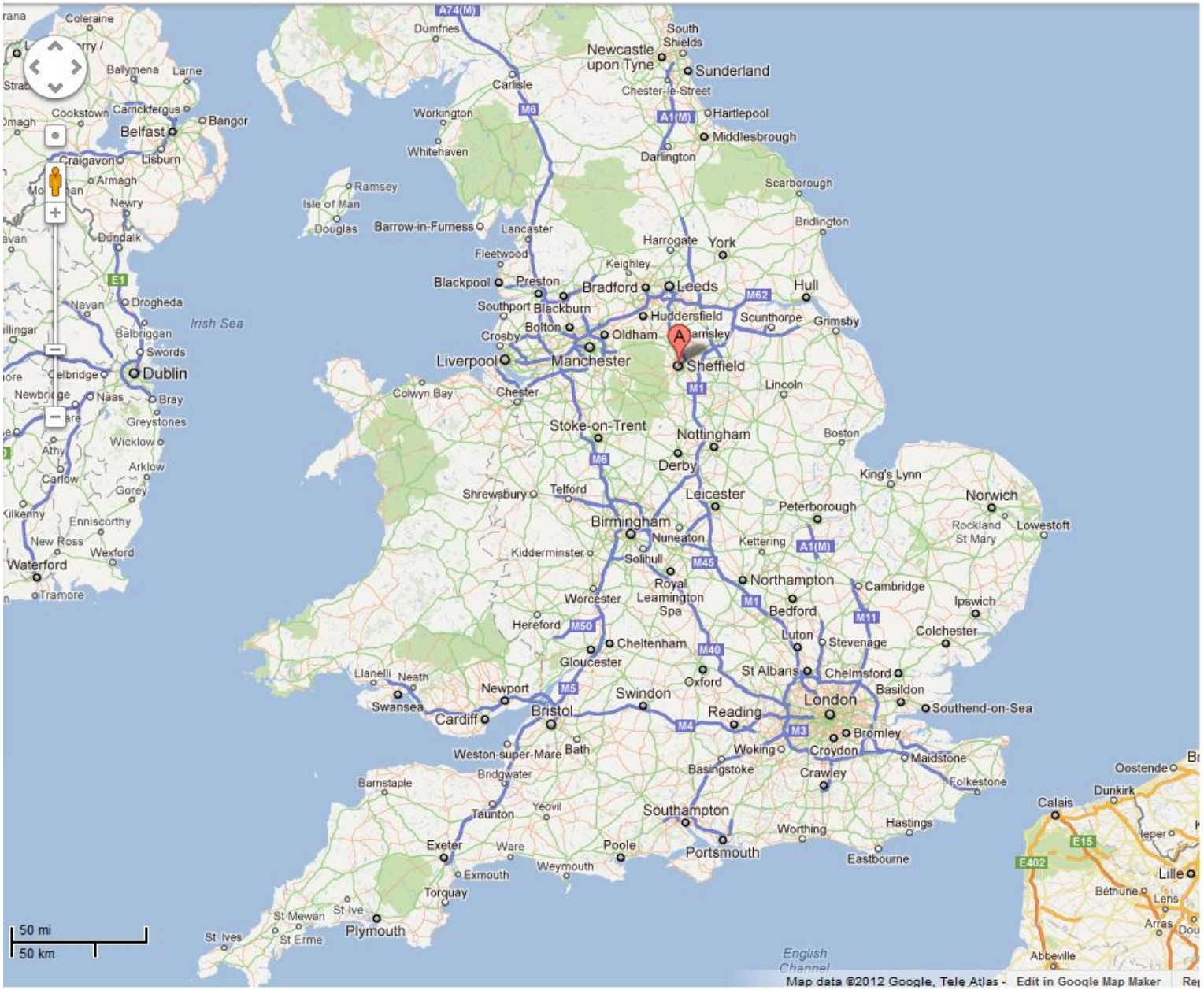
Gaussian processes for data-driven modelling and uncertainty quantification: a hands-on tutorial

Andreas Damianou

Department of Computer Science, University of Sheffield, UK

Brown University, 16/02/2016

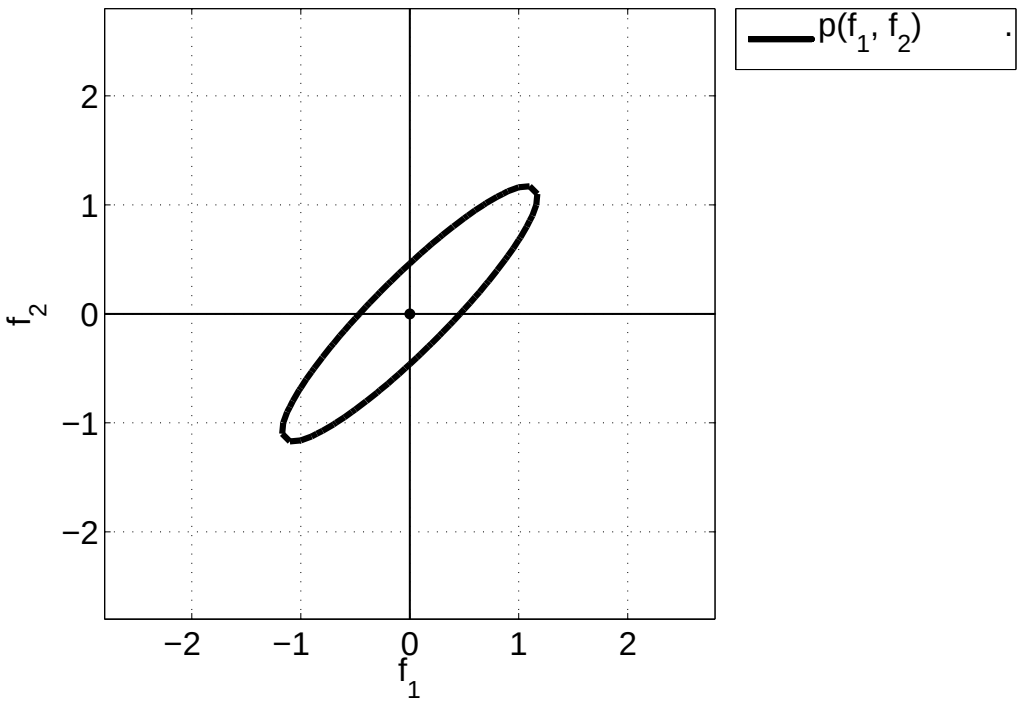
Sheffield


Sheffield
Manchester
Liverpool
Leeds
Bradford
Newcastle upon Tyne
Sunderland
Middlesbrough
Hartlepool
Darlington
Carlisle
Workington
Whitehaven
Dumfries
Barrow-in-Furness
Lancaster
Blackpool
Preston
Southport
Blackburn
Bolton
Oldham
Huddersfield
Scunthorpe
Grimsby
Hull
York
Harrogate
Keighley
Fleetwood
Scarborough
Bridlington
Lincoln
Boston
Nottingham
Derby
Stoke-on-Trent
Chester
Colwyn Bay
Shrewsbury
Telford
Leicester
Peterborough
Norwich
King's Lynn
Birmingham
Nuneaton
Kettering
Northampton
Cambridge
Ipswich
Bedford
Luton
Stevenage
Colchester
Hereford
Worcester
Royal Leamington Spa
Kidderminster
Solihull
Cheltenham
Gloucester
Oxford
St Albans
Chelmsford
Basildon
Southend-on-Sea
London
Reading
Swindon
Bristol
Cardiff
Newport
Swansea
Llanelli
Neath
Weston-super-Mare
Bath
Bridgwater
Taunton
Yeovil
Barnstaple
Exeter
Exmouth
Torquay
Plymouth
St Ives
St Mewan
St Erme
Ware
Poole
Weymouth
Southampton
Portsmouth
Worthing
Hastings
Eastbourne
Basingstoke
Woking
Croydon
Bromley
Crawley
Maidstone
Folkestone
Ramsey
Isle of Man
Douglas
Irish Sea
Dublin
Drogheda
Dundalk
Newry
Armagh
Belfast
Bangor
Larne
Ballymena
Coleraine
Cookstown
Carrickfergus
Craigavon
Lisburn
Navan
Balbriggan
Swords
Bray
Greystones
Wicklow
Arklow
Gorey
Enniscorthy
Wexford
Waterford
Tramore
New Ross
Carlow
Kilkenny
Athy
Celbridge
Naas
Newbridge
Tullamore
Rockland St Mary
Lowestoft
Oostende
Dunkirk
Calais
Lille
Béthune
Lens
Arras
Abbeville
English Channel
50 mi
50 km
Map data ©2012 Google, Tele Atlas - Edit in Google Map Maker

Outline

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

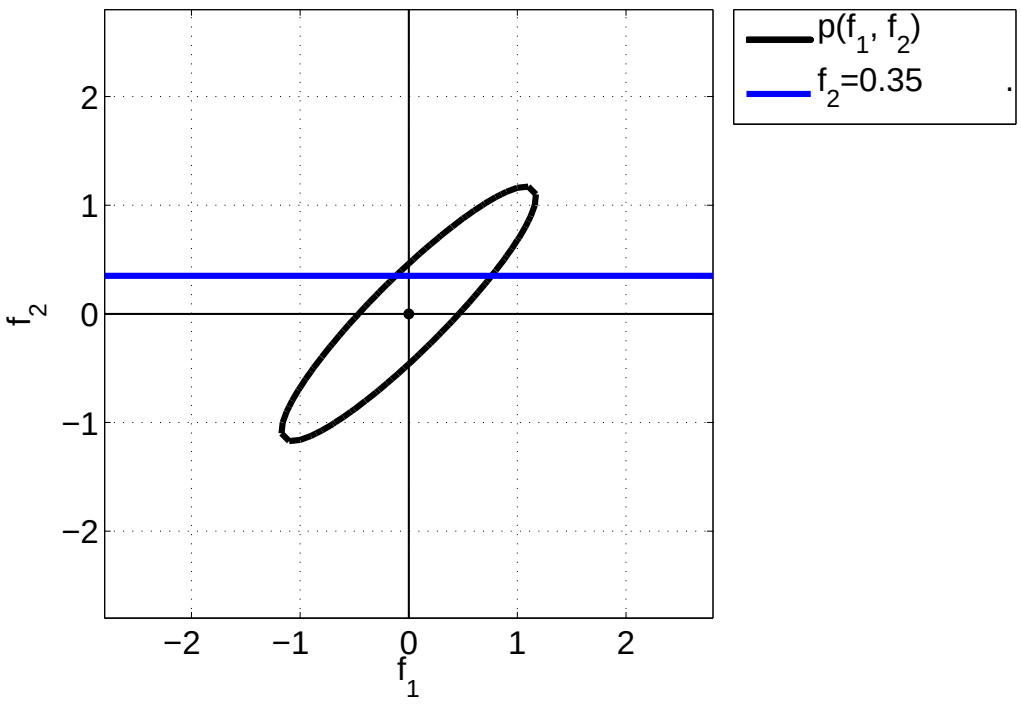


Covariance between f_1 and f_2:

$$\mathbf{K} = \begin{bmatrix} 1 & 0.92 \\ 0.92 & 1 \end{bmatrix}$$

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

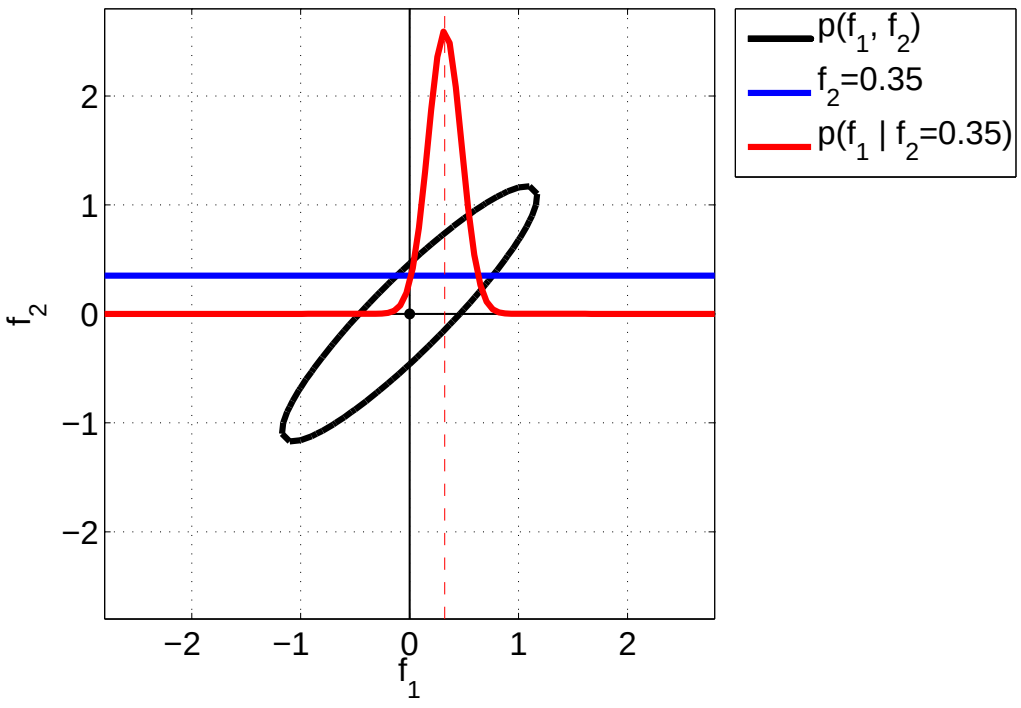


Covariance between f_1 and f_2:

$$\mathbf{K} = \begin{bmatrix} 1 & 0.92 \\ 0.92 & 1 \end{bmatrix}$$

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

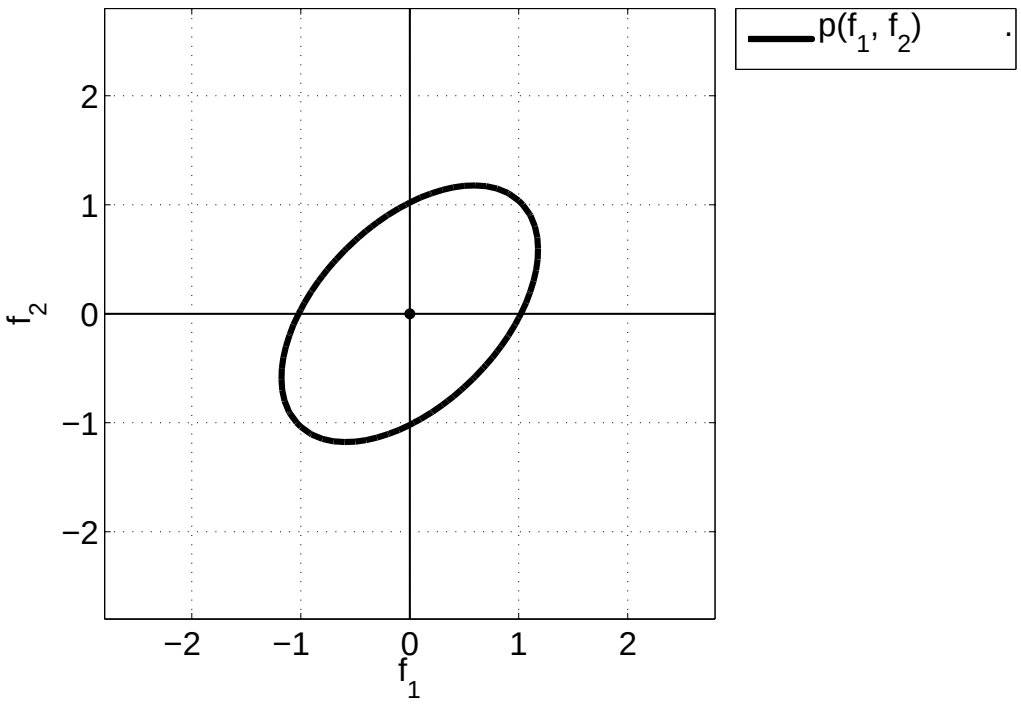


Covariance between f_1 and f_2:

$$\mathbf{K} = \begin{bmatrix} 1 & 0.92 \\ 0.92 & 1 \end{bmatrix}$$

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

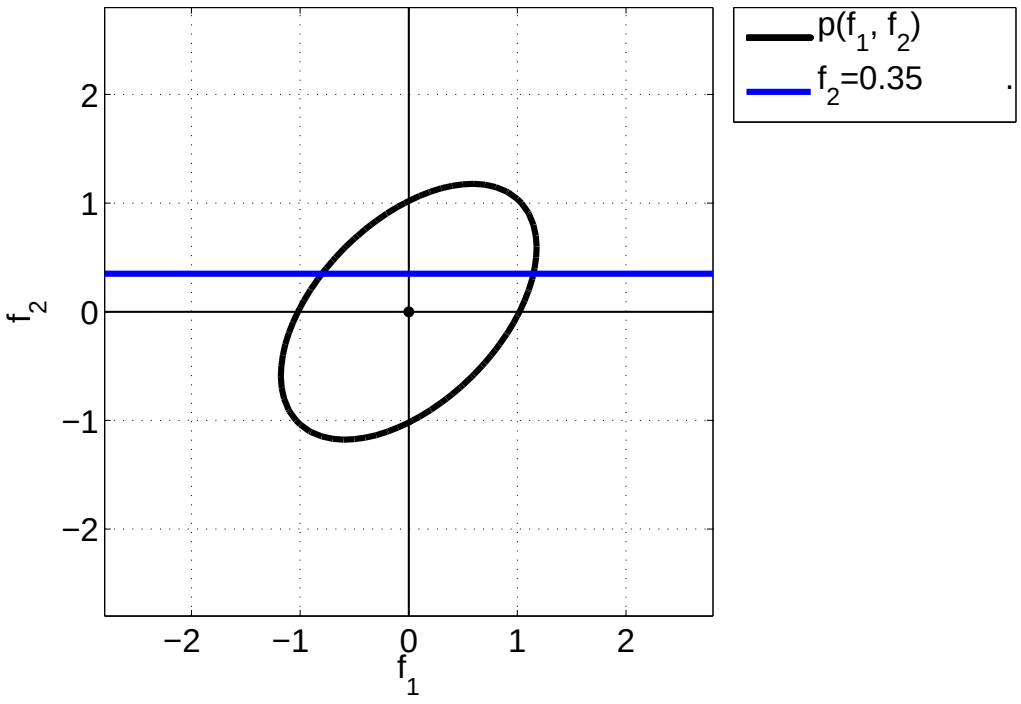


Covariance between f_1 and f_2:

$$\mathbf{K} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

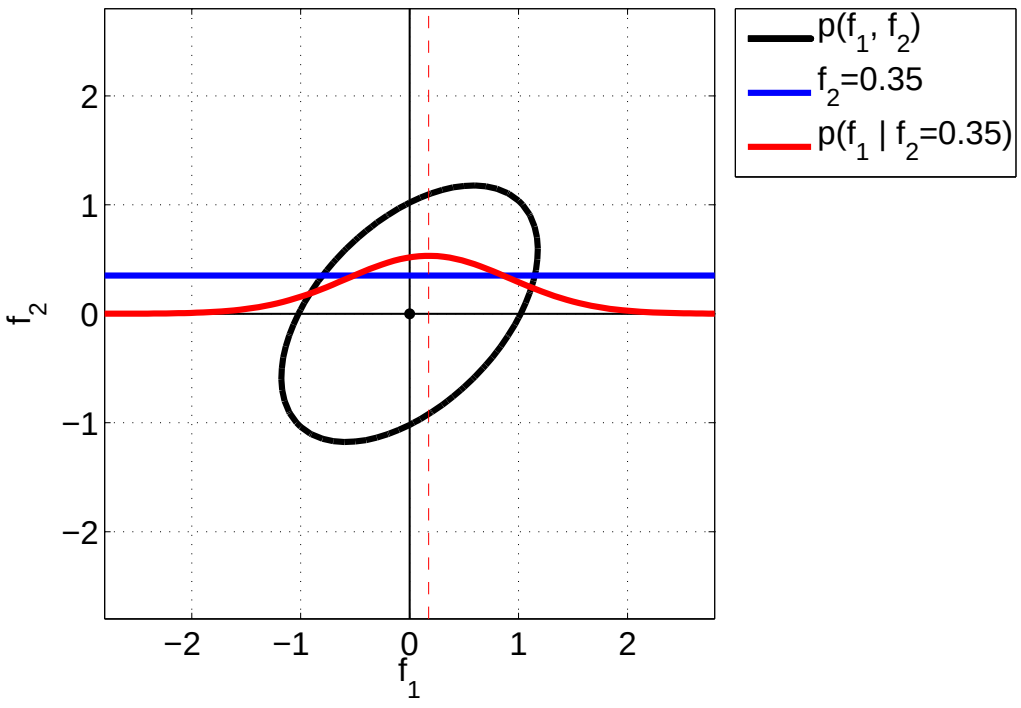


Covariance between f_1 and f_2:

$$\mathbf{K} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

Gaussian distribution

Probability model: $p(f_1, f_2) \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$

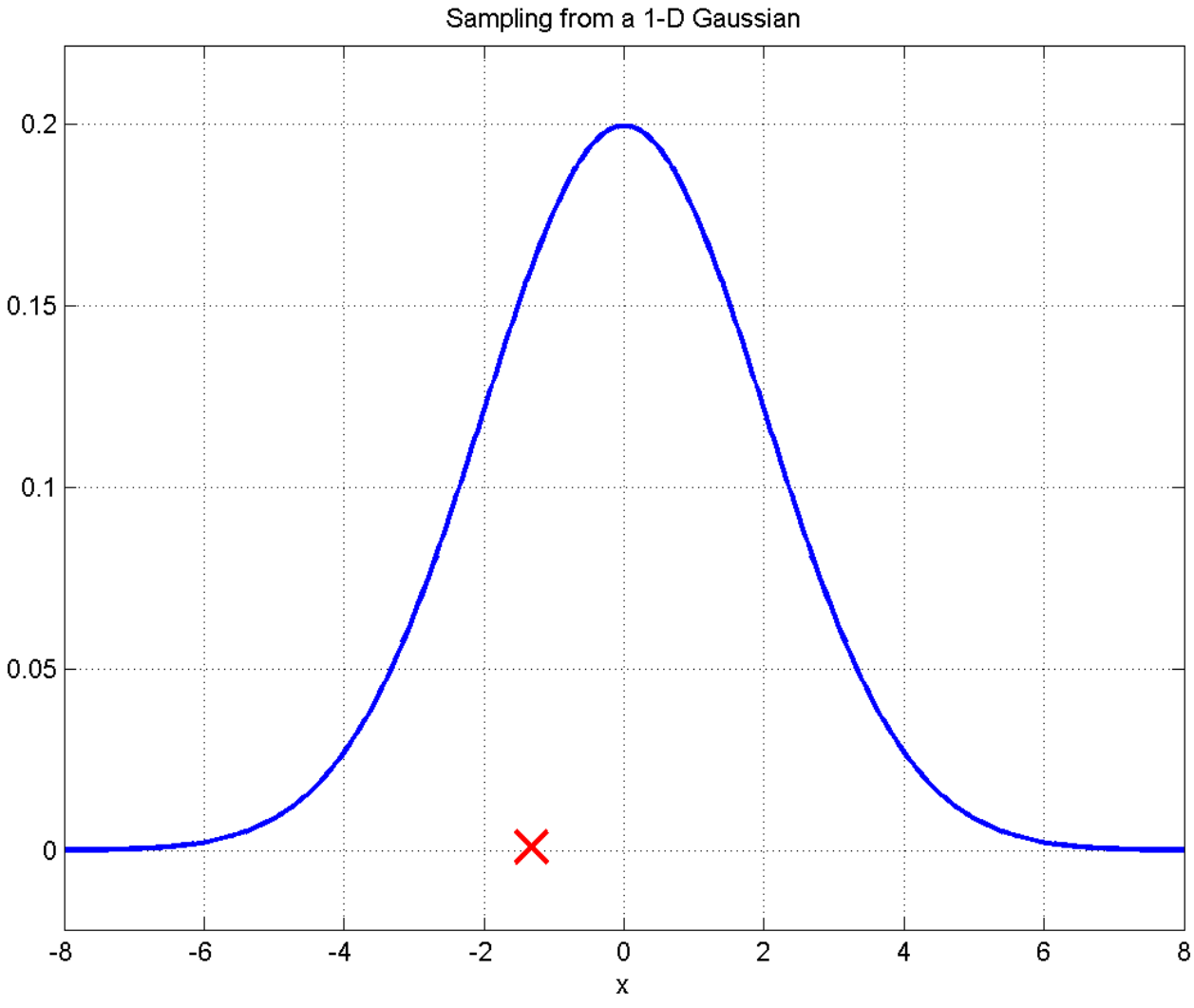


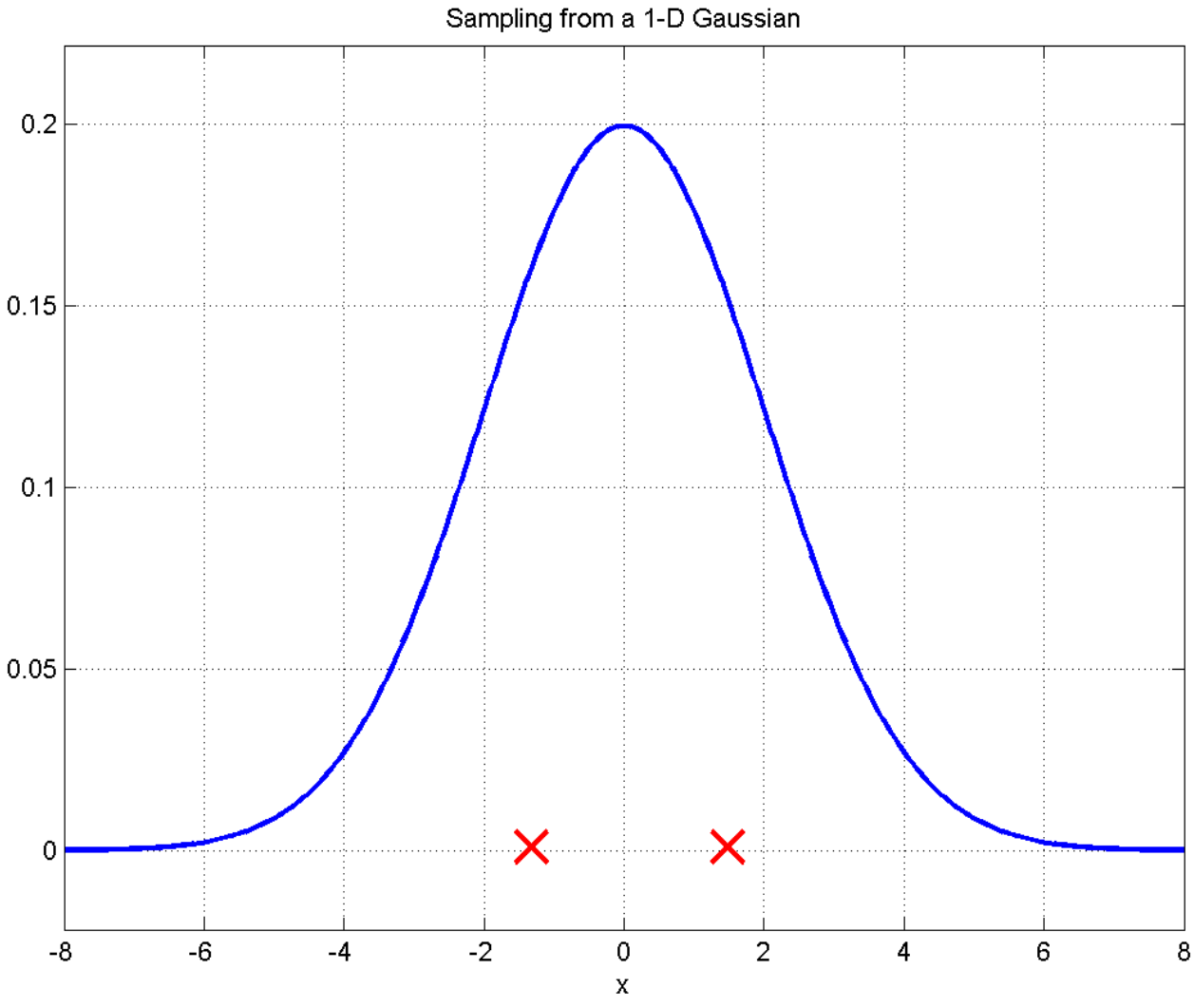
Covariance between f_1 and f_2:

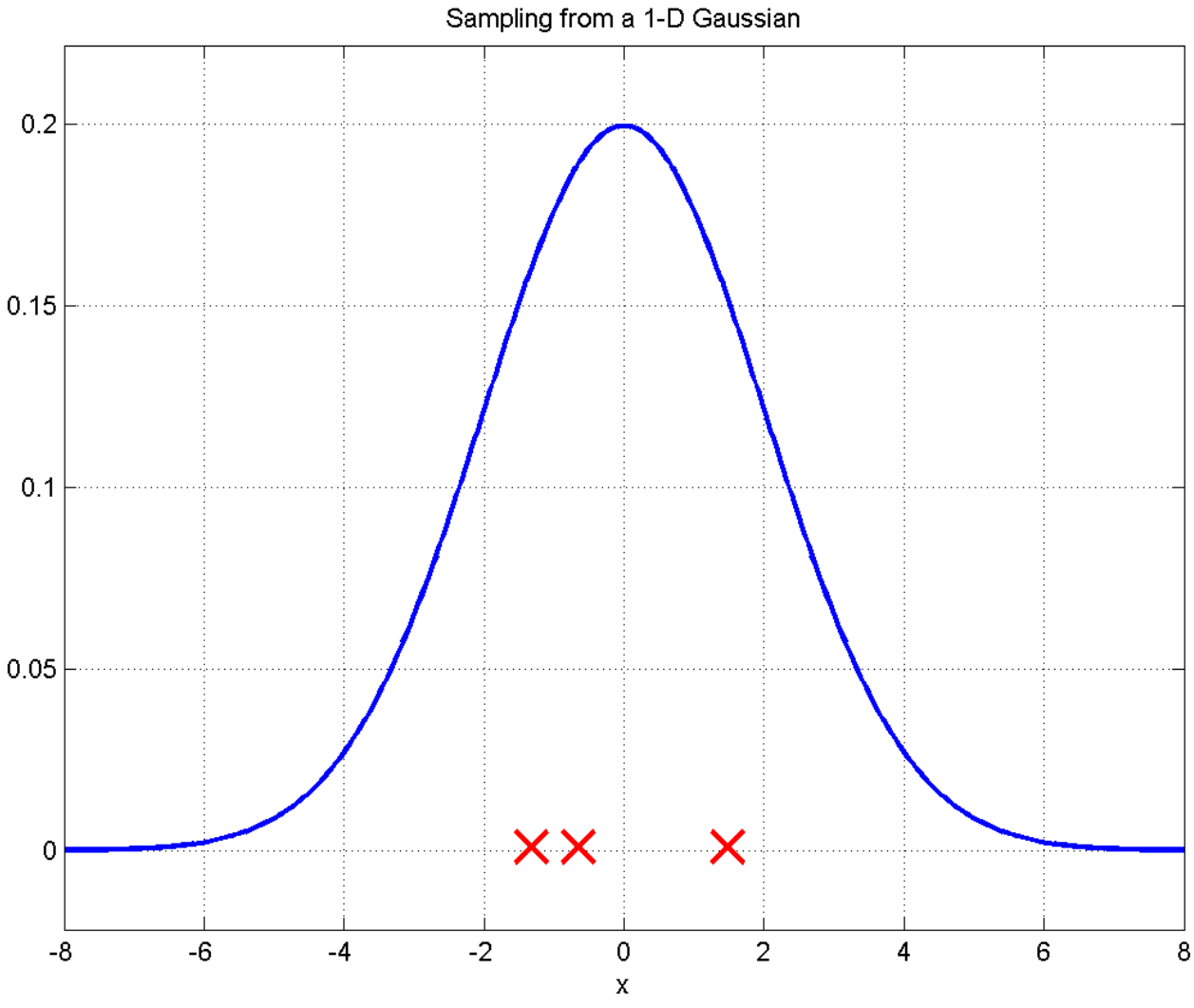
$$\mathbf{K} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

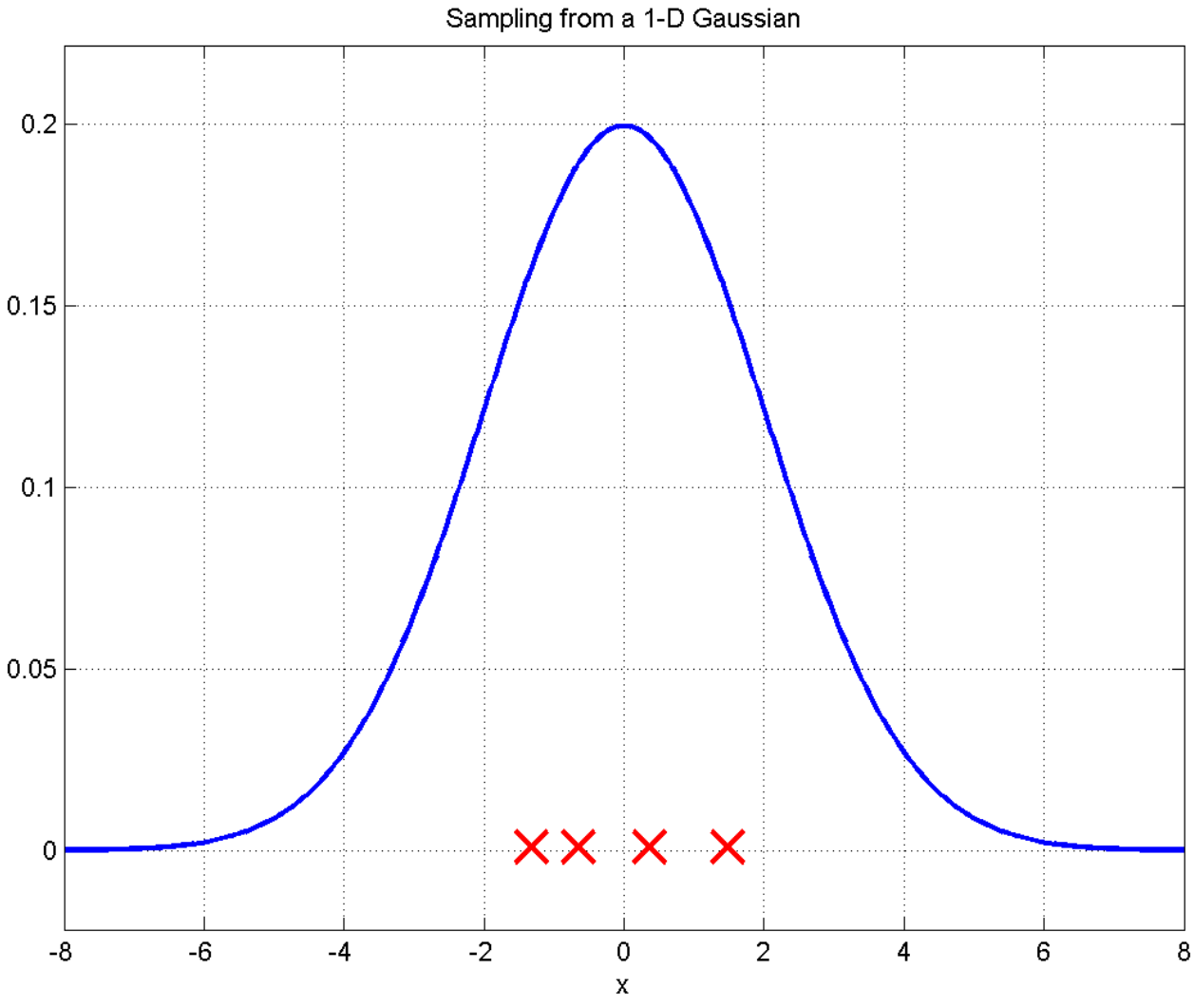
Introducing Gaussian Processes:

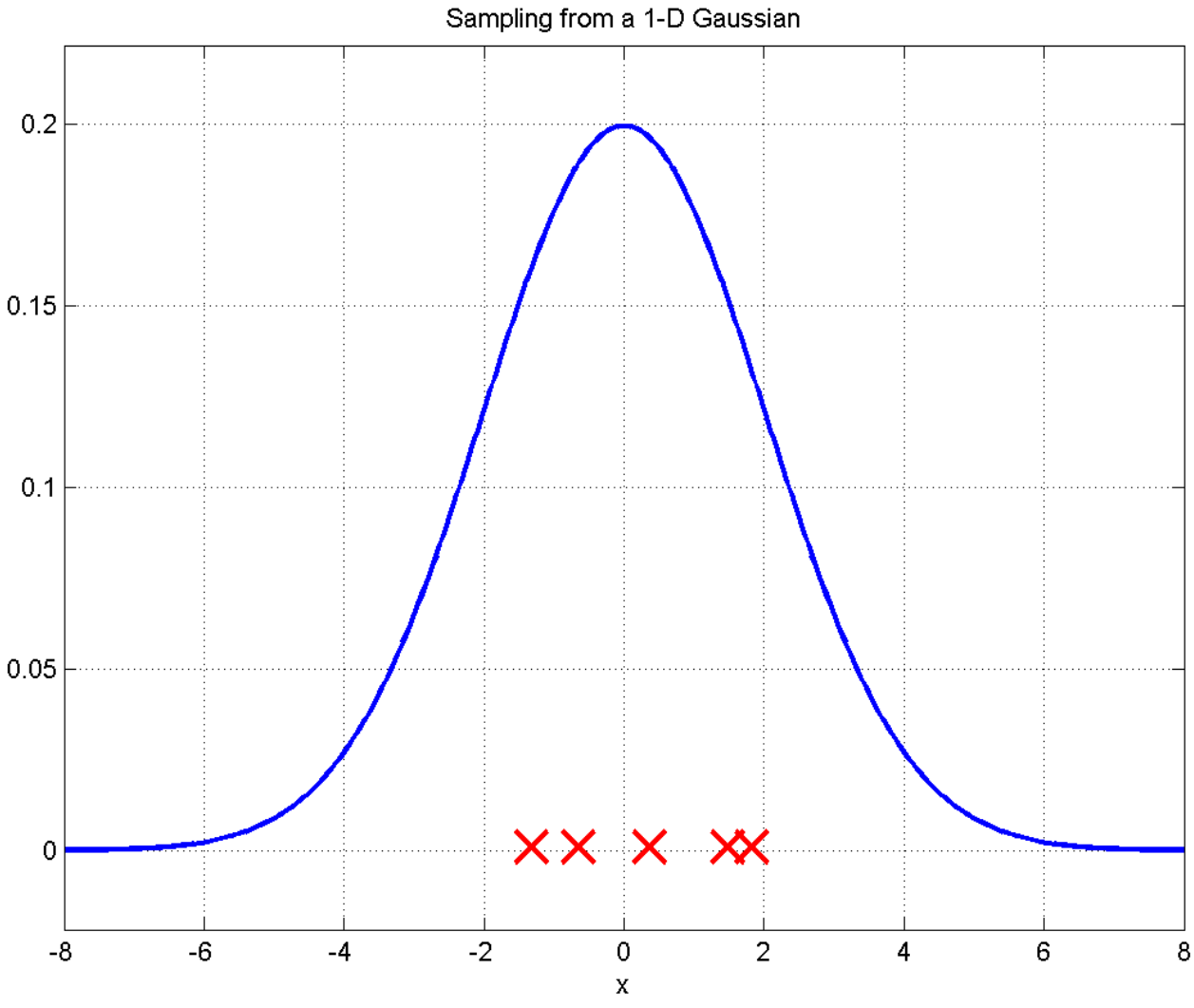
- A Gaussian distribution depends on a mean and a covariance matrix.
- A Gaussian process depends on a mean and a covariance function.

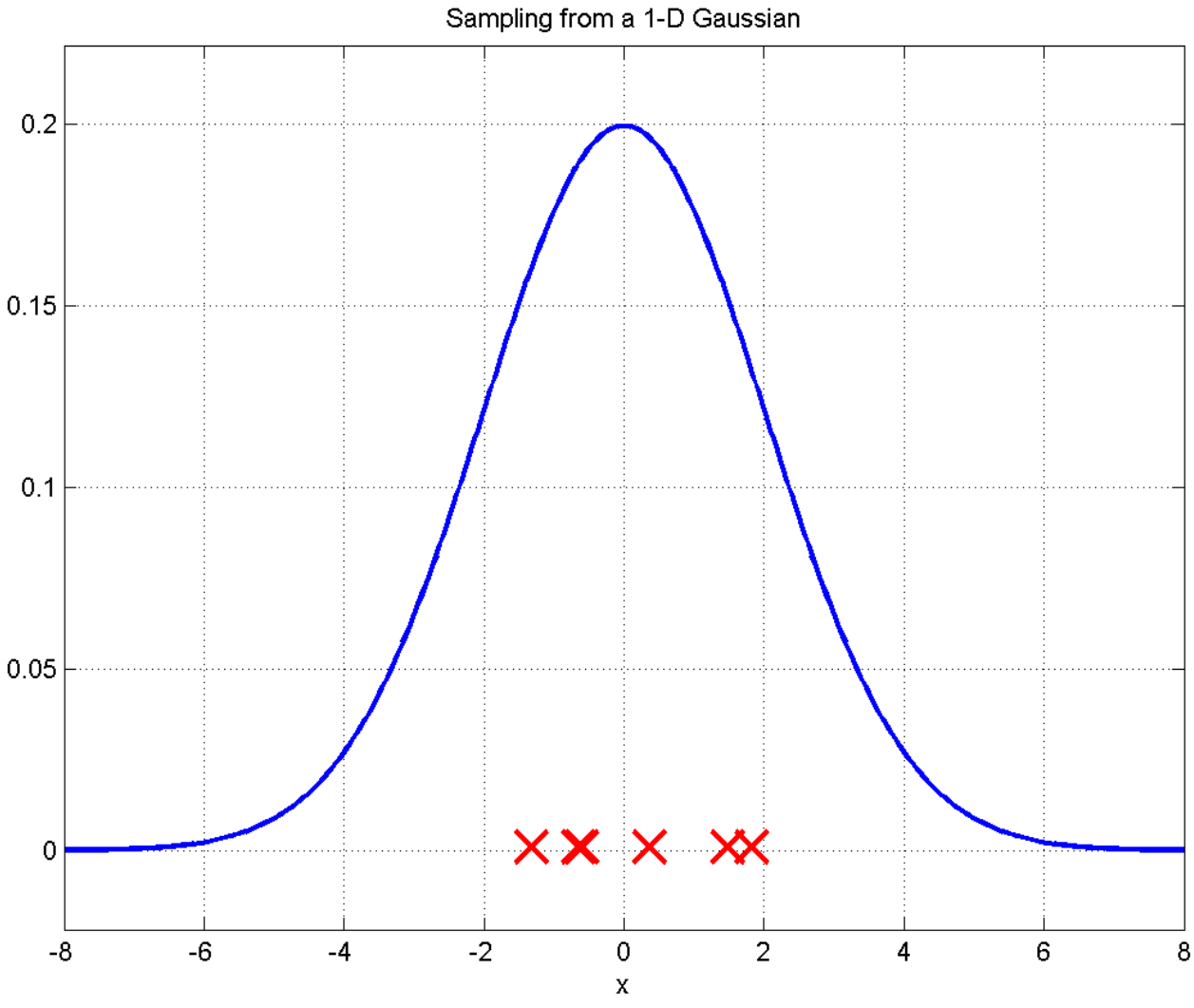

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

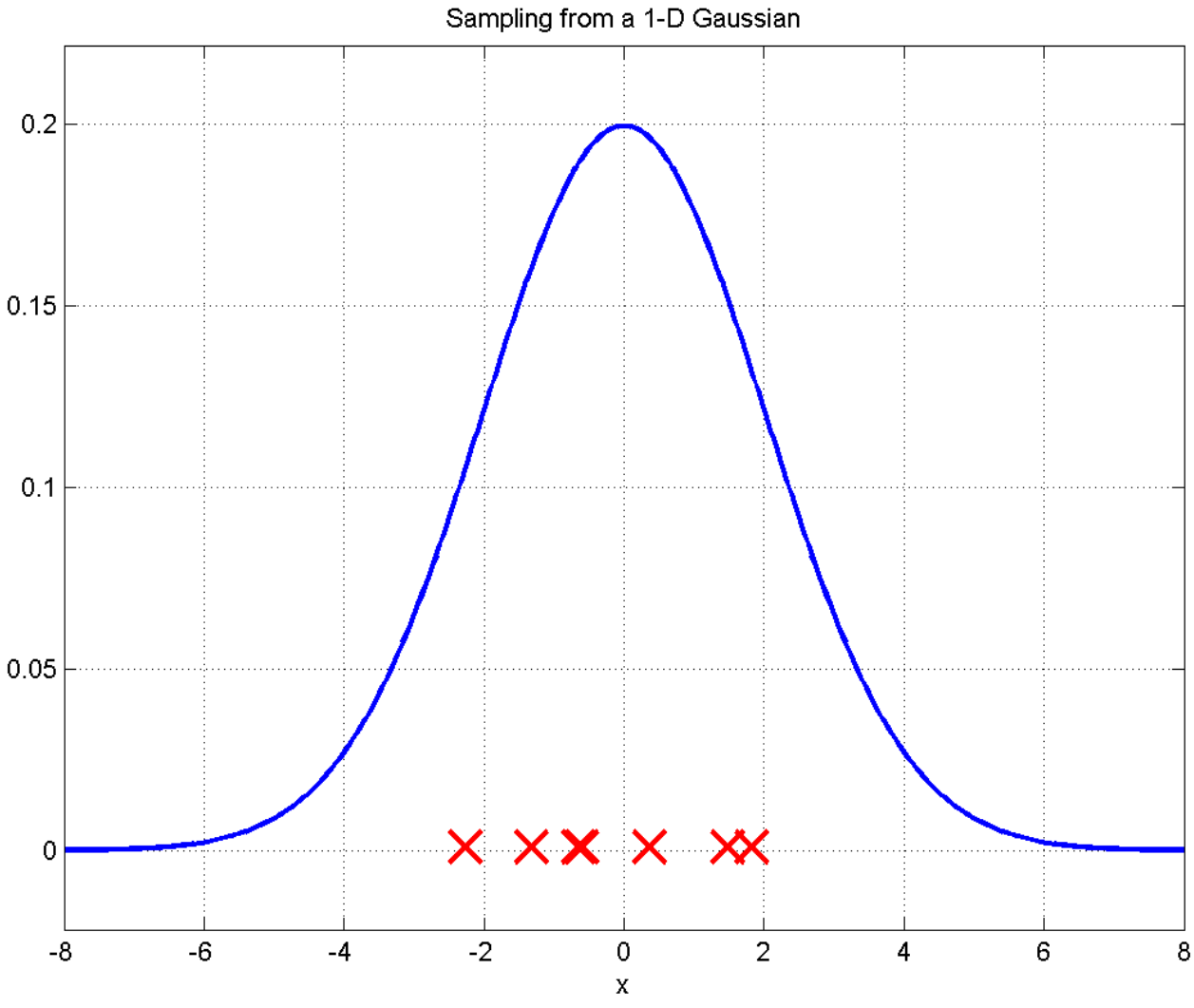

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

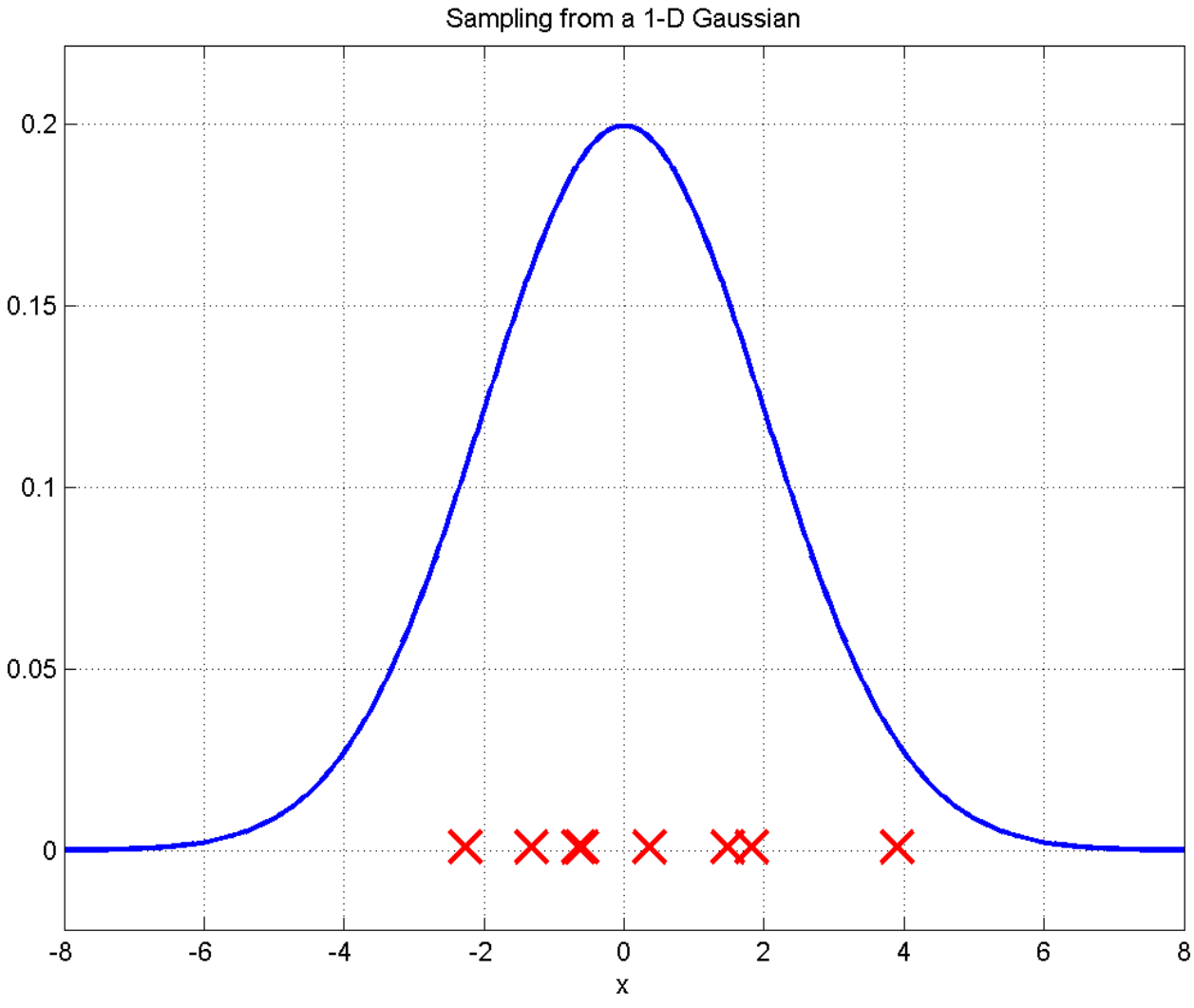

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

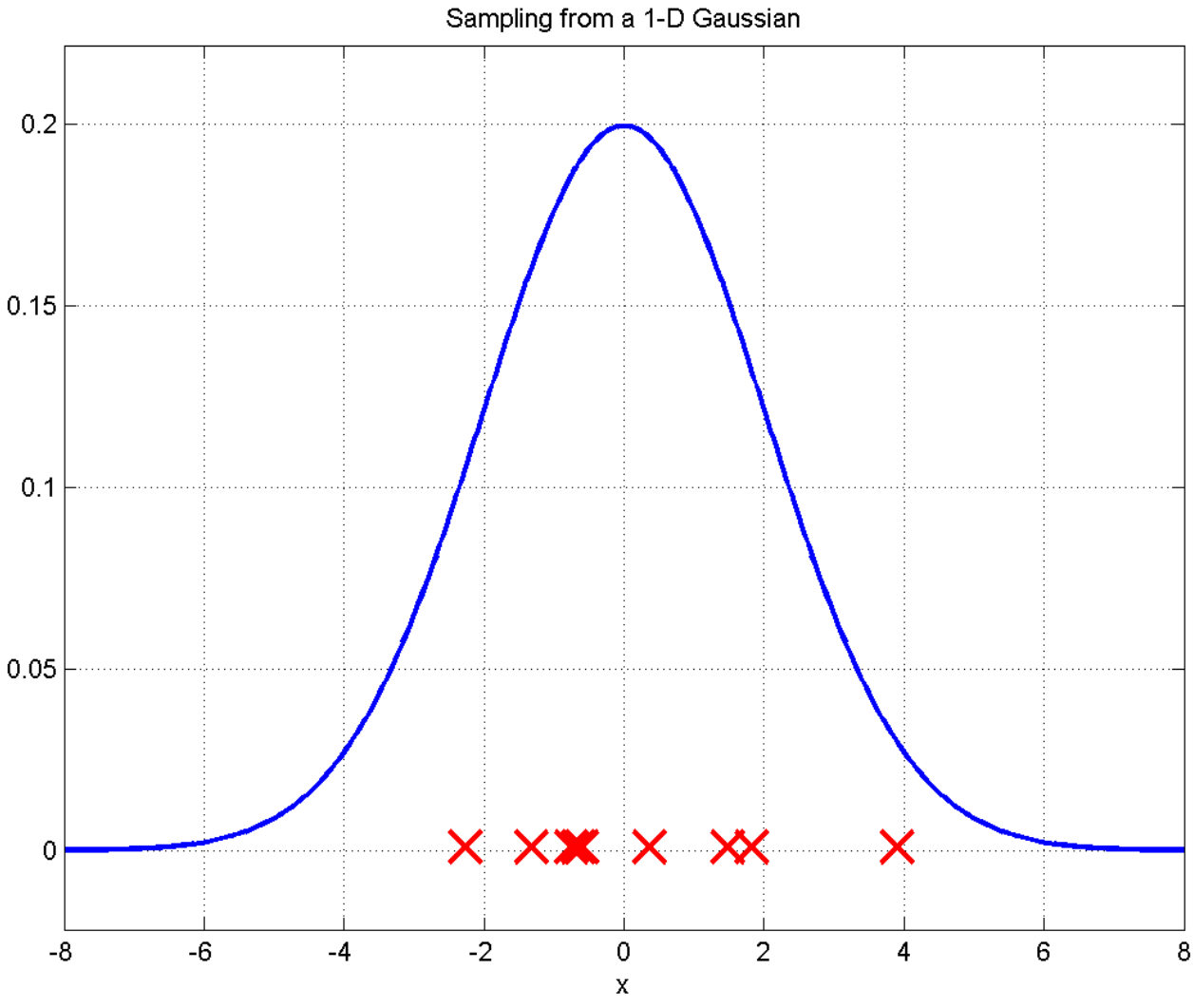

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

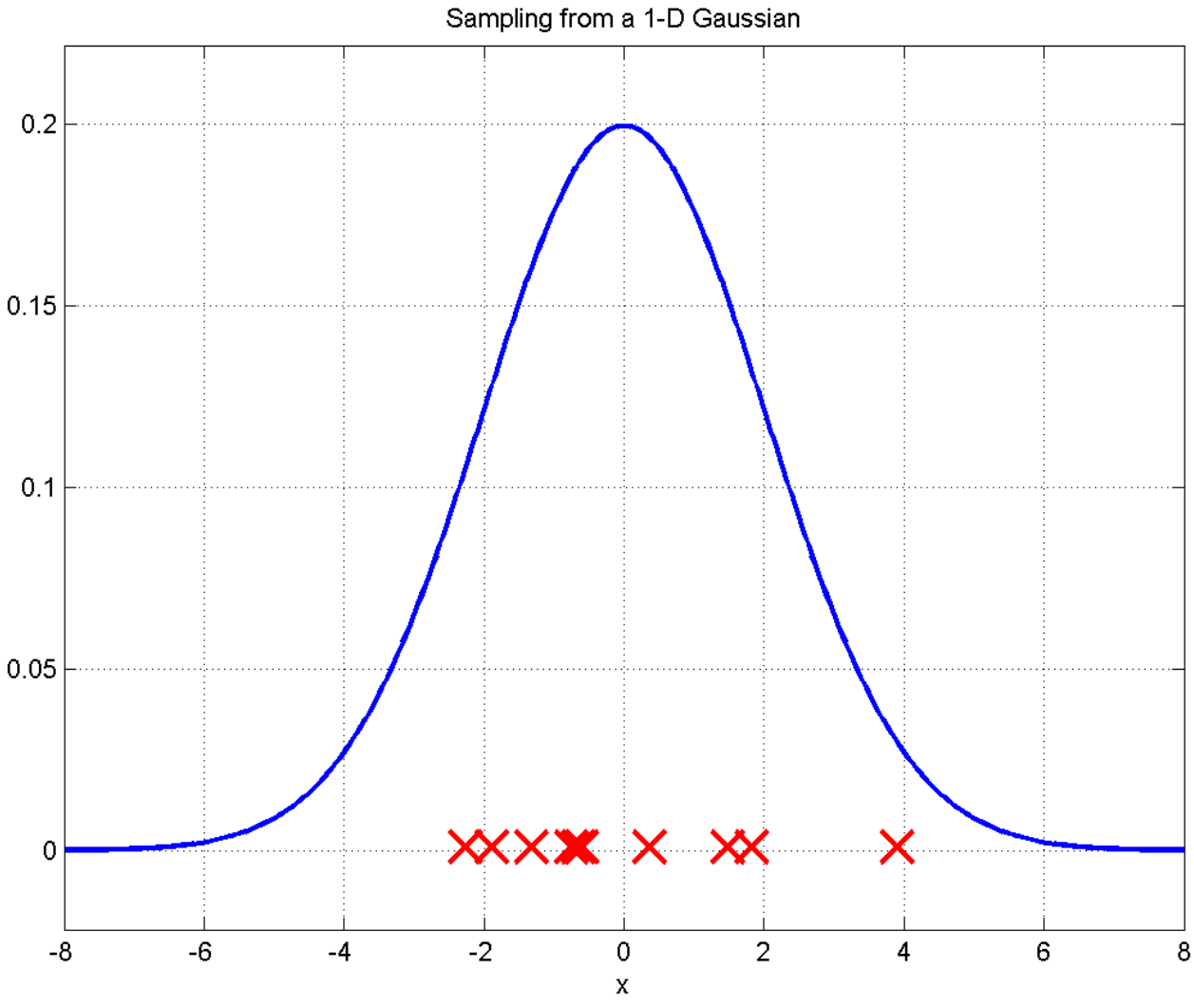

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

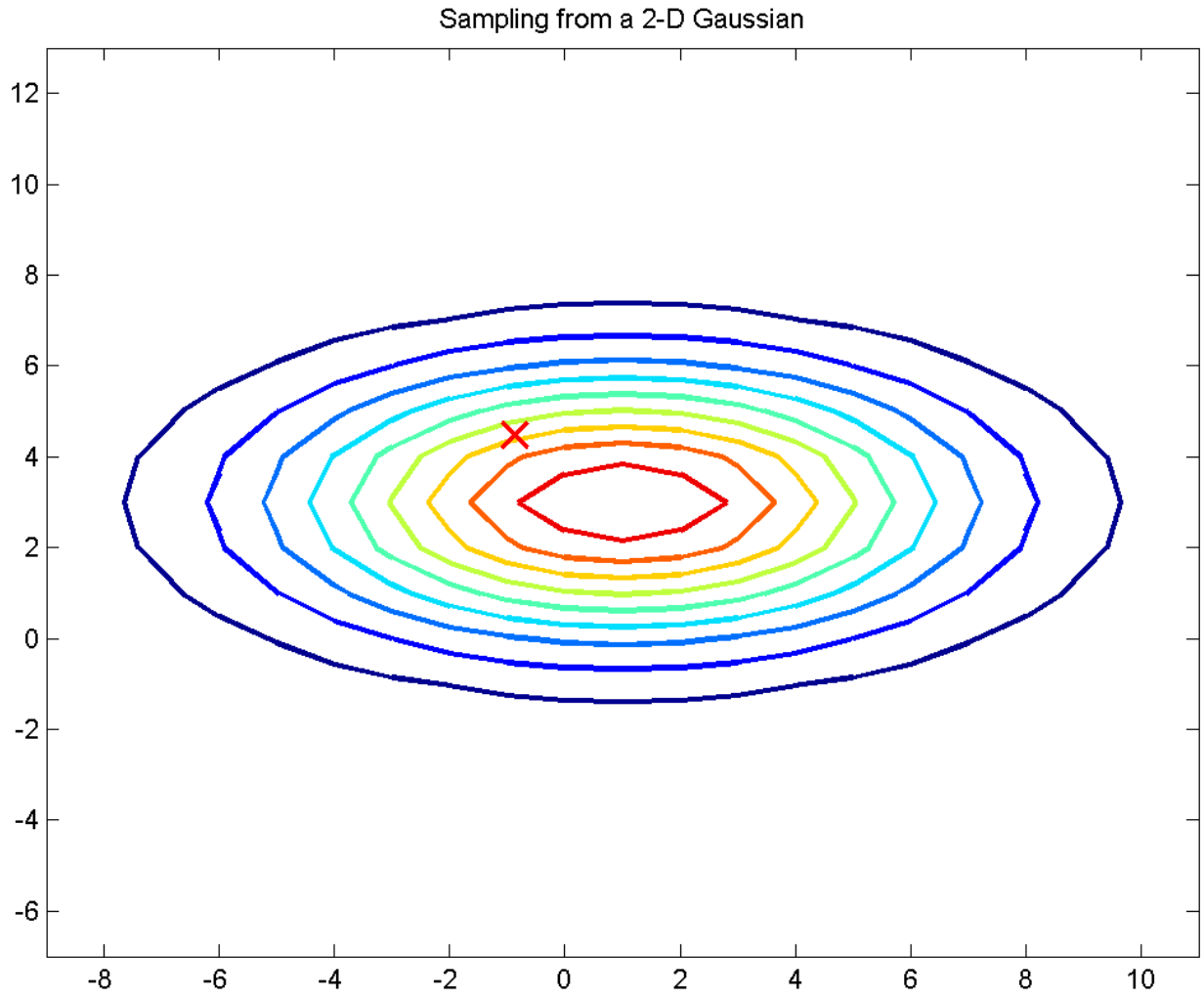

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

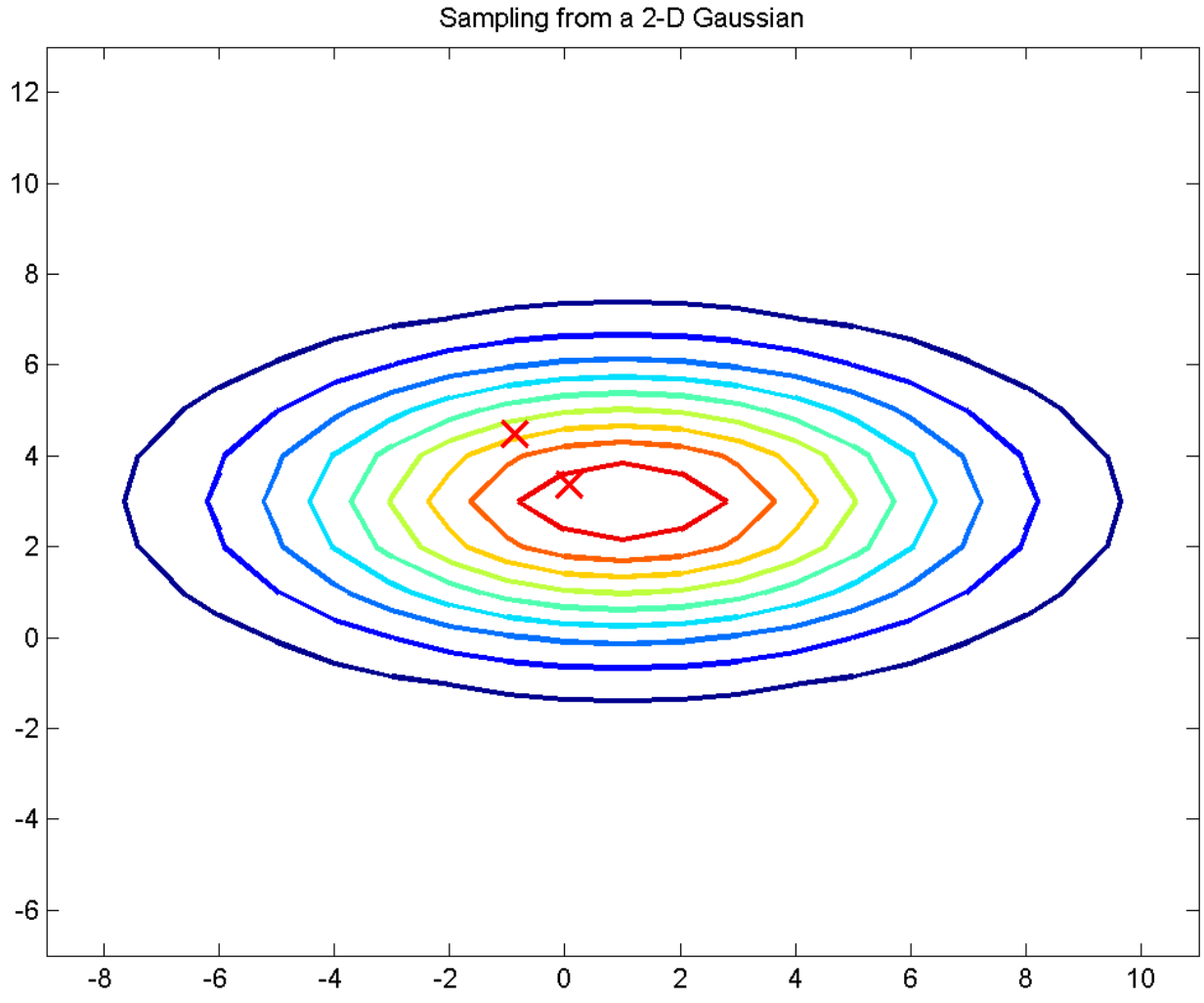

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

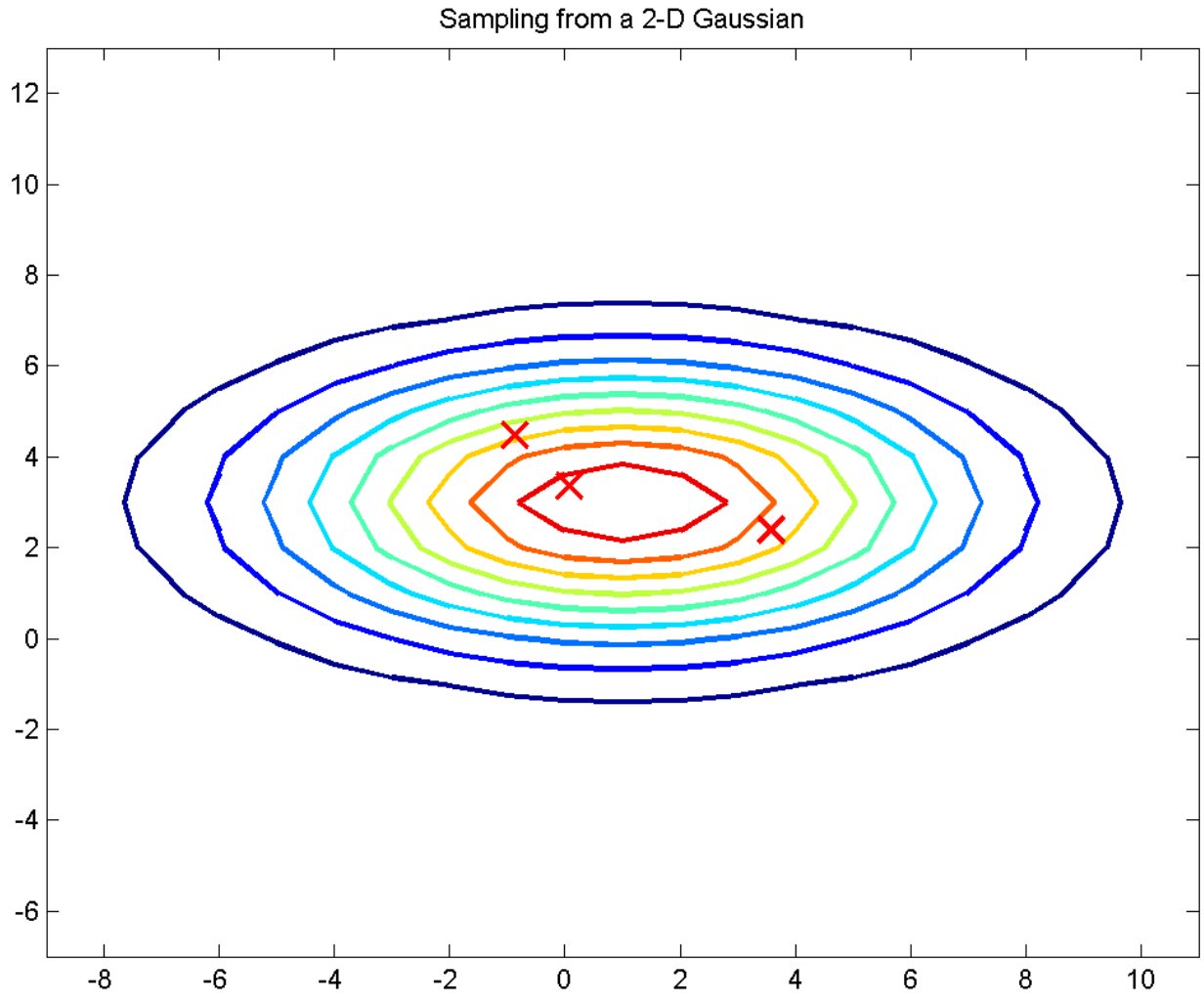

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

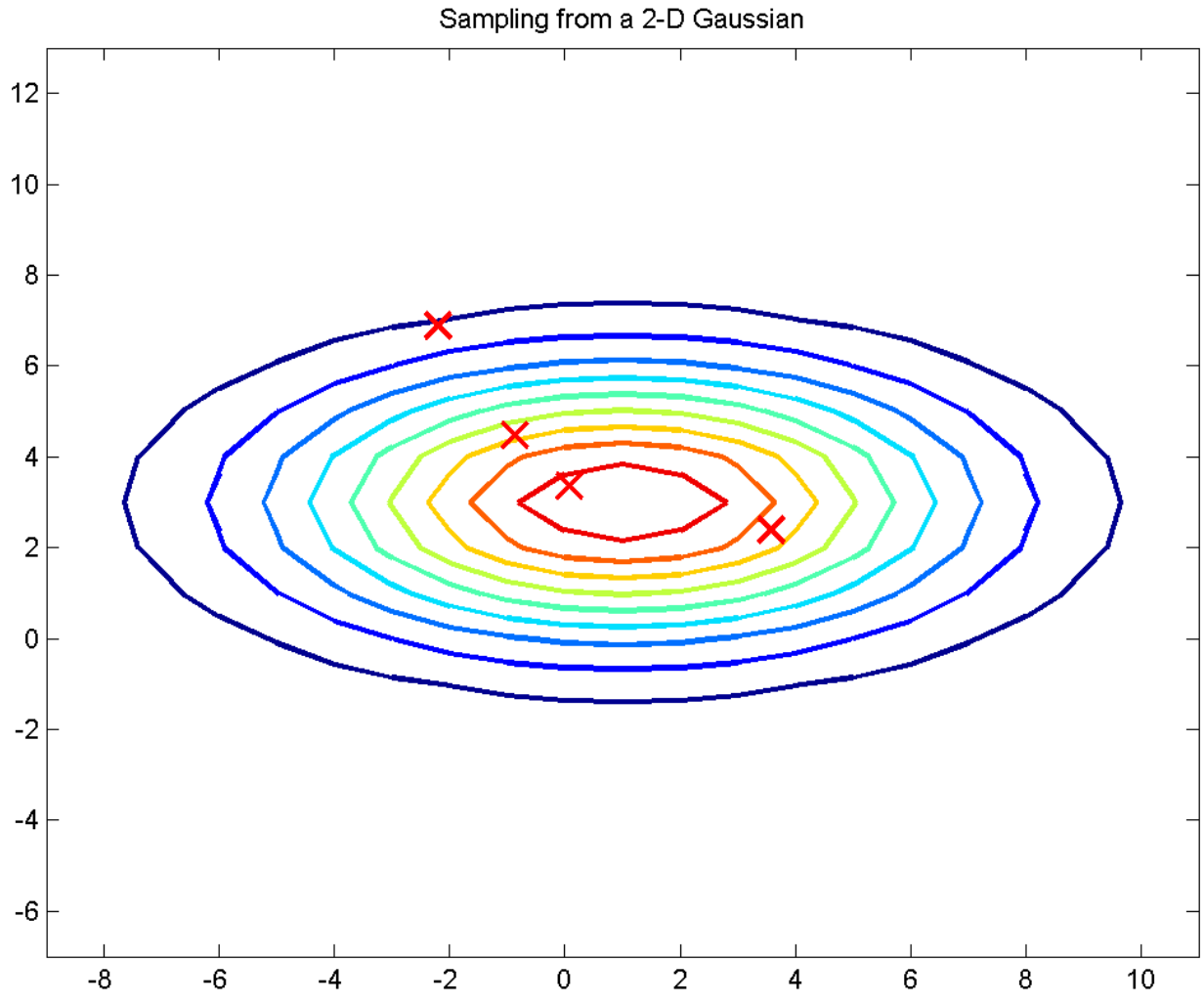

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

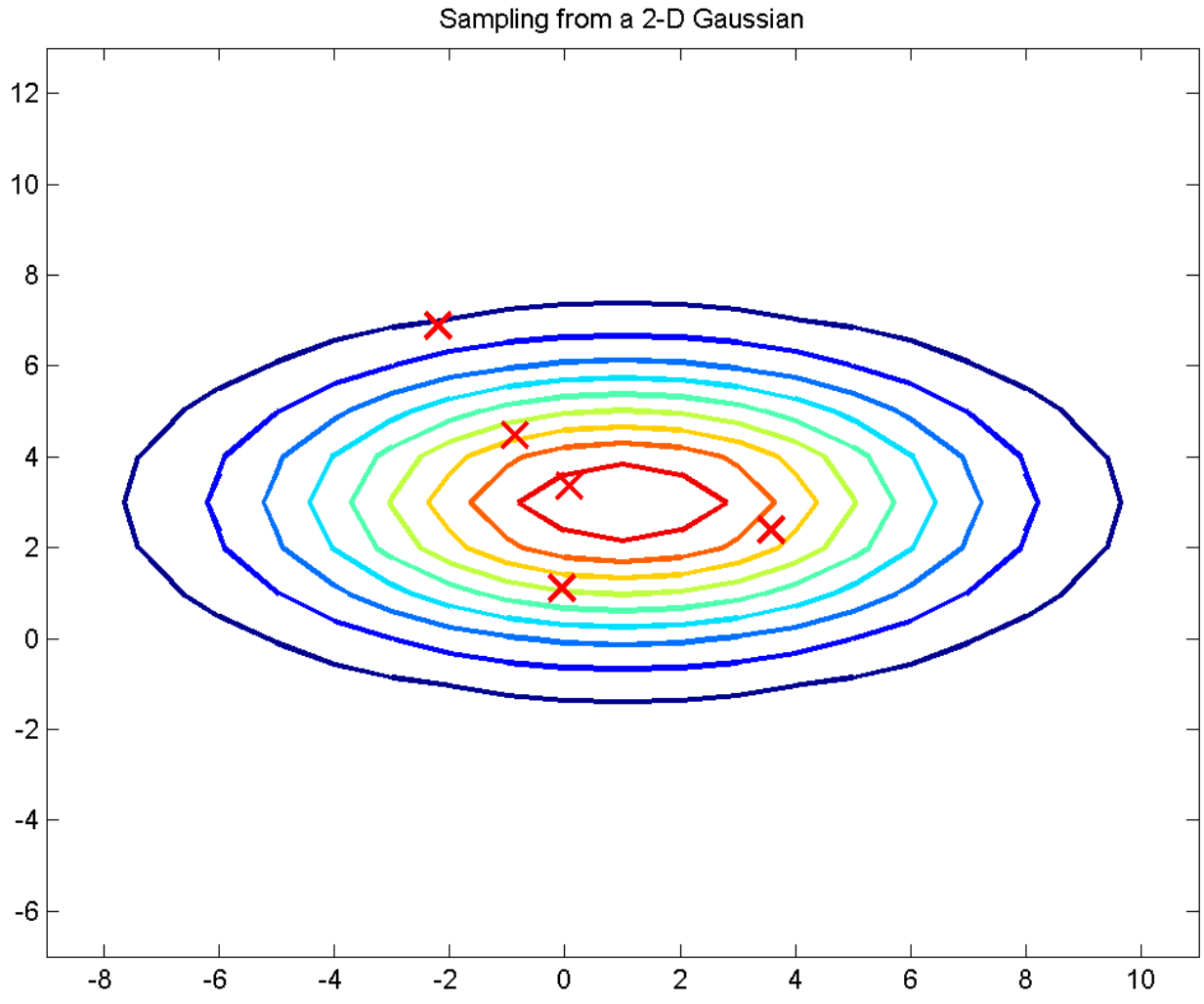

Sampling from a 1-D Gaussian
0.2
0.15
0.1
0.05
0
-8
-6
-4
-2
0
2
4
6
8
x

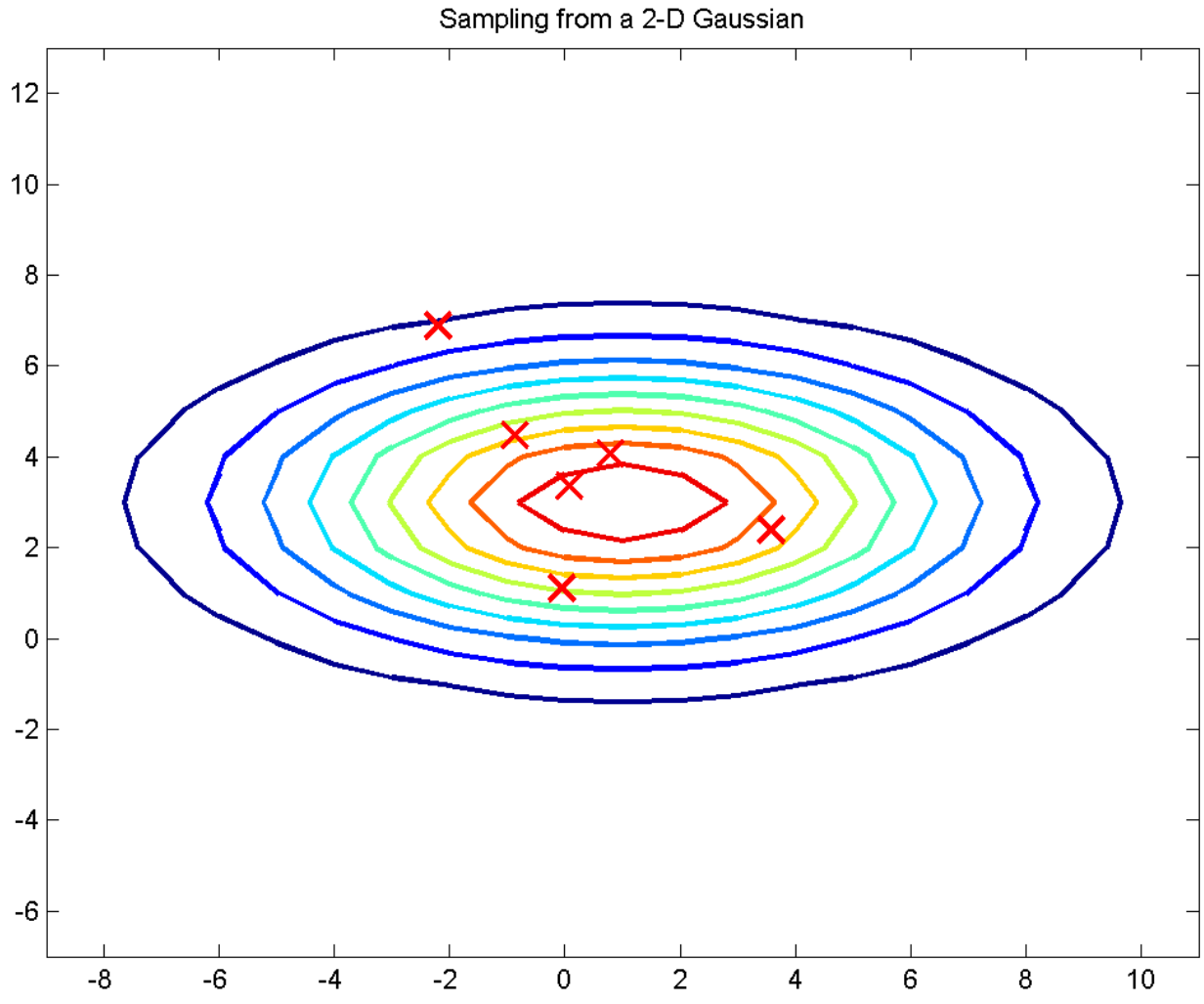

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

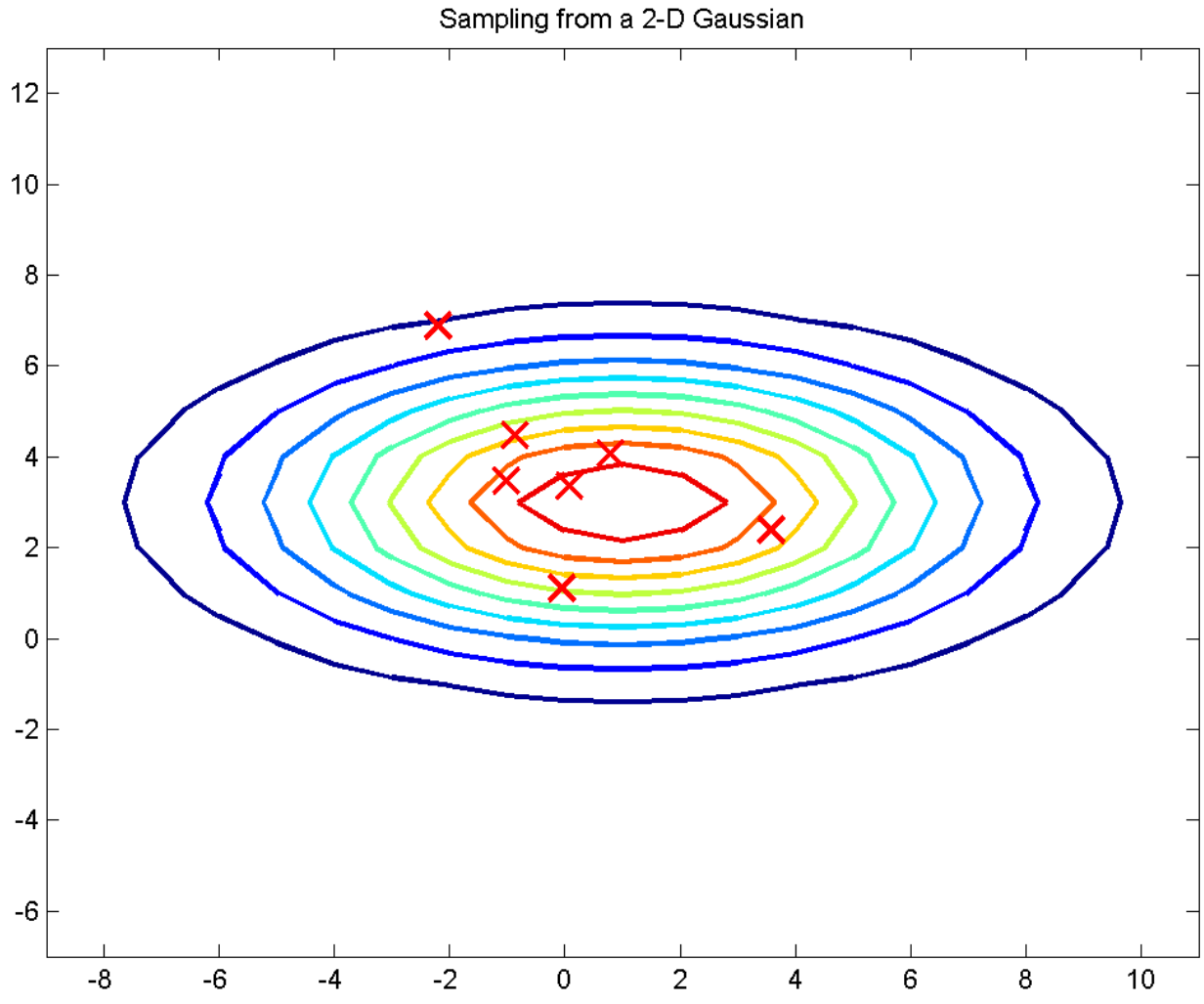

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

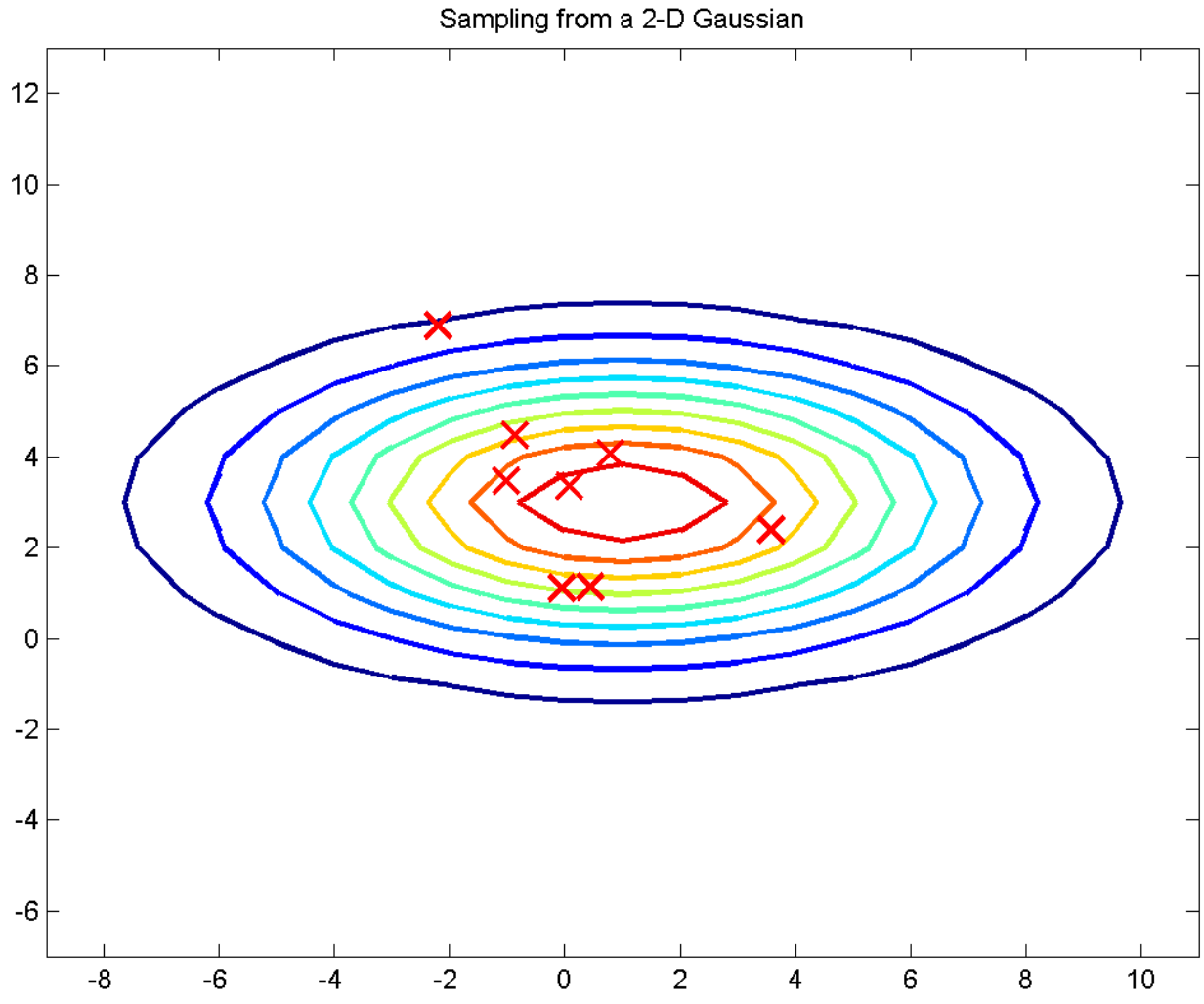

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

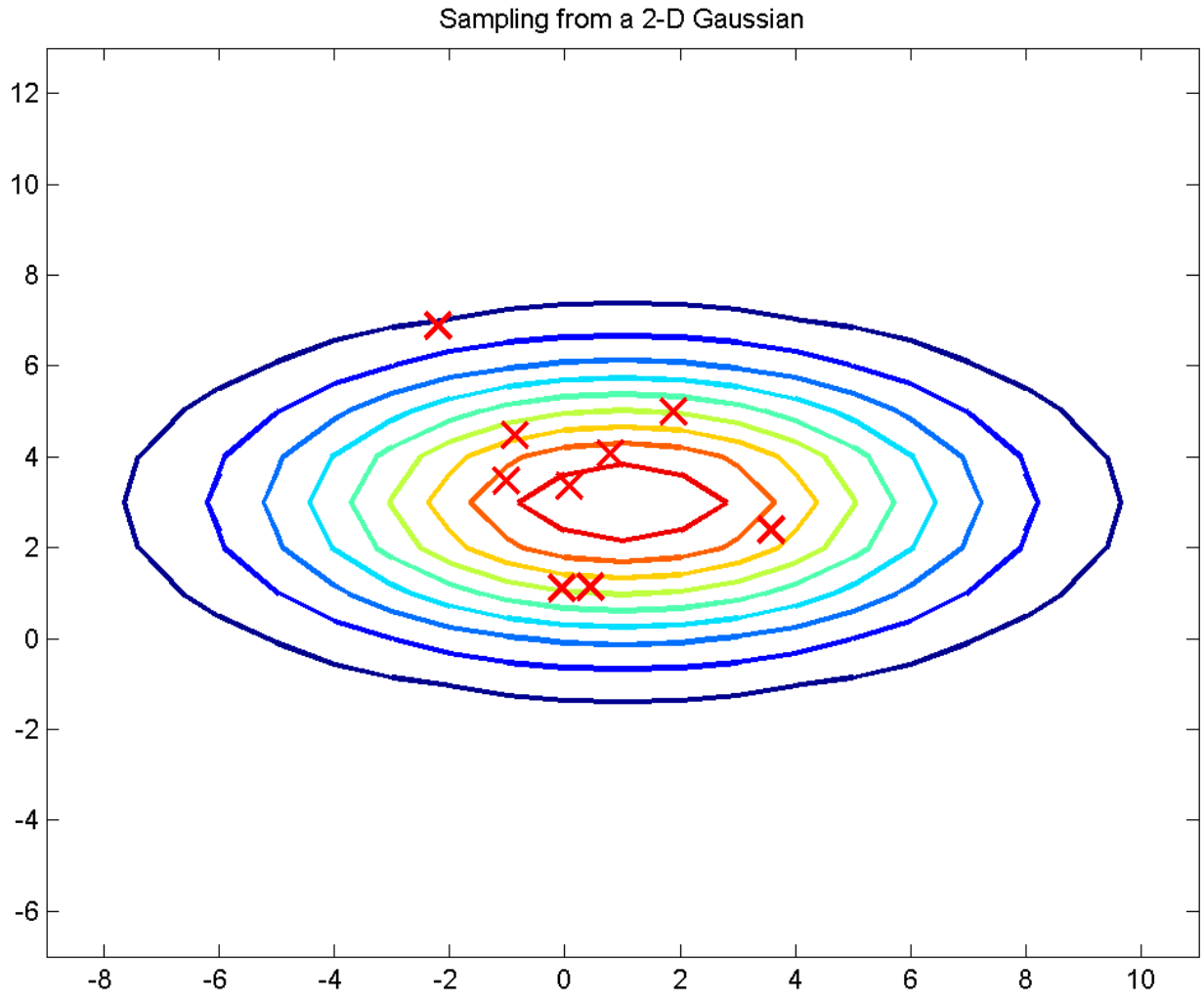

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

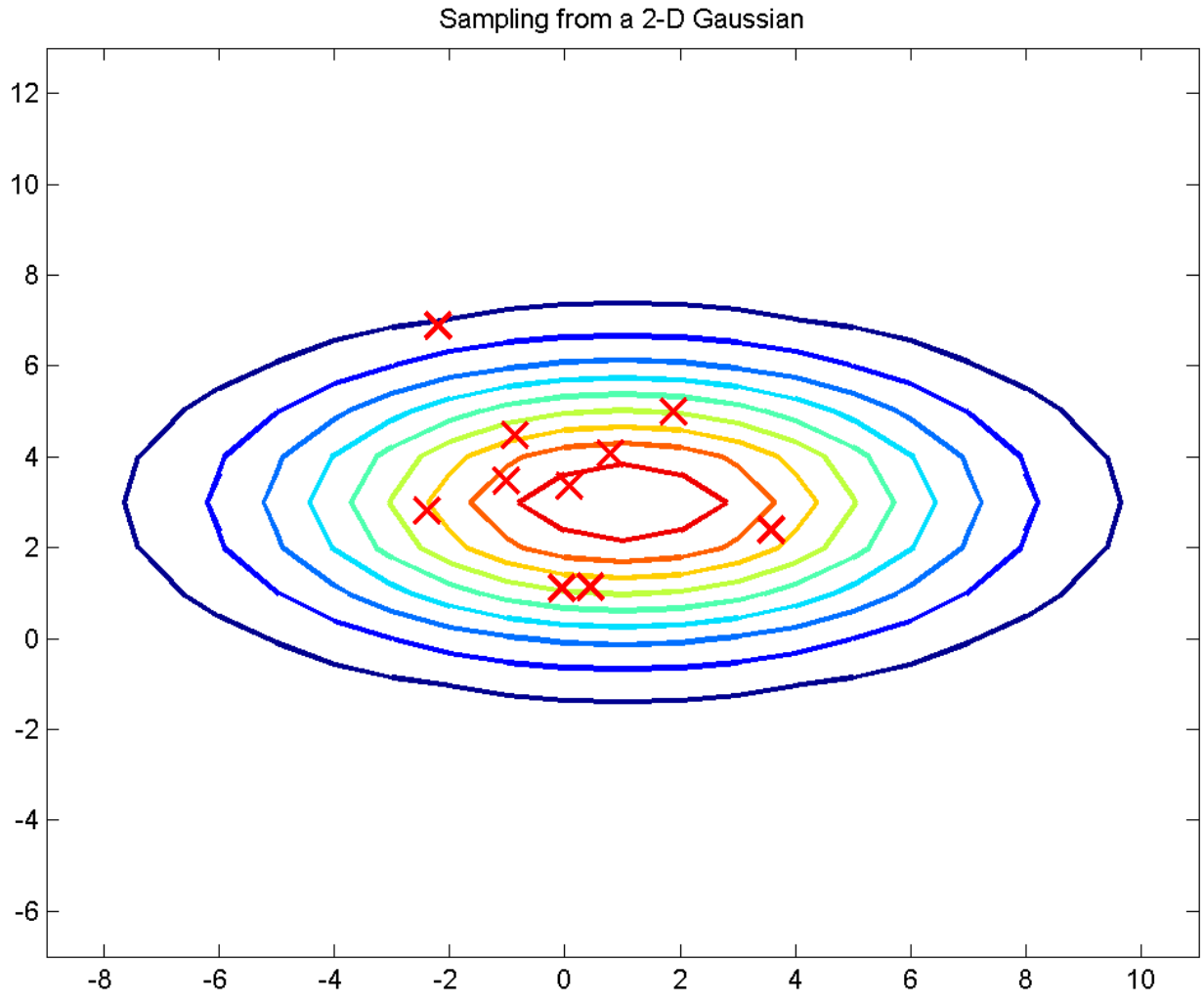

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

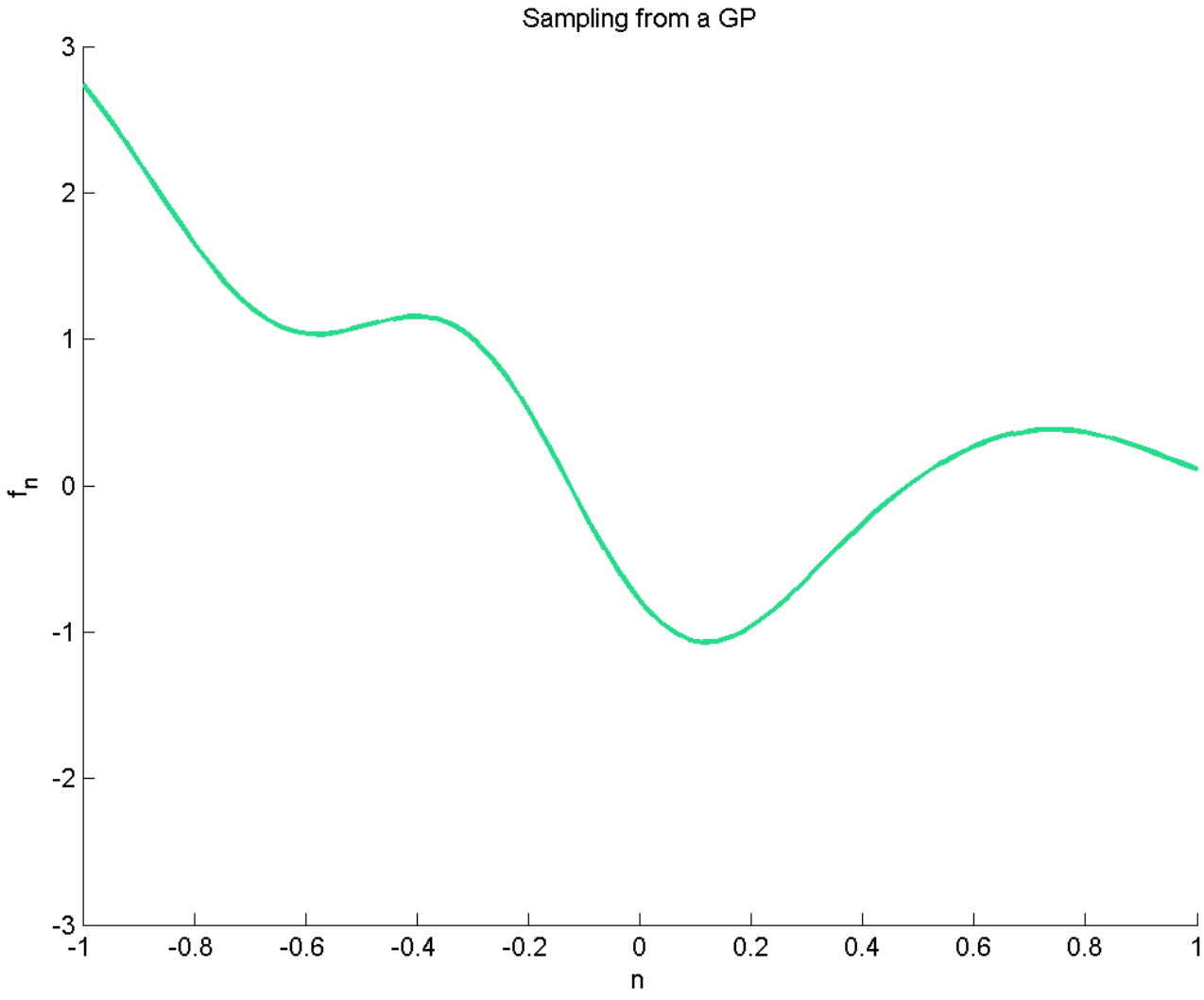

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

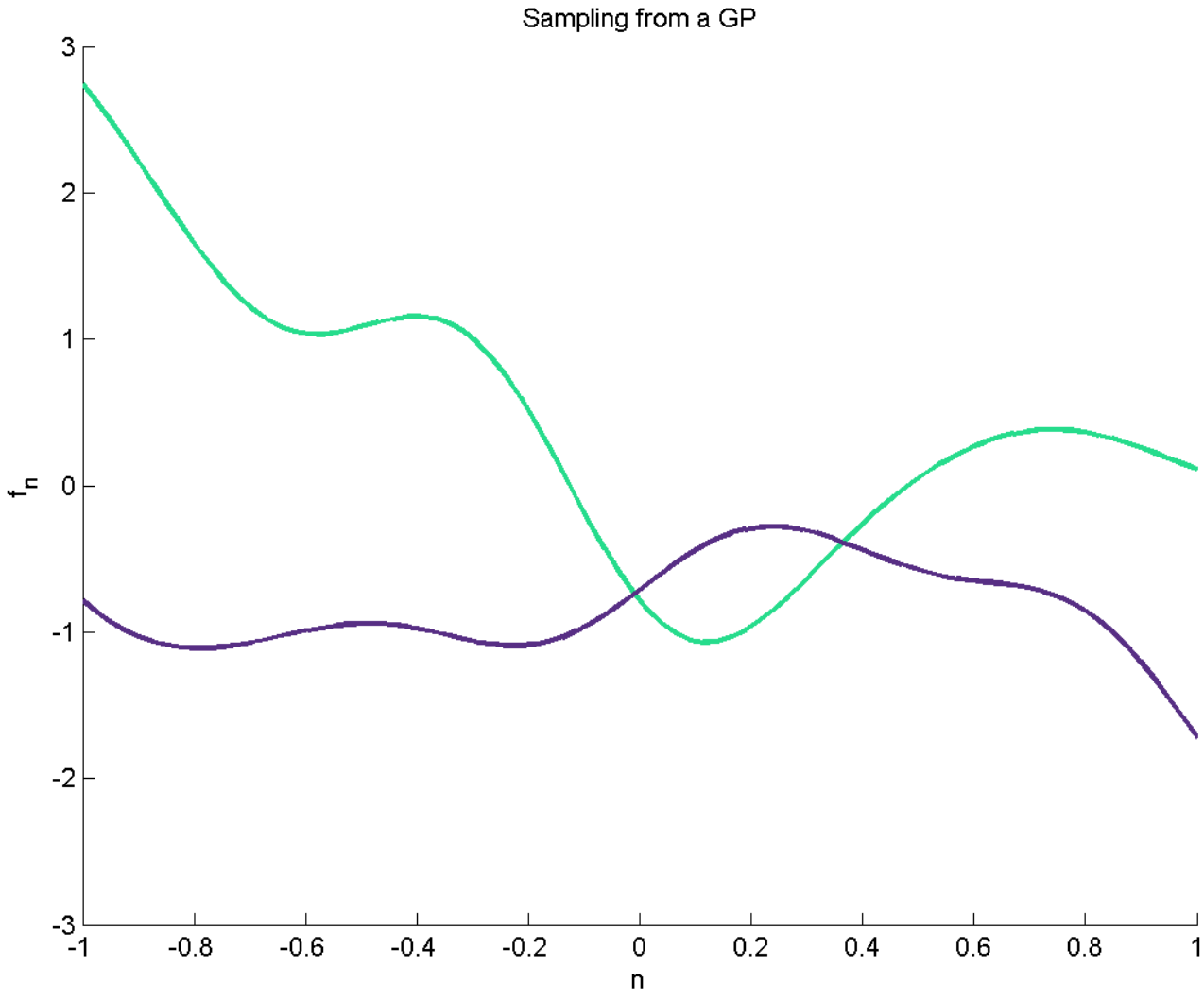

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

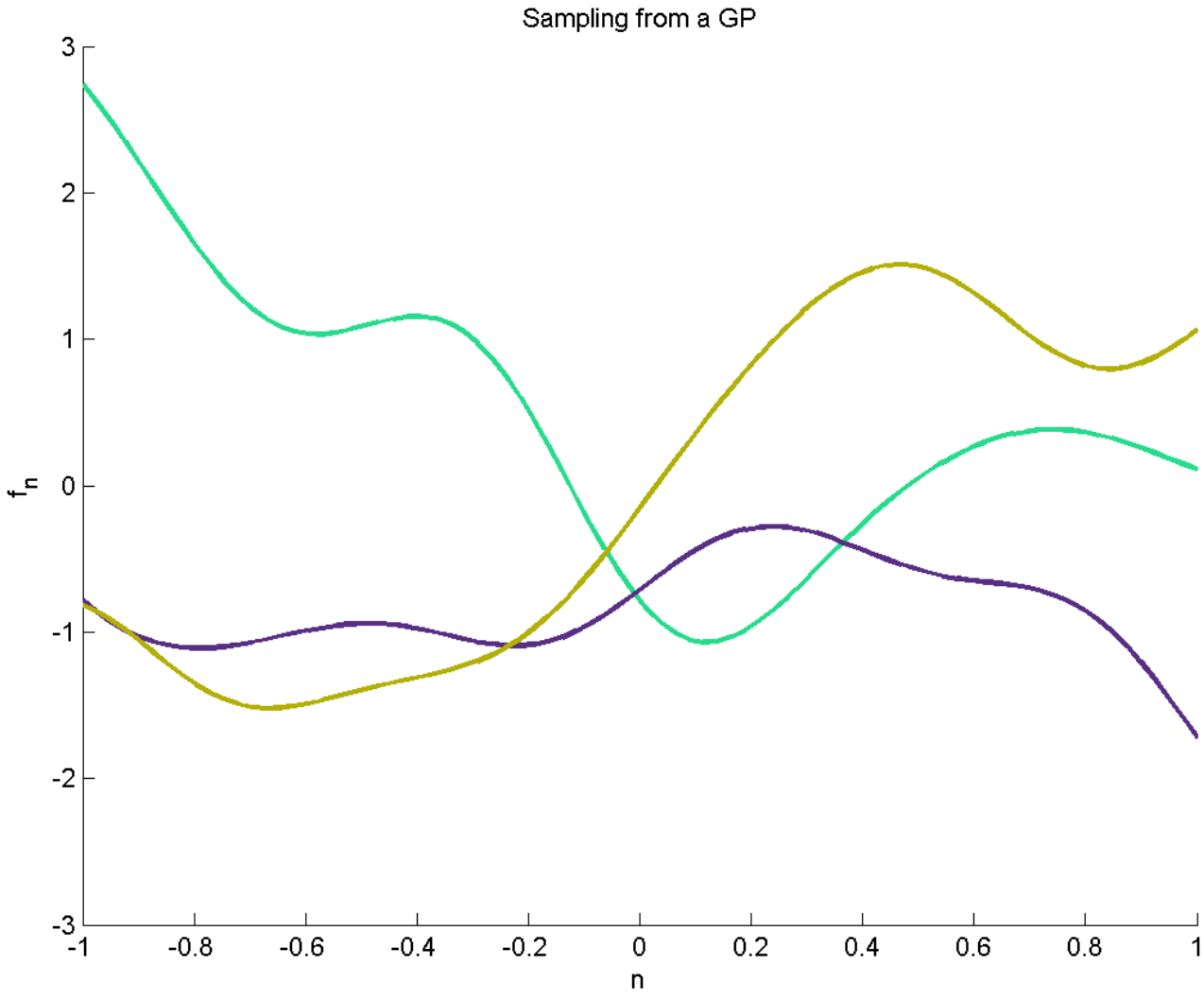

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

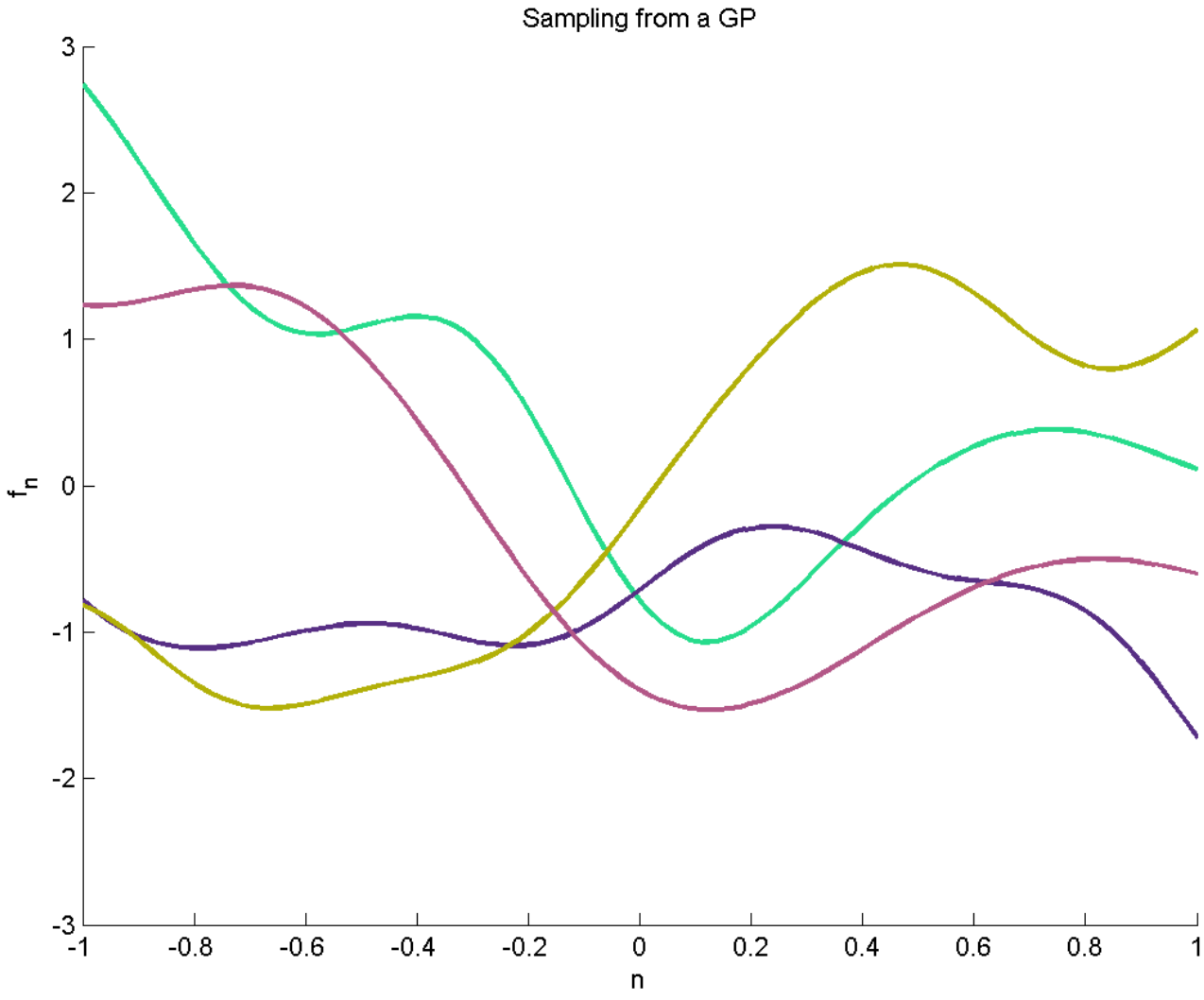

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

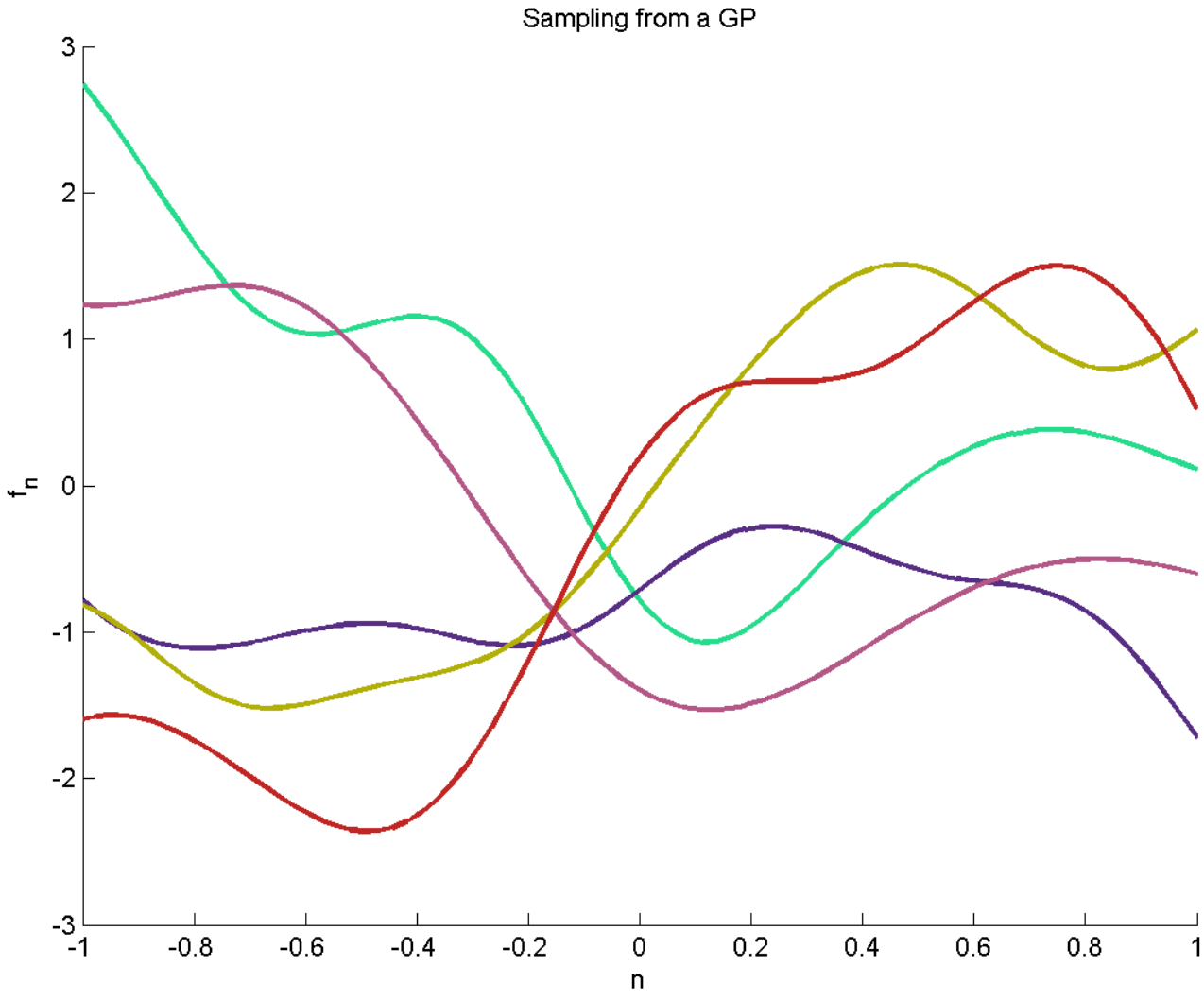

Sampling from a 2-D Gaussian
12
10
8
6
4
2
0
-2
-4
-6
-8
-6
-4
-2
0
2
4
6
8
10

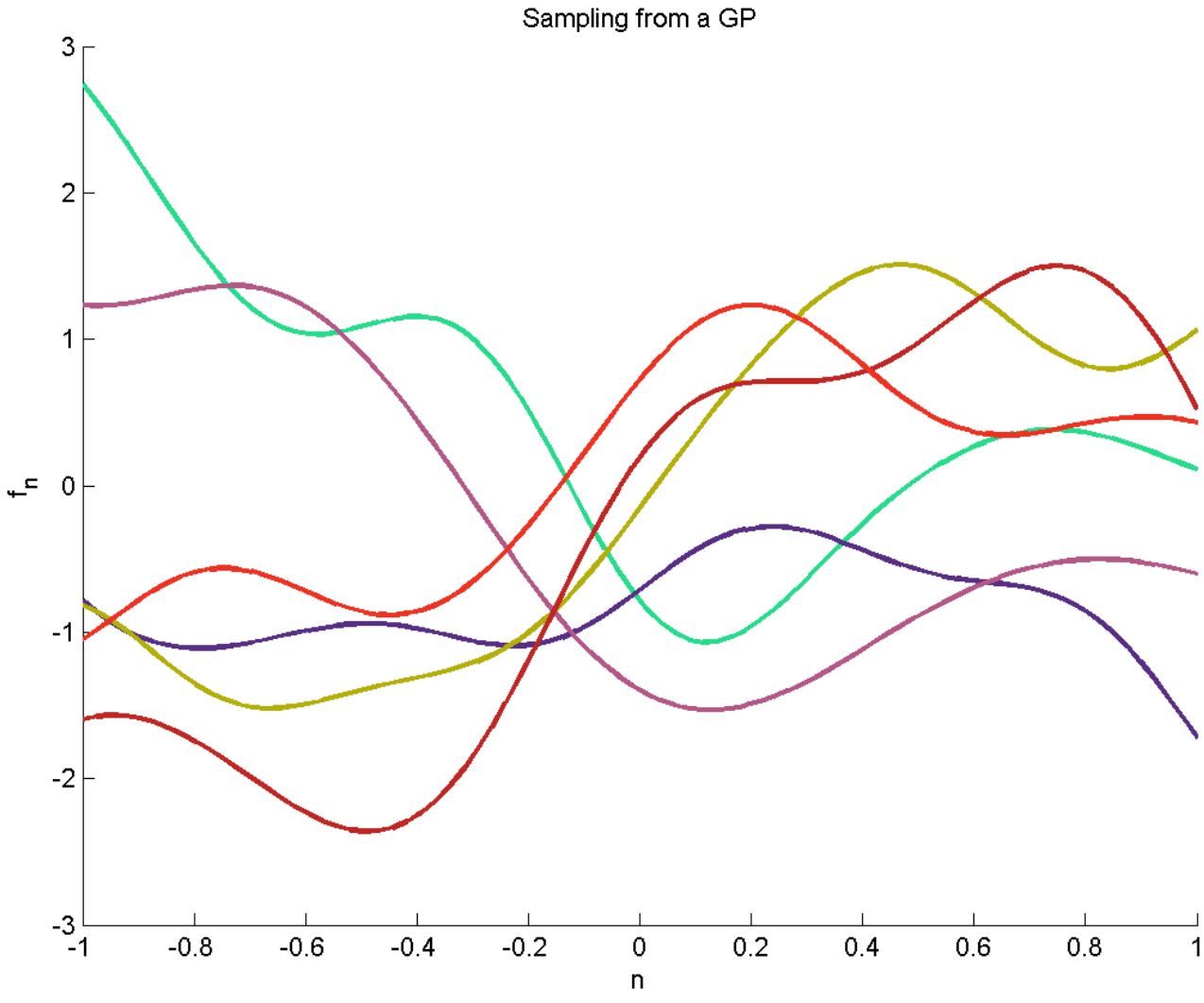

Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n

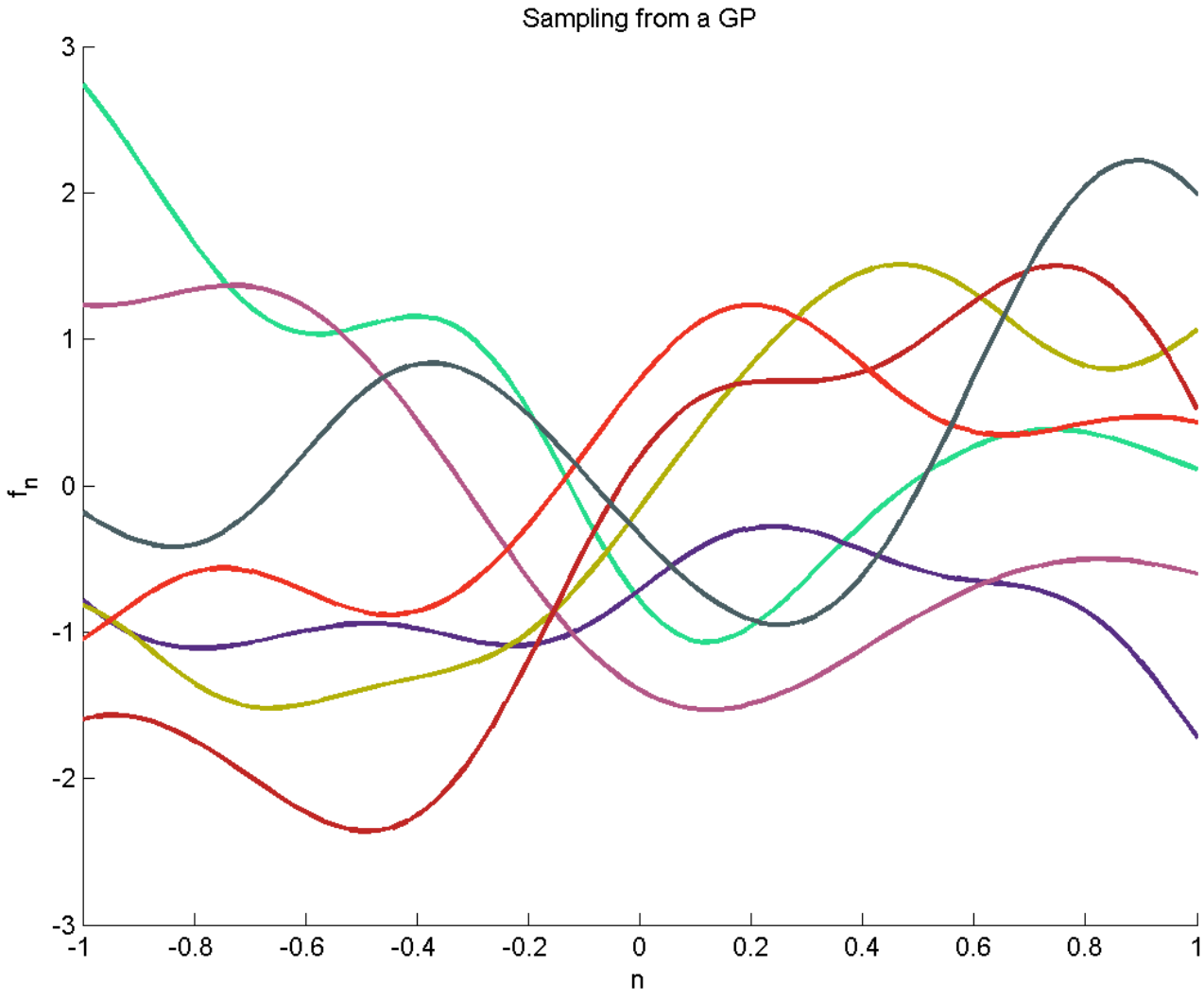

Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n

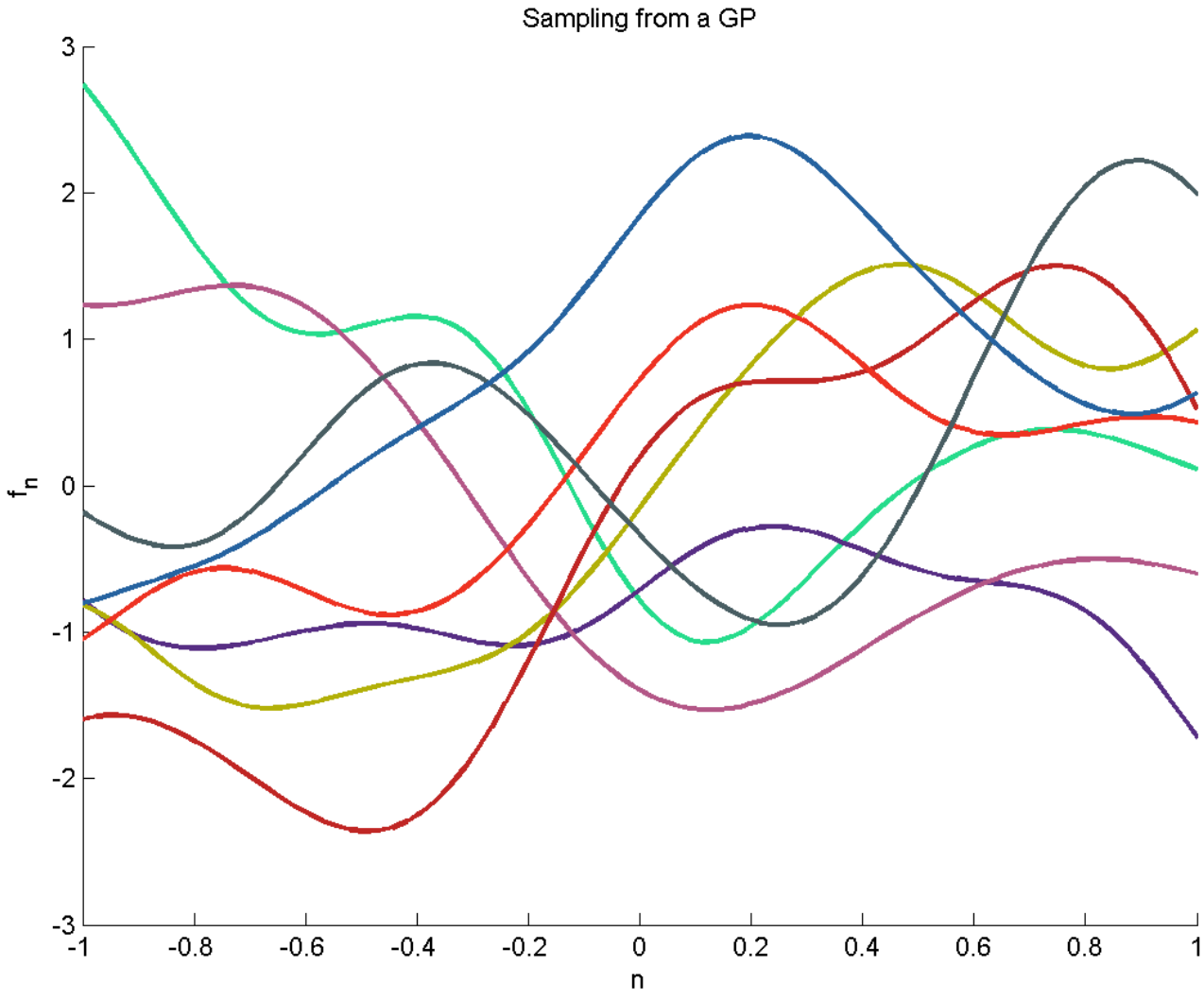

Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n

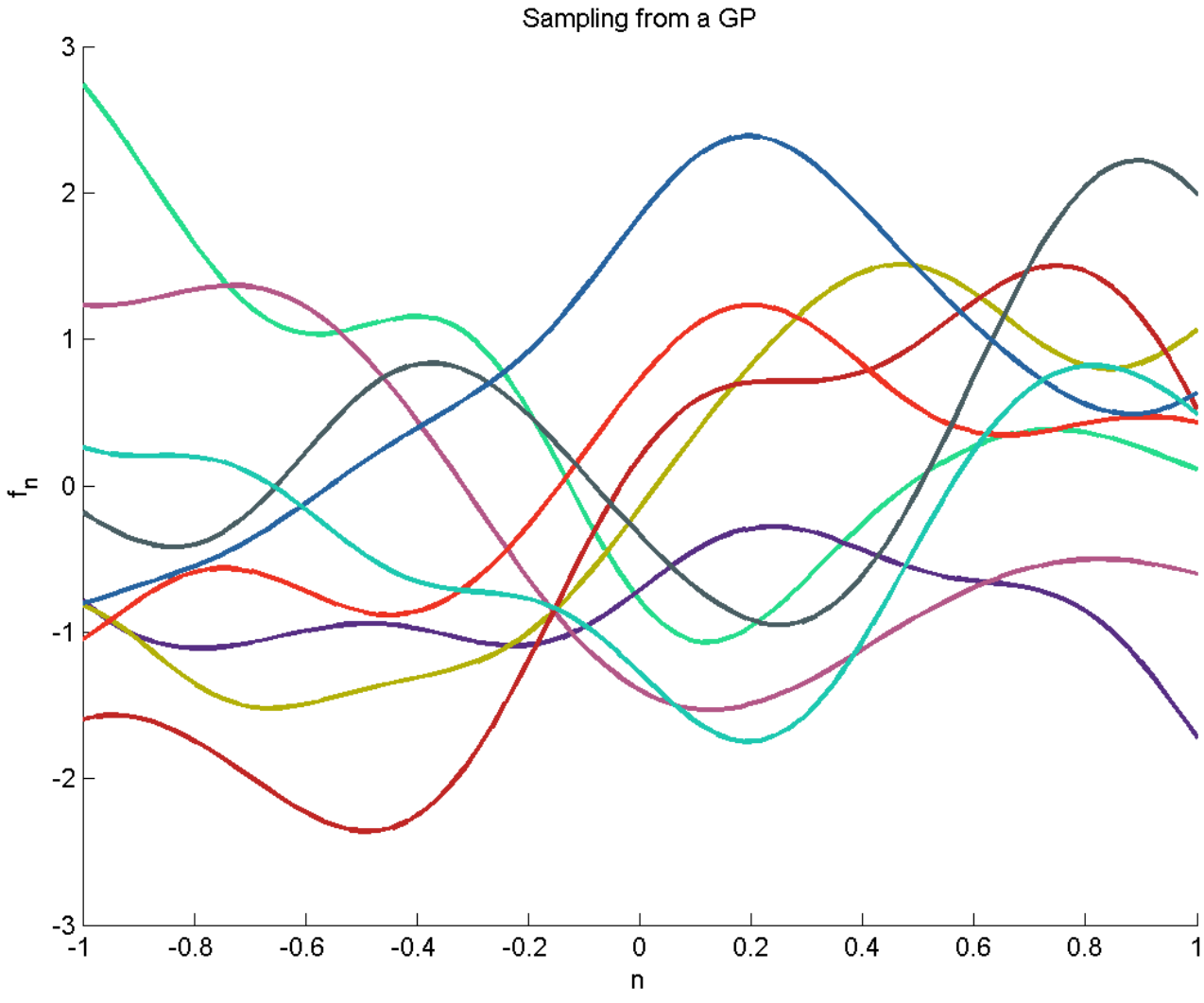

Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n

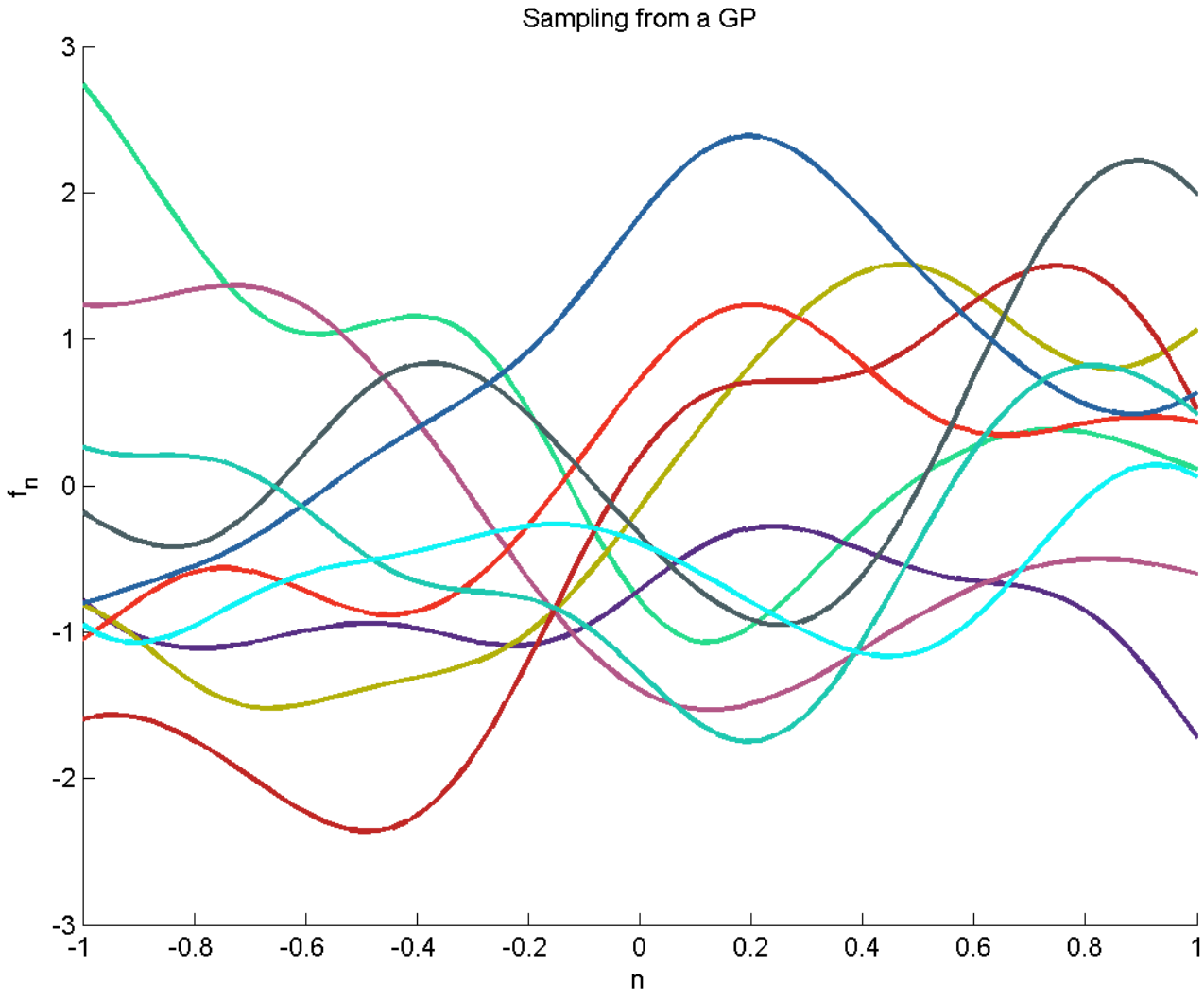

Sampling from a GP
3
2
1
0
-1
-2
-3
f_n
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
n


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


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

GO TO NOTEBOOK (start)

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:

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

Marginalisation property:

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

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:

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

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 $\mathbf{f} = f(x)$:

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

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:

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

$\mathbf{K}_*$ 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:

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

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

Noise model

- So far we assumed: $\mathbf{f} = f(\mathbf{X})$
- Assuming that we only observe noisy versions $\mathbf{y}$ of the true outputs $\mathbf{f}$:

$$\mathbf{y} = f(\mathbf{X}) + \epsilon, \quad \text{where:}$$

$$f \sim \mathcal{GP}(0, k(x, x'))$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

The above construction, gives us the following probabilities:

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2 \mathbf{I})$$

$$p(\mathbf{f}|\mathbf{x}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, K_{ff}) = (2\pi)^{n/2} |K_{ff}|^{-1/2} \exp\left(-\frac{1}{2}\mathbf{f}^T K_{ff} \mathbf{f}\right)$$

$$p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{x}) d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{0}, K_{ff} + \sigma^2 \mathbf{I})$$

$p(\mathbf{y}|\mathbf{x})$ is called the **marginal likelihood** and is tractable because of our choice for noise ϵ which is normally distributed.

Noise model

- So far we assumed: $\mathbf{f} = f(\mathbf{X})$
- Assuming that we only observe noisy versions $\mathbf{y}$ of the true outputs $\mathbf{f}$:

$$\mathbf{y} = f(\mathbf{X}) + \epsilon, \quad \text{where:}$$

$$f \sim \mathcal{GP}(0, k(x, x'))$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

The above construction, gives us the following probabilities:

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2 \mathbf{I})$$

$$p(\mathbf{f}|\mathbf{x}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, K_{ff}) = (2\pi)^{n/2} |K_{ff}|^{-1/2} \exp\left(-\frac{1}{2}\mathbf{f}^T K_{ff} \mathbf{f}\right)$$

$$p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{x}) d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{0}, K_{ff} + \sigma^2 \mathbf{I})$$

$p(\mathbf{y}|\mathbf{x})$ is called the **marginal likelihood** and is tractable because of our choice for noise ϵ which is normally distributed.

Noise model

- So far we assumed: $\mathbf{f} = f(\mathbf{X})$
- Assuming that we only observe noisy versions $\mathbf{y}$ of the true outputs $\mathbf{f}$:

$$\mathbf{y} = f(\mathbf{X}) + \epsilon, \quad \text{where:}$$

$$f \sim \mathcal{GP}(0, k(x, x'))$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

The above construction, gives us the following probabilities:

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \sigma^2 \mathbf{I})$$

$$p(\mathbf{f}|\mathbf{x}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, K_{ff}) = (2\pi)^{n/2} |K_{ff}|^{-1/2} \exp\left(-\frac{1}{2}\mathbf{f}^T K_{ff} \mathbf{f}\right)$$

$$p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{x}) d\mathbf{f} = \mathcal{N}(\mathbf{y}|\mathbf{0}, K_{ff} + \sigma^2 \mathbf{I})$$

$p(\mathbf{y}|\mathbf{x})$ is called the **marginal likelihood** and is tractable because of our choice for noise ϵ which is normally distributed.

Predictions in the noise model

$$\mathbf{y}^*|\mathbf{y}, \mathbf{x}, \mathbf{x}_* \sim \mathcal{N}(\boldsymbol{\mu}_{\text{pred}}, \mathbf{K}_{\text{pred}})$$

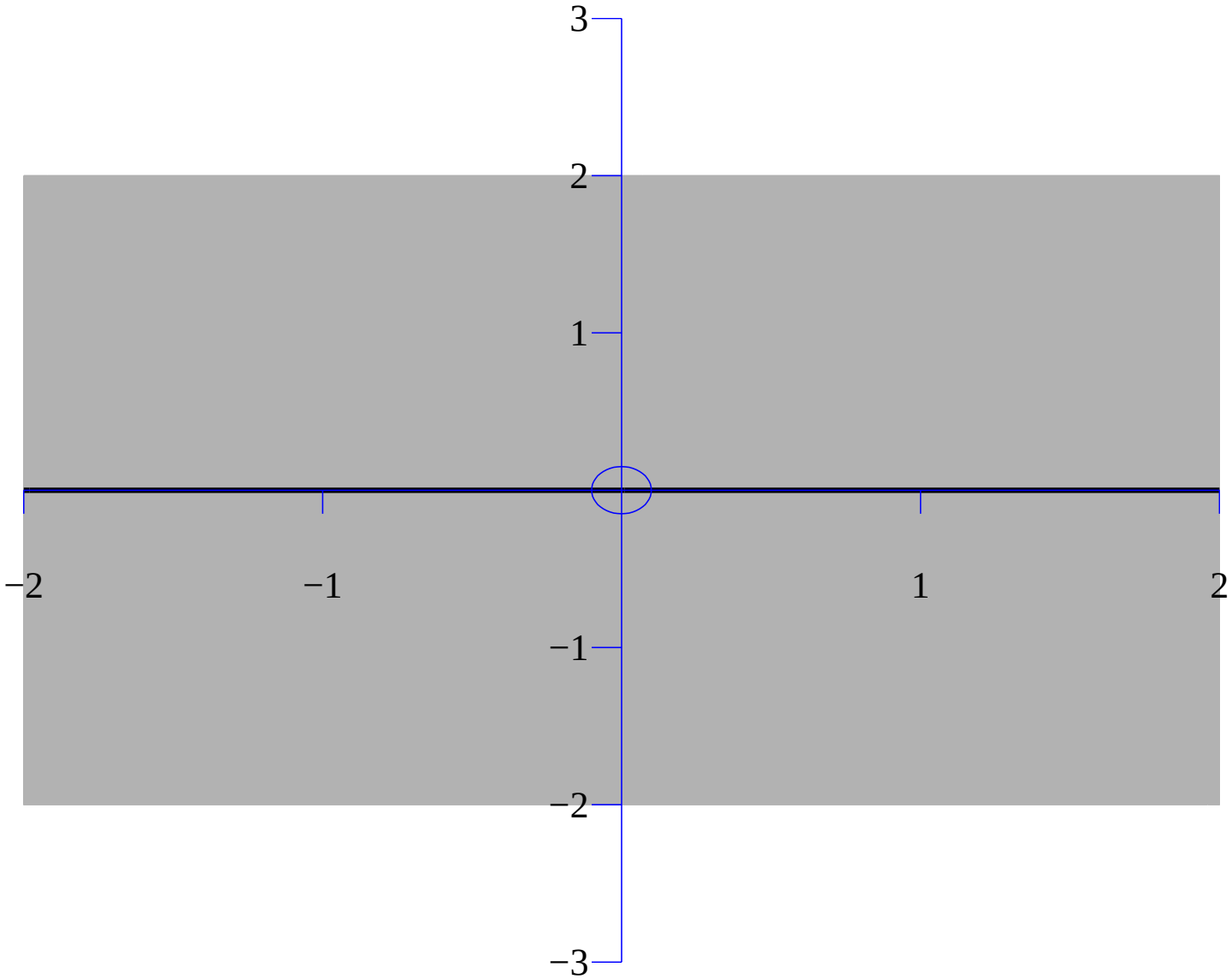
with

$$\boldsymbol{\mu}_{\text{pred}} = \mathbf{K}_*^\top \left[\mathbf{K} + \sigma^2 \mathbf{I}\right]^{-1} \mathbf{y}$$

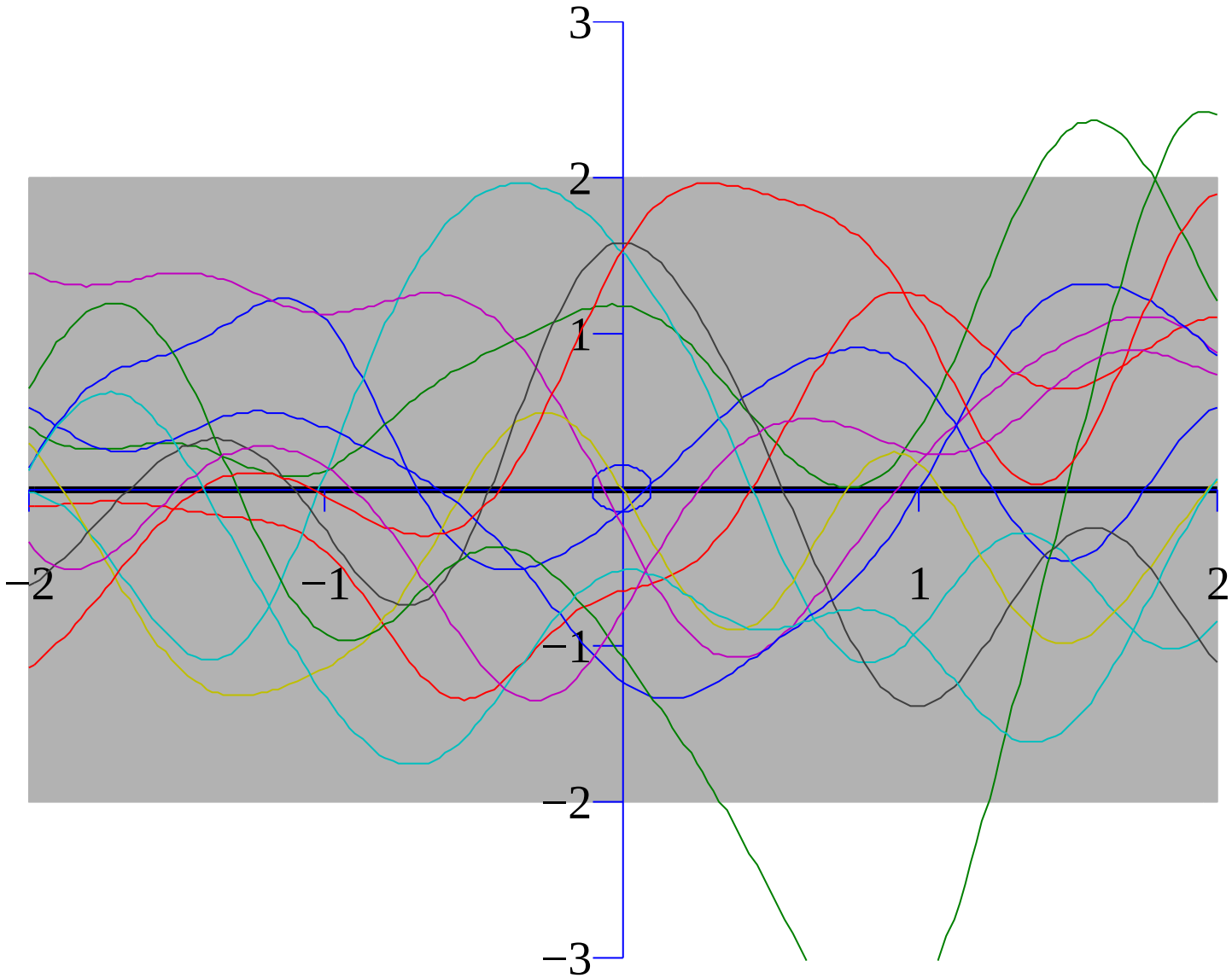
and

$$\mathbf{K}_{\text{pred}} = \mathbf{K}_{*,*} - \mathbf{K}_*^\top \left[\mathbf{K} + \sigma^2 \mathbf{I}\right]^{-1} \mathbf{K}_*.$$

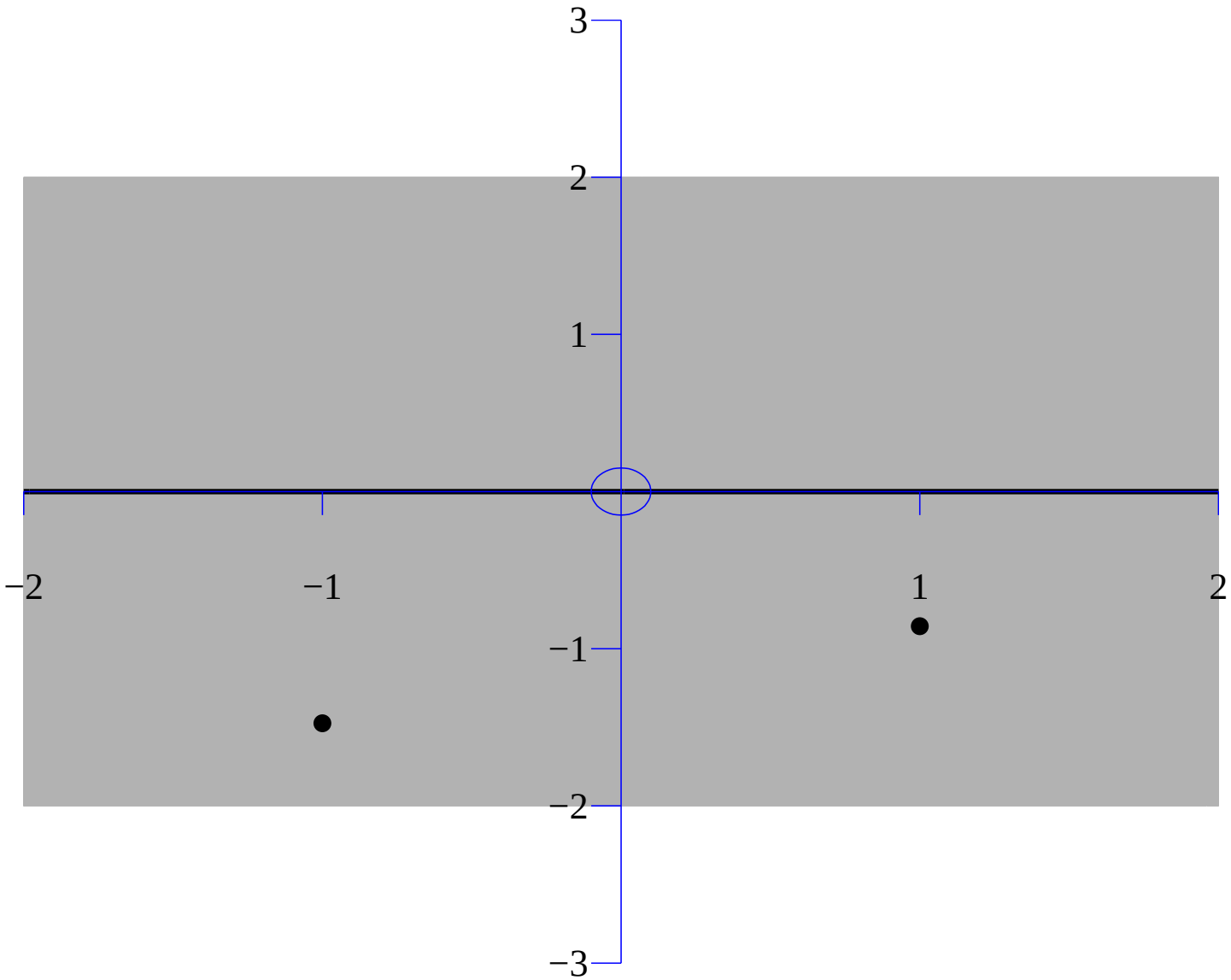
Fitting the data *(shaded area is* **uncertainty***)*



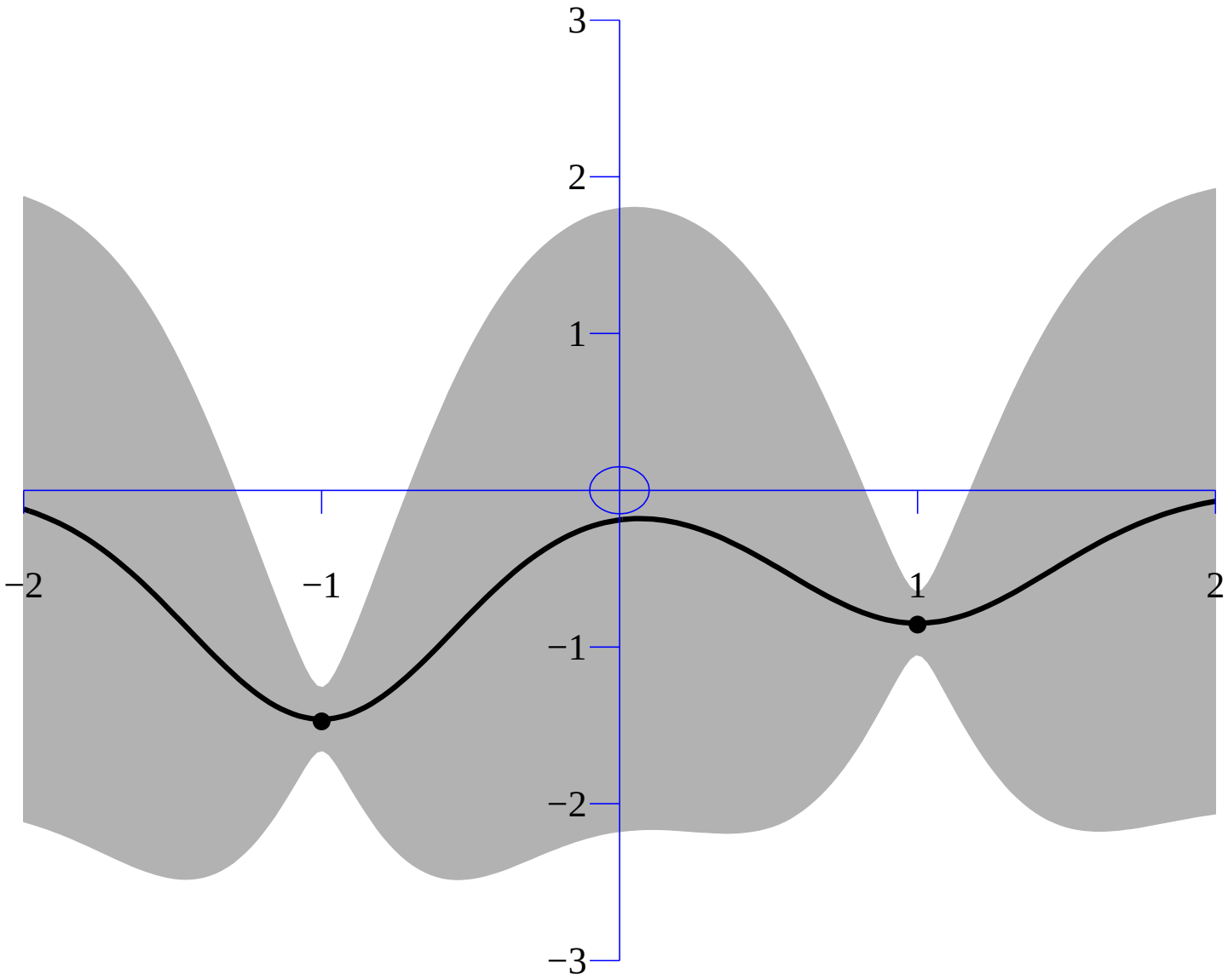
Fitting the data - Prior Samples



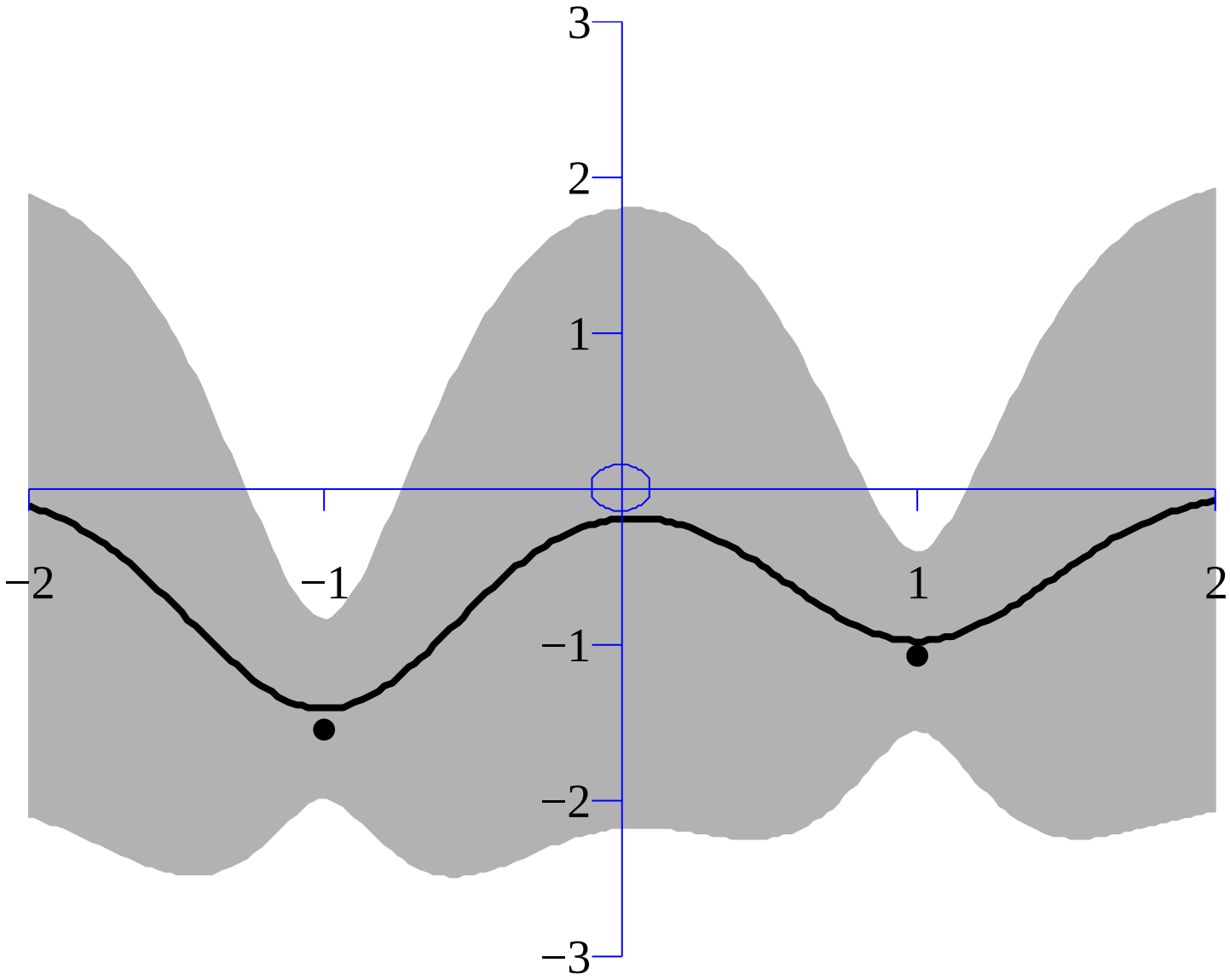
Fitting the data



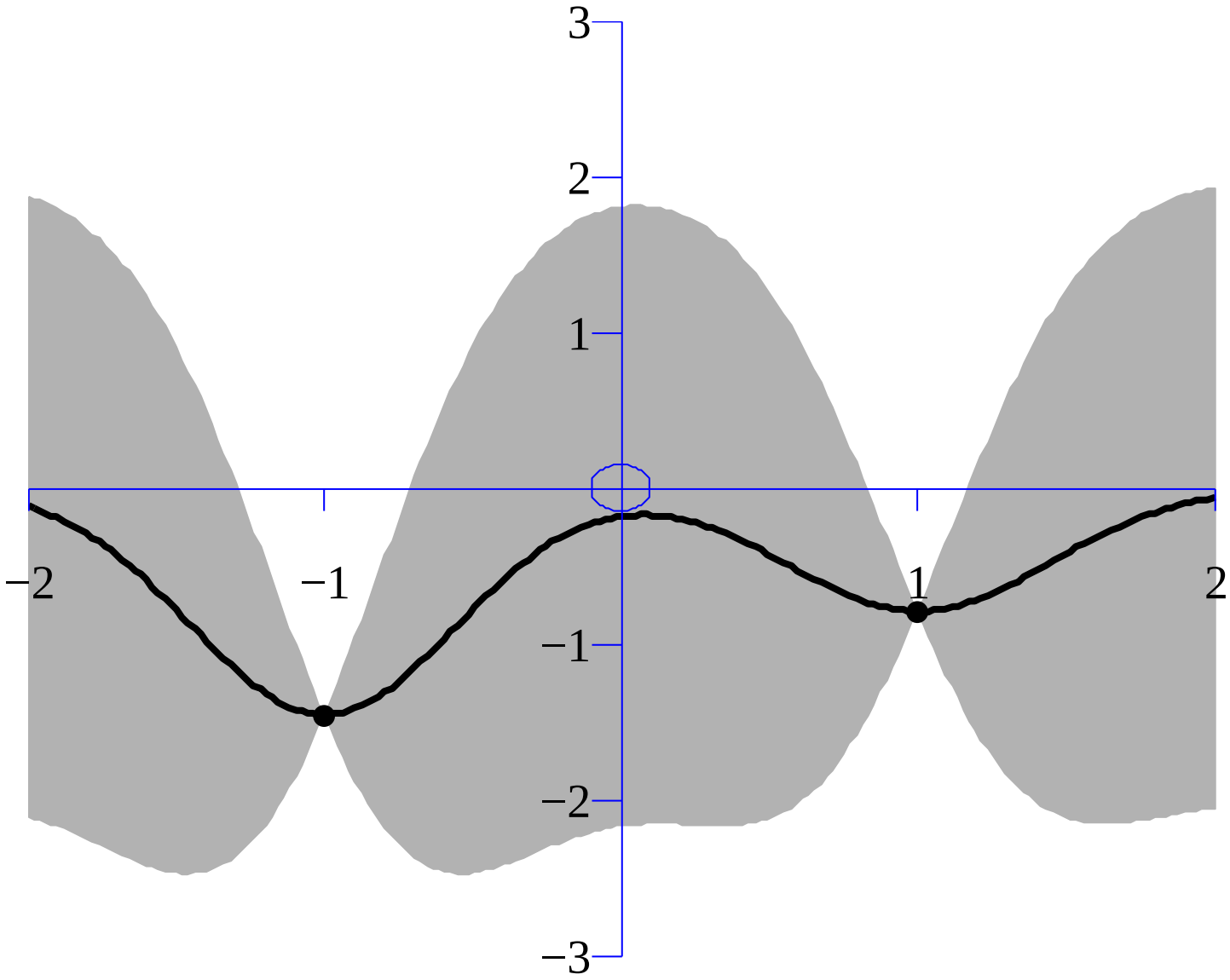
Fitting the data



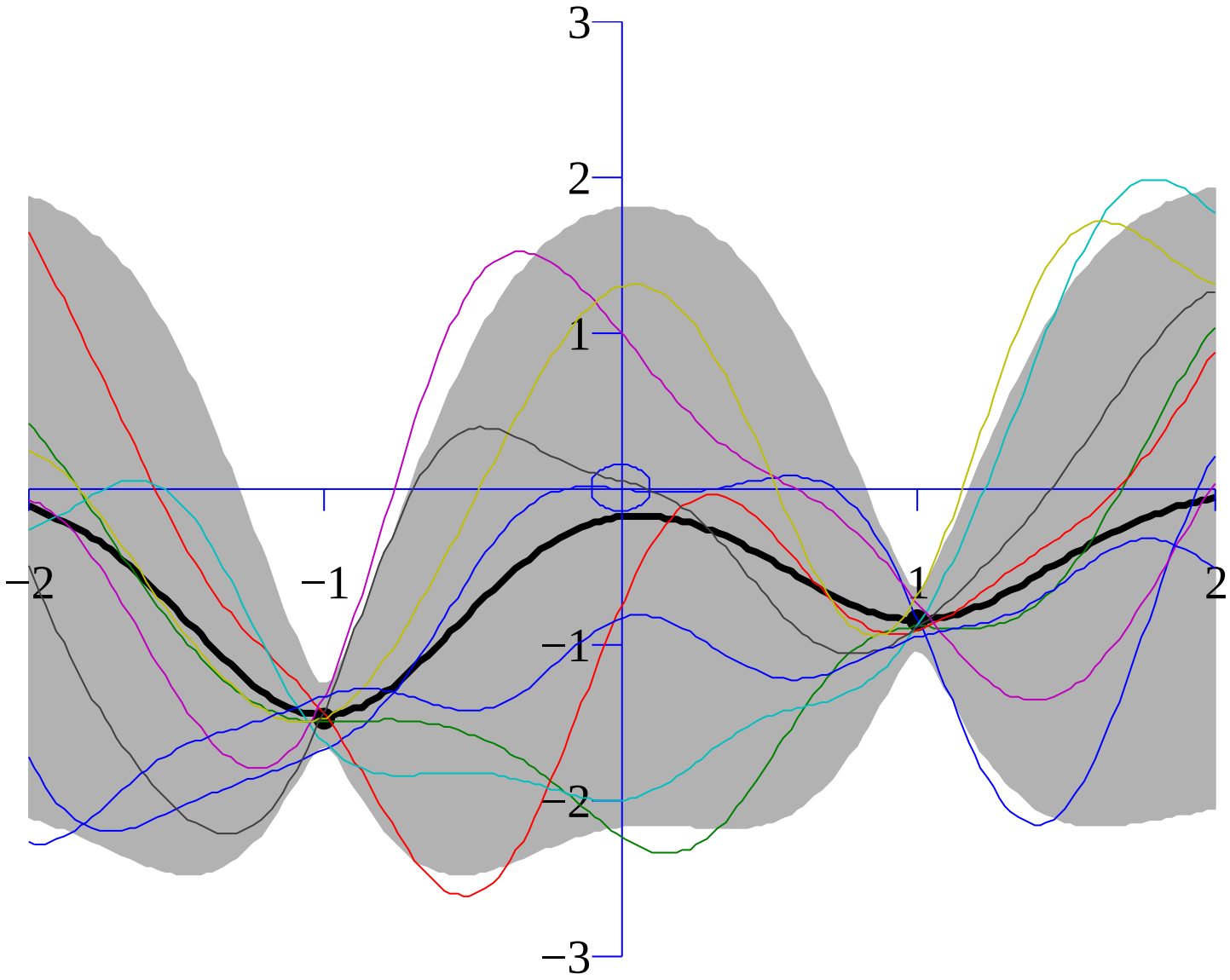
Fitting the data - more noise



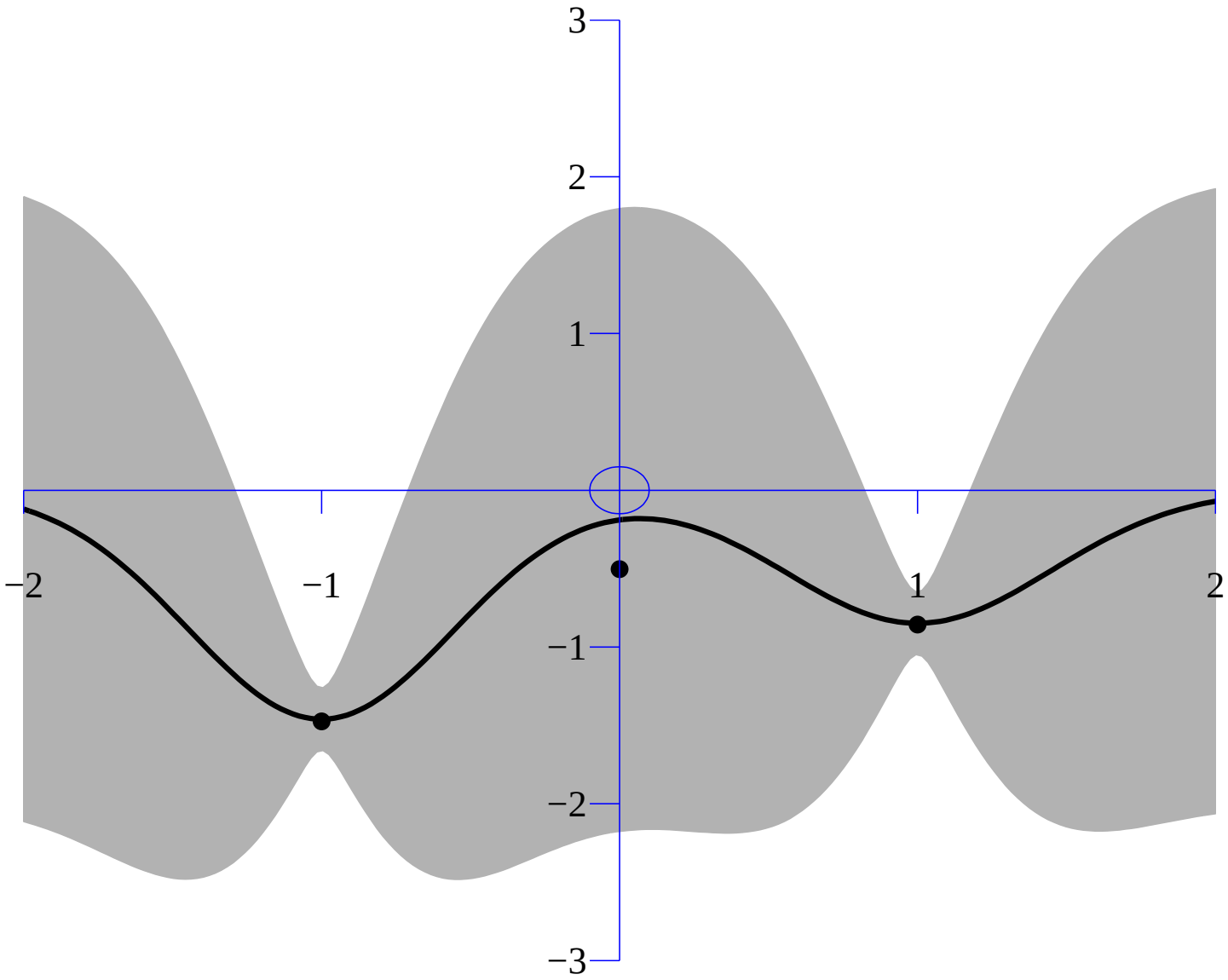
Fitting the data - no noise



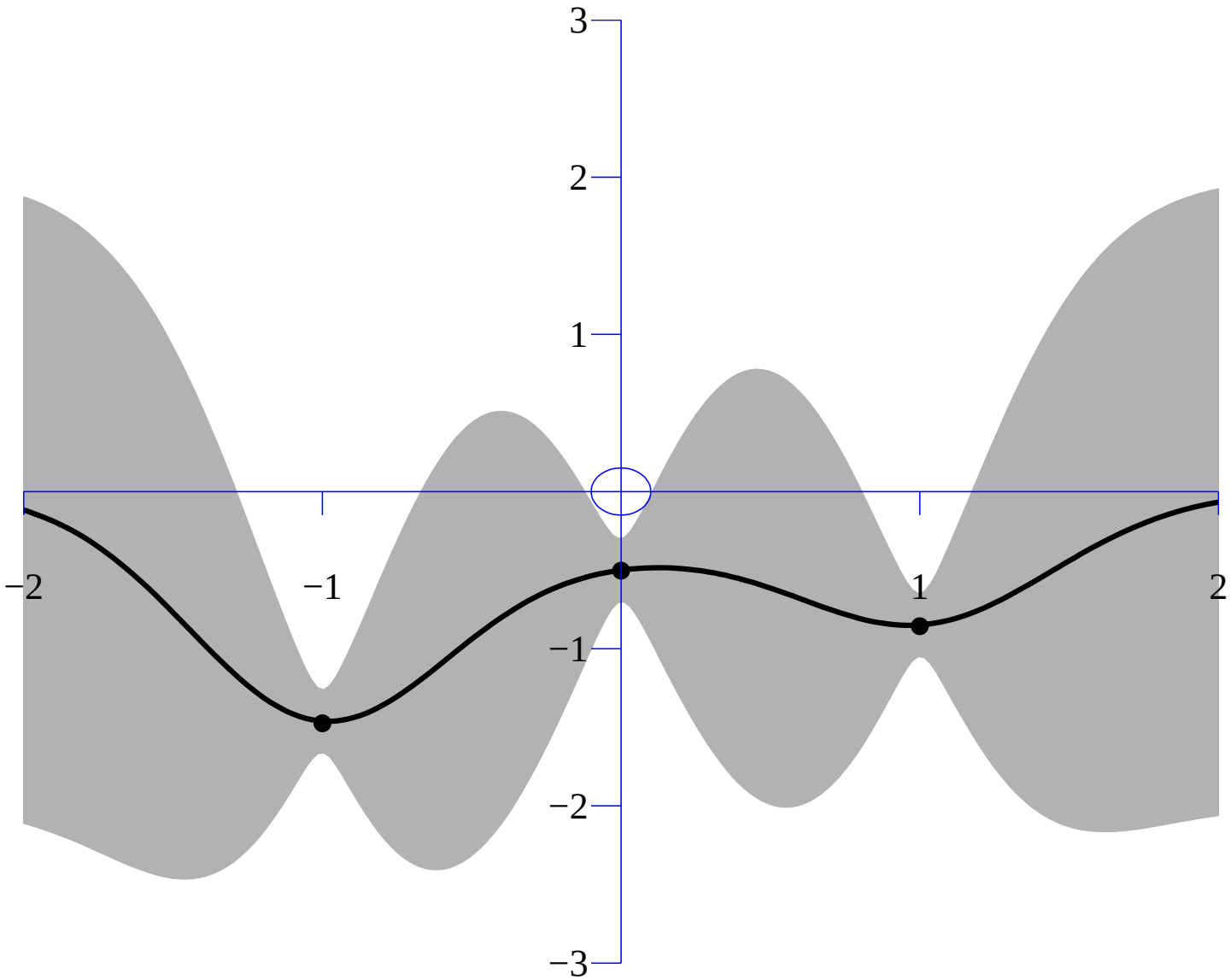
Fitting the data - Posterior samples



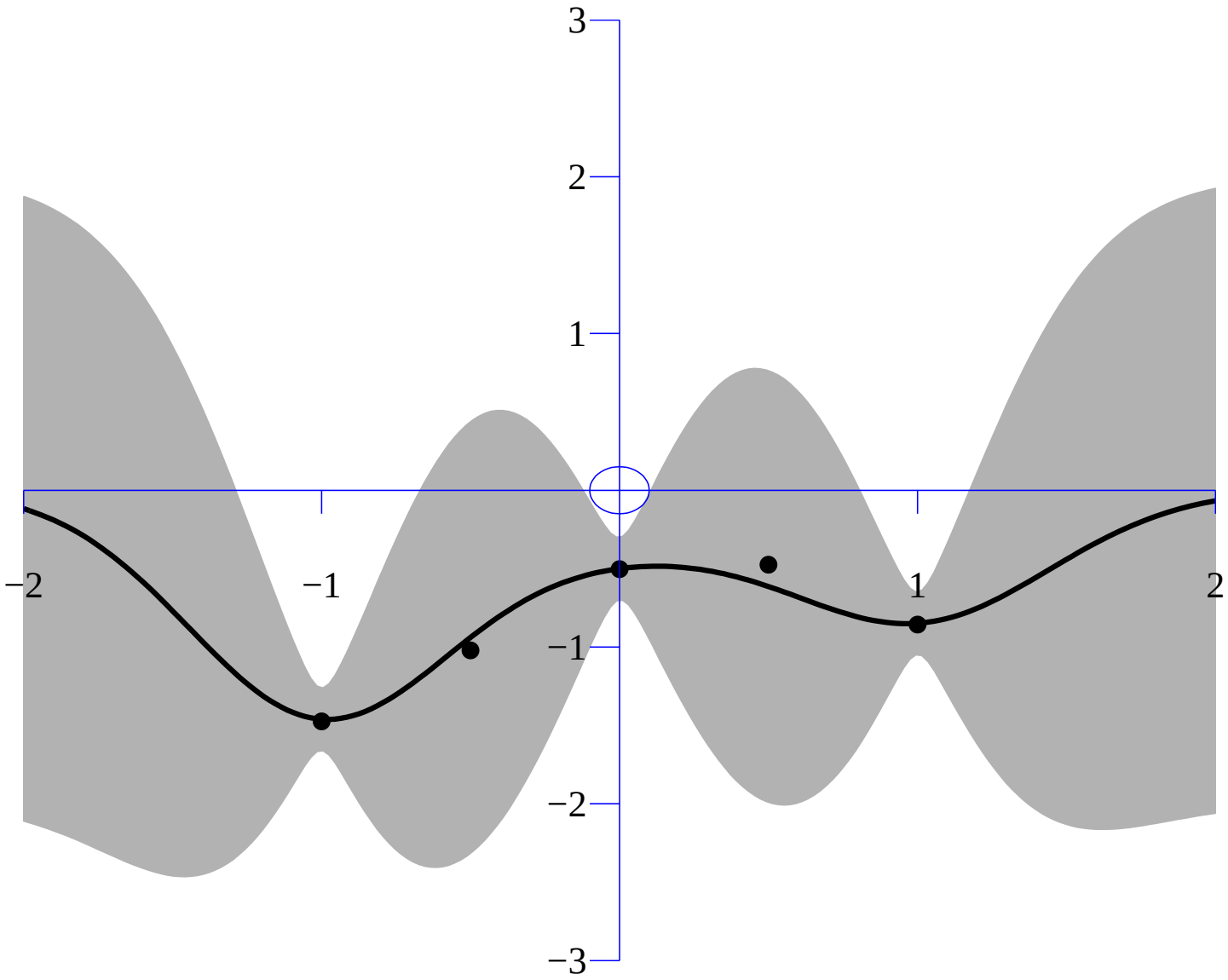
Fitting the data



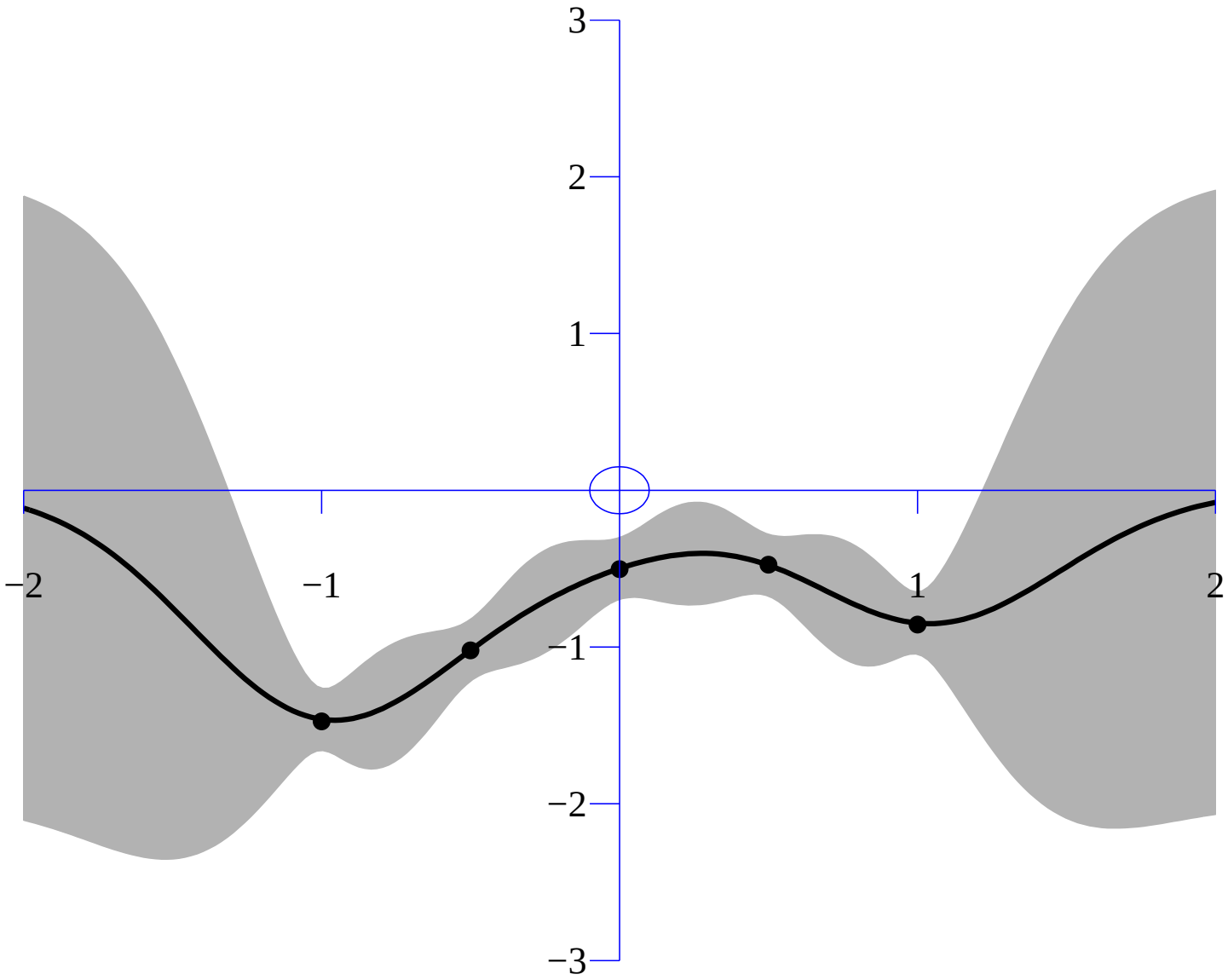
Fitting the data



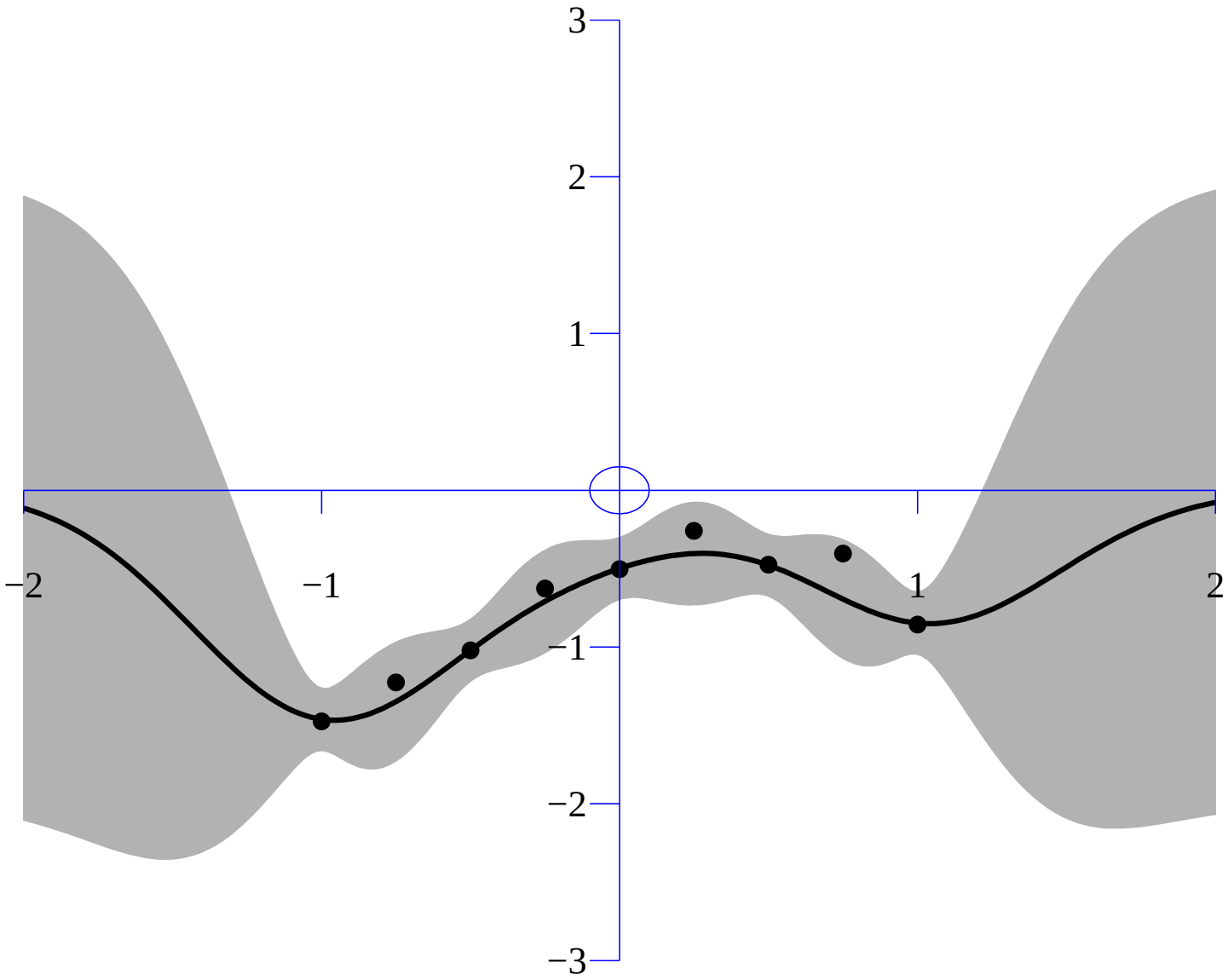
Fitting the data



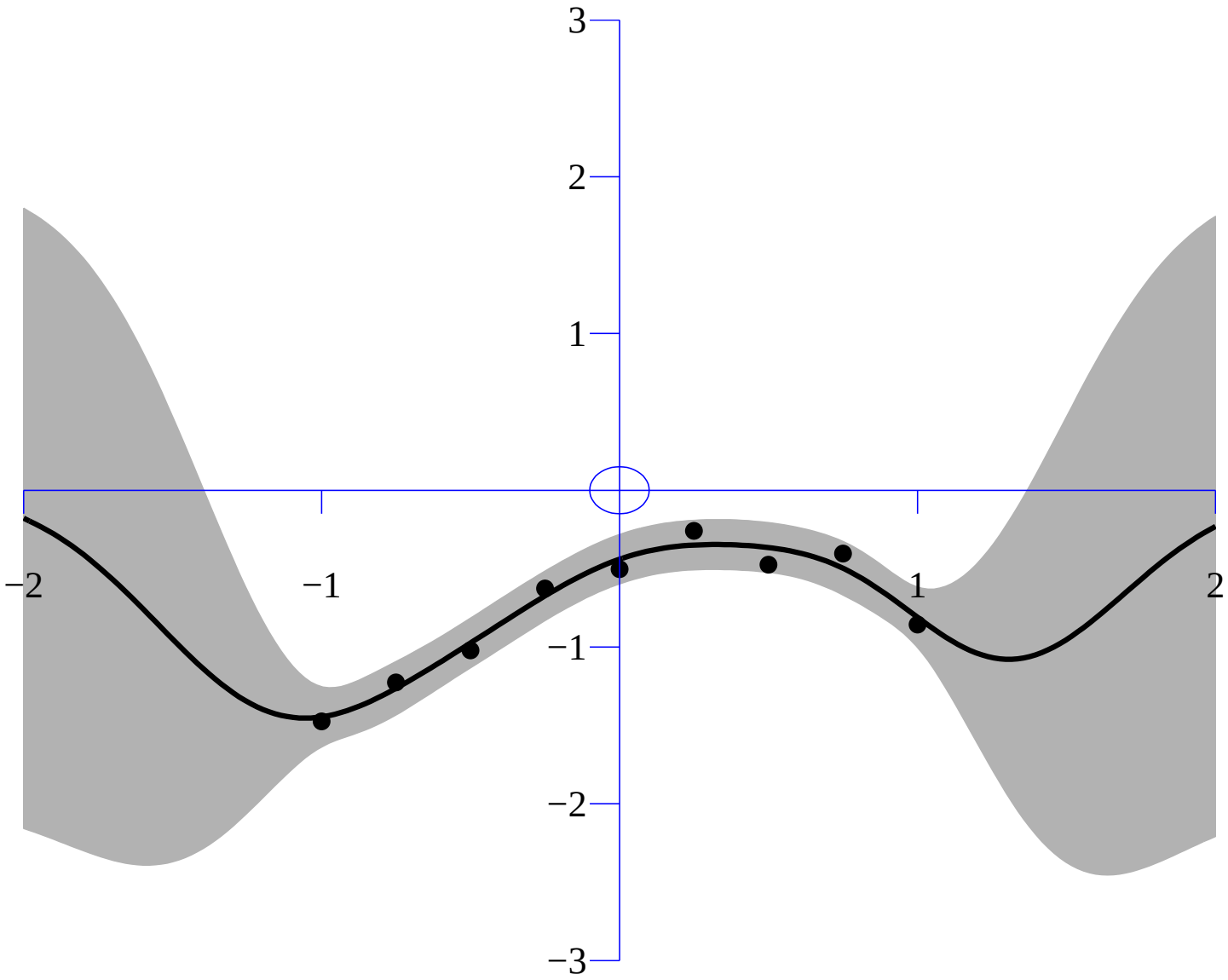
Fitting the data



Fitting the data

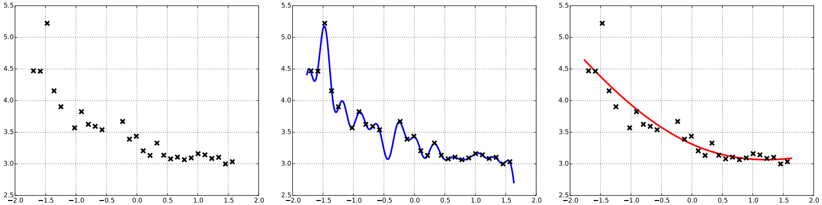


Fitting the data



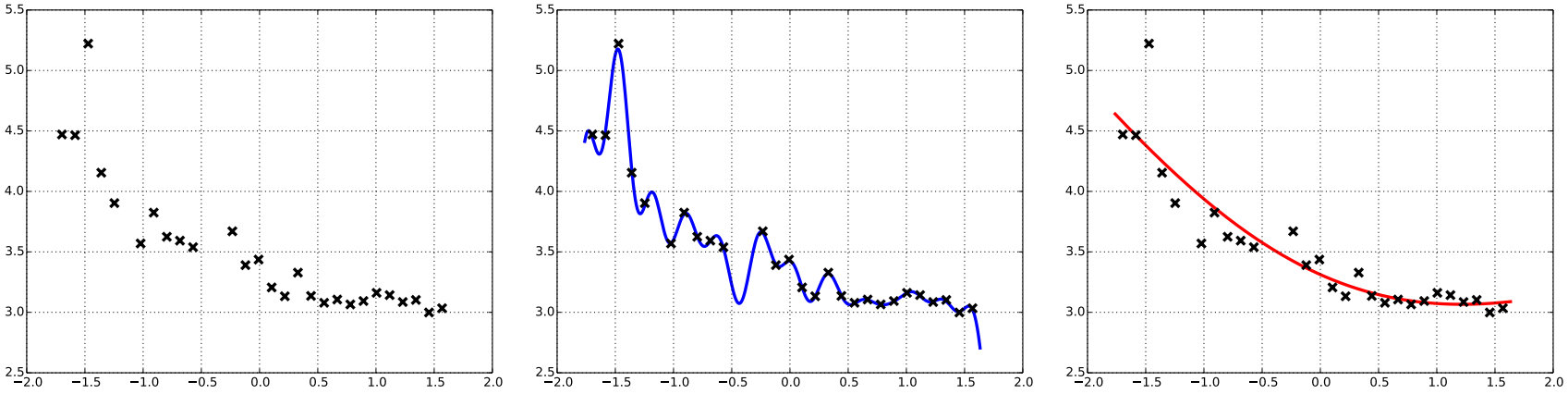
GO TO NOTEBOOK (Cov. functions)

Curve fitting



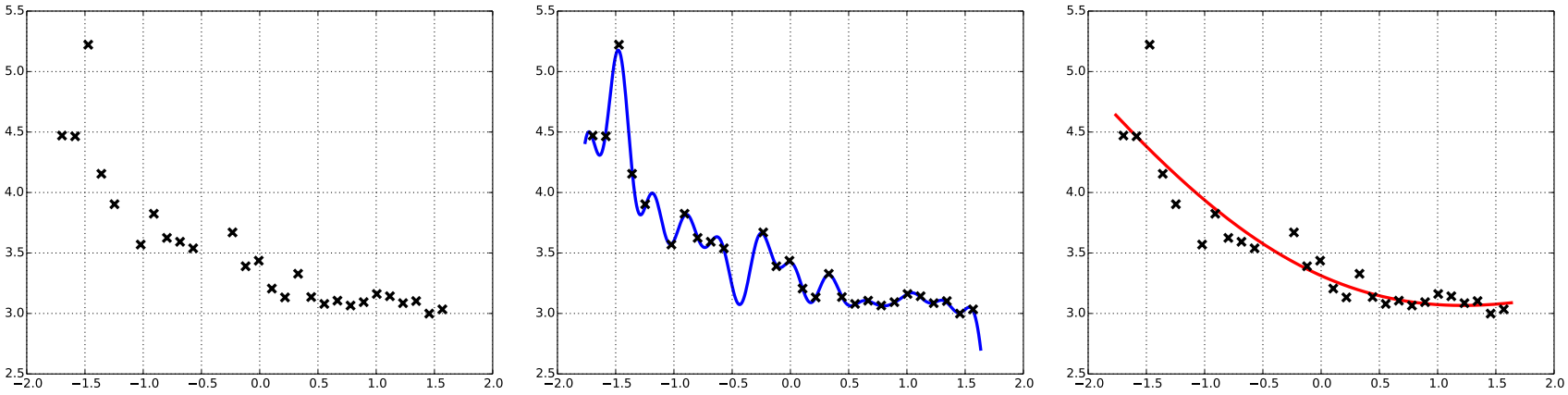
- Which curve fits the data better?

Curve fitting



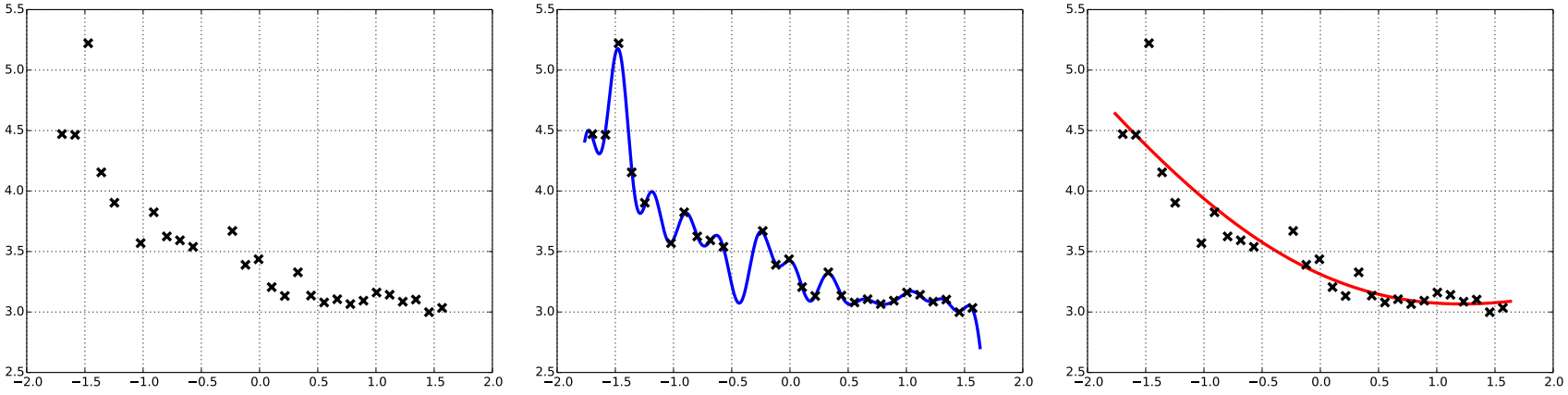
- Which curve fits the data better?
- Which curve is more "complex"?

Curve fitting



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

Curve fitting



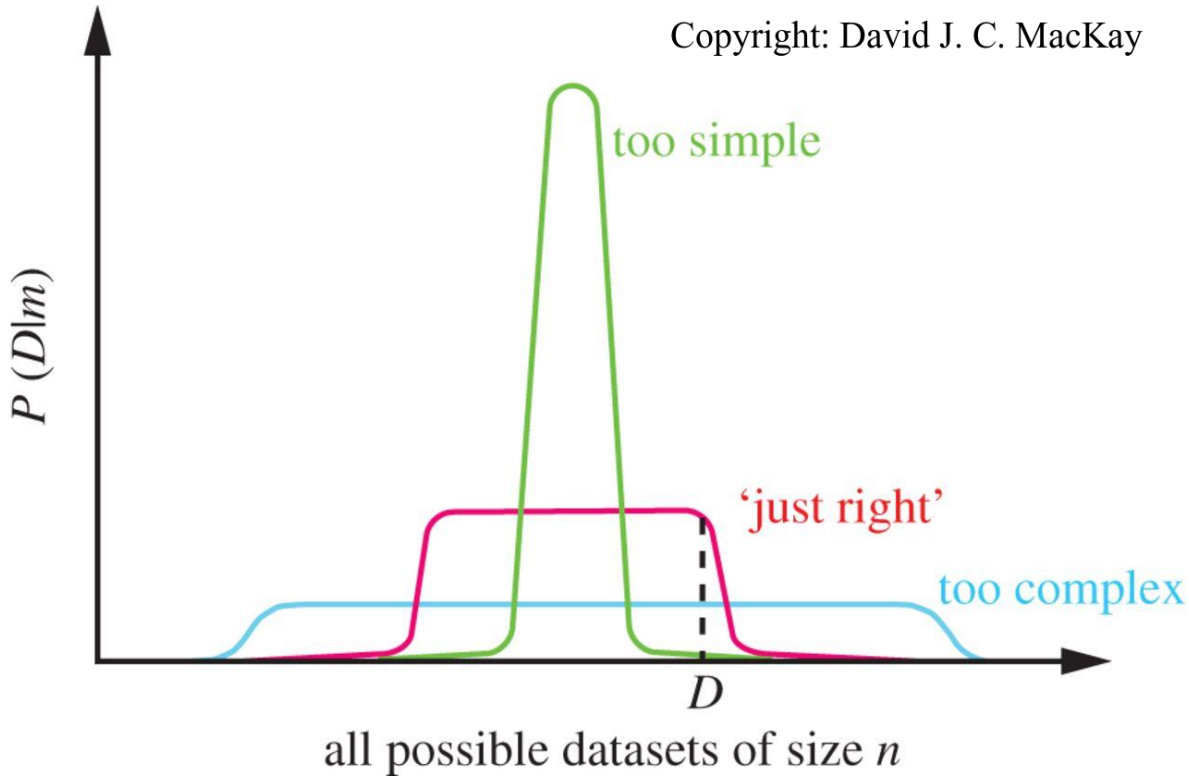
- Which curve fits the data better?
- Which curve is more "complex"?
- Which curve is better overall?

Need a good balance between data fit vs overfitting!

(Bayesian) Occam's Razor

"A plurality is not to be posited without necessity". *W. of Ockham*

"Everything should be made as simple as possible, but not simpler". *A. Einstein*



Evidence is higher for the model that is not "unnecessarily complex" but still "explains" the data D.

How do GPs solve the overfitting problem (i.e. regularize)?

- Answer: Integrate over the function itself!
- This is associated with the Bayesian methodology.
- So, we will average out all possible function forms, under a (GP) prior!

Recap:

$$\text{ML:} \quad \underset{\mathbf{w}}{\operatorname{argmax}}\ p(\mathbf{y}|\mathbf{w}, \phi(\mathbf{x})) \qquad \text{e.g. } \mathbf{y} = \phi(\mathbf{x})^\top \mathbf{w} + \boldsymbol{\epsilon}$$

$$\text{Bayesian:} \quad \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) \underbrace{p(\mathbf{f}|\mathbf{x}, \boldsymbol{\theta})}_{\text{GP prior}} \qquad \text{e.g. } \mathbf{y} = f(\mathbf{x}, \boldsymbol{\theta}) + \boldsymbol{\epsilon}$$

- $\boldsymbol{\theta}$ are *hyper*parameters
- The Bayesian approach (GP) automatically balances the data-fitting with the complexity penalty.

How do GPs solve the overfitting problem (i.e. regularize)?

- Answer: Integrate over the function itself!
- This is associated with the Bayesian methodology.
- So, we will average out all possible function forms, under a (GP) prior!

Recap:

$$\text{ML:} \quad \underset{\mathbf{w}}{\operatorname{argmax}}\ p(\mathbf{y}|\mathbf{w}, \phi(\mathbf{x})) \qquad \text{e.g. } \mathbf{y} = \phi(\mathbf{x})^\top \mathbf{w} + \boldsymbol{\epsilon}$$

$$\text{Bayesian:} \quad \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f}) \underbrace{p(\mathbf{f}|\mathbf{x}, \boldsymbol{\theta})}_{\text{GP prior}} \qquad \text{e.g. } \mathbf{y} = f(\mathbf{x}, \boldsymbol{\theta}) + \boldsymbol{\epsilon}$$

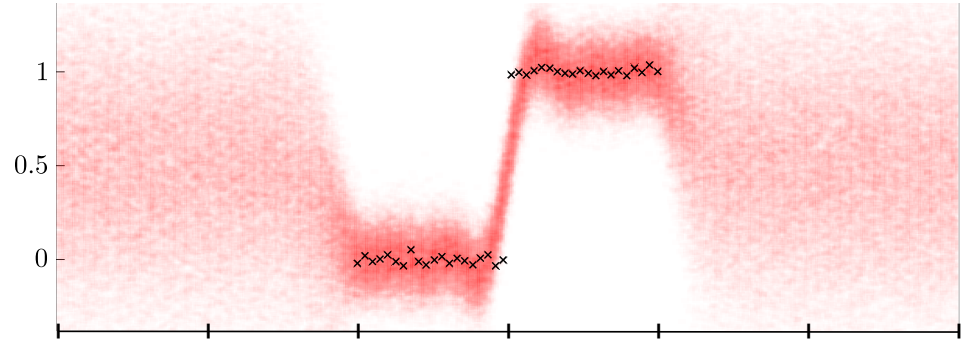
- $\boldsymbol{\theta}$ are *hyper*parameters
- The Bayesian approach (GP) automatically balances the data-fitting with the complexity penalty.

GO TO NOTEBOOK

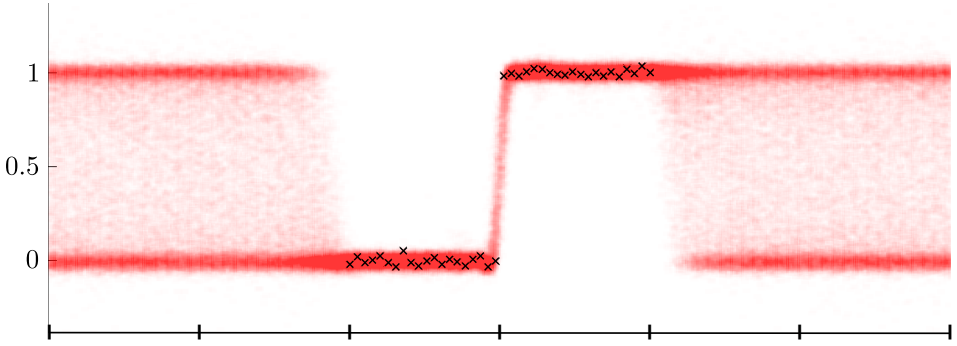
Teaser for tomorrow

- Unsupervised learning with GPs (Bayesian non-linear dimensionality reduction)
- Deep GPs
- Dynamical systems
- ..and more...

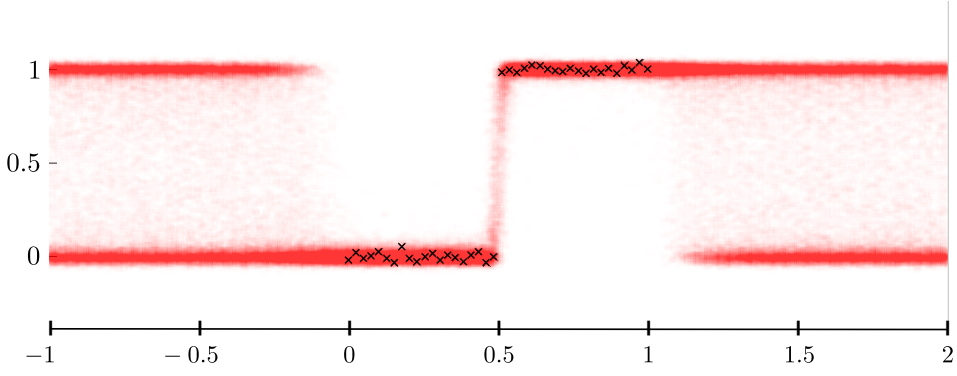
Deep GP: Step function (credits for idea to J. Hensman)



(standard GP)



(deep GP - 1 hidden layer)



(deep GP - 3 hidden layers)

Thanks

Thanks to Prof. Neil Lawrence, his SheffieldML group and the GPy developers.

Thanks to George Karniadakis, Paris Perdikaris for hosting me

EOF