Gaussian processes

A hands-on tutorial

Slides and code: https://github.com/paraklas/GPTutorial

Paris Perdikaris
Massachusetts Institute of Technology, Department of Mechanical Engineering
Web: http://web.mit.edu/parisp/www/
Email: parisp@mit.edu

ICERM
Providence, RI
June 5th, 2017

Tuesday ***June 6, 2017***

Time	Description	Speaker
9:00 – 9:45	Probabilistic Dimensionality Reduction	Neil Lawrence, University of Sheffield and Amazon Research Cambridge
10:00 – 10:30	Coffee Break	
10:30 – 11:15	Bayesian optimization for automating model selection	Roman Garnett, Washington University in St. Louis

Wednesday ***June 7, 2017***

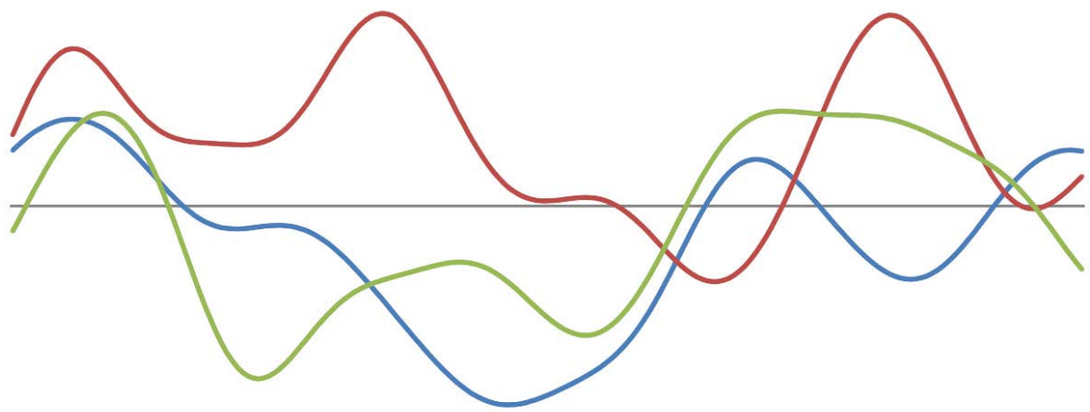
Time	Description	Speaker
9:00 – 9:45	Bayesian Calibration of Simulators with Structured Discretization Uncertainty	Oksana Chkrebtii, The Ohio State University
10:30 – 11:15	Numerical Gaussian Processes for Time-dependent and Non-linear Partial Differential Equations	Maziar Raissi, Brown University
11:30 – 12:15	Variational Reformulation of the Uncertainty Propagation Problem in Linear Partial Differential Equations	Ilias Bilionis, Purdue University
2:30 – 3:15	Bayesian Probabilistic Numerical Methods. (Part I)	Chris Oates, Newcastle University
3:30 – 4:00	Coffee/Tea Break	
4:00 – 4:45	Bayesian Probabilistic Numerical Methods. (Part II)	Jon Cockayne, University of Warwick

GPs will be mentioned in ~50% of the workshop talks!

Data-driven modeling with Gaussian processes

"The linear algebra of computation under uncertainty"

Priors over functions: $f \sim \mathcal{GP}(\mu(x), K(\mathbf{x}, \mathbf{x}'; \theta))$

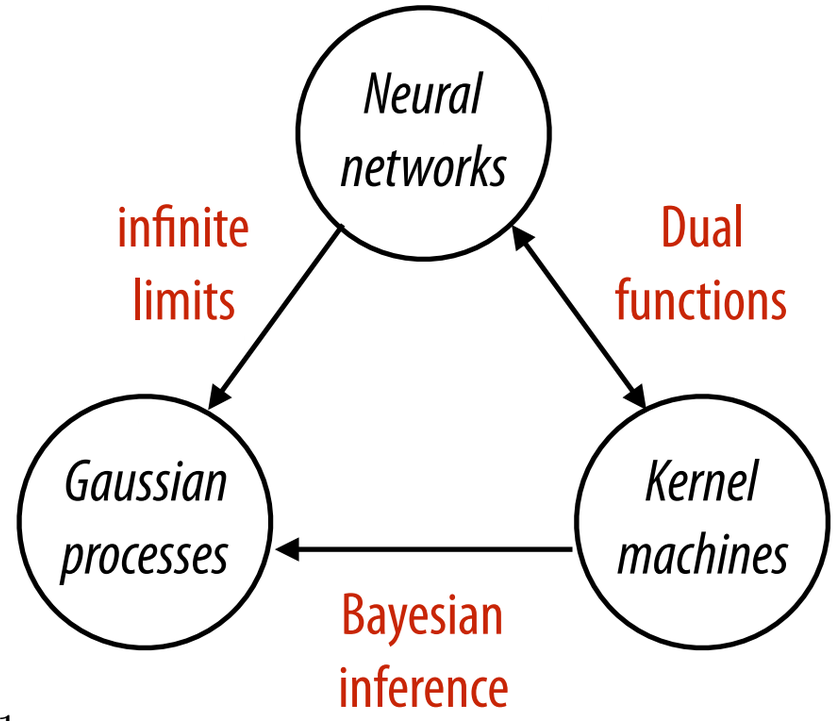


Samples from a GP prior

Marginalization:

$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$. Then:

$$p(\mathbf{f}_A) = \int_{\mathbf{f}_B} p(\mathbf{f}_A, \mathbf{f}_B) \mathrm{d}\mathbf{f}_B = \mathcal{N}(\boldsymbol{\mu}_A, \mathbf{K}_{AA})$$



Posterior is also Gaussian:

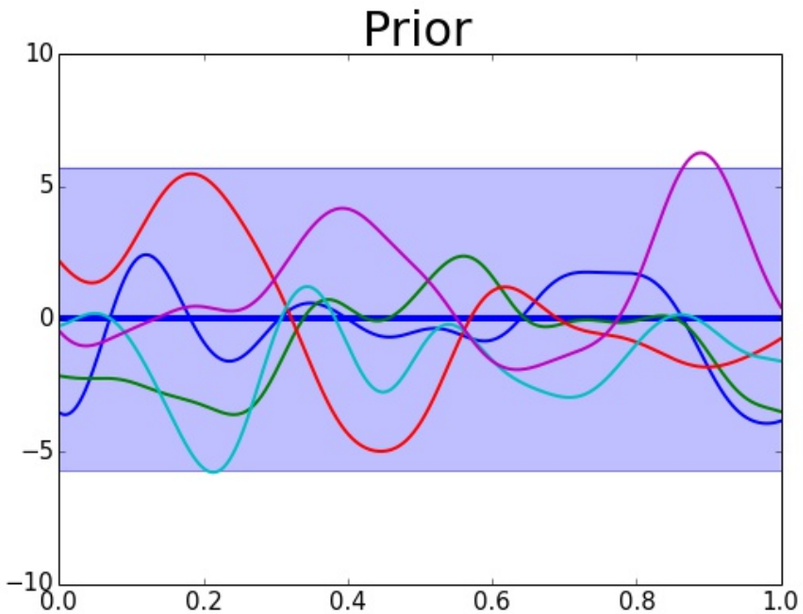
$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$. Then:

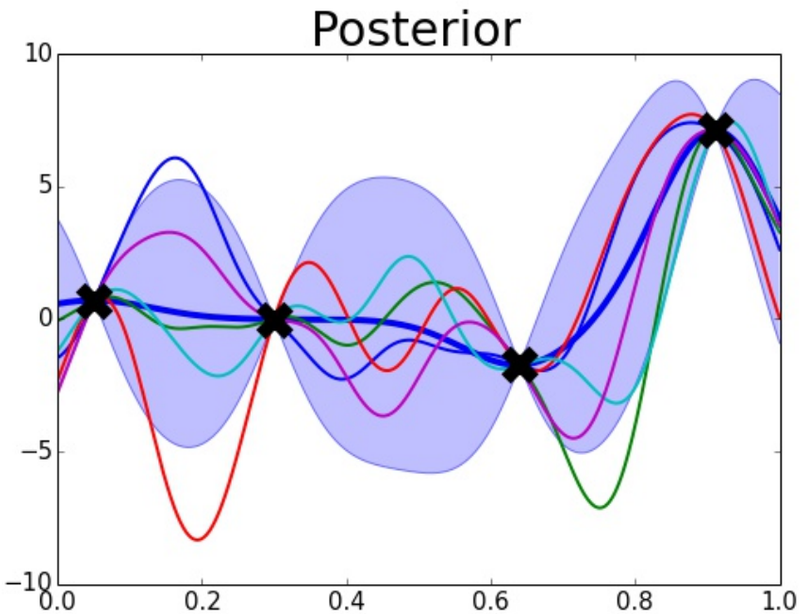
$$p(\mathbf{f}_A|\mathbf{f}_B) = \mathcal{N}(\boldsymbol{\mu}_A + \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}(\mathbf{f}_B - \boldsymbol{\mu}_B), \mathbf{K}_{AA} - \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}\mathbf{K}_{BA})$$

Rasmussen, C. E. Gaussian processes for machine learning (2006)

Data-driven modeling with Gaussian processes

$$y = f(\boldsymbol{x}) + \epsilon \qquad f \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}))$$





Training via maximizing the marginal likelihood

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2}\log|\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log 2\pi$$

Prediction via conditioning on available data

$$p(f_*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}(f_*|\mu_*, \sigma_*^2),$$
$$\mu_*(\boldsymbol{x}_*) = \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{y},$$
$$\sigma_*^2(\boldsymbol{x}_*) = \boldsymbol{k}_{**} - \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{k}_{N*},$$

Rasmussen, C. E. Gaussian processes for machine learning (2006)

Workflow

1.) Model specification: Choosing a prior

$$f \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}))$$

2.) Training the model: Inference algorithm

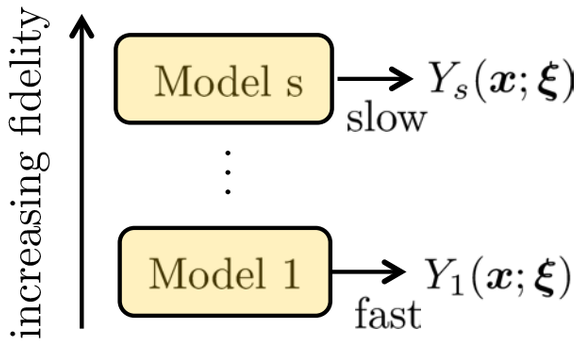
$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2}\log|\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I}| - \frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{y} - \frac{N}{2}\log 2\pi$$

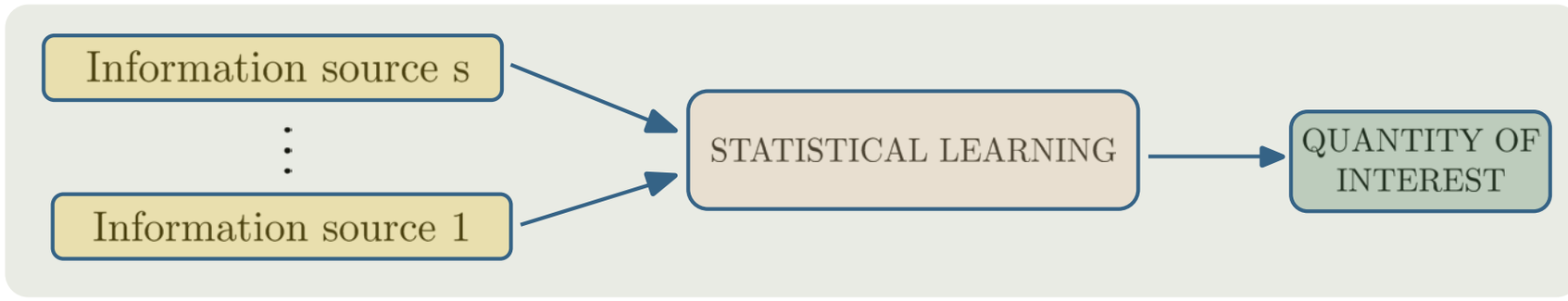
3.) Obtain predictions & quantify uncertainty

$$p(f_*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}(f_*|\mu_*, \sigma_*^2),$$
$$\mu_*(\boldsymbol{x}_*) = \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{y},$$
$$\sigma_*^2(\boldsymbol{x}_*) = \boldsymbol{k}_{**} - \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{k}_{N*},$$

4.) Data acquisition

Multi-fidelity modeling





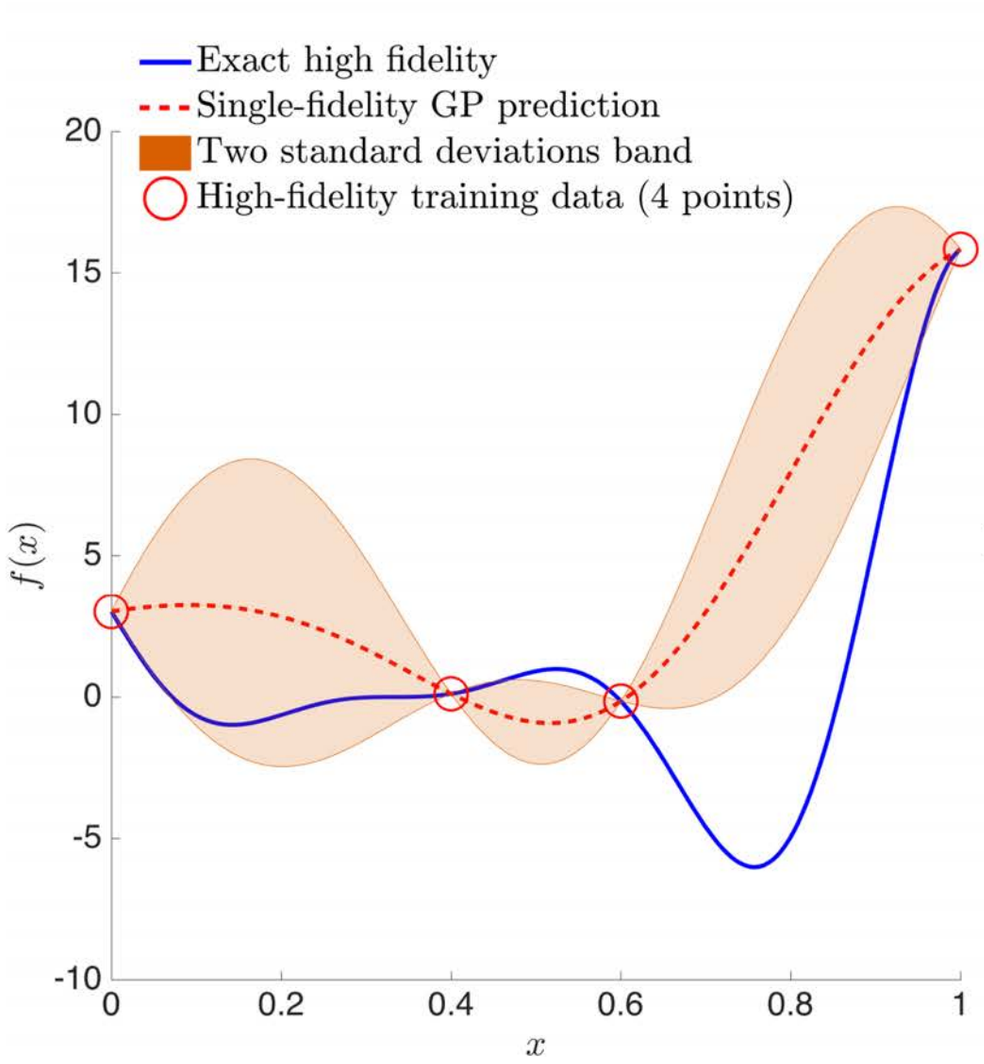
Number of runs is limited by time and computational resources

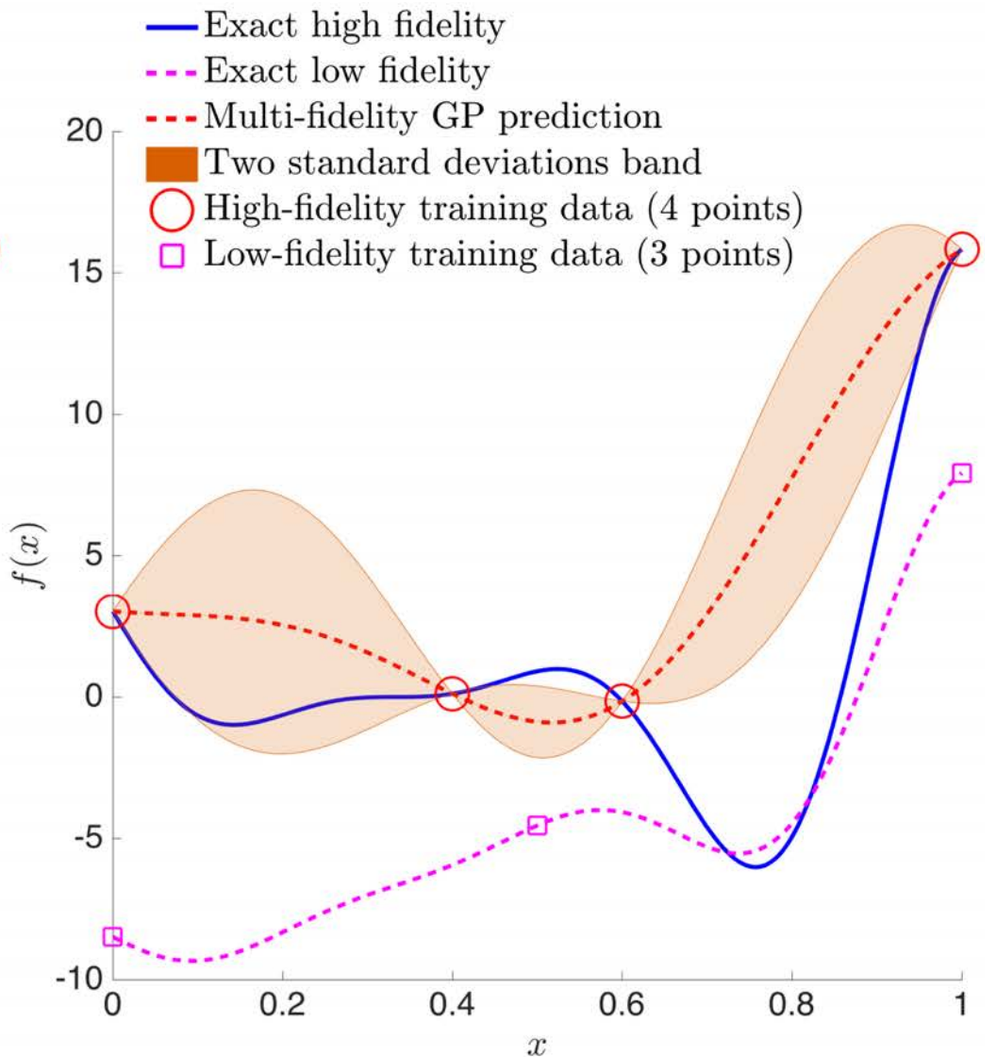
⇓

We cannot compute at all $(\boldsymbol{x};\boldsymbol{\xi})$

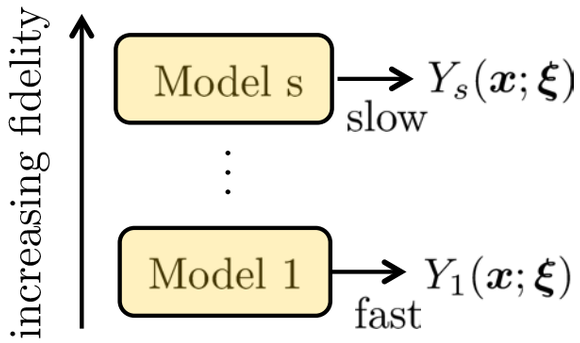
⇓

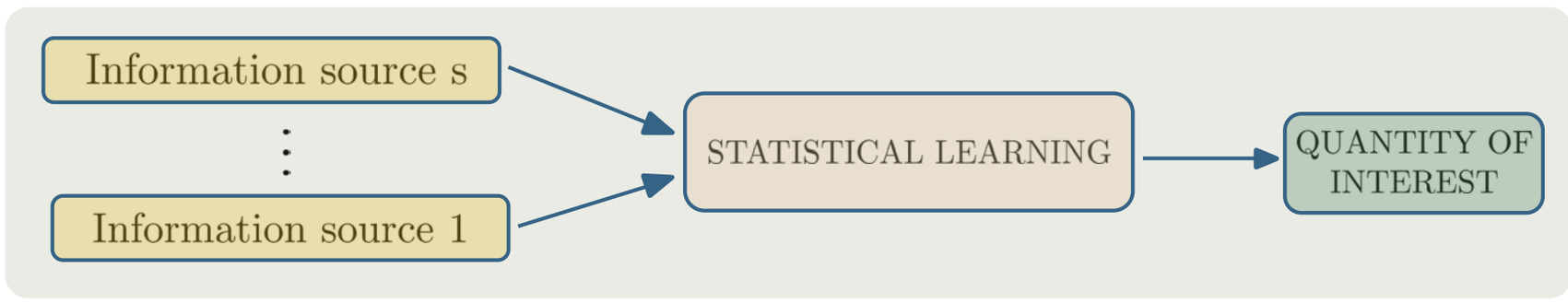
Prediction of $Z_i(\boldsymbol{x}) = \mathbb{E}[f(Y_i(\boldsymbol{x};\boldsymbol{\xi}))]$ is a problem of statistical inference





Multi-fidelity modeling





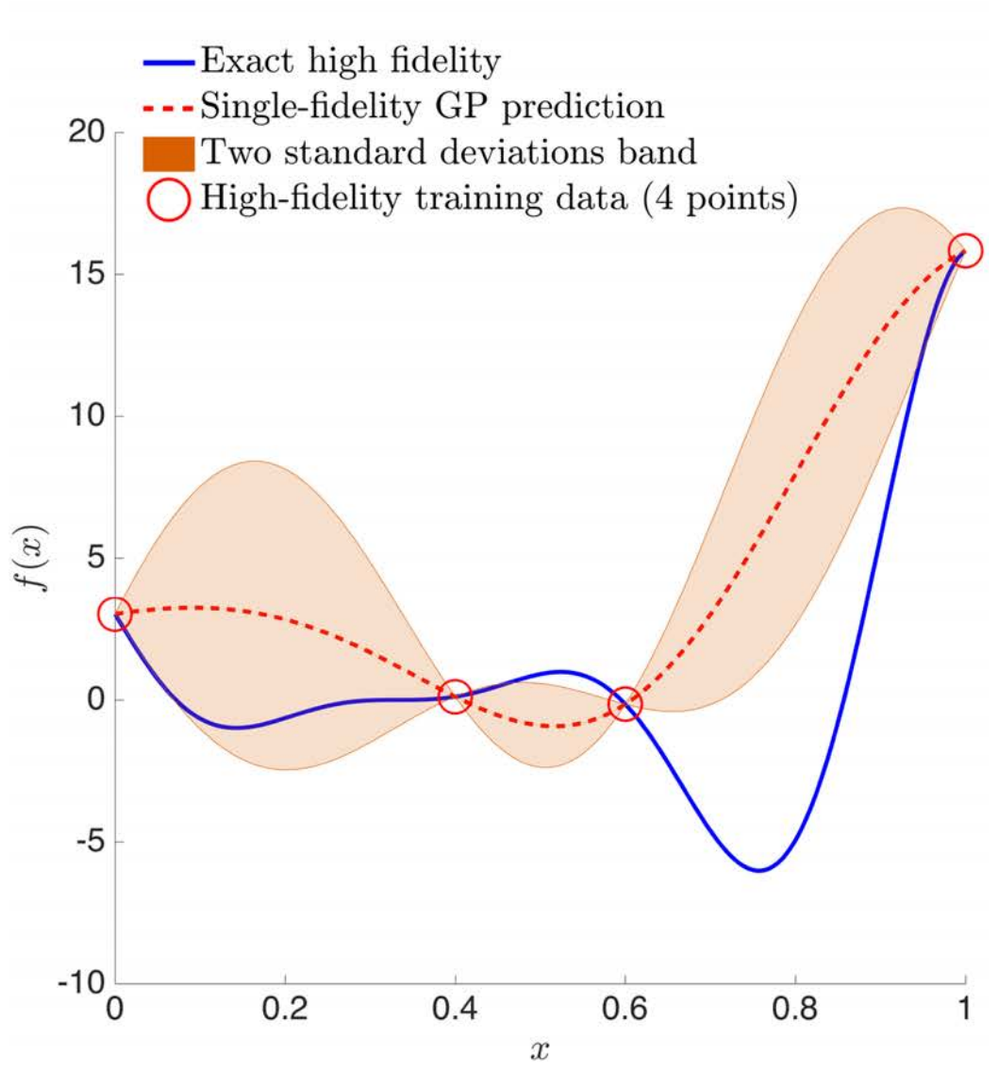
Number of runs is limited by time and computational resources

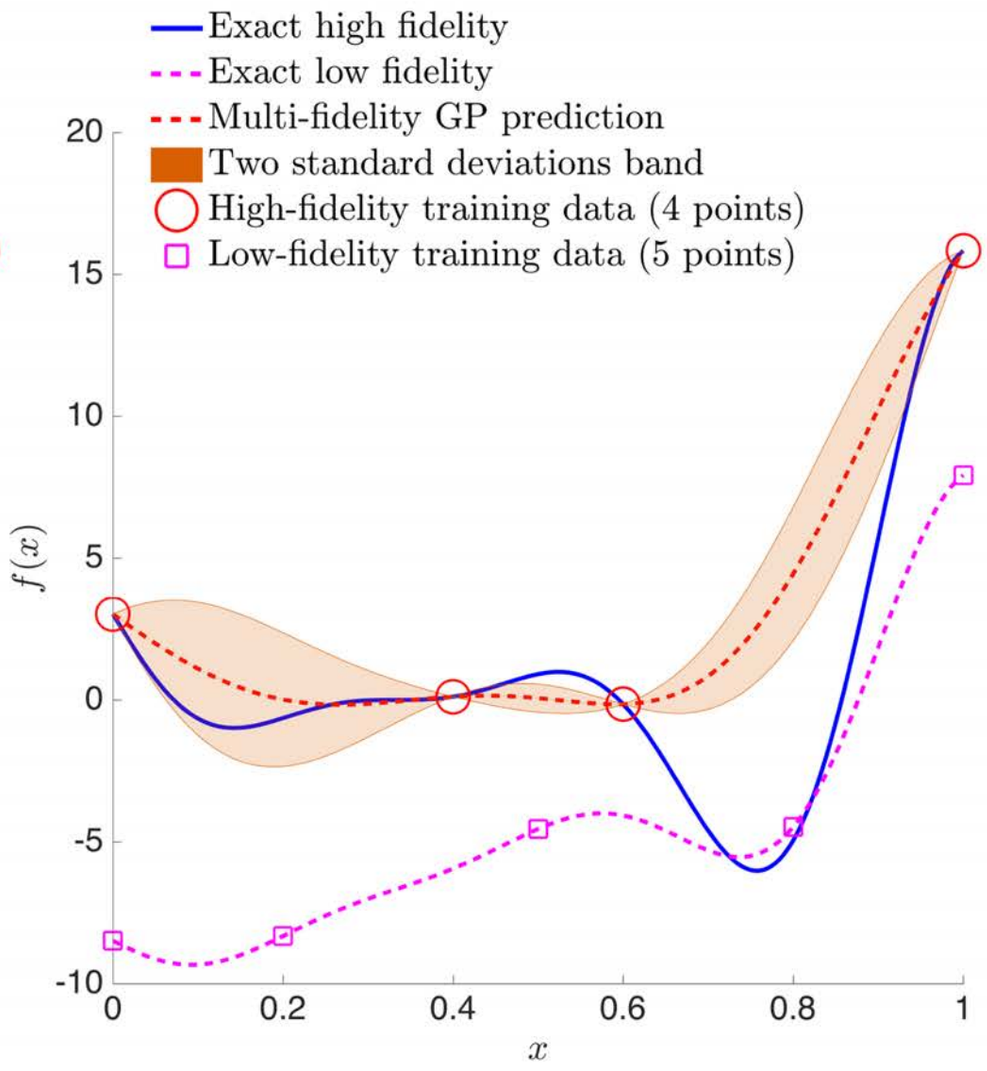
⇓

We cannot compute at all $(\boldsymbol{x};\boldsymbol{\xi})$

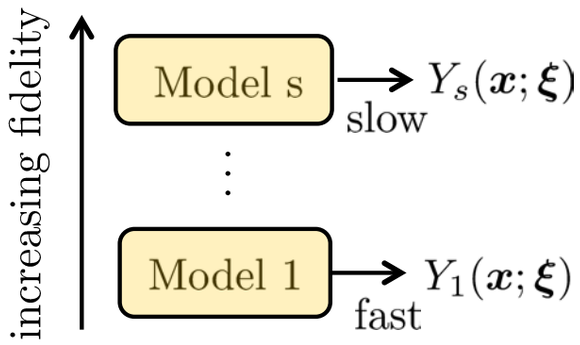
⇓

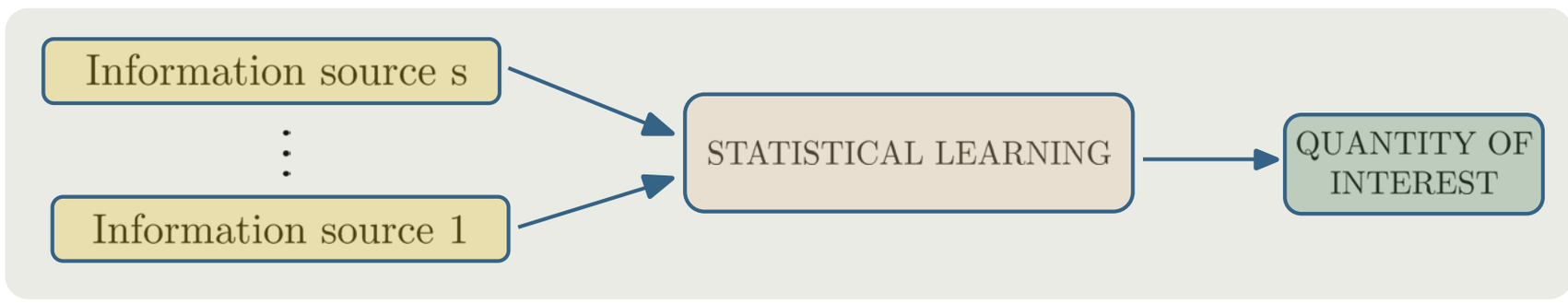
Prediction of $Z_i(\boldsymbol{x}) = \mathbb{E}[f(Y_i(\boldsymbol{x};\boldsymbol{\xi}))]$ is a problem of statistical inference





Multi-fidelity modeling





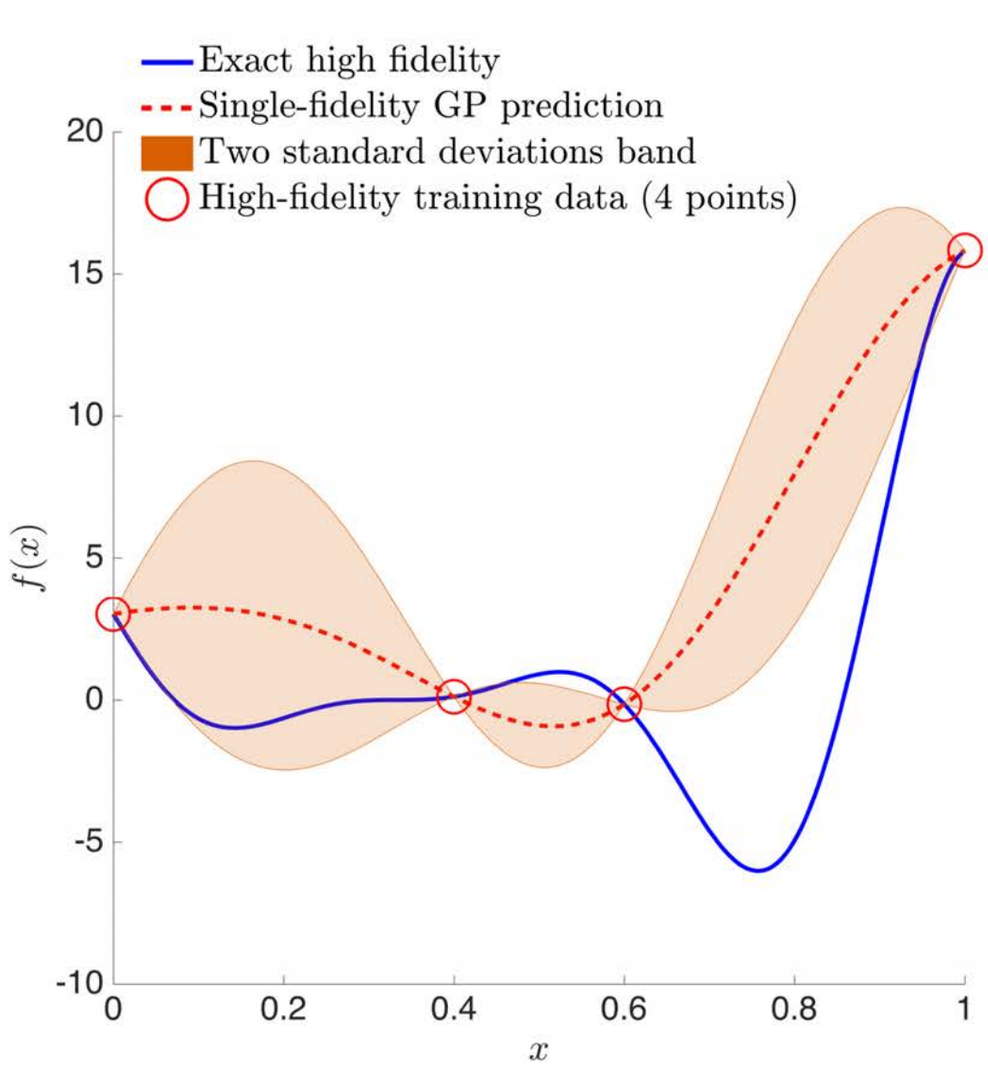
Number of runs is limited by time and computational resources

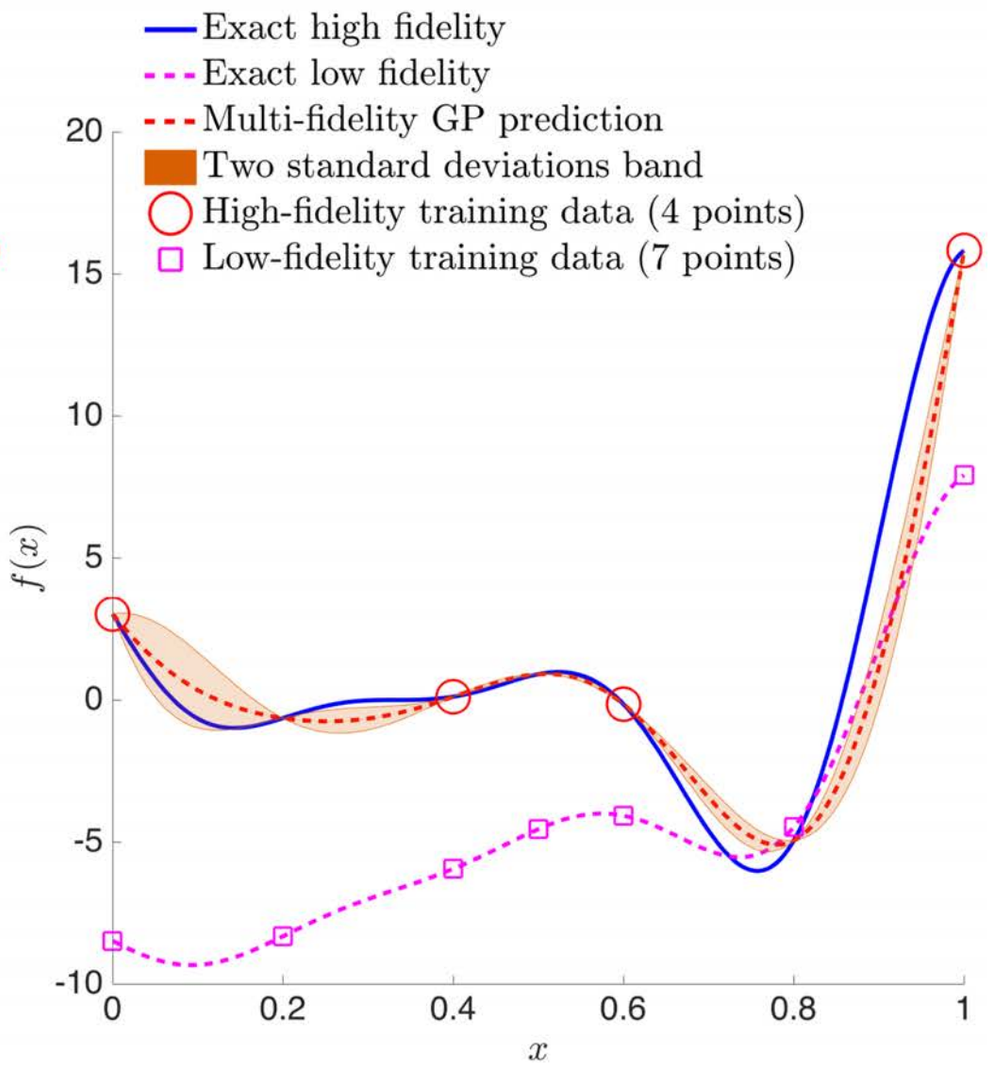
⇓

We cannot compute at all $(\boldsymbol{x};\boldsymbol{\xi})$

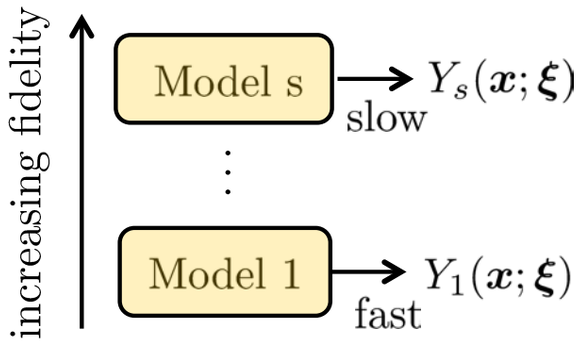
⇓

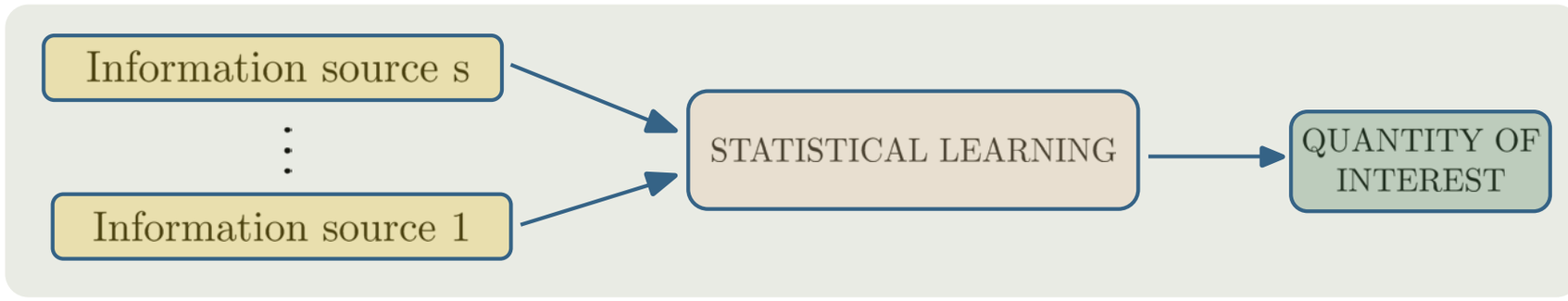
Prediction of $Z_i(\boldsymbol{x}) = \mathbb{E}[f(Y_i(\boldsymbol{x};\boldsymbol{\xi}))]$ is a problem of statistical inference





Multi-fidelity modeling





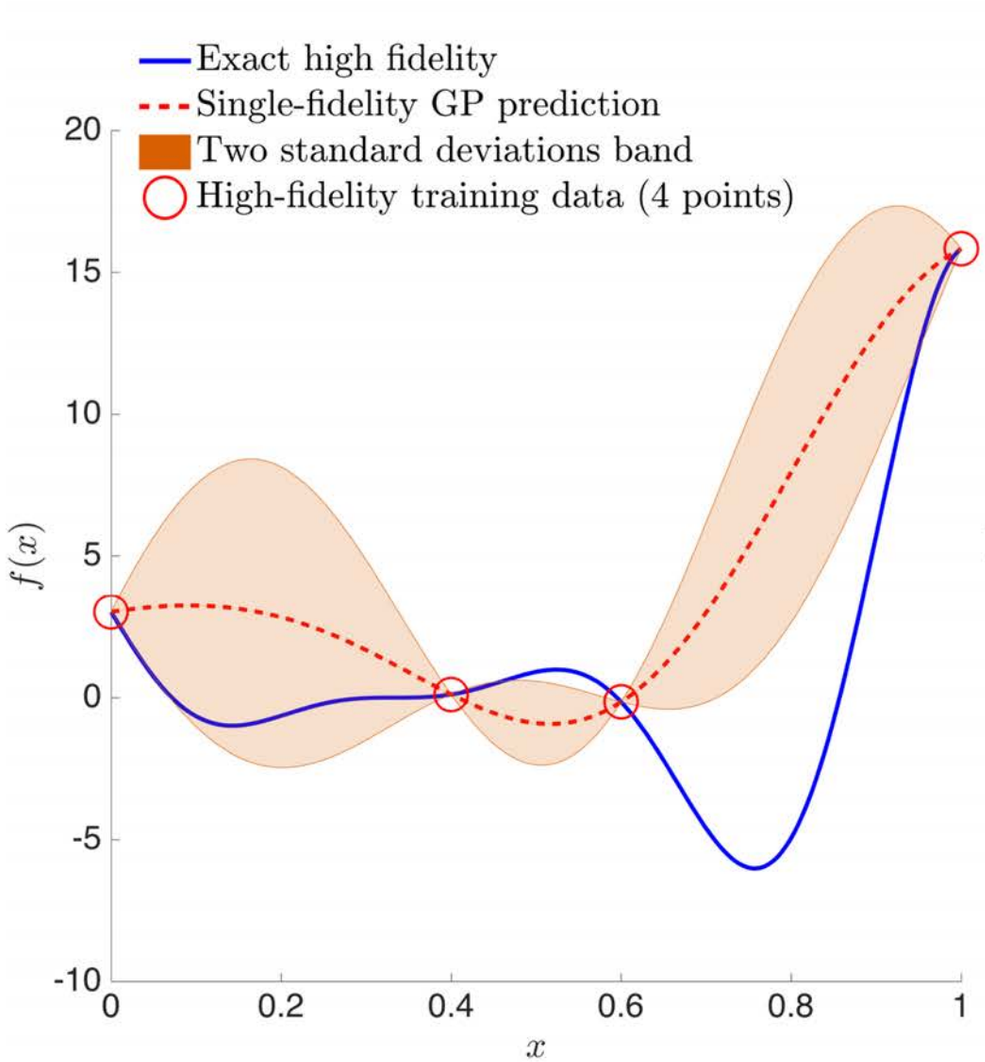
Number of runs is limited by time and computational resources

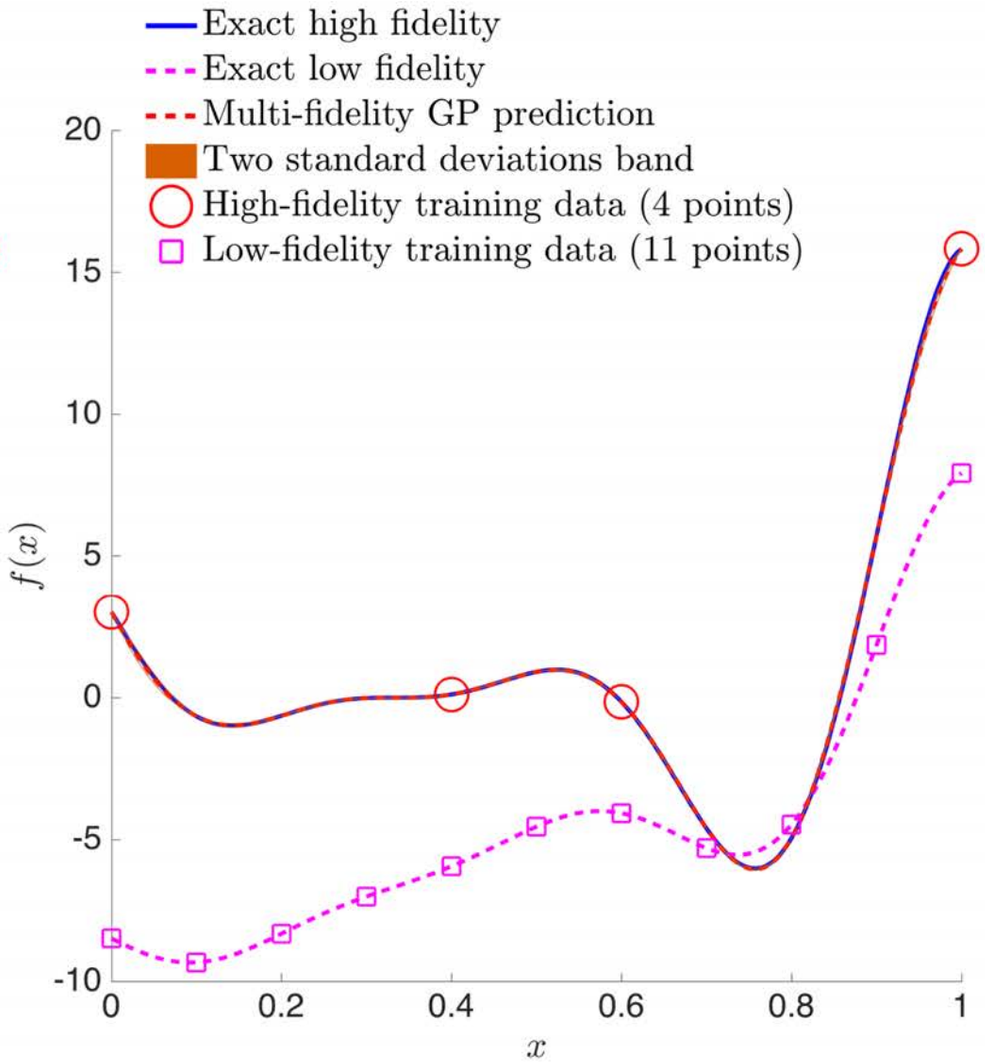
⇓

We cannot compute at all $(\boldsymbol{x};\boldsymbol{\xi})$

⇓

Prediction of $Z_i(\boldsymbol{x}) = \mathbb{E}[f(Y_i(\boldsymbol{x};\boldsymbol{\xi}))]$ is a problem of statistical inference





Multi-fidelity modeling with GPs

Auto-regressive model: AR1

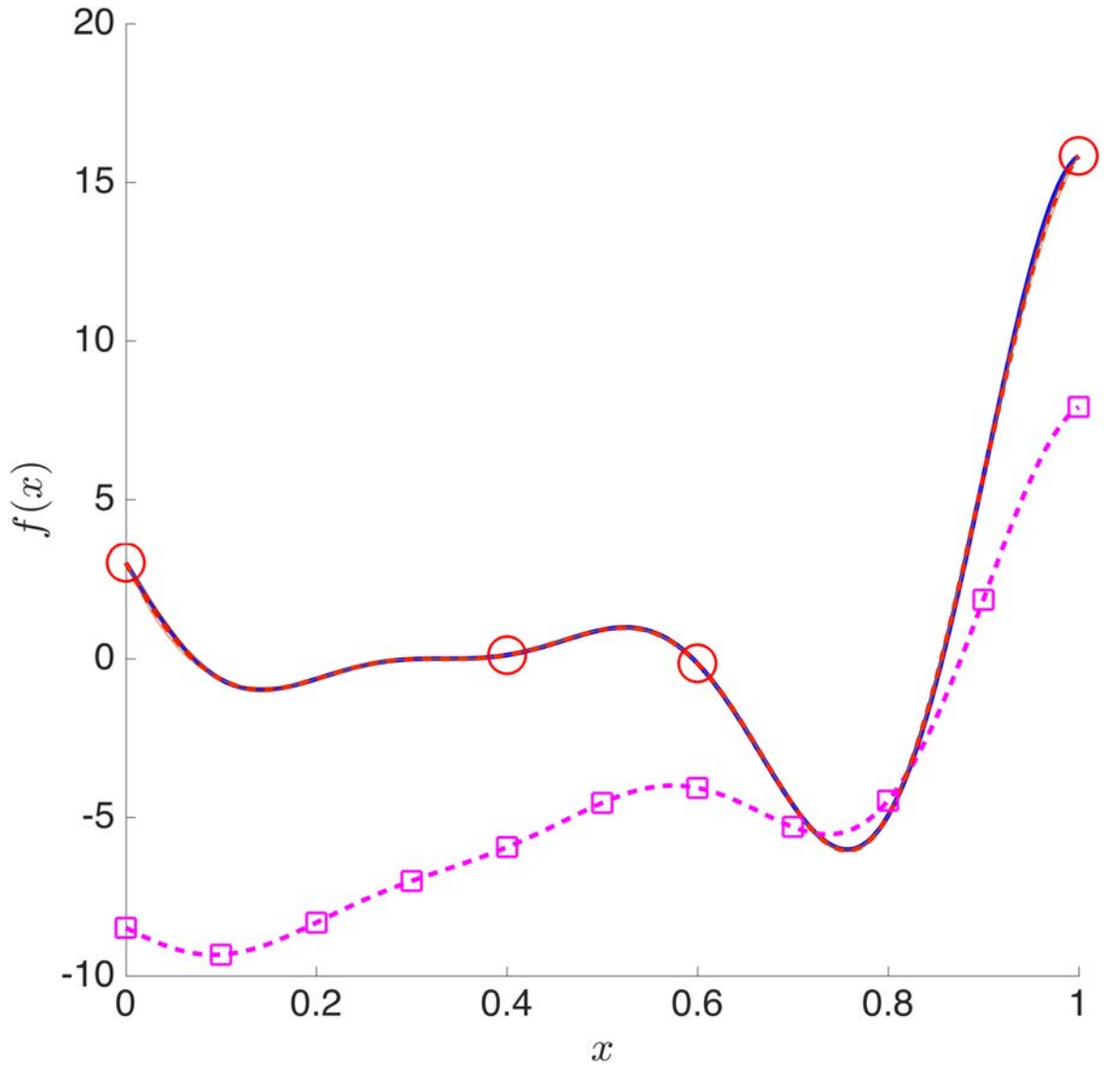
$$f_t(\mathbf{x}) = \rho_{t-1}(\mathbf{x}) f_{t-1}(\mathbf{x}) + \delta_t(\mathbf{x})$$

$$t = 1, \ldots, s$$

Predicting the Output from a Complex Computer Code When Fast Approximations Are Available

M. C. Kennedy; A. O'Hagan

Biometrika, Vol. 87, No. 1. (Mar., 2000), pp. 1-13.



Predictive posterior

$$p(f_*|\boldsymbol{y}, \boldsymbol{X}, \boldsymbol{x}_*) = \mathcal{N}(f_*|\mu_*, \sigma_*^2),$$

$$\mu_*(\boldsymbol{x}_*) = \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{y},$$

$$\sigma_*^2(\boldsymbol{x}_*) = \boldsymbol{k}_{**} - \boldsymbol{k}_{*N}(\boldsymbol{K} + \sigma_\epsilon^2 \boldsymbol{I})^{-1}\boldsymbol{k}_{N*},$$

$$\begin{matrix} N_1 \\ N_2 \end{matrix} \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}$$

Block covariance matrix

M.C Kennedy, and A. O'Hagan. *Predicting the output from a complex computer code when fast approximations are available,* 2000.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

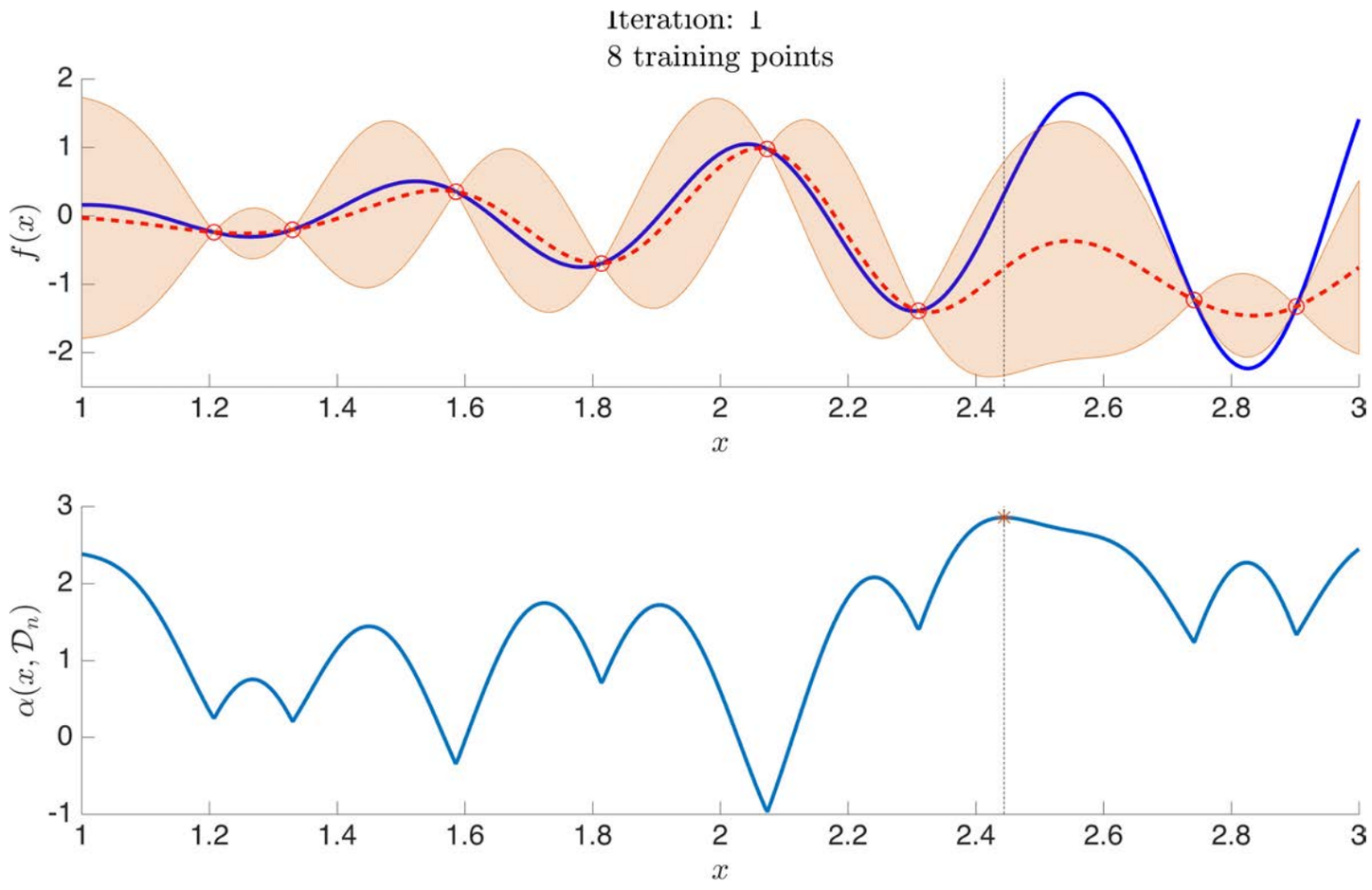
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

e.g. sample at the locations that maximize the expected improvement

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

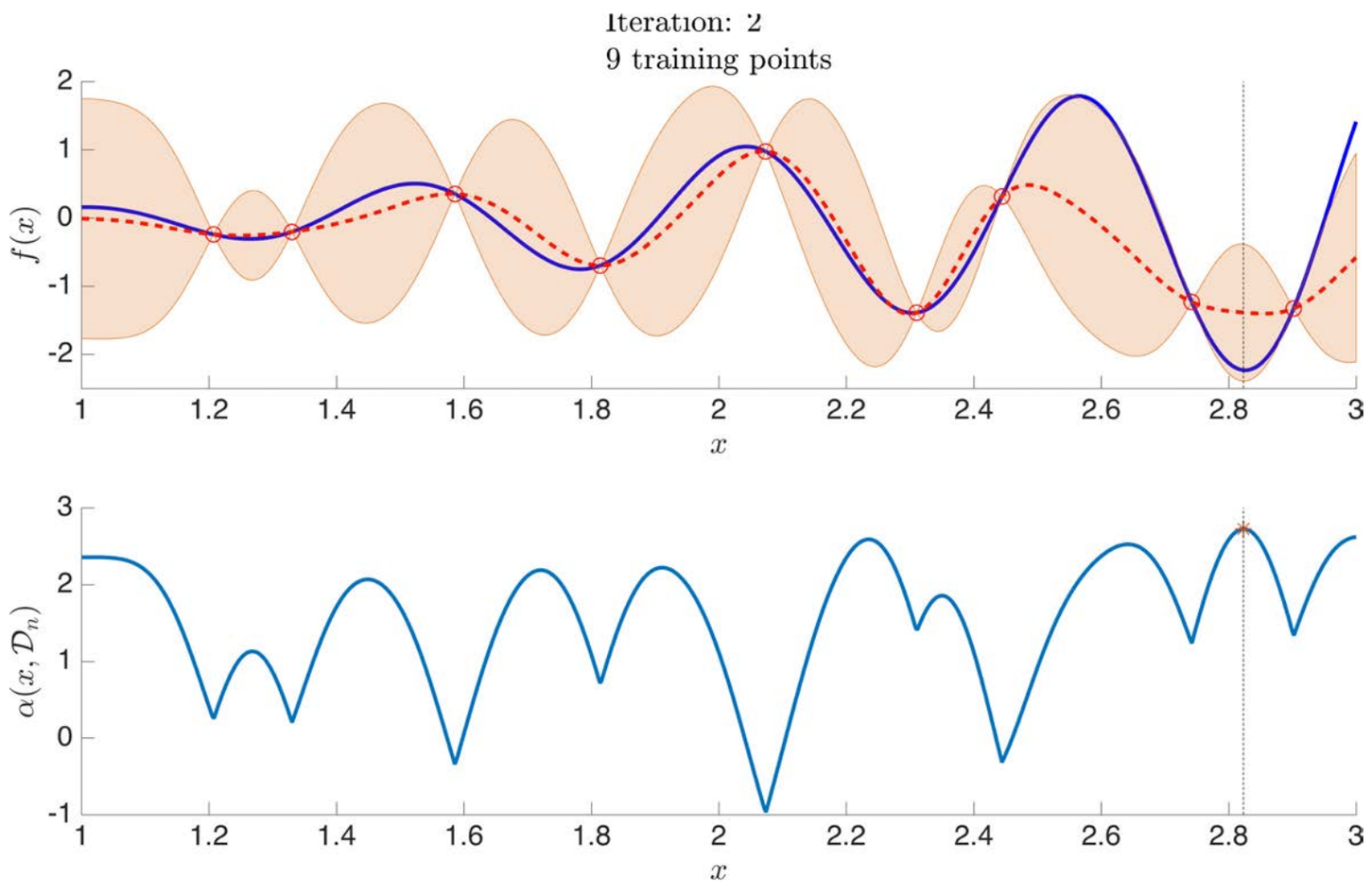
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

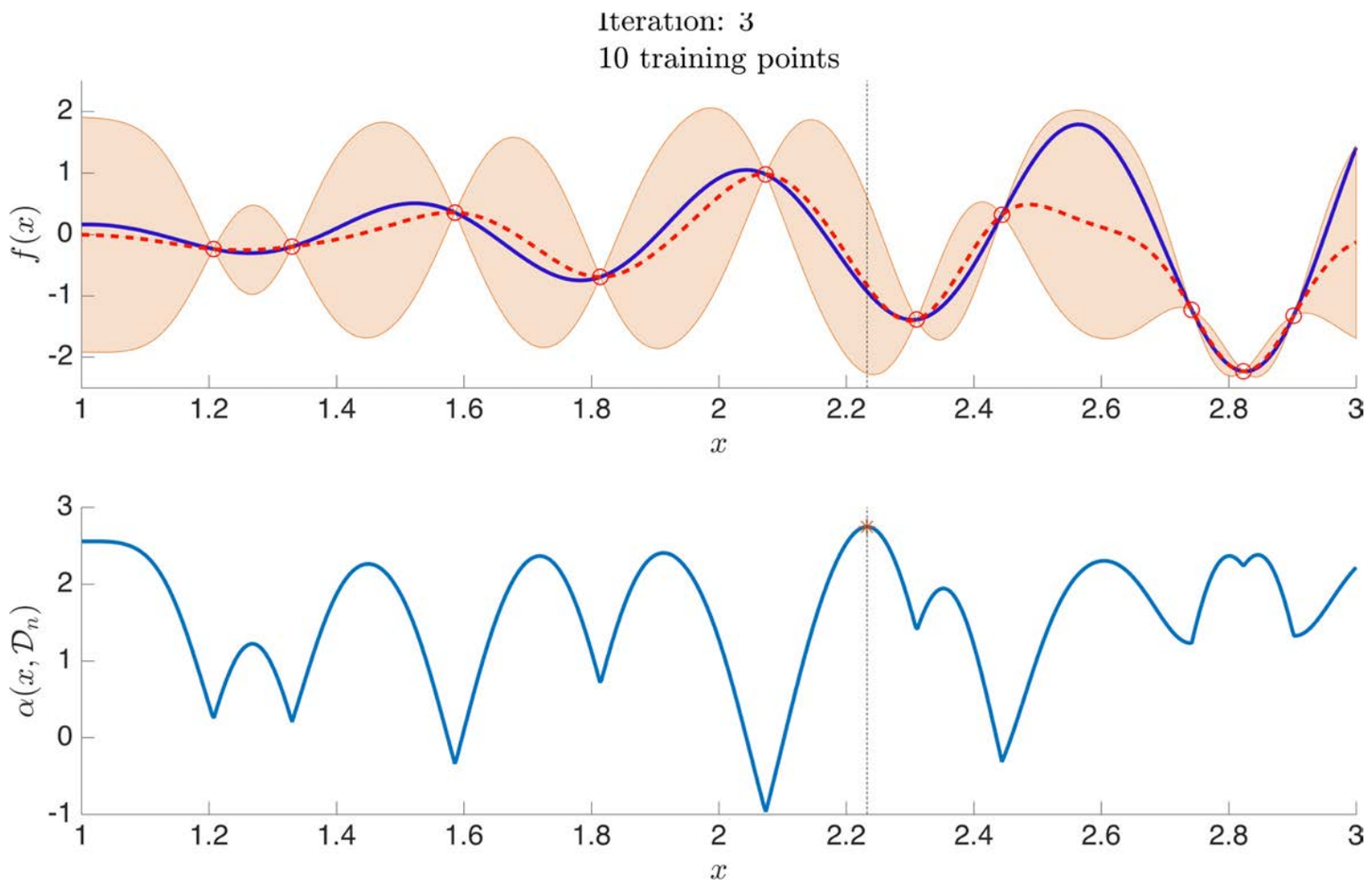
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

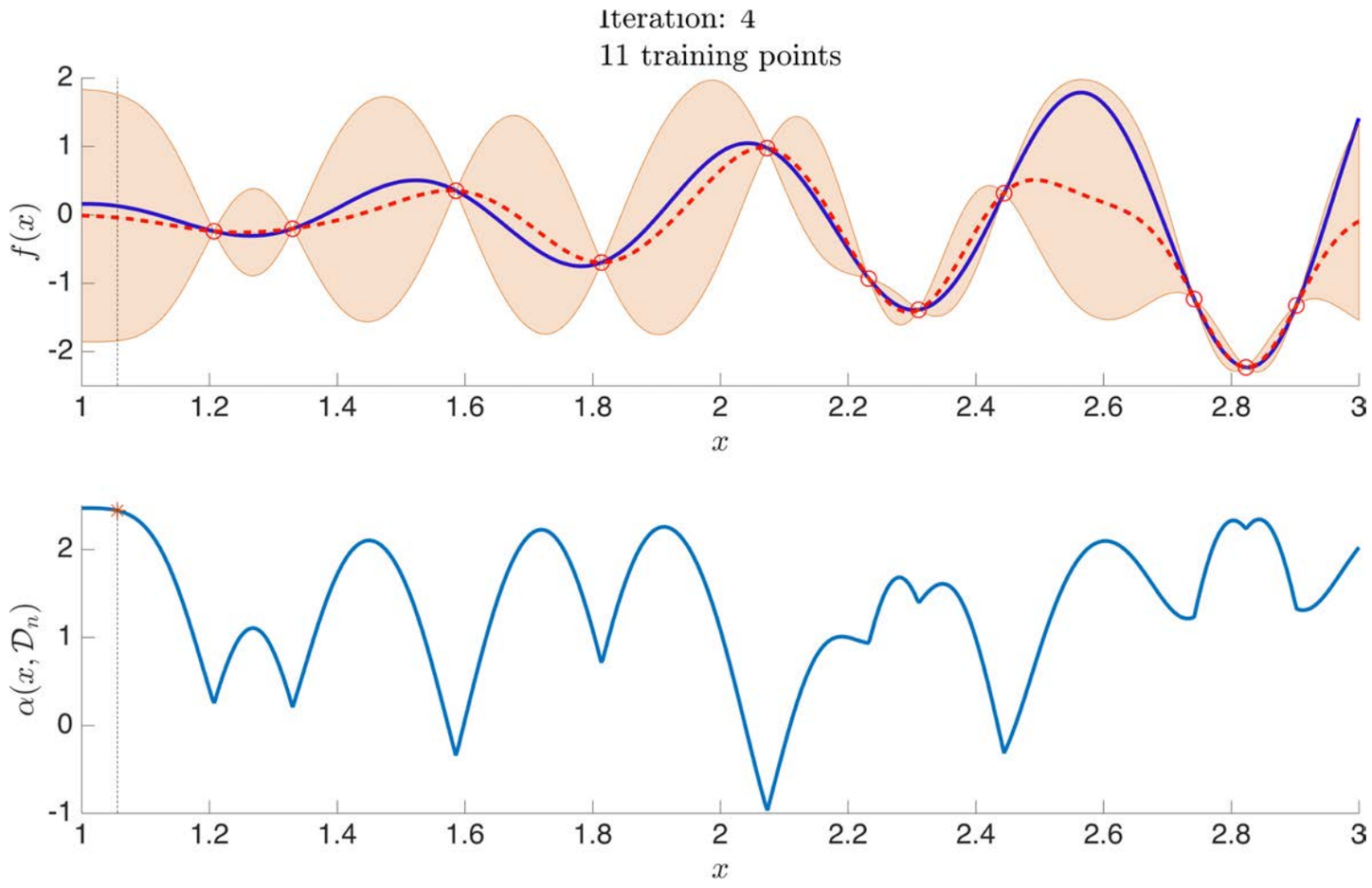
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

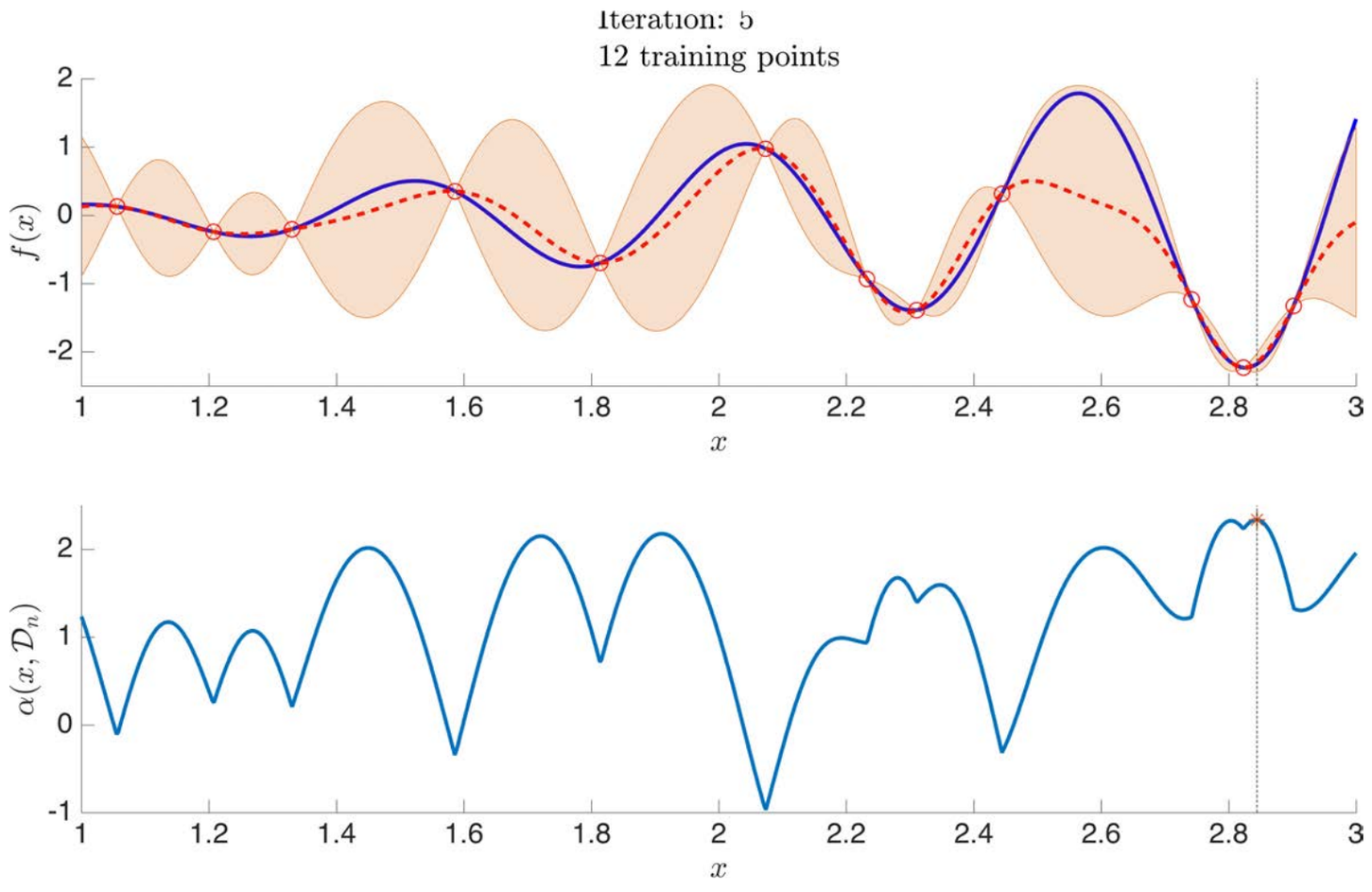
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

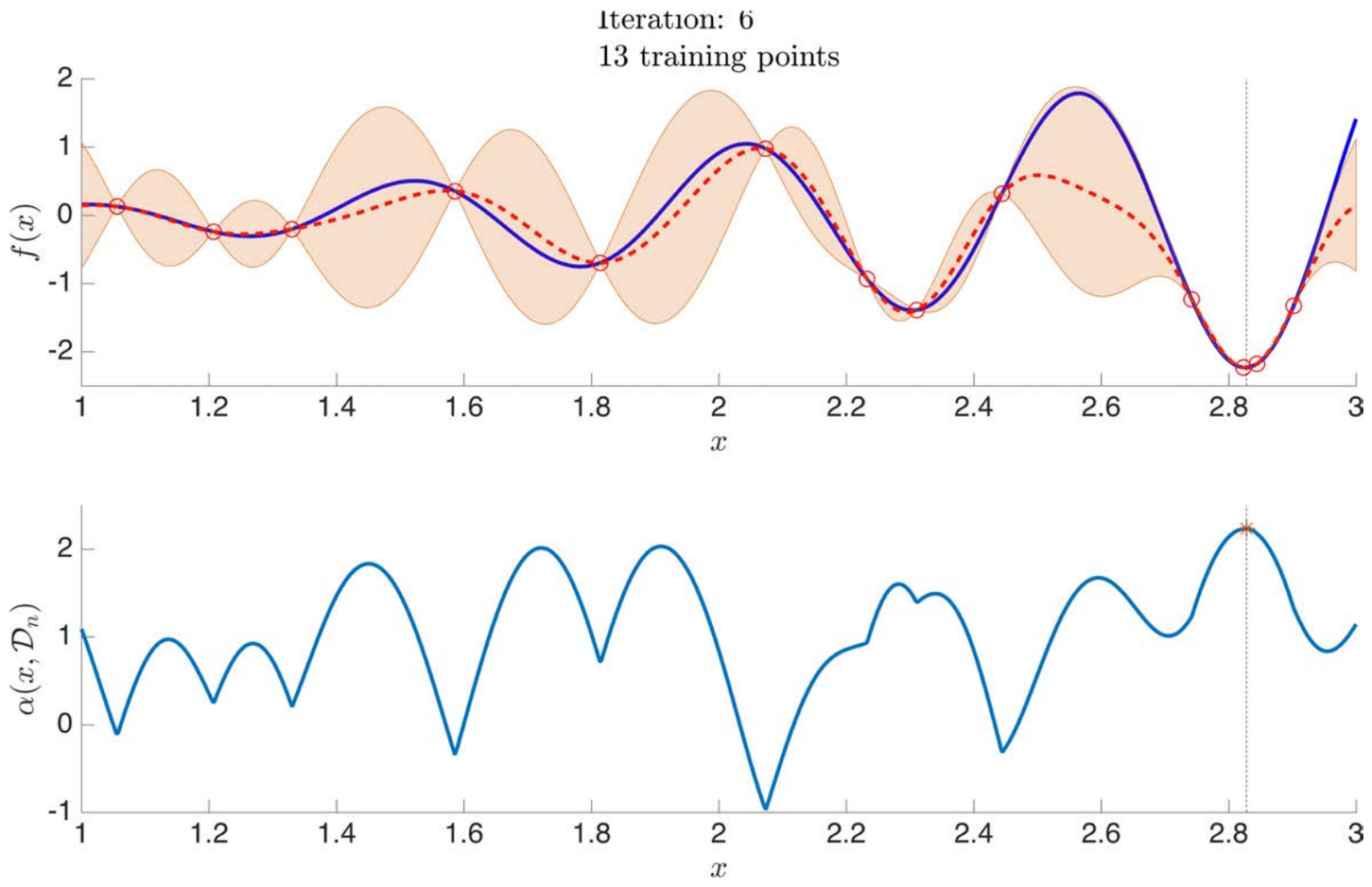
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Bayesian optimization

Goal: Estimate the global minimum of a function: $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^d} g(\mathbf{x})$ (potentially intractable)

Setup: g(x) is a black-box and expensive to evaluate objective function, noisy observations, no gradients.

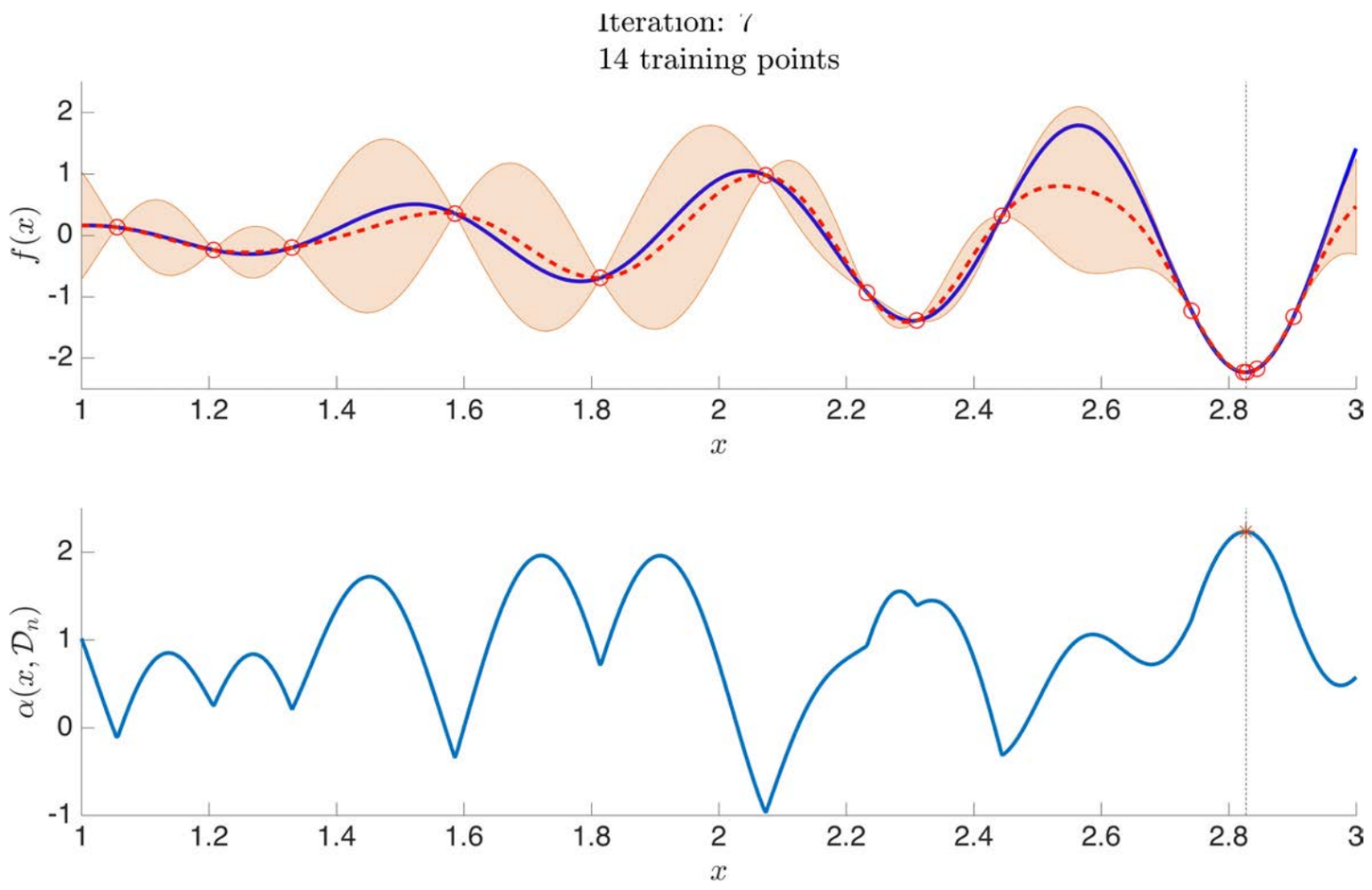
Idea: Approximate g(x) using a GP surrogate: $y = f(\mathbf{x}) + \epsilon, \quad f \sim \mathcal{GP}\left(f|0, k(\mathbf{x}, \mathbf{x}'; \theta)\right)$

Utilize the posterior to guide a sequential or parallel sampling policy by optimizing a chosen expected utility function

$$\alpha(\mathbf{x}; \mathcal{D}_n) = \mathbb{E}_\theta \mathbb{E}_{v\,|\,\mathbf{x},\theta}[U(\mathbf{x}, v, \theta)]$$

The optimization problem is transformed to:

$$\mathbf{x}_{n+1} = \arg\max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$



Remark:

Acquisition functions aim to balance the trade-off between exploration and exploitation.

Shahriari, Bobak, et al. "Taking the human out of the loop: A review of bayesian optimization." Proceedings of the IEEE 104.1 (2016): 148-175.

Some software packages

Gaussian processes:
http://www.gaussianprocess.org/gpml/
https://github.com/SheffieldML/GPy
https://github.com/GPflow/GPflow

Automatic differentiation:
https://github.com/HIPS/autograd
https://www.tensorflow.org
http://deeplearning.net/software/theano/
http://pytorch.org

Bayesian optimization:
https://github.com/SheffieldML/GPyOpt
https://github.com/HIPS/Spearmint

Probabilistic programming:
http://edwardlib.org
http://mc-stan.org
https://pymc-devs.github.io/pymc3/index.html