

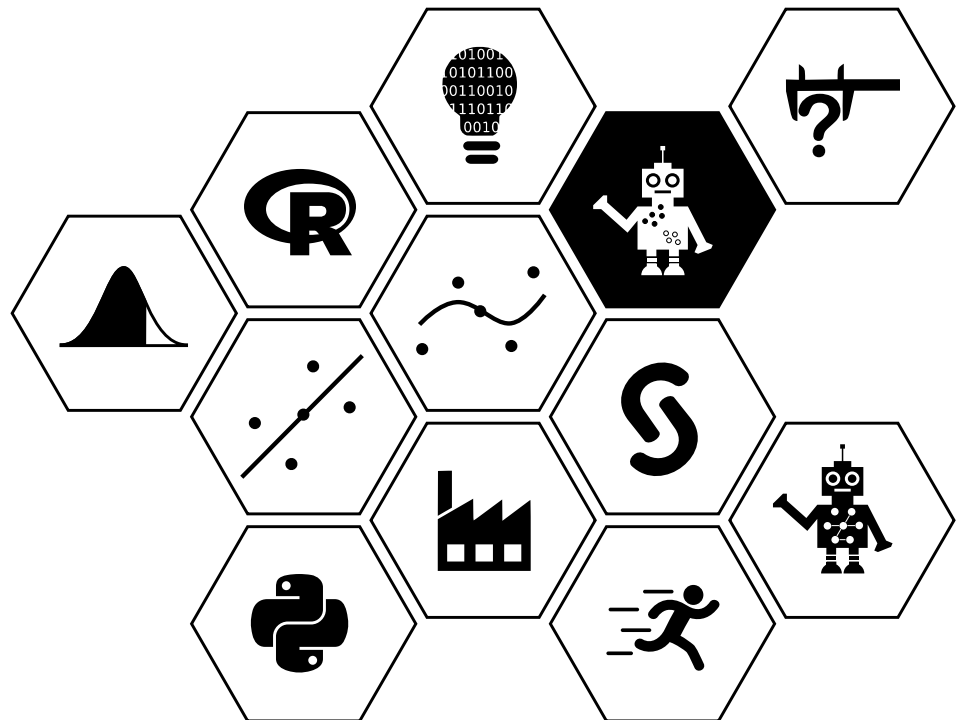
# Data Mining and Machine Learning I

Charis Chanialidis & Nema Dean

Academic Year 2021-22

Week 1:

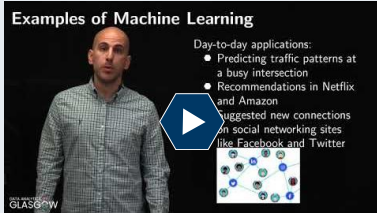
## Dimension Reduction and PCA



## General Information: Data Mining and Machine Learning I

### Course description

This course introduces students to machine learning methods and modern data mining techniques, with an emphasis on practical issues and applications.



### Introduction to the course

<https://youtu.be/Fadln3DMDQA>

Duration: 4m28s

### Course schedule

The course will consist of ten weeks, which will be released on a Monday by 11am BST. The topics are:

- Week 1 - Dimension reduction and PCA
- Week 2 - Principal component regression
- Week 3 - Classification
- Week 4 - Tree-based methods
- Week 5 - Support vector machines
- Week 6 - Peer assessment (there will be no material uploaded during that week)
- Week 7 - Neural Networks (Part I)
- Week 8 - Neural Networks (Part II)
- Week 9 - Hierarchical-based clustering techniques
- Week 10 - Partitioning-based clustering techniques

Note that between Weeks 5 and 6 (the week of 6th June 2022) there is a break week with no material or assessments.

### Office hours

There are 2 regular live session per week (excepting the break week), generally on Wednesdays at 8pm (BST) and Fridays at 11am (BST). These are an opportunity to ask any questions you may have about the course. They will be recorded with the recordings made available later for anyone who cannot attend. You can also post questions on both moodle forums and Teams.

### Aims

The aims of this course are:

- to introduce students to different methods for dimension reduction and clustering (unsupervised learning);
- to introduce students to a range of classification methods, beyond those covered in [Predictive Modelling](#);
- to introduce students to neural networks and deep learning;
- to introduce students to kernel methods and support vector machines.

### Intended learning outcomes

By the end of this course students will be able to:

- apply and interpret methods of dimension reduction including principal component analysis, multidimensional scaling and the biplot;
- apply and interpret classical methods for cluster analysis;
- apply and interpret advanced methods for classification;
- fit neural networks and support vector machines to data and assess their predictive ability objectively.

## Revision material

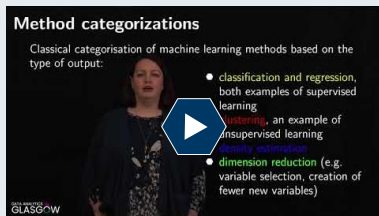
It will be beneficial for you if you revise the following subjects:

- linear regression (from [Learning from Data and Predictive Modelling](#));
- eigenvectors and eigenvalues (from Preliminary Maths);
- classification (from [Predictive Modelling](#)).

## Assessment and Feedback Calendar

The course assessment is comprised of 100% continuous assessment. Details of the assessment are:

Week	Topics	Type of Assessment	%	Period of Submission	Feedback
1 (3rd May)	W:1-2	Oral assessment	20	weeks 3 & 4	weeks 5
2 (9th May)	See Moodle	Peer assessment	20	week 6	week 7
5 (30th May)	W:3-5	Individual project	40	week 10	week 12
10 (11th Jul)	W:7-10	Moodle quiz	20	week 12	week 12



### Overview of the course

<https://youtu.be/nzxfekG0gyA>

Duration: 3m43s

## Resources

The course will be self-contained in the learning material. However, for each week we will point to chapters in the following books that you might wish to consult for additional material:

- Trevor Hastie, Robert Tibshirani and Jerome Friedman - [The Elements of Statistical Learning](#)
- Gareth James, Daniela Witten, Trevor Hastie & Robert Tibshirani - [An Introduction to Statistical Learning](#)
- Alex Smola & S.V.N. Vishwanathan - [Introduction to Machine Learning](#)
- David Barber - [Bayesian Reasoning and Machine Learning](#)
- Christopher M. Bishop - [Pattern Recognition and Machine Learning](#)
- Simon Rogers, Mark Girolami - [A First Course in Machine Learning](#)



### Supplementary material:

There is a list of free, open source books on Data Mining and Machine Learning on Github. The list covers more than the aims of this course and can be really helpful if someone wants to dig deeper (and/or cannot wait for the Data Mining and Machine Learning II course next summer).

# Dimension Reduction and Principal Component Analysis

## The problem with too much data

We're currently living in the most data rich period of human history (to date). Which seems like it should be a good thing but data does not necessarily always equal information, knowledge or insight. Quite apart from the ethics of how such data is collected and used, the sheer amount of data available can be more overwhelming than useful. The goal of extracting useful information from available data is the main role of the data scientist.



Why big data is in trouble: they forgot about applied statistics

<https://www.kdnuggets.com/2016/07/big-data-trouble-forgot-applied-statistics.html>

Jeff Leek points out a major problem with the analysis of big data

The Data Mining and Machine Learning module in particular introduces numerous different learning techniques (both supervised and unsupervised) that can assist in this process. The issue with many more advanced, complex techniques (including some in this course) is that while they can uncover hidden, intricate relationships and evidence, many of them have problems when the data involved are big.

Data can be "big" in three different ways:

- many observations,
- many variables,
- both!

The more complicated models often do not scale well as either the number of observations and/or the number of variables increase dramatically. So some kind of dimension reduction method is needed either prior to or in conjunction with the other model in order to allow it to run.

## Dimension reduction

Dimension reduction techniques generally fall into two categories:

- filter techniques: applied to the data before the other method
  - either unsupervised (with no particular outcome used)
  - or supervised (applied based on optimisation with respect to an outcome of interest) and
- wrapper techniques: applied as part of the other method (essentially "wrapped" around it).

We will see examples of the latter both in this course (tree-based methods) and in next year's Data Mining and Machine Learning II (regularised regression).

These methods can also be split into:

- techniques that retain the original variables (albeit fewer of them), known as *feature or variable selection*, and
- techniques that create/extract a smaller set of new variables based on the old ones, known as *feature extraction*.

## Variable selection

Variable selection tries to choose a subset of the original variables that is in some way optimal, either in terms of an outcome of interest or some more general measure. This is preferred when the model (or output) allows for interpretation of parameters relating to the original variables and this interpretation is part of the goal of the analysis. You have actually already done a crude version of this when building regression models by removing variables according to their lack of significance. If, however, the variables are of no particular interest in a substantive way, we may choose to use them to construct new variables with particular desirable properties. One of the most common and popular versions of this, is called principal component analysis which we will focus on in this week.

## Introduction to Principal Component Analysis (PCA)

Principal component analysis is a technique for continuous data that takes  $p$  different linear combinations of the  $p$  original variables to create  $p$  new variables, called components. These components are constructed to be uncorrelated with one another.

You may be wondering how this is in any way useful for reducing dimensions, given that we have the same number of variables as we started with; that is  $p$ . The only apparent difference is that we have removed any intervariable correlation. However, these variables are also ordered in decreasing order of variability, i.e. the first variable has the largest amount of variability, the second variable has the second largest amount of variability, etc.

The golden rule when using PCA to reduce dimensionality is

$$\text{Variation} = \text{Information}$$

Our goal, therefore, is to reduce the dimensionality of the data while still retaining most of the information contained in the original data. If we can capture a large amount of the variability by keeping the first  $q$  components (out of  $p$ ) where  $q \ll p$ , then we will achieve that goal. Often we assume that the small amount of variability lost with the discarded  $p - q$  components may just be random noise rather than any systematic source of variation.

Note that another use of PCA rather than dimensionality reduction is to produce uncorrelated variables for use in methods where excess correlation may cause problems (like multicollinearity in linear regression). You'll see an example of this type of usage in the following week where we'll discuss principal component regression.

### Short intro to PCA

Let's start with describing the situation that we are faced with and the problem at hand.

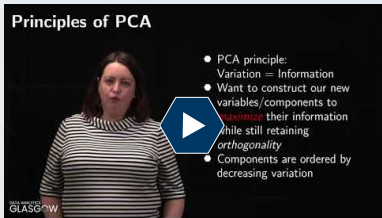
- We start with  $n$  observations  $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$  where each one is  $p$  dimensional.
- If  $p$  is large then that could make parameter estimation really difficult. In addition it will not be easy to visualise the observations due to the high dimensions.

We would like to keep  $q$  dimensions (where  $q \ll p$ ) AND preserve the properties of interest.

We want to project a  $p$ -dimensional observation into  $q$  dimensions. In Figure 1 we see the idea of projection. Specifically, we have ten two-dimensional observations (i.e.  $p = 2$ ),  $x_i$  where  $i = 1, \dots, 10$ , which are projected into one dimension (i.e.  $q = 1$ ),  $y_{i1}$ . To be precise they are actually projected into two dimensions but the second one,  $y_{i2}$ , takes the value 0 always. As a result, that dimension holds no information at all about the new observation. In addition, in this case the one (and only) dimension of the new observations is one of the original two dimensions (i.e.  $y_{i1} = x_{i1}$ ), but this is not necessary. As we mentioned earlier, we would like to retain any "interesting" structure in the data when we perform the projection. But what do we mean by "interesting" structure and how can we locate "interesting" structures in the data? This is where the previous rule (i.e. *Variation = Information*) comes to play, and we will soon see that information about the structure of the data is present; in a projection that has high variance.

### Intuition behind PCA

You can picture PCA as a way of looking for new axes for our data. The original axes represent our original variables and what PCA is trying to do is rotate these axes around the fixed data (also keeping the origin fixed) so that the first axis lines up with the line/direction of most variability in our data, the second axis is perpendicular/orthogonal to the first new axis and follows the line of second most variability, the third is orthogonal to the first two new axes and follows the line of third most variability and so on. This is demonstrated in the following video.



### An Introduction to Principal Component Analysis

<https://youtu.be/OIrVLTWo0c>

Duration: 14m46s

### Basic idea of PCA

The basic idea of principal component analysis is to find a small number of uncorrelated linear combinations of the original variables which explain most of the variation in the data.

The linear combinations of the variables are a rotation of our data which are made up of  $n$  observations:  $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)$  where every observation  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$ , is a  $p$  dimensional random vector. This is true for all  $i$  (i.e.  $i = 1, 2, \dots, n$ ). (You can think of the previous notation as having data that are comprised of  $n$  observations and  $p$  variables)

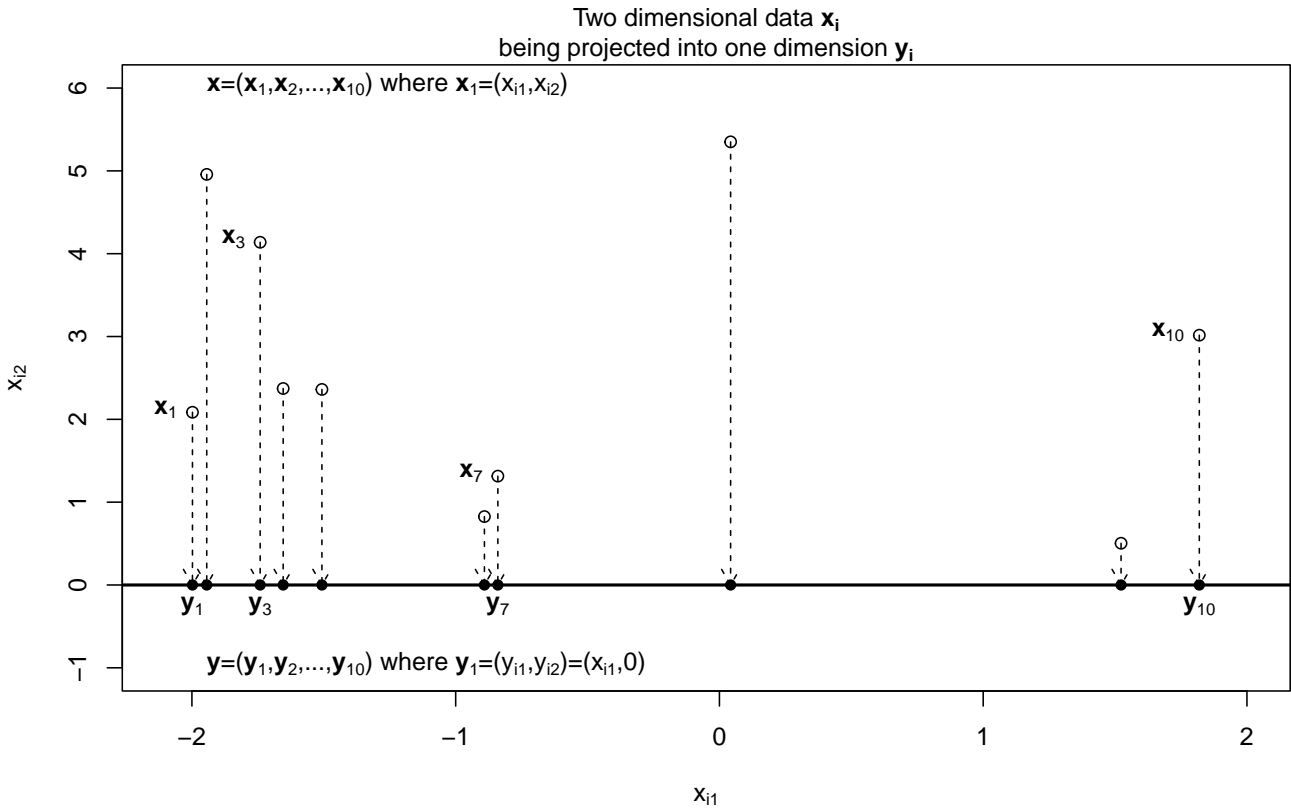


Figure 1: Simple example of projecting two-dimensional data into one dimension

The initial objective is to find a linear transformation

$$y_{i1} = \mathbf{a}_1^T \mathbf{x}_i = \sum_{j=1}^p a_{1j} x_{ij},$$

where  $\mathbf{a}_1 = (a_{11}, \dots, a_{1p})^T$  is a vector of constants such that the  $\text{Var}(y_{i1})$  is maximised, subject to the normalising constraint  $\mathbf{a}_1^T \mathbf{a}_1 = 1$ .

We can prove mathematically that choosing  $\mathbf{a}_1$  to be the eigenvector,  $\mathbf{u}_{.1}$ , associated with the largest eigenvalue,  $\lambda_1$  of  $\Sigma$ , the variance-covariance matrix of our data, gives us a first principal component with the correct properties. Similarly the  $j^{\text{th}}$  principal component is simply the eigenvector,  $\mathbf{u}_{.j}$ , associated with the  $j^{\text{th}}$  largest eigenvalue,  $\lambda_j$ . The proof of this can be seen in the following supplementary material (but is not really needed to understand PCA).



#### Supplementary material:

Suppose  $\mathbf{X}_i = (X_{i1}, \dots, X_{ip})'$  is a  $p$  dimensional random vector (i.e. a  $p$ -vector) with known variance-covariance matrix  $\Sigma$ , i.e.  $\text{Var}(\mathbf{X}_i) = \Sigma$ . To find the first principal component  $Y_{i1}$  with maximum variance we maximise  $\text{Var}(Y_{i1}) = \text{Var}(\mathbf{a}_1^T \mathbf{X}_i) = \mathbf{a}_1^T \text{Var}(\mathbf{X}_i) \mathbf{a}_1 = \mathbf{a}_1^T \Sigma \mathbf{a}_1$  subject to the normalising constraint  $\mathbf{a}_1^T \mathbf{a}_1 = 1$ . Note that  $\mathbf{a}_1^T \Sigma \mathbf{a}_1$  is a **scalar**.

Assuming that  $\Sigma$  is a positive-definite symmetric matrix, we know that it will have a spectral decomposition

$$\Sigma = U \Delta U^T = \sum_{j=1}^p \lambda_j \mathbf{u}_{.j} \mathbf{u}_{.j}^T.$$

where  $\Delta$  is a diagonal matrix of eigenvalues in decreasing order and  $U$  is an orthogonal matrix with  $\lambda_j$ 's corresponding eigenvector,  $\mathbf{u}_{.j}$ , in column  $j$ . Then

$$\text{Var}(Y_{i1}) = \mathbf{a}^T \Sigma \mathbf{a} = \mathbf{a}^T U \Delta U^T \mathbf{a} = \mathbf{b}^T \Delta \mathbf{b}$$

with  $\mathbf{b} = U^T \mathbf{a}$ . If  $\mathbf{a}^T \mathbf{a} = 1$  then  $\mathbf{b}^T \mathbf{b} = \mathbf{a}^T U U^T \mathbf{a} = 1$  since we know that  $U$  is orthogonal. But

$$\text{Var}(Y_{i1}) = \mathbf{b}^T \Delta \mathbf{b} = \sum_{j=1}^p \lambda_j b_j b_j^T,$$

where  $b_j$  is the  $j^{\text{th}}$  element of vector  $\mathbf{b}$ . The unit length vector that maximizes this is  $\mathbf{b}_1 = (1, 0, \dots, 0)^T$  and the corresponding  $\mathbf{a}_1$  is the first eigenvector (i.e. the first column of  $U$ ) since  $\mathbf{a}_1 = U\mathbf{b}_1$ , noting that since  $U$  is orthogonal  $U^T U = U U^T = I_p$ .

Now suppose we want to obtain  $Y_{i2} = \mathbf{a}_2^T X_i$  with  $\mathbf{a}_2^T \mathbf{a}_2 = 1$  which maximizes the  $\text{Var}(\mathbf{a}_2^T X_i)$  subject to the constraint  $\mathbf{a}_2^T \mathbf{a}_1 = 0$  (since  $Y_{i2}$  is uncorrelated to  $Y_{i1}$ ). We proceed as before and  $\mathbf{b}_2 = (0, 1, \dots, 0)^T$  and the corresponding  $\mathbf{a}_2$  is the second eigenvector of  $\Sigma$  (i.e. the second column of  $U$ ).

The  $j^{\text{th}}$  principal component is the  $j^{\text{th}}$  eigenvector giving the linear combination,  $\mathbf{a}_j^T X_i = \mathbf{u}_j^T X$ , picked by this procedure.

## Terminology

Before we start looking at actually using PCA, we need to define some terms.

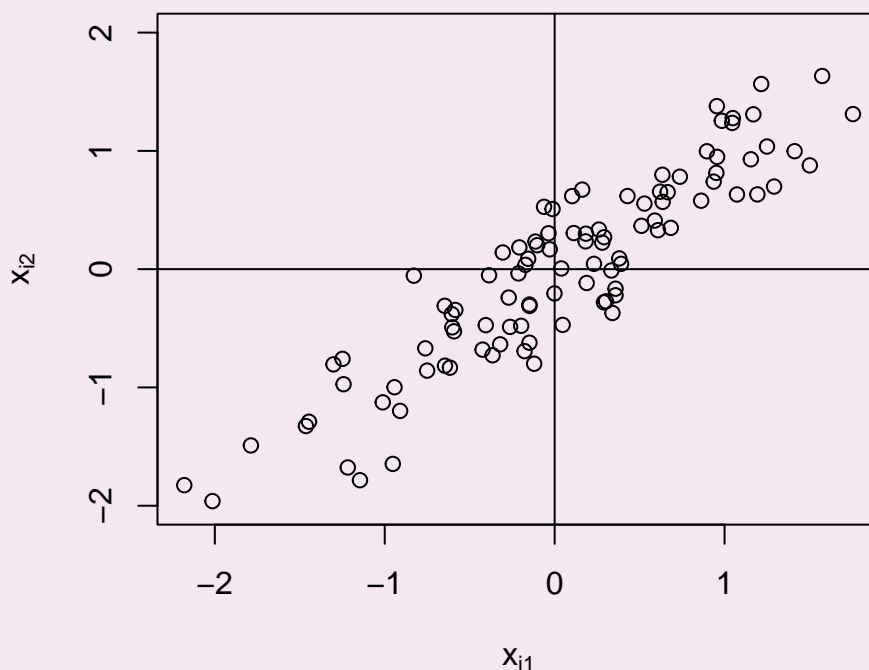
- Realisations of the new random variable  $\mathbf{y}_i = (y_{i1}, \dots, y_{ip})$  are called the (principal) component scores,
- $\mathbf{a}_1, \dots, \mathbf{a}_p$ , and the linear transformations they define, are called the principal components and
- the elements of the eigenvectors  $\mathbf{a}_1, \dots, \mathbf{a}_p$ , i.e. the entries in the vectors  $\mathbf{a}_j = (a_{j1}, \dots, a_{jp})^T$ , the  $a_{jk}$ 's are called the component loadings. (where  $a_{jk}$  is the  $k^{\text{th}}$  variable's loading for the  $j^{\text{th}}$  principal component)

In practice these loadings are often used to attempt to interpret principal components, which we'll discuss in the examples later on.



### Task 1.

Say we have two-dimensional data given by the following graph.



What will this data look like in the new principal component space (where the vertical and horizontal axes are now aligned with the PCs)? Will it have the same shape?

## Properties of PCA

We can show (due to the mathematical properties of eigenvalues and eigenvectors) the following about PCA:

1.  $Y_{.1}, Y_{.2}, \dots, Y_{.p}$  are pairwise-uncorrelated and  $\text{Var}(\mathbf{Y}) = \text{diag}(\lambda_1, \dots, \lambda_p) = \Delta$ .  
(i.e. the variance of the  $k^{\text{th}}$  component is  $\lambda_k$ , the  $k^{\text{th}}$  eigenvalue of  $\Sigma$ )
2. The "total variance" is preserved under the principal component transformation.  
(i.e.  $\sum_{j=1}^p \text{Var}(Y_{.j}) = \sum_{j=1}^p \text{Var}(X_{.j})$ )

3. The first  $k$  principal components account for the proportion  $\frac{\sum_{l=1}^k \lambda_l}{\sum_{j=1}^p \lambda_j}$  of the total variance.  
(as implied by the previous statement)



### Supplementary material:

Proof of 1:

$$\text{Cov}(\mathbf{Y}_{.j}, \mathbf{Y}_{.k}) = \mathbf{a}_j^T \Sigma \mathbf{a}_k = \mathbf{a}_j^T \lambda_k \mathbf{a}_k = \lambda_k \mathbf{a}_j^T \mathbf{a}_k$$

since  $\Sigma \mathbf{a}_k = \lambda_k \mathbf{a}_k$ . But  $\mathbf{a}_j^T \mathbf{a}_k = 0$  if  $j \neq k$  and 1 if  $j = k$ . Therefore:

$$\begin{aligned} \text{Cov}(\mathbf{Y}_{.j}, \mathbf{Y}_{.k}) &= \lambda_k \text{ if } j = k; \\ \text{Cov}(\mathbf{Y}_{.j}, \mathbf{Y}_{.k}) &= 0 \text{ if } j \neq k. \end{aligned}$$

hence  $\text{Var}(\mathbf{Y}) = \text{diag}(\lambda_1, \dots, \lambda_p) = \Delta$  and  $\text{Var}(\mathbf{Y}_{.j}) = \lambda_j$ .

Proof of 2:

$$\begin{aligned} \sum_{j=1}^p \text{Var}(\mathbf{Y}_{.j}) &= \text{trace}(\Delta) = \sum_{j=1}^p \lambda_j, \\ &= \text{trace}(U^T \Sigma U) = \text{trace}(\Sigma U U^T) \\ &= \text{trace}(\Sigma) = \sum_{j=1}^p \text{Var}(\mathbf{X}_{.j}). \end{aligned}$$

Note that  $U^T \Sigma U = U^T U \Delta U^T U = \Delta$ , where  $\Delta = \text{diag}(\lambda_1, \dots, \lambda_p)$  the diagonal matrix of eigenvalues in descending order and for square matrices  $A, B, C$ , the ordering of multiplication within the trace operator can be switched, i.e.  $\text{trace}(ABC) = \text{trace}(BCA)$ .

Proof of 3 follows from 1 and 2.

## PCA Use in Reducing Dimensionality

So we have  $p$  components where before we had  $p$  variables. Where's the dimension reduction?

The principal components  $p \times p$  matrix  $A = [a_1, \dots, a_p]$  provides a useful coordinate system to display the data graphically, since most of the variation of the data will be contained in the first few components. By working with the first  $k$  principal components, (accounting for  $\frac{\sum_{l=1}^k \lambda_l}{\sum_{j=1}^p \lambda_j} \cdot 100\%$  of variance) dimension reduction is usually achieved without substantial loss of information. The projection of  $\mathbf{X}$  onto this coordinate system is  $\mathbf{Y} = A^T \mathbf{X}$ . Since this is an orthogonal transformation, it is a rotation.

**Covariance versus correlation** Note that PCA can be sensitive to scale differences in the variables, e.g. if the first variable has a very high variance compared to the others, it will be over-represented in the PCA (i.e. will get a greater loading than is warranted). Therefore, when the variables are measured in different units or the variances are very different over different variables, it is wise to perform PCA on the correlation matrix rather than the covariance matrix. This is equivalent to standardising the variables.

**PCA in practice** In practice we will not have the true population covariance matrix  $\Sigma$  or the true correlation matrix, so we can estimate the population covariance matrix with the standard sample covariance matrix for observation vectors  $\mathbf{x}_1, \dots, \mathbf{x}_n$  with  $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$ :

$$\mathbf{S} = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

or for the  $n \times p$  matrix  $\mathbf{X}$ :

$$(n-1)\mathbf{S} = \left(\mathbf{X} - \frac{1}{n} \mathbf{1} \mathbf{1}^T \mathbf{X}\right)^T \left(\mathbf{X} - \frac{1}{n} \mathbf{1} \mathbf{1}^T \mathbf{X}\right) = (\mathbf{X}^T \mathbf{X} - n \bar{\mathbf{x}} \bar{\mathbf{x}}^T)$$

where  $\mathbf{1}$  is a  $p \times 1$  column vector of ones and  $\bar{\mathbf{x}} = (\bar{x}_1, \dots, \bar{x}_p)^T$  denotes the vector of variable means.



If we construct a diagonal matrix  $D$  with entries equal to the reciprocal of the square root of the diagonal entries of  $S$ , i.e.  $D_{ii} = 1/\sqrt{S_{ii}}$ , then the sample correlation matrix  $R$  is given by

$$R = DSD$$

We perform an eigenanalysis on either  $S$  or  $R$  (which as positive semi-definite matrices will give non-negative eigenvalues). Given the resulting eigenvector matrix  $U$  we usually select  $q$  ( $< p$ ) columns and use the resulting  $p \times q$  matrix  $A$  to construct the component scores for each observation  $x_i$  via:

$$y_i = A^T x_i$$

Each observation results in a  $q$ -vector of scores,  $y_i = (y_{i1}, \dots, y_{iq})^T$ .

Sometimes, the mean of the variables is subtracted before projecting onto the principal component space giving the component score equation:

$$y_i = A^T (x_i - \bar{x})$$

The scores produced in  $R$  will be centred in this way. If we choose  $q$  to be 2 or 3, the data can then be visualised in the reduced space (via scatterplots or pairs plots).

**Choosing the number of principal components** While we want to reduce the dimensionality as much as possible, we also don't want to lose too much information when doing so. There are a number of different strategies for deciding on the number of components,  $q$ , to keep. Here we list the three most popular.

1. **Proportion of Variation:** The variance of each component  $j$  is given by the corresponding eigenvalue of  $S/R$ ,  $\lambda_j$ . We know that the total variance of the original matrix is equal to the sum of all eigenvalues,  $\sum_{j=1}^p \lambda_j$ , so we decide on a proportion of variance that we want retained by the principal component analysis, usually at least 80%, and keep the number of eigenvalues that will give at least this proportion of variance, i.e.  $q$  such that  $\sum_{l=1}^q \lambda_l / \sum_{j=1}^p \lambda_j \geq 0.8$ . (Note that for the sample correlation matrix  $R$ ,  $\sum_{j=1}^p \lambda_j = p$ .)
2. **Cattell's method:** Plot the cumulative proportion of variance explained by the first  $k$  components,  $\sum_{l=1}^k \lambda_l / \sum_{j=1}^p \lambda_j$  versus the number of components  $k$ . This is known as a scree plot. We choose the number of components where the scree plot levels off. Another type of scree plot simply plots the component variation, the eigenvalues,  $\lambda_j$  versus the component numbers  $j$ . Again the point before the plot levels off is chosen to be the number of retained components.
3. **Kaiser's method:** Alternatively, we can choose to retain the components whose eigenvalues are greater than the average eigenvalue,  $\bar{\lambda} = \sum_{j=1}^p \lambda_j / p$  (for the correlation matrix this will be 1). Since the components are in decreasing order of eigenvalue, this equates to choosing the number of components where the component beyond the last one selected has eigenvalue less than the average.

## Summary of PCA

- **When is PCA useful?** We will only be able to reduce the dimensionality without losing large amounts of information when the original variables are highly correlated. When the correlation between variables is weak, we will need a large number of components to capture enough variability. So, before using PCA in practice, you should always examine the sample correlation matrix to see if PCA is worthwhile.
- **Sample Correlation or Sample Covariance Matrix?** If the variables have been recorded on different scales or they have very different variances, then it is advisable to base the analysis on the sample correlation matrix. Note that the correlation matrix can only be used in cases without constant variables.
- **How many components should be retained?** If PCA is appropriate, you should always specify which method of selecting the number of components in advance of running the analysis: either proportion of variation (along with the proportion desired), Cattell's scree plot method or Kaiser's method.
- **Interpretation of the components** Since the components are simply a mathematical construct to preserve variation, there is no particular reason why the resulting principal components (the eigenvectors/loadings vectors) should have an interpretation. You can think of the loading elements of the same approximate value as being an average of those variables. If there are some positive and some negative loadings in an eigenvector, you can consider this as the difference between 2 averages (or the contrast). Note that the eigenvectors are only uniquely defined up to a sign change.
- **Assumptions** No model or distribution is assumed for the data in order to run PCA.
- **Inference** It is possible to perform inference for the population principal components, but this will not be considered here. It is more usual in practice to use PCA as a descriptive technique than a method of formal analysis.

- **Influence of outliers on PCA** One topic that could be explored further is the effect of outliers on a PCA analysis. Outliers can have a drastic effect on the sample mean vector and sample covariance/correlation matrix and indeed it is correct to assume that PCA can also be severely distorted by outliers. Hence it is advisable to check the data for outliers prior to performing PCA or, alternatively, to use a robust estimate of the covariance/correlation matrix when applying the PCA.
- **Examine the PC Scores Plot.** It is useful to look at the PC Scores Plot to explore any structure in the data, e.g. clusters, unusual observations, etc.

## Applying PCA in R

There are two common commands in R for principal component analysis, `prcomp` and `princomp`. We will focus on `princomp` but the two are very similar (`princomp` uses `eigen` to compute the PCA, while `prcomp` uses `svd` which can be more stable in some cases, particularly when there are more variables than observations in a dataset).

Specifically:

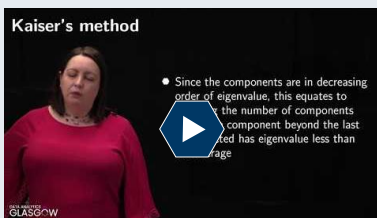
- the first argument of `princomp` is the  $n \times p$  data matrix while
- the second argument is `cor` which is set to `TRUE` if the sample correlation matrix is to be used in the PCA and `FALSE` if the sample covariance matrix is to be used (this is the default).

The fitted object produces a list with various entries:

- the entry `sdev` gives the standard deviations (the square roots of the eigenvalues) of the components,
- the `loadings` entry is a matrix with the eigenvectors in the columns,
- `centre` gives the means subtracted,
- if the argument `scores = TRUE` then the output will also include an  $n \times p$  matrix of principal component scores.

Using `plot` on the fitted object will result in a scree plot.

The following video looks at a three-dimensional dataset on turtles and discusses how to apply PCA and choose the number of components to retain.



### PCA Worked Example (First Part)

[https://youtu.be/B1Y65\\_T\\_fuc](https://youtu.be/B1Y65_T_fuc)

Duration: 17m07s



### Example 1.

Let's take a look at a dataset on wine. This dataset gives the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The data were sourced from the UCI Machine Learning Repository and you can find more information about them [here](#).

```
wine <- read.csv("wine.data.csv")
str(wine, vec.len = 2)

'data.frame':  178 obs. of  14 variables:
 $ Class          : int  1 1 1 1 1 ...
 $ Alcohol        : num  14.2 13.2 ...
 $ Malic.acid     : num  1.71 1.78 2.36 1.95 2.59 ...
 $ Ash            : num  2.43 2.14 2.67 2.5 2.87 ...
 $ Alcalinity.of.ash : num  15.6 11.2 18.6 16.8 21 ...
 $ Magnesium      : int  127 100 101 113 118 ...
 $ Total.phenols  : num  2.8 2.65 2.8 3.85 2.8 ...
 $ Flavanoids     : num  3.06 2.76 3.24 3.49 2.69 ...
 $ Nonflavanoid.phenols : num  0.28 0.26 0.3 0.24 0.39 ...
 $ Proanthocyanins : num  2.29 1.28 2.81 2.18 1.82 ...
 $ Colour.intensity : num  5.64 4.38 5.68 7.8 4.32 ...
 $ Hue            : num  1.04 1.05 1.03 0.86 1.04 ...
 $ OD280.OD315.of.diluted.wines: num  3.92 3.4 3.17 3.45 2.93 ...
 $ Proline        : int  1065 1050 1185 1480 735 ...

summary(wine)

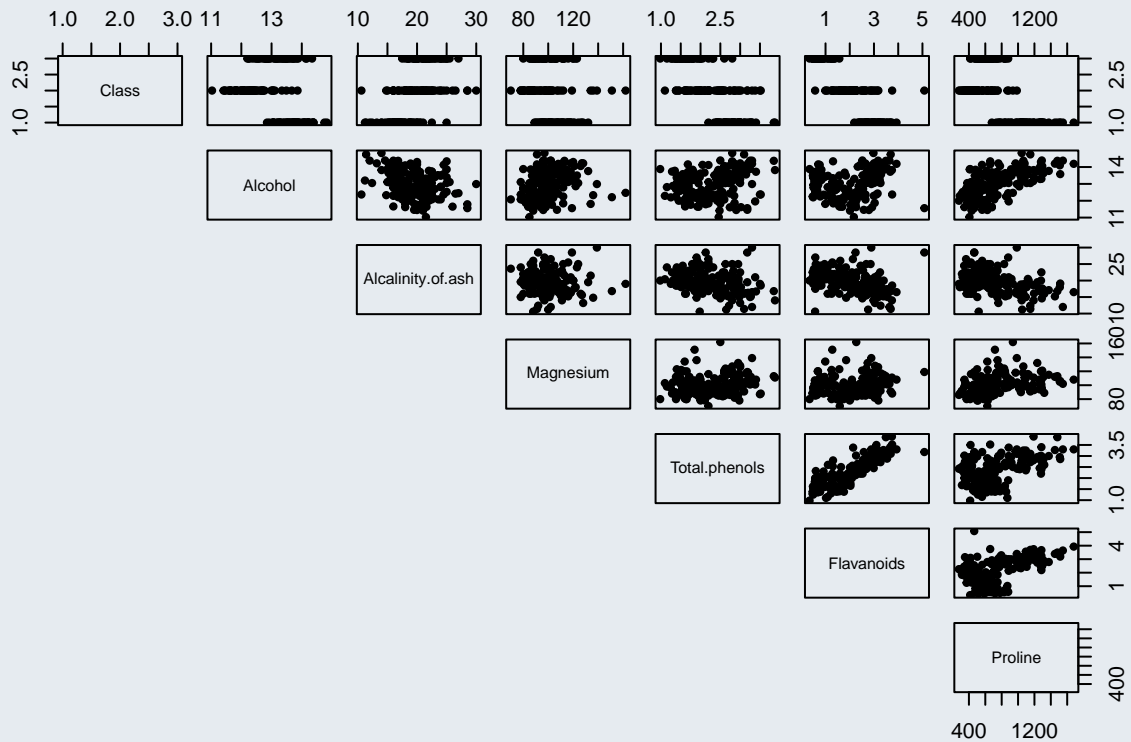
      Class      Alcohol      Malic.acid      Ash
Min.   :1.000   Min.   :11.03   Min.   :0.740   Min.   :1.360
1st Qu.:1.000   1st Qu.:12.36   1st Qu.:1.603   1st Qu.:2.210
Median :2.000   Median :13.05   Median :1.865   Median :2.360
Mean   :1.938   Mean   :13.00   Mean   :2.336   Mean   :2.367
3rd Qu.:3.000   3rd Qu.:13.68   3rd Qu.:3.083   3rd Qu.:2.558
Max.   :3.000   Max.   :14.83   Max.   :5.800   Max.   :3.230
Alcalinity.of.ash  Magnesium      Total.phenols      Flavanoids
Min.   :10.60     Min.   : 70.00   Min.   :0.980   Min.   :0.340
1st Qu.:17.20     1st Qu.: 88.00   1st Qu.:1.742   1st Qu.:1.205
Median :19.50     Median : 98.00   Median :2.355   Median :2.135
Mean   :19.49     Mean   : 99.74   Mean   :2.295   Mean   :2.029
3rd Qu.:21.50     3rd Qu.:107.00  3rd Qu.:2.800   3rd Qu.:2.875
Max.   :30.00     Max.   :162.00   Max.   :3.880   Max.   :5.080
Nonflavanoid.phenols Proanthocyanins Colour.intensity      Hue
Min.   :0.1300     Min.   :0.410   Min.   : 1.280   Min.   :0.4800
1st Qu.:0.2700     1st Qu.:1.250   1st Qu.: 3.220   1st Qu.:0.7825
Median :0.3400     Median :1.555   Median : 4.690   Median :0.9650
Mean   :0.3619     Mean   :1.591   Mean   : 5.058   Mean   :0.9574
3rd Qu.:0.4375     3rd Qu.:1.950   3rd Qu.: 6.200   3rd Qu.:1.1200
Max.   :0.6600     Max.   :3.580   Max.   :13.000   Max.   :1.7100
OD280.OD315.of.diluted.wines  Proline
Min.   :1.270           Min.   : 278.0
1st Qu.:1.938           1st Qu.: 500.5
Median :2.780           Median : 673.5
Mean   :2.612           Mean   : 746.9
3rd Qu.:3.170           3rd Qu.: 985.0
Max.   :4.000           Max.   :1680.0

# skim gives an alternative to summary library(skimr)
# skim_with(integer=list(complete=NULL,missing=NULL,p0=NULL,p100=NULL,hist =
# NULL),numeric = list(complete=NULL,missing=NULL,p0=NULL,p100=NULL,hist = NULL))
# If you want specific columns to disappear from the output you use the following
```

```
# skim_with(integer=list(complete=NULL,missing=NULL,p0=NULL,p100=NULL,hist =
# NULL), numeric = list(complete=NULL,missing=NULL,p0=NULL,p100=NULL,hist =
# NULL)) devtools::install_github('haozhu233/kableExtra') library(kableExtra)
# kable(skim(wine))
```

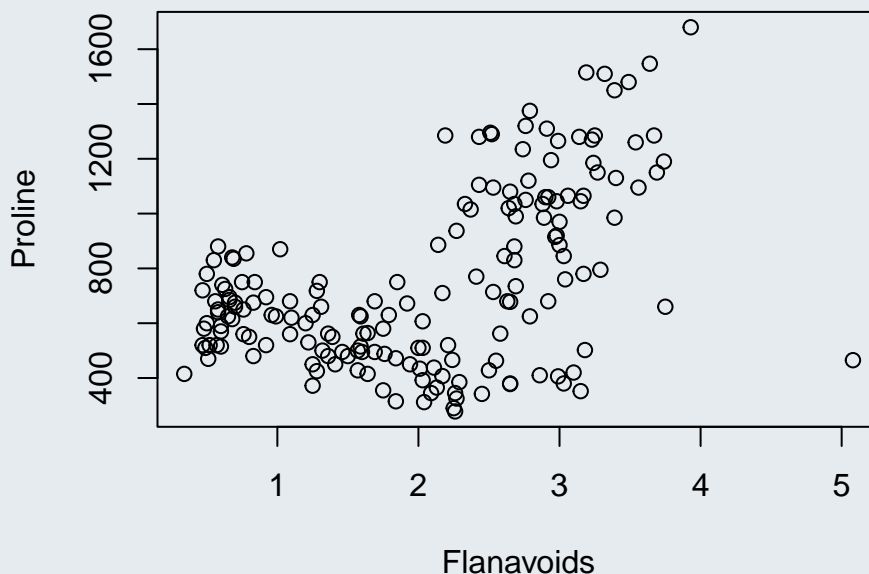
The first thing we always want to do with data is plot them to see if we can see any unusual features (particularly outliers and skewness which could affect our PCA).

```
pairs(wine[,c(1:2,5:8,14)],pch=20,lower.panel = NULL)
```



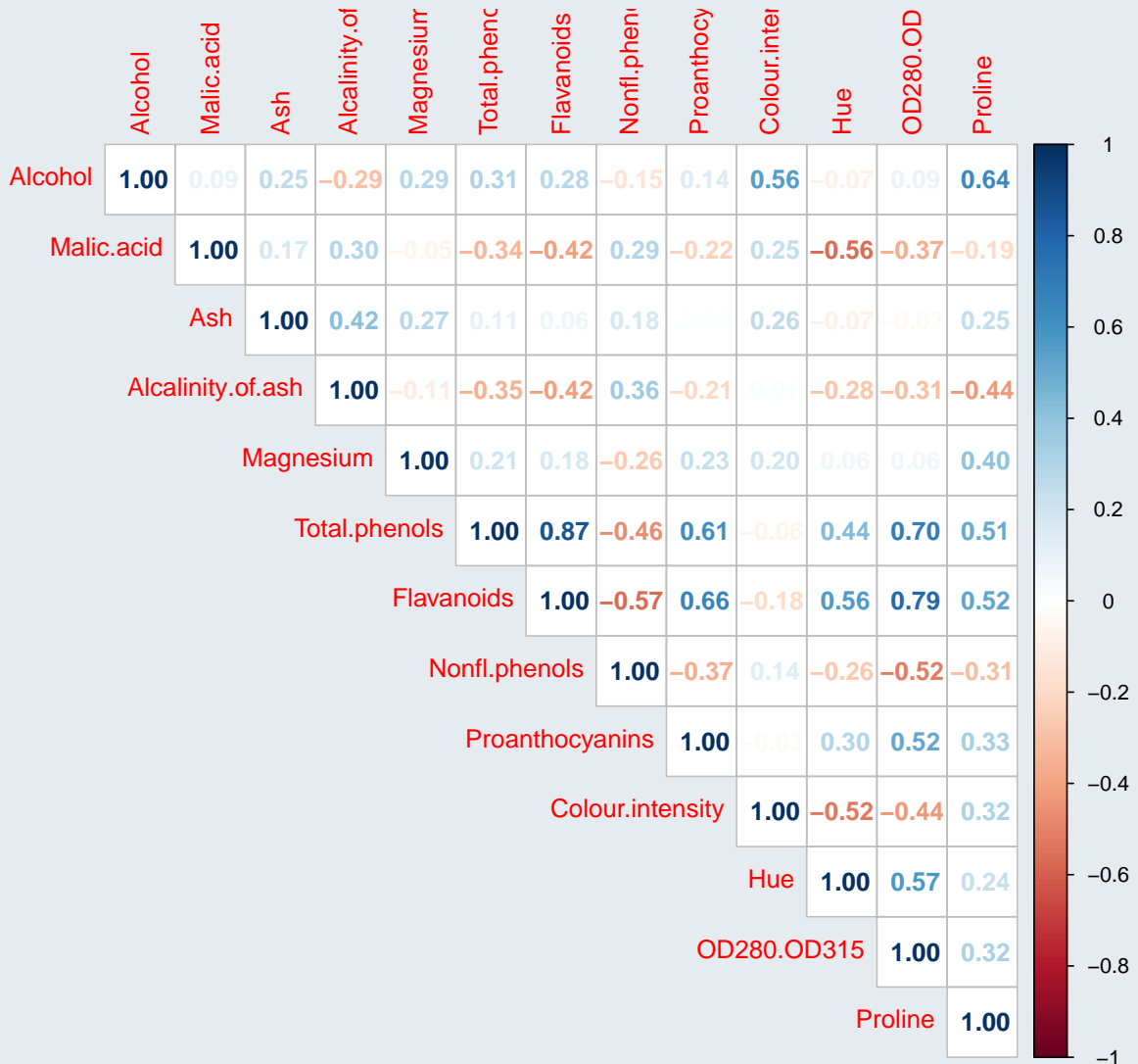
It's a little hard to see anything clearly but we can see that the first variable is not continuous (it only takes three values). We can also see potential outliers in a couple of scatterplots e.g. lower right corner of Flavanoids and Proline plot. There is evidence of some strong linear relationships between some pairs of variables, e.g. Total phenols and Flavanoids in particular. We can use the base command `plot` AND the `identify` command to look at the Flavanoids and Proline graph and identify the outlier (on the far right).

```
plot(wine$Flavanoids,wine$Proline,xlab="Flavanoids",ylab="Proline")
#Use the mouse to click on the point you want and the Esc key to stop the command
outlier<-identify(cbind(wine$Flavanoids,wine$Proline))
```



We can now remove the outlier (i.e. the 122th observation) and the first variable (i.e. Class) from our dataset and look at a correlation matrix.

```
wine.new<-wine[-122,-1]
library(corrplot);M <- cor(wine.new);corrplot(M, method = "number",type="upper")
```



Looking at the correlation matrix we see some fairly high correlations (0.87 between Flavanoids and Total phenols, 0.79 between Flavanoids and OD280 OD315 of diluted wines) and other correlations in the more moderate range. So it does seem realistic that some dimension reduction could be possible.

Should we use a correlation or covariance matrix in our PCA? Let's have a look at the variances first.

```
round(diag(var(wine.new)),2)
```

Alcohol	Malic.acid	Ash	Alcalinity.of.ash
0.65	1.25	0.07	10.75
Magnesium	Total.phenols	Flavanoids	Nonfl.phenols
203.03	0.39	0.95	0.02
Proanthocyanins	Colour.intensity	Hue	OD280.OD315
0.33	5.40	0.05	0.50
Proline			
99276.11			

Well, looking at the variances we can see huge differences between the variables (e.g. Ash and Proline, etc) which strongly suggests we use the correlation matrix.

```
wine.pca<-princomp(wine.new, cor=T)
wine.pca
```

```
Call:
princomp(x = wine.new, cor = T)
```

Standard deviations:

Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8
2.1861050	1.5899422	1.1601299	0.9617078	0.9282584	0.8004340	0.7439816	0.5794238
Comp.9	Comp.10	Comp.11	Comp.12	Comp.13			
0.5397900	0.5024390	0.4813398	0.4073751	0.2986393			

13 variables and 177 observations.

As expected, since we had 13 variables to start with (and 177 observations), we get 13 components. The standard deviation of the first component is large compared to the others but it is hard to see from this how many components we should retain.

In order to make this decision we have to decide which of the three methods we wish to use to decide on the number of retained components. For this example, we'll see all of them but remember, in practice, we should only ever have used one.

**Proportion of variance:** Let's say we wanted to decide on this based on proportion of variance and we wanted to have at least 90% of the original variability explained. We need to see the cumulative proportions of variance for all the components which we get using the `summary` command.

```
summary(wine.pca)
```

Importance of components:

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5
Standard deviation	2.1861050	1.5899422	1.1601299	0.96170782	0.92825842
Proportion of Variance	0.3676196	0.1944551	0.1035309	0.07114476	0.06628182
Cumulative Proportion	0.3676196	0.5620747	0.6656056	0.73675038	0.80303220
	Comp.6	Comp.7	Comp.8	Comp.9	Comp.10
Standard deviation	0.8004340	0.74398164	0.57942382	0.53978997	0.50243903
Proportion of Variance	0.0492842	0.04257759	0.02582554	0.02241332	0.01941884
Cumulative Proportion	0.8523164	0.89489400	0.92071953	0.94313286	0.96255170
	Comp.11	Comp.12	Comp.13		
Standard deviation	0.48133977	0.40737507	0.298639313		
Proportion of Variance	0.01782215	0.01276573	0.006860418		
Cumulative Proportion	0.98037386	0.99313958	1.000000000		

We can see here that if we wanted at least 90%, the smallest number of components that will give us that is 8 (as 7 only explain 89%).



### Task 2.

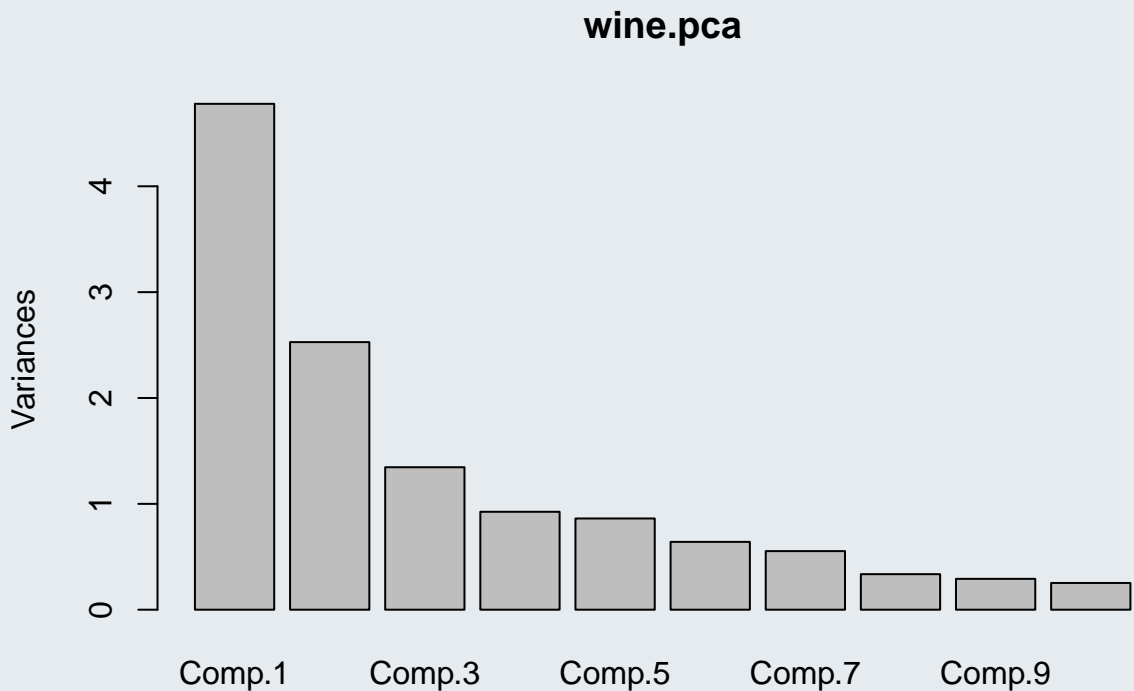
How many components should we have chosen if we wanted 80% of the original variability explained? What about 95%?



### Example 2.

**Cattell's method:** To use this method, we need to produce a scree plot of the proportion of variance versus component number. We expect this to drop off (since the eigenvalues are in decreasing order) but what we are looking for in the plot is a bend, where the rate of decrease suddenly reduces. The component number at the bend is the number we look to retain. Remember, because this method uses plot assessment, it is quite subjective and not unusual for different people to see different answers. We produce this using the `plot` command to the `princomp` fitted object.

```
plot(wine.pca)
```



This isn't an unambiguous plot, unfortunately. There are a couple of different places that we could say represent a change in rate of decrease: between components 3 and 4, or 5 and 6 or 7 and 8. If we wanted to really reduce the dimensionality we could argue the first one and say we are retaining 3 components.

**Kaiser's method:** Here we need to look at finding the average eigenvalue to discover which set of components have variation above it that we will retain. Because we used the correlation matrix, we know the average should be 1 but let's check.

```
#Extract the component standard deviations
```

```
sd.pca<-wine.pca$sdev
```

```
#Find the average variance, take the mean after squaring them
```

```
ave.var<-mean((sd.pca^2))
```

```
ave.var
```

```
[1] 1
```

```
#Find which components have higher than average variance (TRUE)
```

```
sd.pca^2>ave.var
```

```
Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9 Comp.10
  TRUE  TRUE  TRUE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE  FALSE
Comp.11 Comp.12 Comp.13
  FALSE  FALSE  FALSE
```

So based on this, we would retain the first 3 PCs.



### Task 3.

Some social scientists use Joliffe's rule, which says that for a PCA run on correlation, only those PCs with variation above 0.6 should be retained. Using a version of the previous code to pull out the variances of the PCs, find out how many PCs should be retained according to this rule.

The following video discusses interpreting loadings, calculating scores and the general set of questions asked when running a PCA.



### PCA Worked Example (Second Part)

[https://youtu.be/l\\_4cyUTmlkc](https://youtu.be/l_4cyUTmlkc)

Duration: 12m38s



#### Example 3.

Now we're going to look at if there is any interpretation that can be attached to the loadings of the PCA for our wine data. In order to do that we extract the loadings (or rotation) matrix from the fitted object.

Interpreting these can be something of an art. We look at each column in turn, look for variables with reasonably large positive values, group them together and then look for variables with reasonably large negative values and group them together. The PC is interpreted as being the contrast between these groups of variables. If all variables of reasonable size loading have the same sign then that PC is interpreted as the (weighted) average or overall score of these variables.

```
wine.pca$loadings[,1:8]
```

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5
Alcohol	0.149122514	0.485555741	0.17407847	0.009290005	0.26592996
Malic.acid	-0.242463819	0.223443550	-0.08786790	-0.540664071	-0.05603543
Ash	-0.009586478	0.328328599	-0.64011224	0.194647375	0.13900243
Alcalinity.of.ash	-0.250346241	-0.008123314	-0.61163461	-0.083786308	-0.08370602
Magnesium	0.138690160	0.299087400	-0.11171780	0.367268278	-0.72229570
Total.phenols	0.391800881	0.063885614	-0.13714253	-0.205480320	0.13876016
Flavanoids	0.426727328	-0.005371811	-0.11224538	-0.157668691	0.10673293
Nonfl.phenols	-0.298883085	0.030423001	-0.15662737	0.183269298	0.50008515
Proanthocyanins	0.309978301	0.037335041	-0.15316499	-0.399805096	-0.15844164
Colour.intensity	-0.087565645	0.524727427	0.17887557	-0.066407169	0.07868683
Hue	0.293857121	-0.275287273	-0.13045957	0.419157734	0.17838345
OD280.OD315	0.373092215	-0.164998633	-0.16727410	-0.188596177	0.09167382
Proline	0.287635764	0.363611710	0.09622339	0.227194542	0.16078018
	Comp.6	Comp.7	Comp.8		
Alcohol	0.206791287	0.06076710	0.461324360		
Malic.acid	0.516225841	-0.43803175	0.090514669		
Ash	0.141761650	0.14255139	-0.280914365		
Alcalinity.of.ash	-0.125541975	0.30971341	0.433857052		
Magnesium	0.027466546	-0.32614912	-0.095474838		
Total.phenols	-0.085524119	0.02038041	-0.390037222		
Flavanoids	-0.006399587	0.04580641	-0.129026484		
Nonfl.phenols	-0.277102460	-0.60022016	-0.173481138		
Proanthocyanins	-0.566623853	-0.33126816	0.286481235		
Colour.intensity	-0.399170503	0.23274848	-0.003260189		
Hue	0.081005629	-0.22090863	0.465274293		
OD280.OD315	0.267109729	0.02984656	-0.040043776		
Proline	0.108103550	-0.07375147	0.060731298		

`#summary(wine.pca)` Feel free to try this also

Recall that the blanks indicate zeros, i.e. variables with no weight in that particular component. We can see that there are two sets of variables with reasonable size weights here, ones with pluses and ones with minuses so we would interpret this component as the difference between the average of alcohol, magnesium, total phenols, flavanoids, proanthocyanins, hue, OD280 OD315 of diluted wines and proline and the average of malic acid, alcalinity of ash, nonflavanoid phenols and colour intensity. We can do the same with all other components that we have decided to retain.





#### Task 4.

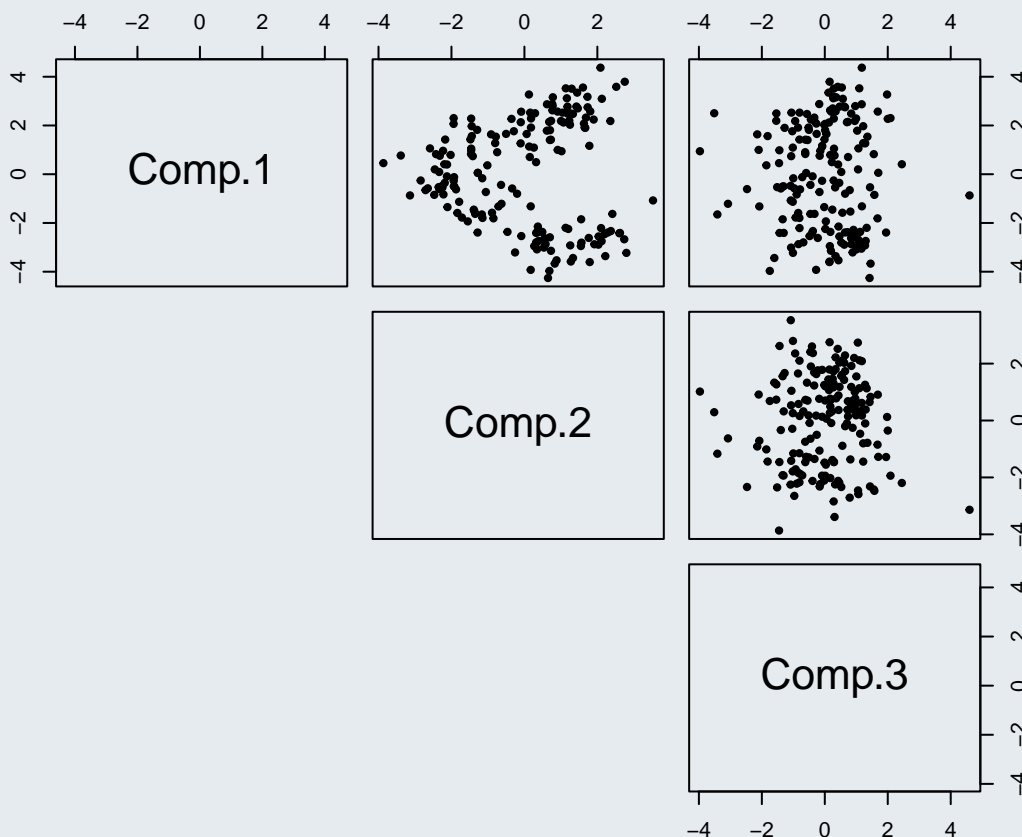
How would you interpret the third principal component in Example 3?



#### Example 4.

We are usually interested in examining what the data look like in the new reduced space. The scores for the PCs on the original data are automatically produced by `princomp`. So we can look at the pairs plot of the 3 PCs we decided to retain.

```
scores.wine<-wine.pca$scores[,1:3]
pairs(scores.wine,pch=20, lower.panel = NULL)
```



Interestingly, we see evidence of potentially two or more groups in the data (rather than a single homogeneous population). There may also be a couple of outliers. We could iterate this process by finding the outliers, removing them from the data and running PCA again. Often PCA is an iterative process, rather than a one and done kind of deal.



#### Task 5.

Try to calculate a first component score for the new observation (12, 4, 3, 25, 100, 2, 1, 0.4, 2, 4, 1, 2, 600) first by hand (using R as a calculator) and then using the `predict` command. You will need the centring vector, `wine.pca$center`, and the scaling vector, `wine.pca$scale`, as well as the first component loadings, `wine.pca$loadings[,1]`.

You will have to centre the new observation by taking away the centre vector. Then, because we used the correlation matrix and so we were working with standardised data, you have to scale the resulting centred vector by dividing by the scale vector. Finally you should take the inner product of the resulting standardised version of the new observation with the vector of first principal component loadings, resulting in the score.



### Example 5.

In this example we are going to look at a new data set on employment in 26 European countries but you are going to be doing all the work!

The data gives for each of 26 European countries the percentage of the total workforce in each in 1979 employed in nine different industries (Hand et al, 1994)

Variable name	Description
Agriculture	% employed in agriculture
Mining	% employed in mining
Manufacture	% employed in manufacturing
Power	% employed in power supply industries
Construction	% employed in construction
Service	% employed in service industries
Finance	% employed in finance
Social	% employed in social and personal services
Transport	% employed in transport & communications

```
employ<-read.table("eurojob.txt",header=T,row.names=1);head(employ,4)
```

```

      AGRIC MINING MANU POWER CONSTR SERVICE FINANCE SOCIAL TRANS
Belgium  3.3    0.9 27.6   0.9    8.2   19.1    6.2   26.6   7.2
Denmark  9.2    0.1 21.8   0.6    8.3   14.6    6.5   32.2   7.1
France  10.8   0.8 27.5   0.9    8.9   16.8    6.0   22.6   5.7
WGerm   6.7    1.3 35.8   0.9    7.3   14.4    5.0   22.3   6.1

```

First we want to explore the data. So we start by producing numerical summaries

```
summary(employ)
```

```

      AGRIC          MINING          MANU          POWER
Min.   : 2.70   Min.   :0.100   Min.   : 7.90   Min.   :0.1000
1st Qu.: 7.70   1st Qu.:0.525   1st Qu.:23.00   1st Qu.:0.6000
Median :14.45   Median :0.950   Median :27.55   Median :0.8500
Mean   :19.13   Mean   :1.254   Mean   :27.01   Mean   :0.9077
3rd Qu.:23.68   3rd Qu.:1.800   3rd Qu.:30.20   3rd Qu.:1.1750
Max.   :66.80   Max.   :3.100   Max.   :41.20   Max.   :1.9000

      CONSTR          SERVICE          FINANCE          SOCIAL
Min.   : 2.800   Min.   : 5.20   Min.   : 0.500   Min.   : 5.30
1st Qu.: 7.525   1st Qu.: 9.25   1st Qu.: 1.225   1st Qu.:16.25
Median : 8.350   Median :14.40   Median : 4.650   Median :19.65
Mean   : 8.165   Mean   :12.96   Mean   : 4.000   Mean   :20.02
3rd Qu.: 8.975   3rd Qu.:16.88   3rd Qu.: 5.925   3rd Qu.:24.12
Max.   :11.500   Max.   :19.10   Max.   :11.300   Max.   :32.40

      TRANS
Min.   :3.200
1st Qu.:5.700
Median :6.700
Mean   :6.546
3rd Qu.:7.075
Max.   :9.400

```

```

#skim gives an alternative to summary
#kable(skim(employ))

```

The next obvious step is to produce some graphical summaries.

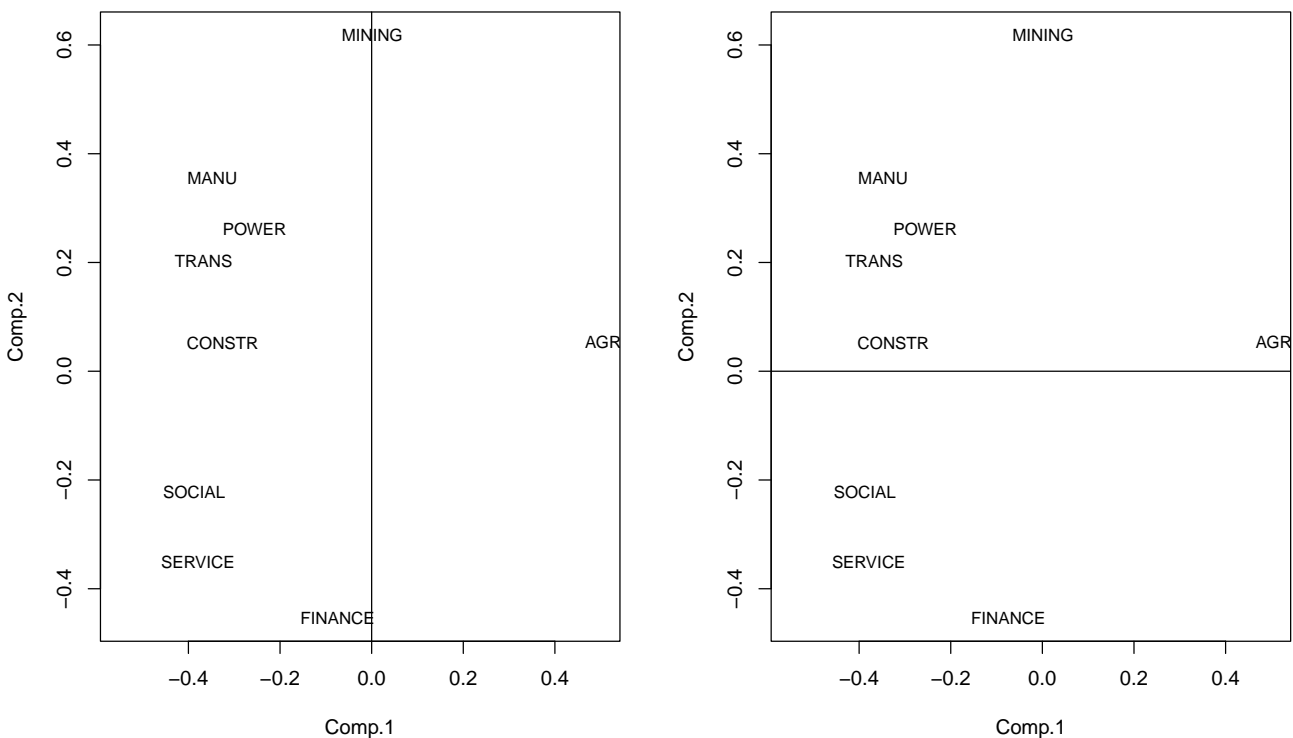


### Task 6.

- Produce a sensible plot or plots for these data and comment on them.
- Produce two important numerical summaries for deciding on how to run PCA and to tell how successful it is likely to be. Comment on these.
- Run PCA on the appropriate matrix and look at the output.
- Assuming we are most concerned with preserving information, how many coefficients should we retain if we want to have 90% of the original variability kept?
- Assuming we want to use Cattell's method, how many components would we retain?
- Assuming we want to use Kaiser's method, how many components would we retain?
- Assuming we have decided to retain 2 components, is there any useful interpretation to be had for these?

We can also visualise the loadings for each of the original variables on the new PC's in the following way (which can make interpretation easier).

```
#For the first two PC's
employ.pca<-princomp(employ,cor=T)
par(mfrow=c(1,2))
plot(employ.pca$loadings[,1:2],type="n",xlim=c(-.55,.5))
abline(v=0)
text(employ.pca$loadings[,1:2],colnames(employ),cex=0.8)
plot(employ.pca$loadings[,1:2],type="n",xlim=c(-.55,.5))
abline(h=0)
text(employ.pca$loadings[,1:2],colnames(employ),cex=0.8)
```



From the first plot, looking along the horizontal axis for interpretation of the first PC's loadings, we see that Finance and Mining are near or on the zero line and so don't come into play. The other sets of variables are on either side of the line, indicating a difference between two averages. We can do something similar with the second plot for the second PC.



### Task 7.

Say we have the following entries for our observations

```
obs1<-c(5.1,0.5,32.3,0.8,8.1,16.7,4.3,21.2,6.3)
obs2<-c(4.2,0.7,25.4,0.7,9.3,15.0,5.8,31.0,6.9)
```

Calculate the scores for the two new observations and produce a scatterplot of the data's scores for the first 2 PCs and comment.



### Task 8.

The following code will produce the PCA output given in the last 2 videos. Try re-running the code using `prcomp` instead of `princomp`. (Just start with the data with the outlier removed)

```
#Setting the random generator seed to ensure similar responses when re-running code
set.seed(135)
#####
#Reading in and preparing the data
#####
#Open the library ade3 where the data is
library(ade4)
#Load the tortues dataset
data("tortues")
#Look at the first few lines of the data
head(tortues)
#Extract the females turtles data into a new dataset called fem.turt
fem.turt<-tortues[tortues[,4]=="F",-4]
#Take the log of all the variables in the new dataset
log.fem.turt<-log(fem.turt)
#Name the variables
colnames(log.fem.turt)<-c("log.length","log.width", "log.breadth")

#####
#Summary Plots
#####
#Create a pairsplot of the data
pairs(log.fem.turt,pch=20, lower.panel=NULL)
#Create a 3-d scatterplot of the data
library(lattice)
cloud(log.length~log.width*log.breadth,data=log.fem.turt)
#Rotate the 3-d scatterplot of the data
library(TeachingDemos)
#Use your mouse to drag the sliders to change the plot
rotate.cloud(log.length~log.width*log.breadth,data=log.fem.turt)

#####
#Numerical Summaries
#####
#Correlation matrix
round(cor(log.fem.turt),2)
#Standard deviations
apply(log.fem.turt,2,sd)

#####
#Principal Component Analysis
#####
pca.turt<-princomp(log.fem.turt);pca.turt

#####
#Looking at the scores
#####
head(pca.turt$scores);plot(pca.turt$scores,pch=20)
```

```

#outlier<-identify(pca.turt$scores)

#####
#Run PCA on dataset excluding outlier
#####
pca.turt.new<-princomp(log.fem.turt[-10,]);pca.turt.new

#####
#Deciding on number of PCs to retain
#####
plot(pca.turt.new);summary(pca.turt.new)
sd.pca<-summary(pca.turt.new)$sdev
tot.var<-sum(sd.pca^2)
ave.var<-tot.var/ncol(log.fem.turt)
ave.var
sd.pca^2>ave.var
#####
#Interpreting the loadings
#####
pca.turt.new$loadings
#####
#Calculating new scores
#####
new.data<-data.frame(log.length=c(4.8),log.width=c(4.7),log.breadth=c(3.9))
predict(pca.turt.new,new.data)

```

## Learning outcomes for Week 1

- Intuition behind the need for dimensionality reduction in the data.
- Understanding of the Principal Component Analysis (PCA) technique.
- Performing PCA in R and interpreting its output.
- Different ways of choosing the number of principal components.



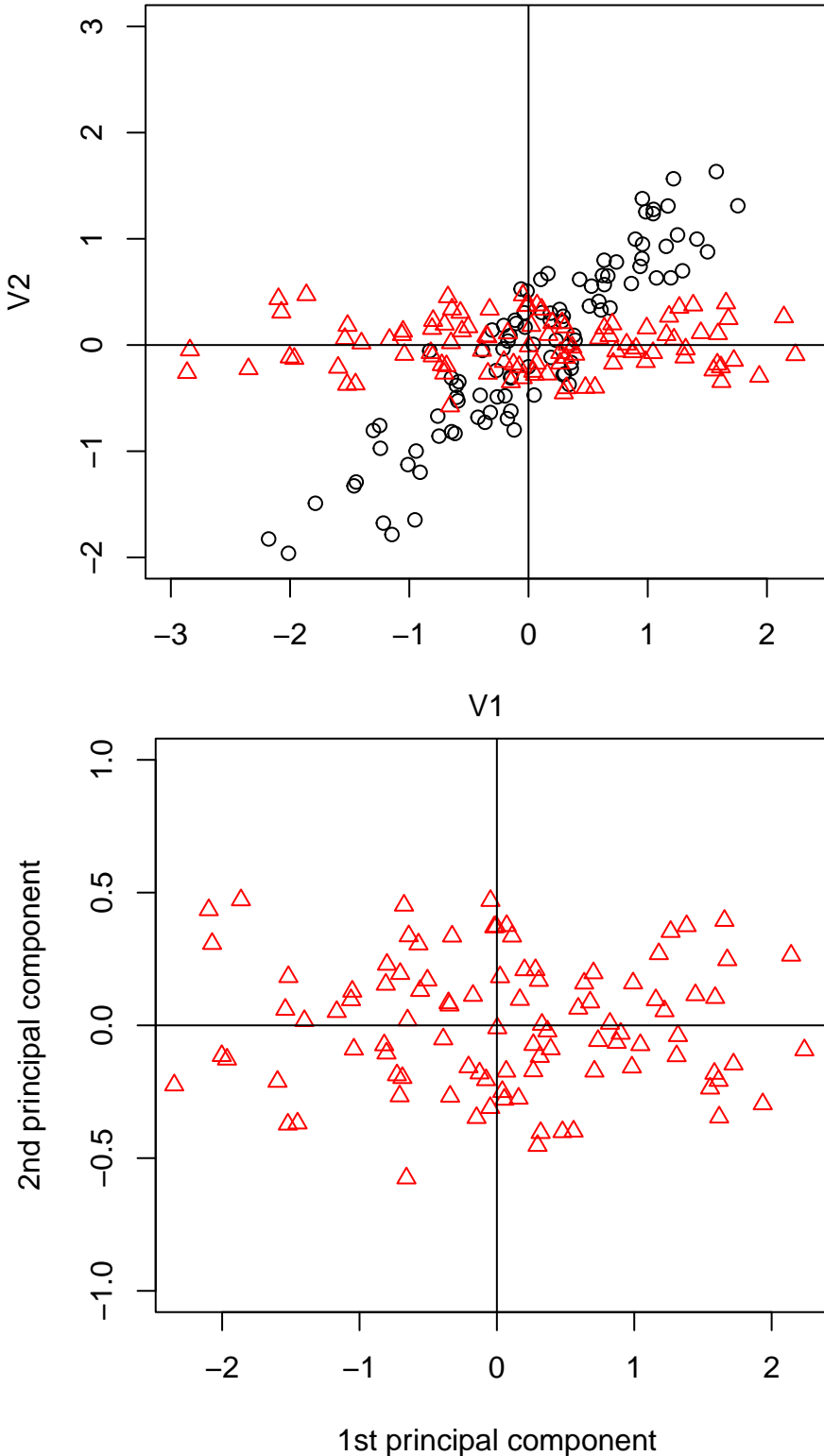
### Supplementary material:

For some additional material, have a look at:

1. Practical Guide to Principal Component Analysis (PCA) in R & Python
2. PCA Tutorial in Python (with nice image analysis example)
3. Principal Component Analysis in SAS
4. Tutorial on PCA in SAS
5. Principal Components: Mathematics, Example, Interpretation
6. Interactive explanation of PCA
7. A more mathematical point of view on PCA

## Answers to tasks

*Answer to Task 1.* The orientation of the data will no longer look as before. The points lying close to a diagonal line indicate strong correlation between the original variables and principal components have no correlation. Our data should look like a cloud of points that no longer follow a line but just look like a random scatter of points centred at the origin (0, 0) in the new space. The first graph superimposes the principal component scores data (shown in red triangles) on the original graph; while the second graph shows only the principal component scores data.



*Answer to Task 2.* Looking at the Cumulative Proportion line we see:

- for 80% we would only need 5 components

- for 95% we would need 10 components.

*Answer to Task 3.* Applying that rule in R we have:

```
wine.pca$sdev^2
  Comp.1   Comp.2   Comp.3   Comp.4   Comp.5   Comp.6   Comp.7
4.77905526 2.52791634 1.34590144 0.92488193 0.86166370 0.64069465 0.55350868
  Comp.8   Comp.9   Comp.10  Comp.11  Comp.12  Comp.13
0.33573196 0.29137321 0.25244498 0.23168797 0.16595445 0.08918544

wine.pca$sdev^2>0.6
  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8  Comp.9  Comp.10
  TRUE   TRUE   TRUE   TRUE   TRUE   TRUE   FALSE   FALSE   FALSE   FALSE
Comp.11  Comp.12  Comp.13
  FALSE  FALSE  FALSE
```

So we see that according to the new rule, we would retain 6 components.

*Answer to Task 4.* Although there are a large number of positive and negative non-zero loadings, the really dominant ones in terms of size (approximately 0.6) are ash and alkalinity of ash so this component mainly seems to be interested in ash and its properties in wine.

*Answer to Task 5.* To calculate the first component score

```
#By hand using R as a calculator
new.x<-matrix(c(12,4,3,25,100,2,1,0.4,2,4,1,2,600),nrow=1)
colnames(new.x)<-colnames(wine.new)
centre.wine<-wine.pca$center
scale.wine<-wine.pca$scale
first.load<-wine.pca$loadings[,1]

new.x.cent<-new.x-centre.wine
new.x.stand<-new.x.cent/scale.wine
new.x.stand%*%first.load
```

```
  [,1]
[1,] -1.852166
```

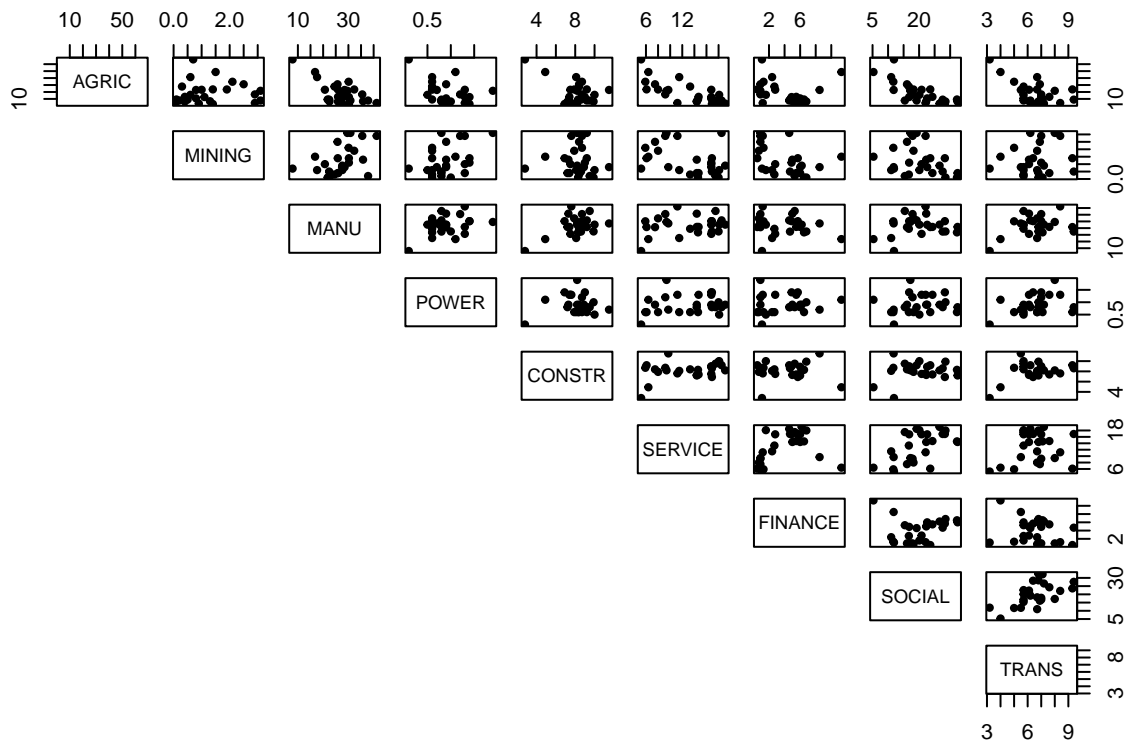
```
#Using the predict command
```

```
predict(wine.pca,as.data.frame(new.x))
  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7
[1,] -1.852166 0.196511 -2.909757 -0.290249 -0.5356328 0.1049906 -0.488592
  Comp.8  Comp.9  Comp.10  Comp.11  Comp.12  Comp.13
[1,] 0.1793053 -1.296705 -0.7004643 0.4829155 0.1600023 0.7121765
```

We can see the answer by hand is the same as the first element of the predict result. If we were using PCA on the covariance matrix, we would skip the scaling stage (only centering before taking the inner product).

*Answer to Task 6.* Let's first produce the pairs plot in R

```
#Look at the pairs plots for the data
pairs(employ,pch=20, lower.panel=NULL)
```



Comment: There are a couple of possible outliers. (We'll leave these for the moment and see if they appear again in our scores plot in the end for whether we need to re-run PCA without them included.) The data doesn't appear to fall into groups. There seem to be some linear relationships between pairs of variables but they are not very strong.

It's important to have a look at the correlation and variance of the variables.

```
#Display the correlation matrix to 2 d.p.s
#library(matrixcalc)
round(cor(employ),2)
```

	AGRIC	MINING	MANU	POWER	CONSTR	SERVICE	FINANCE	SOCIAL	TRANS
AGRIC	1.00	0.04	-0.67	-0.40	-0.54	-0.74	-0.22	-0.75	-0.56
MINING	0.04	1.00	0.45	0.41	-0.03	-0.40	-0.44	-0.28	0.16
MANU	-0.67	0.45	1.00	0.39	0.49	0.20	-0.16	0.15	0.35
POWER	-0.40	0.41	0.39	1.00	0.06	0.20	0.11	0.13	0.38
CONSTR	-0.54	-0.03	0.49	0.06	1.00	0.36	0.02	0.16	0.39
SERVICE	-0.74	-0.40	0.20	0.20	0.36	1.00	0.37	0.57	0.19
FINANCE	-0.22	-0.44	-0.16	0.11	0.02	0.37	1.00	0.11	-0.25
SOCIAL	-0.75	-0.28	0.15	0.13	0.16	0.57	0.11	1.00	0.57
TRANS	-0.56	0.16	0.35	0.38	0.39	0.19	-0.25	0.57	1.00

```
#Look at the standard deviations of the variables
round(sqrt(diag(cov(employ))),1)
```

AGRIC	MINING	MANU	POWER	CONSTR	SERVICE	FINANCE	SOCIAL	TRANS
15.5	1.0	7.0	0.4	1.6	4.6	2.8	6.8	1.4

Comment: There are some strong correlation (around -0.7) but a lot of weak (close to zero) correlations as well. It may be possible to achieve some dimension reduction here but not a lot. The variation for some variables is much bigger than for others, therefore we should use the correlation matrix and *not* the covariance matrix in PCA.

We can use the princomp command.

```
employ.pca<-princomp(employ,cor=T)
employ.pca
```

Call:

```
princomp(x = employ, cor = T)
```

Standard deviations:

Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6
1.867391569	1.459511268	1.048311791	0.997237674	0.737033056	0.619215363



```

      Comp.7      Comp.8      Comp.9
0.475135828 0.369851221 0.006754636

```

9 variables and 26 observations.

The summary function can give us information about how many components we should keep.

```
summary(employ.pca)
```

Importance of components:

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5
Standard deviation	1.8673916	1.4595113	1.0483118	0.9972377	0.73703306
Proportion of Variance	0.3874613	0.2366859	0.1221064	0.1104981	0.06035753
Cumulative Proportion	0.3874613	0.6241472	0.7462536	0.8567517	0.91710919

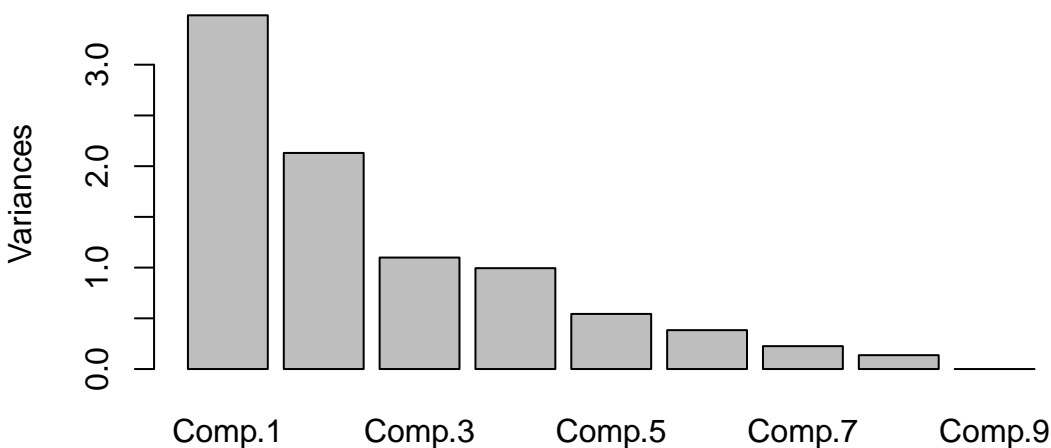
	Comp.6	Comp.7	Comp.8	Comp.9
Standard deviation	0.61921536	0.47513583	0.36985122	6.754636e-03
Proportion of Variance	0.04260307	0.02508378	0.01519888	5.069456e-06
Cumulative Proportion	0.95971227	0.98479605	0.99999493	1.000000e+00

Looking at the Cumulative Proportion line, it is clear that we need to keep 5 components to retain 90% variability.

For Cattell's method we will need to produce a scree plot.

```
plot(employ.pca)
```

### employ.pca



There are some different possibilities here, depending on how you read the graph. Suggesting 2, 4 or 6 components seems reasonable as that's where we see sudden drops/leveling off in variation.

For Kaiser's method we need to look at finding the average eigenvalue to discover which set of components have variation above it that we will retain.

```
employ.pca$sdev^2>1
```

Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Comp.9
TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Here we have cheated a bit, since we know the average variance is going to be 1 whenever we use the correlation matrix. We can see that we would retain the first 3 components in this case.

Let's have a look at the loadings from each component first.

```
employ.pca$loadings
```

Loadings:

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Comp.9
AGRIC	0.524				0.213	0.153			0.806
MINING		0.618	-0.201		-0.164	-0.101	-0.726		
MANU	-0.347	0.355	-0.150	-0.346	-0.385	-0.288	0.479	0.126	0.366
POWER	-0.256	0.261	-0.561	0.393	0.295	0.357	0.256	-0.341	
CONSTR	-0.325		0.153	-0.668	0.472	0.130	-0.221	-0.356	

SERVICE	-0.379	-0.350	-0.115		-0.284	0.615	-0.229	0.388	0.238
FINANCE		-0.454	-0.587		0.280	-0.526	-0.187	0.174	0.145
SOCIAL	-0.387	-0.222	0.312	0.412	-0.220	-0.263	-0.191	-0.506	0.351
TRANS	-0.367	0.203	0.375	0.314	0.513	-0.124		0.545	

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Comp.9
SS loadings	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Proportion Var	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
Cumulative Var	0.111	0.222	0.333	0.444	0.556	0.667	0.778	0.889	1.000

- Component 1 seems to be the difference between the average of manufacturing, power, construction, service, social and transportation industries, and the agricultural industry. So this new variable will distinguish between countries with agricultural economies and those with industrial economies.
- Component 2 seems to be the difference between the average of mining, manufacturing, power and transportation industries, and the average of service, finance and social industries. So this new variable distinguishes between countries with relatively large and relatively small service sectors.

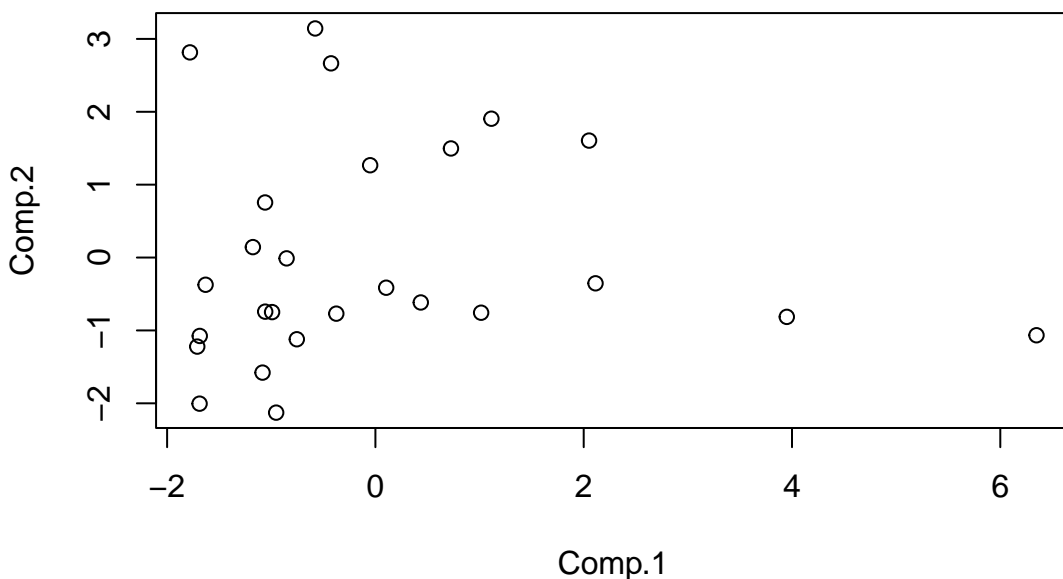
*Answer to Task 7.* We can calculate the scores using the following code

```
newdata<-rbind(obs1,obs2);colnames(newdata)<-colnames(employ)
new.data<-as.data.frame(newdata);new.data.scores<-predict(employ.pca,new.data)
new.data.scores[, 1:6]
```

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6
obs1	-0.9890221	-0.7608272	-0.01661647	-0.46329452	-0.80507442	0.03884813
obs2	-1.4684276	-1.3729179	0.70027077	0.02475385	-0.03188834	-0.66059019

Let's produce the scatterplot first.

```
employ.scores2<-as.data.frame(employ.pca$scores[,1:2])
plot(employ.scores2$Comp.1,employ.scores2$Comp.2,xlab="Comp.1", ylab="Comp.2")
```



There definitely seems to be an issue with at least one outlier. It would be worth identifying and removing it/them and re-running PCA to see if it affects the results.

*Answer to Task 8.* The following code uses `prcomp` to analyse the turtle data.

```
#Setting the random generator seed to ensure similar responses when re-running code
set.seed(135)
#####
#Principal Component Analysis
#####
pca.turt.2<-prcomp(log.fem.turt[-10,]);pca.turt.2
```

```
#####
#Deciding on number of PCs to retain
#####
plot(pca.turt.2);summary(pca.turt.2)
sd.pca<-summary(pca.turt.2)$sdev
tot.var<-sum(sd.pca^2)
ave.var<-tot.var/ncol(log.fem.turt)
ave.var
sd.pca^2>ave.var

#####
#Interpreting the loadings and calculating new scores
#####
pca.turt.2$rotation
new.data<-data.frame(log.length=c(4.8),log.width=c(4.7),log.breadth=c(3.9))
predict(pca.turt.2,new.data)

So not too different!
```