Bayesian Optimization Tutorial

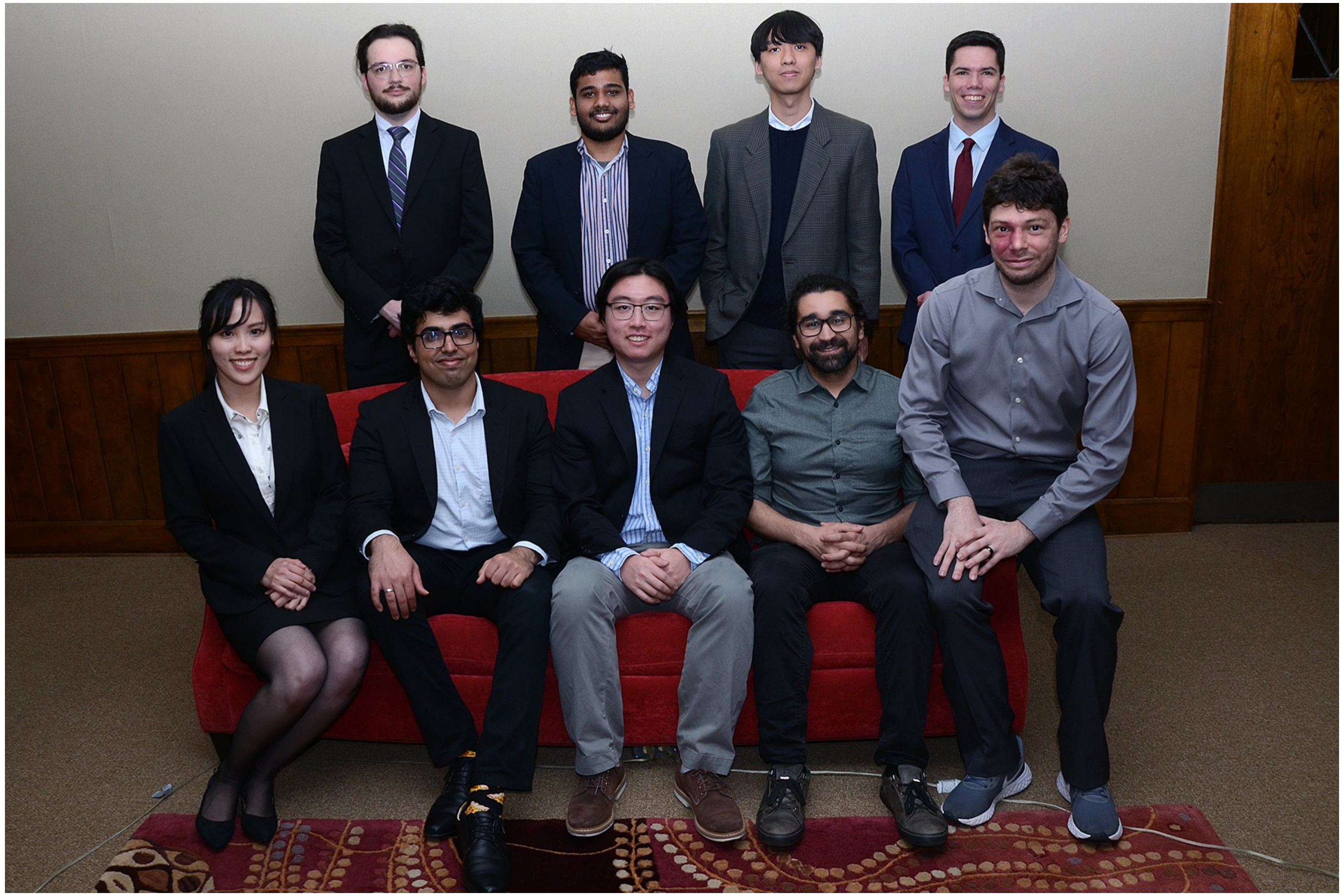
Why Go Beyond Traditional Optimization?

Joel Paulson
Assistant Professor, Department of Chemical and Biomolecular Engineering, The Ohio State University

Great Lakes PSE Student Workshop, 2023

For copies of slides & code, see
https://github.com/joelpaulson/Great_Lakes_PSE_Workshop_2023

Thank you to My Group for Help Developing Materials! (Especially the Code for the Modules)



What is an Optimization Problem?

decision (design) variables

$$\min_{x} \quad f(x)$$

objective function

$$\text{s.t.} \quad x \in \Omega$$

constraints (feasible region)

- Optimization problems are **pervasive** in every application domain
  - differentiate problems based on characteristics → determine what solver to use

- There are a huge number of available optimization algorithms; difficult to *a priori* know the best one but we can eliminate some options

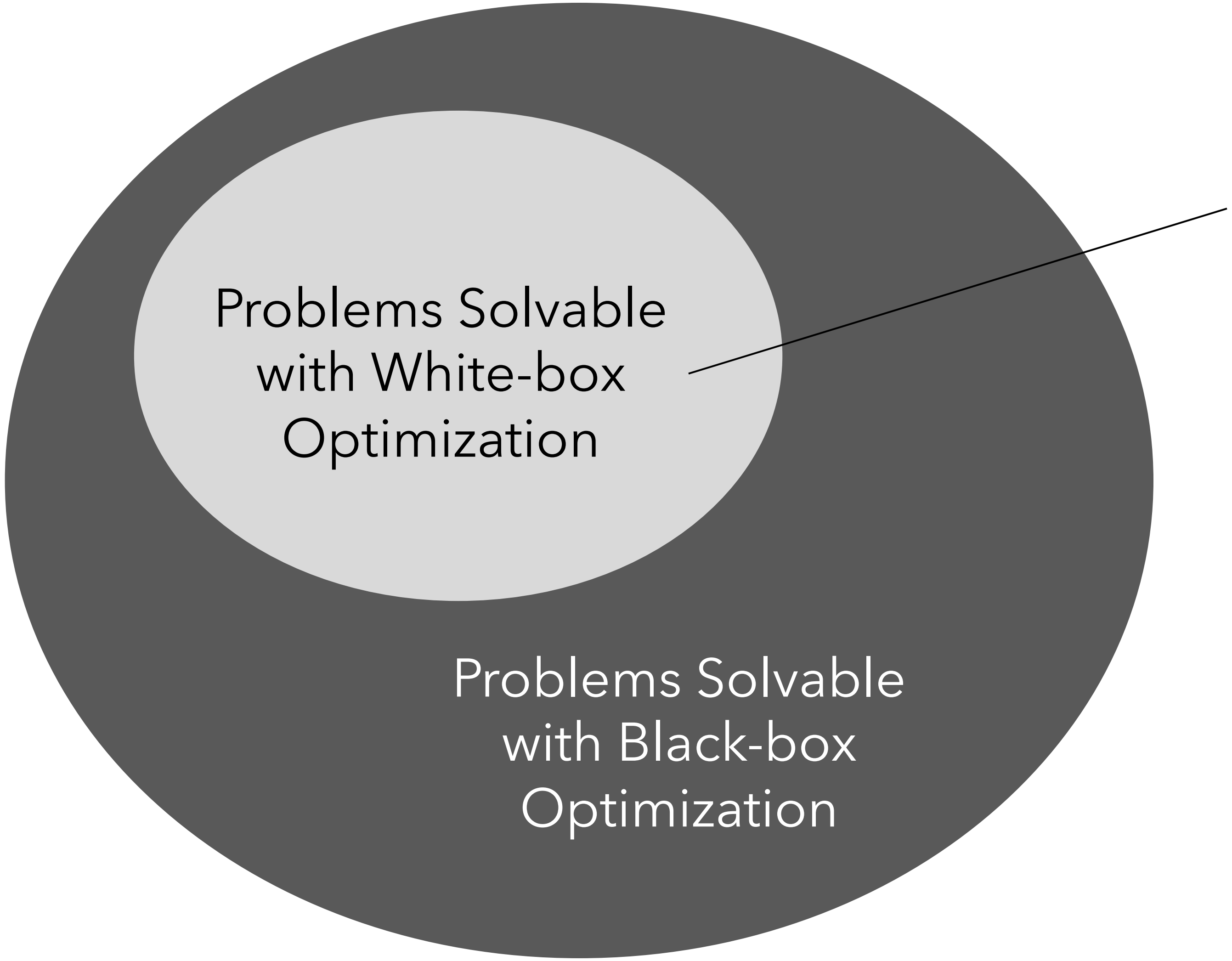
How to Classify Optimization Algorithms?

- A simple way to "partition" the algorithms into two major buckets are "white-box" and "black-box" (i.e., not white box)

- White-box means that we need an "equation-oriented model" of the system so that the mathematical structure of $f(x)$ and Ω satisfy certain important assumptions
  - The exact assumptions depend on the method, but they will typically require the functions to be differentiable and/or easy to build relaxations of them

- Any method that only requires evaluations of $f(x)$ and $x \in \Omega$ at specific points can then be classified as "black box"

How to Classify Optimization Algorithms?



Since white-box algorithms make stronger assumptions, they can only be used to tackle a subset of problems when compared to black-box algorithms

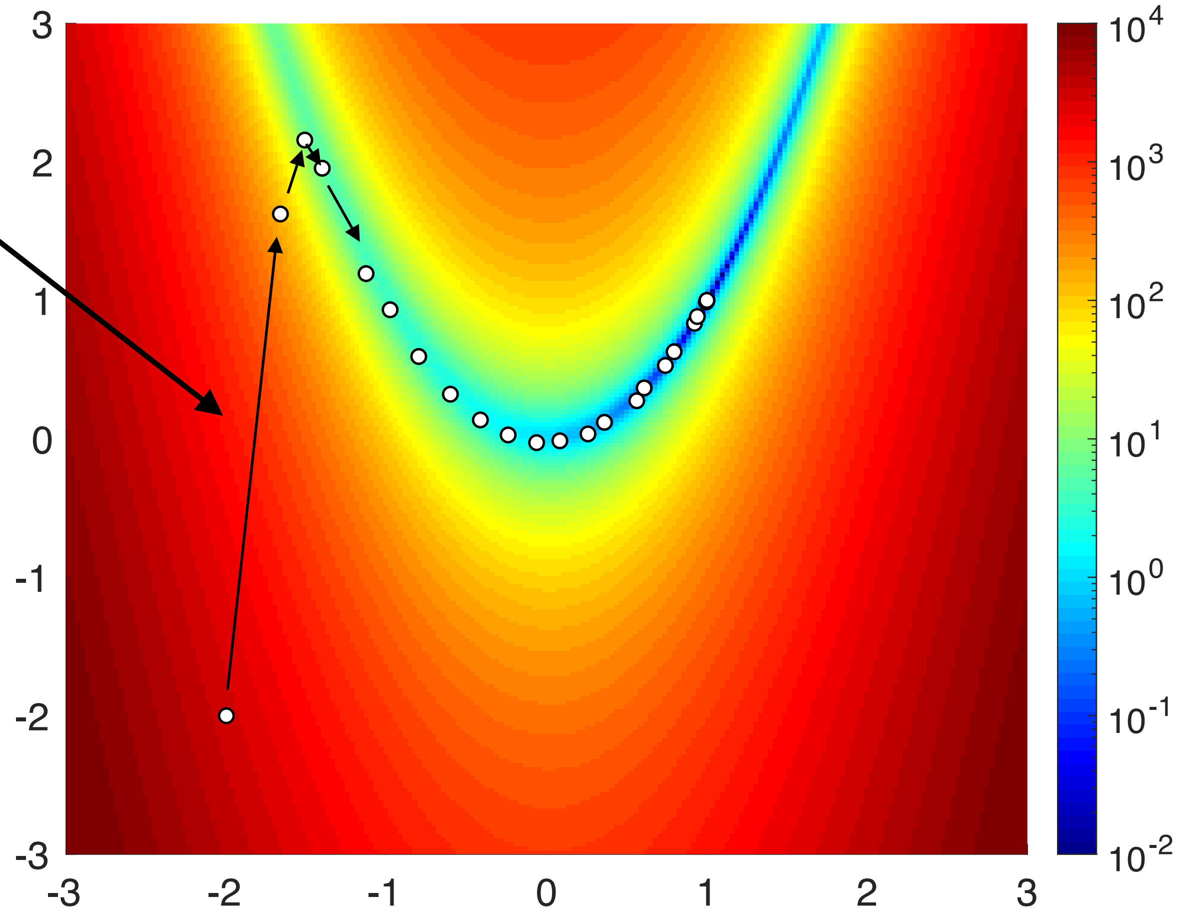
→ The main value of black-box methods are their generality (not necessarily efficient)

Example of White-Box Optimization: Newton's Method

Use derivatives to take step toward reducing objective, i.e.,

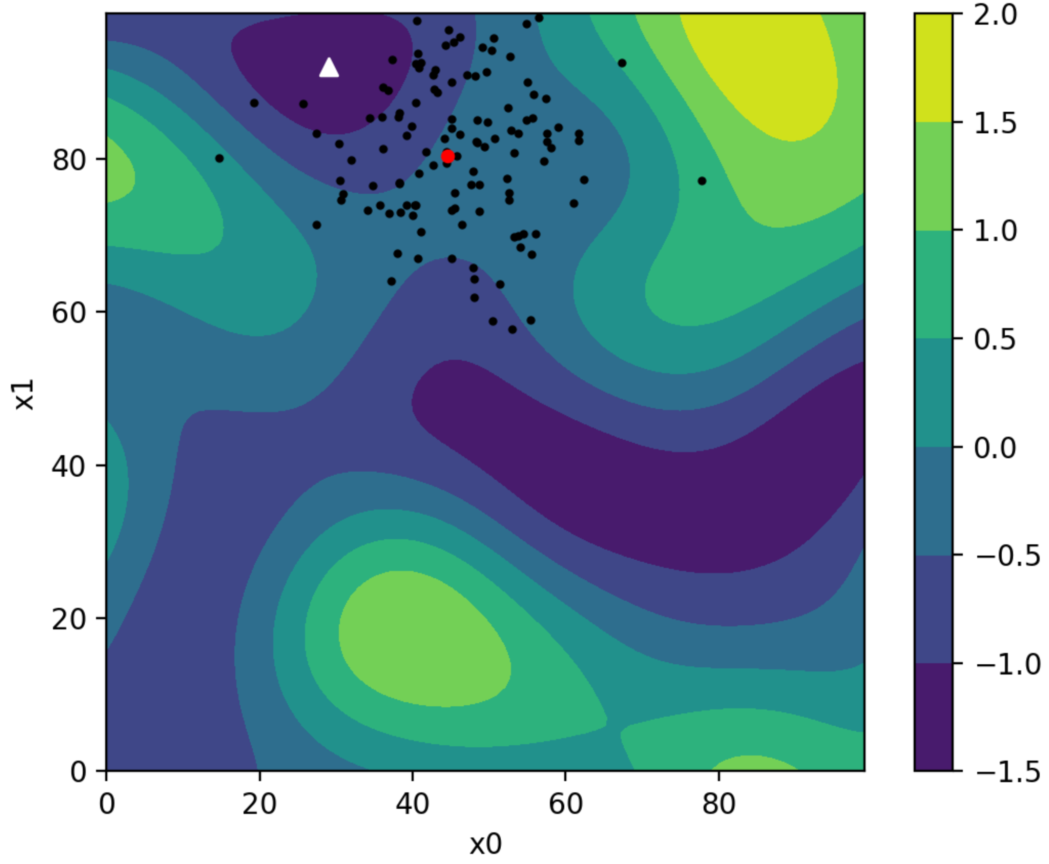
$$x_{k+1} = x_k - \alpha_k \underline{\left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k)}$$

This type of algorithm is "local" (requires initial guess) & requires ability to compute derivatives (expensive when the structure of the function is unknown)



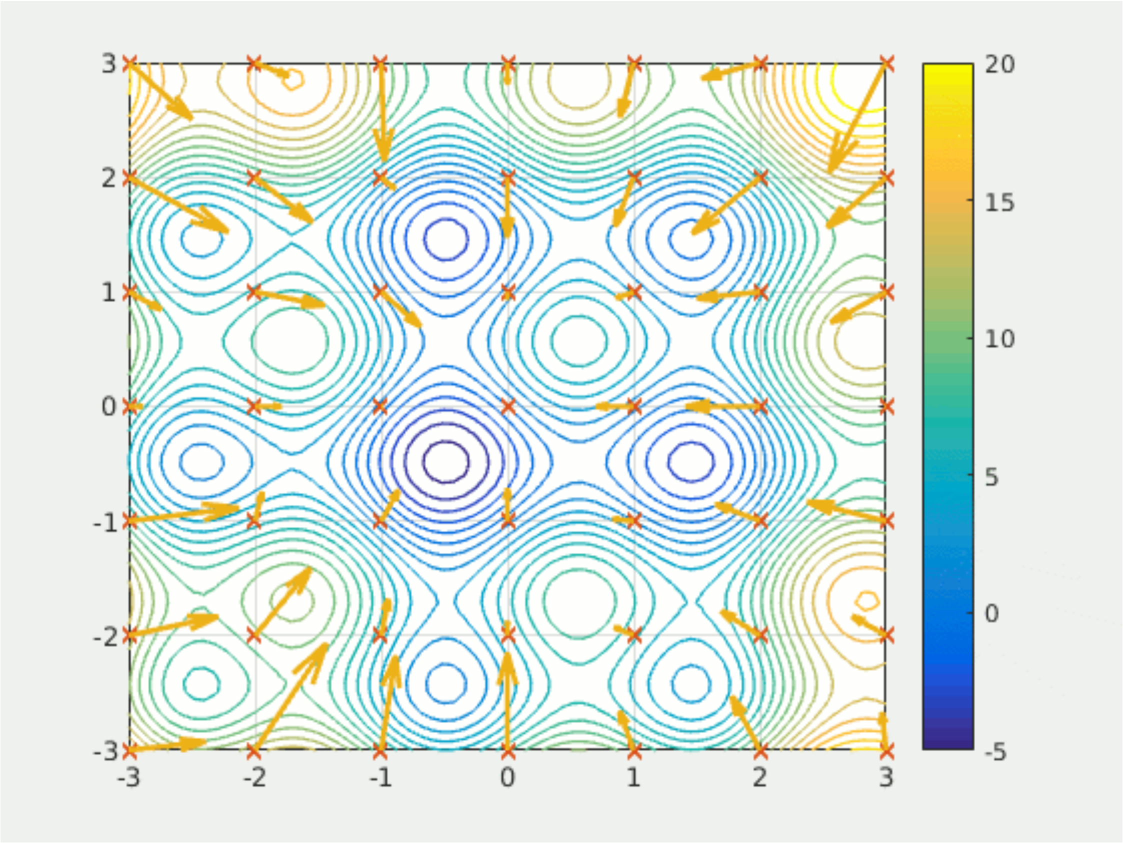
Examples of Black-Box (Derivative-Free) Optimization

Covariance Matrix Adaptive Evolutionary Strategy (CMA-ES)



https://thurinj.github.io/CMA-ES.html

Particle Swarm Optimization (PSO)



https://en.wikipedia.org/wiki/Particle_swarm_optimization

Many derivative-free optimization methods, which to choose?

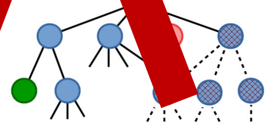
This is really our only option when function evaluations are ***expensive*** (enables us to use all data)

Derivative Free Optimization (DFO)

Deterministic DFO

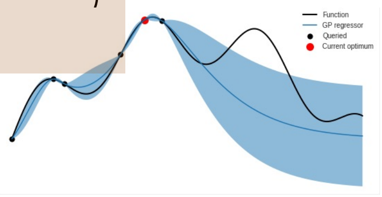
Lipschitzian-based partitioning techniques

- DIRECT Algorithm
- Branch and bound (BB) search



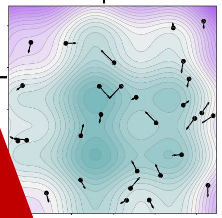
Surrogate-based search algorithms

- Trust-region methods
- Branch & fit algorithm
- Radial basis function (RBF) methods
- ***Bayesian Optimization***



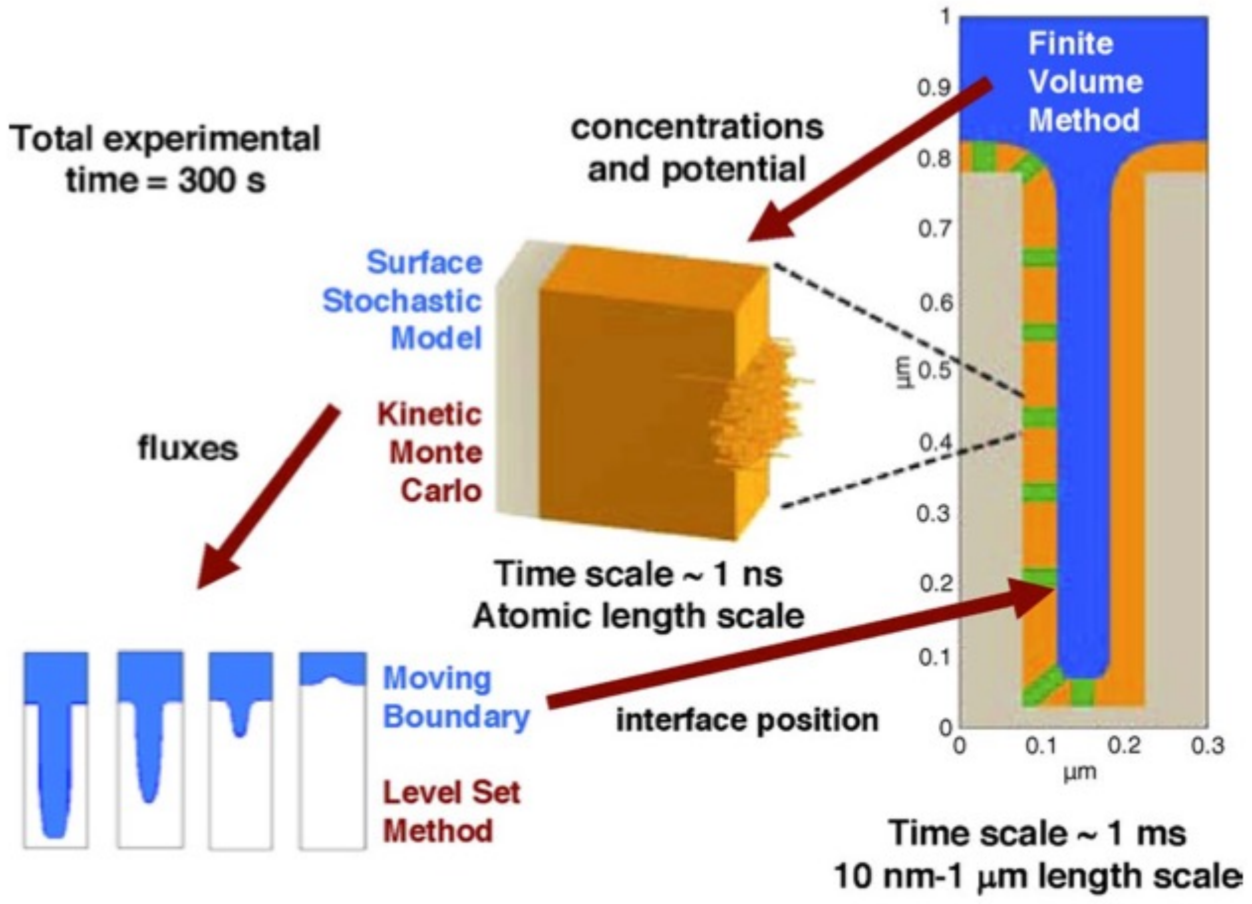
Stochastic DFO

- Hit-and-run
- Simulated annealing
- Genetic Algorithm
- Particle swarm



Expensive functions, they are everywhere

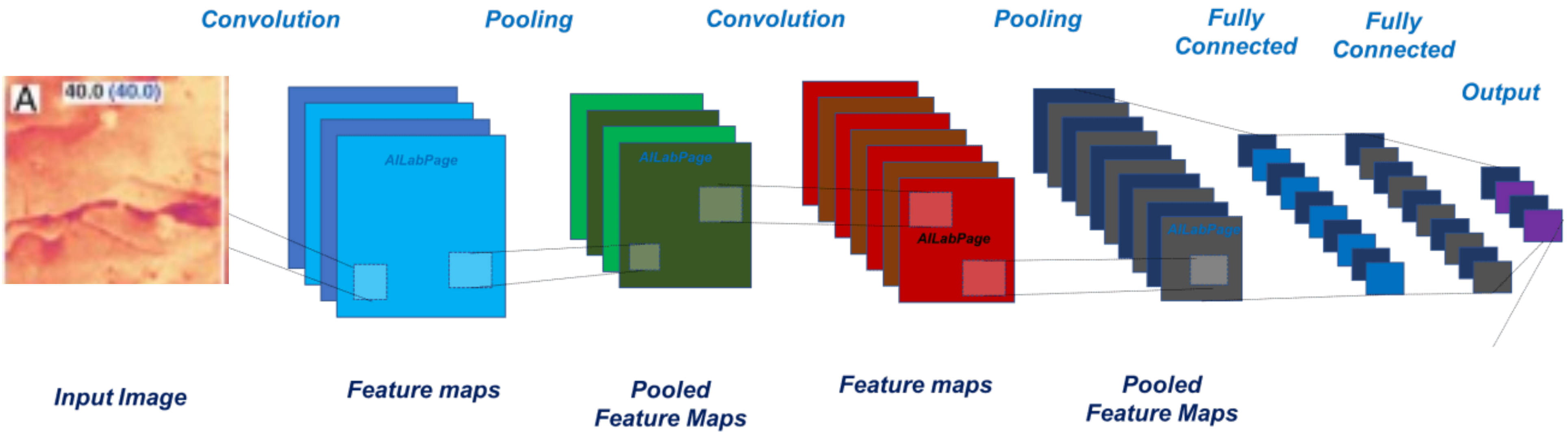
- **Optimizing multi-scale simulation models**



- **Objective:**
  Minimize surface roughness
- **Design variables:**
  Chemical additive concentrations & reaction temperature

Expensive functions, they are everywhere

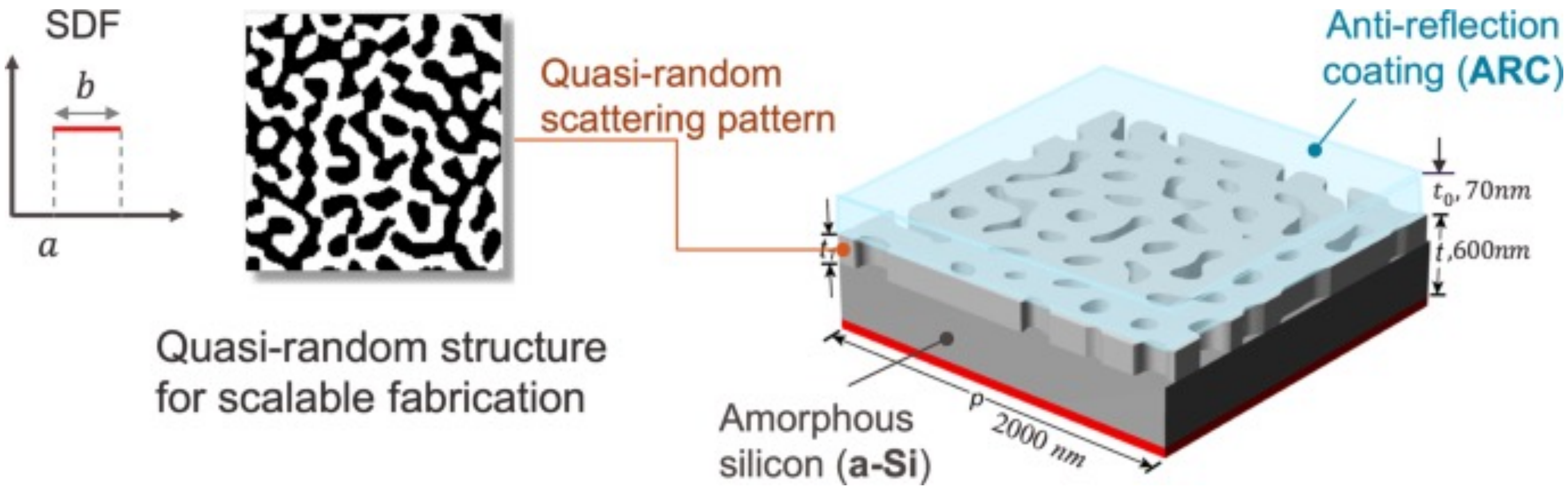
- **Automated machine learning**



- **Objective:** Maximize classification accuracy for image-based chemical sensor
- **Design variables:** Number of layers, number of nodes per layer, learning rates, regularization penalties, activation functions, etc.

Expensive functions, they are everywhere

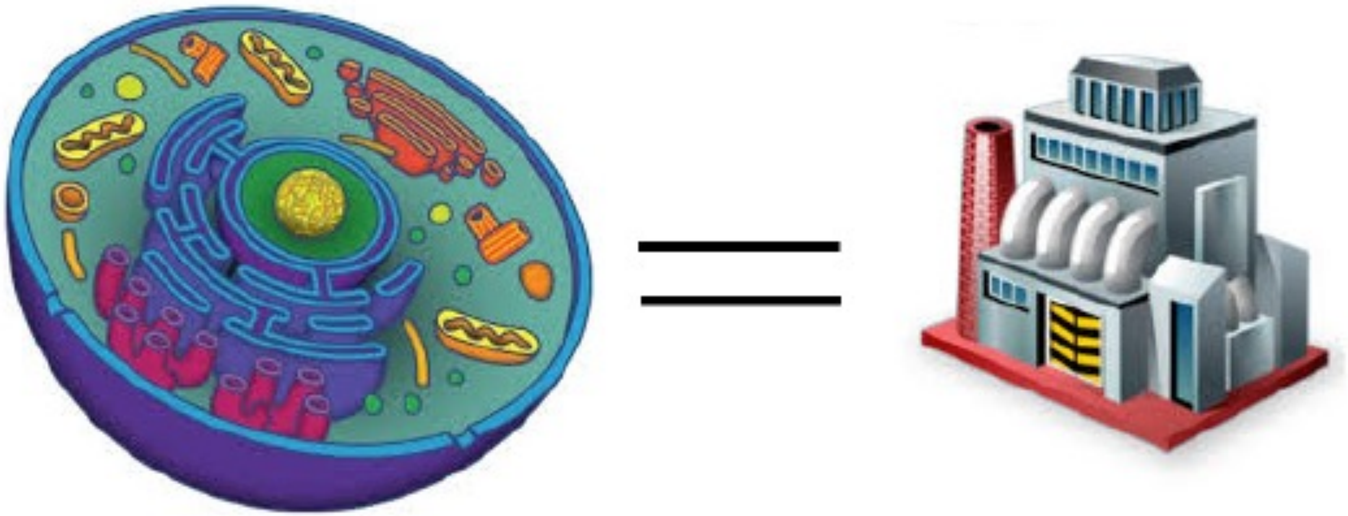
- **Material and drug discovery**



- **Objective:** Maximize light adsorption in quasi-random solar cell
- **Design variables:** Type of amorphous silicon (a-Si), light trapping pattern for fabrication, & overall thickness

Expensive functions, they are everywhere

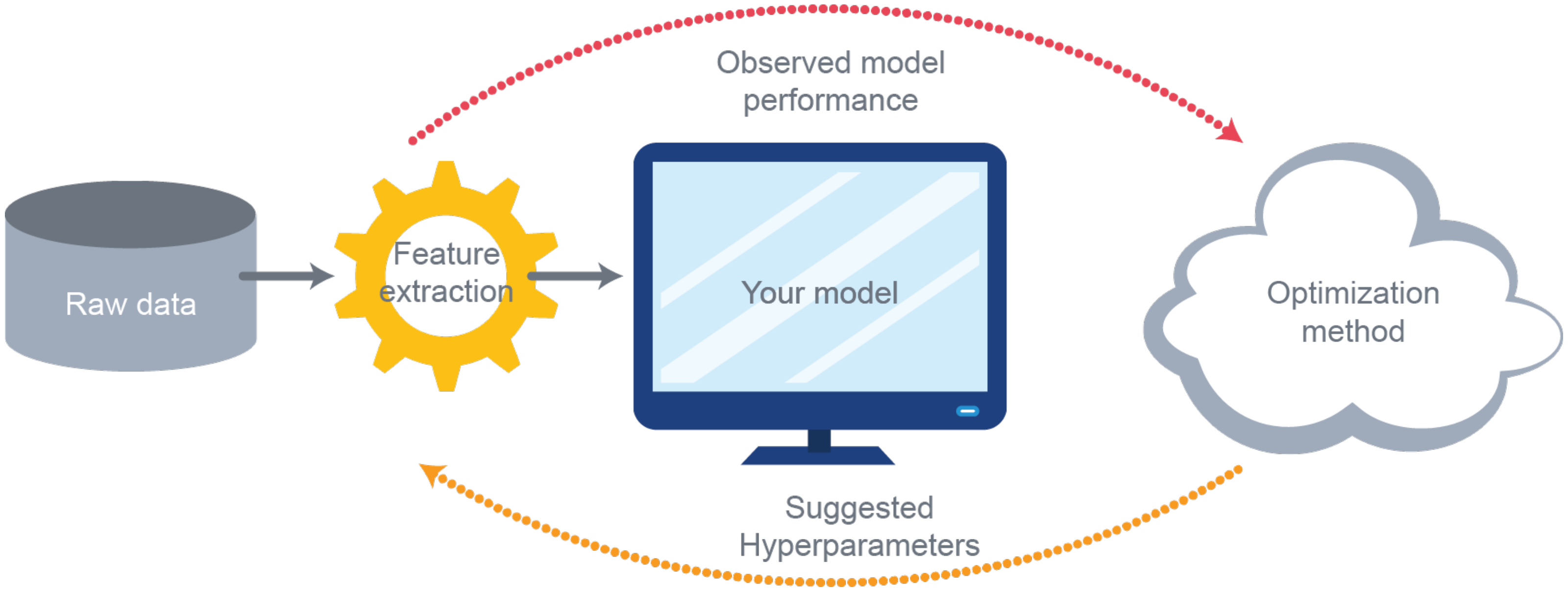
- **Design of experiments: Gene optimization**



  - **Objective:** Maximize efficiency of the cell factory to make product (e.g., proteins)
  - **Design variables:** Gene sequence (e.g., ATTGGTUGA…) & culture conditions (e.g,. pH)

Expensive functions, they are everywhere

- **Tuning hyperparameters in optimization codes**



- **Objective:** Minimize solution time for family of scheduling/planning problems
- **Design variables:** Algorithmic parameters in solver (e.g., CPLEX has 76 design parameters)

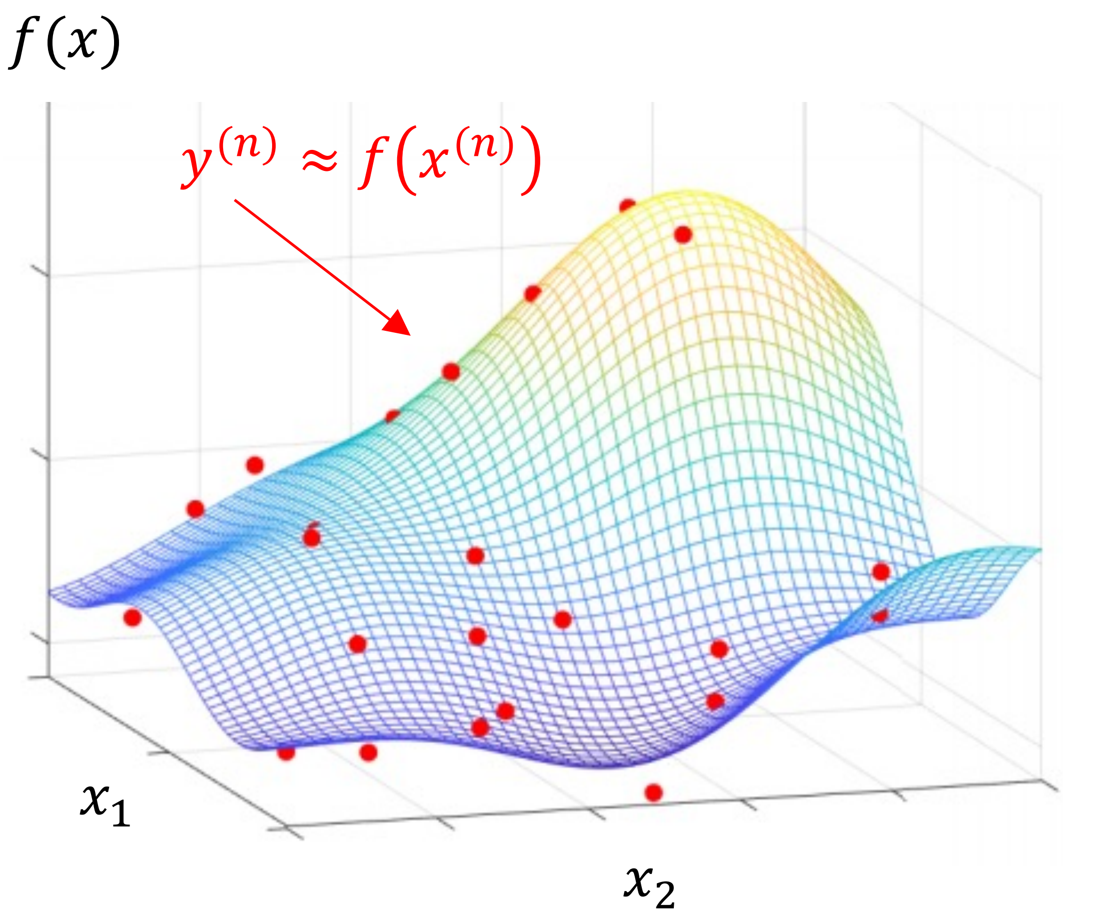
https://sigopt.com/blog/common-problems-in-hyperparameter-optimization/

Expensive functions, they are everywhere

- **Many other problems:**
  - Robotics, aerospace, control, reinforcement learning
  - Tuning websites with A/B testing
  - Calibrating expensive simulators to experimental data
  - etc.…

Standard Goal in Bayesian Optimization:

Optimize functions f : $\mathbb{R}^d \rightarrow \mathbb{R}$ that are:

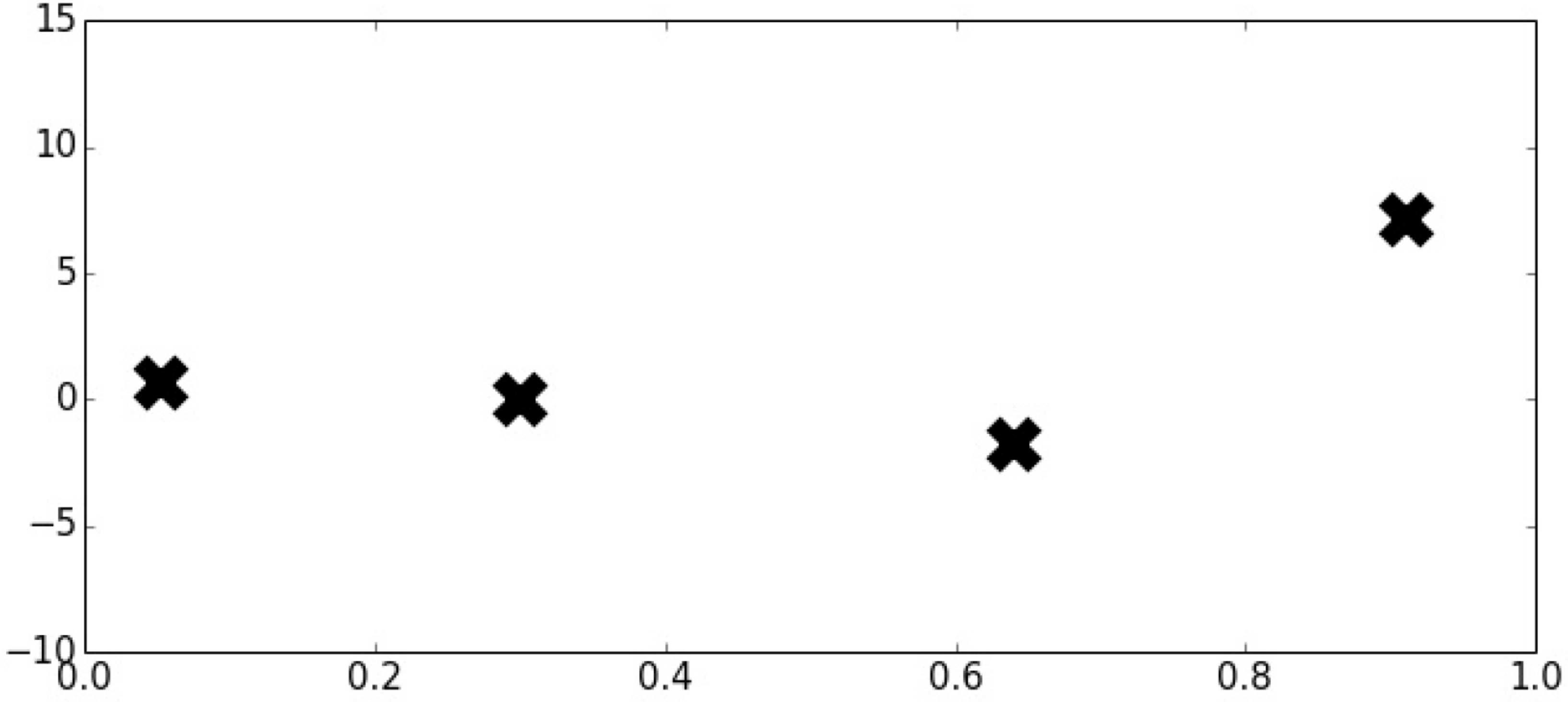


- f(·) is explicitly unknown & non-convex
  - lacks known special structure, e.g., convexity
- f(·) is derivative-free
  - cannot simply get gradients
- f(·) is expensive to evaluate
  - # of evaluations is **severely limited**
- f(·)'s evaluations may be noisy
  - noise independent & ~normally distributed, but unknown variance

*We will deal with black-box constraints later

Illustrative example to build some intuition

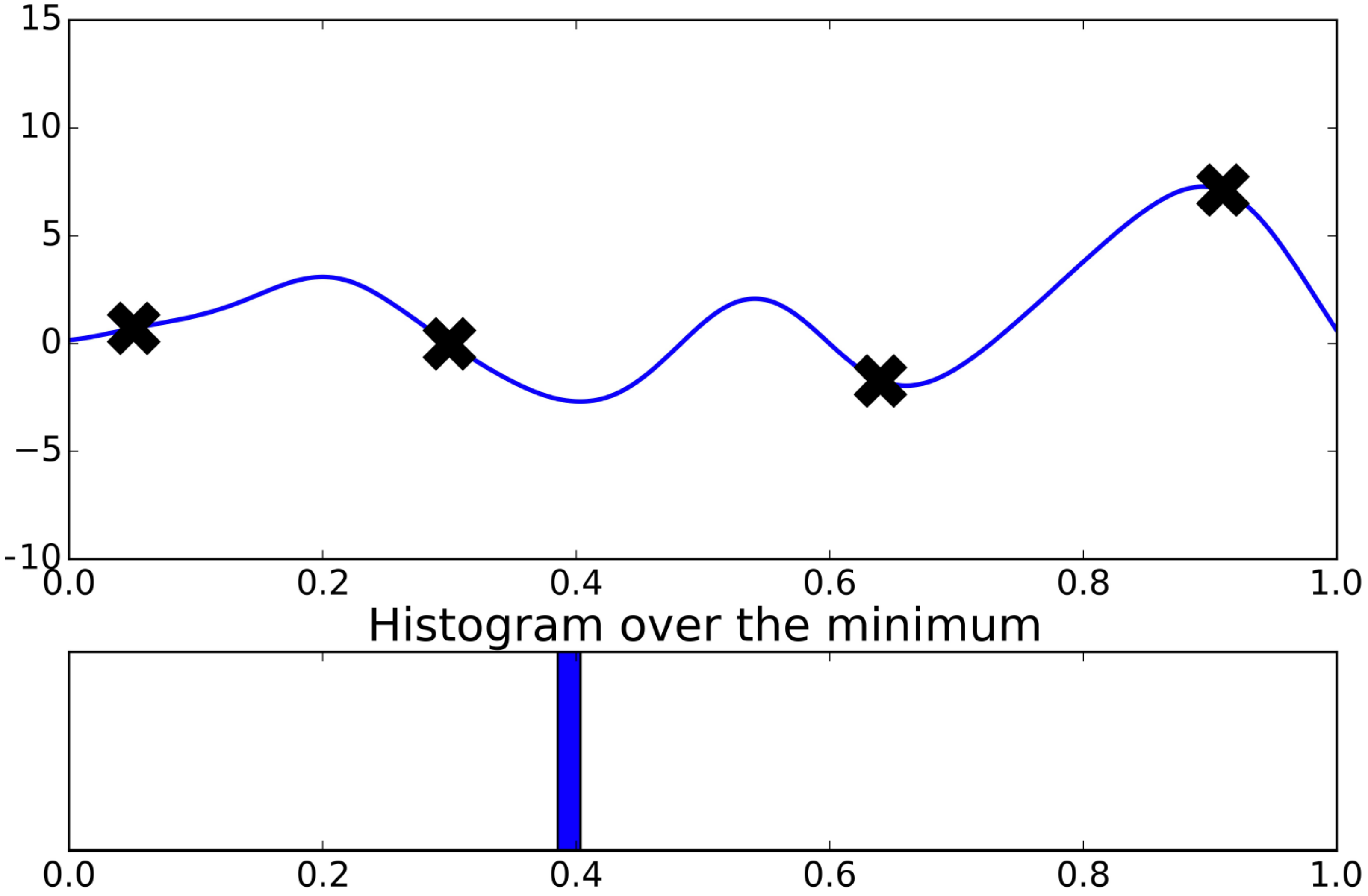
We have four function evaluations



- Where is the minimum of the function f(·)?
- Where should we take our next evaluation?

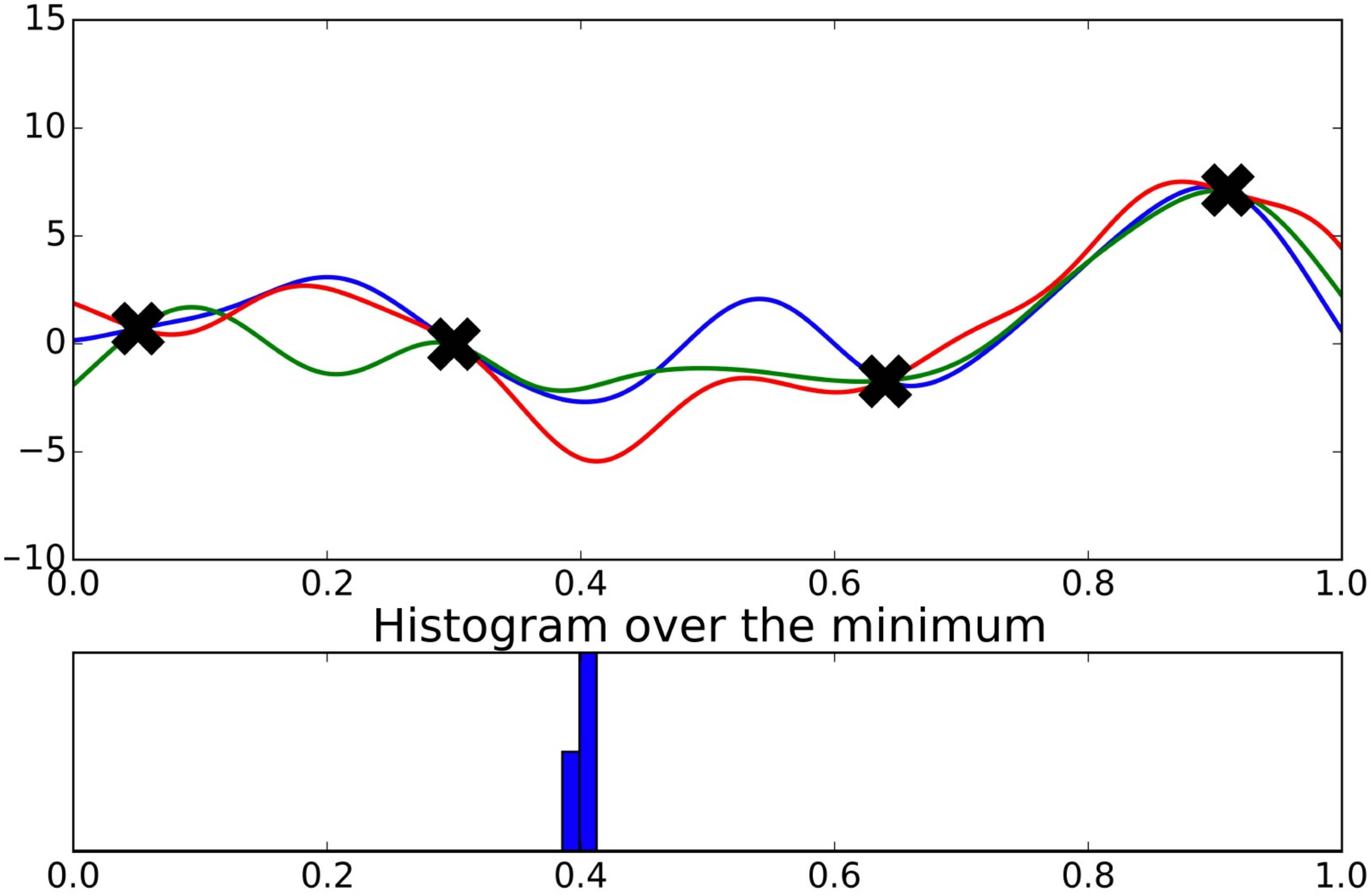
Intuitive solution, fit a surrogate model

One curve; which one should we select?



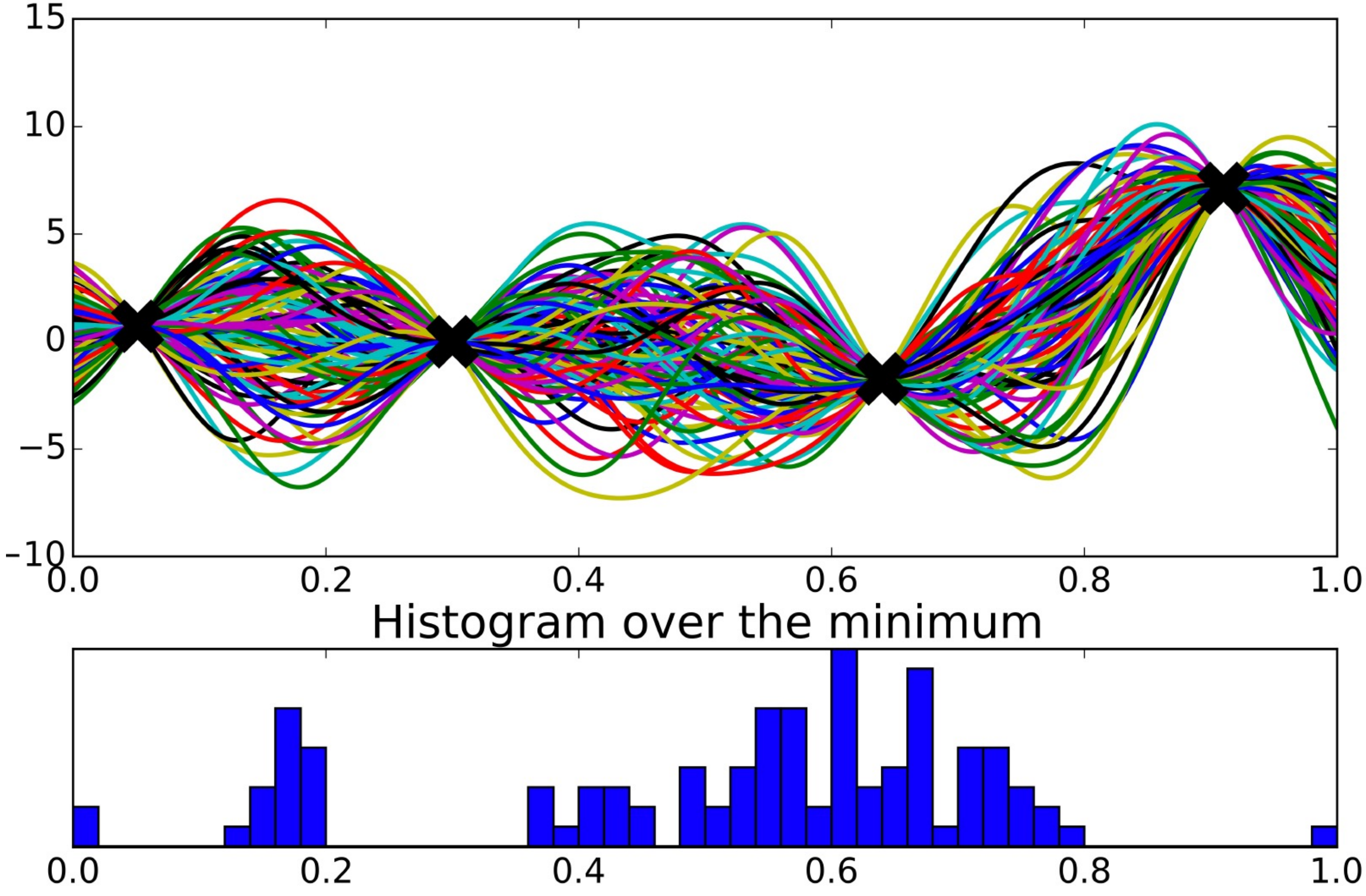
Intuitive solution, fit a surrogate model

Three curves



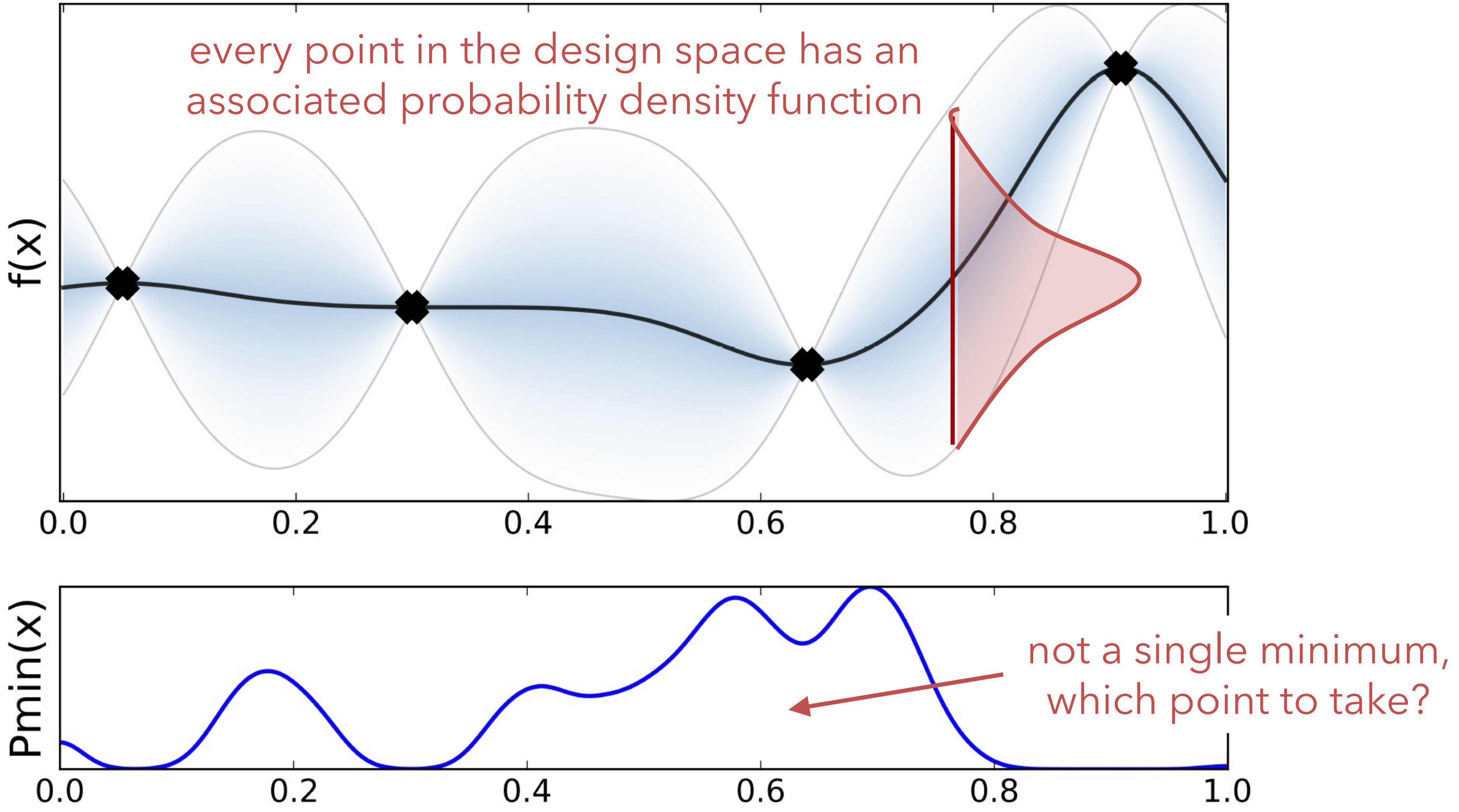
Intuitive solution, fit a surrogate model

One hundred curves



Intuitive solution, fit a surrogate model
Infinite curves

(Need the help of information theory to properly define models + metrics)



Bird's-eye View of Bayesian Optimization

Module 3

while {budget not exhausted}

Module 1

Fit a Bayesian machine learning model
(usually Gaussian process regression)
to observations {x, f(x)}

Find x that maximizes acquisition(x, posterior)

Module 2

Sample x & then observe f(x)

end

More Information

Module 4

Workshop Schedule

9:00 – 9:20	Introduction: Why Go Beyond Traditional Optimization?
9:20 – 10:20	Module 1: Probabilistic Surrogate Modeling*
10:20 – 10:30	Break
10:30 – 11:20	Module 2: Quantifying the Value of Information*
11:20 – 12:20	Module 3: The BO Feedback Loop*
12:20 – 12:30	Break
12:30 – 1:00	Module 4: Beyond Bayesian Optimization

*module includes Python code review / exercises