Learning and Inference with Gaussian Processes

An Overview of Gaussian Processes and the Gaussian Process Latent Variable Model

Neil Lawrence
Department of Computer Science
University of Sheffield

3rd November 2006

Outline

Online Resources

All source code and slides are available online

- This talk available from my home page (see talks link on side).
- MATLAB examples in the 'oxford' toolbox (vrs 0.13).
  - `http://www.dcs.shef.ac.uk/~neil/oxford/`.
- And the 'fgplvm' toolbox (vrs 0.141).
  - `http://www.dcs.shef.ac.uk/~neil/fgplvm/`.
- MATLAB commands used for examples given in `typewriter font`.

Introduction to Gaussian Processes

Inference about functions

- Many Machine Learning problems can be reduced to inference about functions.
  - We will see some examples later.
- Gaussian processes (GPs) are probabilistic models for functions. O'Hagan [1978, 1992], Rasmussen and Williams [2006]
- GPs allow inference about functions in the presence of uncertainty.

Defining a Distribution over Functions

Gaussian Process

- What is meant by a distribution over functions?
- Functions are infinite dimensional objects:
  - Defining a distribution over functions seems non-sensical.

Gaussian Distribution

- Start with a standard Gaussian distribution.
- Consider the distribution over a fixed number of instantiations of the function.

Gaussian Distribution

Zero mean Gaussian distribution

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

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

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

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

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

demGPSample

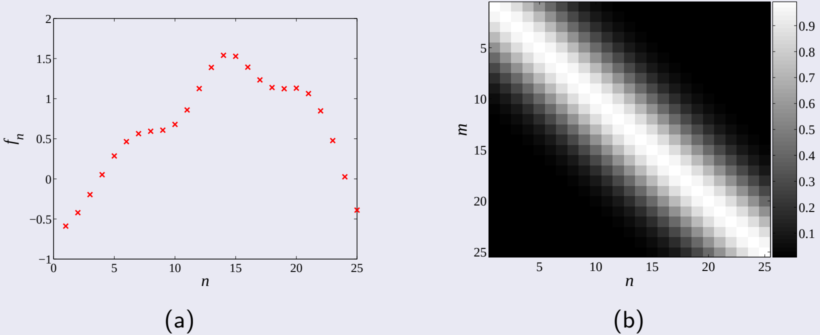


Figure: (a) 25 instantiations of a function, f_n, (b) greyscale covariance matrix.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Covariance Function

The covariance matrix

- Covariance matrix shows correlation between points f_m and f_n if n is near to m.
- Less correlation if n is distant from m.
- Our ordering of points means that the *function appears smooth*.
- Let's focus on the joint distribution of two points form the 25.

Prediction of f_2 from f_1

demGPCov2D([1 2])

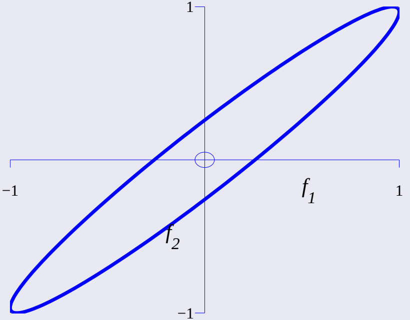


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

demGPCov2D([1 2])

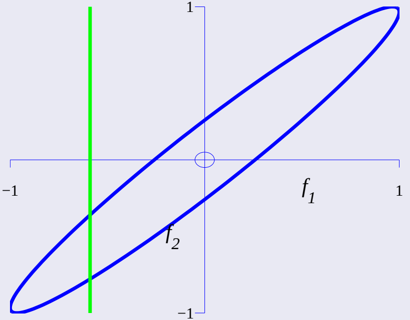


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_2 from f_1

demGPCov2D([1 2])

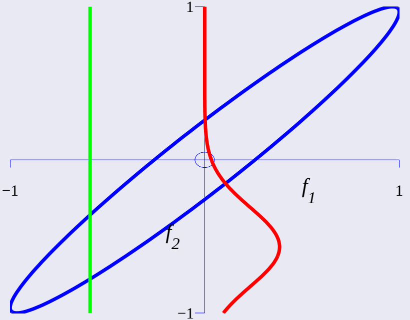


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$ is $\mathbf{K}_{12} = \begin{bmatrix} 1 & 0.966 \\ 0.966 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

demGPCov2D([1 5])

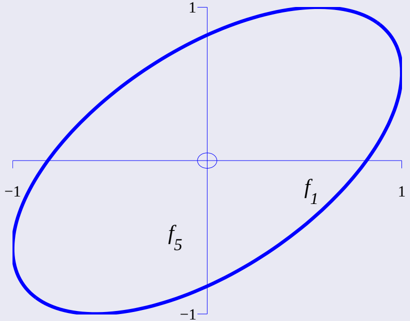


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

demGPCov2D([1 5])

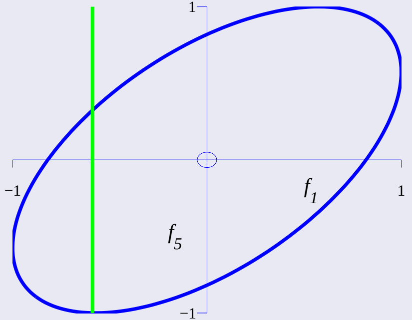


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

Prediction of f_5 from f_1

demGPCov2D([1 5])

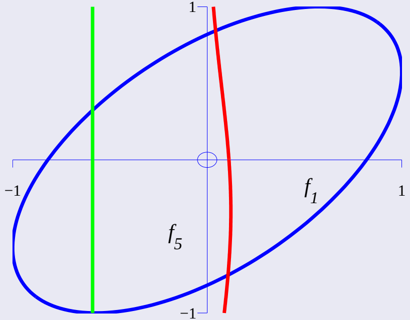


Figure: Covariance for $\begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$ is $\mathbf{K}_{15} = \begin{bmatrix} 1 & 0.574 \\ 0.574 & 1 \end{bmatrix}$.

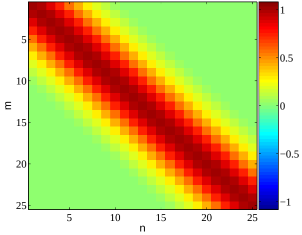
Covariance Functions

Where did this covariance matrix come from?

RBF Kernel Function

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

- Covariance matrix is built using the *inputs* to the function $\mathbf{x}_n$.
- For the example above it was based on Euclidean distance.
- The covariance function is also know as a kernel.

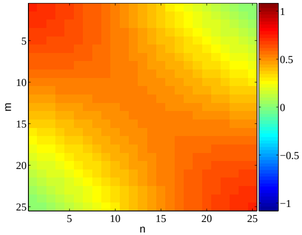


Different Covariance Functions

MLP Kernel Function

$$k\left(\mathbf{x}_m, \mathbf{x}_n\right) = \alpha \sin^{-1}\left(\frac{w\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n + b}{\sqrt{w\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_m + b + 1}\sqrt{w\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_n + b + 1}}\right)$$

- A non-stationary covariance matrix [Williams, 1997].
- Derived from a multi-layer perceptron (MLP).

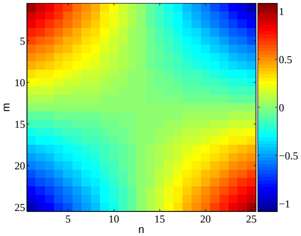


Different Covariance Functions

Linear Kernel Function

$$k\left(\mathbf{x}_m, \mathbf{x}_n\right) = \alpha \mathbf{x}_m^{\mathrm{T}} \mathbf{x}_n$$

- Allows for a linear trend.
- Note the anti-correlations in the matrix.

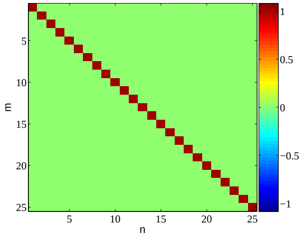


Different Covariance Functions

White noise

$$k(\mathbf{x}_m, \mathbf{x}_n) = \alpha\delta_{mn}$$

- Where δ_{mn} is the Kronecker delta.
- Simply represents uncorrelated independent noise.



Covariance Samples

demCovFuncSample

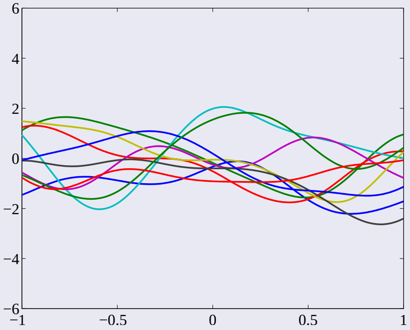


Figure: RBF kernel with $\gamma = 10$, $\alpha = 1$

Covariance Samples

demCovFuncSample

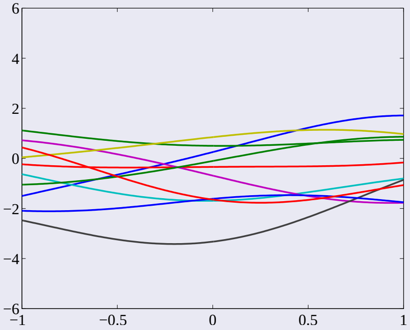


Figure: RBF kernel with $l = 1$, $\alpha = 1$

Covariance Samples

demCovFuncSample

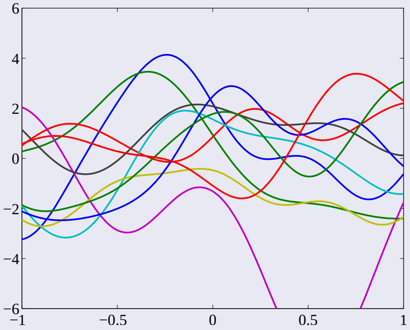


Figure: RBF kernel with $l = 0.3$, $\alpha = 4$

Covariance Samples

demCovFuncSample

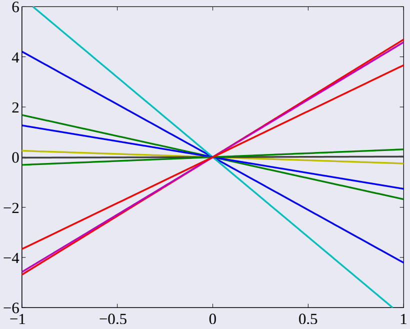


Figure: linear kernel with $\alpha = 16$

Covariance Samples

demCovFuncSample

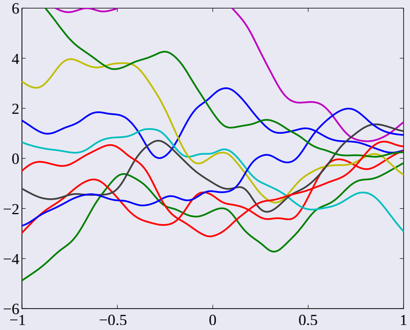


Figure: MLP kernel with $\alpha = 8$, $w = 100$ and $b = 100$

Covariance Samples

demCovFuncSample

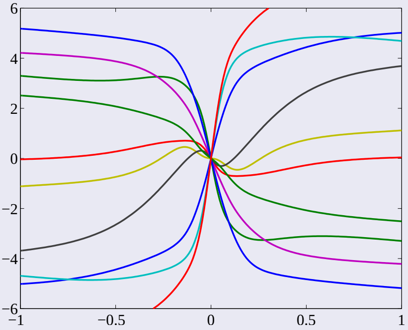


Figure: MLP kernel with $\alpha = 8$, $b = 0$ and $w = 100$

Covariance Samples

demCovFuncSample

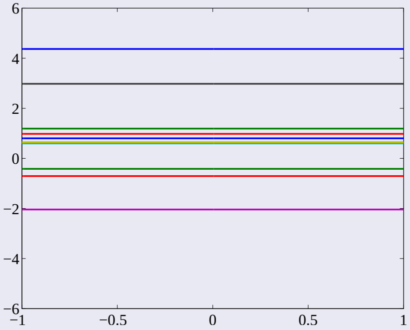


Figure: bias kernel with $\alpha = 1$ and

Covariance Samples

demCovFuncSample

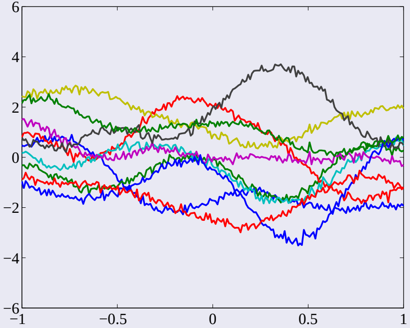


Figure: summed combination of: RBF kernel, $\alpha = 1$, $l = 0.3$; bias kernel, α =1; and white noise kernel, $\beta = 100$

Joint Distribution

Making Predictions

- Covariance function provides the joint distribution over the instantiations.
- Conditional distribution provides predictions.
- Denoting the training set as $\mathbf{f}$ and test set as $\mathbf{f}_*$.
    - Predict using $p(\mathbf{f}_*|\mathbf{f})$.
    - This conditional distribution is also Gaussian.

Gaussian Process Interpolation

demInterpolation

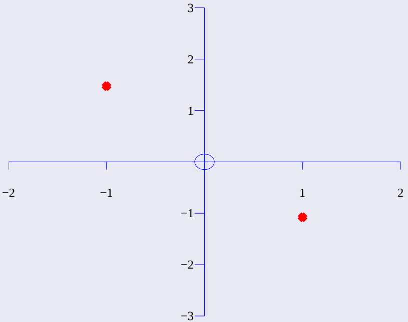


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

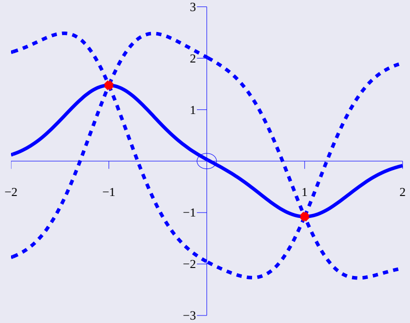


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

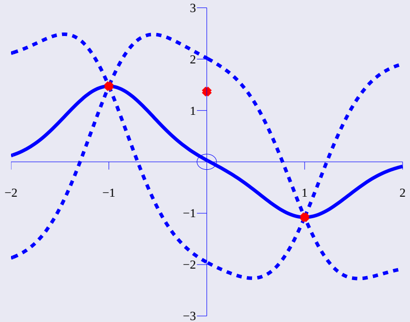


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

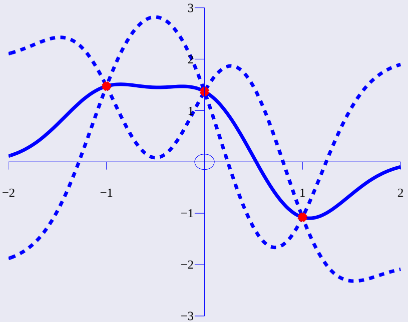


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

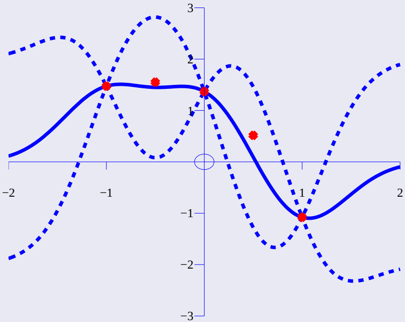


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

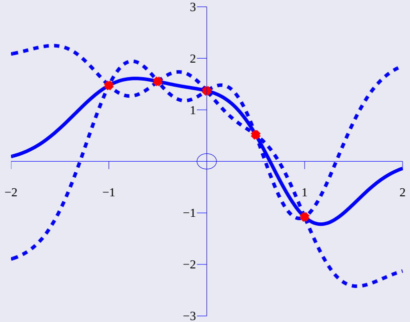


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

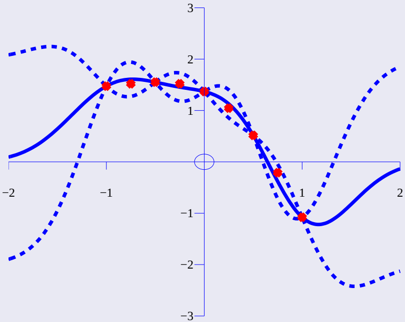


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Gaussian Process Interpolation

demInterpolation

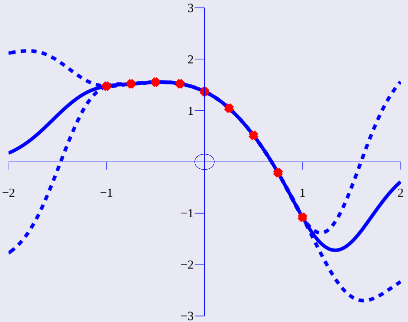


Figure: Real example: BACCO (see *e.g.* [Oakley and O'Hagan, 2002]). Interpolation through outputs from slow computer simulations (*e.g.* atmospheric carbon levels).

Noise Models

Graph of a GP

- Relates input variables, $\mathbf{X}$, to vector, $\mathbf{y}$, through $\mathbf{f}$ given kernel parameters θ.
- Plate notation indicates independence of $y_n|f_n$.
- Noise model, $p(y_n|f_n)$ can take several forms.
- Simplest is Gaussian noise.

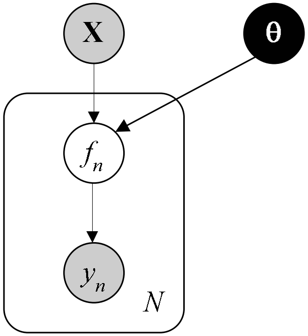


Figure: The Gaussian process depicted graphically.

Gaussian Process Regression

demRegression

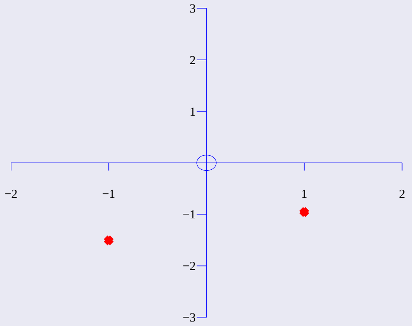


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

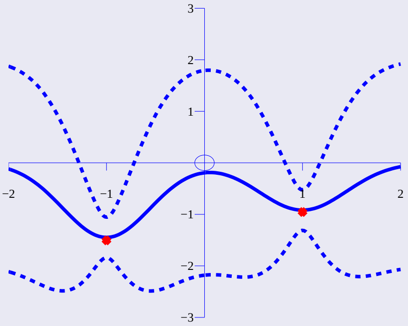


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

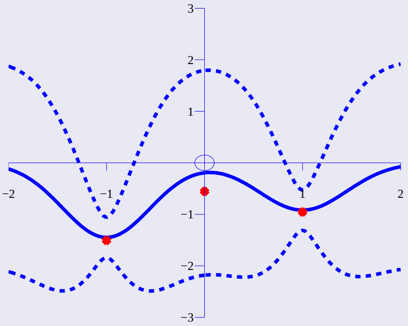


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

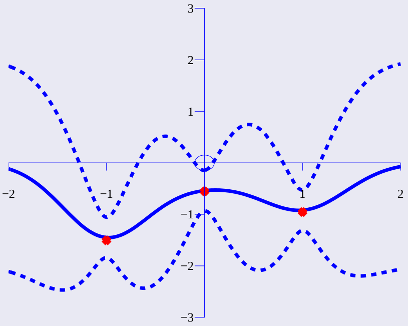


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

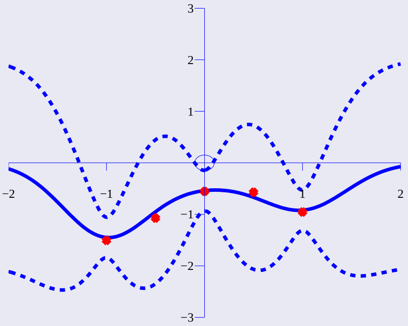


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

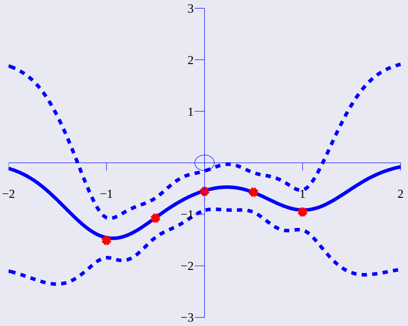


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

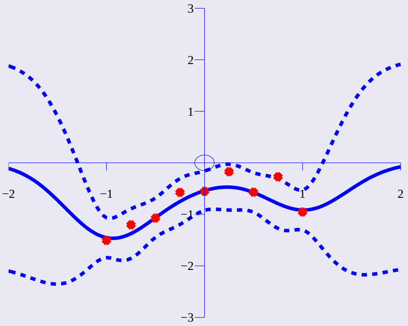


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

demRegression

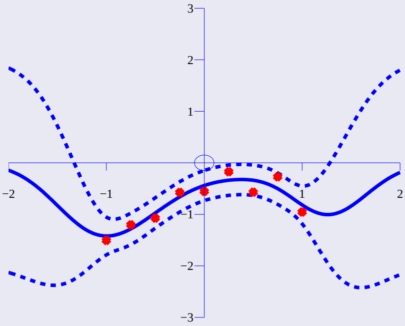


Figure: Examples include WiFi localization, C14 callibration curve.

A Paradigm Shift from i.i.d.

Parameteric Model

$$p(y_n|\mathbf{x}_n, \mathbf{w}) = N\left(y_n|\mathbf{w}^{\mathrm{T}}\mathbf{x}_n, \sigma^2\right)$$

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{n=1}^{N} p(y_n|\mathbf{x}_n, \mathbf{w})$$

Parameteric models normally assume independence given parameters.

A Paradigm Shift from i.i.d.

Gaussian process

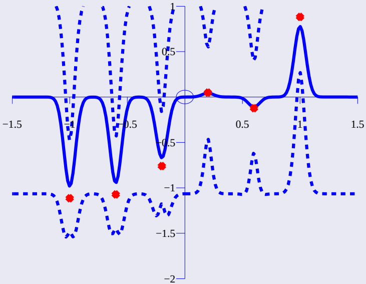
$$p(\mathbf{y}|\mathbf{X}) = N(\mathbf{y}|\mathbf{0}, \mathbf{K})$$

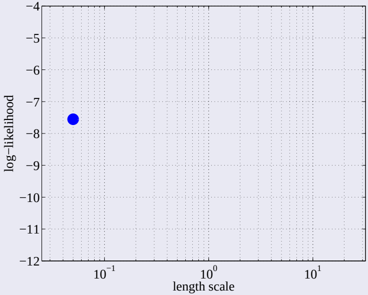
In GPs no i.i.d. assumption is made the kernel expresses correlations.

Learning Kernel Parameters

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

demOptimiseKern

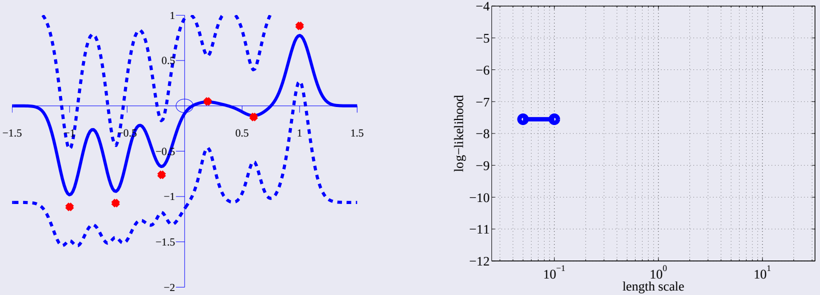




Learning Kernel Parameters

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

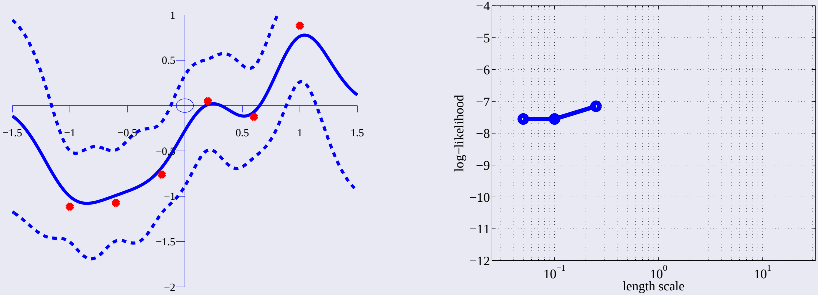
demOptimiseKern



Learning Kernel Parameters

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

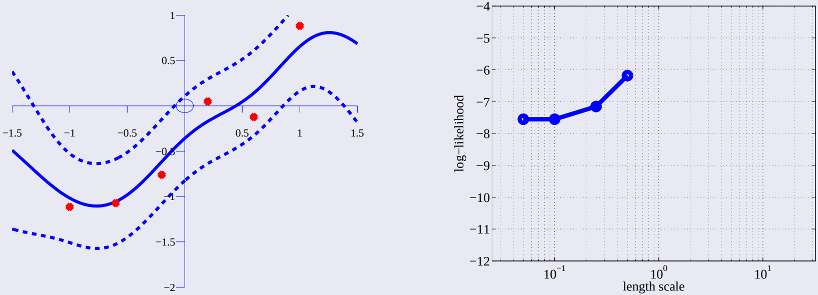
demOptimiseKern



Learning Kernel Parameters

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

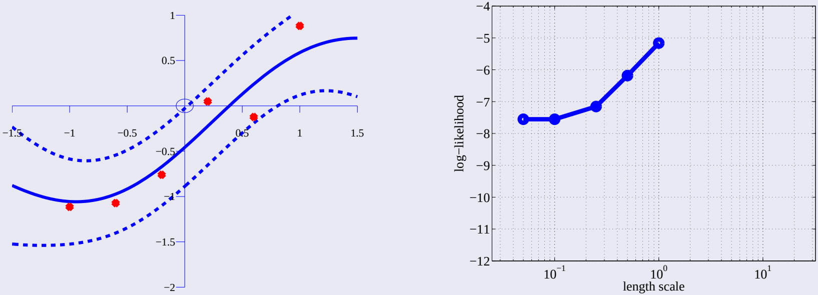
demOptimiseKern



Learning Kernel Parameters

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

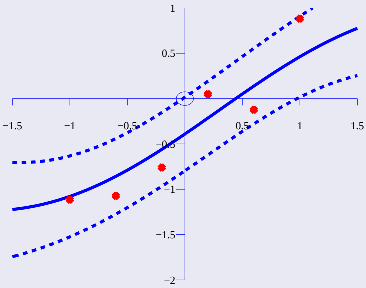
demOptimiseKern

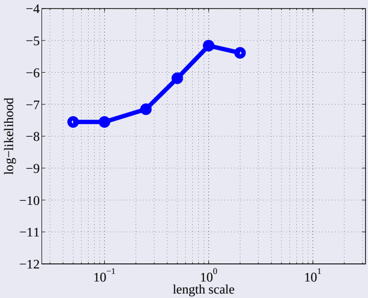


Learning Kernel Parameters

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

demOptimiseKern

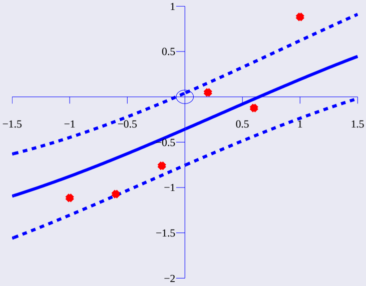


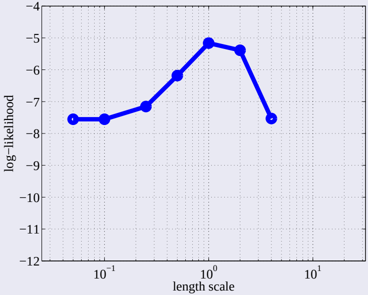


Learning Kernel Parameters

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

demOptimiseKern

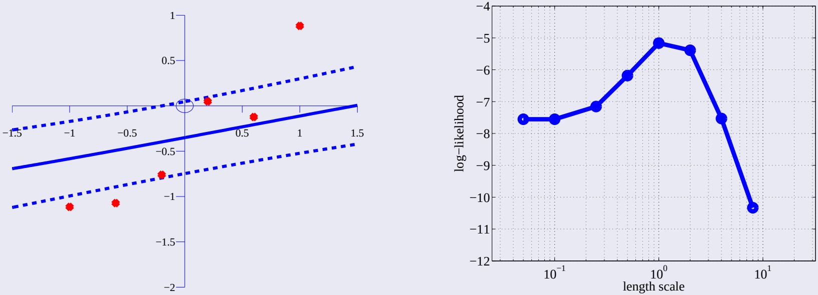




Learning Kernel Parameters

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

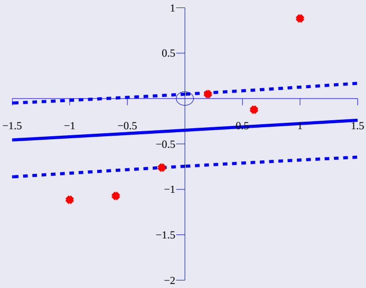
demOptimiseKern

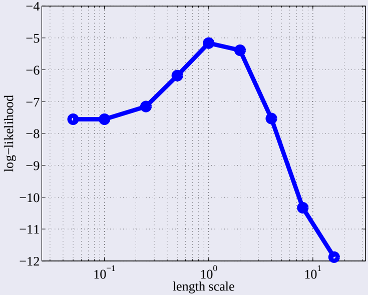


Learning Kernel Parameters

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

demOptimiseKern

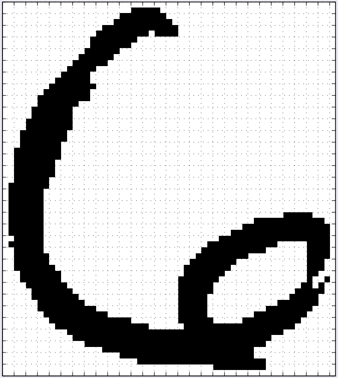




High Dimensional Data

USPS Data Set Handwritten Digit

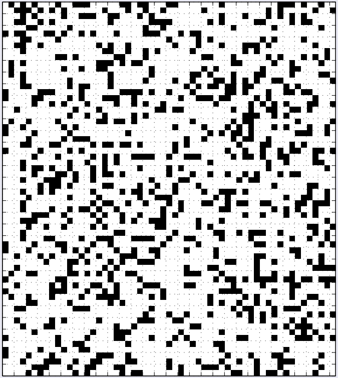
- 3648 Dimensions
  - 64 rows by 57 columns
- Space contains more than just this digit.
- Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



High Dimensional Data

USPS Data Set Handwritten Digit

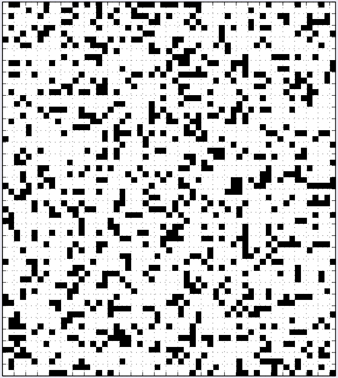
- 3648 Dimensions
  - 64 rows by 57 columns
- Space contains more than just this digit.
- Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



High Dimensional Data

USPS Data Set Handwritten Digit

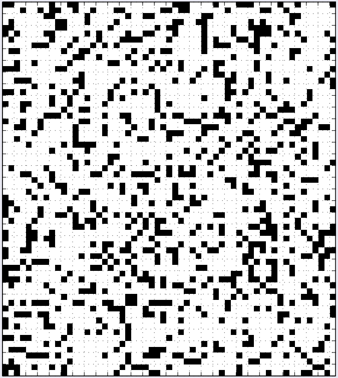
- 3648 Dimensions
  - 64 rows by 57 columns
- Space contains more than just this digit.
- Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



High Dimensional Data

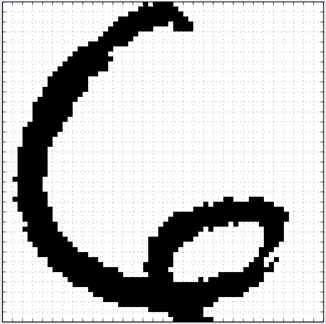
USPS Data Set Handwritten Digit

- 3648 Dimensions
  - 64 rows by 57 columns
- Space contains more than just this digit.
- Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



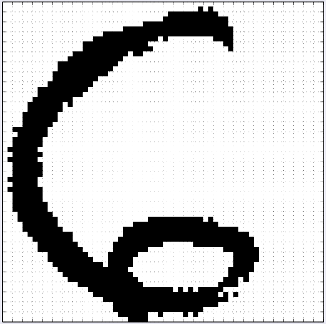
Simple Model of Digit

Rotate a 'Prototype'



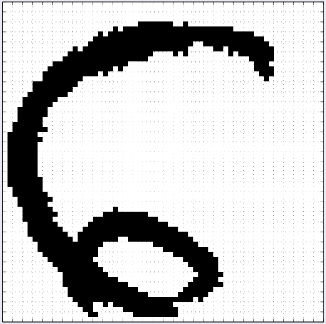
Simple Model of Digit

Rotate a 'Prototype'



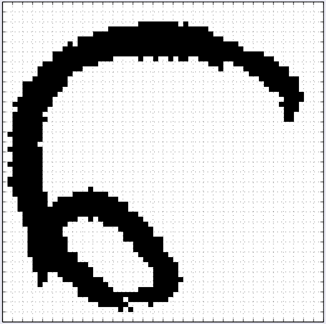
Simple Model of Digit

Rotate a 'Prototype'



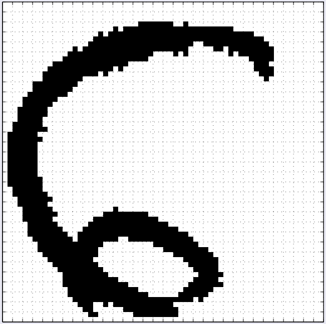
Simple Model of Digit

Rotate a 'Prototype'



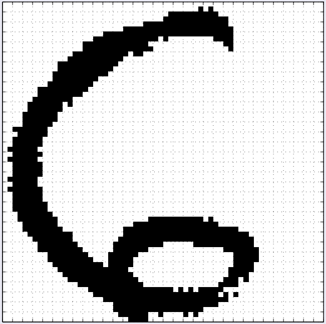
Simple Model of Digit

Rotate a 'Prototype'



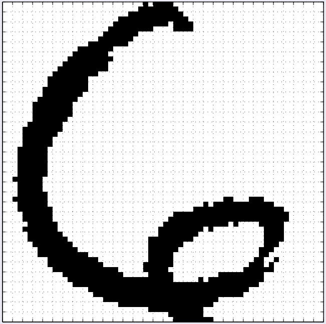
Simple Model of Digit

Rotate a 'Prototype'



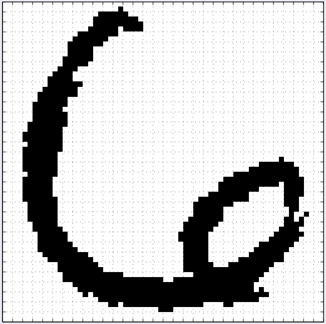
Simple Model of Digit

Rotate a 'Prototype'



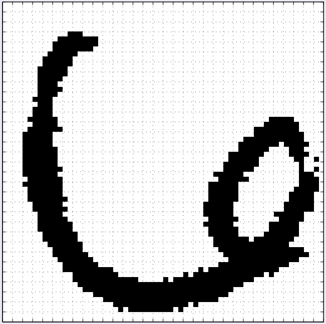
Simple Model of Digit

Rotate a 'Prototype'



Simple Model of Digit

Rotate a 'Prototype'

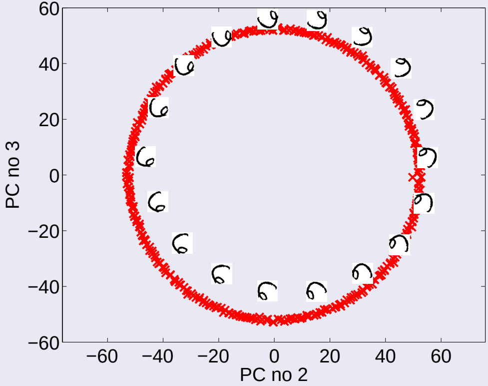


MATLAB Demo

```
demDigitsManifold[2 3], 'all')
```

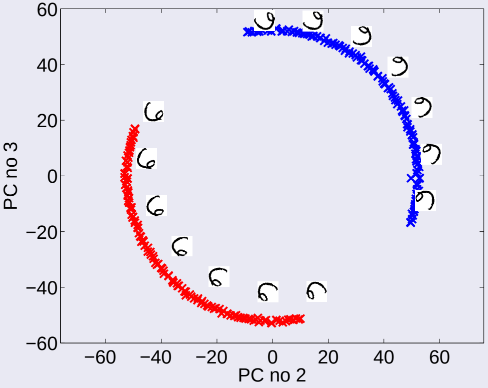
MATLAB Demo

demDigitsManifold[2 3], 'all')



MATLAB Demo

demDigitsManifold([2 3], 'sixnine')



Low Dimensional Manifolds

Pure Rotation is too Simple

- In practice the data may undergo several distortions.
  - *e.g.* digits undergo 'thinning', translation and rotation.
- For data with 'structure':
  - we expect fewer distortions than dimensions;
  - we therefore expect the data to live on a lower dimensional manifold.
- Conclusion: deal with high dimensional data by looking for lower dimensional non-linear embedding.

Existing Methods

Spectral Approaches

- Classical Multidimensional Scaling (MDS) [Mardia et al., 1979].
  - Uses eigenvectors of similarity matrix.
    - Isomap [Tenenbaum et al., 2000] is MDS with a particular proximity measure.
  - Kernel PCA [Schölkopf et al., 1998]
    - Provides an low dimensional representation and a mapping.
    - Mapping is implied throught he use of a kernel function as a similarity matrix.
  - Locally Linear Embedding [Roweis and Saul, 2000].
    - Looks to preserve locally linear relationships in a low dimensional space.

Existing Methods II

Iterative Methods

- Multidimensional Scaling (MDS)
  - Iterative optimisation of a stress function [Kruskal, 1964].
  - Sammon Mappings [Sammon, 1969].
    - Strictly speaking not a mapping — similar to iterative MDS.
- NeuroScale [Lowe and Tipping, 1997]
  - Augmentation of iterative MDS methods with a mapping.

Existing Methods III

Probabilistic Approaches

- Probabilistic PCA [Tipping and Bishop, 1999, Roweis, 1998]
  - A linear method.
- Density Networks [MacKay, 1995]
  - Use importance sampling and a multi-layer perceptron.
- Generative Topographic Mapping (GTM) [Bishop et al., 1998]
  - Uses a grid based sample and an RBF network.

Existing Methods III

Probabilistic Approaches

- Probabilistic PCA [Tipping and Bishop, 1999, Roweis, 1998]
  - A linear method.
- Density Networks [MacKay, 1995]
  - Use importance sampling and a multi-layer perceptron.
- Generative Topographic Mapping (GTM) [Bishop et al., 1998]
  - Uses a grid based sample and an RBF network.

Existing Methods III

Probabilistic Approaches

- Probabilistic PCA [Tipping and Bishop, 1999, Roweis, 1998]
    - A linear method.
- Density Networks [MacKay, 1995]
    - Use importance sampling and a multi-layer perceptron.
- Generative Topographic Mapping (GTM) [Bishop et al., 1998]
    - Uses a grid based sample and an RBF network.

Existing Methods III

Probabilistic Approaches

- Probabilistic PCA [Tipping and Bishop, 1999, Roweis, 1998]
  - A linear method.
- Density Networks [MacKay, 1995]
  - Use importance sampling and a multi-layer perceptron.
- Generative Topographic Mapping (GTM) [Bishop et al., 1998]
  - Uses a grid based sample and an RBF network.

Difficulty for Probabilistic Approaches

Propagate a probability distribution through a non-linear mapping.

The New Model

A Probabilistic Non-linear PCA

- PCA has a probabilistic interpretation [Tipping and Bishop, 1999].
- It is difficult to 'non-linearise'.

Dual Probabilistic PCA

- We present a new probabilistic interpretation of PCA [Lawrence, 2005].
- This interpretation can be made non-linear.
- The result is non-linear probabilistic PCA.

Notation

q— dimension of latent/embedded space

d— dimension of data space

n— number of data points

centred data, $\mathbf{Y} = [\mathbf{y}_{1,:}, \ldots, \mathbf{y}_{n,:}]^{\mathrm{T}} = [\mathbf{y}_{:,1}, \ldots, \mathbf{y}_{:,d}] \in \Re^{n \times d}$

latent variables, $\mathbf{X} = [\mathbf{x}_{1,:}, \ldots, \mathbf{x}_{n,:}]^{\mathrm{T}} = [\mathbf{x}_{:,1}, \ldots, \mathbf{x}_{:,q}] \in \Re^{n \times q}$

mapping matrix, $\mathbf{W} \in \Re^{d \times q}$

$\mathbf{a}_{i,:}$ is a vector from the ith row of a given matrix $\mathbf{A}$

$\mathbf{a}_{:,j}$ is a vector from the jth row of a given matrix $\mathbf{A}$

Reading Notation

$\mathbf{X}$ and $\mathbf{Y}$ are *design matrices*

- Covariance given by $n^{-1}\mathbf{Y}^{\mathrm{T}}\mathbf{Y}$.
- Inner product matrix given by $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$.

Linear Dimensionality Reduction

Linear Latent Variable Model

- Represent data, $\mathbf{Y}$, with a lower dimensional set of latent variables $\mathbf{X}$.
- Assume a linear relationship of the form

$$\mathbf{y}_{i,:} = \mathbf{W}\mathbf{x}_{i,:} + \boldsymbol{\eta}_{i,:},$$

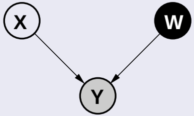
where

$$\eta_{i,:} \sim N\left(\mathbf{0}, \sigma^2\mathbf{I}\right).$$

Linear Latent Variable Model

Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Standard** Latent variable approach:
  - Define Gaussian prior over *latent space*, **X**.
  - Integrate out *latent variables*.

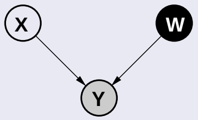


$$p\left(\mathbf{Y}|\mathbf{X},\mathbf{W}\right)=\prod_{i=1}^{n}N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^{2}\mathbf{I}\right)$$

Linear Latent Variable Model

Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Standard** Latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out *latent variables*.

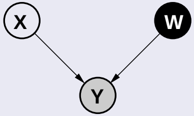


$$p\left(\mathbf{Y}|\mathbf{X},\mathbf{W}\right)=\prod_{i=1}^{n}N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^{2}\mathbf{I}\right)$$

Linear Latent Variable Model

Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Standard** Latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out *latent variables*.



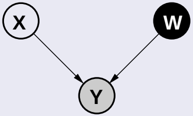
$$p(\mathbf{Y}|\mathbf{X}, \mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

$$p(\mathbf{X}) = \prod_{i=1}^{n} N\left(\mathbf{x}_{i,:}|\mathbf{0}, \mathbf{I}\right)$$

Linear Latent Variable Model

Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Standard** Latent variable approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out *latent variables*.



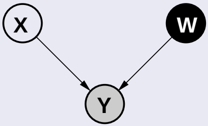
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

$$p(\mathbf{X}) = \prod_{i=1}^{n} N\left(\mathbf{x}_{i,:}|\mathbf{0},\mathbf{I}\right)$$

$$p(\mathbf{Y}|\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{0},\mathbf{W}\mathbf{W}^{\top}+\sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model II

Probabilistic PCA Max. Likelihood Soln [Tipping and Bishop, 1999]



$$p(\mathbf{Y}|\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{0}, \mathbf{W}\mathbf{W}^{\top} + \sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model II

Probabilistic PCA Max. Likelihood Soln [Tipping and Bishop, 1999]

$$p(\mathbf{Y}|\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{0}, \mathbf{C}\right), \quad \mathbf{C} = \mathbf{W}\mathbf{W}^{\top} + \sigma^2\mathbf{I}$$

$$\log p(\mathbf{Y}|\mathbf{W}) = -\frac{n}{2}\log|\mathbf{C}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{C}^{-1}\mathbf{Y}^{\top}\mathbf{Y}\right) + \mathrm{const.}$$

If $\mathbf{U}_q$ are first q principal eigenvectors of $n^{-1}\mathbf{Y}^{\top}\mathbf{Y}$ and the corresponding eigenvalues are Λ_q,

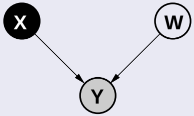
$$\mathbf{W} = \mathbf{U}_q\mathbf{L}\mathbf{V}^{\top}, \quad \mathbf{L} = \left(\Lambda_q - \sigma^2\mathbf{I}\right)^{\frac{1}{2}}$$

where $\mathbf{V}$ is an arbitrary rotation matrix.

Linear Latent Variable Model III

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Novel** Latent variable approach:
  - Define Gaussian prior over *parameteters*, $\mathbf{W}$.
  - Integrate out *parameters*.

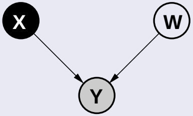


$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model III

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Novel** Latent variable approach:
  - Define Gaussian prior over *parameteters*, **W**.
  - Integrate out *parameters*.

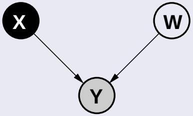


$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model III

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Novel** Latent variable approach:
  - Define Gaussian prior over *parameteters*, **W**.
  - Integrate out *parameters*.



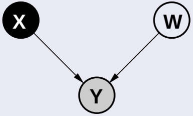
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

$$p(\mathbf{W}) = \prod_{i=1}^{d} N\left(\mathbf{w}_{i,:}|\mathbf{0},\mathbf{I}\right)$$

Linear Latent Variable Model III

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Novel** Latent variable approach:
    - Define Gaussian prior over *parameteters*, $\mathbf{W}$.
    - Integrate out *parameters*.



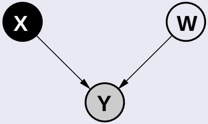
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

$$p(\mathbf{W}) = \prod_{i=1}^{d} N\left(\mathbf{w}_{i,:}|\mathbf{0},\mathbf{I}\right)$$

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0},\mathbf{X}\mathbf{X}^{\top}+\sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model IV

Dual Probabilistic PCA Max. Likelihood Soln [Lawrence, 2004]



$$p\left(\mathbf{Y}|\mathbf{X}\right)=\prod_{j=1}^{d}N\left(\mathbf{y}_{:,j}|\mathbf{0},\mathbf{X}\mathbf{X}^{\top}+\sigma^{2}\mathbf{I}\right)$$

Linear Latent Variable Model IV

Dual Probabilistic PCA Max. Likelihood Soln [Lawrence, 2004]

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}), \quad \mathbf{K} = \mathbf{X}\mathbf{X}^{\top} + \sigma^2\mathbf{I}$$

$$\log p(\mathbf{Y}|\mathbf{X}) = -\frac{d}{2}\log|\mathbf{K}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\top}\right) + \text{const.}$$

If $\mathbf{U}'_q$ are first q principal eigenvectors of $d^{-1}\mathbf{Y}\mathbf{Y}^{\top}$ and the corresponding eigenvalues are Λ_q,

$$\mathbf{X} = \mathbf{U}'_q\mathbf{L}\mathbf{V}^{\top}, \quad \mathbf{L} = \left(\Lambda_q - \sigma^2\mathbf{I}\right)^{\frac{1}{2}}$$

where $\mathbf{V}$ is an arbitrary rotation matrix.

Linear Latent Variable Model IV

Probabilistic PCA Max. Likelihood Soln [Tipping and Bishop, 1999]

$$p(\mathbf{Y}|\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{0}, \mathbf{C}\right), \quad \mathbf{C} = \mathbf{W}\mathbf{W}^{\top} + \sigma^2\mathbf{I}$$

$$\log p(\mathbf{Y}|\mathbf{W}) = -\frac{n}{2}\log|\mathbf{C}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{C}^{-1}\mathbf{Y}^{\top}\mathbf{Y}\right) + \text{const.}$$

If $\mathbf{U}_q$ are first q principal eigenvectors of $n^{-1}\mathbf{Y}^{\top}\mathbf{Y}$ and the corresponding eigenvalues are Λ_q,

$$\mathbf{W} = \mathbf{U}_q\mathbf{L}\mathbf{V}^{\top}, \quad \mathbf{L} = \left(\Lambda_q - \sigma^2\mathbf{I}\right)^{\frac{1}{2}}$$

where $\mathbf{V}$ is an arbitrary rotation matrix.

Equivalence of Formulations

The Eigenvalue Problems are equivalent

- Solution for Probabilistic PCA (solves for the mapping)

$$\mathbf{Y}^{\top}\mathbf{Y}\mathbf{U}_q = \mathbf{U}_q\Lambda_q \qquad \mathbf{W} = \mathbf{U}_q\mathbf{L}\mathbf{V}^{\top}$$

- Solution for Dual Probabilistic PCA (solves for the latent positions)

$$\mathbf{Y}\mathbf{Y}^{\top}\mathbf{U}_q' = \mathbf{U}_q'\Lambda_q \qquad \mathbf{X} = \mathbf{U}_q'\mathbf{L}\mathbf{V}^{\top}$$

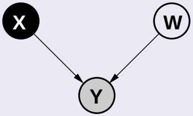
- Equivalence is from

$$\mathbf{U}_q = \mathbf{Y}^{\top}\mathbf{U}_q'\Lambda_q^{-\frac{1}{2}}$$

Non-Linear Latent Variable Model

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
- **Novel** Latent variable approach:
  - Define Gaussian prior over *parameteters*, $\mathbf{W}$.
  - Integrate out *parameters*.



$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

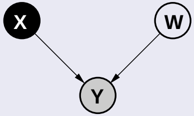
$$p(\mathbf{W}) = \prod_{i=1}^{d} N\left(\mathbf{w}_{i,:}|\mathbf{0},\mathbf{I}\right)$$

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0},\mathbf{X}\mathbf{X}^{\mathrm{T}}+\sigma^2\mathbf{I}\right)$$

Non-Linear Latent Variable Model

Dual Probabilistic PCA

- Inspection of the marginal likelihood shows ...
  - The covariance matrix is a covariance function.
  - We recognise it as the 'linear kernel'.
  - We call this the Gaussian Process Latent Variable model (GP-LVM).

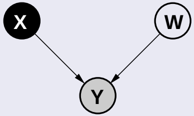


$$p\left(\mathbf{Y}|\mathbf{X}\right) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{X}\mathbf{X}^{\top} + \sigma^2\mathbf{I}\right)$$

Non-Linear Latent Variable Model

Dual Probabilistic PCA

- Inspection of the marginal likelihood shows ...
  - The covariance matrix is a covariance function.
  - We recognise it as the ‘linear kernel’.
  - We call this the Gaussian Process Latent Variable model (GP-LVM).



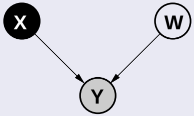
$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

$$\mathbf{K} = \mathbf{X}\mathbf{X}^{\top} + \sigma^2\mathbf{I}$$

Non-Linear Latent Variable Model

Dual Probabilistic PCA

- Inspection of the marginal likelihood shows ...
  - The covariance matrix is a covariance function.
  - We recognise it as the 'linear kernel'.
  - We call this the Gaussian Process Latent Variable model (GP-LVM).



$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

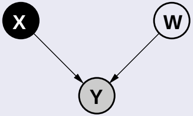
$$\mathbf{K} = \mathbf{X}\mathbf{X}^{\top} + \sigma^2\mathbf{I}$$

This is a product of Gaussian processes with linear kernels.

Non-Linear Latent Variable Model

Dual Probabilistic PCA

- Inspection of the marginal likelihood shows ...
  - The covariance matrix is a covariance function.
  - We recognise it as the 'linear kernel'.
  - We call this the Gaussian Process Latent Variable model (GP-LVM).



$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

$$\mathbf{K} = ?$$

Replace linear kernel with non-linear kernel for non-linear model.

Non-linear Latent Variable Models

RBF Kernel

- For example, use the RBF kernel

$$k\left(\mathbf{x}_{i,:}, \mathbf{x}_{j,:}\right) = \alpha \exp\left(-\frac{\left(\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right)^{\top}\left(\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right)}{2l^2}\right).$$

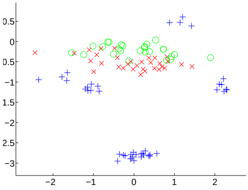
- No longer possible to optimise wrt $\mathbf{X}$ via an eigenvalue problem.
- Instead find gradients with respect to $\mathbf{X}, \alpha, l$ and σ^2 and optimise using conjugate gradients.

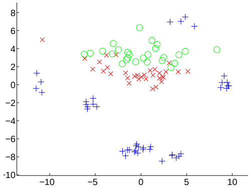
Oil Data I

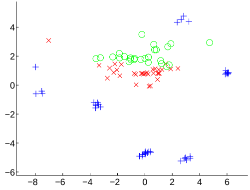
Example Data set

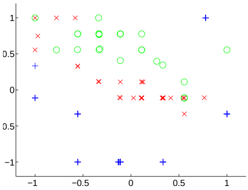
- Oil flow data [Bishop and James, 1993].
- Three phases of flow (stratified, annular, homogenous).
- Twelve measurement probes.
- 1000 data points.
- We sub-sampled to 100 data points
- Compare, with KPCA, MDS, Sammon mappings, PCA and GTM.

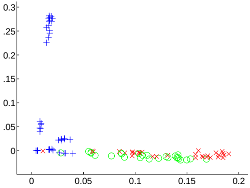
Oil Data II

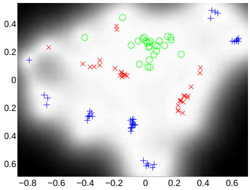

(a) PCA


(b) Non-metric MDS


(c) Sammon Mapping


(d) GTM


(e) Kernel PCA


(f) GP-LVM

Oil Data III

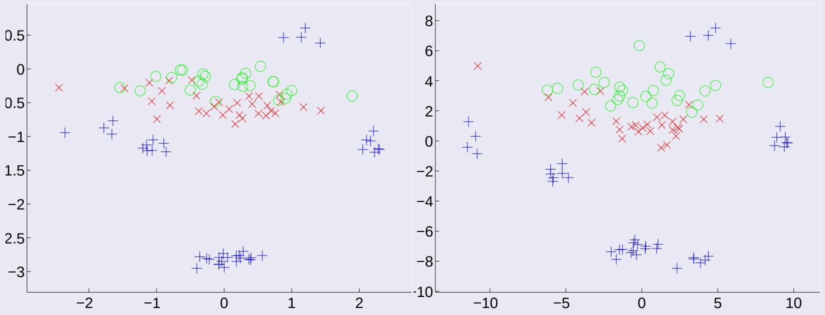


Figure: *Left* PCA, *right* Non-metric MDS

Oil Data III

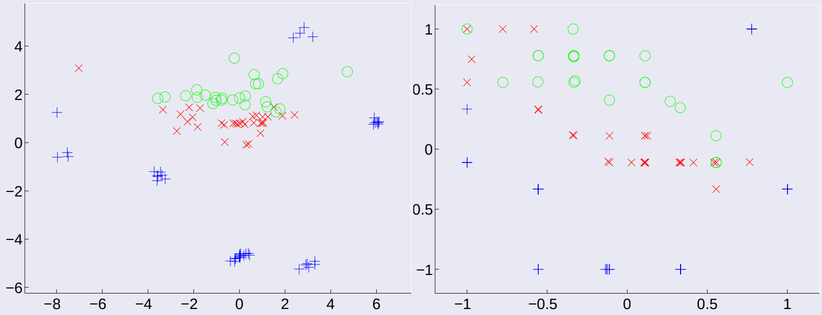


Figure: *Left* Sammon Mapping, *right* GTM

Oil Data III

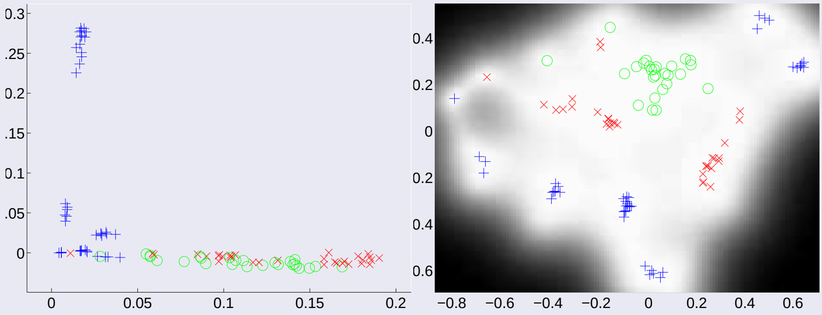


Figure: *Left* Kernel PCA, *right* GP-LVM

Oil Data IV

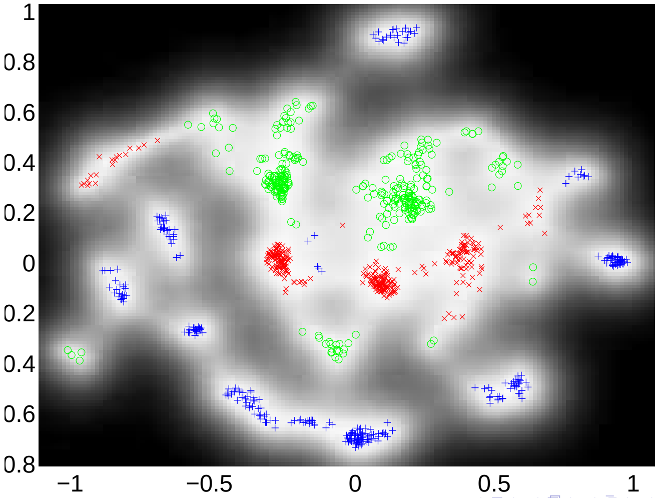
Nearest neighbour errors in $\mathbf{X}$ space

- Nearest neighbour classification in latent space.

Method	PCA	Non-metric MDS	Sammon Mapping
Errors	20	13	6
Method	GTM*	Kernel PCA*	GP-LVM
Errors	7	13	4

* These models require parameter selection.

Full Oil Data Set I



Full Oil Data Set II

Nearest Neighbour error in $\mathbf{X}$

- Nearest neighbour classification in latent space.

Method	PCA	GTM	GP-LVM
Errors	162	11	1

cf 2 errors in data space.

Stick Man

Generalization with less Data than Dimensions

- Powerful uncertainly handling of GPs leads to suprising properties.
- Non-linear models can be used where there are fewer data points than dimensions *without overfitting*.
- Example: Modelling a stick man in 102 dimensions with 55 data points!

Stick Man II

demStick1

Figure: The latent space for the stick man motion capture data.

Stick Man II

demStick1

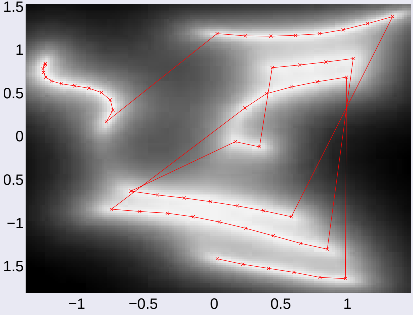


Figure: The latent space for the stick man motion capture data.

Applications

Style Based Inverse Kinematics

Facilitating animation through modelling human motion with the GP-LVM [Grochow et al., 2004]

Tracking

Tracking using models of human motion learnt with the GP-LVM [Urtasun et al., 2005]

Face Animation

Modelling facial motion capture data for synthesis of emotion and speech.

Back Constraints I

Local Distance Preservation [Lawrence and Quiñonero Candela, 2006]

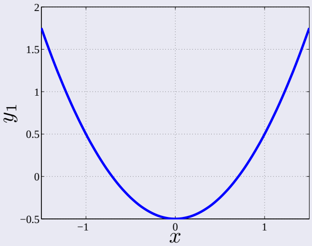
- Most dimensional reduction techniques preserve local distances.
- The GP-LVM does not.
- GP-LVM maps smoothly from latent to data space.
  - Points close in latent space are close in data space.
  - This does not imply points close in data space are close in latent space.
- Kernel PCA maps smoothly from data to latent space.
  - Points close in data space are close in latent space.
  - This does not imply points close in latent space are close in data space.

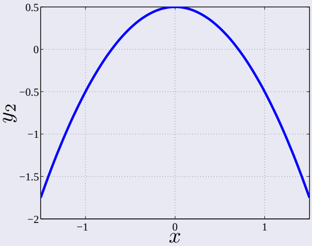
Back Constraints II

Forward Mapping (demBackMapping in oxford toolbox)

- Mapping from 1-D latent space to 2-D data space.

$$y_1 = x^2 - 0.5, \quad y_2 = -x^2 + 0.5$$



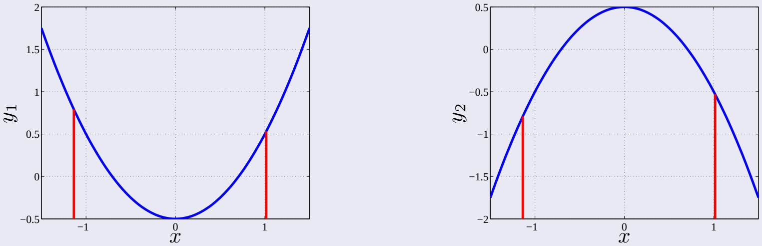


Back Constraints II

Forward Mapping (demBackMapping in oxford toolbox)

- Mapping from 1-D latent space to 2-D data space.

$$y_1 = x^2 - 0.5, \quad y_2 = -x^2 + 0.5$$

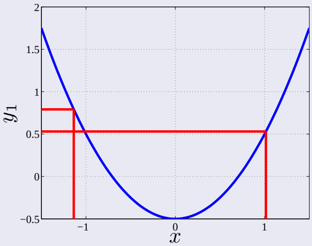


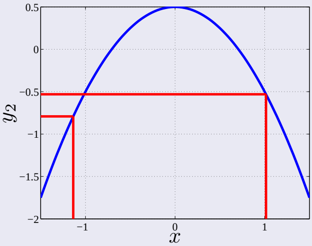
Back Constraints II

Forward Mapping (demBackMapping in oxford toolbox)

- Mapping from 1-D latent space to 2-D data space.

$$y_1 = x^2 - 0.5, \quad y_2 = -x^2 + 0.5$$



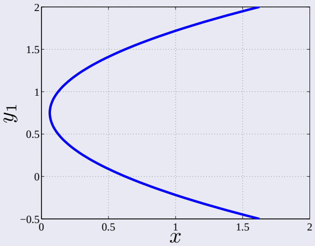


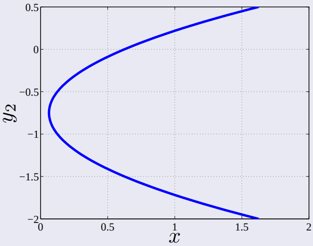
Back Constraints II

Backward Mapping (demBackMapping in oxford toolbox)

- Mapping from 2-D data space to 1-D latent.

$$x = 0.5\left(y_1^2 + y_2^2 + 1\right)$$



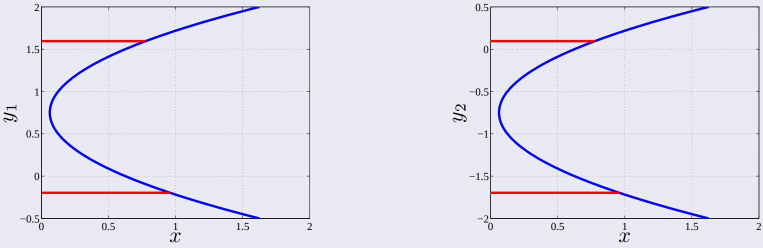


Back Constraints II

Backward Mapping (demBackMapping in oxford toolbox)

- Mapping from 2-D data space to 1-D latent.

$$x = 0.5\left(y_1^2 + y_2^2 + 1\right)$$

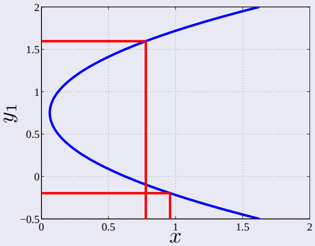


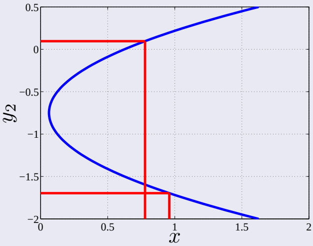
Back Constraints II

Backward Mapping (demBackMapping in oxford toolbox)

- Mapping from 2-D data space to 1-D latent.

$$x = 0.5\left(y_1^2 + y_2^2 + 1\right)$$





NeuroScale

Multi-Dimensional Scaling with a Mapping

- Lowe and Tipping [1997] made latent positions a function of the data.

$$x_{ij} = f_j\left(\mathbf{y}_i; \mathbf{w}\right)$$

- Function was either multi-layer perceptron or a radial basis function network.
- Their motivation was different from ours:
  - They wanted to add the advantages of a true mapping to multi-dimensional scaling.

Back Constraints in the GP-LVM

Back Constraints

- We can use the same idea to force the GP-LVM to respect local distances.
  - By constraining each $\mathbf{x}_i$ to be a 'smooth' mapping from $\mathbf{y}_i$ local distances can be respected.
- This works because in the GP-LVM we maximise wrt latent variables, we don't integrate out.
- Can use any 'smooth' function:
  1. Neural network.
  2. RBF Network.
  3. Kernel based mapping.

Optimising BC-GPLVM

Computing Gradients

- GP-LVM normally proceeds by optimising

$$L(\mathbf{X}) = \log p(\mathbf{Y}|\mathbf{X})$$

  with respect to $\mathbf{X}$ using $\frac{dL}{d\mathbf{X}}$.
- The back constraints are of the form

$$x_{ij} = f_j(\mathbf{y}_{i,:}; \mathbf{B})$$

  where $\mathbf{B}$ are parameters.
- We can compute $\frac{dL}{d\mathbf{B}}$ via chain rule and optimise parameters of mapping.

Motion Capture Results

demStick1 and demStick3

Figure: The latent space for the motion capture data with (*right*) and without (*left*) dynamics. The dynamics us a Gaussian process with an RBF kernel.

.

Motion Capture Results

demStick1 and demStick3

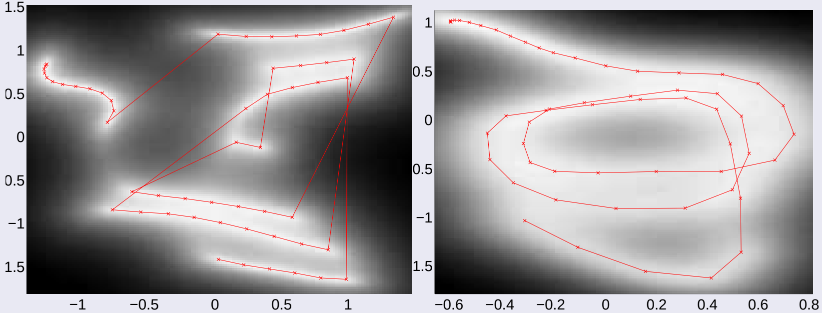
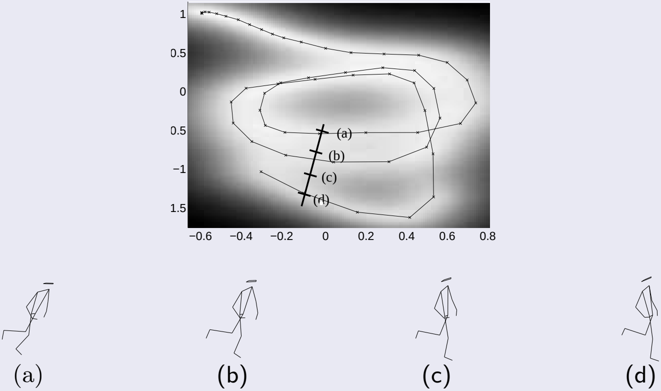


Figure: The latent space for the motion capture data with (*right*) and without (*left*) dynamics. The dynamics us a Gaussian process with an RBF kernel.

Stick Man Results

demStickResults



Projection into data space from four points in the latent space. The inclination of the runner changes becoming more upright.

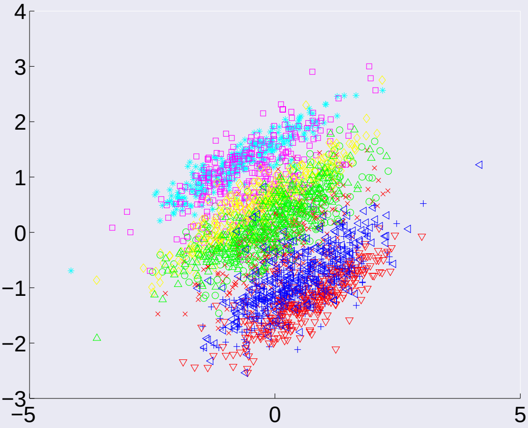
Vowel Data

Vocal Joystick Data

- Vowel sounds from a vocal joystick system [Bilmes et al., 2006].
  - http://ssli.ee.washington.edu/vj
- Vowels are from a single speaker and represented as:
  - cepstral coefficients (12 dimensions) and
  - 'deltas' (further 12 dimensions).
- 2700 data points in total (300 for each vowel).

PCA Results

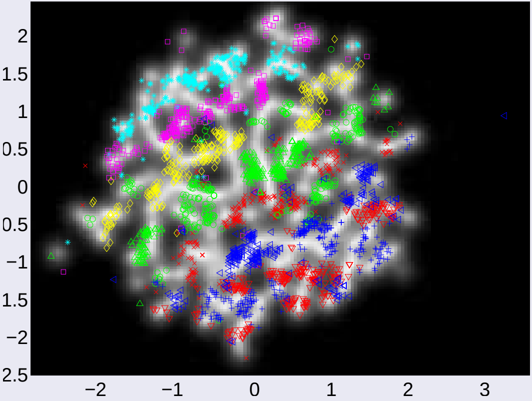
PCA (used as initialisation for GP-LVM



The different vowels are shown as follows: */a/* red cross */ae/* green circle */ao/* blue plus */e/* cyan asterix */i/* pink square */ibar/* yellow diamond */o/* red down triangle */schwa/* green up triangle and */u/* blue left triangle.

GP-LVM Results

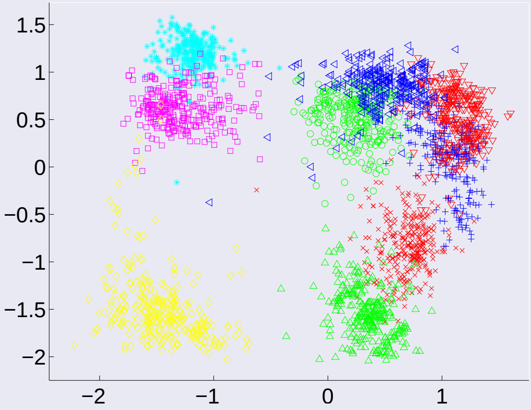
demVowels2



The different vowels are shown as follows: */a/* red cross */ae/* green circle */ao/* blue plus */e/* cyan asterix */i/* pink square */ibar/* yellow diamond */o/* red down triangle */schwa/* green up triangle and */u/* blue left triangle.

Isomap Results

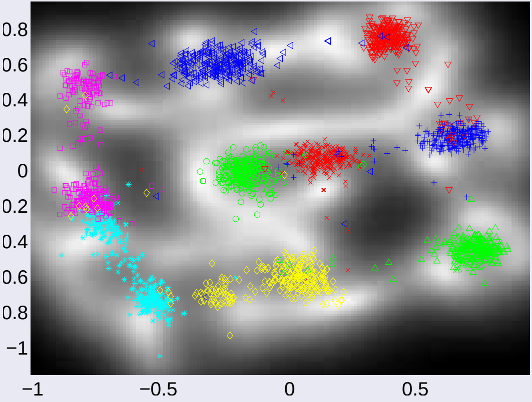
demVowelsIsomap



The different vowels are shown as follows: */a/* red cross */ae/* green circle */ao/* blue plus */e/* cyan asterix */i/* pink square */ibar/* yellow diamond */o/* red down triangle */schwa/* green up triangle and */u/* blue left triangle.

BC-GPLVM Results

demVowels3



The different vowels are shown as follows: /*a*/ red cross /*ae*/ green circle /*ao*/ blue plus /*e*/ cyan asterix /*i*/ pink square /*ibar*/ yellow diamond /*o*/ red down triangle /*schwa*/ green up triangle and /*u*/ blue left triangle.

1-Nearest Neighbour in $\mathbf{X}$

Comparison of the Approaches

- Nearest neighbour classification in latent space.

Method	GP-LVM	Isomap	BC-GP-LVM
Errors	226	458	155

cf 24 errors in data space.

Adding Dynamics

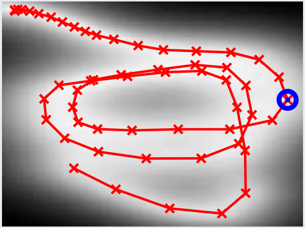
MAP Solutions for Dynamics Models

- Data often has a temporal ordering.
- Markov-based dynamics are often used.
- For the GP-LVM
  - Marginalising such dynamics is intractable.
  - But: MAP solutions are trivial to implement.
- Many choices: Kalman filter, Markov chains *etc.*.
- Wang et al. [2006] suggest using a Gaussian Process.

Gaussian Process Dynamics

GP-LVM with Dynamics

- Gaussian process mapping in latent space between time points.

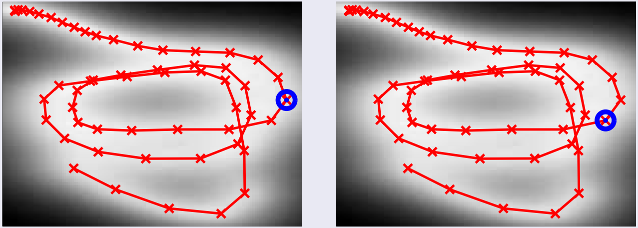


t

Gaussian Process Dynamics

GP-LVM with Dynamics

- Gaussian process mapping in latent space between time points.

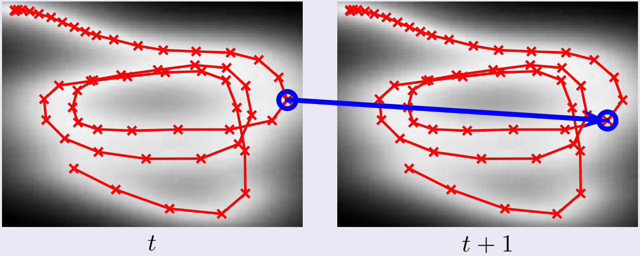


t $\qquad$ $t+1$

Gaussian Process Dynamics

GP-LVM with Dynamics

- Gaussian process mapping in latent space between time points.



Motion Capture Results

demStick1 and demStick2

Figure: The latent space for the motion capture data with (*right*) and without (*left*) back constraints based on an RBF kernel.

Motion Capture Results

demStick1 and demStick2

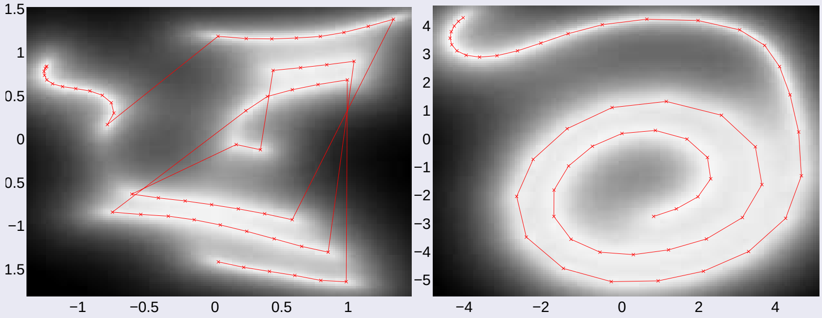


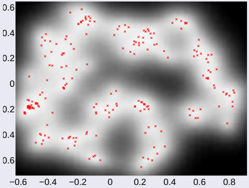
Figure: The latent space for the motion capture data with (*right*) and without (*left*) back constraints based on an RBF kernel.

Robot SLAM I

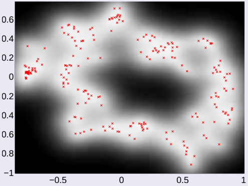
Navigating by WiFi

- Wireless access point signal strengths measured by robot moving around building.
  - 215 separate signal strength readings.
  - 30 separate access points.
- Robot moves in two dimensions so we expect data to be inherently 2-D.
- Learn GP-LVM, GP-LVM with Dynamics, back constrained GP-LVM and back constrained GP-LVM with dynamics.

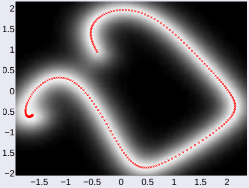
Robot SLAM II



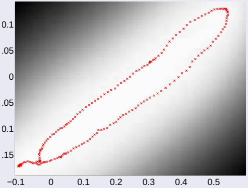
(a) Standard GP-LVM



(b) Standard GP-LVM



(c) Standard GP-LVM



(d) Standard GP-LVM

Summary

- Gaussian Processes are a powerful flexible way to make inference about functions.
  - Applications in graphics, vision, speech, robotics ...
- GP-LVM is a Probabilistic Non-Linear Generalisation of PCA.
- Works Effectively as a Probabilistic Model in High Dimensional Spaces.
- Back constraints can be introduced to force local distance preservation.
- Dynamics can be introduced for modelling data with a temporal structure.
- Applications in graphics, vision, speech, robotics.
- And finally ...

Acknowledgements

Collaborators:

- Vocal Joystick
  - Jeff Bilmes and John Malkin (University of Washington)
- Other ongoing work with (PASCAL EU FP6 Network Funded)
  - Phil Torr, Carl Henrik Ek (Oxford Brookes) and Raquel Urtasun (MIT)
- Face Animation.
  - Manuel Sanchez (now at Electronic Arts).
- Robotics
  - Brian Ferris and Dieter Fox. (University of Washington)

References

Jeff Bilmes, Jonathan Malkin, Xiao Li, Susumu Harada, Kelley Kilanski, Katrin Kirchhoff, Richard Wright, Amarnag Subramanya, James Landay, Patricia Dowden, and Howard Chizeck. The vocal joystick. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*. IEEE, May 2006. To appear.

Christopher M. Bishop and Gwilym D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, A327:580–593, 1993.

Christopher M. Bishop, Marcus Svensén, and Christopher K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10(1):215–234, 1998.

Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic. Style-based inverse kinematics. In *ACM Transactions on Graphics (SIGGRAPH 2004)*, 2004.

Joseph B. Kruskal. Multidimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–28, 1964.

Neil D. Lawrence. Gaussian process models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, Nov 2005.

Neil D. Lawrence and Joaquin Quiñonero Candela. Local distance preservation in the GP-LVM through back constraints. In William Cohen and Andrew Moore, editors, *Proceedings of the International Conference in Machine Learning*, volume 23, pages 513–520. Omnipress, 2006. ISBN 1-59593-383-2.

David Lowe and Michael E. Tipping. Neuroscale: Novel topographic feature extraction with radial basis function networks. In Mozer et al. [1997], pages 543–549.

David J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A*, 354(1):73–80, 1995.

Kantilal V. Mardia, John T. Kent, and John M. Bibby. *Multivariate analysis*. Academic Press, London, 1979.

Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors. *Advances in Neural Information Processing Systems*, volume 9, Cambridge, MA, 1997. MIT Press.

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

Anthony O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society, B*, 40

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.

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

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:

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

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

Conditional Distribution

Deriving the Conditional

- In log space this is equivalent to subtraction of

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

giving

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

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.