Parallel, Batch and Milti-objetive BO

Julián D. Arias Londoño

Dpto. de Señales, Sistemas y Radiocomunicaciones
E.T.S. Ingenieros de Telecomunicación
Universidad Politécnica de Madrid

2023

1. Feedback
   - Bayesian optimization
2. Parallel and batch BO
   - Thompson sampling
   - Multi-points Expected improvement
3. Multi-objective BO
   - Multi-objective acquisition function
   - Multi-output $\mathcal{GP}$s

BO optimization algorithm

In order to find:

$$\mathbf{x}_{max} = \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{1}$$

The Bayesian optimization procedure is as follows. For $t = 1, 2, \ldots$ repeat [2]:

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a possibly noisy sample $y_t = f(\mathbf{x}_t) + \varepsilon_t$ from the objective function f.
3. Add the sample to previous samples $\mathcal{D}_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t)\}$ and update the GP.

where $\mathcal{I}_{t-1}$ represents the available data set $\mathcal{D}_{1:t-1}$ and the GP structure (kernel, likelihood and parameter values) at $t-1$ step.

BO optimization algorithm

There are many different aspects to expand the study of BO as a field, some of they are:

- How to speed up the optimization process since standard BO provides only one single candidate solution at a time.

BO optimization algorithm

There are many different aspects to expand the study of BO as a field, some of they are:

- How to speed up the optimization process since standard BO provides only one single candidate solution at a time.
- How to address multi-objective problems

BO optimization algorithm

There are many different aspects to expand the study of BO as a field, some of they are:

- How to speed up the optimization process since standard BO provides only one single candidate solution at a time.
- How to address multi-objective problems
- How to deal with restrictions: known and **unknown!**

BO optimization algorithm

There are many different aspects to expand the study of BO as a field, some of they are:

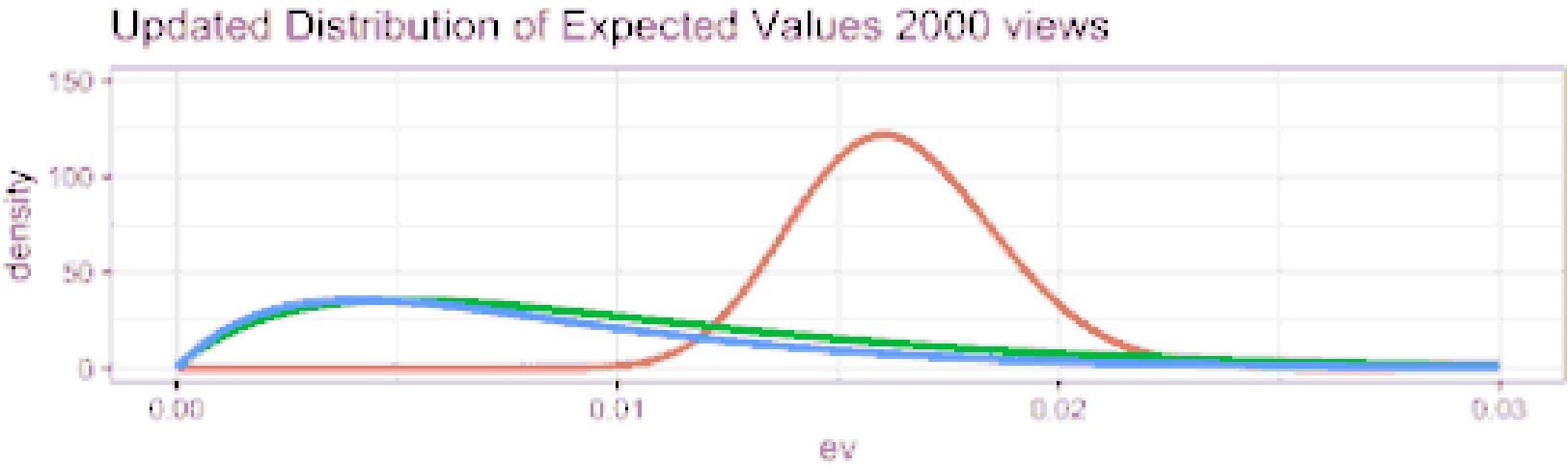
- How to speed up the optimization process since standard BO provides only one single candidate solution at a time.
- How to address multi-objective problems
- How to deal with restrictions: known and **unknown!**
- There are several Acquisition functions to review with interesting properties

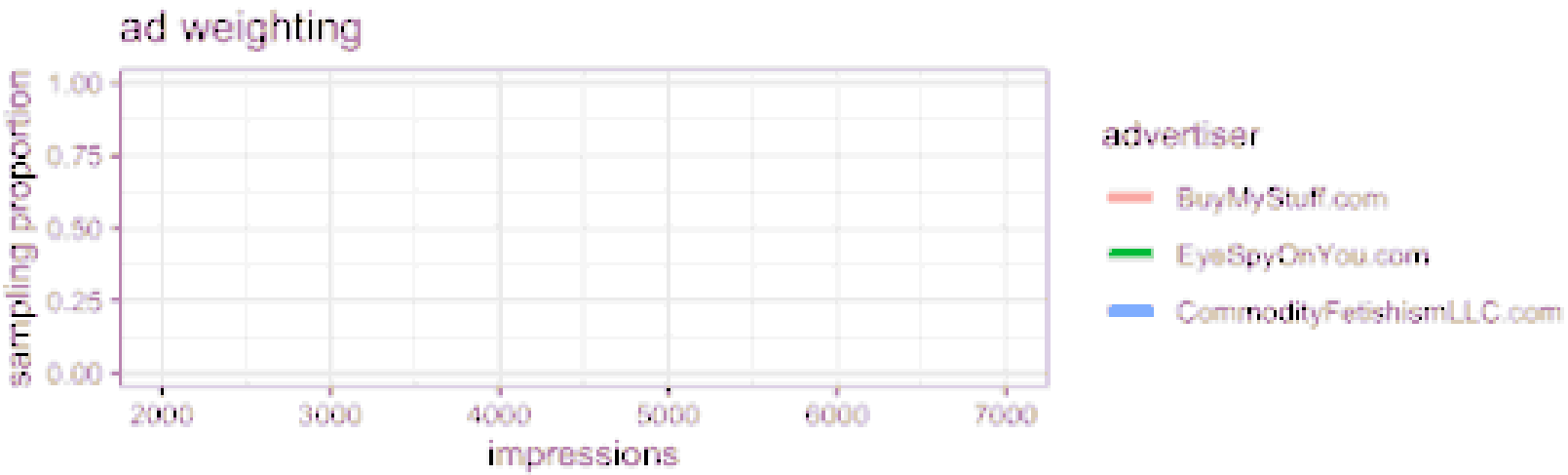
BO optimization algorithm

There are many different aspects to expand the study of BO as a field, some of they are:

- How to speed up the optimization process since standard BO provides only one single candidate solution at a time.
- How to address multi-objective problems
- How to deal with restrictions: known and **unknown!**
- There are several Acquisition functions to review with interesting properties
- Composite, multi-fidelity problems, and many more.

Thompson sampling





GP posterior distribution

Let's remember the conditional distribution $\mathbf{f}^*|\mathbf{X}, \mathbf{y}, \mathbf{X}^*$ when we assume a prior GP [6]:

$$\begin{aligned} \bar{\mathbf{f}^*} &= K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}\mathbf{y} \qquad (2) \\ \text{cov}(\mathbf{f}^*) &= K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}K(\mathbf{X}, \mathbf{X}^*) \end{aligned}$$

GP posterior distribution

Let's remember the conditional distribution $\mathbf{f}^*|\mathbf{X}, \mathbf{y}, \mathbf{X}^*$ when we assume a prior GP [6]:

$$\begin{aligned} \bar{\mathbf{f}^*} &= K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}\mathbf{y} \\ \text{cov}(\mathbf{f}^*) &= K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}K(\mathbf{X}, \mathbf{X}^*) \end{aligned} \tag{2}$$

This is the posterior distribution of the functions given the training data and a set of new sample points $\mathbf{X}^*$.

$$\mathbf{f}^*|\mathbf{X}, \mathbf{y}, \mathbf{X}^* \sim \mathcal{N}\left(\bar{\mathbf{f}^*}, \text{cov}(\mathbf{f}^*)\right) \tag{3}$$

GP posterior distribution

Let's remember the conditional distribution $\mathbf{f}^*|\mathbf{X}, \mathbf{y}, \mathbf{X}^*$ when we assume a prior GP [6]:

$$\begin{aligned} \bar{\mathbf{f}}^* &= K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}\mathbf{y} \\ \text{cov}(\mathbf{f}^*) &= K(\mathbf{X}^*, \mathbf{X}^*) - K(\mathbf{X}^*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma^2\mathbf{I}]^{-1}K(\mathbf{X}, \mathbf{X}^*) \end{aligned} \tag{2}$$

This is the posterior distribution of the functions given the training data and a set of new sample points $\mathbf{X}^*$.

$$\mathbf{f}^*|\mathbf{X}, \mathbf{y}, \mathbf{X}^* \sim \mathcal{N}\left(\bar{\mathbf{f}}^*, \text{cov}(\mathbf{f}^*)\right) \tag{3}$$

So, we can sample from it and the optimum corresponds to TS! selection.

BO using TS

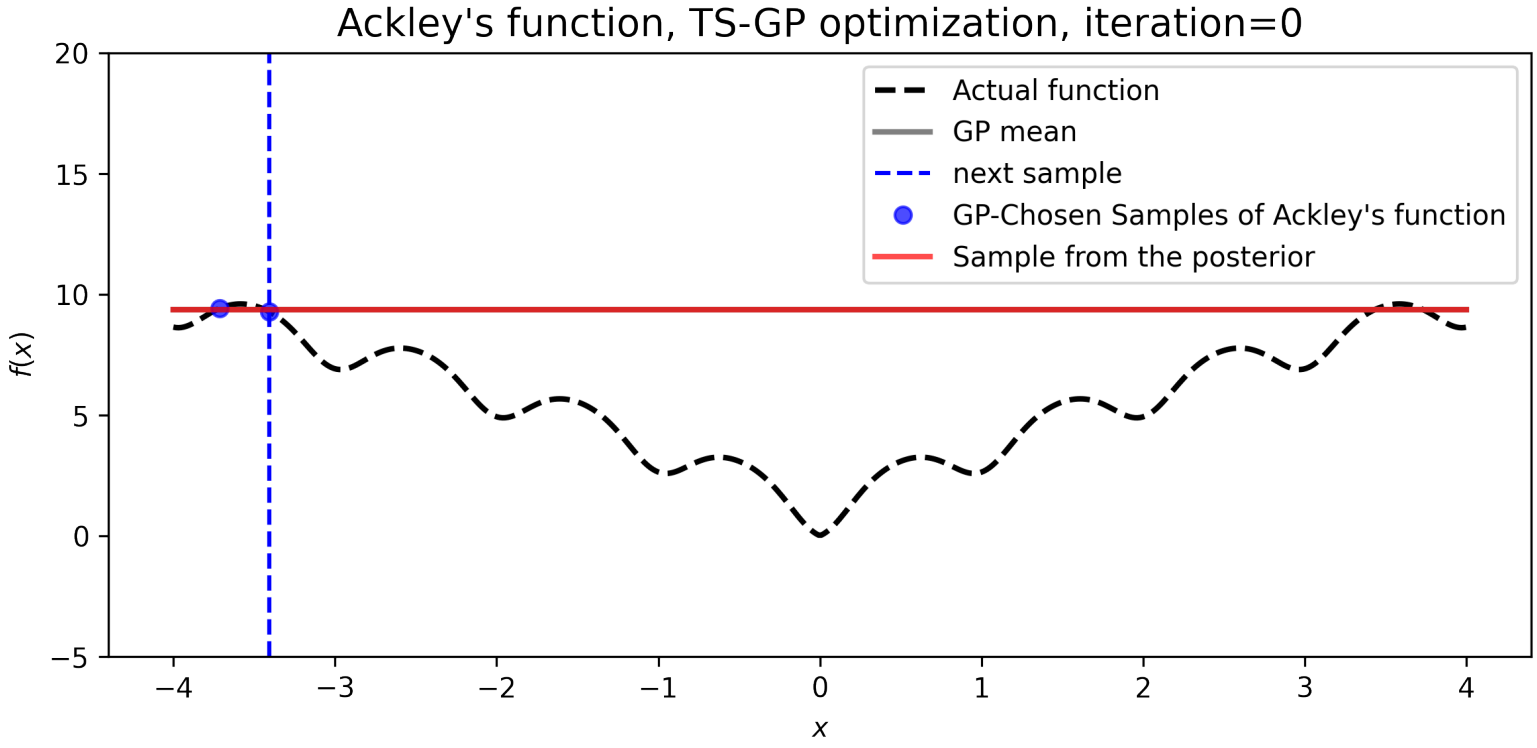


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

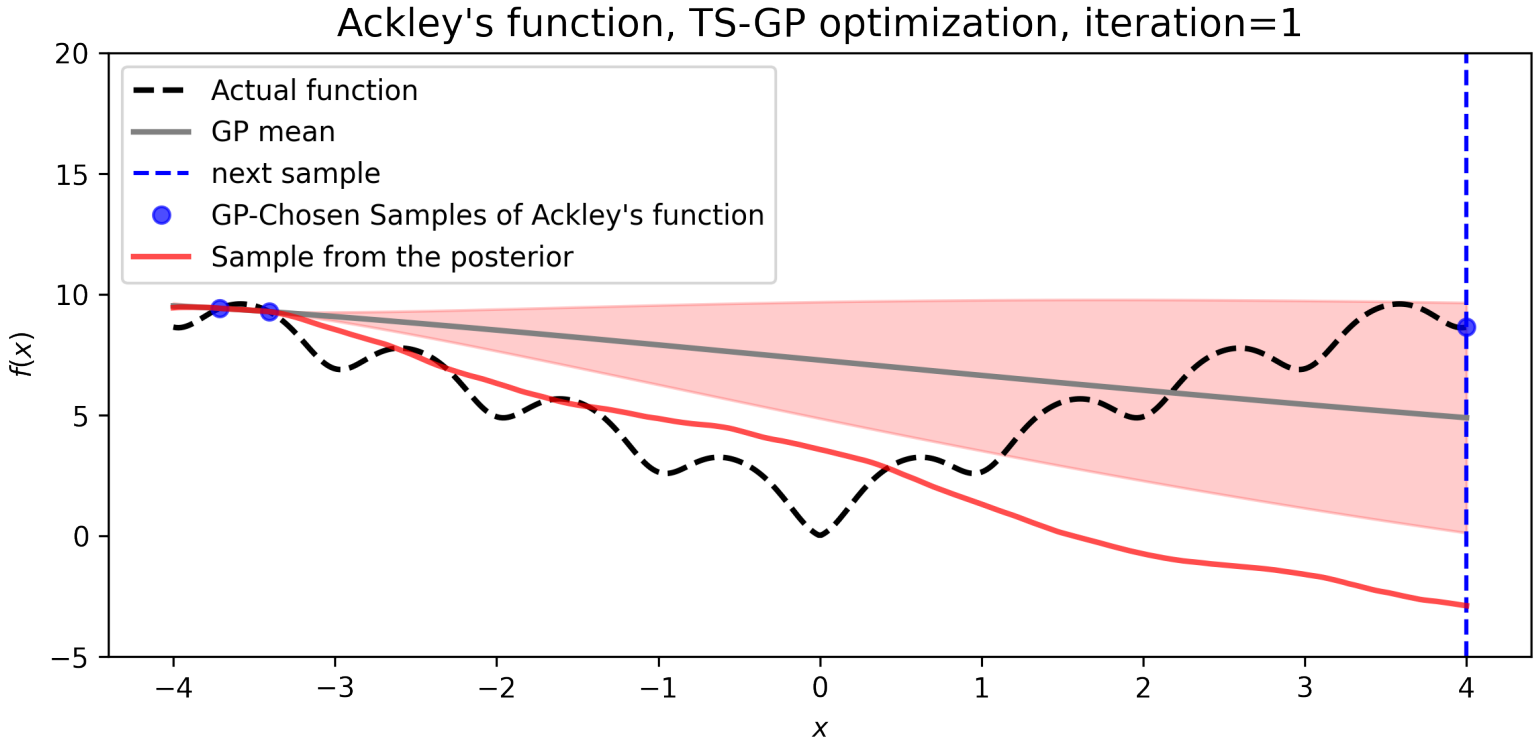


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

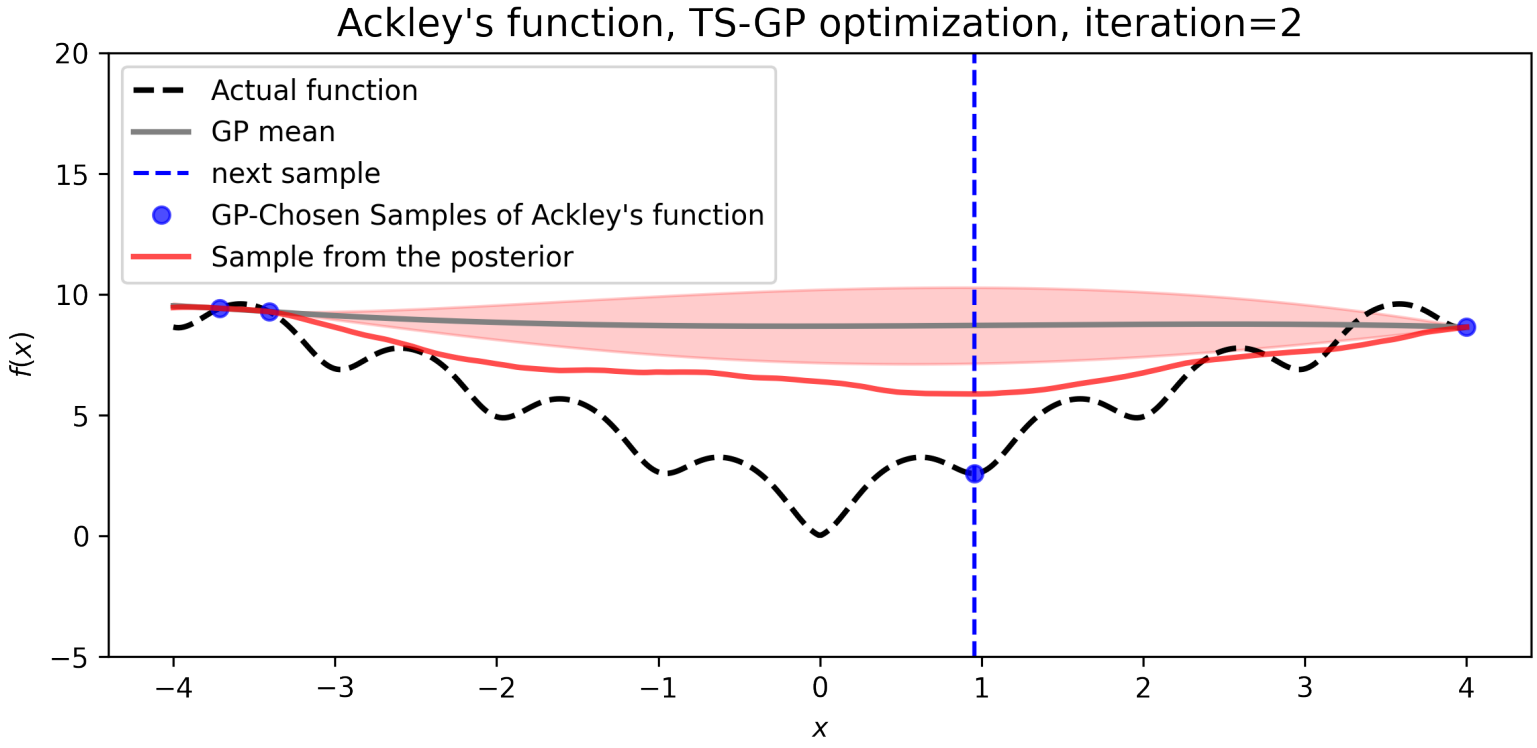


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

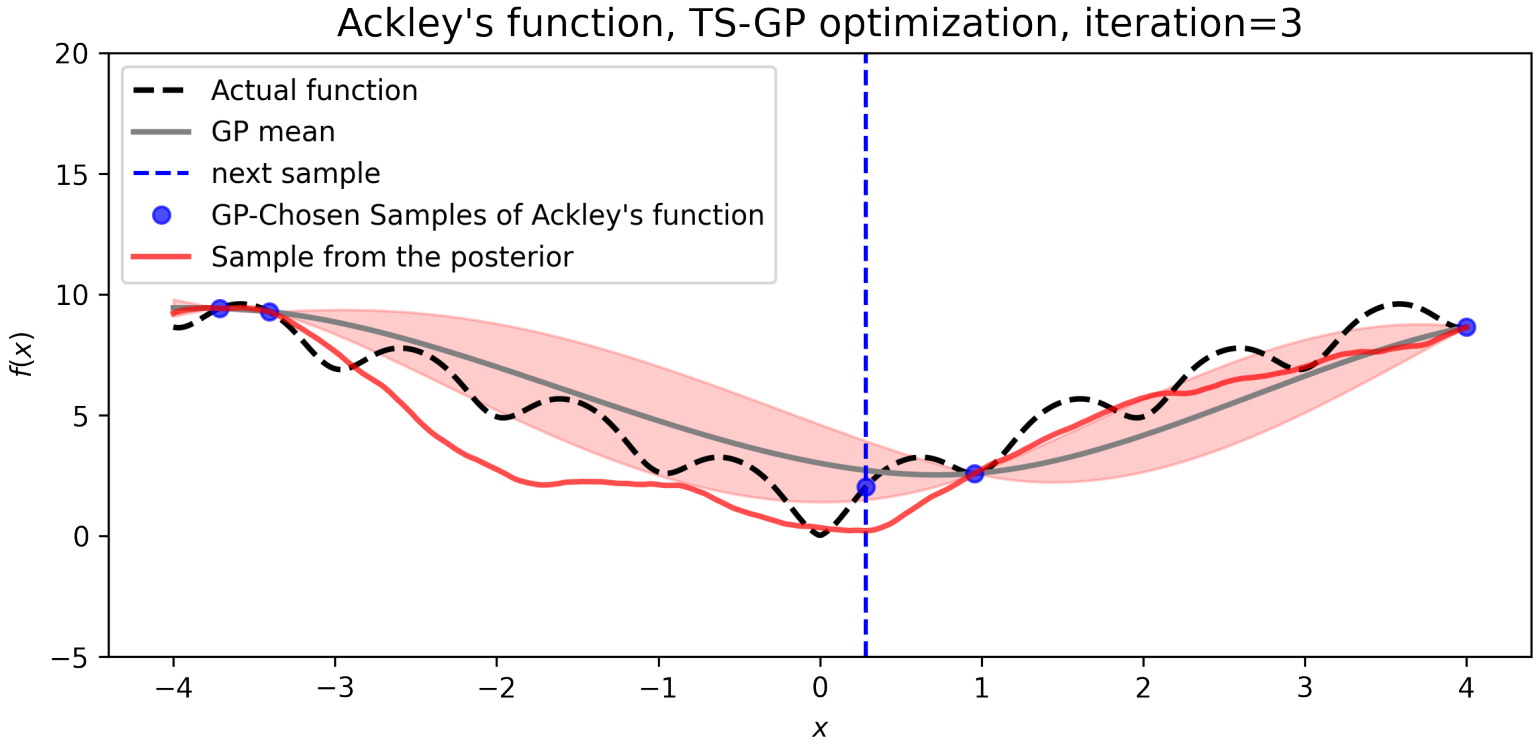


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

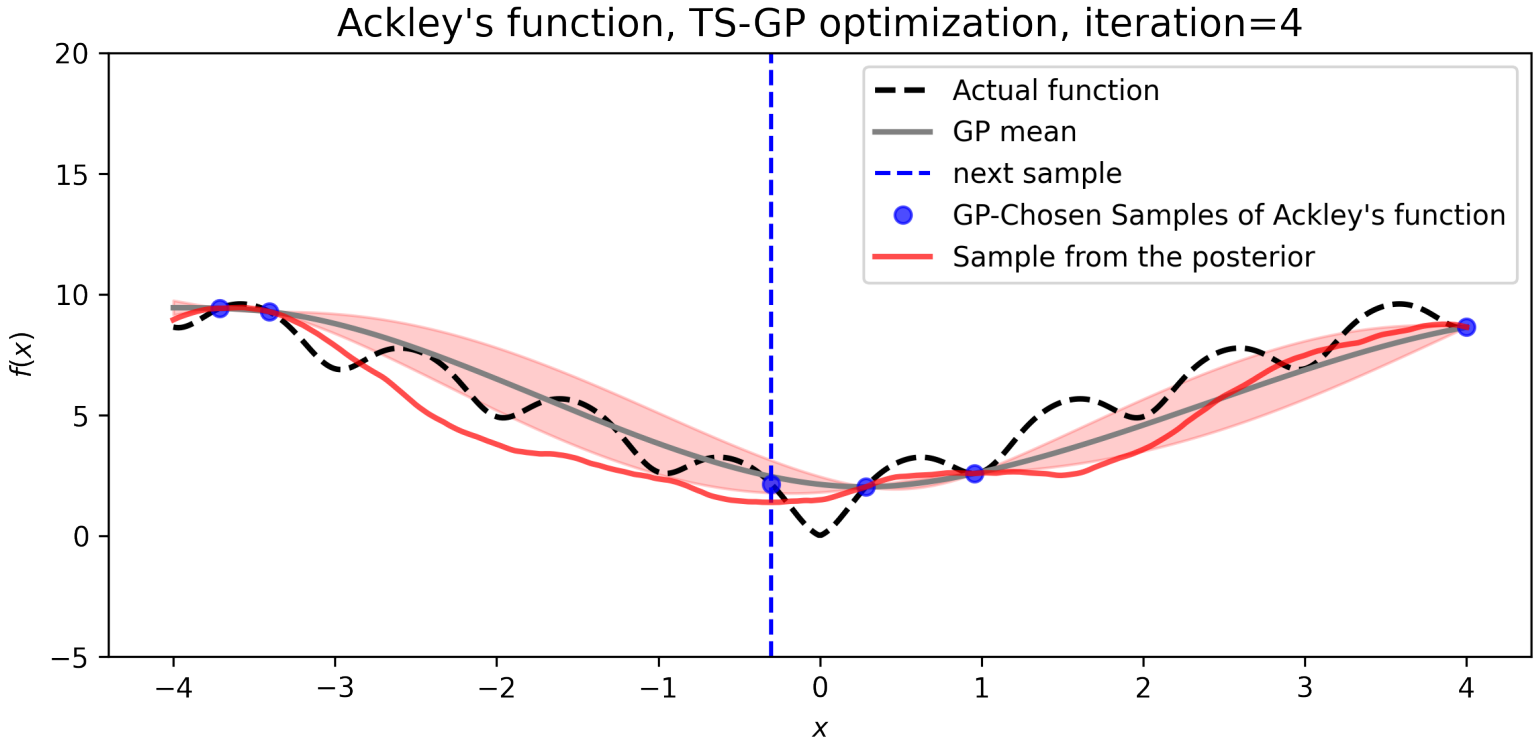


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

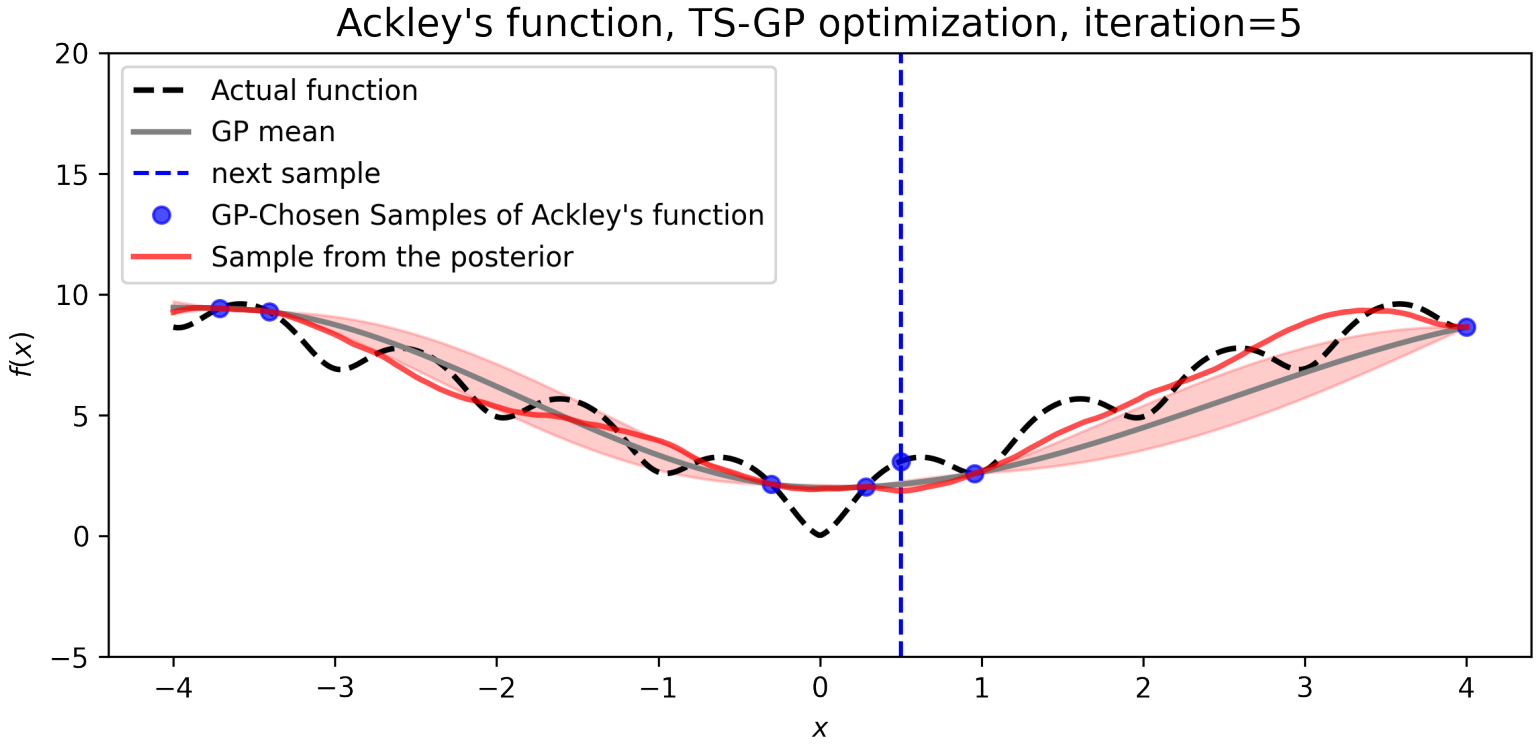


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

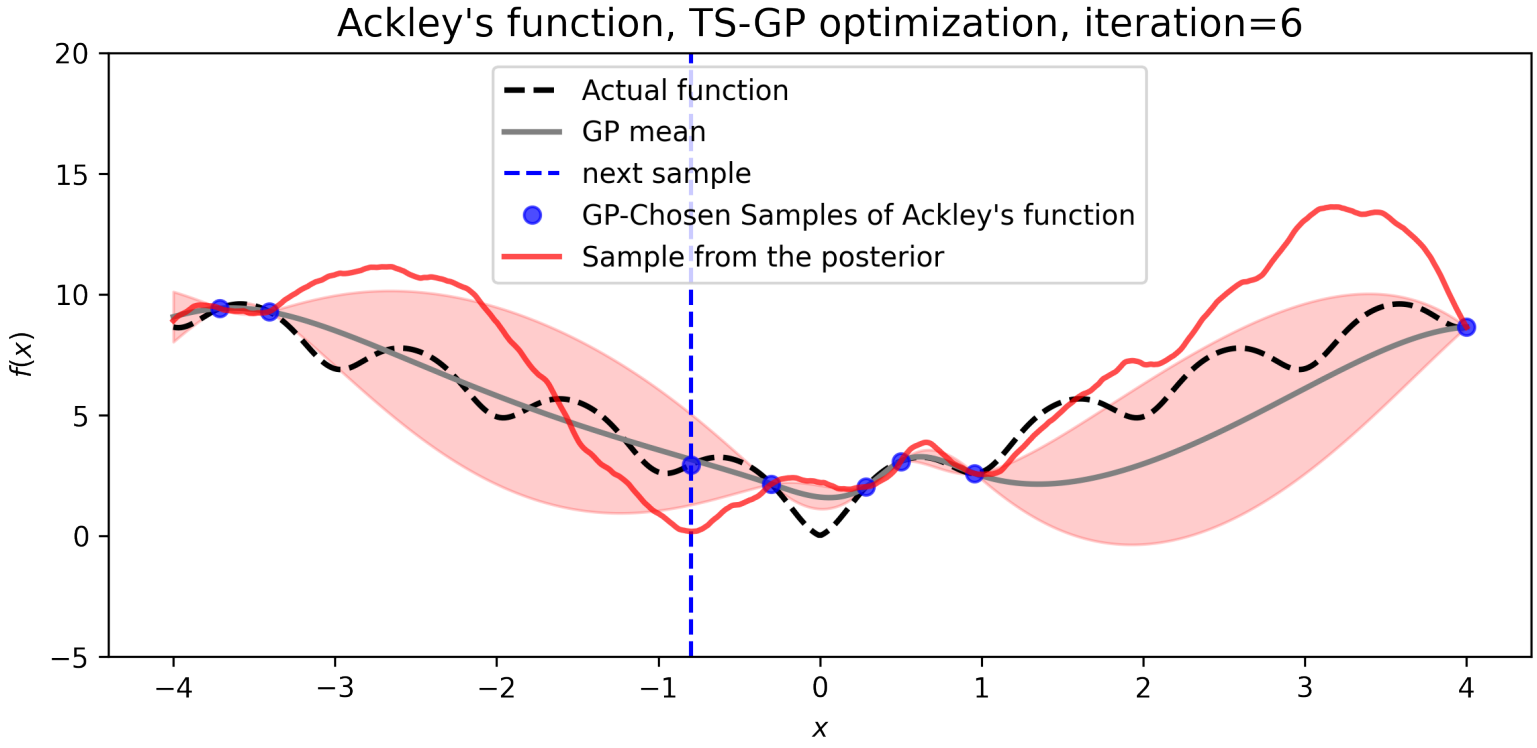


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

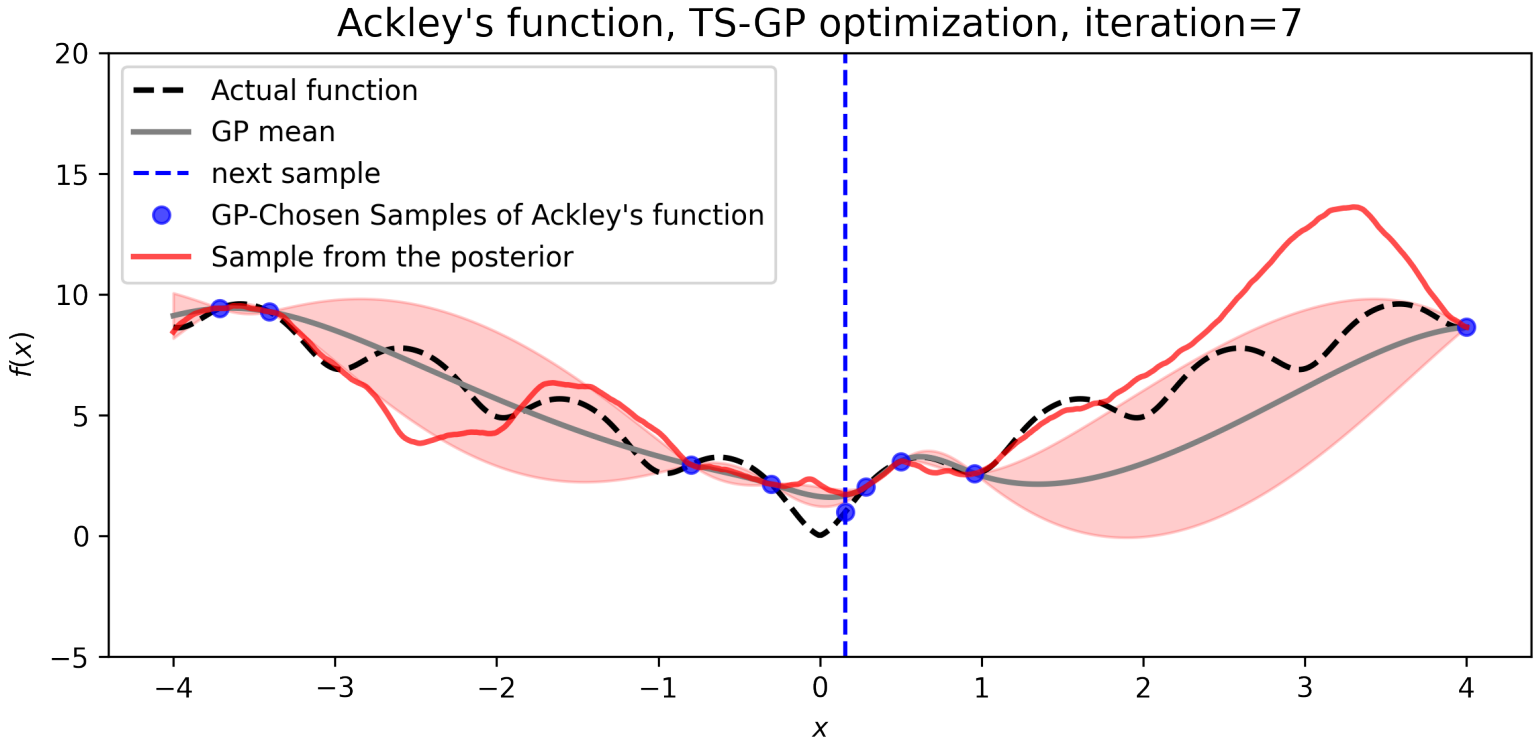


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

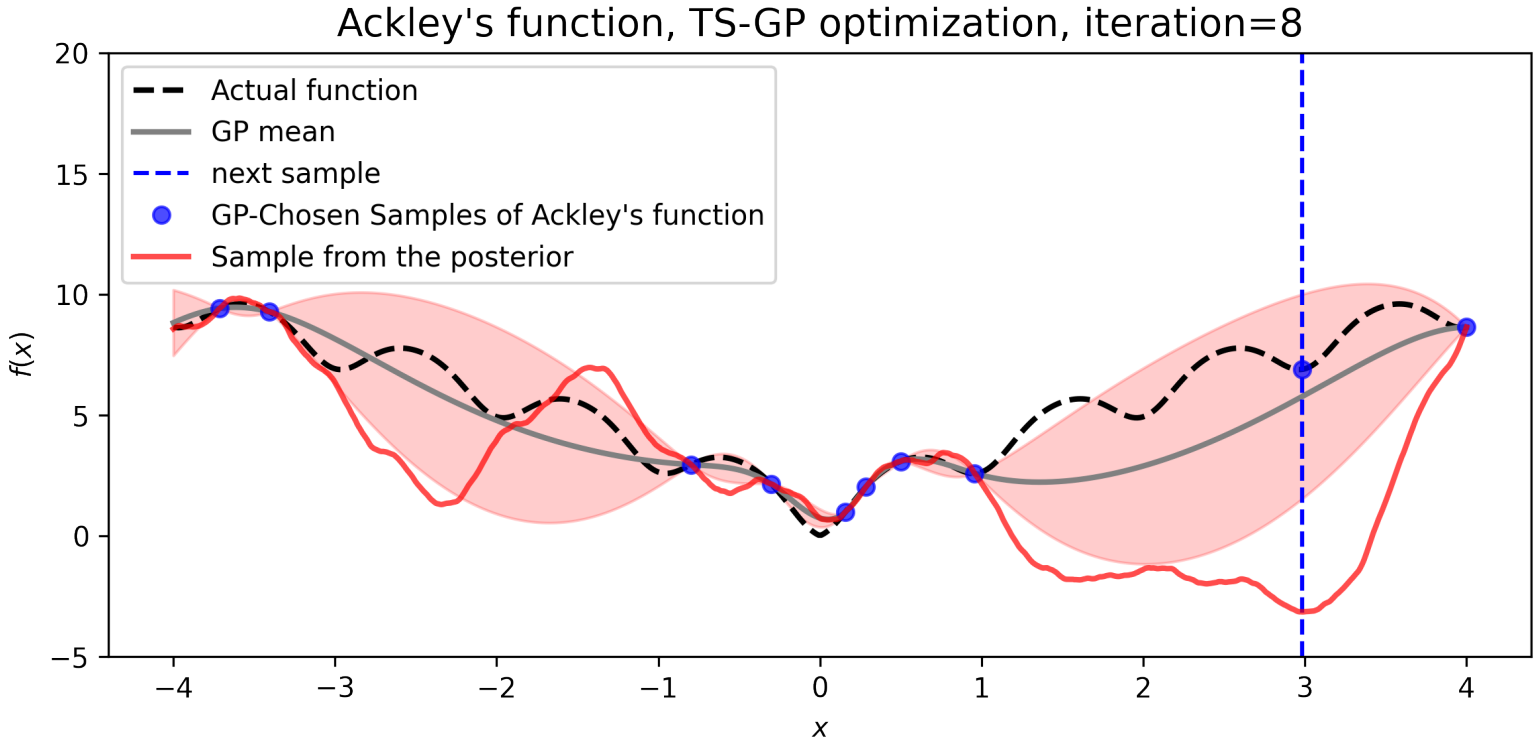


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

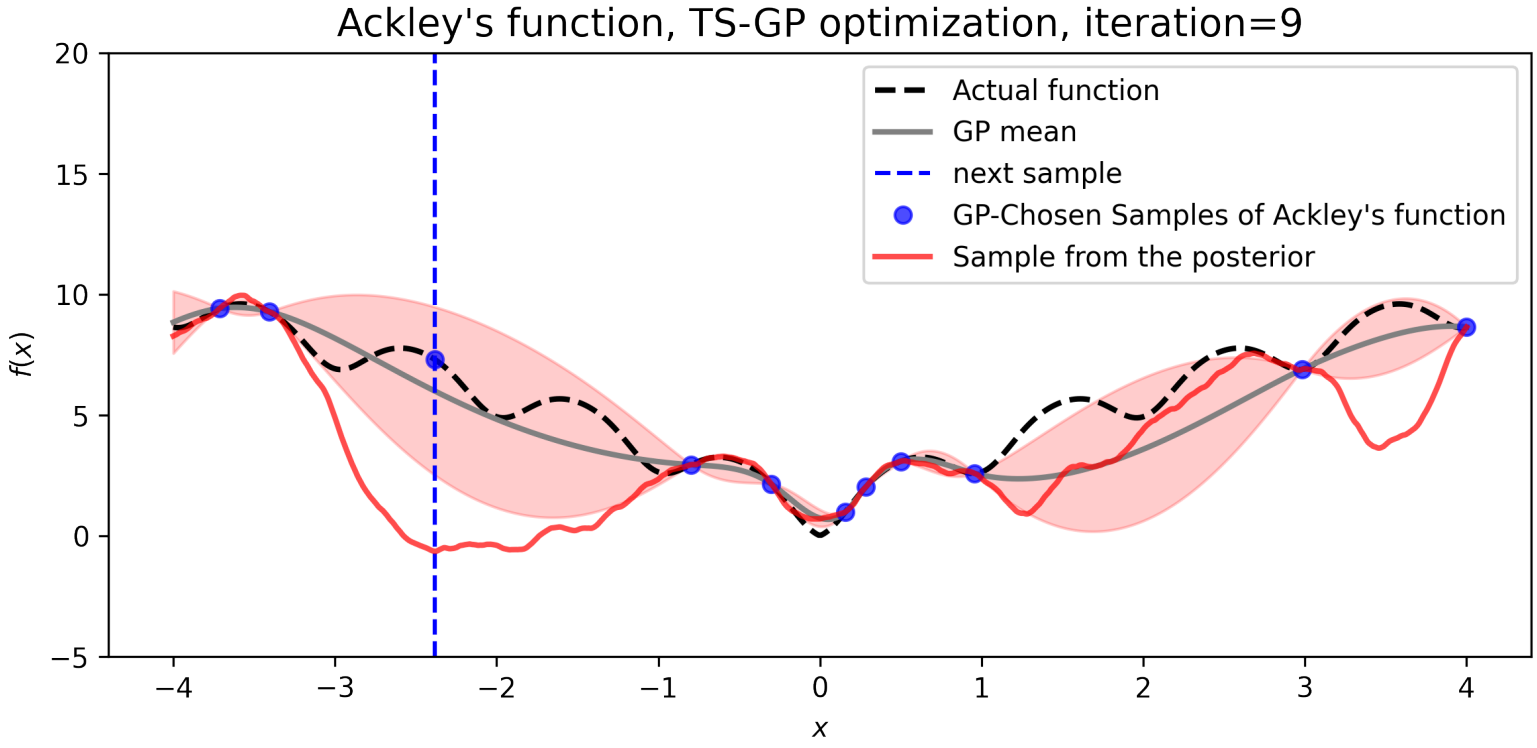


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

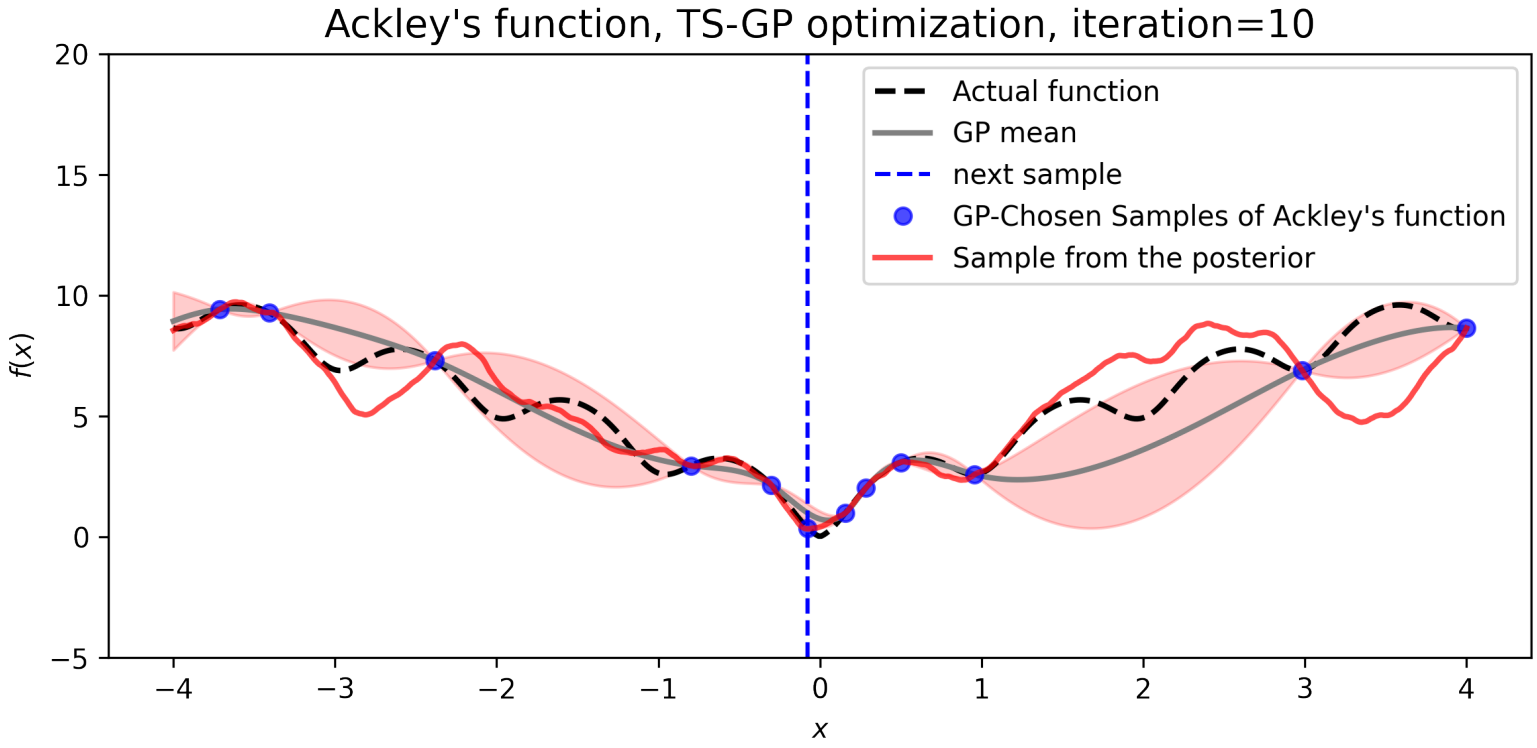


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

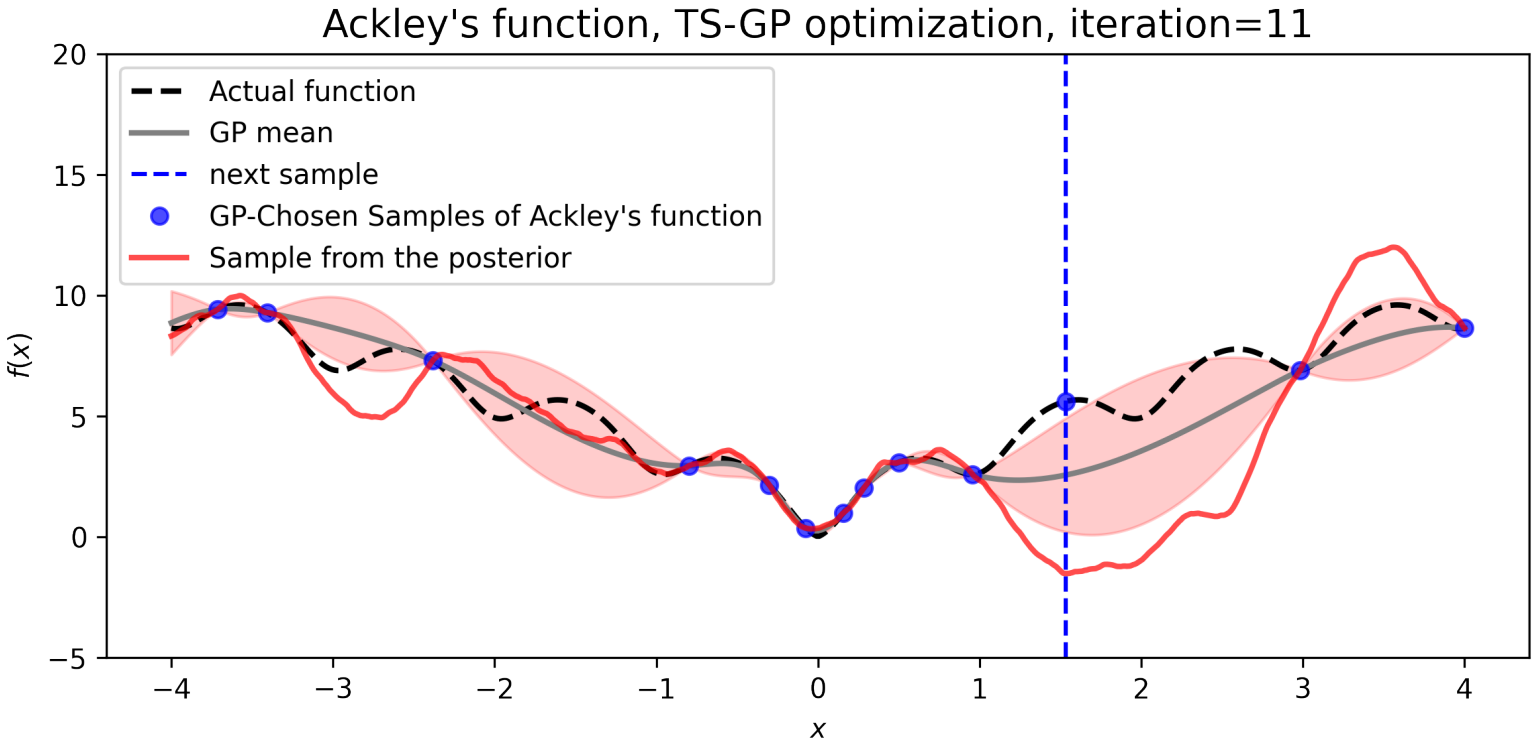


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

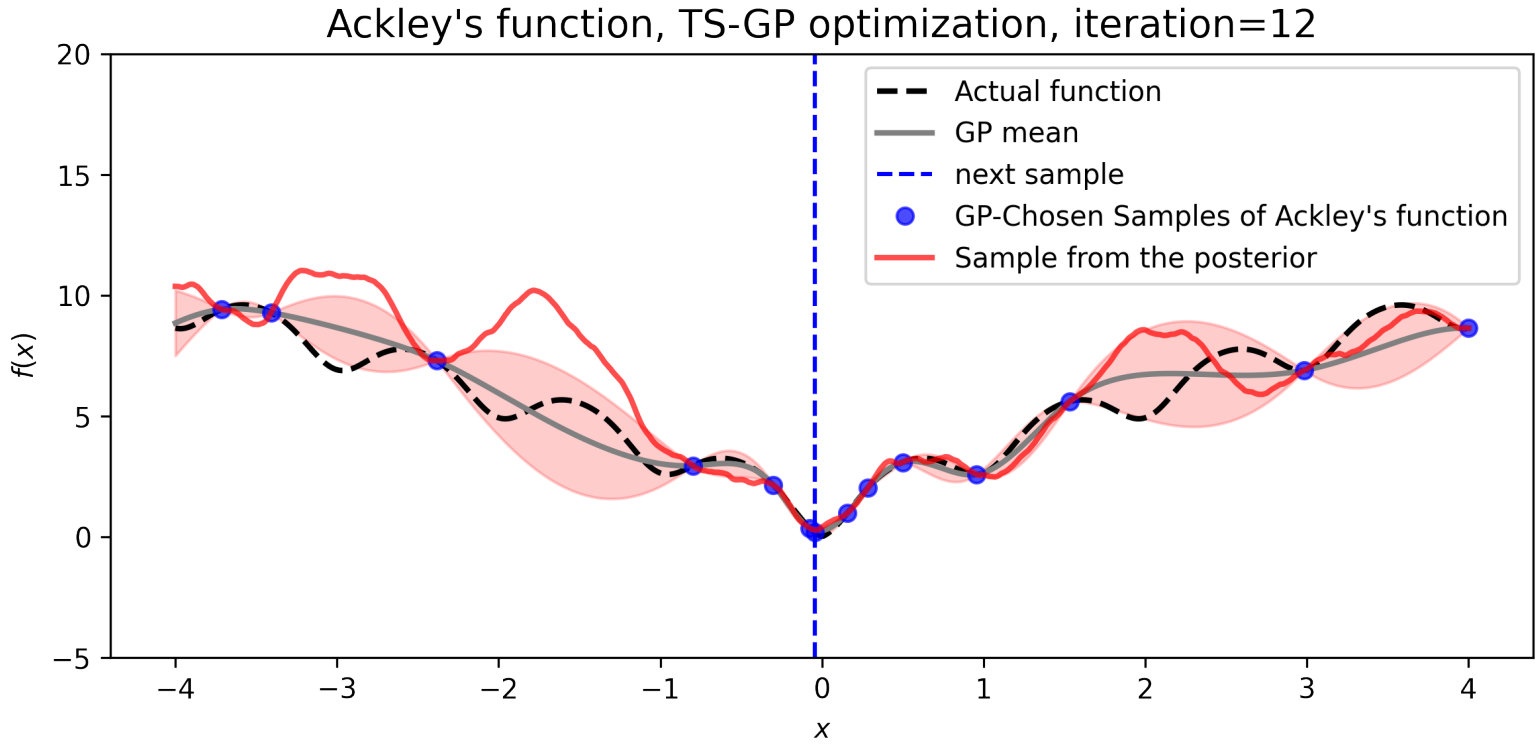


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

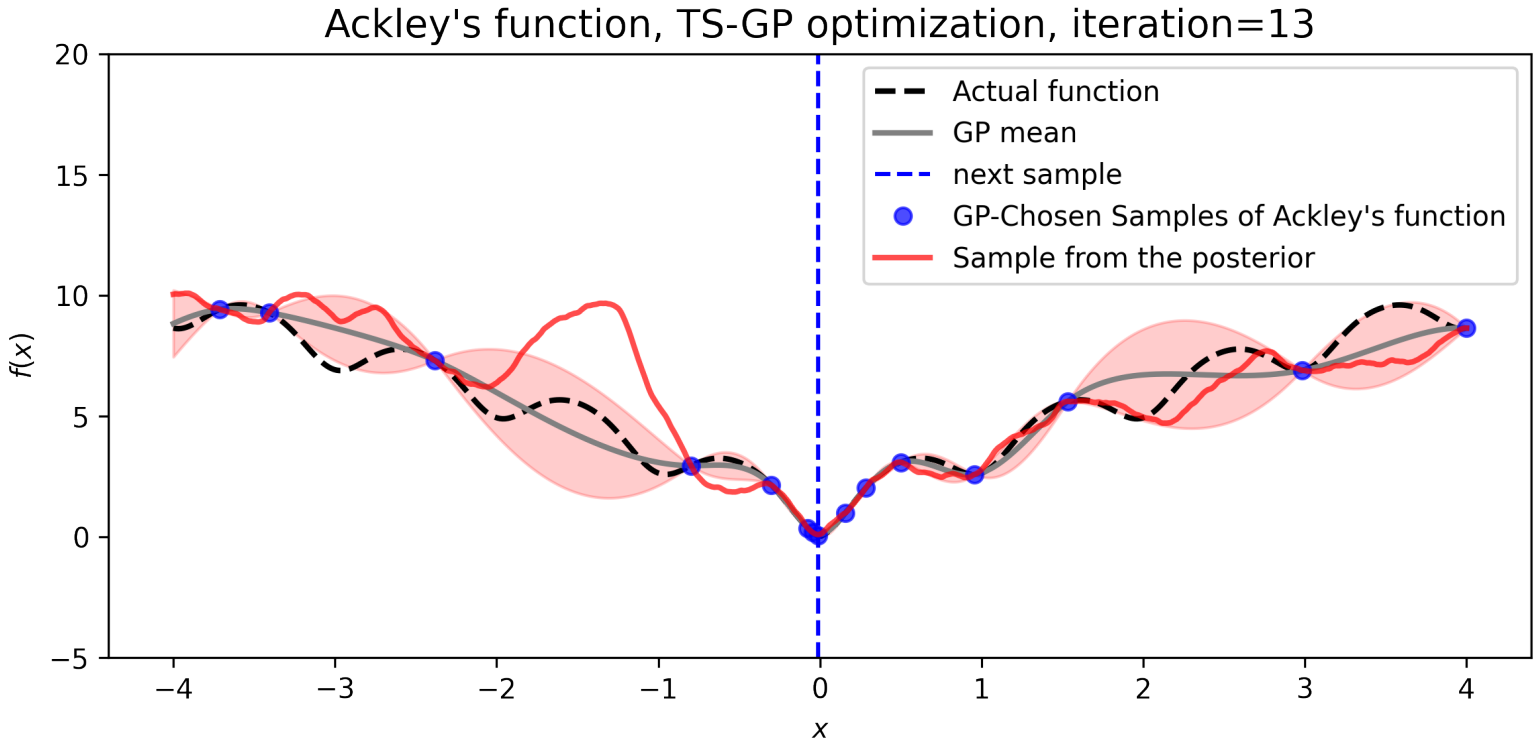


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

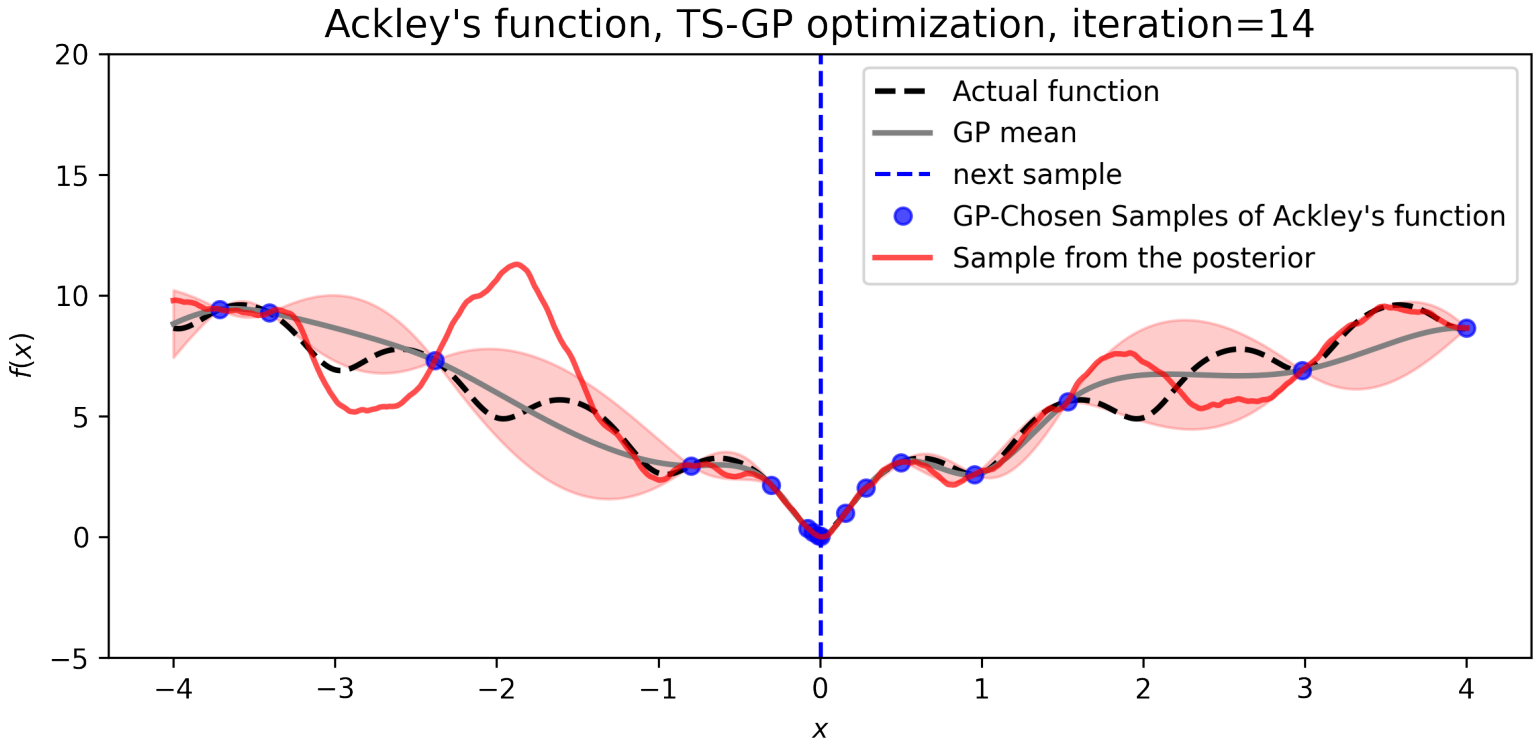


Figure: Iterations of the BO loop for a TS acquisition function

BO using TS

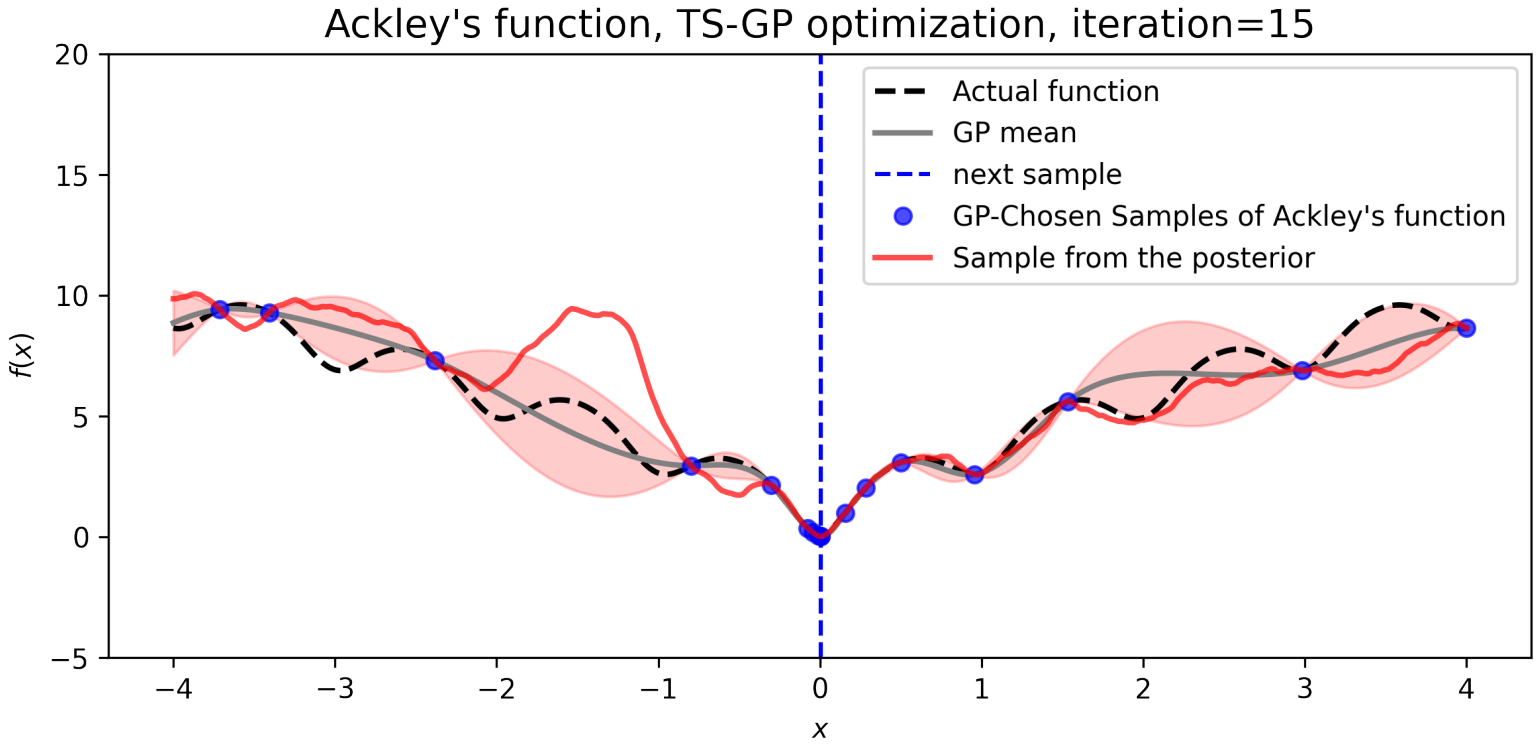


Figure: Iterations of the BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

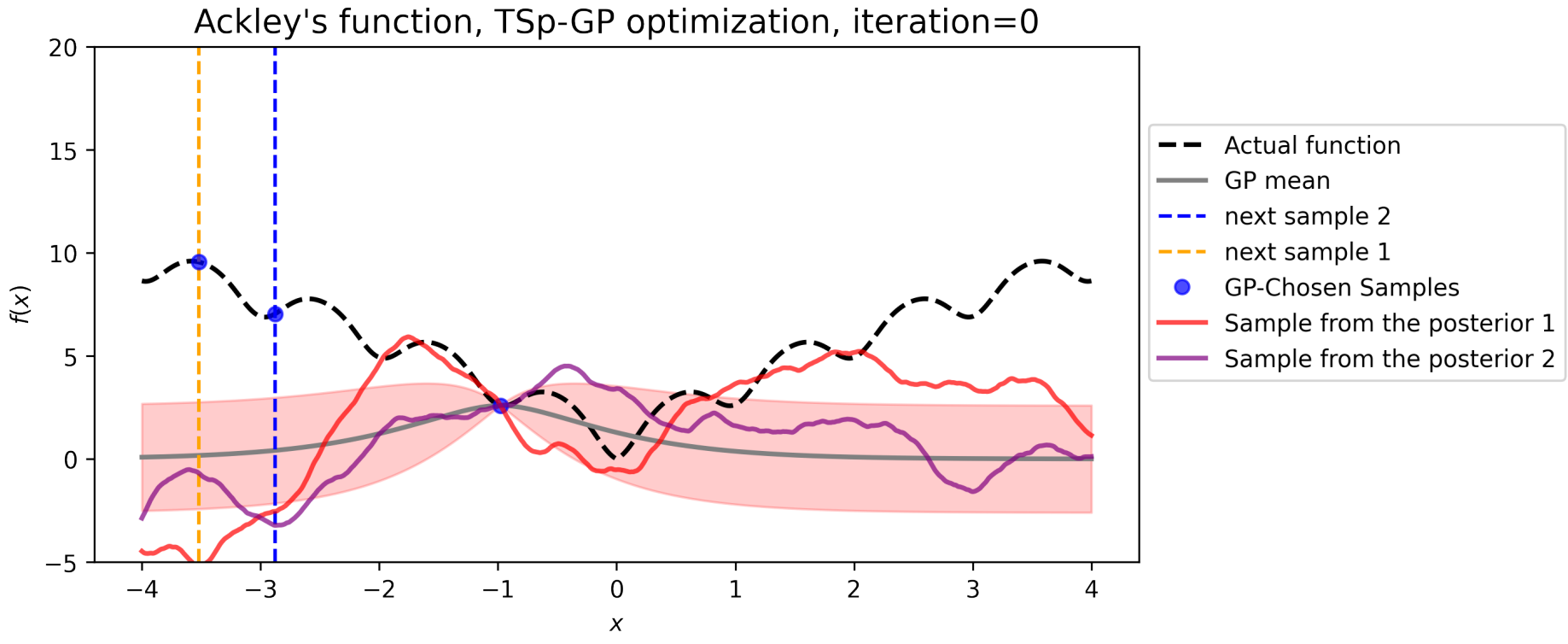


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

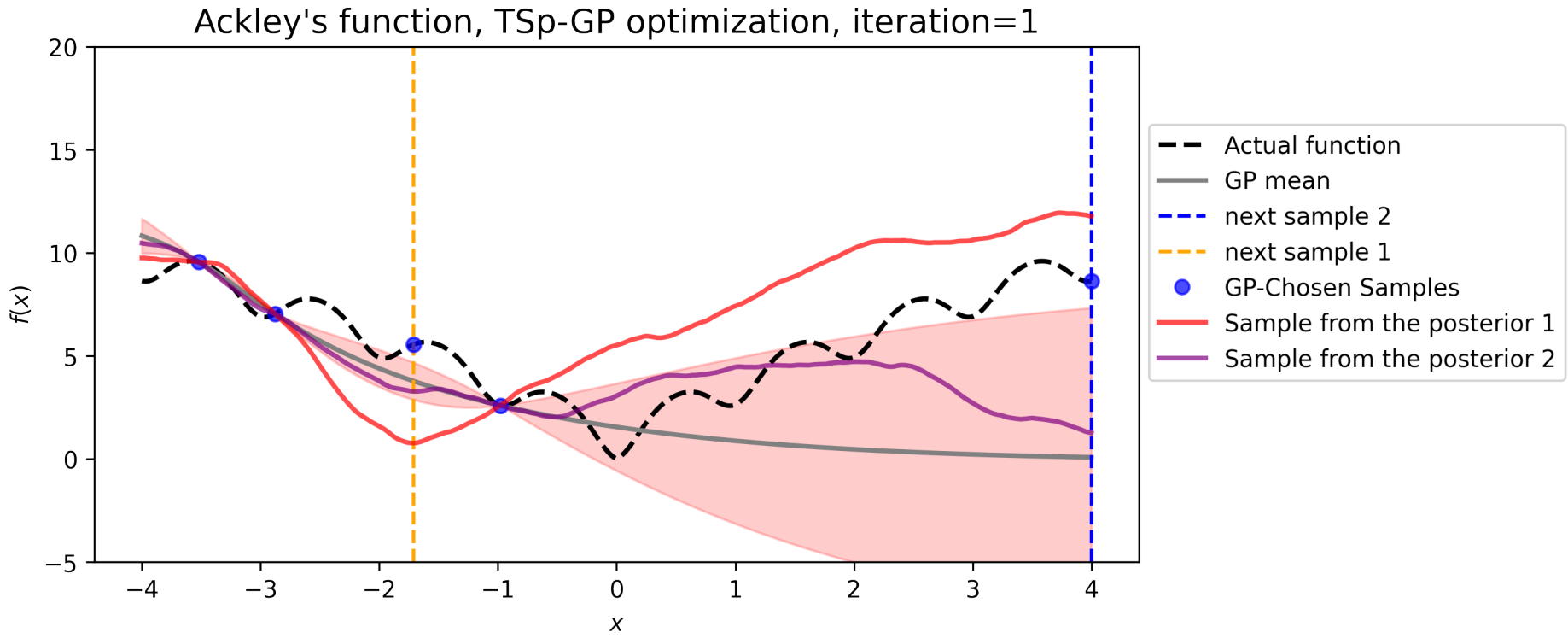


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

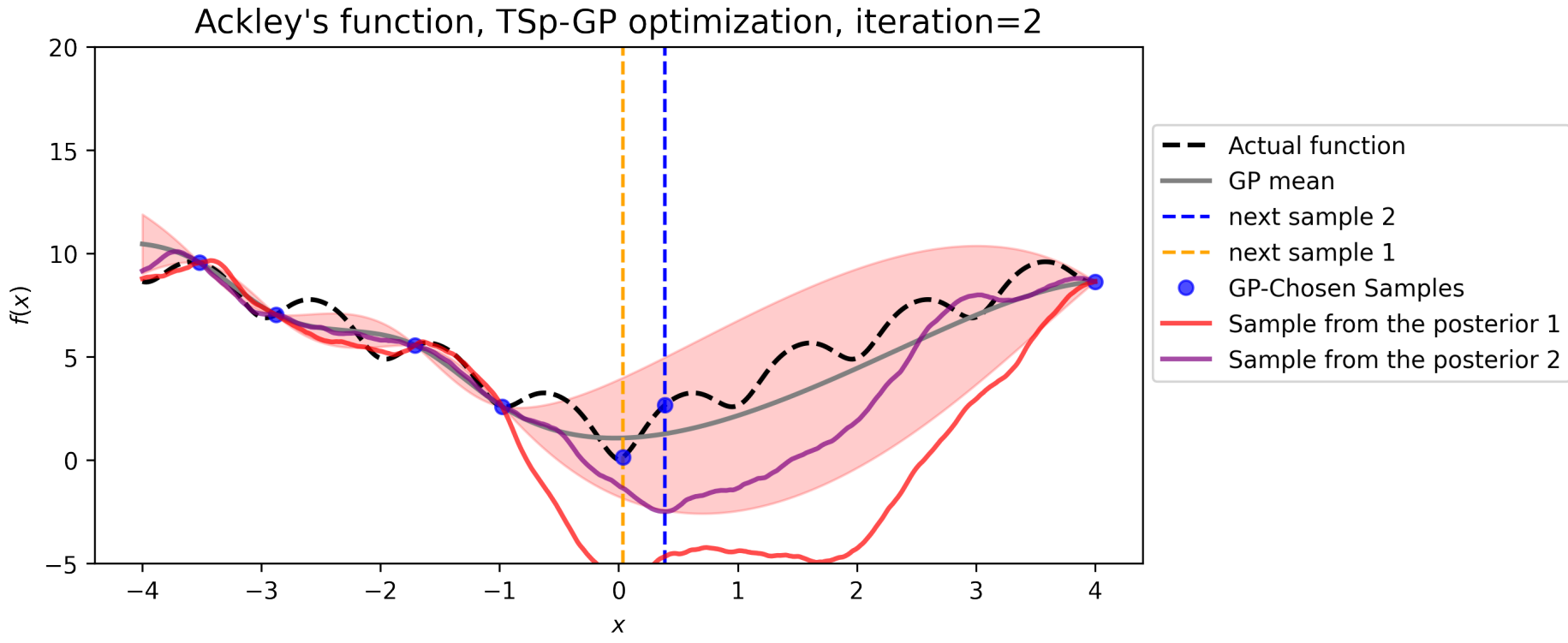


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

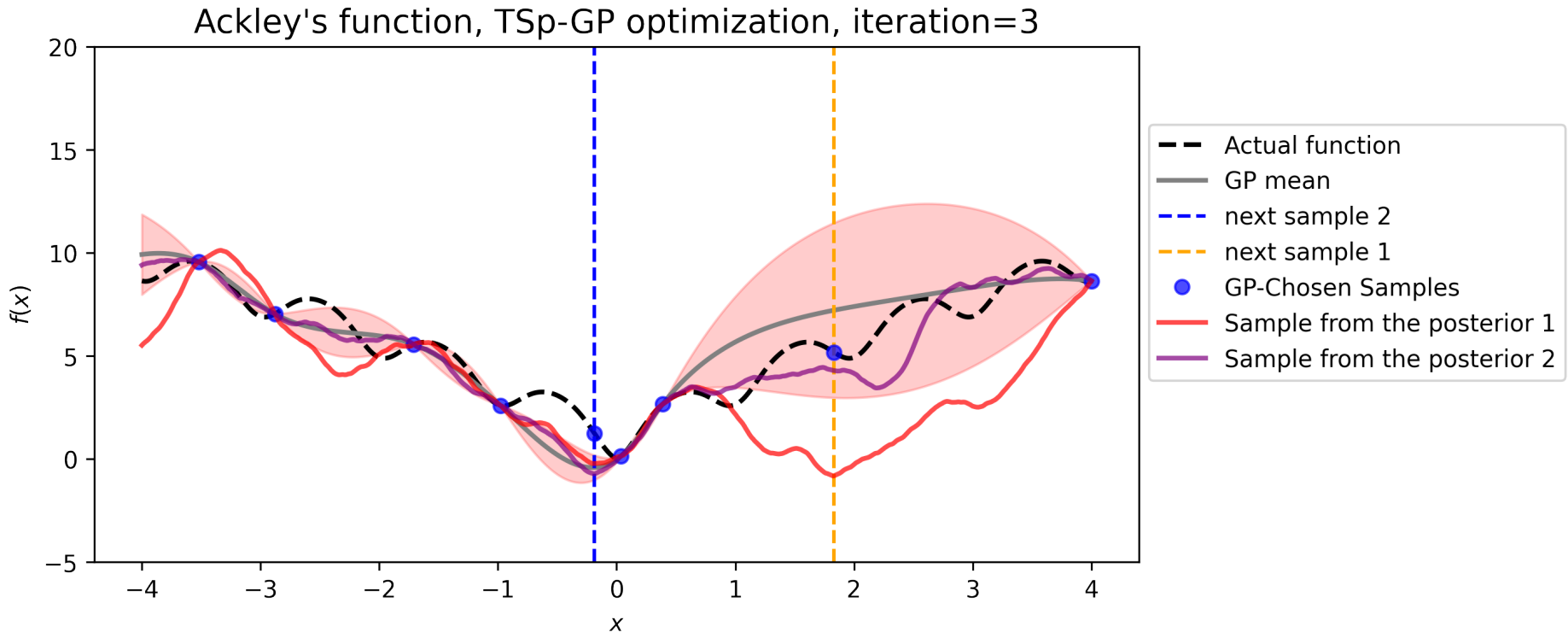


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

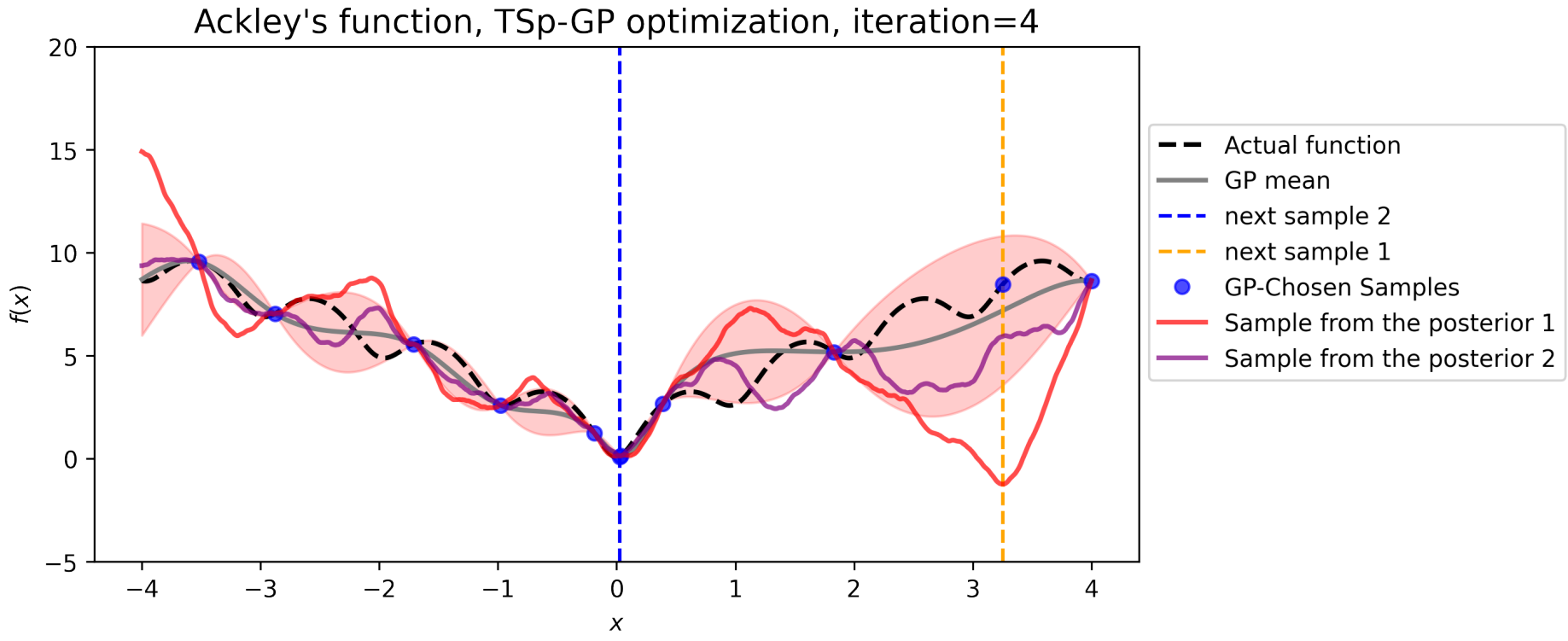


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

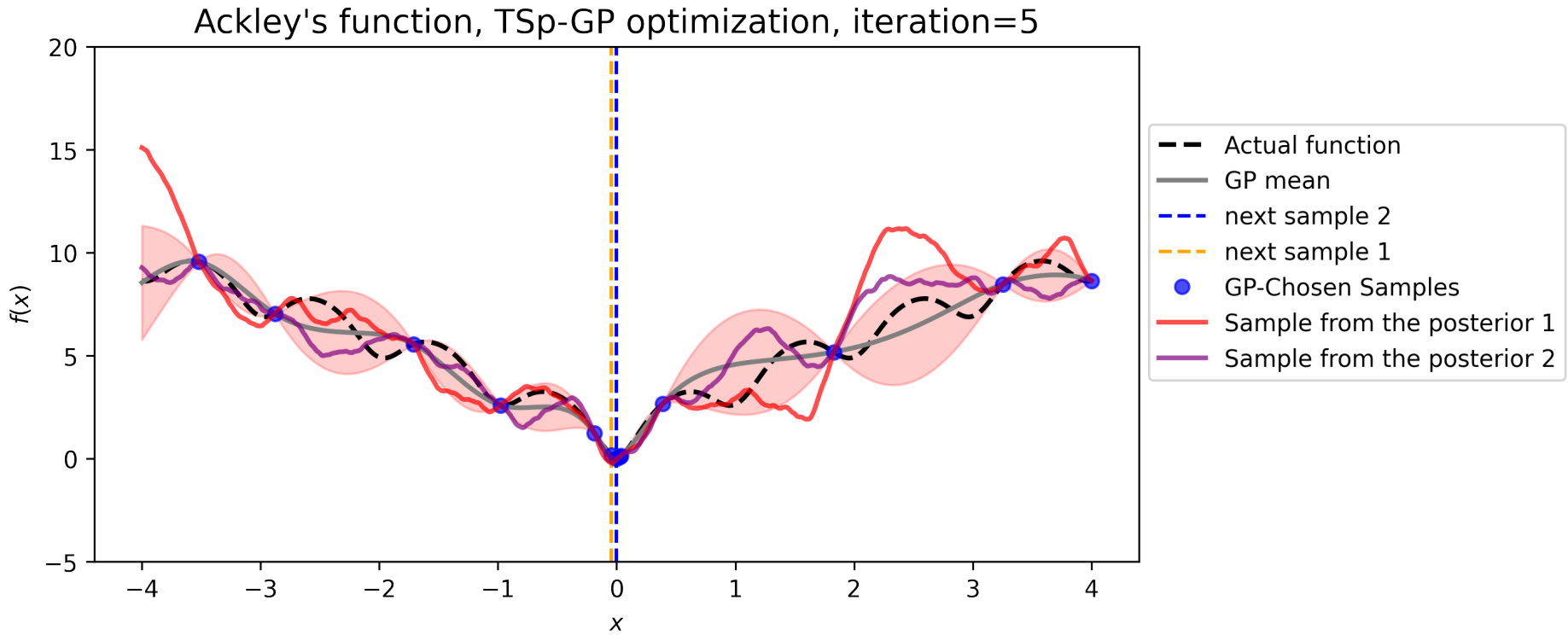


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

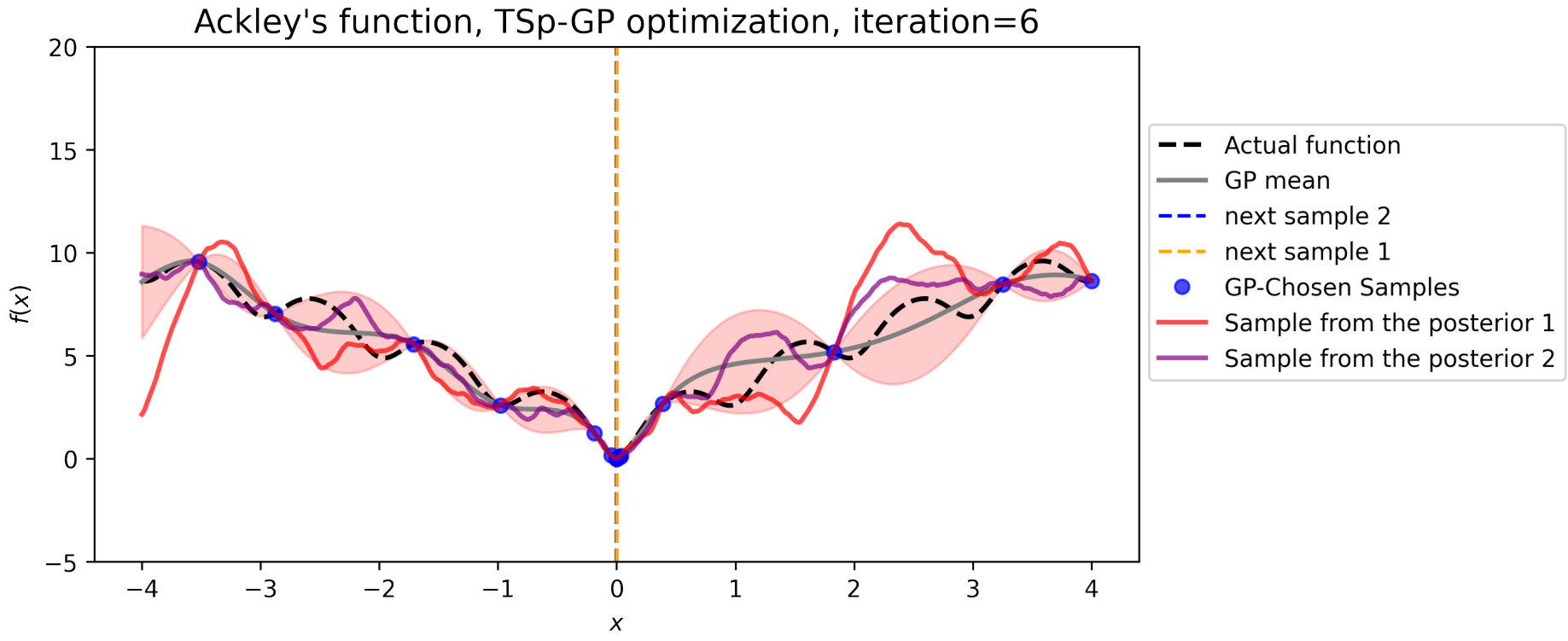


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

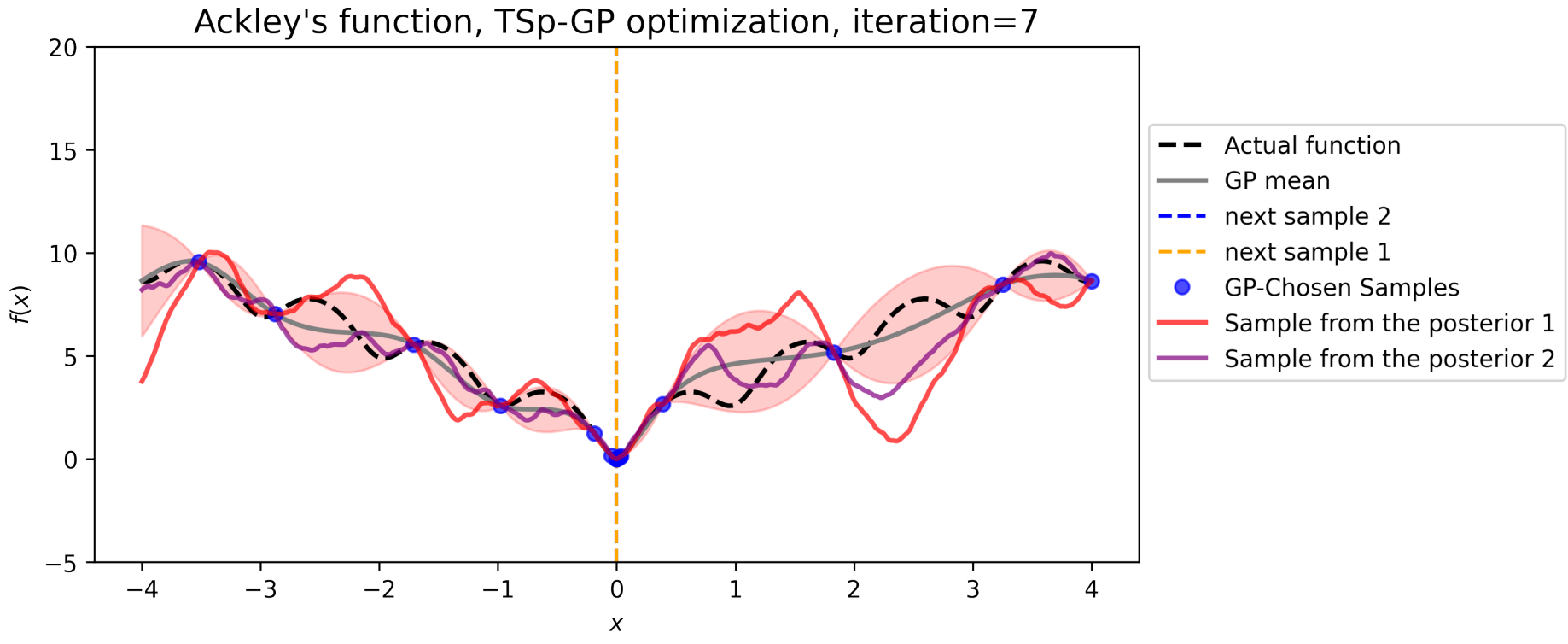


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

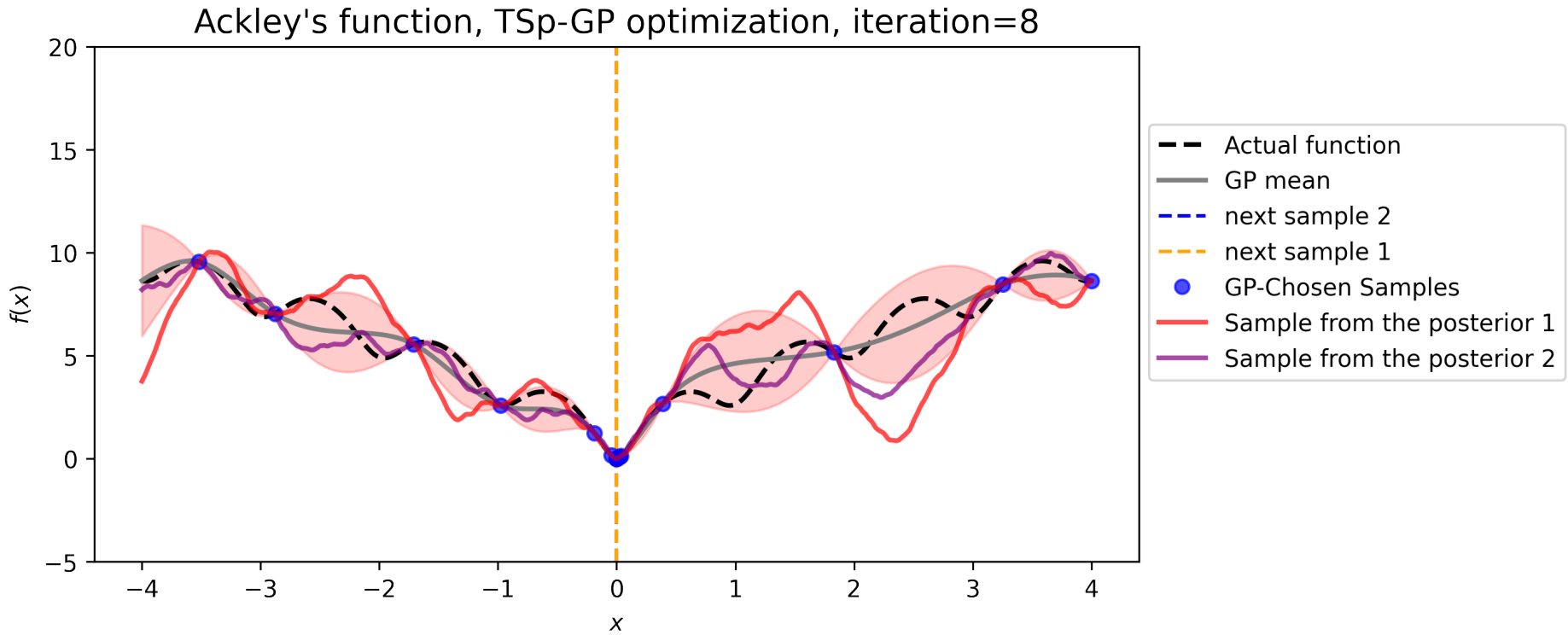


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

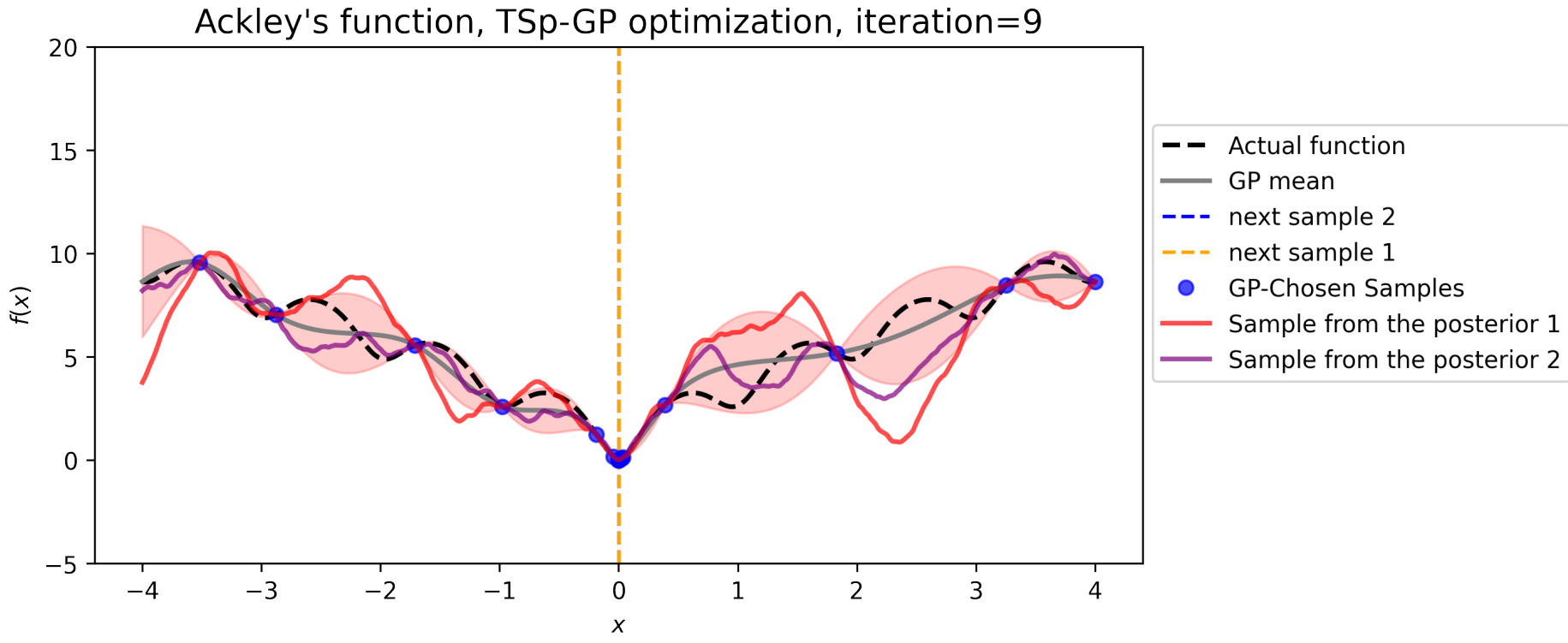


Figure: Iterations of a parallel BO loop for a TS acquisition function

Parallel BO using TS

It is straightforward to use TS for parallel (batch) BO. We must sample b times from the posterior during each BO iteration.

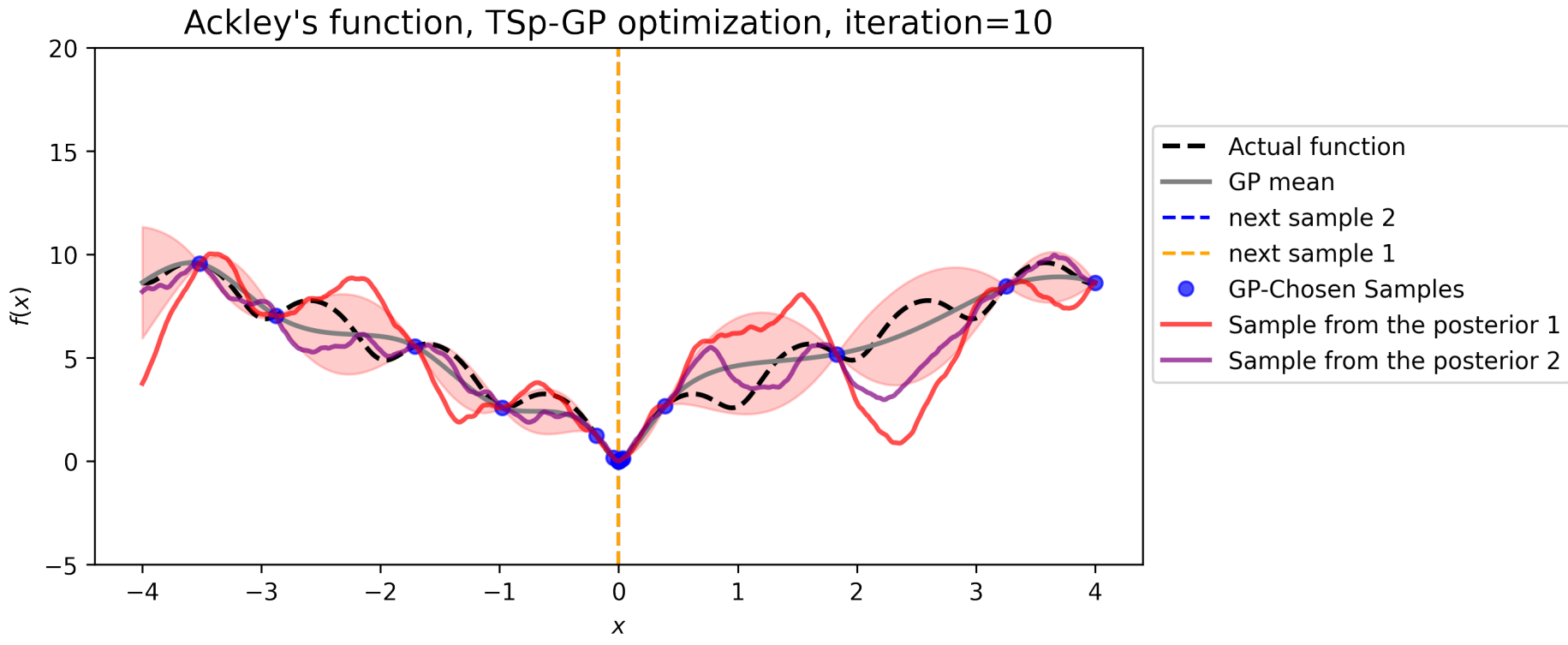


Figure: Iterations of a parallel BO loop for a TS acquisition function

Multi-points Expected improvement (q-EI)

The basic idea of q-EI is to find the q points that jointly maximise the expected improvement.

$$\begin{aligned}\mathrm{EI}(\mathbf{X}^{t+1},\cdots,\mathbf{X}^{t+q}) &= \mathbb{E}[\max((f(\mathbf{X}^{t+1}) - f(\mathbf{X}^*))^+,\cdots, \\ &\qquad (f(\mathbf{X}^{t+q}) - f(\mathbf{X}^*))^+ | \mathcal{I}_t] \\ &= \mathbb{E}[(\min(\mathbf{X}^{t+1},\cdots,\mathbf{X}^{t+q}) - f(\mathbf{X}^*))^+ | \mathcal{I}_t]\end{aligned}$$

Therefore,

$$q - \mathrm{EI}(\mathbf{X}) = \mathbb{E}_t[(\min_{i=1,\cdots,q}(f(\mathbf{X}^{t+i})) - f(\mathbf{X}_t^*))^+] \tag{4}$$

Multi-points Expected improvement (q-EI)

The basic idea of q-EI is to find the q points that jointly maximise the expected improvement.

$$\begin{aligned}\mathrm{EI}(\mathbf{X}^{t+1},\cdots,\mathbf{X}^{t+q}) &= \mathbb{E}[\max((f(\mathbf{X}^{t+1}) - f(\mathbf{X}^*))^+,\cdots, \\ &\qquad (f(\mathbf{X}^{t+q}) - f(\mathbf{X}^*))^+ | \mathcal{I}_t] \\ &= \mathbb{E}[(\min(\mathbf{X}^{t+1},\cdots,\mathbf{X}^{t+q}) - f(\mathbf{X}^*))^+ | \mathcal{I}_t]\end{aligned}$$

Therefore,

$$q - \mathrm{EI}(\mathbf{X}) = \mathbb{E}_t[(\min_{i=1,\cdots,q}(f(\mathbf{X}^{t+i})) - f(\mathbf{X}_t^*))^+] \tag{4}$$

This estimation can be done sequentially (batch) or simultaneously (parallel).

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a fake sample from the surrogate model $y'_t = \mu(\mathbf{x}_t)$

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a fake sample from the surrogate model $y'_t = \mu(\mathbf{x}_t)$
3. Add the fake sample to previously observed samples $\mathcal{D}'_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y'_t)\}$ and train a new fake GP.

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a fake sample from the surrogate model $y'_t = \mu(\mathbf{x}_t)$
3. Add the fake sample to previously observed samples $\mathcal{D}'_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y'_t)\}$ and train a new fake GP.
4. Find the next sampling point $\mathbf{x}_{t+1}$ by optimizing the acquisition function over the fake GP: $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}'_t)$

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a fake sample from the surrogate model $y'_t = \mu(\mathbf{x}_t)$
3. Add the fake sample to previously observed samples $\mathcal{D}'_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y'_t)\}$ and train a new fake GP.
4. Find the next sampling point $\mathbf{x}_{t+1}$ by optimizing the acquisition function over the fake GP: $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}'_t)$
5. Obtain noisy samples y_t, y_{t+1} from the objective function f.

Sequential q-EI algorithm

1. Find the next sampling point $\mathbf{x}_t$ by optimizing the acquisition function over the GP: $\mathbf{x}_t = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}_{t-1})$
2. Obtain a fake sample from the surrogate model $y'_t = \mu(\mathbf{x}_t)$
3. Add the fake sample to previously observed samples $\mathcal{D}'_{1:t} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y'_t)\}$ and train a new fake GP.
4. Find the next sampling point $\mathbf{x}_{t+1}$ by optimizing the acquisition function over the fake GP: $\mathbf{x}_{t+1} = \text{argmax}_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{I}'_t)$
5. Obtain noisy samples y_t, y_{t+1} from the objective function f.
6. Add the samples to previous samples $\mathcal{D}_{1:t+1} = \{\mathcal{D}_{1:t-1}, (\mathbf{x}_t, y_t), (\mathbf{x}_{t+1}, y_{t+1})\}$ and update the GP.

Example of batch q-EI

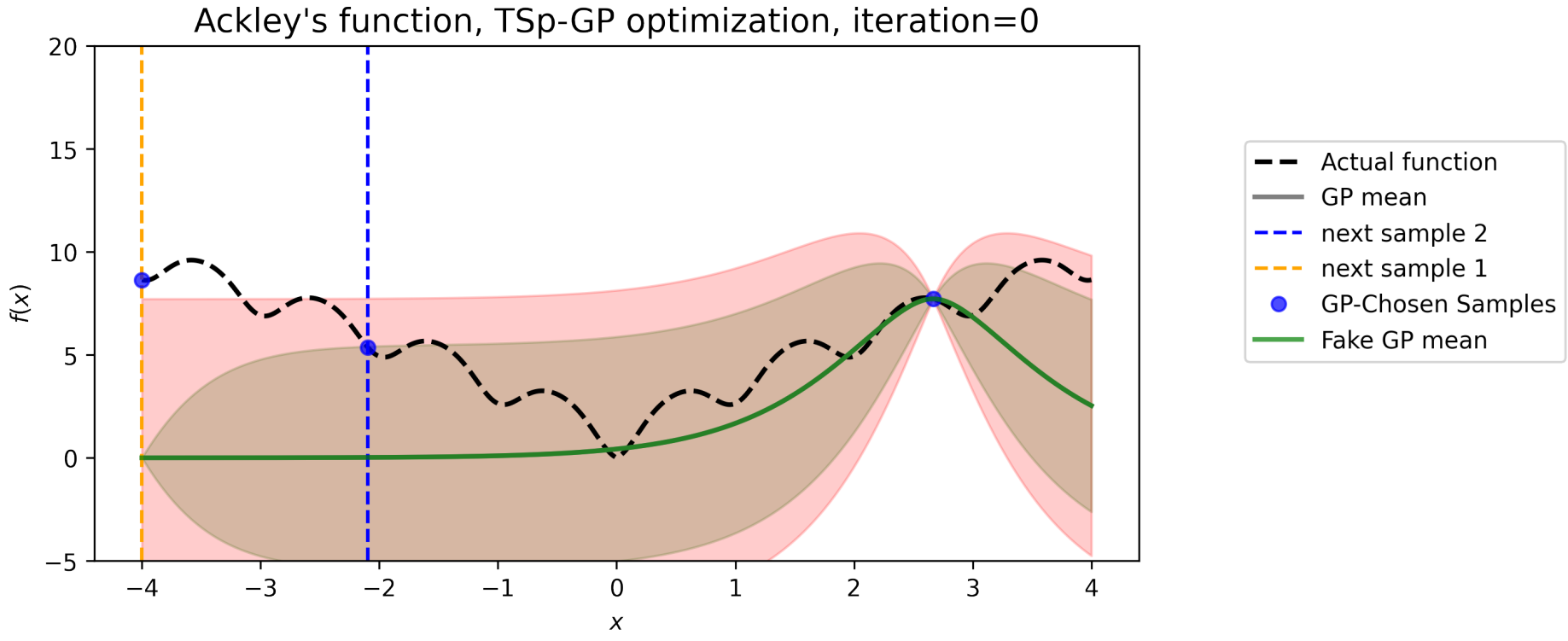


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

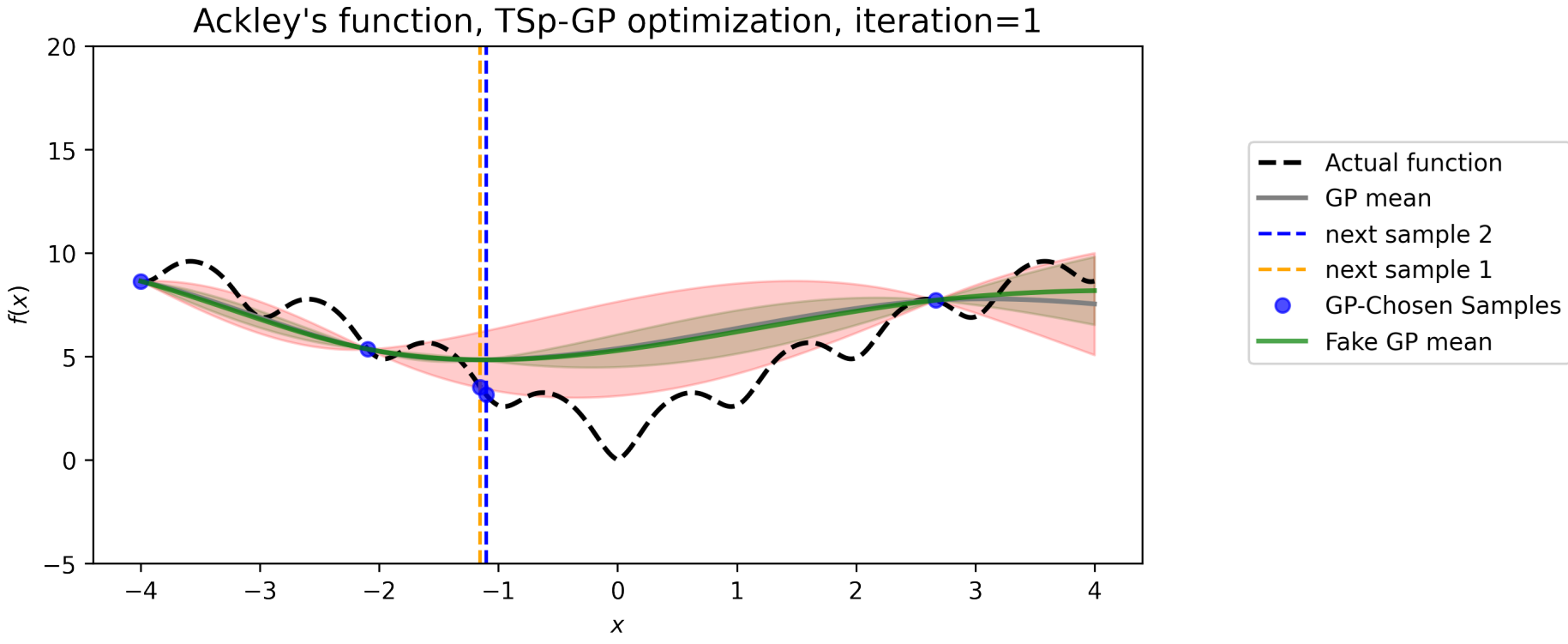


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

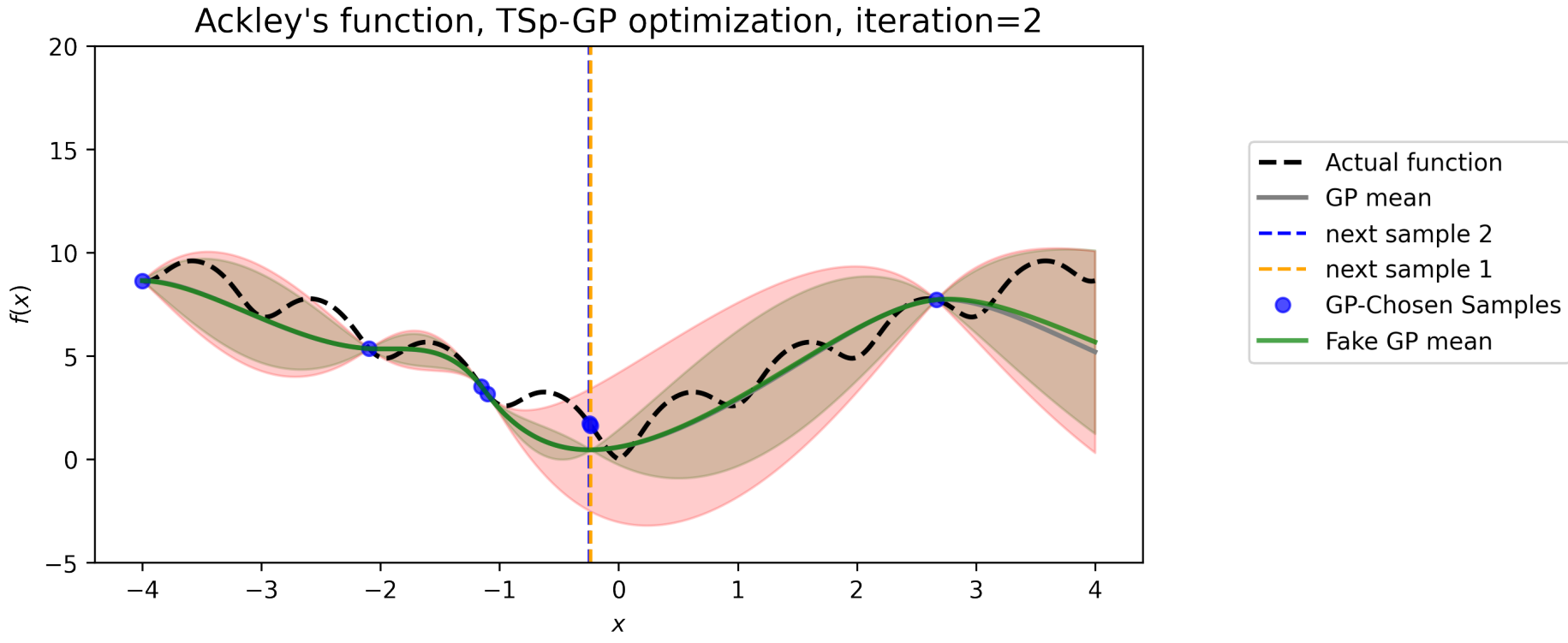


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

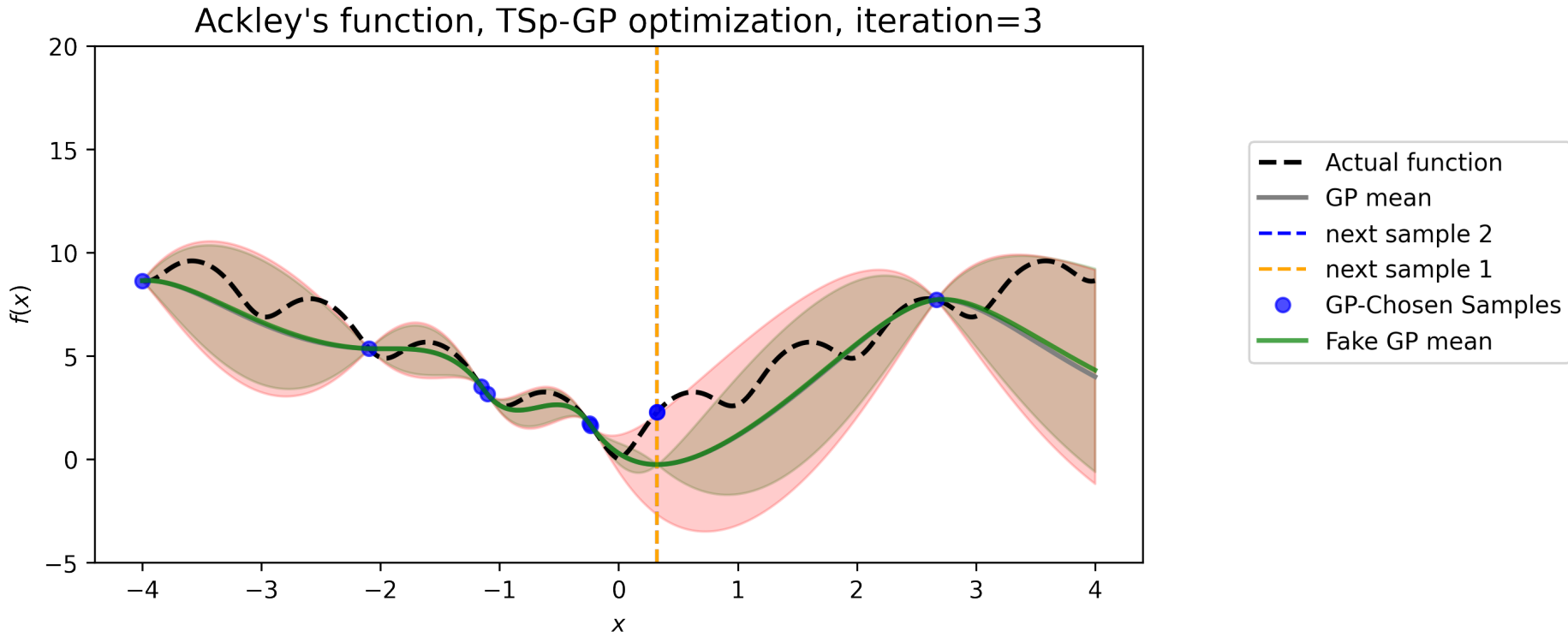


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

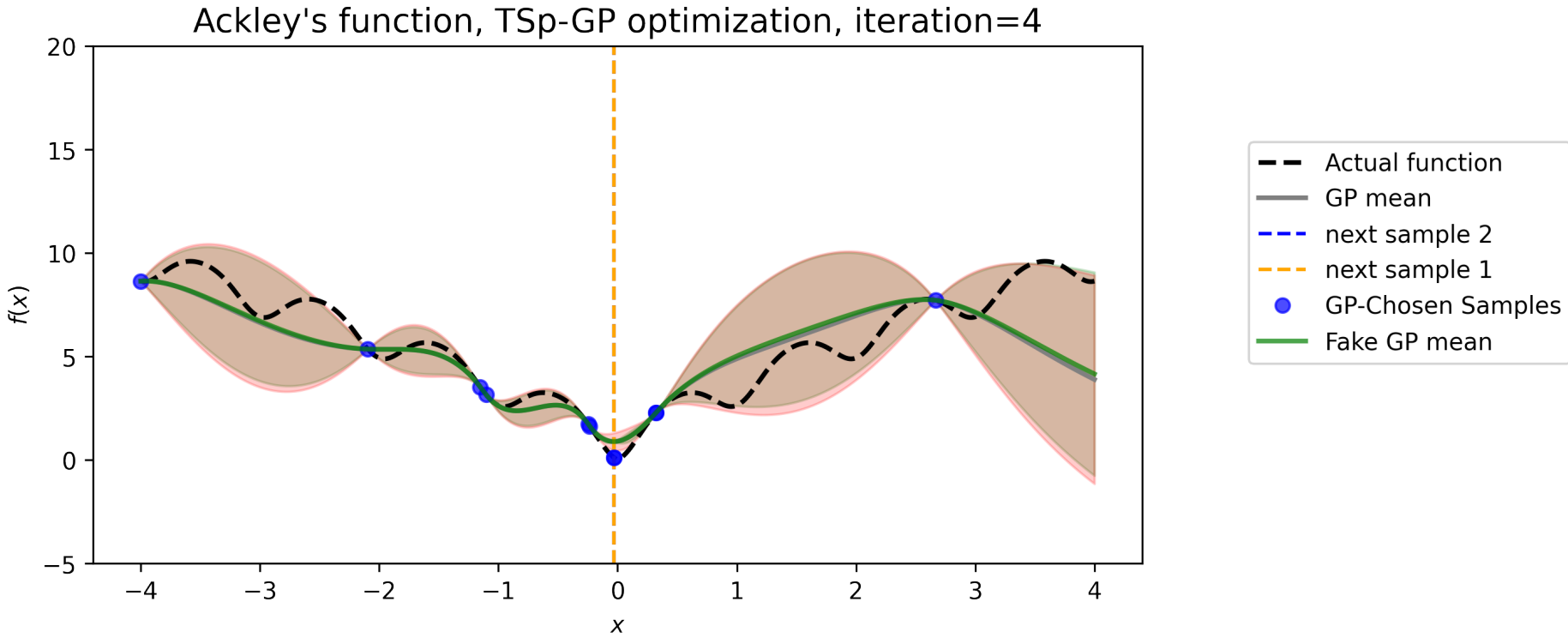


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

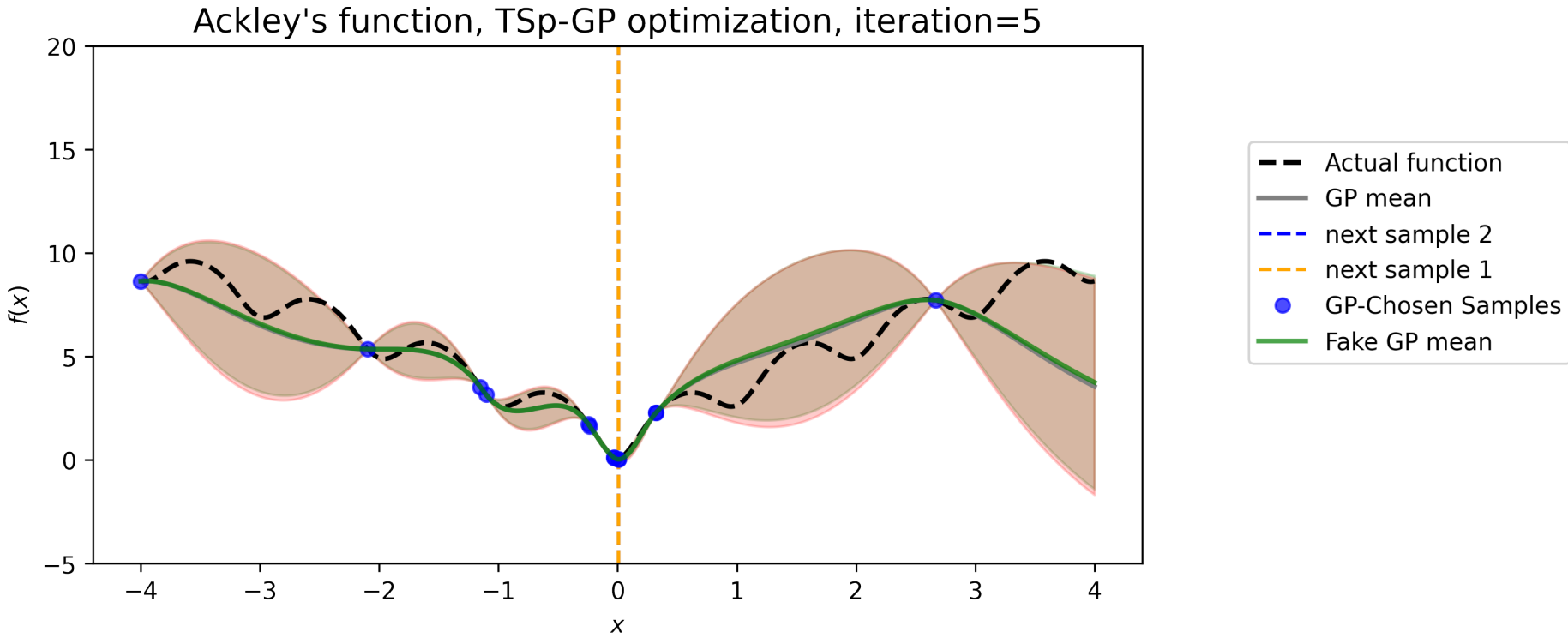


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

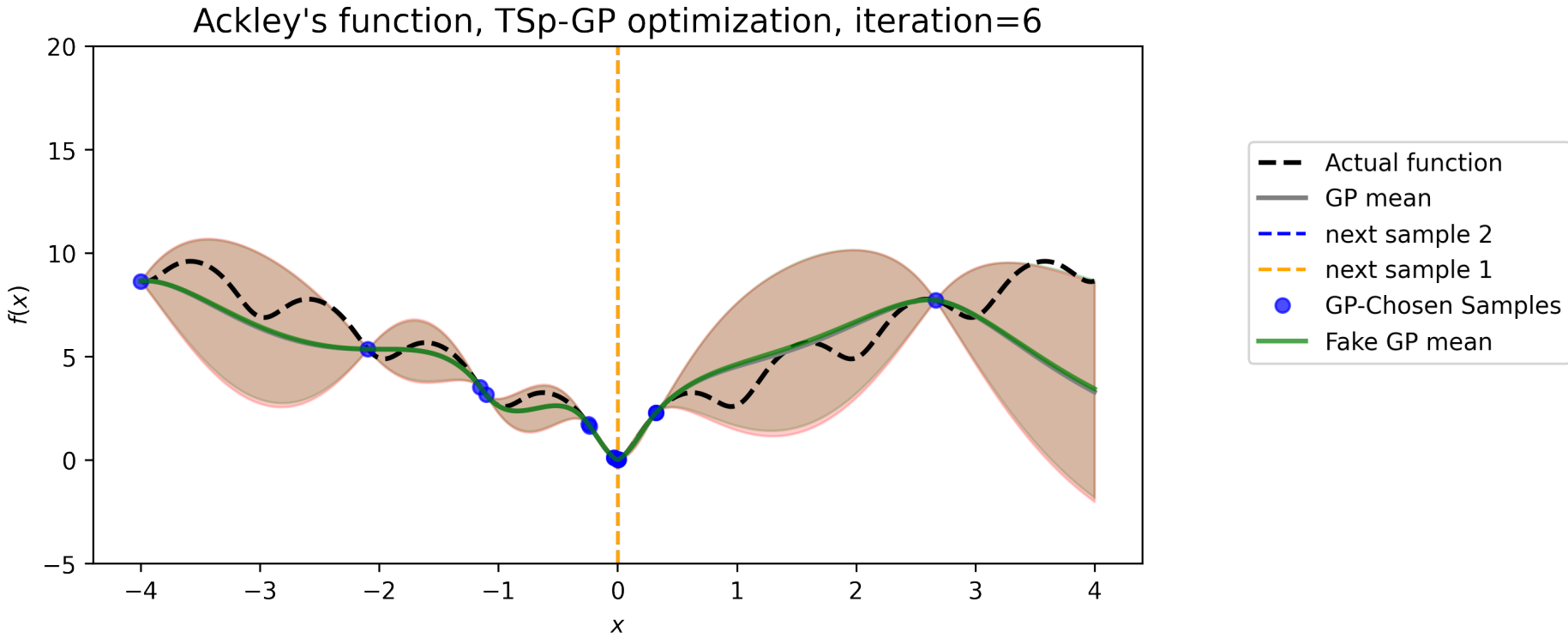


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

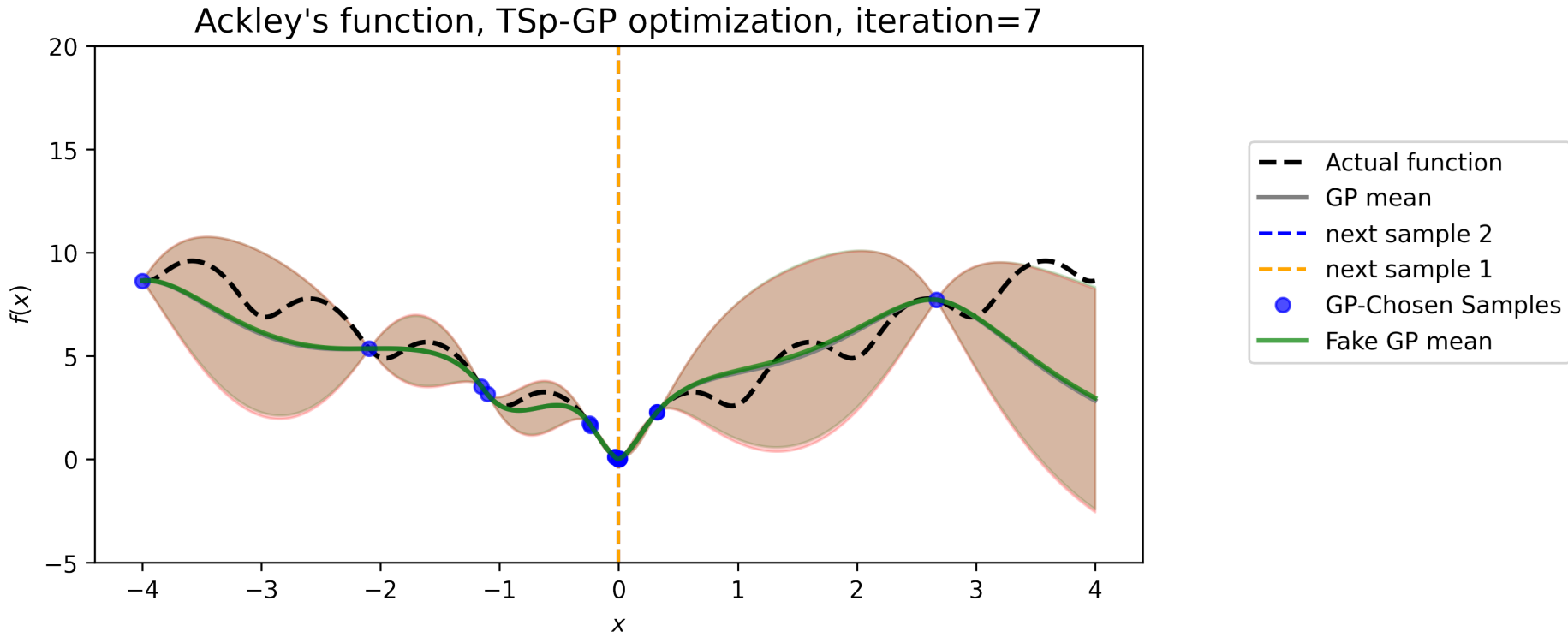


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

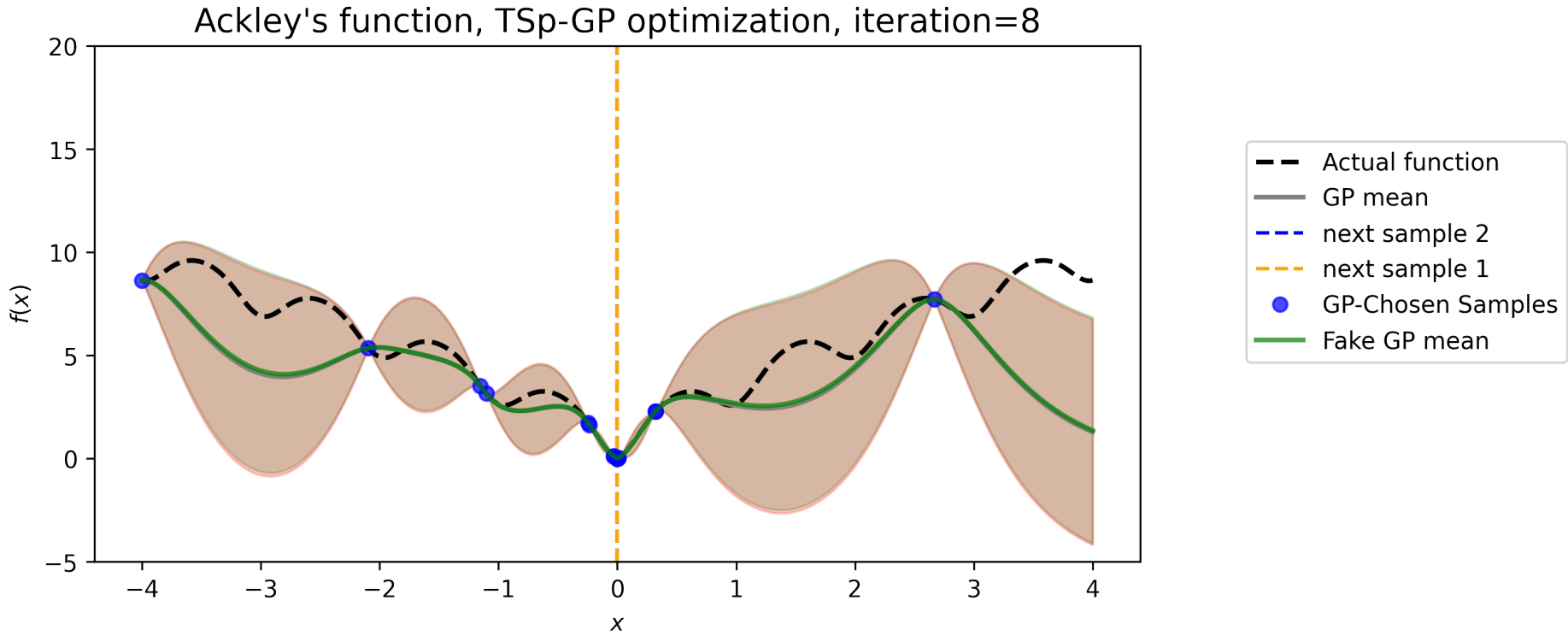


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

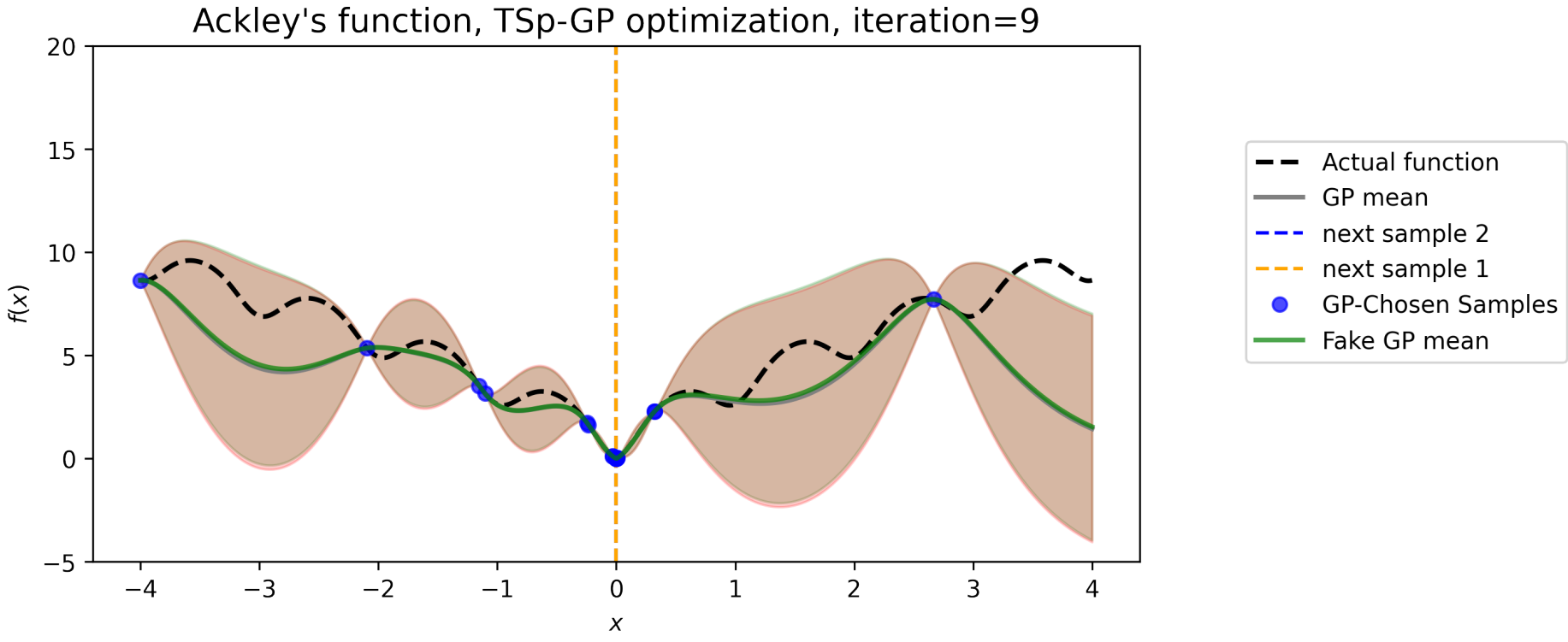


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

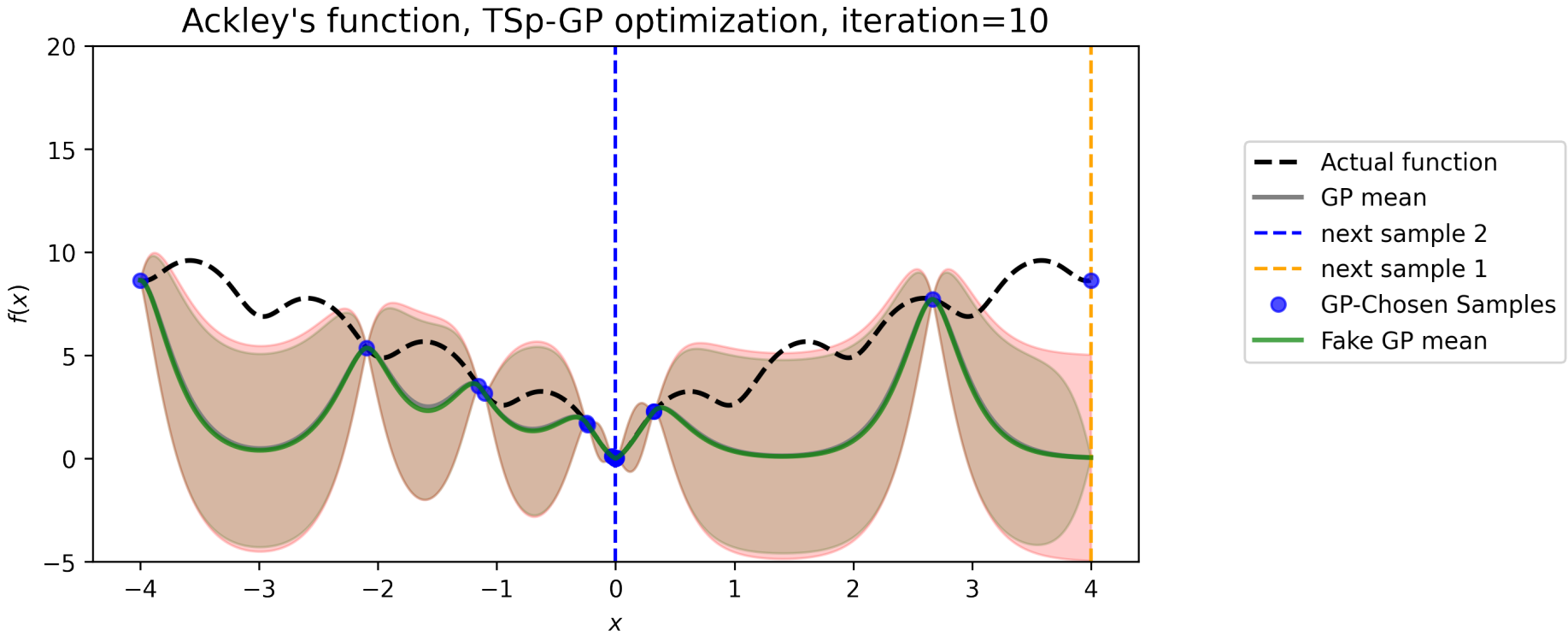


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

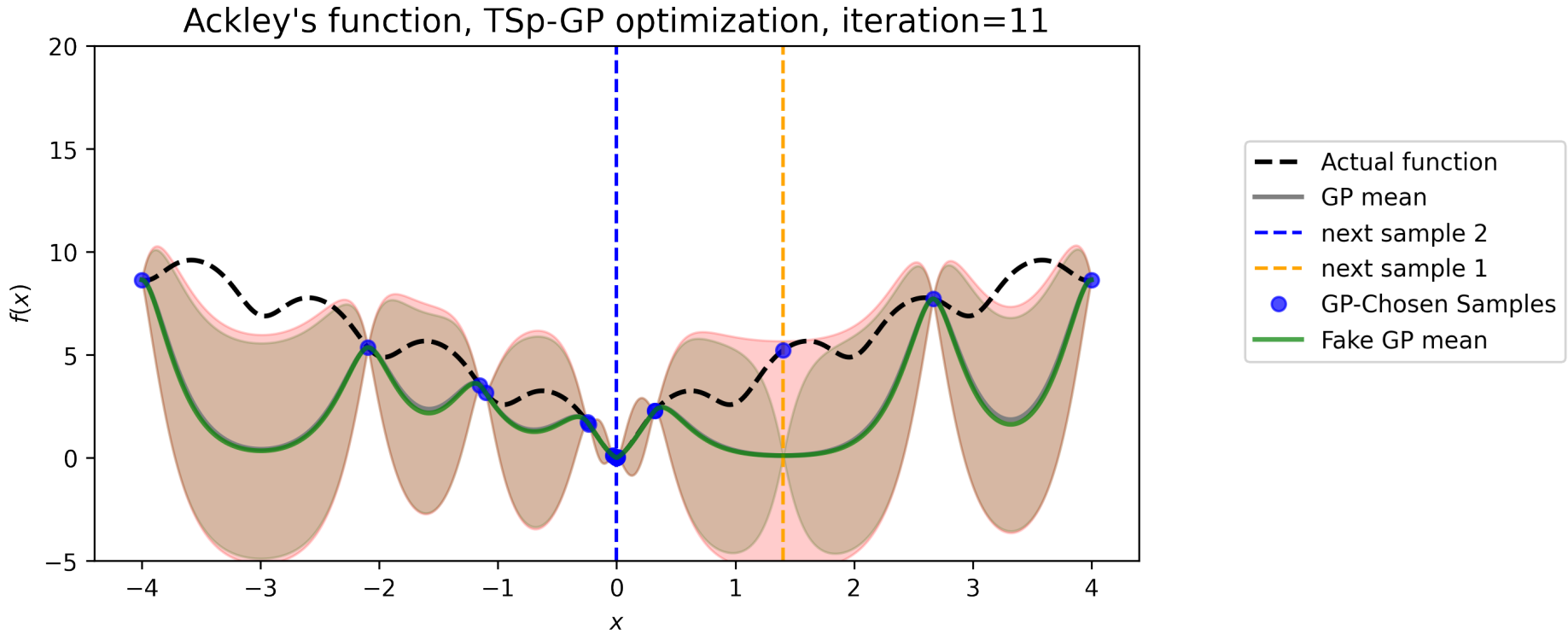


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

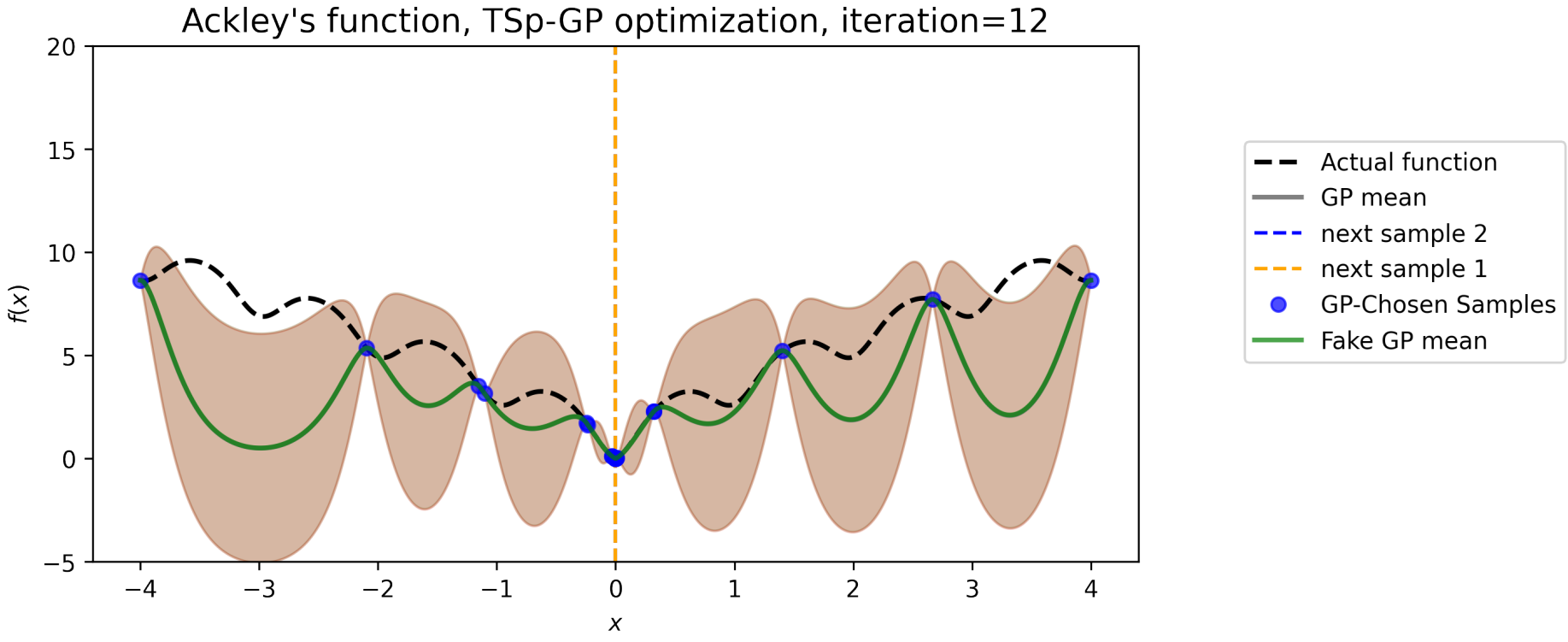


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

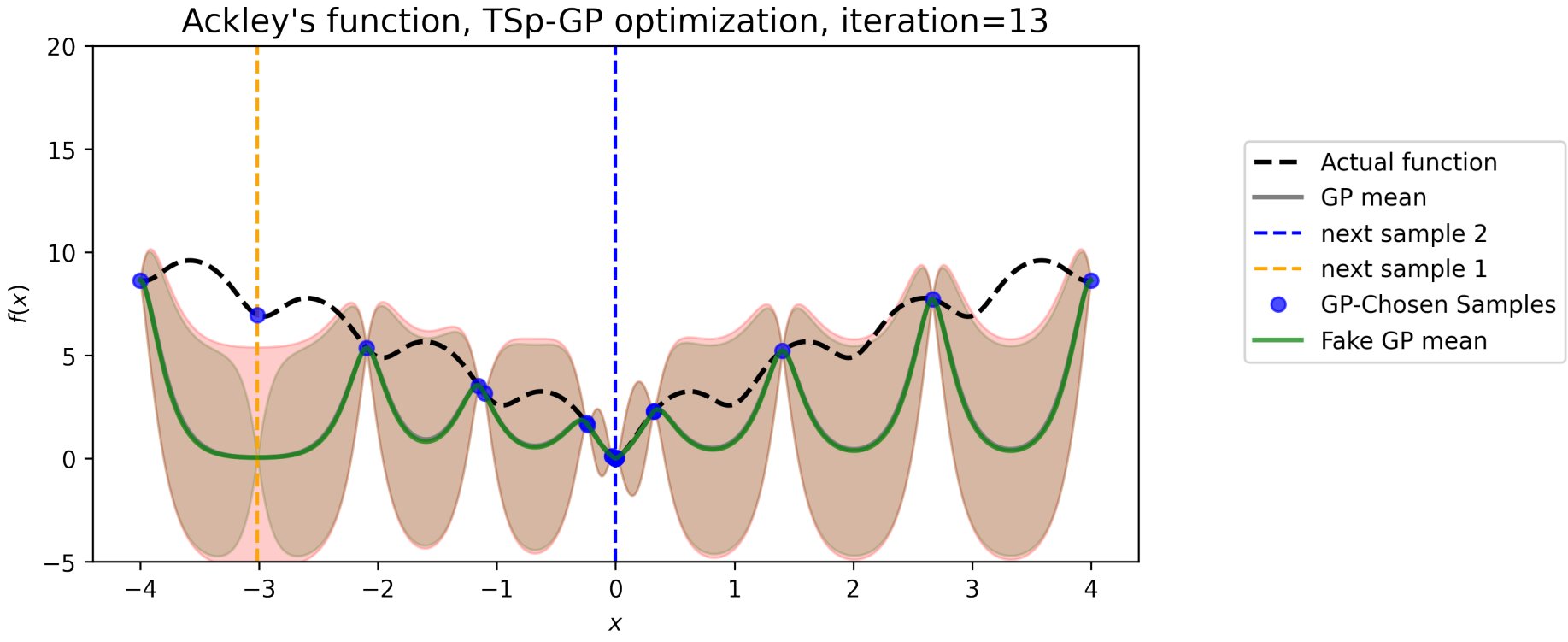


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

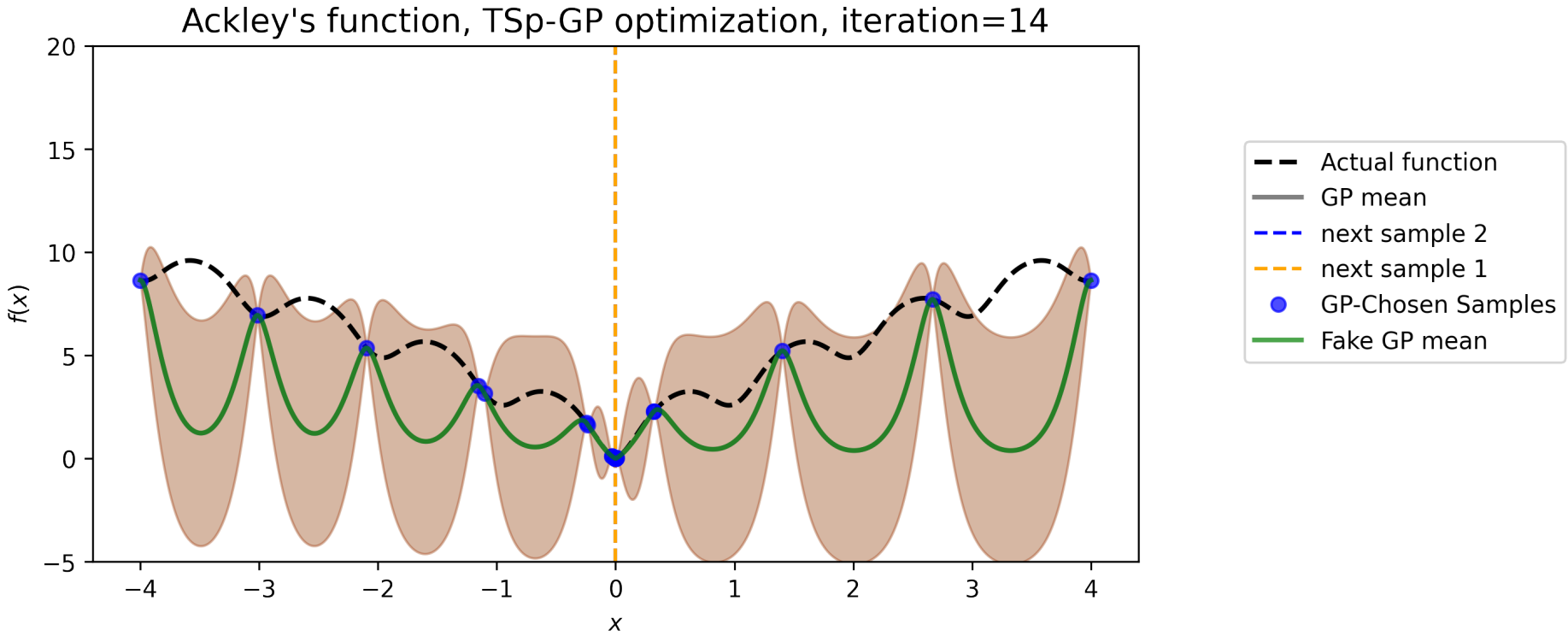


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

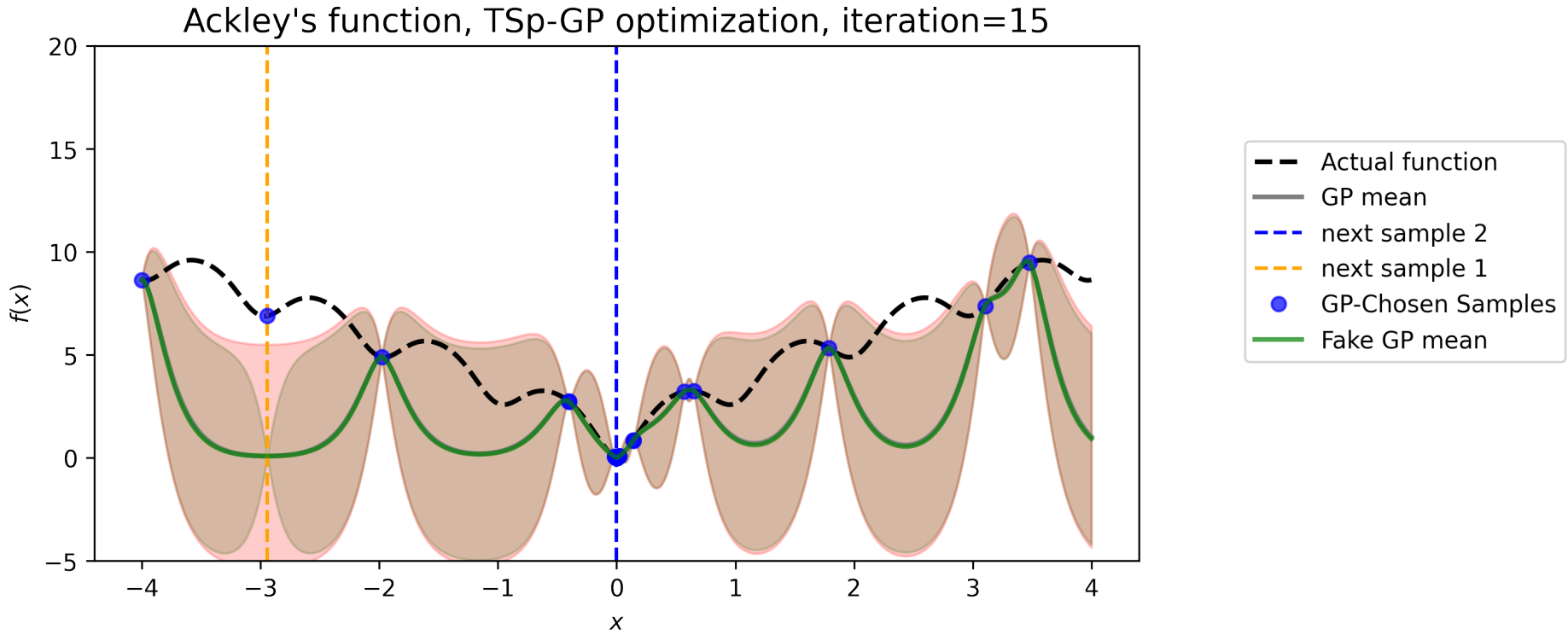


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

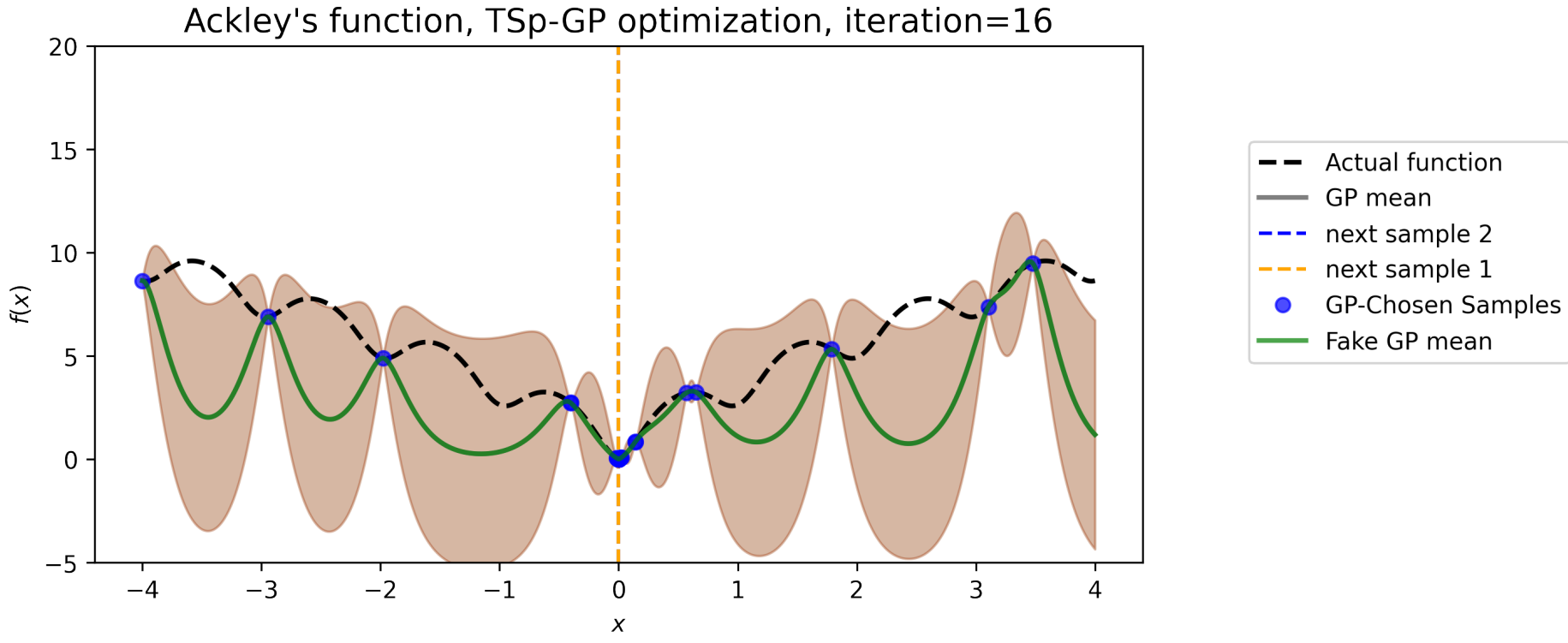


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

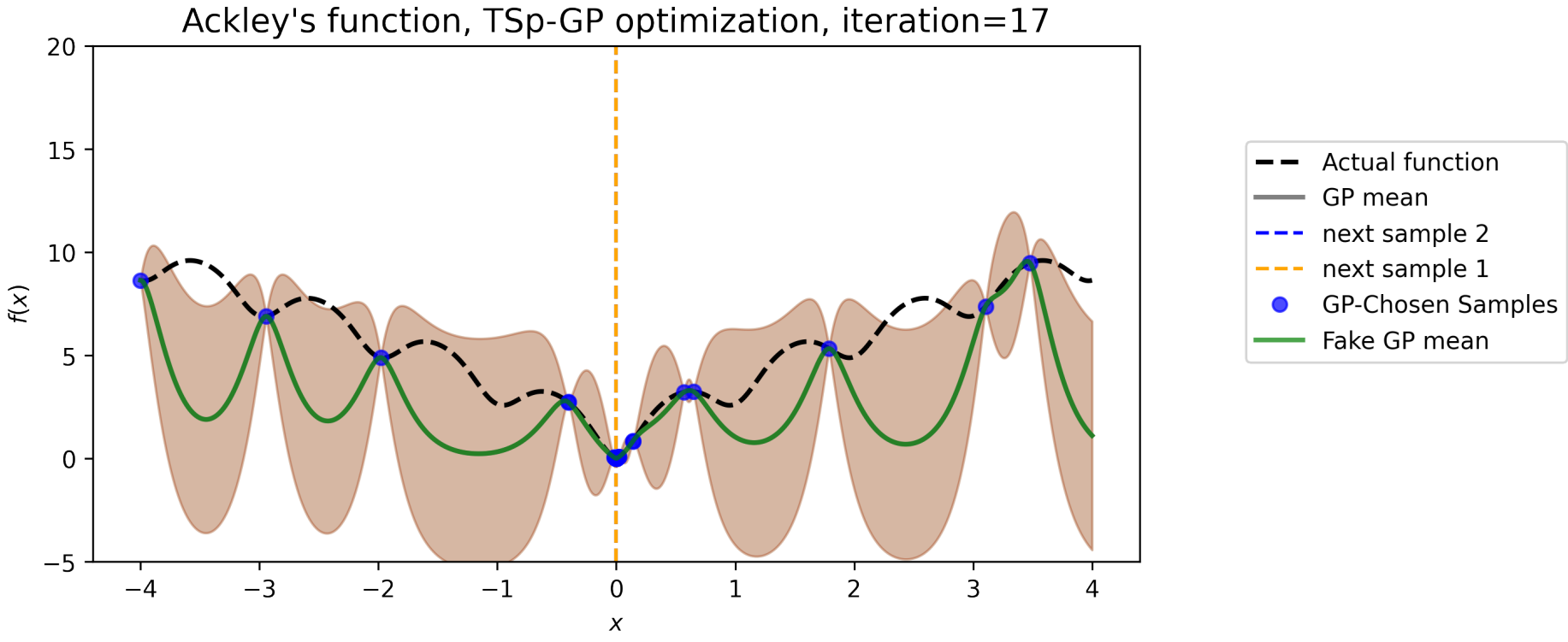


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Example of batch q-EI

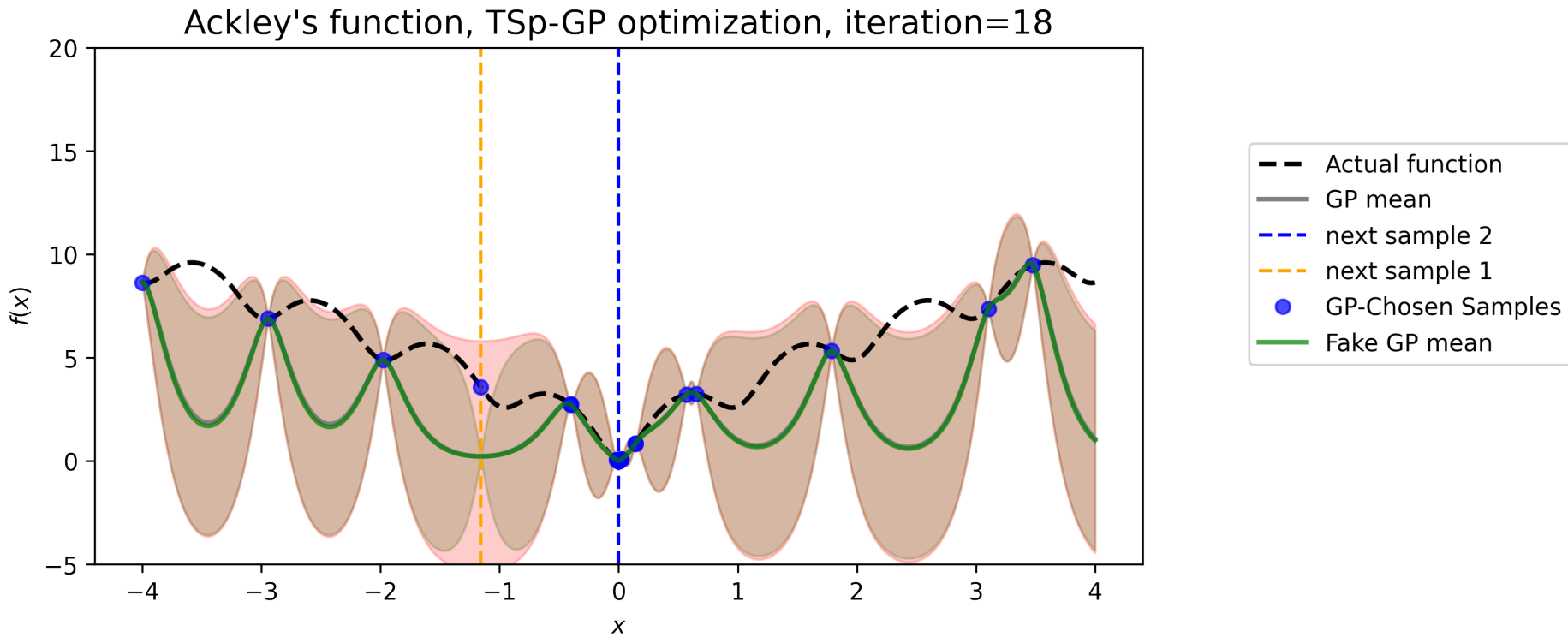


Figure: Iterations of a batch BO loop for a q-EI acquisition function

Modified (q-EI)

The problem with the original formulation is that the q candidate sample points are similar for several iterations, making the surrogate GP and the fake GP almost identical.
In [7], it is proposed:

$$H = \{(\mathbf{x}_1, \cdots, \mathbf{x}_q) : \mathbf{x}_i \in \mathcal{X}, ||\mathbf{x}_i - \mathbf{x}_j|| \geq r, ||\mathbf{x}_i - \mathbf{x}^\ell|| \geq r, i \neq j,$$

$1 \leq i, j \leq q, 1 \leq \ell \leq t$
The easiest way to include this modification is by penalising the acquisition function.

Example of Modified q-EI

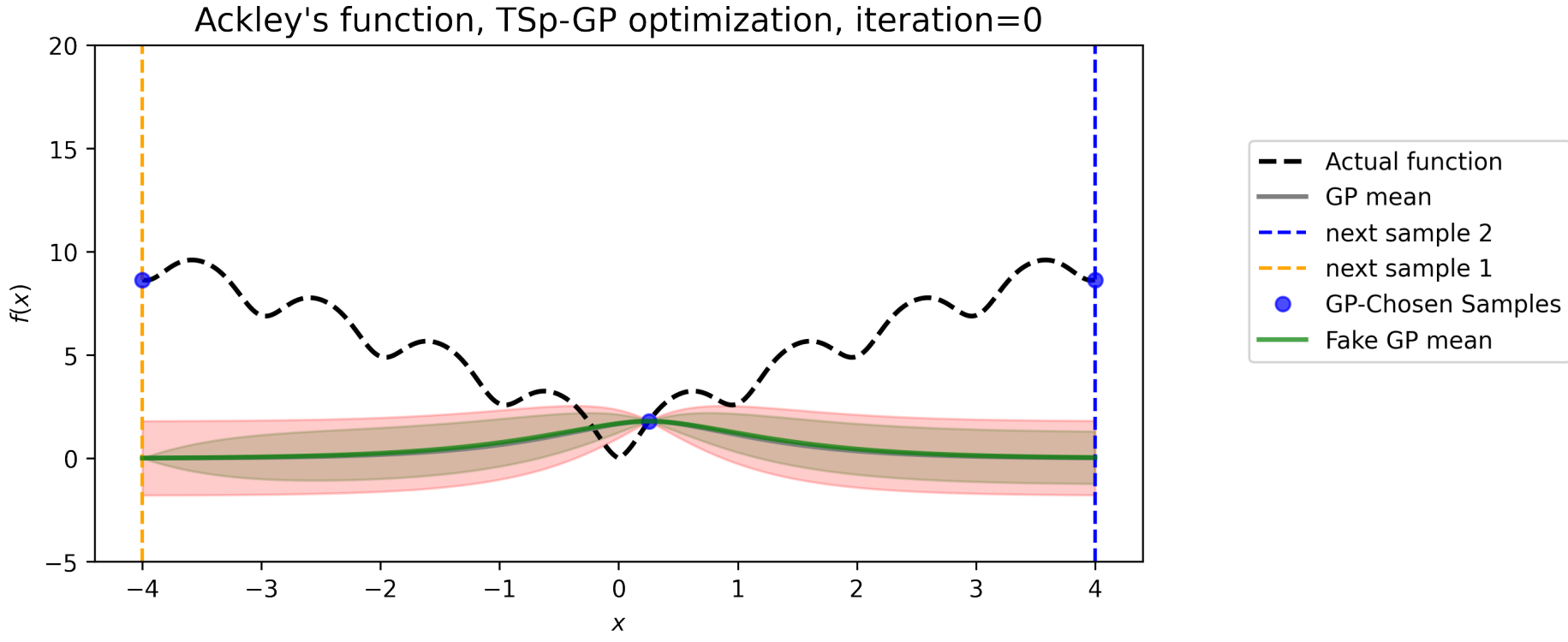


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

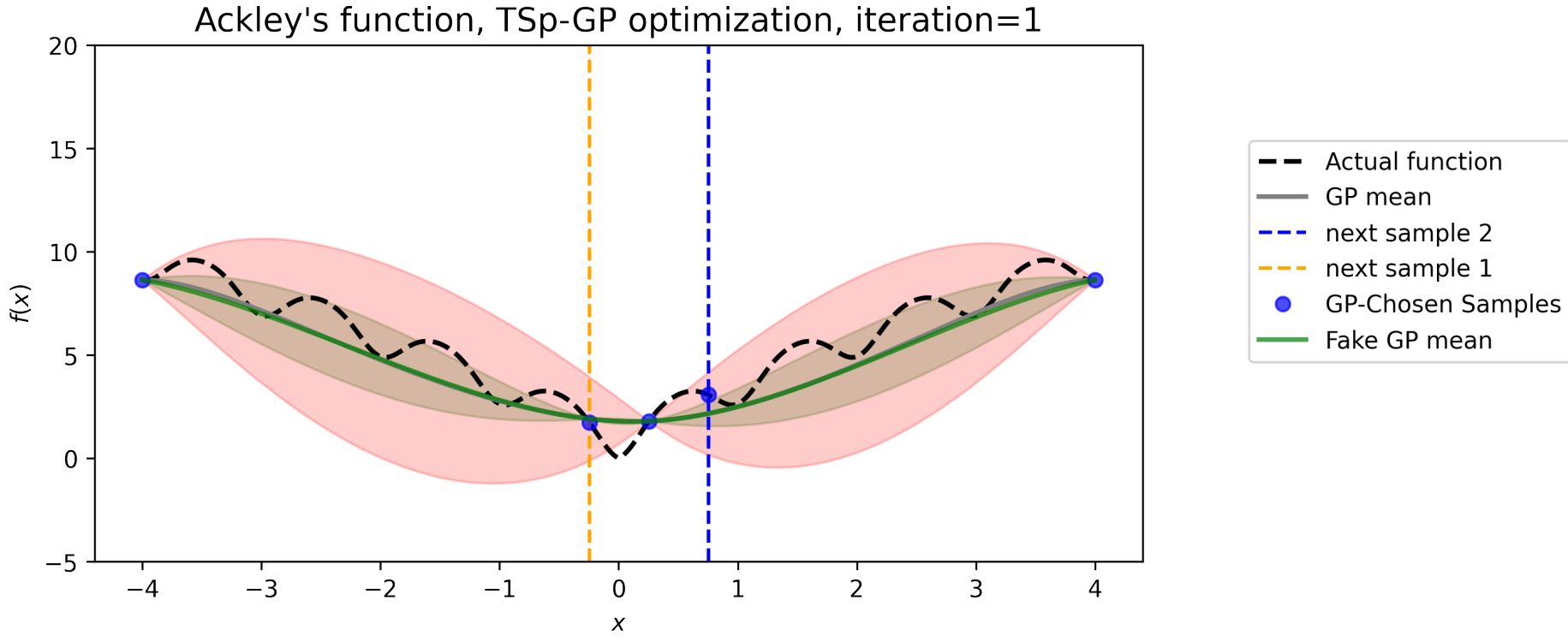


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

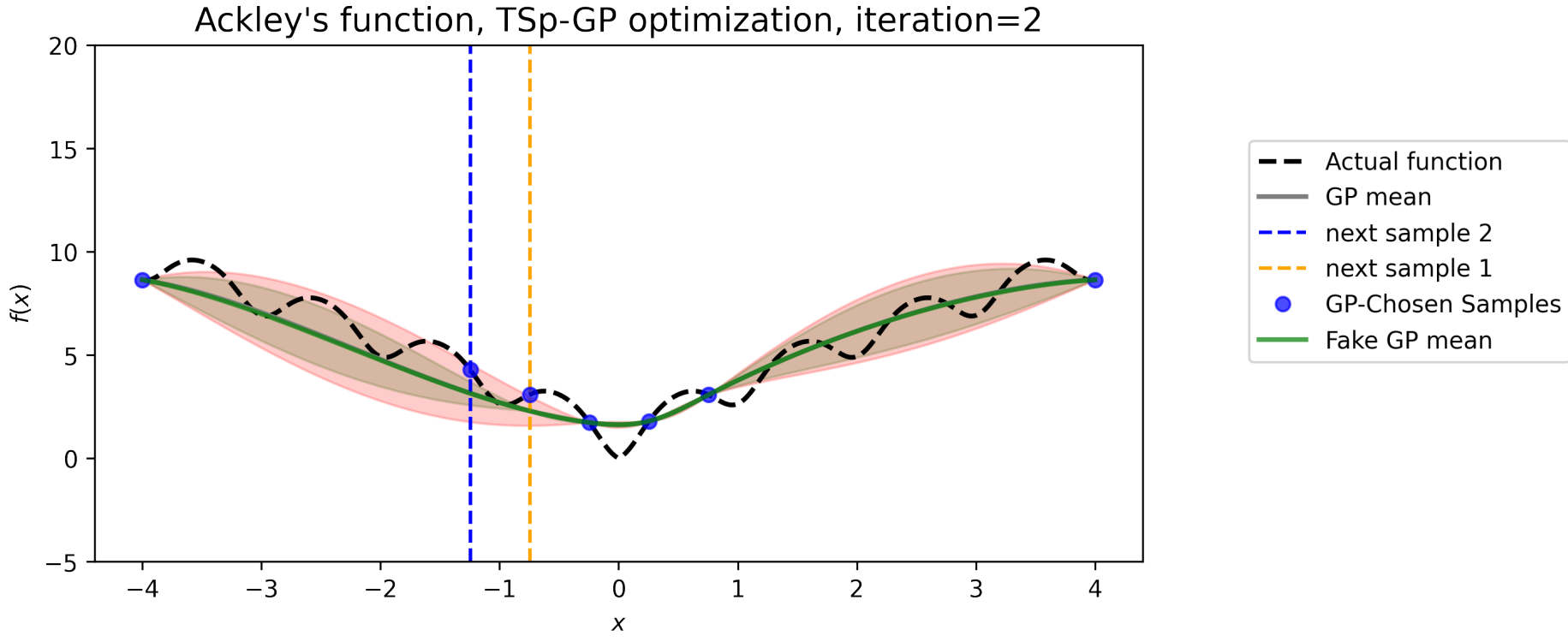


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

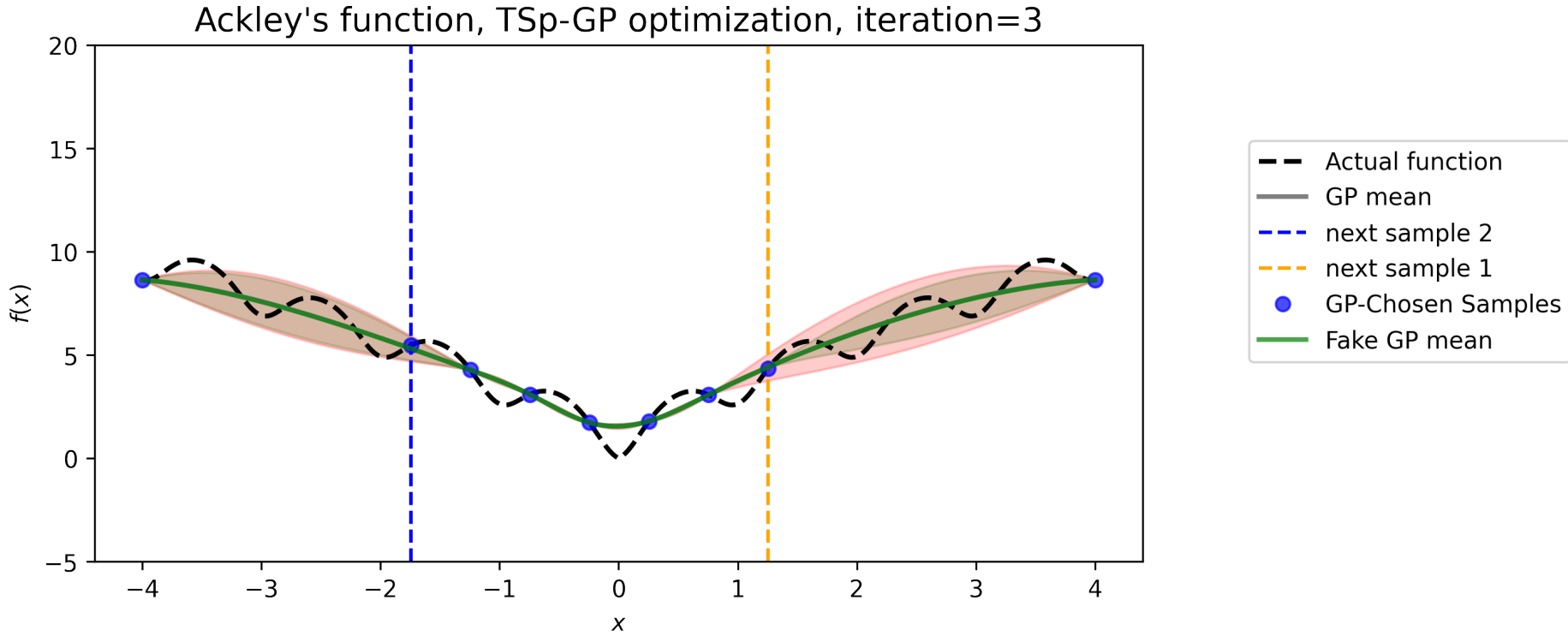


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

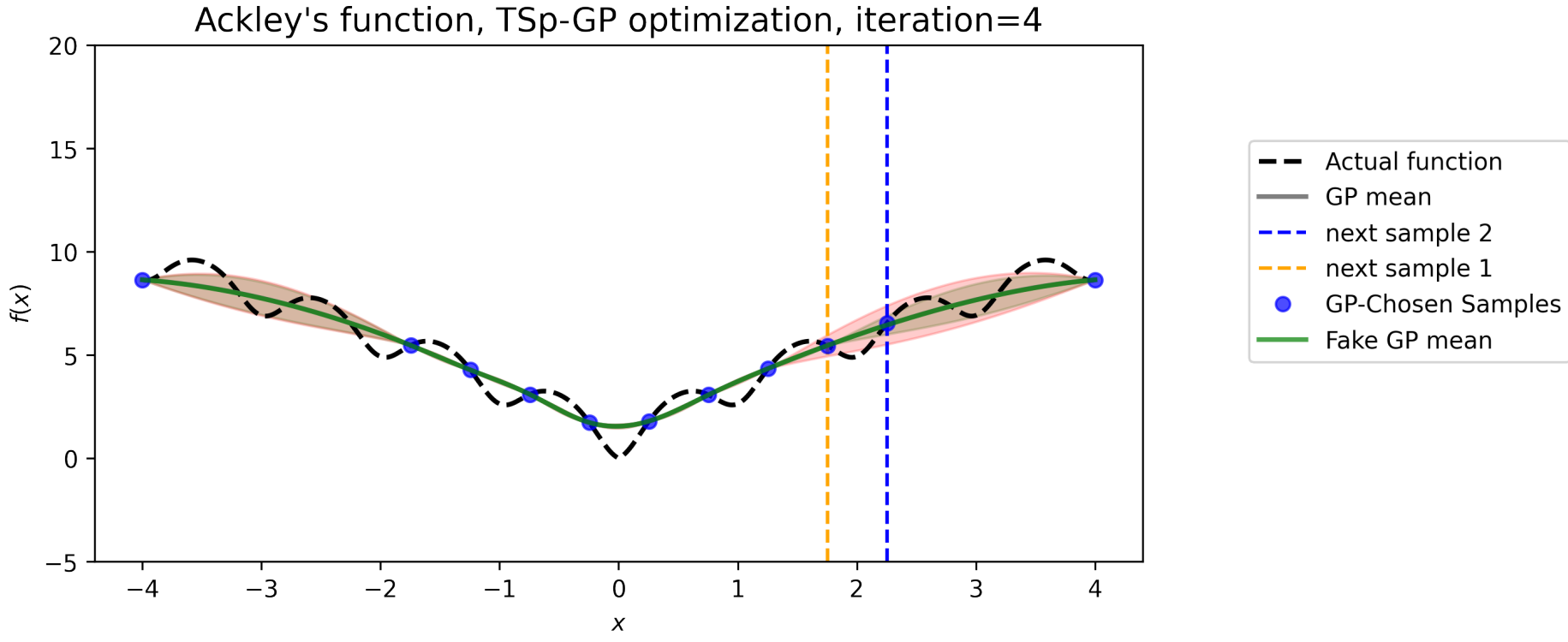


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

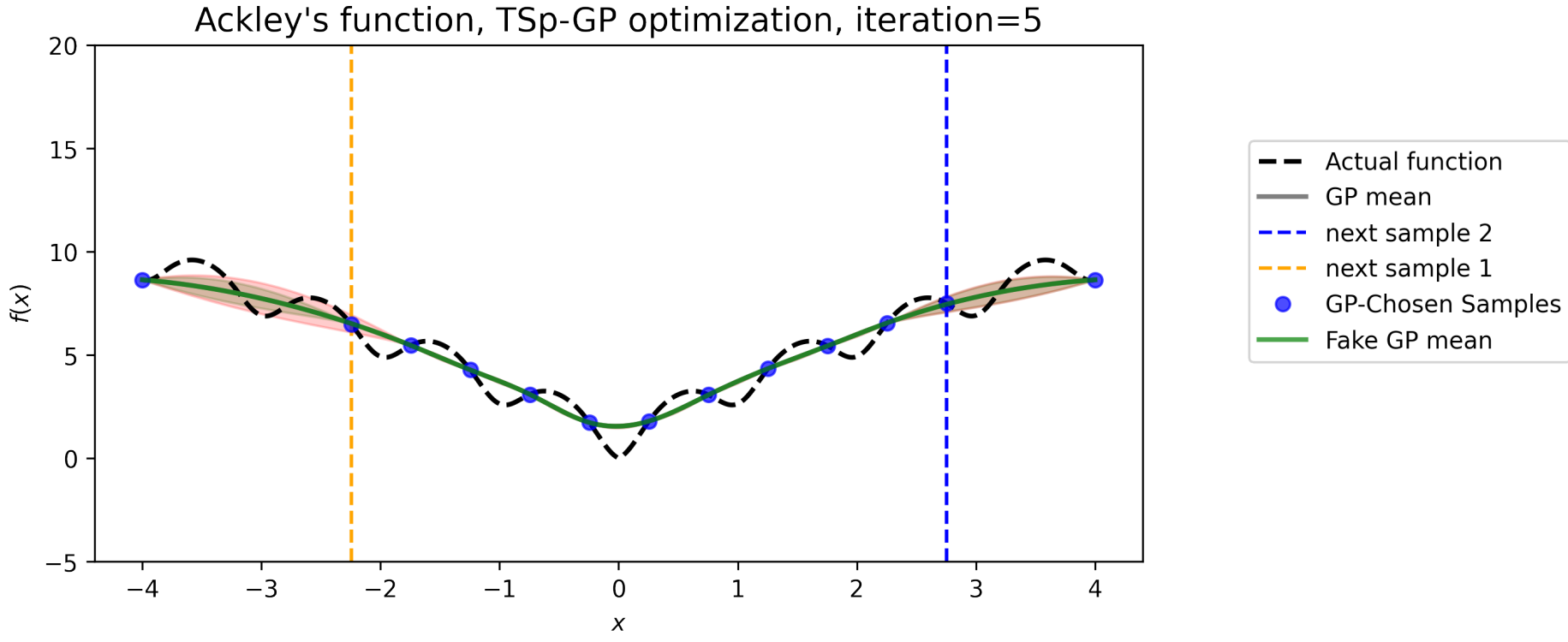


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

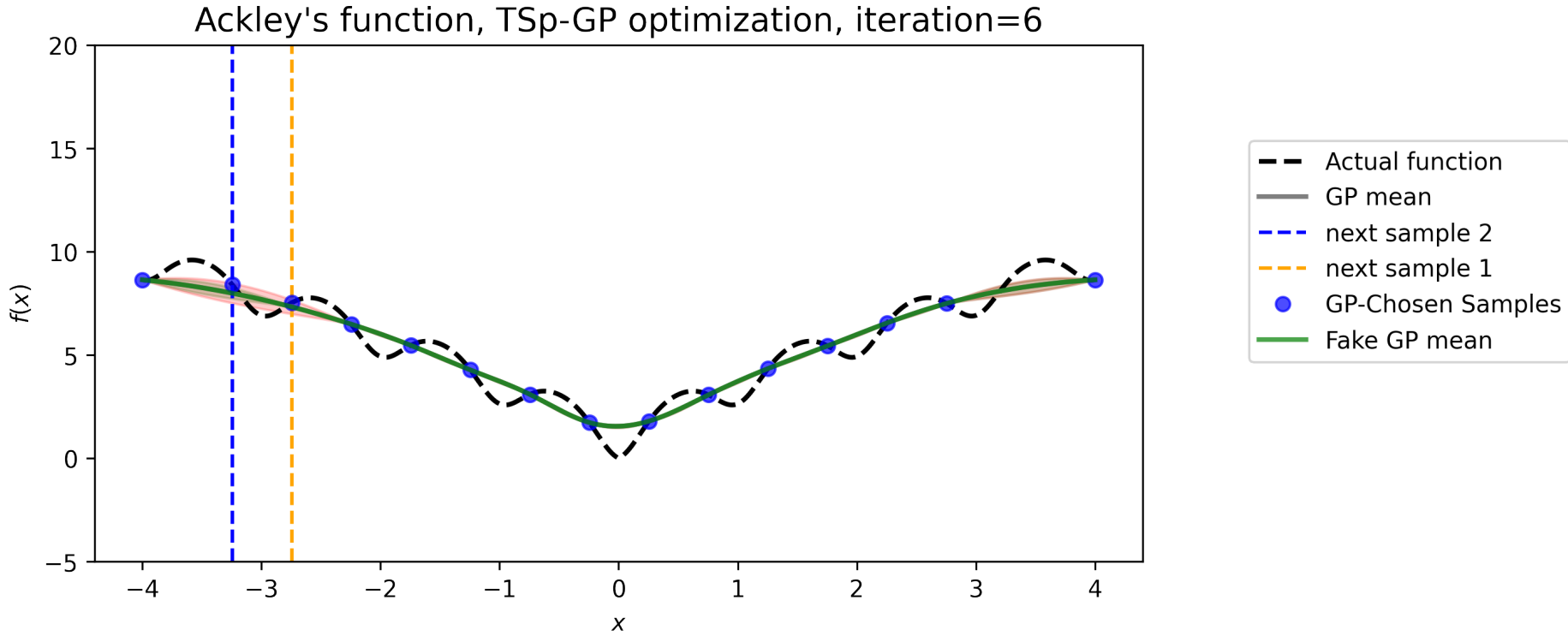


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

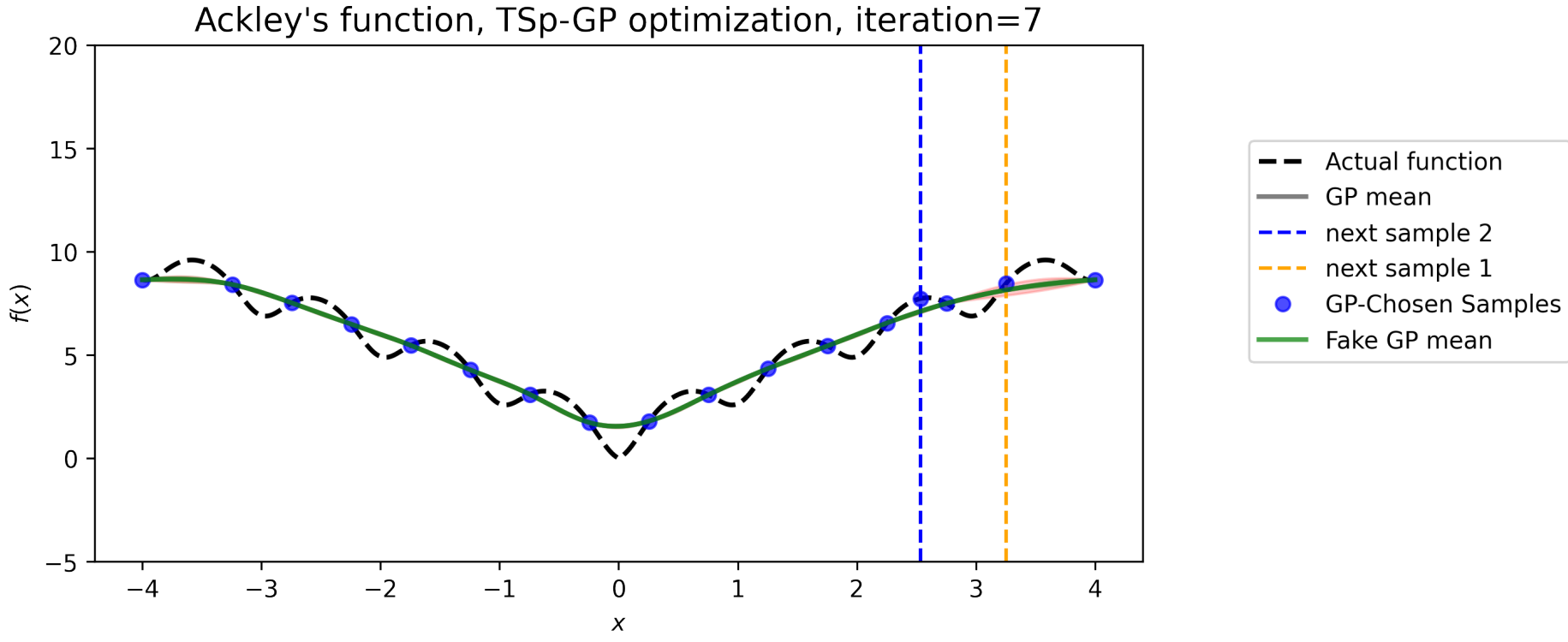


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

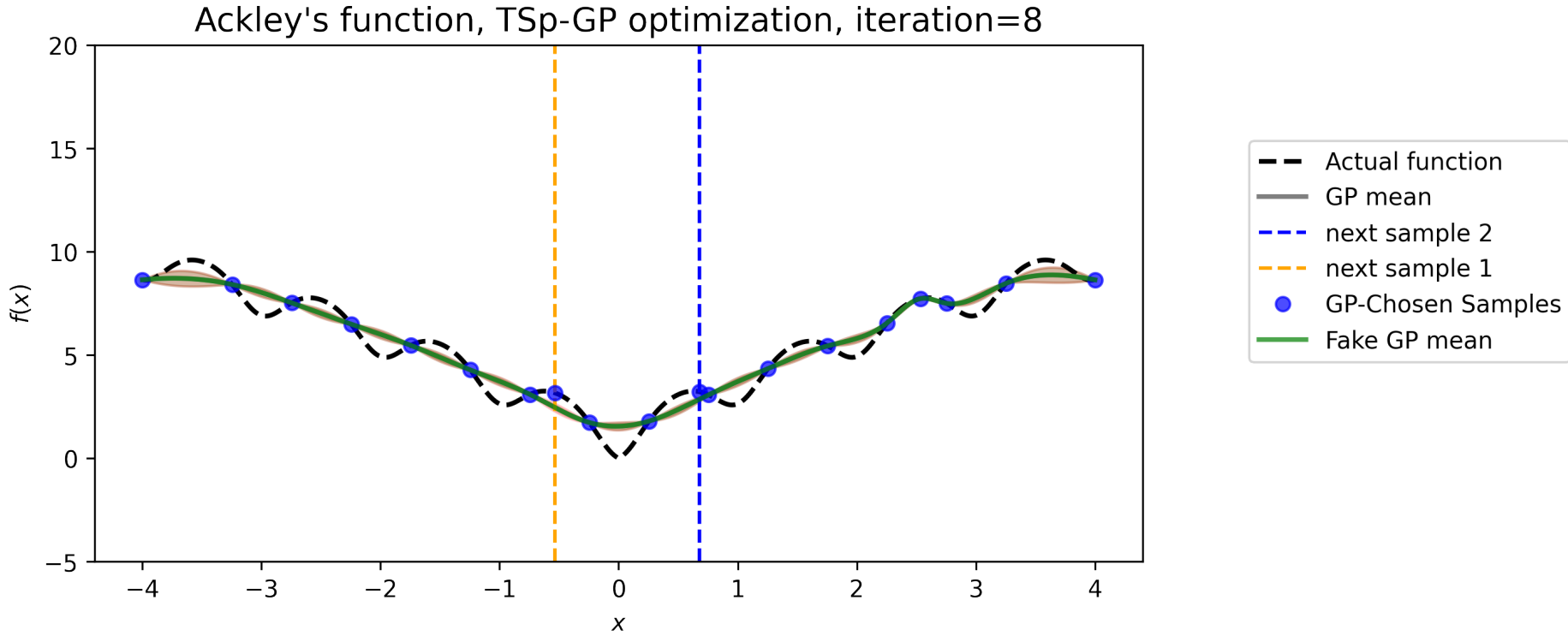


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

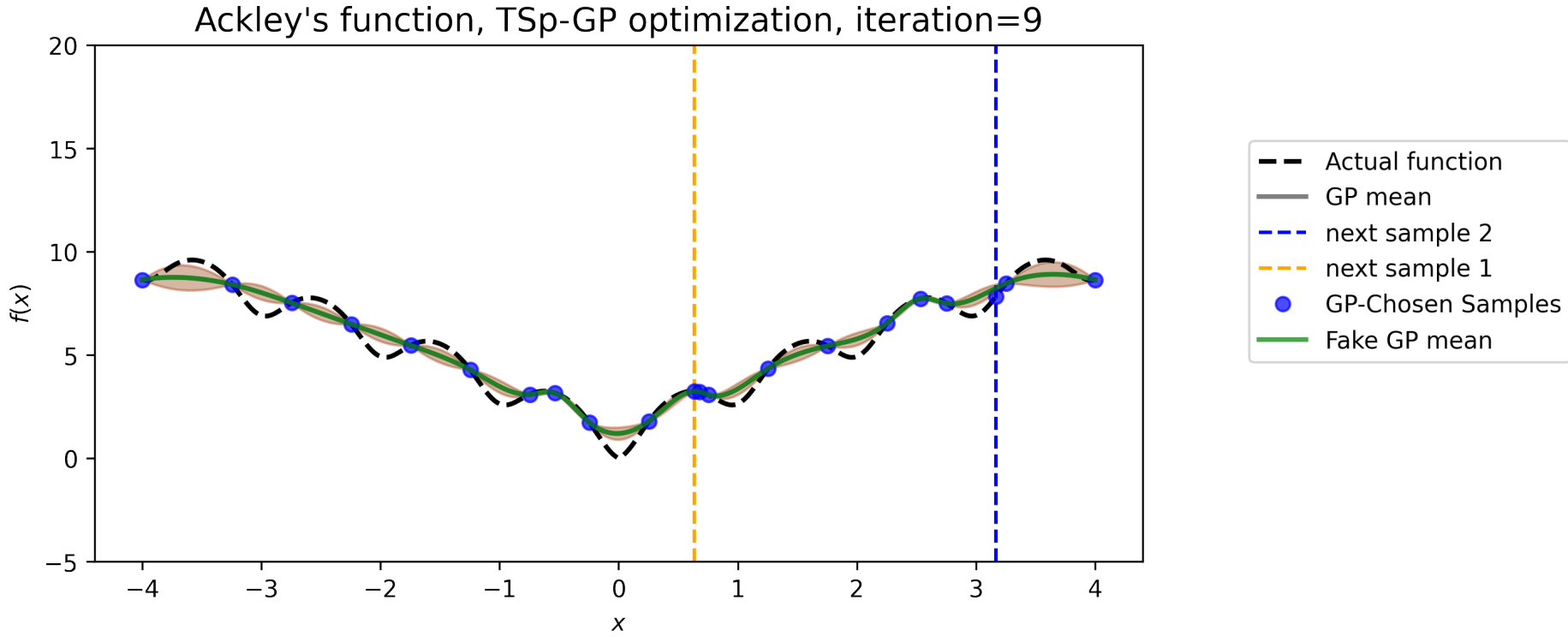


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

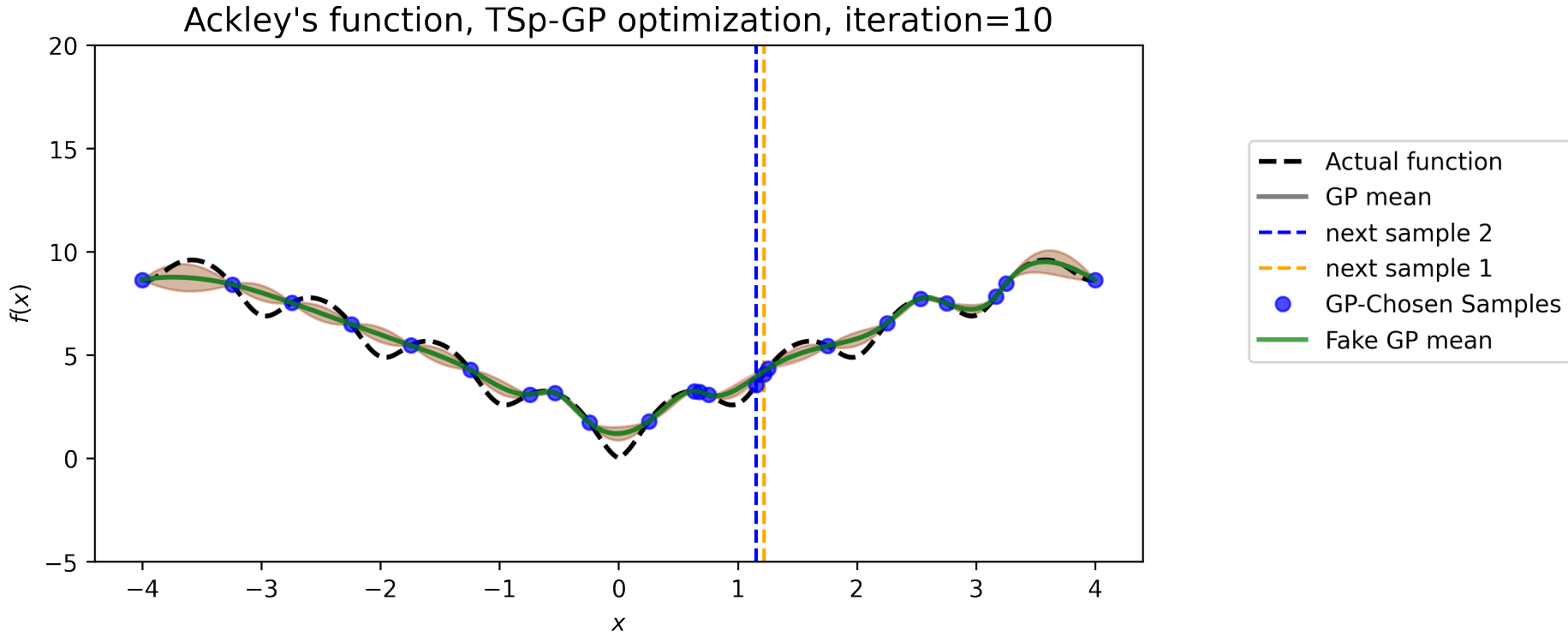


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

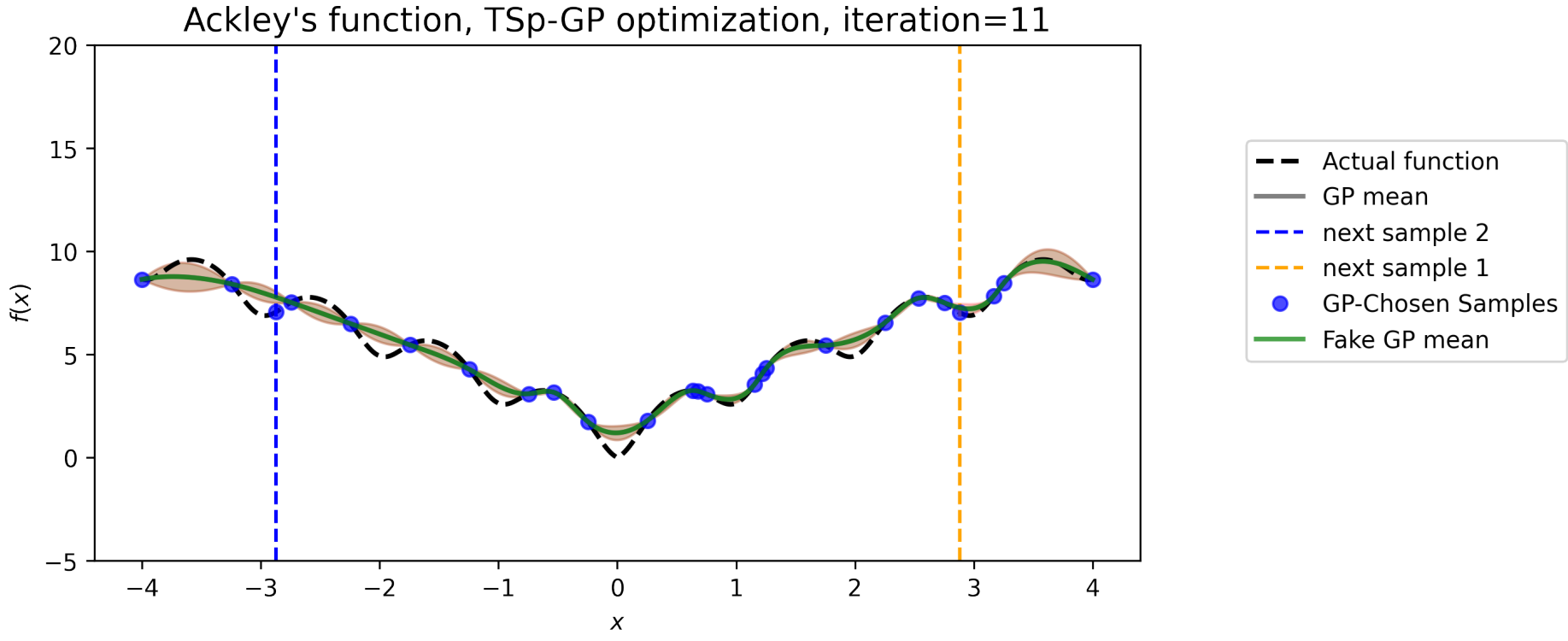


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

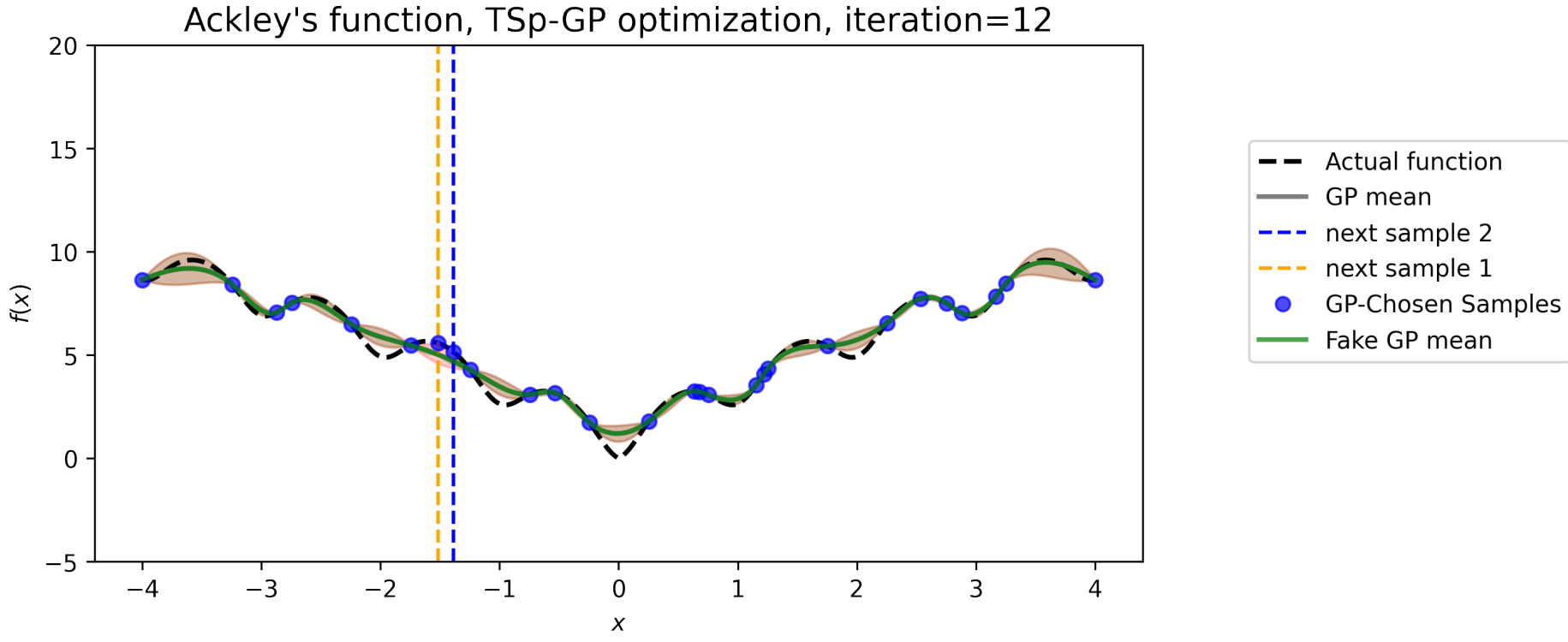


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

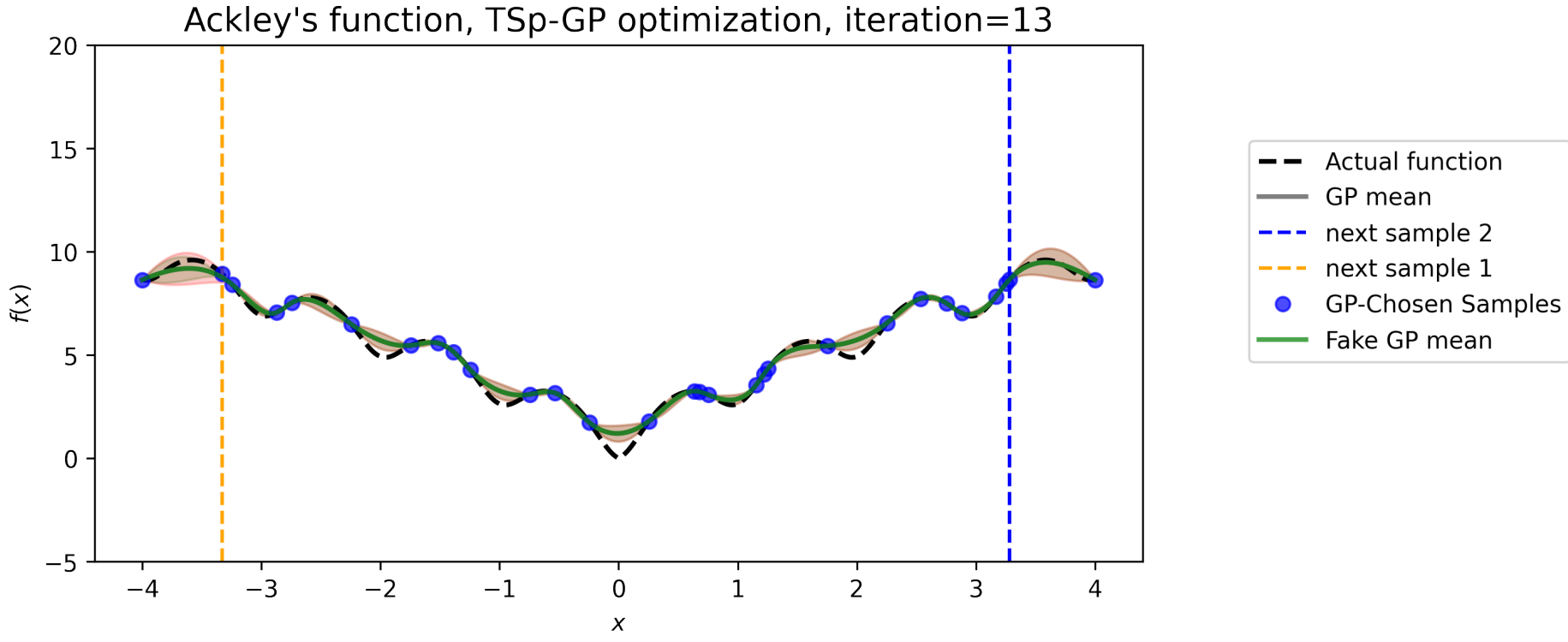


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

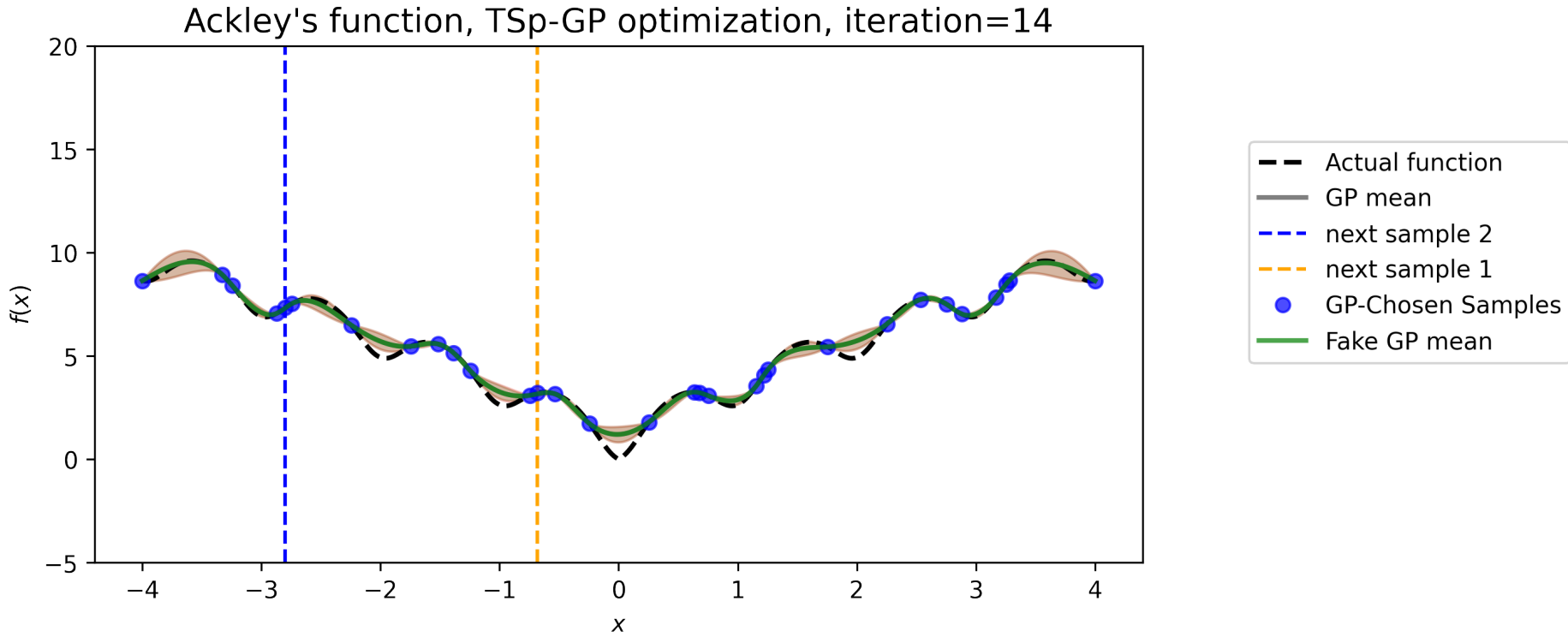


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 0.5$

Example of Modified q-EI

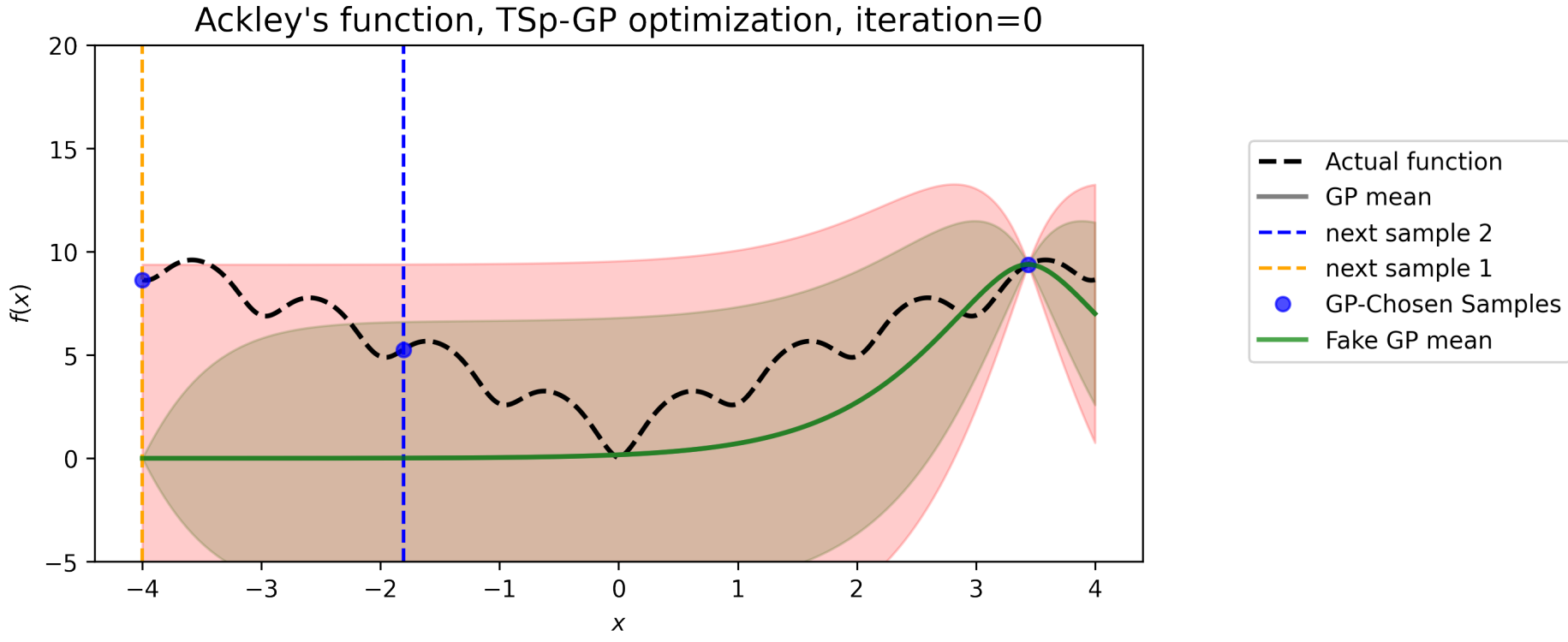


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

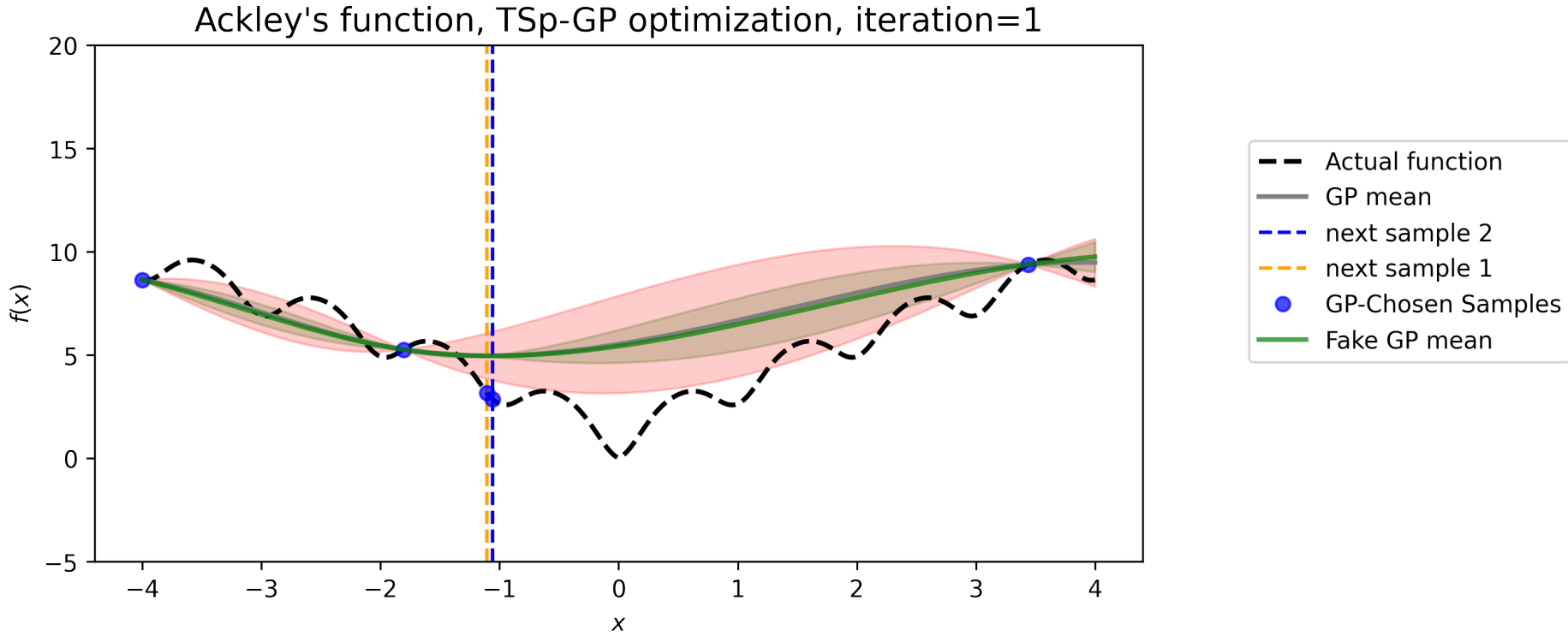


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

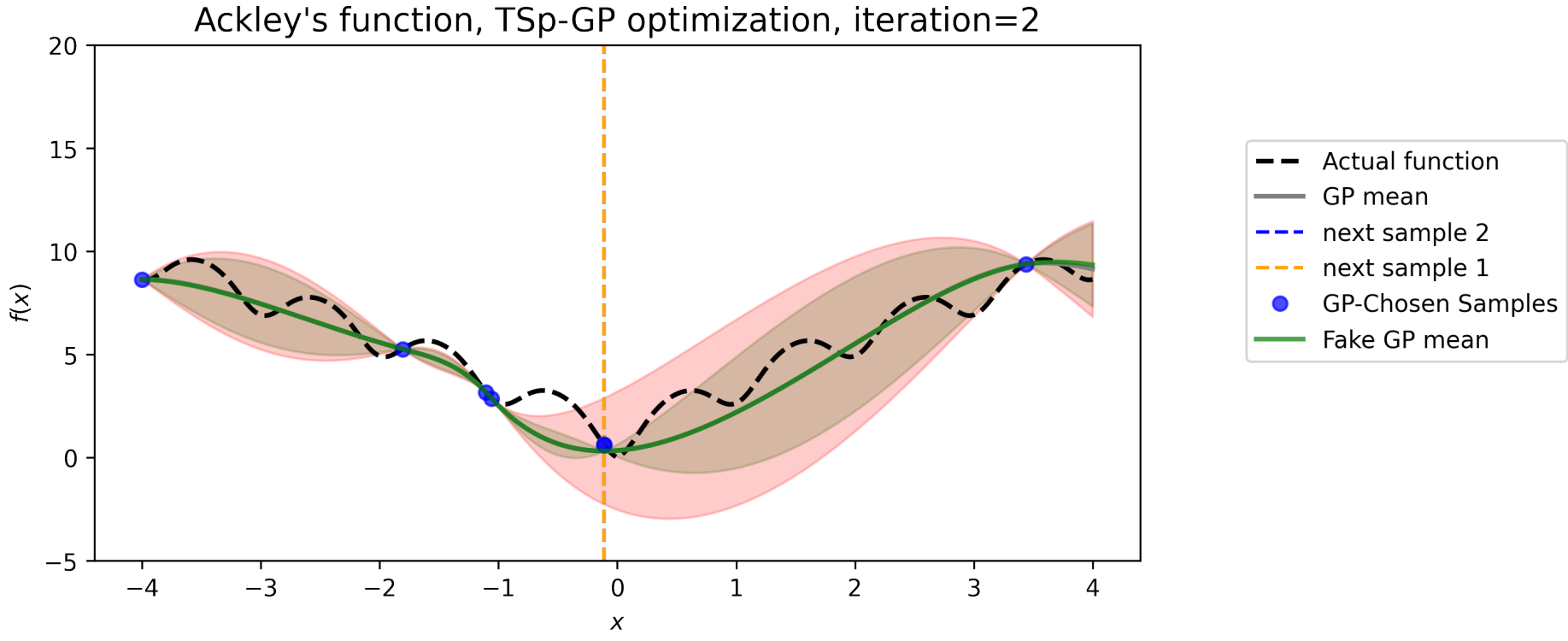


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

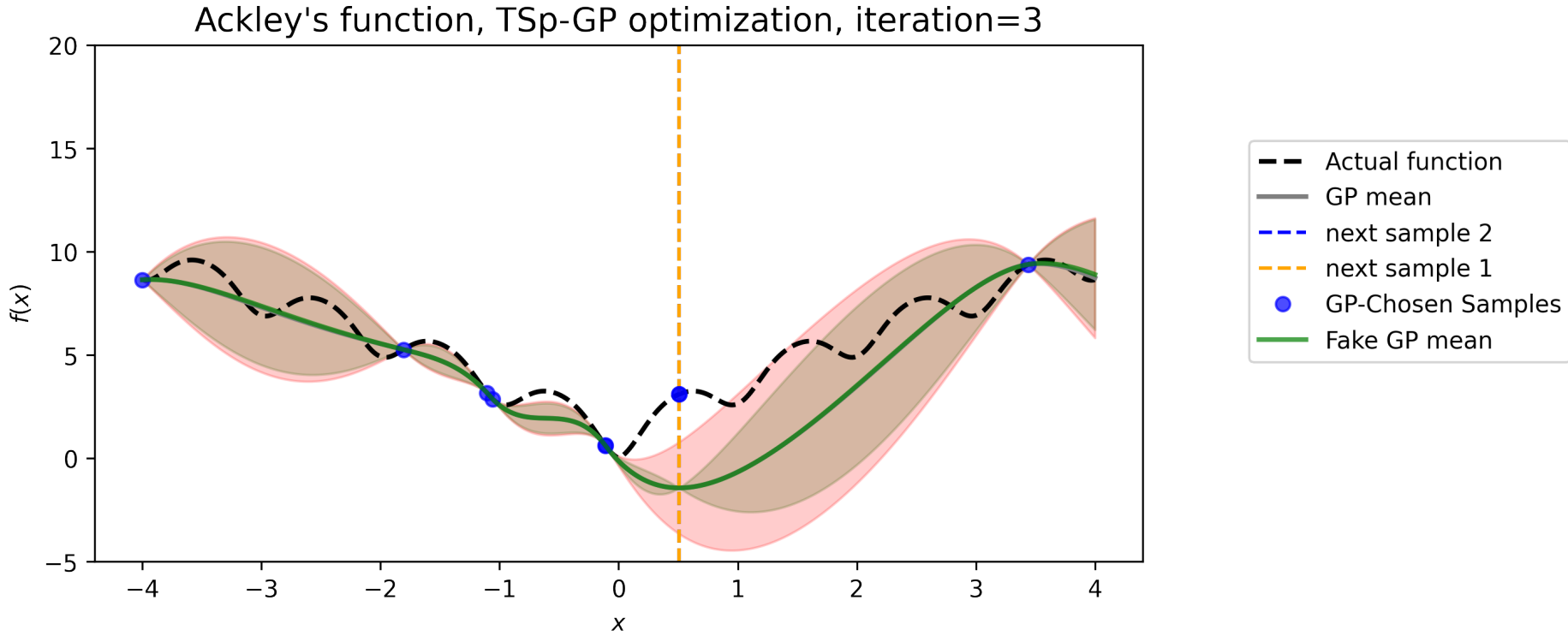


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

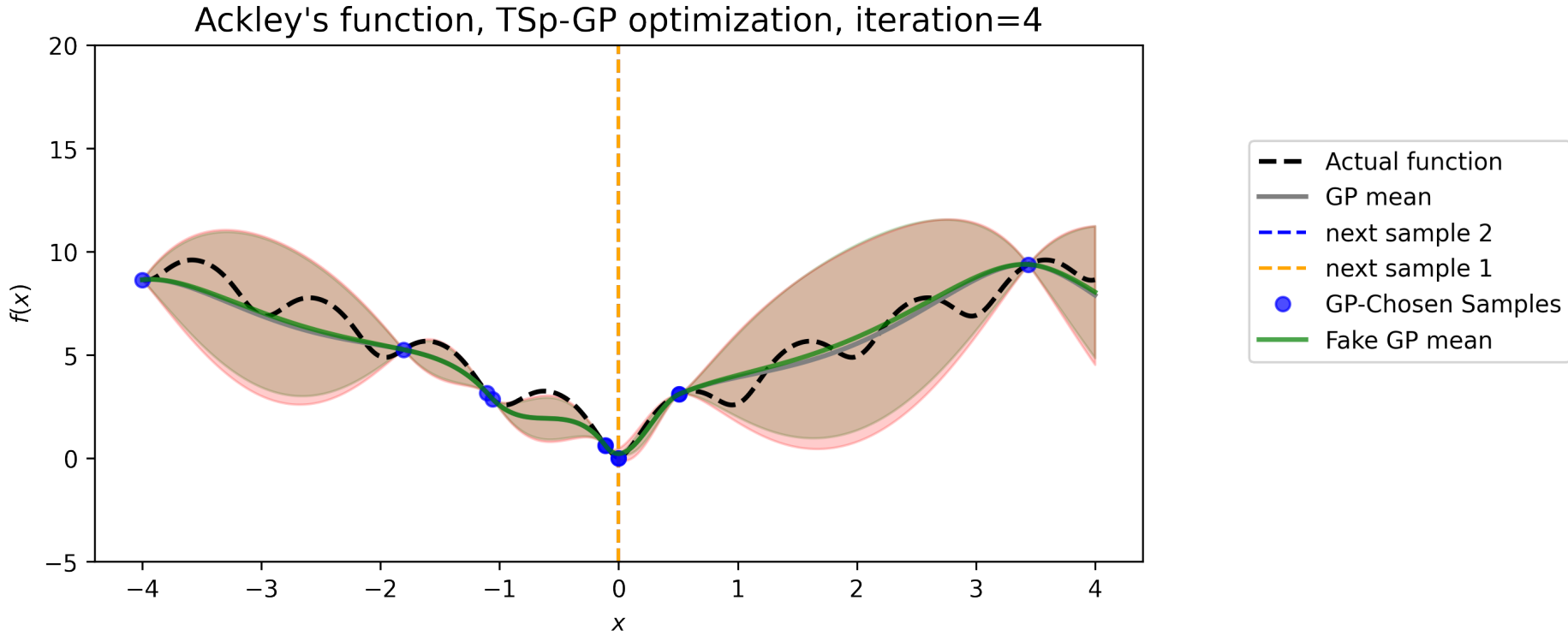


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

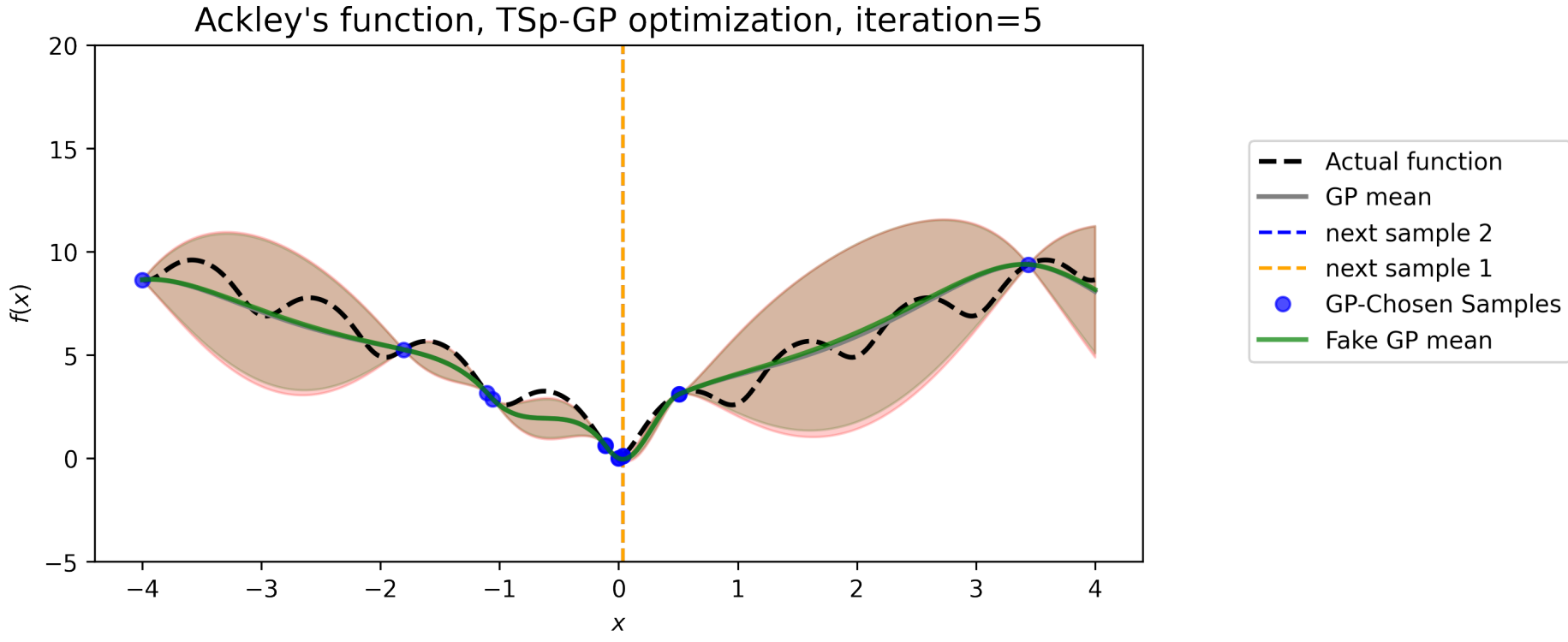


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

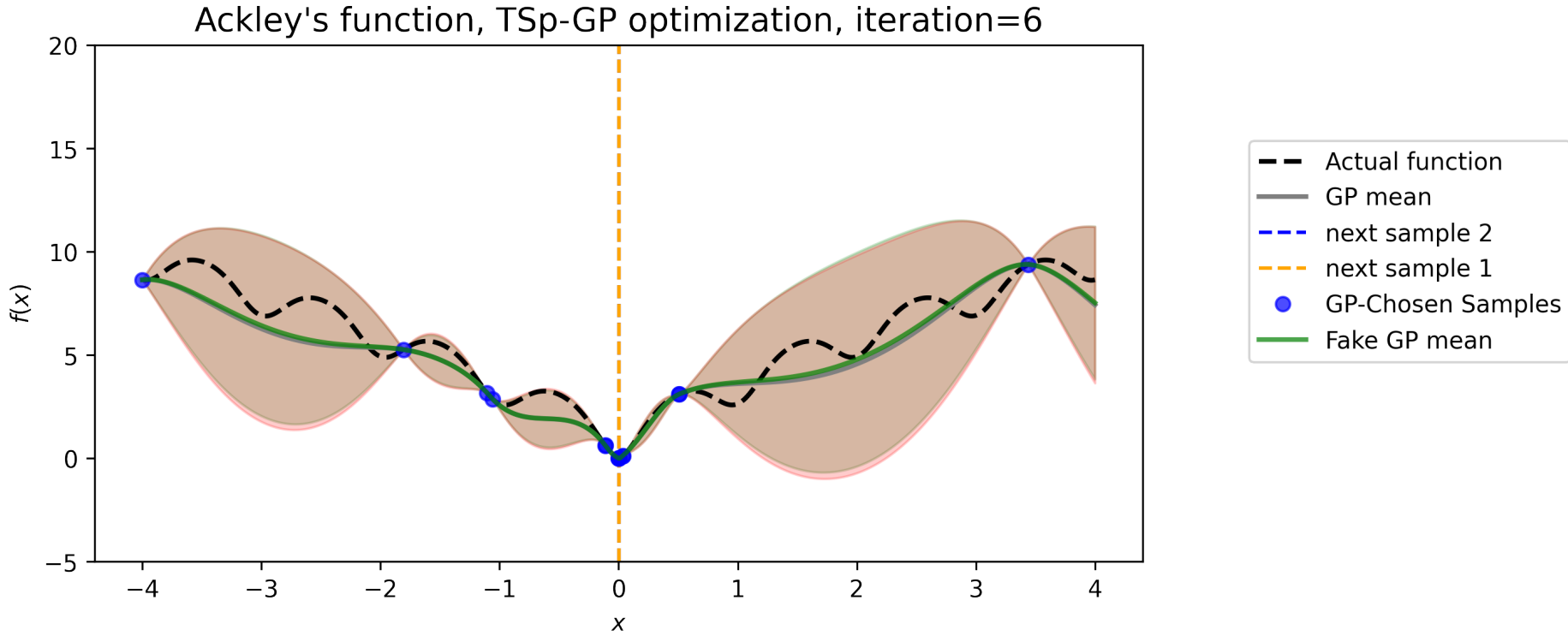


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

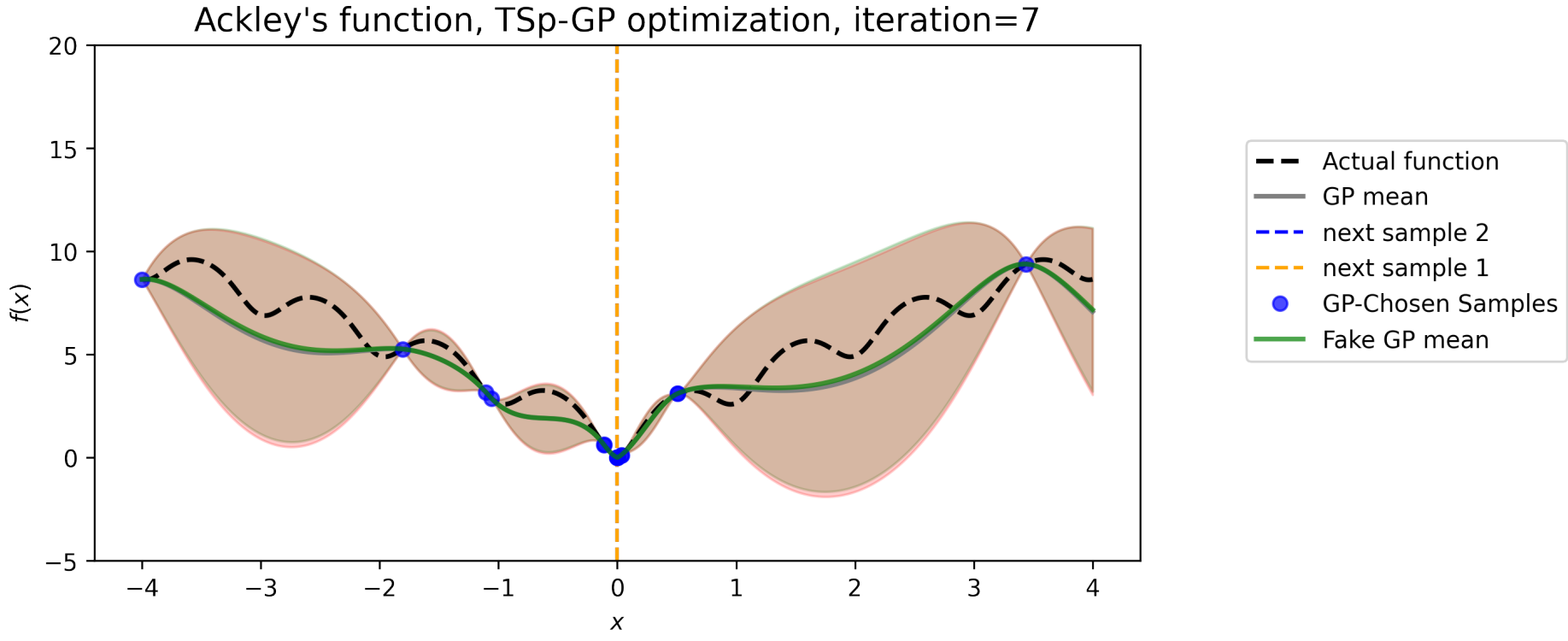


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

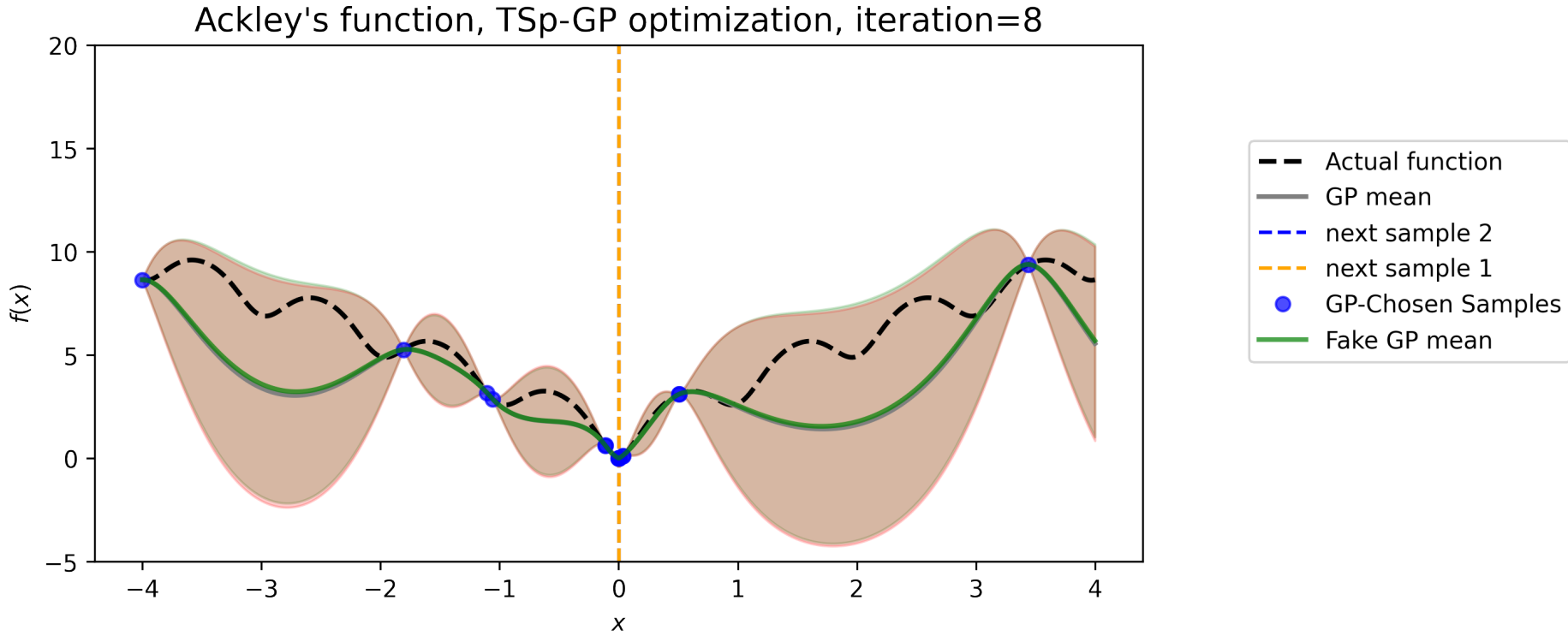


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

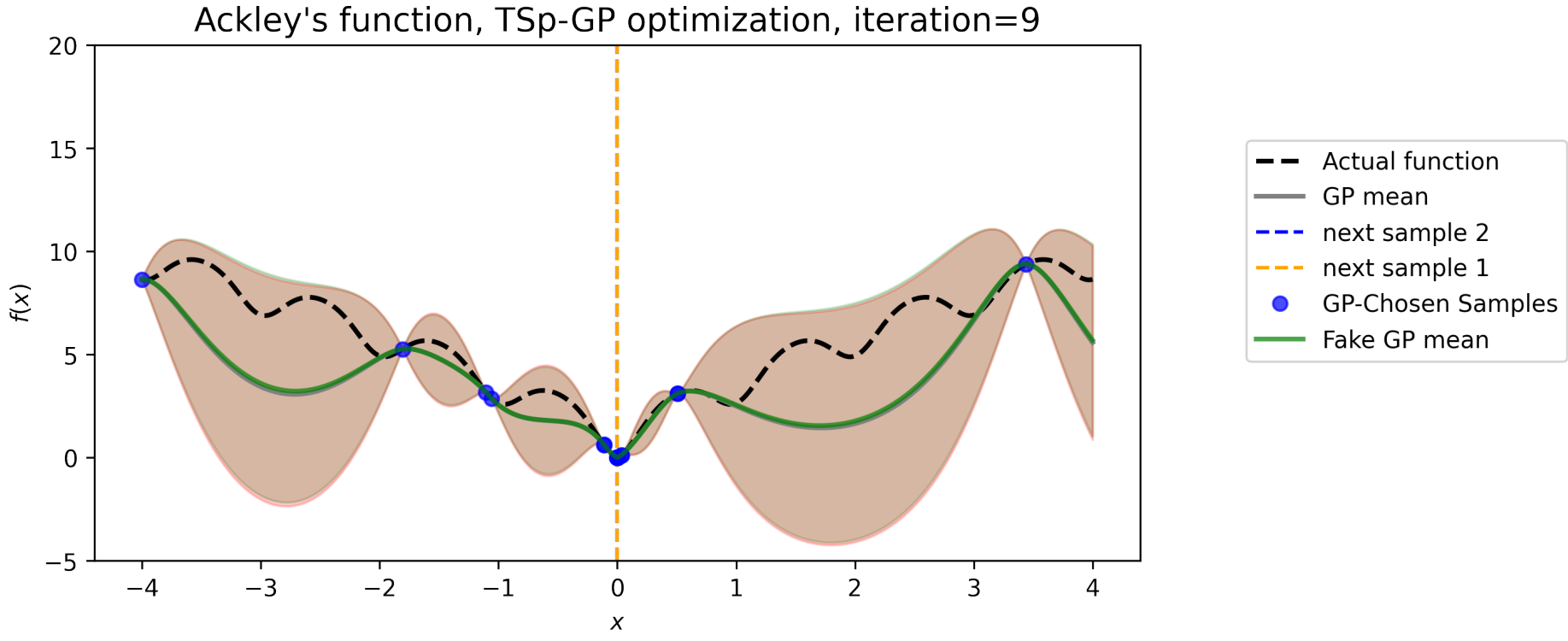


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

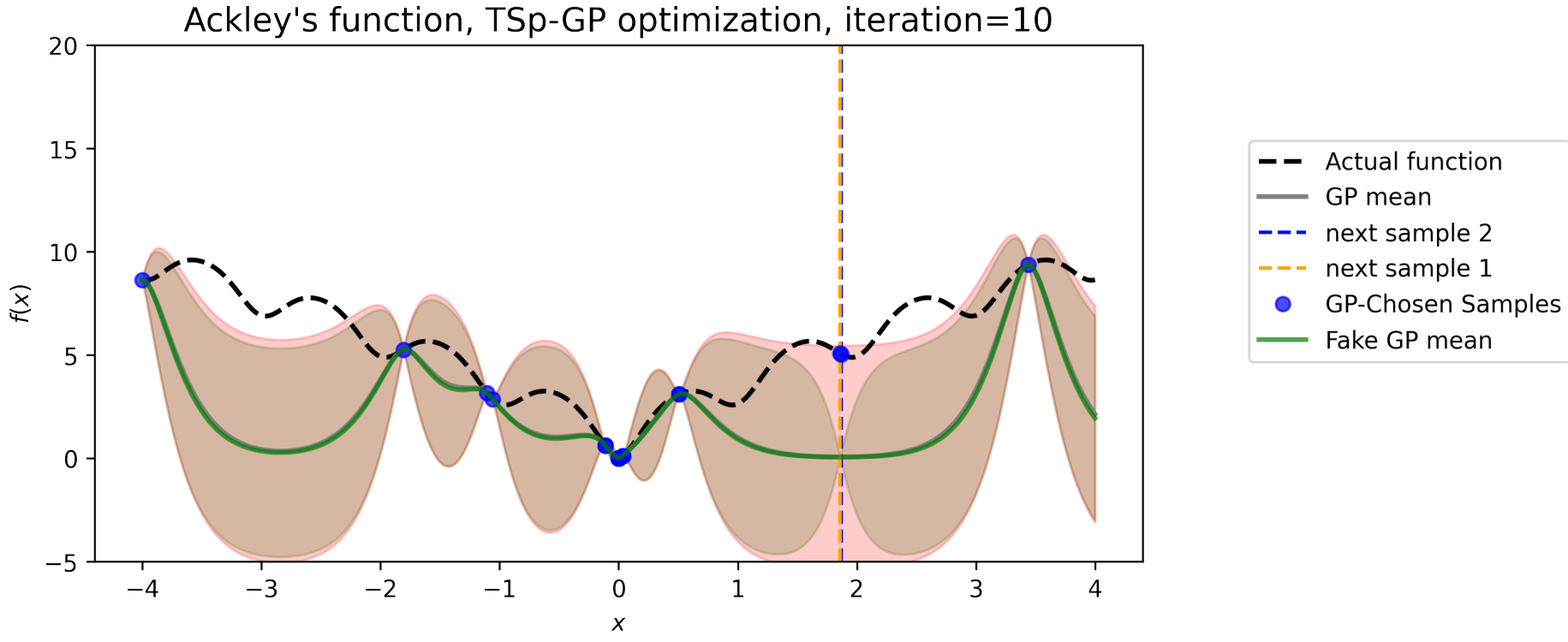


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

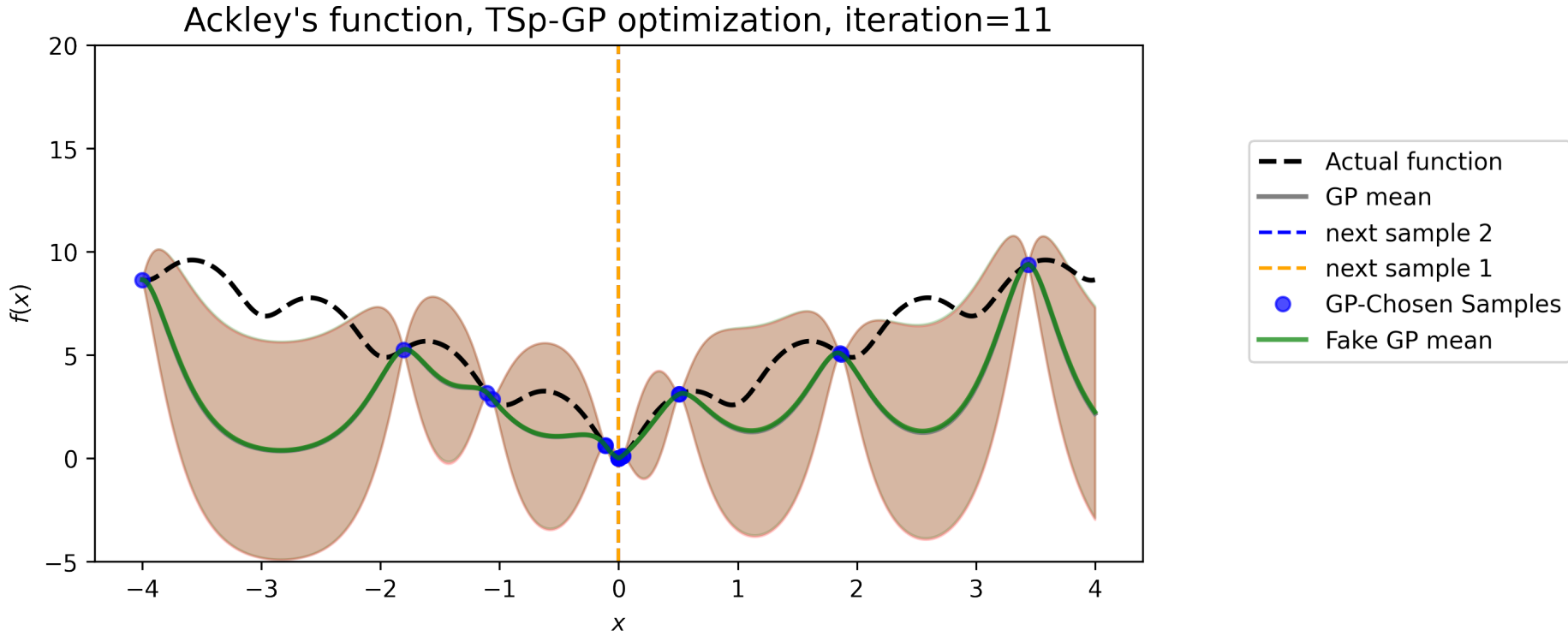


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

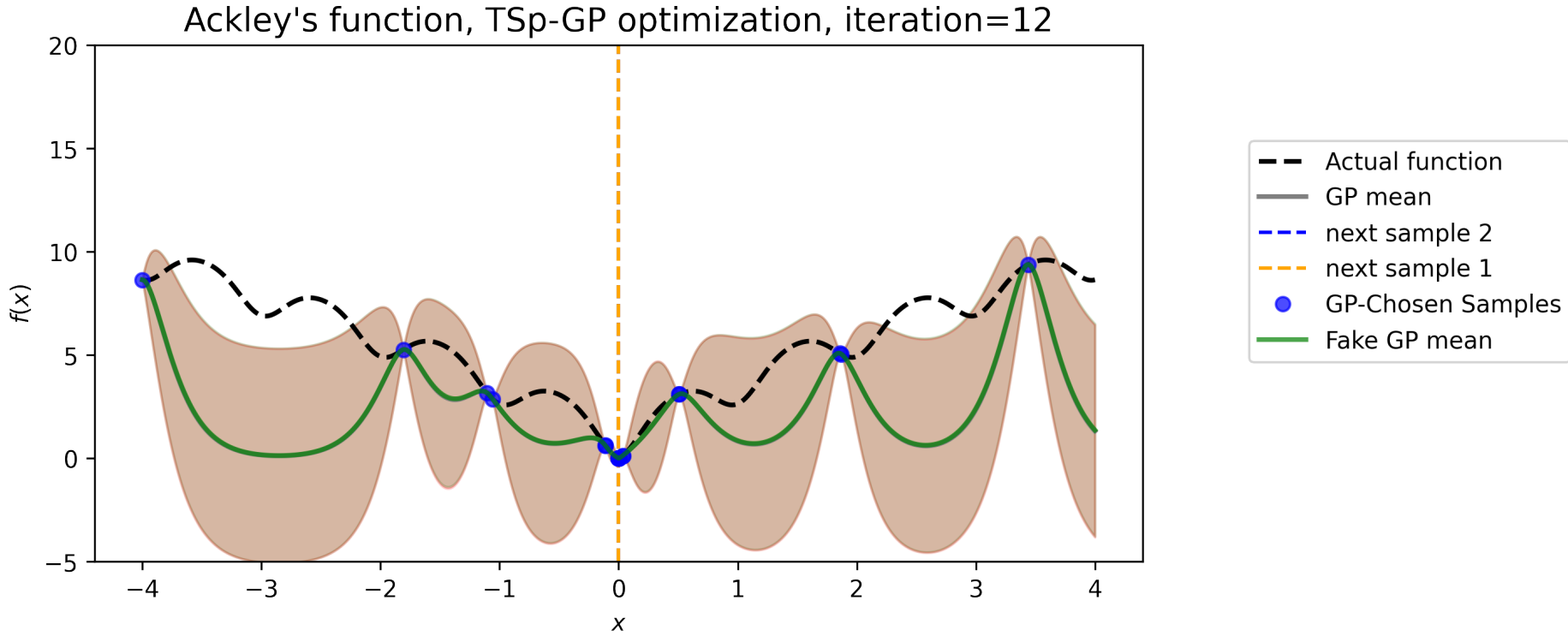


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

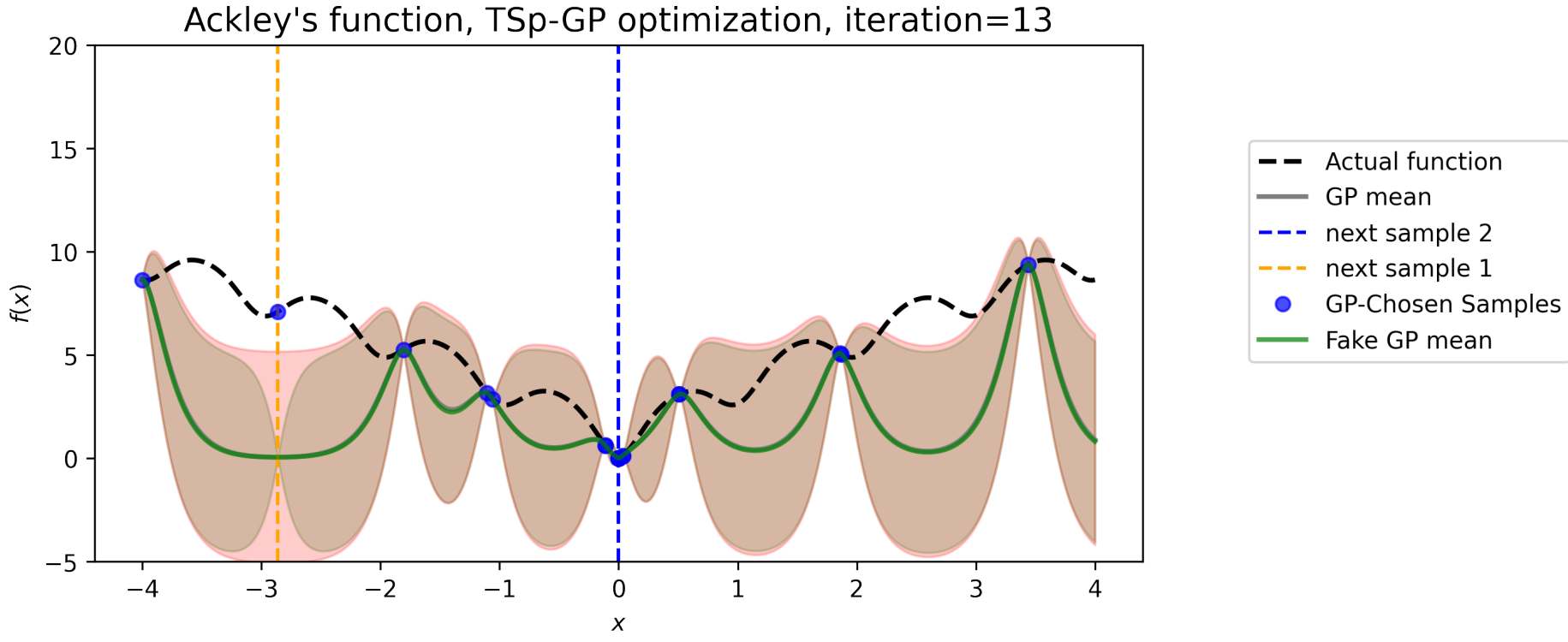


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Example of Modified q-EI

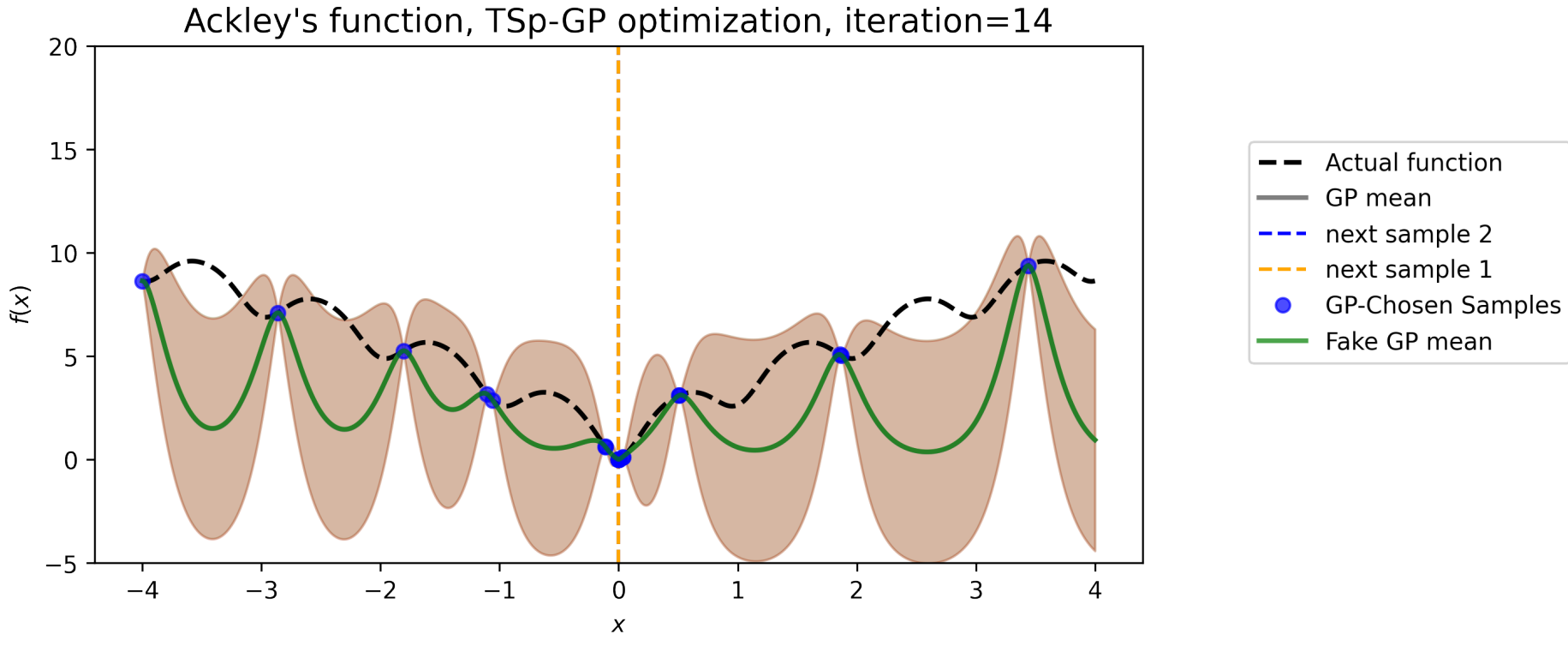


Figure: Iterations of a batch BO loop for a q-EI acquisition function and $r = 10^{-5}$

Parallel Bayesian Global Optimization of Expensive Functions*

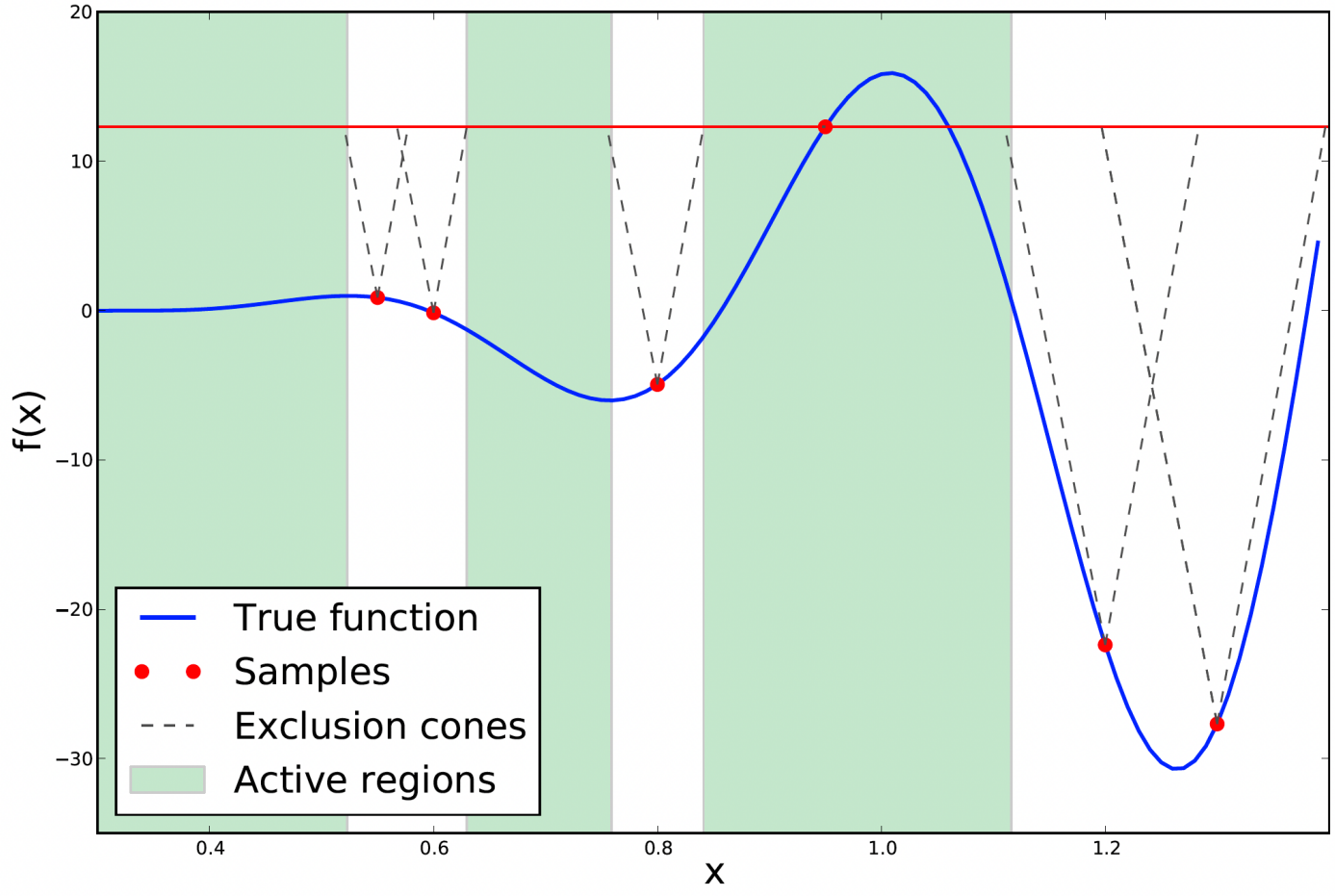
Jialei Wang[†1], Scott C. Clark[‡2], Eric Liu[§3], and Peter I. Frazier[¶1]

[1]School of Operations Research and Information Engineering, Cornell University
[2]SigOpt, 244 Kearny St, San Francisco, CA
[3]Yelp, Inc., 140 New Montgomery, San Francisco, CA

5 May 2019

Batch Bayesian Optimization via Local Penalisation

$$\mathbf{x}_{t,k} = \arg\max_{\mathbf{x}\in\mathcal{X}} \left\{ g(\alpha(\mathbf{x}; \mathcal{I}_{t,0})) \prod_{j=1}^{k-1} \varphi(\mathbf{x}; \mathbf{x}_{t,j}) \right\}$$



Local Penalisation strategy

$g(z) = z$ if $\alpha(\mathbf{x})$ is positive, and $g(z) = \ln(1 + e^z)$ elsewhere. It is a differentiable transformation that keeps the objective strictly positive without changing the location of its extrema. $\varphi(\mathbf{x}; \mathbf{x}_{t,j}) = 1 - p(\mathbf{x} \in B_{r_j}(\mathbf{x}_j))$.

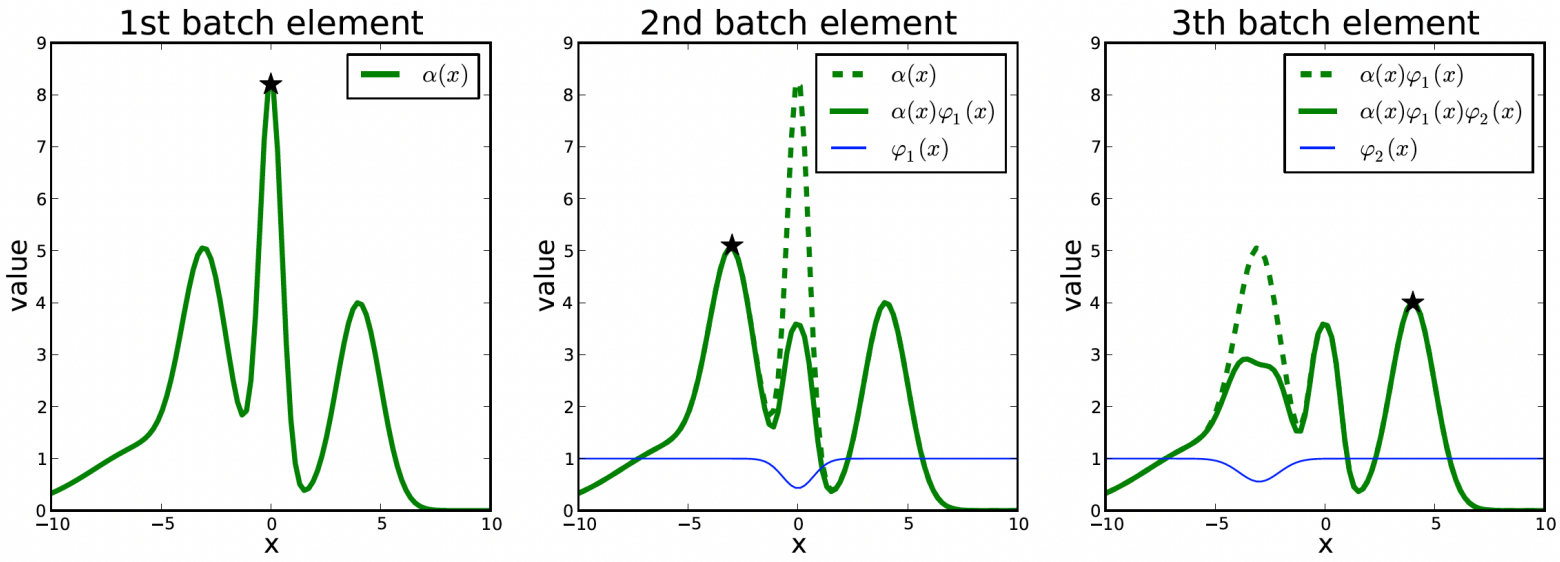


Figure: Penalised Acquisition function. Image taken from [5]

Example of LPS

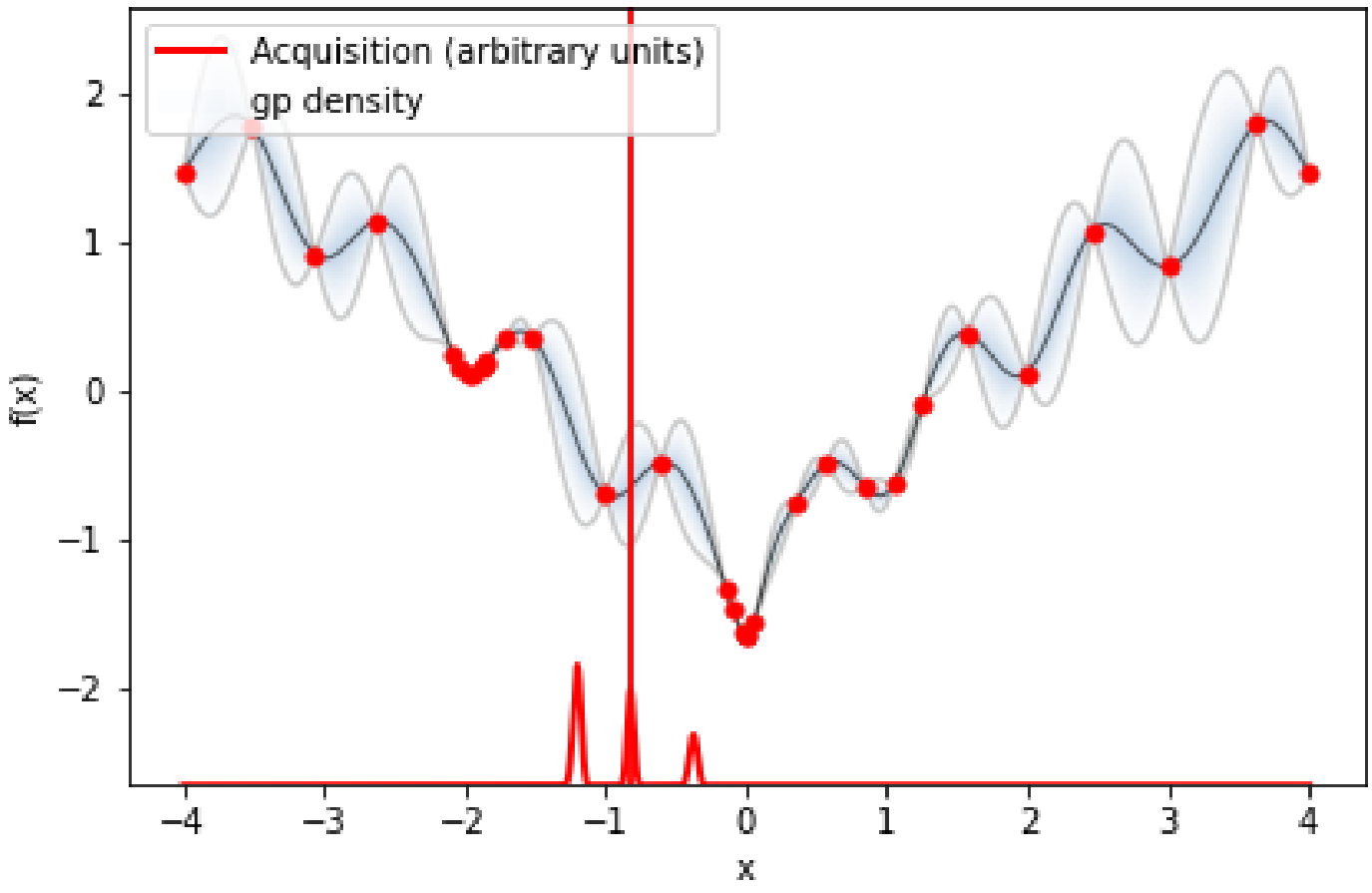


Figure: LPS applied to Ackley's function

Example of LPS

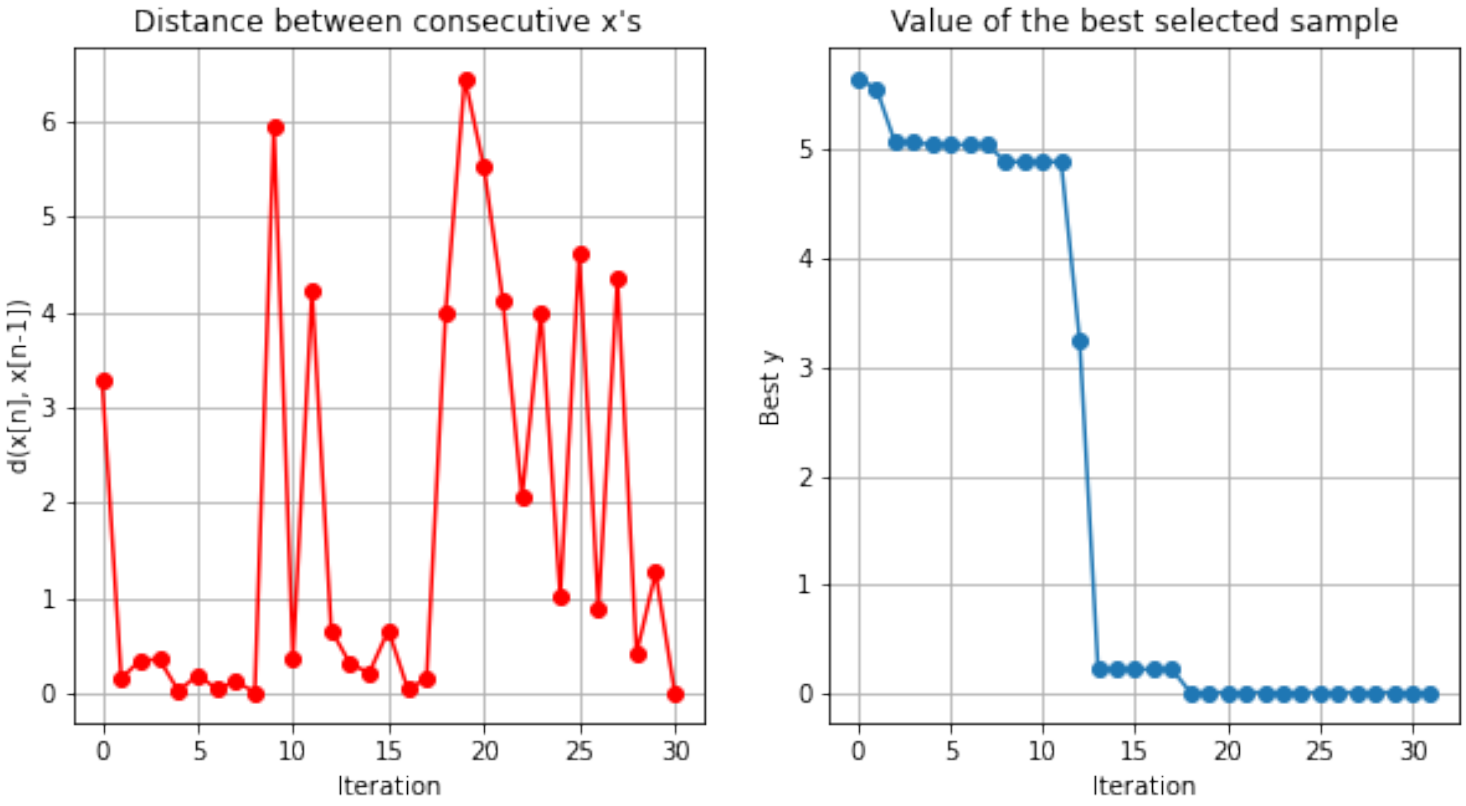


Figure: LPS applied to Ackley's function

Multi-objective Bayesian Optimization

Now the problem becomes that of optimising a *vector-valued objective* $\boldsymbol{f}(\mathbf{x}) : R^d \to \mathbb{R}^M$ with $\boldsymbol{f}(\mathbf{x}) = (f^{(1)}(\mathbf{x}), \cdots, f^{(M)}(\mathbf{x}))$; $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$.

Multi-objective Bayesian Optimization

Now the problem becomes that of optimising a *vector-valued objective* $\boldsymbol{f}(\mathbf{x}) : R^d \to \mathbb{R}^M$ with $\boldsymbol{f}(\mathbf{x}) = (f^{(1)}(\mathbf{x}), \cdots, f^{(M)}(\mathbf{x}))$; $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$.

Definitions

- The goal is to identify the set of *Pareto optimal* solutions.

Multi-objective Bayesian Optimization

Now the problem becomes that of optimising a *vector-valued objective* $\boldsymbol{f}(\mathbf{x}) : R^d \to \mathbb{R}^M$ with $\boldsymbol{f}(\mathbf{x}) = (f^{(1)}(\mathbf{x}), \cdots, f^{(M)}(\mathbf{x}))$; $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$.

Definitions

- The goal is to identify the set of *Pareto optimal* solutions.
- A solution $\boldsymbol{f}(\mathbf{x})$ *dominates* another solution $\boldsymbol{f}(\mathbf{x}')$ if $f^{(m)}(\mathbf{x}) \geq f^{(m)}(\mathbf{x}') \; \forall m = 1, \cdots, M$ and there exists $m' \in \{1, \cdots, M\}$ such that $f^{(m')}(\mathbf{x}) > f^{(m')}(\mathbf{x}')$.

Multi-objective Bayesian Optimization

Now the problem becomes that of optimising a *vector-valued objective* $\boldsymbol{f}(\mathbf{x}) : R^d \to \mathbb{R}^M$ with $\boldsymbol{f}(\mathbf{x}) = (f^{(1)}(\mathbf{x}), \cdots, f^{(M)}(\mathbf{x}))$; $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$.

Definitions

- The goal is to identify the set of *Pareto optimal* solutions.
- A solution $\boldsymbol{f}(\mathbf{x})$ *dominates* another solution $\boldsymbol{f}(\mathbf{x}')$ if $f^{(m)}(\mathbf{x}) \geq f^{(m)}(\mathbf{x}') \; \forall m = 1, \cdots, M$ and there exists $m' \in \{1, \cdots, M\}$ such that $f^{(m')}(\mathbf{x}) > f^{(m')}(\mathbf{x}')$.
- This is write as $\boldsymbol{f}(\mathbf{x}) \succ \boldsymbol{f}(\mathbf{x}')$

Multi-objective Bayesian Optimization

Now the problem becomes that of optimising a *vector-valued objective* $\boldsymbol{f}(\mathbf{x}) : R^d \to \mathbb{R}^M$ with $\boldsymbol{f}(\mathbf{x}) = (f^{(1)}(\mathbf{x}), \cdots, f^{(M)}(\mathbf{x}))$; $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$.

Definitions

- The goal is to identify the set of *Pareto optimal* solutions.
- A solution $\boldsymbol{f}(\mathbf{x})$ *dominates* another solution $\boldsymbol{f}(\mathbf{x}')$ if $f^{(m)}(\mathbf{x}) \geq f^{(m)}(\mathbf{x}')\ \forall m = 1, \cdots, M$ and there exists $m' \in \{1, \cdots, M\}$ such that $f^{(m')}(\mathbf{x}) > f^{(m')}(\mathbf{x}')$.
- This is write as $\boldsymbol{f}(\mathbf{x}) \succ \boldsymbol{f}(\mathbf{x}')$
- $\mathcal{P}^{=}\{\boldsymbol{f}(\mathbf{x})\ s.t.\ \nexists\, \mathbf{x}' \in \mathcal{X} : \boldsymbol{f}(\mathbf{x}') \succ \boldsymbol{f}(\mathbf{x})\}$

Hypervolume Improvement

$$\mathrm{HVI}(\mathbf{y}, \mathcal{P}) = HV(\mathcal{P} \cup \{\mathbf{y}\}) - HV(\mathcal{P})$$

[8]

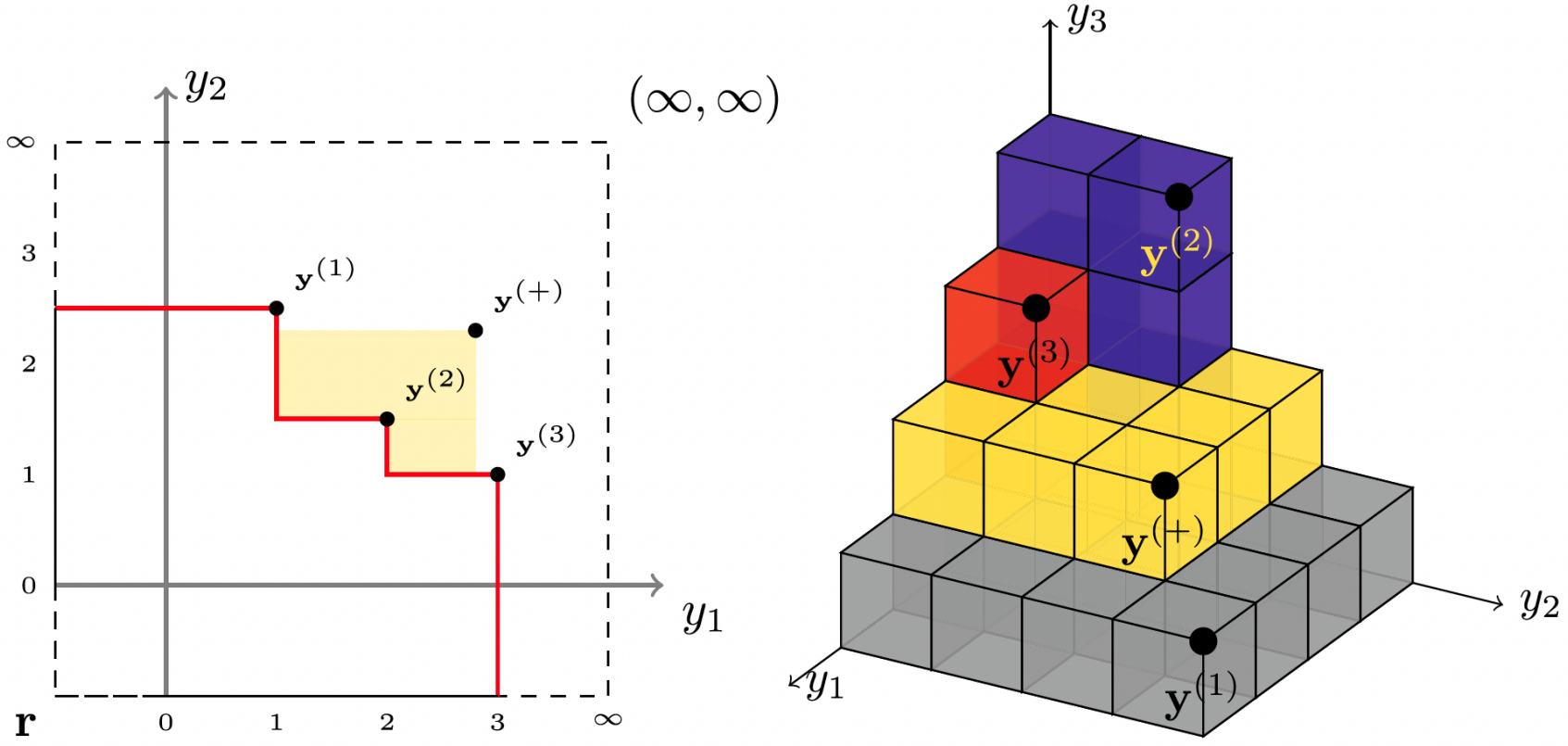


Fig. 1 The left and right figures illustrate *Hypervolume Improvement* in a 2-D and a 3-D example, respectively

Expected Hypervolume Improvement

$$\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{\mathbb{R}^M} \mathrm{HVI}(\mathbf{y}, \mathcal{P}) \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}$$

[8]

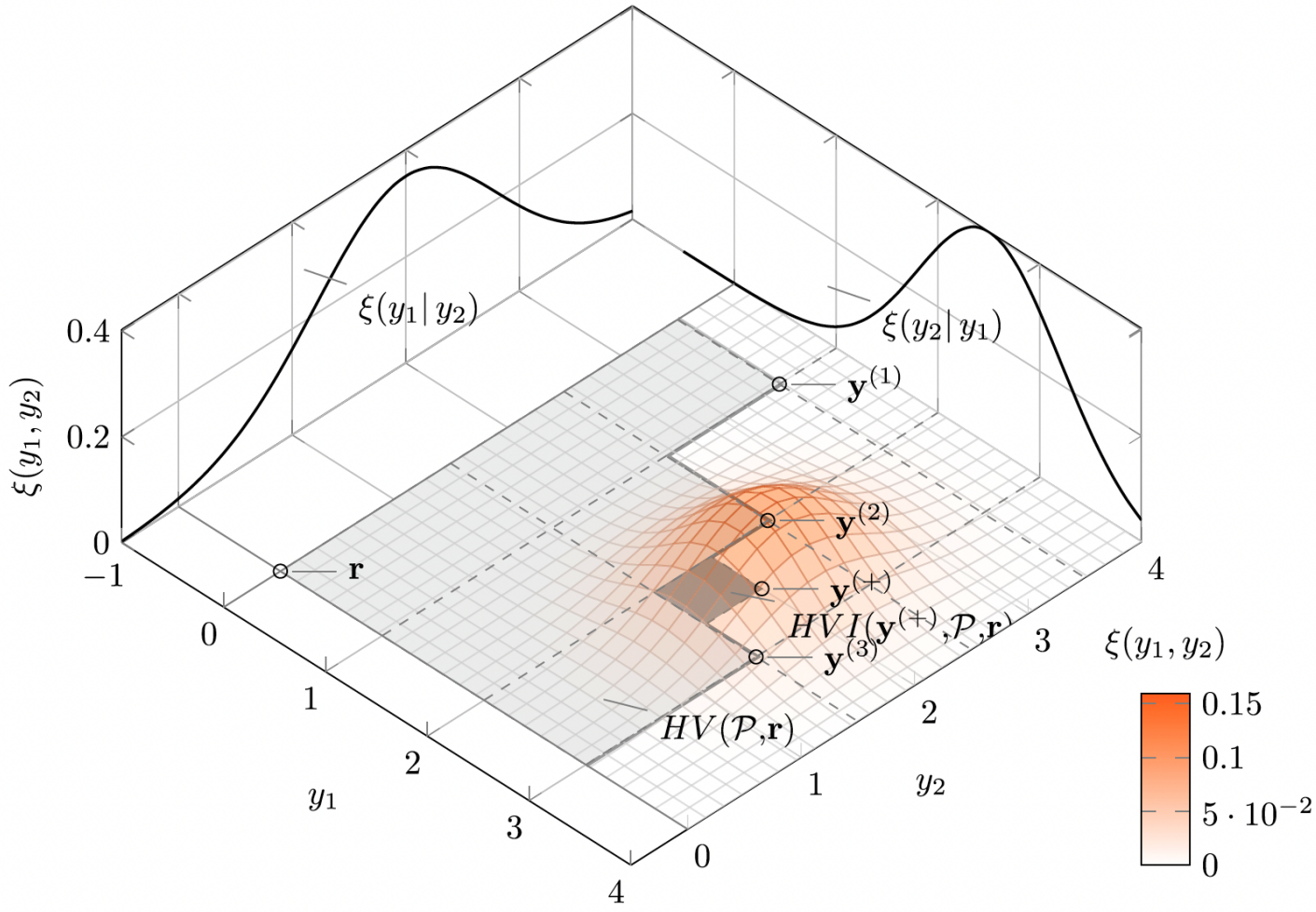
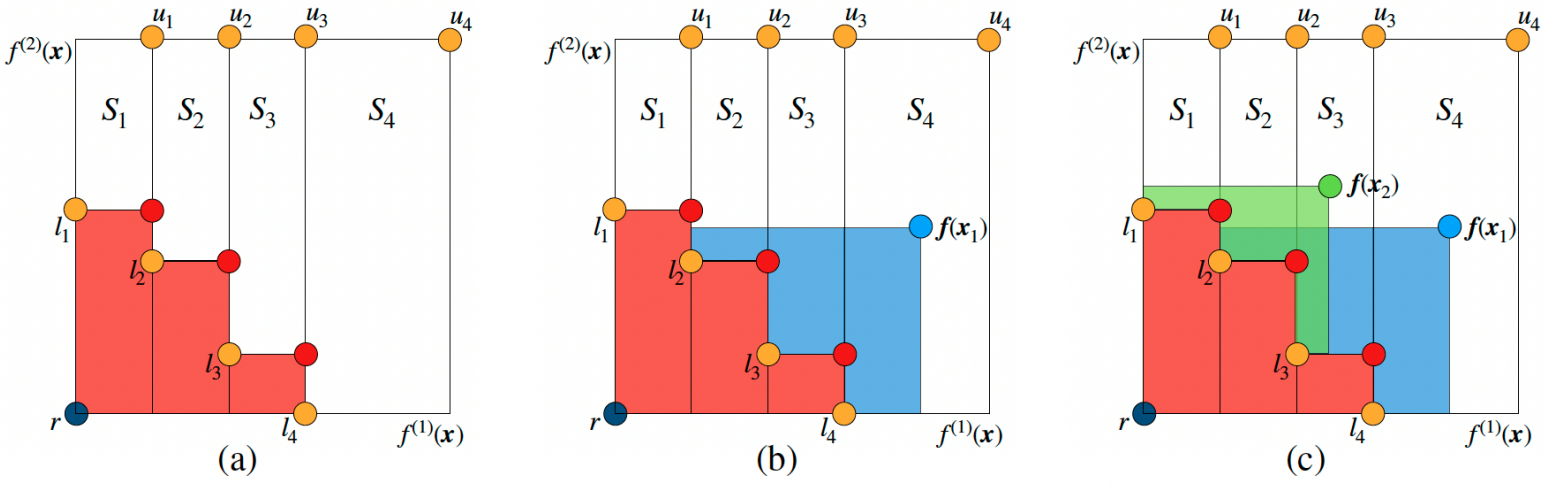
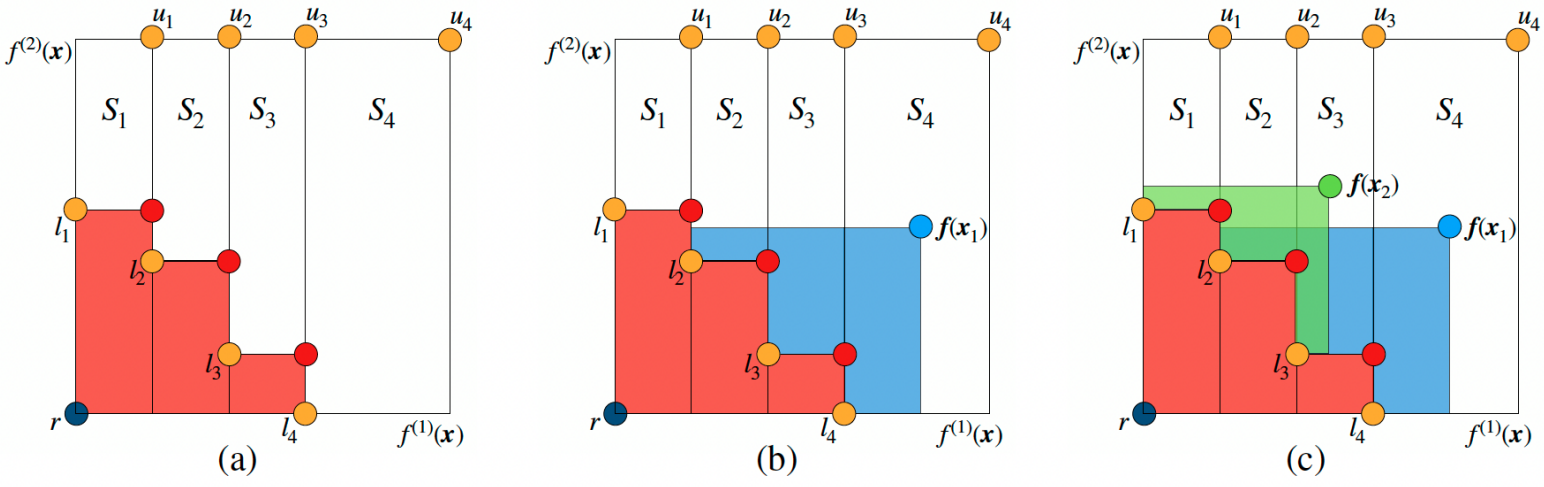


Fig. 2 Expected hypervolume improvement in 2-D (cf. Example 2)

Hypervolume improvement computation using box decompositions

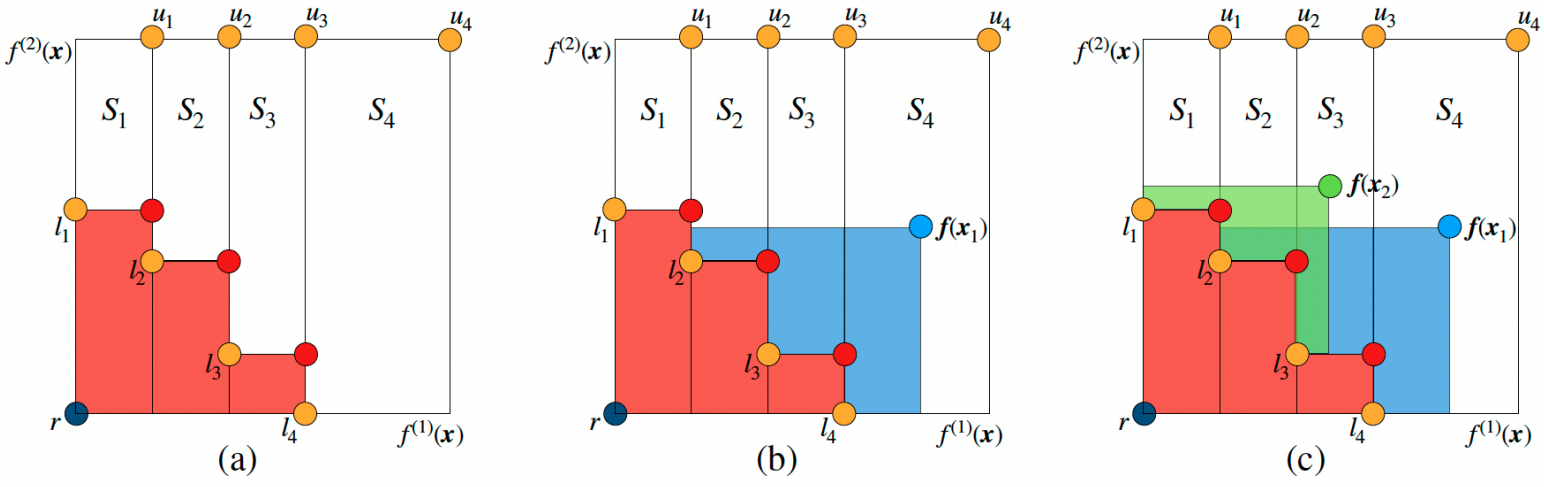


Hypervolume improvement computation using box decompositions



$$qEHVI(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{\mathbb{R}^M, q} \mathrm{HVI}(\mathcal{Y}, \mathcal{P}) \boldsymbol{\Xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathcal{Y}) d\mathcal{Y}$$

Hypervolume improvement computation using box decompositions



$$qEHVI(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{\mathbb{R}^M, q} \text{HVI}(\mathcal{Y}, \mathcal{P}) \boldsymbol{\Xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathcal{Y}) d\mathcal{Y}$$

Given posterior samples, the computational complexity of estimating qEHVI is $O(MNK(2^q - 1))$. But, The number of boxes required for a decomposition of the non-dominated space is unknown for $M \geq 4$ [3].

MOBO algorithm

1. Find the Pareto frontier $\mathcal{P}_t$ using the current set of samples $\mathcal{D}_{1:t}$.

MOBO algorithm

1. Find the Pareto frontier $\mathcal{P}_t$ using the current set of samples $\mathcal{D}_{1:t}$.
2. Find the next q sampling points $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\}$ by optimising the acquisition function over the GPs: $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\} = \text{argmax}_{\mathbf{x}_{1,\cdots,q}} \alpha_{qEHVI}(\mathbf{x}; \mathcal{P}_t, \mathcal{I}_t)$

MOBO algorithm

1. Find the Pareto frontier $\mathcal{P}_t$ using the current set of samples $\mathcal{D}_{1:t}$.
2. Find the next q sampling points $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\}$ by optimising the acquisition function over the GPs: $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\} = \text{argmax}_{\mathbf{x}_{1,\cdots,q}} \alpha_{qEHVI}(\mathbf{x}; \mathcal{P}_t, \mathcal{I}_t)$
3. Obtain samples $y_{t+1}, \cdots, y_{t+q}$ from the objective function f

MOBO algorithm

1. Find the Pareto frontier $\mathcal{P}_t$ using the current set of samples $\mathcal{D}_{1:t}$.
2. Find the next q sampling points $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\}$ by optimising the acquisition function over the GPs: $\{\mathbf{x}_{t+1}, \cdots, \mathbf{x}_{t+q}\} = \text{argmax}_{\mathbf{x}_{1,\cdots,q}} \alpha_{qEHVI}(\mathbf{x}; \mathcal{P}_t, \mathcal{I}_t)$
3. Obtain samples $y_{t+1}, \cdots, y_{t+q}$ from the objective function f
4. Add the samples to previous samples $\mathcal{D}_{1:t+q} = \{\mathcal{D}_{1:t}, (\mathbf{x}_{t+1}, \mathbf{y}_{t+1}), \cdots, (\mathbf{x}_{t+q}, \mathbf{y}_{t+q})\}$ and update the GPs.

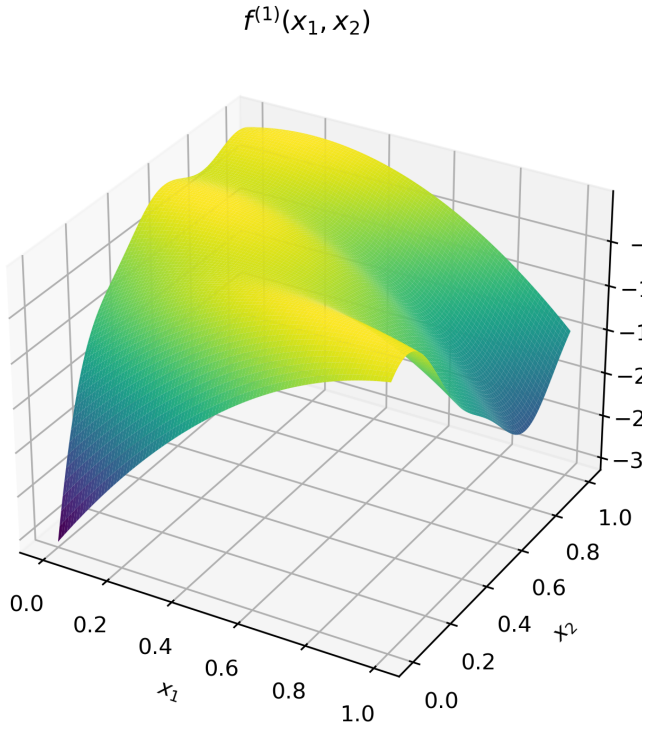
Example of MOBO using qEHVI algorithm

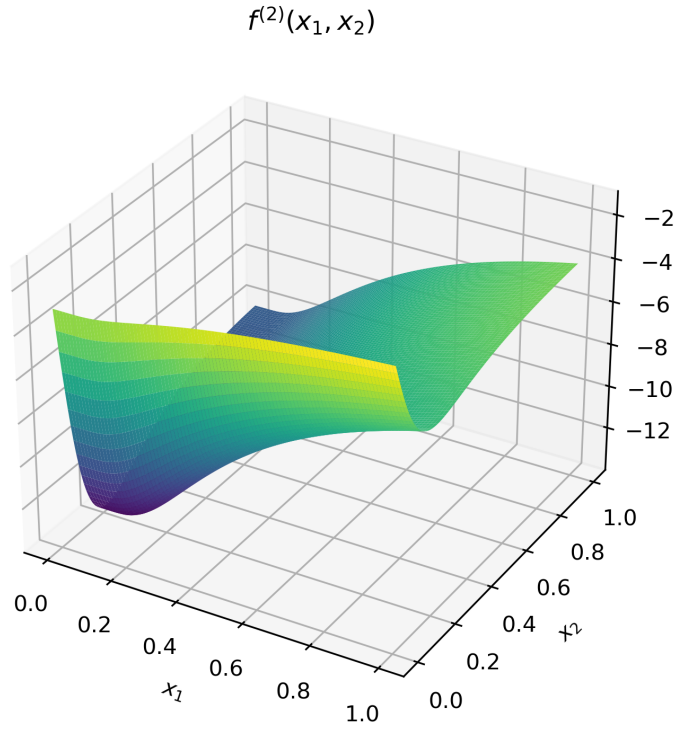
Branin-Currin test function. The two objectives are

$$f^{(1)}(x_1', x_2') = (x_2' - \frac{5.1}{4\pi^2}(x_1')^2 + \frac{5}{\pi}x_1' - r)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1') + 10$$

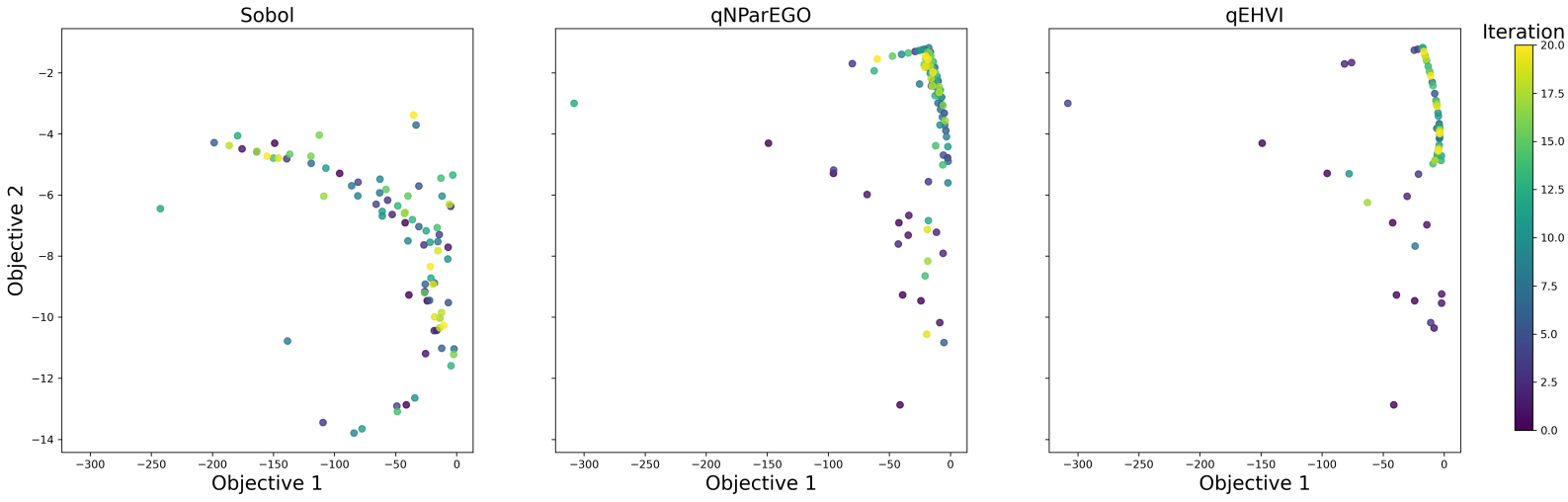
$$f^{(2)}(x_1, x_2) = \left[1 - \exp\left(-\frac{1}{(2x_2)}\right)\right] \frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}$$

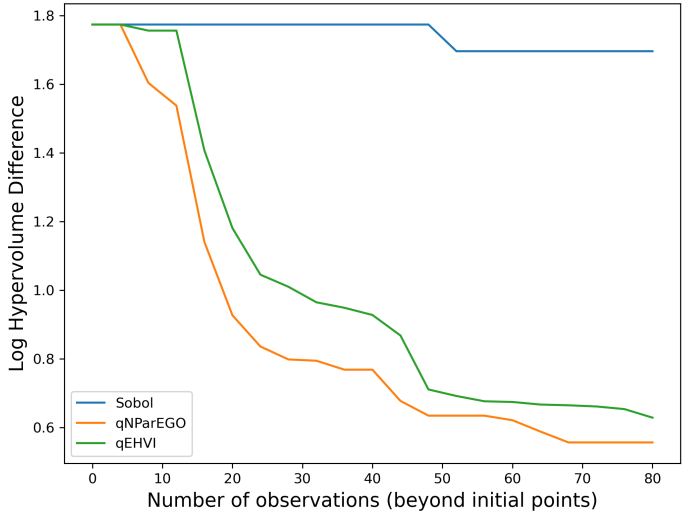
where $x_1, x_2 \in [0, 1]$, $x_1' = 15x_1 - 5$, and $x_2' = 15x_2$





Example of MOBO using qEHVI algorithm

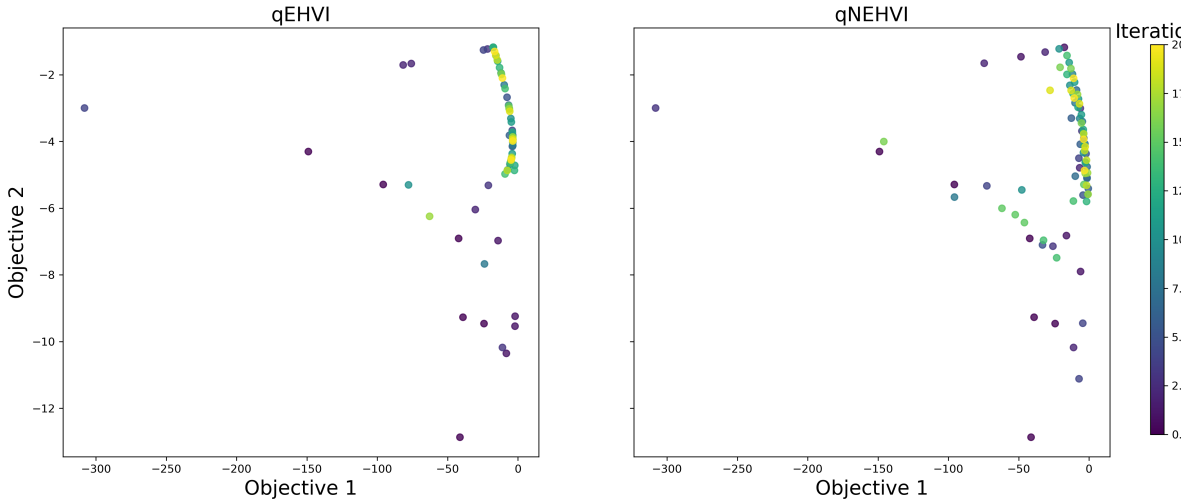


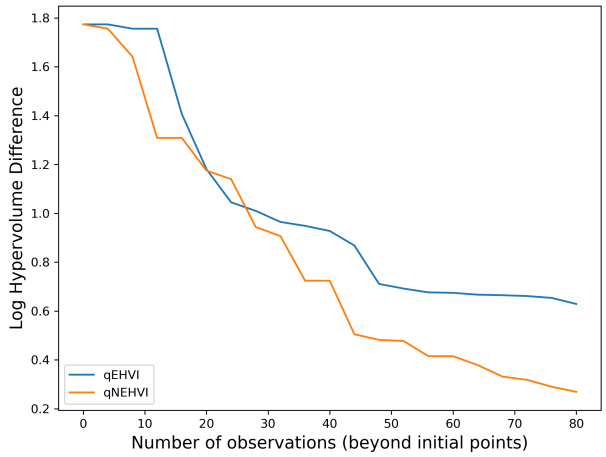


Noisy Expected Hypervolume Improvement

This AC also integrates over the noise component and provides better performance in terms of **exploration** [4].

$$\alpha_{NEHVI}(\mathbf{x}; \mathcal{I}_t) = \frac{1}{N} \sum_{n=1}^{N} \text{HVI}(\tilde{\boldsymbol{f}}_n(\mathbf{x}), \mathcal{P}_n)$$





Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\begin{bmatrix} \mathbf{f^{(1)}} \\ \mathbf{f^{(2)}} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K_1 & \mathbf{0} \\ \mathbf{0} & K_2 \end{bmatrix} \right)$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{f} \sim \mathcal{N}\left(\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}K_1 & \mathbf{0}\\ \mathbf{0} & K_2\end{bmatrix}\right)$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_1 & \mathbf{0} \\ \mathbf{0} & K_2 \end{bmatrix}\right)$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{f} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}})$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \begin{bmatrix} K_1 & \mathbf{0} \\ \mathbf{0} & K_2 \end{bmatrix}$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \begin{bmatrix} K_1 & ? \\ ? & K_2 \end{bmatrix}$$

Multi-output surrogate model

In the previous formulation $\boldsymbol{f}(\mathbf{x})$ corresponds to a set of independent GPs. This can be expressed as:

$$f^{(1)}(\mathbf{x}) \sim \mathcal{GP}(0, k_1(\mathbf{x}, \mathbf{x}')) \quad f^{(2)}(\mathbf{x}) \sim \mathcal{GP}(0, k_2(\mathbf{x}, \mathbf{x}'))$$

The sample sites for every function could be different, but in an optimization problem, it is possible to assume that the sample sites are shared.

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \begin{bmatrix} K_1 & ? \\ ? & K_2 \end{bmatrix}$$

The problem addressed by Multi-output GPs is to build a cross-covariance function $\text{cov}[\mathbf{f}^{(1)}(\mathbf{x}), \mathbf{f}^{(2)}(\mathbf{x}')]$ such that $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is positive semi-definite.

Intrinsic coregionalization model (ICM)

We assume the following generative model for $\mathbf{f}(\mathbf{x})$ [1]:

- Sample from a GP $u(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ to obtain $u^1(\mathbf{x})$
- Obtain $f^{(1)}(\mathbf{x})$ and $f^{(2)}(\mathbf{x})$ by linearly transforming $u^1(\mathbf{x})$.

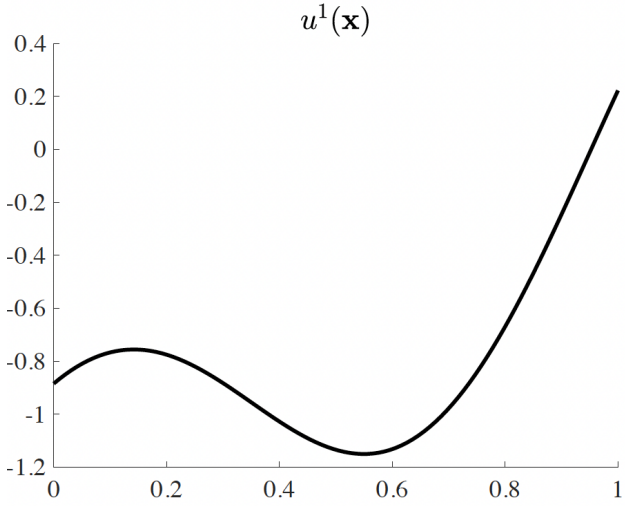
$$f^{(1)}(\mathbf{x}) = a_1^1 u^1(\mathbf{x}) \quad f^{(2)}(\mathbf{x}) = a_2^1 u^1(\mathbf{x})$$
$$\text{cov}[f^{(1)}(\mathbf{x}), f^{(2)}(\mathbf{x}')] = \mathbf{a}\mathbf{a}^\top k(\mathbf{x}, \mathbf{x}')$$

Intrinsic coregionalization model (ICM)

We assume the following generative model for $\mathbf{f}(\mathbf{x})$ [1]:

- Sample from a GP $u(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ to obtain $u^1(\mathbf{x})$
- Obtain $f^{(1)}(\mathbf{x})$ and $f^{(2)}(\mathbf{x})$ by linearly transforming $u^1(\mathbf{x})$.

$$f^{(1)}(\mathbf{x}) = a_1^1 u^1(\mathbf{x}) \quad f^{(2)}(\mathbf{x}) = a_2^1 u^1(\mathbf{x})$$
$$\text{cov}[f^{(1)}(\mathbf{x}), f^{(2)}(\mathbf{x}')] = \mathbf{a}\mathbf{a}^\top k(\mathbf{x}, \mathbf{x}')$$

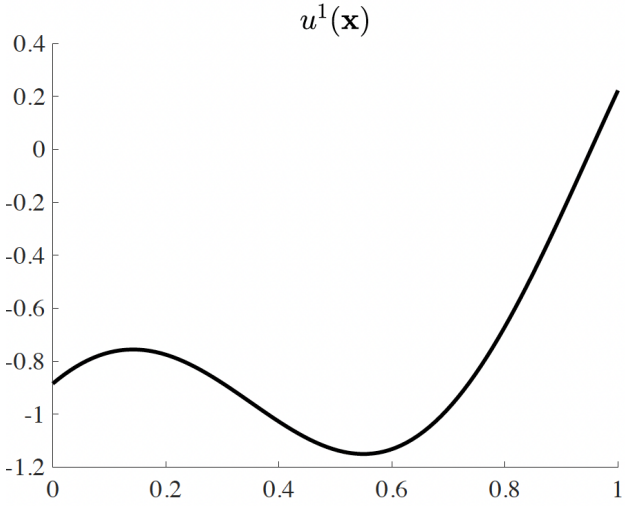


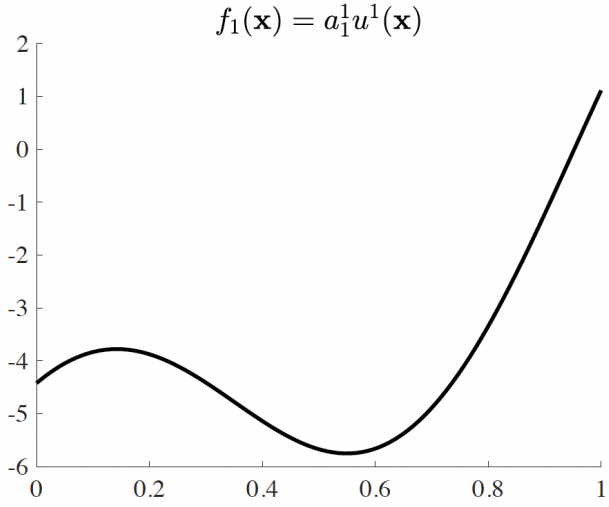
Intrinsic coregionalization model (ICM)

We assume the following generative model for $\mathbf{f}(\mathbf{x})$ [1]:

- Sample from a GP $u(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ to obtain $u^1(\mathbf{x})$
- Obtain $f^{(1)}(\mathbf{x})$ and $f^{(2)}(\mathbf{x})$ by linearly transforming $u^1(\mathbf{x})$.

$$f^{(1)}(\mathbf{x}) = a_1^1 u^1(\mathbf{x}) \quad f^{(2)}(\mathbf{x}) = a_2^1 u^1(\mathbf{x})$$
$$\text{cov}[f^{(1)}(\mathbf{x}), f^{(2)}(\mathbf{x}')] = \mathbf{a}\mathbf{a}^\top k(\mathbf{x}, \mathbf{x}')$$



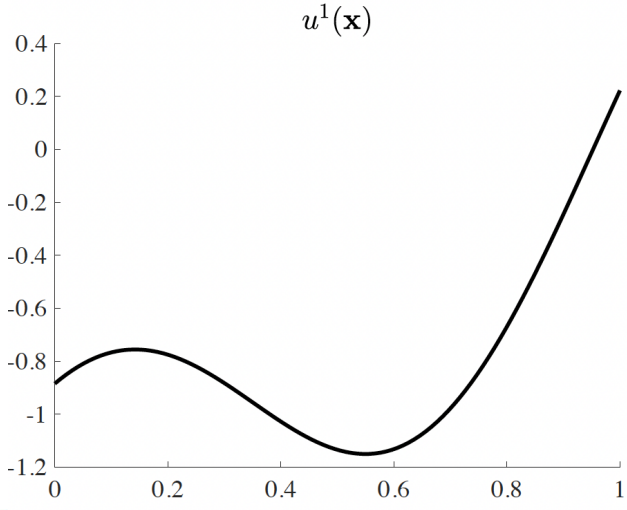


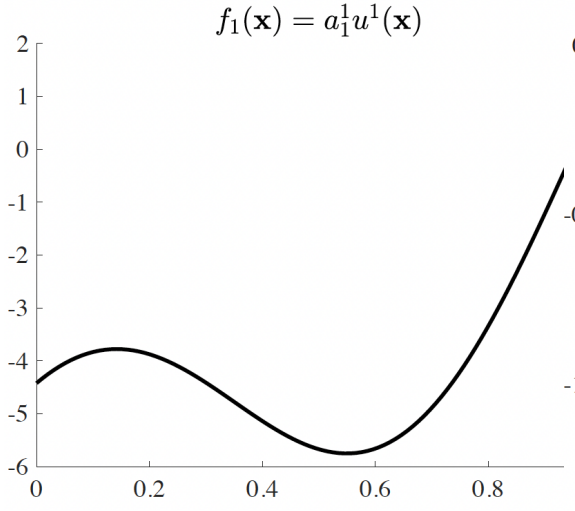
Intrinsic coregionalization model (ICM)

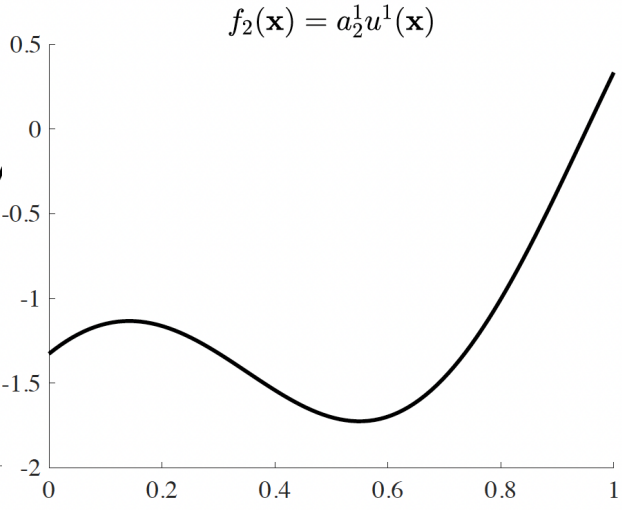
We assume the following generative model for $\mathbf{f}(\mathbf{x})$ [1]:

- Sample from a GP $u(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ to obtain $u^1(\mathbf{x})$
- Obtain $f^{(1)}(\mathbf{x})$ and $f^{(2)}(\mathbf{x})$ by linearly transforming $u^1(\mathbf{x})$.

$$f^{(1)}(\mathbf{x}) = a_1^1 u^1(\mathbf{x}) \quad f^{(2)}(\mathbf{x}) = a_2^1 u^1(\mathbf{x})$$
$$\text{cov}[f^{(1)}(\mathbf{x}), f^{(2)}(\mathbf{x}')] = \mathbf{a}\mathbf{a}^\top k(\mathbf{x}, \mathbf{x}')$$

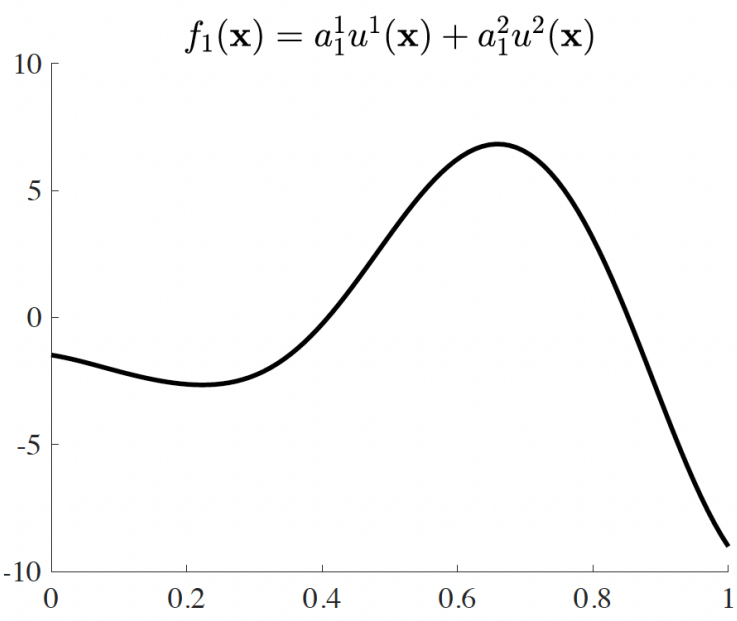


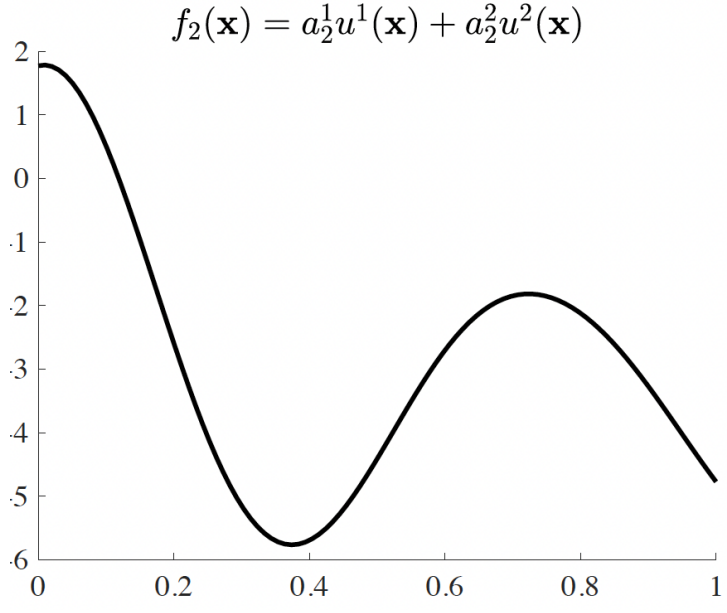




ICM: more complex configurations

- Sample twice from a GP $u(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ to obtain $u^1(\mathbf{x})$ and $u^2(\mathbf{x})$
- $f^{(1)}(\mathbf{x}) = a_1^1 u^1(\mathbf{x}) + a_1^2 u^2(\mathbf{x})$ $\quad f^{(2)}(\mathbf{x}) = a_2^1 u^1(\mathbf{x}) + a_2^2 u^2(\mathbf{x})$
- $\mathbf{B} = \mathbf{a}^1(\mathbf{a}^1)^\top + \mathbf{a}^2(\mathbf{a}^2)^\top$
- $\text{cov}[f^{(1)}(\mathbf{x}), f^{(2)}(\mathbf{x}')] = \mathbf{B} \otimes k(\mathbf{x}, \mathbf{x}')$





ICM: general case

$$f^{(m)}(\mathbf{x}) = \sum_{i=1}^{R} a_m^i u^i(\mathbf{x})$$

ICM: general case

$$f^{(m)}(\mathbf{x}) = \sum_{i=1}^{R} a_m^i u^i(\mathbf{x})$$

Semiparametric Latent Factor Model (SLFM)

SLFM uses Q samples from $u_q(\mathbf{x})$ processes with different covariance functions. $f^{(m)}(\mathbf{x}) = \sum_{q=1}^{Q} a_{m,q} u_q(\mathbf{x})$. Where $u_q(\mathbf{x})$ are GPs with covariance functions $k_q(\mathbf{x}, \mathbf{x}')$.

ICM: general case

$$f^{(m)}(\mathbf{x}) = \sum_{i=1}^{R} a_m^i u^i(\mathbf{x})$$

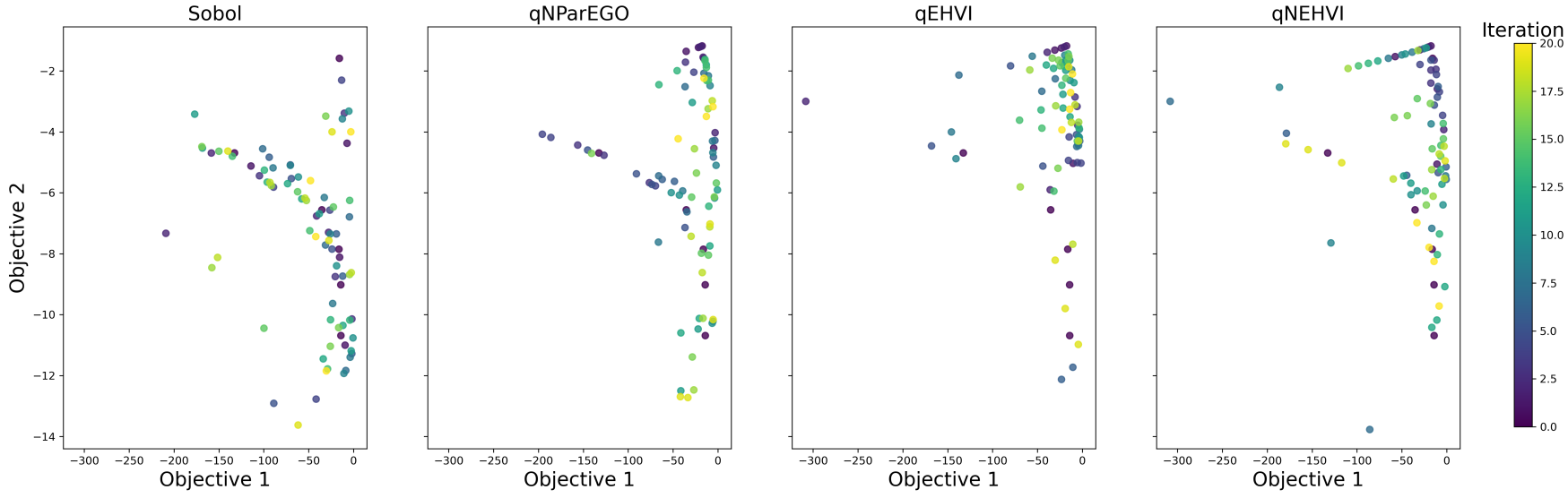
Semiparametric Latent Factor Model (SLFM)

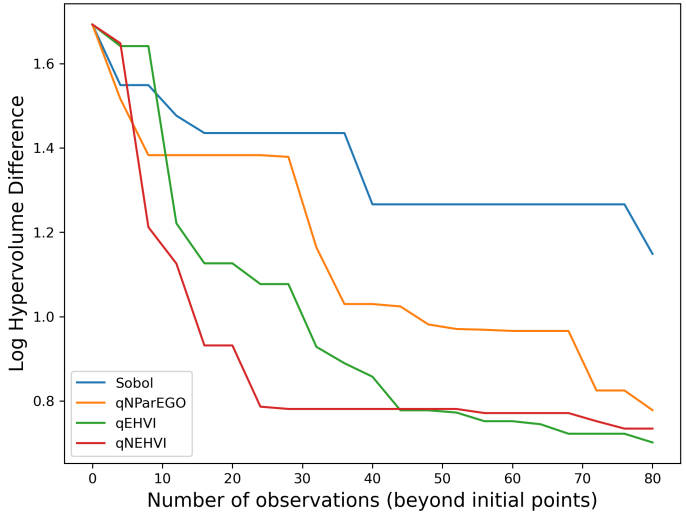
SLFM uses Q samples from $u_q(\mathbf{x})$ processes with different covariance functions. $f^{(m)}(\mathbf{x}) = \sum_{q=1}^{Q} a_{m,q} u_q(\mathbf{x})$. Where $u_q(\mathbf{x})$ are GPs with covariance functions $k_q(\mathbf{x}, \mathbf{x}')$.

Linear model of coregionalization (LMC)

LCM generalises the ICM and the SLFM allowing several independent samples from GPs with different covariances. $f^{(m)}(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} a_{m,q}^i u_q^i(\mathbf{x})$. Where $u_q^i(\mathbf{x})$ are GPs with covariance functions $k_q(\mathbf{x}, \mathbf{x}')$.

MOBO using ICM kernels





Questions?

[1] Mauricio A Alvarez, Lorenzo Rosasco, Neil D Lawrence, et al. "Kernels for vector-valued functions: A review". In: *Foundations and Trends® in Machine Learning* 4.3 (2012), pp. 195–266.

[2] Eric Brochu, Vlad M Cora, and Nando De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: *arXiv preprint arXiv:1012.2599* (2010).

[3] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. "Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9851–9864.

[4] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. "Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 2187–2200.

[5] Javier González et al. "Batch Bayesian optimization via local penalization". In: *Artificial intelligence and statistics*. PMLR. 2016, pp. 648–657.

[6] Carl Edward Rasmussen and Christopher K Williams. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

[7] Jialei Wang et al. "Parallel Bayesian global optimization of expensive functions". In: *Operations Research* 68.6 (2020), pp. 1850–1865.

[8] Kaifeng Yang et al. "Efficient computation of expected hypervolume improvement using box decomposition algorithms". In: *Journal of Global Optimization* 75 (2019), pp. 3–34.

Thank You

Julián D. Arias-Londoño
julian.arias@upm.es