Deep and Multi-fidelity learning with Gaussian processes

Andreas Damianou, Sr ML Scientist

Advances in Data Science Seminar Series, Univ. Manchester
15 Oct. 2019

andreasdamianou.com

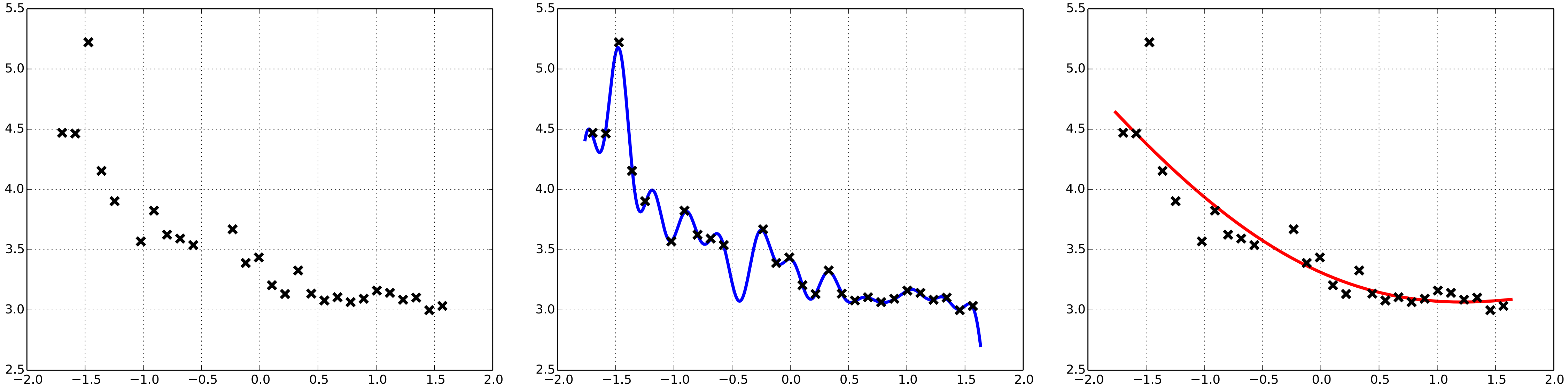
Various parts of this talk come from work with:

Neil Lawrence
Kurt Cutajar
Paris Perdikaris
Mark Pullin
Javier Gonzalez

Outline

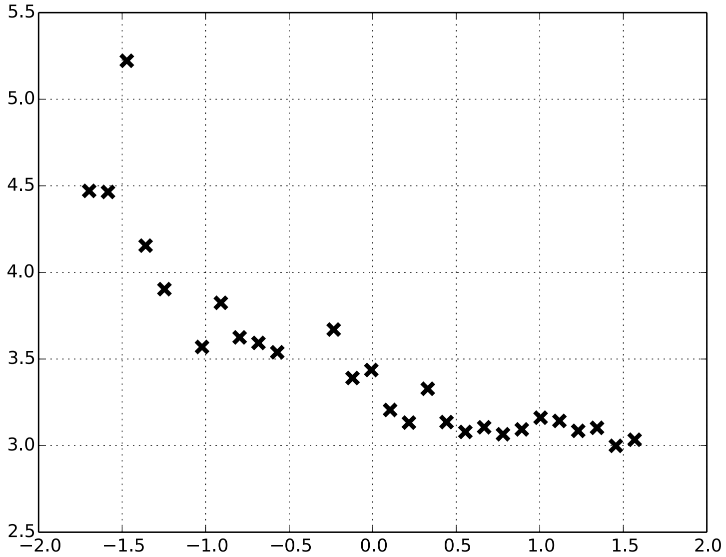
- Gaussian processes (GPs):
  - Reasoning about functions through smoothness assumptions
- Deep Gaussian processes (DGPs)
  - A much richer class of (deep) models: compositions of GPs
- Multi-fidelity modelling with DGPs:
  - Learn from multiple sources by treating the layers of DGP as fidelities

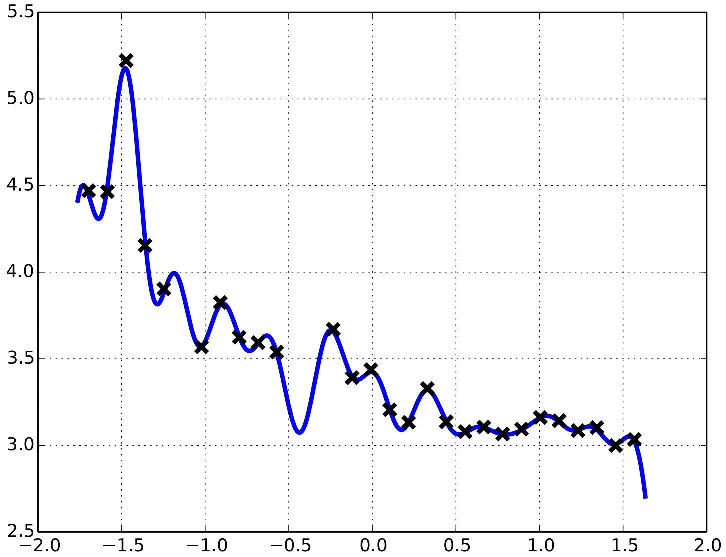
Curve fitting

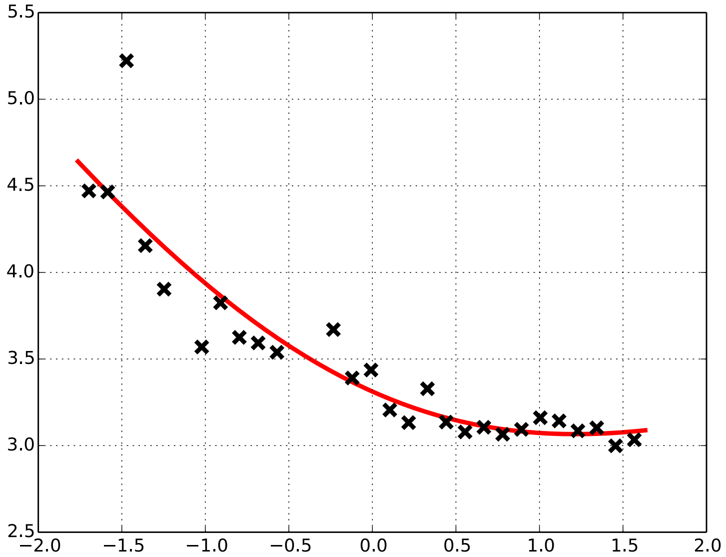


- Which curve fits the data better?
- Which curve is more "complex"?
- Which curve is better overall?

Curve fitting



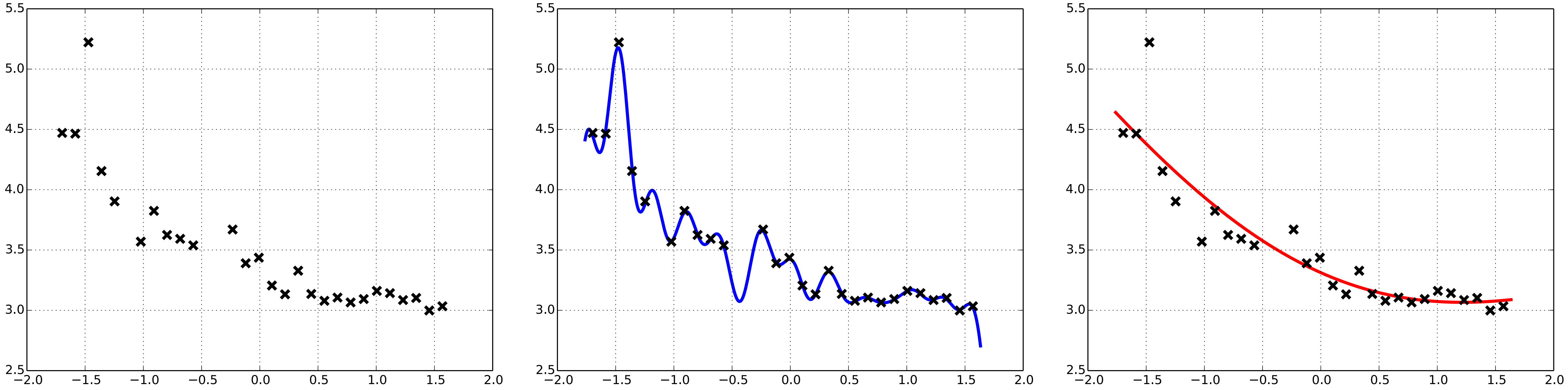




- Which curve fits the data better?
- Which curve is more "complex"?
- Which curve is better overall?

- Assumptions
- Occam's razor

Curve fitting



- Which curve fits the data better?
- Which curve is more "complex"?
- Which curve is better overall?

- Assumptions
- Occam's razor

Gaussian process

Posterior probability

- Posterior inference over space of **functions**

$$\textit{posterior} \propto \textcolor{red}{\textit{likelihood}} \times \textcolor{blue}{\textit{prior}}$$

Signal from observed data

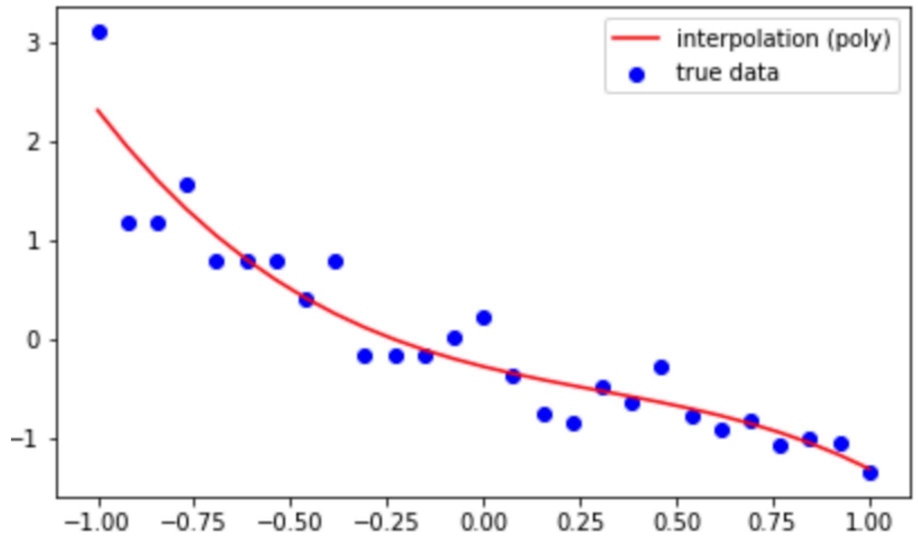
prior assumptions

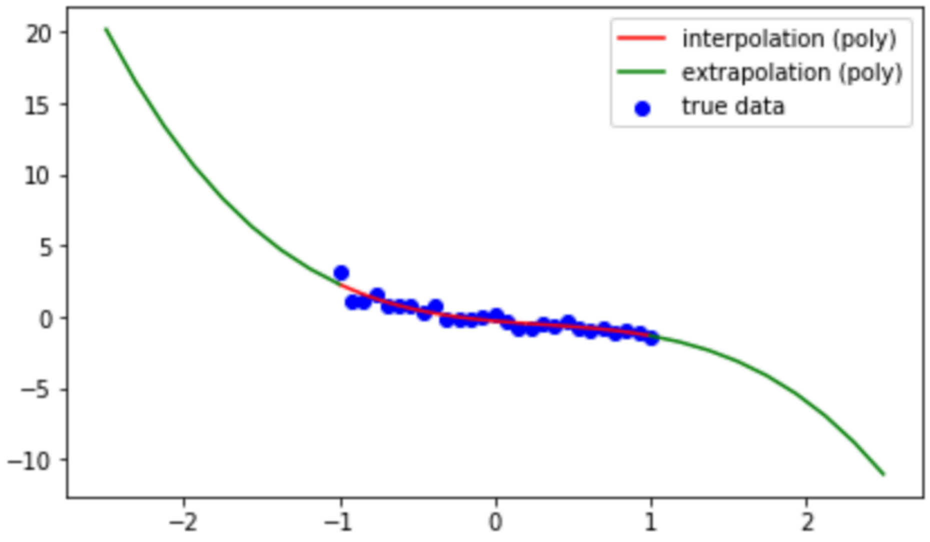
Part 1: Gaussian processes

See also:

adamian.github.io/talks/Damianou_GP_tutorial.html

Polynomial Regression

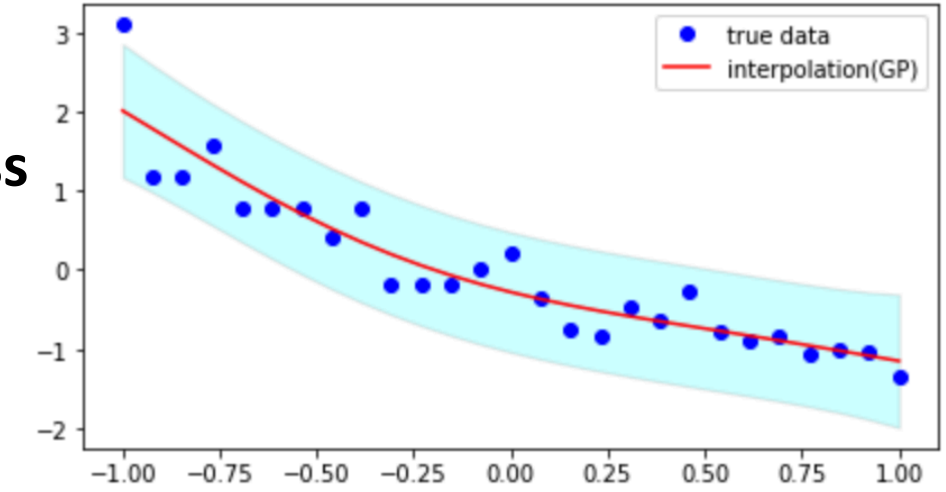


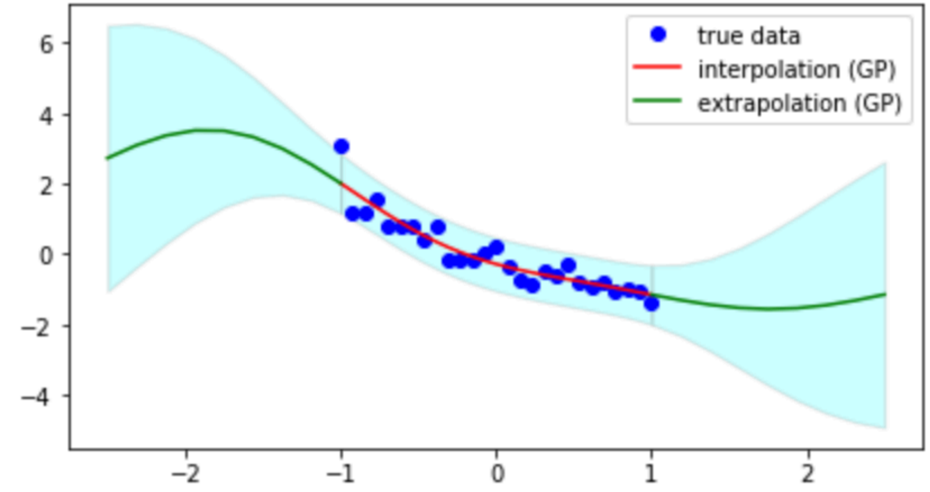


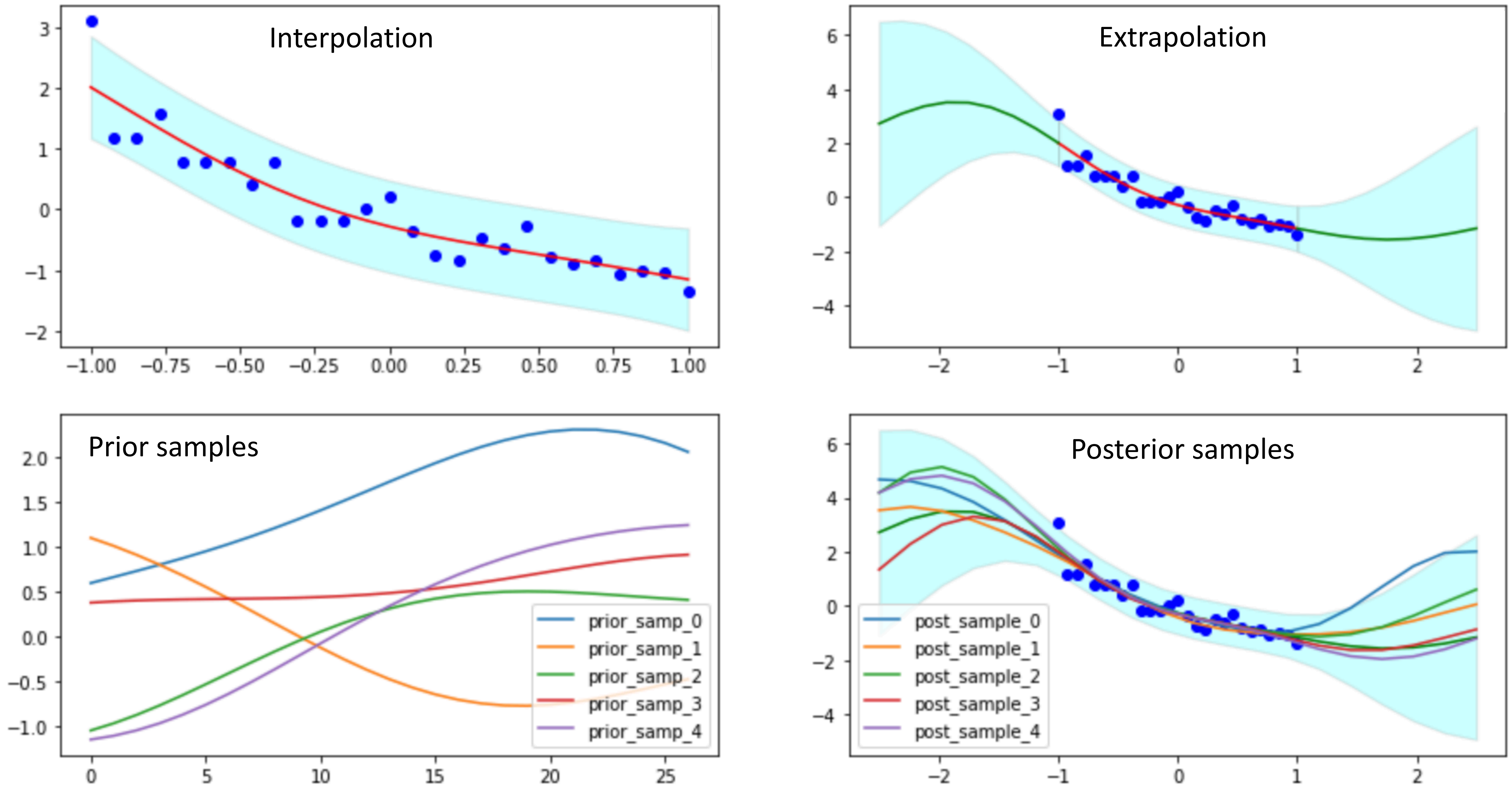
Interpolation

Extrapolation

Gaussian Process Regression

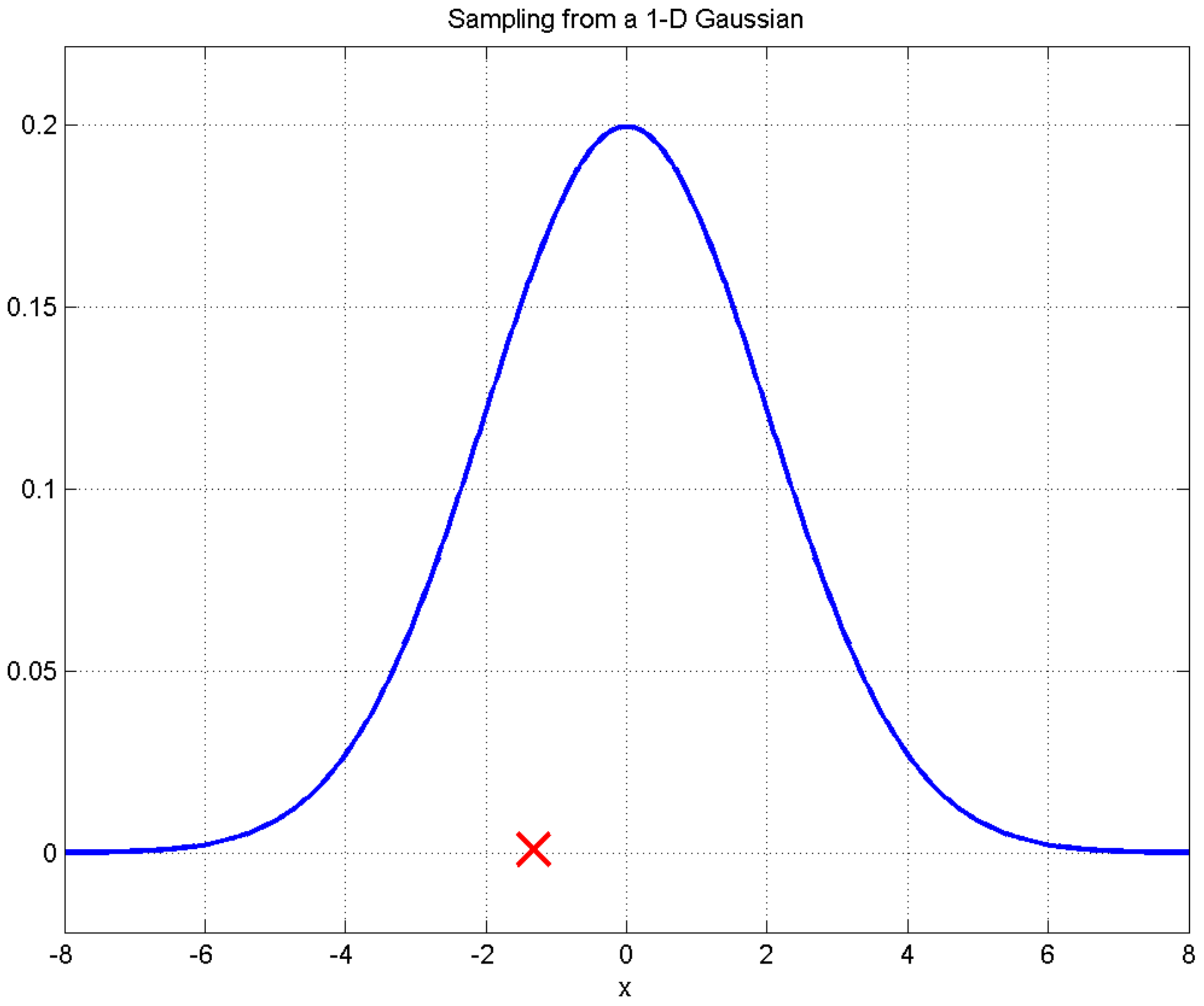


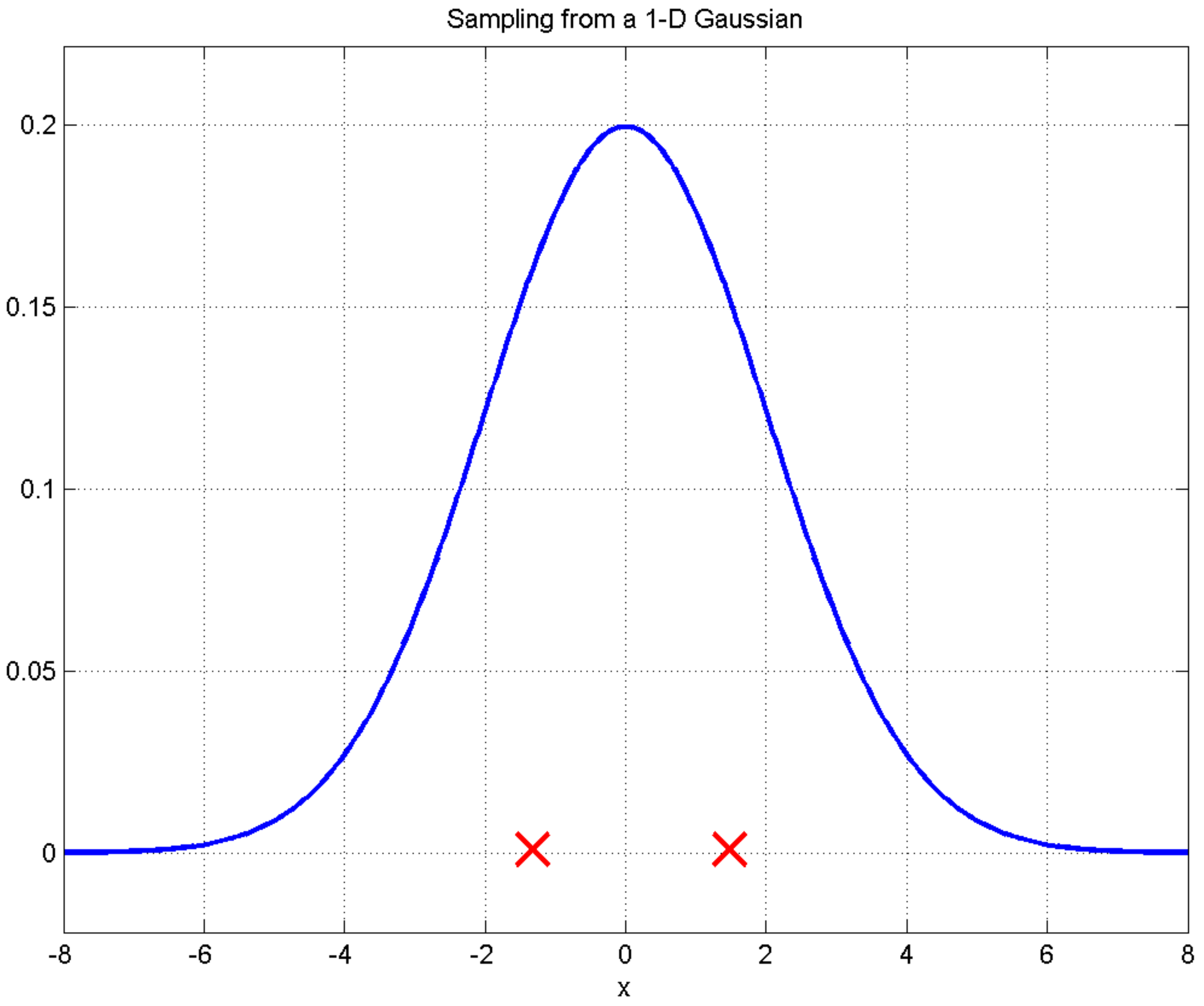


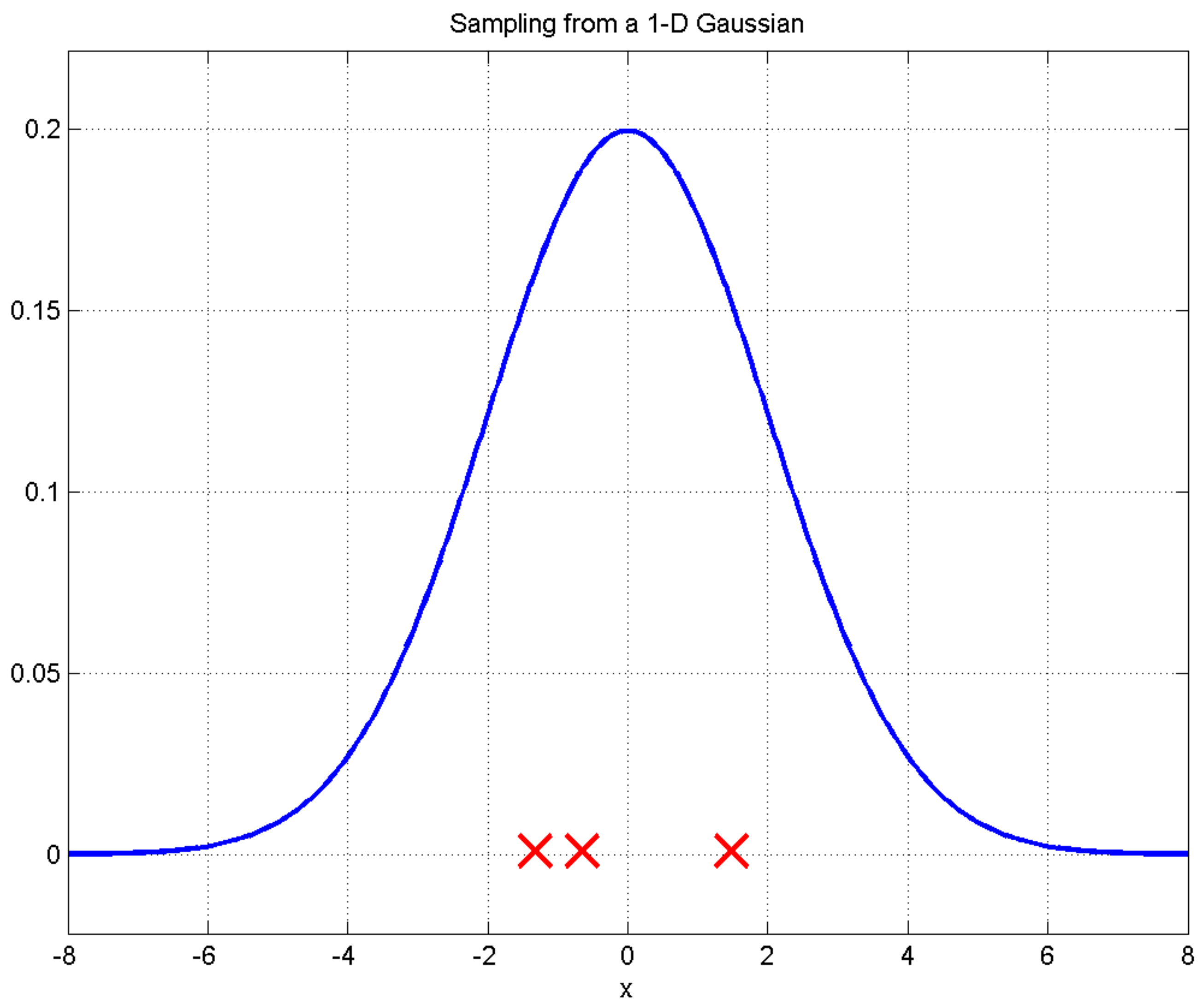

Interpolation
Extrapolation
Prior samples
prior_samp_0
prior_samp_1
prior_samp_2
prior_samp_3
prior_samp_4
Posterior samples
post_sample_0
post_sample_1
post_sample_2
post_sample_3
post_sample_4

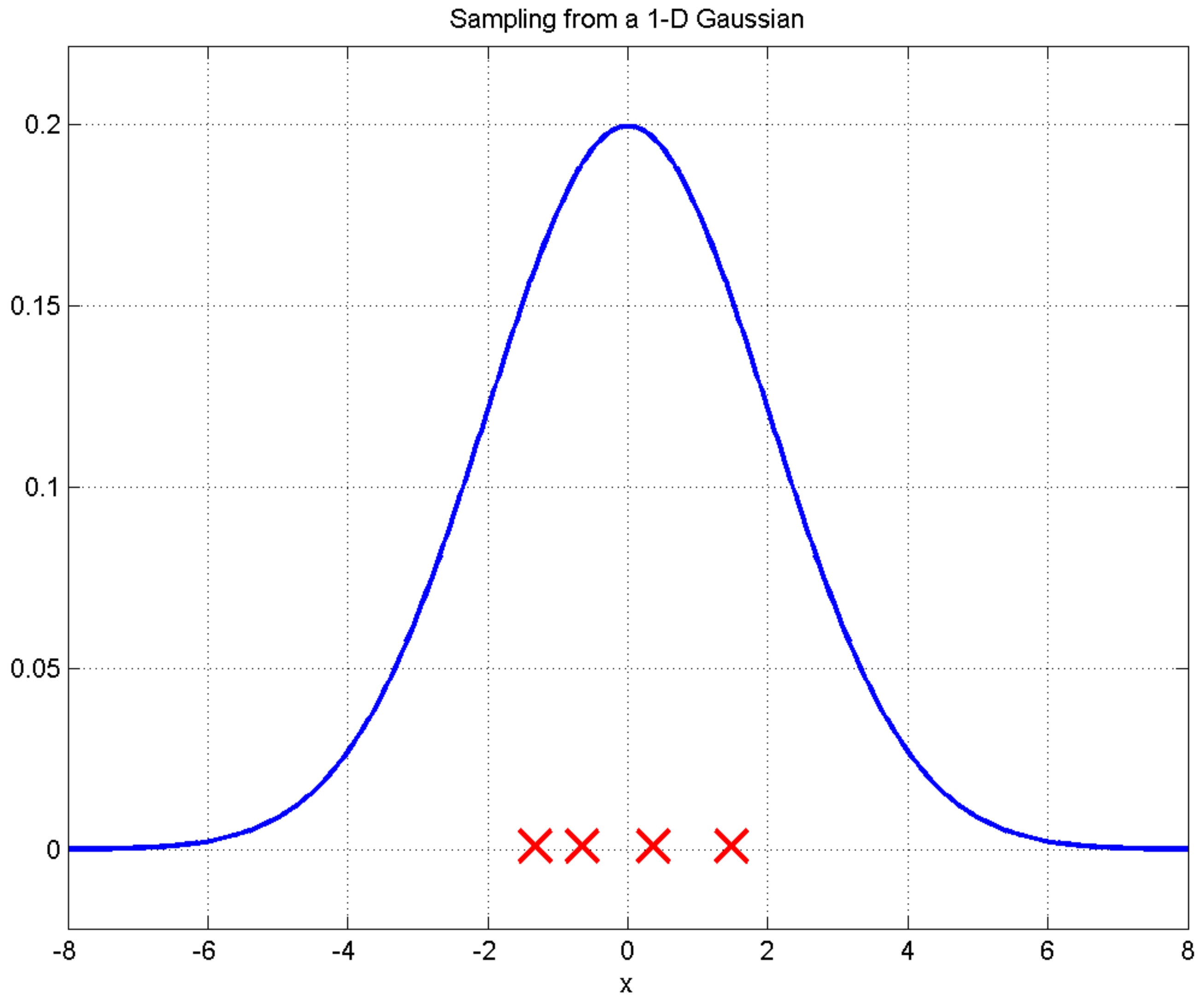
Introducing Gaussian Processes:

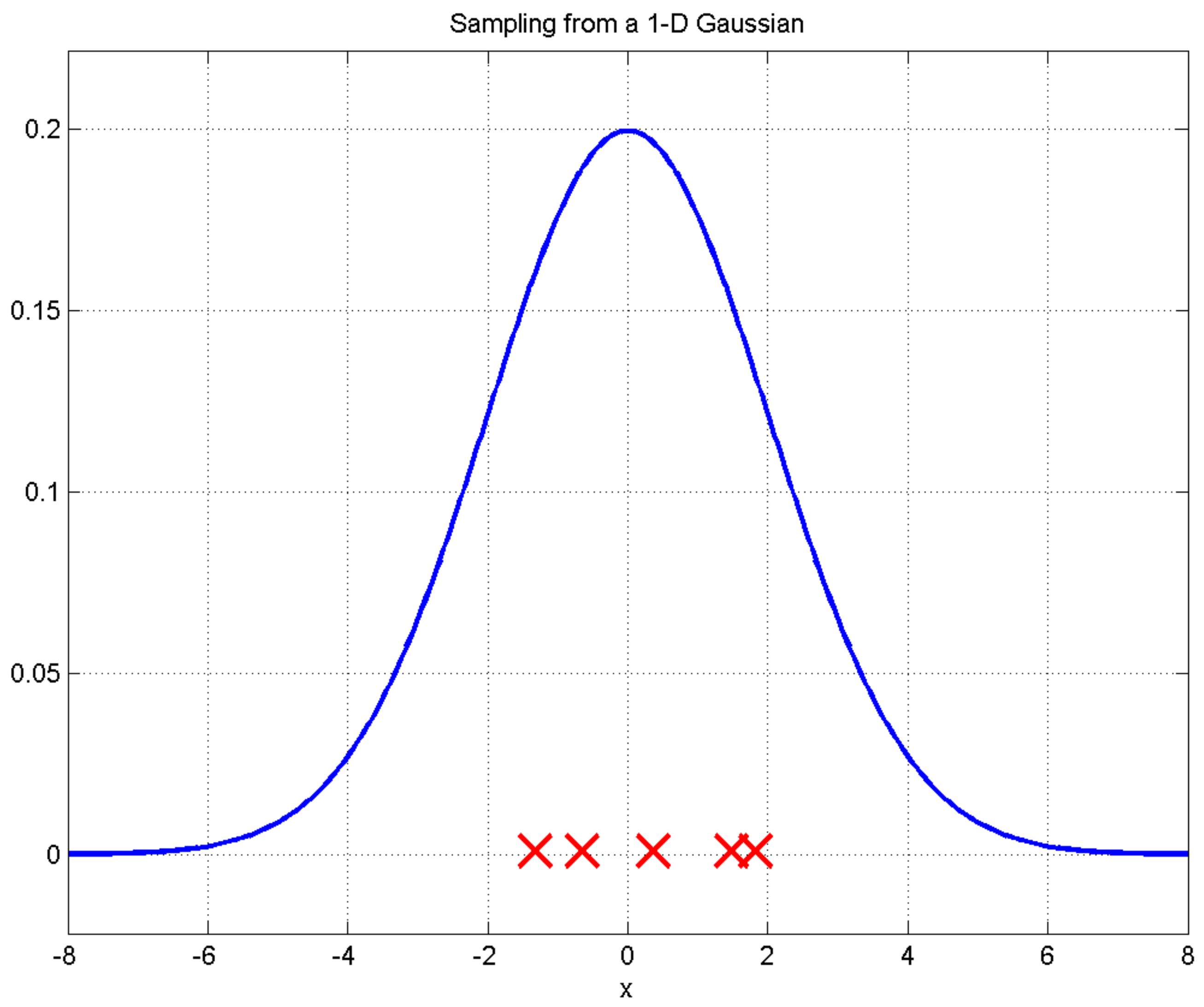
- A Gaussian distribution depends on a mean and a covariance matrix.
- A Gaussian process depends on a mean and a covariance function.


Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

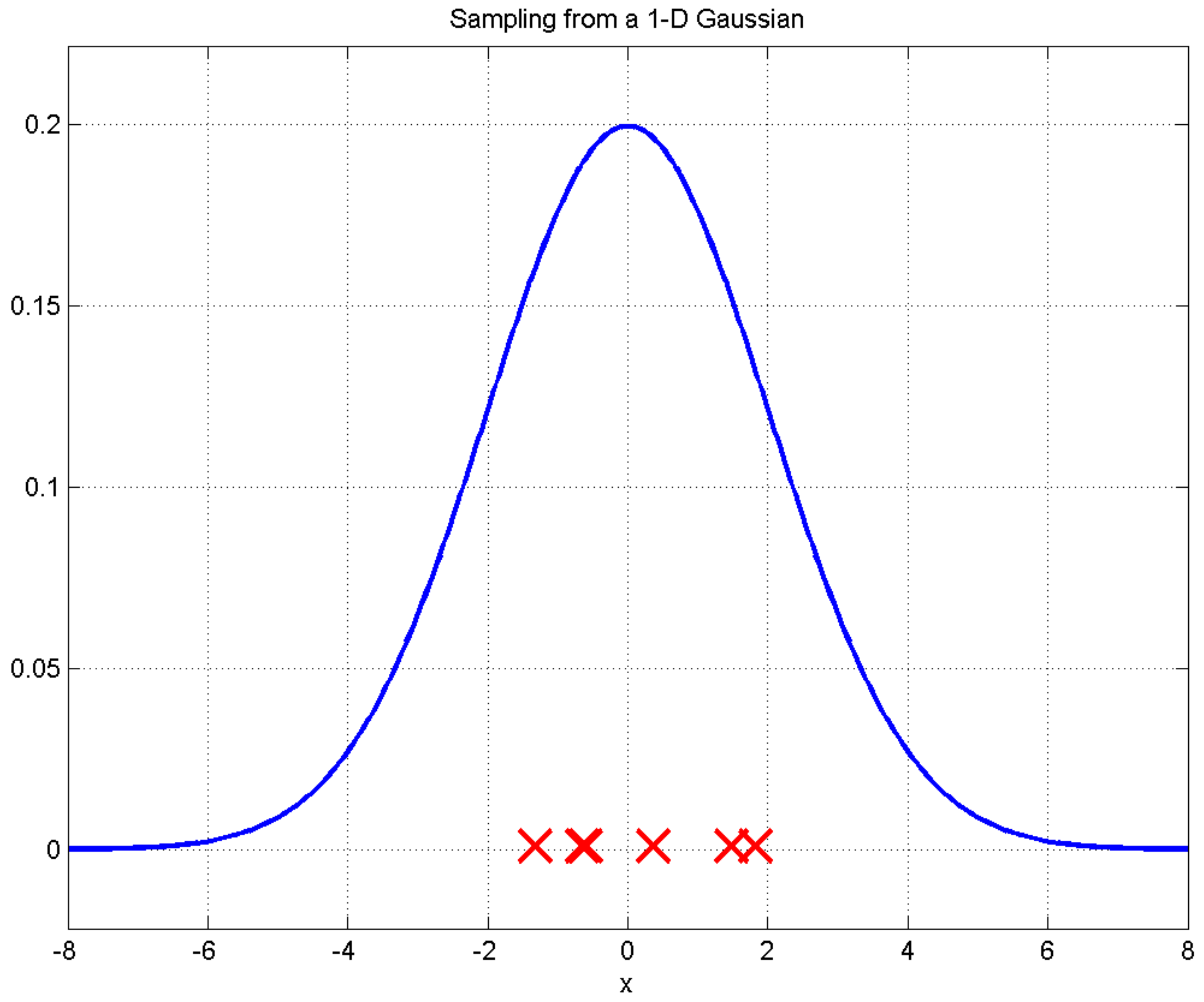

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

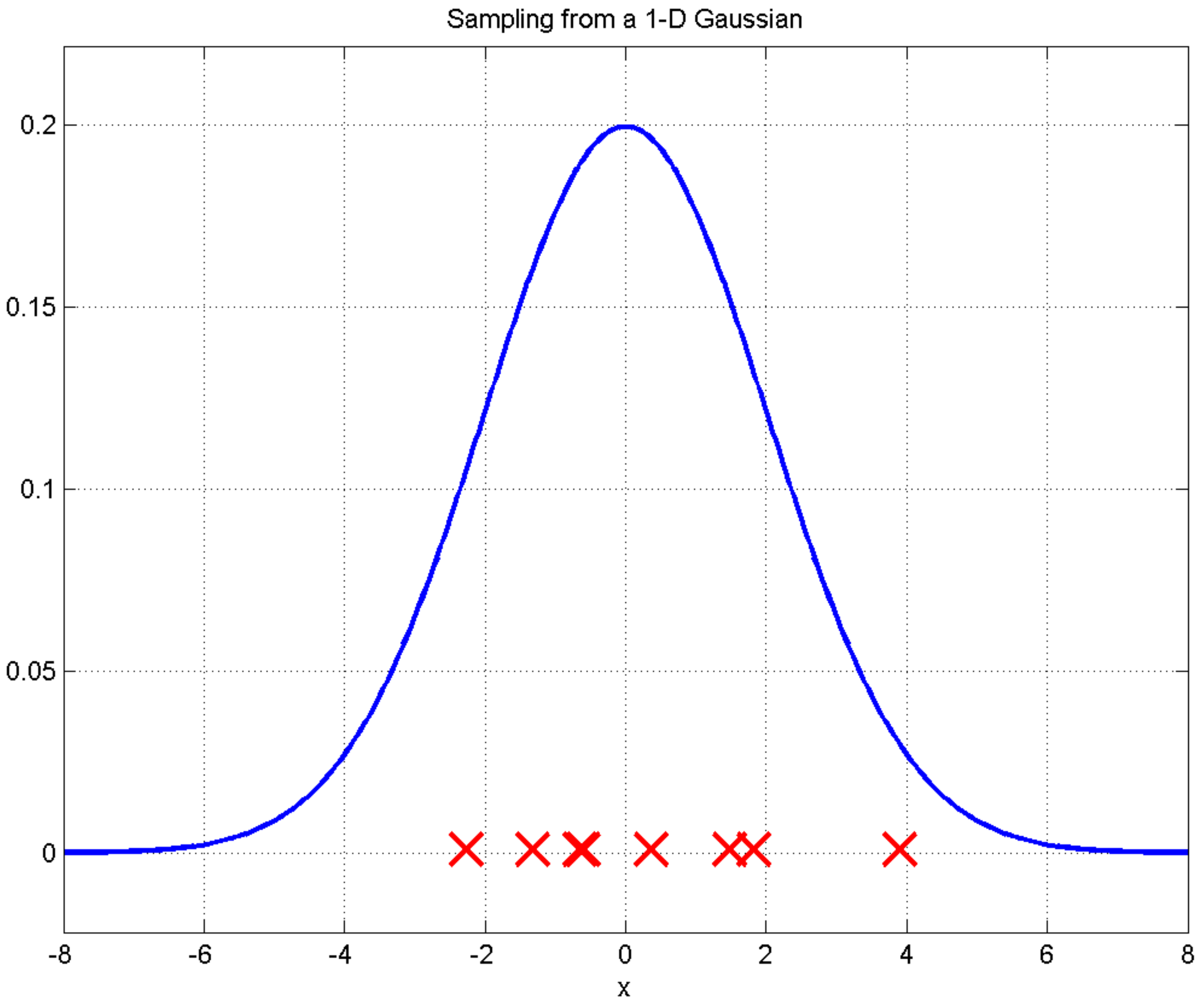

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x


Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

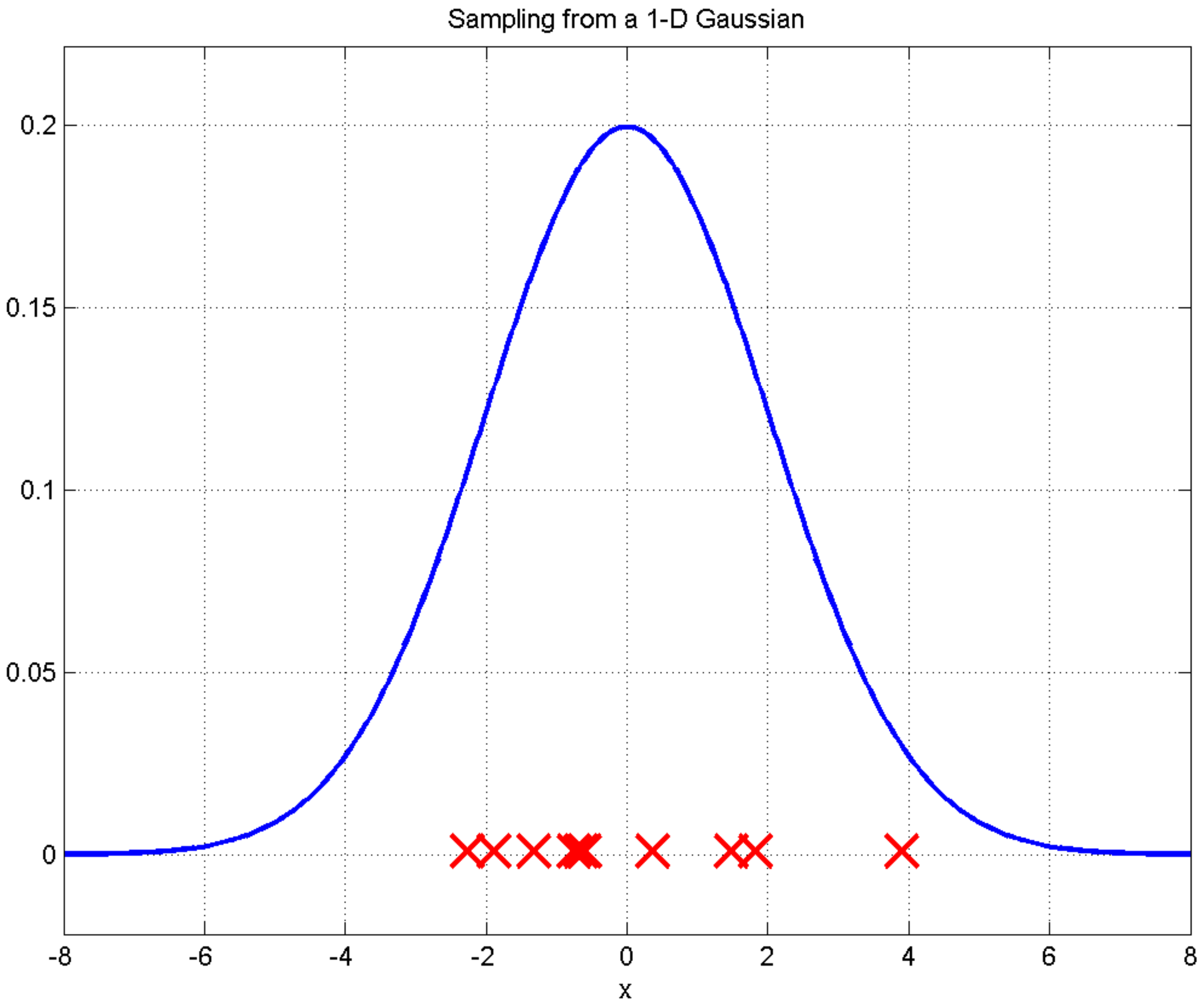


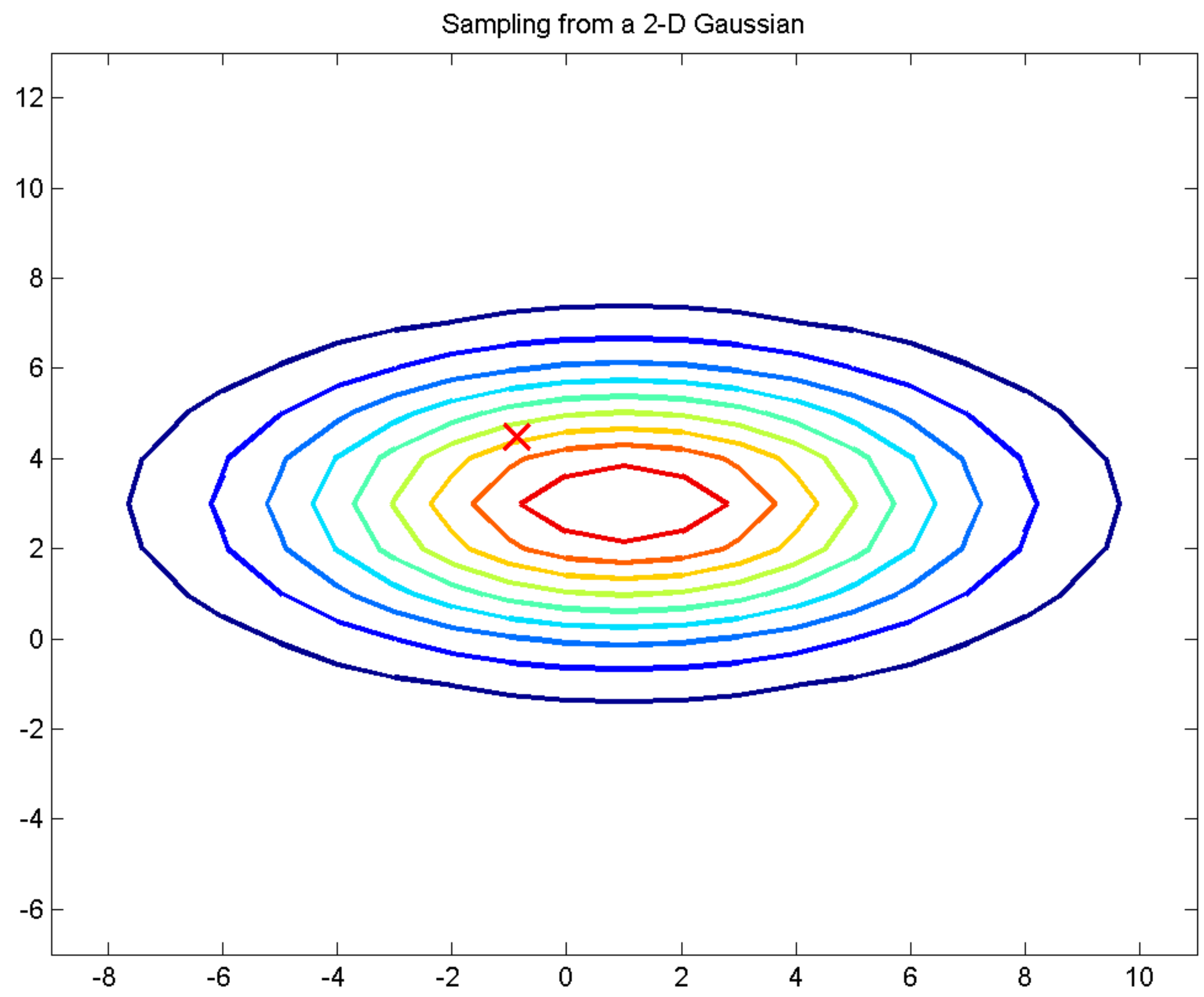
Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

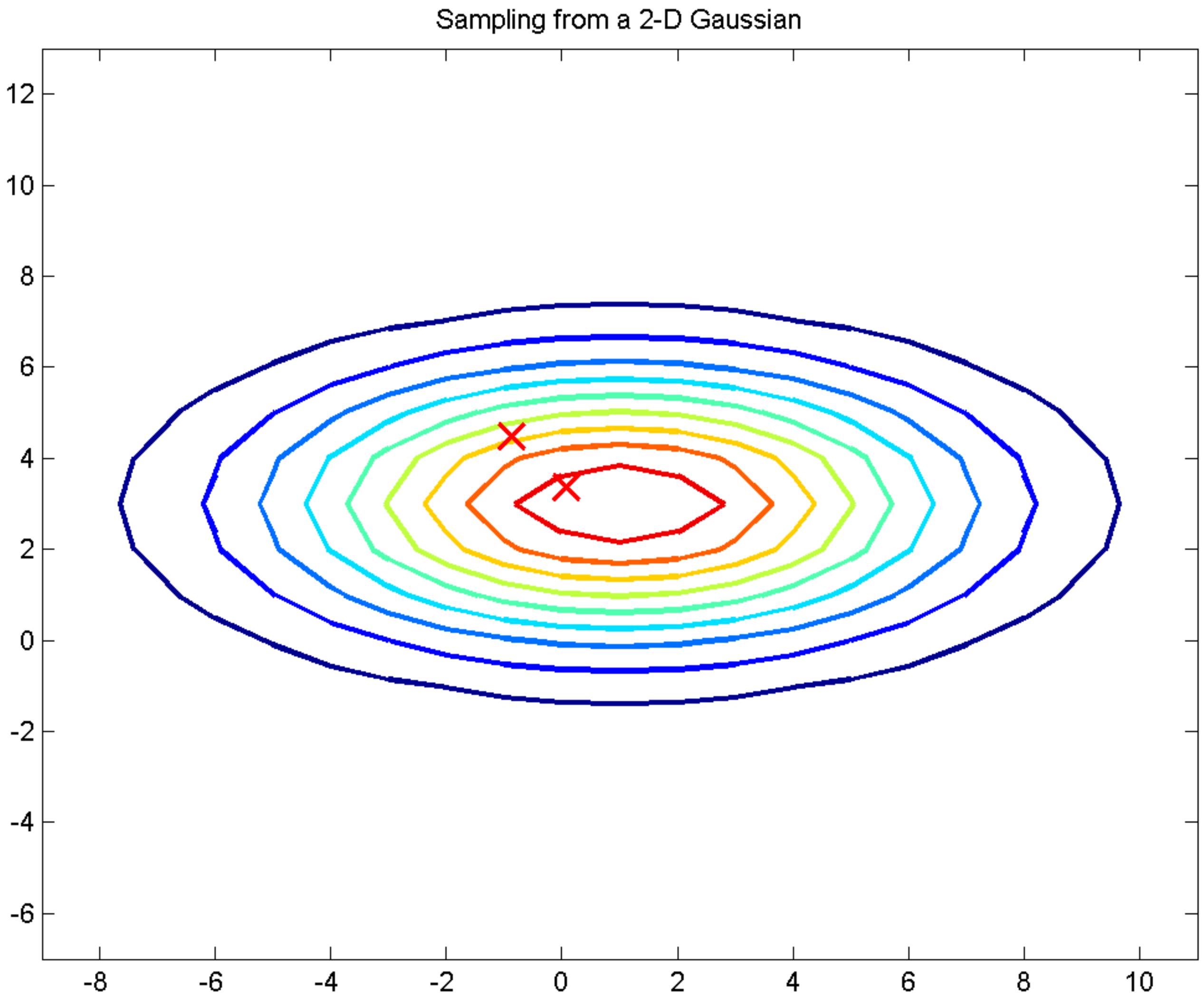

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

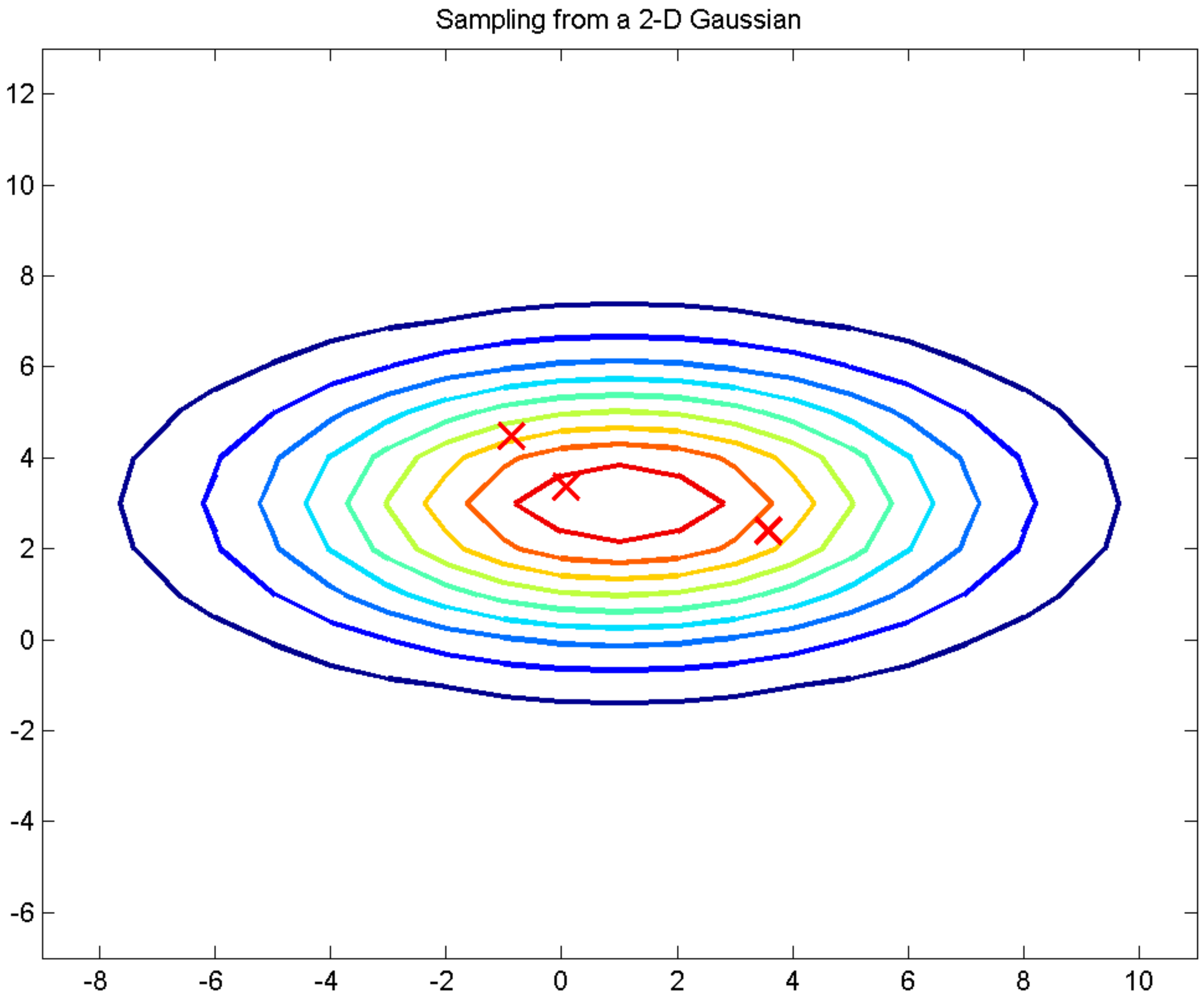


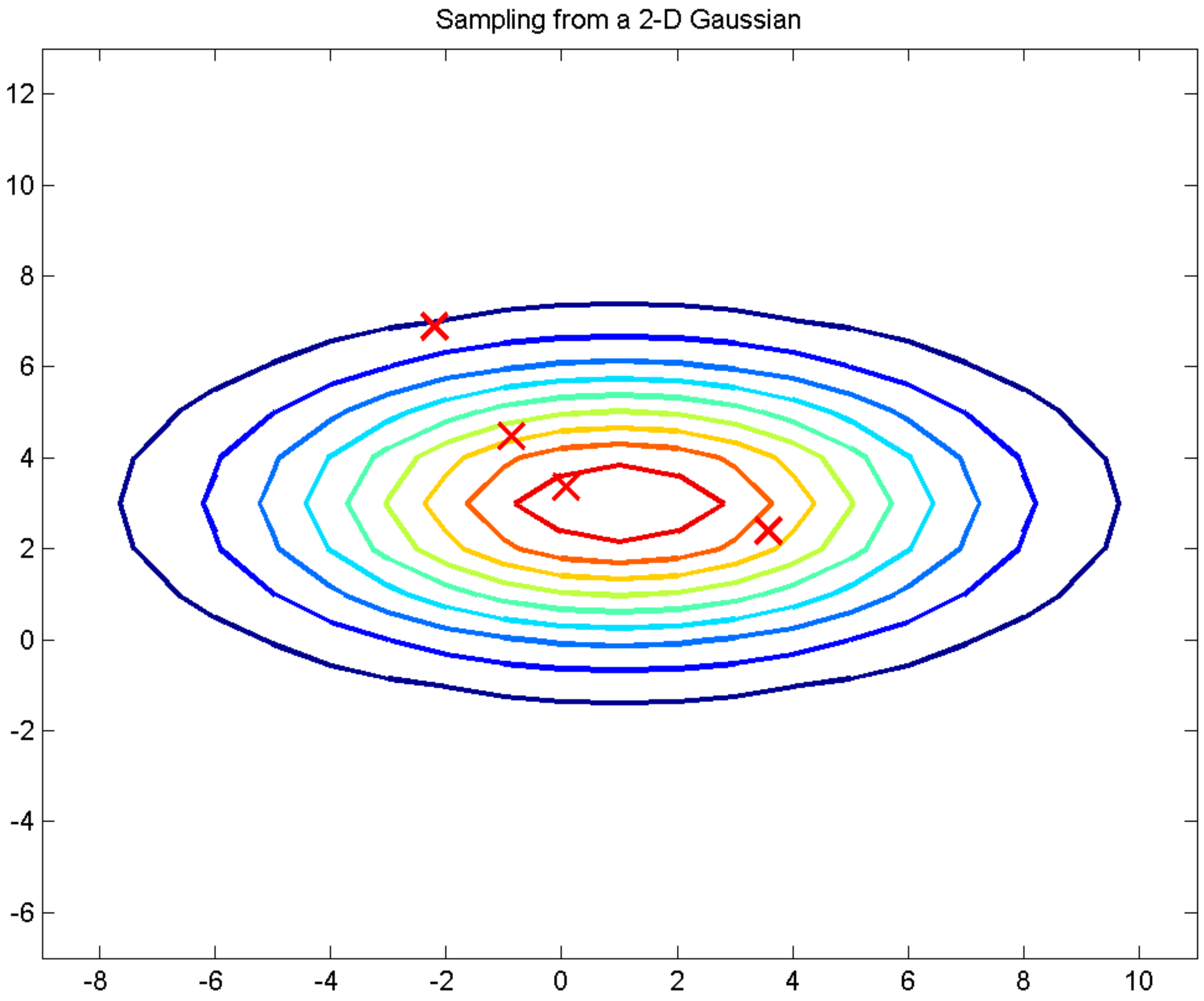
Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

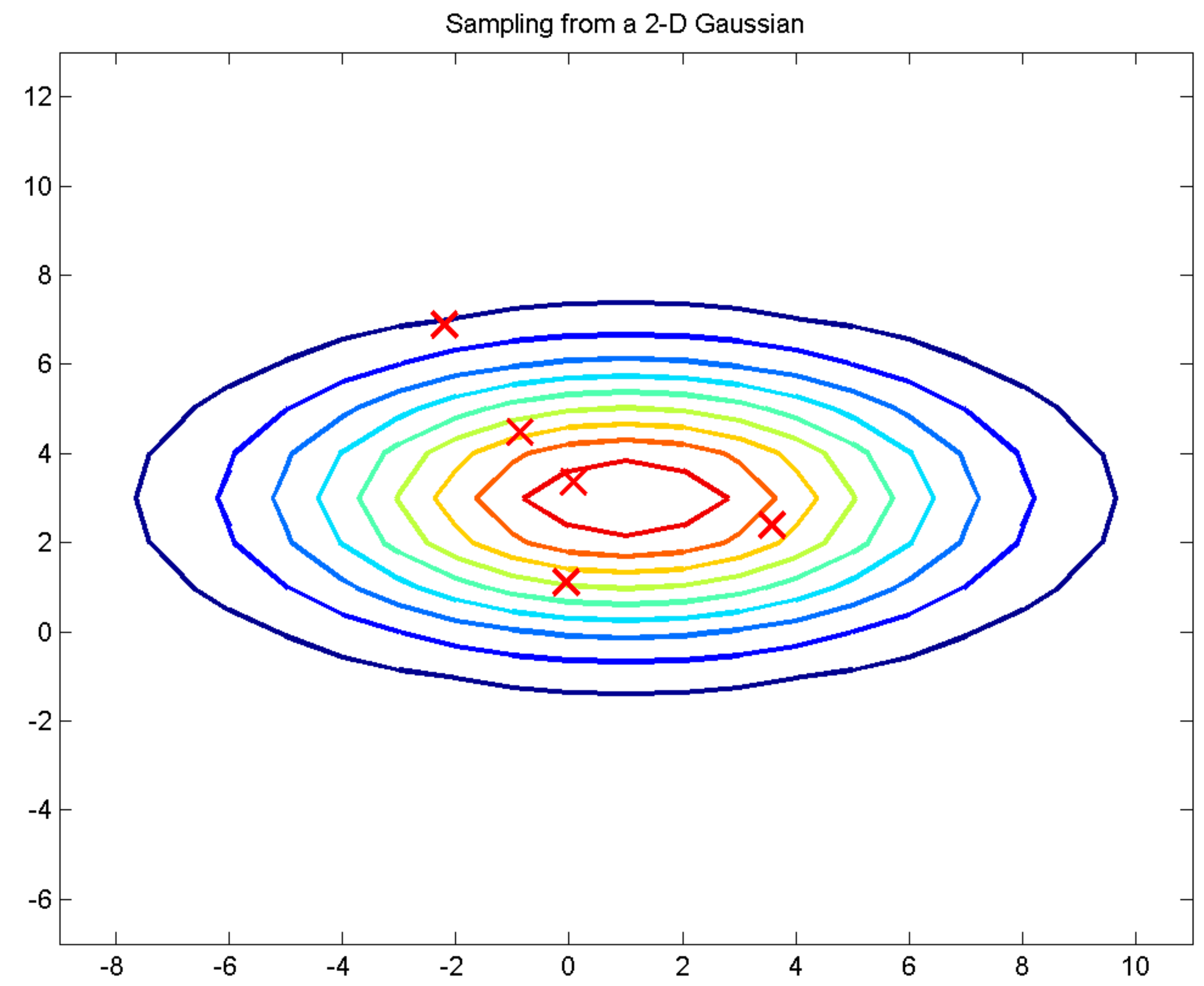

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

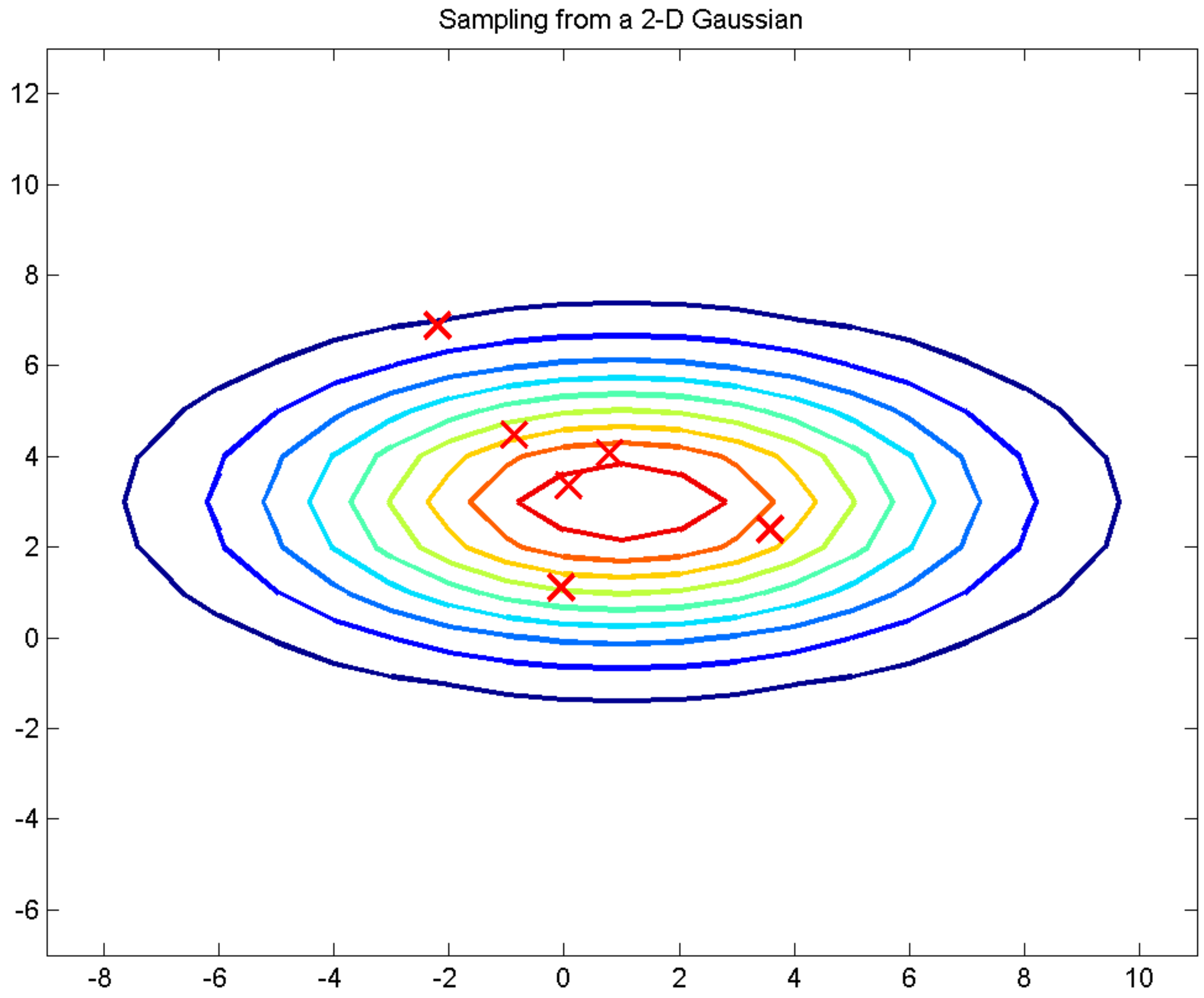

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10


Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

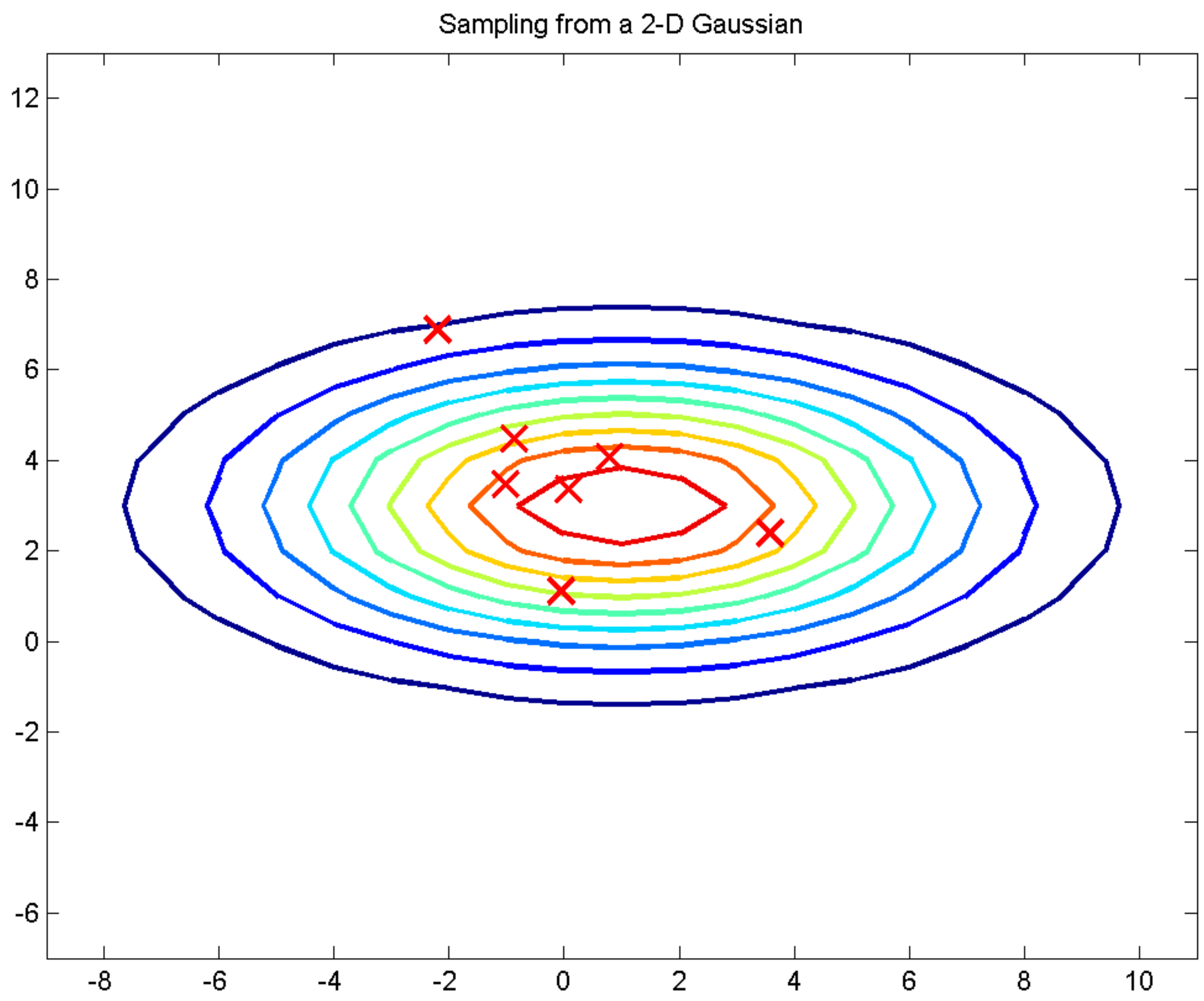

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

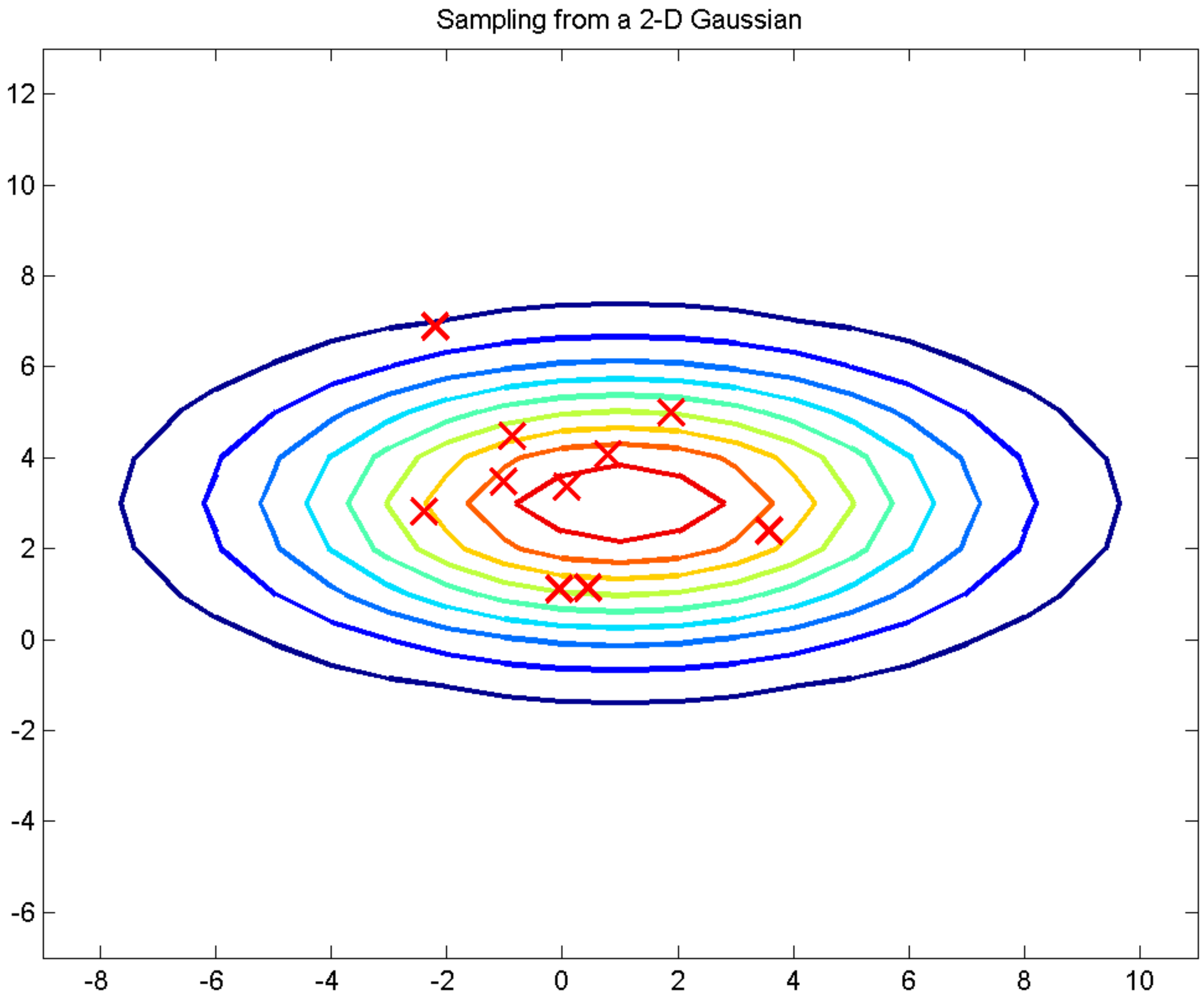

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

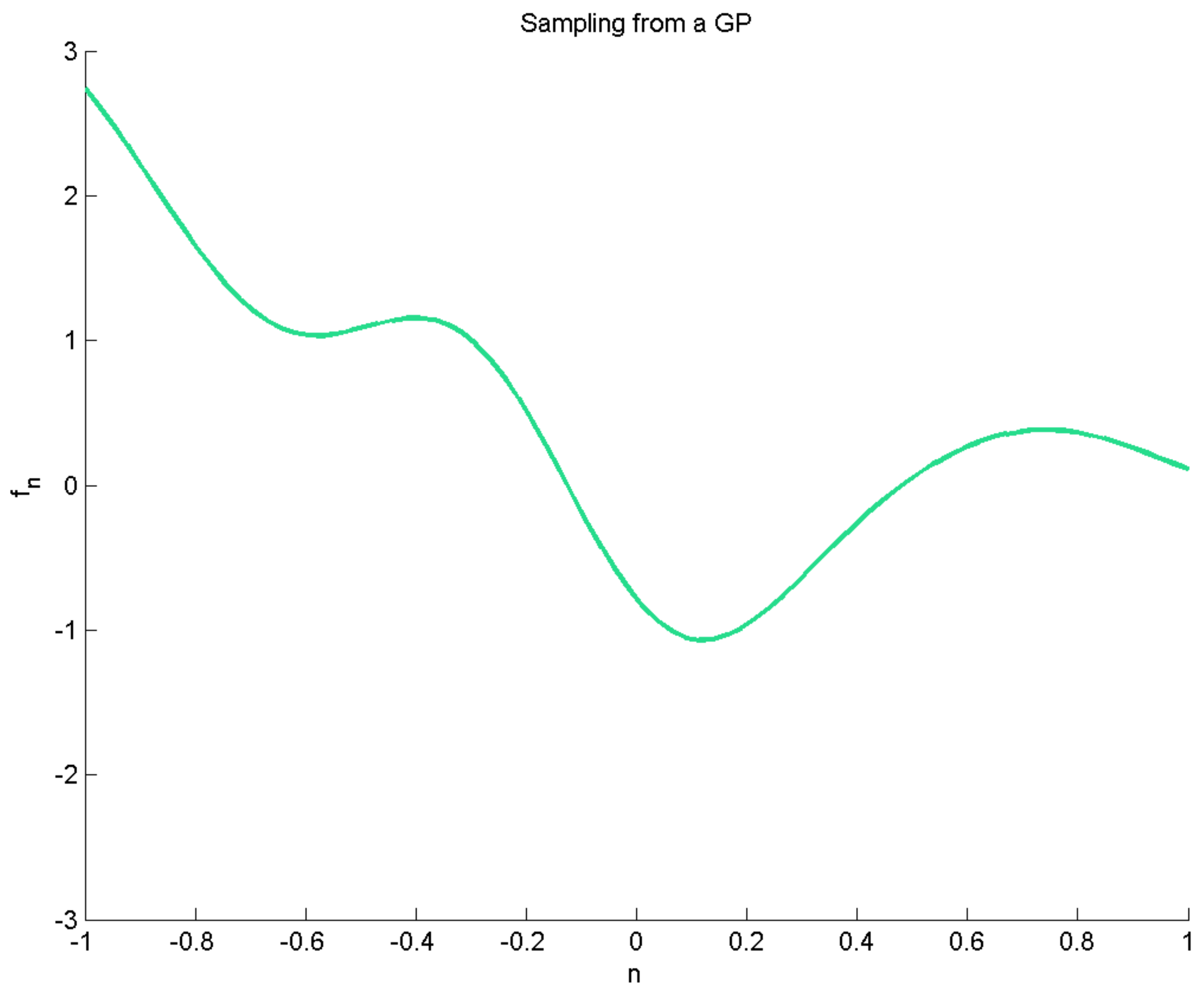

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10



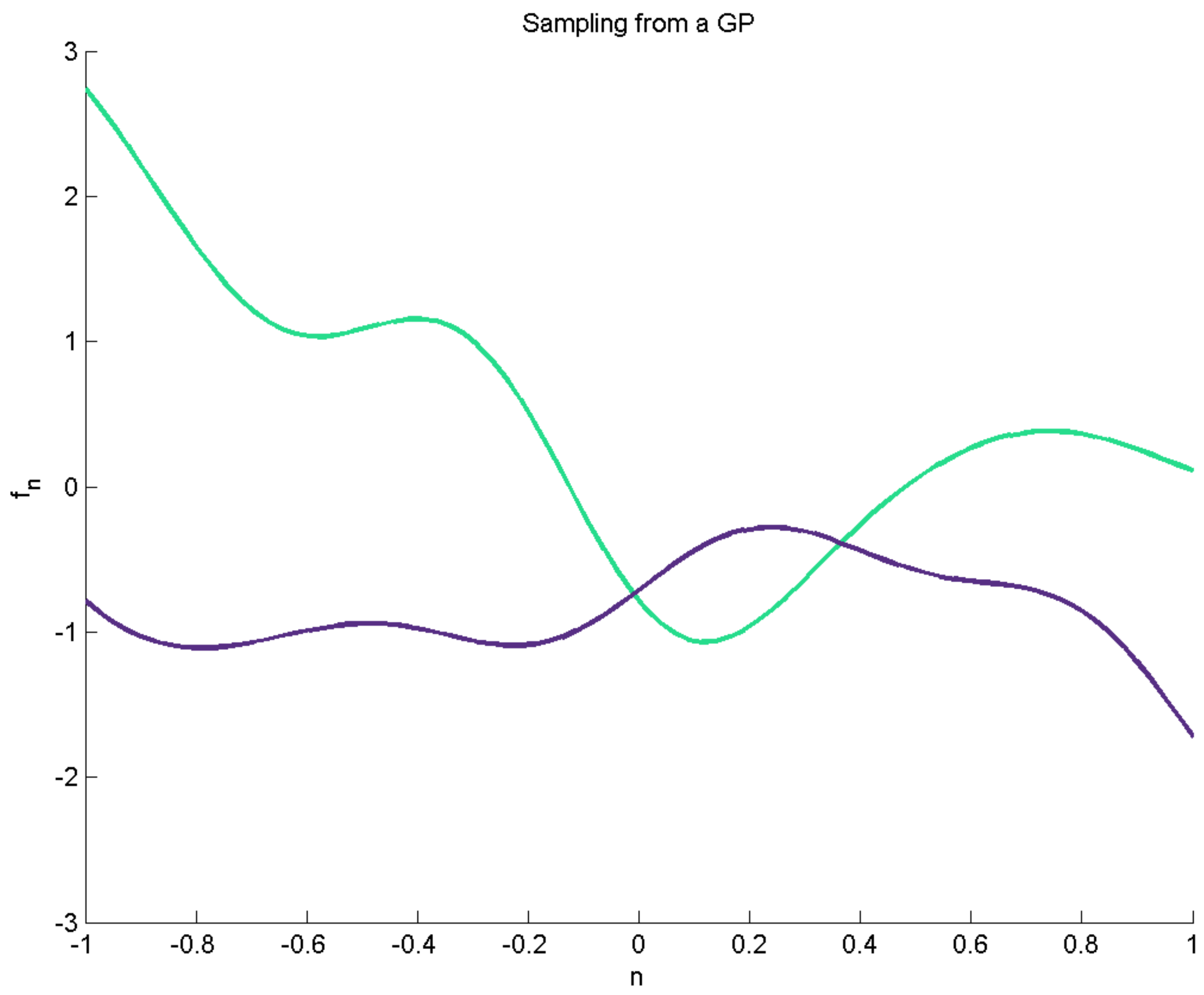
Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

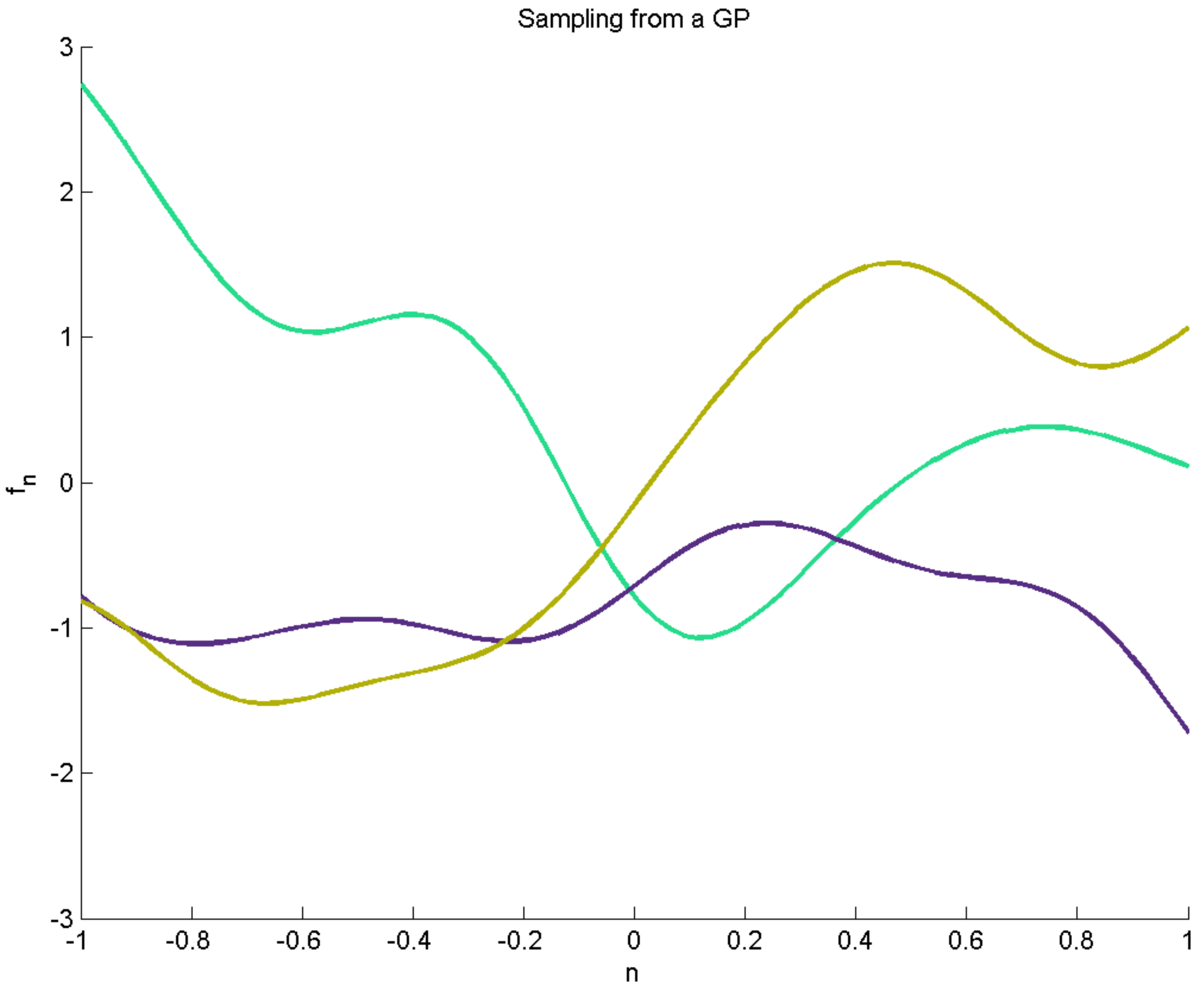

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

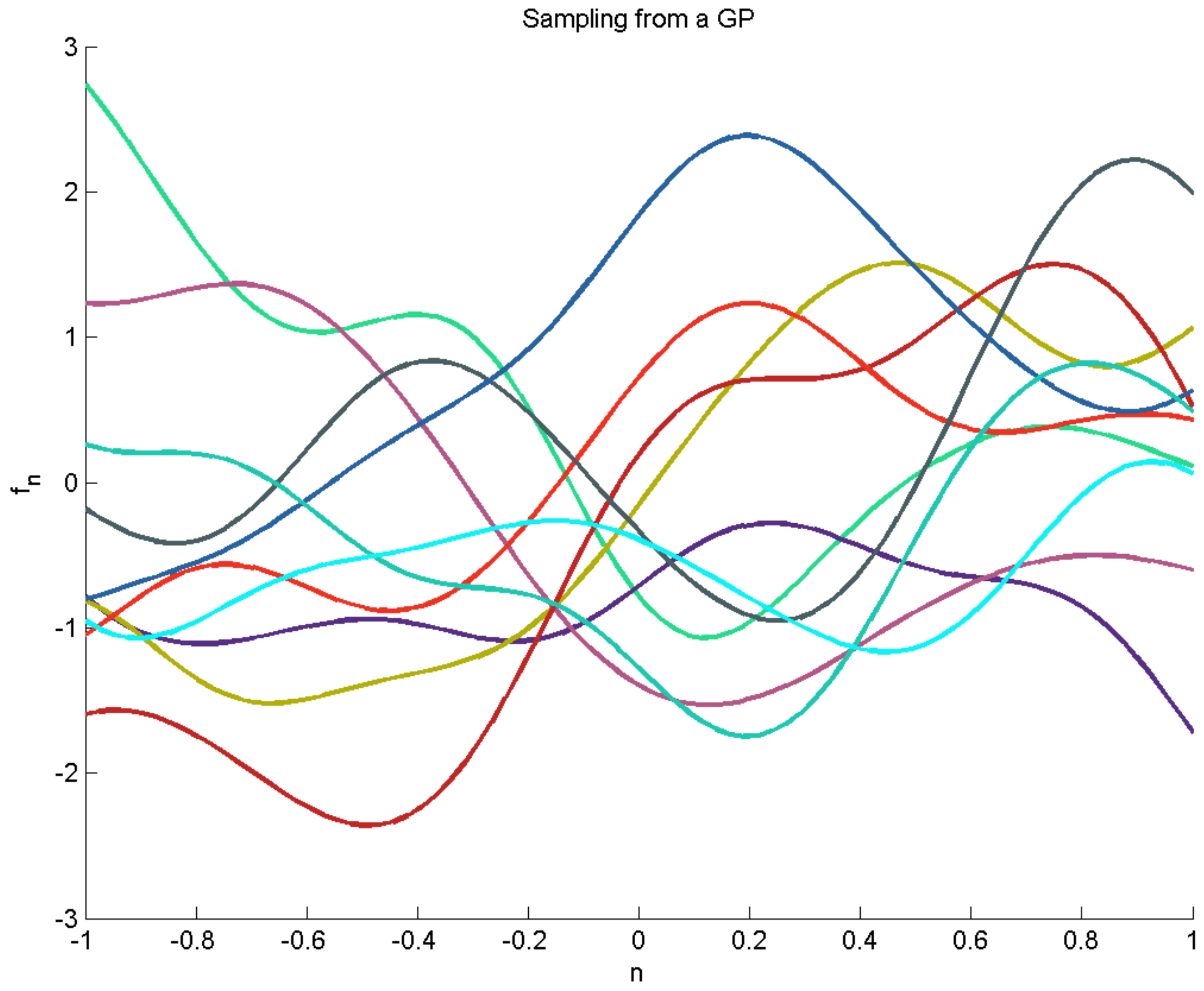

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10



Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n


Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n


Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n


Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n

Infinite model... but we *always* work with finite sets!

Let's start with a multivariate Gaussian:

$$p(\underbrace{f_1, f_2, \cdots, f_s}_{\mathbf{f}_A}, \underbrace{f_{s+1}, f_{s+2}, \cdots, f_N}_{\mathbf{f}_B}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}).$$

with:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{bmatrix} \text{ and } \mathbf{K} = \begin{bmatrix} \mathbf{K}_{AA} & \mathbf{K}_{AB} \\ \mathbf{K}_{BA} & \mathbf{K}_{BB} \end{bmatrix}$$

Marginalisation property:

$$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}). \quad \text{Then:}$$

$$p(\mathbf{f}_A) = \int_{\mathbf{f}_B} p(\mathbf{f}_A, \mathbf{f}_B) d\mathbf{f}_B = \mathcal{N}(\boldsymbol{\mu}_A, \mathbf{K}_{AA})$$

Infinite model... but we *always* work with finite sets!

Let's start with a multivariate Gaussian:

$$p(\underbrace{f_1, f_2, \cdots, f_s}_{\mathbf{f}_A}, \underbrace{f_{s+1}, f_{s+2}, \cdots, f_N}_{\mathbf{f}_B}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}).$$

with:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{bmatrix} \text{ and } \mathbf{K} = \begin{bmatrix} \mathbf{K}_{AA} & \mathbf{K}_{AB} \\ \mathbf{K}_{BA} & \mathbf{K}_{BB} \end{bmatrix}$$

Marginalisation property:

$$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}). \quad \text{Then:}$$

$$p(\mathbf{f}_A) = \int_{\mathbf{f}_B} p(\mathbf{f}_A, \mathbf{f}_B) \mathrm{d}\mathbf{f}_B = \mathcal{N}(\boldsymbol{\mu}_A, \mathbf{K}_{AA})$$

Infinite model... but we *always* work with finite sets!

In the GP context $\mathbf{f} = f(x)$:

$$\boldsymbol{\mu}_{\infty} = \begin{bmatrix} \boldsymbol{\mu}_{\mathbf{f}} \\ \cdots \\ \cdots \end{bmatrix} \quad \text{and} \quad \mathbf{K}_{\infty} = \begin{bmatrix} \mathbf{K}_{\mathbf{ff}} & \cdots \\ \cdots & \cdots \end{bmatrix}$$

Infinite model... but we *always* work with finite sets!

In the GP context $\mathbf{f} = f(x)$:

$$\boldsymbol{\mu}_\infty = \begin{bmatrix} \boldsymbol{\mu}_\mathbf{f} \\ \cdots \\ \cdots \end{bmatrix} \text{ and } \mathbf{K}_\infty = \begin{bmatrix} \mathbf{K}_\mathbf{ff} & \cdots \\ \cdots & \cdots \end{bmatrix}$$

Covariance function: Maps locations x_i, x_j of the input domain $\mathcal{X}$ to an entry in the covariance matrix:

$$\mathbf{K}_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$$

For all available inputs:

$$\mathbf{K} = \mathbf{K}_\mathbf{ff} = k(\mathbf{X}, \mathbf{X})$$

GP: joint Gaussian distribution of the training and the (potentially infinite!) test data:

$\mathbf{f}^* = f(\mathbf{x}^*)$

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^\top & \mathbf{K}_{*,*} \end{bmatrix}\right)$$

GP: joint Gaussian distribution of the training and the (potentially infinite!) test data:

$\mathbf{f}^* = f(\mathbf{x}^*)$

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^\top & \mathbf{K}_{*,*} \end{bmatrix}\right)$$

$\mathbf{K}_*$ is the (cross)-covariance matrix obtained by evaluating the covariance function in pairs of training inputs $\mathbf{X}$ and test inputs $\mathbf{X}_*$, ie.

$$\mathbf{f}_* = k(\mathbf{X}, \mathbf{X}_*).$$

Similarly:

$$\mathbf{K}_{**} = k(\mathbf{X}_*, \mathbf{X}_*).$$

Posterior is also Gaussian!

$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}).$ Then:

$$p(\mathbf{f}_A|\mathbf{f}_B) = \mathcal{N}(\boldsymbol{\mu}_A + \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}(\mathbf{f}_B - \boldsymbol{\mu}_B), \mathbf{K}_{AA} - \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}\mathbf{K}_{BA})$$

In the GP context this can be used for inter/extrapolation:

$$p(f_*|f_1, \cdots, f_N) = p(f(x_*)|f(x_1), \cdots, f(x_N)) \sim \mathcal{N}$$

$$p(\mathbf{f}_*|\mathbf{f_1}, \cdots, \mathbf{f_N}) = p(f(x_*)|f(x_1), \cdots, f(x_N))$$

$$\sim \mathcal{N}(\mathbf{K}_*^\top \mathbf{K}^{-1}\mathbf{f} \, , \, \mathbf{K}_{*,*} - \mathbf{K}_*^\top \mathbf{K}^{-1}\mathbf{K}_*)$$

$p(f(x_*)|f(x_1), \cdots, f(x_N))$ is a posterior **process**!

Posterior is also Gaussian!

$p(\mathbf{f}_A, \mathbf{f}_B) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$. Then:

$$p(\mathbf{f}_A|\mathbf{f}_B) = \mathcal{N}(\boldsymbol{\mu}_A + \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}(\mathbf{f}_B - \boldsymbol{\mu}_B), \mathbf{K}_{AA} - \mathbf{K}_{AB}\mathbf{K}_{BB}^{-1}\mathbf{K}_{BA})$$

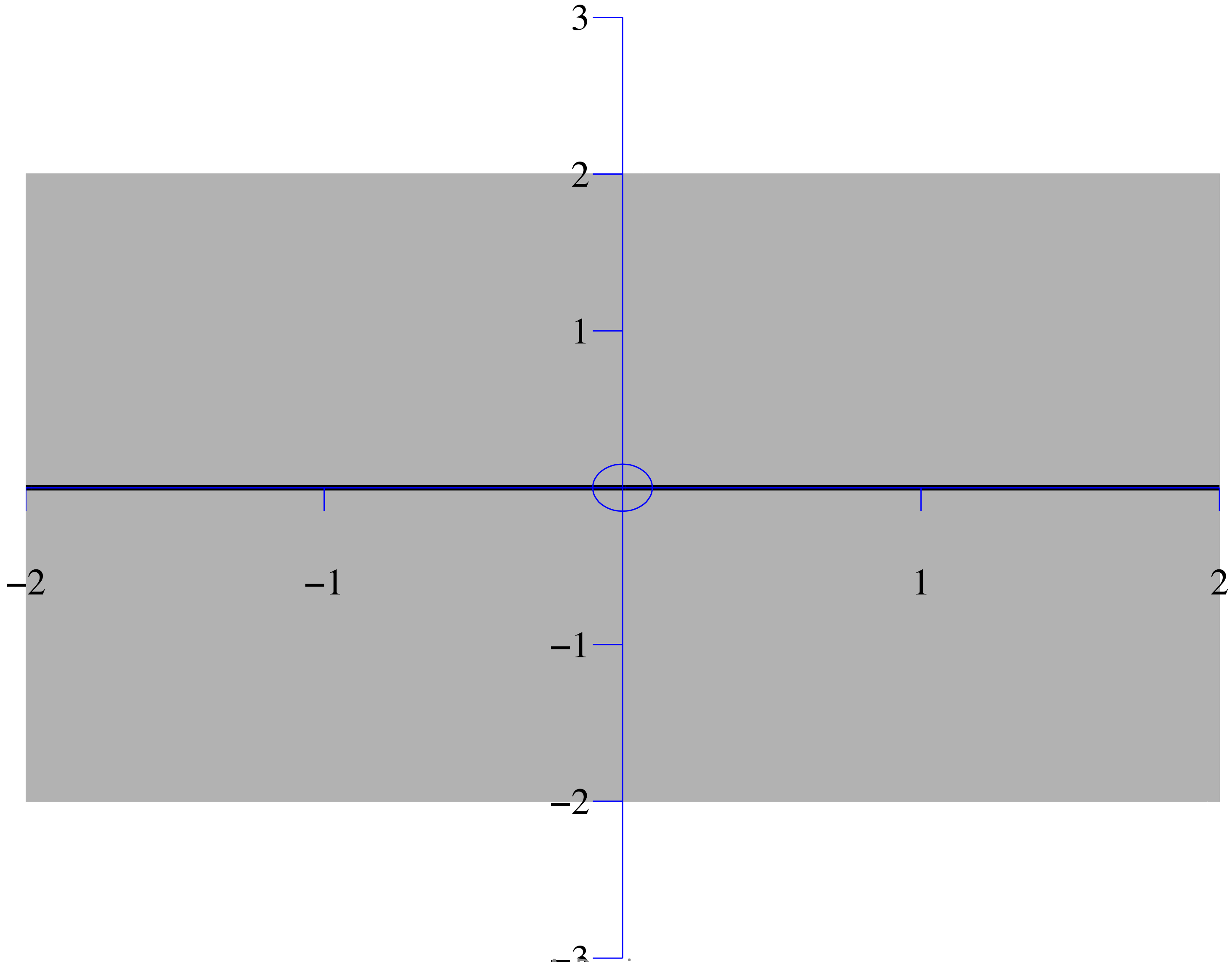
In the GP context this can be used for inter/extrapolation:

$$p(f_*|f_1, \cdots, f_N) = p(f(x_*)|f(x_1), \cdots, f(x_N)) \sim \mathcal{N}$$

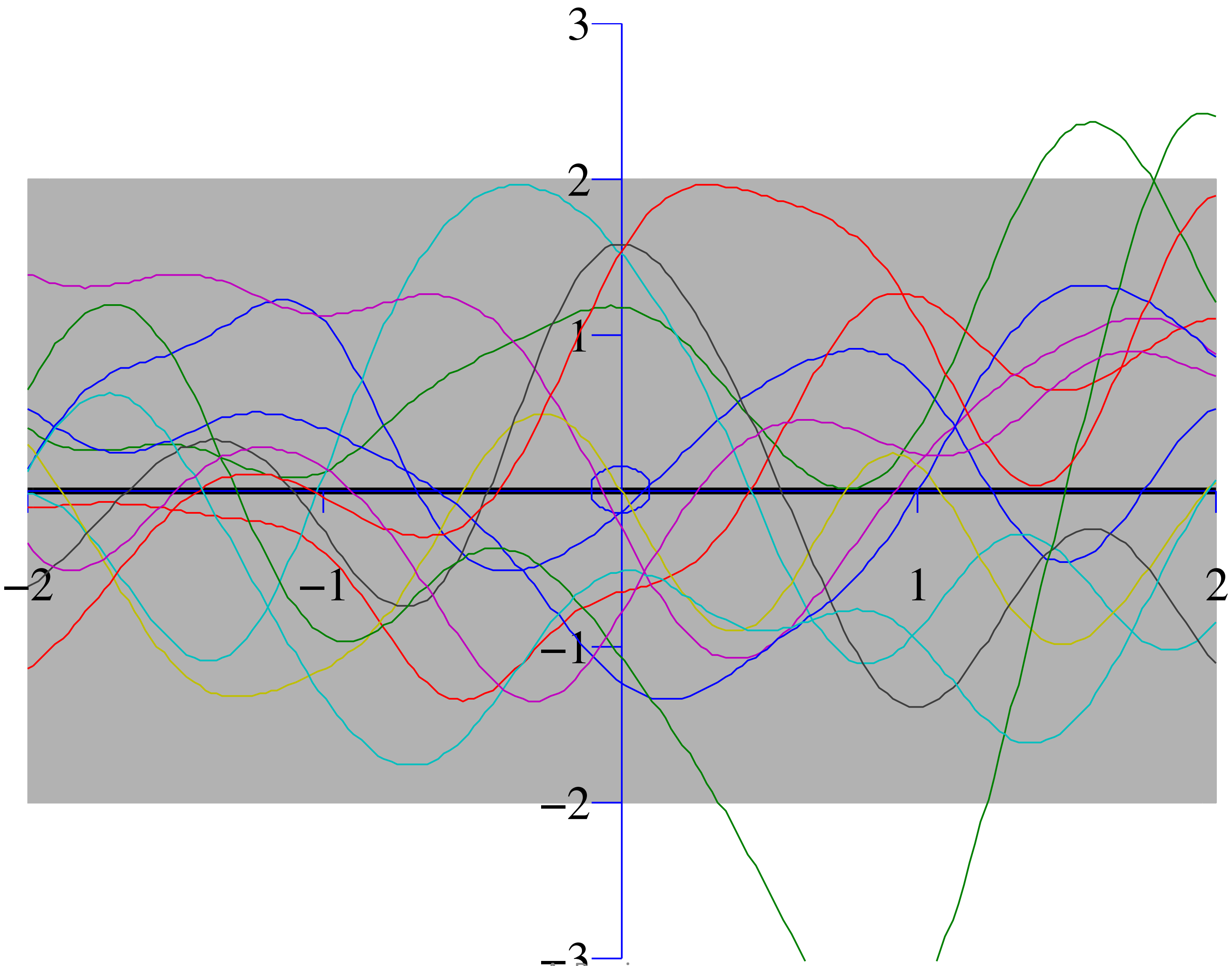
$$\begin{aligned} p(\mathbf{f}_*|\mathbf{f_1}, \cdots, \mathbf{f_N}) &= p(f(x_*)|f(x_1), \cdots, f(x_N)) \\ &\sim \mathcal{N}(\mathbf{K}_*^\top \mathbf{K}^{-1}\mathbf{f}\ ,\ \mathbf{K}_{*,*} - \mathbf{K}_*^\top \mathbf{K}^{-1}\mathbf{K}_*) \end{aligned}$$

$p(f(x_*)|f(x_1), \cdots, f(x_N))$ is a posterior **process**!

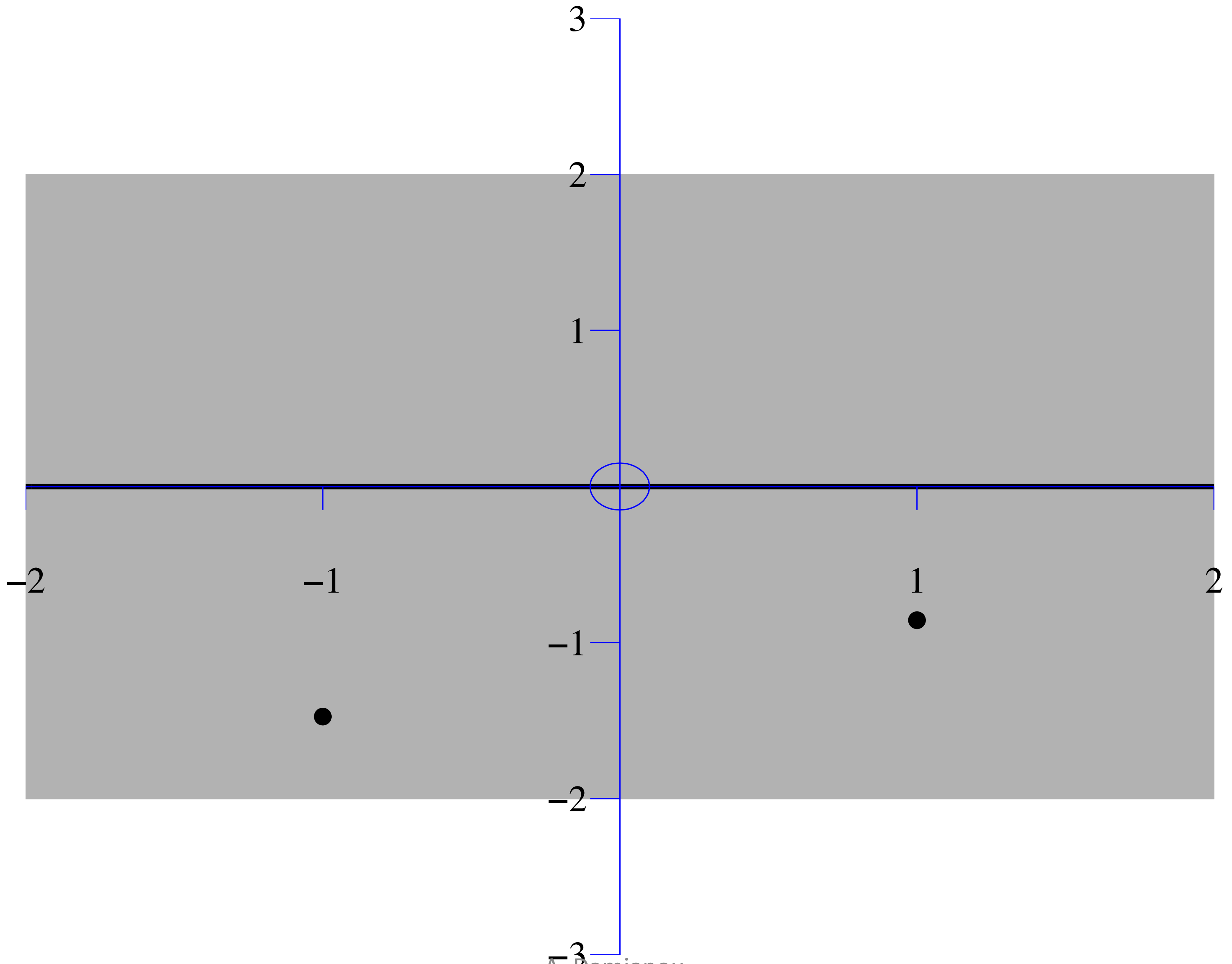
Fitting the data *(shaded area is* **uncertainty***)*



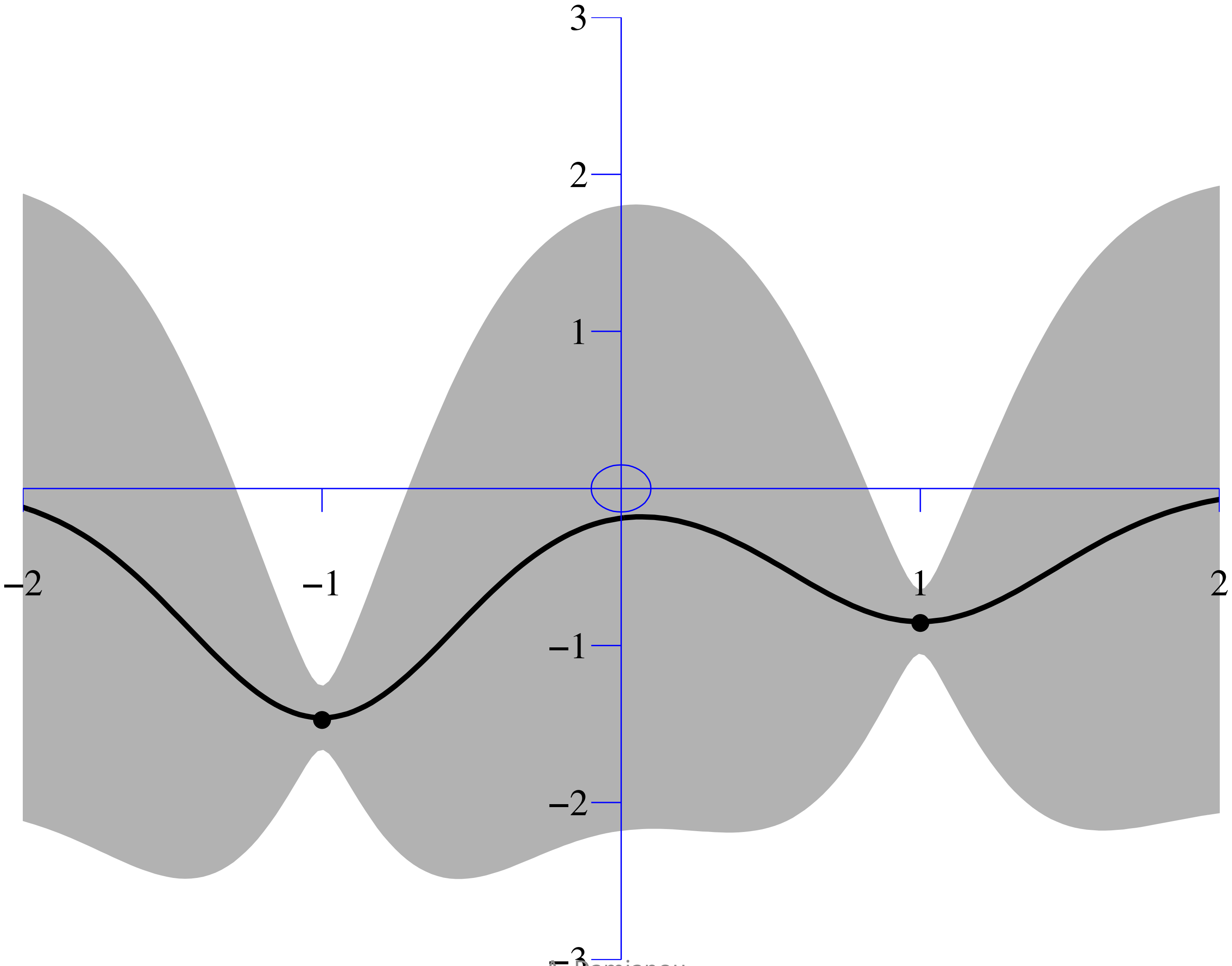
Fitting the data - Prior Samples



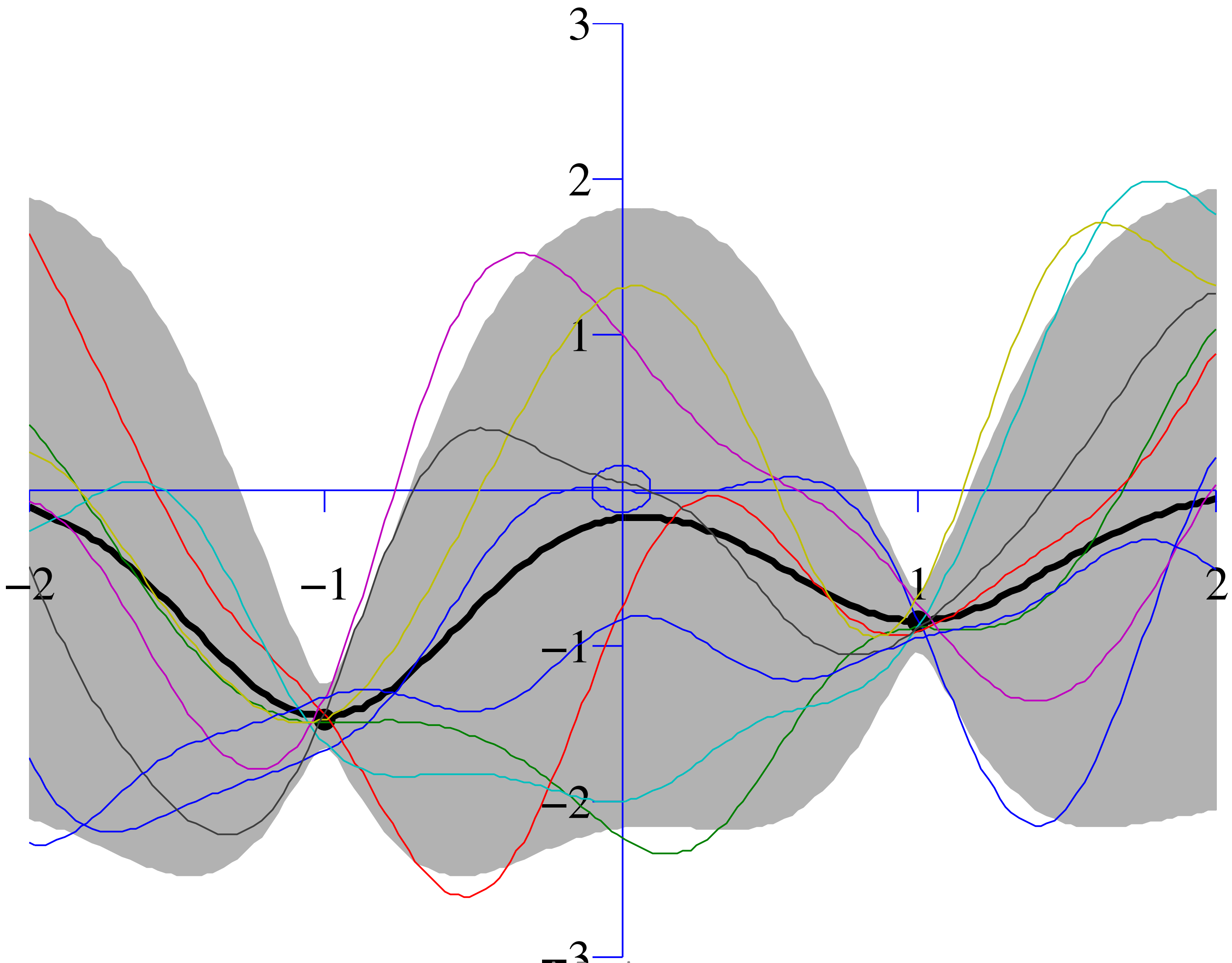
Fitting the data



Fitting the data



Fitting the data - Posterior samples

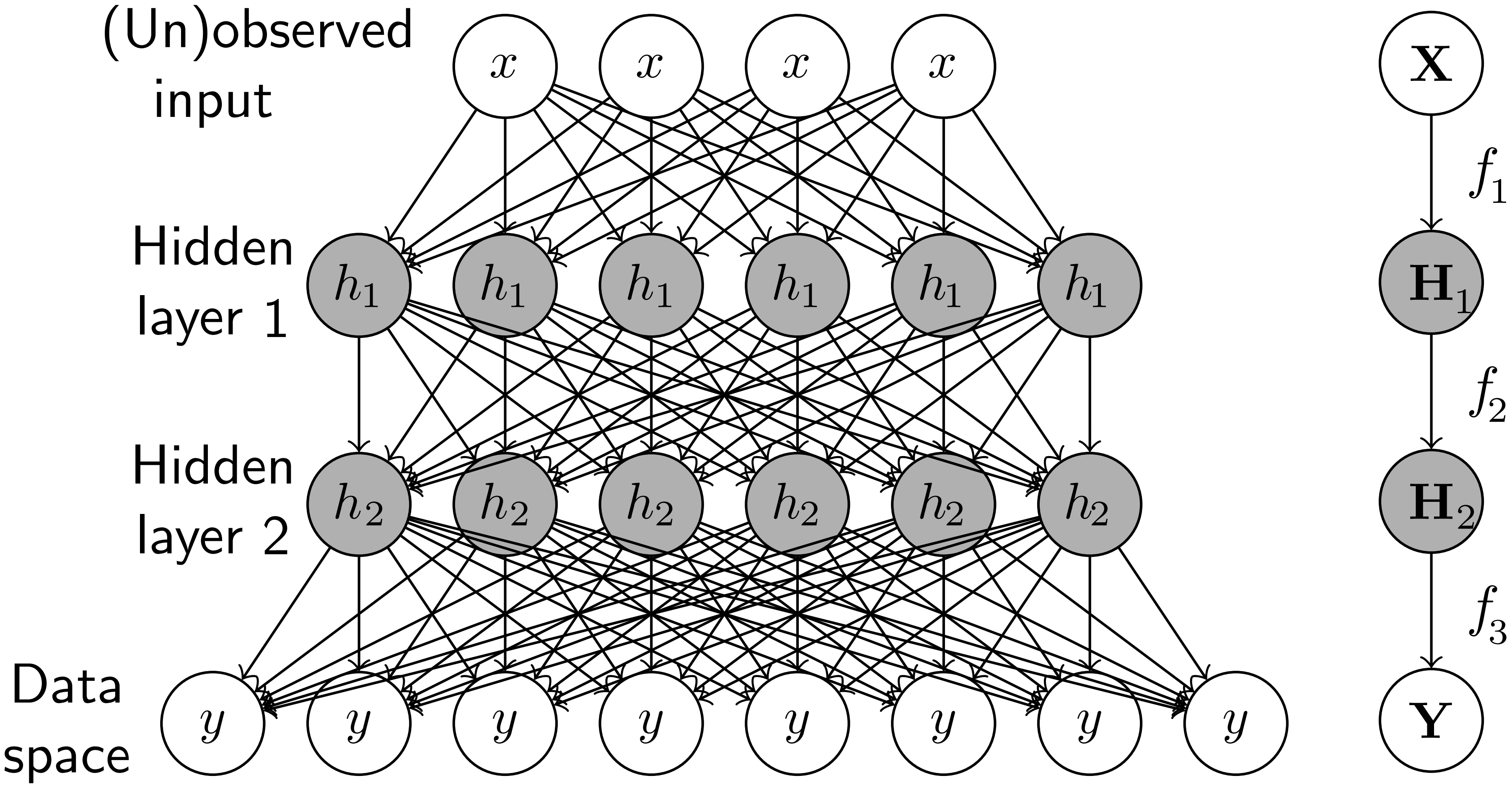


Summary – Gaussian processes

- GPs generalize Gaussian distributions to infinite dimensions (i.e. functions)
- A GP does not have parameters. We only make implicit assumptions about the properties of the functions (e.g. smoothness).
- Predictions are analytic and come with uncertainty.

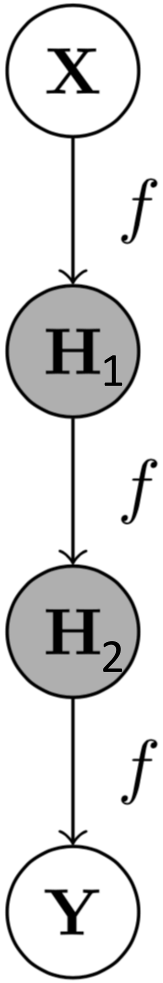
Part 2: Deep Gaussian processes

A general *family* of probabilistic models



$$\mathbf{Y} = f_3(f_2(\cdots f_1(\mathbf{X}))), \qquad \mathbf{H}_i = f_i(\mathbf{H}_{i-1})$$

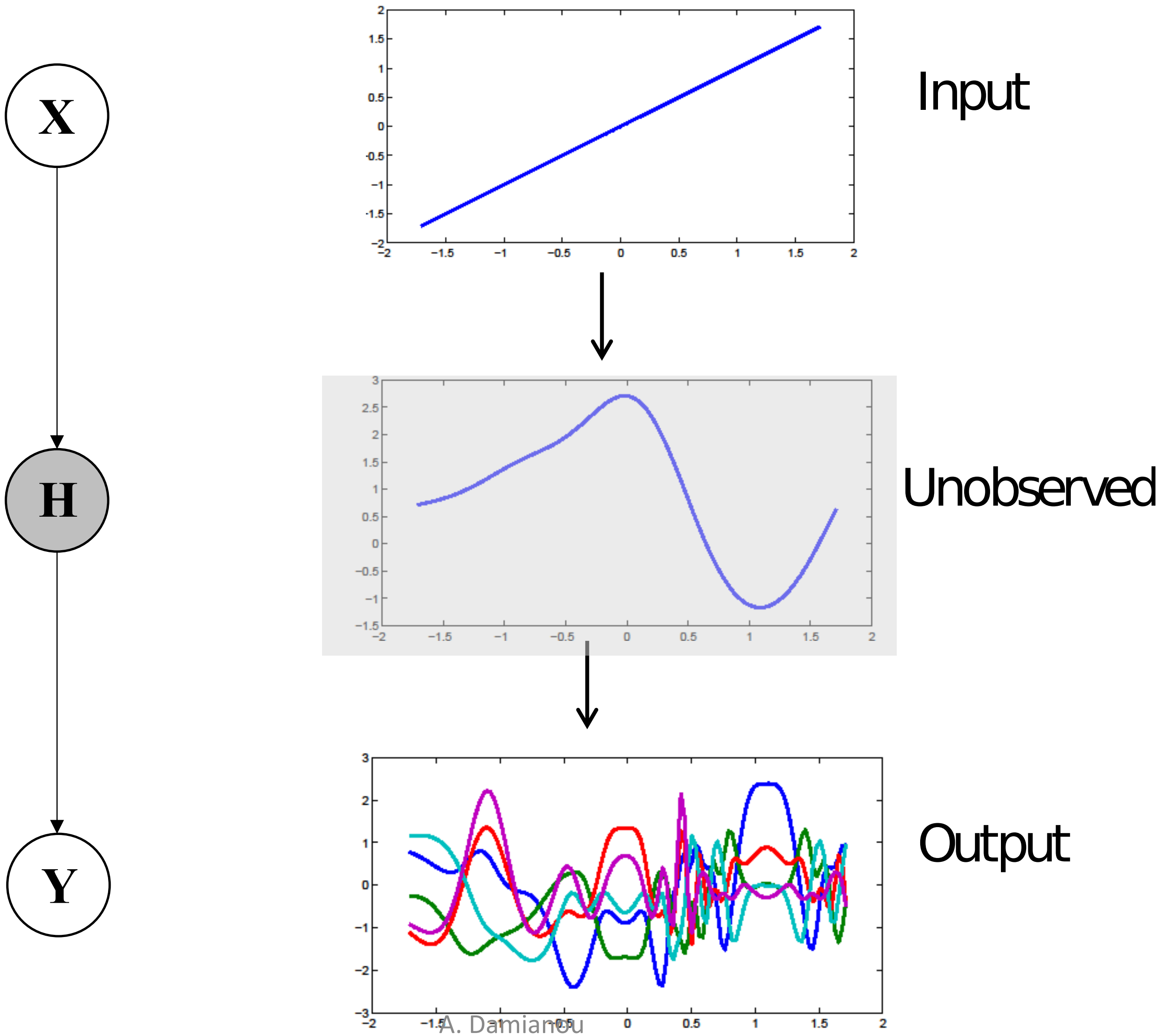
Deep Gaussian process



- Nested function composition
- Non-parametric, non-linear mappings *f*
- Mappings *f* marginalized out analytically
- NOT a GP!

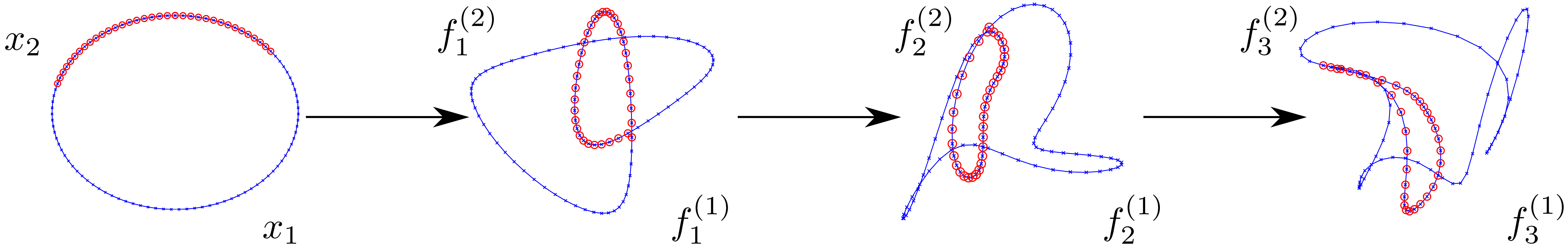
Damianou & Lawrence, 2013, Damianou, PhD Thesis 2015

Sampling from a Deep GP

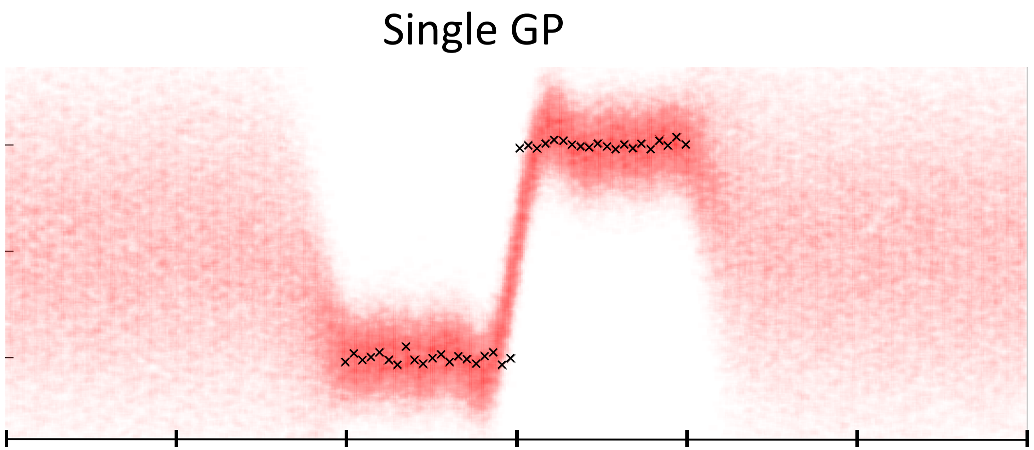


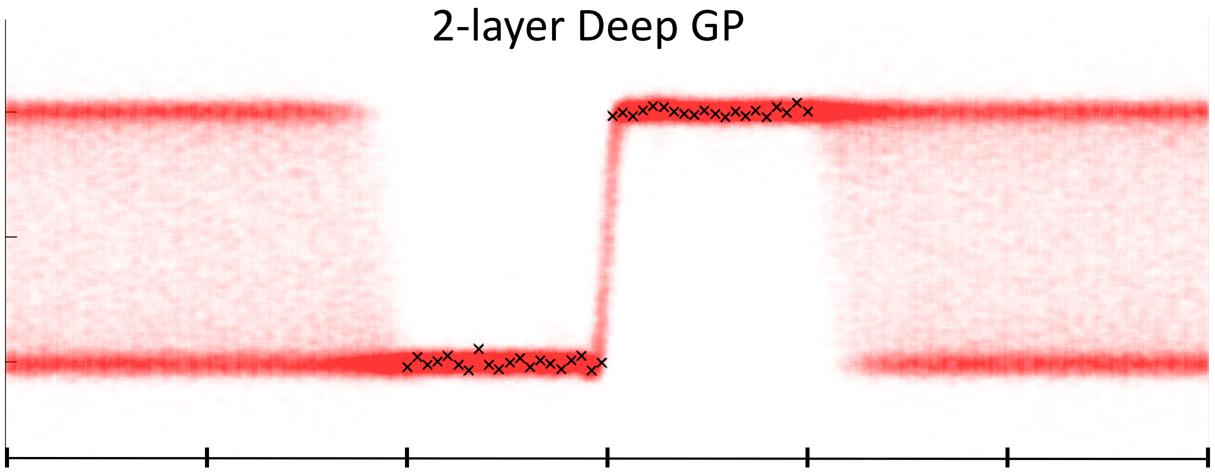
Feature learning

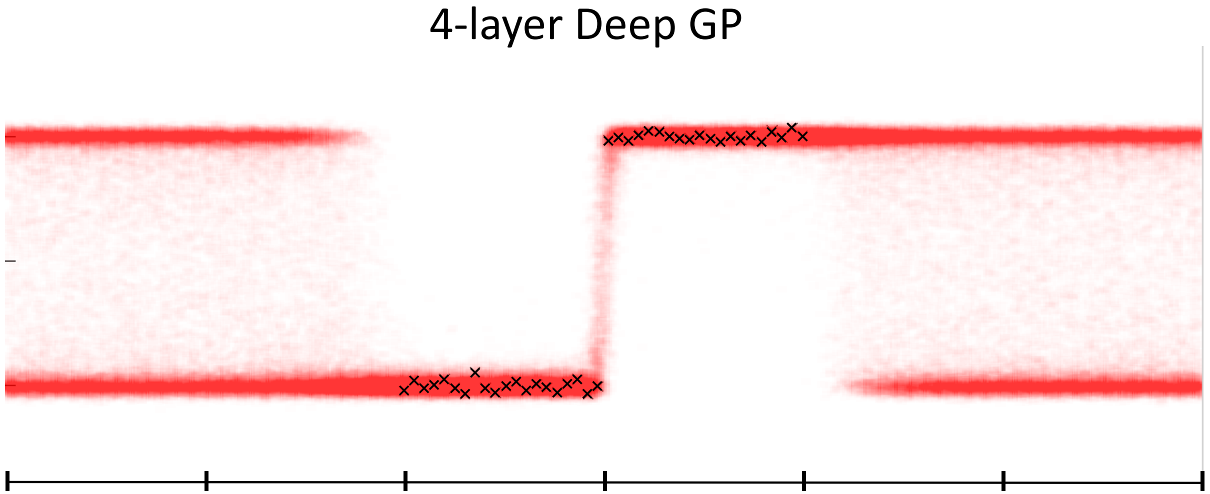
Regularities are learned as “knots” in the latent space, carried over from layer to layer.



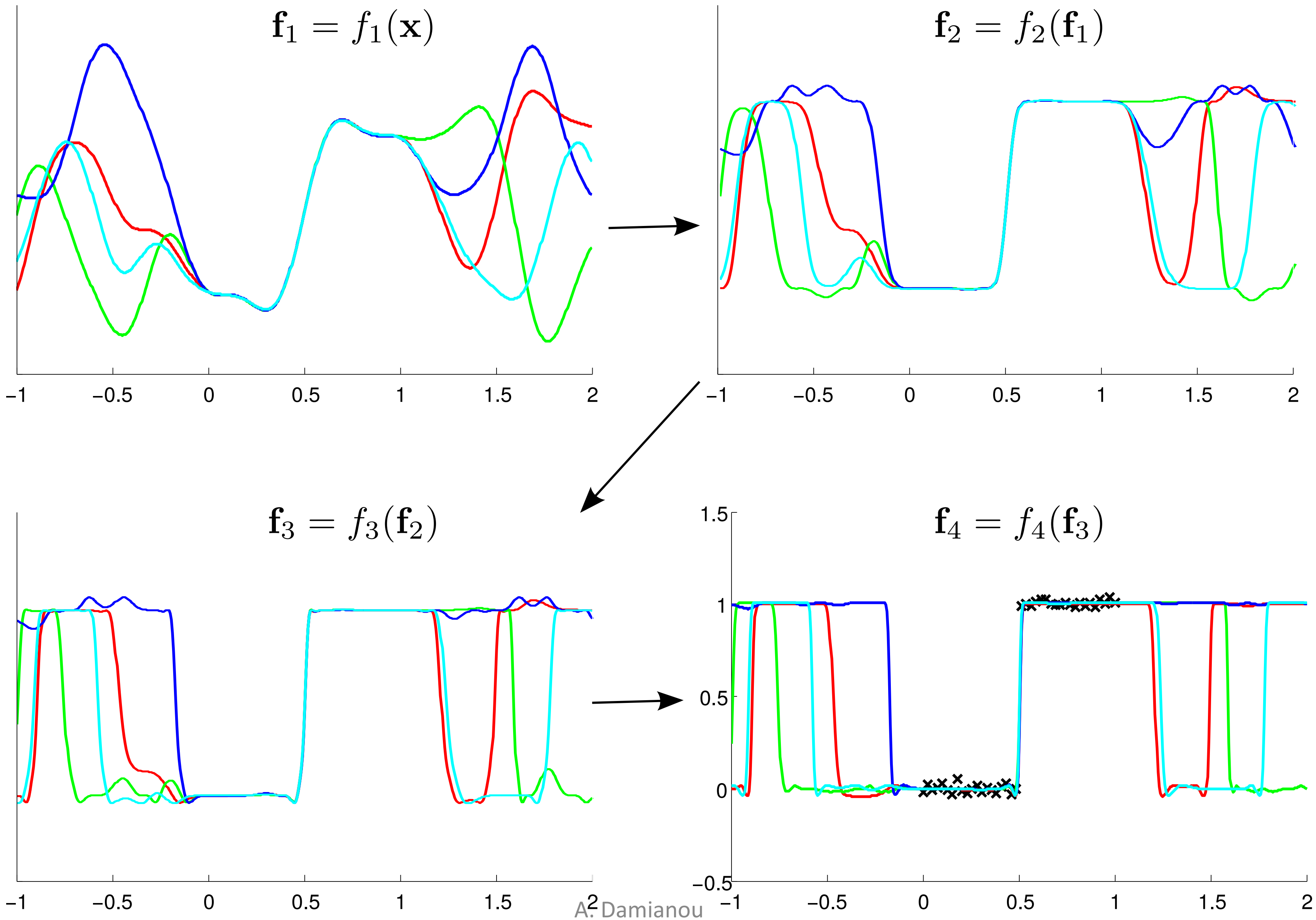
Step function example







Successive warping to learn the step function



Properties

- Unsupervised learning possible due to Bayesian regularization
- Very data efficient
- Scalability also possible with newer techniques

- Intractable objective
- Classification is more challenging

Summary – Deep Gaussian processes

- A DGP is a GP whose input is a GP, whose input is a GP…

- Propagate uncertainty across layers (not only point estimates)

Summary – Deep Gaussian processes

- A DGP is a GP whose input is a GP, whose input is a GP…

- Propagate uncertainty across layers (not only point estimates)

- What if layers represent different *kinds* of observation spaces? E.g. different *fidelities*?

Part 3: Multi-fidelity modeling

Multi-fidelity data

High fidelity observations

Low fidelity observations

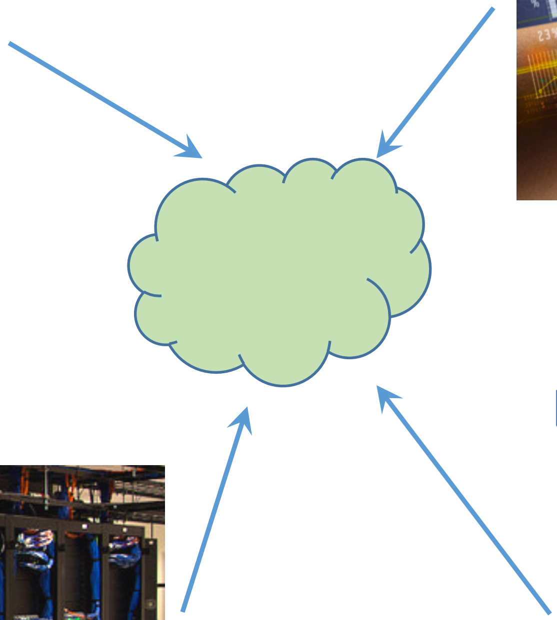
High fidelity simulations

Low fidelity simulations

Multi-fidelity data

High fidelity observations

Low fidelity observations



High fidelity simulations

Low fidelity simulations

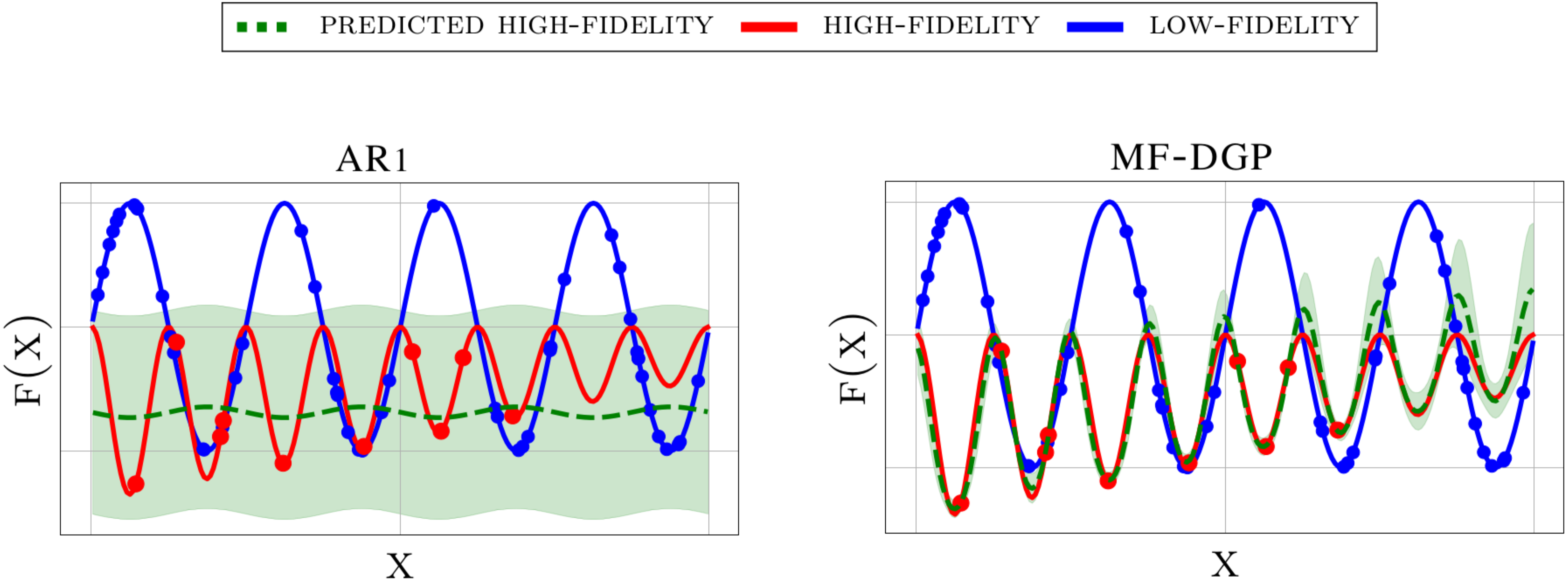
X_H

$Y_H =$ OK
OK
OK
OK
ERROR!
OK
OK

X_L

$Y_L =$ OK
OK
ERROR!
OK
ERROR!
OK
ERROR!

Fusing information from multiple fidelities



We want to trust the high-fidelity data, where we have them, and where we don't have them to learn how to reason based on low-fidelity data.

Linear GP multi-fidelity

$$f_H(x) = \rho_H f_L(x) + \delta_H(x)$$

f_H High fidelity function (GP)

ρ_H Contribution of low fidelity (const)

f_L Low fidelity function (GP)

δ_H Bias between fidelities (GP)

Kennedy & O'Hagan 2000, Le Gratiet & Garnier 2014

Non-linear multi-fidelity GP => Deep GP

$$f_H(x) = \rho_H f_L(x) + \delta_H(x)$$

Linear relationship between fidelities

$$f_H(x) = \rho_H(f_L(x), x) + \delta_H(x)$$

Non-linear relationship between fidelities
(if ρ is a GP -> overall a deep GP!)

Non-linear multi-fidelity GP => Deep GP

$$f_H(x) = \rho_H f_L(x) + \delta_H(x)$$

Linear relationship between fidelities

$$f_H(x) = \rho_H(f_L(x), x) + \delta_H(x)$$

Non-linear relationship between fidelities
(if ρ is a GP -> overall a deep GP!)

$$\Downarrow$$

$$f_H(x) = g_H(f_L^*(x), x)$$

δ absorbed into g

$f_L^*(x)$ denotes the posterior of the GP modeling the low-fidelity data.

Perdikaris et al., Proc. Royal Society, 2017

Algorithm

1. Train f_L on (X_L, Y_L)

2. Compute $f_L^*(X_H)$

3. Train f_H on $((X_L, Y_L), f_L^*(X_H))$

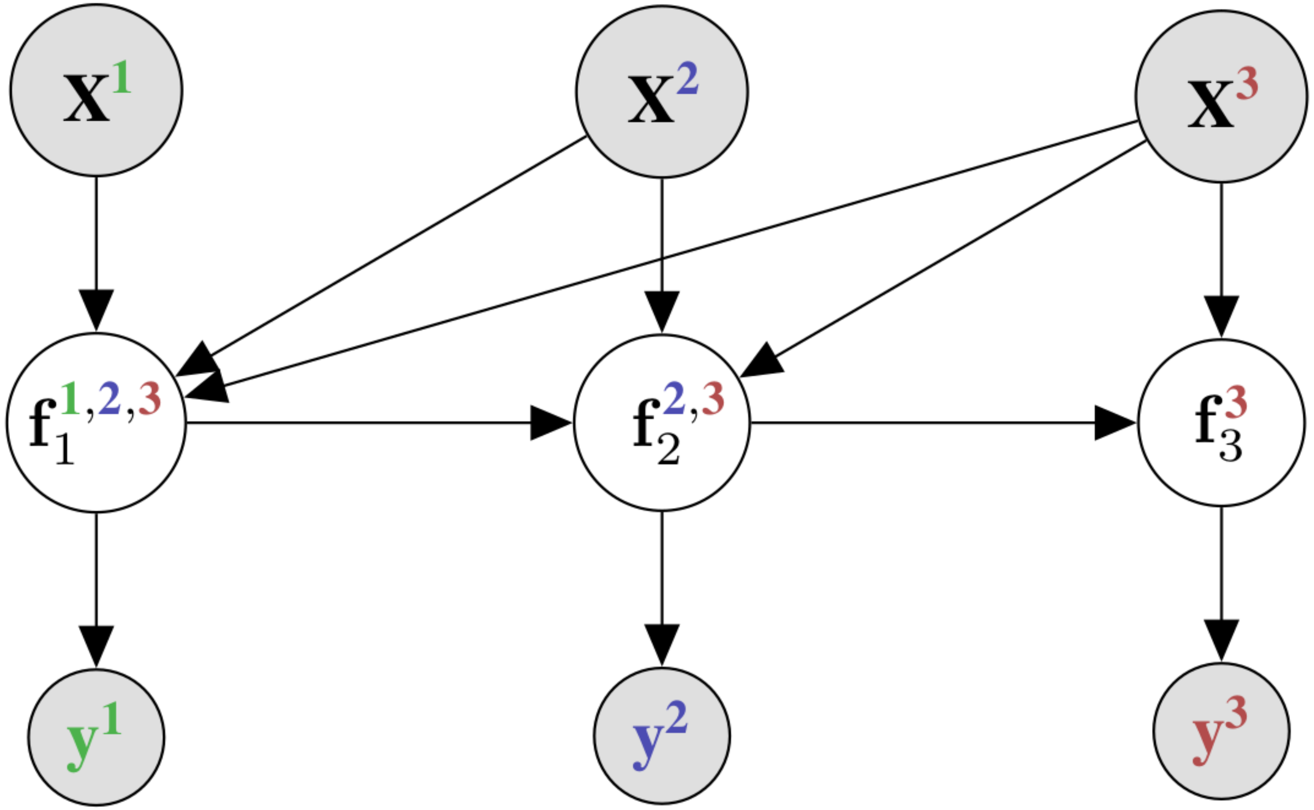
Predictions

$$p(f_H^*(x^*)) =$$

$$\int \underbrace{p(f_H(x^*, f_L^*(x^*))|y_H, x_H, x^*)}_{\text{Local posterior from fidelity } H} \; \underbrace{p(f_L^*(x^*))}_{\text{Predictive from fidelity } L} \, df_L^*$$

Communication between fidelities during training

$f_{\text{layer}}^{\text{fidelity}}$



Cutajar et al. arXiv: 1903.07320

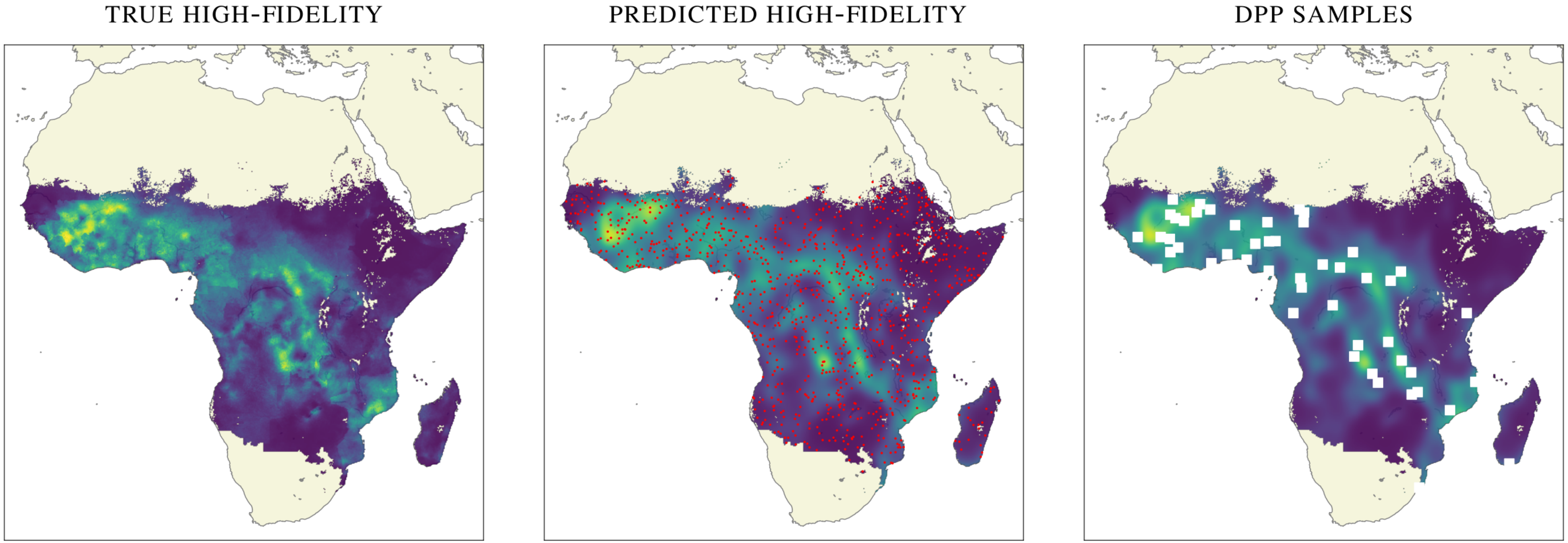


Figure 5: Real-world experiment indicating the infection rate of *Plasmodium falciparum* among African children. Lighter-shaded regions denote higher infection rates in that area of the continent. *Left:* True infection rates recorded for the year 2015. *Center:* MF-DGP predictions given low-fidelity data from 2005 and limited high-fidelity training points (marked in red) from 2015. *Right:* White squares show the samples drawn from a DPP using the posterior covariance of the MF-DGP model as its kernel.

Sequential design for multi-fidelity GP

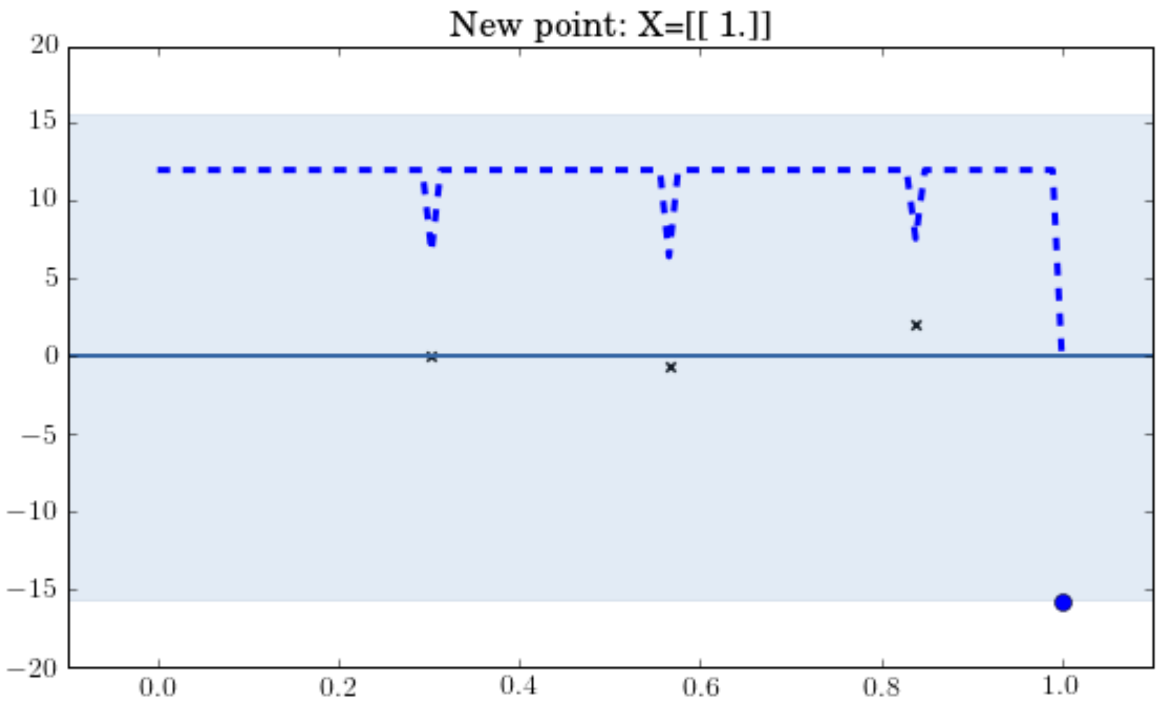
- **Strategy for collecting data points** across fidelities

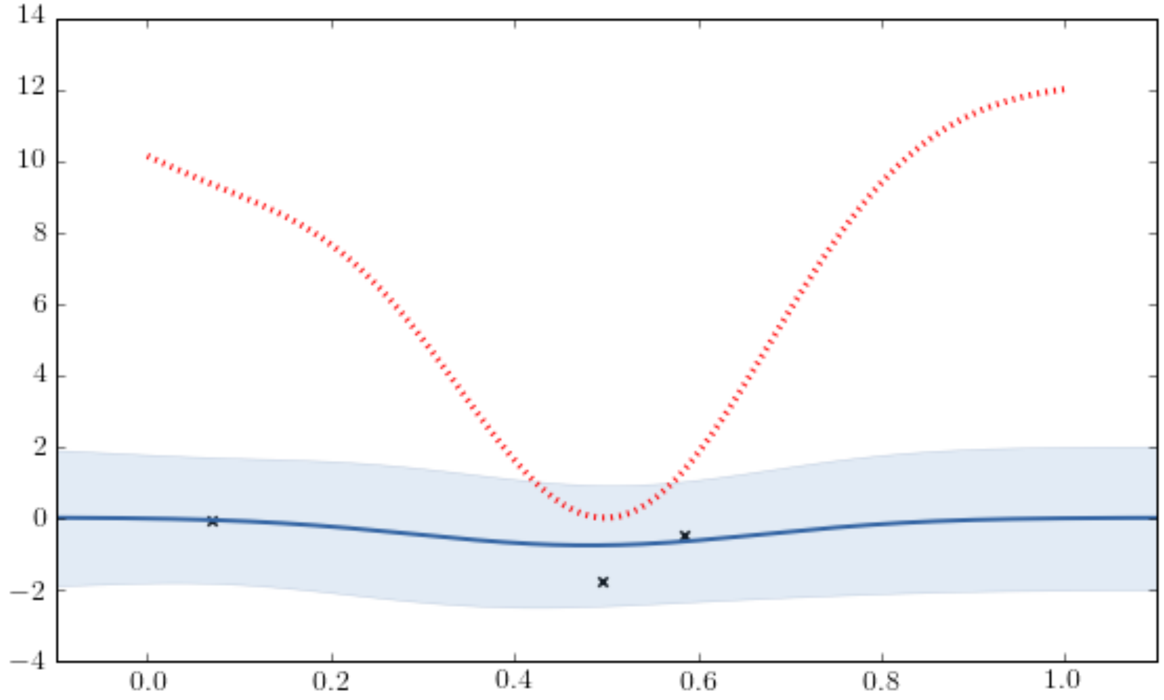
- Each fidelity evaluation comes with a different **cost**, proportional to the level of fidelity

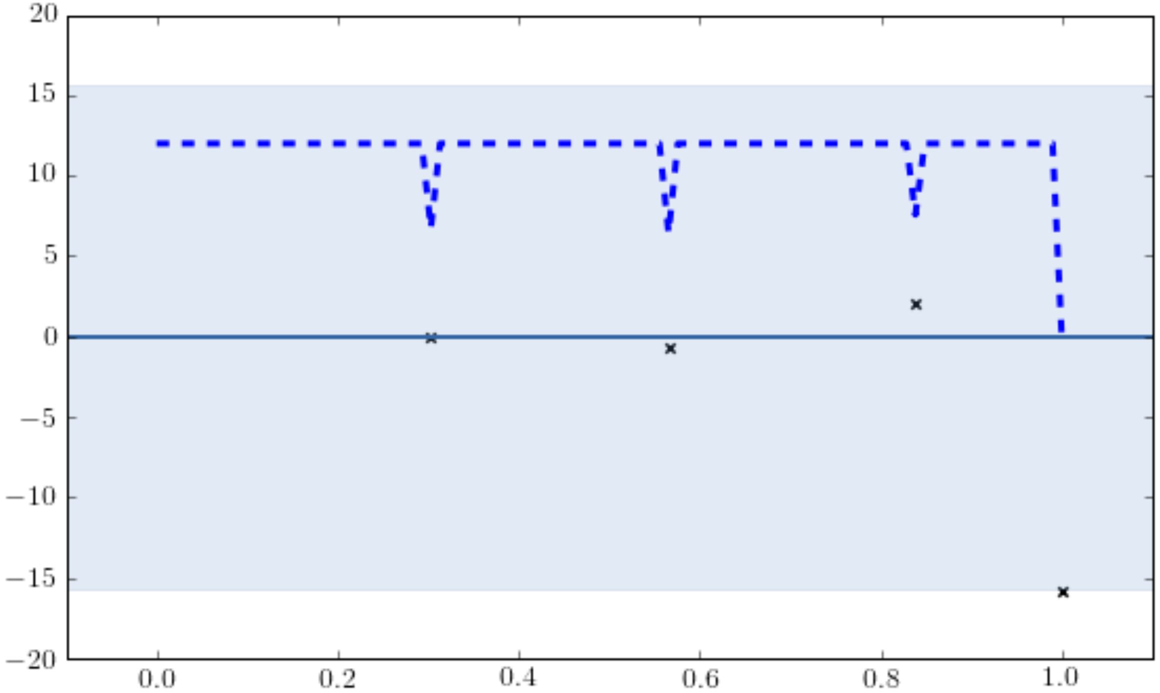
- Approach: Formulation as a multi-fidelity **bandit** GP problem in the UCB setting *(Kandasamy et al. 2016)*

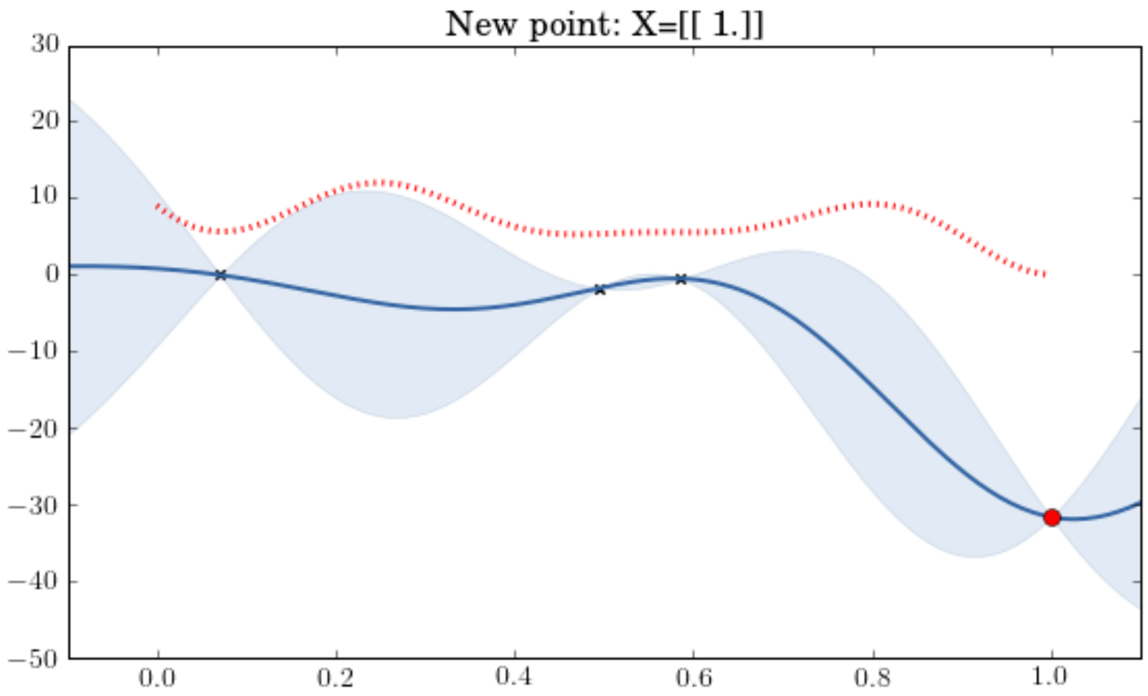
Demonstration

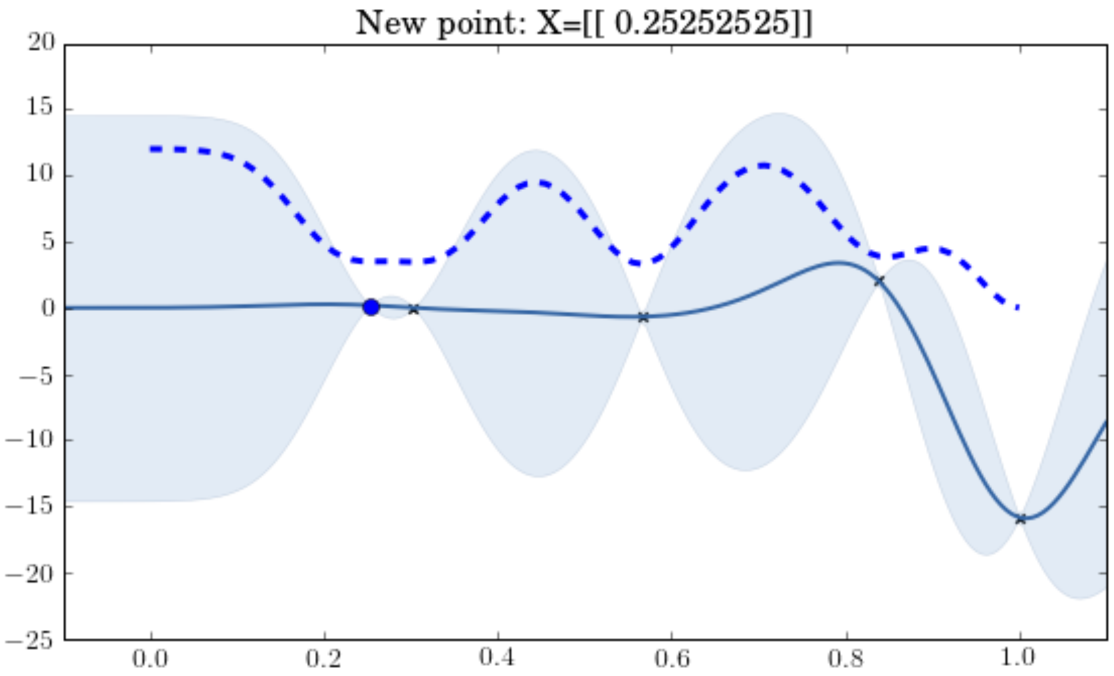
- Find the maximum of the high-fidelity function
- Consider cost while collecting points from each fidelity

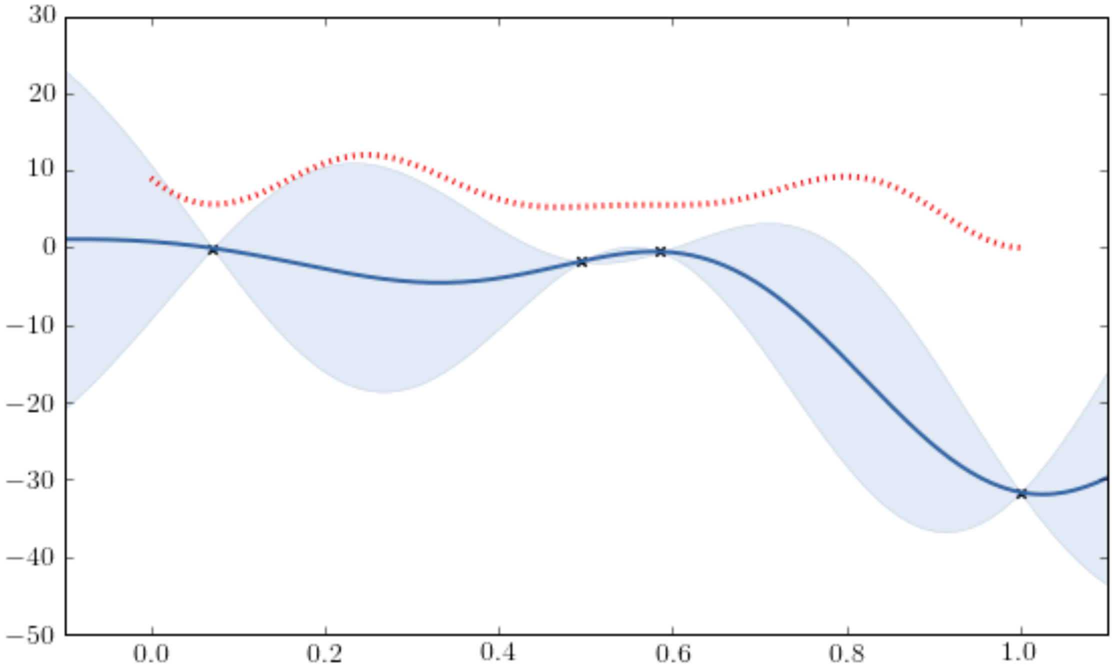

New point: X=[[1.]]
20
15
10
5
0
−5
−10
−15
−20
0.0
0.2
0.4
0.6
0.8
1.0

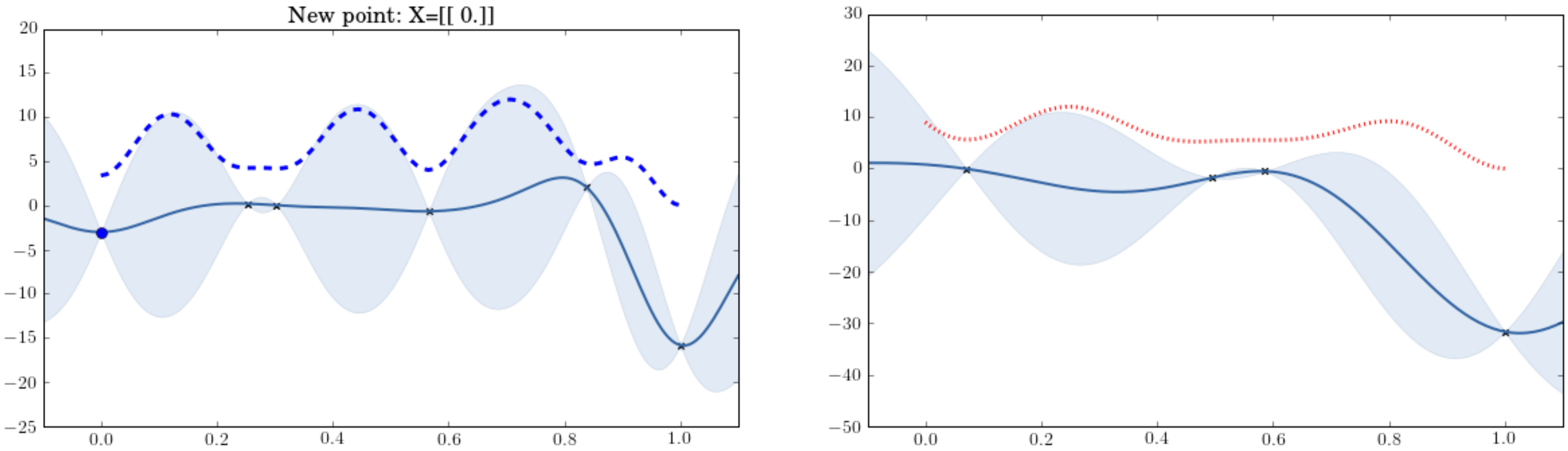

14
12
10
8
6
4
2
0
−2
−4
0.0
0.2
0.4
0.6
0.8
1.0


20
15
10
5
0
−5
−10
−15
−20
0.0
0.2
0.4
0.6
0.8
1.0

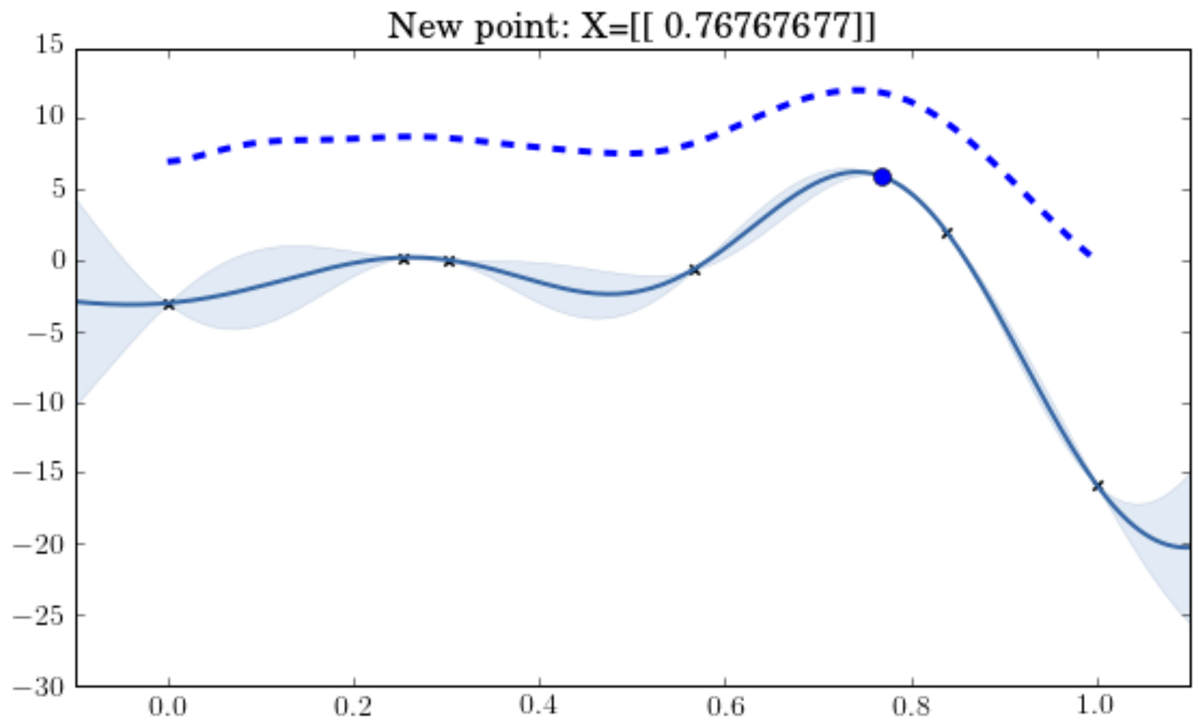

New point: X=[[1.]]
30
20
10
0
−10
−20
−30
−40
−50
0.0
0.2
0.4
0.6
0.8
1.0

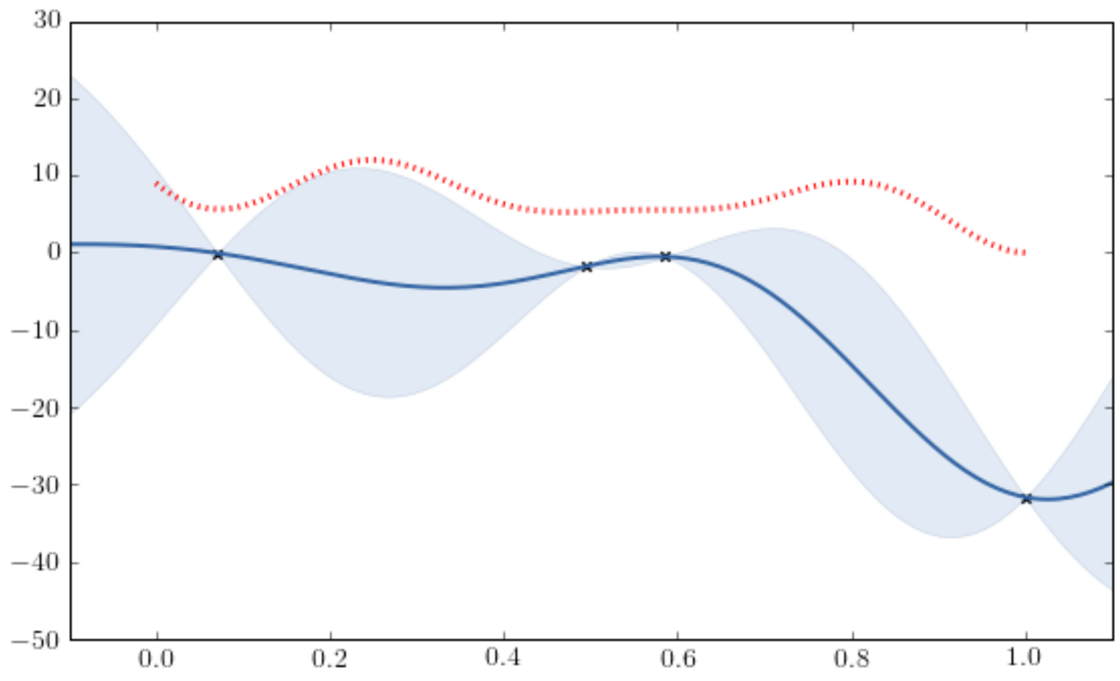

New point: X=[[0.25252525]]
20
15
10
5
0
−5
−10
−15
−20
−25
0.0
0.2
0.4
0.6
0.8
1.0

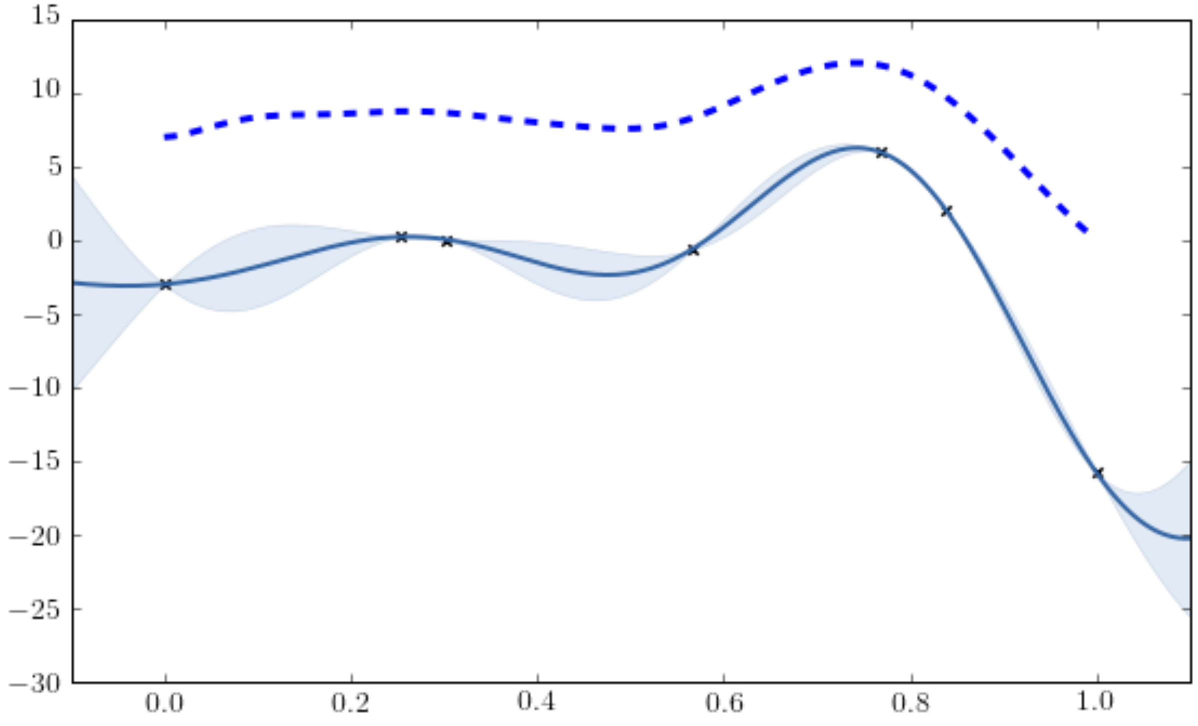

30
20
10
0
−10
−20
−30
−40
−50
0.0
0.2
0.4
0.6
0.8
1.0

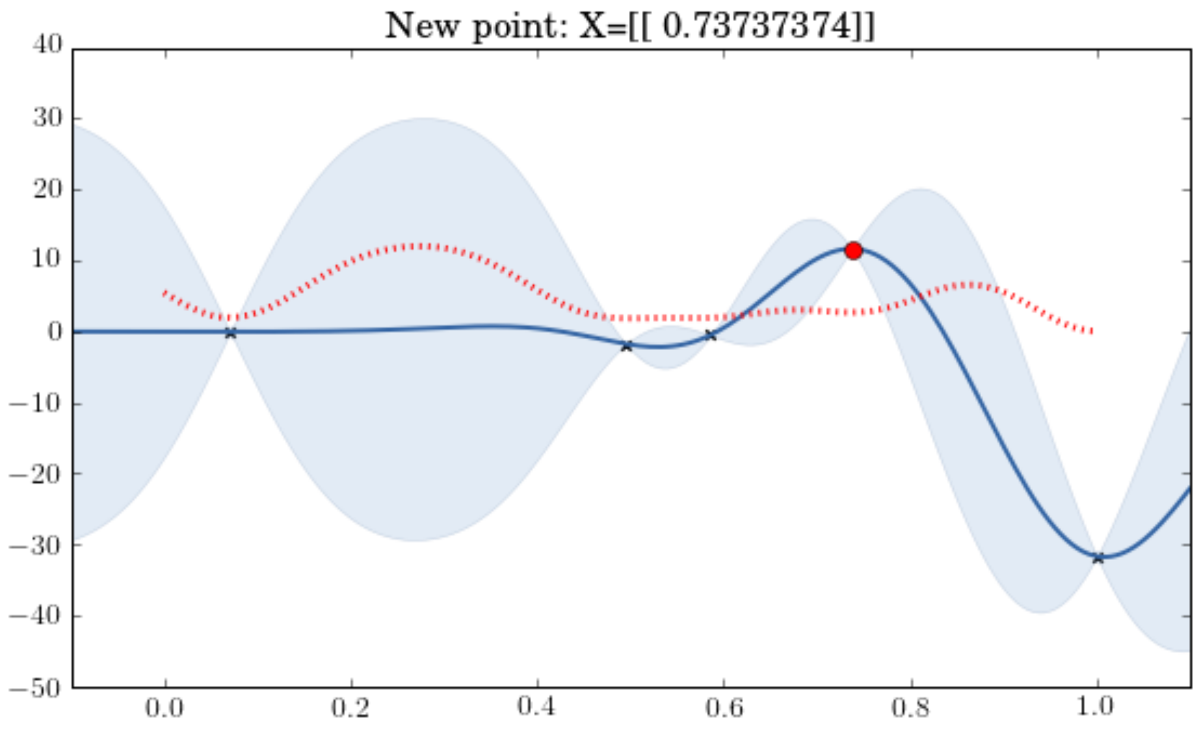


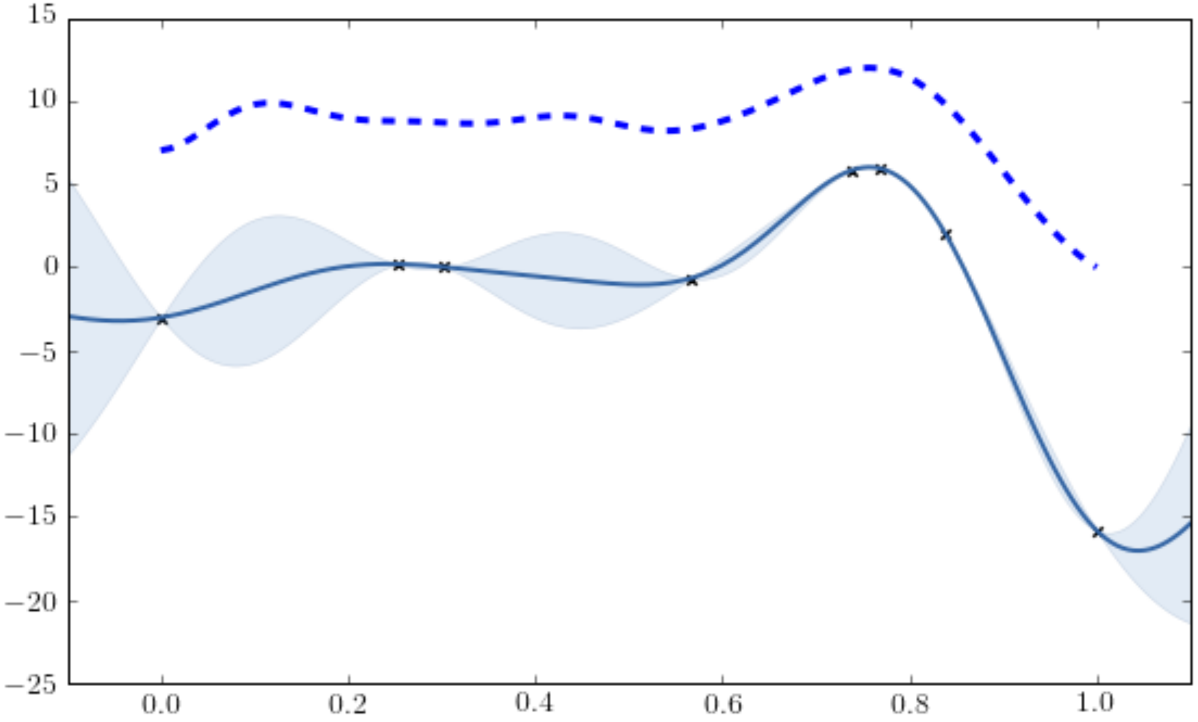
New point: X=[[0.]]

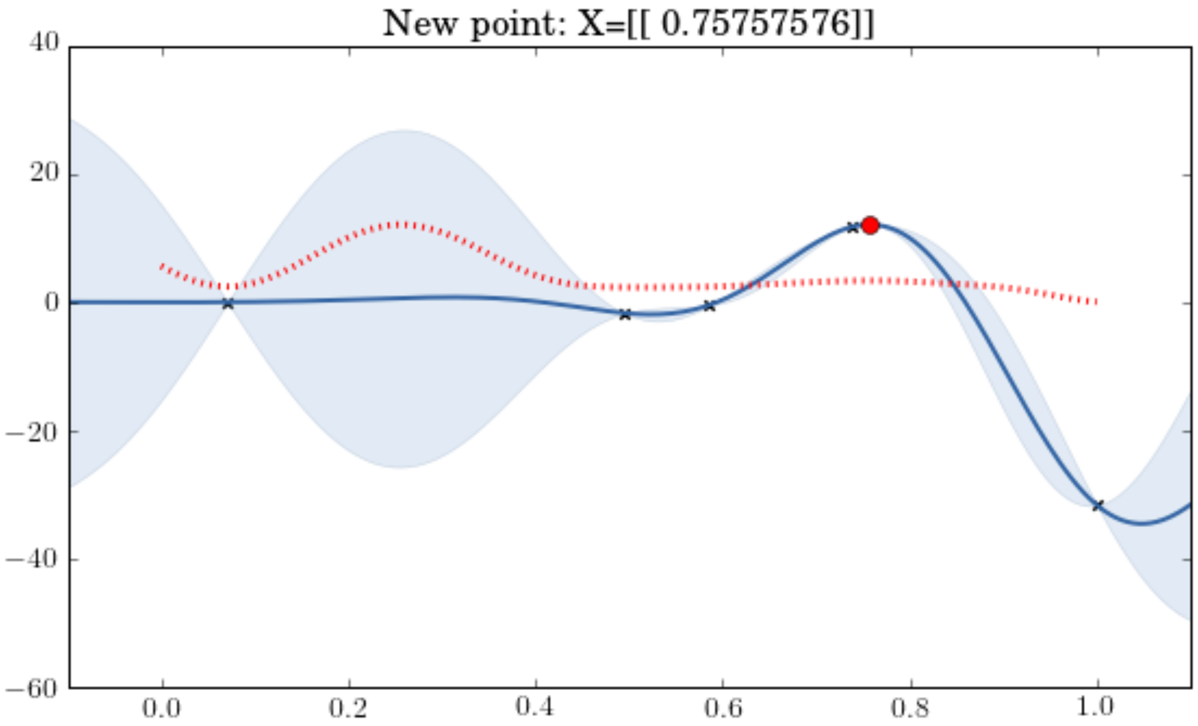

New point: X=[[0.76767677]]
15
10
5
0
−5
−10
−15
−20
−25
−30
0.0
0.2
0.4
0.6
0.8
1.0


30
20
10
0
−10
−20
−30
−40
−50
0.0
0.2
0.4
0.6
0.8
1.0


15
10
5
0
−5
−10
−15
−20
−25
−30
0.0
0.2
0.4
0.6
0.8
1.0


New point: X=[[0.73737374]]
40
30
20
10
0
−10
−20
−30
−40
−50
0.0
0.2
0.4
0.6
0.8
1.0


15
10
5
0
−5
−10
−15
−20
−25
0.0
0.2
0.4
0.6
0.8
1.0


New point: X=[[0.75757576]]
40
20
0
−20
−40
−60
0.0
0.2
0.4
0.6
0.8
1.0

Conclusion

- GPs: Non-parametric inference over the space of functions
- Deep GPs learn rich mappings in a regularized and data efficient manner
- Multi-fidelity (D)GPs allow us to fuse multiple fidelity data
- We can actively acquire data of different fidelities using the GP emulators in each fidelity.

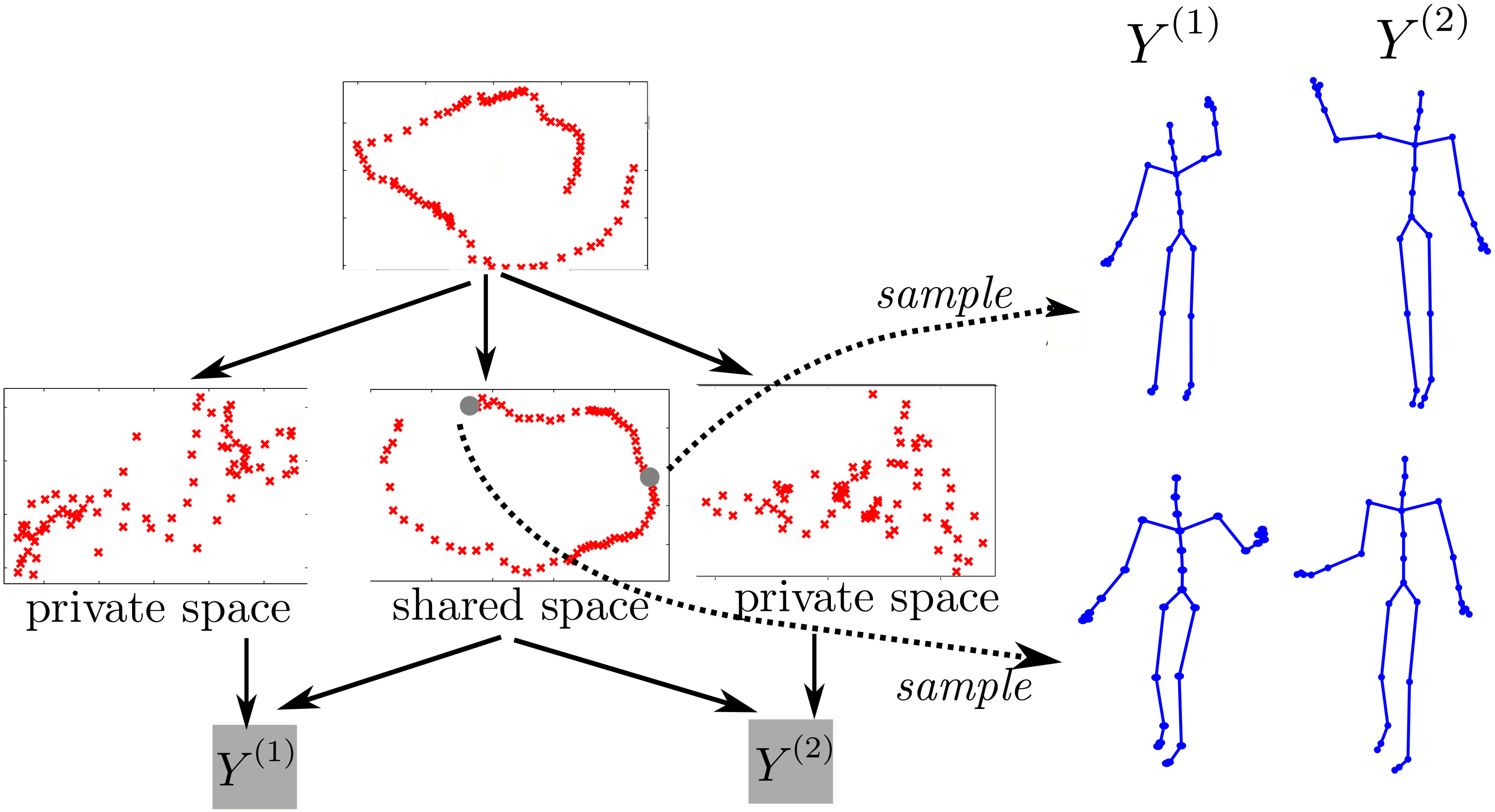
Thanks!

Questions?

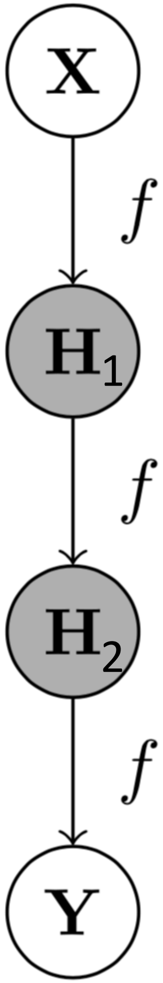
See also: http://adamian.github.io/talks/Damianou_GP_tutorial.html

Appendix

Unsupervised learning for multiple views



Deep Gaussian process

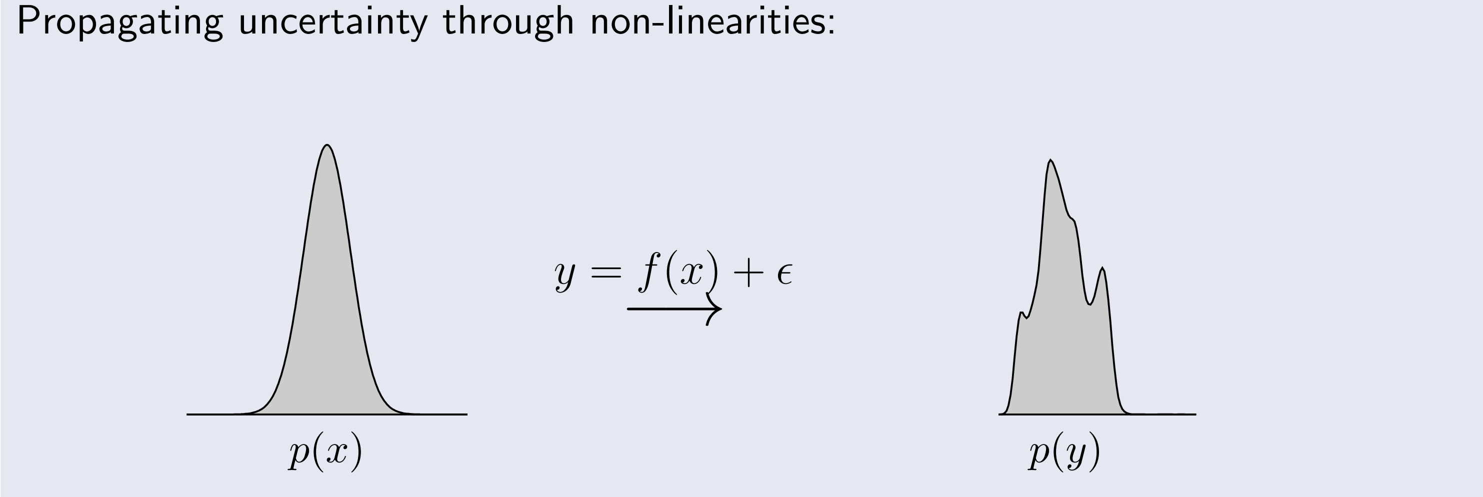


- Objective: $p(y|x) = \int_{h_2} \left(p(y|h_2) \int_{h_1} p(h_2|h_1) p(h_1|x) \right)$
- $p(h_2|x) \quad = \int_{h_1, f_2} \quad p(h_2|f_2) p(f_2|h_1) \quad p(h_1|x)$

Inference in Deep GPs: uncertainty propagation

- Objective: $p(y|x) = \int_{h_2} \left(p(y|h_2) \int_{h_1} p(h_2|h_1) p(h_1|x) \right)$

- $p(h_2|x) \quad = \int_{h_1, f_2} p(h_2|f_2) \underbrace{p(f_2|h_1)}_{\text{contains } (k(h_1,h_1))^{-1}} p(h_1|x)$



Variational bound and its properties

Bound on the log marginal likelihood *log p(y)*

$$\mathcal{F} = \overbrace{\sum_{l=2}^{L+1} \left\langle \sum_{n=1}^{N} \mathcal{L}(\mathbf{h}_l^{(n)}, \mathbf{u}_l) \right\rangle_{\mathcal{Q}}}^{\text{Data fit}} - \sum_{l=2}^{L+1} \mathsf{KL}\left(q(\mathbf{u}_l) \parallel p(\mathbf{u}_l)\right) \underbrace{- \mathsf{KL}\left(q(\mathbf{h}_1) \parallel p(\mathbf{h}_1)\right)}_{\text{Regularization}} + \sum_{l=2}^{L} \underbrace{\mathcal{H}\left(q(\mathbf{h}_l)\right)}_{\text{Regularization}}$$

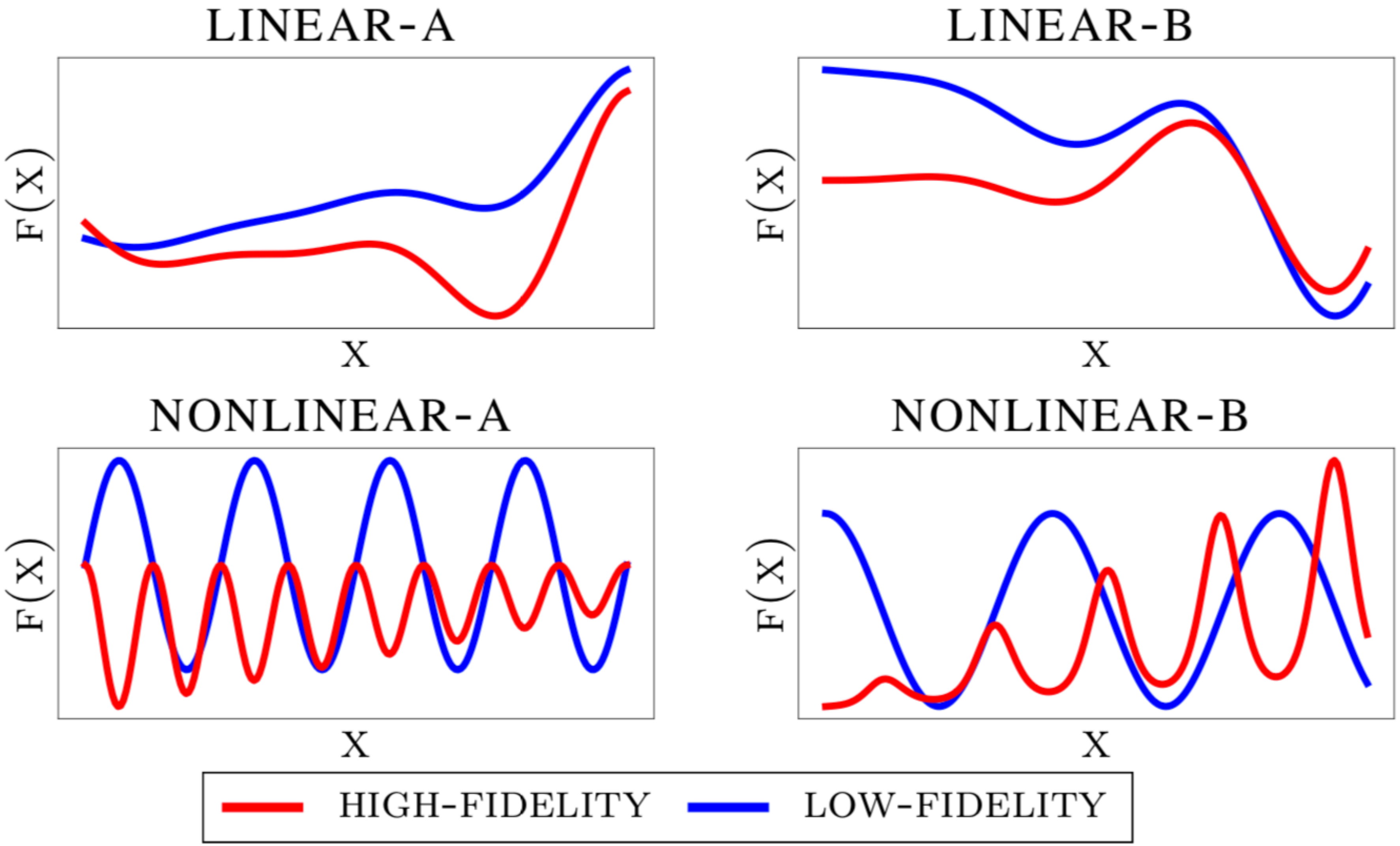


Figure 3: Synthetic examples. *Top:* Linear mapping between fidelities. *Bottom:* Nonlinear mapping.

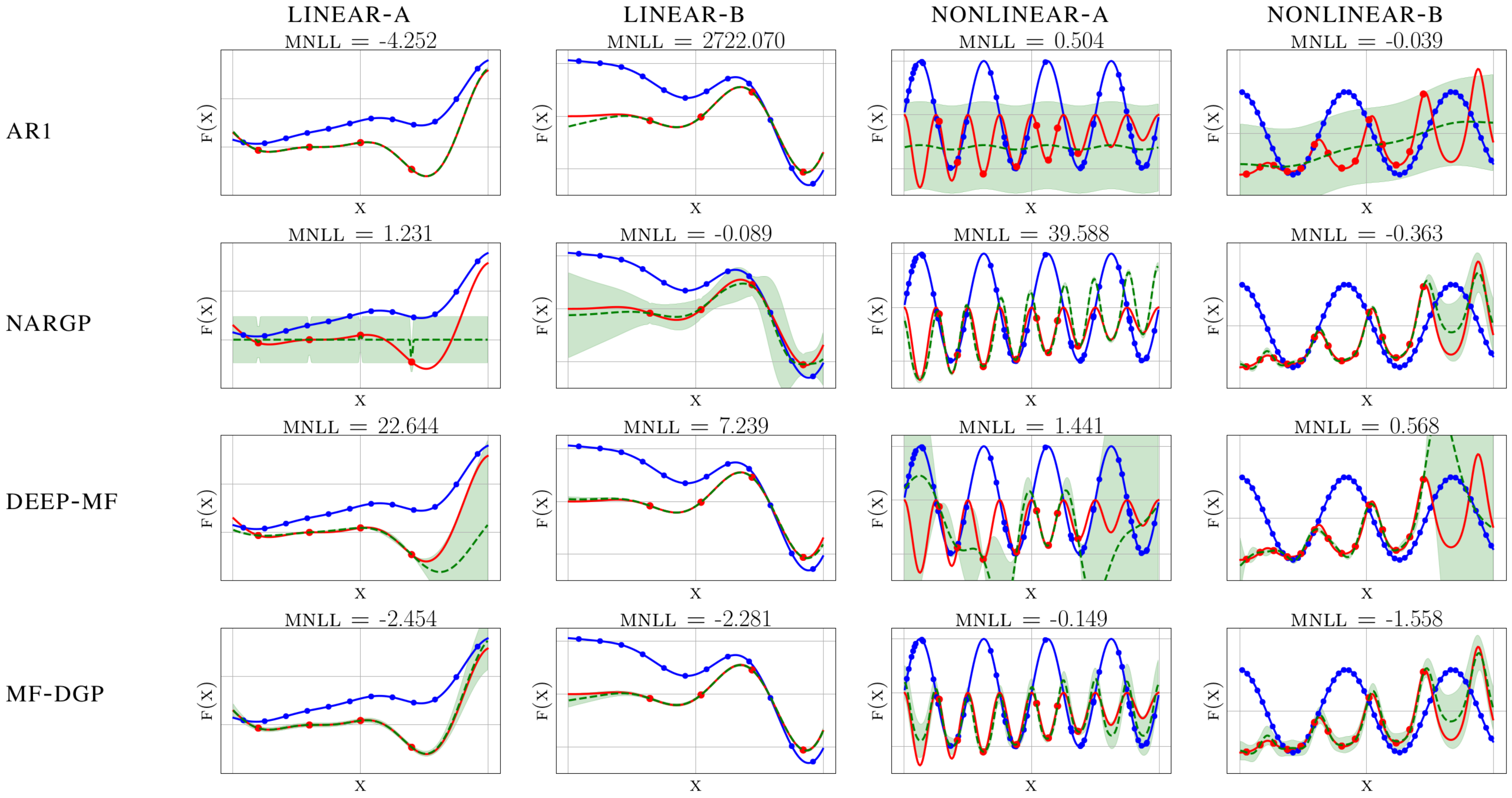


Figure 4: Cross-comparison across methods and synthetic examples for challenging multi-fidelity scenarios. MF-DGP yields conservative uncertainty estimates where few high-fidelity observations are available.

Benchmark examples

Table 1: Model Comparison on Multi-fidelity Benchmark Examples.

BENCHMARK	D_{in}	FIDELITY ALLOCATION	AR1 R^2	AR1 RMSE	AR1 MNLL	NARGP R^2	NARGP RMSE	NARGP MNLL	MF-DGP R^2	MF-DGP RMSE	MF-DGP MNLL
CURRIN	2	12-5	0.913	0.677	20.105	0.903	0.740	20.817	**0.935**	**0.601**	**0.763**
PARK	4	30-5	**0.985**	0.575	465.377	0.954	0.928	743.119	**0.985**	**0.565**	**1.383**
BOREHOLE	8	60-5	**1.000**	**0.005**	**-3.946**	0.973	0.063	-1.054	0.999	0.015	-2.031
BRANIN	2	80-30-10	0.891	0.044	-1.740	0.929	0.053	-1.223	**0.965**	**0.030**	**-2.572**
HARTMANN-3D	3	80-40-20	**0.998**	**0.043**	0.440	0.305	0.755	0.637	0.994	0.075	**-0.731**