Dimensionality Reduction

Neil D. Lawrence
neill@cs.man.ac.uk

Mathematics for Data Modelling
University of Sheffield
January 23rd 2008

Outline

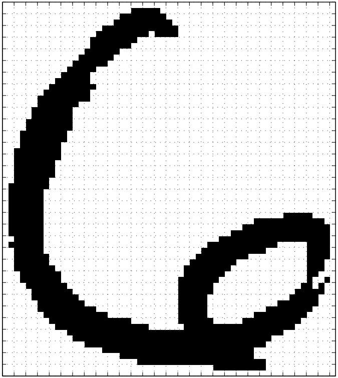
Online Resources

- All source code and slides are available online
- This talk available from my home page (see talks link on left hand side).
- MATLAB examples in the 'dimred' toolbox (vrs 0.1)
  - `http://www.cs.man.ac.uk/~neill/dimred/`.
- MATLAB commands used for examples given in `typewriter font`.

Outline

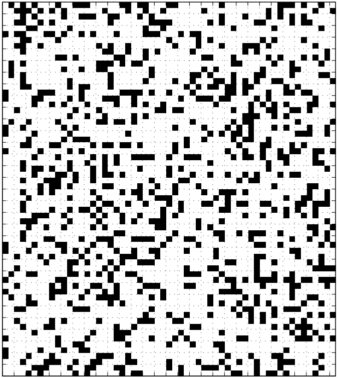
USPS Data Set Handwritten Digit

- 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!



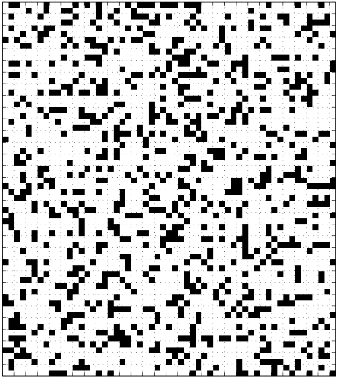
USPS Data Set Handwritten Digit

- 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!



USPS Data Set Handwritten Digit

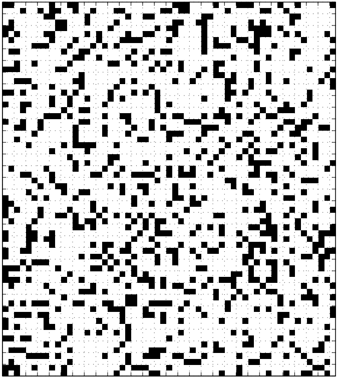
- 3648 Dimensions
- 64 rows by 57 columns
- Space contains more than just this digit.
- Even if we sample every nanosecond from now until the end of the universe, you won't see the original six!



High Dimensional Data

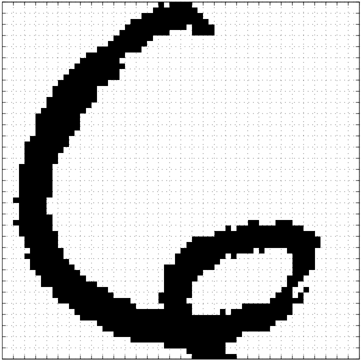
USPS Data Set Handwritten Digit

- 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!



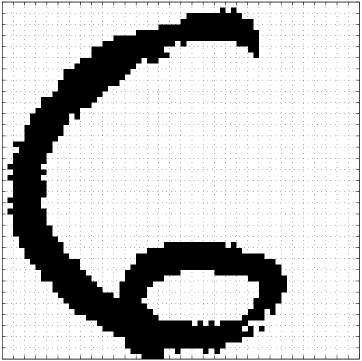
Simple Model of Digit

- Rotate a 'Prototype'

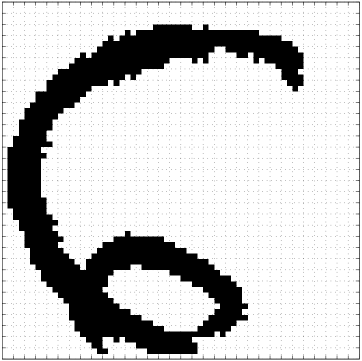


Simple Model of Digit

- Rotate a 'Prototype'

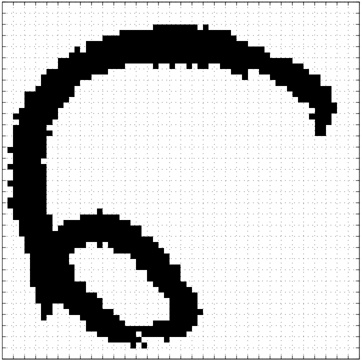


- Rotate a 'Prototype'



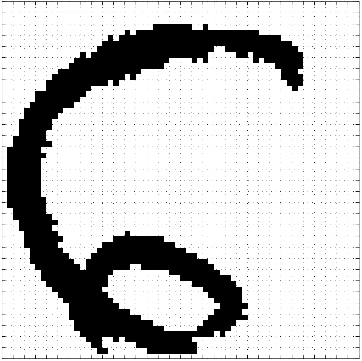
Simple Model of Digit

- Rotate a 'Prototype'



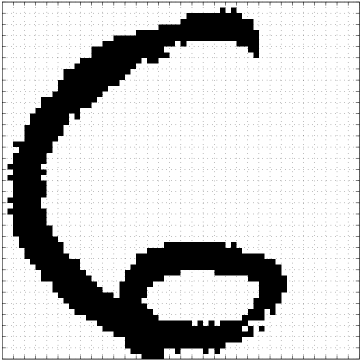
Simple Model of Digit

- Rotate a 'Prototype'



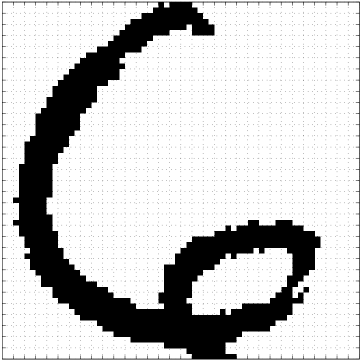
Simple Model of Digit

- Rotate a 'Prototype'



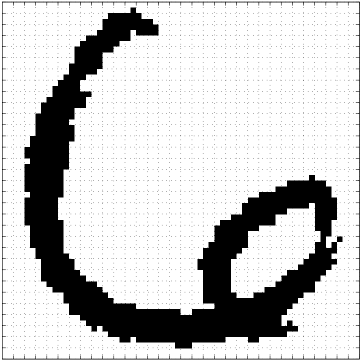
Simple Model of Digit

- Rotate a 'Prototype'

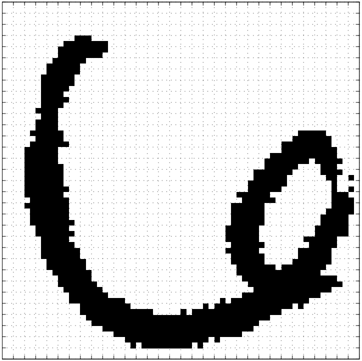


Simple Model of Digit

- Rotate a 'Prototype'



- Rotate a 'Prototype'

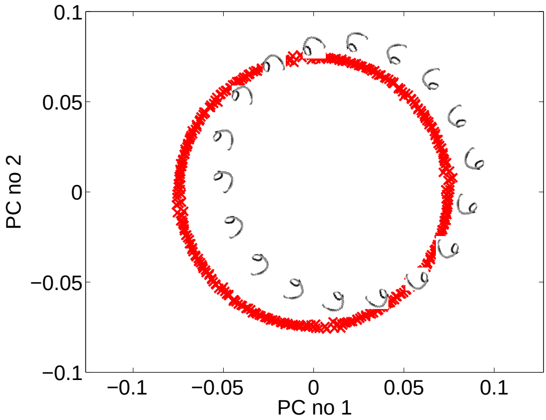


MATLAB Demo

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

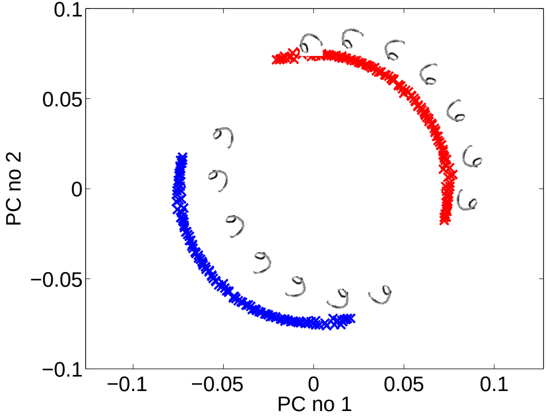
MATLAB Demo

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



MATLAB Demo

```
demDigitsManifold([1 2], 'sixnine')
```



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.

Outline

Notation

q— dimension of latent/embedded space

D— dimension of data space

N— number of data points

data matrix, $\mathbf{Y} = [\mathbf{y}_{1,:}, \ldots, \mathbf{y}_{N,:}]^{\mathrm{T}} = [\mathbf{y}_{:,1}, \ldots, \mathbf{y}_{:,D}] \in \Re^{N \times D}$

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, $\mathbf{W} \in \Re^{D \times q}$

centering matrix, $\mathbf{H} = \mathbf{I} - N^{-1}\mathbf{1}\mathbf{1}^{\mathrm{T}} \in \Re^{N \times N}$

Reading Notation

- $\mathbf{a}_{i,:}$ 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 $\mathbf{A}$.
- $\mathbf{X}$ and $\mathbf{Y}$ are *design matrices.*
- Centred data matrix given by $\hat{\mathbf{Y}} = \mathbf{H}\mathbf{Y}$. ▸ Background
- Sample covariance given by $\mathbf{S} = N^{-1}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}$.
- Centred inner product matrix given by $\mathbf{K} = \hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}$.

Outline

Data Representation

- Classical statistical approach: represent via proximities. [Mardia, 1972]
- Proximity data: similarities or dissimilarities.
- Example of a dissimilarity matrix: a *distance matrix*.

$$d_{i,j} = \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2 = \sqrt{\left(\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right)^{\mathrm{T}} \left(\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right)}$$

- For a data set can display as a matrix.

Interpoint Distances for Rotated Sixes

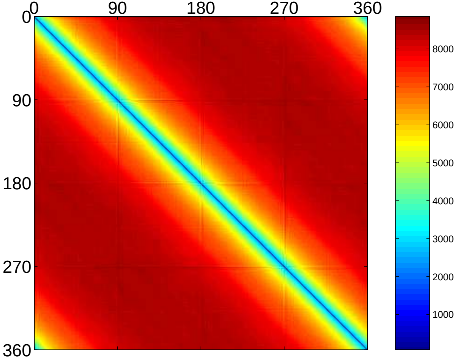


Figure: Interpoint distances for the rotated digits data.

Multidimensional Scaling

- Find a configuration of points, $\mathbf{X}$, such that each

$$\delta_{i,j} = \left\|\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right\|_2$$

  closely matches the corresponding $d_{i,j}$ in the distance matrix.
- Need an objective function for matching $\mathbf{\Delta} = \left(\delta_{i,j}\right)_{i,j}$ to $\mathbf{D} = \left(d_{i,j}\right)_{i,j}$.

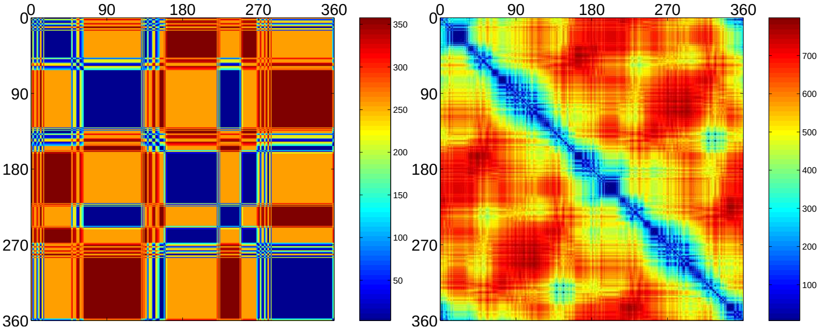
Feature Selection

- An entrywise L_1 norm on difference between squared distances

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left|d_{ij}^2 - \delta_{ij}^2\right|.$$

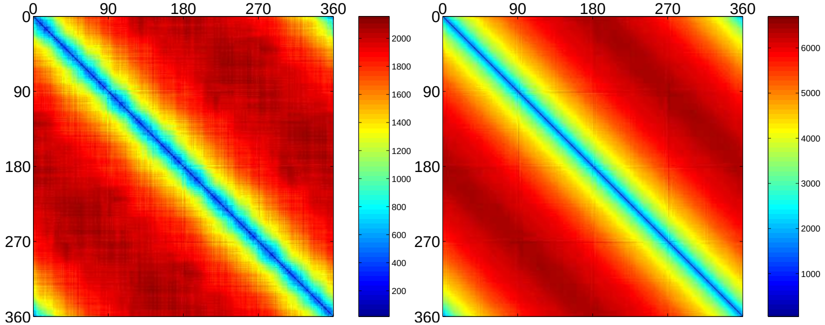
- Reduce dimension by selecting features from data set.
- Select for $\mathbf{X}$, in turn, the column from $\mathbf{Y}$ that most reduces this error until we have the desired q.
- To minimise $E(\mathbf{Y})$ we compose $\mathbf{X}$ by extracting the columns of $\mathbf{Y}$ which have the largest variance. ▸ Derive Algorithm

Reconstruction from Latent Space



Left: distances reconstructed with two dimensions. *Right*: distances reconstructed with 10 dimensions.

Reconstruction from Latent Space



Left: distances reconstructed with 100 dimensions. *Right*: distances reconstructed with 1000 dimensions.

Feature Selection

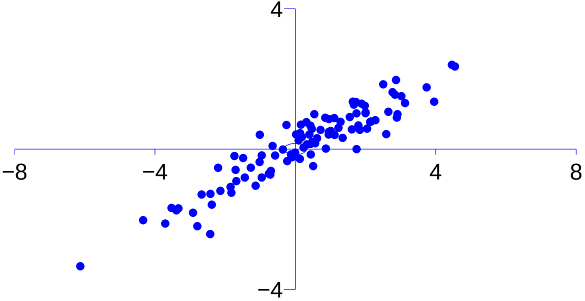


Figure: `demRotationDist`. Feature selection via distance preservation.

Feature Selection

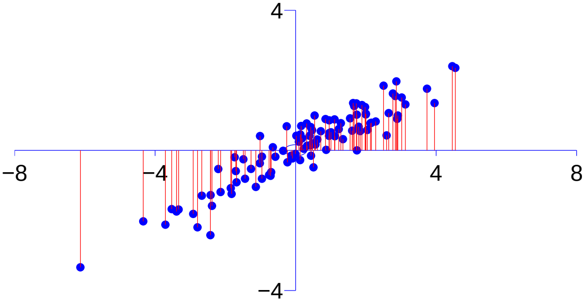


Figure: `demRotationDist`. Feature selection via distance preservation.

Feature Selection

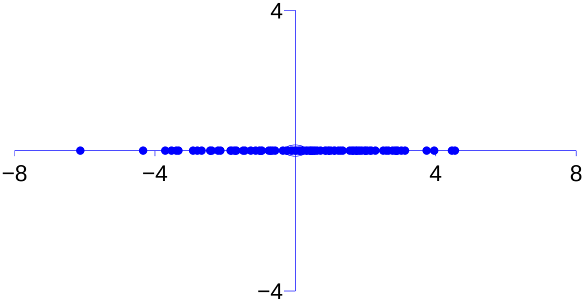


Figure: `demRotationDist`. Feature selection via distance preservation.

Feature Extraction

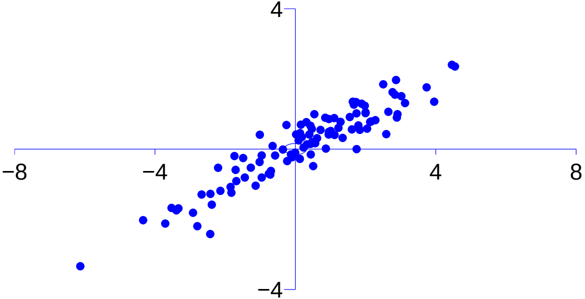


Figure: `demRotationDist`. Rotation preserves interpoint distances. .

Feature Extraction

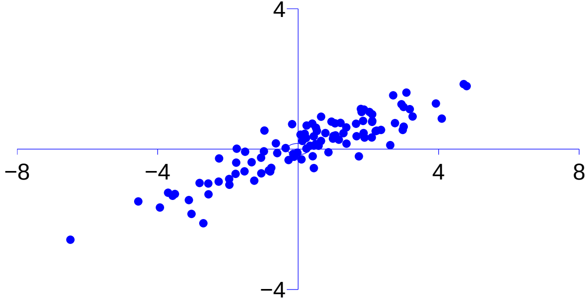


Figure: `demRotationDist`. Rotation preserves interpoint distances. .

Feature Extraction

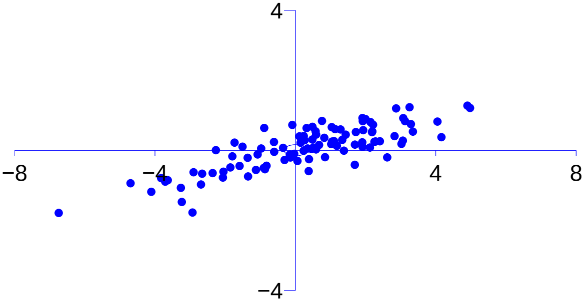


Figure: `demRotationDist`. Rotation preserves interpoint distances. .

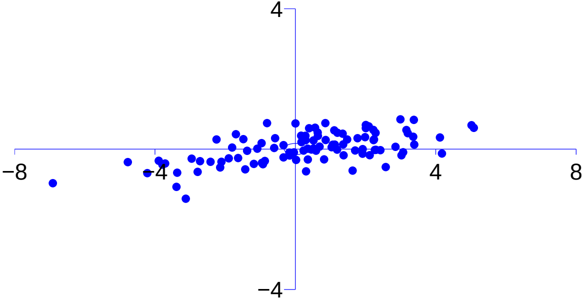


Figure: `demRotationDist`. Rotation preserves interpoint distances. .

Feature Extraction

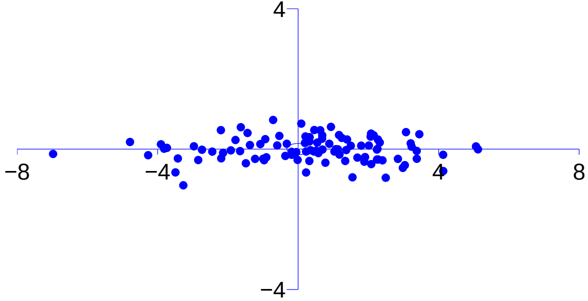


Figure: `demRotationDist`. Rotation preserves interpoint distances. .

Feature Extraction

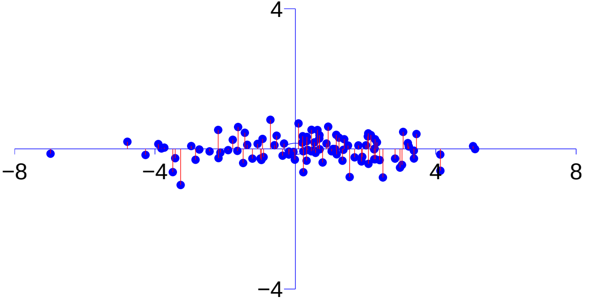


Figure: `demRotationDist`. Rotation preserves interpoint distances. Residuals are much reduced.

Feature Extraction

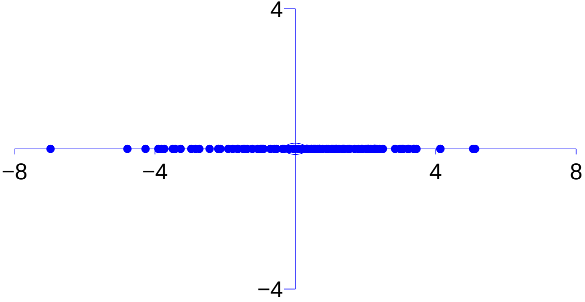


Figure: `demRotationDist`. Rotation preserves interpoint distances. Residuals are much reduced.

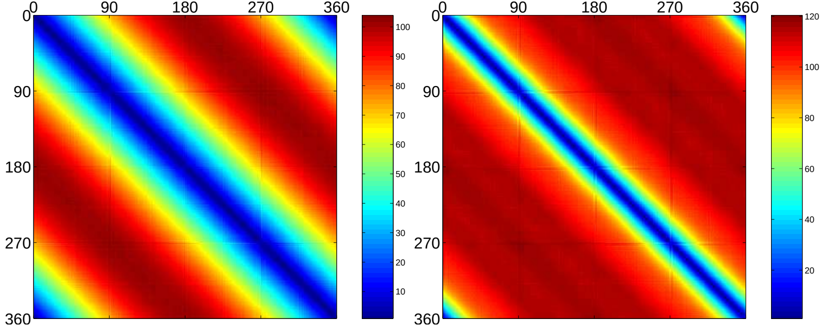
Which Rotation?

- We need the rotation that will minimise residual error.
- We already derived an algorithm for discarding directions.
- Discard direction with *maximum variance*.
- Error is then given by the sum of residual variances.

$$E(\mathbf{X}) = 2N^2 \sum_{k=q+1}^{D} \sigma_k^2.$$

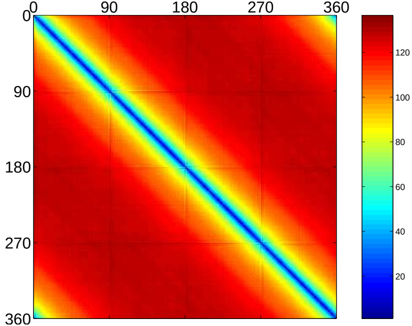
- Rotations of data matrix *do not* effect this analysis.

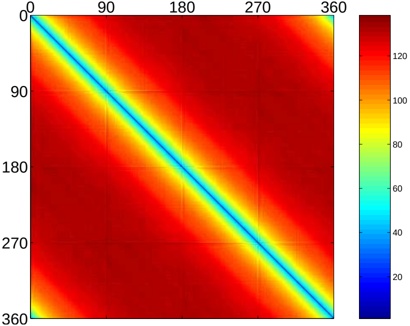
Rotation Reconstruction from Latent Space



Left: distances reconstructed with two dimensions. *Right*: distances reconstructed with 10 dimensions.

Rotation Reconstruction from Latent Space





Left: distances reconstructed with 100 dimensions. *Right*: distances reconstructed with 360 dimensions.

Reminder: Principal Component Analysis

- How do we find these directions?
- Find directions in data with maximal variance.
  - That's what PCA does!
- **PCA**: rotate data to extract these directions.
- **PCA**: work on the sample covariance matrix $\mathbf{S} = N^{-1}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}$.

Principal Component Analysis

- Find a direction in the data, $\mathbf{x}_{:,1} = \hat{\mathbf{Y}}\mathbf{r}_1$, for which variance is maximised.

$$\begin{aligned} \mathbf{r}_1 &= \operatorname{argmax}_{\mathbf{r}_1} \operatorname{var}\left(\hat{\mathbf{Y}}\mathbf{r}_1\right) \\ \text{subject to}: &\quad \mathbf{r}_1^{\top}\mathbf{r}_1 = 1 \end{aligned}$$

- Can rewrite in terms of sample covariance
-
$$\operatorname{var}\left(\mathbf{x}_{:,1}\right) = N^{-1}\left(\hat{\mathbf{Y}}\mathbf{r}_1\right)^{\top}\hat{\mathbf{Y}}\mathbf{r}_1 = \mathbf{r}_1^{\top}\underbrace{\left(N^{-1}\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}}\right)}_{\text{sample covariance}}\mathbf{r}_1 = \mathbf{r}_1^{\top}\mathbf{S}\mathbf{r}_1$$

Lagrangian

- Solution via constrained optimisation:

$$L(\mathbf{r}_1, \lambda_1) = \mathbf{r}_1^{\mathrm{T}}\mathbf{S}\mathbf{r}_1 + \lambda_1\left(1 - \mathbf{r}_1^{\mathrm{T}}\mathbf{r}_1\right)$$

- Gradient with respect to $\mathbf{r}_1$

$$\frac{\mathrm{d}L(\mathbf{r}_1, \lambda_1)}{\mathrm{d}\mathbf{r}_1} = 2\mathbf{S}\mathbf{r}_1 - 2\lambda_1\mathbf{r}_1$$

rearrange to form

$$\mathbf{S}\mathbf{r}_1 = \lambda_1\mathbf{r}_1.$$

Which is recognised as an eigenvalue problem.

Lagrange Multiplier

- Recall the gradient,

$$\frac{\mathrm{d}L\left(\mathbf{r}_1, \lambda_1\right)}{\mathrm{d}\mathbf{r}_1} = 2\mathbf{S}\mathbf{r}_1 - 2\lambda_1\mathbf{r}_1 \tag{1}$$

  to find λ_1 premultiply (1) by $\mathbf{r}_1^{\mathrm{T}}$ and rearrange giving

$$\lambda_1 = \mathbf{r}_1^{\mathrm{T}}\mathbf{S}\mathbf{r}_1.$$

- Maximum variance is therefore *necessarily* the maximum eigenvalue of **S**.
- This is the *first principal component.*

Further Directions

- Find orthogonal directions to earlier extracted directions with maximal variance.
- Orthogonality constraints, for $j < k$ we have

$$\mathbf{r}_j^{\top}\mathbf{r}_k = \mathbf{0} \quad \mathbf{r}_k^{\top}\mathbf{r}_k = 1$$

- Lagrangian

$$L\left(\mathbf{r}_k, \lambda_k, \boldsymbol{\gamma}\right) = \mathbf{r}_k^{\top}\mathbf{S}\mathbf{r}_k + \lambda_k\left(1 - \mathbf{r}_k^{\top}\mathbf{r}_k\right) + \sum_{j=1}^{k-1}\gamma_j\mathbf{r}_j^{\top}\mathbf{r}_k$$

$$\frac{\mathrm{d}L\left(\mathbf{r}_k, \lambda_k\right)}{\mathrm{d}\mathbf{r}_k} = 2\mathbf{S}\mathbf{r}_k - 2\lambda_k\mathbf{r}_k + \sum_{j=1}^{k-1}\gamma_j\mathbf{r}_j$$

Further Eigenvectors

- Gradient of Lagrangian:

$$\frac{\mathrm{d}L\left(\mathbf{r}_k, \lambda_k\right)}{\mathrm{d}\mathbf{r}_k} = 2\mathbf{S}\mathbf{r}_k - 2\lambda_k\mathbf{r}_k + \sum_{j=1}^{k-1}\gamma_j\mathbf{r}_j \tag{2}$$

- Premultipling (2) by $\mathbf{r}_i$ with $i < k$ implies

$$\gamma_i = 0$$

  which allows us to write

$$\mathbf{S}\mathbf{r}_k = \lambda_k\mathbf{r}_k.$$

- Premultiplying (2) by $\mathbf{r}_k$ implies

$$\lambda_k = \mathbf{r}_k^{\mathrm{T}}\mathbf{S}\mathbf{r}_k.$$

- This is the *kth principal component*.

Principal Coordinates Analysis

- The rotation which finds directions of maximum variance is the eigenvectors of the covariance matrix.
- The variance in each direction is given by the eigenvalues.
- **Problem:** working directly with the sample covariance, $\mathbf{S}$, may be impossible.
- For example: perhaps we are given distances between data points, but not absolute locations.
  - No access to absolute positions: cannot compute original sample covariance.

An Alternative Formalism

- Matrix representation of eigenvalue problem for first q eigenvectors.

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \mathbf{R}_q\mathbf{\Lambda}_q \quad \mathbf{R}_q \in \Re^{D\times q} \tag{3}$$

- Premultiply by $\hat{\mathbf{Y}}$:

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \hat{\mathbf{Y}}\mathbf{R}_q\mathbf{\Lambda}_q$$

- Postmultiply by $\mathbf{\Lambda}_q^{-\frac{1}{2}}$

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}} = \hat{\mathbf{Y}}\mathbf{R}_q\mathbf{\Lambda}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

An Alternative Formalism

- Matrix representation of eigenvalue problem for first q eigenvectors.

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \mathbf{R}_q\boldsymbol{\Lambda}_q \quad \mathbf{R}_q \in \Re^{D\times q} \tag{3}$$

- Premultiply by $\hat{\mathbf{Y}}$:

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \hat{\mathbf{Y}}\mathbf{R}_q\boldsymbol{\Lambda}_q$$

- Postmultiply by $\boldsymbol{\Lambda}_q^{-\frac{1}{2}}$

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q\boldsymbol{\Lambda}_q^{-\frac{1}{2}} = \hat{\mathbf{Y}}\mathbf{R}_q\boldsymbol{\Lambda}_q^{-\frac{1}{2}}\boldsymbol{\Lambda}_q$$

An Alternative Formalism

- Matrix representation of eigenvalue problem for first q eigenvectors.

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \mathbf{R}_q\boldsymbol{\Lambda}_q \quad \mathbf{R}_q \in \Re^{D\times q} \tag{3}$$

- Premultiply by $\hat{\mathbf{Y}}$:

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\mathbf{R}_q = \hat{\mathbf{Y}}\mathbf{R}_q\boldsymbol{\Lambda}_q$$

- Postmultiply by $\boldsymbol{\Lambda}_q^{-\frac{1}{2}}$

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{U}_q\boldsymbol{\Lambda}_q \quad \mathbf{U}_q = \hat{\mathbf{Y}}\mathbf{R}_q\boldsymbol{\Lambda}_q^{-\frac{1}{2}}$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\top}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\top}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\top}\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Full eigendecomposition of sample covariance

$$\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top}$$

- Implies that

$$\left(\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}}\right)^2 = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top}\mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\top}.$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Full eigendecomposition of sample covariance

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}$$

- Implies that

$$\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2 = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}.$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Full eigendecomposition of sample covariance

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}$$

- Implies that

$$\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2 = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}.$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Full eigendecomposition of sample covariance

$$\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}$$

- Implies that

$$\left(\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}\right)^2 = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}.$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\top}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\top}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\top}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\top}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Full eigendecomposition of sample covariance

$$\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}} = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top}$$

- Implies that

$$\left(\hat{\mathbf{Y}}^{\top}\hat{\mathbf{Y}}\right)^2 = \mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top}\mathbf{R}\mathbf{\Lambda}\mathbf{R}^{\top} = \mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\top}.$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

  where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.
- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \mathbf{\Lambda}_q^2$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \boldsymbol{\Lambda}_q^{-\frac{1}{2}}\left[\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\boldsymbol{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q\right]\boldsymbol{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \left[\begin{array}{c}\mathbf{I}_q\\ \mathbf{0}\end{array}\right] \in \Re^{D\times q}$$

  where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.
- Premultiplying by eigenvalues gives,

$$\boldsymbol{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \left[\begin{array}{c}\boldsymbol{\Lambda}_q\\ \mathbf{0}\end{array}\right]$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\boldsymbol{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \boldsymbol{\Lambda}_q^2$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\left[\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q\right]\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \mathbf{\Lambda}_q^2$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{\Lambda}_q^{-\frac{1}{2}}\mathbf{\Lambda}_q^2\mathbf{\Lambda}_q^{-\frac{1}{2}}$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \mathbf{\Lambda}_q^2$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\mathbf{U}_q^{\mathrm{T}}\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \boldsymbol{\Lambda}_q$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\boldsymbol{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \boldsymbol{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\boldsymbol{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \boldsymbol{\Lambda}_q^2$$

$\mathbf{U}_q$ Diagonalizes the Inner Product Matrix

- Need to prove that $\mathbf{U}_q$ are eigenvectors of inner product matrix.

$$\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}\mathbf{U}_q = \mathbf{U}_q\mathbf{\Lambda}_q$$

- Product of the first q eigenvectors with the rest,

$$\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{I}_q \\ \mathbf{0} \end{bmatrix} \in \Re^{D\times q}$$

where we have used $\mathbf{I}_q$ to denote a $q \times q$ identity matrix.

- Premultiplying by eigenvalues gives,

$$\mathbf{\Lambda}\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \begin{bmatrix} \mathbf{\Lambda}_q \\ \mathbf{0} \end{bmatrix}$$

- Multiplying by self transpose gives

$$\mathbf{R}_q^{\mathrm{T}}\mathbf{R}\mathbf{\Lambda}^2\mathbf{R}^{\mathrm{T}}\mathbf{R}_q = \mathbf{\Lambda}_q^2$$

Equivalent Eigenvalue Problems

- Two eigenvalue problems are equivalent. One solves for the rotation, the other solves for the location of the rotated points.
- When $D < N$ it is easier to solve for the rotation, $\mathbf{R}_q$. But when $D > N$ we solve for the embedding (principal coordinate analysis).
- In MDS we may not know $\mathbf{Y}$, cannot compute $\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}$ from distance matrix.
- Can we compute $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}$ instead?

The Covariance Interpretation

- $N^{-1}\hat{\mathbf{Y}}^{\mathrm{T}}\hat{\mathbf{Y}}$ is the data covariance.
- $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}$ is a centred inner product matrix.
  - Also has an interpretation as a covariance matrix (Gaussian processes).
  - It expresses correlation and anti correlation between *data points*.
  - Standard covariance expresses correlation and anti correlation between *data dimensions*.

Distance to Similarity: A Gaussian Covariance Interpretation

- Translate between covariance and distance.
  - Consider a vector sampled from a zero mean Gaussian distribution,

$$\mathbf{z} \sim N(\mathbf{0}, \mathbf{K}).$$

  - Expected square distance between two elements of this vector is

$$d_{i,j}^2 = \left\langle (z_i - z_j)^2 \right\rangle$$

$$d_{i,j}^2 = \left\langle z_i^2 \right\rangle + \left\langle z_j^2 \right\rangle - 2 \left\langle z_i z_j \right\rangle$$

under a zero mean Gaussian with covariance given by $\mathbf{K}$ this is

$$d_{i,j}^2 = k_{i,i} + k_{j,j} - 2k_{i,j}.$$

Take the distance to be square root of this,

$$d_{i,j} = \left(k_{i,i} + k_{j,j} - 2k_{i,j}\right)^{\frac{1}{2}}.$$

Standard Transformation

- This transformation is known as the *standard transformation* between a similarity and a distance [Mardia et al., 1979, pg 402].
- If the covariance is of the form $\mathbf{K} = \hat{\mathbf{Y}}\hat{\mathbf{Y}}^{\mathrm{T}}$ then $k_{i,j} = \mathbf{y}_{i,:}^{\mathrm{T}}\mathbf{y}_{j,:}$ and

$$d_{i,j} = \left(\mathbf{y}_{i,:}^{\mathrm{T}}\mathbf{y}_{i,:} + \mathbf{y}_{j,:}^{\mathrm{T}}\mathbf{y}_{j,:} - 2\mathbf{y}_{i,:}^{\mathrm{T}}\mathbf{y}_{j,:}\right)^{\frac{1}{2}} = \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2 .$$

- For other distance matrices this gives us an approach to covert to a similarity matrix or kernel matrix so we can perform classical MDS.

Example: Road Distances with Classical MDS

- Classical example: redraw a map from road distances (see e.g. Mardia et al. 1979).
- Here we use distances across Europe.
  - Between each city we have road distance.
  - Enter these in a distance matrix.
  - Convert to a similarity matrix using the covariance interpretation.
  - Perform eigendecomposition.
- See `http://www.cs.man.ac.uk/~neill/dimred` for the data we used.

Distance Matrix

Convert distances to similarities using “covariance interpretation”.

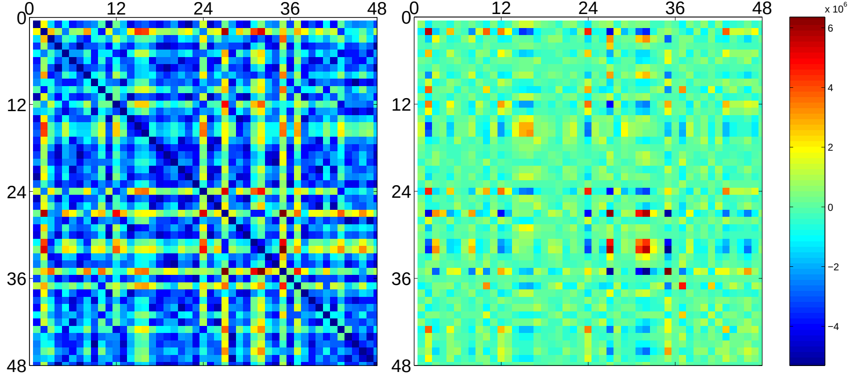


Figure: *Left*: road distances between European cities visualised as a matrix. *Right*: similarity matrix derived from these distances. If this matrix is a covariance matrix, then expected distance between samples from this covariance is given on the *left*.

Example: Road Distances with Classical MDS

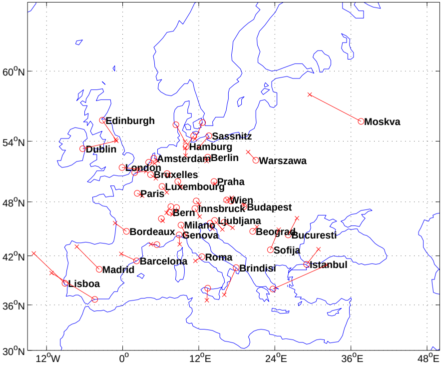


Figure: `demCmdsRoadData`. Reconstructed locations projected onto true map using Procrustes rotations.

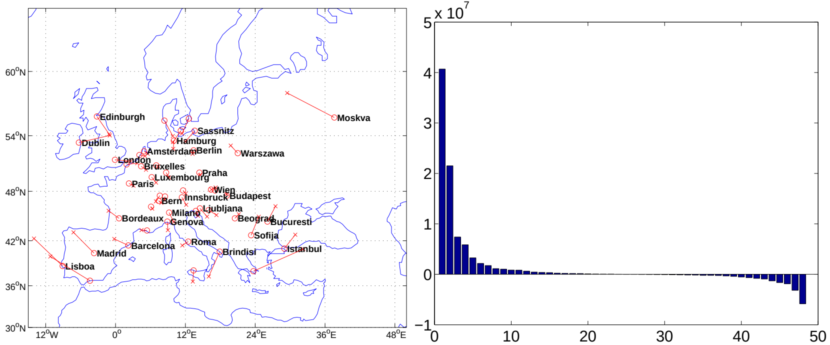


Figure: Eigenvalues of the similarity matrix are negative in this case.

European Cities Distance Matrices

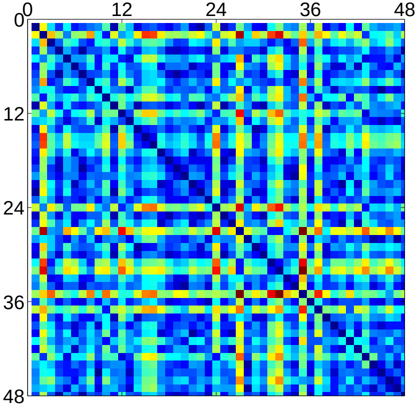


Figure: *Left*: the original distance matrix. *Right*: the reconstructed distance matrix.

Other Distance Similarity Measures

- Can use similarity/distance of your choice.
- Beware though!
  - The similarity must be positive semi definite for the distance to be Euclidean.
  - Why? Can immediately see positive definite is sufficient from the "covariance intepretation".
  - For more details see [Mardia et al., 1979, Theorem 14.2.2].

A Class of Similarities for Vector Data

- All Mercer kernels are positive semi definite.
- Example, squared exponential (also known as RBF or Gaussian)

$$k_{i,j} = \exp\left(-\frac{\left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|^2}{2l^2}\right).$$

  This leads to a kernel eigenvalue problem.
- This is known as Kernel PCA Schölkopf et al. 1998.

Implied Distance Matrix

- What is the equivalent distance $d_{i,j}$?

$$d_{i,j} = \sqrt{k_{i,i} + k_{j,j} - 2k_{i,j}}$$

- If point separation is large, $k_{i,j} \rightarrow 0$. $k_{i,i} = 1$ and $k_{j,j} = 1$.

$$d_{i,j} = \sqrt{2}$$

- Kernel with RBF kernel projects along axes PCA can produce poor results.
- Uses many dimensions to keep dissimilar objects a constant amount apart.

Implied Distances on Rotated Sixes

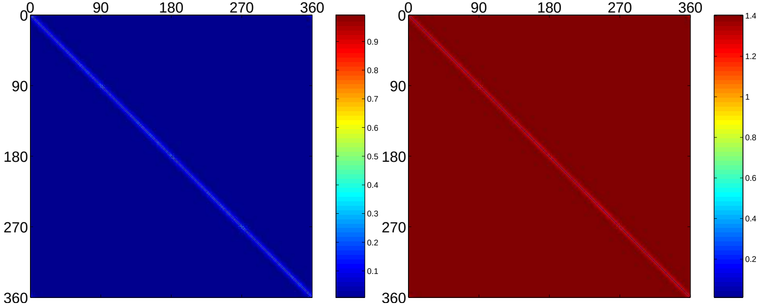


Figure: *Left*: similarity matrix for RBF kernel on rotated sixes. *Right*: implied distance matrix for kernel on rotated sixes. Note that most of the distances are set to $\sqrt{2} \approx 1.41$.

Kernel PCA on Rotated Sixes

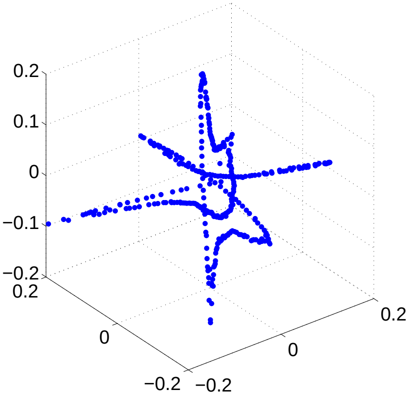


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

Kernel PCA on Rotated Sixes

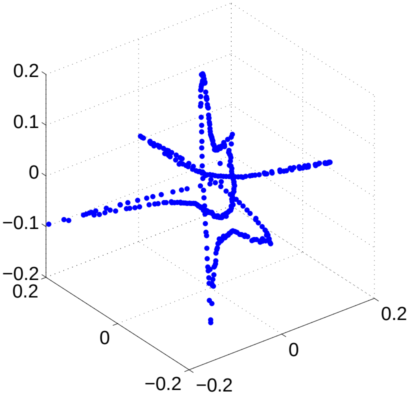


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

Kernel PCA on Rotated Sixes

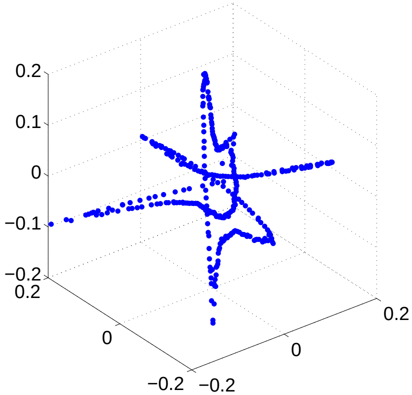


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

Kernel PCA on Rotated Sixes

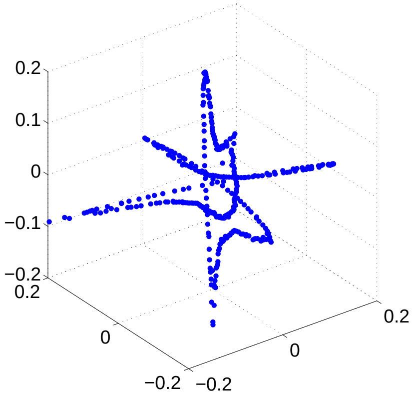


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

Kernel PCA on Rotated Sixes

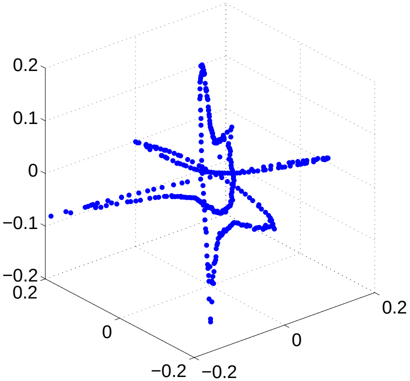


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

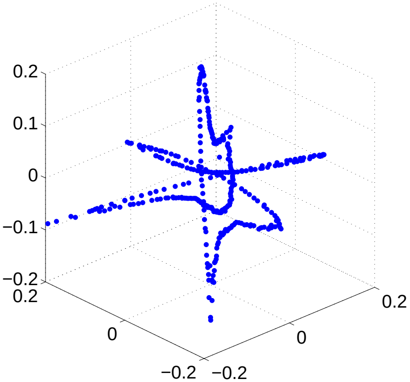
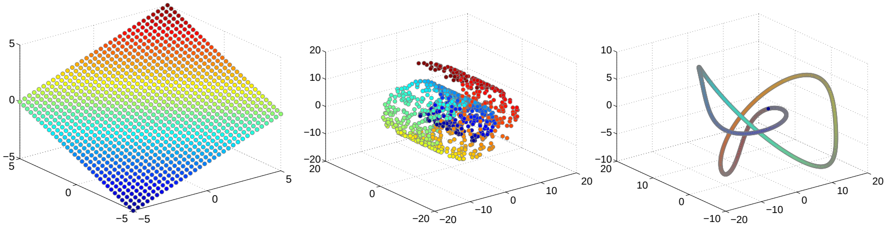


Figure: `demSixKpca`. The fifth, sixth and seventh dimensions of the latent space for kernel PCA. Points spread out along axes so that dissimilar points are always $\sqrt{2}$ apart.

MDS Conclusions

- Multidimensional scaling: preserve a distance matrix.
- Classical MDS
  - a particular objective function
  - for Classical MDS distance matching is equivalent to maximum variance
  - spectral decomposition of the similarity matrix
- For Euclidean distances in $\mathbf{Y}$ space classical MDS is equivalent to PCA.
  - known as principal coordinate analysis (PCO)
- Haven't discussed choice of distance matrix.

Outline



(a) `'plane'` (b) `'swissroll'` (c) `'trefoil'`

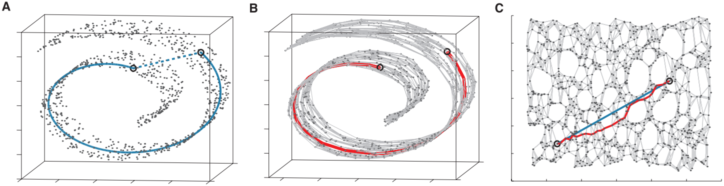
Figure: Illustrative data sets for the talk. Each data set is generated by calling `generateManifoldData(dataType)`. The `dataType` argument is given below each plot.

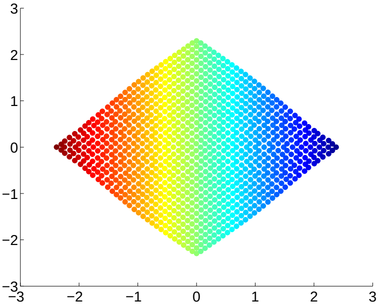
Isomap

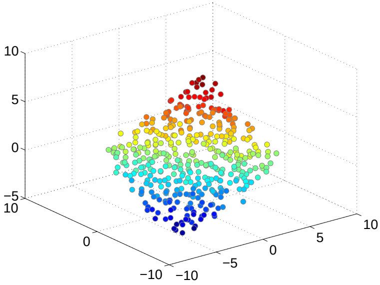
- *Tenenbaum et al. 2000*
- MDS finds geometric configuration preserving distances
- MDS applied to Manifold distance
- Geodesic Distance = Manifold Distance
- Cannot compute geodesic distance without knowing manifold

Isomap

- Isomap: define neighbours and compute distances between neighbours.
- Geodesic Distance approximated by shortest path through adjacency matrix.



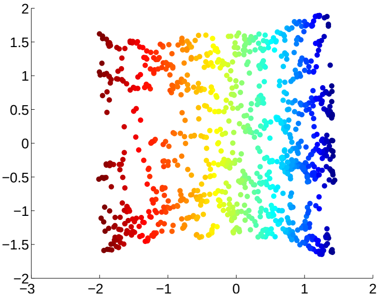


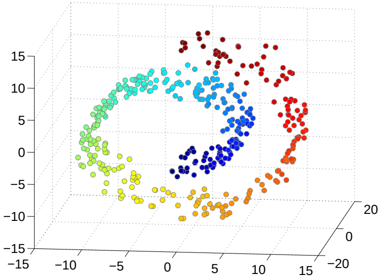


demIsomap

[1] Data generation Carl Henrik Ek

Isomap Examples[1]

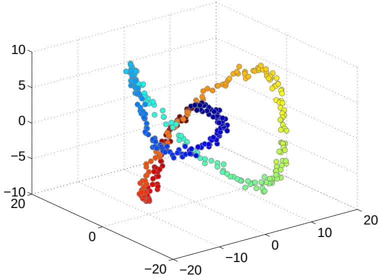




demIsomap

[1] Data generation Carl Henrik Ek

Isomap Examples[1]



demIsomap

[1] Data generation Carl Henrik Ek

Isomap: Summary

- MDS on shortest path approximation of manifold distance

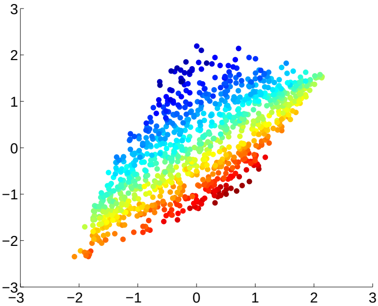
+ Simple
+ Intrinsic dimension from eigen spectra

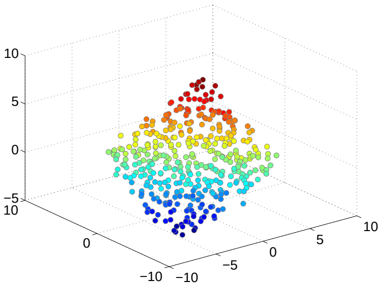
- Solves a very large eigenvalue problem
- Cannot handle holes or non-convex manifold
- Sensitive to “short circuit”

Inverse Covariance

- From the "covariance interpretation" we think of the similarity matrix as a covariance.
- Each element of the covariance is a function of two data points.
- Another option is to specify the inverse covariance.
  If the inverse covariance between two points is zero. Those points are independent given all other points.
  - This is a *conditional independence*.
  - Describes how points are connected.
- Laplacian Eigenmaps and LLE can both be seen as specifiying the inverse covariance.

LLE Examples[2]

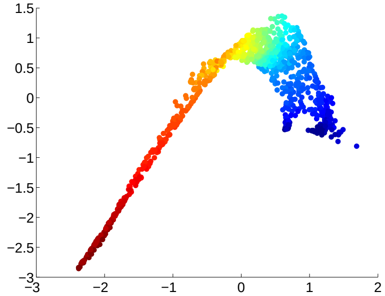


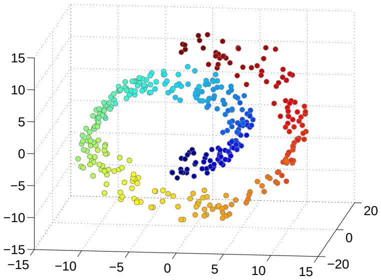


demLle

[2] 7 neighbours used. No playing with settings.

LLE Examples[2]

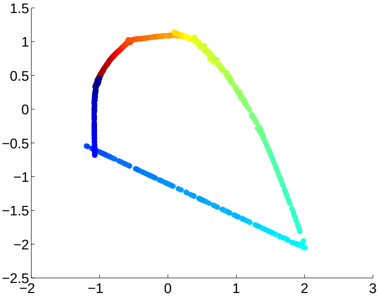


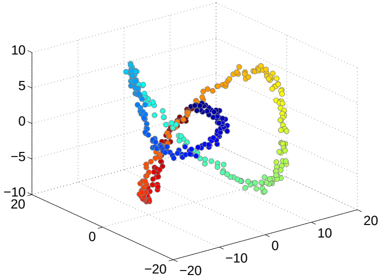


demLle

[2] 7 neighbours used. No playing with settings.

LLE Examples[2]





demLle

[2] 7 neighbours used. No playing with settings.

Generative

- Observed data have been sampled from manifold
- Spectral methods start in the "wrong" end
- *"It's a lot easier to make a mess than to clean it up!"*
  - Things break or disapear
- How to model observation "generation"?

Generative

- Observed data have been sampled from manifold
- Spectral methods start in the "wrong" end
- *"It's a lot easier to make a mess than to clean it up!"*
  - Things break or disapear
- How to model observation "generation"?

Generative

- Observed data have been sampled from manifold
- Spectral methods start in the “wrong” end
- *“It’s a lot easier to make a mess than to clean it up!”*
  - Things break or disapear
- How to model observation “generation”?

Generative

- Observed data have been sampled from manifold
- Spectral methods start in the “wrong” end
- *“It’s a lot easier to make a mess than to clean it up!”*
  - Things break or disapear
- How to model observation “generation”?

Outline

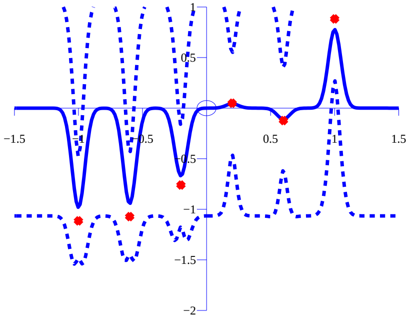
Model Selection

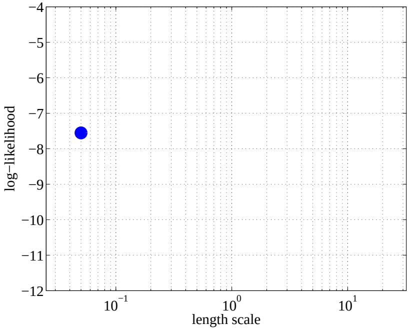
- Observed data have been sampled from low dimensional manifold
- $\mathbf{y} = f(\mathbf{x})$
- Idea: Model f rank embedding according to
  1. Data fit of f
  2. Complexity of f
- How to model f?
  1. Making as few assumtpions about f as possible?
  2. Allowing f from as "rich" class as possible?

Gaussian Processes

- Generalisation of Gaussian Distribution over **infinite** index sets
- Can be used specify distributions over functions
- Regression

$$
\begin{aligned}
\mathbf{y} &= f(\mathbf{x}) + \epsilon \\
p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi}) &= \int p(\mathbf{Y}|f, \mathbf{X}, \mathbf{\Phi}) p(f|\mathbf{X}, \mathbf{\Phi}) df \\
p(f|\mathbf{X}, \mathbf{\Phi}) &= \mathcal{N}(\mathbf{0}, \mathbf{K}) \\
\hat{\mathbf{\Phi}} &= \mathrm{argmax}_{\mathbf{\Phi}} p(\mathbf{Y}|\mathbf{X}, \mathbf{\Phi})
\end{aligned}
$$

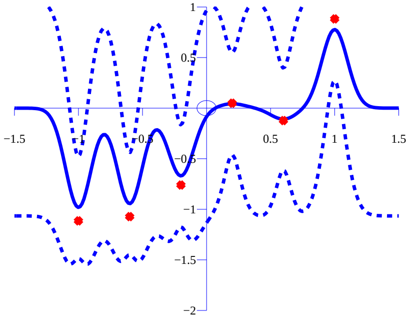


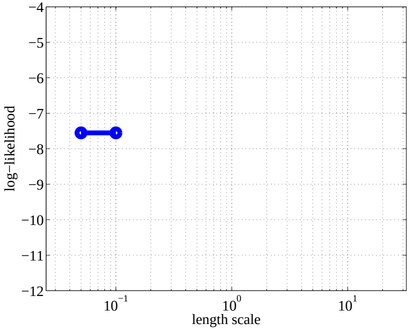


$$\log\, p(\mathbf{Y}|\mathbf{X}) \;=\; \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\,\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log\,2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

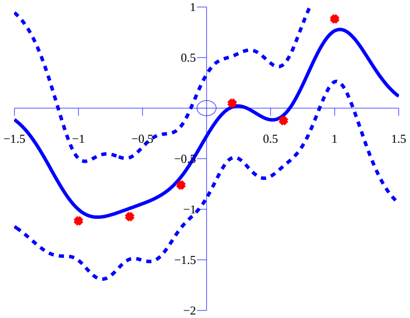


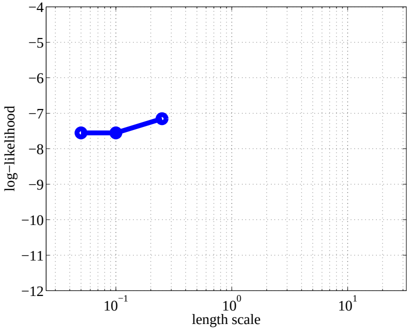


$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

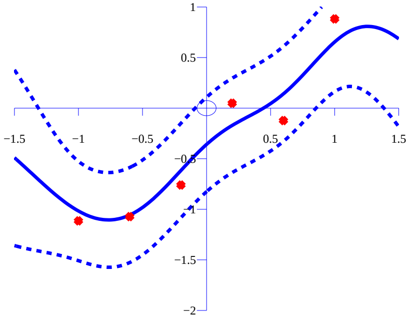


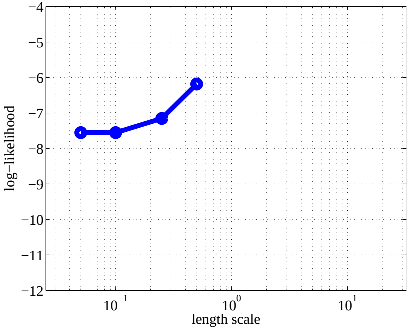


$$\log\, p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\,\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log\, 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

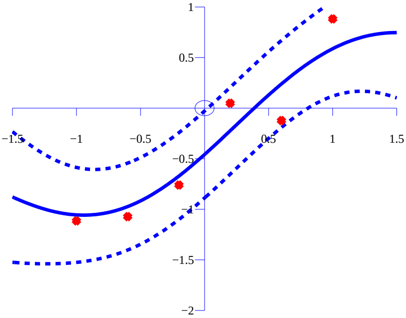


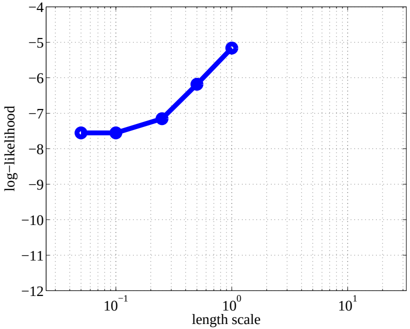


$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

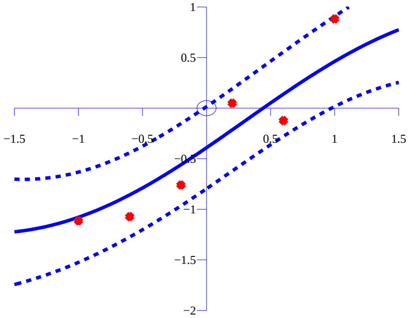


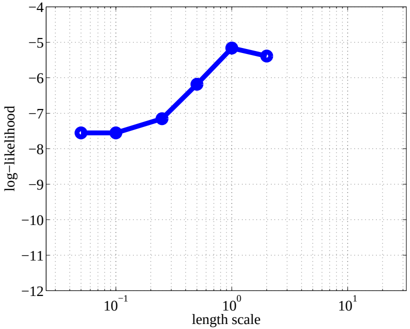


$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

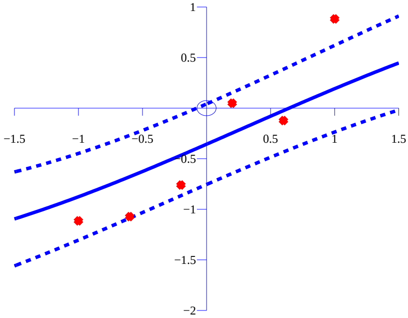


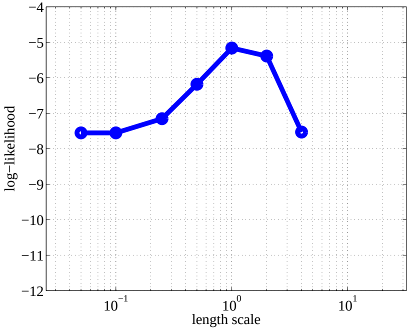


$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Processes[3]

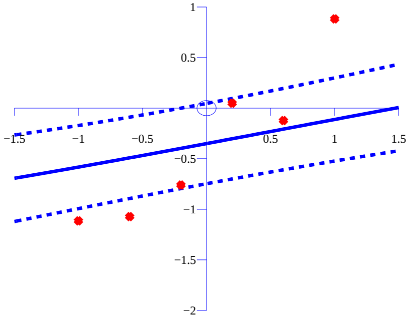


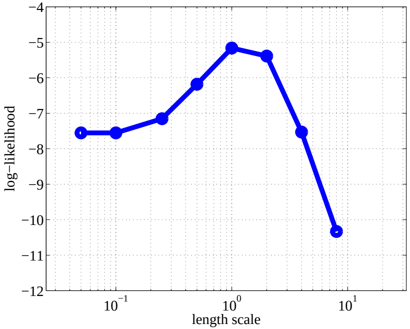


$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

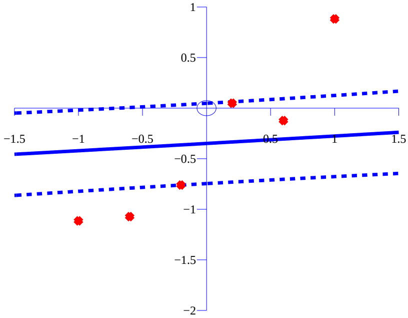
Gaussian Processes[3]

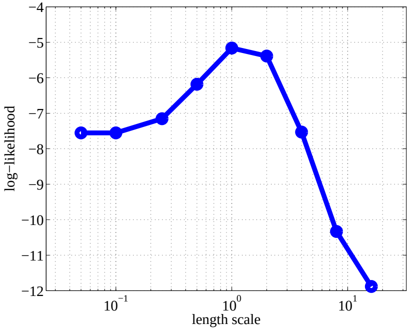




$$\log\ p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{data-fit} - \underbrace{\frac{1}{2}\log\ \det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{complexity} - \frac{N}{2}\log\ 2\pi$$

[3] Images: N.D. Lawrence





$$\log p(\mathbf{Y}|\mathbf{X}) = \underbrace{-\frac{1}{2}\mathbf{Y}^{\mathrm{T}}(\mathbf{K}+\beta^{-1}\mathbf{I})^{-1}\mathbf{Y}}_{\textit{data-fit}} - \underbrace{\frac{1}{2}\log\det(\mathbf{K}+\beta^{-1}\mathbf{I})}_{\textit{complexity}} - \frac{N}{2}\log 2\pi$$

[3] Images: N.D. Lawrence

Gaussian Process Latent Variable Models

- GP-LVM models sampling process

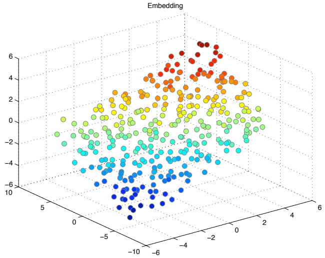
$$
\begin{aligned}
\mathbf{y} &= f(\mathbf{x}) + \epsilon \\
p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\Phi}) &= \int p(\mathbf{Y}|f, \mathbf{X}, \boldsymbol{\Phi}) p(f|\mathbf{X}, \boldsymbol{\Phi}) df \\
p(f|\mathbf{X}, \boldsymbol{\Phi}) &= \mathcal{N}(\mathbf{0}, \mathbf{K}) \\
\left\{\hat{\mathbf{X}}, \hat{\boldsymbol{\Phi}}\right\} &= \mathrm{argmax}_{\mathbf{X}, \boldsymbol{\Phi}} p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\Phi})
\end{aligned}
$$

- Linear: Closed form solution
- Non-Linear: Gradient based solution

Model Selection

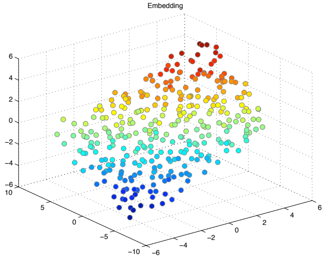
- *Lawrence* - 2003 suggested the use of Spectral algorithms to initialise the latent space **Y**
- *Harmeling* - 2007 evaluated the use of GP-LVM objective for model selection
  - Comparisons between **Procrustes** score to ground truth and GP-LVM objective

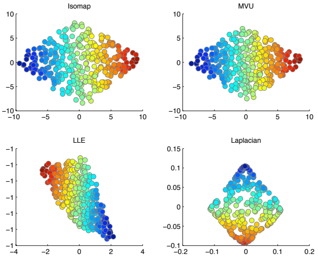
Model Selection: Results[4]



[4] Model selection results kindly provided by Carl Henrik Ek.

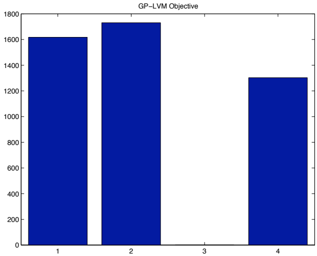
Model Selection: Results[4]

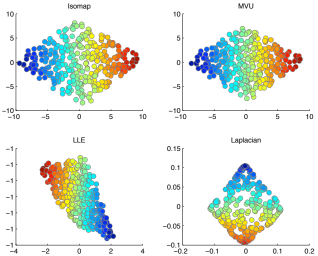




[4]Model selection results kindly provided by Carl Henrik Ek.

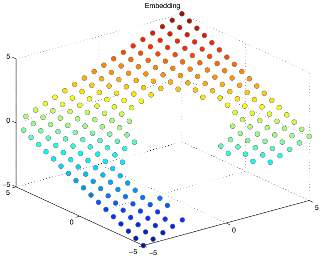
Model Selection: Results[4]





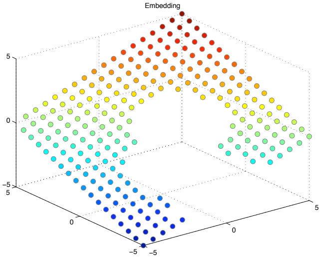
[4]Model selection results kindly provided by Carl Henrik Ek.

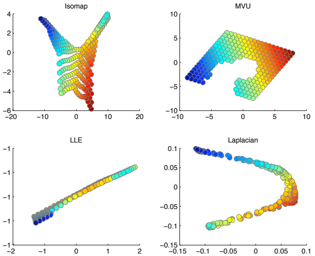
Model Selection: Results[4]



[4] Model selection results kindly provided by Carl Henrik Ek.

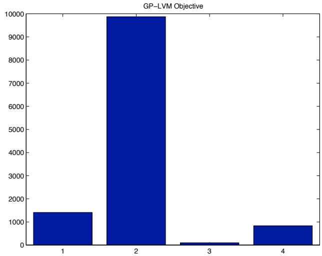
Model Selection: Results[4]

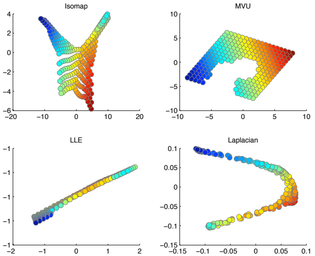




[4] Model selection results kindly provided by Carl Henrik Ek.

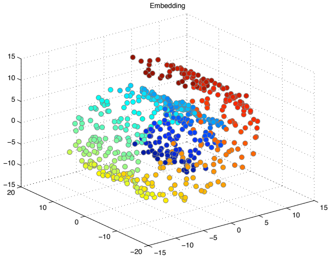
Model Selection: Results[4]



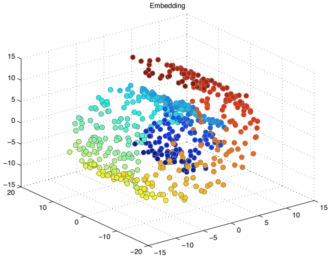


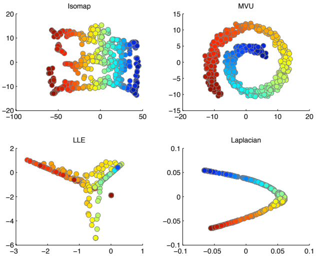
[4]Model selection results kindly provided by Carl Henrik Ek.

Model Selection: Results[4]

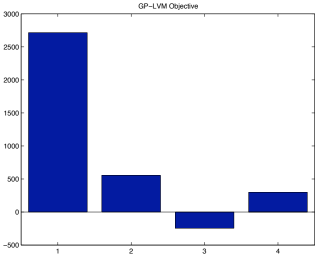


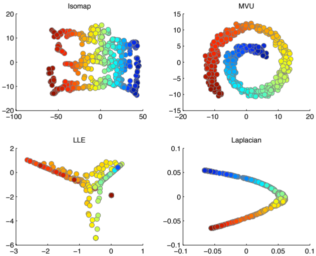
[4]Model selection results kindly provided by Carl Henrik Ek.





[4] Model selection results kindly provided by Carl Henrik Ek.





[4] Model selection results kindly provided by Carl Henrik Ek.

Conclusion

- Assume "local" structure contains enough "characteristics" to unravel global structure
- \+ Intuative
- \- Hard to set parameters without knowing manifold
- \- Learns embeddings not mappings *i.e. Visualisations*
- \- Models problem "wrong" way around
- \- Sensitive to noise
- \+ Currently best strategy to initialise generative models

References I

K. V. Mardia. *Statistics of Directional Data*. Academic Press, London, 1972.

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

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

J. B. Tenenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319.

Material

- Acknowledgement: Carl Henrik Ek for GP log likelihood examples.
- My examples given here
  http://www.cs.man.ac.uk/~neill/dimred/
- This talk
  http://www.cs.man.ac.uk/~neill/

Outline

- Distance Matching

Centering Matrix

If $\hat{\mathbf{Y}}$ is a version of $\mathbf{Y}$ with the mean removed then:

$$\hat{\mathbf{Y}} = \mathbf{H}\mathbf{Y}$$

$$\begin{aligned}
\hat{\mathbf{Y}} &= \left(\mathbf{I} - N^{-1}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\mathbf{Y} \\
&= \mathbf{Y} - \mathbf{1}\left(N^{-1}\mathbf{1}^{\mathrm{T}}\mathbf{Y}\right) \\
&= \mathbf{Y} - \mathbf{1}\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_{i,:}\right)^{\mathrm{T}} \\
&= \mathbf{Y} - \begin{bmatrix} \bar{\mathbf{y}}_{\cdot,:} \\ \bar{\mathbf{y}}_{\cdot,:} \\ \vdots \\ \bar{\mathbf{y}}_{\cdot,:} \end{bmatrix}
\end{aligned}$$

Return

Feature Selection Derivation

- Squared distance can be re-expressed as

$$d_{ij}^2 = \sum_{k=1}^{D} (y_{i,k} - y_{j,k})^2 .$$

- Can re-order the columns of $\mathbf{Y}$ without affecting the distances.
  - Choose ordering: first q columns of $\mathbf{Y}$ are the those that will best represent the distance matrix.
  - Substitution $\mathbf{x}_{:,k} = \mathbf{y}_{:,k}$ for $k = 1 \ldots q$.
- Distance in latent space is given by:

$$\delta_{ij}^2 = \sum_{k=1}^{q} (x_{i,k} - x_{j,k})^2 = \sum_{k=1}^{q} (y_{i,k} - y_{j,k})^2$$

Feature Selection Derivation II

- Can rewrite

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left|d_{ij}^2 - \delta_{ij}^2\right|.$$

as

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=q+1}^{D}\left(y_{i,k} - y_{j,k}\right)^2.$$

- Introduce mean of each dimension, $\bar{y}_k = \frac{1}{N}\sum_{i=1}^{N} y_{i,k}$,

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=q+1}^{D}\left(\left(y_{i,k} - \bar{y}_k\right) - \left(y_{j,k} - \bar{y}_k\right)\right)^2$$

- Expand brackets

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=q+1}^{D}\left(y_{i,k} - \bar{y}_k\right)^2 + \left(y_{j,k} - \bar{y}_k\right)^2 - 2\left(y_{j,k} - \bar{y}_k\right)\left(y_{i,k} - \bar{y}_k\right)$$

Feature Selection Derivation III

- Expand brackets

$$E(\mathbf{X}) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=q+1}^{D}\left(y_{i,k}-\bar{y}_k\right)^2 + \left(y_{j,k}-\bar{y}_k\right)^2 - 2\left(y_{j,k}-\bar{y}_k\right)\left(y_{i,k}-\bar{y}_k\right)$$

  Bring sums in

$$E(\mathbf{X}) = \sum_{k=q+1}^{D}\left(N\sum_{i=1}^{N}\left(y_{i,k}-\bar{y}_k\right)^2 + N\sum_{j=1}^{N}\left(y_{j,k}-\bar{y}_k\right)^2 - 2\sum_{j=1}^{N}\left(y_{j,k}-\bar{y}_k\right)\sum_{i=1}^{N}\left(y_{i,k}-\bar{y}_k\right)\right)$$

- Recognise as the sum of the variances discarded columns of $\mathbf{Y}$,

$$E(\mathbf{X}) = 2N^2\sum_{k=q+1}^{D}\sigma_k^2.$$

- We should compose $\mathbf{X}$ by extracting the columns of $\mathbf{Y}$ which have the largest variance. Return Selection Return Rotation