

# **Latent variable methods for neural population analysis**

Maneesh Sahani

Professor of Theoretical Neuroscience and Machine Learning  
Gatsby Computational Neuroscience Unit  
University College London

July 2016

## Latent variable methods

Most neural codes are distributed

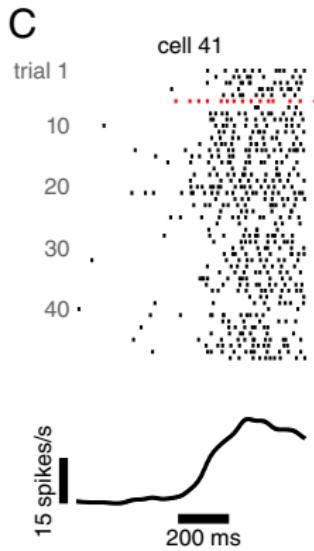
- ▶ Each neuron fires in response to a range of stimulus values (broad tuning) and may depend on more than one aspect of the stimulus (mixed selectivity).
- ▶ Firing of population must be taken together to identify stimulus.

Neurons are noisy

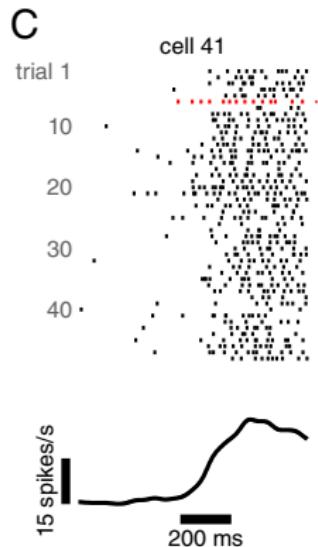
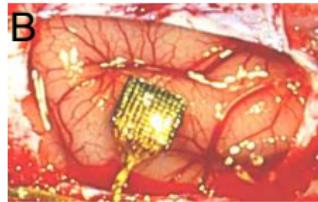
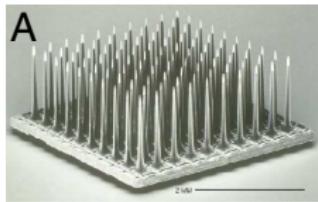
- ▶ Synaptic release failures.
- ▶ Branch-point spike propagation failures.
- ▶ Channel noise.
- ▶ Network chaos may amplify such noise.

Overall, network computation must be carried in the coordinated activity of many neurons.

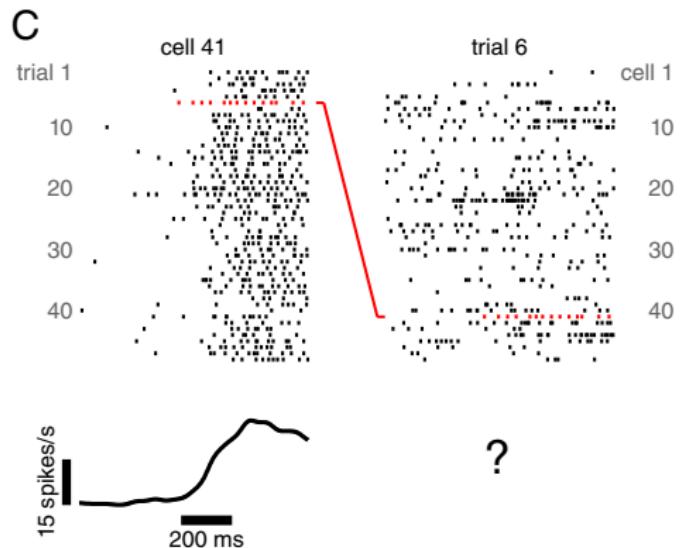
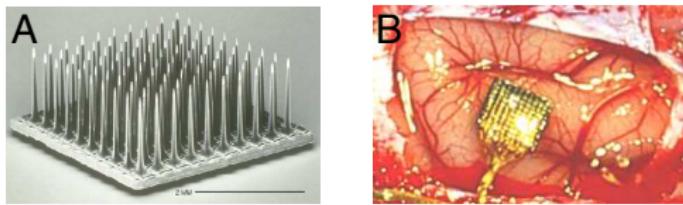
## Population recording



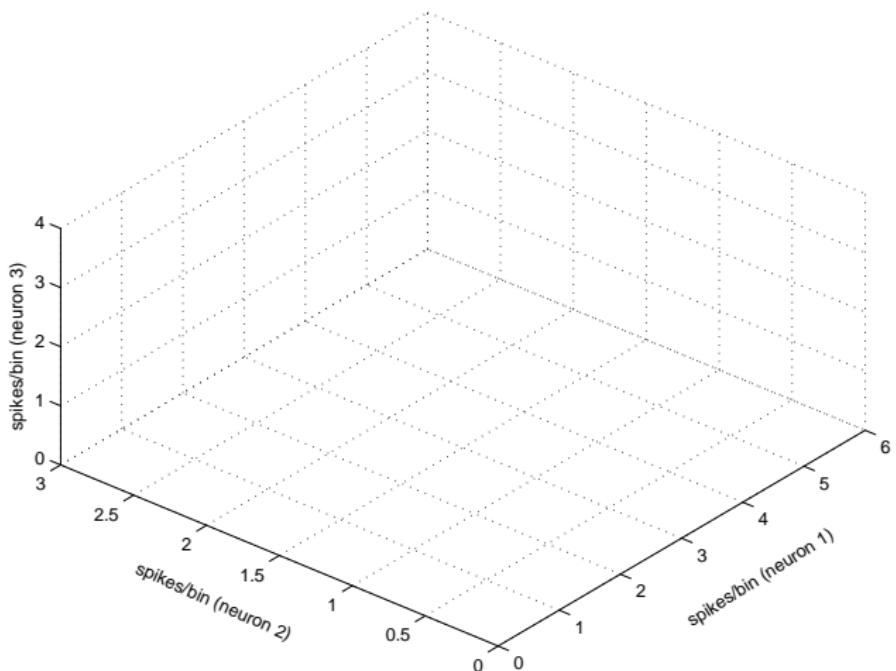
## Population recording



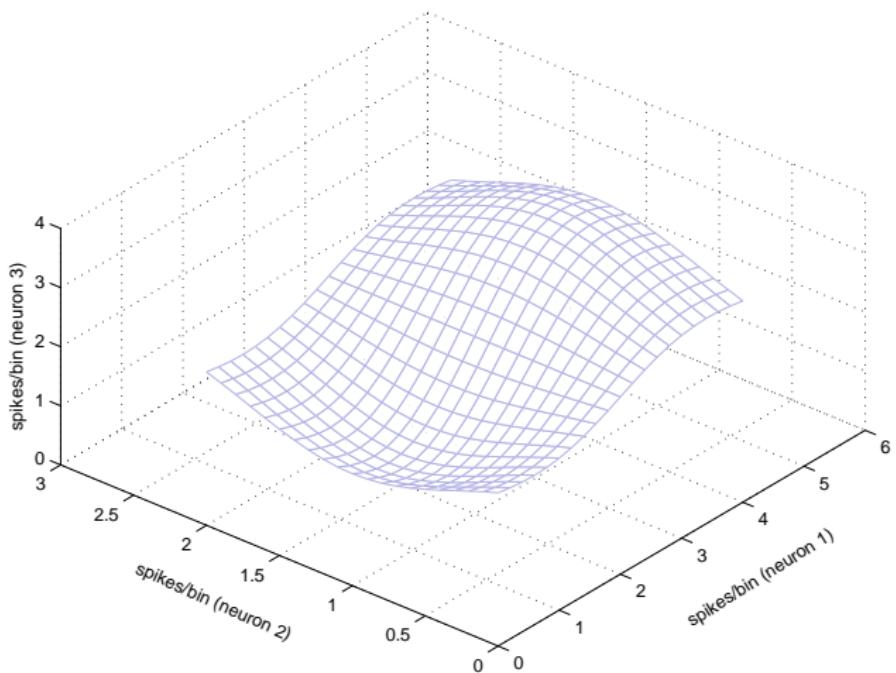
## Population recording



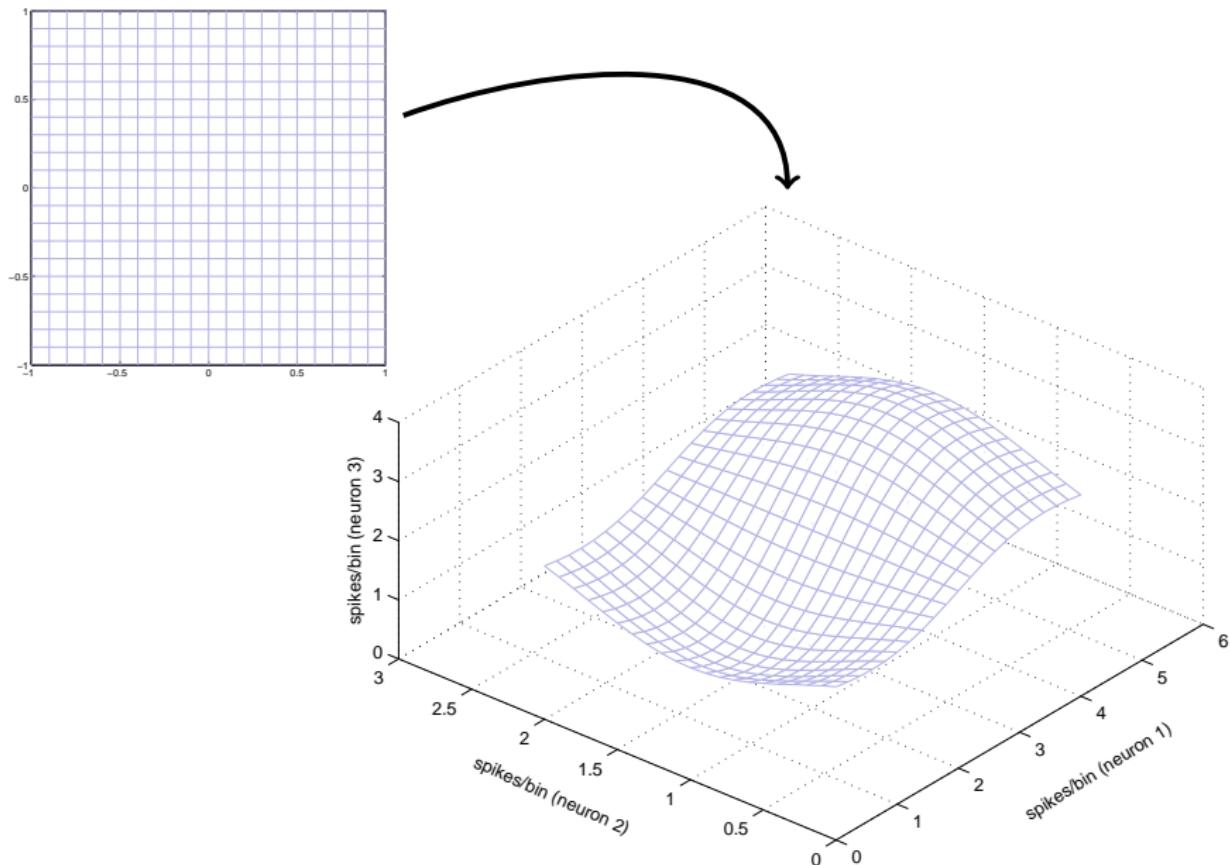
## Latent variable methods



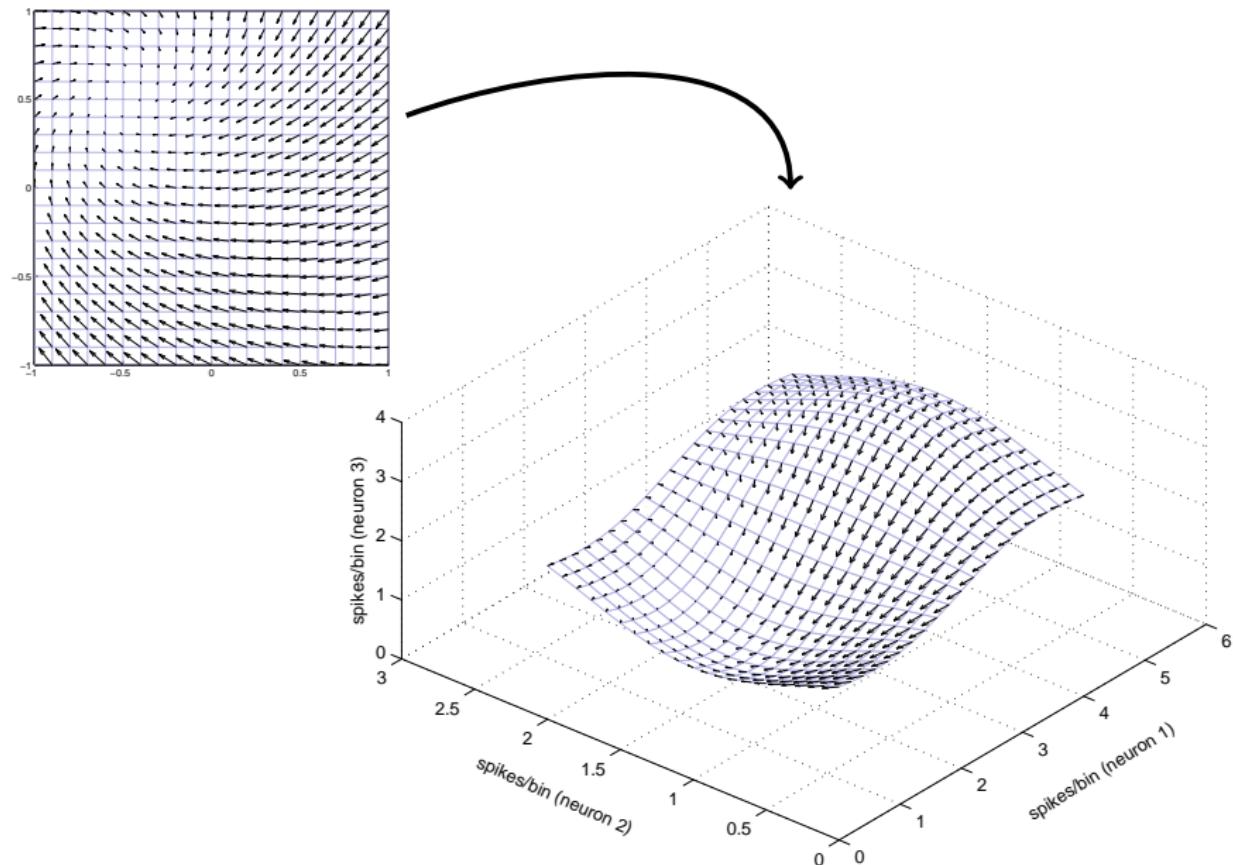
## Latent variable methods



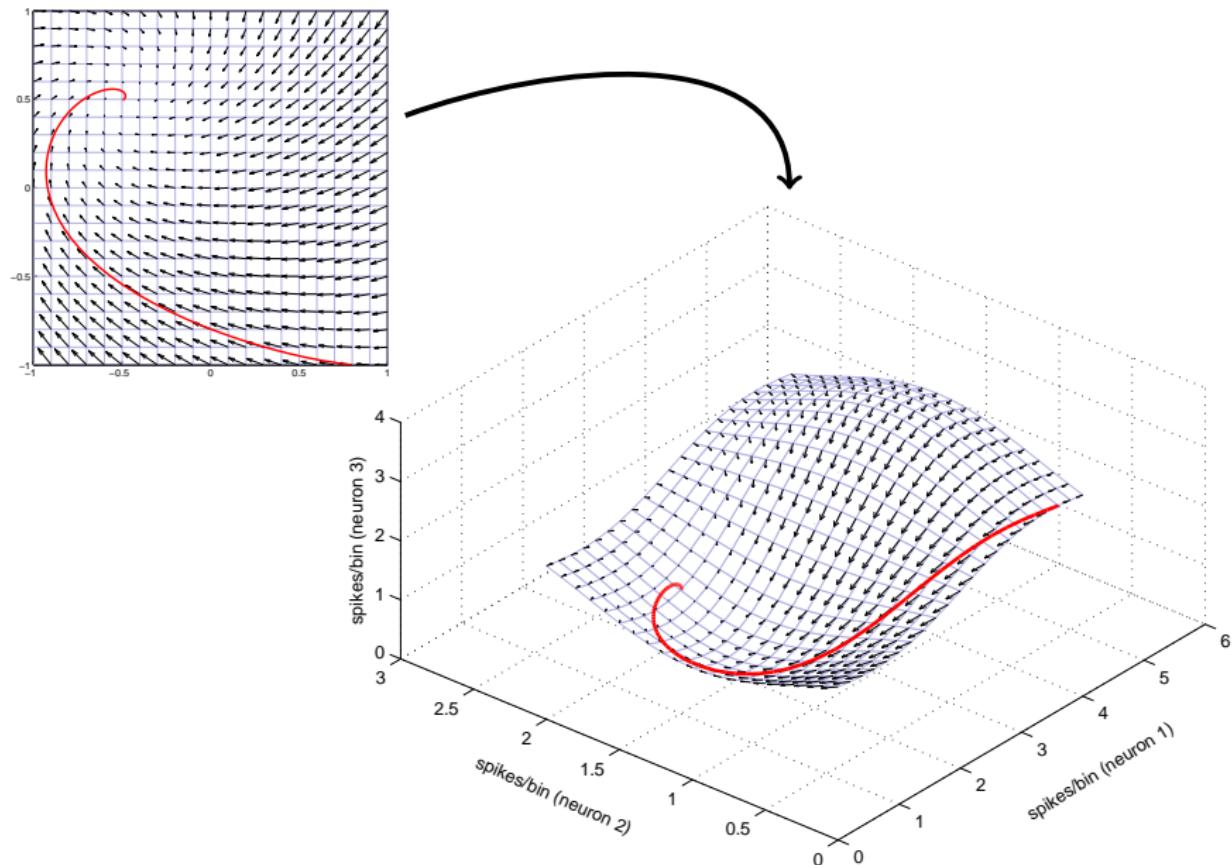
## Latent variable methods



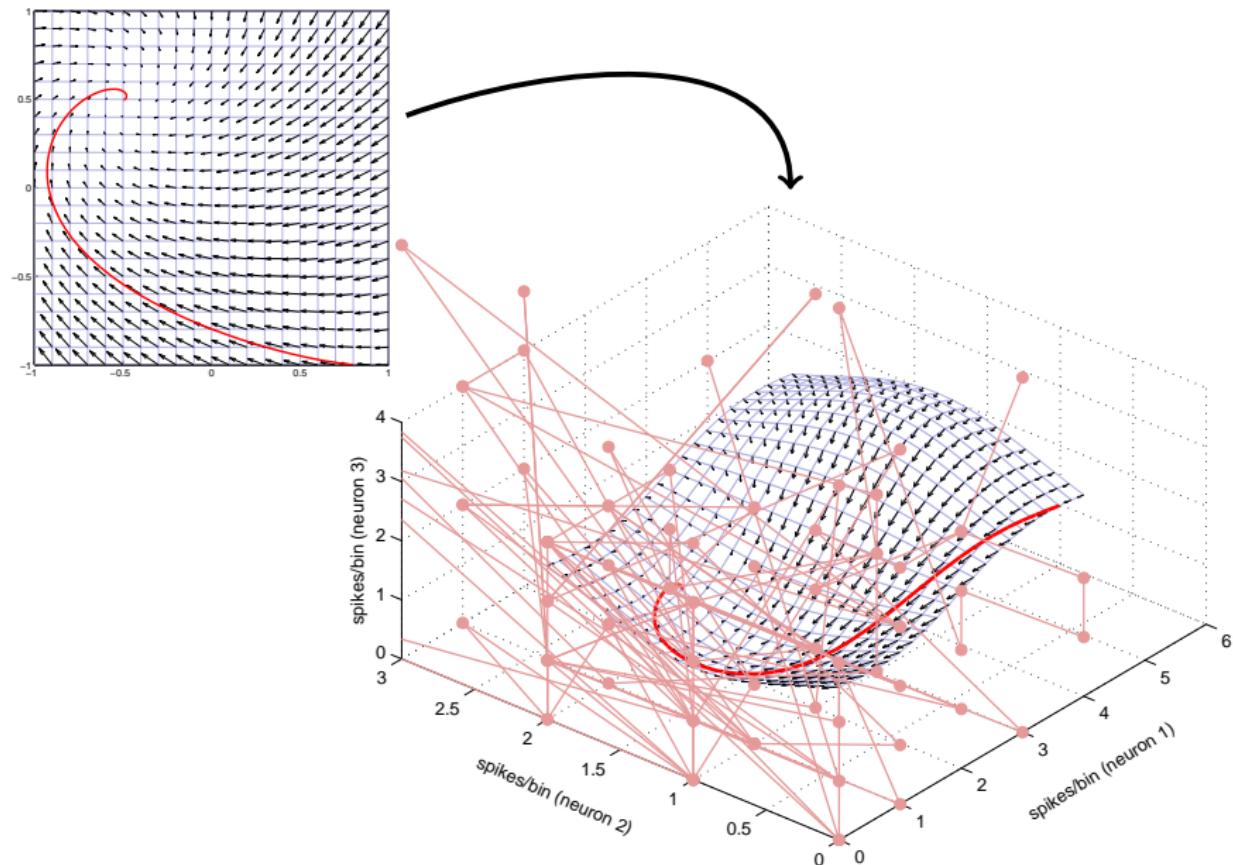
## Latent variable methods



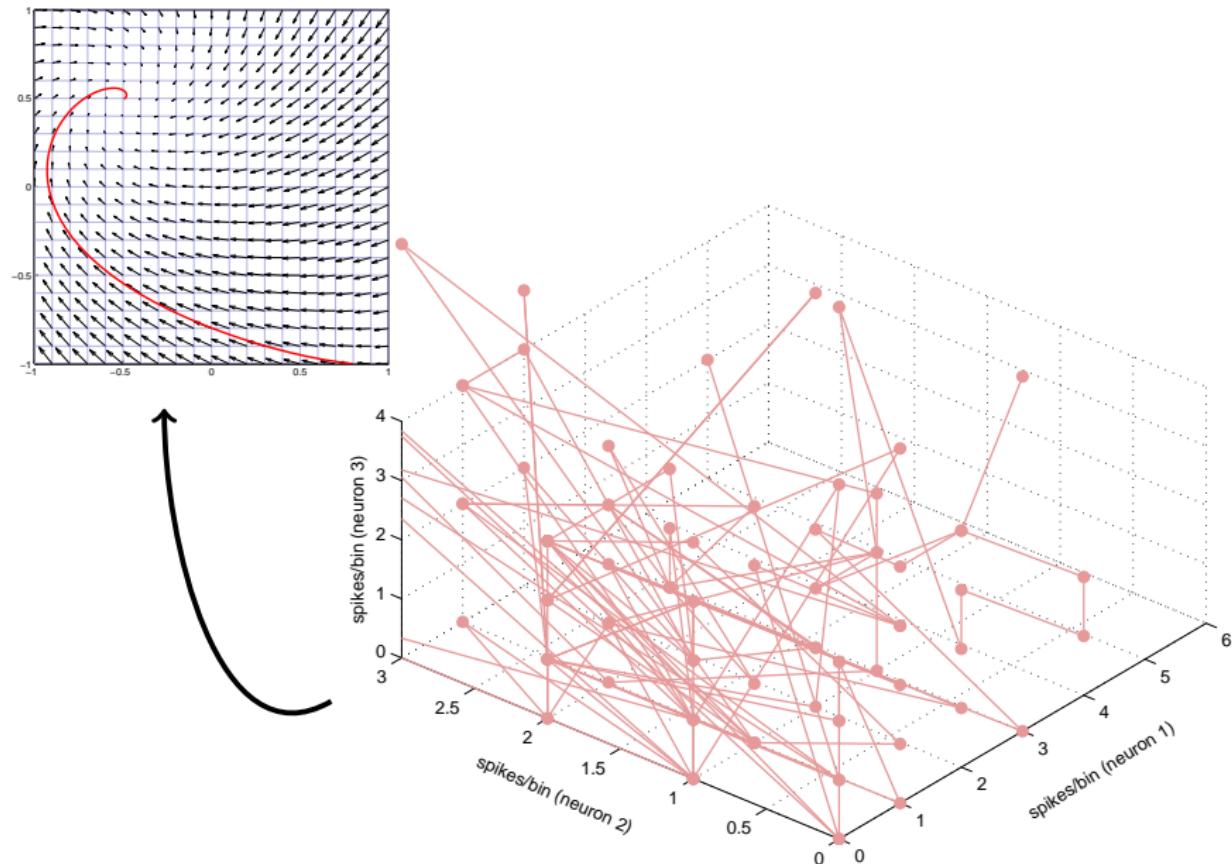
## Latent variable methods



## Latent variable methods



## Latent variable methods



## Two ideas

- ▶ Static dimensionality reduction
  - ▶ Requires data (population states) to be confined to low-dimensional manifold, with relatively small off-manifold noise.
  - ▶ In fact measured single-trial noise seems substantial, so single-trial analysis would require that the dominant modes of variability are not noise, but computational variability *within* the manifold.
  - ▶ Conversely, if computational variability is small, then trial-averaging (PSTHs) may reduce off-manifold variation and allow dimensionality reduction.
- ▶ Low-dimensional latent dynamics
  - ▶ Noise may lift data off manifold, but only “manifold projection” influences future evolution.
  - ▶ Conceptually familiar from population coding – independent (or otherwise non-code-shaped) noise is easy to average away.

## Notation

Data:  $\mathbf{x}_i = \begin{bmatrix} x_{1i} \\ x_{2i} \\ \vdots \\ x_{ni} \end{bmatrix} \in \mathbb{R}^n$  can be collected into a matrix:

$$X = \underbrace{\left[ \begin{array}{c|c|c|c} & \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ \hline & | & | & & | \\ \mathbf{x}_1 & | & | & & | \\ \hline & | & | & & | \end{array} \right]}_m}_{\text{m}} \Bigg\} n$$

We will assume the data are **centred**:  $\bar{\mathbf{x}} = \langle \mathbf{x} \rangle = \sum_{i=0}^m \mathbf{x}_i = \mathbf{0}$ .

## Two matrices of interest

The **data covariance** or **scatter** matrix:

$$S = \left\langle (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \right\rangle = \frac{1}{m} \sum_i \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{m} \mathbf{X} \mathbf{X}^T \in \mathbb{R}^{n \times n}.$$

## Two matrices of interest

The **data covariance** or **scatter** matrix:

$$S = \left\langle (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \right\rangle = \frac{1}{m} \sum_i \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{m} \mathbf{X} \mathbf{X}^T \in \mathbb{R}^{n \times n}.$$

The **inner product** or **Gram** matrix:

$$G = \begin{bmatrix} \mathbf{x}_1^T \mathbf{x}_1 & \mathbf{x}_1^T \mathbf{x}_2 & \cdots & \mathbf{x}_1^T \mathbf{x}_m \\ \mathbf{x}_2^T \mathbf{x}_1 & \mathbf{x}_2^T \mathbf{x}_2 & \cdots & \mathbf{x}_2^T \mathbf{x}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_m^T \mathbf{x}_1 & \mathbf{x}_m^T \mathbf{x}_2 & \cdots & \mathbf{x}_m^T \mathbf{x}_m \end{bmatrix} = \mathbf{X}^T \mathbf{X} \in \mathbb{R}^{m \times m}.$$

## Two matrices of interest

The **data covariance** or **scatter** matrix:

$$S = \left\langle (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \right\rangle = \frac{1}{m} \sum_i \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{m} \mathbf{X} \mathbf{X}^T \in \mathbb{R}^{n \times n}.$$

The **inner product** or **Gram** matrix:

$$G = \begin{bmatrix} \mathbf{x}_1^T \mathbf{x}_1 & \mathbf{x}_1^T \mathbf{x}_2 & \cdots & \mathbf{x}_1^T \mathbf{x}_m \\ \mathbf{x}_2^T \mathbf{x}_1 & \mathbf{x}_2^T \mathbf{x}_2 & \cdots & \mathbf{x}_2^T \mathbf{x}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_m^T \mathbf{x}_1 & \mathbf{x}_m^T \mathbf{x}_2 & \cdots & \mathbf{x}_m^T \mathbf{x}_m \end{bmatrix} = \mathbf{X}^T \mathbf{X} \in \mathbb{R}^{m \times m}.$$

Both are **real**, **symmetric** and **positive semi-definite**.

## Two matrices of interest

The **data covariance** or **scatter** matrix:

$$S = \left\langle (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \right\rangle = \frac{1}{m} \sum_i \mathbf{x}_i \mathbf{x}_i^T = \frac{1}{m} X X^T \in \mathbb{R}^{n \times n}.$$

The **inner product** or **Gram** matrix:

$$G = \begin{bmatrix} \mathbf{x}_1^T \mathbf{x}_1 & \mathbf{x}_1^T \mathbf{x}_2 & \cdots & \mathbf{x}_1^T \mathbf{x}_m \\ \mathbf{x}_2^T \mathbf{x}_1 & \mathbf{x}_2^T \mathbf{x}_2 & \cdots & \mathbf{x}_2^T \mathbf{x}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_m^T \mathbf{x}_1 & \mathbf{x}_m^T \mathbf{x}_2 & \cdots & \mathbf{x}_m^T \mathbf{x}_m \end{bmatrix} = X^T X \in \mathbb{R}^{m \times m}.$$

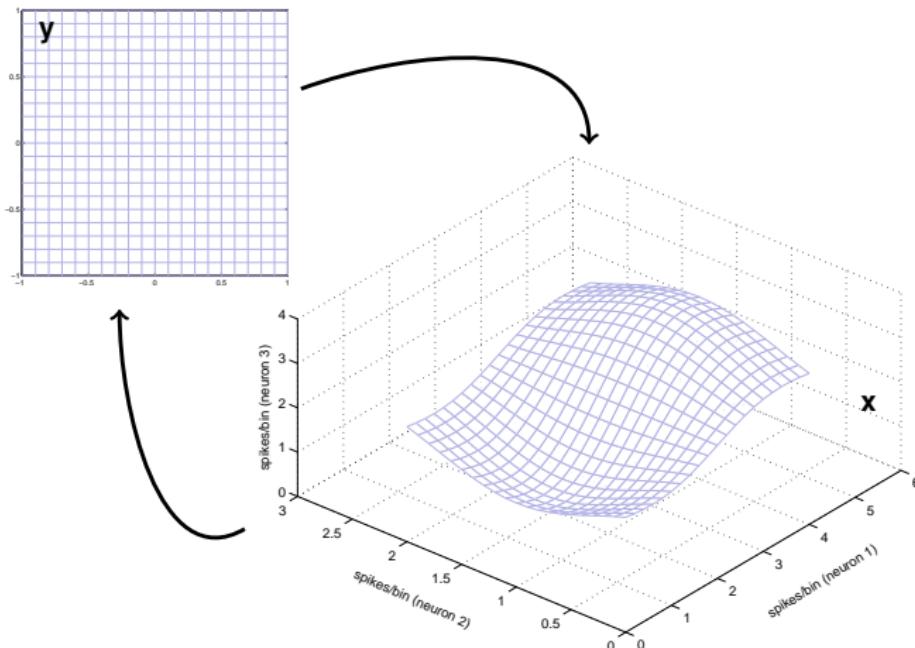
Both are **real, symmetric** and **positive semi-definite**.

$S$  and  $G$  share eigenvalues upto scale  $m$ . If  $X = V \Sigma U^T$  is the SVD, then  $S = V \frac{\Sigma^2}{m} V^T$  and  $G = U \Sigma^2 U^T$  are eigendecompositions.

## Dimensionality reduction

**Goal:** Find the latent manifold.

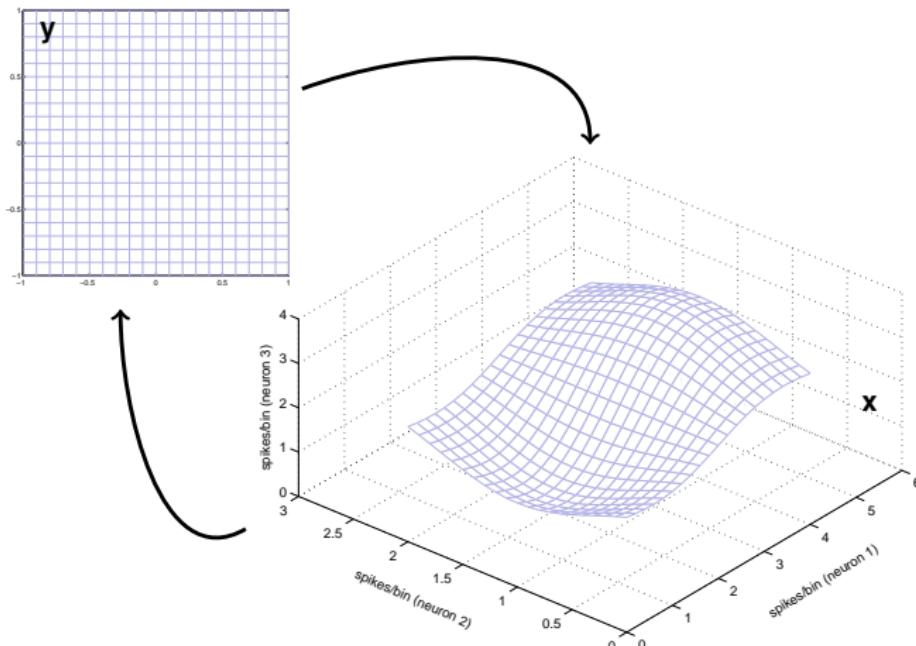
More precisely, find  $\mathbf{y}_i \in \mathbb{R}^k$ , ( $k < n$ ) so that  $\mathbf{y}_i$  parameterises the location of  $\mathbf{x}_i$  on the manifold.



## Dimensionality reduction

**Goal:** Find the latent manifold.

More precisely, find  $\mathbf{y}_i \in \mathbb{R}^k$ , ( $k < n$ ) so that  $\mathbf{y}_i$  parameterises the location of  $\mathbf{x}_i$  on the manifold.



### Core ideas:

- ▶ preserve “local” structure
- ▶ preserve “information”

## Principal Components Analysis (PCA) - a linear approach

Let  $\mathbf{y}_i = P^T \mathbf{x}_i$  for a **projection** matrix  $P^T$ .

## Principal Components Analysis (PCA) - a linear approach

Let  $\mathbf{y}_i = P^T \mathbf{x}_i$  for a **projection** matrix  $P^T$ .

$P \in \mathbb{R}^{n \times k}$  defines a **linear** mapping from data to manifold, and vice versa.

- ▶ preserves local structure
- ▶ preserves global structure

## Principal Components Analysis (PCA) - a linear approach

Let  $\mathbf{y}_i = P^T \mathbf{x}_i$  for a **projection** matrix  $P^T$ .

$P \in \mathbb{R}^{n \times k}$  defines a **linear** mapping from data to manifold, and vice versa.

- ▶ preserves local structure
- ▶ preserves global structure

**Idea:** look for projection that keeps data as spread out as possible  $\Rightarrow$  **most variance**.

- ▶ preserves “information”

## Principal Components Analysis (PCA) - a linear approach

Let  $\mathbf{y}_i = P^T \mathbf{x}_i$  for a **projection** matrix  $P^T$ .

$P \in \mathbb{R}^{n \times k}$  defines a **linear** mapping from data to manifold, and vice versa.

- ▶ preserves local structure
- ▶ preserves global structure

**Idea:** look for projection that keeps data as spread out as possible  $\Rightarrow$  **most variance**.

- ▶ preserves “information”

### PCA:

- ▶ Find direction of greatest variance –  $\rho_{(1)}$ .

$$\rho_{(1)} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmax}} \sum_i (\mathbf{x}_i^T \mathbf{u})^2$$

- ▶ Find direction orthogonal to  $\rho_{(1)}$  with greatest variance –  $\rho_{(2)}$

⋮

- ▶ Find direction orthogonal to  $\{\rho_{(1)}, \rho_{(2)}, \dots, \rho_{(j-1)}\}$  with greatest variance –  $\rho_{(j)}$ .

- ▶ Terminate when remaining variance drops below a threshold.

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that  $\mathbf{u}$  is an **eigenvector**, with scalar **eigenvalue**  $\omega$ , of a matrix  $S$  if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$  can have any norm, but we will define it to be unity (i.e.,  $\mathbf{u}^T\mathbf{u} = 1$ ).

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that  $\mathbf{u}$  is an **eigenvector**, with scalar **eigenvalue**  $\omega$ , of a matrix  $S$  if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$  can have any norm, but we will define it to be unity (i.e.,  $\mathbf{u}^T\mathbf{u} = 1$ ).

For a covariance matrix  $S = \langle \mathbf{x}\mathbf{x}^T \rangle$  (which is  $D \times D$ , symmetric, positive semi-definite):

- ▶ In general there are  $D$  eigenvector-eigenvalue pairs  $(\mathbf{u}_{(j)}, \omega_{(j)})$ , except if two or more eigenvectors share the same eigenvalue (in which case the eigenvectors are degenerate — any linear combination is also an eigenvector).

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that  $\mathbf{u}$  is an **eigenvector**, with scalar **eigenvalue**  $\omega$ , of a matrix  $S$  if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$  can have any norm, but we will define it to be unity (i.e.,  $\mathbf{u}^T\mathbf{u} = 1$ ).

For a covariance matrix  $S = \langle \mathbf{x}\mathbf{x}^T \rangle$  (which is  $D \times D$ , symmetric, positive semi-definite):

- ▶ In general there are  $D$  eigenvector-eigenvalue pairs  $(\mathbf{u}_{(i)}, \omega_{(i)})$ , except if two or more eigenvectors share the same eigenvalue (in which case the eigenvectors are degenerate — any linear combination is also an eigenvector).
- ▶ The  $D$  eigenvectors are orthogonal (or orthogonalisable, if  $\omega_{(i)} = \omega_{(j)}$ ). Thus, they form an **orthonormal basis**.  $\sum_i \mathbf{u}_{(i)}\mathbf{u}_{(i)}^T = I$ .

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that  $\mathbf{u}$  is an **eigenvector**, with scalar **eigenvalue**  $\omega$ , of a matrix  $S$  if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$  can have any norm, but we will define it to be unity (i.e.,  $\mathbf{u}^T\mathbf{u} = 1$ ).

For a covariance matrix  $S = \langle \mathbf{x}\mathbf{x}^T \rangle$  (which is  $D \times D$ , symmetric, positive semi-definite):

- ▶ In general there are  $D$  eigenvector-eigenvalue pairs  $(\mathbf{u}_{(i)}, \omega_{(i)})$ , except if two or more eigenvectors share the same eigenvalue (in which case the eigenvectors are degenerate — any linear combination is also an eigenvector).
- ▶ The  $D$  eigenvectors are orthogonal (or orthogonalisable, if  $\omega_{(i)} = \omega_{(j)}$ ). Thus, they form an **orthonormal basis**.  $\sum_i \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T = I$ .
- ▶ Any vector  $\mathbf{v}$  can be written as

$$\mathbf{v} = \left( \sum_i \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T \right) \mathbf{v} = \sum_i (\mathbf{u}_{(i)}^T \mathbf{v}) \mathbf{u}_{(i)} = \sum_i v_{(i)} \mathbf{u}_{(i)}$$

## Eigendecomposition of a covariance matrix

The eigendecomposition of a covariance matrix makes finding the PCs easy.

Recall that  $\mathbf{u}$  is an **eigenvector**, with scalar **eigenvalue**  $\omega$ , of a matrix  $S$  if

$$S\mathbf{u} = \omega\mathbf{u}$$

$\mathbf{u}$  can have any norm, but we will define it to be unity (i.e.,  $\mathbf{u}^T\mathbf{u} = 1$ ).

For a covariance matrix  $S = \langle \mathbf{x}\mathbf{x}^T \rangle$  (which is  $D \times D$ , symmetric, positive semi-definite):

- ▶ In general there are  $D$  eigenvector-eigenvalue pairs  $(\mathbf{u}_{(i)}, \omega_{(i)})$ , except if two or more eigenvectors share the same eigenvalue (in which case the eigenvectors are degenerate — any linear combination is also an eigenvector).
- ▶ The  $D$  eigenvectors are orthogonal (or orthogonalisable, if  $\omega_{(i)} = \omega_{(j)}$ ). Thus, they form an **orthonormal basis**.  $\sum_i \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T = I$ .
- ▶ Any vector  $\mathbf{v}$  can be written as

$$\mathbf{v} = \left( \sum_i \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T \right) \mathbf{v} = \sum_i (\mathbf{u}_{(i)}^T \mathbf{v}) \mathbf{u}_{(i)} = \sum_i v_{(i)} \mathbf{u}_{(i)}$$

- ▶ The original matrix  $S$  can be written:

$$S = \sum_i \omega_{(i)} \mathbf{u}_{(i)} \mathbf{u}_{(i)}^T = UWU^T$$

where  $U = [\mathbf{u}_{(1)}, \mathbf{u}_{(2)}, \dots, \mathbf{u}_{(D)}]$  collects the eigenvectors and  $W = \text{diag}[(\omega_{(1)}, \omega_{(2)}, \dots, \omega_{(D)})]$ .

## PCA and eigenvectors

- The variance in direction  $\mathbf{u}_{(i)}$  is

$$\left\langle (\mathbf{x}^\top \mathbf{u}_{(i)})^2 \right\rangle = \left\langle \mathbf{u}_{(i)}^\top \mathbf{x} \mathbf{x}^\top \mathbf{u}_{(i)} \right\rangle = \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^\top \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

## PCA and eigenvectors

- The variance in direction  $\mathbf{u}_{(i)}$  is

$$\left\langle (\mathbf{x}^\top \mathbf{u}_{(i)})^2 \right\rangle = \left\langle \mathbf{u}_{(i)}^\top \mathbf{x} \mathbf{x}^\top \mathbf{u}_{(i)} \right\rangle = \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^\top \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

- The variance in an arbitrary direction  $\mathbf{v}$  is

$$\begin{aligned}\left\langle (\mathbf{x}^\top \mathbf{v})^2 \right\rangle &= \left\langle \left( \mathbf{x}^\top \left( \sum_i v_{(i)} \mathbf{u}_{(i)} \right) \right)^2 \right\rangle = \sum_{ij} v_{(i)} \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(j)} v_{(j)} \\ &= \sum_{ij} v_{(i)} \omega_{(j)} v_{(j)} \mathbf{u}_{(i)}^\top \mathbf{u}_{(j)} = \sum_i v_{(i)}^2 \omega_{(i)}\end{aligned}$$

## PCA and eigenvectors

- The variance in direction  $\mathbf{u}_{(i)}$  is

$$\left\langle (\mathbf{x}^\top \mathbf{u}_{(i)})^2 \right\rangle = \left\langle \mathbf{u}_{(i)}^\top \mathbf{x} \mathbf{x}^\top \mathbf{u}_{(i)} \right\rangle = \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^\top \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

- The variance in an arbitrary direction  $\mathbf{v}$  is

$$\begin{aligned}\left\langle (\mathbf{x}^\top \mathbf{v})^2 \right\rangle &= \left\langle \left( \mathbf{x}^\top \left( \sum_i v_{(i)} \mathbf{u}_{(i)} \right) \right)^2 \right\rangle = \sum_{ij} v_{(i)} \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(j)} v_{(j)} \\ &= \sum_{ij} v_{(i)} \omega_{(j)} v_{(j)} \mathbf{u}_{(i)}^\top \mathbf{u}_{(j)} = \sum_i v_{(i)}^2 \omega_{(i)}\end{aligned}$$

- If  $\mathbf{v}^\top \mathbf{v} = 1$ , then  $\sum_i v_{(i)}^2 = 1$  and so  $\operatorname{argmax}_{\|\mathbf{v}\|=1} \left\langle (\mathbf{x}^\top \mathbf{v})^2 \right\rangle = \mathbf{u}_{(\max)}$   
The direction of greatest variance is the eigenvector the largest eigenvalue.

## PCA and eigenvectors

- The variance in direction  $\mathbf{u}_{(i)}$  is

$$\left\langle (\mathbf{x}^\top \mathbf{u}_{(i)})^2 \right\rangle = \left\langle \mathbf{u}_{(i)}^\top \mathbf{x} \mathbf{x}^\top \mathbf{u}_{(i)} \right\rangle = \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^\top \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

- The variance in an arbitrary direction  $\mathbf{v}$  is

$$\begin{aligned}\left\langle (\mathbf{x}^\top \mathbf{v})^2 \right\rangle &= \left\langle \left( \mathbf{x}^\top \left( \sum_i v_{(i)} \mathbf{u}_{(i)} \right) \right)^2 \right\rangle = \sum_{ij} v_{(i)} \mathbf{u}_{(i)}^\top \mathbf{S} \mathbf{u}_{(j)} v_{(j)} \\ &= \sum_{ij} v_{(i)} \omega_{(j)} v_{(j)} \mathbf{u}_{(i)}^\top \mathbf{u}_{(j)} = \sum_i v_{(i)}^2 \omega_{(i)}\end{aligned}$$

- If  $\mathbf{v}^\top \mathbf{v} = 1$ , then  $\sum_i v_{(i)}^2 = 1$  and so  $\operatorname{argmax}_{\|\mathbf{v}\|=1} \langle (\mathbf{x}^\top \mathbf{v})^2 \rangle = \mathbf{u}_{(\max)}$   
The direction of greatest variance is the eigenvector the largest eigenvalue.
- In general, the PCs are exactly the eigenvectors of the empirical covariance matrix, ordered by decreasing eigenvalue.

## PCA and eigenvectors

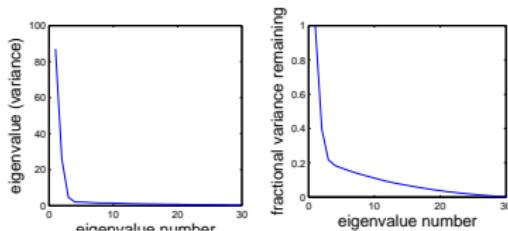
- The variance in direction  $\mathbf{u}_{(i)}$  is

$$\left\langle (\mathbf{x}^T \mathbf{u}_{(i)})^2 \right\rangle = \left\langle \mathbf{u}_{(i)}^T \mathbf{x} \mathbf{x}^T \mathbf{u}_{(i)} \right\rangle = \mathbf{u}_{(i)}^T \mathbf{S} \mathbf{u}_{(i)} = \mathbf{u}_{(i)}^T \omega_{(i)} \mathbf{u}_{(i)} = \omega_{(i)}$$

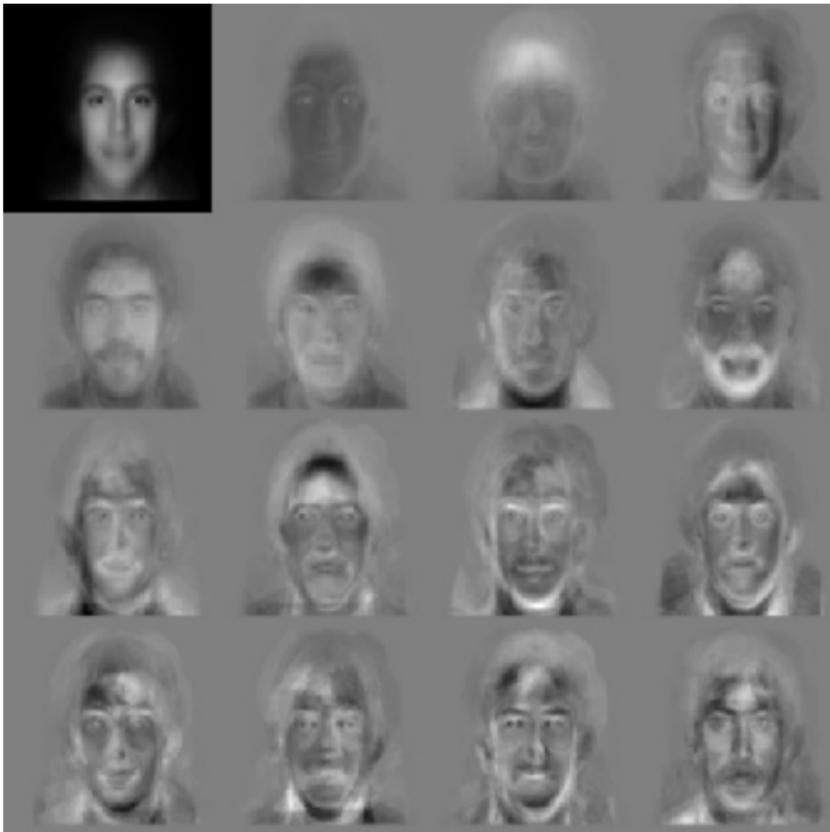
- The variance in an arbitrary direction  $\mathbf{v}$  is

$$\begin{aligned}\left\langle (\mathbf{x}^T \mathbf{v})^2 \right\rangle &= \left\langle \left( \mathbf{x}^T \left( \sum_i v_{(i)} \mathbf{u}_{(i)} \right) \right)^2 \right\rangle = \sum_{ij} v_{(i)} \mathbf{u}_{(i)}^T \mathbf{S} \mathbf{u}_{(j)} v_{(j)} \\ &= \sum_{ij} v_{(i)} \omega_{(j)} v_{(j)} \mathbf{u}_{(i)}^T \mathbf{u}_{(j)} = \sum_i v_{(i)}^2 \omega_{(i)}\end{aligned}$$

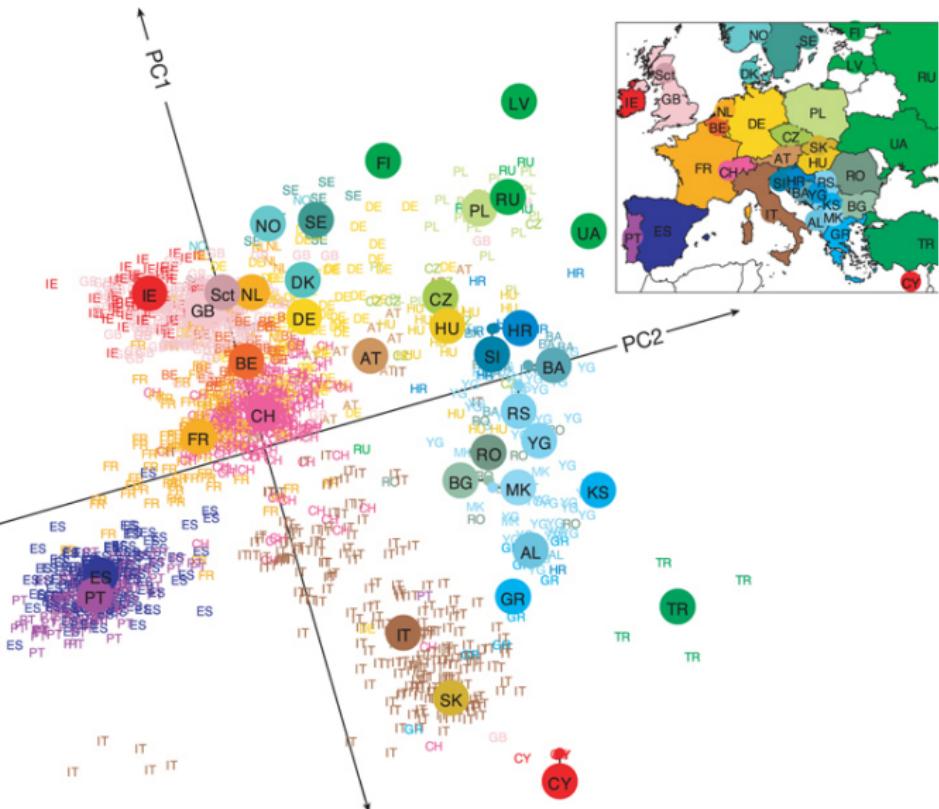
- If  $\mathbf{v}^T \mathbf{v} = 1$ , then  $\sum_i v_{(i)}^2 = 1$  and so  $\operatorname{argmax}_{\|\mathbf{v}\|=1} \left\langle (\mathbf{x}^T \mathbf{v})^2 \right\rangle = \mathbf{u}_{(\max)}$   
The direction of greatest variance is the eigenvector the largest eigenvalue.
- In general, the PCs are exactly the eigenvectors of the empirical covariance matrix, ordered by decreasing eigenvalue.
- The **eigenspectrum** shows how the variance is distributed across dimensions; can identify transitions that might separate signal from noise, or the number of PCs that capture a pre-determined fraction of variance.



## PCA and latent structure: Eigenfaces

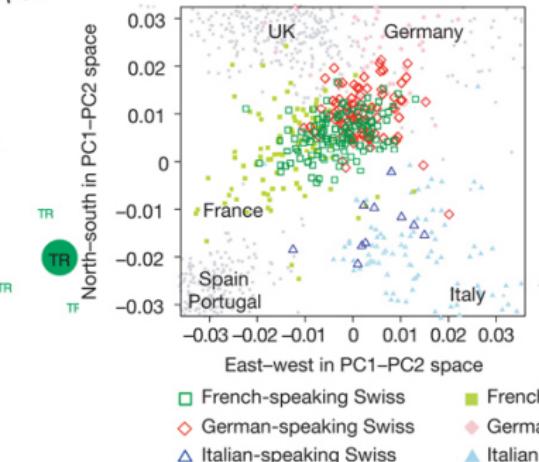
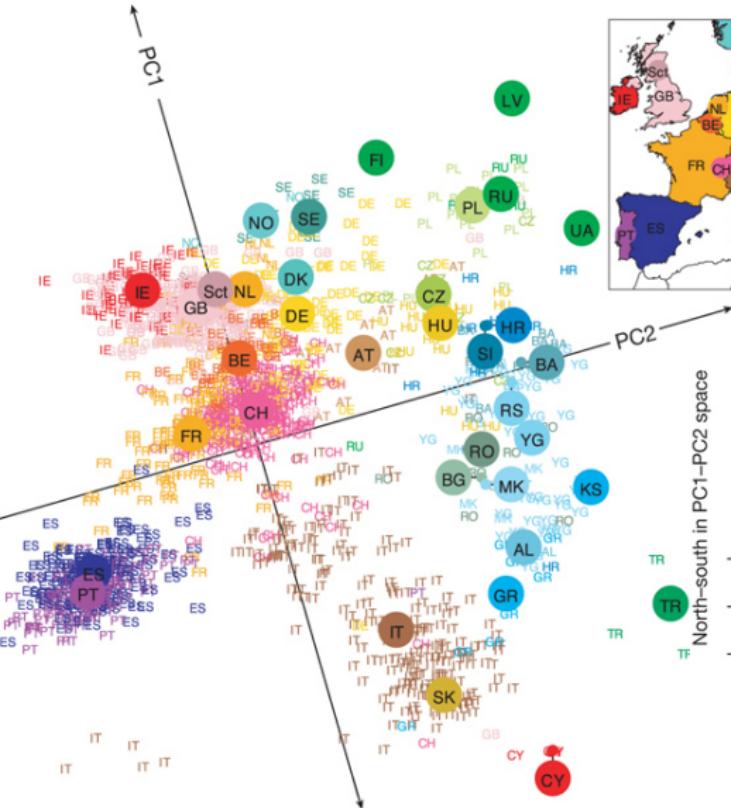


# PCA and latent structure: Genetic variation within Europe



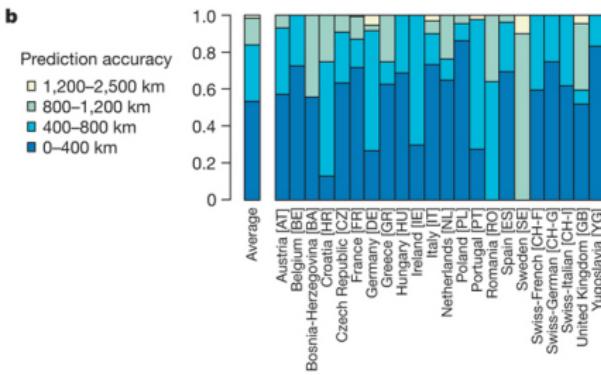
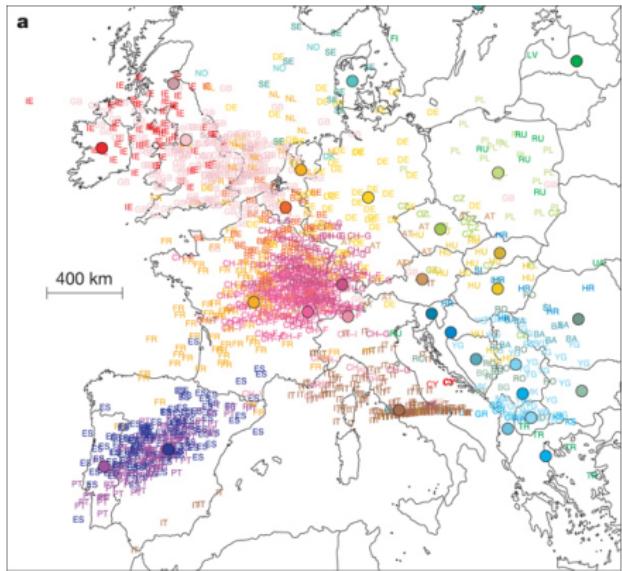
Novembre et al. (2008) Nature 456:98-101

# PCA and latent structure: Genetic variation within Europe



Novembre et al. (2008) Nature 456:98-101

# Genetic variation within Europe



Novembre et al. (2008) Nature 456:98-101

## Equivalent definitions of PCA

- ▶ Find  $K$  directions of greatest variance in data.
- ▶ Find  $K$ -dimensional orthogonal projection that *preserves* greatest variance.
- ▶ Find  $K$ -dimensional vectors  $\mathbf{y}_i$  and matrix  $\Lambda$  so that  $\hat{\mathbf{x}}_i = \Lambda \mathbf{y}_i$  is as close as possible (in squared distance) to  $\mathbf{x}_i$ .
- ▶ Find the approximate rank- $K$  factorisation of the data matrix  $X \approx \Lambda Y$  with smallest squared error.
- ▶ ... (many others)

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

So we want to maximise the entropy of  $\mathbf{y}$ .

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

So we want to maximise the entropy of  $\mathbf{y}$ . What is the entropy of a Gaussian?

$$H(\mathbf{z}) = - \int d\mathbf{z} p(\mathbf{z}) \ln p(\mathbf{z}) = \frac{1}{2} \ln |\Sigma| + \frac{D}{2}(1 + \ln 2\pi)$$

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

So we want to maximise the entropy of  $\mathbf{y}$ . What is the entropy of a Gaussian?

$$H(\mathbf{z}) = - \int d\mathbf{z} p(\mathbf{z}) \ln p(\mathbf{z}) = \frac{1}{2} \ln |\Sigma| + \frac{D}{2}(1 + \ln 2\pi)$$

Therefore we want the distribution of  $\mathbf{y}$  to have largest volume (i.e. det of covariance matrix).

$$\Sigma_y = A\Sigma_x A^T = A U W_x U^T A^T$$

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

So we want to maximise the entropy of  $\mathbf{y}$ . What is the entropy of a Gaussian?

$$H(\mathbf{z}) = - \int d\mathbf{z} p(\mathbf{z}) \ln p(\mathbf{z}) = \frac{1}{2} \ln |\Sigma| + \frac{D}{2}(1 + \ln 2\pi)$$

Therefore we want the distribution of  $\mathbf{y}$  to have largest volume (i.e. det of covariance matrix).

$$\Sigma_y = A\Sigma_x A^T = A U \Lambda_x U^T A^T$$

So,  $A$  should be aligned with the columns of  $U$  which are associated with the largest eigenvalues (variances).

## Another view of PCA: Mutual information

**Problem:** Given  $\mathbf{x}$ , find  $\mathbf{y} = A\mathbf{x}$  with columns of  $A$  unit vectors, s.t.  $I(\mathbf{y}; \mathbf{x})$  is maximised (assuming that  $P(\mathbf{x})$  is Gaussian).

$$I(\mathbf{y}; \mathbf{x}) = H(\mathbf{y}) + H(\mathbf{x}) - H(\mathbf{y}, \mathbf{x}) = H(\mathbf{y})$$

So we want to maximise the entropy of  $\mathbf{y}$ . What is the entropy of a Gaussian?

$$H(\mathbf{z}) = - \int d\mathbf{z} p(\mathbf{z}) \ln p(\mathbf{z}) = \frac{1}{2} \ln |\Sigma| + \frac{D}{2}(1 + \ln 2\pi)$$

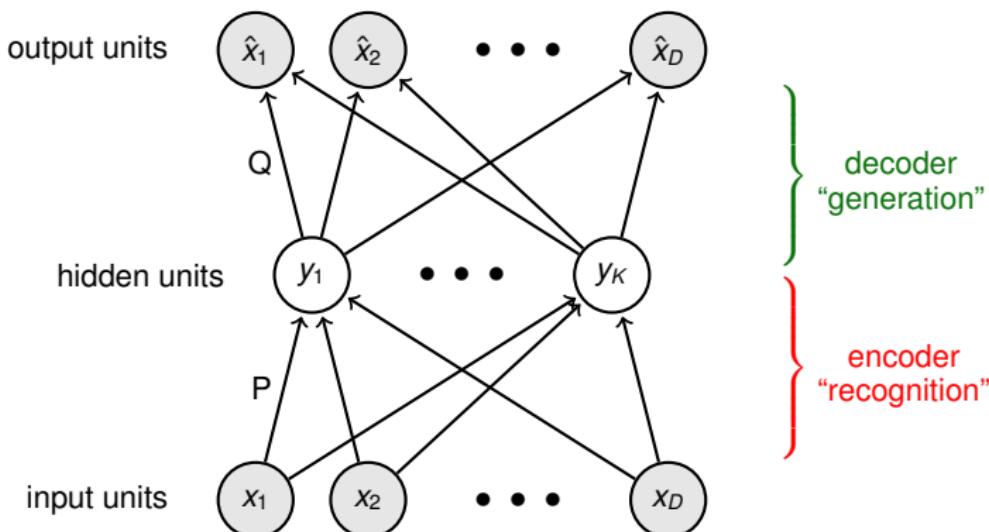
Therefore we want the distribution of  $\mathbf{y}$  to have largest volume (i.e. det of covariance matrix).

$$\Sigma_y = A\Sigma_x A^T = AUW_x U^T A^T$$

So,  $A$  should be aligned with the columns of  $U$  which are associated with the largest eigenvalues (variances).

Projection to the principal component subspace preserves the most information about the (Gaussian) data.

## Linear autoencoders: From supervised learning to PCA



$$\text{Learning: } \underset{P,Q}{\operatorname{argmin}} \|\hat{\mathbf{x}} - \mathbf{x}\|^2 \quad \hat{\mathbf{x}} = Q\mathbf{y} \quad \mathbf{y} = P\mathbf{x}$$

At the optimum,  $P$  and  $Q$  perform the projection and reconstruction steps of PCA. (Baldi & Hornik 1989).

## Noise models

PCA is probably the most common form of dimensionality reduction.

## Noise models

PCA is probably the most common form of dimensionality reduction.

But PCA (and most non-linear dimensionality reduction methods) do not formally take noise into account.

## Noise models

PCA is probably the most common form of dimensionality reduction.

But PCA (and most non-linear dimensionality reduction methods) do not formally take noise into account.

- ▶ In fact, PCA is close enough to “probabilistic PCA” (pPCA) that we can treat it as assuming **small** equal variance (**isotropic**) Gaussian noise.
- ▶ Measurements suggest that variability scales with mean firing rate  $\Rightarrow$  **large** (relative to manifold spread), **anisotropic** and **non-stationary**.

## Noise models

PCA is probably the most common form of dimensionality reduction.

But PCA (and most non-linear dimensionality reduction methods) do not formally take noise into account.

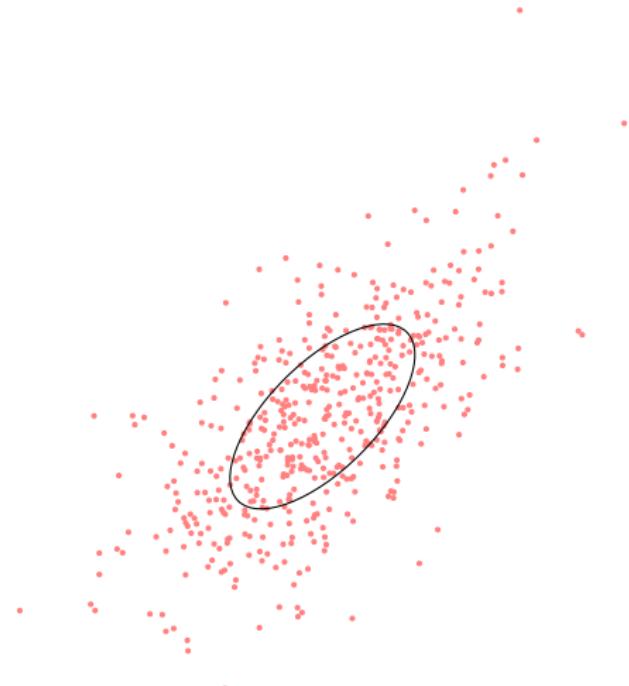
- ▶ In fact, PCA is close enough to “probabilistic PCA” (pPCA) that we can treat it as assuming **small** equal variance (**isotropic**) Gaussian noise.
- ▶ Measurements suggest that variability scales with mean firing rate  $\Rightarrow$  **large** (relative to manifold spread), **anisotropic** and **non-stationary**.

To handle substantial neuronal variability we need an explicit noise model:

- ▶ Factor analysis (Gaussian noise with different variance for each neuron).
- ▶ Poisson factor models (variance scales with predicted mean).

## Latent variables and Gaussians

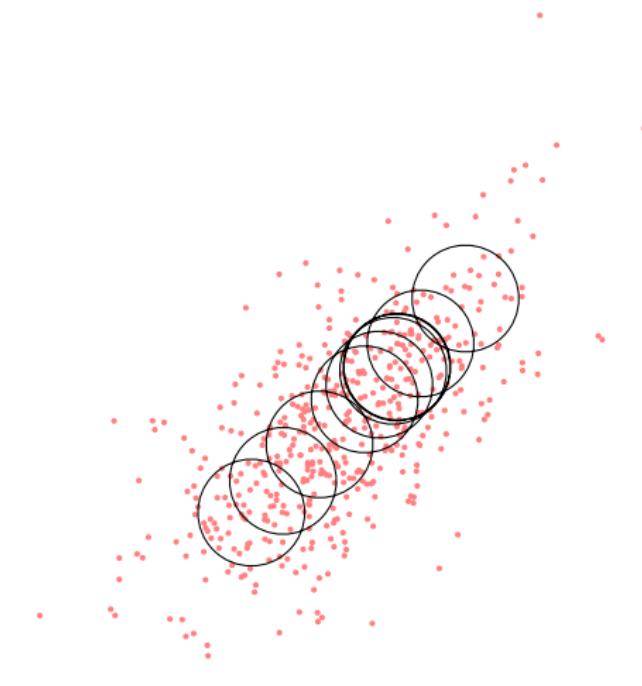
Gaussian correlation can be composed from latent components and uncorrelated noise.



$$\mathbf{x} \sim \mathcal{N} \left( \mathbf{0}, \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} \right)$$

## Latent variables and Gaussians

Gaussian correlation can be composed from latent components and uncorrelated noise.



$$\mathbf{x} \sim \mathcal{N} \left( \mathbf{0}, \begin{bmatrix} 3 & 2 \\ 2 & 3 \end{bmatrix} \right)$$

$\Leftrightarrow$

$$y \sim \mathcal{N}(0, 1)$$

$$\mathbf{x} \sim \mathcal{N} \left( \sqrt{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix} y, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)$$

## Probabilistic Principal Components Analysis (PPCA)

If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

## Probabilistic Principal Components Analysis (PPCA)

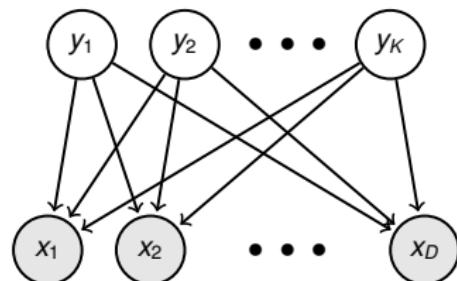
If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \psi)$  Gaussian noise
- ▶  $K < D$



## Probabilistic Principal Components Analysis (PPCA)

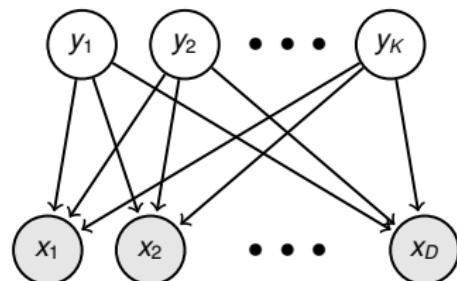
If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \psi)$  Gaussian noise
- ▶  $K < D$



Model for observations  $\mathbf{x}$  is a correlated Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \psi I)$$

where  $\Lambda$  is a  $D \times K$  matrix.

## Probabilistic Principal Components Analysis (PPCA)

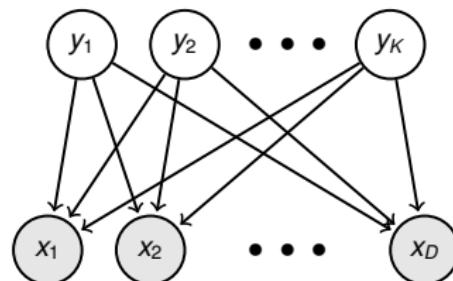
If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \psi)$  Gaussian noise
- ▶  $K < D$



Model for observations  $\mathbf{x}$  is a correlated Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \psi I)$$

$$p(\mathbf{x}) = \int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y}$$

where  $\Lambda$  is a  $D \times K$  matrix.

## Probabilistic Principal Components Analysis (PPCA)

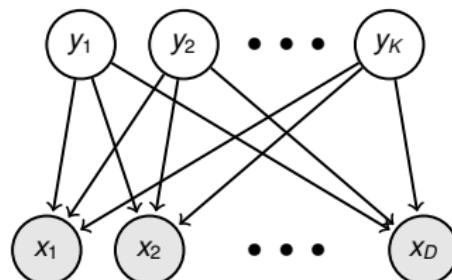
If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \psi)$  Gaussian noise
- ▶  $K < D$



Model for observations  $\mathbf{x}$  is a correlated Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$\text{Note: } \mathbb{E}_{\mathbf{x}} [f(x)] = \mathbb{E}_{\mathbf{y}} [\mathbb{E}_{\mathbf{x}|\mathbf{y}} [f(x)]]$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \psi I)$$

$$\mathbb{V}_{\mathbf{x}} [x] = \mathbb{E}_{\mathbf{y}} [\mathbb{V}[\mathbf{x}|\mathbf{y}]] + \mathbb{V}_{\mathbf{y}} [\mathbb{E}[\mathbf{x}|\mathbf{y}]]$$

$$p(\mathbf{x}) = \int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y} = \mathcal{N}\left(\mathbb{E}_{\mathbf{y}} [\Lambda\mathbf{y}], \mathbb{E}_{\mathbf{y}} [\Lambda\mathbf{y}\mathbf{y}^T\Lambda^T] + \psi I\right)$$

where  $\Lambda$  is a  $D \times K$  matrix.

## Probabilistic Principal Components Analysis (PPCA)

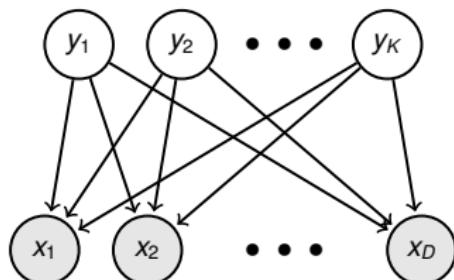
If the uncorrelated noise is assumed to be isotropic, this model is called PPCA.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \psi)$  Gaussian noise
- ▶  $K < D$



Model for observations  $\mathbf{x}$  is a correlated Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$\text{Note: } \mathbb{E}_{\mathbf{x}} [f(x)] = \mathbb{E}_{\mathbf{y}} [\mathbb{E}_{\mathbf{x}|\mathbf{y}} [f(x)]]$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \psi I)$$

$$\mathbb{V}_{\mathbf{x}} [x] = \mathbb{E}_{\mathbf{y}} [\mathbb{V}[\mathbf{x}|\mathbf{y}]] + \mathbb{V}_{\mathbf{y}} [\mathbb{E}[\mathbf{x}|\mathbf{y}]]$$

$$p(\mathbf{x}) = \int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y} = \mathcal{N}\left(\mathbb{E}_{\mathbf{y}} [\Lambda\mathbf{y}], \mathbb{E}_{\mathbf{y}} [\Lambda\mathbf{y}\mathbf{y}^T\Lambda^T] + \psi I\right) = \mathcal{N}\left(\mathbf{0}, \Lambda\Lambda^T + \psi I\right)$$

where  $\Lambda$  is a  $D \times K$  matrix.

## PPCA likelihood

The marginal distribution on  $\mathbf{x}$  gives us the PPCA likelihood:

$$\log p(\mathcal{X}|\Lambda, \psi) = -\frac{N}{2} \log |2\pi(\Lambda\Lambda^T + \psi I)| - \frac{1}{2} \text{Tr} \left[ (\Lambda\Lambda^T + \psi I)^{-1} \underbrace{\sum_n \mathbf{x}\mathbf{x}^T}_{NS} \right]$$

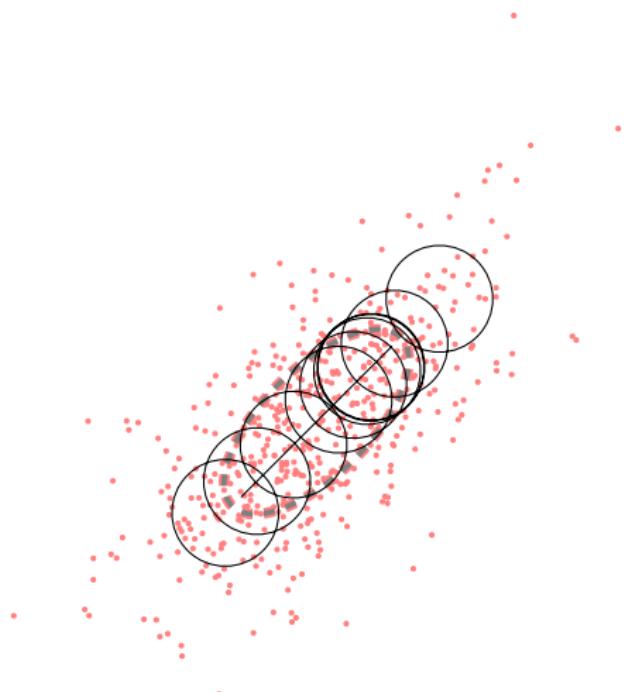
To find the ML values of  $(\Lambda, \psi)$  we could optimise numerically (gradient ascent / Newton's method), or we could use a different iterative algorithm called EM which we'll introduce soon.

In fact, however, ML for PPCA is more straightforward in principle, as we will see by first considering the limit  $\psi \rightarrow 0$ .

[Note: We may also add a constant mean  $\mu$  to the output, so as to model data that are not distributed around 0. In this case, the ML estimate  $\hat{\mu} = \frac{1}{N} \sum_n \mathbf{x}_n$  and we can define  $S = \frac{1}{N} \sum_n (\mathbf{x} - \hat{\mu})(\mathbf{x} - \hat{\mu})^T$  in the likelihood above.]

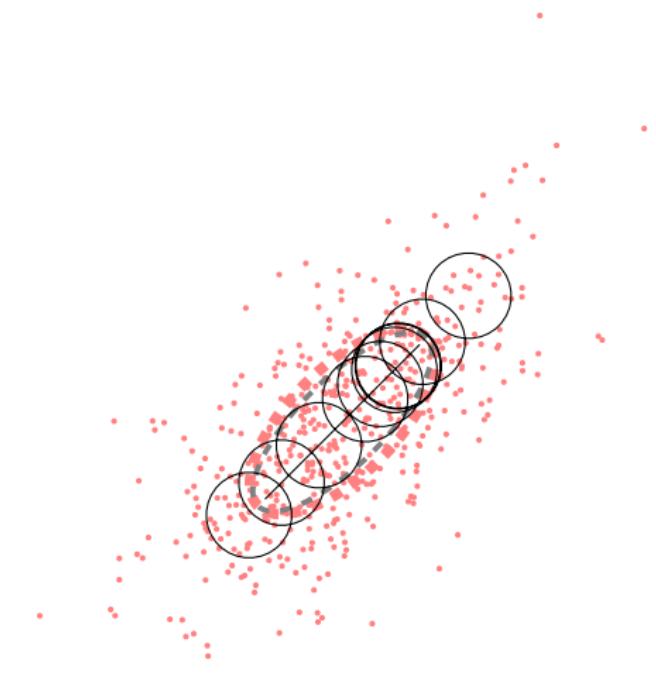
## The $\psi \rightarrow 0$ limit

As  $\psi \rightarrow 0$ , the latent model can only capture  $K$  dimensions of variance.



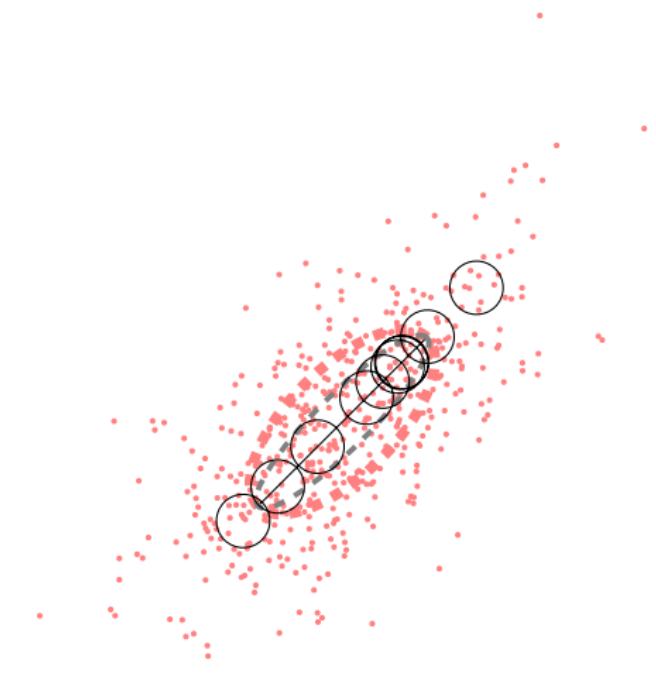
## The $\psi \rightarrow 0$ limit

As  $\psi \rightarrow 0$ , the latent model can only capture  $K$  dimensions of variance.



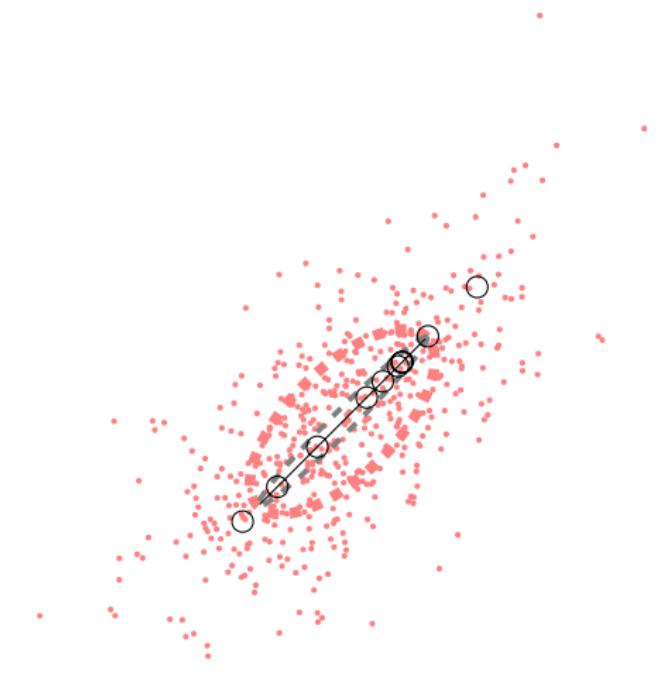
## The $\psi \rightarrow 0$ limit

As  $\psi \rightarrow 0$ , the latent model can only capture  $K$  dimensions of variance.



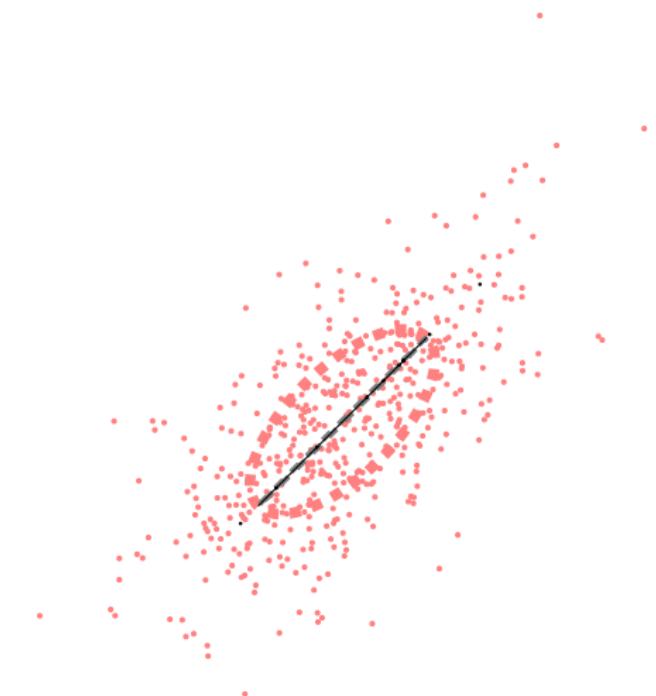
## The $\psi \rightarrow 0$ limit

As  $\psi \rightarrow 0$ , the latent model can only capture  $K$  dimensions of variance.



## The $\psi \rightarrow 0$ limit

As  $\psi \rightarrow 0$ , the latent model can only capture  $K$  dimensions of variance.



In a Gaussian model, the ML parameters will find the  $K$ -dimensional space of **most** variance  
 $\Rightarrow$  **PCA**.

## ML learning for PPCA

$$\ell = -\frac{N}{2} \log |2\pi C| - \frac{N}{2} \text{Tr}[C^{-1}S] \quad \text{where } C = \Lambda\Lambda^T + \psi I$$

$$\frac{\partial \ell}{\partial \Lambda} = \frac{N}{2} \left( -\frac{\partial}{\partial \Lambda} |C| - \frac{\partial}{\partial \Lambda} \text{Tr}[C^{-1}S] \right) = \frac{N}{2} (-C^{-1}\Lambda + C^{-1}SC^{-1}\Lambda)$$

So at the stationary points we have  $SC^{-1}\Lambda = \Lambda$ . This implies either:

- ▶  $\Lambda = 0$ , which turns out to be a minimum.
- ▶  $C = S \Rightarrow \Lambda\Lambda^T = S - \psi I$ . Now  $\text{rank}(\Lambda\Lambda^T) \leq K \Rightarrow \text{rank}(S - \psi I) \leq K \Rightarrow S$  has  $D - K$  eigenvalues  $= \psi$  and  $\Lambda$  aligns with space of remaining eigenvectors.
- ▶ or, taking the SVD:  $\Lambda = ULV^T$ :

$$\begin{aligned} S(ULV^T VL U^T + \psi I)^{-1} ULV^T &= ULV^T && \times VL^{-1} \\ \Rightarrow S(UL^2 U^T + \psi I)^{-1} U &= U && U(L^2 + \psi I) = (UL^2 U^T + \psi I)U \\ \Rightarrow SU(L^2 + \psi I)^{-1} &= U && \Rightarrow (UL^2 U^T + \psi I)^{-1} U = U(L^2 + \psi I)^{-1} \\ \Rightarrow SU &= U \underbrace{(L^2 + \psi I)}_{\text{diagonal}} && \times (L^2 + \psi I) \end{aligned}$$

$\Rightarrow$  columns of  $U$  are eigenvectors of  $S$  with eigenvalues given by  $l_i^2 + \psi$ .

Thus,  $\Lambda = ULV^T$  spans a space defined by  $K$  eigenvectors of  $S$ ; and the lengths of the column vectors of  $L$  are given by the eigenvalues  $-\psi$  ( $V$  selects an arbitrary basis in the latent space).

Remains to show (we won't, but it's intuitively reasonable) that the global ML solution is attained when  $\Lambda$  aligns with the  $K$  leading eigenvectors.

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

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

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda \mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda \mathbf{y}_n)\right\} \end{aligned}$$

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda \mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda \mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda \mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda \mathbf{y}_n)]\right\} \end{aligned}$$

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \end{aligned}$$

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda \mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda \mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda \mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda \mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \Sigma^{-1} \mathbf{y}_n - 2\mathbf{y}_n^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \Sigma^{-1} \mathbf{y}_n - 2\mathbf{y}_n^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

So  $\Sigma = (I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} = I - \beta \Lambda$  and  $\mu = \Sigma \Lambda^\top \Psi^{-1} \mathbf{x}_n = \beta \mathbf{x}_n$ . Where  $\beta = \Sigma \Lambda^\top \Psi^{-1}$ .

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \Sigma^{-1} \mathbf{y}_n - 2\mathbf{y}_n^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

So  $\Sigma = (I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} = I - \beta \Lambda$  and  $\mu = \Sigma \Lambda^\top \Psi^{-1} \mathbf{x}_n = \beta \mathbf{x}_n$ . Where  $\beta = \Sigma \Lambda^\top \Psi^{-1}$ .

- ▶ Thus,  $\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} \Lambda^\top \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^\top + \Psi)^{-1} \mathbf{x}_n$

## PPCA latents

- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \Sigma^{-1} \mathbf{y}_n - 2\mathbf{y}_n^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

So  $\Sigma = (I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} = I - \beta \Lambda$  and  $\mu = \Sigma \Lambda^\top \Psi^{-1} \mathbf{x}_n = \beta \mathbf{x}_n$ . Where  $\beta = \Sigma \Lambda^\top \Psi^{-1}$ .

- ▶ Thus,  $\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} \Lambda^\top \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^\top + \Psi)^{-1} \mathbf{x}_n$
- ▶ This is not the same projection. PPCA takes into account noise **in** the principal subspace.

## PPCA latents

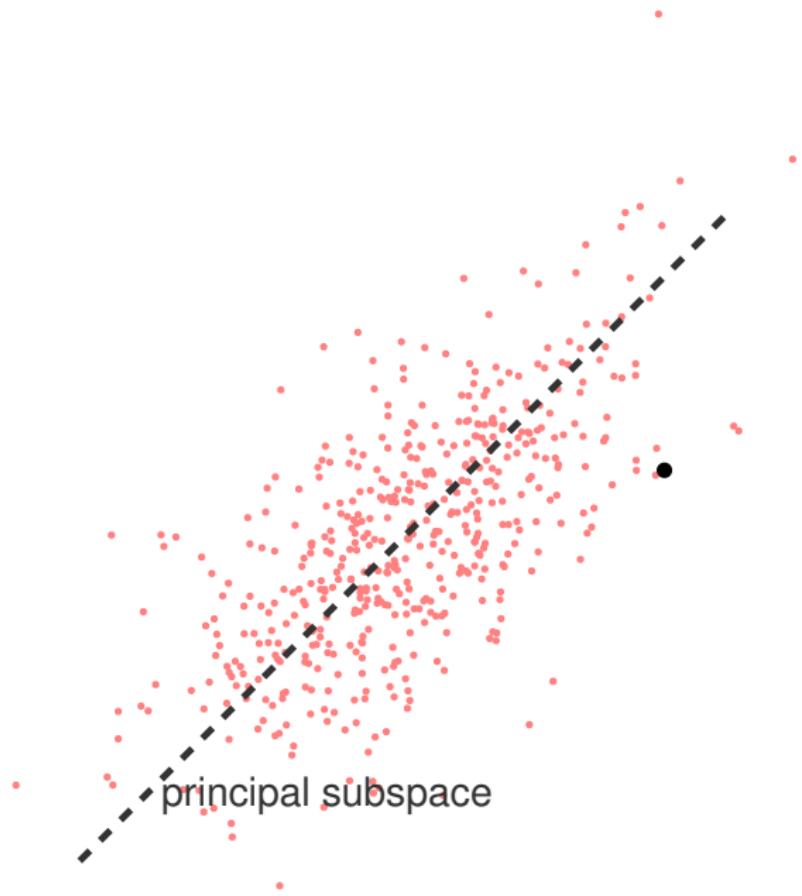
- ▶ In PCA the “noise” is orthogonal to the subspace, and we can project  $\mathbf{x}_n \rightarrow \tilde{\mathbf{x}}_n$  trivially.
- ▶ In PPCA, the noise is more sensible (equal in all directions). But what is the projection?  
Find the expected value of  $\mathbf{y}_n | \mathbf{x}_n$  and then take  $\tilde{\mathbf{x}}_n = \Lambda \bar{\mathbf{y}}_n$ .
- ▶ **Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n | \theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n | \mathbf{y}_n) \\ &= (2\pi)^{-\frac{K}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top (I + \Lambda^\top \Psi^{-1} \Lambda) \mathbf{y}_n - 2\mathbf{y}_n^\top \Lambda^\top \Psi^{-1} \mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^\top \Sigma^{-1} \mathbf{y}_n - 2\mathbf{y}_n^\top \Sigma^{-1} \mu + \mu^\top \Sigma^{-1} \mu]\right\} \end{aligned}$$

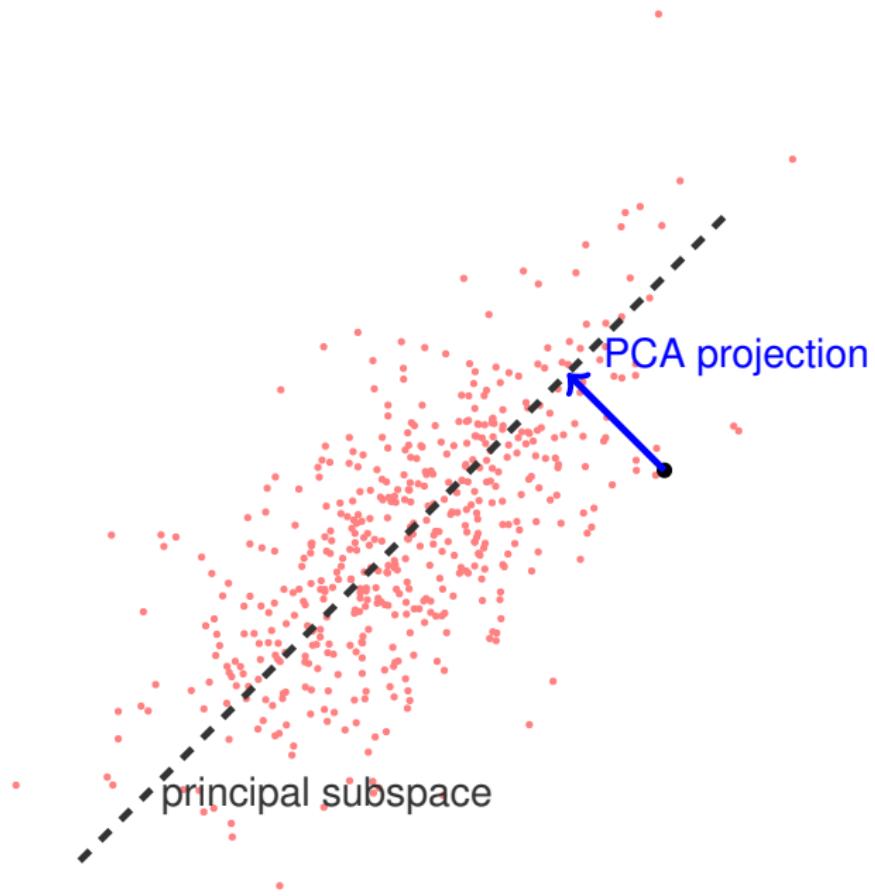
So  $\Sigma = (I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} = I - \beta \Lambda$  and  $\mu = \Sigma \Lambda^\top \Psi^{-1} \mathbf{x}_n = \beta \mathbf{x}_n$ . Where  $\beta = \Sigma \Lambda^\top \Psi^{-1}$ .

- ▶ Thus,  $\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^\top \Psi^{-1} \Lambda)^{-1} \Lambda^\top \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^\top + \Psi)^{-1} \mathbf{x}_n$
- ▶ This is not the same projection. PPCA takes into account noise **in** the principal subspace.
- ▶ As  $\psi \rightarrow 0$ , the PPCA estimate  $\rightarrow$  the PCA value.

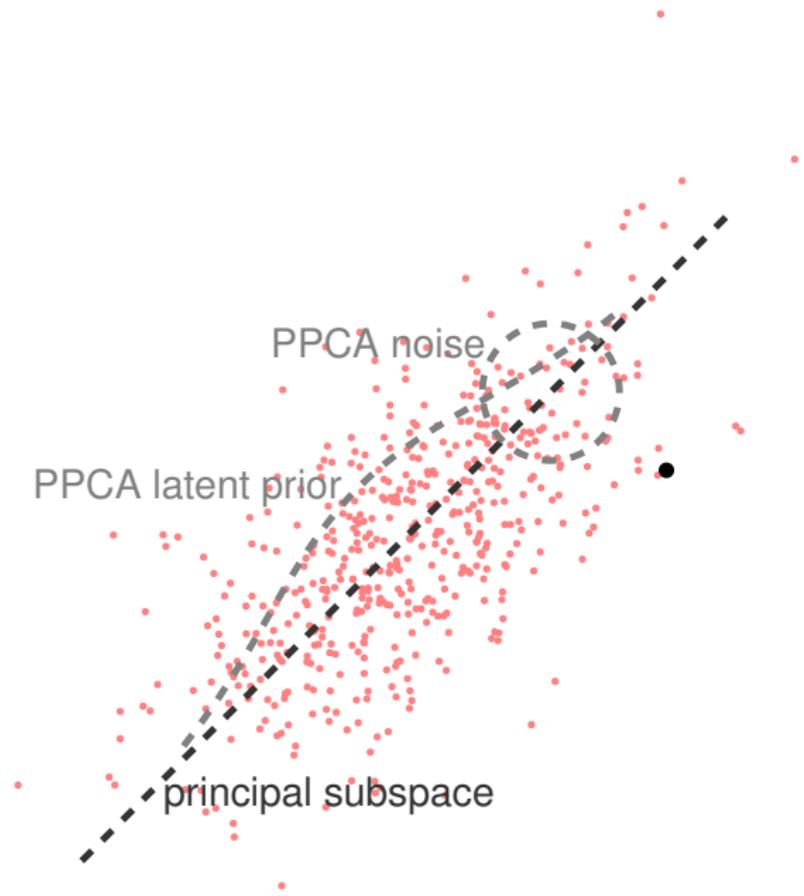
## PPCA latents



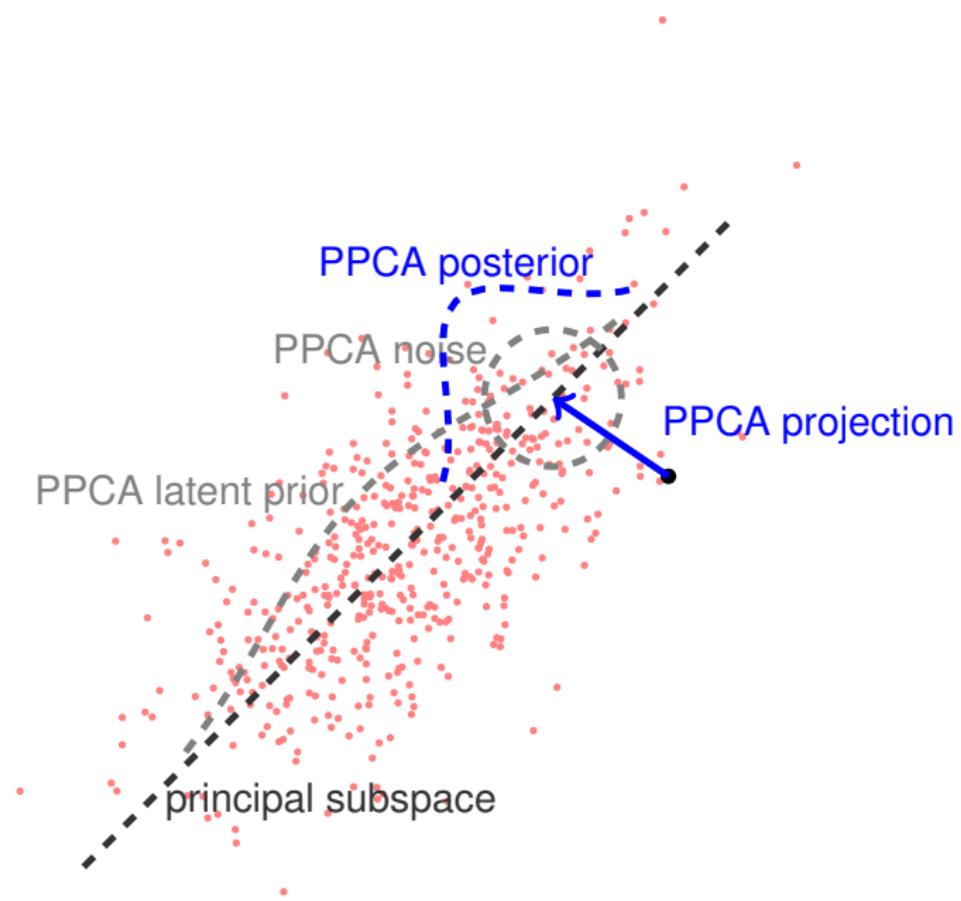
## PPCA latents



## PPCA latents



## PPCA latents



## Factor Analysis

If dimensions are not equivalent, equal variance assumption is inappropriate.

Data:  $\mathcal{D} = \mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}; \mathbf{x}_i \in \mathbb{R}^D$

Latents:  $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}; \mathbf{y}_i \in \mathbb{R}^K$

Linear generative model:  $x_d = \sum_{k=1}^K \Lambda_{dk} y_k + \epsilon_d$

- ▶  $y_k$  are independent  $\mathcal{N}(0, 1)$  Gaussian factors
- ▶  $\epsilon_d$  are independent  $\mathcal{N}(0, \Psi_{dd})$  Gaussian noise
- ▶  $K < D$

Model for observations  $\mathbf{x}$  is still a correlated Gaussian:

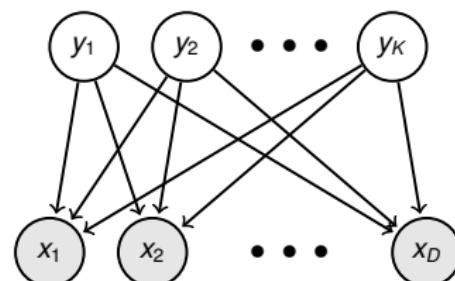
$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \Psi)$$

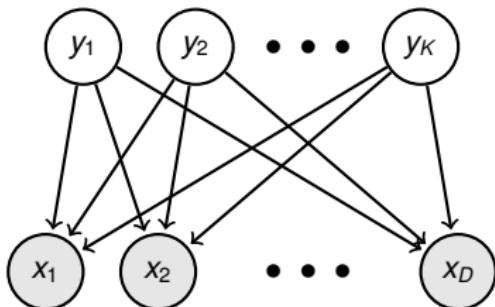
$$p(\mathbf{x}) = \int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y} = \mathcal{N}\left(\mathbf{0}, \Lambda\Lambda^\top + \Psi\right)$$

where  $\Lambda$  is a  $D \times K$ , and  $\Psi$  is  $K \times K$  and diagonal.

**Dimensionality Reduction:** Finds a low-dimensional projection of high dimensional data that captures the correlation structure of the data.



## Factor Analysis (cont.)



- ▶ ML learning finds  $\Lambda$  (“common factors”) and  $\Psi$  (“unique factors” or “uniquenesses”) given data
- ▶ parameters (corrected for symmetries):  $DK + D - \frac{K(K-1)}{2}$
- ▶ If number of parameters  $> \frac{D(D+1)}{2}$  model is not identifiable (even after accounting for rotational degeneracy discussed later)
- ▶ no closed form solution for ML params:  $\mathcal{N}(0, \Lambda\Lambda^T + \Psi)$

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

And  $\Lambda$  is not unique: the latent space may be transformed by an arbitrary orthogonal transform  $U$  ( $U^T U = UU^T = I$ ) without changing the likelihood.

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

And  $\Lambda$  is not unique: the latent space may be transformed by an arbitrary orthogonal transform  $U$  ( $U^T U = UU^T = I$ ) without changing the likelihood.

$$\tilde{\mathbf{y}} = U\mathbf{y} \quad \text{and} \quad \tilde{\Lambda} = \Lambda U^T \quad \Rightarrow \quad \tilde{\Lambda} \tilde{\mathbf{y}} = \Lambda U^T U\mathbf{y} = \Lambda\mathbf{y}$$

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

And  $\Lambda$  is not unique: the latent space may be transformed by an arbitrary orthogonal transform  $U$  ( $U^T U = UU^T = I$ ) without changing the likelihood.

$$\tilde{\mathbf{y}} = U\mathbf{y} \quad \text{and} \quad \tilde{\Lambda} = \Lambda U^T \quad \Rightarrow \quad \tilde{\Lambda} \tilde{\mathbf{y}} = \Lambda U^T U\mathbf{y} = \Lambda\mathbf{y}$$

$$-\ell = \frac{1}{2} \log |2\pi(\Lambda \Lambda^T + \Psi)| + \frac{1}{2} \mathbf{x}^T (\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}$$

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

And  $\Lambda$  is not unique: the latent space may be transformed by an arbitrary orthogonal transform  $U$  ( $U^T U = UU^T = I$ ) without changing the likelihood.

$$\tilde{\mathbf{y}} = U\mathbf{y} \quad \text{and} \quad \tilde{\Lambda} = \Lambda U^T \quad \Rightarrow \quad \tilde{\Lambda} \tilde{\mathbf{y}} = \Lambda U^T U\mathbf{y} = \Lambda\mathbf{y}$$

$$-\ell = \frac{1}{2} \log \left| 2\pi(\Lambda U^T U \Lambda^T + \Psi) \right| + \frac{1}{2} \mathbf{x}^T (\Lambda U^T U \Lambda^T + \Psi)^{-1} \mathbf{x}$$

## Factor Analysis projections

Our analysis for PPCA still applies:

$$\tilde{\mathbf{x}}_n = \Lambda(I + \Lambda^T \Psi^{-1} \Lambda)^{-1} \Lambda^T \Psi^{-1} \mathbf{x}_n = \mathbf{x}_n - \Psi(\Lambda \Lambda^T + \Psi)^{-1} \mathbf{x}_n$$

but now  $\Psi$  is diagonal but not spherical.

Note, though, that  $\Lambda$  is generally different from that found by PPCA.

And  $\Lambda$  is not unique: the latent space may be transformed by an arbitrary orthogonal transform  $U$  ( $U^T U = UU^T = I$ ) without changing the likelihood.

$$\tilde{\mathbf{y}} = U\mathbf{y} \quad \text{and} \quad \tilde{\Lambda} = \Lambda U^T \quad \Rightarrow \quad \tilde{\Lambda} \tilde{\mathbf{y}} = \Lambda U^T U\mathbf{y} = \Lambda\mathbf{y}$$

$$\begin{aligned} -\ell &= \frac{1}{2} \log |2\pi(\Lambda U^T U \Lambda^T + \Psi)| + \frac{1}{2} \mathbf{x}^T (\Lambda U^T U \Lambda^T + \Psi)^{-1} \mathbf{x} \\ &= \frac{1}{2} \log |2\pi(\tilde{\Lambda} \tilde{\Lambda}^T + \Psi)| + \frac{1}{2} \mathbf{x}^T (\tilde{\Lambda} \tilde{\Lambda}^T + \Psi)^{-1} \mathbf{x} \end{aligned}$$

## Gradient methods for learning FA

Optimise negative log-likelihood:

$$-\ell = \frac{1}{2} \log |2\pi(\Lambda\Lambda^T + \Psi)| + \frac{1}{2}\mathbf{x}^T(\Lambda\Lambda^T + \Psi)^{-1}\mathbf{x}$$

w.r.t.  $\Lambda$  and  $\Psi$  (need matrix calculus) subject to constraints.

- ▶ No spectral short-cut exists.
- ▶ Likelihood can have more than one (local) optimum, making it difficult to find the global value.
- ▶ For some data ("Heywood cases") likelihood may grow unboundedly by taking one or more  $\Psi_{dd} \rightarrow 0$ . Can eliminate by assuming a prior on  $\Psi$  with zero density at  $\Psi_{dd} = 0$ , but results sensitive to precise choice of prior.

Expectation maximisation (later) provides an alternative approach to maximisation, but doesn't solve these issues.

## FA vs PCA

- ▶ PCA and PPCA are rotationally invariant; FA is not

If  $\mathbf{x} \rightarrow U\mathbf{x}$  for unitary  $U$ , then  $\rho_{(i)}^{\text{PCA}} \rightarrow U\rho_{(i)}^{\text{PCA}}$

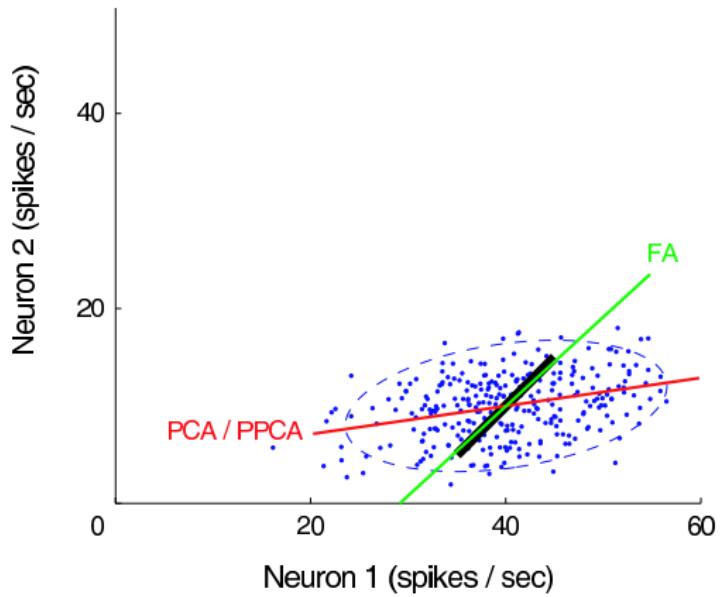
- ▶ FA is measurement scale invariant; PCA and PPCA are not

If  $\mathbf{x} \rightarrow S\mathbf{x}$  for diagonal  $S$ , then  $\rho_{(i)}^{\text{FA}} \rightarrow S\rho_{(i)}^{\text{FA}}$

- ▶ FA and PPCA define a probabilistic model; PCA does not

[Note: it may be tempting to try to eliminate the scale-dependence of (P)PCA by pre-processing data to equalise total variance on each axis. But P(PCA) assume equal *noise* variance. Total variance has contributions from both  $\Lambda\Lambda^\top$  and noise, so this approach does not exactly solve the problem.]

## FA vs PCA for neural data



## The Expectation Maximisation (EM) algorithm

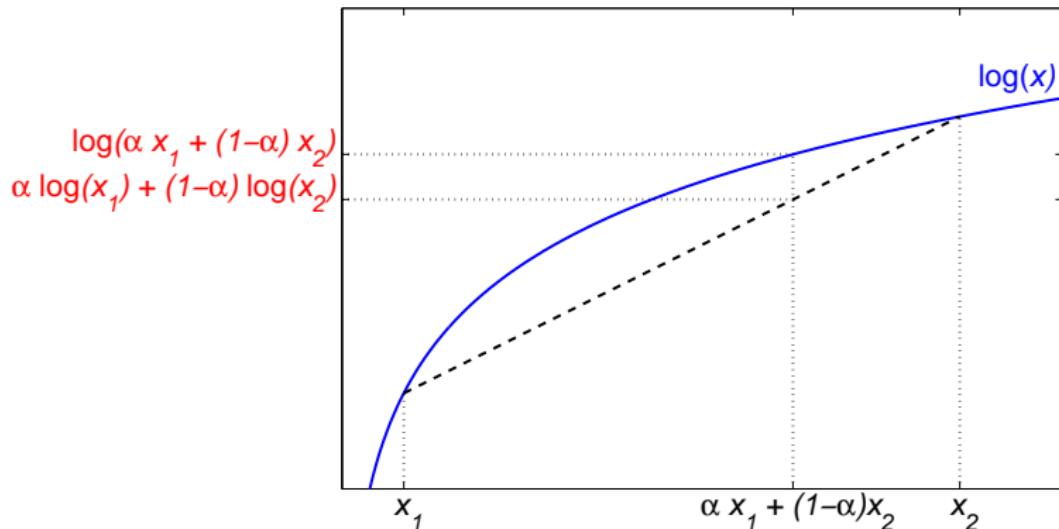
The EM algorithm finds a (local) maximum of a latent variable model likelihood. It starts from arbitrary values of the parameters, and iterates two steps:

**E step:** Fill in values of latent variables according to posterior given data.

**M step:** Maximise likelihood as if latent variables were not hidden.

- ▶ Useful in models where learning would be easy if hidden variables were, in fact, observed (e.g. MoGs).
- ▶ Decomposes difficult problems into series of tractable steps.
- ▶ No learning rate.
- ▶ Framework lends itself to principled approximations.

## Jensen's inequality



For  $\alpha_i \geq 0$ ,  $\sum \alpha_i = 1$  and any  $\{x_i > 0\}$

$$\log \left( \sum_i \alpha_i x_i \right) \geq \sum_i \alpha_i \log(x_i)$$

Equality if and only if  $\alpha_i = 1$  for some  $i$  (and therefore all others are 0).

## The free energy for a latent variable model

Observed data  $\mathcal{X} = \{\mathbf{x}_i\}$ ; Latent variables  $\mathcal{Y} = \{\mathbf{y}_i\}$ ; Parameters  $\theta$ .

**Goal:** Maximize the log likelihood (i.e. ML learning) wrt  $\theta$ :

$$\ell(\theta) = \log P(\mathcal{X}|\theta) = \log \int P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y},$$

## The free energy for a latent variable model

Observed data  $\mathcal{X} = \{\mathbf{x}_i\}$ ; Latent variables  $\mathcal{Y} = \{\mathbf{y}_i\}$ ; Parameters  $\theta$ .

**Goal:** Maximize the log likelihood (i.e. ML learning) wrt  $\theta$ :

$$\ell(\theta) = \log P(\mathcal{X}|\theta) = \log \int P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y},$$

Any distribution,  $q(\mathcal{Y})$ , over the hidden variables can be used to obtain a lower bound on the log likelihood using Jensen's inequality:

$$\ell(\theta) = \log \int q(\mathcal{Y}) \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \geq \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y}.$$

## The free energy for a latent variable model

Observed data  $\mathcal{X} = \{\mathbf{x}_i\}$ ; Latent variables  $\mathcal{Y} = \{\mathbf{y}_i\}$ ; Parameters  $\theta$ .

**Goal:** Maximize the log likelihood (i.e. ML learning) wrt  $\theta$ :

$$\ell(\theta) = \log P(\mathcal{X}|\theta) = \log \int P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y},$$

Any distribution,  $q(\mathcal{Y})$ , over the hidden variables can be used to obtain a lower bound on the log likelihood using Jensen's inequality:

$$\ell(\theta) = \log \int q(\mathcal{Y}) \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \geq \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \stackrel{\text{def}}{=} \mathcal{F}(q, \theta).$$

## The free energy for a latent variable model

Observed data  $\mathcal{X} = \{\mathbf{x}_i\}$ ; Latent variables  $\mathcal{Y} = \{\mathbf{y}_i\}$ ; Parameters  $\theta$ .

**Goal:** Maximize the log likelihood (i.e. ML learning) wrt  $\theta$ :

$$\ell(\theta) = \log P(\mathcal{X}|\theta) = \log \int P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y},$$

Any distribution,  $q(\mathcal{Y})$ , over the hidden variables can be used to obtain a lower bound on the log likelihood using Jensen's inequality:

$$\ell(\theta) = \log \int q(\mathcal{Y}) \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \geq \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \stackrel{\text{def}}{=} \mathcal{F}(q, \theta).$$

Now,

$$\begin{aligned} \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} &= \int q(\mathcal{Y}) \log P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y} - \int q(\mathcal{Y}) \log q(\mathcal{Y}) d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y} + \mathbf{H}[q], \end{aligned}$$

where  $\mathbf{H}[q]$  is the entropy of  $q(\mathcal{Y})$ .

## The free energy for a latent variable model

Observed data  $\mathcal{X} = \{\mathbf{x}_i\}$ ; Latent variables  $\mathcal{Y} = \{\mathbf{y}_i\}$ ; Parameters  $\theta$ .

**Goal:** Maximize the log likelihood (i.e. ML learning) wrt  $\theta$ :

$$\ell(\theta) = \log P(\mathcal{X}|\theta) = \log \int P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y},$$

Any distribution,  $q(\mathcal{Y})$ , over the hidden variables can be used to obtain a lower bound on the log likelihood using Jensen's inequality:

$$\ell(\theta) = \log \int q(\mathcal{Y}) \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \geq \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \stackrel{\text{def}}{=} \mathcal{F}(q, \theta).$$

Now,

$$\begin{aligned} \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} &= \int q(\mathcal{Y}) \log P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y} - \int q(\mathcal{Y}) \log q(\mathcal{Y}) d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{Y}, \mathcal{X}|\theta) d\mathcal{Y} + \mathbf{H}[q], \end{aligned}$$

where  $\mathbf{H}[q]$  is the entropy of  $q(\mathcal{Y})$ .

So:

$$\mathcal{F}(q, \theta) = \langle \log P(\mathcal{Y}, \mathcal{X}|\theta) \rangle_{q(\mathcal{Y})} + \mathbf{H}[q]$$

## The E and M steps of EM

The lower bound on the log likelihood is given by:

$$\mathcal{F}(q, \theta) = \langle \log P(\mathcal{Y}, \mathcal{X} | \theta) \rangle_{q(\mathcal{Y})} + \mathbf{H}[q],$$

EM alternates between:

- ▶ **E step:** optimize  $\mathcal{F}(q, \theta)$  wrt distribution over hidden variables holding parameters fixed:

$$q^{(k)}(\mathcal{Y}) := \underset{q(\mathcal{Y})}{\operatorname{argmax}} \mathcal{F}(q(\mathcal{Y}), \theta^{(k-1)}).$$

## The E and M steps of EM

The lower bound on the log likelihood is given by:

$$\mathcal{F}(q, \theta) = \langle \log P(\mathcal{Y}, \mathcal{X} | \theta) \rangle_{q(\mathcal{Y})} + \mathbf{H}[q],$$

EM alternates between:

- ▶ **E step:** optimize  $\mathcal{F}(q, \theta)$  wrt distribution over hidden variables holding parameters fixed:

$$q^{(k)}(\mathcal{Y}) := \operatorname{argmax}_{q(\mathcal{Y})} \mathcal{F}(q(\mathcal{Y}), \theta^{(k-1)}).$$

- ▶ **M step:** maximize  $\mathcal{F}(q, \theta)$  wrt parameters holding hidden distribution fixed:

$$\theta^{(k)} := \operatorname{argmax}_{\theta} \mathcal{F}(q^{(k)}(\mathcal{Y}), \theta) = \operatorname{argmax}_{\theta} \langle \log P(\mathcal{Y}, \mathcal{X} | \theta) \rangle_{q^{(k)}(\mathcal{Y})}$$

The second equality comes from the fact that the entropy of  $q(\mathcal{Y})$  does not depend directly on  $\theta$ .

## The E and M steps of EM

The lower bound on the log likelihood is given by:

$$\mathcal{F}(q, \theta) = \langle \log P(\mathcal{Y}, \mathcal{X} | \theta) \rangle_{q(\mathcal{Y})} + \mathbf{H}[q],$$

EM alternates between:

- ▶ **E step:** optimize  $\mathcal{F}(q, \theta)$  wrt distribution over hidden variables holding parameters fixed:

$$q^{(k)}(\mathcal{Y}) := \underset{q(\mathcal{Y})}{\operatorname{argmax}} \mathcal{F}(q(\mathcal{Y}), \theta^{(k-1)}).$$

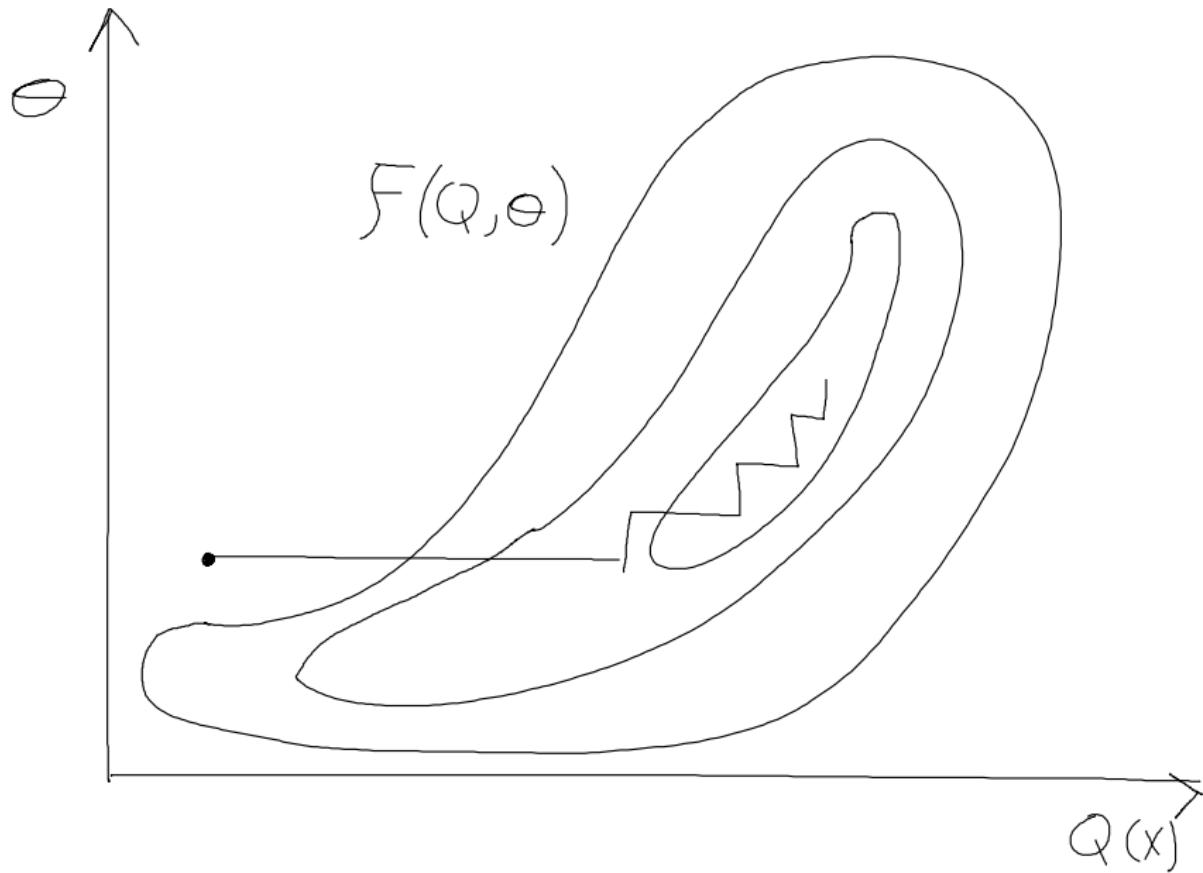
- ▶ **M step:** maximize  $\mathcal{F}(q, \theta)$  wrt parameters holding hidden distribution fixed:

$$\theta^{(k)} := \underset{\theta}{\operatorname{argmax}} \mathcal{F}(q^{(k)}(\mathcal{Y}), \theta) = \underset{\theta}{\operatorname{argmax}} \langle \log P(\mathcal{Y}, \mathcal{X} | \theta) \rangle_{q^{(k)}(\mathcal{Y})}$$

The second equality comes from the fact that the entropy of  $q(\mathcal{Y})$  does not depend directly on  $\theta$ .

**Note:** partial M-steps or (less commonly) E-steps can suffice.

## EM as coordinate ascent in $\mathcal{F}$



## The E step

The free energy can be re-written

$$\mathcal{F}(q, \theta) = \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y}$$

## The E step

The free energy can be re-written

$$\begin{aligned}\mathcal{F}(q, \theta) &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)P(\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y}\end{aligned}$$

## The E step

The free energy can be re-written

$$\begin{aligned}\mathcal{F}(q, \theta) &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)P(\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{X}|\theta) d\mathcal{Y} + \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)}{q(\mathcal{Y})} d\mathcal{Y}\end{aligned}$$

## The E step

The free energy can be re-written

$$\begin{aligned}\mathcal{F}(q, \theta) &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)P(\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{X}|\theta) d\mathcal{Y} + \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \ell(\theta) - \mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)]\end{aligned}$$

The second term is the Kullback-Leibler divergence.

## The E step

The free energy can be re-written

$$\begin{aligned}\mathcal{F}(q, \theta) &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)P(\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{X}|\theta) d\mathcal{Y} + \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \ell(\theta) - \mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)]\end{aligned}$$

The second term is the Kullback-Leibler divergence.

This means that, for fixed  $\theta$ ,  $\mathcal{F}$  is bounded above by  $\ell$ , and achieves that bound when  $\mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)] = 0$ .

## The E step

The free energy can be re-written

$$\begin{aligned}\mathcal{F}(q, \theta) &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}, \mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)P(\mathcal{X}|\theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \int q(\mathcal{Y}) \log P(\mathcal{X}|\theta) d\mathcal{Y} + \int q(\mathcal{Y}) \log \frac{P(\mathcal{Y}|\mathcal{X}, \theta)}{q(\mathcal{Y})} d\mathcal{Y} \\ &= \ell(\theta) - \mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)]\end{aligned}$$

The second term is the Kullback-Leibler divergence.

This means that, for fixed  $\theta$ ,  $\mathcal{F}$  is bounded above by  $\ell$ , and achieves that bound when  $\mathbf{KL}[q(\mathcal{Y}) \| P(\mathcal{Y}|\mathcal{X}, \theta)] = 0$ .

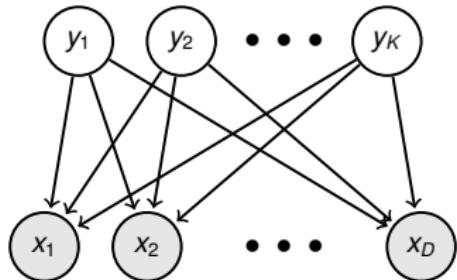
But  $\mathbf{KL}[q \| p]$  is zero if and only if  $q = p$ .

So, the E step simply sets

$$q^{(k)}(\mathcal{Y}) = P(\mathcal{Y}|\mathcal{X}, \theta^{(k-1)})$$

and, after an E step, the free energy equals the likelihood.

## EM for Factor Analysis



The model for  $\mathbf{x}$ :

$$p(\mathbf{x}|\theta) = \int p(\mathbf{y}|\theta)p(\mathbf{x}|\mathbf{y}, \theta)d\mathbf{y} = \mathcal{N}(0, \Lambda\Lambda^T + \Psi)$$

Model parameters:  $\theta = \{\Lambda, \Psi\}$ .

**E step:** For each data point  $\mathbf{x}_n$ , compute the posterior distribution of hidden factors given the observed data:  $q_n(\mathbf{y}_n) = p(\mathbf{y}_n|\mathbf{x}_n, \theta_t)$ .

**M step:** Find the  $\theta_{t+1}$  that maximises  $\mathcal{F}(q, \theta)$ :

$$\begin{aligned}\mathcal{F}(q, \theta) &= \sum_n \int q_n(\mathbf{y}_n) [\log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) - \log q_n(\mathbf{y}_n)] d\mathbf{y}_n \\ &= \sum_n \int q_n(\mathbf{y}_n) [\log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta)] d\mathbf{y}_n + c.\end{aligned}$$

## The E step for Factor Analysis

**E step:** For each data point  $\mathbf{x}_n$ , compute the posterior distribution of hidden factors given the observed data:  $q_n(\mathbf{y}_n) = p(\mathbf{y}_n|\mathbf{x}_n, \theta) = p(\mathbf{y}_n, \mathbf{x}_n|\theta)/p(\mathbf{x}_n|\theta)$

**Tactic:** write  $p(\mathbf{y}_n, \mathbf{x}_n|\theta)$ , consider  $\mathbf{x}_n$  to be fixed. What is this as a function of  $\mathbf{y}_n$ ?

$$\begin{aligned} p(\mathbf{y}_n, \mathbf{x}_n) &= p(\mathbf{y}_n)p(\mathbf{x}_n|\mathbf{y}_n) \\ &= (2\pi)^{-\frac{\kappa}{2}} \exp\left\{-\frac{1}{2}\mathbf{y}_n^T\mathbf{y}_n\right\} |2\pi\Psi|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)\right\} \\ &= c \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^T\mathbf{y}_n + (\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)]\right\} \\ &= c' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^T(I + \Lambda^T\Psi^{-1}\Lambda)\mathbf{y}_n - 2\mathbf{y}_n^T\Lambda^T\Psi^{-1}\mathbf{x}_n]\right\} \\ &= c'' \times \exp\left\{-\frac{1}{2}[\mathbf{y}_n^T\Sigma^{-1}\mathbf{y}_n - 2\mathbf{y}_n^T\Sigma^{-1}\mu_n + \mu_n^T\Sigma^{-1}\mu_n]\right\} \end{aligned}$$

So  $\Sigma = (I + \Lambda^T\Psi^{-1}\Lambda)^{-1} = I - \beta\Lambda$  and  $\mu_n = \Sigma\Lambda^T\Psi^{-1}\mathbf{x}_n = \beta\mathbf{x}_n$ . Where  $\beta = \Sigma\Lambda^T\Psi^{-1}$ . Note that  $\mu_n$  is a linear function of  $\mathbf{x}_n$  and  $\Sigma$  does not depend on  $\mathbf{x}_n$ .

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta)$$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta)$$

$$= c - \frac{1}{2}\mathbf{y}_n^\top \mathbf{y}_n - \frac{1}{2}\log |\Psi| - \frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^\top \Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n)$$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\begin{aligned} & \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \\ &= c - \frac{1}{2}\mathbf{y}_n^T\mathbf{y}_n - \frac{1}{2}\log |\Psi| - \frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n) \\ &= c' - \frac{1}{2}\log |\Psi| - \frac{1}{2} \left[ \mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \mathbf{y}_n^T\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n \right] \end{aligned}$$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\begin{aligned}& \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \\&= c - \frac{1}{2}\mathbf{y}_n^T\mathbf{y}_n - \frac{1}{2}\log|\Psi| - \frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n) \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \mathbf{y}_n^T\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\right] \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \text{Tr}[\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\mathbf{y}_n^T]\right]\end{aligned}$$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\begin{aligned}& \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \\&= c - \frac{1}{2}\mathbf{y}_n^T\mathbf{y}_n - \frac{1}{2}\log|\Psi| - \frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n) \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \mathbf{y}_n^T\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\right] \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \text{Tr}\left[\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\mathbf{y}_n^T\right]\right]\end{aligned}$$

Taking expectations wrt  $q_n(\mathbf{y}_n)$ :

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\begin{aligned}& \log p(\mathbf{y}_n|\theta) + \log p(\mathbf{x}_n|\mathbf{y}_n, \theta) \\&= c - \frac{1}{2}\mathbf{y}_n^T\mathbf{y}_n - \frac{1}{2}\log|\Psi| - \frac{1}{2}(\mathbf{x}_n - \Lambda\mathbf{y}_n)^T\Psi^{-1}(\mathbf{x}_n - \Lambda\mathbf{y}_n) \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \mathbf{y}_n^T\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\right] \\&= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mathbf{y}_n + \text{Tr}\left[\Lambda^T\Psi^{-1}\Lambda\mathbf{y}_n\mathbf{y}_n^T\right]\right]\end{aligned}$$

Taking expectations wrt  $q_n(\mathbf{y}_n)$ :

$$= c' - \frac{1}{2}\log|\Psi| - \frac{1}{2}\left[\mathbf{x}_n^T\Psi^{-1}\mathbf{x}_n - 2\mathbf{x}_n^T\Psi^{-1}\Lambda\mu_n + \text{Tr}\left[\Lambda^T\Psi^{-1}\Lambda(\mu_n\mu_n^T + \Sigma)\right]\right]$$

## The M step for Factor Analysis

**M step:** Find  $\theta_{t+1}$  by maximising  $\mathcal{F} = \sum_n \langle \log p(\mathbf{y}_n | \theta) + \log p(\mathbf{x}_n | \mathbf{y}_n, \theta) \rangle_{q_n(\mathbf{y}_n)} + c$

$$\begin{aligned}& \log p(\mathbf{y}_n | \theta) + \log p(\mathbf{x}_n | \mathbf{y}_n, \theta) \\&= c - \frac{1}{2} \mathbf{y}_n^T \mathbf{y}_n - \frac{1}{2} \log |\Psi| - \frac{1}{2} (\mathbf{x}_n - \Lambda \mathbf{y}_n)^T \Psi^{-1} (\mathbf{x}_n - \Lambda \mathbf{y}_n) \\&= c' - \frac{1}{2} \log |\Psi| - \frac{1}{2} \left[ \mathbf{x}_n^T \Psi^{-1} \mathbf{x}_n - 2 \mathbf{x}_n^T \Psi^{-1} \Lambda \mathbf{y}_n + \mathbf{y}_n^T \Lambda^T \Psi^{-1} \Lambda \mathbf{y}_n \right] \\&= c' - \frac{1}{2} \log |\Psi| - \frac{1}{2} \left[ \mathbf{x}_n^T \Psi^{-1} \mathbf{x}_n - 2 \mathbf{x}_n^T \Psi^{-1} \Lambda \mathbf{y}_n + \text{Tr} \left[ \Lambda^T \Psi^{-1} \Lambda \mathbf{y}_n \mathbf{y}_n^T \right] \right]\end{aligned}$$

Taking expectations wrt  $q_n(\mathbf{y}_n)$ :

$$= c' - \frac{1}{2} \log |\Psi| - \frac{1}{2} \left[ \mathbf{x}_n^T \Psi^{-1} \mathbf{x}_n - 2 \mathbf{x}_n^T \Psi^{-1} \Lambda \mu_n + \text{Tr} \left[ \Lambda^T \Psi^{-1} \Lambda (\mu_n \mu_n^T + \Sigma) \right] \right]$$

Note that we don't need to know everything about  $q(\mathbf{y}_n)$ , just the moments  $\langle \mathbf{y}_n \rangle$  and  $\langle \mathbf{y}_n \mathbf{y}_n^T \rangle$ . These are the **expected sufficient statistics**.

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} \left[ \Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \right] \right]$$

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} \left[ \Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \right] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} \left[ \Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \right] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0$$

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} \left[ \Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \right] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\begin{aligned}\frac{\partial \mathcal{F}}{\partial \Lambda} &= \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0 \\ \Rightarrow \hat{\Lambda} &= \left( \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top \right) \left( \textcolor{red}{N\Sigma} + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right)^{-1}\end{aligned}$$

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} [\Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma)] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0$$

$$\Rightarrow \hat{\Lambda} = \left( \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top \right) \left( \textcolor{red}{N\Sigma} + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right)^{-1}$$

$$\frac{\partial \mathcal{F}}{\partial \Psi^{-1}} = \frac{N}{2} \Psi - \frac{1}{2} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

Note: we should actually only take derivatives w.r.t.  $\Psi_{dd}$  since  $\Psi$  is diagonal.

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} [\Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma)] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0$$

$$\Rightarrow \hat{\Lambda} = \left( \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top \right) \left( \textcolor{red}{N\Sigma} + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right)^{-1}$$

$$\frac{\partial \mathcal{F}}{\partial \Psi^{-1}} = \frac{N}{2} \Psi - \frac{1}{2} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

$$\Rightarrow \hat{\Psi} = \frac{1}{N} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

Note: we should actually only take derivatives w.r.t.  $\Psi_{dd}$  since  $\Psi$  is diagonal.

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} [\Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma)] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0$$

$$\Rightarrow \hat{\Lambda} = \left( \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top \right) \left( \textcolor{red}{N\Sigma} + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right)^{-1}$$

$$\frac{\partial \mathcal{F}}{\partial \Psi^{-1}} = \frac{N}{2} \Psi - \frac{1}{2} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

$$\Rightarrow \hat{\Psi} = \frac{1}{N} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

$$\hat{\Psi} = \textcolor{red}{\Lambda \Sigma \Lambda^\top} + \frac{1}{N} \sum_n (\mathbf{x}_n - \Lambda \boldsymbol{\mu}_n)(\mathbf{x}_n - \Lambda \boldsymbol{\mu}_n)^\top \quad (\text{squared residuals})$$

Note: we should actually only take derivatives w.r.t.  $\Psi_{dd}$  since  $\Psi$  is diagonal.

## The M step for Factor Analysis (cont.)

$$\mathcal{F} = c' - \frac{N}{2} \log |\Psi| - \frac{1}{2} \sum_n \left[ \mathbf{x}_n^\top \Psi^{-1} \mathbf{x}_n - 2\mathbf{x}_n^\top \Psi^{-1} \Lambda \boldsymbol{\mu}_n + \text{Tr} \left[ \Lambda^\top \Psi^{-1} \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \right] \right]$$

Taking derivatives wrt  $\Lambda$  and  $\Psi^{-1}$ , using  $\frac{\partial \text{Tr}[AB]}{\partial B} = A^\top$  and  $\frac{\partial \log |A|}{\partial A} = A^{-\top}$ :

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = \Psi^{-1} \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top - \Psi^{-1} \Lambda \left( N\Sigma + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right) = 0$$

$$\Rightarrow \hat{\Lambda} = \left( \sum_n \mathbf{x}_n \boldsymbol{\mu}_n^\top \right) \left( \textcolor{red}{N\Sigma} + \sum_n \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top \right)^{-1}$$

$$\frac{\partial \mathcal{F}}{\partial \Psi^{-1}} = \frac{N}{2} \Psi - \frac{1}{2} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

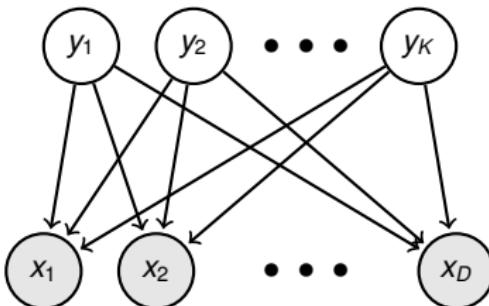
$$\Rightarrow \hat{\Psi} = \frac{1}{N} \sum_n \left[ \mathbf{x}_n \mathbf{x}_n^\top - \Lambda \boldsymbol{\mu}_n \mathbf{x}_n^\top - \mathbf{x}_n \boldsymbol{\mu}_n^\top \Lambda^\top + \Lambda (\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top + \Sigma) \Lambda^\top \right]$$

$$\hat{\Psi} = \textcolor{red}{\Lambda \Sigma \Lambda^\top} + \frac{1}{N} \sum_n (\mathbf{x}_n - \Lambda \boldsymbol{\mu}_n)(\mathbf{x}_n - \Lambda \boldsymbol{\mu}_n)^\top \quad (\text{squared residuals})$$

Note: we should actually only take derivatives w.r.t.  $\Psi_{dd}$  since  $\Psi$  is diagonal.

As  $\Sigma \rightarrow 0$  these become the equations for ML linear regression

## Poisson models



The Poisson factor model is easy to write down:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$p(\mathbf{x}|\mathbf{y}) = \prod_d \text{Poisson}\left(f\left(\sum_k \Lambda_{dk} y_k\right)\right) \quad \leftarrow \text{note } f() \text{ needed to ensure mean} > 0$$

...but the marginal distribution on  $\mathbf{x}$  is not.

Evaluation of likelihood and ML parameter estimates are intractable:

- ▶ Optimise jointly over  $\mathbf{y}_{(n)}$  and  $\Lambda$ : Exponential Family PCA. Can be convexified (for *canonical* and some other  $f$ ) and dimensionality found automatically using “nuclear norm” or trace-norm penalty and ADMM.
- ▶ Approximate learning: sampling, “Laplace” approximation, “variational” approximations, “expectation propagation”.

## Dynamics

- ▶ Slow features analysis: SFA
- ▶ [Noise (and slowness)] Gaussian Process Factor Analysis : GPFA
- ▶ [Markov dynamics] Linear Gaussian State-Space Models: LGSSM  
also called (Hidden) Linear Dynamical Systems models: LDS.
  - ▶ related to the Kalman Filter
  - ▶ a particular 0 noise limit → SFA.
  - ▶ consistent spectral learning [Subspace Identification: SSID] possible, but inefficient.
- ▶ Poisson noise: PLDS
  - ▶ EM intractable – requires approximation.
  - ▶ SSID can be adapted exactly.

## Gaussian process latents

$$\mathbf{y}(t) \sim \mathcal{GP} [\mu(t); K_\theta(t, t')]$$

state model

$$\mathbf{x}(t) \sim \text{Dist}[f(\mathbf{y}(t))]$$

observation model

$\mathcal{GP}$  is a Gaussian process: this implies that any finite set of measurements at fixed times is jointly normal.

- ▶ Includes linear-Gaussian dynamical systems (LDS).

$$\mathbf{y}_t \sim \mathcal{N} (\mathbf{A}\mathbf{y}_{t-1}, Q)$$

- ▶ Allows generalisation to non-(first-order-)Markov systems.

## Gaussian process dynamics

$$\mathbf{y}(t) \sim \mathcal{GP} [\mu(t); K_{\theta}(t, t')]$$

$$\mathbf{x}(t) \sim \text{Dist}[f(\mathbf{y}(t))]$$

- ▶  $K_{\theta}(t, t')$  gives the covariance between values of  $\mathbf{y}(t)$  and  $\mathbf{y}(t')$ .
- ▶ Parameterised by covariance. LDS (or auto-regressive models) are parameterised by precision (inverse covariance).
- ▶ Easier to specify priors with interesting properties:

– LDS:

$$K(t, t') \propto a^{|t-t'|}$$

– Smooth:

$$K(t, t') \propto \exp(-(t - t')^2 / 2\lambda)$$

– Oscillatory:

$$K(t, t') \propto \sin(2\pi\omega(t - t'))$$

– Stationary “Brownian”:

$$K(t, t') \propto [1 - |t - t'|/\lambda]^+$$

- ▶ Inference naively  $O(T^3)$  instead of  $O(T)$ .
  - ▶ Numerical methods based on regularities in matrices.
  - ▶ Sparsifying methods select (or create) subset of data with similar predictive power.

## Link functions

$$\mathbf{y}(t) \sim \mathcal{GP} [\mu(t); K_\theta(t, t')]$$

$$\mathbf{x}(t) \sim Dist[\mathbf{f}(\mathbf{y}(t))]$$

$f$  maps the latent GP values to (mean) intensity.

- ▶ Nonlinear
  - ▶ Exponential – **danger**: emphasises variability at high values.
  - ▶ Threshold-linear or soft-threshold.
- ▶ Linear
  - ▶ Requires observation model tolerant of negative values.
  - ▶ Alternatively, can use a truncated prior.
    - ▶ Requires approximation (but so does non-linearity).
    - ▶ Posterior often not far from Gaussian (multi-d truncation – draws are surprisingly smooth).
    - ▶ EP can be powerful approximation technique.

## Observation models

$$\mathbf{y}(t) \sim \mathcal{GP} [\mu(t); K_\theta(t, t')]$$

$$\mathbf{x}(t) \sim \textcolor{red}{Dist}[f(\mathbf{y}(t))]$$

- ▶ Point process (continuous time)
  - ▶ Rescaled renewal process. (next)
  - ▶ Inhomogeneous Markov-interval.

$$\lambda(t) = f(\mathbf{y}(t), s_{last}) \quad (\text{often } = f(\mathbf{y}(t)) \cdot h(s_{last}))$$

- ▶ GLM-like sum.

$$\lambda(t) = f\left(\mathbf{y}(t) + \sum_i \alpha_i h(s_i)\right)$$

- ▶ Spike count (discrete time)
  - ▶ Poisson counts.
  - ▶ (Square-rooted) Gaussian counts.

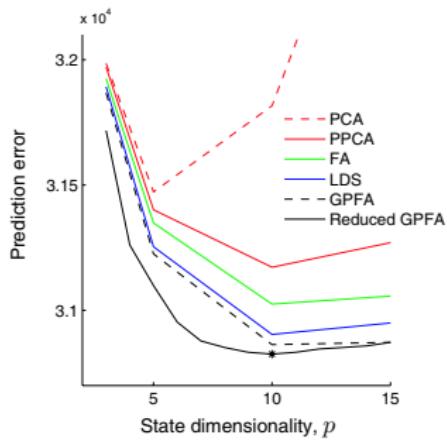
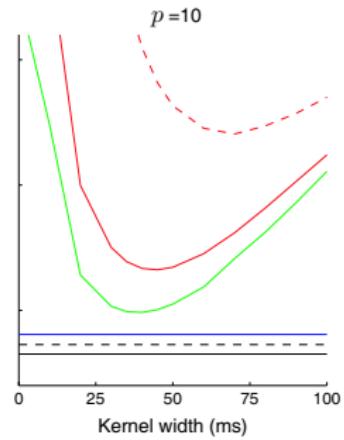
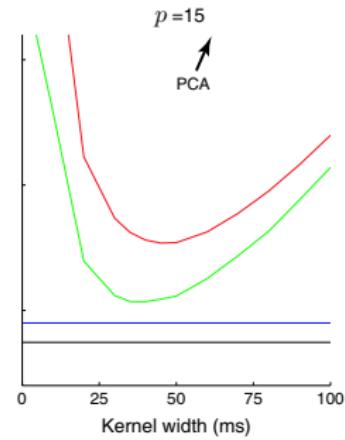
Spike train binned (10 – 20 ms) to yield spike counts.

$$x_i(t) \sim \mathcal{GP}[\mathbf{0}; K_i]$$

$$K_i(t_1, t_2) = (1 - \sigma_n^2) \exp\left(-\frac{(t_1 - t_2)^2}{2\tau_i^2}\right) + \sigma_n^2 \delta_{t_1, t_2}$$

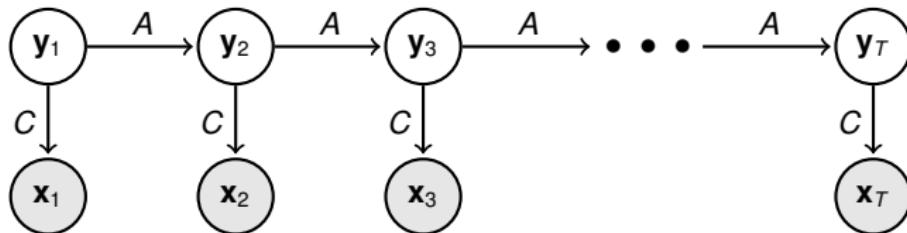
$$\mathbf{x}(t) | \mathbf{y}(t) \sim \mathcal{N}(C\mathbf{y}(t) + \mathbf{d}, R)$$

- ▶ Spike counts may be square-rooted to stabilise variance of (and Gaussianise) Poisson counts
- ▶ The model is jointly Gaussian! Exact inference and learning is possible using Factor-Analysis-like methods.

**A****B****C**

## Learning explicit dynamics

State space models.

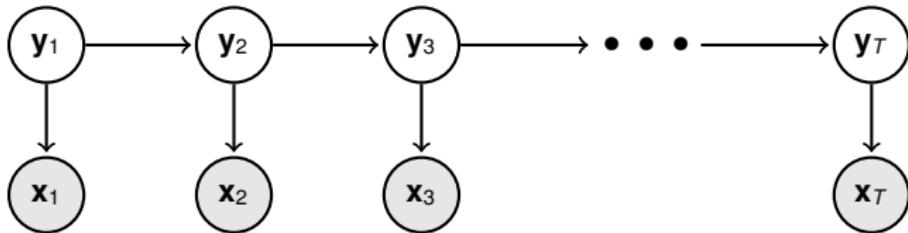


$$\mathbf{y}_t | \mathbf{y}_{t-1} \sim \mathcal{N}(\mathbf{A}\mathbf{y}_{t-1}, Q)$$

$$\mathbf{x}_t | \mathbf{y}_t \sim \mathcal{N}(C\mathbf{y}_t, R)$$

- ▶ Dynamics in latent space are self-contained.
- ▶ An **innovations** process introduces stochasticity, and allows inference and learning to compensate for model mismatch.
- ▶ Poisson, or other point-process observation models are not easy to handle. (But see Smith & Brown 2003, Yu et al. 2006, Macke et al. 2011, Buesing et al. 2012).

## Latent-chain models



Joint probability factorizes:

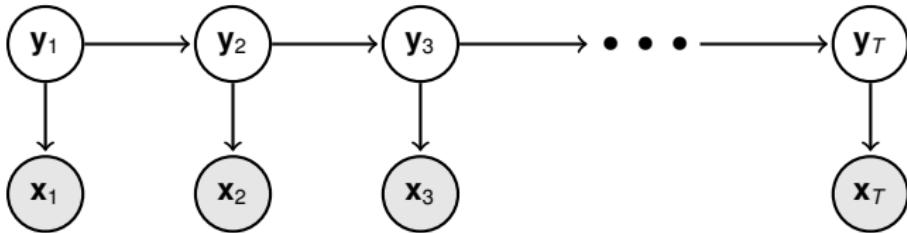
$$P(\mathbf{y}_{1:T}, \mathbf{x}_{1:T}) = P(\mathbf{y}_1)P(\mathbf{x}_1|\mathbf{y}_1)\prod_{t=2}^T P(\mathbf{y}_t|\mathbf{y}_{t-1})P(\mathbf{x}_t|\mathbf{y}_t)$$

where  $\mathbf{y}_t$  and  $\mathbf{x}_t$  are both real-valued vectors, and  $\square_{1:T} \equiv \square_1, \dots, \square_T$ .

Two frequently-used tractable models:

- ▶ Linear-Gaussian state-space models
- ▶ Hidden Markov models

## Linear-Gaussian state-space models (SSMs)



In a [linear Gaussian SSM](#) all conditional distributions are linear and Gaussian:

Output equation:

$$\mathbf{x}_t = \mathbf{C}\mathbf{y}_t + \mathbf{v}_t$$

State dynamics equation:

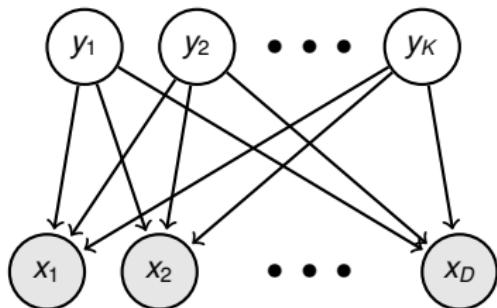
$$\mathbf{y}_t = \mathbf{A}\mathbf{y}_{t-1} + \mathbf{w}_t$$

where  $\mathbf{v}_t$  and  $\mathbf{w}_t$  are uncorrelated zero-mean multivariate Gaussian noise vectors.

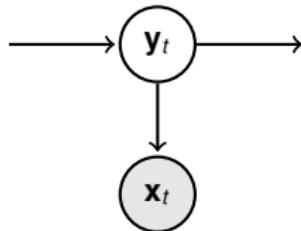
Also assume  $\mathbf{y}_1$  is multivariate Gaussian. The joint distribution over all variables  $\mathbf{x}_{1:T}, \mathbf{y}_{1:T}$  is (one big) multivariate Gaussian.

These models are also known as stochastic [linear dynamical systems](#), [Kalman filter models](#).

## From factor analysis to state space models



Factor analysis:  $x_i = \sum_{j=1}^K \Lambda_{ij} y_j + \epsilon_i$

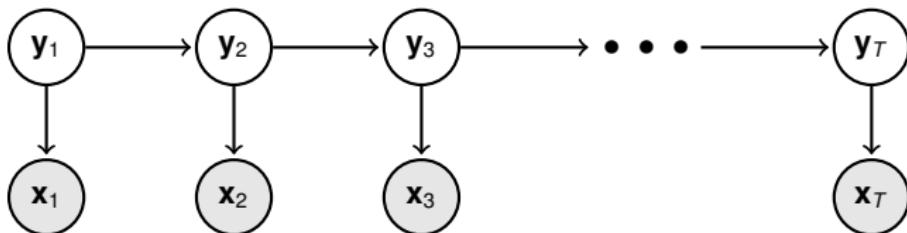


SSM output:  $x_{t,i} = \sum_{j=1}^K C_{ij} y_{t,j} + v_i$ .

### Interpretation 1:

- ▶ Observations confined near low-dimensional subspace (as in FA/PCA).
- ▶ Successive observations are generated from correlated points in the latent space.
- ▶ However:
  - ▶ FA requires  $K < D$  and  $\Psi$  diagonal; SSMs may have  $K \geq D$  and *arbitrary* output noise. Why?
  - ▶ Thus ML estimates of subspace by FA and SSM may differ.

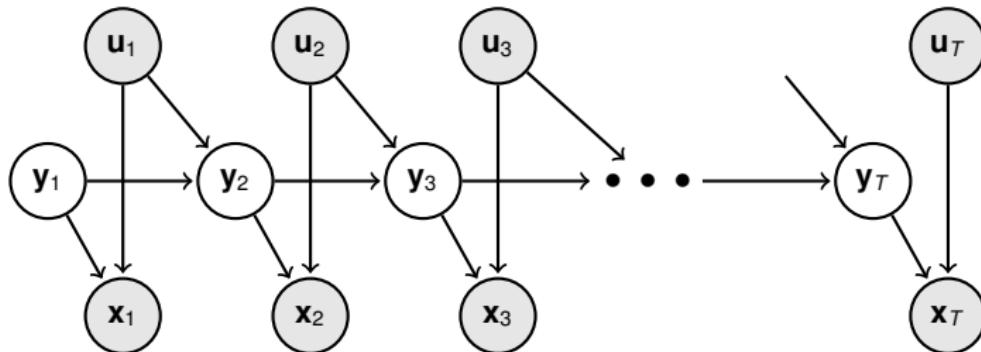
## Linear dynamical systems



### Interpretation 2:

- ▶ Markov chain with **linear dynamics**  $\mathbf{y}_t = A\mathbf{y}_{t-1} + \dots$
- ▶ ... perturbed by Gaussian **innovations** noise – may describe stochasticity, unknown control, or model mismatch.
- ▶ Observations are a linear projection of the dynamical state, with additive iid Gaussian noise.
- ▶ Note:
  - ▶ Dynamical process ( $\mathbf{y}_t$ ) may be higher dimensional than the observations ( $\mathbf{x}_t$ ).
  - ▶ Observations **do not** form a Markov chain – longer-scale dependence reflects/reveals latent dynamics.

## State Space Models with Control Inputs



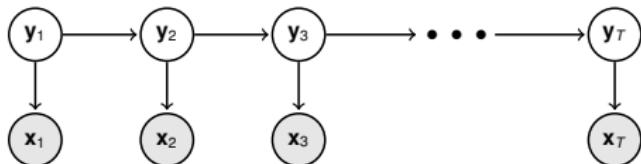
State space models can be used to model the input–output behaviour of controlled systems. The observed variables are divided into **inputs ( $u_t$ )** and **outputs ( $x_t$ )**.

**State dynamics equation:**  $\mathbf{y}_t = A\mathbf{y}_{t-1} + B\mathbf{u}_{t-1} + \mathbf{w}_t$ .

**Output equation:**  $\mathbf{x}_t = C\mathbf{y}_t + D\mathbf{u}_t + \mathbf{v}_t$ .

Note that we can have many variants, e.g.  $\mathbf{y}_t = A\mathbf{y}_{t-1} + B\mathbf{u}_t + \mathbf{w}_t$  or even  $\mathbf{y}_t = A\mathbf{y}_{t-1} + B\mathbf{x}_{t-1} + \mathbf{w}_t$ .

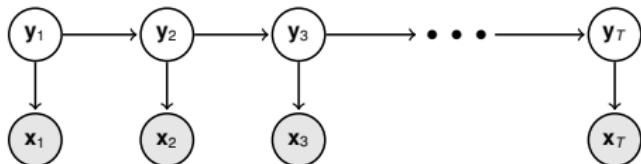
## Chain models: ML Learning with EM



$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\mu_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

The structure of learning and inference is dictated by the factored structure.

## Chain models: ML Learning with EM

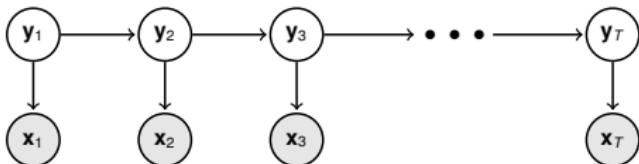


$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

The structure of learning and inference is dictated by the factored structure.

$$P(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{y}_1, \dots, \mathbf{y}_T) = P(\mathbf{y}_1) \prod_{t=2}^T P(\mathbf{y}_t | \mathbf{y}_{t-1}) \prod_{t=1}^T P(\mathbf{x}_t | \mathbf{y}_t)$$

## Chain models: ML Learning with EM



$$\begin{aligned} \mathbf{y}_1 &\sim \mathcal{N}(\mu_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R) \end{aligned}$$

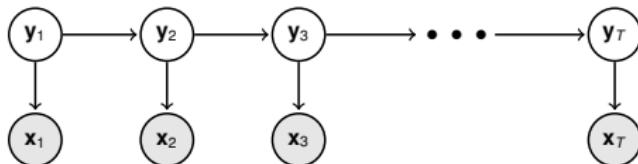
The structure of learning and inference is dictated by the factored structure.

$$P(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{y}_1, \dots, \mathbf{y}_T) = P(\mathbf{y}_1) \prod_{t=2}^T P(\mathbf{y}_t | \mathbf{y}_{t-1}) \prod_{t=1}^T P(\mathbf{x}_t | \mathbf{y}_t)$$

**Learning (M-step):**

$$\begin{aligned} \text{argmax } & \langle \log P(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{y}_1, \dots, \mathbf{y}_T) \rangle_{q(\mathbf{y}_1, \dots, \mathbf{y}_T)} = \\ & \text{argmax } \left[ \langle \log P(\mathbf{y}_1) \rangle_{q(\mathbf{y}_1)} + \sum_{t=2}^T \langle \log P(\mathbf{y}_t | \mathbf{y}_{t-1}) \rangle_{q(\mathbf{y}_t, \mathbf{y}_{t-1})} + \sum_{t=1}^T \langle \log P(\mathbf{x}_t | \mathbf{y}_t) \rangle_{q(\mathbf{y}_t)} \right] \end{aligned}$$

## Chain models: ML Learning with EM



$$\begin{aligned} \mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R) \end{aligned}$$

The structure of learning and inference is dictated by the factored structure.

$$P(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{y}_1, \dots, \mathbf{y}_T) = P(\mathbf{y}_1) \prod_{t=2}^T P(\mathbf{y}_t | \mathbf{y}_{t-1}) \prod_{t=1}^T P(\mathbf{x}_t | \mathbf{y}_t)$$

**Learning (M-step):**

$$\begin{aligned} \text{argmax } &\langle \log P(\mathbf{x}_1, \dots, \mathbf{x}_T, \mathbf{y}_1, \dots, \mathbf{y}_T) \rangle_{q(\mathbf{y}_1, \dots, \mathbf{y}_T)} = \\ &\text{argmax } \left[ \langle \log P(\mathbf{y}_1) \rangle_{q(\mathbf{y}_1)} + \sum_{t=2}^T \langle \log P(\mathbf{y}_t | \mathbf{y}_{t-1}) \rangle_{q(\mathbf{y}_t, \mathbf{y}_{t-1})} + \sum_{t=1}^T \langle \log P(\mathbf{x}_t | \mathbf{y}_t) \rangle_{q(\mathbf{y}_t)} \right] \end{aligned}$$

So the expectations needed (in E-step) are derived from singleton and pairwise marginals.

## Chain models: Inference

Three general inference problems:

Filtering:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t)$

Smoothing:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T)$  (also  $P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_1, \dots, \mathbf{x}_T)$  for learning)

Prediction:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_{t-\Delta t})$

## Chain models: Inference

Three general inference problems:

Filtering:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t)$

Smoothing:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T)$  (also  $P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_1, \dots, \mathbf{x}_T)$  for learning)

Prediction:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_{t-\Delta t})$

Naively, these marginal posteriors seem to require very large integrals (or sums)

$$P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t) = \int \dots \int d\mathbf{y}_1 \dots d\mathbf{y}_{t-1} P(\mathbf{y}_1, \dots, \mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t)$$

## Chain models: Inference

Three general inference problems:

Filtering:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t)$

Smoothing:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T)$  (also  $P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_1, \dots, \mathbf{x}_T)$  for learning)

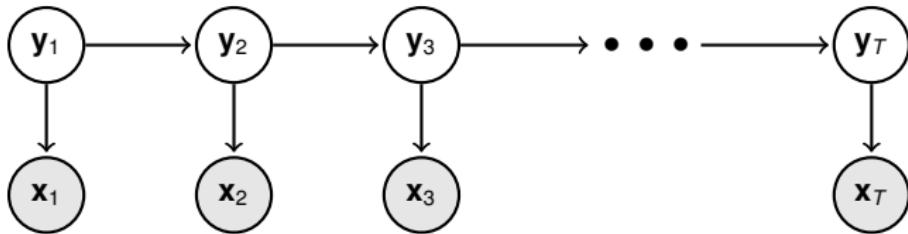
Prediction:  $P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_{t-\Delta t})$

Naively, these marginal posteriors seem to require very large integrals (or sums)

$$P(\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t) = \int \dots \int d\mathbf{y}_1 \dots d\mathbf{y}_{t-1} P(\mathbf{y}_1, \dots, \mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_t)$$

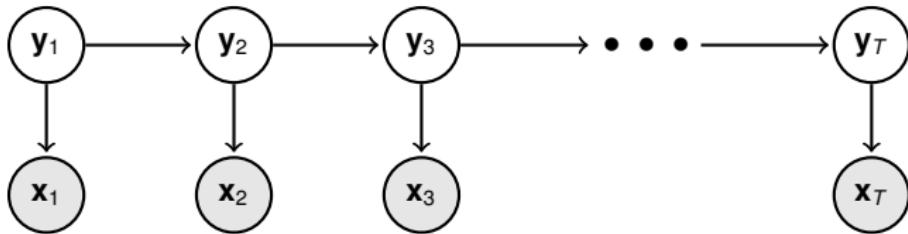
but again the factored structure of the distributions will help us. The algorithms rely on a form of temporal updating or message passing.

## Probability updating: “Bayesian filtering”



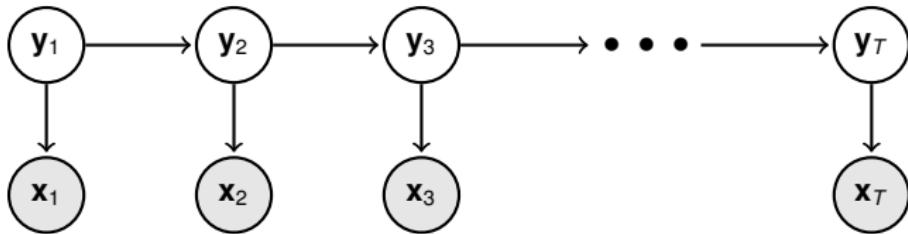
$$P(\mathbf{y}_t | \mathbf{x}_{1:t}) = \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_{1:t}) d\mathbf{y}_{t-1}$$

## Probability updating: “Bayesian filtering”



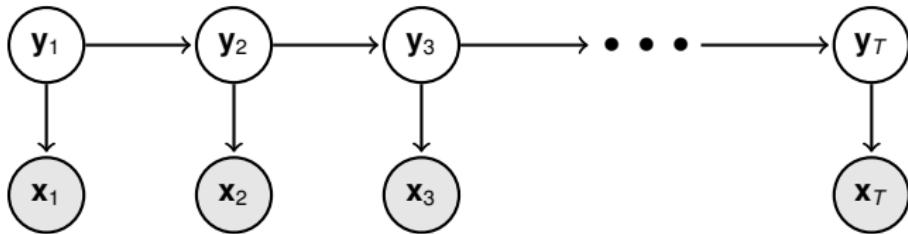
$$P(\mathbf{y}_t | \mathbf{x}_{1:t}) = \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_t, \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1}$$

## Probability updating: “Bayesian filtering”



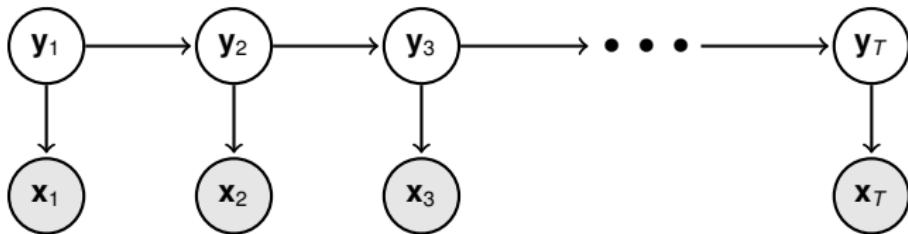
$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:t}) &= \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_t, \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int \frac{P(\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_{1:t-1})}{P(\mathbf{x}_t | \mathbf{x}_{1:t-1})} d\mathbf{y}_{t-1} \end{aligned}$$

## Probability updating: “Bayesian filtering”



$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:t}) &= \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_t, \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int \frac{P(\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_{1:t-1})}{P(\mathbf{x}_t | \mathbf{x}_{1:t-1})} d\mathbf{y}_{t-1} \\ &\propto \int P(\mathbf{x}_t | \mathbf{y}_t, \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \end{aligned}$$

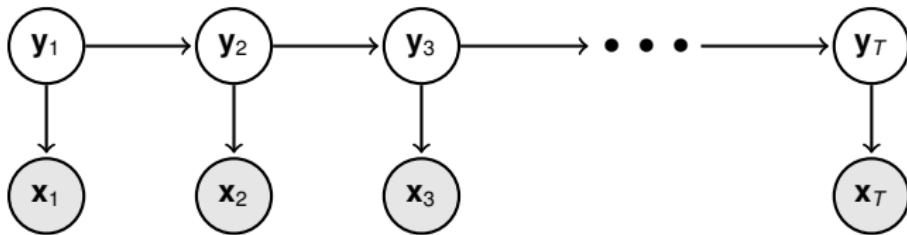
## Probability updating: “Bayesian filtering”



$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:t}) &= \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_t, \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int \frac{P(\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_{1:t-1})}{P(\mathbf{x}_t | \mathbf{x}_{1:t-1})} d\mathbf{y}_{t-1} \\ &\propto \int P(\mathbf{x}_t | \mathbf{y}_t, \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int P(\mathbf{x}_t | \mathbf{y}_t) P(\mathbf{y}_t | \mathbf{y}_{t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \end{aligned}$$

Markov property

## Probability updating: “Bayesian filtering”

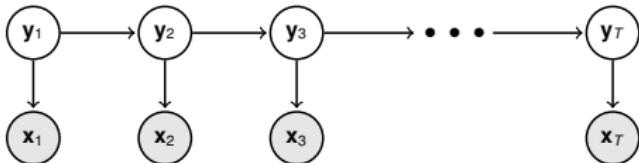


$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:t}) &= \int P(\mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_t, \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int \frac{P(\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{t-1} | \mathbf{x}_{1:t-1})}{P(\mathbf{x}_t | \mathbf{x}_{1:t-1})} d\mathbf{y}_{t-1} \\ &\propto \int P(\mathbf{x}_t | \mathbf{y}_t, \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{x}_{1:t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \\ &= \int P(\mathbf{x}_t | \mathbf{y}_t) P(\mathbf{y}_t | \mathbf{y}_{t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{y}_{t-1} \end{aligned}$$

Markov property

This is a **forward recursion** based on Bayes rule.

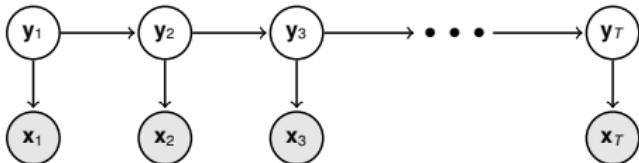
## The LGSSM: Kalman Filtering



$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

For the SSM, the sums become integrals.

## The LGSSM: Kalman Filtering

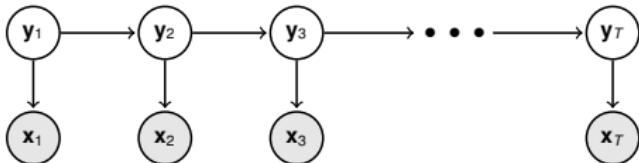


$$\begin{aligned}y_1 &\sim \mathcal{N}(\mu_0, Q_0) \\y_t | y_{t-1} &\sim \mathcal{N}(A y_{t-1}, Q) \\x_t | y_t &\sim \mathcal{N}(C y_t, R)\end{aligned}$$

For the SSM, the sums become integrals. Let  $\hat{y}_1^0 = \mu_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(y_1 | x_1) = \mathcal{N}(\hat{y}_1^0 + K_1(x_1 - C\hat{y}_1^0), \hat{V}_1^0 - K_1 C \hat{V}_1^0)$$

## The LGSSM: Kalman Filtering



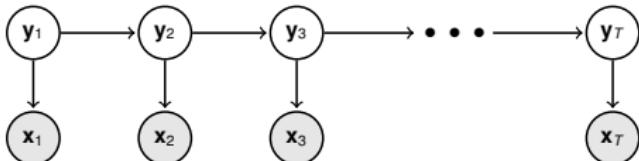
$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

For the SSM, the sums become integrals. Let  $\hat{\mathbf{y}}_1^0 = \boldsymbol{\mu}_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(\mathbf{y}_1 | \mathbf{x}_1) = \mathcal{N}(\hat{\mathbf{y}}_1^0 + K_1(\mathbf{x}_1 - C\hat{\mathbf{y}}_1^0), \hat{V}_1^0 - K_1 C \hat{V}_1^0)$$

$$K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

## The LGSSM: Kalman Filtering

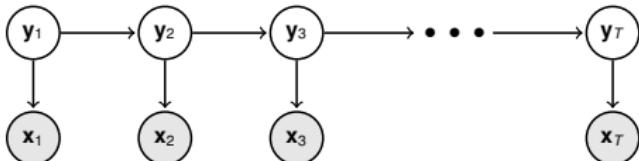


$$\begin{aligned}y_1 &\sim \mathcal{N}(\mu_0, Q_0) \\y_t | y_{t-1} &\sim \mathcal{N}(A y_{t-1}, Q) \\x_t | y_t &\sim \mathcal{N}(C y_t, R)\end{aligned}$$

For the SSM, the sums become integrals. Let  $\hat{y}_1^0 = \mu_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(y_1 | x_1) = \mathcal{N}\left(\underbrace{\hat{y}_1^0 + K_1(x_1 - C\hat{y}_1^0)}_{\hat{y}_1^1}, \underbrace{\hat{V}_1^0 - K_1 C \hat{V}_1^0}_{\hat{V}_1^1}\right) \quad K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

## The LGSSM: Kalman Filtering



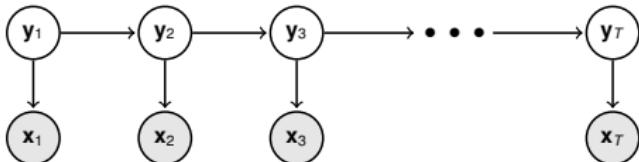
$$\begin{aligned}y_1 &\sim \mathcal{N}(\mu_0, Q_0) \\y_t | y_{t-1} &\sim \mathcal{N}(A y_{t-1}, Q) \\x_t | y_t &\sim \mathcal{N}(C y_t, R)\end{aligned}$$

For the SSM, the sums become integrals. Let  $\hat{y}_1^0 = \mu_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(y_1 | x_1) = \mathcal{N}\left(\underbrace{\hat{y}_1^0 + K_1(x_1 - C\hat{y}_1^0)}_{\hat{y}_1^1}, \underbrace{\hat{V}_1^0 - K_1 C \hat{V}_1^0}_{\hat{V}_1^1}\right) \quad K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

In general, we define  $\hat{y}_t^T \equiv E[y_t | x_1, \dots, x_T]$  and  $\hat{V}_t^T \equiv V[y_t | x_1, \dots, x_T]$ .

## The LGSSM: Kalman Filtering



$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

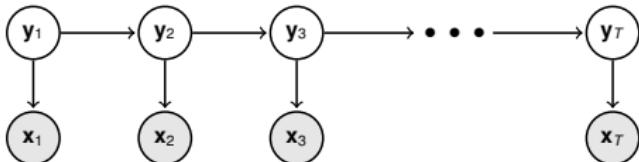
For the SSM, the sums become integrals. Let  $\hat{\mathbf{y}}_1^0 = \boldsymbol{\mu}_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(\mathbf{y}_1 | \mathbf{x}_1) = \mathcal{N}(\underbrace{\hat{\mathbf{y}}_1^0 + K_1(\mathbf{x}_1 - C\hat{\mathbf{y}}_1^0)}_{\hat{\mathbf{y}}_1^1}, \underbrace{\hat{V}_1^0 - K_1 C \hat{V}_1^0}_{\hat{V}_1^1}) \quad K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

In general, we define  $\hat{\mathbf{y}}_t^T \equiv E[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$  and  $\hat{V}_t^T \equiv V[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$ . Then,

$$P(\mathbf{y}_t | \mathbf{x}_{1:t-1}) = \int d\mathbf{y}_{t-1} P(\mathbf{y}_t | \mathbf{y}_{t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1})$$

## The LGSSM: Kalman Filtering



$$\begin{aligned}\mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R)\end{aligned}$$

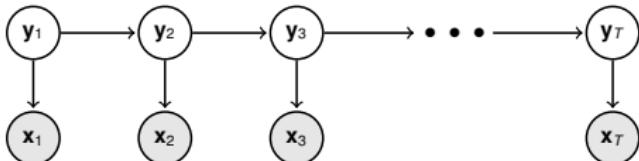
For the SSM, the sums become integrals. Let  $\hat{\mathbf{y}}_1^0 = \boldsymbol{\mu}_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(\mathbf{y}_1 | \mathbf{x}_1) = \mathcal{N}(\underbrace{\hat{\mathbf{y}}_1^0 + K_1(\mathbf{x}_1 - C\hat{\mathbf{y}}_1^0)}_{\hat{\mathbf{y}}_1^1}, \underbrace{\hat{V}_1^0 - K_1 C \hat{V}_1^0}_{\hat{V}_1^1}) \quad K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

In general, we define  $\hat{\mathbf{y}}_t^T \equiv E[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$  and  $\hat{V}_t^T \equiv V[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$ . Then,

$$P(\mathbf{y}_t | \mathbf{x}_{1:t-1}) = \int d\mathbf{y}_{t-1} P(\mathbf{y}_t | \mathbf{y}_{t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) = \mathcal{N}\left(\underbrace{A\hat{\mathbf{y}}_{t-1}^{t-1}}_{\hat{\mathbf{y}}_t^{t-1}}, \underbrace{A\hat{V}_{t-1}^{t-1} A^\top + Q}_{\hat{V}_t^{t-1}}\right)$$

## The LGSSM: Kalman Filtering



$$\begin{aligned} \mathbf{y}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_0, Q_0) \\ \mathbf{y}_t | \mathbf{y}_{t-1} &\sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \\ \mathbf{x}_t | \mathbf{y}_t &\sim \mathcal{N}(C\mathbf{y}_t, R) \end{aligned}$$

For the SSM, the sums become integrals. Let  $\hat{\mathbf{y}}_1^0 = \boldsymbol{\mu}_0$  and  $\hat{V}_1^0 = Q_0$ ; then (cf. FA)

$$P(\mathbf{y}_1 | \mathbf{x}_1) = \underbrace{\mathcal{N}(\hat{\mathbf{y}}_1^0 + K_1(\mathbf{x}_1 - C\hat{\mathbf{y}}_1^0), \hat{V}_1^0 - K_1 C \hat{V}_1^0)}_{\hat{\mathbf{y}}_1^1} \quad K_1 = \hat{V}_1^0 C^\top (C \hat{V}_1^0 C^\top + R)^{-1}$$

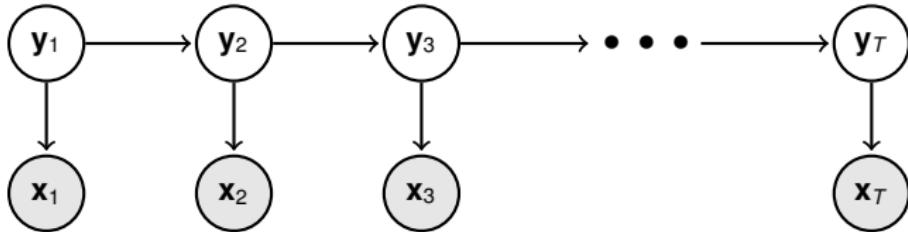
In general, we define  $\hat{\mathbf{y}}_t^T \equiv E[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$  and  $\hat{V}_t^T \equiv V[\mathbf{y}_t | \mathbf{x}_1, \dots, \mathbf{x}_T]$ . Then,

$$P(\mathbf{y}_t | \mathbf{x}_{1:t-1}) = \int d\mathbf{y}_{t-1} P(\mathbf{y}_t | \mathbf{y}_{t-1}) P(\mathbf{y}_{t-1} | \mathbf{x}_{1:t-1}) = \mathcal{N}\left(\underbrace{A\hat{\mathbf{y}}_{t-1}^{t-1}}_{\hat{\mathbf{y}}_t^{t-1}}, \underbrace{A\hat{V}_{t-1}^{t-1} A^\top + Q}_{\hat{V}_t^{t-1}}\right)$$

$$P(\mathbf{y}_t | \mathbf{x}_{1:t}) = \mathcal{N}\left(\underbrace{\hat{\mathbf{y}}_t^{t-1} + K_t(\mathbf{x}_t - C\hat{\mathbf{y}}_t^{t-1})}_{\hat{\mathbf{y}}_t^t}, \underbrace{\hat{V}_t^{t-1} - K_t C \hat{V}_t^{t-1}}_{\hat{V}_t^t}\right)$$

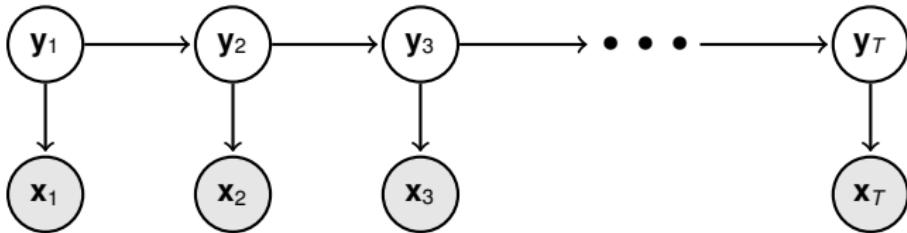
$$K_t = \underbrace{\hat{V}_t^{t-1} C^\top (C \hat{V}_t^{t-1} C^\top + R)^{-1}}_{\text{Kalman gain}}$$

## The marginal posterior: “Bayesian smoothing”



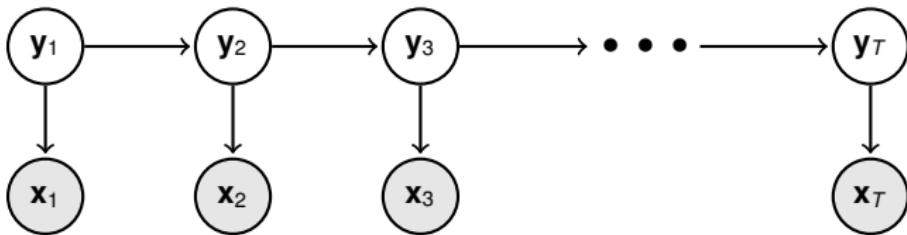
$$P(\mathbf{y}_t | \mathbf{x}_{1:T})$$

## The marginal posterior: “Bayesian smoothing”



$$P(\mathbf{y}_t | \mathbf{x}_{1:T}) = \frac{P(\mathbf{y}_t, \mathbf{x}_{t+1:T} | \mathbf{x}_{1:t})}{P(\mathbf{x}_{t+1:T} | \mathbf{x}_{1:t})}$$

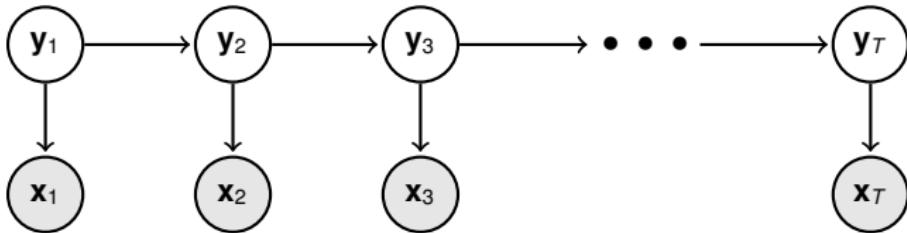
## The marginal posterior: “Bayesian smoothing”



$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:T}) &= \frac{P(\mathbf{y}_t, \mathbf{x}_{t+1:T} | \mathbf{x}_{1:t})}{P(\mathbf{x}_{t+1:T} | \mathbf{x}_{1:t})} \\ &= \frac{P(\mathbf{x}_{t+1:T} | \mathbf{y}_t) P(\mathbf{y}_t | \mathbf{x}_{1:t})}{P(\mathbf{x}_{t+1:T} | \mathbf{x}_{1:t})} \end{aligned}$$

The marginal combines a **backward message** with the **forward message** found by filtering.

## The LGSSM: Kalman smoothing



We use a slightly different decomposition:

$$\begin{aligned} P(\mathbf{y}_t | \mathbf{x}_{1:T}) &= \int P(\mathbf{y}_t, \mathbf{y}_{t+1} | \mathbf{x}_{1:T}) d\mathbf{y}_{t+1} \\ &= \int P(\mathbf{y}_t | \mathbf{y}_{t+1}, \mathbf{x}_{1:T}) P(\mathbf{y}_{t+1} | \mathbf{x}_{1:T}) d\mathbf{y}_{t+1} \\ &= \underset{\text{Markov property}}{\int} P(\mathbf{y}_t | \mathbf{y}_{t+1}, \mathbf{x}_{1:t}) P(\mathbf{y}_{t+1} | \mathbf{x}_{1:T}) d\mathbf{y}_{t+1} \end{aligned}$$

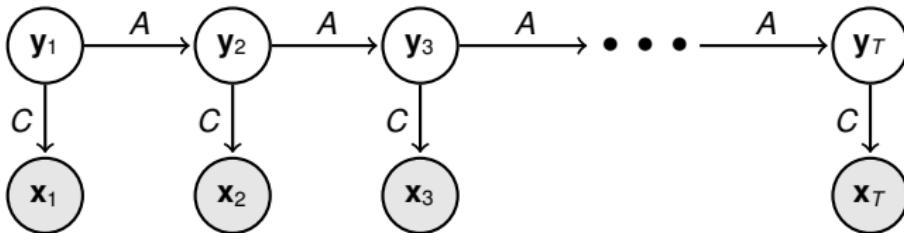
This gives the additional **backward recursion**:

$$\mathbf{J}_t = \hat{V}_t^t A^T (\hat{V}_{t+1}^t)^{-1}$$

$$\hat{\mathbf{y}}_t^T = \hat{\mathbf{y}}_t^t + \mathbf{J}_t (\hat{\mathbf{y}}_{t+1}^T - A \hat{\mathbf{y}}_t^t)$$

$$\hat{V}_t^T = \hat{V}_t^t + \mathbf{J}_t (\hat{V}_{t+1}^T - \hat{V}_{t+1}^t) \mathbf{J}_t^T$$

## ML Learning for SSMs using batch EM



Parameters:  $\theta = \{\mu_0, Q_0, A, Q, C, R\}$

Free energy:

$$\mathcal{F}(q, \theta) = \int d\mathbf{y}_{1:T} q(\mathbf{y}_{1:T}) (\log P(x_{1:T}, \mathbf{y}_{1:T} | \theta) - \log q(\mathbf{y}_{1:T}))$$

**E-step:** Maximise  $\mathcal{F}$  w.r.t.  $q$  with  $\theta$  fixed:

$$q^*(\mathbf{y}) = p(\mathbf{y} | \mathbf{x}, \theta)$$

This can be achieved with a two-state extension of the Kalman smoother.

**M-step:** Maximize  $\mathcal{F}$  w.r.t.  $\theta$  with  $q$  fixed.

This boils down to solving a few weighted least squares problems, since all the variables in:

$$p(\mathbf{y}, \mathbf{x} | \theta) = p(\mathbf{y}_1)p(\mathbf{x}_1 | \mathbf{y}_1) \prod_{t=2}^T p(\mathbf{y}_t | \mathbf{y}_{t-1})p(\mathbf{x}_t | \mathbf{y}_t)$$

form a multivariate Gaussian.

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$C_{\text{new}} = \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q$$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$C_{\text{new}} = \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q$$

$$= \underset{C}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right\rangle_q + \text{const}$$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$\begin{aligned} C_{\text{new}} &= \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q \\ &= \underset{C}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{C}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{x}_t^T R^{-1} \mathbf{x}_t - 2\mathbf{x}_t^T R^{-1} C \langle \mathbf{y}_t \rangle + \langle \mathbf{y}_t^T C^T R^{-1} C \mathbf{y}_t \rangle \right\} \end{aligned}$$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$\begin{aligned} C_{\text{new}} &= \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q \\ &= \underset{C}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{C}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{x}_t^T R^{-1} \mathbf{x}_t - 2 \mathbf{x}_t^T R^{-1} C \langle \mathbf{y}_t \rangle + \langle \mathbf{y}_t^T C^T R^{-1} C \mathbf{y}_t \rangle \right\} \\ &= \underset{C}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ C \sum_t \langle \mathbf{y}_t \rangle \mathbf{x}_t^T R^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ C^T R^{-1} C \left\langle \sum_t \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$\begin{aligned} C_{\text{new}} &= \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q \\ &= \underset{C}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{C}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{x}_t^T R^{-1} \mathbf{x}_t - 2 \mathbf{x}_t^T R^{-1} C \langle \mathbf{y}_t \rangle + \langle \mathbf{y}_t^T C^T R^{-1} C \mathbf{y}_t \rangle \right\} \\ &= \underset{C}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ C \sum_t \langle \mathbf{y}_t \rangle \mathbf{x}_t^T R^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ C^T R^{-1} C \left\langle \sum_t \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

using  $\frac{\partial \operatorname{Tr}[AB]}{\partial A} = B^T$ , we have  $\frac{\partial \{\cdot\}}{\partial C} = R^{-1} \sum_t \mathbf{x}_t \langle \mathbf{y}_t \rangle^T - R^{-1} C \left\langle \sum_t \mathbf{y}_t \mathbf{y}_t^T \right\rangle$

## The M step for $C$

$$p(\mathbf{x}_t | \mathbf{y}_t) \propto \exp \left[ -\frac{1}{2} (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right] \Rightarrow$$

$$\begin{aligned} C_{\text{new}} &= \underset{C}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{x}_t | \mathbf{y}_t) \right\rangle_q \\ &= \underset{C}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{x}_t - C\mathbf{y}_t)^T R^{-1} (\mathbf{x}_t - C\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{C}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{x}_t^T R^{-1} \mathbf{x}_t - 2 \mathbf{x}_t^T R^{-1} C \langle \mathbf{y}_t \rangle + \langle \mathbf{y}_t^T C^T R^{-1} C \mathbf{y}_t \rangle \right\} \\ &= \underset{C}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ C \sum_t \langle \mathbf{y}_t \rangle \mathbf{x}_t^T R^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ C^T R^{-1} C \left\langle \sum_t \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

using  $\frac{\partial \operatorname{Tr}[AB]}{\partial A} = B^T$ , we have  $\frac{\partial \{\cdot\}}{\partial C} = R^{-1} \sum_t \mathbf{x}_t \langle \mathbf{y}_t \rangle^T - R^{-1} C \left\langle \sum_t \mathbf{y}_t \mathbf{y}_t^T \right\rangle$

$$\Rightarrow C_{\text{new}} = \left( \sum_t \mathbf{x}_t \langle \mathbf{y}_t \rangle^T \right) \left( \sum_t \langle \mathbf{y}_t \mathbf{y}_t^T \rangle \right)^{-1}$$

Notice that this is exactly the *same equation* as in factor analysis and linear regression!

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$A_{\text{new}} = \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q$$

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$\begin{aligned} A_{\text{new}} &= \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q \\ &= \underset{A}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\rangle_q + \text{const} \end{aligned}$$

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$\begin{aligned} A_{\text{new}} &= \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q \\ &= \underset{A}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{A}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{y}_{t+1}^T Q^{-1} \mathbf{y}_{t+1} - 2 \left\langle \mathbf{y}_{t+1}^T Q^{-1} A \mathbf{y}_t \right\rangle + \left\langle \mathbf{y}_t^T A^T Q^{-1} A \mathbf{y}_t \right\rangle \right\} \end{aligned}$$

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$\begin{aligned} A_{\text{new}} &= \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q \\ &= \underset{A}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{A}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{y}_{t+1}^T Q^{-1} \mathbf{y}_{t+1} - 2 \left\langle \mathbf{y}_{t+1}^T Q^{-1} A \mathbf{y}_t \right\rangle + \left\langle \mathbf{y}_t^T A^T Q^{-1} A \mathbf{y}_t \right\rangle \right\} \\ &= \underset{A}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_{t+1}^T \right\rangle Q^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ A^T Q^{-1} A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

## The M step for A

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$\begin{aligned} A_{\text{new}} &= \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q \\ &= \underset{A}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{A}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{y}_{t+1}^T Q^{-1} \mathbf{y}_{t+1} - 2 \left\langle \mathbf{y}_{t+1}^T Q^{-1} A \mathbf{y}_t \right\rangle + \left\langle \mathbf{y}_t^T A^T Q^{-1} A \mathbf{y}_t \right\rangle \right\} \\ &= \underset{A}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_{t+1}^T \right\rangle Q^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ A^T Q^{-1} A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

using  $\frac{\partial \operatorname{Tr}[AB]}{\partial A} = B^T$ , we have  $\frac{\partial \{\cdot\}}{\partial A} = Q^{-1} \sum_t \left\langle \mathbf{y}_{t+1} \mathbf{y}_t^T \right\rangle - Q^{-1} A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle$

## The M step for $A$

$$p(\mathbf{y}_{t+1} | \mathbf{y}_t) \propto \exp \left\{ -\frac{1}{2} (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\} \Rightarrow$$

$$\begin{aligned} A_{\text{new}} &= \underset{A}{\operatorname{argmax}} \left\langle \sum_t \ln p(\mathbf{y}_{t+1} | \mathbf{y}_t) \right\rangle_q \\ &= \underset{A}{\operatorname{argmax}} \left\langle -\frac{1}{2} \sum_t (\mathbf{y}_{t+1} - A\mathbf{y}_t)^T Q^{-1} (\mathbf{y}_{t+1} - A\mathbf{y}_t) \right\rangle_q + \text{const} \\ &= \underset{A}{\operatorname{argmax}} \left\{ -\frac{1}{2} \sum_t \mathbf{y}_{t+1}^T Q^{-1} \mathbf{y}_{t+1} - 2 \left\langle \mathbf{y}_{t+1}^T Q^{-1} A \mathbf{y}_t \right\rangle + \left\langle \mathbf{y}_t^T A^T Q^{-1} A \mathbf{y}_t \right\rangle \right\} \\ &= \underset{A}{\operatorname{argmax}} \left\{ \operatorname{Tr} \left[ A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_{t+1}^T \right\rangle Q^{-1} \right] - \frac{1}{2} \operatorname{Tr} \left[ A^T Q^{-1} A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right] \right\} \end{aligned}$$

using  $\frac{\partial \operatorname{Tr}[AB]}{\partial A} = B^T$ , we have  $\frac{\partial \{\cdot\}}{\partial A} = Q^{-1} \sum_t \left\langle \mathbf{y}_{t+1} \mathbf{y}_t^T \right\rangle - Q^{-1} A \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle$

$$\Rightarrow A_{\text{new}} = \left( \sum_t \left\langle \mathbf{y}_{t+1} \mathbf{y}_t^T \right\rangle \right) \left( \sum_t \left\langle \mathbf{y}_t \mathbf{y}_t^T \right\rangle \right)^{-1}$$

This is still analogous to factor analysis and linear regression, with expected correlations.

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$P(\mathbf{y}) = \mathcal{N}(\mathbf{0}, I)$$

$$P(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\Lambda\mathbf{y}, \Psi)$$

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$\begin{aligned} P(\mathbf{y}) &= \mathcal{N}(\mathbf{0}, I) \\ P(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\Lambda\mathbf{y}, \Psi) \end{aligned} \quad \& \quad \begin{aligned} \tilde{\mathbf{y}} &= U\mathbf{y} \\ \tilde{\Lambda} &= \Lambda U^T \end{aligned}$$

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$\begin{aligned} P(\mathbf{y}) &= \mathcal{N}(\mathbf{0}, I) & \tilde{\mathbf{y}} = U\mathbf{y} & \Rightarrow P(\tilde{\mathbf{y}}) = \mathcal{N}\left(U\mathbf{0}, UIU^T\right) = \mathcal{N}(\mathbf{0}, I) \\ P(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\Lambda\mathbf{y}, \Psi) & \tilde{\Lambda} = \Lambda U^T & \Rightarrow P(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}\left(\Lambda U^T U\mathbf{y}, \Psi\right) = \mathcal{N}\left(\tilde{\Lambda}\tilde{\mathbf{y}}, \Psi\right) \end{aligned}$$

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$\begin{aligned} P(\mathbf{y}) &= \mathcal{N}(\mathbf{0}, I) & \tilde{\mathbf{y}} = U\mathbf{y} & \Rightarrow P(\tilde{\mathbf{y}}) = \mathcal{N}(U\mathbf{0}, UIU^T) = \mathcal{N}(\mathbf{0}, I) \\ P(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\Lambda\mathbf{y}, \Psi) & \tilde{\Lambda} = \Lambda U^T & \Rightarrow P(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}(\Lambda U^T U\mathbf{y}, \Psi) = \mathcal{N}(\tilde{\Lambda}\tilde{\mathbf{y}}, \Psi) \end{aligned}$$

The LGSSM likelihood is conserved with respect to [any invertible](#) transform of the latent:

$$\begin{aligned} P(\mathbf{y}_{t+1}|\mathbf{y}_t) &= \mathcal{N}(A\mathbf{y}_t, Q) \\ P(\mathbf{x}_t|\mathbf{y}_t) &= \mathcal{N}(C\mathbf{y}, R) \end{aligned}$$

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$\begin{aligned} P(\mathbf{y}) &= \mathcal{N}(\mathbf{0}, I) & \tilde{\mathbf{y}} = U\mathbf{y} & P(\tilde{\mathbf{y}}) = \mathcal{N}(U\mathbf{0}, UIU^T) = \mathcal{N}(\mathbf{0}, I) \\ P(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\Lambda\mathbf{y}, \Psi) & \tilde{\Lambda} = \Lambda U^T & P(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}(\Lambda U^T U\mathbf{y}, \Psi) = \mathcal{N}(\tilde{\Lambda}\tilde{\mathbf{y}}, \Psi) \end{aligned}$$

The LGSSM likelihood is conserved with respect to [any invertible](#) transform of the latent:

$$\begin{aligned} P(\mathbf{y}_{t+1}|\mathbf{y}_t) &= \mathcal{N}(A\mathbf{y}_t, Q) & \tilde{\mathbf{y}} = G\mathbf{y} & \tilde{A} = GAG^{-1} \\ P(\mathbf{x}_t|\mathbf{y}_t) &= \mathcal{N}(C\mathbf{y}, R) & \tilde{Q} = GQG^T & \tilde{C} = CG^{-1} \end{aligned}$$

## Degeneracies

Recall that the FA likelihood is conserved with respect to [orthogonal](#) transformations of  $\mathbf{y}$ :

$$\begin{aligned} P(\mathbf{y}) &= \mathcal{N}(\mathbf{0}, I) & \tilde{\mathbf{y}} = U\mathbf{y} & P(\tilde{\mathbf{y}}) = \mathcal{N}(U\mathbf{0}, UIU^T) = \mathcal{N}(\mathbf{0}, I) \\ P(\mathbf{x}|\mathbf{y}) &= \mathcal{N}(\Lambda\mathbf{y}, \Psi) & \Lambda = \Lambda U^T & P(\mathbf{x}|\tilde{\mathbf{y}}) = \mathcal{N}(\Lambda U^T U\mathbf{y}, \Psi) = \mathcal{N}(\tilde{\Lambda}\tilde{\mathbf{y}}, \Psi) \end{aligned}$$

The LGSSM likelihood is conserved with respect to [any invertible](#) transform of the latent:

$$\begin{aligned} P(\mathbf{y}_{t+1}|\mathbf{y}_t) &= \mathcal{N}(A\mathbf{y}_t, Q) & \tilde{\mathbf{y}} = G\mathbf{y} & \tilde{A} = GAG^{-1} \\ P(\mathbf{x}_t|\mathbf{y}_t) &= \mathcal{N}(C\mathbf{y}, R) & \tilde{Q} = GQG^T & \tilde{C} = CG^{-1} \\ \Rightarrow P(\tilde{\mathbf{y}}_{t+1}|\tilde{\mathbf{y}}_t) &= \mathcal{N}(GAG^{-1}G\mathbf{y}_t, GQG^T) = \mathcal{N}(\tilde{A}\tilde{\mathbf{y}}_t, \tilde{Q}) \\ P(\mathbf{x}_t|\tilde{\mathbf{y}}_t) &= \mathcal{N}(CG^{-1}G\mathbf{y}, R) = \mathcal{N}(\tilde{C}\tilde{\mathbf{y}}, R) \end{aligned}$$

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple **local maxima**. This complicates any ML-based approach to learning (including EM and gradient methods).

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple **local maxima**. This complicates any ML-based approach to learning (including EM and gradient methods).

Strategies:

- ▶ Restart EM/gradient ascent from different (often random) initial parameter values.

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple local maxima. This complicates any ML-based approach to learning (including EM and gradient methods).

Strategies:

- ▶ Restart EM/gradient ascent from different (often random) initial parameter values.
- ▶ Initialise output parameters with a stationary model:

PCA → FA → LGSSM

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple local maxima. This complicates any ML-based approach to learning (including EM and gradient methods).

Strategies:

- ▶ Restart EM/gradient ascent from different (often random) initial parameter values.
- ▶ Initialise output parameters with a stationary model:

PCA → FA → LGSSM

- ▶ Stochastic gradient methods and momentum. “Overstepping” EM.

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple local maxima. This complicates any ML-based approach to learning (including EM and gradient methods).

Strategies:

- ▶ Restart EM/gradient ascent from different (often random) initial parameter values.
- ▶ Initialise output parameters with a stationary model:

PCA → FA → LGSSM

- ▶ Stochastic gradient methods and momentum. “Overstepping” EM.
- ▶ Deterministic annealing.

## The likelihood landscape

Besides the degenerate solutions, LGSSM likelihood functions tend to have multiple local maxima. This complicates any ML-based approach to learning (including EM and gradient methods).

Strategies:

- ▶ Restart EM/gradient ascent from different (often random) initial parameter values.
- ▶ Initialise output parameters with a stationary model:

PCA → FA → LGSSM

- ▶ Stochastic gradient methods and momentum. “Overstepping” EM.
- ▶ Deterministic annealing.
- ▶ Non-ML learning (spectral methods).

## **Slow Feature Analysis (SFA)**

Is there a zero-noise limit for an LGSSM analogous to PCA?

## Slow Feature Analysis (SFA)

Is there a zero-noise limit for an LGSSM analogous to PCA?

We can take:

$$A = \text{diag} [a_1 \dots a_K] \rightarrow I \quad (\text{with } a_1 \leq a_2 \leq \dots \leq a_K \leq 1)$$

$$Q = I - AA^T \rightarrow 0 \quad (\text{setting the stationary latent covariance to } I)$$

$$R \rightarrow 0.$$

This limit yields a spectral algorithm called Slow Feature Analysis (or SFA).

## Slow Feature Analysis (SFA)

Is there a zero-noise limit for an LGSSM analogous to PCA?

We can take:

$$A = \text{diag} [a_1 \dots a_K] \rightarrow I \quad (\text{with } a_1 \leq a_2 \leq \dots \leq a_K \leq 1)$$

$$Q = I - AA^T \rightarrow 0 \quad (\text{setting the stationary latent covariance to } I)$$

$$R \rightarrow 0.$$

This limit yields a spectral algorithm called Slow Feature Analysis (or SFA).

SFA is conventionally defined as slowness pursuit (cf PCA and variance pursuit):

Given zero-mean series  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$  find a  $K \times D$  matrix  $W (= C^T)$  such that  $\mathbf{y}_t = W\mathbf{x}_t$  changes as slowly as possible.

## Slow Feature Analysis (SFA)

Is there a zero-noise limit for an LGSSM analogous to PCA?

We can take:

$$A = \text{diag} [a_1 \dots a_K] \rightarrow I \quad (\text{with } a_1 \leq a_2 \leq \dots \leq a_K \leq 1)$$

$$Q = I - AA^T \rightarrow 0 \quad (\text{setting the stationary latent covariance to } I)$$

$$R \rightarrow 0.$$

This limit yields a spectral algorithm called Slow Feature Analysis (or SFA).

SFA is conventionally defined as slowness pursuit (cf PCA and variance pursuit):

Given zero-mean series  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$  find a  $K \times D$  matrix  $W (= C^T)$  such that  $\mathbf{y}_t = W\mathbf{x}_t$  changes as slowly as possible.

Specifically, find

$$W = \underset{w}{\operatorname{argmin}} \sum_t \|\mathbf{y}_t - \mathbf{y}_{t-1}\|^2 \quad \text{subject to} \quad \sum_t \mathbf{y}_t \mathbf{y}_t^T = I.$$

The variance constraint prevents the trivial solutions  $W = 0$  and  $W = \mathbf{1}\mathbf{w}^T$ .

## Slow Feature Analysis (SFA)

Is there a zero-noise limit for an LGSSM analogous to PCA?

We can take:

$$A = \text{diag} [a_1 \dots a_K] \rightarrow I \quad (\text{with } a_1 \leq a_2 \leq \dots \leq a_K \leq 1)$$

$$Q = I - AA^T \rightarrow 0 \quad (\text{setting the stationary latent covariance to } I)$$

$$R \rightarrow 0.$$

This limit yields a spectral algorithm called Slow Feature Analysis (or SFA).

SFA is conventionally defined as slowness pursuit (cf PCA and variance pursuit):

Given zero-mean series  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$  find a  $K \times D$  matrix  $W (= C^T)$  such that  $\mathbf{y}_t = W\mathbf{x}_t$  changes as slowly as possible.

Specifically, find

$$W = \underset{w}{\operatorname{argmin}} \sum_t \|\mathbf{y}_t - \mathbf{y}_{t-1}\|^2 \quad \text{subject to} \quad \sum_t \mathbf{y}_t \mathbf{y}_t^T = I.$$

The variance constraint prevents the trivial solutions  $W = 0$  and  $W = \mathbf{1}\mathbf{w}^T$ .

$W$  can be found by solving the generalised eigenvalue problem

$$WA = \Omega WB \quad \text{where } A = \sum_t (\mathbf{x}_t - \mathbf{x}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1})^T \text{ and } B = \sum_t \mathbf{x}_t \mathbf{x}_t^T.$$

See <http://www.gatsby.ucl.ac.uk/~maneesh/papers/turner-sahani-2007-sfa.pdf>.

## (Generalised) Method of Moments estimators

What if we don't want the  $A \rightarrow I$  limit?

## (Generalised) Method of Moments estimators

What if we don't want the  $A \rightarrow I$  limit?

The limit is necessary to make the spectral estimate and the (limiting) ML estimates coincide. However, if we give up on ML, then there are general spectral estimates available.

## (Generalised) Method of Moments estimators

What if we don't want the  $A \rightarrow I$  limit?

The limit is necessary to make the spectral estimate and the (limiting) ML estimates coincide. However, if we give up on ML, then there are general spectral estimates available.

The ML parameter estimates are defined:

$$\theta^{\text{ML}} = \underset{\theta}{\operatorname{argmax}} \log P(\mathcal{X}|\theta)$$

and as you found, if  $P \in \text{ExpFam}$  with sufficient statistic  $T$  then

$$\langle T(\mathbf{x}) \rangle_{\theta^{\text{ML}}} = \frac{1}{N} \sum_i T(\mathbf{x}_i) \quad (\text{moment matching}).$$

## (Generalised) Method of Moments estimators

What if we don't want the  $A \rightarrow I$  limit?

The limit is necessary to make the spectral estimate and the (limiting) ML estimates coincide. However, if we give up on ML, then there are general spectral estimates available.

The ML parameter estimates are defined:

$$\theta^{\text{ML}} = \underset{\theta}{\operatorname{argmax}} \log P(\mathcal{X}|\theta)$$

and as you found, if  $P \in \text{ExpFam}$  with sufficient statistic  $T$  then

$$\langle T(\mathbf{x}) \rangle_{\theta^{\text{ML}}} = \frac{1}{N} \sum_i T(\mathbf{x}_i) \quad (\text{moment matching}).$$

We can generalise this condition to arbitrary  $T$  even if  $P \notin \text{ExpFam}$ .

That is, we find estimate  $\theta^*$  with

$$\langle T(\mathbf{x}) \rangle_{\theta^*} = \frac{1}{N} \sum_i T(\mathbf{x}_i) \quad \text{or} \quad \theta^* = \underset{\theta}{\operatorname{argmin}} \left\| \langle T(\mathbf{x}) \rangle_{\theta} - \frac{1}{N} \sum_i T(\mathbf{x}_i) \right\|_c$$

## (Generalised) Method of Moments estimators

What if we don't want the  $A \rightarrow I$  limit?

The limit is necessary to make the spectral estimate and the (limiting) ML estimates coincide. However, if we give up on ML, then there are general spectral estimates available.

The ML parameter estimates are defined:

$$\theta^{\text{ML}} = \underset{\theta}{\operatorname{argmax}} \log P(\mathcal{X}|\theta)$$

and as you found, if  $P \in \text{ExpFam}$  with sufficient statistic  $T$  then

$$\langle T(\mathbf{x}) \rangle_{\theta^{\text{ML}}} = \frac{1}{N} \sum_i T(\mathbf{x}_i) \quad (\text{moment matching}).$$

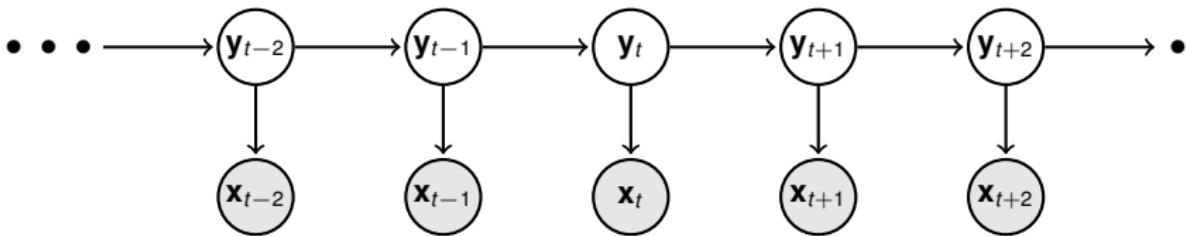
We can generalise this condition to arbitrary  $T$  even if  $P \notin \text{ExpFam}$ .

That is, we find estimate  $\theta^*$  with

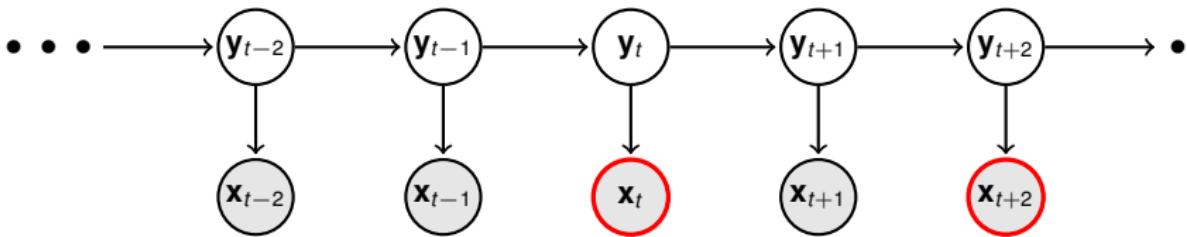
$$\langle T(\mathbf{x}) \rangle_{\theta^*} = \frac{1}{N} \sum_i T(\mathbf{x}_i) \quad \text{or} \quad \theta^* = \underset{\theta}{\operatorname{argmin}} \|\langle T(\mathbf{x}) \rangle_{\theta} - \frac{1}{N} \sum_i T(\mathbf{x}_i)\|_C$$

Judicious choice of  $T$  and metric  $C$  might make solution unique (no local optima) and consistent (correct given infinite within-model data).

## Ho-Kalman SSID for LGSSMs

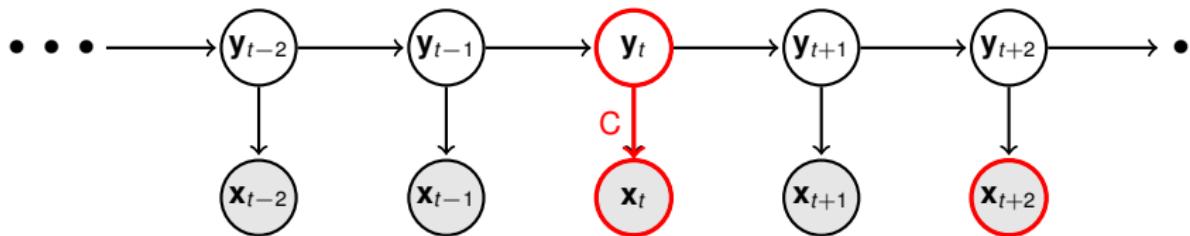


## Ho-Kalman SSID for LGSSMs



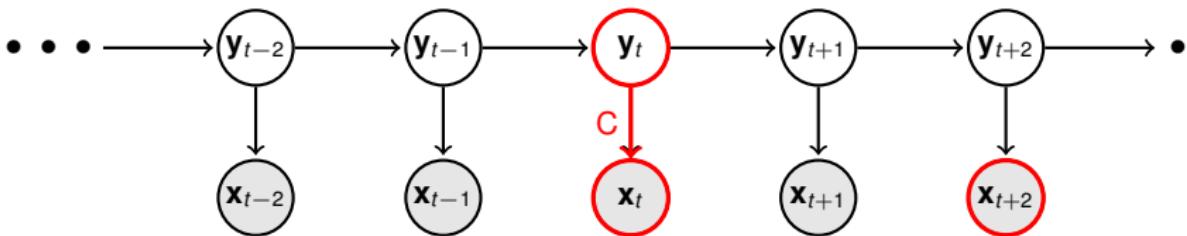
$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle$$

## Ho-Kalman SSID for LGSSMs



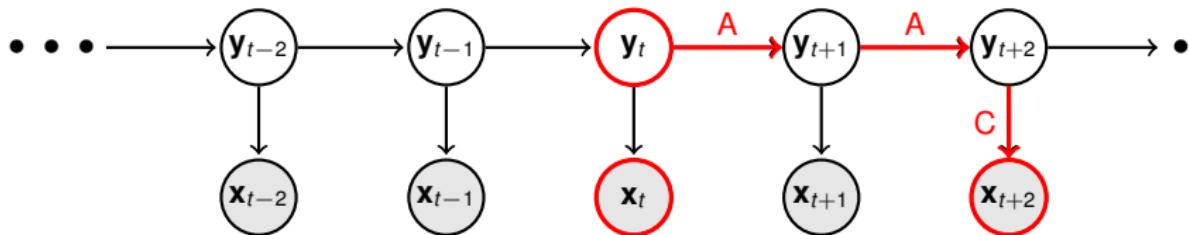
$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} (C \mathbf{y}_t + \boldsymbol{\eta})^\top \rangle$$

## Ho-Kalman SSID for LGSSMs



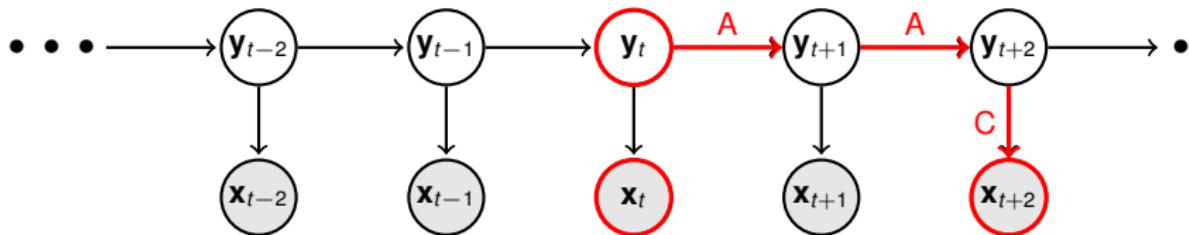
$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top$$

## Ho-Kalman SSID for LGSSMs



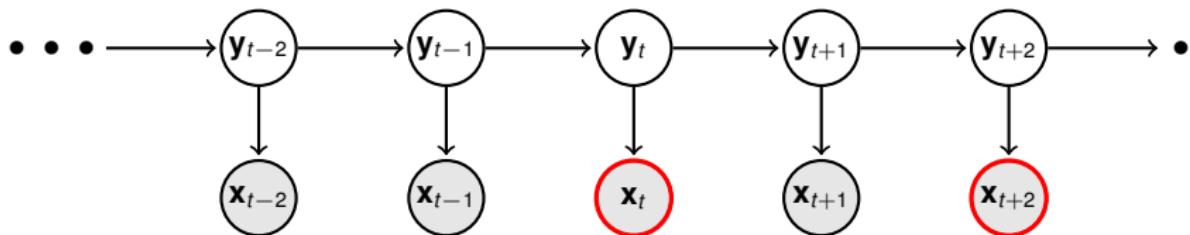
$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = \langle (CA^\tau \mathbf{y}_t + \eta) \mathbf{y}_t^\top \rangle C^\top$$

## Ho-Kalman SSID for LGSSMs



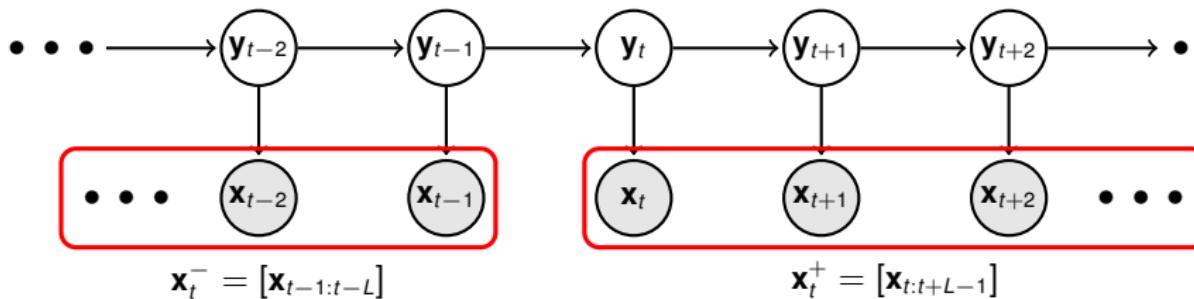
$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = C A^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top$$

## Ho-Kalman SSID for LGSSMs



$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = CA^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top = CA^\tau \Pi C^\top$$

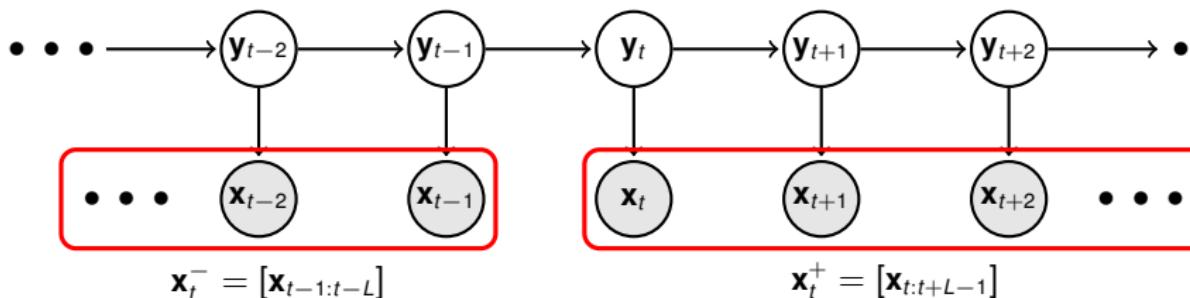
## Ho-Kalman SSID for LGSSMs



$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = C A^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top = C A^\tau \Pi C^\top$$

$$H \equiv \langle \mathbf{x}_t^+, \mathbf{x}_t^{-\top} \rangle$$

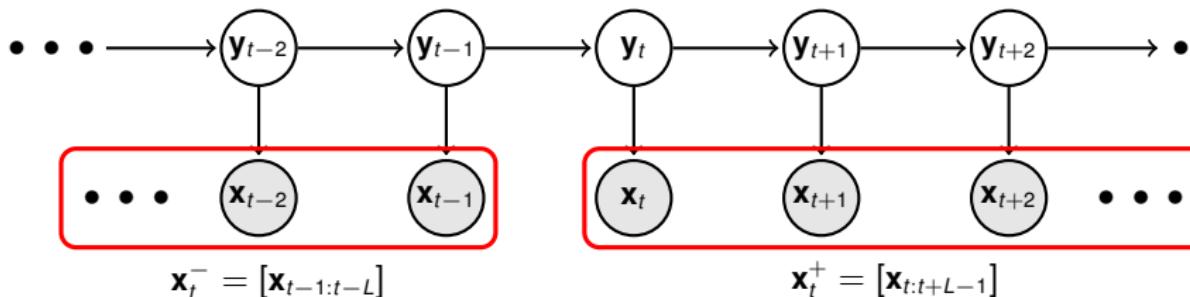
## Ho-Kalman SSID for LGSSMs



$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = CA^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top = CA^\tau \Pi C^\top$$

$$H \equiv \langle \mathbf{x}_t^+, \mathbf{x}_t^{-\top} \rangle = \begin{bmatrix} M_1 & M_2 & \cdots & M_L \\ M_2 & M_3 & & \\ \vdots & & & \vdots \\ M_L & & \cdots & M_{2L-1} \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{L-1} \end{bmatrix} [A\Pi C^\top \quad \cdots \quad A^L\Pi C^\top]$$

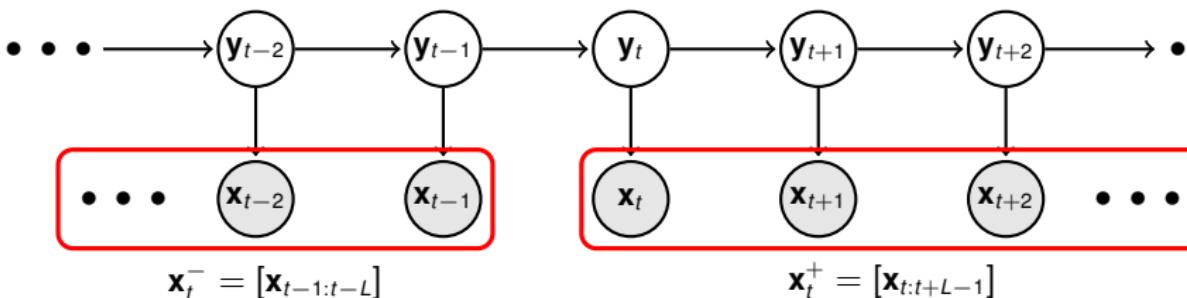
## Ho-Kalman SSID for LGSSMs



$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = C A^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top = C A^\tau \Pi C^\top$$

$$H \equiv \langle \mathbf{x}_t^+, \mathbf{x}_t^{-\top} \rangle = \begin{bmatrix} M_1 & M_2 & \cdots & M_L \\ M_2 & M_3 & & \\ \vdots & & & \vdots \\ M_L & & \cdots & M_{2L-1} \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{L-1} \end{bmatrix} \begin{bmatrix} A\Pi C^\top & \ddots & A^L \Pi C^\top \end{bmatrix}_{K \times LD}$$

## Ho-Kalman SSID for LGSSMs



$$M_\tau \equiv \langle \mathbf{x}_{t+\tau} \mathbf{x}_t^\top \rangle = \langle \mathbf{x}_{t+\tau} \mathbf{y}_t^\top \rangle C^\top = CA^\tau \langle \mathbf{y}_t \mathbf{y}_t^\top \rangle C^\top = CA^\tau \Pi C^\top$$

$$H \equiv \langle \mathbf{x}_t^+, \mathbf{x}_t^{-\top} \rangle = \begin{bmatrix} M_1 & M_2 & \cdots & M_L \\ M_2 & M_3 & & \\ \vdots & & & \vdots \\ M_L & & \cdots & M_{2L-1} \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{L-1} \end{bmatrix} \begin{bmatrix} A\Pi C^\top & \ddots & A^L \Pi C^\top \end{bmatrix}_{LD \times K}$$

Off-diagonal correlation unaffected by noise.  $SVD(\frac{1}{T} \sum \mathbf{x}_t^+ \mathbf{x}_t^{-\top})$  yields least-squares estimates of  $\Xi$  and  $\Upsilon$ . Regression between blocks of  $\Xi$  yields  $\hat{A}$  and  $\hat{C}$ .

## ML and spectral learning

Spectral learning:

Maximum likelihood learning:

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.
- ▶ Assumes stationarity. May be inappropriate for short sequences.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.
- ▶ Assumes stationarity. May be inappropriate for short sequences.
- ▶ HMM learning returns OOM parameters – may not correspond to any HMM or indeed to proper probabilistic model.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.
- ▶ Assumes stationarity. May be inappropriate for short sequences.
- ▶ HMM learning returns OOM parameters – may not correspond to any HMM or indeed to proper probabilistic model.
- ▶ Not easily generalised to nonlinear or more complex models (but see  
<http://www.gatsby.ucl.ac.uk/~maneesh/papers/buesing-etal-2012-nips.pdf>).

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.
- ▶ Assumes stationarity. May be inappropriate for short sequences.
- ▶ HMM learning returns OOM parameters – may not correspond to any HMM or indeed to proper probabilistic model.
- ▶ Not easily generalised to nonlinear or more complex models (but see  
<http://www.gatsby.ucl.ac.uk/~maneesh/papers/buesing-etal-2012-nips.pdf>).
- ▶ In practice, error in recovered parameters is often large.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).
- ▶ Generalises to “principled” approximate algorithms for nonlinear or complex models.

## ML and spectral learning

Spectral learning:

- ▶ Efficient closed-form solution finds global optimum.
- ▶ Consistent (recovers true parameters upto degeneracies from infinite within-model data).
- ▶ Eigen-/singular-value spectrum clue to latent dimensionality.
- ▶ Assumes stationarity. May be inappropriate for short sequences.
- ▶ HMM learning returns OOM parameters – may not correspond to any HMM or indeed to proper probabilistic model.
- ▶ Not easily generalised to nonlinear or more complex models (but see  
<http://www.gatsby.ucl.ac.uk/~maneesh/papers/buesing-etal-2012-nips.pdf>).
- ▶ In practice, error in recovered parameters is often large.
- ▶ Often valuable as initialisation for ML methods.

Maximum likelihood learning:

- ▶ Requires iterative maximisation.
- ▶ Many local optima.
- ▶ Consistent and asymptotically efficient (if the global maximum can be found).
- ▶ Generalises to “principled” approximate algorithms for nonlinear or complex models.

## **Supervised methods**

## Not so latent variables

- ▶ Controlled experiments use repeated **trials**
  - ▶ One or more experimental parameter or **factor** varied systematically.
  - ▶ Each unique configuration of factors is a **condition**.
- ▶ May also observe (generally continuous-valued) behavioural outputs or a random/natural stimulus: **covariates**.
- ▶ Ideally, **unsupervised** structure in data would reflect these values.
- ▶ Weak signals? Non-linearities?
- ▶ Unsupervised projections may not naturally separate the different factors: **unmixing**.
- ▶ We will look at **supervised** methods designed to relate multivariate data to known experimental factors or covariates.
- ▶ Methods we consider are also used to study structure in the condition averages:  
equivalent to having one trial per condition
  - ▶ averaging may make noise more Gaussian
  - ▶ **but still not equal variance**

## Two cases

The tools needed in two different cases are slightly different:

- ▶ **Categorical** factors: discrete repeated values (almost always experimental control).
  - ▶ Stimulus (say, object) identity.
  - ▶ Behavioural instruction.
  - ▶ “Context” signal.
- ▶ We sometimes ignore the metricity of factors: time bin, gabor orientation, ...
- ▶ **Continuous or ordinal** covariates: experimental factors or covariates themselves lie in a metric space.
  - ▶ time in trial
  - ▶ orientation
  - ▶ reaching movement kinematics

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$S_t = \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \end{aligned}$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} - \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} + \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \end{aligned}$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} - \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} + \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle - \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle - \left\langle \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \end{aligned}$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} - \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} + \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle - \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle - \left\langle \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \\ &= \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle^\top - \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle \bar{\mathbf{x}}_t^{(\kappa)\top} + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \end{aligned}$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i:k^{(i)}=\kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i:k^{(i)}=\kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} - \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} + \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i:k^{(i)}=\kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle - \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle - \left\langle \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \\ &= \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle^\top - \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle \bar{\mathbf{x}}_t^{(\kappa)\top} + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \\ &= \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle_\kappa + \left\langle \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i:k^{(i)}=\kappa} \right\rangle_\kappa \end{aligned}$$

## Categorical factors: decomposition of variance

Suppose on  $i$ th trial we have:

- ▶ factor value  $k^{(i)} \in 1 \dots K$
- ▶ recorded (binned) data  $\mathbf{x}_t^{(i)} \in \mathbb{R}^N$ ,  $t = [1 \dots T]$ ,  $N = \#$  neurons; remove global mean.

Consider time  $t$ .

- ▶ For each condition  $\kappa$  we have the condition mean (PSTH):  $\bar{\mathbf{x}}_t^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{i: k^{(i)} = \kappa}$
- ▶ Let us write  $\mathbf{x}_t^{(i)} = \bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)}$ .
- ▶ Then total scatter or variance:

$$\begin{aligned} S_t &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle = \left\langle (\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \left\langle (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}_t^{(\kappa)} + \Delta \mathbf{x}_t^{(i)})^\top \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} - \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} + \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa \\ &= \left\langle \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle - \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle - \left\langle \Delta \mathbf{x}_t^{(i)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \\ &= \left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} - \bar{\mathbf{x}}_t^{(\kappa)} \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle^\top - \left\langle \Delta \mathbf{x}_t^{(i)} \right\rangle \bar{\mathbf{x}}_t^{(\kappa)\top} + \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle \right\rangle_\kappa \\ &= \underbrace{\left\langle \bar{\mathbf{x}}_t^{(\kappa)} \bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle_\kappa}_{\text{Var(cond. mean)}} + \underbrace{\left\langle \left\langle \Delta \mathbf{x}_t^{(i)} \Delta \mathbf{x}_t^{(i)\top} \right\rangle_{i: k^{(i)} = \kappa} \right\rangle_\kappa}_{\text{Mean(cond. var)}} = S_t^{(\text{signal})} + S_t^{(\text{noise})} \end{aligned}$$

## Multifactor decomposition of variance

We can consider time bin  $t$  to be another factor (and may have many experimental factors).

Write

- ▶  $\bar{\mathbf{x}}_t = \left\langle \mathbf{x}_t^{(i)} \right\rangle_i$
- ▶  $\bar{\mathbf{x}}^{(\kappa)} = \left\langle \mathbf{x}_t^{(i)} \right\rangle_{t,i:k^{(i)}=\kappa}$
- ▶  $\Delta\bar{\mathbf{x}}_t^{(\kappa)} = \bar{\mathbf{x}}_t^{(\kappa)} - \bar{\mathbf{x}}_t - \bar{\mathbf{x}}^{(\kappa)}$

Then

$$\begin{aligned} S^{(\text{total})} &= \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)\top} \right\rangle_{t,i} = \left\langle (\bar{\mathbf{x}}_t + \bar{\mathbf{x}}^{(k^{(i)})} + \Delta\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta\mathbf{x}_t^{(i)})(\bar{\mathbf{x}}_t + \bar{\mathbf{x}}^{(k^{(i)})} + \Delta\bar{\mathbf{x}}_t^{(k^{(i)})} + \Delta\mathbf{x}_t^{(i)})^\top \right\rangle \\ &= \left\langle \bar{\mathbf{x}}_t \bar{\mathbf{x}}_t^\top \right\rangle_t + \left\langle \bar{\mathbf{x}}^{(\kappa)} \bar{\mathbf{x}}^{(\kappa)\top} \right\rangle_\kappa + \left\langle \Delta\bar{\mathbf{x}}_t^{(\kappa)} \Delta\bar{\mathbf{x}}_t^{(\kappa)\top} \right\rangle_{t,\kappa} + \left\langle \Delta\mathbf{x}_t^{(i)} \Delta\mathbf{x}_t^{(i)\top} \right\rangle_{t,i} \\ &= S^{(\text{time})} + S^{(\text{factor})} + S^{(\text{interact})} + S^{(\text{noise})} \end{aligned}$$

In general, for multiple factors:

$$\begin{aligned} S^{(\text{total})} &= S^{(t)} + S^{(f_1)} + S^{(f_2)} + \dots \\ &\quad + S^{(t \times f_1)} + S^{(t \times f_2)} + S^{(f_1 \times f_2)} + \dots \\ &\quad + S^{(t \times f_1 \times f_2)} + \dots + S^{(t \times f_1 \times f_2 \times \dots)} + \dots \\ &\quad + S^{(\text{noise})} \end{aligned}$$

This decomposition is fundamental to the Multivariate Analysis of Variance (MANOVA).

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).
- ▶ Rotate the projection vectors so as to find a good compromise between retaining variance related to signal and avoiding other sources.

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).
- ▶ Rotate the projection vectors so as to find a good compromise between retaining variance related to signal and avoiding other sources.  
Two ideas:

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).
- ▶ Rotate the projection vectors so as to find a good compromise between retaining variance related to signal and avoiding other sources.

Two ideas:

  - ▶ Maximise projected signal to noise ratio.

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).
- ▶ Rotate the projection vectors so as to find a good compromise between retaining variance related to signal and avoiding other sources.

Two ideas:

  - ▶ Maximise projected signal to noise ratio.
  - ▶ Minimise error between reconstructed trial and signal.

## Studying factor-related variance

The idea behind our first group of methods is to look for a projection of the data that captures the structure related to one factor at a time.

- ▶ A first thought: Use PCA / FA / etc. on the condition means.
  - ▶ Maximises projected signal variance, but does not reject variance from trial-to-trial noise, or from other factors (unmixing).
- ▶ Rotate the projection vectors so as to find a good compromise between retaining variance related to signal and avoiding other sources.

Two ideas:

  - ▶ Maximise projected signal to noise ratio.
  - ▶ Minimise error between reconstructed trial and signal.

We consider one factor at a time:  $S^{(\text{total})} = S^{(\text{factor})} + S^{(\text{other})} = S_F + S_\Delta$ .

## Linear Discriminant Analysis (LDA)

Originally due to Fisher (1936), widely discussed in text books.

$$\text{Find } \mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmax}} \frac{\mathbf{w}^T S_F \mathbf{w}}{\mathbf{w}^T S_\Delta \mathbf{w}}$$

In this context,  $S_F$  is usually called **between-class scatter** – scatter between condition means.  $S_\Delta$  is the average **within-class scatter**.

The projection is (heuristically) designed to maximise separation of the classes.

[The same idea, slightly generalised, has been discussed in neuroscience as “Denoising Source Separation” (Simon and de Cheveigné) and “Joint Decorrelation” (de Cheveigné and Parra).]

## Linear Discriminant Analysis (LDA)

First note that  $\frac{\mathbf{w}^T S_F \mathbf{w}}{\mathbf{w}^T S_\Delta \mathbf{w}} = \frac{\mathbf{w}^T S_\Delta^{1/2} S_\Delta^{-1/2} S_F S_\Delta^{-1/2} S_\Delta^{1/2} \mathbf{w}}{\mathbf{w}^T S_\Delta^{1/2} S_\Delta^{1/2} \mathbf{w}}$  so that we can define  $\tilde{\mathbf{w}} = S_\Delta^{1/2} \mathbf{w}$  and find

$$\tilde{\mathbf{w}}^* = \operatorname{argmax}_{\tilde{\mathbf{w}}} \frac{\tilde{\mathbf{w}}^T S_\Delta^{-1/2} S_F S_\Delta^{-1/2} \tilde{\mathbf{w}}}{\tilde{\mathbf{w}}^T \tilde{\mathbf{w}}} = \operatorname{argmax}_{\|\tilde{\mathbf{w}}\|=1} \tilde{\mathbf{w}}^T S_\Delta^{-1/2} S_F S_\Delta^{-1/2} \tilde{\mathbf{w}}$$

finally mapping back to obtain  $\mathbf{w}^* = S_\Delta^{-1/2} \tilde{\mathbf{w}}^*$ .

It may be easiest to think of this as a two-stage process:

- ▶ Whiten the non-factor scatter (transform data to  $\tilde{\mathbf{x}}_t^{(i)} = S_\Delta^{-1/2} \mathbf{x}_t^{(i)}$ ), so that  $\tilde{S}_\Delta = I$ .
- ▶ Run PCA on the means  $\bar{\mathbf{x}}^{(\kappa)}$  in the whitened space; diagonalising  $\tilde{S}_F = S_\Delta^{-1/2} S_F S_\Delta^{-1/2}$ .  
 $\Rightarrow \tilde{S}_F \tilde{\mathbf{w}}^* = \lambda \tilde{\mathbf{w}}^*$   
 $\Rightarrow S_\Delta^{-1/2} S_F S_\Delta^{-1/2} S_\Delta^{1/2} \mathbf{w}^* = \lambda S_\Delta^{1/2} \mathbf{w}^*$   
 $\Rightarrow S_\Delta^{-1} S_F \mathbf{w}^* = \lambda \mathbf{w}^*$

So solutions are eigenvectors of  $S_\Delta^{-1} S_F$  (or generalised eigenvectors of  $S_\Delta$  and  $S_F$ ).

We can use more than one eigenvector of  $\tilde{S}_F$  to capture subspace with maximal whitened signal variance, although these will not be orthogonal when transformed back to the original space.

## Demixed Principal Component Analysis (DPCA)

Two slightly different recent proposals from Machens and collaborators [NIPS and ArXiv]. We will describe the ArXiv version.

$$\text{Find } \underset{\mathbf{w}, \|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\bar{\mathbf{x}}^{(k(i))} - \mathbf{u}\mathbf{w}^\top \mathbf{x}_t^{(i)}\|^2$$

**Reduced rank regression.** Compress data to optimally preserve information about factor means: compare to bottleneck view of PCA.

Similar intuition to LDA, but slightly different cost function.

## DPCA

Reduced rank regression has a well-known solution: The output direction ( $\mathbf{u}^*$ ) will align with maximum output-variance mode of MSE regression.

That is:

$$\text{let } Q = \left\langle \mathbf{x}_t^{(i)} \mathbf{x}_t^{(i)} \right\rangle^{-1} \left\langle \mathbf{x}_t^{(i)} \bar{\mathbf{x}}^{(k^{(i)})} \right\rangle = (S_{Tot})^{-1} S_F$$

$$\text{then } \mathbf{u}^* = \text{eig}(Q^\top S_{Tot} Q) = \text{eig}(S_F S_{Tot}^{-1} S_{Tot} S_{Tot}^{-1} S_F) = \text{eig}(S_F S_{Tot}^{-1} S_F)$$

$$\text{and } \mathbf{w}^* = Q \mathbf{u}^*$$

Now,

$$S_F S_{Tot}^{-1} S_F \mathbf{u}^* = \mathbf{u}^* \lambda$$

$$\Rightarrow S_{Tot}^{-1} S_F S_F S_{Tot}^{-1} S_F \mathbf{u}^* = S_{Tot}^{-1} S_F \mathbf{u}^* \lambda$$

$$\Rightarrow S_{Tot}^{-1} S_F^2 \mathbf{w}^* = \mathbf{w}^* \lambda$$

So solutions are eigenvectors of  $S_{Tot}^{-1} S_F^2$ .

## DPCA – alternative derivation

We can write the objective as:

$$\begin{aligned}\mathcal{C}(U, W) &= \sum_{i,t} \|\bar{\mathbf{x}}^{(k(i))} - UW^\top \mathbf{x}_t^{(i)}\|^2 \propto \text{Tr} \left[ \left\langle (\bar{\mathbf{x}}^{(k(i))} - UW^\top \mathbf{x}_t^{(i)}) (\bar{\mathbf{x}}^{(k(i))} - UW^\top \mathbf{x}_t^{(i)})^\top \right\rangle \right] \\ &= \text{Tr} \left[ \left\langle ((I - UW^\top) \bar{\mathbf{x}}^{(k(i))} - UW^\top \Delta \mathbf{x}_t^{(i)}) ((I - UW^\top) \bar{\mathbf{x}}^{(k(i))} - UW^\top \Delta \mathbf{x}_t^{(i)})^\top \right\rangle \right] \\ &= \text{Tr} \left[ (I - UW^\top) (I - UW^\top)^\top S_F + WU^\top UW^\top S_\Delta \right] \\ &= \text{Tr} \left[ (I - UW^\top) (I - UW^\top)^\top S_F + WW^\top S_\Delta \right] \\ &= \text{Tr} \left[ S_F + WW^\top S_{Tot} - 2UW^\top S_F \right]\end{aligned}$$

Differentiate wrt  $W$  to find maximum:

$$\frac{\partial \mathcal{C}}{\partial W} = 2S_{Tot}W - 2S_FU = 0 \quad \Rightarrow \quad W^* = S_{Tot}^{-1}S_FU$$

So

$$\begin{aligned}\mathcal{C}(U) &= \text{Tr}[S_f] + \text{Tr} \left[ U^\top S_F S_{Tot}^{-1} S_{Tot} S_{Tot}^{-1} S_F U - 2U^\top S_F S_{Tot}^{-1} S_F U \right] \\ &= \text{Tr}[S_f] - \text{Tr} \left[ U^\top S_F S_{Tot}^{-1} S_F U \right]\end{aligned}$$

and  $U^*$  is given by the dominant eigenvectors of  $S_F S_{Tot}^{-1} S_F$ , giving us the same result.

## DPCA – an aside

What if we require  $W = U$  (i.e. projection and reconstruction are complementary orthogonal projections)?

Then, we can re-write the cost function again:

$$\begin{aligned}\mathcal{C}(U, W) &= \text{Tr} \left[ S_F + WW^T S_{Tot} - 2UW^T S_F \right] \\ &= \text{Tr} \left[ S_F + WW^T (S_F + S_\Delta) - 2WW^T S_F \right] \\ &= \text{const} + \text{Tr} \left[ W^T (S_\Delta - S_F) W \right]\end{aligned}$$

So with this constraint DPCA will find a projection which maximises the *difference* between  $S_F$  and  $S_\Delta$ . Recall that LDA maximises the corresponding *ratio*.

## Relating LDA and DPCA

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$\begin{aligned} S_{\Delta}\mathbf{w} &= \lambda S_F\mathbf{w} \\ \Rightarrow S_{\Delta}\mathbf{w} + S_F\mathbf{w} &= (\lambda + 1)S_F\mathbf{w} \end{aligned}$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F\mathbf{w} = (\lambda + 1)S_F\mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1)S_F\mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1)S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u}\mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

$$\text{Then } Q = S_{Tot}^{-1} M, \quad \mathbf{u}^* = \operatorname{eig}(Q^T S_{Tot} Q) = \operatorname{eig}(M^T S_{Tot}^{-1} M) \text{ and } \mathbf{w}^* = Q \mathbf{u}^*.$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

Then  $Q = S_{Tot}^{-1} M$ ,  $\mathbf{u}^* = \operatorname{eig}(Q^T S_{Tot} Q) = \operatorname{eig}(M^T S_{Tot}^{-1} M)$  and  $\mathbf{w}^* = Q \mathbf{u}^*$ .

So  $M^T S_{Tot}^{-1} M \mathbf{u}^* = \mathbf{u}^* \lambda$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

Then  $Q = S_{Tot}^{-1} M$ ,  $\mathbf{u}^* = \operatorname{eig}(Q^T S_{Tot} Q) = \operatorname{eig}(M^T S_{Tot}^{-1} M)$  and  $\mathbf{w}^* = Q \mathbf{u}^*$ .

So  $M^T S_{Tot}^{-1} M \mathbf{u}^* = \mathbf{u}^* \lambda$

$$\Rightarrow S_{Tot}^{-1} M M^T S_{Tot}^{-1} M \mathbf{u}^* = S_{Tot}^{-1} M \mathbf{u}^* \lambda$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

Then  $Q = S_{Tot}^{-1} M$ ,  $\mathbf{u}^* = \operatorname{eig}(Q^T S_{Tot} Q) = \operatorname{eig}(M^T S_{Tot}^{-1} M)$  and  $\mathbf{w}^* = Q \mathbf{u}^*$ .

So  $M^T S_{Tot}^{-1} M \mathbf{u}^* = \mathbf{u}^* \lambda$

$$\Rightarrow S_{Tot}^{-1} M M^T S_{Tot}^{-1} M \mathbf{u}^* = S_{Tot}^{-1} M \mathbf{u}^* \lambda$$

$$\Rightarrow S_{Tot}^{-1} M M^T \mathbf{w}^* = \mathbf{w}^* \lambda$$

## Relating LDA and DPCA

1. LDA projections are given by generalised eigenvectors:

$$S_{\Delta}\mathbf{w} = \lambda S_F \mathbf{w}$$

$$\Rightarrow S_{\Delta}\mathbf{w} + S_F \mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

$$\Rightarrow S_{Tot}\mathbf{w} = (\lambda + 1) S_F \mathbf{w}$$

so LDA projects are also generalised eigenvectors of  $(S_{Tot}, S_F)$

$\Rightarrow$  eigenvectors of  $S_{Tot}^{-1} S_F$  if inverse exists.

2. Define  $\mathbf{k}^{(i)}$  to be the  $K$ -dimensional indicator vector (1 for coordinate  $k^{(i)}$ , 0 else). Then

$$\mathbf{w} = \underset{\|\mathbf{u}\|=1}{\operatorname{argmin}} \sum_{i,t} \|\mathbf{k}^{(i)} - \mathbf{u} \mathbf{w}^T \mathbf{x}_t^{(i)}\|^2 \text{ yields LDA:}$$

$$\text{Let } M = \left\langle \mathbf{x}_t^{(i)} \mathbf{k}^{(i)\top} \right\rangle = [\bar{\mathbf{x}}^{(1)} \bar{\mathbf{x}}^{(2)} \dots].$$

$$\text{Then } Q = S_{Tot}^{-1} M, \quad \mathbf{u}^* = \operatorname{eig}(Q^T S_{Tot} Q) = \operatorname{eig}(M^T S_{Tot}^{-1} M) \text{ and } \mathbf{w}^* = Q \mathbf{u}^*.$$

$$\text{So} \quad M^T S_{Tot}^{-1} M \mathbf{u}^* = \mathbf{u}^* \lambda$$

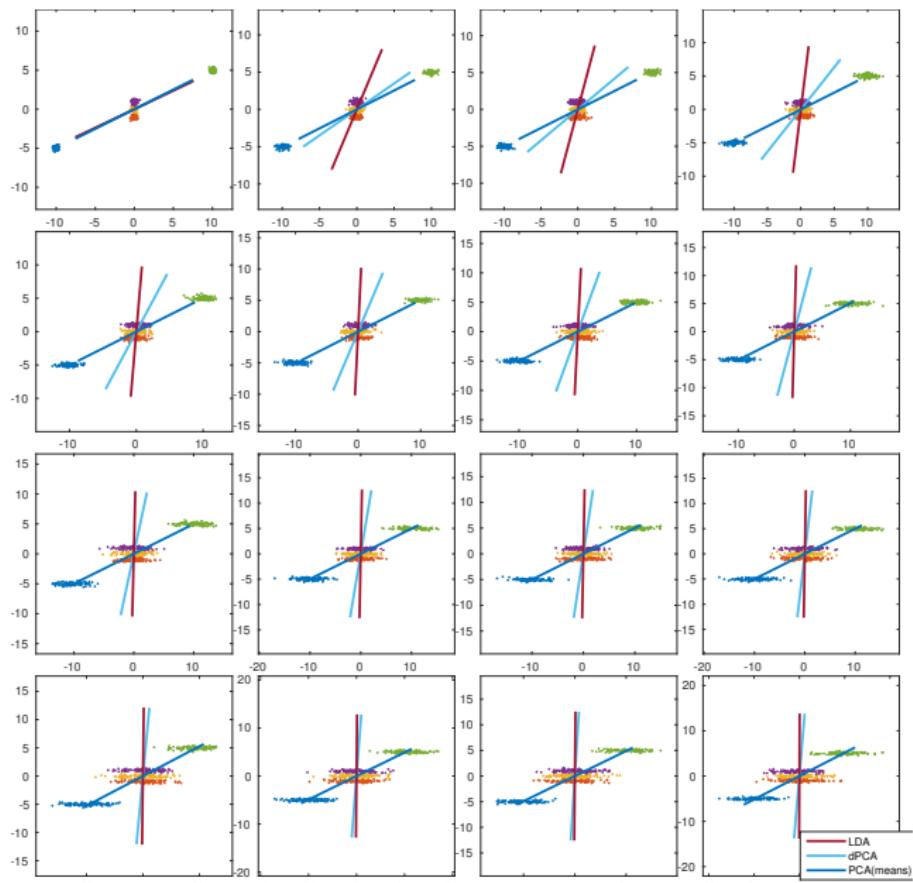
$$\Rightarrow S_{Tot}^{-1} M M^T S_{Tot}^{-1} M \mathbf{u}^* = S_{Tot}^{-1} M \mathbf{u}^* \lambda$$

$$\Rightarrow S_{Tot}^{-1} \underbrace{M M^T}_{KS_F} \mathbf{w}^* = \mathbf{w}^* \lambda$$

## Comparison

	LDA	DPCA
Cost	$\max \text{Tr} [(W^\top S_\Delta W)^{-1} (W^\top S_F W)]$	$\min_{U^\top U=I} \text{Tr} [W^\top S_{Tot} W - 2W^\top S_F U]$
Eigenprob	$S_\Delta^{-1} S_F \equiv S_{Tot}^{-1} S_F$	$S_{Tot}^{-1} S_F^2$
RRR	$\mathbf{x}_t^{(i)} \rightarrow \mathbf{k}^{(i)}$	$\mathbf{x}_t^{(i)} \rightarrow \bar{\mathbf{x}}^{(k^{(i)})}$

# Comparison



## Continuous / ordinal covariates

- ▶ Regression
- ▶ Canonical covariance analysis: CVA
- ▶ Canonical correlation analysis: CCA

## Canonical Correlations/Covariance Analysis

Data vector pairs:  $\mathcal{D} = \{(\mathbf{u}_1, \mathbf{v}_1), (\mathbf{u}_2, \mathbf{v}_2) \dots\}$  in spaces  $\mathcal{U}$  and  $\mathcal{V}$ .

### Classic CCA

- ▶ Find unit vectors  $\mathbf{v}_1 \in \mathcal{U}$ ,  $\phi_1 \in \mathcal{V}$  such that the (Pearson) correlation of  $\mathbf{u}_i^T \mathbf{v}_1$  and  $\mathbf{v}_i^T \phi_1$  is maximised.
- ▶ As with PCA, repeat in orthogonal (wrt data covariance) subspaces.
- ▶ **svd**( $\Sigma_u^{-1/2} \Sigma_{uv} \Sigma_v^{-1/2}$ )

### CVA

- ▶ **svd**( $\Sigma_{uv}$ )

### Probabilistic CCA

- ▶ Generative model with latent  $\mathbf{y}_i \in \mathbb{R}^K$ :

$$\mathbf{y} \sim \mathcal{N}(0, I)$$

$$\mathbf{u} \sim \mathcal{N}(\Upsilon \mathbf{y}, \Psi_u) \quad \Psi_u \succcurlyeq 0$$

$$\mathbf{v} \sim \mathcal{N}(\Phi \mathbf{y}, \Psi_v) \quad \Psi_v \succcurlyeq 0$$

- ▶ Block diagonal noise.

## Modes of covariation

What form does this population-movement covariation take?

## Modes of covariation

What form does this population-movement covariation take?

“Canonical Covariance Analysis”:

- ▶ For each reach target: find mean movement trajectory and mean firing profile (PSTH).

$$\bar{\mathbf{m}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{m}_t^{n(c)}$$

$$\bar{\mathbf{r}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{r}_t^{n(c)}$$

$[\mathbf{m}_t \in \mathbb{R}^{\# \text{ move params}}; \mathbf{r}_t \in \mathbb{R}^{\# \text{neurons}}]$

## Modes of covariation

What form does this population-movement covariation take?

“Canonical Covariance Analysis”:

- ▶ For each reach target: find mean movement trajectory and mean firing profile (PSTH).

$$\bar{\mathbf{m}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{m}_t^{n(c)}$$

$$\bar{\mathbf{r}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{r}_t^{n(c)}$$

$[\mathbf{m}_t \in \mathbb{R}^{\# \text{ move params}}; \mathbf{r}_t \in \mathbb{R}^{\# \text{neurons}}]$

- ▶ For each trial: find deviation from condition means.

$$\delta \mathbf{m}_t^{n(c)} = \mathbf{m}_t^{n(c)} - \bar{\mathbf{m}}_t^c$$

$$\delta \mathbf{r}_t^{n(c)} = \mathbf{r}_t^{n(c)} - \bar{\mathbf{r}}_t^c$$

## Modes of covariation

What form does this population-movement covariation take?

“Canonical Covariance Analysis”:

- ▶ For each reach target: find mean movement trajectory and mean firing profile (PSTH).

$$\bar{\mathbf{m}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{m}_t^{n(c)}$$

$$\bar{\mathbf{r}}_t^c = \frac{1}{N_{\text{trials}}^c} \sum_n \mathbf{r}_t^{n(c)}$$

$[\mathbf{m}_t \in \mathbb{R}^{\# \text{ move params}}; \mathbf{r}_t \in \mathbb{R}^{\# \text{neurons}}]$

- ▶ For each trial: find deviation from condition means.

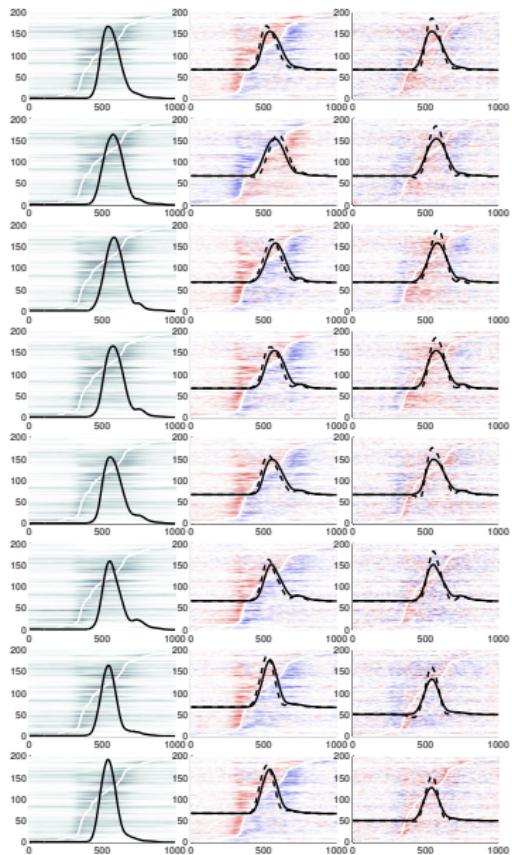
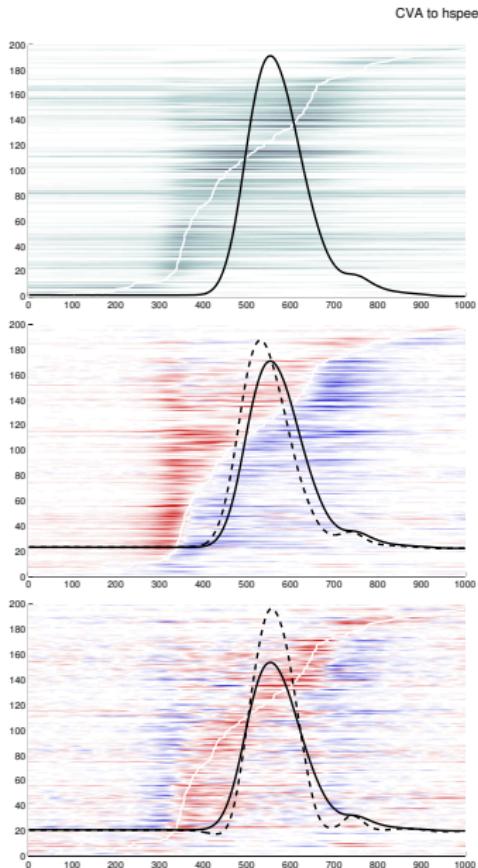
$$\delta\mathbf{m}_t^{n(c)} = \mathbf{m}_t^{n(c)} - \bar{\mathbf{m}}_t^c$$

$$\delta\mathbf{r}_t^{n(c)} = \mathbf{r}_t^{n(c)} - \bar{\mathbf{r}}_t^c$$

- ▶ For all trials: find simultaneous projection of deviations in movement and activity that have the highest covariance

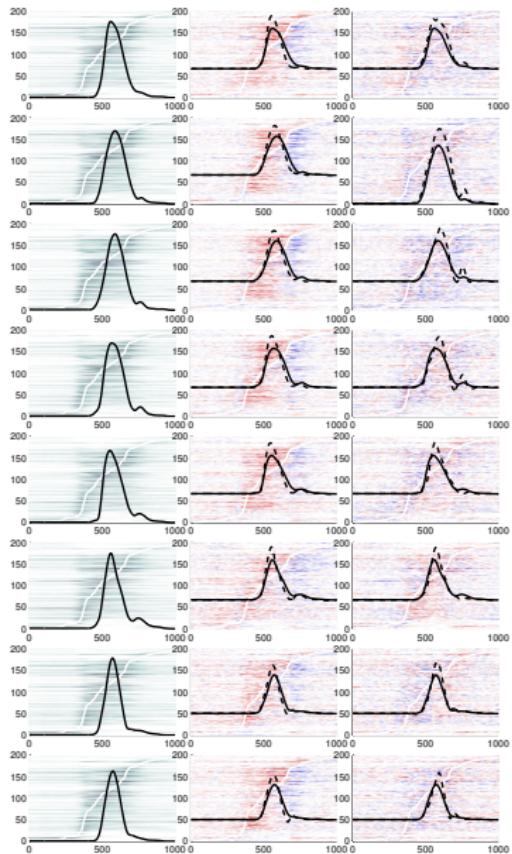
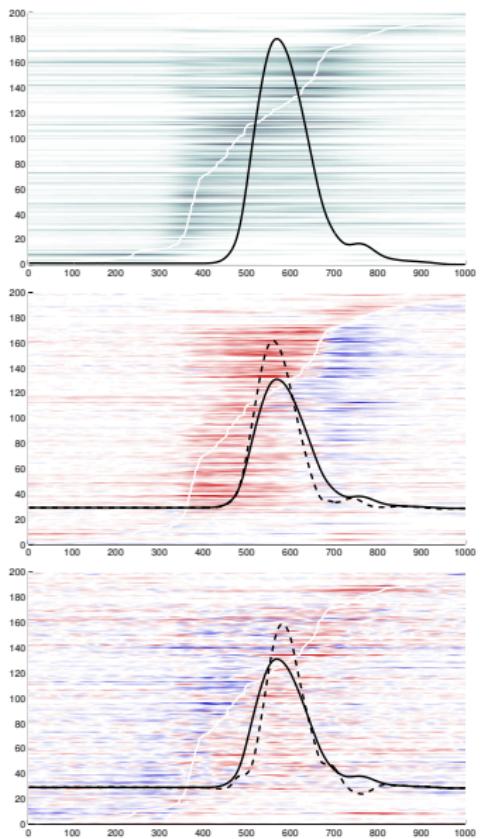
$$(\mathbf{M}_t, \mathbf{R}_t) = \operatorname{argmax} \sum_c \sum_n \underbrace{\left( \sum_t \mathbf{M}_t^\top \delta\mathbf{m}_t^{n(c)} \right)}_{\text{matrix dot products}} \underbrace{\left( \sum_t \mathbf{R}_t^\top \delta\mathbf{r}_t^{n(c)} \right)}_{\text{matrix dot products}}$$

## CVA: speed profile



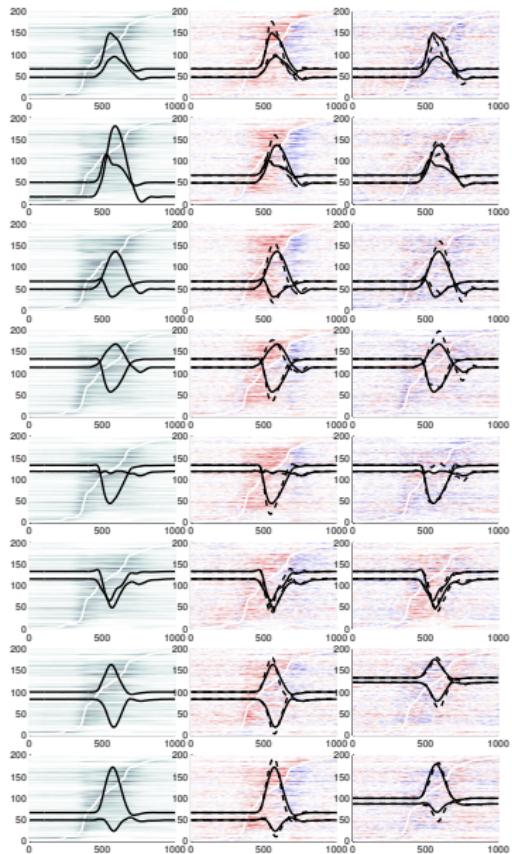
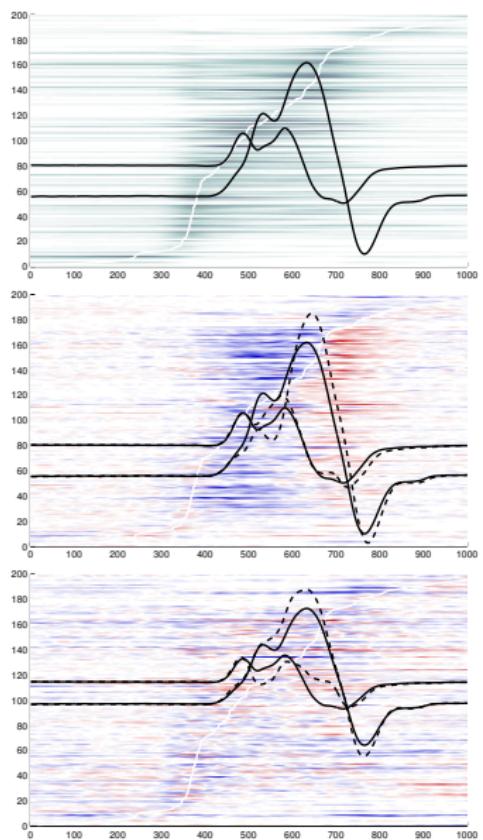
# CVA: speed profile aligned to movement start

CVA to hspeed: Monkey H; aligned rt5



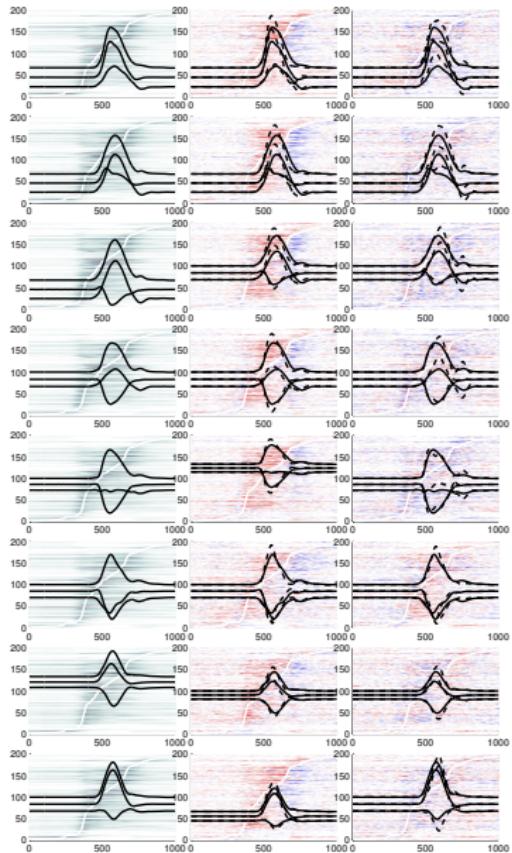
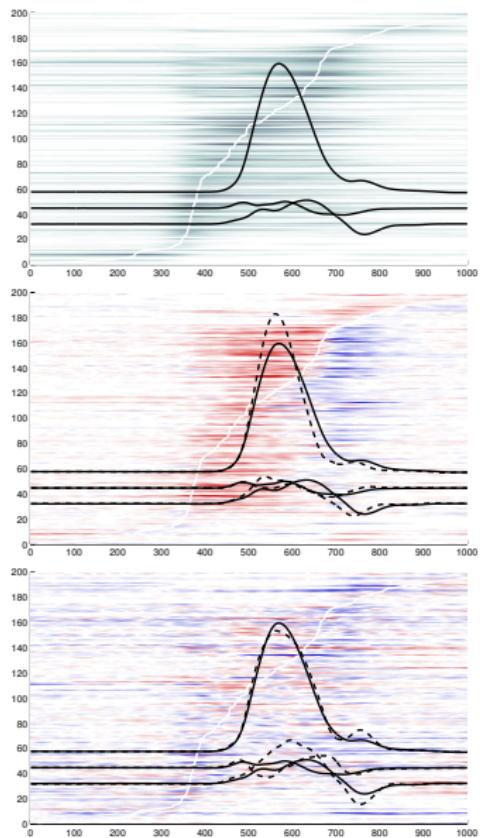
# CVA: velocity profile aligned to movement start

CVA to hhvelo vhvelo: Monkey H; aligned rt5



# CVA: speed and velocity aligned to movement start

CVA to hspeed hhvelo vhvelo: Monkey H; aligned rt5



## **Nonlinear manifold methods**

## Nonlinear Dimensionality Reduction

We can see matrix factorisation methods as performing linear dimensionality reduction.

There are many ways to generalise PCA and FA to deal with data which lie on a nonlinear manifold:

- ▶ Nonlinear autoencoders
- ▶ Generative topographic mappings (GTM) and Kohonen self-organising maps (SOM)
- ▶ Multi-dimensional scaling (MDS)
- ▶ Kernel PCA (based on MDS representation)
- ▶ Isomap
- ▶ Locally linear embedding (LLE)
- ▶ Stochastic Neighbour Embedding
- ▶ Gaussian Process Latent Variable Models (GPLVM)

## Another view of PCA: matching inner products

We have viewed PCA as providing a decomposition of the covariance or scatter matrix  $S$ . We obtain similar results if we approximate the Gram matrix:

$$\text{minimise } \mathcal{E} = \sum_{ij} (G_{ij} - \mathbf{y}_i \cdot \mathbf{y}_j)^2$$

for  $\mathbf{y} \in \mathbb{R}^k$ .

That is, look for a  $k$ -dimensional embedding in which dot products (which depend on lengths, and angles) are preserved as well as possible.

We will see that this is also equivalent to preserving distances between points.

## Another view of PCA: matching inner products

Consider the eigendecomposition of  $G$ :

$$G = U \Lambda U^T \quad \text{arranged so} \quad \lambda_1 \geq \cdots \geq \lambda_m \geq 0$$

The best rank- $k$  approximation  $G \approx Y^T Y$  is given by:

$$\begin{aligned} Y^T &= [U]_{1:m, 1:k} [\Lambda^{1/2}]_{1:k, 1:k}; \\ &= [U \Lambda^{1/2}]_{1:m, 1:k} \end{aligned}$$

$$Y = [\Lambda^{1/2} U^T]_{1:k, 1:m}$$

$$\left[ \begin{array}{c} \sqrt{\lambda_1} \mathbf{u}_1^T \\ \sqrt{\lambda_2} \mathbf{u}_2^T \\ \vdots \\ \sqrt{\lambda_k} \mathbf{u}_k^T \\ \vdots \\ \sqrt{\lambda_m} \mathbf{u}_m^T \end{array} \right]$$

## Another view of PCA: matching inner products

Consider the eigendecomposition of  $G$ :

$$G = U \Lambda U^T \quad \text{arranged so} \quad \lambda_1 \geq \dots \geq \lambda_m \geq 0$$

The best rank- $k$  approximation  $G \approx Y^T Y$  is given by:

$$\begin{aligned} Y^T &= [U]_{1:m, 1:k} [\Lambda^{1/2}]_{1:k, 1:k}; \\ &= [U \Lambda^{1/2}]_{1:m, 1:k} \end{aligned}$$

$$Y = [\Lambda^{1/2} U^T]_{1:k, 1:m}$$

$$Y^T = \left[ \begin{array}{c|c|c|c} & \sqrt{\lambda_1} \mathbf{u}_1^T & & \\ \hline \mathbf{y}_1 & & \sqrt{\lambda_2} \mathbf{u}_2^T & \\ \hline & & \vdots & \\ \hline \mathbf{y}_2 & & \sqrt{\lambda_k} \mathbf{u}_k^T & \\ \hline & & \vdots & \\ \hline \mathbf{y}_m & & \sqrt{\lambda_m} \mathbf{u}_m^T & \end{array} \right]$$

## Another view of PCA: matching inner products

Consider the eigendecomposition of  $G$ :

$$G = U \Lambda U^T \quad \text{arranged so} \quad \lambda_1 \geq \cdots \geq \lambda_m \geq 0$$

The best rank- $k$  approximation  $G \approx Y^T Y$  is given by:

$$\begin{aligned} Y^T &= [U]_{1:m, 1:k} [\Lambda^{1/2}]_{1:k, 1:k}; \\ &= [U \Lambda^{1/2}]_{1:m, 1:k} \end{aligned}$$

$$Y = [\Lambda^{1/2} U^T]_{1:k, 1:m}$$

$$Y^T = [U]_{1:m, 1:k} [\Lambda^{1/2}]_{1:k, 1:k} = [U \Lambda^{1/2}]_{1:m, 1:k} = [\Lambda^{1/2} U^T]_{1:k, 1:m}$$

The same operations can be performed on the kernel Gram matrix  $\Rightarrow$  Kernel PCA.

## Multidimensional Scaling

Suppose all we were given were distances or symmetric “dissimilarities”  $\Delta_{ij}$ .

$$\Delta = \begin{bmatrix} 0 & \Delta_{12} & \Delta_{13} & \Delta_{14} \\ \Delta_{12} & 0 & \Delta_{23} & \Delta_{24} \\ \Delta_{13} & \Delta_{23} & 0 & \Delta_{34} \\ \Delta_{14} & \Delta_{24} & \Delta_{34} & 0 \end{bmatrix}$$

**Goal:** Find vectors  $\mathbf{y}_i$  such that  $\|\mathbf{y}_i - \mathbf{y}_j\| \approx \Delta_{ij}$ .

This is called **Multidimensional Scaling (MDS)**.

## Metric MDS

Assume the dissimilarities represent Euclidean distances between points in some high-D space.

$$\Delta_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\| \text{ with } \sum_i \mathbf{x}_i = \mathbf{0}.$$

We have:

$$\begin{aligned}\Delta_{ij}^2 &= \|\mathbf{x}_i\|^2 + \|\mathbf{x}_j\|^2 - 2\mathbf{x}_i \cdot \mathbf{x}_j \\ \sum_k \Delta_{ik}^2 &= m\|\mathbf{x}_i\|^2 + \sum_k \|\mathbf{x}_k\|^2 - \mathbf{0} \\ \sum_k \Delta_{kj}^2 &= \sum_k \|\mathbf{x}_k\|^2 + m\|\mathbf{x}_j\|^2 - \mathbf{0} \\ \sum_{kl} \Delta_{kl}^2 &= 2m \sum_k \|\mathbf{x}_k\|^2\end{aligned}$$

$$\Rightarrow G_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j = \frac{1}{2} \left( \frac{1}{m} \sum_k (\Delta_{ik}^2 + \Delta_{kj}^2) - \frac{1}{m^2} \sum_{kl} \Delta_{kl}^2 - \Delta_{ij}^2 \right)$$

## Metric MDS and eigenvalues

We will actually minimize the error in the dot products:

$$\mathcal{E} = \sum_{ij} (G_{ij} - \mathbf{y}_i \cdot \mathbf{y}_j)^2$$

As in PCA, this is given by the top slice of the eigenvector matrix.

$$\left[ \begin{array}{c} \sqrt{\lambda_1} \mathbf{u}_1^T \\ \sqrt{\lambda_2} \mathbf{u}_2^T \\ \vdots \\ \sqrt{\lambda_k} \mathbf{u}_k^T \\ \vdots \\ \sqrt{\lambda_m} \mathbf{u}_m^T \end{array} \right]$$

$\mathbf{y}_1 \quad \mathbf{y}_2 \quad \cdots \quad \mathbf{y}_m$

## Interpreting MDS

$$G = \frac{1}{2} \left( \frac{1}{m} (\Delta^2 \mathbf{1} + \mathbf{1} \Delta^2) - \Delta^2 - \frac{1}{m^2} \mathbf{1}^\top \Delta^2 \mathbf{1} \right)$$

$$G = U \Lambda U^\top; \quad Y = [\Lambda^{1/2} U^\top]_{1:k, 1:m}$$

( $\mathbf{1}$  is a matrix of ones.)

- ▶ **Eigenvectors.** Ordered, scaled and truncated to yield low-dimensional embedded points  $\mathbf{y}_i$ .
- ▶ **Eigenvalues.** Measure how much each dimension contributes to dot products.
- ▶ **Estimated dimensionality.** Number of significant (nonnegative – negative possible if  $\Delta_{ij}$  are not metric) eigenvalues.

## MDS and PCA

### Dual matrices:

$$S = \frac{1}{m} XX^T \quad \text{scatter matrix} \quad (n \times n)$$

$$G = X^T X \quad \text{Gram matrix} \quad (m \times m)$$

- ▶ **Same eigenvalues** up to a constant factor.
- ▶ **Equivalent on metric data**, but MDS can run on non-metric dissimilarities.
- ▶ **Computational cost** is different.
  - ▶ PCA:  $O((m + k)n^2)$
  - ▶ MDS:  $O((n + k)m^2)$

## Non-metric MDS

MDS can be generalised to permit a monotonic mapping:

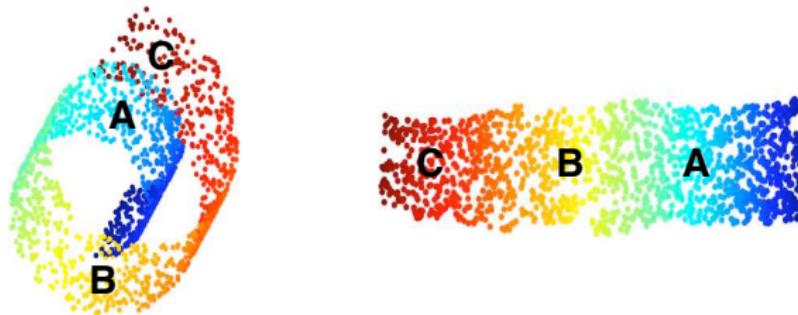
$$\Delta_{ij} \rightarrow g(\Delta_{ij}),$$

even if this violates metric rules (like the triangle inequality).

This can introduce a non-linear warping of the manifold.

But

**Rank ordering of Euclidean distances is  
NOT preserved in “manifold learning”.**

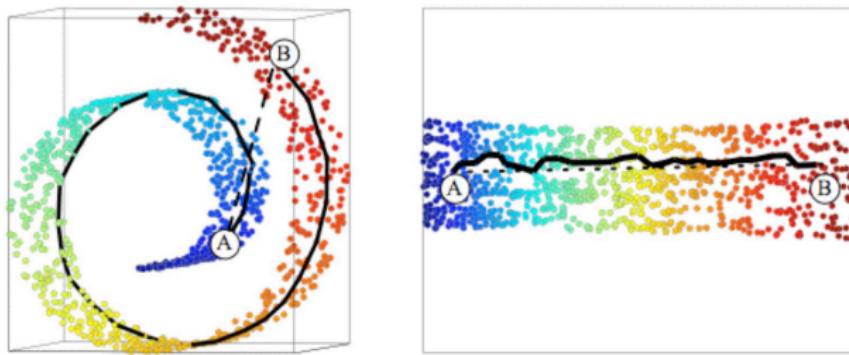


$$d(A,C) < d(A,B)$$

$$d(A,C) > d(A,B)$$

## Isomap

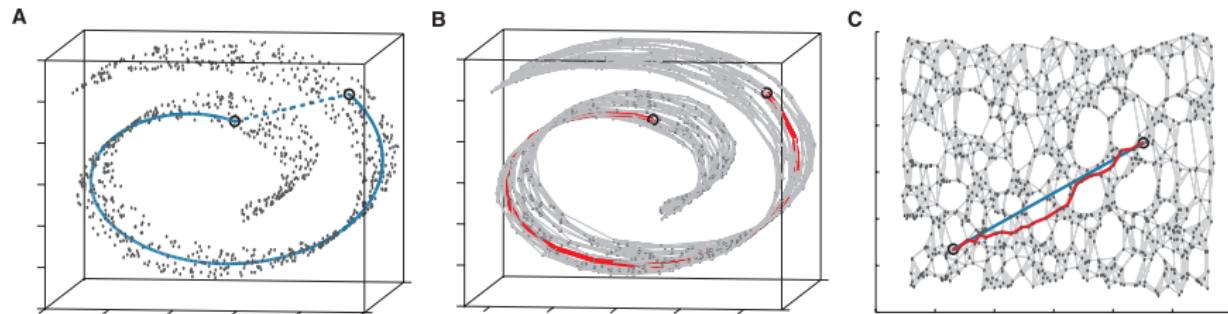
**Idea:** try to trace distance along the manifold. Use geodesic instead of (transformed) Euclidean distances in MDS.



- ▶ preserves local structure
- ▶ estimates “global” structure
- ▶ preserves information (MDS)

## Stages of Isomap

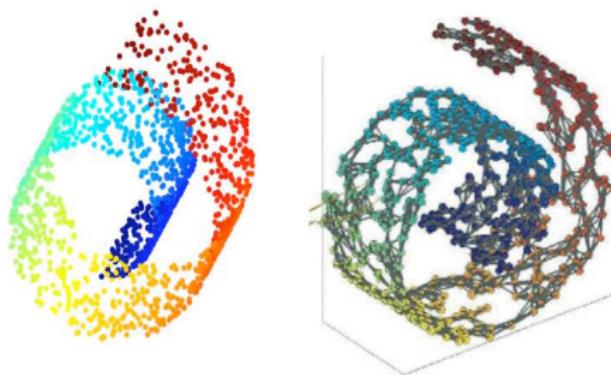
1. Identify neighbourhoods around each point (local points, assumed to be local on the manifold). Euclidean distances are preserved within a neighbourhood.
2. For points outside the neighbourhood, estimate distances by hopping between points within neighbourhoods.
3. Embed using MDS.



## Step 1: Adjacency graph

First we construct a graph linking each point to its neighbours.

- ▶ vertices represent input points
- ▶ undirected edges connect neighbours (weight = Euclidean distance)



Forms a discretised approximation to the submanifold, assuming:

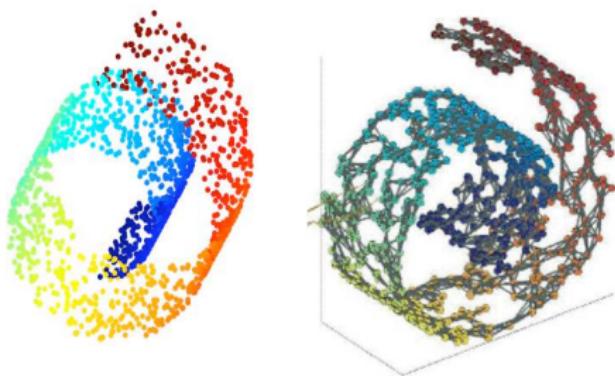
- ▶ Graph is singly-connected.
- ▶ Graph neighborhoods reflect manifold neighborhoods. No “short cuts”.

**Defining the neighbourhood** is critical:  $k$ -nearest neighbours, inputs within a ball of radius  $r$ , prior knowledge.

## Step 2: Geodesics

Estimate distances by shortest path in graph.

$$\Delta_{ij} = \min_{\text{path}(\mathbf{x}_i, \mathbf{x}_j)} \left\{ \sum_{e_i \in \text{path}(\mathbf{x}_i, \mathbf{x}_j)} \delta_i \right\}$$

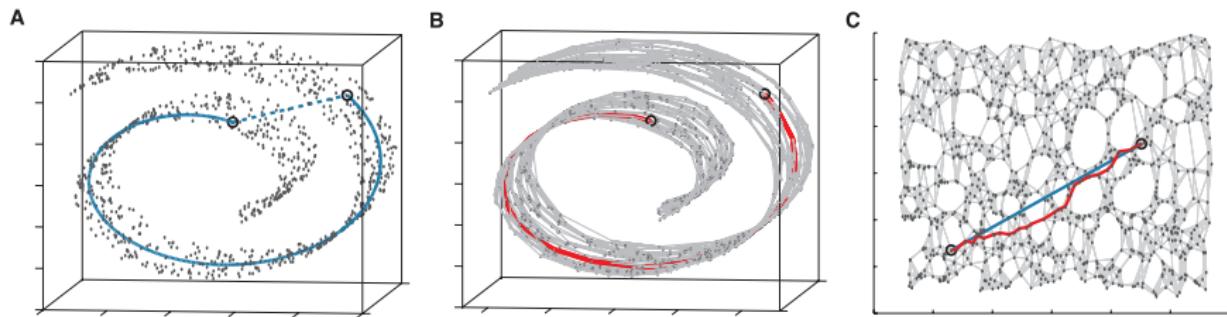


- ▶ Standard graph problem. Solved by Dijkstra's algorithm (and others).
- ▶ Better estimates for denser sampling.
- ▶ Short cuts very dangerous (“average” path distance?) .

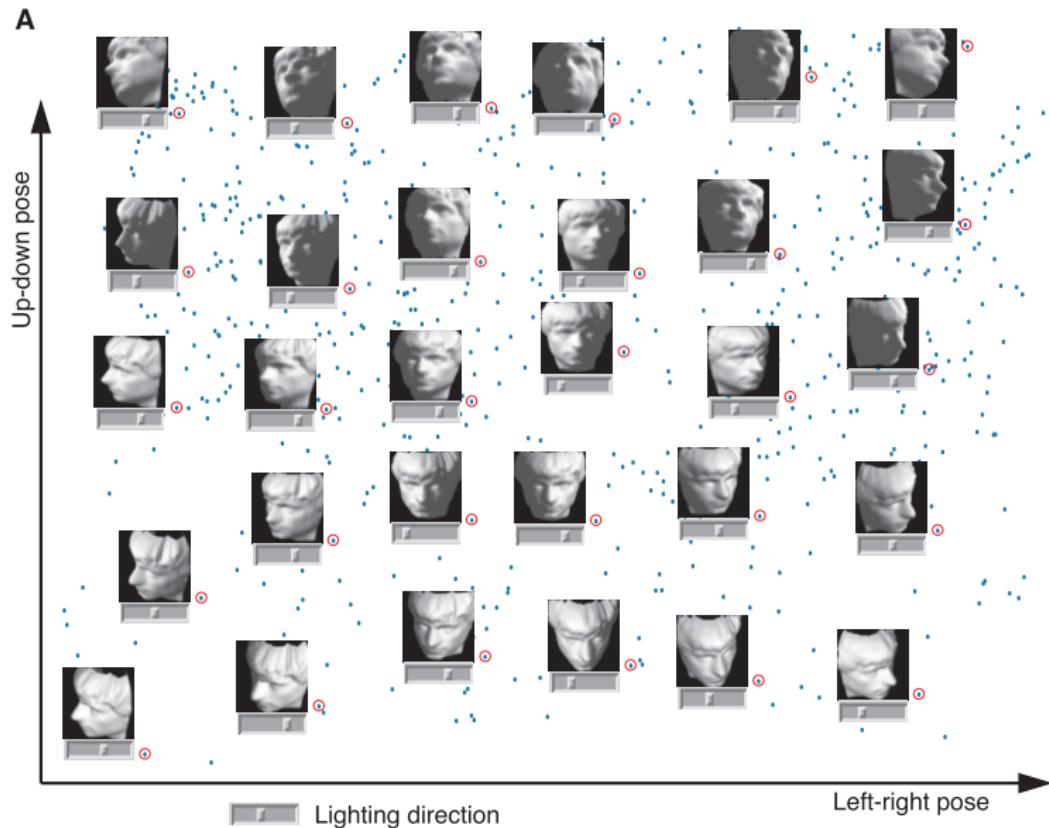
## Step 3: Embed

Embed using metric MDS (path distances obey the triangle inequality)

- ▶ Eigenvectors of Gram matrix yield low-dimensional embedding.
- ▶ Number of significant eigenvalues estimates dimensionality.



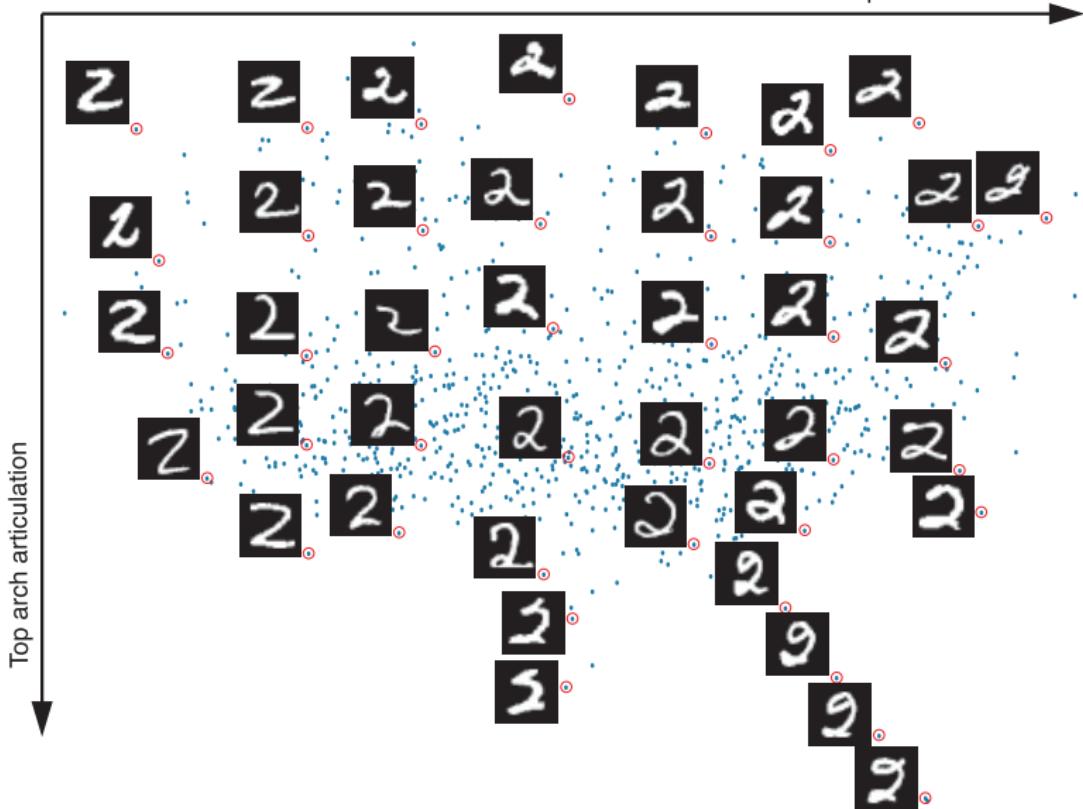
# Isomap example 1



## Isomap example 2

B

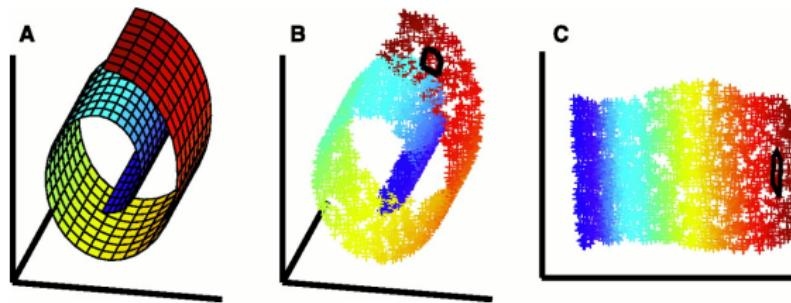
Bottom loop articulation



## Locally Linear Embedding (LLE)

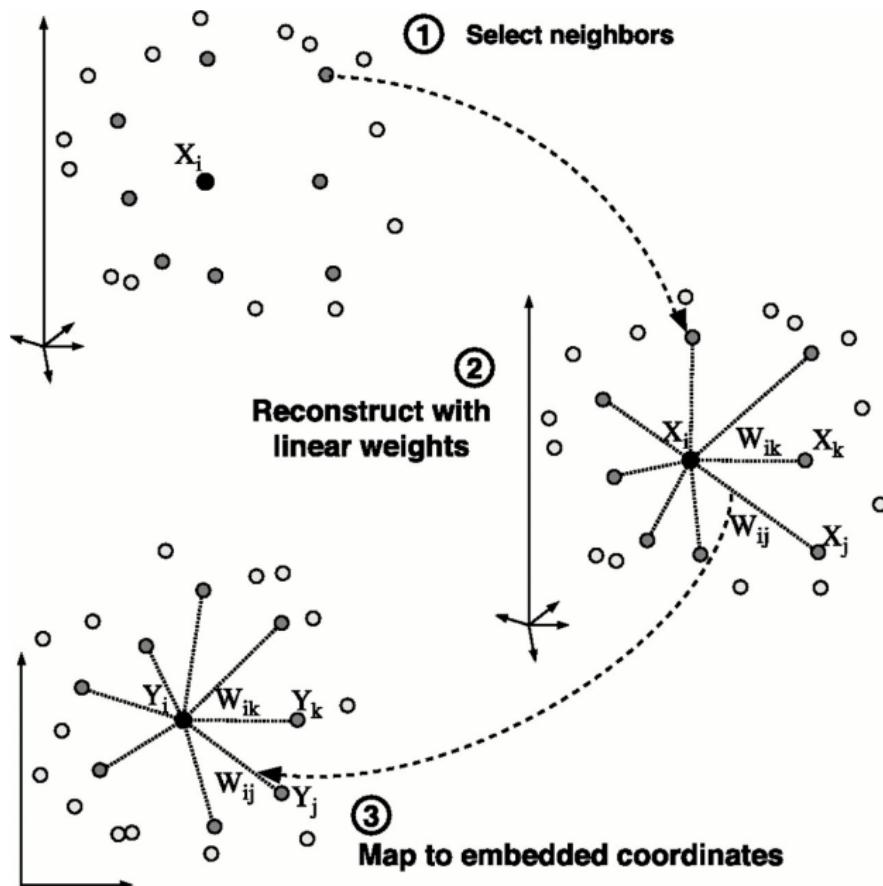
MDS and isomap preserve local and global (estimated, for isomap) **distances**. PCA preserves local and global **structure**.

**Idea:** estimate local (linear) structure of manifold. Preserve this as well as possible.



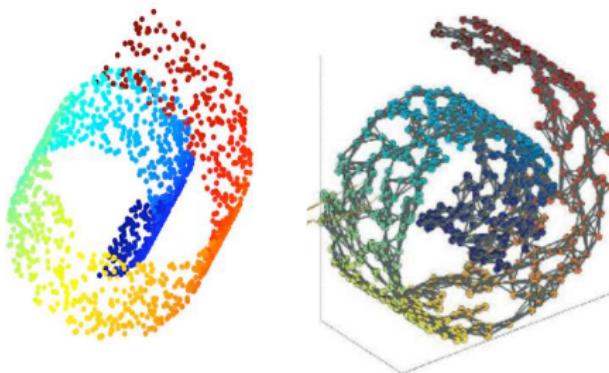
- ▶ preserves local structure (not just distance)
- ▶ not explicitly global
- ▶ preserves only local information

## Stages of LLE



## Step 1: Neighbourhoods

Just as in isomap, we first define neighbouring points for each input. Equivalent to the isomap graph, but we won't need the graph structure.



Forms a discretised approximation to the submanifold, assuming:

- ▶ Graph is singly-connected — although will “work” if not.
- ▶ Neighborhoods reflect manifold neighborhoods. No “short cuts”.

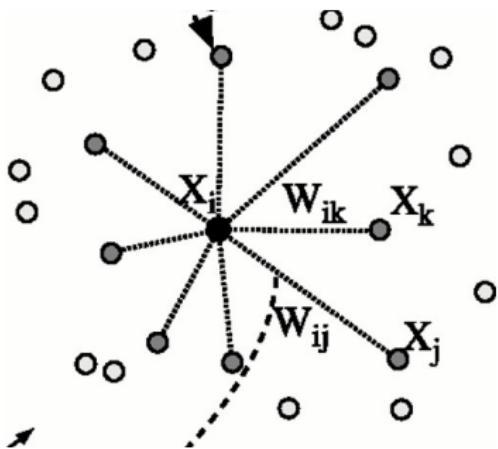
**Defining the neighbourhood** is critical:  $k$ -nearest neighbours, inputs within a ball of radius  $r$ , prior knowledge.

## Step 2: Local weights

Estimate local weights to minimize error

$$\Phi(W) = \sum_i \left\| \mathbf{x}_i - \sum_{j \in \text{Ne}(i)} W_{ij} \mathbf{x}_j \right\|^2$$

$$\sum_{j \in \text{Ne}(i)} W_{ij} = 1$$



- ▶ Linear regression – under- or over-constrained depending on  $|\text{Ne}(i)|$ .
- ▶ Local structure – optimal weights are invariant to rotation, translation and scaling.
- ▶ Short cuts less dangerous (one in many).

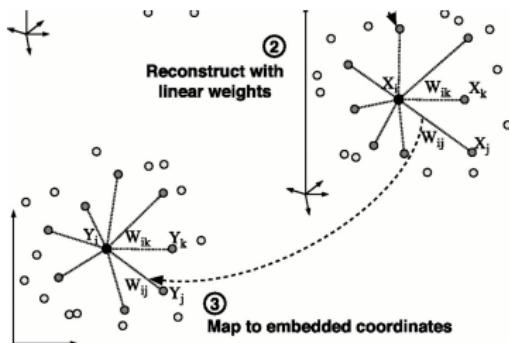
### Step 3: Embed

Minimise reconstruction errors in  $\mathbf{y}$ -space under the **same** weights:

$$\psi(Y) = \sum_i \left\| \mathbf{y}_i - \sum_{j \in \text{Ne}(i)} W_{ij} \mathbf{y}_j \right\|^2$$

subject to:

$$\sum_i \mathbf{y}_i = \mathbf{0}; \quad \sum_i \mathbf{y}_i \mathbf{y}_i^\top = ml$$



We can re-write the cost function in quadratic form:

$$\psi(Y) = \sum_{ij} \Psi_{ij} [Y^\top Y]_{ij} \text{ with } \Psi = (I - W)^\top (I - W)$$

Minimise by setting  $Y$  to equal the **bottom**  $2 \dots k + 1$  eigenvectors of  $\Psi$ . (Bottom eigenvector always  $\mathbf{1}$  – discard due to centering constraint)

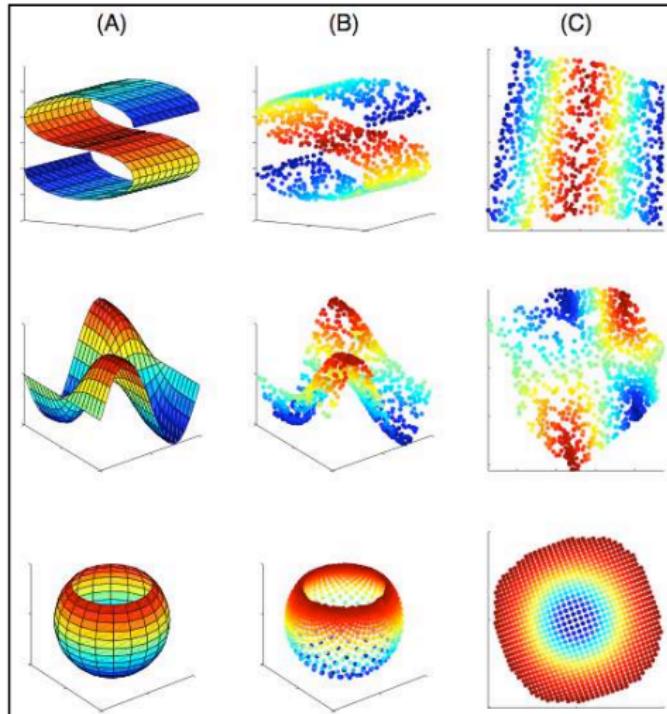
## LLE example 1

# Surfaces

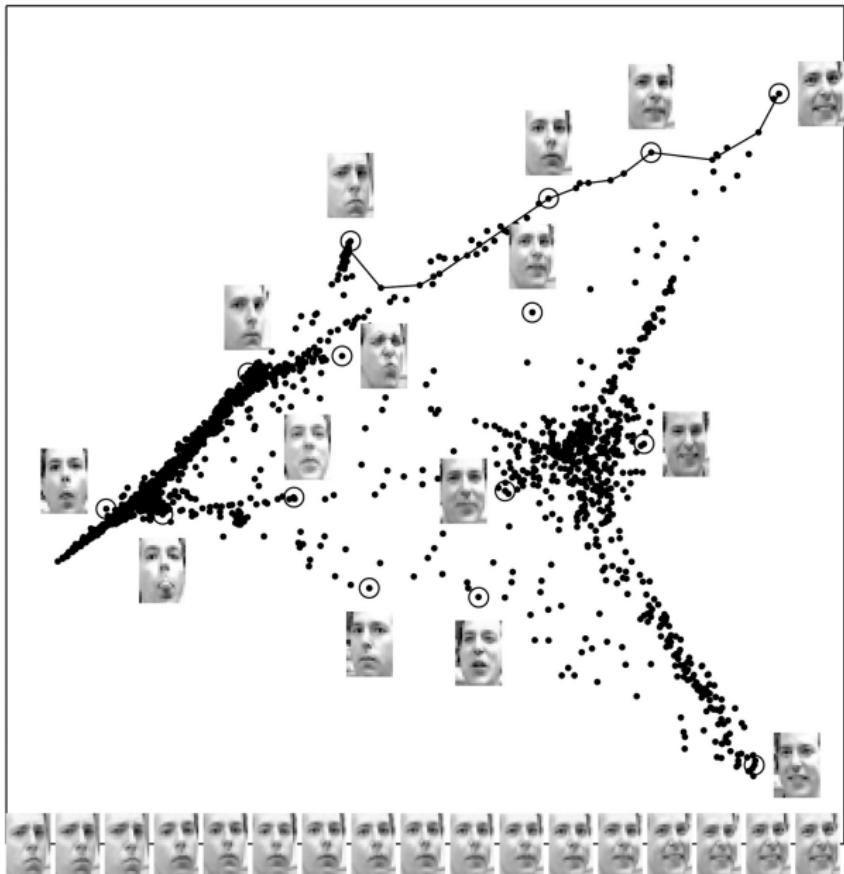
N=1000  
inputs

k=8  
nearest  
neighbors

D=3  
d=2  
dimensions

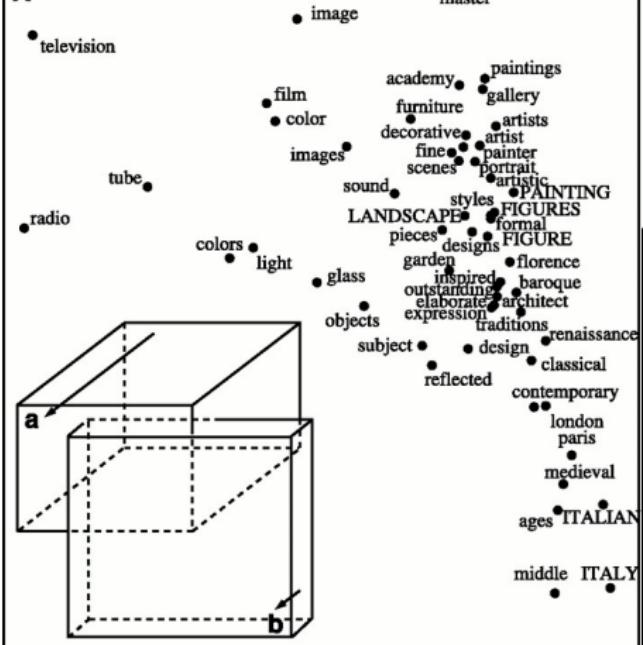


## LLE example 2

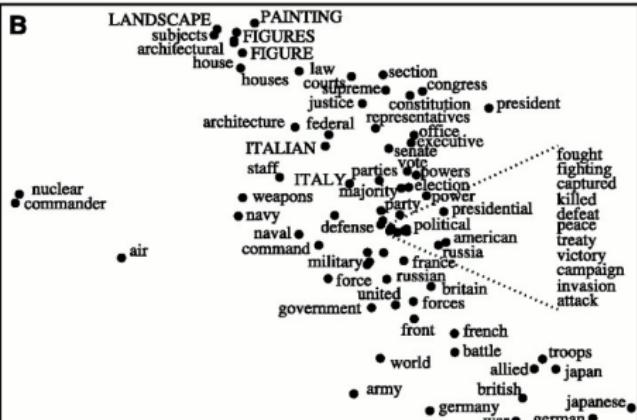


## LLE example 3

**A**



**B**



## LLE and Isomap

### Many similarities

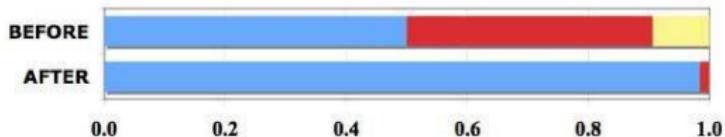
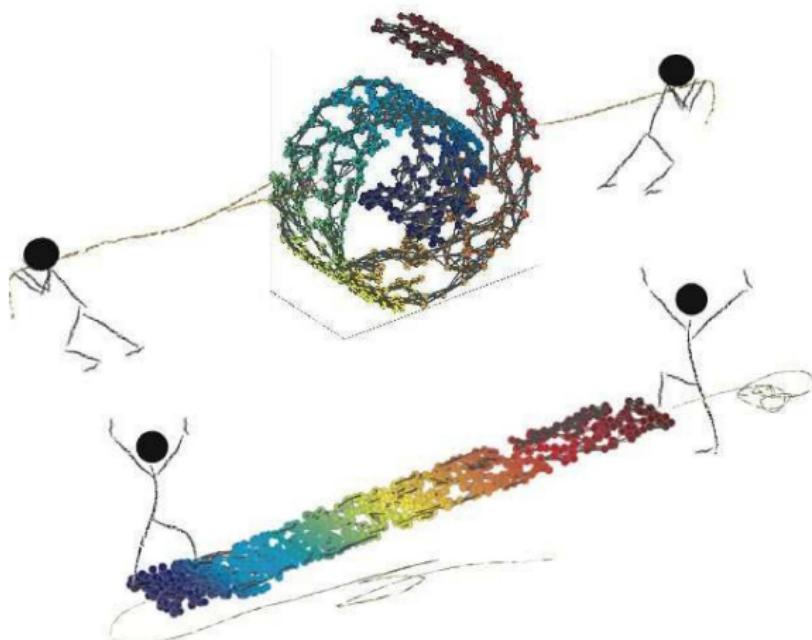
- ▶ Graph-based, spectral methods
- ▶ No local optima

### Essential differences

- ▶ LLE does not estimate dimensionality
- ▶ Isomap can be shown to be consistent; no theoretical guarantees for LLE.
- ▶ LLE diagonalises a **sparse** matrix – more efficient than isomap.
- ▶ Local weights vs. local & global distances.

## Maximum Variance Unfolding

Unfold neighbourhood graph preserving local structure.



## Maximum Variance Unfolding

Unfold neighbourhood graph preserving local structure.

1. Build the neighbourhood graph.
2. Find  $\{\mathbf{y}_i\} \subset \mathbb{R}^n$  (points in **high-D** space) with maximum variance, preserving local distances. Let  $K_{ij} = \mathbf{y}_i^\top \mathbf{y}_j$ . Then:

Maximise  $\text{Tr}[K]$  subject to:

$$\sum_{ij} K_{ij} = 0 \quad (\text{centered})$$

$$K \succeq 0 \quad (\text{positive definite})$$

$$\underbrace{K_{ii} - 2K_{ij} + K_{jj}}_{\|\mathbf{y}_i - \mathbf{y}_j\|^2} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for } j \in \text{Ne}(i) \quad (\text{locally metric})$$

This is a **semi-definite program**: convex optimisation with unique solution.

3. Embed  $\mathbf{y}_i$  in  $\mathbb{R}^k$  using linear methods (PCA/MDS).

## Stochastic Neighbour Embedding

Softer “probabilistic” notions of neighbourhood and consistency.

High-D “transition” probabilities:

$$p_{j|i} = \frac{e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}}{\sum_{k \neq i} e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_k\|^2/\sigma^2}} \quad \text{for } j \neq i, \quad p_{i|i} = 0$$

Find  $\{\mathbf{y}_i\} \subset \mathbb{R}^k$  to:

$$\text{minimise} \sum_{ij} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} \quad \text{with } q_{j|i} = \frac{e^{-\frac{1}{2}\|\mathbf{y}_i - \mathbf{y}_j\|^2}}{\sum_{k \neq i} e^{-\frac{1}{2}\|\mathbf{y}_i - \mathbf{y}_k\|^2}}.$$

Nonconvex optimisation is initialisation dependent.

Scale  $\sigma$  plays a similar role to neighbourhood definition:

- ▶ Fixed  $\sigma$ : resembles a fixed-radius ball.
- ▶ Choose  $\sigma_i$  to maintain consistent entropy in  $p_{j|i}$  of  $\log_2 k$ : similar to  $k$ -nearest neighbours.

## SNE variants

- ▶ Symmetrise probabilities ( $p_{ij} = p_{ji}$ )

$$p_{ij} = \frac{e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2}}{\sum_{k \neq i} e^{-\frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_k\|^2/\sigma^2}} \quad \text{for } j \neq i$$

- ▶ Gaussian Process Latent Variable Models. Lawrence. Advances in Neural Information Processing Systems, 2004.  
Define  $q_{ij}$  analogously, optimise joint KL.

- ▶ Heavy-tailed embedding distributions allow embedding to lower dimensions than true manifold:

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|\mathbf{y}_k - \mathbf{y}_i\|^2)^{-1}}$$

Student-t distribution defines “**t-SNE**”.

Focus is on visualisation, rather than manifold discovery.

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\mathbf{y}_i | \mathbf{x}_i, \Lambda \sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I)$$

$$\mathbf{x}_i \sim \mathcal{N}(0, I)$$

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\begin{aligned}\mathbf{y}_i | \mathbf{x}_i, \Lambda &\sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I) \\ \mathbf{x}_i &\sim \mathcal{N}(0, I)\end{aligned}$$

Usually: compute posterior over  $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ , maximizing likelihood over  $\Lambda$ .

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\begin{aligned}\mathbf{y}_i | \mathbf{x}_i, \Lambda &\sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I) \\ \mathbf{x}_i &\sim \mathcal{N}(0, I)\end{aligned}$$

Usually: compute posterior over  $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ , maximizing likelihood over  $\Lambda$ .

Suppose we know the values of the latent  $X$ , then we can integrate out  $\Lambda$  (c.f. linear regression), giving a conditional probability of  $Y = [\mathbf{y}_1 \dots \mathbf{y}_N]^\top$ :

$$\begin{aligned}\Lambda &\sim \mathcal{N}(0, \alpha^{-1} I) \\ p(Y|X) &\sim |2\pi K|^{-\frac{D}{2}} \exp\left(-\frac{1}{2} \text{Tr}[K^{-1} Y Y^\top]\right) \quad K = \alpha X X^\top + \beta I\end{aligned}$$

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\begin{aligned}\mathbf{y}_i | \mathbf{x}_i, \Lambda &\sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I) \\ \mathbf{x}_i &\sim \mathcal{N}(0, I)\end{aligned}$$

Usually: compute posterior over  $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ , maximizing likelihood over  $\Lambda$ .

Suppose we know the values of the latent  $X$ , then we can integrate out  $\Lambda$  (c.f. linear regression), giving a conditional probability of  $Y = [\mathbf{y}_1 \dots \mathbf{y}_N]^\top$ :

$$\begin{aligned}\Lambda &\sim \mathcal{N}(0, \alpha^{-1} I) \\ p(Y|X) &\sim |2\pi K|^{-\frac{D}{2}} \exp\left(-\frac{1}{2} \text{Tr}[K^{-1} Y Y^\top]\right) \quad K = \alpha X X^\top + \beta I\end{aligned}$$

This is just  $D$  independent Gaussian processes, one for each dimension of  $Y$ ! Each Gaussian process describes a mapping from latent space  $\mathbf{x}$  to one dimension of  $\mathbf{y}$ .

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\begin{aligned}\mathbf{y}_i | \mathbf{x}_i, \Lambda &\sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I) \\ \mathbf{x}_i &\sim \mathcal{N}(0, I)\end{aligned}$$

Usually: compute posterior over  $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ , maximizing likelihood over  $\Lambda$ .

Suppose we know the values of the latent  $X$ , then we can integrate out  $\Lambda$  (c.f. linear regression), giving a conditional probability of  $Y = [\mathbf{y}_1 \dots \mathbf{y}_N]^\top$ :

$$\begin{aligned}\Lambda &\sim \mathcal{N}(0, \alpha^{-1} I) \\ p(Y|X) &\sim |2\pi K|^{-\frac{D}{2}} \exp\left(-\frac{1}{2} \text{Tr}[K^{-1} Y Y^\top]\right) \quad K = \alpha X X^\top + \beta I\end{aligned}$$

This is just  $D$  independent Gaussian processes, one for each dimension of  $Y$ ! Each Gaussian process describes a mapping from latent space  $\mathbf{x}$  to one dimension of  $\mathbf{y}$ .

Replacing the linear kernel with nonlinear kernels gives nonlinear mappings—nonlinear dimensionality reduction.

## Gaussian Process Latent Variable Models

Recap: probabilistic PCA

$$\begin{aligned}\mathbf{y}_i | \mathbf{x}_i, \Lambda &\sim \mathcal{N}(\Lambda \mathbf{x}_i, \beta^{-1} I) \\ \mathbf{x}_i &\sim \mathcal{N}(0, I)\end{aligned}$$

Usually: compute posterior over  $X = [\mathbf{x}_1, \dots, \mathbf{x}_N]^\top$ , maximizing likelihood over  $\Lambda$ .

Suppose we know the values of the latent  $X$ , then we can integrate out  $\Lambda$  (c.f. linear regression), giving a conditional probability of  $Y = [\mathbf{y}_1 \dots \mathbf{y}_N]^\top$ :

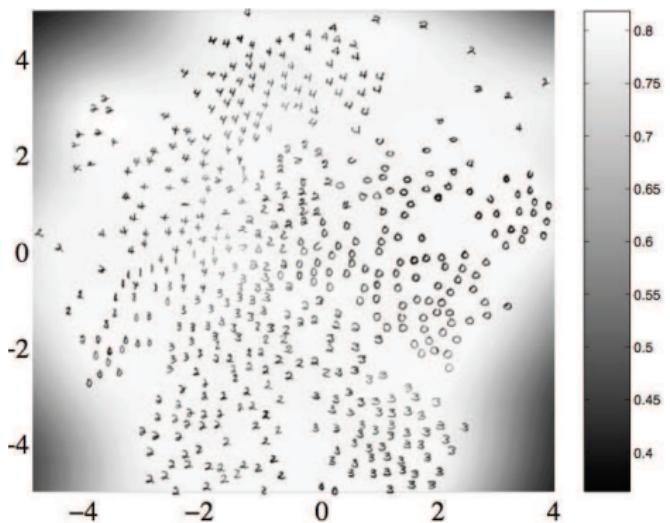
$$\begin{aligned}\Lambda &\sim \mathcal{N}(0, \alpha^{-1} I) \\ p(Y|X) &\sim |2\pi K|^{-\frac{D}{2}} \exp\left(-\frac{1}{2} \text{Tr}[K^{-1} Y Y^\top]\right) \quad K = \alpha X X^\top + \beta I\end{aligned}$$

This is just  $D$  independent Gaussian processes, one for each dimension of  $Y$ ! Each Gaussian process describes a mapping from latent space  $\mathbf{x}$  to one dimension of  $\mathbf{y}$ .

Replacing the linear kernel with nonlinear kernels gives nonlinear mappings—nonlinear dimensionality reduction.

But now dependence on  $X$  is complicated—instead of computing a posterior over  $X$  we can only find point values that maximise the likelihood (jointly with the hyperparameters).

# Gaussian Process Latent Variable Models



## Advert: Locally-linear latent variable models

Newer work from my group [on arXiv, submitted to NIPS] merges GPLVM with two ideas from the spectral manifold methods:

- ▶ the neighbourhood graph
- ▶ local linearity in tangent spaces

and provides a fully (variational) Bayesian treatment. This allows us to implement and learn priors in the latent space.