Human Motion Analysis
Lecture 4: Dimensionality reduction II

Raquel Urtasun

TTI Chicago

March 15, 2010

Materials used for this lecture

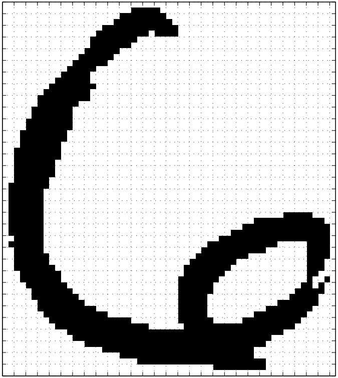
This lecture is based on two

- The ICML 2009 tutorial on dimensionality reduction given by Neil Lawrence. Thanks Neil for your slides!

Why dimensionality reduction

USPS Data Set Handwritten Digit

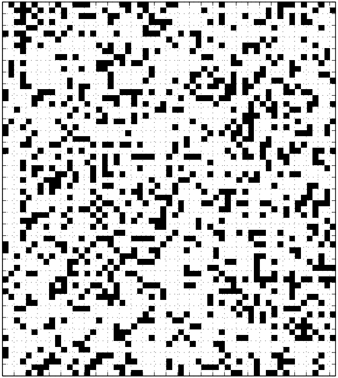
- 3648 Dimensions
  - 64 rows by 57 columns
  - Space contains more than just this digit.



Why dimensionality reduction

USPS Data Set Handwritten Digit

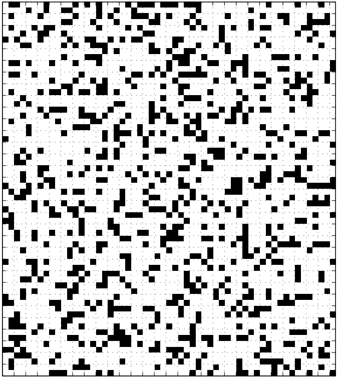
- 3648 Dimensions
  - 64 rows by 57 columns
  - Space contains more than just this digit.
  - Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



Why dimensionality reduction

USPS Data Set Handwritten Digit

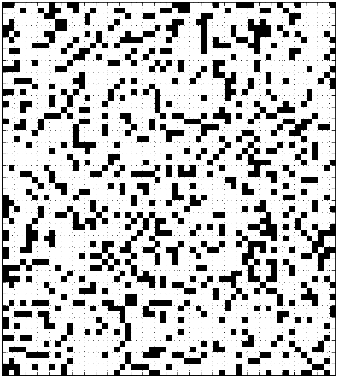
- 3648 Dimensions
  - 64 rows by 57 columns
  - Space contains more than just this digit.
  - Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



Why dimensionality reduction

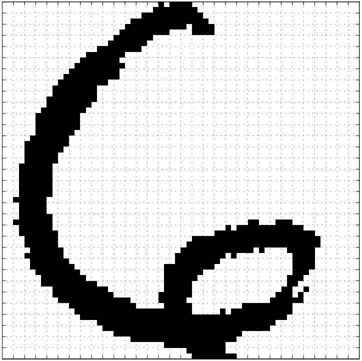
USPS Data Set Handwritten Digit

- 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!



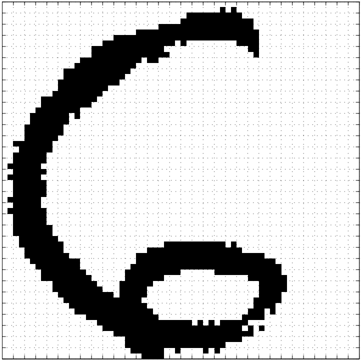
Simple model of a digit

Rotate a 'Prototype'



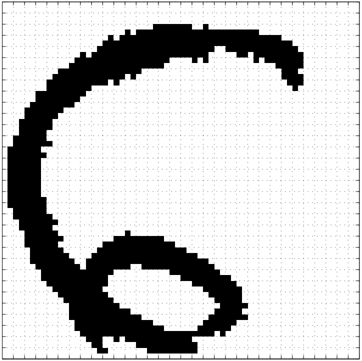
Simple model of a digit

Rotate a 'Prototype'



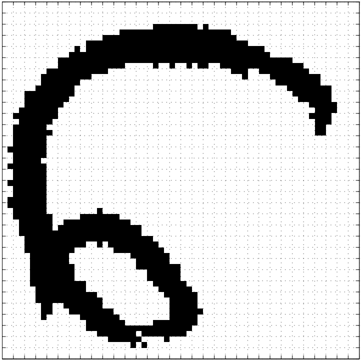
Simple model of a digit

Rotate a 'Prototype'



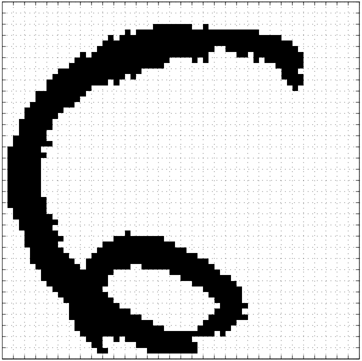
Simple model of a digit

Rotate a 'Prototype'



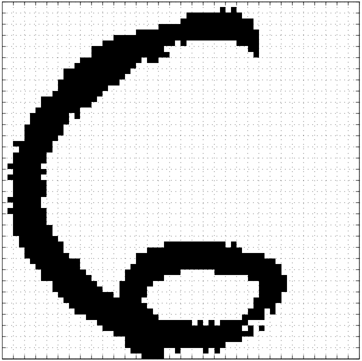
Simple model of a digit

Rotate a 'Prototype'



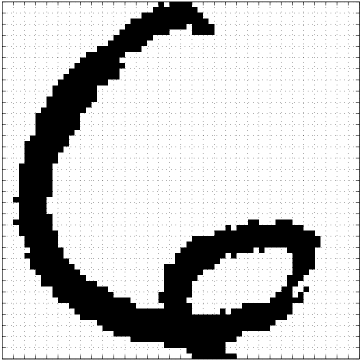
Simple model of a digit

Rotate a 'Prototype'



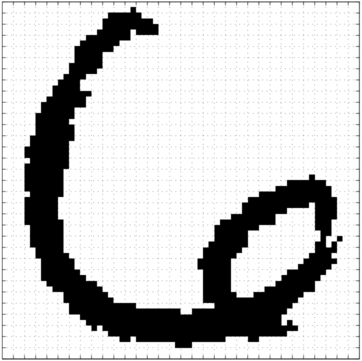
Simple model of a digit

Rotate a 'Prototype'



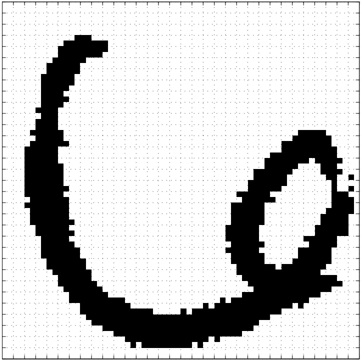
Simple model of a digit

Rotate a 'Prototype'



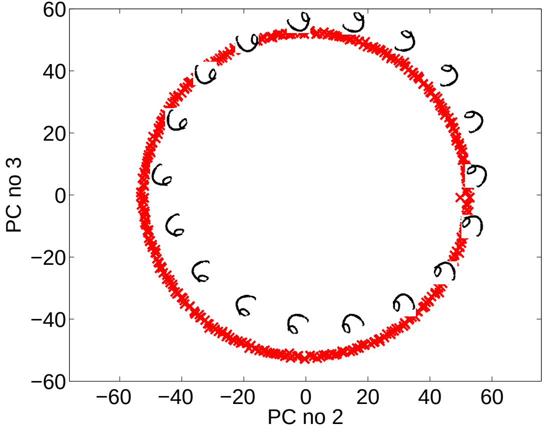
Simple model of a digit

Rotate a 'Prototype'



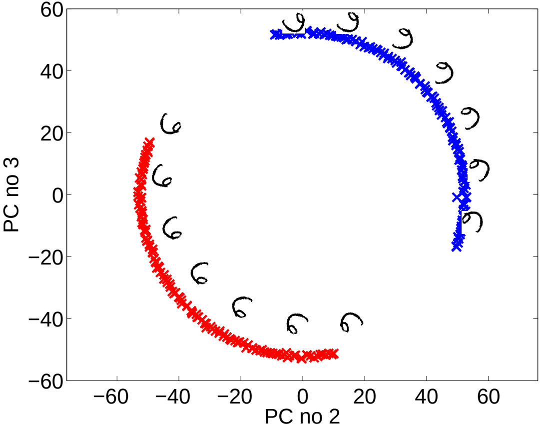
Two dimensional representation

```
demDigitsManifold[1 2], 'all')
```



Two dimensional representation

```
demDigitsManifold([1 2], '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 embedding.

What happened last week?

- How to deal with high-dimensional data.
- We will talk about different dimensionality reduction techniques
  - Linear models: PCA, CCA, etc.
  - Graph based methods: Isomap, Locally linear embedding, laplacian eigenmaps, etc.
  - Latent variable models: GTM and GPLVM
- We will see some examples in practice.

Linear Dimensionality Reduction

- Two dimensional plane projected into a three dimensional space.

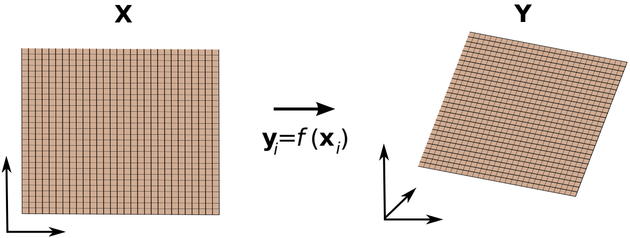


Figure: Mapping a 2D plane to a higher dimensional space in a linear way.

Linear Latent Variable Model

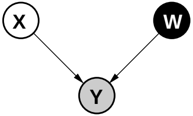
- Represent data, $\mathbf{Y}$, with a lower dimensional set of latent variables $\mathbf{X}$.
- Assume a linear relationship of the form

$$\mathbf{y}_{i,:} = \mathbf{W}\mathbf{x}_{i,:} + \boldsymbol{\eta}_{i,:}, \quad \text{where} \quad \boldsymbol{\eta}_{i,:} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2\mathbf{I}\right).$$

Linear Latent Variable Model

Probabilistic PCA

- Linear-Gaussian relationship between latent variables and data.
- **X** are 'nuisance' variables.

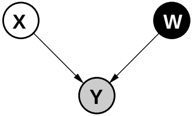


$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

Linear Latent Variable Model

Probabilistic PCA

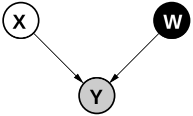
- Linear-Gaussian relationship between latent variables and data.
- **X** are 'nuisance' variables.
- Latent variable model approach:



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

Probabilistic PCA

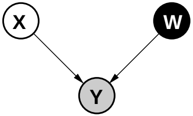
- Linear-Gaussian relationship between latent variables and data.
- **X** are 'nuisance' variables.
- Latent variable model approach:
  - Define Gaussian prior over *latent space*, **X**.



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

Probabilistic PCA

- Linear-Gaussian relationship between latent variables and data.
- **X** are 'nuisance' variables.
- Latent variable model approach:
  - Define Gaussian prior over *latent space*, **X**.
  - Integrate out nuisance *latent variables*.



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

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

Probabilistic PCA

- Linear-Gaussian relationship between latent variables and data.
- $\mathbf{X}$ are 'nuisance' variables.
- Latent variable model approach:
  - Define Gaussian prior over *latent space*, $\mathbf{X}$.
  - Integrate out nuisance *latent variables*.

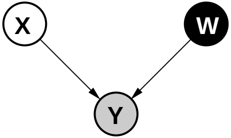
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

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

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

Probabilistic PCA Solution

Probabilistic PCA Max. Likelihood Soln (Tipping and Bishop, 1999b)



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

Probabilistic PCA Solution

Probabilistic PCA Max. Likelihood Soln (Tipping and Bishop, 1999b)

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

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

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

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

where **R** is an arbitrary rotation matrix.

Factor Analysis

- Very similar to PCA, but with a more complex notion of noise:

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \epsilon$$

  with $E\{\epsilon\epsilon^T\} = \Sigma$.

- If the noise is known, then the factors can be estimated using PCA of a modified matrix

$$\mathbf{C} - \Sigma$$

  with $\mathbf{C}$ the covariance matrix of the data.

- If the noise is not know, then there exists different algorithms in the literature to solve this.
- We will not see them in this class.

Why non-linear dimensionality reduction?

- Complex datasets cannot be represented linearly.

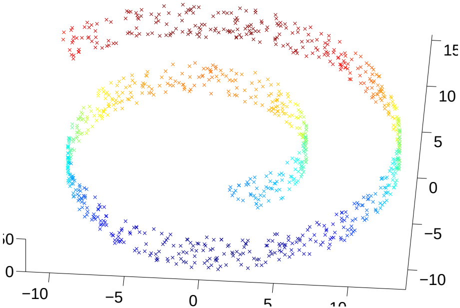


Figure: The 'Swiss Roll' data set is data in three dimensions that is inherently two dimensional.

- We will see non-linear latent variable models and spectral methods.

Non Probabilistic Existing Methods I

Spectral Approaches

- Classical Multidimensional Scaling (MDS) (Mardia et al. 1979) .
  - Uses eigenvectors of similarity matrix.
- Kernel PCA (Scholkopf et al., 1998)
  - Provides a representation and a mapping — representation is high dimensional though!
  - Mapping is implied through the use of a kernel function as a similarity matrix.
- Isomap (Tenenbaum et al., 2000) is MDS with a particular proximity measure.
  - Approximate distances measures along the manifold.
  - Compute neighborhood and compute shortest distance in graph.
  - Use classical MDS on that distance matrix.

Non Probabilistic Existing Methods II

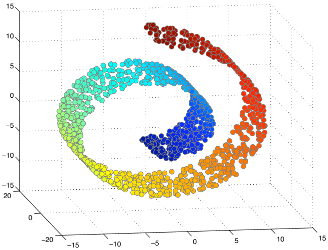
- Locally Linear Embedding (Roweis and Saul, 2000) .
  - Looks to preserve locally linear relationships in a low dimensional space.
  - Compute neighborhood and point find reduced dimensional relationships that preserve local linearity.
- Laplacian Eigenmaps (Belkin and Niyogi, 2003) .
  - Uses spectral graph theory and information geometric arguments to form embedding.
  - Compute neighborhood, graph Laplacian and seek 2nd lowest eigenvector.
- Maximum Variance Unfolding (Weinberger et al., 2004) .
  - Compute neighborhood, constrain local distances to be preserved.
  - Maximise the variance in latent space.

Distance Preservation

Local Distance Preservation

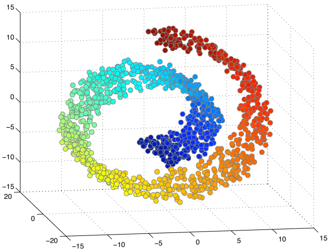
- Most of the above dimensional reduction techniques preserve local distances.
  - Probabilistic Approaches do not.
- Probabilistic approaches map smoothly from latent to data space.
  - Points close in latent space are close in data space.
  - This does not imply points close in data space are close in latent space.
- Spectral approaches map 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.

Locality



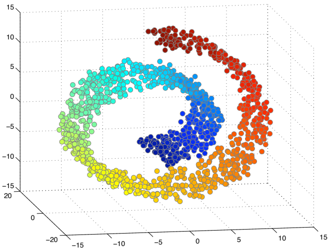
- Algorithms based on local assumption

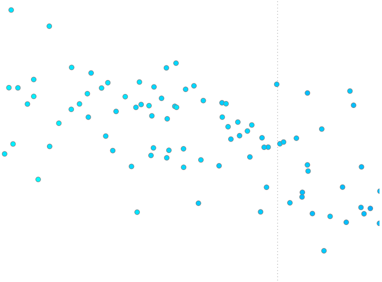
Locality



- Algorithms based on local assumption
- Global noise viewed locally

Locality





- Algorithms based on local assumption
- Global noise viewed locally

Contents of today's lecture

Non-linear latent variable models

- Density networks (MacKay, 1995)
- Generative topographic mapping (GTM) (Bishop et al., 1998a)
- Gaussian process latent variable models (GPLVM) (Lawrence, 2004)
  - Back-constraints (Lawrence et al., 2006)
  - Combining graph-based methods and latent variable models (Urtasun et al., 2008)
  - Automatic determination of dimensionality (Geiger et al., 2009)
  - Hierarchical models (Lawrence et al., 2007)
- Combining linear latent variable models

Non Linear Probabilistic Methods I

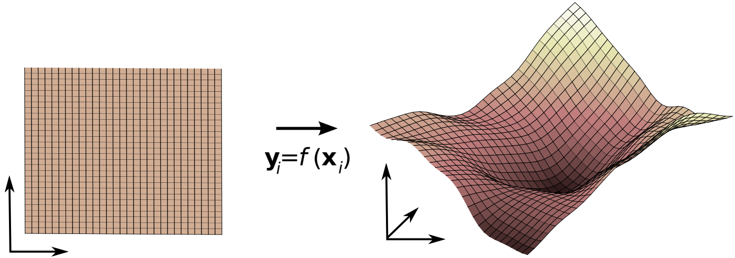


Figure: Mapping a two dimensional plane to a higher dimensional space in a non-linear way.

Non Linear Probabilistic Methods II

Difficulty for Probabilistic Approaches

- Propagate a probability distribution through a non-linear mapping.
- Normalisation of distribution becomes intractable.

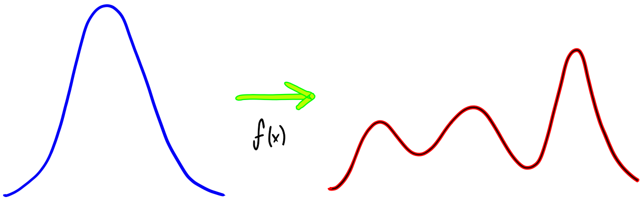


Figure: Gaussian distribution propagated through a non-linear mapping.

Sampling Approach

- Proposed as Density Networks (MacKay, 1995)
- Likelihood is a Gaussian with non-linear mapping from latent space to data space for the mean

$$p(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{N} \prod_{j=1}^{D} \mathcal{N}\left(y_{i,j} | f_j(\mathbf{x}_{i,:}; \boldsymbol{\theta}), \sigma^2\right)$$

$$p(\mathbf{X}) = \mathcal{N}(\mathbf{x}_{i,:} | \mathbf{0}, \mathbf{I})$$

- Take the mapping to be *e.g.* a multi-layer perceptron.
- Key idea: share same samples for all data points $\hat{\mathbf{X}}_n = \hat{\mathbf{X}} = \{\hat{\mathbf{x}}_{k,:}\}_{k=1}^{M}$.
- Saves computation — compute the mapping M times instead of MN

Mapping of Points

- Mapping points to higher dimensions is easy.

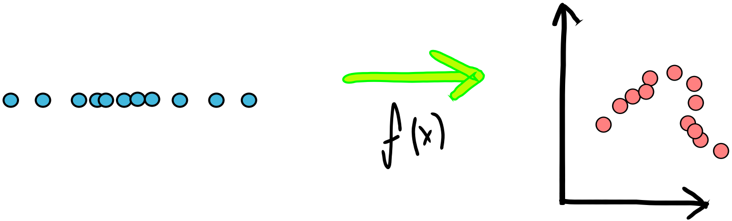


Figure: One dimensional Gaussian mapped to two dimensions.

Mapping of Points

- Mapping points to higher dimensions is easy.

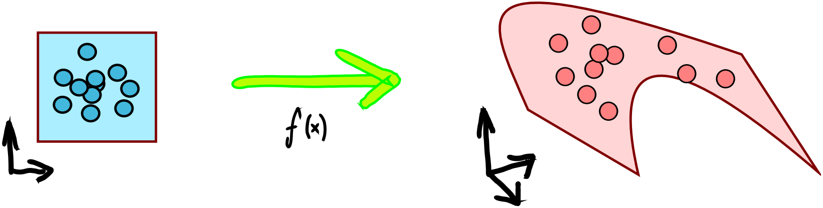


Figure: Two dimensional Gaussian mapped to three dimensions.

Log Likelihood

Sample approximation to log likelihood:

$$\log p\left(\mathbf{Y}|\boldsymbol{\theta}\right) = \sum_{i=1}^{N} \log \frac{1}{M} \sum_{k=1}^{M} p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \bar{\hat{\mathbf{x}}}_{k,:}\right)$$

so we have

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}\right) = \sum_{k=1}^{M} \frac{p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{k,:}\right)}{\sum_{m=1}^{M} p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{m,:}\right)} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{k,:}\right)$$

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}\right) = \sum_{k=1}^{M} \hat{\pi}_{i,k} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p\left(\mathbf{y}_{i,:}|\boldsymbol{\theta}, \hat{\mathbf{x}}_{k,:}\right)$$

Note: $\hat{\pi}_{i,k}$ look a bit like the posterior over component k for data point i.

- Use gradient based optimisation to find the mapping.

Generative Topographic Mapping

- Generative Topographic Mapping (GTM) (Bishop et al., 1998a)
- Key idea: Lay points out on a *grid.*
    - Constrained mixture of Gaussians.

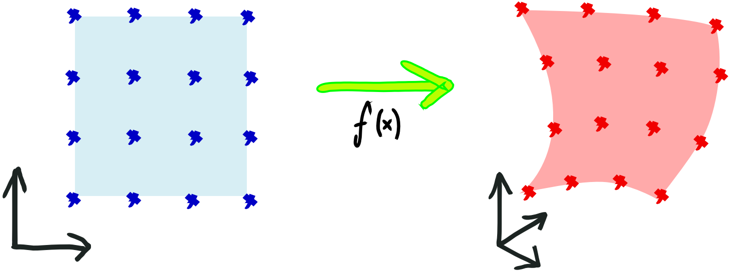


Figure: One dimensional Gaussian mapped to two dimensions.

The GTM Prior

- Prior distribution is a mixture model in a latent space.

$$p(\mathbf{X}) = \prod_{i=1}^{N} p(\mathbf{x}_{i,:})$$

$$p(\mathbf{x}_{i,:}) = \frac{1}{M} \sum_{k=1}^{M} \delta(\mathbf{x}_{i,:} - \hat{\mathbf{x}}_{k,:})$$

- The $\hat{\mathbf{x}}_{k,:}$ are laid out on a regular grid.

Mapping

- Likelihood is a Gaussian with non-linear mapping from latent space to data space for the mean

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} \prod_{j=1}^{D} \mathcal{N}\left(y_{i,j} | f_j\left(\mathbf{x}_{i,:}; \boldsymbol{\theta}, I\right), \sigma^2\right)$$

In the original paper (Bishop et al., 1998b) an RBF network was suggested,

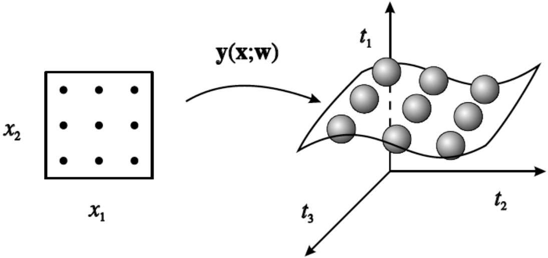
Mapping distribution

- The distribution in data space is

$$p(\mathbf{y}|\theta) = \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{y}|\mathbf{x}_k, \theta)$$

and the log-likelihood becomes

$$\mathcal{L}(\theta) = \sum_{n=1}^{N} \log\left(\frac{1}{M}\sum_{k=1}^{M} p(\mathbf{y}|\hat{\mathbf{x}}_k, \theta)\right)$$



Mapping and E-Step

- Likelihood is a Gaussian with non-linear mapping from latent space to data space for the mean

$$p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} \prod_{j=1}^{D} \mathcal{N}\left(y_{i,j} | f_j(\mathbf{x}_{i,:}; \boldsymbol{\theta}, l), \sigma^2\right)$$

  In the original paper (Bishop et al., 1998b) an RBF network was suggested,

- In the E-step, posterior distribution over k is given by

$$\hat{\pi}_{i,k} = \frac{\prod_{j=1}^{D} \mathcal{N}\left(y_{i,j} | f_j(\hat{\mathbf{x}}_k; \boldsymbol{\theta}, l), \sigma^2\right)}{\sum_{m=1}^{M} \prod_{j=1}^{D} \mathcal{N}\left(y_{i,j} | f_j(\hat{\mathbf{x}}_m; \boldsymbol{\theta}, l), \sigma^2\right)}$$

  sometimes called the "responsibility of component k for data point i".

Likelihood Optimisation

- We then maximise the lower bound on the log likelihood,

$$\log p(\mathbf{y}_{i,:}|\boldsymbol{\theta}) \geq \langle \log p(\mathbf{y}_{i,:}, \hat{\mathbf{x}}_{k,:}|\boldsymbol{\theta})\rangle_{q(k)} - \langle \log q(k)\rangle_{q(k)},$$

- Free energy part of bound

$$\langle \log p(\mathbf{y}_{i,:}, \hat{\mathbf{x}}_{k,:}|\boldsymbol{\theta})\rangle = \sum_{k=1}^{M} \hat{\pi}_{i,k} \log p(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{k,:}, \boldsymbol{\theta}) + \mathrm{const}$$

- When optimising parameters in EM, we ignore dependence of $\hat{\pi}_{i,k}$ on parameters. So we have

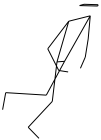
$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \langle \log p(\mathbf{y}_{i,:}, \hat{\mathbf{x}}_{k,:}|\boldsymbol{\theta})\rangle = \sum_{k=1}^{M} \hat{\pi}_{i,k} \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\theta}} \log p(\mathbf{y}_{i,:}|\hat{\mathbf{x}}_{k,:}, \boldsymbol{\theta})$$

  which is very similar to density network result!
- Interpretation of posterior is slightly different.

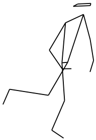
Stick Man Data

- $N = 55$ frames of motion capture.
- xyz locations of 34 points on the body.
- $D = 102$ dimensional data.
- "Run 1" available from `http://accad.osu.edu/research/mocap/mocap_data.htm`.

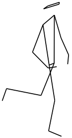
Changing



Angle



of Run



Stick Man Data

`demStickDnet1`

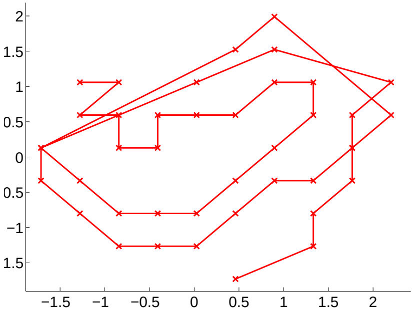


Figure: Stick man data visualised with the GTM using an RBF network with 10×10 points in the grid.

Stick Man Data

demStickDnet2

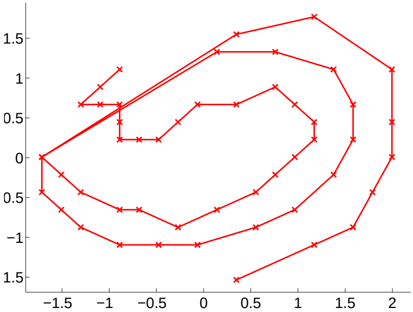


Figure: Stick man data visualised with the GTM using an RBF network with 20×20 points in the grid.

Bubblewrap Effect

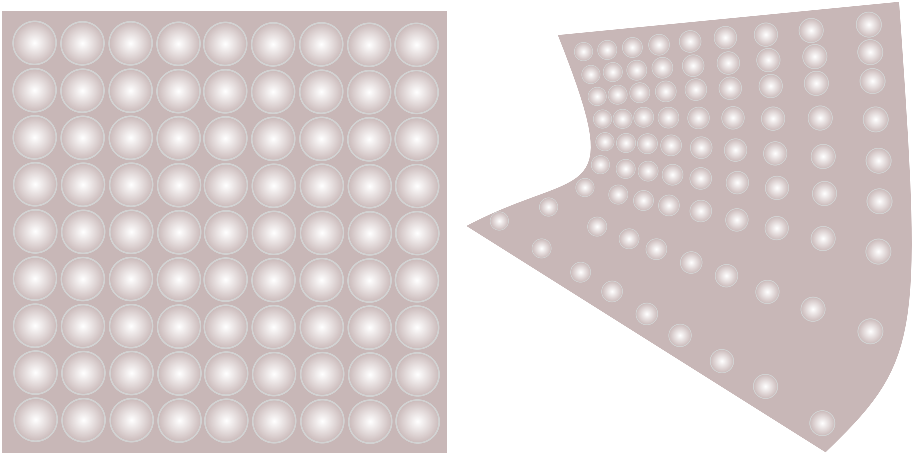


Figure: The manifold is more like bubblewrap than a piece of paper.

Effect of Separated Means

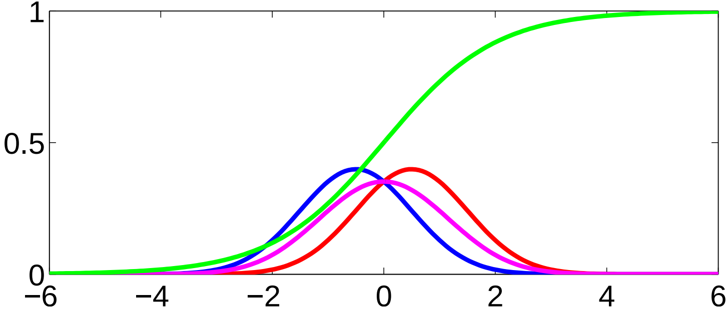


Figure: As Gaussians become further apart the posterior probability becomes more abrupt. 1 standard deviations apart.

Effect of Separated Means

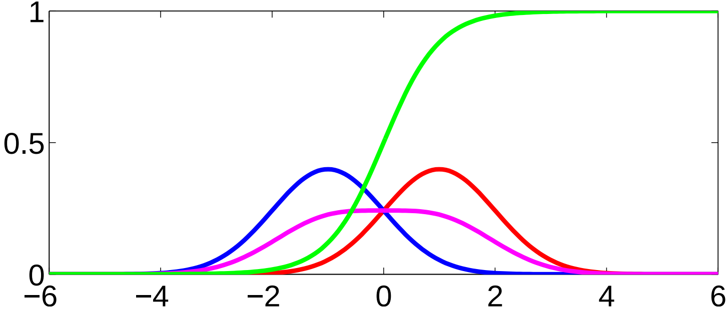


Figure: As Gaussians become further apart the posterior probability becomes more abrupt. 2 standard deviations apart.

Effect of Separated Means

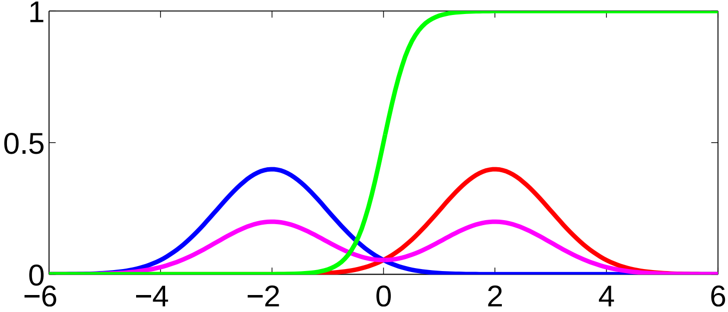


Figure: As Gaussians become further apart the posterior probability becomes more abrupt. 4 standard deviations apart.

Effect of Separated Means

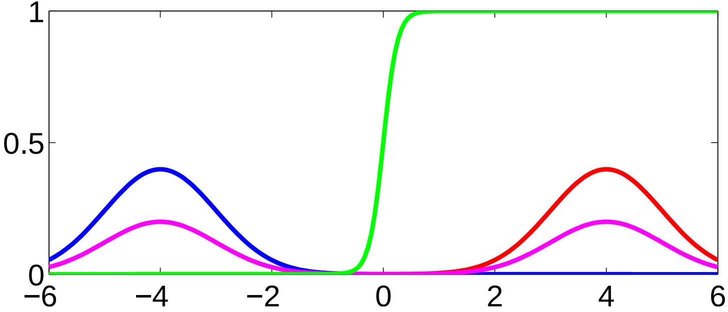


Figure: As Gaussians become further apart the posterior probability becomes more abrupt. 8 standard deviations apart.

Effect of Separated Means

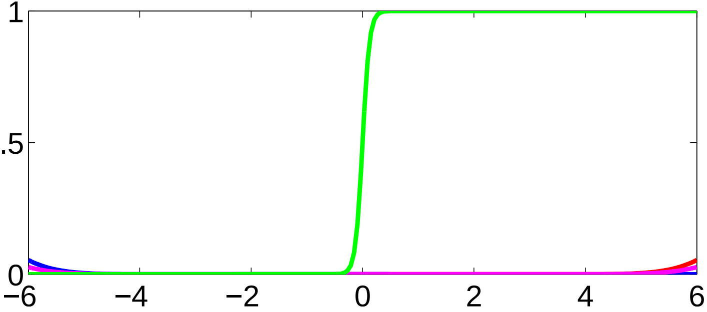


Figure: As Gaussians become further apart the posterior probability becomes more abrupt. 16 standard deviations apart.

Equivalence of GTM and Density Networks

- GTM and Density Networks have the same origin. (Bishop et al. 1996; McKay, 1995).
- In original Density Networks paper MacKay suggested Importance Sampling (MacKay, 1995).
- Early work on GTM also used importance sampling.
- Main innovation in GTM was to lay points out on a grid (inspired by Self Organizing Maps (Kohnonen, 2001).

Summary

- We have explored two point based approaches to dimensionality reduction.
- Approaches seem to generalise well even when dimensions of data is greater than number of points.
- Both approaches are difficult to extend to higher dimensional latent spaces
  - number of samples/centres required increases exponentially with dimension.
- Next we will explore a different probabilistic interpretation of PCA and extend that to non-linear models.

Dual Probabilistic PCA

Probabilistic PCA

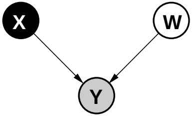
- We have seen that PCA has a probabilistic interpretation (Tipping and Bishop, 1999b) .
- It is difficult to 'non-linearise' directly.
- GTM and Density Networks are an attempt to do so.

Dual Probabilistic PCA

- There is an alternative probabilistic interpretation of PCA (Lawrence, 2005) .
- This interpretation can be made non-linear.
- The result is non-linear probabilistic PCA.

Dual Probabilistic PCA

- Define *linear-Gaussian relationship* between latent variables and data.
  - **Novel** Latent variable approach:

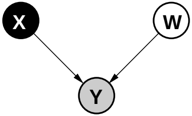


$$p\left(\mathbf{Y}|\mathbf{X},\mathbf{W}\right)=\prod_{i=1}^{N}\mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^{2}\mathbf{I}\right)$$

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 *parameters*, $\mathbf{W}$.

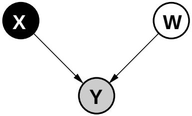


$$p\left(\mathbf{Y}|\mathbf{X},\mathbf{W}\right) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

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 *parameters*, **W**.
  - Integrate out *parameters*.



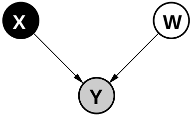
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

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

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 *parameters*, $\mathbf{W}$.
    - Integrate out *parameters*.



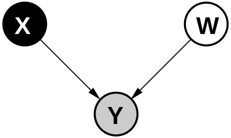
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:}, \sigma^2\mathbf{I}\right)$$

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

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

Linear Latent Variable Model IV

Dual Probabilistic PCA Max. Likelihood Soln (Lawrence, 2004)



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

Linear Latent Variable Model IV

Dual Probabilistic PCA Max. Likelihood Soln (Lawrence, 2004)

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

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

If $\mathbf{U}'_q$ are first q principal eigenvectors of $D^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ and the corresponding eigenvalues are Λ_q,

$$\mathbf{X} = \mathbf{U}'_q\mathbf{L}\mathbf{R}^{\mathrm{T}}, \quad \mathbf{L} = \left(\Lambda_q - \sigma^2\mathbf{I}\right)^{\frac{1}{2}}$$

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

Linear Latent Variable Model IV

Probabilistic PCA Max. Likelihood Soln (Tipping and Bishop, 1999b)

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

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

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

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

where **R** is an arbitrary rotation matrix.

Equivalence of Formulations

The Eigenvalue Problems are equivalent

- Solution for Probabilistic PCA (solves for the mapping)

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

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

$$\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{U}'_q = \mathbf{U}'_q\Lambda_q \qquad \mathbf{X} = \mathbf{U}'_q\mathbf{L}\mathbf{V}^{\mathrm{T}}$$

- Equivalence is from

$$\mathbf{U}_q = \mathbf{Y}^{\mathrm{T}}\mathbf{U}'_q\Lambda_q^{-\frac{1}{2}}$$

Gaussian Process (GP)

Prior for Functions

- Probability Distribution over Functions
- Functions are infinite dimensional.
  - Prior distribution over *instantiations* of the function: finite dimensional objects.
  - Can prove by induction that GP is 'consistent'.
- Mean and Covariance Functions
- Instead of mean and covariance matrix, GP is defined by mean function and covariance function.
  - Mean function often taken to be zero or constant.
  - Covariance function must be *positive definite*.
  - Class of valid covariance functions is the same as the class of *Mercer kernels*.

Gaussian Processes II

Zero mean Gaussian Process

- A (zero mean) Gaussian process likelihood is of the form

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

  where $\mathbf{K}$ is the covariance function or *kernel*.
- The *linear kernel* with noise has the form

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

- Priors over non-linear functions are also possible.
  - To see what functions look like, we can sample from the prior process.

Covariance Samples

`demCovFuncSample`

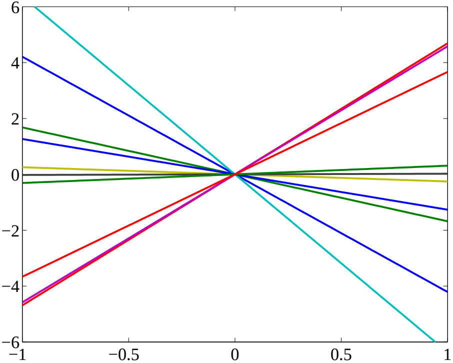


Figure: linear kernel, $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$

Covariance Samples

`demCovFuncSample`

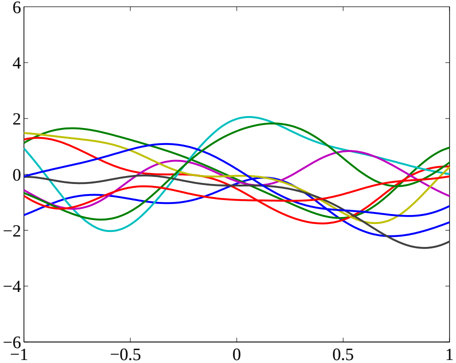


Figure: RBF kernel with $l = 10$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

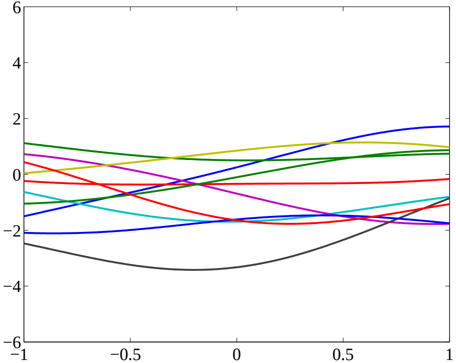


Figure: RBF kernel with $l = 1$, $\alpha = 1$

Covariance Samples

`demCovFuncSample`

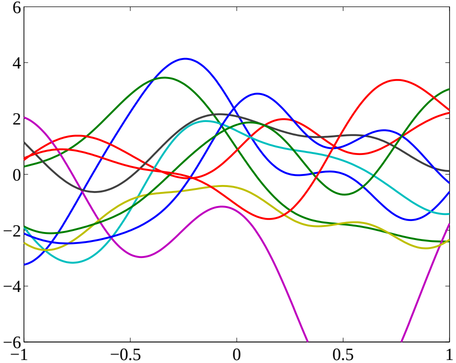


Figure: RBF kernel with $l = 0.3$, $\alpha = 4$

Covariance Samples

`demCovFuncSample`

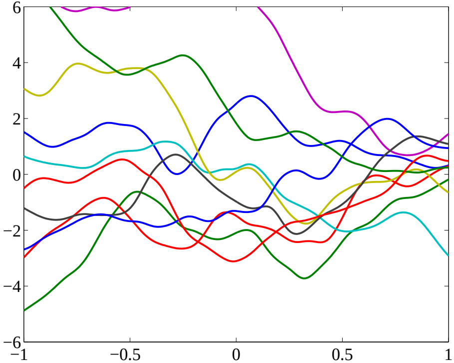


Figure: MLP kernel with $\alpha = 8$, $w = 100$ and $b = 100$

Covariance Samples

`demCovFuncSample`

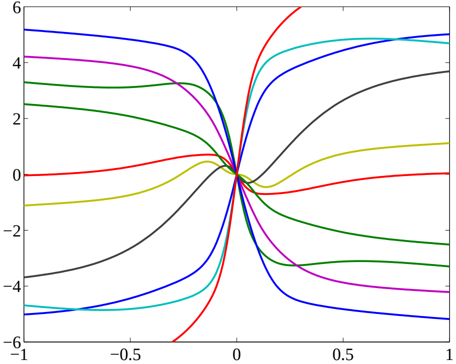


Figure: MLP kernel with $\alpha = 8$, $b = 0$ and $w = 100$

Covariance Samples

`demCovFuncSample`

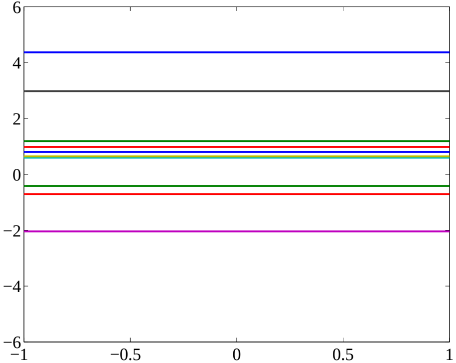


Figure: bias kernel with $\alpha = 1$ and

Covariance Samples

`demCovFuncSample`

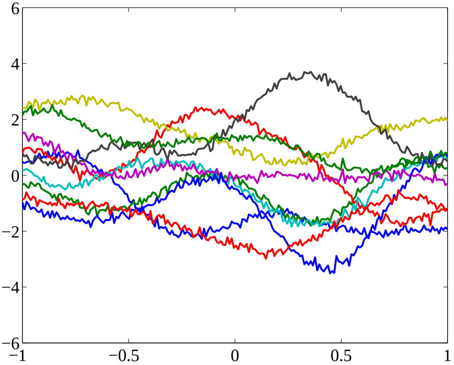


Figure: summed combination of: RBF kernel, $\alpha = 1$, $l = 0.3$; bias kernel, $\alpha = 1$; and white noise kernel, $\beta = 100$

Gaussian Process Regression

Posterior Distribution over Functions

- Gaussian processes are often used for regression.
- We are given a known inputs $\mathbf{X}$ and targets $\mathbf{Y}$.
- We assume a prior distribution over functions by selecting a kernel.
- Combine the prior with data to get a *posterior* distribution over functions.

Gaussian Process Regression

`demRegression`

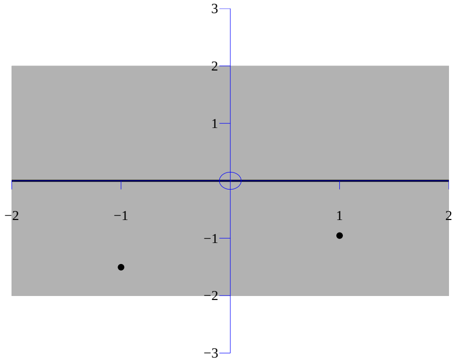


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

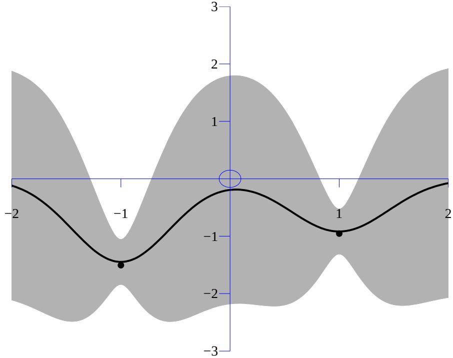


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

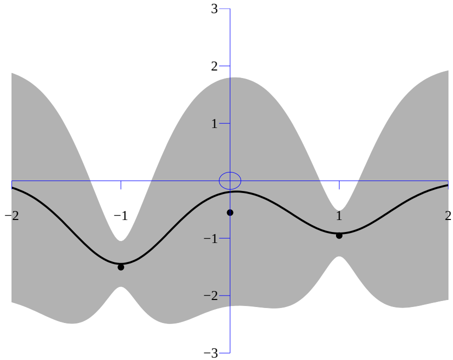


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

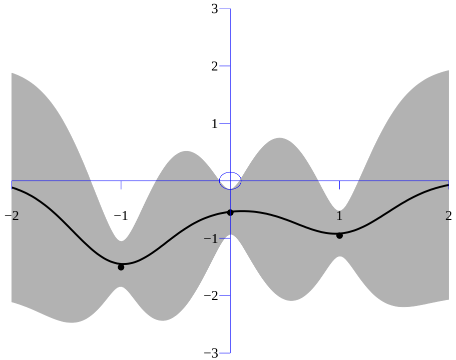


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

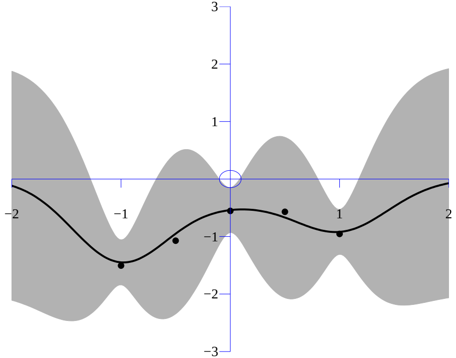


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

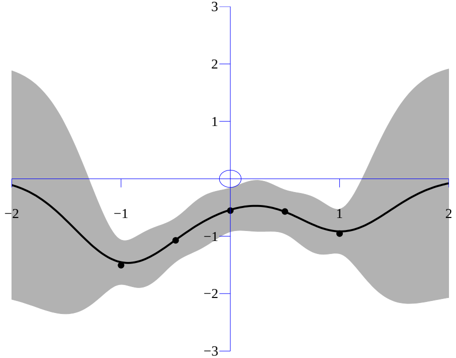


Figure: Examples include WiFi localization, C14 callibration curve.

Gaussian Process Regression

`demRegression`

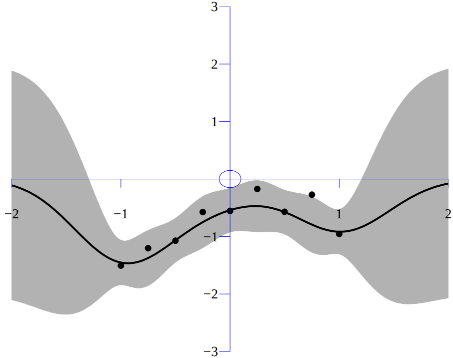


Figure: Examples include WiFi localization, C14 callibration curve.

demRegression

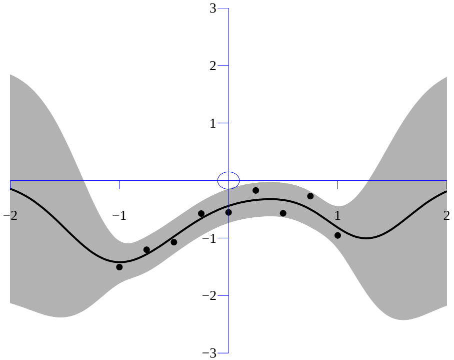
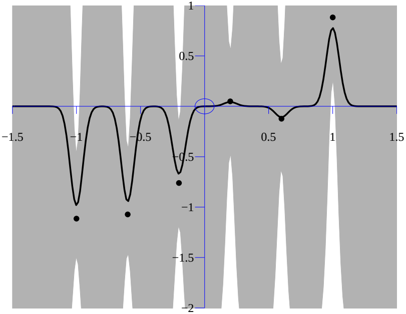


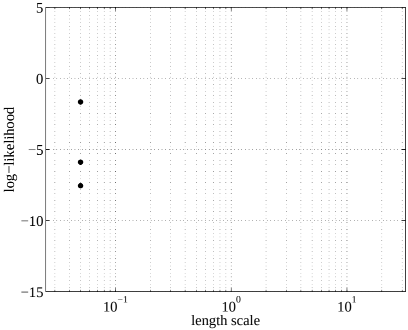
Figure: Examples include WiFi localization, C14 callibration curve.

Learning Kernel Parameters

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

`demOptimiseKern`



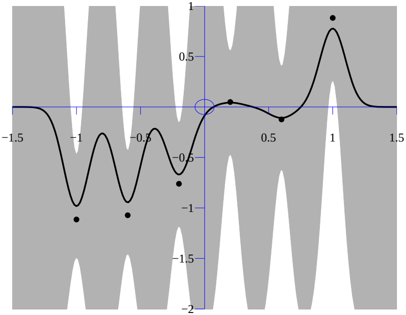


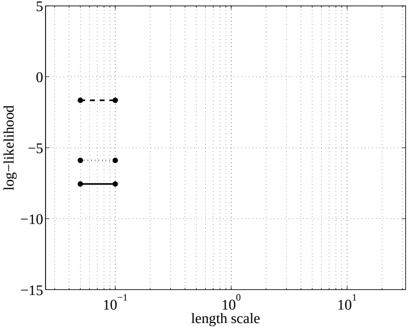
$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)\right)$$

Learning Kernel Parameters

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

`demOptimiseKern`

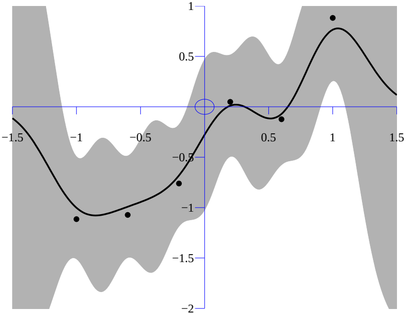


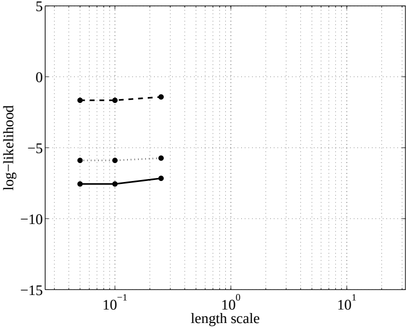


$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)\right)$$

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

`demOptimiseKern`



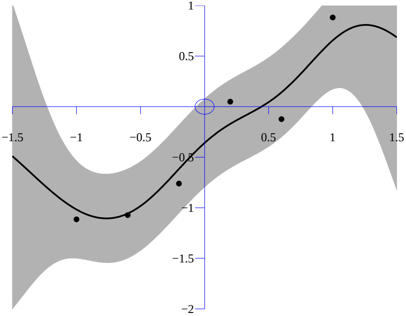


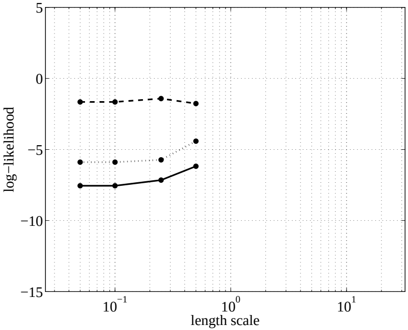
$$\min \left(\frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{T}) \right)$$

Learning Kernel Parameters

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

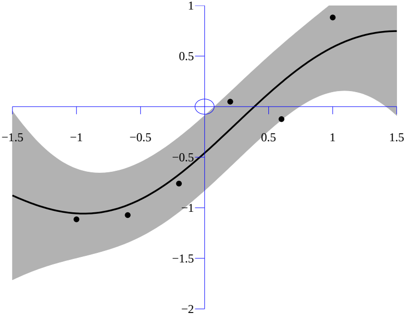
`demOptimiseKern`

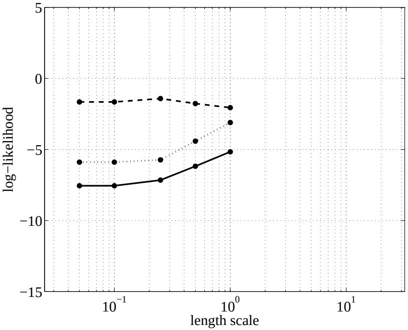




$$\min \left(\frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \right)$$

`demOptimiseKern`

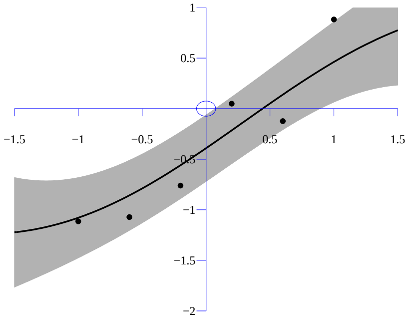


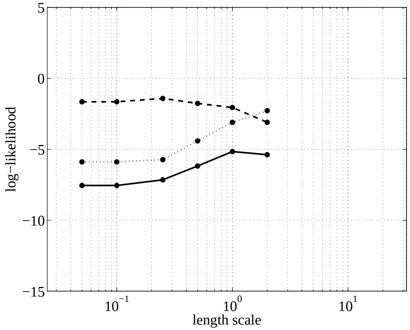


$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)\right)$$

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

`demOptimiseKern`



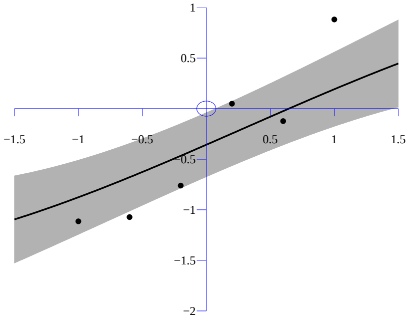


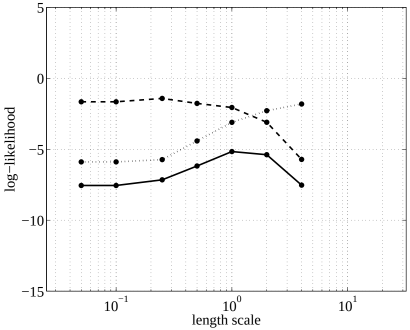
$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{T})\right)$$

Learning Kernel Parameters

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

`demOptimiseKern`



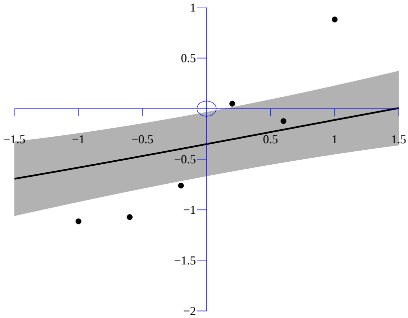


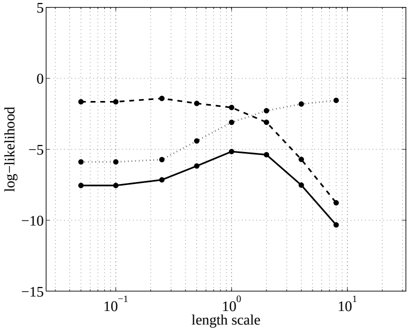
$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{T})\right)$$

Learning Kernel Parameters

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

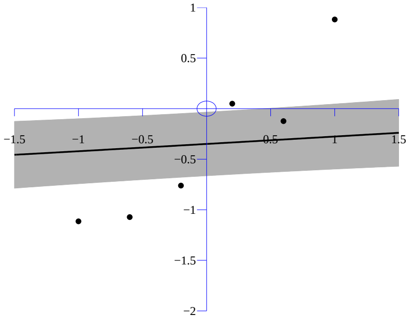
`demOptimiseKern`

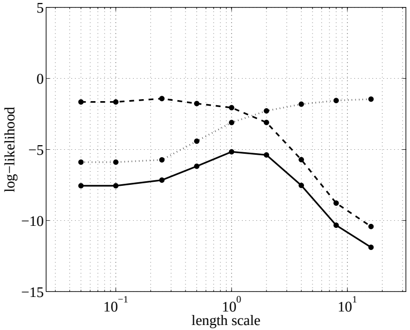




$$\min\left(\frac{D}{2}\ln|\mathbf{K}| + \frac{D}{2}tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)\right)$$

`demOptimiseKern`



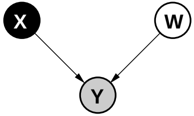


$$\min \left(\frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \right)$$

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*, **W**.
  - Integrate out *parameters*.



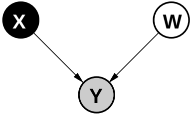
$$p(\mathbf{Y}|\mathbf{X},\mathbf{W}) = \prod_{i=1}^{n} N\left(\mathbf{y}_{i,:}|\mathbf{W}\mathbf{x}_{i,:},\sigma^2\mathbf{I}\right)$$

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

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

Dual Probabilistic PCA

- Inspection of the marginal likelihood shows ...
  - The covariance matrix is a covariance function.

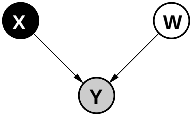


$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{D} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{X}\mathbf{X}^{\mathrm{T}} + \sigma^2\mathbf{I}\right)$$

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'.



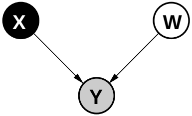
$$p(\mathbf{Y}|\mathbf{X}) = \prod_{j=1}^{D} N\left(\mathbf{y}_{:,j}|\mathbf{0}, \mathbf{K}\right)$$

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

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'.



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

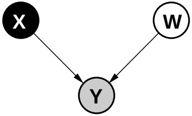
$$\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}} + \sigma^2\mathbf{I}$$

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'.



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

$$\mathbf{K} = ?$$

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

This is called the Gaussian Process Latent Variable Model (GPLVM)

Non-Linear Latent Variable Model

RBF Kernel

- The RBF kernel has the form $k_{i,j} = k\left(\mathbf{x}_{i,:}, \mathbf{x}_{j,:}\right)$, where

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

- 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 gradient methods.

Swiss roll: Initialisation I

'Swiss Roll'

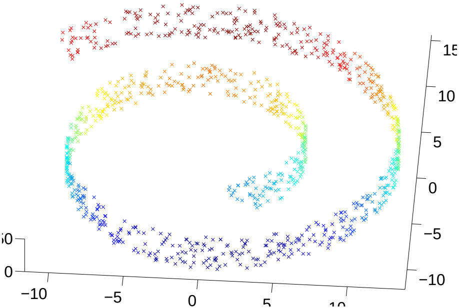


Figure: The 'Swiss Roll' data set is data in three dimensions that is inherently two dimensional.

Swiss Roll: Initialisation II

Quality of solution is Initialisation Dependent

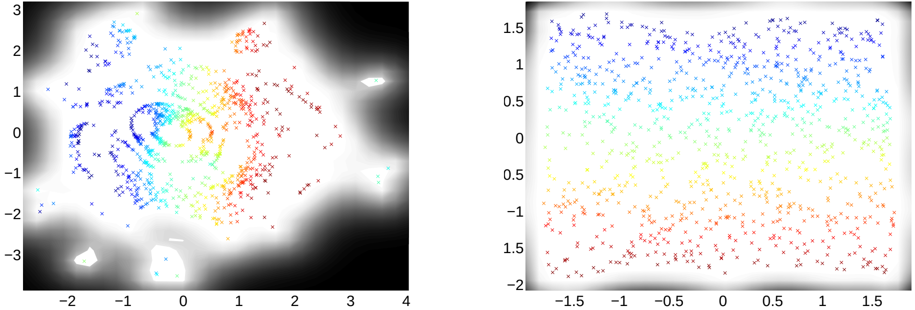


Figure: *Left:* Swiss roll solution initalised by PCA. *Right:* Swiss roll solution initialised by Isomap.

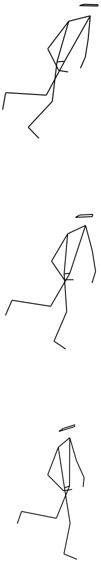
Stick Man Data

- $N = 55$ frames of motion capture.
- xyz locations of 34 points on the body.
- $D = 102$ dimensional data.
- "Run 1" available from `http://accad.osu.edu/research/mocap/mocap_data.htm`.

Changing

Angle

of Run



Stick Man

```
demStick1
```

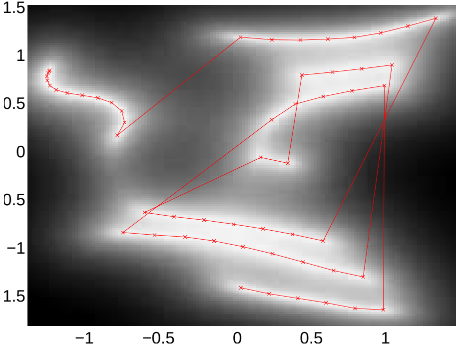


Figure: The latent space for the stick man motion capture data.

Non-smooth latent spaces

Non smooth latent spaces can be avoided by:

- Constrain the forward-mapping: using back-constraints
- Combine graph-based methods and non-linear latent variable models
- Use better optimization schemes that are less prone to get stuck in local minima
- Marginalize the latent space

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 (Lawrence and Quinonero Candela, 2006).
- 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.

`demStick1` **and** `demStick3`

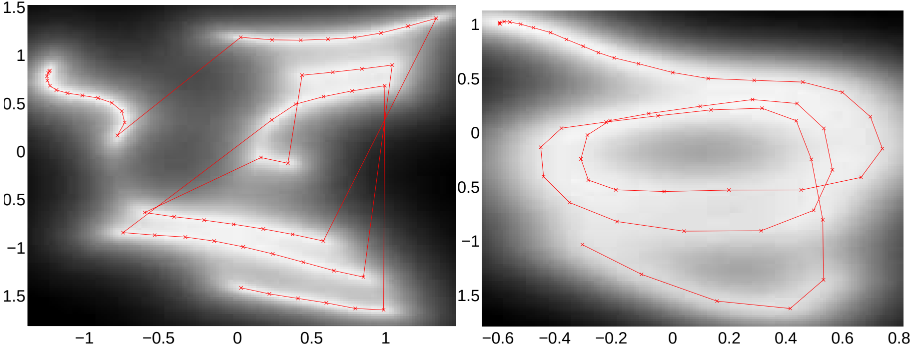
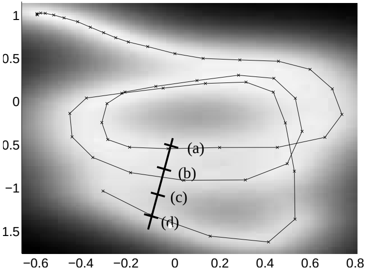


Figure: The latent space for the motion capture data with (*right*) and without (*left*) dynamics. The dynamics us a Gaussian process with an RBF kernel.

`demStickResults`



(a) (b) (c) (d)

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

Incorporating prior knowledge

- It is useful to use prior knowledge when additional information is available, e.g., cyclic motions, smoothness.
- We design priors over the latent space that incorporate the prior knowledge.
- Our prior is based on the Locally Linear Embedding (LLE) [Roweis, 01] cost function

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \lambda \sum_{i=1}^{N} \sum_{q=1}^{d} ||\mathbf{x}_{i,q} - \sum_{j \in \eta_i} w_{ij,q} \mathbf{x}_{j,q}||^2$$

  with $\mathbf{x}_{i,q}$ the q-th dimension of $\mathbf{x}_i$.
- We define the weights to reflect the prior knowledge.
- This is the Locally Linear GPLVM (LL-GPLVM) (Urtasun et al., 2008)

Incorporating prior knowledge

- It is useful to use prior knowledge when additional information is available, e.g., cyclic motions, smoothness.
- We design priors over the latent space that incorporate the prior knowledge.
- Our prior is based on the Locally Linear Embedding (LLE) [Roweis, 01] cost function

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \lambda \sum_{i=1}^{N} \sum_{q=1}^{d} ||\mathbf{x}_{i,q} - \sum_{j \in \eta_i} w_{ij,q} \mathbf{x}_{j,q}||^2$$

  with $\mathbf{x}_{i,q}$ the q-th dimension of $\mathbf{x}_i$.
- We define the weights to reflect the prior knowledge.
- This is the Locally Linear GPLVM (LL-GPLVM) (Urtasun et al., 2008)

Generate animations by sampling

- We learn style-content separation models using the following sources of prior knowledge (Urtasun et al. 2008)
  - smoothness: points close in observation space should be close in latent space.
  - cyclic structure: points with similar phase should be close.
  - transitions: points where a transition could happen should be close in the latent space.

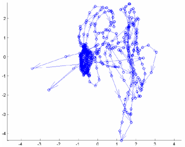


Figure: GPLVM

Figure: Topologies

Figure: Sampling

Problems with the GPLVM

- It relies on the optimization of a non-convex function

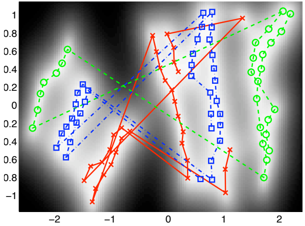
$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{T}) \ .$$

Problems with the GPLVM

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.

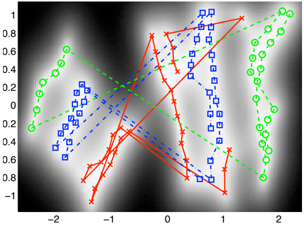


Problems with the GPLVM

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.

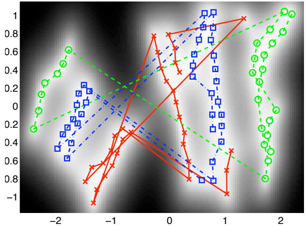


- This is even worst if the dimensionality of the latent space is small.

- It relies on the optimization of a non-convex function

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) \ .$$

- Even with the right dimensionality, they can result in poor representations if initialized far from the optimum.



- This is even worst if the dimensionality of the latent space is small.
- As a consequence this models have only been applied to small databases of a single activity.

Rank priors

- No distortion is introduced by an initialization step; the latent coordinates are initialized to be the original observations

$$\mathbf{X}_{init} = \mathbf{Y}$$

- We introduce a prior over the latent space that encourages latent spaces to be low dimensional.
- Our method is able to estimate the latent space and its dimensionality (Geiger et al., 2009).

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{XX}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.
- We relax the rank minimization and define a prior that encourages sparsity of the eigenvalues, such that:

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \alpha \sum_{i=1}^{D} \phi(s_i)$$

with s_i the eigenvalues of $\bar{\mathbf{X}}\bar{\mathbf{X}}^T$, $\bar{\mathbf{X}}$ the zero-mean $\mathbf{X}$, and ϕ is a function that encourages sparsity.

Continuous dimensionality reduction

- We want to encourage latent space that are low-dimensional.
- Dimensionality can be measure by the rank of $\mathbf{X}\mathbf{X}^T$.
- We would like to penalize the rank, but the rank is a discrete function. The optimization would have to solve a complex combinatorial problem.
- We relax the rank minimization and define a prior that encourages sparsity of the eigenvalues, such that:

$$\mathcal{L} = \frac{D}{2} \ln |\mathbf{K}| + \frac{D}{2} tr(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T) + \alpha \sum_{i=1}^{D} \phi(s_i)$$

with s_i the eigenvalues of $\bar{\mathbf{X}}\bar{\mathbf{X}}^T$, $\bar{\mathbf{X}}$ the zero-mean $\mathbf{X}$, and ϕ is a function that encourages sparsity.

Choice of the penalty function

- Common choice for sparseness is the power family

$$\phi(s_i, p) = |s_i|^p$$

$p = 1$ is a Laplace prior (i.e., L1 norm), which is linear.

- However, our objective function is non-convex. We use a penalty that drives faster to zero the small singular values

$$\phi(s_i) = \log(1 + \beta s_i) \ .$$

Choice of the penalty function

- Common choice for sparseness is the power family

$$\phi(s_i, p) = |s_i|^p$$

$p = 1$ is a Laplace prior (i.e., L1 norm), which is linear.

- However, our objective function is non-convex. We use a penalty that drives faster to zero the small singular values

$$\phi(s_i) = \log(1 + \beta s_i) \ .$$

Estimating the dimensionality

- Minimizing the negative log posterior results in a reduction of the energy of the spectrum. We prevent this by optimizing instead

$$\min \mathcal{L} \quad s.t. \;\; \forall i \; s_i \geq 0, \;\; E(\mathbf{Y}) - E(\mathbf{X}) = 0,$$

  with the energy $E(\mathbf{X}) = \sum_i s_i^2$.

- Finally, we choose the dimensionality to be

$$Q = \text{argmax}_i \frac{s_i}{s_{i+1} + \epsilon}$$

  where $\epsilon \ll 1$, and $s_1 \geq s_2 \geq \cdots \geq s_D$

Estimating the dimensionality

- Minimizing the negative log posterior results in a reduction of the energy of the spectrum. We prevent this by optimizing instead

$$\min \mathcal{L} \quad s.t.\ \forall i\ s_i \geq 0, \quad E(\mathbf{Y}) - E(\mathbf{X}) = 0,$$

  with the energy $E(\mathbf{X}) = \sum_i s_i^2$.
- Finally, we choose the dimensionality to be

$$Q = \text{argmax}_i \frac{s_i}{s_{i+1} + \epsilon}$$

  where $\epsilon \ll 1$, and $s_1 \geq s_2 \geq \cdots \geq s_D$

Dimensionality Estimation Results

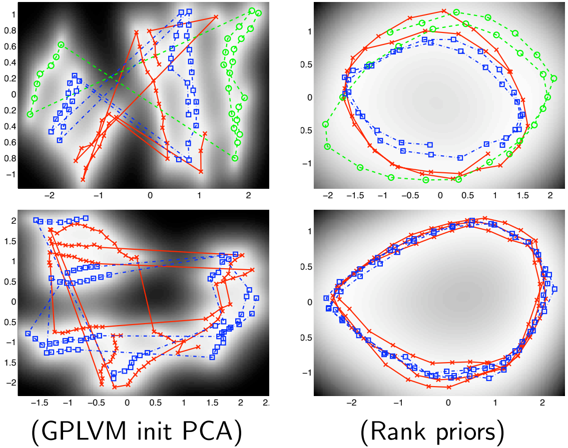


Figure: Running (top) and walking (bottom) models from mocap data. Different subjects are depicted in different colors. Unlike with the GPLVM, the latent coordinates using rank priors are very smooth.

Hierarchical GP-LVM

Stacking Gaussian Processes (Lawrence et al., 2007)

- The input space of the GP is governed by another GP.
- By stacking GPs we can consider more complex hierarchies.
- Ideally we should marginalise latent spaces
    - In practice we seek MAP solutions.

Two Correlated Subjects

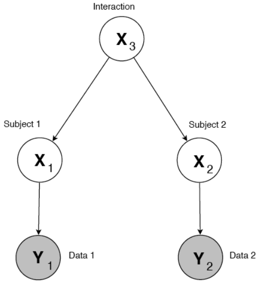


Figure: Hierarchical model of two subjects

We would like to marginalize the latent coordinates

$$p(\mathbf{Y}_1, \mathbf{Y}_2) = \int p(\mathbf{Y}_1|\mathbf{X}_1) \int p(Y_2|\mathbf{X}_2) \int p(\mathbf{X}_1, \mathbf{X}_2|\mathbf{X}_3) d\mathbf{X}_3 d\mathbf{X}_2 \mathbf{X}_1$$

with GP likelihoods

Two Correlated Subjects

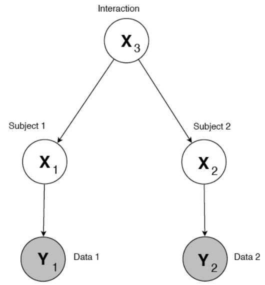


Figure: Hierarchical model of two subjects

Instead do MAP estimation

$$\max\left(\log p(\mathbf{Y}_1|\mathbf{X}_1) + \log p(\mathbf{Y}_2|\mathbf{X}_2) + \log p(\mathbf{X}_1, \mathbf{X}_2|\mathbf{X}_3)\right)$$

with GP likelihoods

Two Correlated Subjects

`demHighFive1`

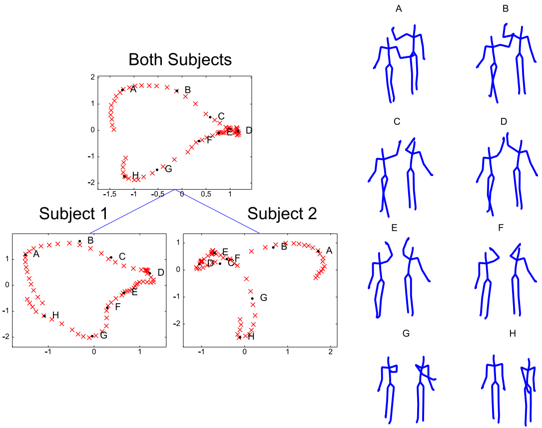


Figure: Hierarchical model of a 'high five'.

Within Subject Hierarchy

Decomposition of Body

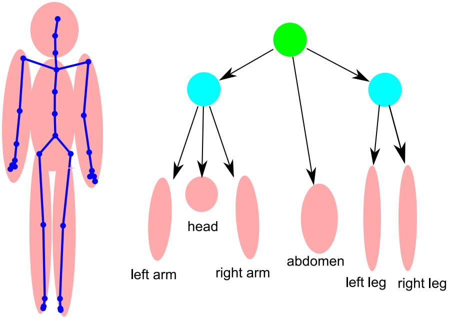


Figure: Decomposition of a subject.

demRunWalk1

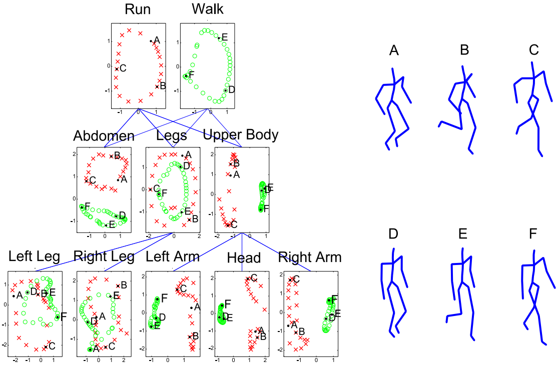


Figure: Hierarchical model of a walk and a run.

Mixture of local models

- For complex data, the manifolds are usually non-linear.
- However, we can characterize these manifolds as locally linear.
- To a good approximation, they can be represented by collections of simpler models, each of which describes a locally linear neighborhood.
- An example of this is a mixture of factor analyzers.

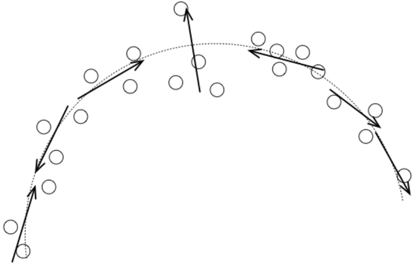


Figure: Mixture of local models

Mixture of factor analyzers

$\mathbf{y}$— observation
s— discrete variable, with $s \in \{1, 2, \cdots, S\}$
$\mathbf{x}_s$— latent representation of the s-th component

- The model is parameterized with a joint distribution

$$p(\mathbf{y}, s, \mathbf{x}_s) = p(\mathbf{y}|s, \mathbf{x}_s)p(\mathbf{x}_s|s)p(s)$$

- The local models are Factor Analyzers

$$p(\mathbf{y}|s, \mathbf{x}_s) = |2\pi\Psi_s|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}[\mathbf{y} - \mu_s - \Lambda_s\mathbf{x}_s]\Psi_s^{-1}[\mathbf{y} - \mu_s - \Lambda_s\mathbf{x}_s]^T\right\}$$

- The marginal distribution $p(\mathbf{y})$ is a mixture of Gaussians.
- This model can be learned using Expectation Maximization (EM) (Ghahramani et al., 1996)

Coordinated mixture of factor analyzers

- The coordinates of neighboring clusters should be similar.
- This is achieved by introducing additional variables **g** that ensure the coordination

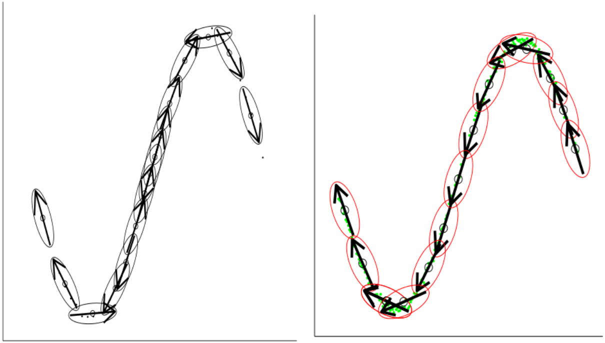


Figure: (**Left**) Mixture of FA. (**right**) Coordinated mixture of FA

Coordinated mixture of factor analyzers II

- Assume a deterministic relationship between local and global variables

$$p(\mathbf{g}|s, \mathbf{x}_s) = \delta(\mathbf{g} - \mathbf{A}_s \mathbf{x}_s - \kappa_s)$$

- We assume that the global coordinates and the data are independent given the mixture component and it's local coodinates $\mathbf{x}_s$
- Introduce additional constraints such that local neighborhood agree on global componets.
- This is achieved by assuring that $p(\mathbf{g}|\mathbf{y}_n)$ is unimodal.

Coordinated mixture of factor analyzers III

- In particular, (Roweis et al., 01) introduced a regularizer that encourage global consitency

$$\Phi = \sum_n \log p(\mathbf{y}_n) - \lambda \sum_{n,s} \int q(\mathbf{g}, s|\mathbf{y}_n) \log \frac{q(\mathbf{g}, s|\mathbf{y}_n)}{p(\mathbf{g}, s|\mathbf{y}_n)}$$

  with q a unimodal family of distributions.
- The regularizer is the sum of Kullback-Leibler (KL) divergences.
- The model is learned using EM.

Why not coordination at the end?

- Noise makes it difficult to coordinate at the end.

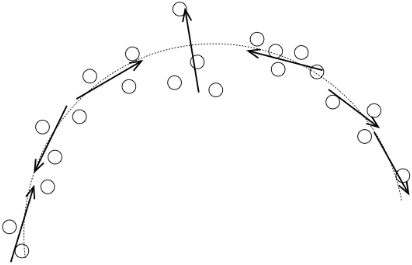


Figure: Problem with late coordination

More?

- If you want to learn more, look at the additional material.
- Otherwise, do the research project on this topic!
- Next week we will do dynamical models.
- Let's do some exercises now!