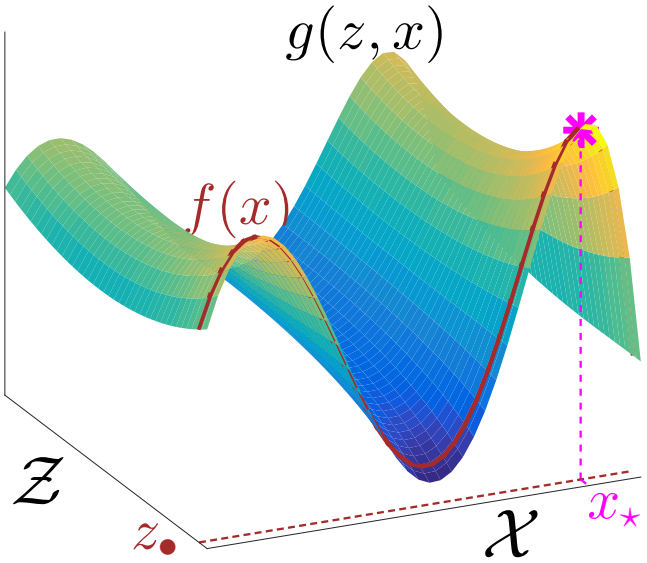
Multi-fidelity Bayesian Optimisation



Kirthevasan Kandasamy

Carnegie Mellon University

Facebook Inc. Menlo Park, CA

September 26, 2017

Slides: www.cs.cmu.edu/∼kkandasa/talks/fb-mf-slides.pdf

Slides are up on my website: www.cs.cmu.edu/∼kkandasa

www.cs.cmu.edu/~kkandasa/index.html

home research software misc.

kirthevasan kandasamy

PhD Student, Carnegie Mellon University

[CV] [Google Scholar] [GitHub] [Contact]

I am a fourth year Machine Learning PhD student (now ABD) in the School of Computer Science at Carnegie Mellon University. I am co-advised by Jeff Schneider and Barnabas Poczos. I am a member of the Auton Lab and the StatML Group. Prior to CMU, I completed my B.Sc in Electronics & Telecommunications Engineering at the University of Moratuwa, Sri Lanka.

My research interests lie in the intersection of statistical and algorithmic Machine Learning. My current research spans bandit problems, Bayesian optimisation, Gaussian processes, nonparametric statistics and graphical models. As of late, I have also hopped on the deep learning bandwagon. For more details, see my publications.

I am generously supported by a Facebook PhD fellowship (2017) and a CMU Presidential fellowship (2015).

Recent updates

Sep 26: Talk at Facebook on Multi-fidelity Bayesian Optimisation [slides]
Sep 27: Talk at Google on Parallelised Thompson Sampling [slides]

Slides

Contact

GHC 8213
Machine Learning Department
School of Computer Science

x

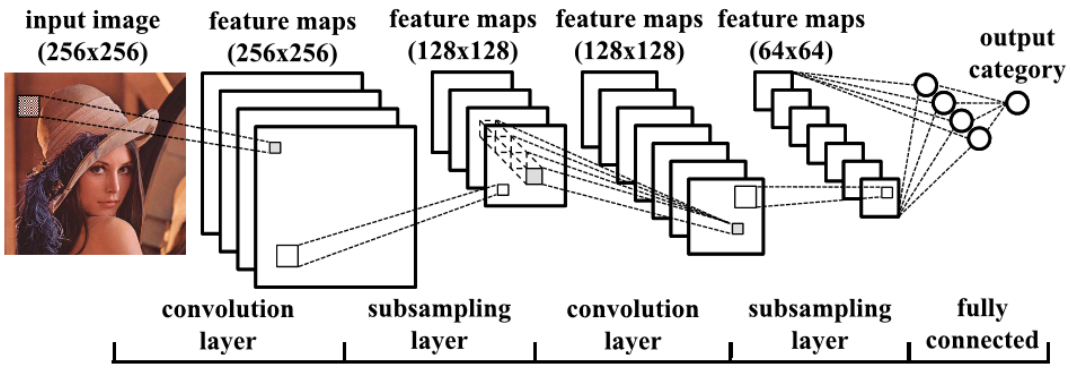
hyper-parameters

Neural Network

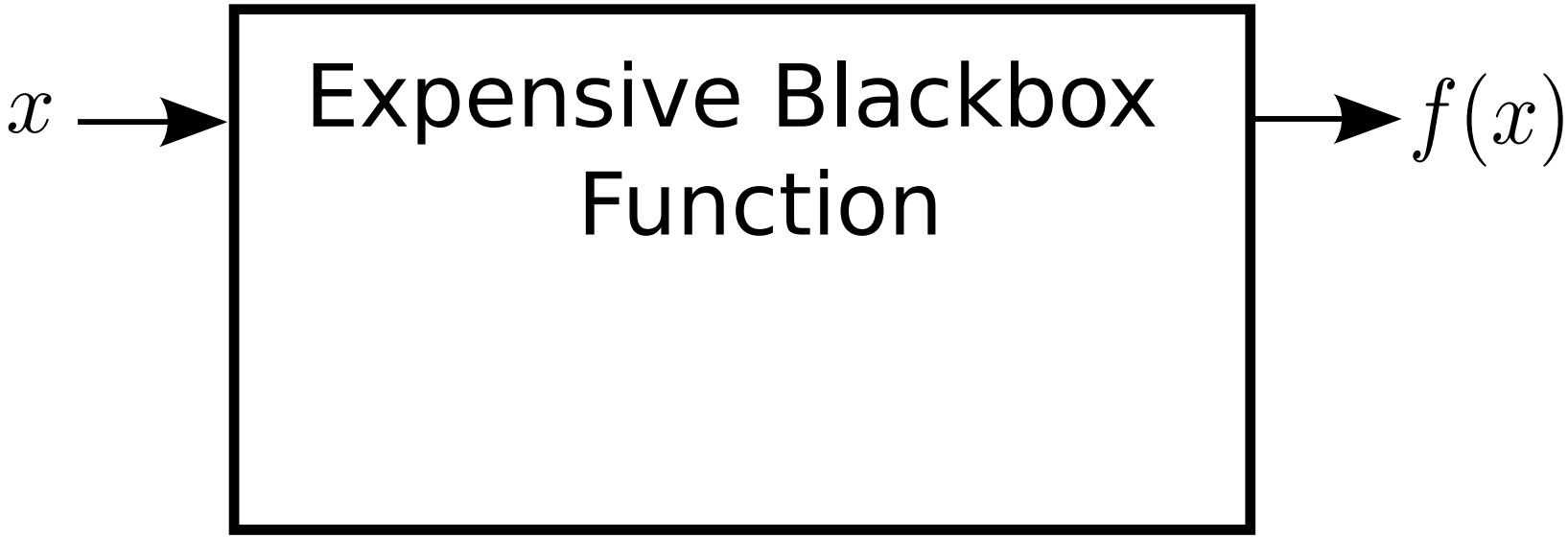
- Train NN using given hyper-parameters
- Compute accuracy on validation set

$f(x)$

cross validation accuracy



Black-box Optimisation



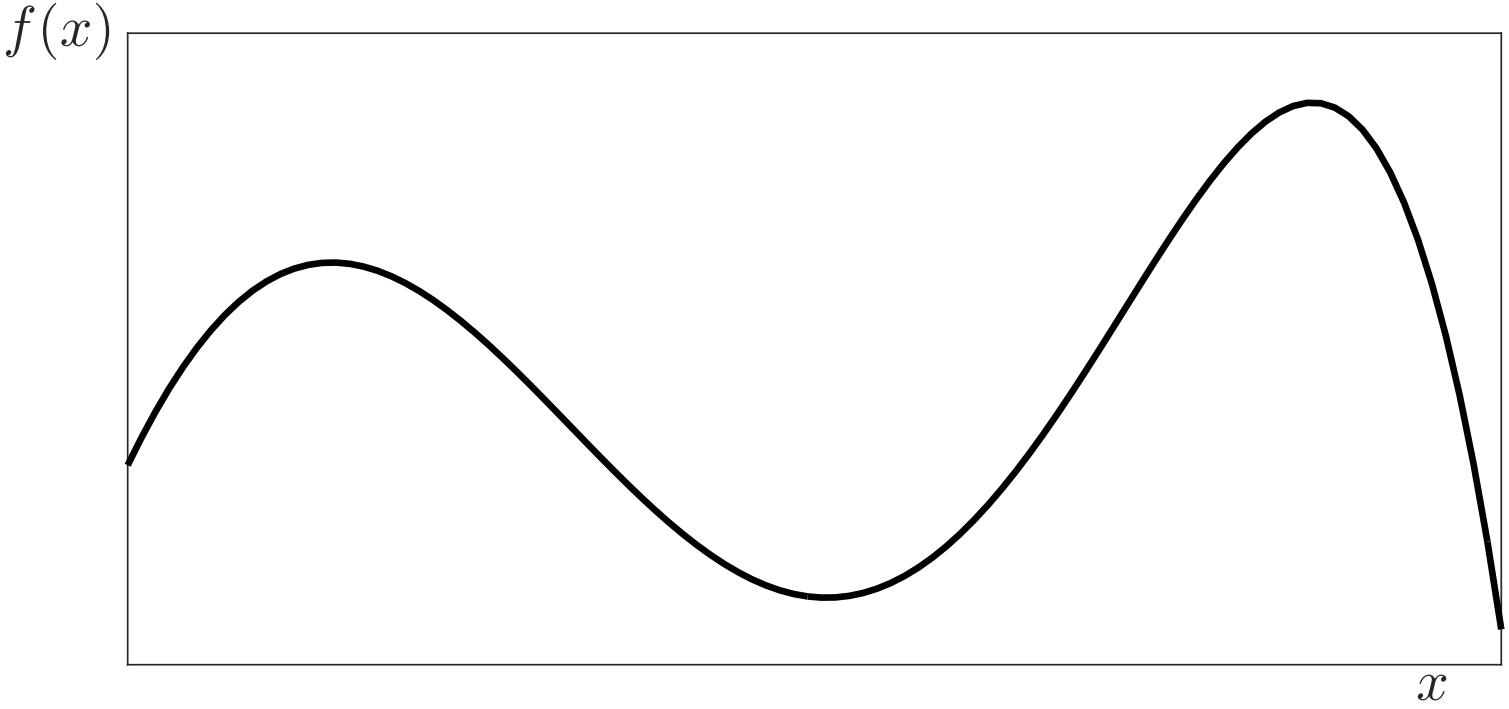
Black-box Optimisation

$x \longrightarrow$ **Expensive Blackbox Function** $\longrightarrow f(x)$

Other Examples:
- ML estimation in astrophysics
- Pre-clinical drug discovery
- Optimal policy in autonomous driving

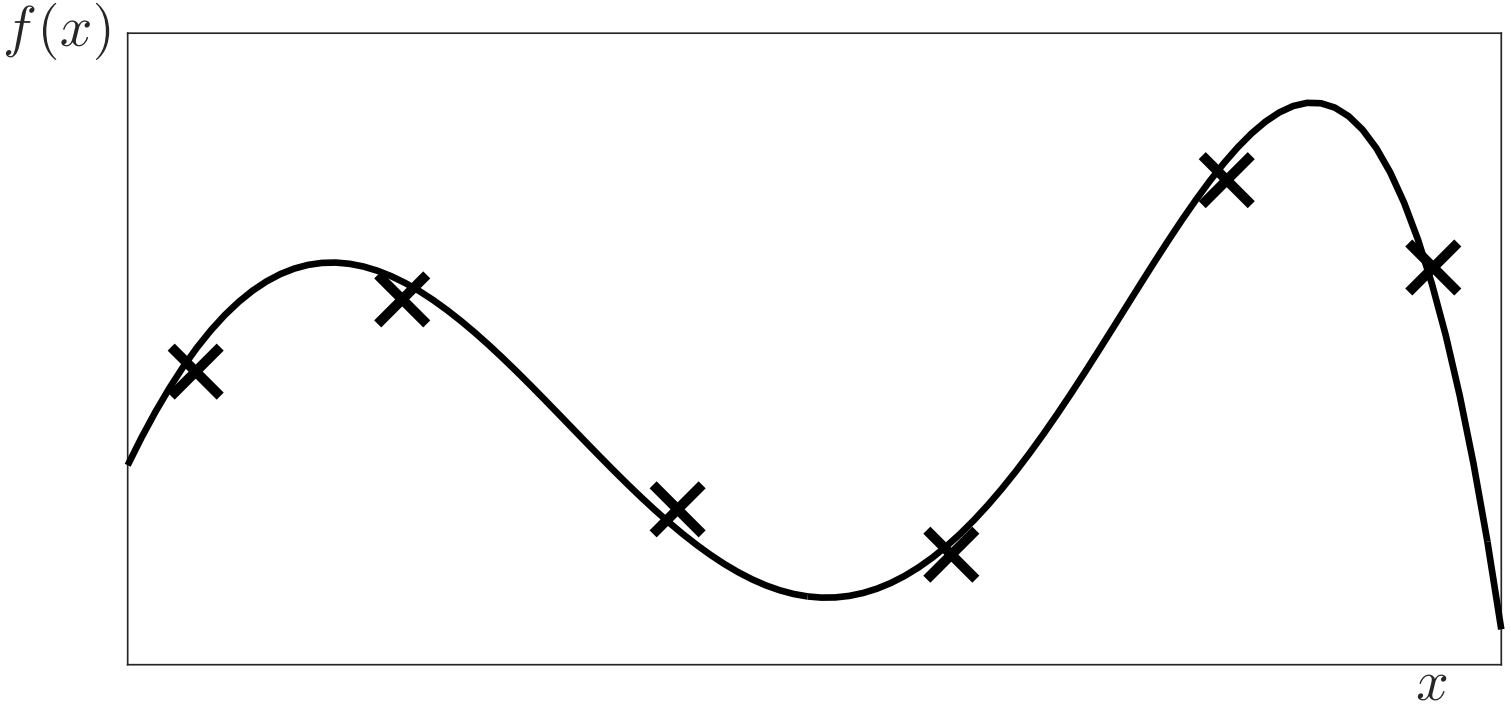
Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.



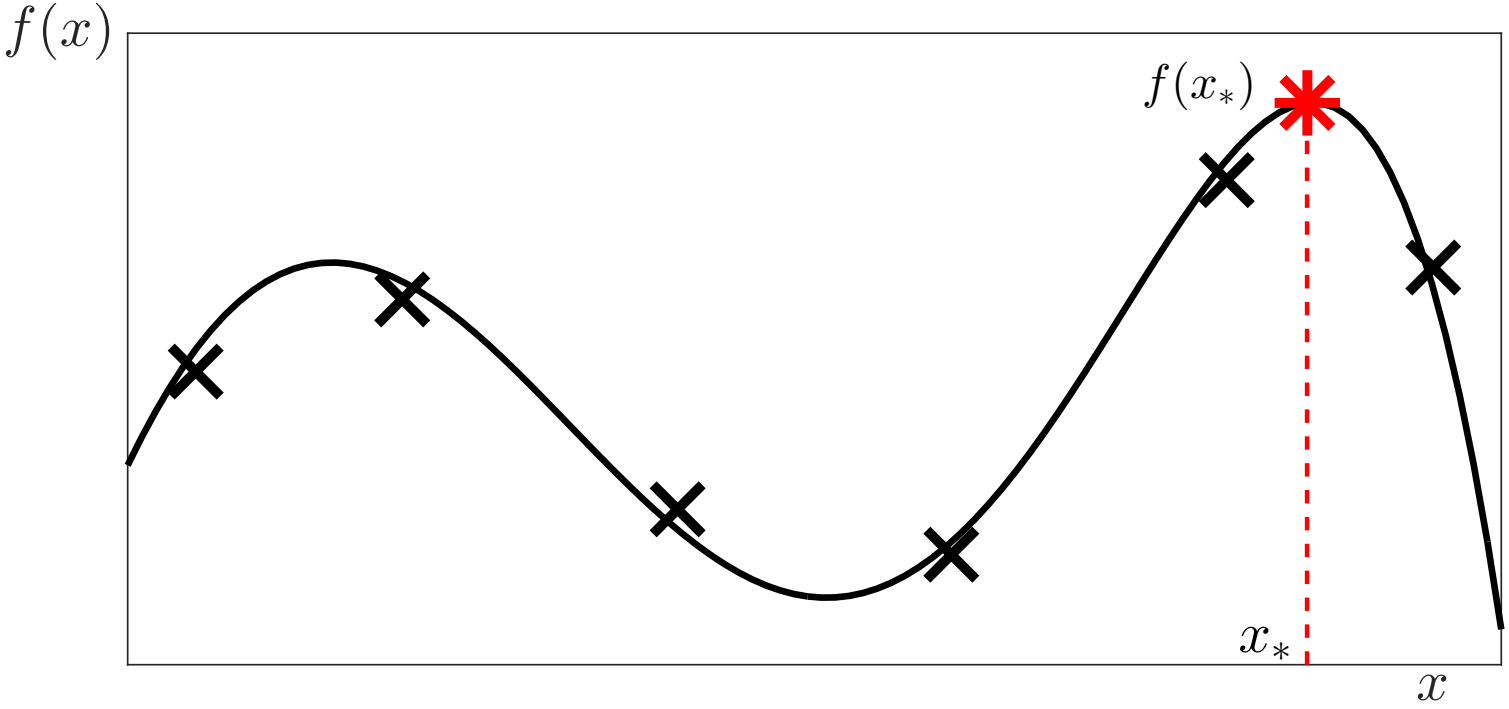
Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.



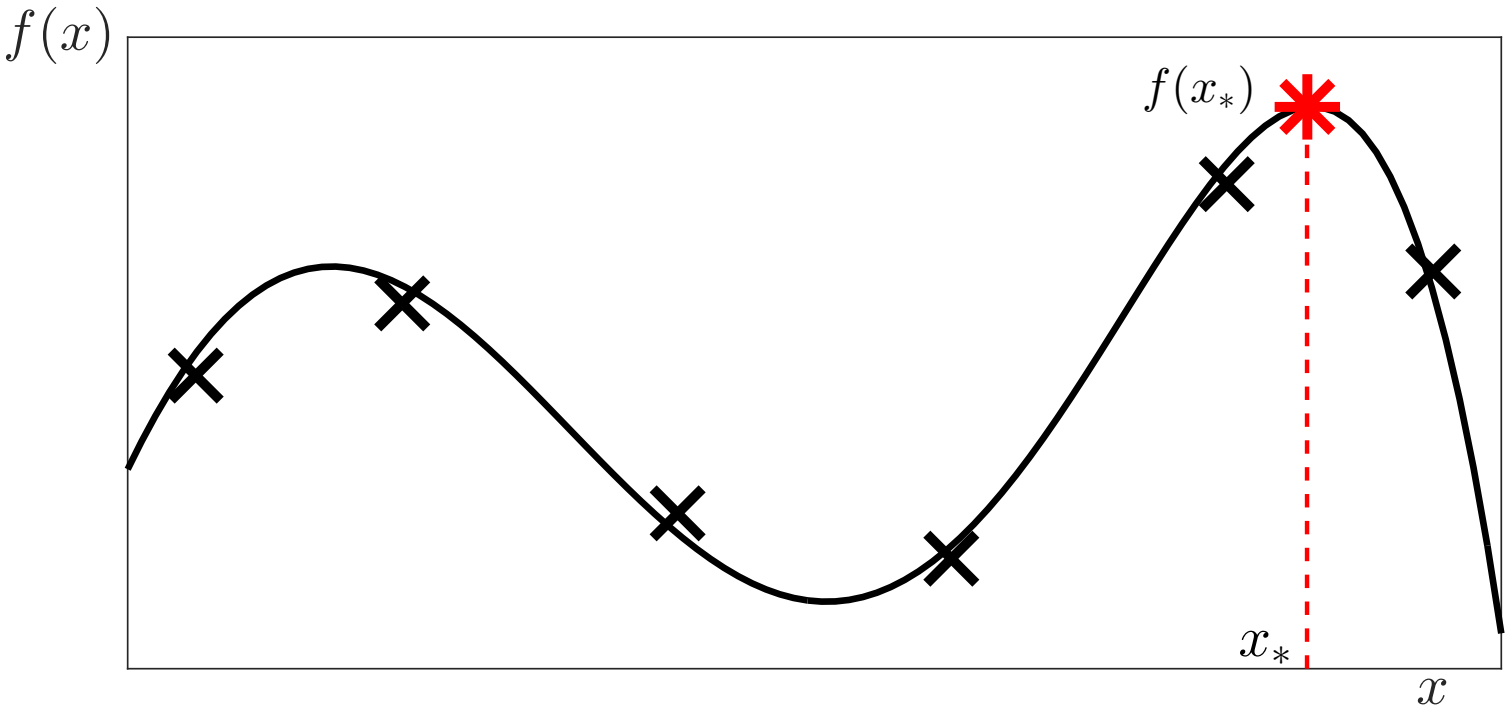
Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.
Let $x_\star = \mathrm{argmax}_x f(x)$.



Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.
Let $x_\star = \operatorname{argmax}_x f(x)$.

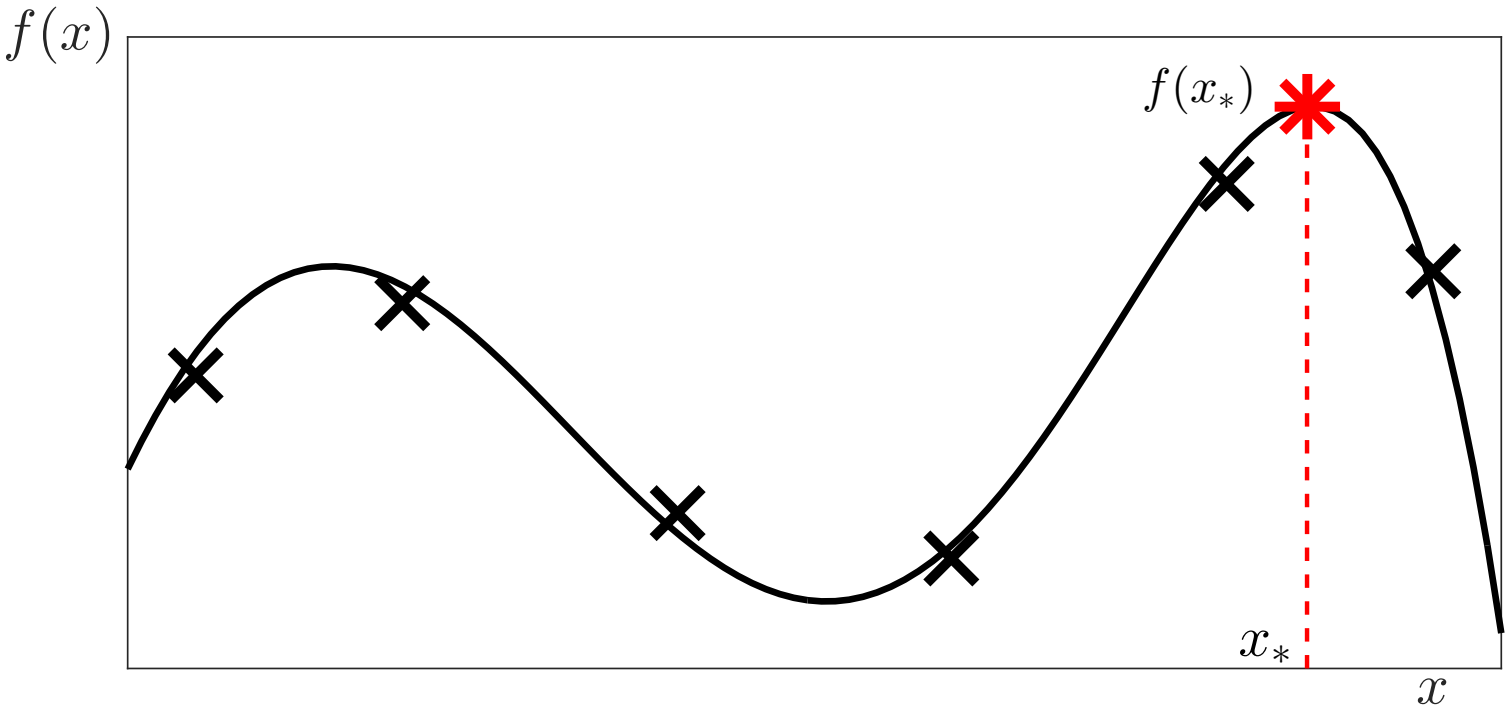


Simple Regret after n evaluations

$$S_n = f(x_\star) - \max_{t=1,\dots,n} f(x_t).$$

Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.
Let $x_\star = \operatorname{argmax}_x f(x)$.

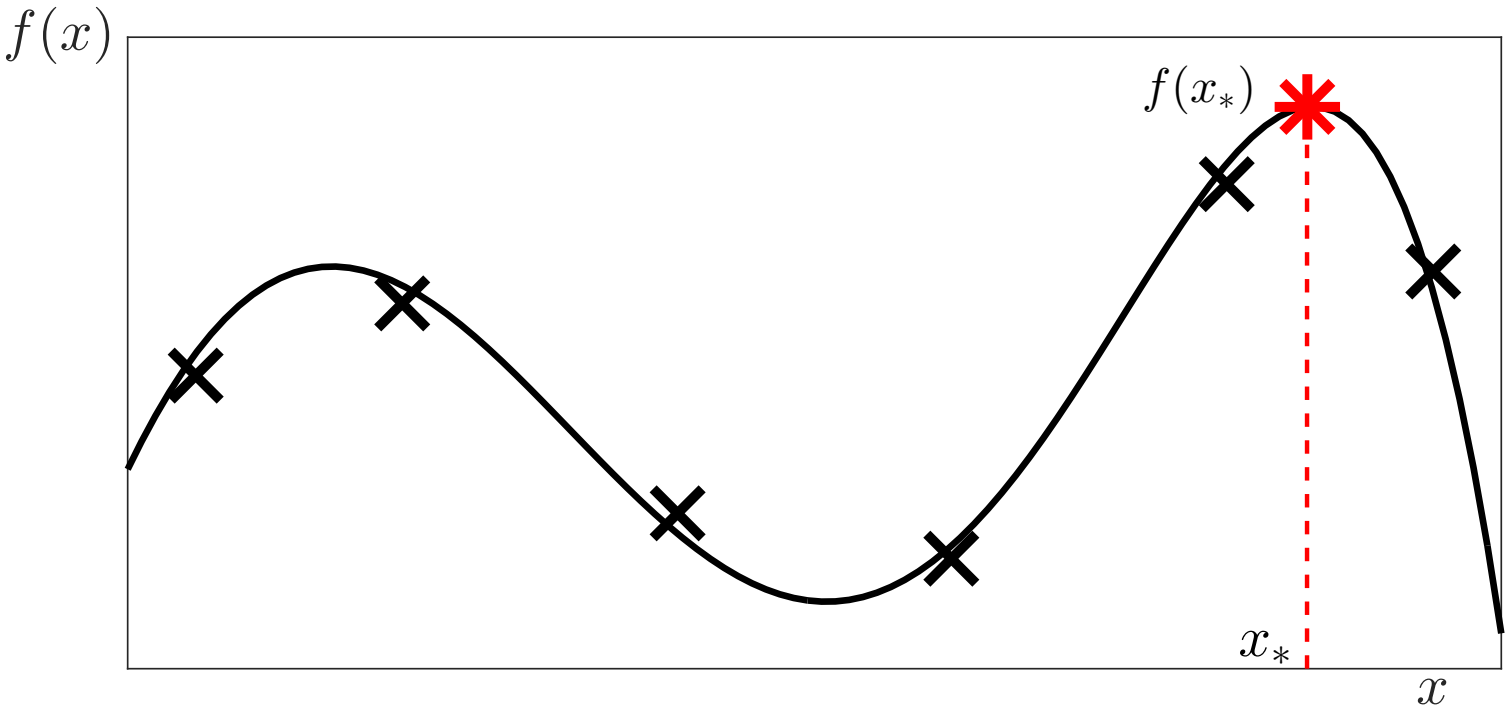


Cumulative Regret after n evaluations

$$R_n = \sum_{t=1}^{n} \big(f(x_\star) - f(x_t)\big).$$

Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ is an expensive, black-box, noisy function, accessible only via noisy evaluations.
Let $x_\star = \operatorname{argmax}_x f(x)$.



Simple Regret after n evaluations

$$S_n = f(x_\star) - \max_{t=1,\dots,n} f(x_t).$$

A walk-through Bayesian Optimisation with Gaussian Processes

- Gaussian Processes (GPs)
- GP-UCB: An algorithm for Bayesian Optimisation (BO)

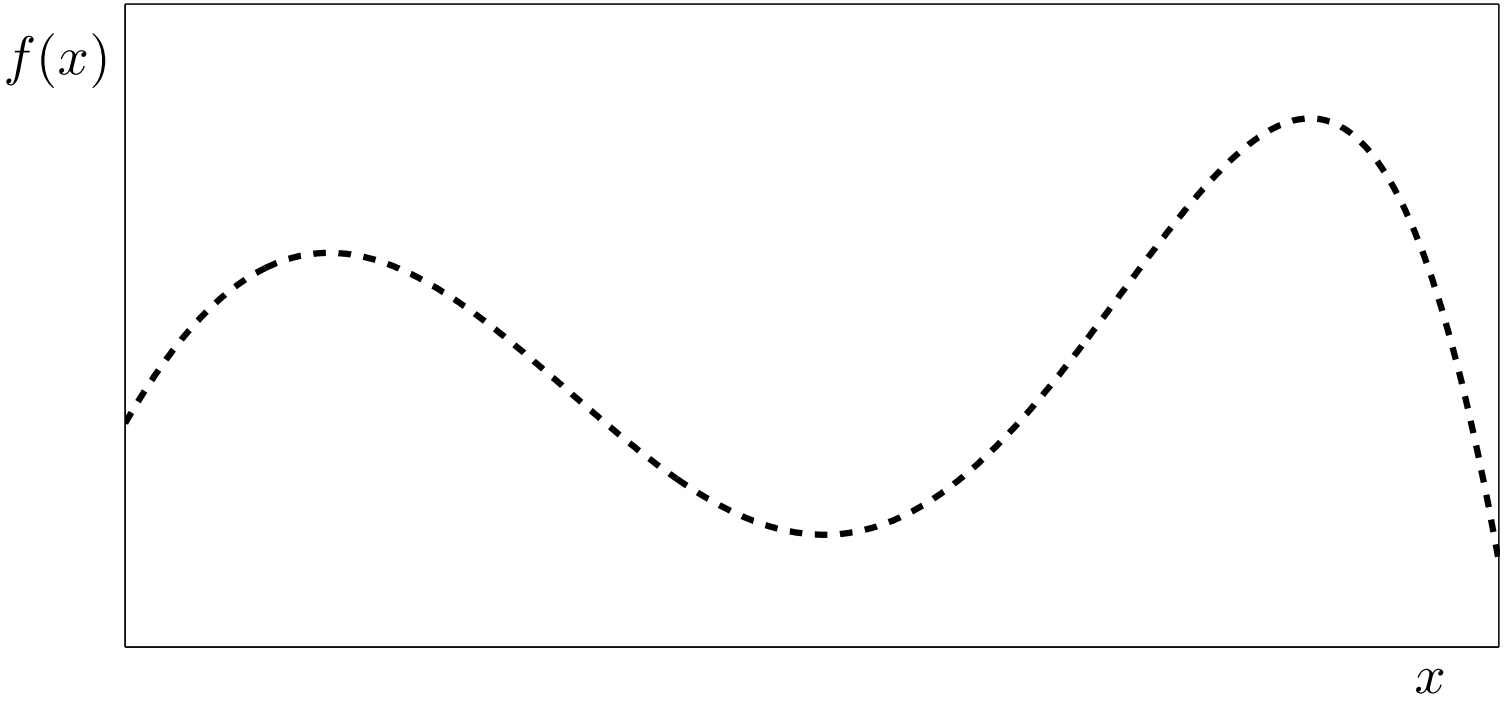
Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

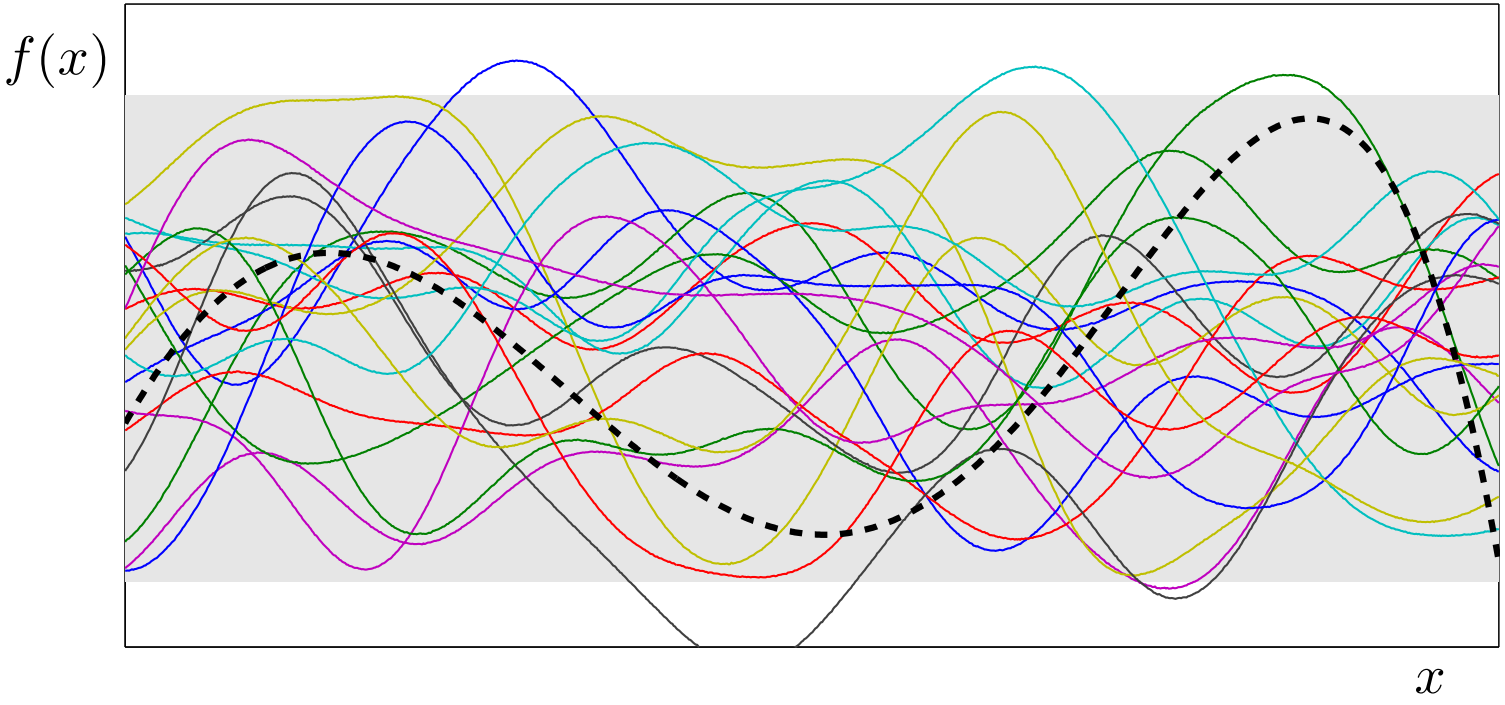
Functions with no observations



Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

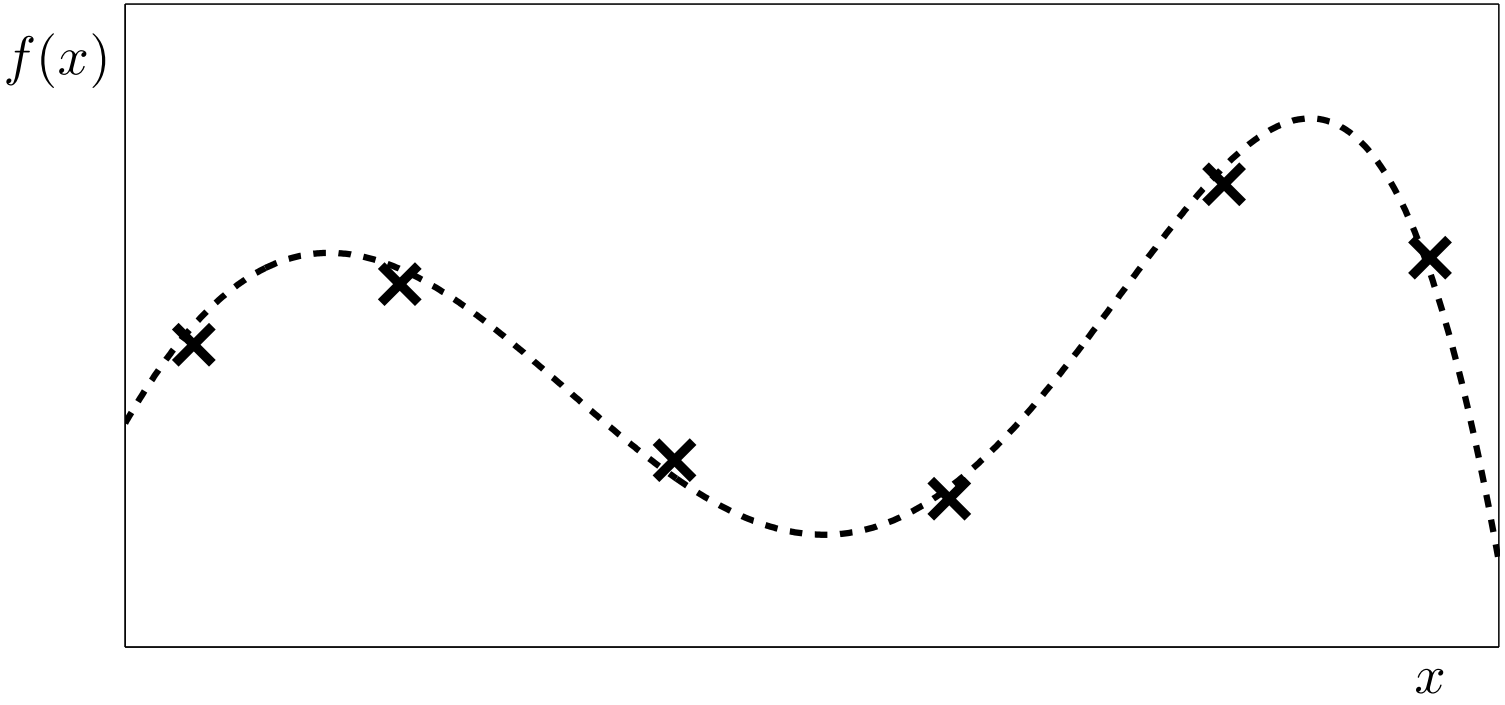
Prior $\mathcal{GP}$



Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

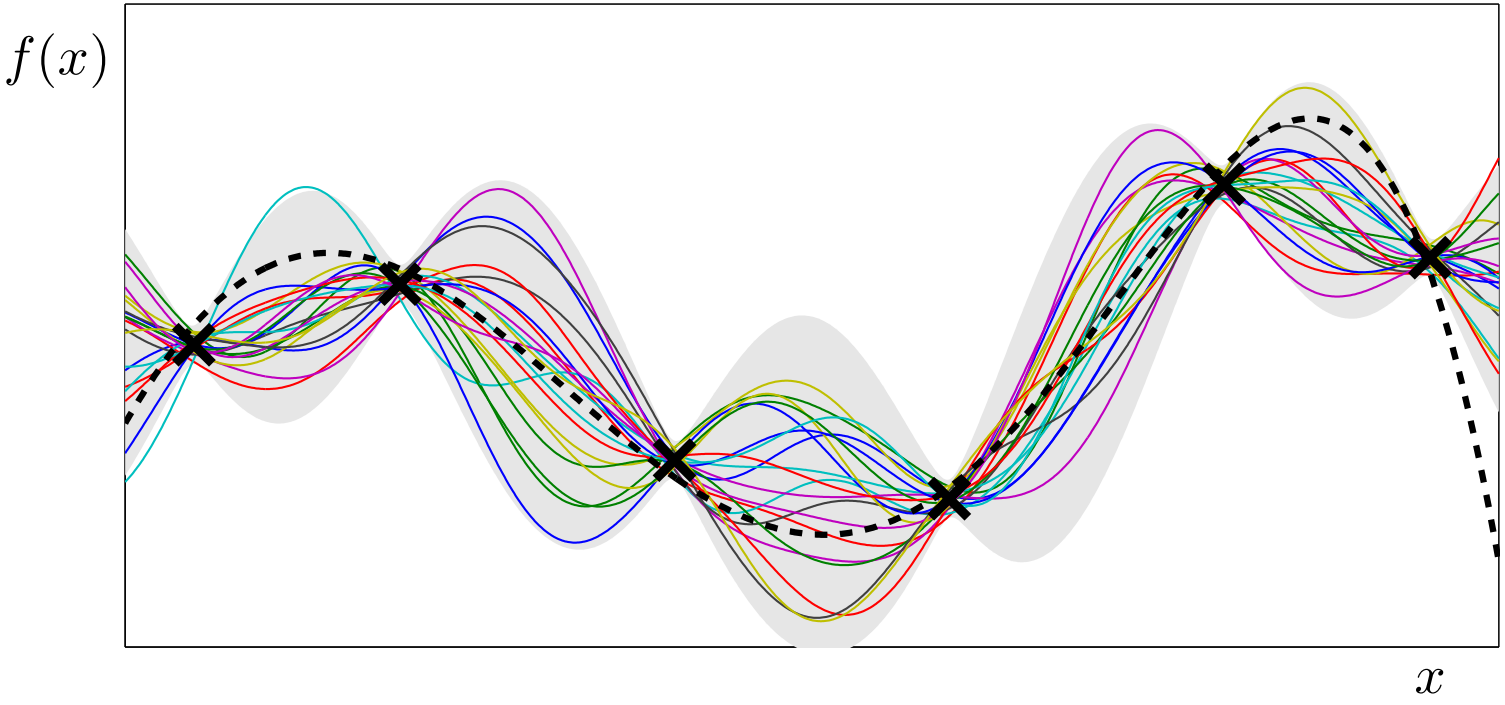
Observations



Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

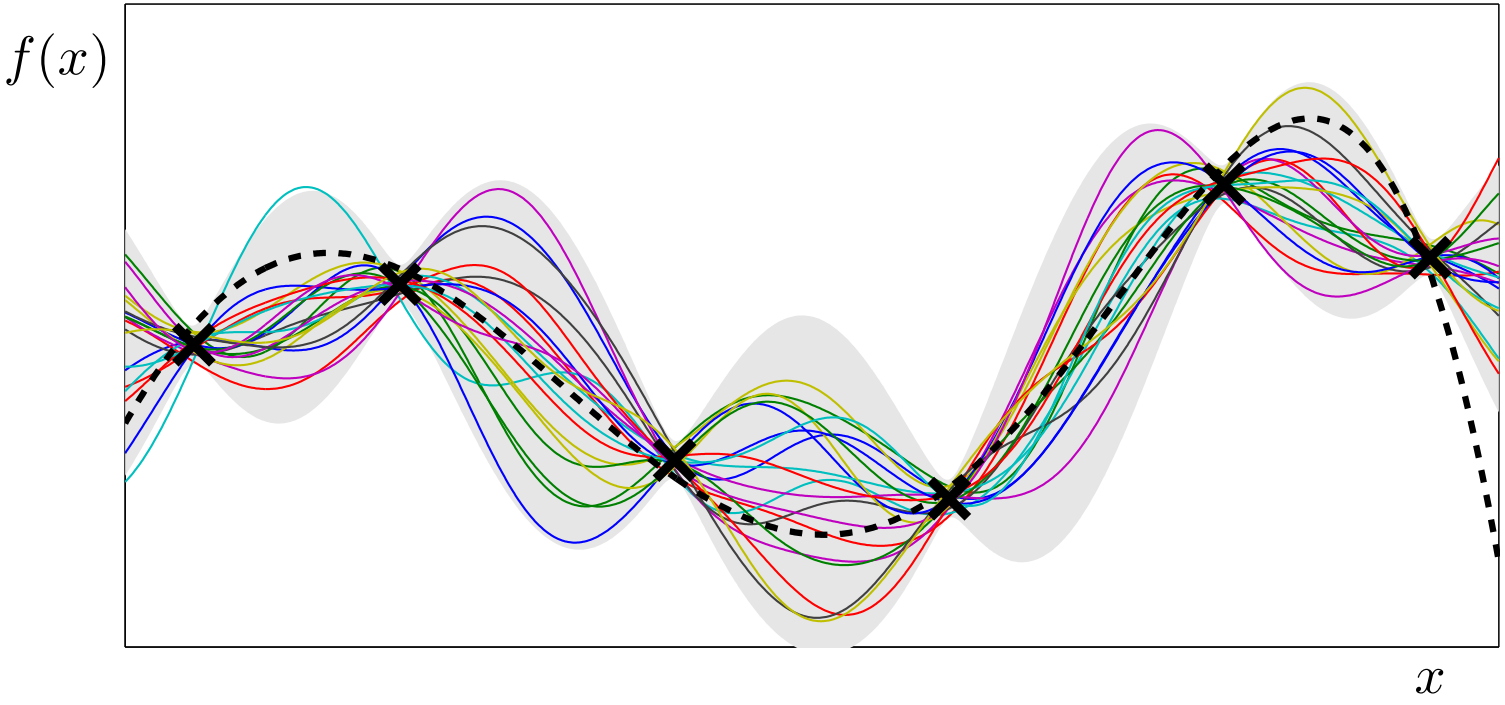
Posterior $\mathcal{GP}$ given observations



Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: A distribution over functions from $\mathcal{X}$ to $\mathbb{R}$.

Posterior $\mathcal{GP}$ given observations



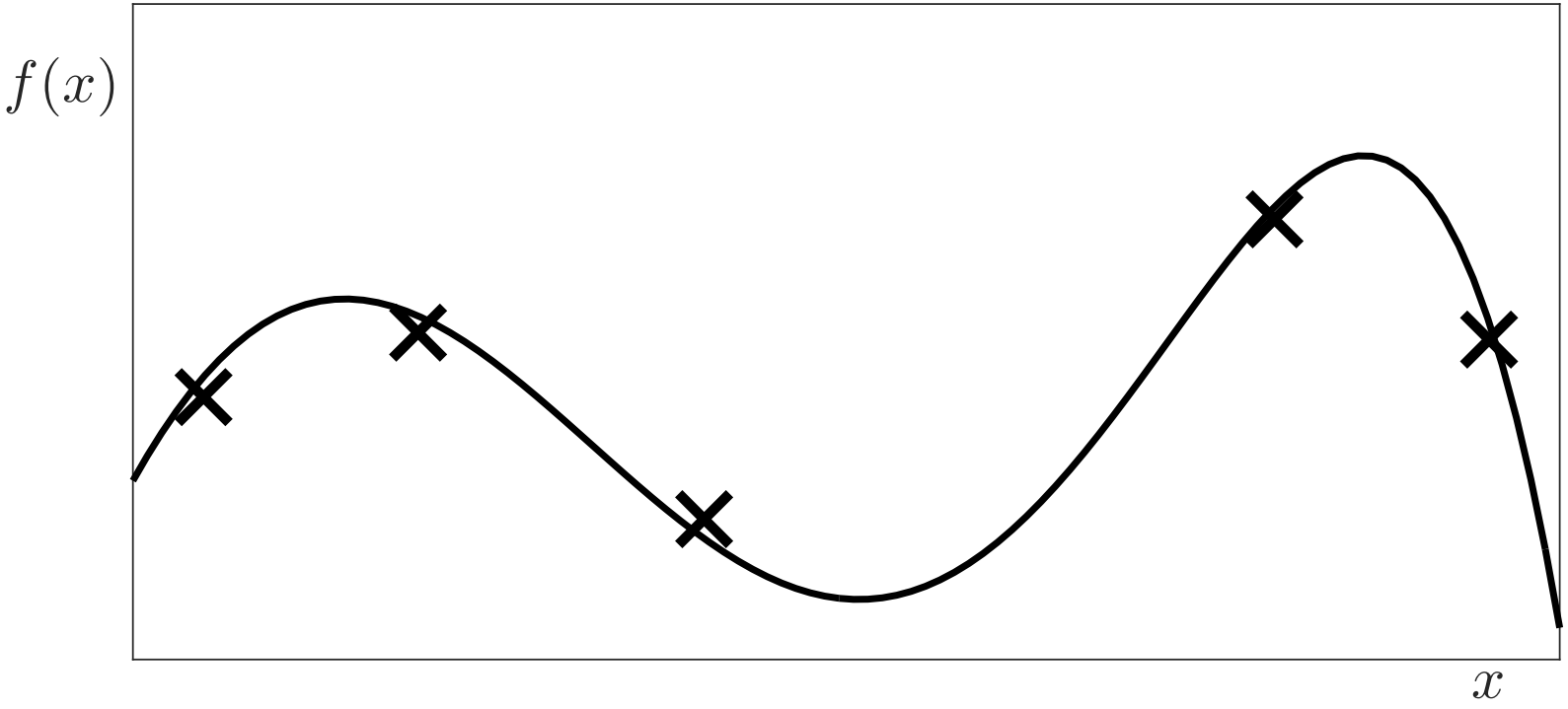
Completely characterised by mean function $\mu : \mathcal{X} \to \mathbb{R}$, and covariance kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$.
After t observations, $\quad f(x) \sim \mathcal{N}(\,\mu_t(x), \sigma_t^2(x)\,)$.

Gaussian Process Bandit (Bayesian) Optimisation

Model $f \sim \mathcal{GP}(\mathbf{0}, \kappa)$.

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010)

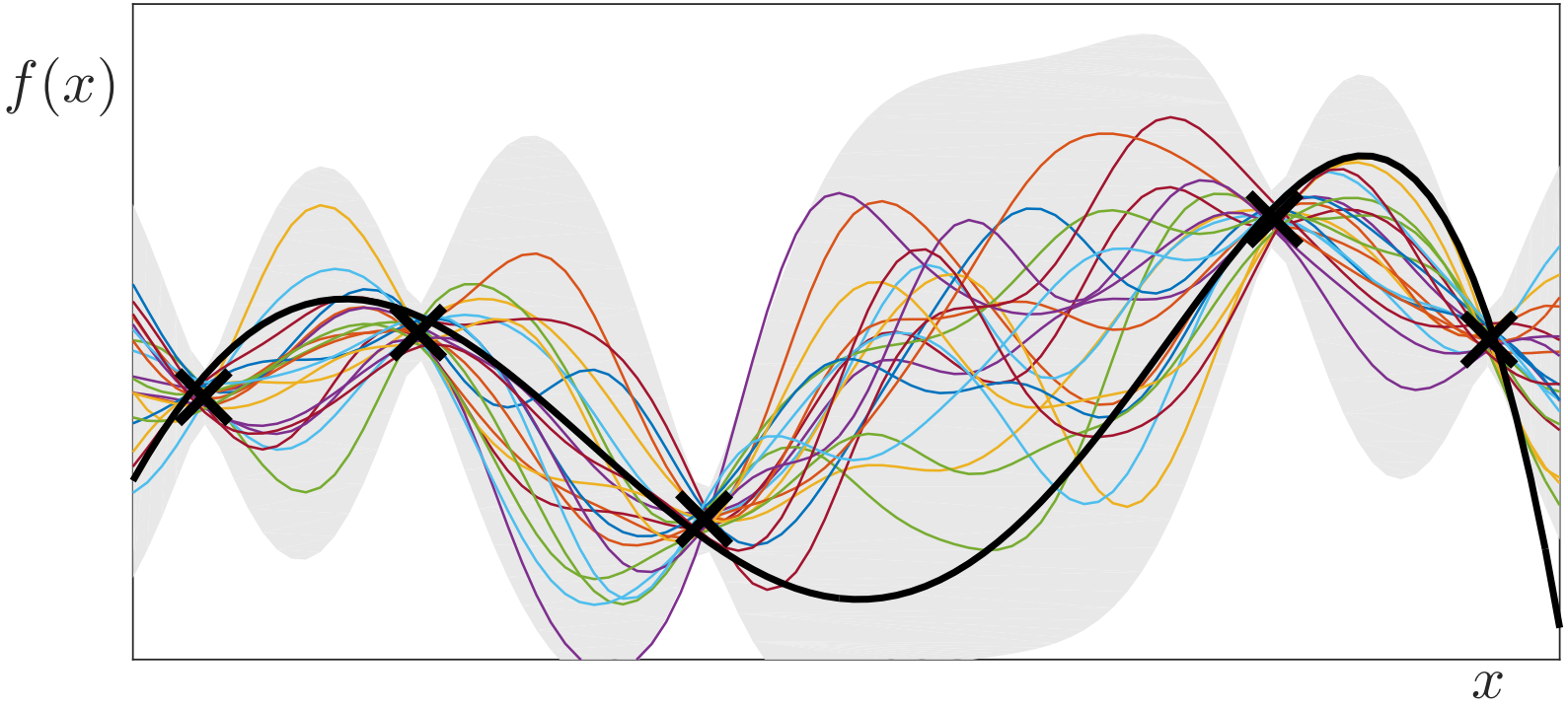


Gaussian Process Bandit (Bayesian) Optimisation

Model $f \sim \mathcal{GP}(\mathbf{0}, \kappa)$.

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010)



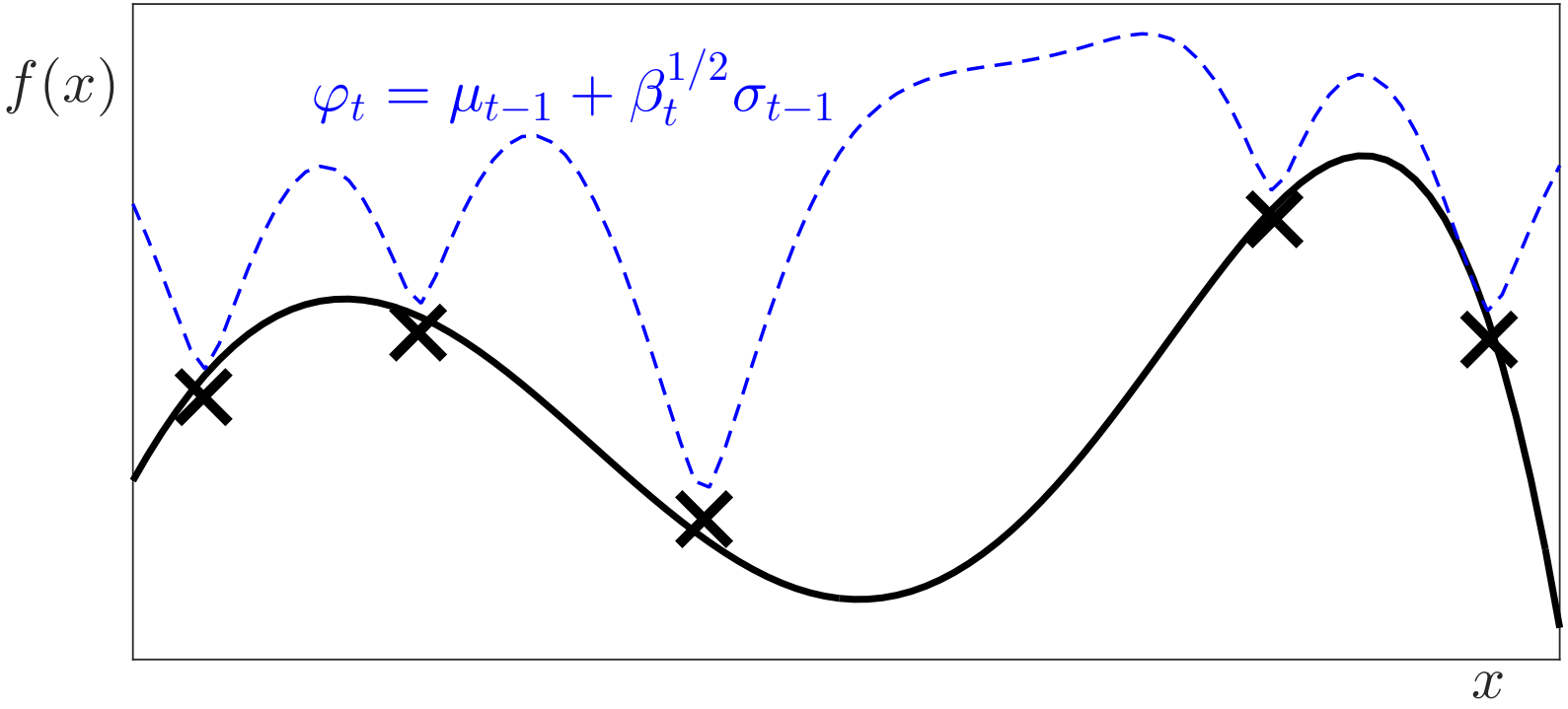
1) Construct posterior $\mathcal{GP}$.

Gaussian Process Bandit (Bayesian) Optimisation

Model $f \sim \mathcal{GP}(\mathbf{0}, \kappa)$.

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010)



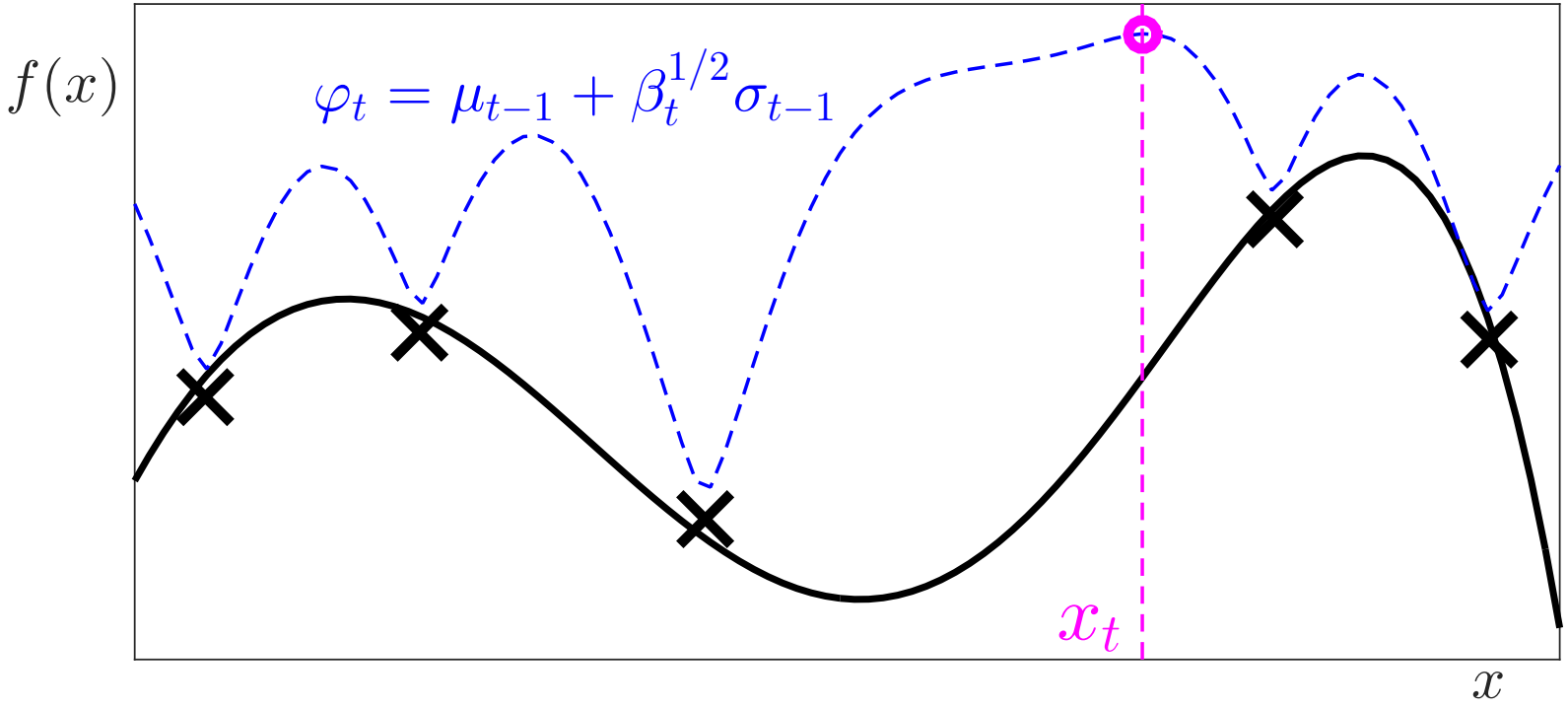
1) Construct posterior $\mathcal{GP}$. 2) $\varphi_t = \mu_{t-1} + \beta_t^{1/2}\sigma_{t-1}$ is a UCB.

Gaussian Process Bandit (Bayesian) Optimisation

Model $f \sim \mathcal{GP}(\mathbf{0}, \kappa)$.

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010)



1) Construct posterior $\mathcal{GP}$. 2) $\varphi_t = \mu_{t-1} + \beta_t^{1/2}\sigma_{t-1}$ is a UCB.

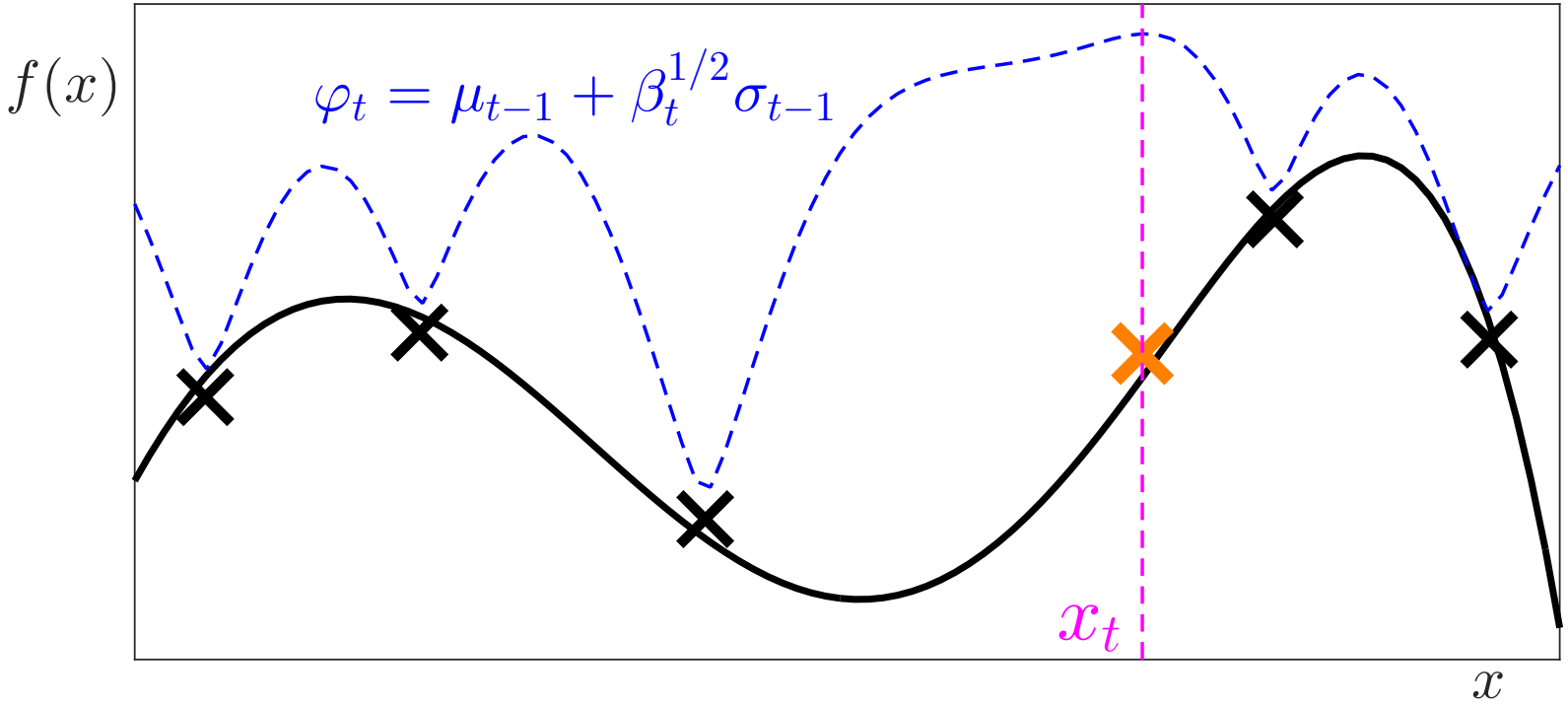
3) Choose $x_t = \operatorname{argmax}_x \varphi_t(x)$.

Gaussian Process Bandit (Bayesian) Optimisation

Model $f \sim \mathcal{GP}(\mathbf{0}, \kappa)$.

Gaussian Process Upper Confidence Bound (GP-UCB)

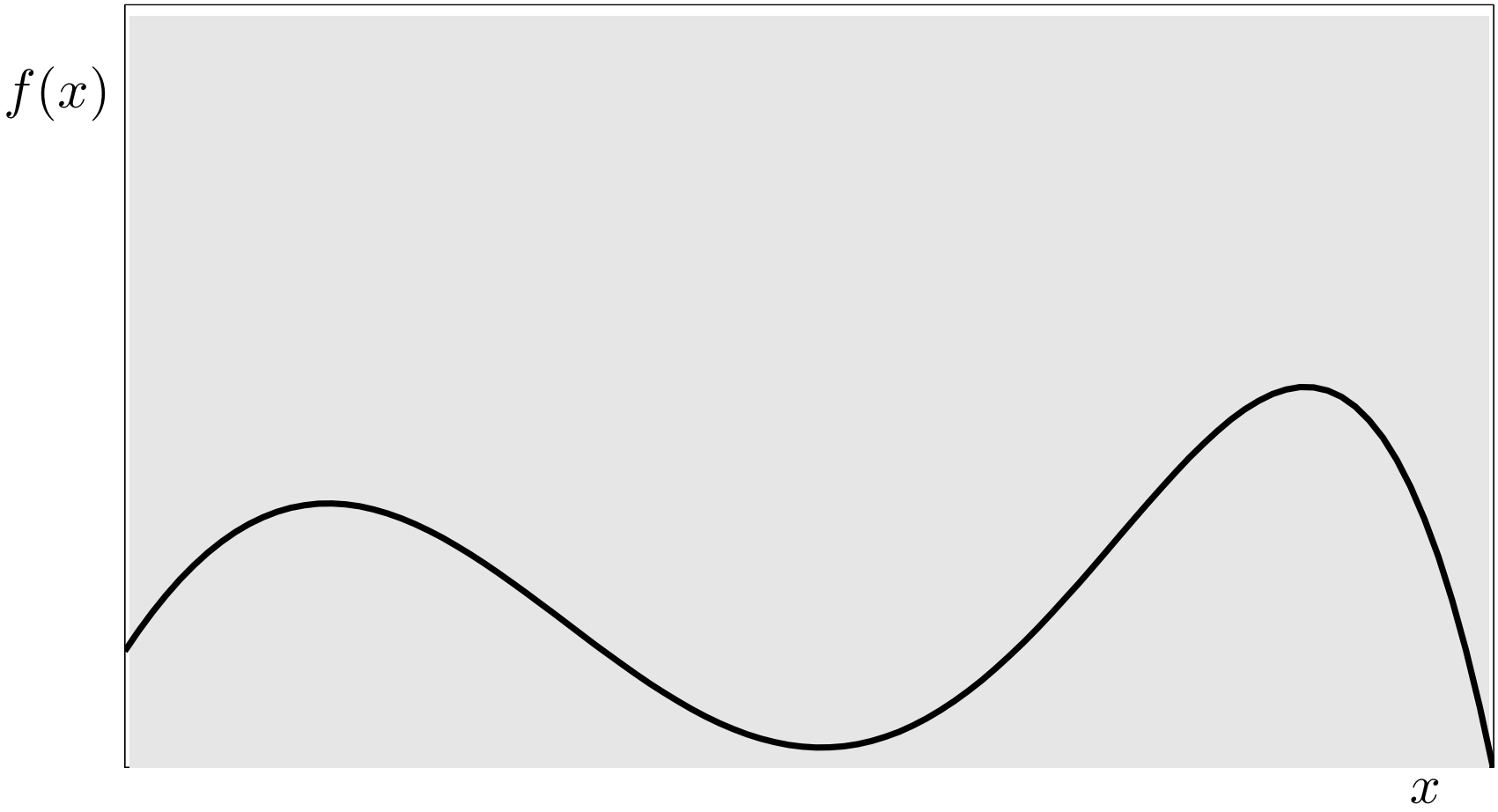
(Srinivas et al. 2010)



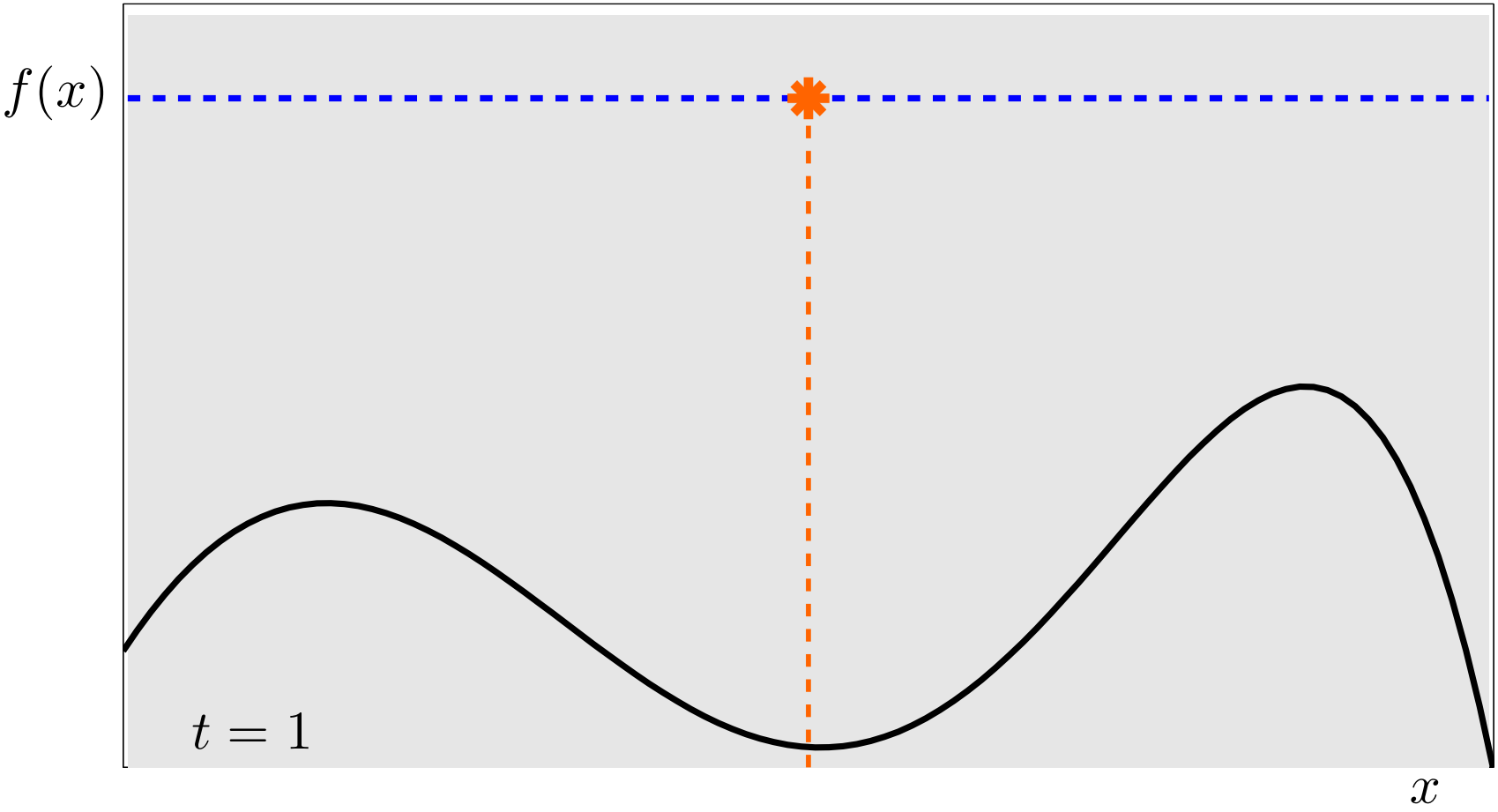
1) Construct posterior $\mathcal{GP}$. 2) $\varphi_t = \mu_{t-1} + \beta_t^{1/2}\sigma_{t-1}$ is a UCB.

3) Choose $x_t = \operatorname{argmax}_x \varphi_t(x)$. 4) Evaluate f at x_t.

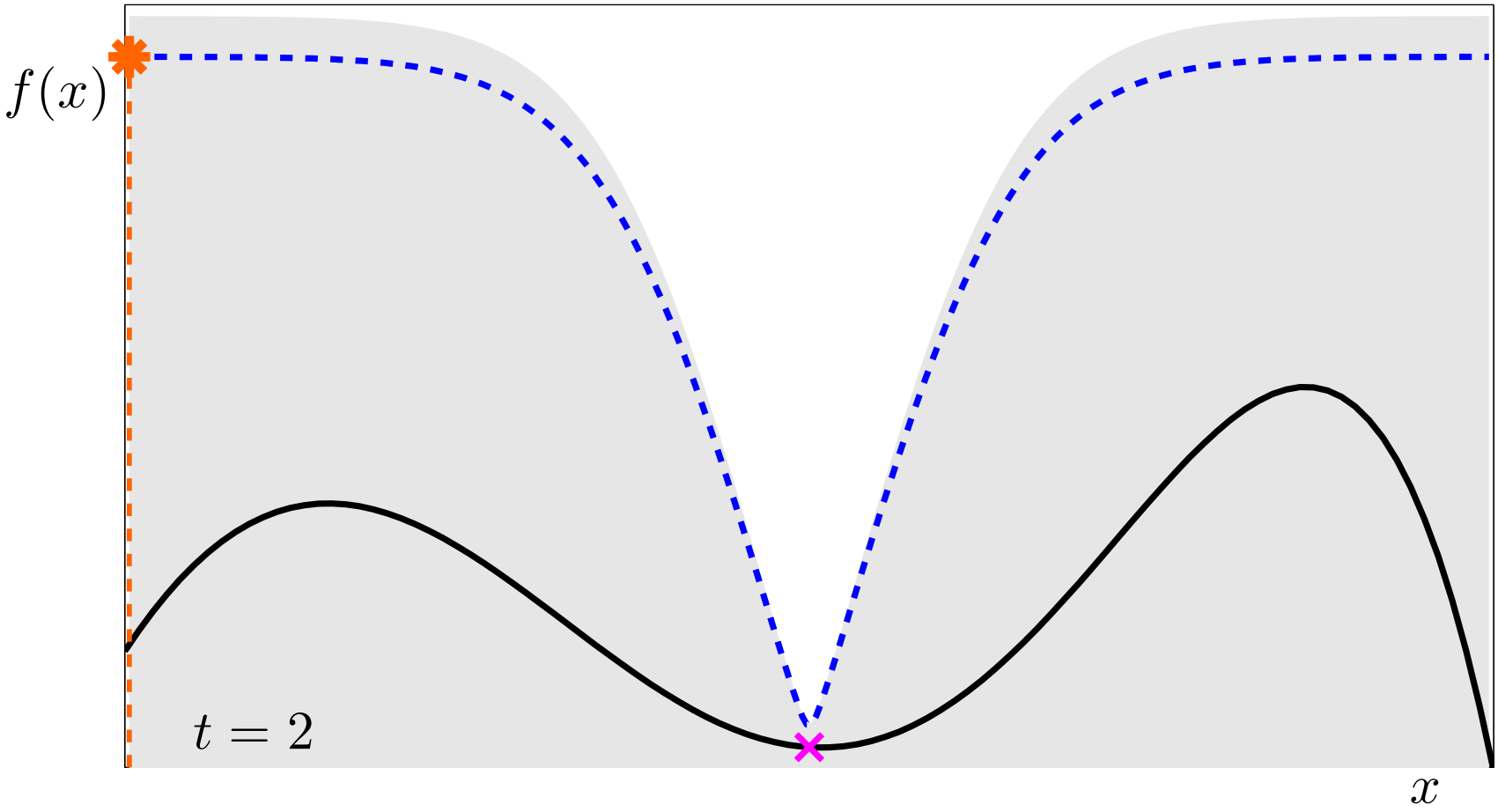
GP-UCB (Srinivas et al. 2010)



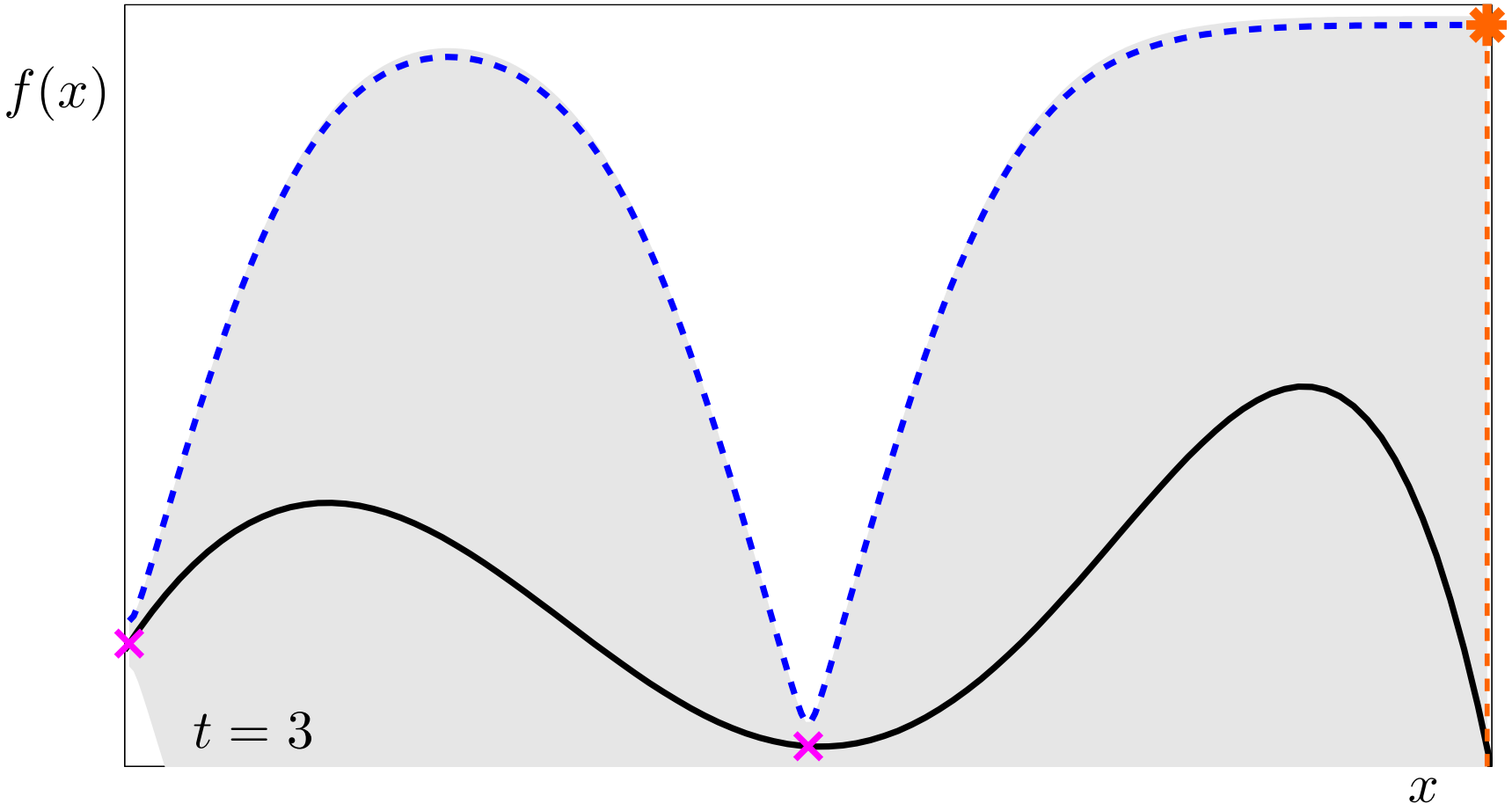
GP-UCB (Srinivas et al. 2010)



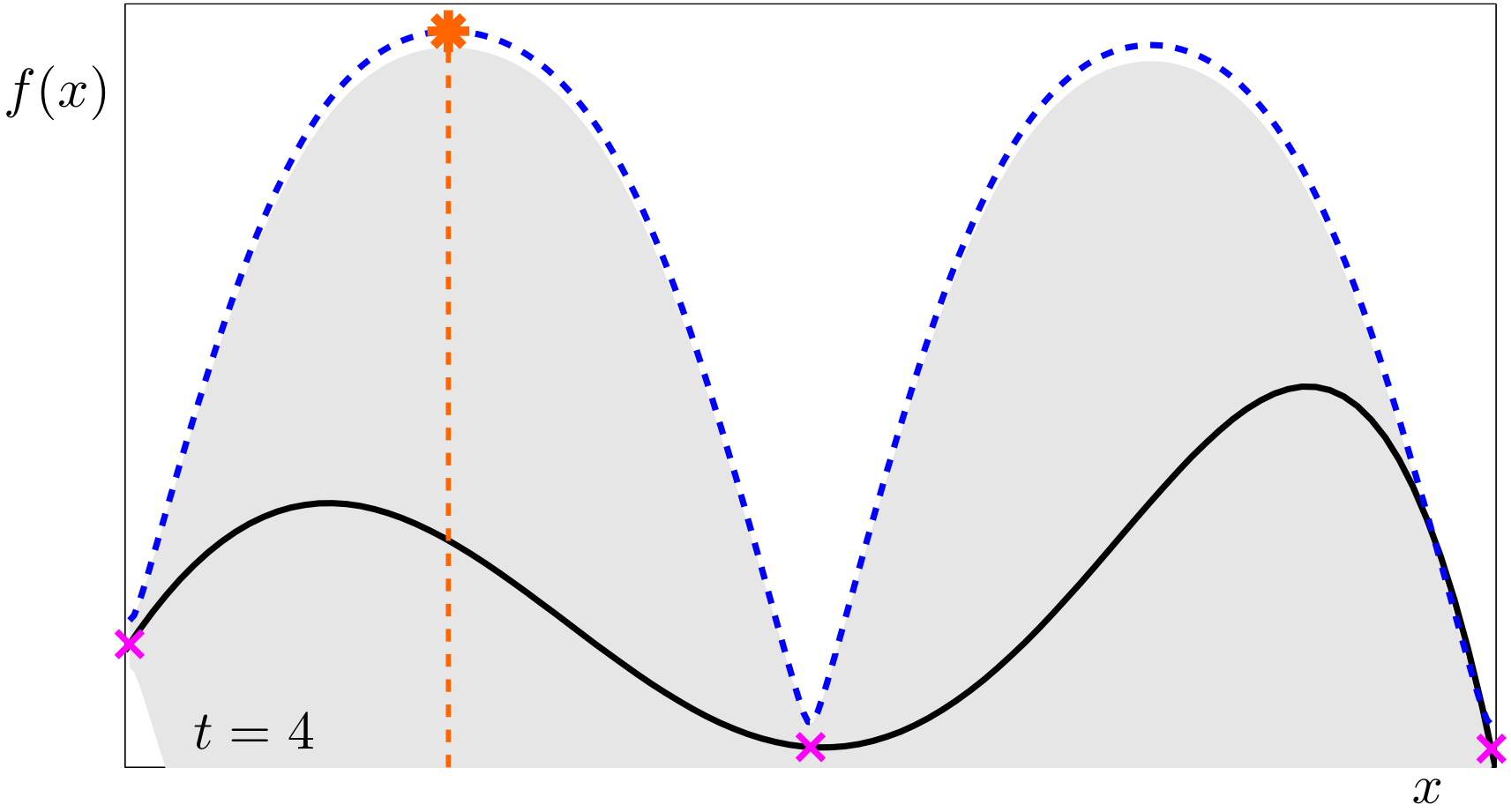
GP-UCB (Srinivas et al. 2010)



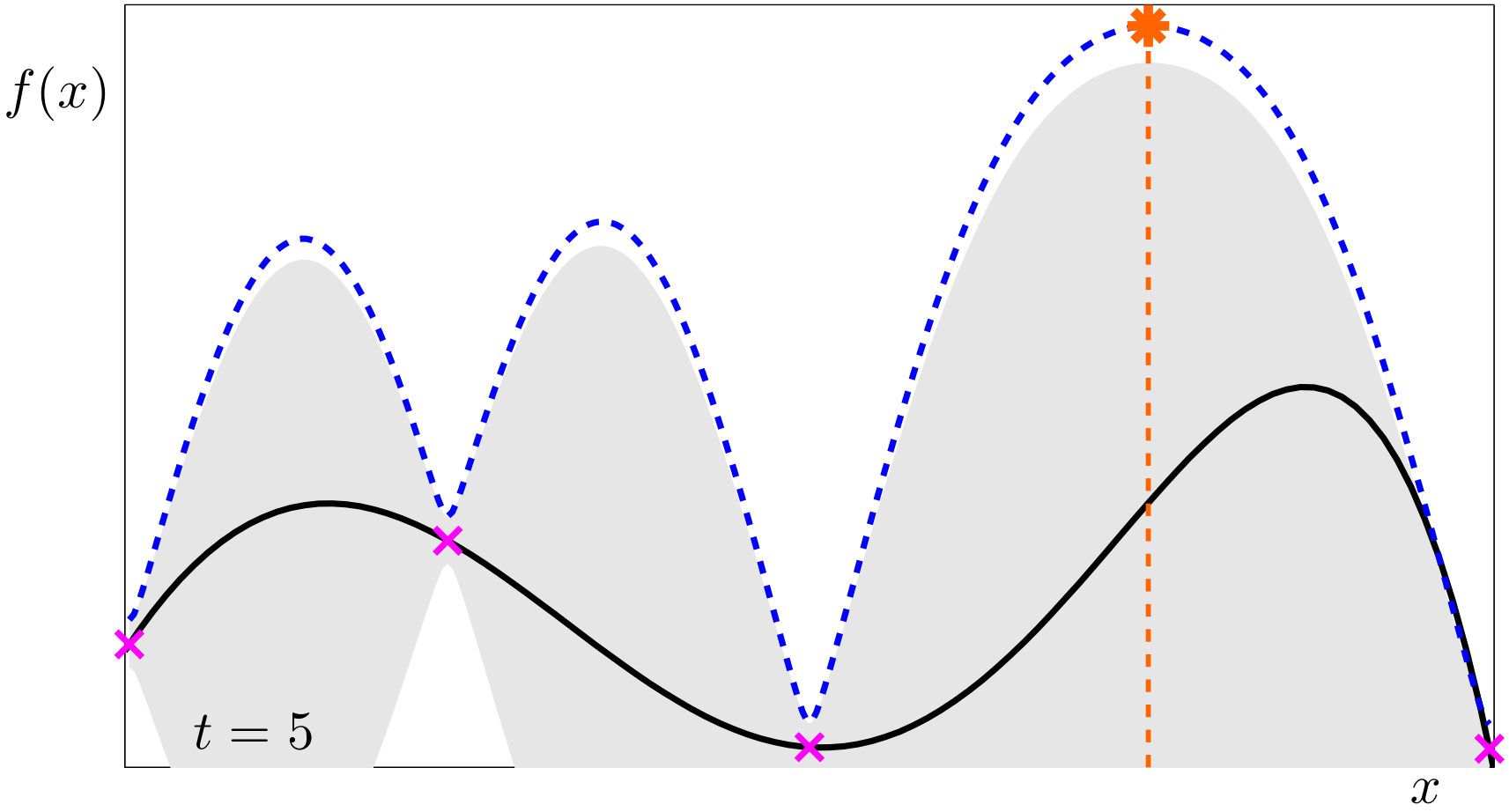
GP-UCB (Srinivas et al. 2010)



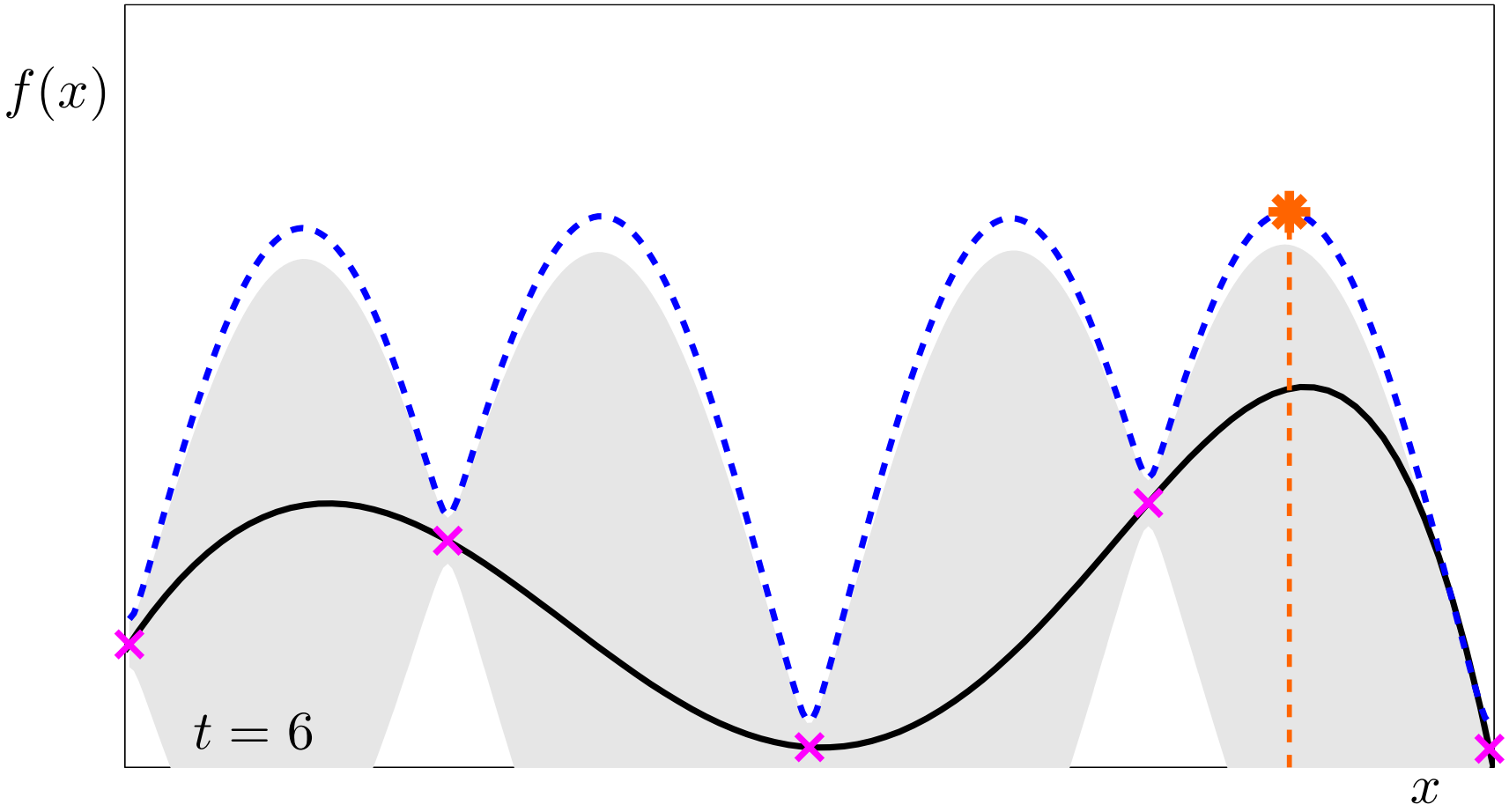
GP-UCB (Srinivas et al. 2010)



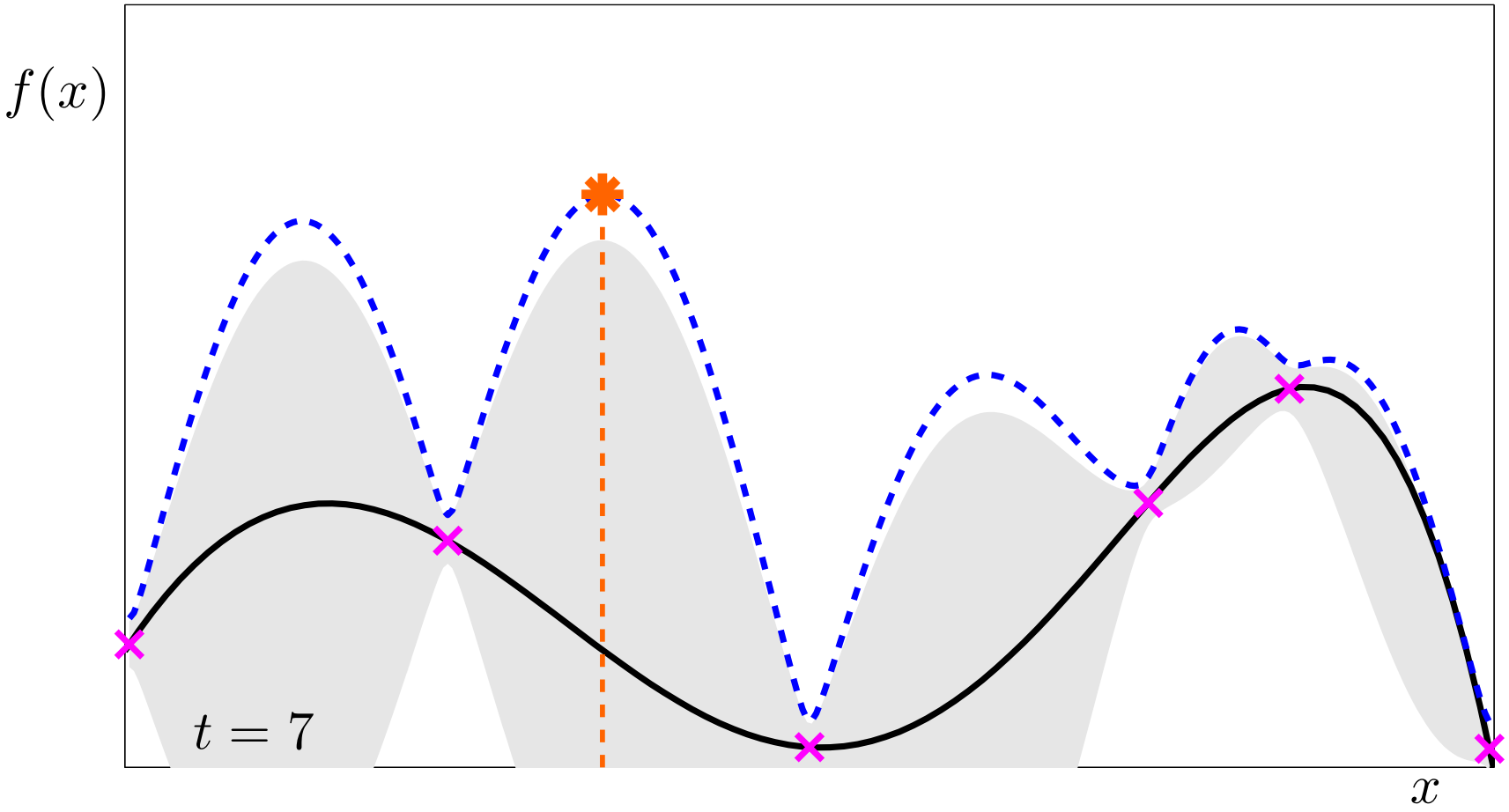
GP-UCB (Srinivas et al. 2010)



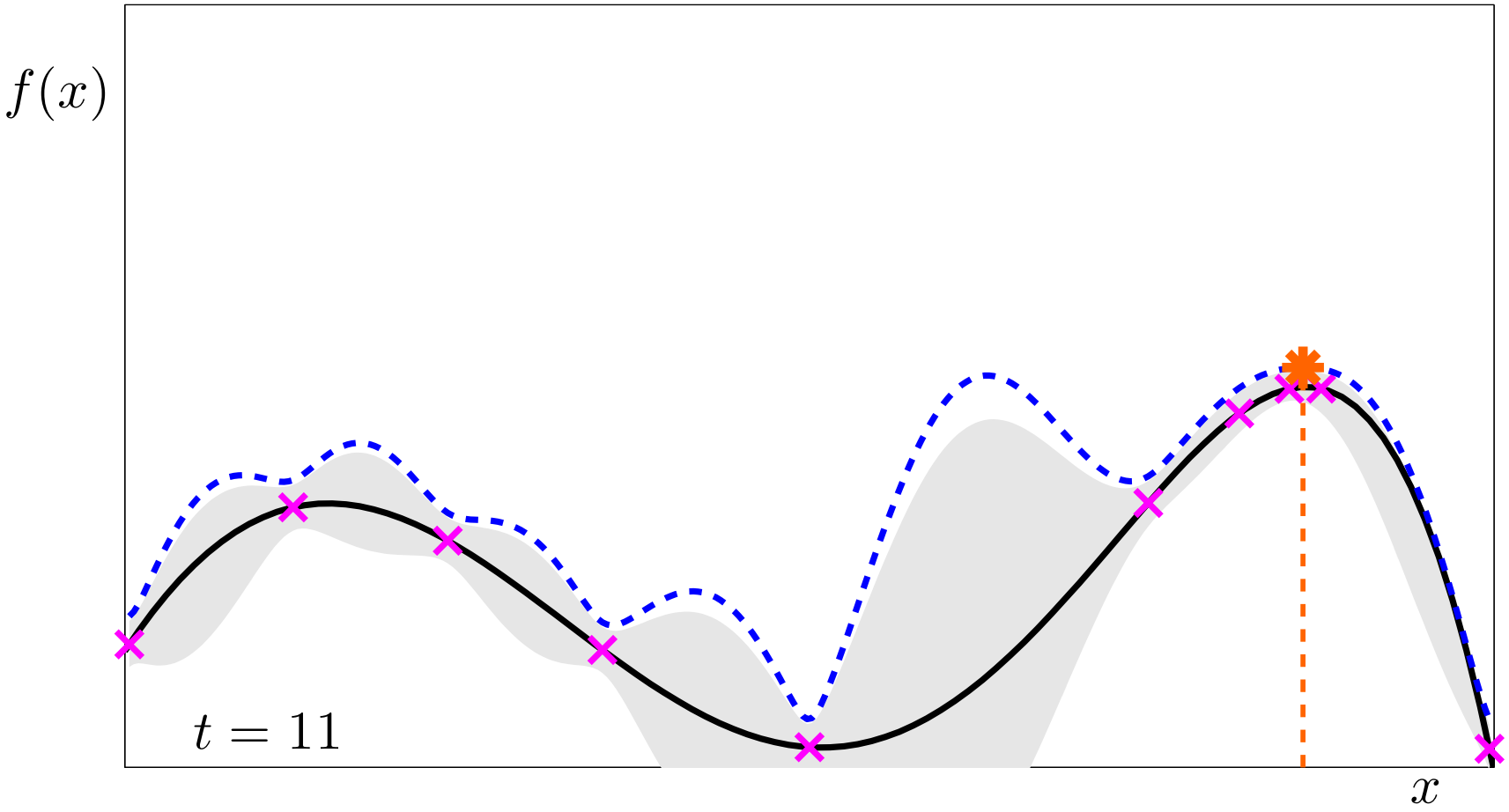
GP-UCB (Srinivas et al. 2010)



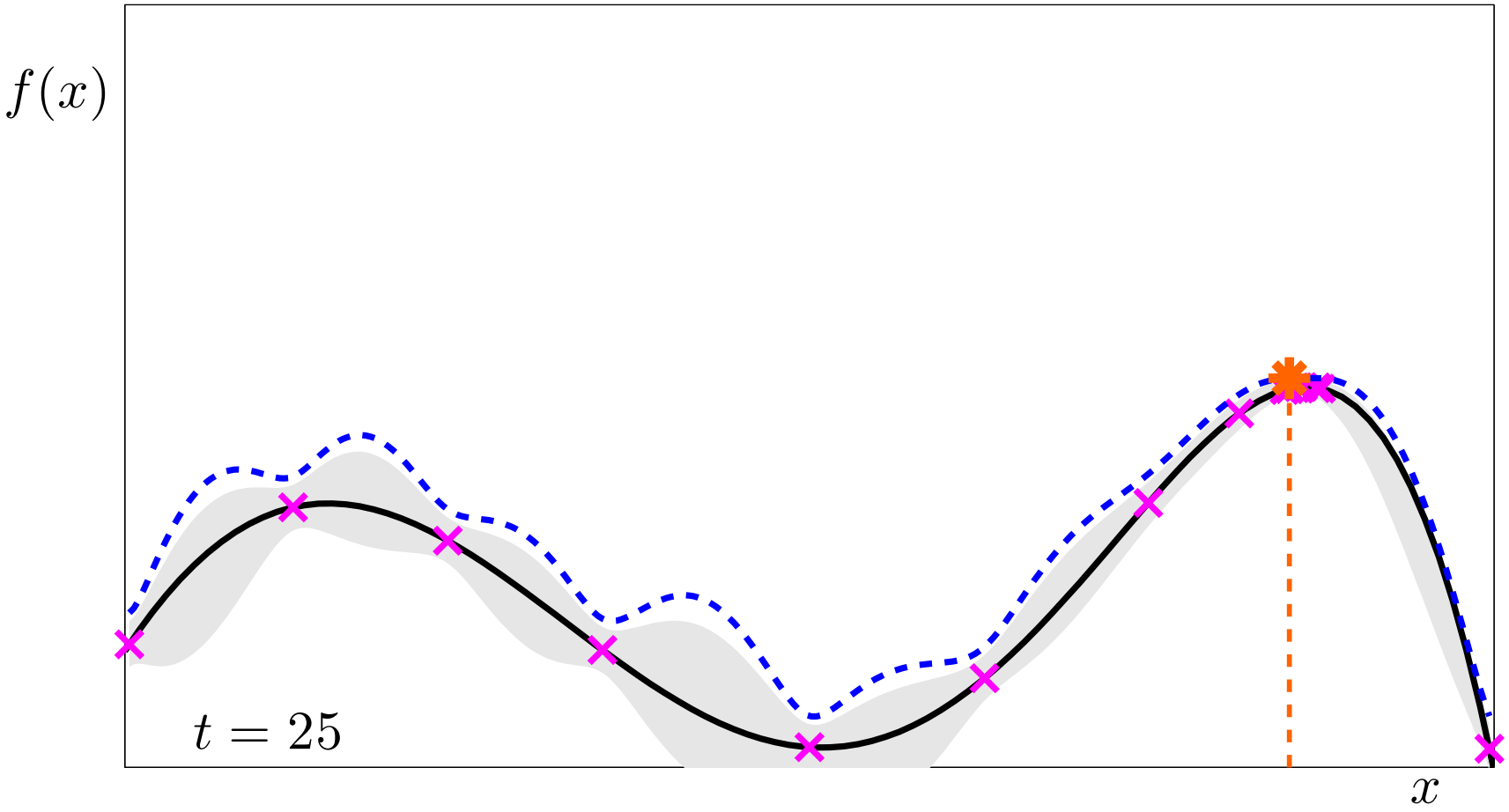
GP-UCB (Srinivas et al. 2010)



GP-UCB (Srinivas et al. 2010)



GP-UCB (Srinivas et al. 2010)



GP-UCB

$$x_t = \operatorname*{argmax}_x \ \ \mu_{t-1}(x) + \beta_t^{1/2} \sigma_{t-1}(x)$$

- μ_{t-1}: Exploitation
- σ_{t-1}: Exploration
- β_t controls the tradeoff. $\beta_t \asymp \log t$.

GP-UCB

$$x_t = \operatorname*{argmax}_x \ \ \mu_{t-1}(x) + \beta_t^{1/2}\sigma_{t-1}(x)$$

- μ_{t-1}: Exploitation
- σ_{t-1}: Exploration
- β_t controls the tradeoff. $\beta_t \asymp \log t$.

GP-UCB, κ is an SE kernel (Srinivas et al. 2010)

$$\text{w.h.p} \qquad S_n \ = \ f(x_\star) - \max_{t=1,\dots,n} f(x_t) \ \lesssim \ \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{n}}$$

$\lesssim$ ignores constants and $\mathrm{polylog}$ terms.

Big picture: scaling up black-box optimisation

Big picture: scaling up black-box optimisation

- Optimising in high dimensional spaces
  e.g.: Tuning models with several hyper-parameters
  Additive models for f lead to statistically and computationally tractable algorithms. (Kandasamy et al. ICML 2015)

Big picture: scaling up black-box optimisation

- Optimising in high dimensional spaces
  e.g.: Tuning models with several hyper-parameters
  Additive models for f lead to statistically and computationally tractable algorithms. (Kandasamy et al. ICML 2015)

- Parallelising function evaluations
  Randomised algorithms scale well to a large number of parallel workers. (Kandasamy et al. Arxiv 2017)

Big picture: scaling up black-box optimisation

- Optimising in high dimensional spaces
  e.g.: Tuning models with several hyper-parameters
  Additive models for f lead to statistically and computationally tractable algorithms. (Kandasamy et al. ICML 2015)

- Parallelising function evaluations
  Randomised algorithms scale well to a large number of parallel workers. (Kandasamy et al. Arxiv 2017)

Extends beyond GPs.

This work: What if we have cheap approximations to f ?

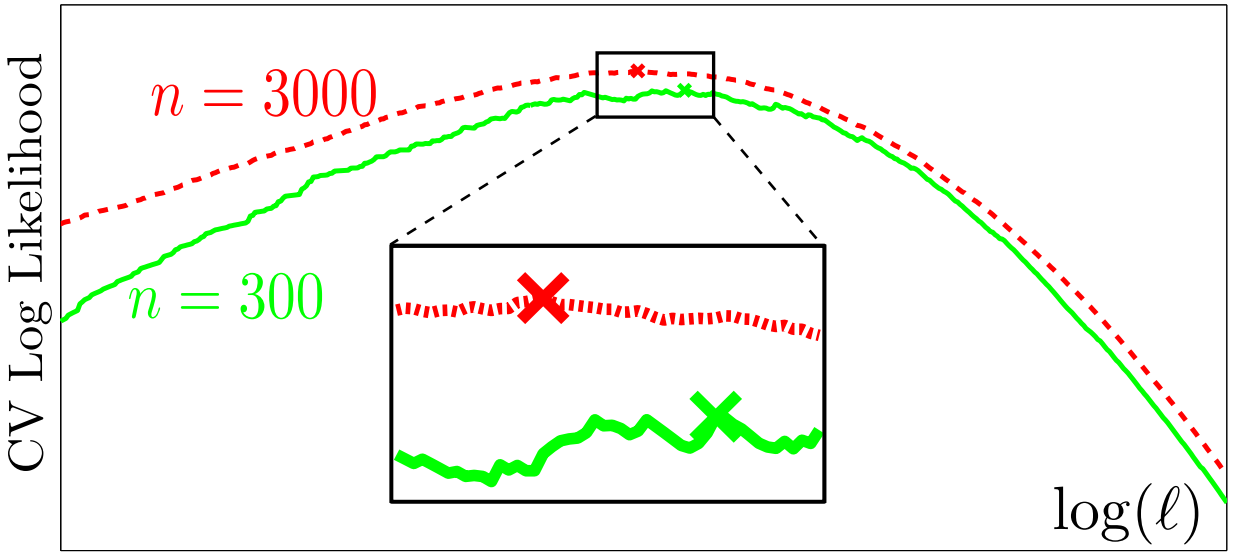
(Kandasamy et al. NIPS 2016a&b, Kandasamy et al. ICML 2017)

This work: What if we have cheap approximations to f ?

(Kandasamy et al. NIPS 2016a&b, Kandasamy et al. ICML 2017)

1. Hyper-parameter tuning: Train & validate with a subset of the data, and/or early stopping before convergence.

   E.g. Bandwidth (ℓ) selection in kernel density estimation.

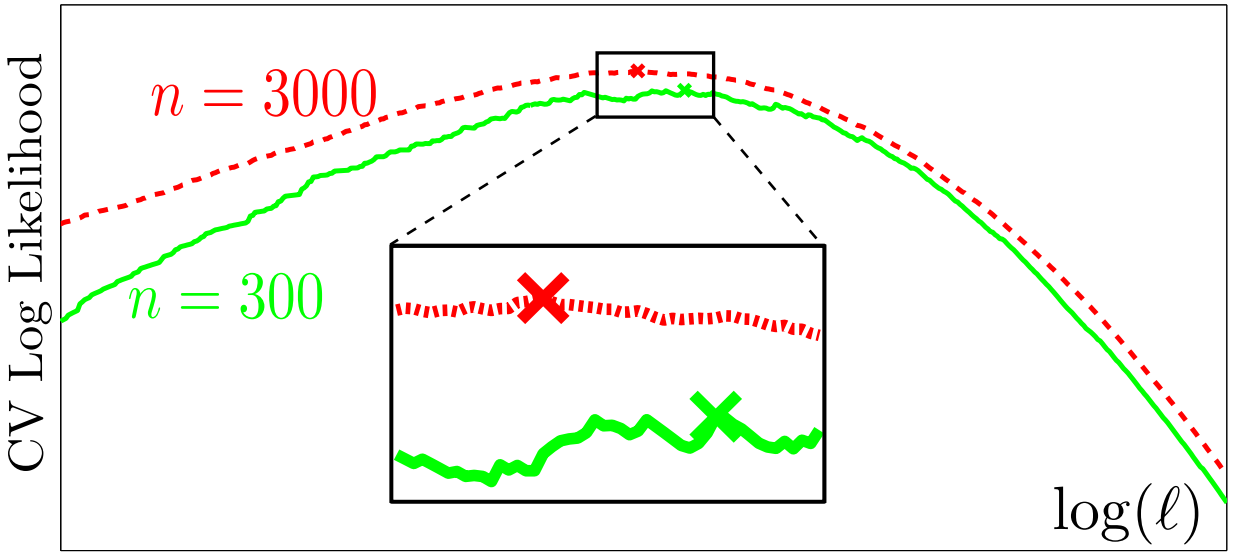


This work: What if we have cheap approximations to f ?

(Kandasamy et al. NIPS 2016a&b, Kandasamy et al. ICML 2017)

1. Hyper-parameter tuning: Train & validate with a subset of the data, and/or early stopping before convergence.

   E.g. Bandwidth (ℓ) selection in kernel density estimation.



2. Computational astrophysics: cosmological simulations and numerical computations with less granularity.

3. Autonomous driving: simulation vs real world experiment.

Prior work in **Multi-fidelity** Methods

For specific applications,

- Industrial design (Forrester et al. 2007)
- Hyper-parameter tuning (Agarwal et al. 2011, Klein et al. 2015, Li et al. 2016)
- Active learning (Zhang & Chaudhuri 2015)
- Robotics (Cutler et al. 2014)

Multi-fidelity optimisation

(Huang et al. 2006, Forrester et al. 2007, March & Wilcox 2012, Poloczek et al. 2016)

Outline

1. A finite number of approximations

   (Kandasamy et al. NIPS 2016b)

   - Formalism, intuition and challenges
   - Algorithm
   - Theoretical results
   - Experiments

2. A continuous spectrum of approximations

   (Kandasamy et al. ICML 2017)

   - Formalism
   - Algorithm
   - Theoretical results
   - Experiments

Outline

1. A finite number of approximations

(Kandasamy et al. NIPS 2016b)

- Formalism, intuition and challenges
- Algorithm
- Theoretical results
- Experiments

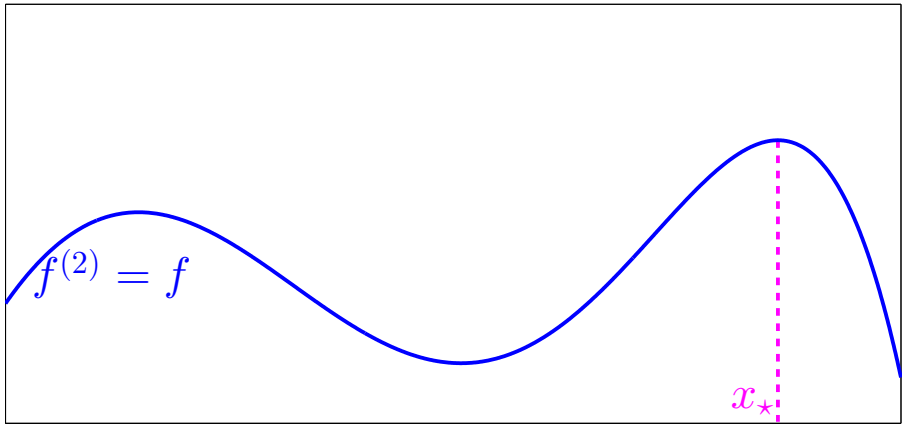
2. A continuous spectrum of approximations

(Kandasamy et al. ICML 2017)

- Formalism
- Algorithm
- Theoretical results
- Experiments

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

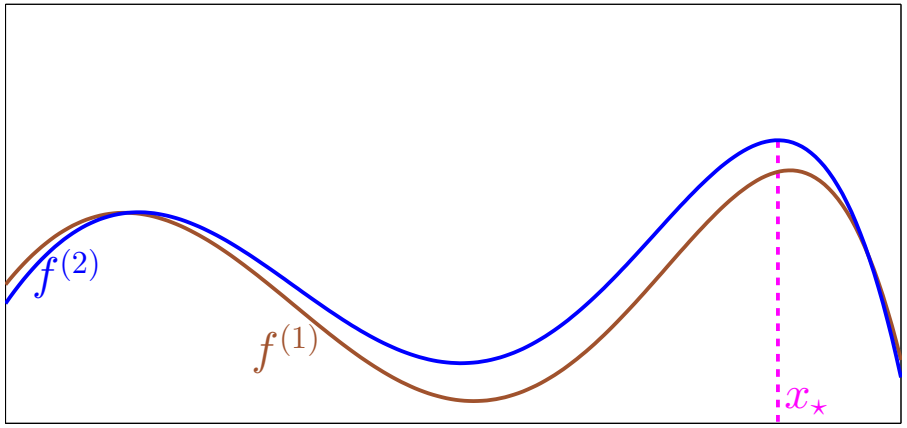
(Kandasamy et al. NIPS 2016b)



- Optimise $f = f^{(2)}$. $\quad x_\star = \mathrm{argmax}_x \, f^{(2)}(x)$.
- **But ..**

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

(Kandasamy et al. NIPS 2016b)

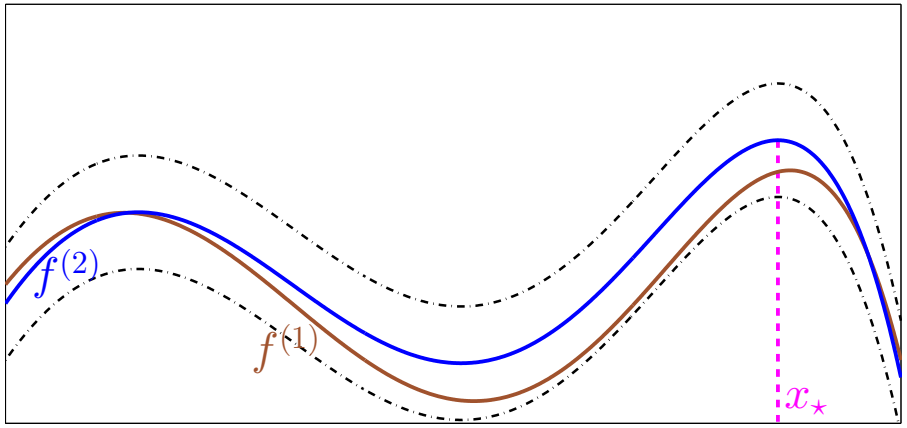


- Optimise $f = f^{(2)}$. $\quad x_\star = \mathrm{argmax}_x \, f^{(2)}(x)$.
- **But ..** we have an approximation $f^{(1)}$ to $f^{(2)}$.
- $f^{(1)}$ costs $\lambda^{(1)}$, $\quad f^{(2)}$ costs $\lambda^{(2)}$. $\quad \lambda^{(1)} < \lambda^{(2)}$.

  "cost": could be computation time, money etc.

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

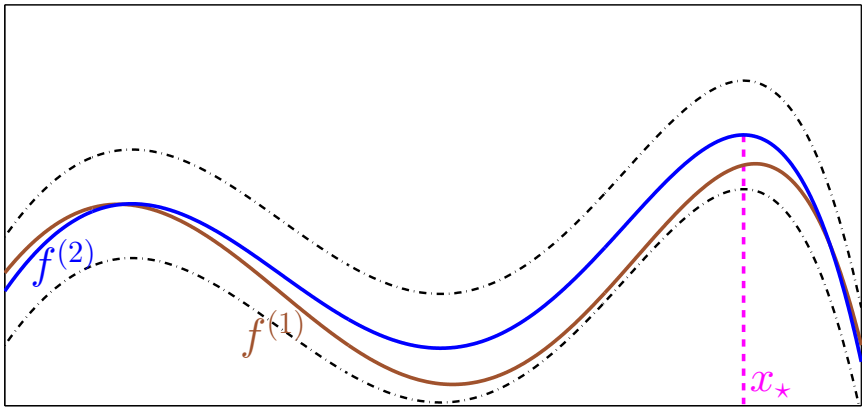
(Kandasamy et al. NIPS 2016b)



- Optimise $f = f^{(2)}$. $\quad x_\star = \mathrm{argmax}_x \, f^{(2)}(x)$.
- **But ..** we have an approximation $f^{(1)}$ to $f^{(2)}$.
- $f^{(1)}$ costs $\lambda^{(1)}$, $\quad f^{(2)}$ costs $\lambda^{(2)}$. $\quad \lambda^{(1)} < \lambda^{(2)}$.
  "cost": could be computation time, money etc.
- $f^{(1)}, f^{(2)} \sim \mathcal{GP}(0, \kappa)$.
- $\|f^{(2)} - f^{(1)}\|_\infty \leq \zeta^{(1)}$. $\quad \zeta^{(1)}$ is known.

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

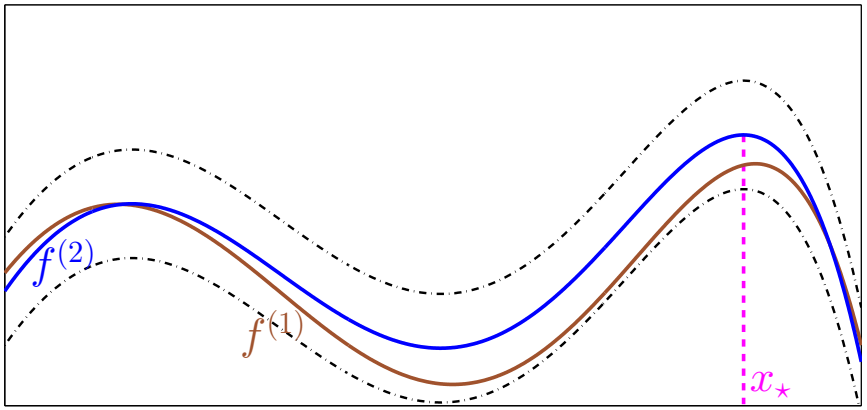
(Kandasamy et al. NIPS 2016b)



At time t**:** Determine the point $x_t \in \mathcal{X}$ and fidelity $m_t \in \{1, 2\}$ for querying.

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

(Kandasamy et al. NIPS 2016b)

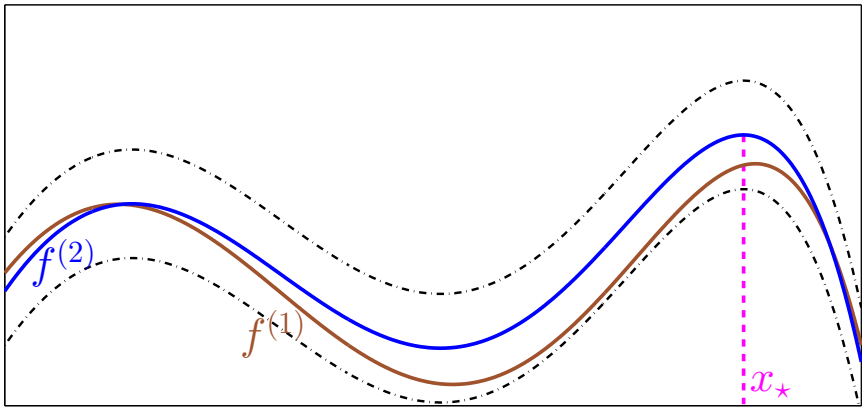


At time t: Determine the point $x_t \in \mathcal{X}$ and fidelity $m_t \in \{1, 2\}$ for querying.

End Goal: Maximise $f^{(2)}$. Don't care for maximum of $f^{(1)}$.

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

(Kandasamy et al. NIPS 2016b)



At time t: Determine the point $x_t \in \mathcal{X}$ and fidelity $m_t \in \{1, 2\}$ for querying.

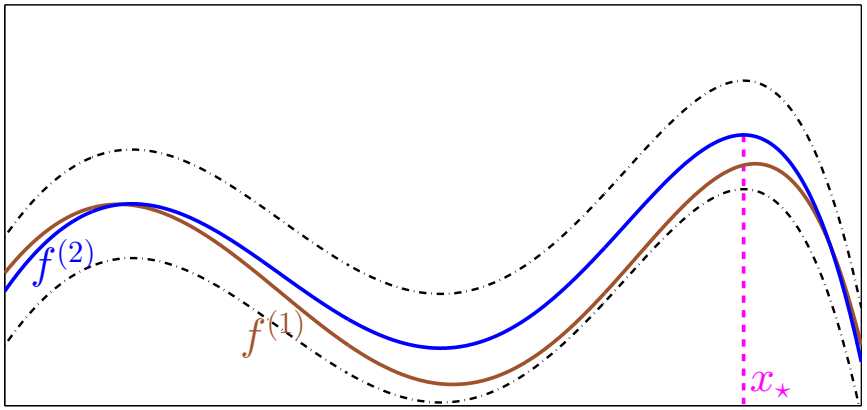
End Goal: Maximise $f^{(2)}$. Don't care for maximum of $f^{(1)}$.

Simple Regret: $$S(\Lambda) = f^{(2)}(x_\star) - \max_{t\,:\,m_t=2} f^{(2)}(x_t)$$

Capital $\Lambda \leftarrow$ amount of the resource spent. E.g. seconds or dollars.

Multi-fidelity Bandit Optimisation in 2 Fidelities (1 Approximation)

(Kandasamy et al. NIPS 2016b)



At time t: Determine the point $x_t \in \mathcal{X}$ and fidelity $m_t \in \{1, 2\}$ for querying.

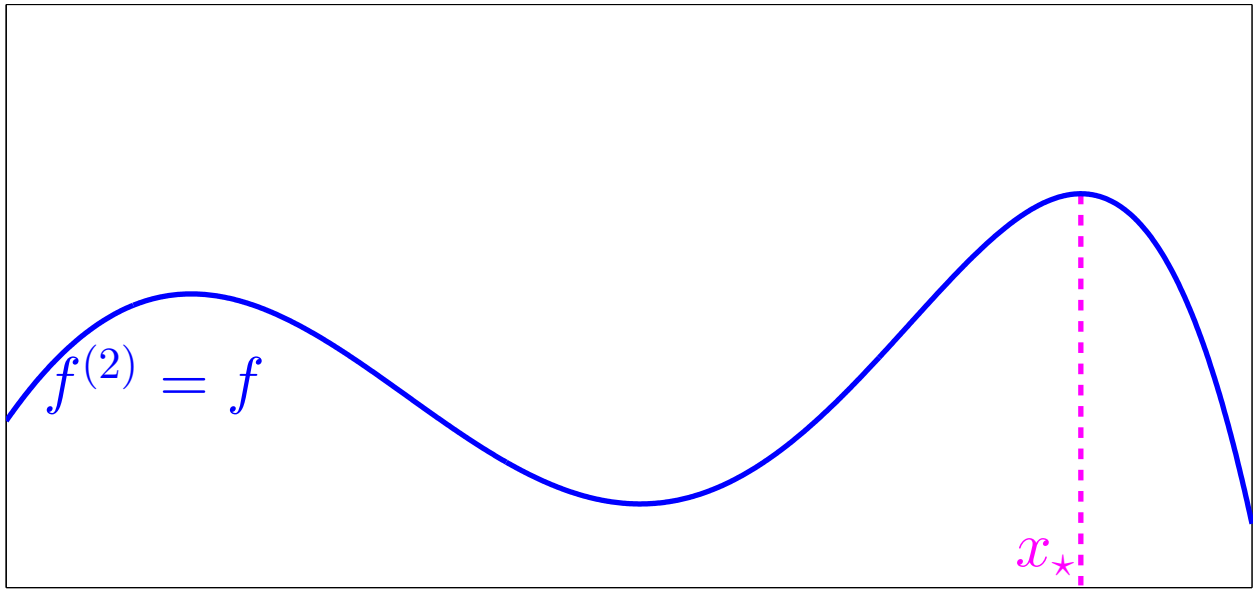
End Goal: Maximise $f^{(2)}$. Don't care for maximum of $f^{(1)}$.

Simple Regret: $$S(\Lambda) = f^{(2)}(x_\star) - \max_{t\,:\,m_t=2} f^{(2)}(x_t)$$

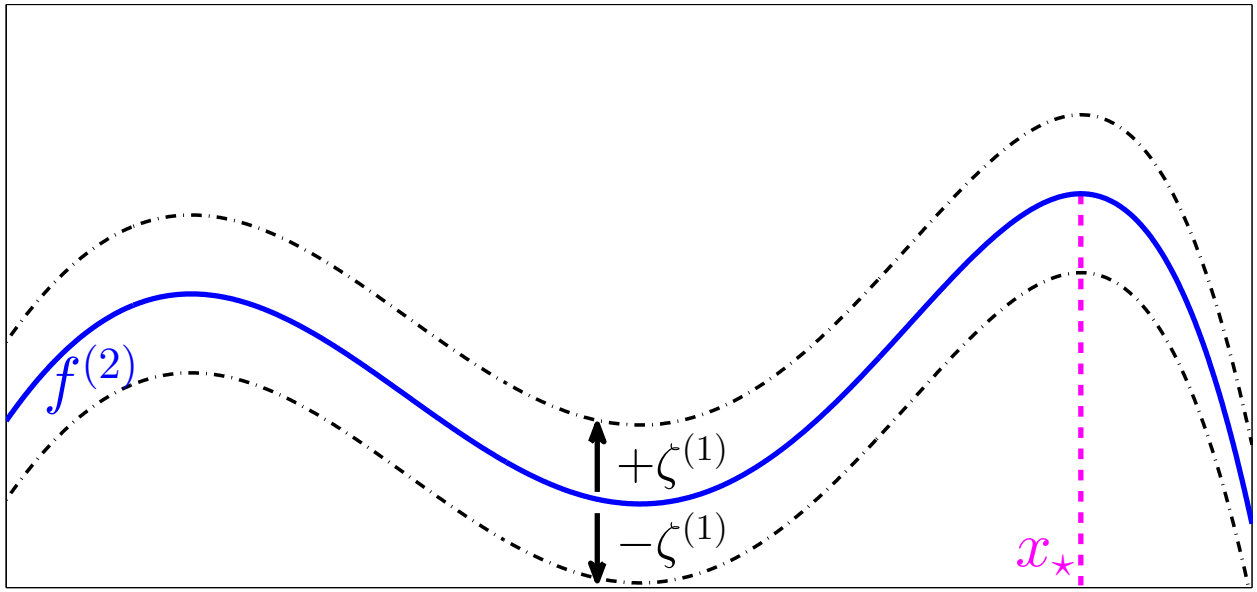
Capital $\Lambda \leftarrow$ amount of the resource spent. E.g. seconds or dollars.

No reward for querying $f^{(1)}$, but use cheap evaluations to guide search for $x_\star$ at $f^{(2)}$.

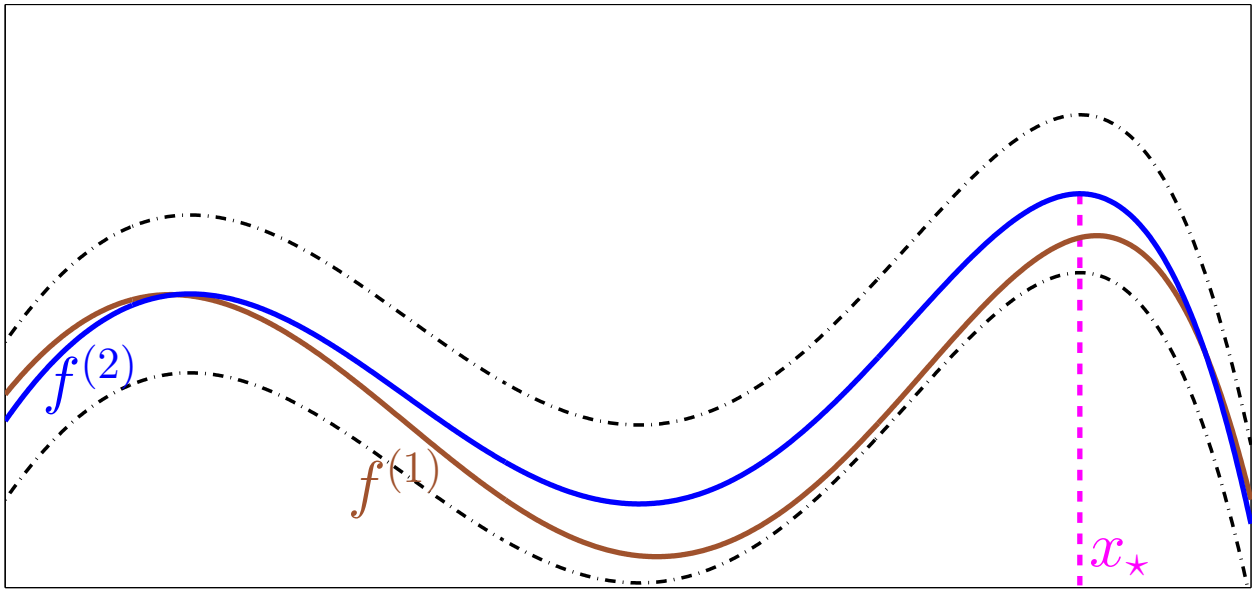
Challenges



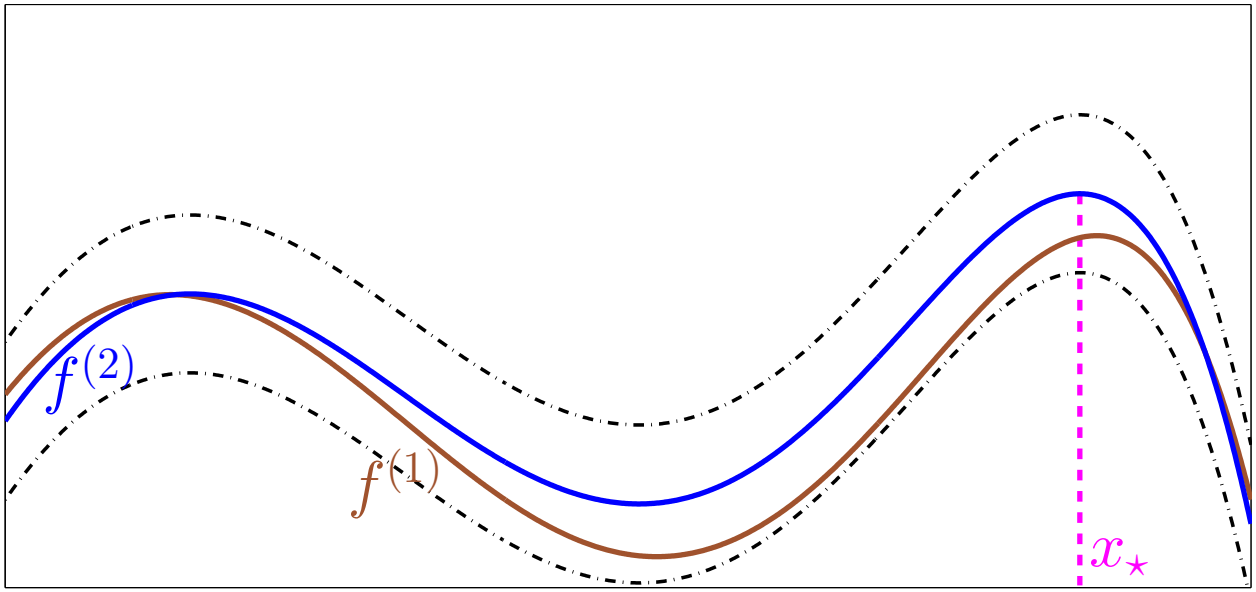
Challenges



Challenges

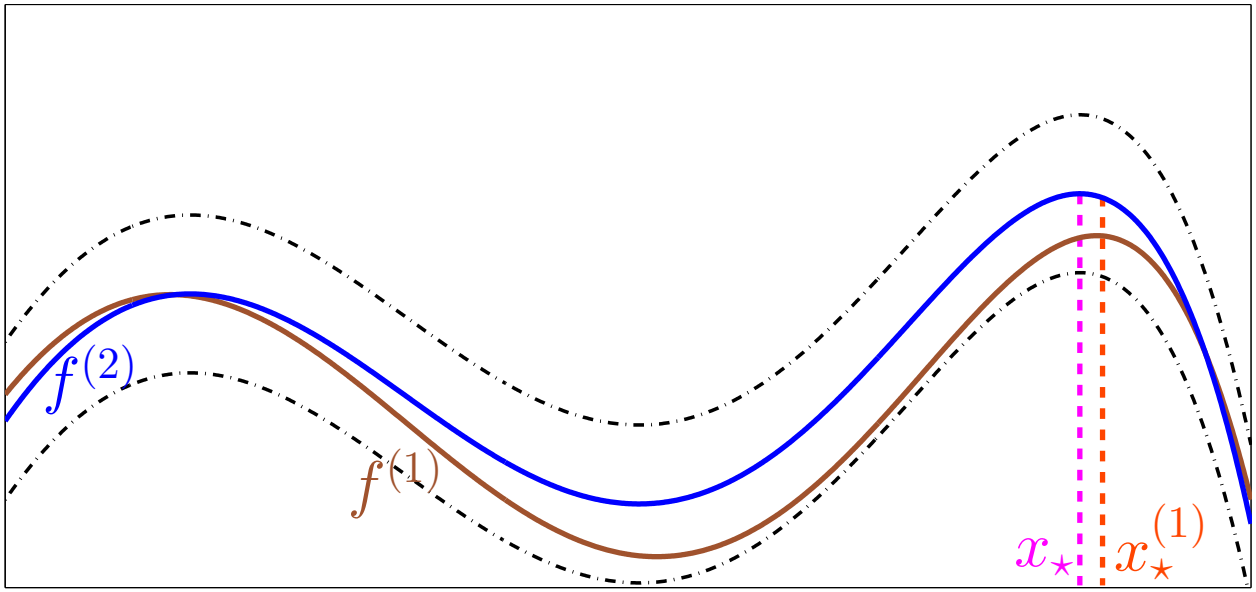


Challenges



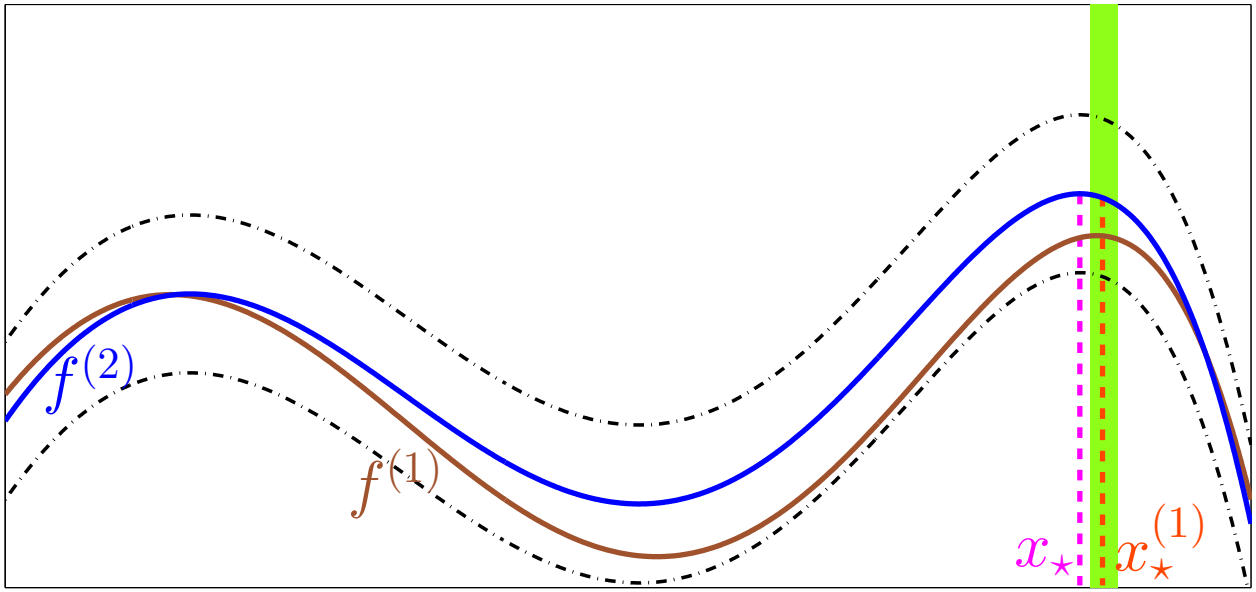
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.

Challenges



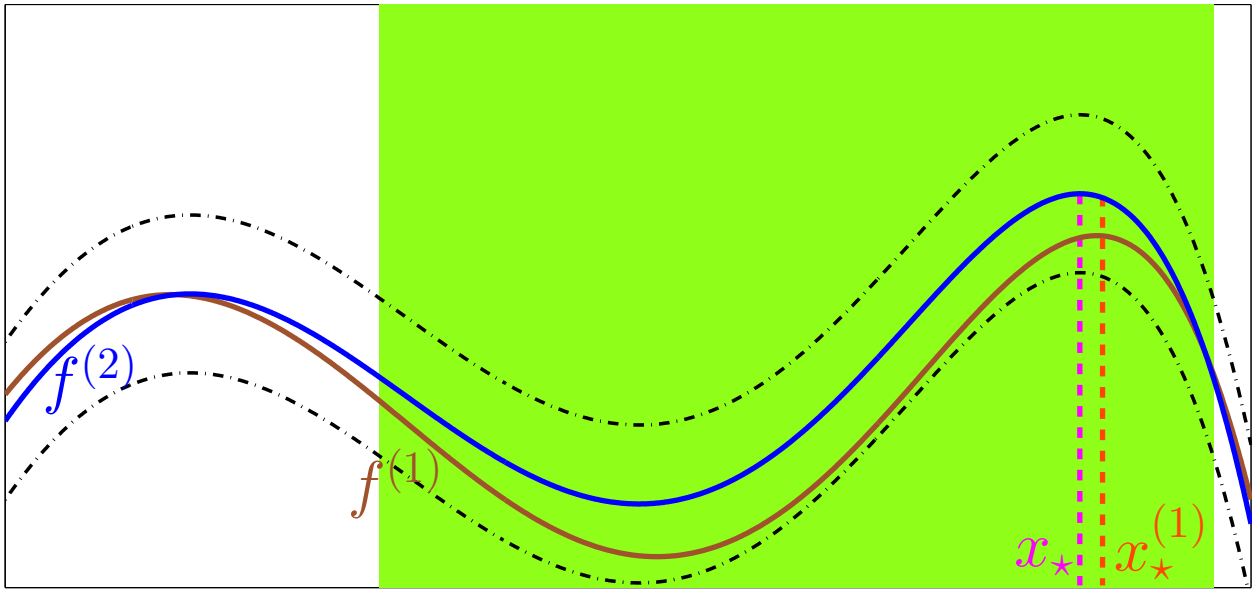
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.

Challenges



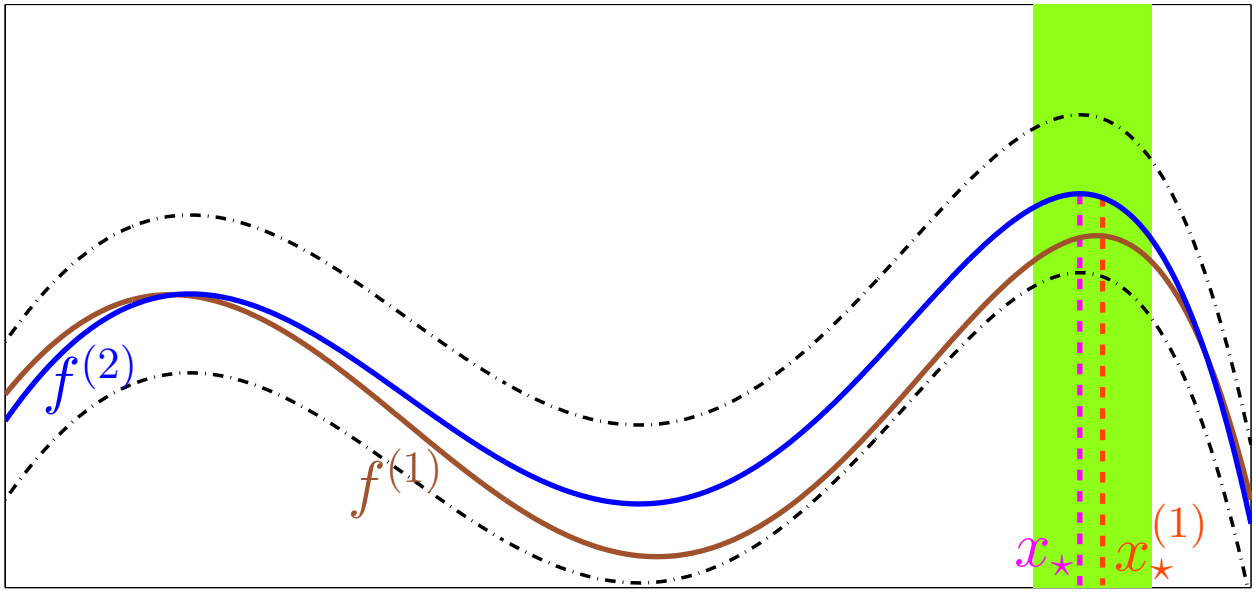
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.

Challenges



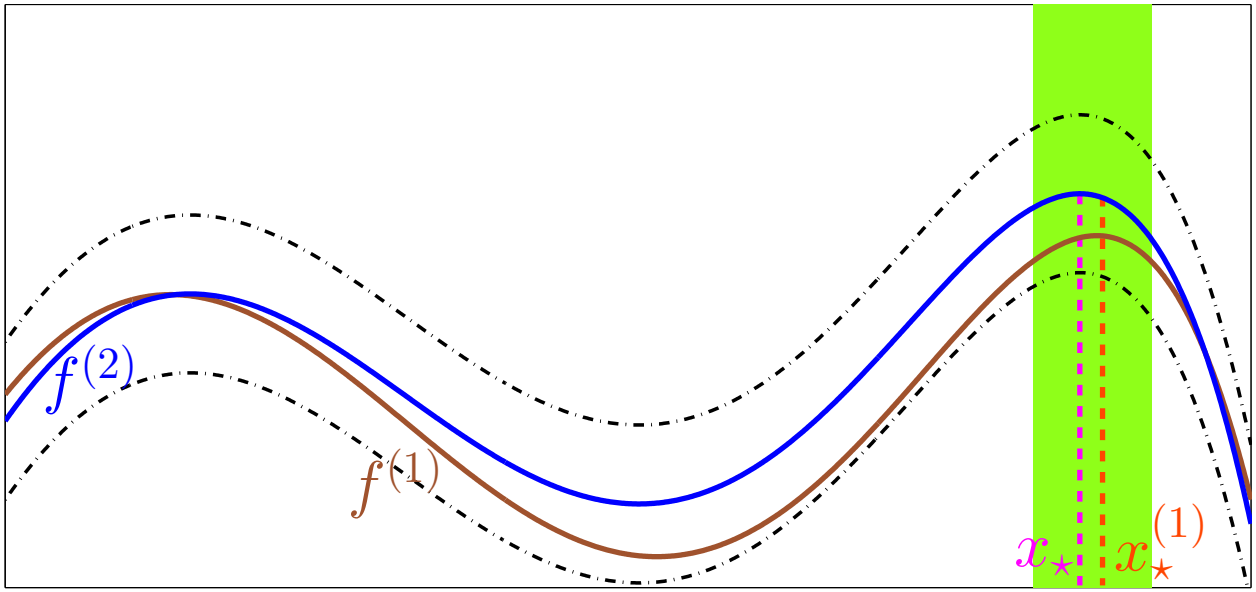
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.

Challenges



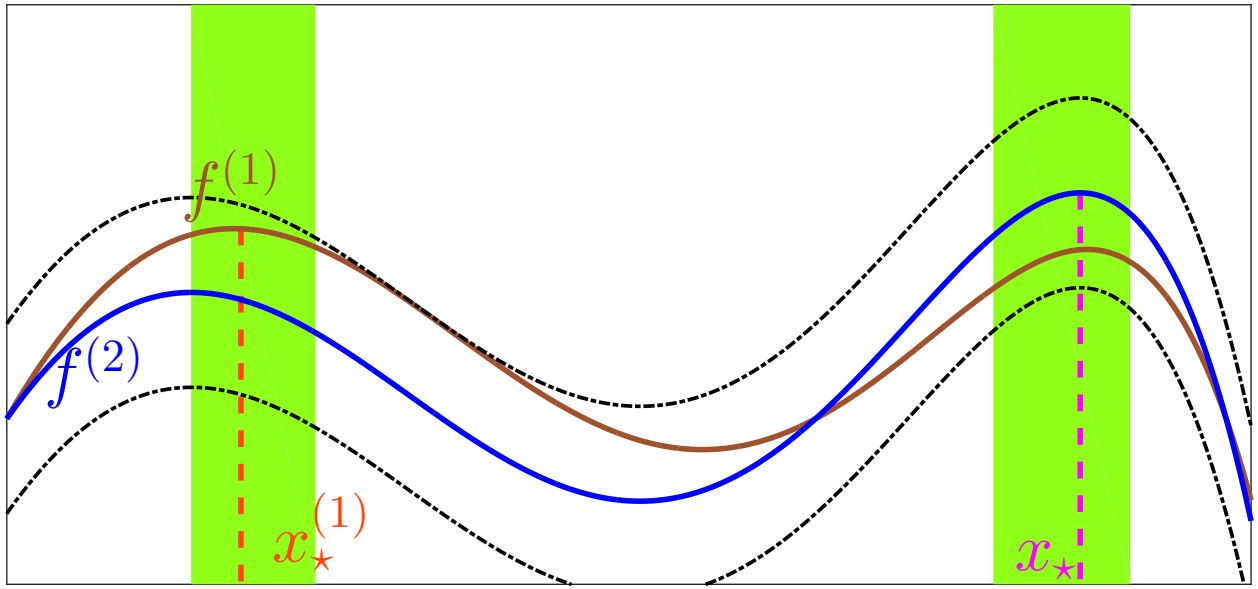
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.

Challenges



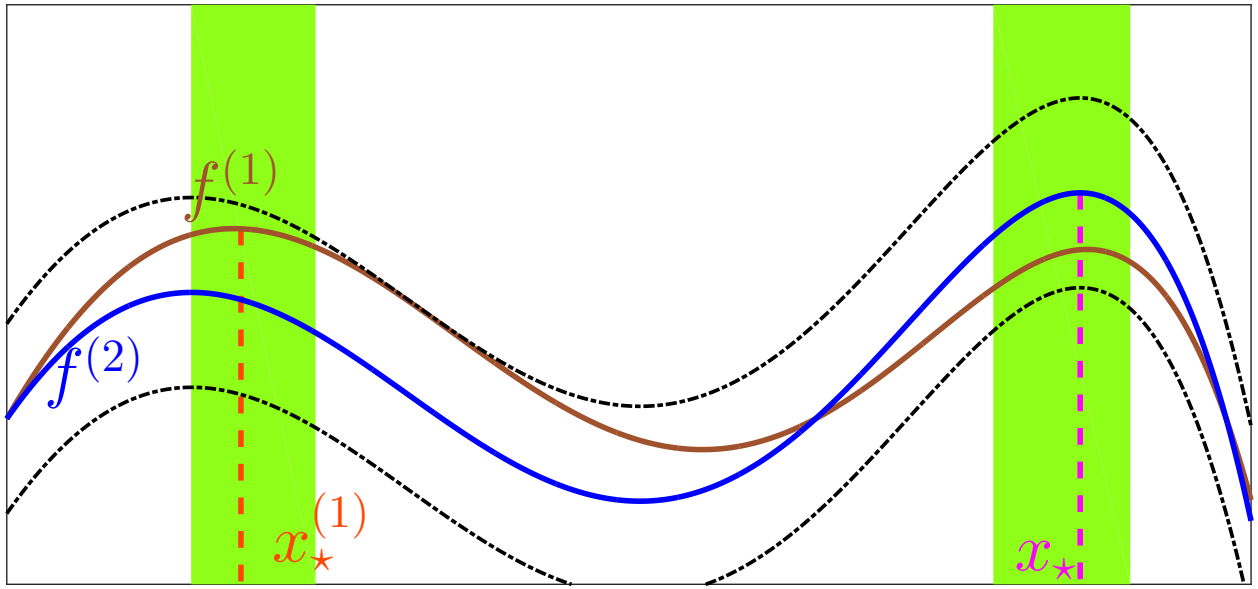
- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.
- Need to explore $f^{(2)}$ sufficiently well around the *high valued regions* of $f^{(1)}$ – but at a not too large region.

Challenges



- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.
- Need to explore $f^{(2)}$ sufficiently well around the *high valued regions* of $f^{(1)}$ – but at a not too large region.

Challenges

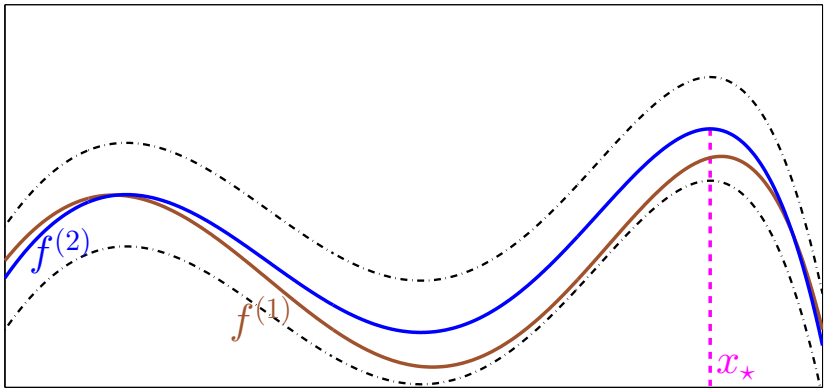


- $f^{(1)}$ is not just a noisy version of $f^{(2)}$.
- Cannot just maximise $f^{(1)}$. $x_\star^{(1)}$ is suboptimal for $f^{(2)}$.
- Need to explore $f^{(2)}$ sufficiently well around the *high valued regions* of $f^{(1)}$ – but at a not too large region.

Key Message: We will explore $\mathcal{X}$ using $f^{(1)}$ and use $f^{(2)}$ *mostly* in a promising region $\mathcal{X}_\alpha$.

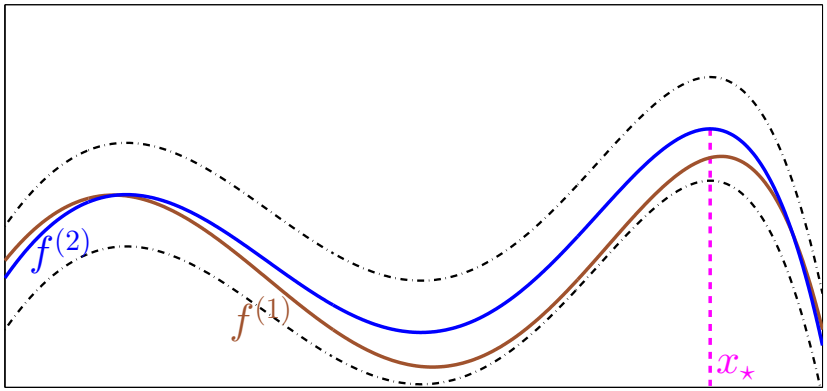
MF-GP-UCB (Kandasamy et al. NIPS 2016b)

Multi-fidelity Gaussian Process Upper Confidence Bound



MF-GP-UCB (Kandasamy et al. NIPS 2016b)

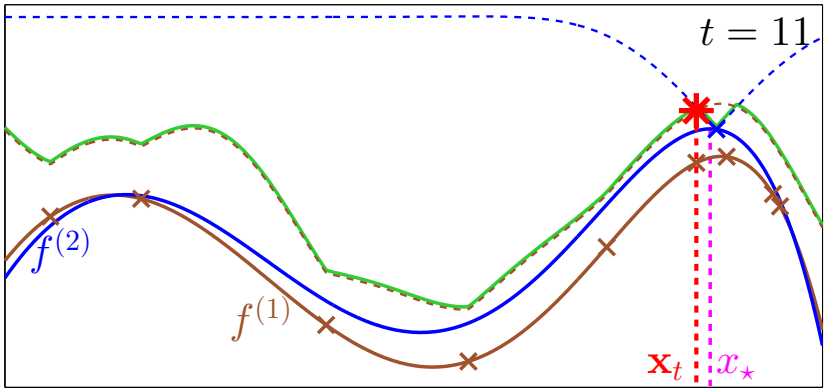
Multi-fidelity Gaussian Process Upper Confidence Bound



- Construct Upper Confidence Bound φ_t for $f^{(2)}$.
  Choose point $x_t = \mathrm{argmax}_{x \in \mathcal{X}} \varphi_t(x)$.

MF-GP-UCB (Kandasamy et al. NIPS 2016b)

Multi-fidelity Gaussian Process Upper Confidence Bound



- Construct Upper Confidence Bound φ_t for $f^{(2)}$.

  Choose point $x_t = \mathrm{argmax}_{x \in \mathcal{X}} \varphi_t(x)$.

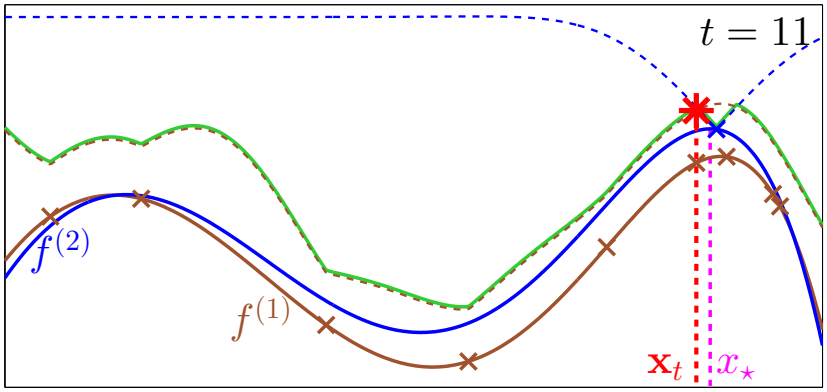
$$\varphi_t^{(1)}(x) = \boxed{\mu_{t-1}^{(1)}(x) + \beta_t^{1/2} \sigma_{t-1}^{(1)}(x)} + \zeta^{(1)}$$

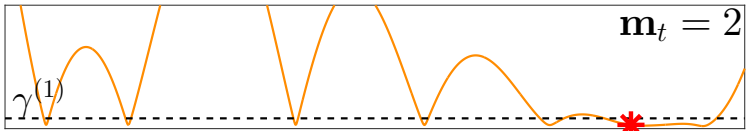
$$\varphi_t^{(2)}(x) = \mu_{t-1}^{(2)}(x) + \beta_t^{1/2} \sigma_{t-1}^{(2)}(x)$$

$$\varphi_t(x) = \min\{\, \varphi_t^{(1)}(x),\ \varphi_t^{(2)}(x) \,\}$$

MF-GP-UCB (Kandasamy et al. NIPS 2016b)

Multi-fidelity Gaussian Process Upper Confidence Bound





- Construct Upper Confidence Bound φ_t for $f^{(2)}$.

  Choose point $x_t = \mathrm{argmax}_{x \in \mathcal{X}} \varphi_t(x)$.

$$\varphi_t^{(1)}(x) = \mu_{t-1}^{(1)}(x) + \beta_t^{1/2} \sigma_{t-1}^{(1)}(x) + \zeta^{(1)}$$

$$\varphi_t^{(2)}(x) = \mu_{t-1}^{(2)}(x) + \beta_t^{1/2} \sigma_{t-1}^{(2)}(x)$$

$$\varphi_t(x) = \min\{ \varphi_t^{(1)}(x),\ \varphi_t^{(2)}(x) \}$$

- Choose fidelity $m_t = \begin{cases} 1 & \text{if } \beta_t^{1/2} \sigma_{t-1}^{(1)}(x_t) > \gamma^{(1)} \\ 2 & \text{otherwise.} \end{cases}$

Theoretical Results for MF-GP-UCB

GP-UCB, κ is an SE kernel (Srinivas et al. 2010)

w.h.p $$S(\Lambda) = f^{(2)}(x_\star) - \max_{t\,:\,m_t=2} f^{(2)}(x_t) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda}}$$

Theoretical Results for MF-GP-UCB

GP-UCB, κ is an SE kernel (Srinivas et al. 2010)

w.h.p $$S(\Lambda) = f^{(2)}(x_\star) - \max_{t\,:\,m_t=2} f^{(2)}(x_t) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda}}$$

MF-GP-UCB, κ is an SE kernel (Kandasamy et al. NIPS 2016b)

w.h.p $$\forall \alpha > 0, \quad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X}_\alpha)}{\Lambda}} + \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda^{2-\alpha}}}$$

$$\mathcal{X}_\alpha = \{x : f^{(2)}(x_\star) - f^{(1)}(x) \leq C_\alpha \zeta^{(1)}\}$$

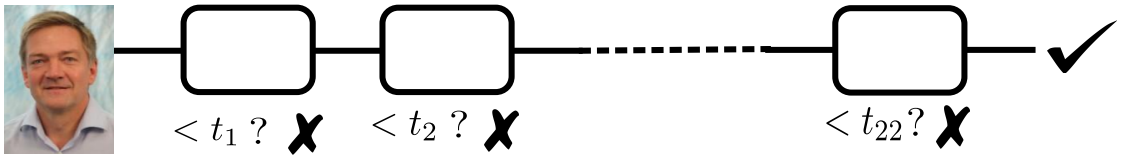
Good approximation (small $\zeta^{(1)}$) $\implies \mathrm{vol}(\mathcal{X}_\alpha) \ll \mathrm{vol}(\mathcal{X})$.

MF-GP-UCB with multiple approximations

MF-GP-UCB with multiple approximations

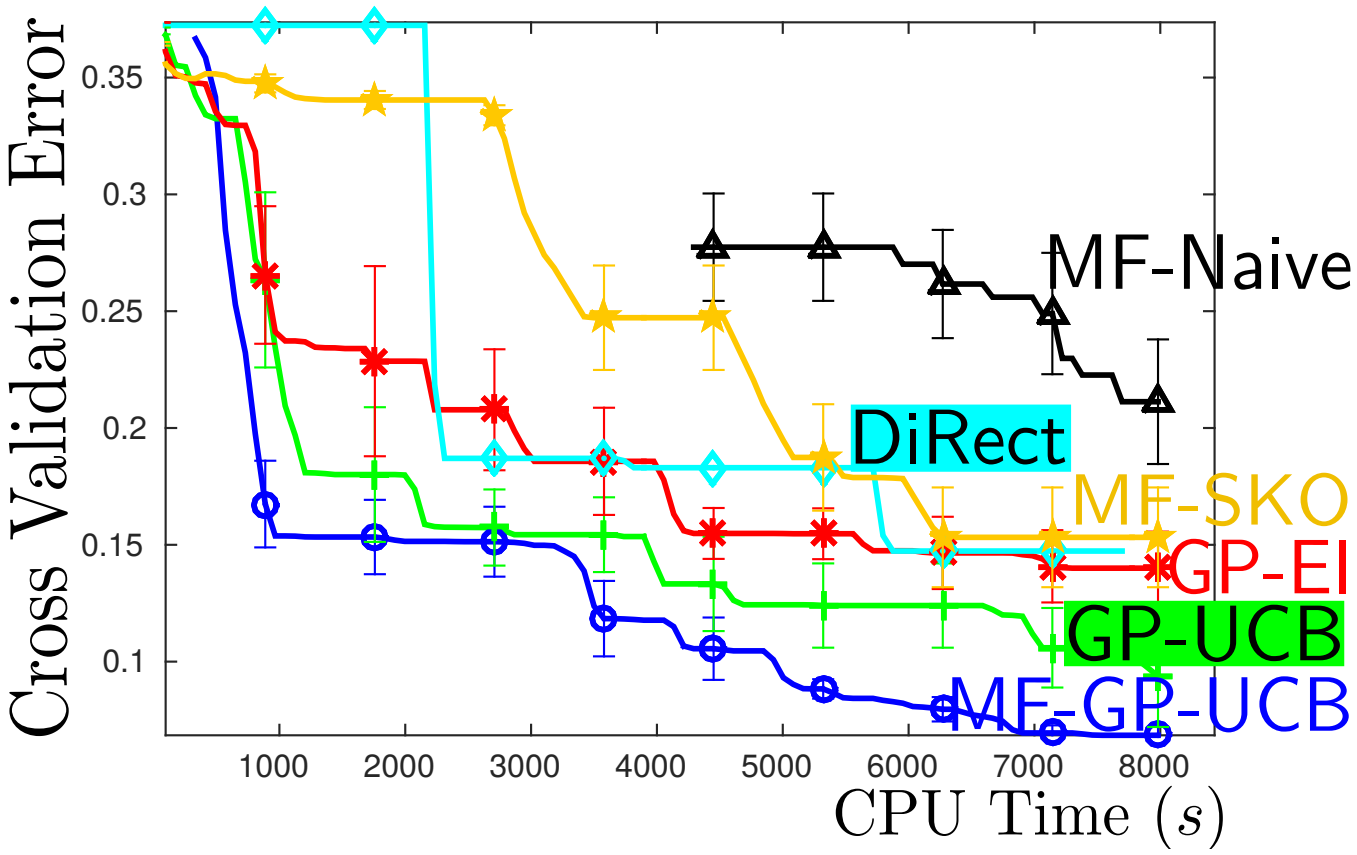
Things work out.

Experiment: Viola & Jones Face Detection



22 Threshold values for each cascade. $(d = 22)$

Fidelities with dataset sizes $(300, 3000)$. $(M = 2)$



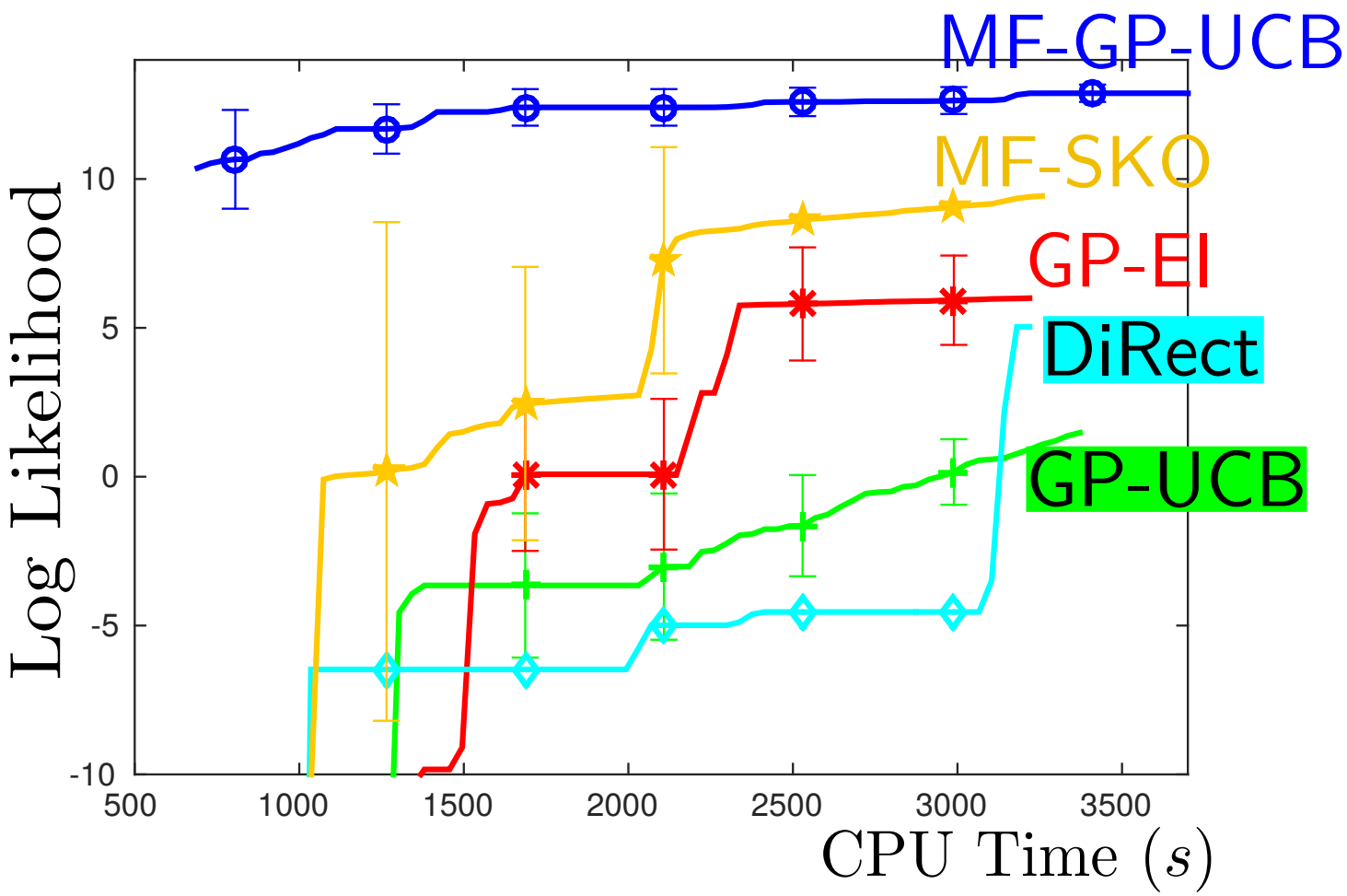
Experiment: Cosmological Maximum Likelihood Inference

- Type Ia Supernovae Data
- Maximum likelihood inference for 3 cosmological parameters:
  - Hubble Constant H_0
  - Dark Energy Fraction Ω_Λ
  - Dark Matter Fraction Ω_M
- Likelihood: Robertson Walker metric (Robertson 1936)
  Requires numerical integration for each point in the dataset.

Experiment: Cosmological Maximum Likelihood Inference

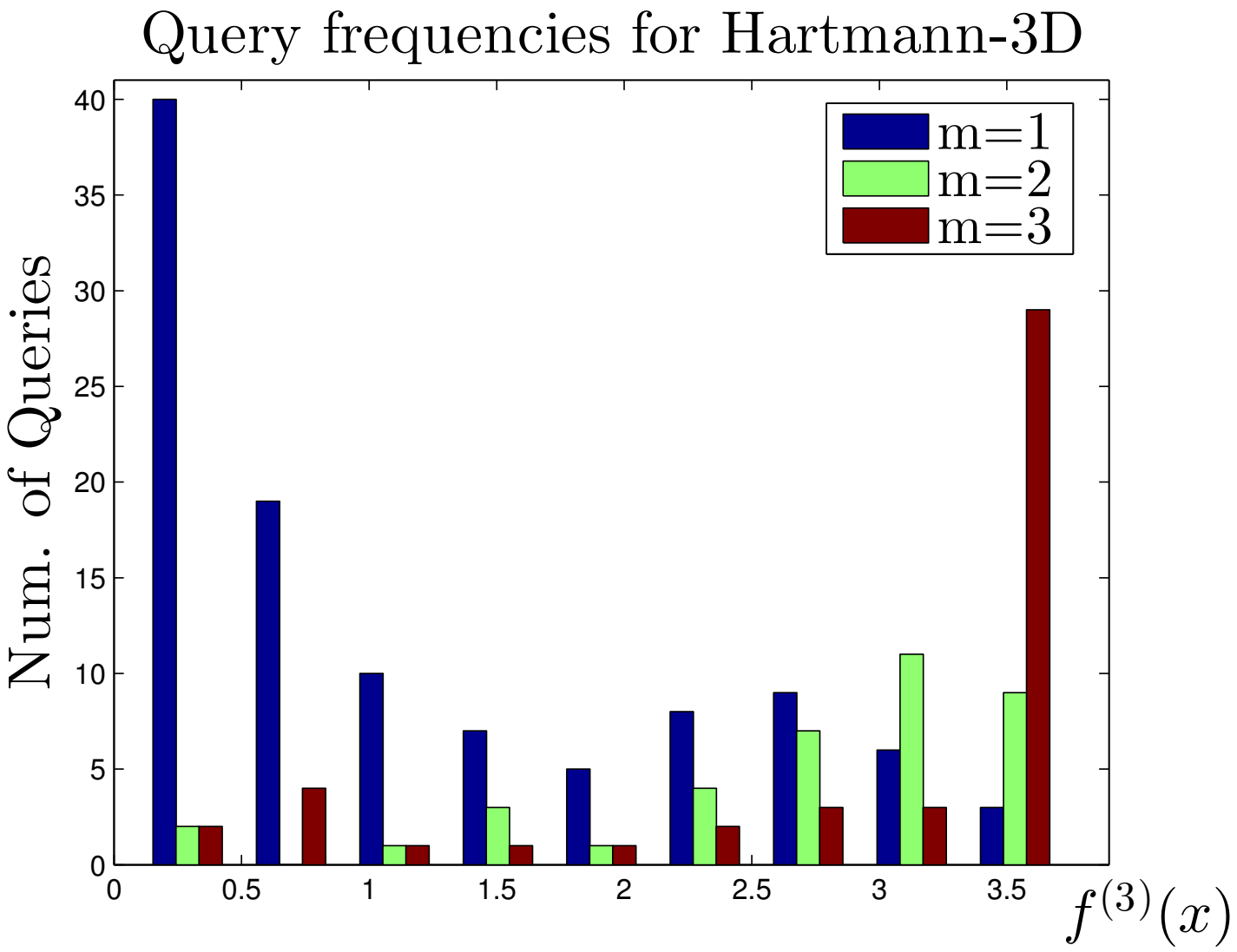
3 cosmological parameters. $(d = 3)$

Fidelities: integration on grids of size $(10^2, 10^4, 10^6)$. $(M = 3)$



MF-GP-UCB Synthetic Experiment: Hartmann-3D

$d = 3, \quad M = 3$



Outline

1. A finite number of approximations

(Kandasamy et al. NIPS 2016b)

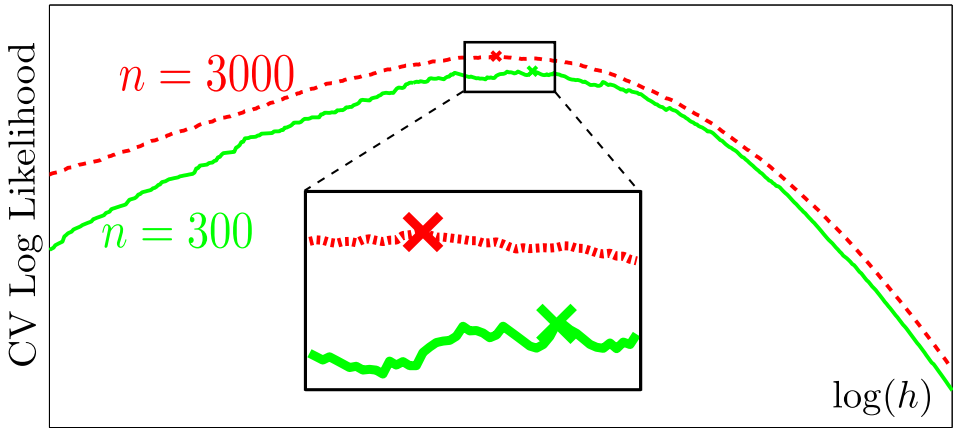
- Formalism, intuition and challenges
- Algorithm
- Theoretical results
- Experiments

2. A continuous spectrum of approximations

(Kandasamy et al. ICML 2017)

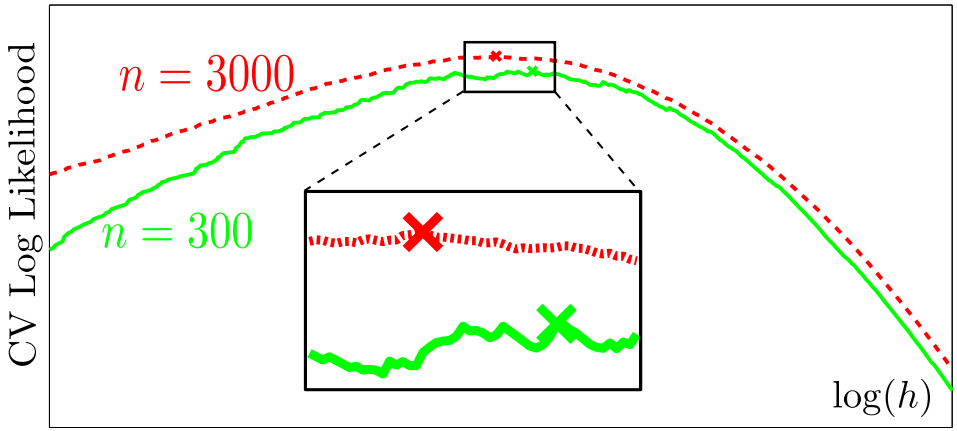
- Formalism
- Algorithm
- Theoretical results
- Experiments

Why continuous approximations?



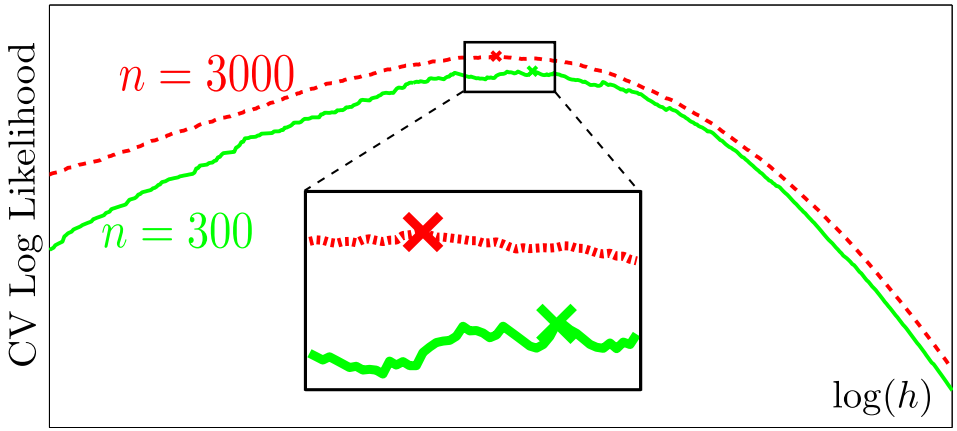
- Use an arbitrary amount of data?

Why continuous approximations?



- Use an arbitrary amount of data?
- Iterative algorithms: use arbitrary number of iterations?

Why continuous approximations?

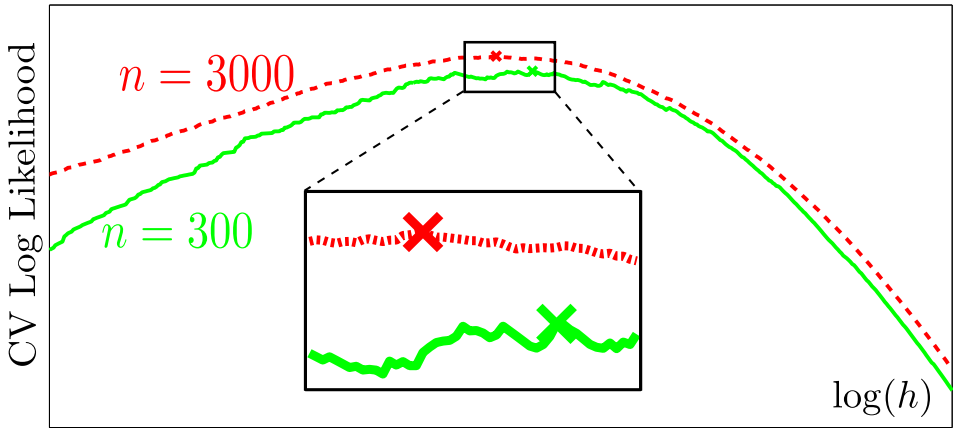


- Use an arbitrary amount of data?
- Iterative algorithms: use arbitrary number of iterations?

E.g. Train an ML model with $N_\bullet$ data and $T_\bullet$ iterations.

- But use $N < N_\bullet$ data and $T < T_\bullet$ iterations to approximate cross validation performance at $(N_\bullet, T_\bullet)$.

Why continuous approximations?



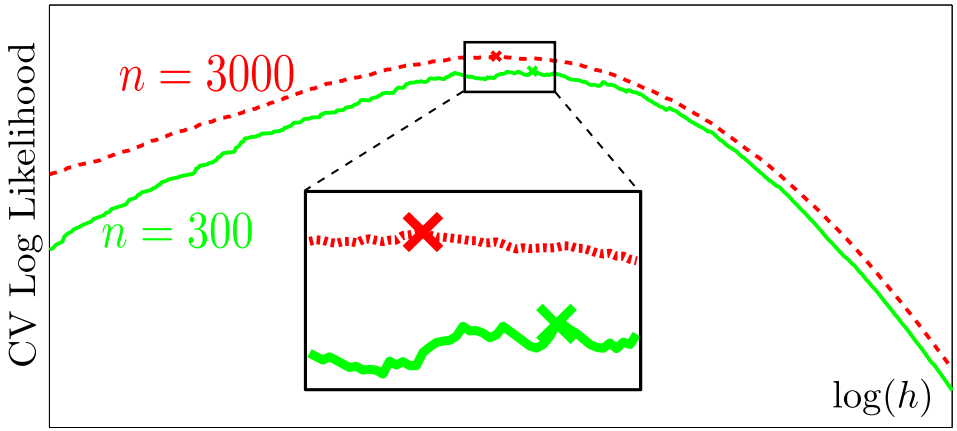
- Use an arbitrary amount of data?
- Iterative algorithms: use arbitrary number of iterations?

E.g. Train an ML model with $N_\bullet$ data and $T_\bullet$ iterations.

- But use $N < N_\bullet$ data and $T < T_\bullet$ iterations to approximate cross validation performance at $(N_\bullet, T_\bullet)$.

Approximations from a *continuous* 2D "fidelity space" (N, T).

Why continuous approximations?



- Use an arbitrary amount of data?
- Iterative algorithms: use arbitrary number of iterations?

E.g. Train an ML model with $N_\bullet$ data and $T_\bullet$ iterations.

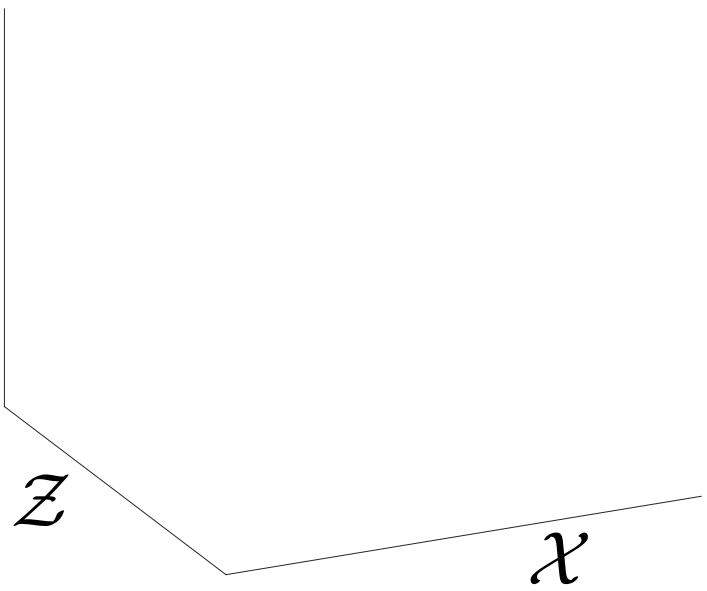
- But use $N < N_\bullet$ data and $T < T_\bullet$ iterations to approximate cross validation performance at $(N_\bullet, T_\bullet)$.

Approximations from a *continuous* 2D "fidelity space" (N, T).

Scientific studies: Simulations and numerical computations at varying *continuous* levels of granularity.

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



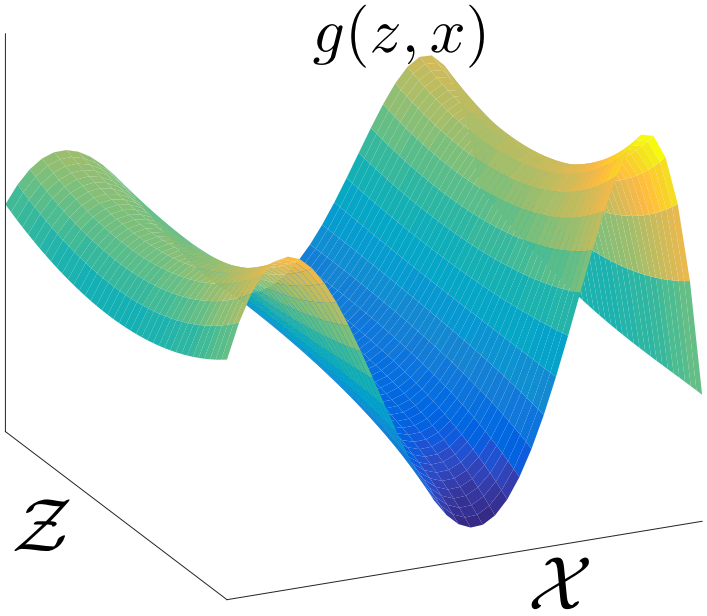
A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ all (N, T) values.

$\mathcal{X} \leftarrow$ all hyper-parameter values.

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ all (N, T) values.

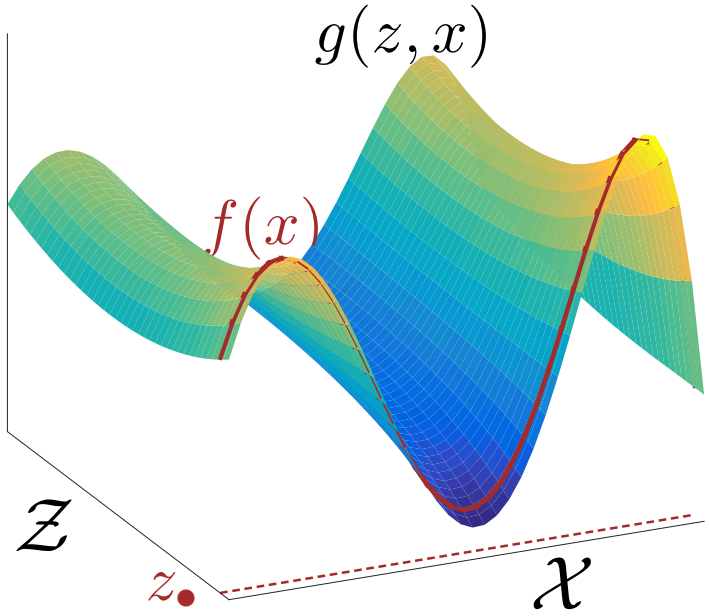
$\mathcal{X} \leftarrow$ all hyper-parameter values.

$g : \mathcal{Z} \times \mathcal{X} \rightarrow \mathbb{R}$.

$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ all (N, T) values.

$\mathcal{X} \leftarrow$ all hyper-parameter values.

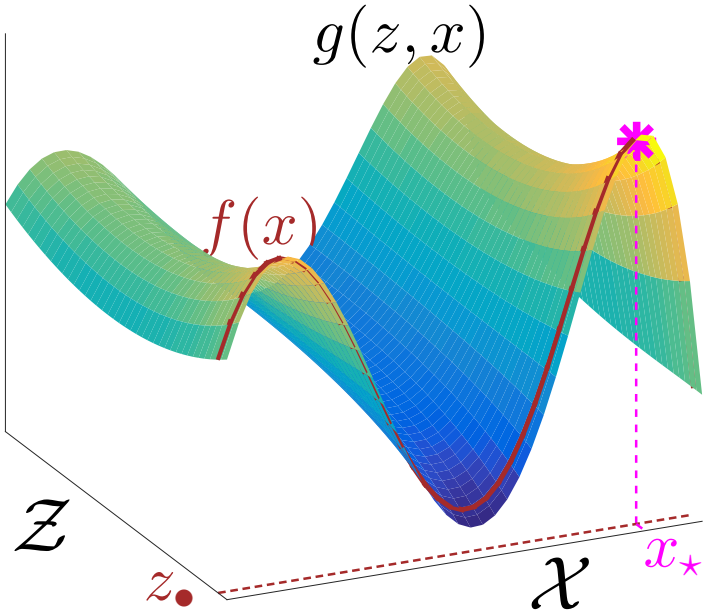
$g : \mathcal{Z} \times \mathcal{X} \rightarrow \mathbb{R}$.

$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

We wish to optimise $f(x) = g(z_\bullet, x)$ where $z_\bullet \in \mathcal{Z}$. $z_\bullet = [N_\bullet, T_\bullet]$.

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ all (N, T) values.

$\mathcal{X} \leftarrow$ all hyper-parameter values.

$g : \mathcal{Z} \times \mathcal{X} \rightarrow \mathbb{R}$.

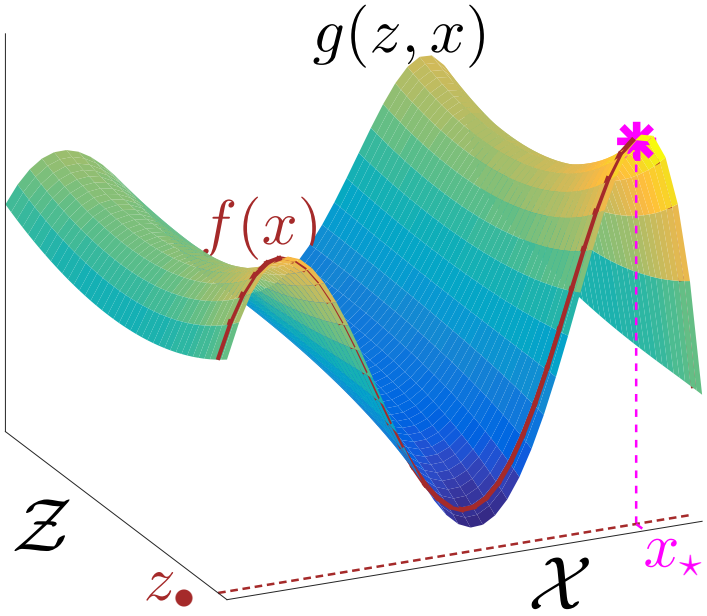
$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

We wish to optimise $f(x) = g(z_\bullet, x)$ where $z_\bullet \in \mathcal{Z}$. $z_\bullet = [N_\bullet, T_\bullet]$.

End Goal: Find $x_\star = \mathrm{argmax}_x f(x)$.

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ all (N, T) values.

$\mathcal{X} \leftarrow$ all hyper-parameter values.

$g : \mathcal{Z} \times \mathcal{X} \rightarrow \mathbb{R}$.

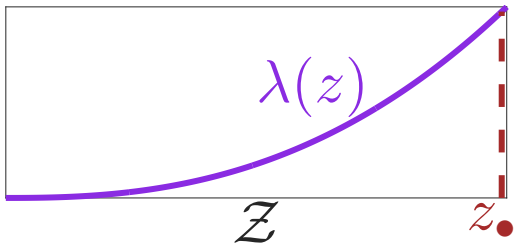
$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

We wish to optimise $f(x) = g(z_\bullet, x)$ where $z_\bullet \in \mathcal{Z}$. $z_\bullet = [N_\bullet, T_\bullet]$.

End Goal: Find $x_\star = \mathrm{argmax}_x f(x)$.

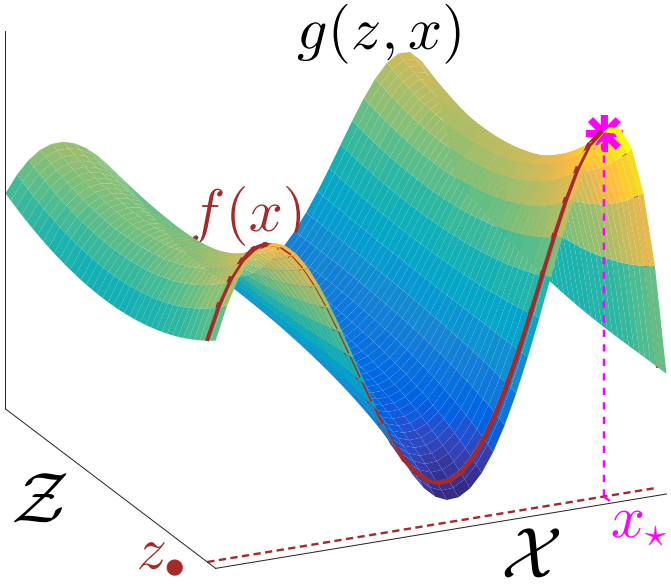
A cost function, $\lambda : \mathcal{Z} \rightarrow \mathbb{R}_+$.

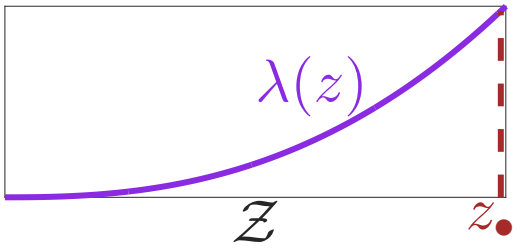
$\lambda(z) = \lambda(N, T) = \mathcal{O}(N^2 T)$.



Multi-fidelity Simple Regret

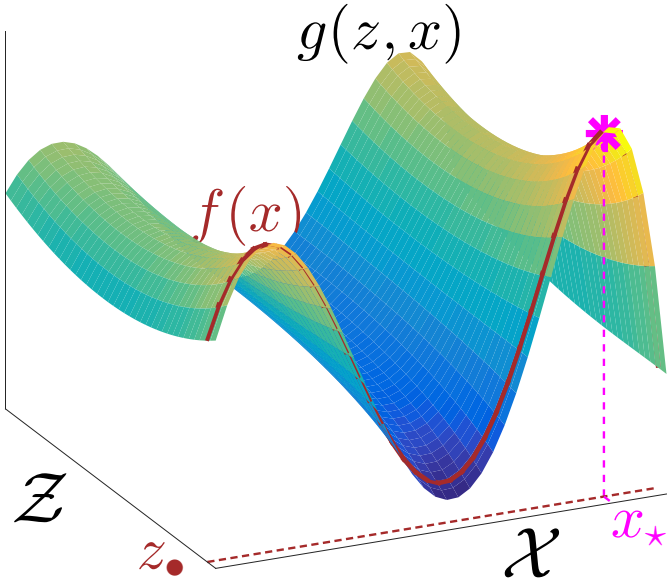
(Kandasamy et al. ICML 2017)

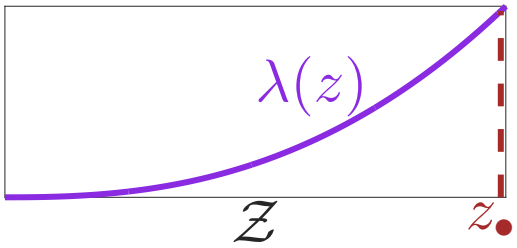




End Goal: Find $x_\star = \mathrm{argmax}_x f(x)$.

Multi-fidelity Simple Regret (Kandasamy et al. ICML 2017)



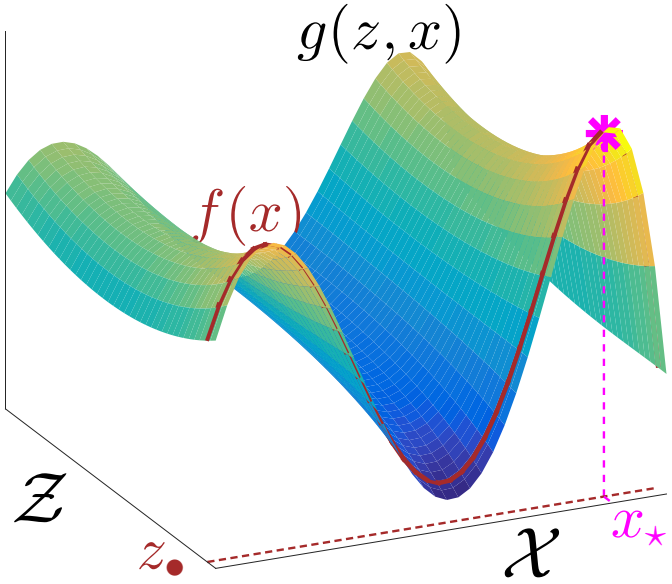


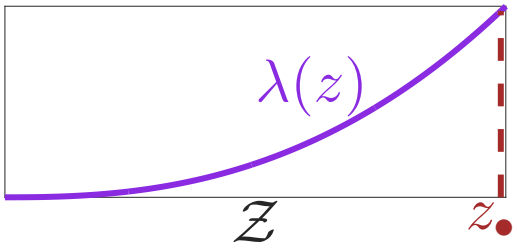
End Goal: Find $x_\star = \mathrm{argmax}_x f(x)$.

Simple Regret after *capital* Λ: $S(\Lambda) = f(x_\star) - \max_{t:\, z_t = z_\bullet} f(x_t)$.

$\Lambda \leftarrow$ amount of a resource spent, e.g. computation time or money.

Multi-fidelity Simple Regret (Kandasamy et al. ICML 2017)





End Goal: Find $x_\star = \mathrm{argmax}_x f(x)$.

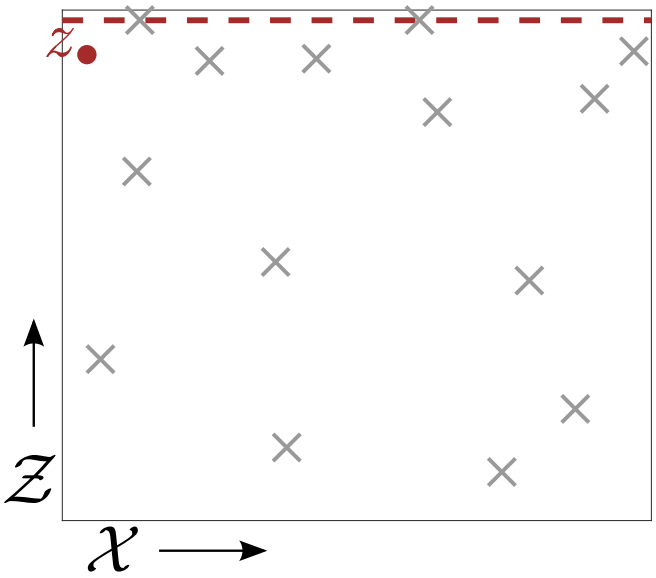
Simple Regret after *capital* Λ: $S(\Lambda) = f(x_\star) - \max_{t:\, z_t = z_\bullet} f(x_t)$.

$\Lambda \leftarrow$ amount of a resource spent, e.g. computation time or money.

No reward for maximising low fidelities, but use cheap evaluations at $z \neq z_\bullet$ to speed up search for $x_\star$.

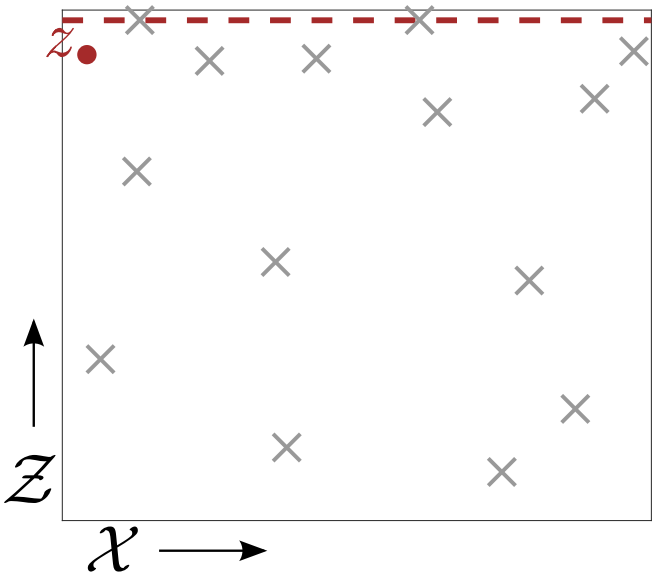
BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)

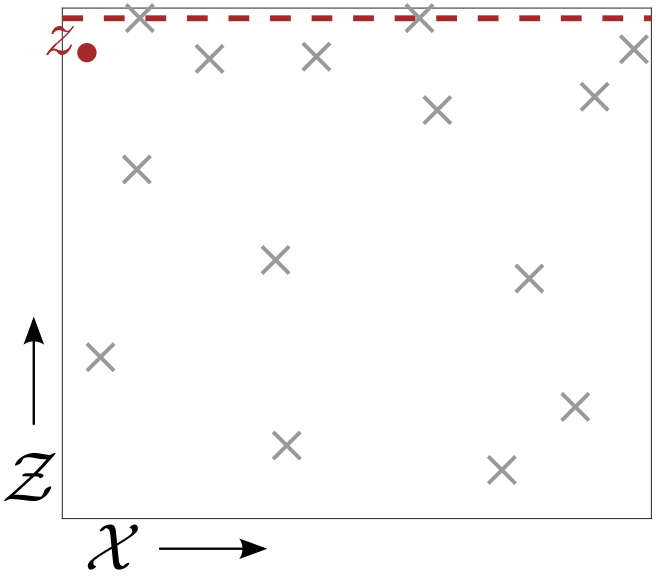


Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean	$\mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$
std-dev	$\sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

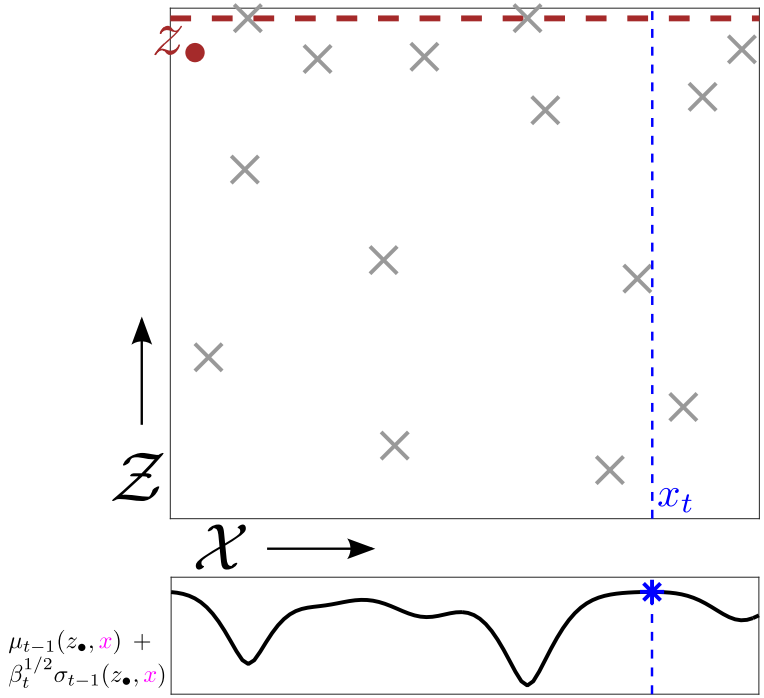
mean	$\mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$
std-dev	$\sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \ \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2} \sigma_{t-1}(z_\bullet, x)$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

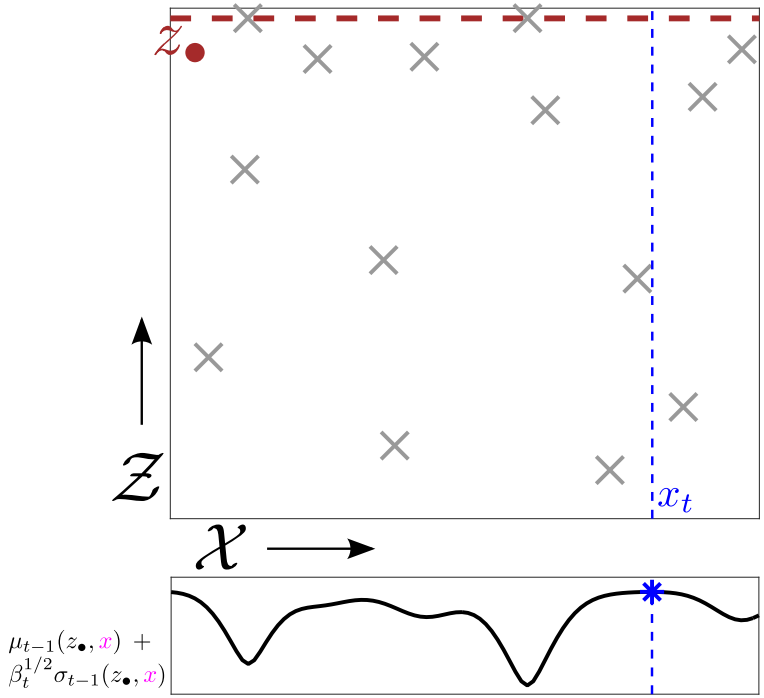
mean	$\mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$
std-dev	$\sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean $\quad \mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$

std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

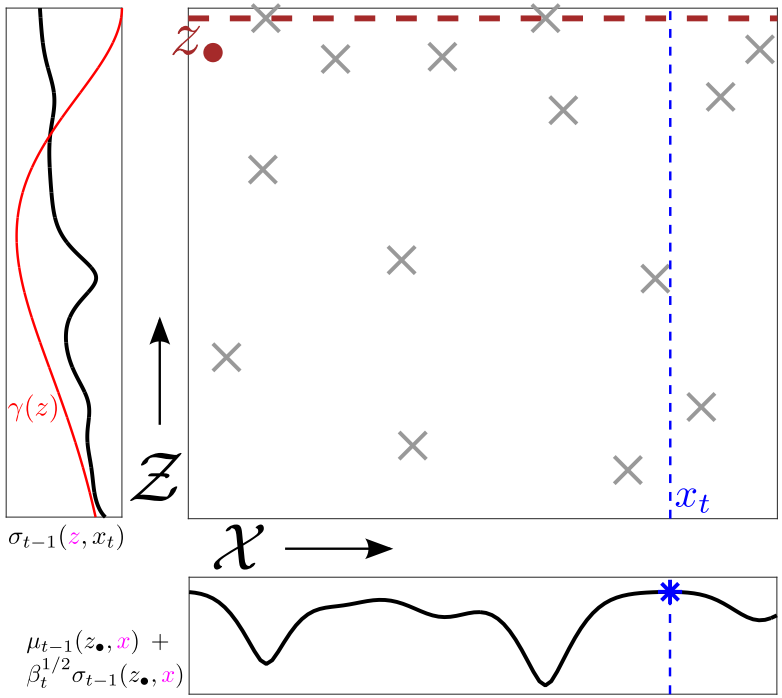
$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

(2) $$\mathcal{Z}_t \approx \{z_\bullet\} \cup \left\{ z : \sigma_{t-1}(z, x_t) \geq \gamma(z) \right\}$$

(3) $$z_t = \operatorname*{argmin}_{z \in \mathcal{Z}_t} \lambda(z) \qquad \text{(cheapest } z \text{ in } \mathcal{Z}_t\text{)}$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean $\quad \mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$

std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

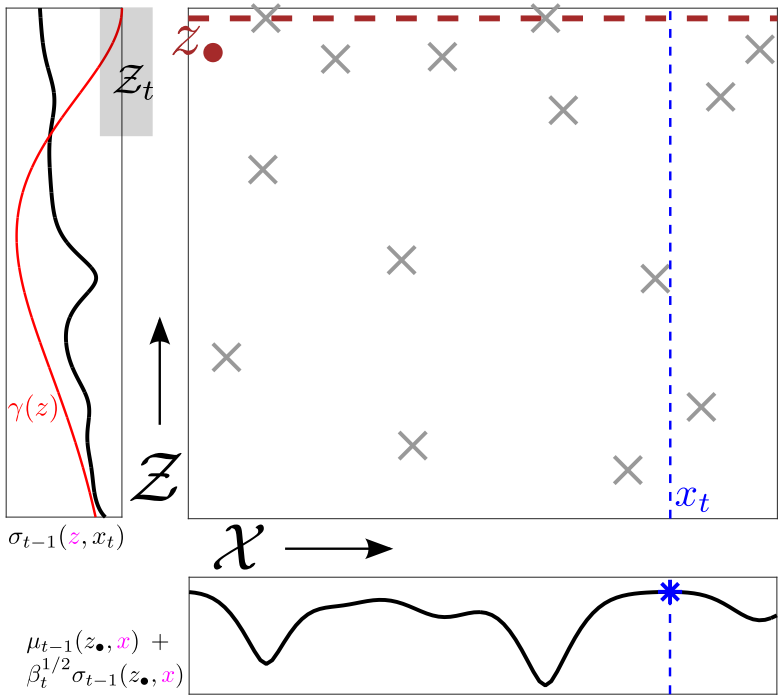
$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

(2) $$\mathcal{Z}_t \approx \{z_\bullet\} \cup \left\{ z : \sigma_{t-1}(z, x_t) \geq \gamma(z) \right\}$$

(3) $$z_t = \operatorname*{argmin}_{z \in \mathcal{Z}_t} \lambda(z) \qquad \text{(cheapest } z \text{ in } \mathcal{Z}_t\text{)}$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean $\quad \mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$

std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

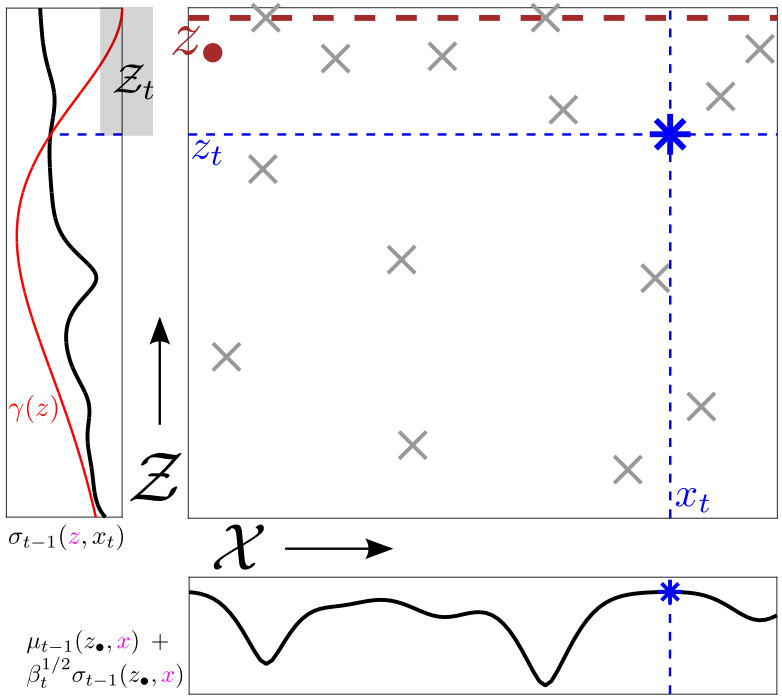
$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

(2) $$\mathcal{Z}_t \approx \{z_\bullet\} \cup \left\{ z : \sigma_{t-1}(z, x_t) \geq \gamma(z) \right\}$$

(3) $$z_t = \operatorname*{argmin}_{z \in \mathcal{Z}_t} \lambda(z) \qquad \text{(cheapest } z \text{ in } \mathcal{Z}_t\text{)}$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean $\quad \mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$

std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

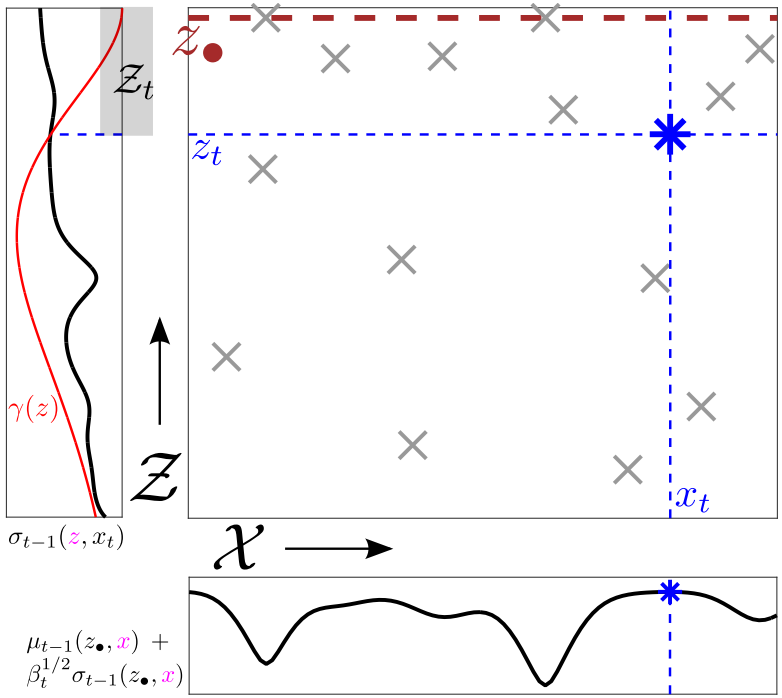
$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

(2) $$\mathcal{Z}_t \approx \{z_\bullet\} \cup \left\{ z : \sigma_{t-1}(z, x_t) \geq \gamma(z) \right\}$$

(3) $$z_t = \operatorname*{argmin}_{z \in \mathcal{Z}_t} \lambda(z) \qquad \text{(cheapest } z \text{ in } \mathcal{Z}_t\text{)}$$

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

mean $\quad \mu_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}$

std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

(1) $x_t \leftarrow$ maximise upper confidence bound for $f(x) = g(z_\bullet, x)$.

$$x_t = \operatorname*{argmax}_{x \in \mathcal{X}} \; \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

(2) $$\mathcal{Z}_t \approx \{z_\bullet\} \cup \left\{ z : \sigma_{t-1}(z, x_t) \geq \gamma(z) = \left(\frac{\lambda(z)}{\lambda(z_\bullet)}\right)^q \xi(z) \right\}$$

(3) $$z_t = \operatorname*{argmin}_{z \in \mathcal{Z}_t} \lambda(z) \qquad \text{(cheapest } z \text{ in } \mathcal{Z}_t\text{)}$$

Theoretical Results for BOCA

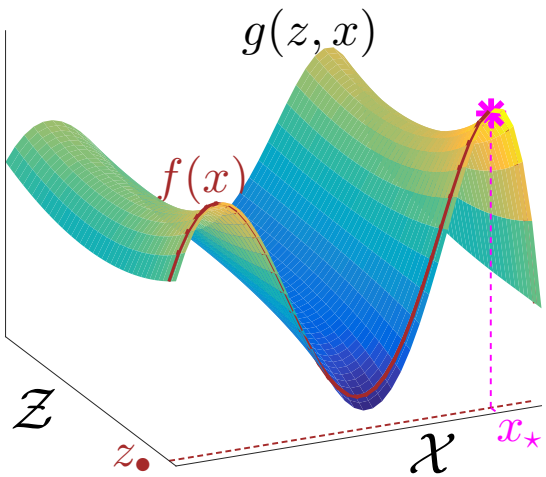
$$g \sim \mathcal{GP}(\mathbf{0}, \kappa), \quad \kappa : (\mathcal{Z} \times \mathcal{X})^2 \to \mathbb{R}.$$

$$\kappa([z, x], [z', x']) = \kappa_{\mathcal{X}}(x, x') \cdot \kappa_{\mathcal{Z}}(z, z')$$

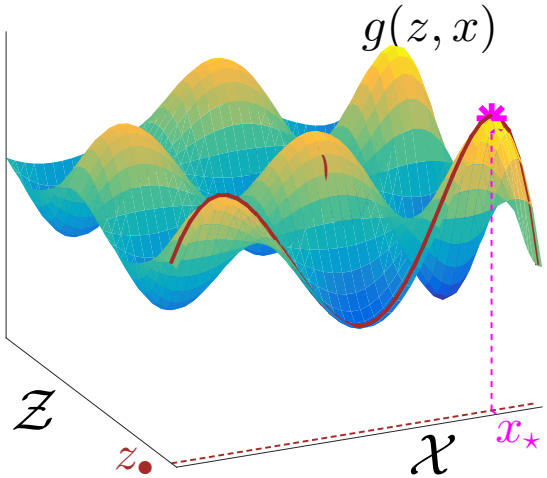
Theoretical Results for BOCA

$$g \sim \mathcal{GP}(\mathbf{0}, \kappa), \quad \kappa : (\mathcal{Z} \times \mathcal{X})^2 \to \mathbb{R}.$$

$$\kappa([z, x], [z', x']) = \kappa_{\mathcal{X}}(x, x') \cdot \kappa_{\mathcal{Z}}(z, z')$$



"good"

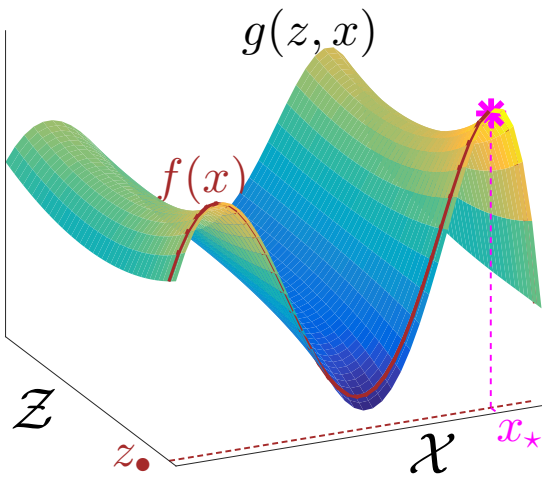


"bad"

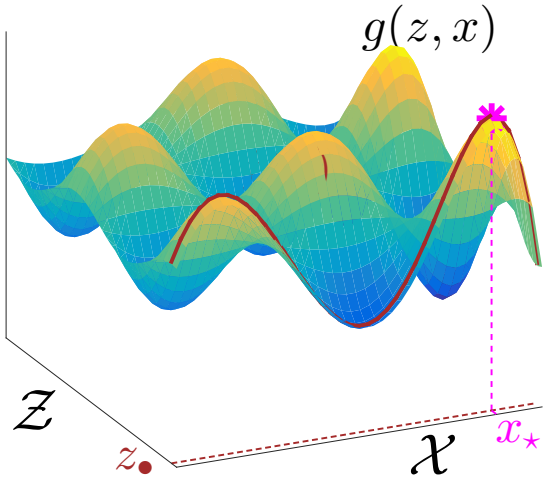
Theoretical Results for BOCA

$$g \sim \mathcal{GP}(\mathbf{0}, \kappa), \quad \kappa : (\mathcal{Z} \times \mathcal{X})^2 \to \mathbb{R}.$$

$$\kappa([z, x], [z', x']) = \kappa_{\mathcal{X}}(x, x') \cdot \kappa_{\mathcal{Z}}(z, z')$$



"good"
large $h_{\mathcal{Z}}$



"bad"
small $h_{\mathcal{Z}}$

E.g.: If $\kappa_{\mathcal{Z}}$ is an SE kernel, bandwidth $h_{\mathcal{Z}}$ controls smoothness.

Theoretical Results for BOCA

GP-UCB $\kappa_{\mathcal{X}}$ is an SE kernel, (Srinivas et al. 2010)

$$\text{w.h.p} \qquad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda}}$$

Theoretical Results for BOCA

GP-UCB $\kappa_{\mathcal{X}}$ is an SE kernel, (Srinivas et al. 2010)

$$\text{w.h.p} \quad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda}}$$

BOCA $\kappa_{\mathcal{X}}, \kappa_{\mathcal{Z}}$ are SE kernels, (Kandasamy et al. ICML 2017)

$$\text{w.h.p} \quad \forall \alpha > 0, \quad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X}_\alpha)}{\Lambda}} + \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda^{2-\alpha}}}$$

$$\mathcal{X}_\alpha = \left\{ x; \;\; f(x_\star) - f(x) \lesssim C_\alpha \frac{1}{h_{\mathcal{Z}}} \right\}$$

Theoretical Results for BOCA

GP-UCB $\kappa_{\mathcal{X}}$ is an SE kernel, (Srinivas et al. 2010)

$$\text{w.h.p} \quad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda}}$$

BOCA $\kappa_{\mathcal{X}}, \kappa_{\mathcal{Z}}$ are SE kernels, (Kandasamy et al. ICML 2017)

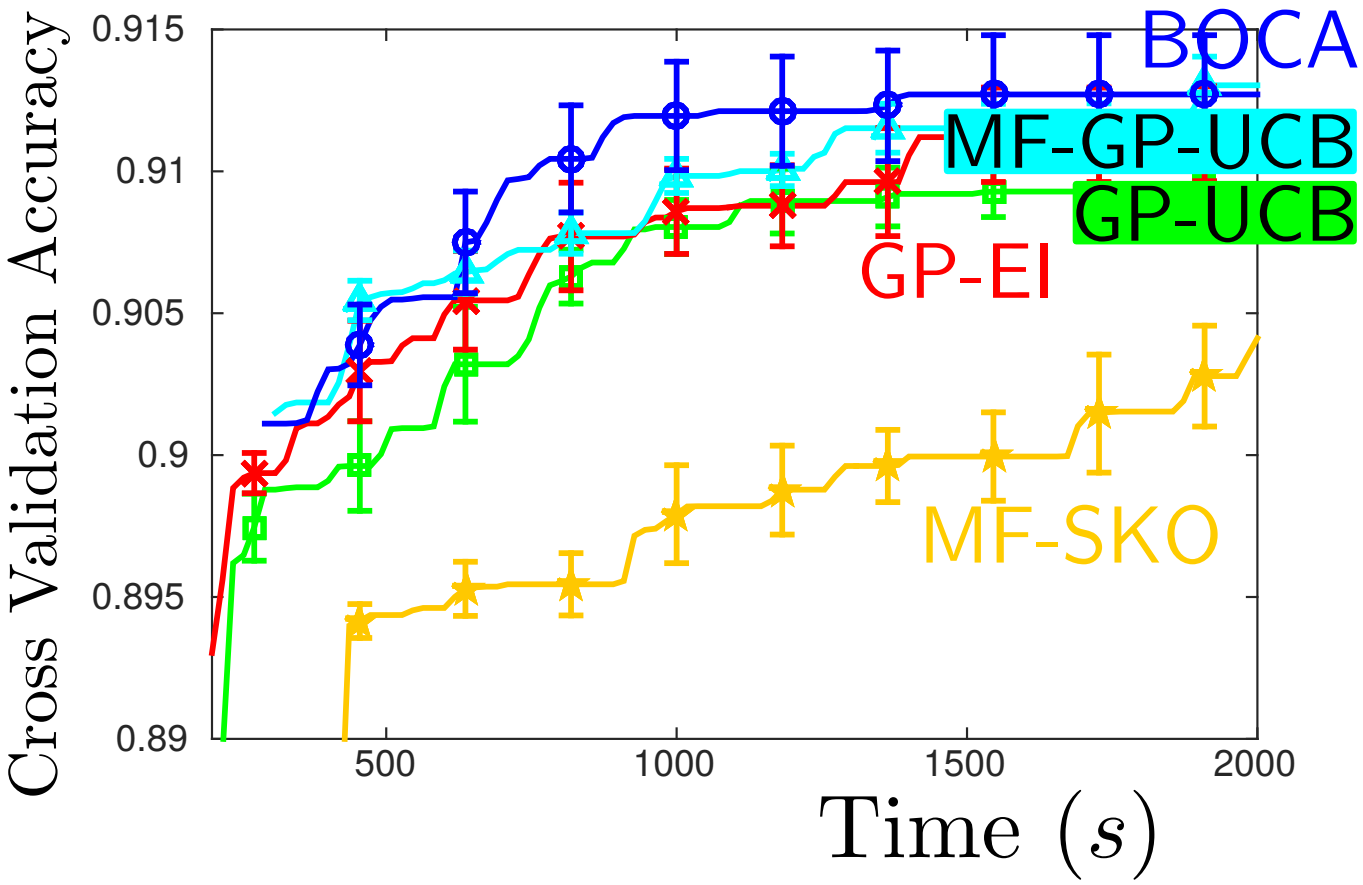
$$\text{w.h.p} \quad \forall \alpha > 0, \quad S(\Lambda) \lesssim \sqrt{\frac{\mathrm{vol}(\mathcal{X}_\alpha)}{\Lambda}} + \sqrt{\frac{\mathrm{vol}(\mathcal{X})}{\Lambda^{2-\alpha}}}$$

$$\mathcal{X}_\alpha = \left\{ x; \;\; f(x_\star) - f(x) \lesssim C_\alpha \frac{1}{h_{\mathcal{Z}}} \right\}$$

If $h_{\mathcal{Z}}$ is large (good approximations), $\mathrm{vol}(\mathcal{X}_\alpha) \ll \mathrm{vol}(\mathcal{X})$, and BOCA is much better than GP-UCB.

Experiment: SVM with 20 News Groups

Tune two hyper-parameters for the SVM.
Dataset has $N_\bullet = 15K$ data and use $T_\bullet = 100$ iterations.
But can choose $N \in [5K, 15K]$ or $T \in [20, 100]$ (2D fidelity space).



More synthetic & real experiments in the paper.

Open Questions, Challenges & Take-aways

- If you know the relationship between the approximations (fidelities), you should use it.
  Estimating it from data on the fly is not impossible, but difficult.

Open Questions, Challenges & Take-aways

- If you know the relationship between the approximations (fidelities), you should use it.
  Estimating it from data on the fly is not impossible, but difficult.

- There might be better/different models for the approximations that might suit your problem.
  - E.g. approximations that are good in certain regions but bad in other regions.

Summary

Multi-fidelity K-armed bandits (Kandasamy et al. NIPS 2016a)

- An algorithm MF-UCB and an upper bound on the regret.
- An almost matching lower bound.

Summary

Multi-fidelity K-armed bandits (Kandasamy et al. NIPS 2016a)

- An algorithm MF-UCB and an upper bound on the regret.
- An almost matching lower bound.

Key takeaways (Kandasamy et al. NIPS 2016a, Kandasamy et al. NIPS 2016b, Kandasamy et al. ICML 2017)

- Upper confidence bound strategy
- Choose higher fidelity only after controlling uncertainty at lower fidelities.
- Explore the entire space using cheap low fidelities and reserve expensive higher fidelities for promising candidates.
- Theoretically/empirically outperforms strategies which ignore the approximations.

Gautam Dasarathy

Junier Oliva

Jeff Schneider

Barnabas Poczos

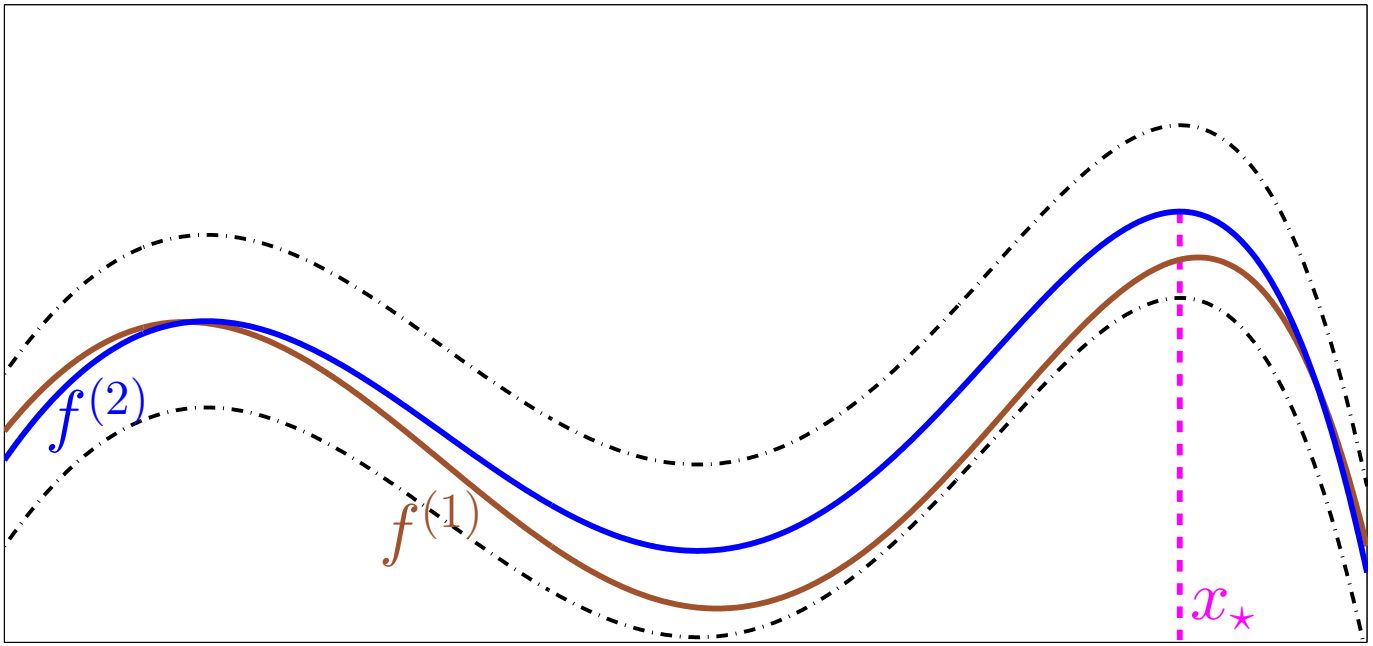
Thank you.

Code for MF-GP-UCB: `github.com/kirthevasank/mf-gp-ucb`

Slides: `www.cs.cmu.edu/∼kkandasa/talks/fb-mf-slides.pdf`

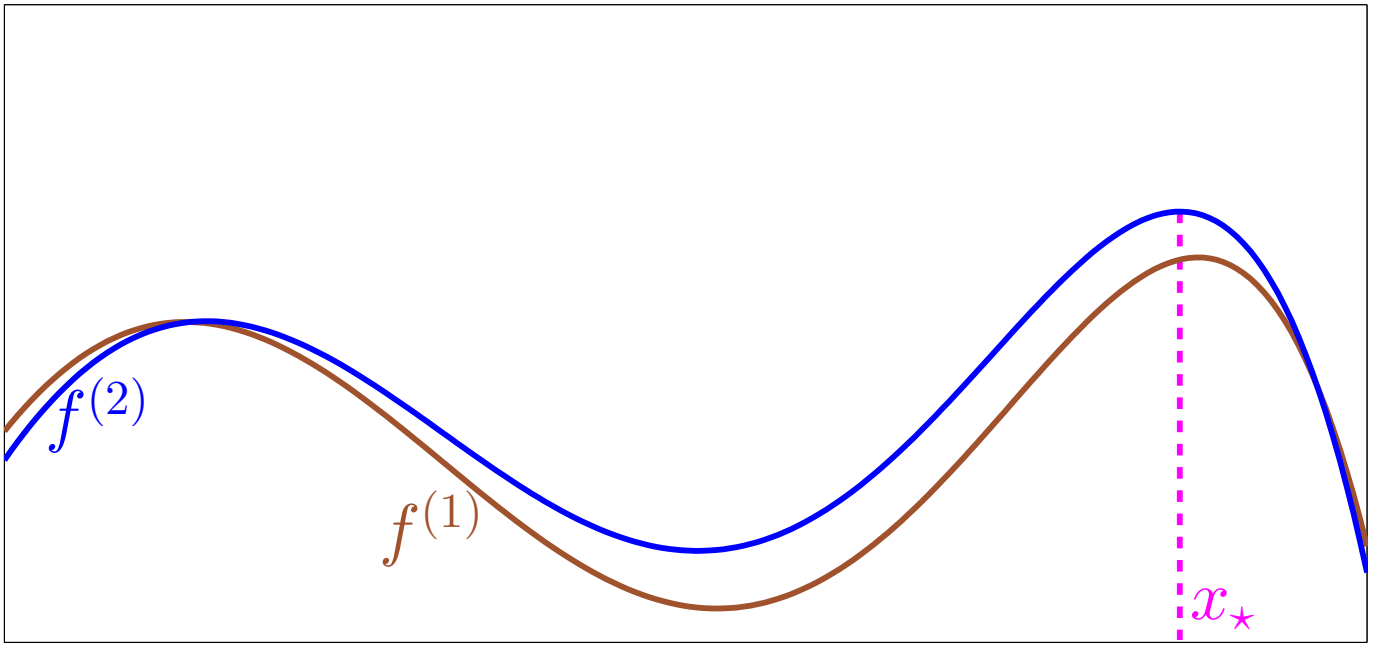
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



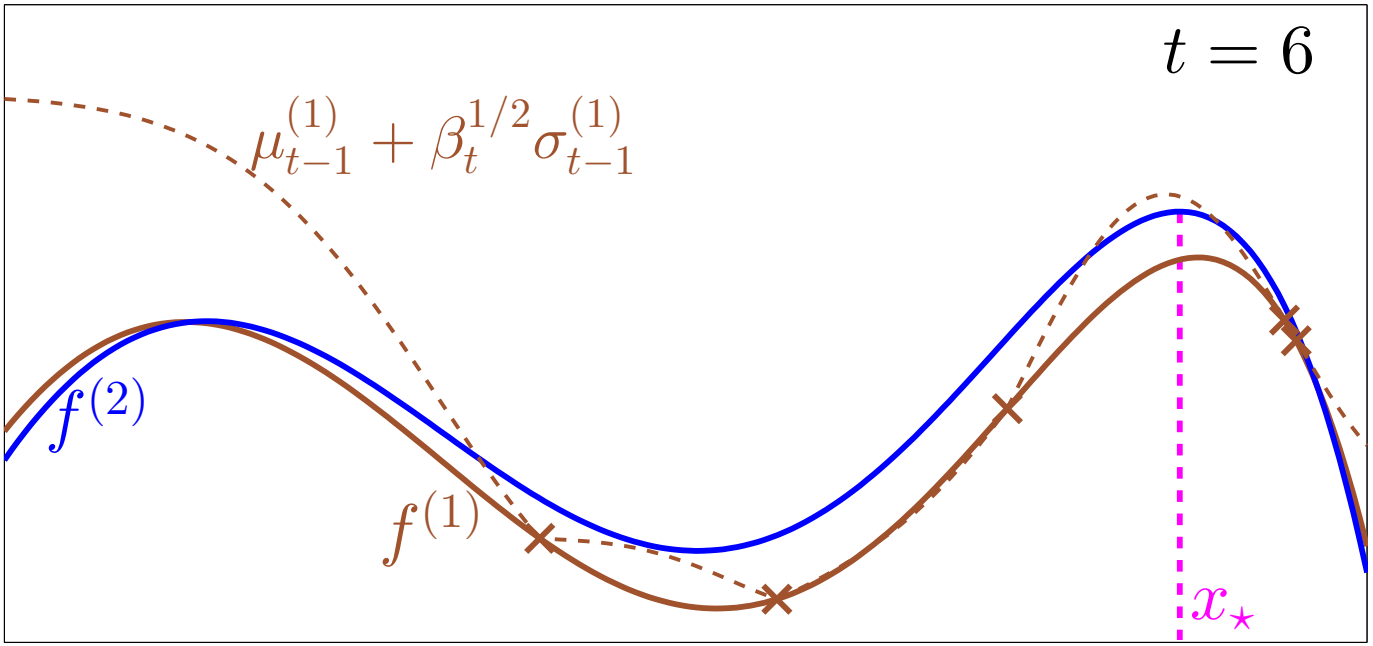
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



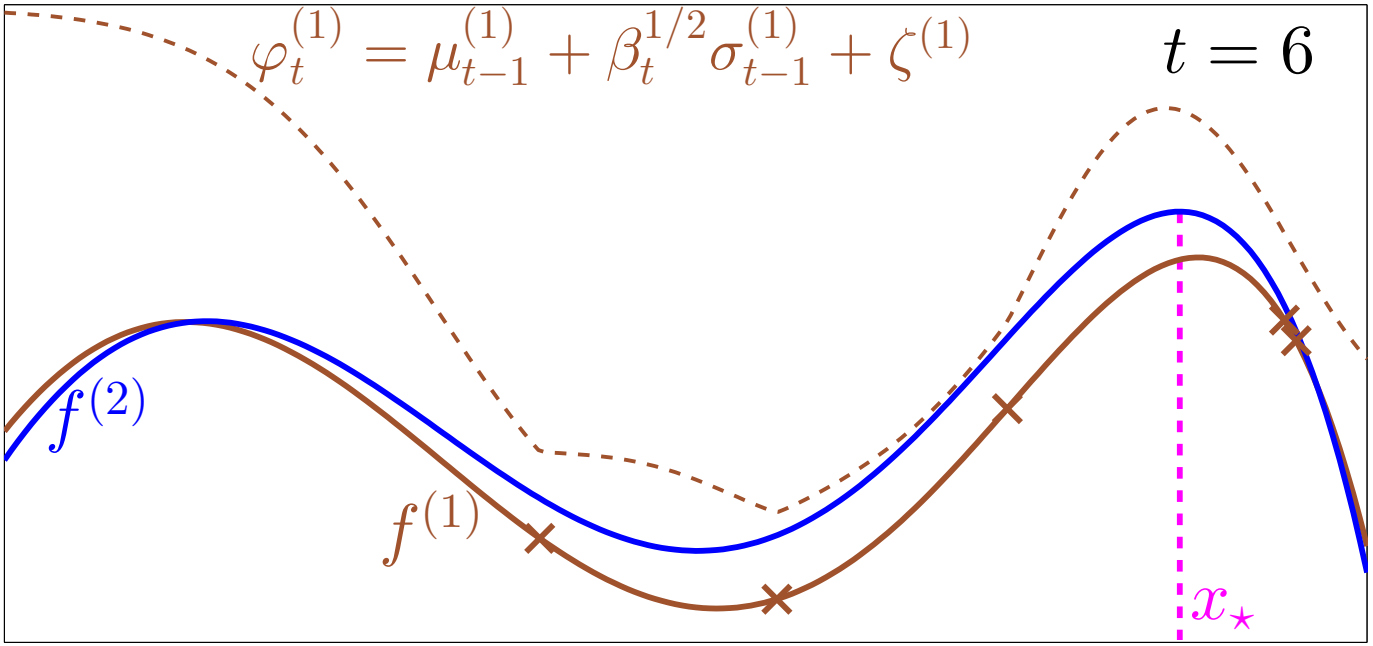
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



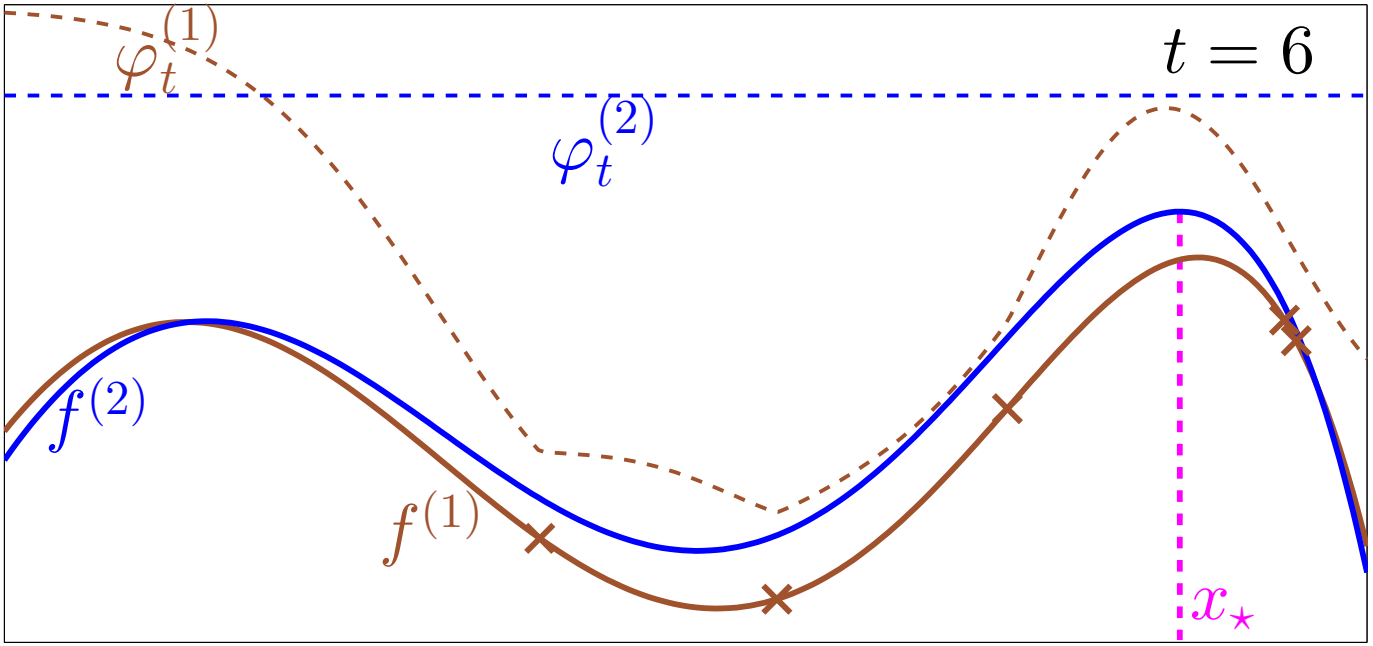
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



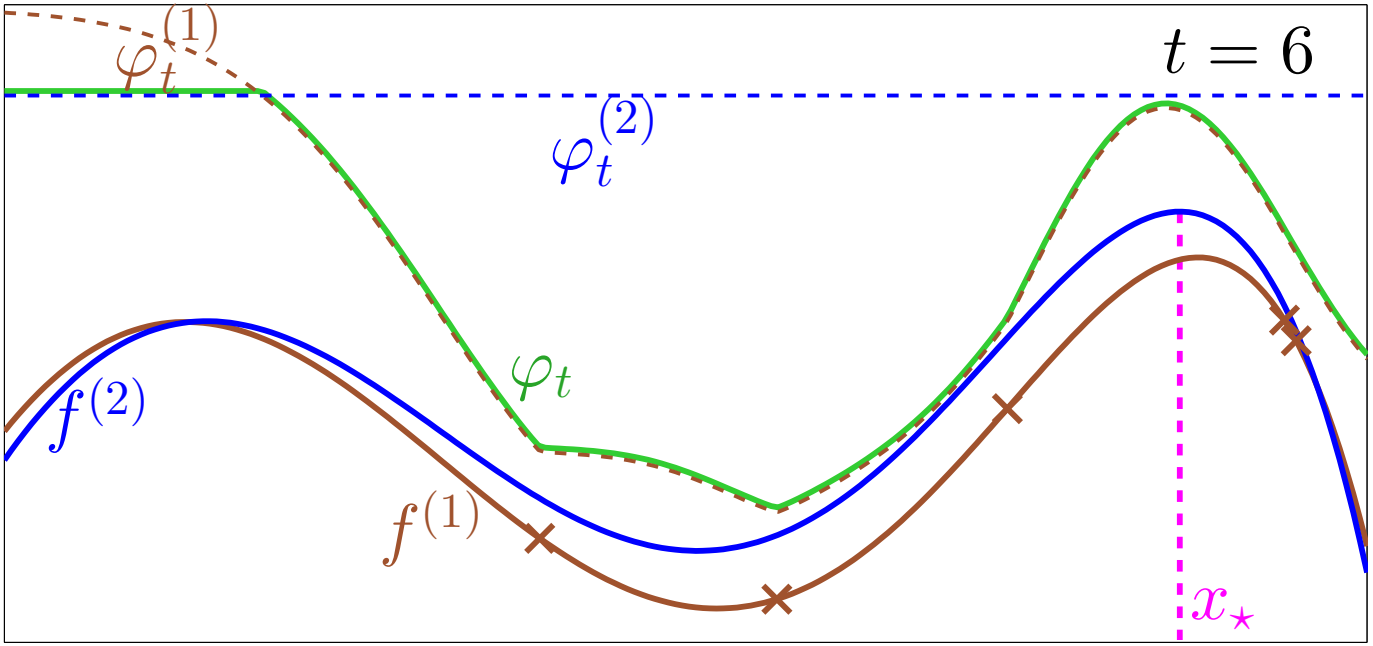
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



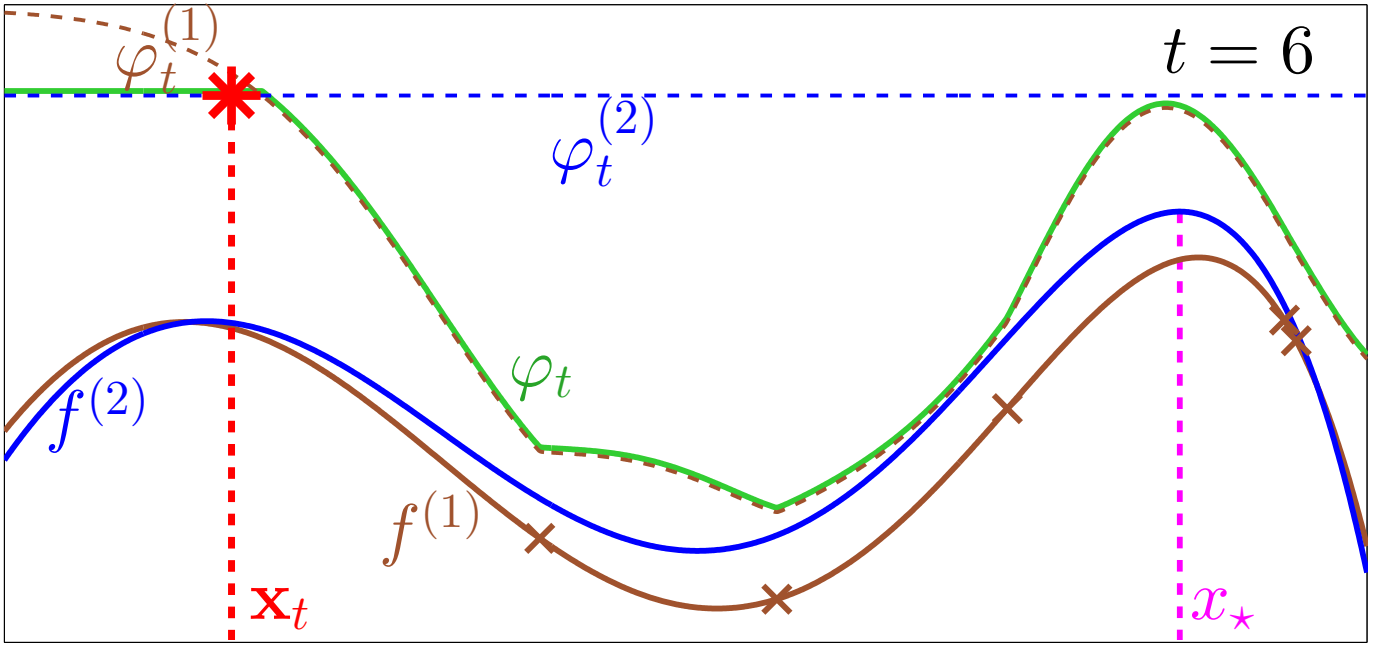
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



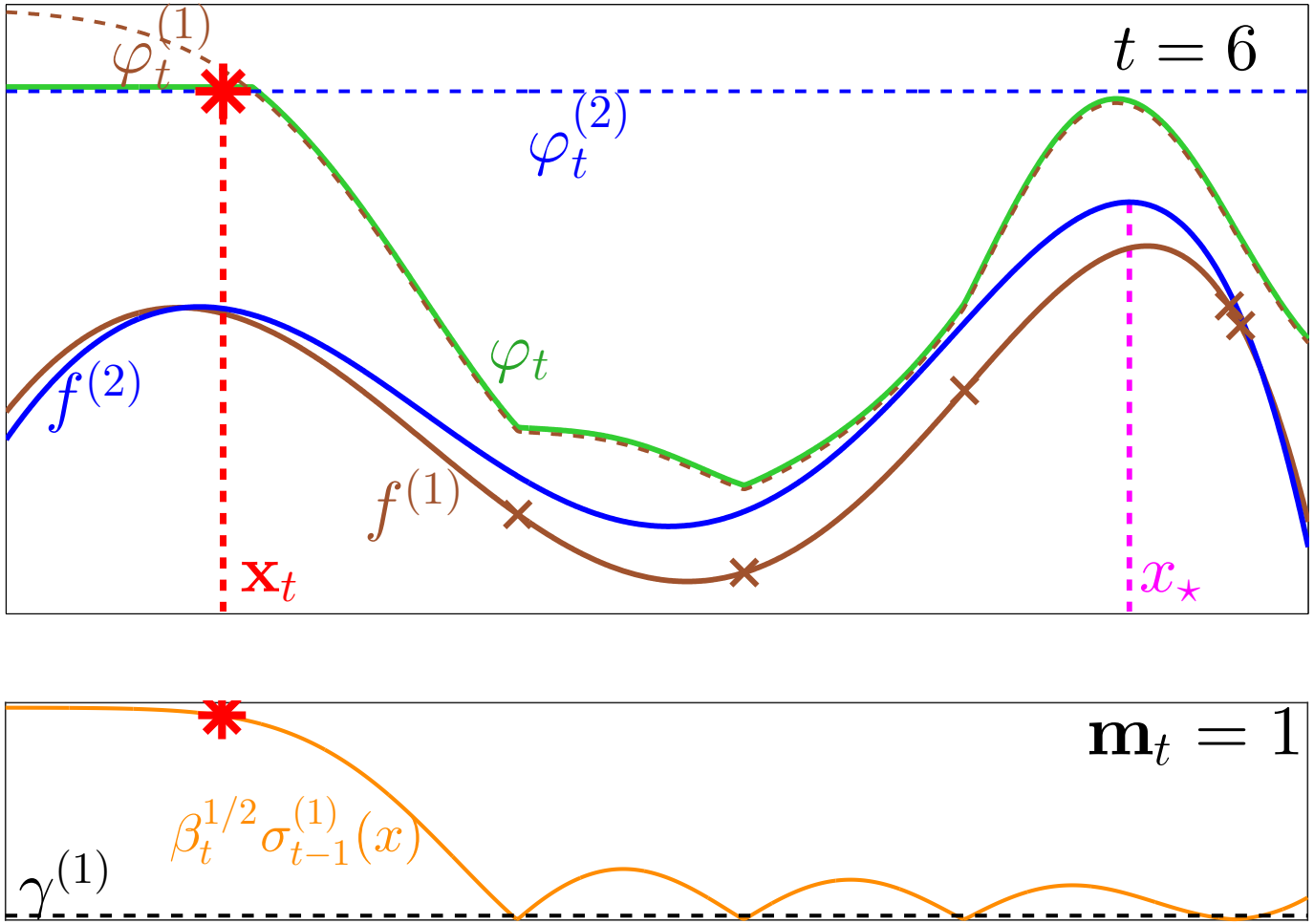
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



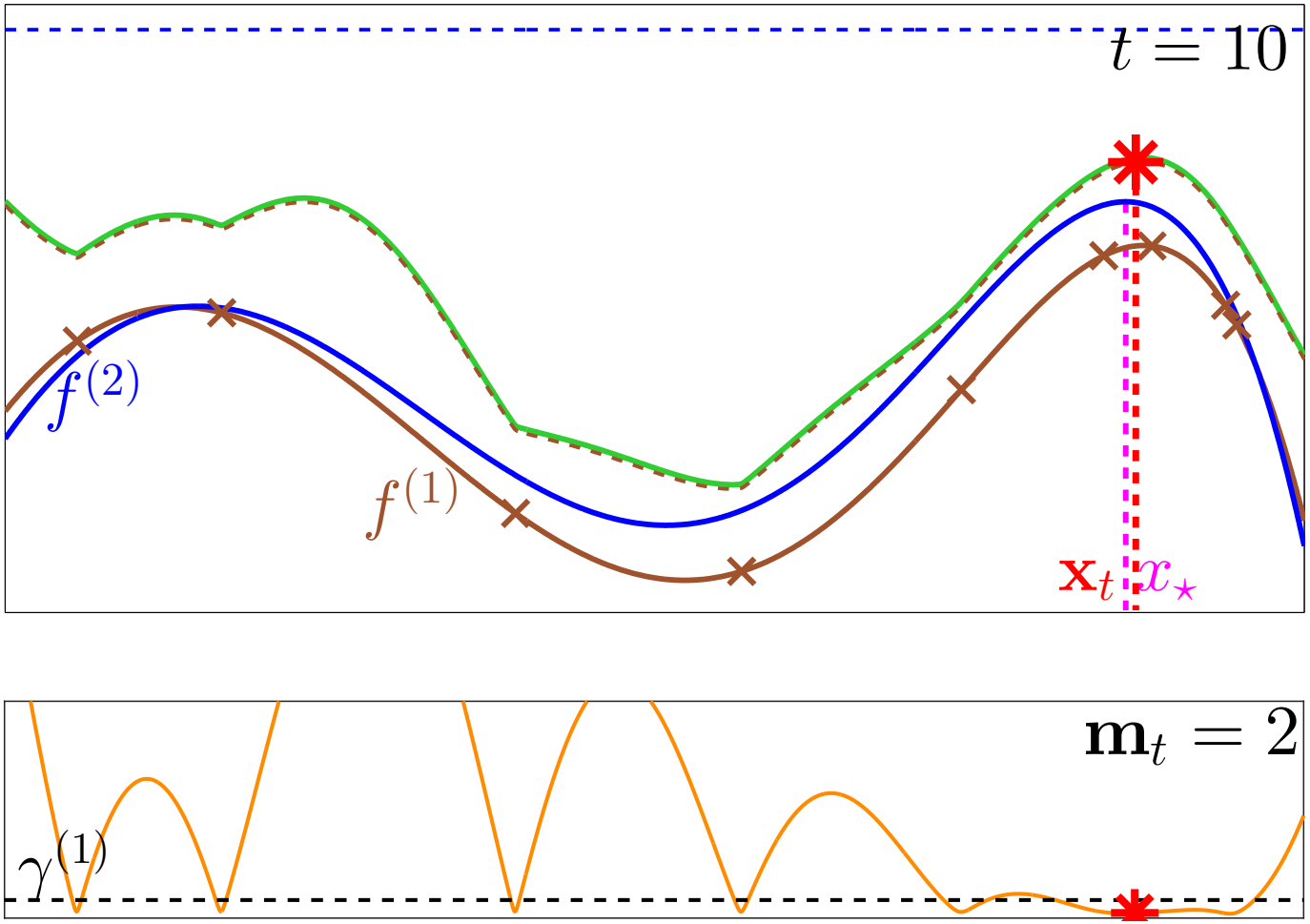
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



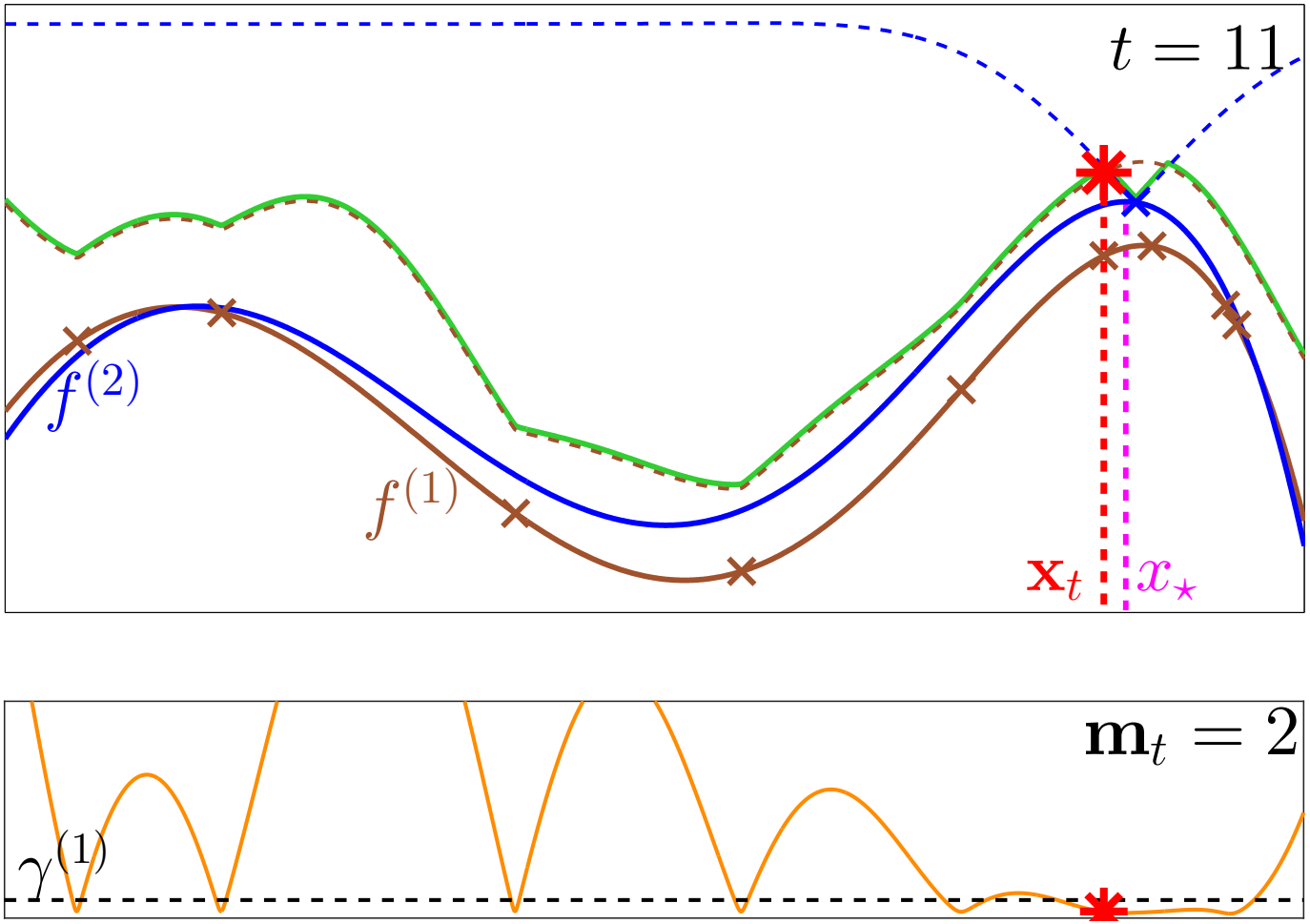
MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



MF-GP-UCB

(Kandasamy et al. NIPS 2016b)



MF-GP-UCB

(Kandasamy et al. NIPS 2016b)