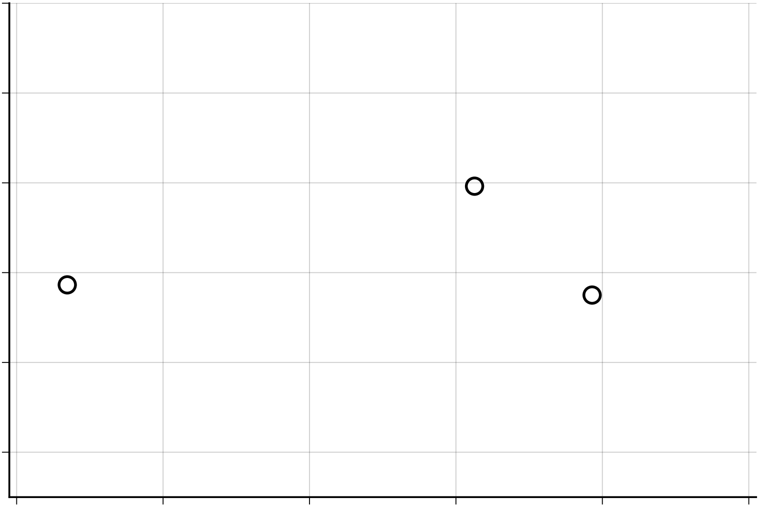
# Part I: Gaussian Processes for Regression and Classification

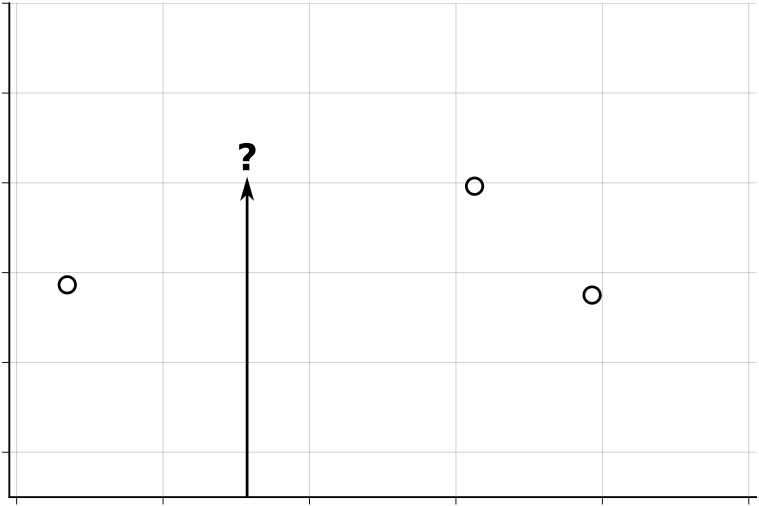
**Daniel Hernández–Lobato**
Computer Science Department
Universidad Autónoma de Madrid

http://dhnzl.org, daniel.hernandez@uam.es

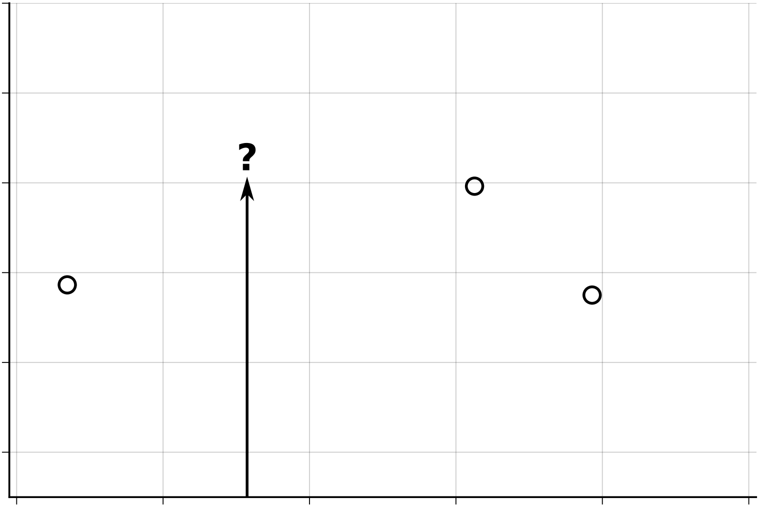
# Motivation: Regression Problems



# Motivation: Regression Problems



# Motivation: Regression Problems




**We have to specify a model that may depend on parameters w.**

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^\mathsf{T}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^\mathsf{T}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer w from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer $\mathbf{w}$ from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.

We follow a Bayesian approach to machine learning:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}\,, \qquad p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}\,.$$

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer w from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.

We follow a Bayesian approach to machine learning:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}\,, \qquad p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}\,.$$

Prior: **Initial belief** on the values of $\mathbf{w}$ before observing the data.

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer w from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.

We follow a Bayesian approach to machine learning:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}\,, \qquad p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}\,.$$

Prior: **Initial belief** on the values of $\mathbf{w}$ before observing the data.

Likelihood: **How well** each value of $\mathbf{w}$ explains $\mathcal{D}$.

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer w from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$.

We follow a Bayesian approach to machine learning:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}\,, \qquad p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}\,.$$

Prior: **Initial belief** on the values of **w** before observing the data.

Likelihood: **How well** each value of **w** explains $\mathcal{D}$.

Posterior: **Updated belief** on the values of **w** after observing $\mathcal{D}$.

# The Standard Linear Model

We may consider a standard linear regression model:

$$f(\mathbf{x}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}\,, \qquad y = f(\mathbf{x}) + \epsilon\,, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,,$$

The task of interest is **to infer w from data** $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$.

We follow a Bayesian approach to machine learning:

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}\,, \qquad p(\mathbf{w}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{w}, \mathbf{X})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})}\,.$$

Prior: **Initial belief** on the values of **w** before observing the data.

Likelihood: **How well** each value of **w** explains $\mathcal{D}$.

Posterior: **Updated belief** on the values of **w** after observing $\mathcal{D}$.

Marginal Likelihood: **Probability** of observing **y** under the model.

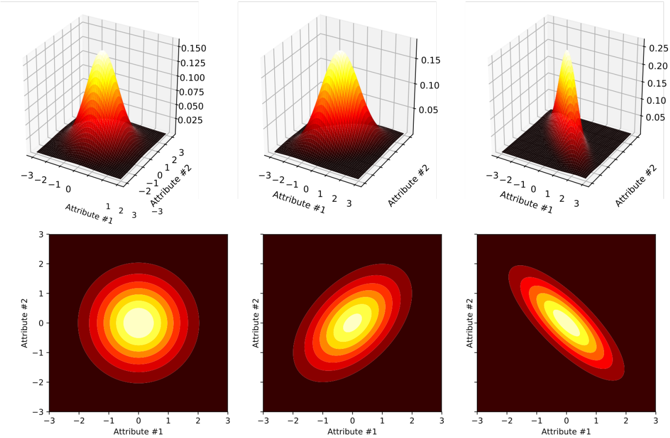
# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

# Multivariate Gaussian Distribution

$$p(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{N}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left\{-0.5 \cdot (\mathbf{w} - \boldsymbol{\mu})^\mathsf{T} \boldsymbol{\Sigma}^{-1} (\mathbf{w} - \boldsymbol{\mu})\right\}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$$



# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}\sigma^{-2} + \mathbf{I}$.

# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\top\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\top\mathbf{X}\sigma^{-2} + \mathbf{I}$.

Marginal Likelihood: Given by $\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{X}\mathbf{X}^\top + \mathbf{I}\sigma^2)$.

(Bishop, 2006)

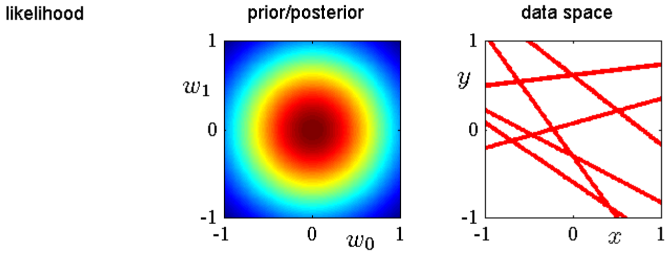
# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}\sigma^{-2} + \mathbf{I}$.

Marginal Likelihood: Given by $\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{X}\mathbf{X}^\mathsf{T} + \mathbf{I}\sigma^2)$.




(Bishop, 2006)

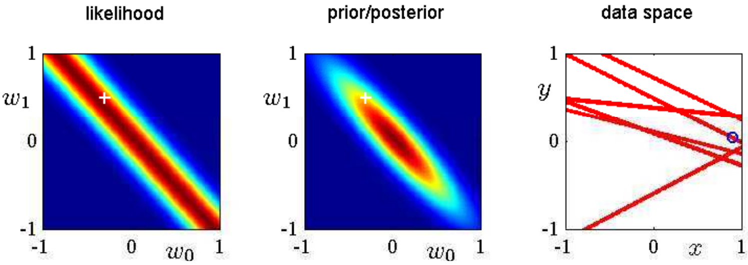
# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}\sigma^{-2} + \mathbf{I}$.

Marginal Likelihood: Given by $\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{X}\mathbf{X}^\mathsf{T} + \mathbf{I}\sigma^2)$.




(Bishop, 2006)

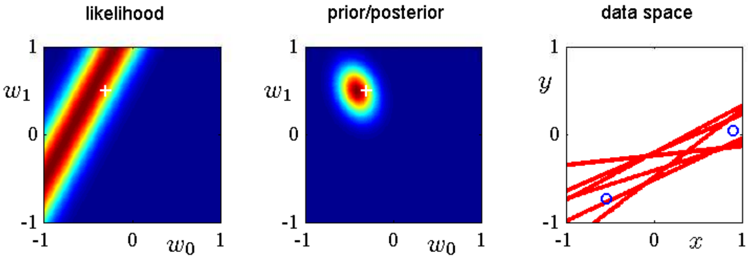
# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}\sigma^{-2} + \mathbf{I}$.

Marginal Likelihood: Given by $\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{X}\mathbf{X}^\mathsf{T} + \mathbf{I}\sigma^2)$.




(Bishop, 2006)

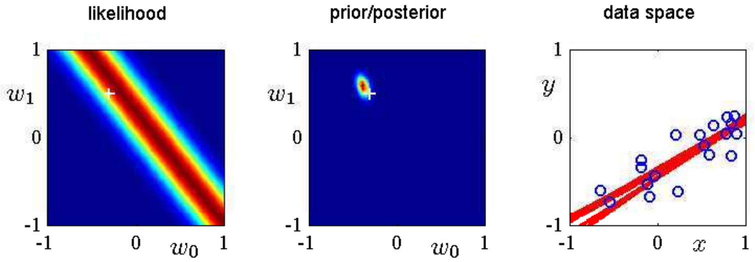
# The Standard Linear Model

Prior: We consider an **isometric Gaussian prior** $\mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{I})$.

Likelihood: Defined by the model as $\mathcal{N}(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I})$.

Posterior: Given by $\mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})$ with $\mathbf{A} = \mathbf{X}^\mathsf{T}\mathbf{X}\sigma^{-2} + \mathbf{I}$.

Marginal Likelihood: Given by $\mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{X}\mathbf{X}^\mathsf{T} + \mathbf{I}\sigma^2)$.




(Bishop, 2006)

# The Standard Linear Model

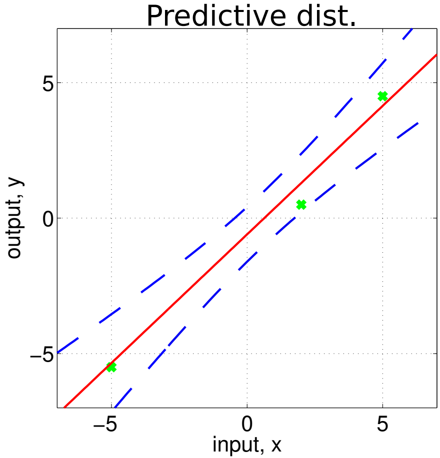
The predictive distribution is obtained by marginalizing $\mathbf{w}$:

$$p(y_\star|\mathbf{x}_\star) = \int p(y_\star|\mathbf{x}_\star, \mathbf{w})p(\mathbf{w}|\mathbf{X}, \mathbf{y})d\mathbf{w} = \mathcal{N}(y_\star|\sigma^{-2}\mathbf{x}_\star^\mathsf{T}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{x}_\star^\mathsf{T}\mathbf{A}^{-1}\mathbf{x}_\star + \sigma^2)$$

# The Standard Linear Model

The predictive distribution is obtained by marginalizing $\mathbf{w}$:

$$p(y_\star|\mathbf{x}_\star) = \int p(y_\star|\mathbf{x}_\star, \mathbf{w})p(\mathbf{w}|\mathbf{X}, \mathbf{y})d\mathbf{w} = \mathcal{N}(y_\star|\sigma^{-2}\mathbf{x}_\star^\mathsf{T}\mathbf{A}^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}, \mathbf{x}_\star^\mathsf{T}\mathbf{A}^{-1}\mathbf{x}_\star + \sigma^2)$$




(Rasmussen & Williams, 2006)

# Non-Linear Regression

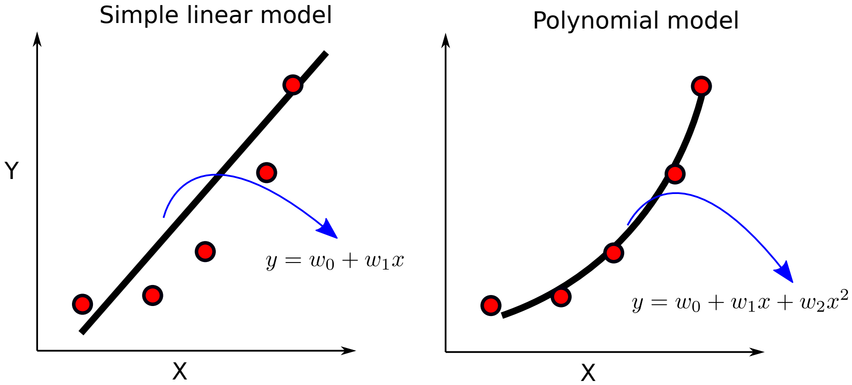
Non-linear problems can be addressed by performing feature expansions:

$$\phi(x) = (1, x, x^2, x^3, \ldots)^\mathsf{T}$$

# Non-Linear Regression

Non-linear problems can be addressed by performing feature expansions:

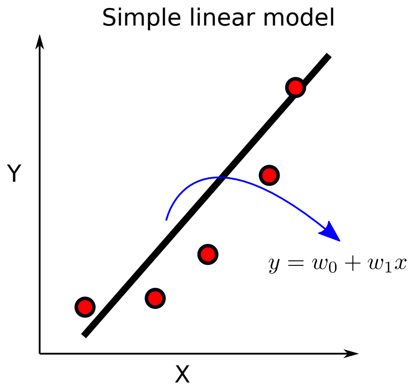
$$\phi(x) = (1, x, x^2, x^3, \ldots)^\mathsf{T}$$

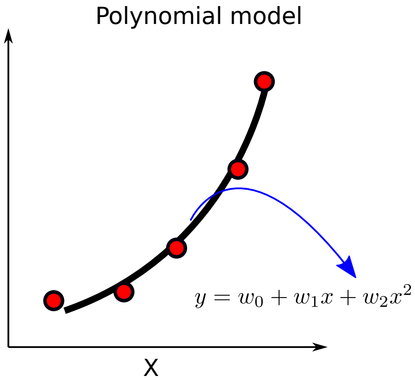


# Non-Linear Regression

Non-linear problems can be addressed by performing feature expansions:

$$\phi(x) = (1, x, x^2, x^3, \dots)^\mathsf{T}$$




**Any other non-linear feature expansion is possible!**

# Non-Linear Regression

Consider working with $\phi(\mathbf{x})$ instead of $\mathbf{x}$. The model is:

$$y = f(\mathbf{x}) + \epsilon = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}) + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,.$$

# Non-Linear Regression

Consider working with $\phi(\mathbf{x})$ instead of $\mathbf{x}$. The model is:

$$y = f(\mathbf{x}) + \epsilon = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}) + \epsilon \qquad\qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,.$$

The posterior and predictive distribution are:

$$\begin{aligned} p(\mathbf{w}|\mathbf{X}, \mathbf{y}) &= \mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})\,, \\ p(y_\star|\mathbf{X}, \mathbf{x}_\star) &= \mathcal{N}(y_\star|\sigma^{-2}\phi(\mathbf{x}_\star)^\mathsf{T}\mathbf{A}^{-1}\mathbf{\Phi}^\mathsf{T}\mathbf{y}), \phi(\mathbf{x}_\star)^\mathsf{T}\mathbf{A}^{-1}\phi(\mathbf{x}_\star) + \sigma^2), \end{aligned}$$

where $\mathbf{\Phi} = \phi(\mathbf{X})$ and $\mathbf{A} = \mathbf{\Phi}^\mathsf{T}\mathbf{\Phi}\sigma^{-2} + \mathbf{I}$.

# Non-Linear Regression

Consider working with $\phi(\mathbf{x})$ instead of $\mathbf{x}$. The model is:

$$y = f(\mathbf{x}) + \epsilon = \mathbf{w}^\mathsf{T}\phi(\mathbf{x}) + \epsilon \qquad\qquad \epsilon \sim \mathcal{N}(0, \sigma^2)\,.$$

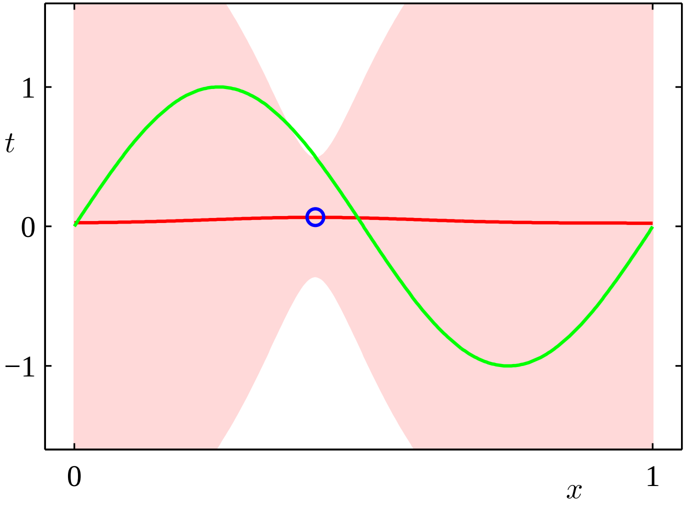
The posterior and predictive distribution are:

$$\begin{aligned} p(\mathbf{w}|\mathbf{X},\mathbf{y}) &= \mathcal{N}(\mathbf{w}|\sigma^{-2}\mathbf{A}^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{y}, \mathbf{A}^{-1})\,, \\ p(y_\star|\mathbf{X},\mathbf{x}_\star) &= \mathcal{N}(y_\star|\sigma^{-2}\phi(\mathbf{x}_\star)^\mathsf{T}\mathbf{A}^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{y}), \phi(\mathbf{x}_\star)^\mathsf{T}\mathbf{A}^{-1}\phi(\mathbf{x}_\star) + \sigma^2), \end{aligned}$$

where $\boldsymbol{\Phi} = \phi(\mathbf{X})$ and $\mathbf{A} = \boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi}\sigma^{-2} + \mathbf{I}$.

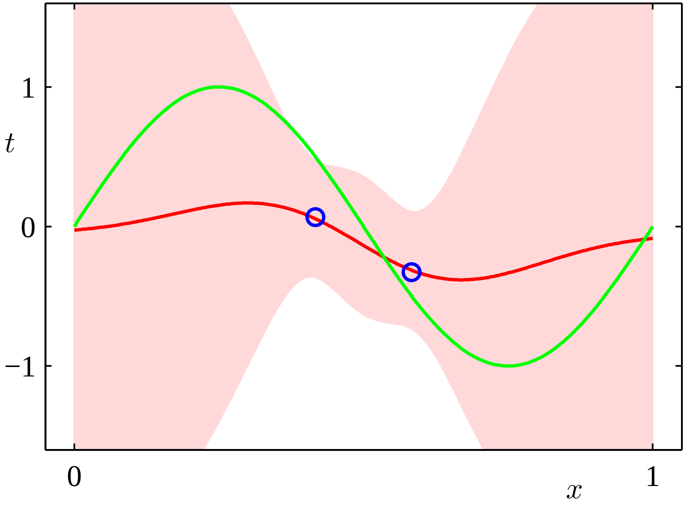
**All computations are tractable and result in Gaussian distributions!**

# Non-Linear Regression



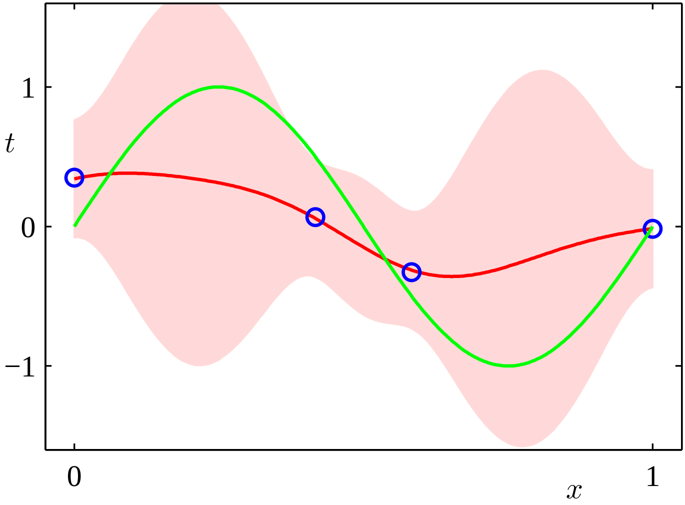

(Bishop,2006)

# Non-Linear Regression



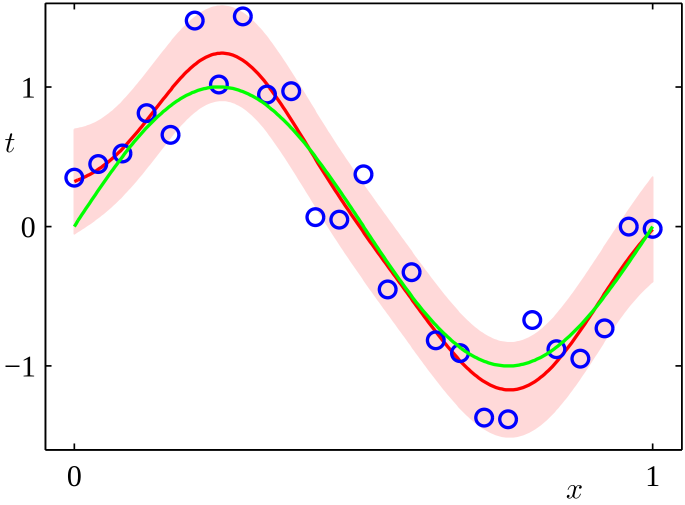

(Bishop,2006)

# Non-Linear Regression



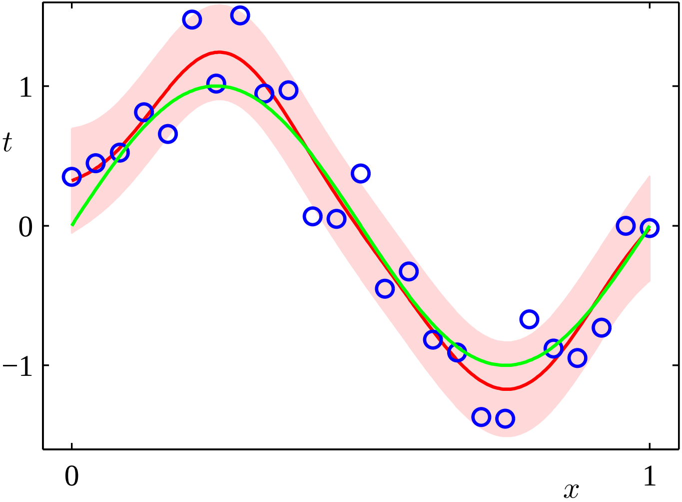

(Bishop,2006)

# Non-Linear Regression




(Bishop,2006)

# Non-Linear Regression




**The predictive distribution tells us what our model does not know!**

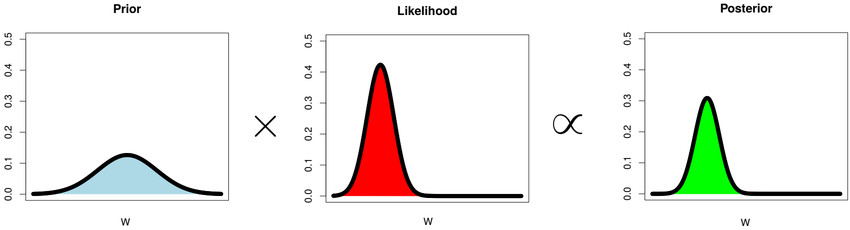
(Bishop,2006)

# Function Space View

**An equivalent way of reaching identical results is possible by considering inference in function space.**

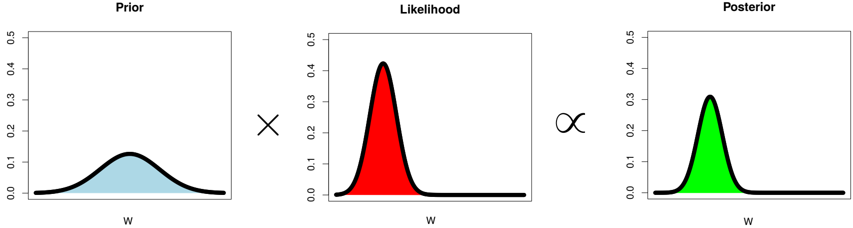
# Function Space View

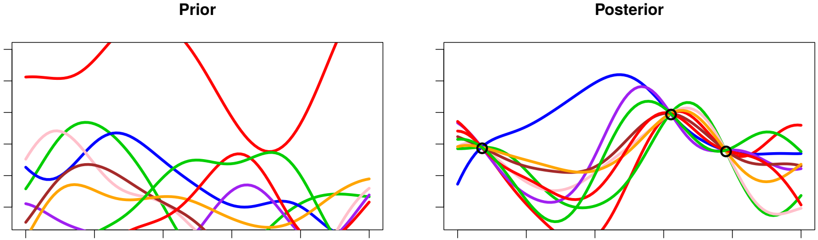
**An equivalent way of reaching identical results is possible by considering inference in function space.**



# Function Space View

**An equivalent way of reaching identical results is possible by considering inference in function space.**



# Gaussian Processes

The previous random functions are samples from a Gaussian process.

# Gaussian Processes

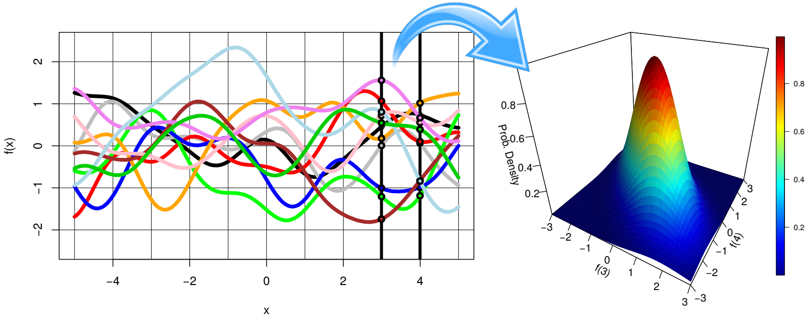
The previous random functions are samples from a Gaussian process.

**Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^N$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an $N$-dimensional Gaussian distribution.**

# Gaussian Processes

The previous random functions are samples from a Gaussian process.

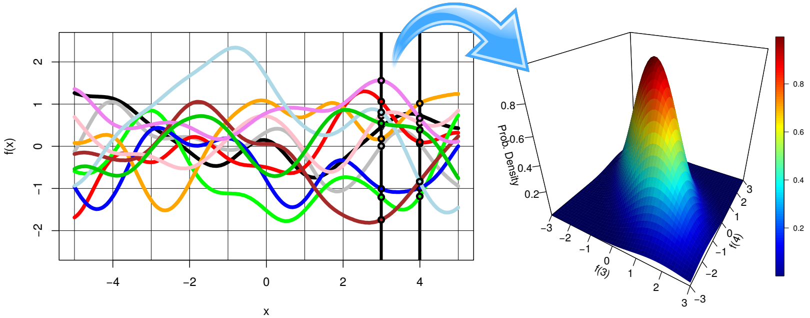
**Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^N$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ follows an $N$-dimensional Gaussian distribution.**



# Gaussian Processes

The previous random functions are samples from a Gaussian process.

**Distribution over functions $f(\cdot)$ so that for any finite $\{\mathbf{x}_i\}_{i=1}^{N}$, $(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^{\mathsf{T}}$ follows an $N$-dimensional Gaussian distribution.**




Straight-forward for the prior and posterior. Since the they are Gaussian for $\mathbf{w}$, $f$ is the sum of Gaussian random variables and is also Gaussian!

# Advantages of the Function Space Inference

1. **We can compute the predictive distribution without explicitly computing the posterior for w!**

# Advantages of the Function Space Inference

1. **We can compute the predictive distribution without explicitly computing the posterior for w!**

2. **Due to Gaussian form of the process values, there are many closed-form solutions for questions about the data.**

# Advantages of the Function Space Inference

1. **We can compute the predictive distribution without explicitly computing the posterior for w!**

2. **Due to Gaussian form of the process values, there are many closed-form solutions for questions about the data.**

3. **We need not compute $\phi(\mathbf{x})$, only $\phi(\mathbf{x}_i)^\mathsf{T}\phi(\mathbf{x}_j)$. This allows to use feature expansions of infinite size!**

# Advantages of the Function Space Inference

1. We can compute the predictive distribution without explicitly computing the posterior for $\mathbf{w}$!

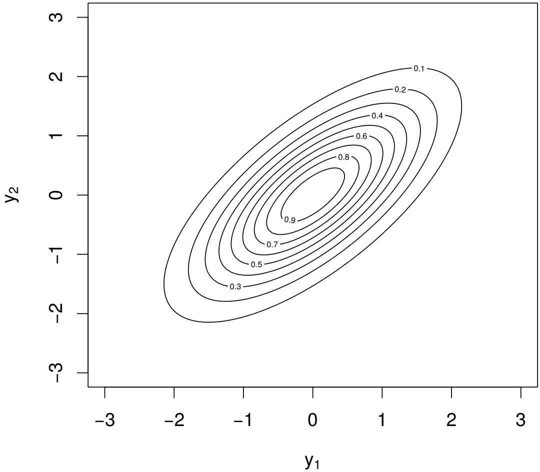
2. Due to Gaussian form of the process values, there are many closed-form solutions for questions about the data.

3. We need not compute $\phi(\mathbf{x})$, only $\phi(\mathbf{x}_i)^{\mathsf{T}}\phi(\mathbf{x}_j)$. This allows to use feature expansions of infinite size!

4. This results in a non-parametric model that becomes more flexible as more data is observed!

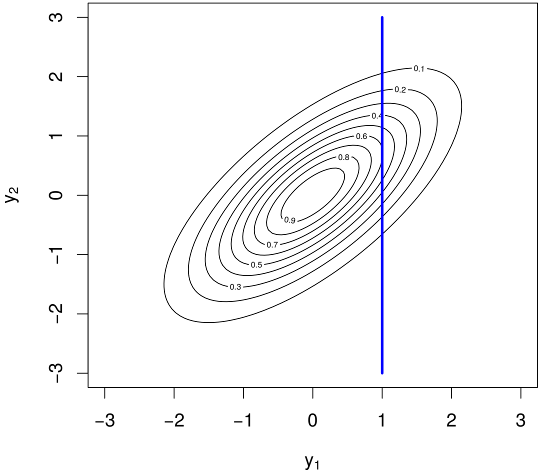
# Gaussian Distribution

$$p(\mathbf{y}|\boldsymbol{\Sigma}) \propto \exp\left\{-0.5\mathbf{y}^{\top}\boldsymbol{\Sigma}^{-1}\mathbf{y}\right\} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1.0 & 0.7 \\ 0.7 & 1.0 \end{bmatrix}.$$



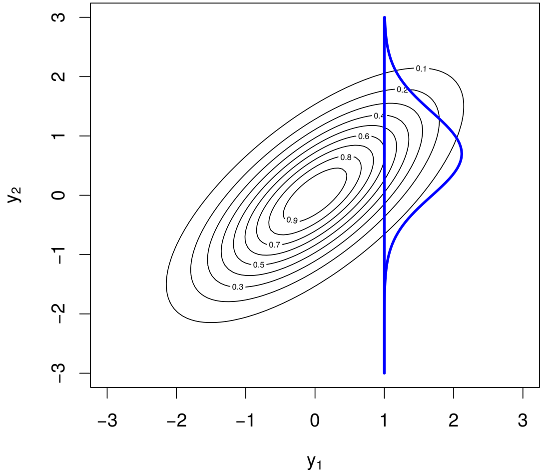
# Gaussian Distribution

$$p(\mathbf{y}|\boldsymbol{\Sigma}) \propto \exp\left\{-0.5\mathbf{y}^\top\boldsymbol{\Sigma}^{-1}\mathbf{y}\right\} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1.0 & 0.7 \\ 0.7 & 1.0 \end{bmatrix}.$$



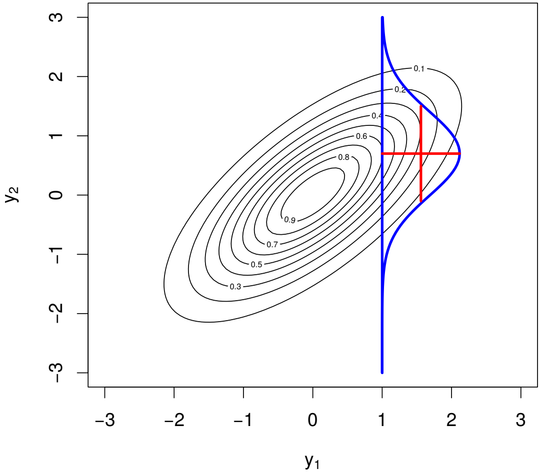
# Gaussian Distribution

$$p(y_2|y_1, \boldsymbol{\Sigma}) \propto \exp\left\{-0.5(y_2 - \mu_\star)\boldsymbol{\Sigma}_\star^{-1}(y_2 - \mu_\star)\right\} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1.0 & 0.7 \\ 0.7 & 1.0 \end{bmatrix}.$$



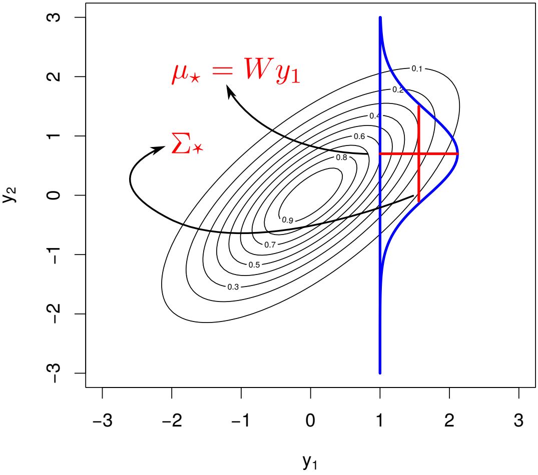
# Gaussian Distribution

$$p(y_2|y_1, \boldsymbol{\Sigma}) \propto \exp\left\{-0.5(y_2 - \mu_\star)\boldsymbol{\Sigma}_\star^{-1}(y_2 - \mu_\star)\right\} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1.0 & 0.7 \\ 0.7 & 1.0 \end{bmatrix}.$$

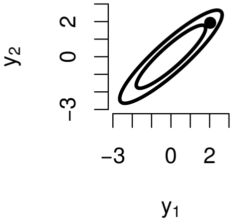


# Gaussian Distribution

$$p(y_2|y_1, \boldsymbol{\Sigma}) \propto \exp\left\{-0.5(y_2 - \mu_\star)\Sigma_\star^{-1}(y_2 - \mu_\star)\right\} \qquad \boldsymbol{\Sigma} = \begin{bmatrix} 1.0 & 0.7 \\ 0.7 & 1.0 \end{bmatrix}.$$

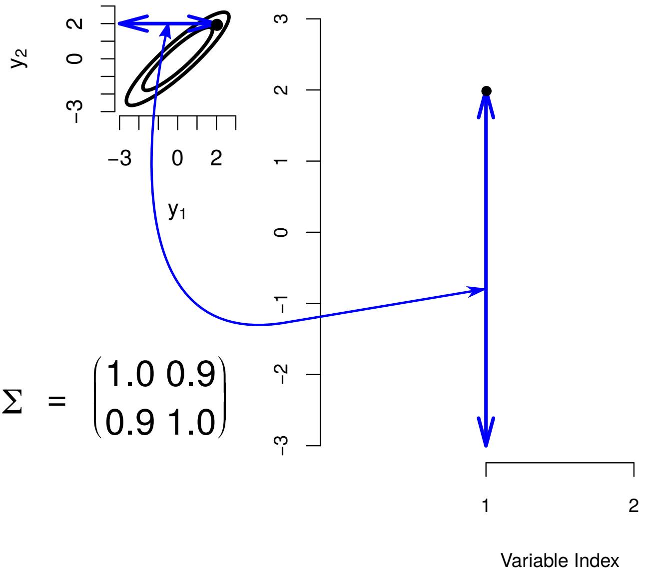


# Two Dimensional Example



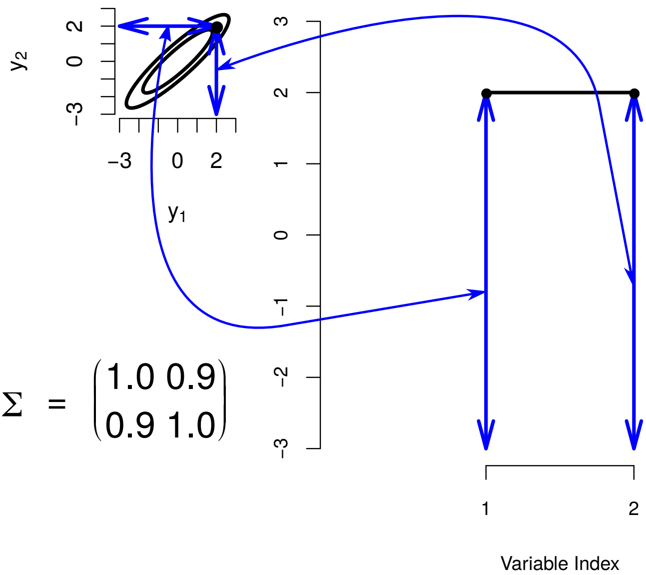

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$

# Two Dimensional Example



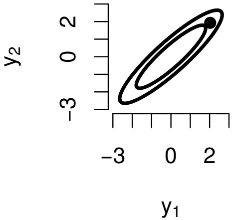

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$

# Two Dimensional Example

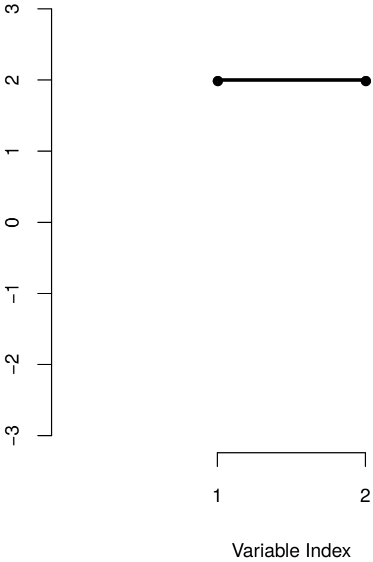



$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$

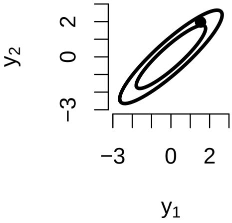
# Two Dimensional Example



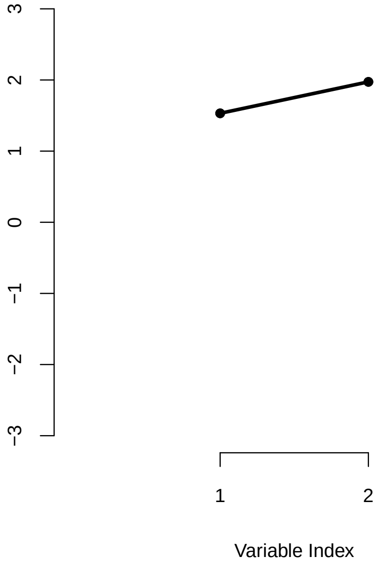

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$



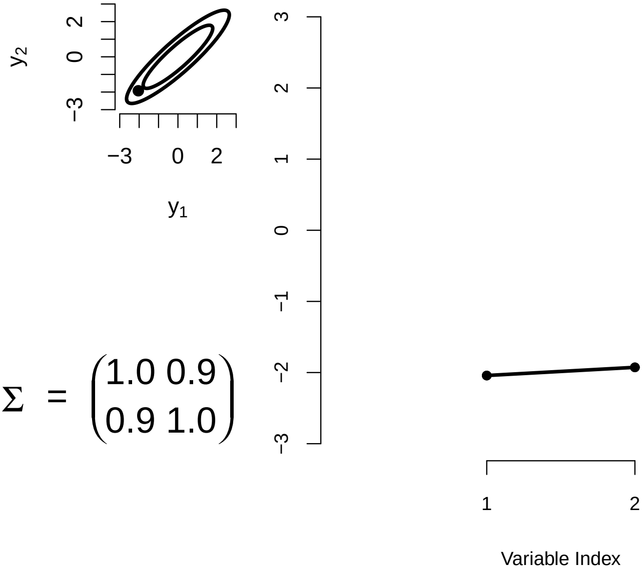
# Two Dimensional Example




$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$

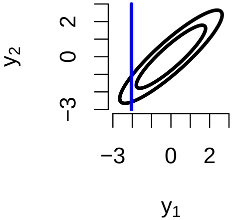


# Two Dimensional Example

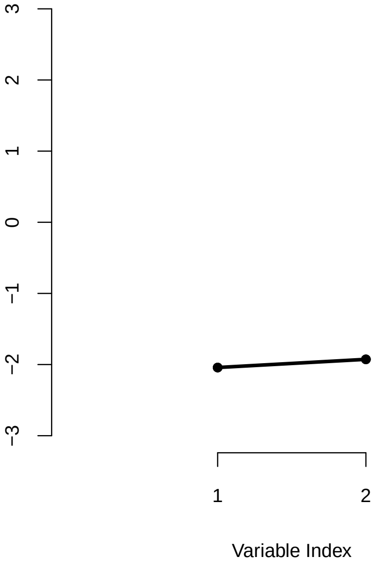



$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$

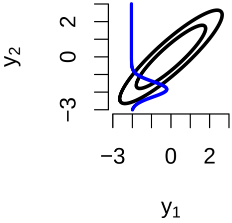
# Two Dimensional Example



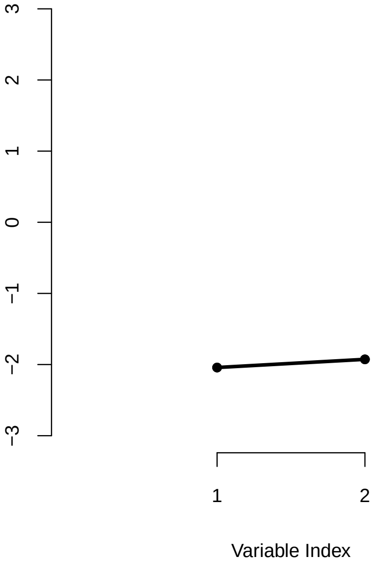

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$



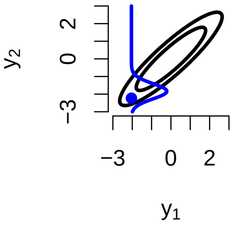
# Two Dimensional Example



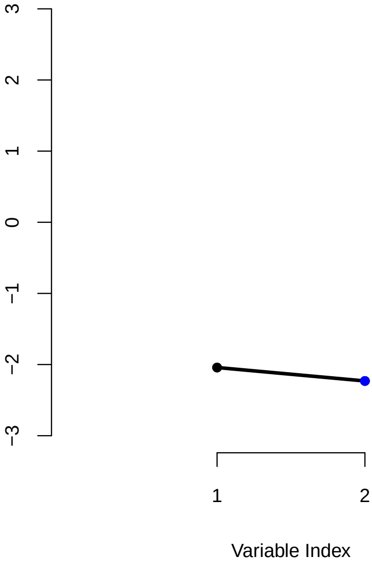

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$



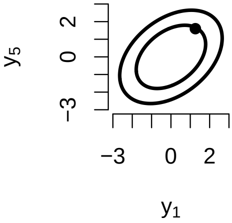
# Two Dimensional Example



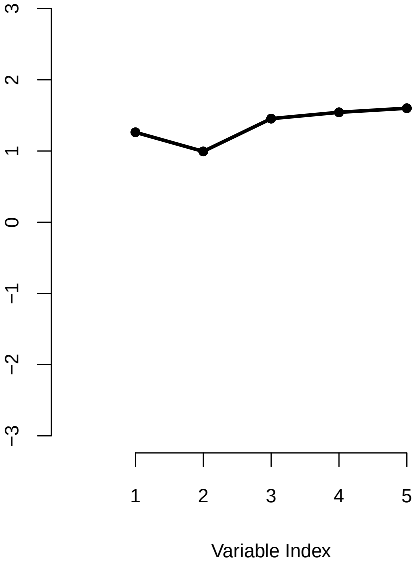

$$\Sigma = \begin{pmatrix} 1.0 & 0.9 \\ 0.9 & 1.0 \end{pmatrix}$$



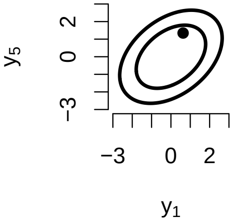
# Five Dimensional Example



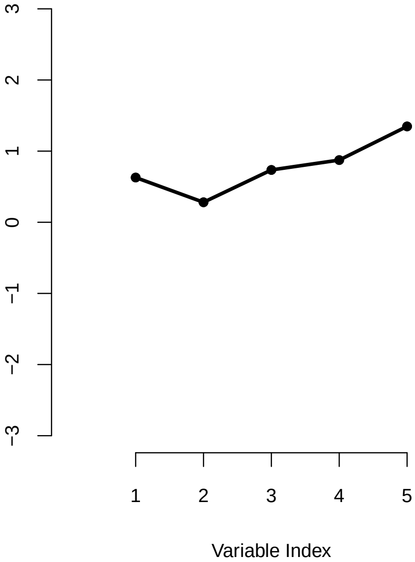

$$\Sigma = \begin{bmatrix} 1.0 & .9 & .8 & .6 & .4 \\ .9 & 1.0 & .9 & .8 & .6 \\ .8 & .9 & 1.0 & .9 & .8 \\ .6 & .8 & .9 & 1.0 & .9 \\ .4 & .6 & .8 & .9 & 1.0 \end{bmatrix}$$



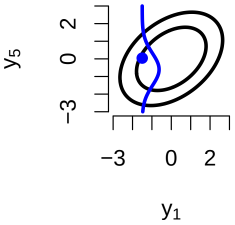
# Five Dimensional Example



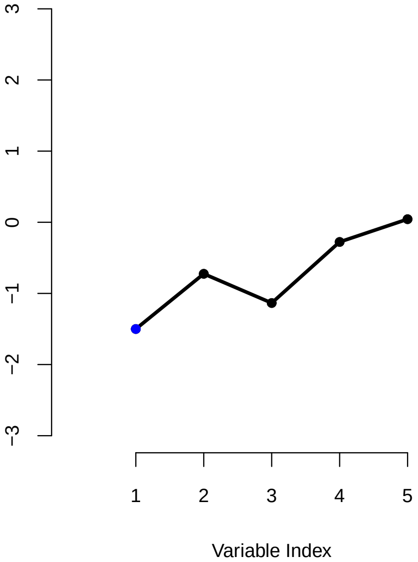

$$\Sigma = \begin{bmatrix} 1.0 & .9 & .8 & .6 & .4 \\ .9 & 1.0 & .9 & .8 & .6 \\ .8 & .9 & 1.0 & .9 & .8 \\ .6 & .8 & .9 & 1.0 & .9 \\ .4 & .6 & .8 & .9 & 1.0 \end{bmatrix}$$



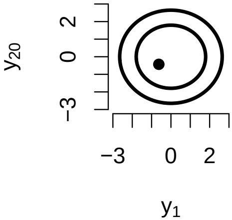
# Five Dimensional Example

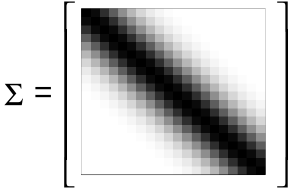


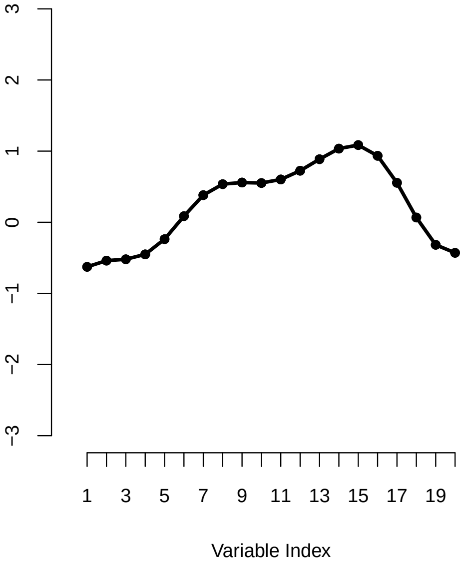

$$\Sigma = \begin{bmatrix} 1.0 & .9 & .8 & .6 & .4 \\ .9 & 1.0 & .9 & .8 & .6 \\ .8 & .9 & 1.0 & .9 & .8 \\ .6 & .8 & .9 & 1.0 & .9 \\ .4 & .6 & .8 & .9 & 1.0 \end{bmatrix}$$



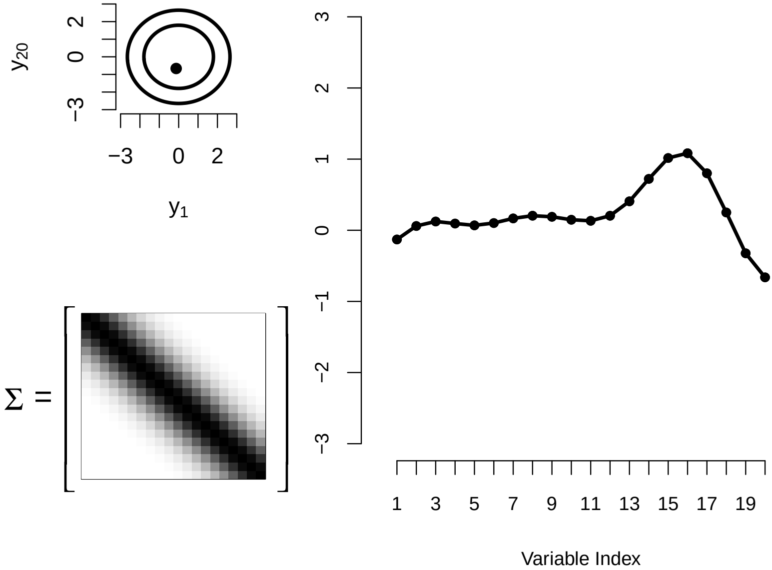
# Twenty Dimensional Example



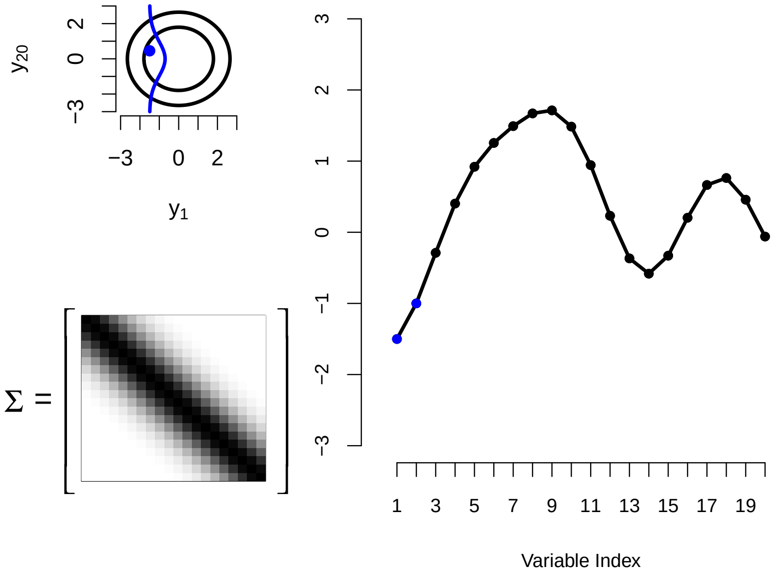



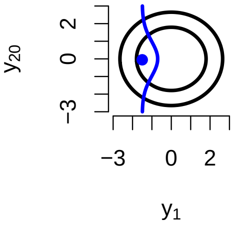
# Twenty Dimensional Example

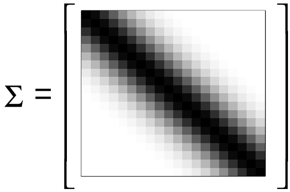


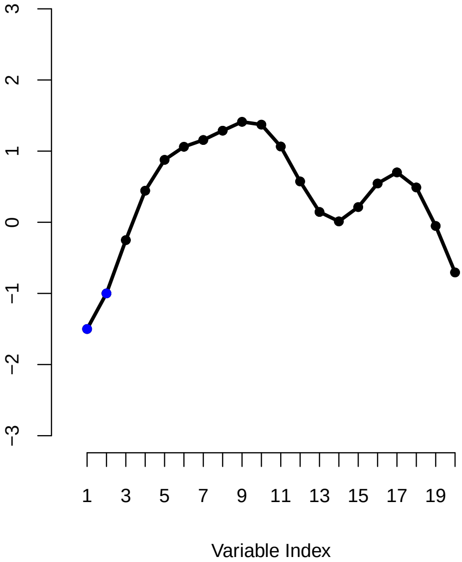
# Twenty Dimensional Example



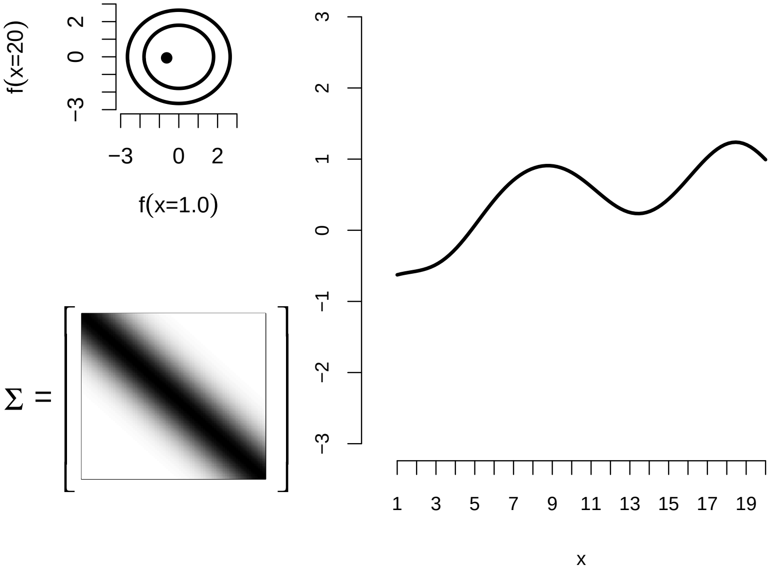
# Twenty Dimensional Example



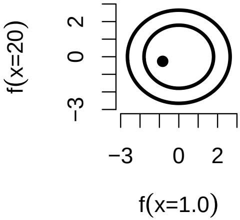



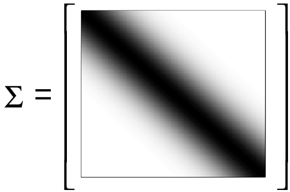



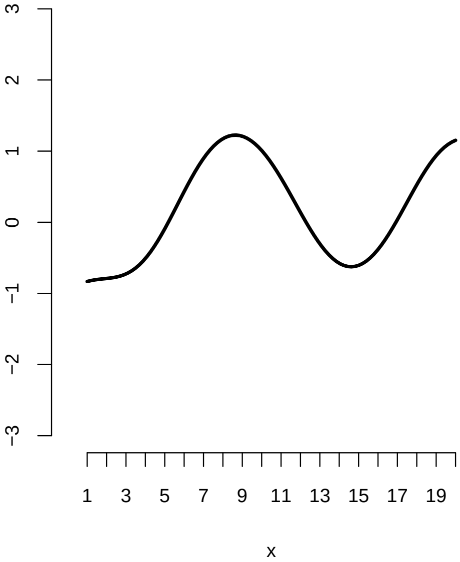
# Infinite Dimensional Example



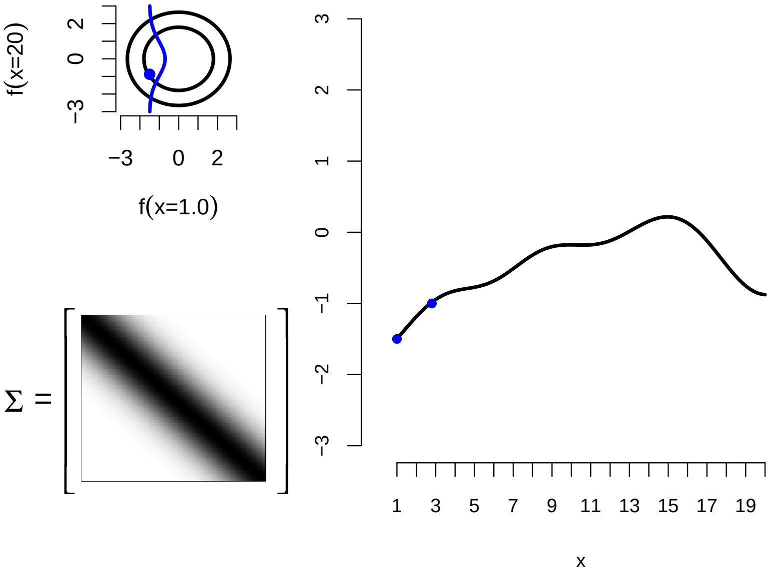
# Infinite Dimensional Example



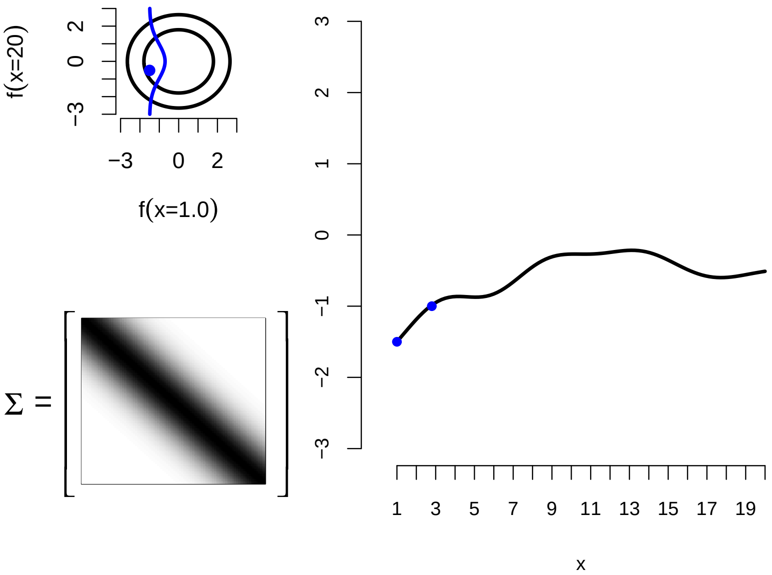



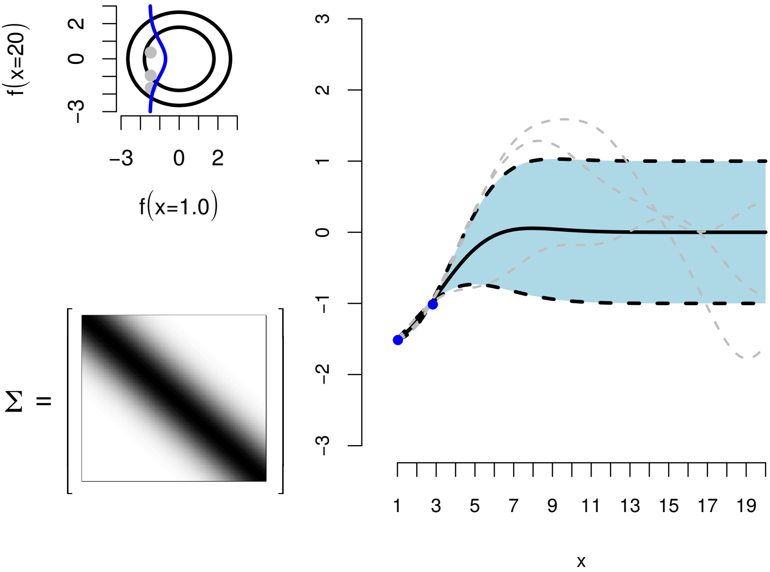
# Infinite Dimensional Example



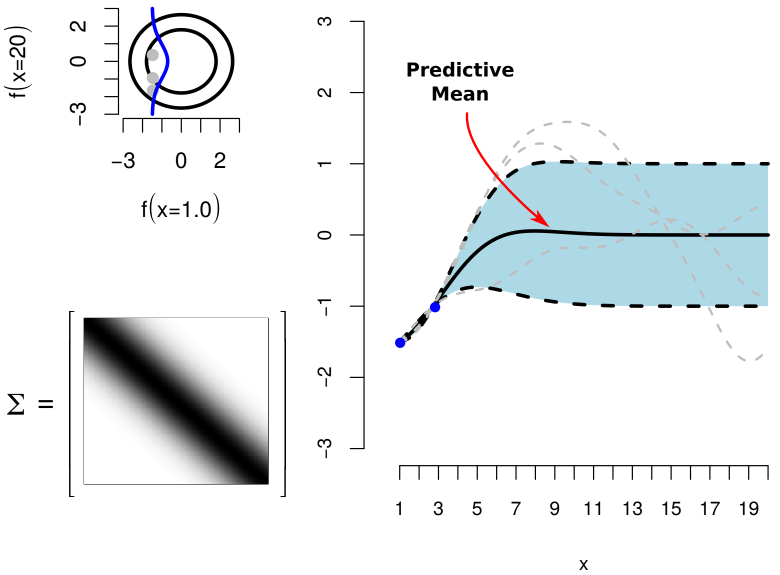
# Infinite Dimensional Example



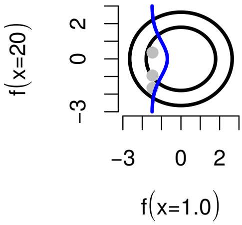
# Predictive Distribution

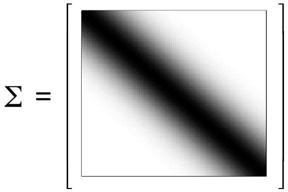


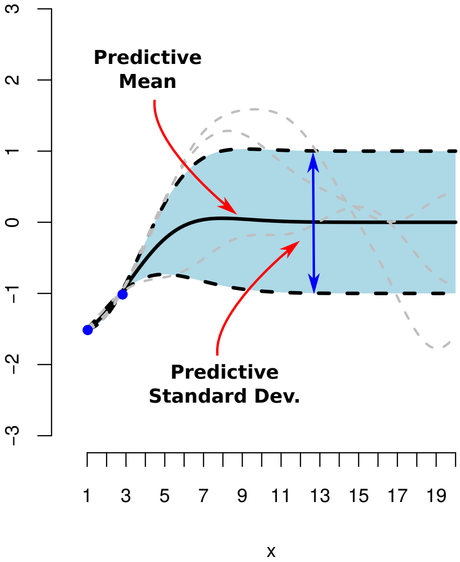
# Predictive Distribution



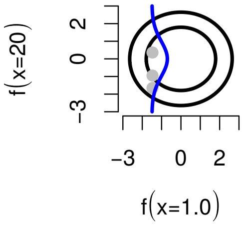
# Predictive Distribution

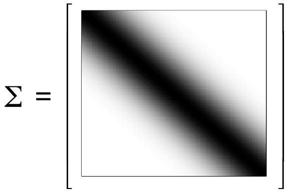


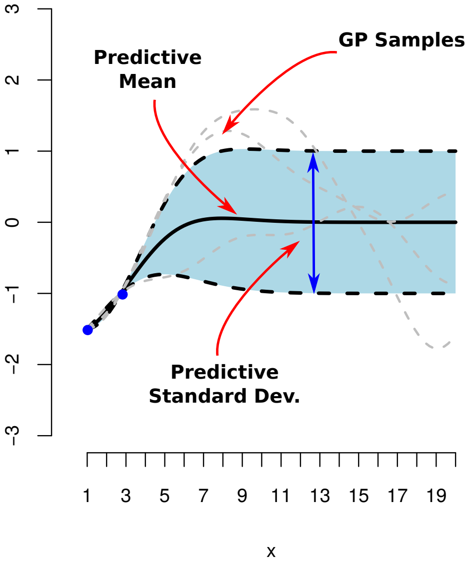



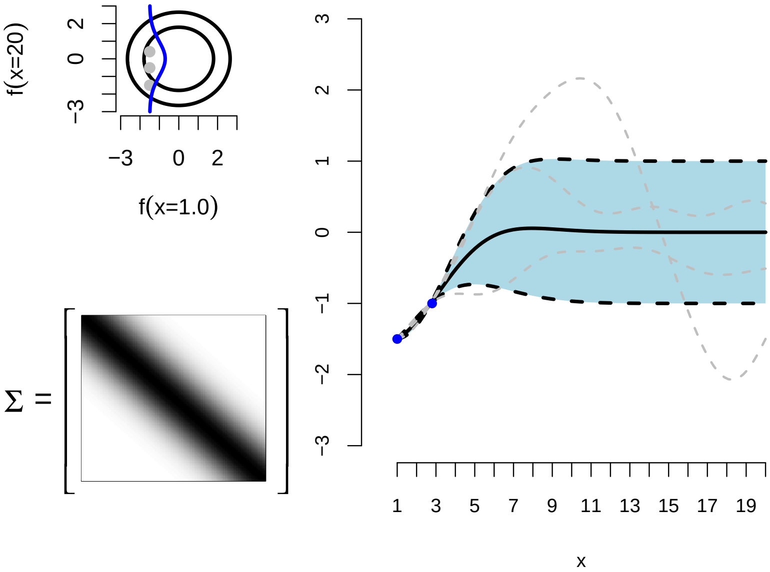
# Predictive Distribution



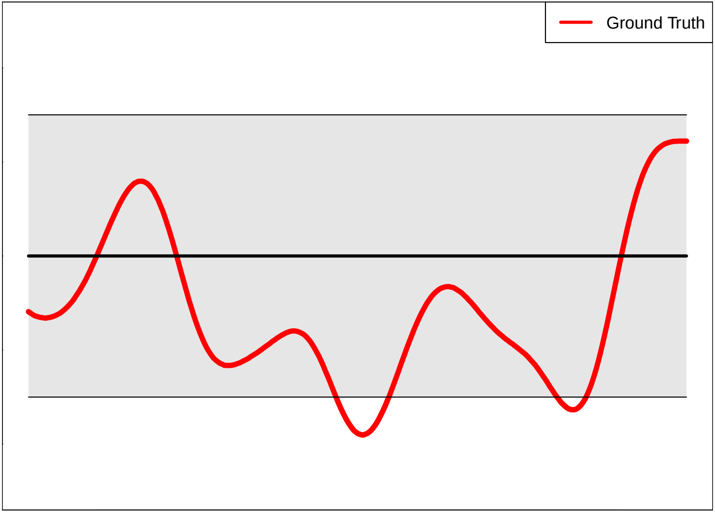



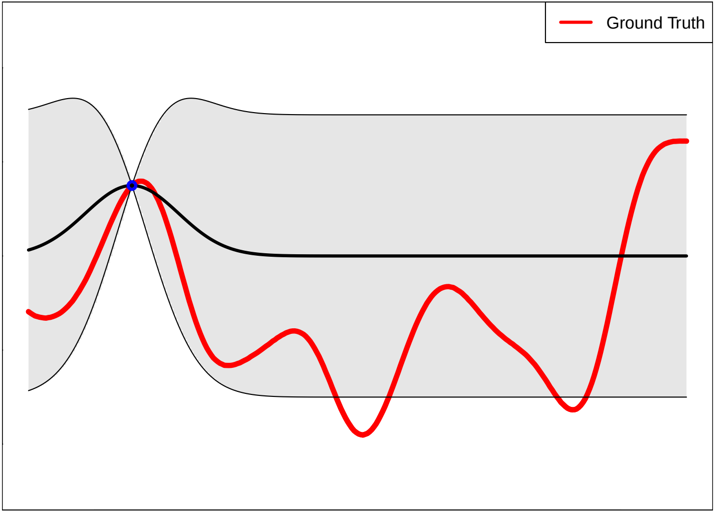
# Predictive Distribution



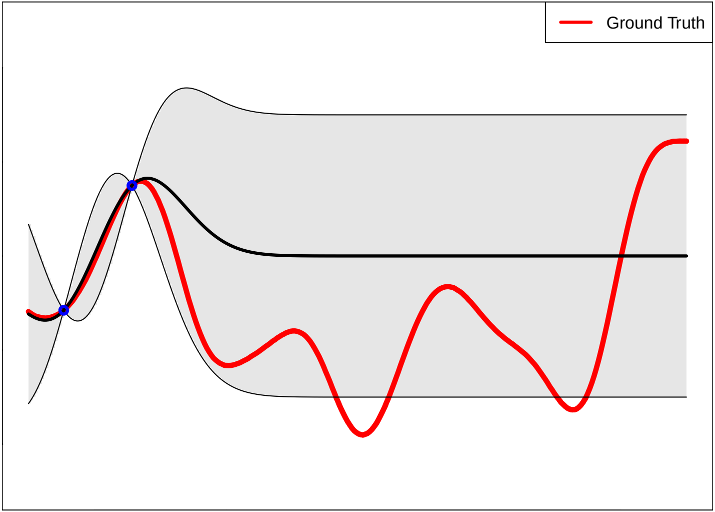
# Predictive Distribution



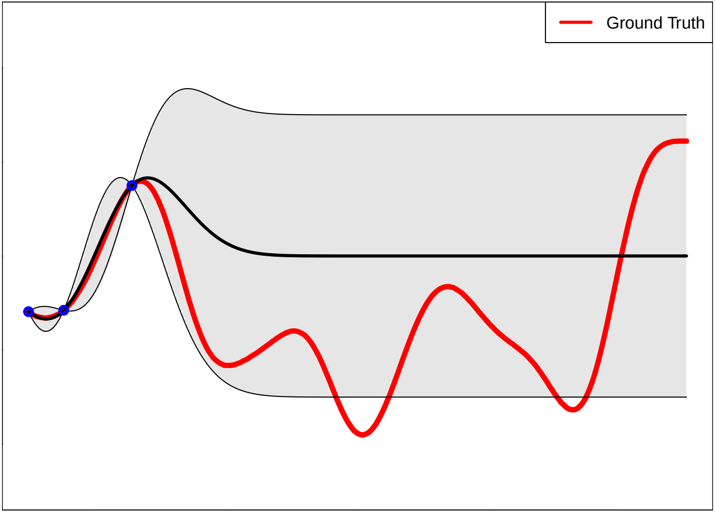
# Predictive Distribution



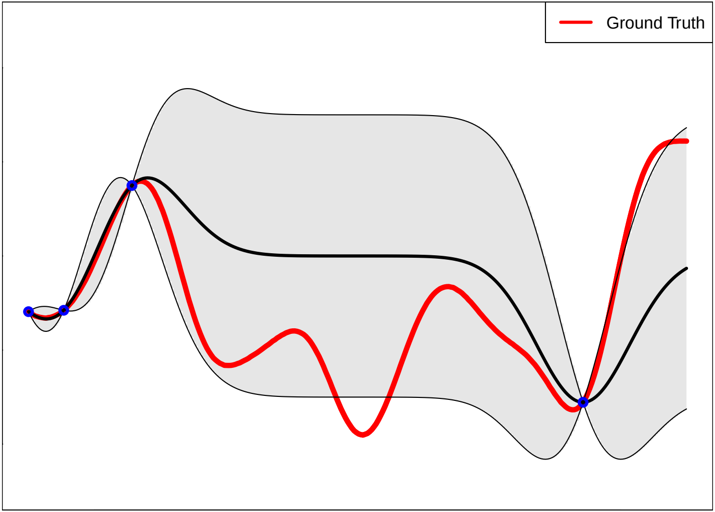
# Predictive Distribution



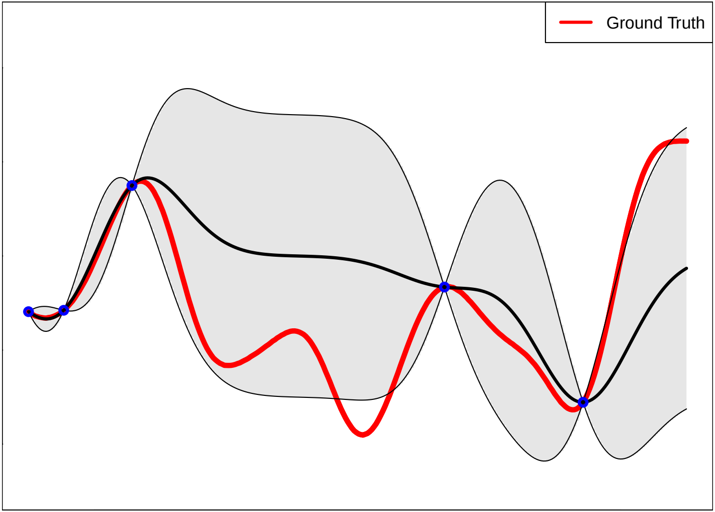
# Predictive Distribution



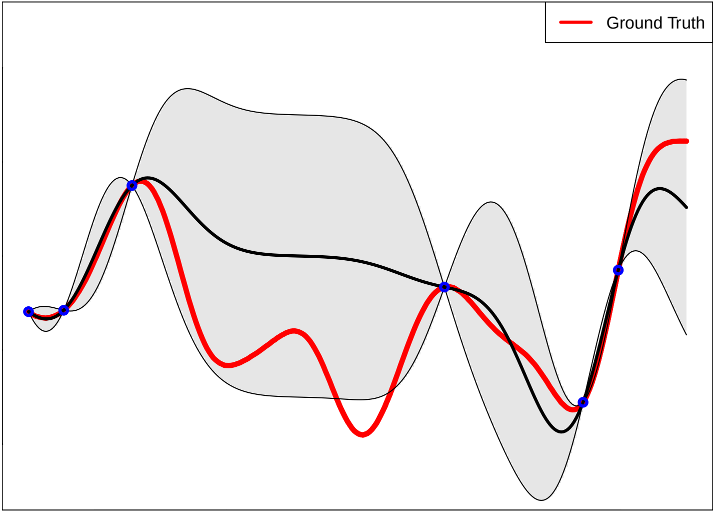
# Predictive Distribution



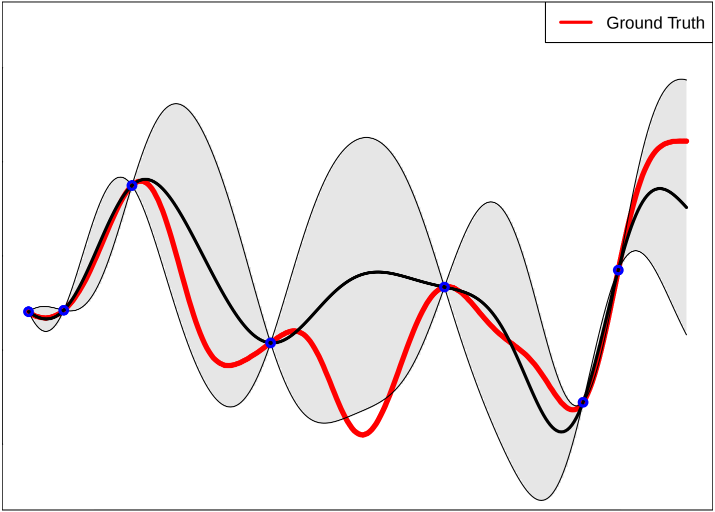
# Predictive Distribution



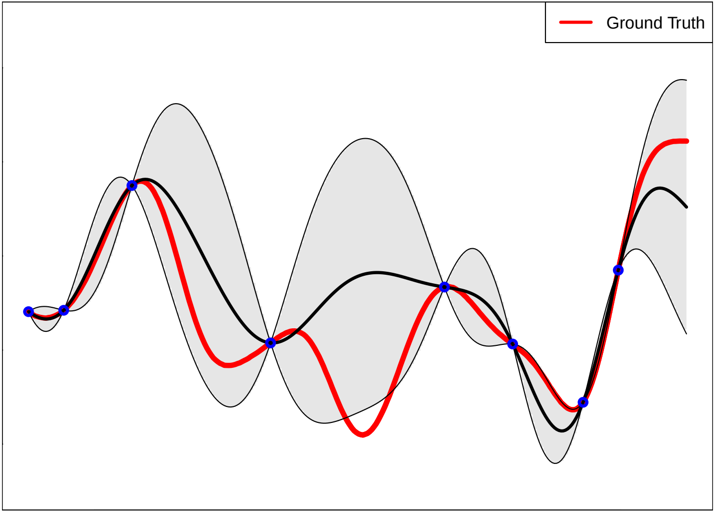
# Predictive Distribution



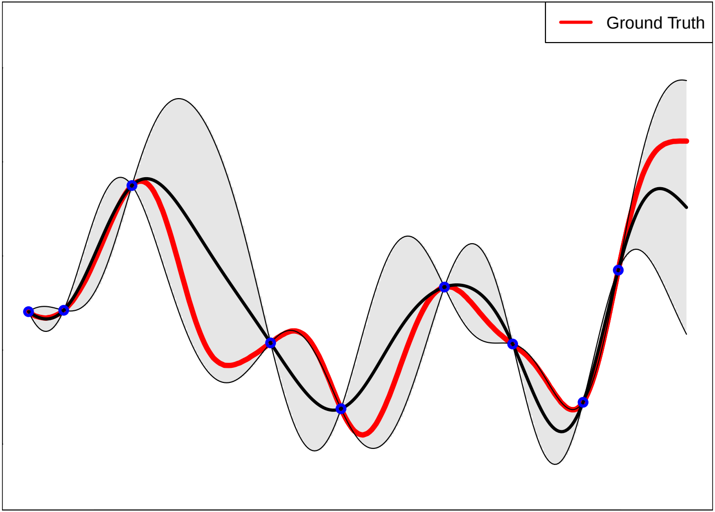
# Predictive Distribution



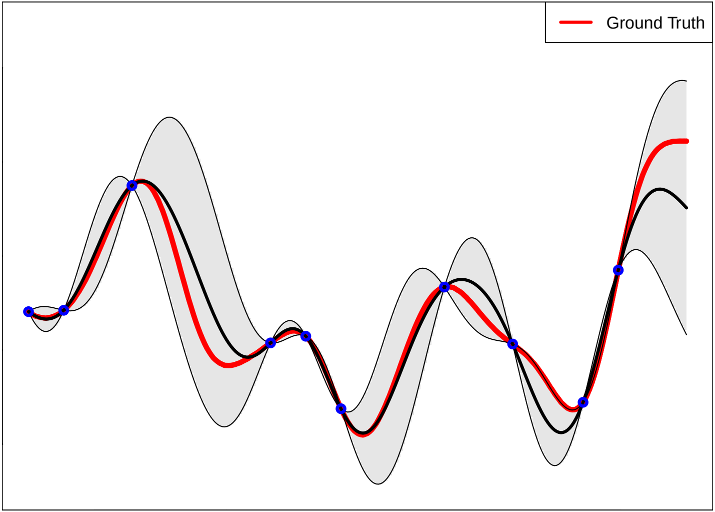
# Predictive Distribution



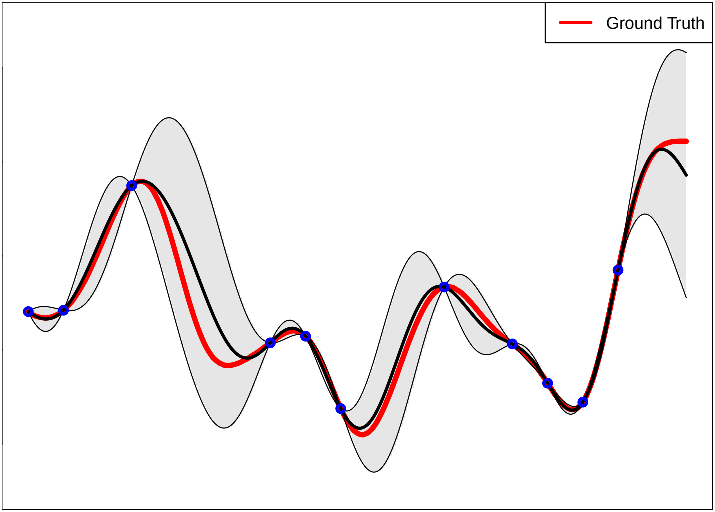
# Predictive Distribution



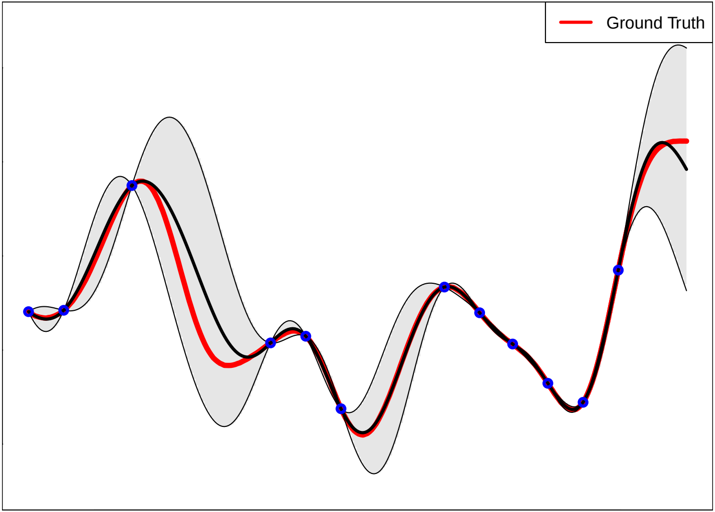
# Predictive Distribution



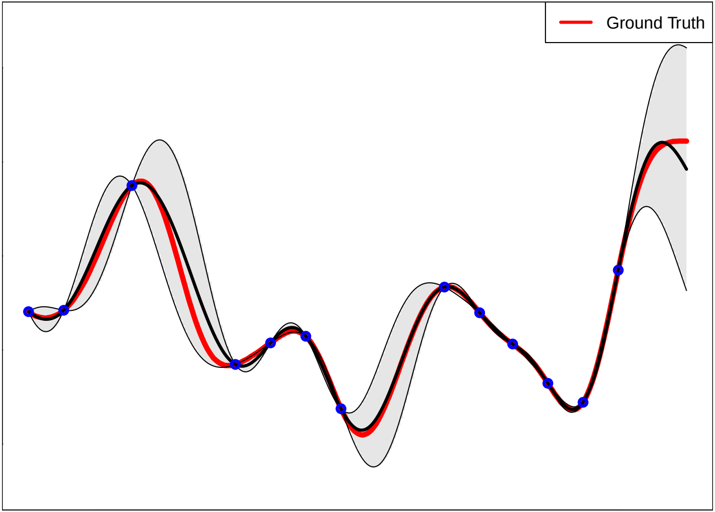
# Predictive Distribution



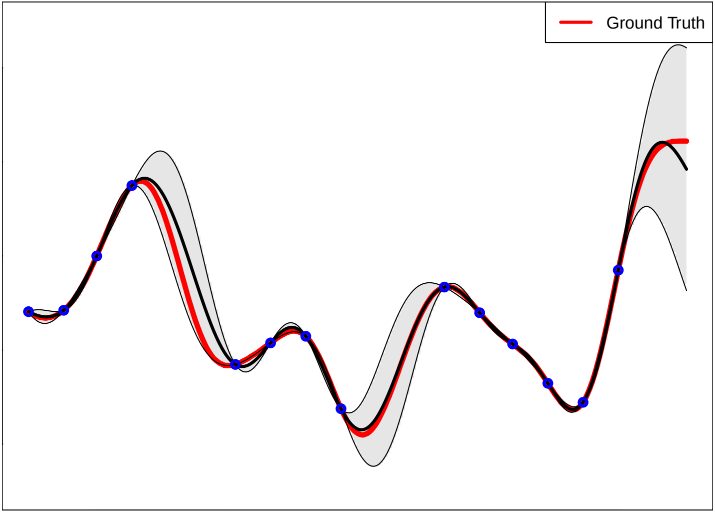
# Predictive Distribution



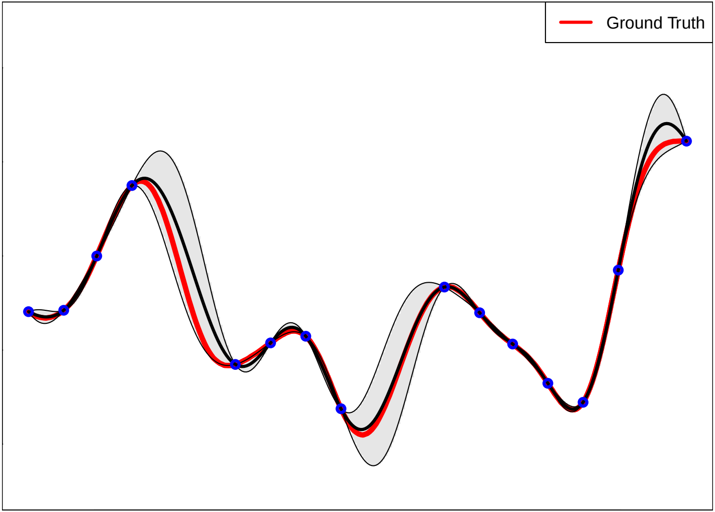
# Predictive Distribution



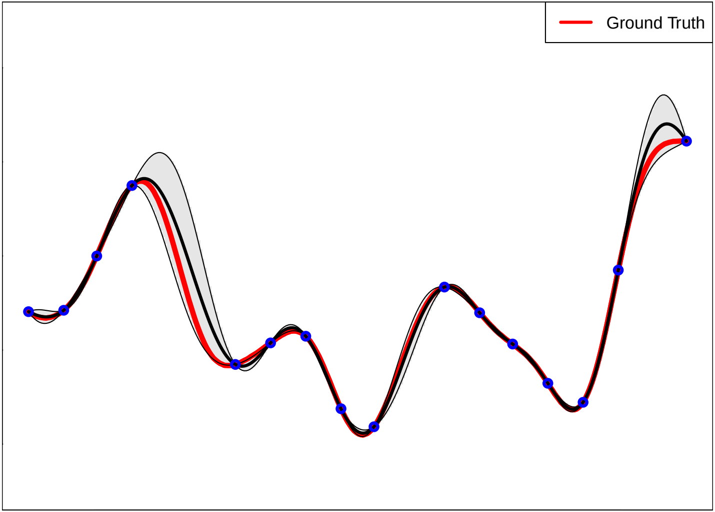
# Predictive Distribution



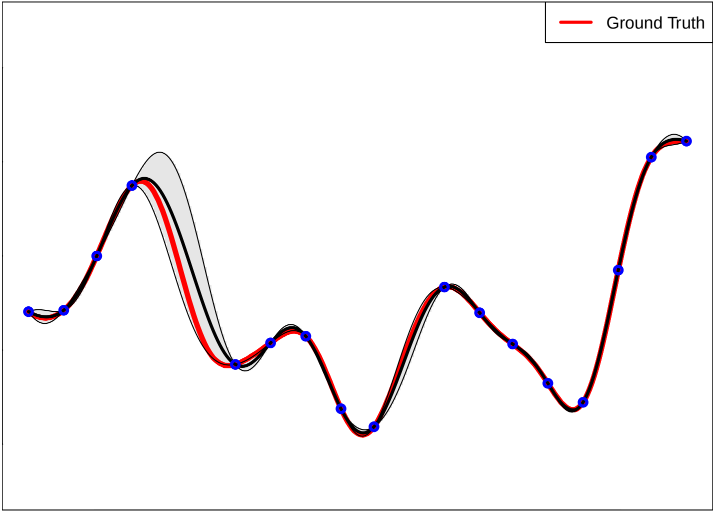
# Predictive Distribution



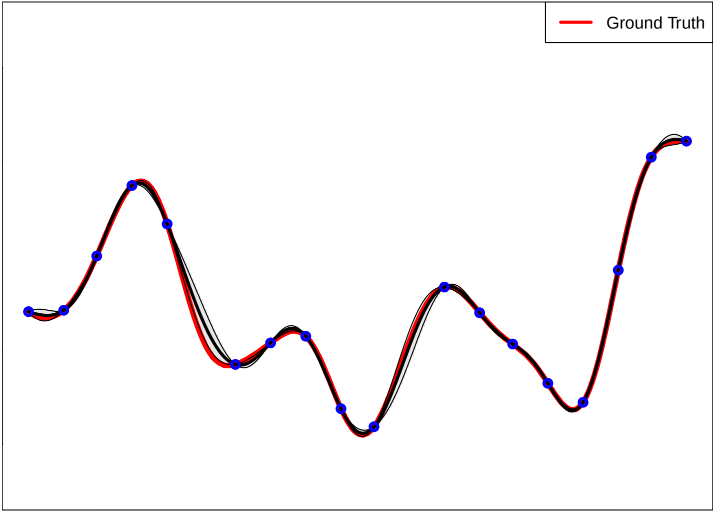
# Predictive Distribution



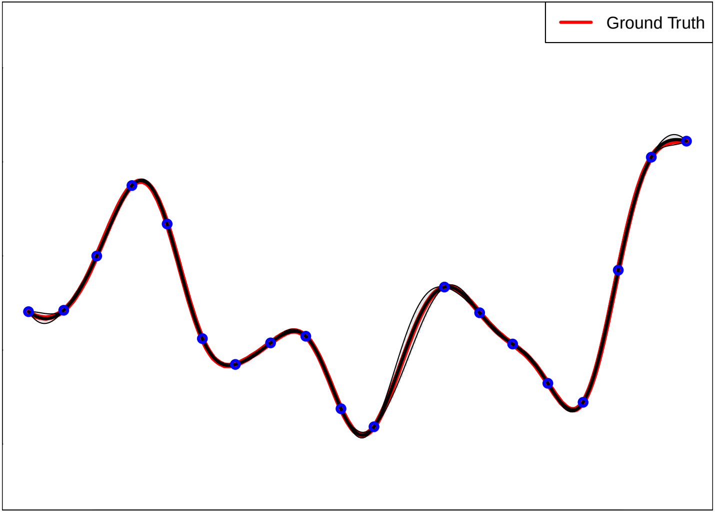
# Predictive Distribution



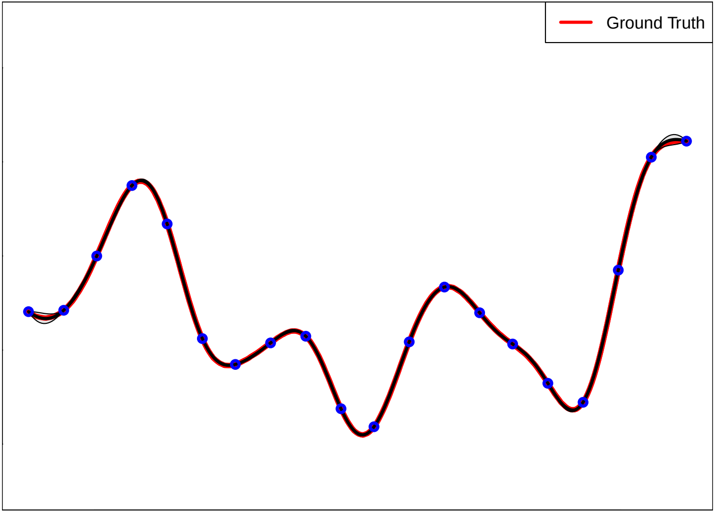
# Predictive Distribution



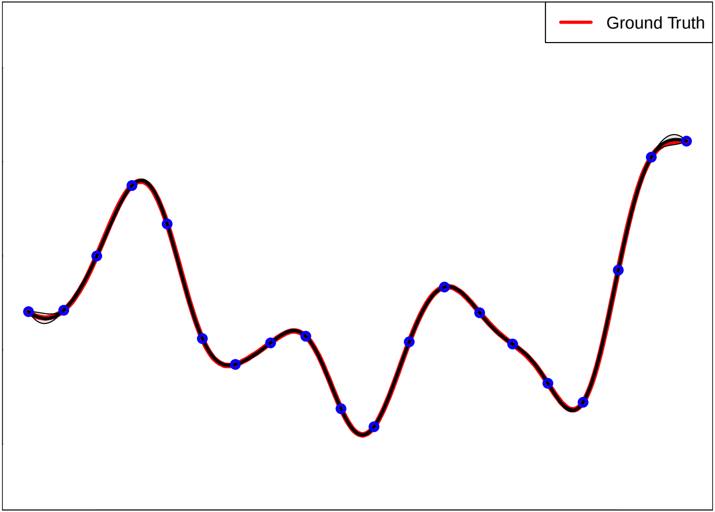
# Predictive Distribution



# Predictive Distribution



# Predictive Distribution




**The model becomes more flexible as we observe more data!**

# Summary so Far...

- A GP is *like* a Gaussian distribution with an **infinitely long mean vector** and an $\infty \times \infty$ **covariance matrix**.

# Summary so Far...

- A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
- The covariance matrix often enforces that function values corresponding to near-by points take **similar values**.

# Summary so Far...

- A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
- The covariance matrix often enforces that function values corresponding to near-by points take **similar values**.
- Due to the Gaussian distribution of finite function values, there are many **closed form expressions** like the predictive distribution.

# Summary so Far...

- A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
- The covariance matrix often enforces that function values corresponding to near-by points take **similar values**.
- Due to the Gaussian distribution of finite function values, there are many **closed form expressions** like the predictive distribution.
- GPs are **non-parametric models** and become more expressive the more data we have.

# Summary so Far...

- A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
- The covariance matrix often enforces that function values corresponding to near-by points take **similar values**.
- Due to the Gaussian distribution of finite function values, there are many **closed form expressions** like the predictive distribution.
- GPs are **non-parametric models** and become more expressive the more data we have.
- The predictive uncertainty is high in regions with **no data**!

# Definition

**A Gaussian process is a collection of random variables, any finite number of which have a Gaussian distribution.**

# Definition

**A Gaussian process is a collection of random variables, any finite number of which have a Gaussian distribution.**

A Gaussian distribution is fully specified by a mean vector, $\boldsymbol{\mu}$, and covariance matrix $\boldsymbol{\Sigma}$:

$$\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \text{indices} \quad i = 1, \ldots, N\,.$$

# Definition

**A Gaussian process is a collection of random variables, any finite number of which have a Gaussian distribution.**

A Gaussian distribution is fully specified by a mean vector, $\boldsymbol{\mu}$, and covariance matrix $\boldsymbol{\Sigma}$:

$$\mathbf{f} = (f_1, \ldots, f_N)^\mathsf{T} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \text{indices} \quad i = 1, \ldots, N\,.$$

A Gaussian process is fully specified by a mean function $m(\mathbf{x})$ and covariance function $C(\mathbf{x}, \mathbf{x}')$:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), C(\mathbf{x}, \mathbf{x}')), \quad \text{indices} \quad \mathbf{x}\,.$$

# GP Prior Mean

The GP prior mean $m(\cdot)$ can be specified by any function!

$$\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x}).$$

# GP Prior Mean

The GP prior mean $m(\cdot)$ can be specified by any function!

$$\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x}).$$

**It determines the global tendency of the latent function before observing the data. Often, simply set to zero.**

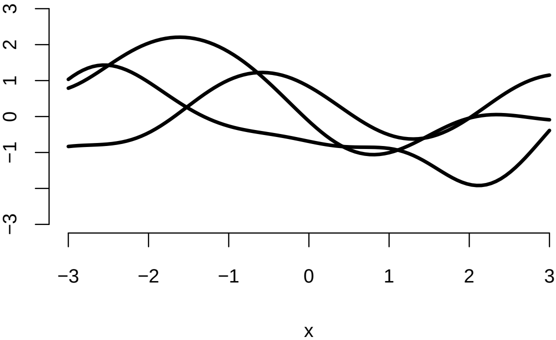
# GP Prior Mean

The GP prior mean $m(\cdot)$ can be specified by any function!

$$\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x}).$$

**It determines the global tendency of the latent function before observing the data. Often, simply set to zero.**

m(x)=0



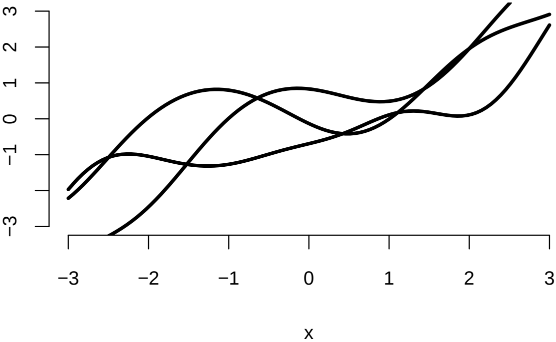
# GP Prior Mean

The GP prior mean $m(\cdot)$ can be specified by any function!

$$\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x}).$$

**It determines the global tendency of the latent function before observing the data. Often, simply set to zero.**

m(x)=x



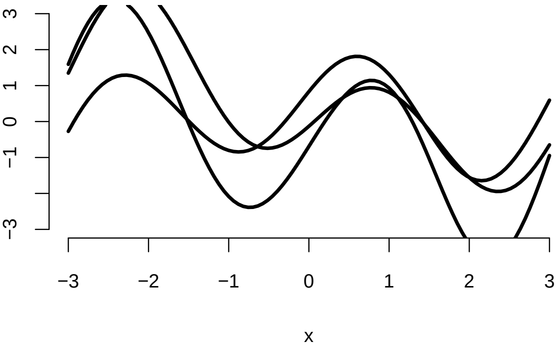
# GP Prior Mean

The GP prior mean $m(\cdot)$ can be specified by any function!

$$\mathbb{E}[f(\mathbf{x})] = m(\mathbf{x}).$$

**It determines the global tendency of the latent function before observing the data. Often, simply set to zero.**

m(x)=2sin(2x)



# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

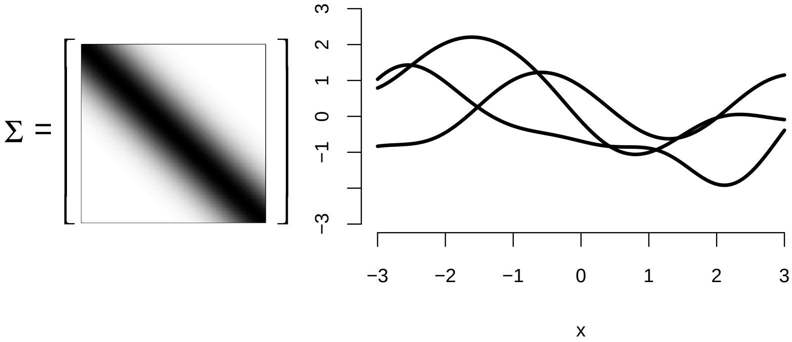
**It determines the global properties of the latent function before observing the data.**

# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

**It determines the global properties of the latent function before observing the data.**

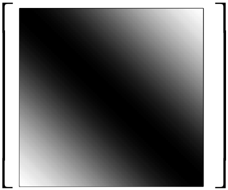


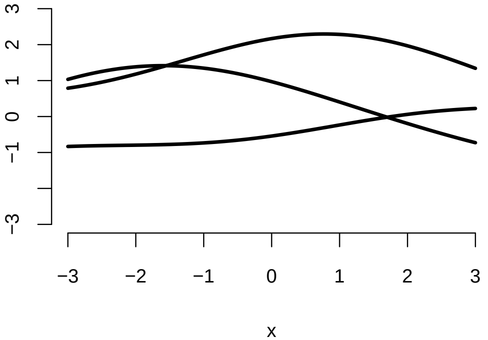
# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

**It determines the global properties of the latent function before observing the data.**

$\Sigma =$ 

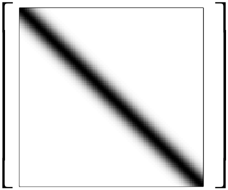


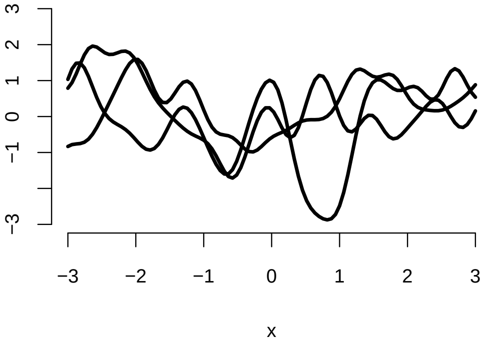
# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

**It determines the global properties of the latent function before observing the data.**

$\Sigma =$ 

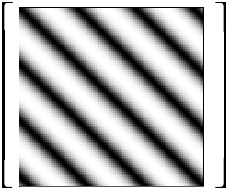


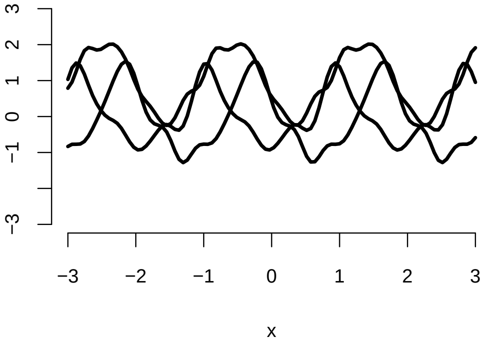
# GP Prior Covariances

The covariance function sets prior covariances among function values!

$$\mathbb{E}\left[(f(\mathbf{x}_i) - m(\mathbf{x}_i))(f(\mathbf{x}_j) - m(\mathbf{x}_j))\right] = C(\mathbf{x}_i, \mathbf{x}_j).$$

**It determines the global properties of the latent function before observing the data.**

$\Sigma =$ 



# Marginalization

**If the GP mean has infinite length and the GP covariance matrix is $\infty \times \infty$, how do we represent a GP on a computer?**

# Marginalization

**If the GP mean has infinite length and the GP covariance matrix is $\infty \times \infty$, how do we represent a GP on a computer?**

We can use the marginalization property of distributions:

$$p(\mathbf{y}_1) = \int p(\mathbf{y}_1, \mathbf{y}_2) d\mathbf{y}_2 \,,$$

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left( \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\mathsf{T} & \mathbf{B} \end{bmatrix} \right) \,,$$

# Marginalization

**If the GP mean has infinite length and the GP covariance matrix is $\infty \times \infty$, how do we represent a GP on a computer?**

We can use the marginalization property of distributions:

$$p(\mathbf{y}_1) = \int p(\mathbf{y}_1, \mathbf{y}_2) d\mathbf{y}_2 \,,$$

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left( \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\mathrm{T} & \mathbf{B} \end{bmatrix} \right) \,,$$

$$p(\mathbf{y}_1) = \mathcal{N}(\mathbf{y}_1 | \mathbf{a}, \mathbf{A}) \,,$$

# Marginalization

**If the GP mean has infinite length and the GP covariance matrix is $\infty \times \infty$, how do we represent a GP on a computer?**

We can use the marginalization property of distributions:

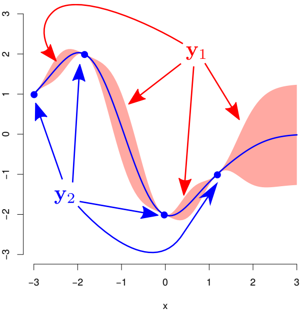
$$p(\mathbf{y}_1) = \int p(\mathbf{y}_1, \mathbf{y}_2) d\mathbf{y}_2 \,,$$

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left( \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\mathsf{T} & \mathbf{B} \end{bmatrix} \right) \,,$$

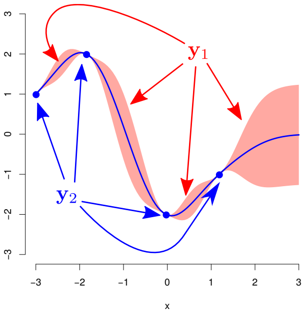
$$p(\mathbf{y}_1) = \mathcal{N}(\mathbf{y}_1 | \mathbf{a}, \mathbf{A}) \,,$$

**We only need to work with finite sets of random variables!**

# Computing the Predictive Distribution

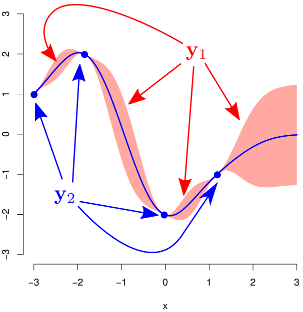


# Computing the Predictive Distribution




$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{B} \end{bmatrix}\right),$$

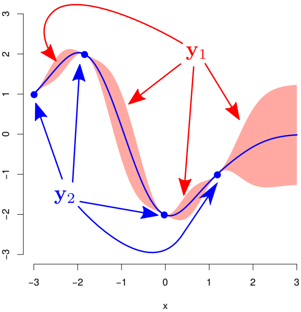
# Computing the Predictive Distribution




$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\mathsf{T} & \mathbf{B} \end{bmatrix}\right),$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)},$$

# Computing the Predictive Distribution

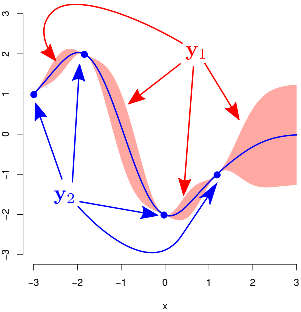



$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{B} \end{bmatrix}\right),$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)},$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}\left(\mathbf{y}_1 \middle| \mathbf{a} + \mathbf{C}\mathbf{B}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathbf{A} - \mathbf{C}\mathbf{B}^{-1}\mathbf{C}^\top\right)$$

# Computing the Predictive Distribution



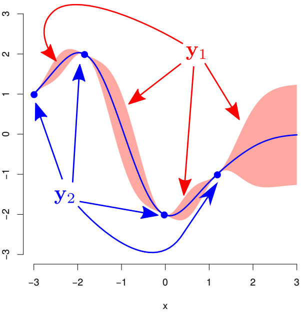

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{B} \end{bmatrix}\right),$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)},$$

$$p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}\left(\mathbf{y}_1 \middle| \mathbf{a} + \mathbf{C}\mathbf{B}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathbf{A} - \mathbf{C}\mathbf{B}^{-1}\mathbf{C}^\top\right)$$

- The predictive mean is linear in $\mathbf{y}_2$.

# Computing the Predictive Distribution



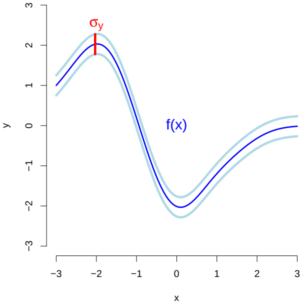

$$p(\mathbf{y}_1, \mathbf{y}_2) = \mathcal{N}\left( \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}, \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{B} \end{bmatrix} \right),$$

$$p(\mathbf{y}_1 | \mathbf{y}_2) = \frac{p(\mathbf{y}_1, \mathbf{y}_2)}{p(\mathbf{y}_2)},$$

$$p(\mathbf{y}_1 | \mathbf{y}_2) = \mathcal{N}\left( \mathbf{y}_1 \middle| \mathbf{a} + \mathbf{C}\mathbf{B}^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathbf{A} - \mathbf{C}\mathbf{B}^{-1}\mathbf{C}^\top \right)$$

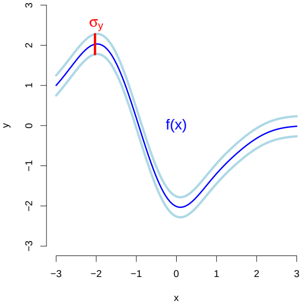
- The predictive mean is linear in $\mathbf{y}_2$.
- The predictive covariance is **more confident** than the prior!.

# Considering Additive Noise




$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon\sigma_y\,,$$
$$p(\epsilon) = \mathcal{N}(\epsilon|0,1)\,.$$

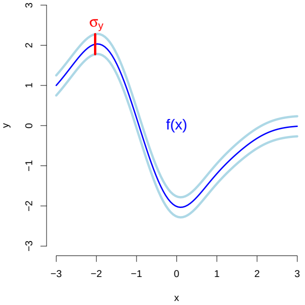
# Considering Additive Noise




$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon\sigma_y\,,$$
$$p(\epsilon) = \mathcal{N}(\epsilon|0,1)\,.$$

**Since $f(\mathbf{x})$ follows a GP and $\epsilon$ is Gaussian $y(\mathbf{x})$ is another GP!**

# Considering Additive Noise

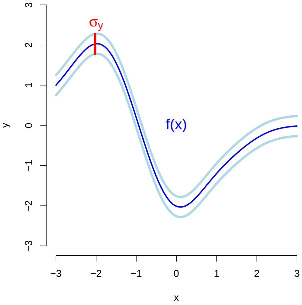



$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon\sigma_y\,,$$
$$p(\epsilon) = \mathcal{N}(\epsilon|0,1)\,.$$

**Since $f(\mathbf{x})$ follows a GP and $\epsilon$ is Gaussian $y(\mathbf{x})$ is another GP!**

$$y(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), C(\mathbf{x},\mathbf{x}') + \mathbb{I}(\mathbf{x}=\mathbf{x}')\sigma_y^2)$$

# Considering Additive Noise




$$y(\mathbf{x}) = f(\mathbf{x}) + \epsilon\sigma_y \,,$$
$$p(\epsilon) = \mathcal{N}(\epsilon|0,1)\,.$$

**Since $f(\mathbf{x})$ follows a GP and $\epsilon$ is Gaussian $y(\mathbf{x})$ is another GP!**

$$y(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), C(\mathbf{x},\mathbf{x}') + \mathbb{I}(\mathbf{x} = \mathbf{x}')\sigma_y^2)$$

The predictive distribution is:

$$p(\mathbf{y}_1|\mathbf{y}_2) = \mathcal{N}\left(\mathbf{y}_1\middle|\mathbf{a} + \mathbf{C}(\mathbf{B} + \mathbf{I}\sigma_y^2)^{-1}(\mathbf{y}_2 - \mathbf{b}), \mathbf{A} - \mathbf{C}(\mathbf{B} + \mathbf{I}\sigma_y^2)^{-1}\mathbf{C}^\mathsf{T} + \mathbf{I}\sigma_y^2\right)$$

# An Example of a Covariance Function

**Squared Exponential**: $$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \left( \frac{x_j - x_j'}{l_j} \right)^2 \right\}$$

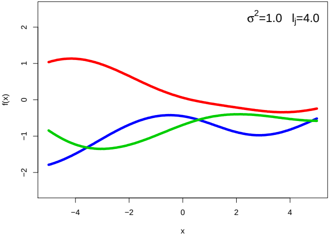
# An Example of a Covariance Function

**Squared Exponential**: $$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{-\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j - x_j'}{l_j}\right)^2\right\}$$

- Vertical scale
- Horizontal scale

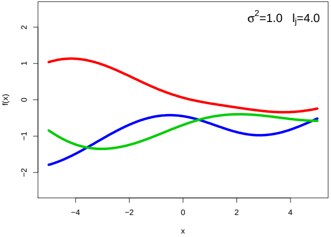
# An Example of a Covariance Function

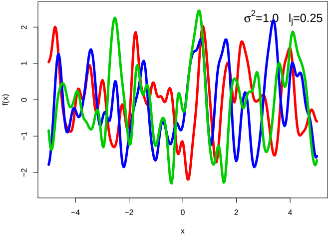
**Squared Exponential**: $C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \left( \frac{x_j - x_j'}{l_j} \right)^2 \right\}$



# An Example of a Covariance Function

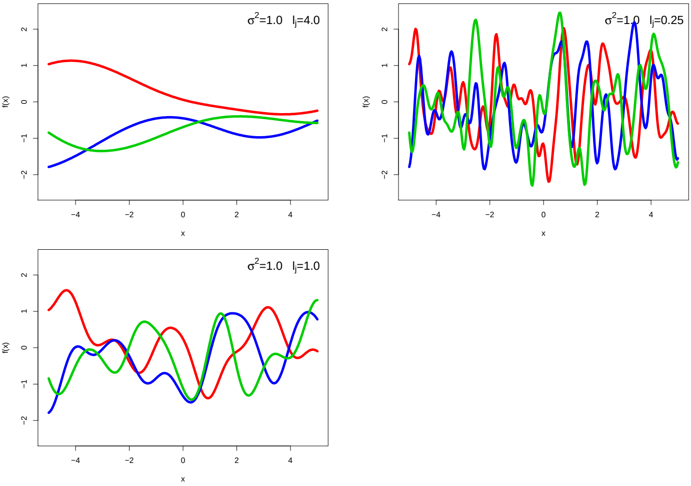
**Squared Exponential**: $$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \left( \frac{x_j - x_j'}{l_j} \right)^2 \right\}$$



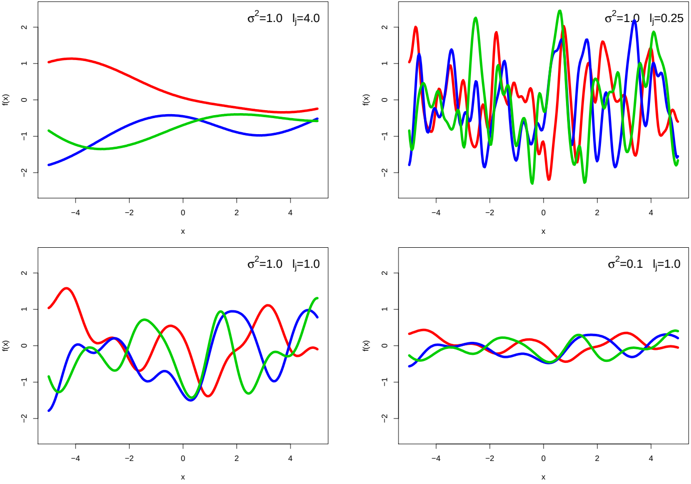
# An Example of a Covariance Function

**Squared Exponential**: $$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \left( \frac{x_j - x_j'}{l_j} \right)^2 \right\}$$



# An Example of a Covariance Function

**Squared Exponential**: $$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left\{ -\frac{1}{2} \sum_{j=1}^{d} \left( \frac{x_j - x_j'}{l_j} \right)^2 \right\}$$



**Run the first cells of the notebook to sample functions from a GP prior and complete task 1!**

# How do we choose the hyper-parameters?

**Intuition**: find parameters $\theta$ that are compatible with the observed data.

$$p(\theta|\mathbf{y}) = \frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}$$

# How do we choose the hyper-parameters?

**Intuition**: find parameters $\theta$ that are compatible with the observed data.

$$p(\theta|\mathbf{y}) = \frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}$$

| what we know after seeing the data (*posterior*) | $\propto$ | what the data tell us (likelihood) | $\times$ | what we know before seeing the data (prior) |
|---|---|---|---|---|

# How do we choose the hyper-parameters?

**Intuition**: find parameters $\theta$ that are compatible with the observed data.

$$p(\theta|\mathbf{y}) = \frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}$$

| what we know after seeing the data (*posterior*) | $\propto$ | what the data tell us (likelihood) | $\times$ | what we know before seeing the data (prior) |
|---|---|---|---|---|

$$\begin{aligned} p(\mathbf{y}|\theta) &\equiv \text{how well does } \theta \text{ explain the observed data} \\ &= \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{\Sigma} + \mathbf{I}\sigma_y^2\right) \end{aligned}$$

# How do we choose the hyper-parameters?

**Intuition**: find parameters $\theta$ that are compatible with the observed data.

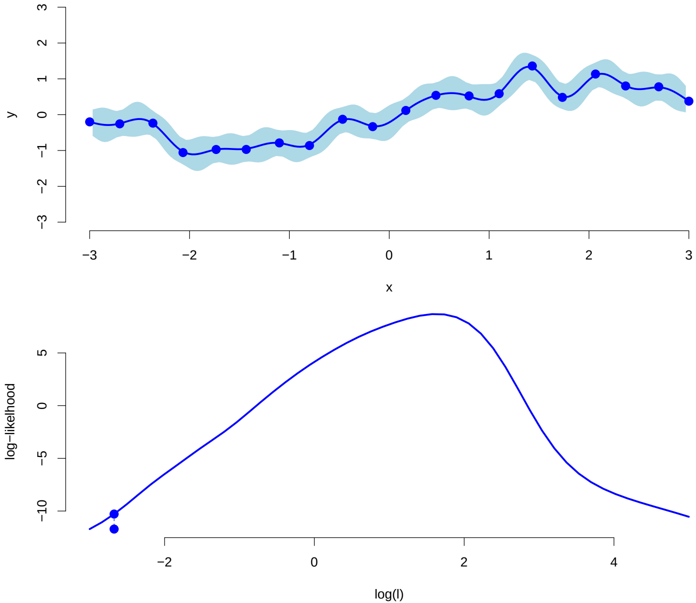
$$p(\theta|\mathbf{y}) = \frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}$$

| what we know after seeing the data (*posterior*) | $\propto$ | what the data tell us (likelihood) | $\times$ | what we know before seeing the data (prior) |
|---|---|---|---|---|

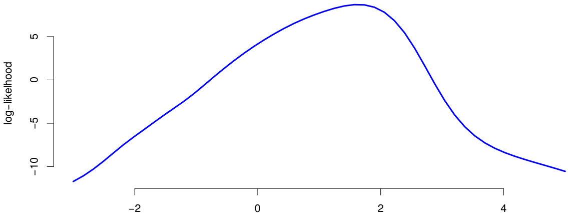
$$p(\mathbf{y}|\theta) \equiv \text{how well does } \theta \text{ explain the observed data}$$
$$= \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{\Sigma} + \mathbf{I}\sigma_y^2\right)$$

**Often, with a reasonable amount of data, maximizing $p(\mathbf{y}|\theta)$ w.r.t. $\theta$ gives good results as it favors the right model!**

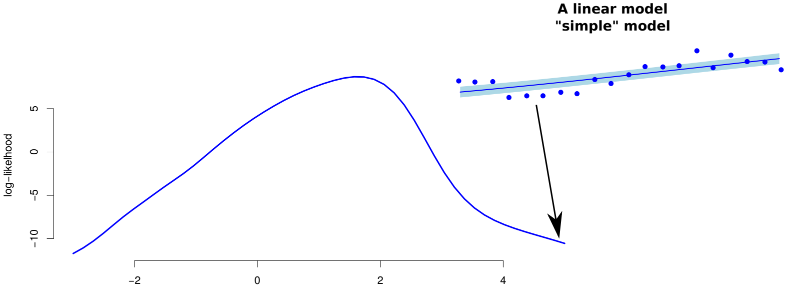
# How do we choose the hyper-parameters?



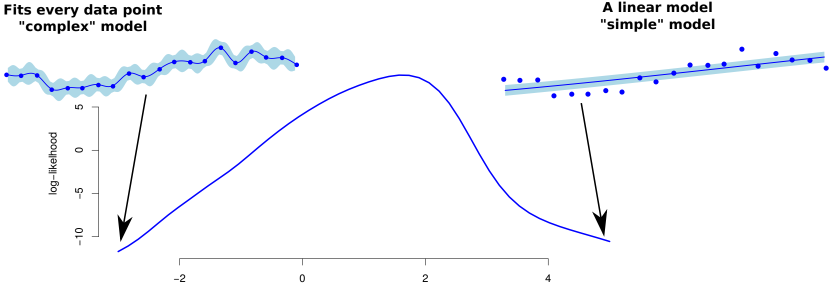
# Why maximizing the likelihood is robust?



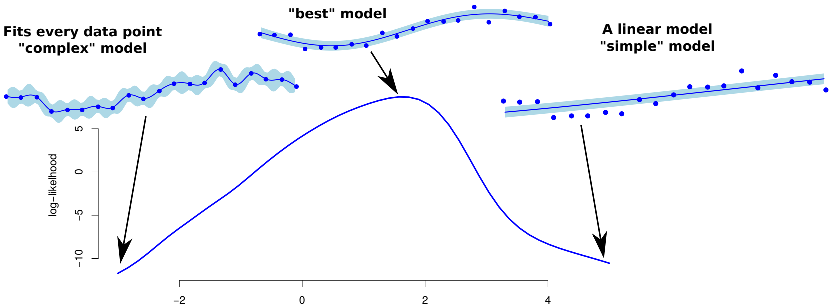
# Why maximizing the likelihood is robust?



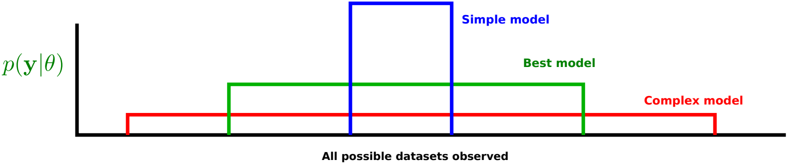
# Why maximizing the likelihood is robust?

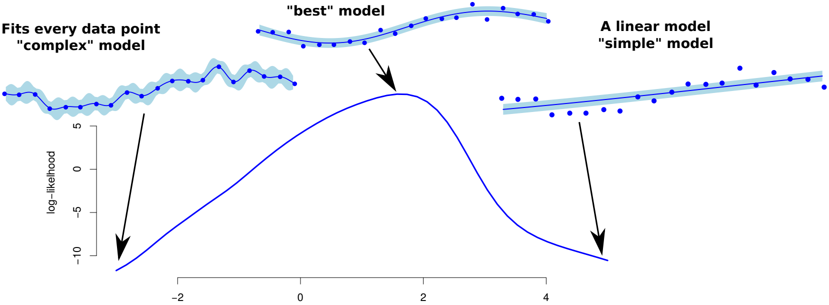


# Why maximizing the likelihood is robust?

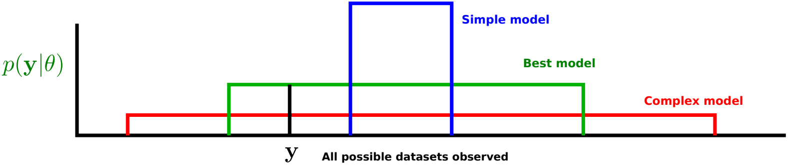


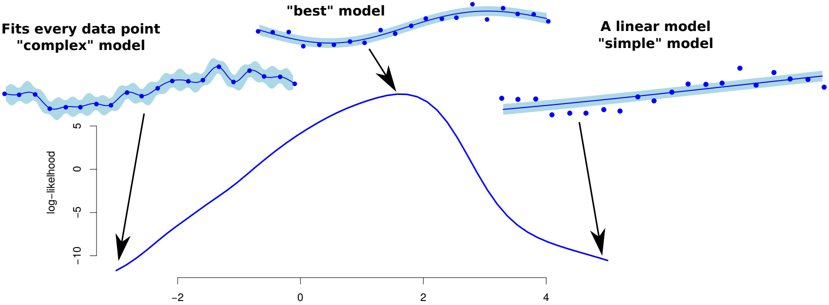
# Why maximizing the likelihood is robust?



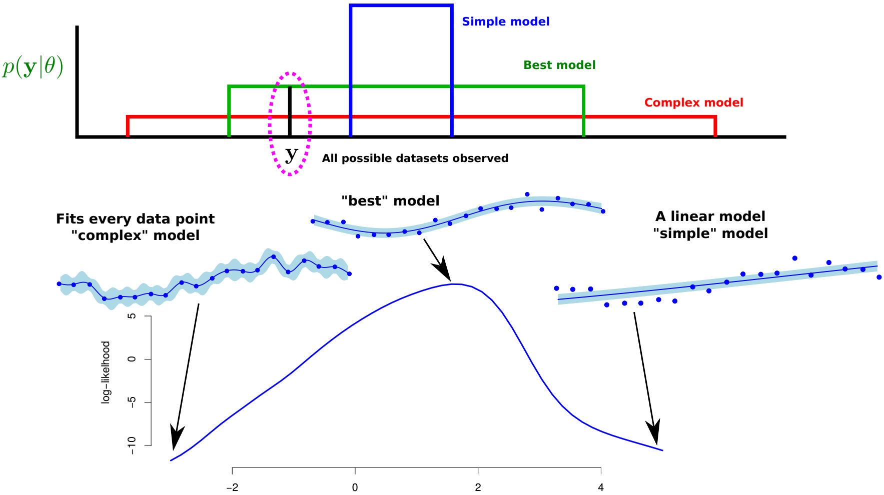



# Why maximizing the likelihood is robust?



# Why maximizing the likelihood is robust?



**Run the cells of the provided notebook to find good model hyper-parameters!**

**Run the cells of the provided notebook to find good model hyper-parameters!**

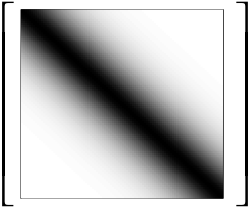
**Compare the predictive distribution with and with-out optimization of the hyper-parameters.**

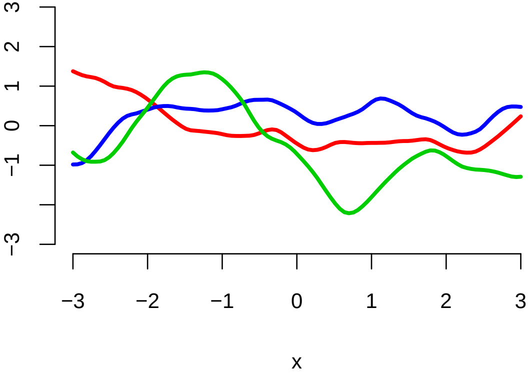
# Covariance Functions: Matérn

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2^{1-\nu)}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu r}}{l} \right)^{\nu} K_{\nu} \left( \frac{\sqrt{2\nu} r}{l} \right)$$

# Covariance Functions: Matérn

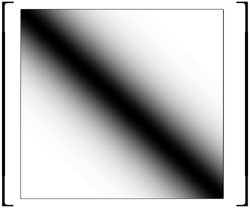
$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2^{1-\nu)}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu r}}{l} \right)^{\nu} K_{\nu} \left( \frac{\sqrt{2\nu r}}{l} \right)$$

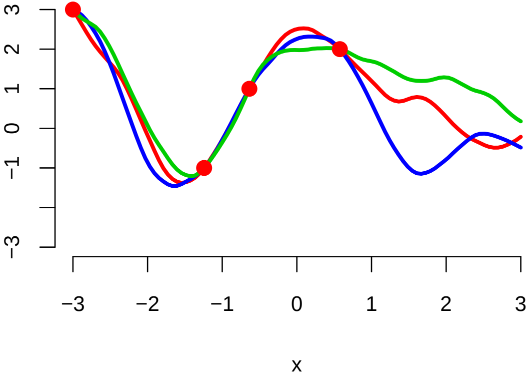
$\Sigma =$ 



# Covariance Functions: Matérn

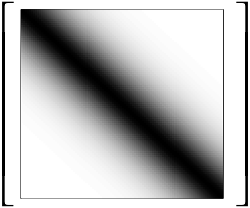
$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2^{1-\nu)}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu r}}{l} \right)^{\nu} K_{\nu} \left( \frac{\sqrt{2\nu r}}{l} \right)$$

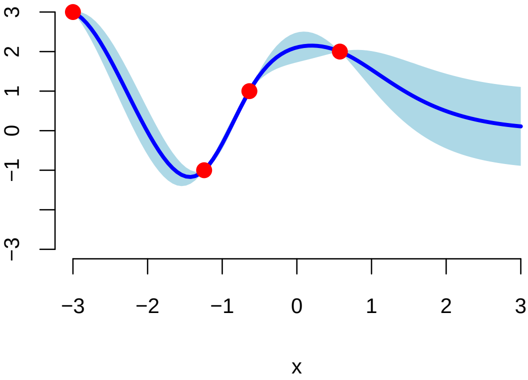
$\Sigma =$ 



# Covariance Functions: Matérn

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2^{1-\nu)}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu r}}{l} \right)^{\nu} K_\nu \left( \frac{\sqrt{2\nu r}}{l} \right)$$

$\Sigma =$ 

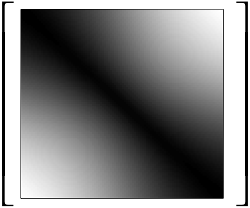


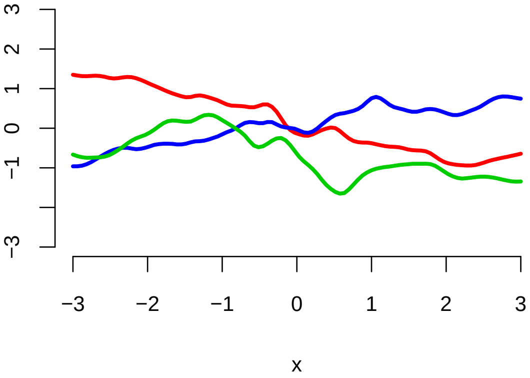
# Covariance Functions: Neural Network

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2}{\pi} \sin^{-1} \left( \frac{\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}'}{\sqrt{(1 + 2\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}')(1 + 2\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}')}} \right)$$

# Covariance Functions: Neural Network

$$C(\mathbf{x},\mathbf{x}') = \sigma^2 \frac{2}{\pi} \sin^{-1}\left(\frac{\mathbf{x}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{x}'}{\sqrt{(1+2\mathbf{x}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{x}')(1+2\mathbf{x}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{x}')}}\right)$$

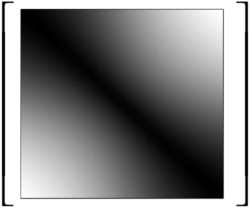
$\Sigma =$ 

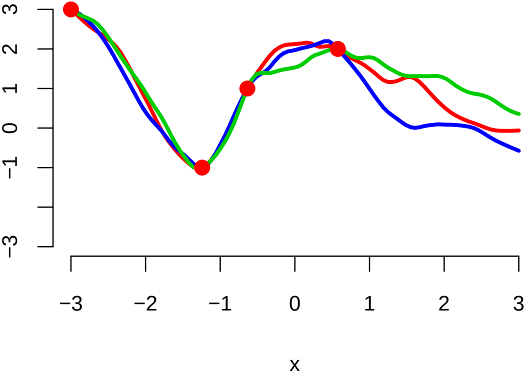


# Covariance Functions: Neural Network

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2}{\pi} \sin^{-1}\left(\frac{\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x}'}{\sqrt{(1 + 2\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x}')(1 + 2\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x}')}}\right)$$

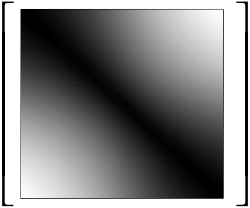
$\Sigma =$

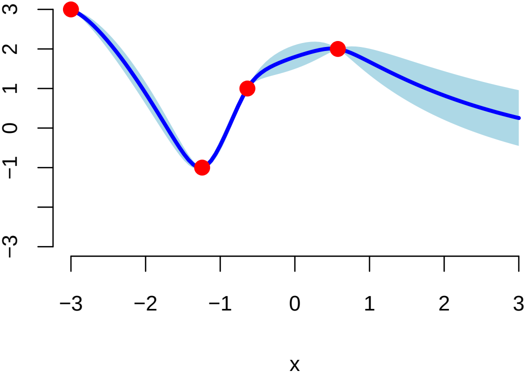




# Covariance Functions: Neural Network

$$C(\mathbf{x}, \mathbf{x}') = \sigma^2 \frac{2}{\pi} \sin^{-1} \left( \frac{\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}'}{\sqrt{(1 + 2\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}')(1 + 2\mathbf{x}^\mathsf{T} \boldsymbol{\Sigma} \mathbf{x}')}} \right)$$

$\Sigma =$ 

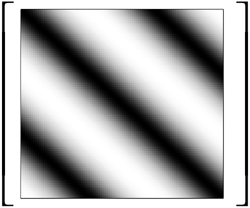


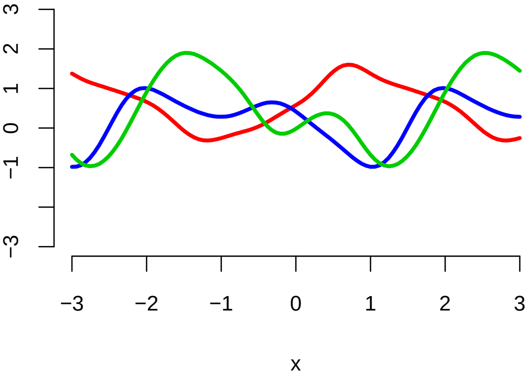
# Covariance Functions: Periodic

$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

# Covariance Functions: Periodic

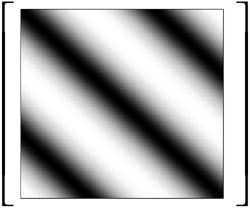
$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

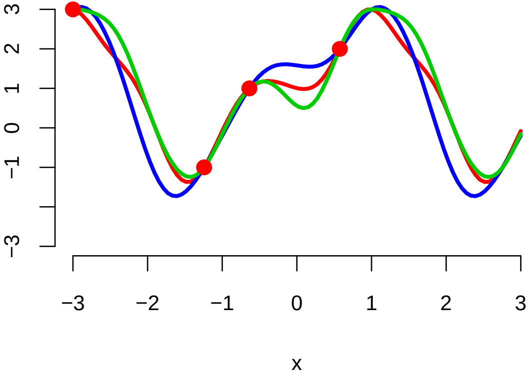
$\Sigma =$ 



# Covariance Functions: Periodic

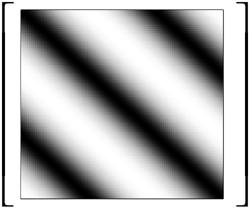
$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{ -\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2} \right\}$$

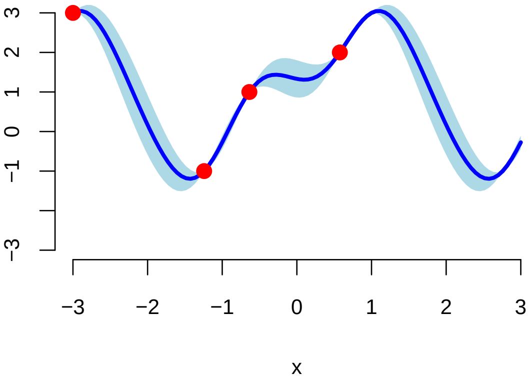
$\Sigma =$ 



# Covariance Functions: Periodic

$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{ -\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2} \right\}$$

$\Sigma =$ 

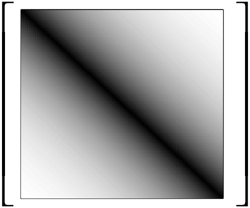


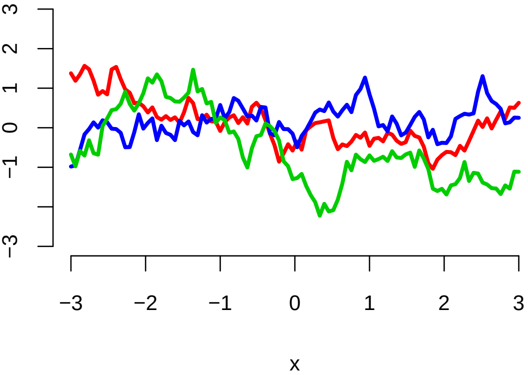
# Covariance Functions: Ornstein-Uhlenbeck

$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{|\mathbf{x} - \mathbf{x}'|}{2l^2}\right\}$$

# Covariance Functions: Ornstein-Uhlenbeck

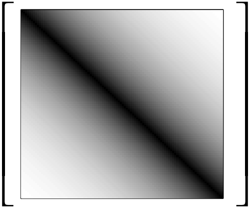
$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{|\mathbf{x} - \mathbf{x}'|}{2l^2}\right\}$$

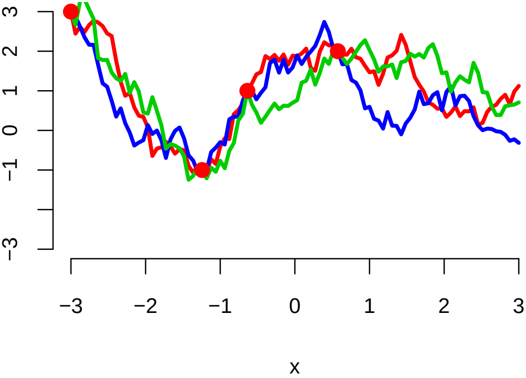
$\Sigma =$ 



# Covariance Functions: Ornstein-Uhlenbeck

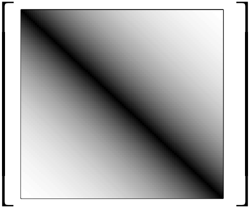
$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{|\mathbf{x} - \mathbf{x}'|}{2l^2}\right\}$$

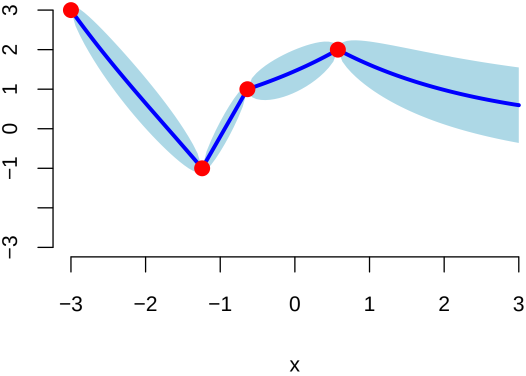
$\Sigma =$ 



# Covariance Functions: Ornstein-Uhlenbeck

$$C(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{|\mathbf{x} - \mathbf{x}'|}{2l^2}\right\}$$

$\Sigma =$ 

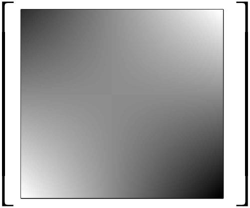


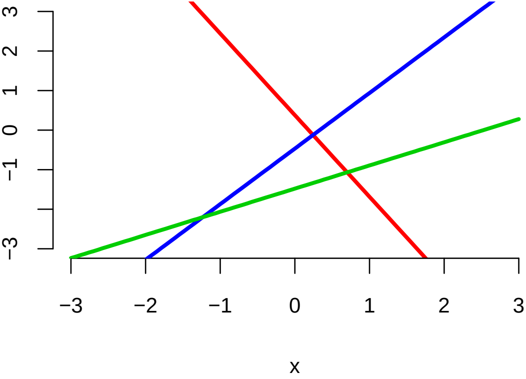
# Covariance Functions: Linear

$$C(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{c})^{\mathsf{T}}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2$$

# Covariance Functions: Linear

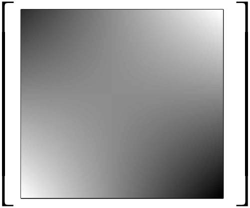
$$C(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{c})^\top (\mathbf{x}' - \mathbf{c}) \sigma_s^2 + \sigma_b^2$$

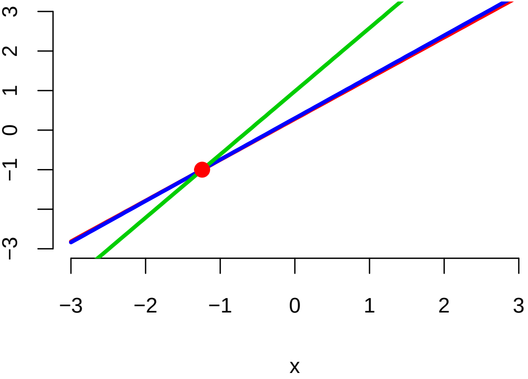
$\Sigma =$ 



# Covariance Functions: Linear

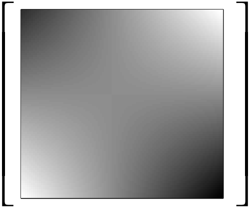
$$C(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{c})^{\top}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2$$

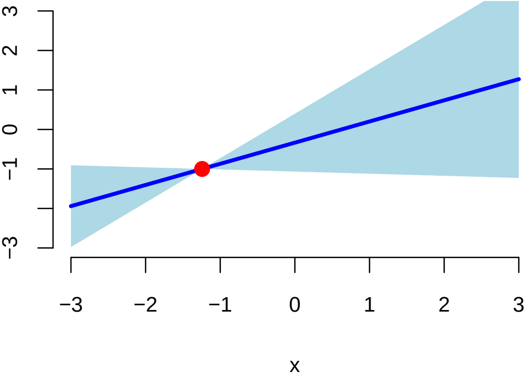
$\Sigma =$ 



# Covariance Functions: Linear

$$C(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{c})^{\top}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2$$

$\Sigma =$ 



**Complete task 2 of the notebook to see the influence of the prior mean and the covariance function on the prior samples and the predictive distribution!**

# Combining Covariance Functions: Multiplication

**The product of two covariance functions is a covariance function!**

# Combining Covariance Functions: Multiplication

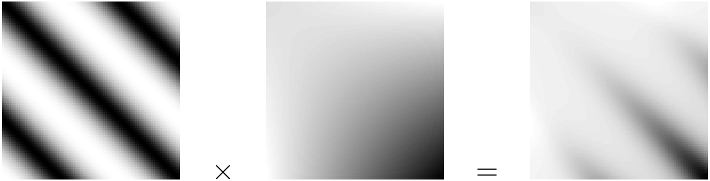
**The product of two covariance functions is a covariance function!**

**Can be thought of as an AND operation!**

# Combining Covariance Functions: Multiplication

**The product of two covariance functions is a covariance function!**

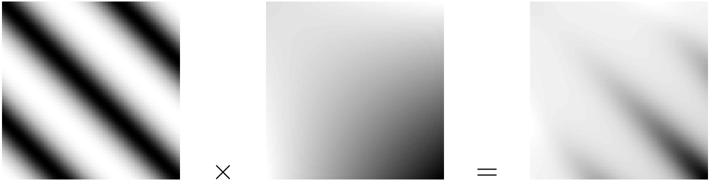
**Can be thought of as an AND operation!**



# Combining Covariance Functions: Multiplication

**The product of two covariance functions is a covariance function!**

**Can be thought of as an AND operation!**



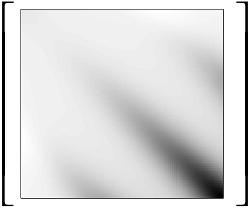
**The resulting covariance function will have high value only if both base covariances have a high value!**

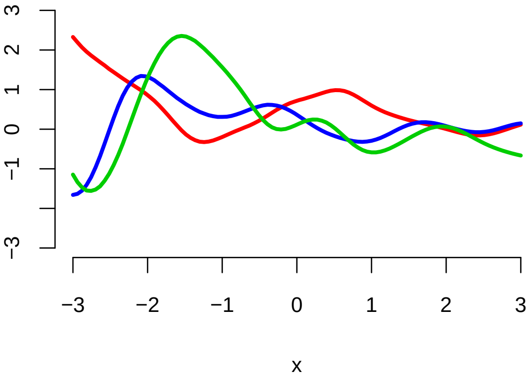
# Multiplication: Linear Times Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c})^\mathsf{T}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2\right) \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

# Multiplication: Linear Times Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left( (\mathbf{x} - \mathbf{c})^{\mathsf{T}} (\mathbf{x}' - \mathbf{c}) \sigma_s^2 + \sigma_b^2 \right) \exp \left\{ - \frac{2 \sin^2 \left( \frac{\pi |\mathbf{x} - \mathbf{x}'|}{p} \right)}{l^2} \right\}$$

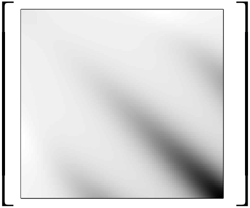
$\Sigma =$ 

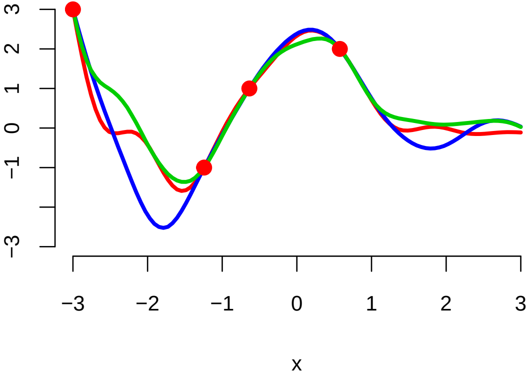


# Multiplication: Linear Times Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c})^{\mathsf{T}}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2\right) \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

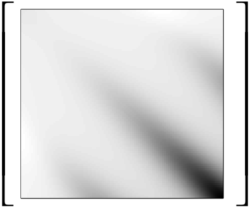
$\Sigma =$

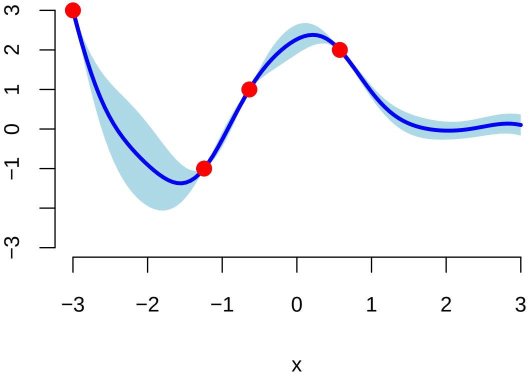




# Multiplication: Linear Times Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c})^{\mathsf{T}}(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2\right) \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

$\Sigma =$ 



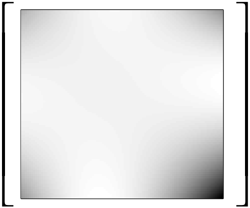
# Multiplication: Linear Times Linear

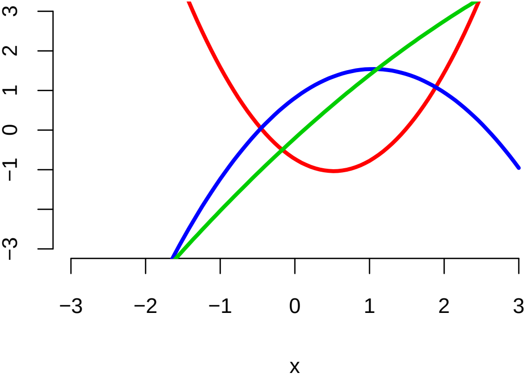
$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c}_1)^\top (\mathbf{x}' - \mathbf{c}_1)\sigma_{s_1}^2 + \sigma_{b_1}^2\right) \left((\mathbf{x} - \mathbf{c}_2)^\top (\mathbf{x}' - \mathbf{c}_2)\sigma_{s_2}^2 + \sigma_{b_2}^2\right)$$

# Multiplication: Linear Times Linear

$$C(\mathbf{x}, \mathbf{x}') = \left( (\mathbf{x} - \mathbf{c}_1)^\top (\mathbf{x}' - \mathbf{c}_1) \sigma_{s_1}^2 + \sigma_{b_1}^2 \right) \left( (\mathbf{x} - \mathbf{c}_2)^\top (\mathbf{x}' - \mathbf{c}_2) \sigma_{s_2}^2 + \sigma_{b_2}^2 \right)$$

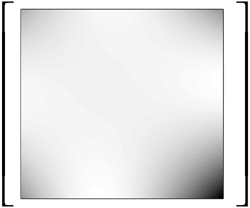
$\Sigma =$

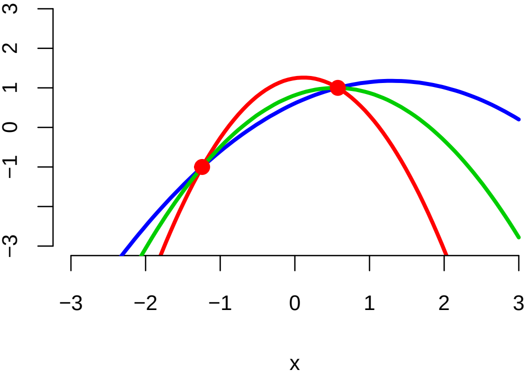




# Multiplication: Linear Times Linear

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c}_1)^\top(\mathbf{x}' - \mathbf{c}_1)\sigma_{s_1}^2 + \sigma_{b_1}^2\right)\left((\mathbf{x} - \mathbf{c}_2)^\top(\mathbf{x}' - \mathbf{c}_2)\sigma_{s_2}^2 + \sigma_{b_2}^2\right)$$

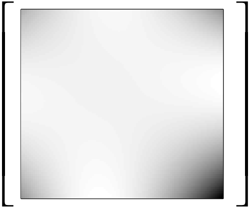
$\Sigma =$ 

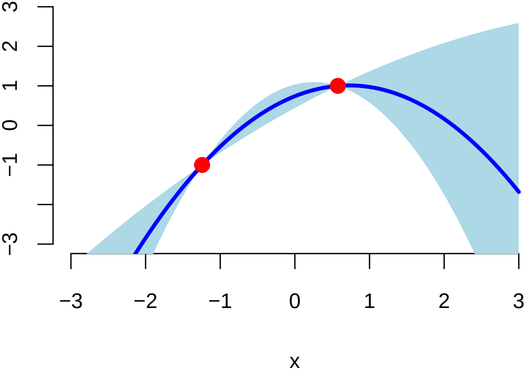


# Multiplication: Linear Times Linear

$$C(\mathbf{x},\mathbf{x}') = \left((\mathbf{x}-\mathbf{c}_1)^{\mathsf{T}}(\mathbf{x}'-\mathbf{c}_1)\sigma_{s_1}^2+\sigma_{b_1}^2\right)\left((\mathbf{x}-\mathbf{c}_2)^{\mathsf{T}}(\mathbf{x}'-\mathbf{c}_2)\sigma_{s_2}^2+\sigma_{b_2}^2\right)$$

$\Sigma =$





# Combining Covariance Functions: Addition

**The addition of two covariance functions is a covariance function!**

# Combining Covariance Functions: Addition

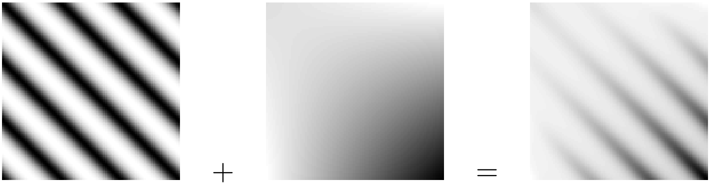
**The addition of two covariance functions is a covariance function!**

**Can be thought of as an OR operation!**

# Combining Covariance Functions: Addition

**The addition of two covariance functions is a covariance function!**

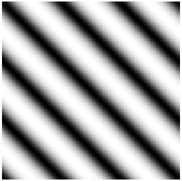
**Can be thought of as an OR operation!**



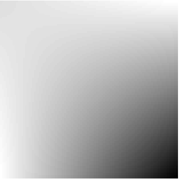
# Combining Covariance Functions: Addition

**The addition of two covariance functions is a covariance function!**

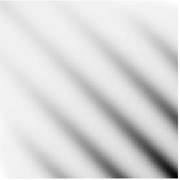
**Can be thought of as an OR operation!**



$+$



$=$



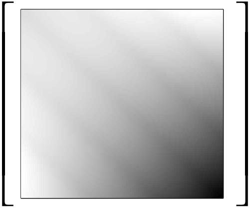
**The resulting covariance function will have high value if either of the base covariances have a high value!**

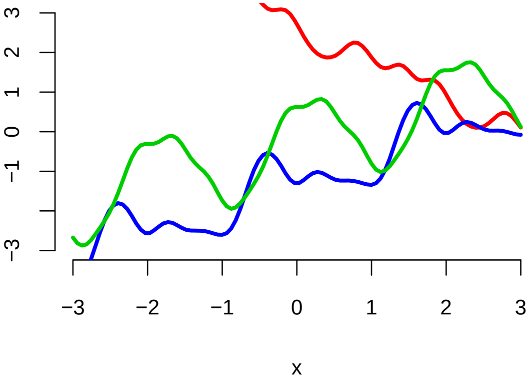
# Addition: Linear Plus Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c})^\top (\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2\right) + \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

# Addition: Linear Plus Periodic

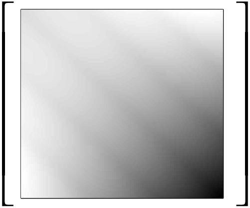
$$C(\mathbf{x},\mathbf{x}') = \left((\mathbf{x}-\mathbf{c})^\top(\mathbf{x}'-\mathbf{c})\sigma_s^2+\sigma_b^2\right) + \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

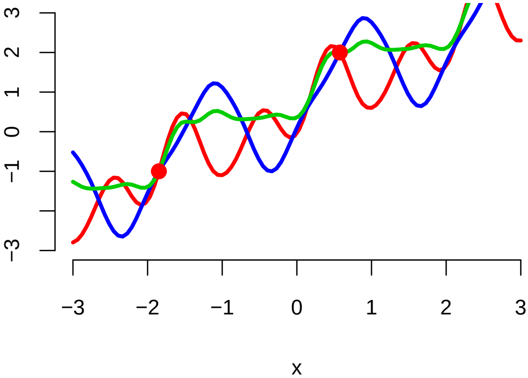
$\Sigma =$ 



# Addition: Linear Plus Periodic

$$C(\mathbf{x}, \mathbf{x}') = \left((\mathbf{x} - \mathbf{c})^\top(\mathbf{x}' - \mathbf{c})\sigma_s^2 + \sigma_b^2\right) + \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

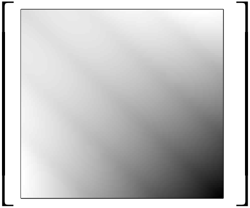
$\Sigma =$ 

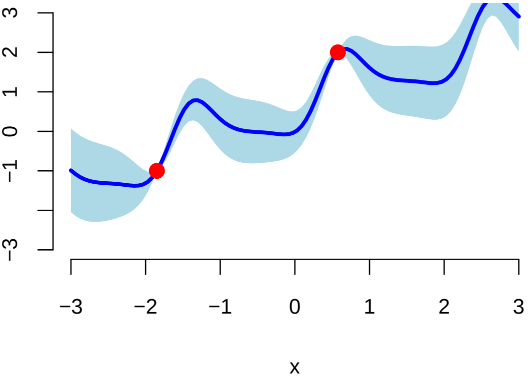


# Addition: Linear Plus Periodic

$$C(\mathbf{x},\mathbf{x}') = \left((\mathbf{x}-\mathbf{c})^{\mathsf{T}}(\mathbf{x}'-\mathbf{c})\sigma_s^2 + \sigma_b^2\right) + \exp\left\{-\frac{2\sin^2\left(\frac{\pi|\mathbf{x}-\mathbf{x}'|}{p}\right)}{l^2}\right\}$$

$\Sigma =$





# Summary about Covariance Functions

- Covariance functions include strong assumptions about $f(\mathbf{x})$.

# Summary about Covariance Functions

- Covariance functions include strong assumptions about $f(\mathbf{x})$.
- Often the sq. exponential or Matérn work fine for regression.

# Summary about Covariance Functions

- Covariance functions include strong assumptions about $f(\mathbf{x})$.
- Often the sq. exponential or Matérn work fine for regression.
- Covariance functions parameters allow to interpret the data.

# Summary about Covariance Functions

- Covariance functions include strong assumptions about $f(\mathbf{x})$.
- Often the sq. exponential or Matérn work fine for regression.
- Covariance functions parameters allow to interpret the data.
- Covariance functions can be combined (sum $+$ and product $\times$).

# Summary about Covariance Functions

- Covariance functions include strong assumptions about $f(\mathbf{x})$.
- Often the sq. exponential or Matérn work fine for regression.
- Covariance functions parameters allow to interpret the data.
- Covariance functions can be combined (sum $+$ and product $\times$).
- The likelihood $p(\mathbf{y})$ can *discriminate* among them (use with care).

**Run the notebook code for extrapolation and interpretation and complete task 3!**

# Classification Problems and Decision Theory

A classification rule will **divide the input space** in regions $\mathcal{R}_k$.

# Classification Problems and Decision Theory

A classification rule will **divide the input space** in regions $\mathcal{R}_k$.

What is the optimal rule in terms of the prediction error?

# Classification Problems and Decision Theory

A classification rule will **divide the input space** in regions $\mathcal{R}_k$.

What is the optimal rule in terms of the prediction error?

Consider a binary problem. The probability of a mistake is:

$$\begin{aligned} p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2) d\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1) d\mathbf{x} \,. \end{aligned}$$

# Classification Problems and Decision Theory

A classification rule will **divide the input space** in regions $\mathcal{R}_k$.

What is the optimal rule in terms of the prediction error?

Consider a binary problem. The probability of a mistake is:

$$\begin{aligned} p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2) d\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1) d\mathbf{x} \,. \end{aligned}$$

Clearly the assign rule that **minimizes** $p(\text{mistake})$ is:

$$\pi(\mathbf{x}) = \begin{cases} \mathcal{C}_1 & \text{if} \quad p(\mathbf{x}, \mathcal{C}_1) \geq p(\mathbf{x}, \mathcal{C}_2) \\ \mathcal{C}_2 & \text{if} \quad p(\mathbf{x}, \mathcal{C}_2) > p(\mathbf{x}, \mathcal{C}_1) \end{cases}$$

# Classification Problems and Decision Theory

A classification rule will **divide the input space** in regions $\mathcal{R}_k$.

What is the optimal rule in terms of the prediction error?

Consider a binary problem. The probability of a mistake is:

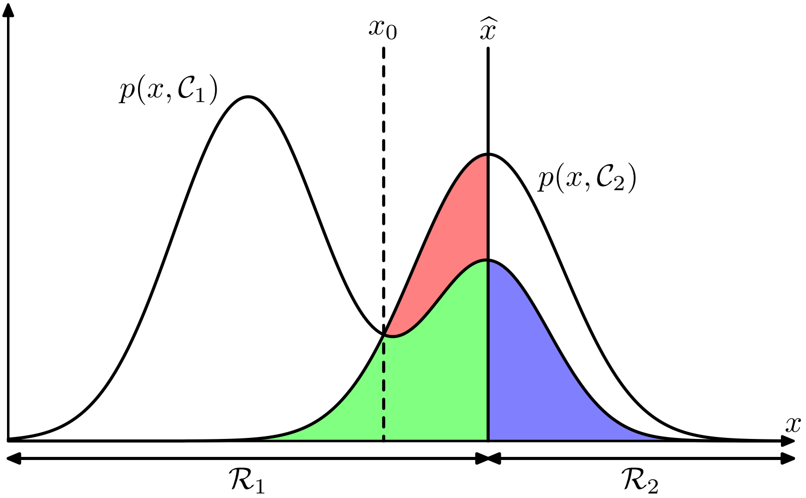
$$
\begin{aligned}
p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\
&= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2) d\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1) d\mathbf{x} \,.
\end{aligned}
$$

Clearly the assign rule that **minimizes** $p(\text{mistake})$ is:

$$
\pi(\mathbf{x}) = \begin{cases} \mathcal{C}_1 & \text{if} \quad p(\mathbf{x}, \mathcal{C}_1) \geq p(\mathbf{x}, \mathcal{C}_2) \\ \mathcal{C}_2 & \text{if} \quad p(\mathbf{x}, \mathcal{C}_2) > p(\mathbf{x}, \mathcal{C}_1) \end{cases}
$$

i.e., we should assign the class for which $p(\mathcal{C}_k|\mathbf{x}) \propto p(\mathbf{x}, \mathcal{C}_1)$ is larger.

# Classification Problems and Decision Theory




(Bishop, 2006)

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

Function squashing to the interval $[0, 1]$ via a link function:

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

Function squashing to the interval $[0, 1]$ via a link function:

- $p(y_i = 1|\mathbf{x}_i) = I(f(\mathbf{x}_i) > 0)$.

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

Function squashing to the interval $[0, 1]$ via a link function:

- $p(y_i = 1|\mathbf{x}_i) = I(f(\mathbf{x}_i) > 0)$.
- $p(y_i = 1|\mathbf{x}_i) = \text{sigmoid}(f(\mathbf{x}_i)) \qquad \text{sigmoid}(x) = (1 + \exp(-x))^{-1}$.

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

Function squashing to the interval $[0, 1]$ via a link function:

- $p(y_i = 1|\mathbf{x}_i) = I(f(\mathbf{x}_i) > 0)$.
- $p(y_i = 1|\mathbf{x}_i) = \text{sigmoid}(f(\mathbf{x}_i)) \qquad \text{sigmoid}(x) = (1 + \exp(-x))^{-1}$.
- $p(y_i = 1|\mathbf{x}_i) = \text{probit}(f(\mathbf{x}_i))$ (c.d.f. of a standard Gaussian).

# Binary Classification Problems

**The goal is to estimate class $\{1, -1\}$ posterior probabilities, e.g., $p(y_i = 1|\mathbf{x}_i)$ from the observed data.**

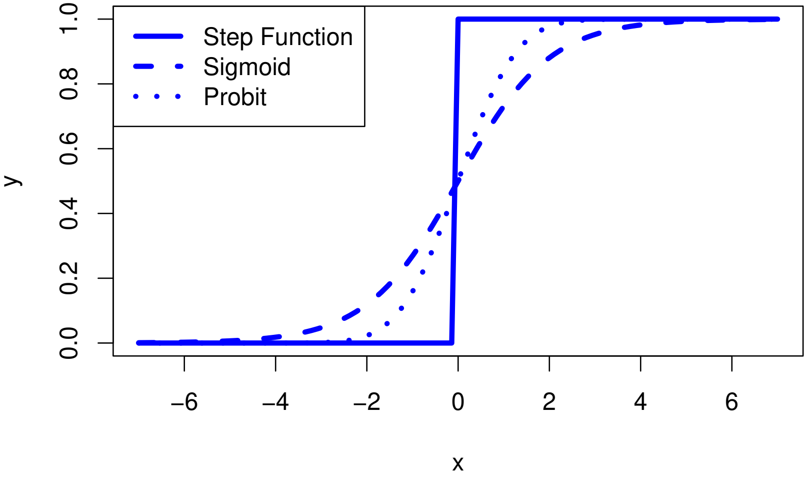
**Gaussian processes are priors for functions that take values in $\mathbb{R}$.**

Function squashing to the interval $[0, 1]$ via a link function:

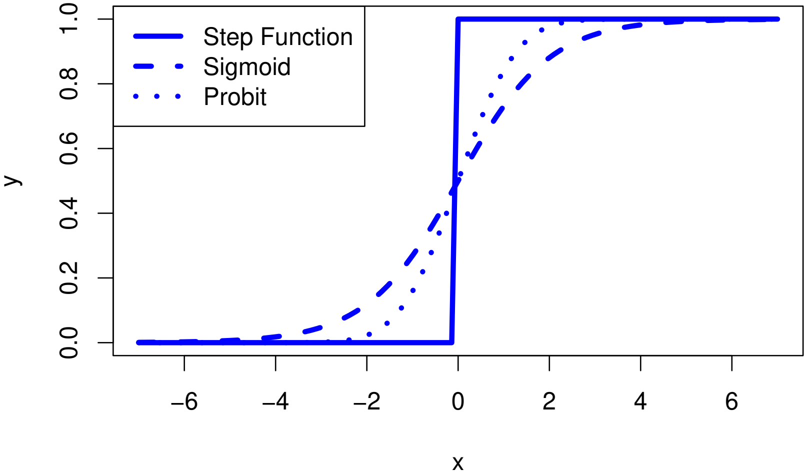
- $p(y_i = 1|\mathbf{x}_i) = I(f(\mathbf{x}_i) > 0)$.
- $p(y_i = 1|\mathbf{x}_i) = \text{sigmoid}(f(\mathbf{x}_i)) \qquad \text{sigmoid}(x) = (1 + \exp(-x))^{-1}$.
- $p(y_i = 1|\mathbf{x}_i) = \text{probit}(f(\mathbf{x}_i))$ (c.d.f. of a standard Gaussian).

with $f(\cdot)$ a latent function modeled by a GP.

# Binary Classification Problems

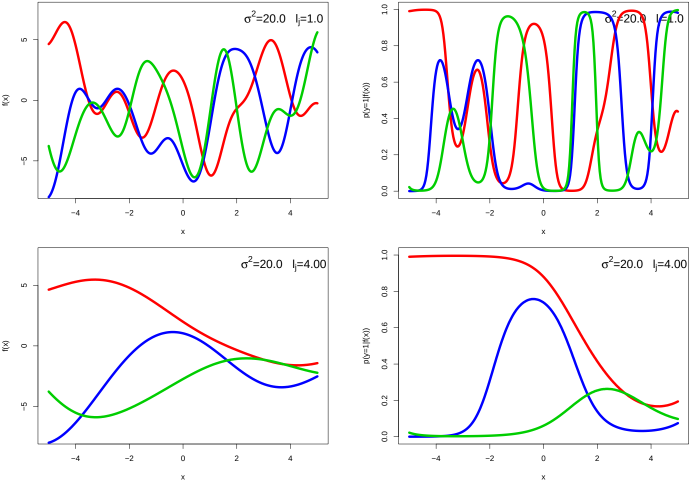


# Binary Classification Problems




**The sigmoid and probit consider logistic and standard Gaussian noise!** $p(y_i = 1|\mathbf{x}_i) = I(f(\mathbf{x}_i) + \epsilon_i > 0)$

# Prior Samples Squashed via the Sigmoid Function



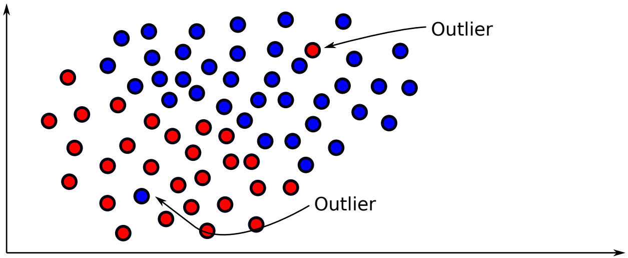
# Noise in The Labels

**The previous link functions only allow for mislabeled instances near the decision boundary!**

# Noise in The Labels

**The previous link functions only allow for mislabeled instances near the decision boundary!**

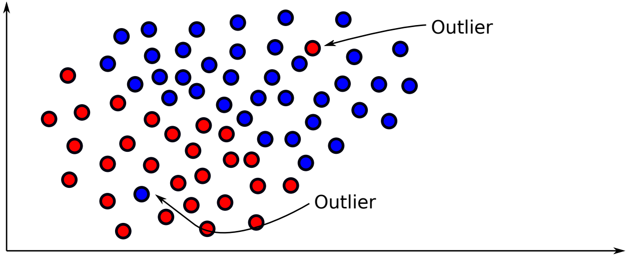
**Outliers** may affect the inference process about $f(\cdot)$:



# Noise in The Labels

**The previous link functions only allow for mislabeled instances near the decision boundary!**

**Outliers** may affect the inference process about $f(\cdot)$:




**Robust likelihood** with probability $\epsilon$ of label flip:

$$p(y|f(\mathbf{x}_i), \epsilon) = (1-\epsilon)\cdot\sigma(f(\mathbf{x}_i)) + \epsilon\cdot(1-\sigma(f(\mathbf{x}_i)))$$

# Bayesian Inference for GP Classification

**Ideally, we would like to make inference about the latent variables, i.e., process values at the observed data.**

# Bayesian Inference for GP Classification

**Ideally, we would like to make inference about the latent variables, i.e., process values at the observed data.**

Let $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ and $y_i \in \{-1, 1\}$:

$$p(\mathbf{f}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}$$

# Bayesian Inference for GP Classification

**Ideally, we would like to make inference about the latent variables, i.e., process values at the observed data.**

Let $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^{\mathsf{T}}$ and $y_i \in \{-1, 1\}$:

$$p(\mathbf{f}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}$$

with

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{N} \sigma(y_i f(\mathbf{x}_i))\,, \qquad p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \boldsymbol{\Sigma})\,,$$

# Bayesian Inference for GP Classification

**Ideally, we would like to make inference about the latent variables, i.e., process values at the observed data.**

Let $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^\mathsf{T}$ and $y_i \in \{-1, 1\}$:

$$p(\mathbf{f}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}$$

with

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{N} \sigma(y_i f(\mathbf{x}_i)), \qquad p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{\Sigma}),$$

**Unfortunately, the posterior is intractable since the likelihood is not Gaussian and must be approximated!**

# The Laplace Approximation: Univariate Case

**The logarithm of a Gaussian is a quadratic function!**

# The Laplace Approximation: Univariate Case

**The logarithm of a Gaussian is a quadratic function!**

$$q(z) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(z-\mu)^2\right\}$$

# The Laplace Approximation: Univariate Case

**The logarithm of a Gaussian is a quadratic function!**

$$q(z) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(z-\mu)^2\right\}$$

Let $f(z)$ be a target unormalized distribution. A truncated Taylor expansion of $\log f(z)$ centered at a mode is:

$$\log f(z) \approx \log f(z_0) - \frac{1}{2}A(z-z_0)^2\,, \qquad A = -\frac{d^2}{dz^2}\log f(z)\Big|_{z=z_0}$$

# The Laplace Approximation: Univariate Case

**The logarithm of a Gaussian is a quadratic function!**

$$q(z) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(z-\mu)^2\right\}$$

Let $f(z)$ be a target unormalized distribution. A truncated Taylor expansion of $\log f(z)$ centered at a mode is:

$$\log f(z) \approx \log f(z_0) - \frac{1}{2}A(z-z_0)^2\,, \qquad A = -\frac{d^2}{dz^2}\log f(z)\Big|_{z=z_0}$$

Taking the exponential we obtain:

$$f(z) \approx f(z_0)\exp\left\{-\frac{A}{2}(z-z_0)^2\right\} = \tilde{q} \qquad q(z) = \mathcal{N}(z|z_0, A^{-1})$$

# The Laplace Approximation: Univariate Case

**The logarithm of a Gaussian is a quadratic function!**

$$q(z) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(z-\mu)^2\right\}$$

Let $f(z)$ be a target unormalized distribution. A truncated Taylor expansion of $\log f(z)$ centered at a mode is:

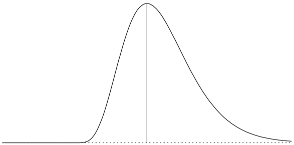
$$\log f(z) \approx \log f(z_0) - \frac{1}{2}A(z-z_0)^2\,, \qquad A = -\frac{d^2}{dz^2}\log f(z)\Big|_{z=z_0}$$

Taking the exponential we obtain:

$$f(z) \approx f(z_0)\exp\left\{-\frac{A}{2}(z-z_0)^2\right\} = \tilde{q} \qquad q(z) = \mathcal{N}(z|z_0, A^{-1})$$

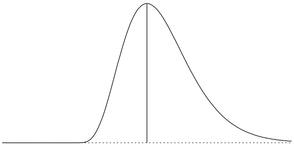
The approximate **normalization constant** $Z_q$ is $f(z_0)\sqrt{\frac{2\pi}{A}}$.

# Laplace Approximation: Illustration

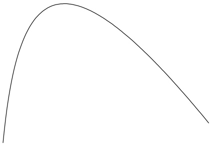


$f(z)$

# Laplace Approximation: Illustration

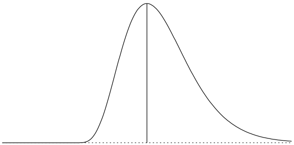


$f(z)$

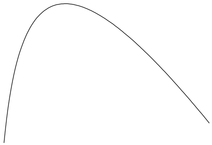


$\log f(z)$

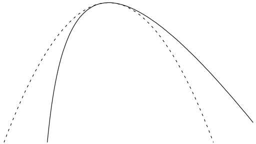
# Laplace Approximation: Illustration



$f(z)$

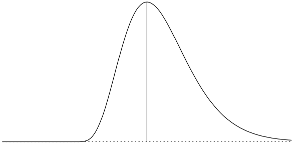


$\log f(z)$

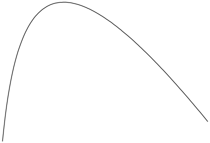


$\log f(z)$ and $\log \tilde{q}(z)$

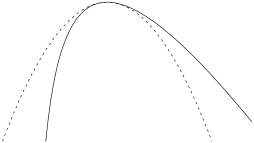
# Laplace Approximation: Illustration



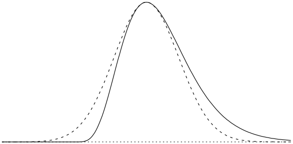
$f(z)$



$\log f(z)$



$\log f(z)$ and $\log \tilde{q}(z)$



$f(z)$ and $q(z)$

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

$$\log f(\mathbf{z}_0) \approx \log f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0)\,, \quad \mathbf{A} = -\left.\nabla^{\mathsf{T}}\nabla \log f(\mathbf{z})\right|_{\mathbf{z}=\mathbf{z}_0}$$

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

$$\log f(\mathbf{z}_0) \approx \log f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathsf{T}} \mathbf{A} (\mathbf{z} - \mathbf{z}_0)\,, \quad \mathbf{A} = -\,\nabla^{\mathsf{T}} \nabla \log f(\mathbf{z})\Big|_{\mathbf{z}=\mathbf{z}_0}$$

Taking the exponential we have:

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

$$\log f(\mathbf{z}_0) \approx \log f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0)\,, \quad \mathbf{A} = -\nabla^{\mathsf{T}}\nabla \log f(\mathbf{z})\Big|_{\mathbf{z}=\mathbf{z}_0}$$

Taking the exponential we have:

$$f(\mathbf{z}) \approx f(\mathbf{z}_0)\exp\left\{-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0))\right\} = \tilde{q}\,, \quad q(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1})$$

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

$$\log f(\mathbf{z}_0) \approx \log f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\,, \quad \mathbf{A} = -\nabla^{\mathsf{T}}\nabla \log f(\mathbf{z})\Big|_{\mathbf{z}=\mathbf{z}_0}$$

Taking the exponential we have:

$$f(\mathbf{z}) \approx f(\mathbf{z}_0)\exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathsf{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0))\right\} = \tilde{q}\,, \quad q(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1})$$

The approximate normalization constant $Z_q$ is $f(z_0)\sqrt{\frac{(2\pi)^M}{|\mathbf{A}|}}$. The **mean** of the Gaussian approximation $q$ is $\mathbf{z}_0$ and the covariance matrix is $\mathbf{A}^{-1}$.

# The Laplace Approximation: Multi-variate Case

The same principle can be applied to approximate an $M$-**dimensional distribution** $p(\mathbf{z}) = f(\mathbf{z})/Z$.

$$\log f(\mathbf{z}_0) \approx \log f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^\mathsf{T}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\,, \quad \mathbf{A} = -\nabla^\mathsf{T}\nabla \log f(\mathbf{z})\Big|_{\mathbf{z}=\mathbf{z}_0}$$

Taking the exponential we have:

$$f(\mathbf{z}) \approx f(\mathbf{z}_0)\exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^\mathsf{T}\mathbf{A}(\mathbf{z}-\mathbf{z}_0))\right\} = \tilde{q}\,, \quad q(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1})$$

The approximate normalization constant $Z_q$ is $f(z_0)\sqrt{\frac{(2\pi)^M}{|\mathbf{A}|}}$. The **mean** of the Gaussian approximation $q$ is $\mathbf{z}_0$ and the covariance matrix is $\mathbf{A}^{-1}$.

The posterior is unimodal and hence $\mathbf{A}$ is positive semidefinite.

# Approximate Predictive Distribution

**Given the Gaussian approximation $q(\mathbf{f})$, we can use the conditional Gaussian to compute an approximate predictive distribution.**

# Approximate Predictive Distribution

**Given the Gaussian approximation $q(\mathbf{f})$, we can use the conditional Gaussian to compute an approximate predictive distribution.**

$$
\begin{aligned}
p(y_\star|\mathbf{y},\mathbf{X}) &\approx \int p(y_\star|f(\mathbf{x}_\star))p(f(\mathbf{x}_\star)|\mathbf{f})q(\mathbf{f})d\mathbf{f}\,df(\mathbf{x}_\star)\,,\\
&= \int p(y_\star|f(\mathbf{x}_\star))q(f(\mathbf{x}_\star))df(\mathbf{x}_\star)\,,
\end{aligned}
$$

# Approximate Predictive Distribution

**Given the Gaussian approximation $q(\mathbf{f})$, we can use the conditional Gaussian to compute an approximate predictive distribution.**

$$
\begin{aligned}
p(y_\star|\mathbf{y}, \mathbf{X}) &\approx \int p(y_\star|f(\mathbf{x}_\star))p(f(\mathbf{x}_\star)|\mathbf{f})q(\mathbf{f})d\mathbf{f}\,df(\mathbf{x}_\star)\,,\\
&= \int p(y_\star|f(\mathbf{x}_\star))q(f(\mathbf{x}_\star))df(\mathbf{x}_\star)\,,
\end{aligned}
$$

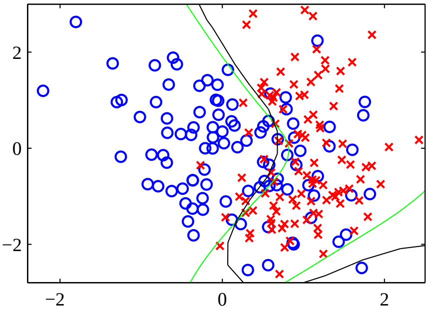
with this last integral evaluated via quadrature and

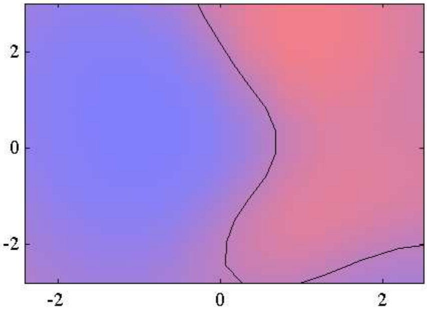
$$
q(f(\mathbf{x}_\star)) = \mathcal{N}(f(\mathbf{x}_\star)|\mathbf{c}_\star^\top\mathbf{C}^{-1}\mathbf{f}_0, C(\mathbf{x}_\star, \mathbf{x}_\star) - \mathbf{c}_\star^\top\mathbf{C}^{-1}\mathbf{c}_\star + \mathbf{c}_\star^\top\mathbf{C}^{-1}\mathbf{A}^{-1}\mathbf{C}^{-1}\mathbf{c}_\star)\,,
$$

$$
p(y_\star|f(\mathbf{x}_\star)) = \sigma(y_\star f(\mathbf{x}_\star))\,.
$$

# Approximate Predictive Distribution

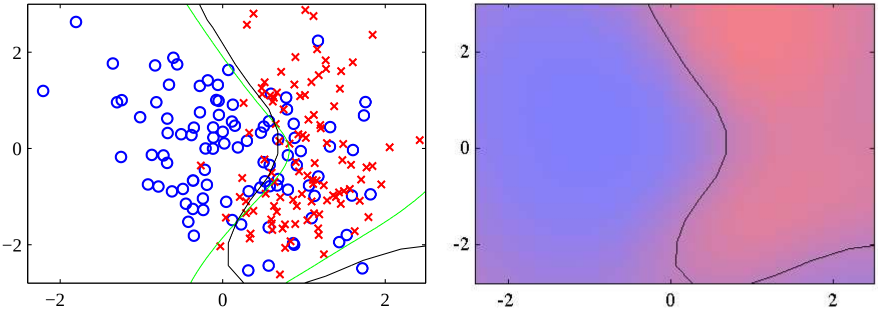
Decision boundary and prediction uncertainty:



# Approximate Predictive Distribution

Decision boundary and prediction uncertainty:




**Prediction uncertainty is higher in regions with no observed data.**

(Bishop, 2006)

**Run the notebook code for binary classification and complete task 4!**

# Multi-class Classification

There are latent process values at $N$ training points for all $C$ classes:

$$\mathbf{f} = (f_1(\mathbf{x}_1), \ldots, f_1(\mathbf{x}_N), f_2(\mathbf{x}_1), \ldots, f_2(\mathbf{x}_N), \ldots, f_C(\mathbf{x}_1), \ldots, f_C(\mathbf{x}_1))^\mathsf{T}$$

# Multi-class Classification

There are latent process values at $N$ training points for all $C$ classes:

$$\mathbf{f} = (f_1(\mathbf{x}_1), \ldots, f_1(\mathbf{x}_N), f_2(\mathbf{x}_1), \ldots, f_2(\mathbf{x}_N), \ldots, f_C(\mathbf{x}_1), \ldots, f_C(\mathbf{x}_1))^\mathsf{T}$$

**The prior for f is $\mathcal{N}(\mathbf{f}|0, \mathbf{C})$ with C a block diagonal.**

# Multi-class Classification

There are latent process values at $N$ training points for all $C$ classes:

$$\mathbf{f} = (f_1(\mathbf{x}_1), \ldots, f_1(\mathbf{x}_N), f_2(\mathbf{x}_1), \ldots, f_2(\mathbf{x}_N), \ldots, f_C(\mathbf{x}_1), \ldots, f_C(\mathbf{x}_1))^{\mathsf{T}}$$

**The prior for f is $\mathcal{N}(\mathbf{f}|0, \mathbf{C})$ with C a block diagonal.**

The likelihood uses a softmax function to obtain class label probabilities:

$$p(y_i = c|\mathbf{x}_i) = \frac{\exp(f_c(\mathbf{x}_i))}{\sum_{c'=1}^{C} \exp(f_{c'}(\mathbf{x}_i))},$$

# Multi-class Classification

There are latent process values at $N$ training points for all $C$ classes:

$$\mathbf{f} = (f_1(\mathbf{x}_1), \dots, f_1(\mathbf{x}_N), f_2(\mathbf{x}_1), \dots, f_2(\mathbf{x}_N), \dots, f_C(\mathbf{x}_1), \dots, f_C(\mathbf{x}_1))^\mathsf{T}$$

**The prior for $\mathbf{f}$ is $\mathcal{N}(\mathbf{f}|0, \mathbf{C})$ with $\mathbf{C}$ a block diagonal.**

The likelihood uses a softmax function to obtain class label probabilities:

$$p(y_i = c|\mathbf{x}_i) = \frac{\exp(f_c(\mathbf{x}_i))}{\sum_{c'=1}^{C} \exp(f_{c'}(\mathbf{x}_i))},$$

**The posterior is approximated using the Laplace approximation with linear cost in $C$!**

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

- **GPy**: Gaussian Processes in Python. Easy-to-use and extend. Supports multi-output GPs, different noise models and different approximate inference methods.

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

- **GPy**: Gaussian Processes in Python. Easy-to-use and extend. Supports multi-output GPs, different noise models and different approximate inference methods.
- **GPML**: Gaussian Processes in Matlab. No longer maintained. Implements the models and methods from the book "Gaussian Process for Machine learning".

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

- **GPy**: Gaussian Processes in Python. Easy-to-use and extend. Supports multi-output GPs, different noise models and different approximate inference methods.
- **GPML**: Gaussian Processes in Matlab. No longer maintained. Implements the models and methods from the book "Gaussian Process for Machine learning".
- **GPflow**: Gaussian processes in python using Tensorflow. Supports GPU acceleration. Focuses on variational inference and MCMC for approximate inference.

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

- **GPy**: Gaussian Processes in Python. Easy-to-use and extend. Supports multi-output GPs, different noise models and different approximate inference methods.
- **GPML**: Gaussian Processes in Matlab. No longer maintained. Implements the models and methods from the book "Gaussian Process for Machine learning".
- **GPflow**: Gaussian processes in python using Tensorflow. Supports GPU acceleration. Focuses on variational inference and MCMC for approximate inference.
- **GPyTorch**: Gaussian processes in python using PyTorch. Supports GPU acceleration. Also supports deep GPs.

# Software for GPs and Deep GPs

There are several packages providing implementations of GPs:

- **GPy**: Gaussian Processes in Python. Easy-to-use and extend. Supports multi-output GPs, different noise models and different approximate inference methods.
- **GPML**: Gaussian Processes in Matlab. No longer maintained. Implements the models and methods from the book "Gaussian Process for Machine learning".
- **GPflow**: Gaussian processes in python using Tensorflow. Supports GPU acceleration. Focuses on variational inference and MCMC for approximate inference.
- **GPyTorch**: Gaussian processes in python using PyTorch. Supports GPU acceleration. Also supports deep GPs.

**Deep GPs**: uses doubly stochastic variational inference and GPflow.

# Summary

1. A GP is *like* a Gaussian distribution with an **infinitely long mean vector** and an $\infty \times \infty$ **covariance matrix**.

# Summary

1. A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.

2. GPs are **non-parametric models** and become more expressive the more data we have. They are also **interpretable**!

# Summary

1. A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
2. GPs are **non-parametric models** and become more expressive the more data we have. They are also **interpretable**!
3. GPs provide predictive uncertainty that is high in regions with **no data**! This allows to know what is not known.

# Summary

1. A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
2. GPs are **non-parametric models** and become more expressive the more data we have. They are also **interpretable**!
3. GPs provide predictive uncertainty that is high in regions with **no data**! This allows to know what is not known.
4. The marginal likelihood enables finding **good hyper-parameters**, as it penalizes too simple and too complex models.

# Summary

1. A GP is *like* a Gaussian distribution with an **infinitely long mean** vector and an $\infty \times \infty$ **covariance matrix**.
2. GPs are **non-parametric models** and become more expressive the more data we have. They are also **interpretable**!
3. GPs provide predictive uncertainty that is high in regions with **no data**! This allows to know what is not known.
4. The marginal likelihood enables finding **good hyper-parameters**, as it penalizes too simple and too complex models.
5. GPs can address **classification problems** too, but **approximate inference** is needed.

# References


- Williams, C. K., Rasmussen, C. E. (2006). Gaussian processes for machine learning (Vol. 2, No. 3, p. 4). Cambridge, MA: MIT press.
- Bishop, C. M. Pattern Recognition and Machine Learning (Information Science and Statistics), Springer, 2006.
- Murphy, K. Machine Learning: a Probabilistic Perspective, The MIT Press, 2012.
- MacKay D. J. C. Information Theory, Inference & Learning Algorithms, 2003, Cambridge University Press.