

Parallel Bayesian Optimization, for Metrics Optimization at Yelp

Peter Frazier

Assistant Professor

Operations Research & Information Engineering

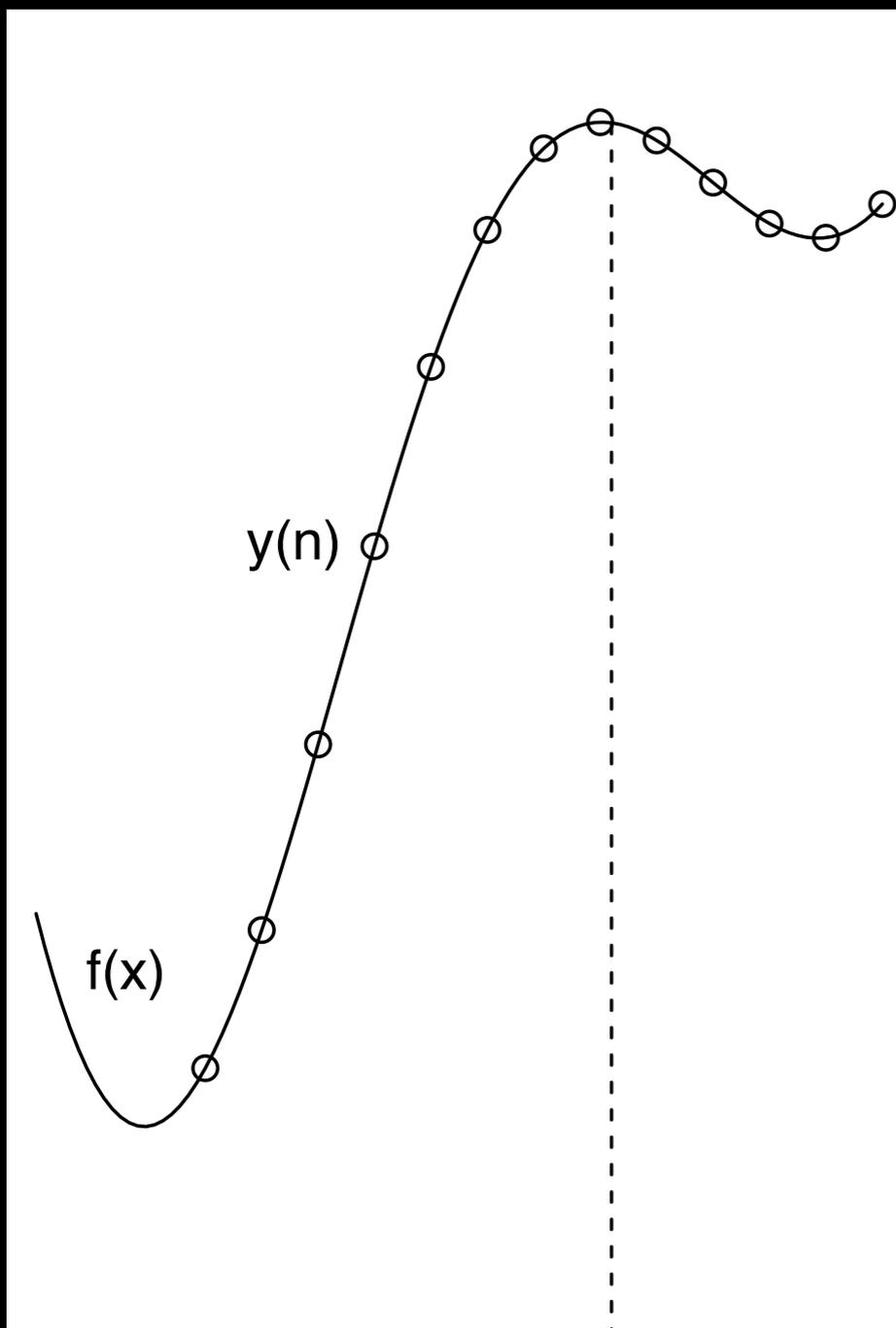
Cornell University

Joint work with:

Jialei Wang (Cornell), Scott Clark (Yelp, SigOpt), Eric Liu (Yelp, SigOpt),

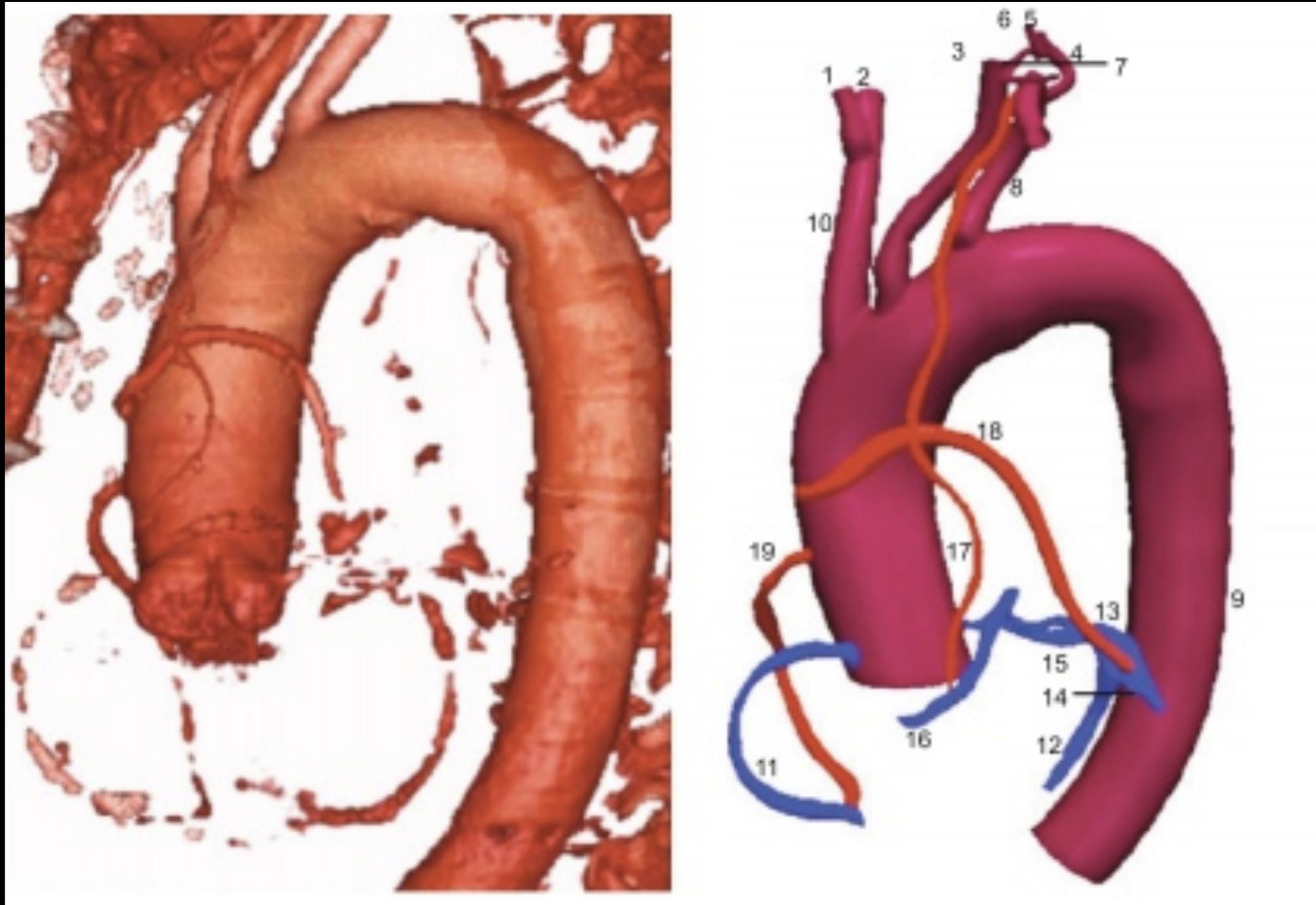
Deniz Oktay (Yelp, MIT), Norases Vesdapunt (Yelp, Stanford)

Consider optimizing an “expensive” function.

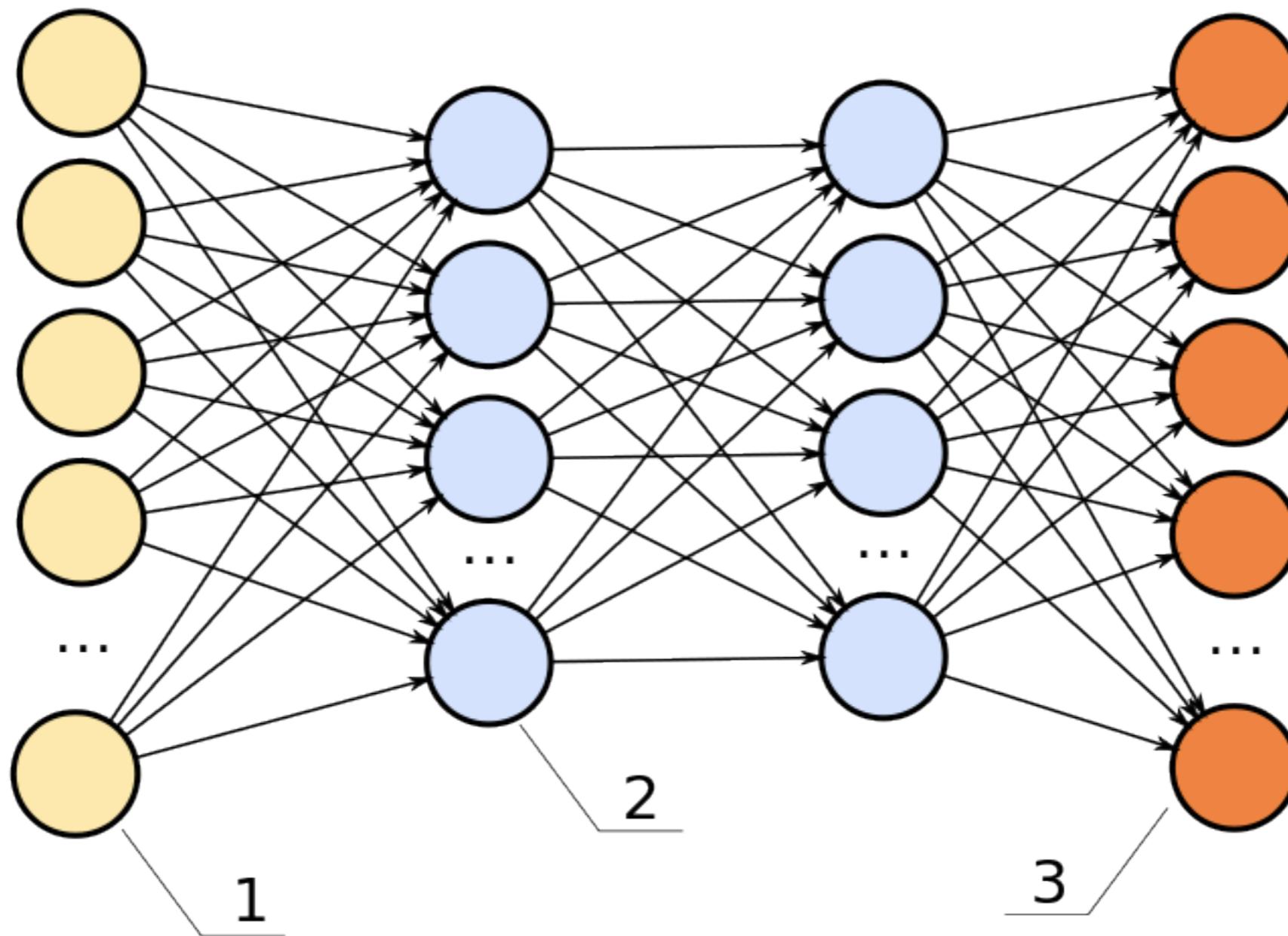


- We'd like to optimize an objective function, $f : \mathbb{R}^d \rightarrow \mathbb{R}$.
- f 's feasible set is simple, e.g., box constraints.
- f is continuous but lacks special structure, e.g., concavity, that would make it easy to optimize.
- f is derivative-free: evaluations do not give gradient information.
- f is “expensive” to evaluate --- the # of times we can evaluate it is severely limited.

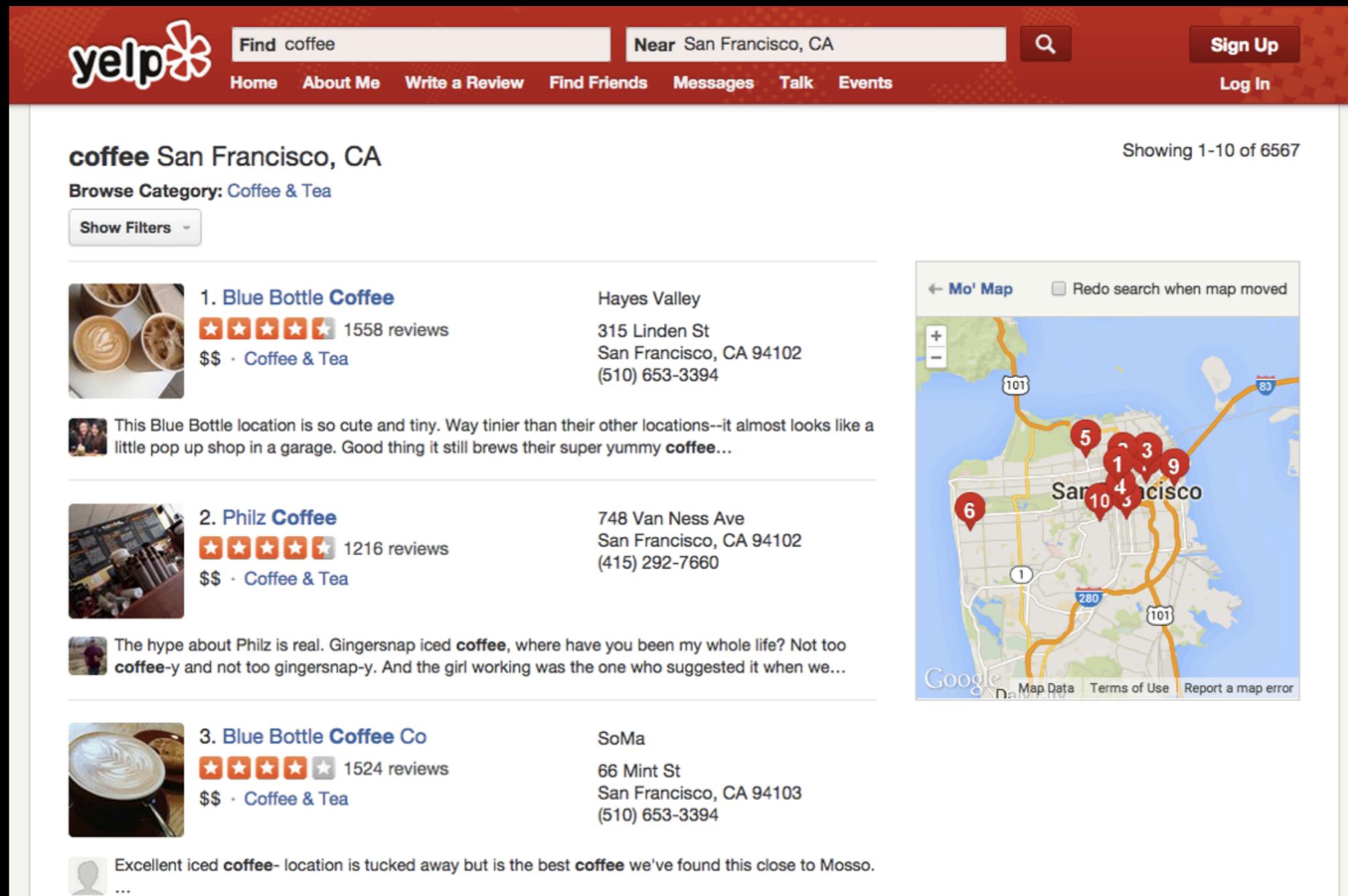
Optimization of expensive functions arises when optimizing physics-based models



Optimization of expensive functions arises when fitting machine learning models



Optimization of expensive functions arises when tuning websites with A/B testing



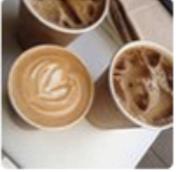
yelp Find coffee Near San Francisco, CA Sign Up Log In

Home About Me Write a Review Find Friends Messages Talk Events

coffee San Francisco, CA Showing 1-10 of 6567

Browse Category: [Coffee & Tea](#)

Show Filters

- **1. Blue Bottle Coffee**
Hayes Valley
315 Linden St
San Francisco, CA 94102
(510) 653-3394
★★★★★ 1558 reviews
\$\$ · Coffee & Tea

This Blue Bottle location is so cute and tiny. Way tinier than their other locations--it almost looks like a little pop up shop in a garage. Good thing it still brews their super yummy **coffee**...
- **2. Philz Coffee**
748 Van Ness Ave
San Francisco, CA 94102
(415) 292-7660
★★★★★ 1216 reviews
\$\$ · Coffee & Tea

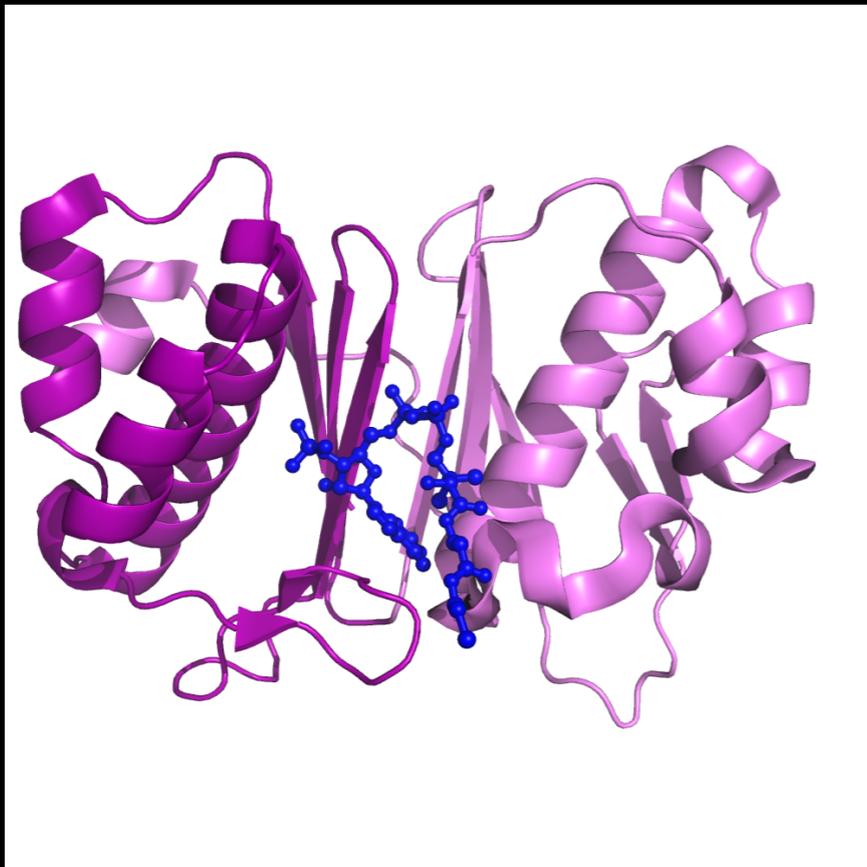
The hype about Philz is real. Gingersnap iced **coffee**, where have you been my whole life? Not too **coffee**-y and not too gingersnap-y. And the girl working was the one who suggested it when we...
- **3. Blue Bottle Coffee Co**
SoMa
66 Mint St
San Francisco, CA 94103
(510) 653-3394
★★★★★ 1524 reviews
\$\$ · Coffee & Tea

Excellent iced **coffee**- location is tucked away but is the best **coffee** we've found this close to Mosso.

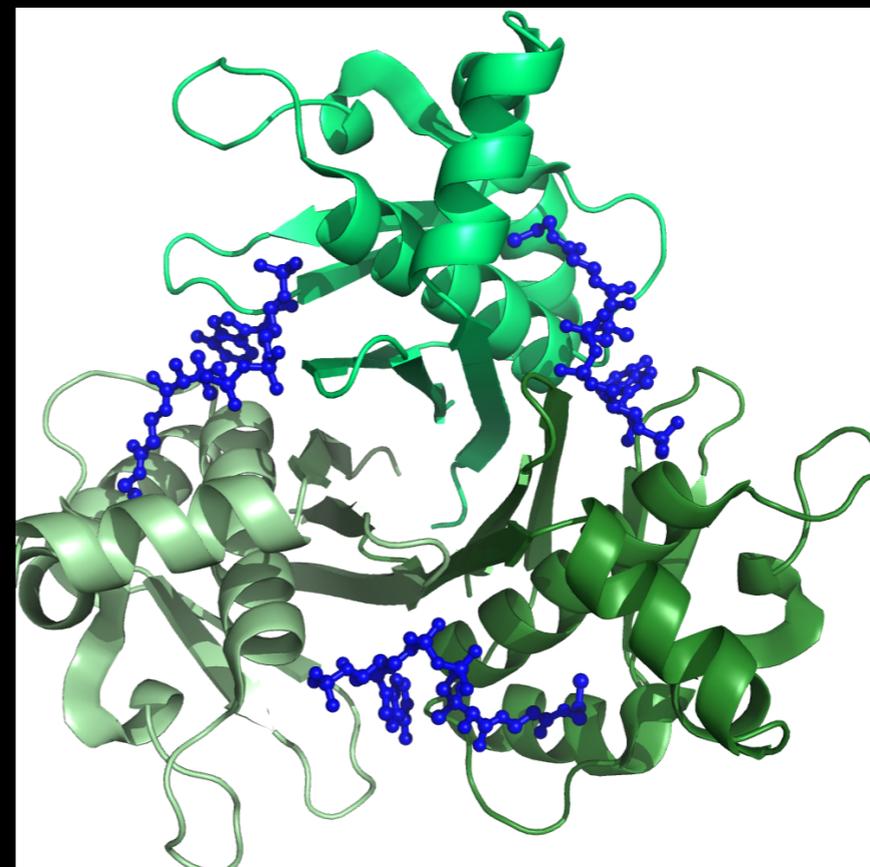
Mo' Map Redo search when map moved

Google Maps Map Data Terms of Use Report a map error

Optimization of expensive functions arises in drug and materials discovery



Sfp
(a protein-modifying enzyme)



AcpS
(another protein-modifying enzyme)

Bayesian Optimization looks like this

Elicit a prior distribution on the function f
(typically a Gaussian process prior).

while (budget is not exhausted) {

 Find the point to sample whose value of
 information is the largest.

 Sample that point.

 Update the posterior distribution.

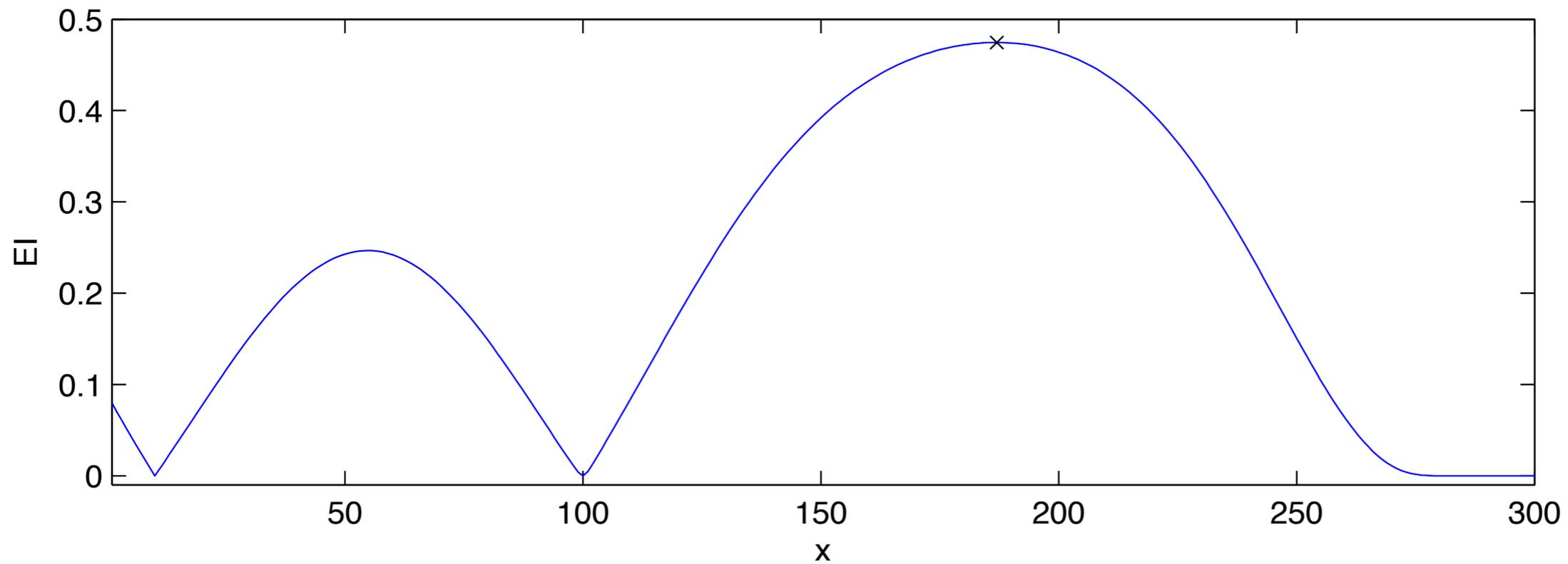
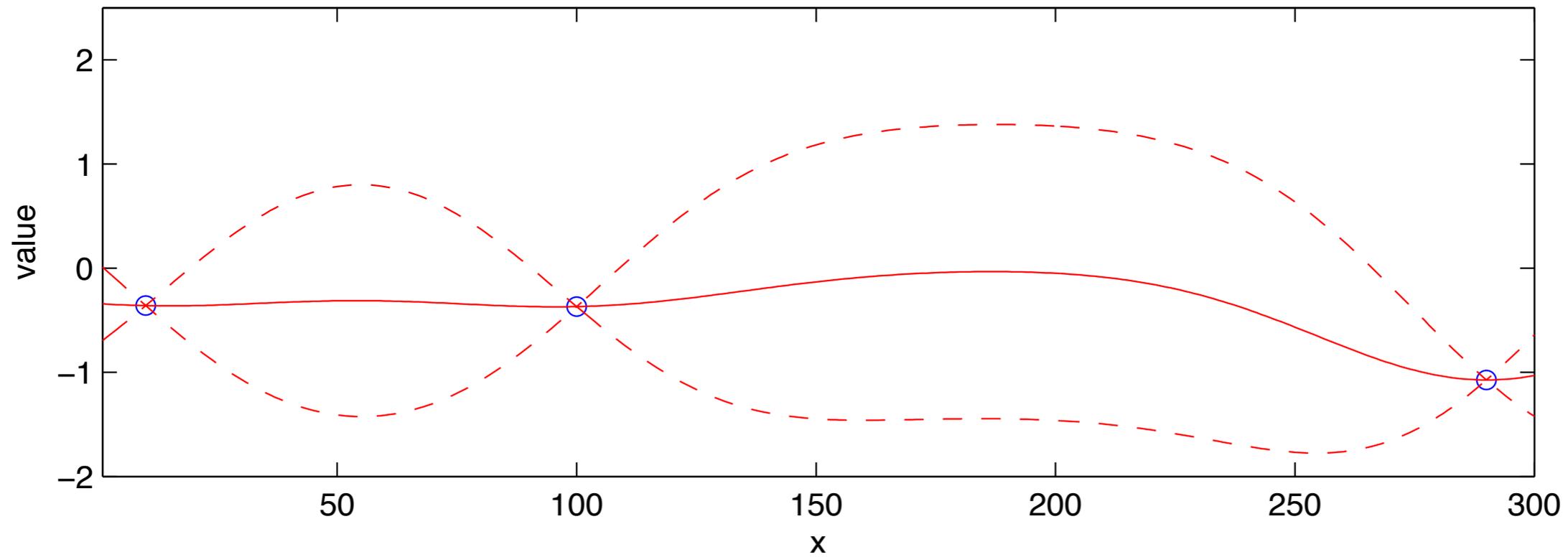
}

Background:

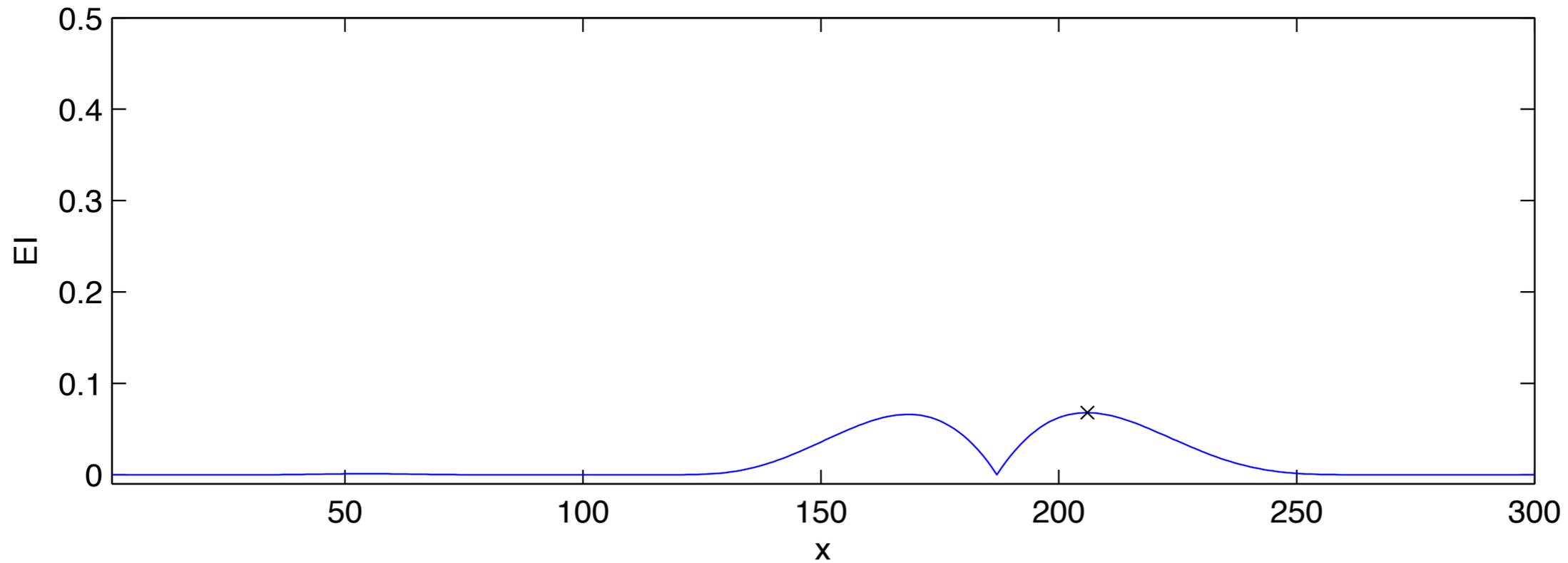
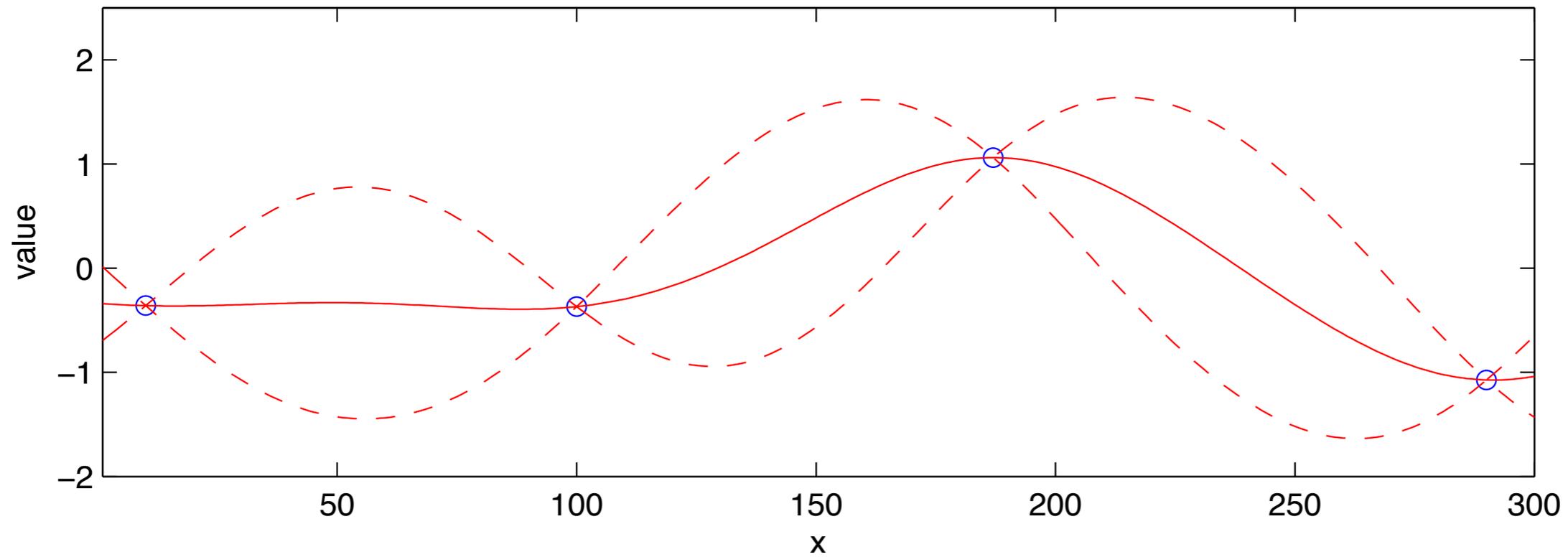
Expected Improvement

- Efficient Global Optimization (EGO)
[Jones, Schonlau & Welch 1998;
Mockus 1972] is a well-known Bayesian
optimization method.
- It does one function evaluation at a time.
- It measures the value of information for
each potential measurement using
“Expected Improvement.”

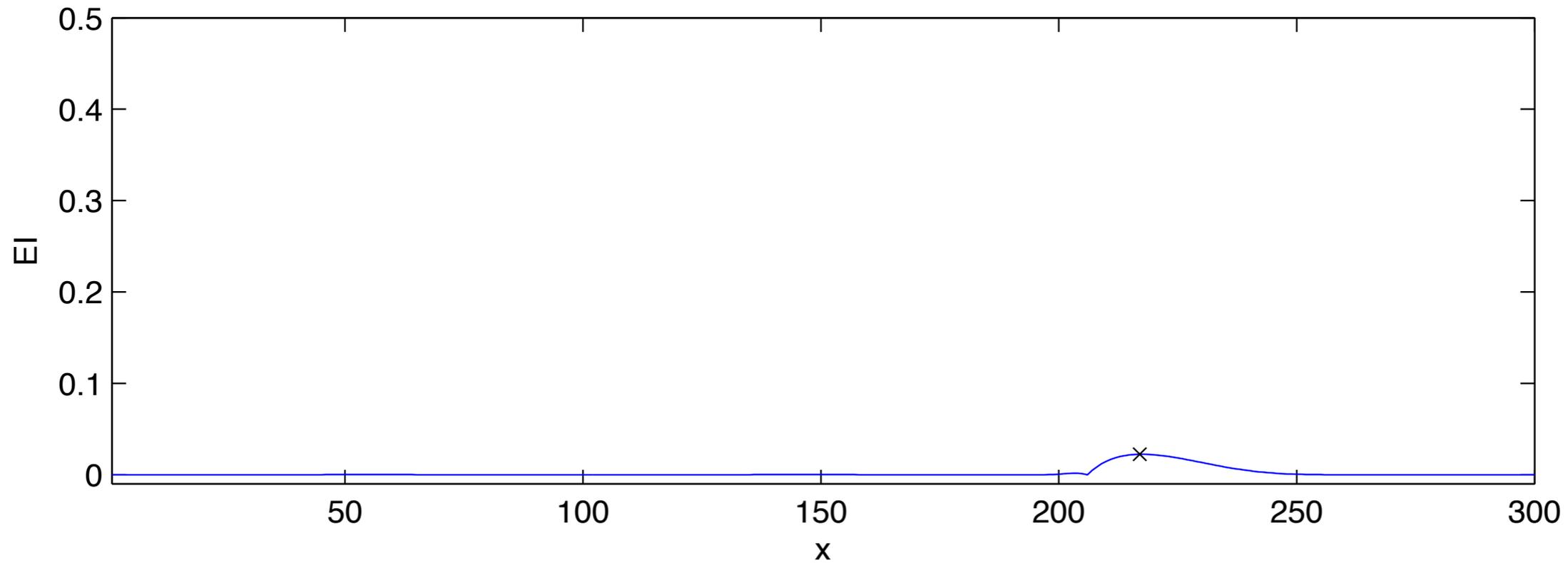
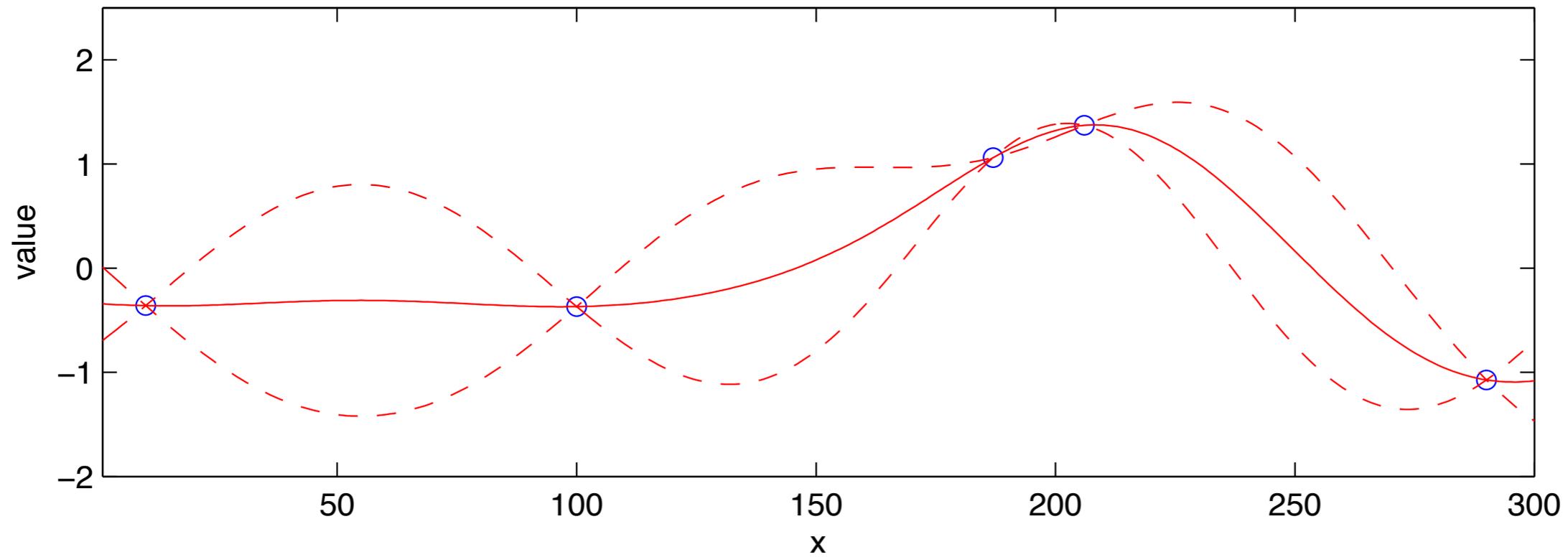
Background: Expected Improvement



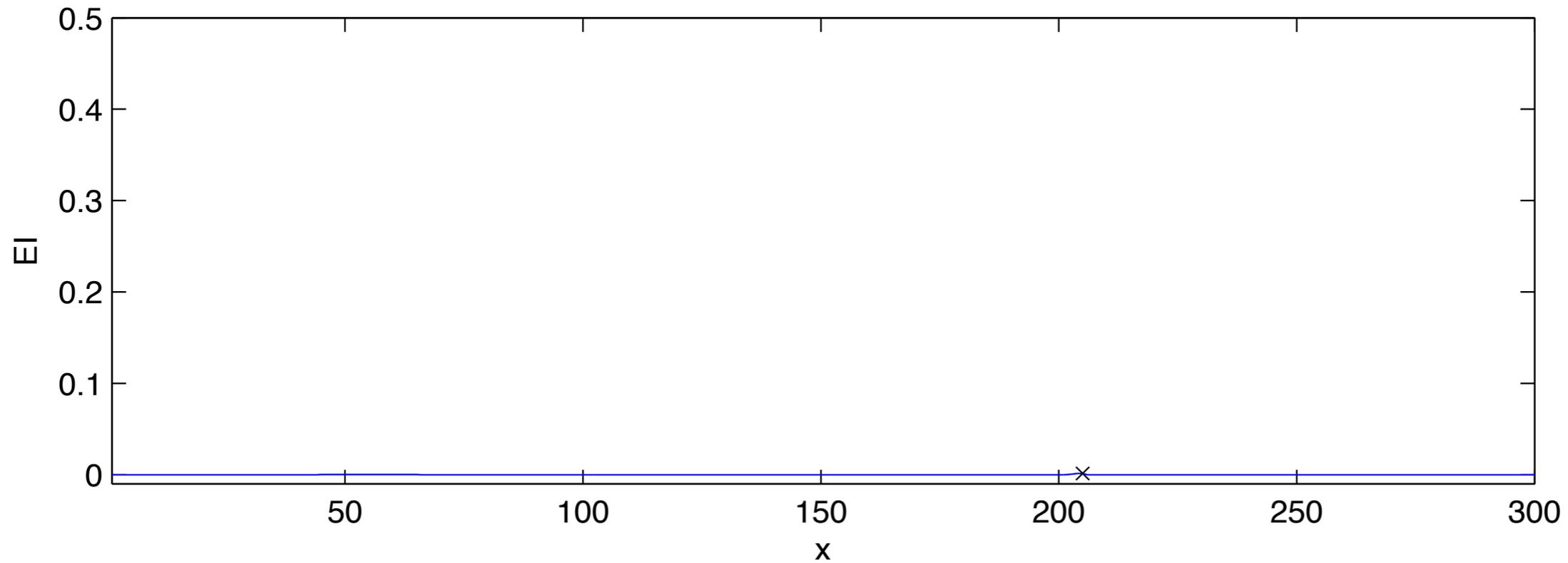
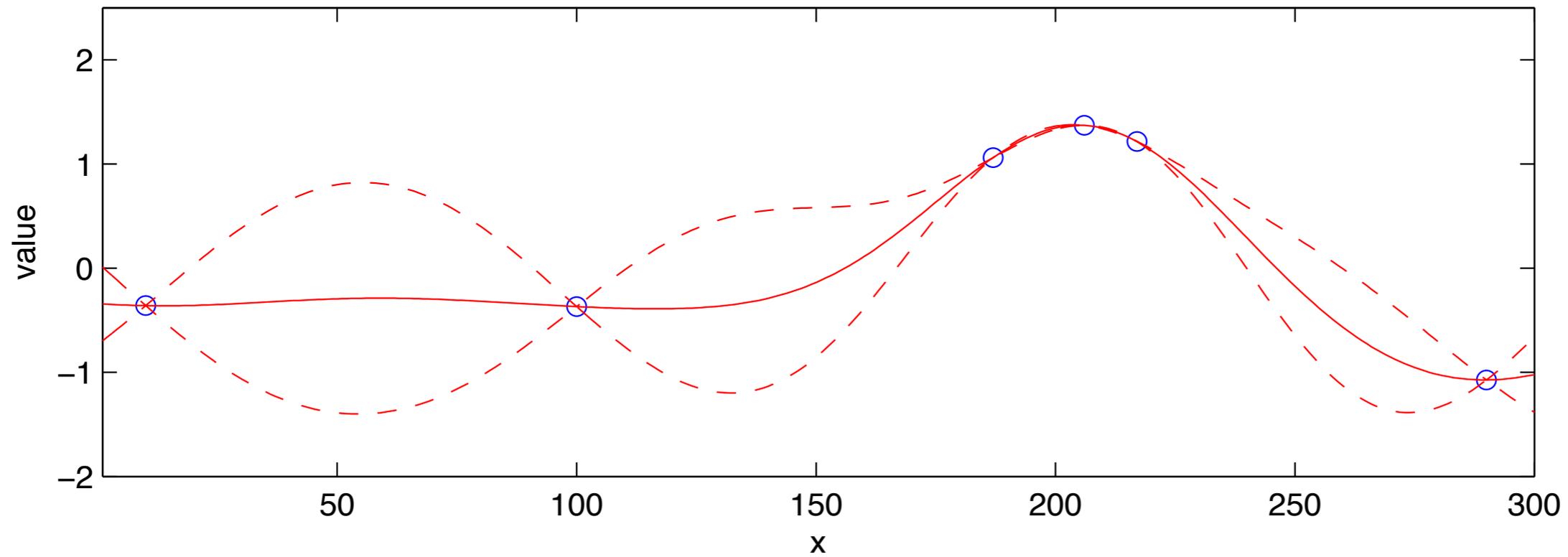
Background: Expected Improvement



Background: Expected Improvement



Background: Expected Improvement



Almost all existing Bayesian Optimization methods take **one** measurement at a time

- EGO / expected improvement take one measurement at a time.
- So do earlier algorithms [Kushner, 1964, Mockus et al., 1978, Mockus, 1989].
- So do later methods [Calvin and Zilinskas, 2005, Villemonteix et al., 2009, Frazier et al., 2009, Huang et al., 2006]
- One exception is a collection of methods by David Ginsbourger and co-authors, and also by Ryan Adams (more later).

We extend Bayesian Optimization to **parallel** function evaluations.



Parallel computer



Parallel A/B tests

- What if we evaluate the function at multiple points simultaneously?
- This happens in parallel computing, A/B testing on the web, and laboratory experiments.
- We use decision theory.
- This was also suggested by Ginsbourger et al., 2007.

We generalize to multiple function evaluations using a decision-theoretic approach

- We've evaluated $x^{(1)}, \dots, x^{(n)}$, and observed $f(x^{(1)}), \dots, f(x^{(n)})$.
- Once sampling stops, we will select the best point found.
- What is the **Bayes-optimal** way to choose the set of points x_1, \dots, x_q to evaluate next?
- In general, we would need to solve a dynamic program.
- When this is the last stage of measurements, the dynamic program becomes a simpler two-stage optimization problem.

We generalize to multiple function evaluations using a decision-theoretic approach

- We've evaluated $x^{(1)}, \dots, x^{(n)}$, & observed $f(x^{(1)}), \dots, f(x^{(n)})$.

- The **best** value observed is

$$f_n^* = \max\{f(x^{(1)}), \dots, f(x^{(n)})\}$$

- If we measure at new points x_1, \dots, x_q , and then stop, then the expected value of our new solution is

$$E_n[\max(f_n^*, \max_{i=1, \dots, q} f(x_i))]$$

We generalize to multiple function evaluations using a decision-theoretic approach

- The **expected improvement** is $E[\text{value of new solution}] - \text{value of old solution}$

- We write this as

$$EI_n(x_1, \dots, x_q) = E_n[\max(f_n^*, \max_{i=1, \dots, q} f(x_i))] - f_n^*$$

- Our algorithm will be to sample at the set of points with largest expected improvement

$$\operatorname{argmax}_{x_1, \dots, x_q} EI(x_1, \dots, x_q)$$

Our approach is Bayes-optimal for one stage of function evaluations

- If we have one stage of function evaluations left, then evaluating

$$\operatorname{argmax}_{x_1, \dots, x_q} \operatorname{EI}(x_1, \dots, x_q)$$

is Bayes-optimal.

- If we have more than one stage left, it is not, but we feel that it is a well-motivated heuristic.

q-EI lacks an easy-to-compute expression

$$\text{EI}_n(x_1, \dots, x_q) = E_n \left[\left(\max_{i=1, \dots, q} f(x_i) - f_n^* \right)^+ \right]$$

- When $q=1$ (no parallelism), this is the expected improvement of Jones et al., 1998, which has a closed-form expression.
- When $q=2$, Ginsbourger et al., 2007 gives an expression using bivariate normal cdfs.
- When $q > 2$,
Ginsbourger et al., 2007 proposes Monte Carlo estimation;
Chevalier and Ginsbourger, 2013 proposes exact evaluation using repeated calls to high-dimensional multivariate normal cdfs.
Both are difficult to optimize.

q-El is hard to optimize

- From Ginsbourger, 2009: “*directly optimizing the q-El becomes extremely expensive as q and d (the dimension of the inputs) grow.*”
- Rather than optimizing the q-El, Ginsbourger et al., 2007 and Chevalier and Ginsbourger, 2013 propose other schemes.

Our contribution

- Our 1st contribution is an efficient method for solving
$$\operatorname{argmax}_{x_1, \dots, x_q} \operatorname{EI}(x_1, \dots, x_q)$$
- This makes the single-batch Bayes-optimal algorithm implementable, not just conceptual.
- Our 2nd contribution is a high-quality open source implementation, currently in use at Yelp.

Our approach to solving

$$\operatorname{argmax}_{x_1, \dots, x_q} \operatorname{EI}(x_1, \dots, x_q)$$

1. Construct an unbiased estimator of

$$\nabla \operatorname{EI}(x_1, \dots, x_q)$$

using infinitesimal perturbation analysis (IPA).

2. Use multistart stochastic gradient ascent to find an approximate solution to

$$\operatorname{argmax}_{x_1, \dots, x_q} \operatorname{EI}(x_1, \dots, x_q)$$

Here's how we estimate $\nabla E I$

- $Y = [f(x_1), \dots, f(x_q)]'$ is multivariate normal.
- Y 's mean vector m and covariance matrix C depend on x_1, \dots, x_q .
- $Y = m + CZ$, where Z is a vector of independent standard normals.
- $E I(x_1, \dots, x_q) = E[h(Y)]$ for some function h .
- If our problem is well-behaved, then we can switch derivative and expectation:

$$\nabla E I(x_1, \dots, x_q) = E[\nabla h(m + cZ)]$$

Here's how we estimate $\nabla E I$

- $Y = [f(x_1), \dots, f(x_q)]'$ is multivariate normal.
- Y 's mean vector m and covariance matrix C depend on x_1, \dots, x_q .
- $Y = m + CZ$, where Z is a vector of independent standard normals.
- $E I(x_1, \dots, x_q) = E[h(Y)]$ for some function h .
- If our problem is well-behaved, then we can switch derivative and expectation:

$$\nabla E I(x_1, \dots, x_q) = E[\nabla h(m + cZ)]$$

This is our gradient estimator, $g(x_1, \dots, x_q, Z)$

Our gradient estimator is unbiased, given mild sufficient conditions

Theorem

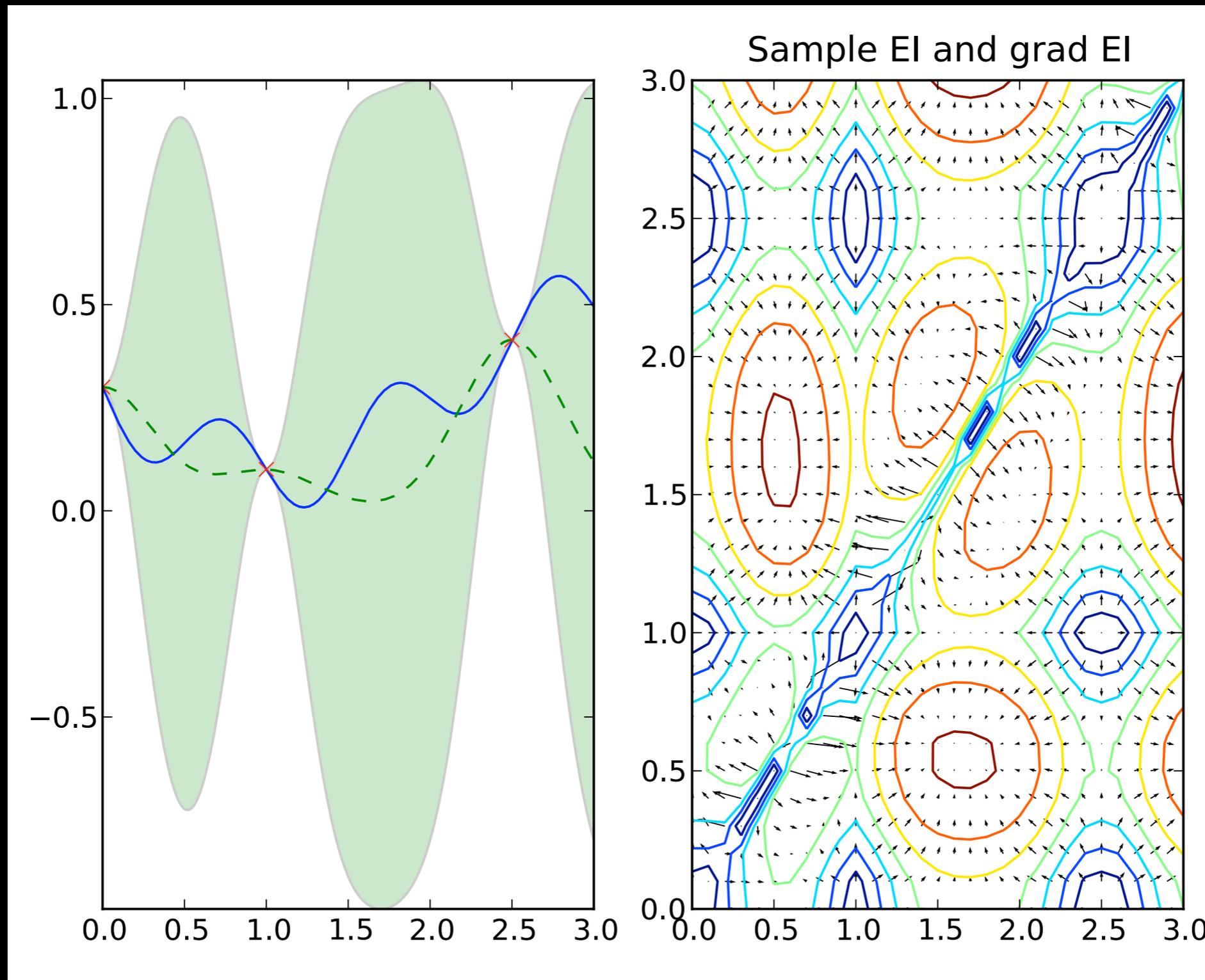
Let $\vec{m}(\vec{x}_1, \dots, \vec{x}_q)$ and $C(\vec{x}_1, \dots, \vec{x}_q)$ be the mean vector and Cholesky factor of the covariance matrix of $(f(\vec{x}_1), \dots, f(\vec{x}_q))$ under the posterior distribution at time n . If the following conditions hold

- $\vec{m}(\cdot)$ and $C(\cdot)$ are three times continuously differentiable in a neighborhood of $\vec{x}_1, \dots, \vec{x}_q$.
- $C(\vec{x}_1, \dots, \vec{x}_q)$ has no duplicated rows.

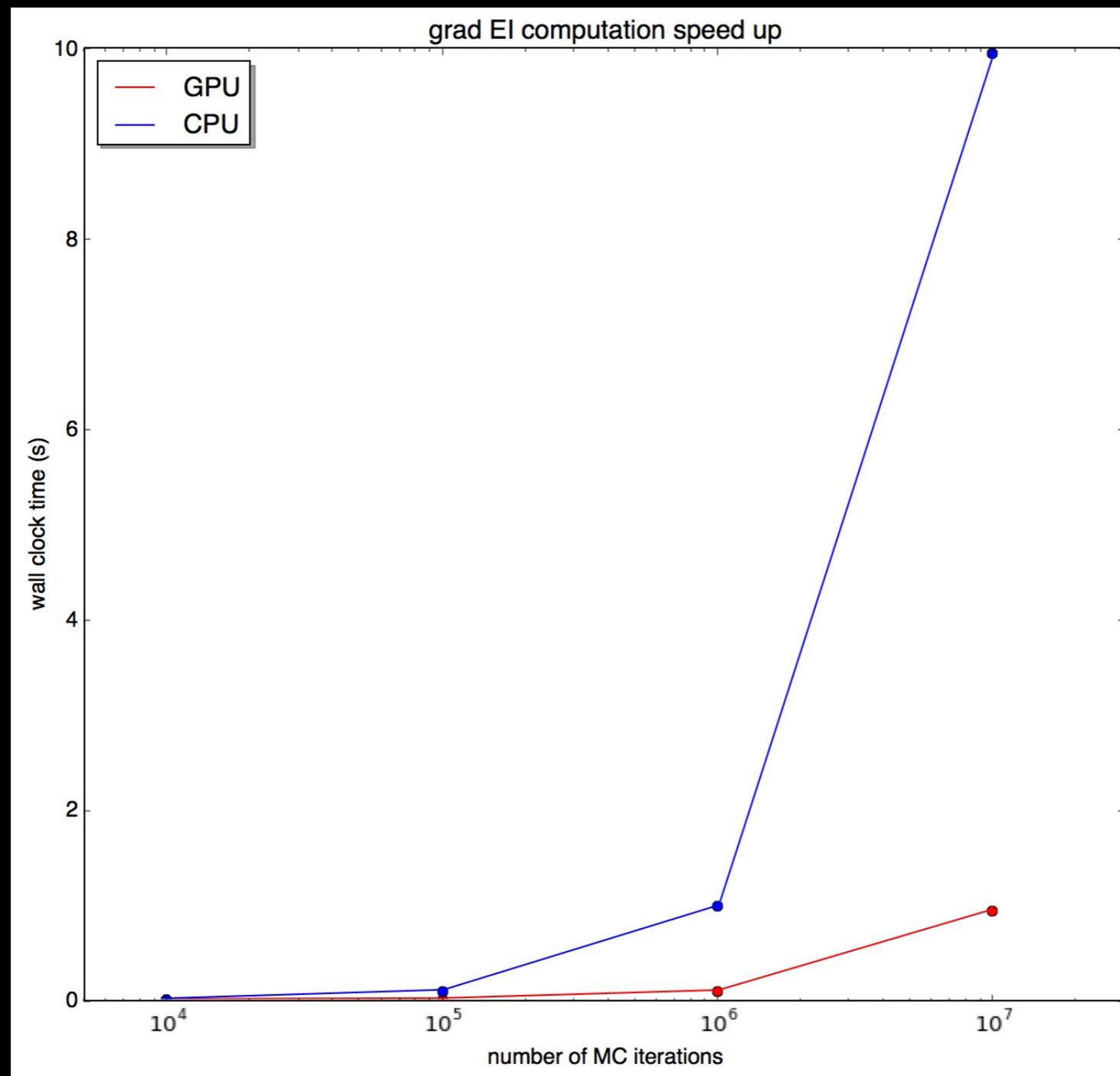
then

$$\nabla \text{EI}(\vec{x}_1, \dots, \vec{x}_q) = \mathbb{E}_n \left[g(\vec{x}_1, \dots, \vec{x}_q, \vec{Z}) \right].$$

Here's what ∇EI looks like



Estimating ∇E_I can be parallelized on a GPU



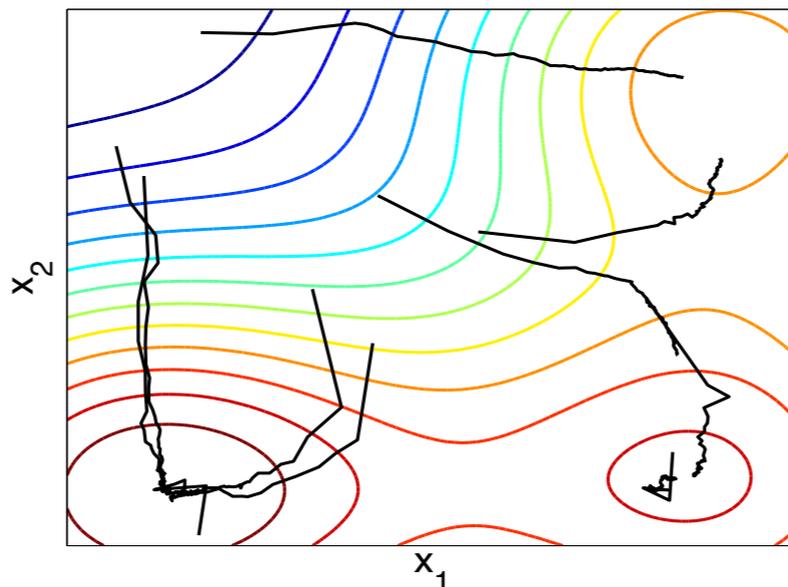
We use this gradient estimator in multistart stochastic gradient ascent

- 1 Select several starting points, uniformly at random.
- 2 From each starting point, iterate using the stochastic gradient method until convergence.

$$(\vec{x}_1, \dots, \vec{x}_q) \leftarrow (\vec{x}_1, \dots, \vec{x}_q) + \alpha_n g(\vec{x}_1, \dots, \vec{x}_q, \omega),$$

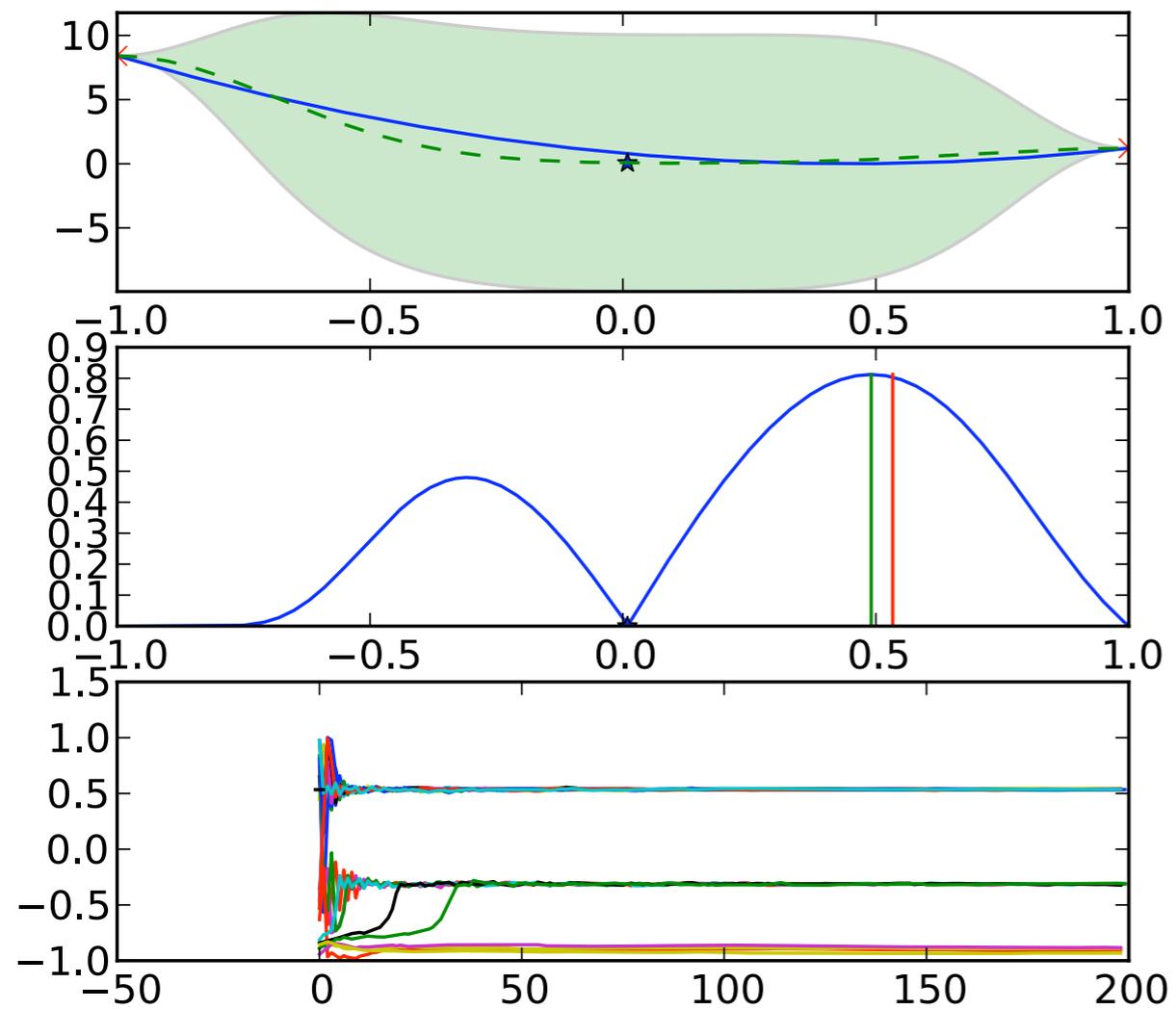
where (α_n) is a stepsize sequence.

- 3 For each starting point, average the iterates to get an estimated stationary point. (Polyak-Ruppert averaging)
- 4 Select the estimated stationary point with the best estimated value as the solution.

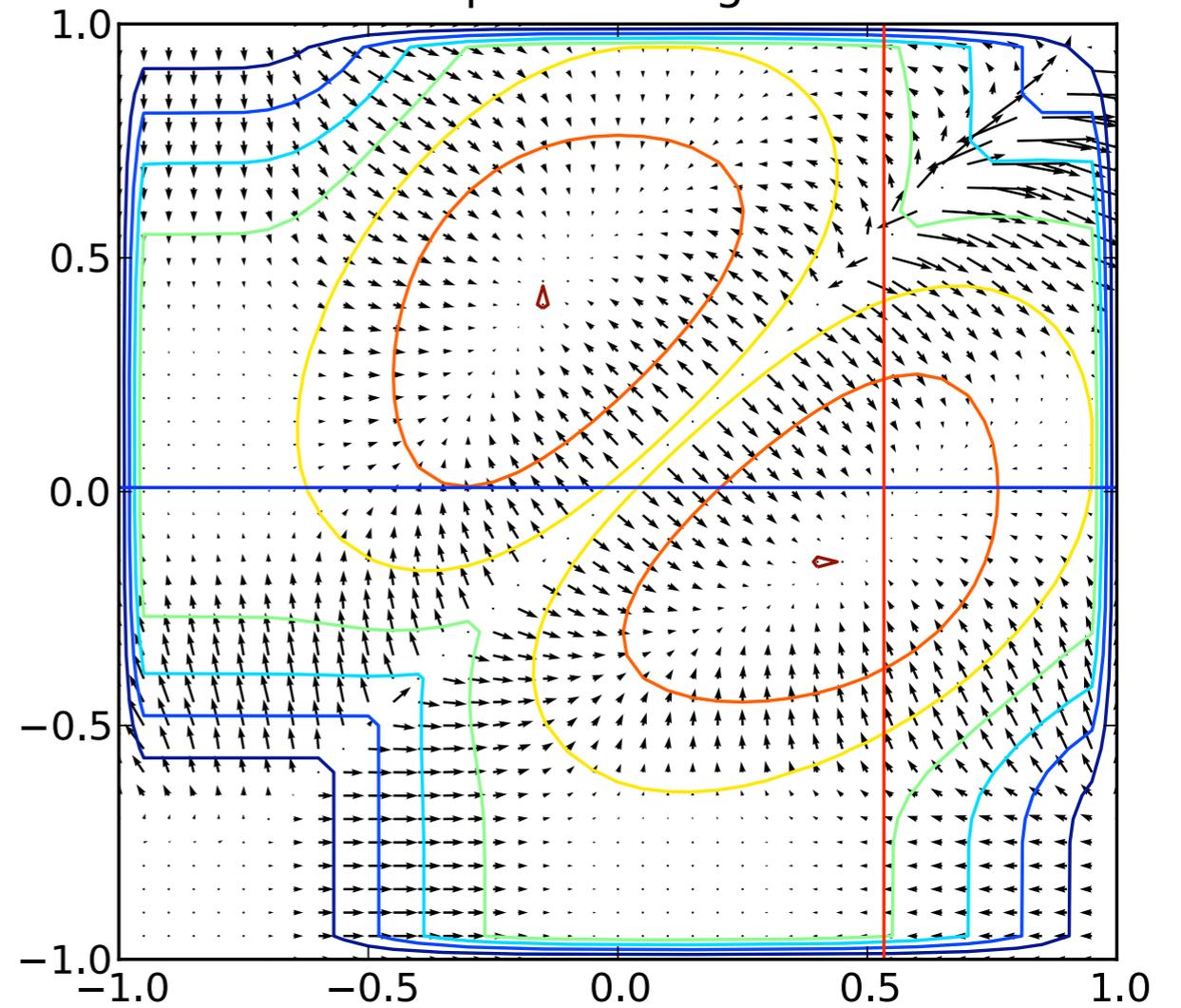


Animation

GPP of points sampled

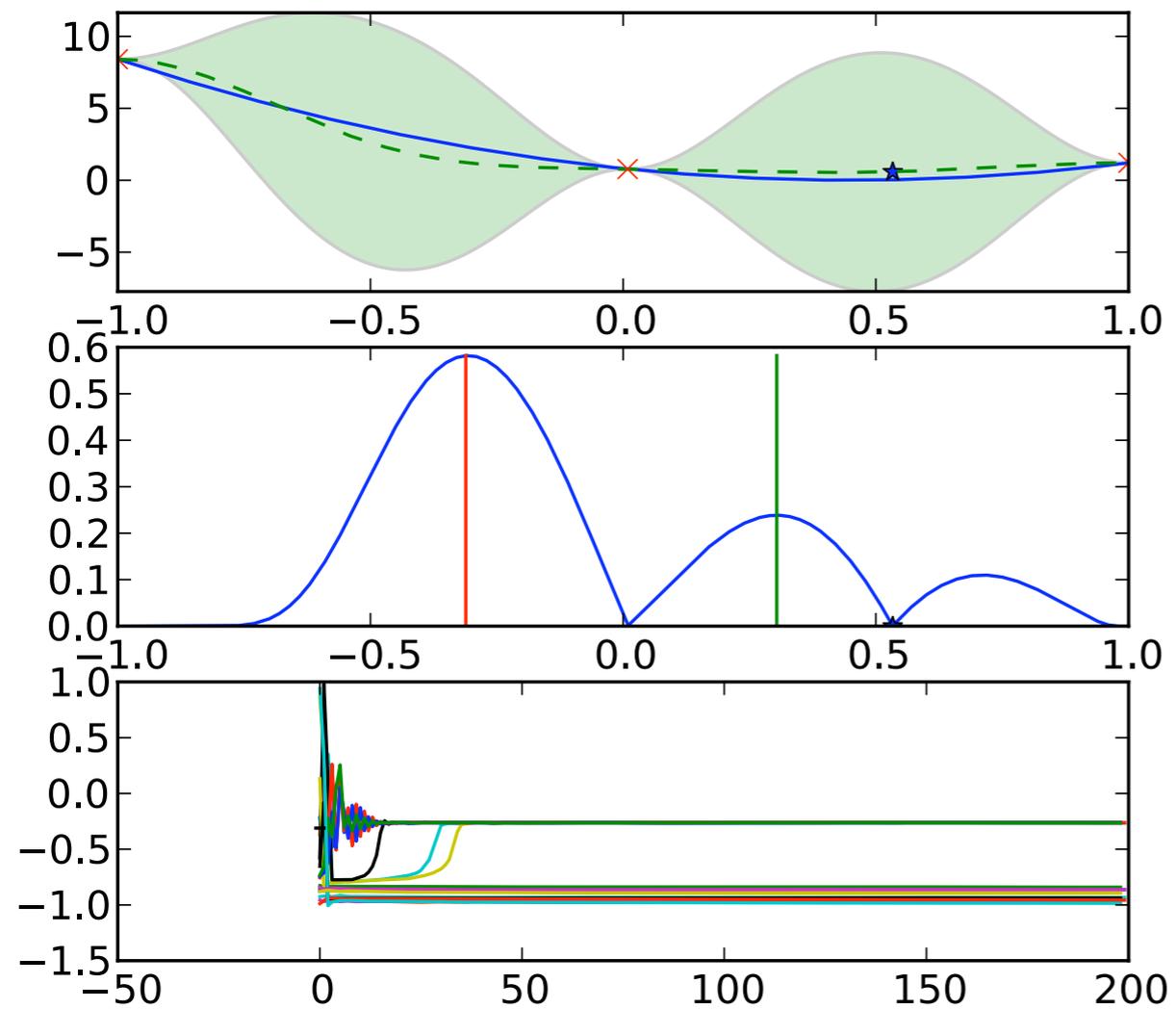


Sample EI and grad EI

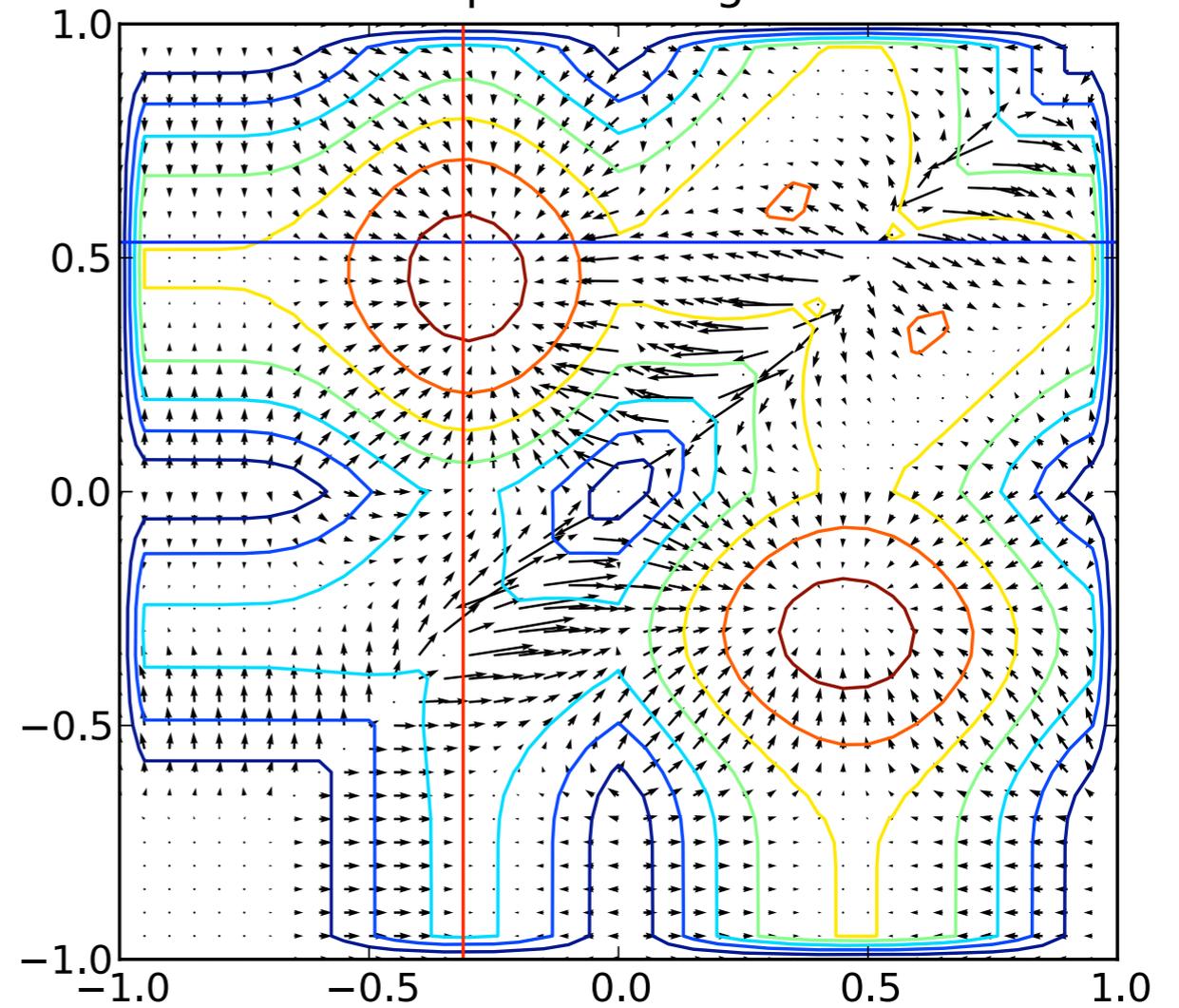


Animation

GPP of points sampled

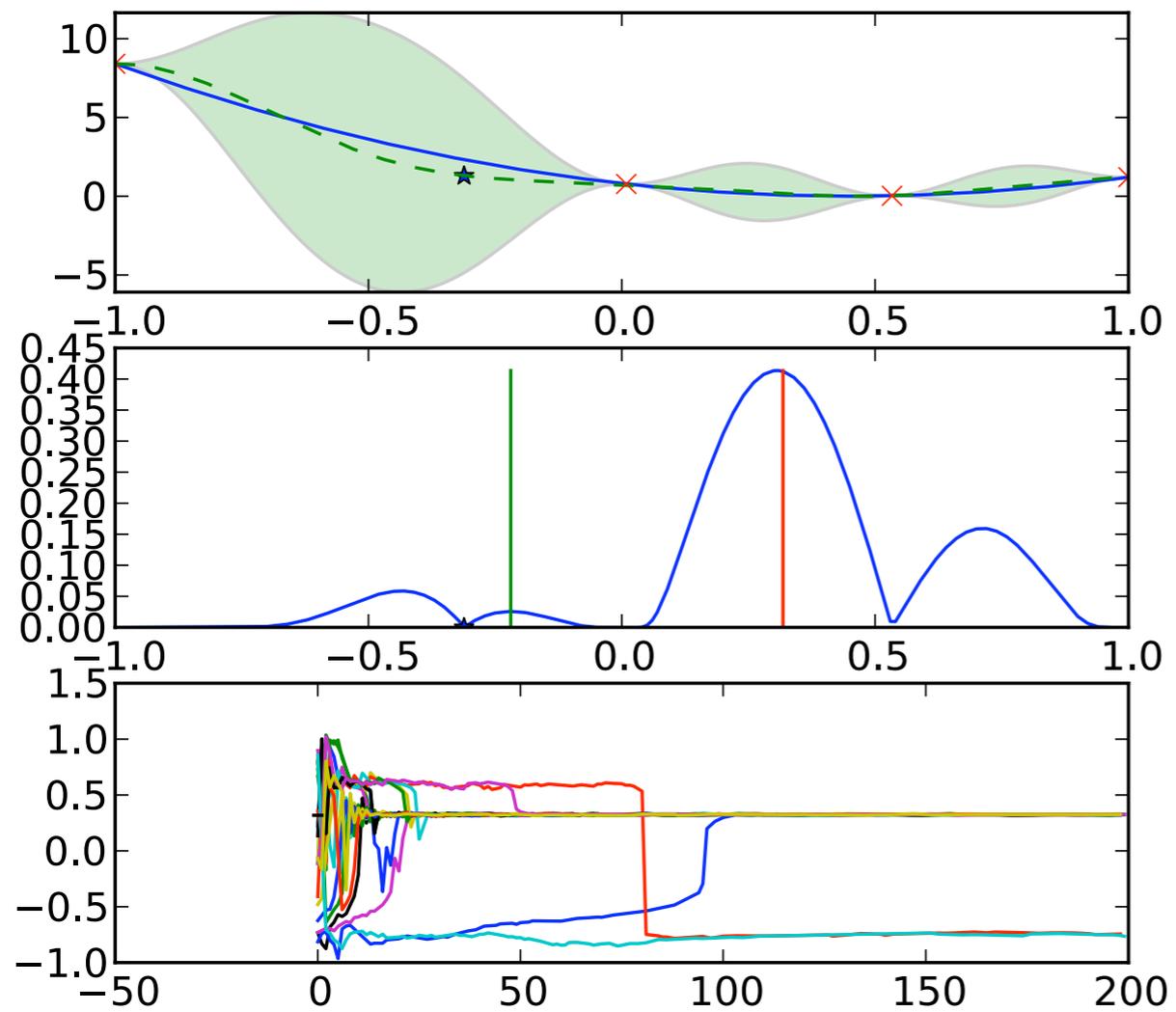


Sample EI and grad EI

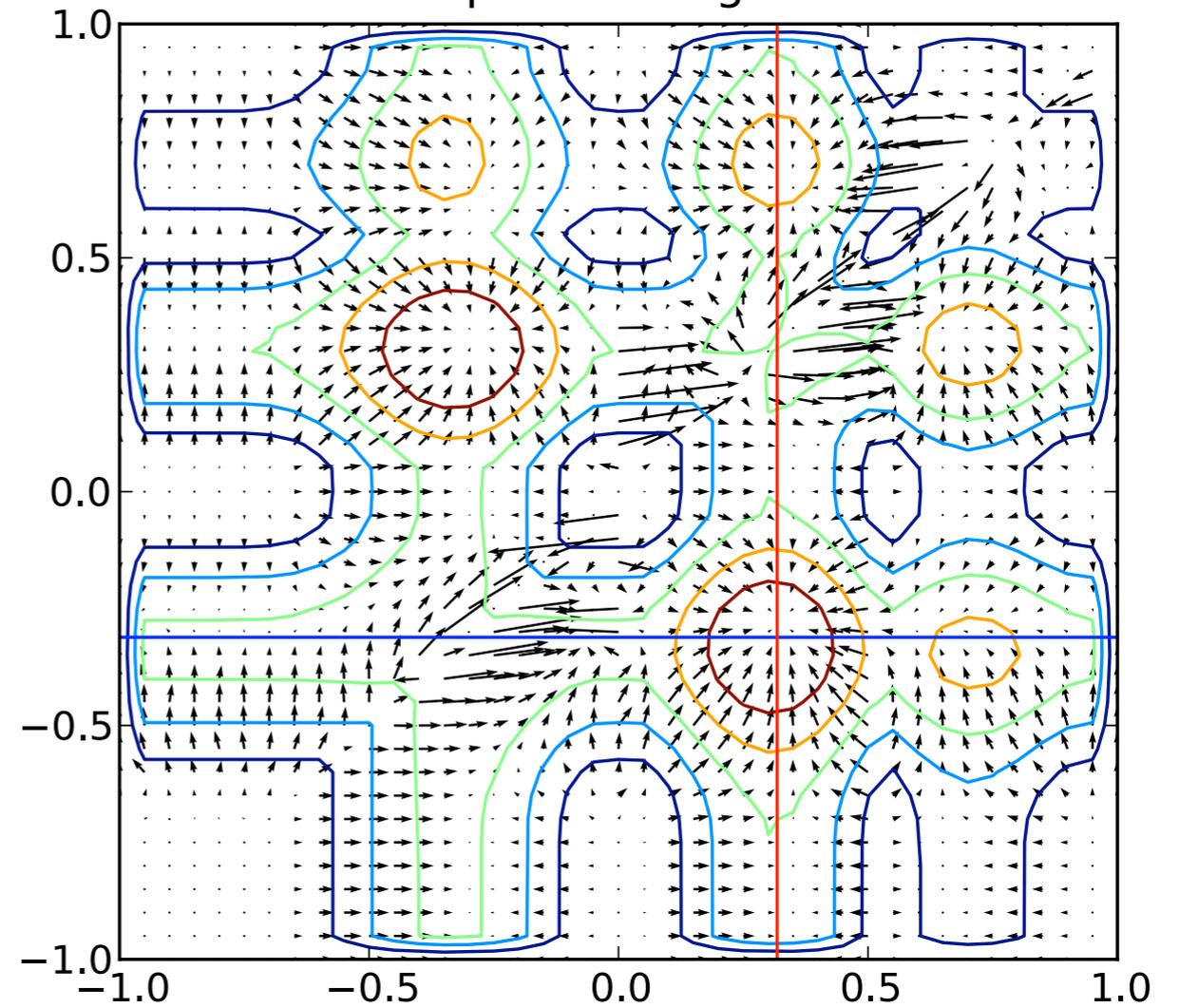


Animation

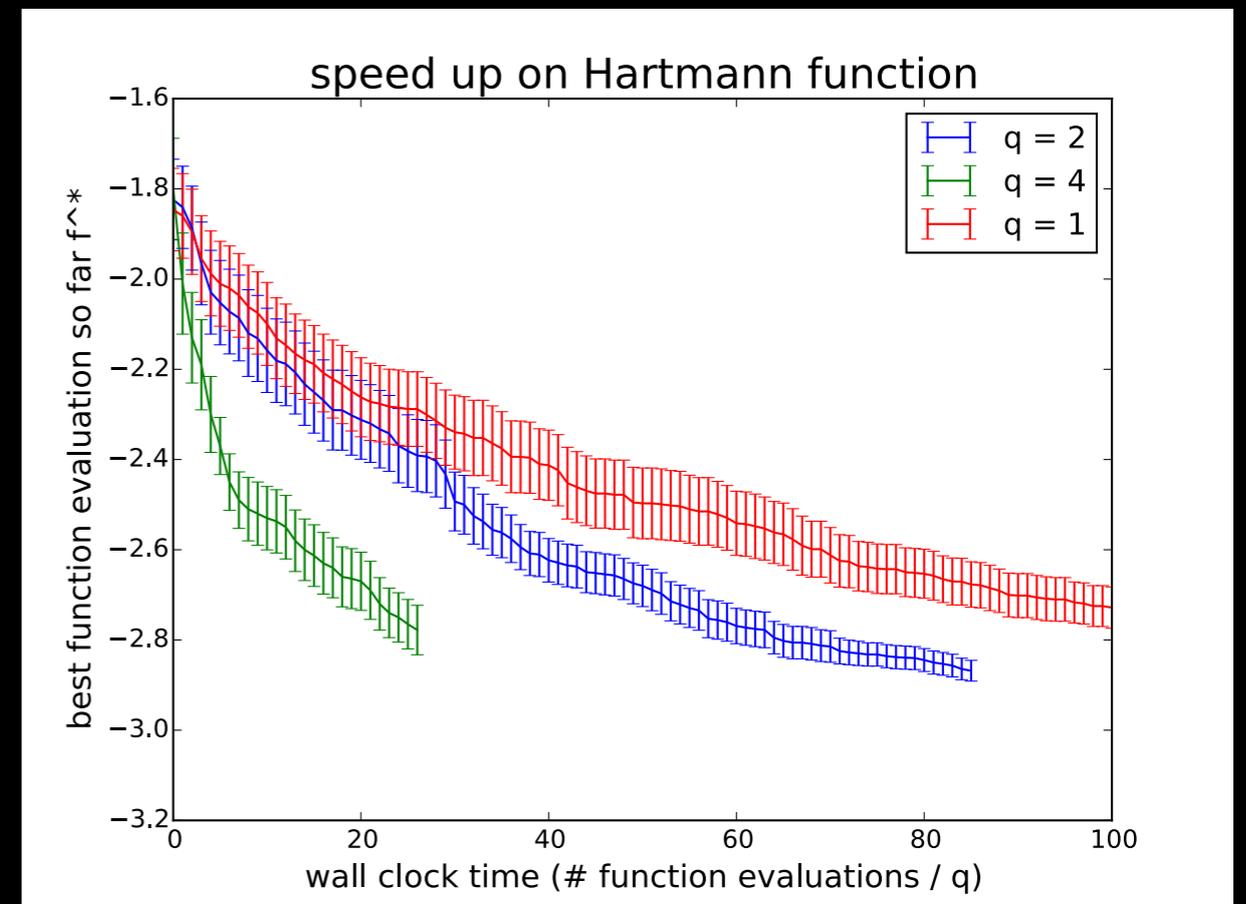
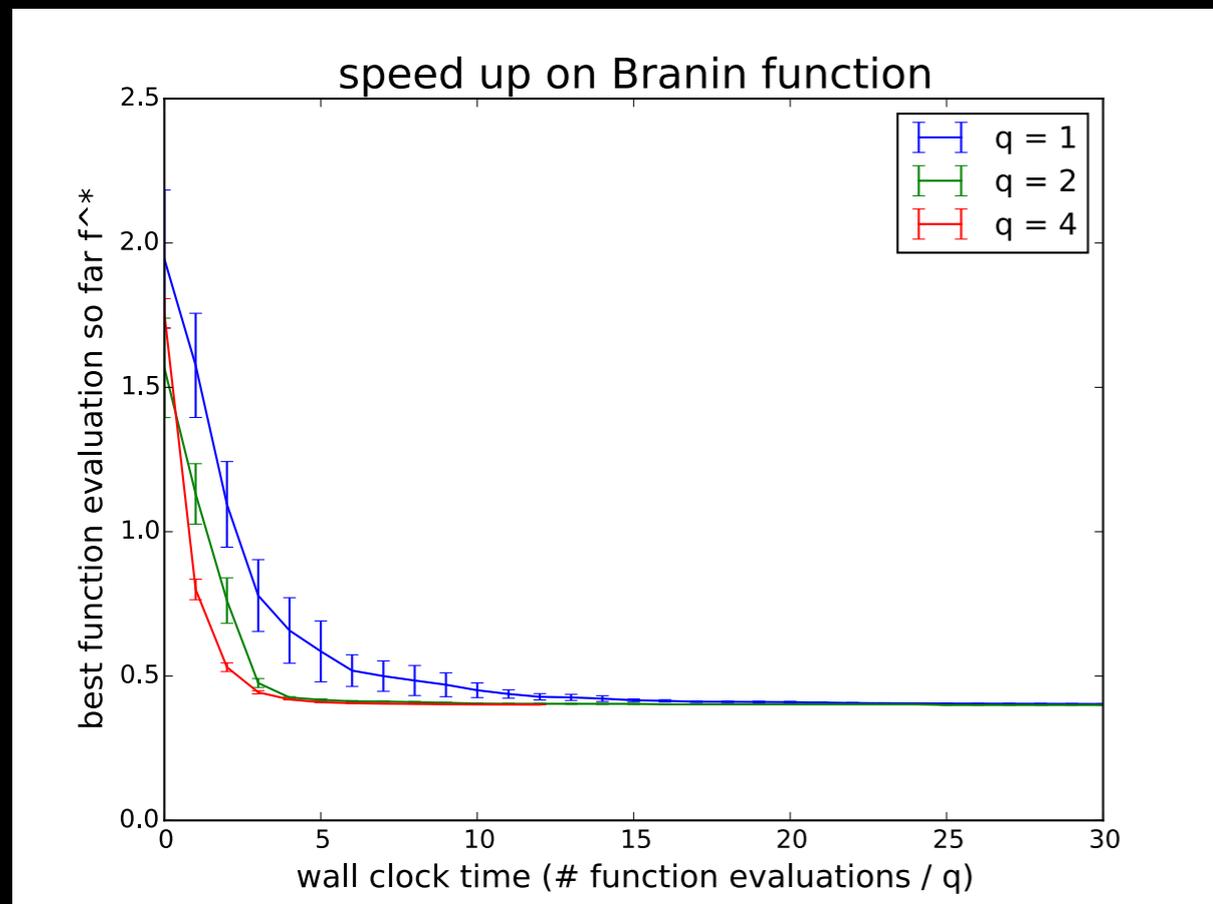
GPP of points sampled



Sample EI and grad EI

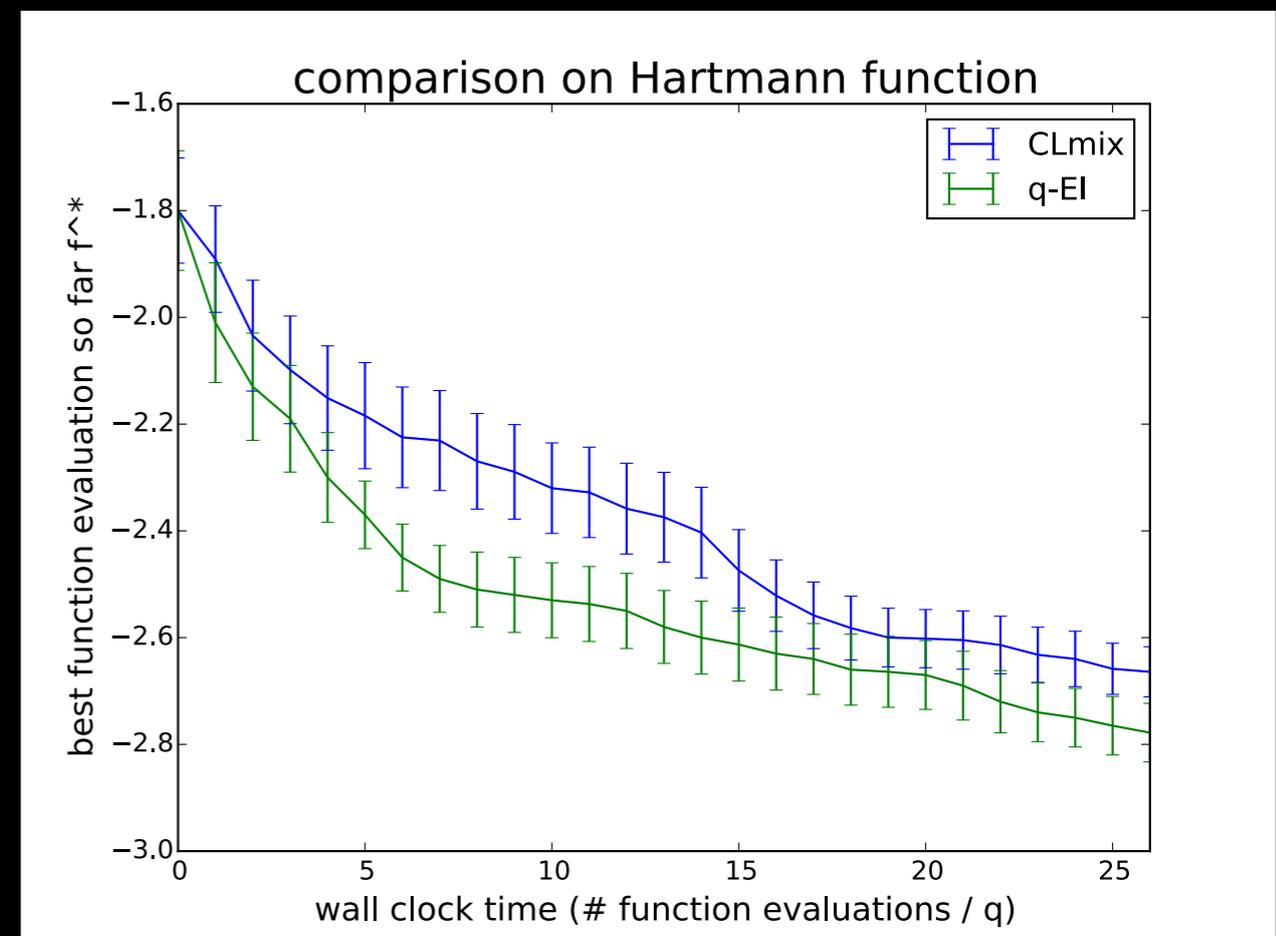
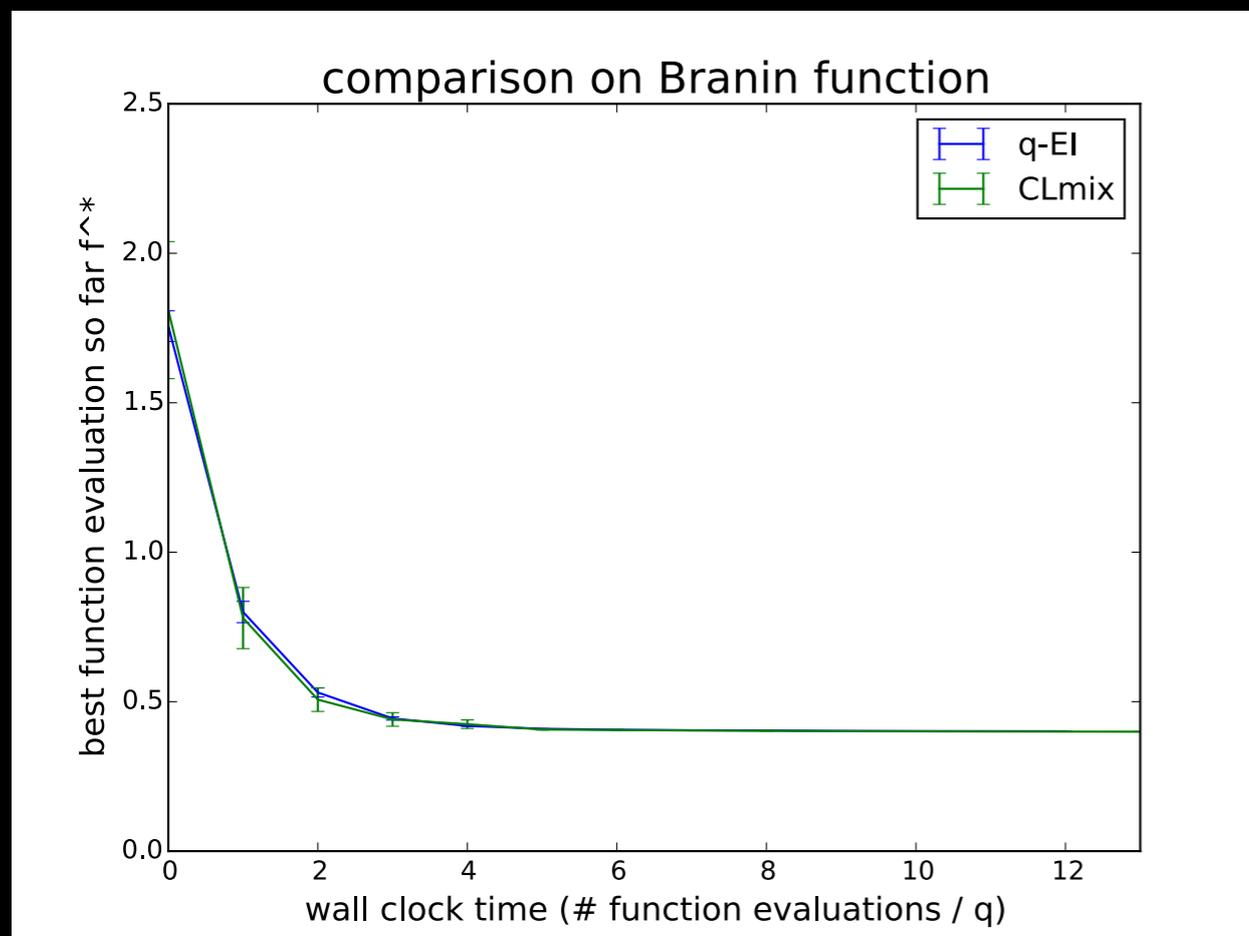


The method works: adding parallelism improves performance



- $q=1$ (one thread) is the EGO method of [Jones et al., 1998]

The method works: it outperforms an approximation to the Bayes-optimal procedure



- Constant Liar (CL) is a class of algorithms proposed by Chevalier & Ginsbourger 2013.
- CL-mix is the best of the CL algorithms.

Our procedure is only Bayes-optimal for a single batch

- If we do just one batch, then our procedure is Bayes-optimal.
- If we run many batches, starting each new batch after the previous one completes, then our procedure is not optimal.

Finding the Bayes-optimal
multi-batch procedure
is hard

- The optimal procedure for $N > 1$ batches is the solution to a partially observable Markov decision process (POMDP).
- This is well-understood theoretically, but very hard computationally.
 - The amount of memory required is exponential in d (the problem dimension), q (the batch size), and N (the number of batches).

We have found Bayes-optimal multi-batch procedures for other related learning problems

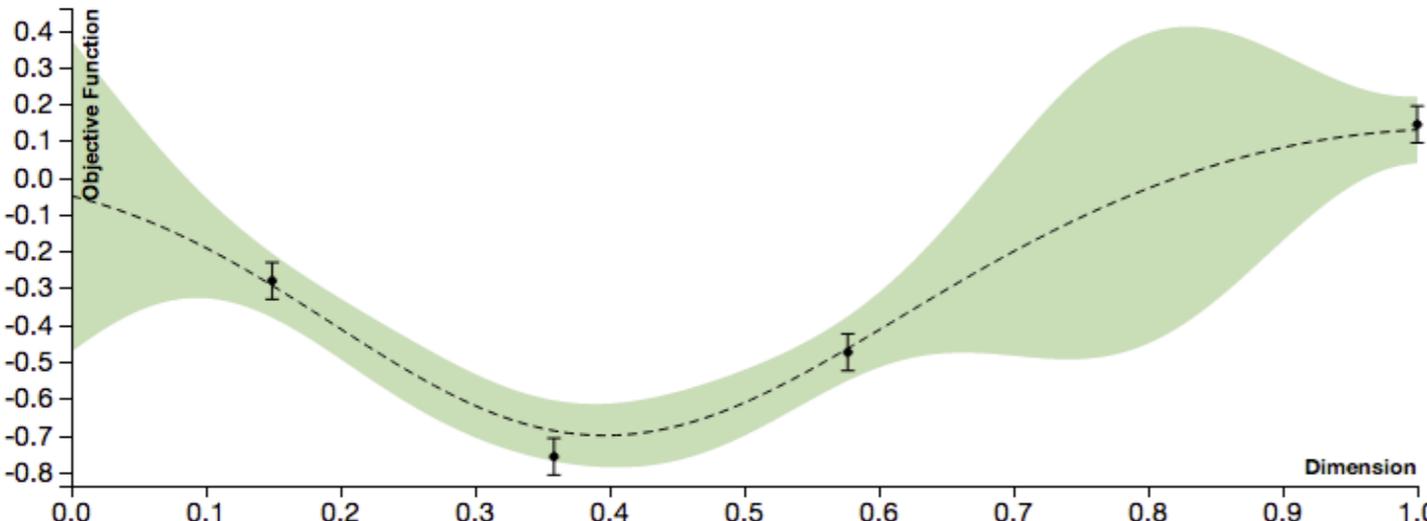
- We have found Bayes-optimal multi-batch procedures, or upper bounds on their value, for these related problems:
 - multiple comparisons [Xie and F., 2013, Hu, F., Xie 2014]
 - stochastic root-finding [Waeber, F., Henderson 2013]
 - ranking & selection (pure exploration MAB) [Xie and F., 2013]
 - information filtering [Zhao and F., 2014]
 - object localization [Jedynak, F., Sznitman, 2012]

With Yelp, we made a high-quality implementation of some of these methods, called MOE (Metrics Optimization Engine)

MOE About Demo

Gaussian Process (GP) ?

Endpoint(s): `gp_mean_var_diag` ?

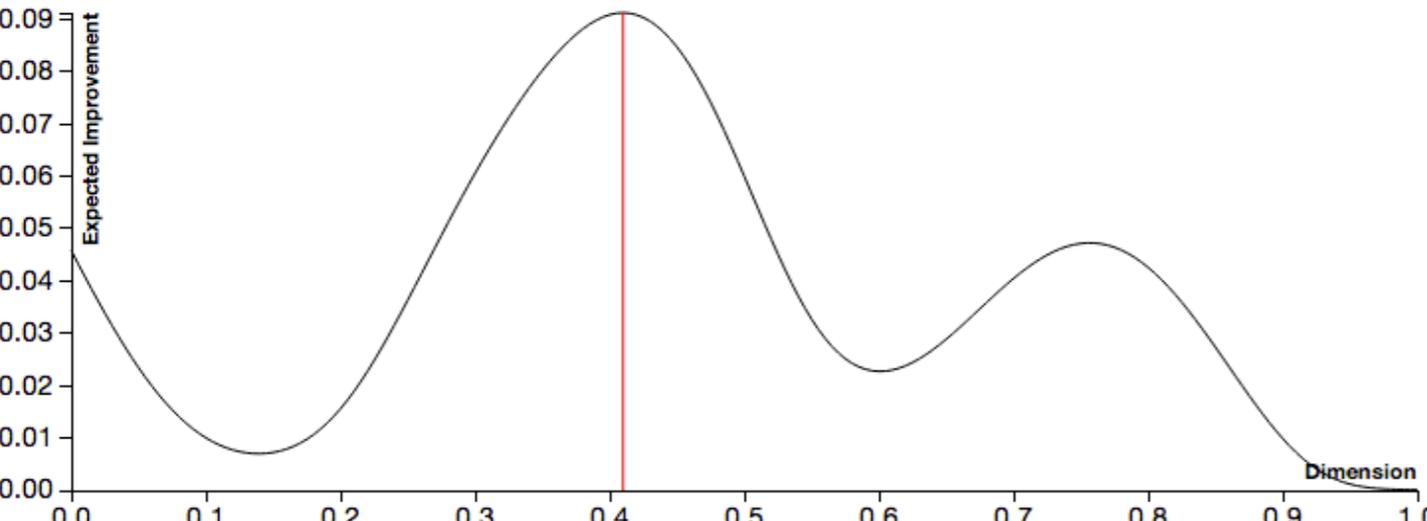


Objective Function

Dimension

Expected Improvement (EI) ?

Endpoint(s): `gp_ei` and `gp_next_points_epi` ?



Expected Improvement

Dimension

GP Parameters ?

Signal Variance ?

Length Scale ?

EI SGD Parameters ?

Multistarts ?

GD Iterations ?

f() = ±

Points Sampled ?

- f(0.3586) = -0.7551 ± 0.1000
- f(0.1492) = -0.2779 ± 0.1000
- f(0.5769) = -0.4718 ± 0.1000
- f(1.0000) = 0.1469 ± 0.1000

MOE is open source

The screenshot shows the GitHub repository page for 'Yelp / MOE'. The browser address bar displays 'https://github.com/Yelp/MOE'. The repository name 'Yelp / MOE' is prominently displayed at the top left. To the right, there are buttons for 'Unwatch' (26), 'Unstar' (417), and 'Fork' (29). Below the repository name, a description reads: 'A global, black box optimization engine for real world metric optimization.' A progress bar indicates repository statistics: 937 commits, 136 branches, 4 releases, and 6 contributors. The current branch is 'master'. A list of recent commits is shown, with the most recent by 'suntzu86' titled 'mark python libs as "SYSTEM"'. The commit list includes folders like 'conda-recipe', 'docs', 'moe', and 'moe_examples', as well as files like '.gitignore', '.mailmap', '.travis.yml', 'AUTHORS.md', 'CHANGELOG.md', 'Dockerfile', and 'LICENSE'. On the right side, there are links for 'Code', 'Issues' (141), 'Pull Requests' (0), 'Wiki', 'Pulse', and 'Graphs'. At the bottom right, there are options to 'Clone in Desktop' and 'Download ZIP', along with the HTTPS clone URL: 'https://github.com/'.

GitHub, Inc. [US] <https://github.com/Yelp/MOE>

This repository Search Explore Gist Blog Help peter-i-frazier

Yelp / MOE Unwatch 26 Unstar 417 Fork 29

A global, black box optimization engine for real world metric optimization.

937 commits 136 branches 4 releases 6 contributors

branch: master MOE / +

mark python libs as "SYSTEM" ...

suntzu86 authored 17 days ago latest commit 97949e56c3

conda-recipe	pre-tagging version bumping	a month ago
docs	Updated youtube links	a month ago
moe	mark python libs as "SYSTEM"	17 days ago
moe_examples	-fixing scoping issues in fixtures: previously T.class_setup style se...	3 months ago
.gitignore	Added test for _make_bandit_historical_info_from_params in views util...	5 months ago
.mailmap	Added .mailmap to clean up authors	5 months ago
.travis.yml	fixing cmake bug (EXISTS vs DFEINED), fixing travis to use virtualenv...	5 months ago
AUTHORS.md	Update AUTHORS.md	6 months ago
CHANGELOG.md	Fixing UCB1 and UCB1-tuned algorithm calculation of upper confidence ...	a month ago
Dockerfile	Updated from reviews	6 months ago
LICENSE	Update LICENSE	

Code Issues 141 Pull Requests 0 Wiki Pulse Graphs

HTTPS clone URL <https://github.com/>

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop Download ZIP

MOE is in production at Yelp & Netflix,
and is being considered by Wayfair, Tripadvisor, & others...

The screenshot shows the GitHub repository page for 'Yelp / MOE'. The browser address bar displays 'https://github.com/Yelp/MOE'. The repository name 'Yelp / MOE' is prominently displayed at the top left. To the right of the repository name are buttons for 'Unwatch' (26), 'Unstar' (417), and 'Fork' (29). Below the repository name is a description: 'A global, black box optimization engine for real world metric optimization.' A progress bar shows 937 commits, 136 branches, 4 releases, and 6 contributors. The current branch is 'master'. A list of files and folders is shown, including 'conda-recipe', 'docs', 'moe', 'moe_examples', '.gitignore', '.mailmap', '.travis.yml', 'AUTHORS.md', 'CHANGELOG.md', 'Dockerfile', and 'LICENSE'. The right sidebar contains navigation links for 'Code', 'Issues' (141), 'Pull Requests' (0), 'Wiki', 'Pulse', and 'Graphs'. At the bottom right, there are buttons for 'Clone in Desktop' and 'Download ZIP', along with the HTTPS clone URL: 'https://github.com/'.

GitHub, Inc. [US] <https://github.com/Yelp/MOE>

This repository Search Explore Gist Blog Help peter-i-frazier

Yelp / MOE Unwatch 26 Unstar 417 Fork 29

A global, black box optimization engine for real world metric optimization.

937 commits 136 branches 4 releases 6 contributors

branch: master MOE / +

mark python libs as "SYSTEM" ...

suntzu86 authored 17 days ago latest commit 97949e56c3

conda-recipe	pre-tagging version bumping	a month ago
docs	Updated youtube links	a month ago
moe	mark python libs as "SYSTEM"	17 days ago
moe_examples	-fixing scoping issues in fixtures: previously T.class_setup style se...	3 months ago
.gitignore	Added test for _make_bandit_historical_info_from_params in views util...	5 months ago
.mailmap	Added .mailmap to clean up authors	5 months ago
.travis.yml	fixing cmake bug (EXISTS vs DFEINED), fixing travis to use virtualenv...	5 months ago
AUTHORS.md	Update AUTHORS.md	6 months ago
CHANGELOG.md	Fixing UCB1 and UCB1-tuned algorithm calculation of upper confidence ...	a month ago
Dockerfile	Updated from reviews	6 months ago
LICENSE	Update LICENSE	6 months ago

Code Issues 141 Pull Requests 0 Wiki Pulse Graphs

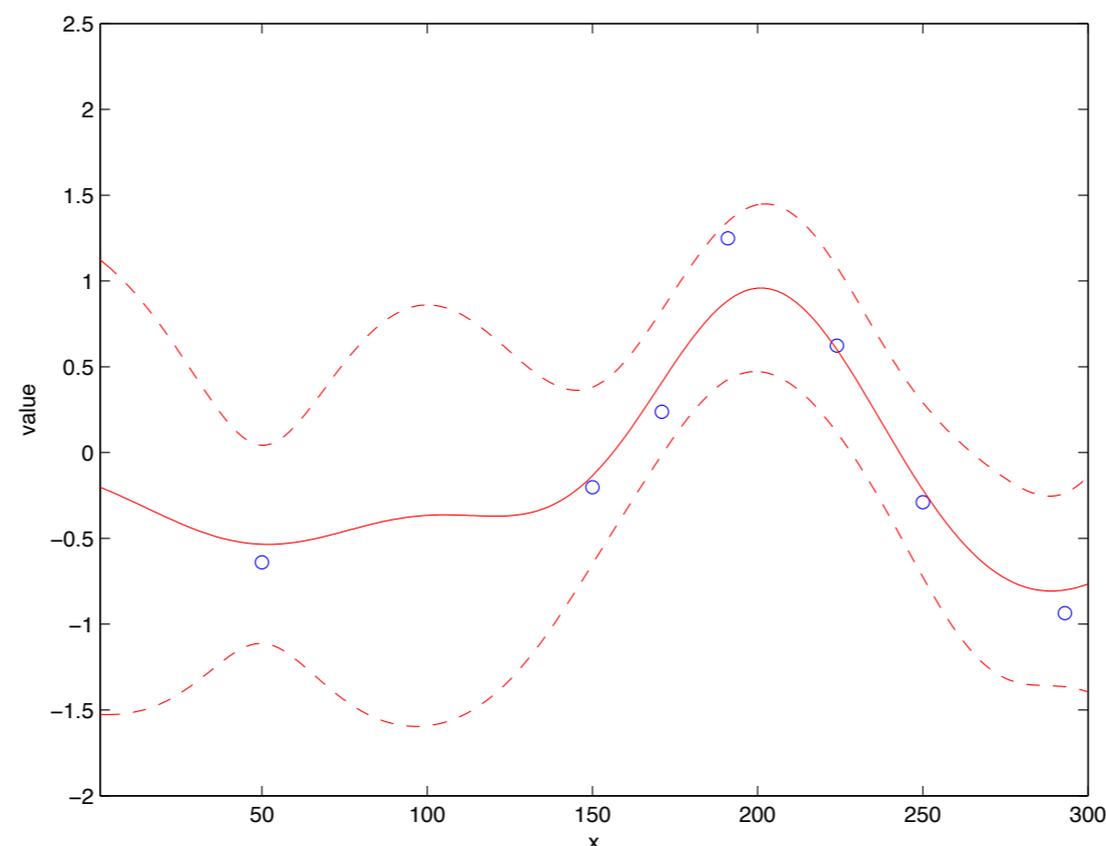
HTTPS clone URL <https://github.com/>

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop Download ZIP

This q-EI method can be used in the noisy case, but it loses its decision-theoretic motivation

- We use Gaussian process regression with normally distributed noise.
- The red line is the posterior mean, $\mu_n(x) = \mathbb{E}_n[f(x)]$
- The largest posterior mean is $\mu_n^* = \max_{i=1,\dots,n} \mu_n(\vec{x}^{(m)})$.



- We use $\text{EI}_n(\vec{x}_1, \dots, \vec{x}_q) = \mathbb{E}_n [(\max_{m=1,\dots,q} \mu_{n+1}(\vec{x}_i) - \mu_n^*)^+]$
- This ignores that $\mu_{n+1}(x) \neq \mu_n(x)$ for previously evaluated x .
- A more principled approach is possible (e.g., generalize knowledge gradient method to multiple evaluations), but we haven't done it yet.

Thanks!

Any Questions?

- This was joint work with:



Scott Clark
Cornell PhD '12
Yelp, SigOpt



Eric Liu
Yelp, SigOpt



Jialei Wang
Cornell PhD
student
Yelp intern



Deniz Oktay
MIT undergraduate
Yelp intern



Norases
Vesdapunt
Stanford under-
graduate
Yelp intern