Grey-Box Bayesian Optimization

Peter I. Frazier
Cornell University
Uber


Raúl Astudillo

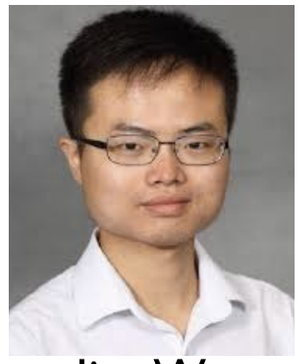

Scott Clark


Matthias Poloczek


Saul Toscano-Palmerin

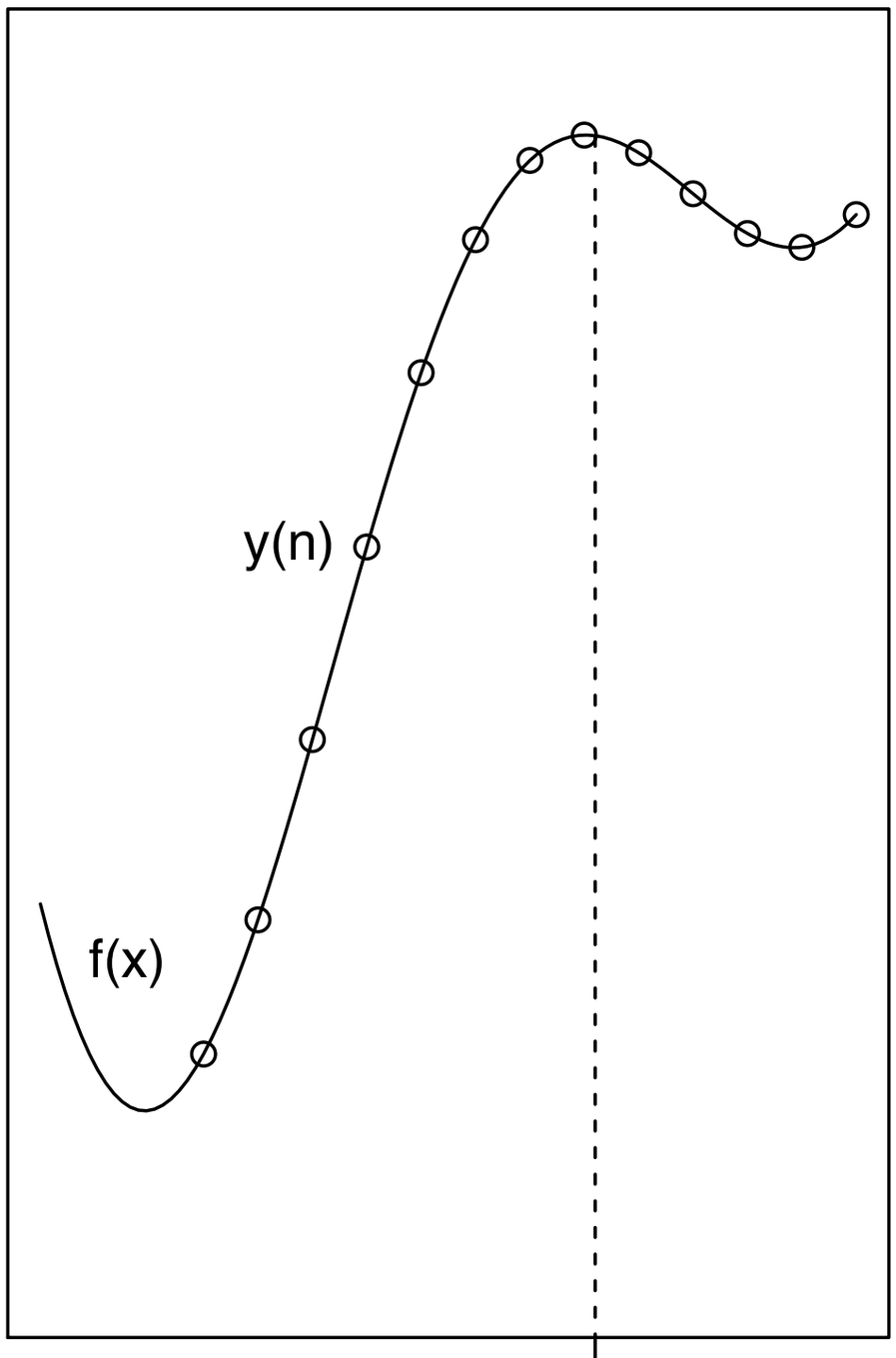

Jialei Wang


Andrew Wilson


Jian Wu

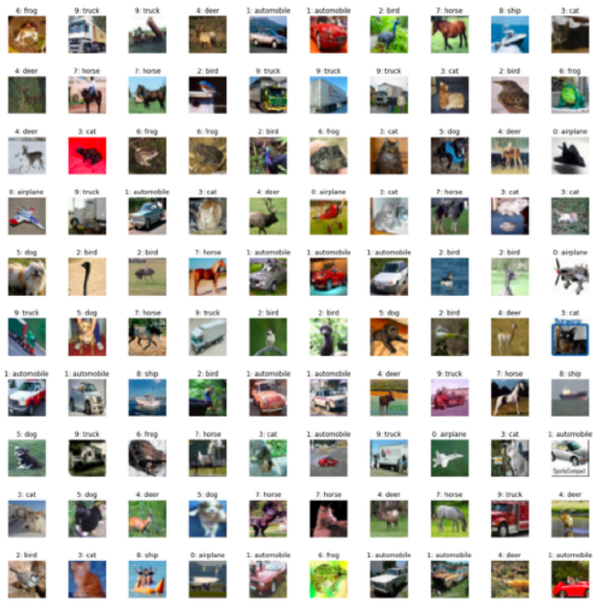
Thank you for funding from AFOSR and NSF!

Bayesian Optimization traditionally considers this black-box optimization problem

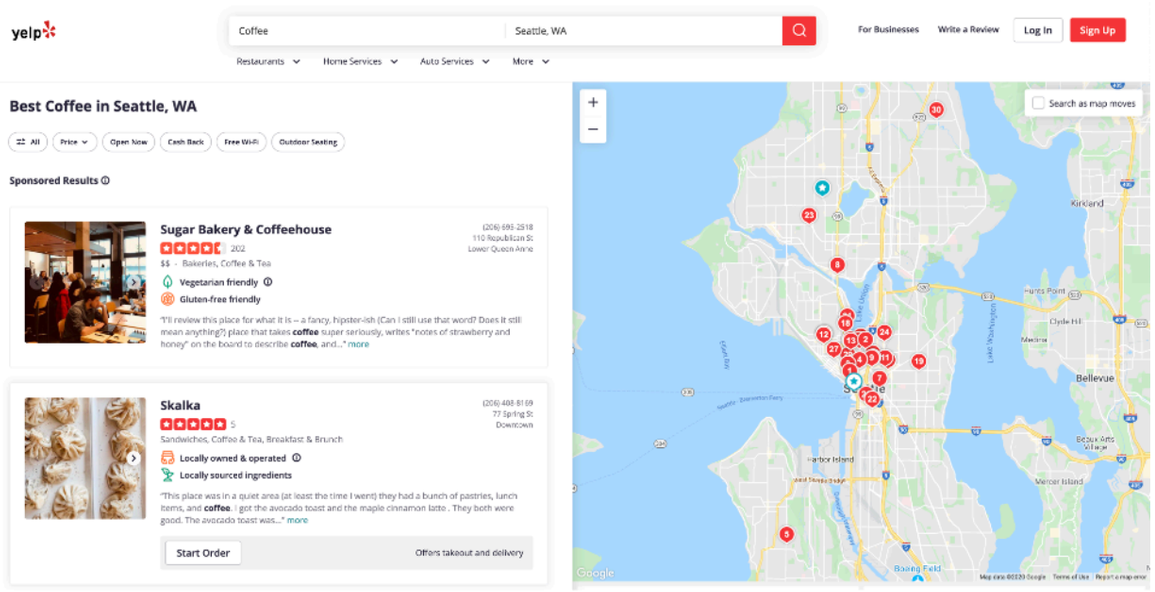


- We'd like to optimize $F : \mathbb{R}^d \rightarrow \mathbb{R}$, where $d < 20$.
- F's feasible set A is simple, e.g., box constraints.
- F is continuous but lacks special structure, e.g., concavity, that would make it easy to optimize.
- F is derivative-free: evaluations do not give gradient information.
- F is expensive to evaluate: the # of times we can evaluate it is severely limited.
- F may be noisy. If noise is present, we'll assume it is independent and normally distributed, with common but unknown variance.

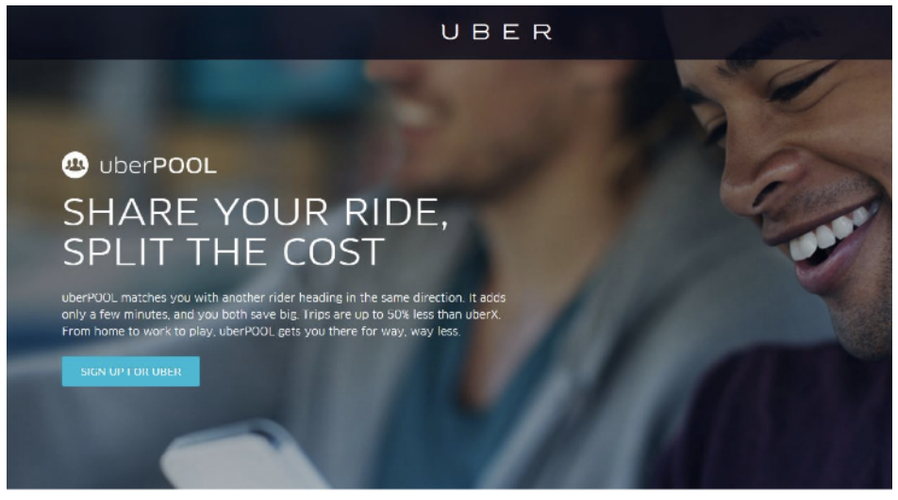
BayesOpt is useful



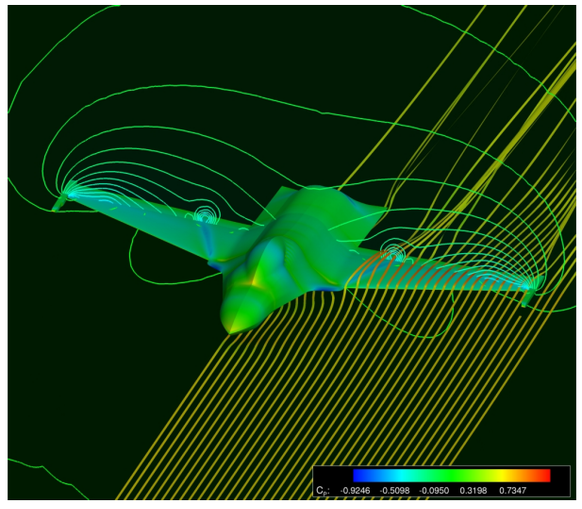
AutoML



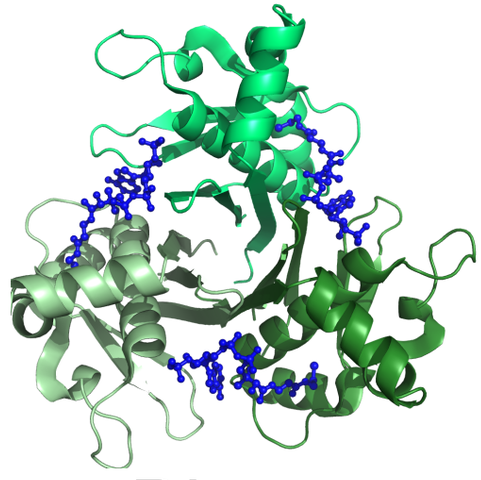
App / Website UX & Algo Design



Market Design



Engineering Design



Drug Discovery & Materials Design

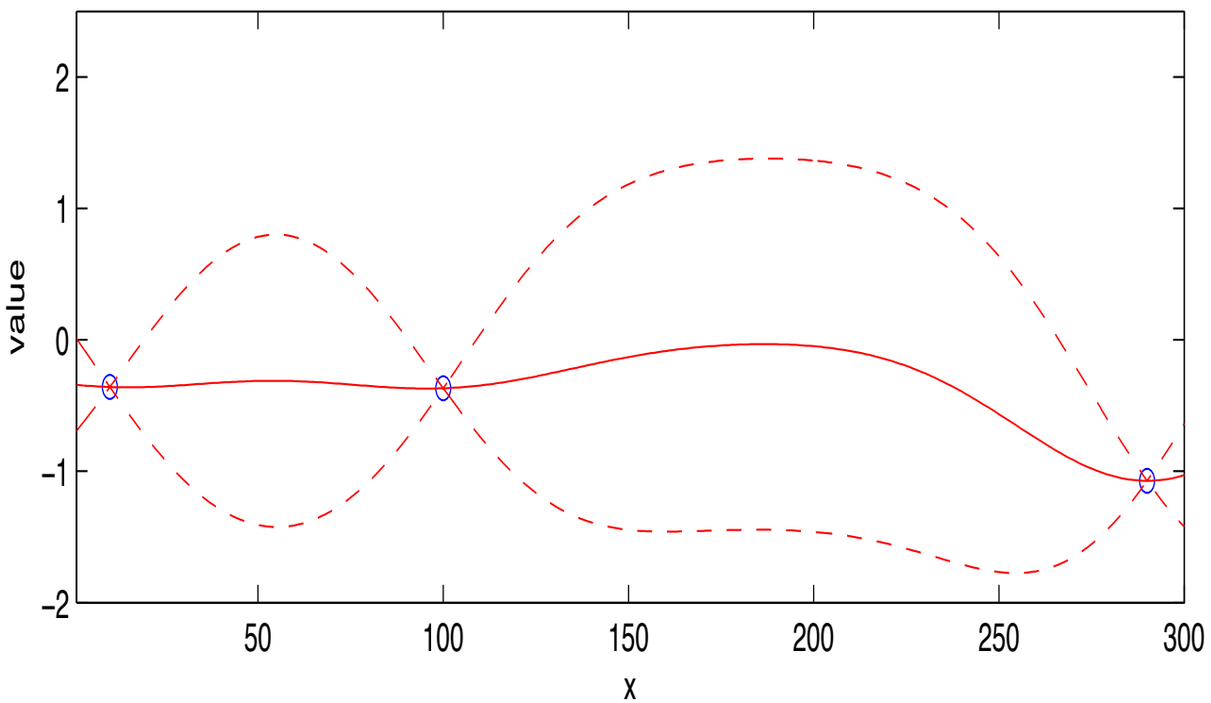
Here's how BayesOpt works, at a high level

Choose an appropriate Bayesian prior on f

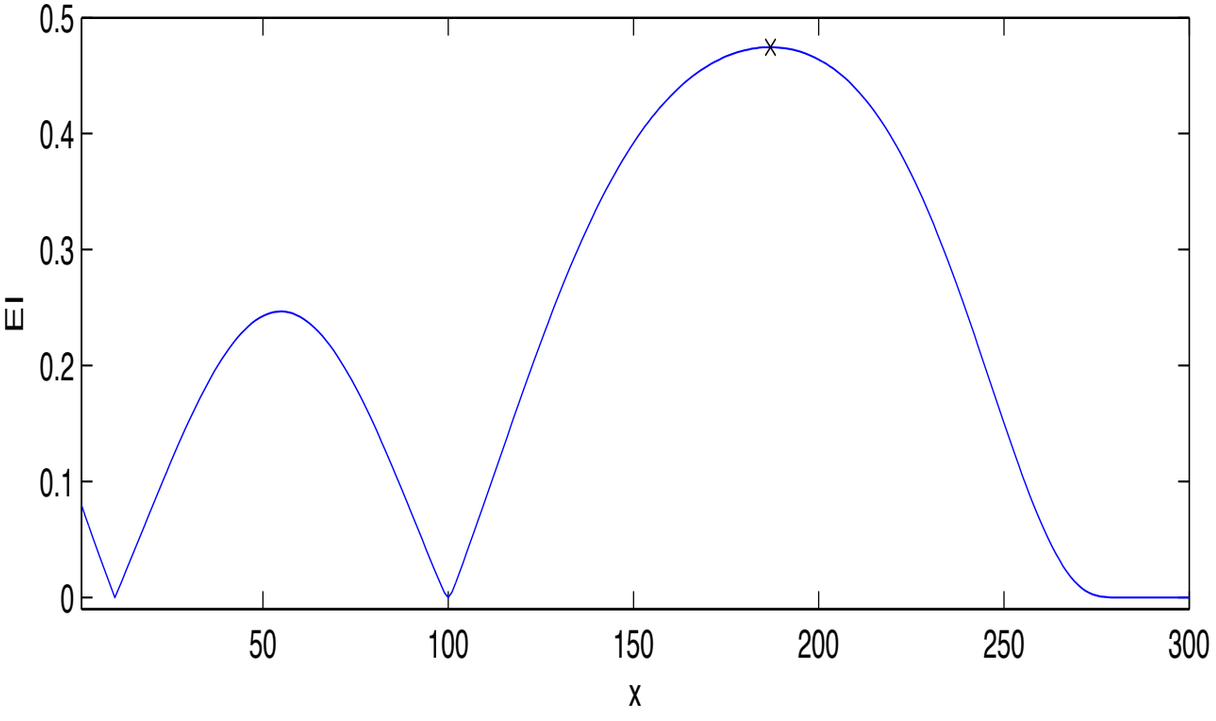
How? Typically we:
(1) assume a Gaussian process prior whose kernel depends on parameters
(2) sample f at a few randomly selected points
(3) use the samples to choose the parameters of the kernel

```
while (budget is not exhausted) {
        Find x that maximizes acquisition(x,posterior)
        Sample x & observe F(x)
        Update the posterior distribution on F
      }
```

Expected Improvement is a widely used acquisition function



- We've evaluated $x^{(1)},\ldots x^{(n)}$, & observed $f(x^{(1)}),\ldots,f(x^{(n)})$ without noise.
- The best value observed is $f^*=\max(f(x^{(1)}),\ldots,f(x^{(n)}))$.



- If we evaluate at x, we observe f(x).
- The *improvement* is $\{f(x)-f^*\}^+$
- The *expected improvement* is $EI(x)=E[\{f(x)-f^*\}^+]$

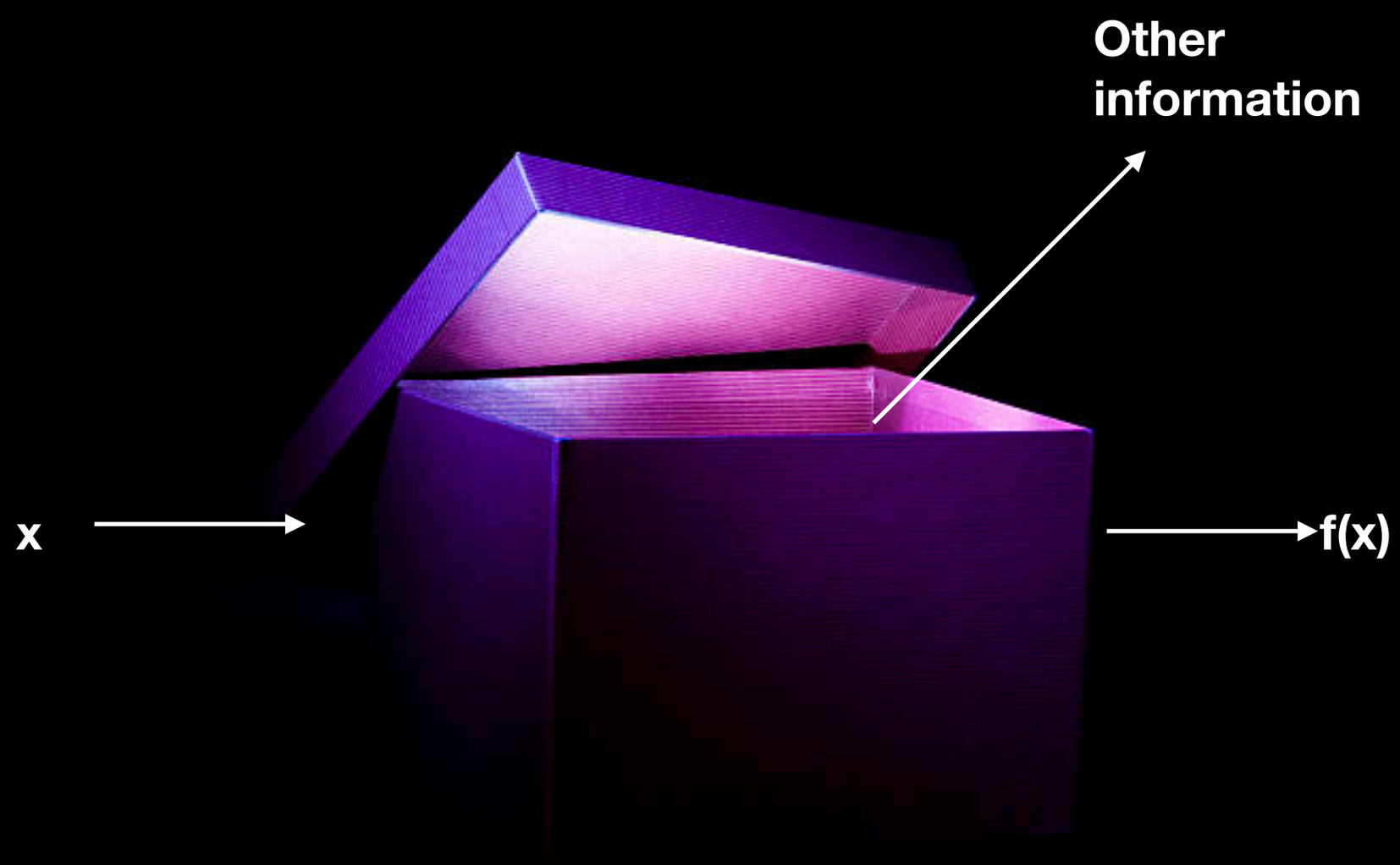
Bayesian Optimization treats the objective function as a black box

Goal: Solve $\min_x f(x)$

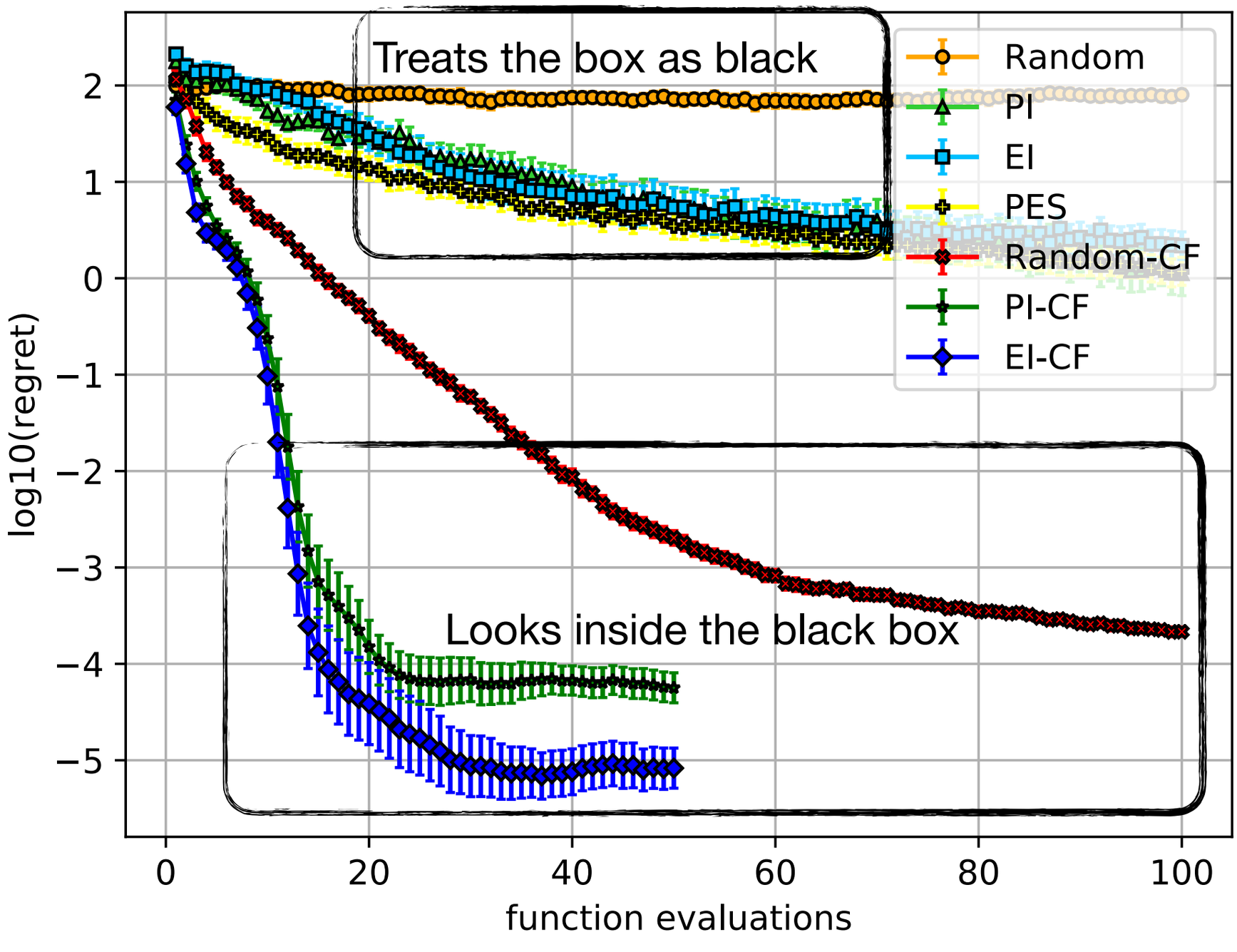
where f(x) is a non-convex derivative-free time-consuming black-box

Kushner 1964; Mockus 1989; Jones, Schonlau & Welch 1998

We can do better by looking inside the box



We can do a lot better by looking inside the box



Compositions, Nested Codes, Function Networks

Wild 2017
Marque-Pucheu, Perrin & Garnier 2019
Astudillo & F. 2019
Uhrenholt & Jensen 2019
Astudillo & F. 2021

Objective-Constituent Evaluations

Williams et al. 2000
Swersky, Snoek & Adams 2013
Marzat et al. 2013
Bogunovic et al. 2018
Toscano-Palmerin & F. 2021
Cakmak, Astudillo & F. 2020
Marque-Pucheu, Perrin & Garnier 2020
Nguyen et al. 2021

Multi-fidelity BO (with inside-the-box observations)

Huang et al. 2006
Forrester et al. 2007
Kandasamy et al. 2017
Poloczek, Wang & F. 2017
Wu, Poloczek, Wilson & F. 2019
Peherstorfer et al. 2018
Takeno et al. 2020

Gradients

J. Wu, Poloczek, Wilson & F. 2017
A. Wu, Aoi, Pillow 2017
Silvola, Vehtari, Vanhatalo, Gonzalez, Anderson 2018
Chen, Qiu, Gao, Jiang, Yang 2020

Compositions, Nested Codes, Function Networks

Wild 2017
Marque-Pucheu, Perrin & Garnier 2019
Astudillo & F. 2019
Uhrenholt & Jensen 2019
Astudillo & F. 2021

Objective-Constituent Evaluations

Williams et al. 2000
Swersky, Snoek & Adams 2013
Marzat et al. 2013
Bogunovic et al. 2018
Toscano-Palmerin & F. 2021
Cakmak, Astudillo & F. 2020
Marque-Pucheu, Perrin & Garnier 2020
Nguyen et al. 2021

Multi-fidelity BO (with inside-the-box observations)

Huang et al. 2006
Forrester et al. 2007
Kandasamy et al. 2017
Poloczek, Wang & F. 2017
Wu, Poloczek, Wilson & F. 2019
Peherstorfer et al. 2018
Takeno et al. 2020

Gradients

J. Wu, Poloczek, Wilson & F. 2017
A. Wu, Aoi, Pillow 2017
Silvola, Vehtari, Vanhatalo, Gonzalez, Anderson 2018
Chen, Qiu, Gao, Jiang, Yang 2020

Bayesian Optimization of Composite Functions

Goal: Solve $\max_{x \in \mathcal{X}} f(x),$

where: $f(x) = g(h(x))$

$h : \mathcal{X} \subset \mathbb{R}^d \to \mathbb{R}^m$ is a time-consuming black box

$g : \mathbb{R}^m \to \mathbb{R}$ and its gradient are fast to compute

$\mathcal{X}$ is a simple compact set in d < 20 dimensions, e.g., a hyper-rectangle

Raúl Astudillo
ICML 2019

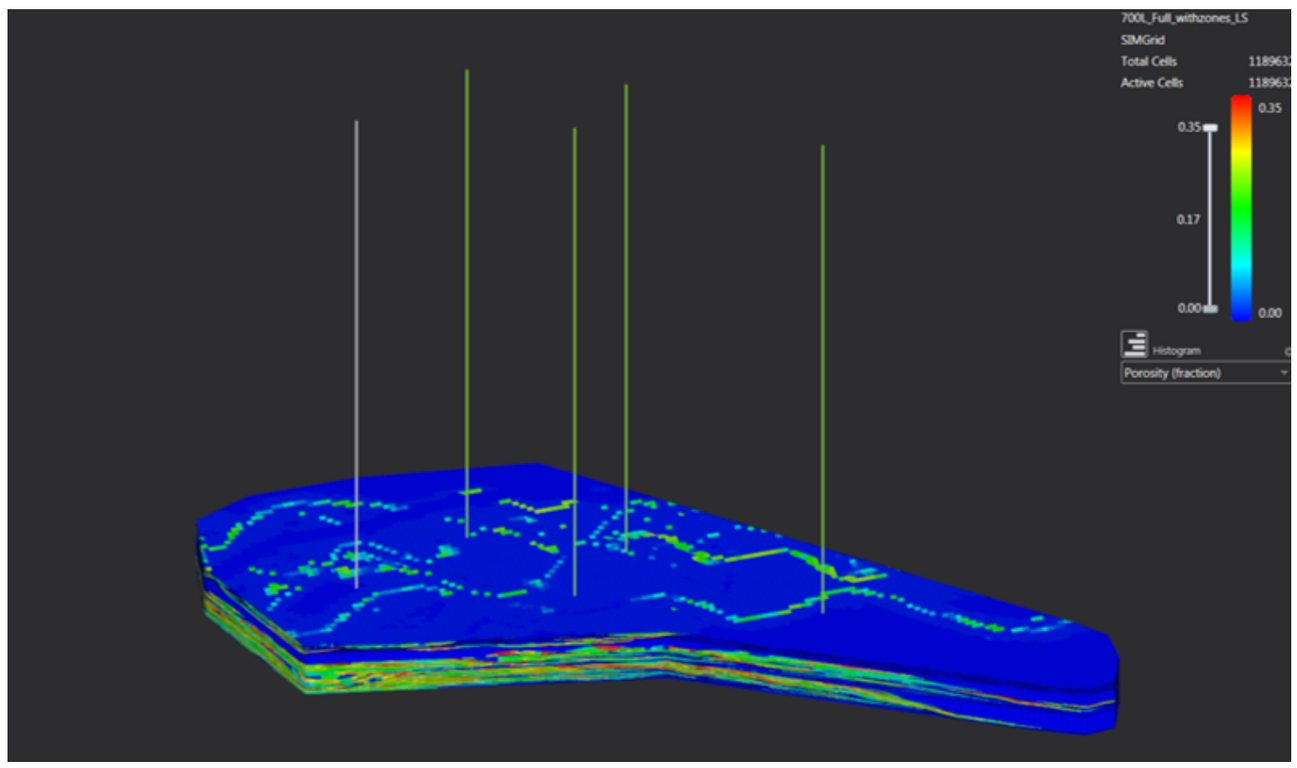
Example: Inverse Problems with Time-Consuming Black Box Forward Models

$$f(x) = g(h(x)) = -\sum_{j=1}^{m}(h_j(x) - y_j)^2,$$

where: x is a parameter vector to calibrate

y is a vector of observational data (e.g., pressures at different test wells)

$h_j(x)$ is the forward model's prediction for the j[th] observation



(Joint work with ExxonMobil)

Other Examples

Inverse Reinforcement Learning (RL)

- y = observed behavior
- h(x) = prediction for observed behavior from RL solver
- g(h(x)) = sum of squared errors (+ regularization)

Materials design

- h(x) = vector of material attributes
- g(h(x)) = performance measure over attributes

Aircraft design

- h(x) = lift and draft as a function of angle of attack and velocity
- g(h(x)) = fuel burn over a mission profile

Related Work

- **Non-Bayesian local/convex optimization of composite functions:**
  Nesterov 2013, Burke & Ferris 1995, Burke 1985, Powell 1983, Fletcher 1982, Anderson and Osborne 1977

- **Non-Bayesian surrogate-based derivative-free nonlinear least squares**:
  Wild 2014; Kelley 2011; Zhang, Conn, Scheinberg 2010

- **Calibration of time-consuming computer models:**
  Overstall & Woods 2013;
  Bliznyuk, Ruppert, Shoemaker, Regis, Wild & Mugunthan 2008

- **BayesOpt with time-consuming constraints:**
  Schonlau, Welch, & Jones, 1998; Gardner, Kusner, Xu, Weinberger, & Cunningham, 2014; Picheny, Gramacy, Wild, & Le Digabel 2016.

- **BayesOpt for sums / multi-task BayesOpt:**
  Swersky, Snoek & Adams, 2013; Toscano-Palmerin & F. 2021

- **Nested Computer Codes (for prediction, rather than optimization)**:
  Marque-Pucheu 2018; Marque-Pucheu, Perrin & Garnier 2019; Marque-Pucheu, Perrin & Garnier 2020

- **Work building on Astudillo & F. ICML 2019**
  Uhrenholt & Jensen 2019; Balandat , Karrer, Jiang, Daulton, Letham, Wilson, Bakshy 2020; Cakmak, Astudillo & F. 2020; Astudillo & F. 2021

Standard BayesOpt

while (budget is not exhausted) {

Fit a Gaussian process to observations x, f(x)

Find x that maximizes $EI(x) = E[\{f(x)-f^*\}^+]$

Observe f(x)

}

*Recall: objective is f(x) = g(h(x))

Grey-Box BayesOpt

while (budget is not exhausted) {

Fit multi-output Gaussian process regression to observations x, h(x)

Find x that maximizes a new acquisition function, EI-CF(x) = $E[\{g(h(x))-f^*\}^+]$

Observe h(x), f(x)

}

*Recall: objective is f(x) = g(h(x))

Let's see why it works with an **example**

- x is a parameter of a simulator,
- $h(x)$ is simulator's prediction under x,
- y is our observed data.

We want to solve

$$\min_x (h(x) - y)^2.$$

Standard BayesOpt

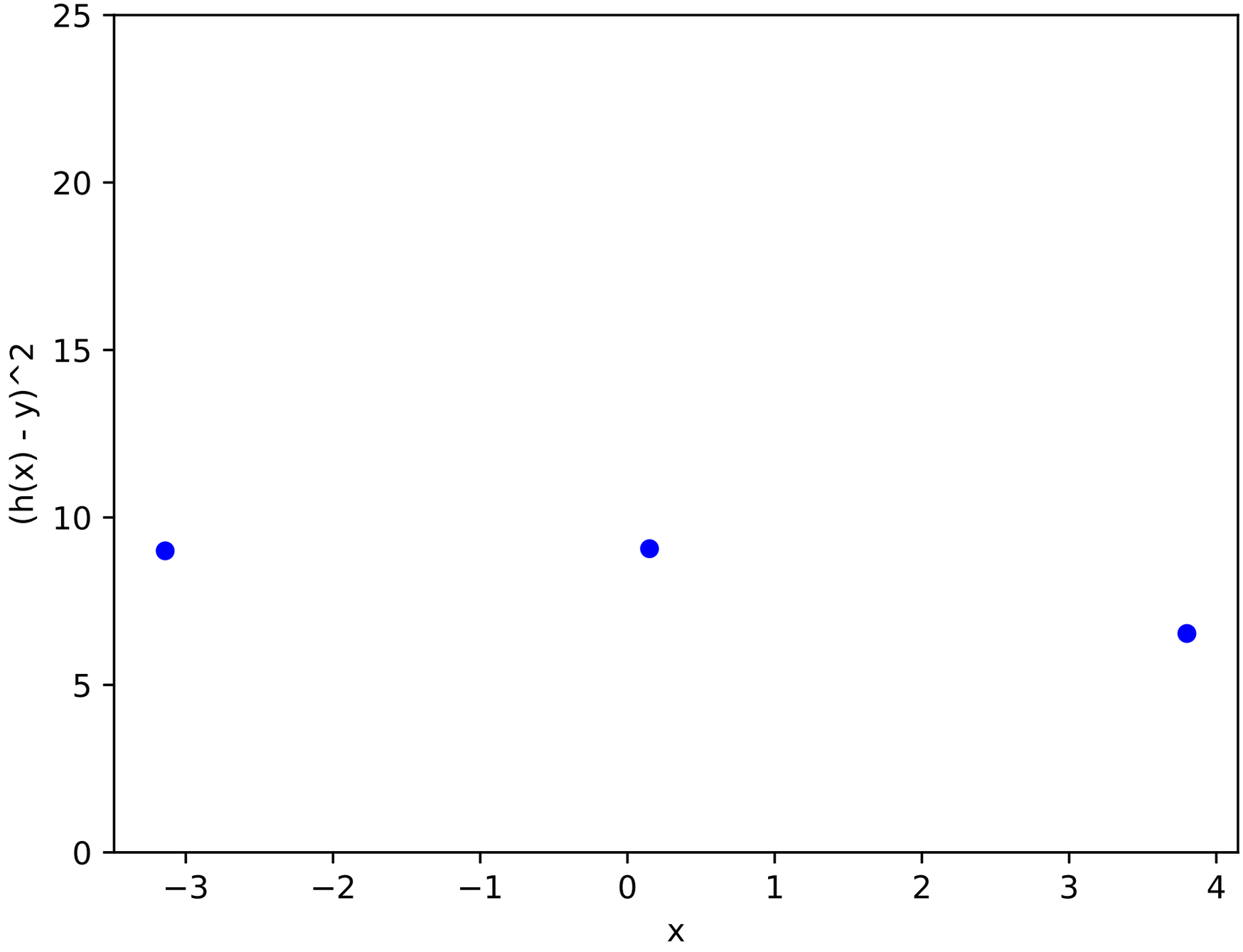


Figure: Evaluations of $(h(x) - y)^2$

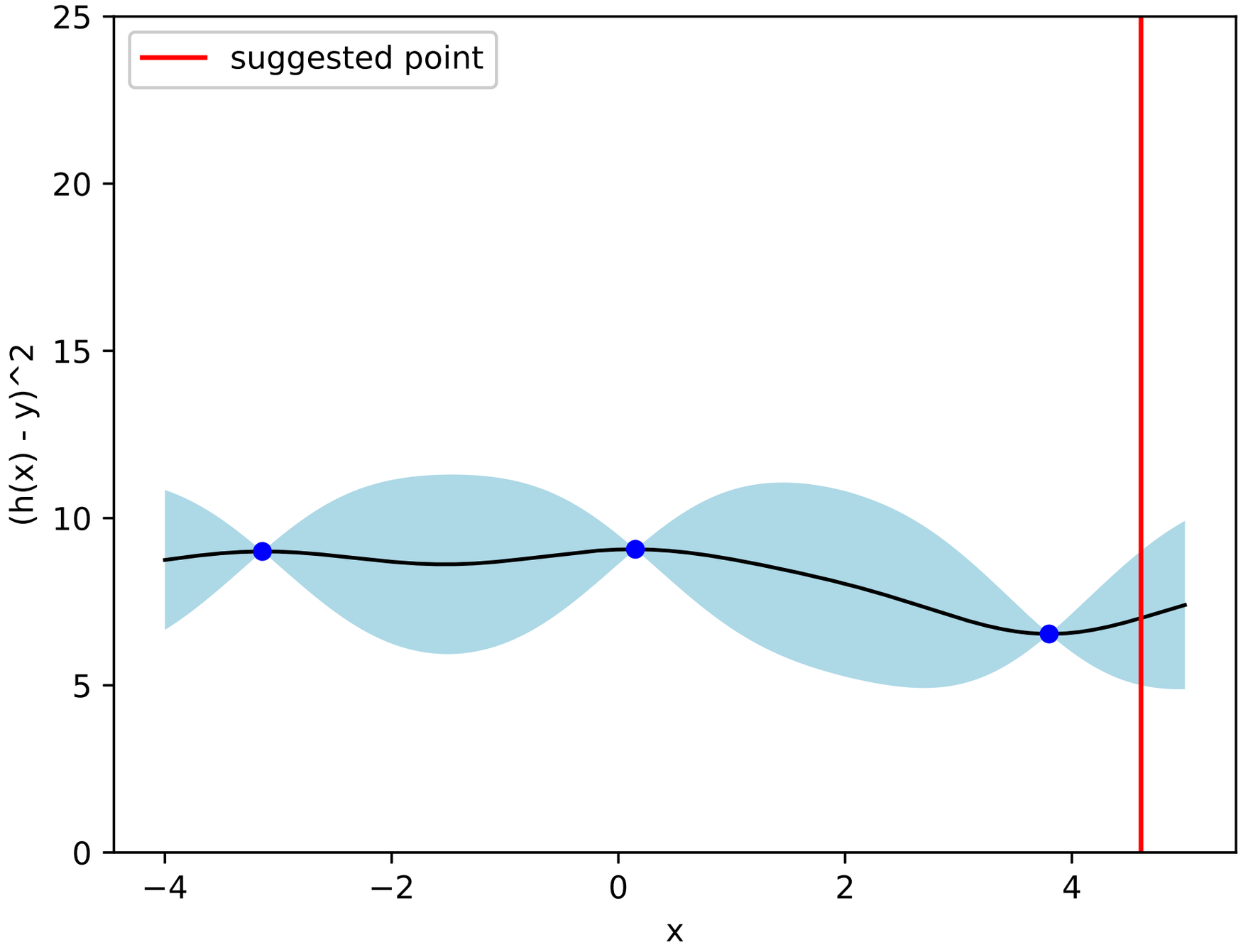


Figure: GP posterior on $(h(x) - y)^2$

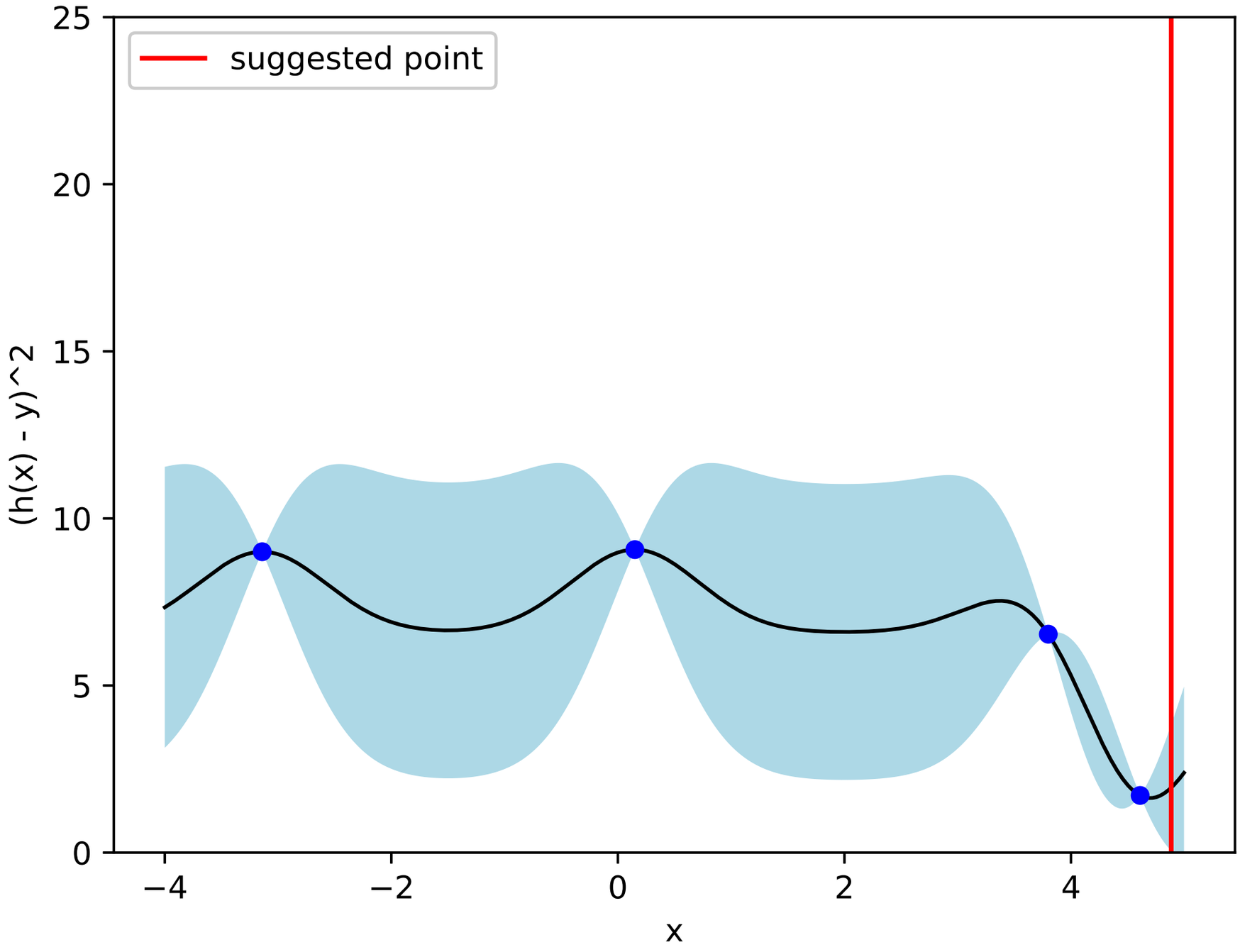
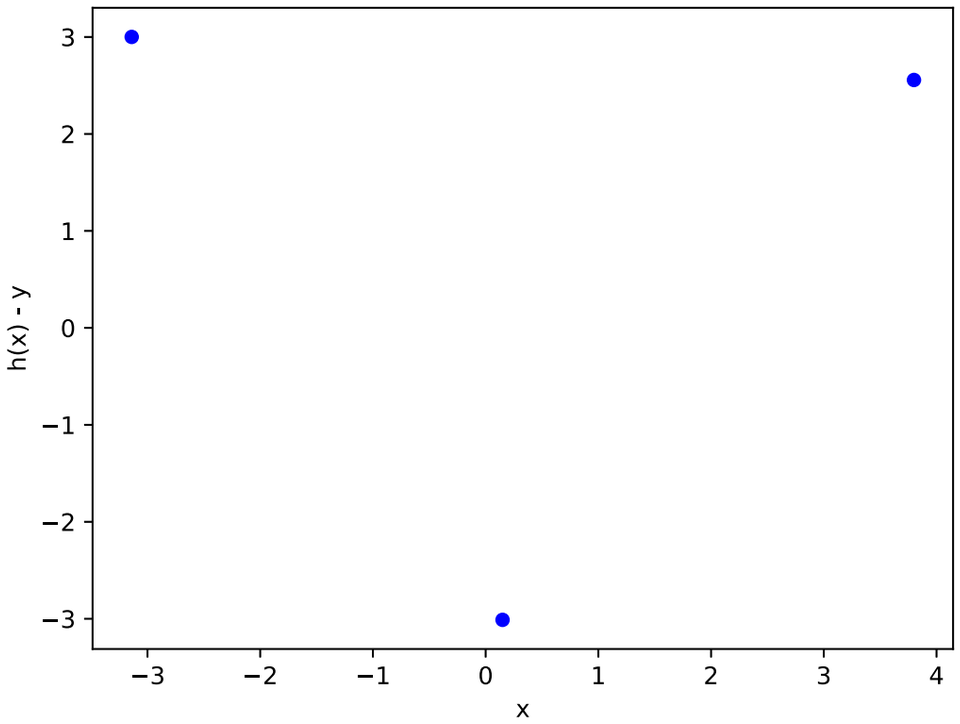
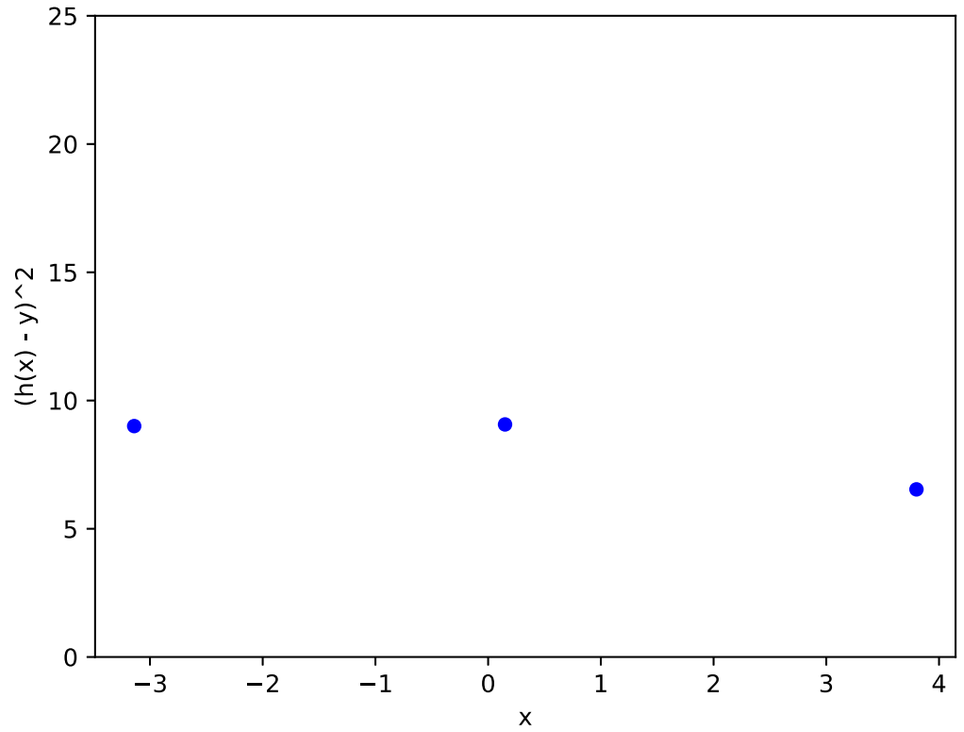


Figure: GP posterior on $(h(x) - y)^2$

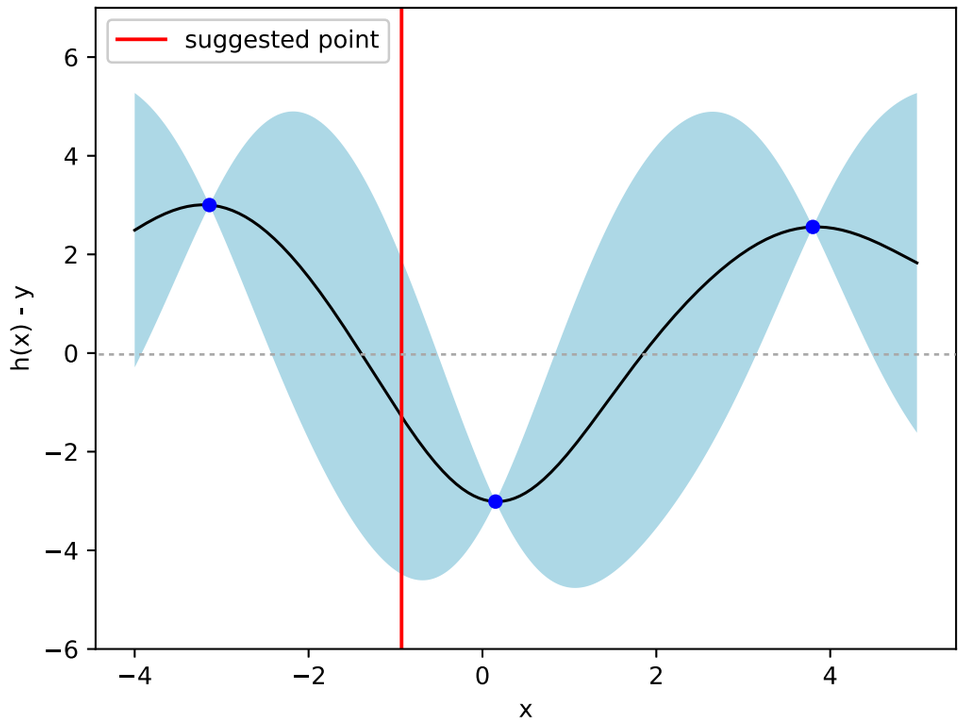
Grey-Box BayesOpt for composite functions



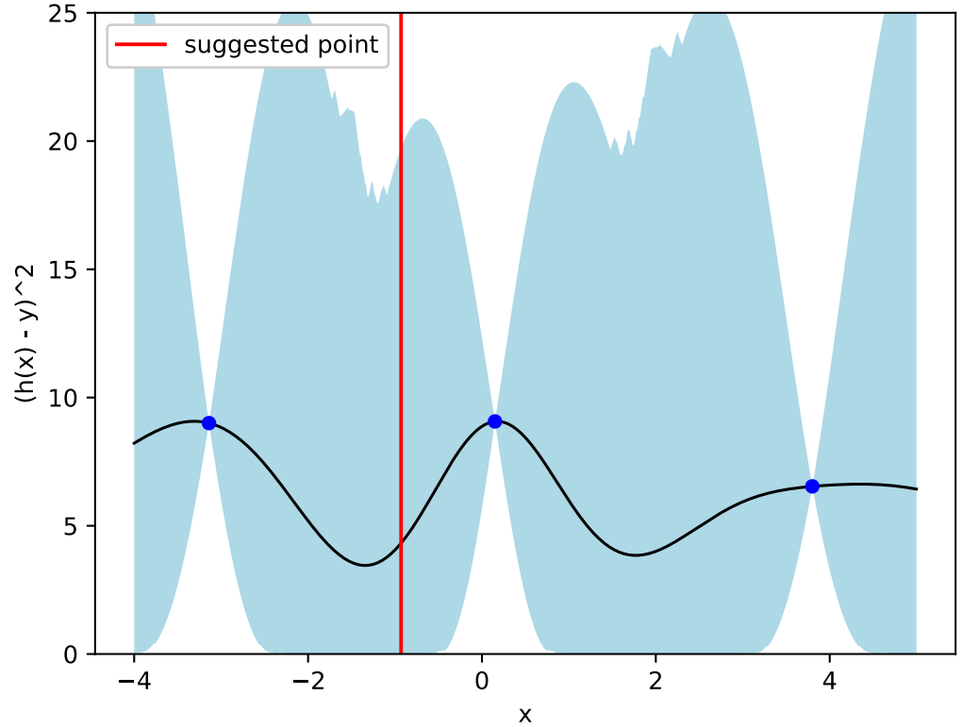
(a) Evaluations of $h(x) - y$



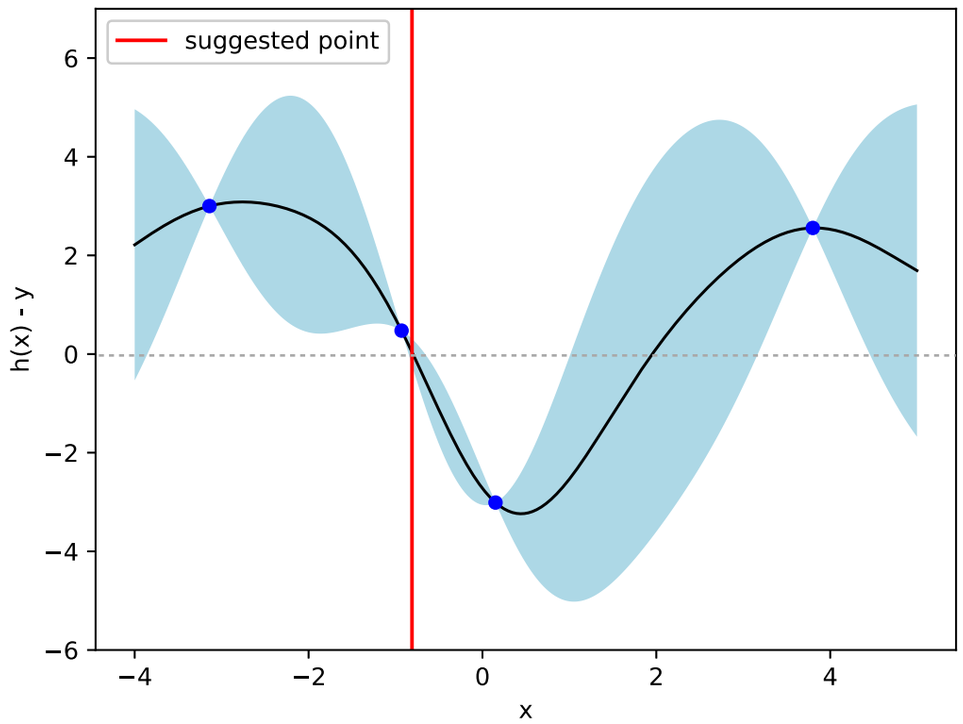
(b) Evaluations of $(h(x) - y)^2$



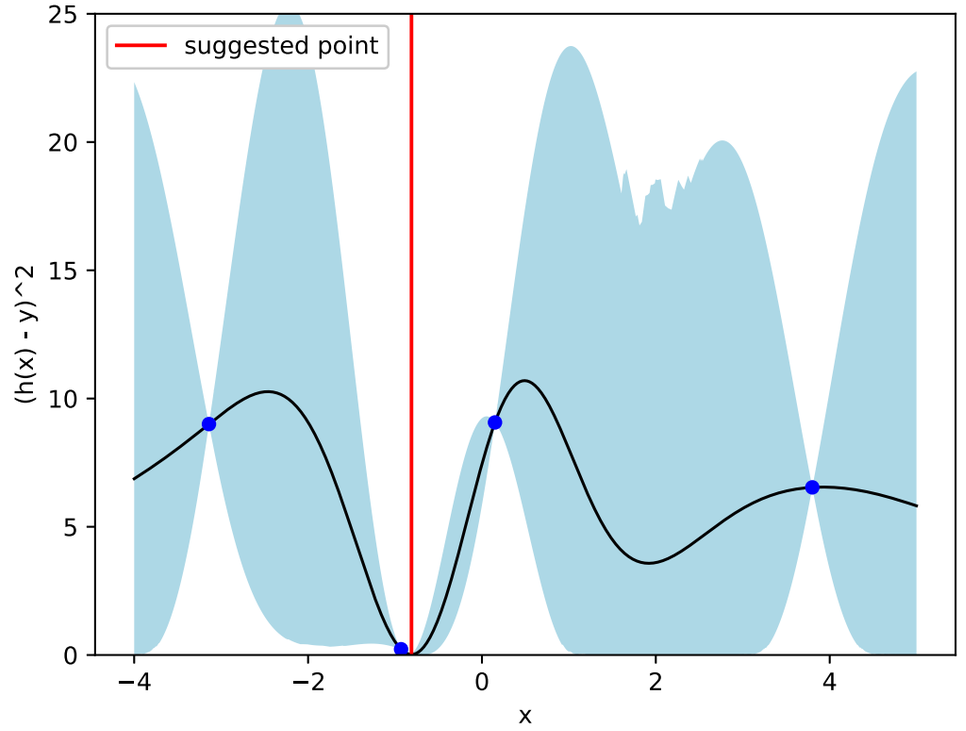
(a) GP posterior on $h(x) - y$



(b) Implied posterior on $(h(x) - y)^2$

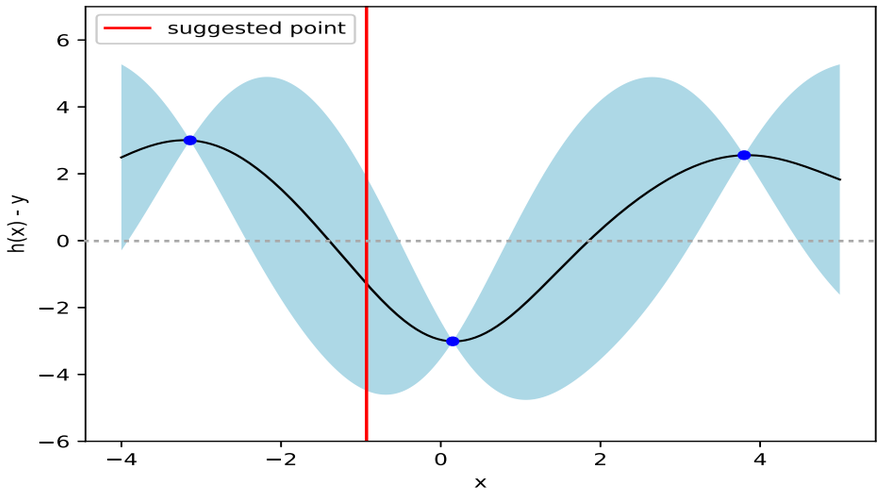


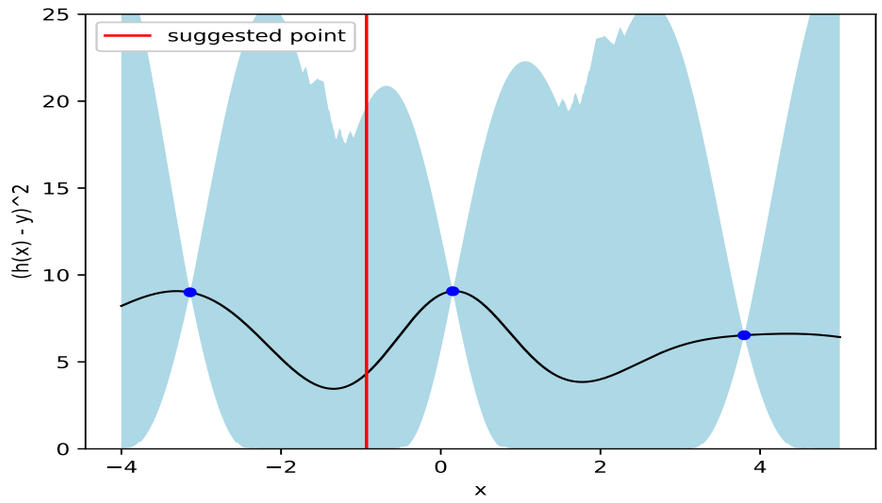
(a) GP posterior on $h(x) - y$

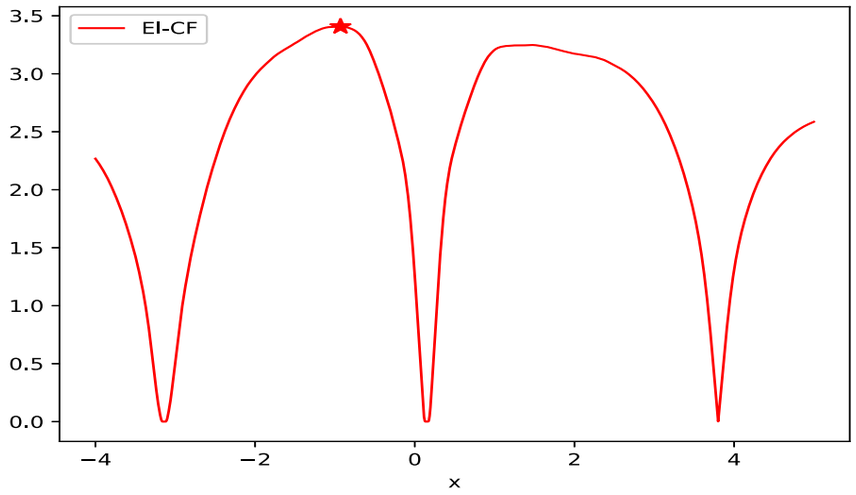


(b) Implied posterior on $(h(x) - y)^2$

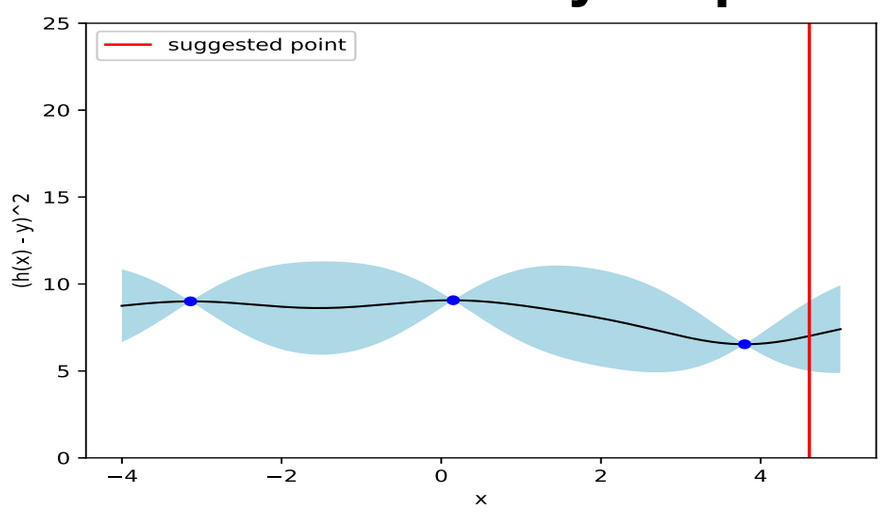
Grey-Box BayesOpt

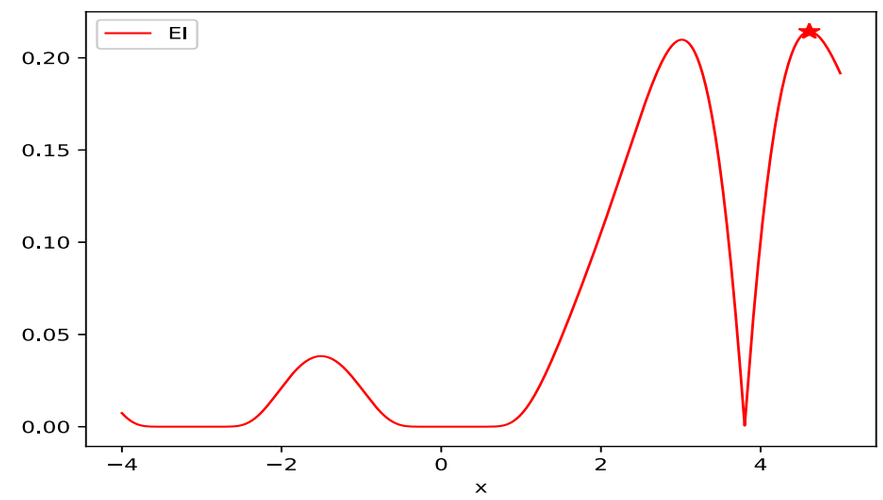






Standard BayesOpt





Implementing Grey-Box BayesOpt for Composite Functions

Challenge: EI for composite functions (EI-CF) is harder to maximize than classical EI

- In standard Bayesian optimization, f(x) is Gaussian, giving EI a closed form.

Challenge:

- When h is a GP and g is nonlinear, f(x) = g(h(x)) is not Gaussian
- EI-CF has **no closed form**, making it hard to optimize

Calculating EI-CF

To estimate EI-CF(x), repeat the following:

1. Sample h(x) from the Gaussian process posterior, $h(x) = \mu(x) + C(x)\, Z$ where $Z \sim N(0, I_m)$
2. Calculate the improvement $\{g(h(x))-f^*\}^+$

Then average the results.

Challenge: a naive approach is slow

Naive approach: Maximize this simulation-based estimate of EI-CF directly, e.g., with a genetic algorithm

Problem: This will be really slow because we lack gradients and evaluations are noisy

A better way to maximize EI-CF: Sample Average Approximation

1. $Z(1), \ldots, Z(K) \sim N(0, I_m)$ via Monte Carlo or quasi-Monte Carlo

2. $\widehat{\text{EI-CF}}(x) = \sum_{k=1}^{K} \{g(\mu(x) + C(x)Z(k)) - h^*\}^+$

3. Optimize $\widehat{\text{EI-CF}}(x)$ using a deterministic optimization solver

Balandat et al. 2020,
"BOTORCH: A Framework for Efficient Monte-Carlo Bayesian Optimization"

What we know about solution quality

Theorem: If g is continuous, and other regularity conditions hold, EI-CF is **asymptotically consistent**, i.e., it finds the true global optimum as the number of evaluations grows to infinity.

Remark: By construction, evaluating at argmax_x EI-CF(x) is **optimal** (in an average-case sense under the prior on h) if:

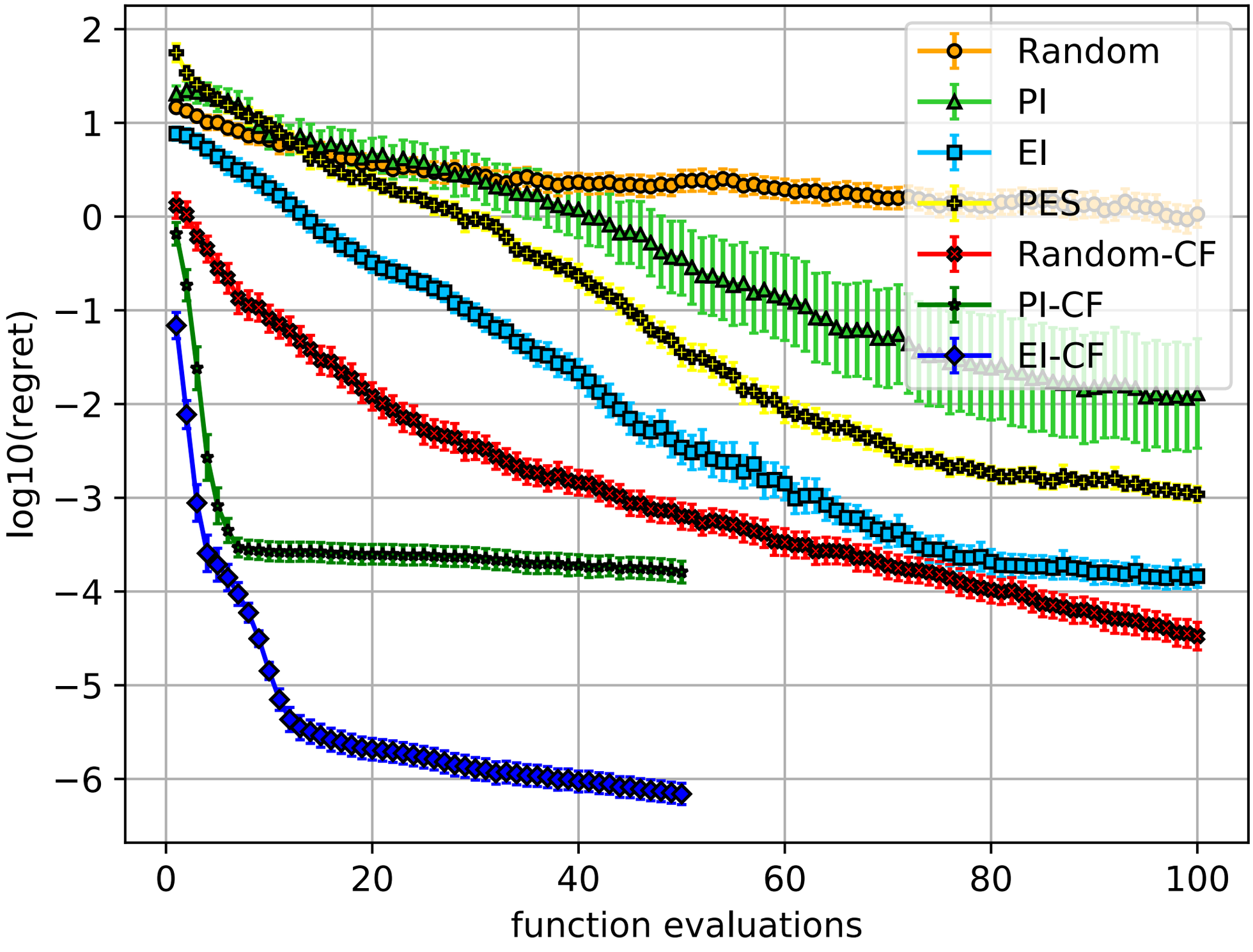
1. this is our last evaluation
2. our solution can only be a previously evaluated point

Environmental model test problem (Bliznyuk et al., 2008)

- Models a chemical accident that has caused a pollutant to spill at two locations.
- Given 12 measurements at different geospatial locations, invert the 4 parameters of this simulator.
- We solve

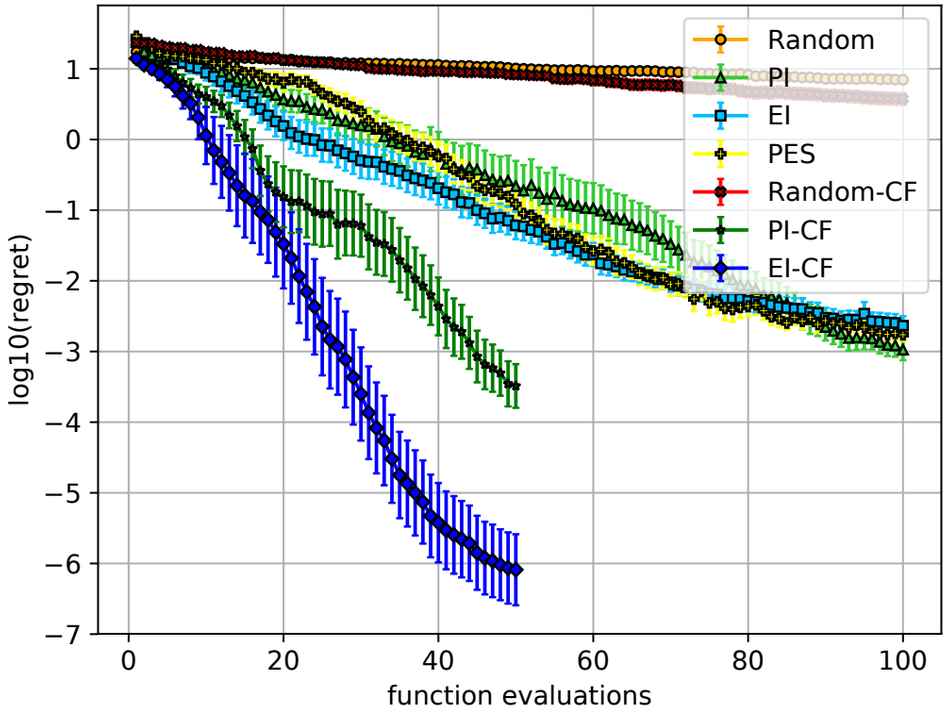
$$\min_{x \in \mathcal{X}} \sum_{j=1}^{12} (s(\theta_j; x^*) - s(\theta_j; x))^2.$$

Environmental model test problem

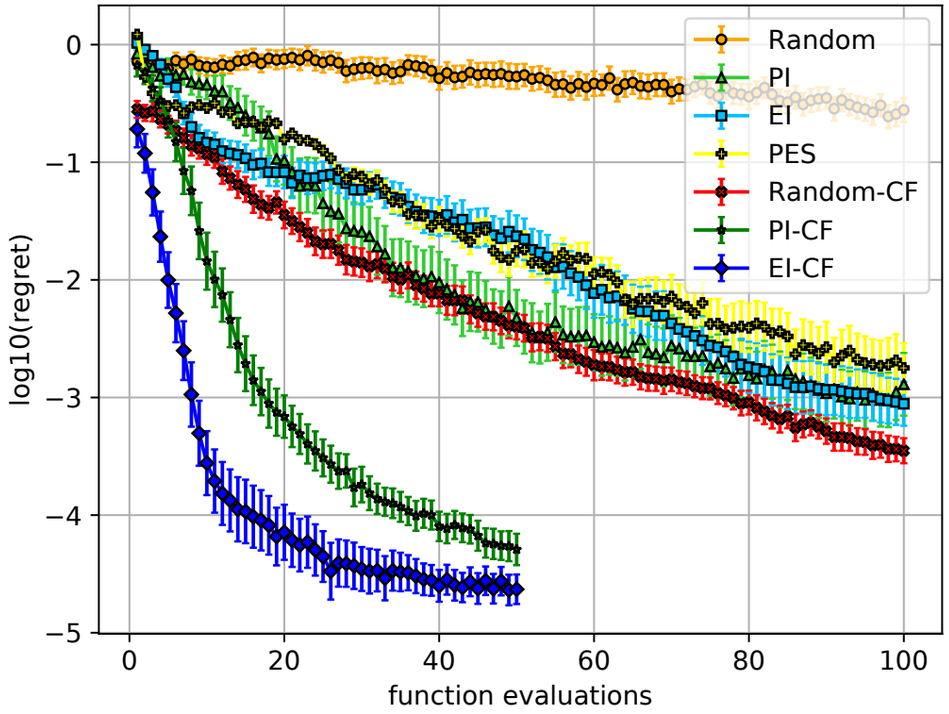


GP-generated test problems

Problem	$\mathcal{X}$	g	m
a	$[0,1]^4$	$g(h(x)) = -\sum_{j=1}^{5}(h_j(x) - y_j^*)^2$	5
b	$[0,1]^3$	$g(h(x)) = -\sum_{j=1}^{4}\exp(h_j(x))$	4



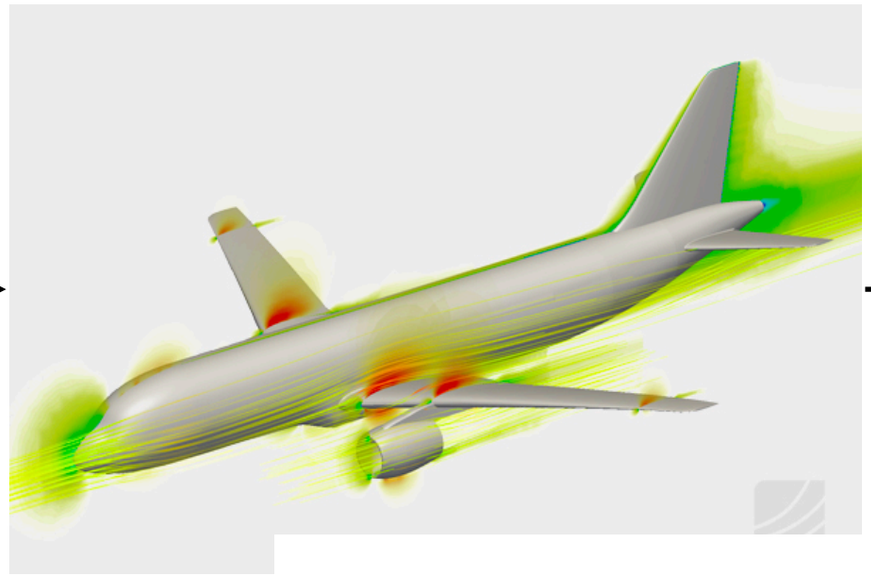
(a)



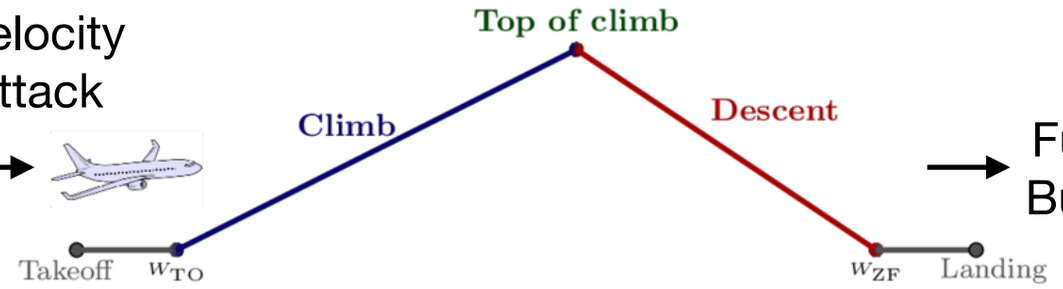
(b)

These ideas extend to compositions where **both** functions are black boxes

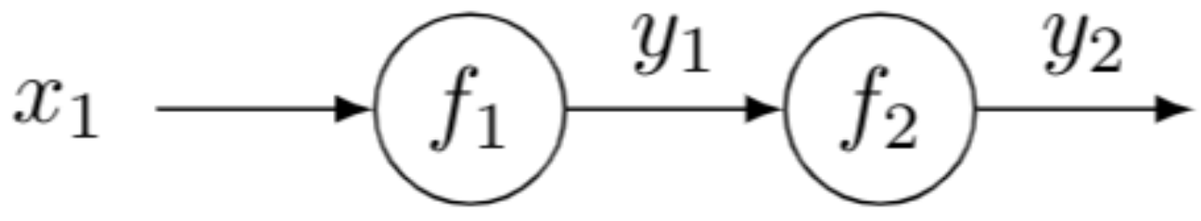
Wing Design →



Lift & Drag as a function of velocity & angle of attack →

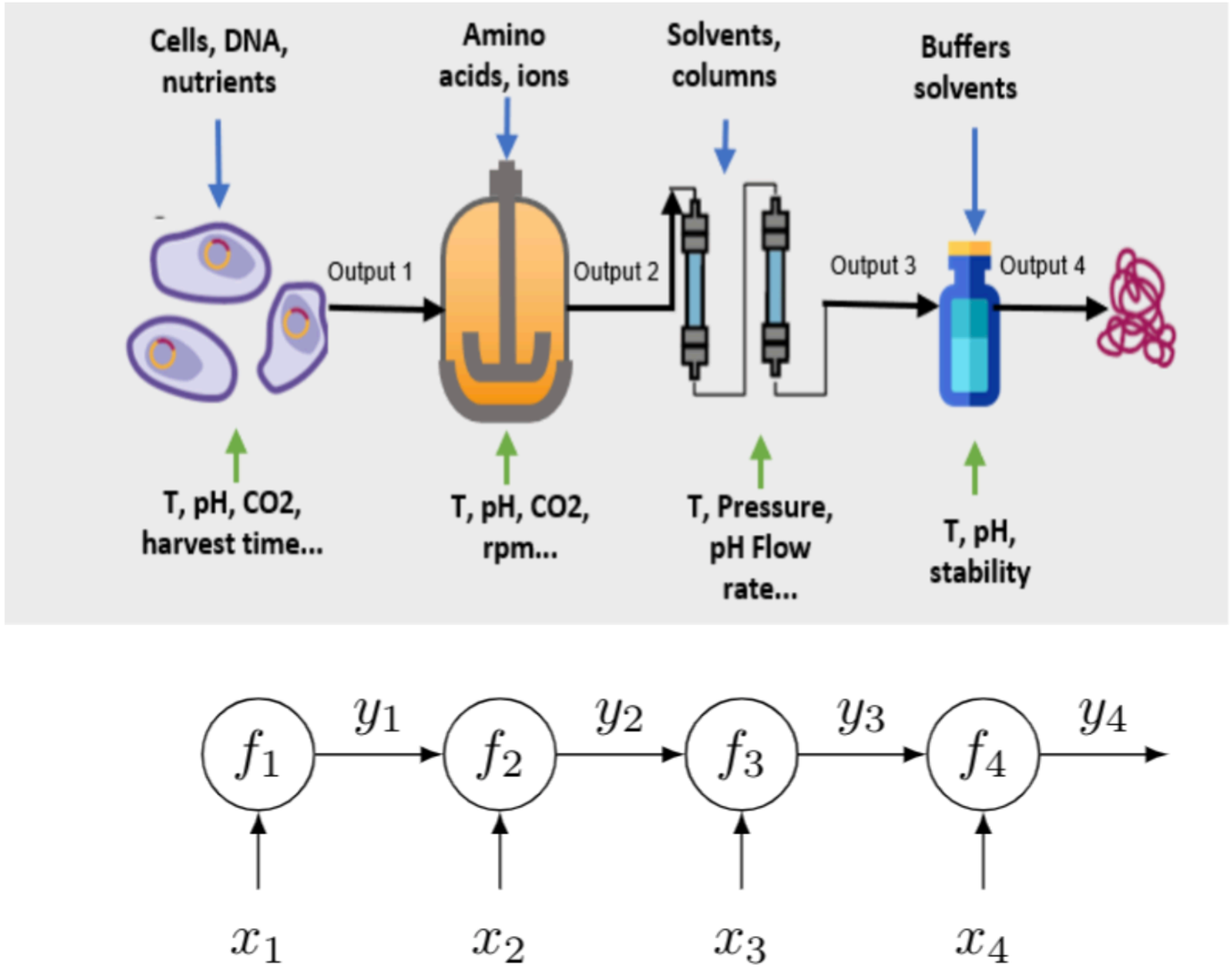


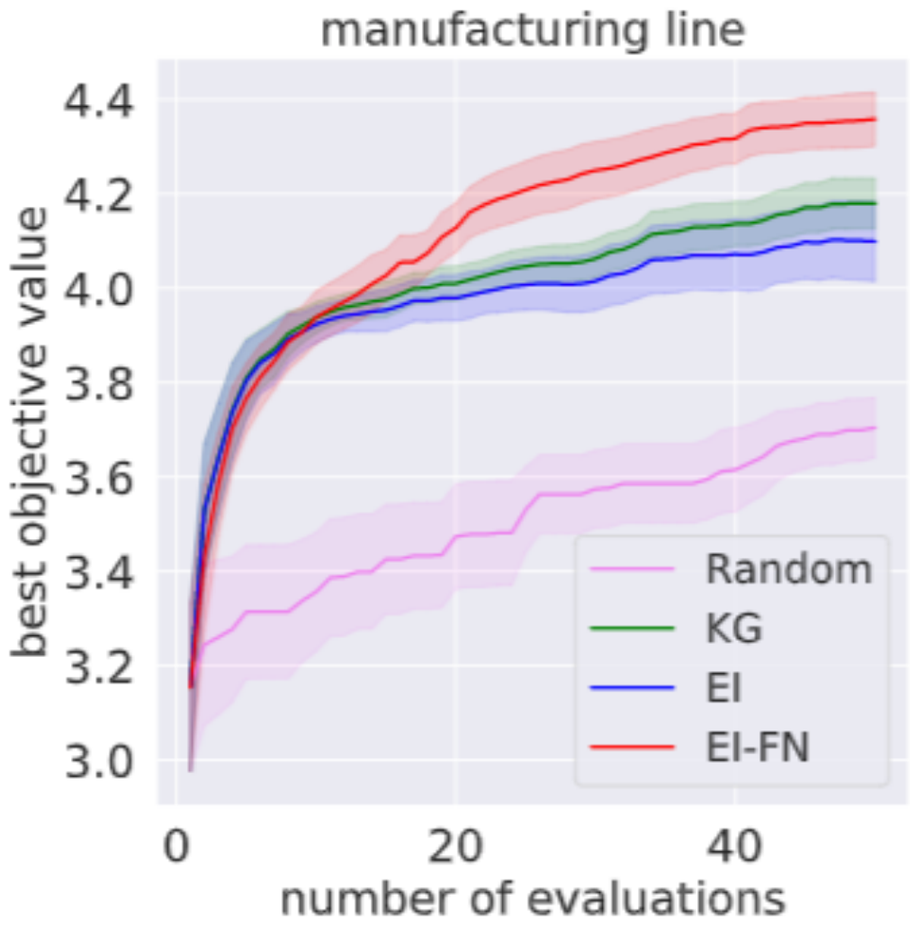
→ Fuel Burn



Astudillo & Frazier, “Bayesian Optimization of Function Networks”, NeurIPS 2021

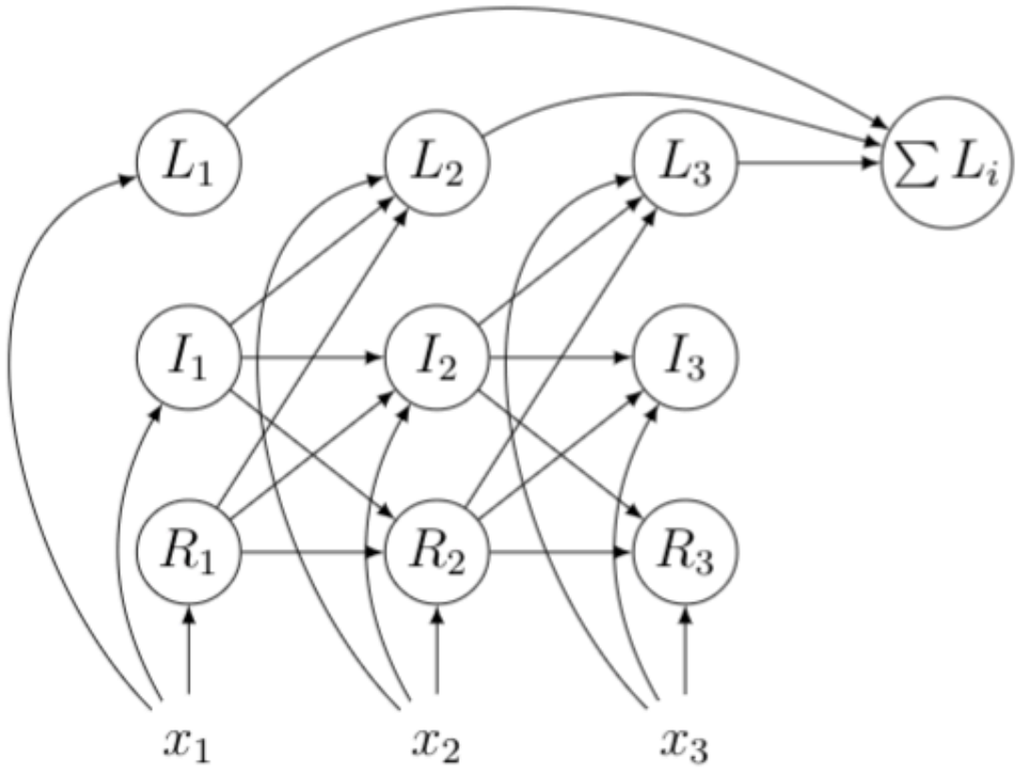
These ideas extend to function networks

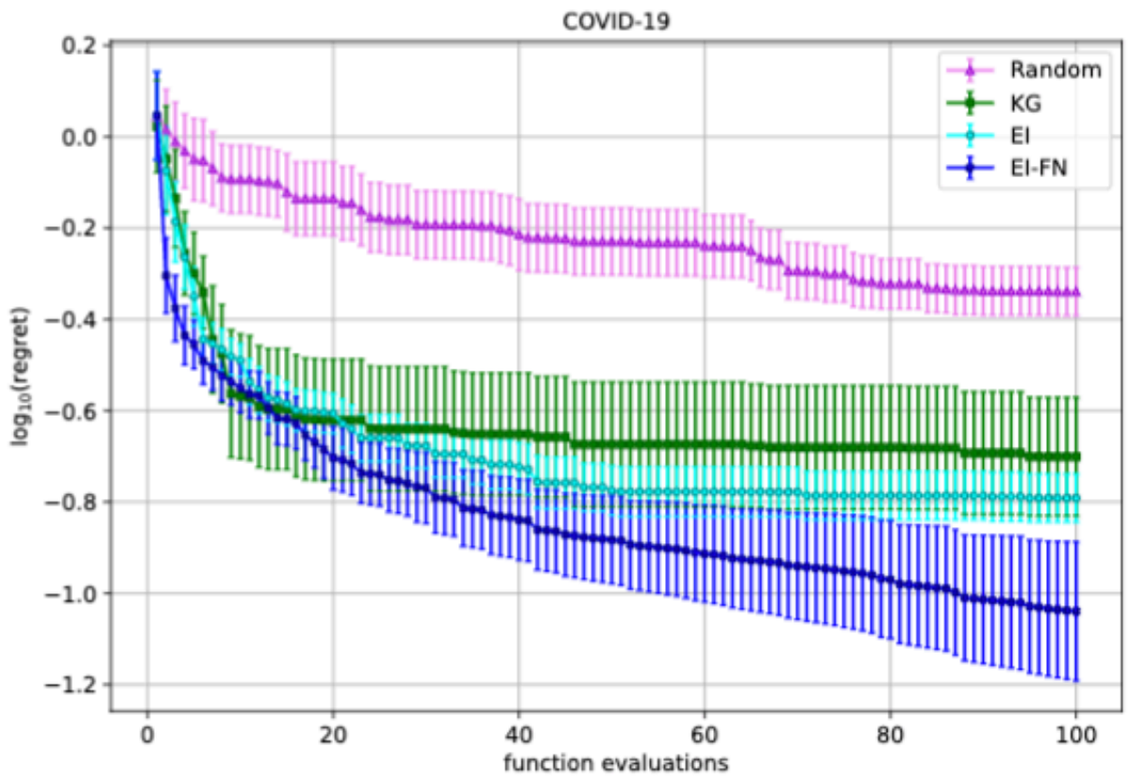




Astudillo & Frazier, “Bayesian Optimization of Function Networks”, NeurIPS 2021

These ideas extend to function networks





Astudillo & Frazier, “Bayesian Optimization of Function Networks”, NeurIPS 2021

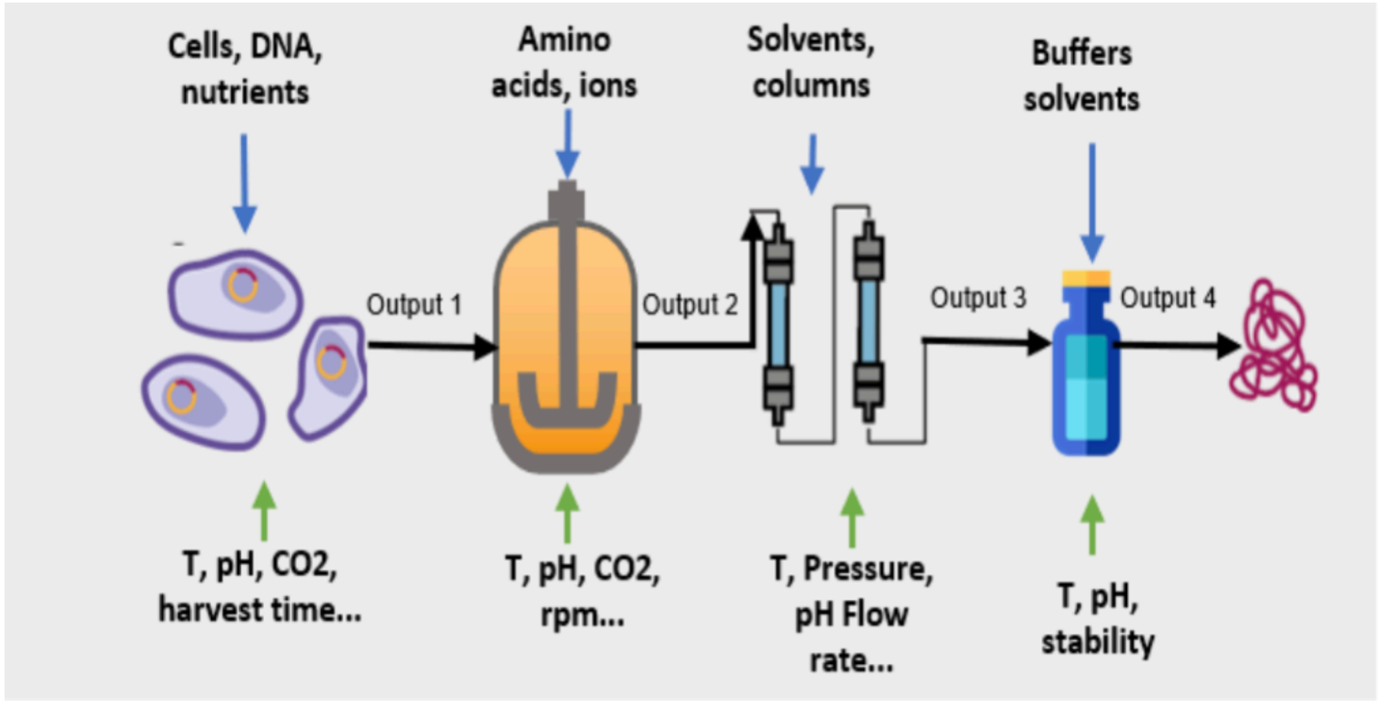
Theorem (Astudillo & Frazier 2021).

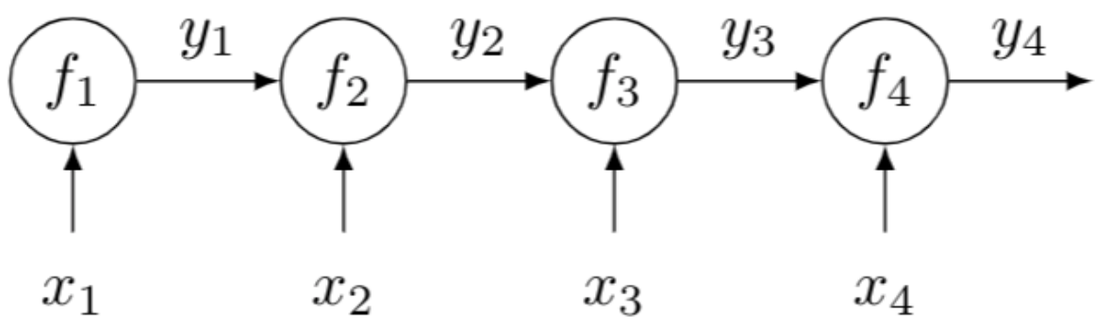
Grey-box EI is asymptotically consistent despite not necessarily sampling $\mathbb{X}$ densely.

Why evaluate the whole function network?

Sometimes, we can **quickly** evaluate a **few** of the objective function's constituents

Williams et al. 2000
Swersky, Snoek & Adams 2013
Marzat et al. 2013
Bogunovic et al. 2018
Toscano-Palmerin & F. 2021
Cakmak, Astudillo & F. 2020
Marque-Pucheu, Perrin & Garnier 2020
Nguyen et al. 2021

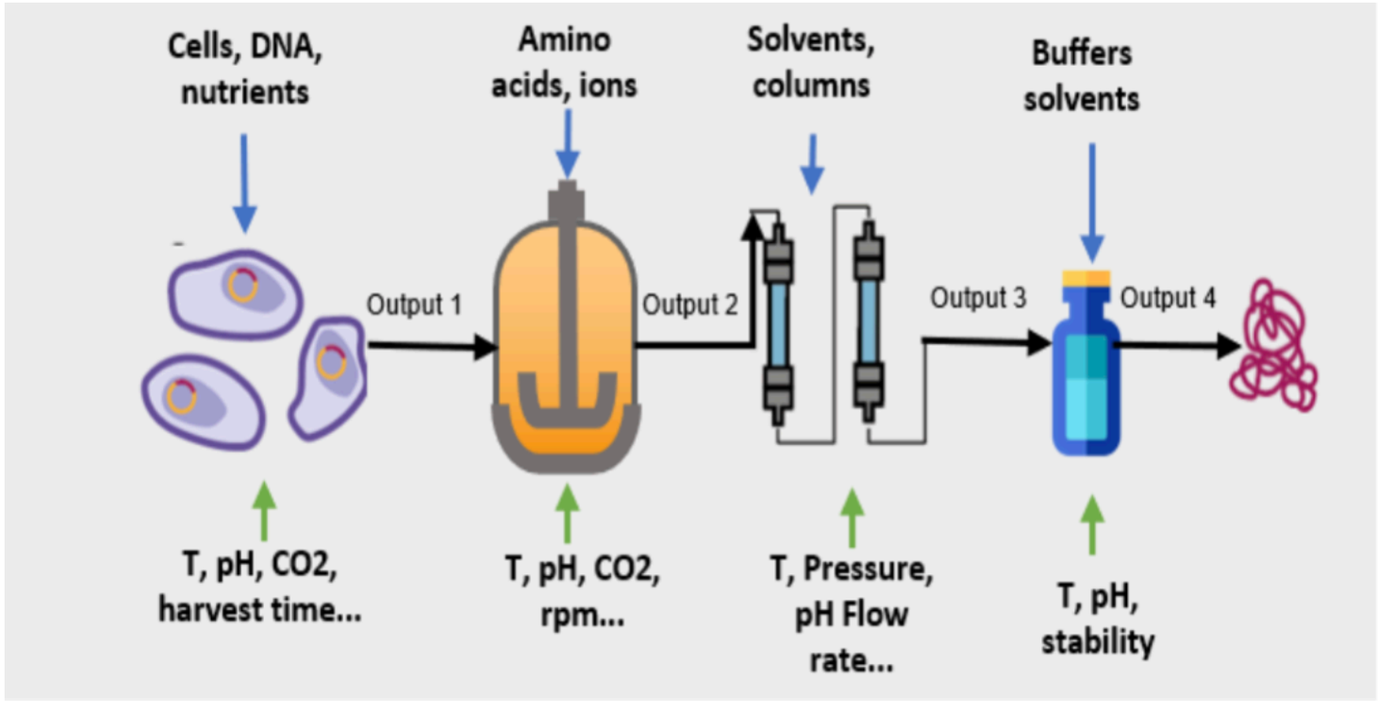


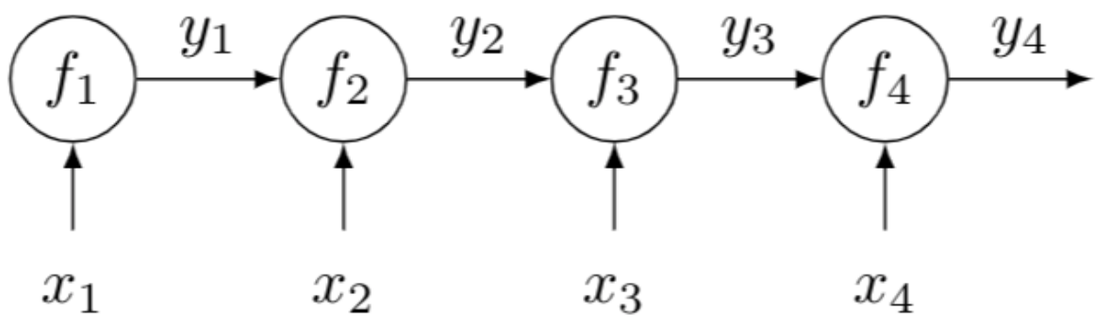


Objective Constituent Evaluations

Sometimes, we can **quickly** evaluate a **few** of the objective function's constituents

Williams et al. 2000
Swersky, Snoek & Adams 2013
Marzat et al. 2013
Bogunovic et al. 2018
Toscano-Palmerin & F. 2021
Cakmak, Astudillo & F. 2020
Marque-Pucheu, Perrin & Garnier 2020
Nguyen et al. 2021

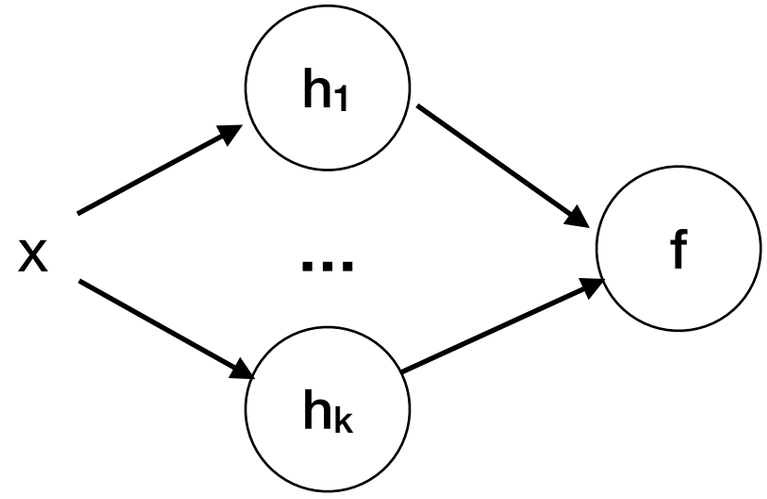




Objective Constituent Evaluations

Linear objectives were studied first
[Swersky, Snoek & Adams 2013; Toscano-Palmerin & F. 2021]

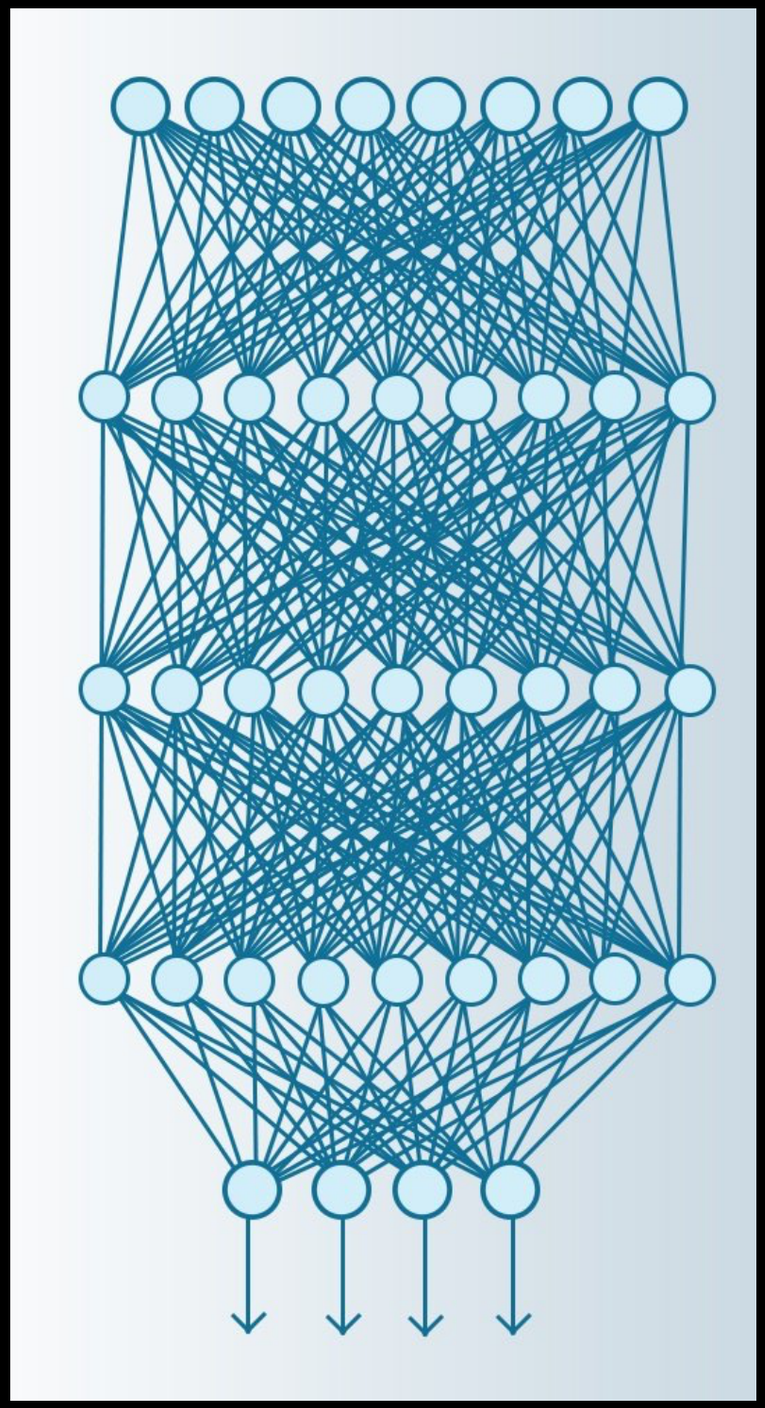
$$f(x) = \sum_{j=1}^{k} h_j(x),$$



where:

- each function h_j is a time-consuming-to-evaluate black box
- evaluations of h_1, .., h_k can be performed separately

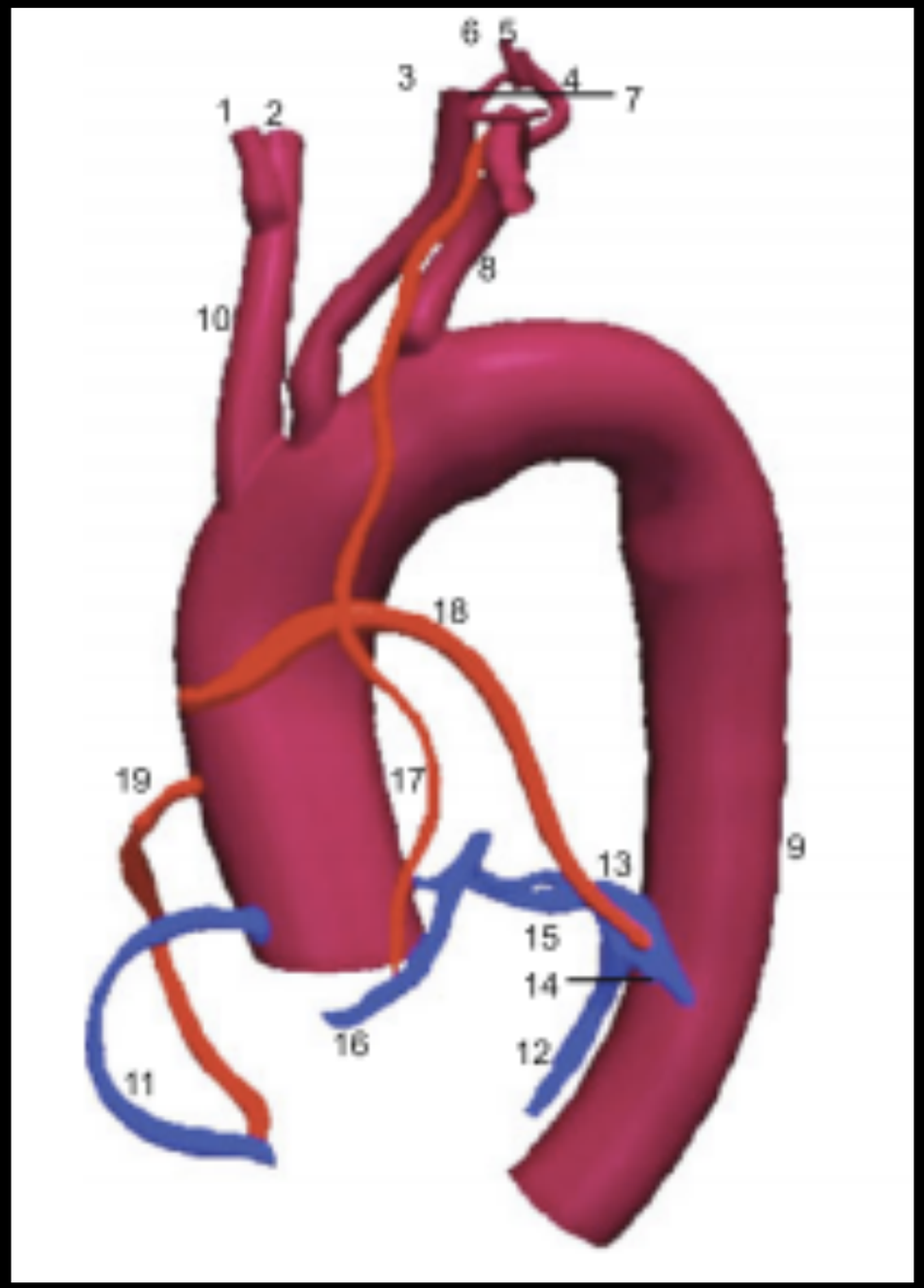
Application: AutoML with Cross-Validation



$$\min_{x \in A \subset \mathbb{R}^a} \frac{1}{n} \sum F(x, w)$$

- x is a set of hyperparameters for a deep neural network
- w indexes folds (held out data) in cross-validation
- F(x,w) is -(training error) on fold w
- We want to minimize cross-validation error

Application: Engineering Design



$$\min_{x \in A \subset \mathbb{R}^d} \int F(x, w)\, p(dw)$$

- x is the design of a cardiovascular bypass graft
- w is a vector of time-varying environmental conditions (e.g., velocity of the blood entering the graft)
- F(x,w) is a performance measure
- We want to maximize average performance

Key Idea: Evaluating individual summands is more efficient than evaluating the whole sum

- Evaluating only one or a small subset of the summands is cheaper and can be highly informative.
- Standard BO requires evaluating $h_1(x),\ldots,h_K(x)$ every iteration.

How do we leverage evaluation of individual summands?

- Model $h = (h_1,...,h_k)$ instead of f alone.
- Use an acquisition function that allows us to quantify the value of evaluating an individual summand hj(x).

Predictive Model

- As before, we model h using a multi-output GP.
- Leveraging correlation between outputs is key here.
- Popular choices to model correlation between outputs include *intrinsic and linear correlation* models.
- The latter, for example, assumes that $h_j(x) = \sum_{i=1}^{L} a_{i,j} h'_i(x)$, where $h'_1, \ldots, h'_L$ are independent.

Challenge: Quantifying the benefit of evaluating $h_j(x)$

- Since EI is defined in terms of the improvement in objective value, it doesn't naturally extend to quantify the benefit of evaluating a single summand hj(x).

- Similarly for the probability of improvement and UCB.

- Other acquisition functions such as knowledge gradient (KG) do have natural extensions

Knowledge gradient (KG) for constituent evaluations

- KG can be easily extended to handle evaluations of a single summand:

$$\mathrm{KG}_n(x,j) = \mathbb{E}_n\left[\mu^*_{n+1} - \mu^*_n \mid (x_{n+1}, j_{n+1}) = (x,j)\right],$$

where

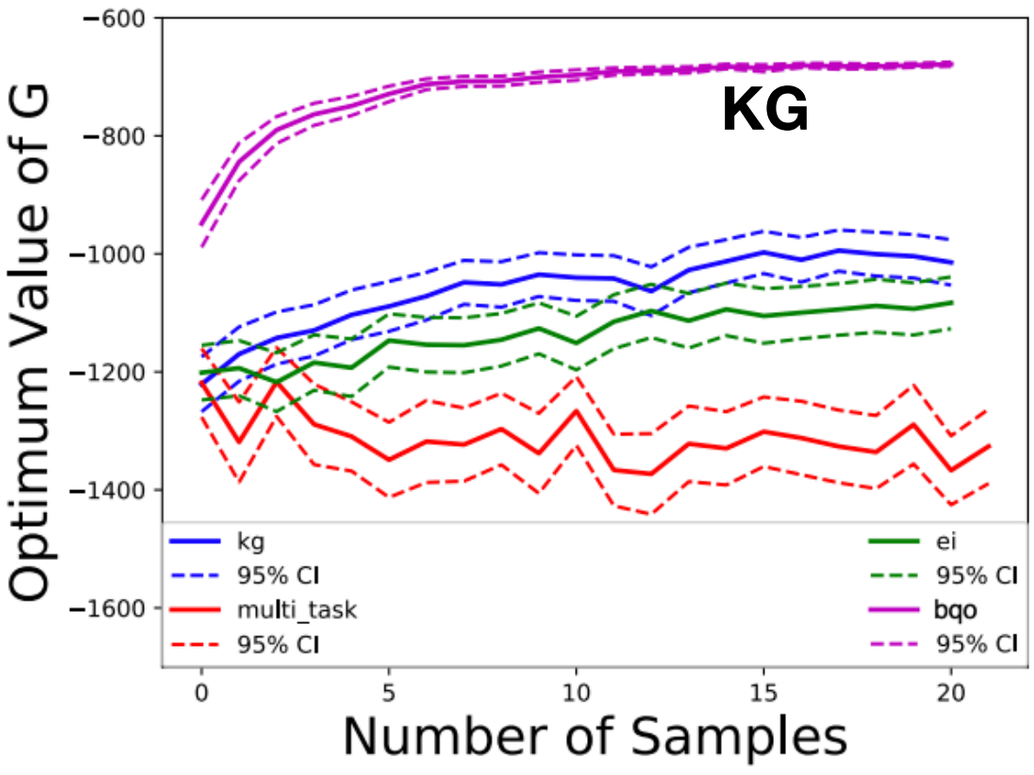
$$\mu^*_m = \max_{x\in\mathbb{X}} \mathbb{E}_m\left[\sum_{j=1}^{k} h_j(x)\right] = \max_{x\in\mathbb{X}} \sum_{j=1}^{k} \mu_{j,m}(x),$$

for $m = n, n+1$.

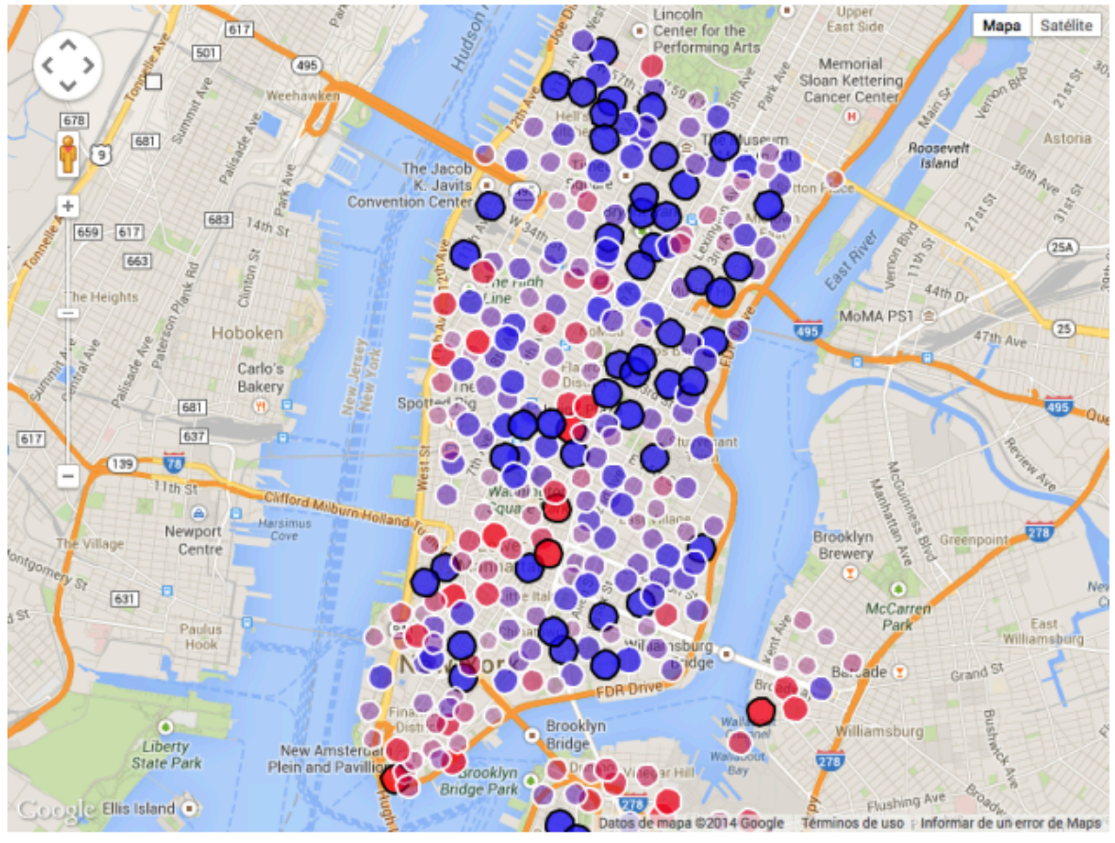
- Thus, KG tells us not only the x but also the j at which to evaluate $h_j(x)$.

Toscano-Palmerin & F., SIAM J. Optimization 2021

This approach can be much faster



(a) Performance comparison between BQO and two Bayesian optimization benchmark, the KG and EI methods, on the Citi Bike Problem from §6.3

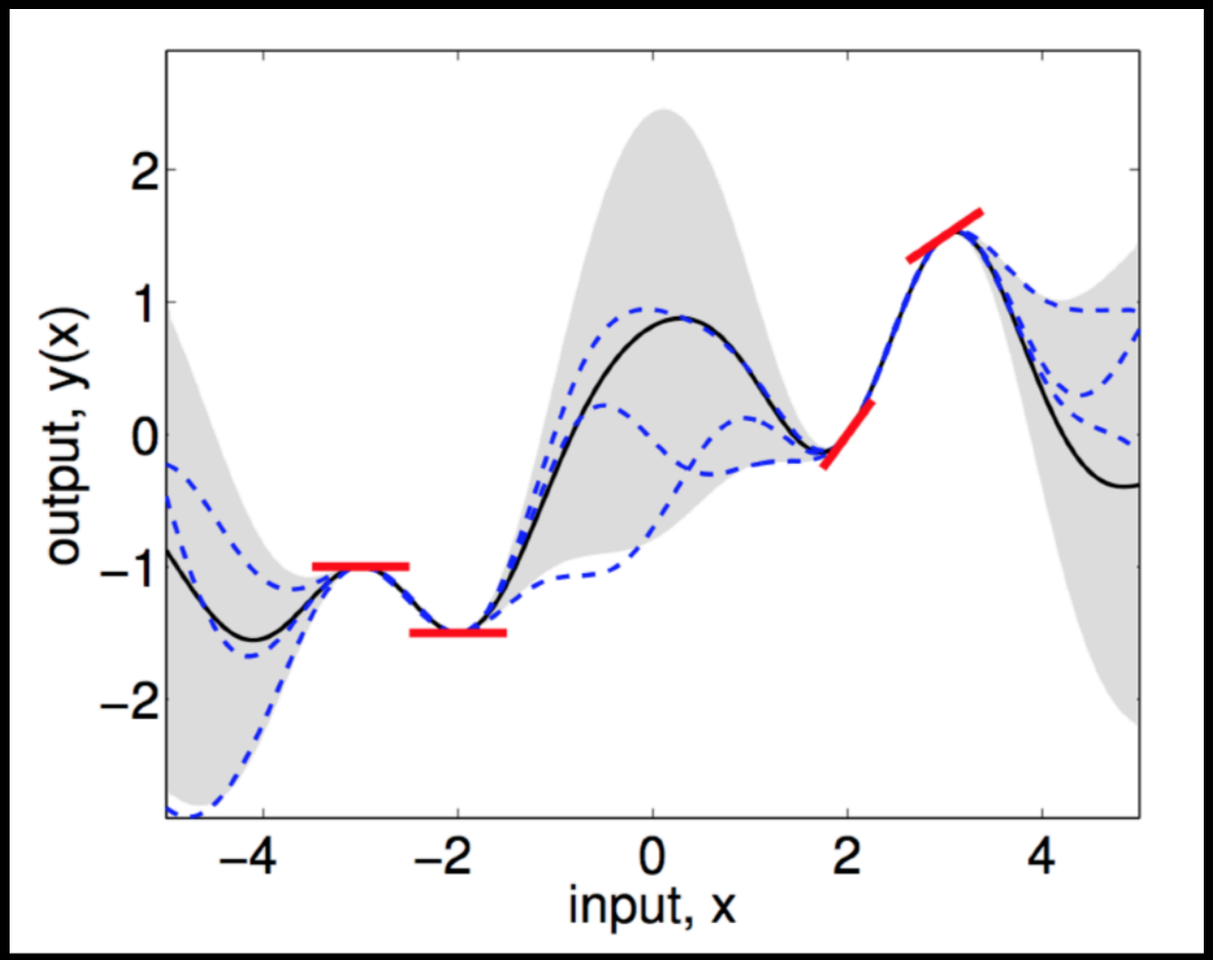


(b) Location of bike stations (circles) in New York City, where size and color represent the ratio of available bikes to available docks.

Toscano-Palmerin & F., SIAM J. Optimization 2021

There’s lots to see inside the box

Grey-Box Example:
Bayesian optimization with derivatives



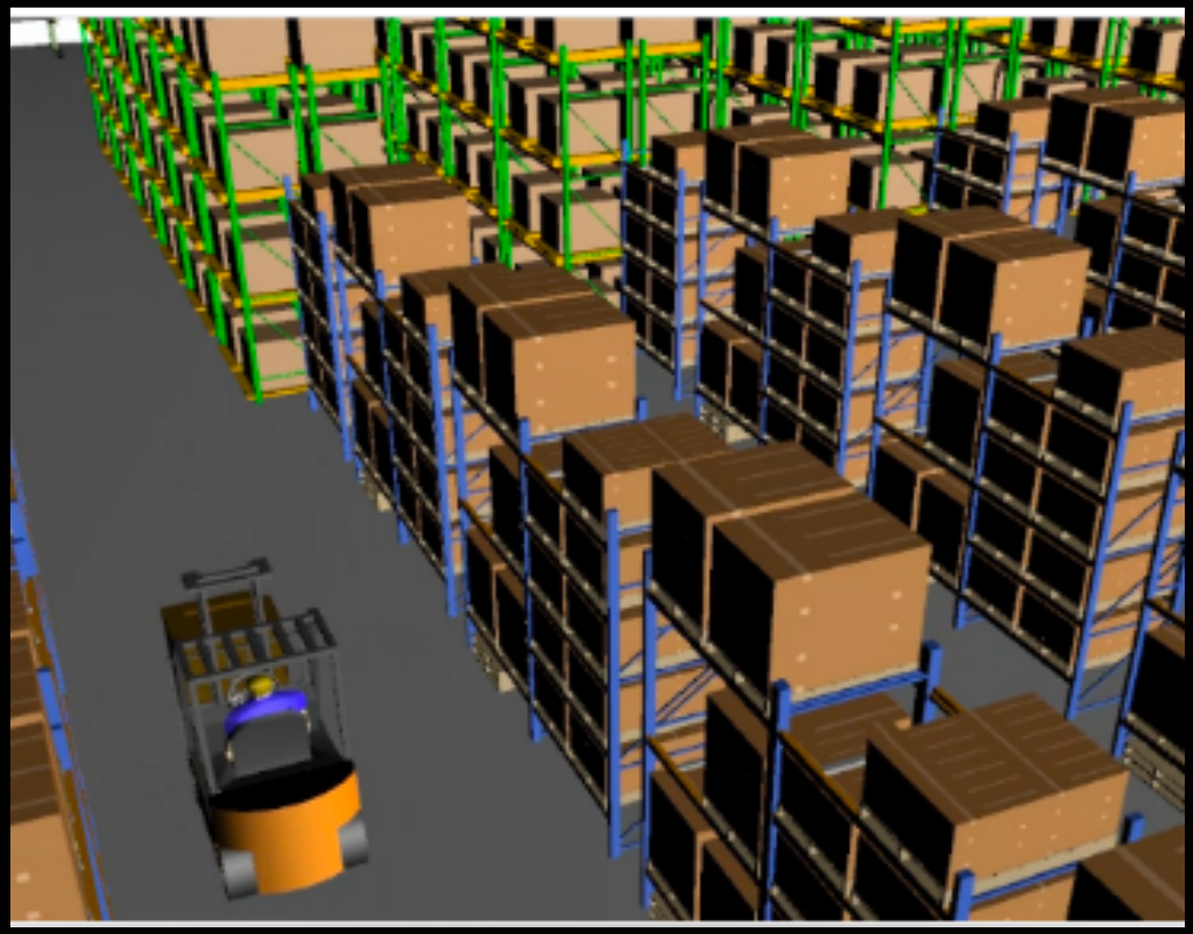
Evaluating at x provides
F(x) **and ∇F(x)**

Bayesian optimization with gradients
J Wu, M Poloczek, AG Wilson, P Frazier, NeurIPS 2017

Grey-Box Example:
Multi-information source optimization

Information Source: A/B test
Pro: Accurate
Con: Slow

Information Source: Simulation
Pro: Fast
Con: Biased



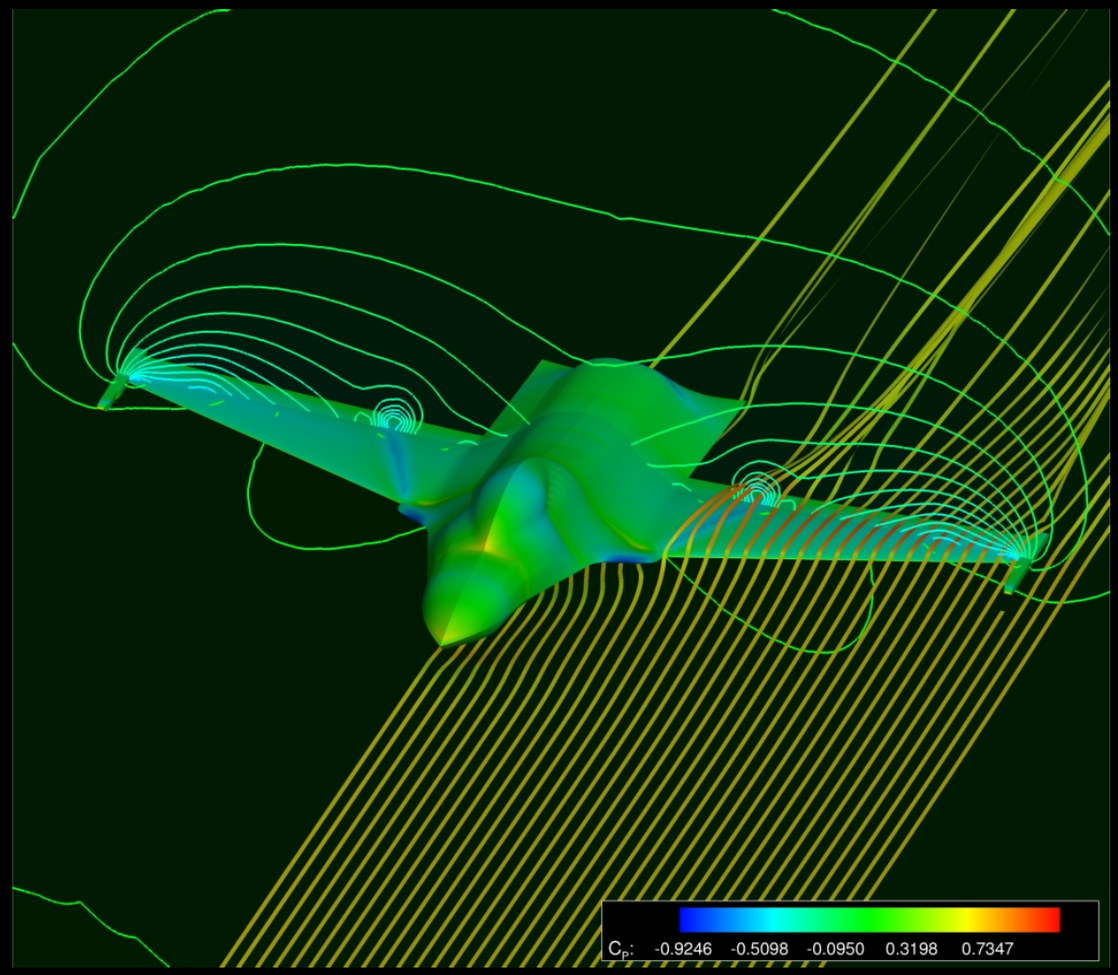
Multi-information source optimization
M Poloczek, J Wang, P Frazier, NeurIPS 2017

Grey-Box Example: Multi-information source optimization

Information Source: Wind Tunnel
Pro: Accurate
Con: Slow

Source: nasa.gov

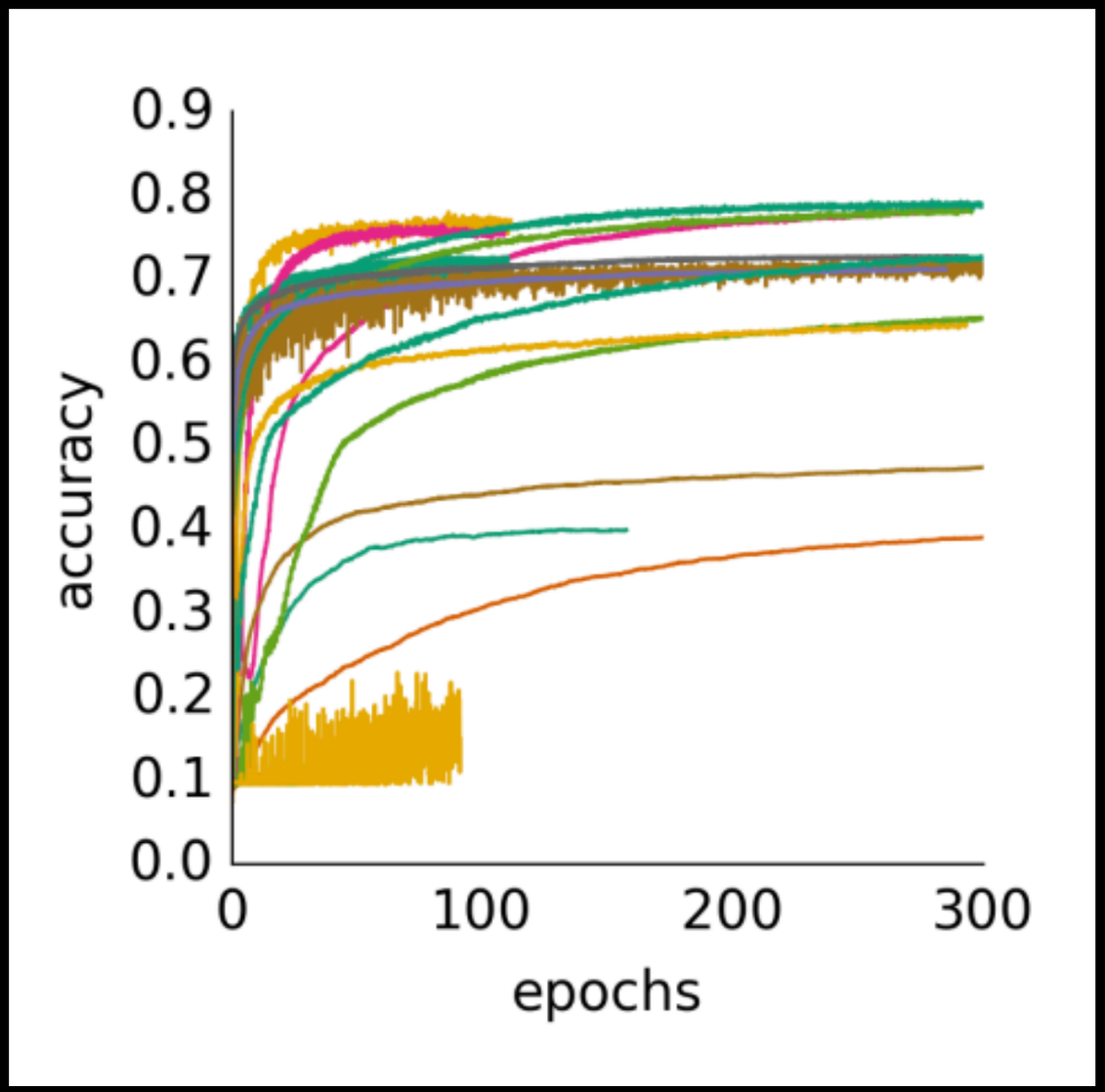
Information Source: Simulation
Pro: Fast
Con: Biased



Source: cfd4aircraft.org

Multi-information source optimization
M Poloczek, J Wang, P Frazier, NeurIPS 2017

Grey-Box Example: AutoML with trace observations



- We observe test error for all epochs, not just the final one
- We can stop early
- We can restart stopped training
- We can run with less training data or less validation data

Practical Multi-fidelity Bayesian Optimization for Hyperparameter Tuning
J Wu, S Toscano-Palmerin, PI Frazier, AG Wilson, UAI 2019

Conclusion

- Lots of headroom in looking inside the box
- Implementation of BO for composites function available in BoTorch (botorch.org)
- "Thinking inside the box: A tutorial on grey-box Bayesian optimization", with Raul Astudillo, https://arxiv.org/abs/2201.00272
- Raul is looking for postdocs, https://raulastudillo.netlify.app/

Thank You!