Introduction to Gaussian Processes

Neil D. Lawrence

University of Siena

6th April 2011

Outline

Outline

Gaussian Distribution

Zero mean Gaussian distribution

- A multi-variate Gaussian distribution is defined by a mean and a covariance matrix.

$$\mathcal{N}\left(\mathbf{f}|\mu, \mathbf{K}\right) = \frac{1}{(2\pi)^{\frac{n}{2}}\left|\mathbf{K}\right|^{\frac{1}{2}}} \exp\left(-\frac{\left(\mathbf{f}-\mu\right)^{\top}\mathbf{K}^{-1}\left(\mathbf{f}-\mu\right)}{2}\right).$$

- We will consider the special case where the mean is zero,

$$\mathcal{N}\left(\mathbf{f}|\mathbf{0}, \mathbf{K}\right) = \frac{1}{(2\pi)^{\frac{n}{2}}\left|\mathbf{K}\right|^{\frac{1}{2}}} \exp\left(-\frac{\mathbf{f}^{\top}\mathbf{K}^{-1}\mathbf{f}}{2}\right).$$

Two Dimensional Gaussian

- Consider height, h/m and weight, w/kg.
- Could sample height from a distribution:

$$p(h) \sim \mathcal{N}(1.7, 0.0225)$$

- And similarly weight:

$$p(w) \sim \mathcal{N}(75, 36)$$

Height and Weight Models

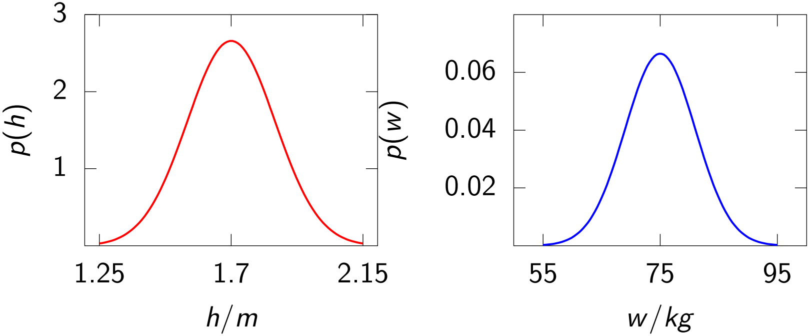
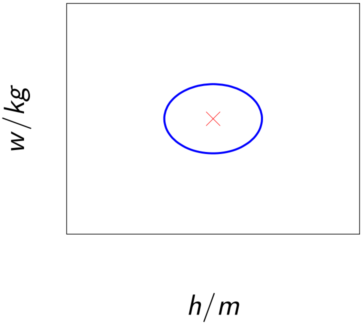
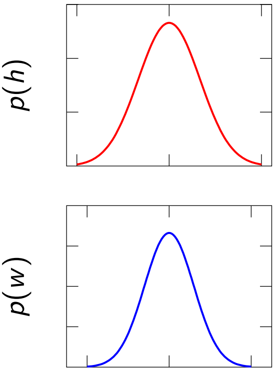


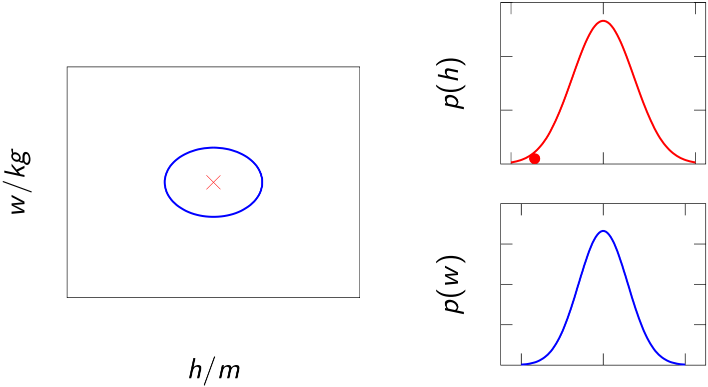
Figure: Gaussian distributions for height and weight.

Sampling Two Dimensional Variables

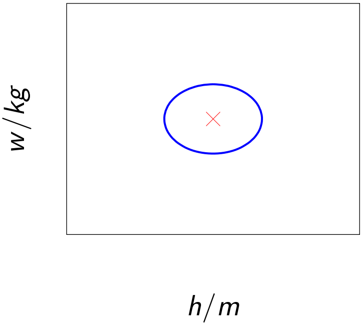


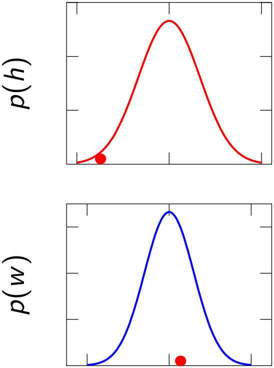


Sampling Two Dimensional Variables

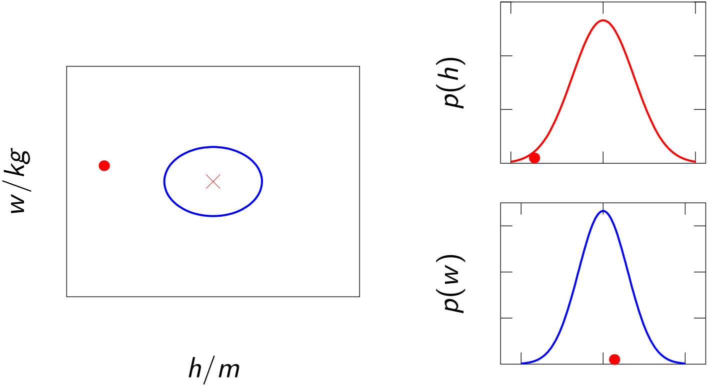


Sampling Two Dimensional Variables

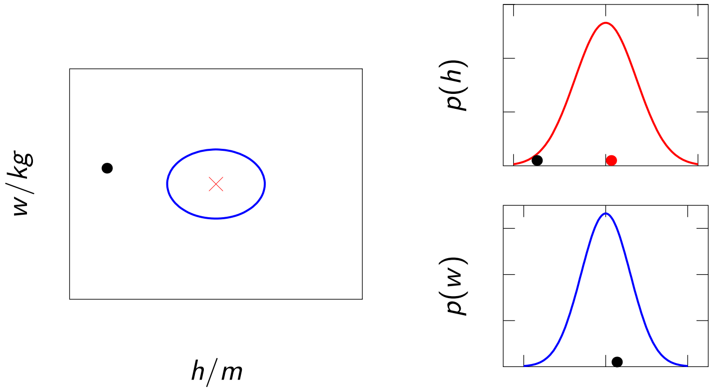




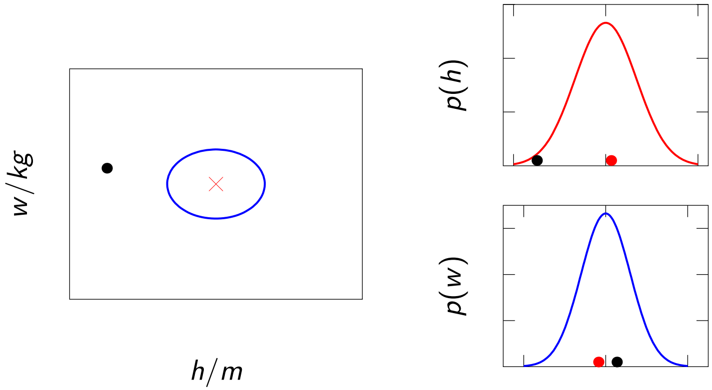
Sampling Two Dimensional Variables



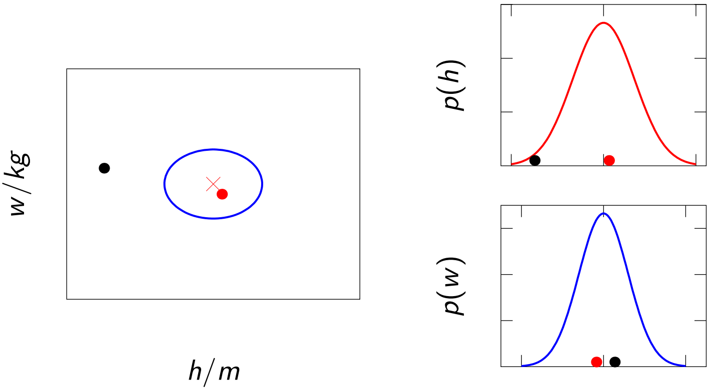
Sampling Two Dimensional Variables



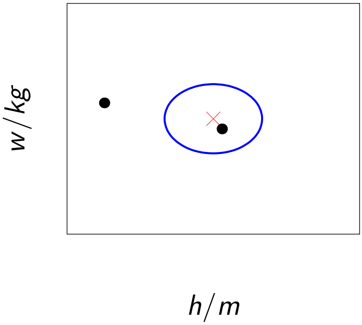
Sampling Two Dimensional Variables

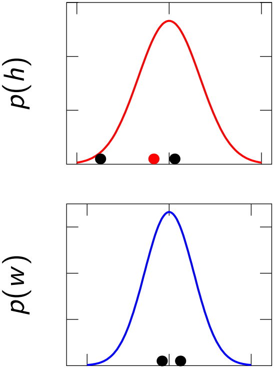


Sampling Two Dimensional Variables

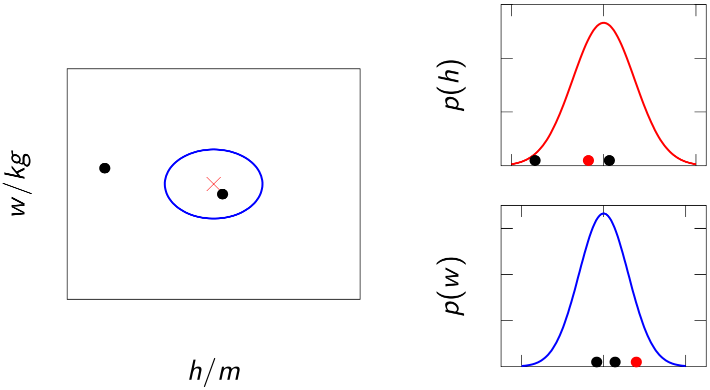


Sampling Two Dimensional Variables

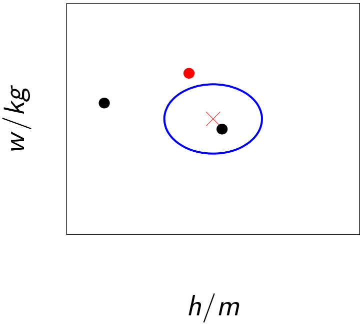


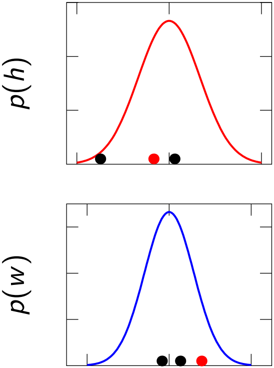


Sampling Two Dimensional Variables

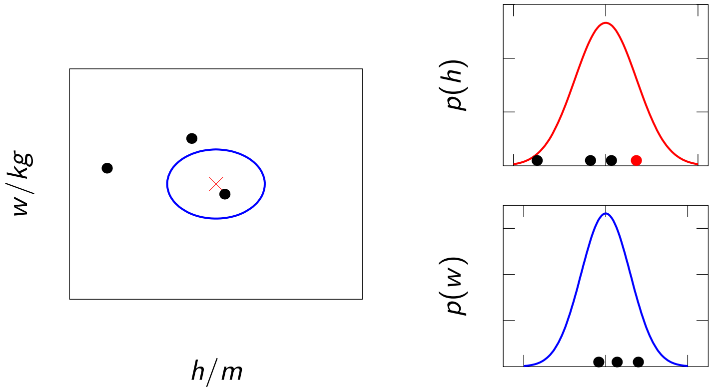


Sampling Two Dimensional Variables

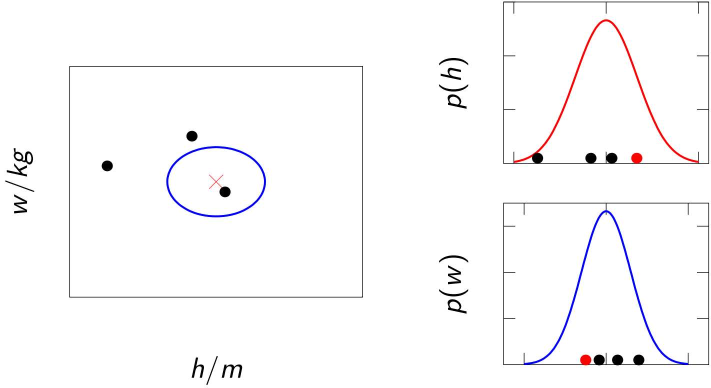




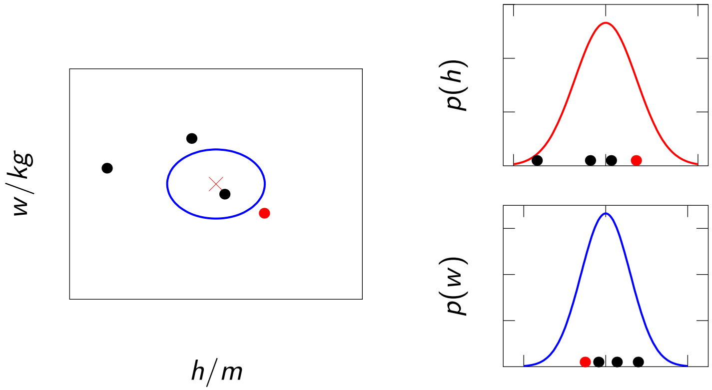
Sampling Two Dimensional Variables



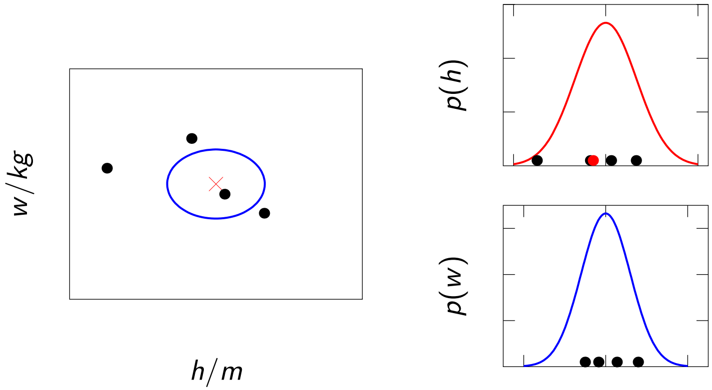
Sampling Two Dimensional Variables



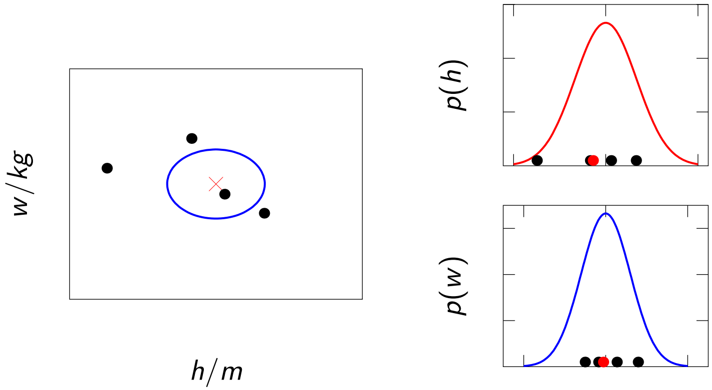
Sampling Two Dimensional Variables



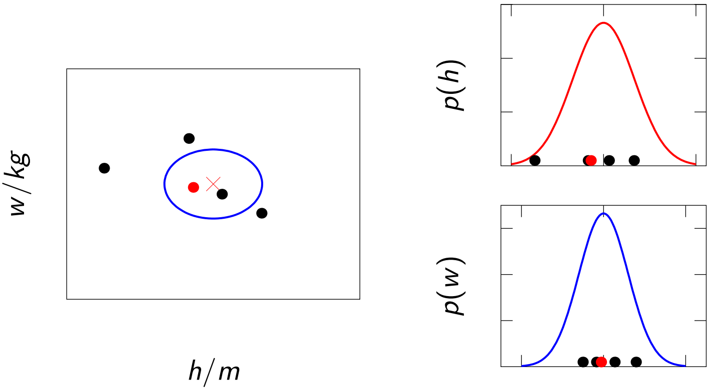
Sampling Two Dimensional Variables



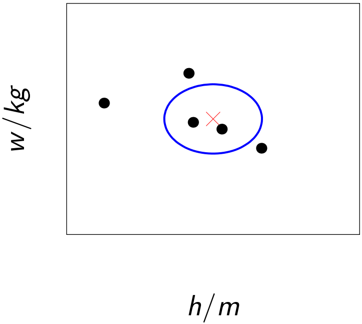
Sampling Two Dimensional Variables

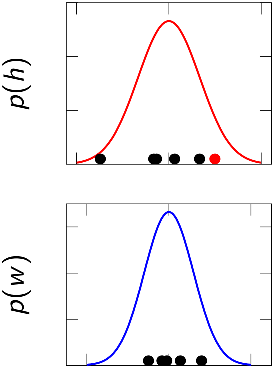


Sampling Two Dimensional Variables

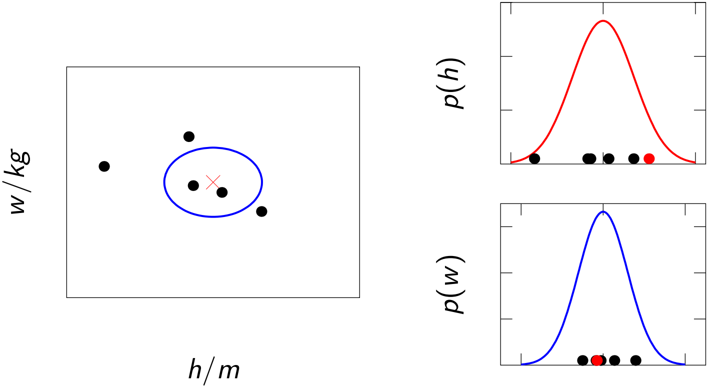


Sampling Two Dimensional Variables

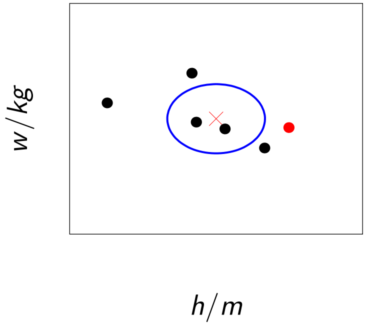


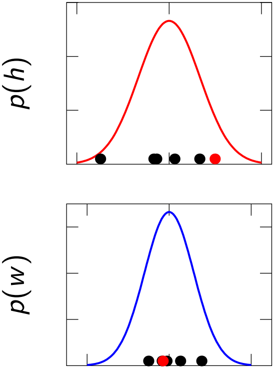


Sampling Two Dimensional Variables

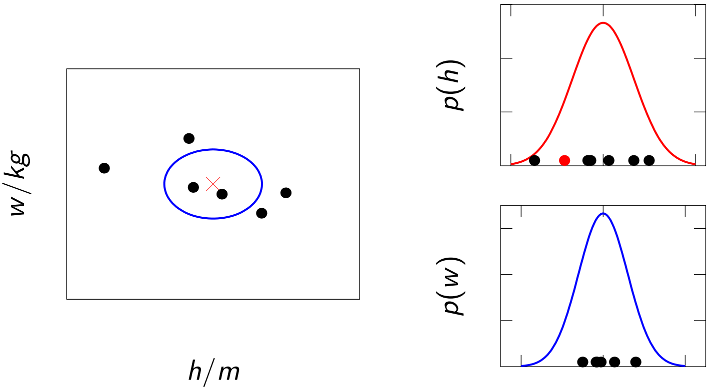


Sampling Two Dimensional Variables

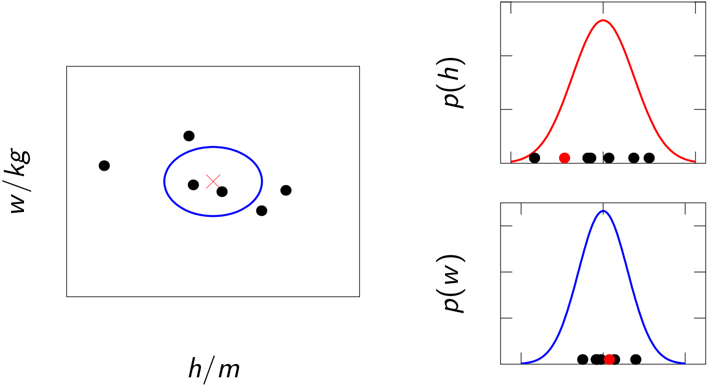




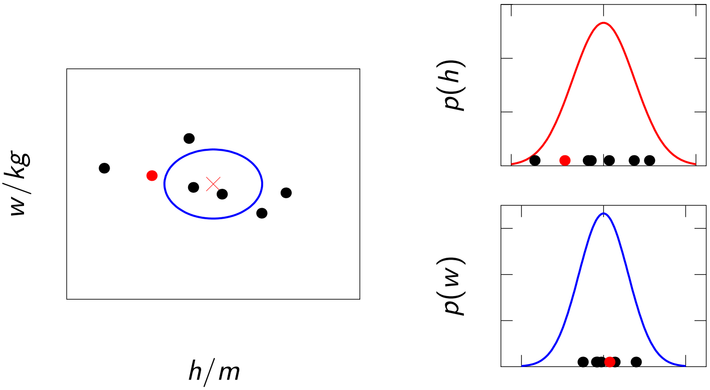
Sampling Two Dimensional Variables



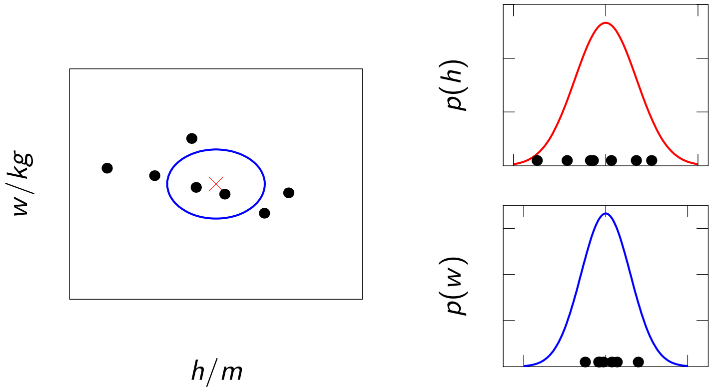
Sampling Two Dimensional Variables



Sampling Two Dimensional Variables



Sampling Two Dimensional Variables



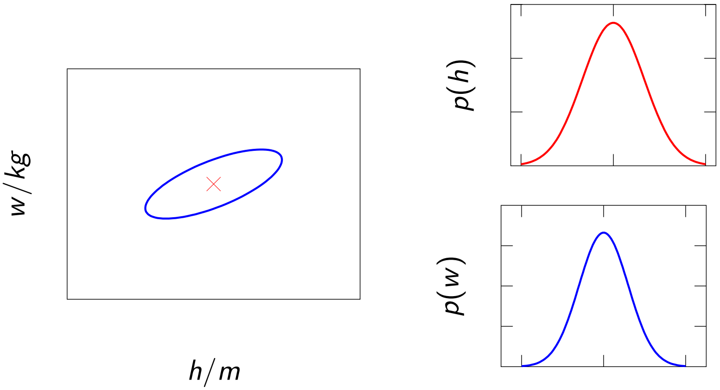
Independence Assumption

- This assumes height and weight are independent.

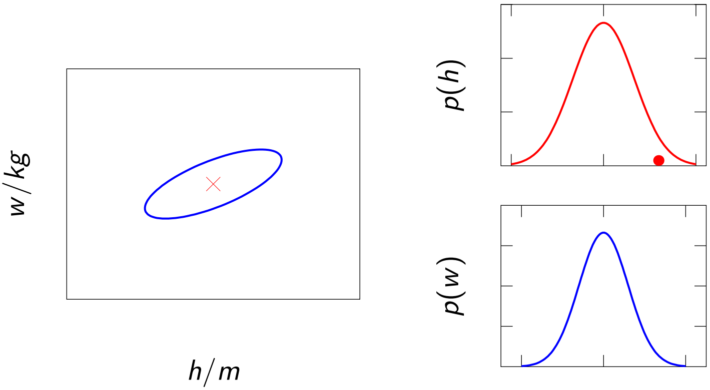
$$p(h, w) = p(h)p(w)$$

- In reality they are dependent (body mass index) $= \frac{w}{h^2}$.

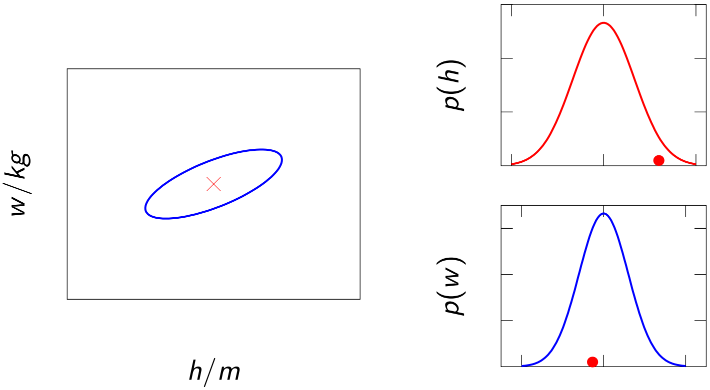
Sampling Two Dimensional Variables



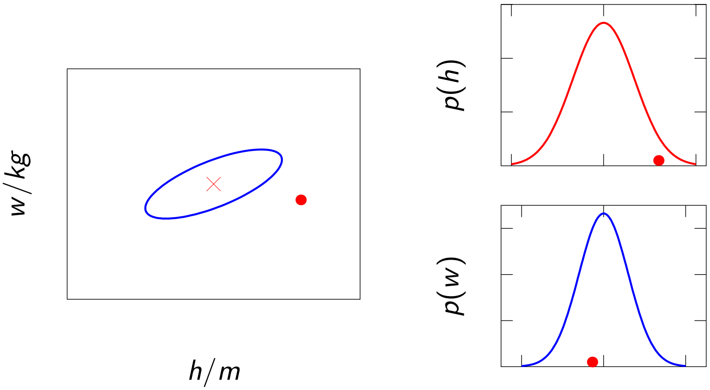
Sampling Two Dimensional Variables



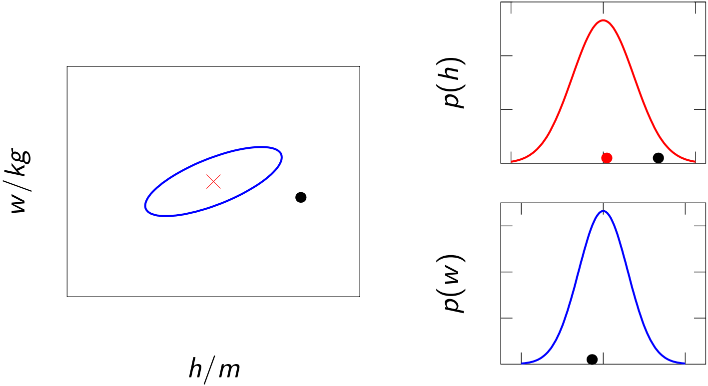
Sampling Two Dimensional Variables



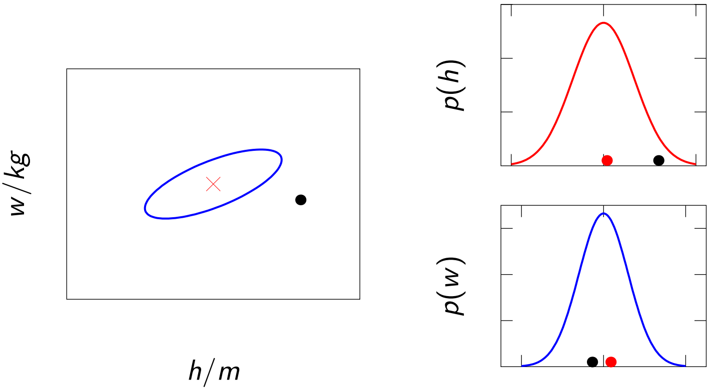
Sampling Two Dimensional Variables



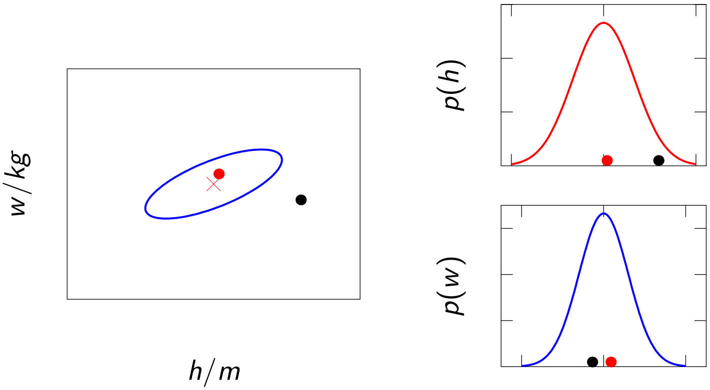
Sampling Two Dimensional Variables



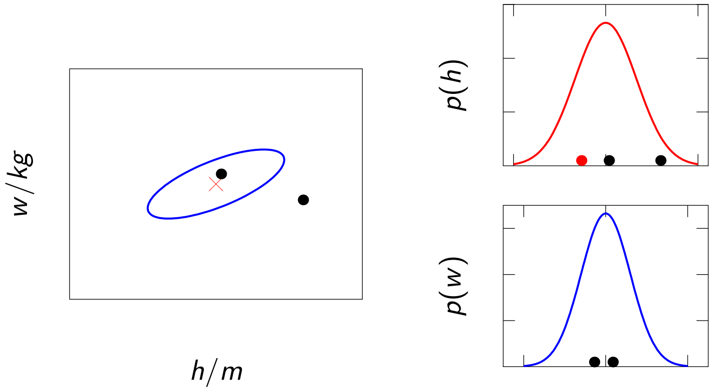
Sampling Two Dimensional Variables



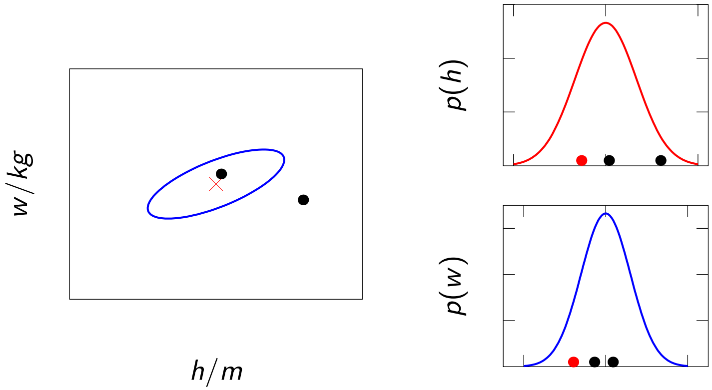
Sampling Two Dimensional Variables



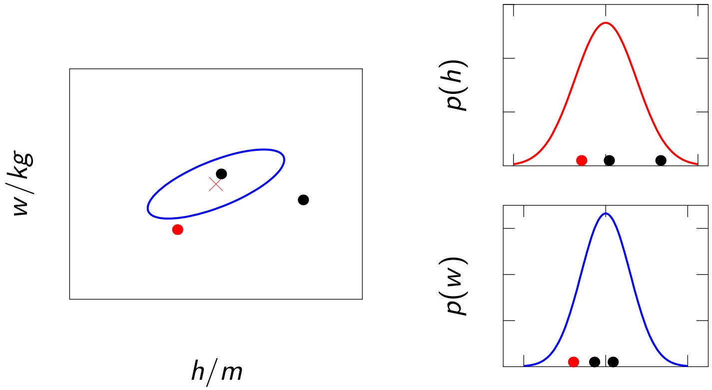
Sampling Two Dimensional Variables



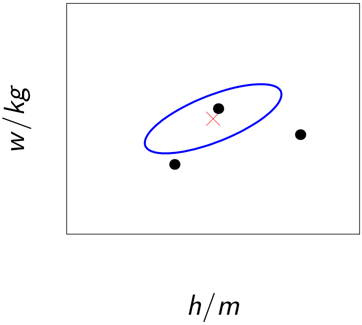
Sampling Two Dimensional Variables

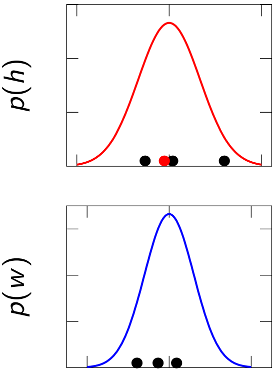


Sampling Two Dimensional Variables

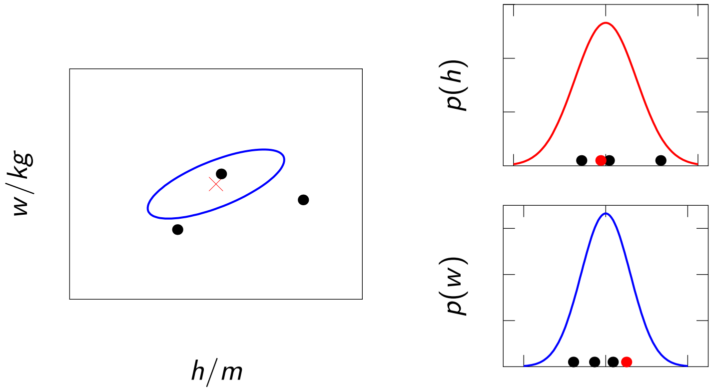


Sampling Two Dimensional Variables

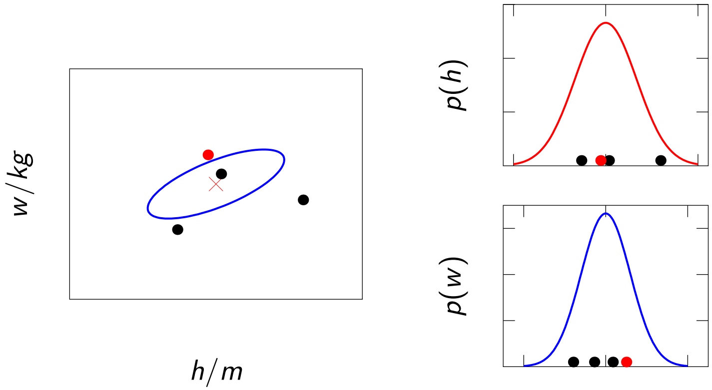




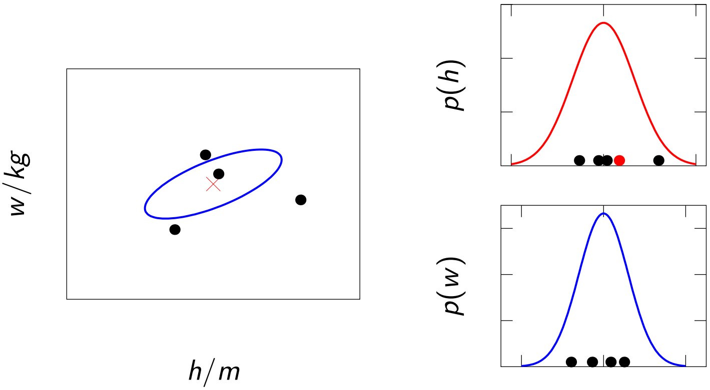
Sampling Two Dimensional Variables



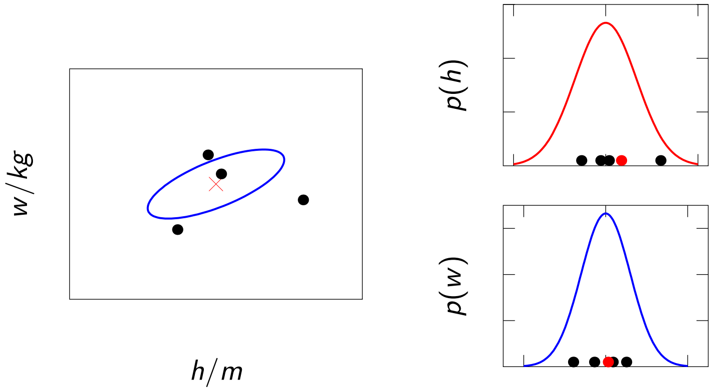
Sampling Two Dimensional Variables



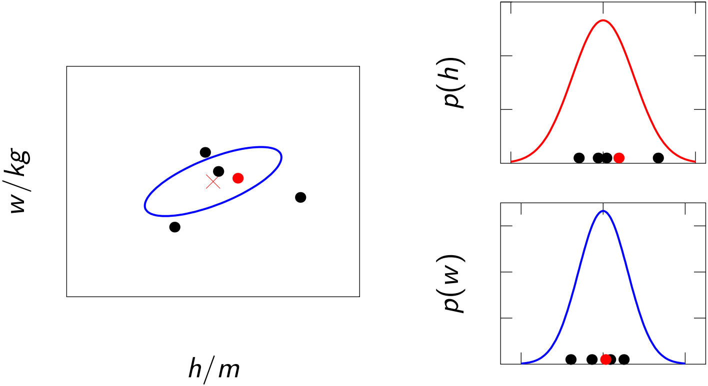
Sampling Two Dimensional Variables



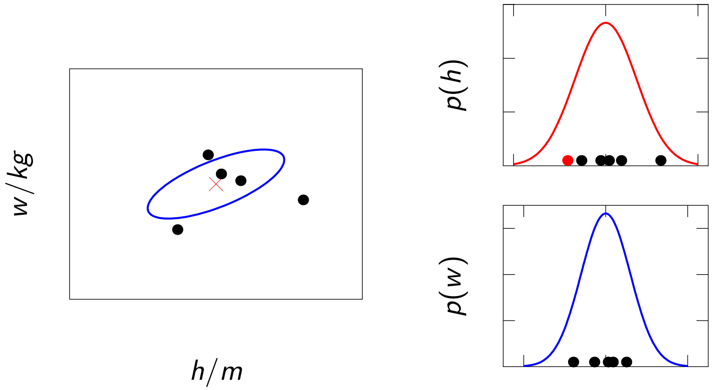
Sampling Two Dimensional Variables



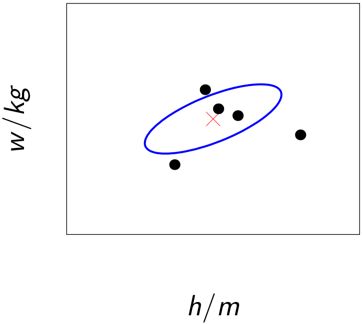
Sampling Two Dimensional Variables

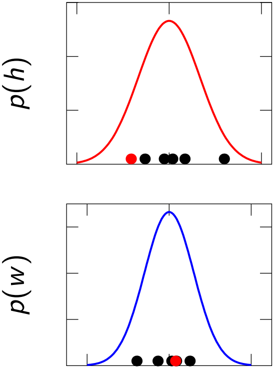


Sampling Two Dimensional Variables

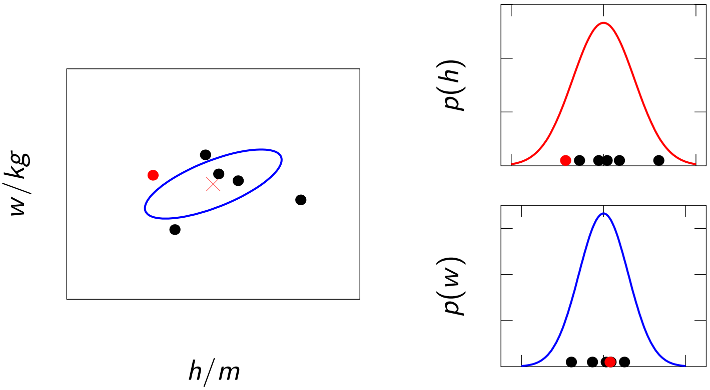


Sampling Two Dimensional Variables

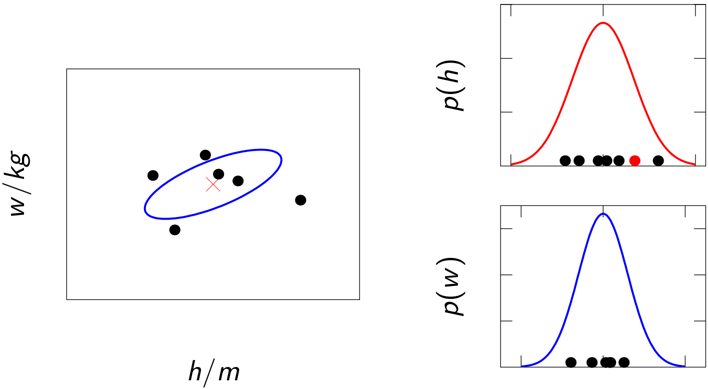




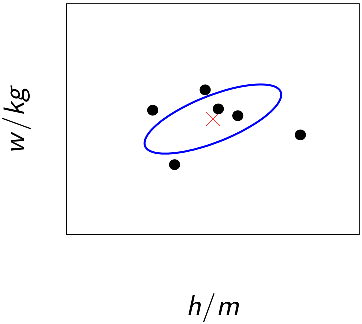
Sampling Two Dimensional Variables

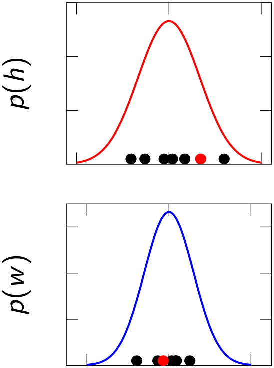


Sampling Two Dimensional Variables

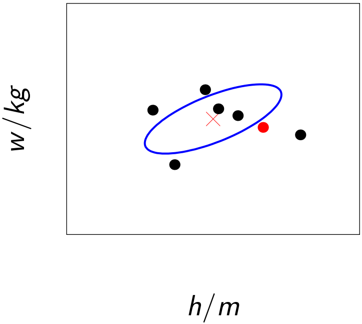


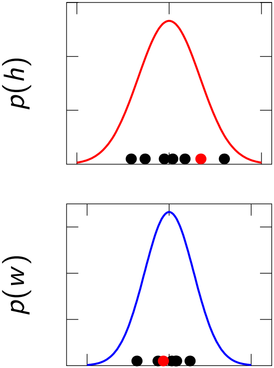
Sampling Two Dimensional Variables



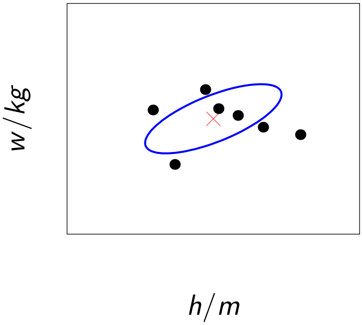


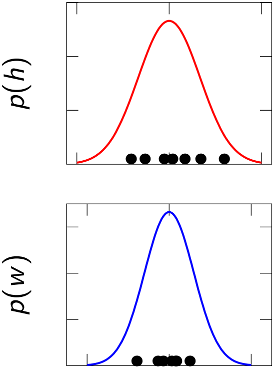
Sampling Two Dimensional Variables





Sampling Two Dimensional Variables





Correlated Gaussian

- Second Gaussian correlated.
- Form from original Gaussian by elongating one direction and rotating.
- For rotation matrix $\mathbf{R}$ and scaling matrix

$$\mathbf{L} = \begin{bmatrix} \ell_1 & 0 \\ 0 & \ell_2 \end{bmatrix}$$

  this gives a covariance matrix:

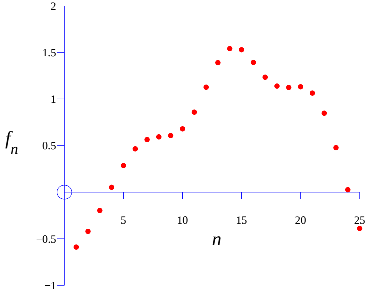
$$\mathbf{K} = \mathbf{R}\mathbf{L}^2\mathbf{R}^\top$$

Sampling a Function

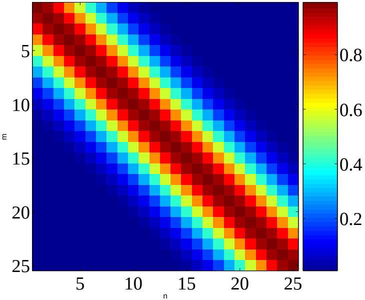
Multi-variate Gaussians

- We will consider a Gaussian with a particular structure of covariance matrix.
- Generate a single sample from this 25 dimensional Gaussian distribution, $\mathbf{f} = [f_1, f_2 \ldots f_{25}]$.
- We will plot these points against their index.

Gaussian Distribution Sample



(a) A 25 dimensional correlated random variable (values ploted against index)



(b) colormap showing correlations between dimensions

Figure: A sample from a 25 dimensional Gaussian distribution.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_i and f_j if i is near to j.
- Less correlation if i is distant from j.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points from the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_i and f_j if i is near to j.
- Less correlation if i is distant from j.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points from the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_i and f_j if i is near to j.
- Less correlation if i is distant from j.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points from the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_i and f_j if i is near to j.
- Less correlation if i is distant from j.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points from the 25.

Prediction of f_2 from f_1

```
demGpCov2D([1 2])
```

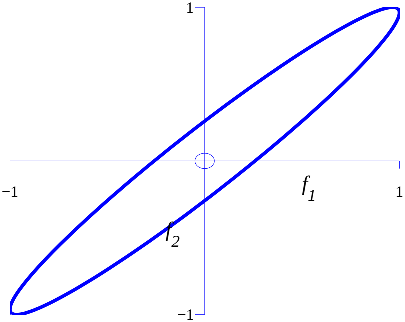


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

```
demGpCov2D([1 2])
```

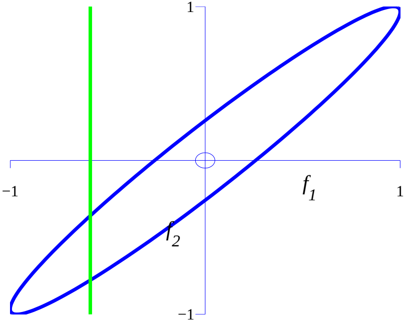


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

```
demGpCov2D([1 2])
```

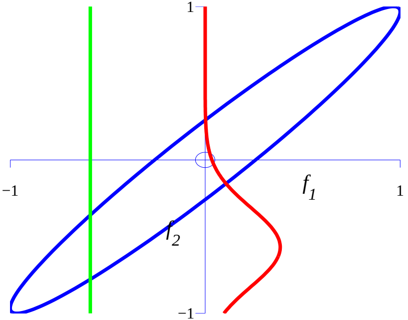


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGpCov2D([1 5])
```

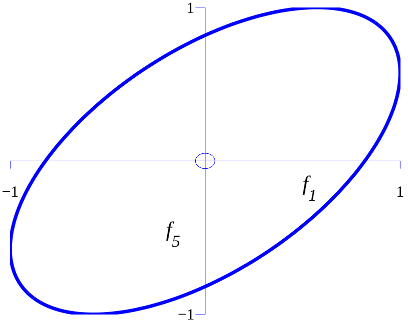


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGpCov2D([1 5])
```

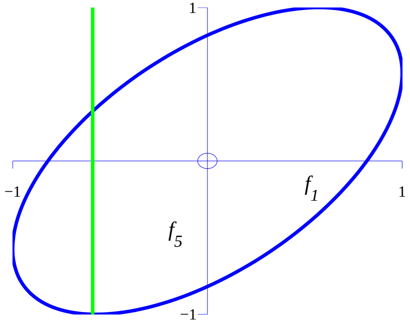


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

```
demGpCov2D([1 5])
```

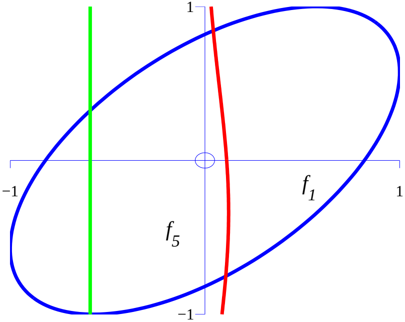
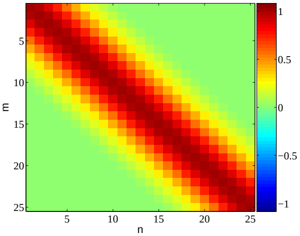


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k\left(\mathbf{x}, \mathbf{x}'\right) = \alpha \exp\left(-\frac{\left\|\mathbf{x} - \mathbf{x}'\right\|^2}{2\ell^2}\right)$$

- Covariance matrix is built using the *inputs* to the function t.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.



Outline

Basis Function Form

Radial basis functions commonly have the form

$$\phi_k(\mathbf{x}_i) = \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

- Basis function maps data into a "feature space" in which a linear sum is a non linear function.

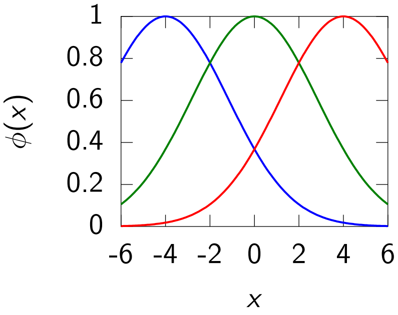


Figure: A set of radial basis functions with width $\ell = 2$ and location parameters $\boldsymbol{\mu} = [-4 \;\; 0 \;\; 4]^\top$.

Basis Function Representations

- Represent a function by a linear sum over a basis,

$$f(\mathbf{x}_{i,:};\mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_{i,:}), \quad (1)$$

- Here: M basis functions and $\phi_k(\cdot)$ is kth basis function and

$$\mathbf{w} = [w_1, \ldots, w_M]^\top .$$

- For standard linear model: $\phi_k(\mathbf{x}_{i,:}) = x_{i,k}$.

Random Functions

Functions derived using:

$$f(x) = \sum_{k=1}^{M} w_k \phi_k(x),$$

where $\mathbf{W}$ is sampled from a Gaussian density,

$$w_k \sim \mathcal{N}(0, \alpha).$$

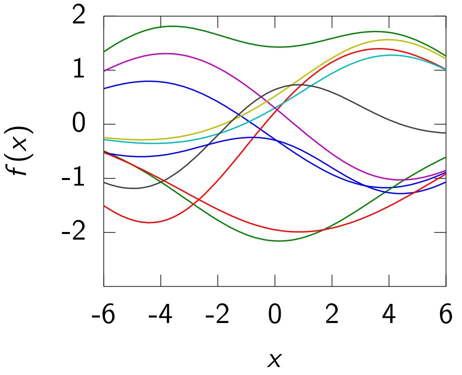


Figure: Functions sampled using the basis set from figure 5. Each line is a separate sample, generated by a weighted sum of the basis set. The weights, $\mathbf{w}$ are sampled from a Gaussian density with variance $\alpha = 1$.

Outline

Model Likelihood

- There are two components to a Bayesian probabilistic model.
  1. the likelihood
  2. the prior
- The likelihood $p(\mathbf{y}|\mathbf{x}, \mathbf{w})$ depends on the data and the parameters.
- The prior $p(\mathbf{w})$ represents our *a priori* belief about parameters.
- Compute posterior with Bayes rule:

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{x})}$$

The Likelihood

- Form likelihood by adding noise:

$$y(\mathbf{x}_i) = f(\mathbf{x}_i; \mathbf{w}) + \epsilon_i,$$

ϵ_i is the noise associated with the *i*th data point.

$$\epsilon_i \sim \mathcal{N}\left(0, \sigma^2\right),$$

- The likelihood is *not* equivalent to a loss.
- For Gaussian distributed noise

$$p\left(\mathbf{y}|\mathbf{x}, \mathbf{w}, \sigma^2\right) = \prod_{i=1}^{M} \mathcal{N}\left(y_i | f_i, \sigma^2\right),$$

- Mean of this Gaussian distributions given by $f_i = f(\mathbf{x}_i; \mathbf{w})$.

Prior and Posterior Distribution

- Prior over $\mathbf{w}$ is Gaussian with covariance matrix $\gamma'\mathbf{I}$,

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{0}, \gamma'\mathbf{I}\right).$$

- Combine prior with likelihood to get posterior distribution:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

with

$$\boldsymbol{\mu}_w = \sigma^{-2}\mathbf{C}_w\boldsymbol{\Phi}^\top\mathbf{y}$$

and

$$\mathbf{C}_w = \left[\sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{1}{\gamma'}\mathbf{I}\right]^{-1}.$$

Prior and Posterior Distribution

- Prior over $\mathbf{w}$ is Gaussian with covariance matrix $\gamma'\mathbf{I}$,

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{0}, \gamma'\mathbf{I}\right).$$

- Combine prior with likelihood to get posterior distribution:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

with

$$\boldsymbol{\mu}_w = \sigma^{-2}\mathbf{C}_w\boldsymbol{\Phi}^\top\mathbf{y}$$

and

$$\mathbf{C}_w = \left[\sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{1}{\gamma'}\mathbf{I}\right]^{-1}.$$

Prior and Posterior Distribution

- Prior over $\mathbf{w}$ is Gaussian with covariance matrix $\gamma'\mathbf{I}$,

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{0}, \gamma'\mathbf{I}\right).$$

- Combine prior with likelihood to get posterior distribution:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

with

$$\boldsymbol{\mu}_w = \sigma^{-2}\mathbf{C}_w\boldsymbol{\Phi}^\top\mathbf{y}$$

and

$$\mathbf{C}_w = \left[\sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{1}{\gamma'}\mathbf{I}\right]^{-1}.$$

Prior and Posterior Distribution

- Prior over $\mathbf{w}$ is Gaussian with covariance matrix $\gamma'\mathbf{I}$,

$$p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}|\mathbf{0}, \gamma'\mathbf{I}\right).$$

- Combine prior with likelihood to get posterior distribution:

$$p(\mathbf{w}|\mathbf{y}, \mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{w}|\boldsymbol{\mu}_w, \mathbf{C}_w\right)$$

with

$$\boldsymbol{\mu}_w = \sigma^{-2}\mathbf{C}_w\boldsymbol{\Phi}^\top\mathbf{y}$$

and

$$\mathbf{C}_w = \left[\sigma^{-2}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \frac{1}{\gamma'}\mathbf{I}\right]^{-1}.$$

Notation Aside

- Constructed a “design matrix” from our basis functions

$$\mathbf{\Phi} = [\phi_1, \ldots, \phi_M],$$

where

$$\phi_j = [\phi_j(\mathbf{x}_1), \ldots, \phi_j(\mathbf{x}_n)]^\top$$

Notation Aside

- Constructed a "design matrix" from our basis functions

$$\mathbf{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_M],$$

where

$$\boldsymbol{\phi}_j = [\phi_j(\mathbf{x}_1), \ldots, \phi_j(\mathbf{x}_n)]^\top$$

Marginal Likelihood

- Can also compute marginal likelihood of data:

$$p(\mathbf{y}|\mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}\right)$$

where

$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}.$$

- This is a joint Gaussian density across observations $\mathbf{y}$.
- If

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)$$

and

$$\epsilon \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \mathbf{I}\right)$$

then $\mathbf{y} = \mathbf{f} + \epsilon$ is

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}\right)$$

Marginal Likelihood

- Can also compute marginal likelihood of data:

$$p(\mathbf{y}|\mathbf{x}, \sigma^2) = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}\right)$$

  where

$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}.$$

- This is a joint Gaussian density across observations $\mathbf{y}$.
- If

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)$$

  and

$$\epsilon \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \mathbf{I}\right)$$

  then $\mathbf{y} = \mathbf{f} + \epsilon$ is

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}\right)$$

Marginal Likelihood

- Can also compute marginal likelihood of data:

$$p(\mathbf{y}|\mathbf{x}, \sigma^2) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

where

$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}.$$

- This is a joint Gaussian density across observations $\mathbf{y}$.
- If

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)$$

and

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

then $\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}$ is

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}\right)$$

Marginal Likelihood

- Can also compute marginal likelihood of data:

$$p(\mathbf{y}|\mathbf{x}, \sigma^2) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

  where

$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}.$$

- This is a joint Gaussian density across observations $\mathbf{y}$.
- If

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)$$

  and

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

  then $\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}$ is

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}\right)$$

Marginal Likelihood

- Can also compute marginal likelihood of data:

$$p(\mathbf{y}|\mathbf{x}, \sigma^2) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

  where

$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}.$$

- This is a joint Gaussian density across observations $\mathbf{y}$.
- If

$$\mathbf{f} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top\right)$$

  and

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

  then $\mathbf{y} = \mathbf{f} + \boldsymbol{\epsilon}$ is

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, \alpha \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \sigma^2 \mathbf{I}\right)$$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \Phi \mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- Φ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \Phi \mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- Φ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- $\boldsymbol{\Phi}$ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- $\boldsymbol{\Phi}$ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- $\boldsymbol{\Phi}$ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- $\boldsymbol{\Phi}$ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Direct Construction of Covariance Matrix

- Use matrix notation to write function,

$$f(\mathbf{x}_i; \mathbf{w}) = \sum_{k=1}^{M} w_k \phi_k(\mathbf{x}_i)$$

  computed at training data gives a vector

$$\mathbf{f} = \boldsymbol{\Phi}\mathbf{w}.$$

- $\mathbf{w}$ and $\mathbf{f}$ are only related by a inner product.
- $\boldsymbol{\Phi}$ is fixed and non-stochastic for a given training set.
- $\mathbf{f}$ is Gaussian distributed.
- it is straightforward to compute distribution for $\mathbf{f}$

Expectations

- We use $\langle \cdot \rangle$ to denote expectations under prior distributions.
- We have
$$\langle \mathbf{f} \rangle = \phi \langle \mathbf{w} \rangle .$$
- Prior mean of $\mathbf{w}$ was zero giving
$$\langle \mathbf{f} \rangle = \mathbf{0}.$$
- Prior covariance of $\mathbf{f}$ is
$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$
$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \boldsymbol{\Phi} \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \boldsymbol{\Phi}^\top,$$
giving
$$\mathbf{K} = \gamma' \boldsymbol{\Phi}\boldsymbol{\Phi}^\top.$$

Expectations

- We use $\langle \cdot \rangle$ to denote expectations under prior distributions.
- We have

$$\langle \mathbf{f} \rangle = \phi \langle \mathbf{w} \rangle .$$

- Prior mean of $\mathbf{w}$ was zero giving

$$\langle \mathbf{f} \rangle = \mathbf{0}.$$

- Prior covariance of $\mathbf{f}$ is

$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \mathbf{\Phi} \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \mathbf{\Phi}^\top,$$

giving

$$\mathbf{K} = \gamma' \mathbf{\Phi}\mathbf{\Phi}^\top.$$

Expectations

- We use $\langle \cdot \rangle$ to denote expectations under prior distributions.
- We have

$$\langle \mathbf{f} \rangle = \phi \langle \mathbf{w} \rangle .$$

- Prior mean of **w** was zero giving

$$\langle \mathbf{f} \rangle = \mathbf{0}.$$

- Prior covariance of **f** is

$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \Phi \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \Phi^\top,$$

giving

$$\mathbf{K} = \gamma' \Phi \Phi^\top .$$

Expectations

- We use $\langle \cdot \rangle$ to denote expectations under prior distributions.
- We have

$$\langle \mathbf{f} \rangle = \phi \langle \mathbf{w} \rangle .$$

- Prior mean of **w** was zero giving

$$\langle \mathbf{f} \rangle = \mathbf{0}.$$

- Prior covariance of **f** is

$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \Phi \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \Phi^\top,$$

giving

$$\mathbf{K} = \gamma' \Phi \Phi^\top.$$

Expectations

- We use $\langle \cdot \rangle$ to denote expectations under prior distributions.
- We have

$$\langle \mathbf{f} \rangle = \phi \langle \mathbf{w} \rangle .$$

- Prior mean of **w** was zero giving

$$\langle \mathbf{f} \rangle = \mathbf{0}.$$

- Prior covariance of **f** is

$$\mathbf{K} = \left\langle \mathbf{f}\mathbf{f}^\top \right\rangle - \langle \mathbf{f} \rangle \langle \mathbf{f} \rangle^\top$$

$$\left\langle \mathbf{f}\mathbf{f}^\top \right\rangle = \boldsymbol{\Phi} \left\langle \mathbf{w}\mathbf{w}^\top \right\rangle \boldsymbol{\Phi}^\top,$$

giving

$$\mathbf{K} = \gamma' \boldsymbol{\Phi} \boldsymbol{\Phi}^\top.$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \gamma' \sum_{\ell}^{M} \phi_\ell\left(\mathbf{x}_i\right) \phi_\ell\left(\mathbf{x}_j\right)$$

or in vector form

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi_:\left(\mathbf{x}_i\right)^\top \phi_:\left(\mathbf{x}_j\right),$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \gamma' \sum_{k=1}^{M} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k(\mathbf{x}_i, \mathbf{x}_j) = \gamma' \sum_{\ell}^{M} \phi_\ell(\mathbf{x}_i)\, \phi_\ell(\mathbf{x}_j)$$

  or in vector form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi_:(\mathbf{x}_i)^\top \phi_:(\mathbf{x}_j),$$

- For the radial basis used this gives

$$k(\mathbf{x}_i, \mathbf{x}_j) = \gamma' \sum_{k=1}^{M} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k(\mathbf{x}_i, \mathbf{x}_j) = \gamma' \sum_{\ell}^{M} \phi_\ell(\mathbf{x}_i) \phi_\ell(\mathbf{x}_j)$$

  or in vector form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi_:(\mathbf{x}_i)^\top \phi_:(\mathbf{x}_j),$$

- For the radial basis used this gives

$$k(\mathbf{x}_i, \mathbf{x}_j) = \gamma' \sum_{k=1}^{M} \exp\left(-\frac{|\mathbf{x}_i - \boldsymbol{\mu}_k|^2 + |\mathbf{x}_j - \boldsymbol{\mu}_k|^2}{2\ell^2}\right).$$

Covariance between Two Points

- The prior covariance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \gamma' \sum_{\ell}^{M} \phi_\ell\left(\mathbf{x}_i\right) \phi_\ell\left(\mathbf{x}_j\right)$$

  or in vector form

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi_:\left(\mathbf{x}_i\right)^\top \phi_:\left(\mathbf{x}_j\right),$$

- For the radial basis used this gives

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \gamma' \sum_{k=1}^{M} \exp\left(-\frac{\left|\mathbf{x}_i - \boldsymbol{\mu}_k\right|^2 + \left|\mathbf{x}_j - \boldsymbol{\mu}_k\right|^2}{2\ell^2}\right).$$

Selecting Number and Location of Basis

- Need to choose
  1. location of centers
  2. number of basis functions
- Consider uniform spacing over a region:

$$k\left(x_i, x_j\right) = \gamma \Delta\mu \sum_{k=1}^{M} \exp\left(-\frac{x_i^2 + x_j^2 - 2\mu_k\left(x_i + x_j\right) + 2\mu_k^2}{2\ell^2}\right),$$

Selecting Number and Location of Basis

- Need to choose
  1. location of centers
  2. number of basis functions
- Consider uniform spacing over a region:

$$k\left(x_i, x_j\right) = \gamma \Delta\mu \sum_{k=1}^{M} \exp\left(-\frac{x_i^2 + x_j^2 - 2\mu_k\left(x_i + x_j\right) + 2\mu_k^2}{2\ell^2}\right),$$

Selecting Number and Location of Basis

- Need to choose
  1. location of centers
  2. number of basis functions
- Consider uniform spacing over a region:

$$k\left(x_i, x_j\right) = \gamma \Delta\mu \sum_{k=1}^{M} \exp\left(-\frac{x_i^2 + x_j^2 - 2\mu_k\left(x_i + x_j\right) + 2\mu_k^2}{2\ell^2}\right),$$

Selecting Number and Location of Basis

- Need to choose
  1. location of centers
  2. number of basis functions
- Consider uniform spacing over a region:

$$k\left(x_i, x_j\right) = \gamma \Delta\mu \sum_{k=1}^{M} \exp\left(-\frac{x_i^2 + x_j^2 - 2\mu_k\left(x_i + x_j\right) + 2\mu_k^2}{2\ell^2}\right),$$

Uniform Basis Functions

- Set each center location to

$$\mu_k = a + \Delta\mu \cdot (k - 1).$$

- Specify the bases in terms of their indices,

$$k(x_i, x_j) = \gamma\Delta\mu \sum_{k=1}^{M} \exp\left(-\frac{x_i^2 + x_j^2}{2\ell^2} - \frac{2(a + \Delta\mu \cdot k)(x_i + x_j) + 2(a + \Delta\mu \cdot k)^2}{2\ell^2} \right).$$

Uniform Basis Functions

- Set each center location to

$$\mu_k = a + \Delta\mu \cdot (k - 1).$$

- Specify the bases in terms of their indices,

$$\begin{aligned} k\left(x_i, x_j\right) = &\gamma \Delta\mu \sum_{k=1}^{M} \exp\Bigg(- \frac{x_i^2 + x_j^2}{2\ell^2} \\ &- \frac{2\left(a + \Delta\mu \cdot k\right)\left(x_i + x_j\right) + 2\left(a + \Delta\mu \cdot k\right)^2}{2\ell^2}\Bigg). \end{aligned}$$

Infinite Basis Functions

- Take $\mu_0 = a$ and $\mu_M = b$ so $b = a + \Delta\mu \cdot (M - 1)$.
- Take limit as $\Delta\mu \to 0$ so $M \to \infty$

$$k(x_i, x_j) = \gamma \int_a^b \exp\Bigg(- \frac{x_i^2 + x_j^2}{2\ell^2} + \frac{2\left(\mu - \frac{1}{2}\left(x_i + x_j\right)\right)^2 - \frac{1}{2}\left(x_i + x_j\right)^2}{2\ell^2}\Bigg) \mathrm{d}\mu,$$

where we have used $k \cdot \Delta\mu \to \mu$.

Infinite Basis Functions

- Take $\mu_0 = a$ and $\mu_M = b$ so $b = a + \Delta\mu \cdot (M - 1)$.
- Take limit as $\Delta\mu \to 0$ so $M \to \infty$

$$k(x_i, x_j) = \gamma \int_a^b \exp\left(-\frac{x_i^2 + x_j^2}{2\ell^2} + \frac{2\left(\mu - \frac{1}{2}\left(x_i + x_j\right)\right)^2 - \frac{1}{2}\left(x_i + x_j\right)^2}{2\ell^2} \right) \mathrm{d}\mu,$$

where we have used $k \cdot \Delta\mu \to \mu$.

Infinite Basis Functions

- Take $\mu_0 = a$ and $\mu_M = b$ so $b = a + \Delta\mu \cdot (M - 1)$.
- Take limit as $\Delta\mu \to 0$ so $M \to \infty$

$$k(x_i, x_j) = \gamma \int_a^b \exp\left(-\frac{x_i^2 + x_j^2}{2\ell^2} + \frac{2\left(\mu - \frac{1}{2}\left(x_i + x_j\right)\right)^2 - \frac{1}{2}\left(x_i + x_j\right)^2}{2\ell^2}\right) \mathrm{d}\mu,$$

where we have used $k \cdot \Delta\mu \to \mu$.

Infinite Basis Functions

- Take $\mu_0 = a$ and $\mu_M = b$ so $b = a + \Delta\mu \cdot (M-1)$.
- Take limit as $\Delta\mu \to 0$ so $M \to \infty$

$$
\begin{aligned}
k(x_i, x_j) =& \gamma \int_a^b \exp\Bigg(- \frac{x_i^2 + x_j^2}{2\ell^2} \\
&+ \frac{2\left(\mu - \frac{1}{2}\left(x_i + x_j\right)\right)^2 - \frac{1}{2}\left(x_i + x_j\right)^2}{2\ell^2}\Bigg) \mathrm{d}\mu,
\end{aligned}
$$

where we have used $k \cdot \Delta\mu \to \mu$.

Result

- Performing the integration leads to

$$k(x_i,x_j) = \gamma\frac{\sqrt{\pi\ell^2}}{2}\exp\left(-\frac{(x_i-x_j)^2}{4\ell^2}\right)$$
$$\times\left[\operatorname{erf}\left(\frac{\left(b-\frac{1}{2}(x_i+x_j)\right)}{\ell}\right)-\operatorname{erf}\left(\frac{\left(a-\frac{1}{2}(x_i+x_j)\right)}{\ell}\right)\right],$$

- Now take limit as $a\to-\infty$ and $b\to\infty$

$$k(x_i,x_j) = \alpha\exp\left(-\frac{(x_i-x_j)^2}{4\ell^2}\right).$$

where $\alpha = \gamma\sqrt{\pi\ell^2}$.

Result

- Performing the integration leads to

$$k(x_i,x_j) = \gamma \frac{\sqrt{\pi\ell^2}}{2} \exp\left(-\frac{(x_i - x_j)^2}{4\ell^2}\right) \times \left[\operatorname{erf}\left(\frac{\left(b - \frac{1}{2}(x_i + x_j)\right)}{\ell}\right) - \operatorname{erf}\left(\frac{\left(a - \frac{1}{2}(x_i + x_j)\right)}{\ell}\right)\right],$$

- Now take limit as $a \to -\infty$ and $b \to \infty$

$$k(x_i, x_j) = \alpha \exp\left(-\frac{(x_i - x_j)^2}{4\ell^2}\right).$$

where $\alpha = \gamma\sqrt{\pi\ell^2}$.

Result

- Performing the integration leads to

$$k(x_i,x_j) = \gamma \frac{\sqrt{\pi\ell^2}}{2} \exp\left(-\frac{(x_i - x_j)^2}{4\ell^2}\right)$$
$$\times \left[\text{erf}\left(\frac{\left(b - \frac{1}{2}(x_i + x_j)\right)}{\ell}\right) - \text{erf}\left(\frac{\left(a - \frac{1}{2}(x_i + x_j)\right)}{\ell}\right)\right],$$

- Now take limit as $a \to -\infty$ and $b \to \infty$

$$k(x_i, x_j) = \alpha \exp\left(-\frac{(x_i - x_j)^2}{4\ell^2}\right).$$

where $\alpha = \gamma\sqrt{\pi\ell^2}$.

Infinite Feature Space

- A RBF model with infinite basis functions is a Gaussian process.
- The covariance function is the exponentiated quadratic.
- **Note:** The functional form for the covariance function and basis functions are similar.
  - this is a special case,
  - in general they are very different
- Similar results can obtained for multi-dimensional input networks Williams (1998).

Infinite Feature Space

- A RBF model with infinite basis functions is a Gaussian process.
- The covariance function is the exponentiated quadratic.
- **Note:** The functional form for the covariance function and basis functions are similar.
  - this is a special case,
  - in general they are very different
- Similar results can obtained for multi-dimensional input networks Williams (1998).

Infinite Feature Space

- A RBF model with infinite basis functions is a Gaussian process.
- The covariance function is the exponentiated quadratic.
- **Note:** The functional form for the covariance function and basis functions are similar.
  - this is a special case,
  - in general they are very different
- Similar results can obtained for multi-dimensional input networks Williams (1998).

Infinite Feature Space

- A RBF model with infinite basis functions is a Gaussian process.
- The covariance function is the exponentiated quadratic.
- **Note:** The functional form for the covariance function and basis functions are similar.
  - this is a special case,
  - in general they are very different
- Similar results can obtained for multi-dimensional input networks Williams (1998).

Nonparametric Gaussian Processes

- This work takes us from parametric to non-parametric.
- The limit implies infinite dimensional $\mathbf{w}$.
- Gaussian processes are generally non-parametric: combine data with covariance function to get model.
- This representation *cannot* be summarized by a parameter vector of a fixed size.

The Parametric Bottleneck

- Parametric models have a representation that does not respond to increasing training set size.
- Bayesian posterior distributions over parameters contain the information about the training data.
  - Use Bayes' rule from training data, $p(\mathbf{w}|\mathbf{y}, \mathbf{x})$,
  - Make predictions on test data

$$p(y_*|\mathbf{x}_*, \mathbf{y}, \mathbf{x}) = \int p(y_*|\mathbf{w}, \mathbf{x}_*) \, p(\mathbf{w}|\mathbf{y}, \mathbf{x}) \mathrm{d}\mathbf{w}) \, .$$

- $\mathbf{w}$ becomes a bottleneck for information about the training set to pass to the test set.
- Solution: increase M so that the bottleneck is so large that it no longer presents a problem.
- How big is big enough for M? Non-parametrics says $M \to \infty$.

The Parametric Bottleneck

- Now no longer possible to manipulate the model through the standard parametric form given in (1).
- However, it *is* possible to express *parametric* as GPs:

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi_{:}\left(\mathbf{x}_i\right)^\top \phi_{:}\left(\mathbf{x}_j\right).$$

- These are known as degenerate covariance matrices.
- Their rank is at most M, non-parametric models have full rank covariance matrices.
- Most well known is the "linear kernel", $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

The Parametric Bottleneck

- Now no longer possible to manipulate the model through the standard parametric form given in (1).
- However, it *is* possible to express *parametric* as GPs:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi_:(\mathbf{x}_i)^\top \phi_:(\mathbf{x}_j).$$

- These are known as degenerate covariance matrices.
- Their rank is at most M, non-parametric models have full rank covariance matrices.
- Most well known is the "linear kernel", $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

The Parametric Bottleneck

- Now no longer possible to manipulate the model through the standard parametric form given in (1).
- However, it *is* possible to express *parametric* as GPs:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi_:(\mathbf{x}_i)^\top \phi_:(\mathbf{x}_j).$$

- These are known as degenerate covariance matrices.
- Their rank is at most M, non-parametric models have full rank covariance matrices.
- Most well known is the "linear kernel", $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

The Parametric Bottleneck

- Now no longer possible to manipulate the model through the standard parametric form given in (1).
- However, it *is* possible to express *parametric* as GPs:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi_:(\mathbf{x}_i)^\top \phi_:(\mathbf{x}_j).$$

- These are known as degenerate covariance matrices.
- Their rank is at most M, non-parametric models have full rank covariance matrices.
- Most well known is the "linear kernel", $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

The Parametric Bottleneck

- Now no longer possible to manipulate the model through the standard parametric form given in (1).
- However, it *is* possible to express *parametric* as GPs:

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi_:\left(\mathbf{x}_i\right)^\top \phi_:\left(\mathbf{x}_j\right).$$

- These are known as degenerate covariance matrices.
- Their rank is at most M, non-parametric models have full rank covariance matrices.
- Most well known is the "linear kernel", $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

Making Predictions

- For non-parametrics prediction at new points $\mathbf{f}_*$ is made by conditioning on $\mathbf{f}$ in the joint distribution.
- In GPs this involves combining the training data with the covariance function and the mean function.
- Parametric is a special case when conditional prediction can be summarized in a *fixed* number of parameters.
- Complexity of parametric model remains fixed regardless of the size of our training data set.
- For a non-parametric model the required number of parameters grows with the size of the training data.

Making Predictions

- For non-parametrics prediction at new points $\mathbf{f}_*$ is made by conditioning on $\mathbf{f}$ in the joint distribution.
- In GPs this involves combining the training data with the covariance function and the mean function.
- Parametric is a special case when conditional prediction can be summarized in a *fixed* number of parameters.
- Complexity of parametric model remains fixed regardless of the size of our training data set.
- For a non-parametric model the required number of parameters grows with the size of the training data.

Making Predictions

- For non-parametrics prediction at new points $\mathbf{f}_*$ is made by conditioning on $\mathbf{f}$ in the joint distribution.
- In GPs this involves combining the training data with the covariance function and the mean function.
- Parametric is a special case when conditional prediction can be summarized in a *fixed* number of parameters.
- Complexity of parametric model remains fixed regardless of the size of our training data set.
- For a non-parametric model the required number of parameters grows with the size of the training data.

Making Predictions

- For non-parametrics prediction at new points $\mathbf{f}_*$ is made by conditioning on $\mathbf{f}$ in the joint distribution.
- In GPs this involves combining the training data with the covariance function and the mean function.
- Parametric is a special case when conditional prediction can be summarized in a *fixed* number of parameters.
- Complexity of parametric model remains fixed regardless of the size of our training data set.
- For a non-parametric model the required number of parameters grows with the size of the training data.

Making Predictions

- For non-parametrics prediction at new points $\mathbf{f}_*$ is made by conditioning on $\mathbf{f}$ in the joint distribution.
- In GPs this involves combining the training data with the covariance function and the mean function.
- Parametric is a special case when conditional prediction can be summarized in a *fixed* number of parameters.
- Complexity of parametric model remains fixed regardless of the size of our training data set.
- For a non-parametric model the required number of parameters grows with the size of the training data.

Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

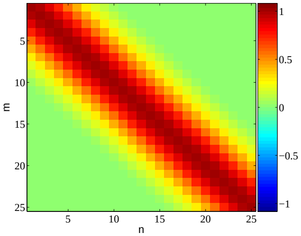
Covariance Functions and Mercer Kernels

- Mercer Kernels and Covariance Functions are similar.
- the kernel perspective does not make a probabilistic interpretation of the covariance function.
- Algorithms can be simpler, but probabilistic interpretation is crucial for kernel parameter optimization.

Exponentiated Quadratic Kernel Function (RBF, Squared Exponential, Gaussian)

$$k\left(\mathbf{x}, \mathbf{x}'\right) = \alpha \exp\left(-\frac{\left\|\mathbf{x} - \mathbf{x}'\right\|^2}{2\ell^2}\right)$$

- Covariance matrix is built using the *inputs* to the function t.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.



Covariance Samples

`demCovFuncSample`

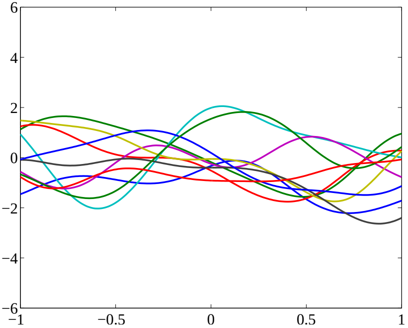


Figure: Exponentiated quadratic kernel with $\ell = 0.3$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

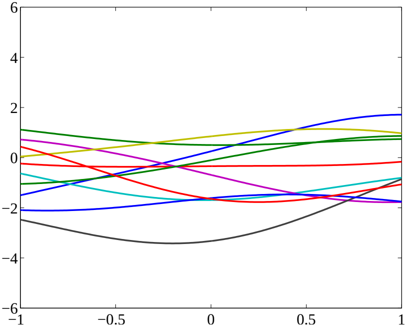


Figure: Exponentiated quadratic kernel with $\ell = 1$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

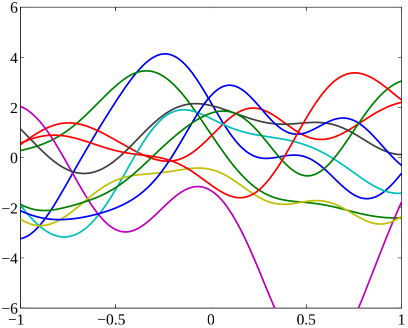


Figure: Exponentiated quadratic kernel with $\ell = 0.3$, $\alpha = 4$

Covariance Samples

`demCovFuncSample`

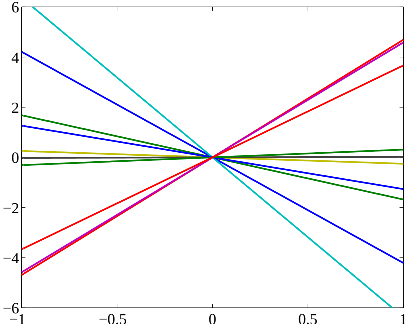


Figure: Linear covariance function, $\alpha = 16$.

Covariance Samples

`demCovFuncSample`

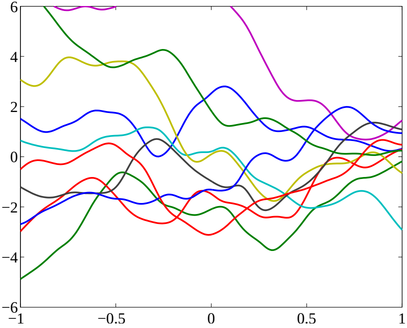


Figure: MLP covariance function, $\sigma_w^2 = 100$, $\sigma_b^2 = 100$, $\alpha = 8$.

Covariance Samples

`demCovFuncSample`

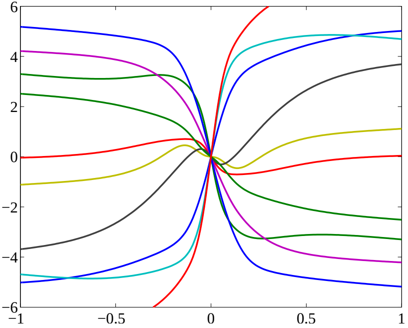


Figure: MLP covariance function, $\sigma_w^2 = 100$, $\sigma_b^2 = 0$, $\alpha = 8$.

Covariance Samples

`demCovFuncSample`

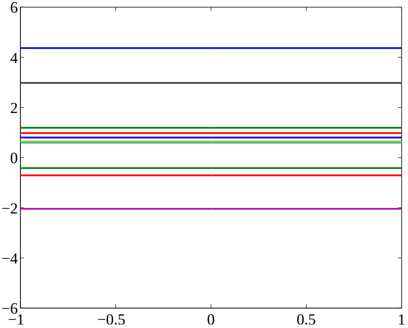


Figure: Bias term, $\alpha = 4$

Covariance Samples

`demCovFuncSample`

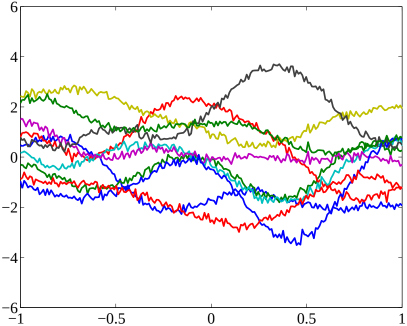


Figure: Exponentiated quadratic $\ell = 0.3$, $\alpha = 1$ plus bias term with $\alpha = 1$ plus white noise with $\alpha = 0.01$.

Covariance Samples

`demCovFuncSample`

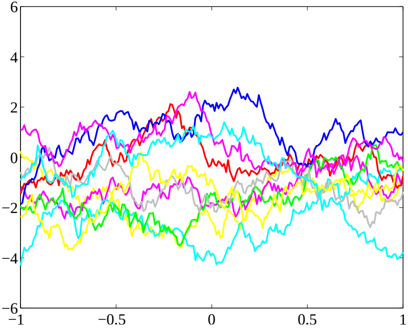


Figure: Ornstein-Uhlenbeck (stationary Gauss-Markov) covariance function $\ell = 1$, $\alpha = 4$.

Gaussian Process Interpolation

`demInterpolation`

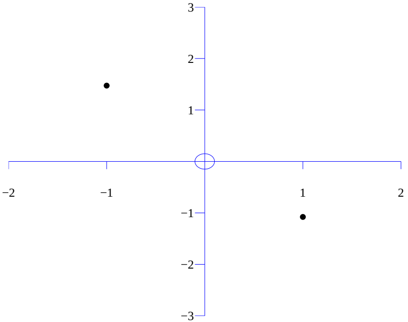


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

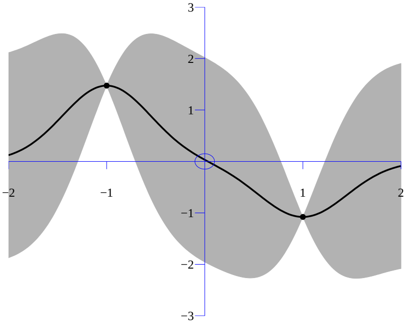


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

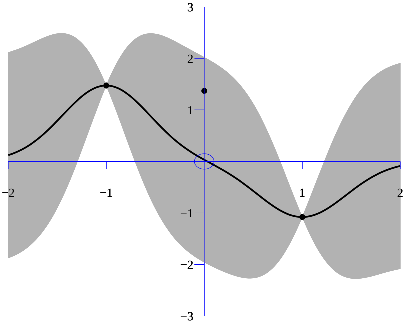


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

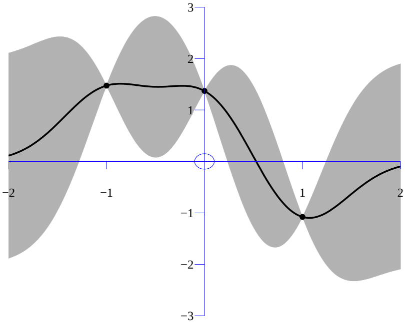


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

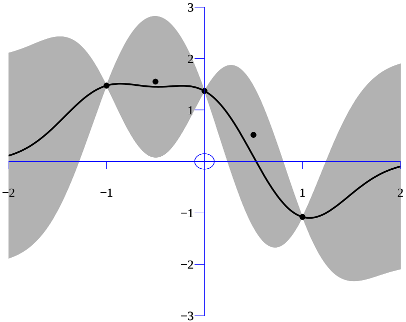


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

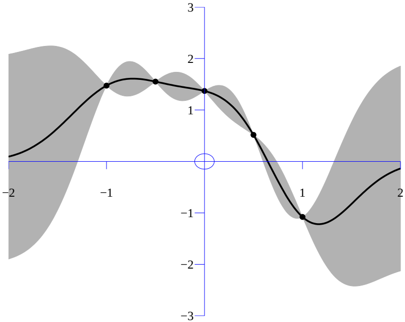


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

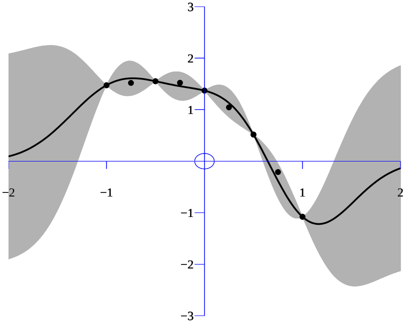


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

`demInterpolation`

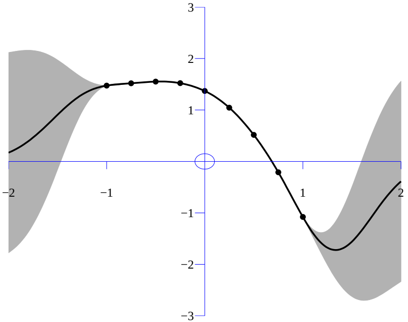


Figure: Real example: BACCO (see *e.g.* (Oakley and O'Hagan, 2002)). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Noise Models

Graph of a GP

- Relates input variables, $\mathbf{x}$, to vector, $\mathbf{y}$, through $\mathbf{f}$ given kernel parameters $\boldsymbol{\theta}$.
- Plate notation indicates independence of $y_i|f_i$.
- Noise model, $p\left(y_i|f_i\right)$ can take several forms.
- Simplest is Gaussian noise.

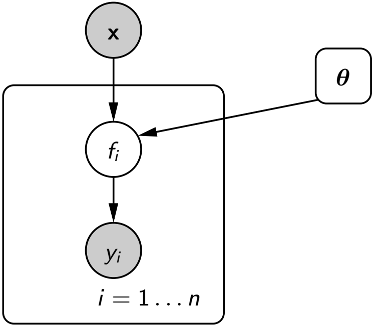


Figure: The Gaussian process depicted graphically.

Gaussian Noise

- Gaussian noise model,

$$p\left(y_i|f_i\right) = \mathcal{N}\left(y_i|f_i, \sigma^2\right)$$

where σ^2 is the variance of the noise.

- Equivalent to a covariance function of the form

$$k(\mathbf{x}_i, \mathbf{x}_j) = \delta_{i,j}\sigma^2$$

where $\delta_{i,j}$ is the Kronecker delta function.

- Additive nature of Gaussians means we can simply add this term to existing covariance matrices.

Gaussian Process Regression

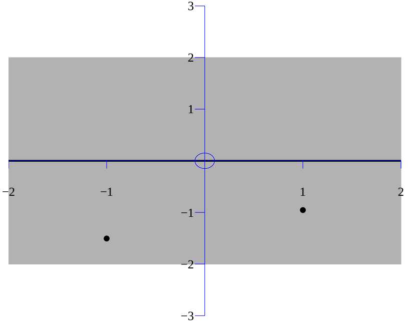


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

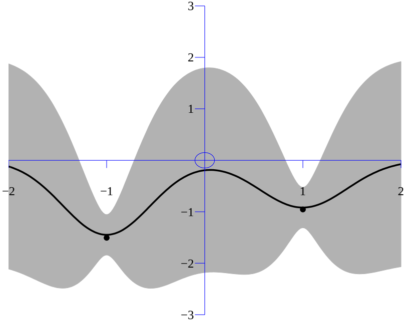


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

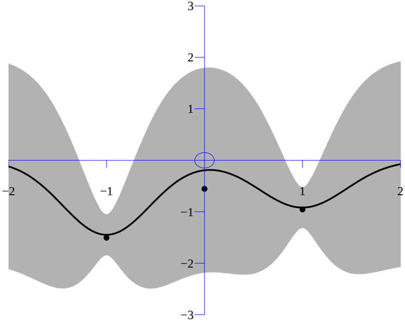


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

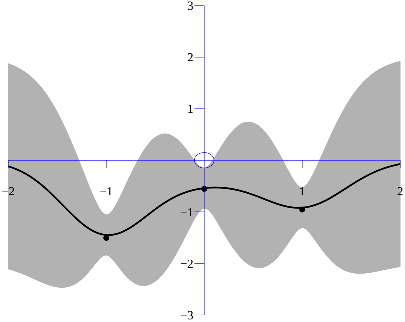


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

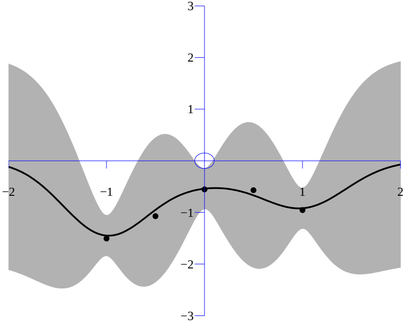


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

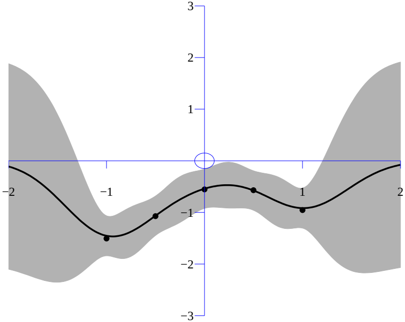


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

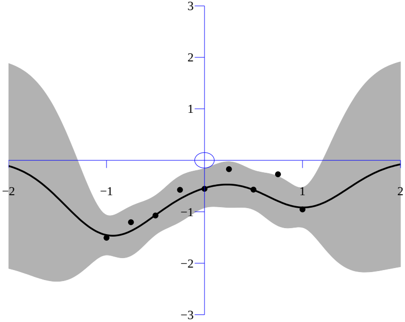


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

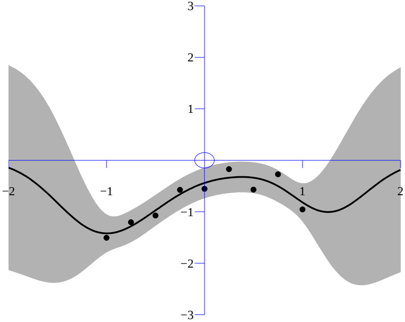
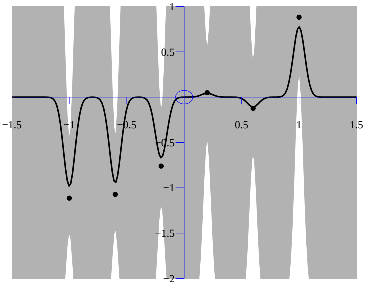
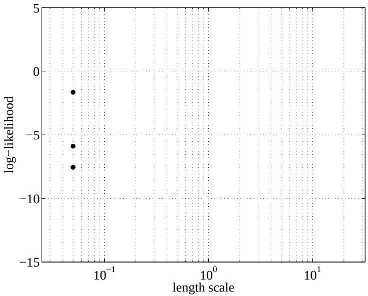


Figure: Examples include WiFi localization, C14 callibration curve.

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

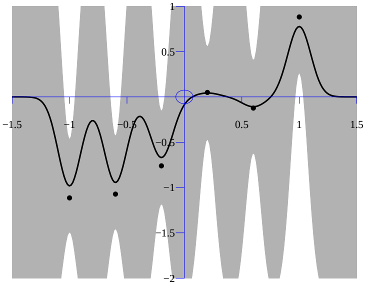


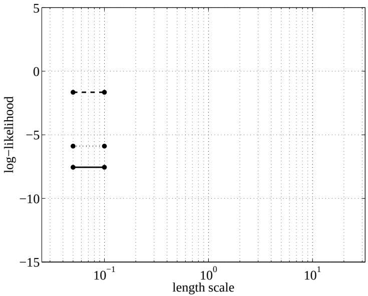


$$\log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}\right) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

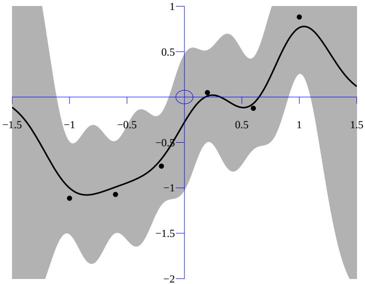


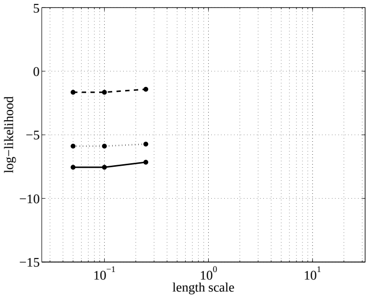


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

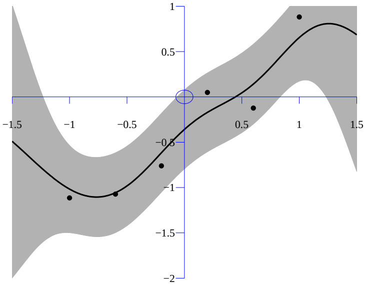


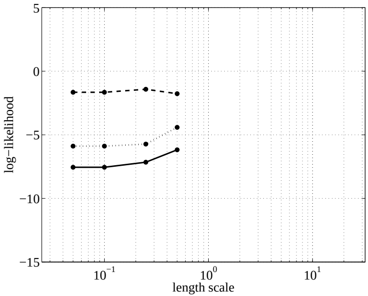


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

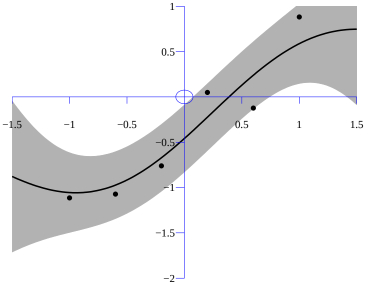


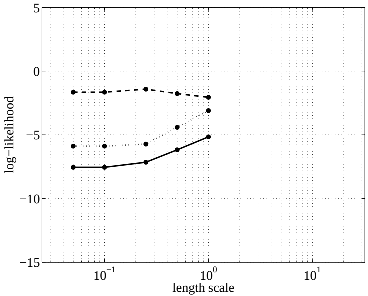


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

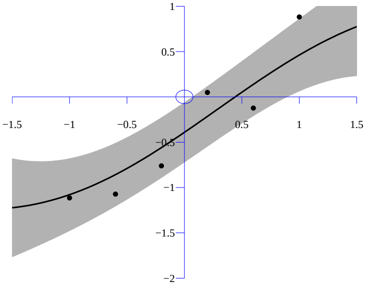


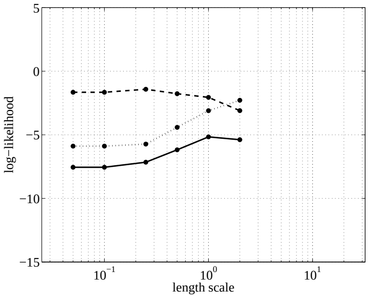


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

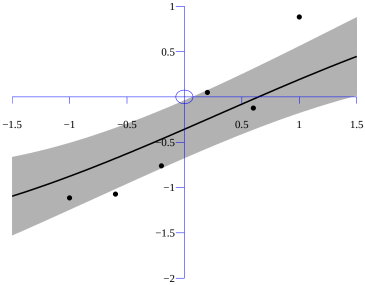


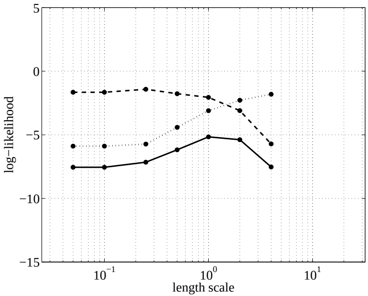


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

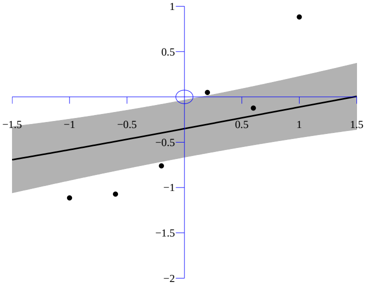


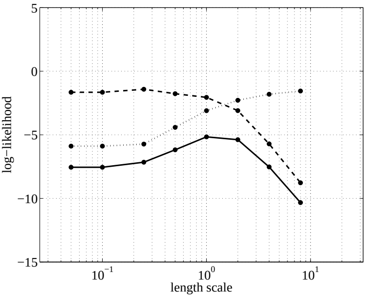


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?

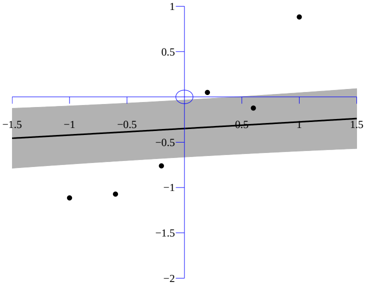


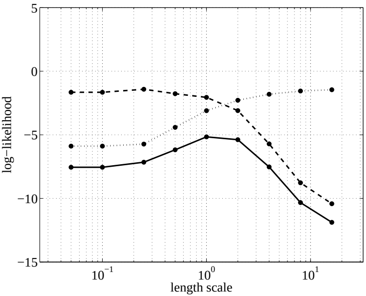


$$\log \mathcal{N}(\mathbf{y}|\mathbf{0},\mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top\mathbf{K}^{-1}\mathbf{y}}{2}$$

Learning Kernel Parameters

Can we determine length scales and noise levels from the data?





$$\log \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}) = -\frac{n}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{\mathbf{y}^\top \mathbf{K}^{-1}\mathbf{y}}{2}$$

Outline

General Noise Models

Graph of a GP

- Relates input variables, $\mathbf{x}$, to vector, $\mathbf{y}$, through $\mathbf{f}$ given kernel parameters $\boldsymbol{\theta}$.
- Plate notation indicates independence of $y_i|f_i$.
- In general $p(y_i|f_i)$ is non-Gaussian.
- We approximate with Gaussian $p(y_i|f_i) \approx \mathcal{N}(m_i|f_i, \beta_i^{-1})$.

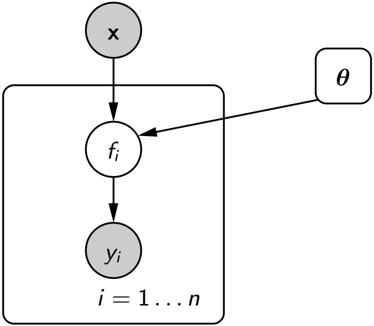


Figure: The Gaussian process depicted graphically.

Gaussian Noise

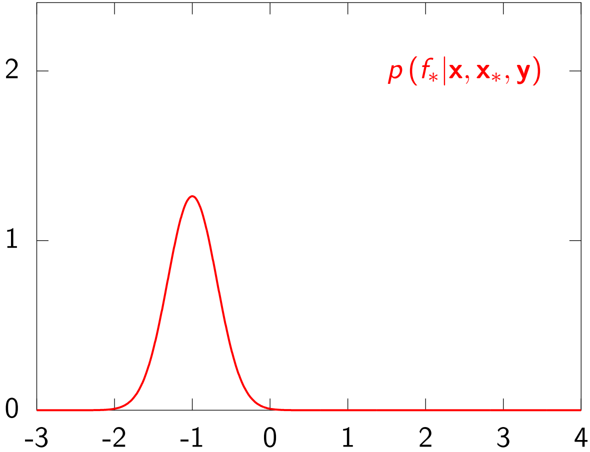


Figure: Inclusion of a data point with Gaussian noise.

Gaussian Noise

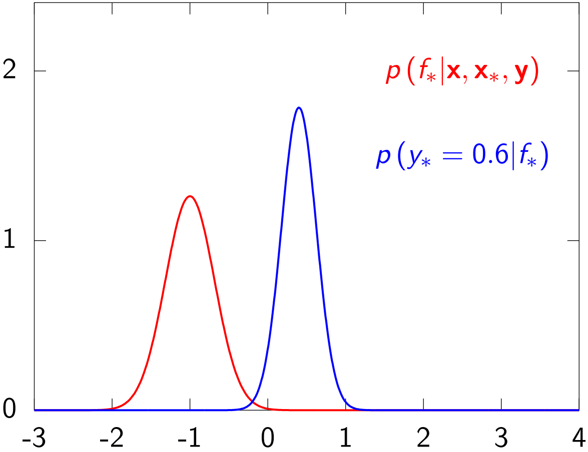


Figure: Inclusion of a data point with Gaussian noise.

Gaussian Noise

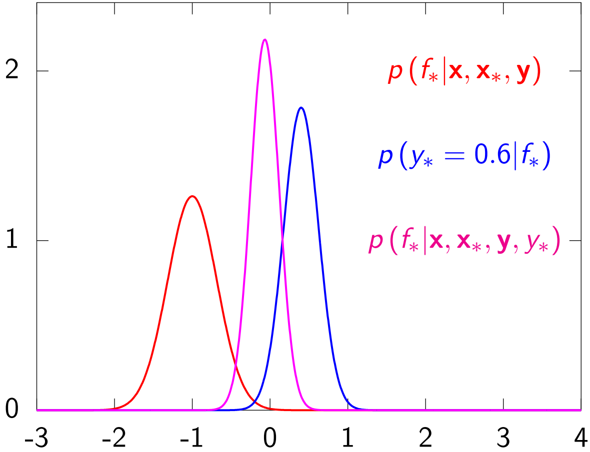


Figure: Inclusion of a data point with Gaussian noise.

Expectation Propagation

Local Moment Matching

- Easiest to consider a single previously unseen data point, y_*, $\mathbf{x}_*$.
- Before seeing data point, prediction of f_* is a GP, $q(f_*|\mathbf{y}, \mathbf{x})$.
- Update prediction using Bayes' Rule,

$$p(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*) = \frac{p(y_*|f_*)\, p(f_*|\mathbf{y}, \mathbf{x}, \mathbf{x}_*)}{p(\mathbf{y}, y_*|\mathbf{x}, \mathbf{x}_*)}.$$

  This posterior is not a Gaussian process if $p(y_*|f_*)$ is non-Gaussian.

Classification Noise Model

Probit Noise Model

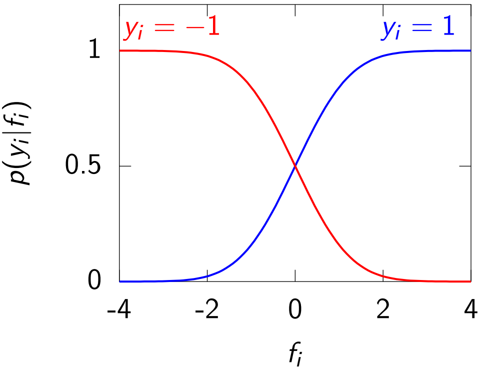


Figure: The probit model (classification). The plot shows $p\left(y_i|f_i\right)$ for different values of y_i. For $y_i = 1$ we have $p\left(y_i|f_i\right) = \phi\left(f_i\right) = \int_{-\infty}^{f_i} \mathcal{N}\left(z|0,1\right) \mathrm{d}z$.

Expectation Propagation II

Match Moments

- Idea behind EP — approximate with a Gaussian process at this stage by matching moments.
- This is equivalent to minimizing the following KL divergence where $q\left(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*\right)$ is constrained to be a GP.

$$q\left(f_*|\mathbf{y}, y_*\mathbf{x}, \mathbf{x}_*\right) = \text{argmin}_{q\left(f_*|\mathbf{y}, y_*\mathbf{x}, \mathbf{x}_*\right)} \text{KL}\left(p\left(f_*|\mathbf{y}, y_*\mathbf{x}, \mathbf{x}_*\right) || q\left(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*\right)\right)$$

- This is equivalent to setting

$$\left\langle f_*\right\rangle_{q(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*)} = \left\langle f_*\right\rangle_{p(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*)}$$

$$\left\langle f_*^2\right\rangle_{q(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*)} = \left\langle f_*^2\right\rangle_{p(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*)}$$

Expectation Propagation III

Equivalent Gaussian

- This is achieved by replacing $p\left(y_*|f_*\right)$ with a Gaussian distribution

$$p\left(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*\right) = \frac{p\left(y_*|f_*\right) p\left(f_*|\mathbf{y}, \mathbf{x}, \mathbf{x}_*\right)}{p\left(\mathbf{y}, y_*|\mathbf{x}, \mathbf{x}_*\right)}$$

becomes

$$q\left(f_*|\mathbf{y}, y_*, \mathbf{x}, \mathbf{x}_*\right) = \frac{\mathcal{N}\left(m_*|f_*, \beta_m^{-1}\right) p\left(f_*|\mathbf{y}, \mathbf{x}, \mathbf{x}_*\right)}{p\left(\mathbf{y}, y_*|\mathbf{x}, \mathbf{x}_*\right)}.$$

Classification

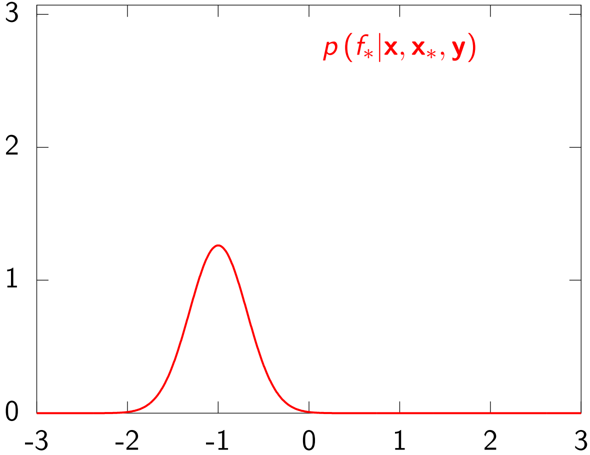


Figure: An EP style update with a classification noise model.

Classification

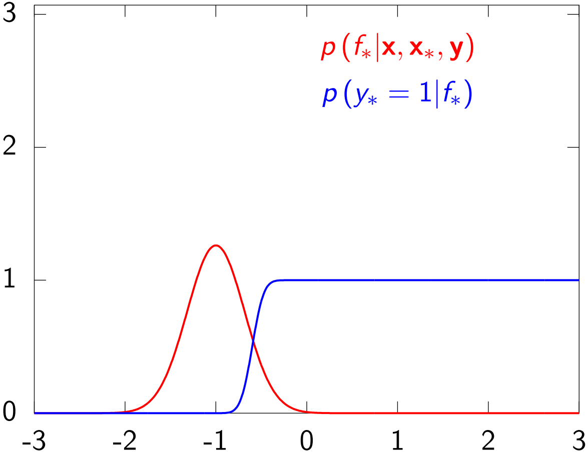


Figure: An EP style update with a classification noise model.

Classification

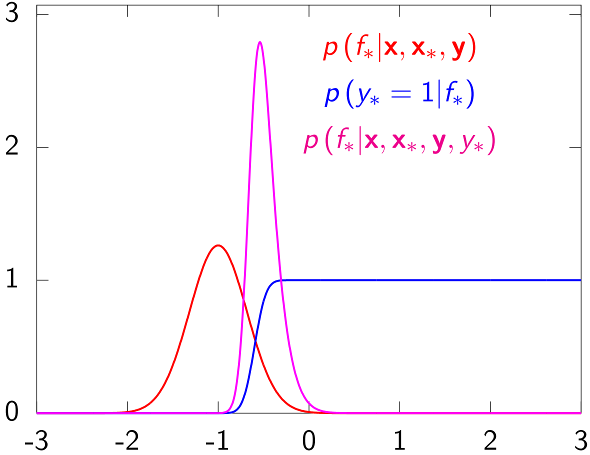


Figure: An EP style update with a classification noise model.

Classification

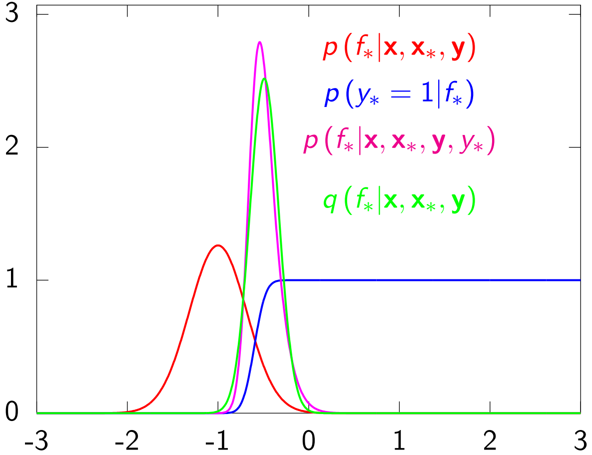


Figure: An EP style update with a classification noise model.

Ordinal Noise Model

Ordered Categories

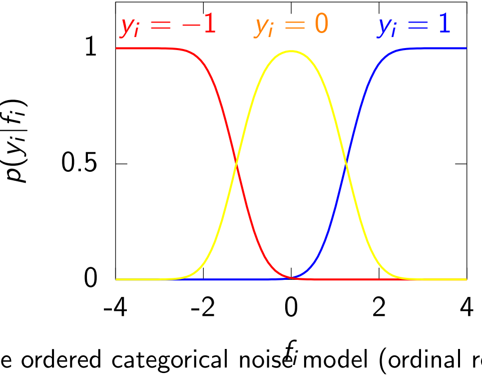


Figure: The ordered categorical noise model (ordinal regression). The plot shows $p(y_i|f_i)$ for different values of y_i. Here we have assumed three categories.

Ordinal Regression

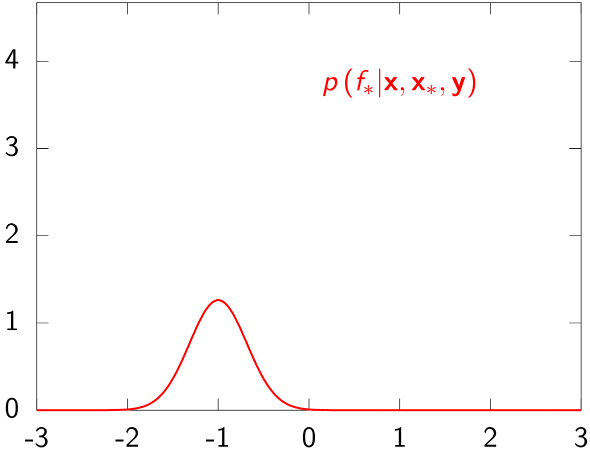


Figure: An EP style update with an ordered category noise model.

Ordinal Regression

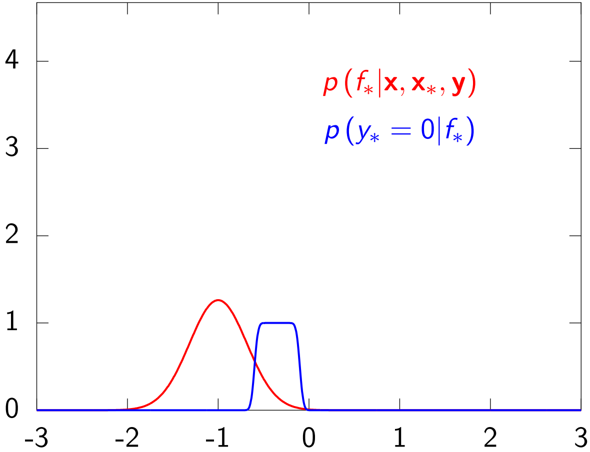


Figure: An EP style update with an ordered category noise model.

Ordinal Regression

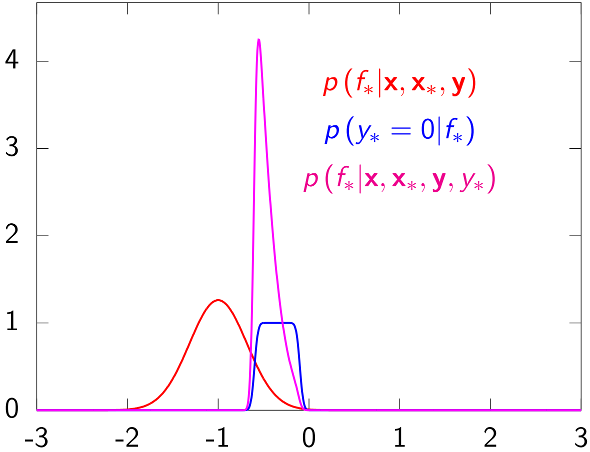


Figure: An EP style update with an ordered category noise model.

Ordinal Regression

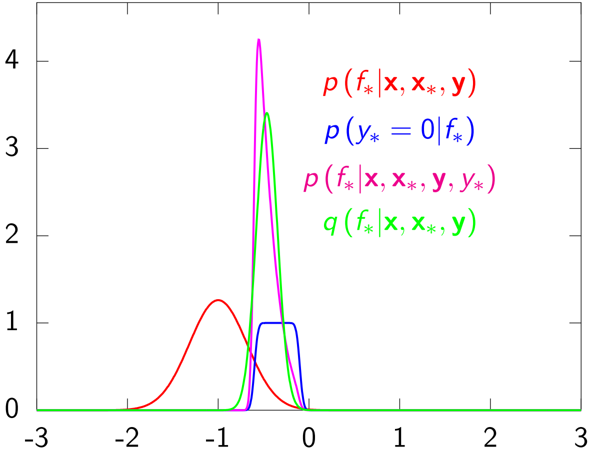


Figure: An EP style update with an ordered category noise model.

The Informative Vector Machine

Reduce Complexity

- Including n data points through ADF still leads to an $O\left(n^3\right)$ complexity.
- IVM algorithm resolves these problems with a sparse representation for the data set.
- Inspiration: the support vector machine.
- IVM use a simple selection heuristic to incorporate d most informative points (Lawrence et al., 2003; Seeger, 2004; Lawrence et al., 2005).
- Computational complexity: $O\left(n^3\right)$ to $O\left(d^2 n\right)$.
- Infromation theoretic (Chaloner and Verdinelli, 1995) criteria used to select points.

Data Point Selection

Entropy Criterion

- Original IVM criterion inspired by support vectors being those that reduce the size of the 'version space' most.
- The equivalent Bayesian interpretation is volume of the posterior: measured by *entropy*.
- Entropy change associted with a data point is simple and quick to compute.
- For jth inclusion of ith data point:

$$
\begin{aligned}
\Delta H_{j,i} &= -\frac{1}{2}\log\left|\Sigma_{j,i}\right| + \frac{1}{2}\log\left|\Sigma_{j-1}\right| \\
&= -\frac{1}{2}\log\left|\mathbf{I} - \Sigma_{j-1}\mathrm{diag}\left(\boldsymbol{\nu}_j\right)\right| \\
&= -\frac{1}{2}\log\left(1 - \nu_{j,i}\varsigma_{j-1,i}\right). \qquad (2)
\end{aligned}
$$

IVM Parameter Updates

Optimising Kernel Parameters

- Need to express the marginal likelihood for optimization.
- Seeger (2004) achieves by expressing the likelihood in terms of both the active and inactive sets.
- We simply express the likelihood in terms of the *active* set only.
- Given the active set, I, and the site parameters, $\mathbf{m}$ and $\boldsymbol{\beta}$, optimise approximation wrt kernel parameters using gradient methods.
- Active set and kernel parameters are interdependent: active set is reselected between optimisations of kernel parameters.

Results

Toy Problems

- Two toy data sets for classification with probit noise. First uses an ARD set up and one irrelevant direction.
- A second demonstation: sampled 500 data points uniformly from a unit square in two dimensions.
  - Sample then made from a GP prior of a function at these points.
  - This function was 'squashed' by a cumulative Gaussian and a class assigned according to this probability.

IVM Classification

Classification

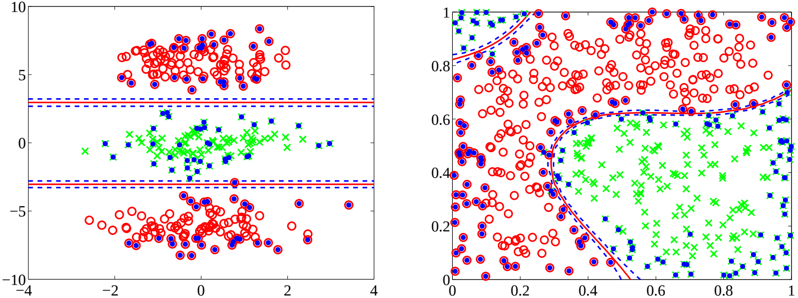


Figure: *Contours*: Red solid line at $p(y|\mathbf{x}) = 0.5$, blue dashed lines at $p(y|\mathbf{x}) = 0.25$ and $p(y|\mathbf{x}) = 0.75$. Active points are blue dots. *Left*: data sampled from from a mixture of Gaussians. *Right*: Data uniformly sampled on the 2–dimensional unit square. Class labels are assigned by sampling from a known Gaussian process prior.

Ordered Categories

Ordered Categories

- Two results from two problems on ordered categorical data.
- First example the categories are separable *linearly*.
- Second example: sampled ordered categorical data in polar co-ordinates.

Toy Problems

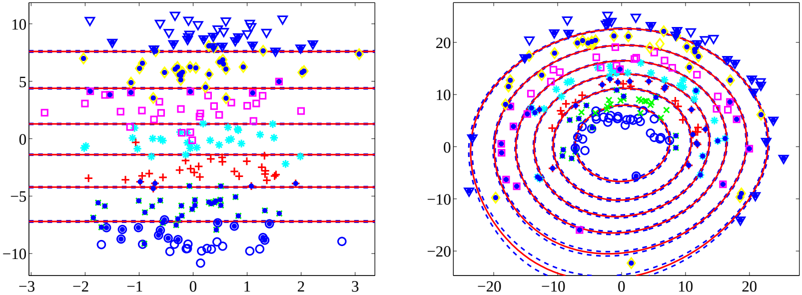


Figure: .*Left*: a linear solution is found. *Right*: this categories in this example were sampled in polar co-ordinates.

USPS digits

Large Data Set

- USPS digit data set of 16×16 greyscale images.
- Contains 7291 training images and 2007 test images.
- Three different kernels with the IVM algorithm.
  - For each data-set we used a 'base kernel' consisting of a linear part, a white noise term and a bias part.
  - Three variations on this base kernel were then used: it was changed by adding first an RBF kernel, then an MLP kernel and finally a variant of the RBF ARD kernel.
  - Set $d = 500$.

USPS digits

Classification error %

	0	1	2	3	4	5	6	7	8	9	Overall
RBF	0.65	0.70	1.40	1.05	1.49	1.25	0.75	0.60	1.20	0.75	4.58
MLP	0.55	0.70	1.49	1.20	1.64	1.25	0.80	0.60	1.20	0.75	4.78
RBF ARD	0.55	0.60	1.49	1.10	1.79	1.20	0.80	0.60	1.20	0.85	4.68

Table: Table of results on the USPS digit data. A comparison with a summary of results on this data-set Schölkopf and Smola (2001, Table 7.4) shows that the IVM is in line with other results on this data. Furthermore these results were achieved with fully automated model selection.

Incorporating Invariances

Virtual Support Vectors

- Invariances present: rotations, translations.
- Could augment the original data set with transformed data points.
- This leads to a rapid expansion in the size of the data set.
- Schölkopf et al. (1996) suggest augmenting only support vectors.
- Augmented points known as 'virtual support vectors'.
- This algorithm gives state-of-the-art performance on the USPS data set.

USPS with Virtual Informative Vectors

Virtual Informative Vectors

- Schölkopf et al. (1996): biggest improvement using translation invariances.
- Applied standard IVM classification algorithm to the data set using an RBF kernel combined with a linear term.
- Took the active set from these experiments and aumented it:
  - original active set plus four translations: up down lweft and right
  - results in an augmented active set of 2500 points.
- Reselect active set of size $d = 1000$ for final results.

Performance on USPS

Classification Error %

0	1	2	3	4	
0.648 ± 0.00	0.389 ± 0.03	0.967 ± 0.06	0.683 ± 0.05	1.06 ± 0.02	
5	**6**	**7**	**8**	**9**	**Overall**
0.747 ± 0.06	0.523 ± 0.03	0.399 ± 0.00	0.638 ± 0.04	0.523 ± 0.04	3.30 ± 0.03

Table: Experiments are summarised by the mean and variance of the % classification error across ten runs with different random seeds. Results match those given by the virtual SVM but model selection was automatic here.

Probabilistic Model

Semi-supervised Noise Model

- New noise model: *the null category noise model*.
- Derives from the general class of *ordered categorical models* (or ordinal regression).

$$p\left(y_i|f_i\right) = \begin{cases} \phi\left(-\left(f_i + \frac{w}{2}\right)\right) & \text{for } y_i = -1 \\ \phi\left(f_i + \frac{w}{2}\right) - \phi\left(f_i - \frac{w}{2}\right) & \text{for } y_i = 0 \\ \phi\left(f_i - \frac{w}{2}\right) & \text{for } y_i = 1 \end{cases},$$

Ordinal Noise Model

Ordered Categories

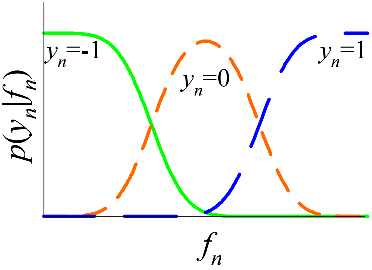
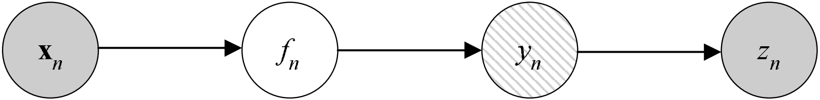


Figure: The ordered categorical noise model (ordinal regression). The plot shows $p(y_i|f_i)$ for different values of y_i. Here we have assumed three categories.

Null Category Noise Model

Noise Model for Semi-supervised Learning

- Indicator variable, $z_i = 1$ if data point is unlabeled.
- We impose the constraint:$p\left(z_i = 1|y_i = 0\right) = 0$.
- Assign missing label probabilities $p\left(z_i = 1|y_i = 1\right) = \gamma_+$ and $p\left(z_i = 1|y_i = -1\right) = \gamma_-$.



Null Category Noise Model

Noise Model for Semi-supervised Learning

- From the graphical representation z_i is d-separated from $\mathbf{x}_{i,:}$.
  - When y_i is observed, the posterior process is updated by using $p(y_i|f_i)$.
  - When the data point is unlabeled the posterior process is updated by

$$p(z_i = 1|f_i) = \sum_{y_i} p(y_i|f_i)\, p(z_i = 1|y_i)\,.$$

- The "effective likelihood function" for a single data point, $L(f_i)$, therefore takes one of three forms:

$$L(f_i) = \begin{cases} H\left(-\left(f_i + \frac{1}{2}\right)\right) & \text{for} \quad y_i = -1,\, z_i = 0 \\ \gamma_- H\left(-\left(f_i + \frac{1}{2}\right)\right) + \gamma_+ H\left(f_i - \frac{1}{2}\right) & \text{for} \quad z_i = 1 \\ H\left(f_i - \frac{1}{2}\right) & \text{for} \quad y_i = 1\, z_i = 0 \end{cases}.$$

Null Category Noise Model

Noise Model for Semi-supervised Learning

- The constraint imposed by $p\left(z_i = 1 | y_i = 0\right) = 0$ implies that:
- An unlabeled data point never comes from the class $y_i = 0$.
  - This is equivalent to a hard assumption that no data comes from the region around the decision boundary.
  - The labeled data only comes from the classes $y_i = 1$ and $y_i = -1$, so we never obtain any evidence for data with $y_i = 0$. We therefore refer to this category as the *null category* and the overall model as a *null category noise model* (NCNM).

Null Category Noise Model

Null Category

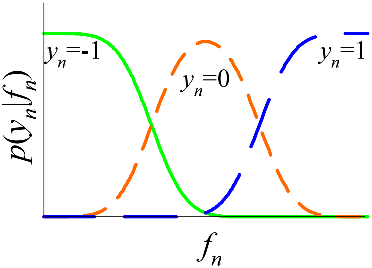


Figure: The null category noise model (semi-supervised classification).Standard noise model for labelled points ($y_i = 0$ is never observed). y_i marginalised for unlabelled points.

Null Category Noise Model

Null Category

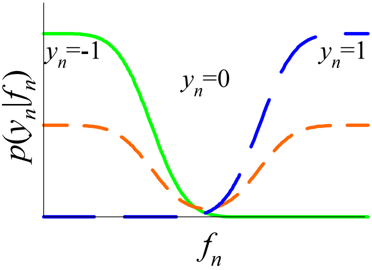


Figure: The null category noise model (semi-supervised classification). Effective noise model with y_i marginalised for unlabelled points.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, -0.3, .1, 0, 1e-2)
```

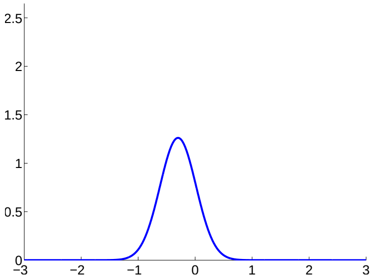


Figure: An EP style update with a classification noise model. *Blue:* $p\left(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y}\right)$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, -0.3, .1, 0, 1e-2)
```

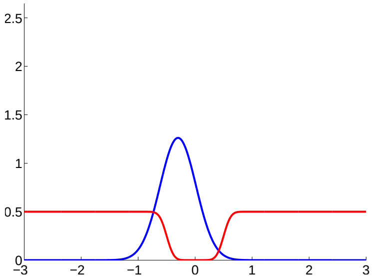


Figure: An EP style update with a classification noise model. *Blue:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$, *Red:* $p(y_* \neq 0|f_*)$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, -0.3, .1, 0, 1e-2)
```

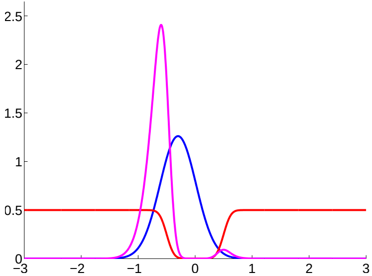


Figure: An EP style update with a classification noise model. *Blue:* $p\left(f_*|\mathbf{x},\mathbf{x}_*,\mathbf{y}\right)$, *Red:* $p\left(y_* \neq 0|f_*\right)$, *Magenta:* $p\left(f_*|\mathbf{x},\mathbf{x}_*,\mathbf{y},y_*\right)$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, -0.3, .1, 0, 1e-2)
```

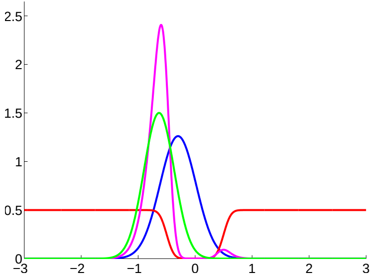


Figure: An EP style update with a classification noise model. *Blue:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$, *Red:* $p(y_* \neq 0|f_*)$, *Magenta:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y}, y_*)$, *Green:* $q(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, 0, .1, 0, 1e-2)
```

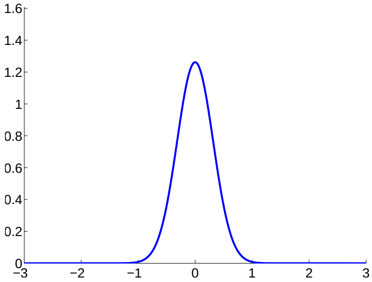


Figure: An EP style update with a classification noise model. *Blue:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, 0, .1, 0, 1e-2)
```

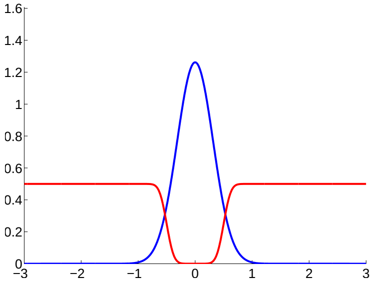


Figure: An EP style update with a classification noise model. *Blue:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$, *Red:* $p(y_* \neq 0|f_*)$.

Sparse Approximations

```
epPointUpdate('ncnm', NaN, 0, .1, 0, 1e-2)
```

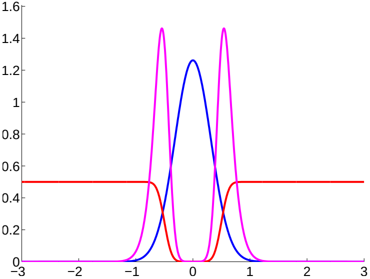


Figure: An EP style update with a classification noise model. *Blue:* $p\left(f_*|\mathbf{x},\mathbf{x}_*,\mathbf{y}\right)$, *Red:* $p\left(y_* \neq 0|f_*\right)$, *Magenta:* $p\left(f_*|\mathbf{x},\mathbf{x}_*,\mathbf{y},y_*\right)$.

Sparse Approximations

`epPointUpdate('ncnm', NaN, 0, .1, 0, 1e-2)`

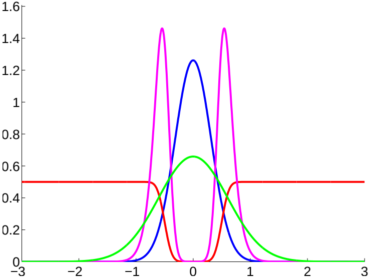


Figure: An EP style update with a classification noise model. *Blue:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$, *Red:* $p(y_* \neq 0|f_*)$, *Magenta:* $p(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y}, y_*)$, *Green:* $q(f_*|\mathbf{x}, \mathbf{x}_*, \mathbf{y})$.

The Null Category

Low Data Density at Decision Boundary

- When a data point is unlabeled the effect will depend on the mean and variance of $p(f_i|\mathbf{x}_{i,:})$.
- If this Gaussian has little mass in the null category region, the posterior will be similar to the prior.
  - If the Gaussian has significant mass in the null category region, the outcome may be loosely described in two ways:
    1. If $p(f_i|\mathbf{x}_{i,:})$ "spans the likelihood", leading to a bimodal posterior: the variance of the posterior will be greater than the variance of the prior.
    2. If $p(f_i|\mathbf{x}_{i,:})$ is "rectified by the likelihood", then the mass of the posterior will be pushed in to one side of the null category.
- Note that the posterior is pushed out to one side or both sides of the null category region.

The Posterior Process

Inference

Figure: Two situations of interest. Diagrams show the prior distribution over f_i (blue dashes) the effective likelihood function from the noise model when $z_i = 1$ (red dashes) and a schematic of the resulting posterior over f_i (green line). *Left*: The posterior is bimodal and has a larger variance than the prior. *Right*: The posterior has one dominant mode and a lower variance than the prior. In both cases the process is pushed away from the null category.

Toy Problem

Crescent Data

- We considered two-dimensional data in which two class-conditional densities interlock.
- There were 400 points in the original data set. Each point was labeled with probability 0.1, leading to 37 labeled points.
- A standard IVM classifier was trained on the labeled data only.
- We then used the null category approach to train a classifier that incorporates the unlabeled data.
- The resulting decision boundary finds a region of low data density and more accurately reflects the underlying data distribution.

Crescent Data

Standard IVM vs Semi-supervised

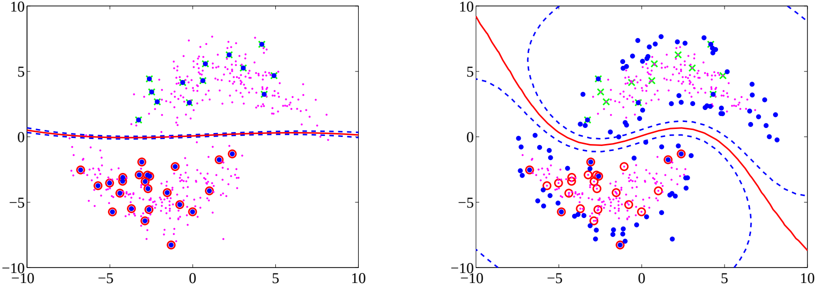


Figure: Data points: small blue dots, are labeled with probability 0.1. Labelled data-points: red circles and green crosses. Active set: large blue dots. *Left*: Learning with standard IVM. *Right*: Learning with the NCNM. Lines show centre and edge of null category.

High-dimensional example

USPS Data 3 vs 5

- As a higher dimensional example we return to the USPS data set.
- Separate the digit 3 from 5: vary probability of unlabelled data between 0.2 and 1.25×10^{-2}.
- Compare four classifiers:
  - standard IVM
  - standard SVM
  - semi-supervised IVM,
  - transductive SVM.
- Each run was completed ten times with different random seeds.

USPS Data

AUC Results

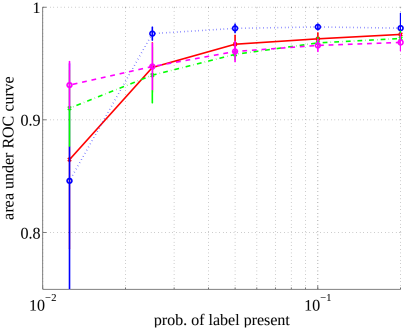


Figure: Mean and standard errors shown. IVM (red solid line), the NCNM (blue dotted line), the SVM (green dash-dot line) and the transductive SVM (pink dashed line).

USPS Data

Digits Results

- Below a label probability of 2.5×10^{-2} both the SVM and transductive SVM outperform the NCNM.
- In this region the estimate θ_1 provided by the NCNM was sometimes very low leading to occasional very poor results (note the large error bar).
- Above 2.5×10^{-2} a clear improvement is obtained for the NCNM over the other models.

Outline

Gaussian Distributions and Processes

Covariance from Basis Functions

Basis Function Representations

Bayesian Review

Building on Regression

Conclusions

Conclusions

Faster GPs through Sparsity

- We have reviewed Gaussian Processes
- Considered approaches to non-Gaussian likelihoods.
- We've shown how we can:
  - learn invariances
  - do semi-supervised learning
  - do multi-task learning
- Next time: further extensions.

References I

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995.

N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In R. Greiner and D. Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21, pages 512–519. Omnipress, 2004. [PDF].

N. D. Lawrence, J. C. Platt, and M. I. Jordan. Extensions of the informative vector machine. In J. Winkler, N. D. Lawrence, and M. Niranjan, editors, *Deterministic and Statistical Methods in Machine Learning*, volume 3635 of *Lecture Notes in Artificial Intelligence*, pages 56–87. Springer-Verlag, Berlin, 2005. [Google Books] .

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 625–632, Cambridge, MA, 2003. MIT Press.

J. Oakley and A. O'Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.

B. Schölkopf, C. J. C. Burges, and V. N. Vapnik. Incorporating invariances in support vector learning machines. In *Artificial Neural Networks — ICANN'96*, volume 1112, pages 47–52, 1996.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2001.

M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, The University of Edinburgh, 2004.

C. K. I. Williams. Computation with infinite neural networks. *Neural Computation*, 10(5):1203–1216, 1998.

Outline

Consistency of Gaussian Processes

Predictive Distribution

Consistency

Consistency of a Gaussian Process

- Predictions remain the same regardless of the number and location of the test points.

$$p\left(\mathbf{f}_*|\mathbf{f}\right) = \int p\left(\mathbf{f}_*, \mathbf{f}_+|\mathbf{f}\right) d\mathbf{f}_+,$$

- For the system to be consistent this conditional probability must be independent of the length of $\mathbf{f}_+$.
- In other words.

$$p\left(\mathbf{f}_*|\mathbf{f}\right) = \int p\left(\mathbf{f}_*, \mathbf{f}_+|\mathbf{f}\right) d\mathbf{f}_+ = \int p\left(\mathbf{f}_*, \hat{\mathbf{f}}_+|\mathbf{f}\right) d\hat{\mathbf{f}}_+$$

Outline

Consistency of Gaussian Processes

Predictive Distribution

Joint Distribution

Joint Distribution

- The covariance function provides the joint distribution over the instantiations.
- Write down the conditional distribution provides predictions.
- Denote the training set as $\mathbf{f}$ and test set as $\mathbf{f}_*$.
  - Predict using $p(\mathbf{f}_*|\mathbf{f})$.

The Conditional Distribution

Partioned Inverse

- Use partitioned inverse to find conditional.

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{\mathbf{f},\mathbf{f}} & \mathbf{K}_{\mathbf{f},*} \\ \mathbf{K}_{*,\mathbf{f}} & \mathbf{K}_{*,*} \end{bmatrix}$$

- Partitioned inverse is then

$$\mathbf{K}^{-1} = \begin{bmatrix} \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1} + \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}\Sigma^{-1}\mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1} & -\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}\Sigma^{-1} \\ -\Sigma^{-1}\mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1} & \Sigma^{-1} \end{bmatrix}$$

where

$$\Sigma = \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}.$$

Joint Distribution

Take Log of the Joint

- Logarithm of the joint distribution:

$$\log p(\mathbf{f}, \mathbf{f}_*) = -\frac{1}{2}\mathbf{f}^\top \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f} - \frac{1}{2}\mathbf{f}^\top \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}\Sigma^{-1}\mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f}$$
$$+ \mathbf{f}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}\Sigma^{-1}\mathbf{f}_* - \frac{1}{2}\mathbf{f}_*^\top\Sigma^{-1}\mathbf{f}_* + \text{const}_1$$

- Conditional is found by dividing joint by the prior, $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}})$.

Conditional Distribution

Deriving the Conditional

- In log space this is equivalent to subtraction of

$$\log p(\mathbf{f}) = -\frac{1}{2}\mathbf{f}^\top \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f} + \text{const}_2$$

giving

$$\log p(\mathbf{f}_*|\mathbf{f}) = \log p(\mathbf{f}_*, \mathbf{f}) - \log p(\mathbf{f}) = \log \mathcal{N}\left(\mathbf{f}_*|\bar{\mathbf{f}}_*, \Sigma\right).$$

where $\bar{\mathbf{f}} = \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f}$ and $\Sigma = \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f},*}$.

Making Predictions

- If we observe points from the function, $\mathbf{f}$.
- We can predict the locations of functions at as yet unseen locations.
- The prediction is also a Gaussian process, with mean $\bar{\mathbf{f}}$ and covariance Σ.
- Often observe corrupted version of function.