Parallelised Bayesian Optimisation via Thompson Sampling

Kirthevasan Kandasamy

Akshay Krishnamurthy

Jeff Schneider

Barnabás Póczos

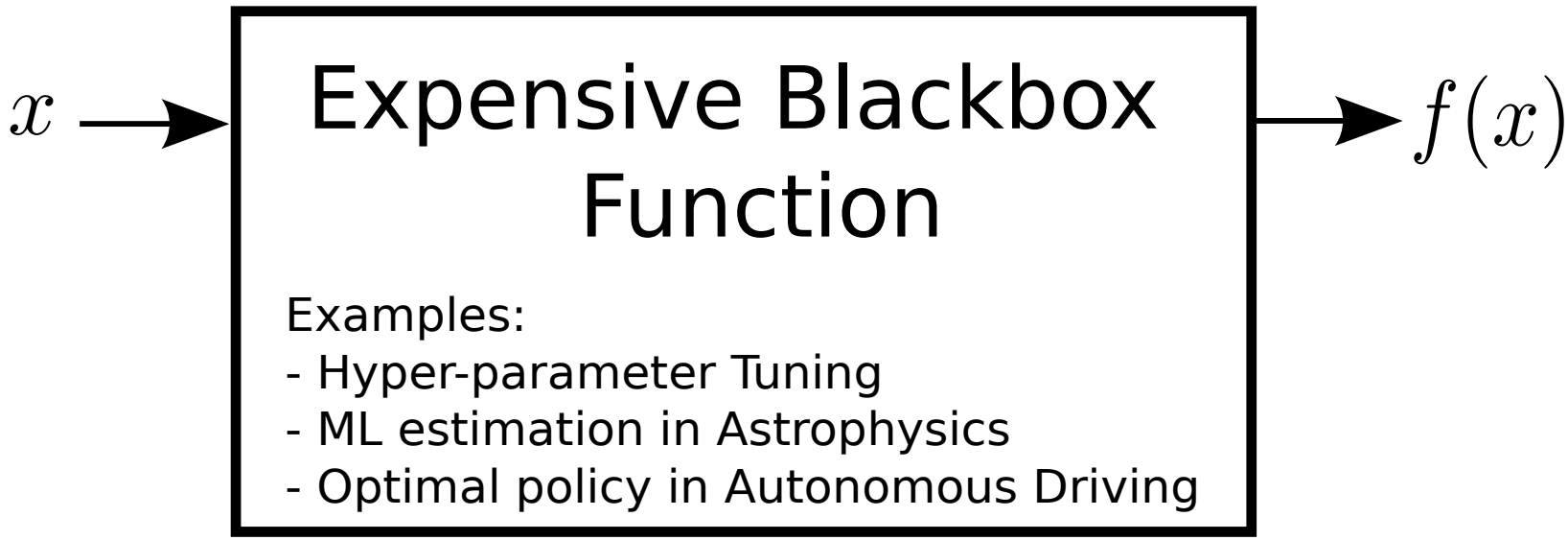
AISTATS 2018

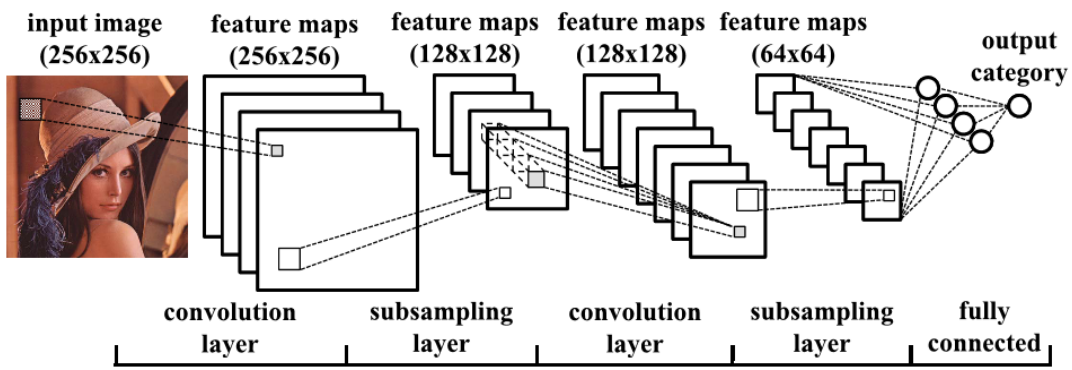
Carnegie Mellon University

UMASS AMHERST

Auton Lab

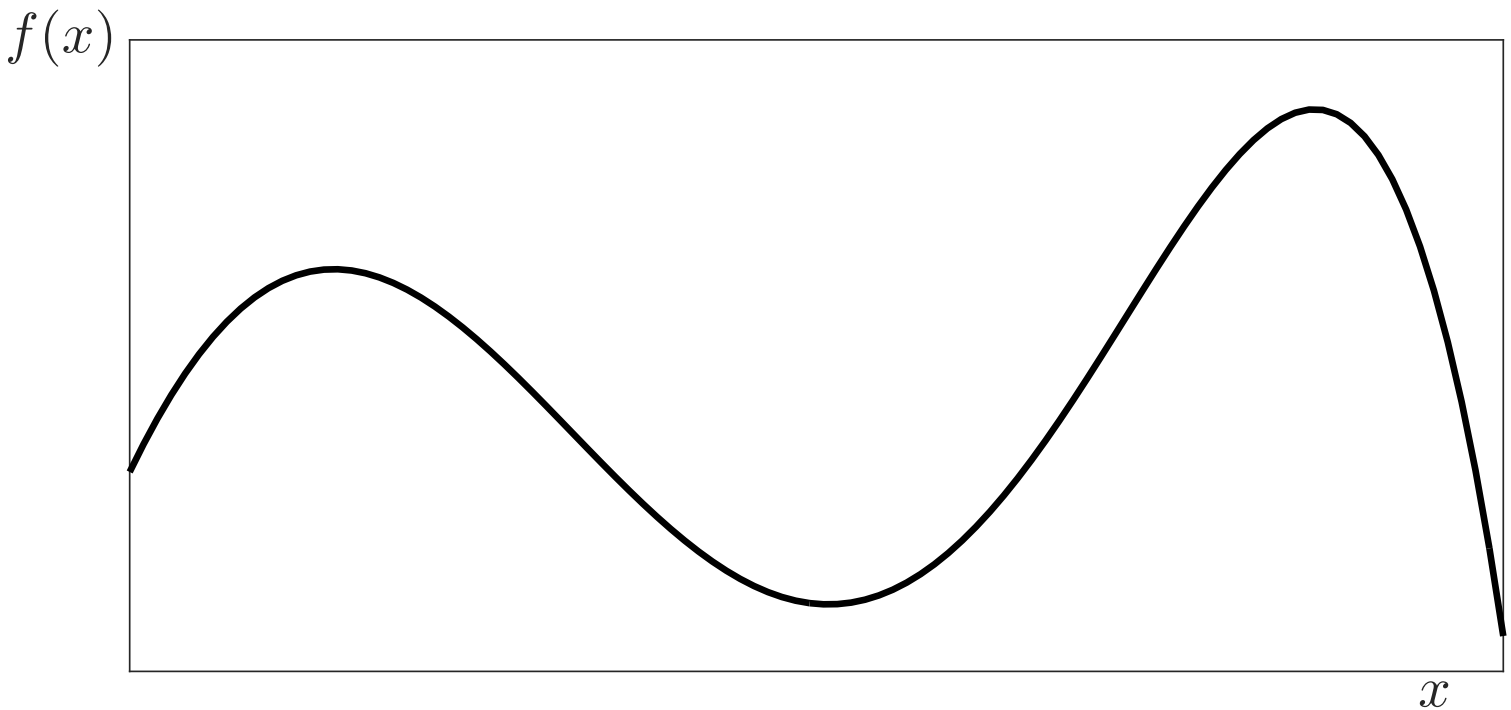
Black-box Optimisation





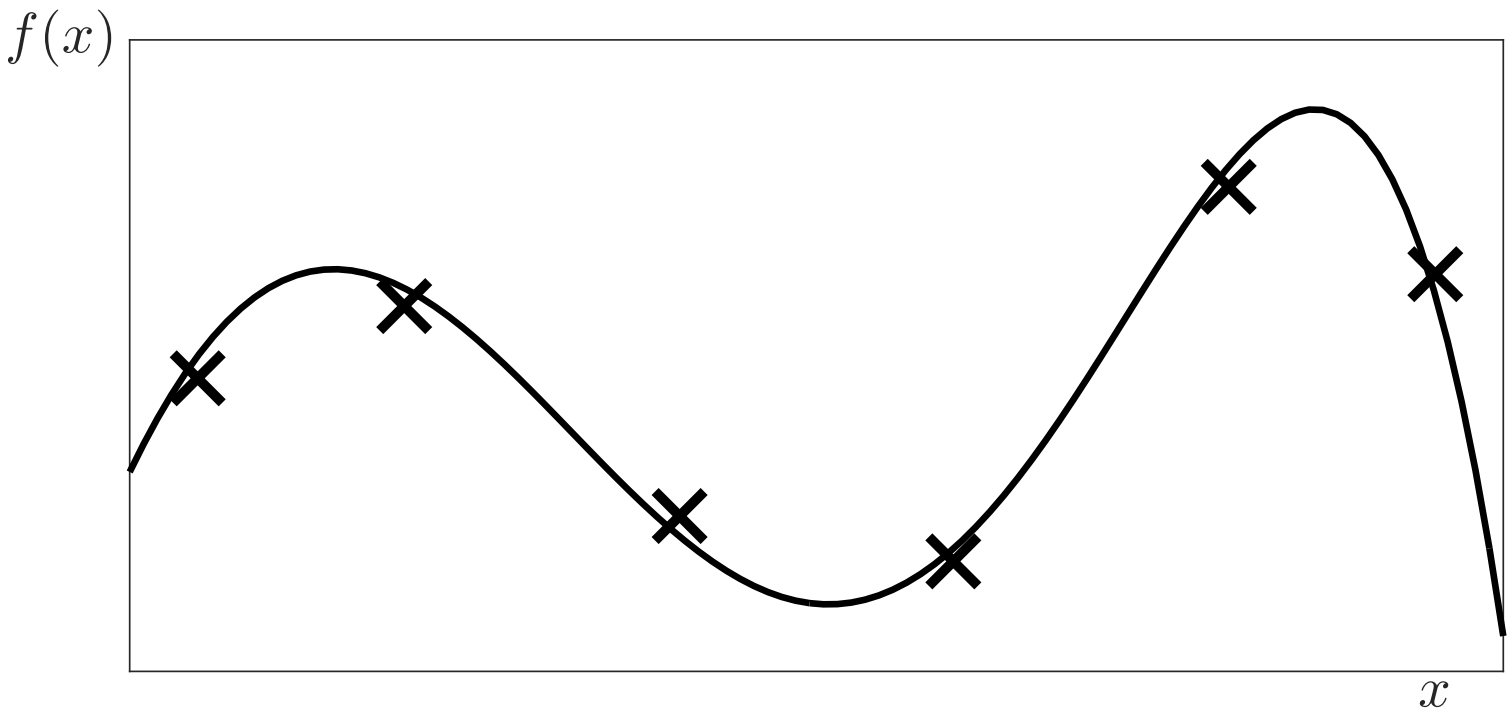
Black-box Optimisation

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



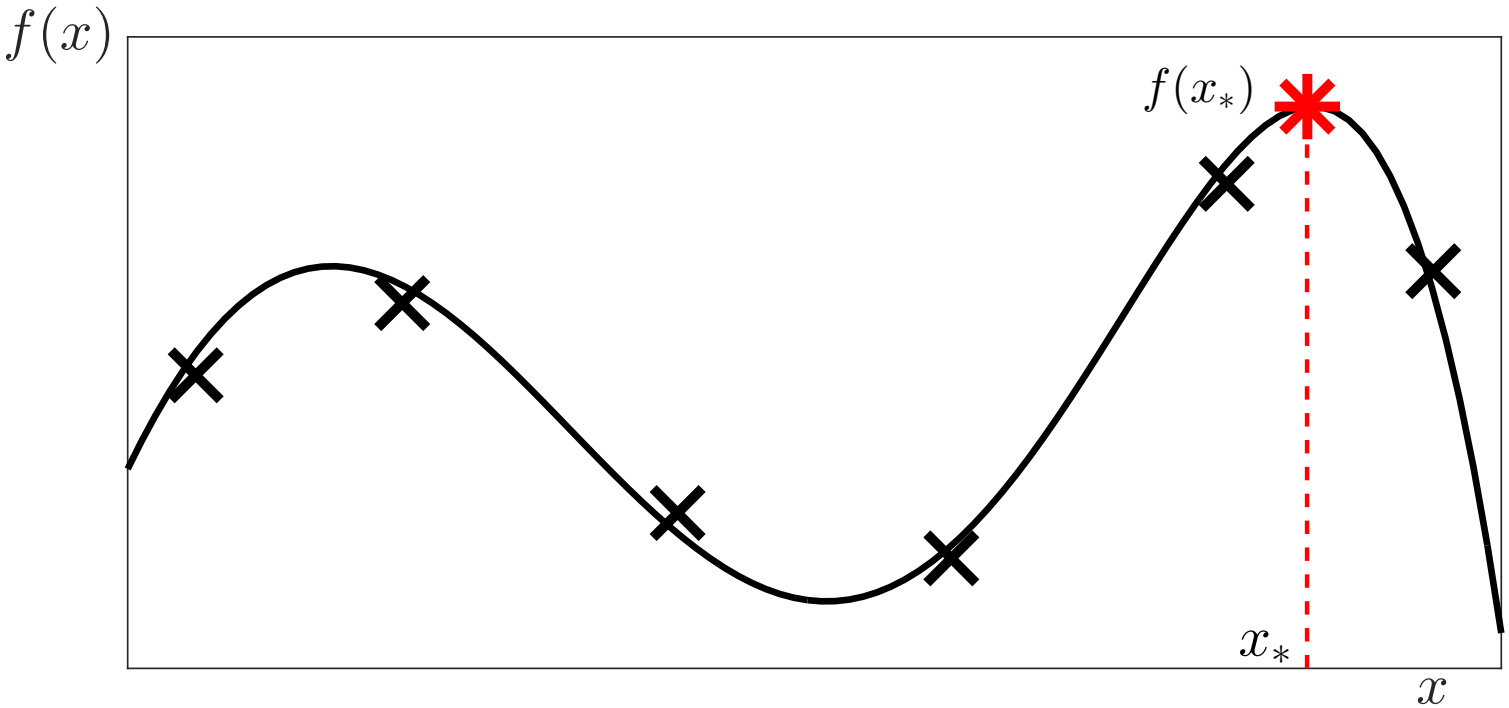
Black-box Optimisation

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



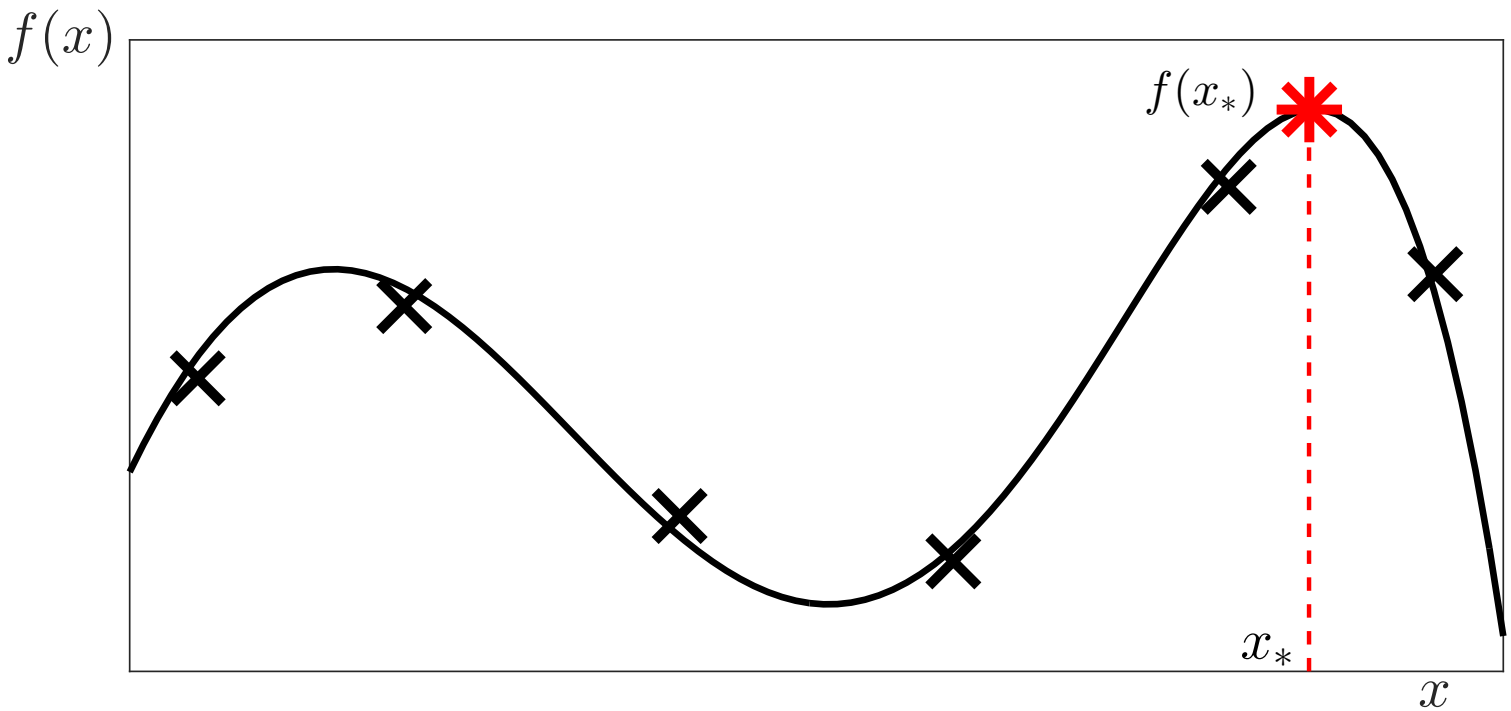
Black-box Optimisation

$f : \mathcal{X} \to \mathbb{R}$ 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

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

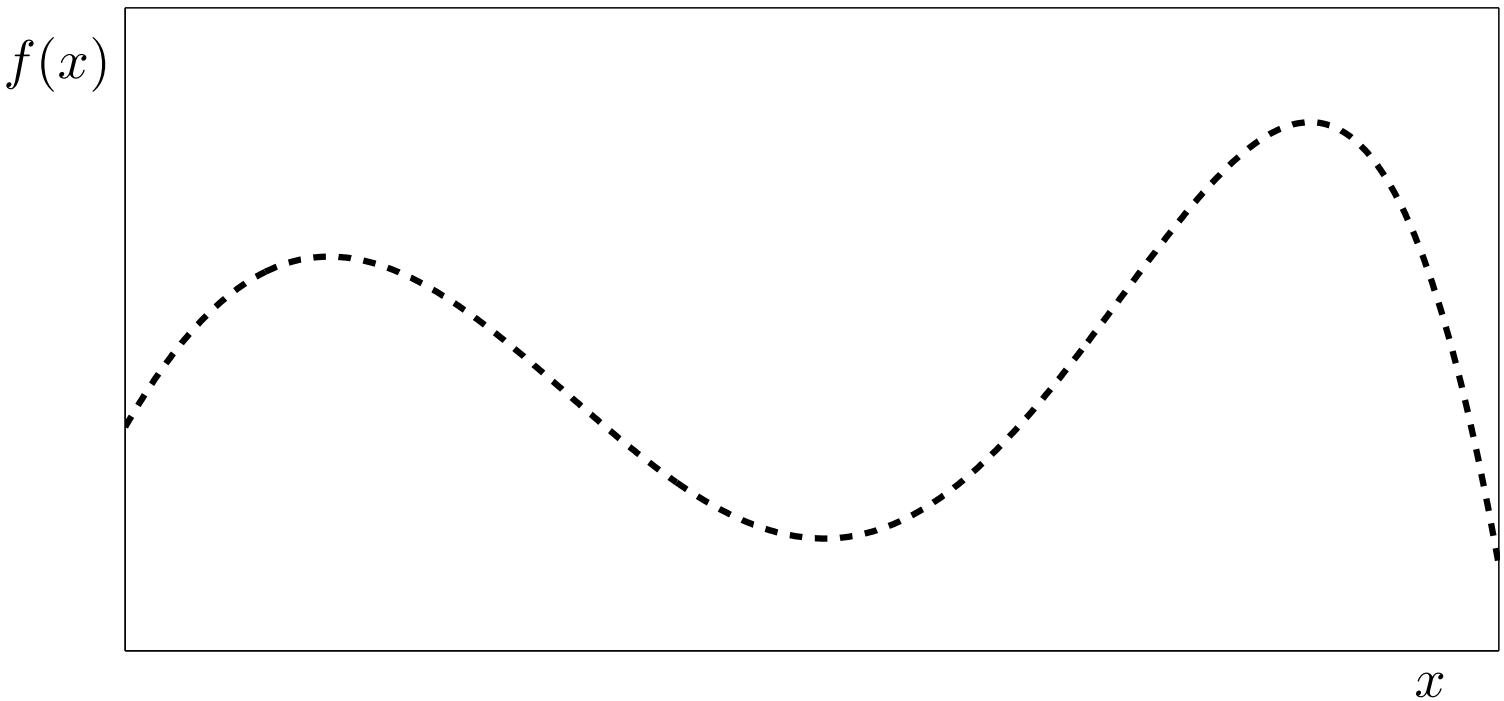
Gaussian Processes ($\mathcal{GP}$)

$\mathcal{GP}(\mu, \kappa)$: 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 $\mathbb{R}$.

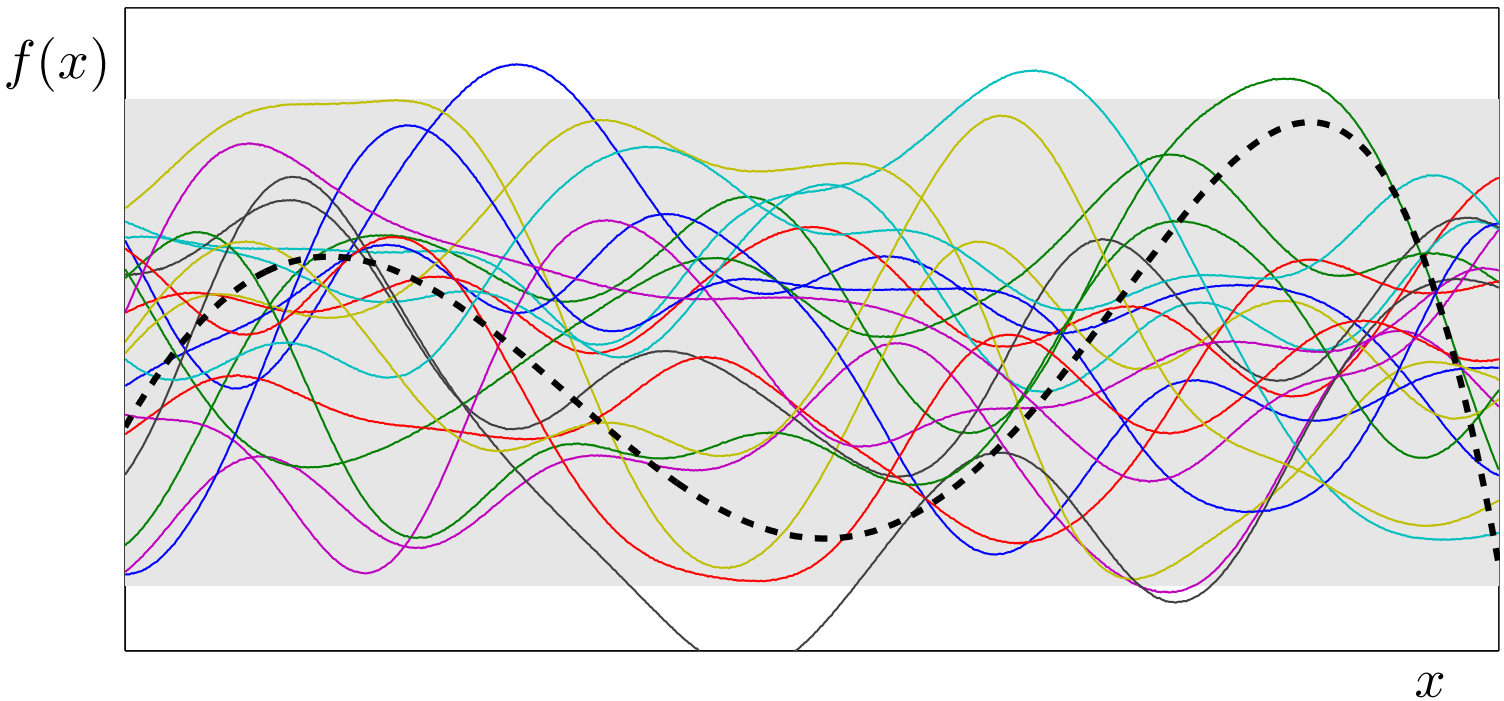
Functions with no observations



Gaussian Processes ($\mathcal{GP}$)

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

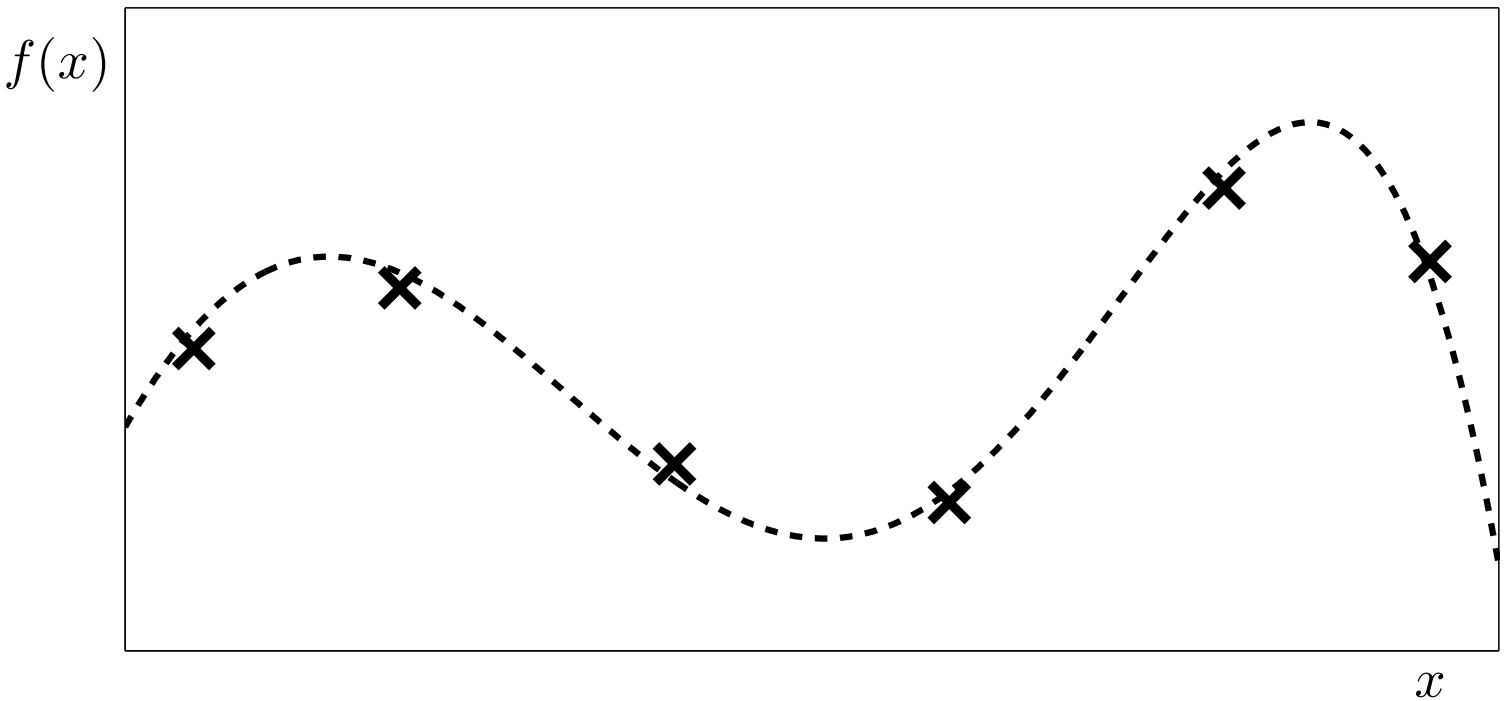
Prior $\mathcal{GP}$



Gaussian Processes ($\mathcal{GP}$)

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

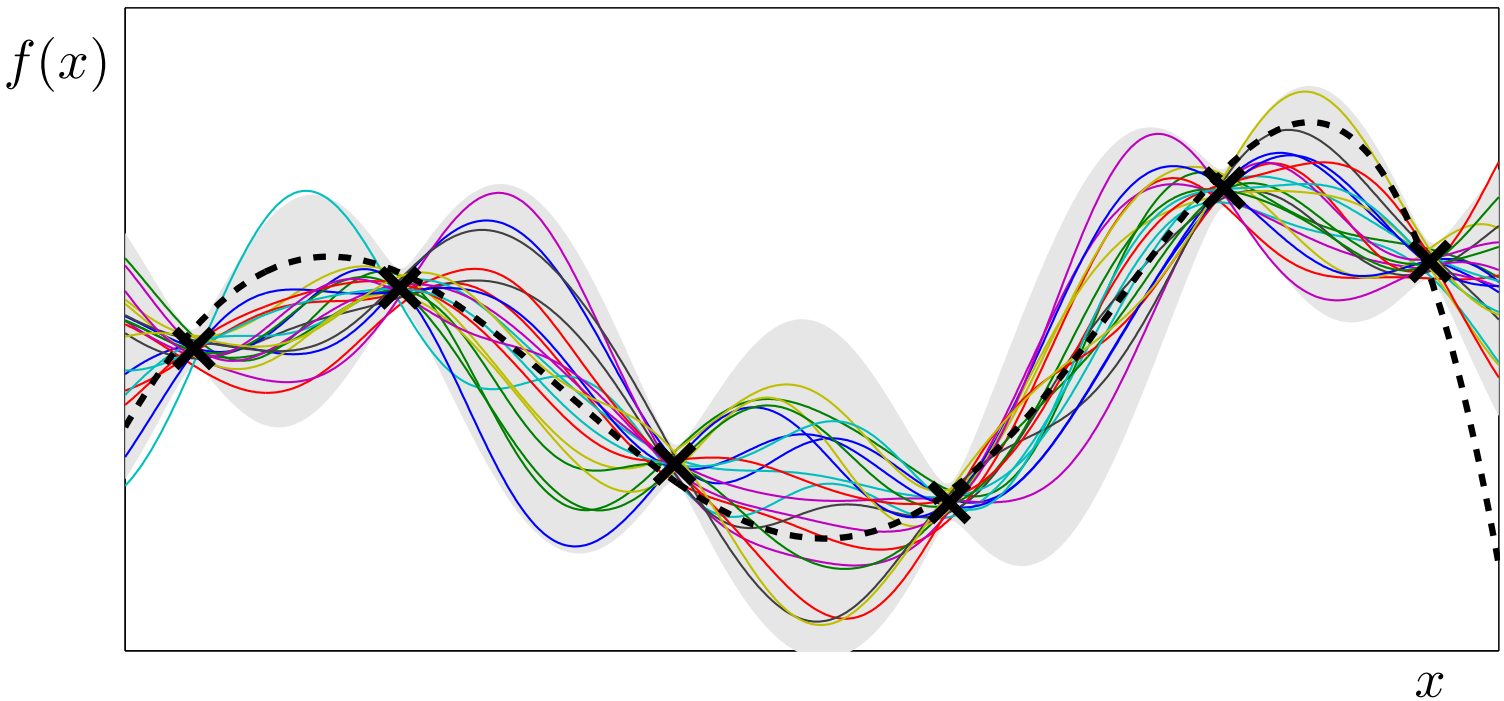
Observations



Gaussian Processes ($\mathcal{GP}$)

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

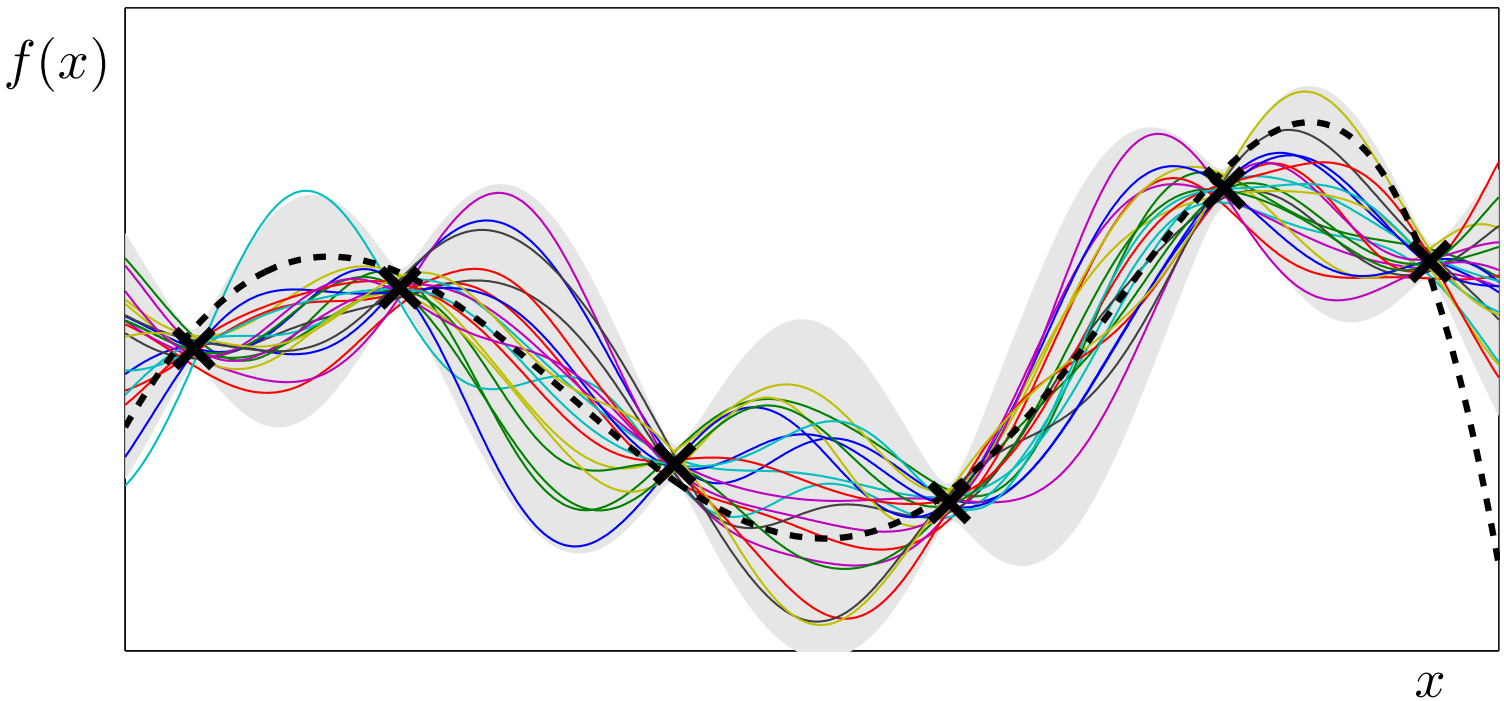
Posterior $\mathcal{GP}$ given observations



Gaussian Processes ($\mathcal{GP}$)

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

Posterior $\mathcal{GP}$ given observations



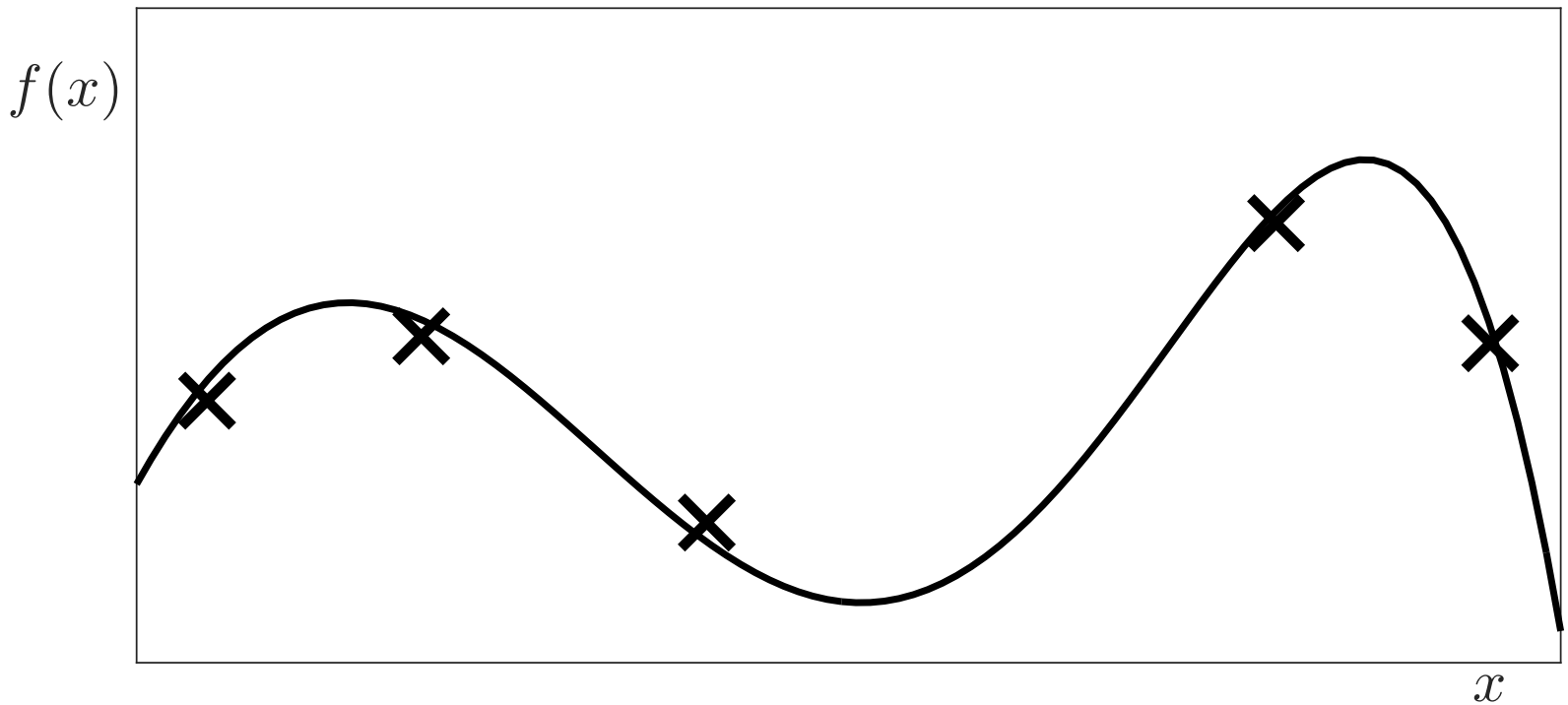
After t observations, $f(x) \sim \mathcal{N}(\,\mu_t(x),\, \sigma_t^2(x)\,)$.

Gaussian Process Bandit (Bayesian) Optimisation

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

Several criteria for picking next point:

GP-UCB (Srinivas et al. 2010), GP-EI (Mockus & Mockus, 1991).

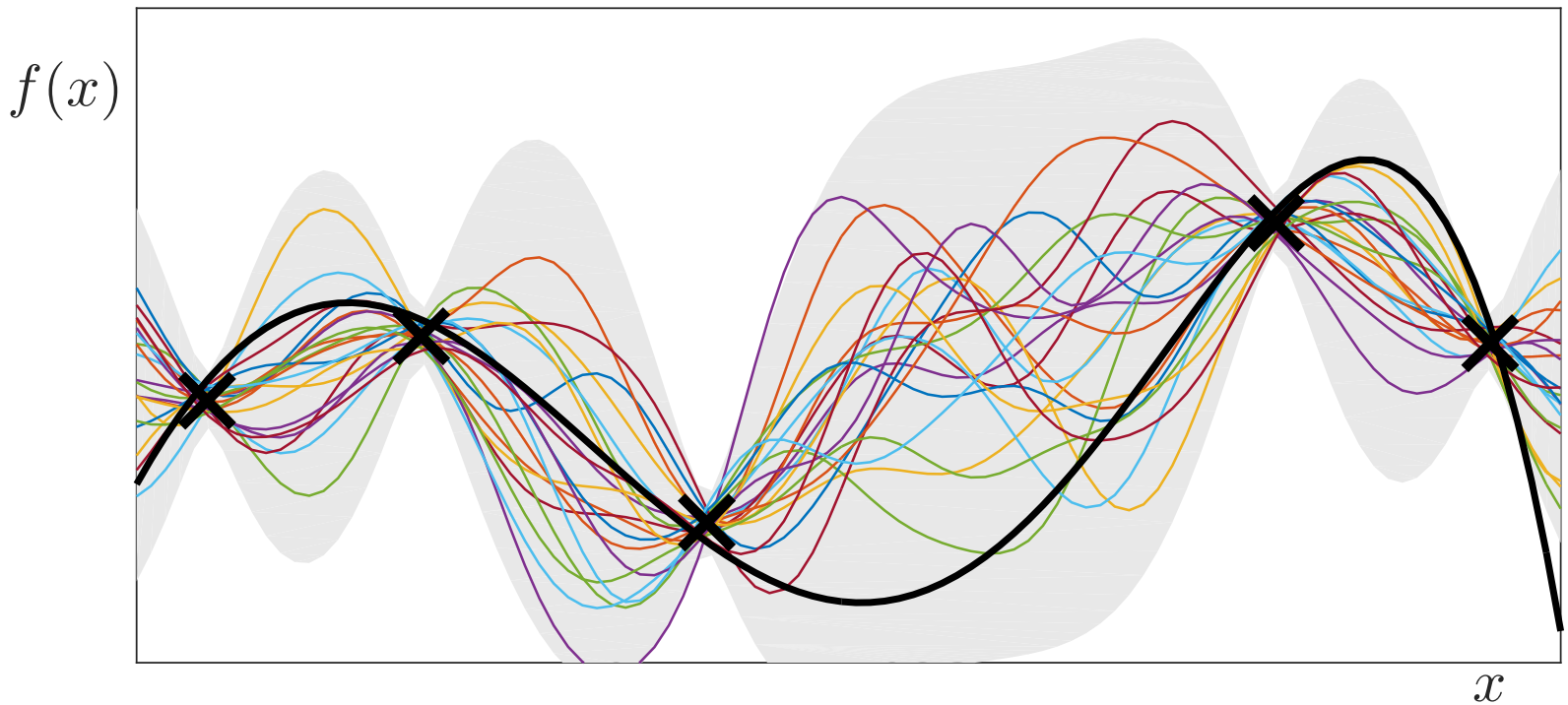


Gaussian Process Bandit (Bayesian) Optimisation

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

Several criteria for picking next point:

GP-UCB (Srinivas et al. 2010), GP-EI (Mockus & Mockus, 1991).



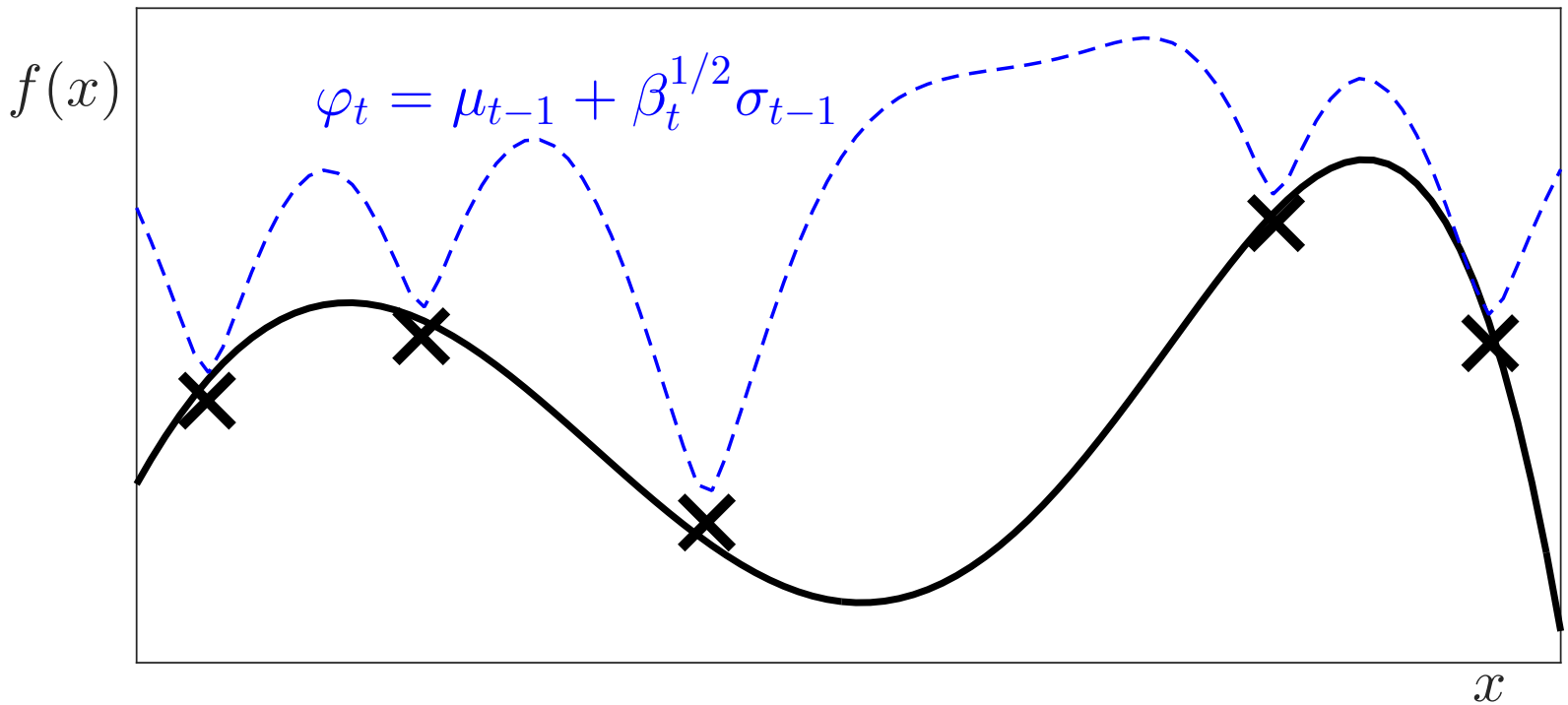
1) Compute posterior $\mathcal{GP}$.

Gaussian Process Bandit (Bayesian) Optimisation

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

Several criteria for picking next point:

GP-UCB (Srinivas et al. 2010), GP-EI (Mockus & Mockus, 1991).



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

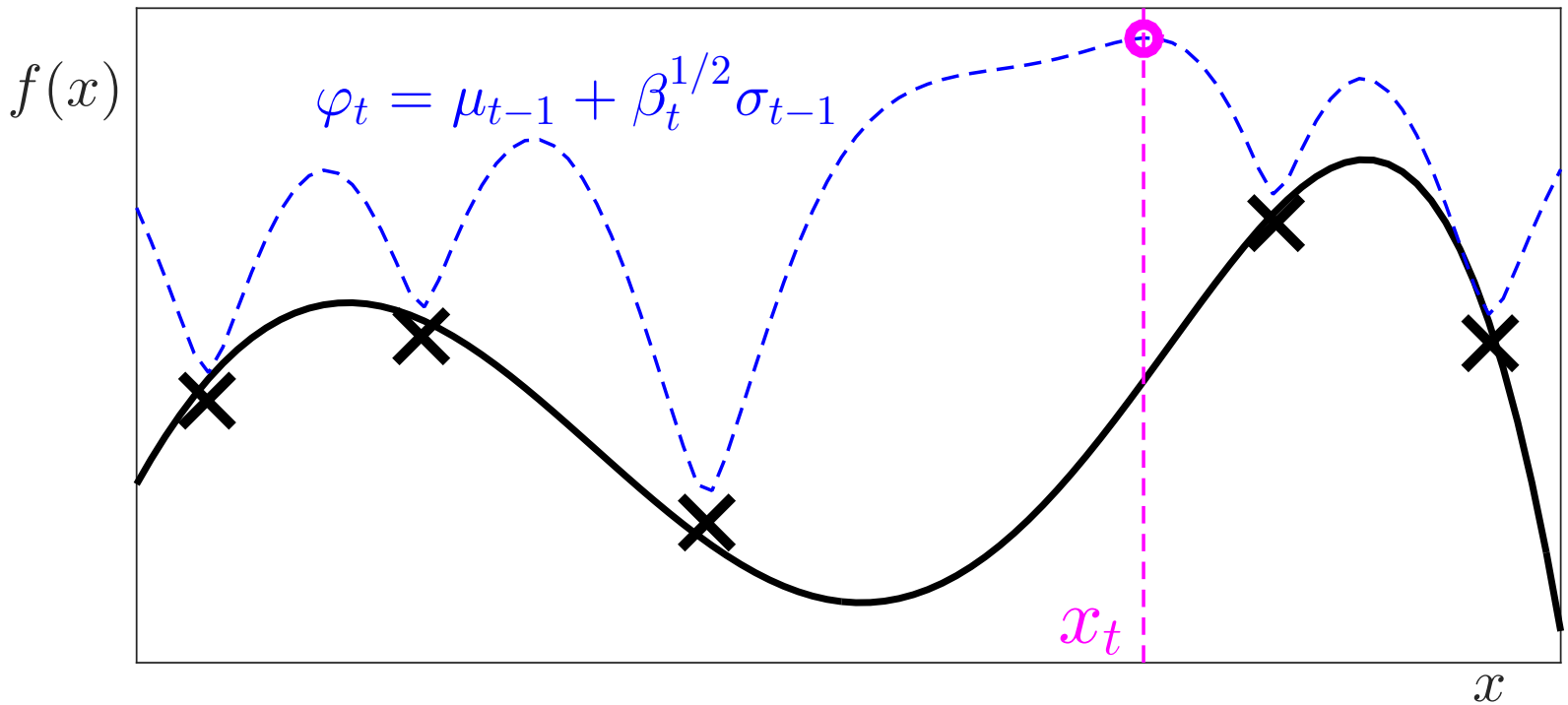
2) Construct acquisition φ_t.

Gaussian Process Bandit (Bayesian) Optimisation

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

Several criteria for picking next point:

GP-UCB (Srinivas et al. 2010), GP-EI (Mockus & Mockus, 1991).



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

2) Construct acquisition φ_t.

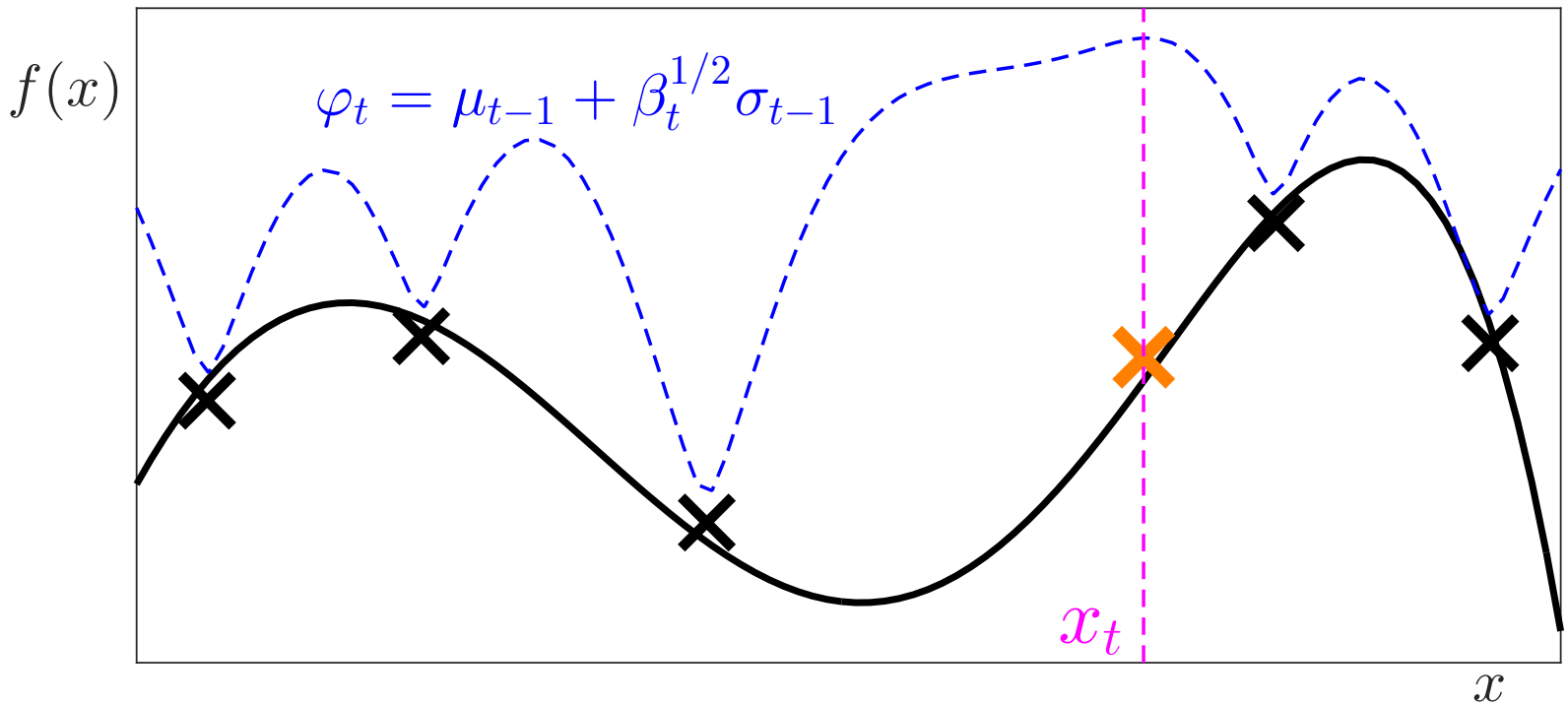
3) Choose $x_t = \mathrm{argmax}_x\, \varphi_t(x)$.

Gaussian Process Bandit (Bayesian) Optimisation

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

Several criteria for picking next point:

GP-UCB (Srinivas et al. 2010), GP-EI (Mockus & Mockus, 1991).



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

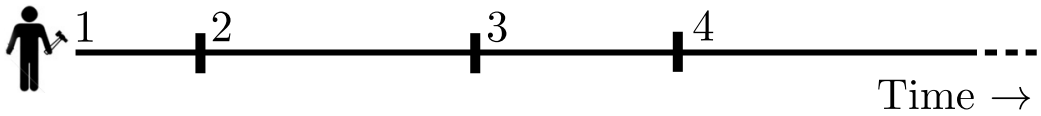
2) Construct acquisition φ_t.

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

4) Evaluate f at x_t.

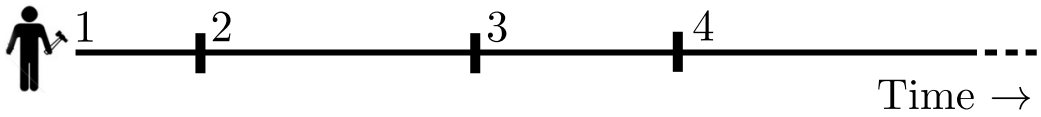
This work: Parallel Evaluations

Sequential evaluations with one worker

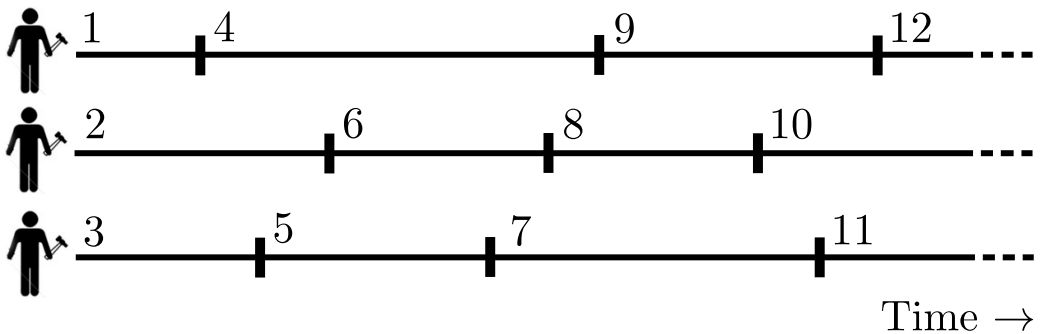


This work: Parallel Evaluations

Sequential evaluations with one worker

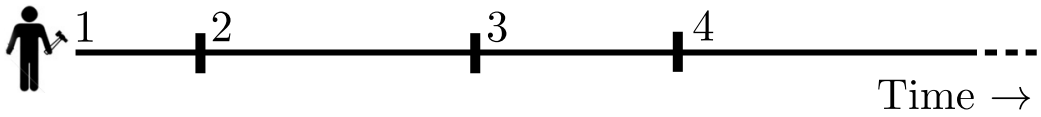


Parallel evaluations with M workers (Asynchronous)

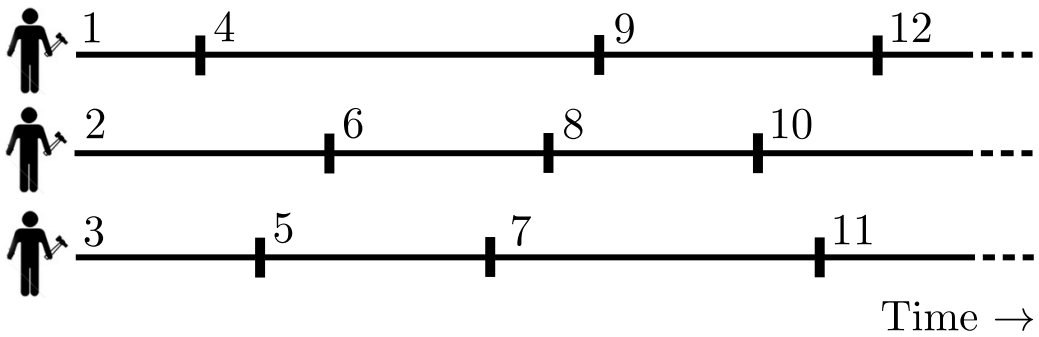


This work: Parallel Evaluations

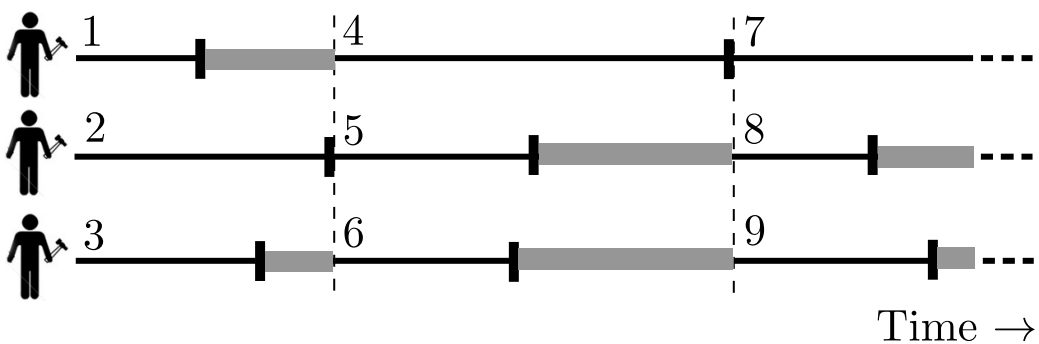
Sequential evaluations with one worker



Parallel evaluations with M workers (Asynchronous)

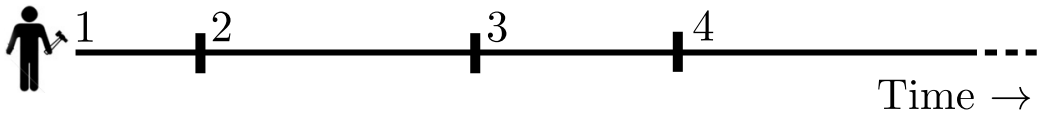


Parallel evaluations with M workers (Synchronous)



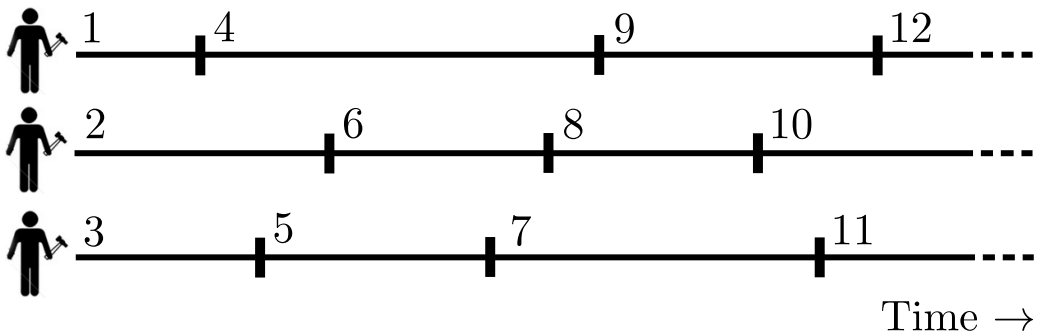
This work: Parallel Evaluations

Sequential evaluations with one worker



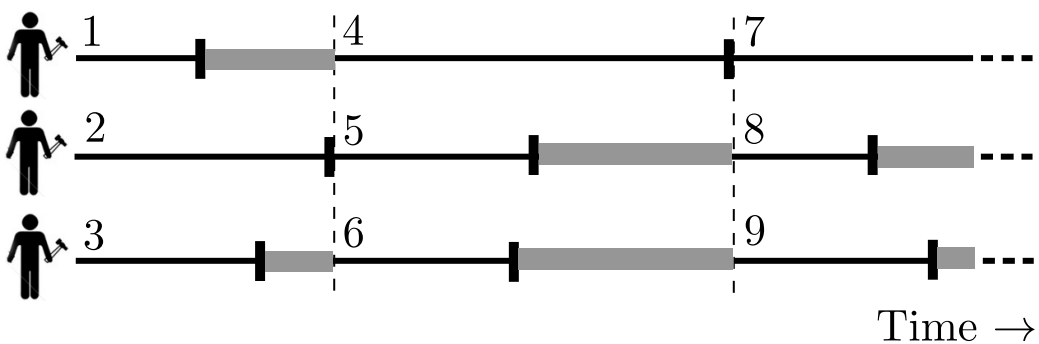
j^{th} job has feedback from all previous $j-1$ evaluations.

Parallel evaluations with M workers (Asynchronous)



j^{th} job missing feedback from exactly $M-1$ evaluations.

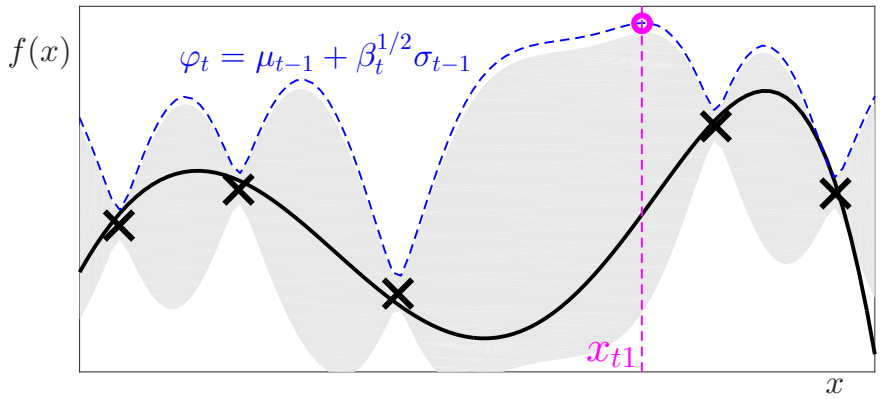
Parallel evaluations with M workers (Synchronous)



j^{th} job missing feedback from $\leq M-1$ evaluations.

Challenges in parallel BO: encouraging diversity

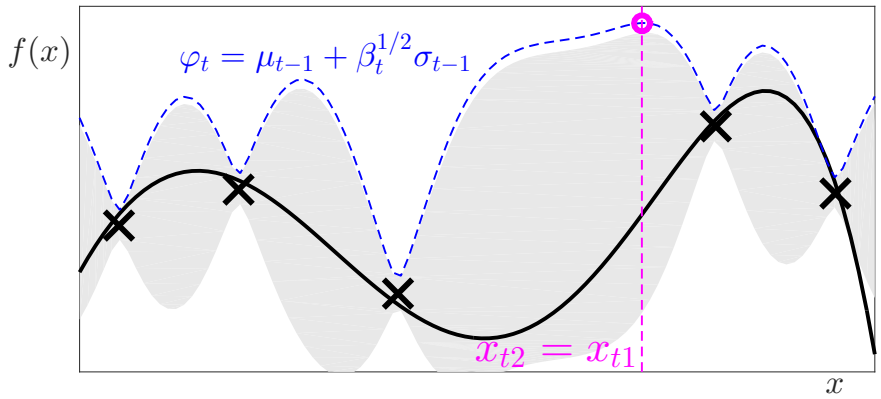
Direct application of UCB in the synchronous setting . . .



- First worker: maximise acquisition, $x_{t1} = \operatorname{argmax} \varphi_t(x)$.

Challenges in parallel BO: encouraging diversity

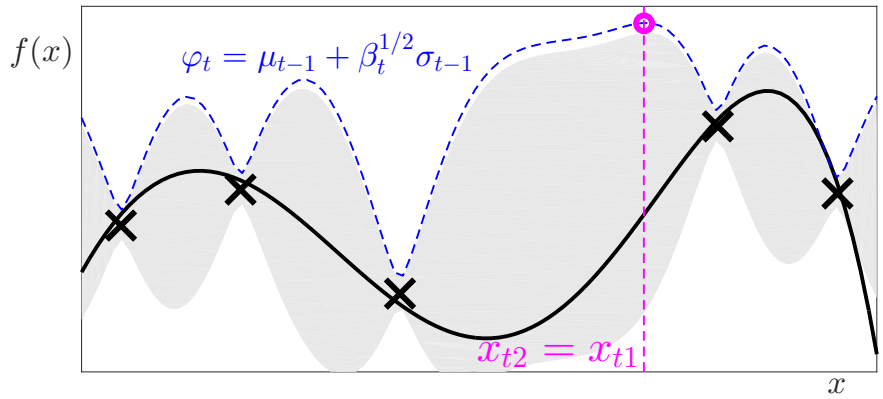
Direct application of UCB in the synchronous setting . . .



- First worker: maximise acquisition, $x_{t1} = \operatorname{argmax} \varphi_t(x)$.
- Second worker: acquisition is the same! $x_{t1} = x_{t2}$

Challenges in parallel BO: encouraging diversity

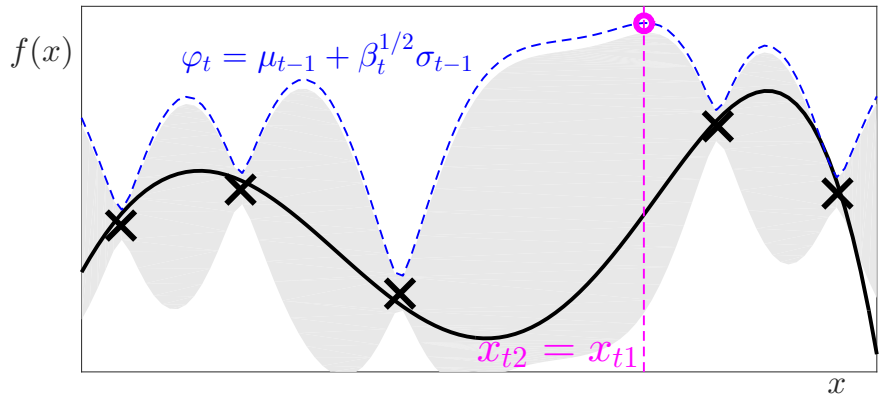
Direct application of UCB in the synchronous setting . . .



- First worker: maximise acquisition, $x_{t1} = \operatorname{argmax} \varphi_t(x)$.
- Second worker: acquisition is the same! $x_{t1} = x_{t2}$
- $x_{t1} = x_{t2} = \cdots = x_{tM}$.

Challenges in parallel BO: encouraging diversity

Direct application of UCB in the synchronous setting . . .



- First worker: maximise acquisition, $x_{t1} = \operatorname{argmax} \varphi_t(x)$.
- Second worker: acquisition is the same! $x_{t1} = x_{t2}$
- $x_{t1} = x_{t2} = \cdots = x_{tM}$.

Direct application of popular (deterministic) strategies, e.g. GP-UCB, GP-EI, etc. do not work. Need to "encourage diversity".

Challenges in parallel BO: encouraging diversity

- Add hallucinated observations.

  (Ginsbourger et al. 2011, Janusevkis et al. 2012)
- Optimise an acquisition over $\mathcal{X}^M$ (e.g. M-product UCB).

  (Wang et al 2016, Wu & Frazier 2017)
- Resort to heuristics, typically requires additional hyper-parameters and/or computational routines.

  (Contal et al. 2013, Gonzalez et al. 2015, Shah & Ghahramani 2015, Wang et al. 2017, Wang et al. 2018)

Challenges in parallel BO: encouraging diversity

- Add hallucinated observations.

  (Ginsbourger et al. 2011, Janusevkis et al. 2012)
- Optimise an acquisition over $\mathcal{X}^M$ (e.g. M-product UCB).

  (Wang et al 2016, Wu & Frazier 2017)
- Resort to heuristics, typically requires additional hyper-parameters and/or computational routines.

  (Contal et al. 2013, Gonzalez et al. 2015, Shah & Ghahramani 2015, Wang et al. 2017, Wang et al. 2018)

Our Approach: Based on Thompson sampling (Thompson, 1933).

- Conceptually simple: *does not require explicit diversity strategies.*

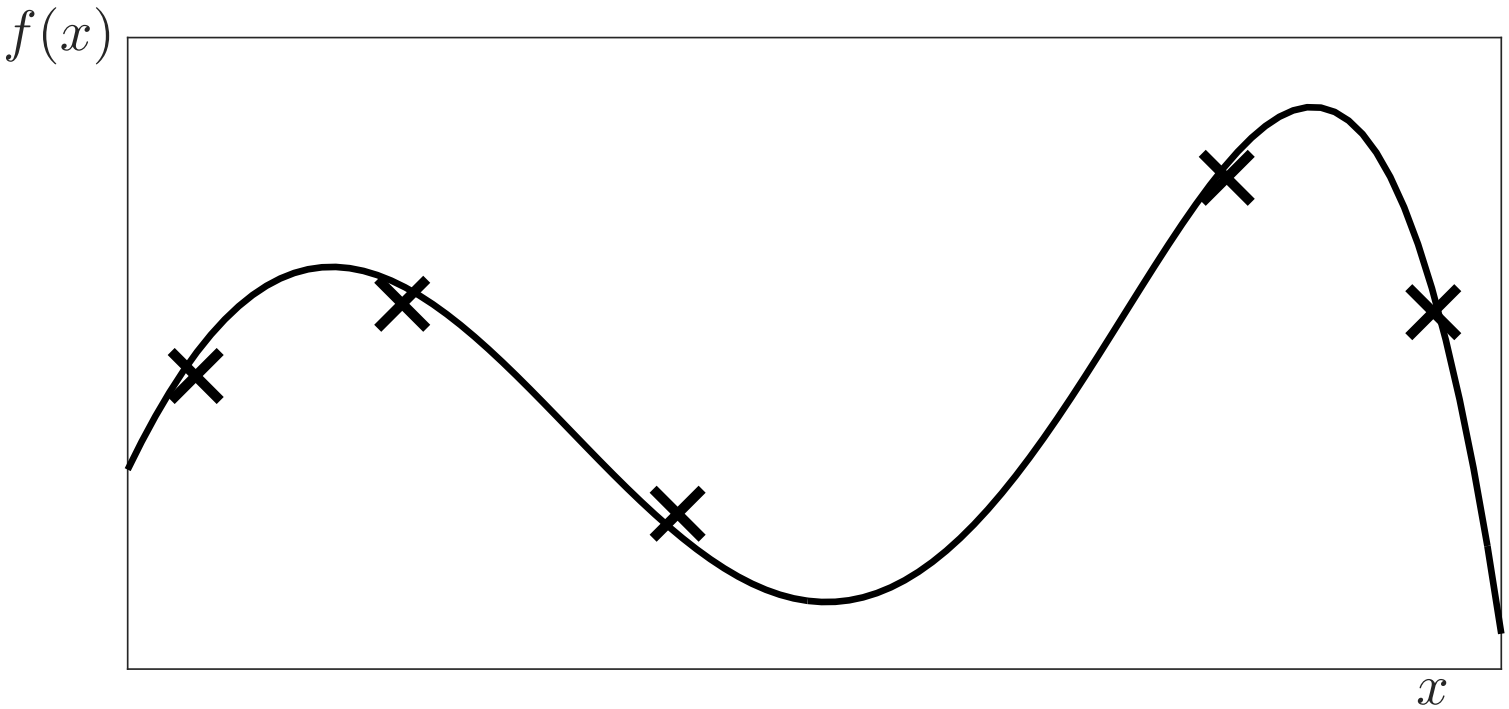
Challenges in parallel BO: encouraging diversity

- Add hallucinated observations.
  (Ginsbourger et al. 2011, Janusevkis et al. 2012)
- Optimise an acquisition over $\mathcal{X}^M$ (e.g. M-product UCB).
  (Wang et al 2016, Wu & Frazier 2017)
- Resort to heuristics, typically requires additional hyper-parameters and/or computational routines.
  (Contal et al. 2013, Gonzalez et al. 2015, Shah & Ghahramani 2015, Wang et al. 2017, Wang et al. 2018)

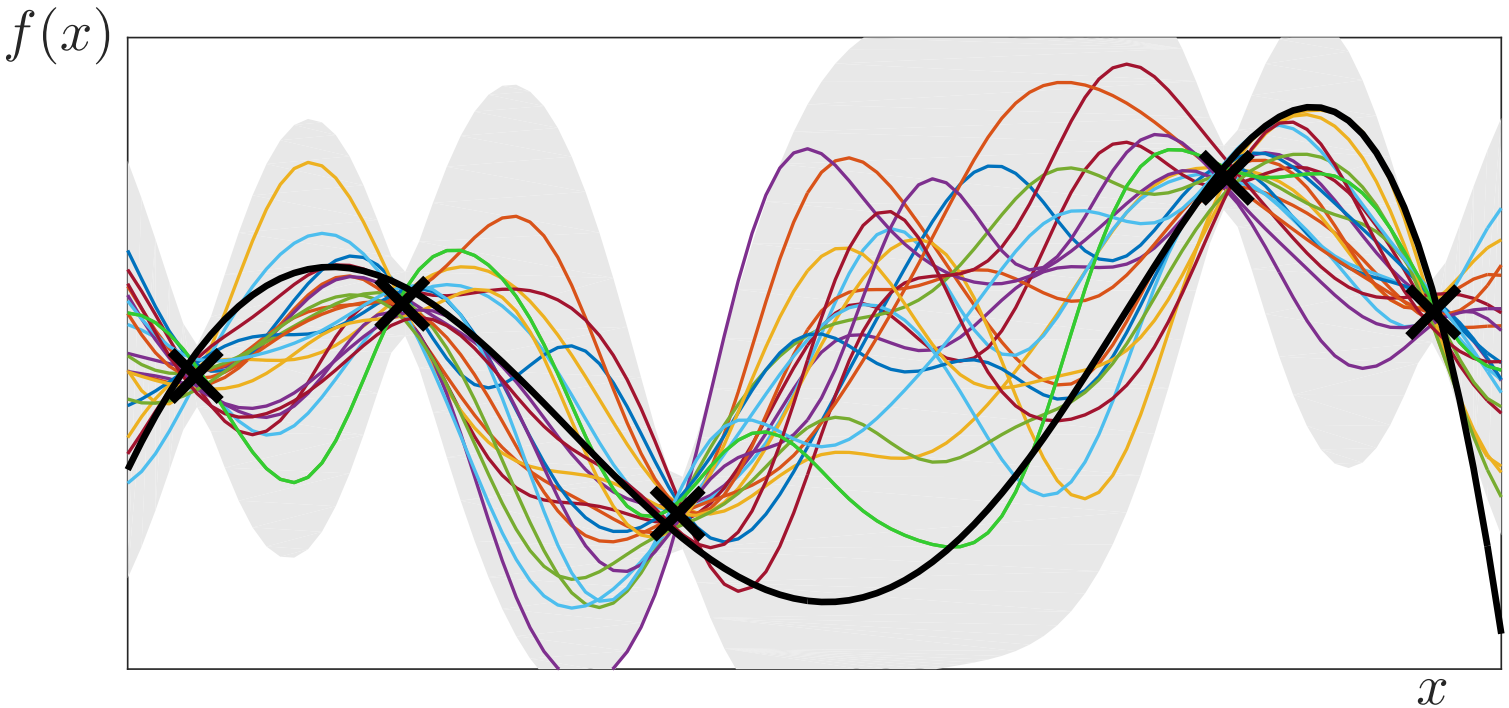
Our Approach: Based on Thompson sampling (Thompson, 1933).

- Conceptually simple: *does not require explicit diversity strategies.*
- Asynchronicity
- Theoretical guarantees

GP Optimisation with Thompson Sampling (Thompson, 1933)

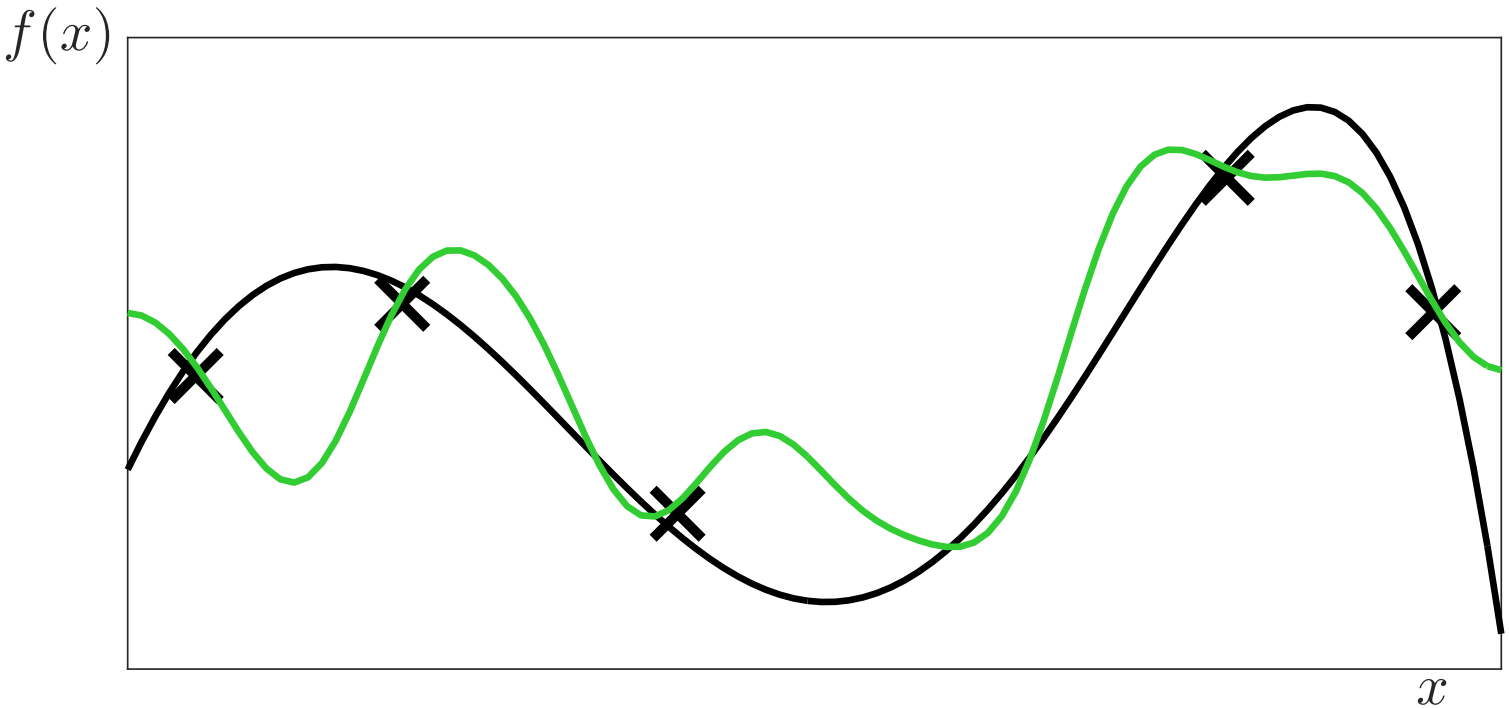


GP Optimisation with Thompson Sampling (Thompson, 1933)



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

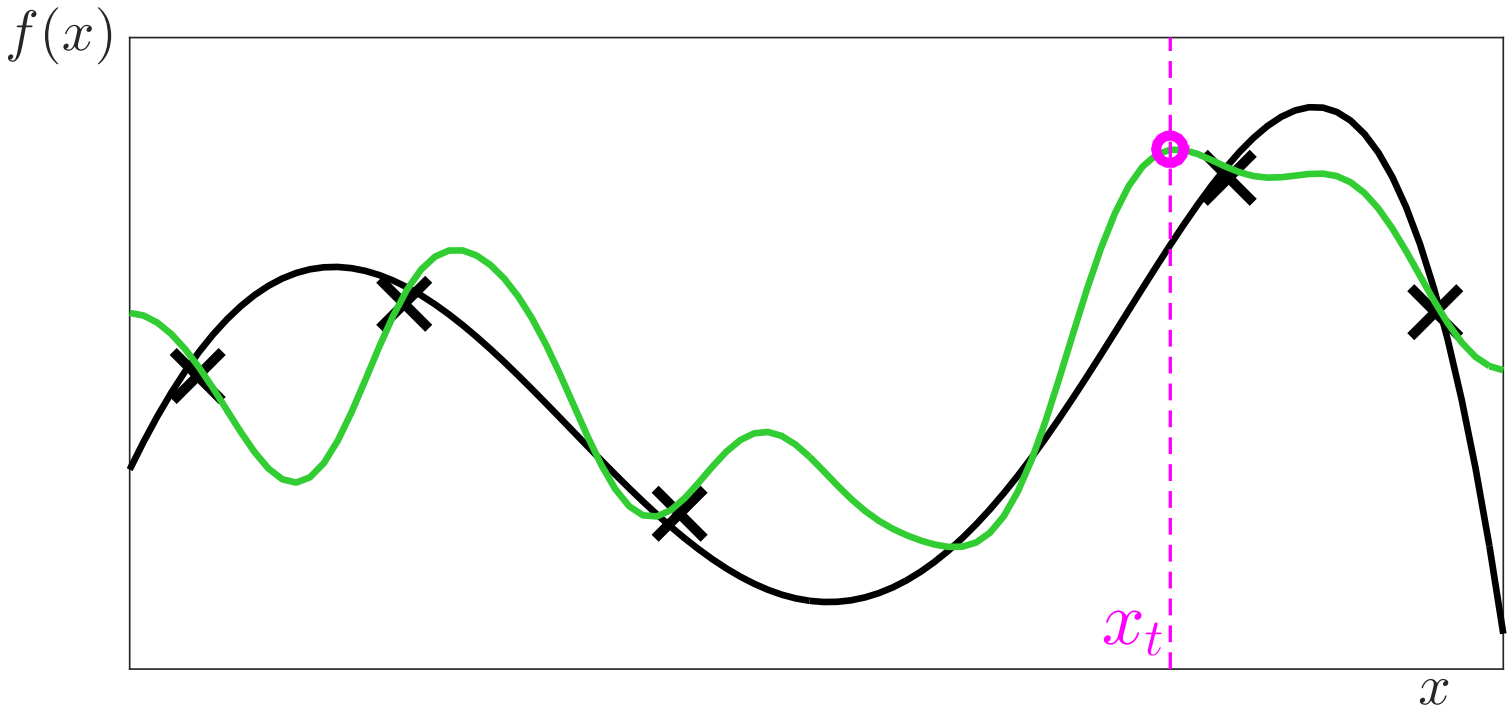
GP Optimisation with Thompson Sampling (Thompson, 1933)



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

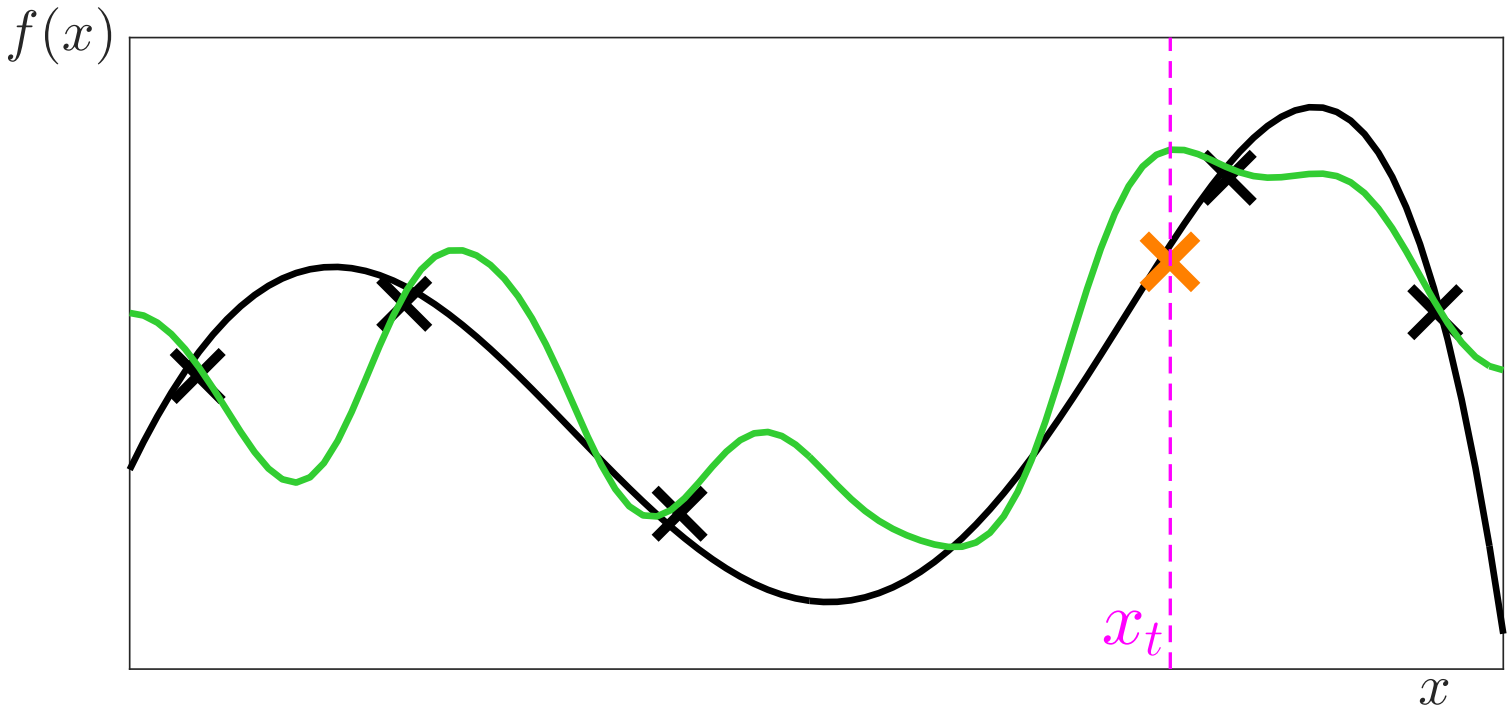
2) Draw sample g from posterior.

GP Optimisation with Thompson Sampling (Thompson, 1933)



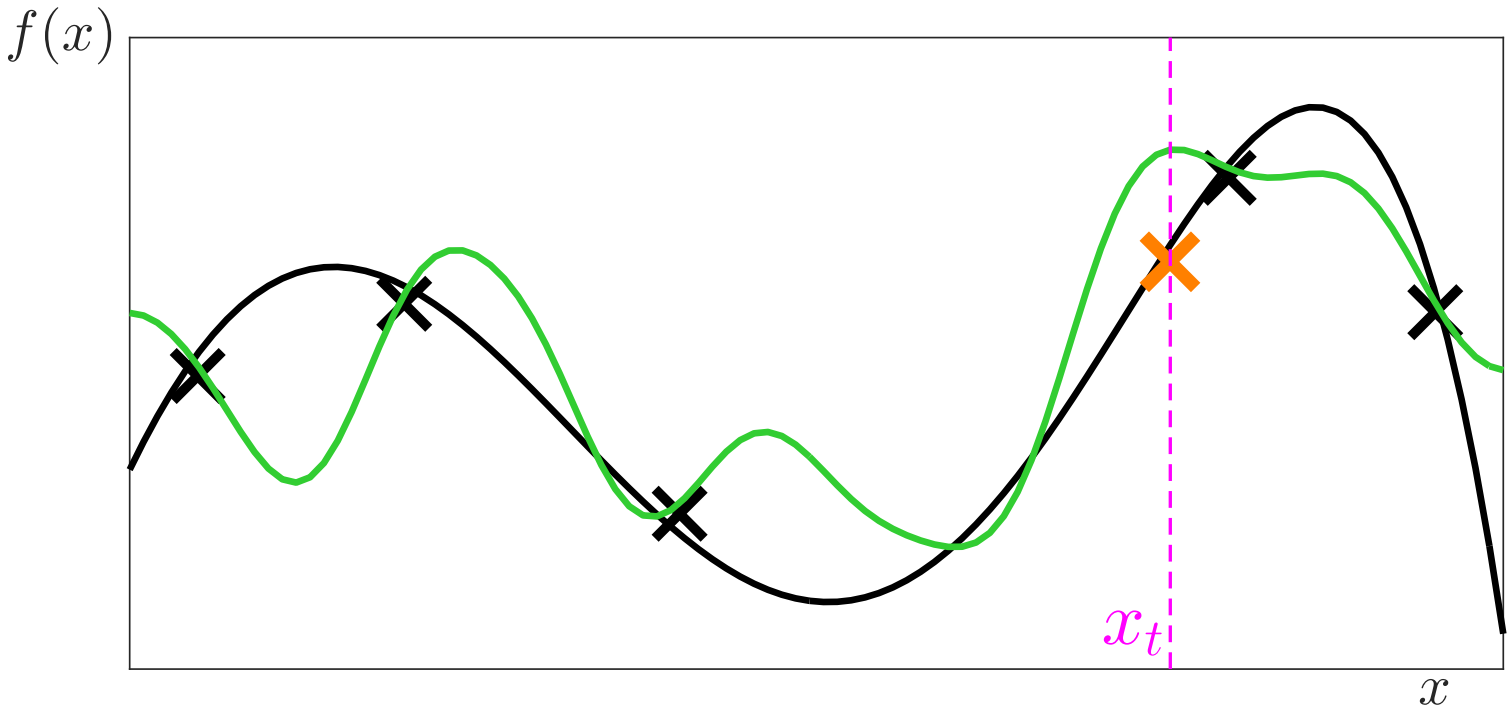
1) Construct posterior $\mathcal{GP}$.
2) Draw sample g from posterior.
3) Choose $x_t = \mathrm{argmax}_x\, g(x)$.

GP Optimisation with Thompson Sampling (Thompson, 1933)



1) Construct posterior $\mathcal{GP}$.
2) Draw sample g from posterior.
3) Choose $x_t = \mathrm{argmax}_x\, g(x)$.
4) Evaluate f at x_t.

GP Optimisation with Thompson Sampling (Thompson, 1933)



1) Construct posterior $\mathcal{GP}$.
2) Draw sample g from posterior.
3) Choose $x_t = \mathrm{argmax}_x\, g(x)$.
4) Evaluate f at x_t.

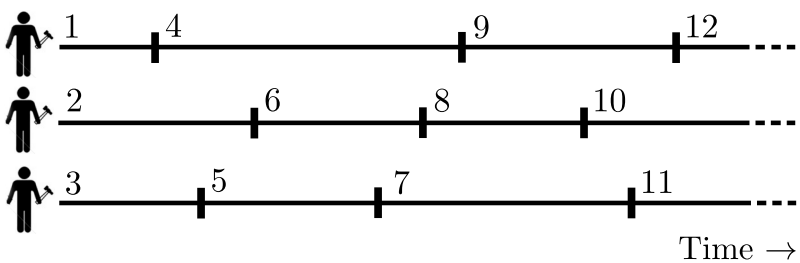
Take-home message: In parallel settings, direct application of sequential TS algorithm works. Inherent randomness adds sufficient diversity when managing M workers.

Parallelised Thompson Sampling

Asynchronous: asyTS

At any given time,

1. $(x', y') \leftarrow$ Wait for a worker to finish.
2. Compute posterior $\mathcal{GP}$.
3. Draw a sample $g \sim \mathcal{GP}$.
4. Re-deploy worker at $\operatorname{argmax} g$.

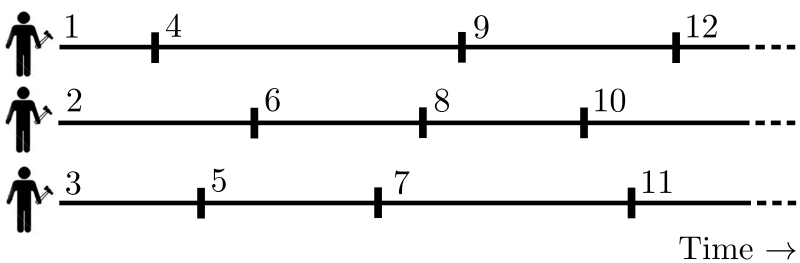


Parallelised Thompson Sampling

Asynchronous: asyTS

At any given time,

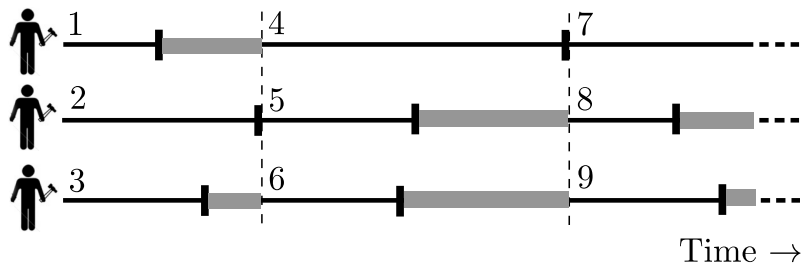
1. $(x', y') \leftarrow$ Wait for a worker to finish.
2. Compute posterior $\mathcal{GP}$.
3. Draw a sample $g \sim \mathcal{GP}$.
4. Re-deploy worker at $\operatorname{argmax} g$.



Synchronous: synTS

At any given time,

1. $\{(x'_m, y'_m)\}_{m=1}^{M} \leftarrow$ Wait for all workers to finish.
2. Compute posterior $\mathcal{GP}$.
3. Draw M samples $g_m \sim \mathcal{GP}$, $\forall m$.
4. Re-deploy worker m at $\operatorname{argmax} g_m$, $\forall m$.

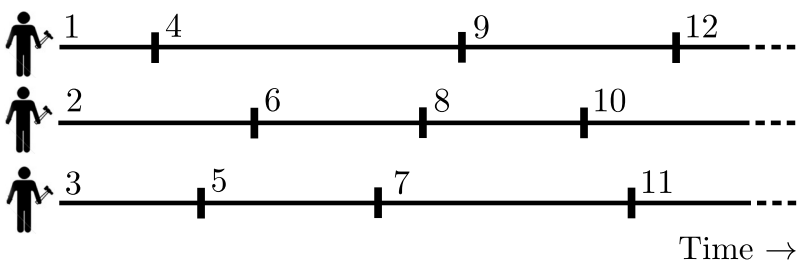


Parallelised Thompson Sampling

Asynchronous: asyTS

At any given time,

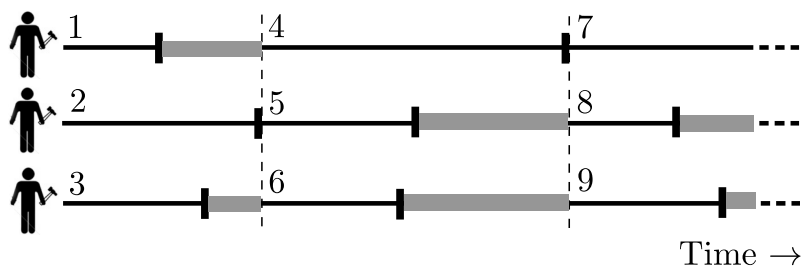
1. $(x', y') \leftarrow$ Wait for a worker to finish.
2. Compute posterior $\mathcal{GP}$.
3. Draw a sample $g \sim \mathcal{GP}$.
4. Re-deploy worker at $\operatorname{argmax} g$.



Synchronous: synTS

At any given time,

1. $\{(x'_m, y'_m)\}_{m=1}^{M} \leftarrow$ Wait for all workers to finish.
2. Compute posterior $\mathcal{GP}$.
3. Draw M samples $g_m \sim \mathcal{GP}$, $\forall m$.
4. Re-deploy worker m at $\operatorname{argmax} g_m$, $\forall m$.



Parallel TS in prior work: (Osband et al. 2016, Israelsen et al. 2016, Hernandez-Lobato et al. 2017)

Simple Regret in Parallel Settings

Simple regret after n evaluations,

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

$n \leftarrow \#$ completed evaluations by all workers.

Simple Regret in Parallel Settings

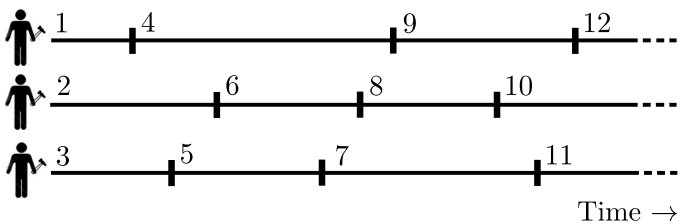
Simple regret after n evaluations,

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

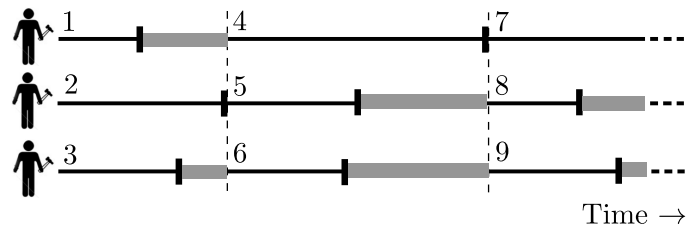
$n \leftarrow$ # completed evaluations by all workers.

Simple regret with time as a resource,

Asynchronous



Synchronous



$$\mathrm{SR}'(T) = f(x_\star) \; - \max_{t=1,\ldots,N} f(x_t).$$

$N \leftarrow$ # completed evaluations by all workers in time T.
(possibly random).

Theoretical Results SR(n)

Several results for sequential Thompson sampling (Agrawal et al. 2012, Kaufmann et al. 2012, Russo & van Roy 2016)

Theoretical Results SR(n)

Several results for sequential Thompson sampling (Agrawal et al. 2012, Kaufmann et al. 2012, Russo & van Roy 2016)

seqTS

$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \sqrt{\frac{\Psi_n \log(n)}{n}}$$

(Russo & van Roy 2014)

$\Psi_n \leftarrow$ Maximum information gain (Srinivas et al. 2010)

GP with SE Kernel in d dimensions, $\Psi_n(\mathcal{X}) \asymp d^d \log(n)^d$.

Theoretical Results SR(n)

Several results for sequential Thompson sampling (Agrawal et al. 2012, Kaufmann et al. 2012, Russo & van Roy 2016)

seqTS

$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \sqrt{\frac{\Psi_n \log(n)}{n}}$$

(Russo & van Roy 2014)

$\Psi_n \leftarrow$ Maximum information gain (Srinivas et al. 2010)

GP with SE Kernel in d dimensions, $\Psi_n(\mathcal{X}) \asymp d^d \log(n)^d$.

Theorem: synTS (Kandasamy et al. 2018)

$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \frac{M\sqrt{\log(M)}}{n} + \sqrt{\frac{\Psi_n \log(n+M)}{n}}$$

Theoretical Results SR(n)

Several results for sequential Thompson sampling (Agrawal et al. 2012, Kaufmann et al. 2012, Russo & van Roy 2016)

seqTS

$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \sqrt{\frac{\Psi_n \log(n)}{n}}$$

(Russo & van Roy 2014)

$\Psi_n \leftarrow$ Maximum information gain (Srinivas et al. 2010)

GP with SE Kernel in d dimensions, $\Psi_n(\mathcal{X}) \asymp d^d \log(n)^d$.

Theorem: synTS (Kandasamy et al. 2018)

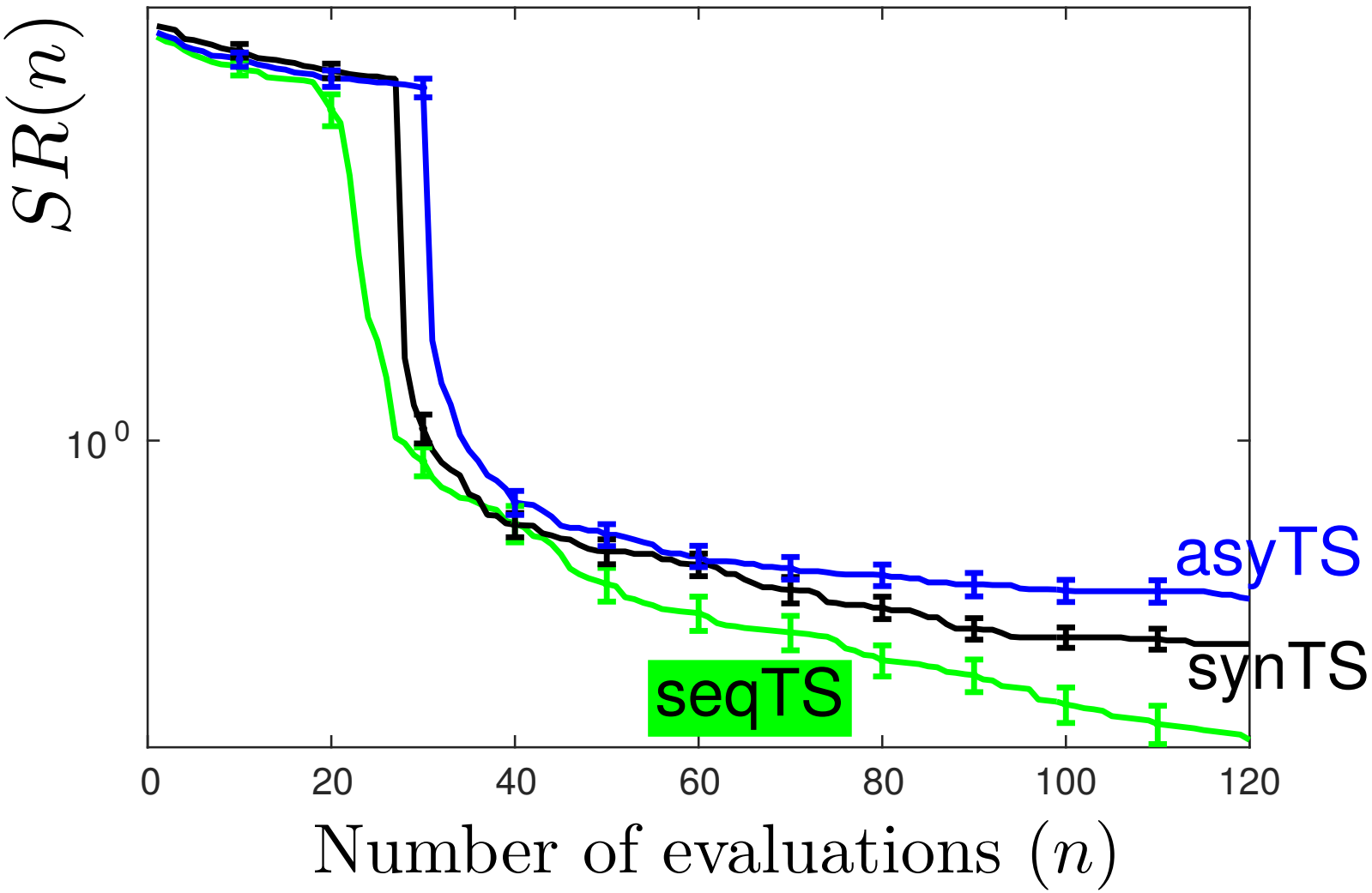
$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \frac{M\sqrt{\log(M)}}{n} + \sqrt{\frac{\Psi_n \log(n+M)}{n}}$$

Theorem: asyTS (Kandasamy et al. 2018)

$$\mathbb{E}[\mathrm{SR}(n)] \lesssim \frac{M\mathrm{polylog}(M)}{n} + \sqrt{\frac{C\Psi_n \log(n)}{n}}$$

Experiment: Park1-4D $M = 10$

Comparison in terms of number of evaluations



Theoretical Results for $\mathrm{SR}'(T)$

Model evaluation time as an independent random variable

- Uniform $\quad \mathrm{unif}(a, b)$ bounded
- Half-normal $\quad \mathcal{HN}(\tau^2)$ sub-Gaussian
- Exponential $\quad \exp(\lambda)$ sub-exponential

Theoretical Results for $\mathrm{SR}'(T)$

Model evaluation time as an independent random variable

- Uniform $\mathrm{unif}(a, b)$ — bounded
- Half-normal $\mathcal{HN}(\tau^2)$ — sub-Gaussian
- Exponential $\exp(\lambda)$ — sub-exponential

Theorem: TS with M parallel workers (Kandasamy et al. 2018)

If evaluation times are the same, synTS $\approx$ asyTS.
When there is high variability in evaluation times, asyTS is much better than synTS.

Theoretical Results for $\mathrm{SR}'(T)$

Model evaluation time as an independent random variable

- Uniform $\quad \mathrm{unif}(a, b)$ bounded
- Half-normal $\quad \mathcal{HN}(\tau^2)$ sub-Gaussian
- Exponential $\quad \exp(\lambda)$ sub-exponential

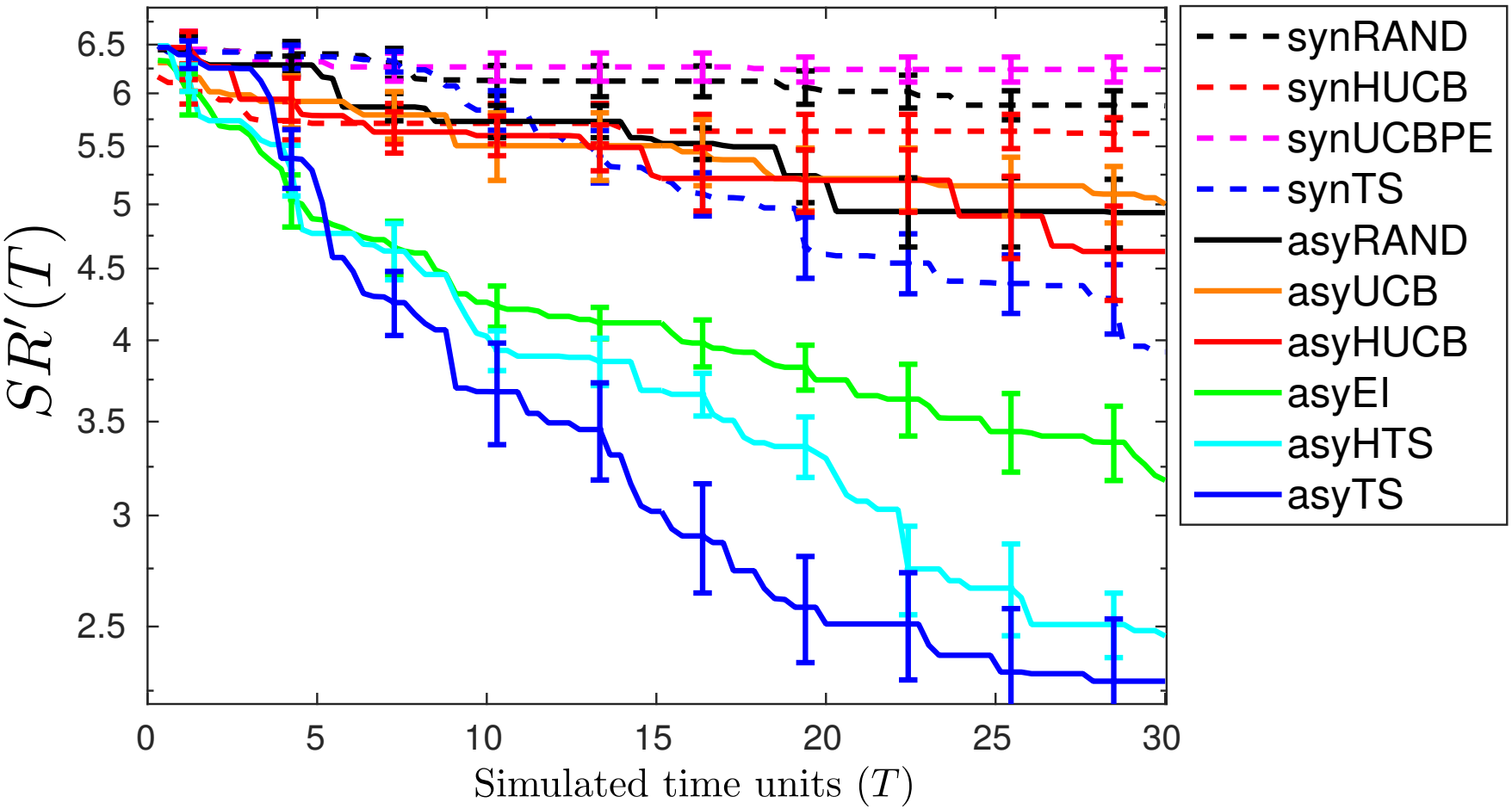
Theorem: TS with M parallel workers (Kandasamy et al. 2018)

If evaluation times are the same, synTS $\approx$ asyTS.
When there is high variability in evaluation times, asyTS is much better than synTS.

- Uniform: constant factor
- Half-normal: $\sqrt{\log(M)}$ factor
- Exponential: $\log(M)$ factor

Experiment: Hartmann-18D $M = 25$

Evaluation time sampled from an exponential distribution



Additional synthetic and real experiments in the paper/poster.

Summary

- synTS, asyTS: direct application of TS to synchronous and asynchronous parallel settings.
- Take-aways: Theory
  - Both perform essentially the same as seqTS in terms of the number of evaluations.
  - When we factor time as a resource, asyTS performs best.
- Take-aways: Practice
  - Conceptually simple and scales better with the number of workers than other methods.

Summary

- synTS, asyTS: direct application of TS to synchronous and asynchronous parallel settings.
- Take-aways: Theory
  - Both perform essentially the same as seqTS in terms of the number of evaluations.
  - When we factor time as a resource, asyTS performs best.
- Take-aways: Practice
  - Conceptually simple and scales better with the number of workers than other methods.

Thank you

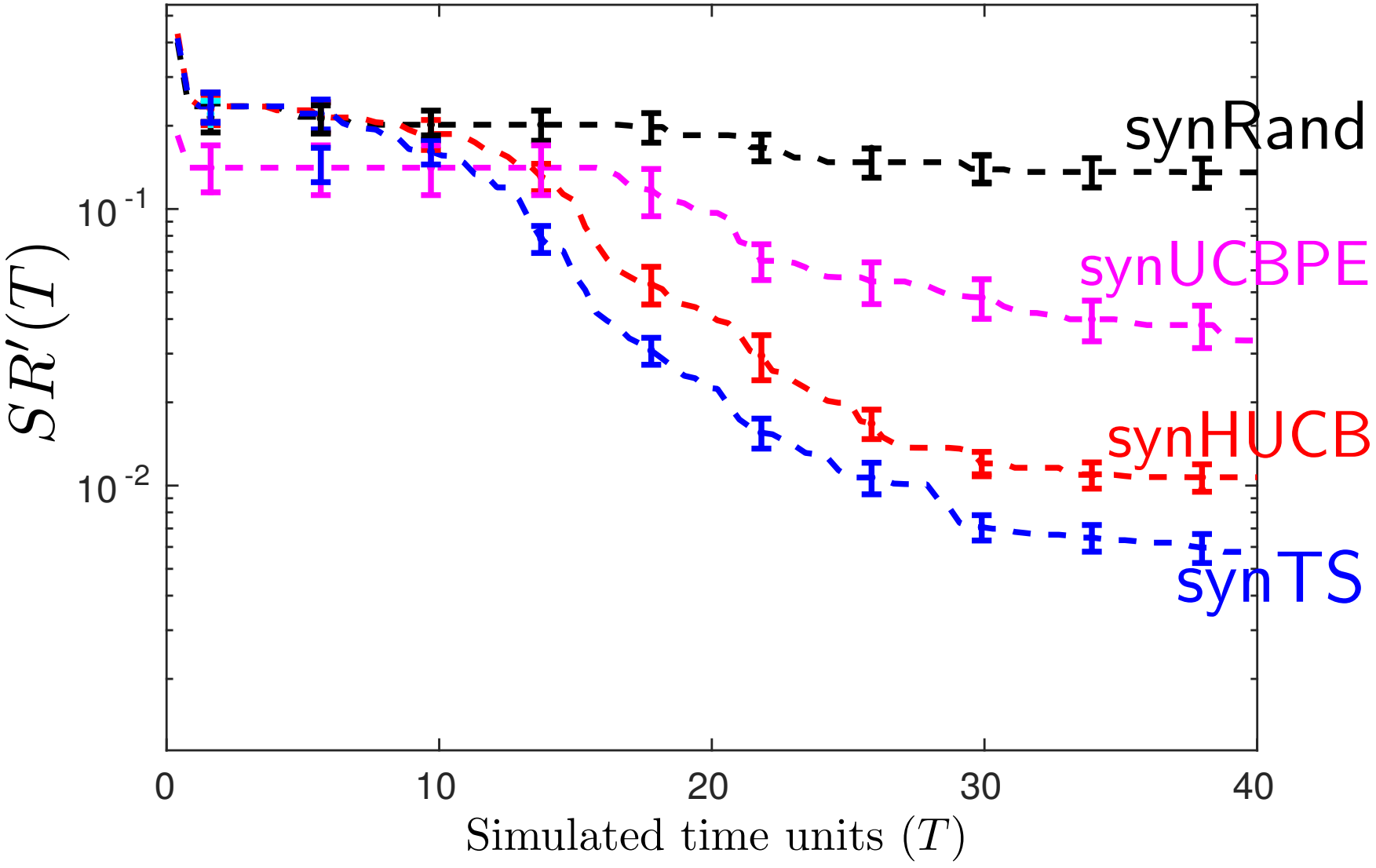
Poster #49, Session 3 (Tuesday evening).

Code: `github.com/kirthevasank/gp-parallel-ts`

Appendix

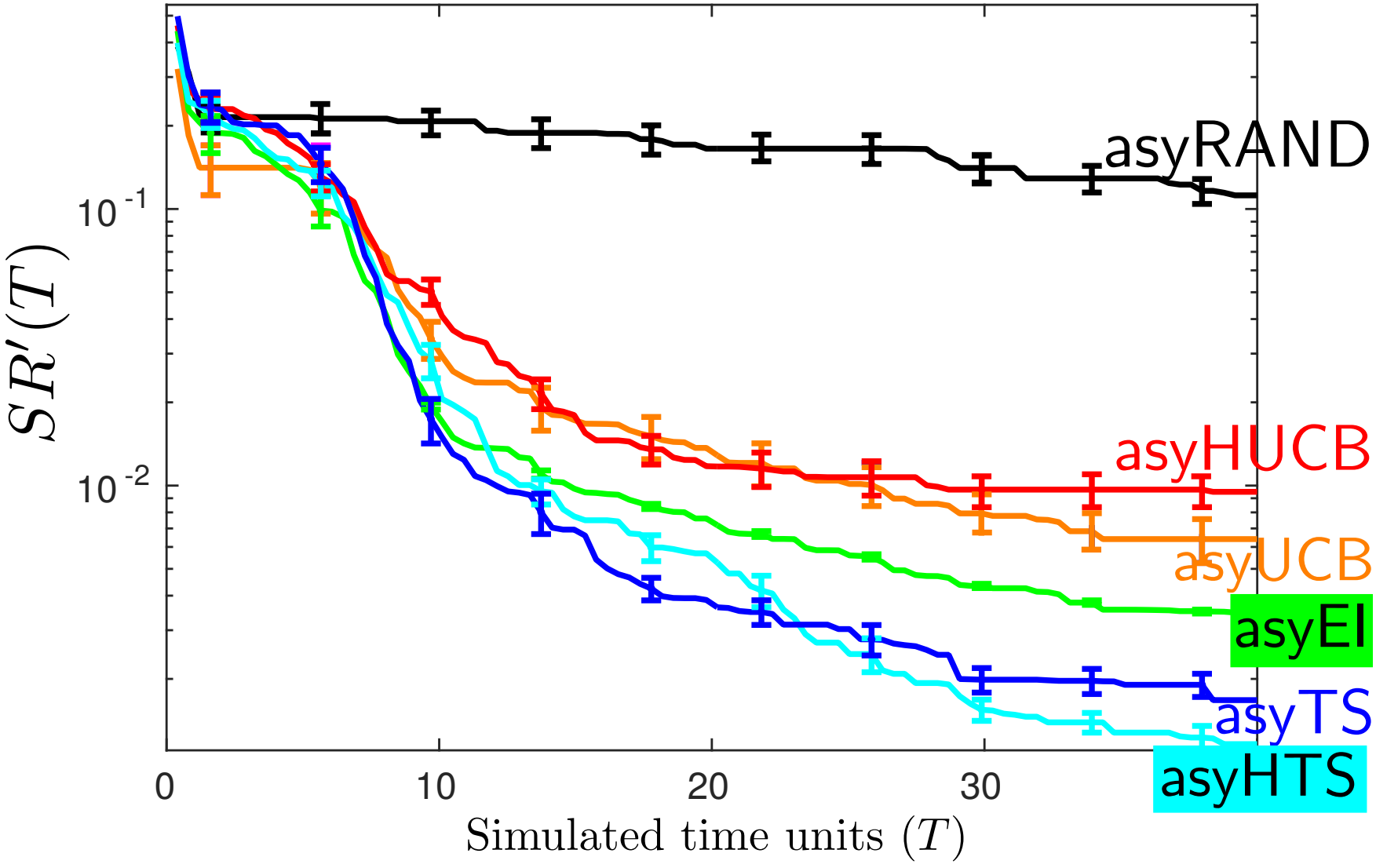
Experiment: Branin-2D $M = 4$

Evaluation time sampled from a uniform distribution



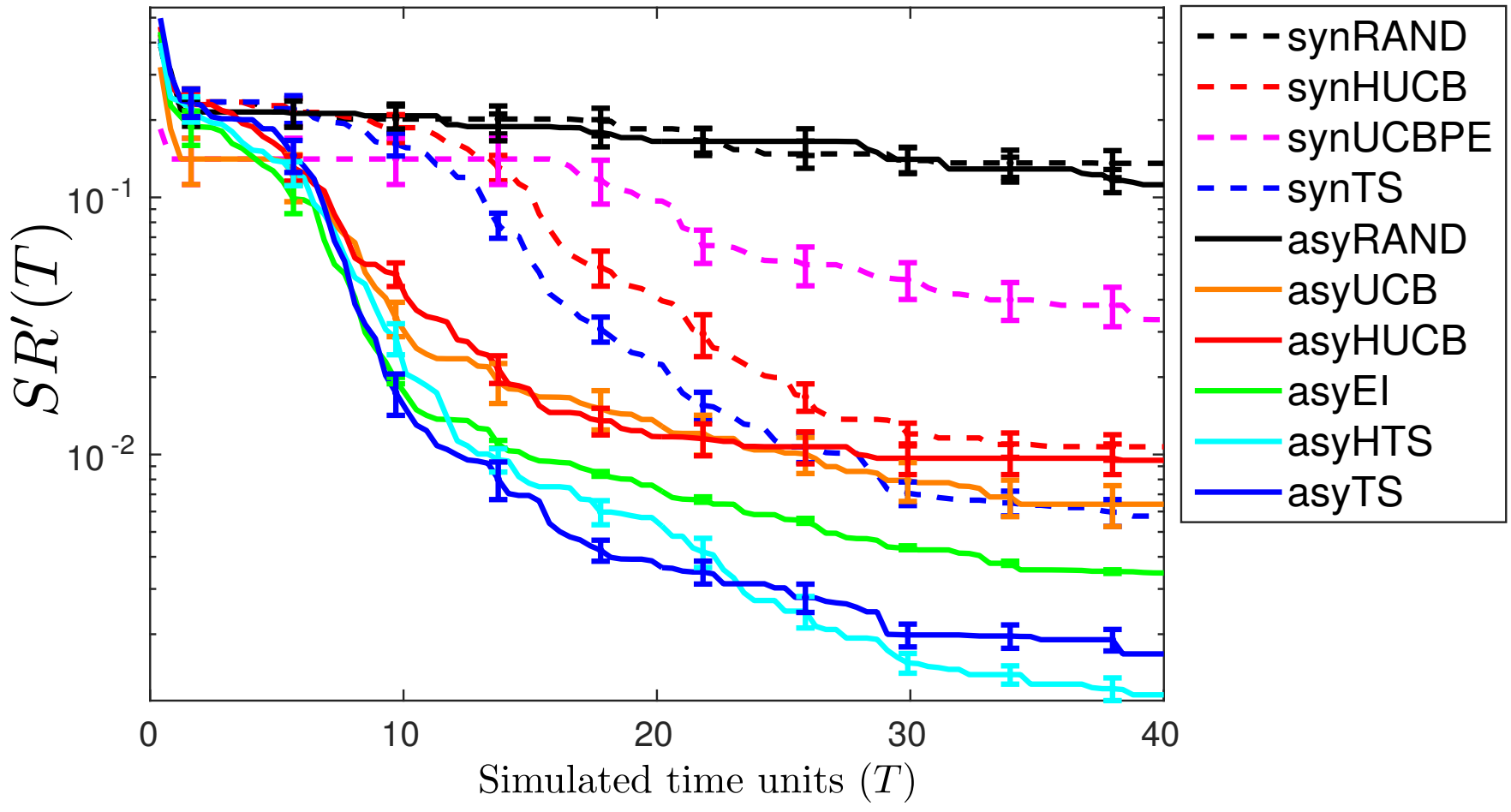
Experiment: Branin-2D $M = 4$

Evaluation time sampled from a uniform distribution



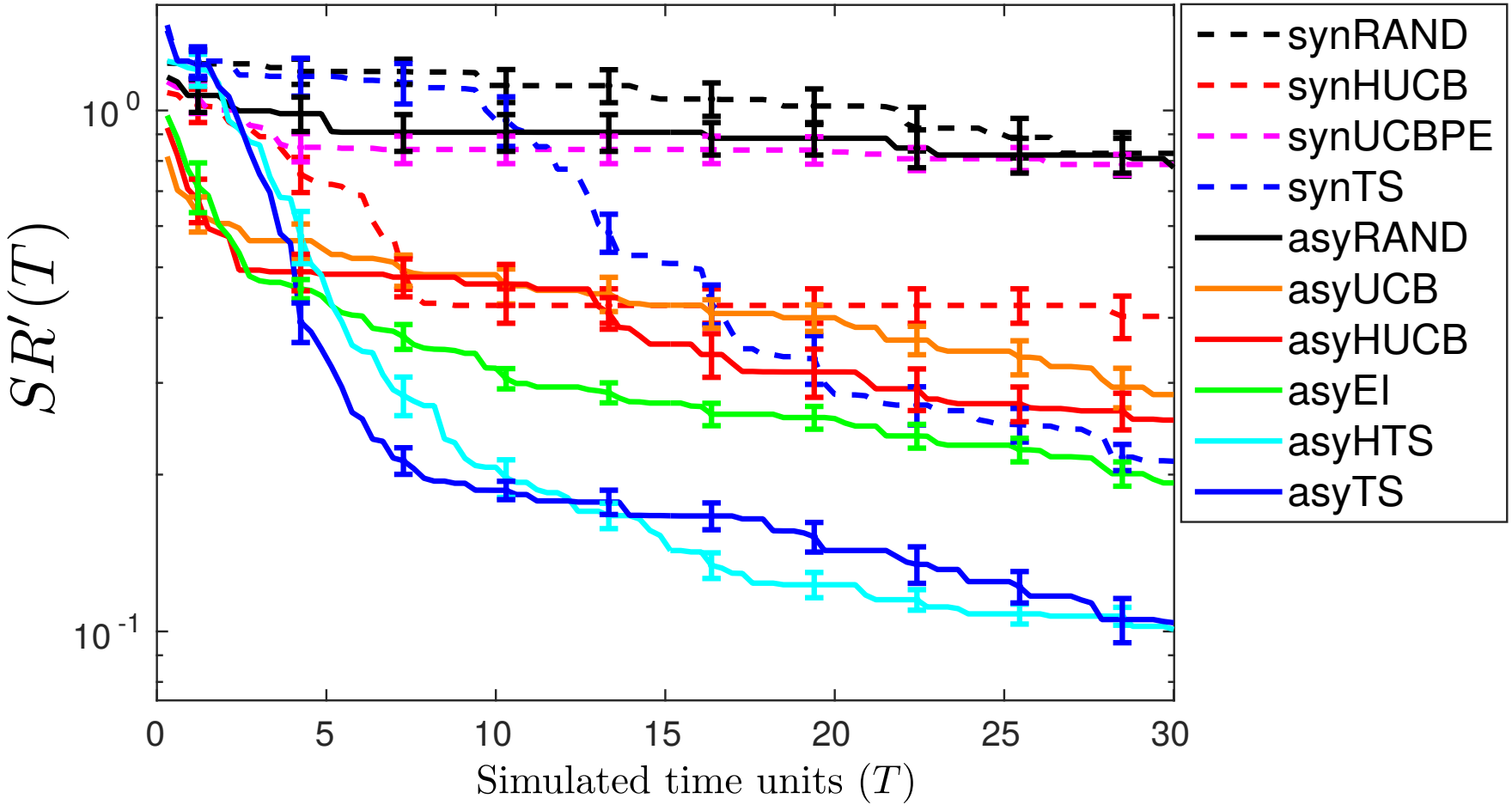
Experiment: Branin-2D $M = 4$

Evaluation time sampled from a uniform distribution



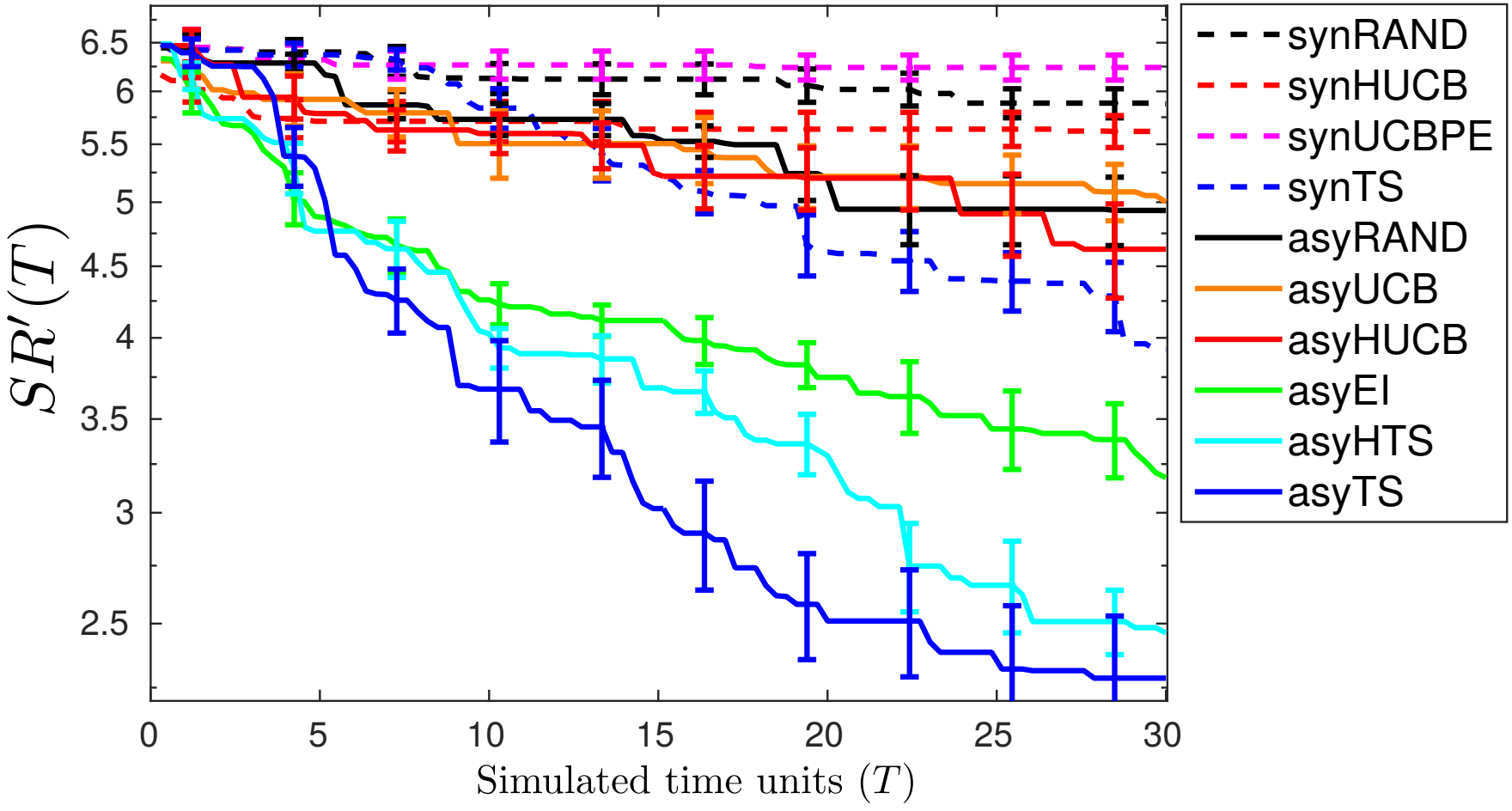
Experiment: Hartmann-6D $M = 12$

Evaluation time sampled from a half-normal distribution



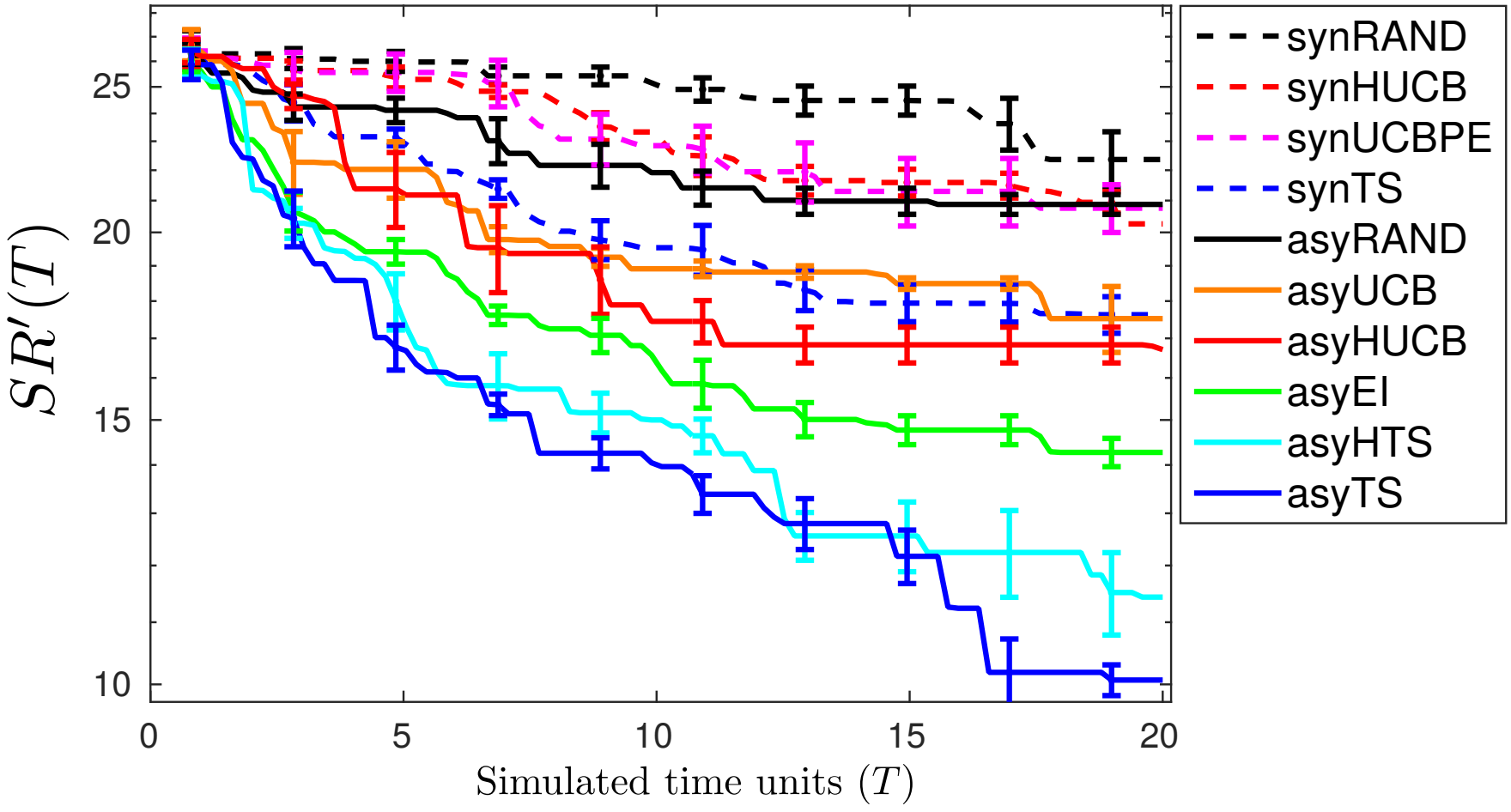
Experiment: Hartmann-18D $M = 25$

Evaluation time sampled from an exponential distribution



Experiment: Currin-Exponential-14D $M = 35$

Evaluation time sampled from a Pareto-3 distribution



Experiment: Model Selection in Cifar10 $M = 4$

Tune # filters in in range $(32, 256)$ for each layer in a 6 layer CNN.
Time taken for an evaluation: 4 - 16 minutes.