Stochastic Gradient Descent for Gaussian Processes

Shreyas Padhy

23 February 2024

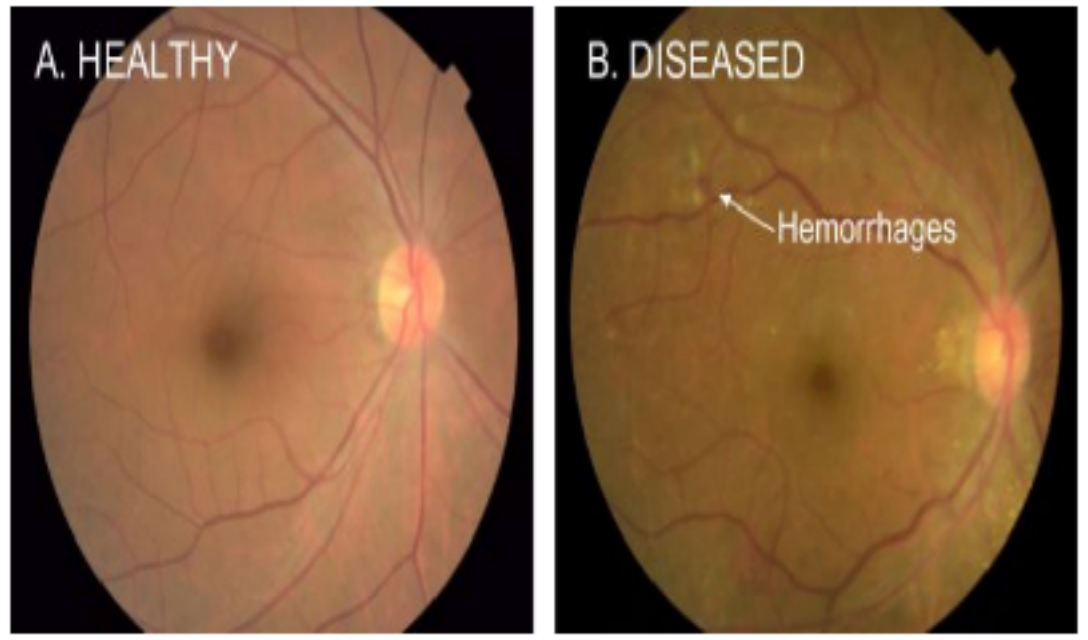
Computational and Biological Learning
DEPARTMENT OF ENGINEERING

UNIVERSITY OF CAMBRIDGE

Why do we need Uncertainty Estimates?

Why do we need Uncertainty Estimates?

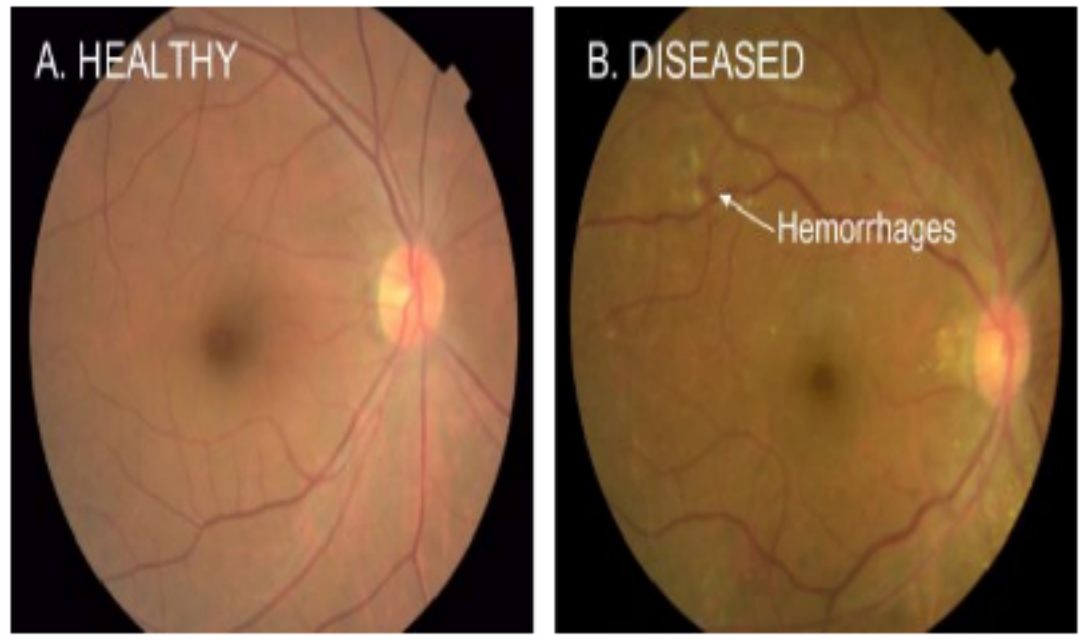
- Deep Learning is massively scalable and extremely powerful at modelling data



[1] Gulshan, Varun, et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." *Jama* 316.22 (2016): 2402-2410.
[2[Sun, P., et al. "Scalability in perception for autonomous driving: Waymo open dataset. arXiv pp. arXiv–1912." (2019).

Why do we need Uncertainty Estimates?

- Deep Learning is massively scalable and extremely powerful at modelling data



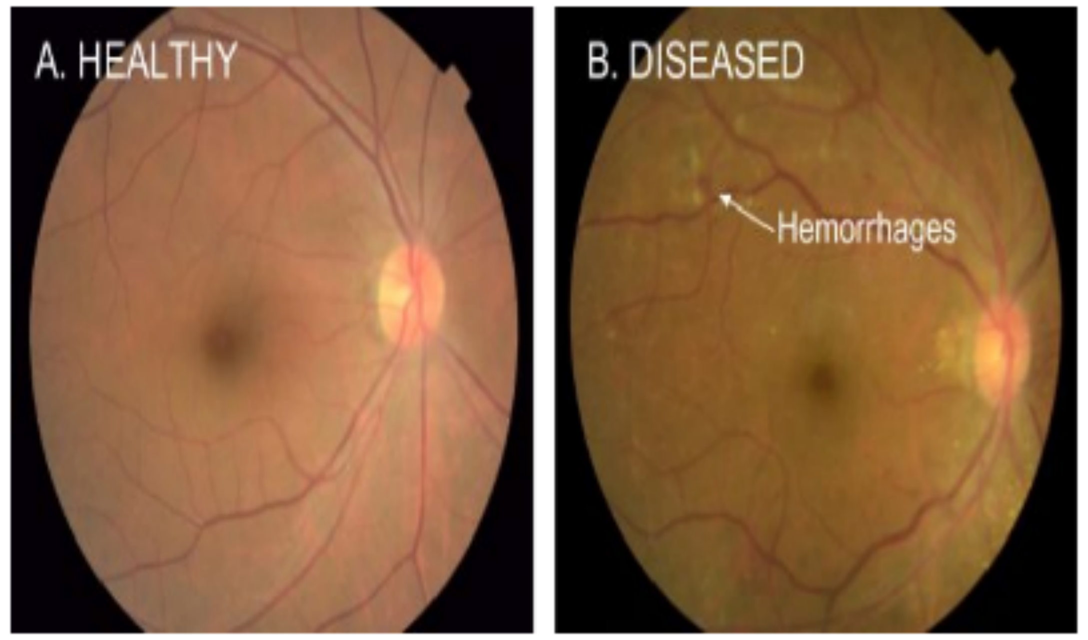
Low Degree of Belief on Prediction

Defer predictions

[1] Gulshan, Varun, et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." *Jama* 316.22 (2016): 2402-2410.
[2[Sun, P., et al. "Scalability in perception for autonomous driving: Waymo open dataset. arXiv pp. arXiv–1912." (2019).

Why do we need Uncertainty Estimates?

- Deep Learning is massively scalable and extremely powerful at modelling data



Low Degree of
Belief on
Prediction

Defer predictions

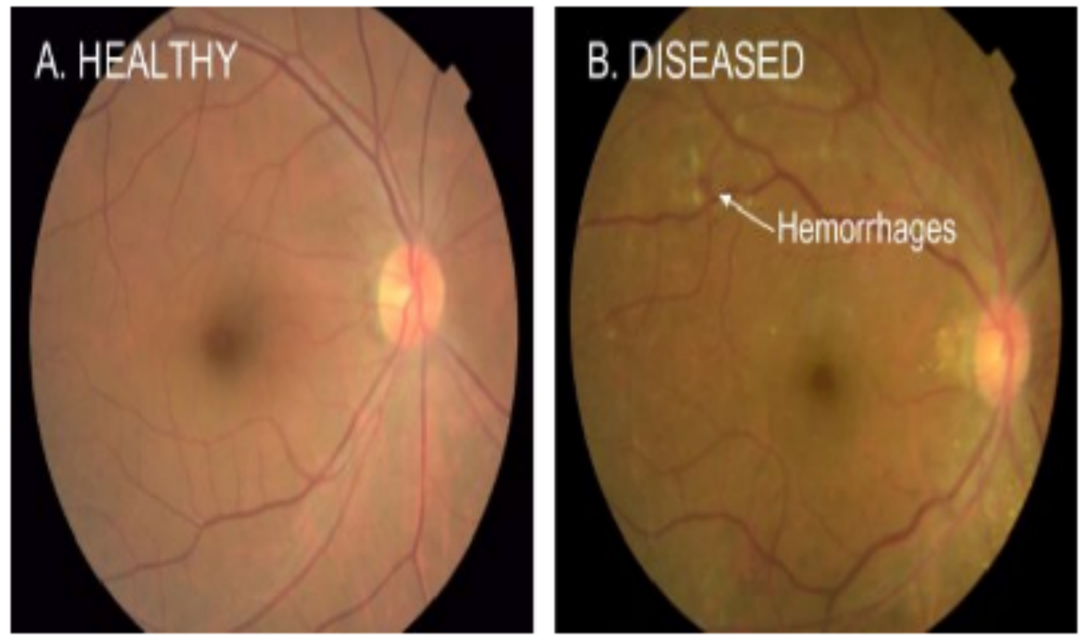
Low Confidence
in Certain
Scenarios

Identify and model
dataset shift

[1] Gulshan, Varun, et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." *Jama* 316.22 (2016): 2402-2410.
[2[Sun, P., et al. "Scalability in perception for autonomous driving: Waymo open dataset. arXiv pp. arXiv–1912." (2019).

Why do we need Uncertainty Estimates?

- Deep Learning is massively scalable and extremely powerful at modelling data



Low Degree of Belief on Prediction

Defer predictions

Low Confidence in Certain Scenarios

Identify and model dataset shift

[1] Gulshan, Varun, et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." *Jama* 316.22 (2016): 2402-2410.
[2[Sun, P., et al. "Scalability in perception for autonomous driving: Waymo open dataset. arXiv pp. arXiv–1912." (2019).

Why do we need Uncertainty Estimates?

- Deep Learning is massively scalable and extremely powerful at modelling data

If we use numerical values to model "uncertainty", simple axioms on these uncertainties follow the laws of probability => Bayes' Rule

[Cox, 1946], [Jaynes, 2003]

Prediction

Scenarios

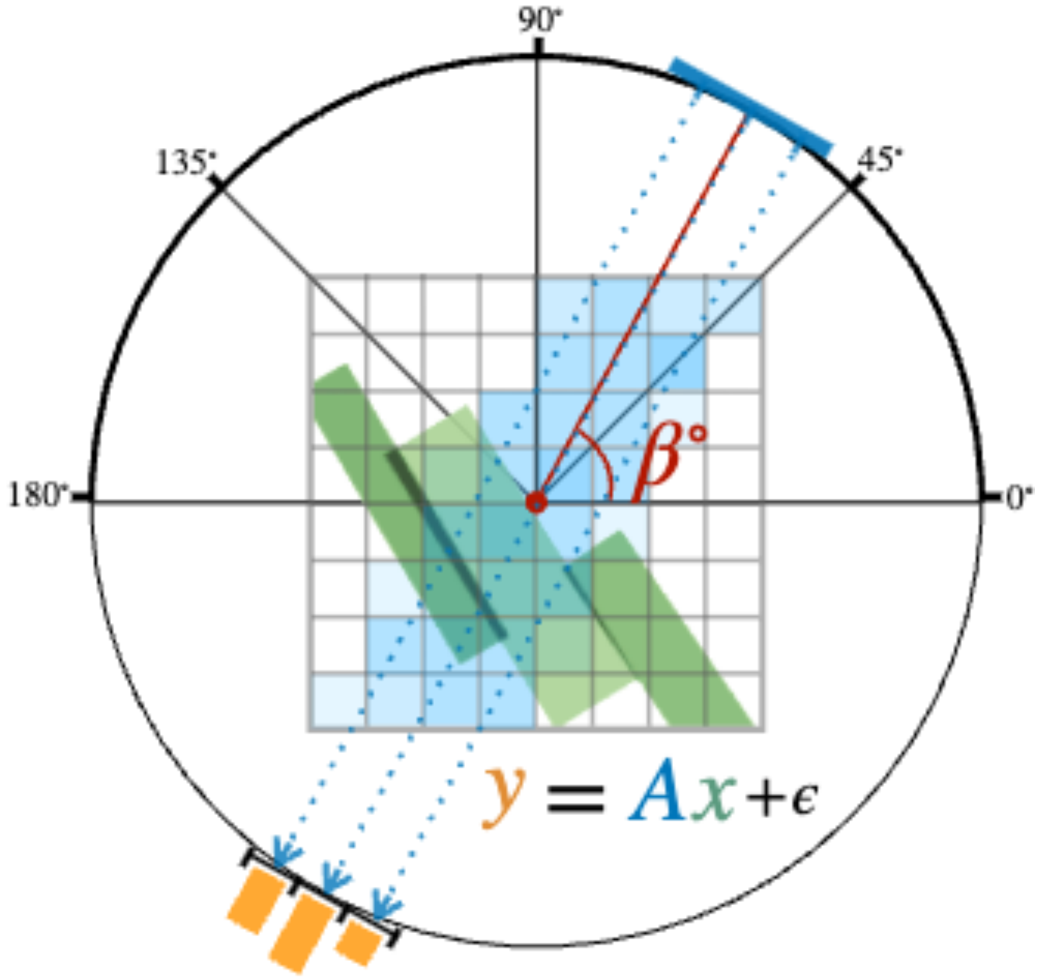
Defer predictions

Identify and model dataset shift

[1] Gulshan, Varun, et al. "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs." *Jama* 316.22 (2016): 2402-2410.
[2[Sun, P., et al. "Scalability in perception for autonomous driving: Waymo open dataset. arXiv pp. arXiv–1912." (2019).

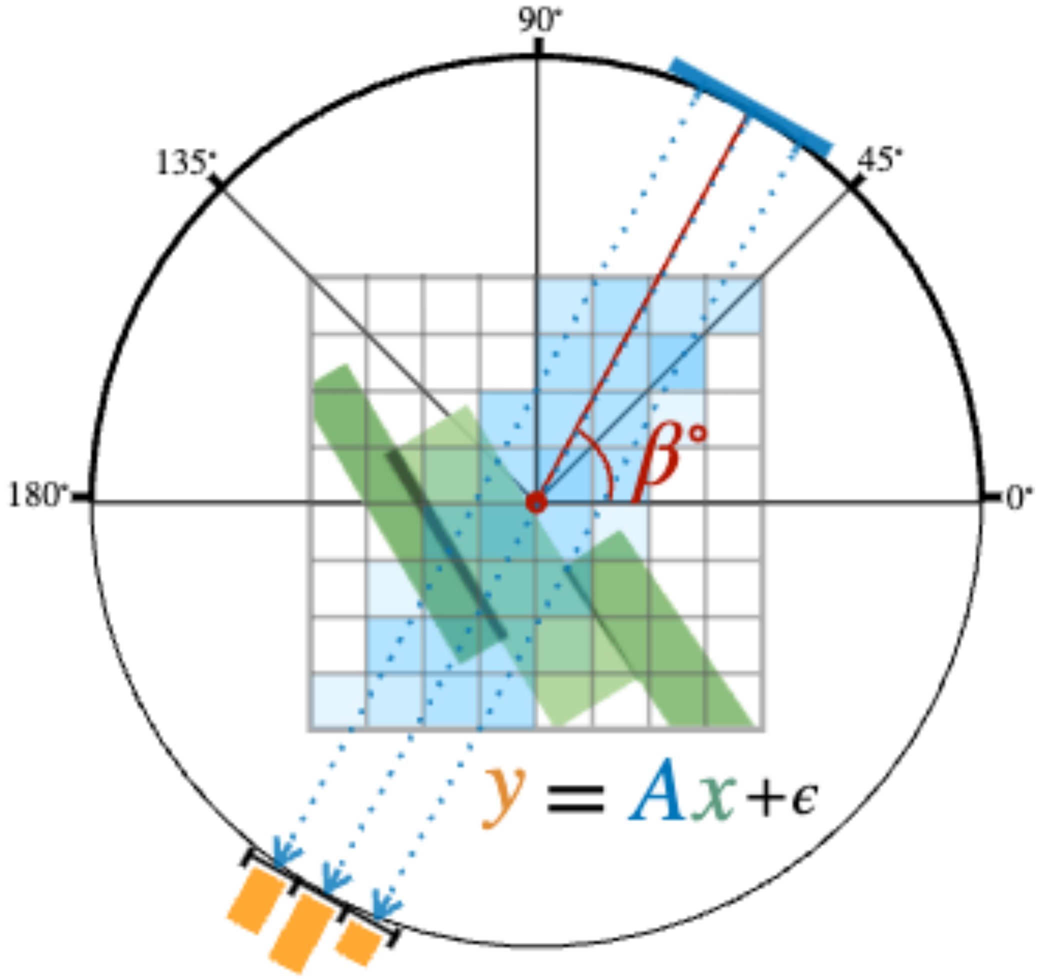
Example: Bayesian Experimental Design

Example: Bayesian Experimental Design



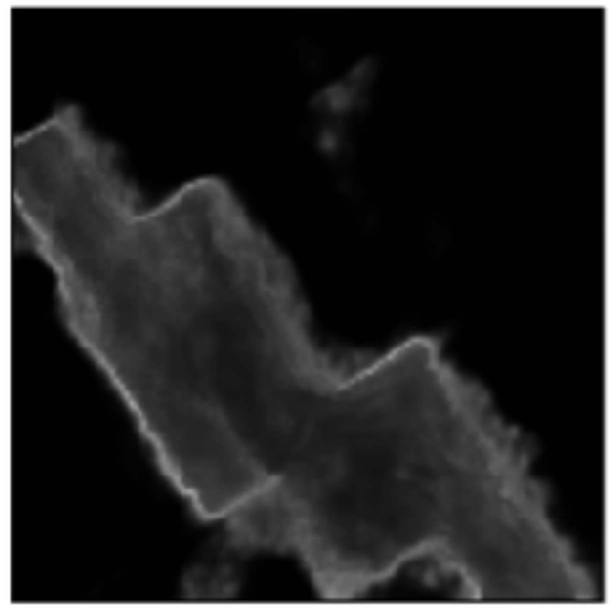
Image

Example: Bayesian Experimental Design



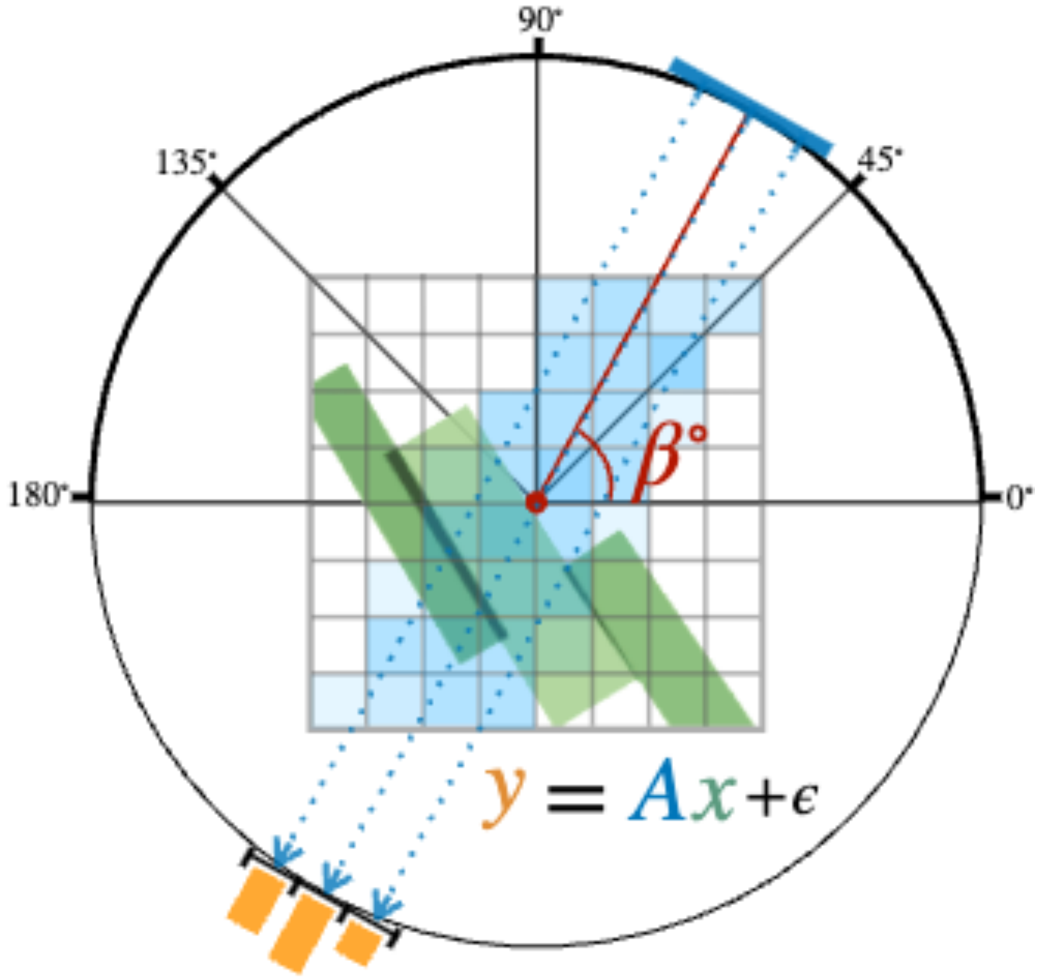
Image

Uncertainty



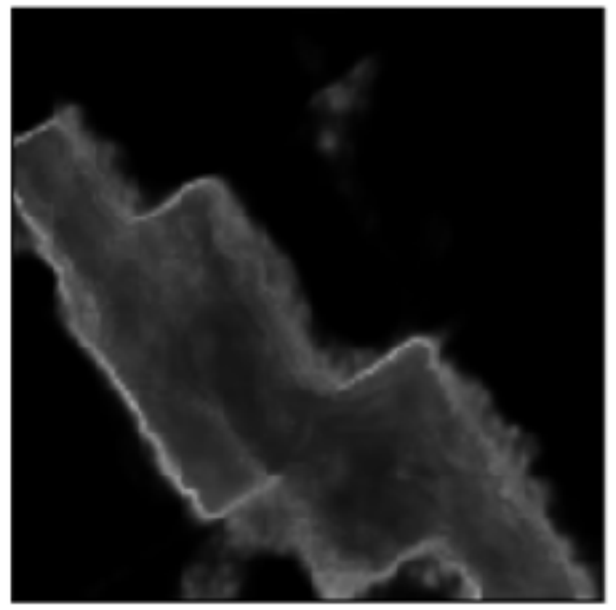
[1] **Barbano, R**., Leuschner, J., Antorán, J., Jin, B. and Hernández-Lobato, J.M., 2022. Bayesian experimental design for computed tomography with the linearised deep image prior. *arXiv*
[2] Antorán, J., **Barbano, R**., Leuschner, J., Hernández-Lobato, J.M. and Jin, B., 2022. A probabilistic deep image prior for computational tomography. *arXiv*

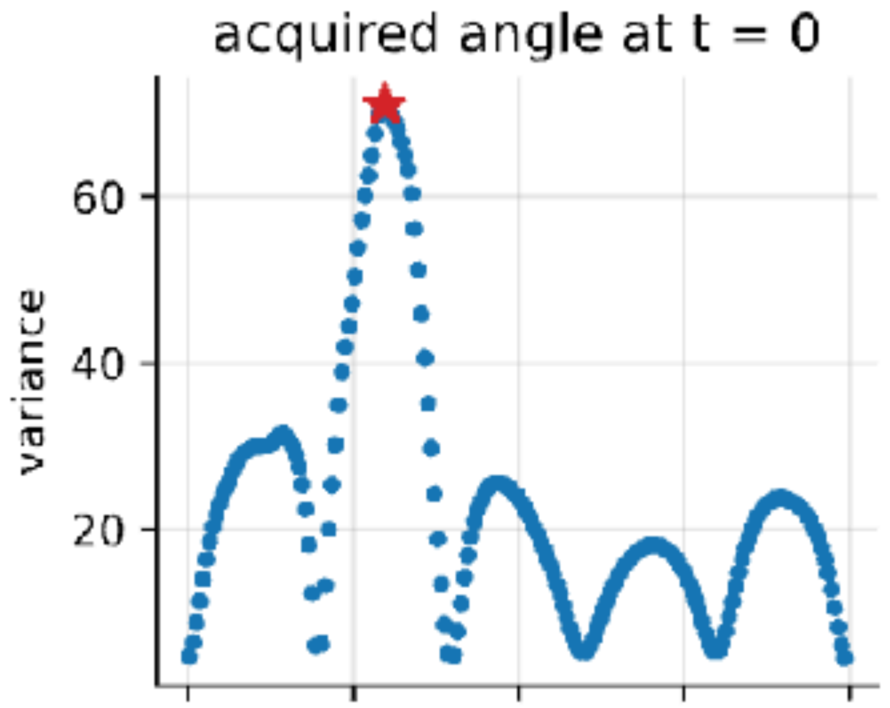
Example: Bayesian Experimental Design



Image

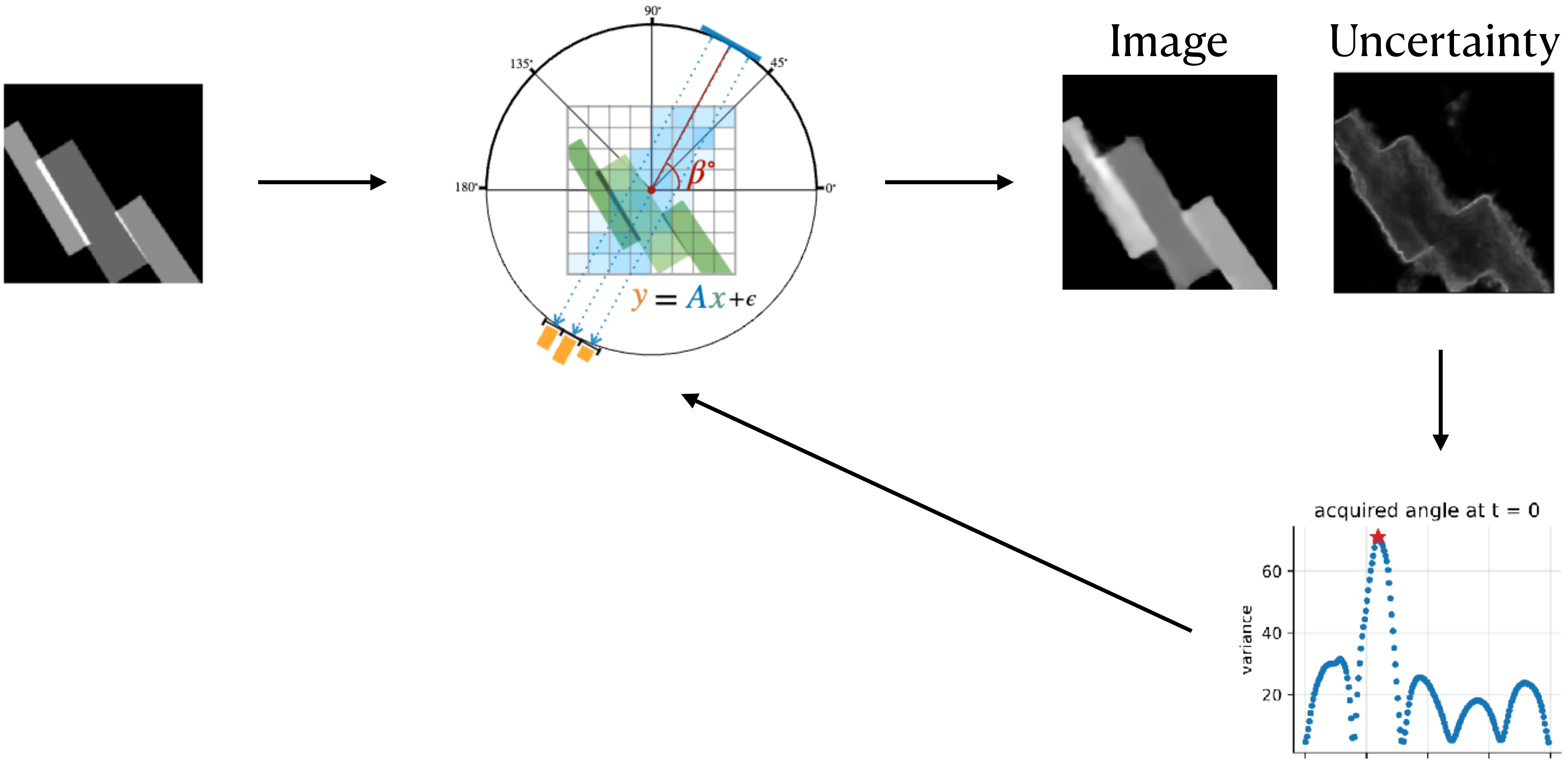
Uncertainty





[1] **Barbano, R.**, Leuschner, J., Antorán, J., Jin, B. and Hernández-Lobato, J.M., 2022. Bayesian experimental design for computed tomography with the linearised deep image prior. *arXiv*
[2] Antorán, J., **Barbano, R.**, Leuschner, J., Hernández-Lobato, J.M. and Jin, B., 2022. A probabilistic deep image prior for computational tomography. *arXiv*

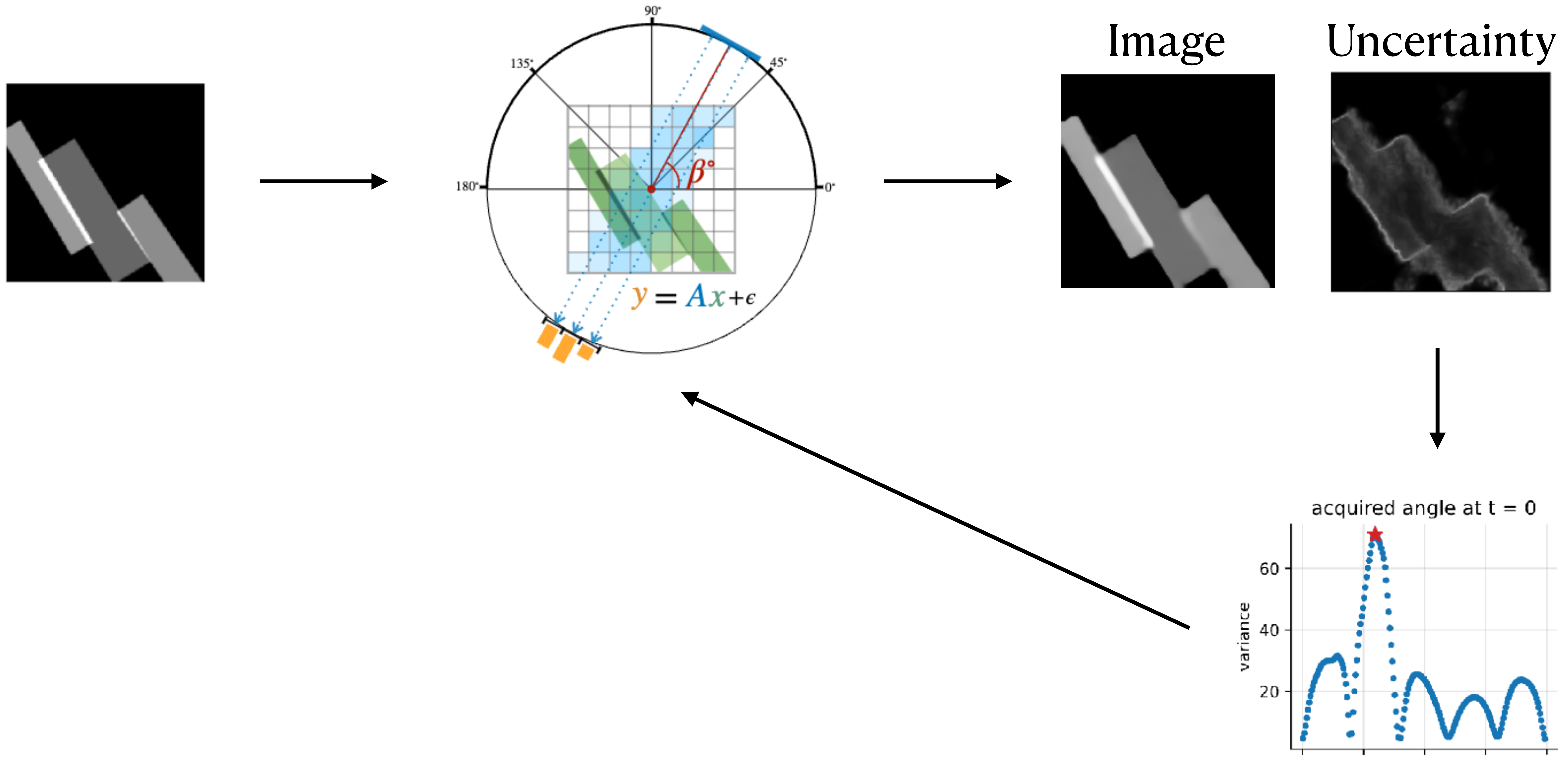
Example: Bayesian Experimental Design



[1] **Barbano, R.**, Leuschner, J., Antorán, J., Jin, B. and Hernández-Lobato, J.M., 2022. Bayesian experimental design for computed tomography with the linearised deep image prior. *arXiv*

[2] Antorán, J., **Barbano, R.**, Leuschner, J., Hernández-Lobato, J.M. and Jin, B., 2022. A probabilistic deep image prior for computational tomography. *arXiv*

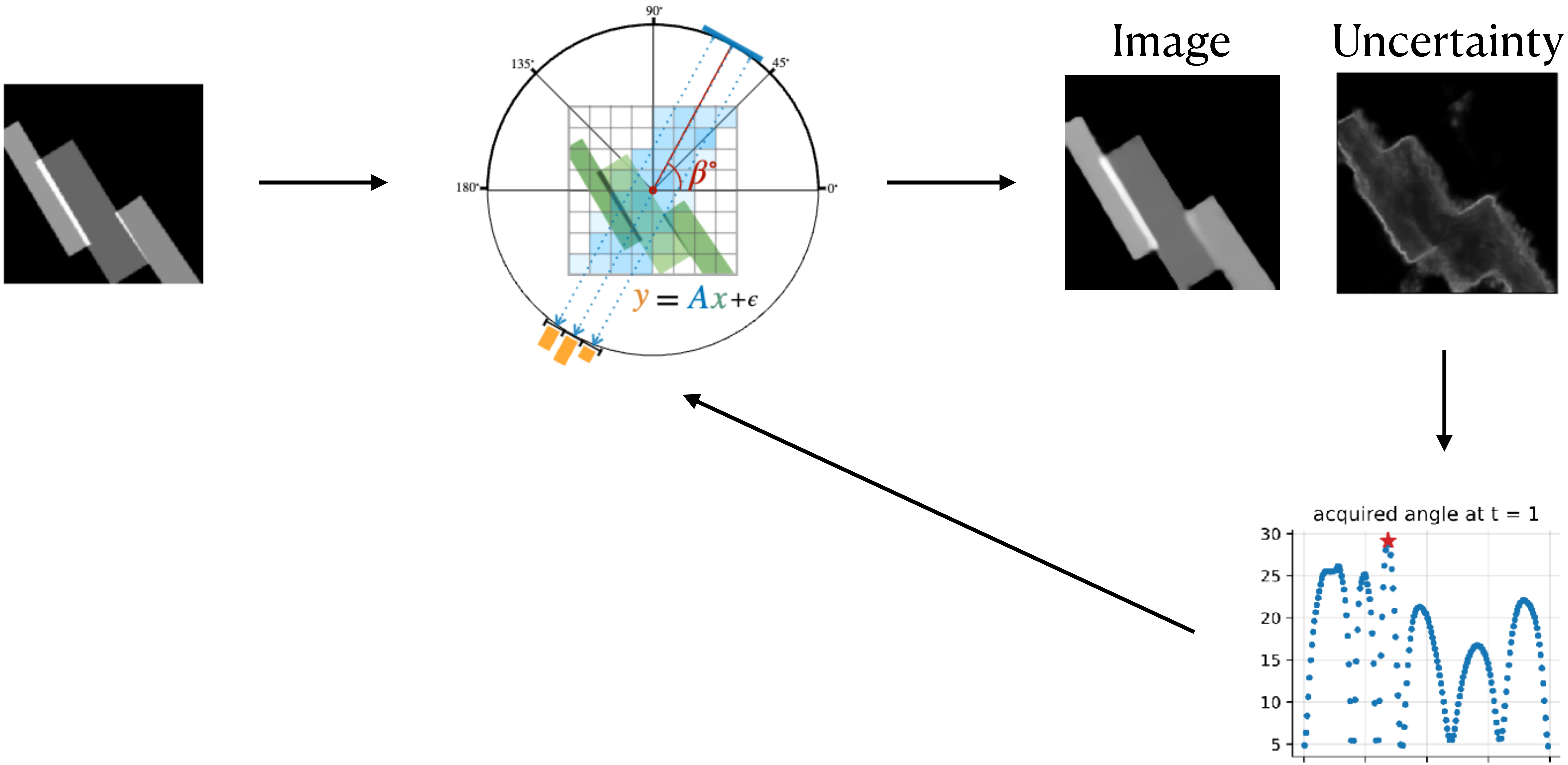
Example: Bayesian Experimental Design



[1] **Barbano, R.**, Leuschner, J., Antorán, J., Jin, B. and Hernández-Lobato, J.M., 2022. Bayesian experimental design for computed tomography with the linearised deep image prior. *arXiv*

[2] Antorán, J., **Barbano, R.**, Leuschner, J., Hernández-Lobato, J.M. and Jin, B., 2022. A probabilistic deep image prior for computational tomography. *arXiv*

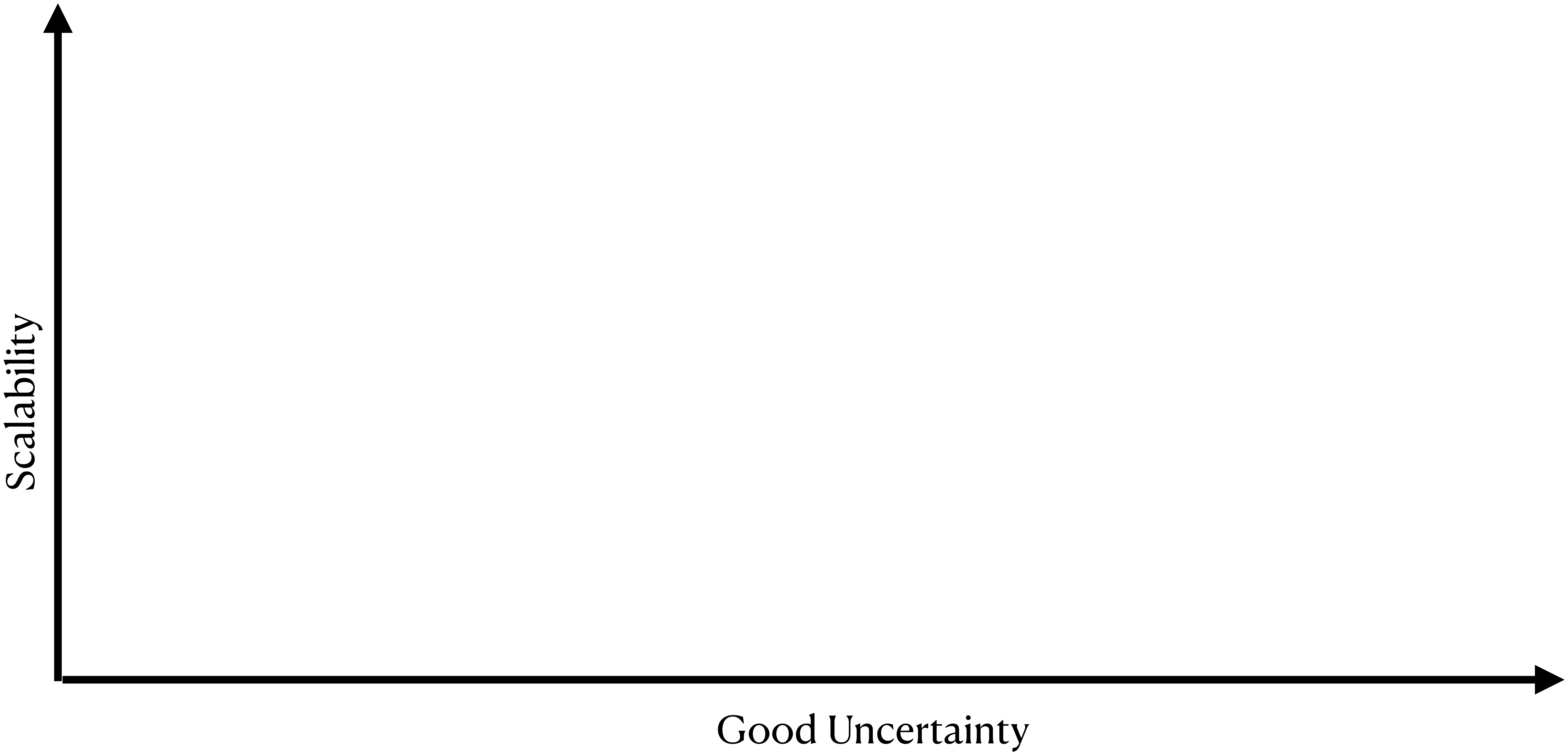
Example: Bayesian Experimental Design



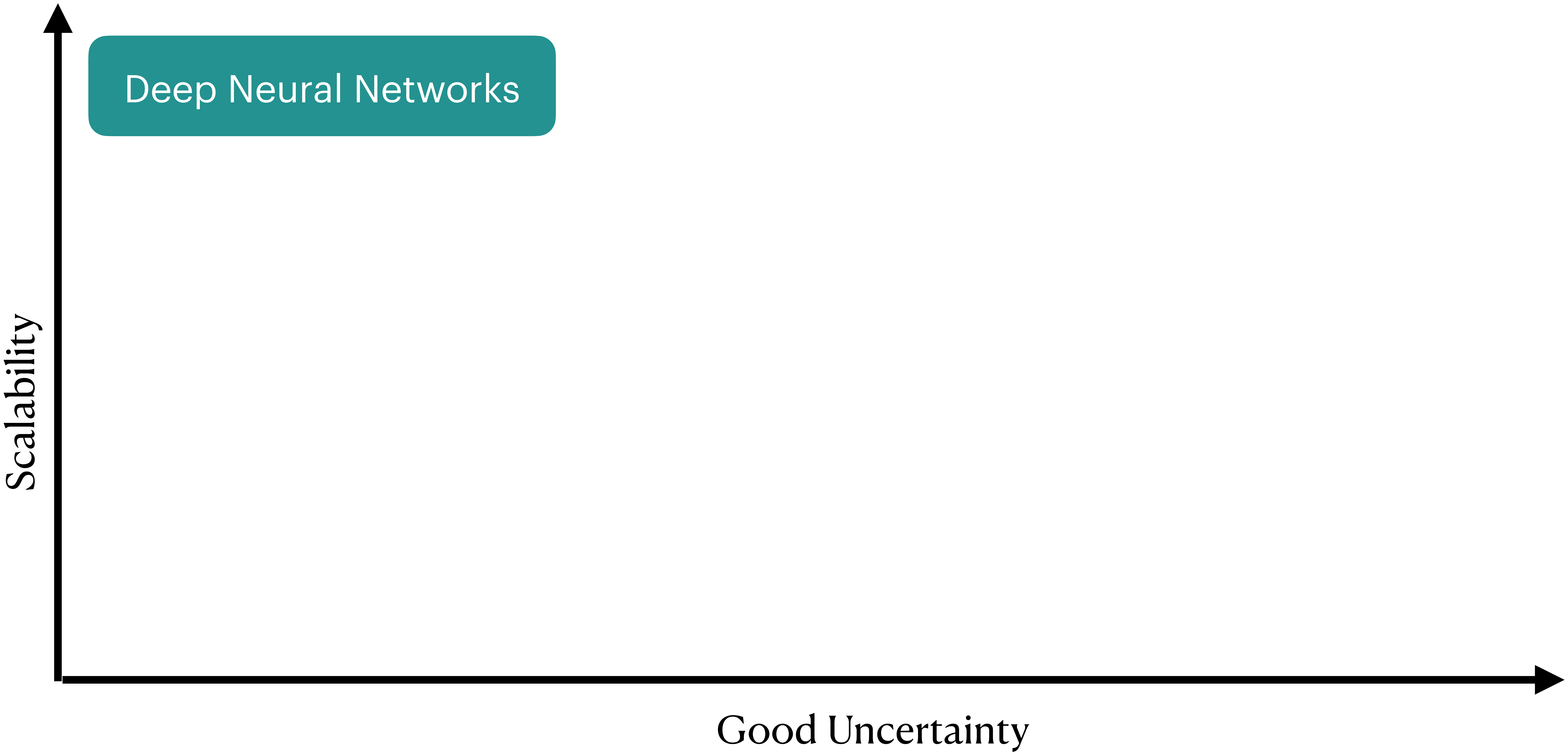
[1] **Barbano, R.**, Leuschner, J., Antorán, J., Jin, B. and Hernández-Lobato, J.M., 2022. Bayesian experimental design for computed tomography with the linearised deep image prior. *arXiv*
[2] Antorán, J., **Barbano, R.**, Leuschner, J., Hernández-Lobato, J.M. and Jin, B., 2022. A probabilistic deep image prior for computational tomography. *arXiv*

The Bayesian Model Landscape

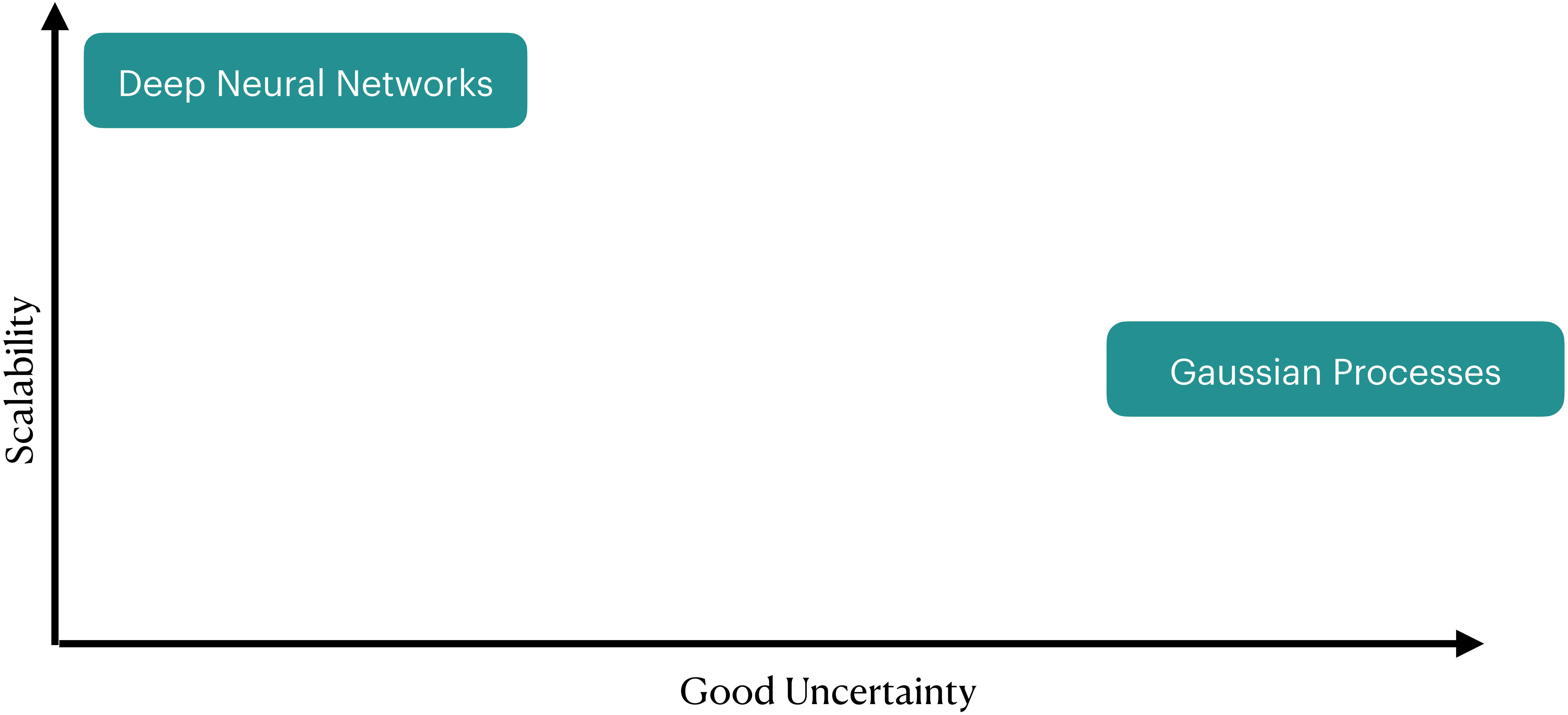
The Bayesian Model Landscape



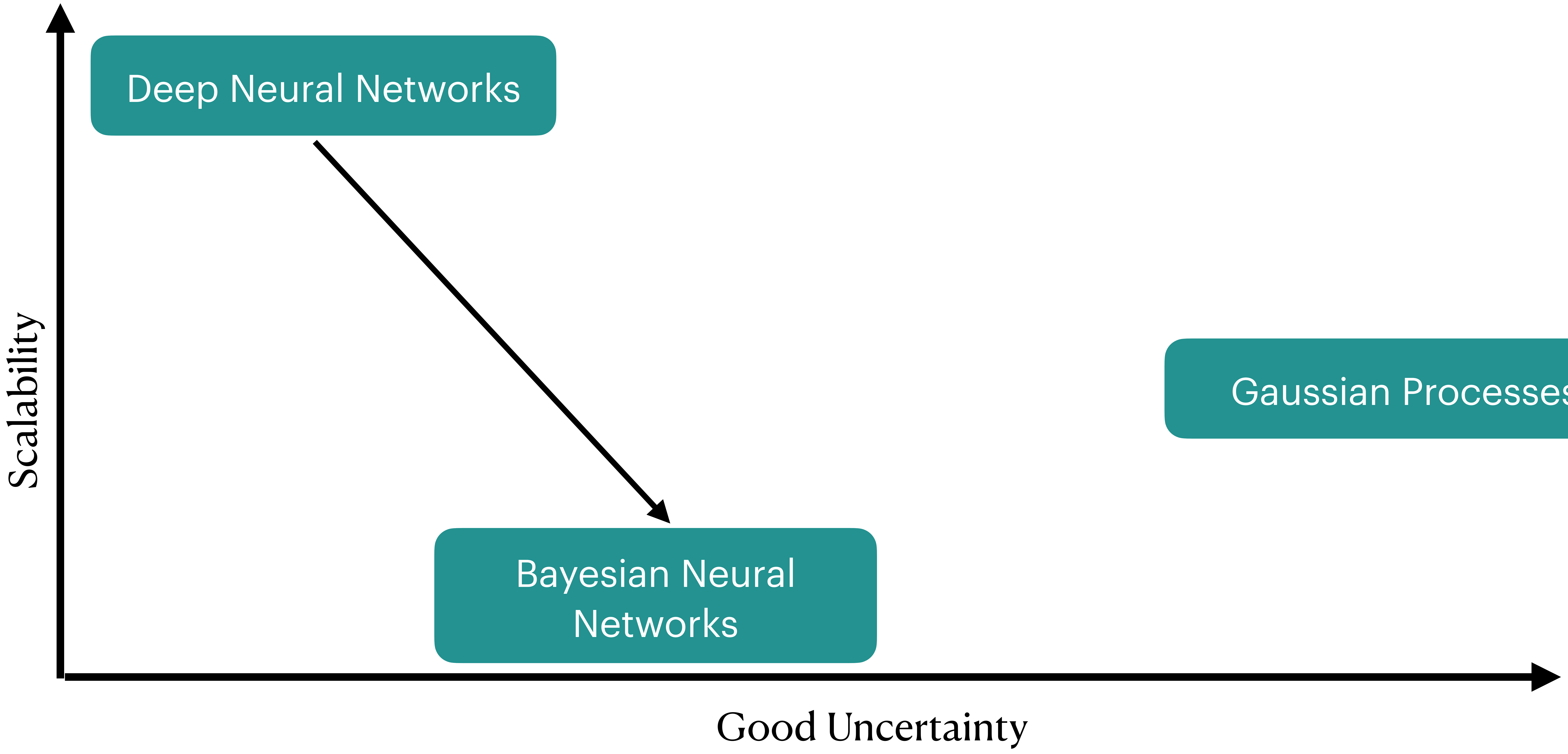
The Bayesian Model Landscape



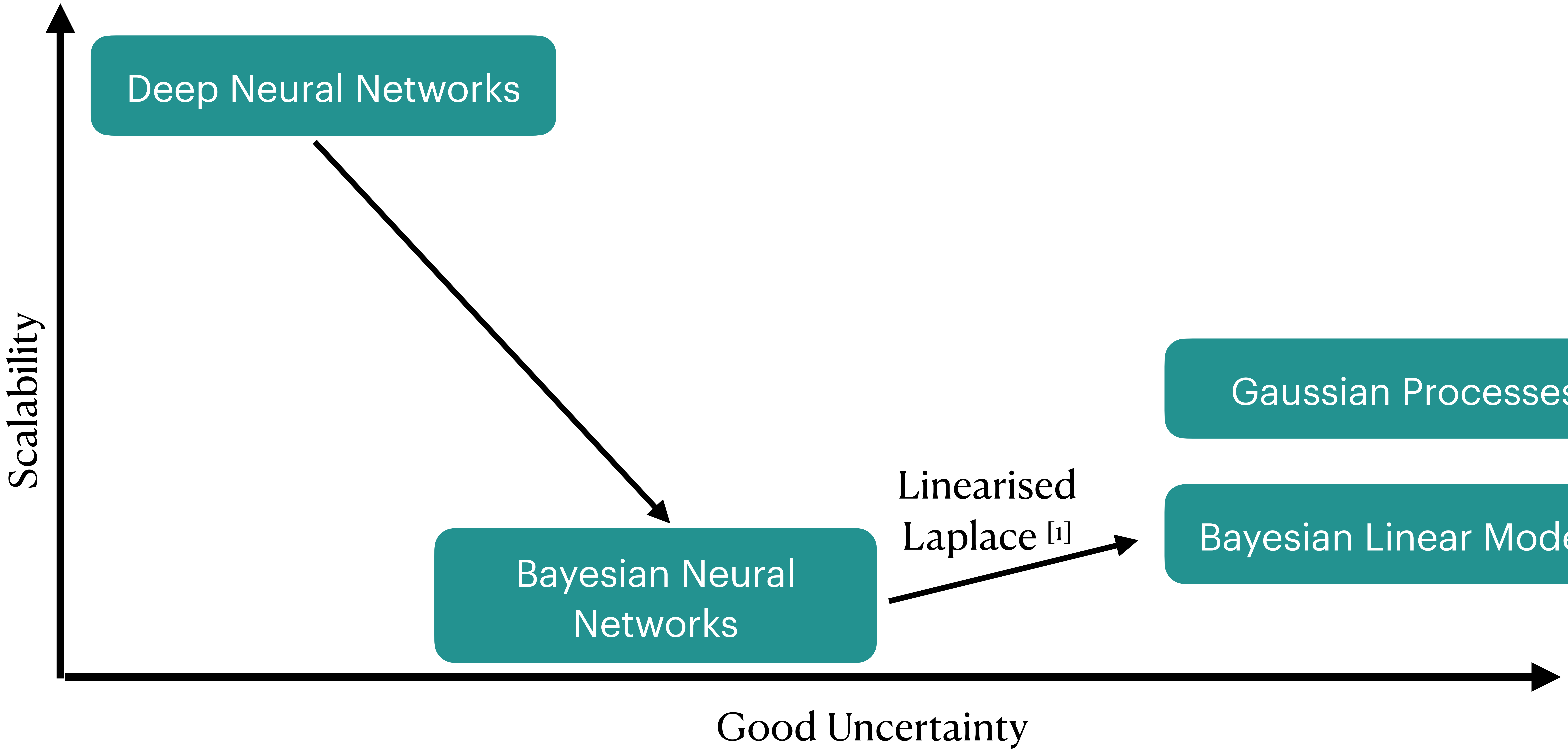
The Bayesian Model Landscape



The Bayesian Model Landscape

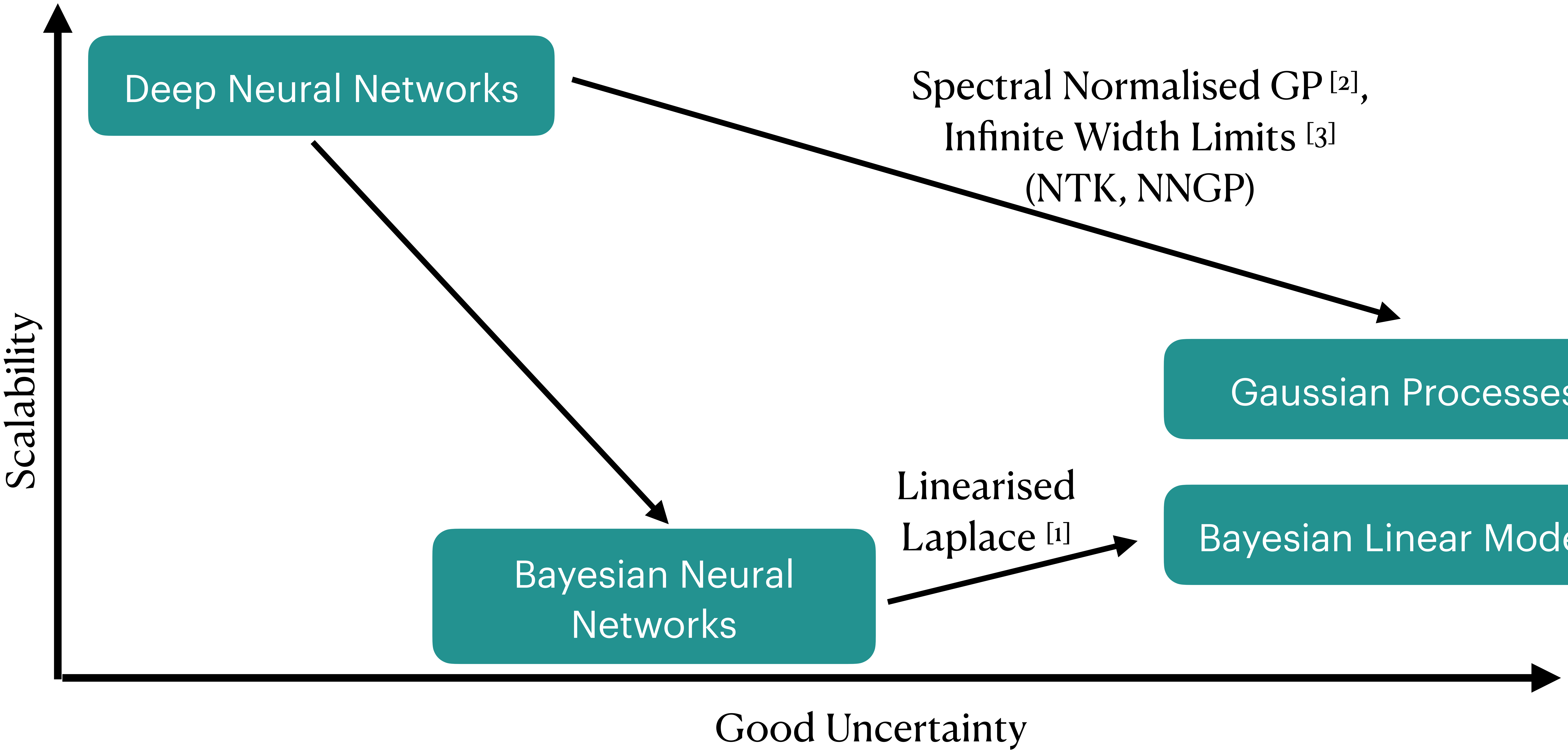


The Bayesian Model Landscape



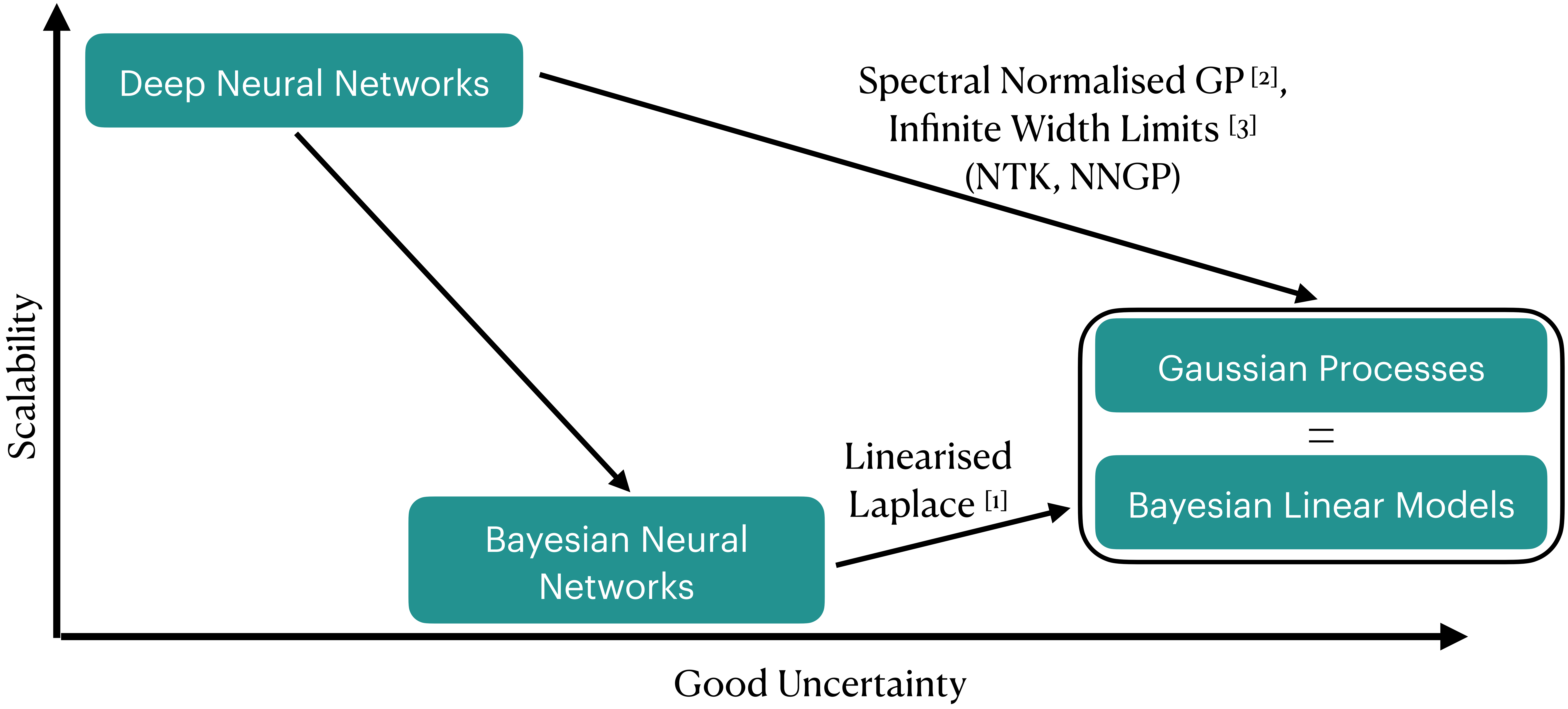
[1] **Padhy, S.***, Antorán, J.,*, **Barbano, R.**, Nalisnick, E., ... and Hernández-Lobato, J.M., Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***

The Bayesian Model Landscape



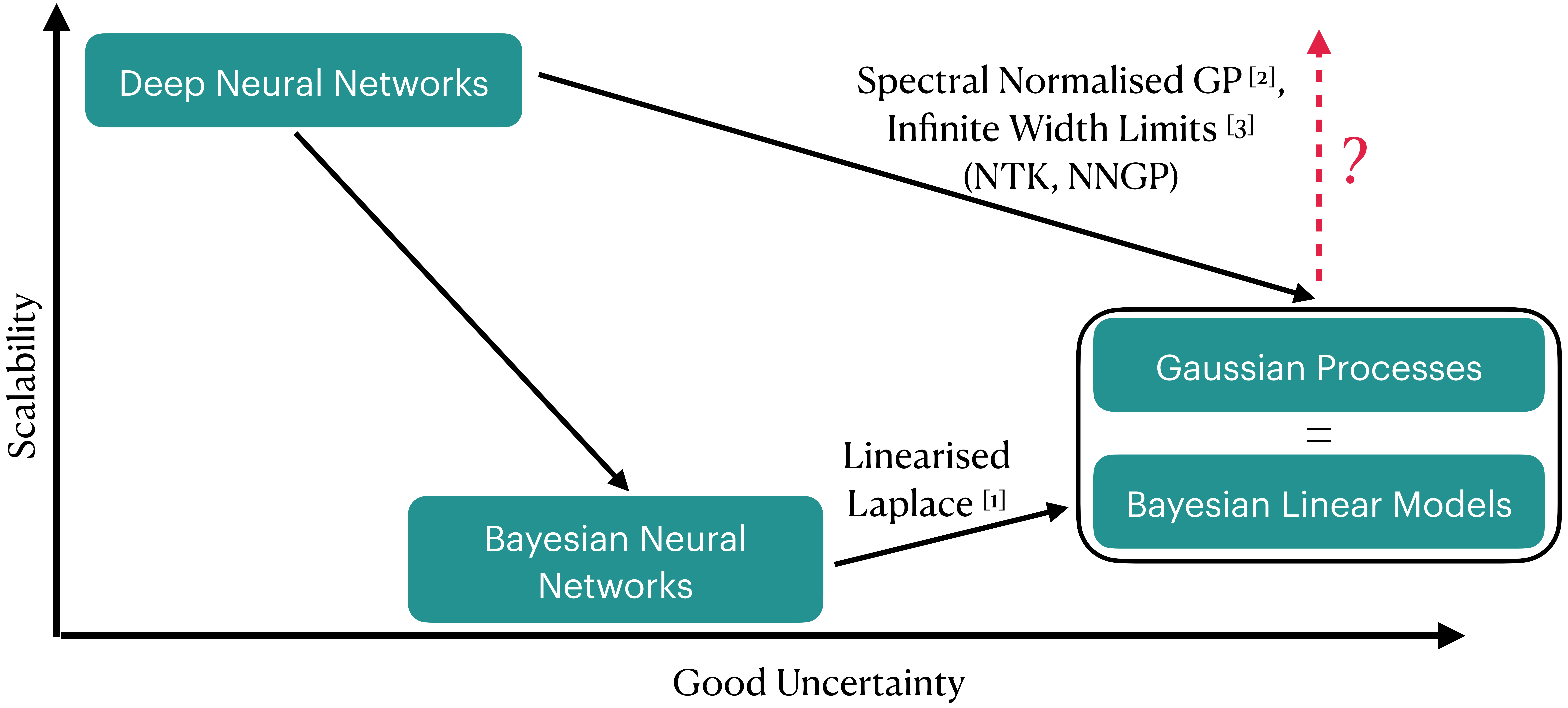
[1] **Padhy, S.***, Antorán, J.,*, **Barbano, R.**, Nalisnick, E., ... and Hernández-Lobato, J.M., Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***
[2] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***
[3] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

The Bayesian Model Landscape



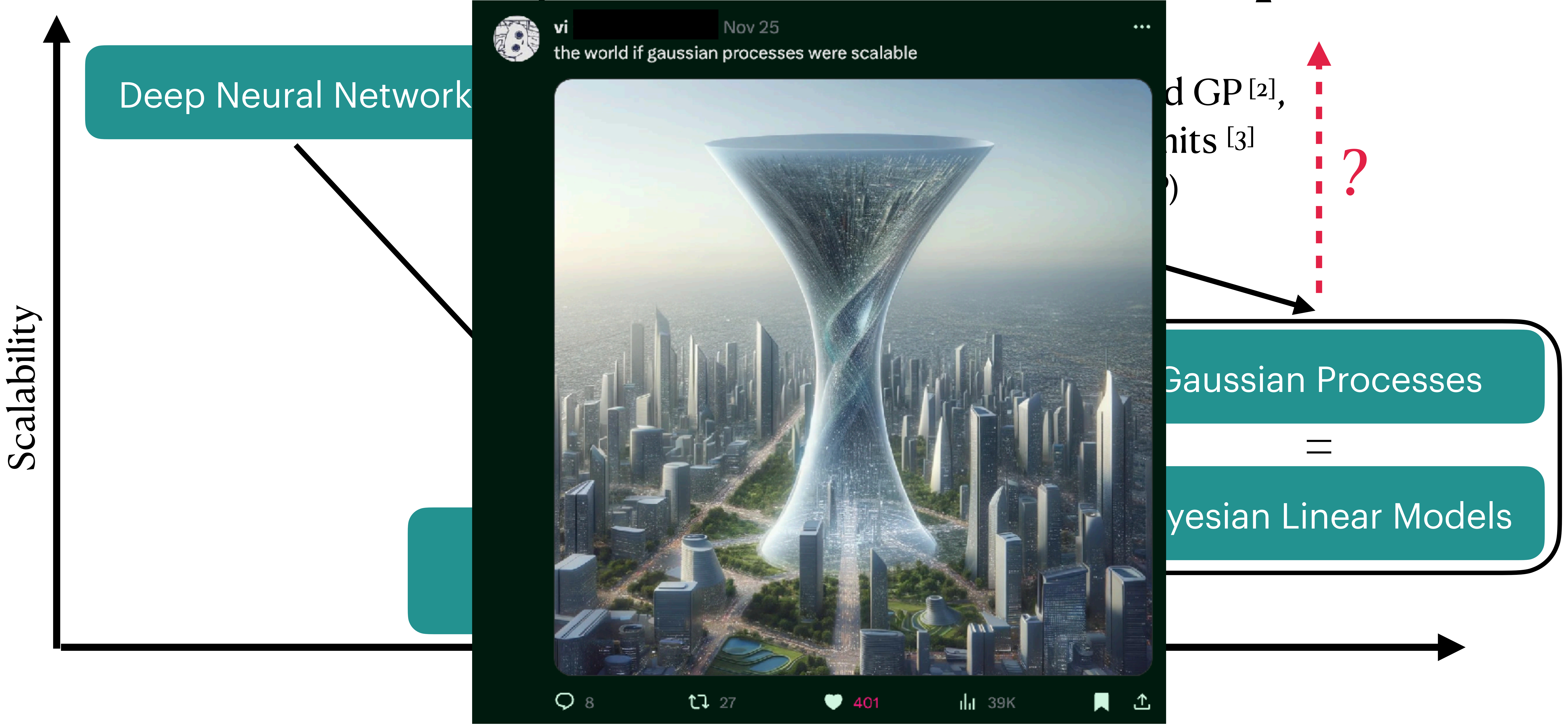
[1] **Padhy, S.***, Antorán, J.,*, **Barbano, R.**, Nalisnick, E., ... and Hernández-Lobato, J.M., Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***
[2] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***
[3] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

The Bayesian Model Landscape



[1] **Padhy, S.***, Antorán, J.,*, **Barbano, R.**, Nalisnick, E., ... and Hernández-Lobato, J.M., Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***
[2] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***
[3] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

The Bayesian Model Landscape



[1] **Padhy, S.***, Antorán, J.,*, **Barbano, R.**, Nalisnick, E., ... and Hernández-Lobato, J.M., Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***
[2] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***
[3] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

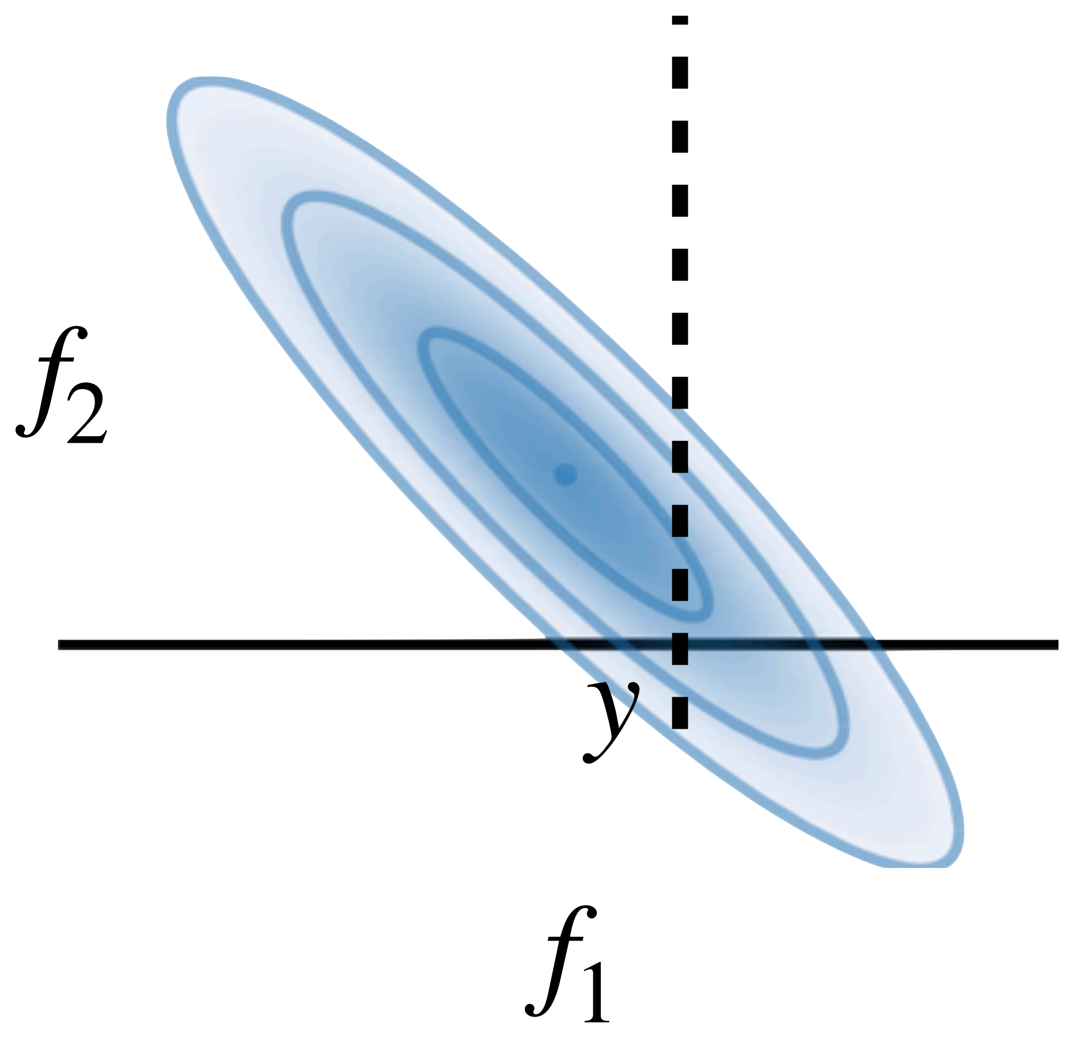
Gaussian Processes: A Primer

- A very flexible *non-parametric* family of models that are entirely defined by pairwise correlations between points
- **Idea:** All datapoints are jointly Gaussian distributed, observing some points conditions the remaining points on them

Gaussian Processes: A Primer

- A very flexible *non-parametric* family of models that are entirely defined by pairwise correlations between points
- **Idea:** All datapoints are jointly Gaussian distributed, observing some points conditions the remaining points on them

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}\right)$$

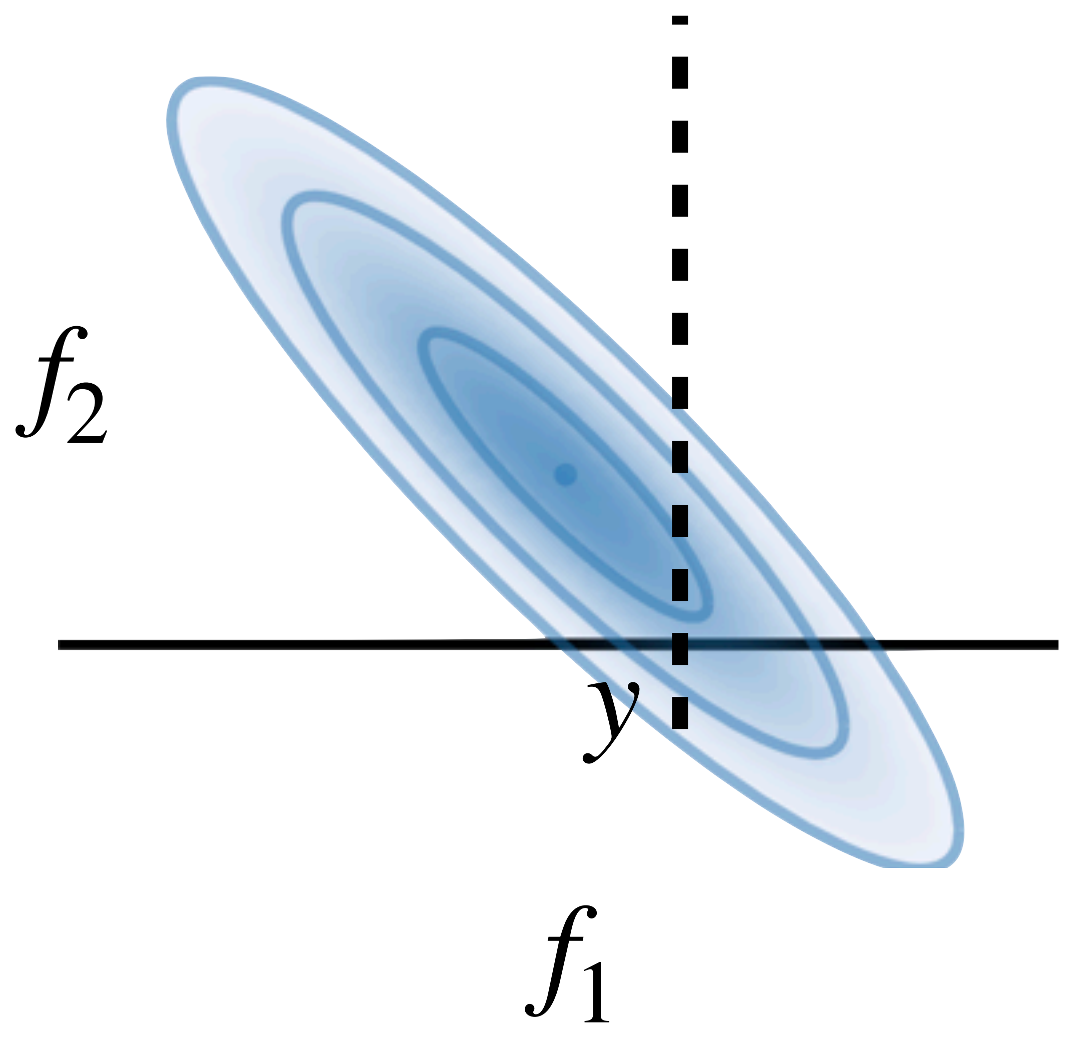


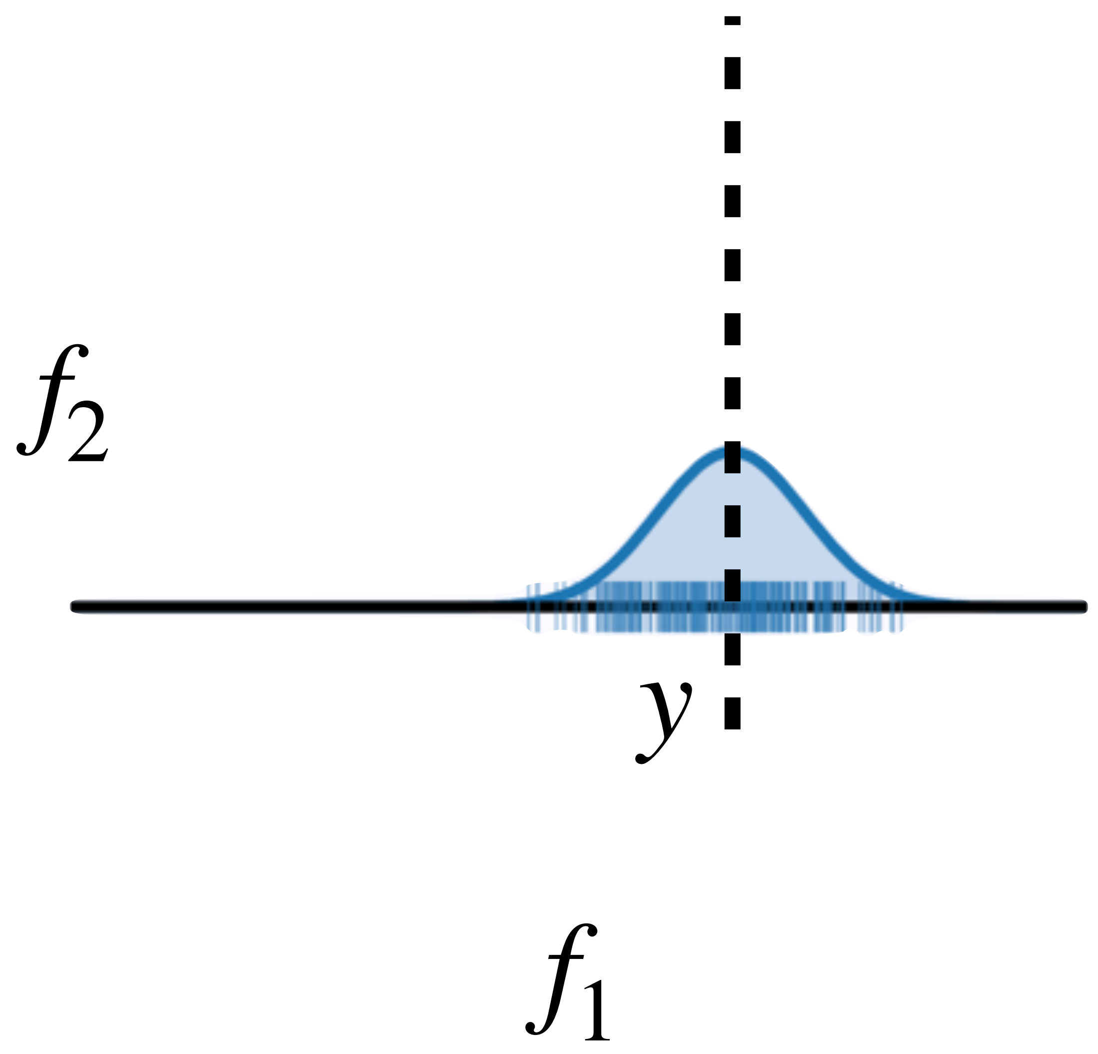
Gaussian Processes: A Primer

- A very flexible *non-parametric* family of models that are entirely defined by pairwise correlations between points
- **Idea:** All datapoints are jointly Gaussian distributed, observing some points conditions the remaining points on them

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix} \right)$$

$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$



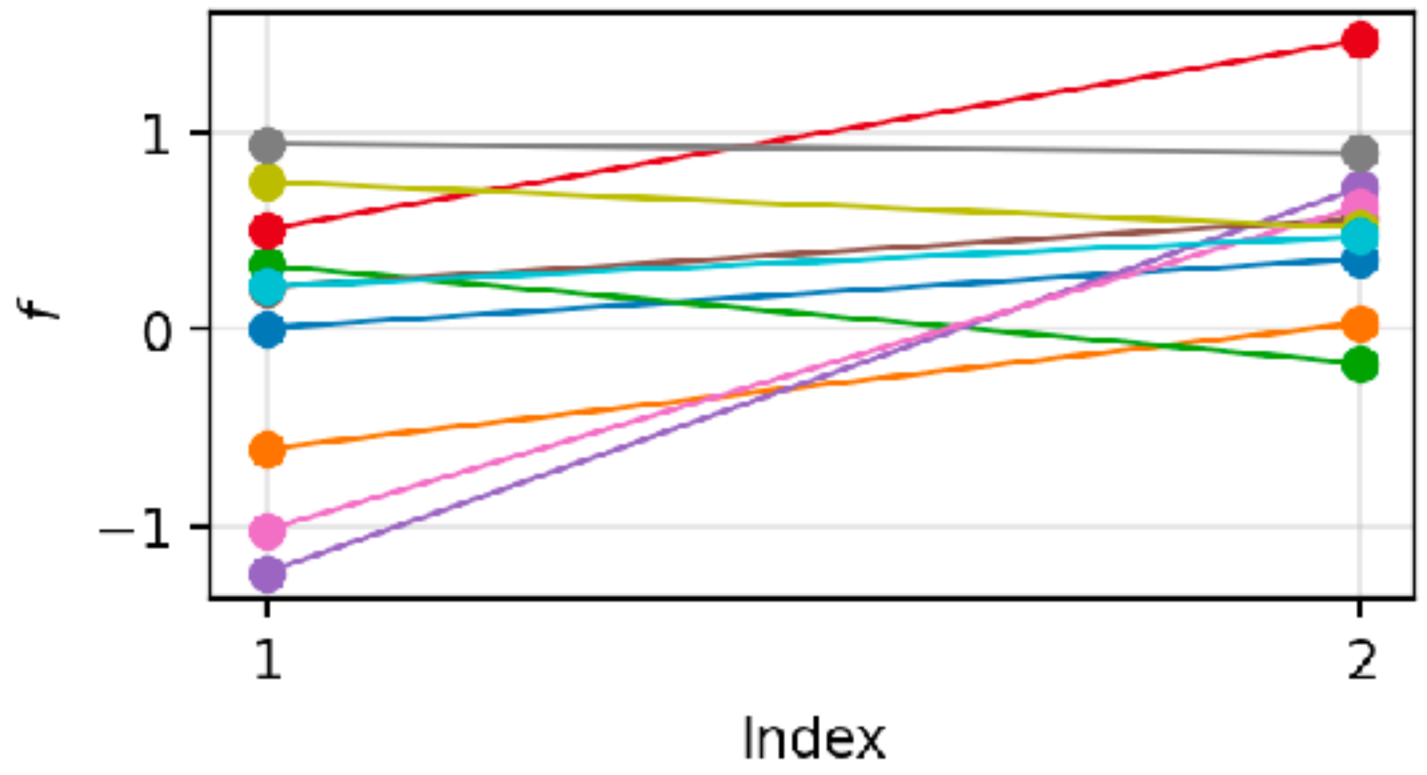


Gaussian Processes: A Primer

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$

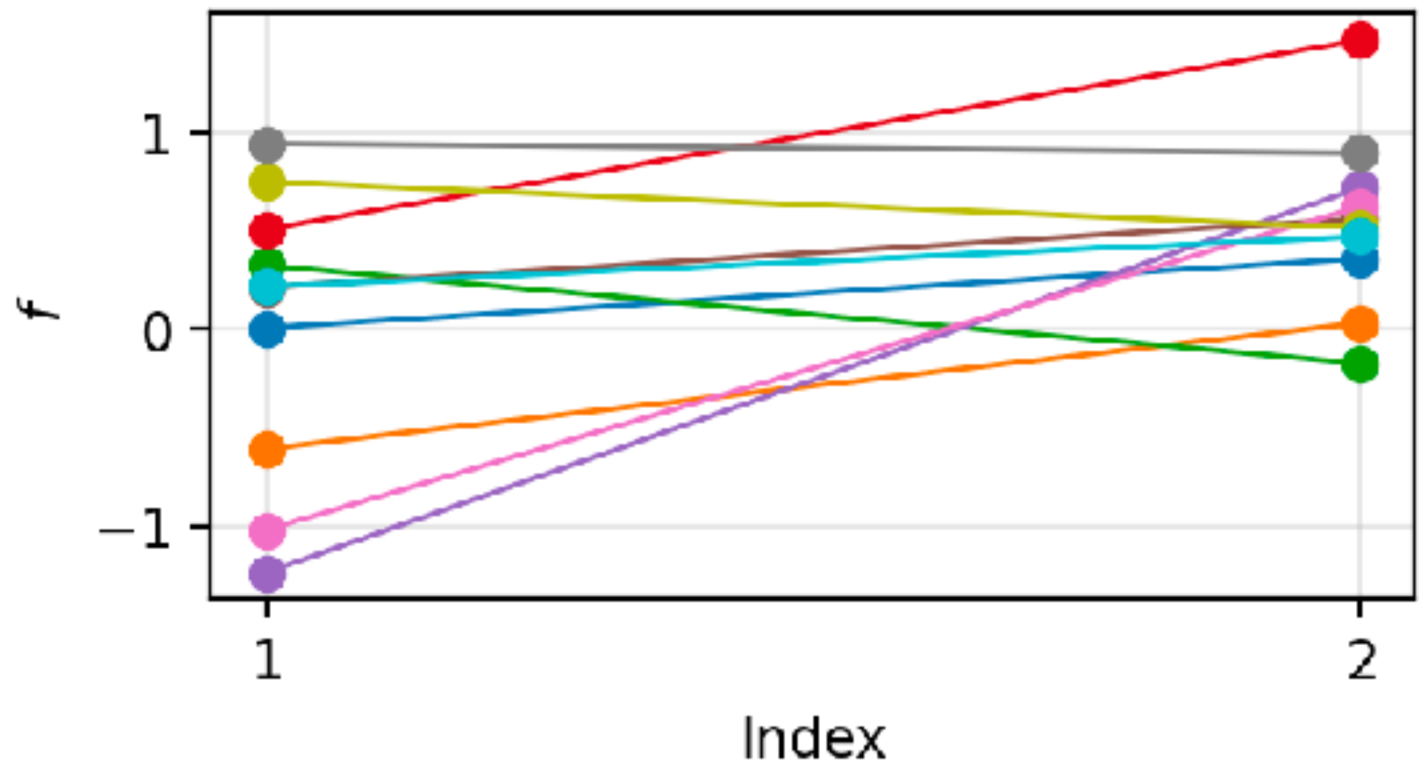
Gaussian Processes: A Primer

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



Gaussian Processes: A Primer

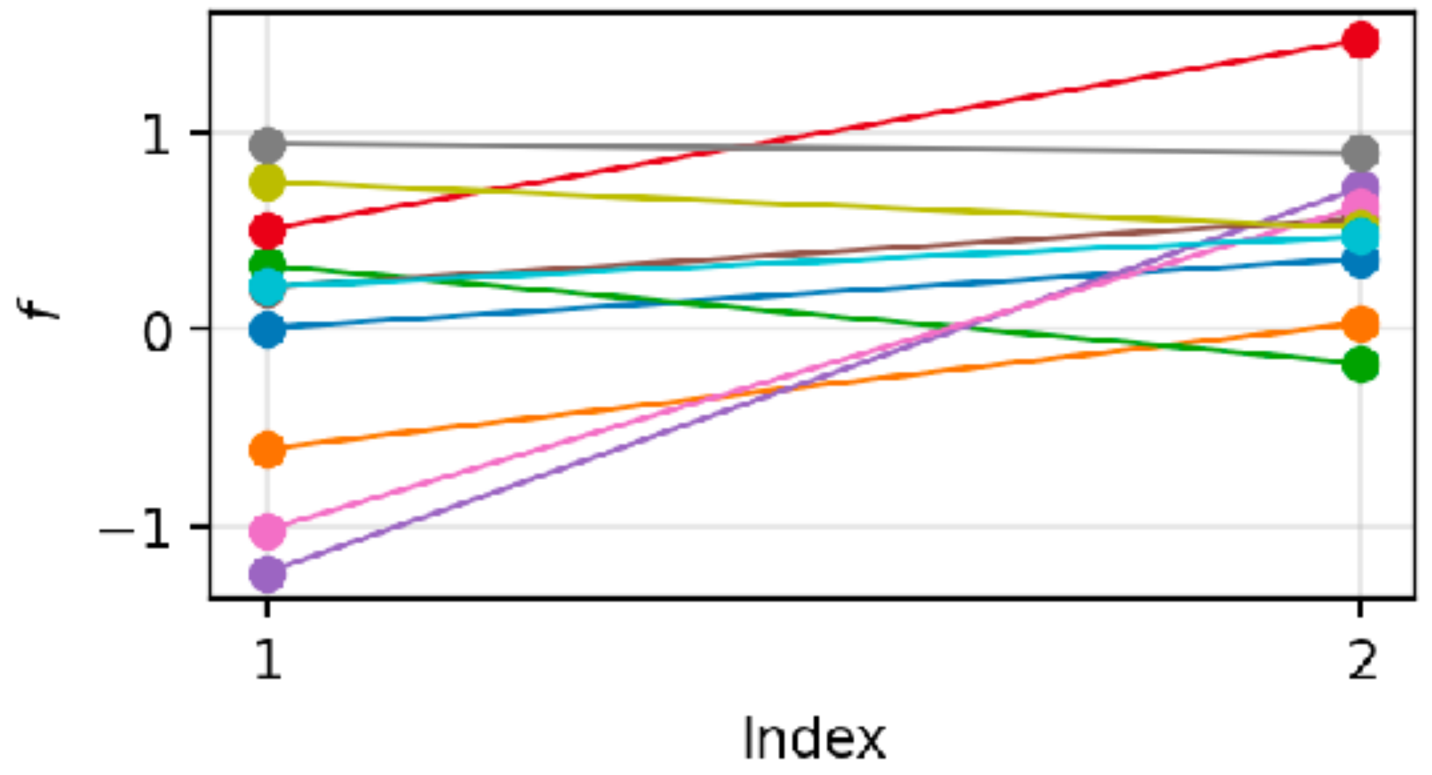
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix} \right)$$



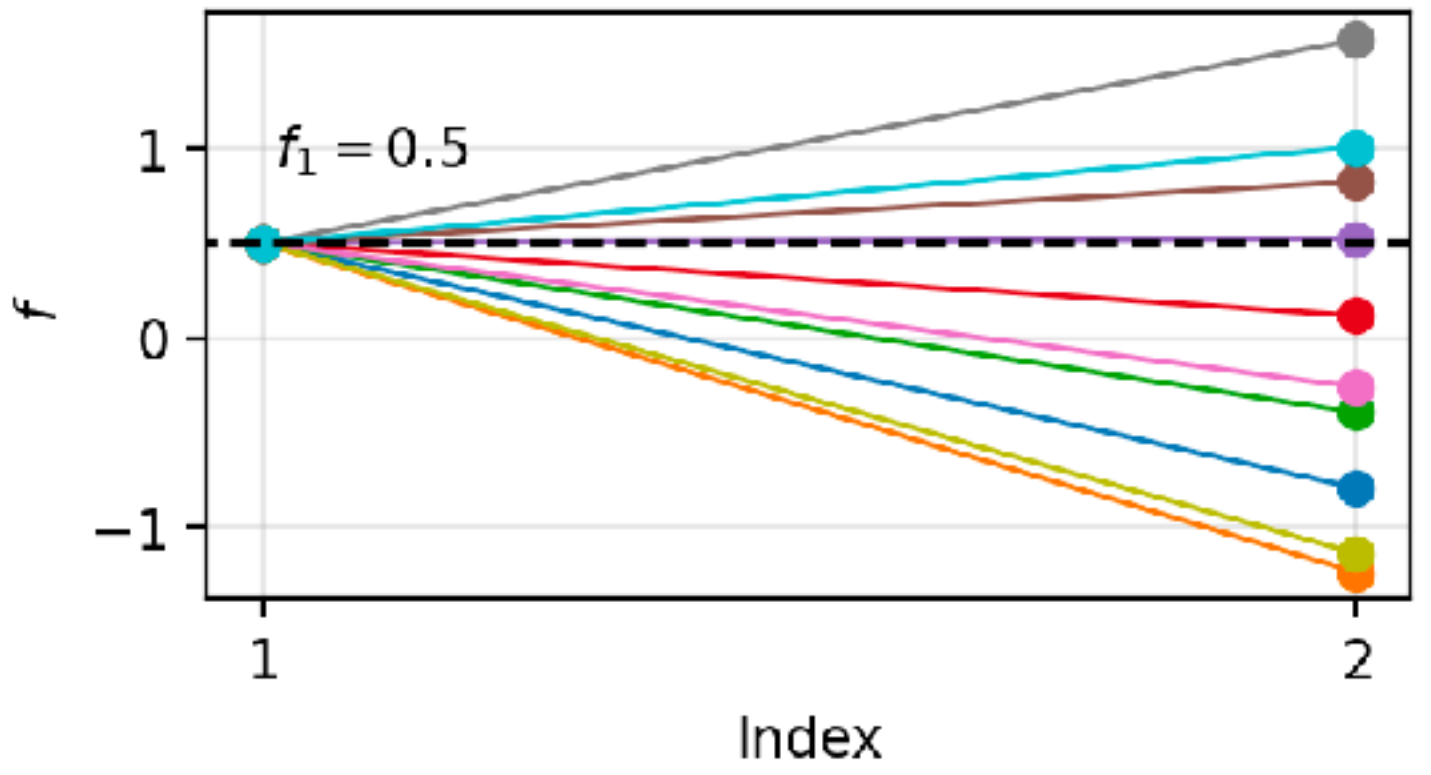
$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$

Gaussian Processes: A Primer

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



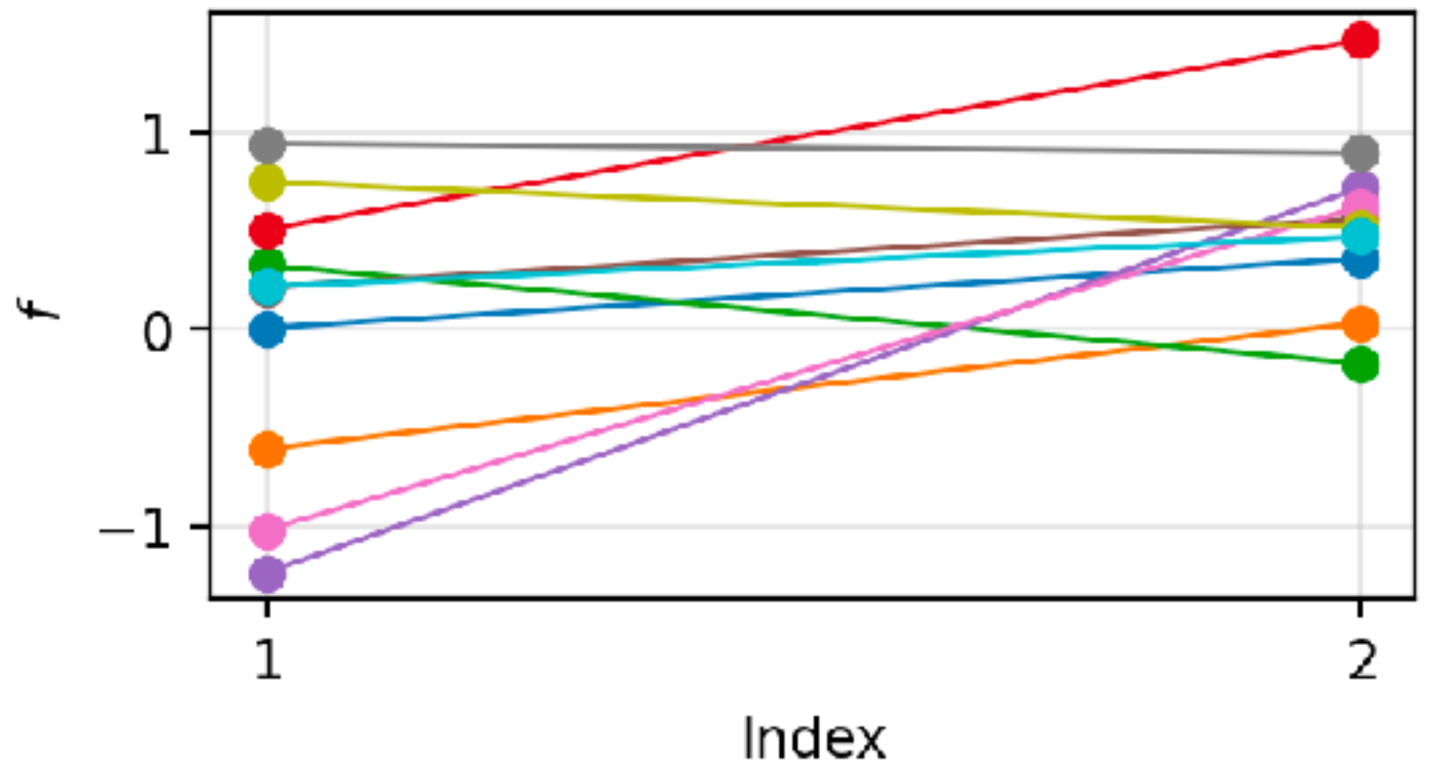
$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$

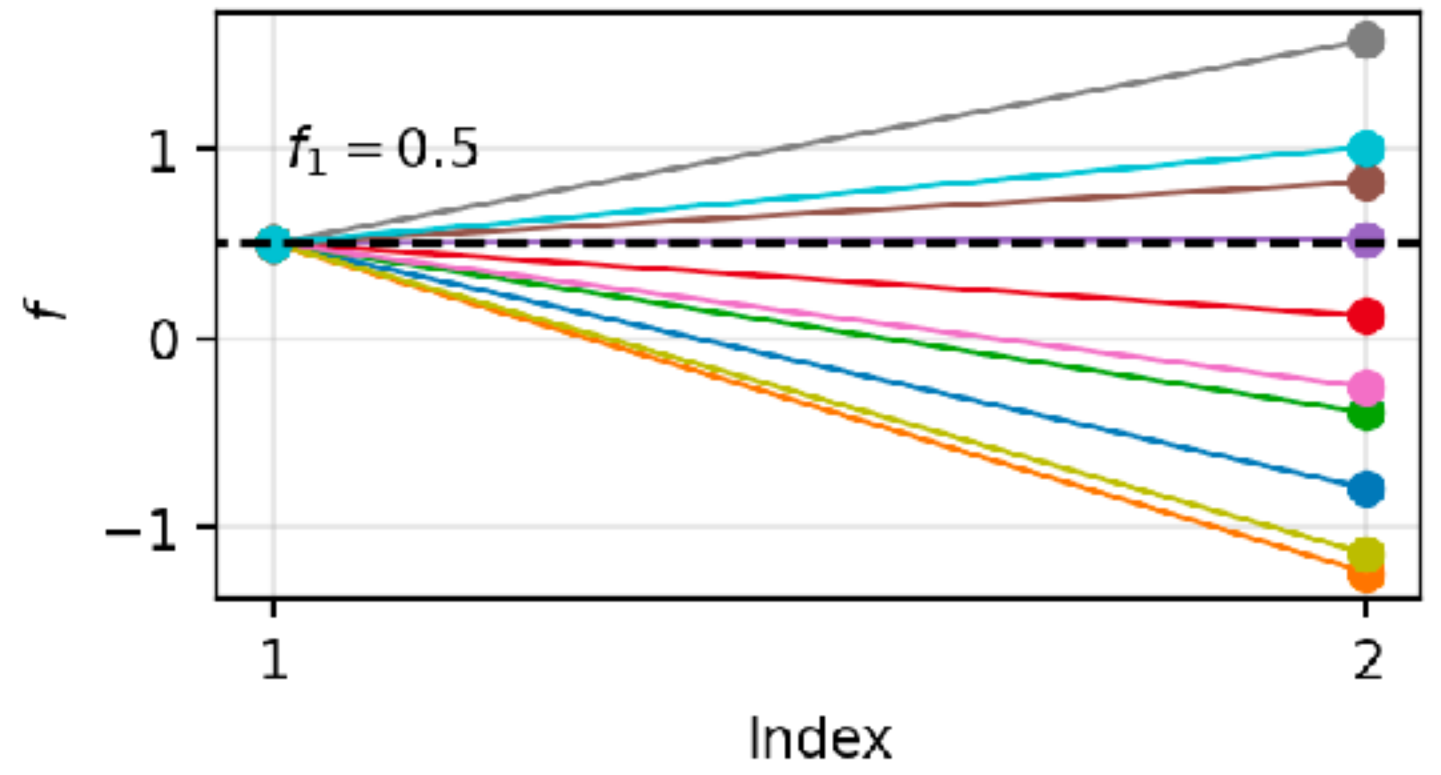


Gaussian Processes: A Primer

$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$

$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$

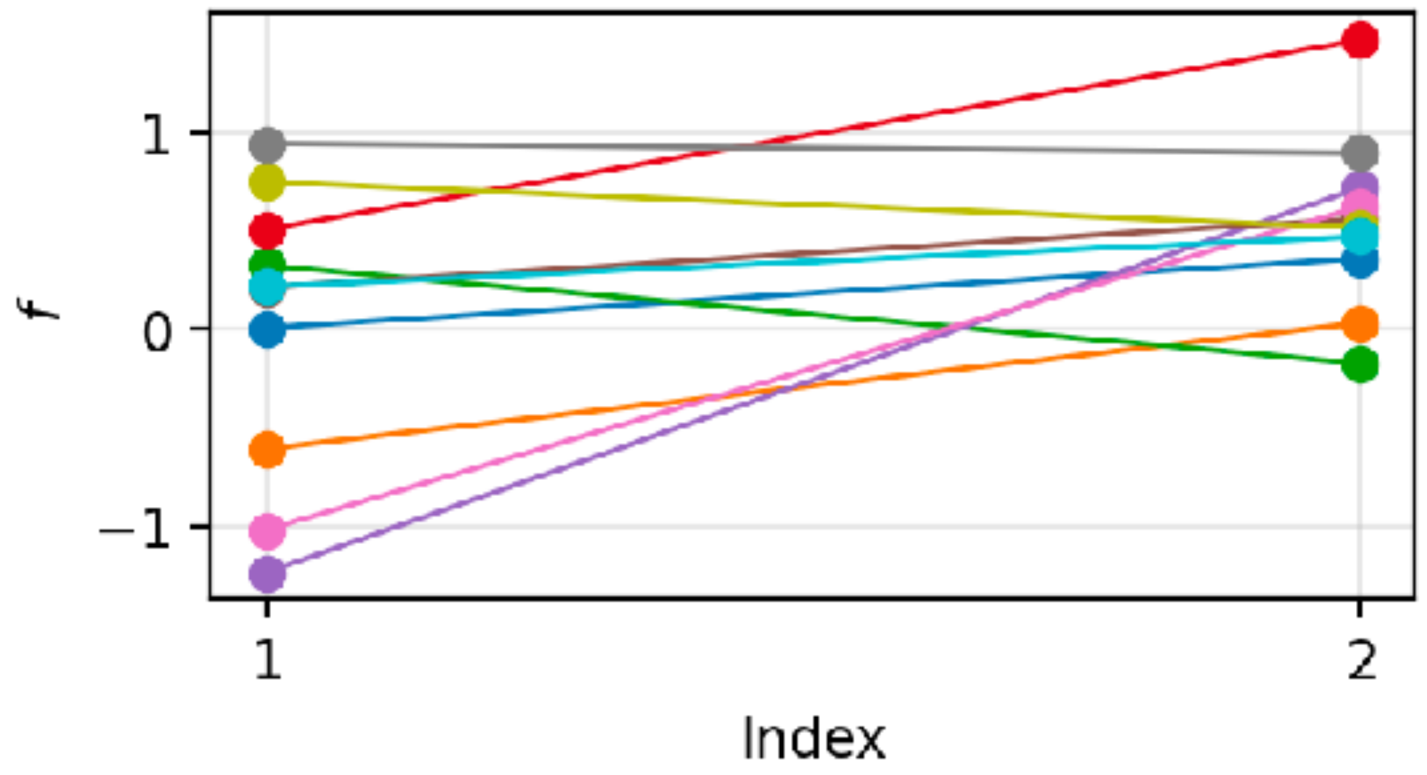




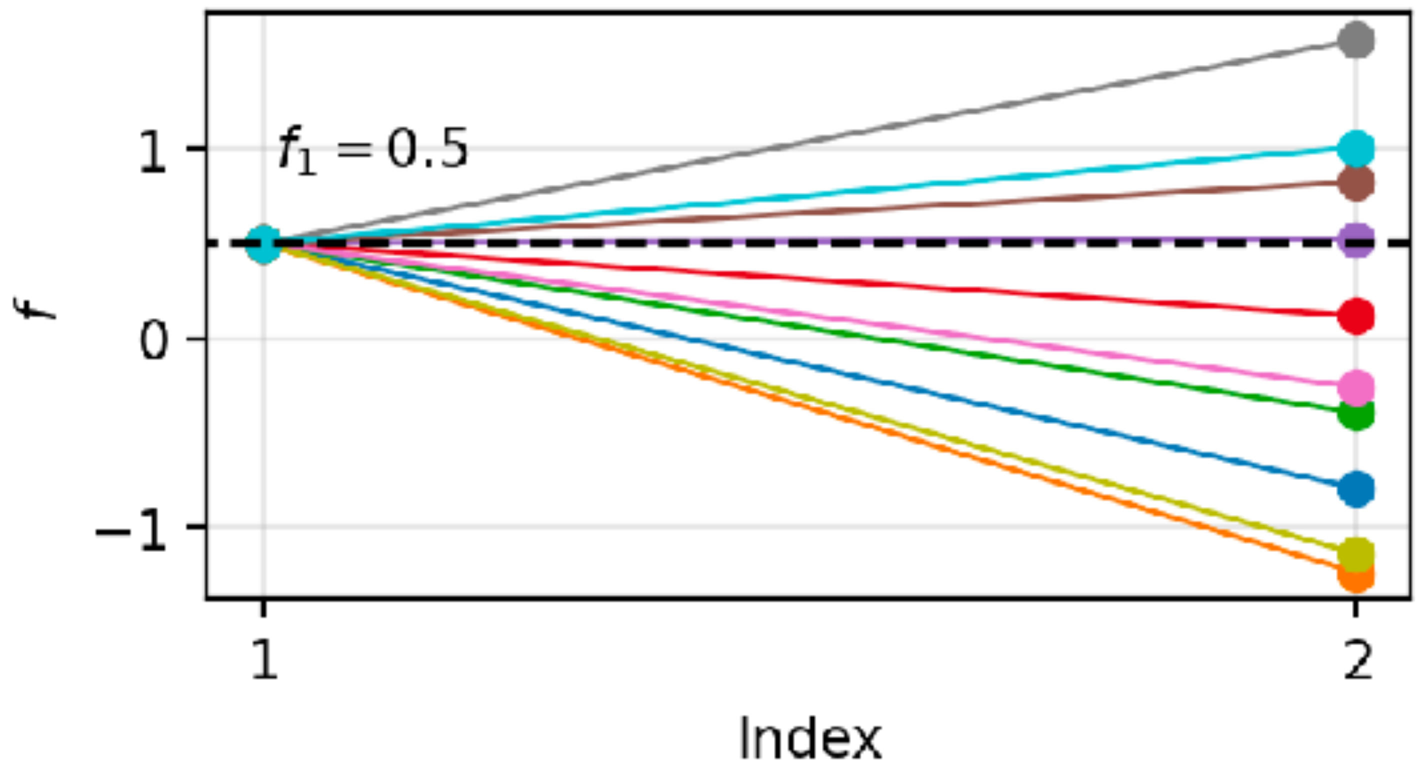
$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{10} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_{10} \end{pmatrix}, \mathbf{K}\right), \qquad K_{ij} = \exp(-(i-j)^2)$$

Gaussian Processes: A Primer

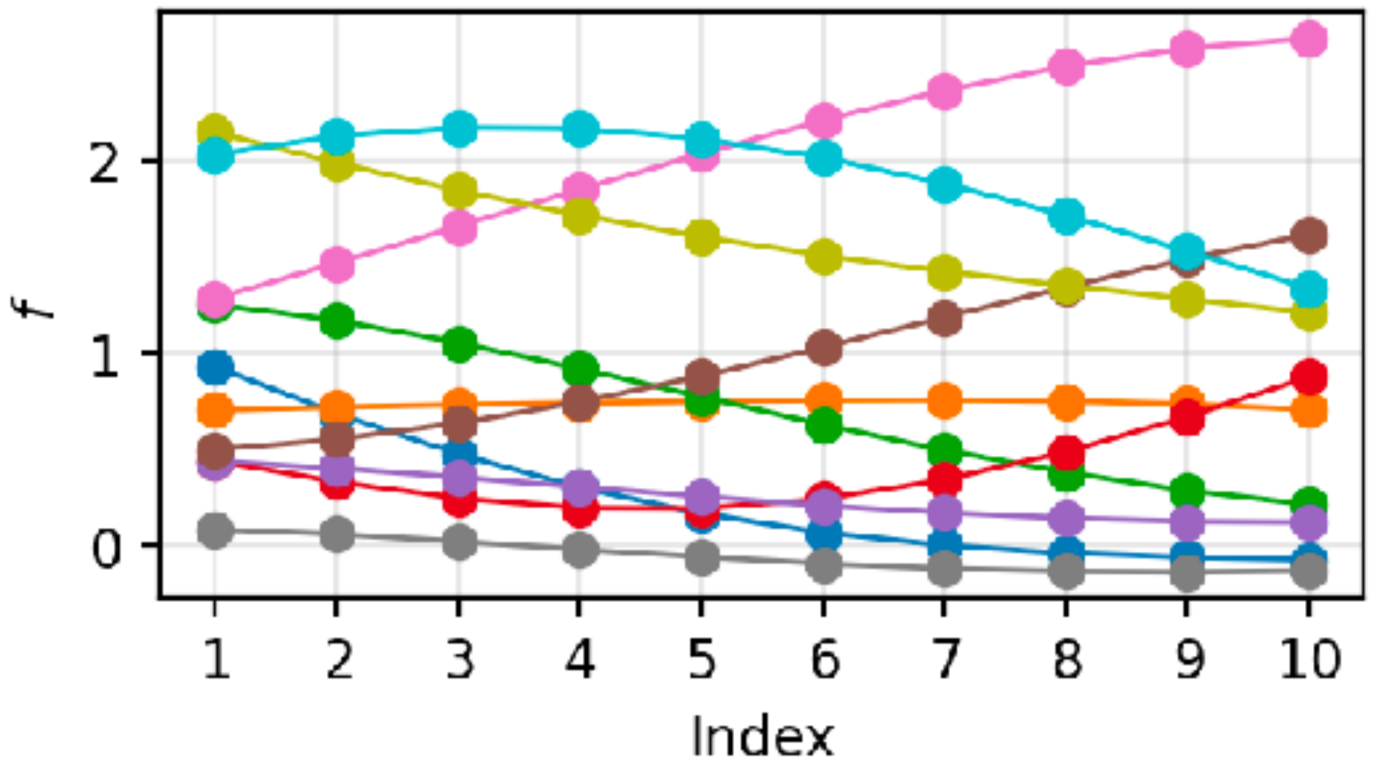
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$

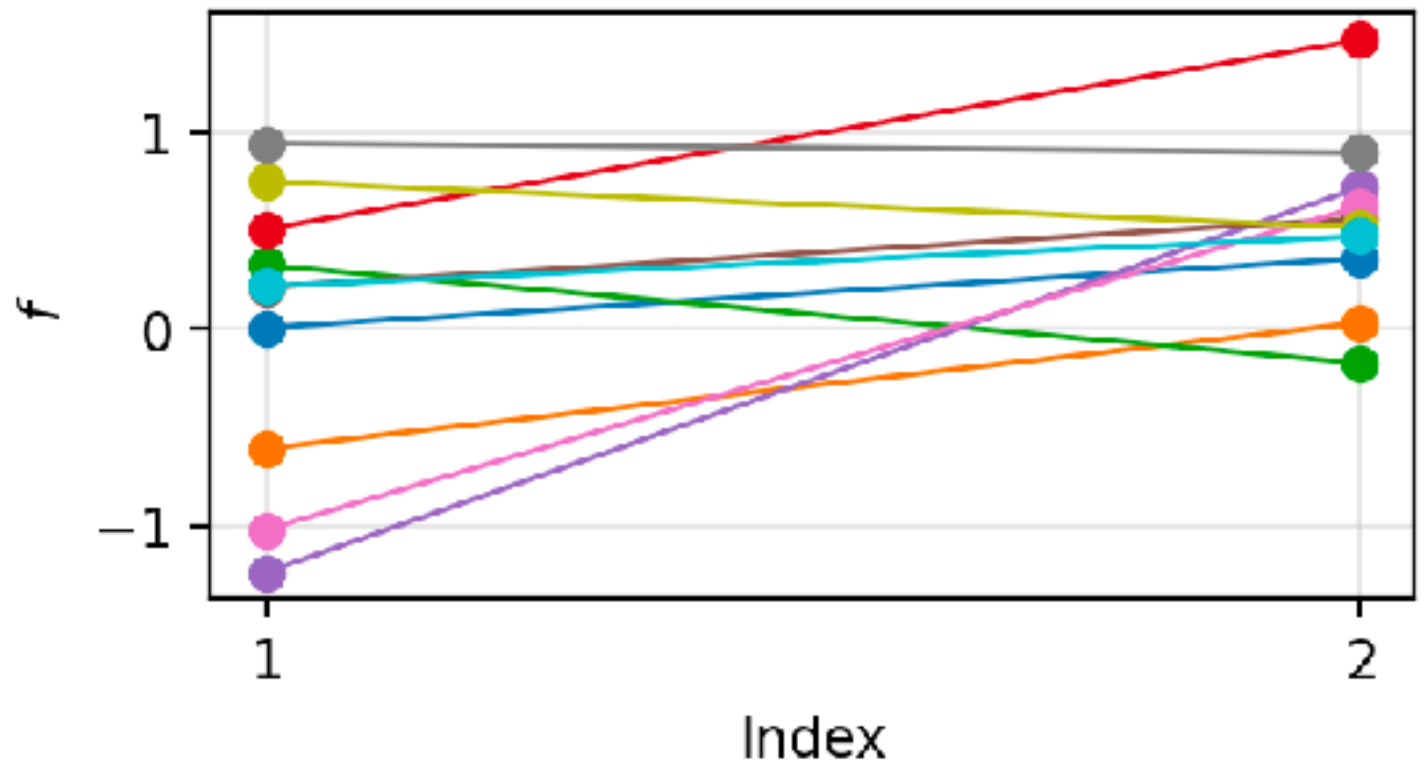


$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{10} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_{10} \end{pmatrix}, \mathbf{K}\right), \qquad K_{ij} = \exp(-(i-j)^2)$$

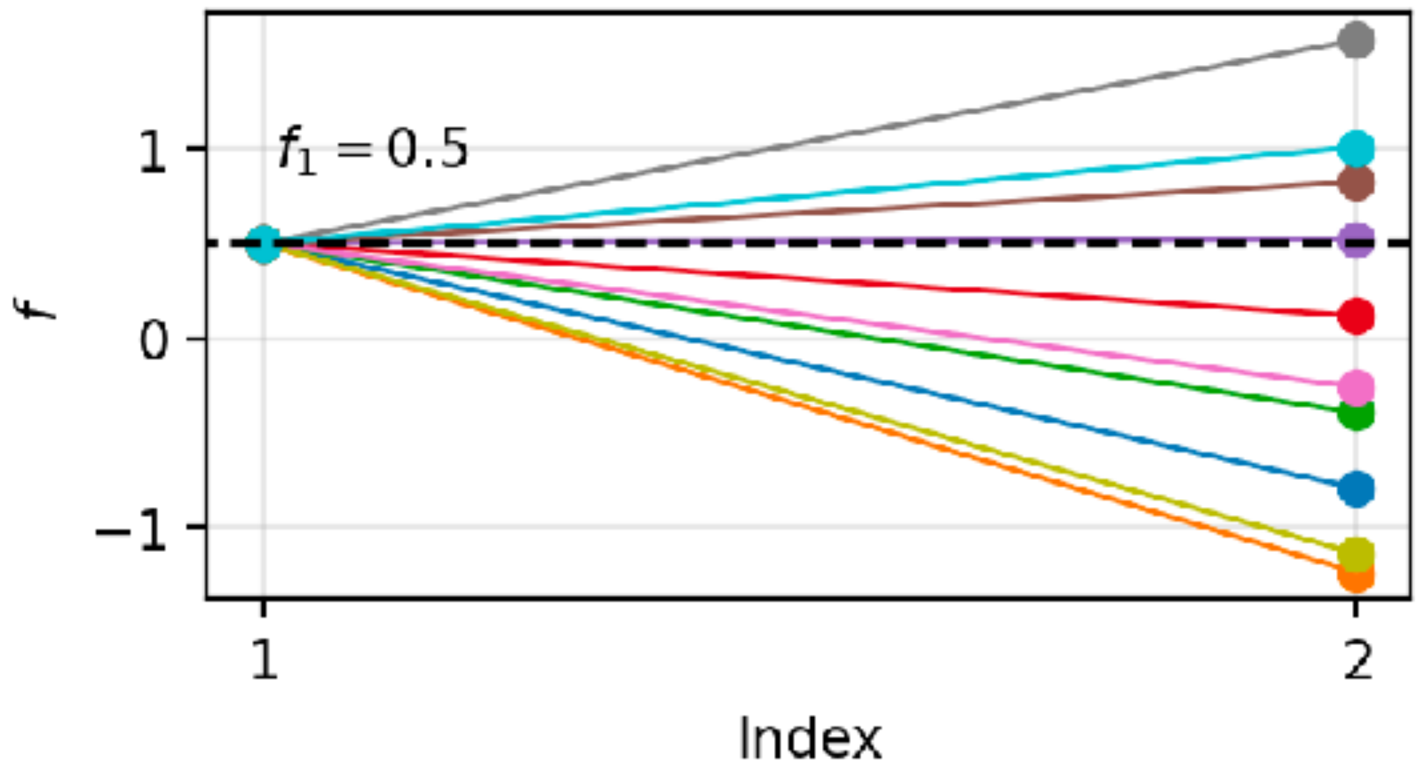


Gaussian Processes: A Primer

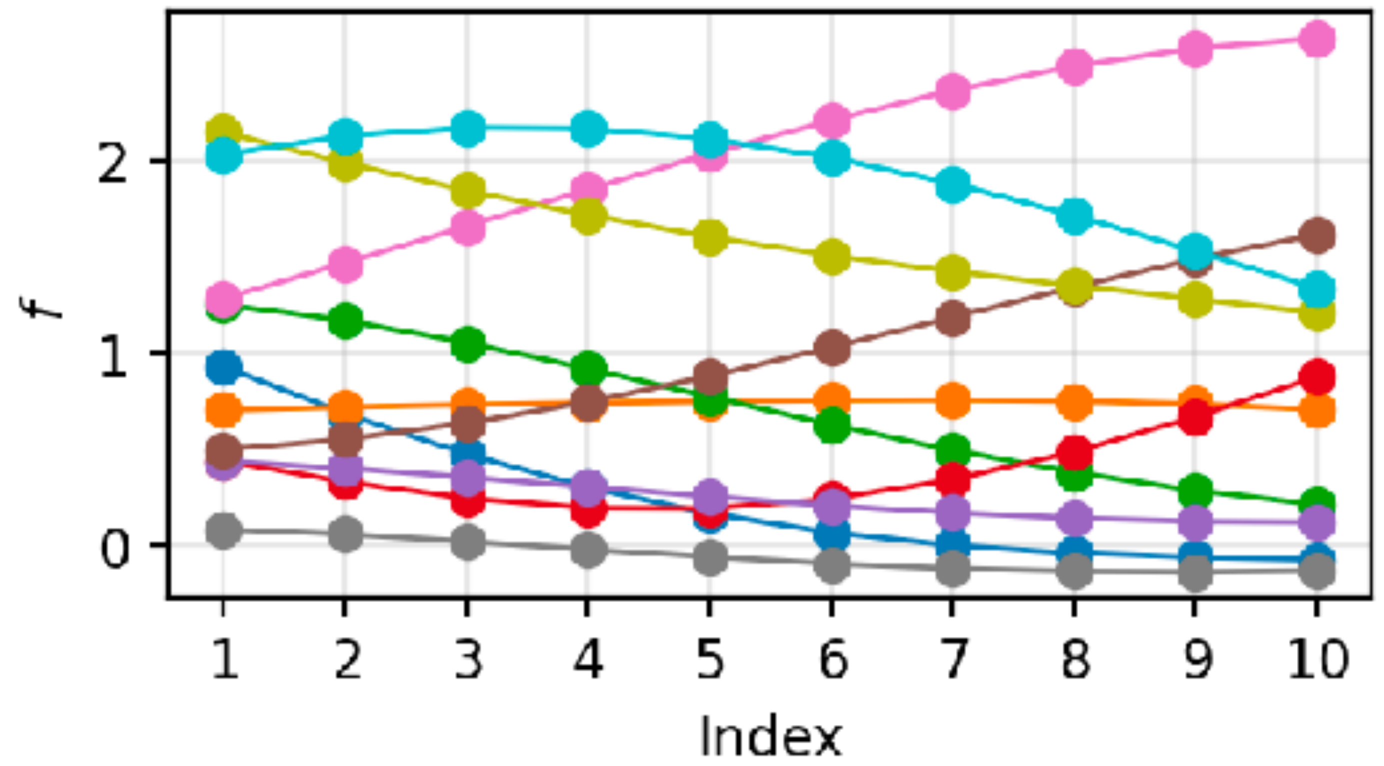
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$



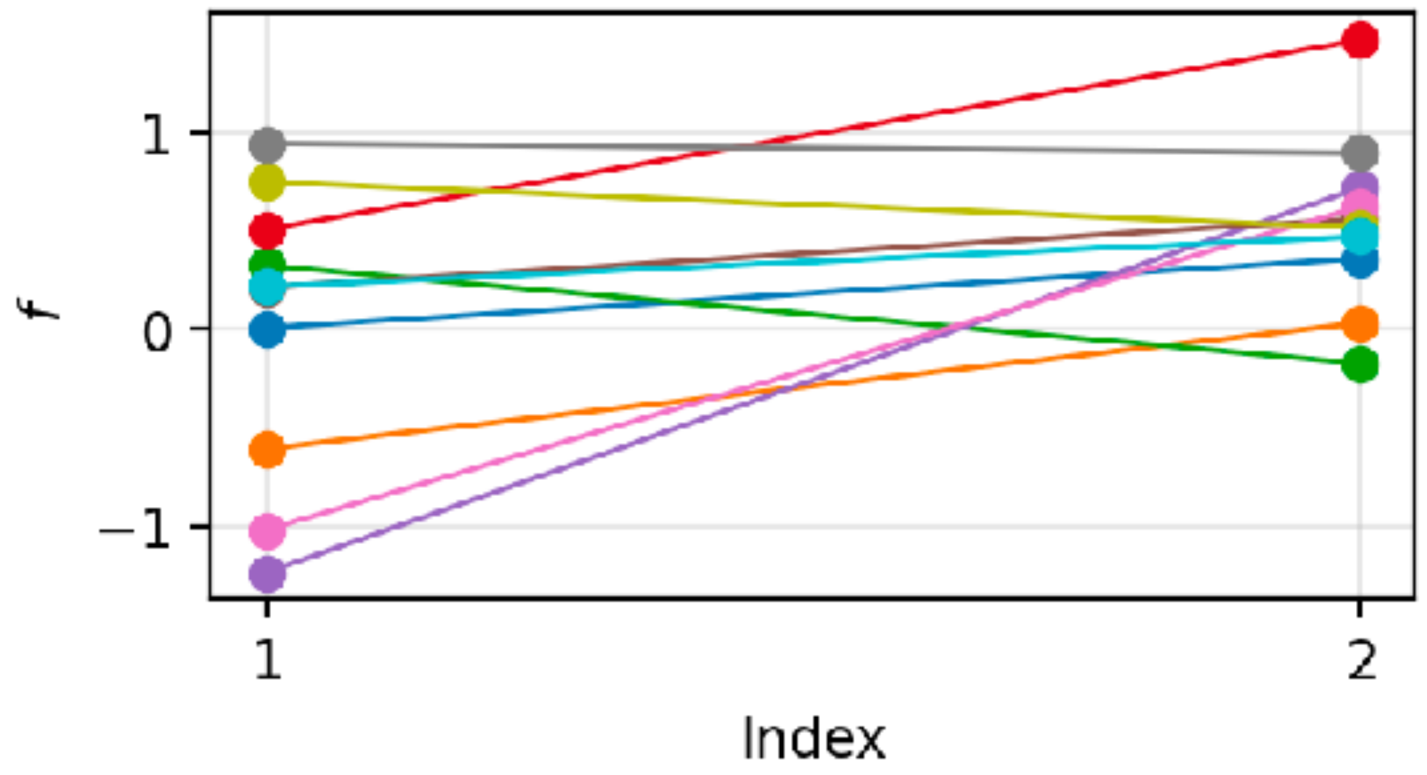
$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{10} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_{10} \end{pmatrix}, \mathbf{K}\right), \qquad K_{ij} = \exp(-(i-j)^2)$$



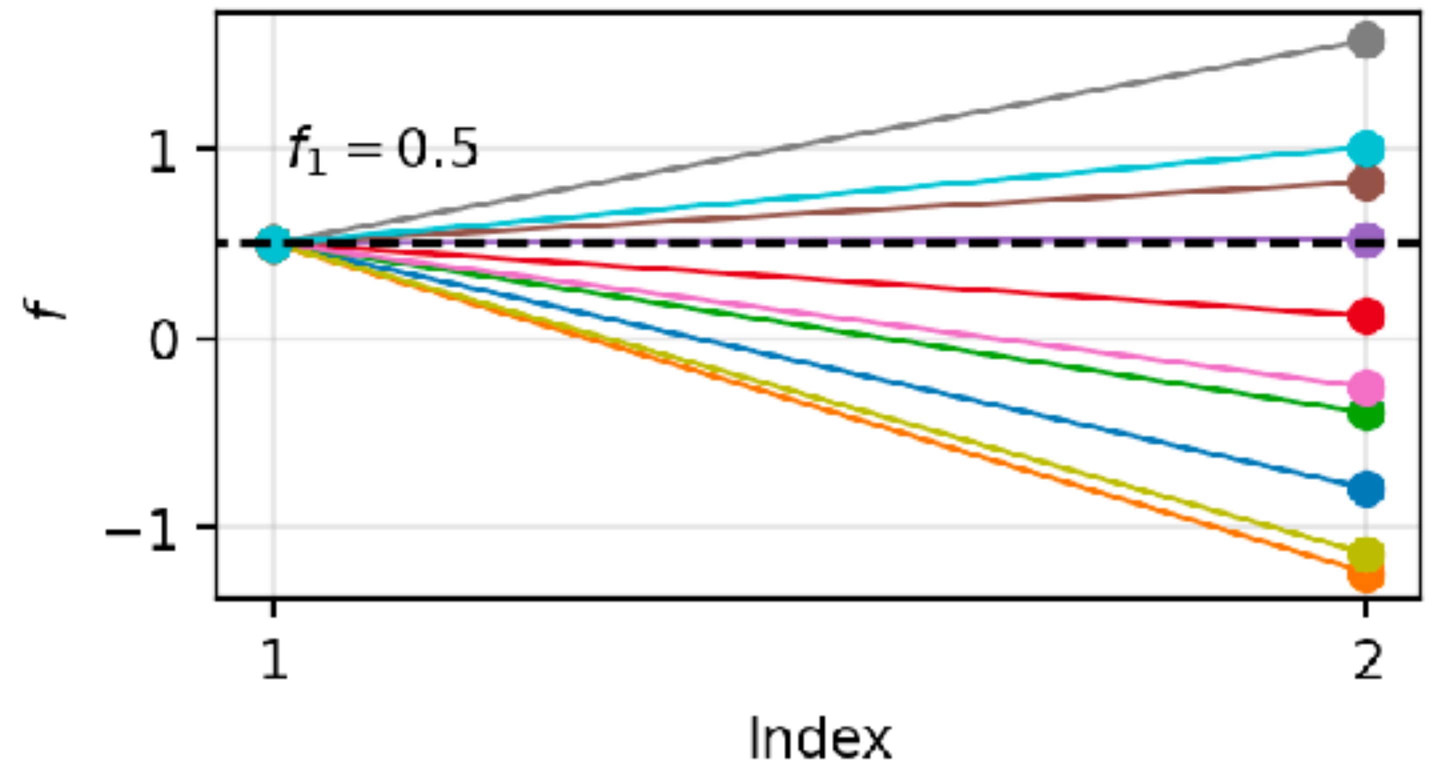
$$f \mid f_6 = 0.5 \sim \mathcal{N}\left(K_{*6}K_{66}^{-1}y, K_{**} - K_{*6}K_{66}^{-1}K_{*6}^T\right)$$

Gaussian Processes: A Primer

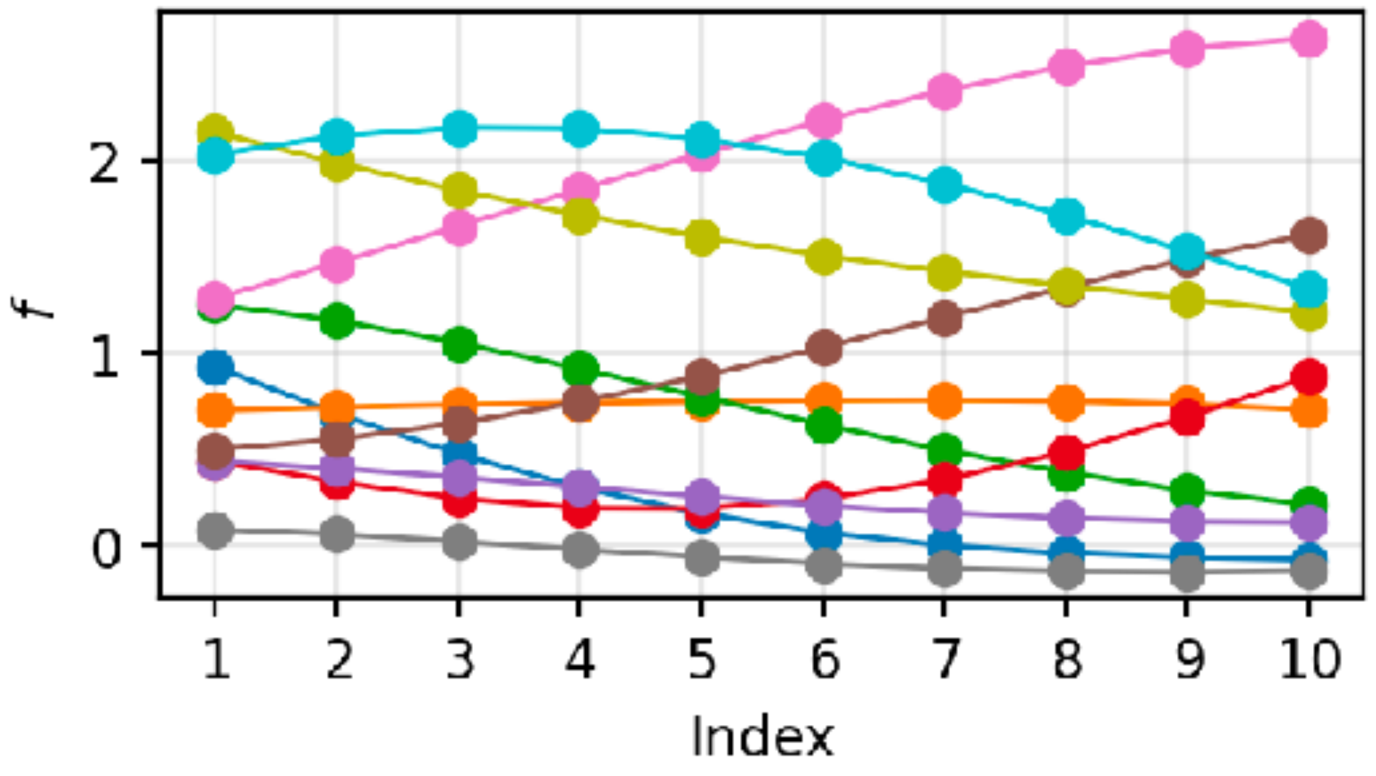
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



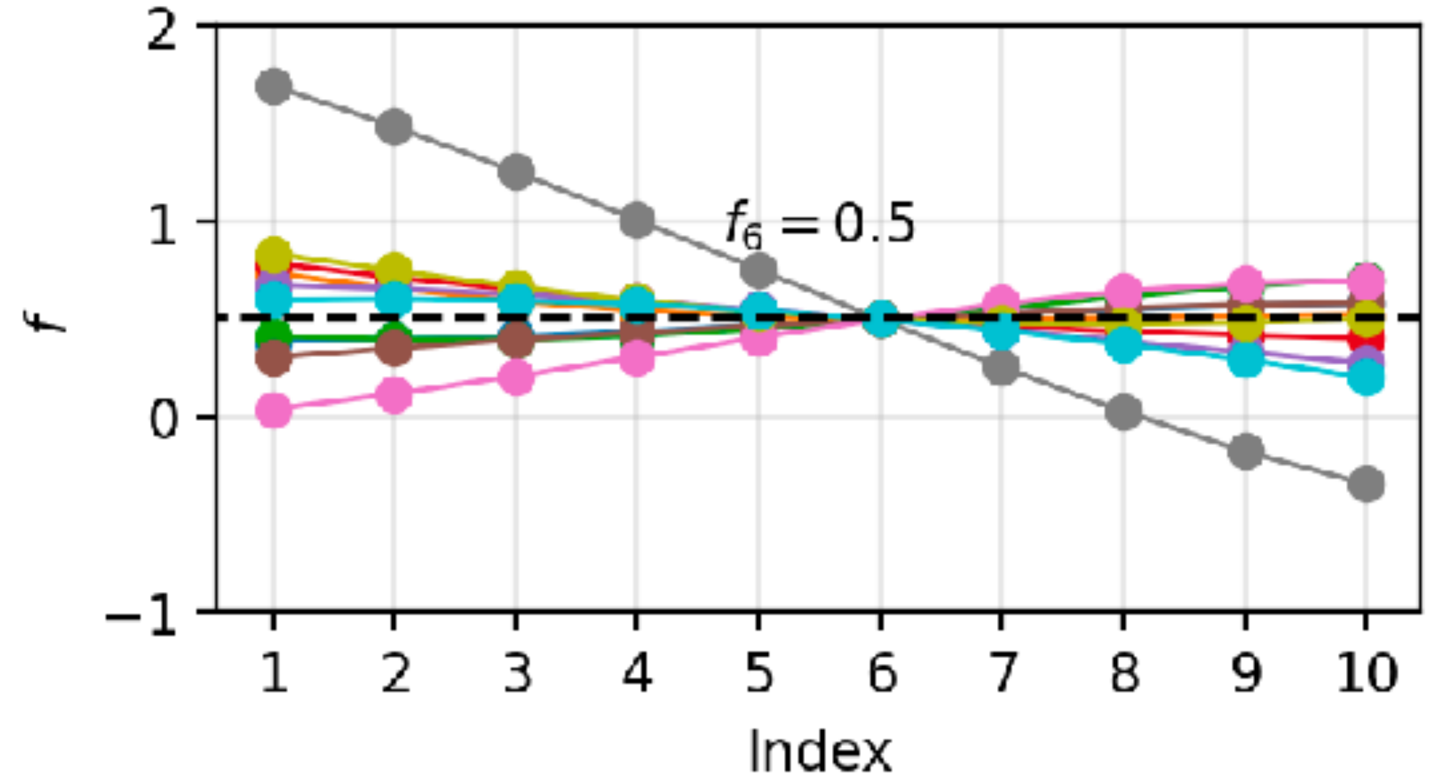
$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$



$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{10} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_{10} \end{pmatrix}, \mathbf{K}\right), \qquad K_{ij} = \exp(-(i-j)^2)$$

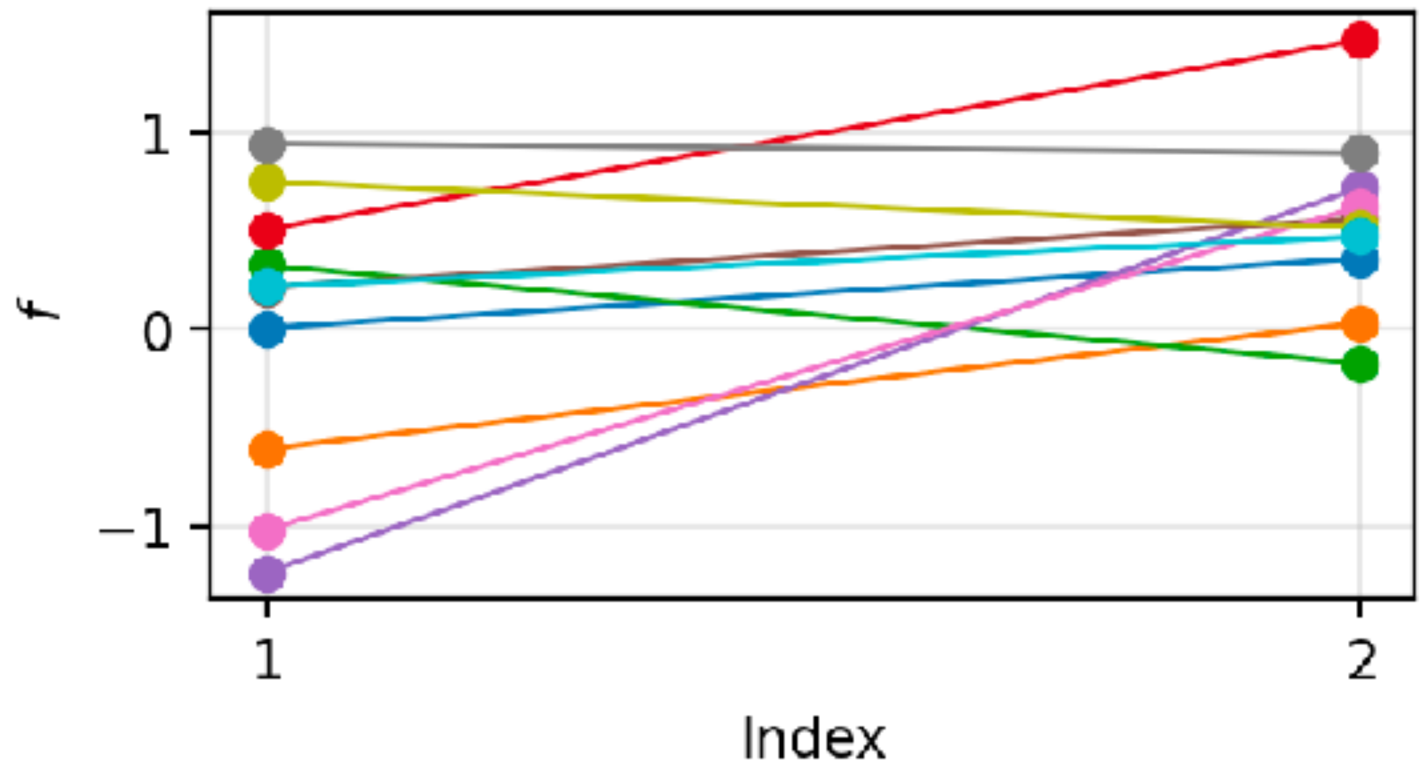


$$f \mid f_6 = 0.5 \sim \mathcal{N}\left(K_{*6}K_{66}^{-1}y, K_{**} - K_{*6}K_{66}^{-1}K_{*6}^{T}\right)$$

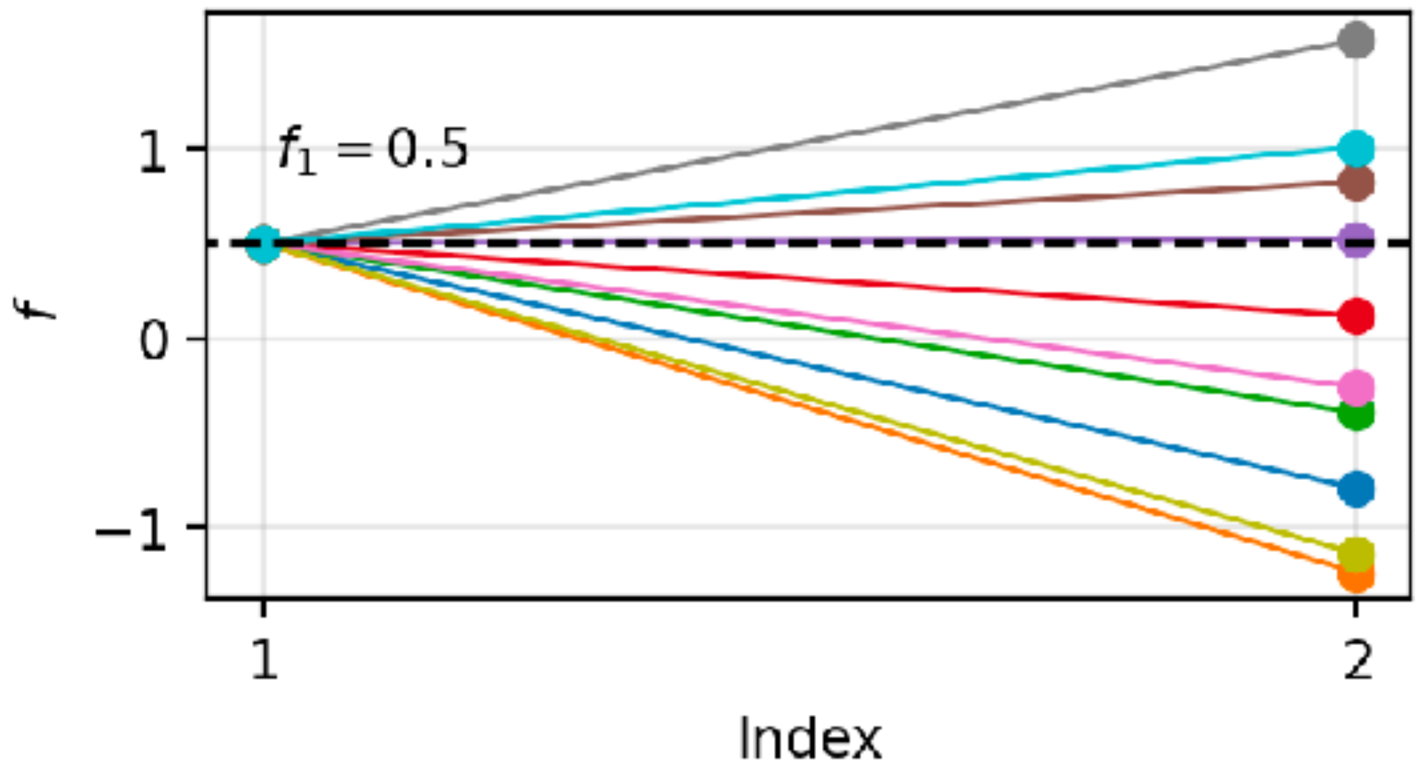


Gaussian Processes: A Primer

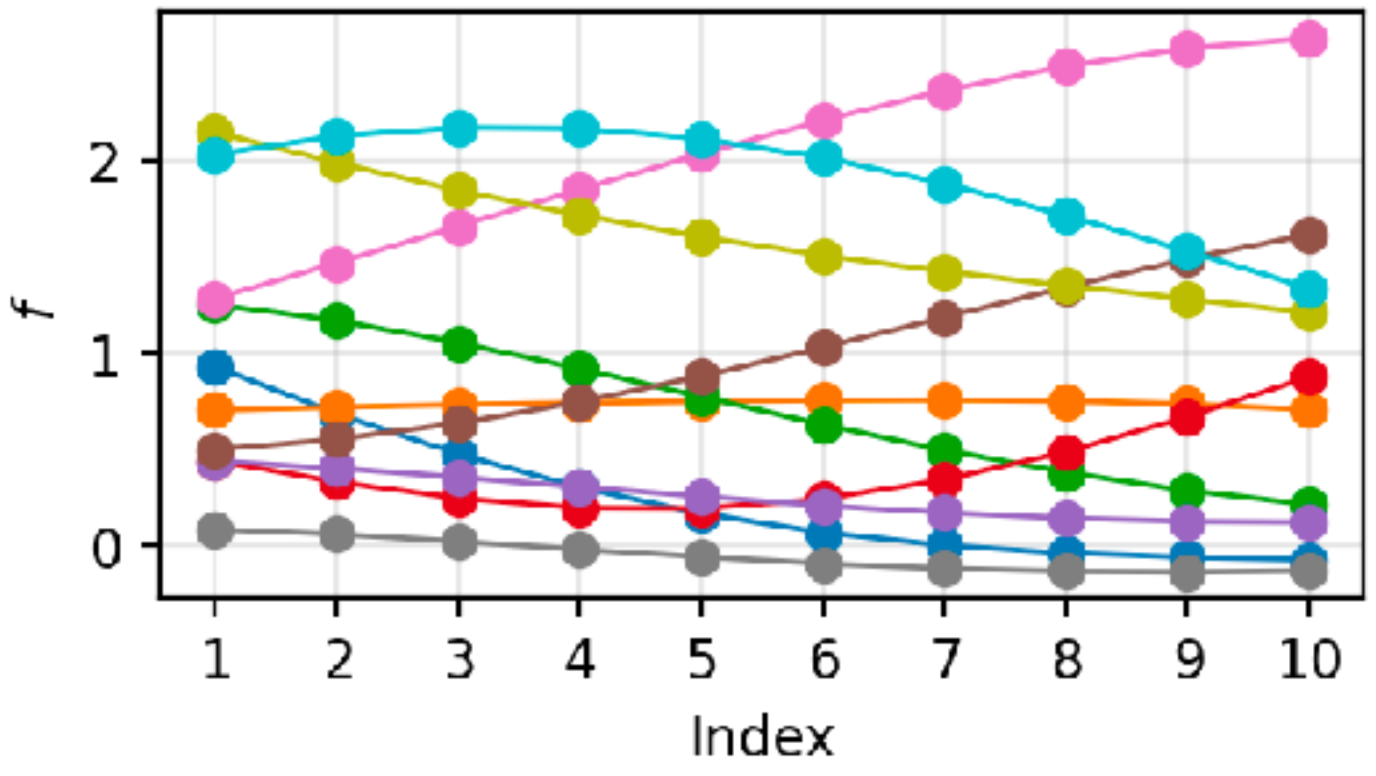
$$\begin{pmatrix} f_1 \\ f_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}, \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{21} \end{pmatrix}\right)$$



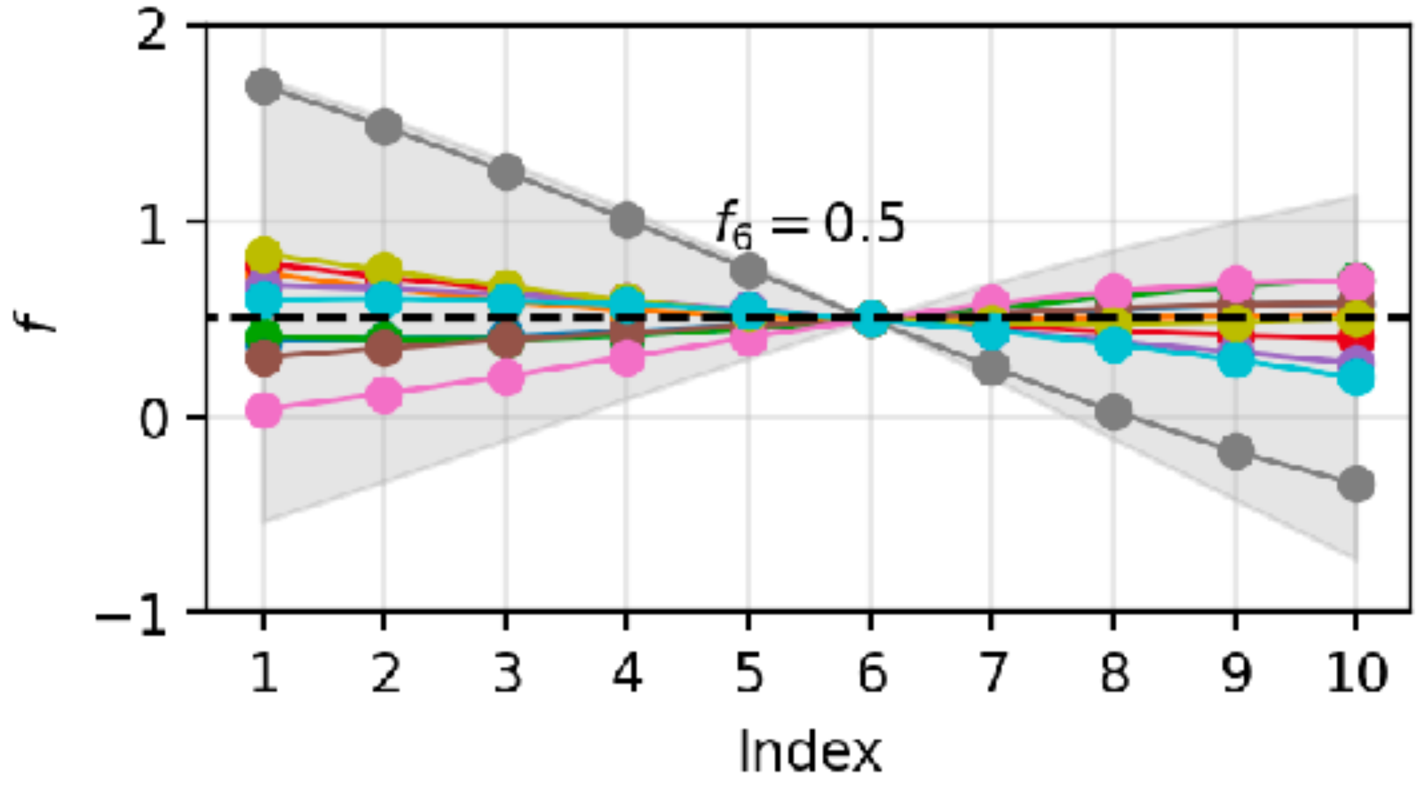
$$f_2 \mid f_1 = y \sim \mathcal{N}\left(k_{21}k_{11}^{-1}y, k_{22} - k_{21}k_{11}^{-1}k_{12}\right)$$



$$\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{10} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m_1 \\ m_2 \\ \vdots \\ m_{10} \end{pmatrix}, \mathbf{K}\right), \qquad K_{ij} = \exp(-(i-j)^2)$$

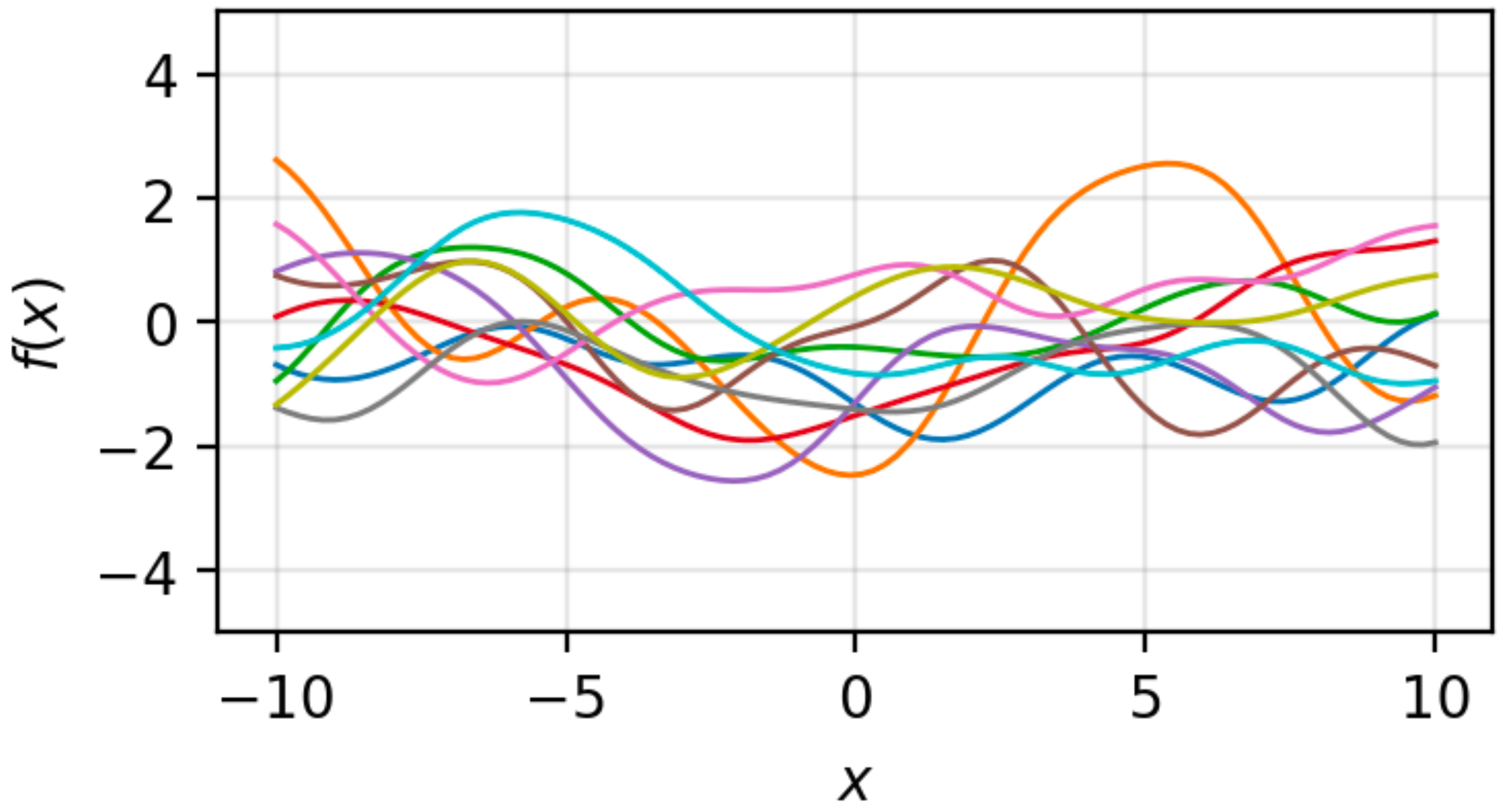


$$f \mid f_6 = 0.5 \sim \mathcal{N}\left(K_{*6}K_{66}^{-1}y, K_{**} - K_{*6}K_{66}^{-1}K_{*6}^T\right)$$



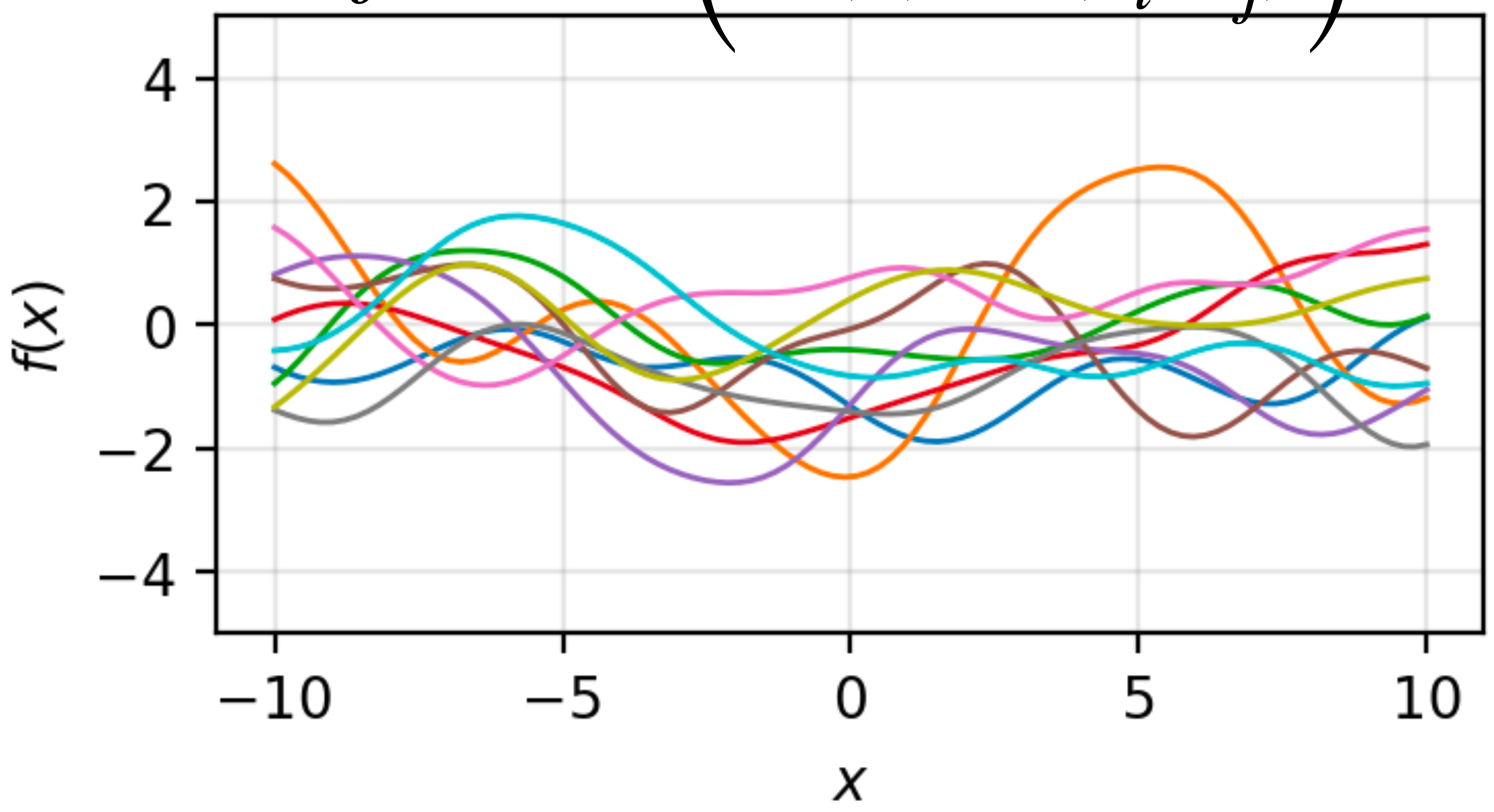
Gaussian Processes: A Primer

Gaussian Processes: A Primer



Gaussian Processes: A Primer

$$f \sim \mathrm{GP}\left(m(x), K(x_i, x_j)\right)$$

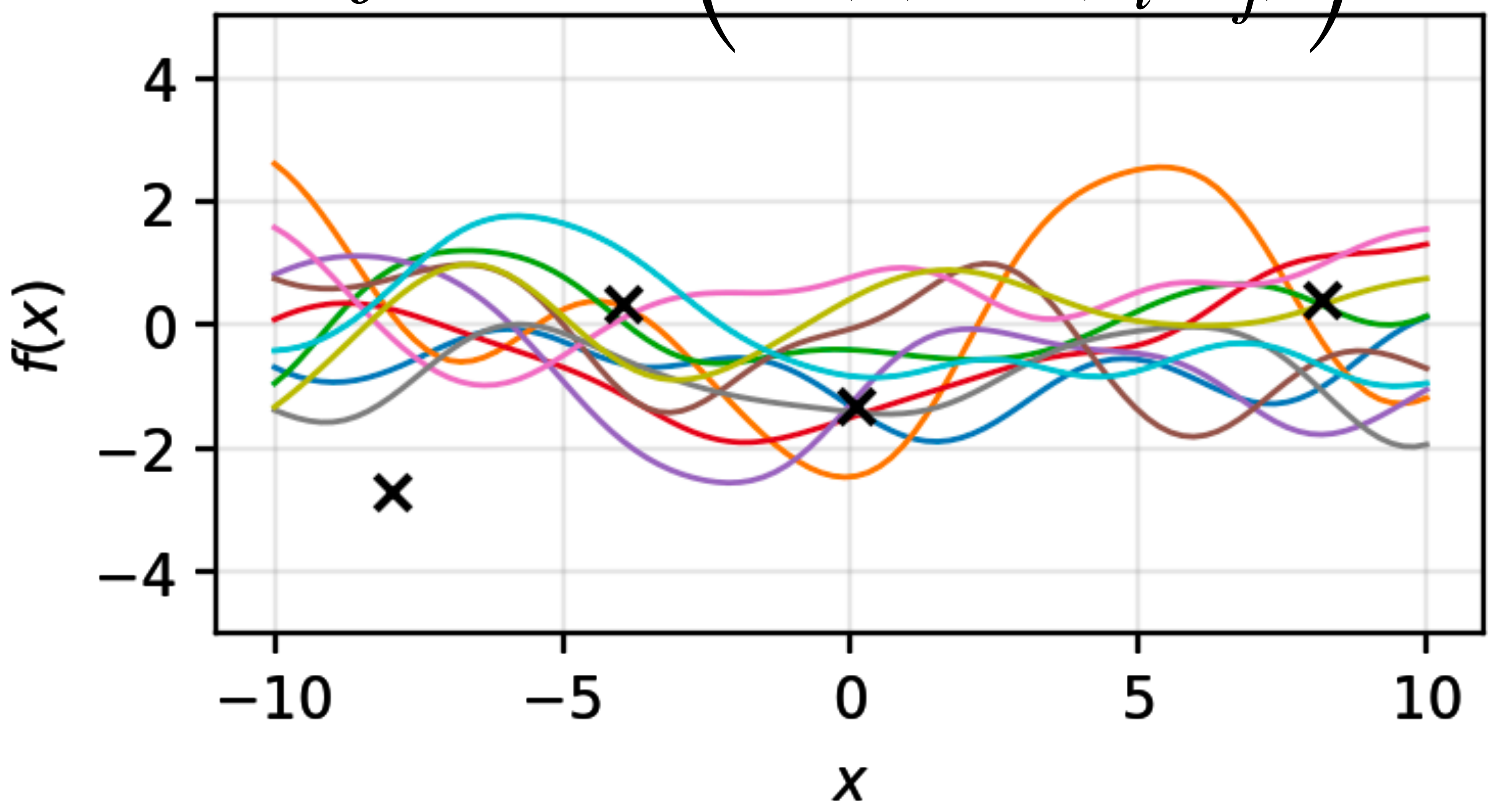


Gaussian Processes: A Primer

Dataset

$$(X, y) = \left\{(X_1, y_1), \ldots, (X_n, y_n)\right\}$$

$$f \sim \mathrm{GP}\left(m(x), K(x_i, x_j)\right)$$

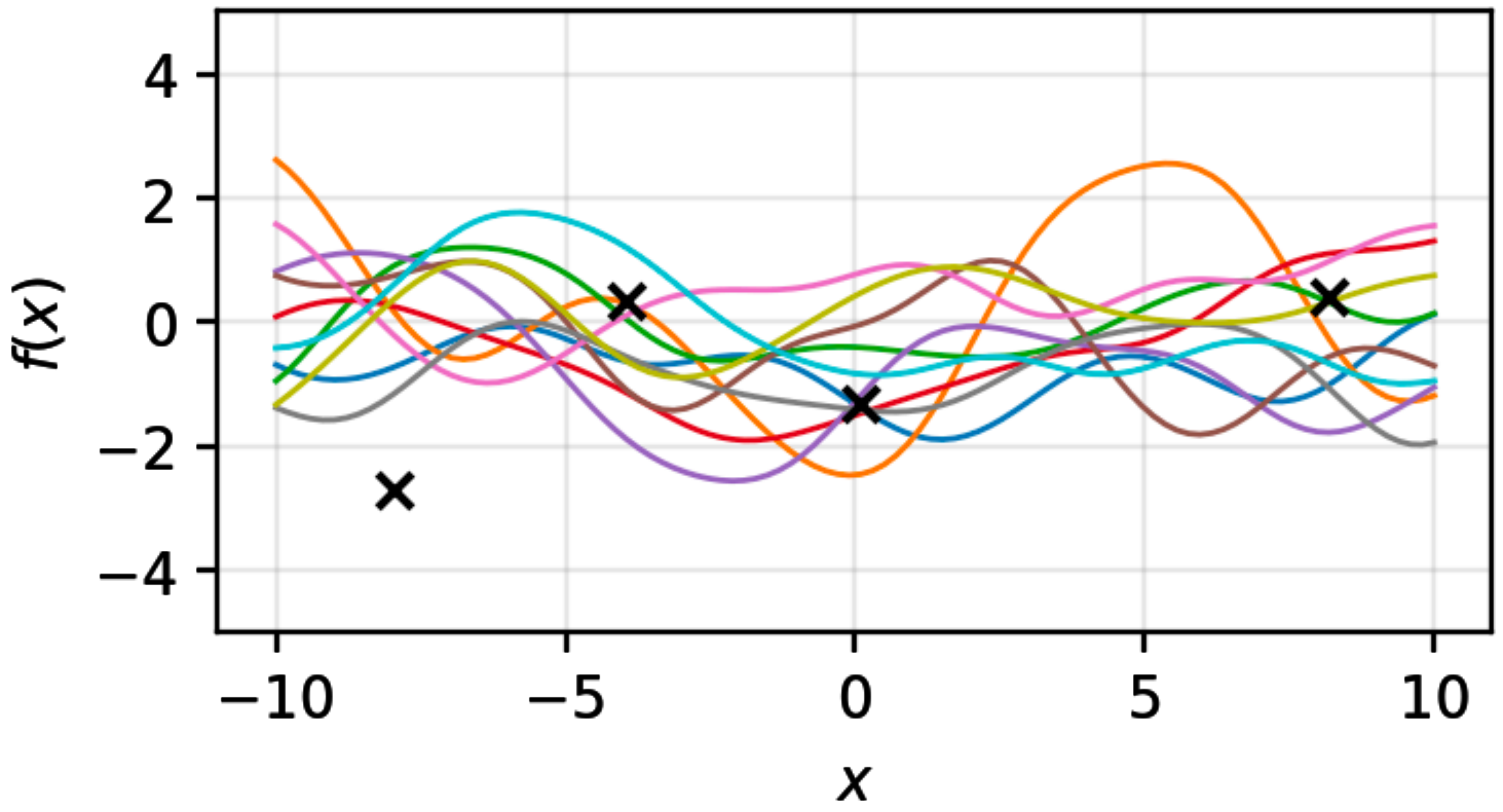


Gaussian Processes: A Primer

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \left\{(X_1, y_1), \ldots, (X_n, y_n)\right\}$$



Gaussian Processes: A Primer

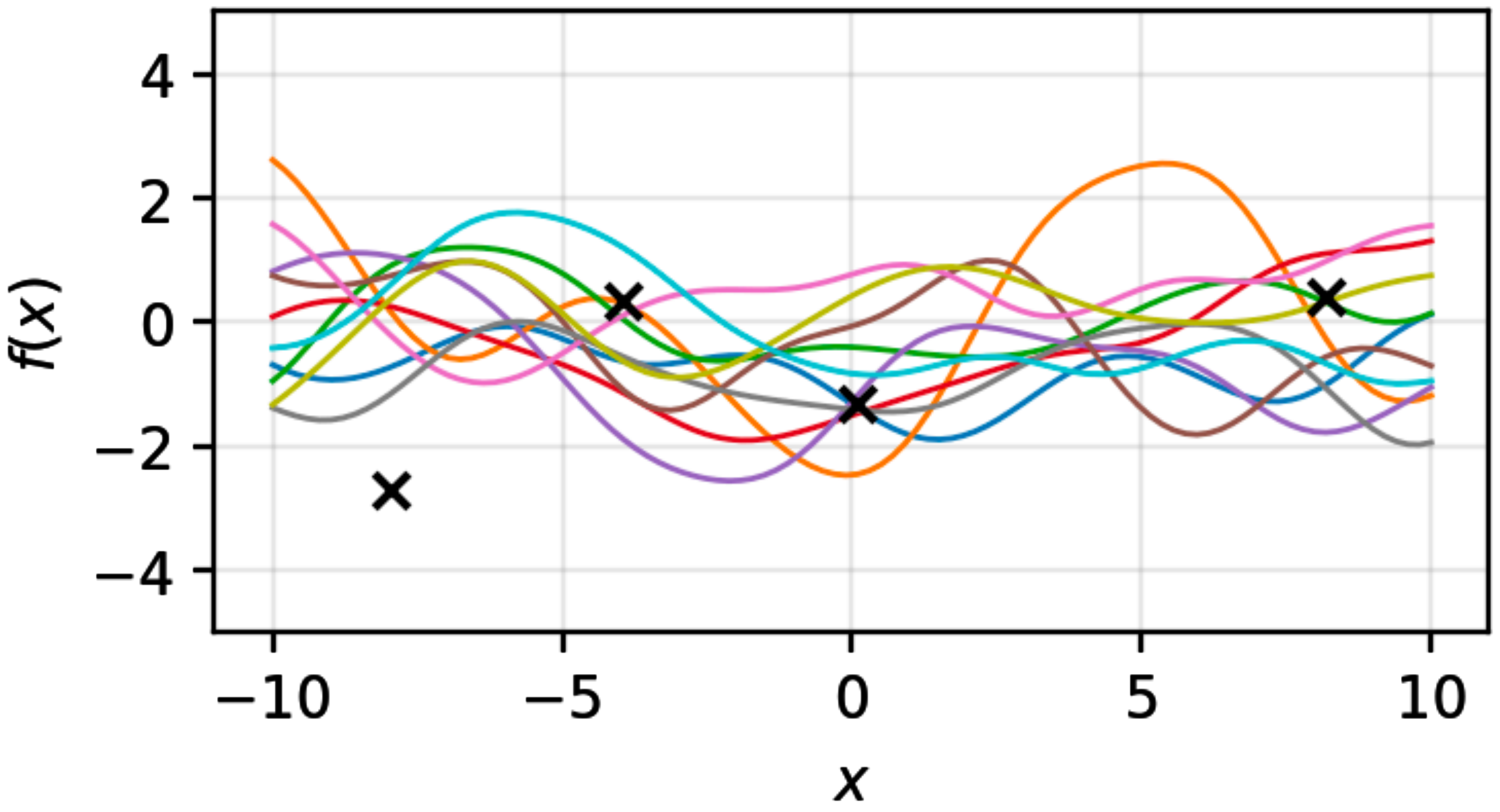
$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \{(X_1, y_1), \ldots, (X_n, y_n)\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$



Gaussian Processes: A Primer

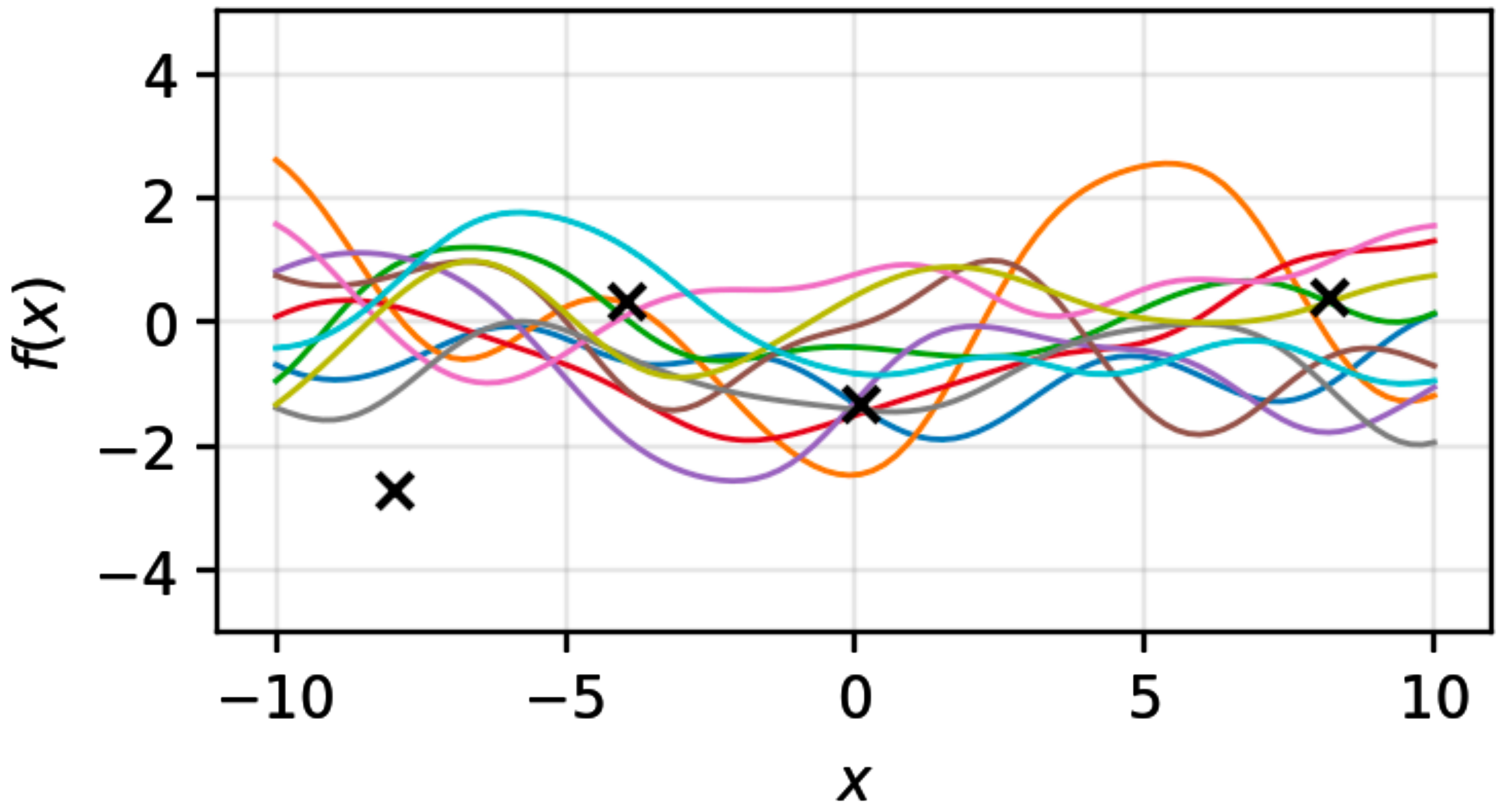
$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \left\{(X_1, y_1), \ldots, (X_n, y_n)\right\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$



Gaussian Processes: A Primer

$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Posterior Distribution

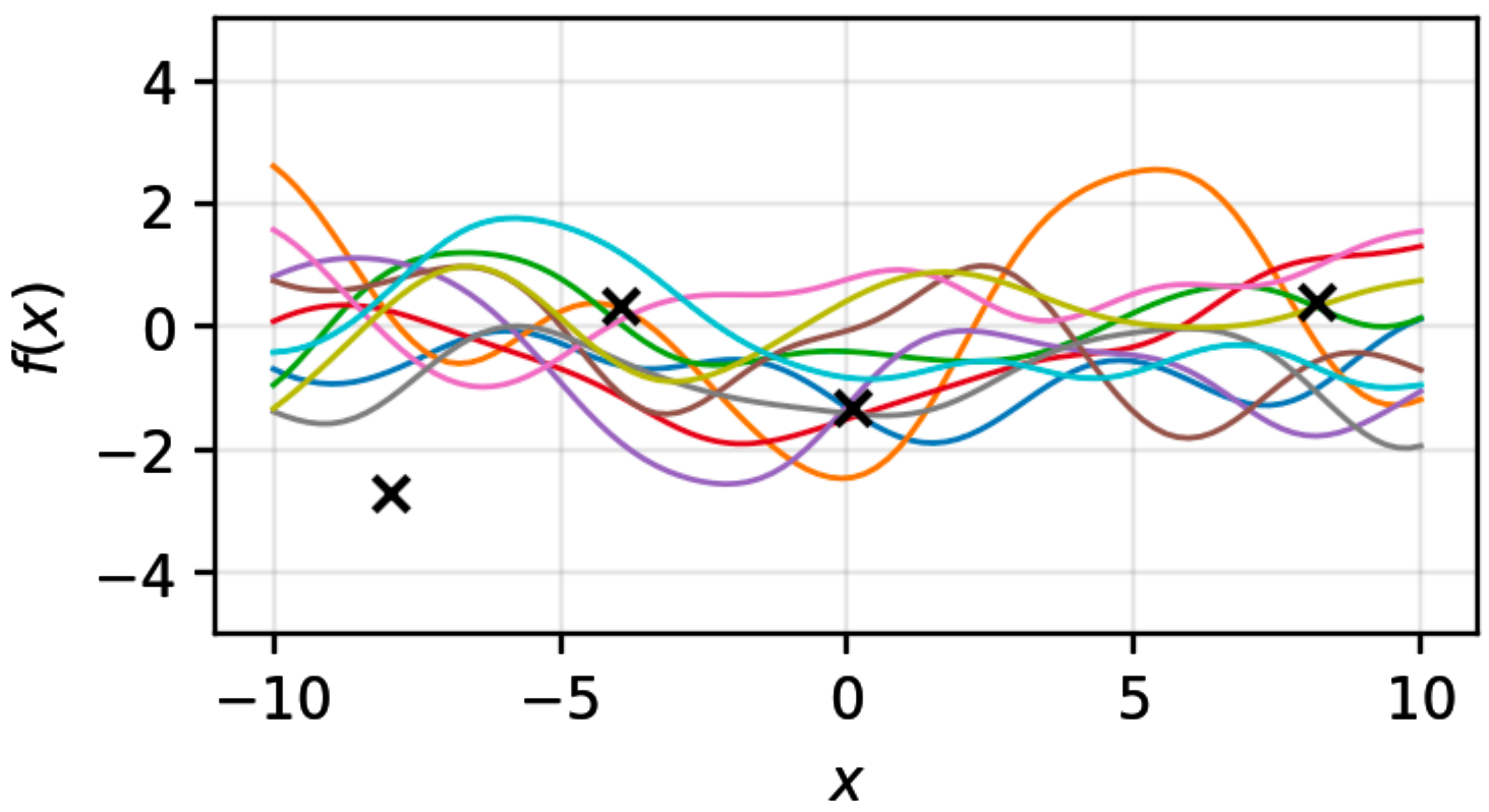
$$p(f_* \mid f, X, y) = \mathcal{N}(\mu_{f|y}, \Sigma_{f|y})$$

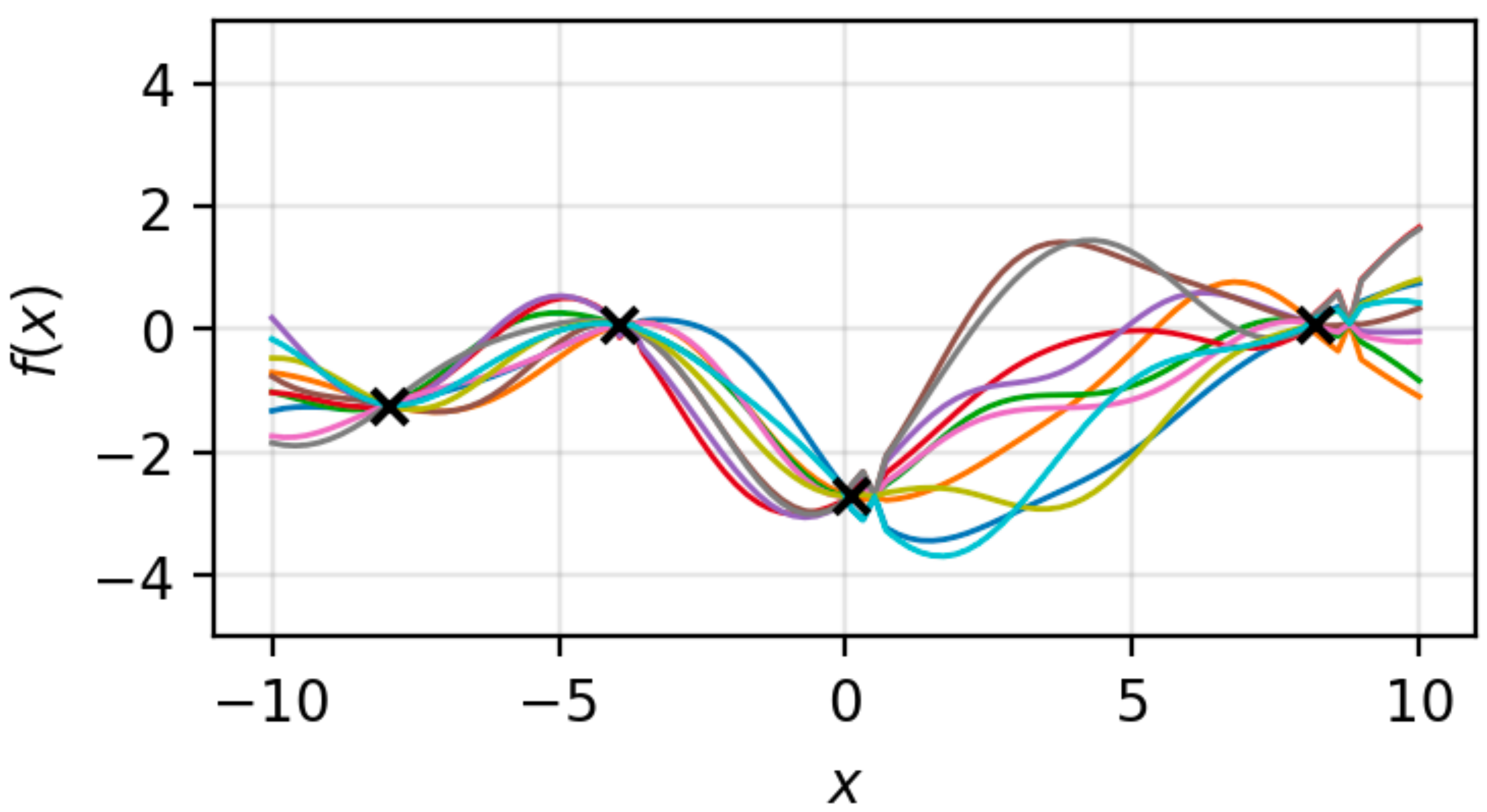
Dataset

$$(X, y) = \left\{(X_1, y_1), \ldots, (X_n, y_n)\right\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$





Gaussian Processes: A Primer

$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \left\{(X_1, y_1), \ldots, (X_n, y_n)\right\}$$

Observation noise assumption

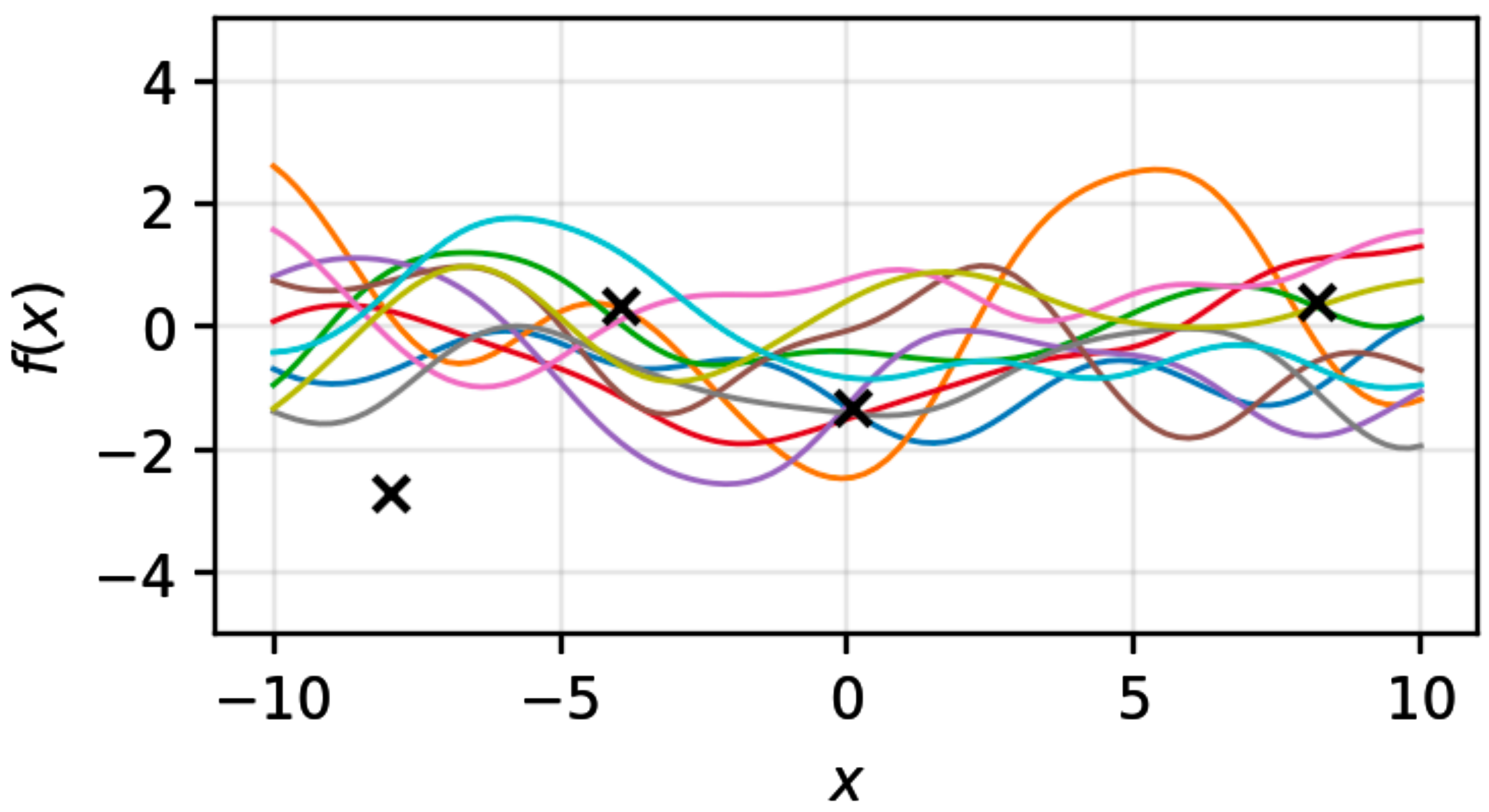
$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$

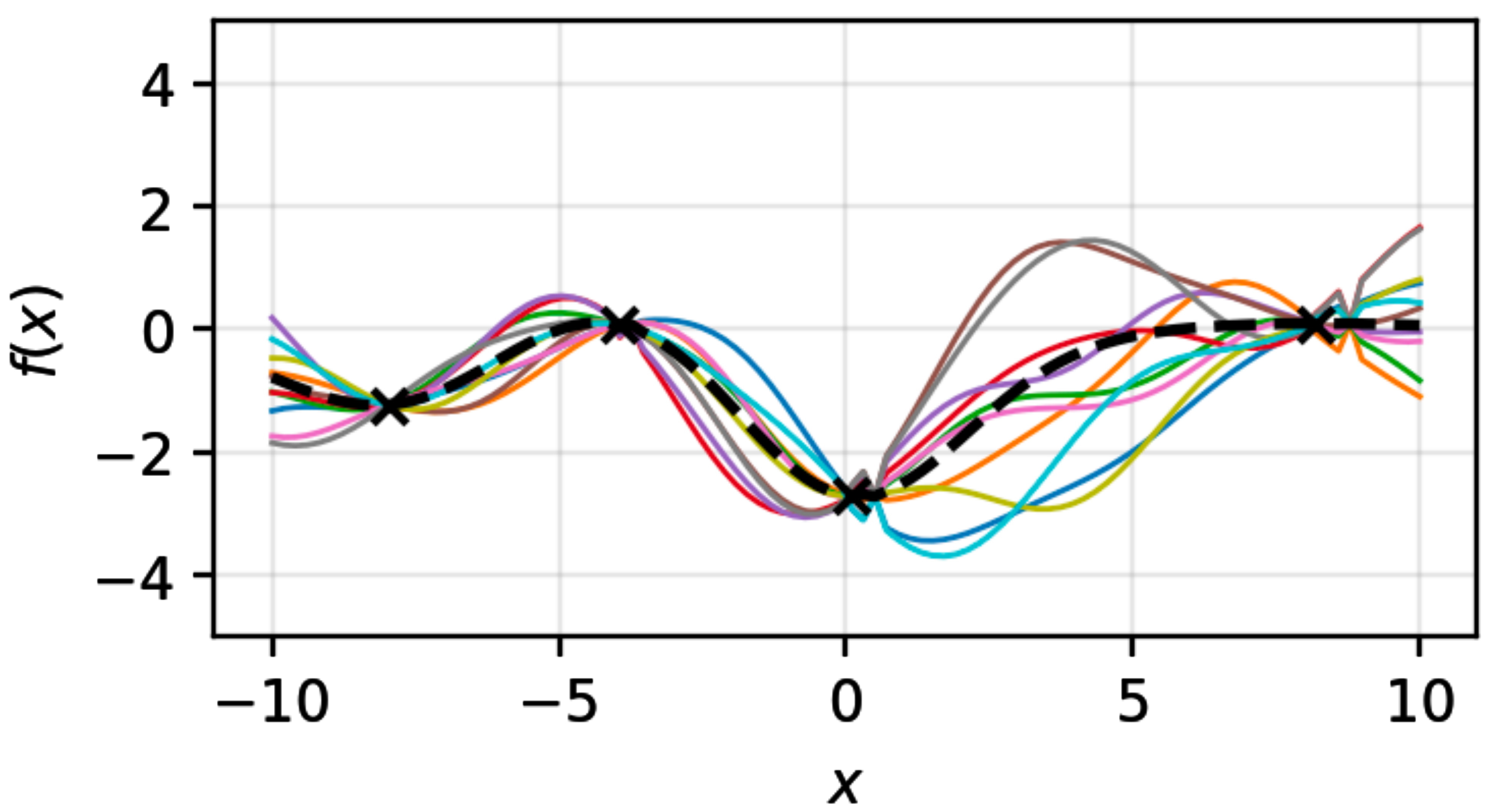
Posterior Distribution

$$p(f_* \,|\, f, X, y) = \mathcal{N}(\mu_{f|y}, \Sigma_{f|y})$$

Predictive Mean

$$\mu_{f|y} = K_{*n}(K_{nn} + \sigma^2 I)^{-1} y$$





Gaussian Processes: A Primer

$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \{(X_1, y_1), \ldots, (X_n, y_n)\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$

Posterior Distribution

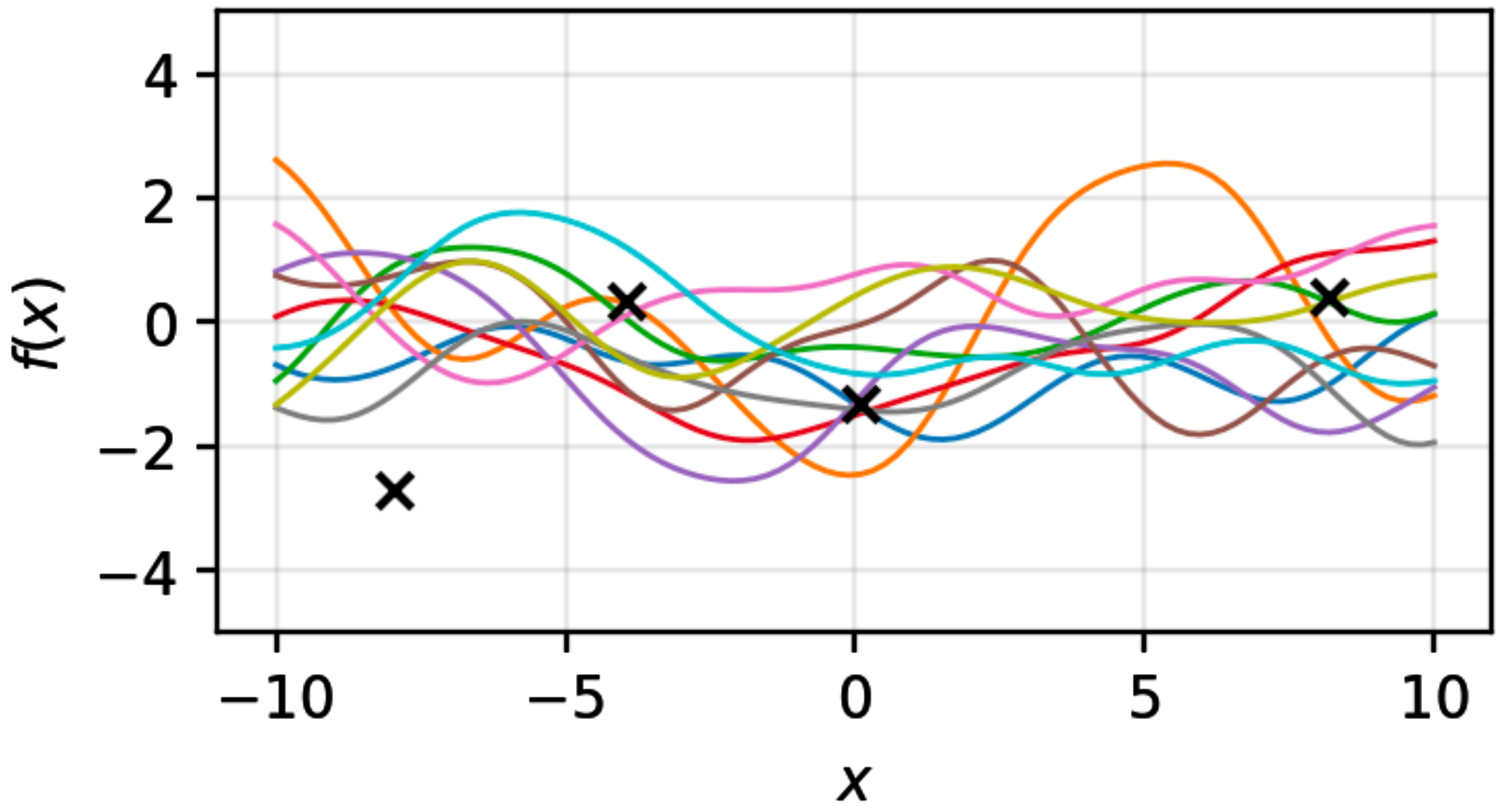
$$p(f_* \mid f, X, y) = \mathcal{N}(\mu_{f|y}, \Sigma_{f|y})$$

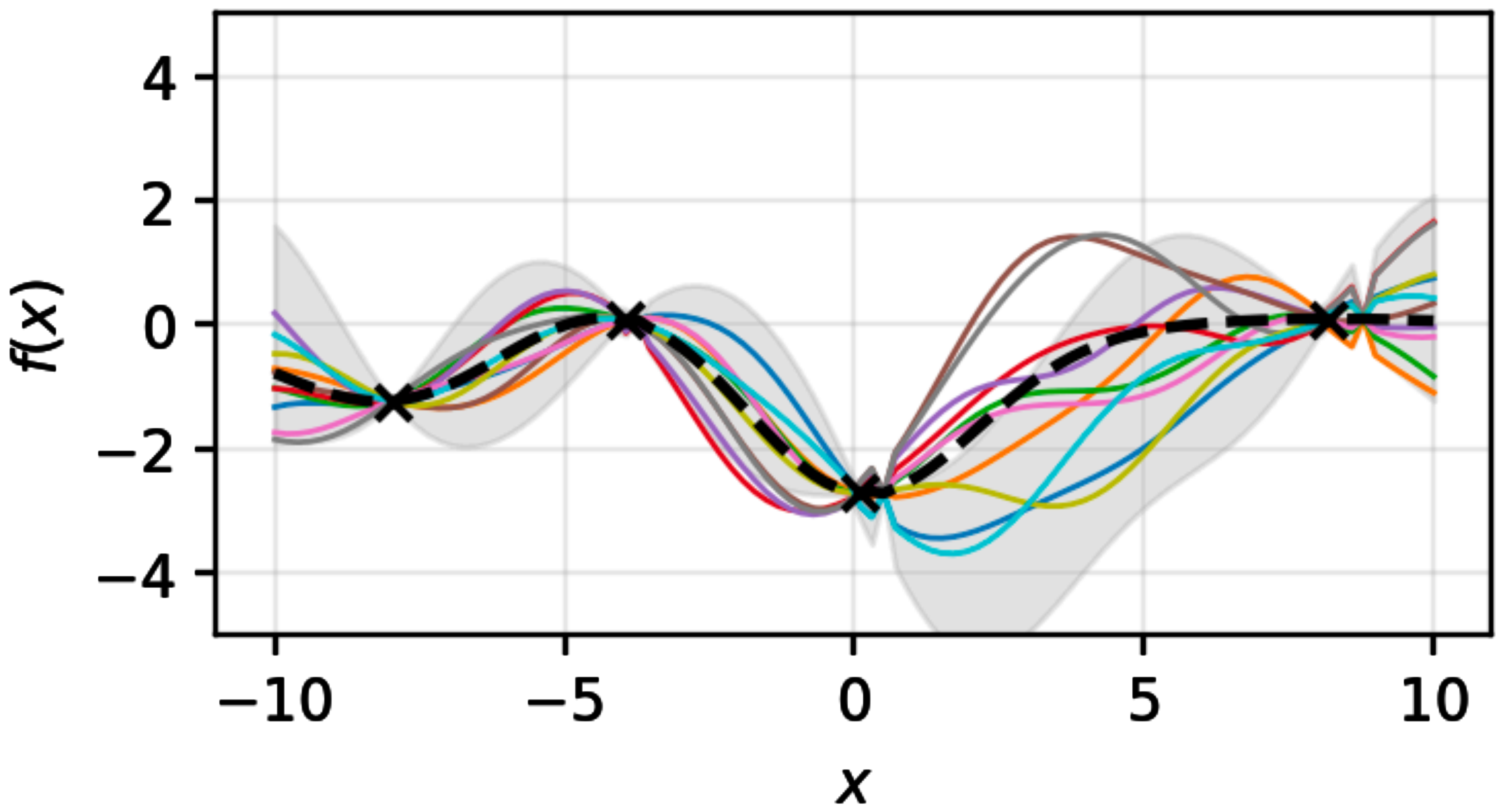
Predictive Mean

$$\mu_{f|y} = K_{*n}(K_{nn} + \sigma^2 I)^{-1} y$$

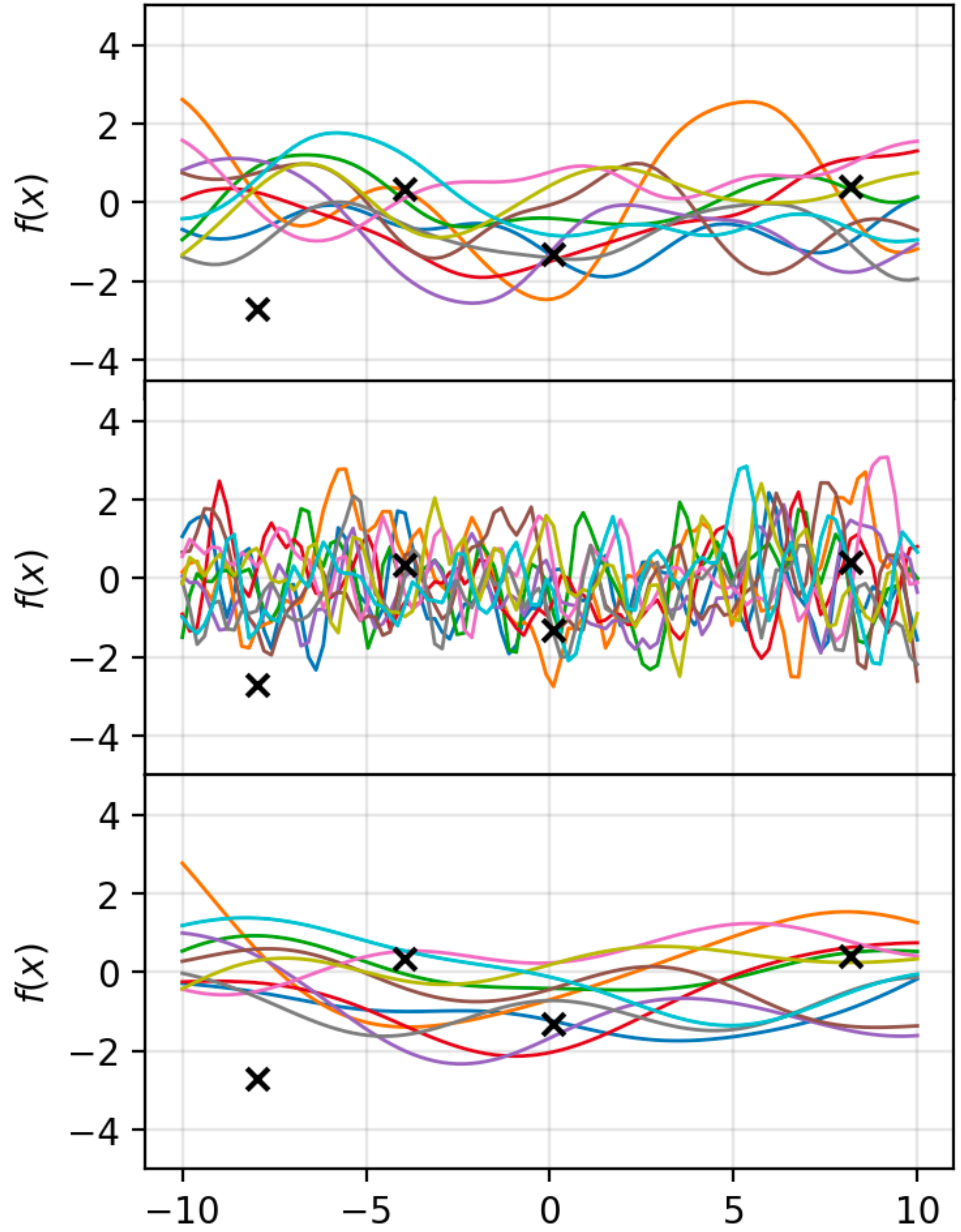
Uncertainty Estimate

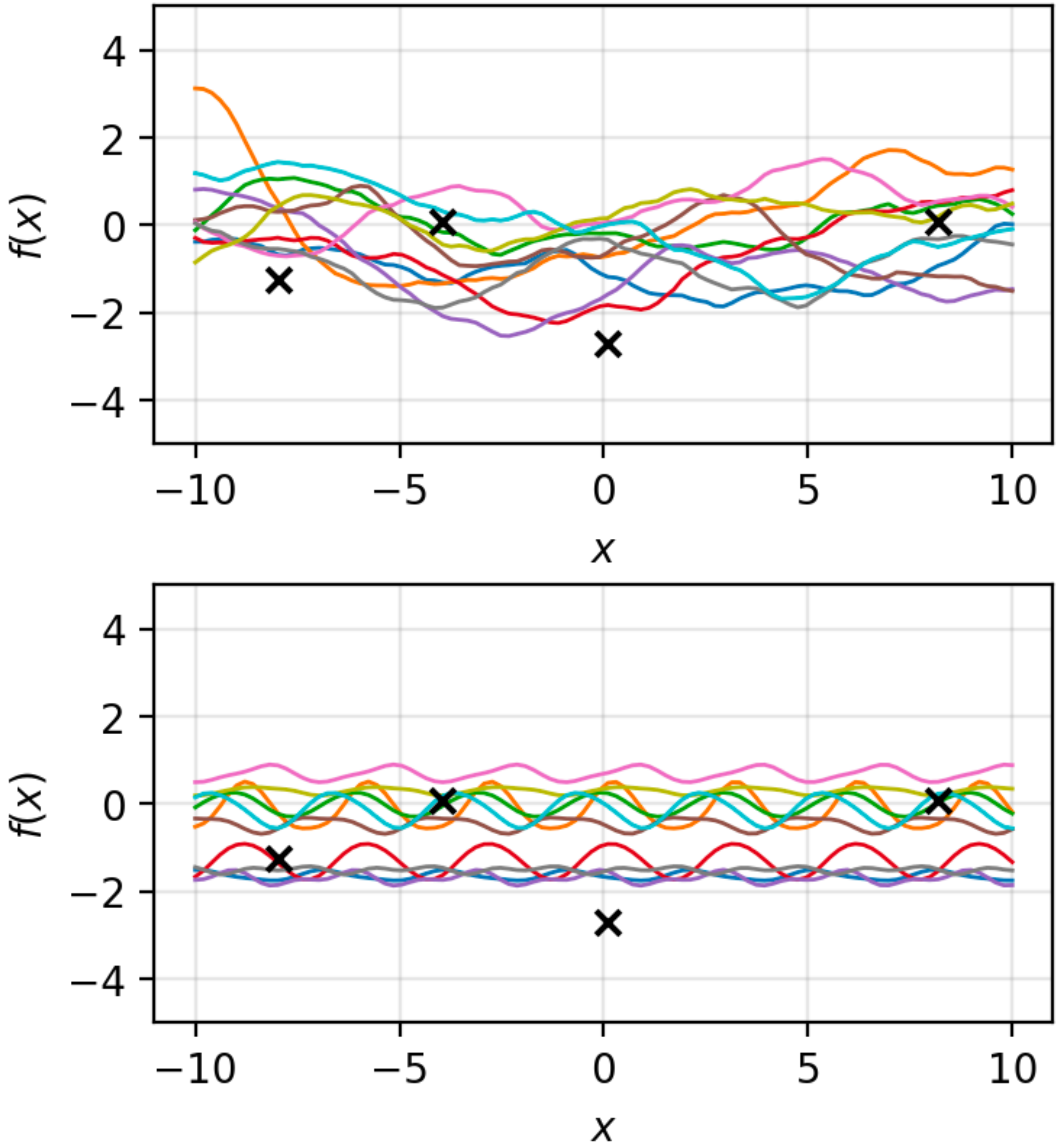
$$\Sigma_{f|y} = K_{**} - K_{*n}^\top (K_{nn} + \sigma^2 I)^{-1} K_{n*}$$





Different Kernels and Hparams





Different Modalities

Different Modalities

$k($  ,  $)$ $>$ $k($  ,  $)$

NNGP/NTK kernels [1]

Linearised Laplace [2]

Deep convolutional kernels

SNGP [3]

[1] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

[2] **Padhy, S.***, Antorán, J.,*, **Barbano, R.,** Nalisnick, E., ... and Hernández-Lobato, J.M.. Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***

[3] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***

[4] **Padhy, S.***, Lin, J. A.*, Antorán, J.*, Tripp, A., Terenin, A., Szepesvári, C., ... & Janz, D. Stochastic Gradient Descent for Gaussian Processes Done Right. ***ICLR 2024***

Different Modalities

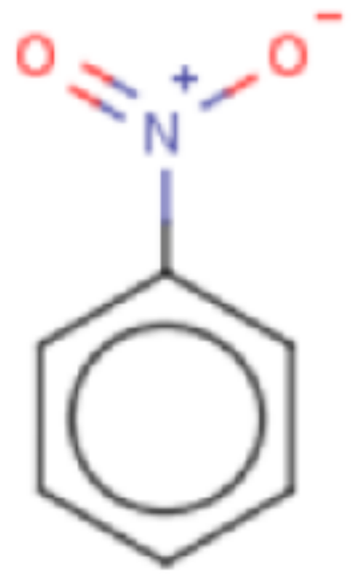
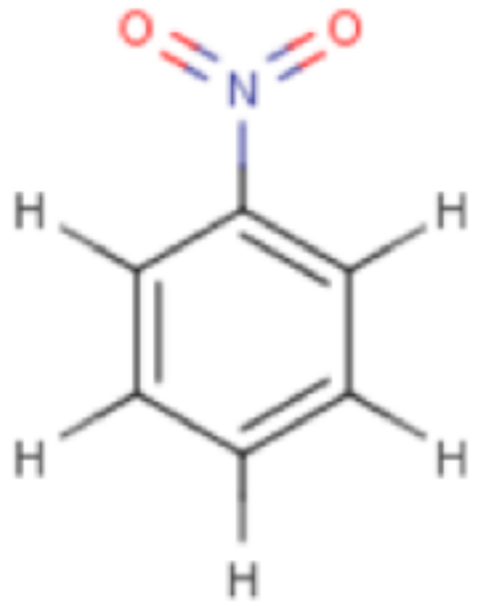
$k($  ,  $)$ > $k($  ,  $)$

NNGP/NTK kernels [1]

Linearised Laplace [2]

Deep convolutional kernels

SNGP [3]

$k($  ,  $)$

Tanimoto kernels [4]

Graph NNGP kernels [1]

[1] Adlam, B., Lee, J., **Padhy, S.,** Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

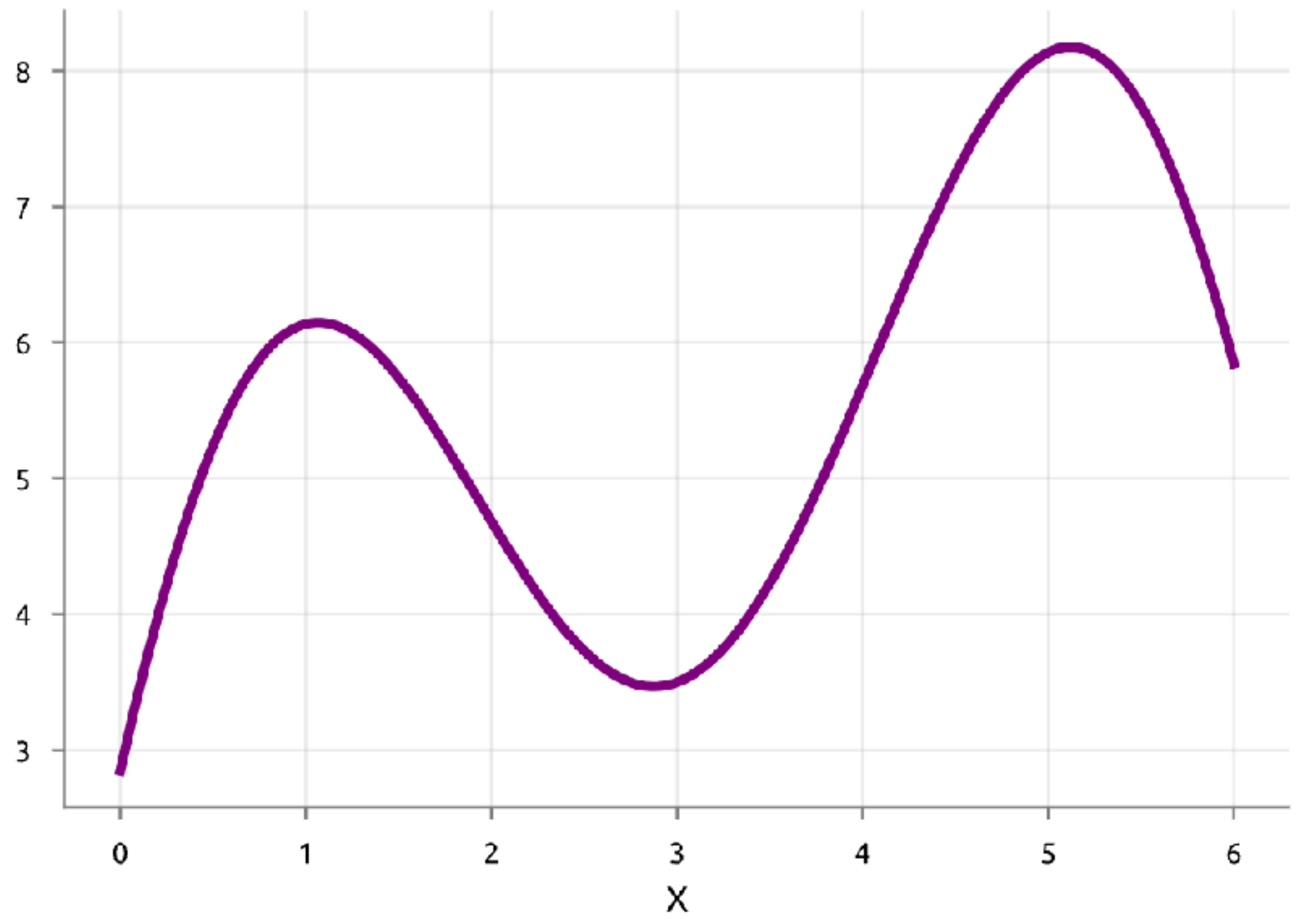
[2] **Padhy, S.***, Antorán, J.,*, **Barbano, R.,** Nalisnick, E., ... and Hernández-Lobato, J.M.. Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***

[3] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***

[4] **Padhy, S.***, Lin, J. A.*, Antorán, J.*, Tripp, A., Terenin, A., Szepesvári, C., ... & Janz, D. Stochastic Gradient Descent for Gaussian Processes Done Right. ***ICLR 2024***

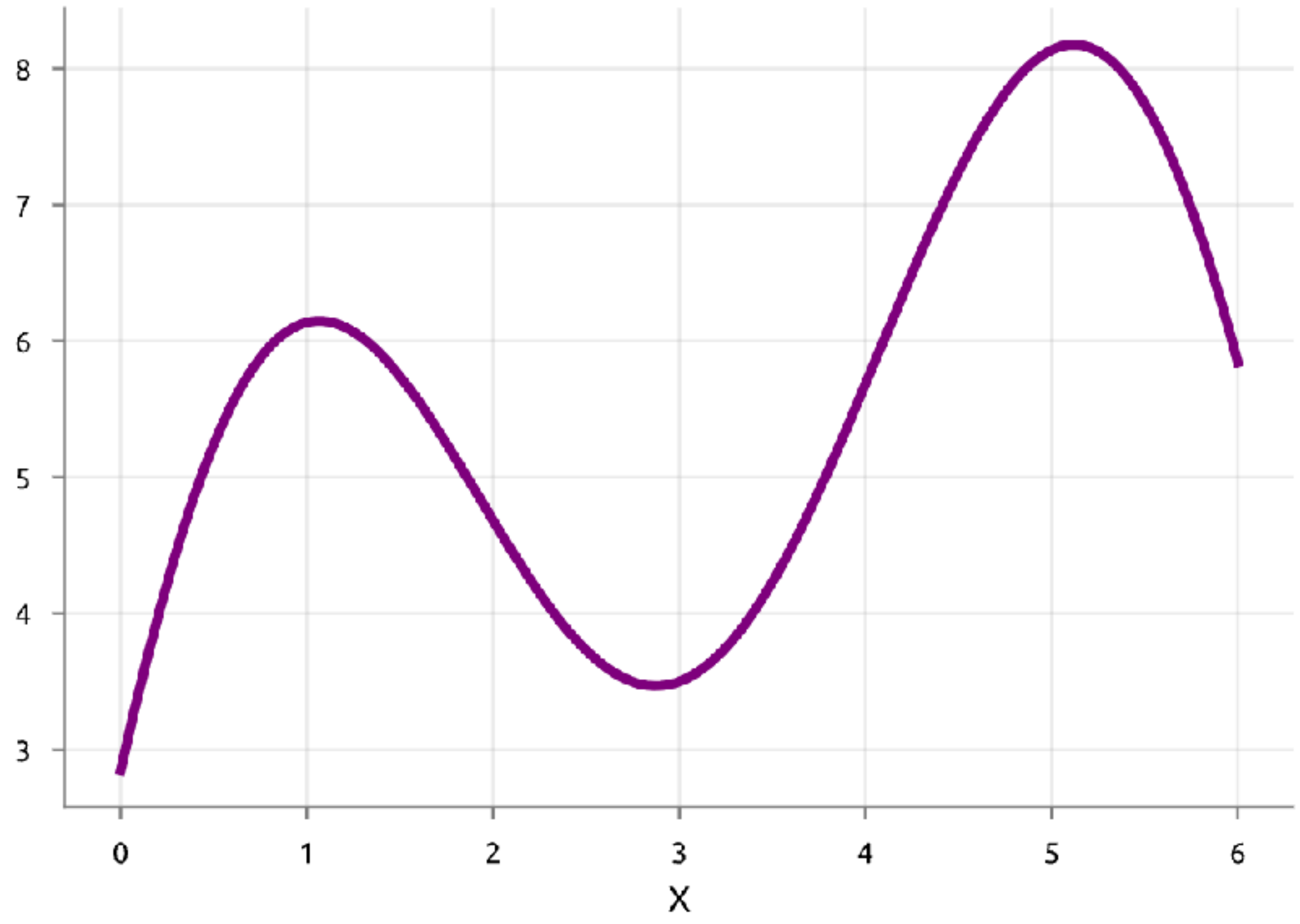
Applications where GPs shine

- We have a function $f(x)$ that is very expensive to evaluate
  - We want to approximate this function cheaply: **Active Learning**
  - We want to find the max value of $f(x)$: **BayesOpt**



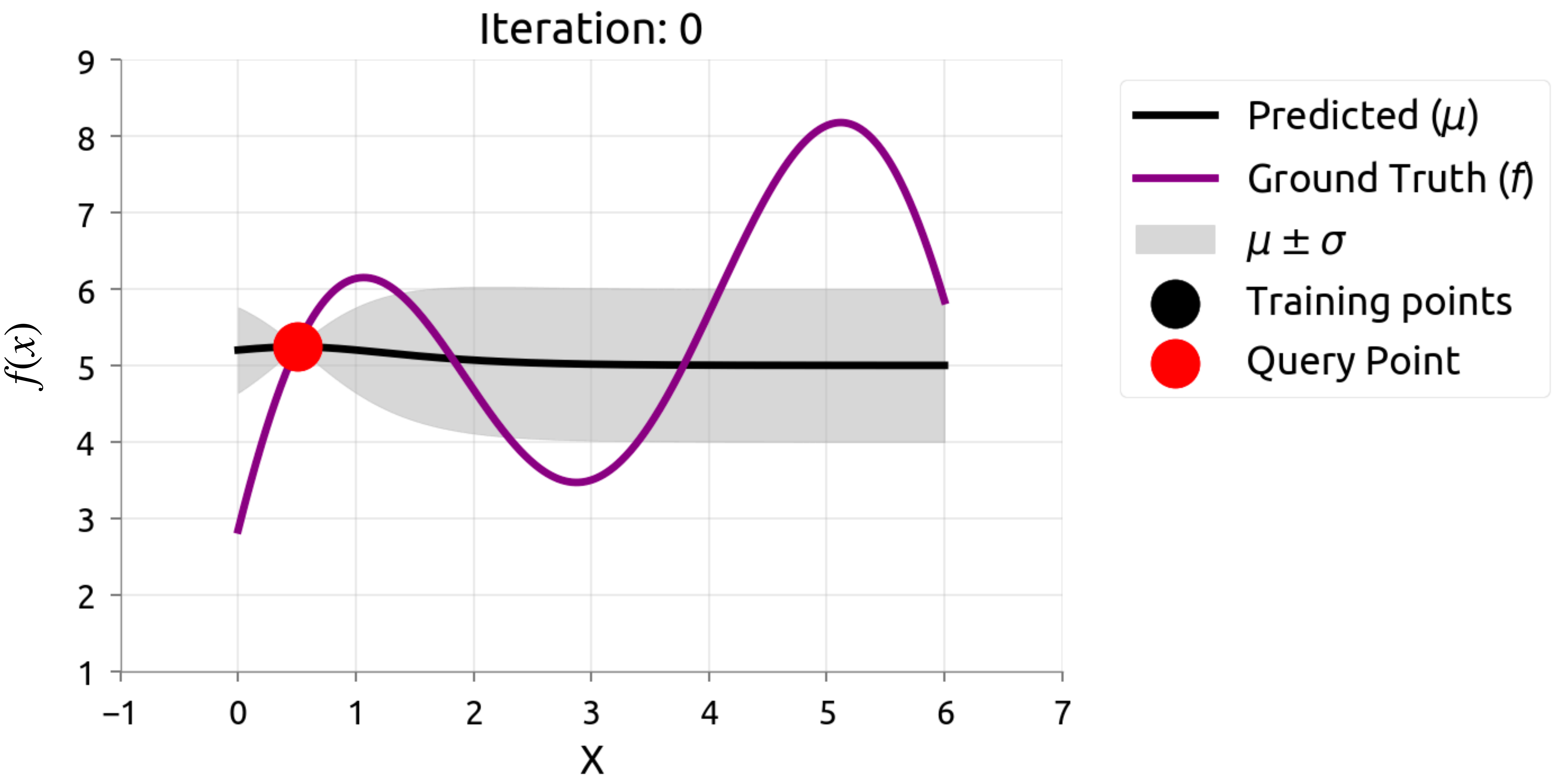
Applications where GPs shine

- We have a function $f(x)$ that is very expensive to evaluate
  - We want to approximate this function cheaply: **Active Learning**
  - We want to find the max value of $f(x)$: **BayesOpt**

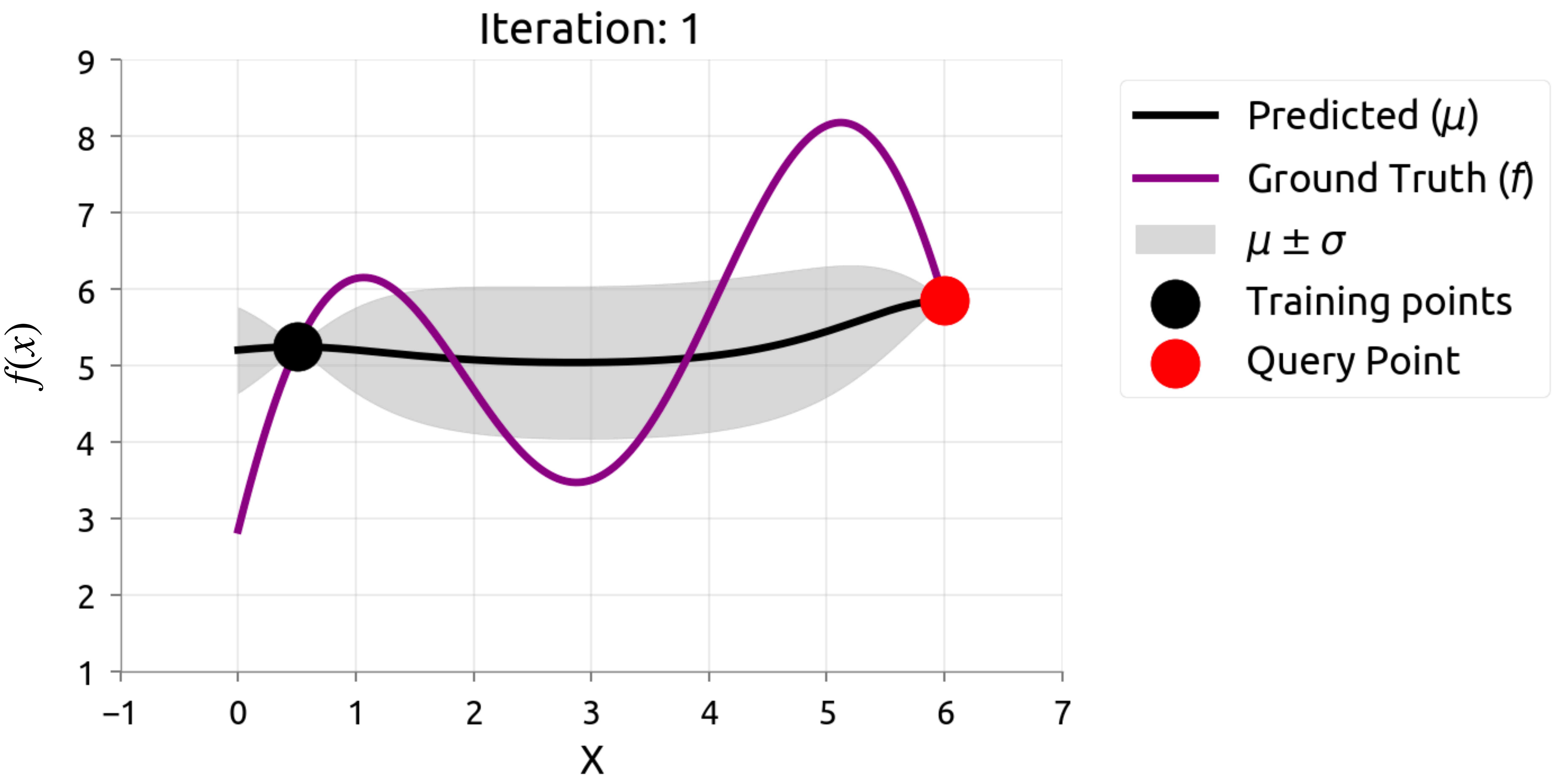


[1] https://distill.pub/2020/bayesian-optimization/

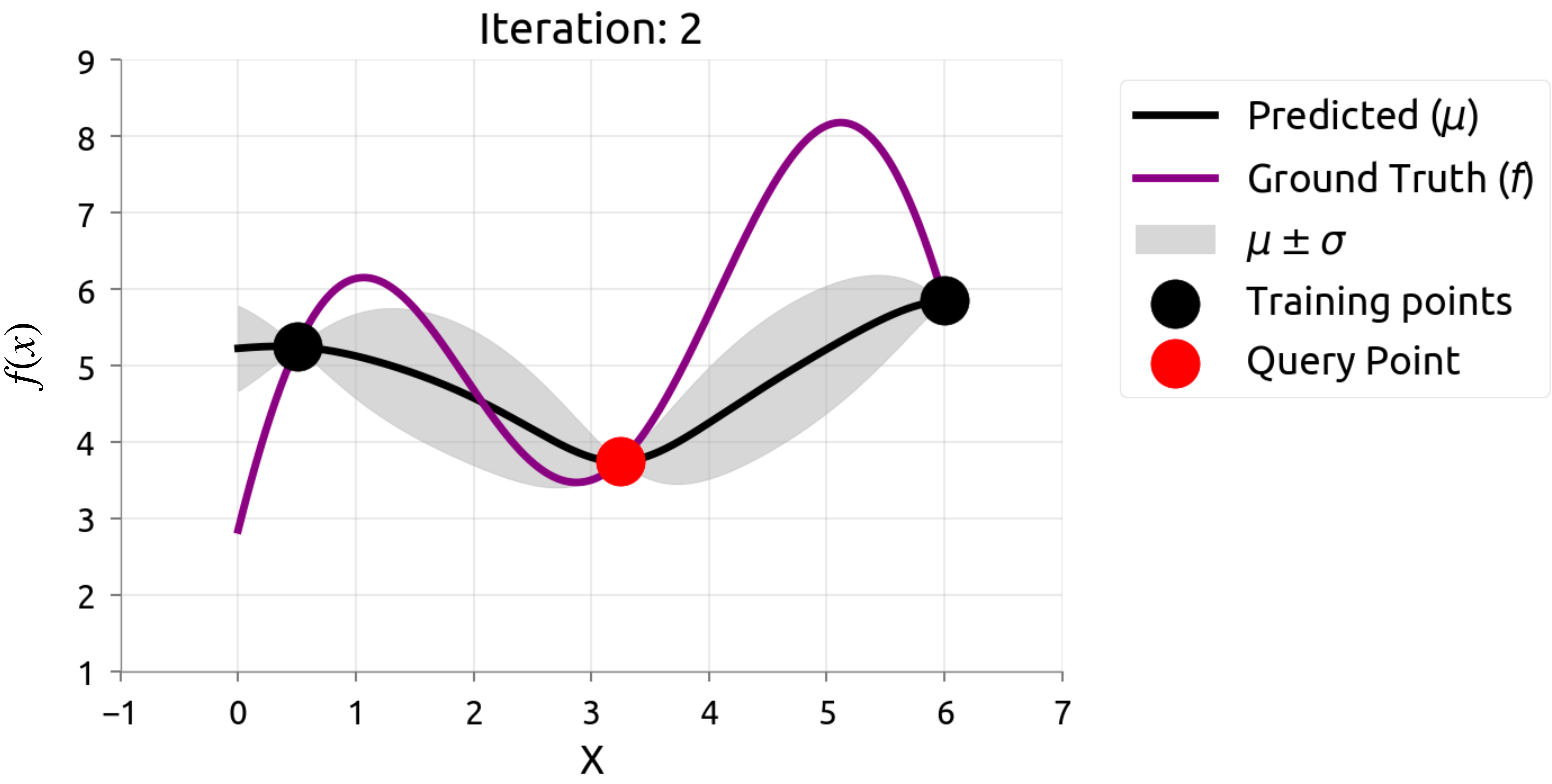
Active Learning



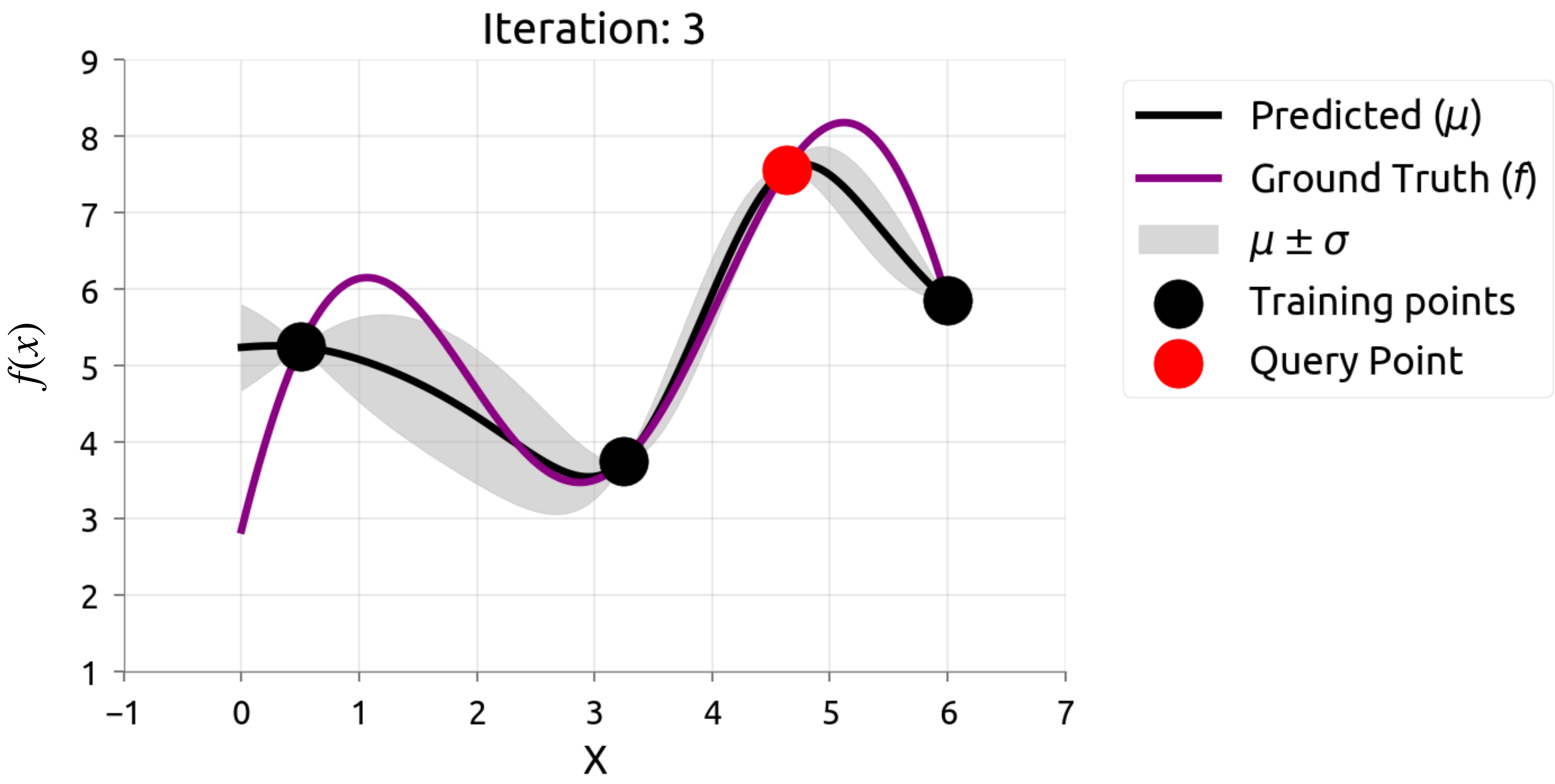
Active Learning



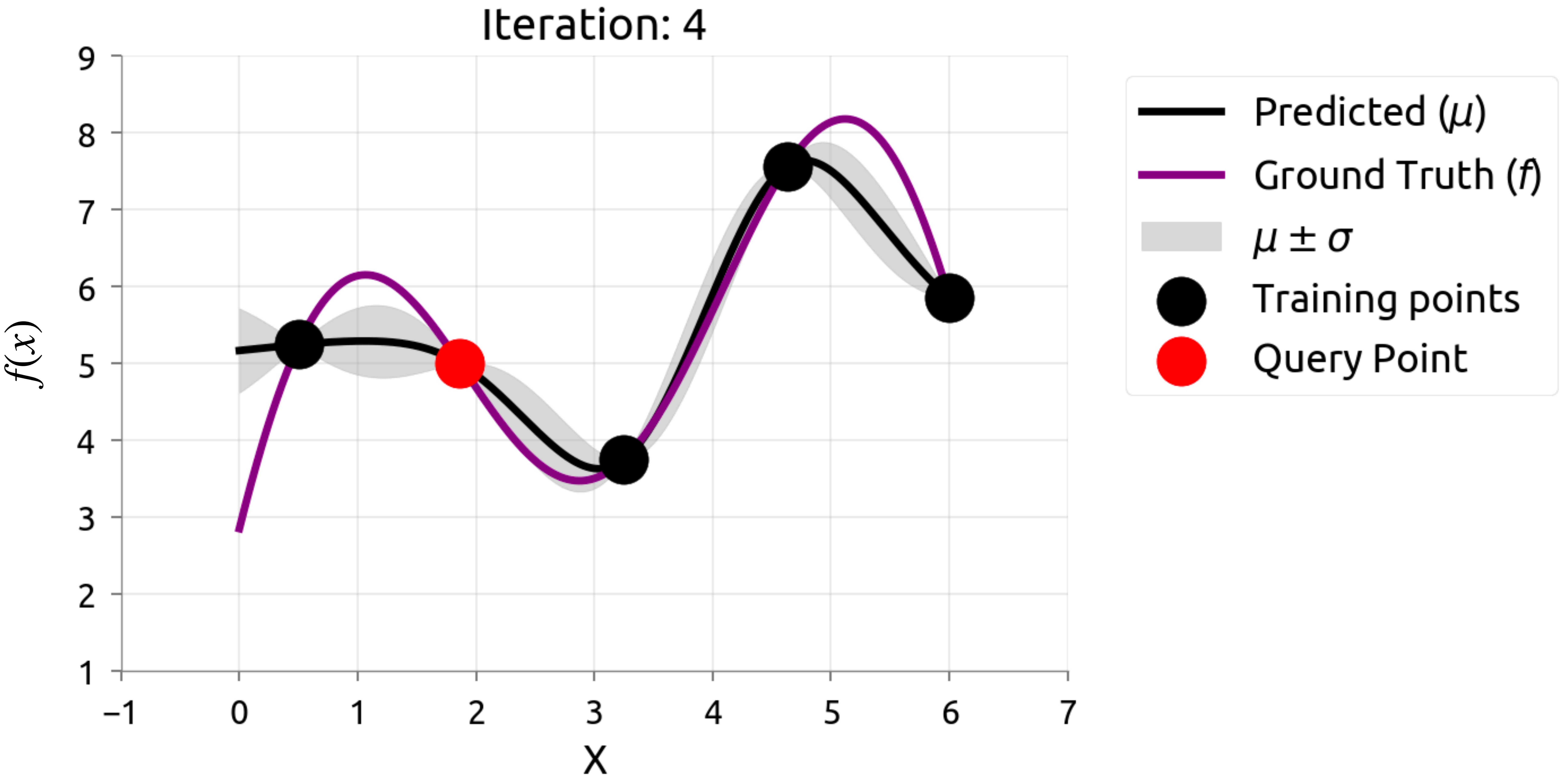
Active Learning



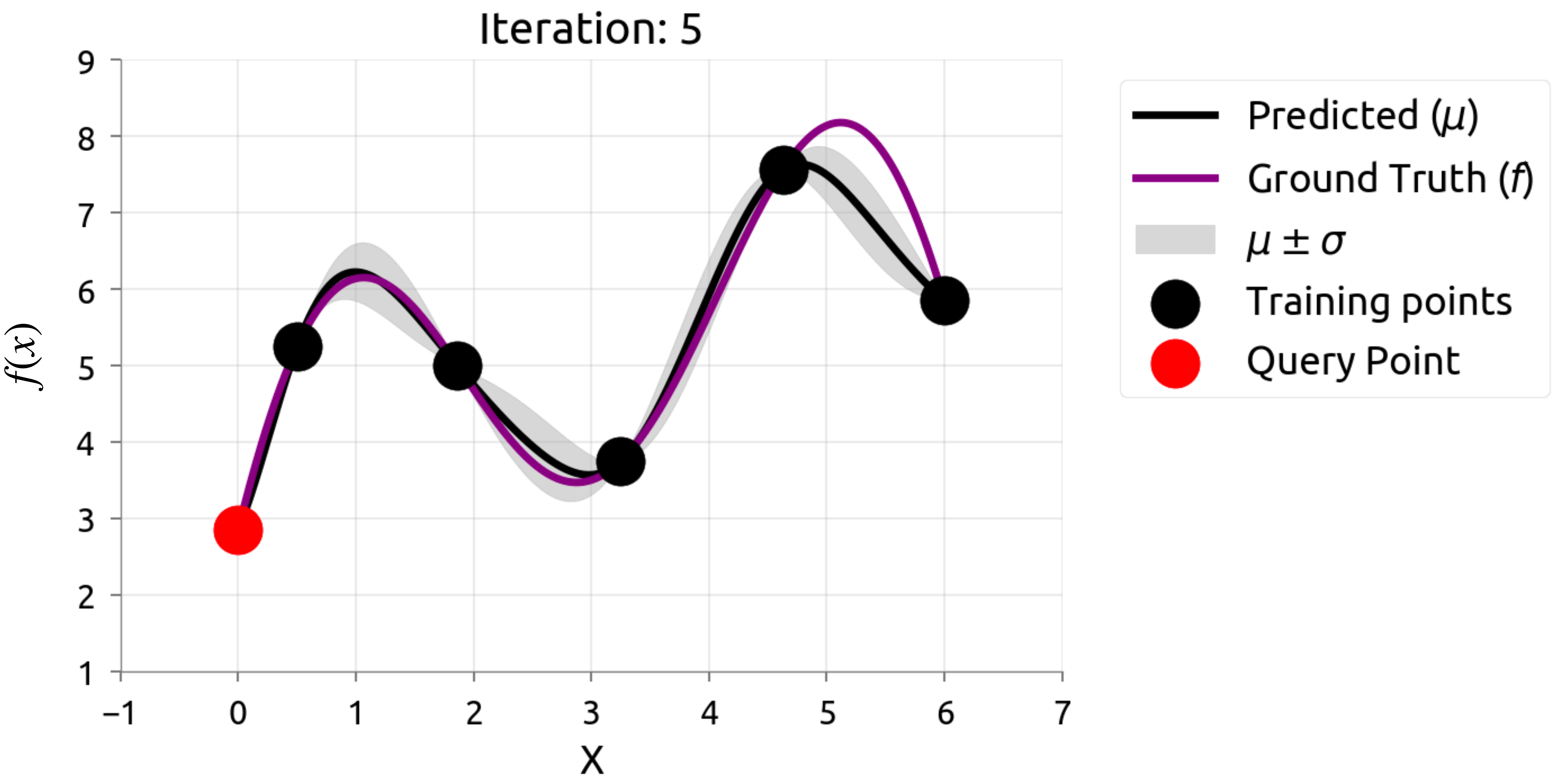
Active Learning



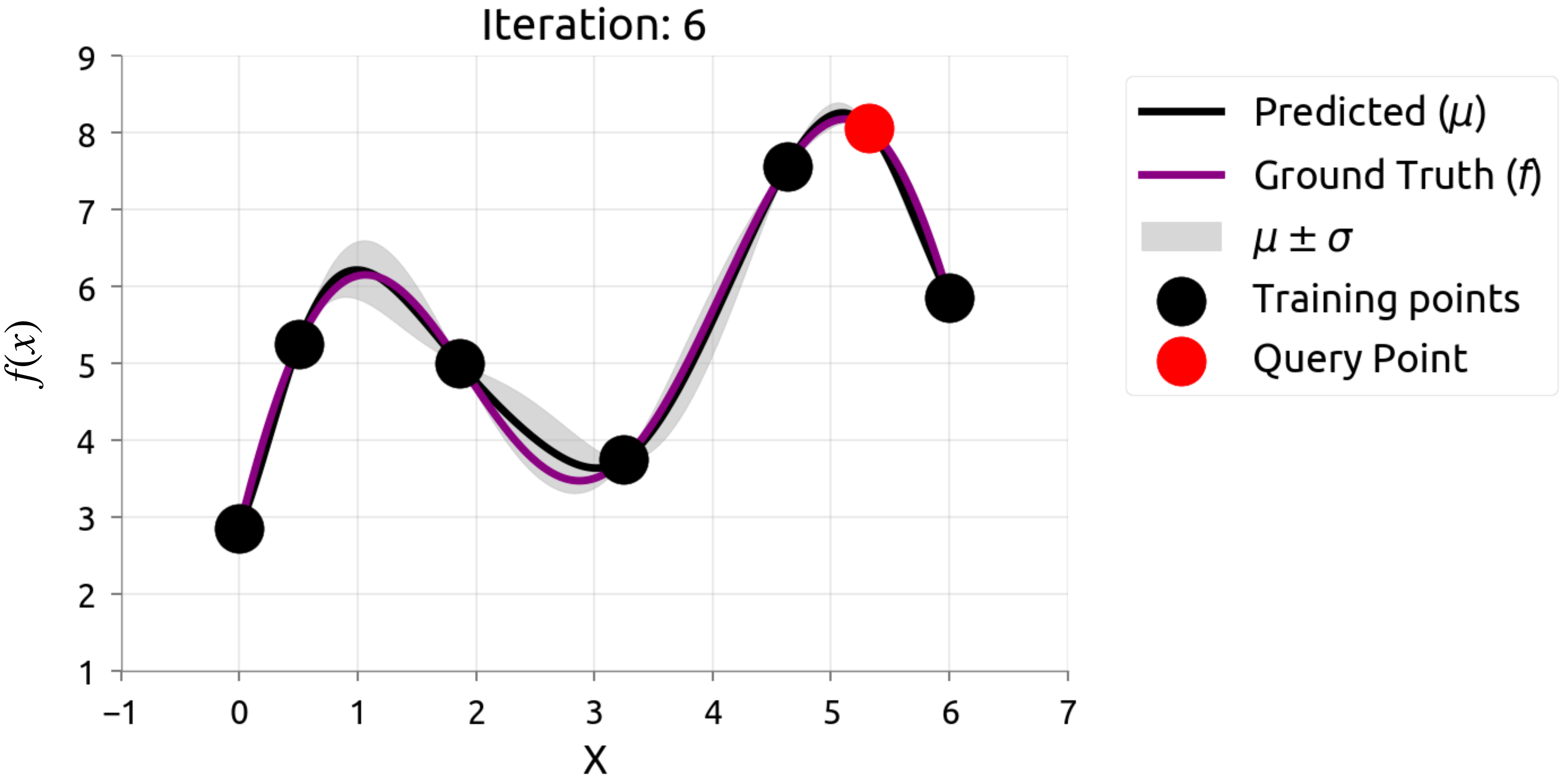
Active Learning



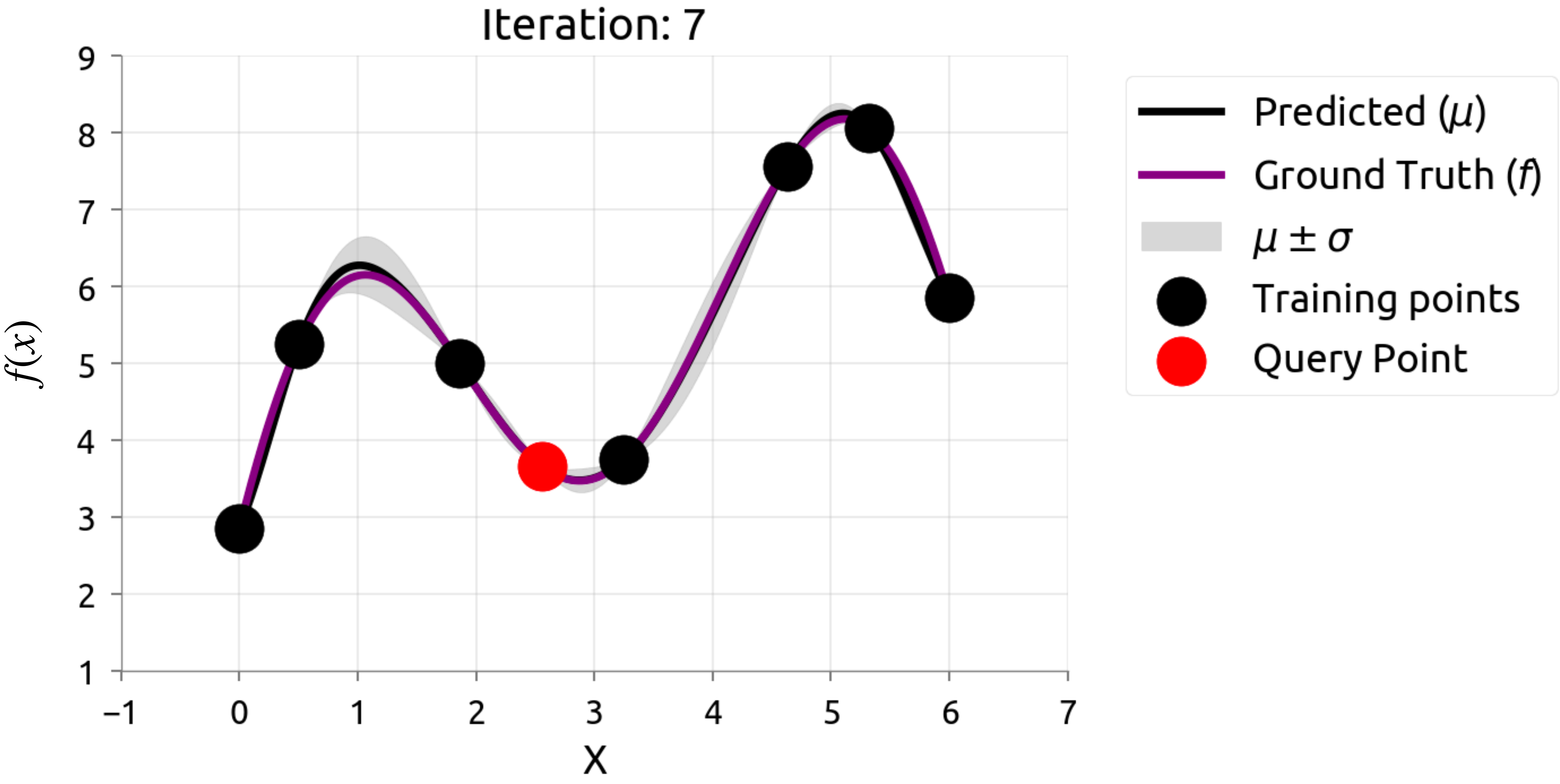
Active Learning



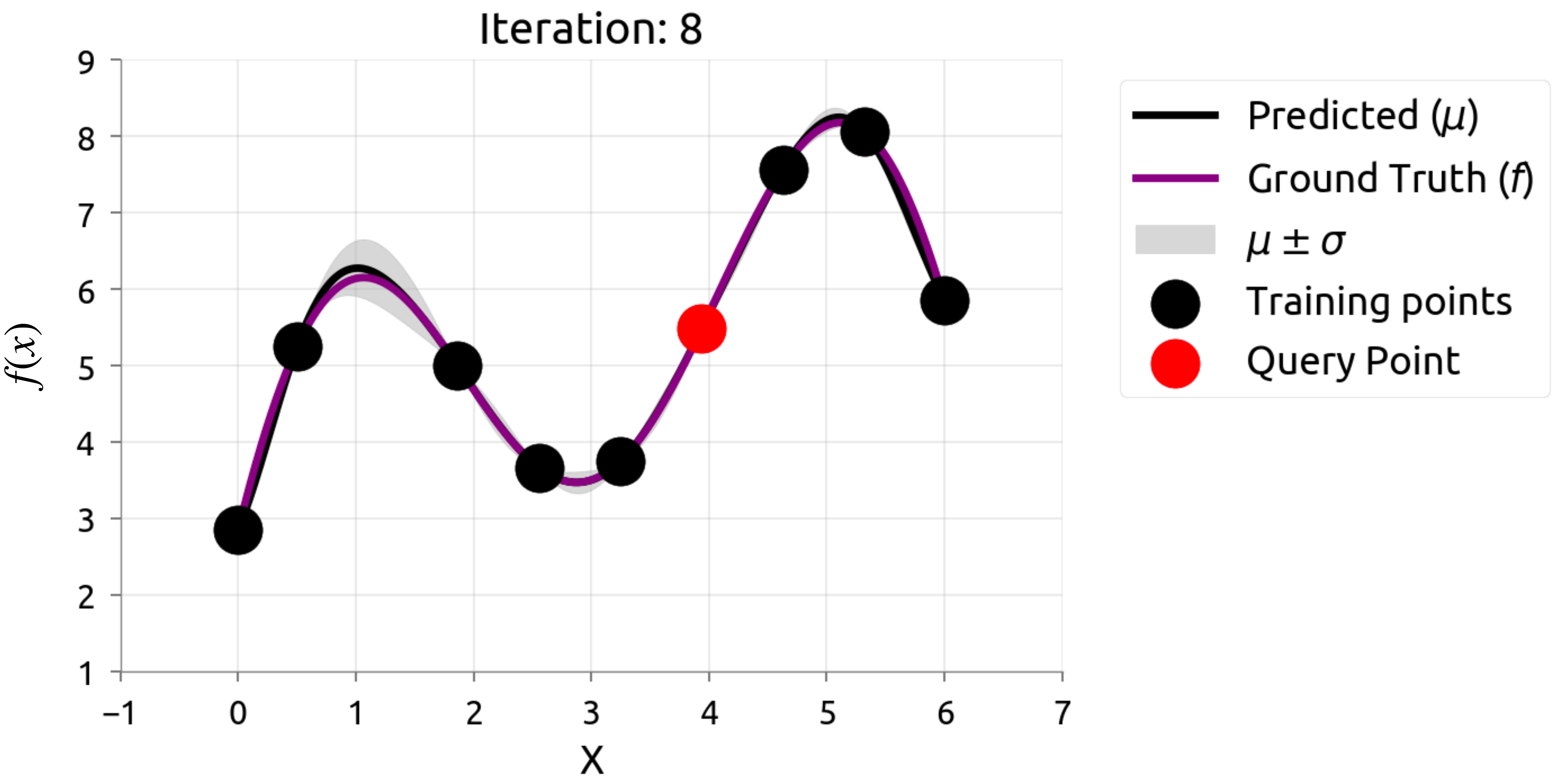
Active Learning



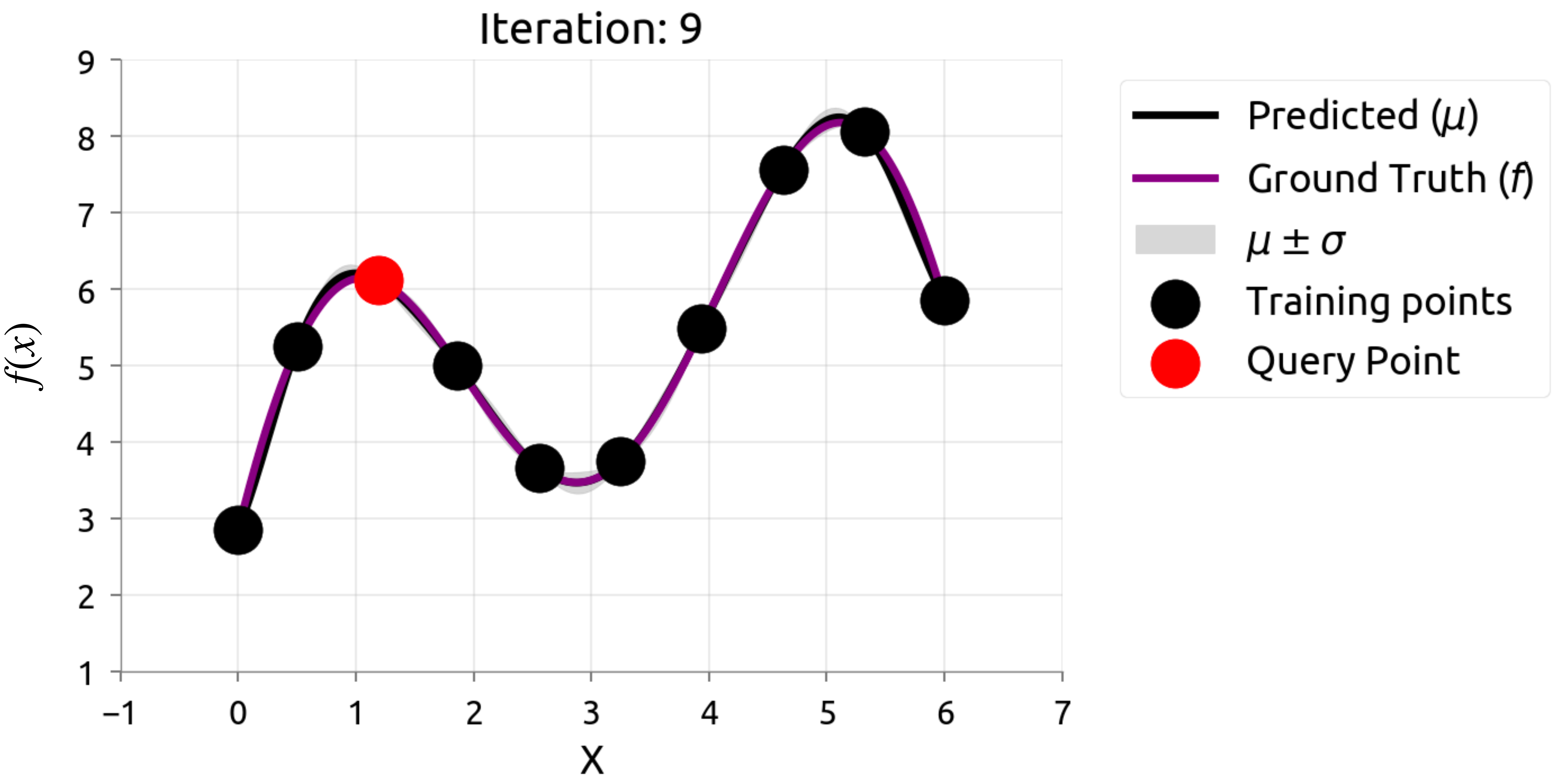
Active Learning



Active Learning

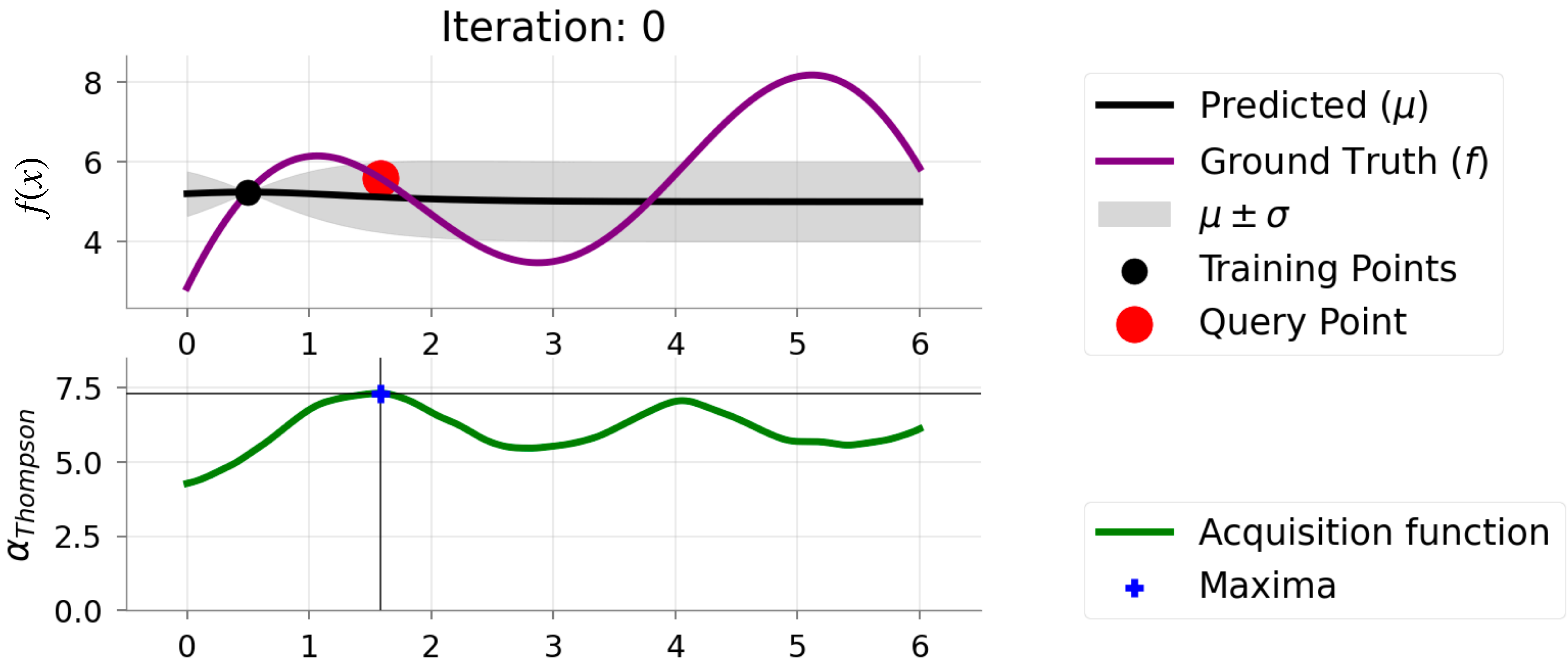


Active Learning

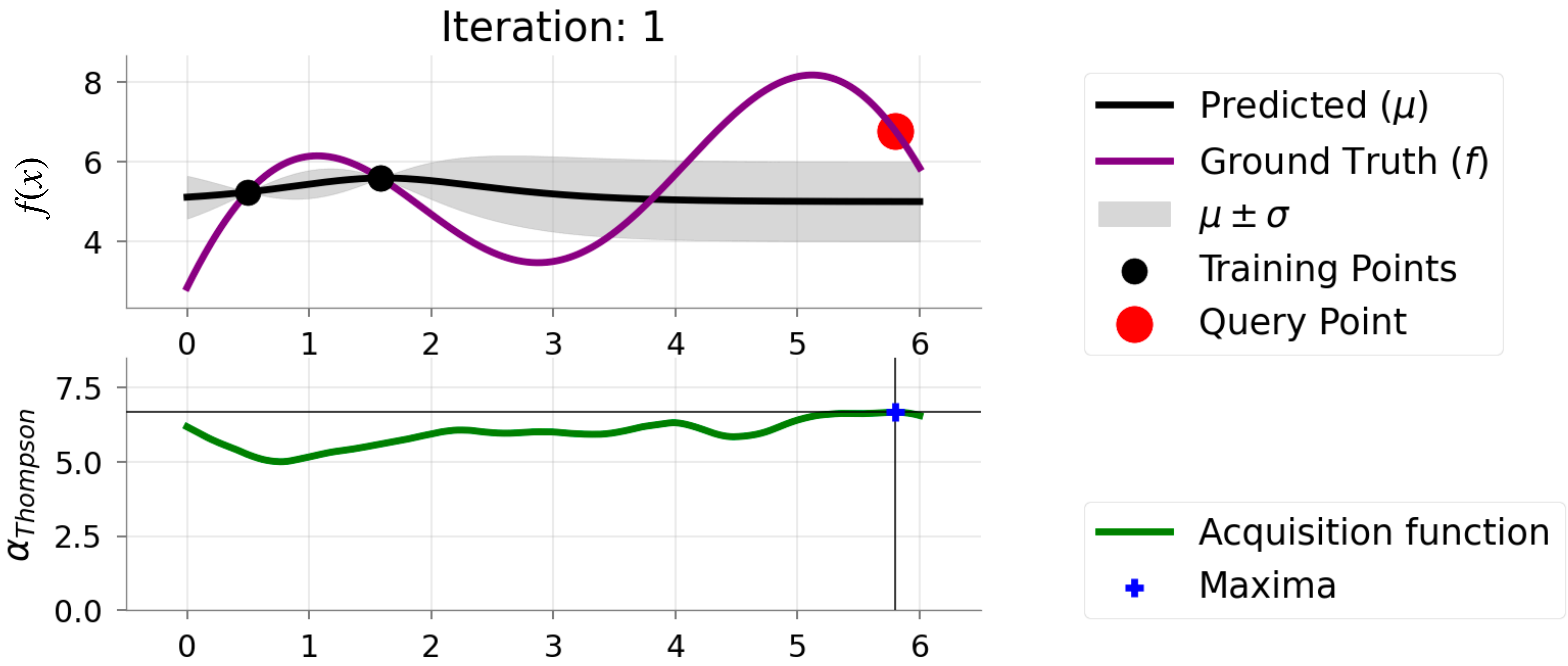


[1] https://distill.pub/2020/bayesian-optimization/

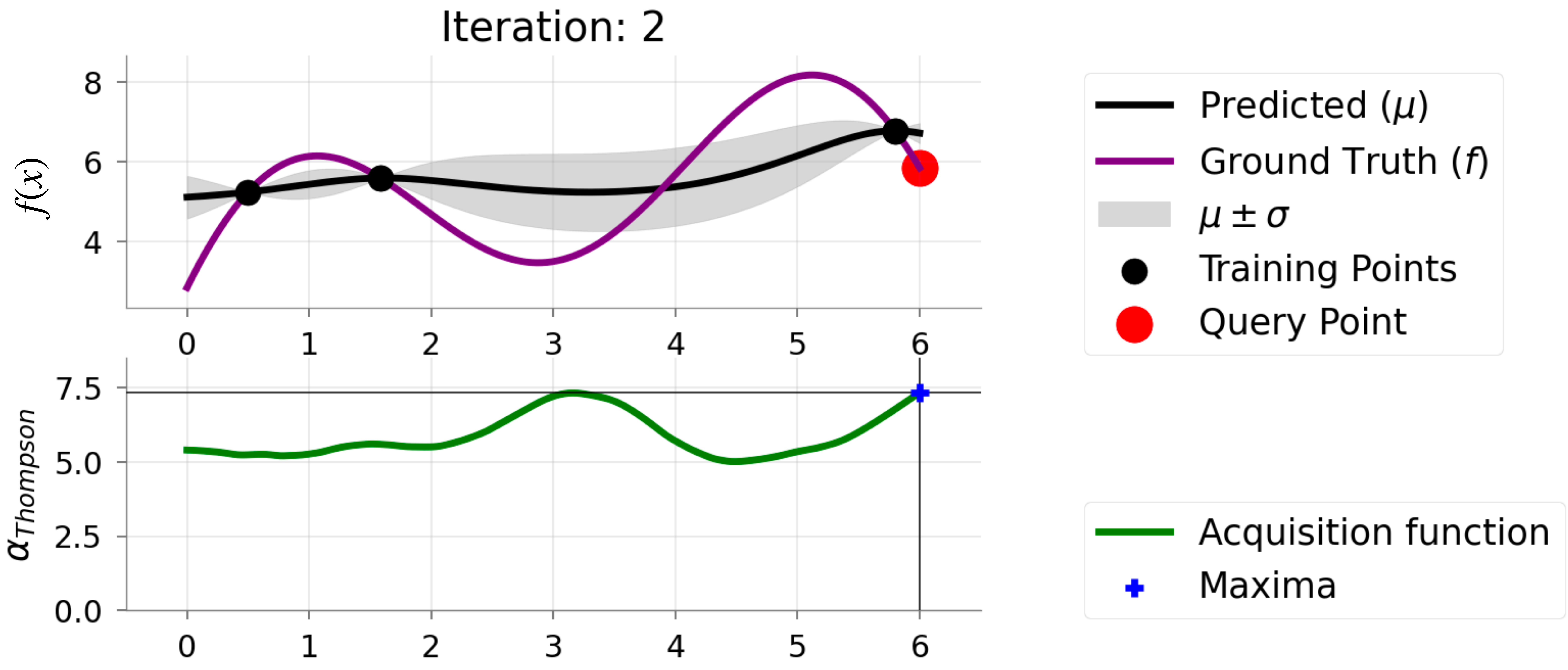
BayesOpt w/ Thompson Sampling



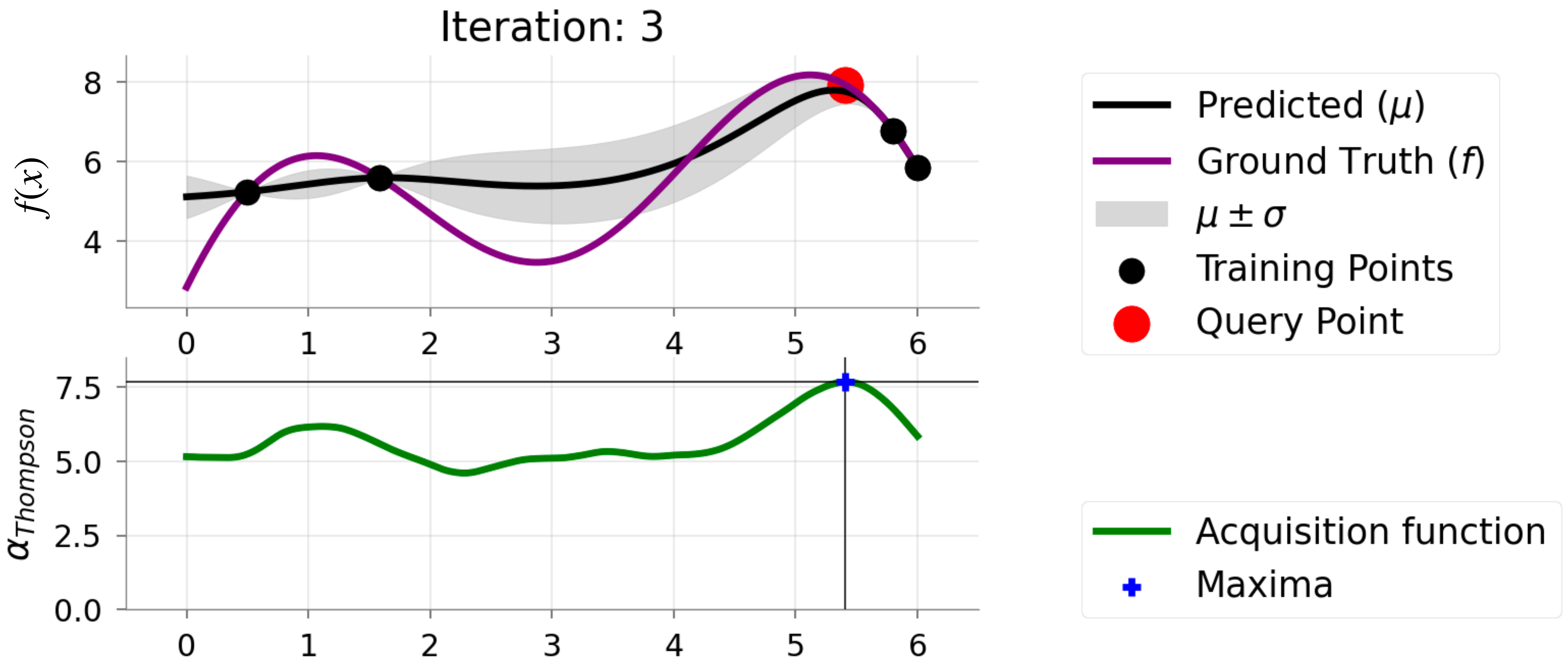
BayesOpt w/ Thompson Sampling



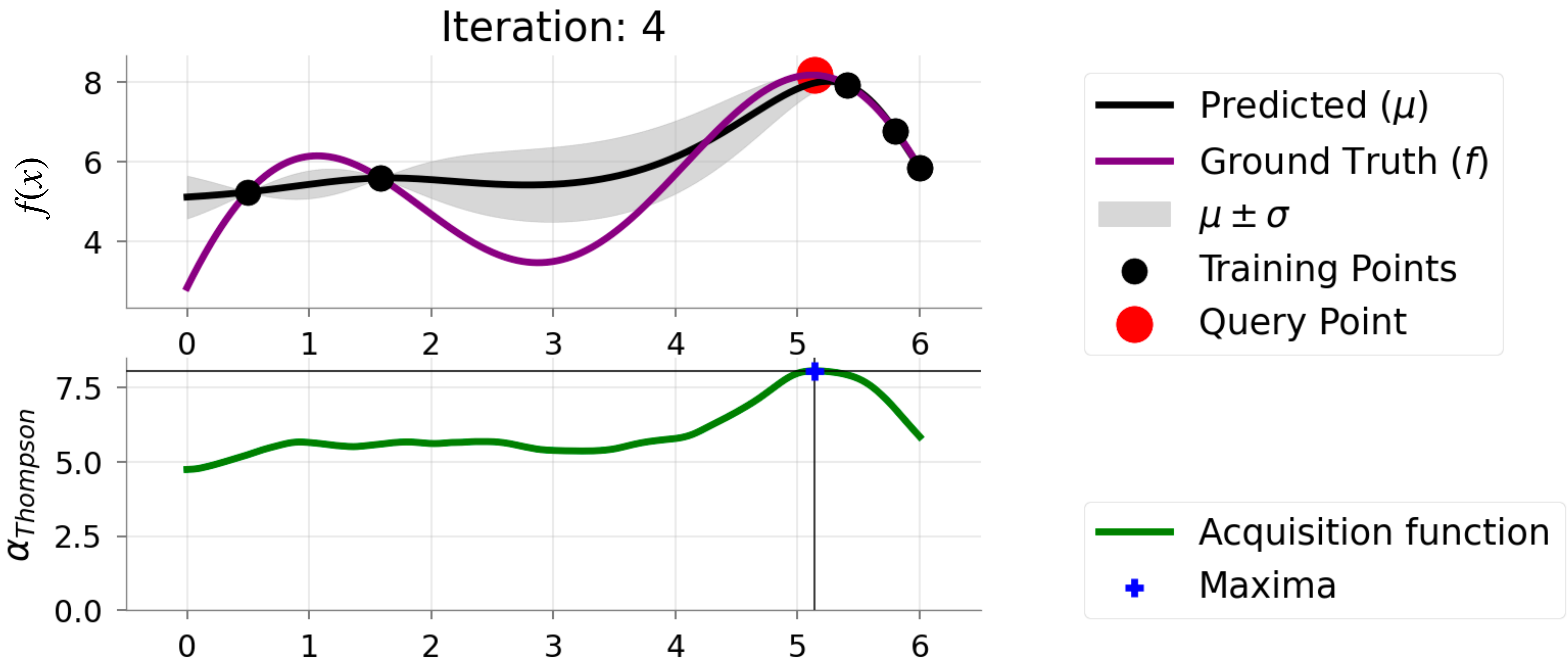
BayesOpt w/ Thompson Sampling



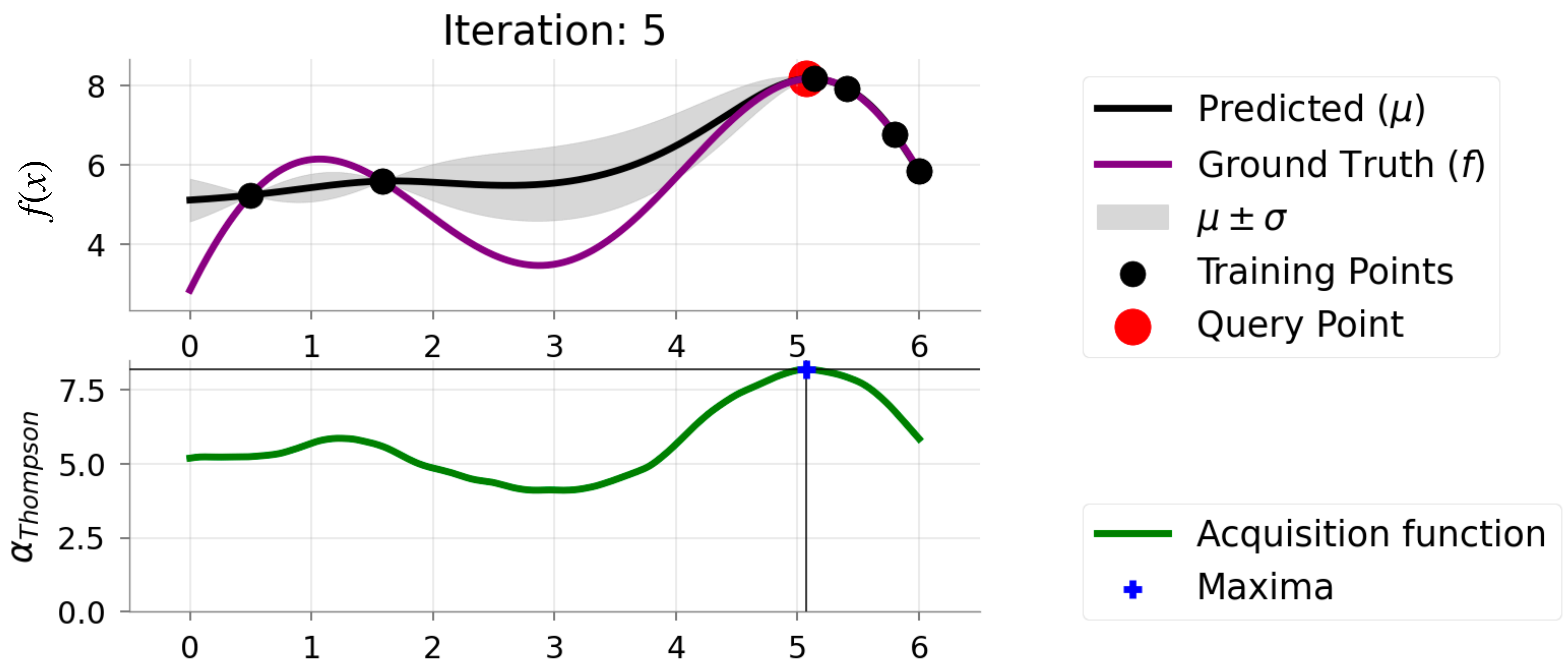
BayesOpt w/ Thompson Sampling



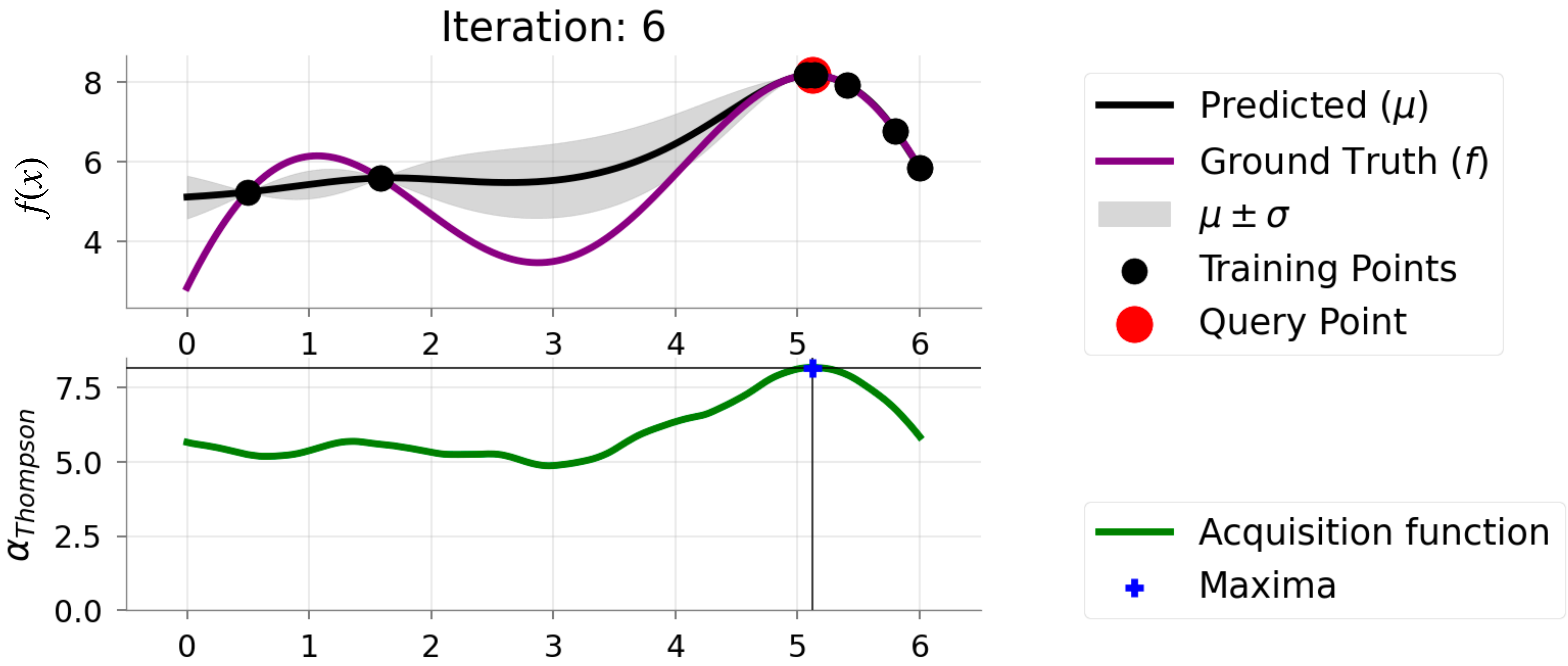
BayesOpt w/ Thompson Sampling



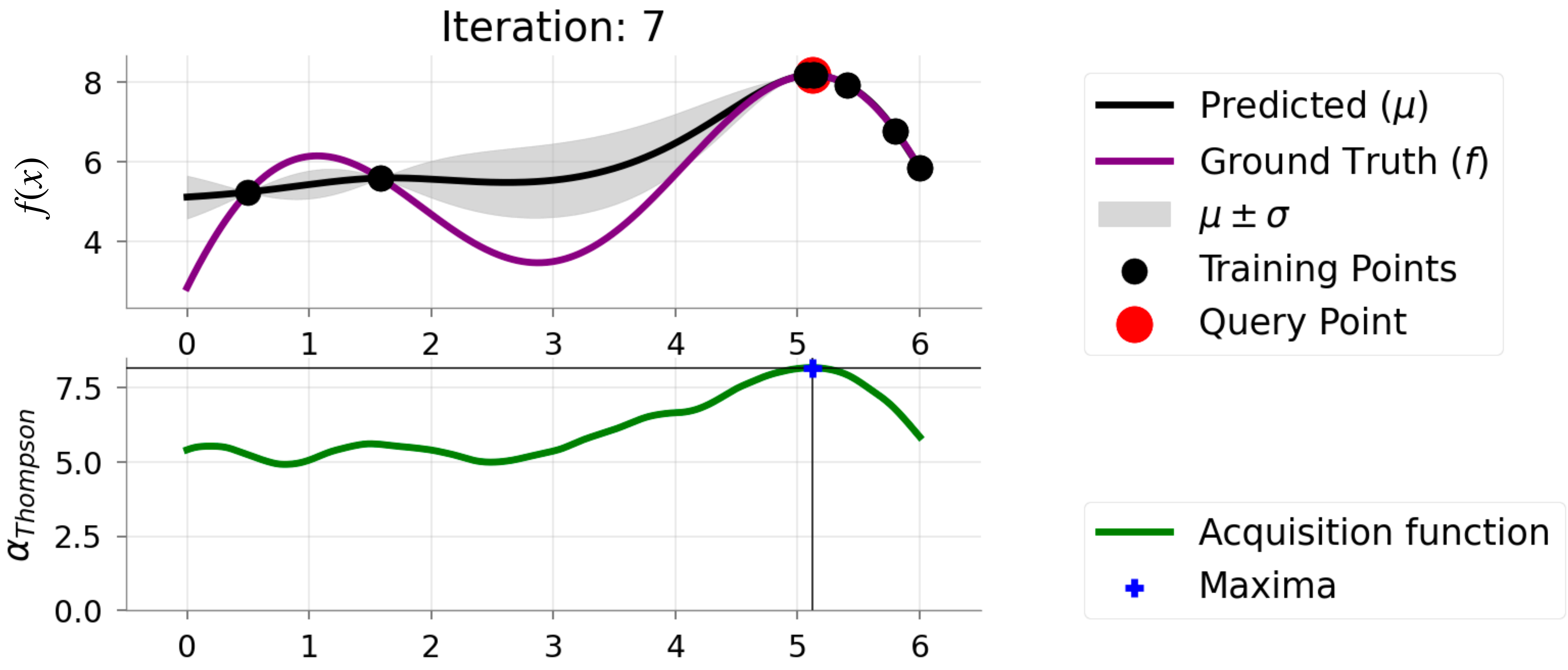
BayesOpt w/ Thompson Sampling



BayesOpt w/ Thompson Sampling



BayesOpt w/ Thompson Sampling



[1] https://distill.pub/2020/bayesian-optimization/

Computational Considerations

$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^{\top} & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \{(X_1, y_1), \ldots, (X_n, y_n)\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$

Posterior Distribution

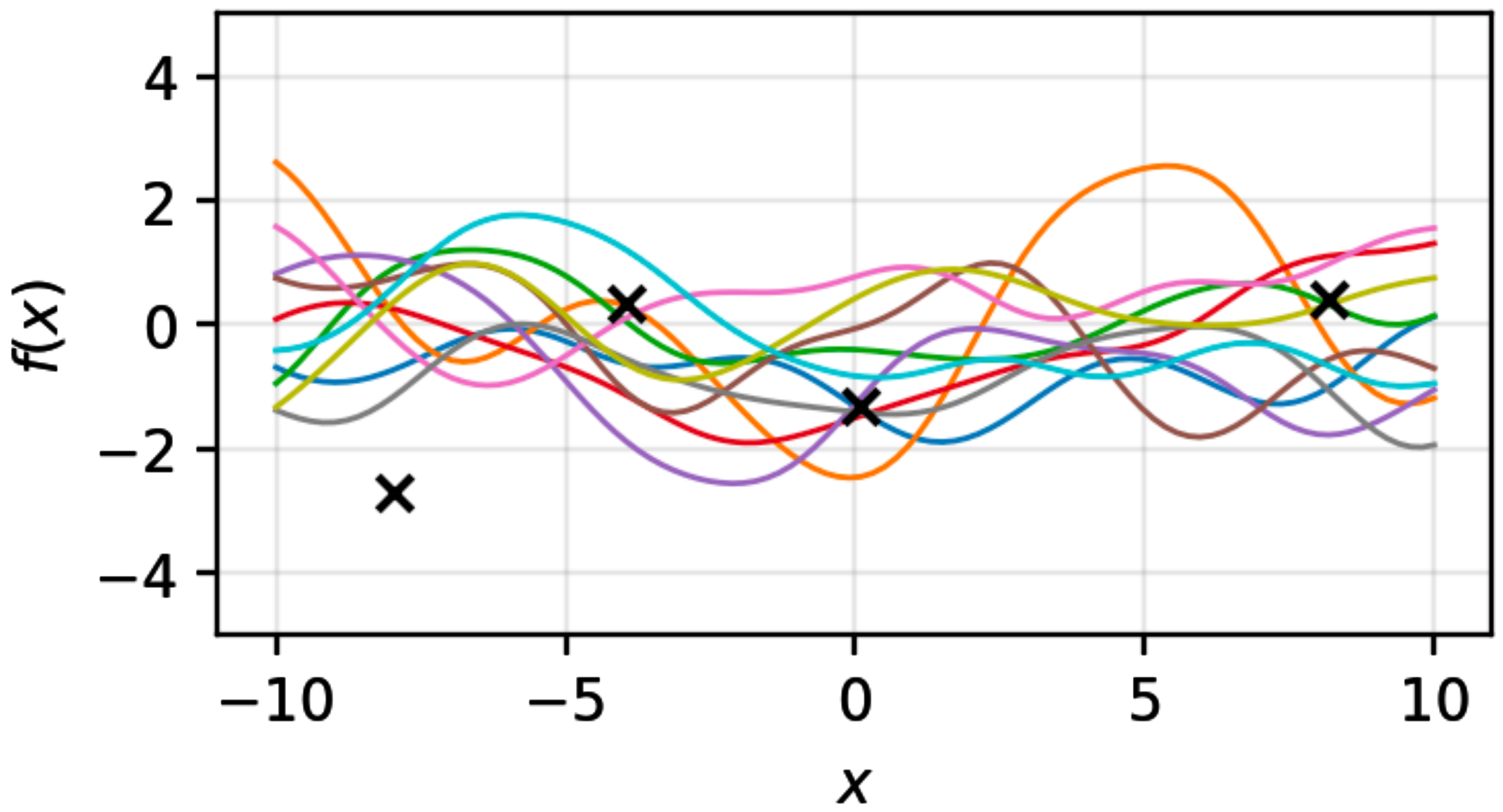
$$p(f_* \,|\, f, X, y) = \mathcal{N}(\mu_{f|y}, \Sigma_{f|y})$$

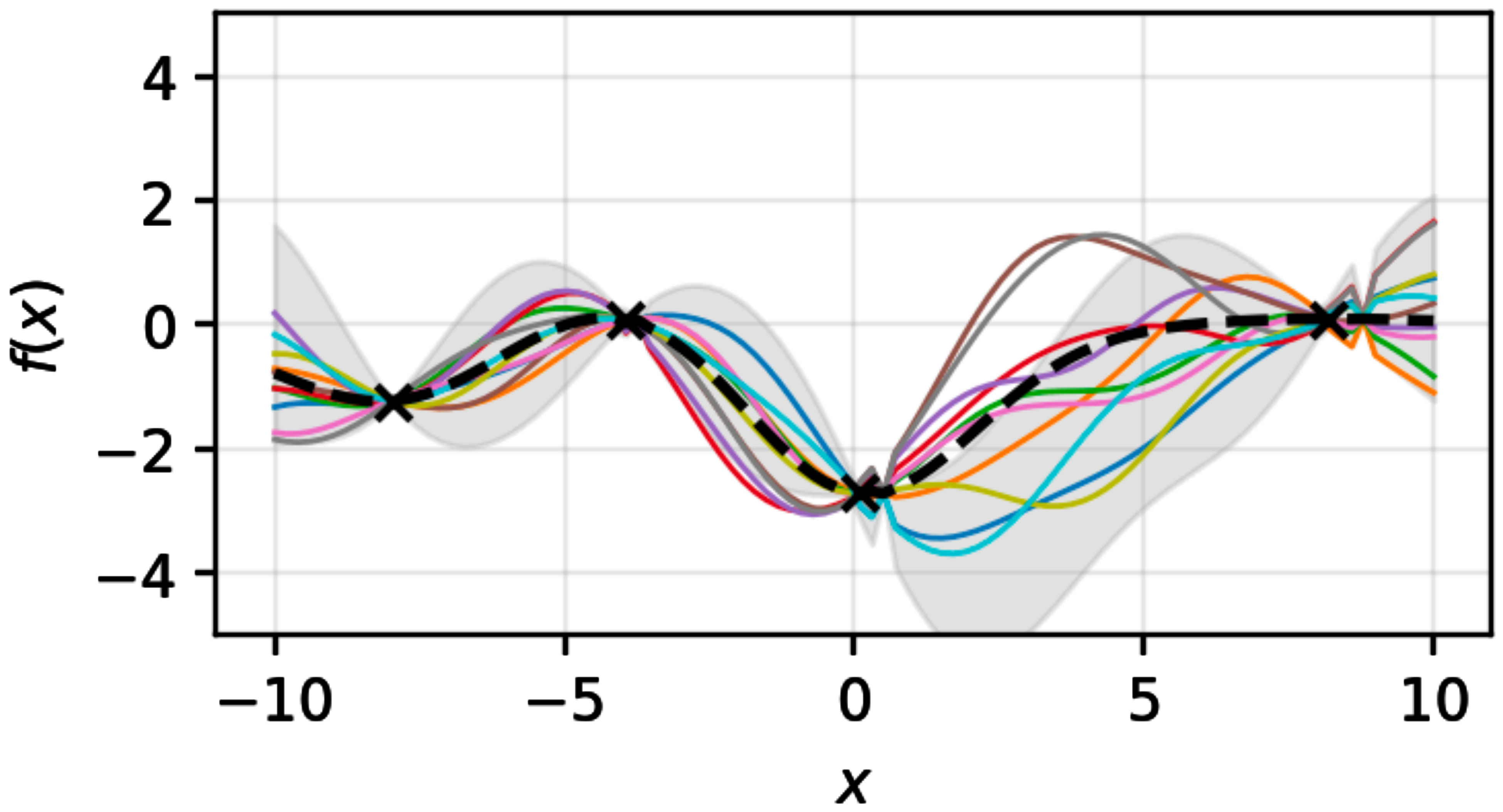
Predictive Mean

$$\mu_{f|y} = K_{*n}(K_{nn} + \sigma^2 I)^{-1} y$$

Uncertainty Estimate

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$





Computational Considerations

$$\left[\begin{pmatrix} f(X_*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Dataset

$$(X, y) = \{(X_1, y_1), \ldots, (X_n, y_n)\}$$

Observation noise assumption

$$y_i \sim \mathcal{N}(f(X_i), \sigma^2)$$

Posterior Distribution

$$p(f_* \mid f, X, y) = \mathcal{N}(\mu_{f|y}, \Sigma_{f|y})$$

Predictive Mean

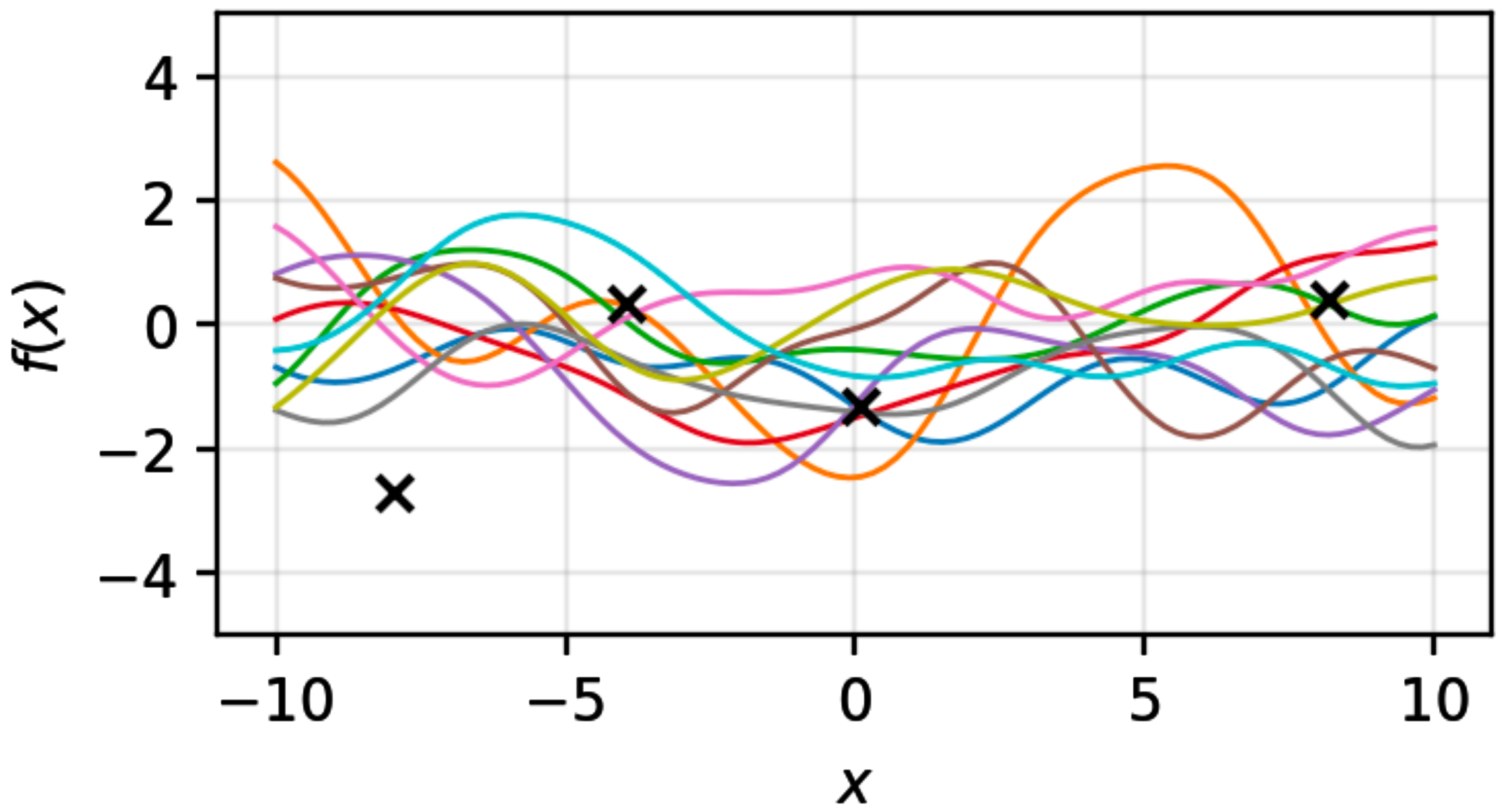
$\mathcal{O}(n^3)$

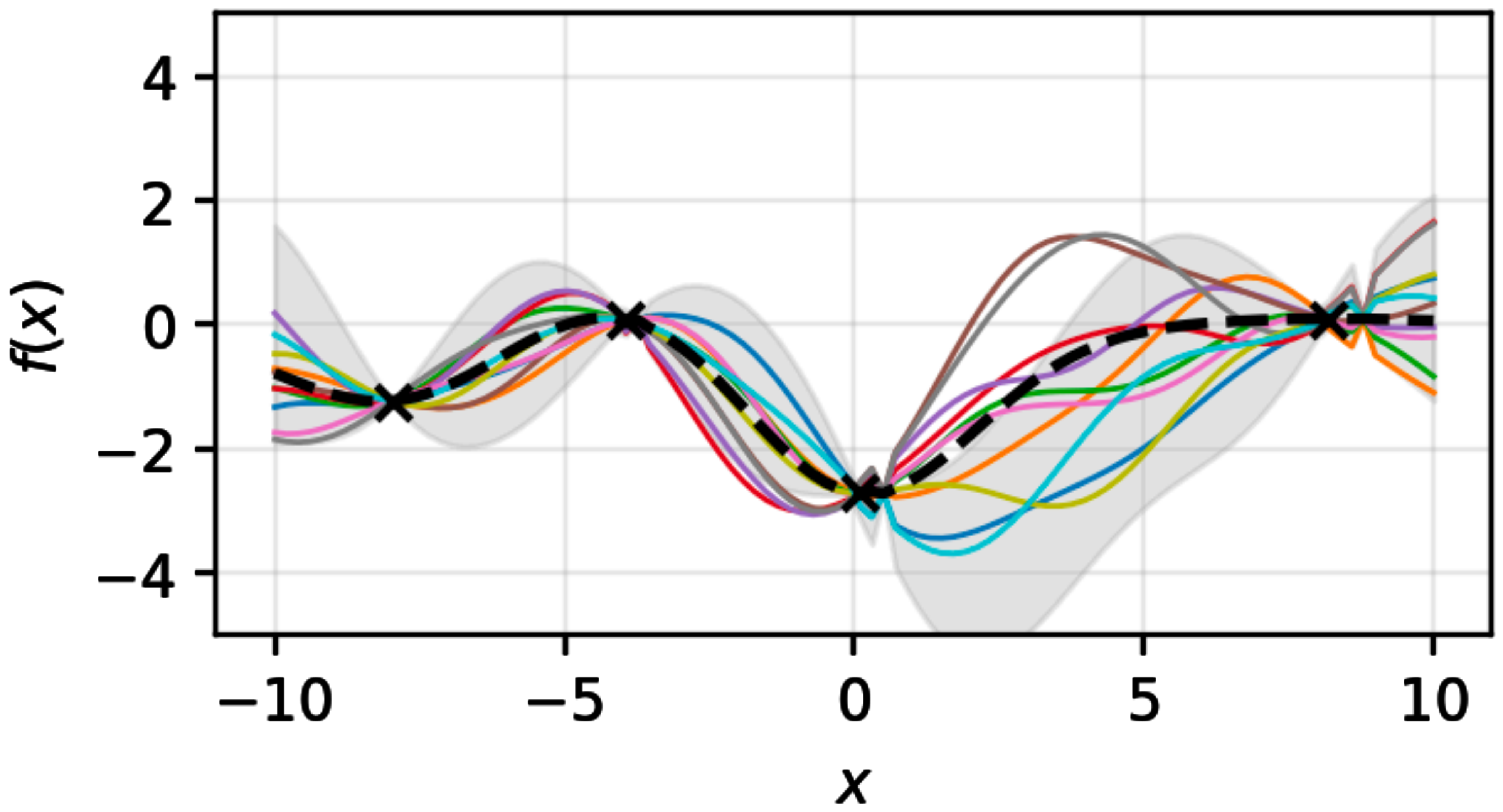
$$\mu_{f|y} = K_{*n}(K_{nn} + \sigma^2 I)^{-1} y$$

Uncertainty Estimate

$\mathcal{O}(n^3)$

$$\Sigma_{f|y} = K_{**} - K_{*n}^\top (K_{nn} + \sigma^2 I)^{-1} K_{n*}$$





Variational Gaussian Processes

Idea: Kernel matrix can be approximated as $K_{nn} \approx UQU^T$ where $Q \in \mathbb{R}^{m \times m}$

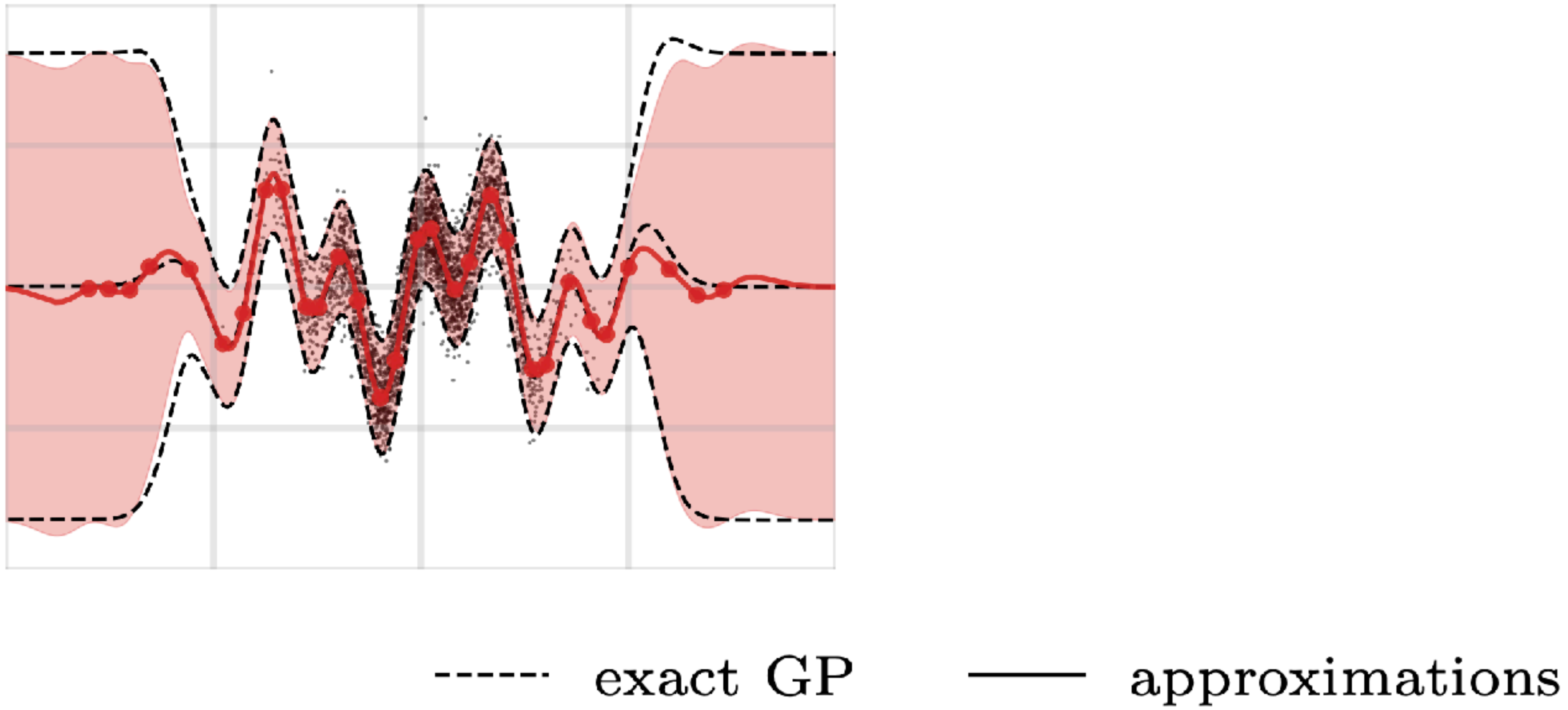
- Cost is $O(nm^2)$ or $O(m^3)$ for m the rank of the approximation

Variational Gaussian Processes

Idea: Kernel matrix can be approximated as $K_{nn} \approx UQU^T$ where $Q \in \mathbb{R}^{m \times m}$

- Cost is $O(nm^2)$ or $O(m^3)$ for m the rank of the approximation

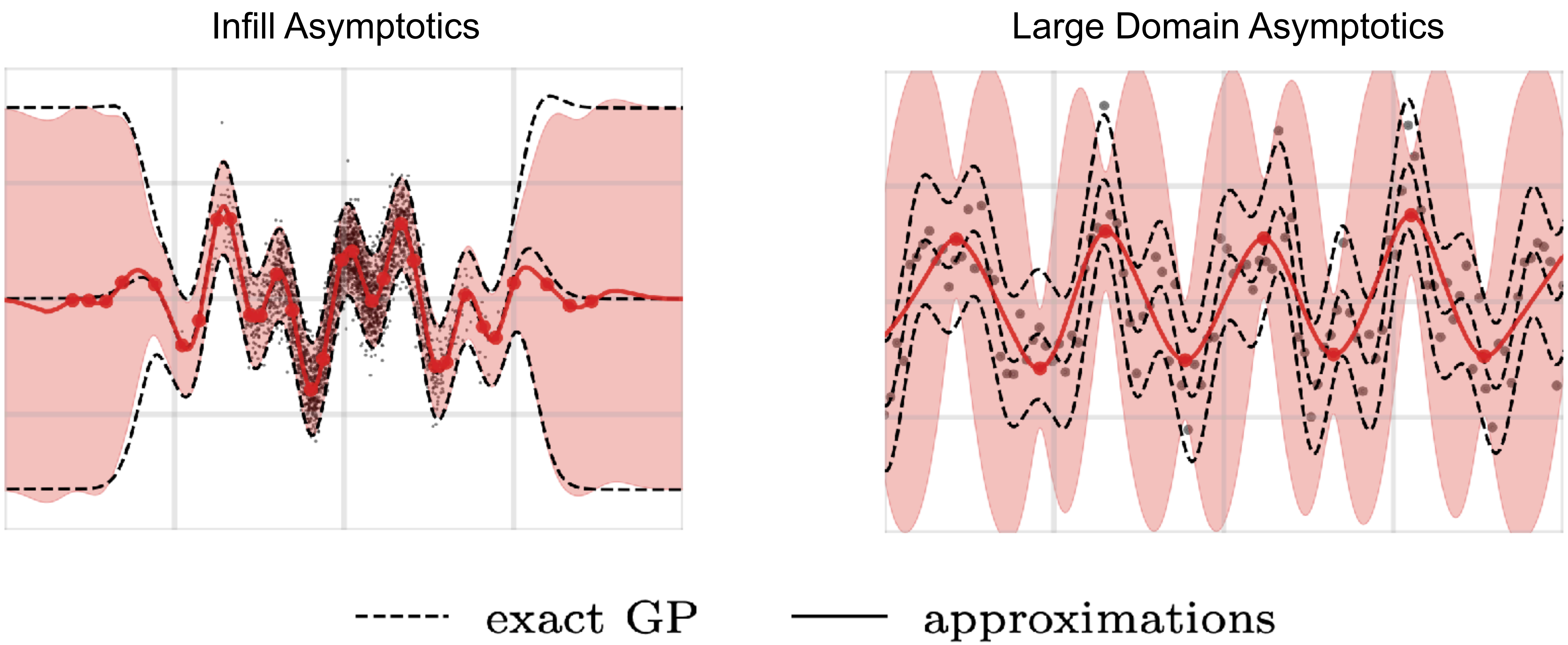
Infill Asymptotics



Variational Gaussian Processes

Idea: Kernel matrix can be approximated as $K_{nn} \approx UQU^T$ where $Q \in \mathbb{R}^{m \times m}$

- Cost is $O(nm^2)$ or $O(m^3)$ for m the rank of the approximation



Conjugate Gradients

Conjugate Gradients

We can solve the linear system $(K_{nn} + \sigma^2 I)^{-1} b$ iteratively instead of inverse

Conjugate Gradients

We can solve the linear system $(K_{nn} + \sigma^2 I)^{-1} b$ iteratively instead of inverse

Each step requires a matrix multiplication with $K_{nn} + \sigma^2 I \in R^{n \times n}$ (cost $O(n^2)$)

Conjugate Gradients

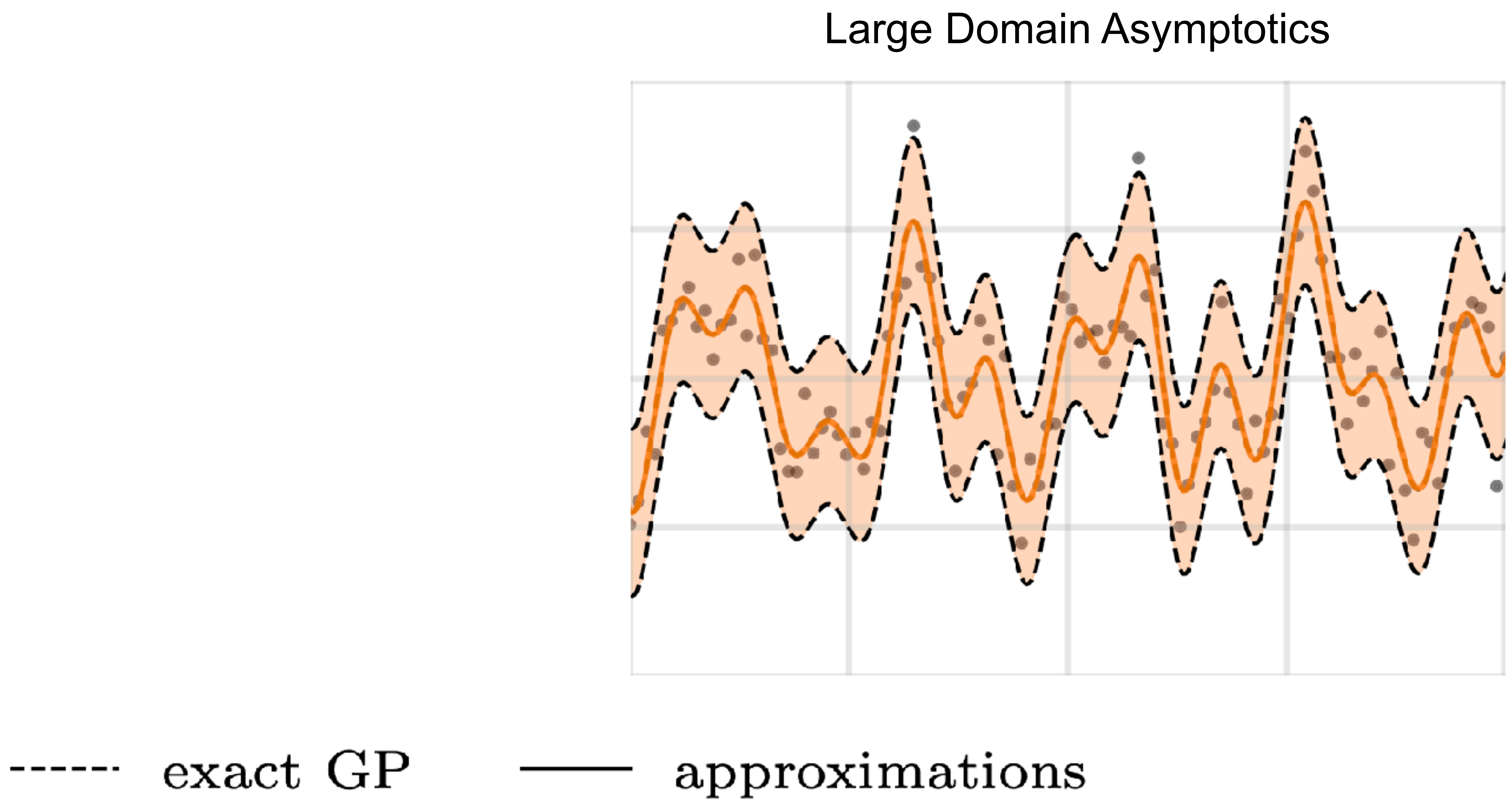
We can solve the linear system $(K_{nn} + \sigma^2 I)^{-1} b$ iteratively instead of inverse

Each step requires a matrix multiplication with $K_{nn} + \sigma^2 I \in R^{n \times n}$ (cost $O(n^2)$)

Algorithm converges in at most n steps but, in practise, for some tolerance ϵ

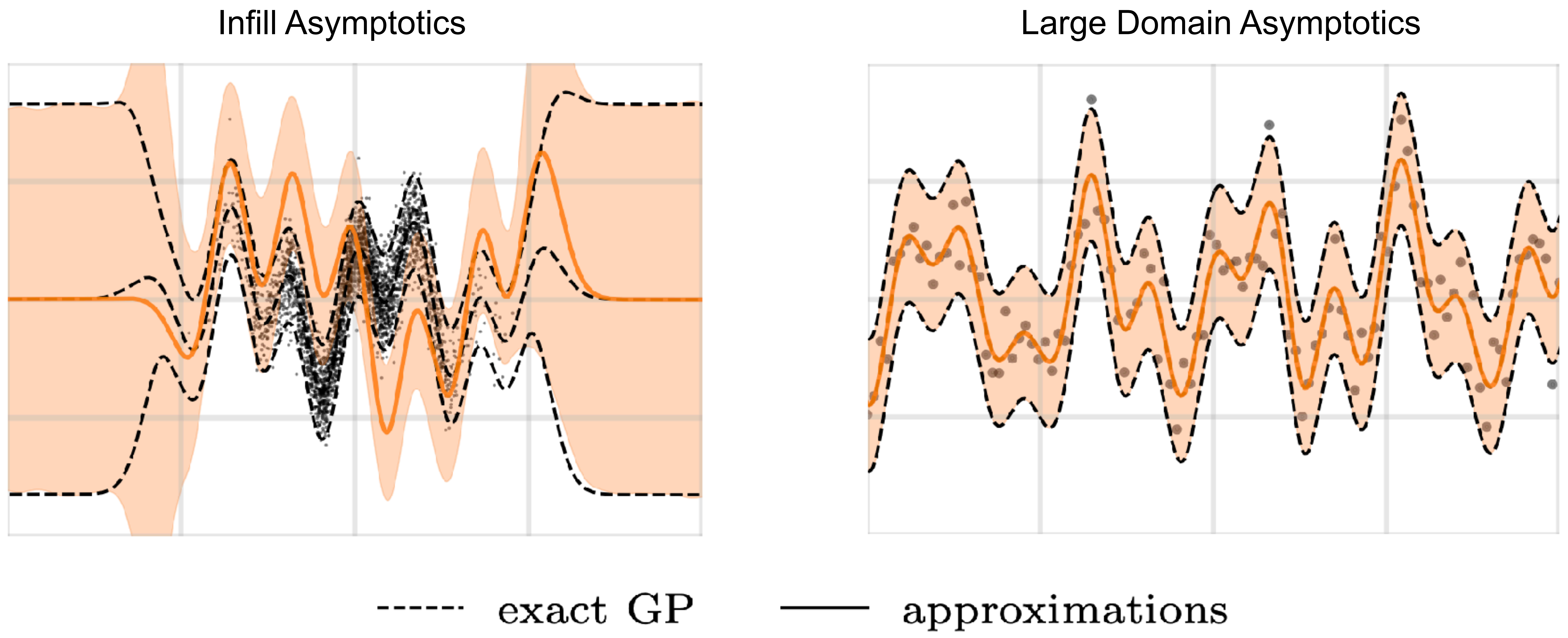
$$O\left(\sqrt{\mathrm{cond}(K + \sigma^2 I)}\ \log \frac{\mathrm{cond}(K_{nn} + \sigma^2 I)\|b\|}{\epsilon}\right) \qquad \mathrm{cond}(K_{nn} + \sigma^2 I) = \frac{\lambda_{\mathrm{max}}(K_{nn} + \sigma^2 I)}{\lambda_{\mathrm{min}}(K_{nn} + \sigma^2 I)}$$

Conjugate Gradients



Conjugate Gradients

Issue: redundant data creates rank deficiency in K_{xx}



Can we SGD in the era of deep learning?

Can we SGD in the era of deep learning?

- Can we cross the $\mathcal{O}(n^3)$ hurdle using SGD?

Can we SGD in the era of deep learning?

- Can we cross the $\mathcal{O}(n^3)$ hurdle using SGD?

- SGD needs -
  - Parametric view of model

Can we SGD in the era of deep learning?

- Can we cross the $\mathcal{O}(n^3)$ hurdle using SGD?

- SGD needs -
  - Parametric view of model
  - Unbiased mini-batch objective

Can we SGD in the era of deep learning?

- Can we cross the $\mathcal{O}(n^3)$ hurdle using SGD?

- SGD needs -
  - Parametric view of model
  - Unbiased mini-batch objective
  - Linear scaling with n

GPs can be parametric? Representer weights

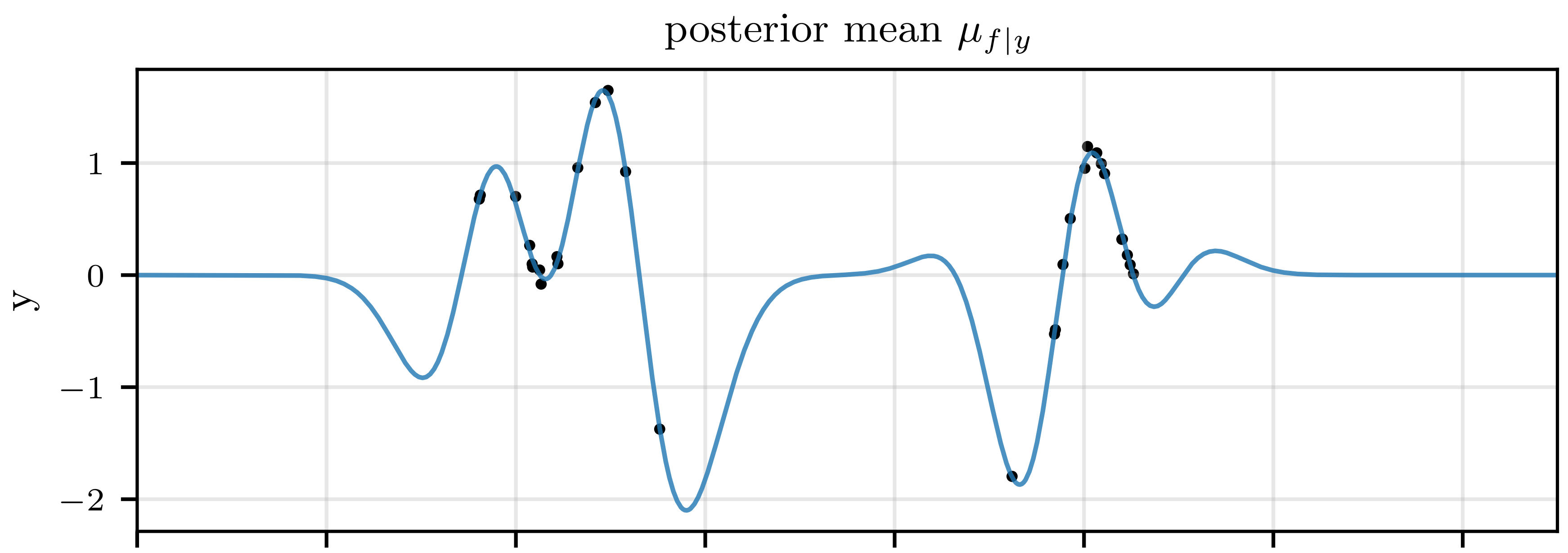
GPs can be parametric? Representer weights

- We have

GPs can be parametric? Representer weights

- We have

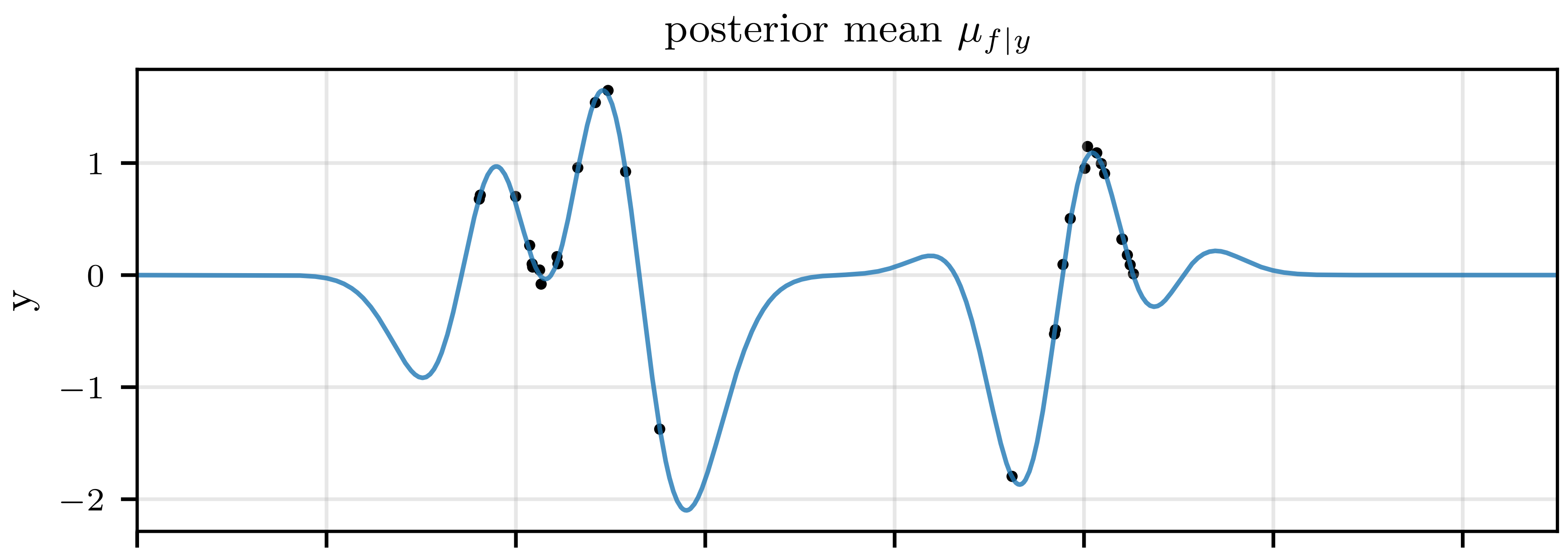
$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$



GPs can be parametric? Representer weights

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

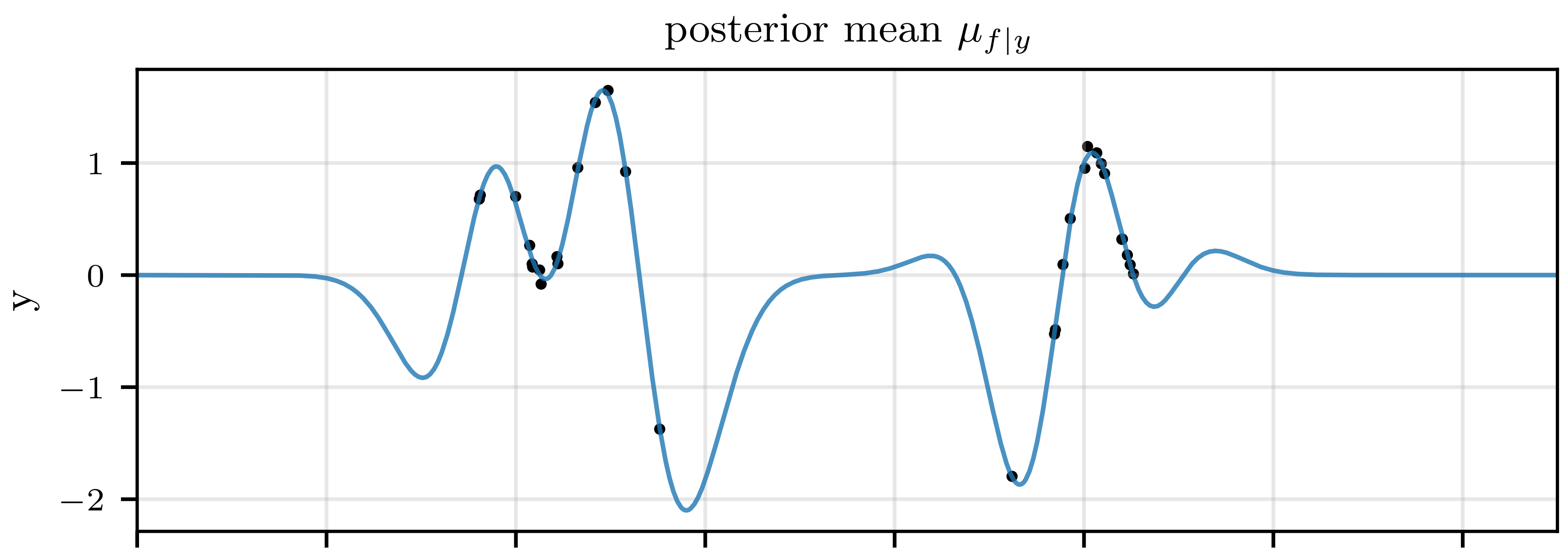


GPs can be parametric? Representer weights

- We have

$$\mu_{f|y}(X^*) = K_{*n} \left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n} \boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

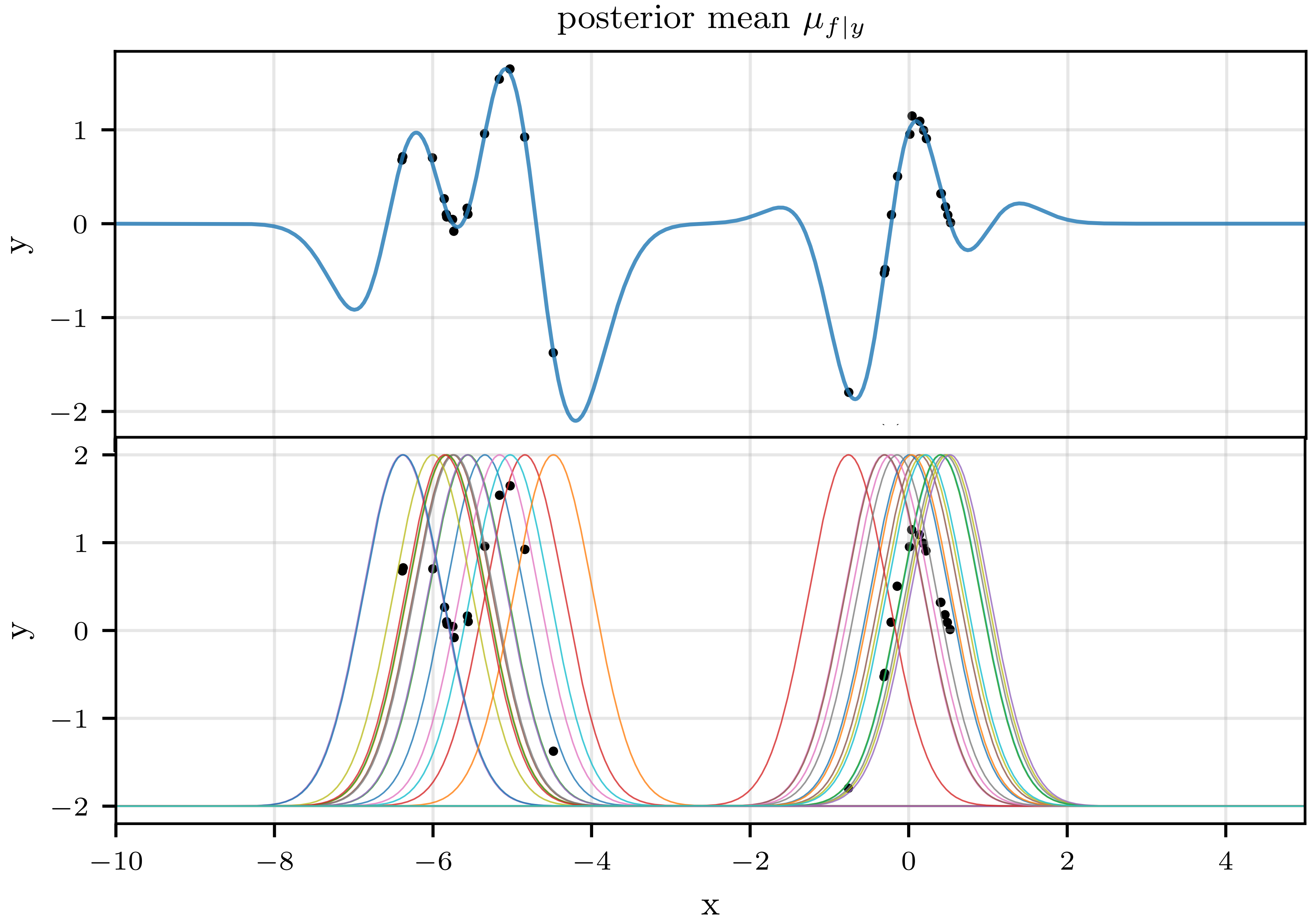


GPs can be parametric? Representer weights

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$



- Therefore, mean for a new test point

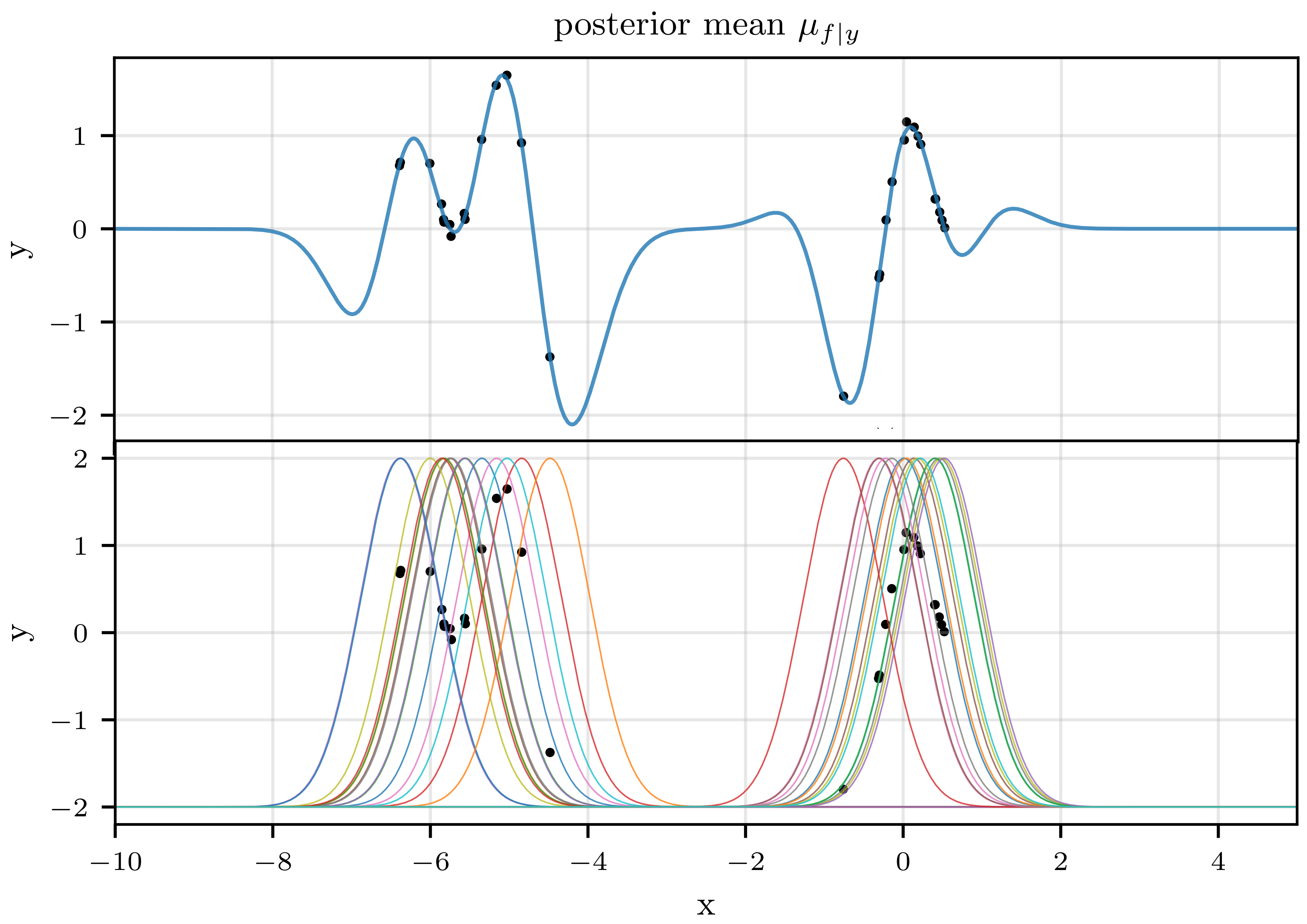
$$\mu_{f|y}(X^*) = v_1^* k(X^*, X_1) + v_2^* k(X^*, X_2) + \ldots + v_n^* k(X^*, X_n)$$

GPs can be parametric? Representer weights

- We have

$$\mu_{f|y}(X^*) = K_{*n} \left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n} \boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$



- Therefore, mean for a new test point

$$\mu_{f|y}(X^*) = v_1^* k(X^*, X_1) + v_2^* k(X^*, X_2) + \ldots + v_n^* k(X^*, X_n)$$

- $\boldsymbol{v}^* \in \mathbb{R}^n$ are **representer weights**

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y$$

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boxed{\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y}$$

n Linear System of Equations

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boxed{\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y}$$

n Linear System of Equations

Conjugate Gradients

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boxed{\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y}$$

n Linear System of Equations

Conjugate Gradients

Stochastic Gradient Descent

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boxed{\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y}$$

n Linear System of Equations

Conjugate Gradients

Stochastic Gradient Descent

$$\mathscr{L}(\mathbf{v}), \qquad \left.\frac{d\mathscr{L}(\mathbf{v})}{d\mathbf{v}}\right|_{\mathbf{v}=\mathbf{v}^*} = 0$$

I. Estimate the Mean of GPs

- We have

$$\mu_{f|y}(X^*) = K_{*n}\left(K_{nn} + \sigma^2 I\right)^{-1} y$$

$$\mu_{f|y}(X^*) = K_{*n}\boldsymbol{v}^* = \sum_{i=1}^{N} K_{*i} v_i^*$$

- Where

$$\boxed{\boldsymbol{v}^* = (K_{nn} + \sigma^2 I)^{-1} y}$$

n Linear System of Equations

Conjugate Gradients

$$\mathscr{L}(\mathbf{v}), \quad \left.\frac{d\mathscr{L}(\mathbf{v})}{d\mathbf{v}}\right|_{\mathbf{v}=\mathbf{v}^*} = 0$$

Stochastic Gradient Descent

$$\boldsymbol{v}^* = \underset{\boldsymbol{v}\in\mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$$

[1] Lin, JA*, Padhy, S.*, Antorán, J.,*, ... and Hernández-Lobato, J.M., 2022. Sampling from Gaussian Process Posteriors using Stochastic Gradient Descent. ***NeurIPS 2023***

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v}\in\mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in\mathbb{R}^N} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v}\in\mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v}\in\mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v} \in \mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

Easily minibatched

$$\frac{N}{B} \sum_{i}^{B} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$$\mathbf{v}^\top K_{nn} \mathbf{v}$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v} \in \mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

Easily minibatched

$$\frac{N}{B} \sum_{i}^{B} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$\mathcal{O}(n^2)$ space

$\mathbf{v}^\top K_{nn} \mathbf{v}$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in\mathbb{R}^N} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|_{K_{nn}}^2$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$\mathcal{O}(n^2)$ space

$$\mathbf{v}^\top K_{nn}\mathbf{v}$$

$$K_{nn} \approx \Phi(x)\Phi(x)^T, \quad \Phi(x) \in \mathbb{R}^{n,L}$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \underset{\boldsymbol{v}\in\mathbb{R}^N}{\arg\min} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|^2_{K_{nn}}$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$\mathcal{O}(n^2)$ space

$$\mathbf{v}^\top K_{nn}\mathbf{v}$$

$$K_{nn} \approx \Phi(x)\Phi(x)^T, \quad \Phi(x) \in \mathbb{R}^{n,L}$$

$$\mathbf{v}^\top K_{nn}\mathbf{v} \approx \sum_{\ell=1}^{L}\left(\boldsymbol{v}^T\phi_\ell(x)\right)^2$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in\mathbb{R}^N} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|^2_{K_{nn}}$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$\mathcal{O}(n^2)$ space

$$\mathbf{v}^\top K_{nn} \mathbf{v}$$

$$K_{nn} \approx \Phi(x)\Phi(x)^T, \quad \Phi(x) \in \mathbb{R}^{n,L}$$

$$\mathbf{v}^\top K_{nn}\mathbf{v} \approx \sum_{\ell=1}^{L} \left(\boldsymbol{v}^T \phi_\ell(x)\right)^2$$

$$\frac{N}{B}\sum_{i}^{B} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \sum_{\ell=1}^{L} \left(\boldsymbol{v}^T \phi_\ell(x)\right)^2$$

I. Estimate the Mean of GPs

- We have $\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in\mathbb{R}^N} \sum_{i=1}^{N} \frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \|\boldsymbol{v}\|^2_{K_{nn}}$

Easily minibatched

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2}$$

$\mathcal{O}(n^2)$ space

$$\mathbf{v}^\top K_{nn}\mathbf{v}$$

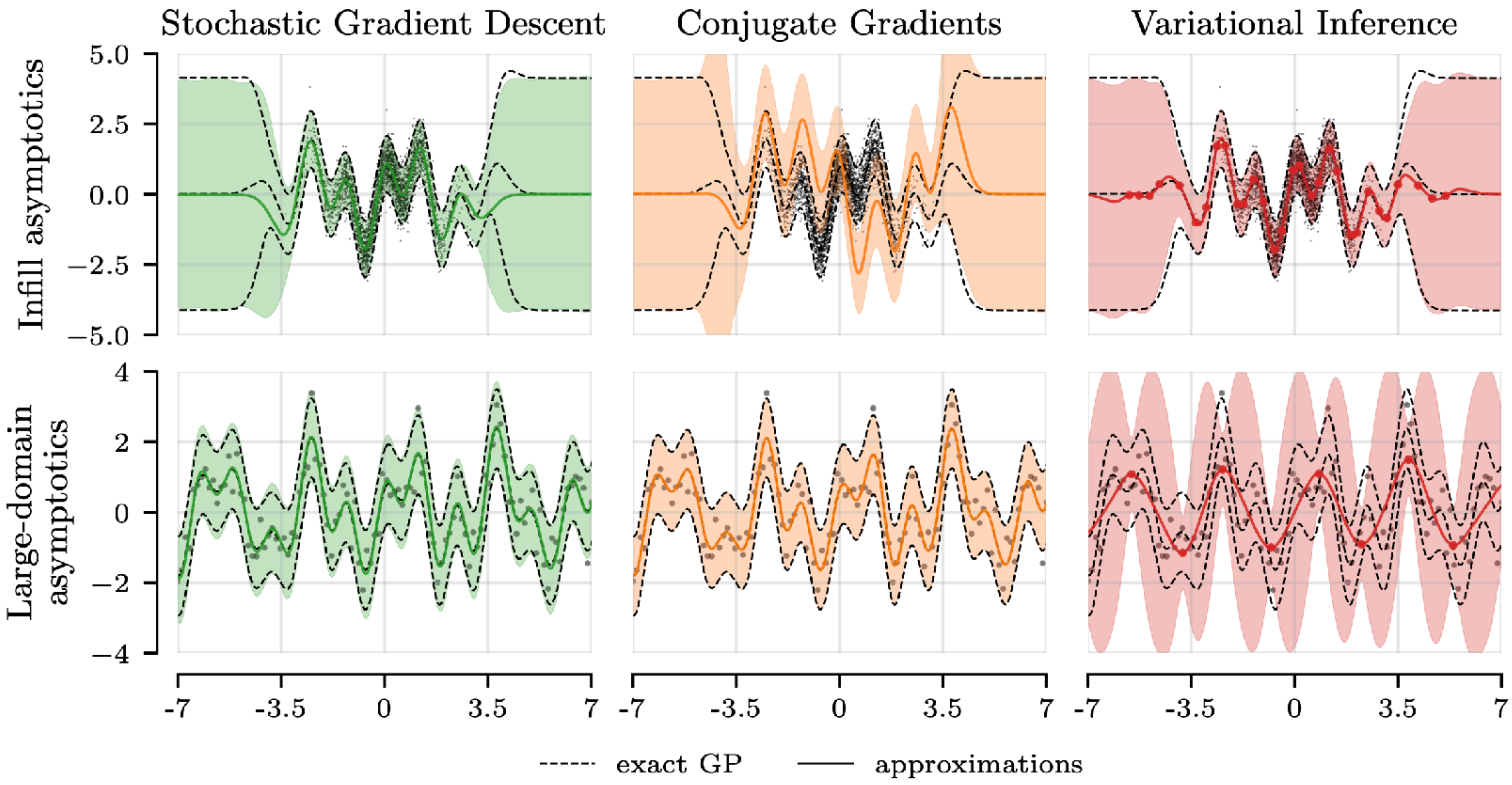
$$K_{nn} \approx \Phi(x)\Phi(x)^T, \quad \Phi(x) \in \mathbb{R}^{n,L}$$

$$\mathbf{v}^\top K_{nn}\mathbf{v} \approx \sum_{\ell=1}^{L}\left(\boldsymbol{v}^T\phi_\ell(x)\right)^2$$

$\mathcal{O}(n)$

$$\frac{N}{B}\sum_{i}^{B}\frac{\left(y_i - K_{x_i,n}\boldsymbol{v}\right)^2}{\sigma^2} + \sum_{\ell=1}^{L}\left(\boldsymbol{v}^T\phi_\ell(x)\right)^2$$

SGD works better on most cases

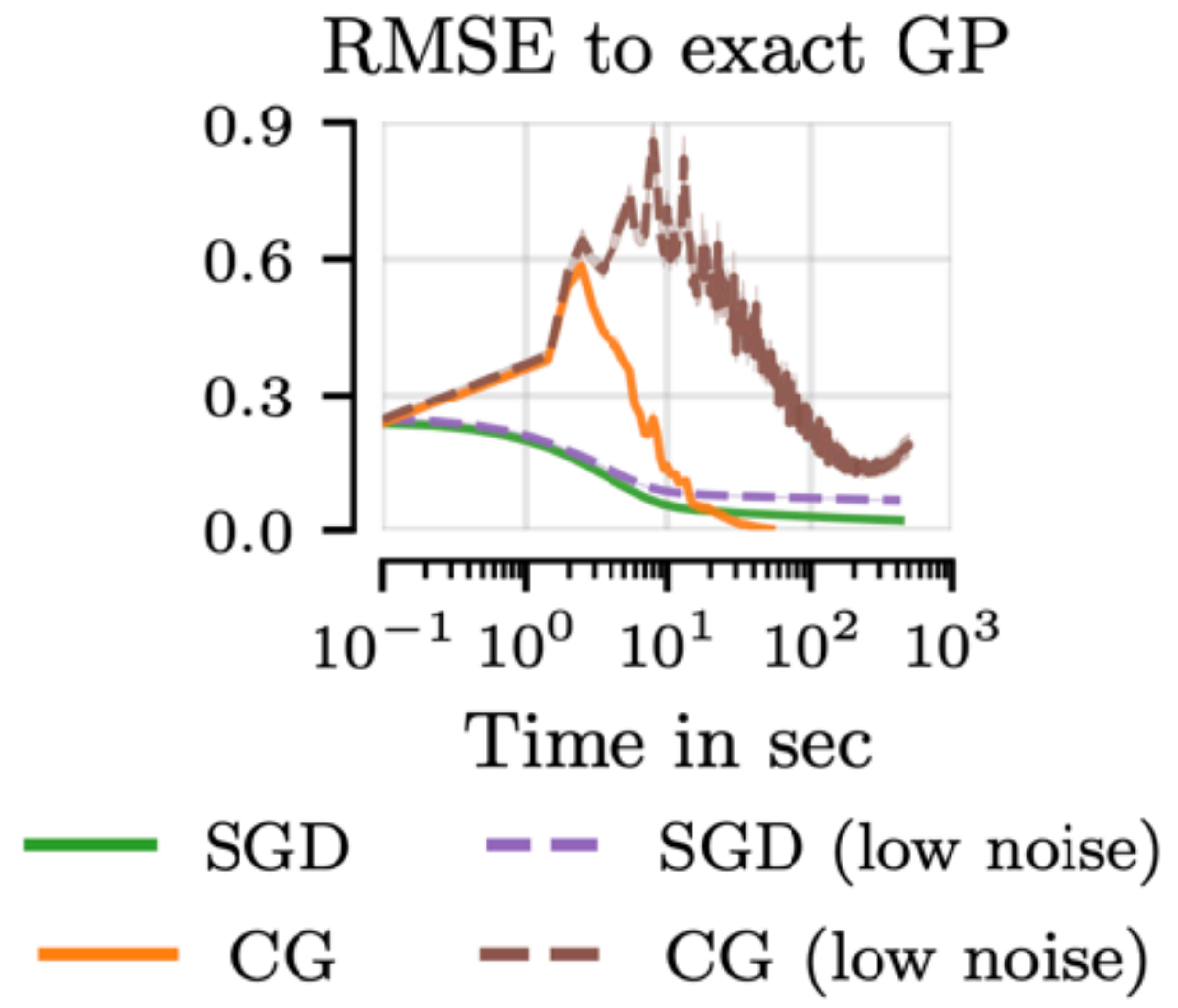


SGD scales much better than CG

- CG has non-monotonic convergence guarantee in $\mathcal{O}\left(\sqrt{\mathrm{cond}(K_{nn} + \sigma^2 I)}\log\frac{\mathrm{cond}(K_{nn} + \sigma^2)\|y\|}{\varepsilon}\right)$ steps
- SGD monotonically converges (to approx. soln), has no dependence on conditioning!

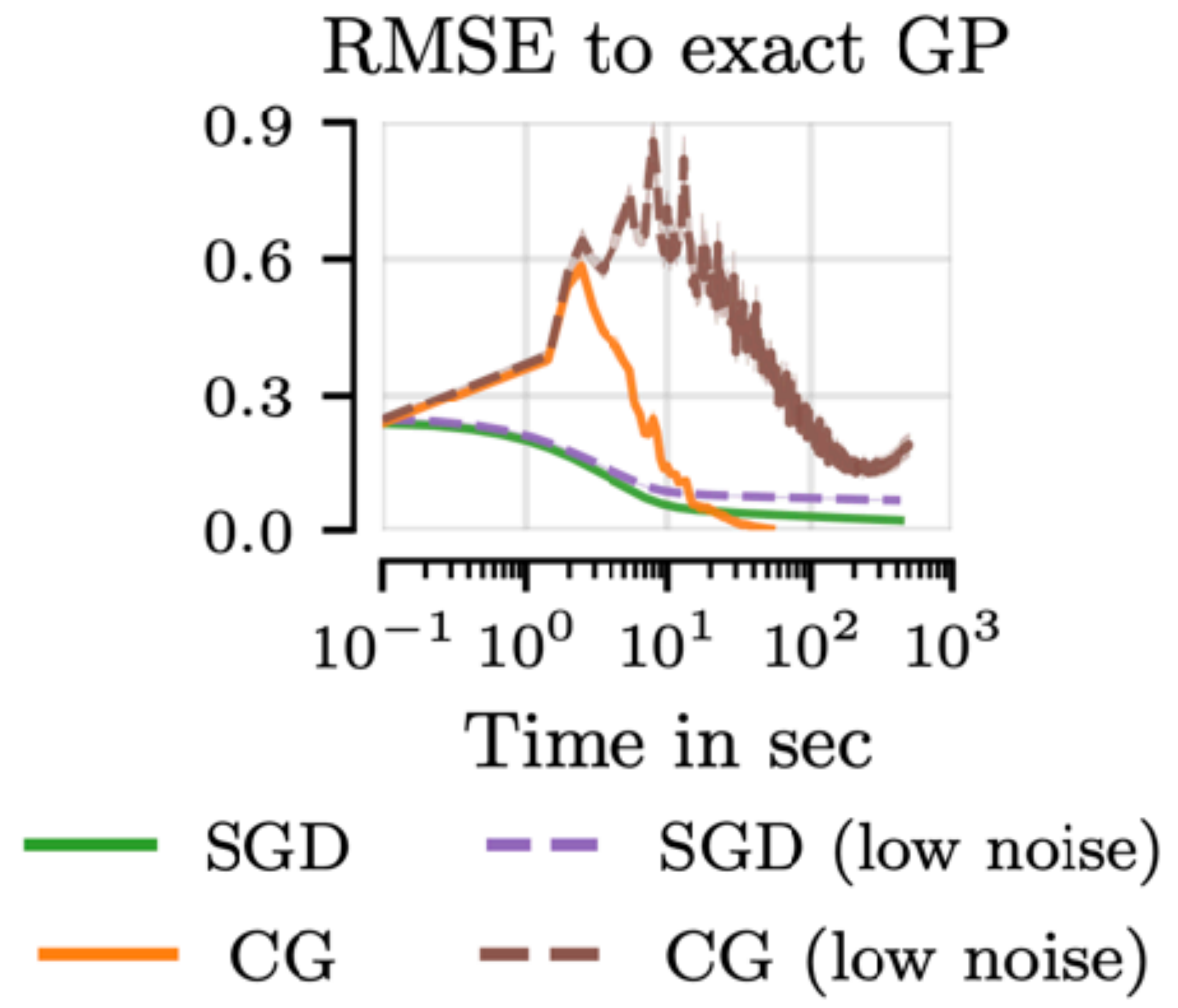
SGD scales much better than CG

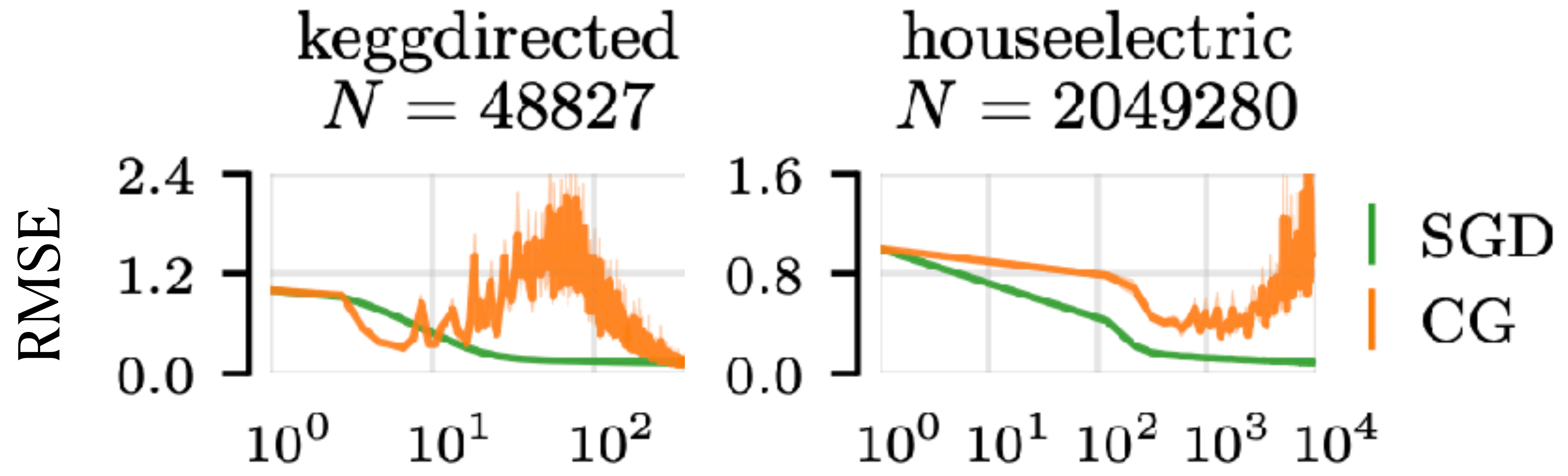
- CG has non-monotonic convergence guarantee in $\mathcal{O}\left(\sqrt{\text{cond}(K_{nn}+\sigma^2 I)}\log\frac{\text{cond}(K_{nn}+\sigma^2)\|y\|}{\varepsilon}\right)$ steps
- SGD monotonically converges (to approx. soln), has no dependence on conditioning!



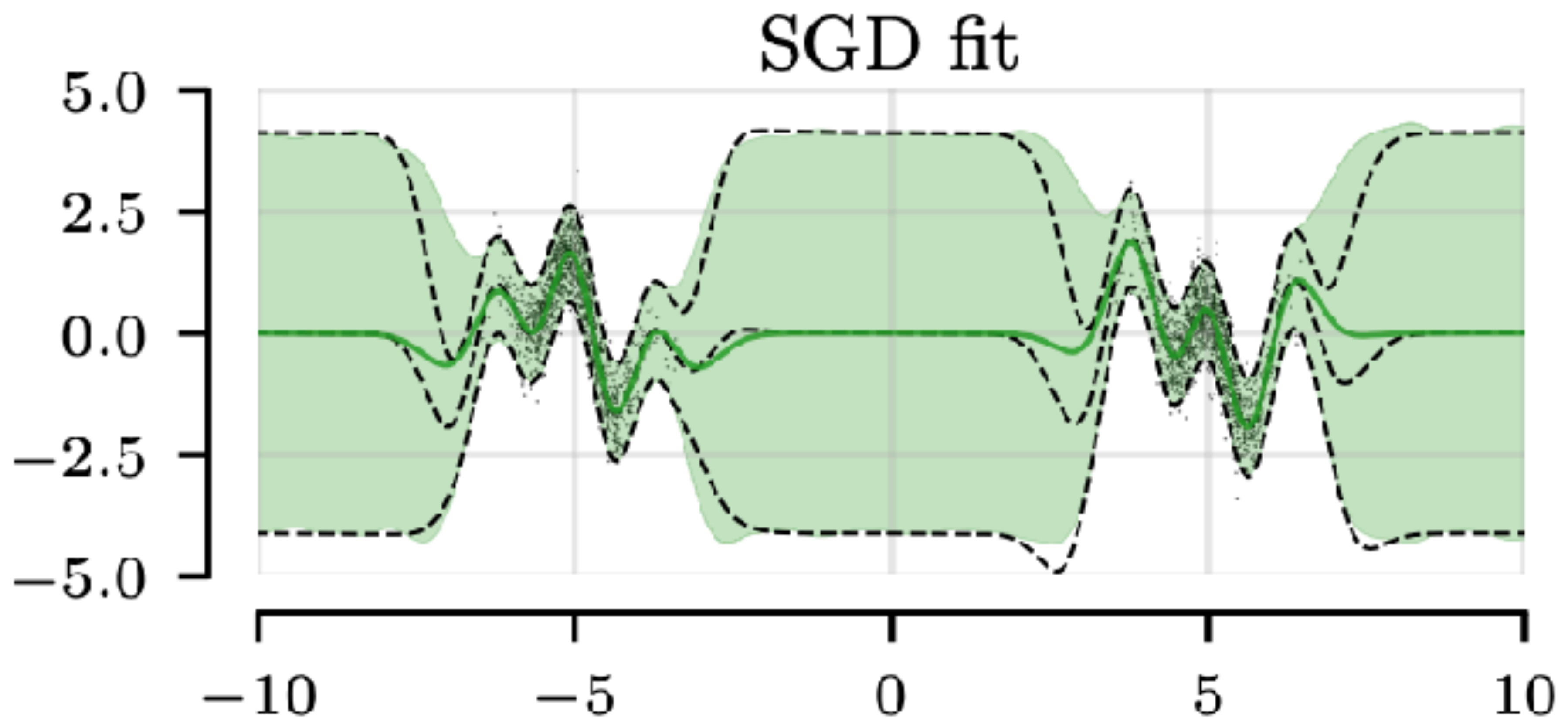
SGD scales much better than CG

- CG has non-monotonic convergence guarantee in $\mathcal{O}\left(\sqrt{\mathrm{cond}(K_{nn} + \sigma^2 I)} \log \frac{\mathrm{cond}(K_{nn} + \sigma^2)\|y\|}{\varepsilon}\right)$ steps
- SGD monotonically converges (to approx. soln), has no dependence on conditioning!

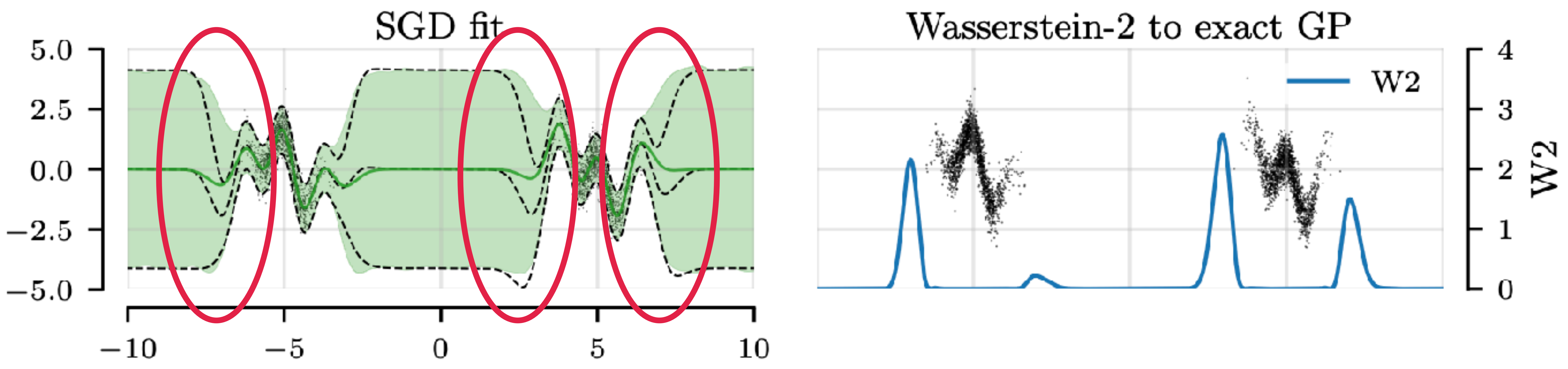




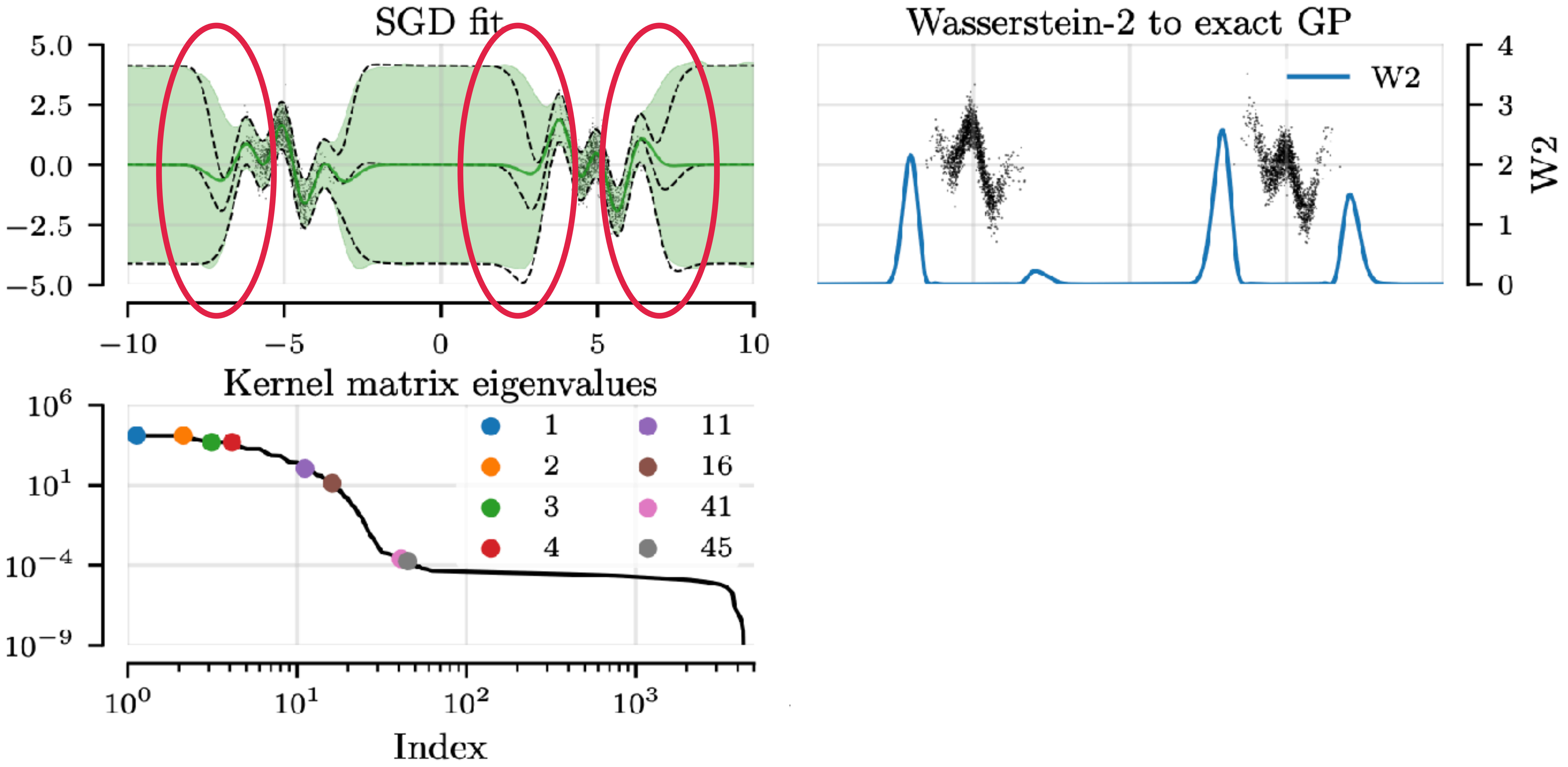
Spectral Analysis of SGD Behaviour



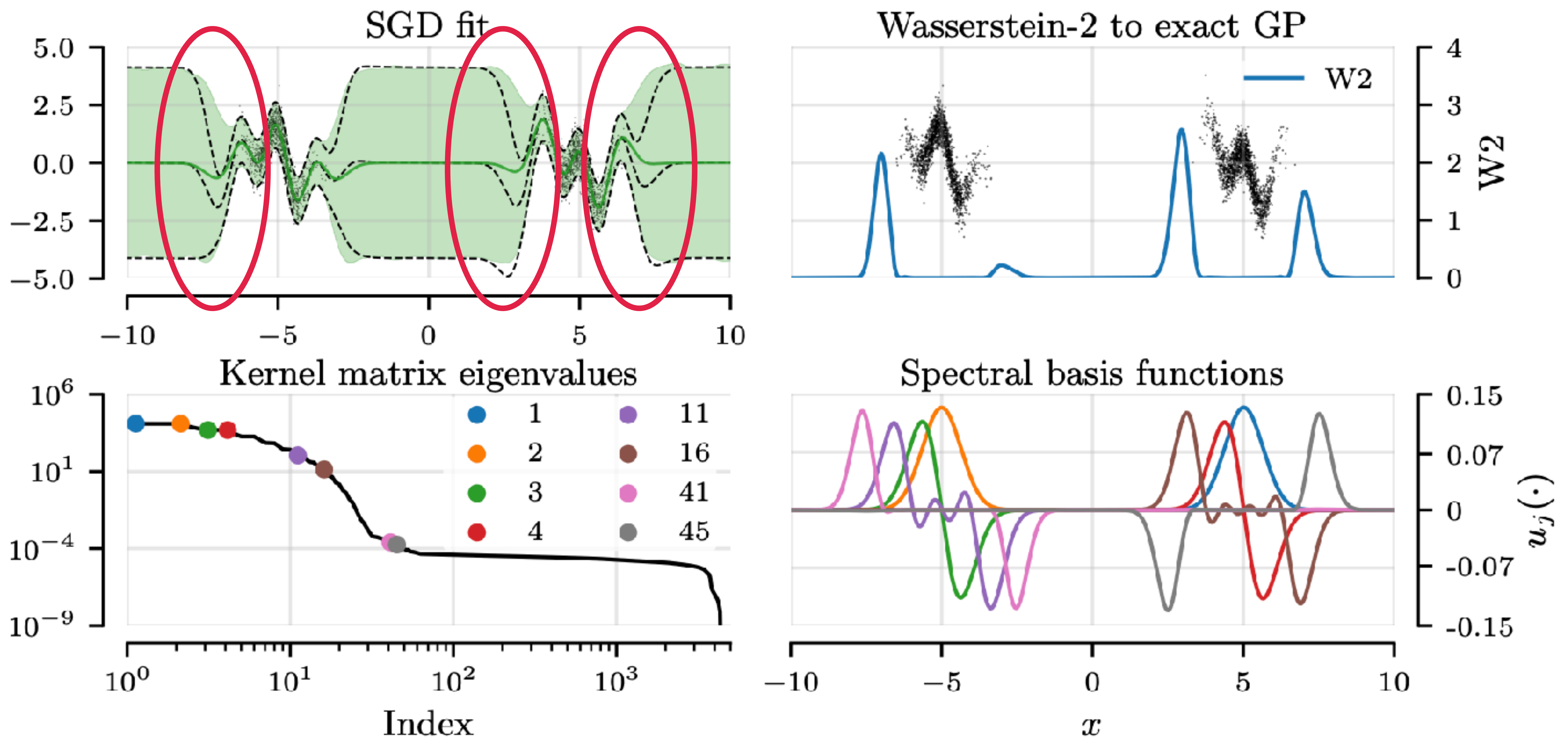
Spectral Analysis of SGD Behaviour



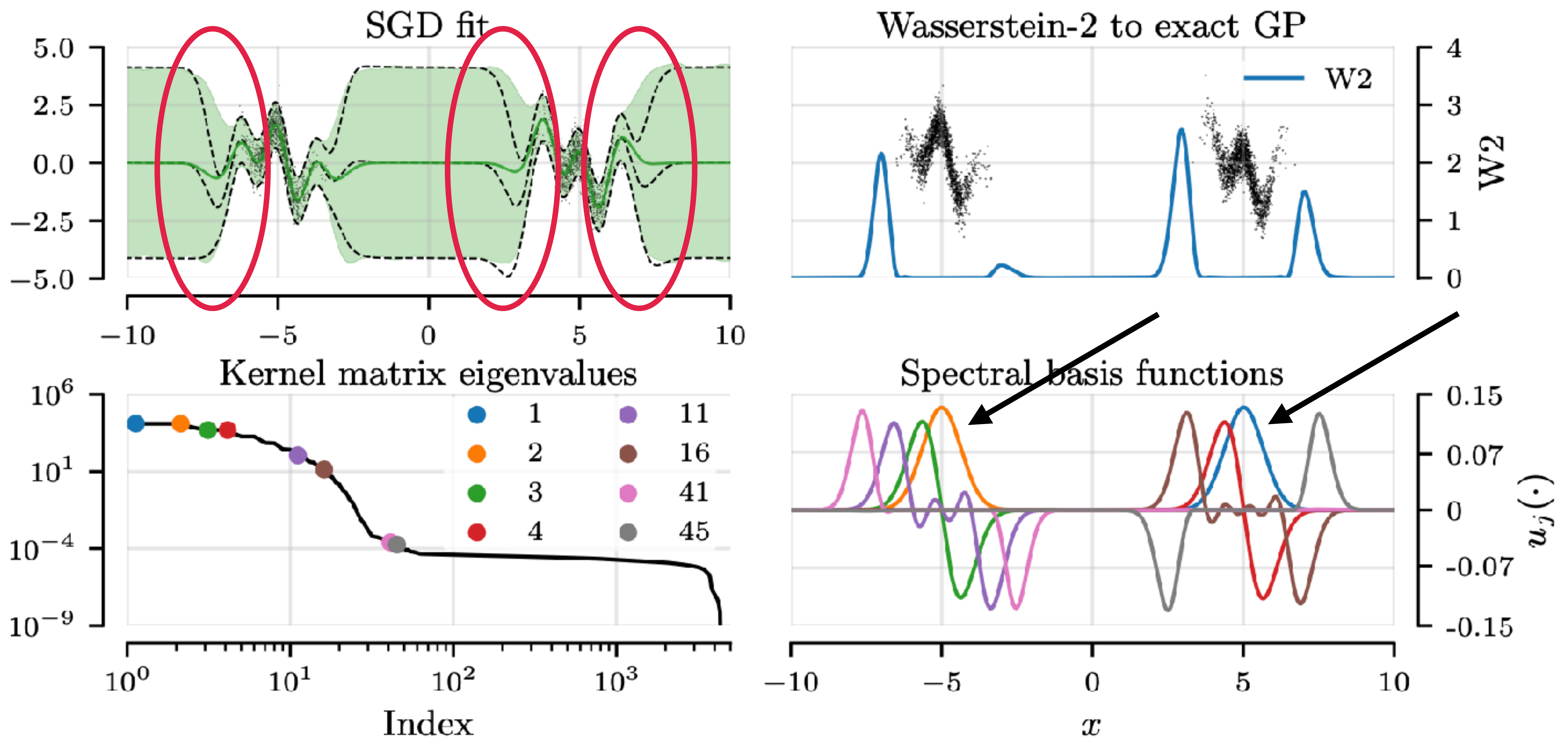
Spectral Analysis of SGD Behaviour



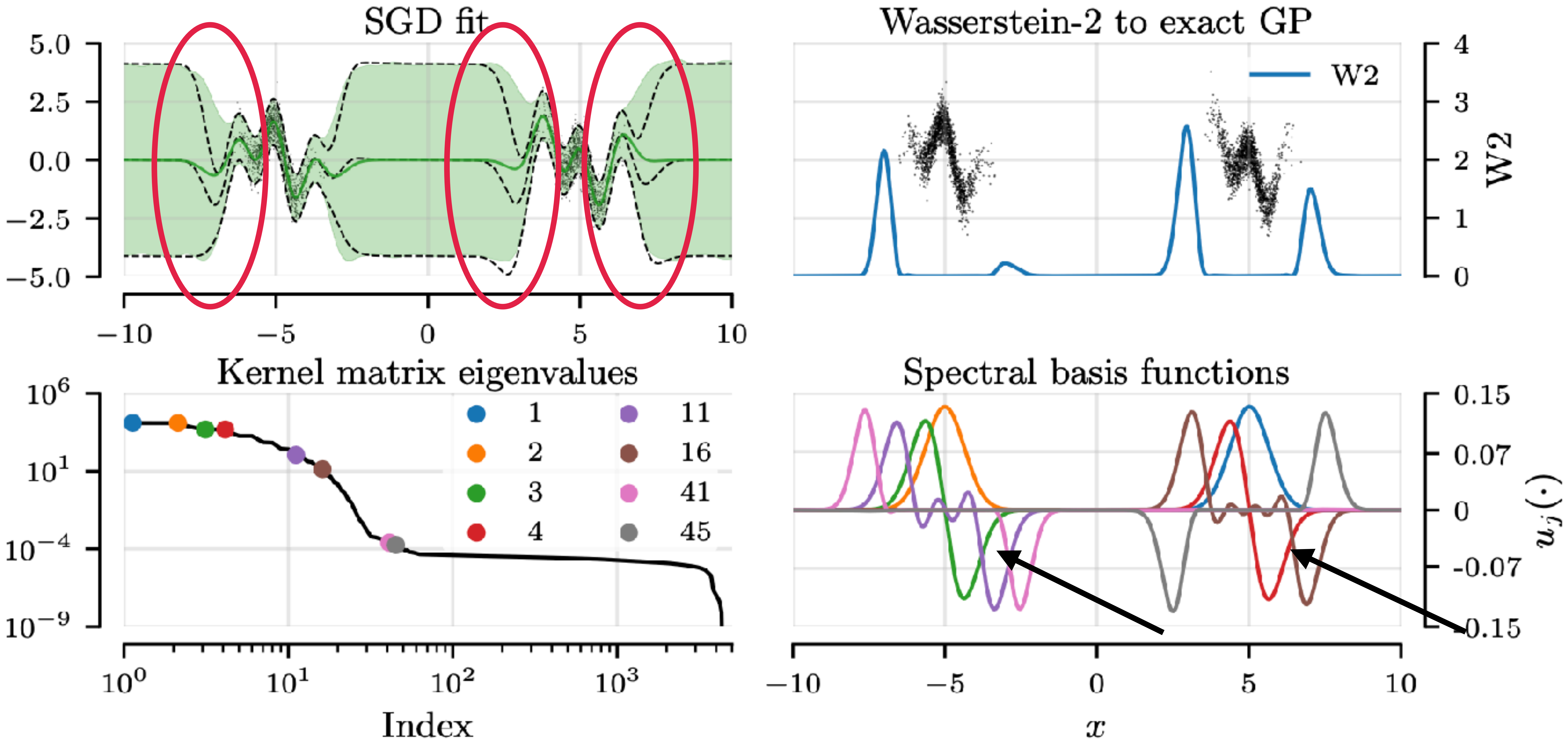
Spectral Analysis of SGD Behaviour



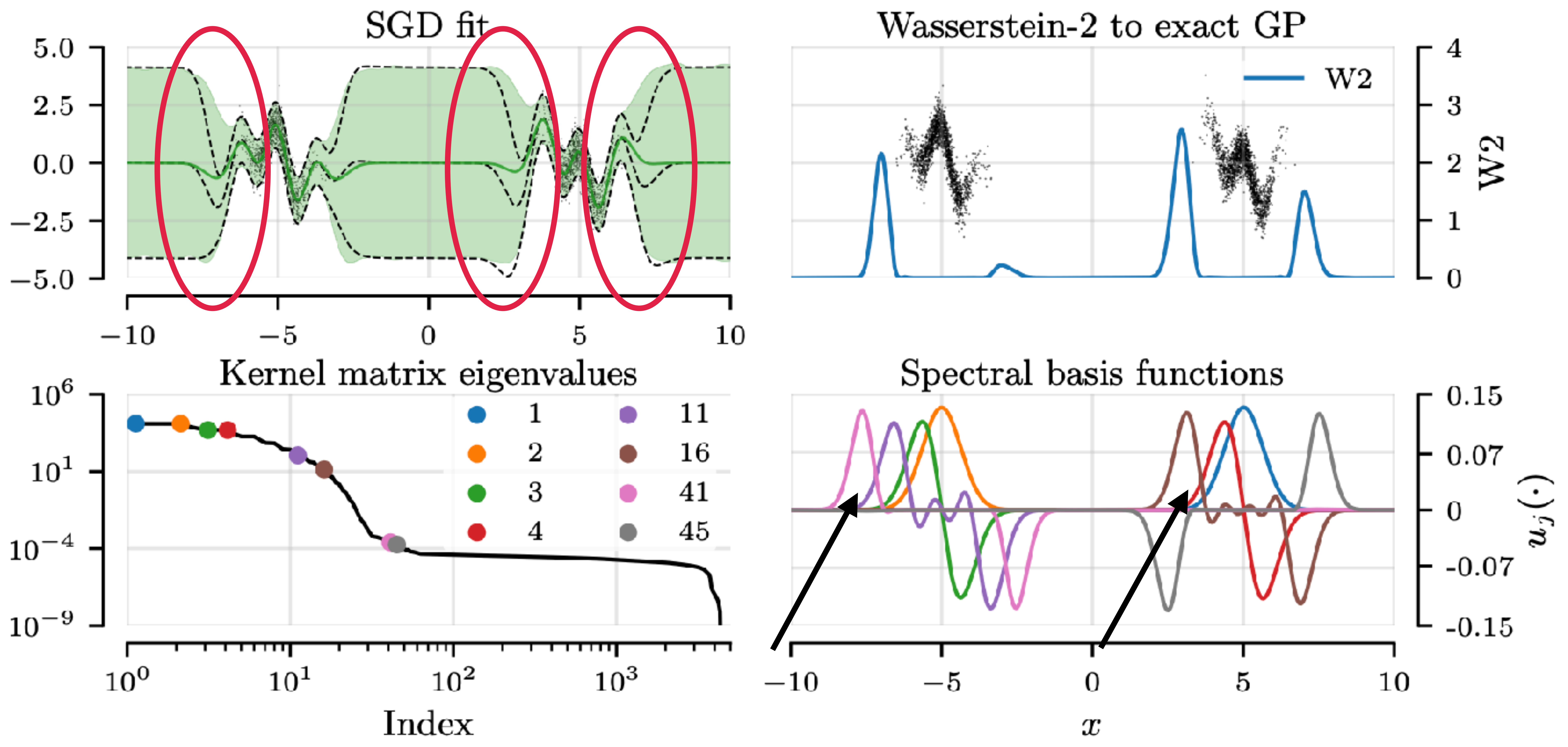
Spectral Analysis of SGD Behaviour



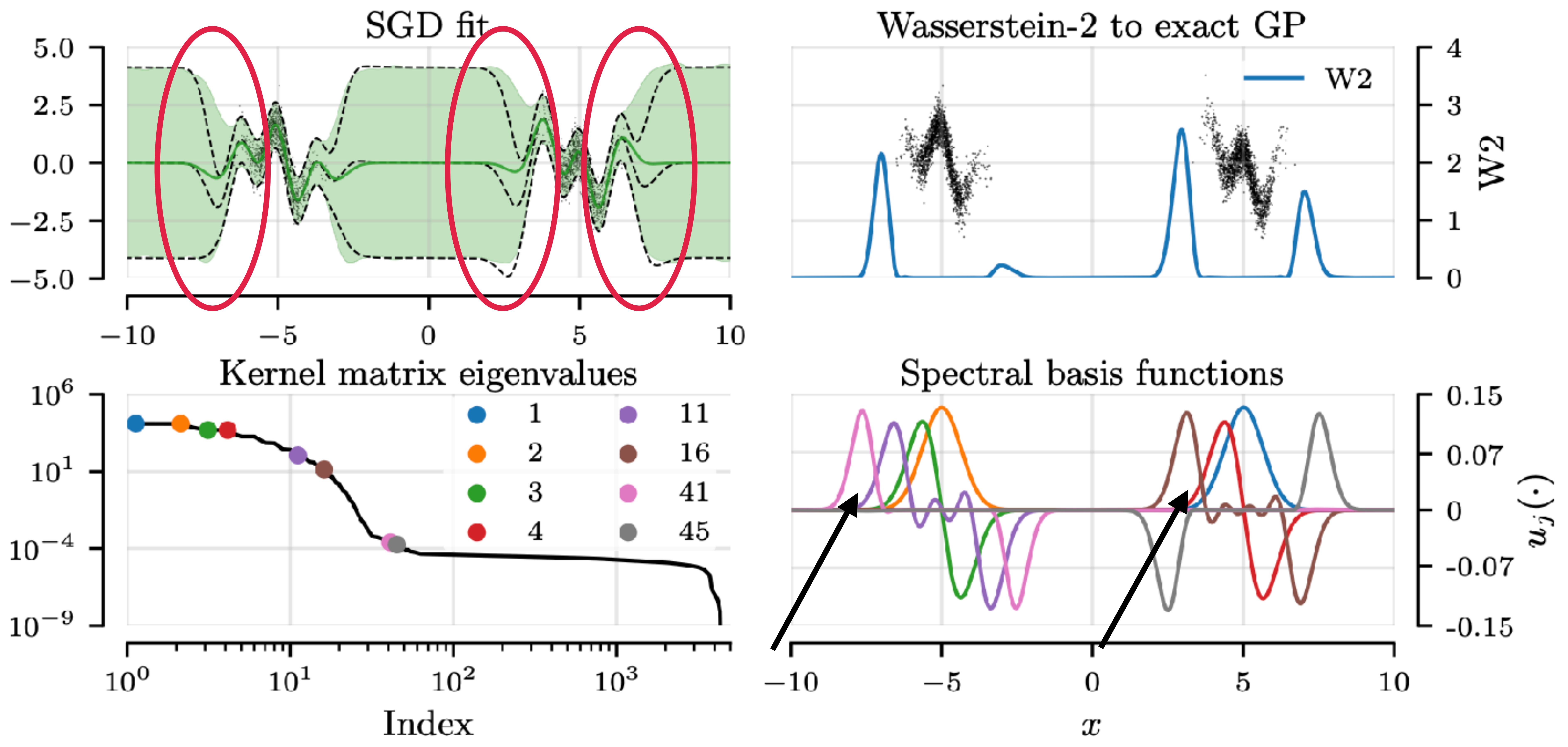
Spectral Analysis of SGD Behaviour



Spectral Analysis of SGD Behaviour



Spectral Analysis of SGD Behaviour



$$\left\| \operatorname{proj}_{u_i} \mu_{f|\mathbf{y}} - \operatorname{proj}_{u_i} \mu_{\mathrm{SGD}} \right\|_{H_k} \leq \frac{4G+1}{\eta} \sqrt{\frac{\log \frac{N}{\delta}}{t\lambda_i}}$$

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix...

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?
    - Option 1: **Cholesky decomposition**

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?
    - Option 1: **Cholesky decomposition**
        1. Decompose $\Sigma = LL^T$

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?
  - Option 1: **Cholesky decomposition**
    1. Decompose $\Sigma = LL^T$
    2. Draw sample from unit Gaussian, $\epsilon \sim \mathcal{N}(0,I)$

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?
  - Option 1: **Cholesky decomposition**
    1. Decompose $\Sigma = LL^T$
    2. Draw sample from unit Gaussian, $\epsilon \sim \mathcal{N}(0,I)$
    3. Sample from posterior is $\mu_{f|y} + L\epsilon$

Can we estimate the uncertainties with SGD?

$$\Sigma_{f|y} = K_{**} - K_{*n}^{\top}(K_{nn} + \sigma^2 I)^{-1} K_{n*}$$

- No, because we can't solve an inverse with a matrix…

- Can we at least draw samples from the posterior $\mathcal{N}\left(\mu_{f|y}, \Sigma_{f|y}\right)$?
  - Option 1: **Cholesky decomposition**
    1. Decompose $\Sigma = LL^T$
    2. Draw sample from unit Gaussian, $\epsilon \sim \mathcal{N}(0,I)$
    3. Sample from posterior is $\mu_{f|y} + L\epsilon$
  - **Can we do better?**

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

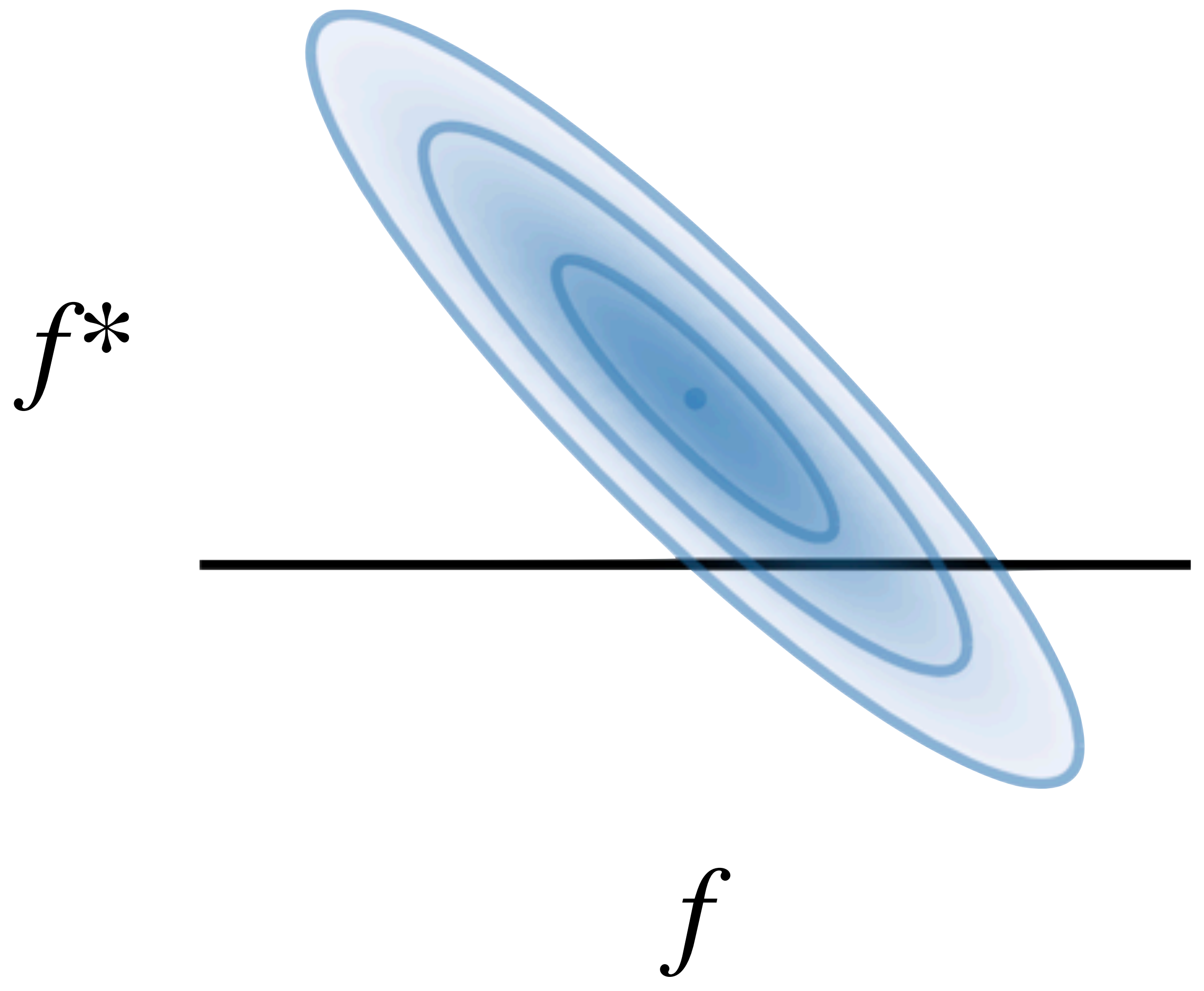
$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$

[1] Wilson, J.T., Borovitskiy, V., Terenin, A., Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, *22*(1), pp.4741-4787.

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$

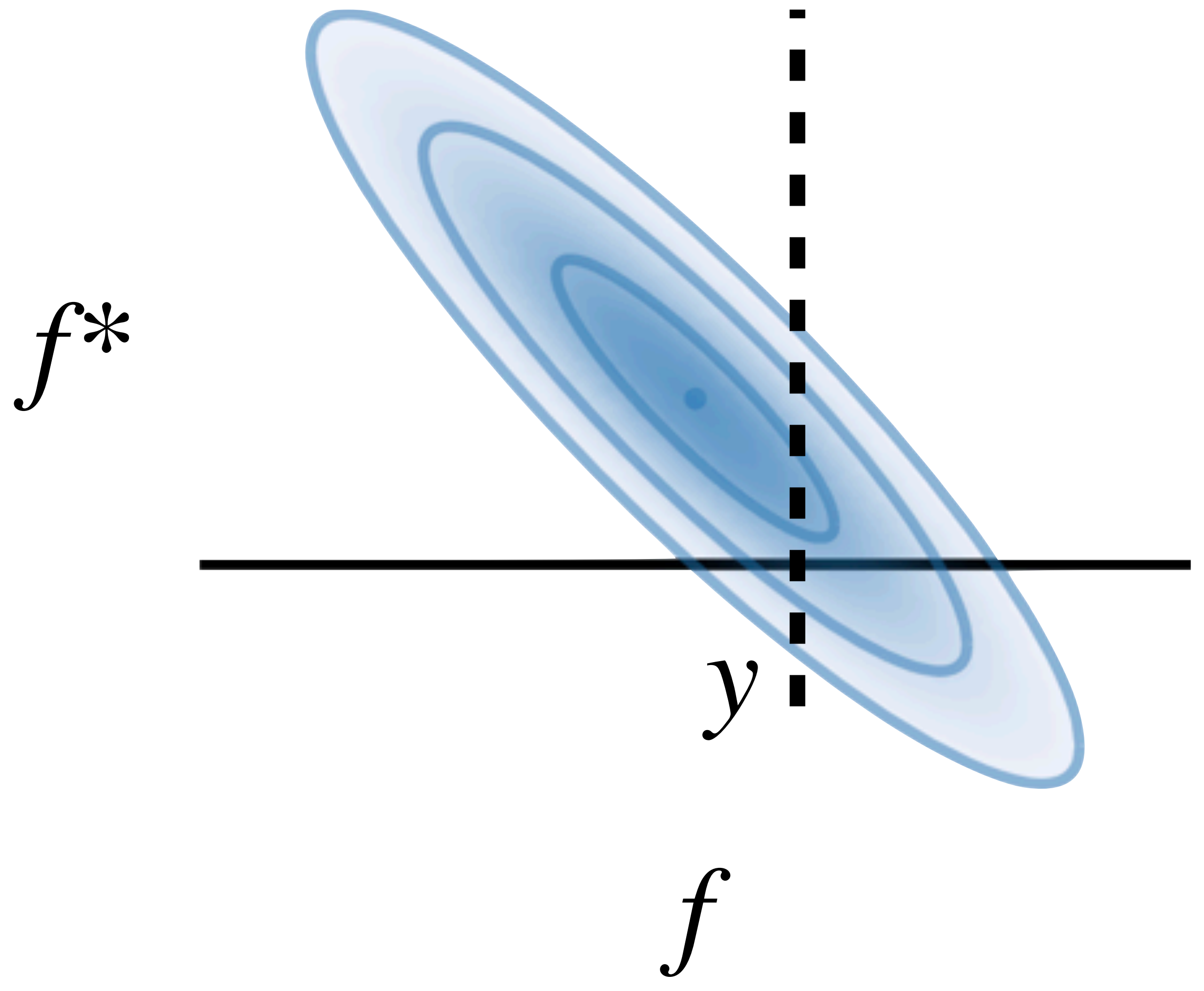


[1] Wilson, J.T., Borovitskiy, V., <u>Terenin, A.</u>, Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, *22*(1), pp.4741-4787.

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$

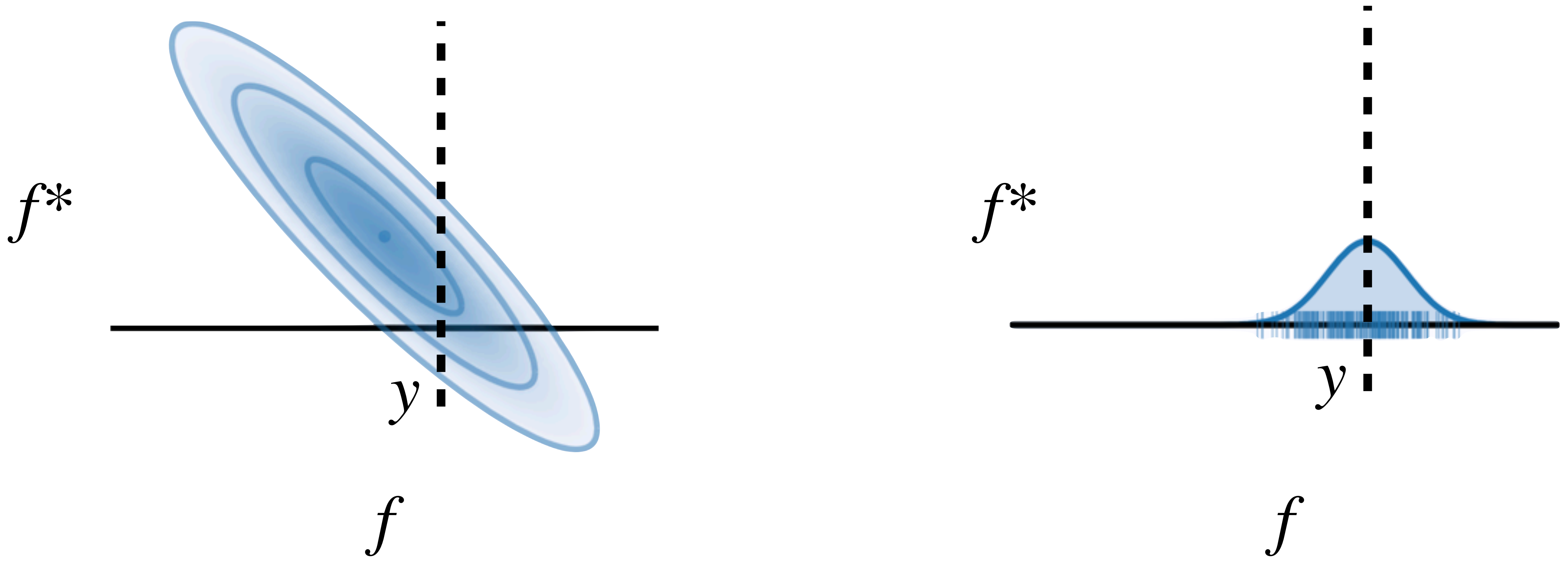


[1] Wilson, J.T., Borovitskiy, V., Terenin, A., Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, *22*(1), pp.4741-4787.

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$



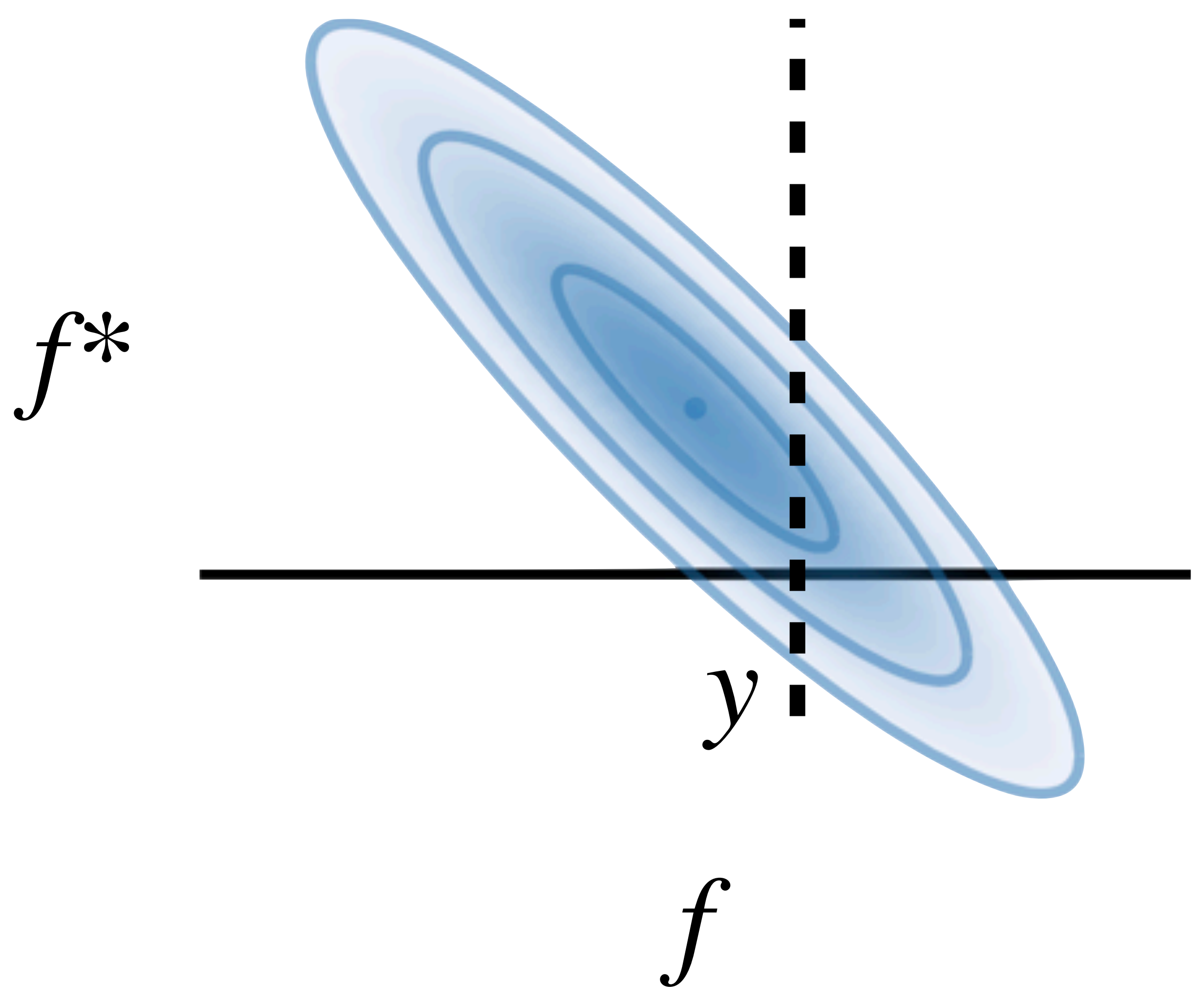
$$p(f^* \mid f = y) = \mathcal{N}(K_{*n}K_{nn}^{-1}y, K_{**} - K_{*n}K_{nn}^{-1}K_{n*})$$

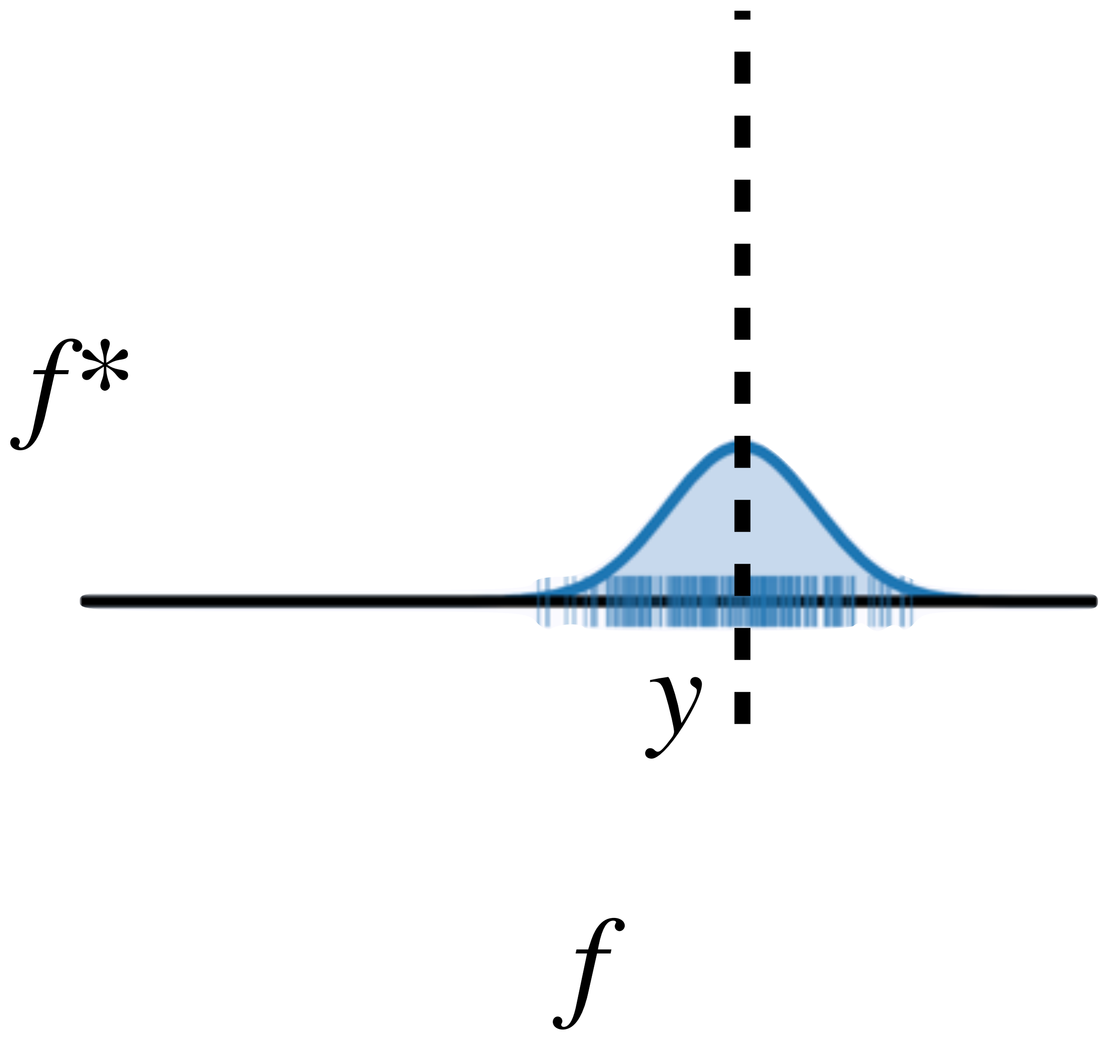
[1] Wilson, J.T., Borovitskiy, V., Terenin, A., Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, 22(1), pp.4741-4787.

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$





Distributional View

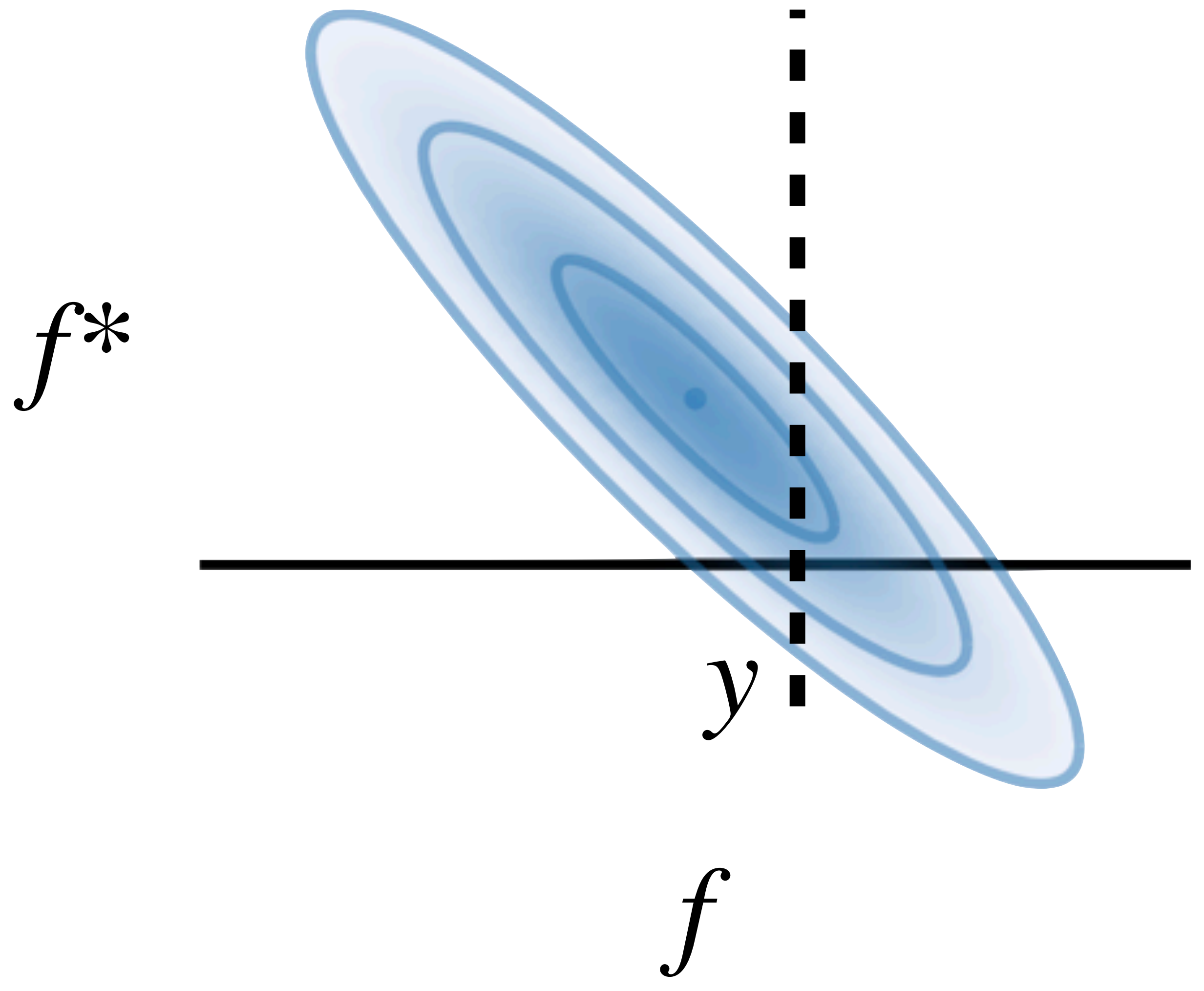
$$p(f^* \mid f = y) = \mathcal{N}(K_{*n}K_{nn}^{-1}y, K_{**} - K_{*n}K_{nn}^{-1}K_{n*})$$

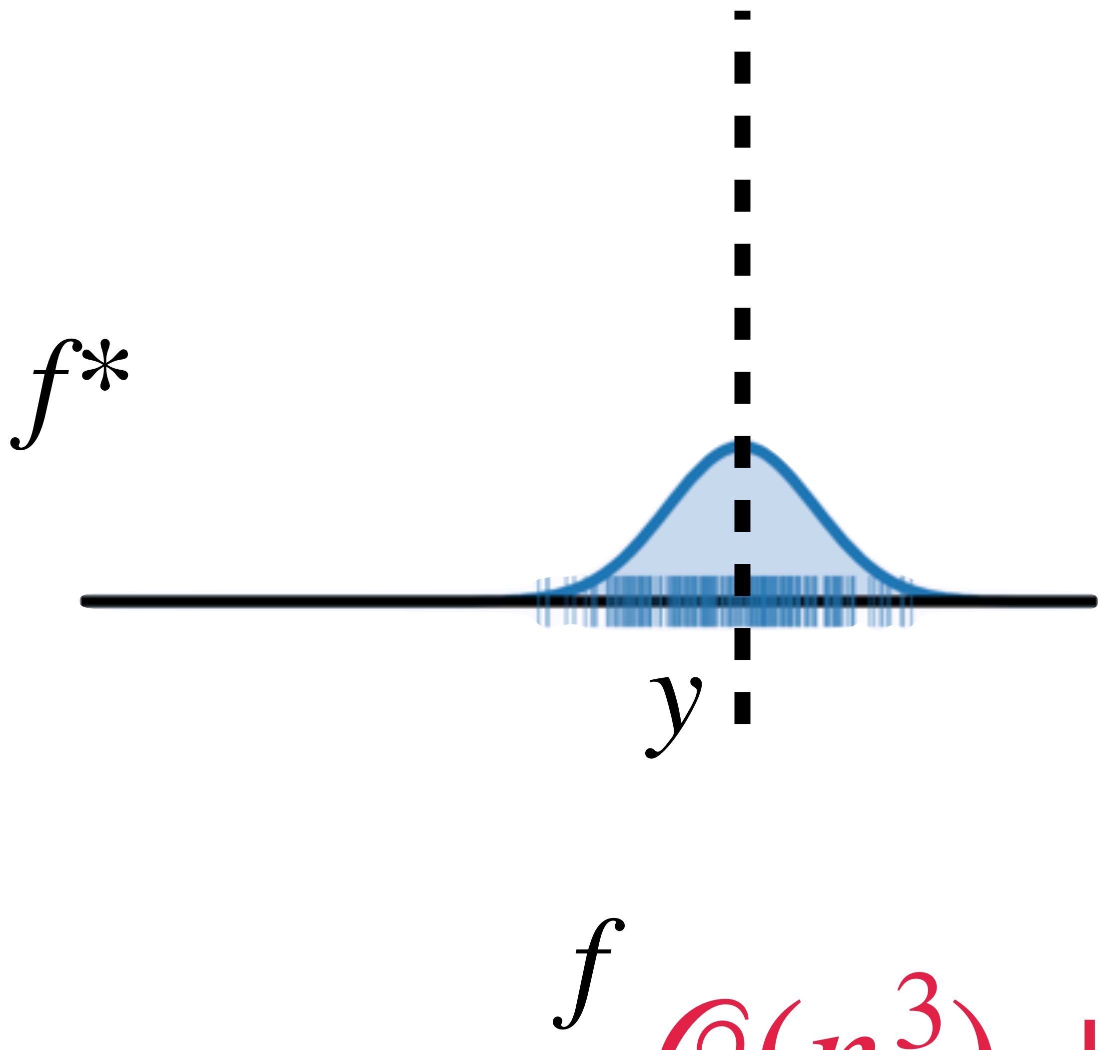
[1] Wilson, J.T., Borovitskiy, V., Terenin, A., Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, 22(1), pp.4741-4787.

A Path to More Efficient Sampling [1]

- Let us take another look at multivariate Gaussian distributions (ignore noise)

$$\left[\begin{pmatrix} f(X_*) \\ f(X) \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} k(X_*, X_*) & k(X_*, X) \\ k(X_*, X)^\top & k(X, X) \end{pmatrix}\right]\right)$$





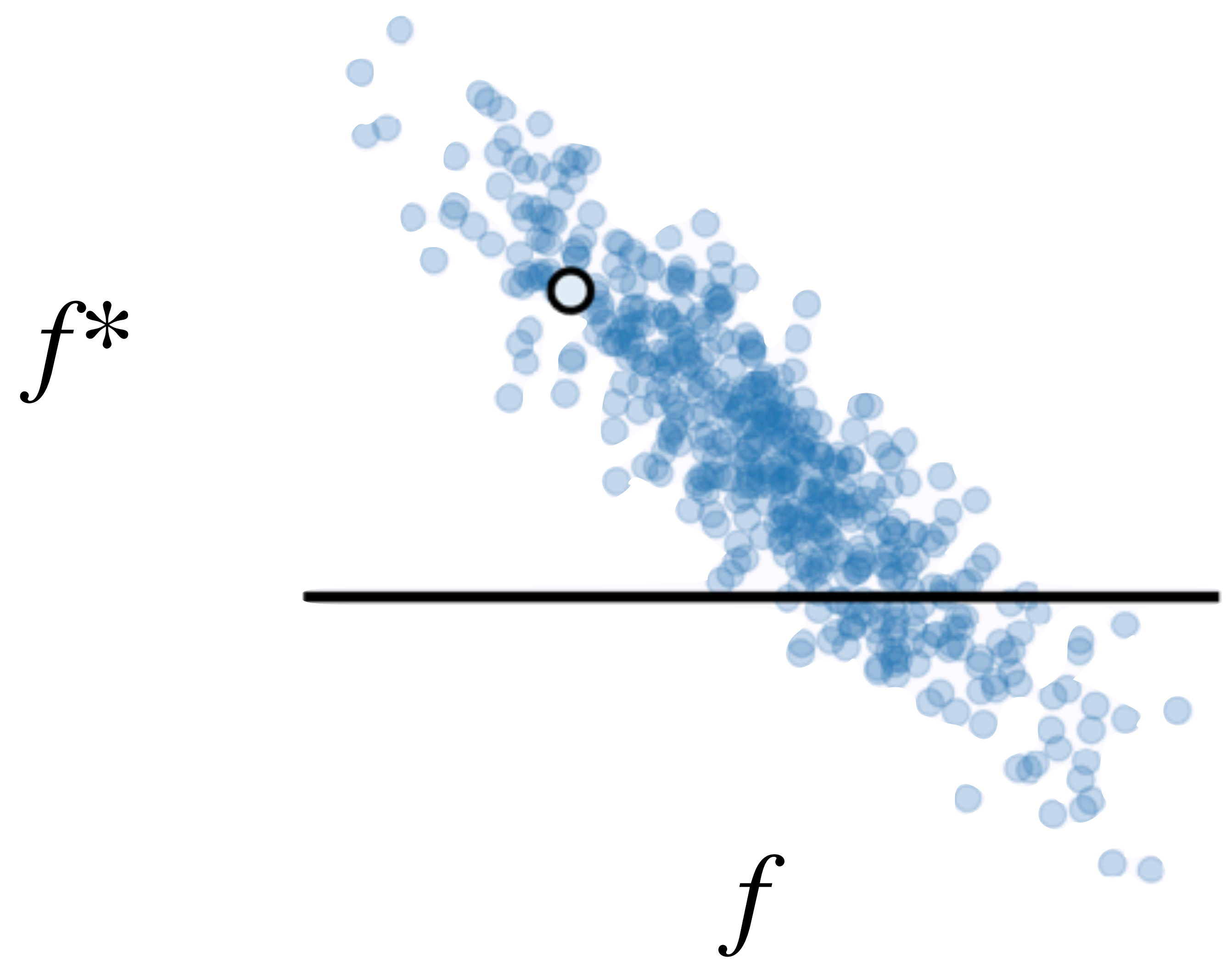
Distributional View

$\mathcal{O}(n^3) + \mathcal{O}(n^{*3})$

$$p(f^* \mid f = y) = \mathcal{N}(K_{*n}K_{nn}^{-1}y, K_{**} - K_{*n}K_{nn}^{-1}K_{n*})$$

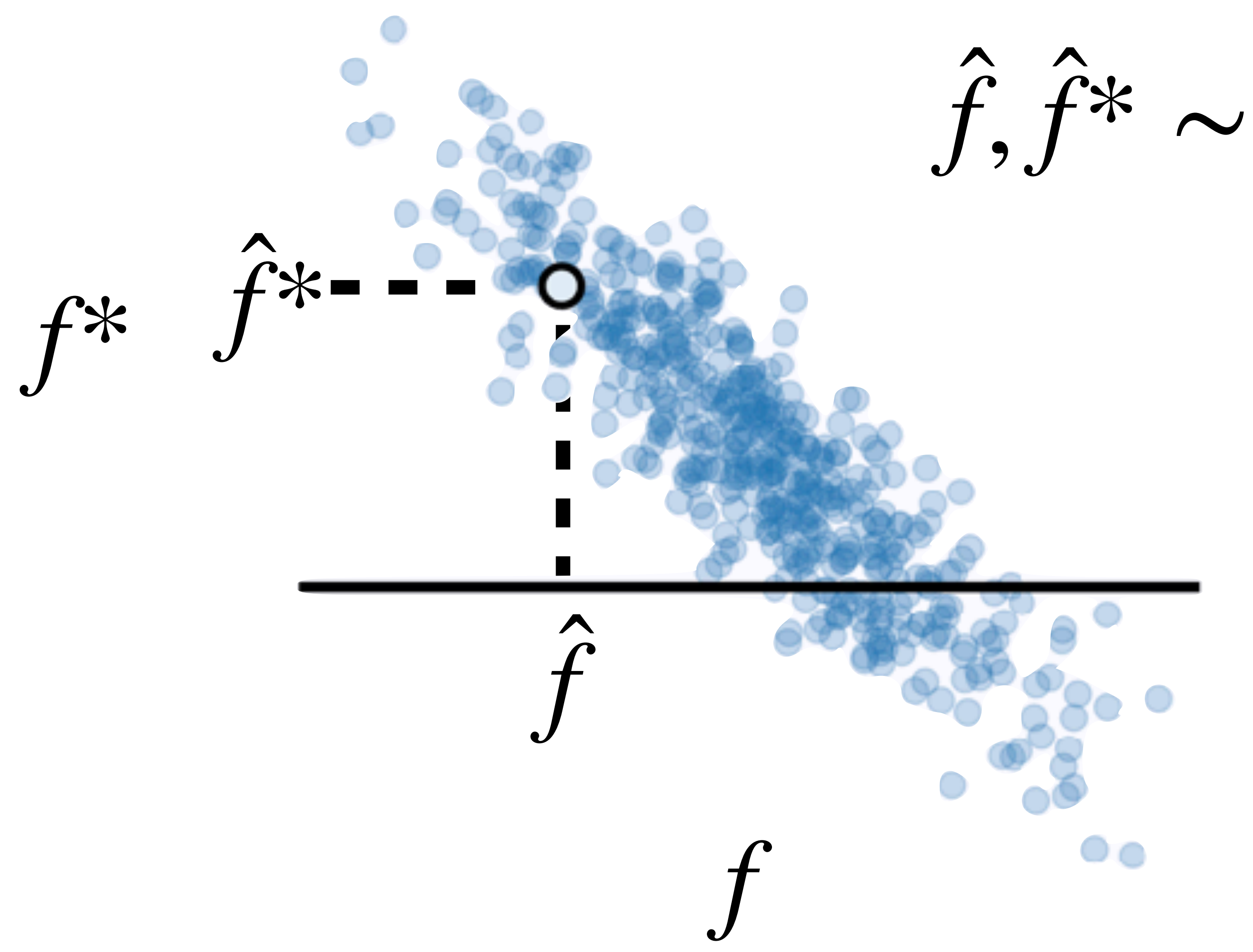
[1] Wilson, J.T., Borovitskiy, V., Terenin, A., Mostowsky, P. and Deisenroth, M.P., 2021. Pathwise conditioning of gaussian processes. *The Journal of Machine Learning Research*, 22(1), pp.4741-4787.

Sample from the Posterior

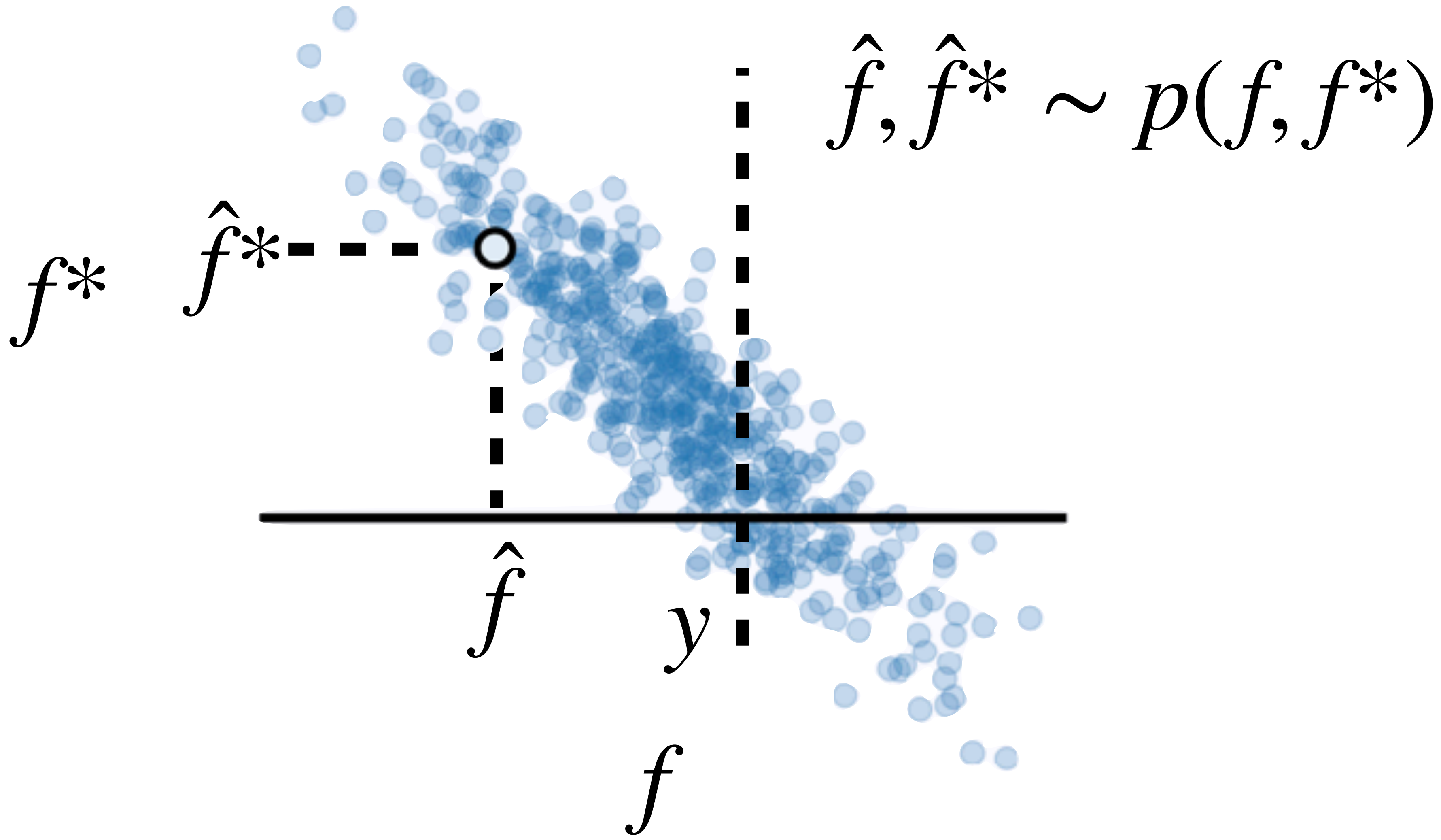


Sample from the Posterior

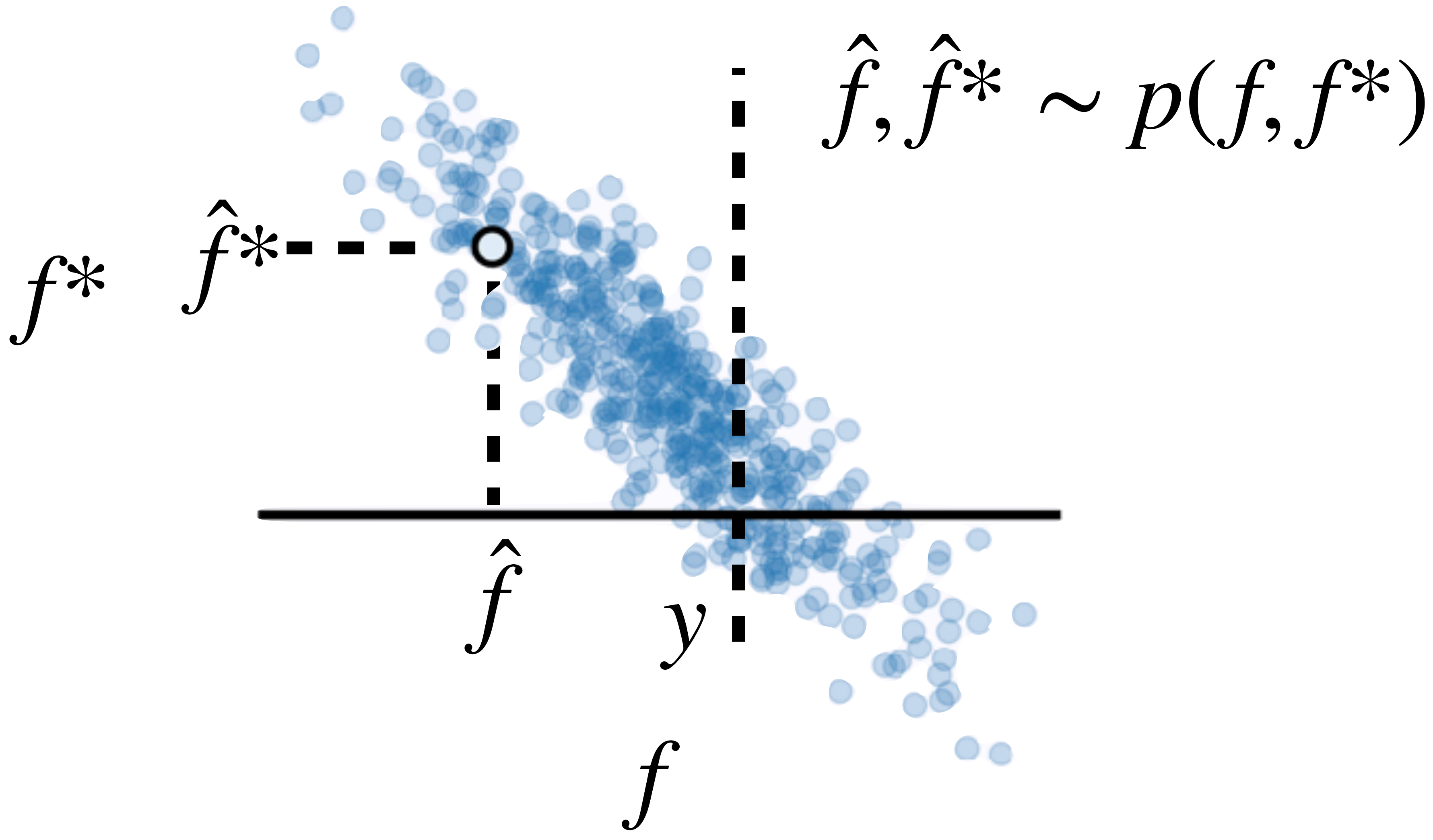
$$\hat{f}, \hat{f}^* \sim p(f, f^*)$$

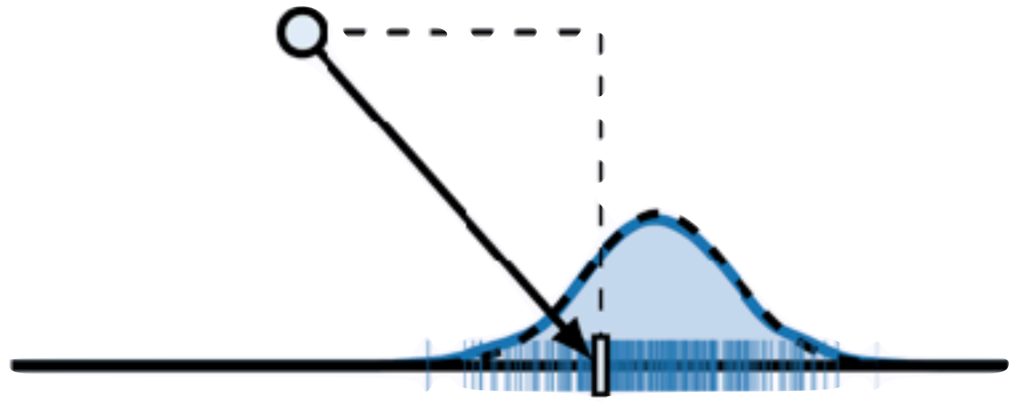


Sample from the Posterior

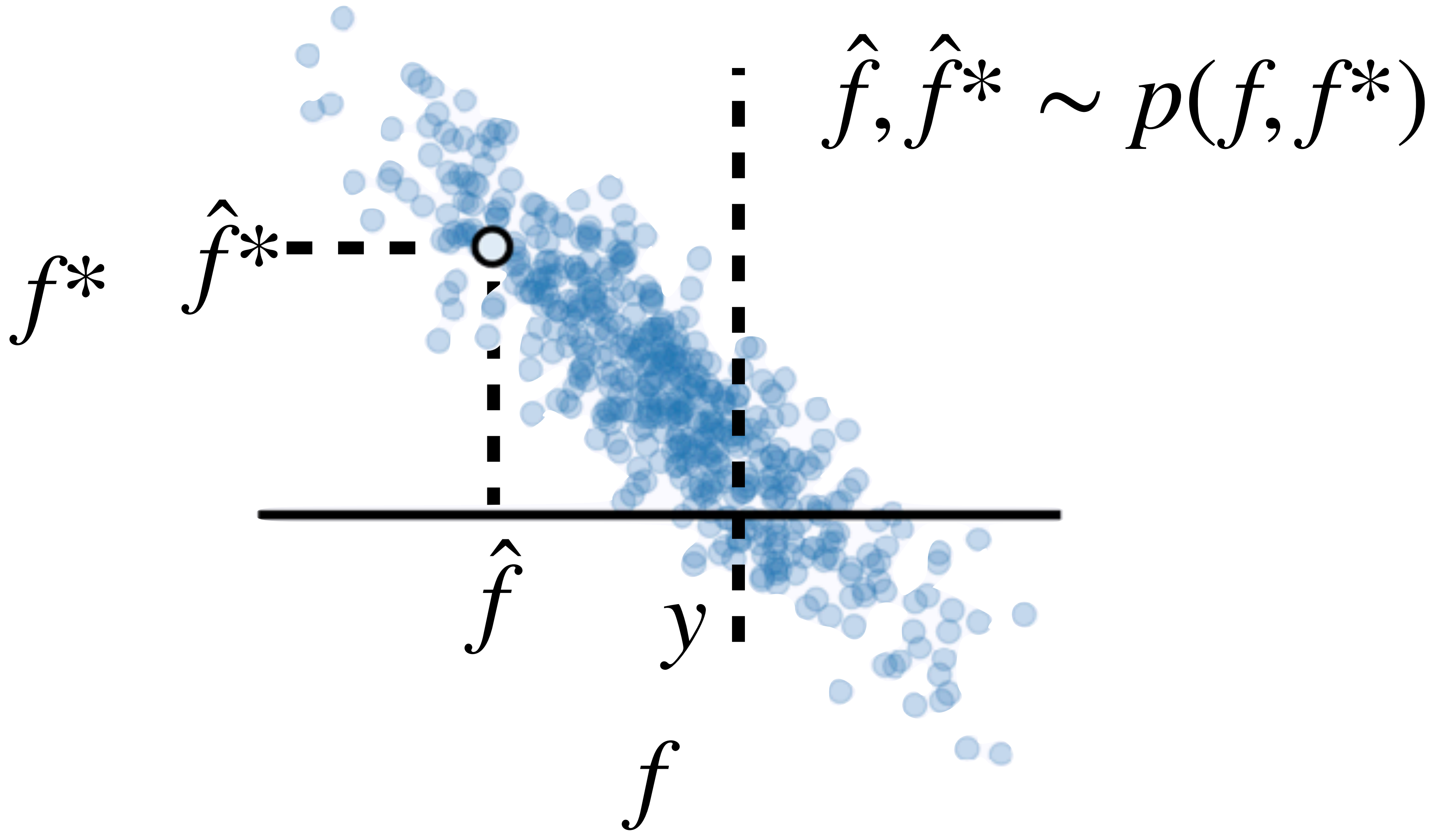


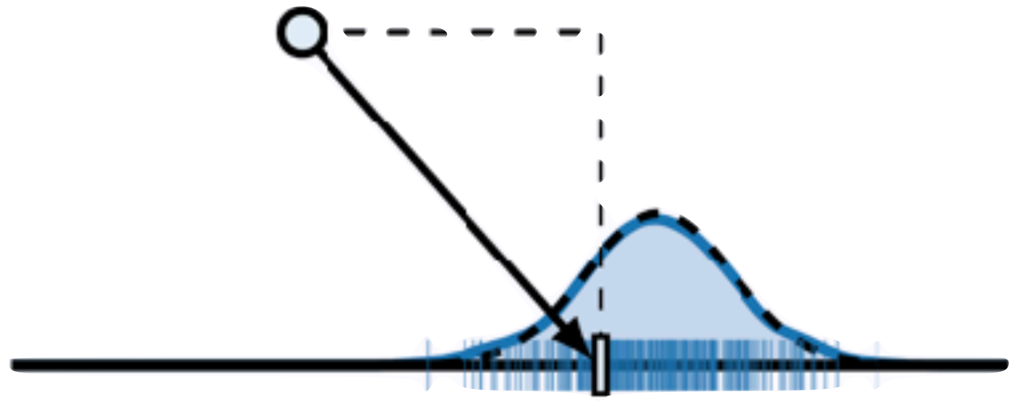
Sample from the Posterior





Sample from the Posterior

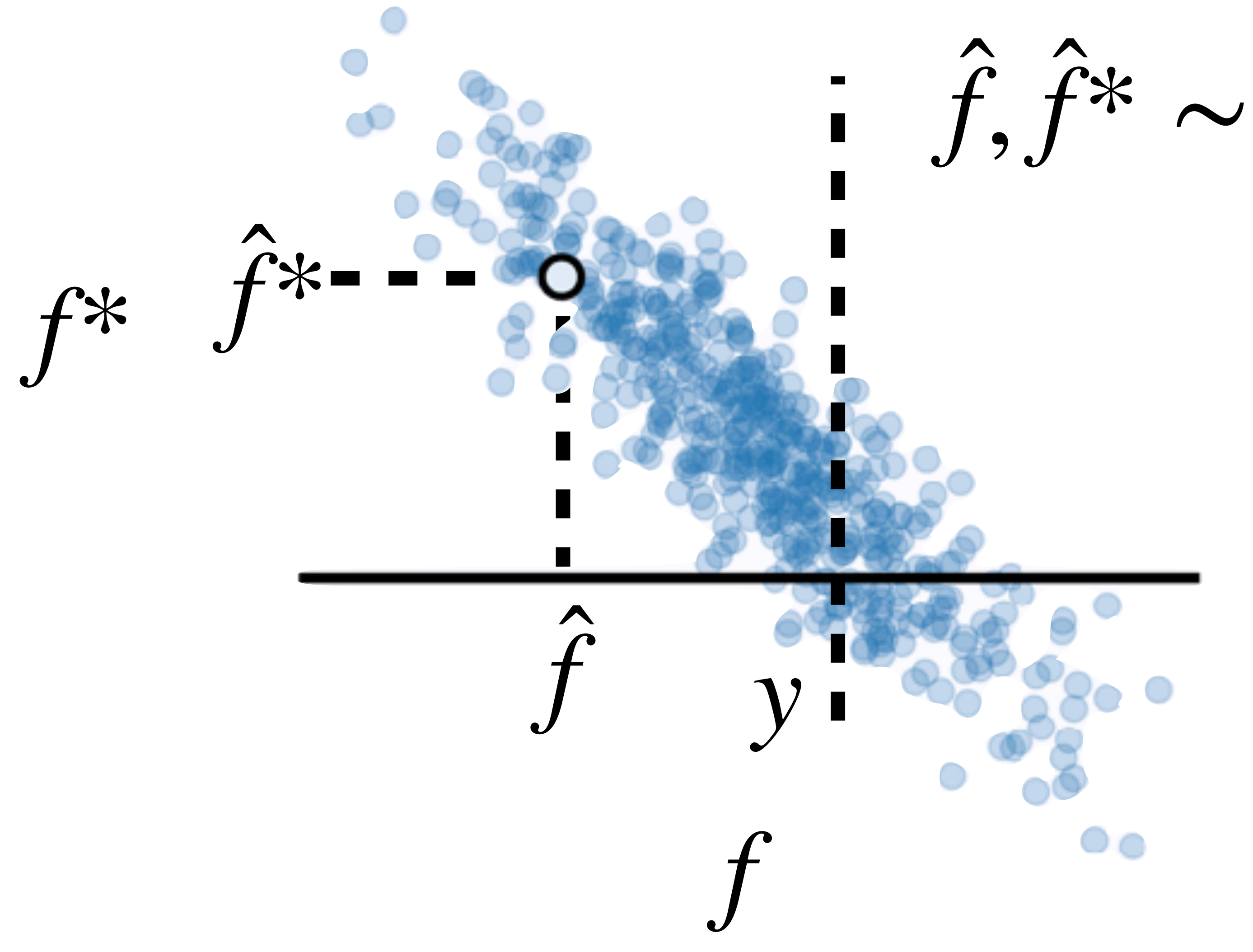


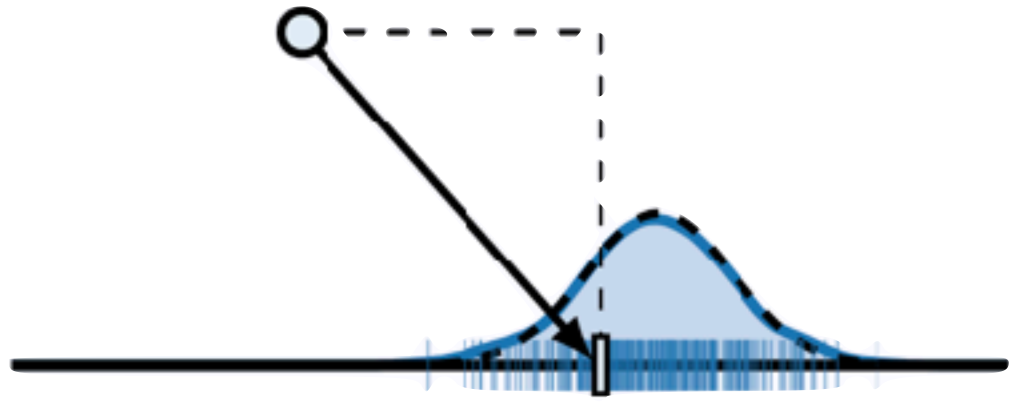


$$(\hat{f}^* \mid \hat{f} = y) = \hat{f}^* + K_{*n} K_{nn}^{-1} (y - \hat{f})$$

(Matheron's Rule)

Sample from the Posterior



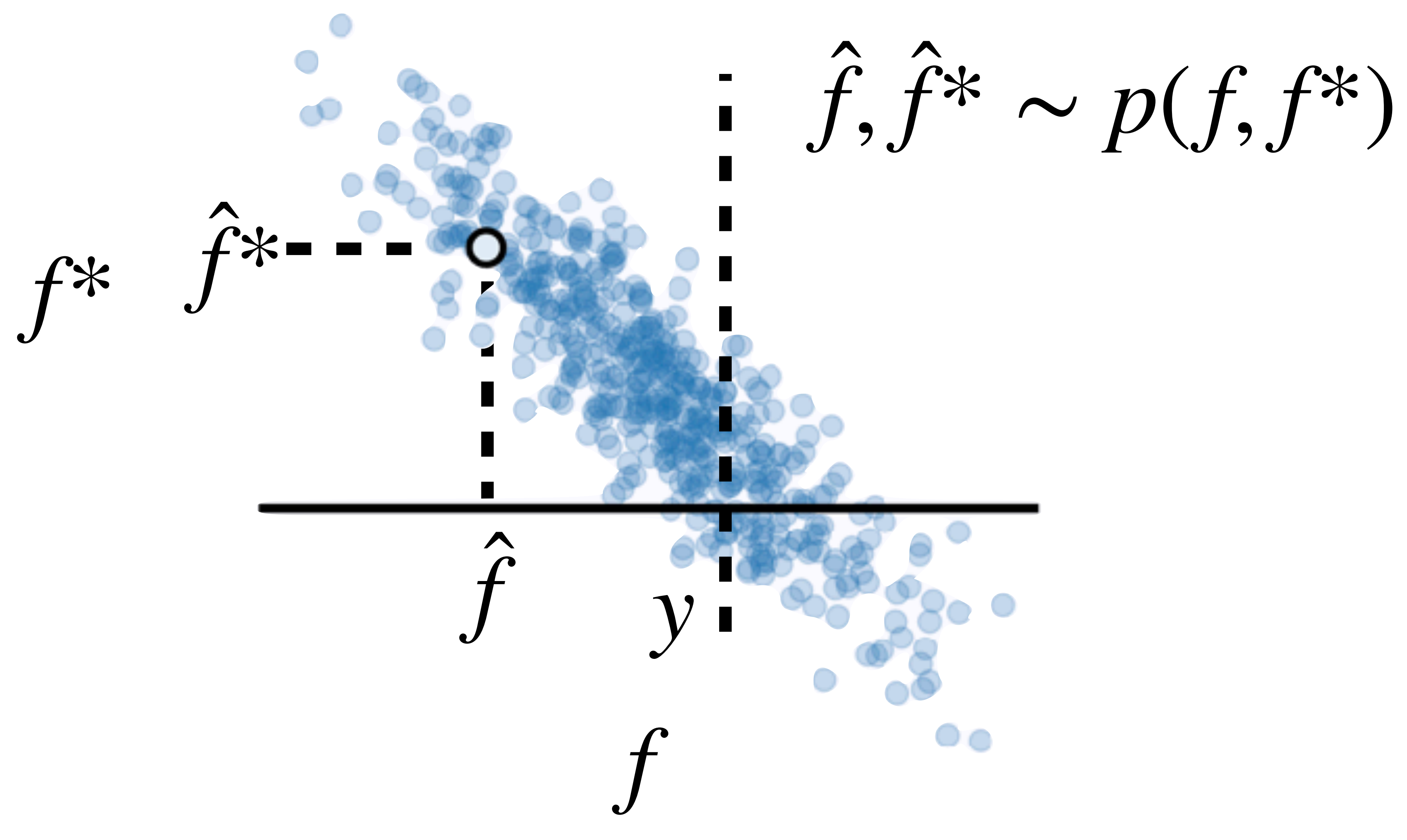


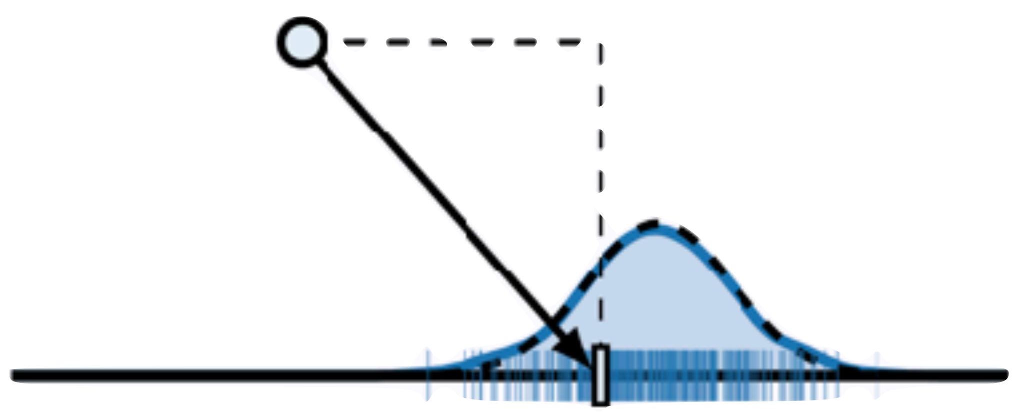
Individual Sample View

$$(\hat{f}^* \mid \hat{f} = y) = \hat{f}^* + K_{*n} K_{nn}^{-1} (y - \hat{f})$$

(Matheron's Rule)

Sample from the Posterior





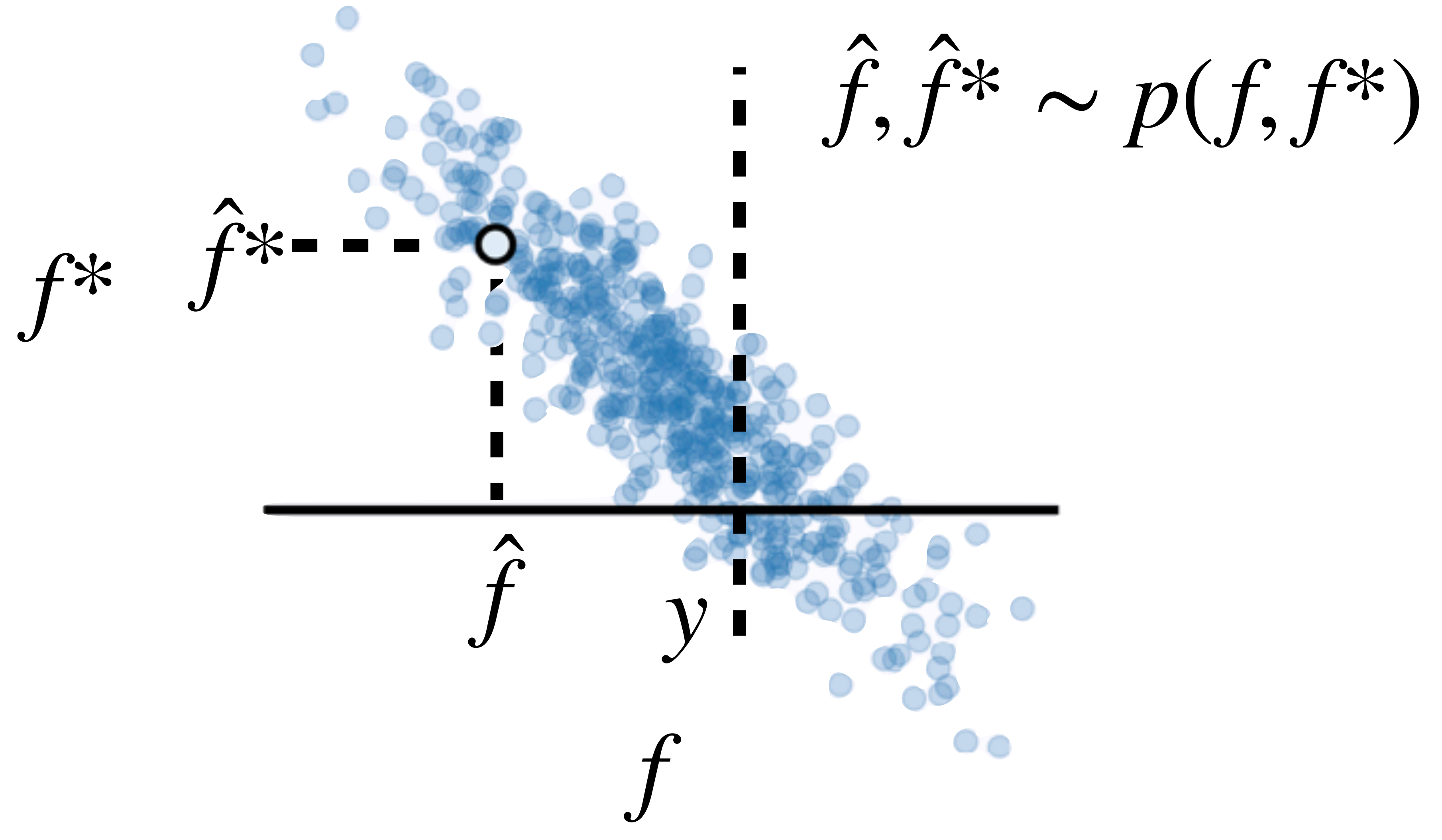
Individual Sample View

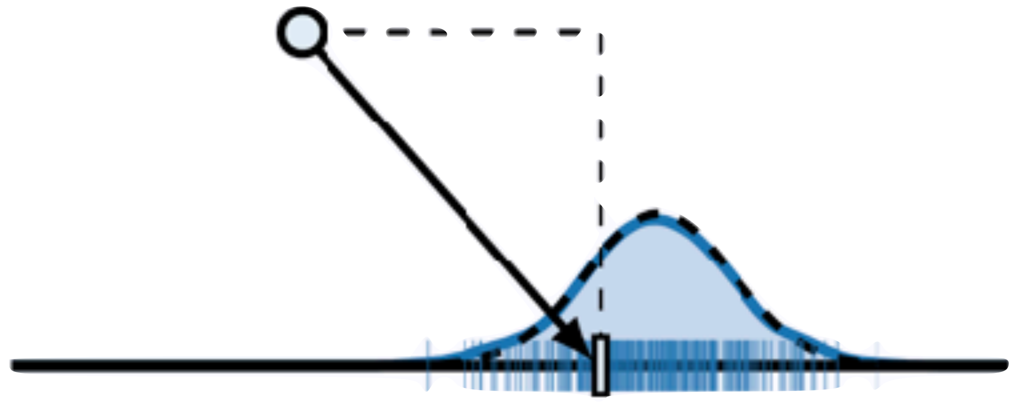
$$(\hat{f}^* \mid \hat{f} = y) = \hat{f}^* + K_{*n} K_{nn}^{-1} (y - \hat{f})$$

(Matheron's Rule)

Going back from a 2D case to a GP conditioned on a dataset $(X, y) \in \mathbb{R}^{n \times d}, \mathbb{R}^n$,

Sample from the Posterior





Individual Sample View

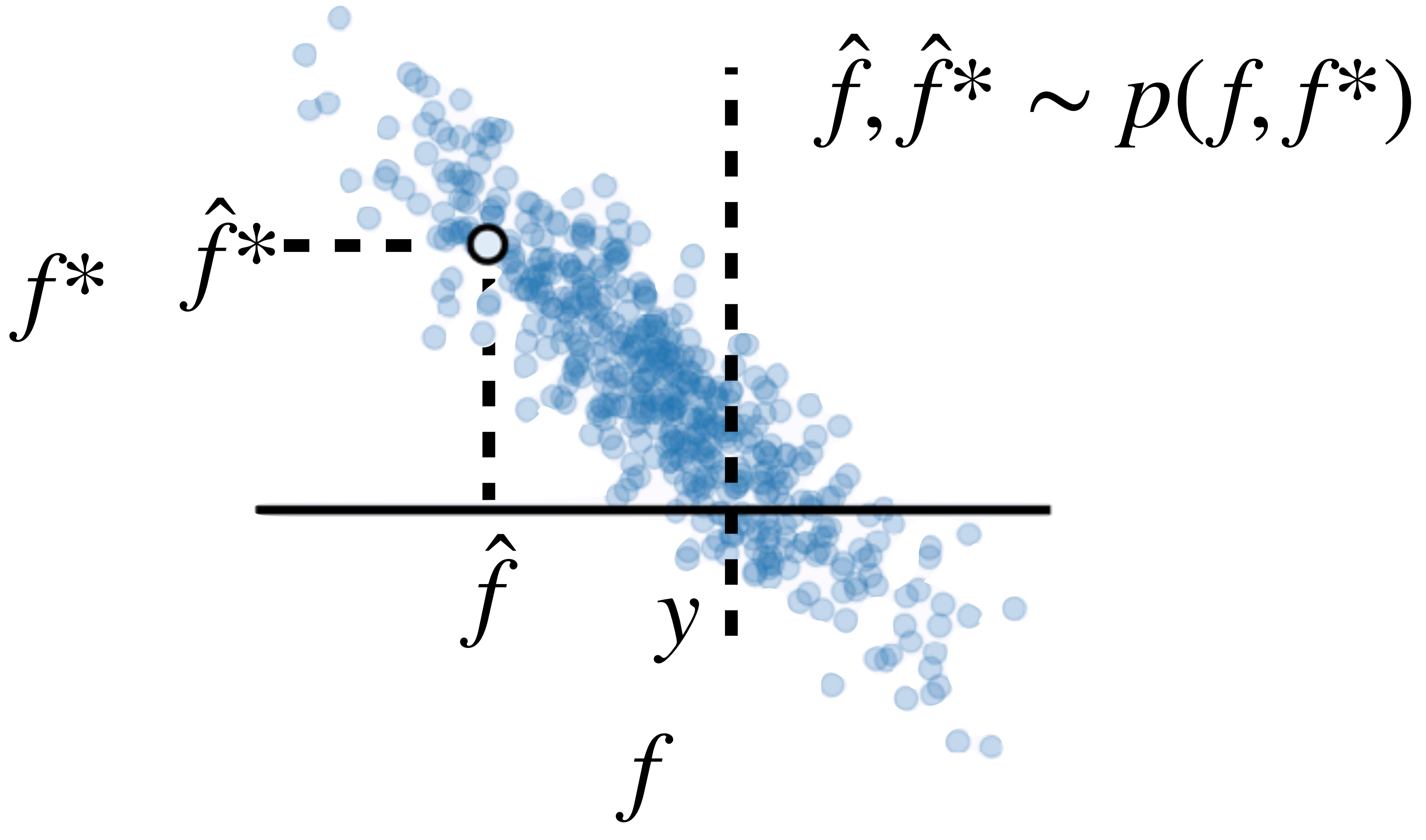
$$(\hat{f}^* \,|\, \hat{f} = y) = \hat{f}^* + K_{*n} K_{nn}^{-1} (y - \hat{f})$$

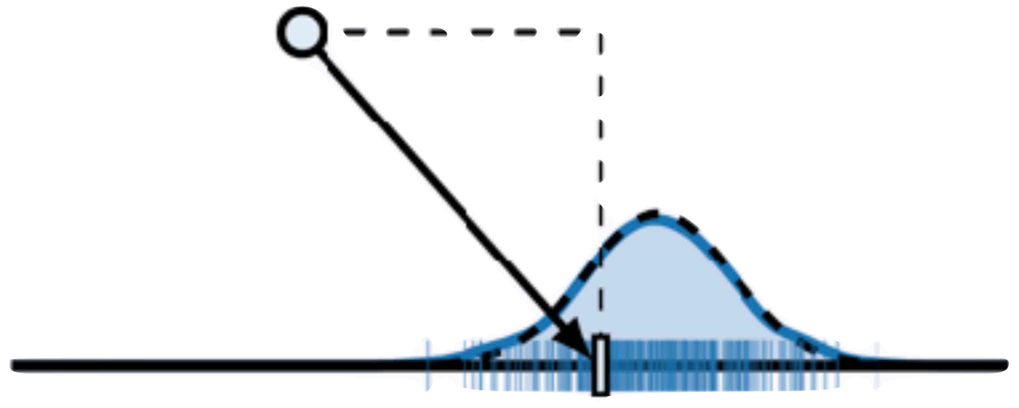
(Matheron's Rule)

Going back from a 2D case to a GP conditioned on a dataset $(X, y) \in \mathbb{R}^{n \times d}, \mathbb{R}^n$,

$$\left[\begin{pmatrix} f(X^*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

Sample from the Posterior





Individual Sample View

$$(\hat{f}^* \mid \hat{f} = y) = \hat{f}^* + K_{*n}K_{nn}^{-1}(y - \hat{f})$$

(Matheron's Rule)

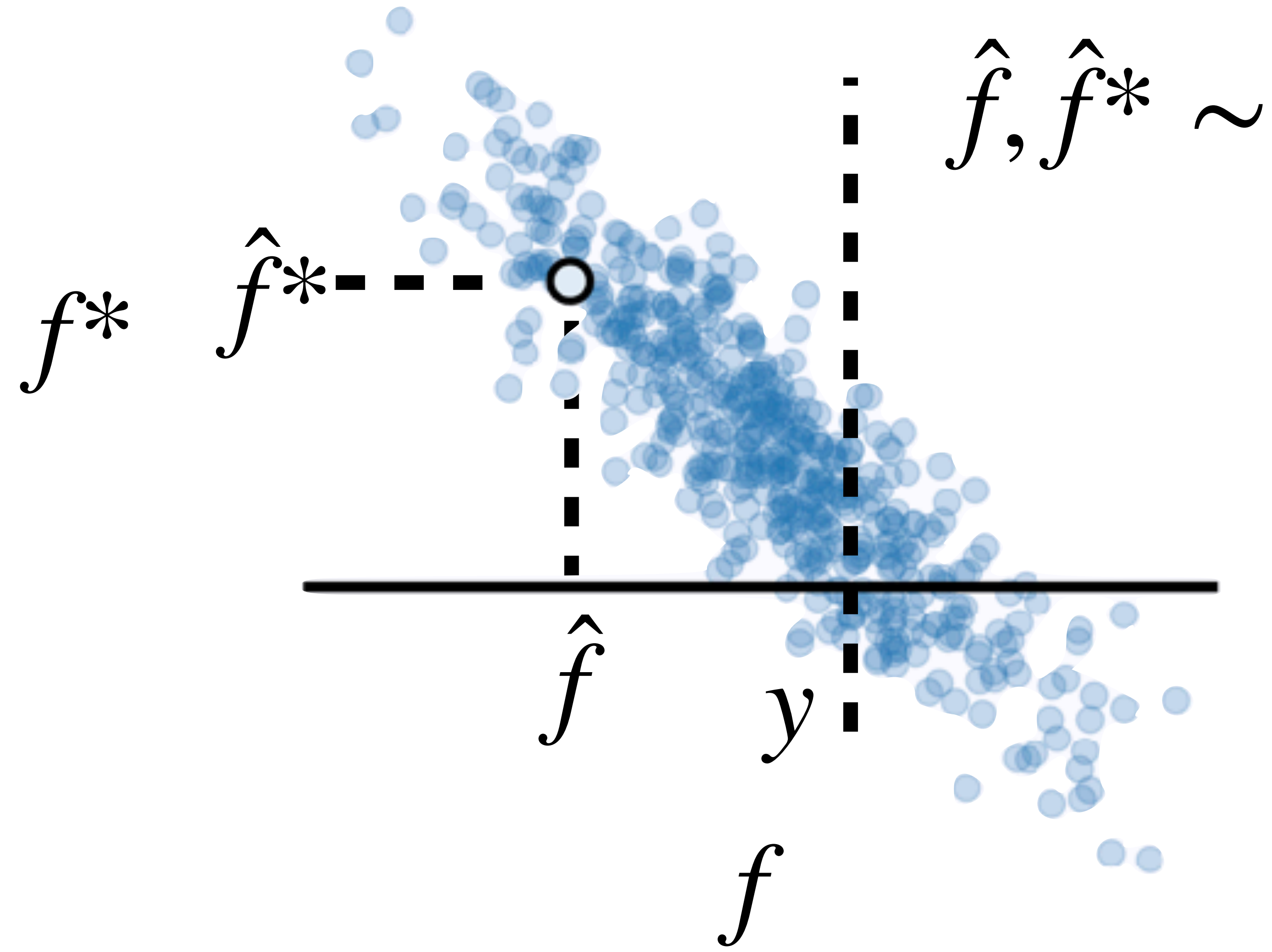
Going back from a 2D case to a GP conditioned on a dataset $(X, y) \in \mathbb{R}^{n \times d}, \mathbb{R}^n$,

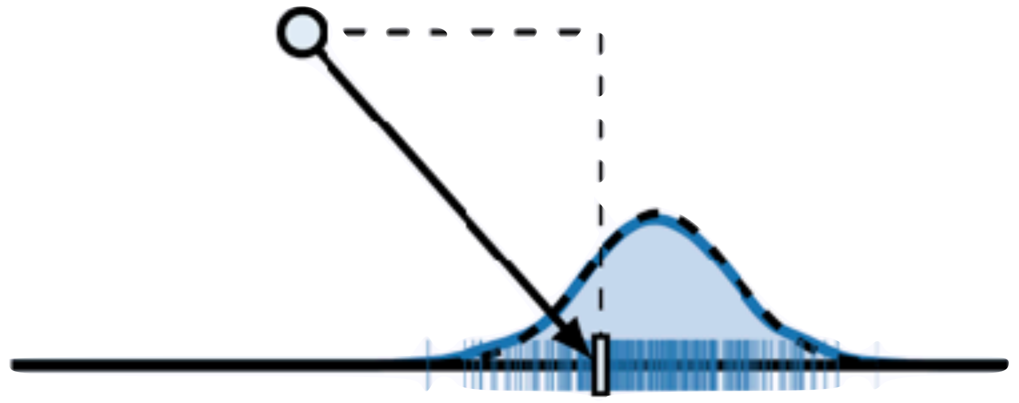
$$\left[\begin{pmatrix} f(X^*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

$$\downarrow$$

$$(f^* \mid f = y) = f(X^*) + K_{*n}K_{nn}^{-1}(y + \epsilon - f(X))$$

Sample from the Posterior





Individual Sample View

$$(\hat{f}^* \,|\, \hat{f} = y) = \hat{f}^* + K_{*n}K_{nn}^{-1}(y - \hat{f})$$

(Matheron's Rule)

Going back from a 2D case to a GP conditioned on a dataset $(X, y) \in \mathbb{R}^{n\times d}, \mathbb{R}^n$,

$$\left[\begin{pmatrix} f(X^*) \\ y \end{pmatrix}\right] \sim \mathcal{N}\left(0, \left[\begin{pmatrix} K_{**} & K_{*n} \\ K_{*n}^\top & K_{nn} + \sigma^2 I \end{pmatrix}\right]\right)$$

$$\downarrow$$

$$(f^* \,|\, f = y) = f(X^*) + K_{*n}K_{nn}^{-1}(y + \epsilon - f(X))$$

$\mathcal{O}(n^3)$

Sample from the Posterior

$(f \mid \boldsymbol{y})(\,\cdot\,) =$

Sample from the Posterior

$$(f \mid \boldsymbol{y})(\,\cdot\,) = f(\,\cdot\,)$$

Sample from the Posterior

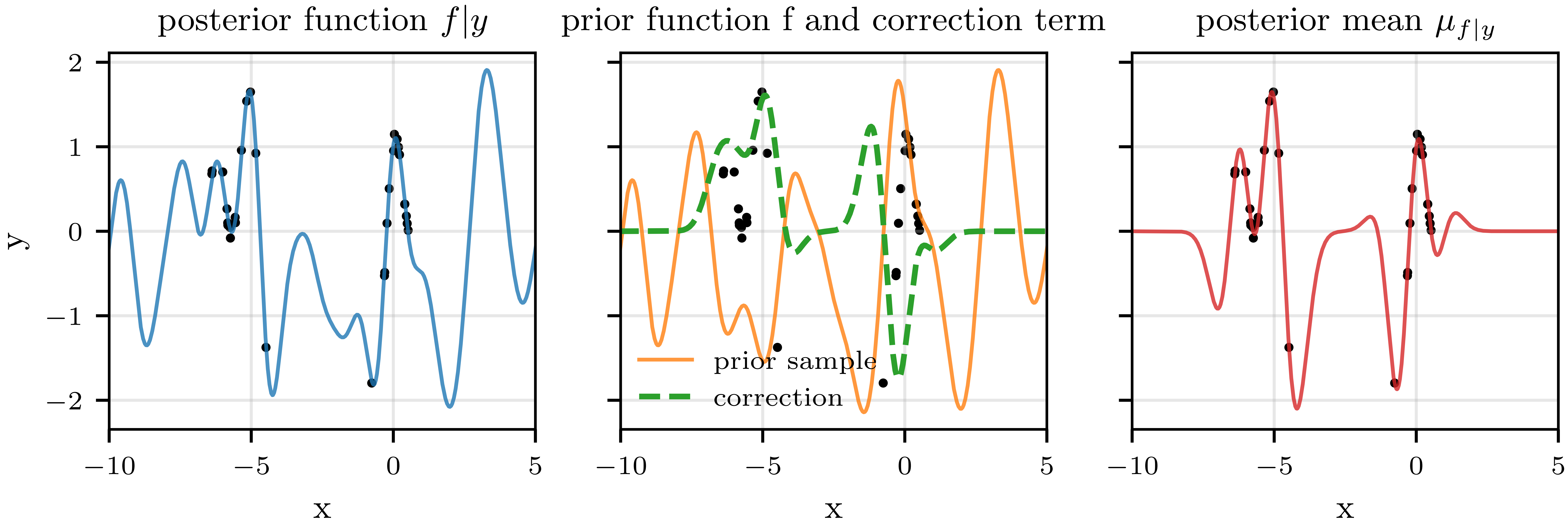
$$(f \mid \boldsymbol{y})(\,\cdot\,) = \; f(\,\cdot\,) \qquad \underbrace{+K_{(\cdot)n}\left(K_{nn} + \sigma^2 I\right)^{-1} y}_{\text{mean } \mu_{f\mid\boldsymbol{y}}(\cdot)}$$

Sample from the Posterior

$$(f \mid \boldsymbol{y})(\,\cdot\,) = \; f(\,\cdot\,) \;\; \underbrace{+K_{(\cdot)n}\left(K_{nn} + \sigma^2 I\right)^{-1}(-f(x) + \epsilon)}_{\text{correction term}} \;\; \underbrace{+K_{(\cdot)n}\left(K_{nn} + \sigma^2 I\right)^{-1} y}_{\text{mean } \mu_{f|\mathbf{y}}(\cdot)}$$

Sample from the Posterior

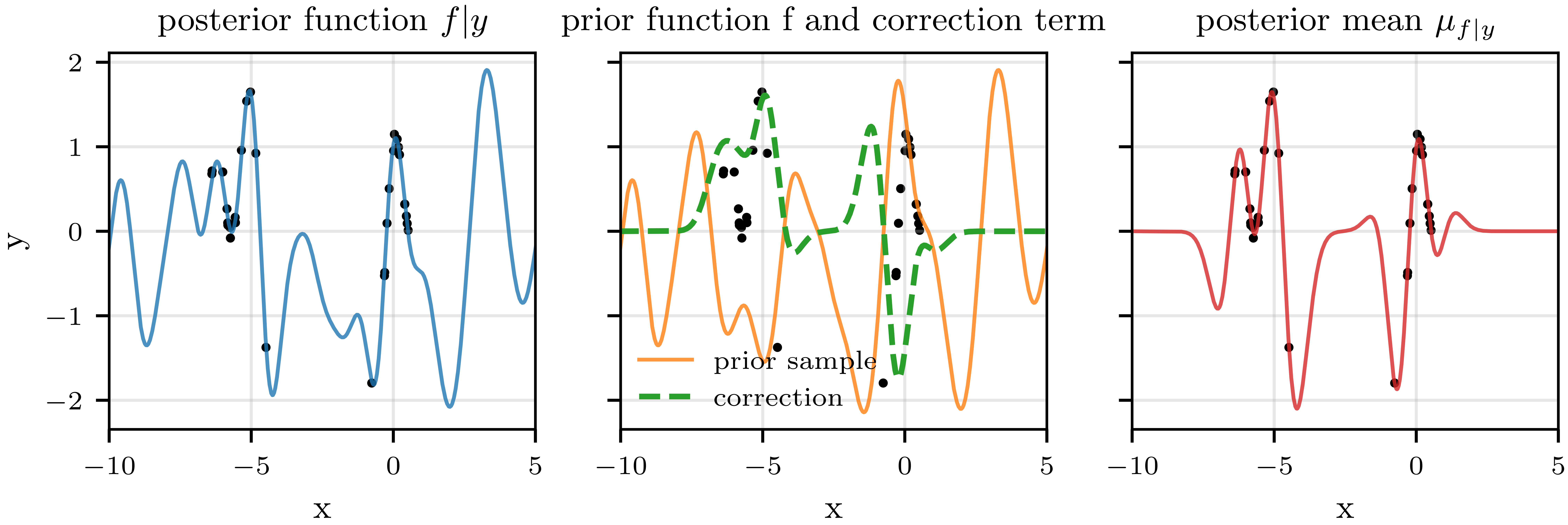
$$(f \mid \boldsymbol{y})(\,\cdot\,) = \; f(\,\cdot\,) \; \underbrace{+K_{(\cdot)n}\left(K_{nn}+\sigma^2 I\right)^{-1}(-f(x)+\epsilon)}_{\text{correction term}} \; \underbrace{+K_{(\cdot)n}\left(K_{nn}+\sigma^2 I\right)^{-1} y}_{\text{mean } \mu_{f|\mathbf{y}}(\cdot)}$$



Sample from the Posterior

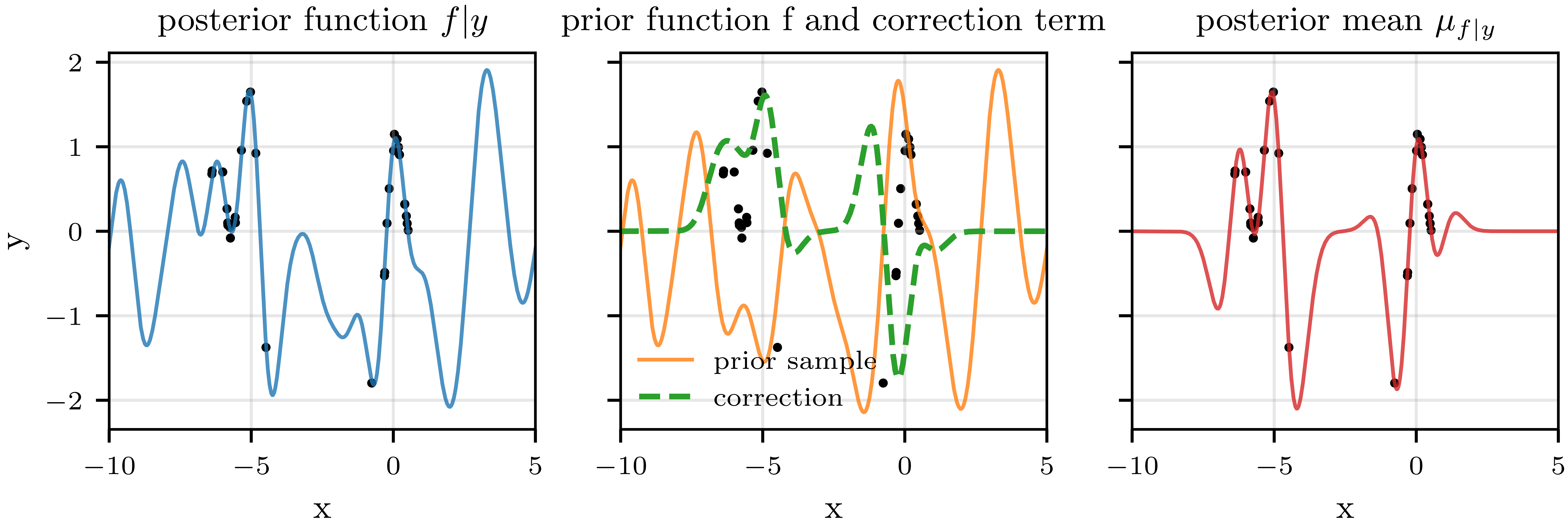
$$(f \mid \boldsymbol{y})(\,\cdot\,) = f(\,\cdot\,) \underbrace{+K_{(\cdot)n}\left(K_{nn}+\sigma^2 I\right)^{-1}(-f(x)+\epsilon)}_{\text{correction term}} \underbrace{+K_{(\cdot)n}\left(K_{nn}+\sigma^2 I\right)^{-1} y}_{\text{mean } \mu_{f|\mathbf{y}}(\cdot)}$$

v^*

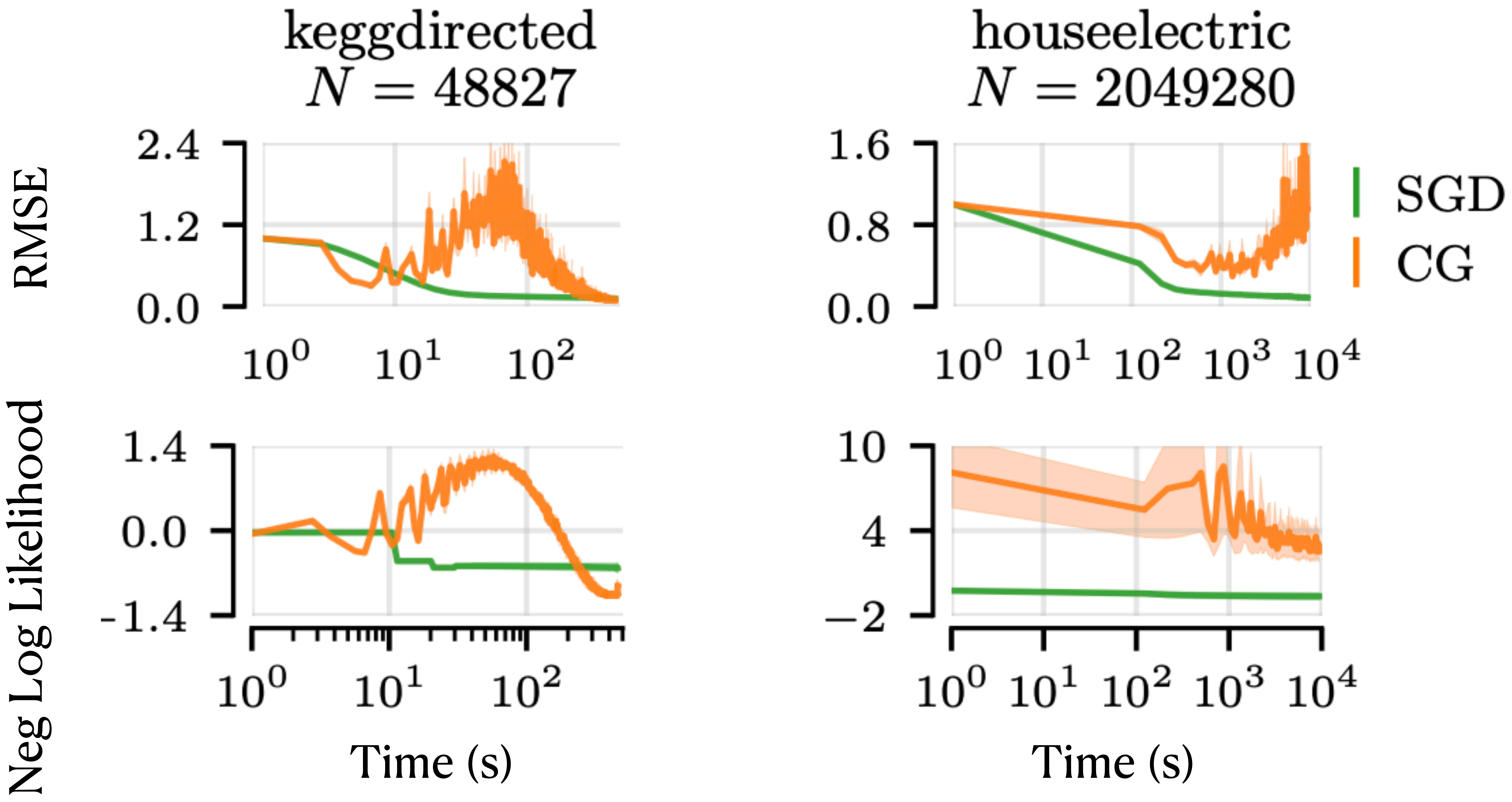


Sample from the Posterior

$$(f \mid \boldsymbol{y})(\cdot) = f(\cdot) \underbrace{+K_{(\cdot)n}\overset{v^*_{\text{sample}}}{\left(K_{nn}+\sigma^2 I\right)^{-1}(-f(x)+\epsilon)}}_{\text{correction term}} \underbrace{+K_{(\cdot)n}\overset{v^*}{\left(K_{nn}+\sigma^2 I\right)^{-1} y}}_{\text{mean } \mu_{f|\mathbf{y}}(\cdot)}$$

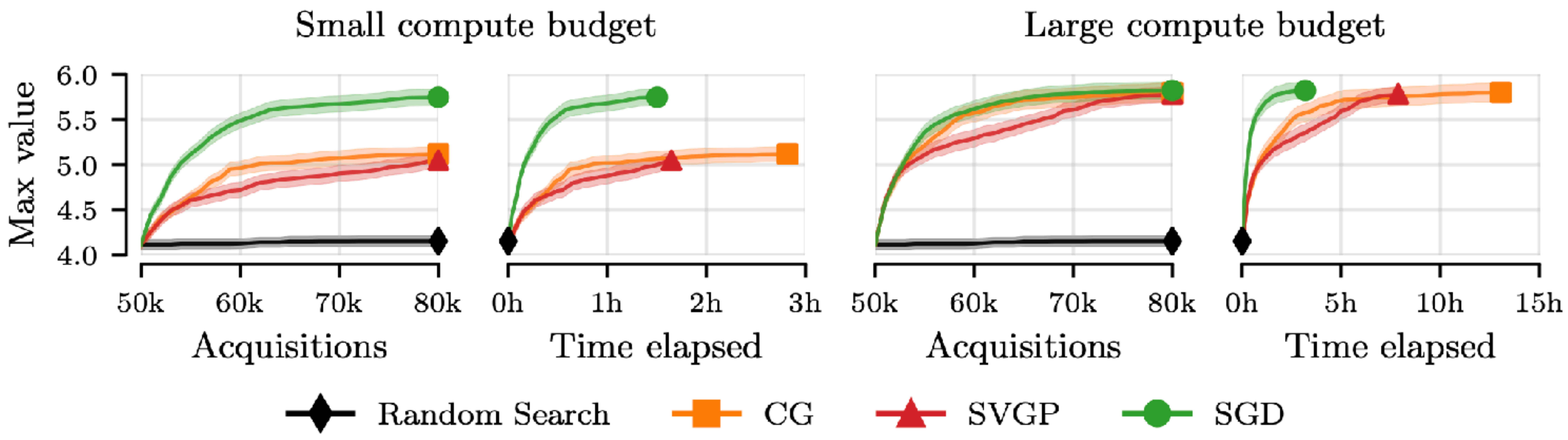


SGD scales much better in uncertainty estimates



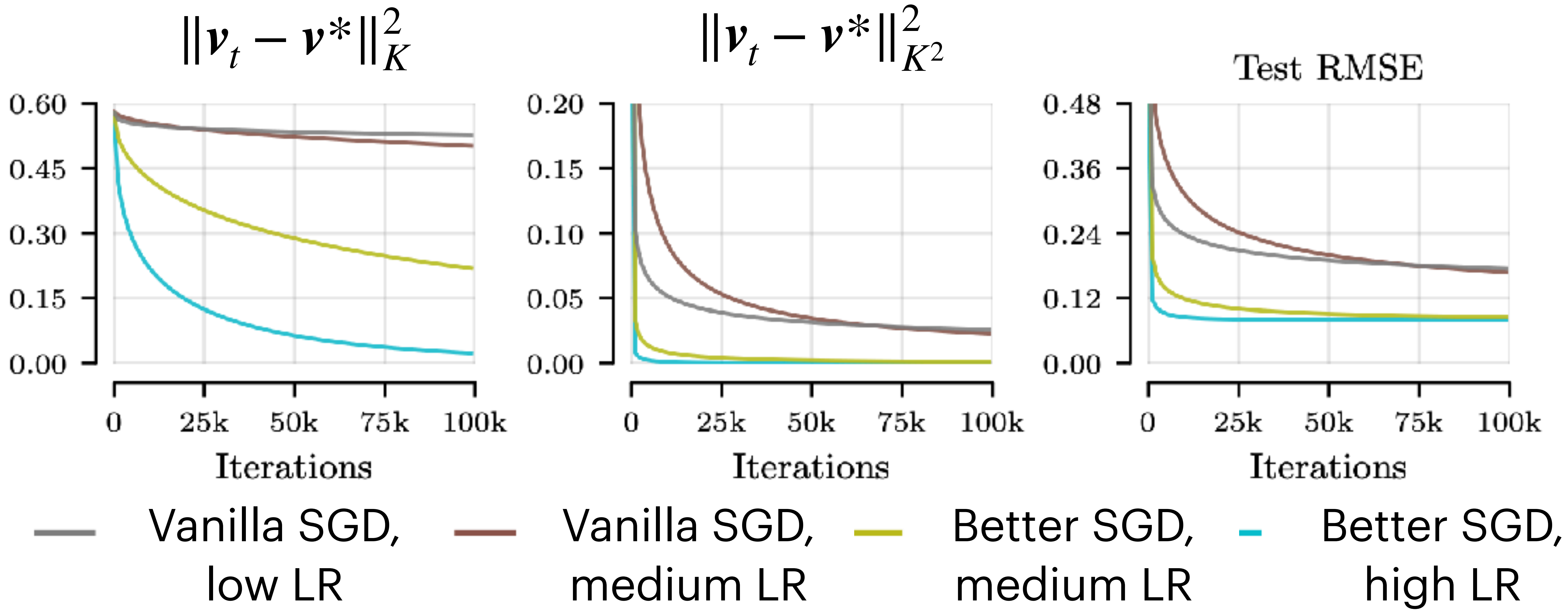
Where can we apply this?

- Sequential Decision Making -> Bayesian Optimisation at a fixed compute budget



Where can we improve this?

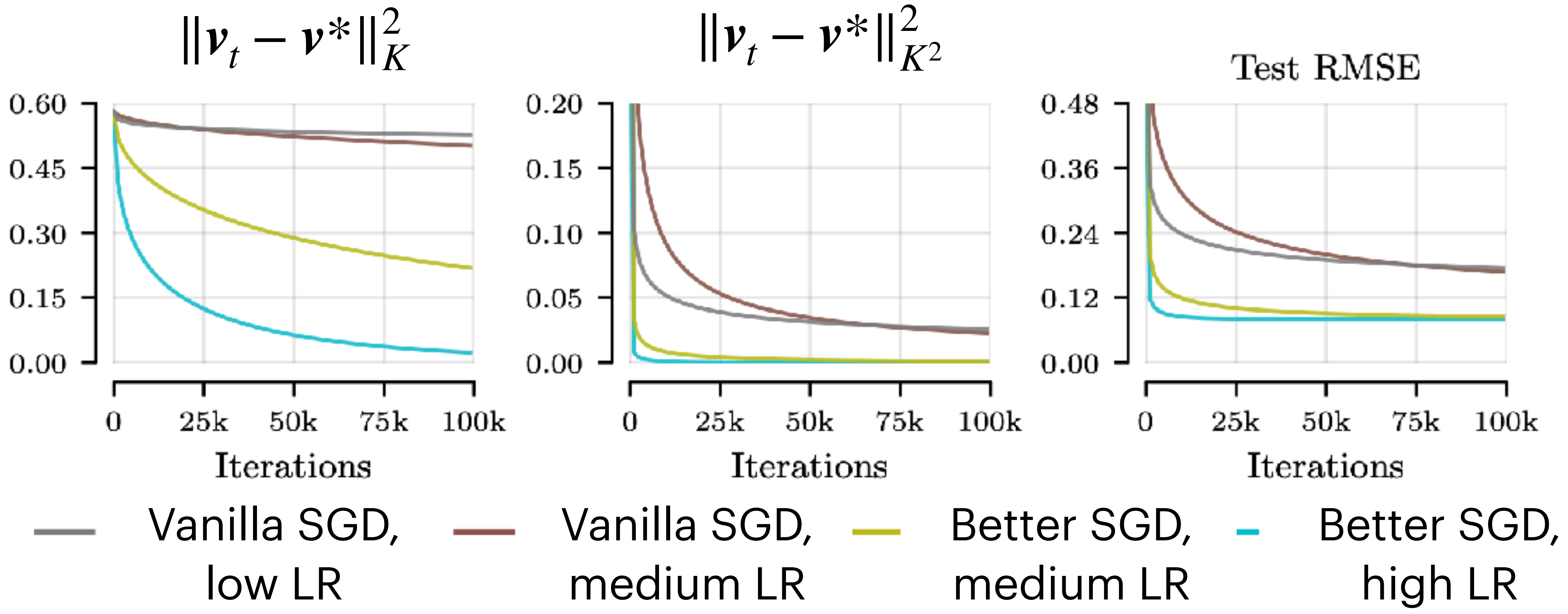
- We can derive an SGD objective that is much faster and even better-conditioned



	Data Size	HOUSEELEC 2M
RMSE	SDD*	**0.04 ± 0.00**
	SGD	0.09 ± 0.00
	CG	0.87 ± 0.14
	SVGP	0.12 ± 0.00
Time (min)	SDD*	**47.8 ± 0.02**
	SGD	69.5 ± 0.06
	CG	157 ± 0.01
	SVGP	154 ± 0.12
NLL	SDD*	**-1.46 ± 0.10**
	SGD	-1.09 ± 0.04
	CG	2.07 ± 0.58
	SVGP	-0.61 ± 0.01

Where can we improve this?

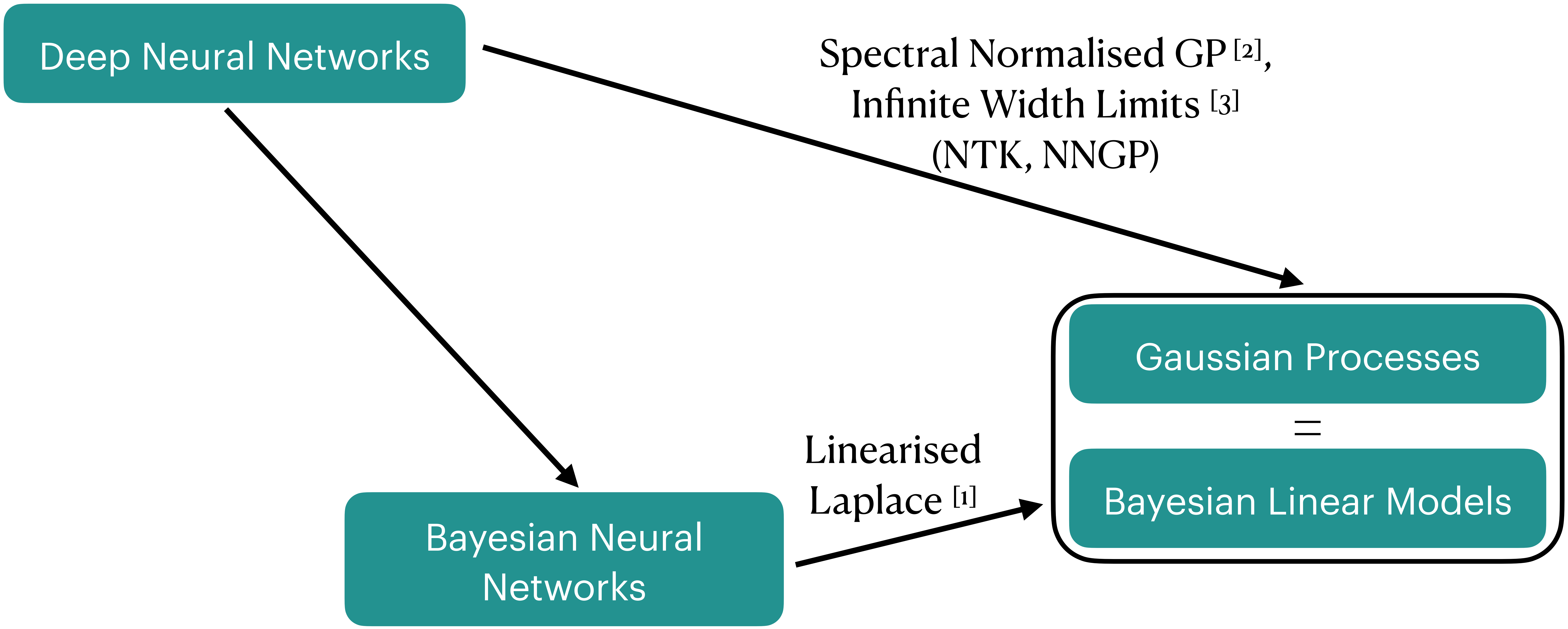
- We can derive an SGD objective that is much faster and even better-conditioned



	Data	HOUSEELEC
	Size	2M
RMSE	SDD*	**0.04 ± 0.00**
	SGD	0.09 ± 0.00
	CG	0.87 ± 0.14
	SVGP	0.12 ± 0.00
Time (min)	SDD*	**47.8 ± 0.02**
	SGD	69.5 ± 0.06
	CG	157 ± 0.01
	SVGP	154 ± 0.12
NLL	SDD*	**-1.46 ± 0.10**
	SGD	-1.09 ± 0.04
	CG	2.07 ± 0.58
	SVGP	-0.61 ± 0.01

[1] **Padhy, S.***, Lin, J. A.*, Antorán, J.*, Tripp, A., Terenin, A., Szepesvári, C., ... & Janz, D. Stochastic Gradient Descent for Gaussian Processes Done Right. ***ICLR 2024***

How can we apply this to Deep Learning?



[1] **Padhy, S.***, Antorán, J.,*, Barbano, R., Nalisnick, E., ... and Hernández-Lobato, J.M., 2022. Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***
[2] **Padhy, S.***, Liu, J. Z.*, Ren, J.*, Lin, Z., Wen, Y., Jerfel, G., ... & Lakshminarayanan, B. A simple approach to improve single-model deep uncertainty via distance-awareness. ***JMLR 2023***
[3] Adlam, B., Lee, J., **Padhy, S.**, Nado, Z. and Snoek, J., 2023. Kernel Regression with Infinite-Width Neural Networks on Millions of Examples. *arXiv preprint*

Uncertainty in Deep NNs

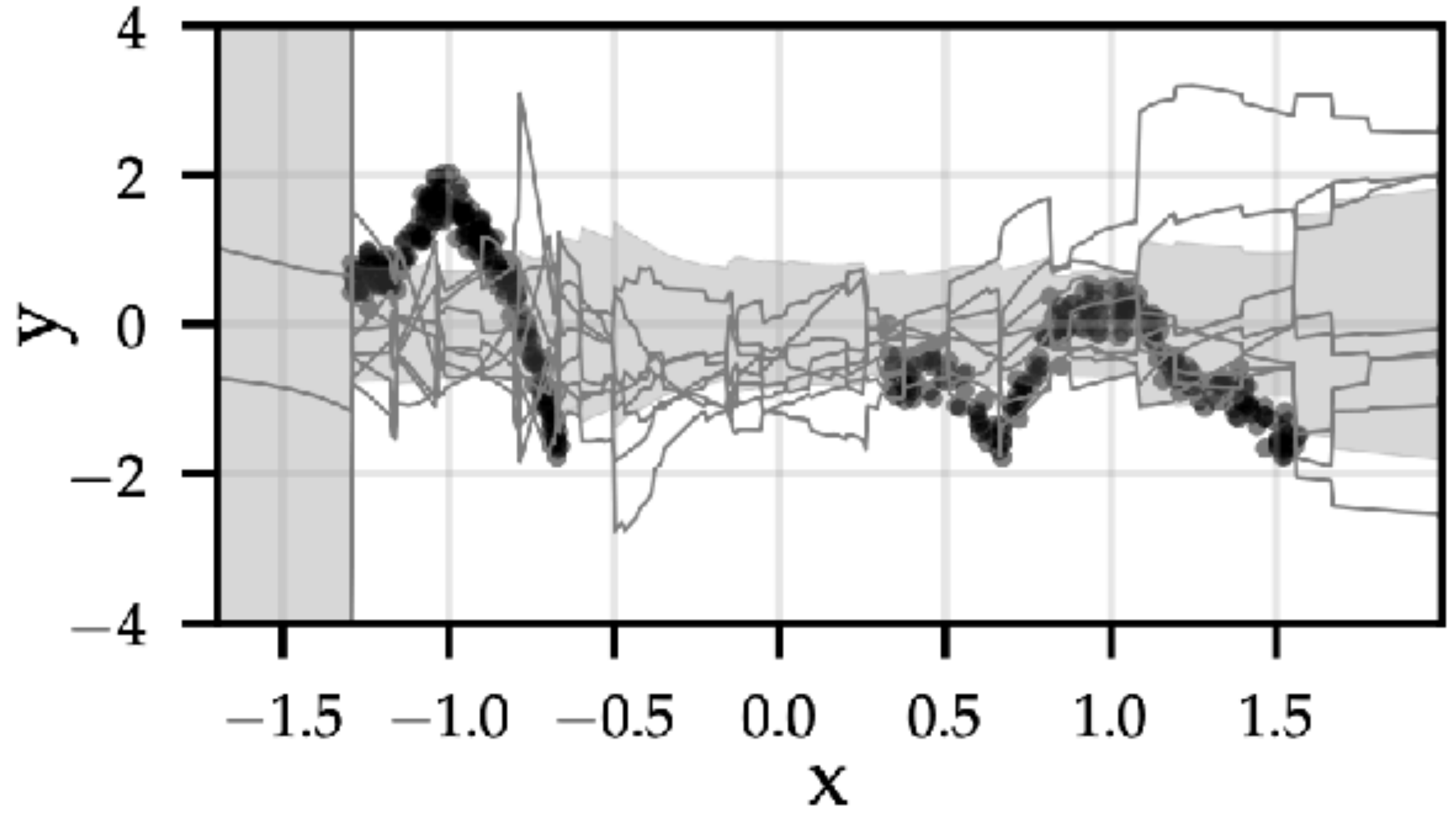
- Given a neural network $f : \mathbb{R}^{d'} \to \mathbb{R}^m$ parameterised by $\theta \in \mathbb{R}^d$
- We estimate uncertainty in $f(x)$ as uncertainty in the tangent **linear** model around MAP $\bar{w}$

$$h(\theta, x) = f(\bar{w}, x) + \nabla_w f(\bar{w}, x)(\theta - \bar{w}), \qquad \theta \sim \mathcal{N}(0,\ A^{-1})$$

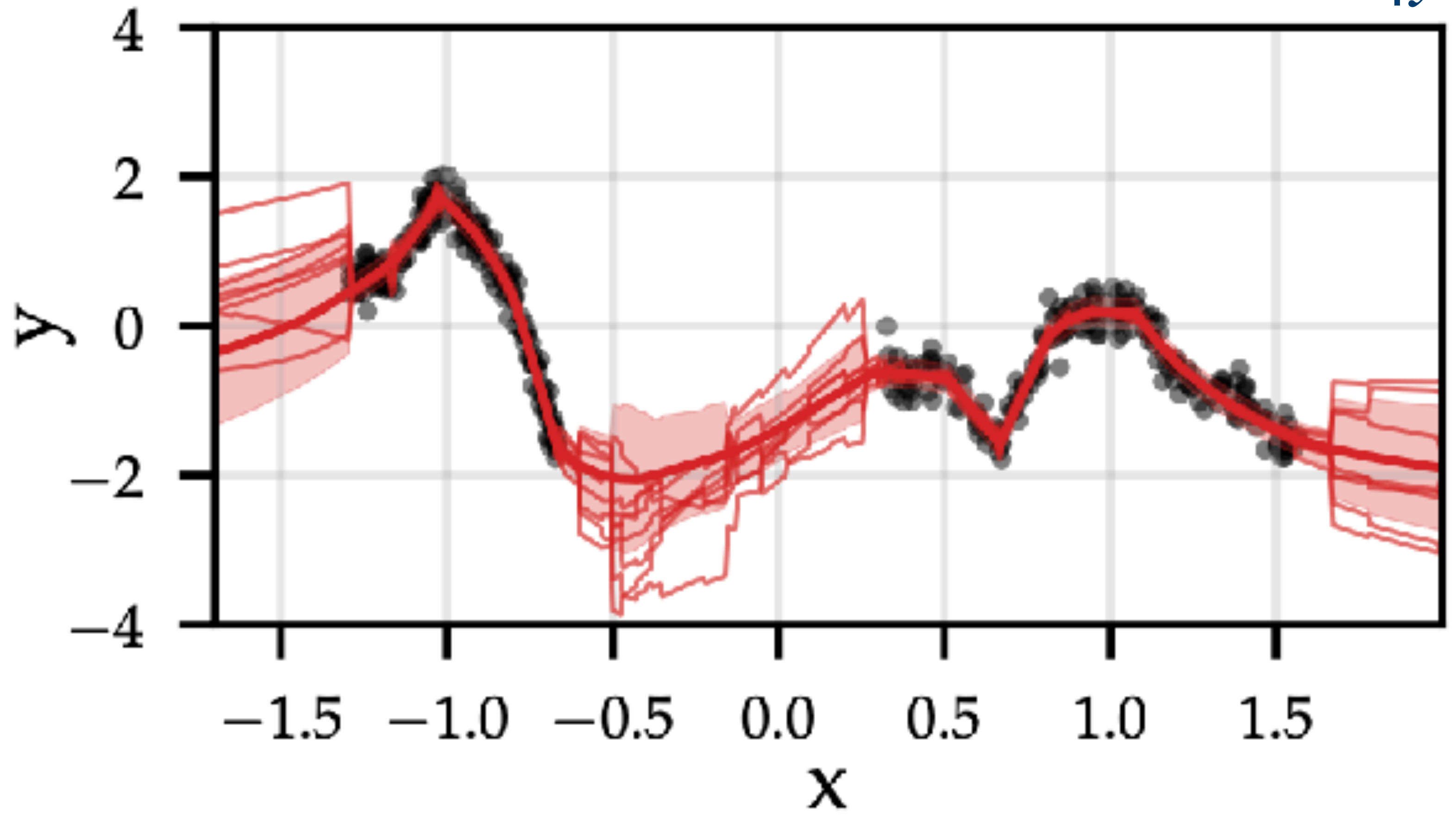
$$h(\theta, x) = \text{MAP solution} + J(x)(\theta - \bar{w})$$

- Turns out $h \sim \mathrm{GP}(0,k)$ where $k(x_i, x_j) = J(x_i)^T A^{-1} J(x_j)$

Prior samples $h \sim GP(0, k)$



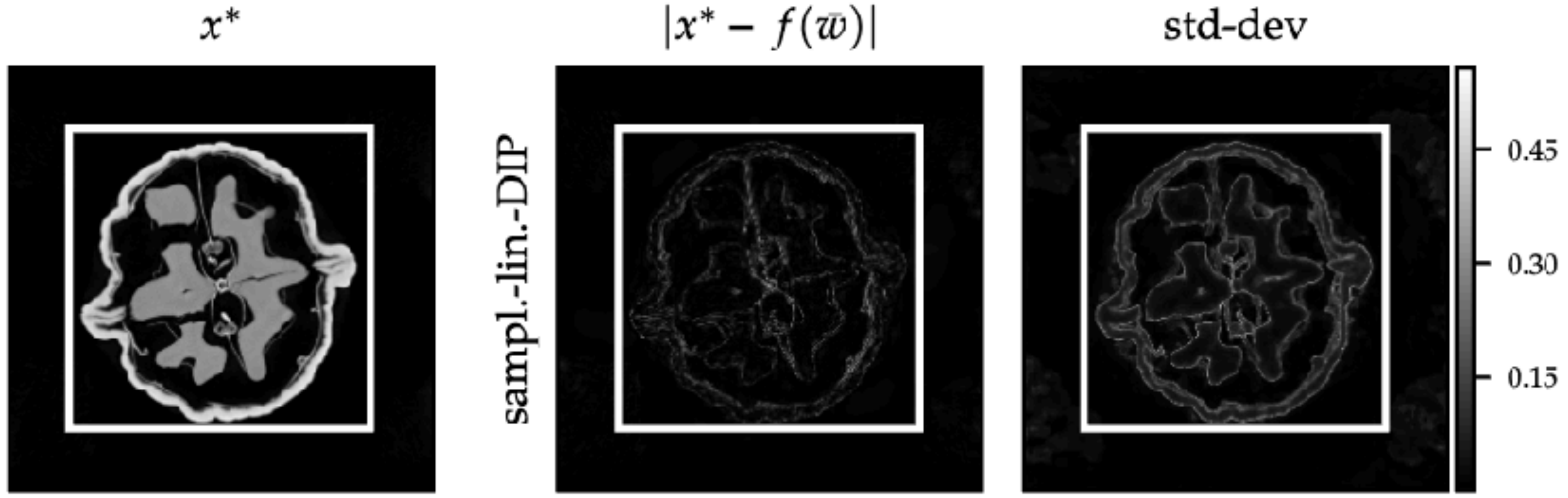
Posterior samples $h \sim GP(\mu_{h|y}, k_{h|y})$



Where can we scale this?

- ImageNet-scale [1] ($nm = 2B, d = 1.5M$)
- 2D Computed Tomography [1] ($m = 13k, d = 3M$)
- Large-scale/ill-conditioned regression [2, 3] ($n = 2M$)

	$m = 7680$			
	LL		wall-clock time (min.)	
Method	marginal	(10×10)	params optim.	prediction
MCDO-UNet	0.028	2.474	0	$3'$
lin.-UNet	2.214	2.601	$1260'$	$196'$
sampl.-lin.-UNet	**2.341**	**2.869**	$12'$	$14'$



	Dataset	HOUSEELEC
	N	2049280
RMSE	SGD	$\mathbf{0.09 \pm 0.00}$
RMSE	CG	0.87 ± 0.14
RMSE	SVGP	0.10 ± 0.02
RMSE †	SGD	$\mathbf{0.09 \pm 0.00}$
RMSE †	CG	0.93 ± 0.19
RMSE †	SVGP	—
Hours	SGD	2.69 ± 0.91
Hours	CG	2.62 ± 0.01
Hours	SVGP	$\mathbf{0.04 \pm 0.00}$

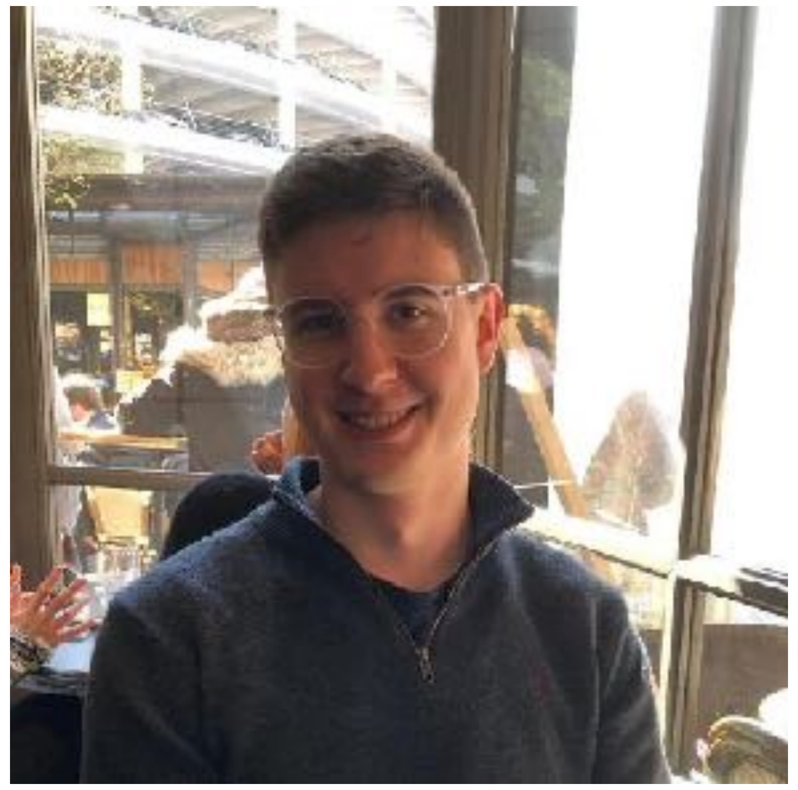
[1] **Padhy, S.***, Antorán, J.,*, Barbano, R., Nalisnick, E., ... and Hernández-Lobato, J.M., 2022. Sampling-based inference for large linear models, with application to linearised Laplace. ***ICLR 2023***

My Collaborators

Andy Lin

Javier Antoran

Riccardo Barbano



Dave Janz

Alex Terenin

Miguel Hernandez-Lobato

Appendix: Linear Models are GPs

$$y_i = \phi(x_i)\theta + \eta_i$$

$$y_i \in \mathbb{R}^m$$

$$\theta \in \mathbb{R}^d$$

$$\phi(x_i) \in \mathbb{R}^{m \times d}$$

$$i \in \{1,\ldots,n\}$$

$$\theta \sim \mathcal{N}(0, A^{-1})$$

$$\eta_i \sim \mathcal{N}(0, B_i^{-1})$$

$$\longrightarrow$$

$$y_i = GP(0, k(\,.\,,\,.\,)) + \eta_i$$

where $K_n n = \Phi^\top A^{-1} \Phi$