DD2434 - Advanced Machine Learning

Gaussian Processes

Carl Henrik Ek
{chek}@csc.kth.se

Royal Institute of Technology

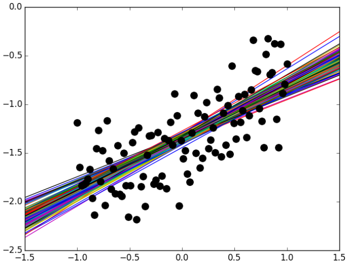
November 5th, 2015

Last Lecture

- General Probabilistic Modelling
  - Probabilistic objects
  - Marginalisation
- Kernels
  - Dual linear regression
  - Implications for modelling

Regression

- Two variates
  - Input data $\mathbf{x}_i \in \mathbb{R}^q$
  - Output data $\mathbf{y}_i \in \mathbb{R}^D$
- Relationship: $f : \mathbf{X} \rightarrow \mathbf{Y}$



Regression

Uncertainty

- We are uncertain in our data
- This means we cannot trust
  - our observations
  - the mapping that we learn
  - the predictions that we make under the mapping

Regression

Uncertainty

- Uncertainty in outputs $\mathbf{y}_i$
  - Addative noise $\mathbf{y}_i = \mathbf{W}\mathbf{x}_i + \epsilon$
  - Gaussian distributed noise $\epsilon \propto \mathcal{N}(0, \sigma^2)$
- Likelihood

Regression

Uncertainty in prediction

- Posterior
  - conditional distribution
  - *after* the relevant information has been taken into account
- What is relevant
  - our belief: prior $p(\mathbf{W})$
  - the observations: likelihood $p(\mathbf{Y}|\mathbf{W}, \mathbf{X})$

Regression

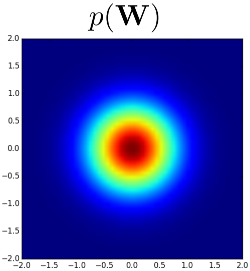
$$p(\mathbf{Y}|\mathbf{W},\mathbf{X}) = \prod_{i}^{N} p(\mathbf{y}_i|\mathbf{W},\mathbf{x}_i) \quad (1)$$

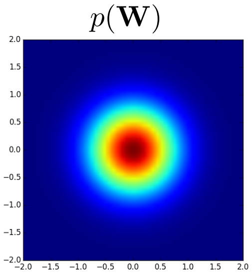
Structure

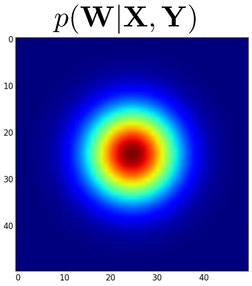
- Do the variables co-vary?
- Are there (in-)dependency structures that I can exploit?

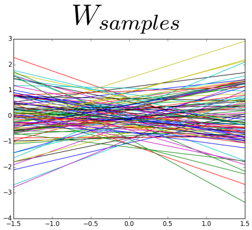
Toolbox

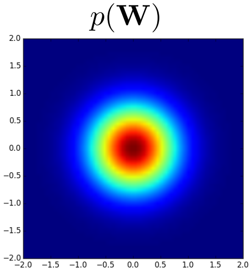
1. Formulate prediction error likelihood
   - Does the likelihood have structure?
2. Formulate belief of model in prior
   - Does the prior have structure
3. Reach the posterior by combining likelihood and prior
4. Choose model based on *evidence* $p(\mathcal{D}|\mathcal{M})$

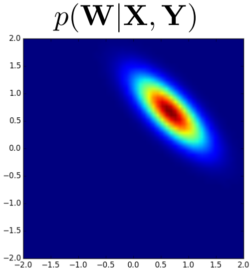

$p(\mathbf{W})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

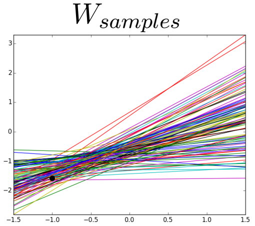

$p(\mathbf{W})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0

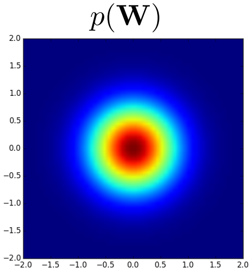

$p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$
0
10
20
30
40

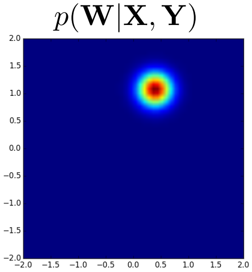

$W_{samples}$
3
2
1
0
−1
−2
−3
−4
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

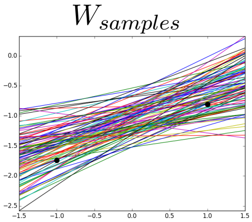

$p(\mathbf{W})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

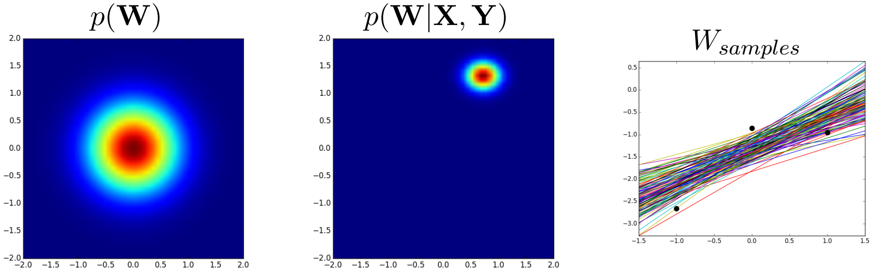

$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

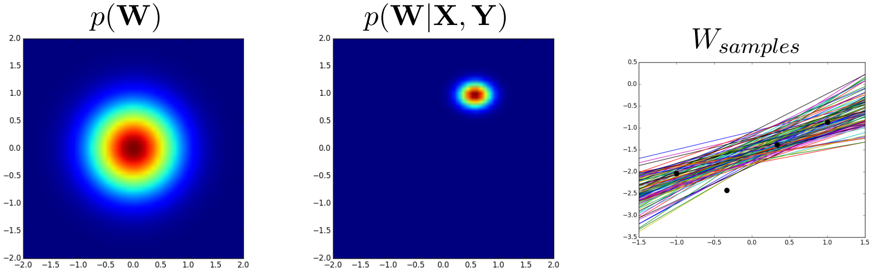

$W_{samples}$
3
2
1
0
−1
−2
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

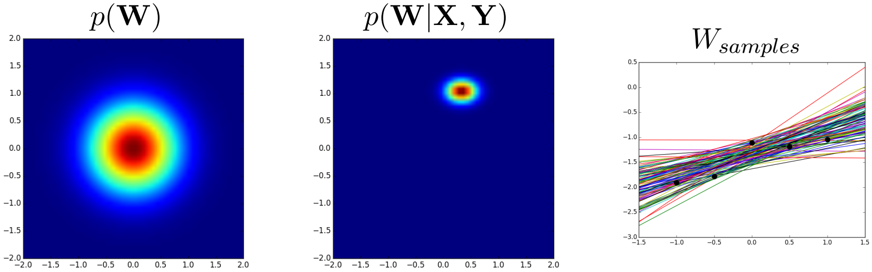

$p(\mathbf{W})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0


$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0


$W_{samples}$
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

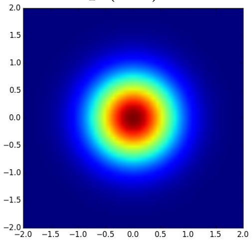

$p(\mathbf{W})$
$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$
$W_{samples}$


$p(\mathbf{W})$
$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$
$W_{samples}$

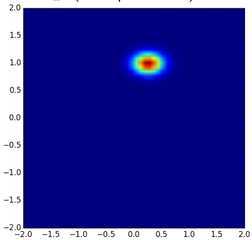


$p(\mathbf{W})$
$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$
$W_{samples}$

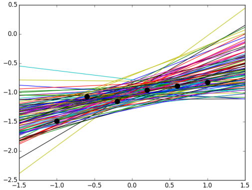
$p(\mathbf{W})$

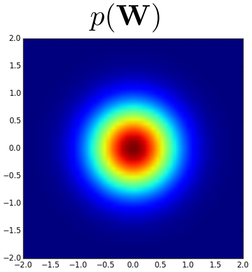


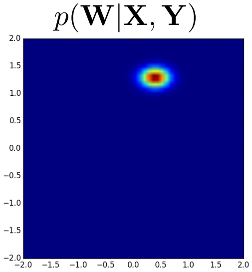
$p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$

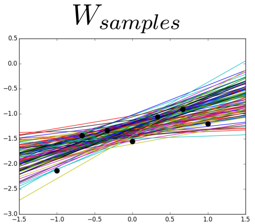


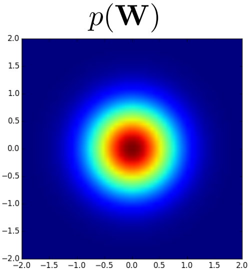
$W_{samples}$

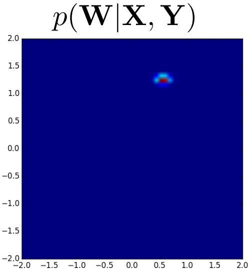


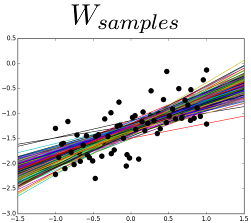

$p(\mathbf{W})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

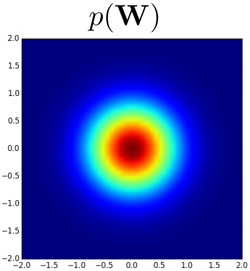

$p(\mathbf{W}|\mathbf{X}, \mathbf{Y})$
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.0 −1.5 −1.0 −0.5 0.0 0.5 1.0 1.5 2.0

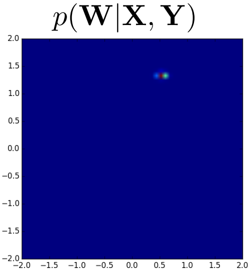

$W_{samples}$
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
−1.5 −1.0 −0.5 0.0 0.5 1.0 1.5

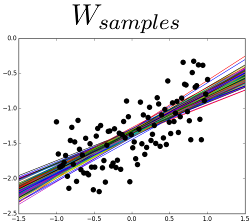

$p(\mathbf{W})$


$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$


$W_{samples}$


$p(\mathbf{W})$


$p(\mathbf{W}|\mathbf{X},\mathbf{Y})$


$W_{samples}$

Conditional[1]

$$p(\mathbf{X}|\mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X})p(\mathbf{X})}{p(\mathbf{Y})} \tag{2}$$

Conjugate Distributions

- The posterior and the prior are in the same *family*
- Relationship with all **three** terms

[1] `Wikipedia`, Bishop 2006, p. 2.4.2

Marginal

$$p(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{W}, \mathbf{X})p(\mathbf{W})\mathrm{d}\mathbf{W} \tag{3}$$

- Average according to belief and how well the model fits the observations
- "Pushes" uncertain belief in parameters (in this case) through to the observations
- Gaussian marginal is Gaussian

Dual Linear Regression[2]

$$[\mathbf{K}]_{ij} = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j \tag{4}$$

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^{\mathrm{T}}\mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{a}\mathbf{K}\mathbf{y} + \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{y} + \frac{\lambda}{2}\mathbf{a}^{\mathrm{T}}\mathbf{K}\mathbf{a} \tag{5}$$

$$\mathbf{a} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y} \tag{6}$$

[2] Bishop 2006, p. 6.1.

Dual Linear Regression[2]

$$[\mathbf{K}]_{ij} = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j \tag{7}$$

$$J(\mathbf{a}) = \frac{1}{2}\mathbf{a}^{\mathrm{T}}\mathbf{K}\mathbf{K}\mathbf{a} - \mathbf{a}\mathbf{K}\mathbf{y} + \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{y} + \frac{\lambda}{2}\mathbf{a}^{\mathrm{T}}\mathbf{K}\mathbf{a} \tag{8}$$

$$\mathbf{a} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y} \tag{9}$$

$$\mathbf{y}(\mathbf{x}_i) = \mathbf{w}\mathbf{x}_i = \mathbf{a}^{\mathrm{T}}\mathbf{X}\mathbf{x}_i = k(\mathbf{x}_i, \mathbf{X})^{\mathrm{T}}(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y} \tag{10}$$

[2] Bishop 2006, p. 6.1.

Kernels

Kernel Functions

- A function such that

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j) = \quad (11)$$

$$= ||\phi(\mathbf{x}_i)||||\phi(\mathbf{x}_j)||\cos(\theta) \quad (12)$$

- If we have $k(\cdot, \cdot)$ we *never* have to know the mapping $\phi(\cdot)$

The benefits of Kernels

- Kernels allows for *implicit* feature mappings
  - We do **NOT** need to know the feature space
  - Example: The space can have infinite dimensionality
  - The mapping can be non-linear but the problem is remains linear!
  - Allows for putting weird things like, strings (DNA) in a vector space

The benefits of Kernels

- Kernels allows for *implicit* feature mappings
  - We do **NOT** need to know the feature space
  - Example: The space can have infinite dimensionality
  - The mapping can be non-linear but the problem is remains linear!
  - Allows for putting weird things like, strings (DNA) in a vector space

The benefits of Kernels

- Kernels allows for *implicit* feature mappings
  - We do **NOT** need to know the feature space
  - Example: The space can have infinite dimensionality
  - The mapping can be non-linear but the problem is remains linear!
  - Allows for putting weird things like, strings (DNA) in a vector space

The benefits of Kernels

- Kernels allows for *implicit* feature mappings
  - We do **NOT** need to know the feature space
  - Example: The space can have infinite dimensionality
  - The mapping can be non-linear but the problem is remains linear!
  - Allows for putting weird things like, strings (DNA) in a vector space

This Lecture

- Kernel Methods
  - Implicit feature spaces
  - Building kernels
- Gaussian Processes
  - Priors over the space of functions
  - Learning parameters of kernels

Introduction

Recap

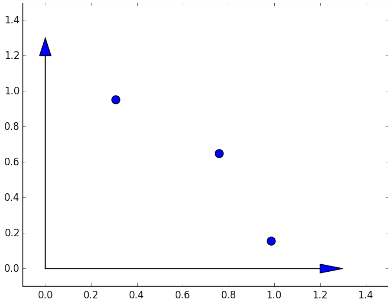
Kernels

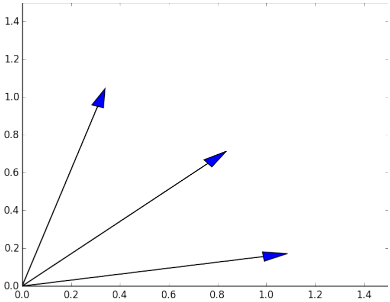
Gaussian Processes

Kernels

$$
\begin{aligned}
\sigma(\mathbf{X},\mathbf{Y}) &= \mathbb{E}\left[(\mathbf{X}-\mathbb{E}[\mathbf{X}])^{\mathrm{T}}(\mathbf{Y}-\mathbb{E}[\mathbf{Y}])\right] = \\
&= \mathbb{E}[\mathbf{X}^{\mathrm{T}}\mathbf{Y}] - \mathbb{E}[\mathbf{X}]^{\mathrm{T}}\mathbb{E}[\mathbf{Y}] = \{\mathbb{E}[\mathbf{X}] = \mathbb{E}[\mathbf{Y}] = \mathbf{0}\} = \\
&= \mathbb{E}[\mathbf{X}^{\mathrm{T}}\mathbf{Y}] \qquad (13)
\end{aligned}
$$

Kernels





Kernels

$$\sigma(\mathbf{X}, \mathbf{Y}) = \begin{bmatrix} x_{11} & x_{21} & x_{31} \\ x_{12} & x_{22} & x_{32} \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \\ y_{31} & y_{32} \end{bmatrix} = \tag{14}$$

$$= \begin{bmatrix} x_{11}y_{11} + x_{21}y_{21} + x_{31}y_{31} & x_{11}y_{12} + x_{21}y_{22} + x_{31}y_{32} \\ x_{12}y_{11} + x_{22}y_{21} + x_{32}y_{31} & x_{12}y_{12} + x_{22}y_{22} + x_{32}y_{32} \end{bmatrix}$$

Kernels

$$\sigma(\mathbf{X},\mathbf{Y}) = \begin{bmatrix} x_{11} & x_{21} & x_{31} \\ x_{12} & x_{22} & x_{32} \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \\ y_{31} & y_{32} \end{bmatrix} = \tag{15}$$

$$= \begin{bmatrix} x_{11}y_{11} + x_{21}y_{21} + x_{31}y_{31} & x_{11}y_{12} + x_{21}y_{22} + x_{31}y_{32} \\ x_{12}y_{11} + x_{22}y_{21} + x_{32}y_{31} & x_{12}y_{12} + x_{22}y_{22} + x_{32}y_{32} \end{bmatrix}$$

$$\sigma(\mathbf{X}^{\mathrm{T}},\mathbf{Y}^{\mathrm{T}}) = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix} \begin{bmatrix} y_{11} & y_{21} & y_{31} \\ y_{12} & y_{22} & y_{32} \end{bmatrix} = \tag{16}$$

$$= \begin{bmatrix} x_{11}y_{11} + x_{12}y_{12} & x_{11}y_{21} + x_{12}y_{22} & x_{11}y_{31} + x_{12}y_{32} \\ x_{21}y_{11} + x_{22}y_{12} & x_{21}y_{21} + x_{22}y_{22} & x_{21}y_{31} + x_{22}y_{32} \\ x_{31}y_{11} + x_{32}y_{12} & x_{31}y_{21} + x_{32}y_{22} & x_{31}y_{31} + x_{32}y_{32} \end{bmatrix}$$

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x},\mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

Kernels and covariances

- Covariance between columns: $\mathbf{X}^{\mathrm{T}}\mathbf{Y}$ (data-dimensions)
- Covariance between rows: $\mathbf{X}\mathbf{Y}^{\mathrm{T}}$ (data-points)
- Kernels: $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{y})$
  - Kernel functions are covariances between data-points
- A kernel function describes the co-variance of the *data* points
- Specific class of functions

Kernels

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 e^{-\frac{1}{2\ell^2}(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)} \tag{17}$$

Squared Exponential

- How does the data vary along the dimensions spanned by the data
- RBF, Squared Exponential, Exponentiated Quadratic
- Co-variance smoothly decays with distance

Building Kernels

Expression	Conditions
$k(\boldsymbol{x},\boldsymbol{z}) = c\,k_1(\boldsymbol{x},\boldsymbol{z})$	c - any non negative real constant.
$k(\boldsymbol{x},\boldsymbol{z}) = f(\boldsymbol{x})k_1(\boldsymbol{x},\boldsymbol{z})f(\boldsymbol{z})$	f - any real-valued function.
$k(\boldsymbol{x},\boldsymbol{z}) = q(k_1(\boldsymbol{x},\boldsymbol{z}))$	q - any polynomial with non-negative coefficients.
$k(\boldsymbol{x},\boldsymbol{z}) = \exp(k_1(\boldsymbol{x},\boldsymbol{z}))$	
$k(\boldsymbol{x},\boldsymbol{z}) = k_1(\boldsymbol{x},\boldsymbol{z}) + k_2(\boldsymbol{x},\boldsymbol{z})$	
$k(\boldsymbol{x},\boldsymbol{z}) = k_1(\boldsymbol{x},\boldsymbol{z})k_2(\boldsymbol{x},\boldsymbol{z})$	
$k(\boldsymbol{x},\boldsymbol{z}) = k_3(\phi(\boldsymbol{x}),\phi(\boldsymbol{z}))$	k_3 - valid kernel in the space mapped by ϕ.
$k(\boldsymbol{x},\boldsymbol{z}) = \langle \boldsymbol{A}\boldsymbol{x},\boldsymbol{z}\rangle = \langle \boldsymbol{x},\boldsymbol{A}\boldsymbol{z}\rangle$	$\boldsymbol{A}$ - symmetric psd matrix.
$k(\boldsymbol{x},\boldsymbol{z}) = k_a(\boldsymbol{x}_a,\boldsymbol{z}_a) + k_b(\boldsymbol{x}_b,\boldsymbol{z}_b)$	$\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ - non-necessarily disjoint partitions of $\boldsymbol{x}$;
$k(\boldsymbol{x},\boldsymbol{z}) = k_a(\boldsymbol{x}_a,\boldsymbol{z}_a)k_b(\boldsymbol{x}_b,\boldsymbol{z}_b)$	k_a and k_b - valid kernels on their respective spaces.

Summary

- Defines inner products in *some* space
- We don't need to know the space, its implicitly defined by the kernel function
- Defines co-variance between *data-points*

Summary

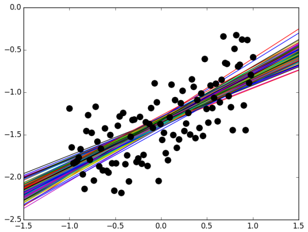
- Defines inner products in *some* space
- We don't need to know the space, its implicitly defined by the kernel function
- Defines co-variance between *data-points*

Summary

- Defines inner products in *some* space
- We don't need to know the space, its implicitly defined by the kernel function
- Defines co-variance between *data-points*

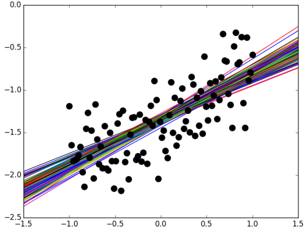
What have you seen up till now?

- Probabilistic modelling
  - likelihood, prior, posterior
  - marginalisation
- Implicit feature spaces
  - kernel functions
- We have assumed the form of the mapping without uncertainty



What have you seen up till now?

- Probabilistic modelling
  - likelihood, prior, posterior
  - marginalisation
- Implicit feature spaces
  - kernel functions
- We have assumed the form of the mapping without uncertainty



Outline

- General Regression
- Introduce uncertainty in mapping
- prior over the space of functions

Outline

- General Regression
- Introduce uncertainty in mapping
- prior over the space of functions

Outline

- General Regression
- Introduce uncertainty in mapping
- prior over the space of functions

Regression

Regression model,

$$\mathbf{y}_i = f(\mathbf{x}_i) + \boldsymbol{\epsilon} \tag{18}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}) \tag{19}$$

Introduce f_i as *instansiation* of function,

$$f_i = f(\mathbf{x}_i), \tag{20}$$

as a new random variable.

Regression

Model,

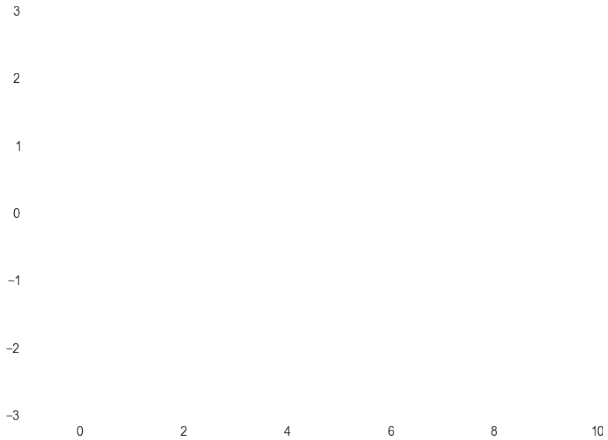
$$p(\mathbf{Y}, \mathbf{f}, \mathbf{X}, \boldsymbol{\theta}) = p(\mathbf{Y}|\mathbf{f})p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})p(\mathbf{X})p(\boldsymbol{\theta}) \tag{21}$$

Want to "push" $\mathbf{X}$ through a mapping f of which we are uncertain,

$$p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}), \tag{22}$$

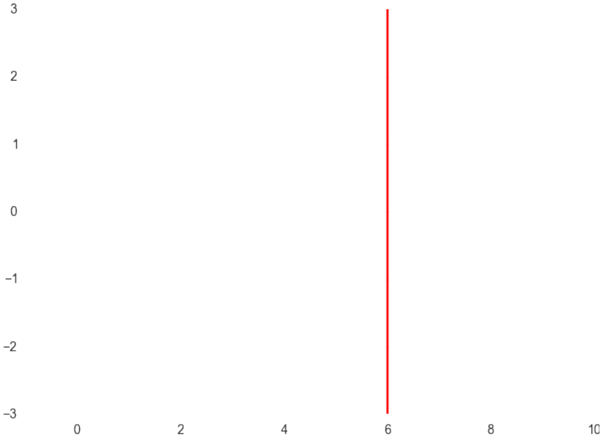
prior over instansiations of function.

Priors over functions[3]



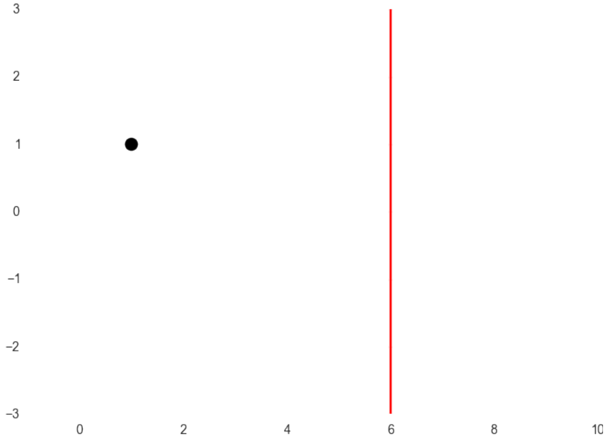
[3] `Lecture7/gp_basics.py`

Priors over functions[3]



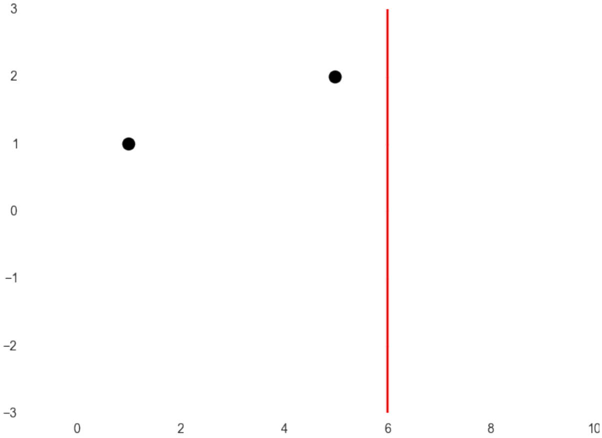
[3] Lecture7/gp_basics.py

Priors over functions[3]



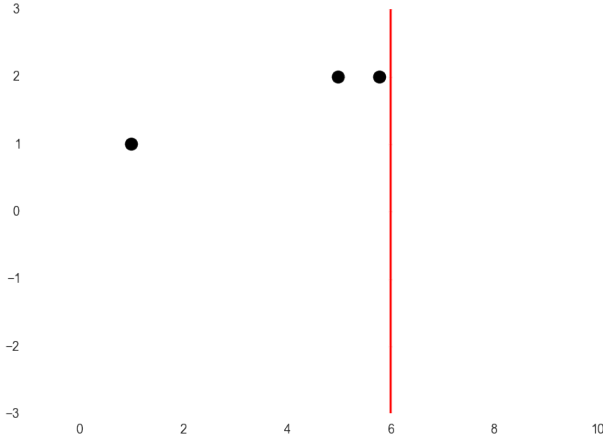
[3] `Lecture7/gp_basics.py`

Priors over functions[3]



[3] `Lecture7/gp_basics.py`

Priors over functions[3]



[3] Lecture7/gp_basics.py

Gaussian Distribution

Joint Distribution,

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \sigma(x_1, x_1) & \sigma(x_1, x_2) \\ \sigma(x_2, x_1) & \sigma(x_2, x_2) \end{bmatrix} \right). \tag{23}$$

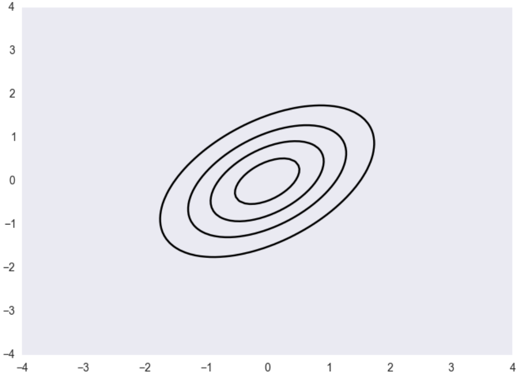
$$\begin{aligned} x_2 | x_1 \sim \mathcal{N}\big(&\mu_2 + \sigma(x_1, x_2)\sigma(x_1, x_1)^{-1}(x_1 - \mu_1), \\ &\sigma(x_2, x_2) - \sigma(x_2, x_1)\sigma(x_1, x_1)^{-1}\sigma(x_1, x_2)\big) \end{aligned} \tag{24}$$

The Gaussian Conditional[4]

$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) \tag{25}$$

[4] `Lecture7/conditional_gaussian.py`

The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) \tag{26}$$

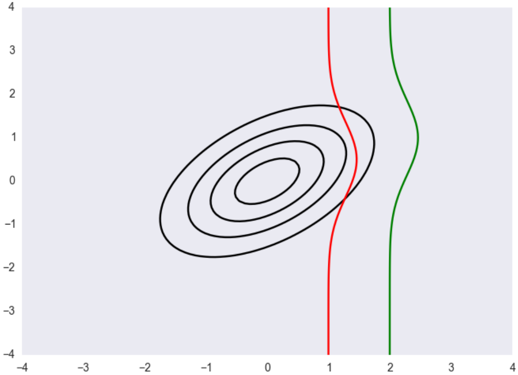
[4] `Lecture7/conditional_gaussian.py`

The Gaussian Conditional[4]

$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) \tag{27}$$

[4] `Lecture7/conditional_gaussian.py`

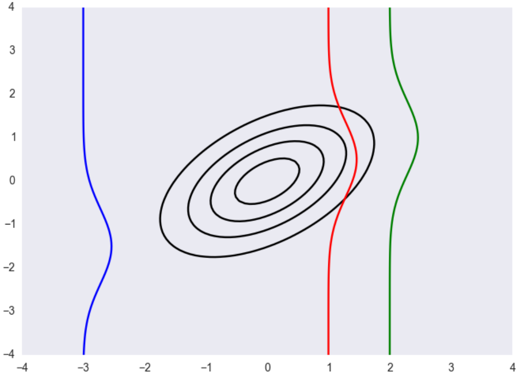
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) \quad (28)$$

[4] `Lecture7/conditional_gaussian.py`

The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) \tag{29}$$

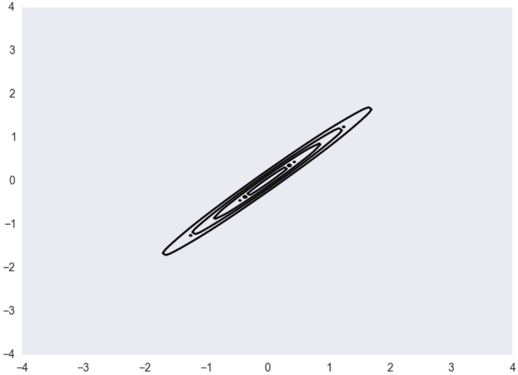
[4] Lecture7/conditional_gaussian.py

The Gaussian Conditional[4]

$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right) \tag{30}$$

[4] `Lecture7/conditional_gaussian.py`

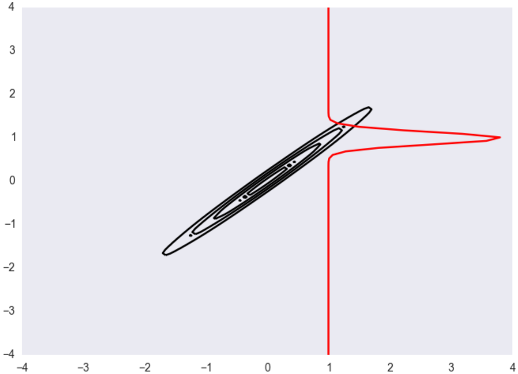
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right) \tag{31}$$

[4] `Lecture7/conditional_gaussian.py`

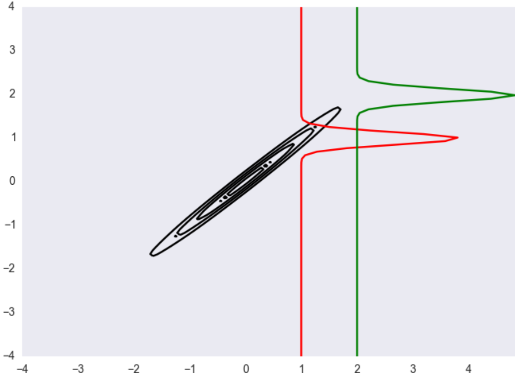
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right) \tag{32}$$

[4] `Lecture7/conditional_gaussian.py`

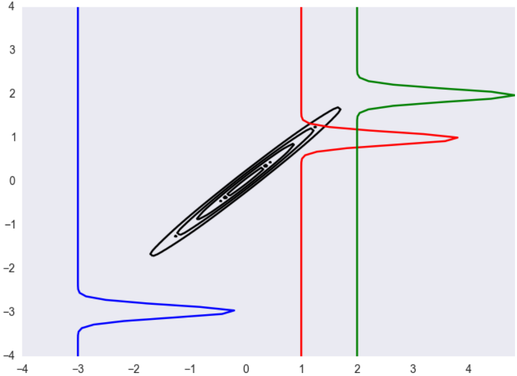
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right) \tag{33}$$

[4] `Lecture7/conditional_gaussian.py`

The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0.99 \\ 0.99 & 1 \end{bmatrix}\right) \tag{34}$$

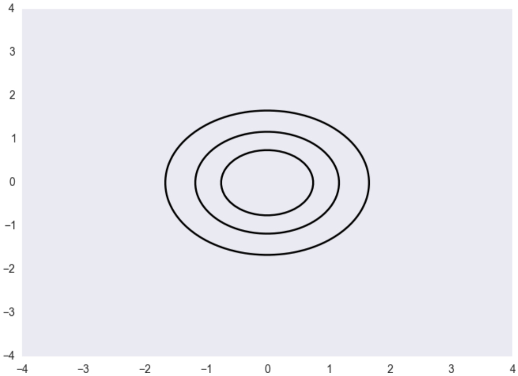
[4] `Lecture7/conditional_gaussian.py`

The Gaussian Conditional[4]

$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \tag{35}$$

[4] `Lecture7/conditional_gaussian.py`

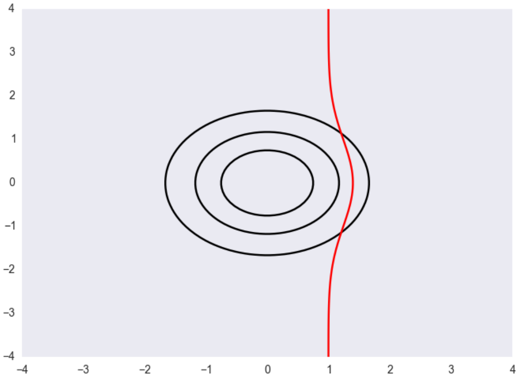
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \tag{36}$$

[4] `Lecture7/conditional_gaussian.py`

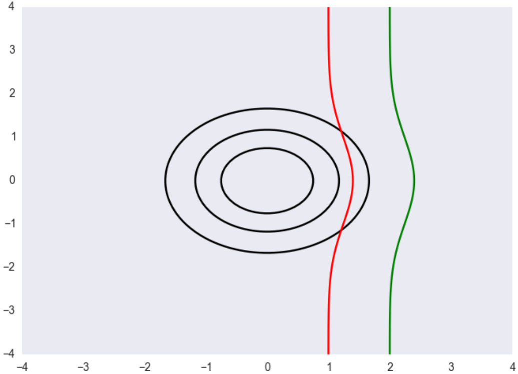
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \tag{37}$$

[4] `Lecture7/conditional_gaussian.py`

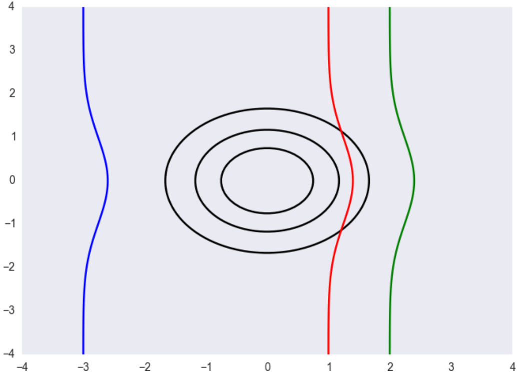
The Gaussian Conditional[4]



$$\mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix},\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \tag{38}$$

[4] `Lecture7/conditional_gaussian.py`

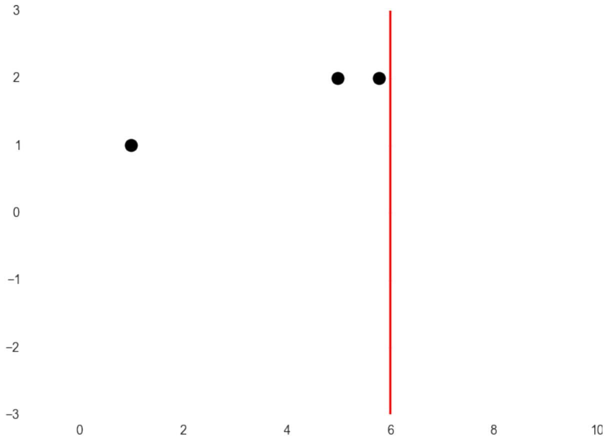
The Gaussian Conditional[4]

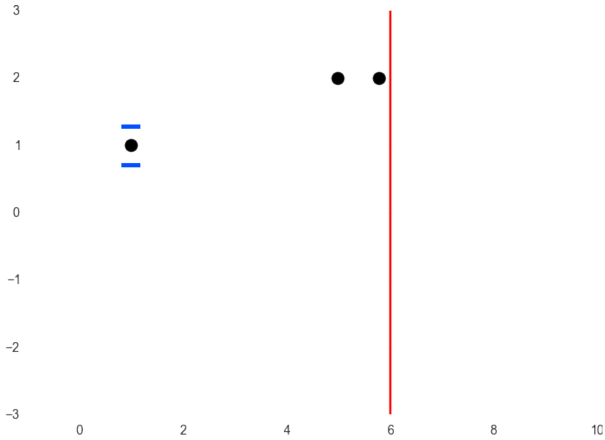


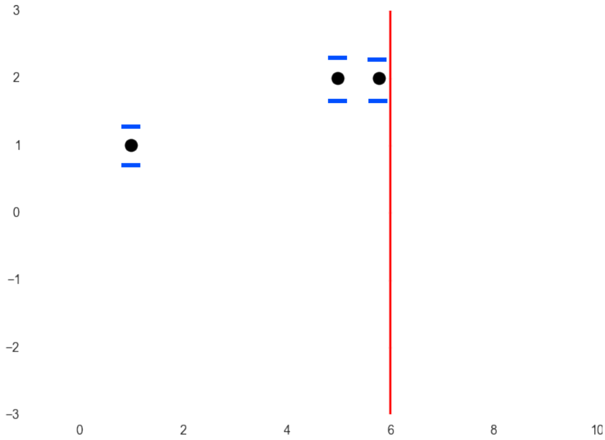
$$\mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \tag{39}$$

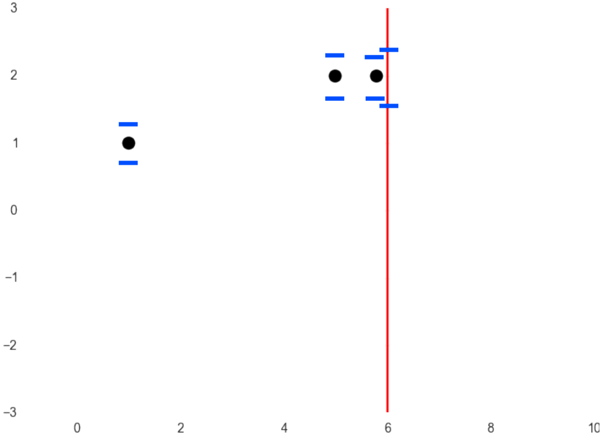
[4] `Lecture7/conditional_gaussian.py`

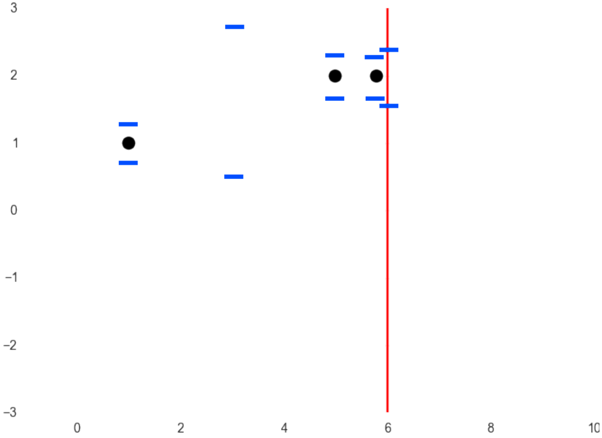

eureka!

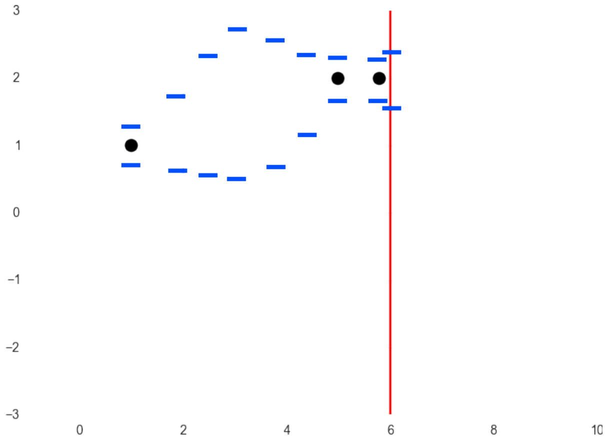

3
2
1
0
−1
−2
−3
0
2
4
6
8
10

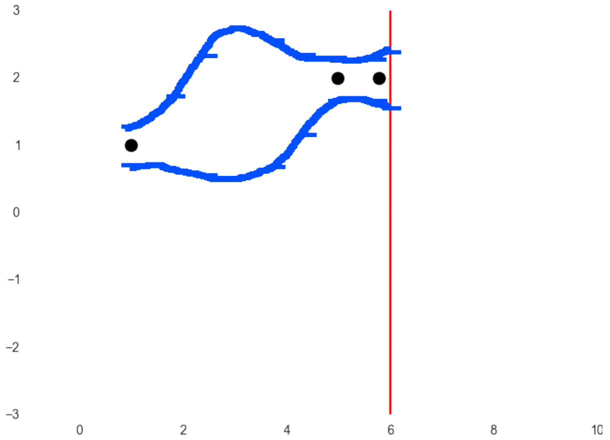

3
2
1
0
−1
−2
−3
0
2
4
6
8
10

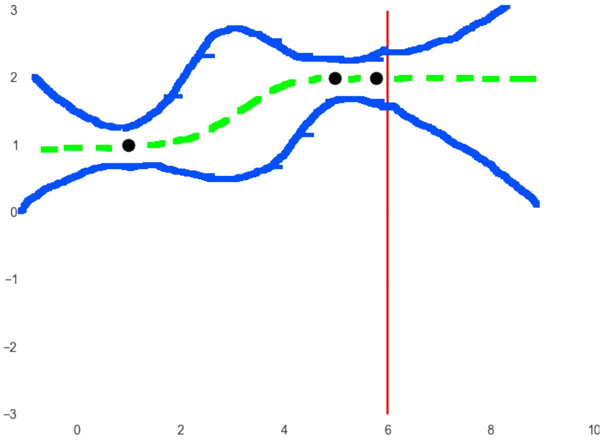

3
2
1
0
−1
−2
−3
0
2
4
6
8
10

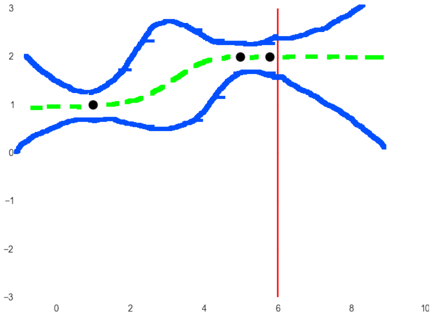

3
2
1
0
−1
−2
−3
0
2
4
6
8
10


3
2
1
0
−1
−2
−3
0
2
4
6
8
10

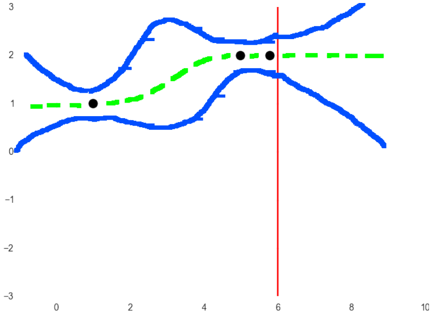

3
2
1
0
−1
−2
−3
0
2
4
6
8
10


3
2
1
0
−1
−2
−3
0
2
4
6
8
10


3
2
1
0
−1
−2
−3
0
2
4
6
8
10

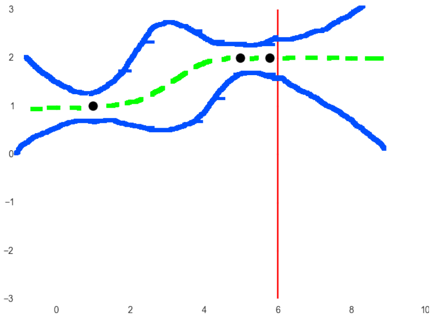


If all instansiations of the function is jointly Gaussian such that the co-variance structure depends on how much information an observation provides for the other we will get the curve above.



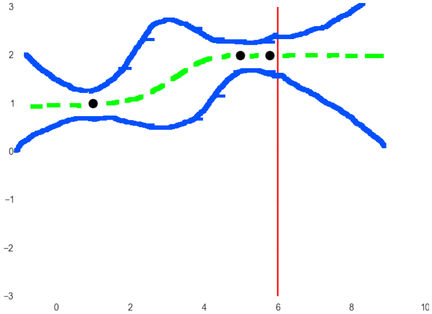
Row space

- Co-variance between each point!
- *Co-variance function is a kernel!*
- We can do all this in induced space, i.e. allow for any function!



Row space

- Co-variance between each point!
- *Co-variance function is a kernel!*
- We can do all this in induced space, i.e. allow for any function!



Row space

- Co-variance between each point!
- *Co-variance function is a kernel!*
- We can do all this in induced space, i.e. allow for any function!

Gaussian Processes[5]

$$p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta}) \sim \mathcal{GP}(\mu(\mathbf{X}), k(\mathbf{X},\mathbf{X})) \tag{40}$$

Defenition

A Gaussian Process is an infinite collection of random variables who **any** subset is jointly gaussian. The process is specified by a mean function $\mu(\cdot)$ and a co-variance function $k(\cdot,\cdot)$

$$f \sim \mathcal{GP}(\mu(\cdot), k(\cdot,\cdot)) \tag{41}$$

[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

$$p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta}) \sim \mathcal{GP}(\mu(\mathbf{X}),k(\mathbf{X},\mathbf{X})) \quad (42)$$

$$\mathbf{y}_i = f_i + \boldsymbol{\epsilon} \quad (43)$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I}) \quad (44)$$

$$p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta}) = \int p(\mathbf{Y}|\mathbf{f})p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta})\mathrm{d}f \quad (45)$$

Connection to Distribution

$\mathcal{GP}$ is infinite, but we only observe finite amount of data. This means conditioning on a subset of the data, the $\mathcal{GP}$ is a just a Gaussian distribution, which is self-conjugate.

[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

The mean function

- Function of only the input location
- What do I expect the function value to be **only** accounting for the input location
- We will assume this to be constant

The co-variance function

- Function of **two** input locations
- How should the information from other locations with **known** function value observations effect my estimate
- Encodes the behavior of the function

[5]Bishop 2006, p. 6.4.2

Gaussian Processes[5]

The mean function

- Function of only the input location
- What do I expect the function value to be **only** accounting for the input location
- We will assume this to be constant

The co-variance function

- Function of **two** input locations
- How should the information from other locations with **known** function value observations effect my estimate
- Encodes the behavior of the function

[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

The mean function

- Function of only the input location
- What do I expect the function value to be **only** accounting for the input location
- We will assume this to be constant

The co-variance function

- Function of **two** input locations
- How should the information from other locations with **known** function value observations effect my estimate
- Encodes the behavior of the function

[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

The Prior

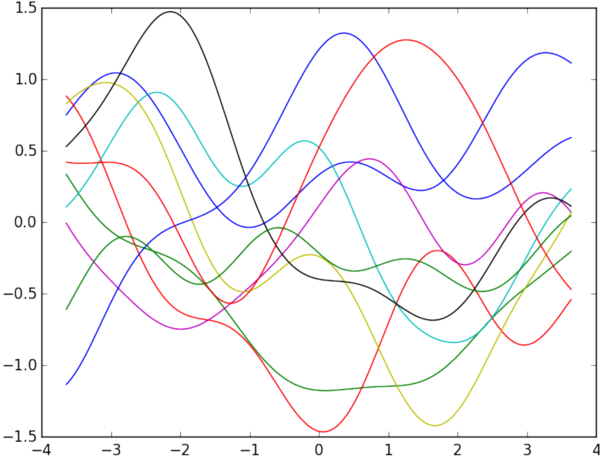
$$p(f|\mathbf{X},\boldsymbol{\theta}) = \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x},\mathbf{x}')) \tag{46}$$

$$\mu(\mathbf{x}) = \mathbf{0} \tag{47}$$

$$k(\mathbf{x}_i,\mathbf{x}_j) = \sigma^2 e^{-\frac{1}{2\ell^2}(\mathbf{x}_i-\mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i-\mathbf{x}_j)} \tag{48}$$

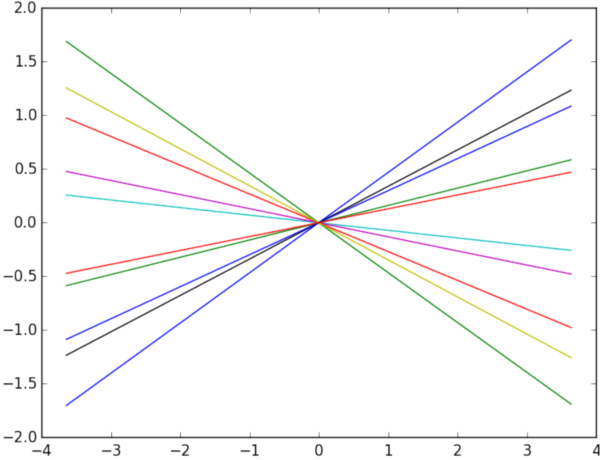
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



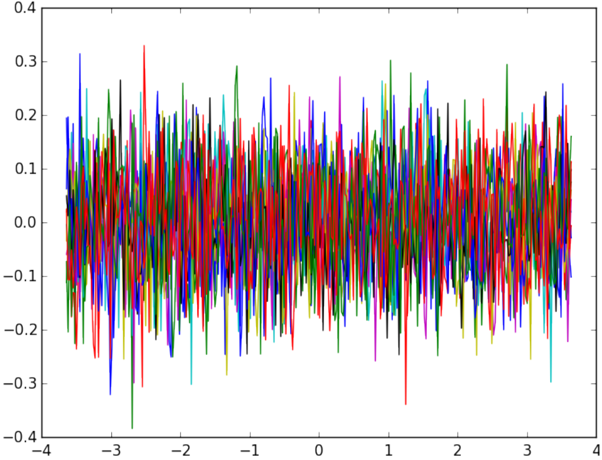
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



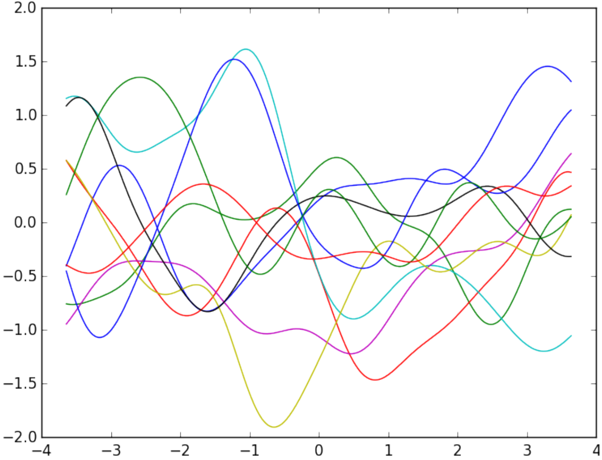
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



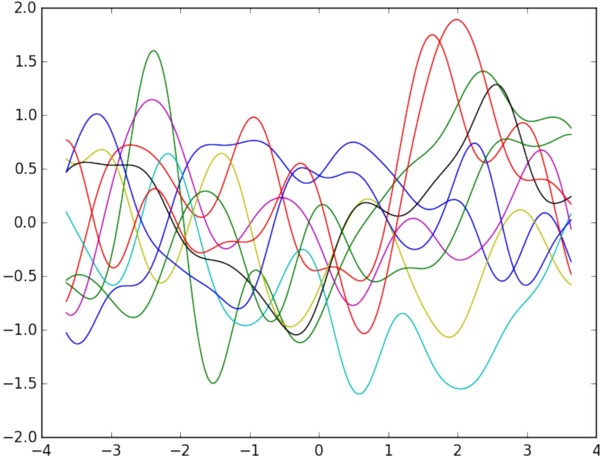
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



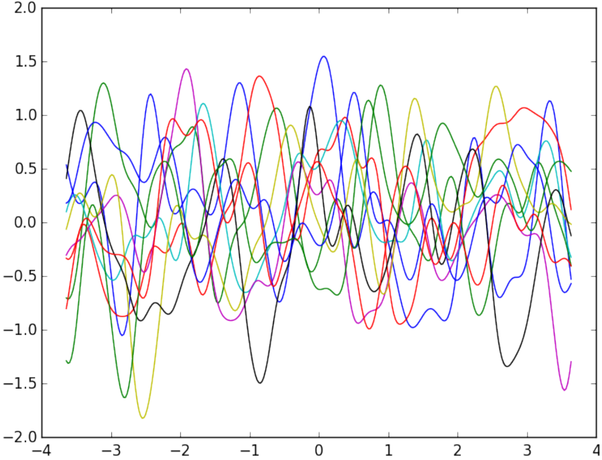
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



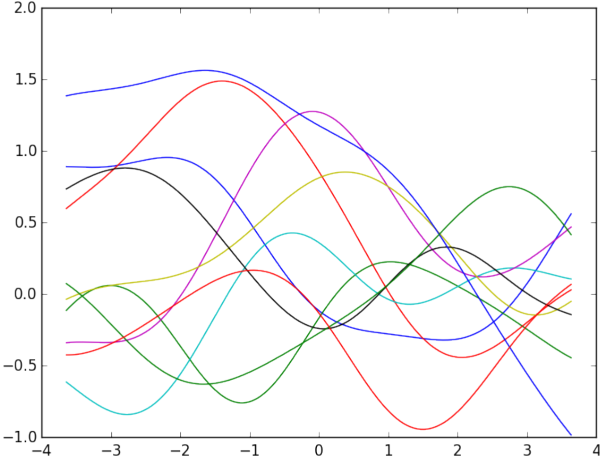
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



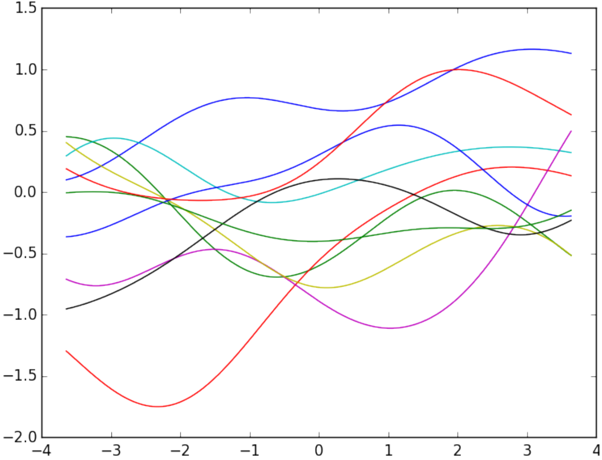
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



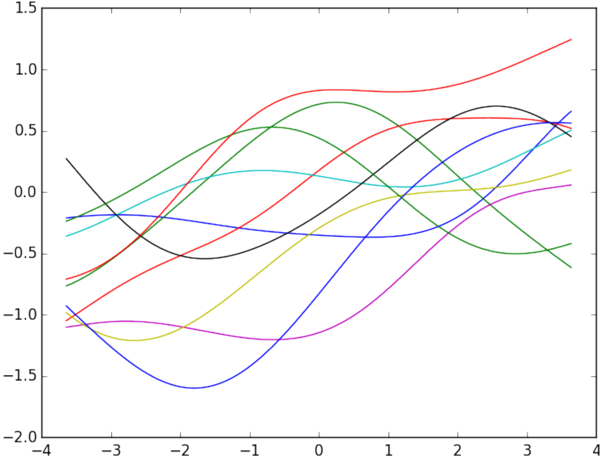
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



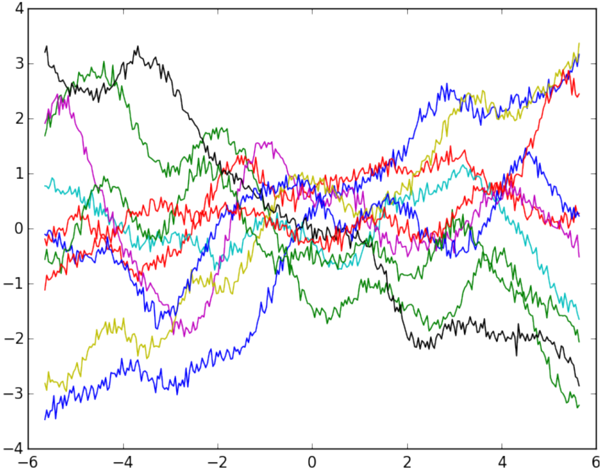
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



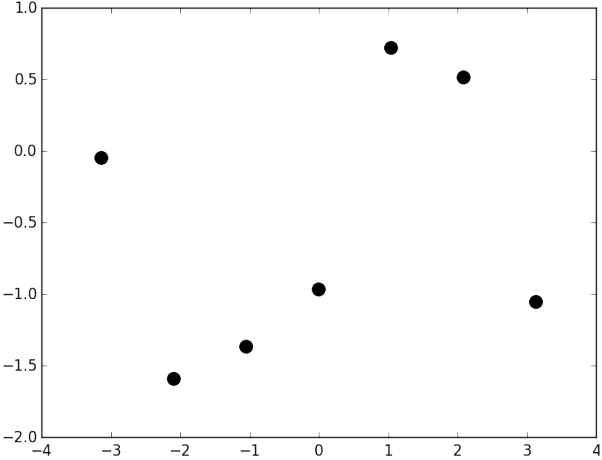
[5]Bishop 2006, p. 6.4.2

Gaussian Processes[5]



[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

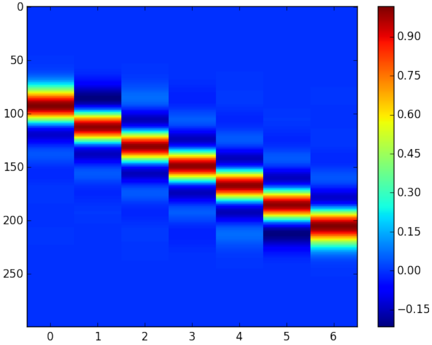
The (predictive) Posterior

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{X},\mathbf{X}) & k(\mathbf{X},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{X}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix} \right) \tag{49}$$

$$\begin{aligned} p(f_*|\mathbf{x}_*,\mathbf{X},\mathbf{f},\boldsymbol{\theta}) = \mathcal{N}(&k(\mathbf{x}_*,\mathbf{X})^{\mathrm{T}}K(\mathbf{X},\mathbf{X})^{-1}\mathbf{f}, \\ &k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{X})^{\mathrm{T}}K(\mathbf{X},\mathbf{X})^{-1}K(\mathbf{X},\mathbf{x}_*)) \end{aligned} \tag{50}$$

[5] Bishop 2006, p. 6.4.2

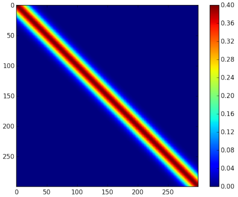
Gaussian Processes[5]



$$k(\mathbf{x}_*, \mathbf{X})^{\mathrm{T}} K(\mathbf{X}, \mathbf{X})^{-1} \mathbf{f} \tag{51}$$

[5] Bishop 2006, p. 6.4.2

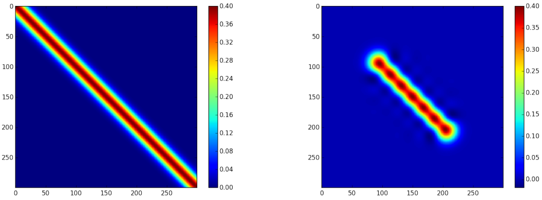
Gaussian Processes[5]



$$k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*, \mathbf{X})^{\mathrm{T}} K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, \mathbf{x}_*) \tag{52}$$

[5] Bishop 2006, p. 6.4.2

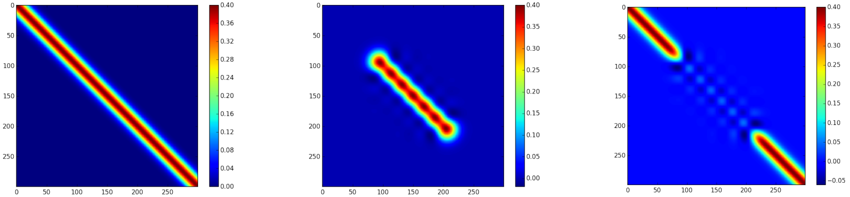
Gaussian Processes[5]



$$k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*, \mathbf{X})^{\mathrm{T}} K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, \mathbf{x}_*) \tag{53}$$

[5] Bishop 2006, p. 6.4.2

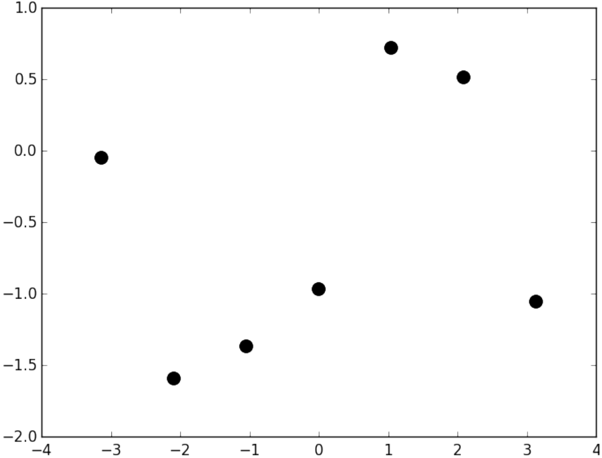
Gaussian Processes[5]



$$k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*, \mathbf{X})^{\mathrm{T}} K(\mathbf{X}, \mathbf{X})^{-1} K(\mathbf{X}, \mathbf{x}_*) \quad (54)$$

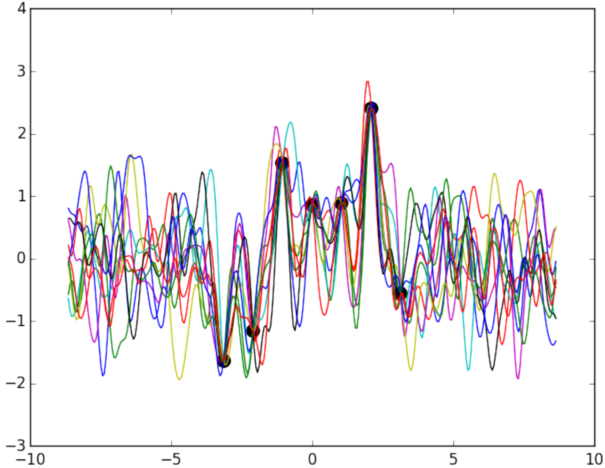
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



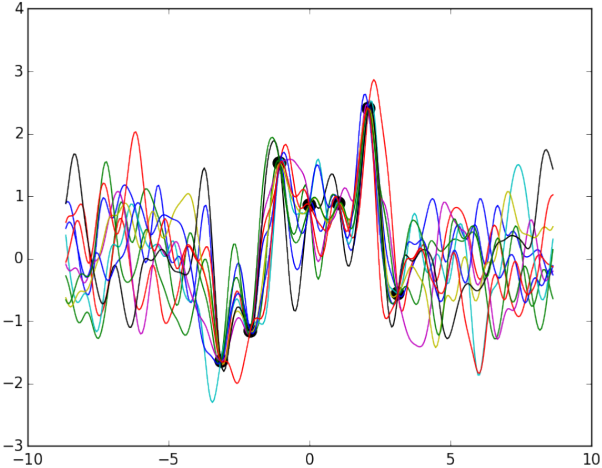
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



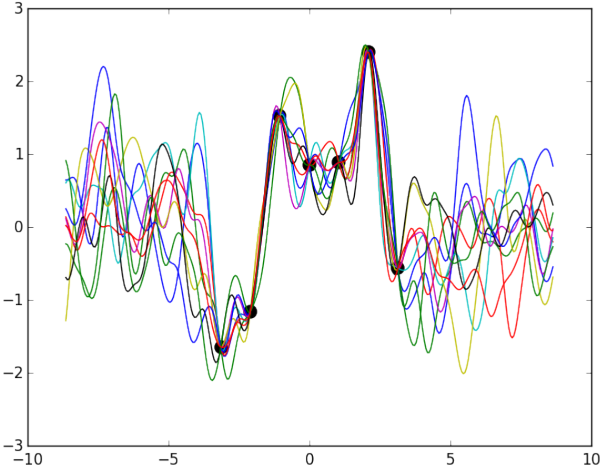
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



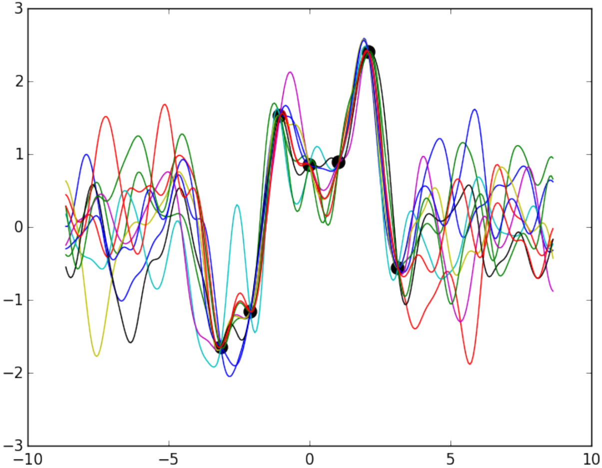
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



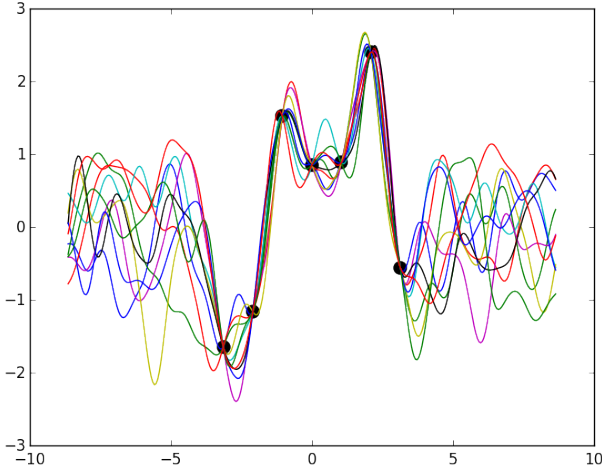
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



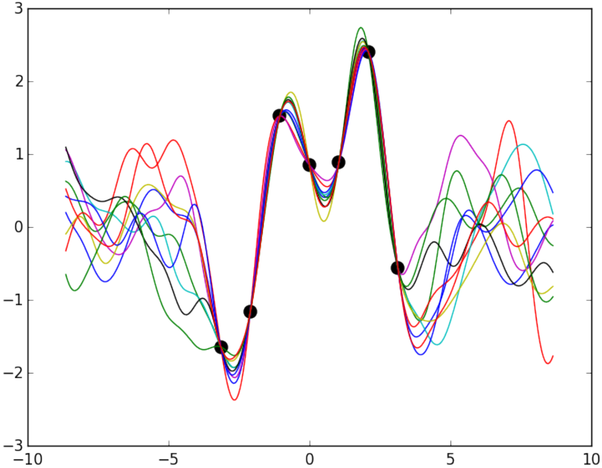
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



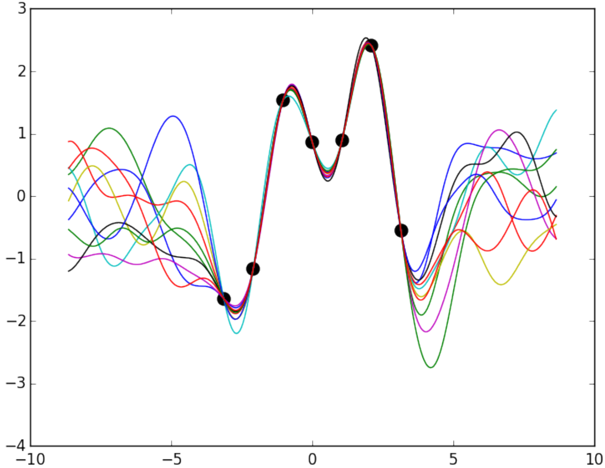
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



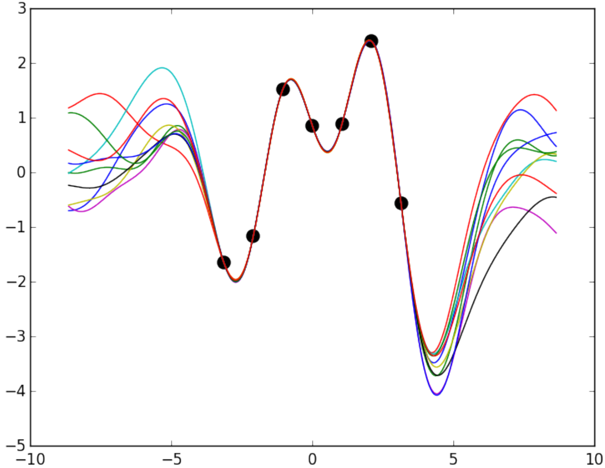
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



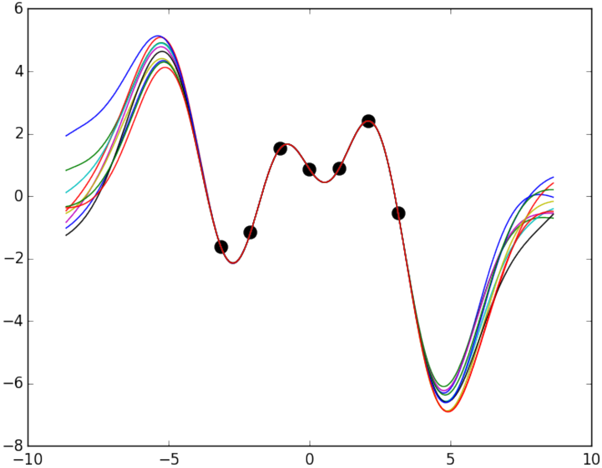
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



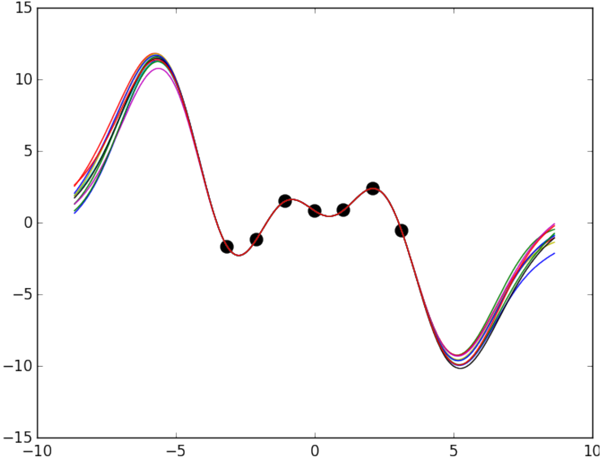
[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]



[5] Bishop 2006, p. 6.4.2

Gaussian Processes[5]

Summary

- $\mathcal{GP}$ is a prior over function realisations
- Introduce new random variable as the output of the mapping
- Joint distribution of **any** observations Gaussian
- Posterior (predictive) distribution is conditional Gaussian

[5] Bishop 2006, p. 6.4.2

Co-variances in practice

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} k(\mathbf{X},\mathbf{X}) + \sigma^2\mathbf{I} & k(\mathbf{X},\mathbf{x}_*) \\ k(\mathbf{x}_*,\mathbf{X}) & k(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix} \right) \tag{55}$$

- The conditional distribution passes *exactly through the data*
  - noise-free observations
- Construct covariance functions by rules for building kernels
  - $k(\mathbf{x}_i, \mathbf{x}_j) = \lambda_1 k_{\text{SE}}(\mathbf{x}_i, \mathbf{x}_j) + \lambda_2 k_{\text{lin}}(\mathbf{x}_i, \mathbf{x}_j) + \lambda_3 k_{\text{white}}(\mathbf{x}_i, \mathbf{x}_j)$

Co-variances in practice

Periodic kernel,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 e^{-\frac{2}{\ell^2}\sin^2\left(\pi\frac{|\mathbf{x}_i - \mathbf{x}_j|}{p}\right)} \quad (56)$$

Periodic functions

- ℓ lengthscale
- p period of function

Co-variances in practice

$$k_{\text{lin}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j) \tag{57}$$

$$k(\mathbf{x}_i, \mathbf{x}_j) = \frac{2}{\pi}\sin^{-1}\left(\frac{2\mathbf{x}_i^{\mathrm{T}}\Sigma\mathbf{x}_j}{\sqrt{(1+2\mathbf{x}_i^{\mathrm{T}}\Sigma\mathbf{x}_i)(1+2\mathbf{x}_j^{\mathrm{T}}\Sigma\mathbf{x}_j)}}\right) \tag{58}$$

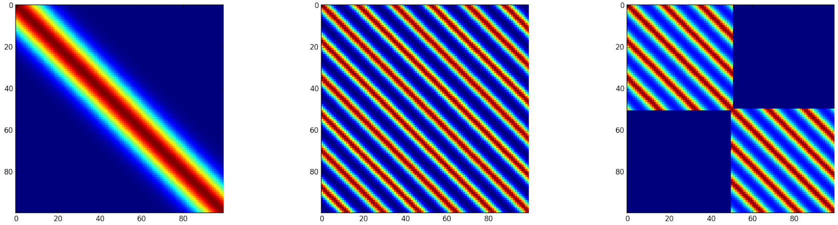
$$\mathbf{x}_i = [1, x_{1i}, \ldots, x_{qi}]^{\mathrm{T}} \tag{59}$$

"Computation with Infinite Neural Networks", Williams

Non-stationary functions

- Non-stationary co-variance
- Functions that have different behaviour in different parts of domain

Co-variances in practice



[6]

$$[\mathbf{K}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) \tag{60}$$

[6] /Lecture7/covariance.py

Co-variances in practice

Summary

- Covariance functions encodes your *preference* in function behavior
- Choosing the right co-variance is very important
- Ask yourself what do you know about the variations in the data

Assignment

You should now be able to do Task 2.2 of the Assignment

Learning in Gaussian Processes[6]

Hyper-parameters

- Prior has parameters
  - referred to as *hyper*-parameters
  - SE have lengthscale and variance
- Learning in $\mathcal{GP}$s implies inferring hyper-parameters from the model

[6]Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{Y}|\mathbf{f})p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})\mathrm{d}f \tag{61}$$

Marginal Likelihood

- We are not interested in $\mathbf{f}$ directly
- Marginalise out $\mathbf{f}$!
- Gaussian marginal is gaussian

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

$$p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta}) = \int p(\mathbf{Y}|\mathbf{f})p(\mathbf{f}|\mathbf{X},\boldsymbol{\theta})\mathrm{d}f \quad (62)$$

Marginal Likelihood

- We are not interested in $\mathbf{f}$ directly
- Marginalise out $\mathbf{f}$!
- Gaussian marginal is gaussian

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{Y}|\mathbf{f})p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})\mathrm{d}f \tag{63}$$

Marginal Likelihood

- We are not interested in $\mathbf{f}$ directly
- Marginalise out $\mathbf{f}$!
- Gaussian marginal is gaussian

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

Learning

- Type-II Maximum Likelihood

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \tag{64}$$

- How is this different to a normal ML estimate?
- Lots of exponentials in objective implies working in log-space
  - Logarithm monotonic function ⇒ does not alter the location of extreme points of a function
  - Minimisation of negative log() rather than maximisation of log() purely practical

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

Learning

- Type-II Maximum Likelihood

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \tag{65}$$

- How is this different to a normal ML estimate?
- Lots of exponentials in objective implies working in log-space
  - Logarithm monotonic function $\Rightarrow$ does not alter the location of extreme points of a function
  - Minimisation of negative log() rather than maximisation of log() purely practical

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

Learning

- Type-II Maximum Likelihood

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \tag{66}$$

- How is this different to a normal ML estimate?
- Lots of exponentials in objective implies working in log-space
  - Logarithm monotonic function $\Rightarrow$ does not alter the location of extreme points of a function
  - Minimisation of negative log() rather than maximisation of log() purely practical

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

Learning

- Type-II Maximum Likelihood

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \tag{67}$$

- How is this different to a normal ML estimate?
- Lots of exponentials in objective implies working in log-space
  - Logarithm monotonic function $\Rightarrow$ does not alter the location of extreme points of a function
  - Minimisation of negative log() rather than maximisation of log() purely practical

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

Learning

- Type-II Maximum Likelihood

$$\hat{\boldsymbol{\theta}} = \text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \tag{68}$$

- How is this different to a normal ML estimate?
- Lots of exponentials in objective implies working in log-space
  - Logarithm monotonic function $\Rightarrow$ does not alter the location of extreme points of a function
  - Minimisation of negative log() rather than maximisation of log() purely practical

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

$$\text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{\theta}} - \log\left(p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta})\right) = \text{argmin}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \tag{69}$$

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi) \tag{70}$$

- Can be minimised using gradient based methods
- Data-fit: $\frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y}$
- Complexity: $\frac{1}{2}\log|\mathbf{K}|$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

$$\text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{\theta}} - \log\left(p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})\right) = \text{argmin}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \tag{71}$$

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi) \tag{72}$$

- Can be minimised using gradient based methods
- Data-fit: $\frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y}$
- Complexity: $\frac{1}{2}\log|\mathbf{K}|$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

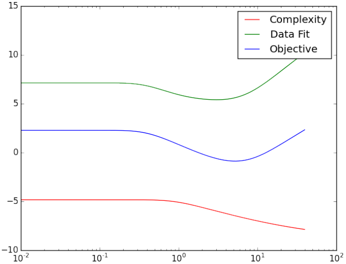
$$\text{argmax}_{\boldsymbol{\theta}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{\theta}} - \log\left(p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})\right) = \text{argmin}_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \tag{73}$$

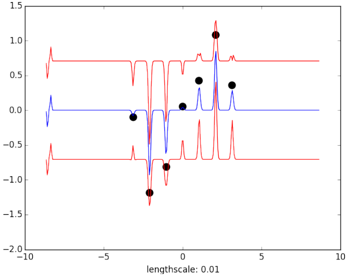
$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi) \tag{74}$$

- Can be minimised using gradient based methods
- Data-fit: $\frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y}$
- Complexity: $\frac{1}{2}\log|\mathbf{K}|$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

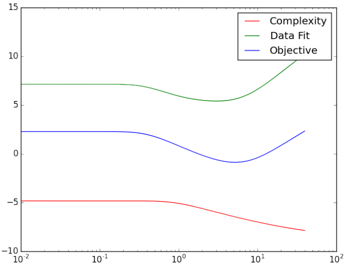


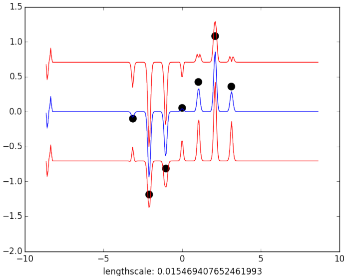


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

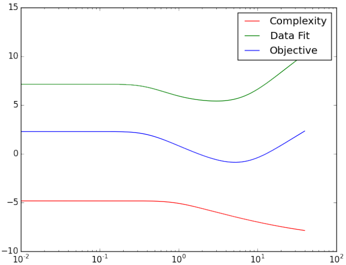


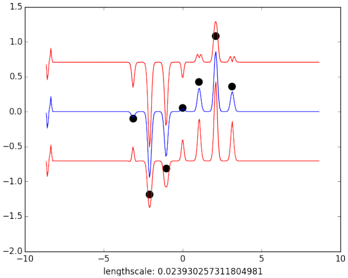


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

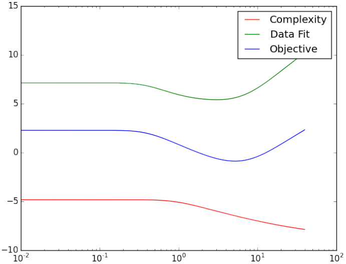


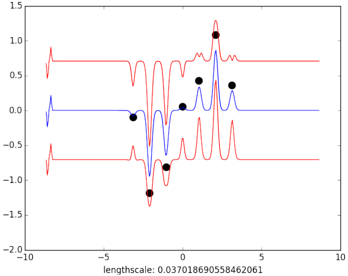


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

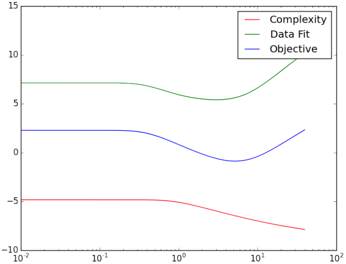


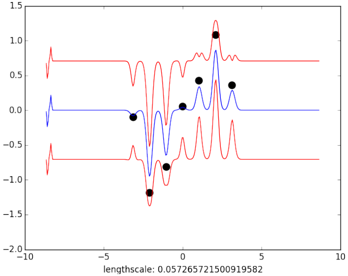


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

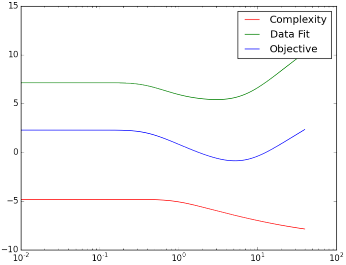


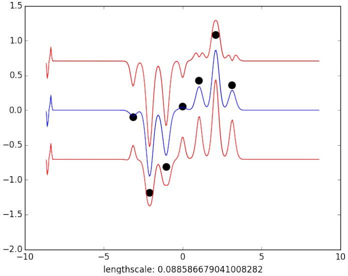


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

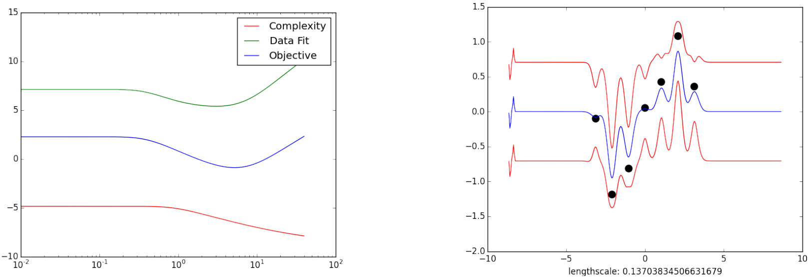




$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

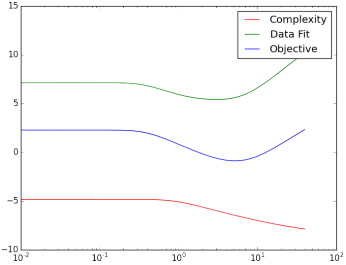
Learning in Gaussian Processes[6]

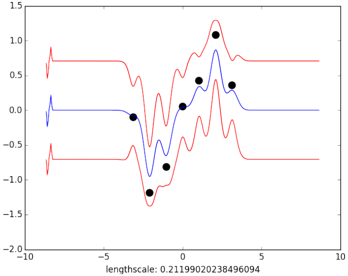


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

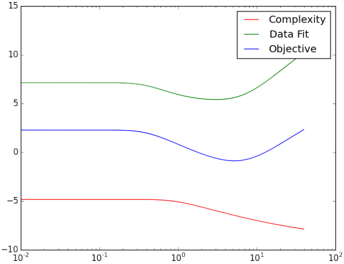


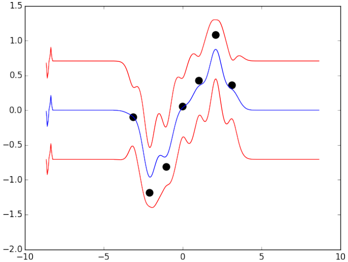


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

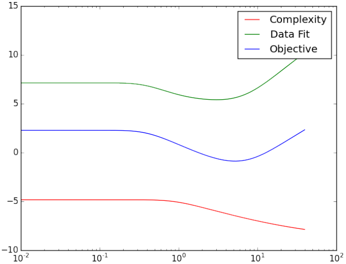


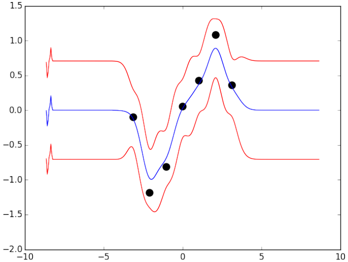


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

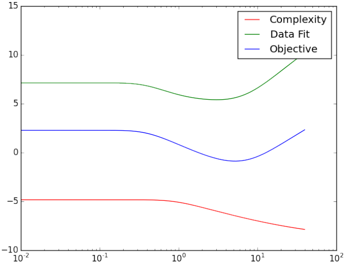


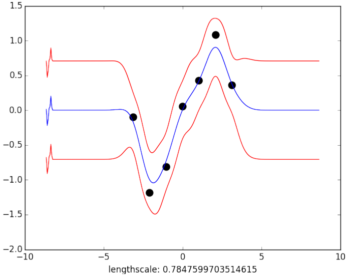


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

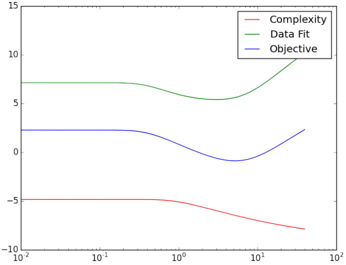


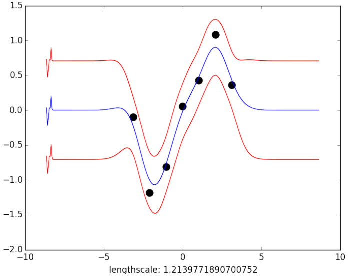


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6]Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

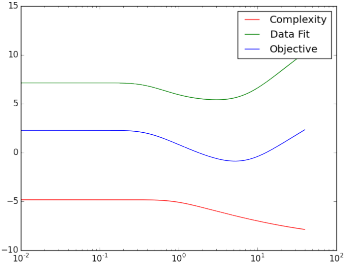


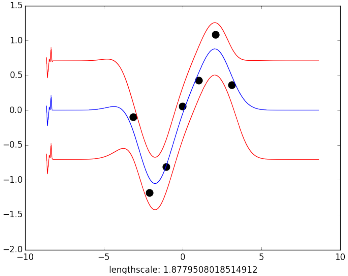


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

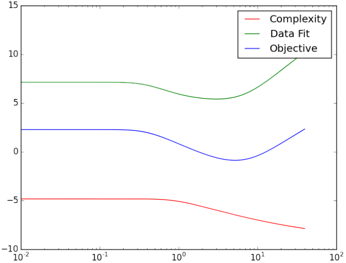


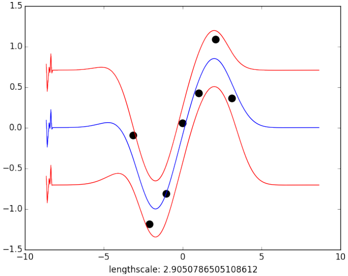


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6]Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

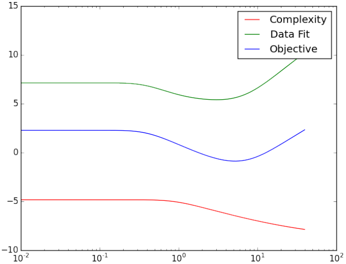


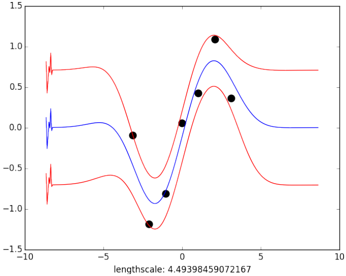


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

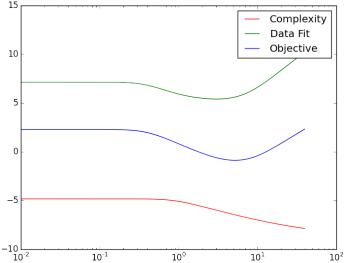


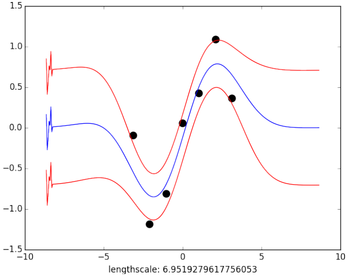


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6]Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

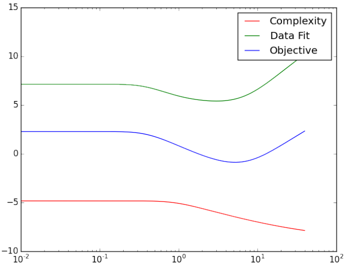


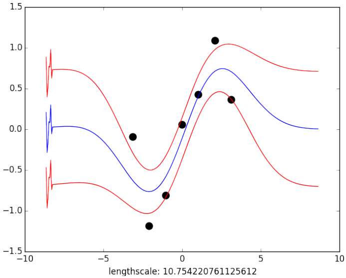


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

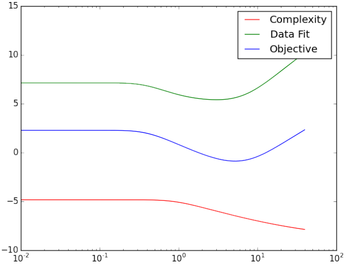


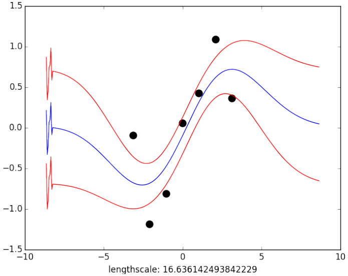


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

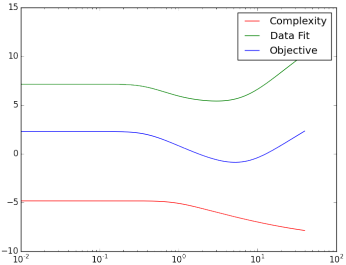


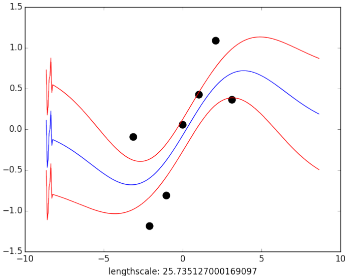


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

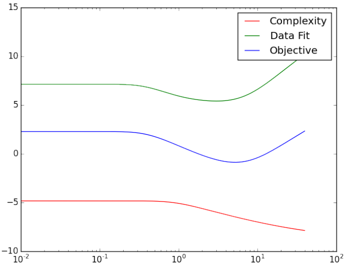


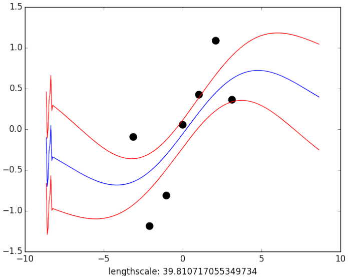


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]

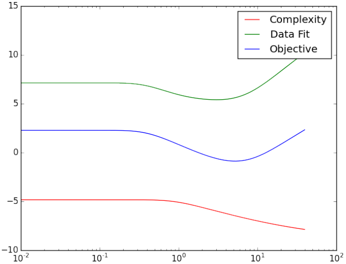


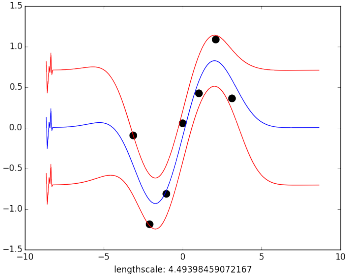


$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Learning in Gaussian Processes[6]





$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{y} + \frac{1}{2}\log|\mathbf{K}| + \frac{N}{2}\log(2\pi)$$

[6] Bishop 2006, p. 6.4.3

Summary

- Kernels are covariance functions of data-points
- Gaussian processes are priors over functions
- $\mathcal{GP}$'s allows us to average over *all* possible functions
- Nothing different compared to Lecture 2, just a different prior!

Summary

- Kernels are covariance functions of data-points
- Gaussian processes are priors over functions
- $\mathcal{GP}$'s allows us to average over *all* possible functions
- Nothing different compared to Lecture 2, just a different prior!

Summary

- Kernels are covariance functions of data-points
- Gaussian processes are priors over functions
- $\mathcal{GP}$'s allows us to average over *all* possible functions
- Nothing different compared to Lecture 2, just a different prior!

Summary

- Kernels are covariance functions of data-points
- Gaussian processes are priors over functions
- $\mathcal{GP}$'s allows us to average over *all* possible functions
- Nothing different compared to Lecture 2, just a different prior!

Next Time

Practical 1

- November 6th 15-17 V1
- My best friend the Gaussian
  - derive Gaussian identities
- Complete assignment Task 2.1 and 2.2

Next Time

Practical 1

- November 6th 15-17 V1
- My best friend the Gaussian
  - derive Gaussian identities
- Complete assignment Task 2.1 and 2.2

e.o.f.

References I

Christopher M Bishop. *Pattern recognition and machine learning*. 2006.

Christopher K I Williams. “Computation with Infinite Neural Networks”. In: *Neural Computation* 10 (July 1998), pp. 1203–1216.