Bayesian Optimization:
From Foundations to Advanced Topics

twitter

@deshwal_aryan

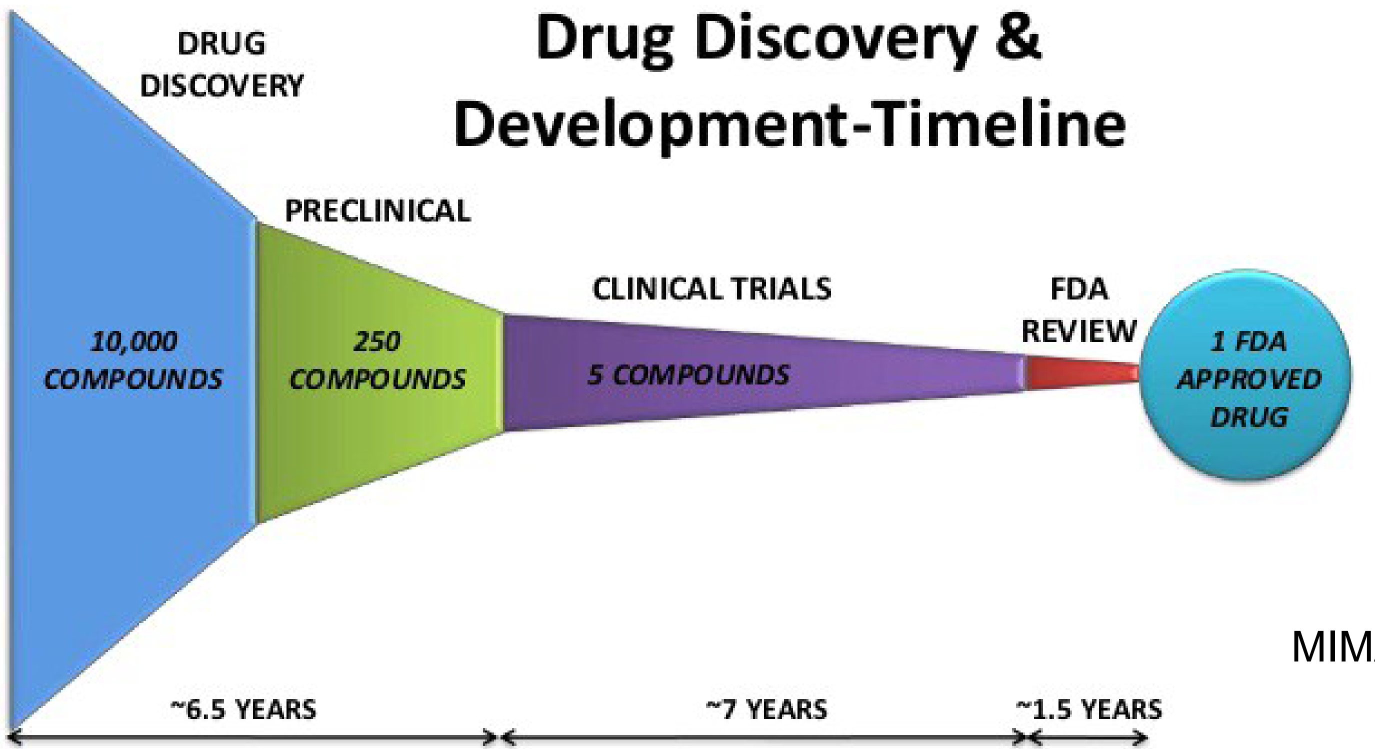
@syrineblk

@janadoppa

Half-day Tutorial

@ AAAI-2022 Conference

Drug/Vaccine Design

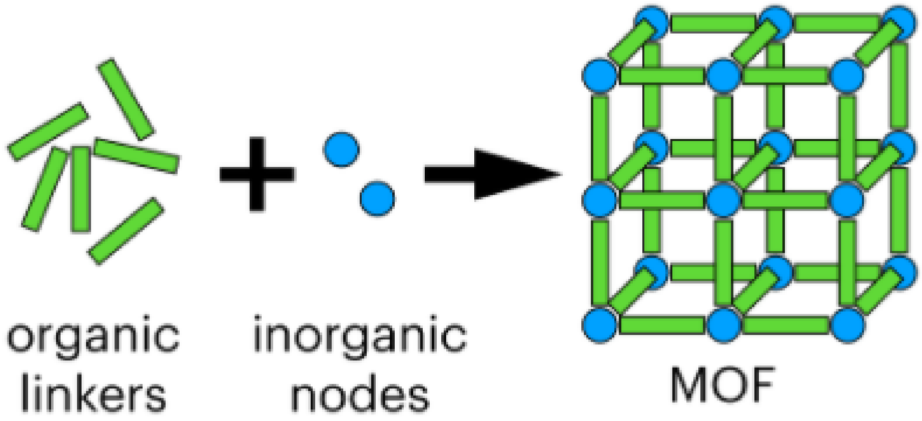


Credit:

MIMA healthcare

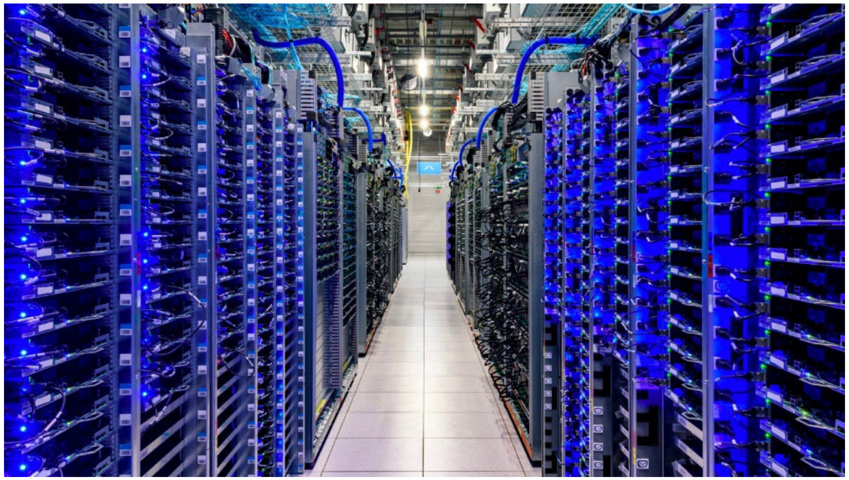
- Accelerate the discovery of promising designs

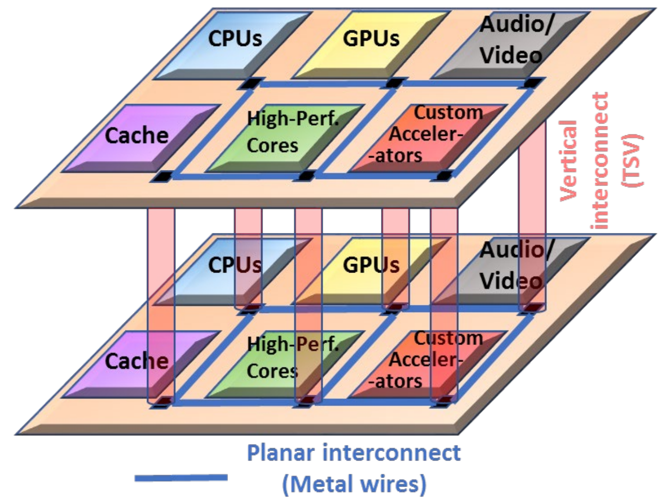
Nanoporous Materials Design



- **Sustainability applications**
  - Storing gases (e.g., hydrogen powered cars)
  - Separating gases (e.g., carbon dioxide from flue gas of coalfired power plants)
  - Detecting gases (e.g., detecting pollutants in outdoor air)

Sustainable Hardware Design for Data Centers





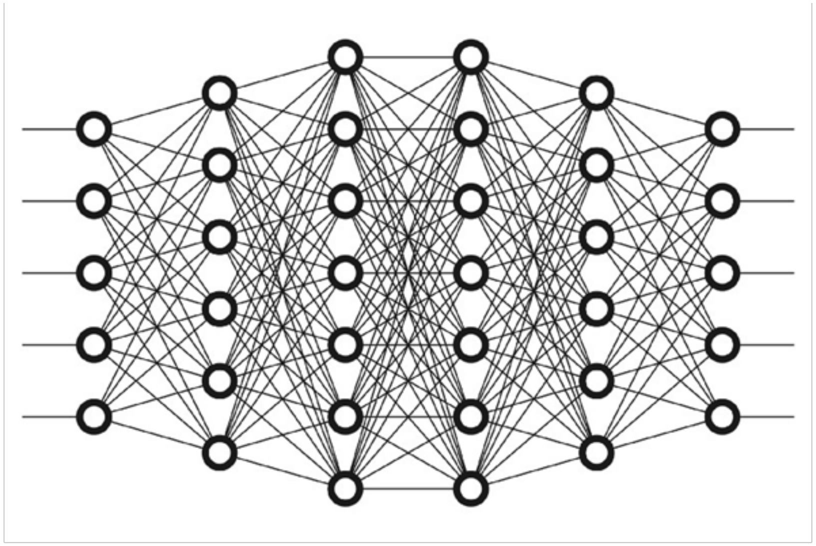
America's Data Centers Are Wasting Huge Amounts of Energy

High-performance and Energy-efficient manycore chips

By 2020, data centers are projected to consume roughly 140 billion kilowatt-hours annually, costing American businesses $13 billion annually in electricity bills and emitting nearly 150 million metric tons of carbon pollution

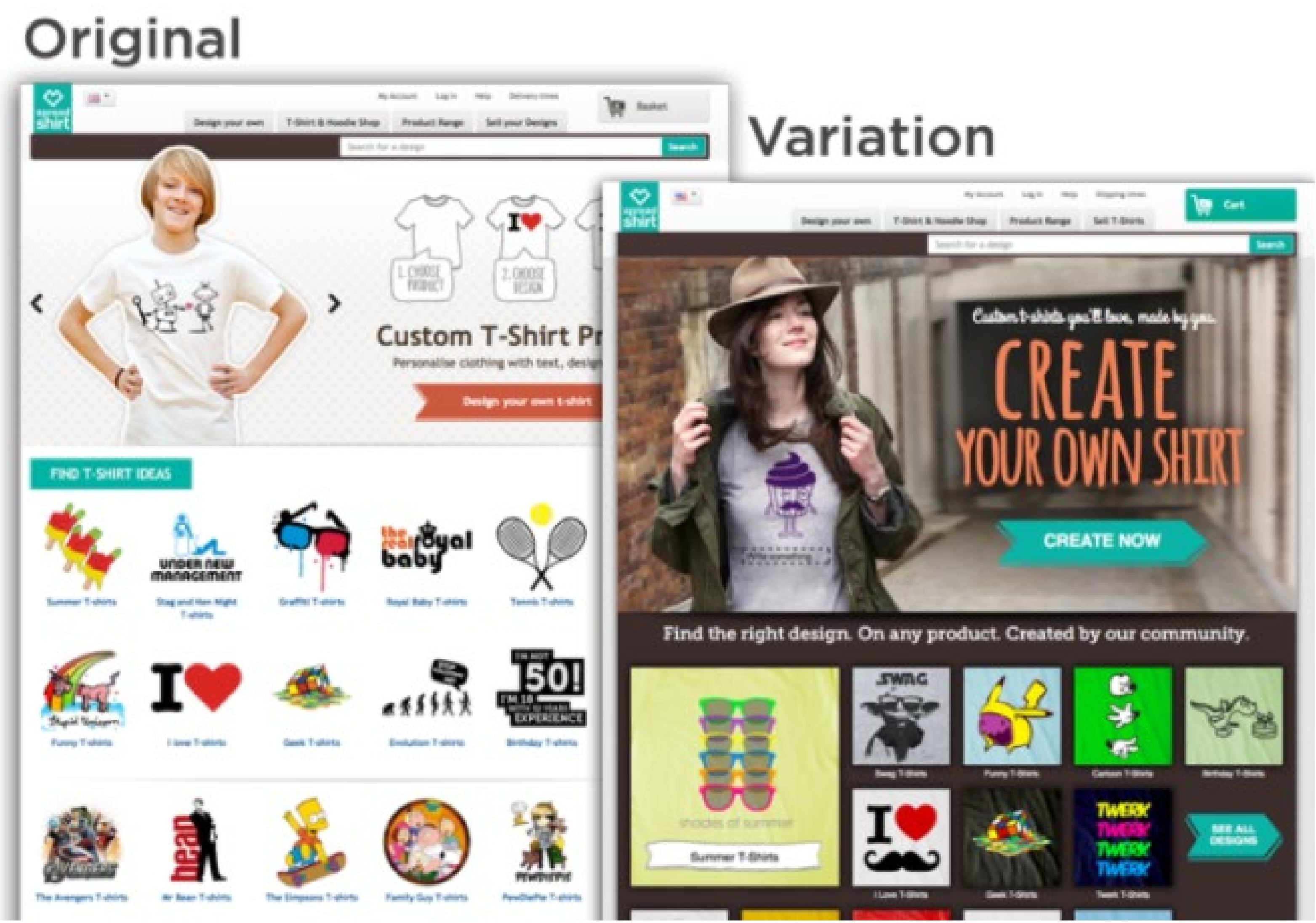
Report from Natural Resources Defense Council:.
https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IB.pdf

Auto ML and Hyperparameter Tuning



- Accuracy of models critically depends on hyper-parameters
  - Optimization algorithm, learning rates, momentum, batch normalization, batch sizes, dropout rates, weight decay, data augmentation, …

A/B Testing to Configure Websites



Making Delicious Cookies

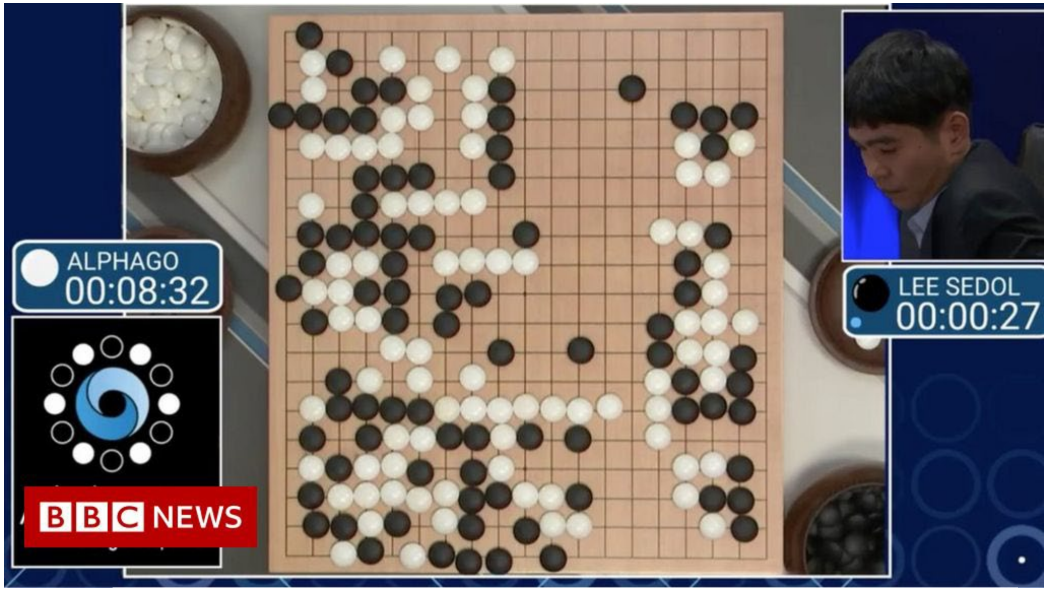
Bayesian Optimization for a Better Dessert

**Greg Kochanski, Daniel Golovin, John Karro, Benjamin Solnik,
Subhodeep Moitra, and D. Sculley**
{gpk, dgg, karro, bsolnik, smoitra, dsculley}@google.com; Google Brain Team

Abstract

We present a case study on applying Bayesian Optimization to a complex real-world system; our challenge was to optimize chocolate chip cookies. The process was a mixed-initiative system where both human chefs, human raters, and a machine optimizer participated in 144 experiments. This process resulted in highly rated cookies that deviated from expectations in some surprising ways – much less sugar in California, and cayenne in Pittsburgh. Our experience highlights the importance of incorporating domain expertise and the value of transfer learning approaches.

Making AlphaGo Better



Bayesian Optimization in AlphaGo

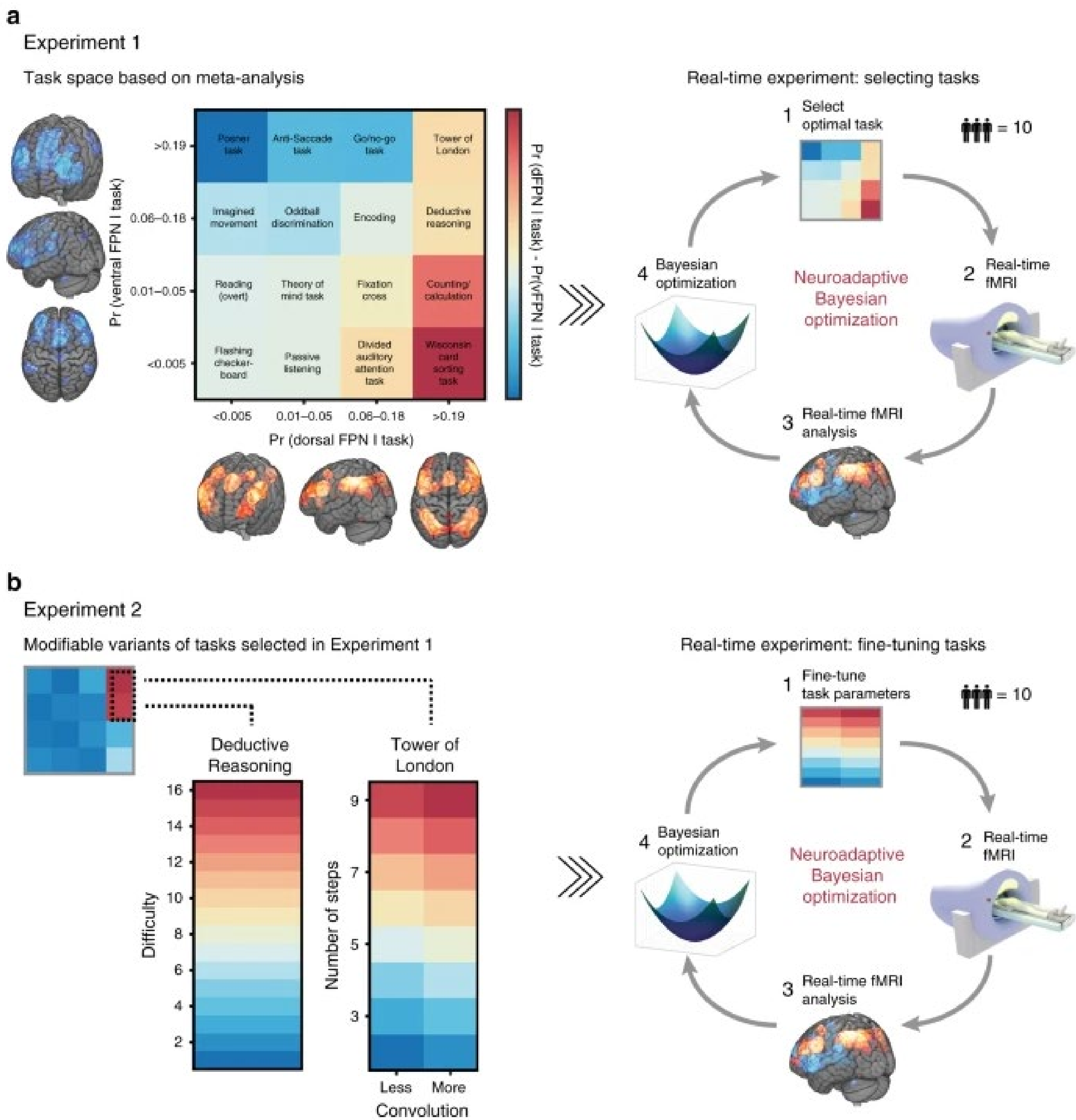
Yutian Chen, Aja Huang, Ziyu Wang, Ioannis Antonoglou, Julian Schrittwieser, David Silver & Nando de Freitas

DeepMind, London, UK
`yutianc@google.com`

Abstract

During the development of AlphaGo, its many hyper-parameters were tuned with Bayesian optimization multiple times. This automatic tuning process resulted in substantial improvements in playing strength. For example, prior to the match with Lee Sedol, we tuned the latest AlphaGo agent and this improved its win-rate from 50% to 66.5% in self-play games. This tuned version was deployed in the final match. Of course, since we tuned AlphaGo many times during its development cycle, the compounded contribution was even higher than this percentage. It is our hope that this brief case study will be of interest to Go fans, and also provide Bayesian optimization practitioners with some insights and inspiration.

Neuroscience and Brain Analytics



Credit: https://www.nature.com/articles/s41467-018-03657-3

Common Attributes of the Search Problem

- Search Space: Many candidate choices (inputs)

- Objective function: Need to perform an expensive experiment to evaluate the objective value of any input

- Optimization problem: find the candidate input with highest objective function value

Common Attributes of the Search Problem

- Search Space: Many candidate choices (inputs)
- Objective function: Need to perform an expensive experiment to evaluate the objective value of any input
- Optimization problem: find the candidate input with highest objective function value

Cannot afford exhaustive search

Common Attributes of the Search Problem

- Search Space: Many candidate choices (inputs)

- Objective function: Need to perform an expensive experiment to evaluate the objective value of any input

- Optimization problem: find the candidate input with highest objective function value

Trial and Error?

Common Attributes of the Search Problem

- Search Space: Many candidate choices (inputs)

- Objective function: Need to perform an expensive experiment to evaluate the objective value of any input

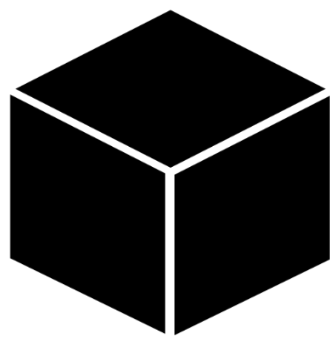
- Optimization problem: find the candidate input with highest objective function value

Can we do better than trial-and-error?

Accelerate Search via Bayesian Optimization

- Efficiently optimize expensive black-box functions

$$x^* = arg \max_{x \in X} f(x)$$

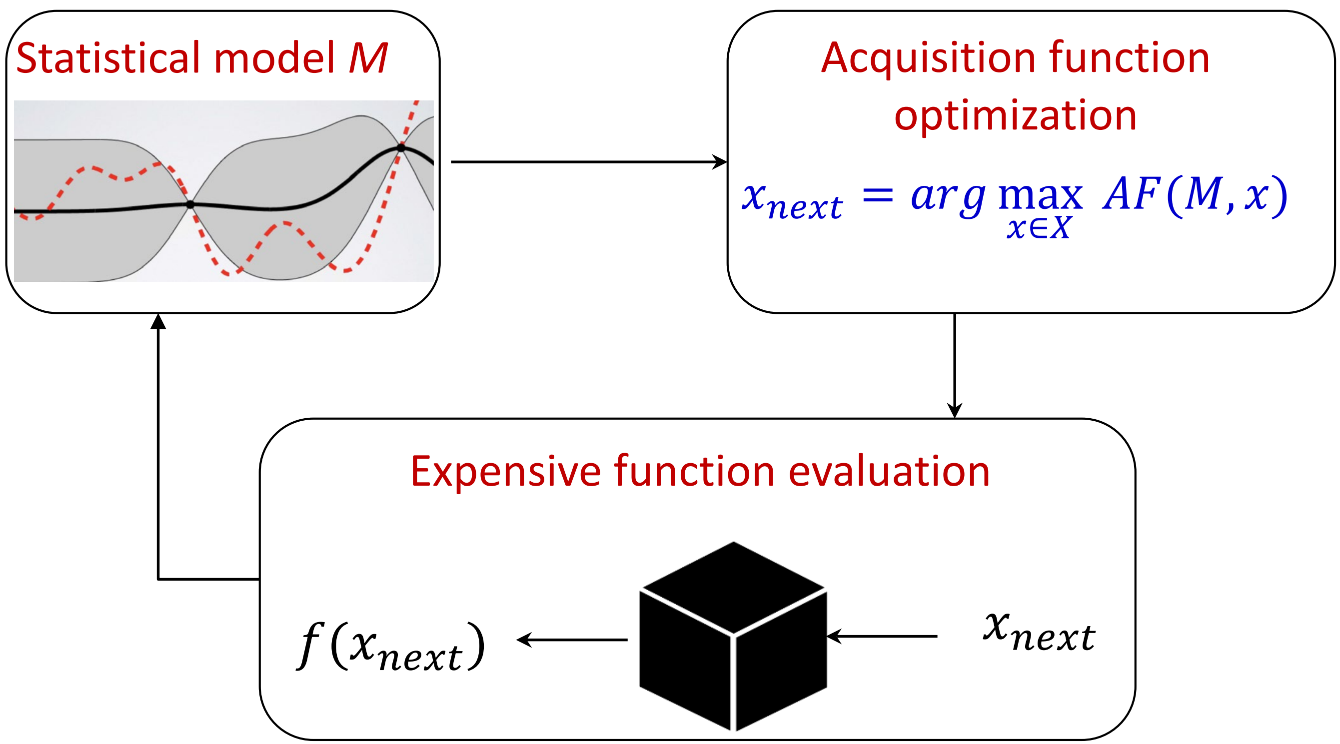
x →  → $f(x)$

input — Function evaluation

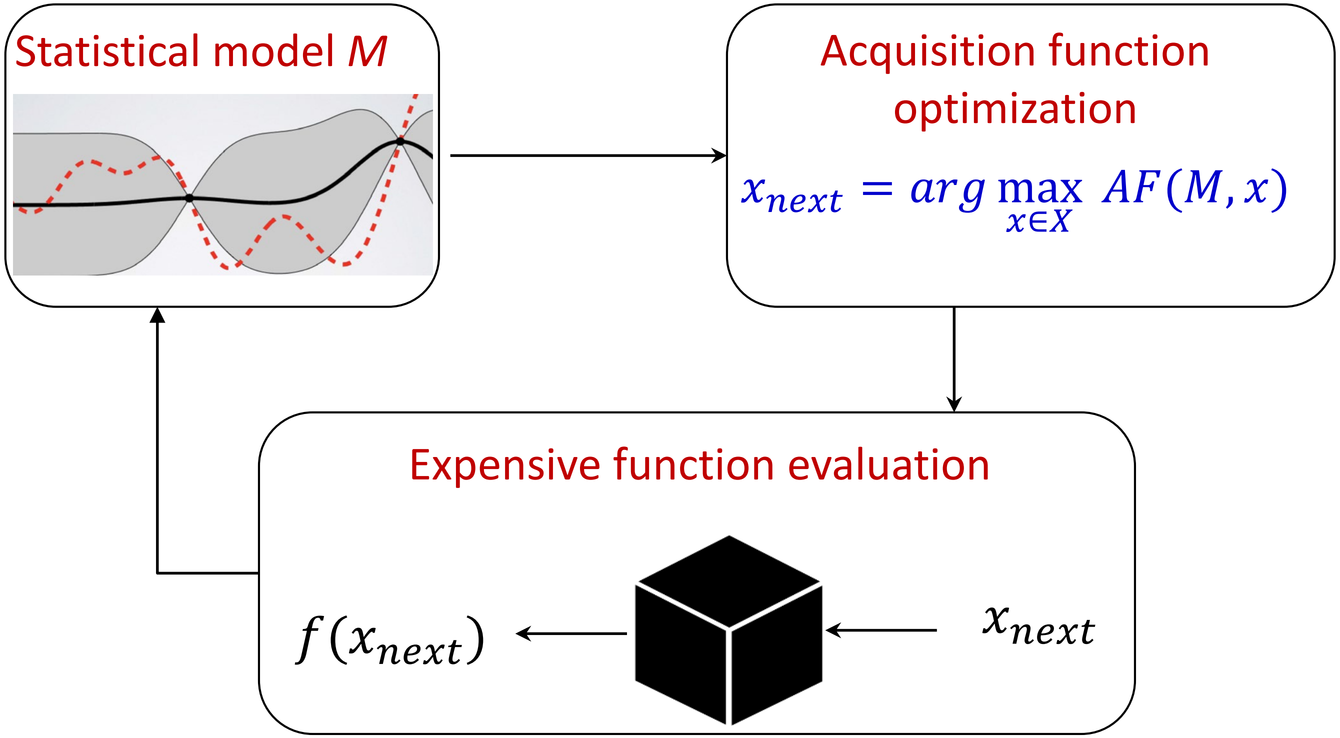
- Black-box queries (aka experiments) are expensive

Bayesian Optimization: Key Idea

- Build a surrogate statistical model and use it to intelligently search the space
  - Replace expensive queries with cheaper queries
  - Use uncertainty of the model to select expensive queries



Bayesian Optimization: Three Key Elements



- Statistical model (e.g., Gaussian process)
- Acquisition function (e.g., Expected improvement)
- Acquisition function optimizer (e.g., local search)

BO Dimensions: Input Space

- **Continuous space**
  - All variables of input x are continuous
- **Discrete / Combinatorial space**
  - Sequences, trees, graphs, sets, permutations etc.
- **Hybrid space**
  - x = mixture of x_d (discrete) and x_c (continuous) variables

BO Dimensions: Input Space

- **Continuous space**
  - All variables of inpu

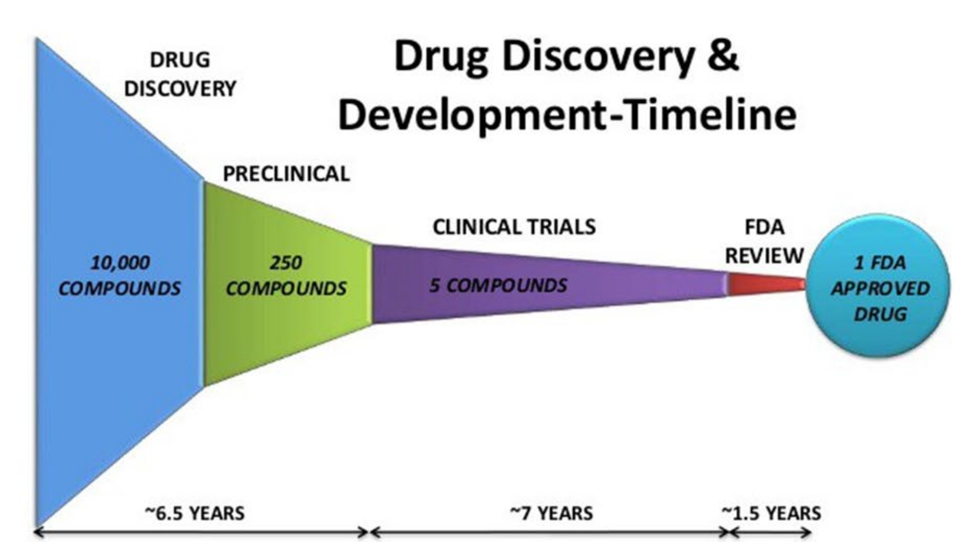
Most of the focus of existing BO work

- **Discrete / Combinatorial space**
  - Sequences, trees, graphs, sets, permutations etc.
- **Hybrid space**
  - x = mixture of x_d (discrete) and x_c (continuous) variables

BO Dimensions: No. of Objectives

- **Single objective**
  - For example, finding hyperparameters to optimize accuracy

- **Multiple objectives**



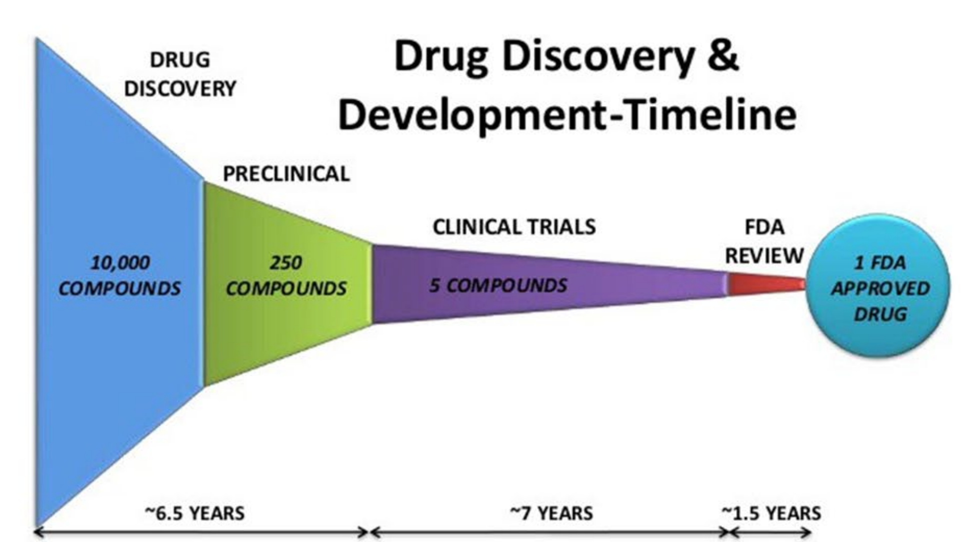
Credit: MIMA healthcare

BO Dimensions: No. of Objectives

- **Single objective**
  - For example, fin ptimize accuracy

Most of the focus of existing BO work

- **Multiple objectives**

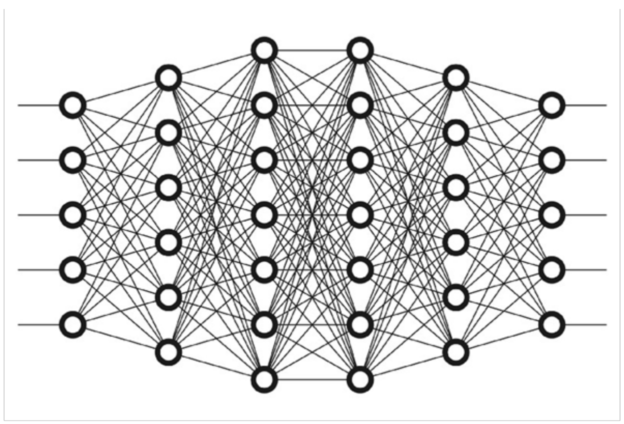


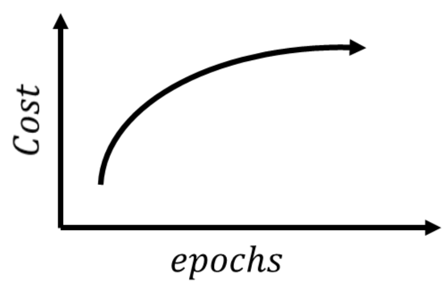
Credit: MIMA healthcare

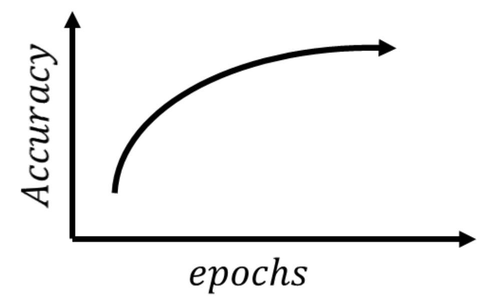
BO Dimensions: No. of Fidelities

- **Single-fidelity setting**
  - Most expensive and accurate function evaluation

- **Multi-fidelity setting**
  - Function evaluations with varying trade-offs in cost and accuracy





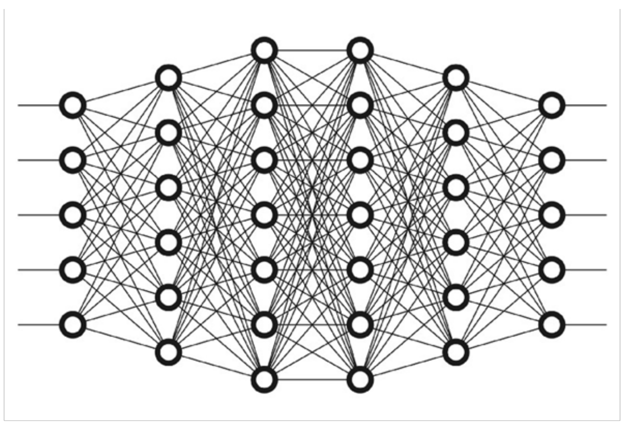


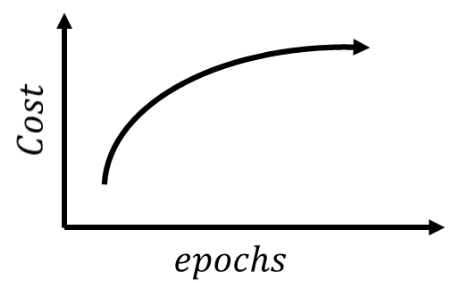
BO Dimensions: No. of Fidelities

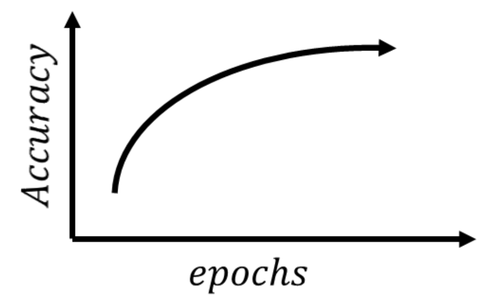
- **Single-fidelity setting**
  - Most expensive and ac

Most of the focus of existing BO work

- **Multi-fidelity setting**
  - Function evaluations with varying trade-offs in cost and accuracy



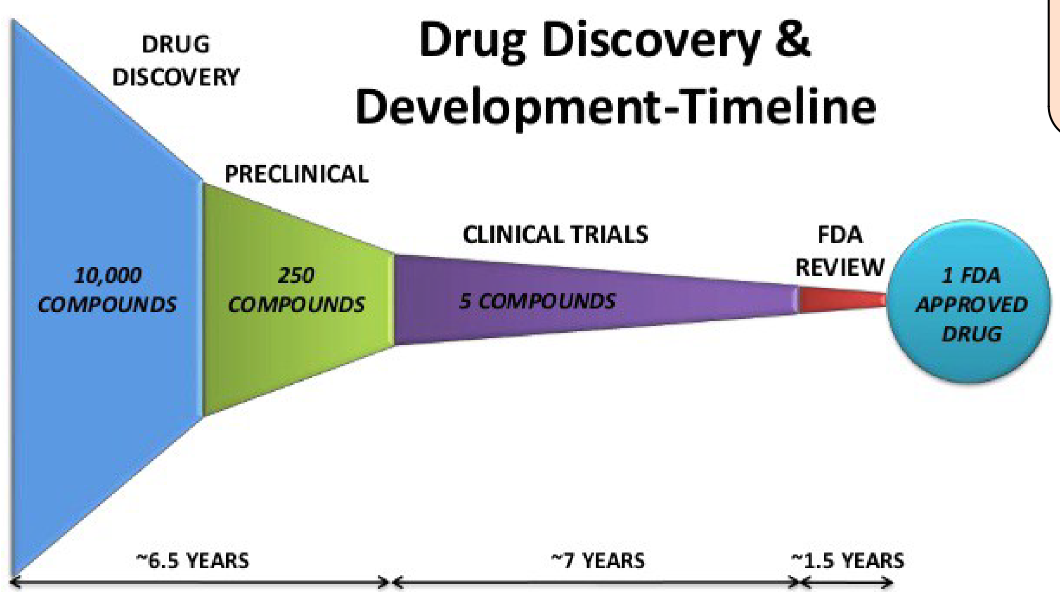




BO Dimensions: Constraints

- **Unconstrained setting**
  - all inputs are valid

- **Constrained setting**



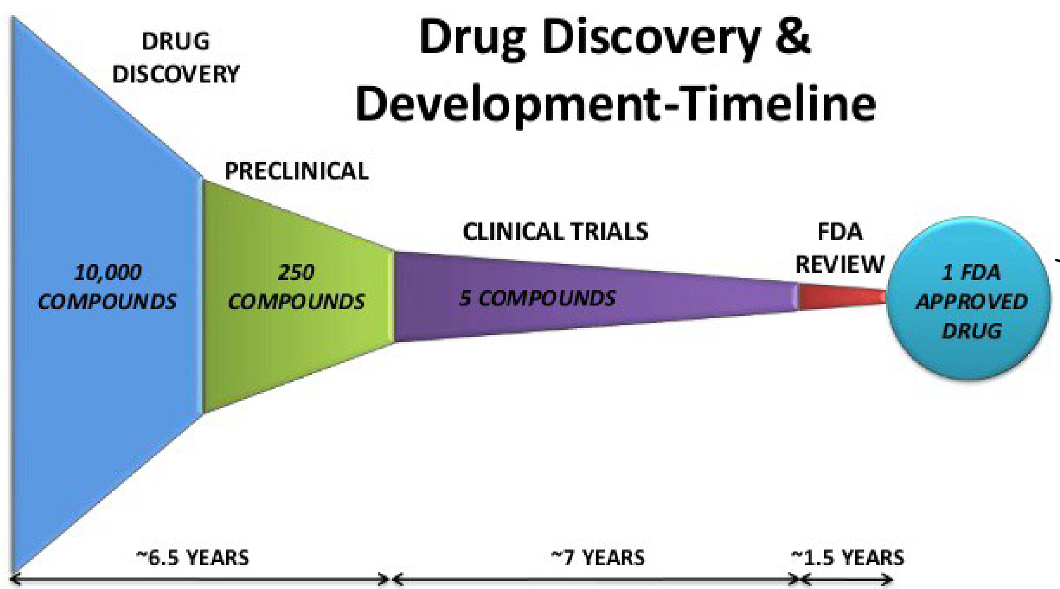
Drugs/Vaccines that are safe

BO Dimensions: Constraints

- **Unconstrained setting**
  - all inputs are valid

Most of the focus of existing BO work

- **Constrained setting**



Drugs/Vaccines that are safe

Outline of the Tutorial

- Background on GPs and Single-Objective BO
- Bayesian Optimization over Combinatorial Spaces
- Bayesian Optimization over Hybrid Spaces

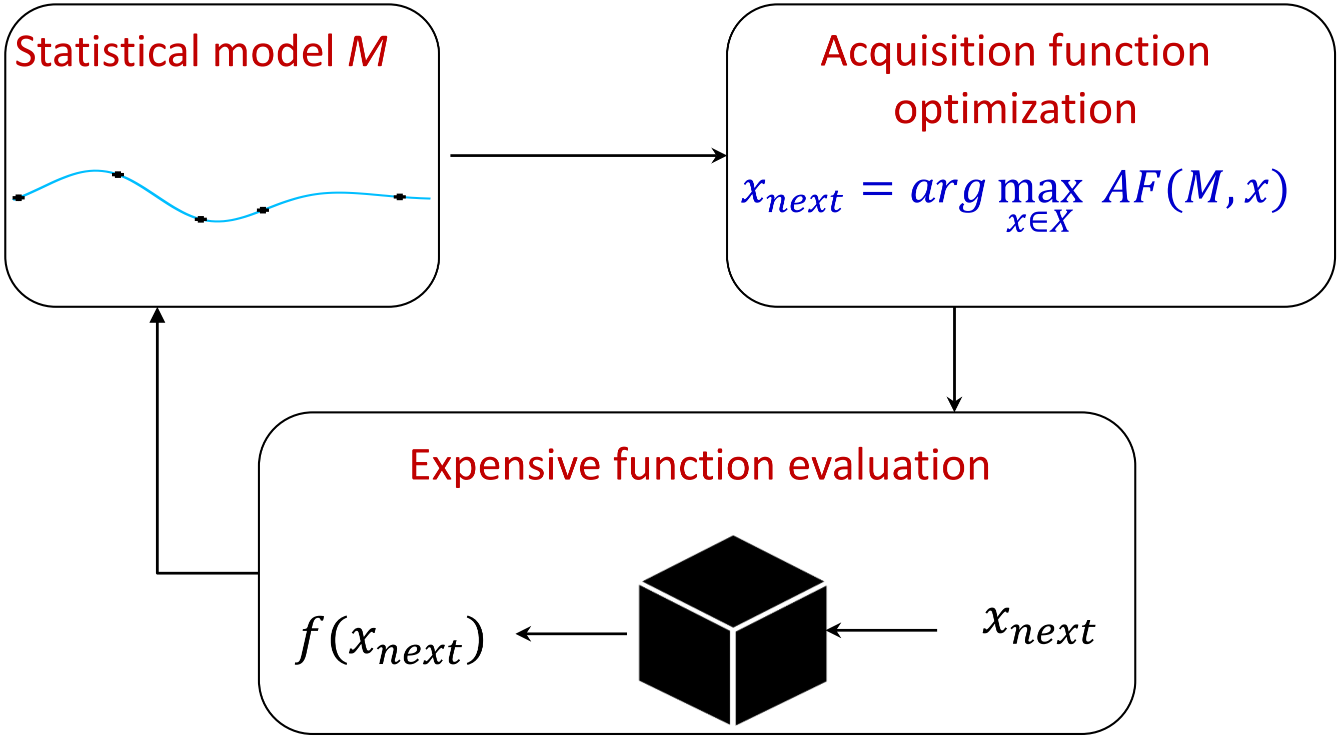
Break

- Multi-Fidelity Bayesian Optimization
- Constrained Bayesian Optimization
- Multi-Objective Bayesian Optimization
- Summary and Outstanding Challenges in BO

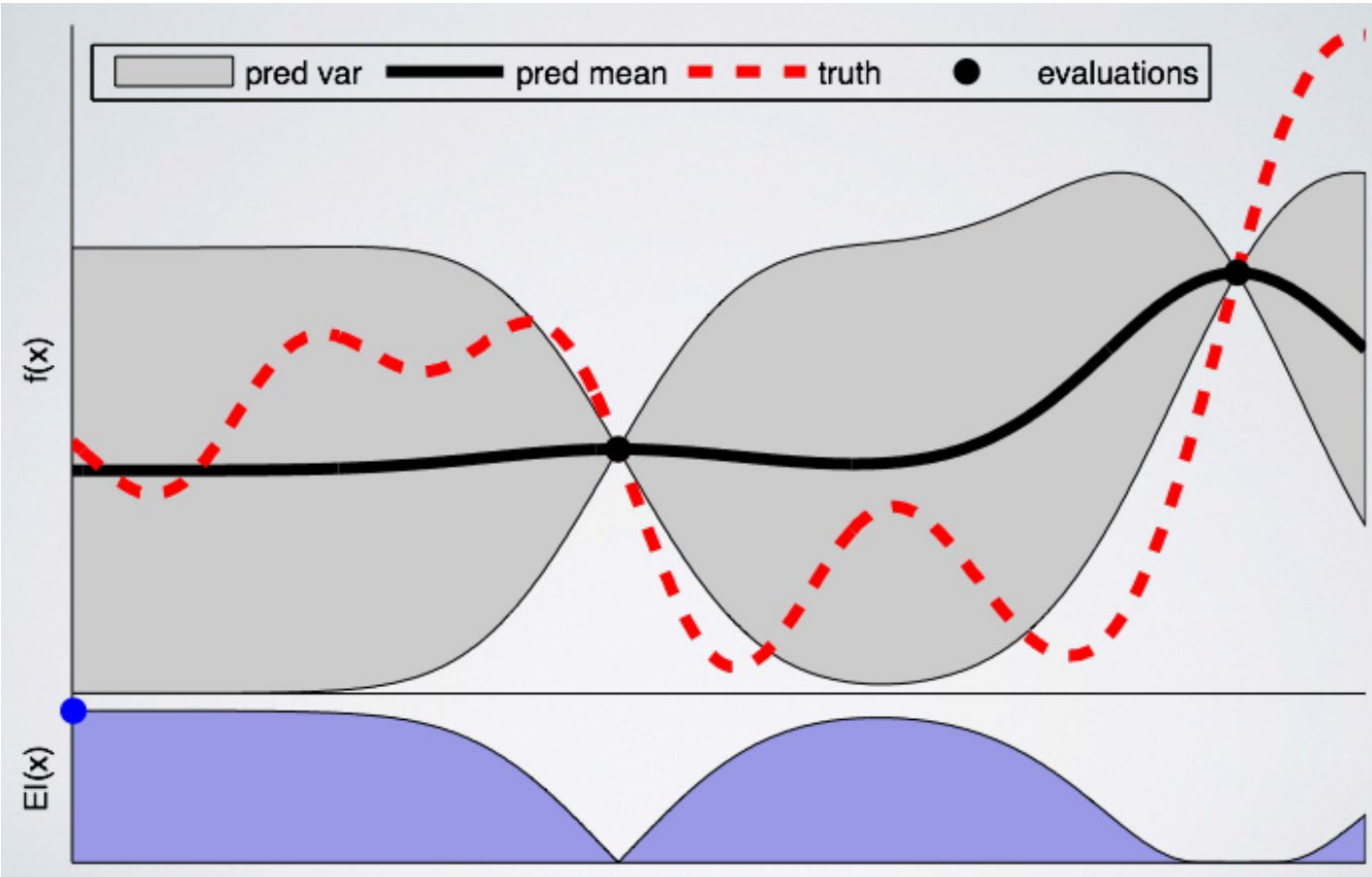
Background on Gaussian Processes
and
Single-Objective Bayesian Optimization

Bayesian Optimization: Key Idea

- Build a surrogate statistical model and use it to intelligently search the space
  - Replace expensive queries with cheaper queries
  - Use uncertainty of the model to select expensive queries



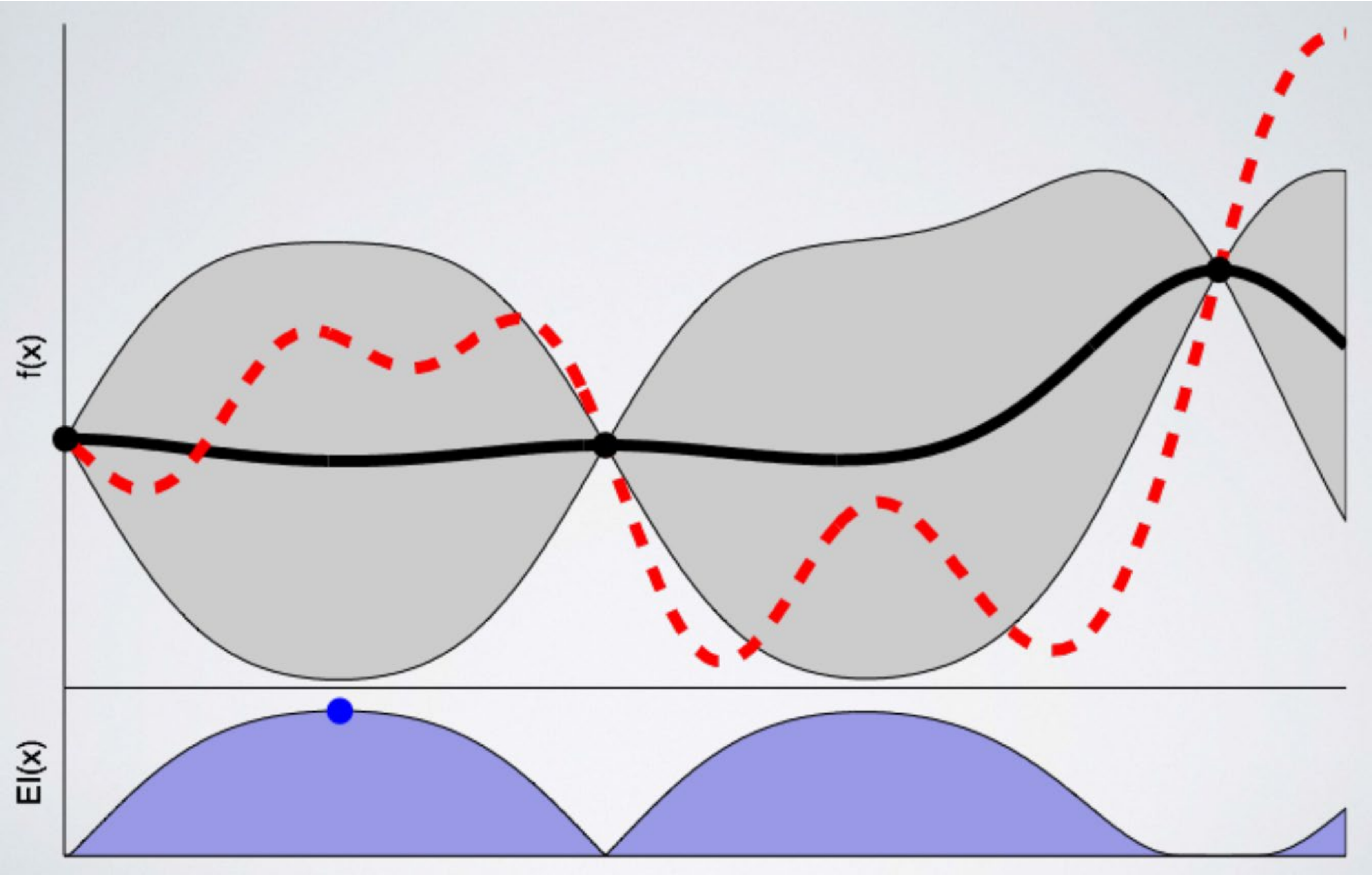
Bayesian Optimization: Illustration



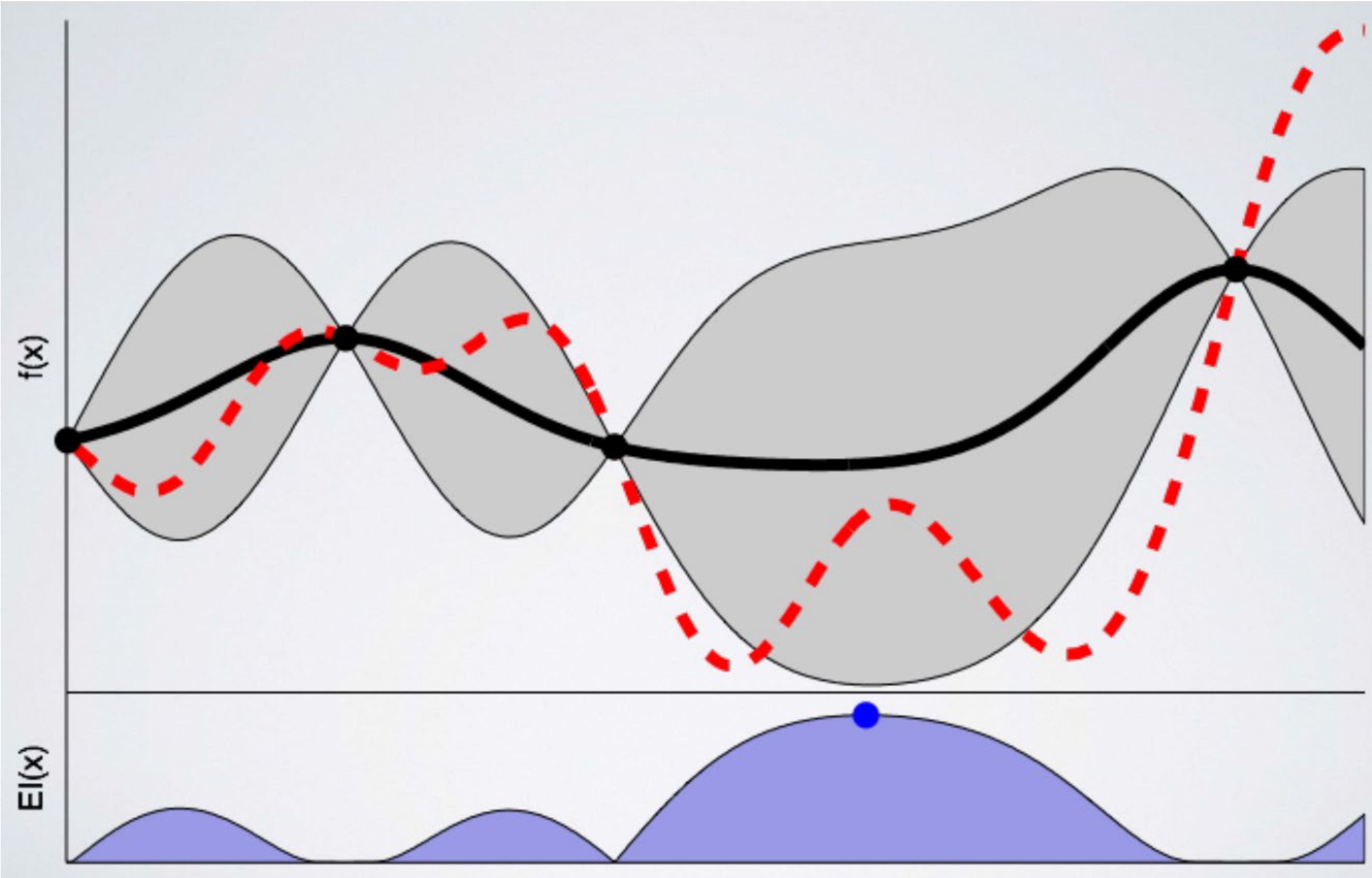
Credit: Ryan Adams

https://www.cs.toronto.edu/~rgrosse/courses/csc411_f18/tutorials/tut8_adams_slides.pdf

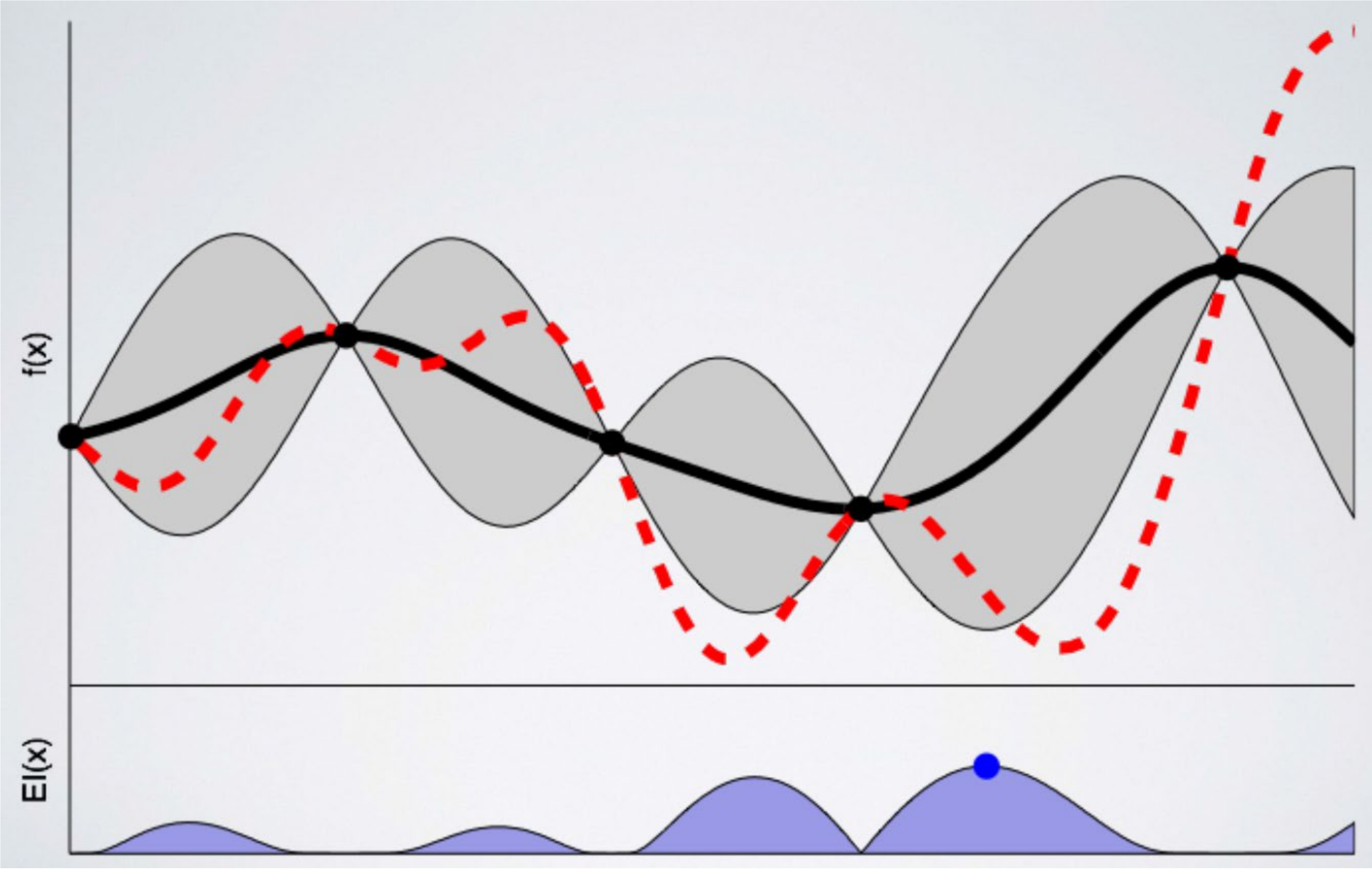
Bayesian Optimization: Illustration



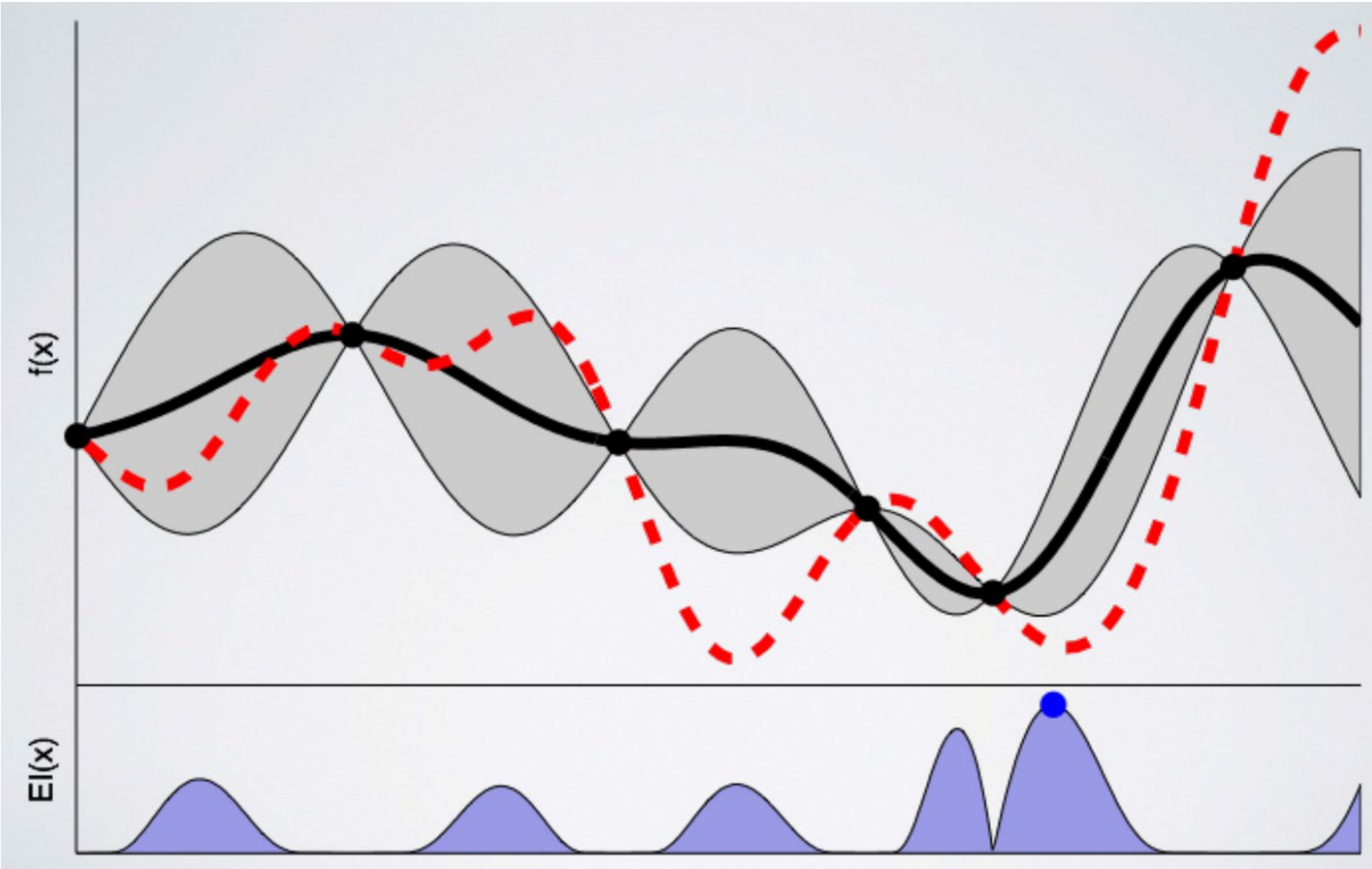
Bayesian Optimization: Illustration



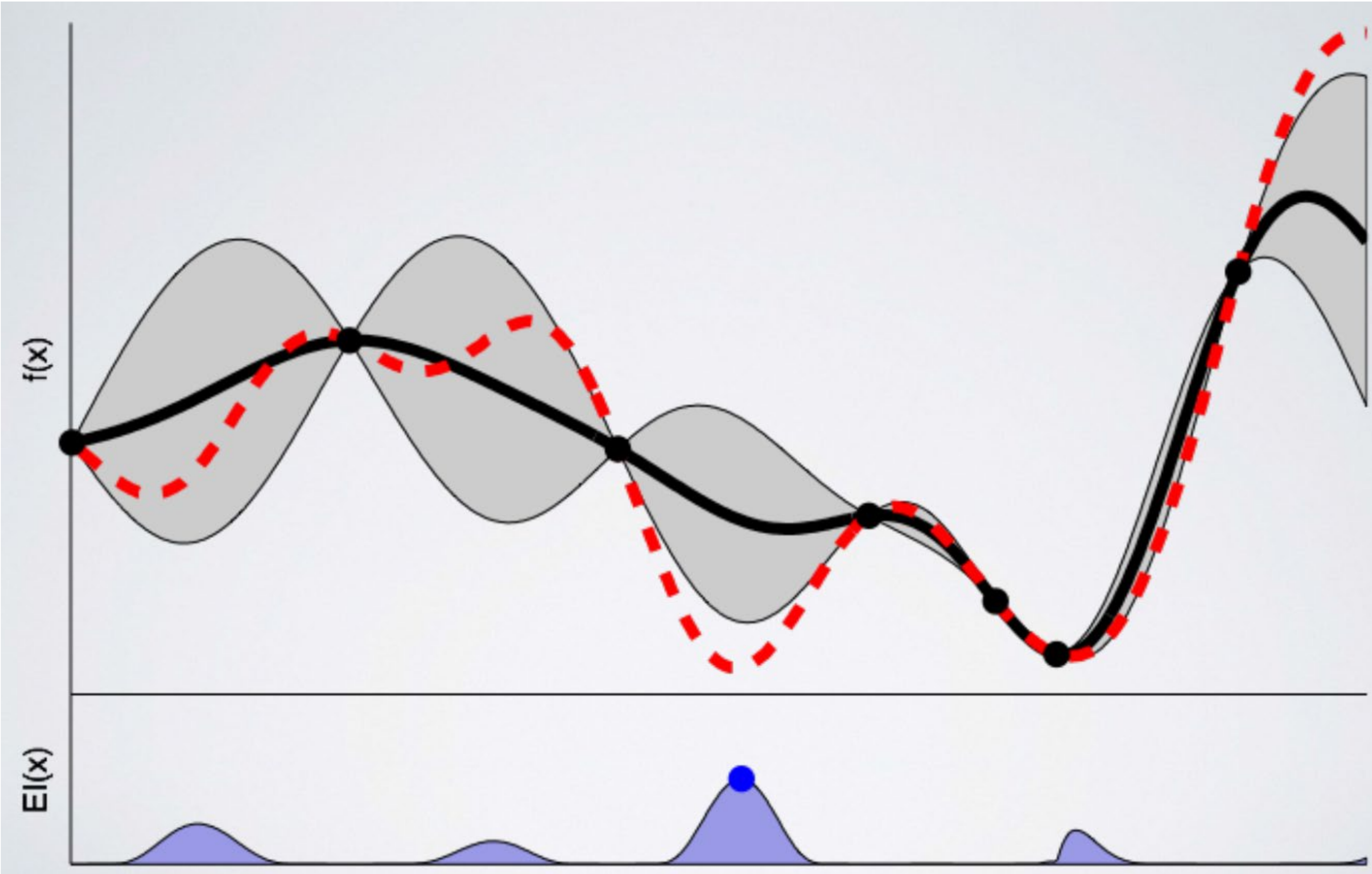
Bayesian Optimization: Illustration



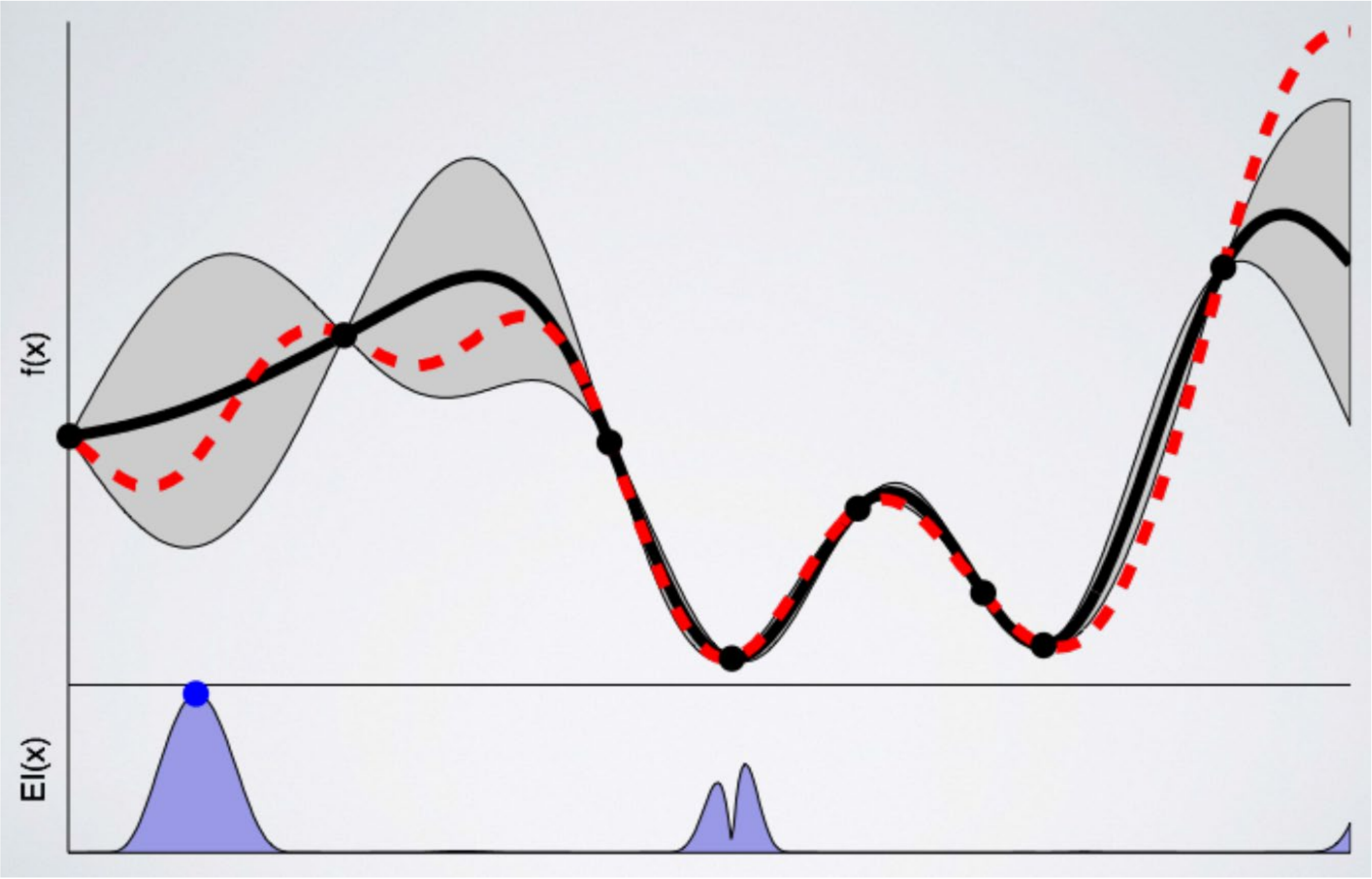
Bayesian Optimization: Illustration



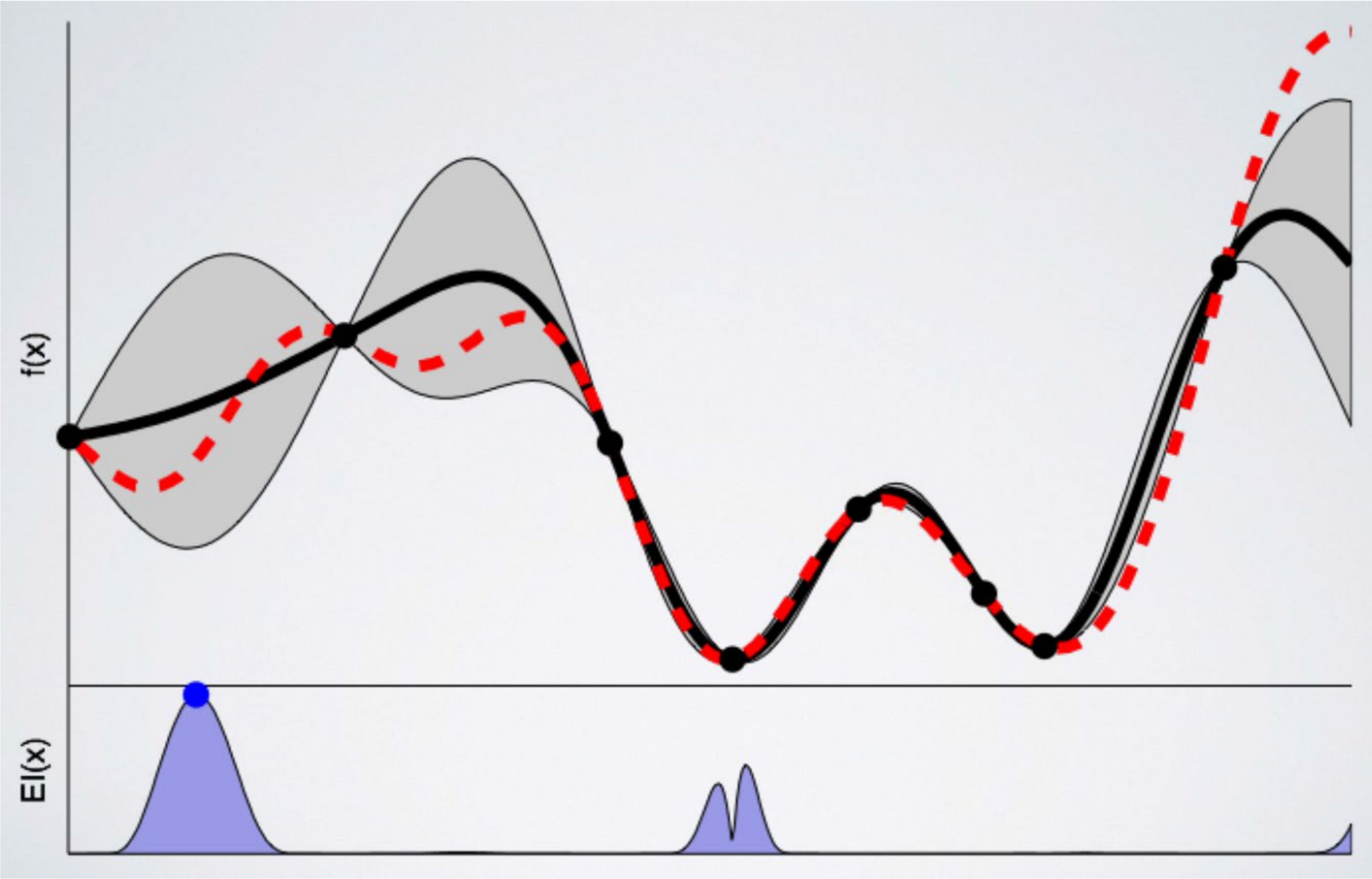
Bayesian Optimization: Illustration



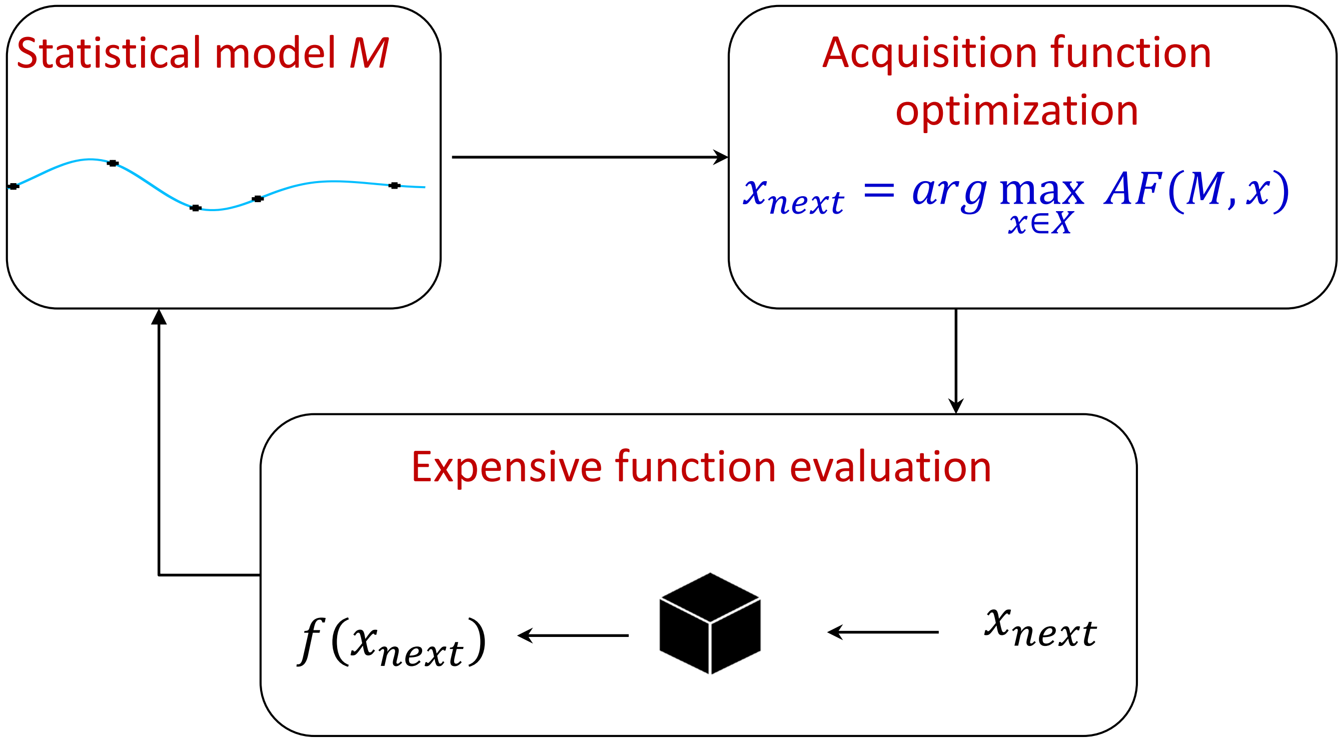
Bayesian Optimization: Illustration



Bayesian Optimization: Illustration

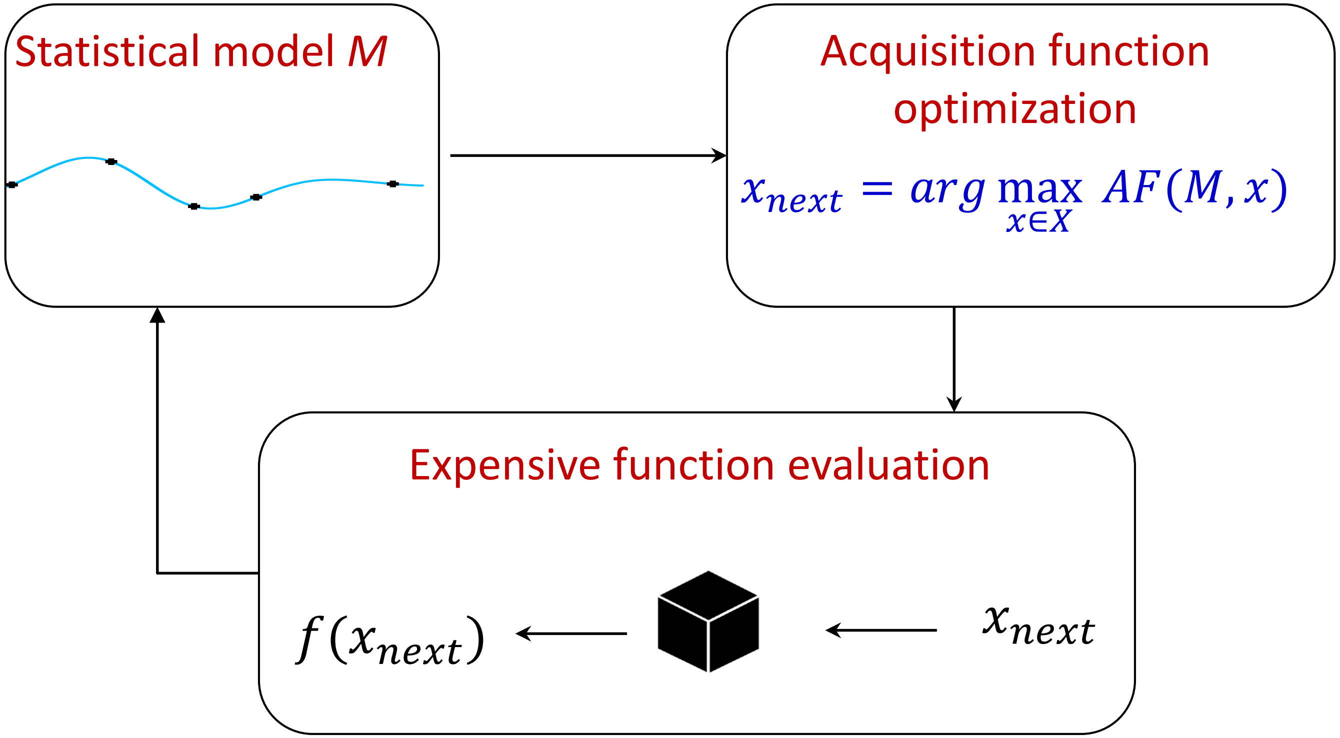


Bayesian Optimization: Three Key Elements



- Statistical model (e.g., Gaussian process)
- Acquisition function (e.g., Expected improvement)
- Acquisition function optimizer (e.g., local search)

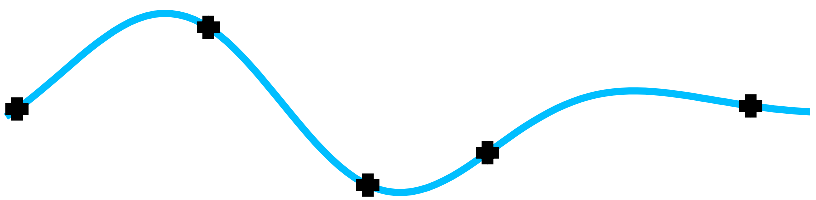
Bayesian Optimization: Three Key Elements



- Statistical model (e.g., Gaussian process)
- Acquisition function (e.g., Expected improvement)
- Acquisition function optimizer (e.g., local search)

BO needs a Probabilistic Model

- To make predictions on unknown input
- To quantify the uncertainty in predictions



- One popular class of such models are Gaussian Processes (also called GPs)

Gaussian Processes: What and Why?

Non-parametric, Bayesian and Kernel driven model

- Non-parametric → Flexibility
- Bayesian → Principled uncertainty estimation
- Kernel driven model → Specification of prior beliefs about rich function classes

Gaussian Process

- **Stochastic process definition**
  - Given any set of input points $\{x_1, x_2, \dots, x_m\}$, the output values follows a multi-variate Gaussian distribution

$$[f(x_1), f(x_2), f(x_3), \dots, f(x_m)] \sim \mathcal{N}(0, \Sigma)$$

- The covariance matrix Σ is given by a kernel function $k(x, x')$, i.e., $\Sigma_{ij} = k\left(x_i, x_j\right)$
  - Kernel captures the similarity between x and x'[1]
  - GPs are fully characterized by the kernel function[2]

Footnotes

1. For people aware of SVMs, it is the same kernel function.
2. Technically, there is also the mean function, but it is not as interesting for most applications.

Gaussian Process: Inference

- **Inference:** Given training data $\{(x_1, y_1), (x_2, y_2), \ \dots \ (x_m, y_m)\}$, the prediction for an unseen point x^*

$$\text{Prediction}(x^*) \sim \mathcal{N}(y^*, \sigma^*)$$

$$y^* = \ \boldsymbol{k}^* \ \boldsymbol{K}^{-1} \ \boldsymbol{Y}$$

$$\sigma^* = k(x^*, x^*) - \boldsymbol{k}^* \ \boldsymbol{K}^{-1} \boldsymbol{k}^*$$

$$\boldsymbol{k}^* = [k(x^*, x_1), k(x^*, x_2), \dots, k(x^*, x_m)]$$

$$\boldsymbol{K}_{ij} = k(x_i, x_j)$$

Gaussian Process: Training

- Training procedure: searching for (kernel) hyper-parameters by optimizing the marginal log-likelihood

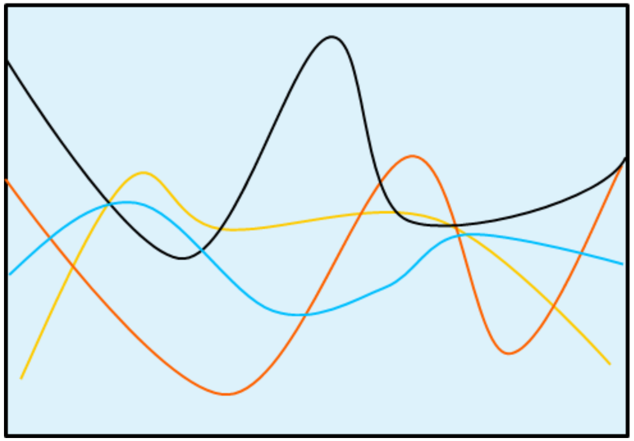
$$\log p(y) = -\frac{1}{2}\boldsymbol{Y}^T\boldsymbol{K}^{-1}\boldsymbol{Y} - \frac{1}{2}\log\det(\boldsymbol{K}) - \frac{n}{2}\log 2\pi$$

- Choice of kernel $k(x, x')$ is critical for good performance
  - Allows to incorporate domain knowledge (e.g., Morgan fingerprints in chemistry)
  - Matern kernel is a popular choice for continuous spaces

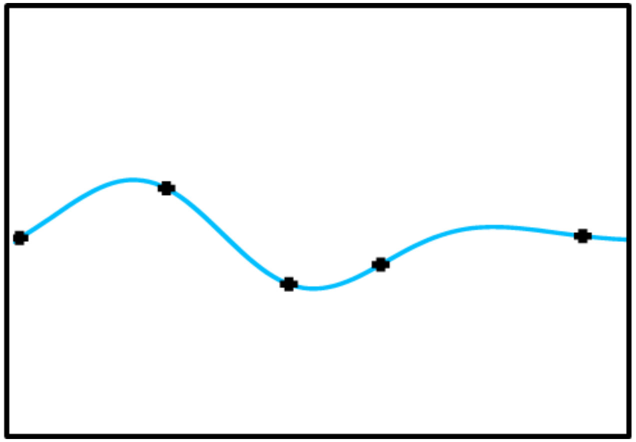
Gaussian Process: Two Views

- Function space view: distribution over functions
  - Function class is characterized by kernel

Prior



Posterior



- Weight space view: Bayesian linear regression in kernel's feature space

$$f(x) = w^T \tau(x)$$

$$k(x, x') = < \tau(x), \tau(x') >$$

Gaussian Processes: Challenges and Solutions

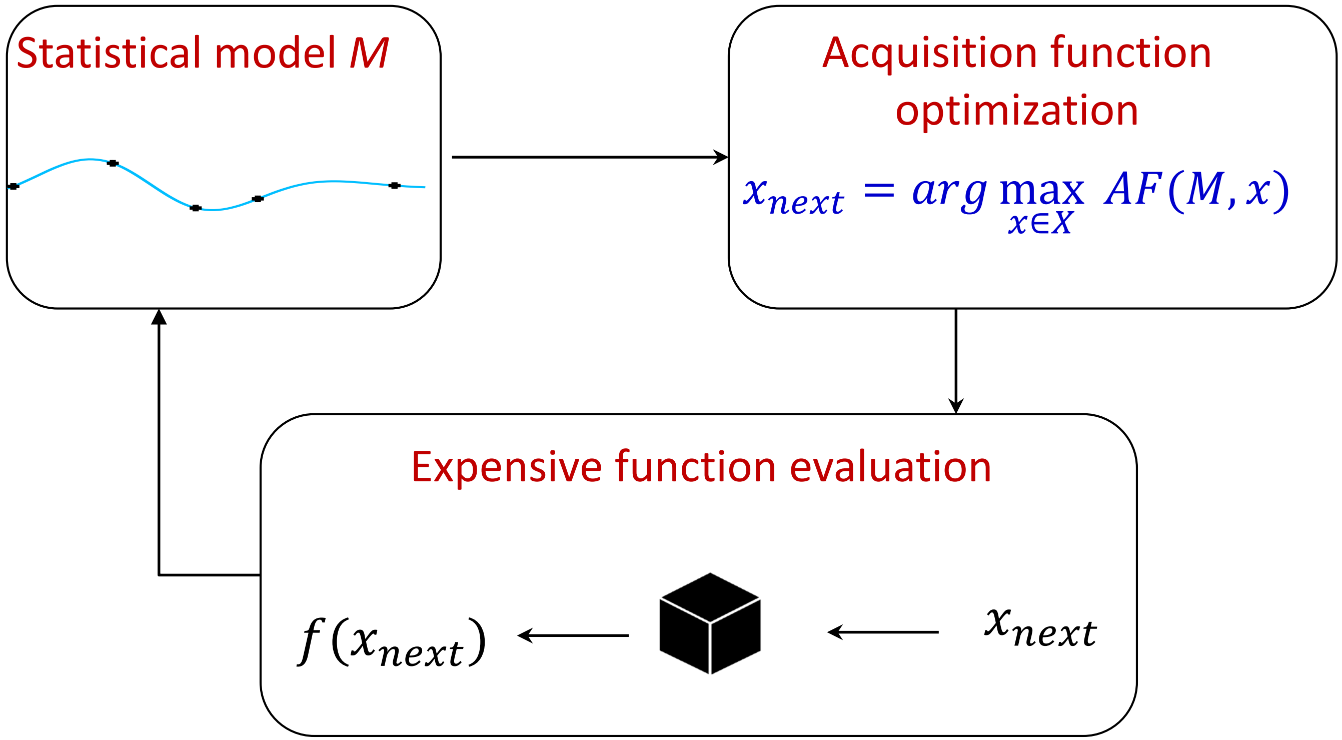
- Scalability: naive time complexity $O(n^3)$

$$\log p(y) = -\frac{1}{2}\boldsymbol{Y}^T\boldsymbol{K}^{-\mathbf{1}}\boldsymbol{Y} - \frac{1}{2}\log\det(\boldsymbol{K}) - \frac{n}{2}\log 2\pi$$

  - Solution: Sparse Gaussian processes

- Non-Gaussian likelihoods
  - No closed form expression, e.g., classification setting
  - Solution: Approximate inference

Bayesian Optimization: Three Key Elements

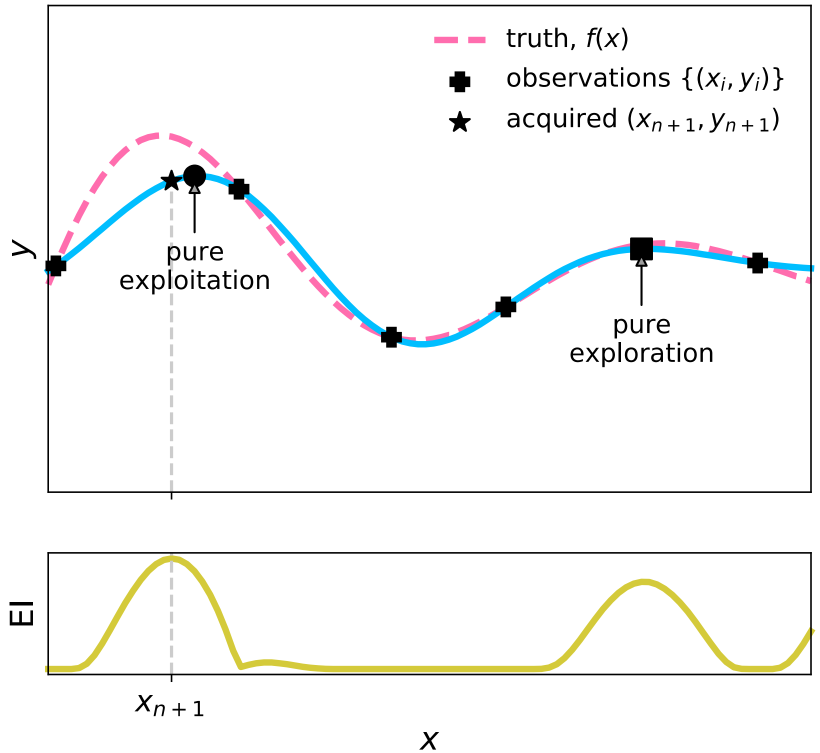


- Statistical model (e.g., Gaussian process)
- Acquisition function (e.g., Expected improvement)
- Acquisition function optimizer (e.g., local search)

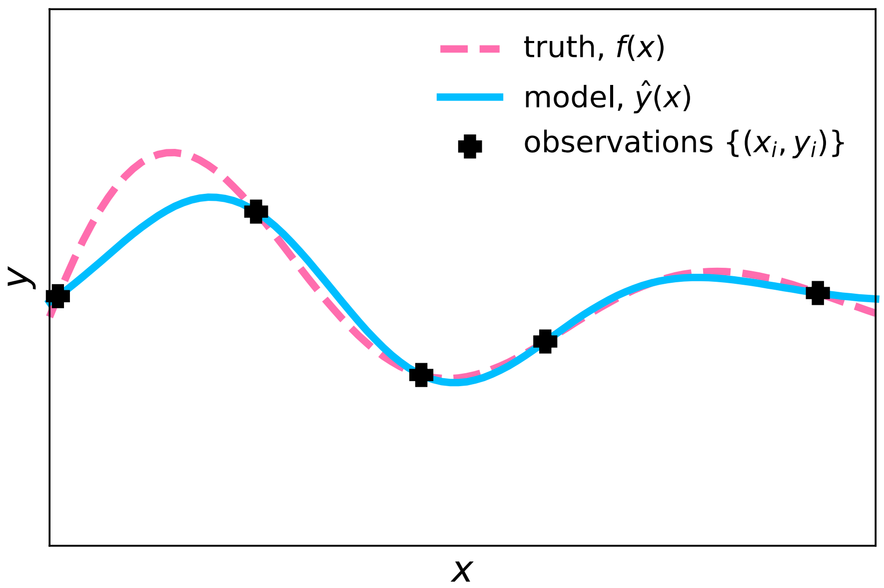
Acquisition Function

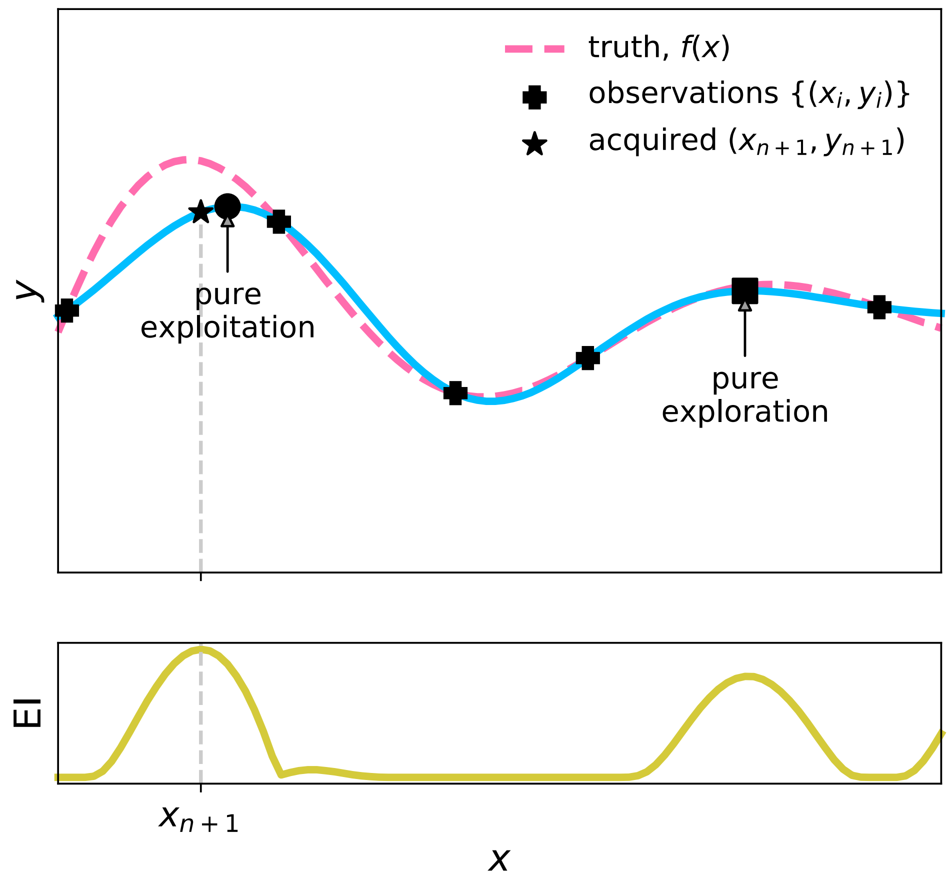
- Intuition: captures utility of evaluating an input

- Challenge: trade-off exploration and exploitation
  - Exploration: seek inputs with high variance
  - Exploitation: seek inputs with high mean



Acquisition Function: Illustration





Acquisition Function: Examples

- Upper Confidence Bound (UCB)
  - Selects input that maximizes upper confidence bound

$$AF(x) = y^*(x) + \beta\,\sigma^*(x)$$

- Expected Improvement (EI)
  - Selects input with highest expected improvement over the incumbent
- Thompson Sampling (TS)
  - Selects optimizer of a function sampled from the surrogate model's posterior
- Knowledge Gradient

Information-Theoretic Acquisition Functions

- Key principle: select inputs for evaluation which provide maximum information about the optimum

- Concretely, pick observations which quickly decrease the entropy of distribution over the optimum

$$AF(x) = \text{ Expected decrease in entropy}$$
$$AF(x) = \; H(\alpha \mid D) \; - E_y[H(\alpha|D \cup \{x, y\}$$
$$= \text{Information Gain}(\alpha; y)$$

- Design choices of α leads to different algorithms

Information-Theoretic Acquisition Functions

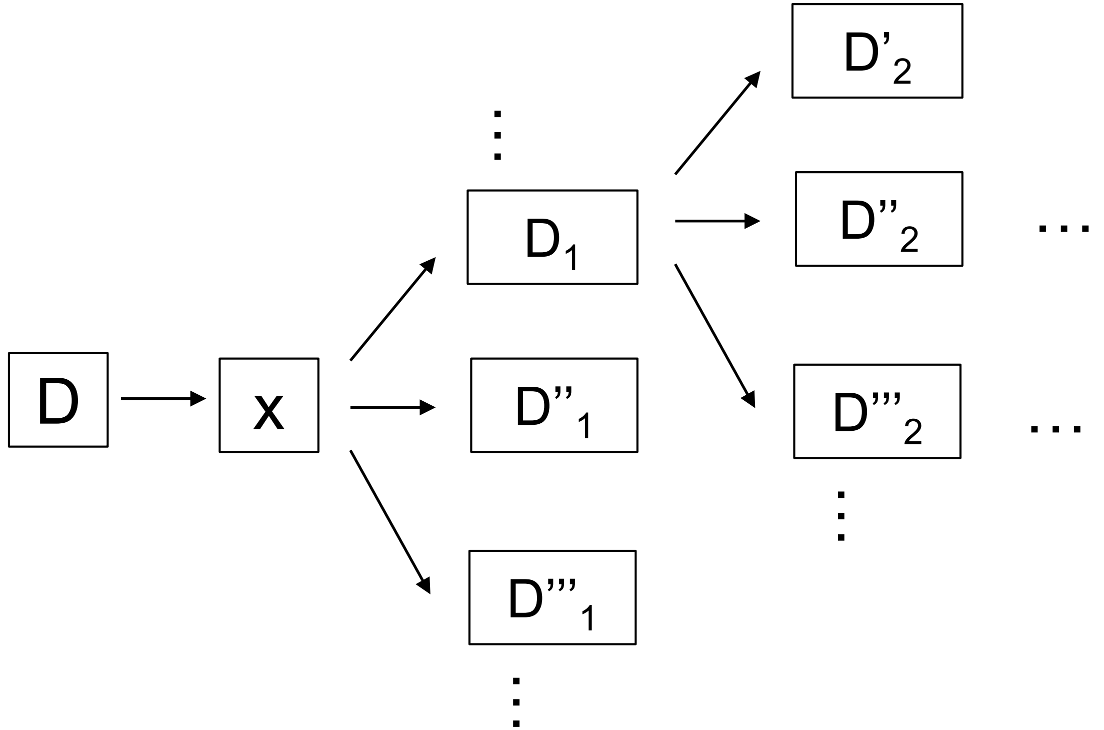
- Design choices of α leads to different algorithms

$$AF(x) = \text{Expected decrease in entropy}$$
$$AF(x) = H(\alpha \mid D) - E_y[H(\alpha | D \cup \{x, y\})]$$
$$= \text{Information Gain}(\alpha; y)$$

- α as input location of optima x^*
  - Entropy Search (ES) / Predictive Entropy Search (PES)
  - Intuitive but requires expensive approximations
- α as output value of optima y^*
  - Max-value Entropy Search (MES) and it's variants
  - Computationally cheaper and more robust

Non-Myopic / Lookahead Acquisition Functions

- Myopic acquisition functions (e.g., EI) reason about immediate utility
- Non-myopic variants consider BO as a MDP and reason about longer decision horizons



Non-Myopic / Lookahead Acquisition Functions

- Non-myopic variants consider BO as MDP and reason about longer decision horizons

$$u_k(x|D) = u_1(x|D) + E_y\left[\max_{x'} u_{t-1}(x'|D \cup \{x, y\})\right]$$

Bellman Recursion

Non-Myopic / Lookahead Acquisition Functions

- Non-myopic variants consider BO as MDP and reason about longer decision horizons

$$u_k(x|D) = u_1(x|D) + E_y\left[\max_{x'} u_{t-1}(x'|D \cup \{x, y\})\right]$$

- Challenge: curse of dimensionality

$$u_k(x|D) = u_1(x|D) + E_y\left[\max_{x1}\{u(x_1|D_1) + E_{y1}[\max_{x2}\{u(x_2|D_2) \ldots .\}]\}\right]$$

Non-Myopic / Lookahead Acquisition Functions

- Non-myopic variants consider BO as MDP and reason about longer decision horizons

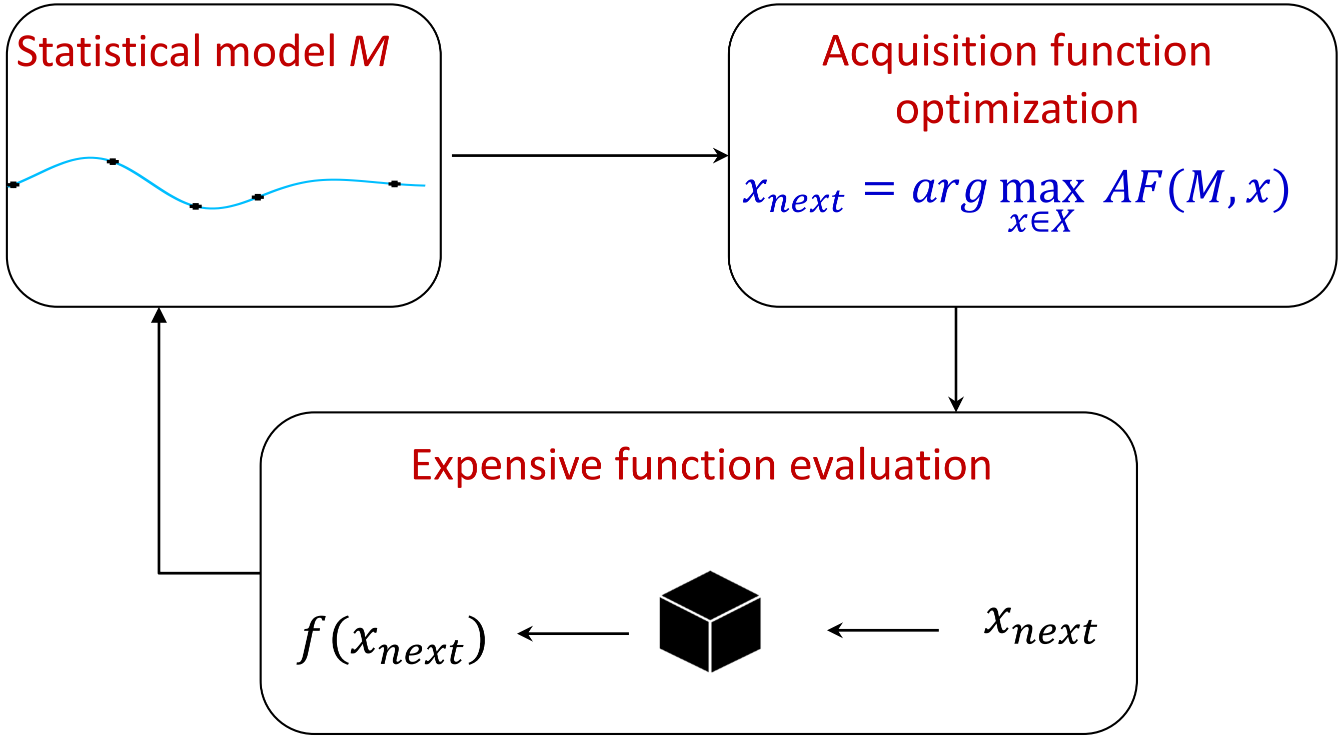
$$u_k(x|D) = u_1(x|D) + E_y\left[\max_{x'} u_{t-1}(x'|D \cup \{x, y\})\right]$$

- Challenge: curse of dimensionality

$$u_k(x|D) = u_1(x|D) + E_y\left[\max_{x1}\{u(x_1|D_1) + E_{y1}[\max_{x2}\{u(x_2|D_2) \ldots .\}]\}\right]$$

- Some solutions
  - Multi-step lookahead policies with approximations
  - Rollout based approximate dynamic programming

Bayesian Optimization: Three Key Elements



- Statistical model (e.g., Gaussian process)
- Acquisition function (e.g., Expected improvement)
- Acquisition function optimizer (e.g., local search)

Acquisition Function Optimizer

- Challenge: non-convex/multi-modal optimization problem

- Commonly used approaches
  - Space partitioning methods (e.g., DIRECT, LOGO)
  - Gradient based methods (e.g., Gradient descent)
  - Evolutionary search (e.g., CMA-ES)

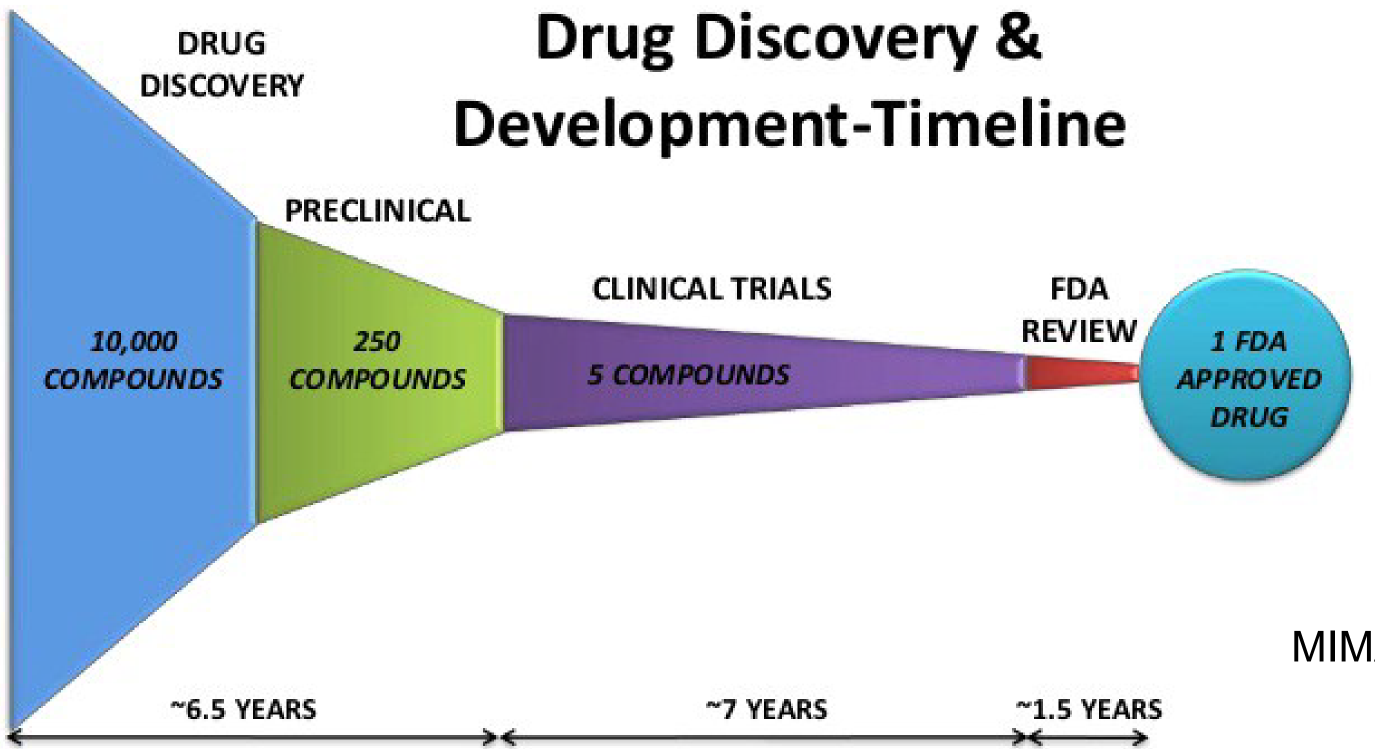
BO Software: BoTorch

- Scalability via automatic differentiation
  - PyTorch/GpyTorch
- Monte-Carlo acquisition functions
  - Express acquisition functions as expectations of utility functions
  - Compute expectations via Monte-Carlo sampling
  - Use the reparameterization trick to make acquisition functions differentiable
- Other software: Trieste (based on TensorFlow)
- Not actively maintained: GPyOpt, Spearmint

Questions ?

Bayesian Optimization over Combinatorial Spaces

Application #1: Drug/Vaccine Design

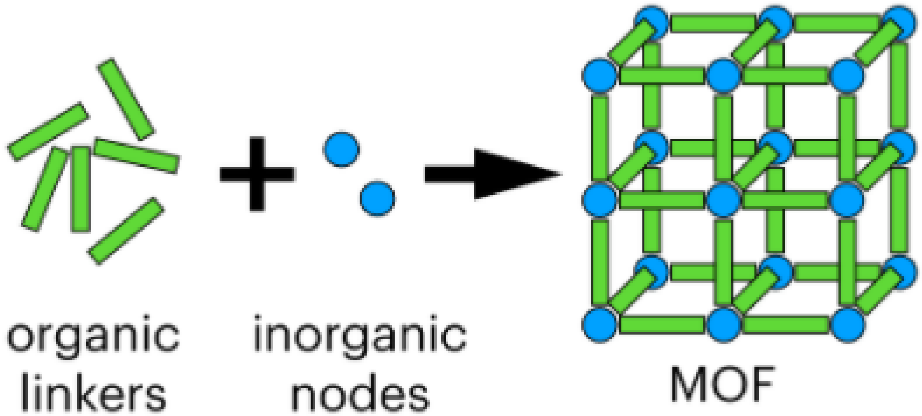


Credit:

MIMA healthcare

- Accelerate the discovery of promising designs

Application #2: Nanoporous Materials Design

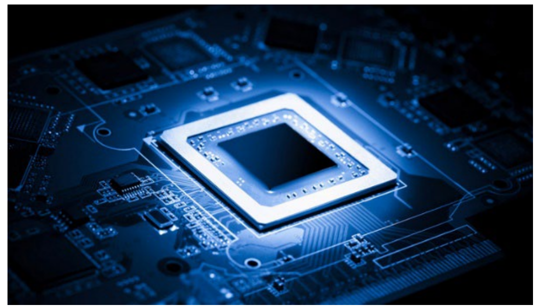


- **Sustainability applications**
  - Storing gases (e.g., hydrogen powered cars)
  - Separating gases (e.g., carbon dioxide from flue gas of coalfired power plants)
  - Detecting gases (e.g., detecting pollutants in outdoor air)

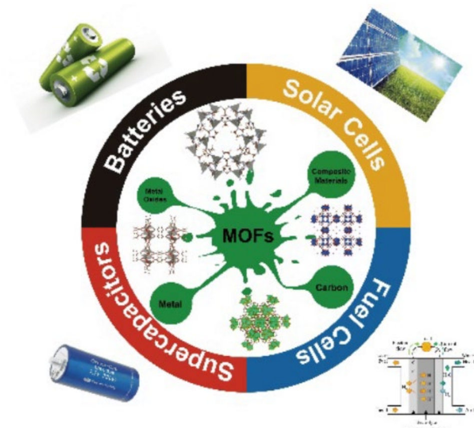
Combinatorial BO: The Problem

- **Goal:** find optimized combinatorial structures

Drug design



Hardware design



Material design

- Many other science and engineering applications

Combinatorial BO: The Problem

- **Given:** a combinatorial space of structures X (e.g., sequences, graphs) and an expensive black-box function $f(x \in X)$ to evaluate each structure $x \in X$

- **Find:** optimized combinatorial structure x^*

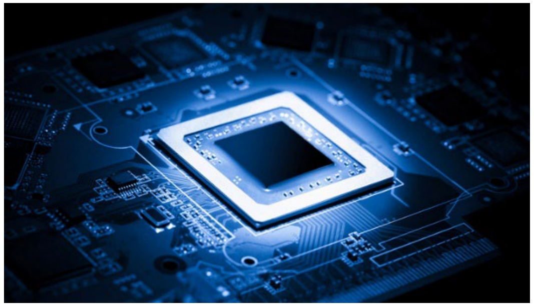
$$\boxed{x^* = arg\max_{x \in X} f(x)}$$

- **Evaluation:** number of function evaluations to (approximately) optimize $f(x)$

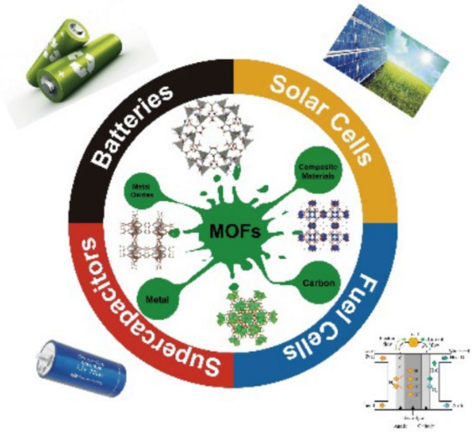
Combinatorial BO: Challenges

- **Goal:** find optimized combinatorial structures

Drug design



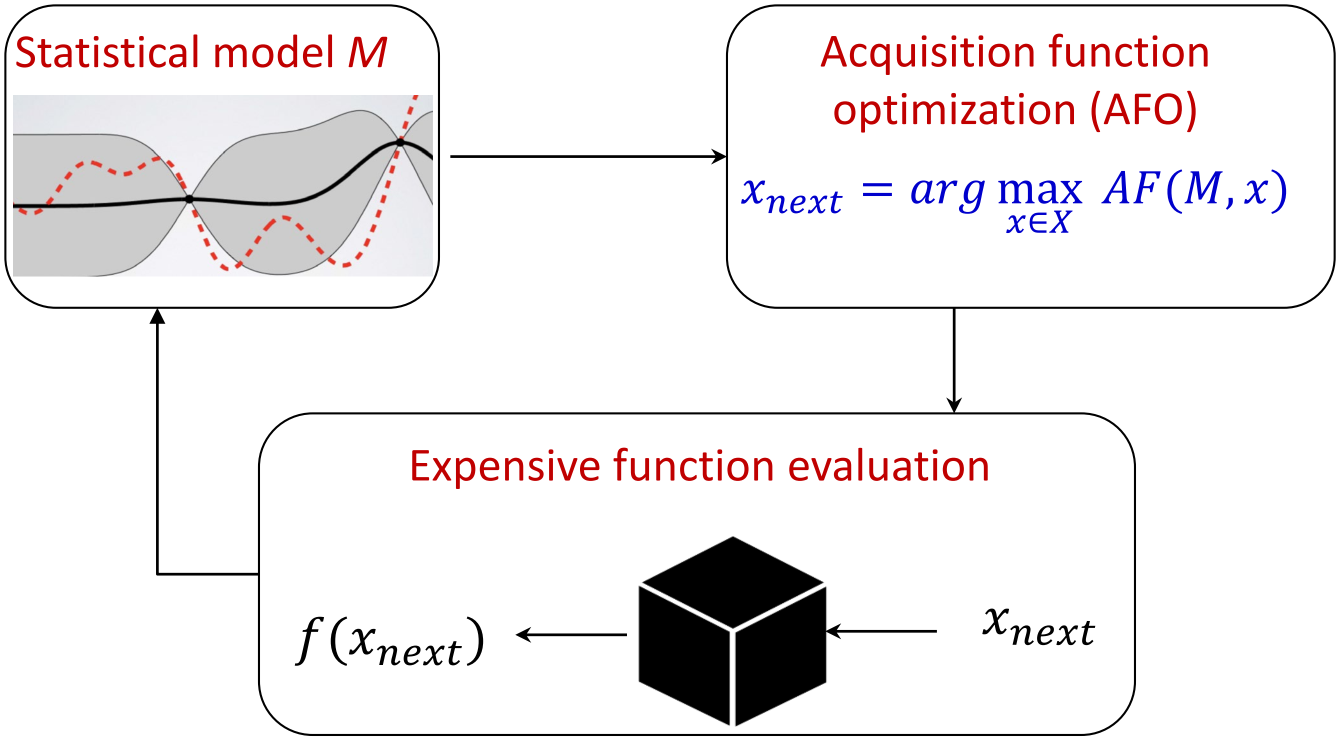
Hardware design



Material design

- **Challenges**
  - Evaluating each candidate design is expensive
  - Large combinatorial space of designs (e.g., sequences, graphs)

Combinatorial BO: Technical Challenges



- Effective modeling over combinatorial structures (e.g., sequences, graphs)
- Solving hard combinatorial optimization problem to select next structure

Definition of Combinatorial Space

- **Space of binary structures** $X = \{0,1\}^n$
  - Each structure $x \in X$ be represented using n binary variables $x_1, x_2, \dots, x_n$

- **Categorical variables**
  - x_i can take more than two candidate values

- **How to deal with categorical variables?**
  - Option 1: Encode them as binary variables (a common practice)
  - Option 2: Modeling and reasoning over categorical variables

Combinatorial BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - BOCS [Baptista et al., 2018]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011] and COMBO [Oh et al., 2019]

- Complex statistical models and tractable/accurate AFO
  - L2S-DISCO [Deshwal et al., 2020] and MerCBO [Deshwal et al., 2021]
  - Reduction to continuous BO [Gómez-Bombarelli et al., 2018]…

Aside: Combinatorial BO vs. Structured Prediction

- **Structured prediction (SP)** [Lafferty et al., 2001] [Bakir et al., 2007]
  - Generalization of classification to structured outputs (e.g., sequences, trees, and graphs)
    - POS tagging, parsing, information extraction, image segmentation
  - CRFs, Structured Perceptron, Structured SVM

- Complexity of cost function vs. tractability of inference
  - Simple cost functions (e.g., first-order) and tractable inference
  - Complex cost functions (e.g., higher-order) and heuristic inference
  - Learning to search for SP [Daume' et al., 2009] [Doppa et al., 2014]

- **Key Difference:** Small data vs. big data setting

Combinatorial BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - BOCS [Baptista et al., 2018]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011] and COMBO [Oh et al., 2019]

- Complex statistical models and tractable/accurate AFO
  - L2S-DISCO [Deshwal et al., 2020] and MerCBO [Deshwal et al., 2021]
  - Reduction to continuous BO [Gómez-Bombarelli et al., 2018]…

BOCS Algorithm [Baptista et al., 2018]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of up to Quadratic (second-order) terms
  - $\phi(x) = [x_1, x_2, \dots, x_d, x_1 . x_2, x_1 . x_3, \dots, x_{d-1} . x_d]$

- Thompson sampling as acquisition function

- Acquisition function optimization
  - Binary quadratic program

$$x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A\, x$$

BOCS Algorithm [Baptista et al., 2018]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of up to Quadratic (second-order) terms
  - $\phi(x) = [x_1, x_2, \ldots, x_d, x_1.x_2, x_1.x_3, \ldots, x_{d-1}.x_d]$

May not be sufficient to capture desired dependencies

- Thompson sampling as acquisition function

- Acquisition function optimization
  - Binary quadratic program

$$x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A\, x$$

BOCS Algorithm [Baptista et al., 2018]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of up to Quadratic (second-order) terms
  - $\phi(x) = [x_1, x_2, \ldots, x_d, x_1.x_2, x_1.x_3, \ldots, x_{d-1}.x_d]$

- Thompson sampling as acquisition function

- Acquisition function optimization
  - Binary quadratic program

Cannot handle declarative constraints for valid structures

$$x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A\, x$$

Combinatorial BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - BOCS [Baptista et al., 2018]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011] and COMBO [Oh et al., 2019]

- Complex statistical models and tractable/accurate AFO
  - L2S-DISCO [Deshwal et al., 2020] and MerCBO [Deshwal et al., 2021]
  - Reduction to continuous BO [Gómez-Bombarelli et al., 2018]…

SMAC Algorithm [Hutter et al., 2010, 2011]

- Random forest as surrogate model
  - works naturally for categorical variables
  - Prediction/Unce inty (= empirical mean/variance over trees)

Uncertainty estimates
can be poor

- Expected impr n function

- Hand-designed local search with restarts for AFO

SMAC Algorithm [Hutter et al., 2010, 2011]

- Random forest as surrogate model
  - works naturally for categorical variables
  - Prediction/Uncertainty (= empirical mean/variance over trees)

- Expected improvement as acquisition function

- Hand-designed local search with restarts for AFO

Can potentially get stuck in local optima

COMBO Algorithm [Oh et al., 2019]

- GP with diffusion kernel [Kondor and Lafferty 2002]
  - Requires a graph representation of the input space X

$$\boxed{K(V,V) = exp(-\beta L(G))}$$

- Expected improvement as acquisition function

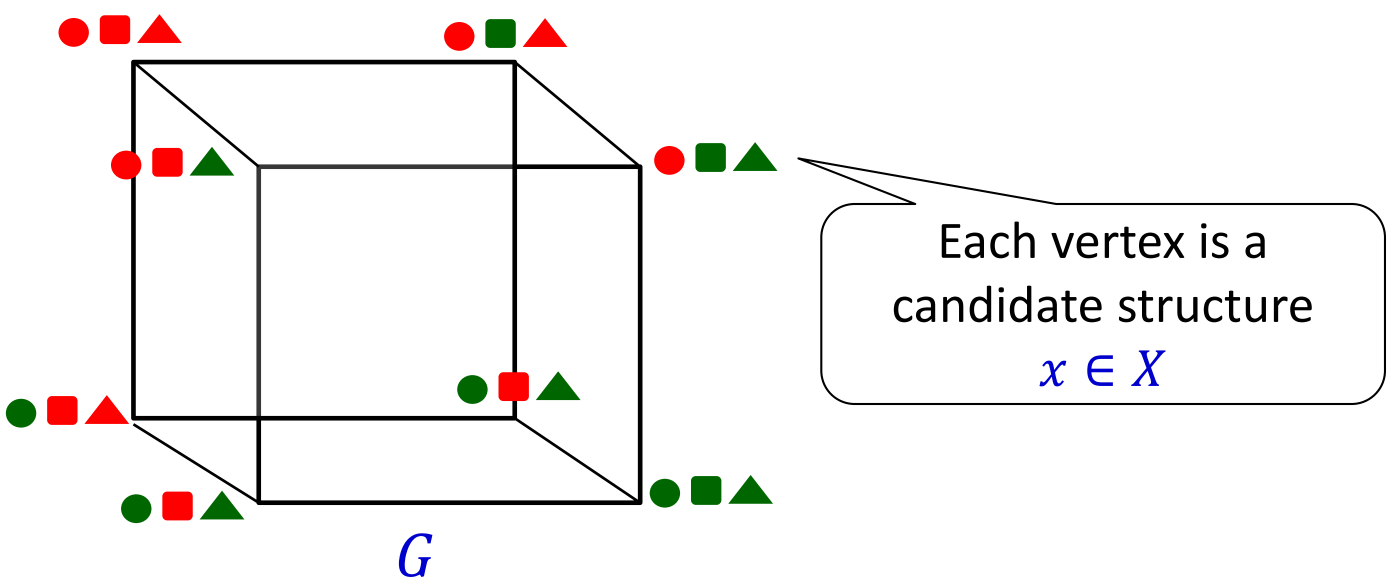
- Local search with random restarts for AFO

COMBO Algorithm [Oh et al., 2019]

- GP with diffusion kernel [Kondor and Lafferty 2002]
  - Requires a graph representation of the input space X

$$K(V,V) = exp(-\beta L(G))$$

- **Combinatorial graph representation** [Oh et al., 2019]

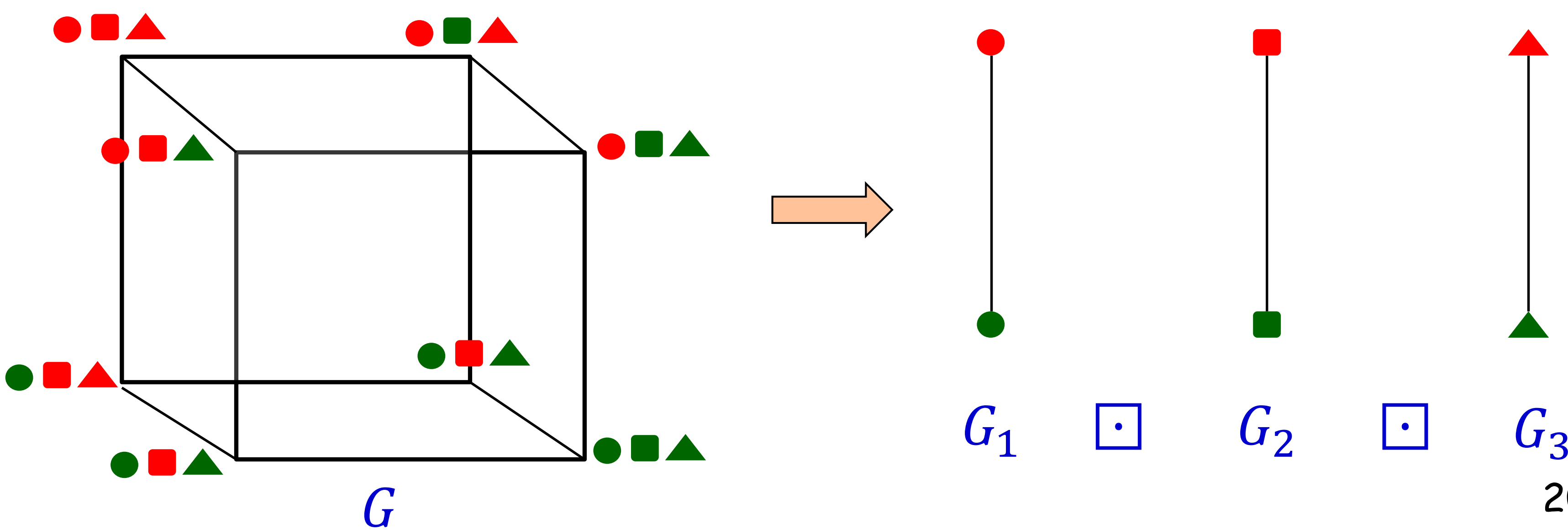


COMBO Algorithm [Oh et al., 2019]

- GP with diffusion kernel [Kondor and Lafferty 2002]
  - Requires a graph representation of the input space X

$$K(V,V) = exp(-\beta L(G))$$

- **Combinatorial graph representation** [Oh et al., 2019]
  - Graph Cartesian product of subgraphs



COMBO Algorithm [Oh et al., 2019]

- GP with diffusion kernel [Kondor and Lafferty 2002]
  - Requires a graph representation of the input space X

Cannot use SOTA acquisition functions if we cannot sample functions from GP posterior

- Expected improvement as acquisition function

- Local search with random restarts for AFO

COMBO Algorithm [Oh et al., 2019]

- GP with diffusion kernel [Kondor and Lafferty 2002]
  - Requires a graph representation of the input space X

$$\boxed{K(V,V) = exp(-\beta L(G))}$$

- Expected improvement as acquisition function

- Local search with random restarts for AFO

Can potentially get stuck in local optima

Combinatorial BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - BOCS [Baptista et al., 2018]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011] and COMBO [Oh et al., 2019]

- Complex statistical models and tractable/accurate AFO
  - L2S-DISCO [Deshwal et al., 2020] and MerCBO [Deshwal et al., 2021]
  - Reduction to continuous BO [Gómez-Bombarelli et al., 2018]…

MerCBO Algorithm [Deshwal et al., 2021]

- Same surrogate model as COMBO
  - GP with discrete diffusion kernel and graph representation
- Thompson sampling as acquisition function
  - Mercer features allow sampling functions from GP posterior
- Acquisition function optimization
  - Binary quadratic program
  - Parametrized submodular relaxation (PSR) solver

$$\boxed{x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A \, x}$$

MerCBO Algorithm [Deshwal et al., 2021]

- Same surrogate model as COMBO
  - GP with discrete diffusion kernel and graph representation

- Thompson sampling as acquisition function
  - Mercer features allow sampling functions from GP posterior

- Acquisition function optimization
  - Binary quadratic program
  - Parametrized submodular relaxation (PSR) solver

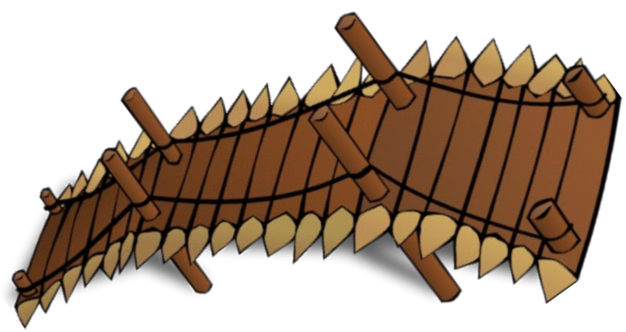
$$x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A\, x$$

MerCBO: Acquisition Function

- Mercer features allow sampling functions from GP posterior

- Missing puzzle to leverage prior acquisition functions
  - Thompson Sampling (TS)
  - Predictive Entropy Search (PES)
  - Max-value Entropy Search (MES)
  - …

BO for continuous spaces



BO for discrete spaces

MerCBO: Mercer Features

- **Key Idea:** exploit the structure of combinatorial graph G to compute its eigenspace in closed-form

MerCBO: Mercer Features

- **Key Idea:** exploit the structure of combinatorial graph G to compute its eigenspace in closed-form

- Graph Laplacian $L(G)$ decomposes over those of sub-graphs

$$L(G) = L(G_1) \oplus L(G_2) \oplus L(G_3)$$

$\oplus$ is Kronecker sum operator

MerCBO: Mercer Features

- **Key Idea:** exploit the structure of combinatorial graph G to compute its eigenspace in closed-form

- Graph Laplacian $L(G)$ decomposes over those of sub-graphs

$$L(G) = L(G_1) \oplus L(G_2) \oplus L(G_3)$$

$\oplus$ is Kronecker sum operator

- [Hammack et al., 2011] Given two graphs G_1 and G_2 with the eigenspace of their Laplacians being $\{\lambda_1, U_1\}$ and $\{\lambda_2, U_2\}$ respectively, the eigenspace of $L(G_1 \boxdot G_2)$ is given by $\{\lambda_1 \bowtie \lambda_2, U_1 \otimes U_2\}$.

MerCBO: Mercer Features

- **Key Idea:** exploit the structure of combinatorial graph G to compute its eigenspace in closed-form

- Graph Laplacian $L(G)$ decomposes over those of sub-graphs

$$L(G) = L(G_1) \oplus L(G_2) \oplus L(G_3)$$

$\oplus$ is Kronecker sum operator

- [Hammack et al., 2011] Given two graphs G_1 and G_2 with the eigenspace of their Laplacians being $\{\lambda_1, U_1\}$ and $\{\lambda_2, U_2\}$ respectively, the eigenspace of $L(G_1 \boxdot G_2)$ is given by $\{\lambda_1 \bowtie \lambda_2, U_1 \otimes U_2\}$.

- Each G_i has eigenvalue {0,2} and eigenvectors {[1, 1], [1, -1]}

MerCBO: Mercer Features

- **Key Idea:** exploit the structure of combinatorial graph G to compute its eigenspace in closed-form

- **Eigenvalue set:** $\{0, 2, \ldots, 2n\}$
  - j^{th} eigenvalue occurs with $\binom{n}{j}$ multiplicity

- **Eigenvector set:** Hadamard matrix (H) of order 2^n

$$H_{ij} = (-1)^{\langle r_i, r_j \rangle}$$

MerCBO: Mercer Features

$$K(x_1, x_2) = \sum_{i=0}^{2^n - 1} e^{-\beta\lambda_i} u_i([x_1])\, \mathbf{u_j}([\mathbf{x_2}])$$

$$K(x_1, x_2) = \sum_{i=0}^{2^n - 1} e^{-\beta\lambda_i} \; -1^{<r_i, x_1>} \; -1^{<r_i, x_2>}$$

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2)$$

$$\boxed{\phi(x)_i = \{\sqrt{e^{-\beta\lambda_i}} \; -1^{<r_i, x>}\}}$$

MerCBO: Mercer Features

$$K(x_1, x_2) = \sum_{i=0}^{2^n - 1} e^{-\beta\lambda_i} \; -1^{<r_i, x_1>} \; -1^{<r_i, x_2>}$$

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2)$$

$$\phi(x)_i = \{\sqrt{e^{-\beta\lambda_i}} \; -1^{<r_i, x>}\}$$

j^{th} order Mercer features: first j distinct eigenvalues

MerCBO Algorithm [Deshwal et al., 2021]

- Same surrogate model as COMBO
  - GP with discrete diffusion kernel and graph representation
- Thompson sampling as acquisition function
  - Mercer features allow sampling functions from GP posterior
- Acquisition function optimization
  - Binary quadratic program
  - Parametrized submodular relaxation (PSR) solver

$$x_{next} = arg \max_{x \in \{0,1\}^d} b^T x + x^T A\, x$$

MerCBO: Acquisition Function Optimization

$$x_{next} = arg \max_{x \in \{0,1\}^n} \; b^T x + x^T A\, x$$

- **Parametrized Submodular Relaxation (PSR) solver**
  - Construct a Λ-parametrized submodular relaxation

$$\boxed{h_\Lambda(x) + x^T A^- x} \quad \leq \quad \boxed{x^T A x + b^T x}$$

Solve using min. graph cut algorithms

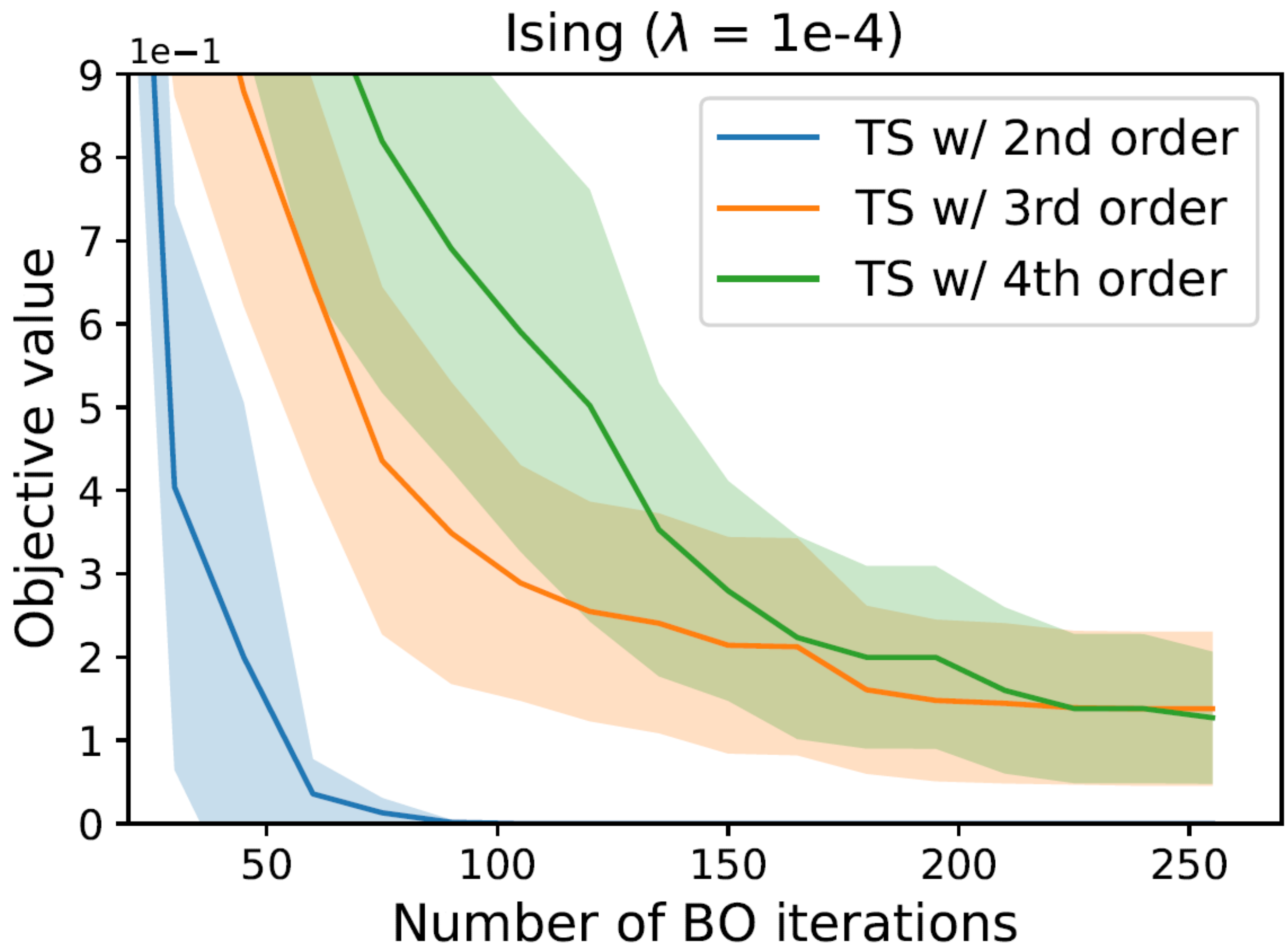
  - Optimize the relaxation over Λ

$$\boxed{h_{\Lambda_1}(x) + x^T A^- x} \quad \leq \quad \boxed{h_{\Lambda_2}(x) + x^T A^- x} \quad \leq \; \ldots. \quad \boxed{x^T A x + b^T x}$$

Inspired by work on prescriptive price optimization [Ito and Fujimaki, 2016]

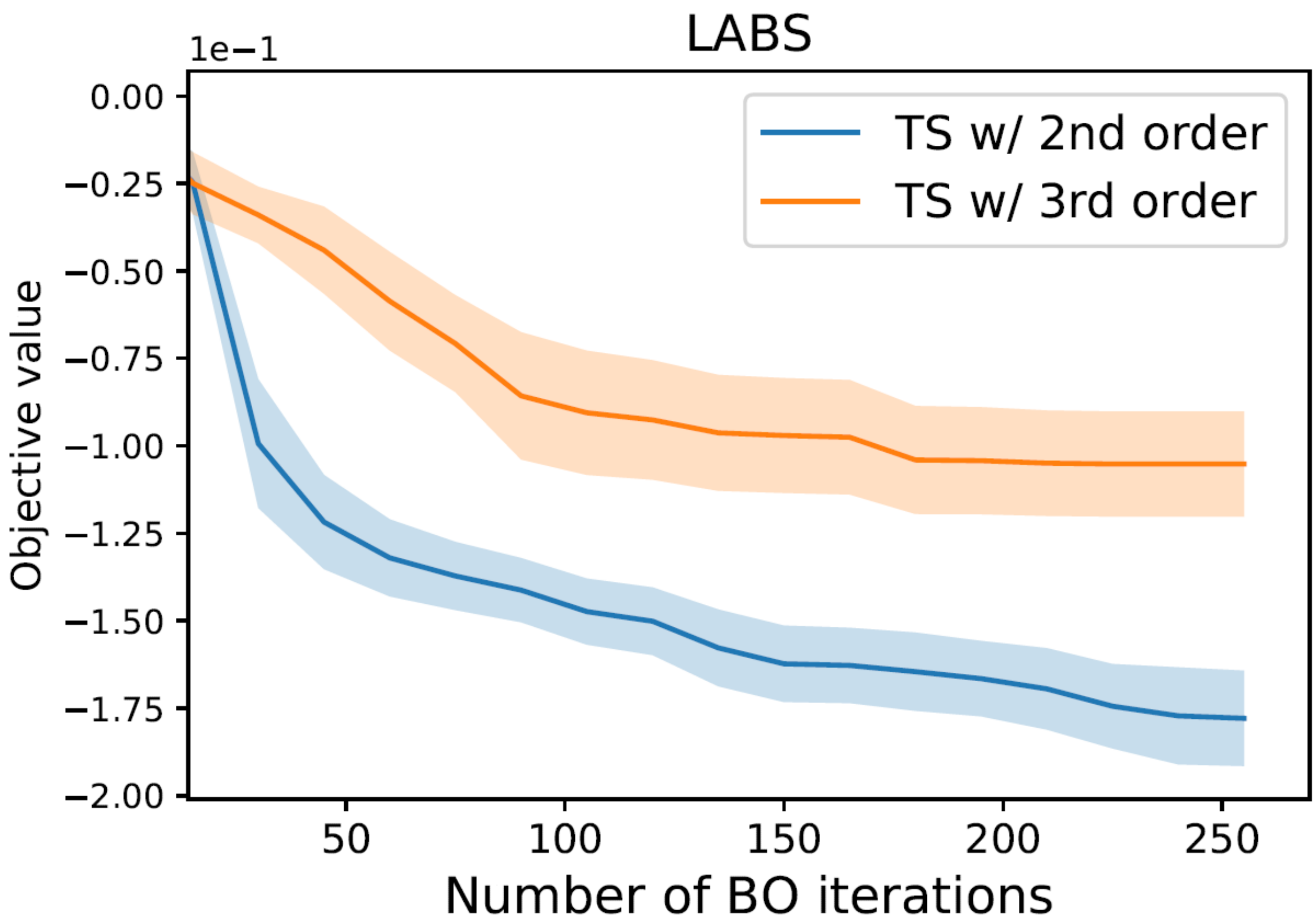
MerCBO Results #1: Order of Features

- Second-order features provide the best trade-off
  - Tractability and good overall BO performance



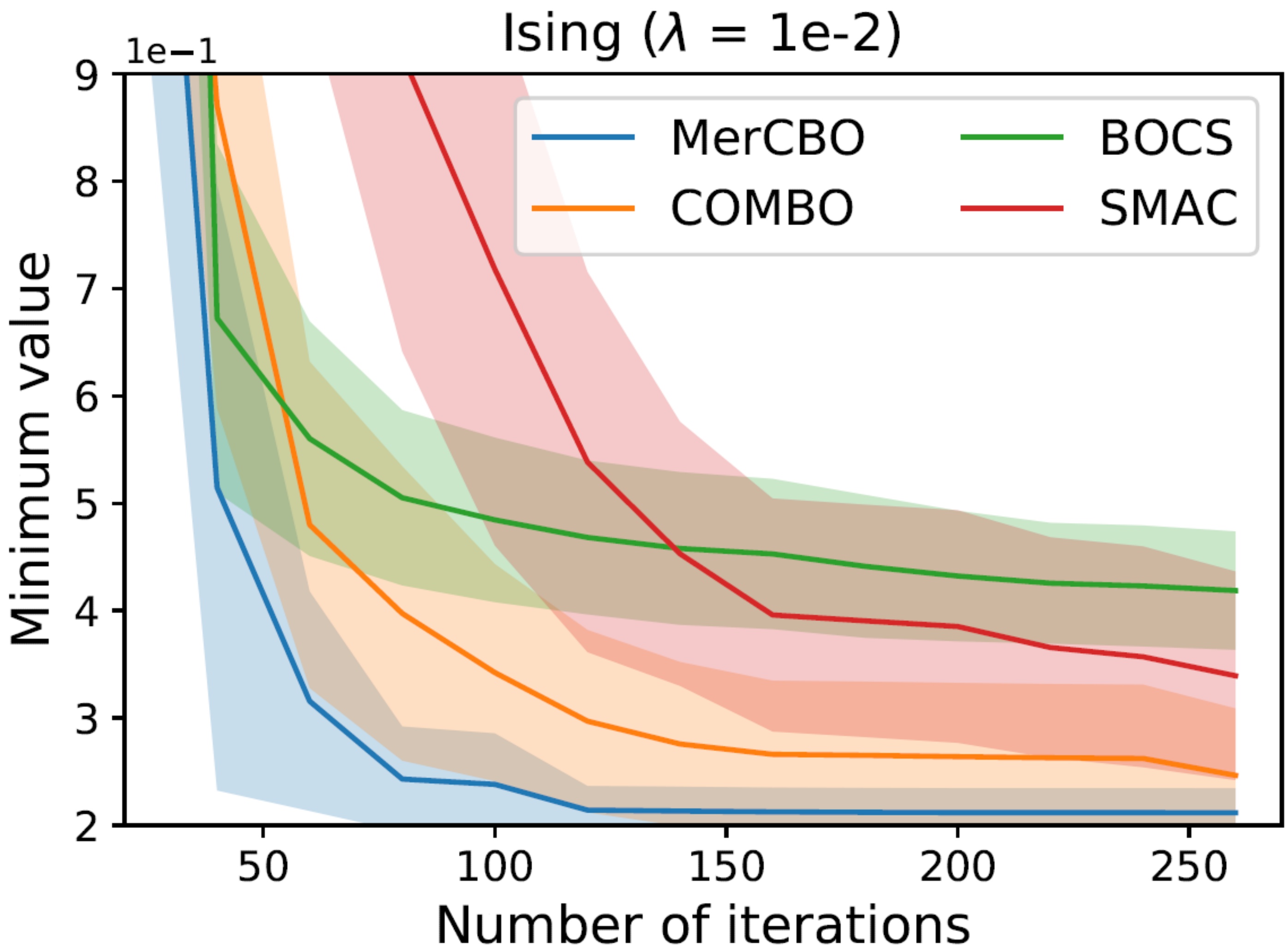
MerCBO Results #1: Order of Features

- Second-order features provide the best trade-off
  - Tractability and good overall BO performance



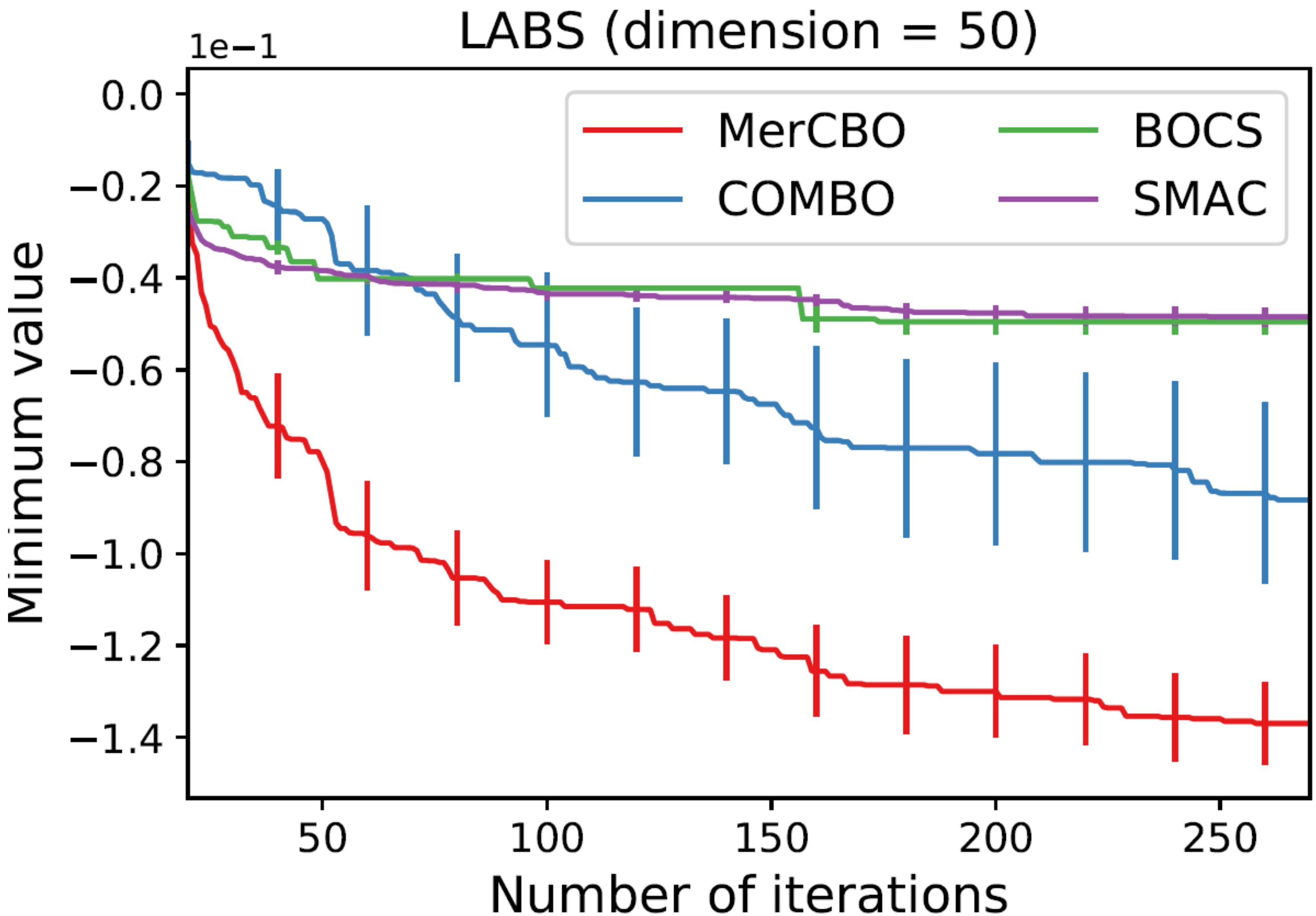
MerCBO Results #2: Comparison with State-of-the-art

- MerCBO outperforms prior methods



MerCBO Results #2: Comparison with State-of-the-art

- MerCBO outperforms prior methods



MerCBO for Biological Sequence Design

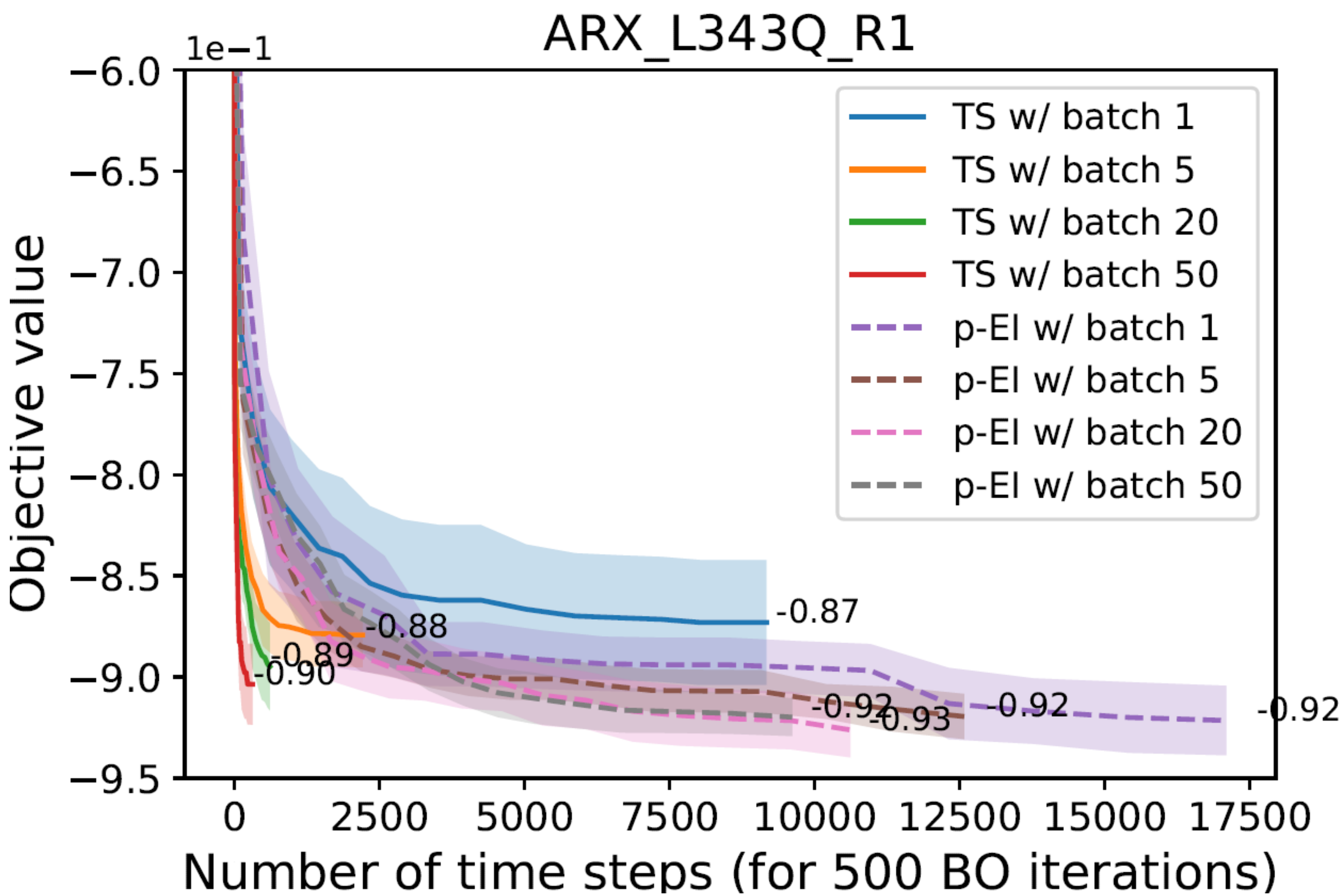
- Design of optimized biological structures such as DNA and proteins have many medical applications

Biological Sequence Design: Three Desiderata

- **Diversity**
  - uncover a diverse set of structures
- **Parallel experiments**
  - Select a batch of structures for evaluation in each round
- **Real-time accelerated design**
  - Use parallel experimental resources to accelerate optimization

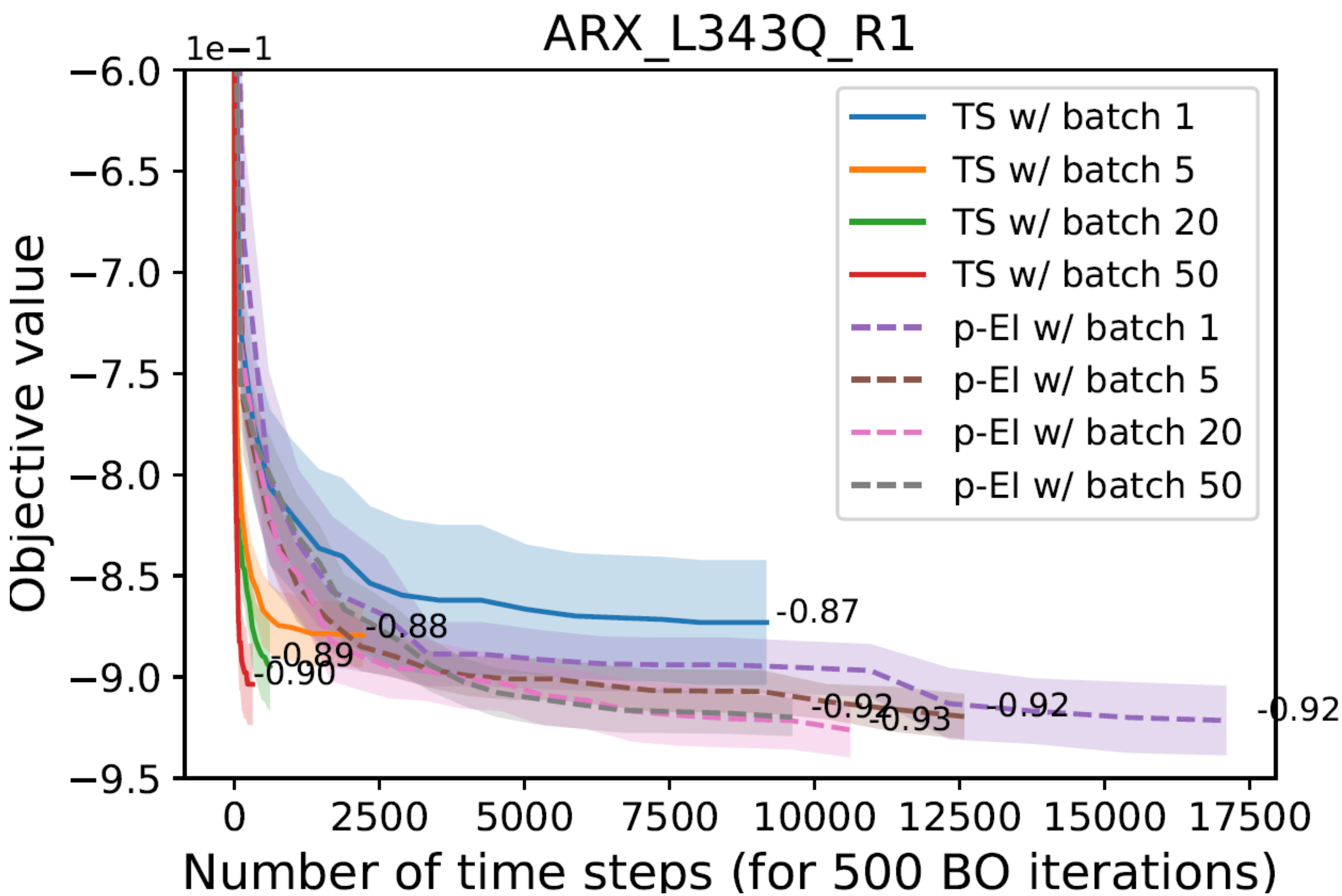
MerCBO Results #3: Real-time acceleration

- TS is better than EI for real-time accelerated design



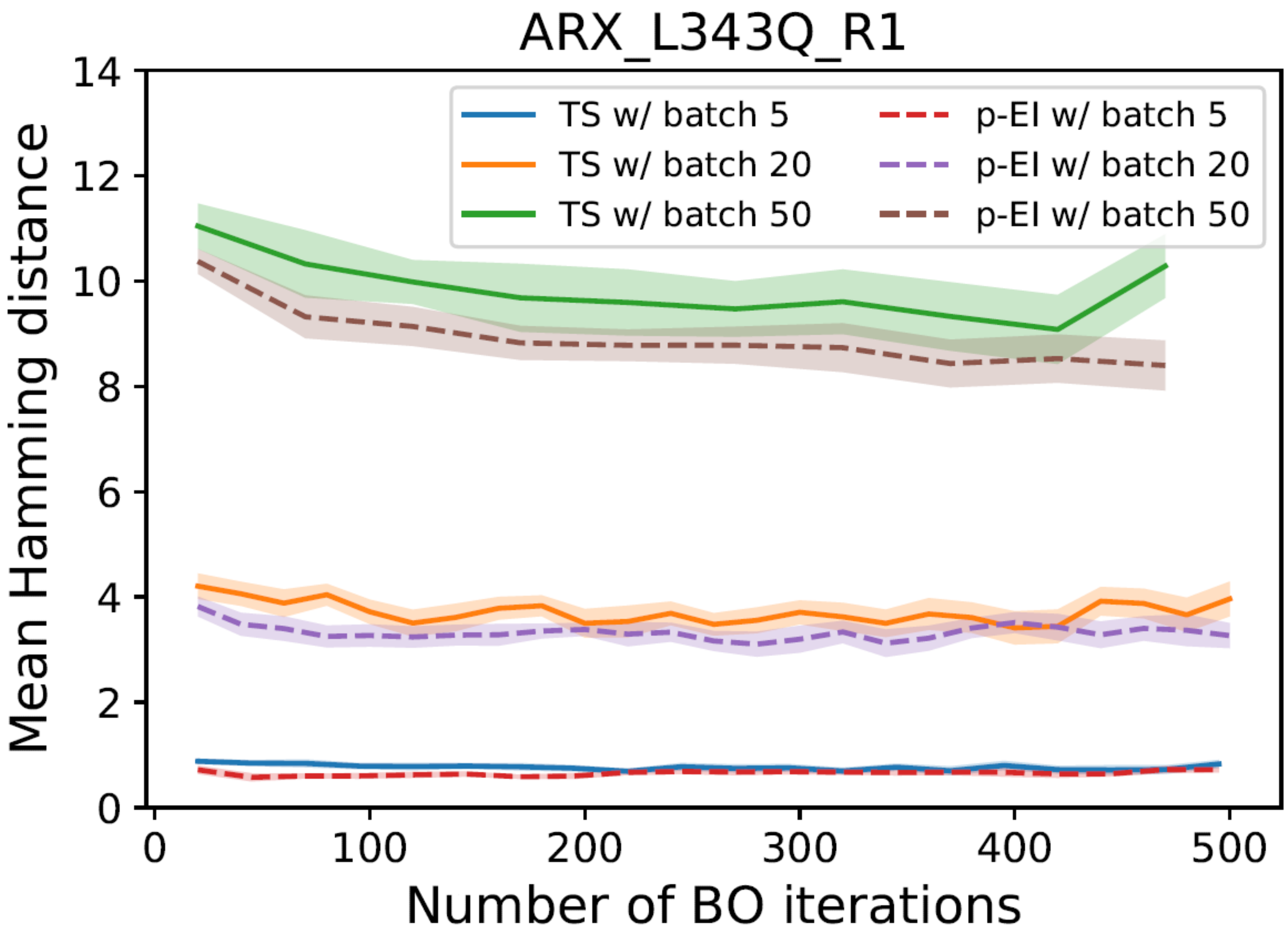
MerCBO Results #3: Real-time acceleration

- TS improvement over EI increases with batch size



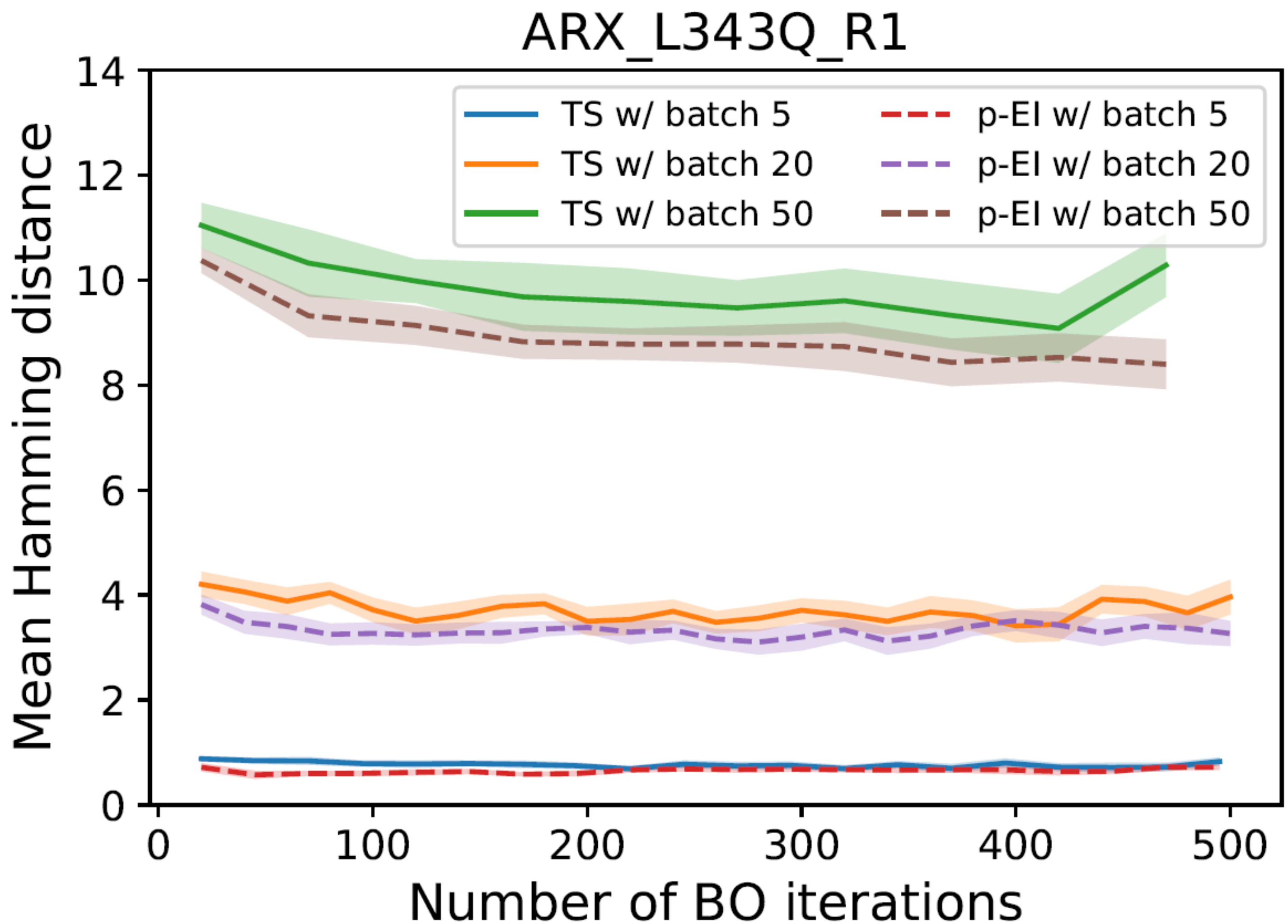
MerCBO Results #4: Diversity of sequences

- TS is better than EI for diversity of sequences



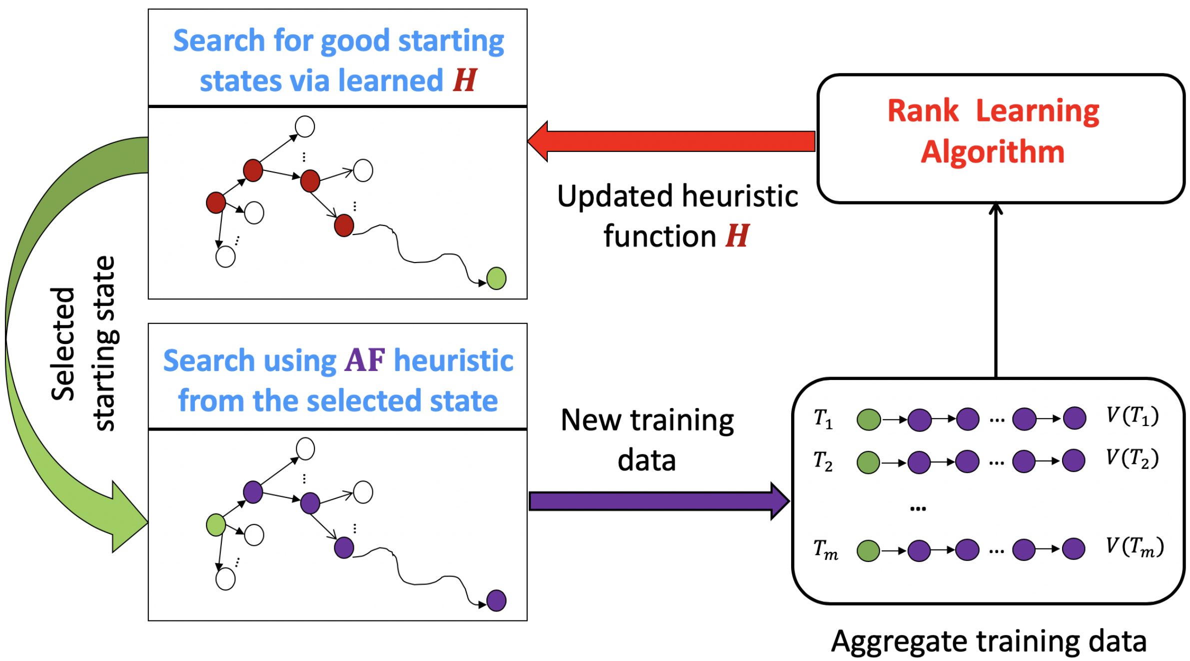
MerCBO Results #4: Diversity of sequences

- TS improvement over EI increases with batch size



Learning to Search Framework [Deshwal et al., 2021]

- Use machine learning to improve the accuracy of search
  - Continuously update the search control knowledge using the training data generated from the previous search experience



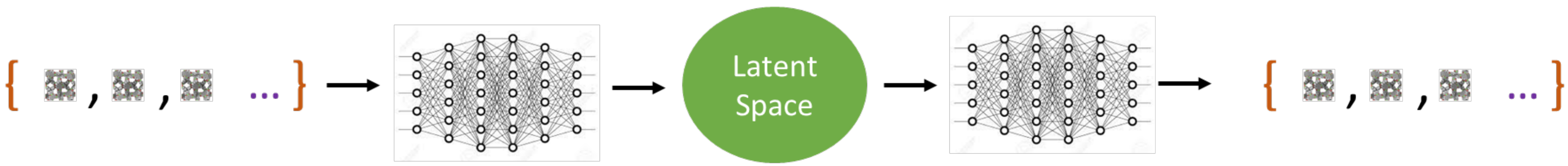
Learning to Search Framework [Deshwal et al., 2021]

- Defines a new family of search-style BO approaches
- Can work with any complex statistical model and acquisition function
- Can handle complex domain constraints to select ``valid'' structures for evaluation

Reduction to Continuous BO [Gómez-Bombarelli et al., 2018]…

- **Key Idea:** Convert discrete space into continuous space

- Train a deep generative model (VAE) using unsupervised structures

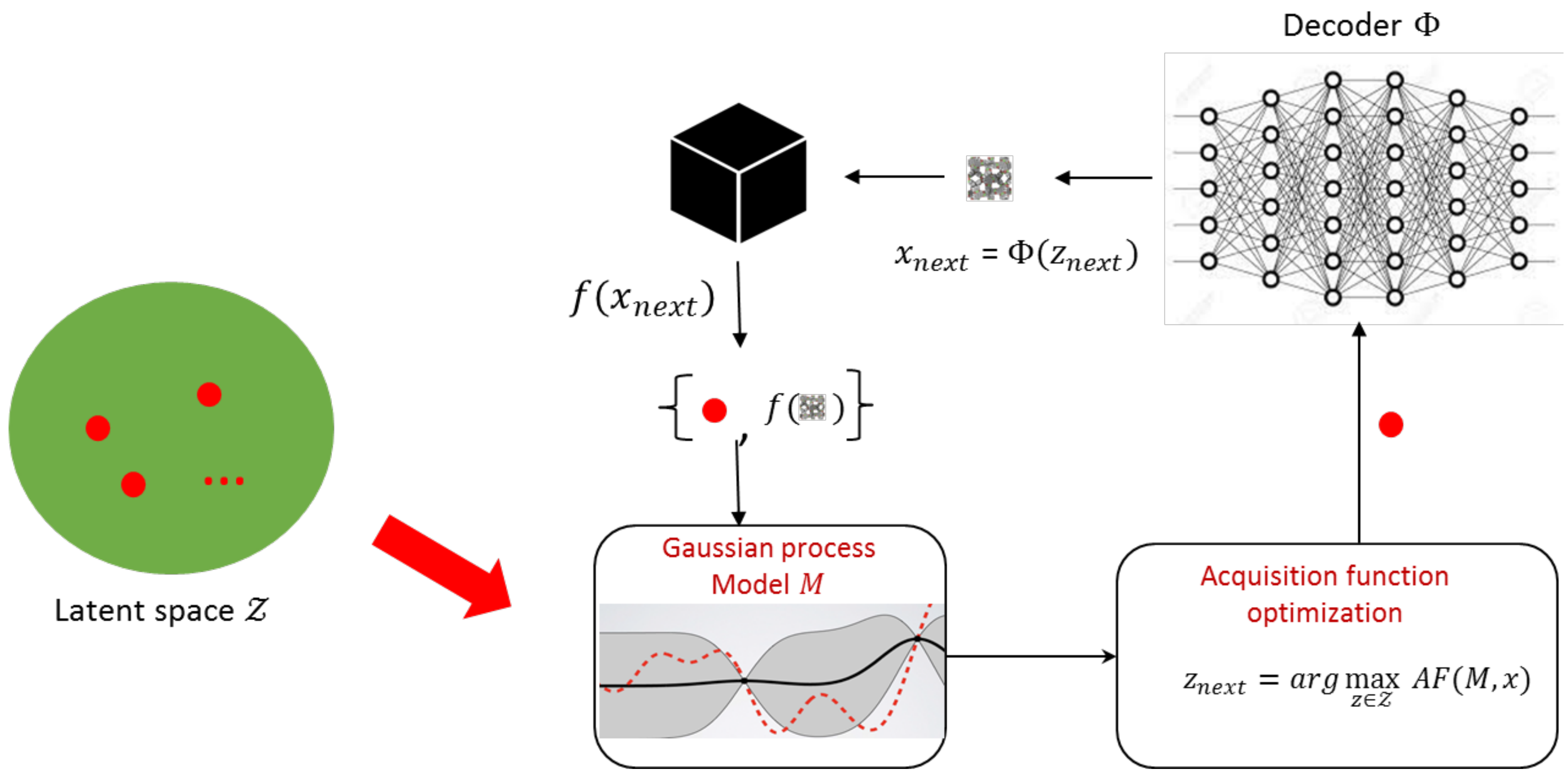


Encoder
Decoder

- Perform BO in the learned continuous latent space
  - Surrogate modeling and acquisition function optimization in latent space (vs. combinatorial space)

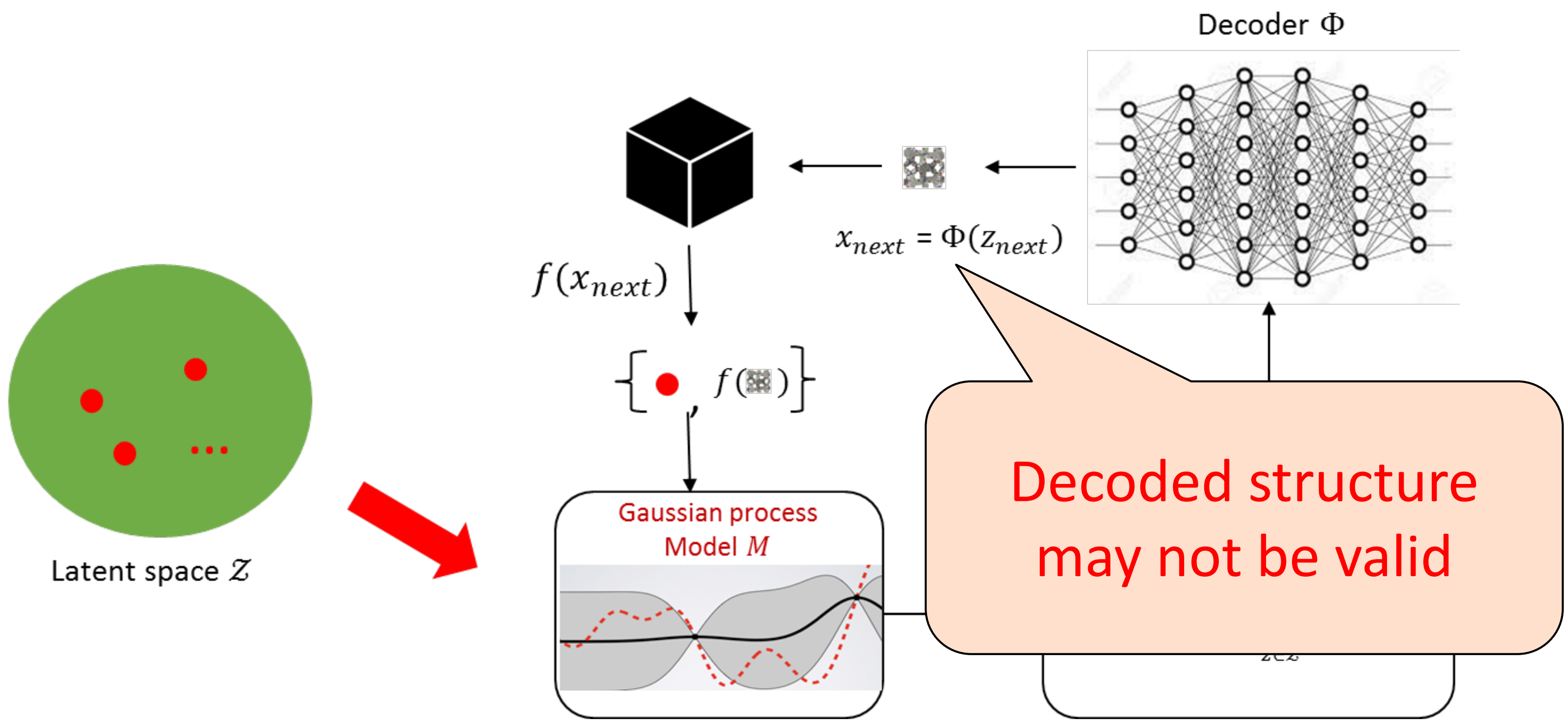
Reduction to Continuous BO [Gómez-Bombarelli et al., 2018]...

- BO in the learned latent space



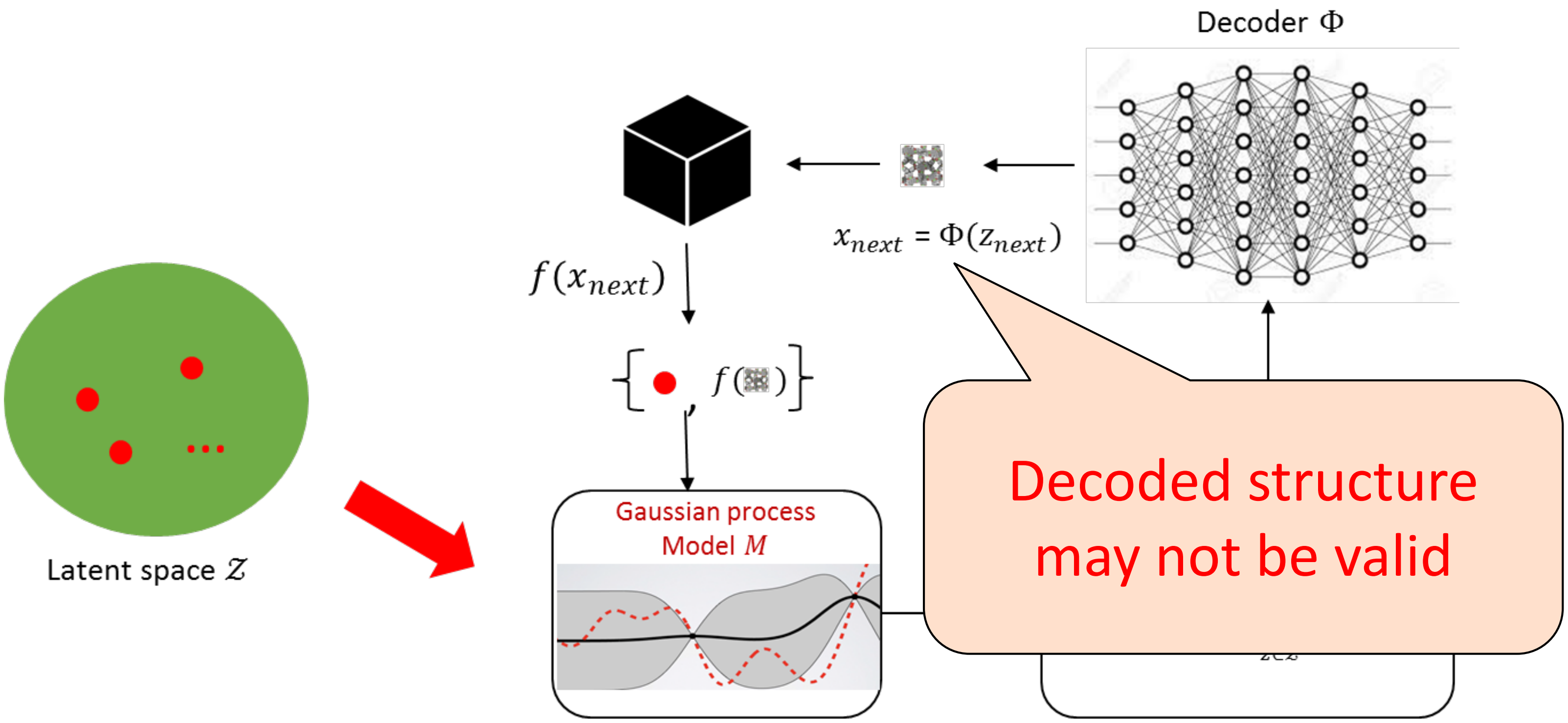
Reduction to Continuous BO [Gómez-Bombarelli et al., 2018]...

- BO in the learned latent space



Reduction to Continuous BO [Gómez-Bombarelli et al., 2018]...

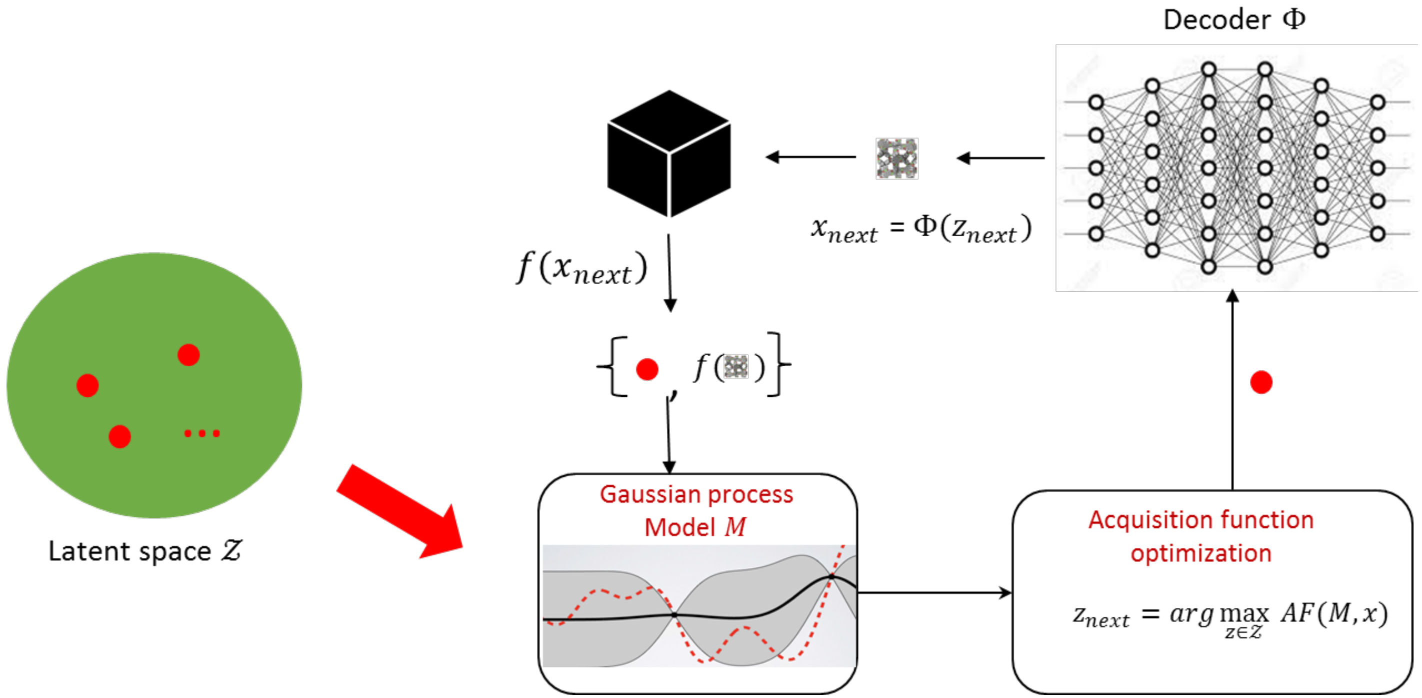
- BO in the learned latent space



- Some recent work to address this challenge
  - Griffiths R.-R. and Hernández-Lobato J. M.: Constrained Bayesian optimization for Automatic Chemical Design Using Variational Autoencoders, Chemical Science, 2019

Reduction to Continuous BO [Gómez-Bombarelli et al., 2018]...

- BO in the learned latent space



- Challenges
  - Doesn't (explicitly) incorporate information about decoded structures
  - Surrogate model may not generalize well for small data setting

Improve Latent Space via Weighted Retraining [Tripp et al., 2020]

- Periodically retrain the deep generative model
- Assign importance weights to training data proportional to their objective function value

Improve Latent Space via Weighted Retraining [Tripp et al., 2020]

- Periodically retrain the deep generative model

- Assign importa ng data proportional to their object

Computationally expensive

Improve Latent Space via Weighted Retraining [Tripp et al., 2020]

- Periodically retrain the deep generative model
- Assign importance weights to training data proportional to their objective function value

Overall approach is not effective for small-data setting

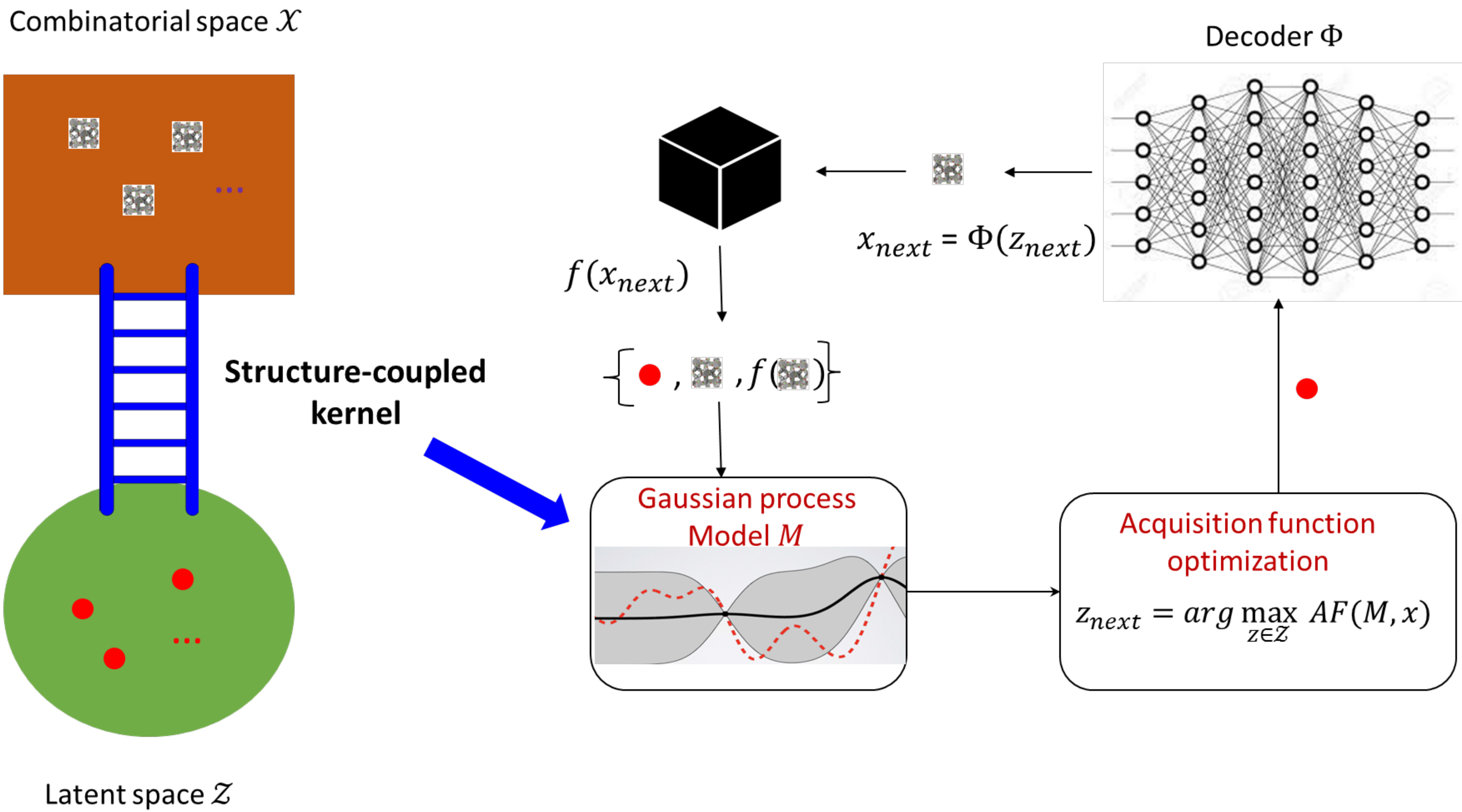
Uncertainty-guided Latent Space BO [Notin et al., 2021]

- Leverage the epistemic uncertainty of the decoder to guide the optimization process

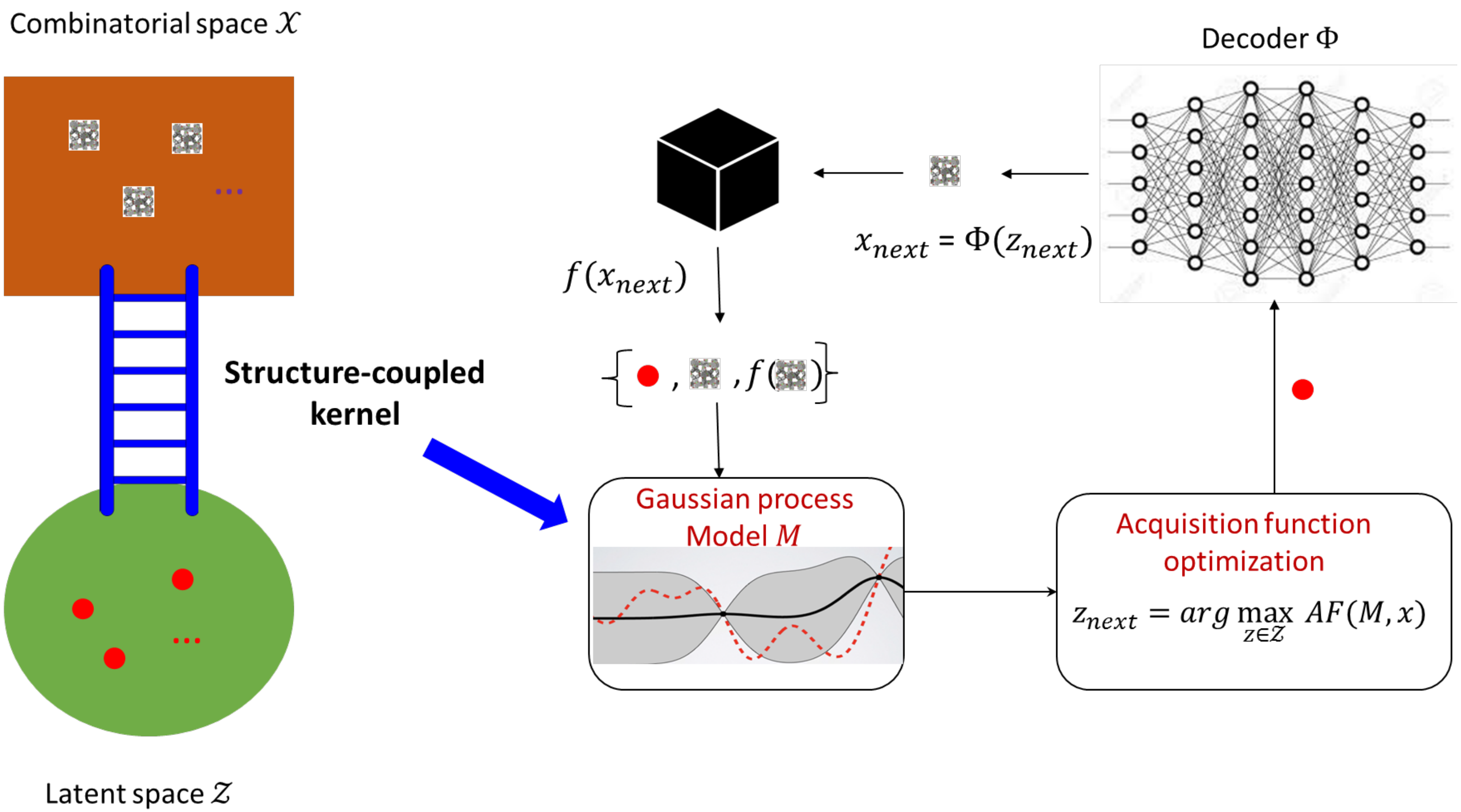
- Importance sampling-based estimator for uncertainty quantification over high-dimensional discrete structures

- No retraining of deep generative model is needed

LADDER Algorithm [Deshwal and Doppa, 2021]



LADDER Algorithm [Deshwal and Doppa, 2021]



- Key Idea: Combines the complementary strengths of deep generative models and structured kernels for better surrogate modeling

Structure-Coupled Kernel

- Structure-coupled kernel (c) combines
  - Continuous kernels over latent space $\mathcal{Z}$ (e.g., Matern)
  - Structured kernels (e.g., generic/hand-designed strings, graphs)

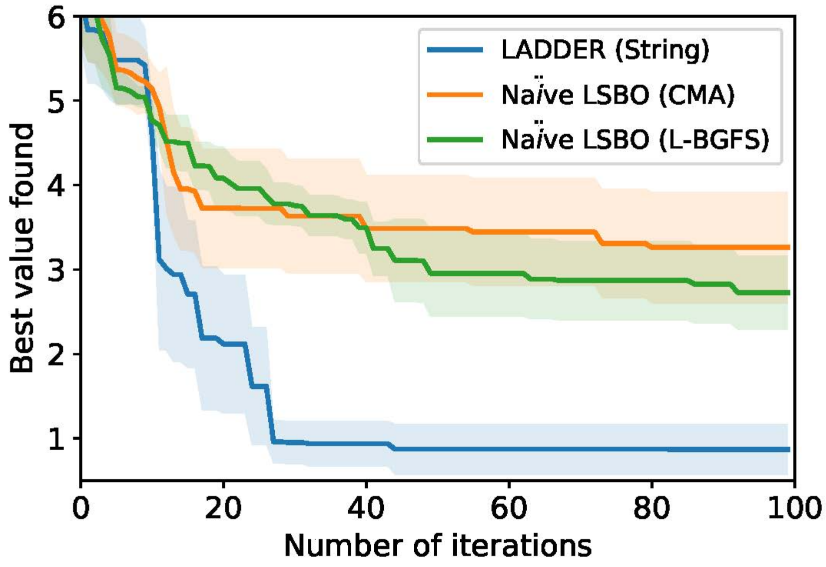
- Key Idea
  - Extrapolate <u>eigenfunctions</u> of the latent space kernel matrix **L** with basis functions from the structured kernel k

$$\boxed{c(z, z') = k_z^T K^{-1} L K^{-1} k_{z'}}$$

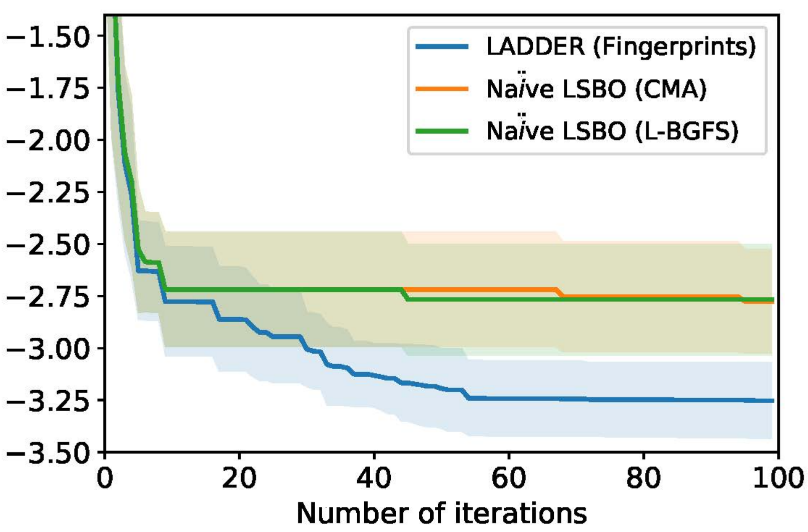
- Generalized Nystrom Extension [Ref]
  - k acts like a smooth extrapolating kernel

Latent Space BO Results #1

- LADDER outperforms latent space BO real benchmarks



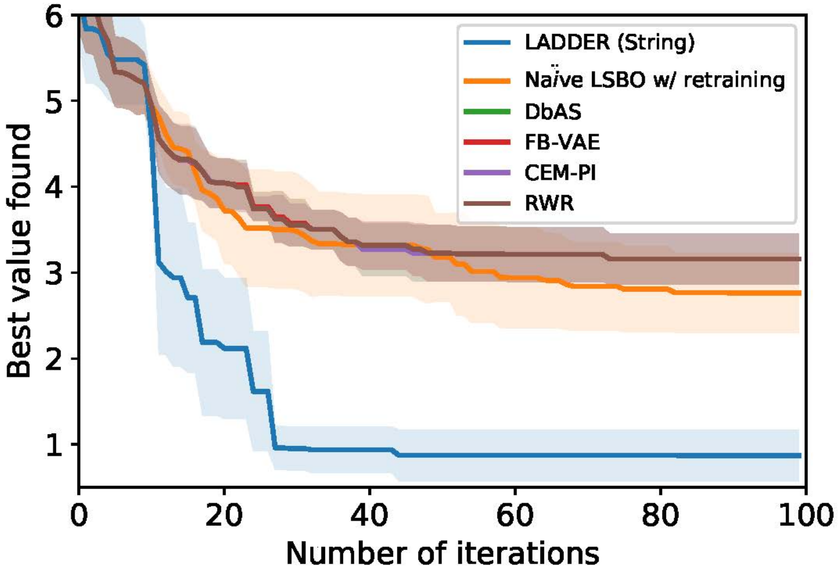
Arithmetic expression task



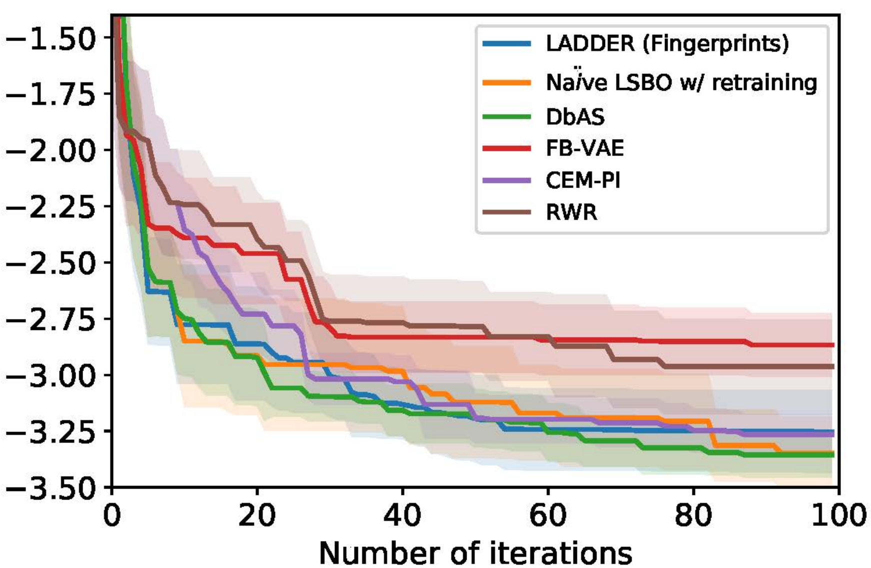
Chemical design task

Latent Space BO Results #2

- LADDER is competitive or better than state-of-the-art methods



Arithmetic expression task



Chemical design task

Code and Software

- MerCBO: https://github.com/aryandeshwal/MerCBO
- LADDER: https://github.com/aryandeshwal/LADDER
- BOPS: https://github.com/aryandeshwal/BOPS
- COMBO: https://github.com/QUVA-Lab/COMBO
- SMAC: https://github.com/automl/SMAC3

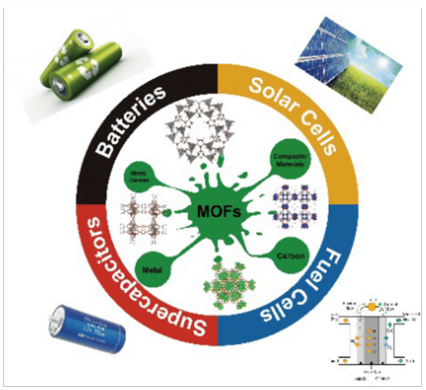
Questions ?

Bayesian Optimization over Hybrid Spaces

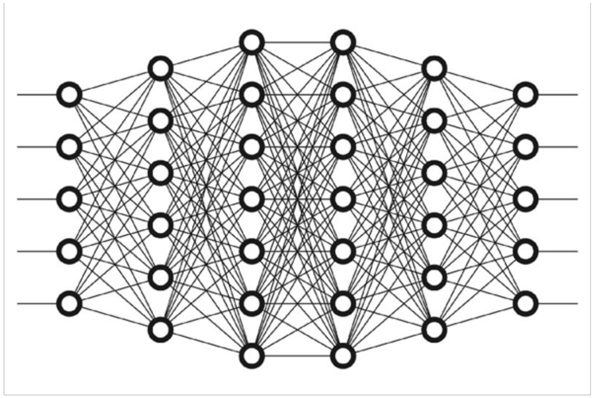
BO Over Hybrid Spaces: The Problem

- **Goal:** find optimized hybrid structures via expensive experiments
  - $x =$ mixture of x_d (discrete) and x_c (continuous) variables

Microbiome design



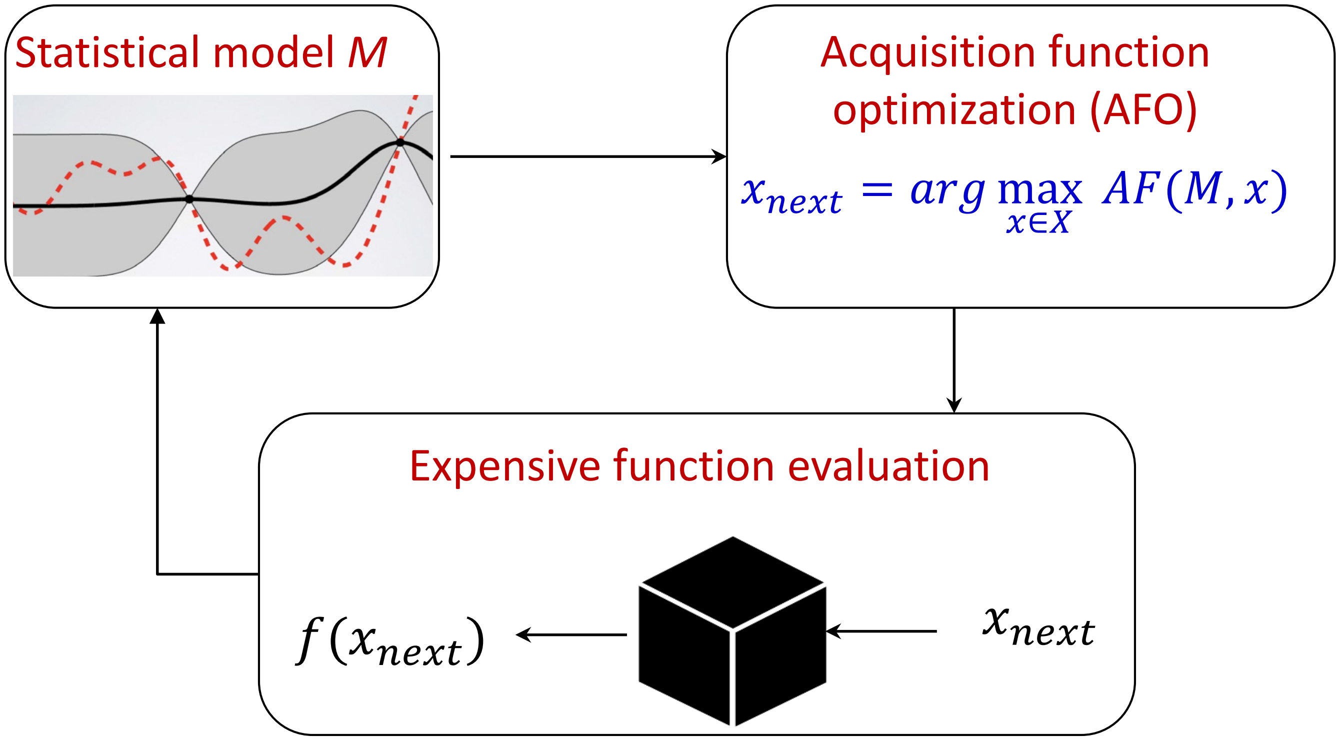
Material design



Hyper-parameter tuning / Auto ML

- Many other science, engineering, industrial applications

Hybrid BO: Technical Challenges



- Effective modeling over hybrid structures (capture complex interactions among discrete and continuous variables)
- Solving hard optimization problem over hybrid spaces to select next structure

Hybrid BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - MiVaBO [Daxberger et al., 2019]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011], HyBO [Deshwal et al., 2021] , BO-FM [Oh et al., 2021]

- Complex statistical models and tractable/accurate AFO
  - Reduction to continuous BO: GEBO [Ahn et al.,, 2022]

Hybrid BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - MiVaBO [Daxberger et al., 2019]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011], HyBO [Deshwal et al., 2021] , BO-FM [Oh et al., 2021]

- Complex statistical models and tractable/accurate AFO
  - Reduction to continuous BO: GEBO [Ahn et al.,, 2022]

MiVaBO [Daxberger et al., 2019]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of continuous (random Fourier features), discrete (BOCS representation for binary variables), and mixed (products of all pairwise combinations) features

- Thompson sampling as acquisition function

- Alternating search for acquisition function optimization
  - Step 1: Search over continuous sub-space
  - Step 2: Search over discrete sub-space using output of Step #1
  - Repeat (if needed)

MiVaBO [Daxberger et al., 2019]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of continuous (random Fourier features), discrete (BOCS representation for binary variables), and mixed (products of all pairwise combinations) features

May not be sufficient to capture desired dependencies

- Thompson

- Alternating search for acquisition function optimization
  - Step 1: Search over continuous sub-space
  - Step 2: Search over discrete sub-space using output of Step #1
  - Repeat (if needed)

MiVaBO [Daxberger et al., 2019]

- Linear surrogate model over binary structures
  - $f(x \in X) = \theta^T . \phi(x)$
  - $\phi(x)$ consists of continuous (random Fourier features), discrete (BOCS representation for binary variables), and mixed (products of all pairwise combinations) features
- Thompson sampling as acquisition function
- Alternating search for acquisition function optimization
  - Step 1: Searc… …r continuous sub-space
  - Step 2: Se… …pace using output of Step #1
  - Repeat (i…

Can potentially get stuck in local optima

Hybrid BO: Summary of Approaches

- Trade-off complexity of model and tractability of AFO

- Simple statistical models and tractable search for AFO
  - MiVaBO [Daxberger et al., 2019]

- Complex statistical models and heuristic search for AFO
  - SMAC [Hutter et al., 2011], HyBO [Deshwal et al., 2021] , BO-FM [Oh et al., 2021]

- Complex statistical models and tractable/accurate AFO
  - Reduction to continuous BO: GEBO [Ahn et al.,, 2022]

SMAC Algorithm [Hutter et al., 2010, 2011]

- Random forest as surrogate model
  - works naturally for categorical/continuous variables
  - Prediction/Uncertainty (= empirical mean/variance over trees)

- Expected improvement as acquisition function

- Hand-designed local search with restarts for AFO

SMAC Algorithm [Hutter et al., 2010, 2011]

- Random forest as surrogate model
  - works naturally for categorical variables
  - Prediction/Uncertainty (= empirical mean/variance over trees)

Uncertainty estimates can be poor

- Expected improvement as acquisition function

- Hand-designed local search with restarts for AFO

SMAC Algorithm [Hutter et al., 2010, 2011]

- Random forest as surrogate model
  - works naturally for categorical variables
  - Prediction/Uncertainty (= empirical mean/variance over trees)
- Expected improvement as acquisition function
- Hand-designed local search with restarts for AFO

Can potentially get stuck in local optima

HyBO Algorithm [Deshwal et al., 2021]

- GP surrogate model with additive diffusion kernels

- Expected improvement as acquisition function

- Alternating search for acquisition function optimization
  - Step 1: Search over continuous sub-space
  - Step 2: Search over discrete sub-space using output of Step #1
  - Repeat (if needed)

HyBO Algorithm [Deshwal et al., 2021]

- GP surrogate model with additive diffusion kernels
  - Exploits the general recipe of additive kernels [Duvenaud et al., 2011]
  - Instantiation w/ discrete & continuous diffusion kernels
  - Bayesian treatment of the hyper-parameters

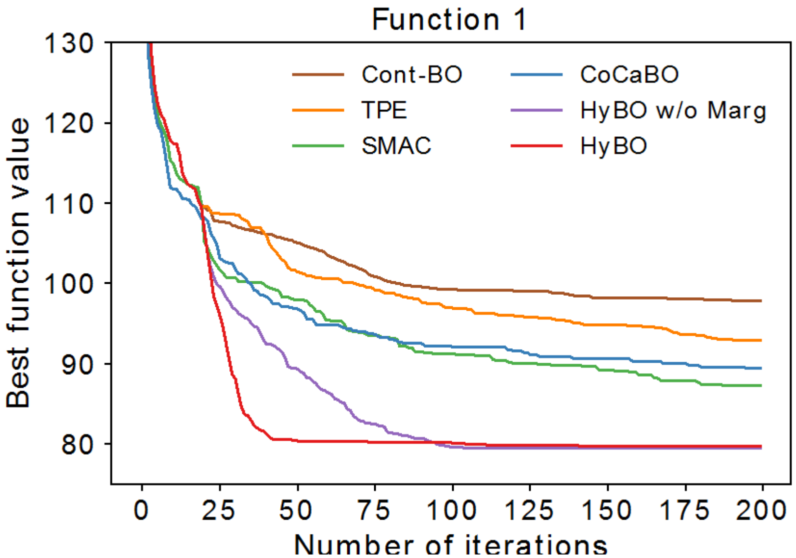
$$\mathcal{K}_{HYB} = \boxed{\sum_{p=1}^{m+n} \left(\boxed{\theta_p^2} \sum_{i_1,\cdots,i_p} \prod_{d=1}^{p} \boxed{k_{i_d}(x_{i_d}, x'_{i_d})} \right)}$$

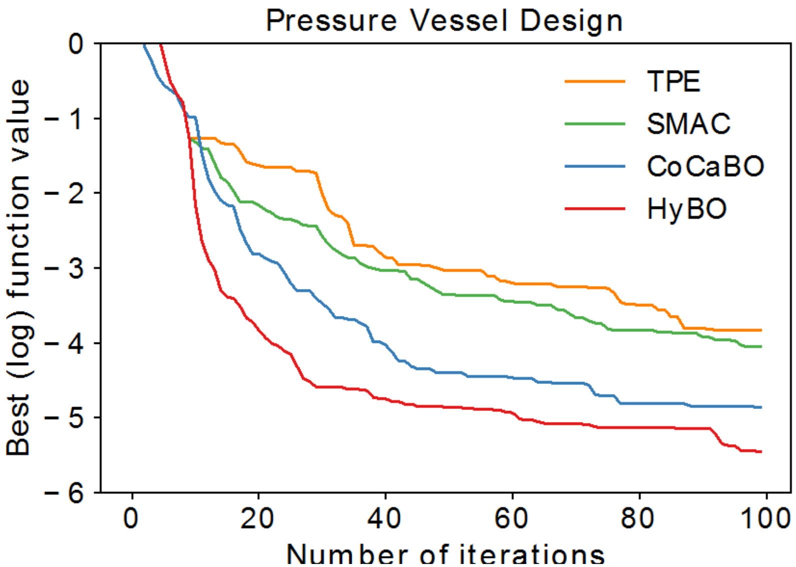
HyBO Algorithm [Deshwal et al., 2021]

- GP surrogate model with additive diffusion kernels
- Expected improvement as acquisition function
- Alternating search for acquisition function optimization
  - Step 1: Search over continuous sub-space
  - Step 2: Search over discrete sub-space using output of Step #1
  - Repeat (if needed)

Can potentially get stuck in local optima

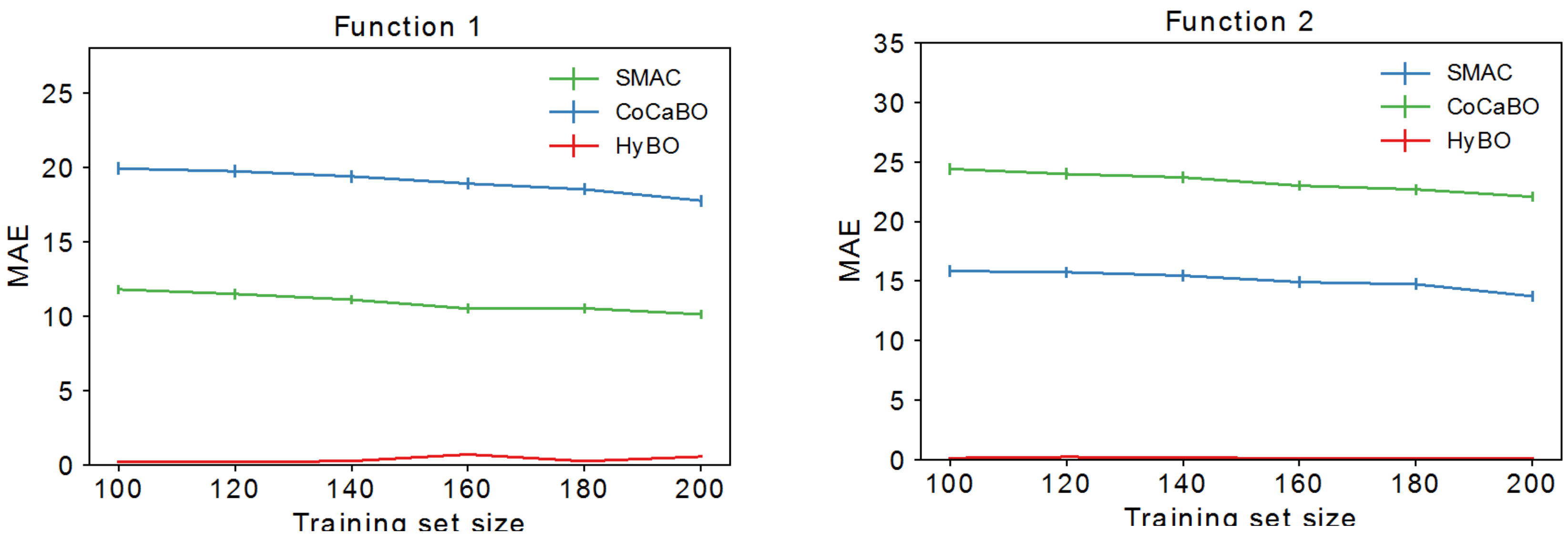
Hybrid BO: Experimental Results #1





- HyBO performs significantly better than prior methods

Hybrid BO: Experimental Results #2



- HyBO's better BO performance is due to better surrogate model

BO-FM Algorithm [Oh et al., 2021]

- GP surrogate model with frequency modulation kernels

- Expected improvement as acquisition function

- Alternating search for acquisition function optimization
  - Step 1: Search over continuous sub-space
  - Step 2: Search over discrete sub-space using output of Step #1
  - Repeat (if needed)

BO-FM Algorithm [Oh et al., 2021]

- GP surrogate model with frequency modulation kernels

- Key idea: Generalize the COMBO kernel [Oh et al., 2019] by parametrizing via a function of continuous variables

$$\text{K} = exp(-\beta L(G))$$

Remember the COMBO kernel

$$\text{K} = U^T exp(-\beta\Sigma)U$$

$$\text{K} = U^T f(\Sigma, X_c, X_{c'})U$$

- Requirement on f for K to be a positive definite kernel
  - f should be positive definite w.r.t $X_c, X_{c'}$

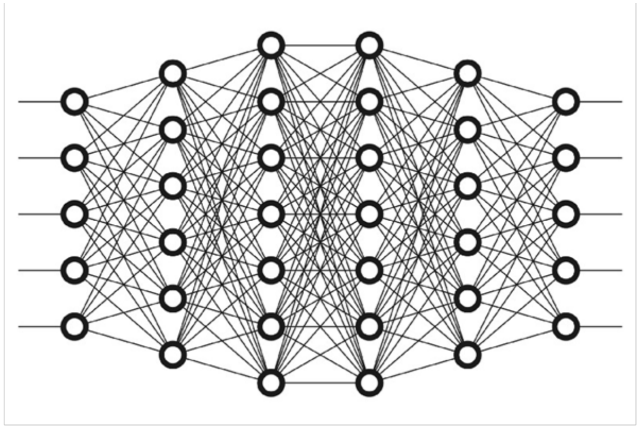
Code and Software

- HyBO: https://github.com/aryandeshwal/HyBO
- SMAC: https://github.com/automl/SMAC3

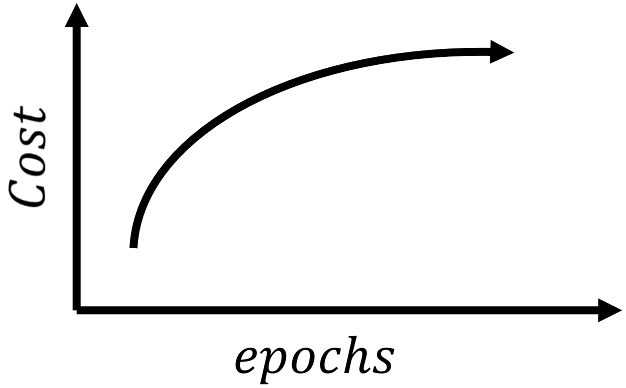
Questions ?

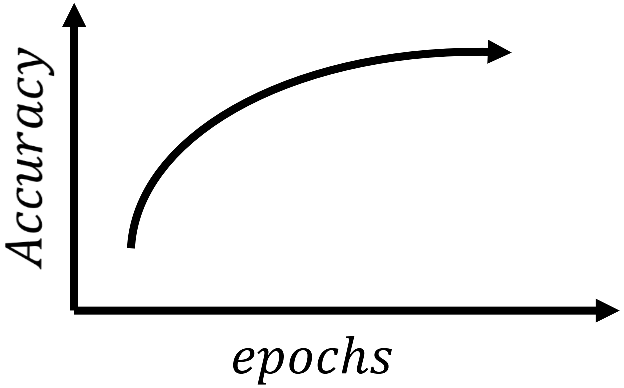
Multi-Fidelity
Bayesian Optimization

Application #1: Auto ML and Hyperparameter Tuning

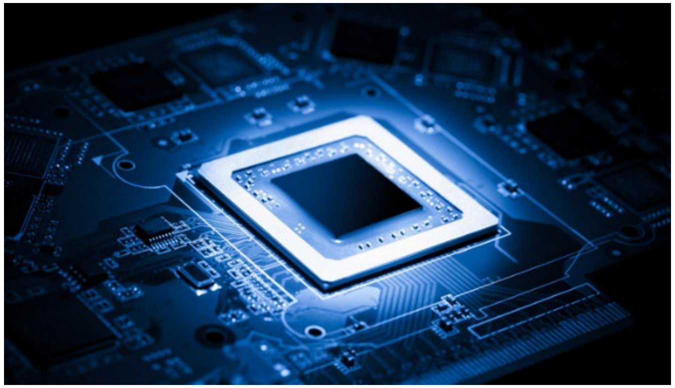


Cost vs. Accuracy trade-offs in evaluating hyperparameter configurations

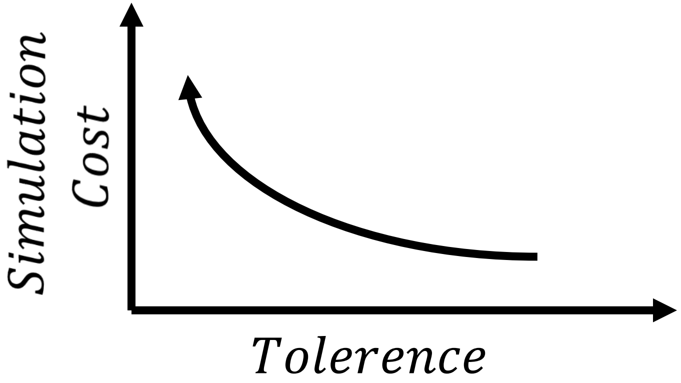


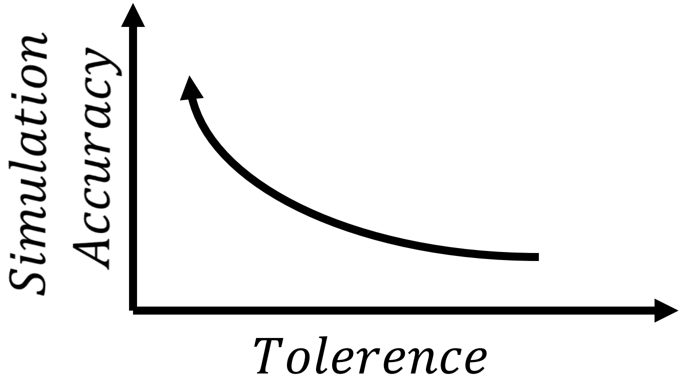


Application #2: Hardware Design via Simulations

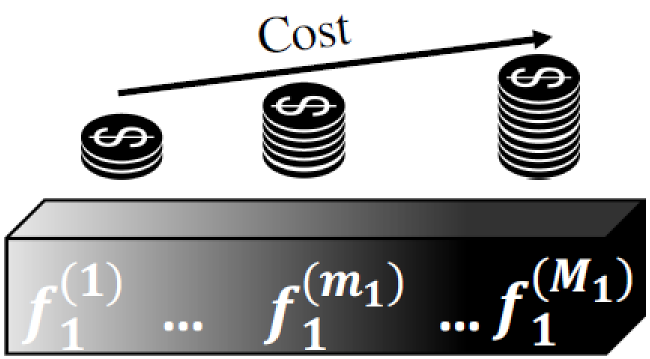


Cost vs. Accuracy trade-offs in evaluating hardware designs

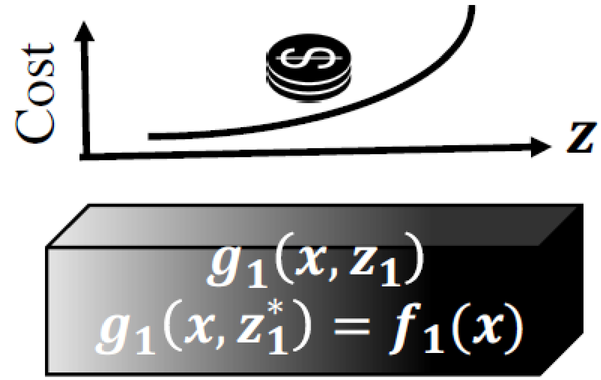




Multi-Fidelity BO: The Problem



Discrete fidelity



Continuous fidelity

- Cost vs. accuracy trade-offs for function approximations
- Continuous-fidelity is the most general case
  - Discrete-fidelity is a special case
- Goal: (approximately) optimize the highest-fidelity function by minimizing the resource cost of experiments

Multi-Fidelity BO: Key Challenges

- Intuition: use cheap (low-fidelity) experiments to gain information and prune the input space; and use costly (high-fidelity) experiments on promising candidates

- Modeling challenge: How to model multi-fidelity functions to allow information sharing?

- Reasoning challenge: How to select the input design and fidelity pair in each BO iteration?

Multi-Fidelity GPs for Modeling

- Desiderata: model relationship/information sharing between different fidelities

- Solution: multi-output GPs with vector-valued kernels

$$\boxed{k(\{x, z\}, \{x', f\}) = k(x, x')k_F(z, f)}$$

- Provides a prediction μ and uncertainty σ for each input and fidelity pair

EI Extension for Multi-Fidelity BO

- Multi-fidelity expected improvement (MF-EI)
  - Extension of EI for multi-fidelity setting
  - Applicable for discrete-fidelity setting

$$EI(x,z) = E\left[\max\left(\tau - y^f\right)\right] cov\left[y^z, y^f\right] C_f / C_z$$

- Acquisition function optimization
  - Enumerate each fidelity z and find the best x fixing z

Information-Theoretic Extensions for Multi-Fidelity BO

$$AF(x) = H(\alpha \mid D) - E_y[H(\alpha | D \cup \{x, y\})]$$
$$= \text{Information Gain}(\alpha; y)$$

- Design choices of α leads to different algorithms

- α as input location of optima x^*
  - Entropy Search (ES) / Predictive Entropy Search (PES)
  - Intuitive but requires expensive approximations

- α as output value of optima y^*
  - Max-value Entropy Search (MES) and it's variants
  - Computationally cheaper and more robust

Information-Theoretic Extensions for Multi-Fidelity BO

$$AF(x,z) = H(\alpha \mid D) - E_y[H(\alpha|D \cup \{x,z,y\})]$$
$$= \text{Information Gain per Unit Cost}(\alpha; y)$$

- Design choices of α leads to different algorithms
- α as input location of optima x^*
  - MF-Predictive Entropy Search (MF-PES)
  - Intuitive but requires expensive approximations
- α as output value of optima y^*
  - MF Max-value Entropy Search (MF-MES)
  - Computationally cheaper and more robust

Continuous-Fidelity BO: BOCA Algorithm

- Two step procedure to select input x and fidelity z separately

- **Selection of input $\boldsymbol{x}$**
  - Optimize UCB $(y^f(x) + \beta\, \sigma^f(x))$ of highest fidelity

- **Selection of fidelity $\boldsymbol{z}$**
  - Reducing fidelity space: $Z_t = \{f\} \cup \{z: \sigma^z(x_{opt}) \geq \gamma(z)\}$
  - If Z_t is not empty, select the cheapest fidelity from it
  - Otherwise, select the highest-fidelity

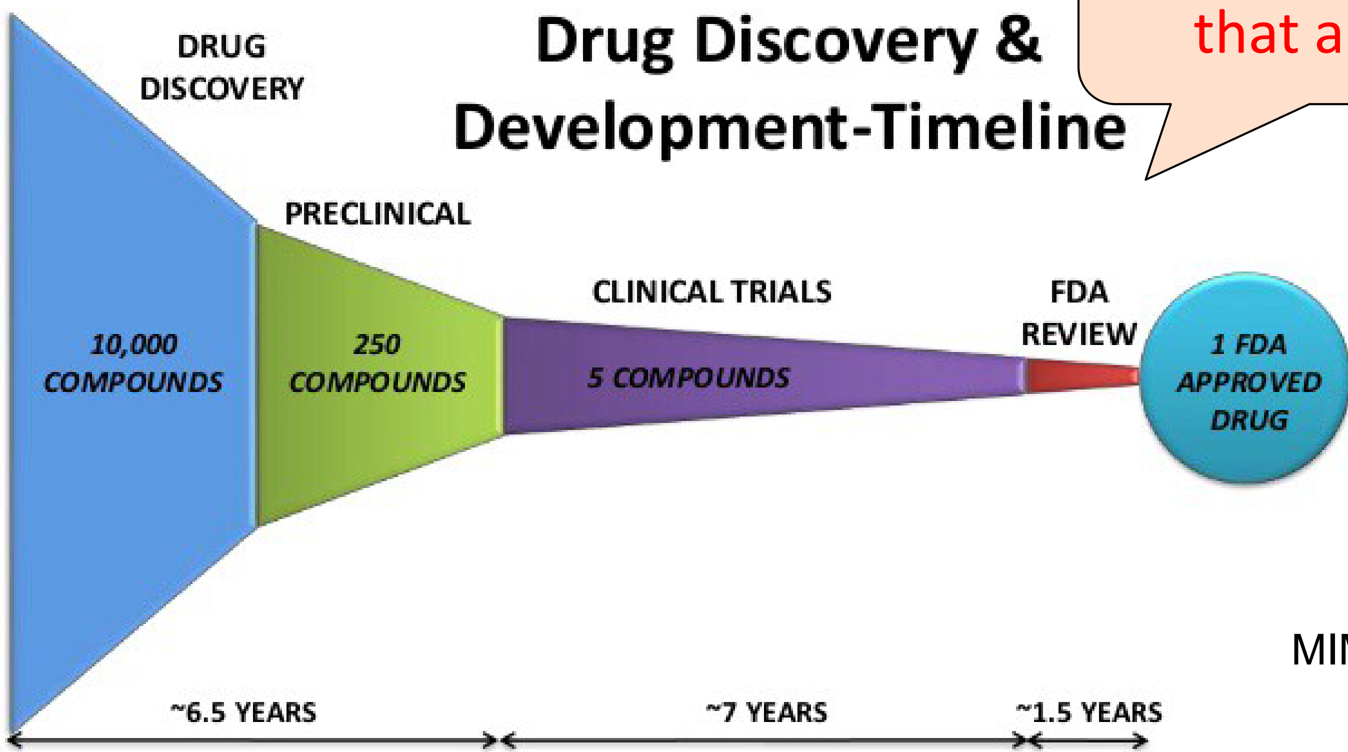
Code and Software

- Multi-fidelity modeling
  - https://mlatcl.github.io/mlphysical/lectures/05-02-multifidelity.html
- BOTorch
  - https://botorch.org/tutorials/discrete_multi_fidelity_bo

Questions ?

Bayesian Optimization with Black-Box Constraints

Application #1: Drug/Vaccine Design



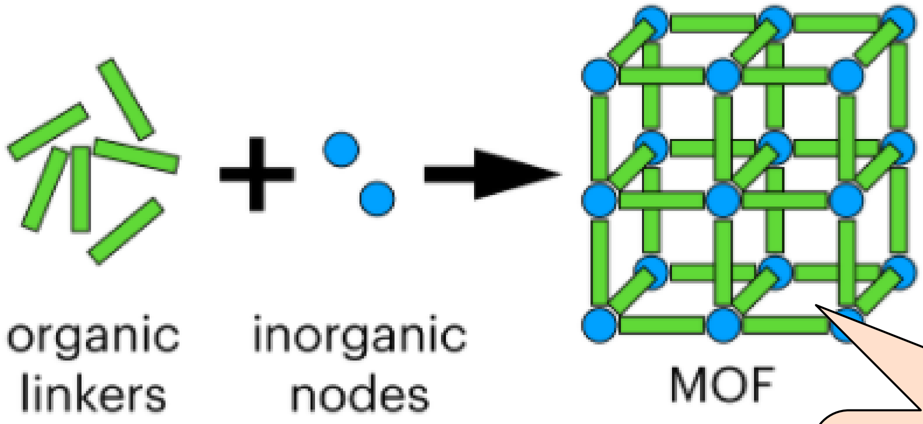
Drugs/Vaccines that are safe

Credit:

MIMA healthcare

- Accelerate the discovery of promising designs

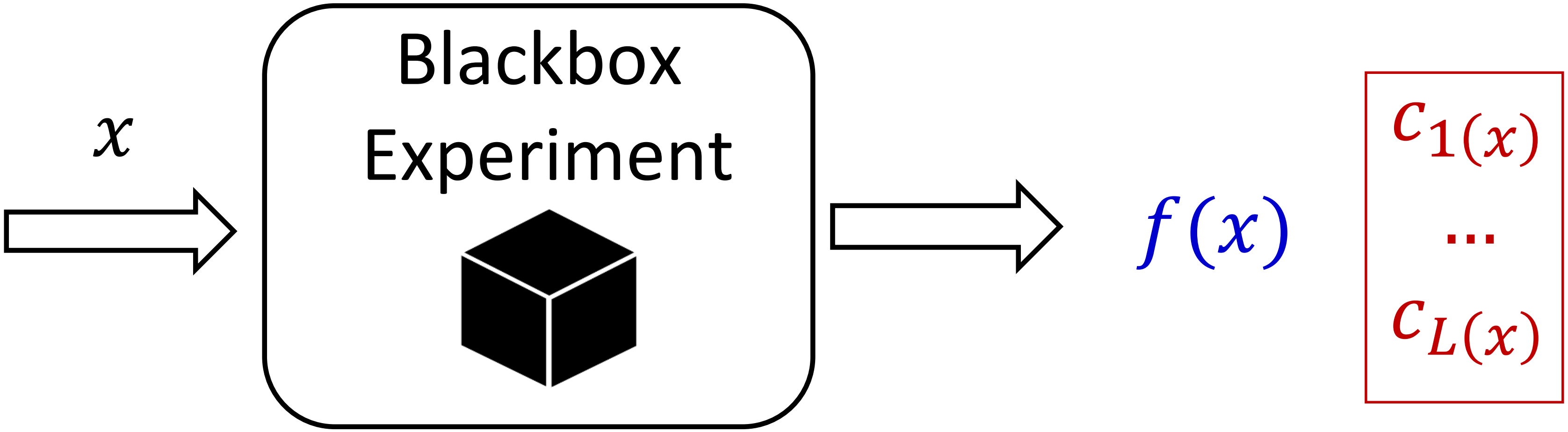
Application #2: Nanoporous Materials Design



Materials that are synthesizable

- **Sustainability applications**
  - Storing gases (e.g., hydrogen powered cars)
  - Separating gases (e.g., carbon dioxide from flue gas of coalfired power plants)
  - Detecting gases (e.g., detecting pollutants in outdoor air)

BO with Black-Box Constraints: The Problem



Objective and constraints
evaluation of design x

- Goal: find the approximate optima from the constrained input space by minimizing the total cost of experiments

BO with Black-Box Constraints: Key Challenges

- Modeling challenge: how to model black-box constraints?
  - GP models will work
- Reasoning challenge: How to select the input design guided by the learned models in each BO iteration?
  - Especially, when no valid inputs (i.e., satisfies constraints) were found from past experiments

Constrained Expected Improvement (c-EI)

- Model each constraint with an independent GP
- Suppose y^{*f} is the best function value from the valid inputs (i.e., satisfies constraints) from past experiments)
  - Assign zero improvement to all invalid inputs

$$\boxed{EI_c(x) = EI(x)\Pi_{i=1}^{k} P(\widetilde{c_i}(x) \geq 0)}$$

- When past experimental data does not contain valid inputs: y^{*f} is not defined

$$\boxed{EI_c(x) = \Pi_{i=1}^{k} P(\widetilde{c_i}(x) \geq 0)}$$

Constrained Predictive Entropy Search (PESC)

$$\alpha(x) = H(x^*|D) - \mathbb{E}_y[H(x^*|D \cup (x,y))]$$

- Approximating conditioned predictive distribution
  - First part has a closed-form solution
  - Second part approximated using expectation propagation

$$\alpha(x) = \log\left(\sigma_f^2(x)\right) + \sum_{k=1}^{K} \log\left(\sigma_{c_k}^2(x)\right) - \frac{1}{M}\left\{\sum_{m=1}^{M} \log\left(\sigma_{f_{CPD}}^2(x|x_m^*)\right) + \sum_{k=1}^{K} \log\left(\sigma_{c_{k_{CPD}}}^2(x|x_m^*)\right)\right\}$$

Constrained Max-value Entropy Search (CMES)

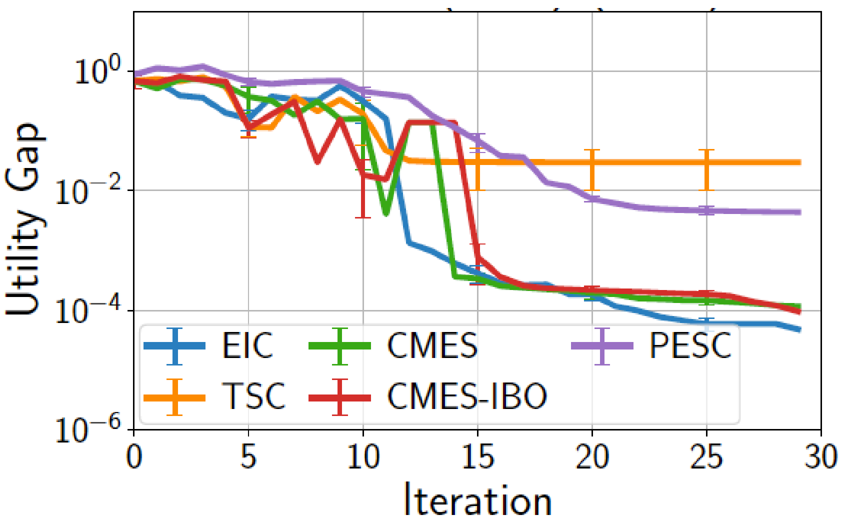
$$\alpha(x) = H(y^*|D) - \mathbb{E}_y[H(y^*|D \cup (x,y))]$$

- **Truncated multivariate distribution approximation**
  - Closed-from expression
  - Issue: can result in negative values

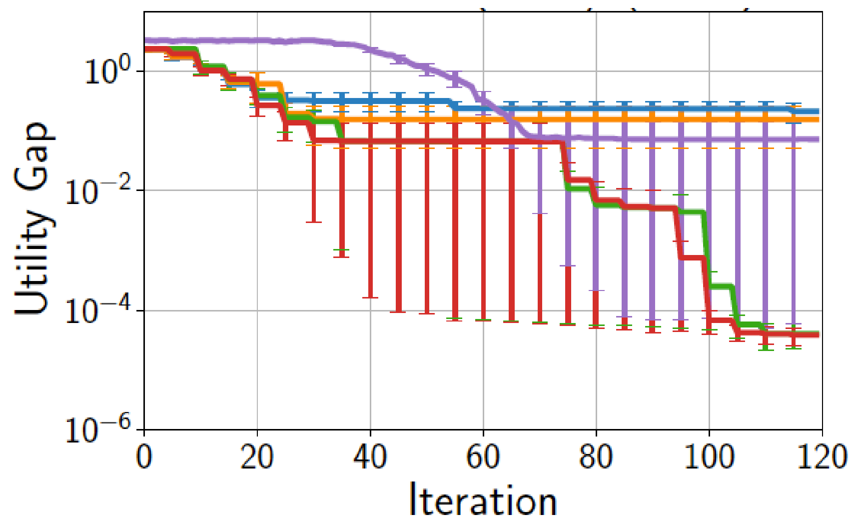
- **Lower bound approximation**
  - Closed-from expression and overcomes negative values issue
  - Maximizes the probability of selecting a valid input point when no feasible path is sampled

Constrained Max-value Entropy Search: Results

Gramacy



Hartmann6



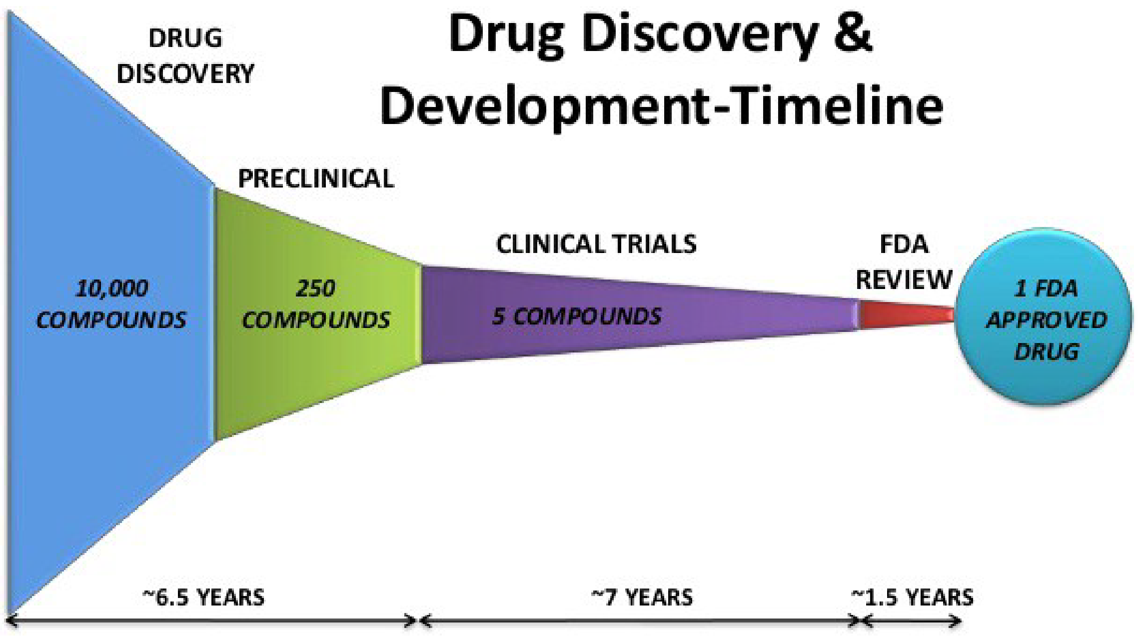
Software and Code

- PESC: github.com/HIPS/Spearmint/tree/PESC

Questions ?

Multi-Objective
Bayesian Optimization

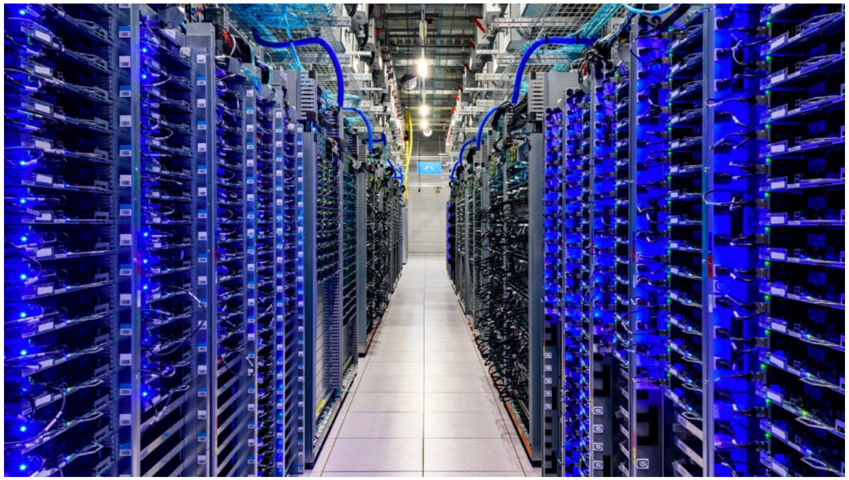
Application #1: Drug/Vaccine Design



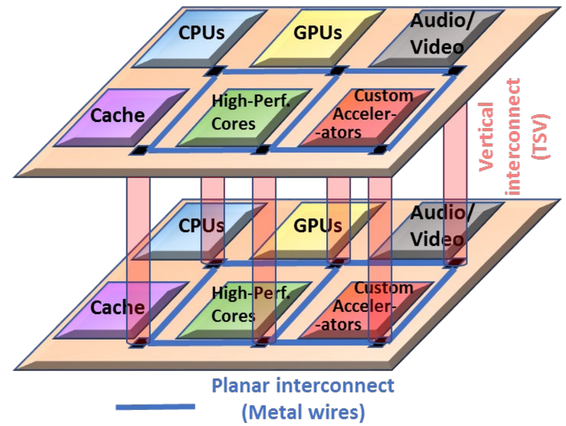
Credit: MIMA healthcare

- Accelerate the discovery of promising designs

Application #2: Hardware Design for Datacenters



High-performance and Energy-efficient manycore chips



America's Data Centers Are Wasting Huge Amounts of Energy

By 2020, data centers are projected to consume roughly 140 billion kilowatt-hours annually, costing American businesses $13 billion annually in electricity bills and emitting nearly 150 million metric tons of carbon pollution

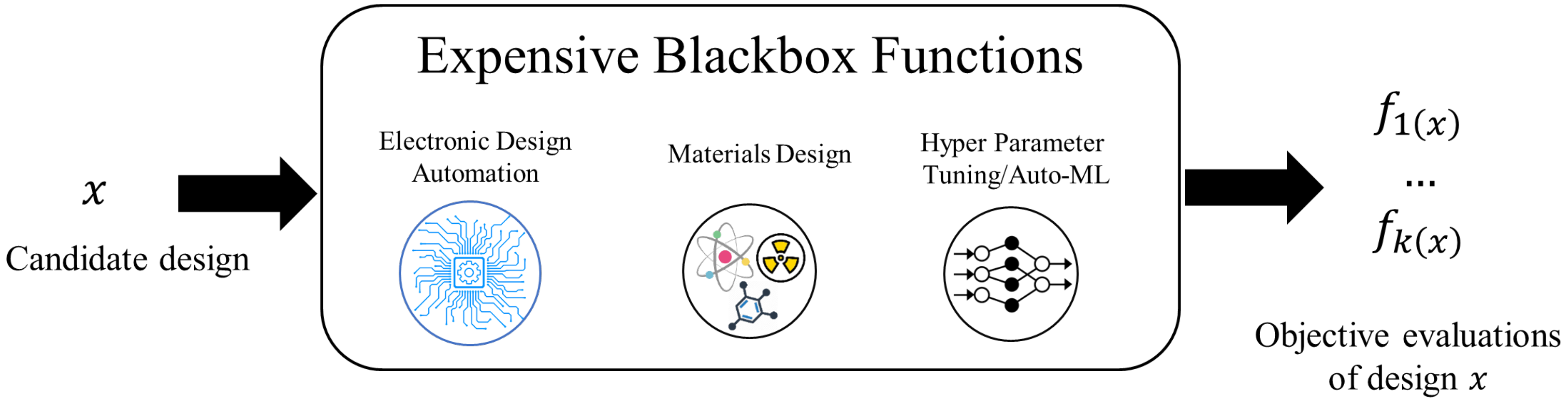
Performance

Reliability

Power

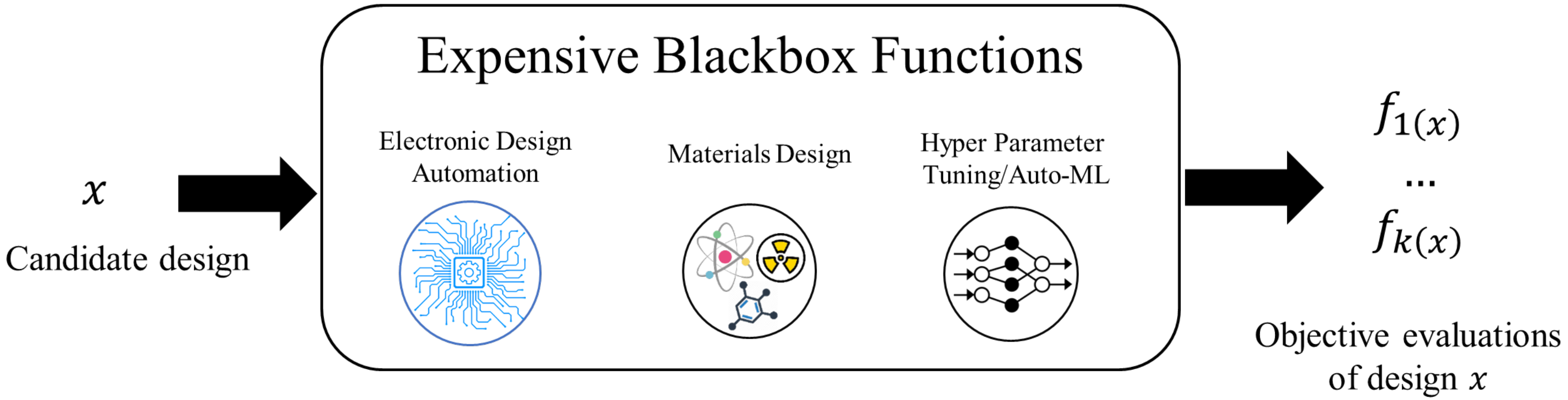
Report from Natural Resources Defense Council:.
https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IB.pdf

Multi-Objective Optimization: The Problem



- Goal: Find designs with optimal trade-offs by minimizing the total resource cost of experiments

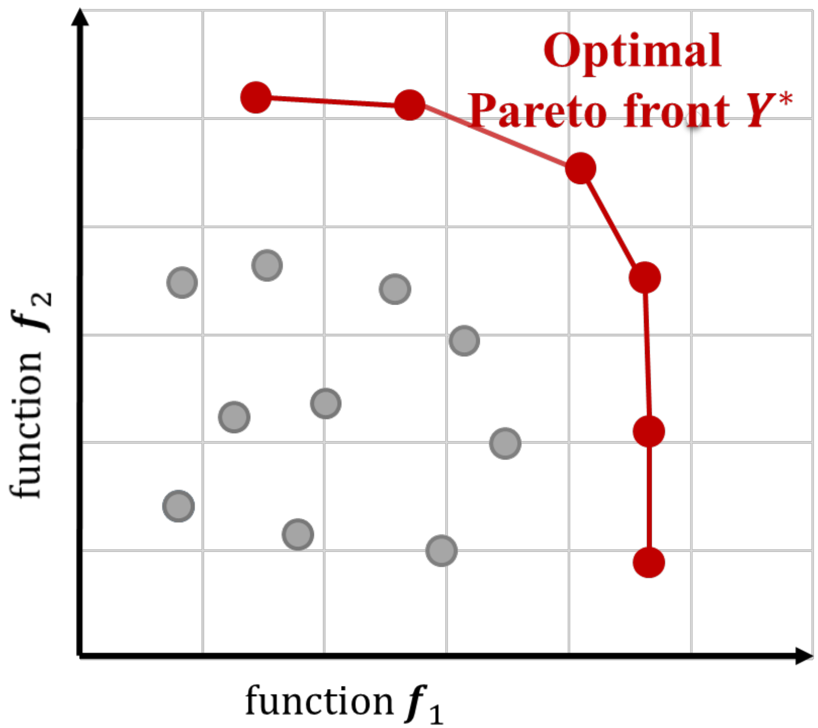
Multi-Objective Optimization: Key Challenge



- Optimize multiple conflicting objective functions

Multi-Objective Optimization: The Solution

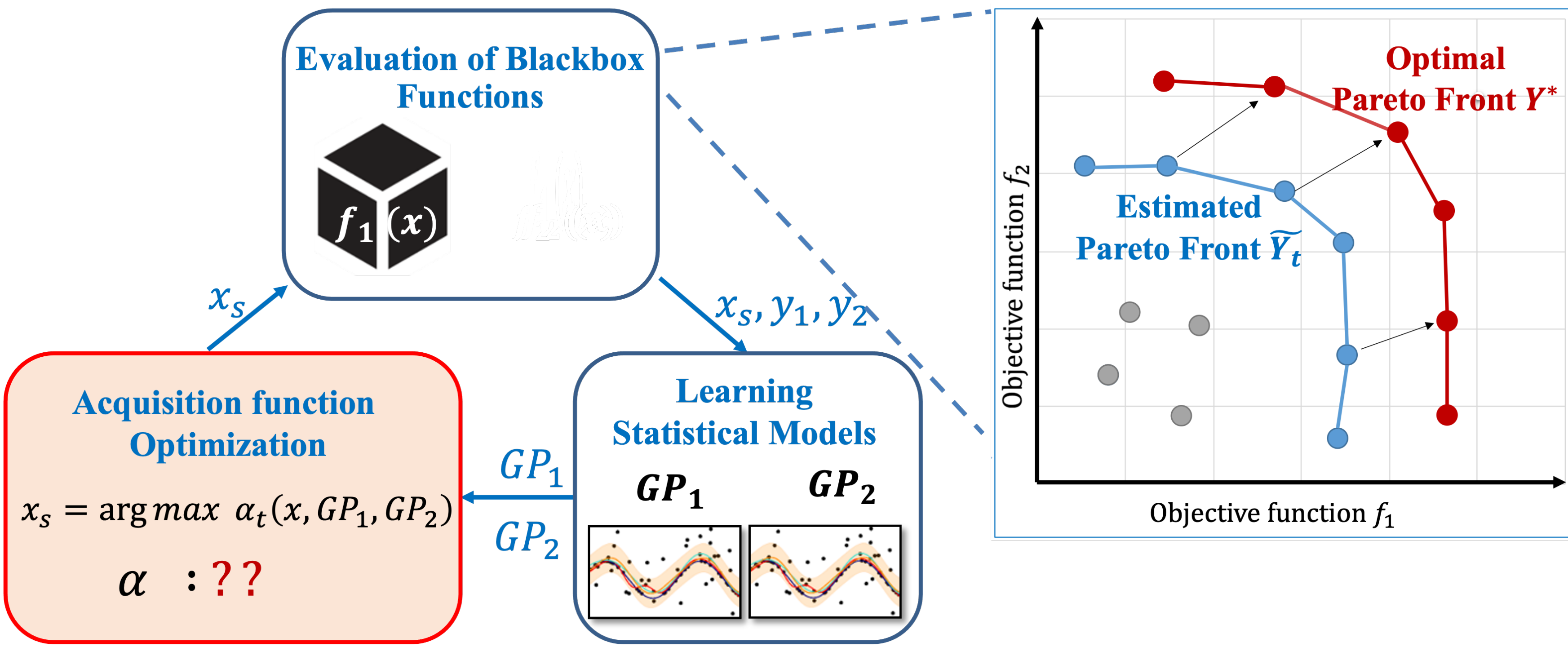
- Set of input designs with optimal trade-offs called the optimal Pareto set χ^*
- Corresponding set of function values called optimal pareto front Pareto front Y^*



- Pareto hypervolume measures the quality of a Pareto front

Single => Multi-Objective BO

- Challenge #1: Statistical modeling
  - Typically, one GP model for each objective function (tractability)
- Challenge #2: Acquisition function design
  - Capture the trade-off between multiple objectives



Multi-Objective BO: Summary of Approaches

- Reduction to single-objective via scalarization
  - ParEGO [Knowles et al., 2006] and MOBO-RS [Paria et al., 2019]
- Hypervolume improvement
  - EHI [Emmerich et al., 2008] , SUR [Picheny et al., 2015] , SMSego [Ponweiser et al., 2008] , qEHVI [Daulton et al., 2020], DGEMO [Lukovic et al. 2020]
- Wrapper methods via single-objective acquisition functions
  - USeMO [Belakaria et al., 2020]
- Information-theoretic methods
  - ϵ-PAL [Zuluaga et al., 2013] , PESMO [Hernandez-Lobato et al., 2016] , MESMO [Belakaria et al., 2019]

Multi-Objective BO: Summary of Approaches

- Reduction to single-objective via scalarization
  - ParEGO [Knowles et al., 2006] and MOBO-RS [Paria et al., 2019]
- Hypervolume improvement
  - EHI [Emmerich et al., 2008] , SUR [Picheny et al., 2015] , SMSego [Ponweiser et al., 2008] , qEHVI [Daulton et al., 2020], DGEMO [Lukovic et al. 2020]
- Wrapper methods via single-objective acquisition functions
  - USeMO [Belakaria et al., 2020]
- Information-theoretic methods
  - ϵ-PAL [Zuluaga et al., 2013] , PESMO [Hernandez-Lobato et al., 2016] , MESMO [Belakaria et al., 2019]

Reduction via Random Scalarization

- Reduce the problem to single objective optimization

- ParEGO [Knowles et al., 2006]
  - BO over scalarized objective function using EI

$$f(x) = \sum_{i=1}^{k} \lambda_i . f_i(x)$$

  - Scalar weights are sampled from a uniform distribution

- MOBO-RS [Paria et al., 2019]
  - Optimize scalarized objective function over a set of scalar weight-vectors using a prior specified by the user

Reduction via Random Scalarization

- ParEGO [Knowles et al., 2006]
  - BO over scalarized objective function using EI

$$f(x) = \sum_{i=1}^{k} \lambda_i . f_i(x)$$

  - Scalar weights are sampled from a uniform distribution
- MOBO-RS [Paria et al., 2019]
  - Optimize scalarized objective function over a set of scalar weight-vectors using a prior specified by the user

Hard to define the scalars or specify priors over scalars, which can lead to sub-optimal results

Multi-Objective BO: Summary of Approaches

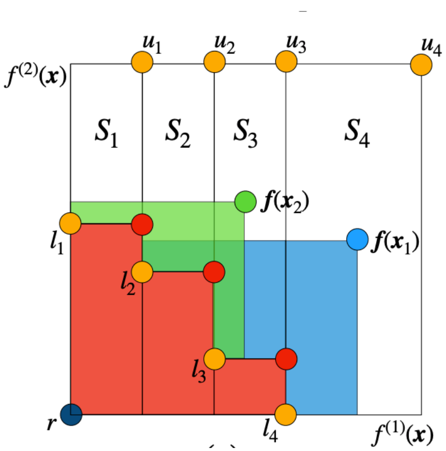
- Reduction to single-objective via scalarization
  - ParEGO [Knowles et al., 2006] and MOBO-RS [Paria et al., 2019]
- Hypervolume improvement
  - EHI [Emmerich et al., 2008] , SUR [Picheny et al., 2015] , SMSego [Ponweiser et al., 2008] , qEHVI [Daulton et al., 2020], DGEMO [Lukovic et al. 2020]
- Wrapper methods via single-objective acquisition functions
  - USeMO [Belakaria et al., 2020]
- Information-theoretic methods
  - ϵ-PAL [Zuluaga et al., 2013] , PESMO [Hernandez-Lobato et al., 2016] , MESMO [Belakaria et al., 2019]

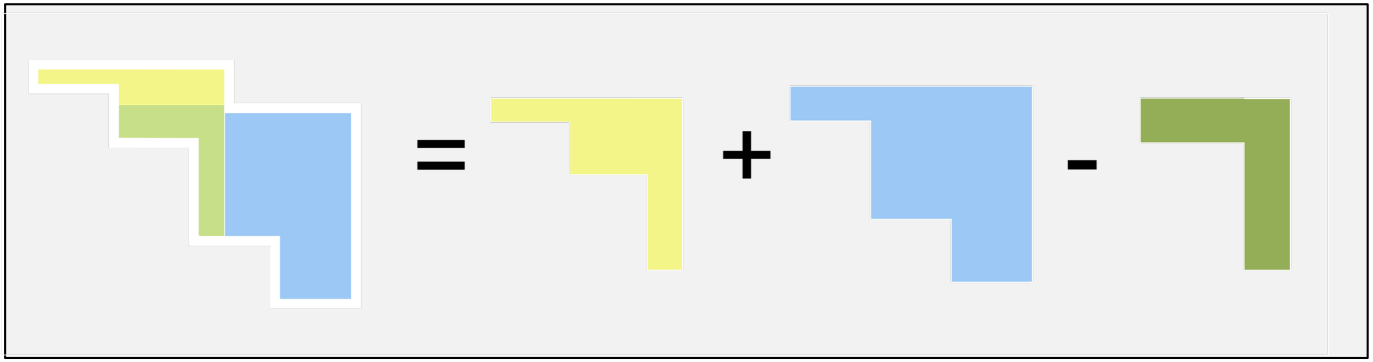
Hypervolume Improvement Approaches

- EHI: Expected improvement in PHV [Emmerich et al., 2008]
- SUR: Probability of improvement in PHV [Picheny et al., 2015]
- SMSego [Ponweiser et al., 2008]
  - Improves the scalability of PHV computation by automatically reducing the search space
- qEHVI [Daulton et al., 2020]
  - Differentiable hypervolume improvement

qEHVI Algorithm [Daulton et al., 2020]

- Parallel EHVI via the Inclusion-Exclusion Principle





- Practical since q is usually small
- The computation of all intersections be parallelized
- The formulation simplifies computation of overlapping hypervolumes

qEHVI Algorithm [Daulton et al., 2020]

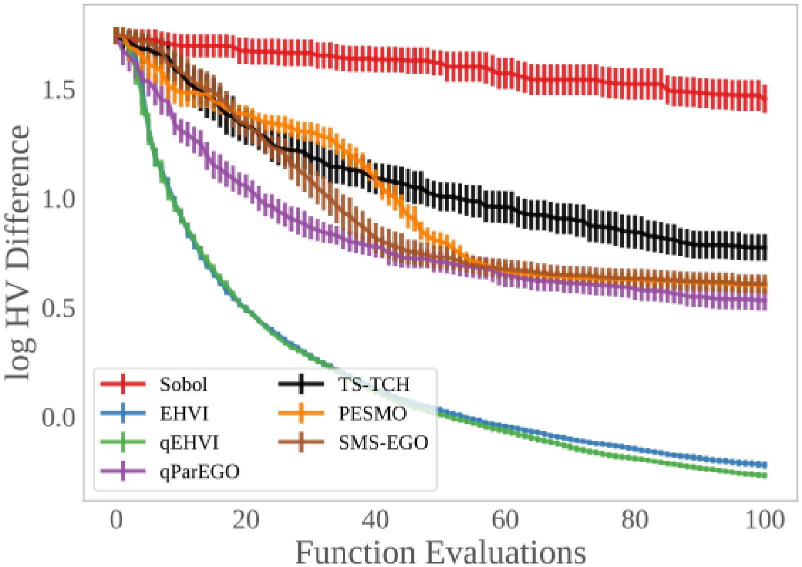
- Differentiable Hypervolume Improvement

  - Sample path gradients via the reparameterization trick
  - Unbiased gradient estimator

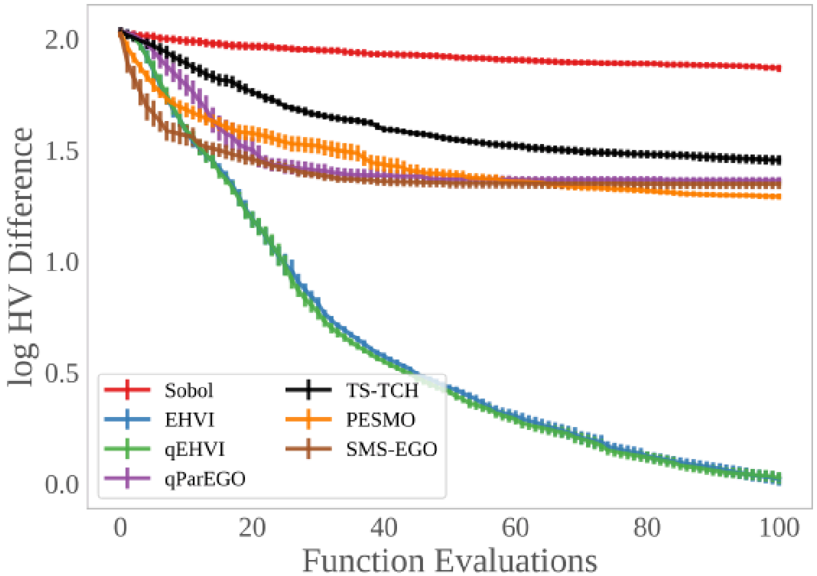
$$\mathbb{E}[\nabla_{\boldsymbol{x}} \hat{\alpha}_{qEHVI}(\boldsymbol{x})] = \nabla_{\boldsymbol{x}} \alpha_{qEHVI}(\boldsymbol{x})$$

qEHVI Algorithm [Daulton et al., 2020]

Vehicle Crash Safety



Branin-Currin



Hypervolume Improvement Approaches

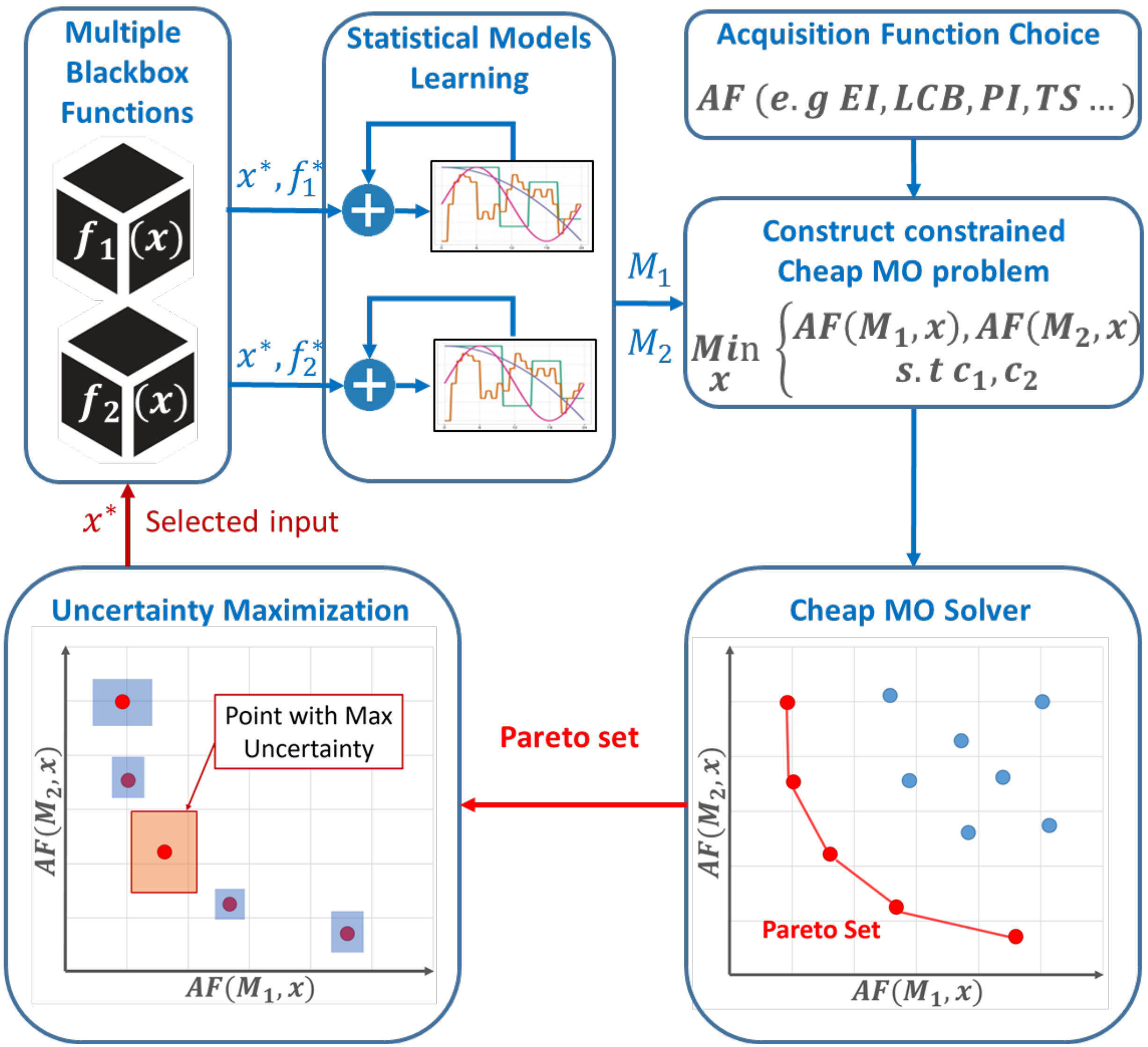
- EHI: Expected improvement in PHV [Emmerich et al., 2008]
- SUR: Probability of improvement in PHV [Picheny et al., 2015]
- SMSego [Ponweiser et al., 2008]
  - Improves the scalability of PHV computation by automatically reducing the search space
- qEHVI [Daulton et al., 2020]
  - Differentiable hypervo

Can potentially lead to more exploitation behavior resulting in sub-optimal solutions

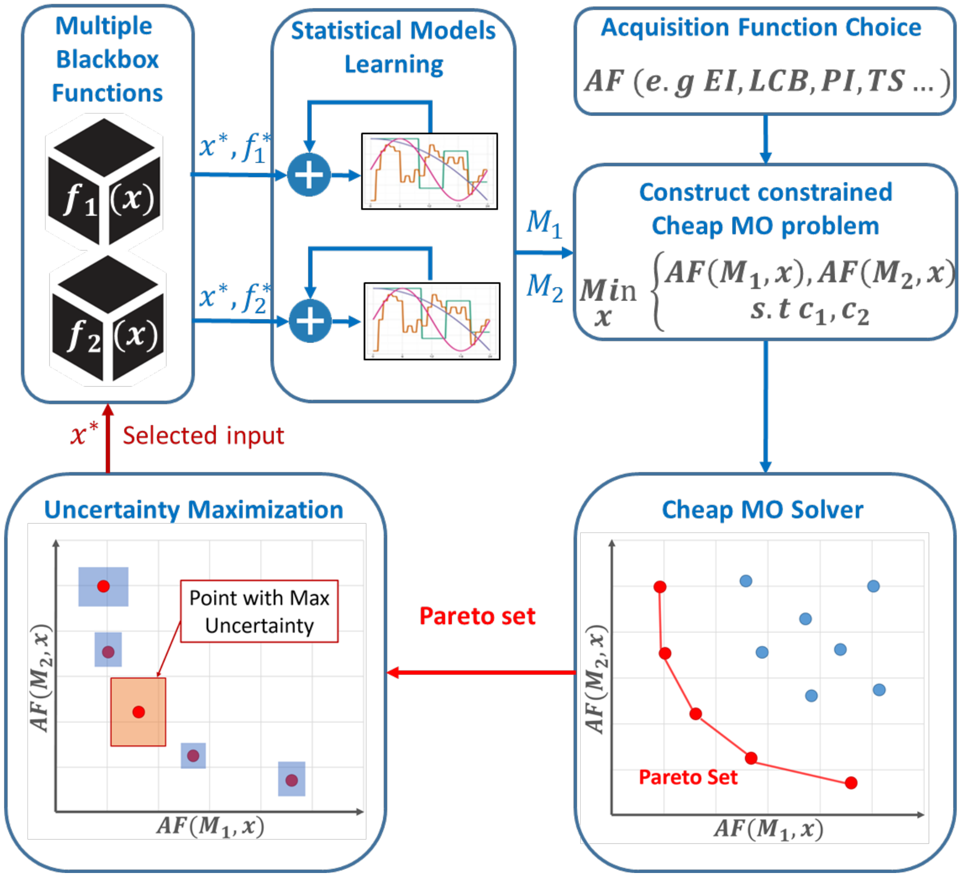
Multi-Objective BO: Summary of Approaches

- Reduction to single-objective via scalarization
  - ParEGO [Knowles et al., 2006] and MOBO-RS [Paria et al., 2019]
- Hypervolume improvement
  - EHI [Emmerich et al., 2008] , SUR [Picheny et al., 2015] , SMSego [Ponweiser et al., 2008] , qEHVI [Daulton et al., 2020]
- Wrapper methods via single-objective acquisition functions
  - USeMO [Belakaria et al., 2020]
- Information-theoretic methods
  - ϵ-PAL [Zuluaga et al., 2013] , PESMO [Hernandez-Lobato et al., 2016] , MESMO [Belakaria et al., 2019]

USeMO Framework [Belakaria et al., 2020]

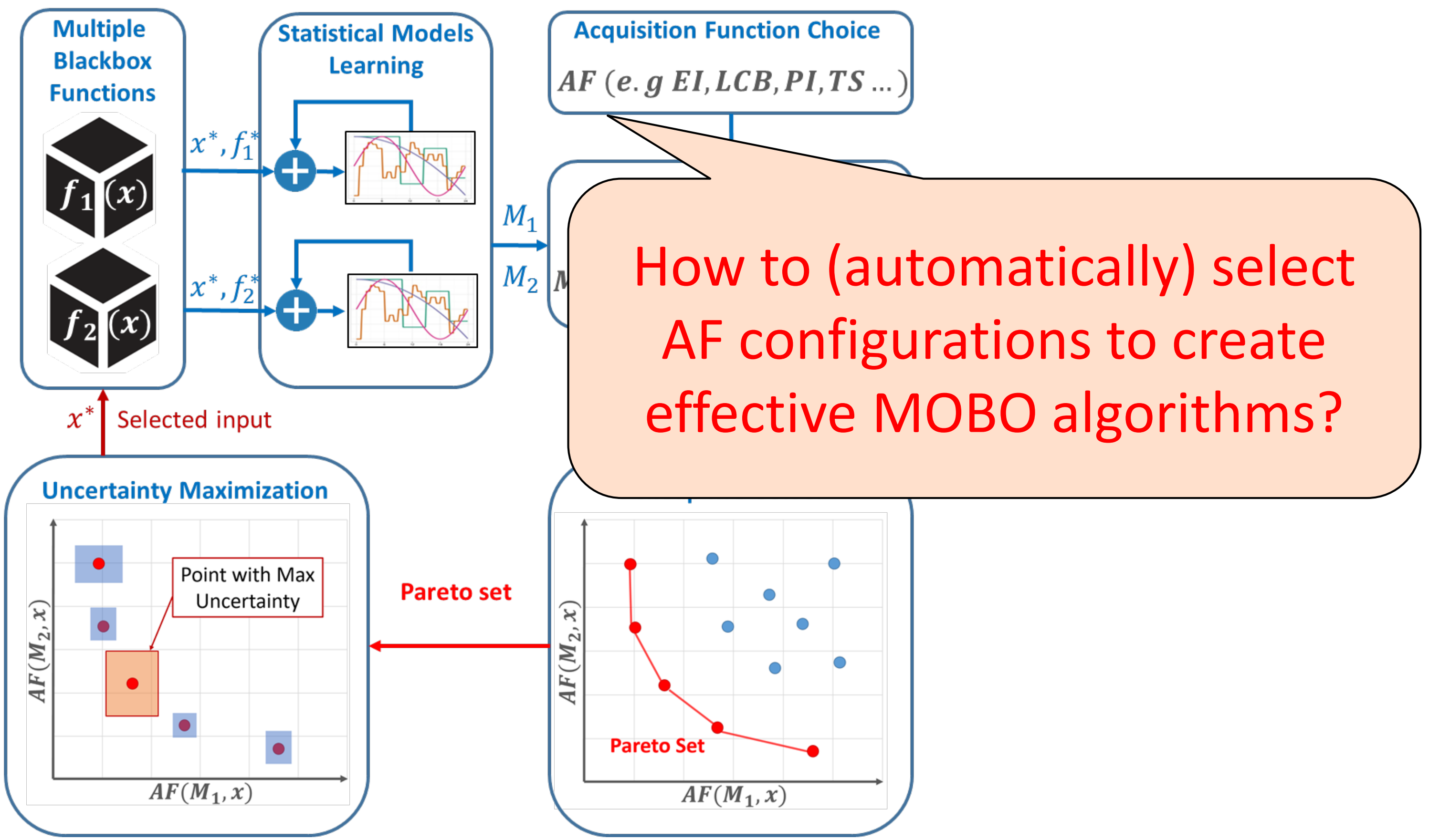


USeMO Framework [Belakaria et al., 2020]



- Allows us to leverage acquisition functions from single-objective BO to solve multi-objective BO problems

USeMO Framework [Belakaria et al., 2020]



- Allows us to leverage acquisition functions from single-objective BO to solve multi-objective BO problems

Multi-Objective BO: Summary of Approaches

- Reduction to single-objective via scalarization
  - ParEGO [Knowles et al., 2006] and MOBO-RS [Paria et al., 2019]
- Hypervolume improvement
  - EHI [Emmerich et al., 2008] , SUR [Picheny et al., 2015] , SMSego [Ponweiser et al., 2008] , qEHVI [Daulton et al., 2020]
- Wrapper methods via single-objective acquisition functions
  - USeMO [Belakaria et al., 2020]
- Information-theoretic methods
  - ϵ-PAL [Zuluaga et al., 2013] , PESMO [Hernandez-Lobato et al., 2016] , MESMO [Belakaria et al., 2019]

ϵ-PAL Algorithm [Zuluaga et al., 2013]

- Classifies candidate inputs into three categories using the learned GP models
  - Pareto-optimal
  - Not Pareto-optimal
  - Uncertain

- In each iteration, selects the candidate input for evaluation to minimize the size of uncertain set

- Accuracy of pruning depends critically on ϵ value

ϵ-PAL Algorithm [Zuluaga et al., 2013]

- Classifies candidate inputs into three categories using the learned GP models
  - Pareto-optimal
  - Not Pareto-optimal
  - Uncertain

Limited applicability as it works only for discrete set of candidate inputs

- In each iteration, selects the candidate input for evaluation to minimize the size of uncertain set

- Accuracy of pruning depends critically on ϵ value

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

- Reminder: Set of input designs with optimal trade-offs is called the optimal Pareto set χ^*

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

$$
\begin{aligned}
\alpha(x) &= I(\{x, y\}, \boxed{\chi^*} | D) \\
&= H(\chi^* | D) - \mathbb{E}_y[H(\chi^* | D \cup \{x, y\})] \\
&= H(y | D, x) - \mathbb{E}_{\chi^*}[H(y | D, x, \chi^*)]
\end{aligned}
$$

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

$$\alpha(x) = I(\{x,y\}, \boxed{\chi^*} | D)$$
$$= H(\chi^* | D) - \mathbb{E}_y[H(\chi^* | D \cup \{x,y\})]$$
$$= H(y | D, x) - \mathbb{E}_{\chi^*}[H(y | D, x, \chi^*)]$$

Equivalent to expected reduction in entropy over the pareto set χ^*

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

$$\alpha(x) = I(\{x,y\}, \boxed{\chi^*} | D)$$
$$= H(\chi^* | D) - \mathbb{E}_y[H(\chi^* | D \cup \{x,y\})]$$
$$= H(y | D, x) - \mathbb{E}_{\chi^*}[H(y | D, x, \chi^*)]$$

Due to symmetric property of information gain

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

$$\alpha(x) = I(\{x, y\}, \boxed{\chi^*} | D)$$
$$= H(\chi^* | D) - \mathbb{E}_y[H(\chi^* | D \cup \{x, y\})]$$
$$= H(y | D, x) - \mathbb{E}_{\chi^*}[H(y | D, x, \chi^*)]$$

Entropy of factorizable Gaussian distribution

PESMO Algorithm [Hernandez-Lobato et al., 2016]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto set χ^*

input dimension d

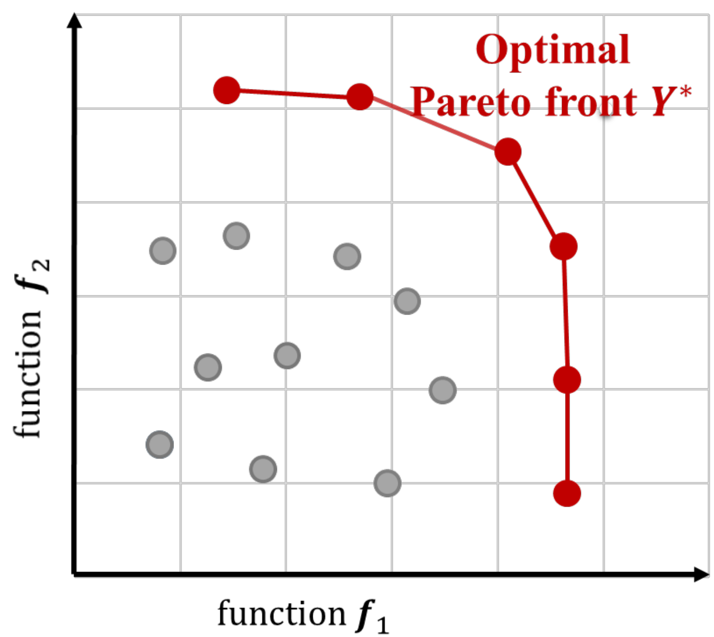
$$
\begin{aligned}
\alpha(x) &= I(\{x, y\}, \boxed{\chi^*} | D) \\
&= H(\chi^* | D) - \mathbb{E}_y[H(\chi^* | D \cup \{x, y\})] \\
&= H(y | D, x) - \boxed{\mathbb{E}_{\chi^*}[H(y | D, x, \chi^*)]}
\end{aligned}
$$

Requires computationally expensive approximation using expectation propagation

MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

- Reminder: Set of function values corresponding to the optimal Pareto set χ^* is called the optimal Pareto front Y^*



MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

$$\begin{aligned}\alpha(x) &= I(\{x, y\}, \boxed{Y^*} | D) \\ &= H(Y^* | D) - \mathbb{E}_y[H(Y^* | D \cup \{x, y\})] \\ &= H(y | D, x) - \mathbb{E}_{Y^*}[H(y | D, x, Y^*)]\end{aligned}$$

MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

$$
\begin{aligned}
\alpha(x) &= I(\{x, y\}, \boxed{Y^*} | D) \\
&= H(Y^* | D) - \mathbb{E}_y[H(Y^* | D \cup \{x, y\})] \\
&= H(y | D, x) - \mathbb{E}_{Y^*}[H(y | D, x, Y^*)]
\end{aligned}
$$

Equivalent to expected reduction in entropy over the pareto front Y^*

MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

$$\alpha(x) = I(\{x, y\}, \boxed{Y^*} | D)$$
$$= H(Y^* | D) - \mathbb{E}_y[H(Y^* | D \cup \{x, y\})]$$
$$= H(y | D, x) - \mathbb{E}_{Y^*}[H(y | D, x, Y^*)]$$

Due to symmetric property of information gain

MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

$$\alpha(x) = I(\{x, y\}, \boxed{Y^*} | D)$$
$$= H(Y^* | D) - \mathbb{E}_y[H(Y^* | D \cup \{x, y\})]$$
$$= H(y | D, x) - \mathbb{E}_{Y^*}[H(y | D, x, Y^*)]$$

Entropy of factorizable Gaussian distribution

MESMO Algorithm [Belakaria et al., 2019]

- Key Idea: select the input that maximizes the information gain about the optimal Pareto front Y^*

Output dimension $k \ll d$

$$\alpha(x) = I(\{x, y\}, \boxed{Y^*} | D)$$
$$= H(Y^* | D) - \mathbb{E}_y[H(Y^* | D \cup \{x, y\})]$$
$$= H(y | D, x) - \boxed{\mathbb{E}_{Y^*}[H(y | D, x, Y^*)]}$$

Closed form using properties of entropy and truncated Gaussian distribution

MESMO Algorithm [Belakaria et al., 2019]

$$\alpha(x) = H(y|D,x) \; - \mathbb{E}_{Y^*}[H(y\,|D,x,Y^*)]$$

- The first term is the entropy of a factorizable k-dimensional Gaussian distribution $P(y\,|D,x)$

$$H(y|D,x) = \frac{K(1+\ln(2\pi))}{2} + \sum_{j=1}^{k} \ln(\sigma_j(x))$$

MESMO Algorithm [Belakaria et al., 2019]

$$\alpha(x) = H(y|D,x) - \mathbb{E}_{Y^*}[H(y\,|D,x,Y^*)]$$

- We can approximately compute the second term via Monte-Carlo sampling (S is the number of samples)

$$\mathbb{E}_{Y^*}[H(y\,|D,x,Y^*)] \approx \frac{1}{S}\sum_{s=1}^{S} H(y\,|D,x,Y_s^*)$$

MESMO Algorithm [Belakaria et al., 2019]

- Approximate computation via Monte-Carlo sampling

$$\mathbb{E}_{Y^*}[H(y\,|D,x,Y^*)] \approx \frac{1}{S}\sum_{s=1}^{S} H(y\,|D,x,Y_s^*)$$

- **Two key steps**
  - How to compute Pareto front samples Y_s^* ?
  - How to compute the entropy with respect to a given Pareto front sample Y_s^*?

MESMO Algorithm [Belakaria et al., 2019]

- Approximate computation via Monte-Carlo sampling

$$\mathbb{E}_{Y^*}[H(y\,|D,x,Y^*)] \approx \frac{1}{S}\sum_{s=1}^{S} H(y\,|D,x,Y_s^*)$$

- How to compute Pareto front samples Y_s^* ?
  - Sample functions from posterior GPs via random Fourier features
  - Solve a cheap MO problem over the sampled functions $\tilde{f}_1 \ldots \tilde{f}_k$ to compute sample Pareto front

MESMO Algorithm [Belakaria et al., 2019]

- How to compute the entropy with respect to a given Pareto front sample Y_s^*?

$$Y_s^* = \{\boldsymbol{v}^1, \dots, \boldsymbol{v}^l\} \; with \; \boldsymbol{v}^i = \{v_1^i, \dots, v_K^i\},$$
$$y_j \leq y_{j_s}^* = \max\{v_1^1, \dots, v_j^l\} \; \forall j \in \{1, \dots, K\}$$

  - Decompose the entropy of a set of independent variables into a sum of entropies of individual variables
  - Model each component y_j as a truncated Gaussian distribution

MESMO Algorithm [Belakaria et al., 2019]

- How to compute the entropy with respect to a given Pareto front sample Y_s^*?

$$Y_s^* = \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^l\} \; with \; \boldsymbol{v}^i = \{v_1^i, \ldots, v_K^i\},$$
$$y_j \leq y_{j_s}^* = \max\{v_1^1, \ldots, v_j^l\} \; \forall j \in \{1, \ldots, K\}$$

$$H(y \mid D, x, Y_s^*) \approx \sum_{j=1}^{K} H\left(y_j \mid D, x, y_{j_s}^*\right)$$

MESMO Algorithm [Belakaria et al., 2019]

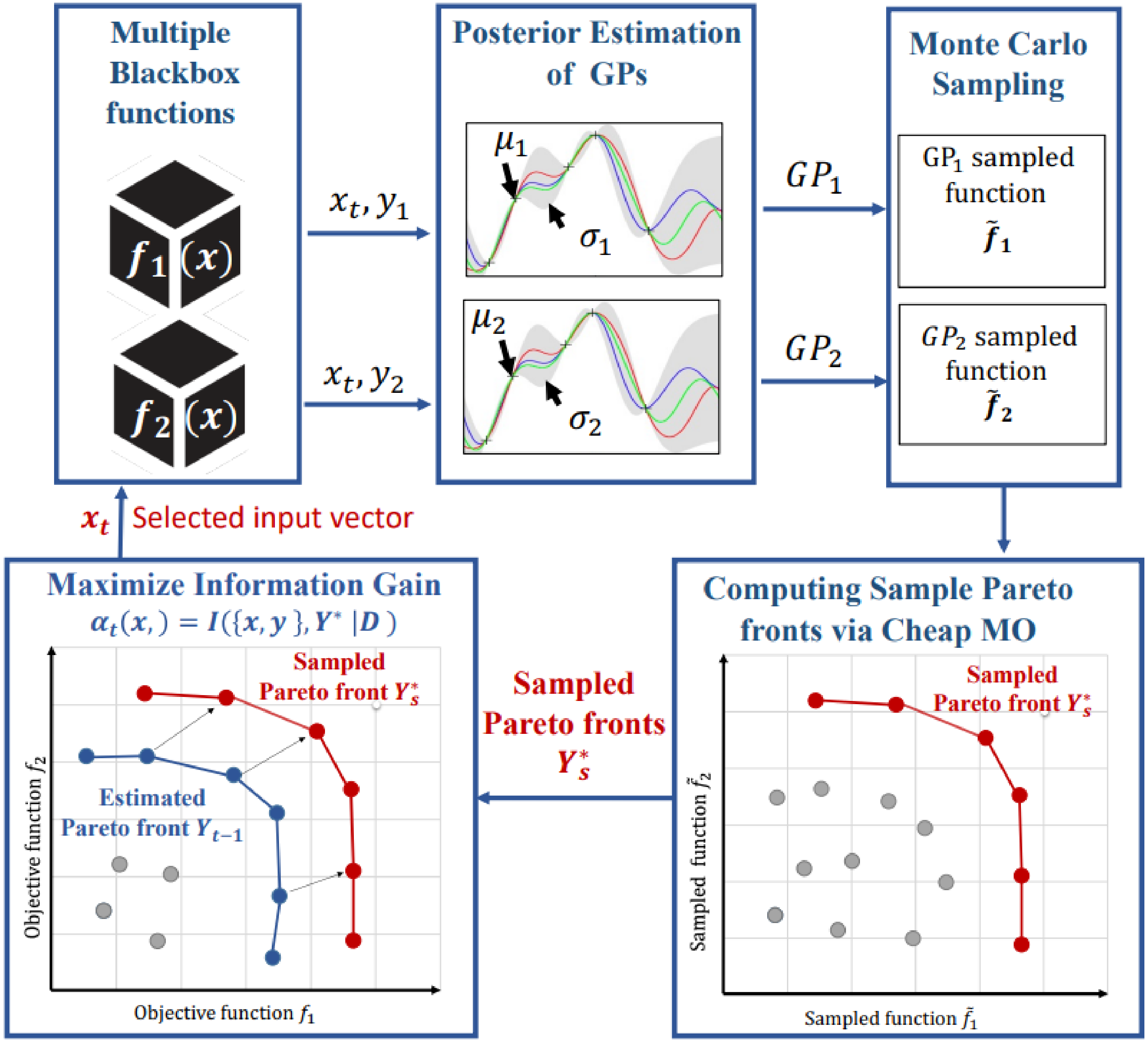
- Final acquisition function

$$\alpha(x) \approx \frac{1}{S} \sum_{s=1}^{S} \sum_{j=1}^{K} \left[\frac{\gamma_s^j(x)\phi\left(\gamma_s^j(x)\right)}{2\Phi\left(\gamma_s^j(x)\right)} - \ln\Phi\left(\gamma_s^j(x)\right) \right]$$

Closed form

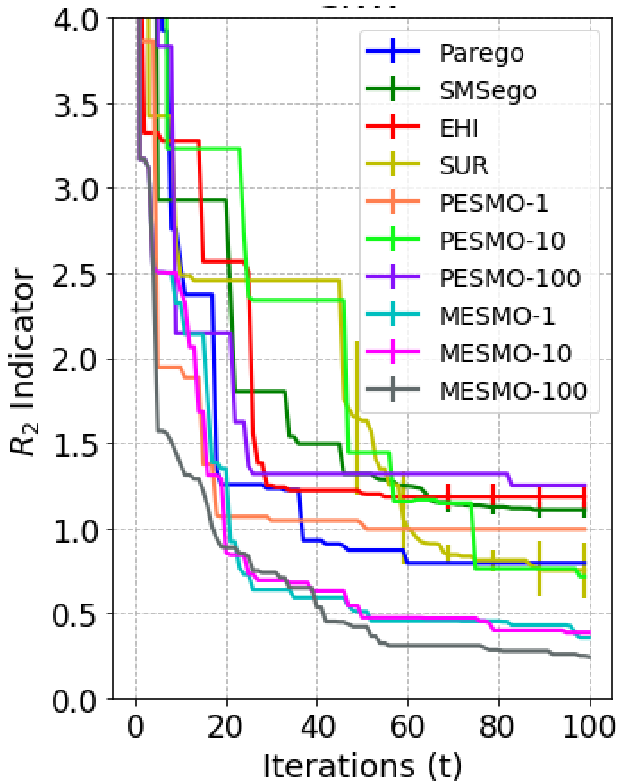
where $\gamma_s^j(x) = \frac{y_{j_s}^{*} - \mu_j(x)}{\sigma_j(x)}$, ϕ and Φ are the p.d.f and c.d.f of a standard normal distribution

MESMO Algorithm [Belakaria et al., 2019]

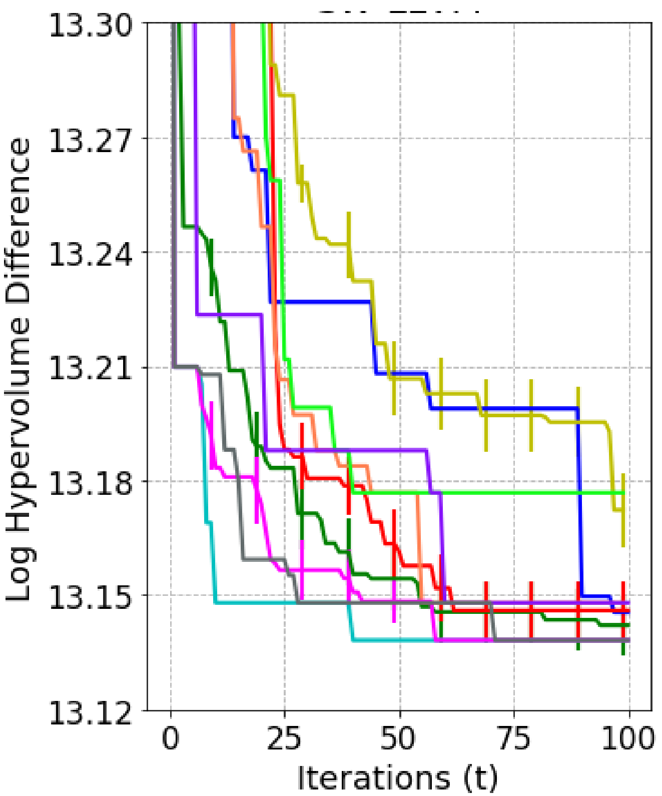


MOBO Experiments and Results #1

Network on Chip Design

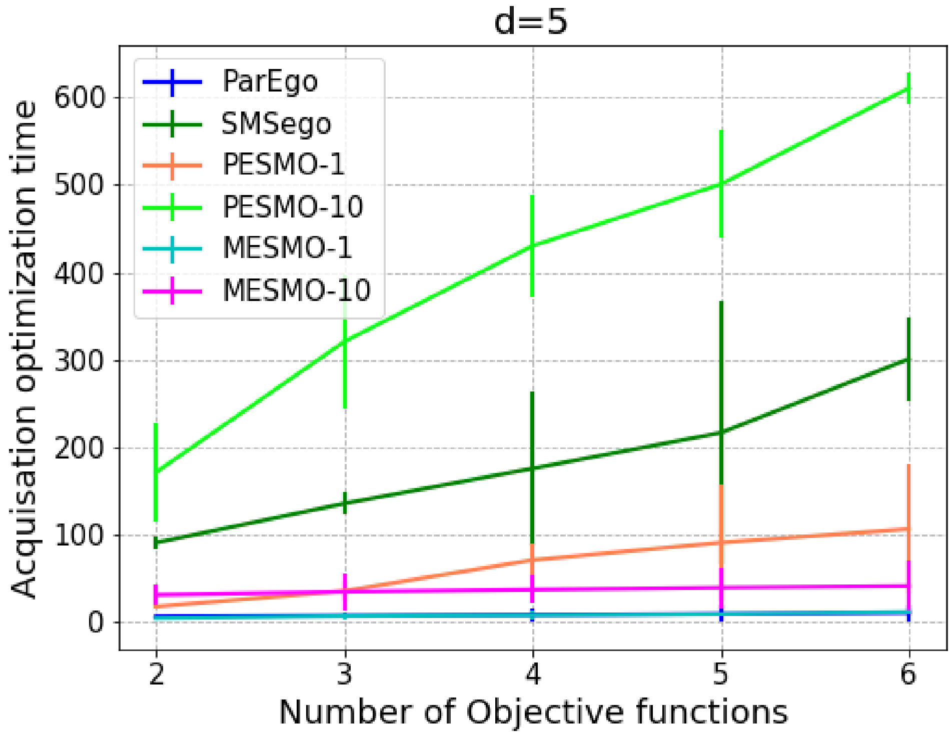


Compiler Settings Optimization



- MESMO is better than PESMO
- MESMO converges faster
- MESMO is robust to the number of samples (even a single sample!)

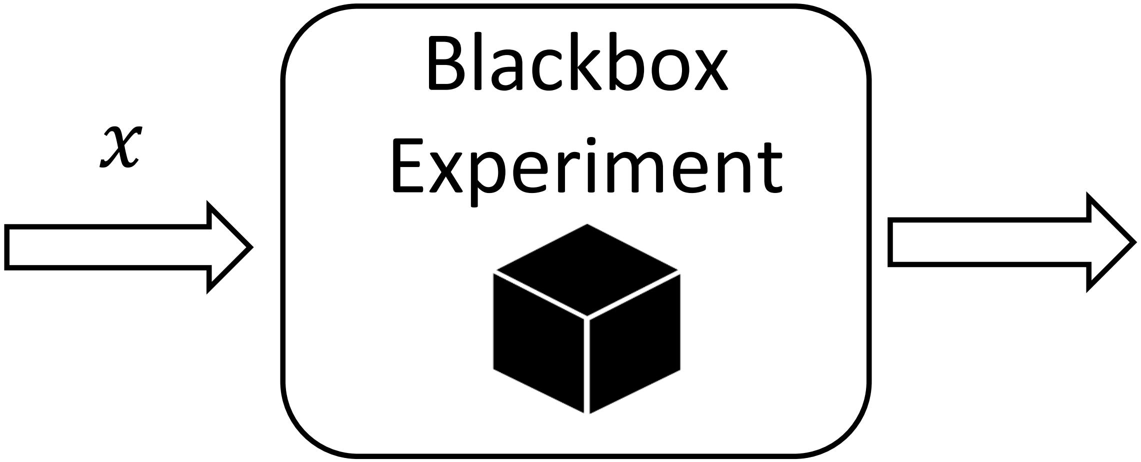
MOBO Experiments and Results #2



- MESMO is highly scalable when compared to PESMO
- MESMO with one sample is comparable to ParEGO
- Time for PESMO and SMSego increases significantly with the number of objectives

Multi-Objective Bayesian Optimization With Black-Box Constraints

MOBO with Black-Box Constraints: The Problem



$f_{1(x)}$ … $f_{K(x)}$

$c_{1(x)}$ … $c_{L(x)}$

Objectives and constraints evaluation of design x

- Goal: find the approximate (optimal) constrained Pareto set by minimizing the total resource cost of experiments

MOBO with Black-Box Constraints: The Problem

Amazon Prime Air autonomous unmanned aerial vehicle (UAV)

- Electrified aviation power system design for UAVs [Belakaria et al., 2021]
  - Multiple Objectives: total energy and mass
  - Safety constraints: thresholds for motor temperature and voltage of cells

MESOC Algorithm [Belakaria et al., 2021]

- Extension of MESMO for constrained setting

$$\alpha(x) \approx \frac{1}{S}\sum_{s=1}^{S}\ [\sum_{j=1}^{K} \frac{\gamma_s^{f_j}(x)\phi\left(\gamma_s^{f_j}(x)\right)}{2\Phi\left(\gamma_s^{f_j}(x)\right)} - \ln\Phi\left(\gamma_s^{f_j}(x)\right) + \sum_{j=1}^{L} \frac{\gamma_s^{c_j}(x)\phi\left(\gamma_s^{c_j}(x)\right)}{2\Phi\left(\gamma_s^{c_j}(x)\right)} - \ln\Phi\left(\gamma_s^{c_j}(x)\right)]$$

Closed form

MESMOC Algorithm [Belakaria et al., 2021]

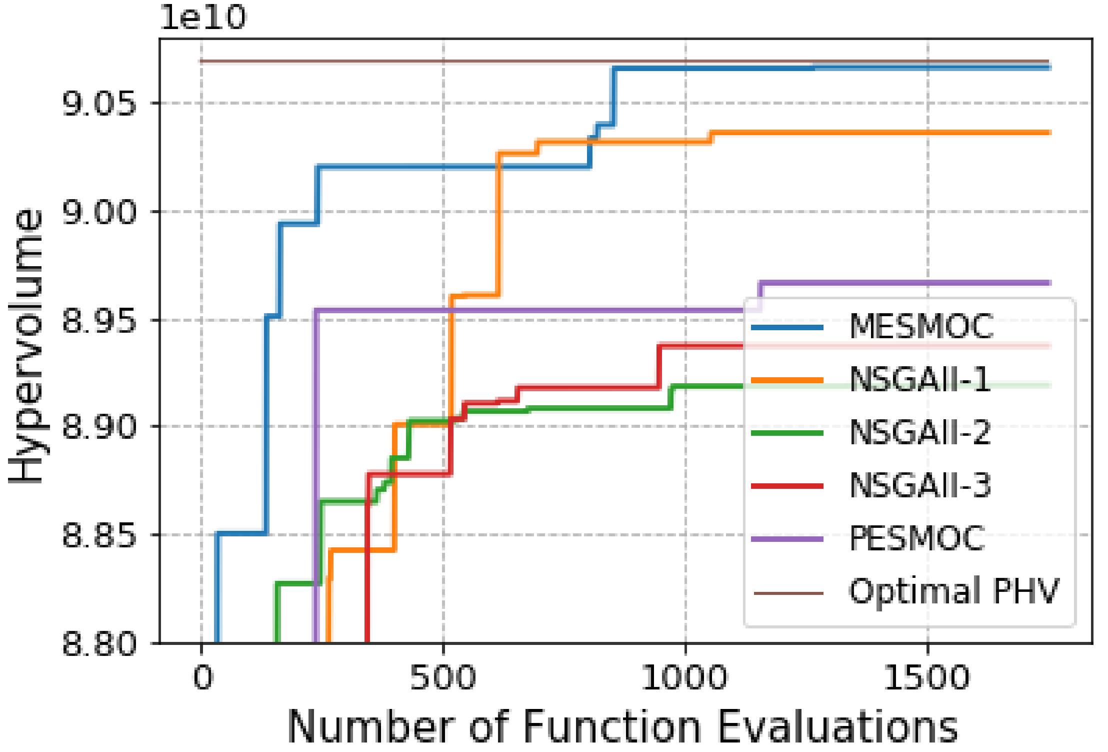
- Solves a cheap MOO over sampled functions ($\widetilde{f_1}, \ldots, \widetilde{f_K}$) constrained by sampled constraints ($\widetilde{c_1}, \ldots, \widetilde{c_L}$)

$$Y_s^* \leftarrow \arg\max_{x \in \chi} (\widetilde{f_1}, \ldots, \widetilde{f_K})$$
$$\text{s.t. } (\widetilde{c_1} \geq 0, \ldots, \widetilde{c_L} \geq 0)$$

- Acquisition function optimization constrained by predictive mean of constraints

$$x_t \leftarrow \arg\max_{x \in \chi} \alpha_t$$
$$\text{s.t. } (\mu_{c_1}(x) \geq 0, \ldots, \mu_{c_L}(x) \geq 0)$$

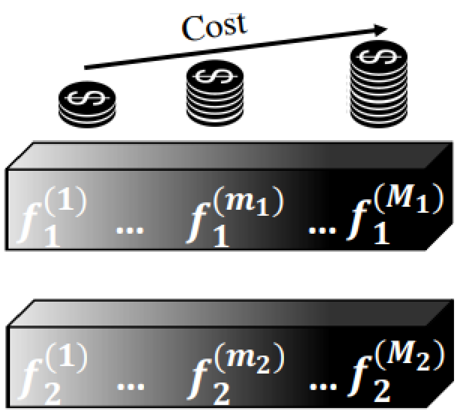
MESMOC Experiments and Results



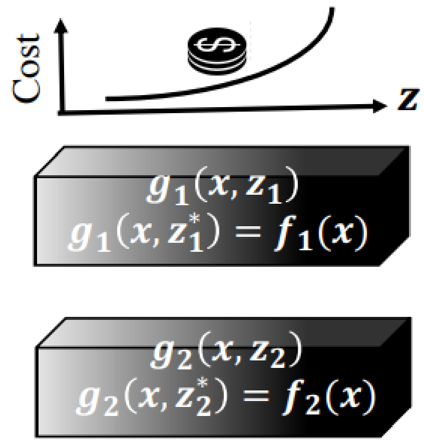
- MESMOC finds near-optimal Pareto front in ~250 evaluations out of ~168,000 designs (<1%)
- 95% of the inputs selected by MESMOC are valid, while the best among baselines was only 39%

Multi-Objective Bayesian Optimization With Multi-Fidelity Function Evaluations

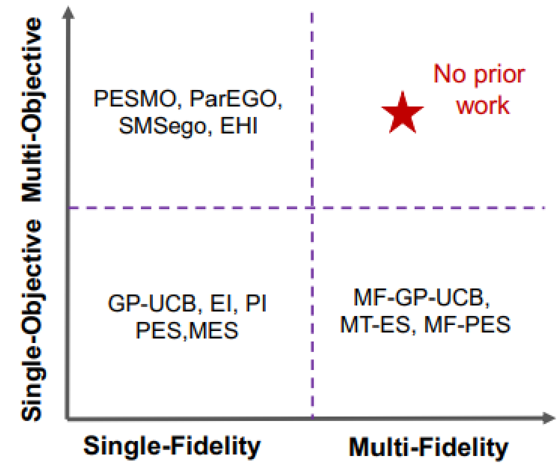
Multi-Fidelity Multi-Objective BO: The Problem



Discrete fidelity



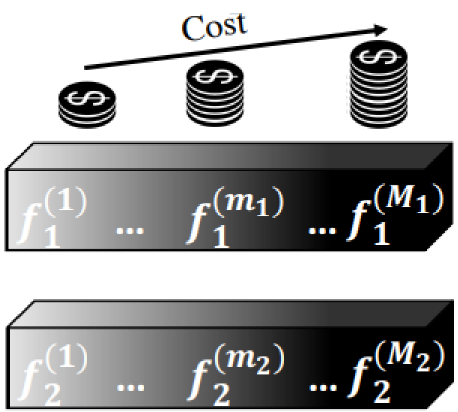
Continuous fidelity

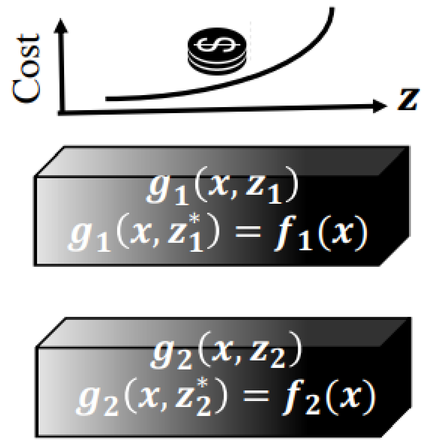


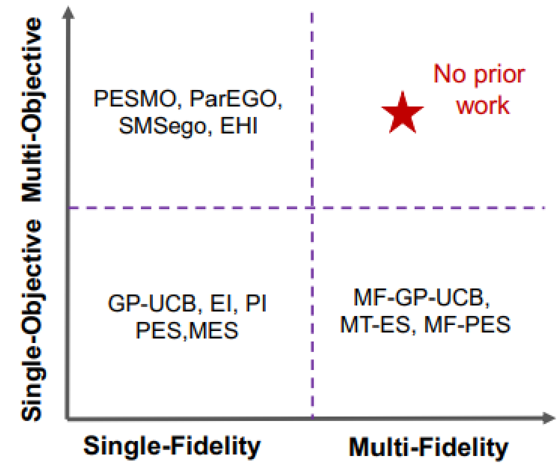
- Continuous-fidelity is the most general case
  - Discrete-fidelity is a special case

- Goal: find the approximate (optimal) Pareto set by minimizing the total resource cost of experiments

Multi-Fidelity Multi-Objective BO: Key Challenges







- How to model functions with multiple fidelities?
- How to join[...] and fidelity-vector pair in each BO iteration?

Already covered

- How to progressively select higher fidelity experiments?

iMOCA Algorithm [Belakaria et al., 2021]

- Key Idea: Select the input and fidelity-vector that maximizes information gain per unit resource cost about the optimal Pareto front Y^*

$$\alpha(\boldsymbol{x}, \boldsymbol{z}) = I(\{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}\}, Y^* | D)/C(\boldsymbol{x}, \boldsymbol{z})$$
$$= (H(\boldsymbol{y}|D, \boldsymbol{x}, \boldsymbol{z}) - \mathbb{E}_{Y^*}[H(\boldsymbol{y}\,|D, \boldsymbol{x}, \boldsymbol{z}, Y^*)])/C(\boldsymbol{x}, \boldsymbol{z})$$

$$= (\textstyle\sum_{j=1}^{K} \ln\left(\sqrt{2\pi e}\,\sigma_{g_j}(\boldsymbol{x}, z_j)\right)$$
$$- \frac{1}{S} \textstyle\sum_{s=1}^{S} \sum_{j=1}^{K} H(y_j |\, D, \boldsymbol{x}, z_j, f_s^{j*}))/\, C(\boldsymbol{x}, \boldsymbol{z})$$

where $C(\boldsymbol{x}, \boldsymbol{z}) = \sum_{j=1}^{K} \frac{c(\boldsymbol{x}, z_j)}{c(\boldsymbol{x}, z_j^*)}$ is the normalized cost over different functions

iMOCA Algorithm [Belakaria et al., 2021]

- **Assumption:** Values at lower fidelities are smaller than maximum value of the highest fidelity $y_j \leq f_s^{j*} \ \forall j \in \{1, \ldots, K\}$

- Truncated Gaussian approximation (Closed-form)

$$\alpha(\boldsymbol{x}, \boldsymbol{z}) \approx \frac{1}{C(\boldsymbol{x},\boldsymbol{z})S} \sum_{s=1}^{S} \sum_{j=1}^{K} \left[\frac{\gamma_s^{(g_j)} \phi\left(\gamma_s^{(g_j)}\right)}{2\Phi\left(\gamma_s^{(g_j)}\right)} - \ln\Phi\left(\gamma_s^{(g_j)}\right) \right]$$

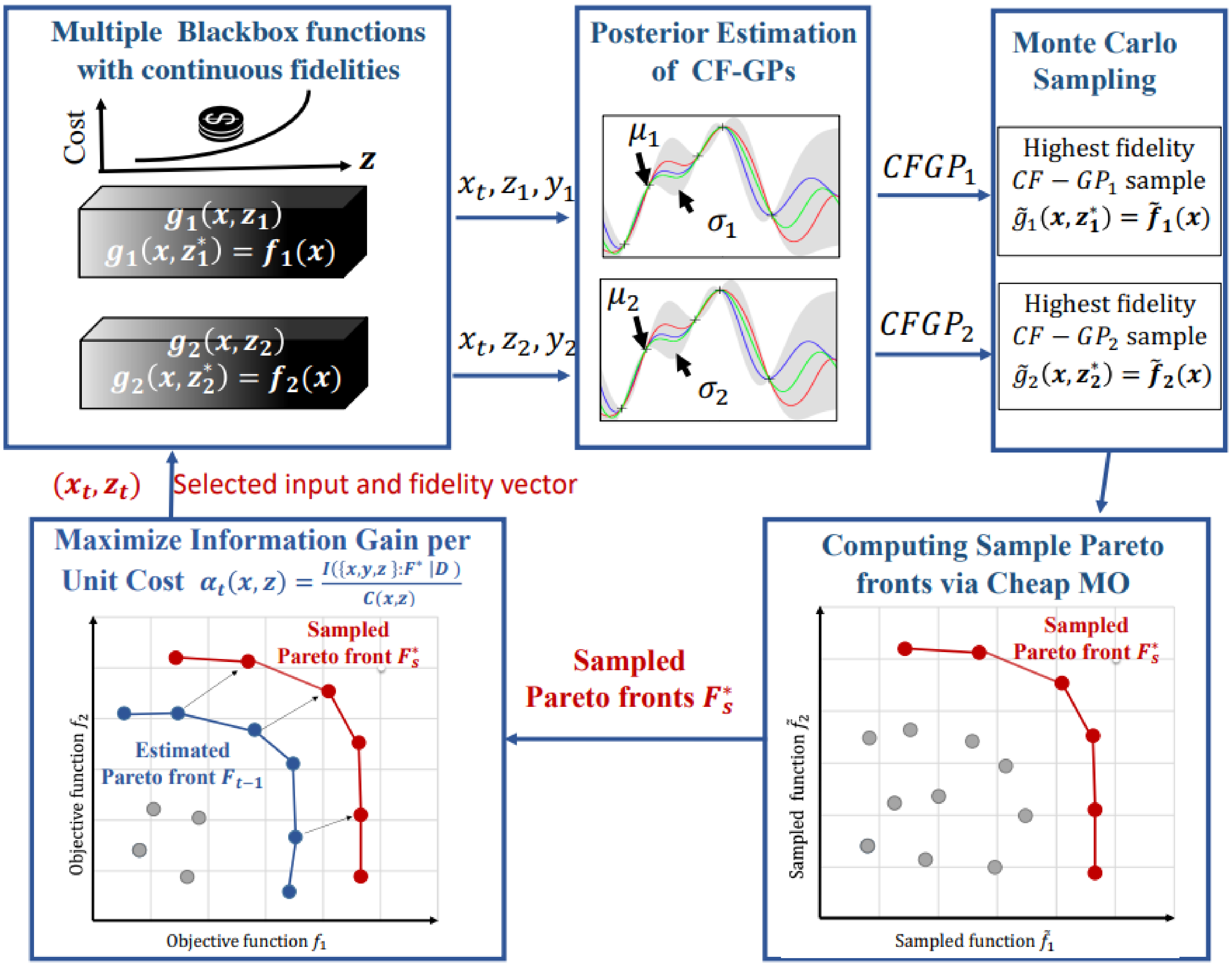
Where $\gamma_s^{(g_j)} = \frac{f_s^{j*} - \mu_{g_j}}{\sigma_{g_j}}$, $\boldsymbol{\phi}$ and $\boldsymbol{\Phi}$ are the p.d.f and c.d.f of a standard normal distribution

iMOCA Algorithm [Belakaria et al., 2021]

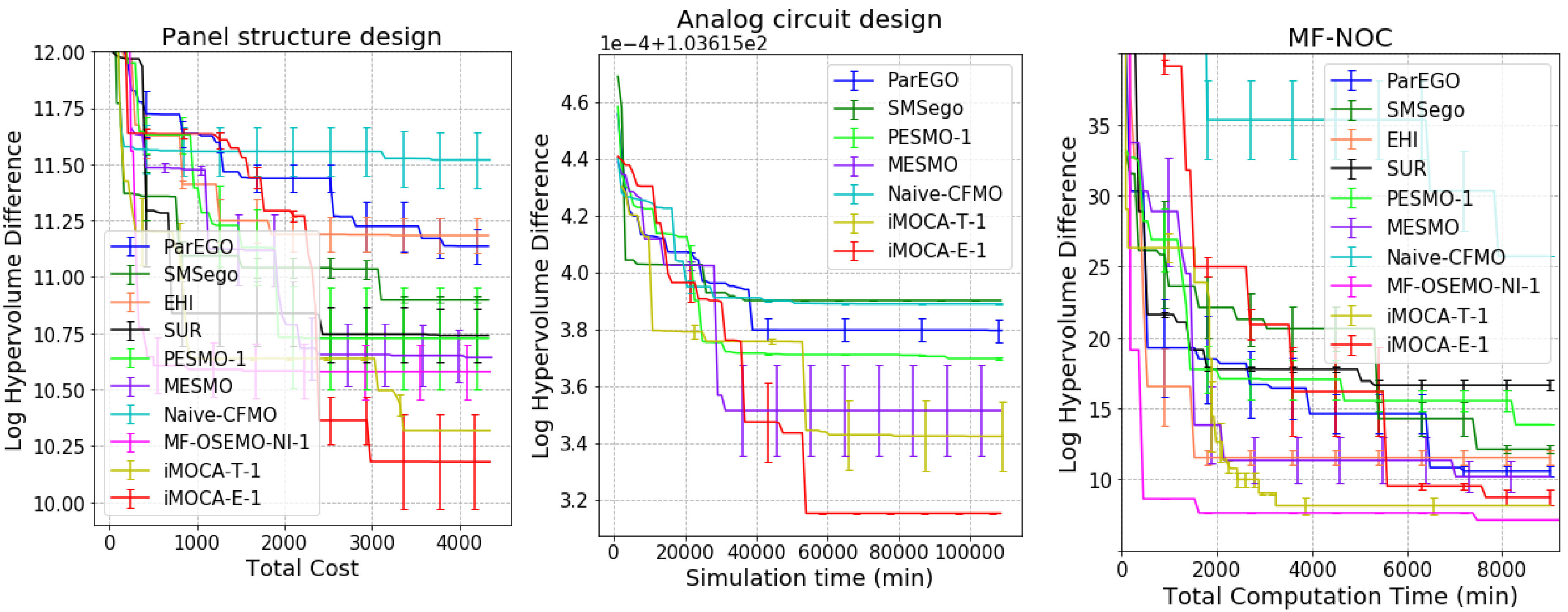
- Challenges of large (potentially infinite) fidelity space
  - Select costly fidelity with less accuracy
  - Tendency to select lower fidelities due to normalization by cost

- iMOCA reduces the fidelity search space using a scheme similar to the BOCA algorithm

iMOCA Algorithm [Belakaria et al., 2021]



iMOCA Experiments and Results



- iMOCA performs better than all baselines
- Both variants of iMOCA converge at a much lower cost
- Robust to the number of samples

iMOCA Experiments and Results

- **Cost reduction factor**
  - Although the metric gives advantage to baselines, the results in the table show a consistently high gain ranging from 52% to 85%

Name	BC	ARS	Circuit	Rocket
$\mathcal{C}_B$	200	300	115000	9500
$\mathcal{C}$	30	100	55000	2000
$\mathcal{G}$	85%	66.6%	52.1%	78.9%

Table: *Best* convergence cost from all baselines $\mathcal{C}_B$, *Worst* convergence cost for iMOCA $\mathcal{C}$, and cost reduction factor $\mathcal{G}$.

Software and code

- github.com/HIPS/Spearmint/tree/PESM
- github.com/belakaria/MESMO
- github.com/belakaria/USeMO
- botorch.org/tutorials/multi_objective_bo
- github.com/yunshengtian/DGEMO
- github.com/belakaria/MESMOC
- github.com/belakaria/MF-OSEMO
- github.com/belakaria/iMOCA

Questions ?

Summary and Open Challenges in BO

Outline of the Tutorial

- Background on GPs and Single-Objective BO
- Bayesian Optimization over Combinatorial Spaces
- Bayesian Optimization over Hybrid Spaces

Break

- Multi-Fidelity Bayesian Optimization
- Constrained Bayesian Optimization
- Multi-Objective Bayesian Optimization
- Summary and Outstanding Challenges in BO

Open Challenges in BO

- **High-dimensional BO**
  - Need more effective approaches for high-dimensional spaces
- **BO over Combinatorial Structures**
  - How to combine domain knowledge, kernels, and (geometric) deep learning to build effective surrogate models?
  - Effective methods to select large and diverse batches?
- **BO over Hybrid Spaces**
  - Methods to sample functions from GP posterior?
  - Effective latent space BO methods?

Open Challenges in BO

- **Constrained BO**
  - Need more effective approaches for input spaces, where no. of invalid inputs >> no. of valid inputs

- **BO over Nested Function Pipelines**
  - Relatively less explored problem

- **BO with Resource Constraints**
  - Real-world experiments need resources and setup time
  - Critical for BO deployment in science and engineering labs

Acknowledgements: Collaborators

- Nano-porous materials

- Hardware design





- Microbial fuel cells

- Electric transportation systems

- Catalysis


- Agriculture





Acknowledgements: Funding

Primary source

Questions ?