Dimensionality Reduction using Principal Component Analysis

Sina Tootoonian

The Francis Crick Institute, London
sina.tootoonian@crick.ac.uk

February 3, 2025

Section 1

Motivation

Neuroscience data arrive high dimensional

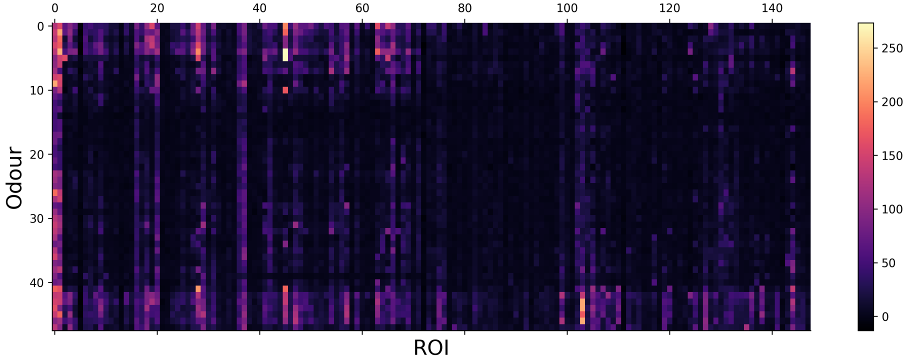


Figure: Response of 150 olfactory glomeruli to 48 odours (Tobias Ackels)

Visualization requires 2-3 dimensions

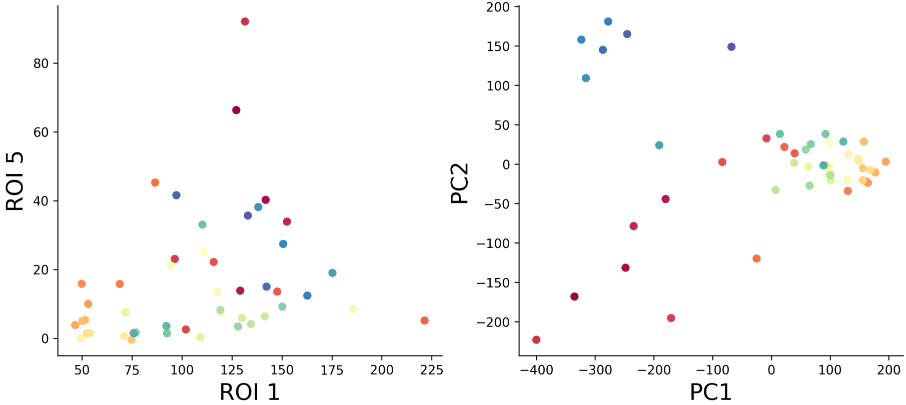


Figure: Odour responses of two ROIs, vs. two principal components.

High-D data can be embeddings of low-D manifolds

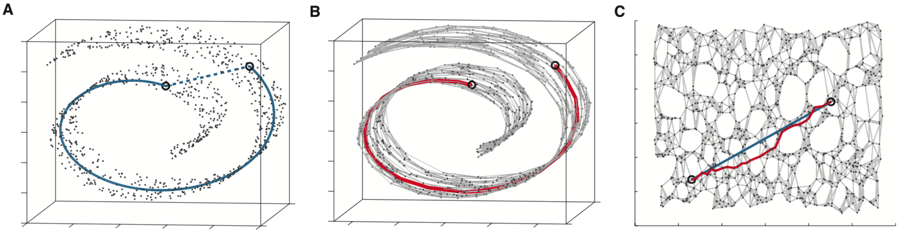


Figure: ‘Swiss roll’ dataset from Tenenbaum et al. 2000.

Data compression / approximation / denoising

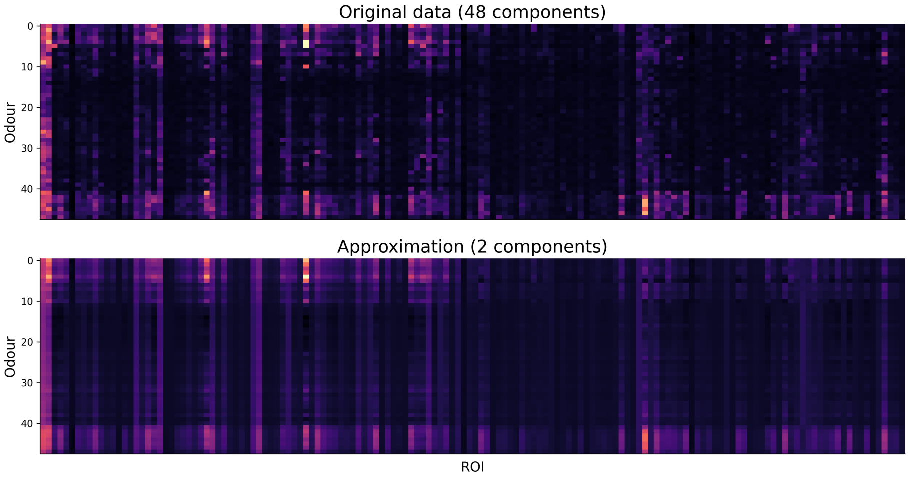


Figure: Raw odour responses and 2-component approximation.

Key idea behind PCA

Find **directions** in data space that **maximize variance**.

- What do we mean by *directions*?
- What do we mean by *variance*?
- Why is *maximizing* a good idea?

Key idea behind PCA

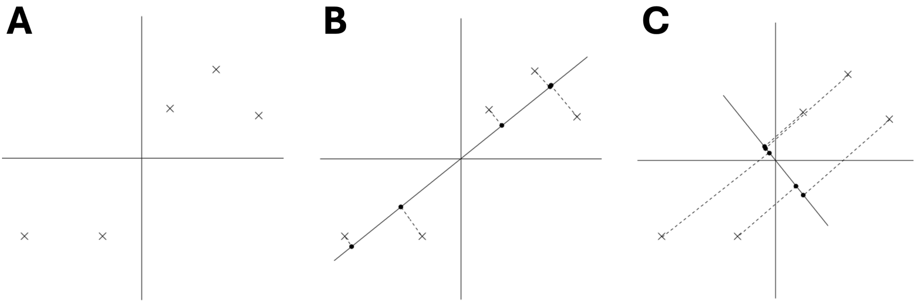


Figure: Projecting data (A) along directions that capture much (B) or little (C) variance. From CS 229.

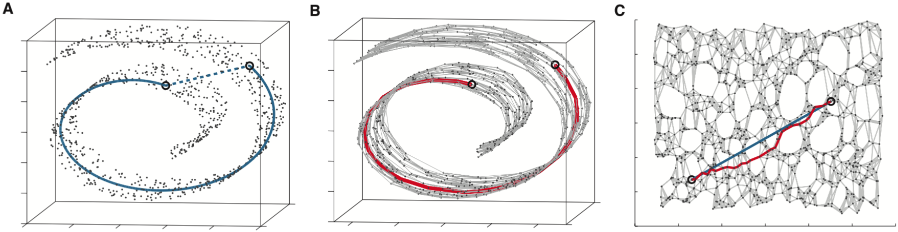
Outline

1. Single neurons
   - Single neurons as coordinates.
   - Variance of single neurons, covariance of populations of neurons.
   - Variance of single neurons as covariance along standard coordinates.
2. Pseudo-neurons
   - Other orthonormal coordinates define **pseudo-neurons**.
   - Variance of pseudo-neurons as covariance along their coordinates.
   - PCA = finding pseudo-neurons with maximum variance.
3. PCA by Singular Value Decomposition
   - Some facts about matrices.
   - Singular Value Decomposition (SVD).
   - Maximum variance directions from SVD.
4. Application to neural responses.
5. Adjacent approaches: Kernel PCA, CCA, LDA.

Section 2

Variance of Single Neurons

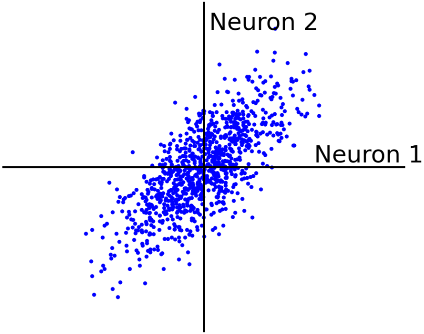
Notions of dimensionality



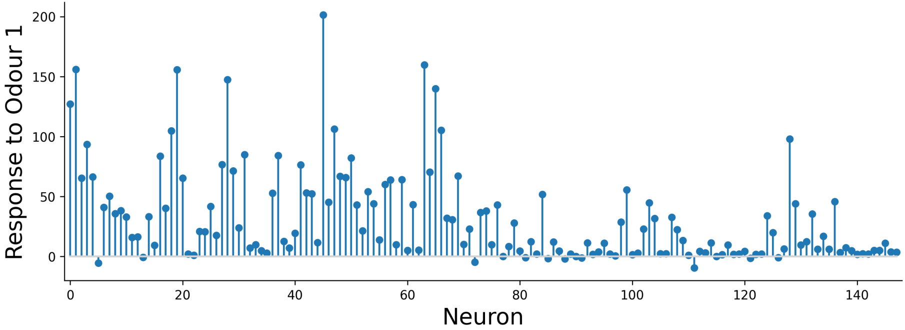
- Extrinsic dimension
- Intrinsic dimension
    - Lower because of correlations in the data
- PCA: Find the intrinsic, **linear** dimension

The importance of coordinate systems

- Laws of physics don't depend on coordinates.
- Laws of neuroscience depends on having the **right** coordinates.
  - Usually not the ones your data is recorded in.
- PCA finds coordinates that are **matched to the data**.



Standard coordinates

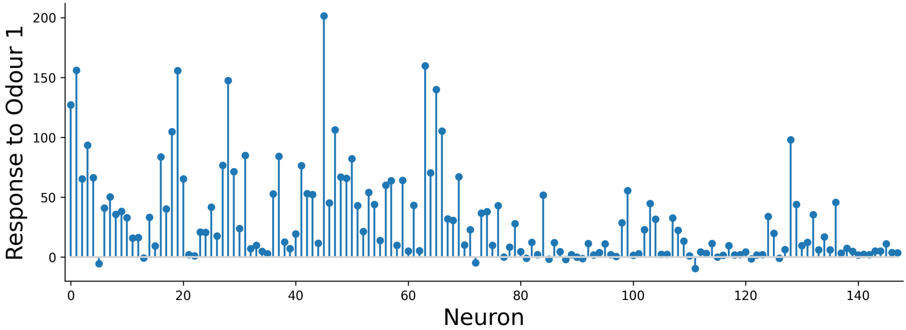


- Notice implicit coordinates:

$$\begin{aligned} \mathbf{x} &= [x_1, x_2, \dots] \\ &= x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \dots \end{aligned}$$

- Each coordinate $\mathbf{e}_1$, $\mathbf{e}_2$ corresponds to unit activity of one neuron.

Coordinates are orthonormal



$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 \ldots$$

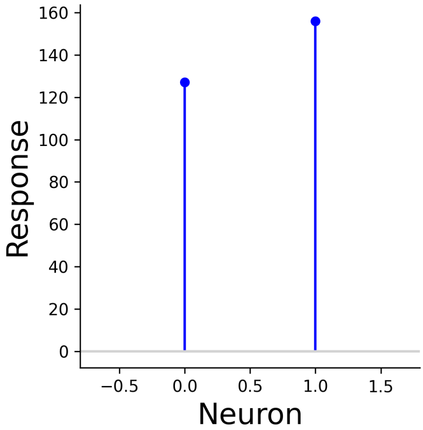
- The coordinate system $\{\mathbf{e}_1, \mathbf{e}_2, \ldots\}$ is:
  - Complete: We can represent any activity vector in it.
  - Ortho...

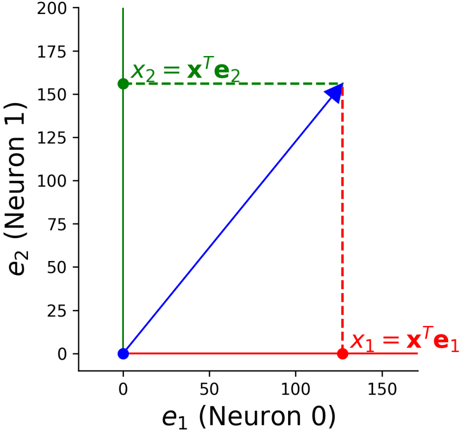
$$\mathbf{e}_1^T\mathbf{e}_2 = 0, \ldots$$

  - ...normal

$$\mathbf{e}_1^T\mathbf{e}_1 = 1, \quad \mathbf{e}_2^T\mathbf{e}_2 = 1, \ldots$$

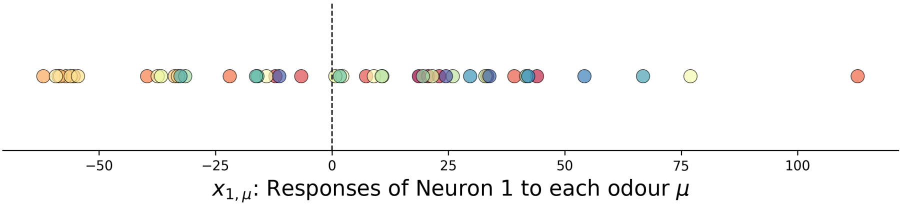
Extracting single unit responses by projection





$$x_1 = \mathbf{x}^T \mathbf{e}_1 = \mathbf{e}_1^T \mathbf{x}$$

Different ways of summarizing activity



- Mean?

$$\overline{x}_1 = \frac{1}{\text{\# stimuli}} \sum_{\mu} x_{1,\mu} = \langle x_{1,\mu} \rangle.$$

- Absolute value?

$$\overline{|x_1|} = \langle |x_{1,\mu}| \rangle$$

- Absolute value relative to mean?

$$\overline{|x_1 - \overline{x}_1|} = \langle |x_{1,\mu} - \overline{x}_1| \rangle.$$

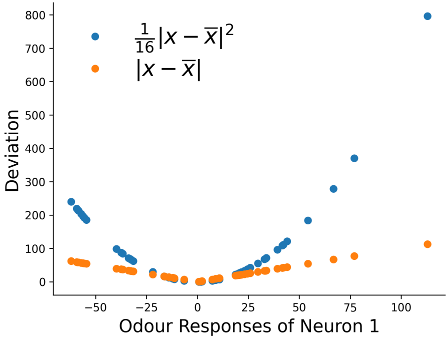
- Squared value?

$$\overline{x_1^2} = \langle x_{1,\mu}^2 \rangle$$

Variance of a single neuron

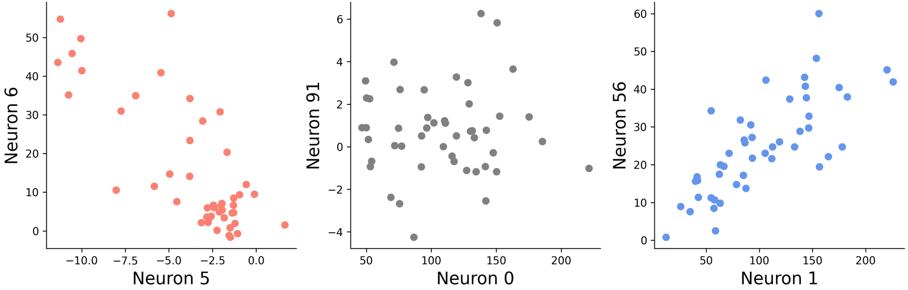
$$\mathrm{var}(x_1) = \langle (x_{1,\mu} - \overline{x}_1)^2 \rangle.$$

- Average energy relative to the mean
- Mathematically tractable ✓
- Susceptible to outliers **X**



Covariance of neural populations

- Neurons don't respond independently, but frequently **co**vary



- **Co**variance measures covariation of a neuron with another:

$$\mathrm{cov}(x_1, x_2) = \langle (x_{1,\mu} - \overline{x}_1)(x_{2,\mu} - \overline{x}_2) \rangle.$$

- Variance is covariation of a neuron with itself!

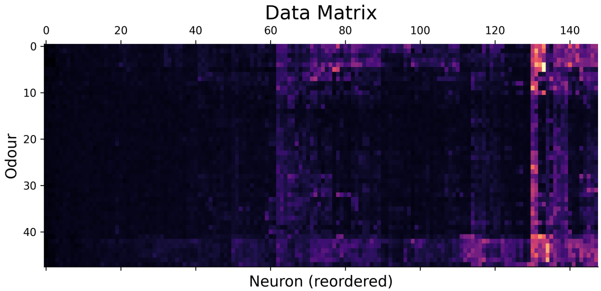
$$\begin{aligned} \mathrm{var}(x_1) &= \langle (x_{1,\mu} - \overline{x}_1)^2 \rangle \\ &= \langle (x_{1,\mu} - \overline{x}_1)(x_{1,\mu} - \overline{x}_1) \rangle. \end{aligned}$$

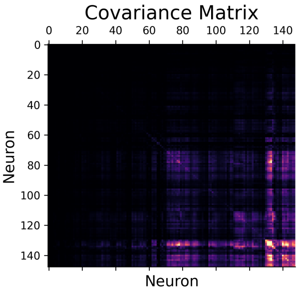
Covariance matrix

- The **covariance matrix** tabulates covariance for all pairs of neurons.

$$\mathrm{cov}(\mathbf{x}) = \begin{bmatrix} \mathrm{var}(x_1) & \mathrm{cov}(x_1, x_2) & \dots \\ \mathrm{cov}(x_1, x_2) & \mathrm{var}(x_2) & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} = \langle (\mathbf{x}_\mu - \overline{\mathbf{x}})(\mathbf{x}_\mu - \overline{\mathbf{x}})^T \rangle$$

- Diagonals have variances
- Off-diagonals have covariances





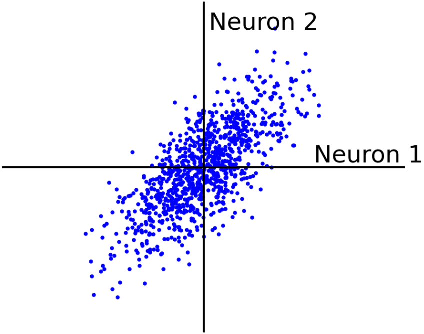
- Not just useful book keeping. . .

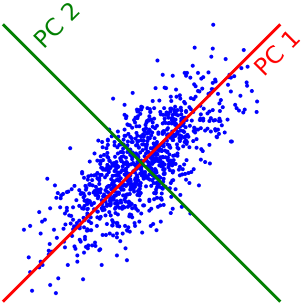
Section 3

Variance of Pseudo-Neurons

Where we're going

- Remember: PCA is about finding **directions** that maximize variance:





- The standard coordinate directions correspond to single neurons.
- The variance of single neurons is variance along these directions.
- We can define other directions as **pseudo-neurons**.
- The variance of pseudo-neurons is variance along these directions.
- PCA = find the pseudo-neurons with the largest variance.

Variance of a single neuron from covariance

- Previously we just 'took' the data $x_{1,\mu}$ for neuron 1.
- This is projecting the population vector $\mathbf{x}_\mu$ along the first coordinate:

$$x_{1,\mu} = \mathbf{x}_\mu^T \mathbf{e}_1.$$

- We can then compute the mean activity

$$\overline{x}_1 = \langle \mathbf{x}_\mu^T \mathbf{e}_1 \rangle = \langle \mathbf{x}_\mu \rangle^T \mathbf{e}_1 = \overline{\mathbf{x}}^T \mathbf{e}_1.$$

- The variance is then

$$\begin{aligned}
\mathrm{var}(x_1) &= \langle (x_{1,\mu} - \bar{\bar{x}}_1)^2 \rangle \\
&= \langle \left(\mathbf{x}_\mu^T \mathbf{e}_1 - \overline{\mathbf{x}}_\mu^T \mathbf{e}_1 \right)^2 \rangle \\
&= \langle \left((\mathbf{x}_\mu - \overline{\mathbf{x}})^T \mathbf{e}_1 \right)^2 \rangle \\
&= \langle \mathbf{e}_1^T (\mathbf{x}_\mu - \overline{\mathbf{x}})(\mathbf{x}_\mu - \overline{\mathbf{x}})^T \mathbf{e}_1 \rangle \\
&= \mathbf{e}_1^T \langle (\mathbf{x}_\mu - \overline{\mathbf{x}})(\mathbf{x}_\mu - \overline{\mathbf{x}})^T \rangle \mathbf{e}_1 \\
&= \mathbf{e}_1^T \,\mathrm{cov}(\mathbf{x}) \mathbf{e}_1.
\end{aligned}$$

- So, the variance of neuron 1 is **covariance along** $\mathbf{e}_1$.

Other orthonormal coordinates define pseudo-neurons

- Previously we described population activity in terms of standard coordinates $\mathbf{e}_1, \mathbf{e}_2, \ldots$ of neurons:

$$\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + \ldots$$

- We can describe the same activity $\mathbf{x}$ in other orthonormal coordinates $\tilde{\mathbf{e}}_1, \tilde{\mathbf{e}}_2, \ldots$ of **pseudo-neurons**:

$$\mathbf{x} = \tilde{x}_1\tilde{\mathbf{e}}_1 + \tilde{x}_2\tilde{\mathbf{e}}_2 + \ldots$$

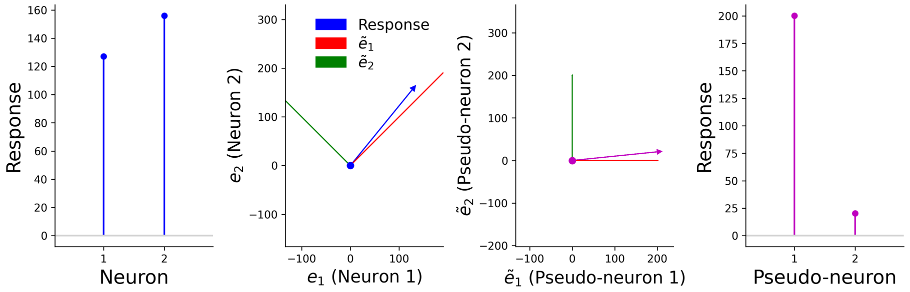


Figure: Responses of neurons and pseudo-neurons to the first odour.

Variance of a pseudo-neuron along $\mathbf{u}_1$

- Activity of the **pseudo**neuron:

$$\tilde{x}_{1,\mu} = \mathbf{x}_\mu^T \mathbf{u}_1.$$

- Mean activity of the pseudoneuron:

$$\bar{\tilde{x}}_1 = \langle \mathbf{x}_\mu^T \mathbf{u}_1 \rangle = \langle \mathbf{x}_\mu \rangle^T \mathbf{u}_1 = \overline{\mathbf{x}}^T \mathbf{u}_1.$$

- The variance is then

$$\begin{aligned}
\mathrm{var}(\tilde{x}_1) &= \langle (\tilde{x}_{1,\mu} - \bar{\tilde{x}}_1)^2 \rangle \\
&= \langle \left(\mathbf{x}_\mu^T \mathbf{u}_1 - \overline{\mathbf{x}}_\mu^T \mathbf{u}_1 \right)^2 \rangle \\
&= \langle \left((\mathbf{x}_\mu - \overline{\mathbf{x}})^T \mathbf{u}_1 \right)^2 \rangle \\
&= \langle \mathbf{u}_1^T (\mathbf{x}_\mu - \overline{\mathbf{x}})(\mathbf{x}_\mu - \overline{\mathbf{x}})^T \mathbf{u}_1 \rangle \\
&= \mathbf{u}_1^T \langle (\mathbf{x}_\mu - \overline{\mathbf{x}})(\mathbf{x}_\mu - \overline{\mathbf{x}})^T \rangle \mathbf{u}_1 \\
&= \mathbf{u}_1^T \,\mathrm{cov}(\mathbf{x}) \mathbf{u}_1.
\end{aligned}$$

- So, the variance of the pseudoneuron is **covariance along** $\mathbf{u}_1$.

Variance of *any* pseudoneuron

- Following the pattern, variance of a pseudoneuron $\tilde{x} = \mathbf{x}^T \mathbf{u}$ is

$$\mathrm{var}(\tilde{x}) = \mathbf{u}^T \,\mathrm{cov}(\mathbf{x})\mathbf{u}.$$

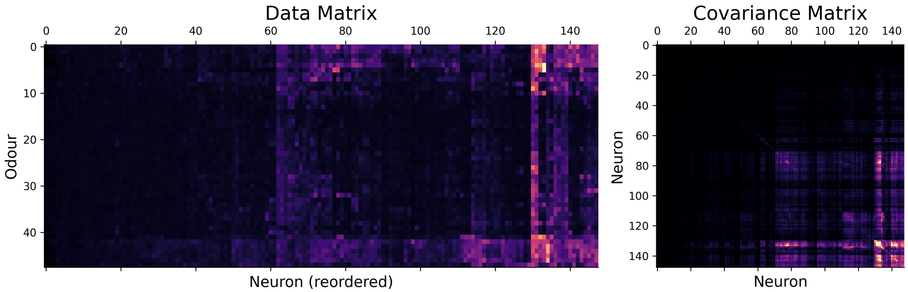
- PCA now becomes finding the $\mathbf{u}$ that maximizes this variance.
- How do we do this? By decomposing the covariance matrix!
- But first...

Section 4

Some Facts about Matrices

Matrices

- Some matrices we've already encountered:



- Data matrix (rectangular)
- Covariance matrix (square, symmetric)
  - Why is it symmetric?

Different ways to view matrices

$$\mathbf{A} = \underbrace{\begin{bmatrix} A_{11}, & A_{12}, & \dots \\ A_{21}, & A_{22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}}_{\text{Table of elements}} = \underbrace{\begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \vdots \\ \mathbf{r}_M^T \end{bmatrix}}_{\text{Stacked rows}} = \underbrace{\begin{bmatrix} \mathbf{c}_1, & \mathbf{c}_2, & \mathbf{c}_3, & \dots & \mathbf{c}_N \end{bmatrix}}_{\text{Stacked columns}}.$$

Matrix operations

- **Linearly** transform N-dimensional inputs $\mathbf{x}$ into M-dimensional outputs $\mathbf{y}$,

$$\mathbf{y} = \mathbf{A}\mathbf{x}.$$

- Can think of this element-wise:

$$y_i = \sum_{j=1}^{N} A_{ij} x_j.$$

- Can think of this as projecting $\mathbf{x}$ on each row,

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1^T \mathbf{x} \\ \mathbf{r}_2^T \mathbf{x} \\ \vdots \\ \mathbf{r}_M^T \mathbf{x} \end{bmatrix}.$$

- Can think of this as summing the columns, weighted by $\mathbf{x}$,

$$\mathbf{y} = \sum_{i=1}^{N} \mathbf{c}_i x_i.$$

Example Matrices

Name	Matrix $\mathbf{A}$	Action $\mathbf{y} = \mathbf{A}\mathbf{x}$
Zero	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathbf{y} = \mathbf{0}$
Identity	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\mathbf{y} = \mathbf{x}$
All ones	$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$	$\mathbf{y} = \begin{bmatrix} \sum_i x_i \\ \sum_i x_i \end{bmatrix}$
Uniform scaling	$\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$	$\mathbf{y} = \begin{bmatrix} kx_1 \\ kx_2 \end{bmatrix}$
Diagonal	$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$	$\mathbf{y} = \begin{bmatrix} ax_1 \\ bx_2 \end{bmatrix}$
Permutation	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\mathbf{y} = \begin{bmatrix} x_2 \\ x_1 \end{bmatrix}$
Rotation	$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$	$\mathbf{y}$ is $\mathbf{x}$ rotated by θ.

Composing transformations

- We can form complex transformations by composing simple ones.
- For example, a scaling and a rotation:

$$\mathbf{y} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling}} \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{\text{rotation}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \underbrace{\mathbf{D}\mathbf{R}}_{\mathbf{A}}\,\mathbf{x}.$$

Section 5

Singular Value Decomposition

All matrices are diagonal matrices (in the right coordinates)

- Diagonal matrices were easy to work with

$$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} ax_1 \\ bx_2 \end{bmatrix}$$

- What about an arbitrary matrix? Looks complex...

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \sum_j A_{1j} x_j \\ \sum_j A_{2j} x_j \end{bmatrix}.$$

- Surprise: Every matrix $\mathbf{A}$ is the composition of just three operations!

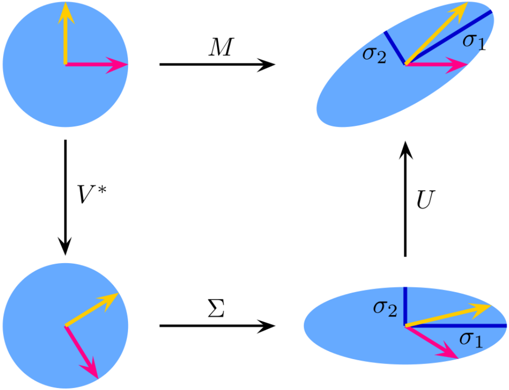
$$\mathbf{A} = \underbrace{\mathbf{U}}_{\text{rotate}} \underbrace{\mathbf{S}}_{\text{scale}} \underbrace{\mathbf{V}^T}_{\text{project}} .$$

Three parts of Singular Value Decomposition

$$\mathbf{A} = \underbrace{\mathbf{U}}_{\text{rotate}} \underbrace{\mathbf{S}}_{\text{scale}} \underbrace{\mathbf{V}^T}_{\text{project}} .$$

- Columns of $\mathbf{V}$ form orthonormal coordinates for the **input** space.
- Columns of $\mathbf{U}$ form orthonormal coordinates for the **output** space
- Diagonal matrix $\mathbf{S}$ of non-negative **singular values** apply a scaling.
- If we:
  - Use $\mathbf{V}$ coordinates for the input, and
  - Use $\mathbf{U}$ coordinates for the output, then
  - $\mathbf{A}$ is a scaling!

Three transformations in Singular Value Decomposition



$$M = U \cdot \Sigma \cdot V^*$$

Figure: Three transformations in Singular Value Decomposition (Wikipedia).

Three matrices of Singular Value Decomposition

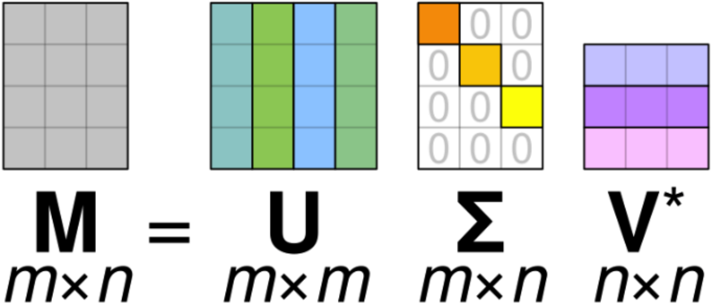


Figure: Three matrices of Singular Value Decomposition (Wikipedia).

Three steps of Singular Value Decomposition

$$\mathbf{A}\mathbf{x} = \underbrace{\mathbf{U}}_{\text{rotate}}\underbrace{\mathbf{S}}_{\text{scale}}\underbrace{\mathbf{V}^T}_{\text{project}}\mathbf{x}.$$

1. Project $\mathbf{x}$ onto the input coordinates:

$$\mathbf{V}^T\mathbf{x} = \begin{bmatrix} \mathbf{v}_1^T\mathbf{x} \\ \dots \\ \mathbf{v}_N^T\mathbf{x} \end{bmatrix} = \begin{bmatrix} \tilde{x}_1 \\ \dots \\ \tilde{x}_N \end{bmatrix}$$

2. Scale by $\mathbf{S}$:

$$\mathbf{S}\mathbf{V}^T\mathbf{x} = \begin{bmatrix} s_1 & 0 & \dots \\ 0 & s_2 & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ \dots \\ \tilde{x}_N \end{bmatrix} = \begin{bmatrix} s_1\tilde{x}_1 \\ s_2\tilde{x}_2 \\ \dots \\ s_N\tilde{x}_N \end{bmatrix}$$

3. Project out using the output coordinates

$$\mathbf{U}\mathbf{S}\mathbf{V}^T\mathbf{x} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N] \begin{bmatrix} s_1\tilde{x}_1 \\ s_2\tilde{x}_2 \\ \dots \\ s_N\tilde{x}_N \end{bmatrix} = \mathbf{u}_1 s_1\tilde{x}_1 + \mathbf{u}_2 s_2\tilde{x}_2 + \dots$$

SVD of simple matrices

Name	$\mathbf{A}$	$\mathbf{U}$	$\mathbf{s}$	$\mathbf{V}$
Zero	$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$[0, \quad 0]$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Identity	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$[1, \quad 1]$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Negation	$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$	$[1, \quad 1]$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
All ones	$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$	$[2, \quad 0]$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$
Diagonal	$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$[2, \quad 3]$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Permutation	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$[1, \quad 1]$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Rotation by θ	$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$	$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$	$[1, \quad 1]$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

SVD of covariance matrices

- Remember why we care: we're after the variance of pseudoneurons

$$\mathbf{u}^T \operatorname{cov}(\mathbf{x})\mathbf{u}.$$

- For covariance matrices, the input and output coordinates are the same

$$\operatorname{cov}(\mathbf{x}) = \mathbf{V}\mathbf{S}\mathbf{V}^T$$

- Equalizer: Inputs are analyzed in $\mathbf{V}$ coordinates and scaled.

$$\operatorname{cov}(\mathbf{x})\mathbf{u} = \sum_i \mathbf{v}_i \underbrace{s_i}_{\text{scale}} \underbrace{\mathbf{v}_i^T \mathbf{u}}_{\text{project}}$$

SVD and eigendecomposition

- All matrices have SVDs: $\mathbf{X} = \mathbf{U}\mathbf{S}\mathbf{V}^T$
- For covariance matrices, the input and output coordinates are the same: $\text{cov}(\mathbf{x}) \propto \mathbf{V}\mathbf{S}^2\mathbf{V}^T$
- Also known as the **eigendecomposition** of the covariance matrix.
- The right singular vectors $\mathbf{V}$ of the data matrix are the eigenvectors/singular vectors of the covariance matrix.
- The singular values of the data matrix and the cov. matrix are closely related:

$$\begin{aligned}\text{Singular values of cov. matrix} &= \text{Eigenvalues of the cov. matrix}\\ &\propto \textbf{Squared}\text{ sing. values of the data matrix.}\end{aligned}$$

Maximum variance direction from SVD

- We can use SVD to read-off the maximum variance direction(s) we need!
- Variance along a direction $\mathbf{u}$

$$
\begin{aligned}
\mathbf{u}^T \operatorname{cov}(\mathbf{x})\mathbf{u} &= \mathbf{u}^T \underbrace{\left(\sum_i \mathbf{v}_i s_i \mathbf{v}_i^T\right)}_{\text{SVD}} \mathbf{u} \\
&= \sum_i (\mathbf{u}^T \mathbf{v}_i) s_i (\mathbf{v}_i^T \mathbf{u}) \\
&= \sum_i s_i (\mathbf{v}_i^T \mathbf{u})^2
\end{aligned}
$$

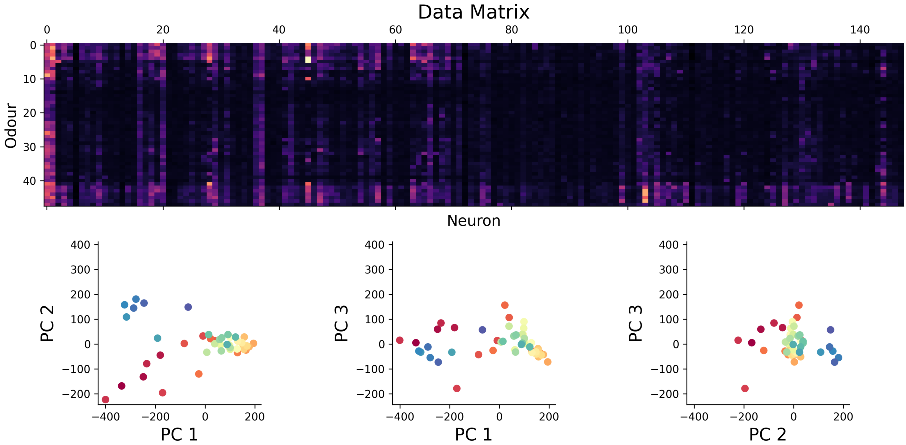
- Maximum variance direction is $\mathbf{v}_1$
- Next highest variance direction is $\mathbf{v}_2$, etc.

Section 6

Dimensionality Reduction with PCA

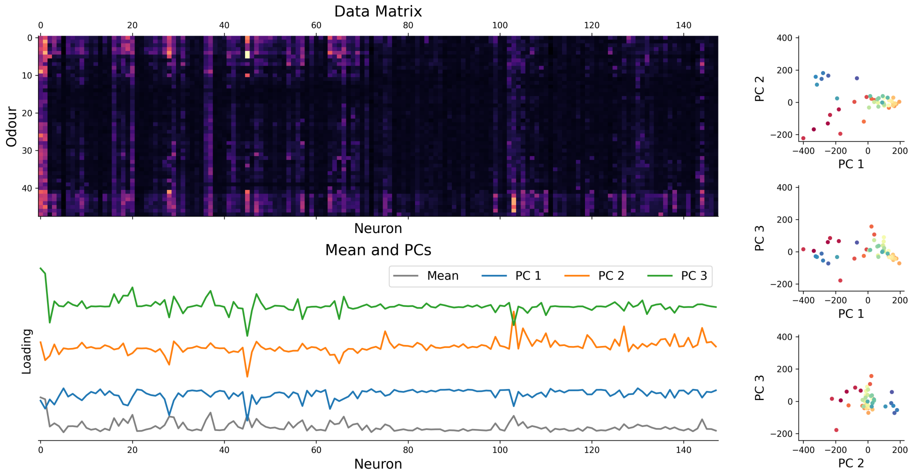
Finally: Dimension Reduction with PCA

- Project data onto maximum variance directions: $\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{V}$.



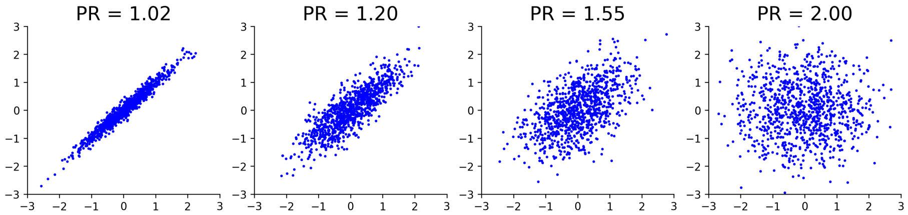
- Notice: projections are decorrelated

Examining the Principal Components



- Variances tell us energy in each direction
- Use this as a measure of dimensionality

$$\mathrm{PR} = \frac{\left(\sum_i s_i\right)^2}{\sum_i s_i^2}.$$



Approximation/Denoising with PCA

- Approximate using first K projections

$$\mathbf{x} \approx \underbrace{\sum_{i=1}^{K} (\mathbf{x}^T \mathbf{v}_i)\mathbf{v}_i}_{\text{Exact}} + \underbrace{\sum_{i=K+1}^{D} (\overline{\mathbf{x}}^T \mathbf{v}_i)\mathbf{v}_i}_{\text{Approximation}} .$$

$$\approx \overline{\mathbf{x}} + \sum_{i=1}^{K} (\mathbf{x} - \overline{\mathbf{x}})^T \mathbf{v}_i \mathbf{v}_i$$

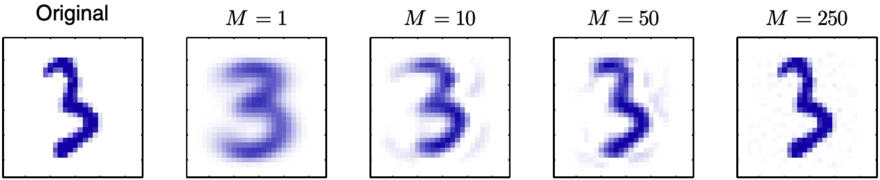


Figure: Approximating digits data using PCA (Bishop Fig 12.5)

Approximation/Denoising with PCA

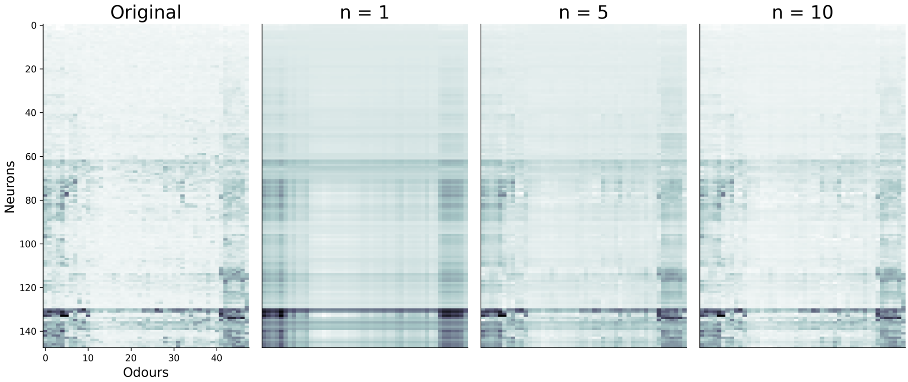


Figure: Approximating odour responses using PCA.

How many dimensions to keep?

- The singular values tell us how much variance is explained by each dimension.
- We can use this to decide how many dimensions to keep.
- **Explained variance** measures the fraction of variance explained by the first K dimensions:

$$\mathsf{EV}(K) = \frac{\sum_{i=1}^{K} s_i}{\sum_{i=1}^{D} s_i}.$$

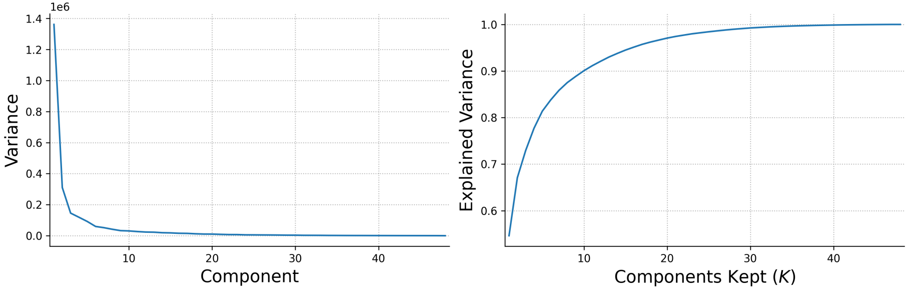


Figure: Explained variance for odour responses dataset.

Section 7

Adjacent Approaches

Exploiting nonlinearity with Kernel PCA

- PCA can be expressed in terms of similarity $k(\mathbf{x}, \mathbf{y})$ between data points.
- PCA uses linear similarity $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}$.
- Kernel PCA generalises this to allow other similarity measures.
- Nonlinear measures are sometimes appropriate.

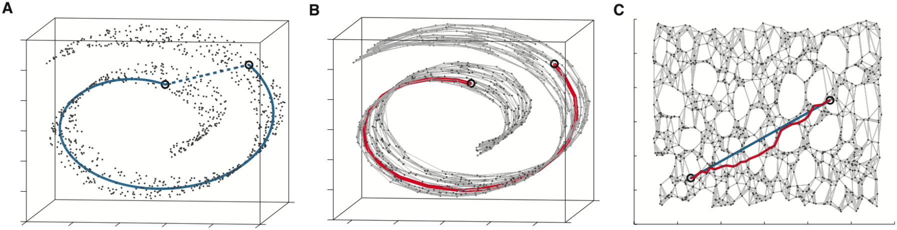
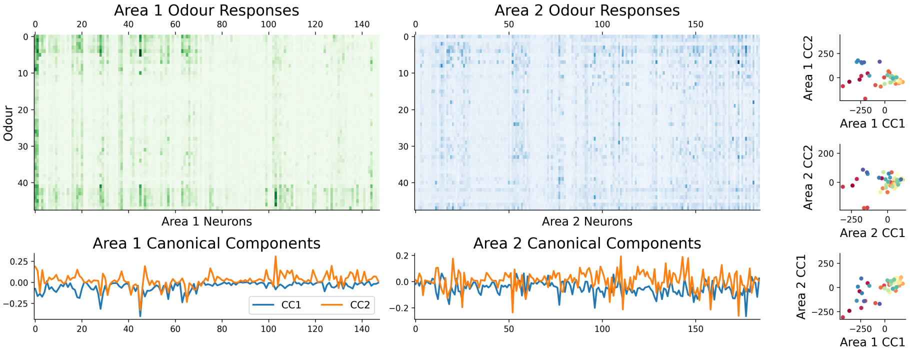


Figure: ISOMAP computes similarity as distance on the manifold.

Comparing different datasets with CCA

- PCA finds maximum variance directions in one dataset
- CCA finds maximum co-variance directions in two datasets



Supervised learning with LDA

- PCA doesn't care about class labels.
- Maximum variance isn't always best for discrimination.
- LDA: Finds directions that best discriminate data.

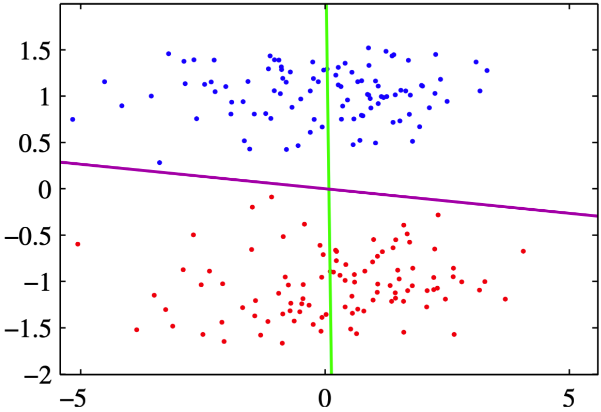


Figure: (Bishop Fig. 12.7) The first PC isn't always best for discrimination.

Section 8

Summary

Summary

- Neural data arrive in single neuron coordinates.
- Other coordinates may be more informative about the data.
- PCA finds coordinates that capture most variance.
- Can be found through SVD.
- Can be used for dimensionality reduction and visualization.
- Can be used for approximation and denoising.
- Many extensions, including Kernel PCA, CCA, LDA.

Section 9

Thanks for listening!