Unsupervised Methods

Juan R Gonzalez

BRGE - Bioinformatics Research Group in Epidemiology
Barcelona Institute for Global Health (ISGlobal)
e-mail:juanr.gonzalez@isglobal.org
http://brge.isglobal.org
and Departament of Mathematics, UAB

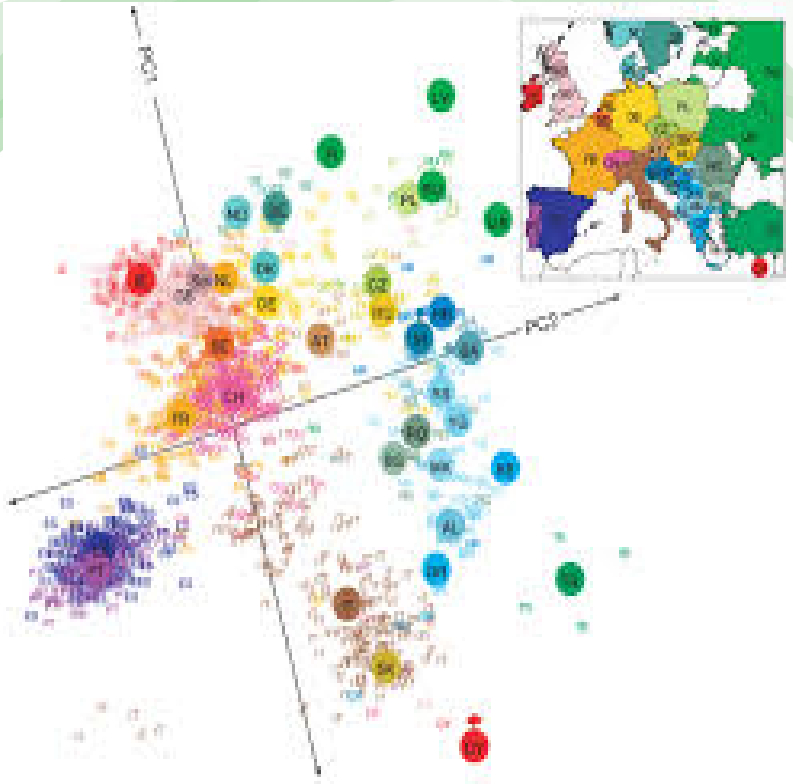
Outline

Outline

Principal component analysis

- The goal is to find a small number of independent linear combinations (principal components) of a set of measured variables that capture as much of the variability in the original variables as possible.
- Supplementary variables can also be considered. They are not included in the calculation of principal components and including them does not affect the results. The supplementary variables are projected on to the loading plot and used to enhance interpretation.
- It is a useful exploratory technique and can help you to create predictive models. For instance, in disease association studies, principal components are regressed against health outcomes.

Principal component analysis



Principal component analysis

- It is used in genomic, transcriptomic or epidemiological studies to find patterns (signatures) with regard to variantes, abundance of mRNAs or exposure/diet, respectively.
- Given a data set X, which is a $n \times p$ matrix, of n individuals and p features

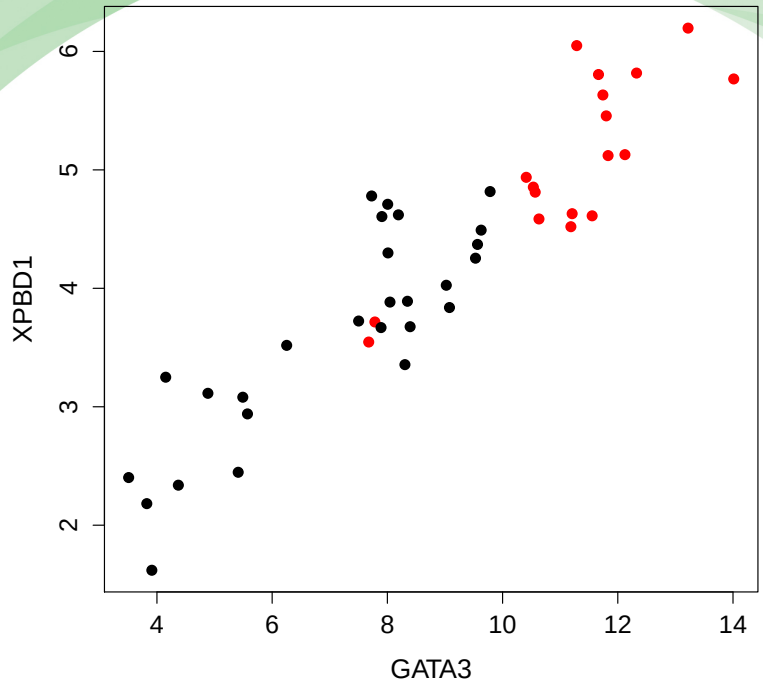
$$X = (x_1, x_2, \ldots, x_p)$$

- we look for new variables that are linear combinations of the original variables $f = q_1 X_1 + q_2 X_2 + \ldots + q_p X_p$ or $f = Xq$ where q are know as loadings.
- We introduce the restriction that for ith component, q's should maximize the variance components of f's

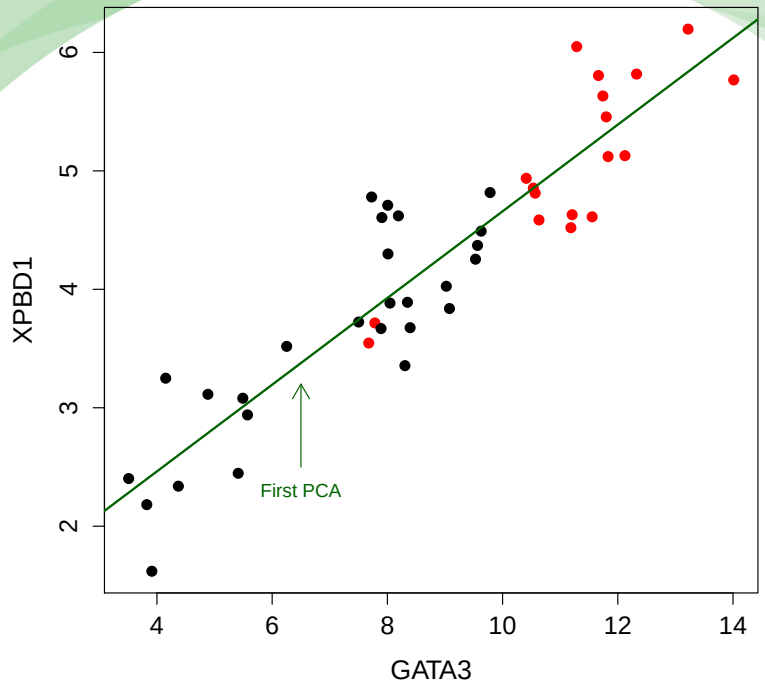
$$\arg\max_{q^i} \text{var}(Xq^i)$$

and q's has to be orthogonal to each other.

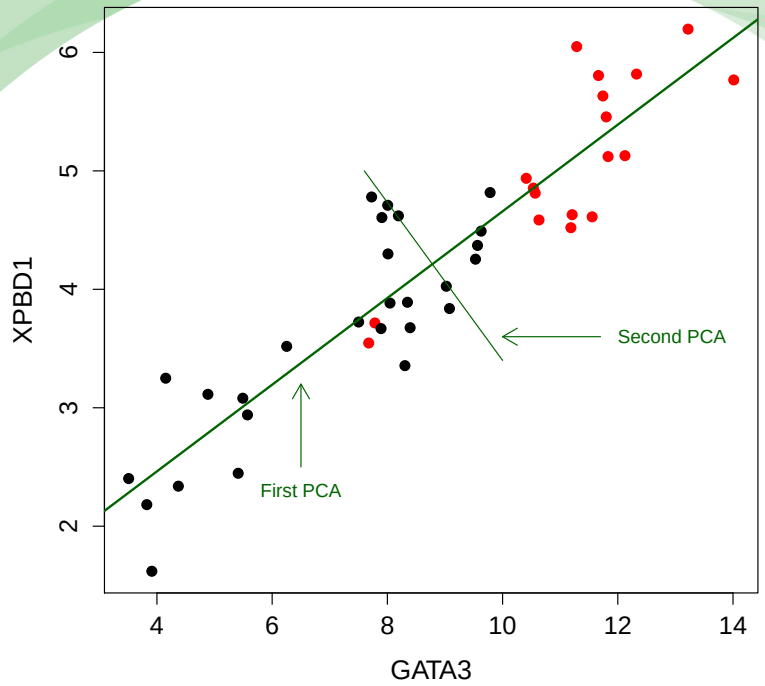
Principal component analysis



Principal component analysis



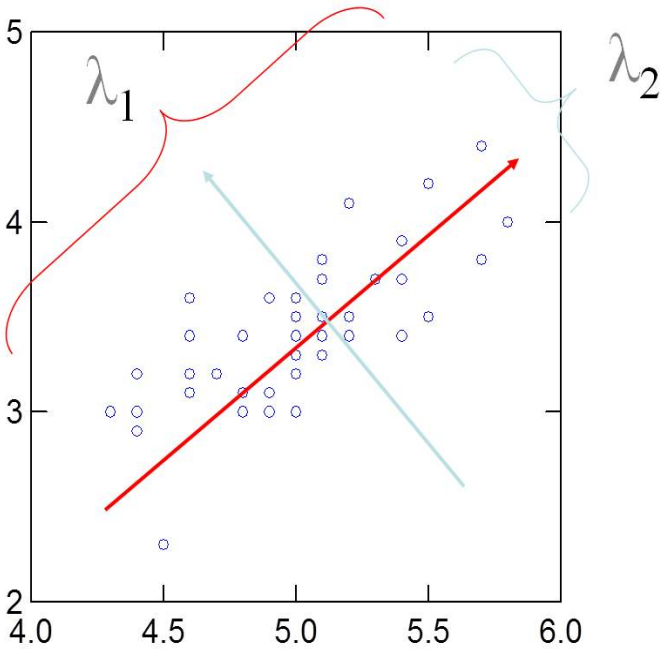
Principal component analysis



Principal component analysis

- Correlations between variables and the principal axes are known as **loadings**
- Each element represents the contribution of a given variable to a component (**eigenvalues**)

Principal component analysis



PCA: How many axes are needed?

- Does the $(k+1)^{th}$ principal axis represent more variance that would e expected by chance?
- Several tests and rules have been proposed (**Hornś method** and bootstrap approach)
- A common *rule of thumb*, when PCA is based on correlations, is that axes with eigenvalues > 1 are worth interpreting

PCA: non-linear relationships

- PCA assumes relationships among variables are LINEAR
- If the sctructure in the data is NONLINEAR (the cloud of points twists and curves its way through *p*-dimensional space) the principal axes will not be an efficient and informative summary of the data
- Use Principal Curve Analysis

Principal components analysis with R

```
require(graphics)
data(USArrests)
head(USArrests)

##            Murder Assault UrbanPop Rape
## Alabama      13.2     236       58 21.2
## Alaska       10.0     263       48 44.5
## Arizona       8.1     294       80 31.0
## Arkansas      8.8     190       50 19.5
## California    9.0     276       91 40.6
## Colorado      7.9     204       78 38.7
```

Principal components analysis with R

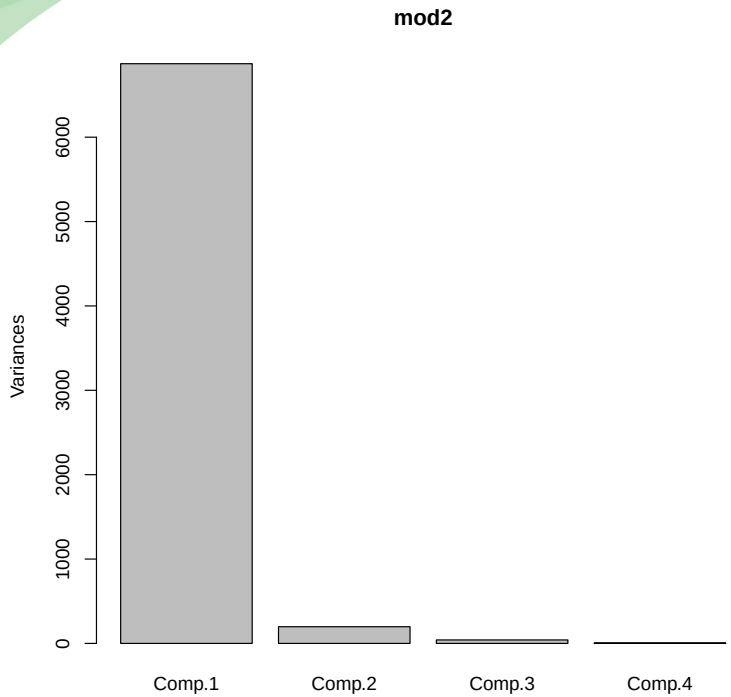
```
princomp(USArrests)

## Call:
## princomp(x = USArrests)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4
## 82.890847 14.069560  6.424204  2.457837
##
##  4  variables and  50 observations.

mod <- princomp(USArrests)
summary(mod)

## Importance of components:
##                             Comp.1      Comp.2      Comp.3       Com
## Standard deviation     82.8908472 14.06956001 6.424204055 2.4578367
## Proportion of Variance  0.9655342  0.02781734 0.005799535 0.0008489
## Cumulative Proportion   0.9655342  0.99335156 0.999151092 1.0000000
```

Principal components analysis with R

```
mod2 <- princomp(USArrests)
summary(mod2)

## Importance of components:
##                             Comp.1      Comp.2      Comp.3       Com
## Standard deviation     82.8908472 14.06956001 6.424204055 2.4578367
## Proportion of Variance  0.9655342  0.02781734 0.005799535 0.0008489
## Cumulative Proportion   0.9655342  0.99335156 0.999151092 1.0000000
```

Principal components analysis with R

```
plot(mod2)
```



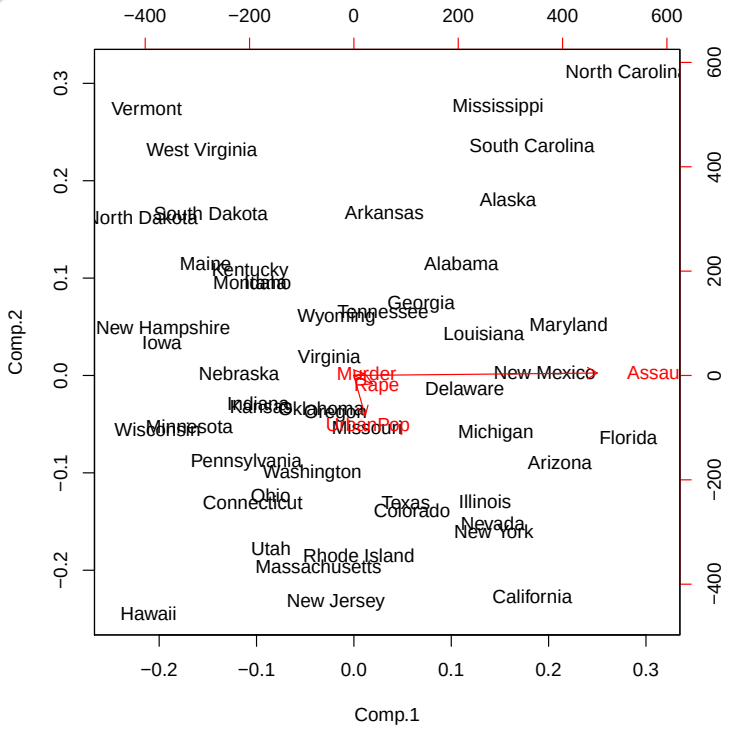
Principal components analysis with R

Help interpreting the results

```
loadings(mod2)

##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4
## Murder                          0.995
## Assault   0.995
## UrbanPop         -0.977 -0.201
## Rape             -0.201  0.974
##
##                Comp.1 Comp.2 Comp.3 Comp.4
## SS loadings      1.00   1.00   1.00   1.00
## Proportion Var   0.25   0.25   0.25   0.25
## Cumulative Var   0.25   0.50   0.75   1.00
```

Principal components analysis with R

```
biplot(mod2)
```

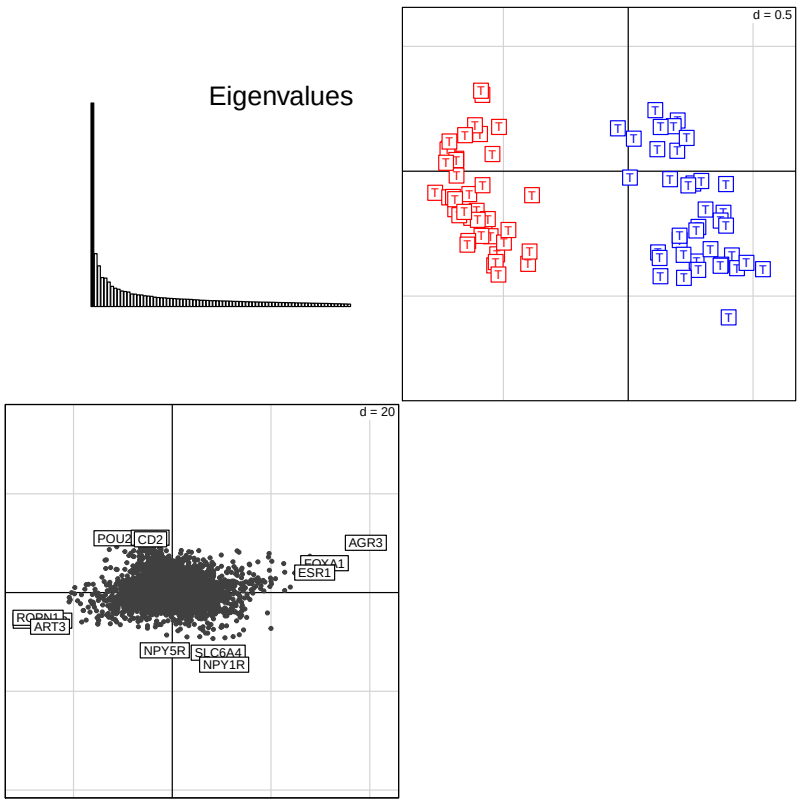


PCA improving visualization

- Data from the Cancer Genome Atlas (TCGA) will be analyzed.
- A subset of the TCGA breast cancer study from Nature 2012 publication have been selected.
- Data `https://tcga-data.nci.nih.gov/docs/publications/brca_`
- Available data are: miRNA, miRNAprecursor, RNAseq, Methylation, proteins from a RPPA array, and GISTIC SNP calls (CNA and LOH). Clinical data are also available.
- We are interested in comparing women with ER+ vs ER-.

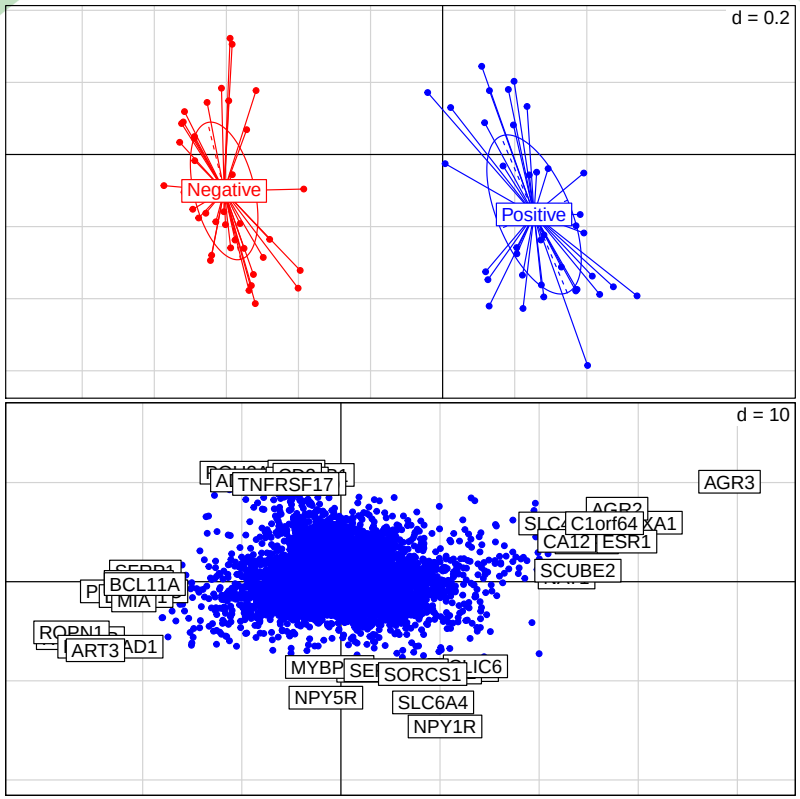
PCA improving visualization

```
library(made4)
load("data/breast_TCGA.RData")
group <- droplevels(breast_multi$clin$ER.Status)
rnaseq <- breast_multi$RNAseq
out <- ord(rnaseq, trans=FALSE, type="pca", classvec=group)
plot(out, nlab=3, arraylabels=rep("T", 79))
```



PCA improving visualization

```
par(mfrow=c(2,1))
plotarrays(out$ord$co, classvec=group)
plotgenes(out, col="blue")
```



PCA improving visualization

A list of variables with higher loadings on axes can be obtained using the following function

```
ax1 <- topgenes(out, axis=1, n=5, ends="pos")
ax2 <- topgenes(out, axis=2, n=5, ends="neg")
cbind(pos=ax1, neg=ax2)

##       pos        neg
## [1,] "AGR3"    "NPY1R"
## [2,] "FOXA1"   "SLC6A4"
## [3,] "ESR1"    "NPY5R"
## [4,] "AGR2"    "SORCS1"
## [5,] "C1orf64" "CST9L"
```

PCA requirements

- Data scale

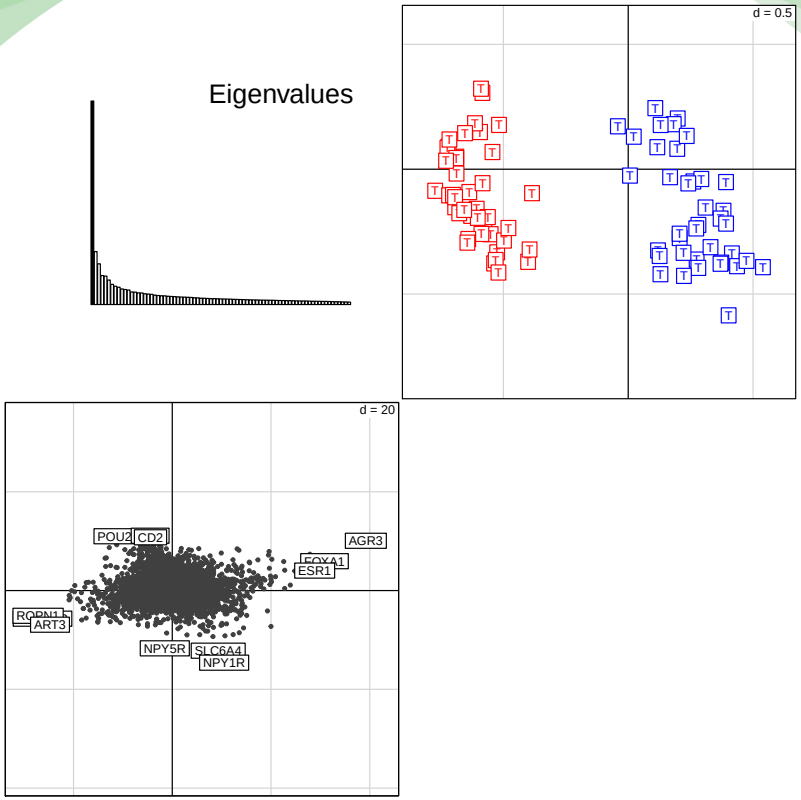
```
rnaseq.s <- scale(rnaseq, center= TRUE, scale = TRUE)
```

- Complete cases

```
library(impute)
rnaseq.s.imp <- impute.knn(rnaseq.s, rowmax = 0.5,
                           colmax = 0.8)$data #samples in columns!!!

## Cluster size 10020 broken into 1909 8111
## Cluster size 1909 broken into 325 1584
## Done cluster 325
## Cluster size 1584 broken into 1498 86
## Done cluster 1498
## Done cluster 86
## Done cluster 1584
## Done cluster 1909
## Cluster size 8111 broken into 5038 3073
## Cluster size 5038 broken into 1351 3687
## Done cluster 1351
## Cluster size 3687 broken into 1738 1949
## Cluster size 1738 broken into 768 970
## Done cluster 768
## Done cluster 970
## Done cluster 1738
## Cluster size 1949 broken into 981 968
## Done cluster 981
## Done cluster 968
```

PCA requirements

```
out <- ord(rnaseq.s.imp, trans=FALSE, type="pca", classvec=group)
plot(out, nlab=3, arraylabels=rep("T", 79))
```



Variance explained

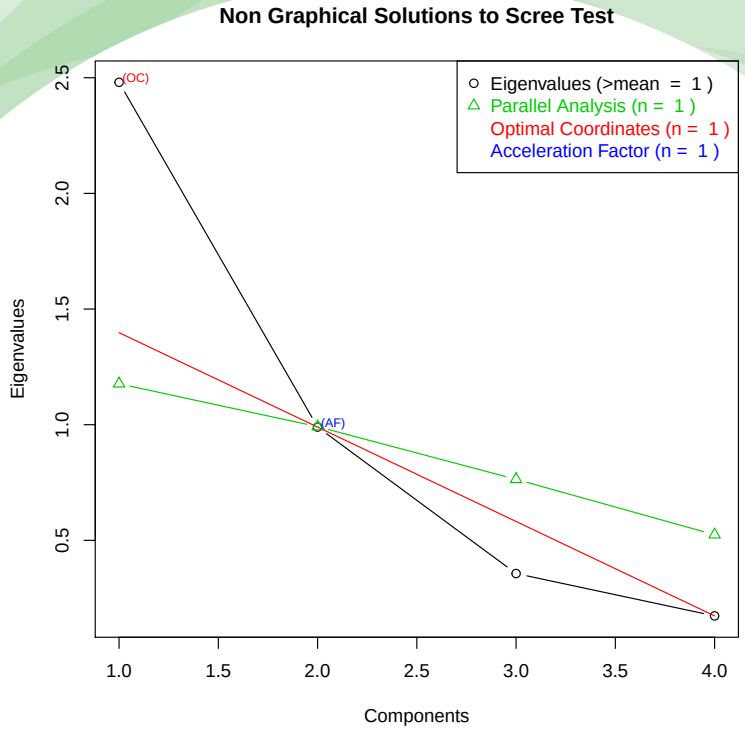
```
summary(out$ord)

## Class: pca dudi
## Call: dudi.pca(df = data.tr, scannf = FALSE, nf = ord.nf)
##
## Total inertia: 79
##
## Eigenvalues:
##     Ax1     Ax2     Ax3     Ax4     Ax5
##  18.526   4.817   3.709   2.640   2.591
##
## Projected inertia (%):
##     Ax1     Ax2     Ax3     Ax4     Ax5
##  23.450   6.097   4.695   3.342   3.279
##
## Cumulative projected inertia (%):
##     Ax1   Ax1:2   Ax1:3   Ax1:4   Ax1:5
##   23.45   29.55   34.24   37.58   40.86
##
## (Only 5 dimensions (out of 79) are shown)
```

PCA number of significant components

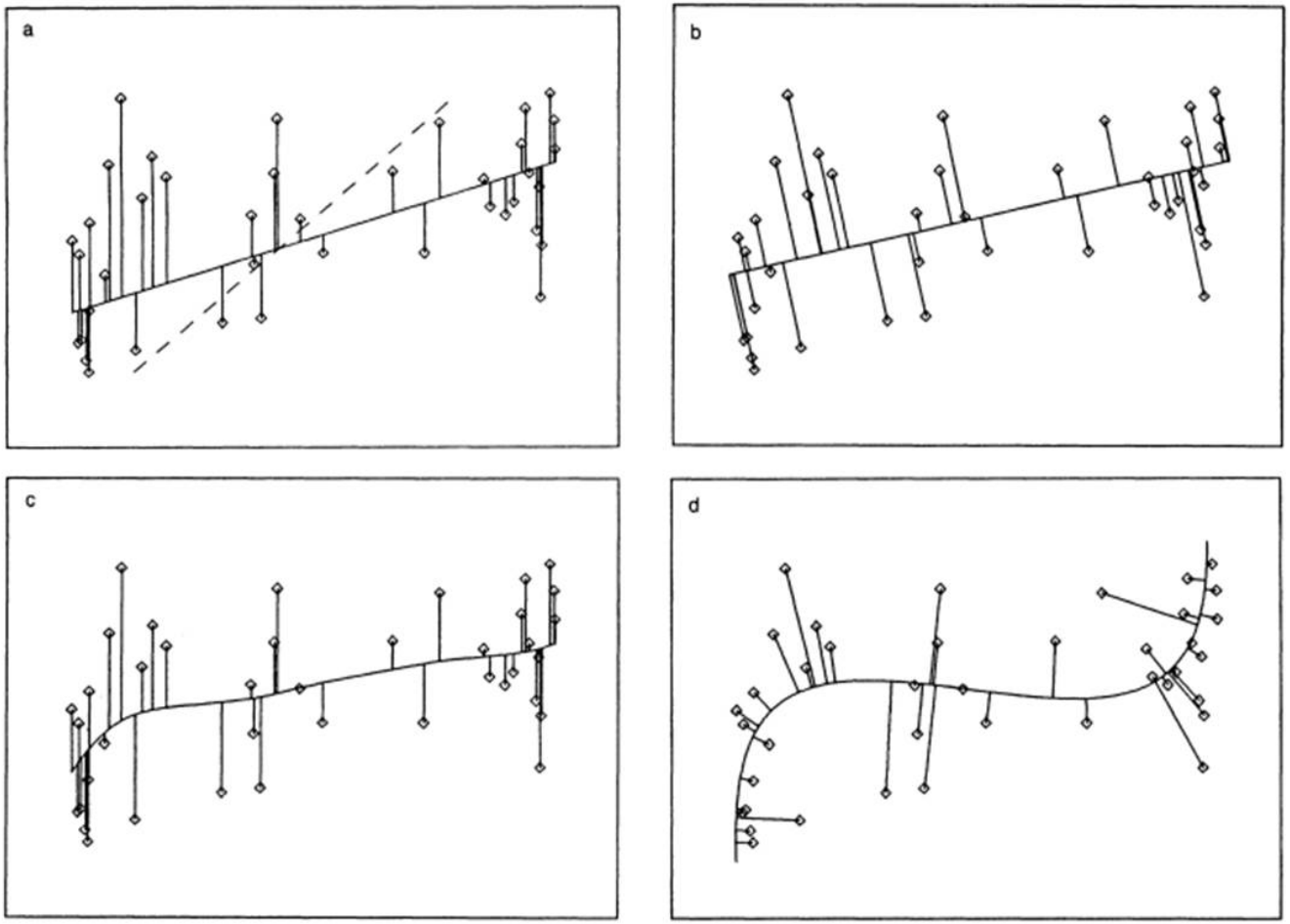
- Parallel analysis (Horn)
  http://files.eric.ed.gov/fulltext/EJ1101205.pdf
- Bootstrap method (Eigenvalues)

```
library(nFactors)
ev <- eigen(cor(USArrests)) # get eigenvalues
ap <- parallel(subject=nrow(USArrests), var=ncol(USArrests),
  rep=100,cent=.05)
nS <- nScree(x=ev$values, aparallel=ap$eigen$qevpea)
plotnScree(nS)
```

PCA number of significant components



Principal curves



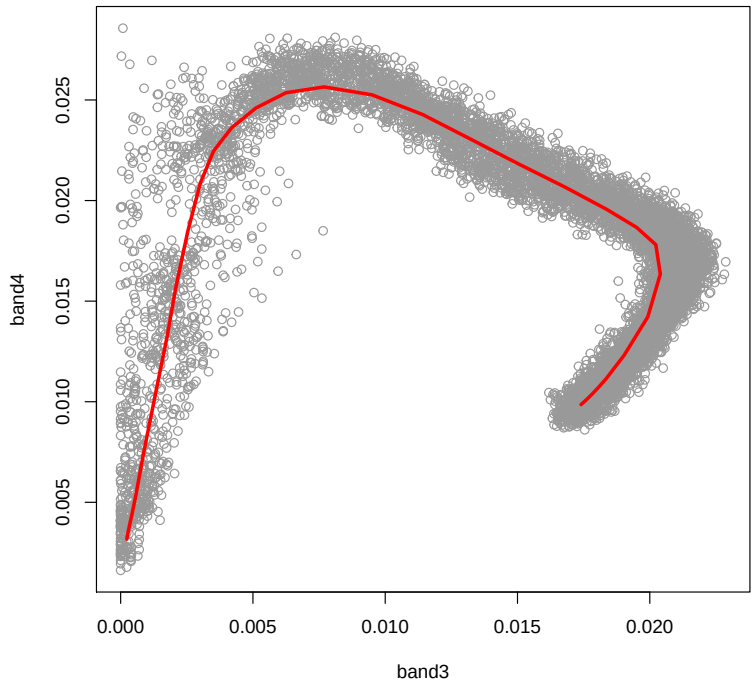
Principal curves

Example: Spectral decomposition of stellar objects, generated in the framework of the Gaia project.

```
require(LPCM)
data(gaia)
dim(gaia)

## [1] 8286   20

names(gaia)

##  [1] "ID"          "metallicity" "gravity"     "temperature" "band1
##  [6] "band2"       "band3"       "band4"       "band5"       "band6
## [11] "band7"       "band8"       "band9"       "band10"      "band1
## [16] "band12"      "band13"      "band14"      "band15"      "band1
```

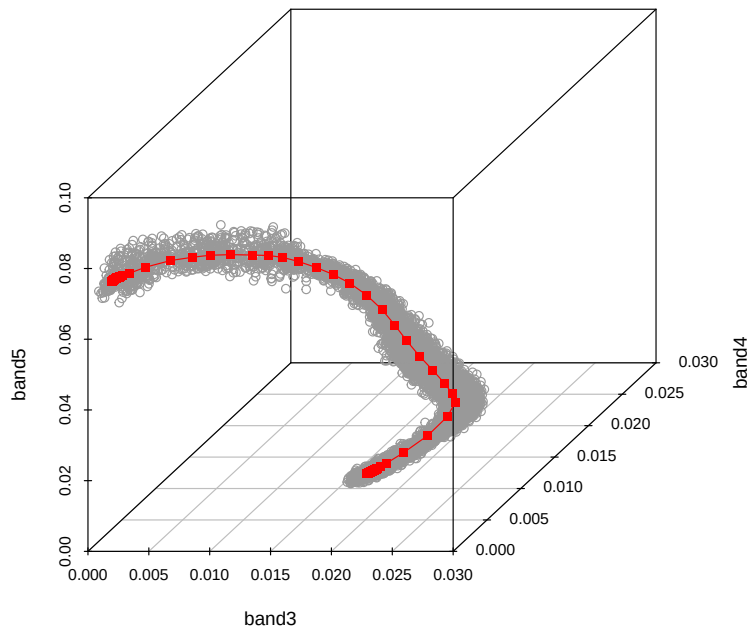
Principal curves

```
lpc1 <-  lpc(gaia[,7:8])
plot(lpc1, curvecol="red", lwd=3)
```



Principal curves

```
require(scatterplot3d)
lpc2 <- lpc(gaia[,7:9])
plot(lpc2, curvecol=2, type=c("curve","mass"))
```



Improvements of PCA

- There are other thechniques such as: Principal co-ordinate analysis (PCoA) or Multidimensional scaling (MDS), correspondence analysis (CA) and nonsymetrical correspondence analysis (NSCA).
- They can be applied to other types of data (e.g. non-continuous). PCoA can also be applied to binary or count data.
- PCA is designed to analyze multi-normal distributed data. If data are skweed, contain extreme outliers, or display nonlinear trends, other methods such as PCoA or MDS should be used instead.
- Nonnegative matrix factorization (NMF) and Independent Component Analysis (ICA) are applied when orthogonality or independence acrross components are not hold.

Improvements of PCA

- There are other thechniques such as: Principal co-ordinate analysis (PCoA) or Multidimensional scaling (MDS), correspondence analysis (CA) and nonsymetrical correspondence analysis (NSCA).
- They can be applied to other types of data (e.g. non-continuous). PCoA can also be applied to binary or count data.
- PCA is designed to analyze multi-normal distributed data. If data are skweed, contain extreme outliers, or display nonlinear trends, other methods such as PCoA or MDS should be used instead.
- Nonnegative matrix factorization (NMF) and Independent Component Analysis (ICA) are applied when orthogonality or independence acrross components are not hold.

Improvements of PCA

- There are other thechniques such as: Principal co-ordinate analysis (PCoA) or Multidimensional scaling (MDS), correspondence analysis (CA) and nonsymetrical correspondence analysis (NSCA).
- They can be applied to other types of data (e.g. non-continuous). PCoA can also be applied to binary or count data.
- PCA is designed to analyze multi-normal distributed data. If data are skweed, contain extreme outliers, or display nonlinear trends, other methods such as PCoA or MDS should be used instead.
- Nonnegative matrix factorization (NMF) and Independent Component Analysis (ICA) are applied when orthogonality or independence acrross components are not hold.

Improvements of PCA

- There are other thechniques such as: Principal co-ordinate analysis (PCoA) or Multidimensional scaling (MDS), correspondence analysis (CA) and nonsymetrical correspondence analysis (NSCA).
- They can be applied to other types of data (e.g. non-continuous). PCoA can also be applied to binary or count data.
- PCA is designed to analyze multi-normal distributed data. If data are skweed, contain extreme outliers, or display nonlinear trends, other methods such as PCoA or MDS should be used instead.
- Nonnegative matrix factorization (NMF) and Independent Component Analysis (ICA) are applied when orthogonality or independence acrross components are not hold.

Improvements of PCA

- Solving the problem for the *i*-th component

$$\arg\max_{q^i} \text{var}(Xq^i)$$

  uses SVD decomposition and it requires and inversion step that can be problematic when $p >> n$
- Several extensions based on reguralization step or L-1 penalization (Least Absolute Shrinkage and Selection Operator, LASSO) can be applied
- Sparse, penalized and reguralized extensions of PCA and related method have been recently proposed in omic data analysis.

Multidimensional scalling

- PCA requires (multivariate) data normallity. This is an strong assumption.
- Multidimensional scalling (MDS) creates a plot displaying the relative positions of a number of variables, given only a table of the distances between them.
- There are two main methods for solving MDS
  1. Classical Multidimensional Scaling reproduces the original metric or distances.
  2. Non-Metric Multidimensional Scaling assumes that only the ranks of the distances are known. Hence, this method produces a map which tries to reproduce these ranks.
  3. MDS can be performed using `cmdscale` function.

Outline

Clustering methods

- The goal is to group samples (rows) or features (columns) or both (bi-Clustering) according to how separated are the groups.
- This 'separation' is measure using a **dissimilarity measure**
- Classes of each individual/feature is not know
- Therefore, it is known as non-supervised Clustering

Clustering methods

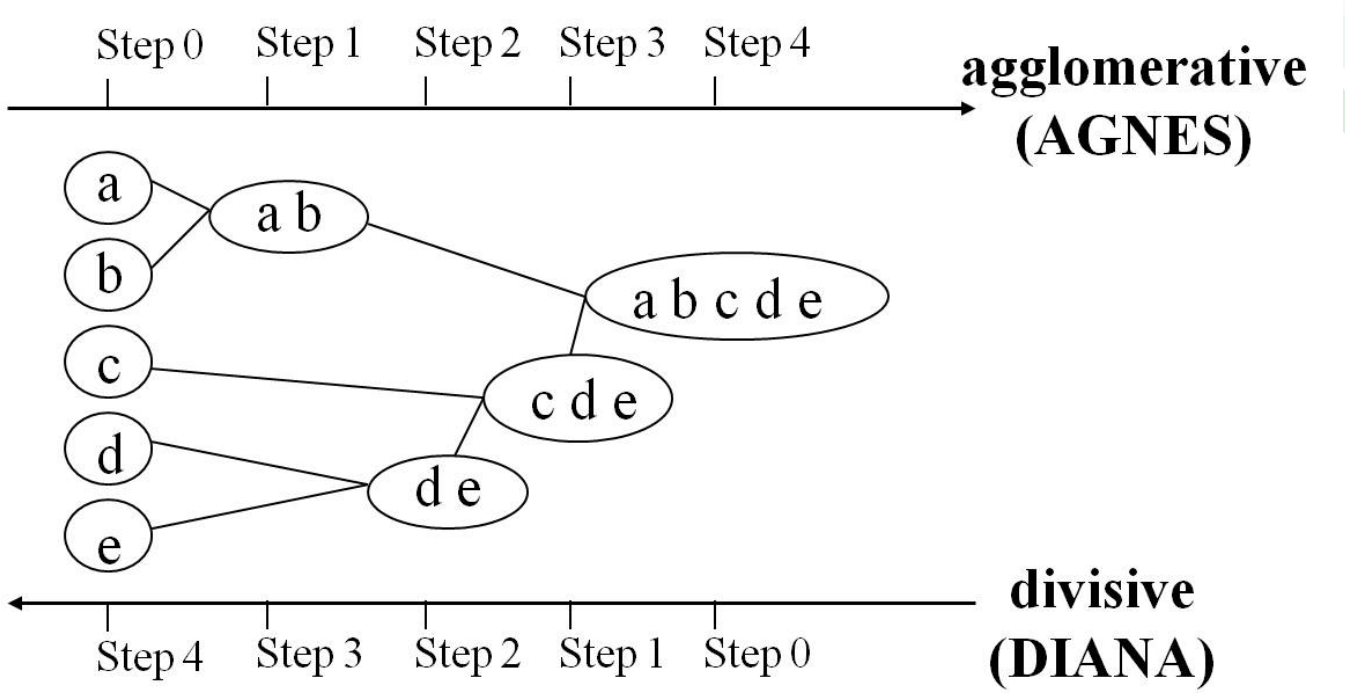
- A good clustering method will produce hihg quality clusters with
  - High intra-class similarity
  - low inter-class similarity
- The quality of a clustering result depends on both the similarity measure used by the method and its implementation
- The quality of a clustering method is also measured by its ability to dicover hidden patterns

Clustering methods

- **Hierarchical algorithms**: Create a hierarchical decomposition (agglomerative o divisive) of the set of data using some criterion
- **Partitioning algorithms**: Construct several partitions and the evaluate them by some criterion
  - k-means: each cluster is represented by the center of the cluster
  - k-medoids (or PAM): each cluster is represented by one of the objects in the cluster
- **Model-based**: A model is hypothesized for each of the clusters and the idea is to find the best fit of the data given the model

Hierarchical clustering

Use distance matrix as clustering criteria. This method does not require the number of clusters, but needs a termination condition.



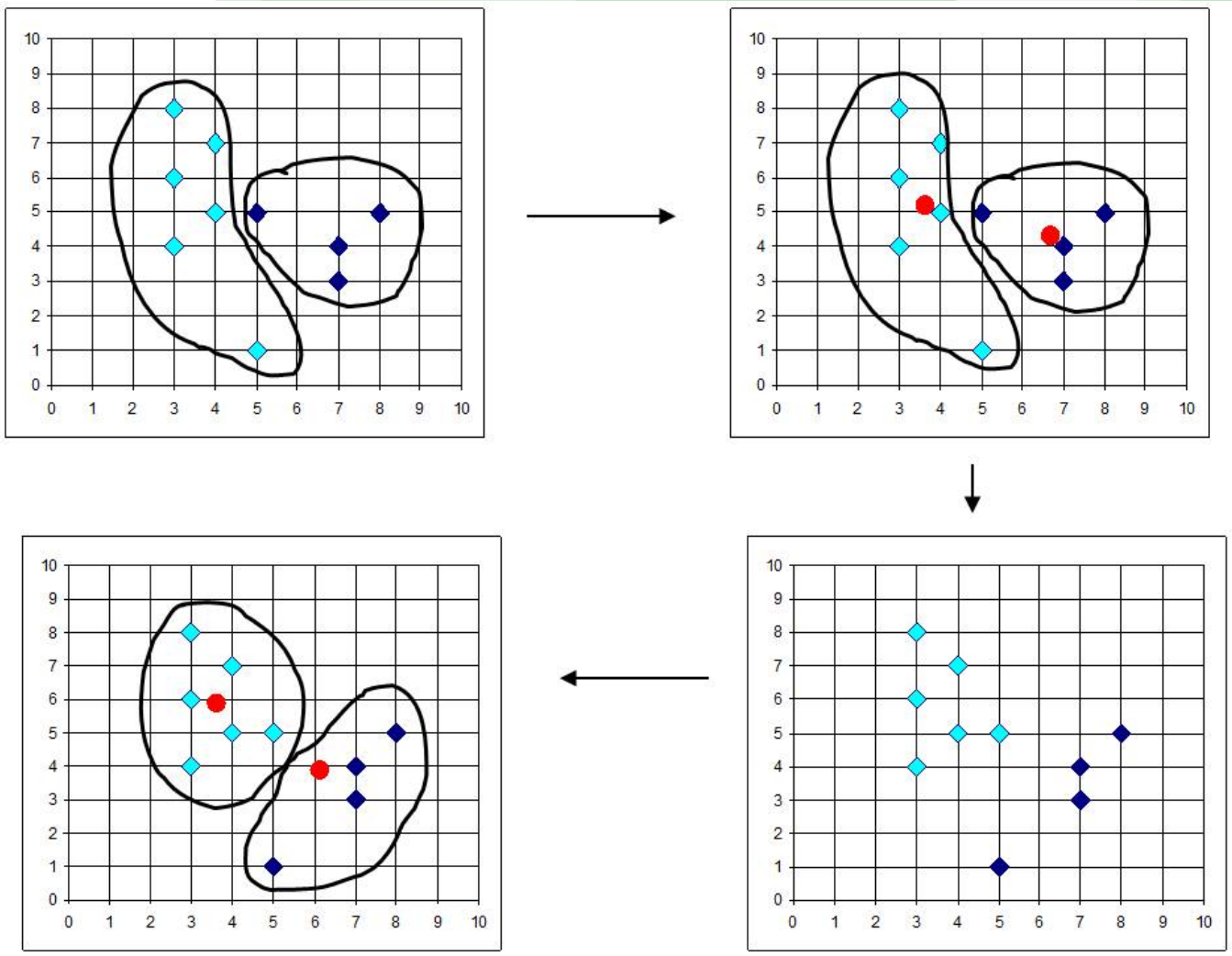
Hierarchical clustering

Distance

- Continuous variables: euclidean, manhattan, canberra, ...
- Categorical variables: binary
- Mixed variables: Gower

Partitioning methods

Construct a partition of a database into a set of k clusters



Partitioning methods

- Strengths
  - Relatively efficient O(nk)
  - Often ends at a local optimum
- Weeknesses
  - Applicable only when mean is define - what about categorical data?
  - Need to specify k, the number of clusters, in advance
  - Unable to handle noisy data and outliers
  - Non suitable to discover clusters with non-convex shapes

Partitioning methods: k-medoids

- Find representative (i.e., the most centrally located) objects, called medoids, in clusters
- PAM (Partitioning Around Medoids)
  - starts from a initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids when total distance is improved in the resulting clustering
  - works fine for small data sets
  - more robust than k-means
  - more complex: $O(k(n-k)^2)$
- CLARA: Uses multiple samples
- CLARANS: Randomized sampling

Model-based methods

$$f(x) = \sum_{g=1}^{G} \pi_G \phi(x|\mu_g, \Sigma_g)$$

where π_g is the probability that an observation belongs to group g and $\phi(x|\mu_g, \Sigma_g)$ is the density of a multivariate Gaussian.

- MCLUST
- Latent class approach

Clustering methods with R

Example: To illustrate interpretation of different Clustering methods, we'll look at a cluster analysis performed on a set of cars from 1978-1979; the data can be found at http://www.stat.berkeley.edu/classes/s133/data/cars.tab. Since the data is a tab-delimited file, we use read.delim:

```
dd <- read.delim("data/cars.tab")
head(dd)

##   Country                        Car  MPG Weight Drive_Ratio Horsepo
## 1    U.S.         Buick Estate Wagon 16.9  4.360        2.73
## 2    U.S. Ford Country Squire Wagon 15.5  4.054        2.26
## 3    U.S.         Chevy Malibu Wagon 19.2  3.605        2.56
## 4    U.S.    Chrysler LeBaron Wagon 18.5  3.940        2.45
## 5    U.S.                  Chevette 30.0  2.155        3.70
## 6   Japan              Toyota Corona 27.5  2.560        3.05
##   Displacement Cylinders
## 1          350         8
## 2          351         8
## 3          267         8
## 4          360         8
## 5           98         4
## 6          134         4
```

Data scalling

It looks like the variable are measured on different scales. This requires data standardization.We will apply 'robust' normalization (scale can also be used). There are functions like `daisy` from `cluster` package that will automatically perform standardization, but it doesn't give you complete control.

```
dd.ok <- dd[ , -c(1,2)]
medians <- apply(dd.ok, 2, median)
mads <- apply(dd.ok, 2, mad) # median absolute deviation
dd.ok <- scale(dd.ok, center=medians, scale=mads)
```

Hiherarchical methods

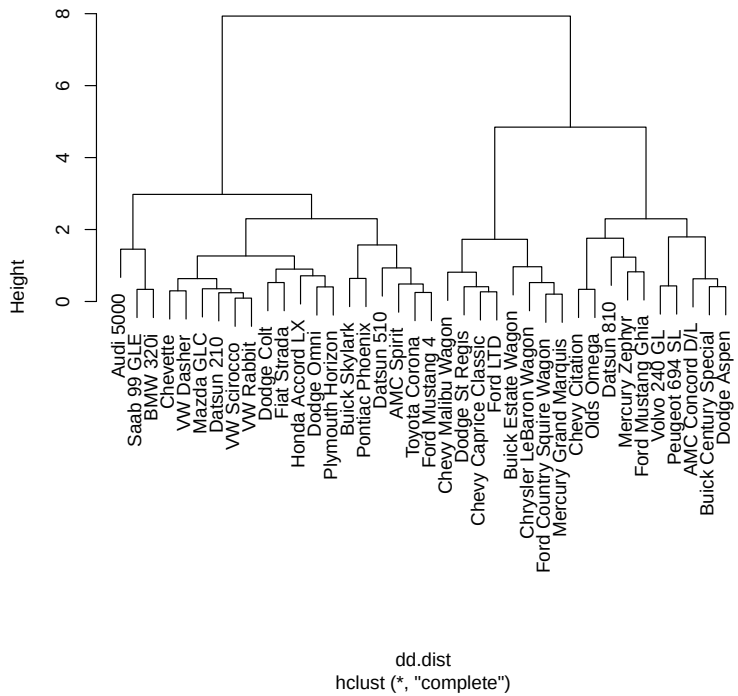
```
dd.dist <- dist(dd.ok)
dd.dist.camb <- dist(dd.ok, method="canberra")
```

```
library(cluster)
dd.dist.gower <- daisy(dd.ok, metric="gower")
```

Hiherarchical methods

Dendogram is the main graphical tool for getting insight into a cluster solution

```
dd.hclust <- hclust(dd.dist)
plot(dd.hclust, labels=dd$Car, main="")
```



Hiherarchical methods

One may be interested in knowing how many samples/features are in each group that is defined by a given height along the y-axis.

```
groups3.hclust <- as.factor(cutree(dd.hclust, 3))
table(groups3.hclust)

## groups3.hclust
##  1  2  3
##  8 20 10
```

Hiherarchical methods

To see which individuals/features are in each group ...

```
dd$Car[groups3.hclust==1]

## [1] Buick Estate Wagon        Ford Country Squire Wagon
## [3] Chevy Malibu Wagon        Chrysler LeBaron Wagon
## [5] Chevy Caprice Classic     Ford LTD
## [7] Mercury Grand Marquis     Dodge St Regis
## 38 Levels: AMC Concord D/L AMC Spirit Audi 5000 ... VW Scirocco
```

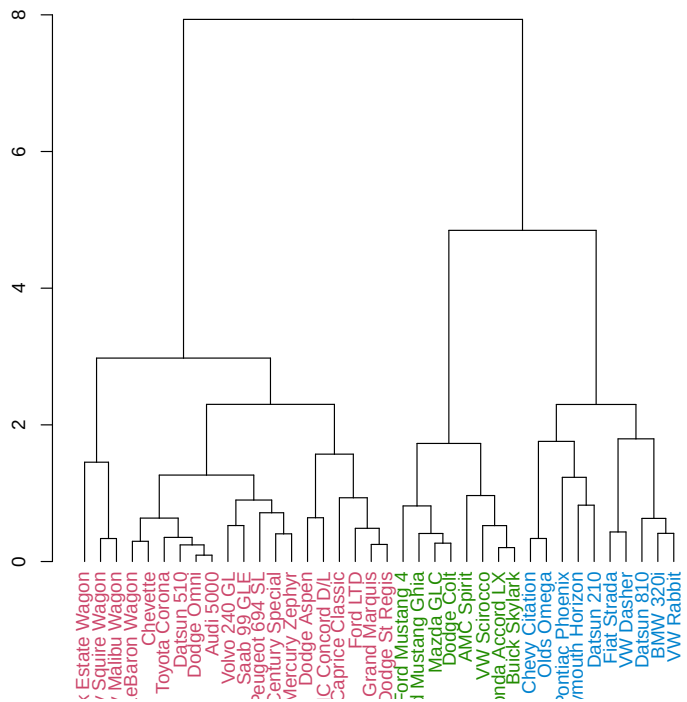
To see which individuals/features are in each group ...

```
sapply(unique(groups3.hclust), function (x) dd$Car[groups3.hclust==x])

## [[1]]
## [1] Buick Estate Wagon        Ford Country Squire Wagon
## [3] Chevy Malibu Wagon        Chrysler LeBaron Wagon
## [5] Chevy Caprice Classic     Ford LTD
## [7] Mercury Grand Marquis     Dodge St Regis
## 38 Levels: AMC Concord D/L AMC Spirit Audi 5000 ... VW Scirocco
##
## [[2]]
##  [1] Chevette          Toyota Corona    Datsun 510       Dodge Omni
##  [5] Audi 5000         Saab 99 GLE      Ford Mustang 4   Mazda GLC
##  [9] Dodge Colt        AMC Spirit       VW Scirocco      Honda Accor
## [13] Buick Skylark     Pontiac Phoenix  Plymouth Horizon Datsun 210
## [17] Fiat Strada       VW Dasher        BMW 320i         VW Rabbit
```

Hiherarchical methods

Dendrogram can be colour

```
dend <- as.dendrogram(dd.hclust)
dend2 <- dendextend::color_labels(dend, k=3)
dendextend::labels(dend2) <- dd$Car
plot(dend2)
```



Partitioning methods

```
require(cluster)
dd.pam <- pam(dd.dist, 3)
dd.kmeans <- kmeans(dd.dist, 3)
groups3.pam <- as.factor(dd.pam$clustering)
groups3.kmeans <-  as.factor(dd.kmeans$cluster)
```

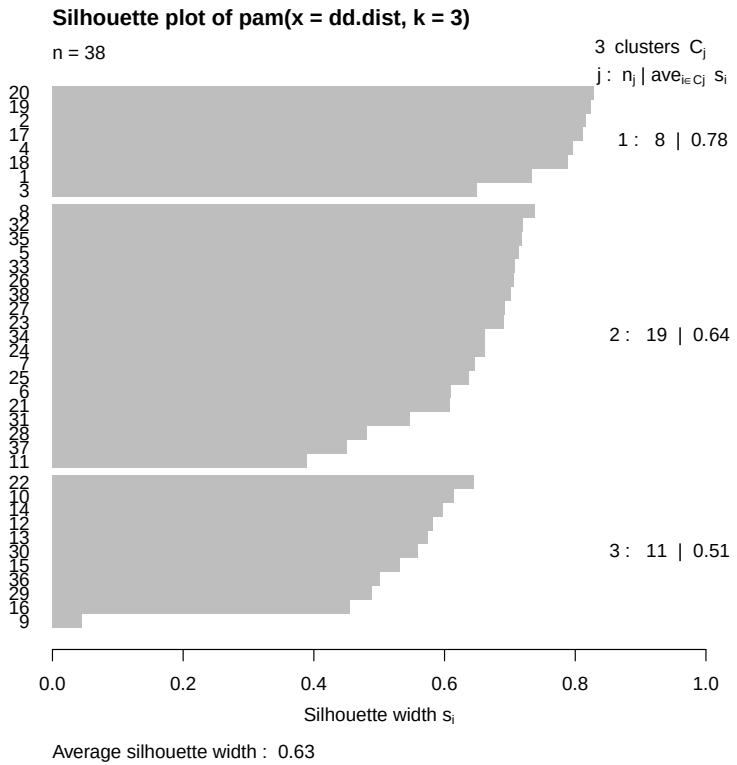
Partitioning methods

```
table(groups3.pam, groups3.hclust)

##              groups3.hclust
## groups3.pam  1  2  3
##            1  8  0  0
##            2  0 19  0
##            3  0  1 10

table(groups3.pam, groups3.kmeans)

##              groups3.kmeans
## groups3.pam  1  2  3
##            1  0  8  0
##            2 19  0  0
##            3  0  0 11
```

Partitioning methods

Silhouette plot describe how good the structure of the clusters is.

```
plot(dd.pam)
```



Partitioning methods

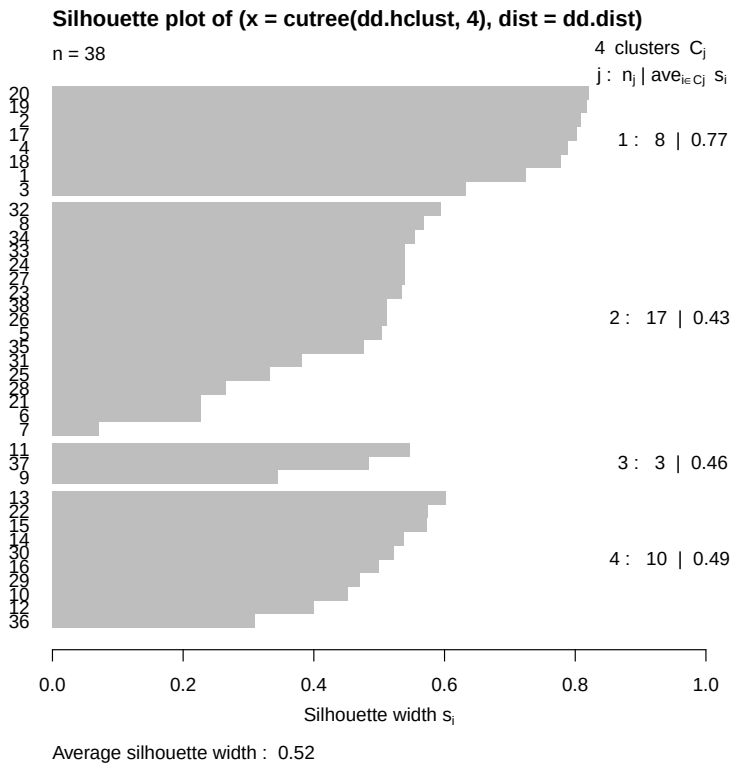
There is a criteria to interpret these values

Range of SC	Interpretation
0.71-1.00	A strong structure has been found
0.51-0.70	A reasonable structure has been found
0.26-0.50	The structure is weak and could be artificial
0-0.25	No substantial structure has been found

Partitioning methods

Silhouette plot can be created for any method

```
plot(silhouette(cutree(dd.hclust,4), dd.dist))
```



Model-based

Mclust assumes Multivariate Gaussian distribution

```
require(mclust)
dd.mclust <- Mclust(dd.ok)
summary(dd.mclust)

## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEV (ellipsoidal, equal shape) model with 6 components:
##
##  log.likelihood  n  df      BIC      ICL
##       293.5089 38 142 70.48055 70.47931
##
## Clustering table:
##  1  2  3  4  5  6
##  8  6 10  7  5  2
```

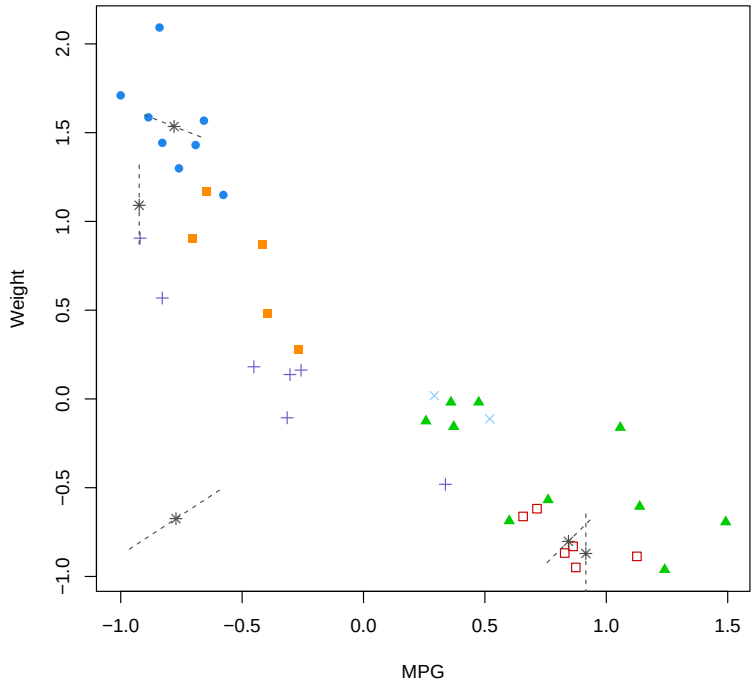
Model-based

Mclust assumes Multivariate Gaussian distribution

```
groups3.mclust <- Mclust(dd.ok, G=3)$class
table(groups3.mclust, groups3.pam)

##                groups3.pam
## groups3.mclust  1  2  3
##              1  8  0  0
##              2  0 19  0
##              3  0  0 11
```

Model-based

```
mclust2Dplot(dd.ok[,1:2], parameters=dd.mclust$parameters,
             z=dd.mclust$z, what = "classification")
```



Outline

Clustering related issues

Key issues

- Number of clusters?
- Are clusters statistically significant?
- Are clusters reproducible?
- Big datasets?
- Sparse data in genomics?

Number of clusters

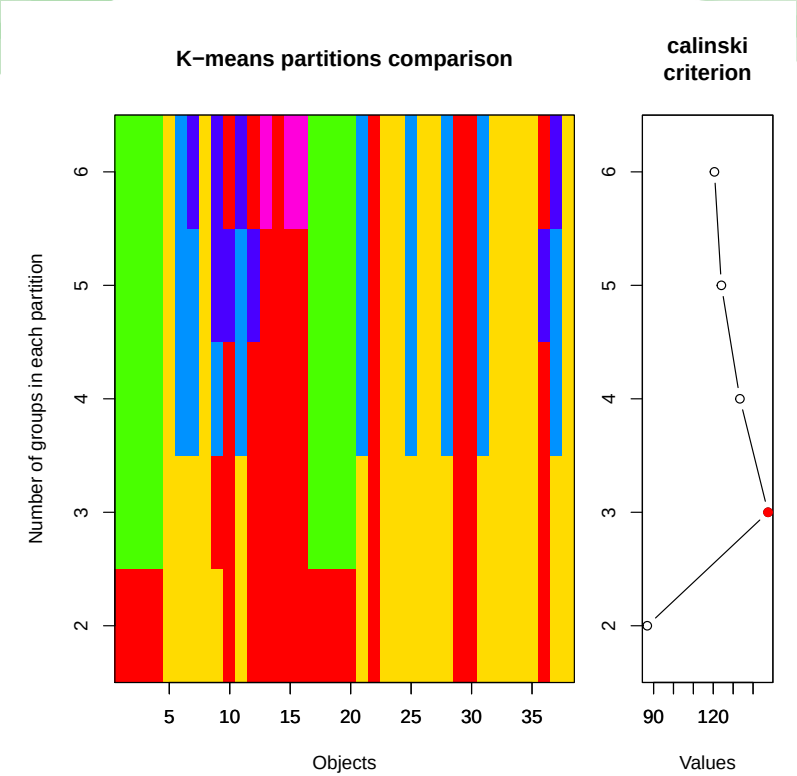
- Model-based: AIC
- Clustering methods:
  - **calinski**: (SSB/(K-1))/(SSW/(n-K)), where n is the number of data points and K is the number of clusters. SSW is the sum of squares within the clusters while SSB is the sum of squares among the clusters. This index is simply an F (ANOVA) statistic.
  - **ssi**: the *Simple Structure Index* multiplicatively combines several elements which influence the interpretability of a partitioning solution. The best partition is indicated by the highest SSI value.

Number of clusters

```
require(vegan)
k.cal <- cascadeKM(dd.ok, inf.gr=2, sup.gr=6, criterion="calinski")
k.ssi <- cascadeKM(dd.ok, inf.gr=2, sup.gr=6, criterion="ssi")
```

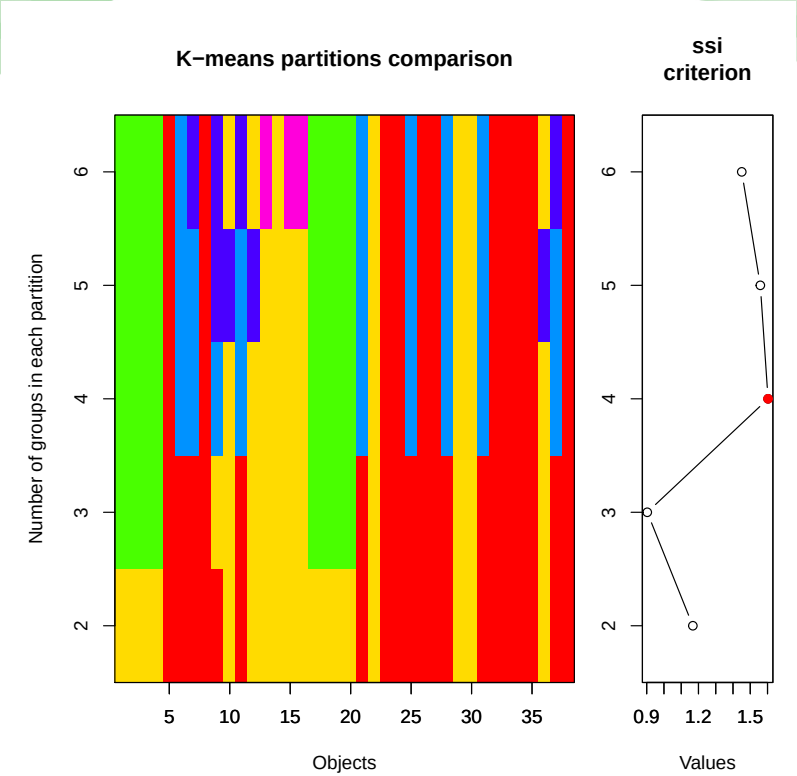
Number of clusters

```
plot(k.cal)
```



Number of clusters

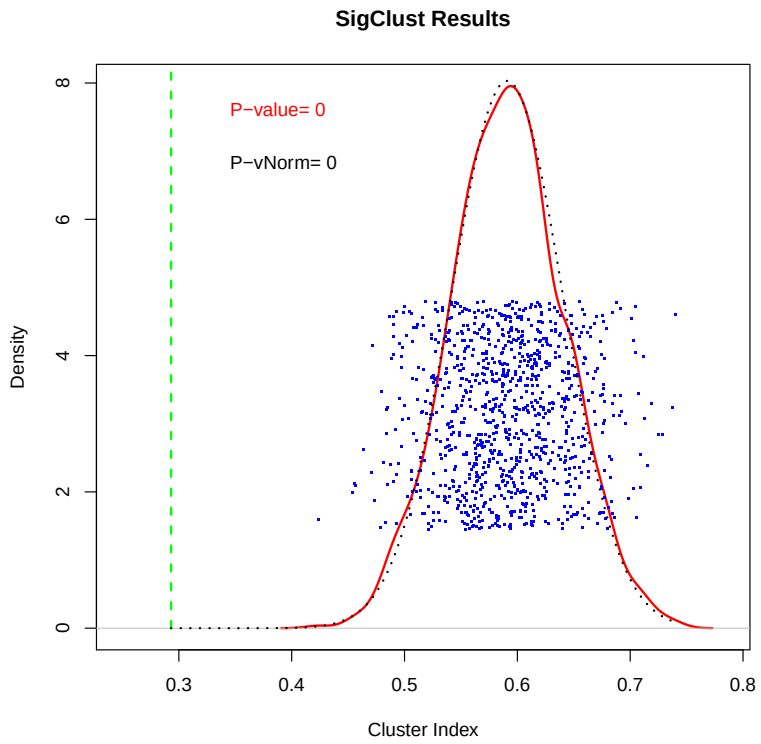
```
plot(k.ssi)
```



Testing clustering results

Test for 2 clusters vs 1

```
require(sigclust)
mod.sig <-sigclust(dd.ok, nsim=1000)
plot(mod.sig, arg="pvalue")
```



Cluster validation

```
require(clValid)
val.intern <- clValid(dd.ok, 2:6, clMethods = c("hierarchical",
 "kmeans", "diana", "pam",  "model"), validation = "internal")
optimalScores(val.intern)

##                  Score       Method Clusters
## Connectivity 2.6317460       kmeans        2
## Dunn         0.5619325 hierarchical        2
## Silhouette   0.6297805       kmeans        3
```

Big datasets

```
require(fastcluster)
hclust.fast <- hclust(dd.dist)
```