Multi-fidelity Bayesian Optimisation with Continuous Approximations

Kirthevasan Kandasamy

Gautam Dasarathy

Jeff Schneider

Barnabás Póczos

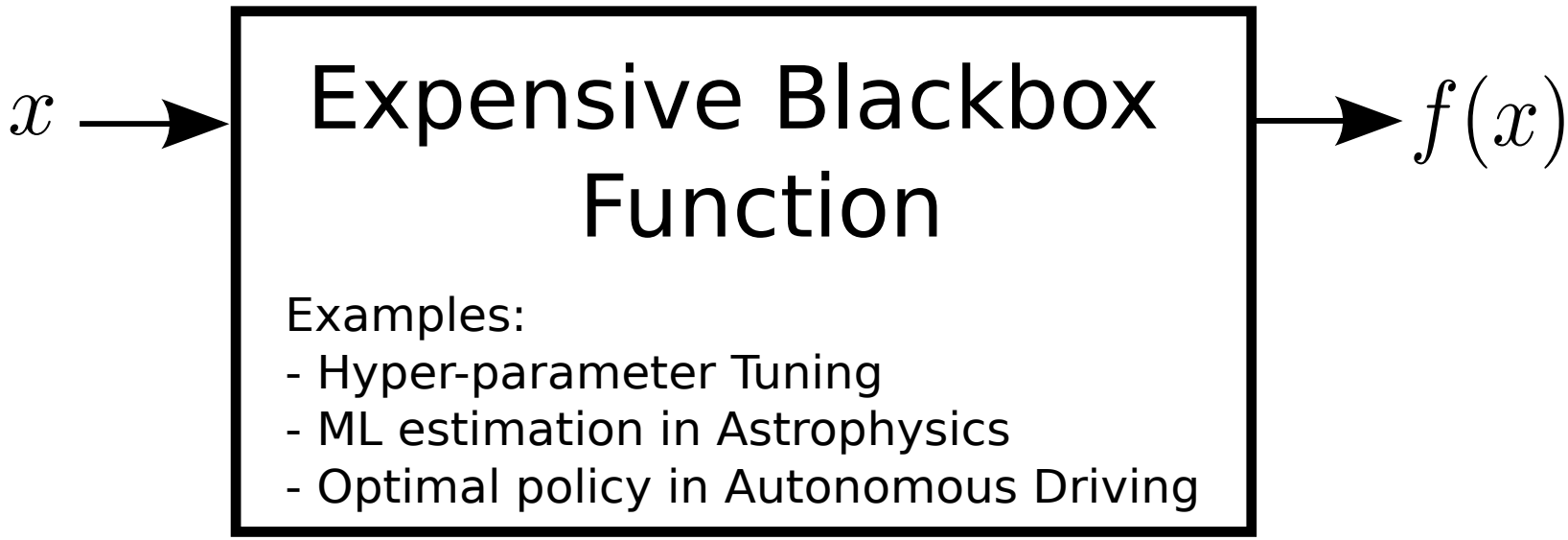
ICML '17

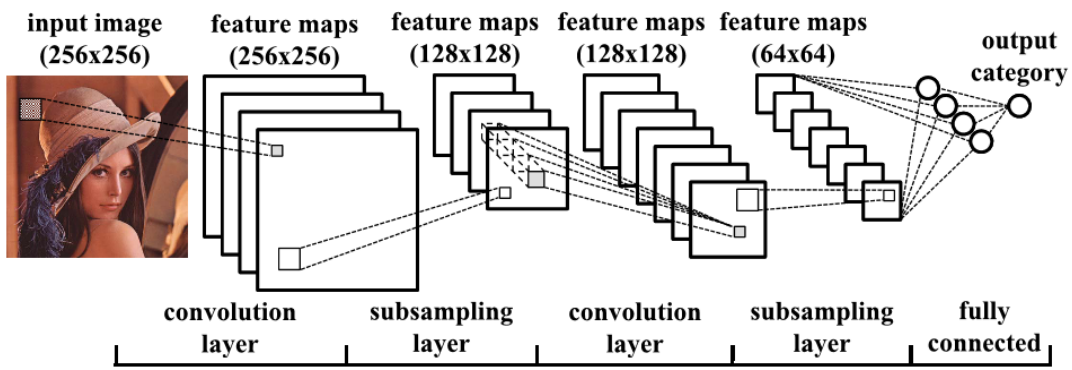
Carnegie Mellon University

RICE

Auton Lab

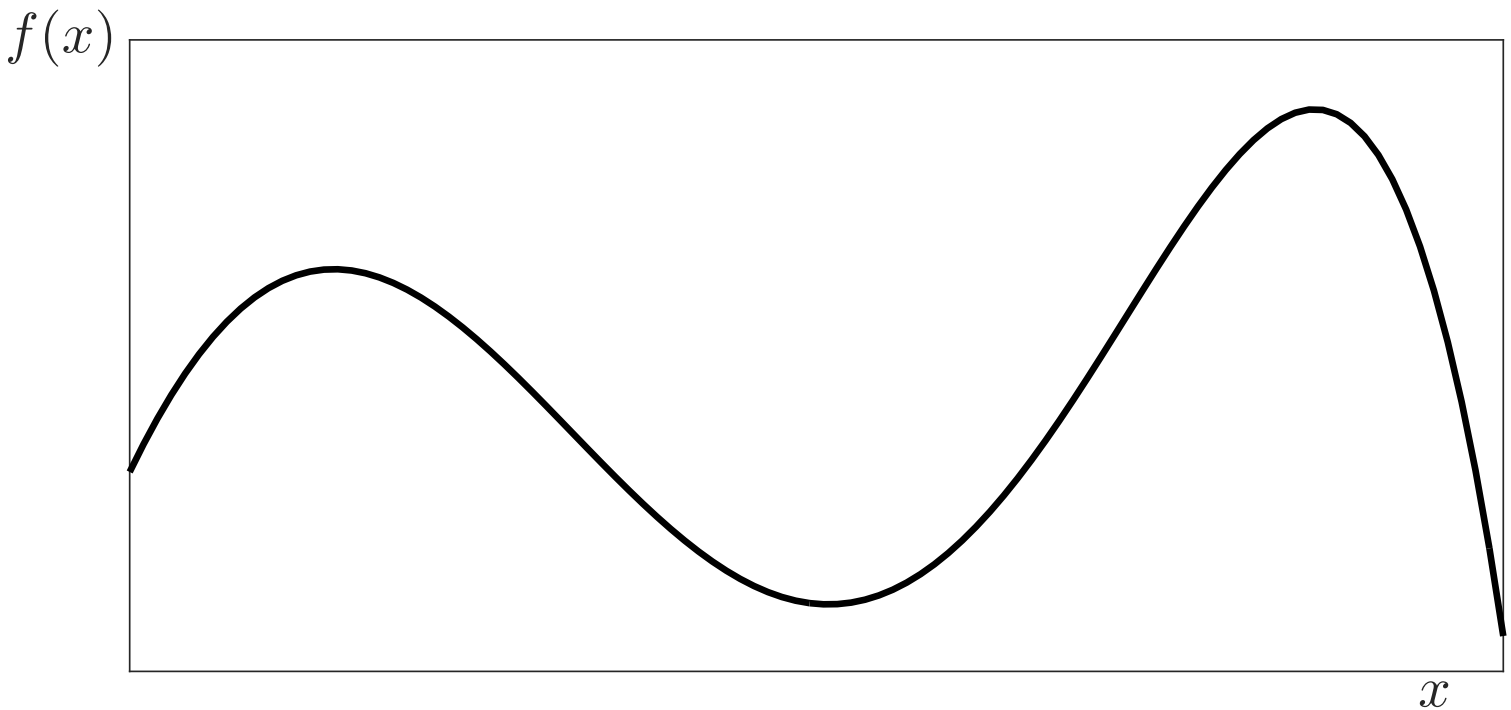
Black-box Optimisation





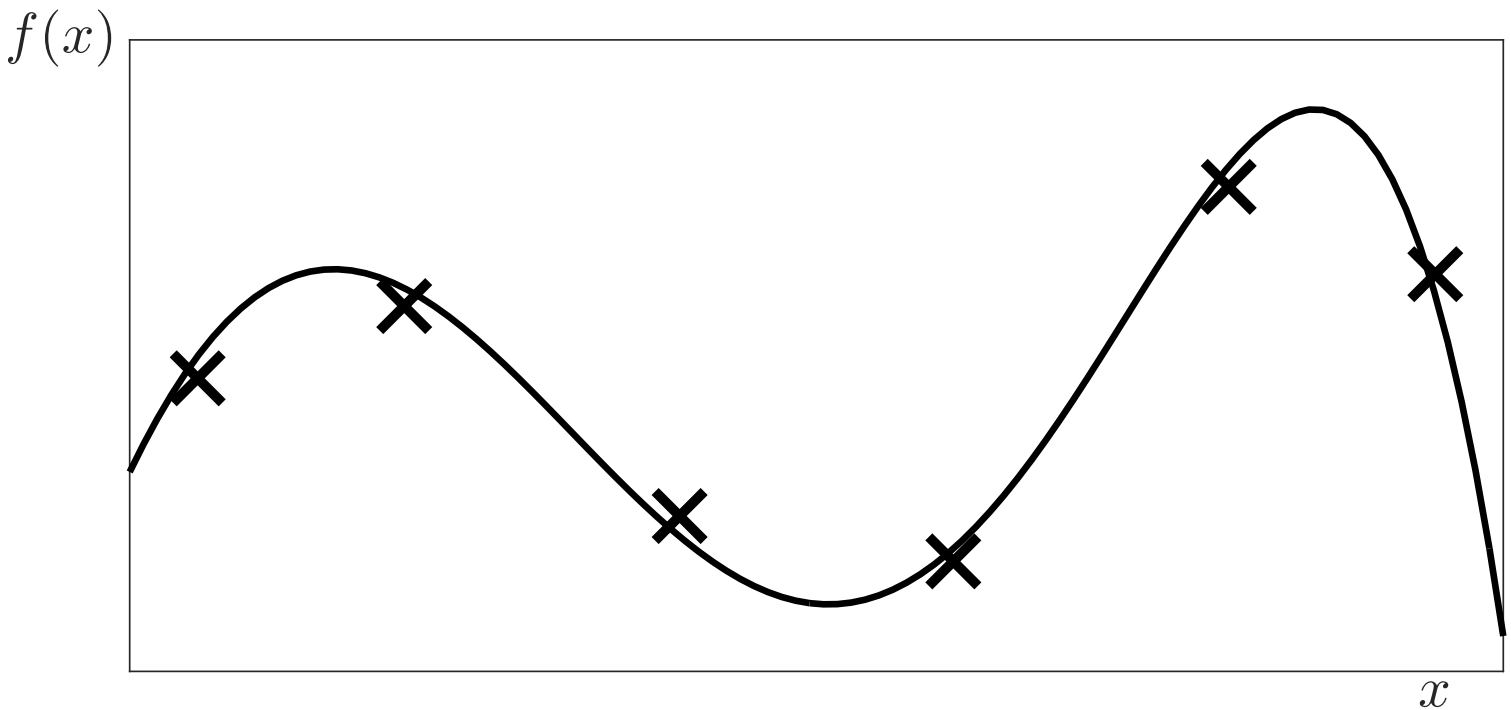
Black-box Optimisation

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



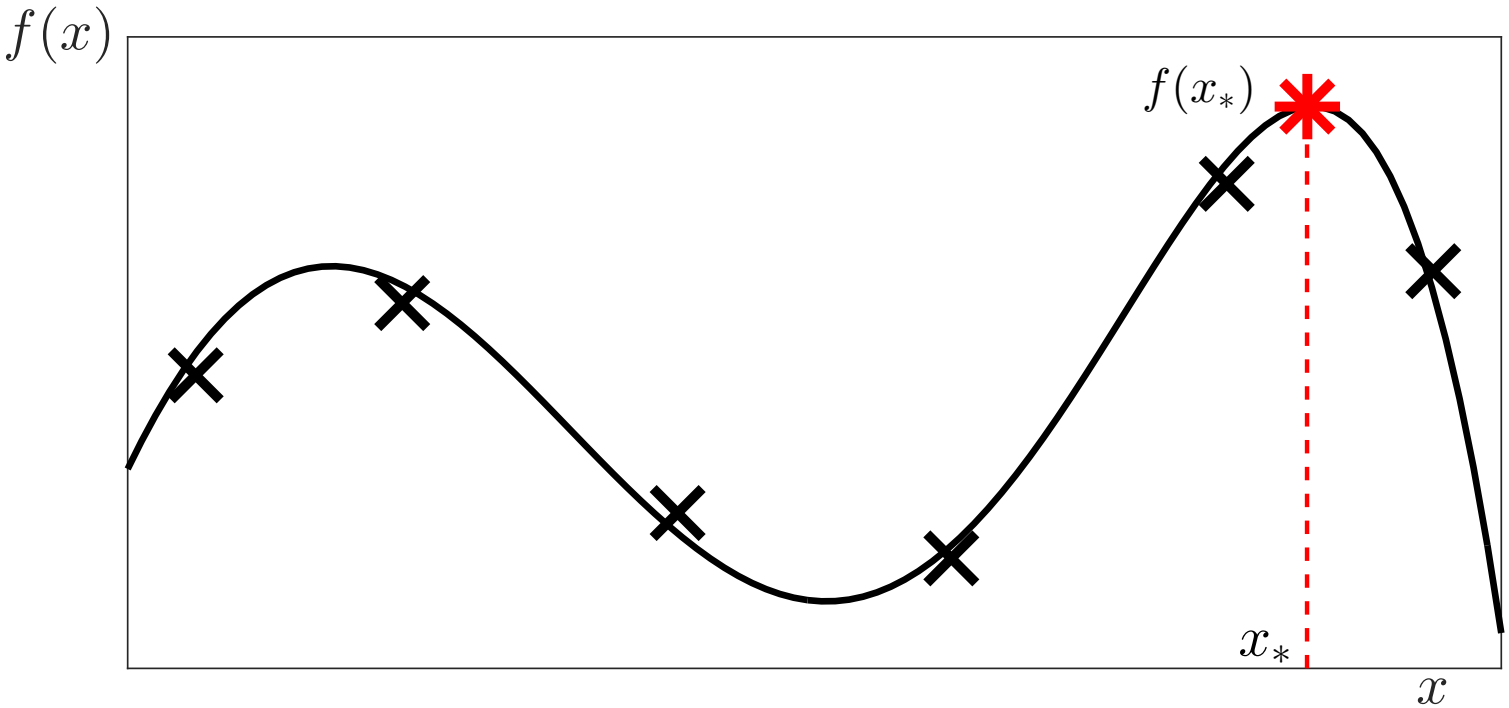
Black-box Optimisation

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



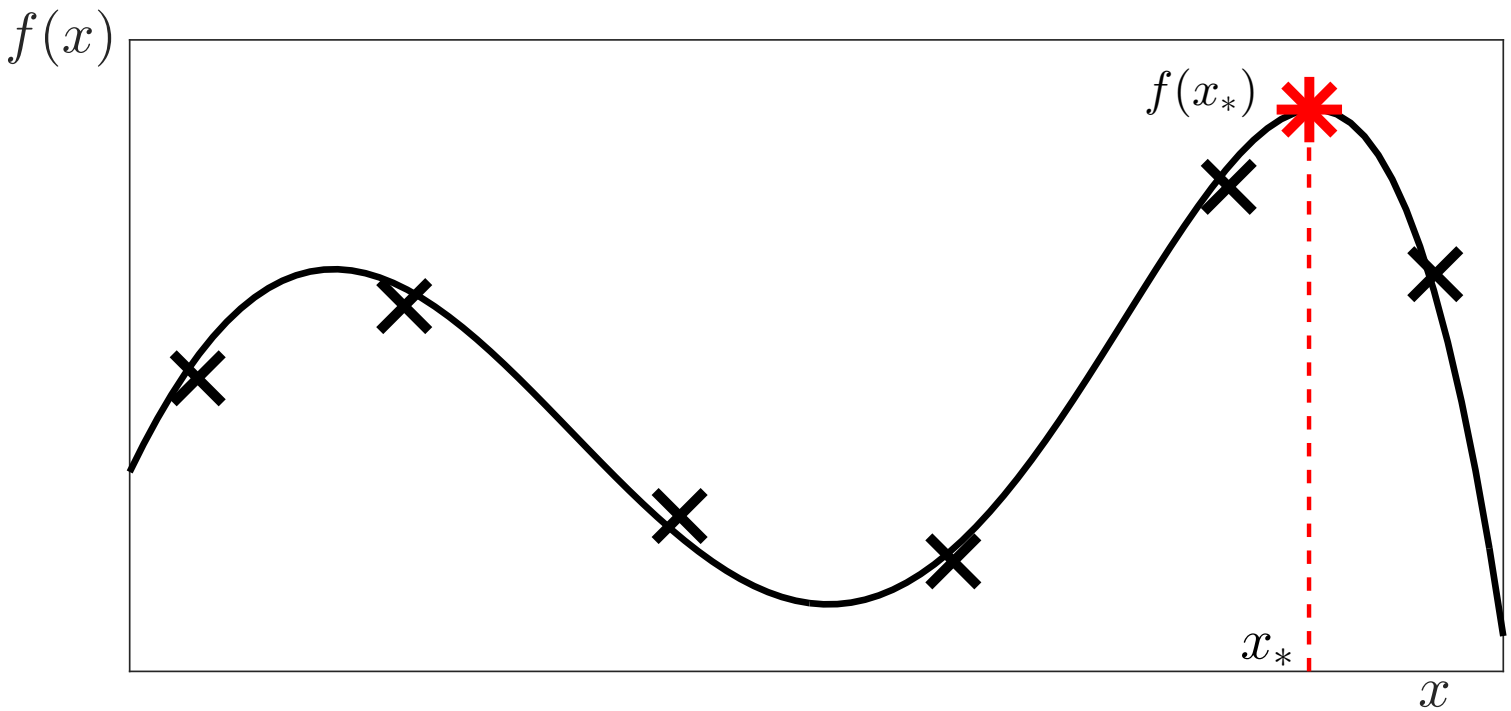
Black-box Optimisation

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



Black-box Optimisation

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



Simple Regret after n evaluations

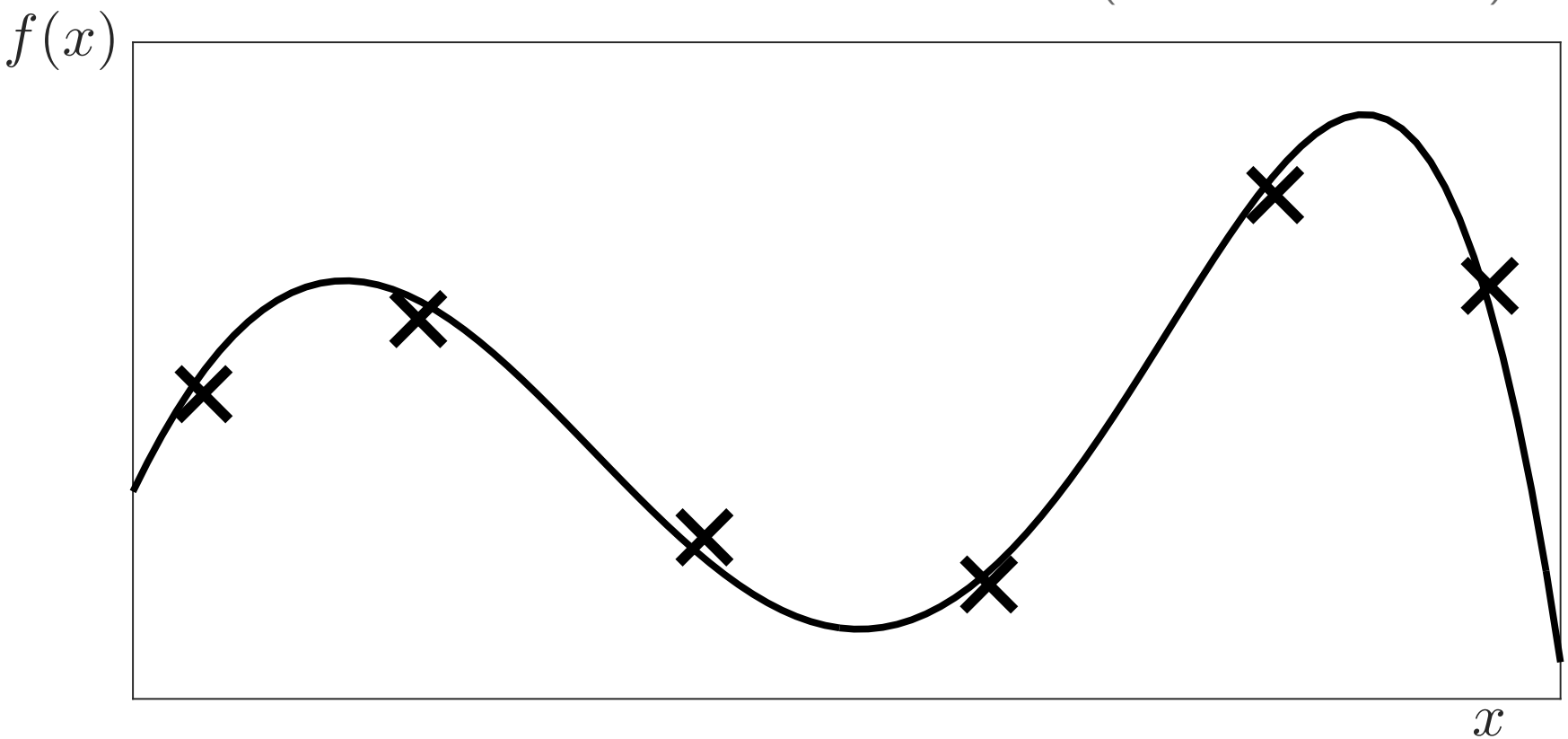
$$S_n = f(x_\star) - \max_{t=1,\ldots,n} f(x_t).$$

Gaussian Process (Bayesian) Optimisation

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

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010).

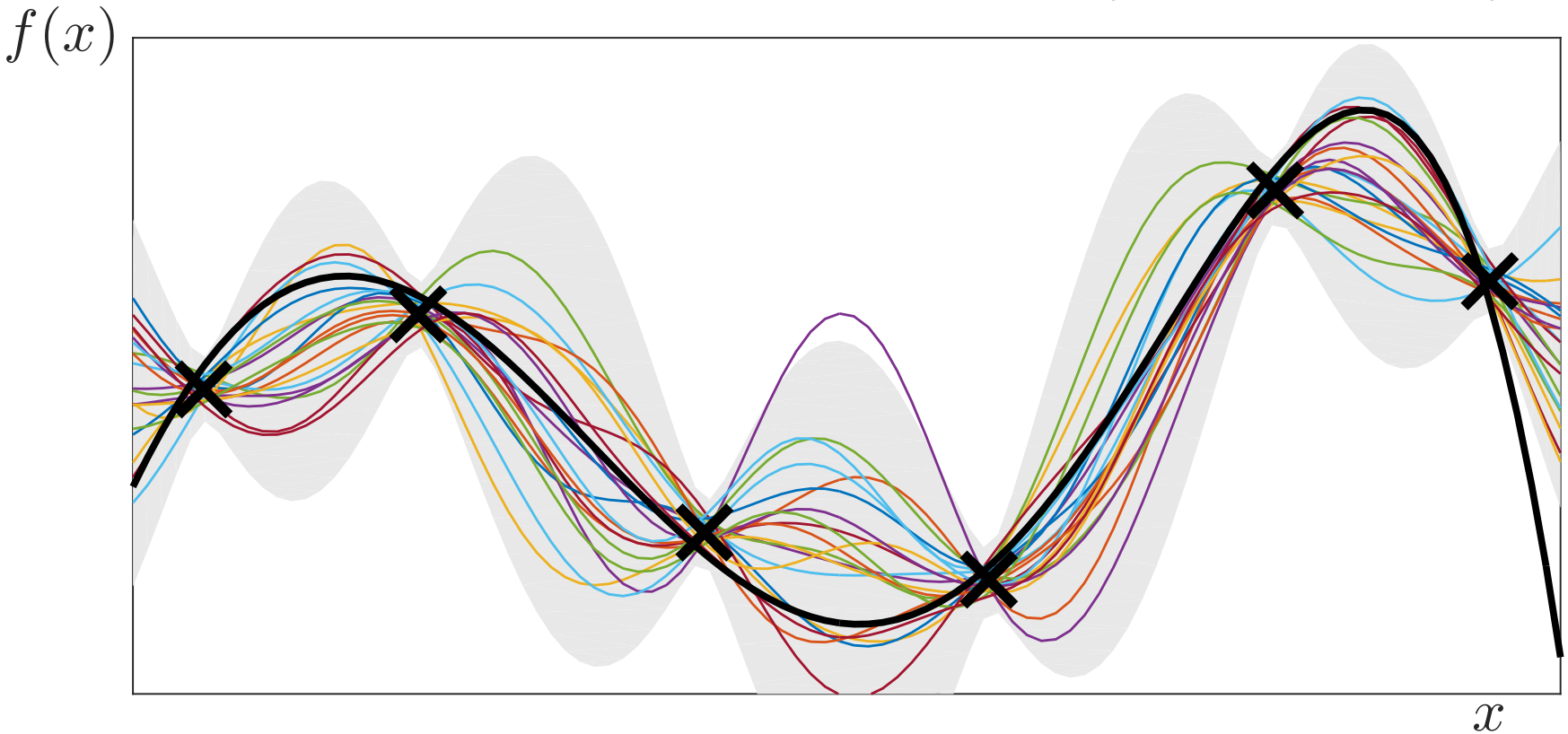


Gaussian Process (Bayesian) Optimisation

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

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010).

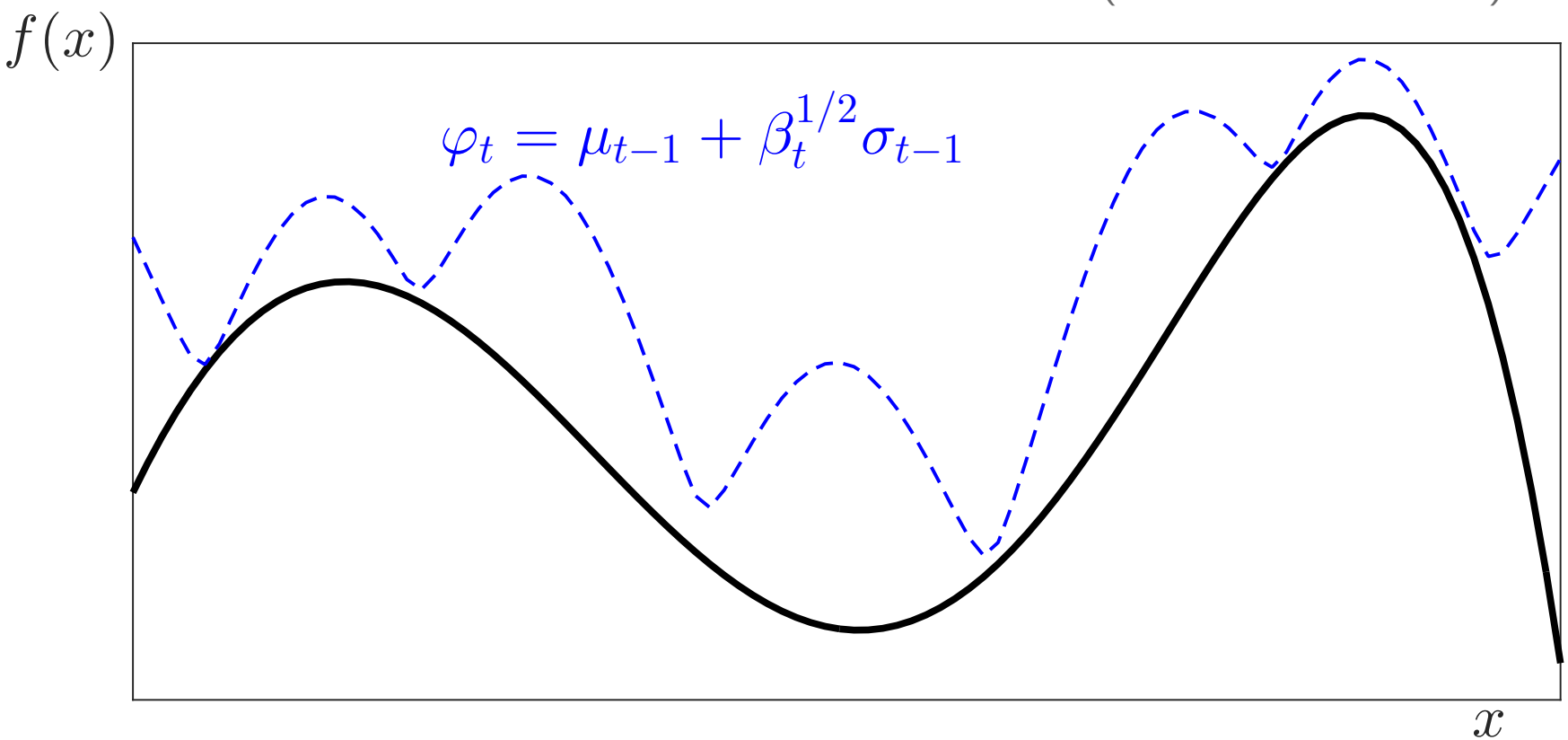


Gaussian Process (Bayesian) Optimisation

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

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010).



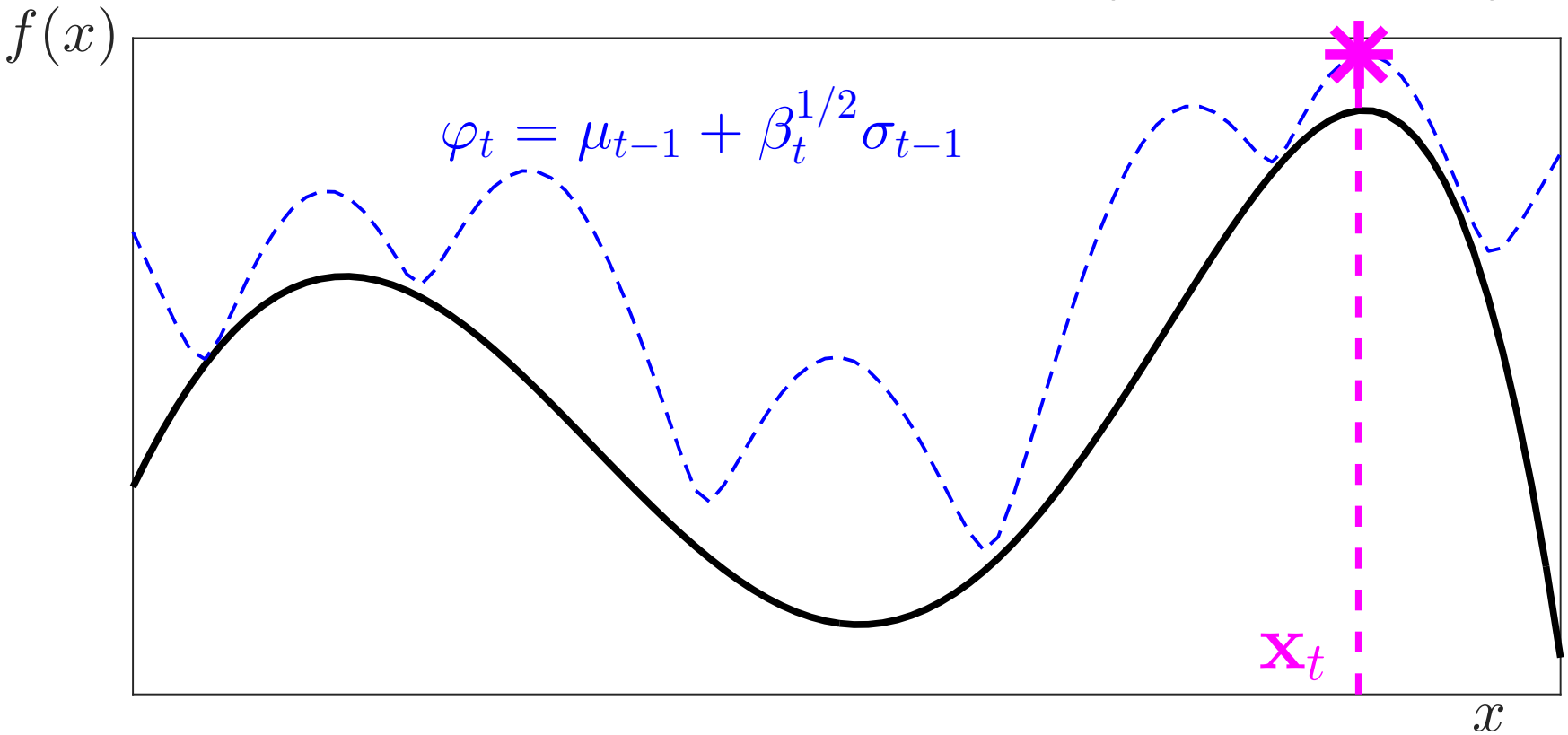
Construct upper conf. bound: $\varphi_t(x) = \mu_{t-1}(x) + \beta_t^{1/2}\sigma_{t-1}(x)$.

Gaussian Process (Bayesian) Optimisation

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

Gaussian Process Upper Confidence Bound (GP-UCB)

(Srinivas et al. 2010).



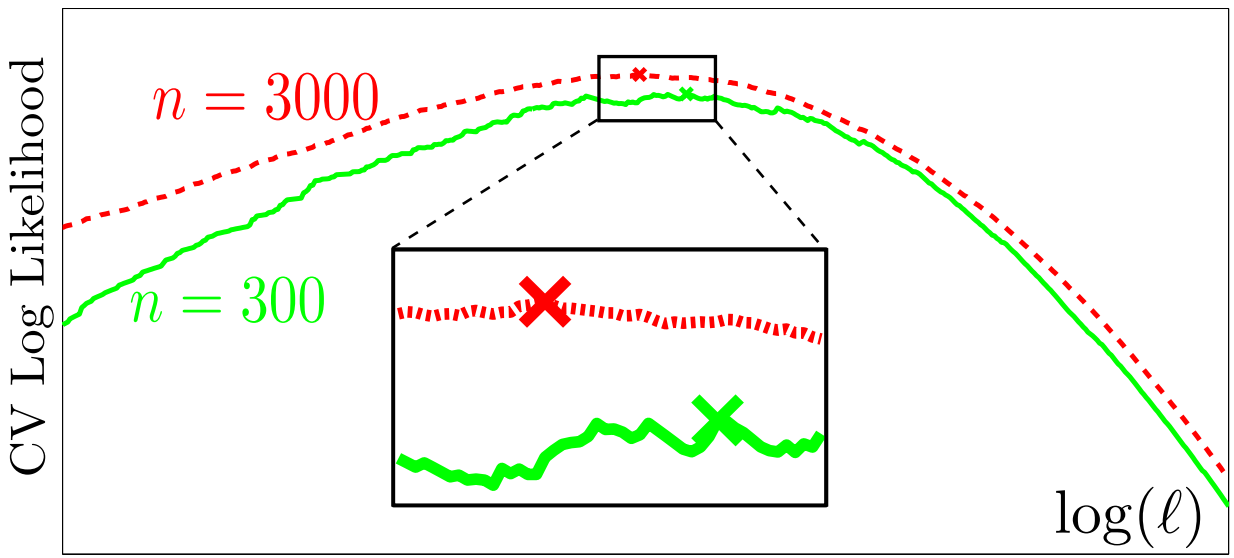
Maximise upper confidence bound.

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

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

E.g. Hyper-parameter tuning: Train & validate with a subset of the data.

Bandwidth (ℓ) selection in kernel density estimation.



Prior work in **Multi-fidelity** Methods

For specific applications, such as hyper-parameter tuning, active learning, robotics etc. (Agarwal et al. 2011, Cutler et al. 2014, Zhang & Chaudhuri 2015, Klein et al. 2015, Li et al. 2016)

Prior work in **Multi-fidelity** Methods

For specific applications, such as hyper-parameter tuning, active learning, robotics etc. (Agarwal et al. 2011, Cutler et al. 2014, Zhang & Chaudhuri 2015, Klein et al. 2015, Li et al. 2016)

Multi-fidelity **optimisation** (Huang et al. 2006, Forrester et al. 2007, March & Wilcox 2012, Poloczek et al. 2016)

Prior work in **Multi-fidelity** Methods

For specific applications, such as hyper-parameter tuning, active learning, robotics etc. (Agarwal et al. 2011, Cutler et al. 2014, Zhang & Chaudhuri 2015, Klein et al. 2015, Li et al. 2016)

Multi-fidelity **optimisation** (Huang et al. 2006, Forrester et al. 2007, March & Wilcox 2012, Poloczek et al. 2016)

.. with theoretical guarantees (Kandasamy et al. 2016a, 2016b)

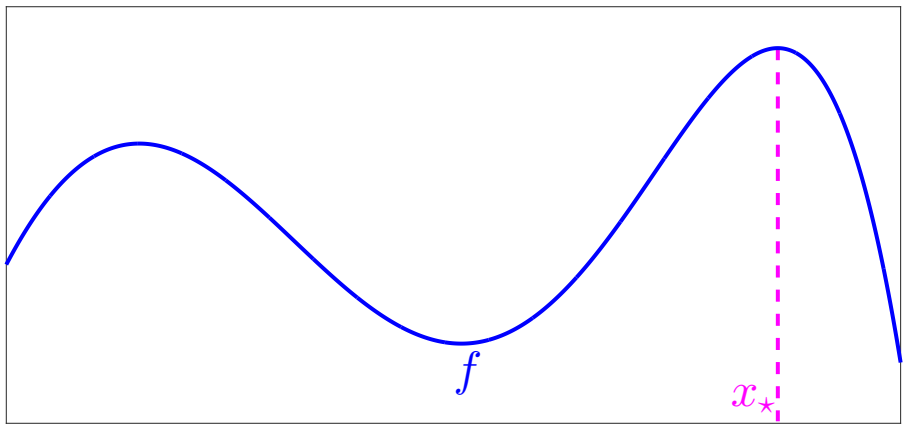
Prior work in **Multi-fidelity** Methods

For specific applications, such as hyper-parameter tuning, active learning, robotics etc. (Agarwal et al. 2011, Cutler et al. 2014, Zhang & Chaudhuri 2015, Klein et al. 2015, Li et al. 2016)

Multi-fidelity **optimisation** (Huang et al. 2006, Forrester et al. 2007, March & Wilcox 2012, Poloczek et al. 2016)

.. with theoretical guarantees (Kandasamy et al. 2016a, 2016b)

Prior work only focus on a finite number of approximations,



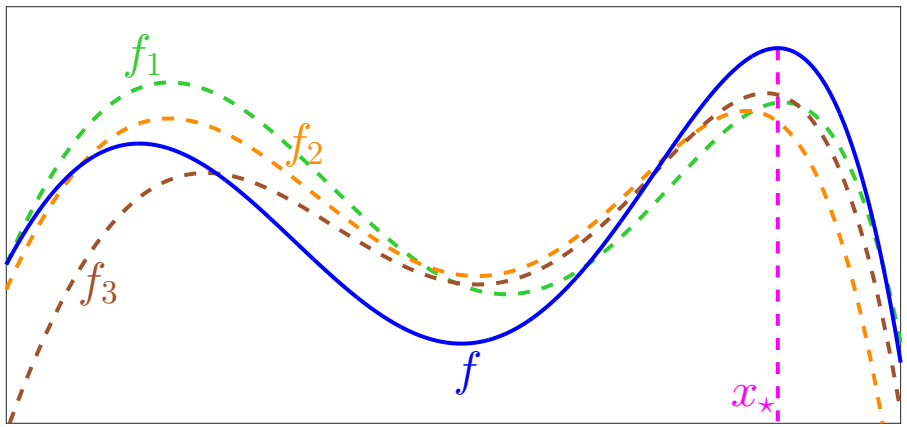
Prior work in **Multi-fidelity** Methods

For specific applications, such as hyper-parameter tuning, active learning, robotics etc. (Agarwal et al. 2011, Cutler et al. 2014, Zhang & Chaudhuri 2015, Klein et al. 2015, Li et al. 2016)

Multi-fidelity **optimisation** (Huang et al. 2006, Forrester et al. 2007, March & Wilcox 2012, Poloczek et al. 2016)

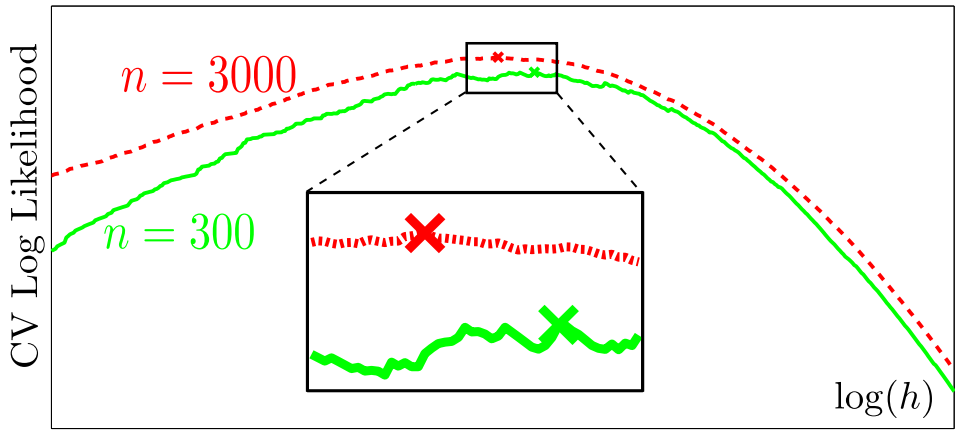
.. with theoretical guarantees (Kandasamy et al. 2016a, 2016b)

Prior work only focus on a finite number of approximations,



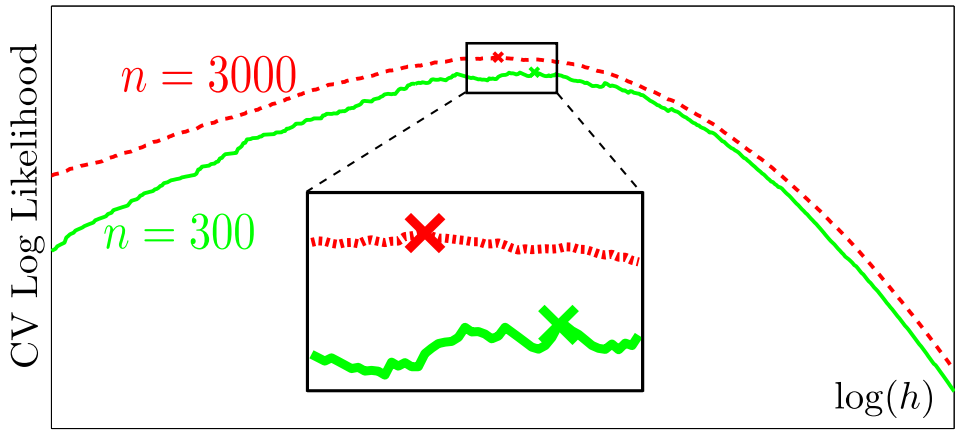
$f_1, f_2, f_3 \approx f$ which are cheaper to evaluate.

Why continuous approximations?



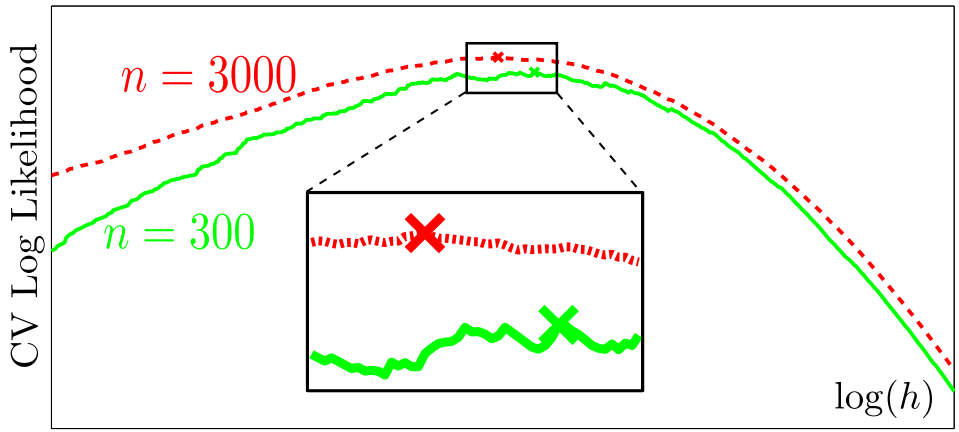
- Use an arbitrary amount of data?

Why continuous approximations?



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

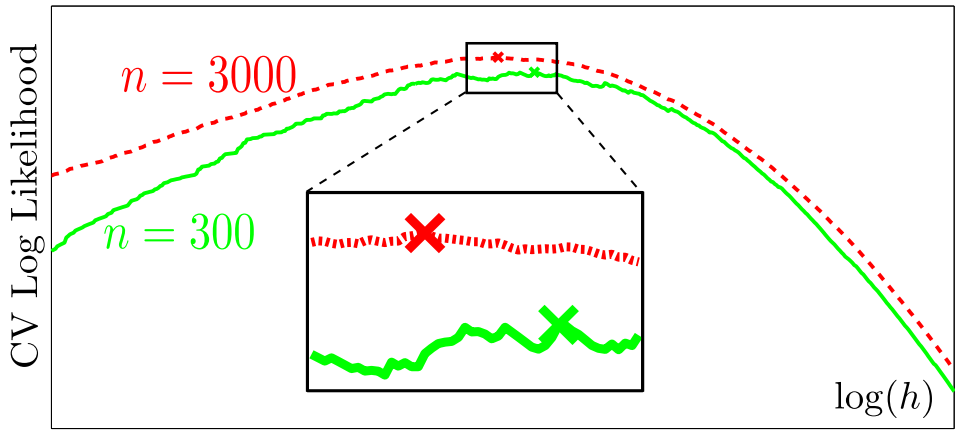
Why continuous approximations?



- 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.

Why continuous approximations?

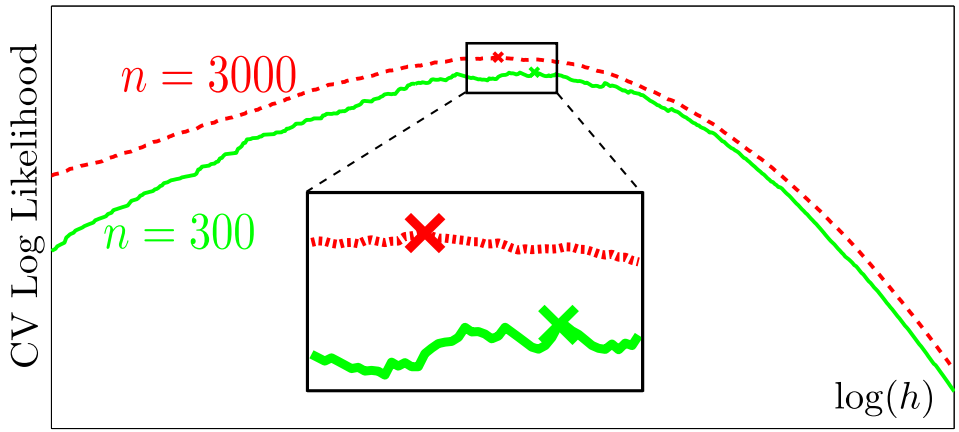


- Use an arbitrary amount of data?
- Iterative algorithms: use arbitrary number of iterations?

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?

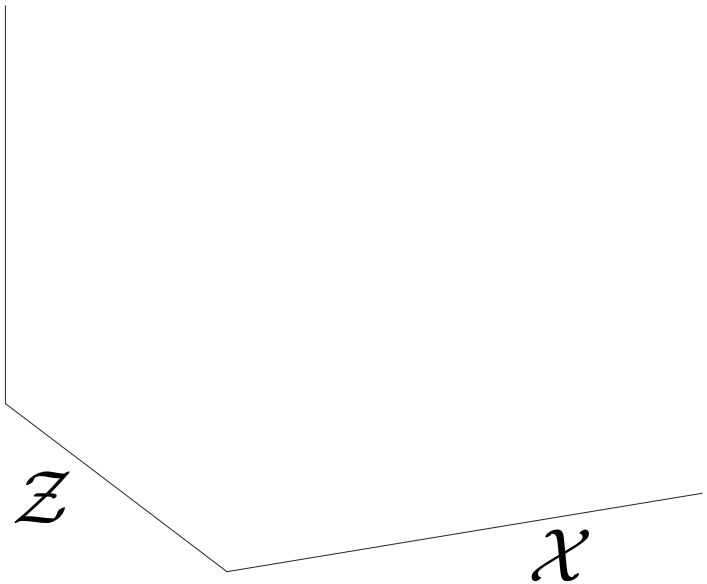
E.g. Train an ML model with $N_\bullet$ data and $T_\bullet$ iterations.

- 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).

Multi-fidelity Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



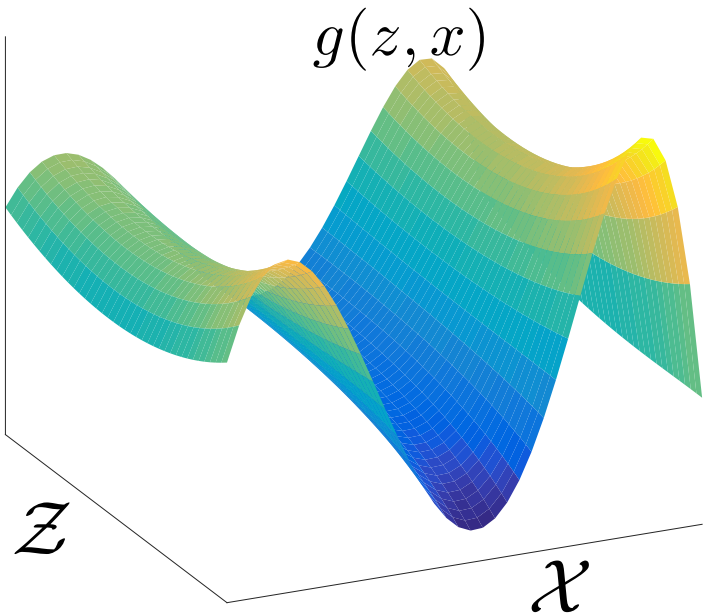
A fidelity space $\mathcal{Z}$ and domain $\mathcal{X}$

$\mathcal{Z} \leftarrow$ 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.

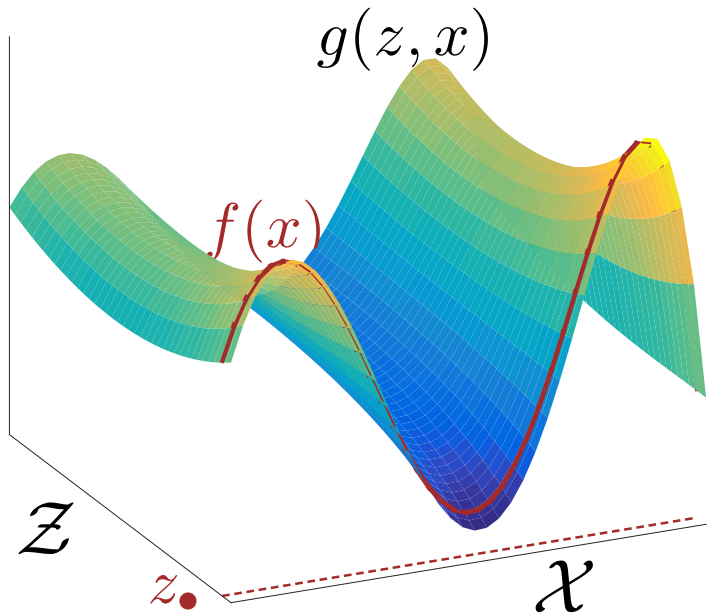
$\mathcal{X} \leftarrow$ all hyper-parameter values.

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

$g([N, T], x) \leftarrow$ 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.

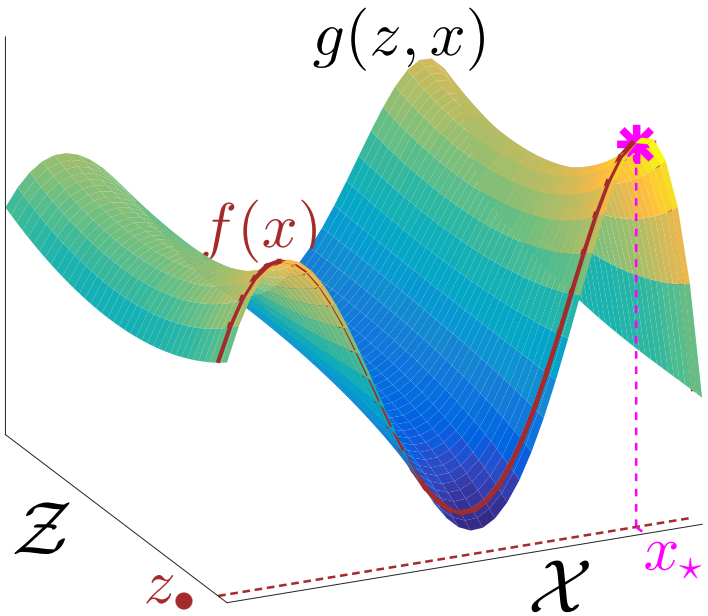
$g : \mathcal{Z} \times \mathcal{X} \rightarrow \mathbb{R}$.

$g([N, T], x) \leftarrow$ 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.

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

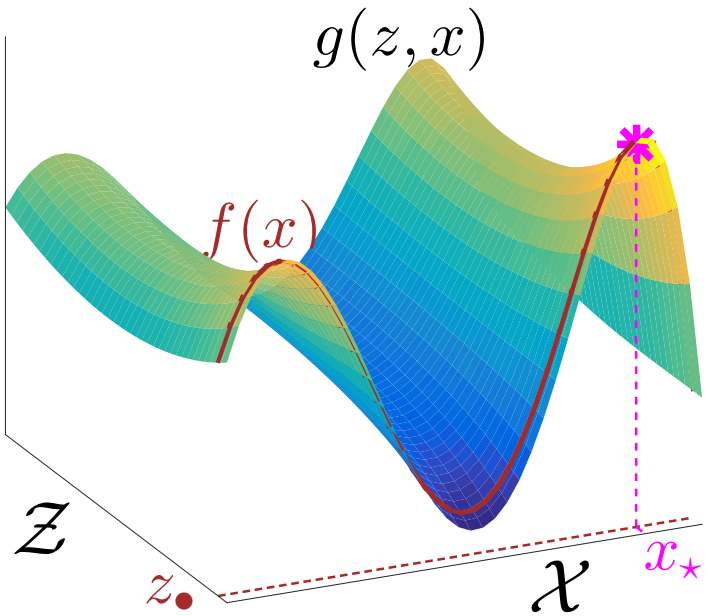
$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

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 $x_\star = \text{argmax}_x f(x)$.

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.

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

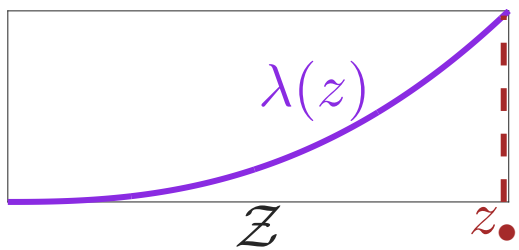
$g([N, T], x) \leftarrow$ cv accuracy when training with N data for T iterations at hyper-parameter x.

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

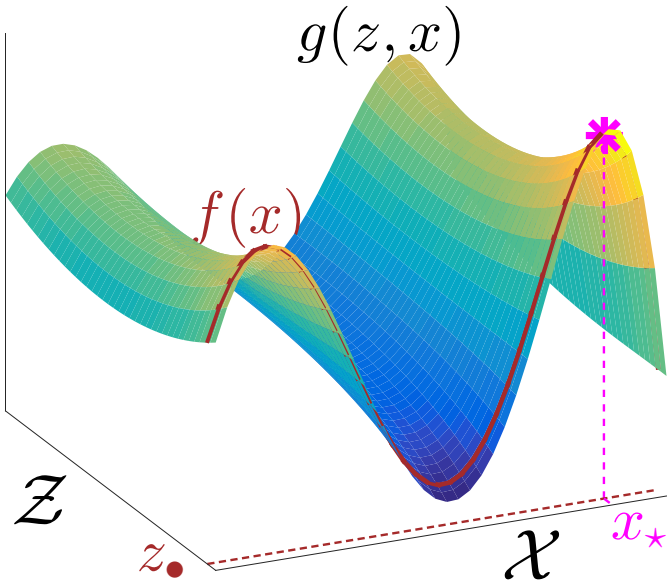
End Goal: Find $x_\star = \text{argmax}_x f(x)$.

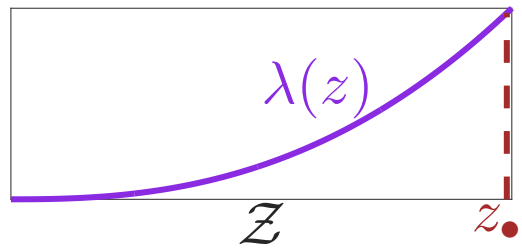
A cost function, $\lambda : \mathcal{Z} \rightarrow \mathbb{R}_+$.

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



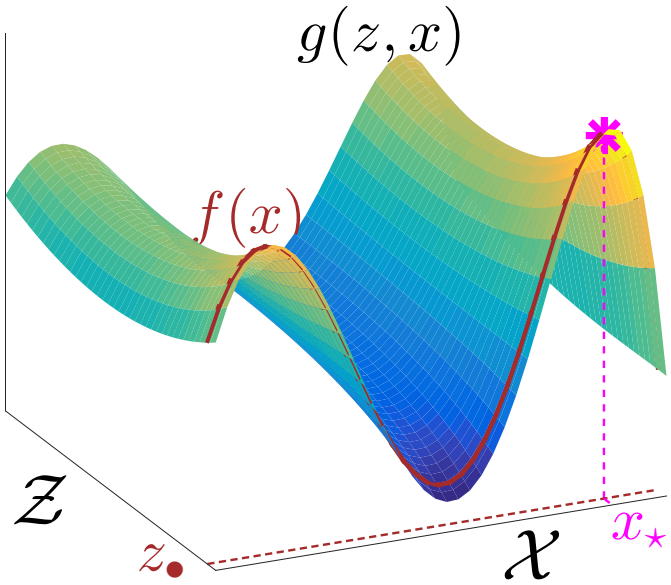
Multi-fidelity Simple Regret (Kandasamy et al. ICML 2017)

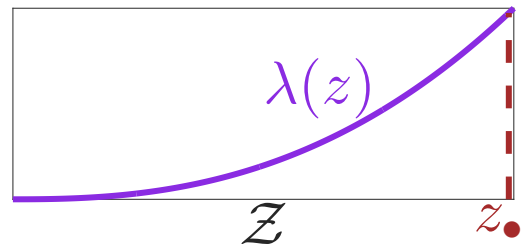




End Goal: Find $x_\star = \text{argmax}_x f(x)$.

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



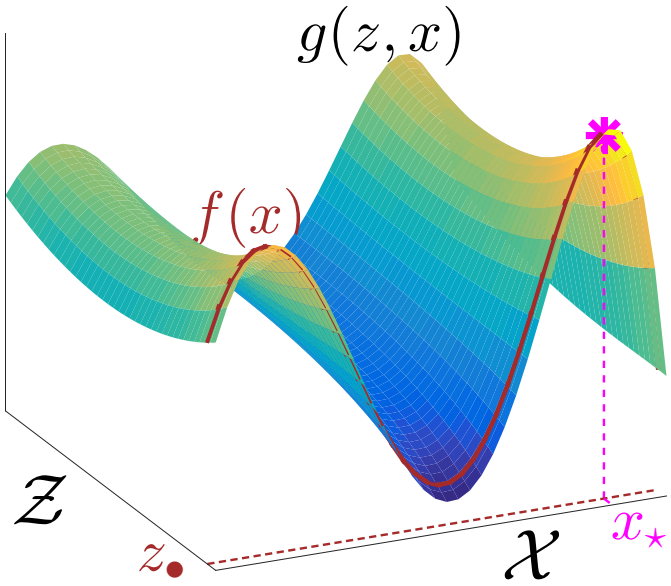


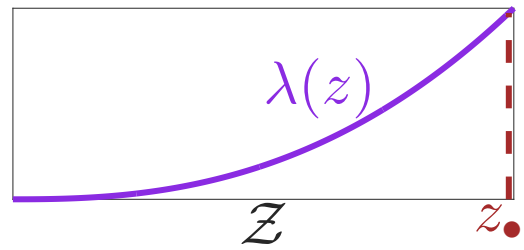
End Goal: Find $x_\star = \text{argmax}_x f(x)$.

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 $x_\star = \text{argmax}_x f(x)$.

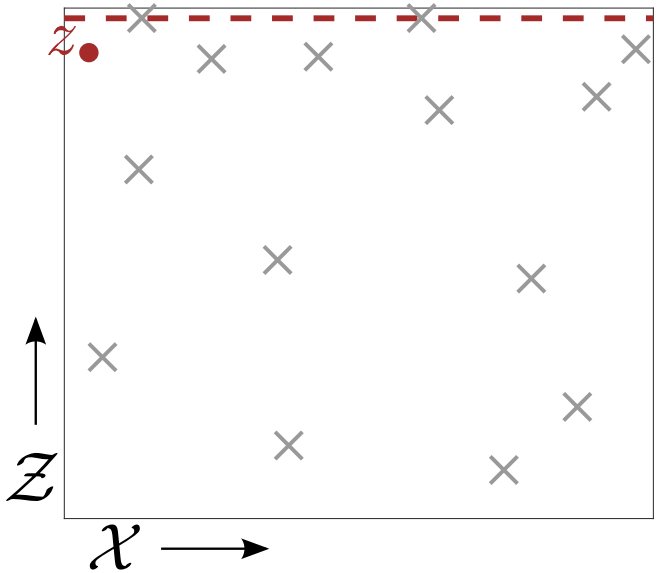
Simple Regret after *capital* Λ: $S(\Lambda) = f(x_\star) - \max_{t:\, z_t = z_\bullet} f(x_t)$.

$\Lambda \leftarrow$ 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 $x_\star$.

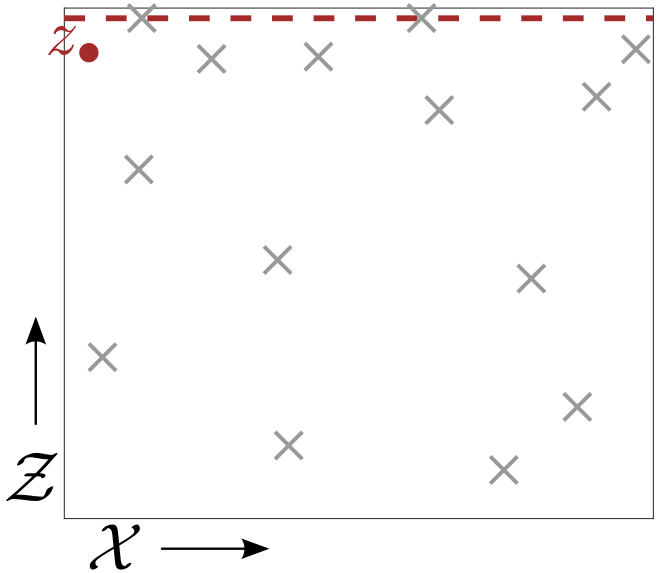
BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



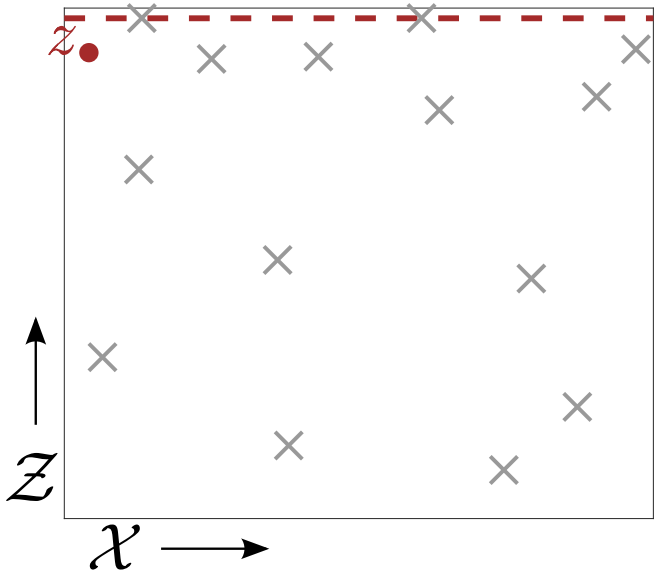
Model $g \sim \mathcal{GP}(0, \kappa)$ and compute posterior $\mathcal{GP}$:

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

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

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



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

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

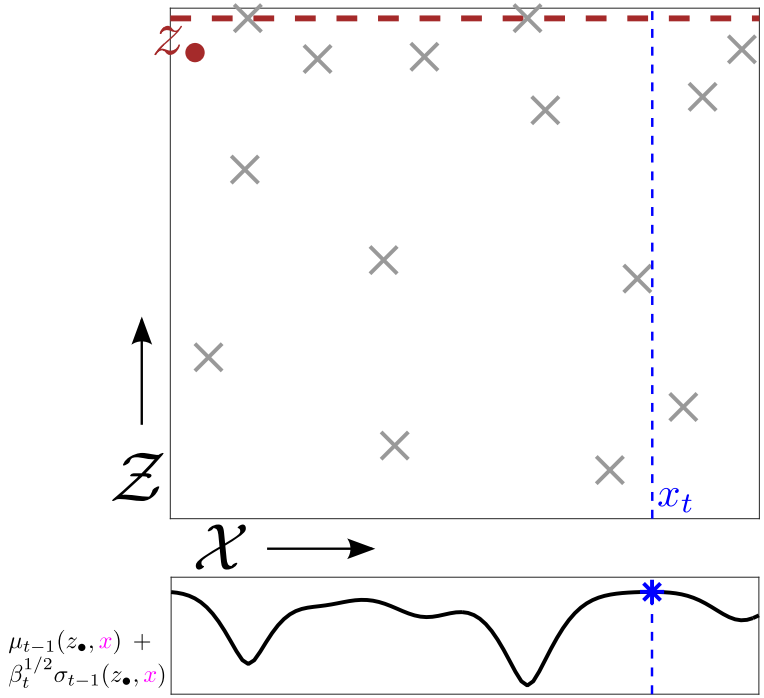
std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

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

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

BOCA: Bayesian Optimisation with Continuous Approximations

(Kandasamy et al. ICML 2017)



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

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

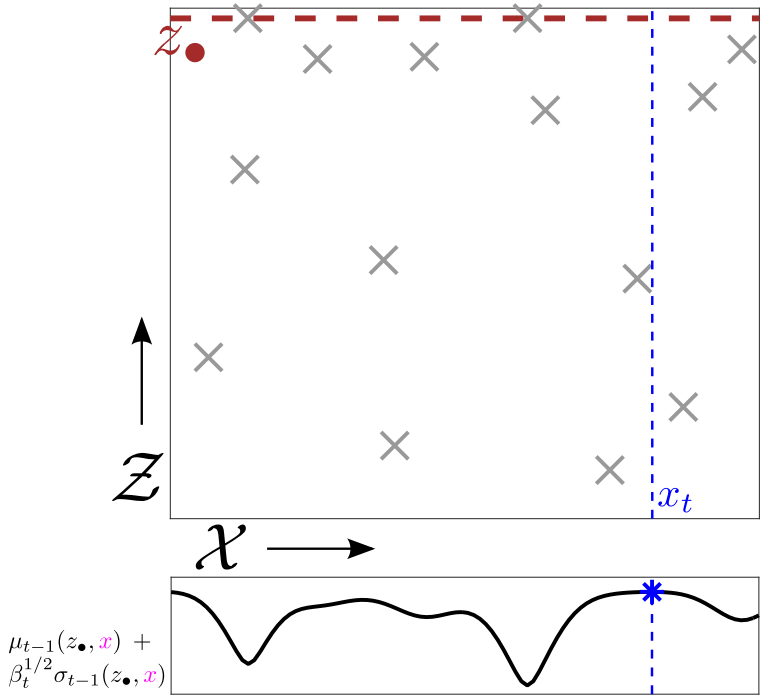
std-dev $\quad \sigma_{t-1} : \mathcal{Z} \times \mathcal{X} \to \mathbb{R}_+$

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

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

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 $f(x) = g(z_\bullet, x)$.

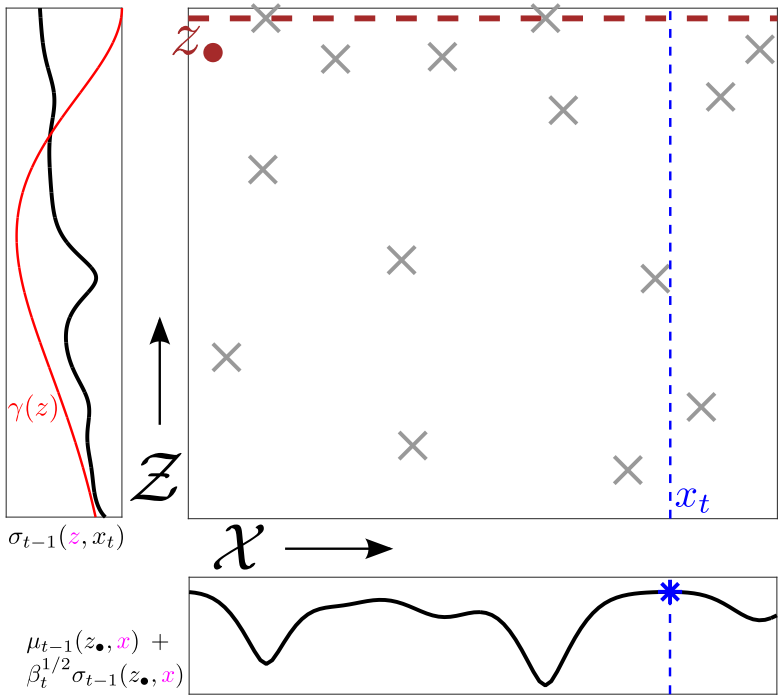
$$x_t = \underset{x \in \mathcal{X}}{\operatorname{argmax}} \ \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

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

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

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 $f(x) = g(z_\bullet, x)$.

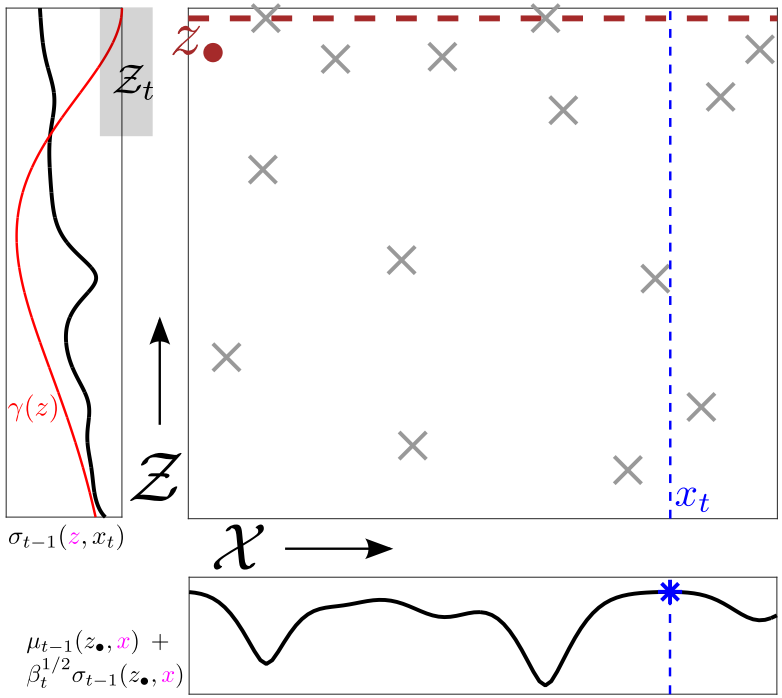
$$x_t = \underset{x \in \mathcal{X}}{\operatorname{argmax}} \ \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

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

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

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 $f(x) = g(z_\bullet, x)$.

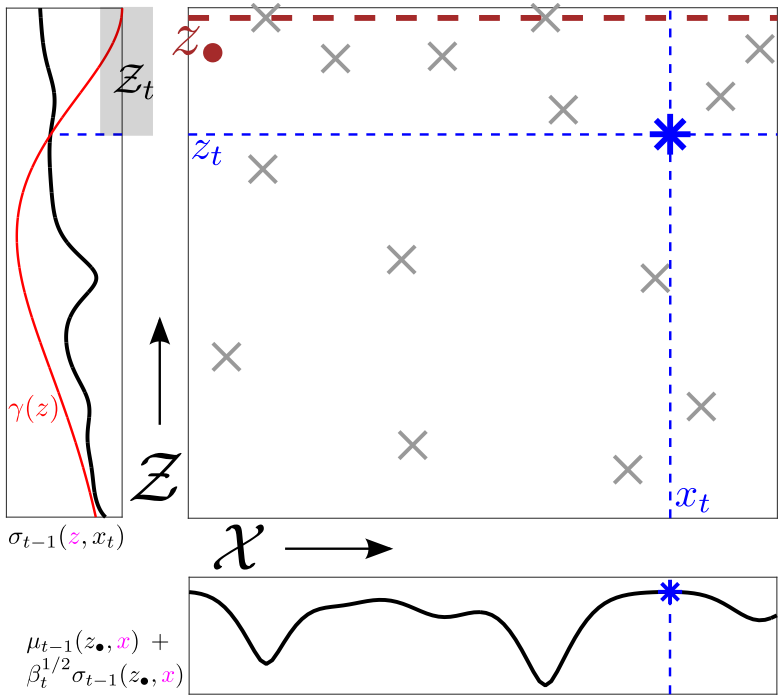
$$x_t = \underset{x \in \mathcal{X}}{\operatorname{argmax}} \ \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

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

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

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 $f(x) = g(z_\bullet, x)$.

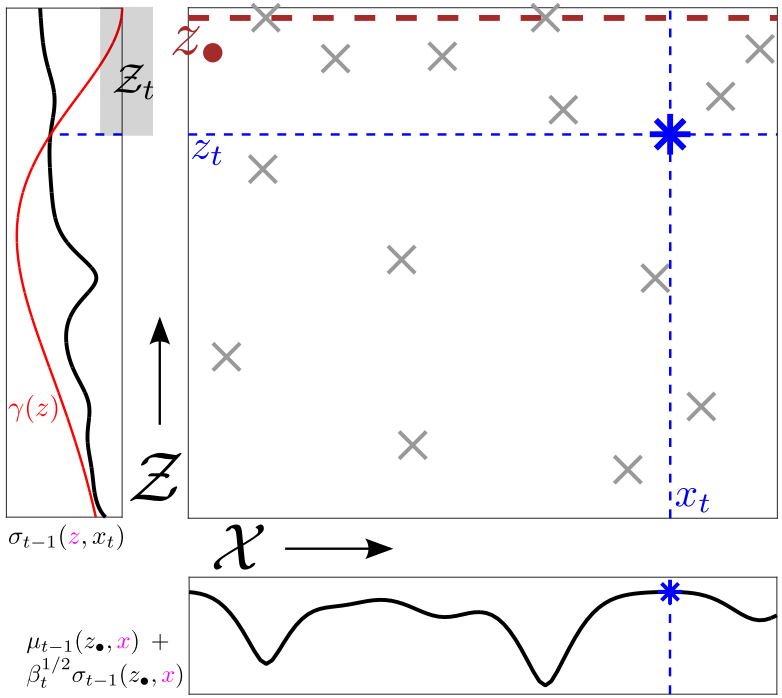
$$x_t = \underset{x \in \mathcal{X}}{\operatorname{argmax}} \ \mu_{t-1}(z_\bullet, x) + \beta_t^{1/2}\sigma_{t-1}(z_\bullet, x)$$

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

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

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 $f(x) = g(z_\bullet, x)$.

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

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

Theoretical Results for BOCA

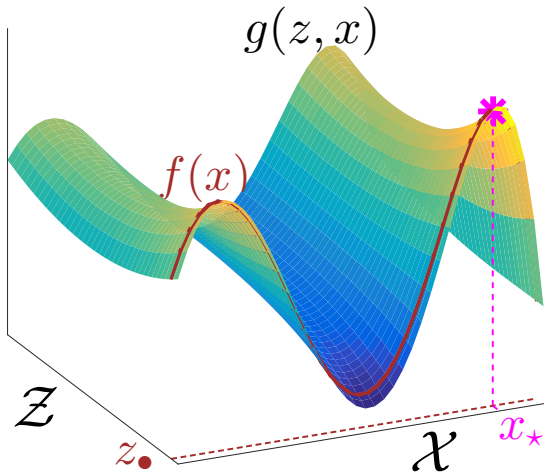
$$g \sim \mathcal{GP}(\mathbf{0}, \kappa), \quad \kappa : (\mathcal{Z} \times \mathcal{X})^2 \to \mathbb{R}.$$

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

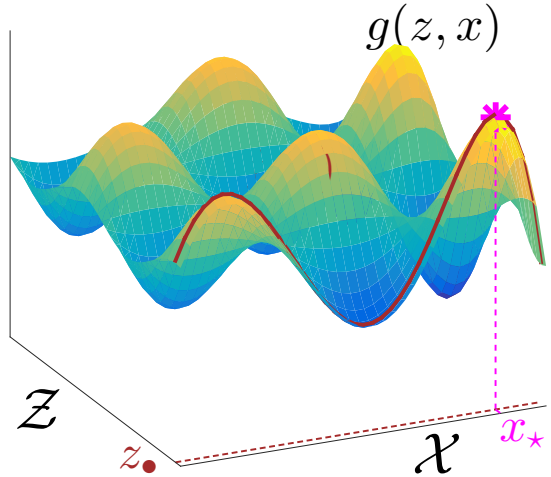
Theoretical Results for BOCA

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

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



"good"

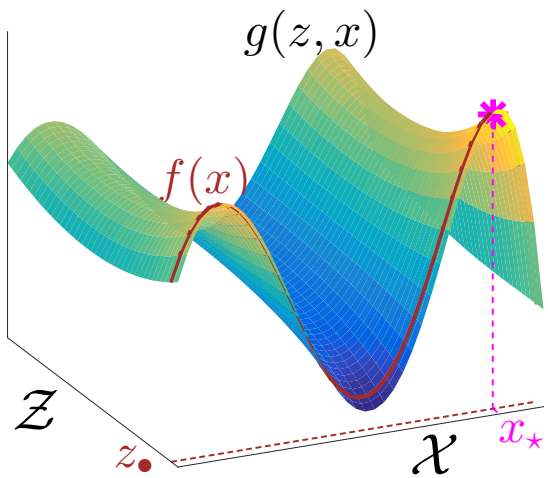


"bad"

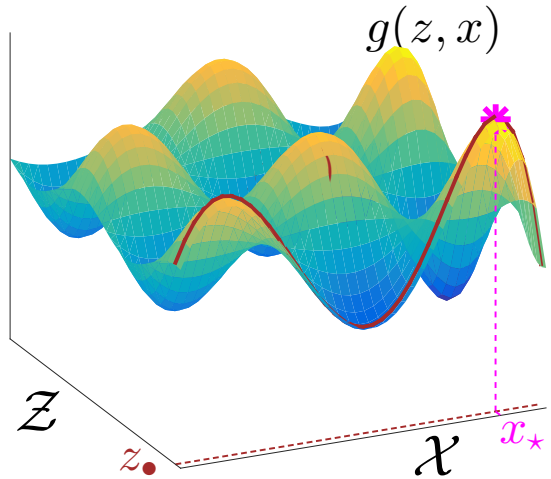
Theoretical Results for BOCA

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

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



"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)

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

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)

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

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

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

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

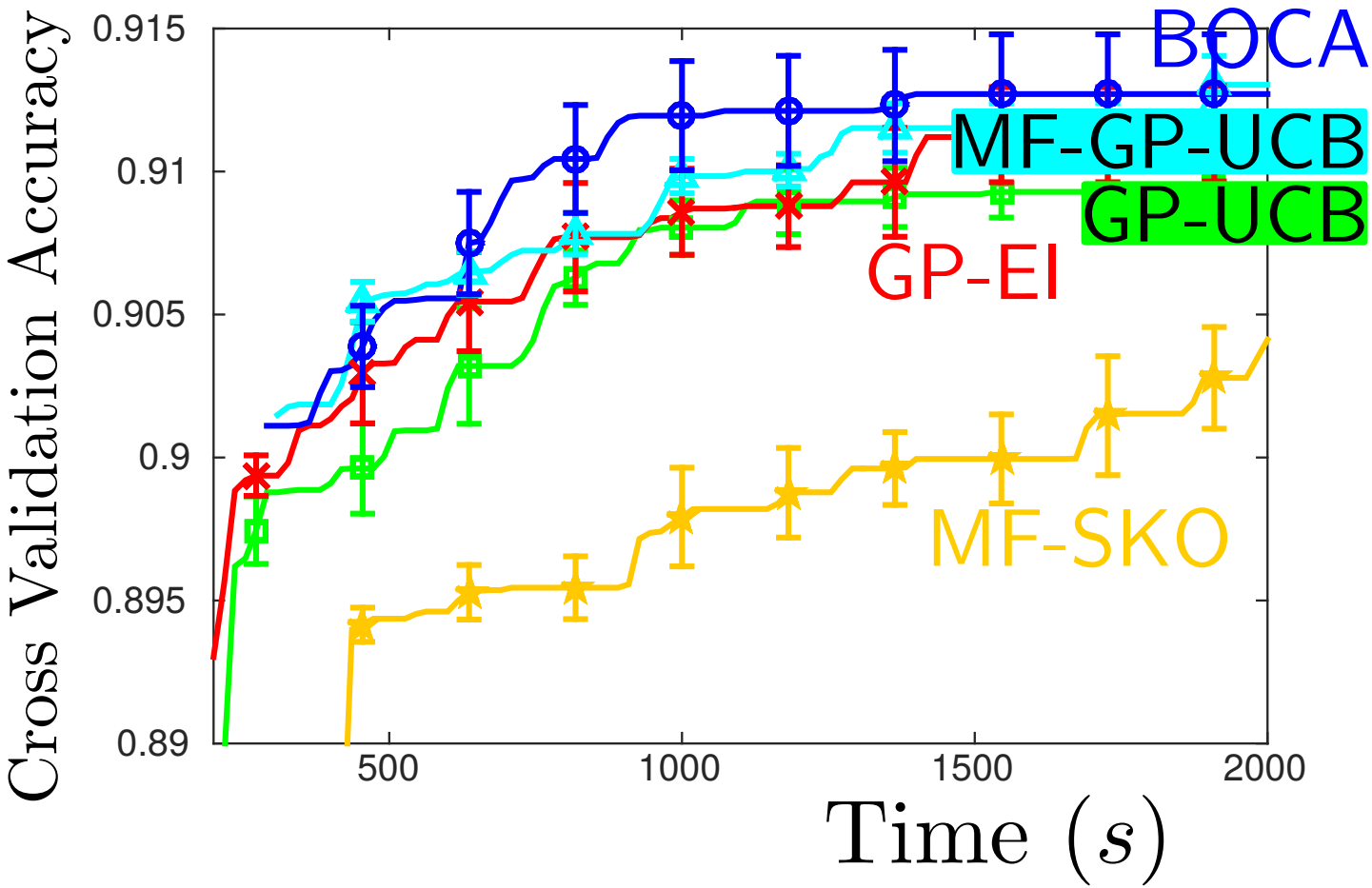
If $h_{\mathcal{Z}}$ is large (good approximations), $\text{vol}(\mathcal{X}_\alpha) \ll \text{vol}(\mathcal{X})$, and BOCA is much better than GP-UCB.

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).

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).



Take-aways

- BOCA: a multi-fidelity optimisation algorithm when you have access to continuous approximations.
- Choose higher fidelity only after controlling uncertainty/variance at lower fidelities.
- Theoretically/empirically outperforms strategies that ignore the approximations or use only a finite number of fidelities.

Take-aways

- BOCA: a multi-fidelity optimisation algorithm when you have access to continuous approximations.
- Choose higher fidelity only after controlling uncertainty/variance at lower fidelities.
- Theoretically/empirically outperforms strategies that ignore the approximations or use only a finite number of fidelities.

Thank you.

Poster tonight @ Gallery #49.