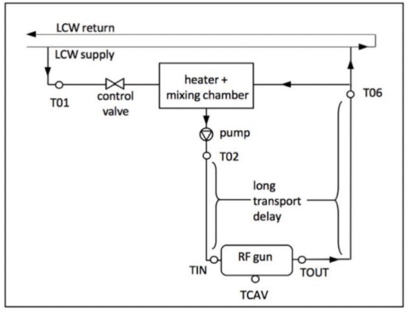
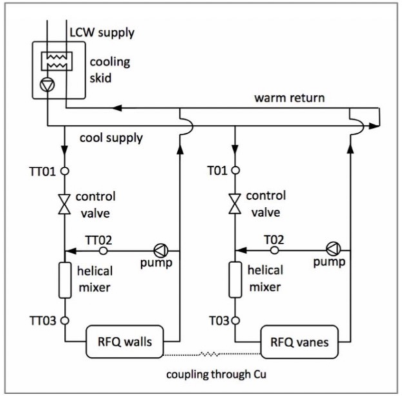


Photo: P. Stabile

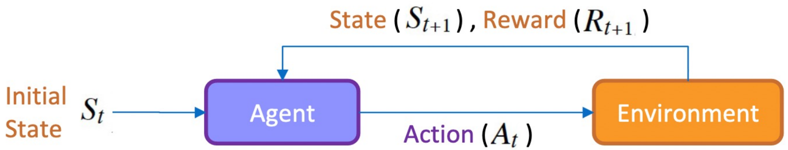


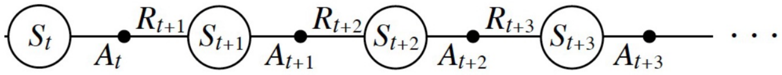
Radio frequency quadrupole (RFQ) for the PIP-II Injector Test

Photo: J. Steimel



Basic Framing of an RL Problem





RL agent interacts with an **environment** over time → *goal is to maximize total returned reward*

State – system information at present time

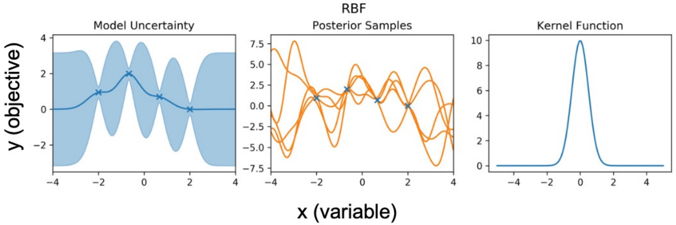
Action – a change the agent can make to the environment

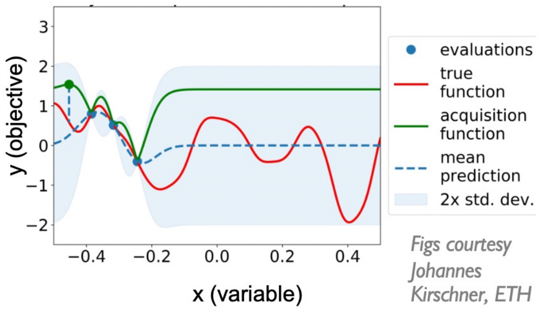
Reward – scalar return from the environment at present time

Episode – sequences of (state1 → action1 → state2 + reward2); ends on some terminal condition

Agent acts according to a **policy** (**π**) – determines actions to take based on observed state

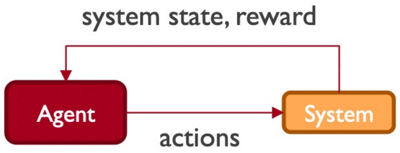
Bayesian Optimization

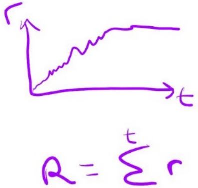




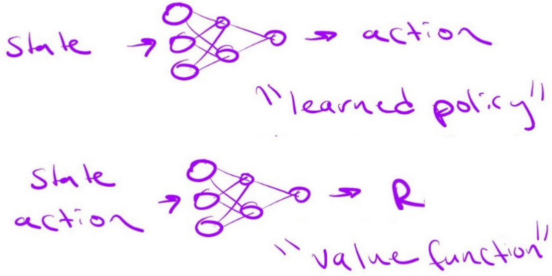
Select sample x → observe objective → refit surrogate model → use model predictions and uncertainty to choose next point according to an acquisition functions

Reinforcement Learning





Many ways to construct agent that learns from reward:



Observe state → take action according to a control policy → observe reward → update policy or value function

Analogous concepts, different terminology and usually different setting:

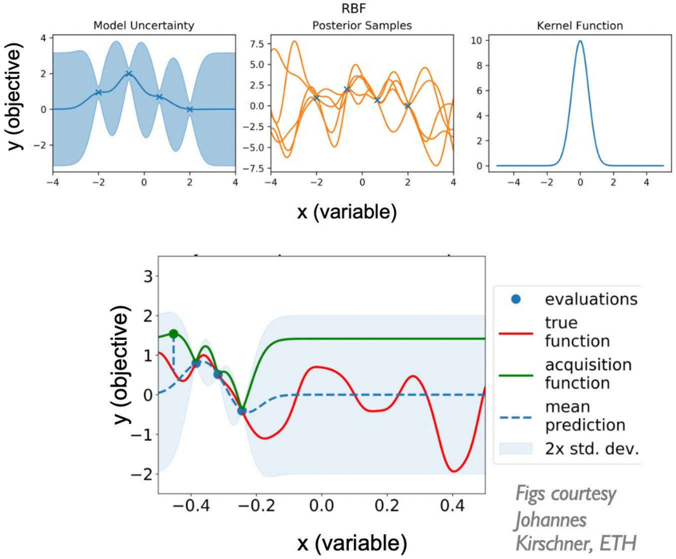
objective → reward

surrogate model → value function

acquisition function → policy

acquire new sample → take an action

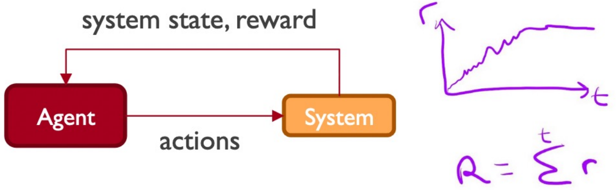
Bayesian Optimization



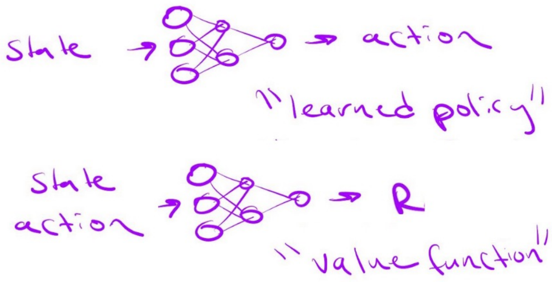
Figs courtesy Johannes Kirschner, ETH

Select sample x → observe objective → refit surrogate model → use model predictions and uncertainty to choose next point according to an acquisition functions

Reinforcement Learning



Many ways to construct agent that learns from reward:



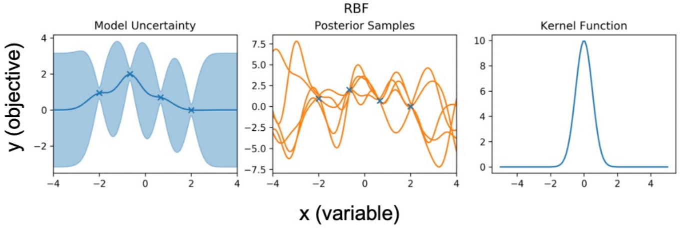
Observe state → take action according to a control policy → observe reward → update policy or value function

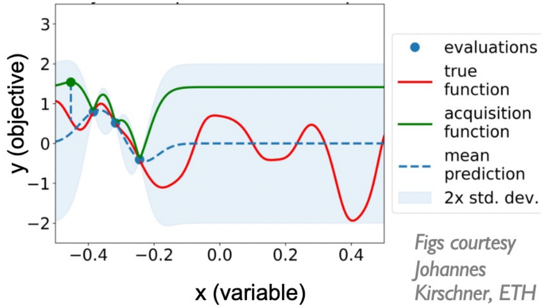
"deep RL" uses neural networks

Analogous concepts, different terminology and usually different setting:

objective → reward
surrogate model → value function
acquisition function → policy
acquire new sample → take an action

Bayesian Optimization

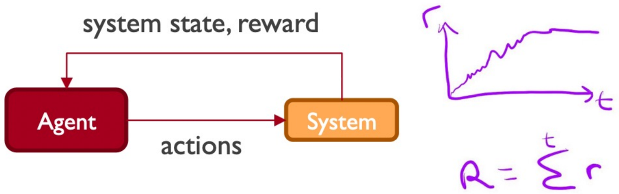




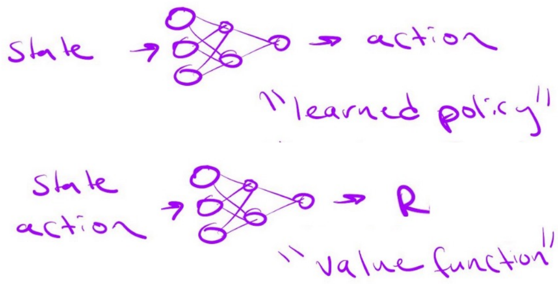
Figs courtesy Johannes Kirschner, ETH

Select sample x → observe objective → refit surrogate model → use model predictions and uncertainty to choose next point according to an acquisition functions

Reinforcement Learning



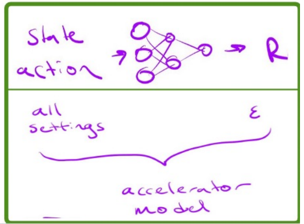
Many ways to construct agent that learns from reward:



Observe state → take action according to a control policy → observe reward → update policy or value function

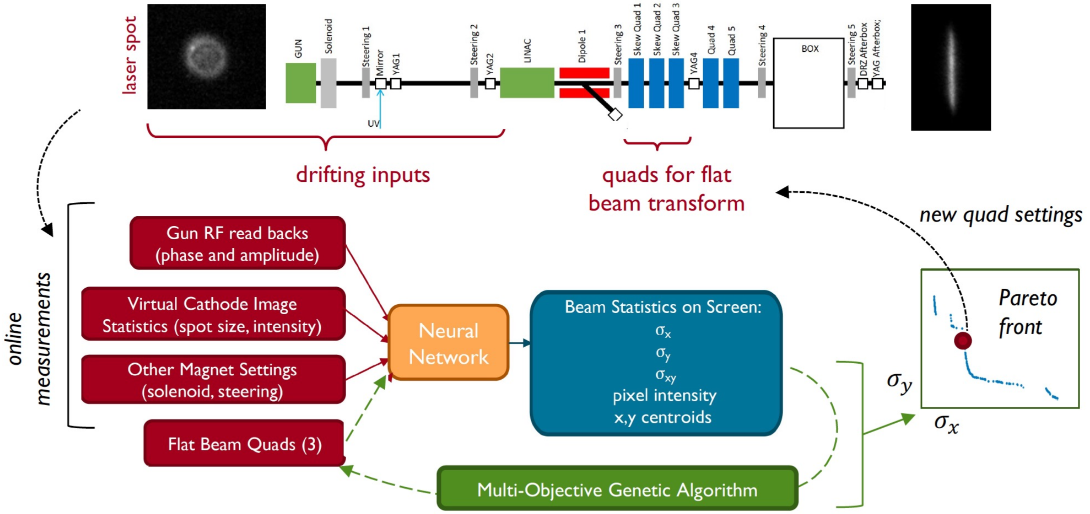
Analogous concepts, different terminology and usually different setting:

objective → reward
surrogate model → value function
acquisition function → policy
acquire new sample → take an action

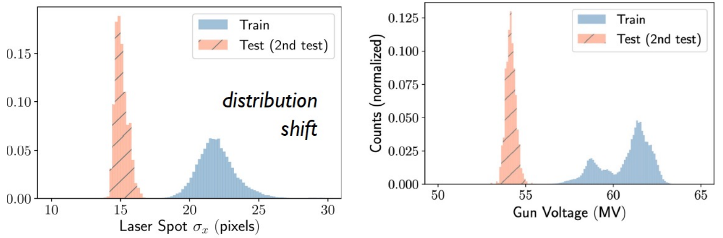


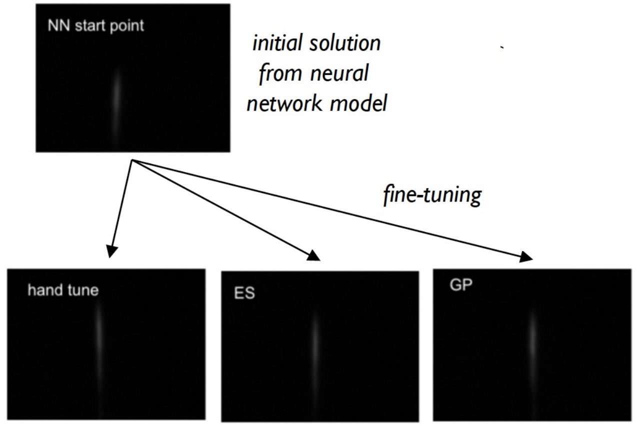
Recall Example from Accelerator Lecture

E. Cropp et al., in preparation

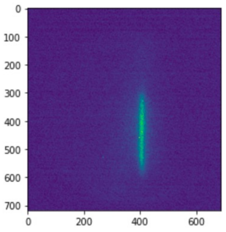


Can work even under distribution shift





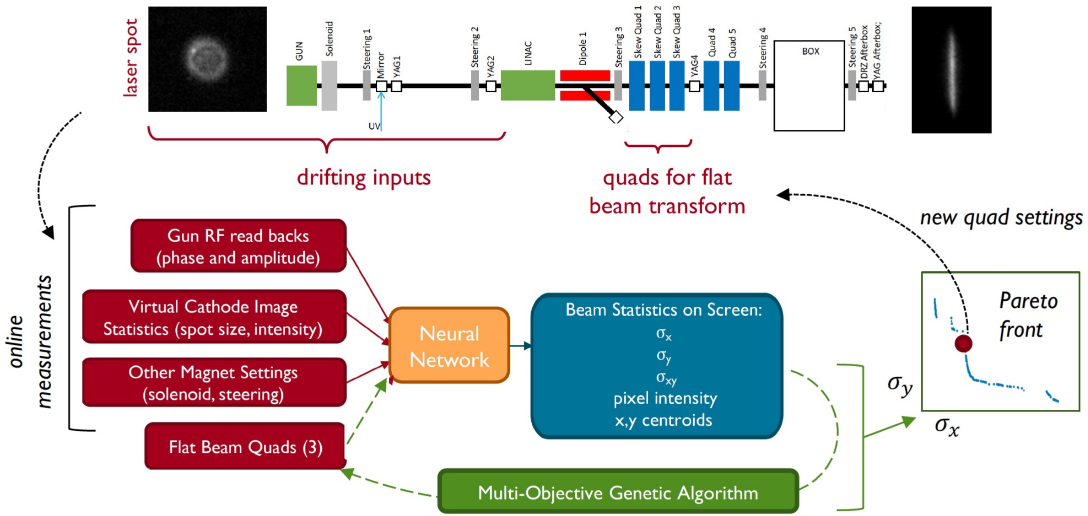
- Round-to-flat beam transforms are challenging to optimize → 2019 study explored ability of a learned model to help
- Trained neural network model to predict fits to beam image, based on archived data
- Tested online multi-objective optimization over model (3 quad settings) given present readings of other inputs
- Used as warm start for other optimizers

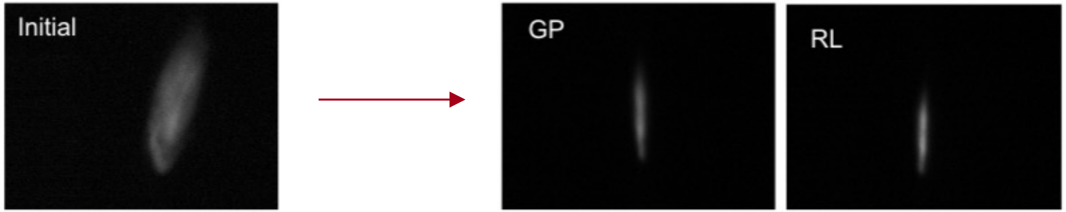


Hand-tuning in seconds vs. tens of minutes

Boost in convergence speed for other algorithms

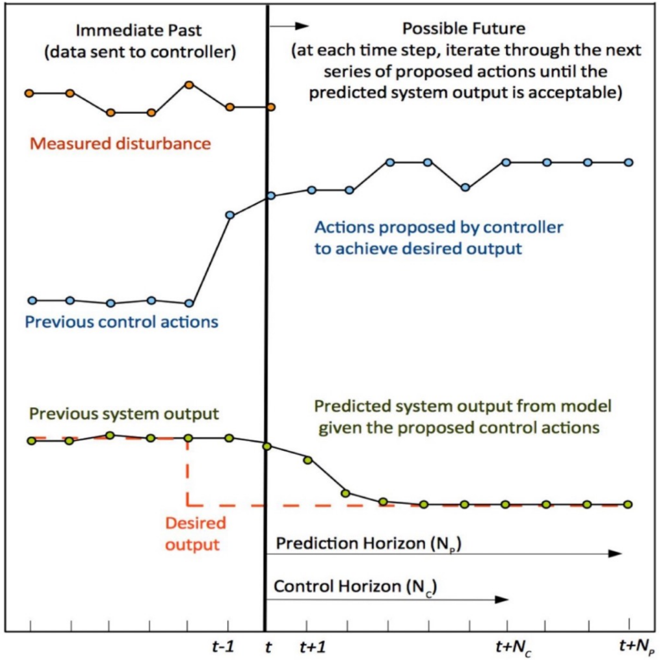
Example: RL on the same system





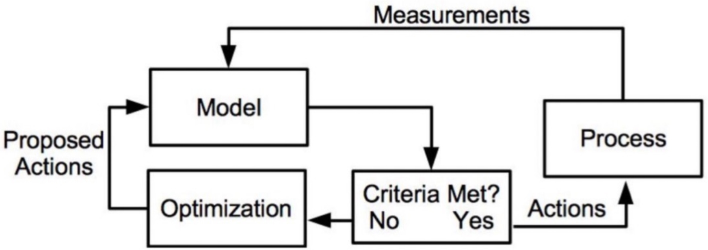
- Used learned NN model as a fast-executing training environment for RL control policy (Deep Deterministic Policy Gradients)
- Then tested on accelerator with/without retraining the policy
- In principle capable of taking both larger jumps and fine-tuning
- Had fastest convergence out of algorithms tested once trained, but required substantial overhead in training

Model Predictive Control

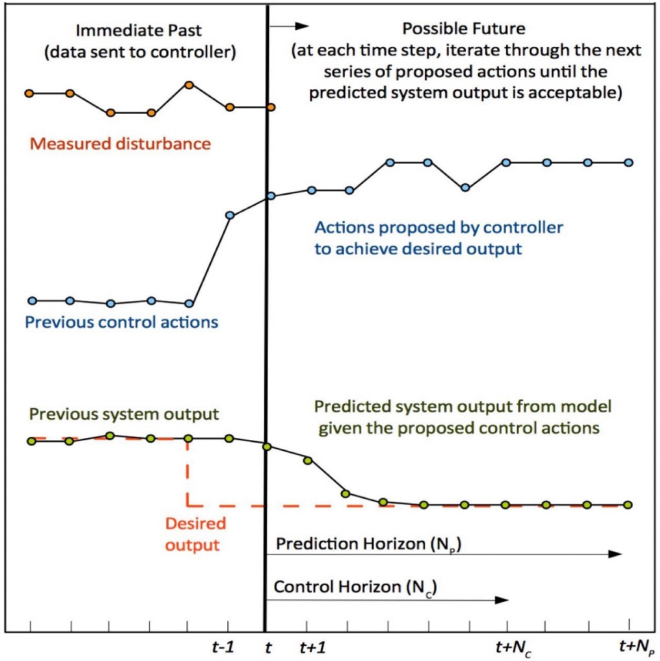


Basic concept:

1. Use a predictive model to assess the outcome of possible future actions
2. Choose the best series of actions
3. Execute the first action
4. Gather next time step of data
5. Repeat

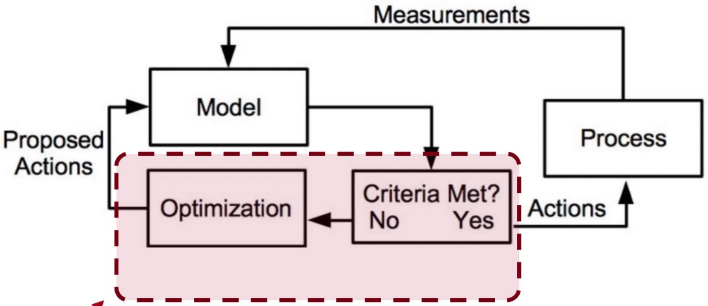


Model Predictive Control



Basic concept:

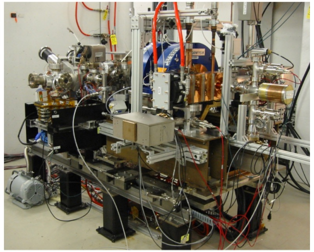
1. Use a predictive model to assess the outcome of possible future actions
2. Choose the best series of actions
3. Execute the first action
4. Gather next time step of data
5. Repeat

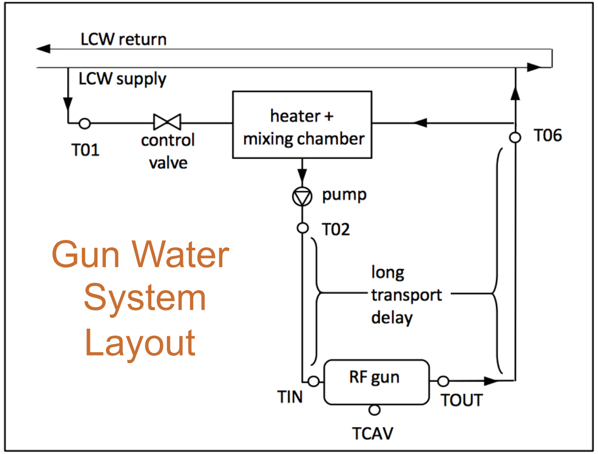


- **RL can be thought of as trying to learn the step for optimization over future time horizon** *(choose optimal action at time t to maximize reward / minimize cost over future)*
- **Without time-dependence, becomes optimization over an online system model** *(as we often use in accelerators)*

Example from FAST RF gun

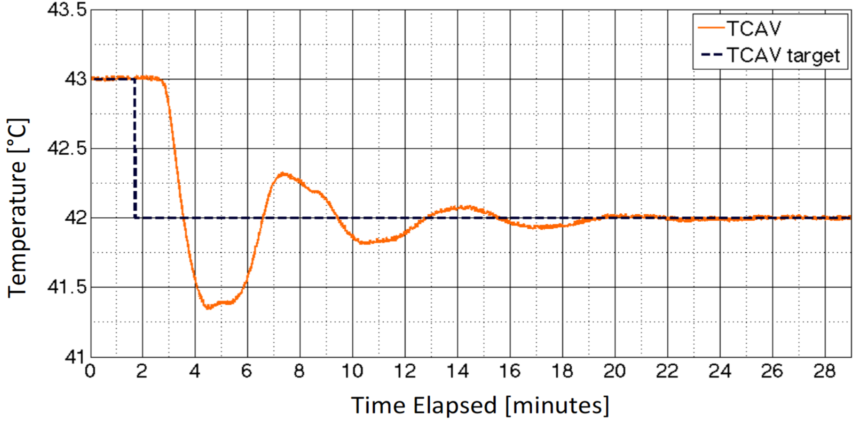
Resonant frequency controlled via temperature
Long transport delays and thermal responses
Two controllable variables: heater power + flow valve





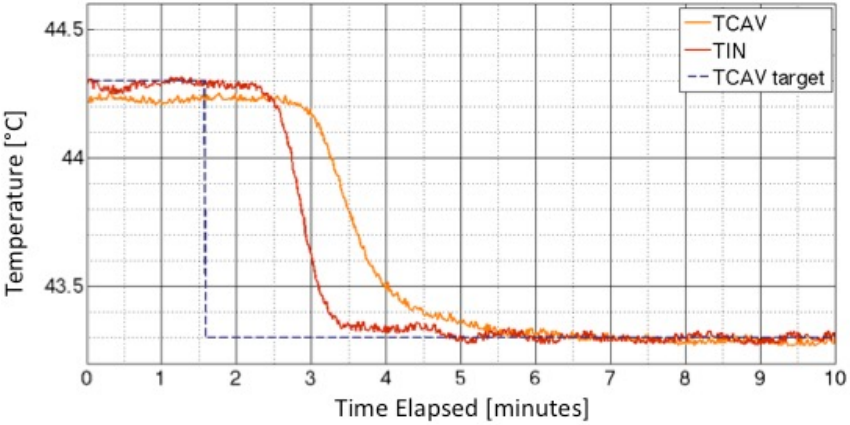
Applied model predictive control with a neural network model trained on measured data: ~ 5x faster settling time + no large overshoot

Existing Feedforward/PID Controller



Oscillations are largely due to the transport delays and water recirculation, not PID gains

Model Predictive Controller



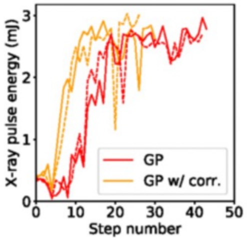
Similar techniques can be applied to cryogenic systems

Edelen, IPAC'15 ; Edelen, TNS, 2016

Both BO and RL have been used for online optimization/control of particle accelerators, with good success

Bayesian Optimization

FEL optimization (Duris et al. 2020, Kirschner et al. PMLR 2019)



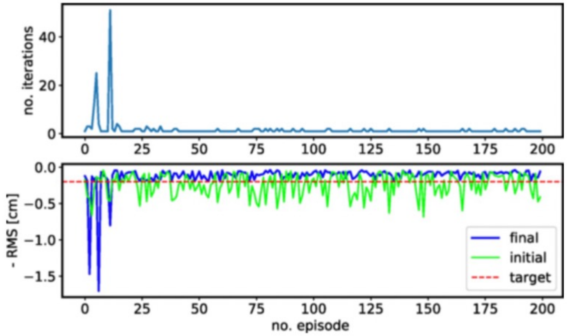
Emittance optimization at SPEAR3 (Hanuka et al. 2021)

Laser plasma accelerators (Jalas et al, PRL 2021, Shaloo et al Nature 2020)

Injection efficiency

Reinforcement Learning

Trajectory control (Kain et al., PRAB 2020)



FEL optimization (O'Shea, et al., 2020)

Magnet power supply (St. John et al., PRAB 2021)

Fast switching between FEL pulse energies (Edelen et al., NeurIPS 2017)

Choice largely depends on need:

- RL (and especially "deep" RL) is well-suited for continuous control, especially when a fast simulator exists for training
- BO is well-suited for optimization of new problems where there is little existing information
- For more detail on RL, see Auralee's USPAS lecture: https://slaclab.github.io/USPAS_ML/slides/Day9_Reinforcement.pdf

Summary

Bayesian optimization encompasses a broad set of flexible tools that are well-suited to solving complicated black-box optimization problems for both operation of instruments and design, particularly in setups where little to no previous information or data is available

Many improvements make Bayesian optimization more sample-efficient and suited to online optimization of experiment setups (e.g. smoother sampling, constraints, physics-informed priors)

Reinforcement Learning came out of a different setting (continuous control, robotics, etc) and is generally well-suited for time-dependent continuous control → in accelerators, it is being examined for both optimization and continuous control

For more details on BO/RL in the context of optimizing/controlling scientific instruments see:

- USPAS course on Optimization and ML for Particle Accelerators: https://slaclab.github.io/USPAS_ML/
- Many more RL pedagogy details and examples in: https://slaclab.github.io/USPAS_ML/slides/Day9_Reinforcement.pdf