Lecture 15. Dimensionality Reduction

COMP90051 Statistical Machine Learning

Semester 2, 2018
Lecturer: Ben Rubinstein

THE UNIVERSITY OF MELBOURNE

This lecture

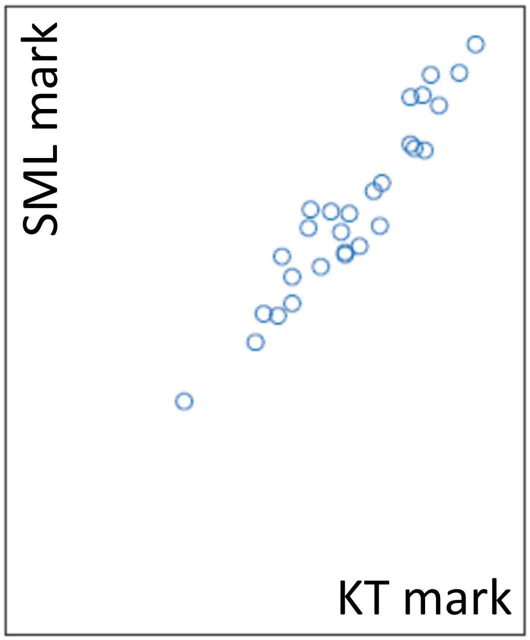
- Principal components analysis
  - Linear dimensionality reduction method
  - Diagonalising covariance matrix

Dimensionality reduction

- Previously in unsupervised learning: Clustering
- *Dimensionality reduction* refers to representing the data using a smaller number of variables (dimensions) while preserving the “interesting” structure of the data
- Such a reduction can serve several purposes
  - Visualisation (e.g., mapping multidimensional data to 2D)
  - Computational efficiency in a pipeline
  - Data compression or statistical efficiency in a pipeline

Exploiting data structure

- Dimensionality reduction in general results in loss of information
- The trick is to ensure that most of the "interesting" information (signal) is preserved, while what is lost is mostly noise
- This is often possible because real data may have inherently fewer dimensions than recorded variables
- **Example**: GPS coordinates are 3D, while car locations on a flat road are actually on 2D manifold
- **Example**: Marks* for Knowledge Technology and Statistical Machine Learning



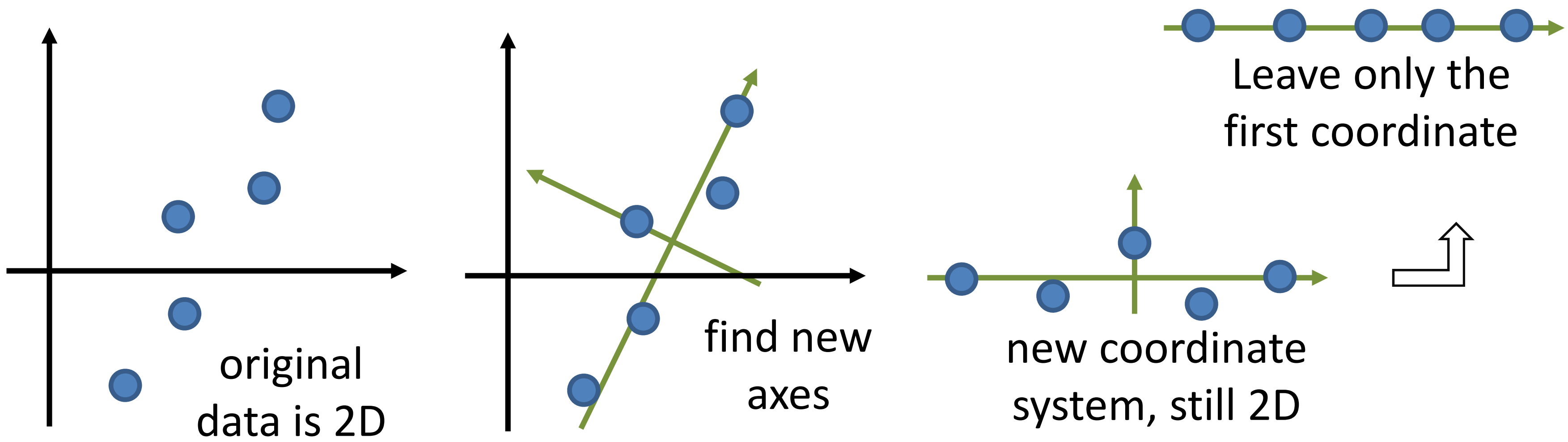
* synthetic data :)

Principal Component Analysis

Finding a rotation of data
that minimises covariance
between (new) variables

Principal component analysis

- Principal component analysis (PCA) is a popular method for dimensionality reduction and data analysis in general
- Given a dataset $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$, $\boldsymbol{x}_i \in \boldsymbol{R}^m$, PCA aims to find a new coordinate system such that most of the variance is concentrated along the first coordinate, then most of the remaining variance along the second coordinate, etc.
- Dimensionality reduction is then based on discarding coordinates except the first $l < m$



Naïve PCA algorithm

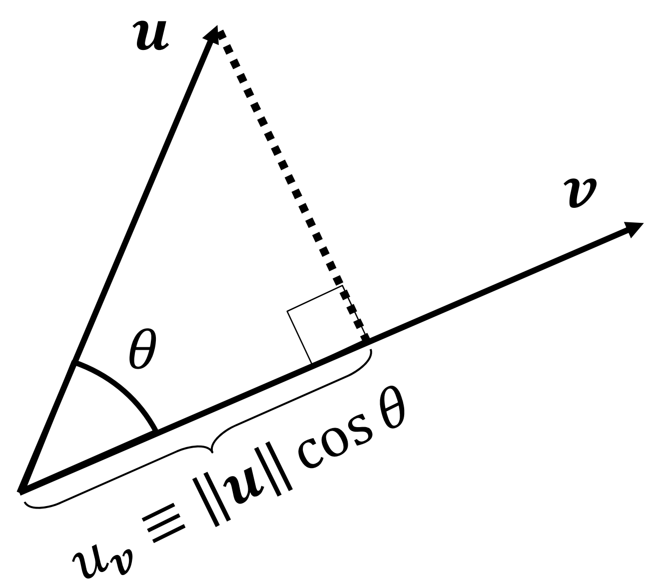
1. Choose a direction as new axis, such that the variance along this axis is maximised
2. Choose the next direction/axis perpendicular to all axes so far, such that the (remaining) variance along this axis is maximised
3. Repeat 2, until you have the same number of axes (i.e., dimensions) as in the original data
4. Project original data on the new axes. This gives new coordinates ("PCA coordinates")
5. For each point, keep only the first l coordinates

Such an algorithm if implemented directly would work, but there's a better solution

Formulating the problem

- The core of PCA is finding the new coordinate system, such that most of the variation is captured by "earlier" axes
- Let's write down this aim formally and see how it can be achieved

- First, recall the geometric definition of a dot product $\boldsymbol{u} \cdot \boldsymbol{v} = u_{\boldsymbol{v}} \|\boldsymbol{v}\|$
- Suppose $\|\boldsymbol{v}\| = 1$, so $\boldsymbol{u} \cdot \boldsymbol{v} = u_{\boldsymbol{v}}$
- Vector $\boldsymbol{v}$ can be considered a candidate coordinate axis, and $u_{\boldsymbol{v}}$ the coordinate of point $\boldsymbol{u}$



Data transformation

- So the new coordinate system is a set of vectors $\boldsymbol{p}_1, \dots, \boldsymbol{p}_m$, where each $\|\boldsymbol{p}_i\| = 1$
- Consider an original data point $\boldsymbol{x}_j$, $j = 1, \dots, n$, and a principal axis $\boldsymbol{p}_i$, $i = 1, \dots, m$
- The corresponding i^{th} coordinate for the first point after the transformation is $(\boldsymbol{p}_i)'(\boldsymbol{x}_1)$
  * For the second point it is $(\boldsymbol{p}_i)'(\boldsymbol{x}_2)$, etc.
- Collate all these numbers into a vector $[(\boldsymbol{p}_i)'(\boldsymbol{x}_1), \dots, (\boldsymbol{p}_i)'(\boldsymbol{x}_n)]' = \left((\boldsymbol{p}_i)'\boldsymbol{X}\right)' = \boldsymbol{X}'\boldsymbol{p}_i$, where $\boldsymbol{X}$ has original data points in columns

Refresher: basic stats

- Consider a random variable U and the corresponding sample $\boldsymbol{u} = [u_1, \dots, u_n]'$
- Sample mean $\bar{u} \equiv \frac{1}{n}\sum_i^n u_i$. Sample variance $\frac{1}{n-1}\sum_{i=1}^n (u_i - \bar{u})^2$
- Suppose the mean was subtracted beforehand (the sample is *centered*). In this case, the variance is a scaled dot product $\frac{1}{n-1}\boldsymbol{u}'\boldsymbol{u}$
- Similarly, if we have a centered random sample $\boldsymbol{v}$ from another random variable, sample covariance is $\frac{1}{n-1}\boldsymbol{u}'\boldsymbol{v}$
- Finally, if our data is $\boldsymbol{x}_1 = [u_1, v_1]', \dots, \boldsymbol{x}_n = [u_n, v_n]'$ organised into a matrix $\boldsymbol{X}$ with data in columns and centered variables in rows, we have that covariance matrix is $\boldsymbol{\Sigma}_X \equiv \frac{1}{n-1}\boldsymbol{X}\boldsymbol{X}'$

The objective of PCA

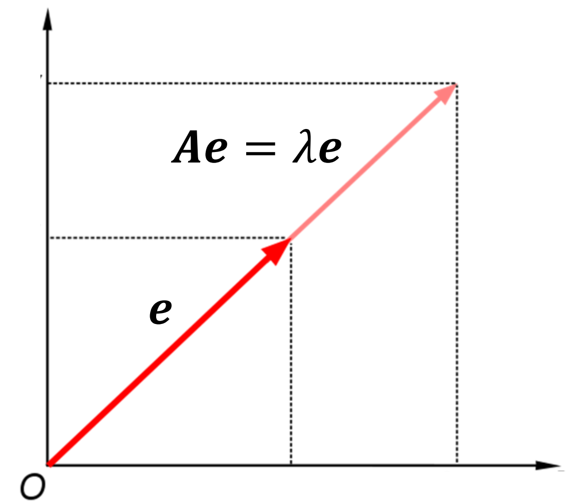
- We shall assume that the data is centered
- Let's start with the objective for the first principal axis. The data projected on this axis is $\boldsymbol{X}'\boldsymbol{p}_1$
- Accordingly, the variance along this principal axis is $\frac{1}{n-1}(\boldsymbol{X}'\boldsymbol{p}_1)'(\boldsymbol{X}'\boldsymbol{p}_1) = \frac{1}{n-1}\boldsymbol{p}_1'\boldsymbol{X}\boldsymbol{X}'\boldsymbol{p}_1 = \boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1$
  - Here $\boldsymbol{\Sigma}_X$ is the covariance matrix of the original data
- PCA should therefore find $\boldsymbol{p}_1$ to maximise $\boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1$, subject to $\|\boldsymbol{p}_1\| = 1$

Solving the optimisation

- PCA aims to find $\boldsymbol{p}_1$ that maximises $\boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1$, subject to $\|\boldsymbol{p}_1\| = \boldsymbol{p}_1'\boldsymbol{p}_1 = 1$
- Constrained → Lagrange mulitipliers. Introduce multiplier λ_1; set derivatives of Lagrangian to zero, solve
- $L = \boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1 - \lambda_1(\boldsymbol{p}_1'\boldsymbol{p}_1 - 1)$
- $\frac{\partial L}{\partial \boldsymbol{p}_1} = 2\boldsymbol{\Sigma}_X\boldsymbol{p}_1 - 2\lambda_1\boldsymbol{p}_1 = 0$
- $\boldsymbol{\Sigma}_X\boldsymbol{p}_1 = \lambda_1\boldsymbol{p}_1$
- The latter is precisely the definition of an eigenvector with λ_1 being the corresponding eigenvalue

Refresher on eigenvectors (1/2)

- Given a square matrix $\boldsymbol{A}$, a column vector $\boldsymbol{e}$ is called an eigenvector if $\boldsymbol{Ae} = \lambda\boldsymbol{e}$. Here λ is the corresponding eigenvalue



- Geometric interpretation: compare $\boldsymbol{Ae}$ with $\boldsymbol{Px}_i$ from previous slides. Here $\boldsymbol{A}$ is a transformation matrix ("new axes") for some vector $\boldsymbol{e}$. Vector $\boldsymbol{e}$ is such that it still points to the same direction after transformation

Refresher on eigenvalues (2/2)

- Algebraic interpretation: if $\boldsymbol{Ae} = \lambda\boldsymbol{e}$ then $(\boldsymbol{A} - \lambda\boldsymbol{I})\boldsymbol{e} = 0$, where $\boldsymbol{I}$ is the identity matrix
- This equation has a non-zero solution $\boldsymbol{e}$ if and only if the determinant is zero $|\boldsymbol{A} - \lambda\boldsymbol{I}| = 0$. Eigenvalues are roots of this equation called characteristic equation
- Eigenvectors and eigenvalues are prominent concepts in linear algebra and arise in many practical applications
- Spectrum of a matrix is a set of its eignevalues
- <u>There are efficient algorithms for computing eigenvectors</u> (not covered)

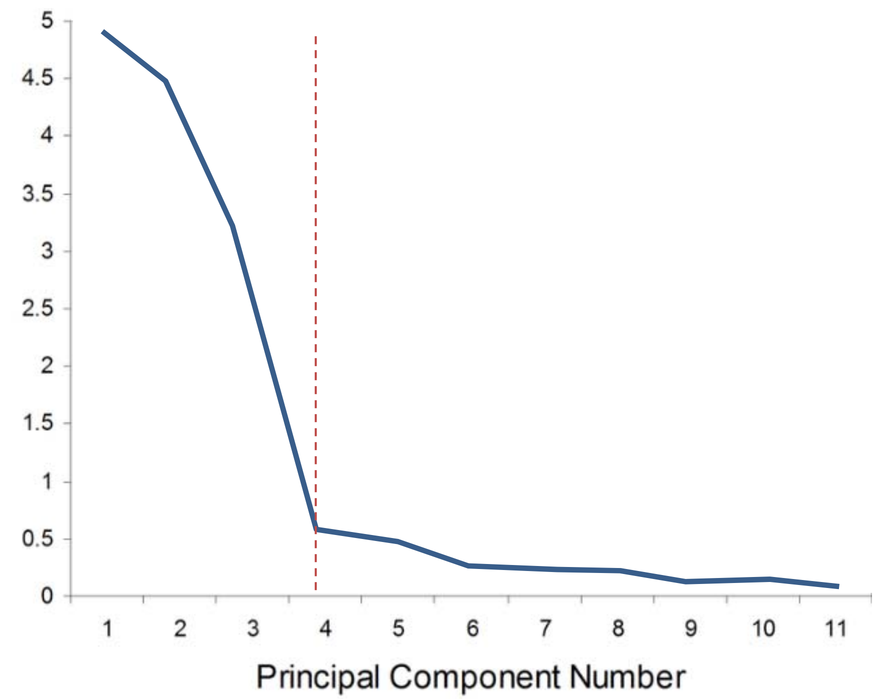
Variance captured by PCs

- In summary: we choose $\boldsymbol{p}_1$ as the eigenvector with largest eigenvalue, of centered covariance matrix $\boldsymbol{\Sigma}_X$

- Variance of data captured by $\boldsymbol{p}_1$:
  * Note we've shown $\lambda_1 = \boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1$,
  * But $\boldsymbol{p}_1'\boldsymbol{\Sigma}_X\boldsymbol{p}_1$ is var of projected data

  → First eigenvalue is variance captured

- Choose dimensions to keep from "knee" in scree plot



Efficient solution for PCA

- The same pattern can be used to find all PCs
  - Constraint $\|\boldsymbol{p}_i\| = 1$ prevents var $\boldsymbol{p}_i'\boldsymbol{\Sigma}_X\boldsymbol{p}_i$ diverging by rescaling $\boldsymbol{p}_i$
  - Each time we add additional constraints that next PC be orthogonal to all previous PCs. Equivalently, we search in their complement.
- Solution is to: setting $\boldsymbol{p}_i$ as all eigenvectors of centered data covariance matrix $\boldsymbol{\Sigma}_X$ in decreasing eigenvalue order
- Really possible with any $\boldsymbol{\Sigma}_X$?
- <u>Lemma</u>: a real symmetric $m \times m$ matrix has m real eigenvalues and corresponding eigenvectors are orthogonal
- <u>Lemma</u>: a PSD matrix further has non-negative eigenvalues.

Summary of PCA (1/2)

- Assume data points are arranged in columns of $\boldsymbol{X}$. That means that the *variables* are in rows
- Ensure that the data is centered: subtract the mean row (the data centroid) from each row
- We seek an *orthonormal* basis $\boldsymbol{p}_1, \dots, \boldsymbol{p}_m$
  - Each basis vector is of unit length and perpendicular to every other
- Find eigenvalues of centered data cov matrix $\boldsymbol{\Sigma}_X \equiv \frac{1}{n-1}\boldsymbol{X}\boldsymbol{X}'$
  - Always possible, relatively efficiently

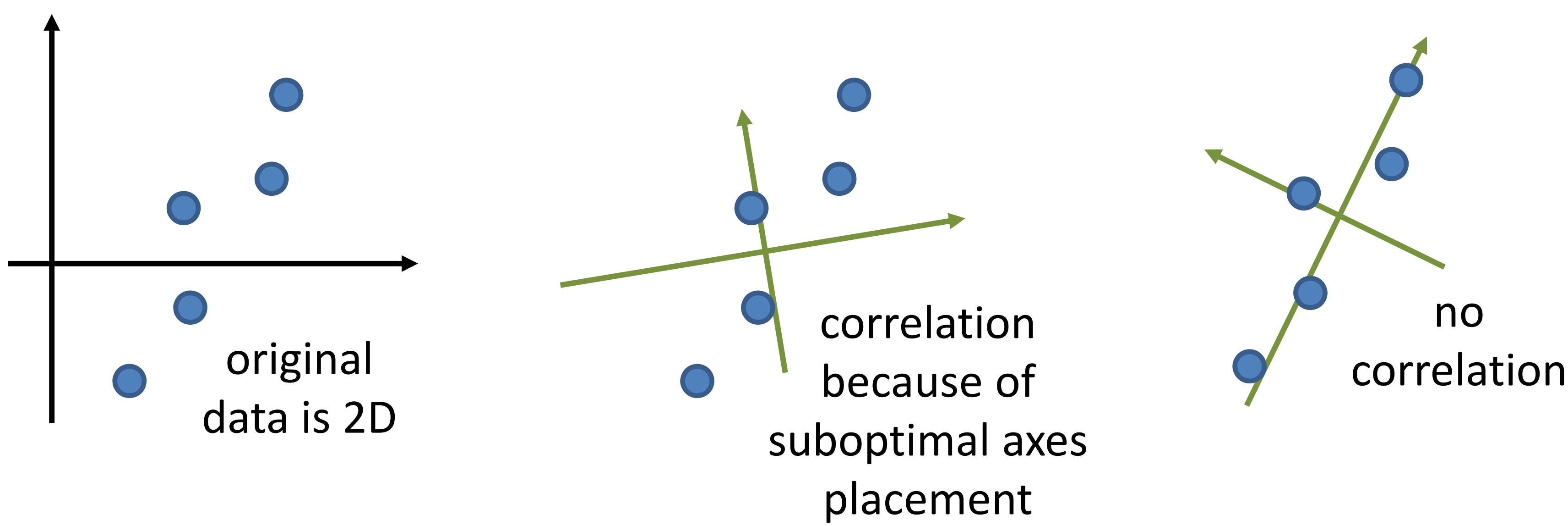
Summary of PCA (2/2)

- Sort eigenvalues from largest to smallest
  - Each eigenvalue equals to variance of data along corresponding PC
- Set $\boldsymbol{p}_1, \dots, \boldsymbol{p}_m$ as corresponding eigenvectors
- Project data $\boldsymbol{X}$ onto these new axes to get coordinates of the transformed data
- Keep only the first s coordinates to reduce dimensionality

- Another view of PCA: *s*-dim plane minimising residual sum-squares to data. (This is exactly spanned by *s* chosen PCs)

Additional effect of PCA

- PCA aims to find axes such that the variance along each subsequent axis is maximised
- Consider candidate axes i and $(i + 1)$. Informally, if there's a correlation between them, this means that axis i can be rotated further to capture more variance
- PCA should end up finding new axes (i.e., the transformation) such that the transformed data is uncorrelated



Spectral theorem for symmetric matrices

- In order to explore this effect further, we need to refer to one of the fundamental results in linear algebra
  - The proof is outside the scope of this subject
  - This is a special case of the singular value decomposition theorem
- Theorem: for any real symmetric matrix $\mathbf{\Sigma}_X$ there exists a real orthogonal matrix $\boldsymbol{P}$ with eigenvectors of $\mathbf{\Sigma}_X$ arranged in rows and a diagonal matrix of eigenvalues $\mathbf{\Lambda}$ such that $\mathbf{\Sigma}_X = \boldsymbol{P}'\mathbf{\Lambda}\boldsymbol{P}$

Diagonalising covariance matrix (1/2)

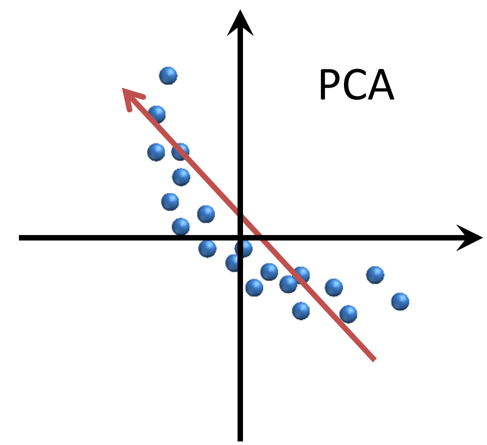
- Form transformation matrix $\boldsymbol{P}$ with evects (new axes) as rows
  - By our problem formulation, $\boldsymbol{P}$ is an orthonormal matrix
- Note that $\boldsymbol{P}'\boldsymbol{P} = \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix
  - To see this recall that each element of the resulting matrix multiplication is a dot product of the corresponding row and column
  - So element (i, j) of $\boldsymbol{P}'\boldsymbol{P}$ is the dot product $\boldsymbol{p}_i'\boldsymbol{p}_j$, which is 1 if $i = j$, and 0 otherwise
- The transformed data is $\boldsymbol{PX}$
  - Similar to above, note that element (i, j) of $\boldsymbol{PX}$ is the dot product $\boldsymbol{p}_i'\boldsymbol{x}_j$, which is the projection of $\boldsymbol{x}_j$ on axis $\boldsymbol{p}_i$, i.e., the new i^{th} coordinate for j^{th} point

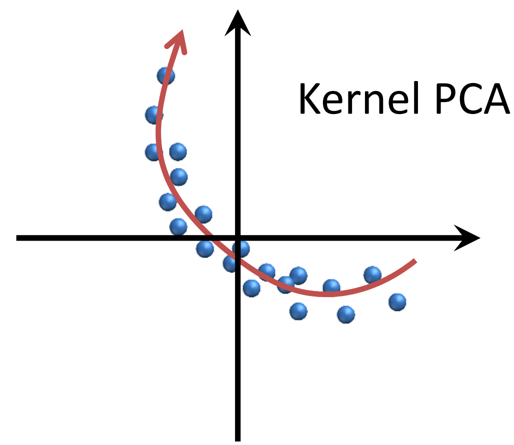
Diagonalising covariance matrix (2/2)

- The covariance of the *transformed data* is
- $\boldsymbol{\Sigma}_{\boldsymbol{PX}} \equiv \frac{1}{n-1}(\boldsymbol{PX})(\boldsymbol{PX})' = \frac{1}{n-1}(\boldsymbol{PX})(\boldsymbol{X}'\boldsymbol{P}') = \boldsymbol{P}\boldsymbol{\Sigma}_X\boldsymbol{P}'$
- By spectral decomposition theorem we have $\boldsymbol{\Sigma}_X = \boldsymbol{P}'\boldsymbol{\Lambda}\boldsymbol{P}$
- Therefore $\boldsymbol{\Sigma}_{\boldsymbol{PX}} = \boldsymbol{PP}'\boldsymbol{\Lambda}\boldsymbol{PP}' = \boldsymbol{\Lambda}$
- The covariance matrix of the transformed data is diagonal with eigenvalues on the diagonal of $\boldsymbol{\Lambda}$
- The transformed data is uncorrelated

Non-linear data and kernel PCA

- Low dimensional approximation need not be linear
- Kernel PCA: map data to feature space, then run PCA
  - Express principal components in terms of data points. Solution uses $\boldsymbol{X}'\boldsymbol{X}$ that can be kernelised $(\boldsymbol{X}'\boldsymbol{X})_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$
  - The solution strategy differs from regular PCA
  - Modular: Changing kernel leads to a different feature space transformation





This lecture

- Principal components analysis
  - Linear dimensionality reduction method
  - Diagonalising covariance matrix

- After non-teaching break: full Bayes ahead