Lecture 9: Multivariate Analysis

Jing Ma, Statistics, TAMU

4 November, 2019

Recap

- Data matrices
- Principal component analysis

This lecture

- Multidimensional scaling (MDS)
- Interpretation tools
- Correspondence analysis

Multidimensional scaling

- Also called *Principal Coordinate Analyses (PCoA)*.
- Suppose we are given a matrix of dissimilarities or distances and we want to build a **useful** map of the observations.

```
require("graphics")
data(eurodist)
# look at raw distances
as.matrix(eurodist)[1:4,1:4]
```

```
##           Athens Barcelona Brussels Calais
## Athens         0      3313     2963   3175
## Barcelona   3313         0     1318   1326
## Brussels    2963      1318        0    204
## Calais      3175      1326      204      0
```

Distances across land

```
load(path.expand("data/distEuroN.RData"))
seteuro = as.matrix(distEuroN)[1:12, 1:12]
pheatmap(seteuro, cluster_rows = TRUE,
  treeheight_row = 0.0001, treeheight_col = 0.8,
  fontsize_col = 8, cellwidth = 13, cellheight = 13)
```

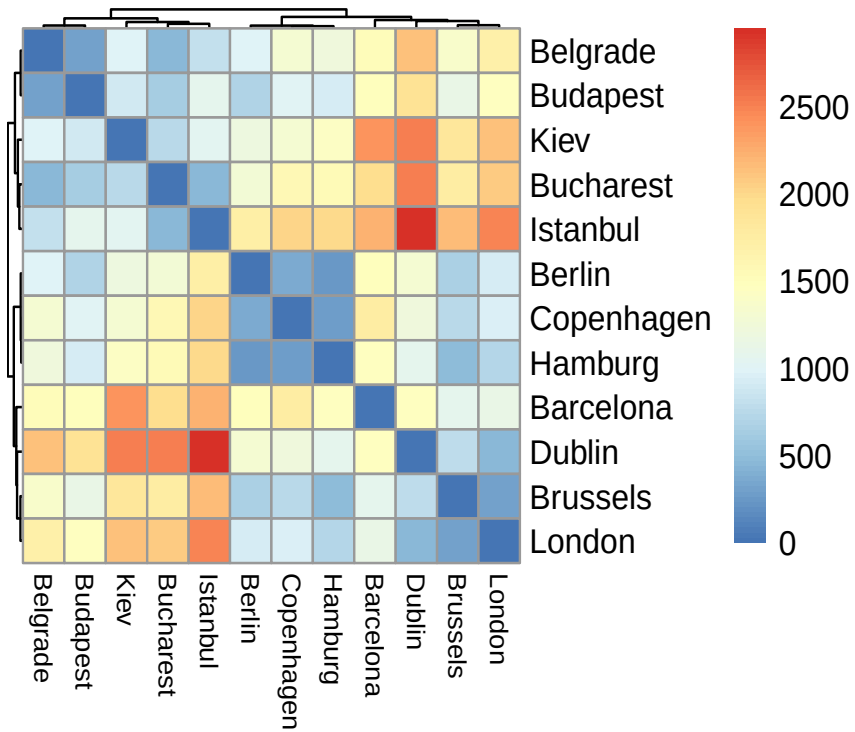


Fig. 1: A heatmap of the distances between some of the cities. The function has re-arranged the order of the cities, grouping the closest ones.

Multidimensional scaling

MDS provides a *map* of their relative locations.

```
MDSEuro = cmdscale(distEuroN, eig = TRUE)
names(MDSEuro)
```

```
## [1] "points" "eig"    "x"      "ac"     "GOF"
```

Screeplot

```
library("tibble")
plotbar = function(res, m = 9) {
  tibble(eig = res$eig[seq_len(m)], k = seq(along = eig)) %>%
  ggplot(aes(x = k, y = eig)) +
    scale_x_discrete("k", limits = seq_len(m)) + theme_minimal() +
    geom_bar(stat="identity", width=0.5, color="orange", fill="pink")
}
plotbar(MDSEuro, m = 5)
```

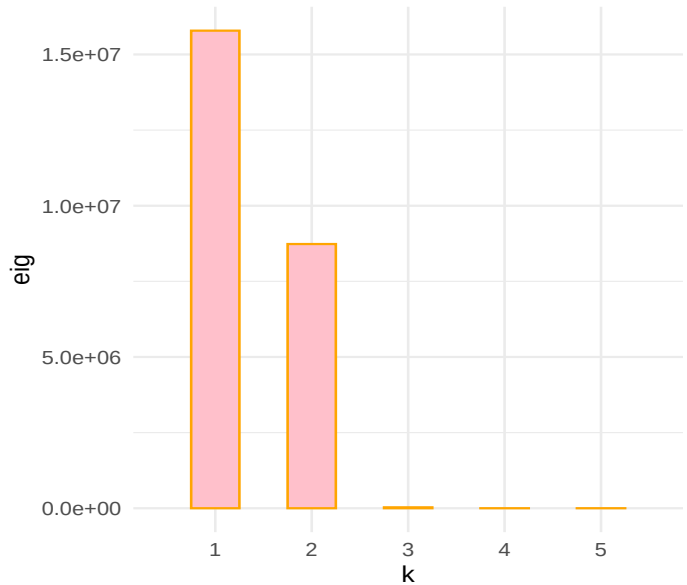
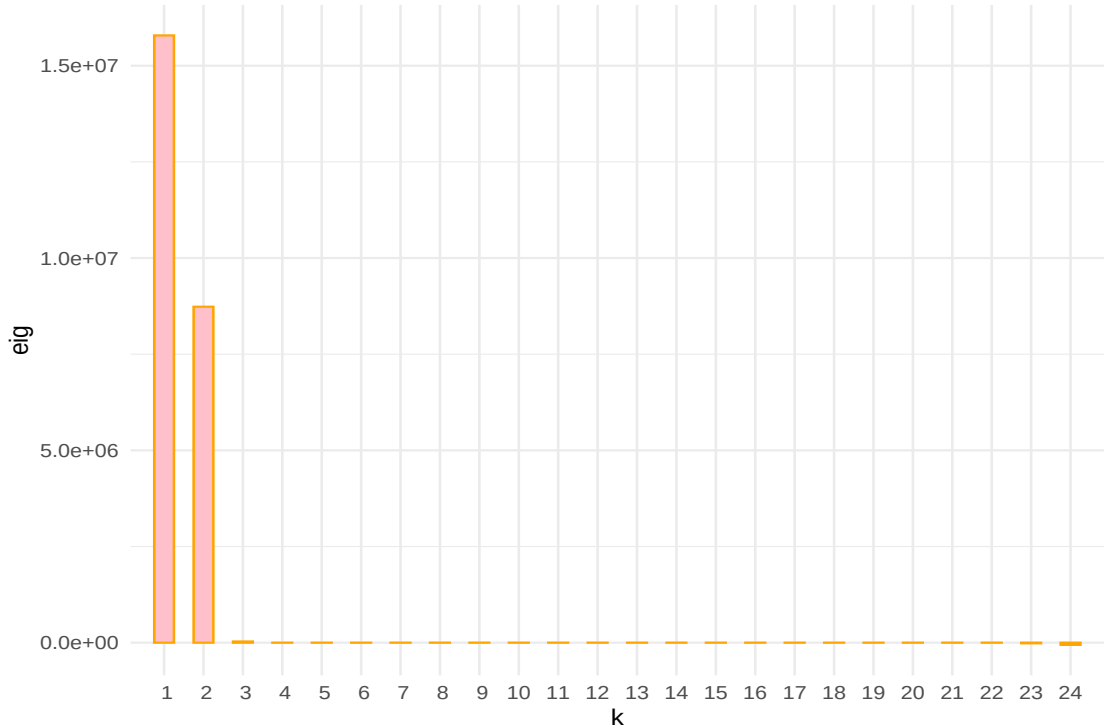


Fig. 2: Screeplot of the first 5 eigenvalues. The drop after the first two eigenvalues is very visible.

Barplot

```
plotbar(MDSEuro, m = length(MDSEuro$eig))
```



- Note that unlike in PCA, there **are** some negative eigenvalues, due to the fact that the data do not come from a Euclidean space.

Two dimensional approximation

We can project these countries onto new coordinates created from the distances.

```
MDSeur = tibble(
  PCo1 = -MDSEuro$points[, 1],
  PCo2 = -MDSEuro$points[, 2],
  labs = rownames(MDSEuro$points))

g <- ggplot(MDSeur, aes(x = PCo1, y = PCo2, label = labs)) +
  geom_point(color = "red") + xlim(-1950, 2000) + ylim(-1150, 1150) +
  coord_fixed() + geom_text(size = 4, hjust = 0.3, vjust = -0.5)
g
```

Two dimensional approximation

- Relative positions are correct.
- The orientation of the map is unconventional, e.g., Istanbul, which is in the South-East of Europe, is at the top left.

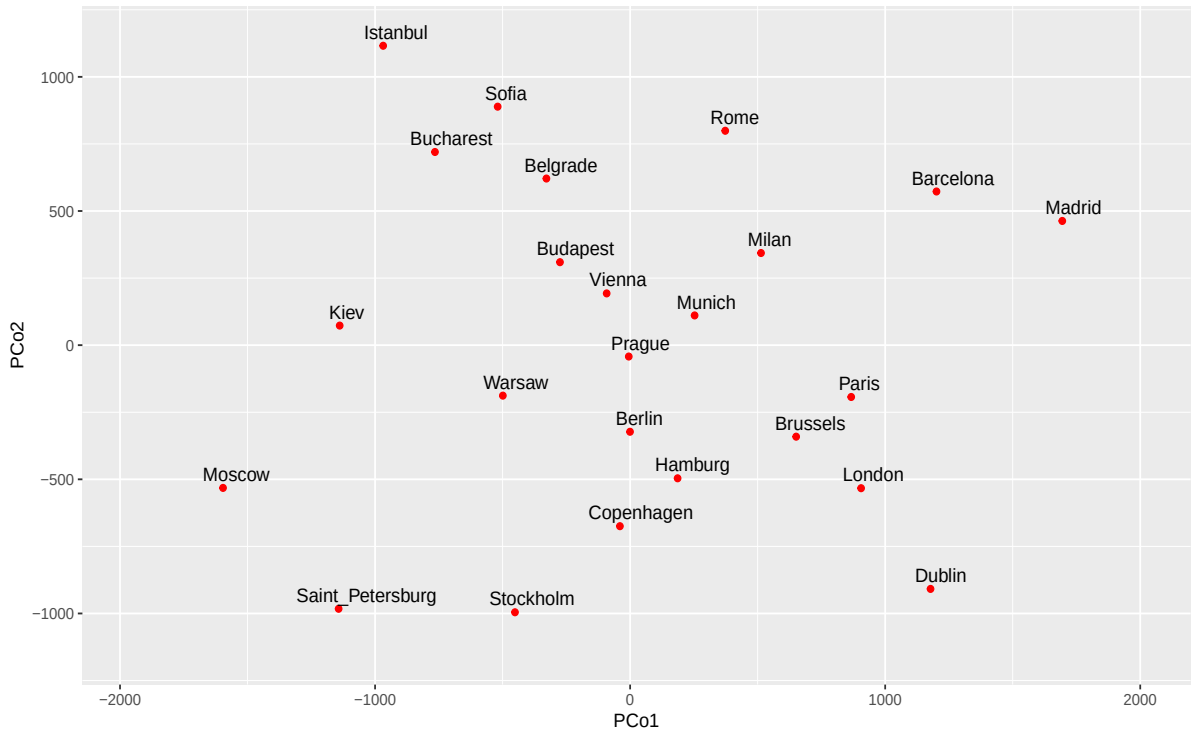


Fig. 3: MDS map of European cities based on their distances.

Two dimensional approximation

- We can reverse the signs of the principal coordinates and redraw the map.

```
g %+% mutate(MDSeur, PCo1 = -PCo1, PCo2 = -PCo2)
```

- We also read in the cities' true longitudes and latitudes and plot these for comparison.

```
Eurodf = readRDS(path.expand("data/Eurodf.rds"))
ggplot(Eurodf, aes(x = Long,y = Lat, label = rownames(Eurodf))) +
  geom_point(color = "blue") +
  geom_text(hjust = 0.5, vjust = -0.5, size=3)
```

Two dimensional approximation

Which cities seem to have the worst representation on the PCoA map?

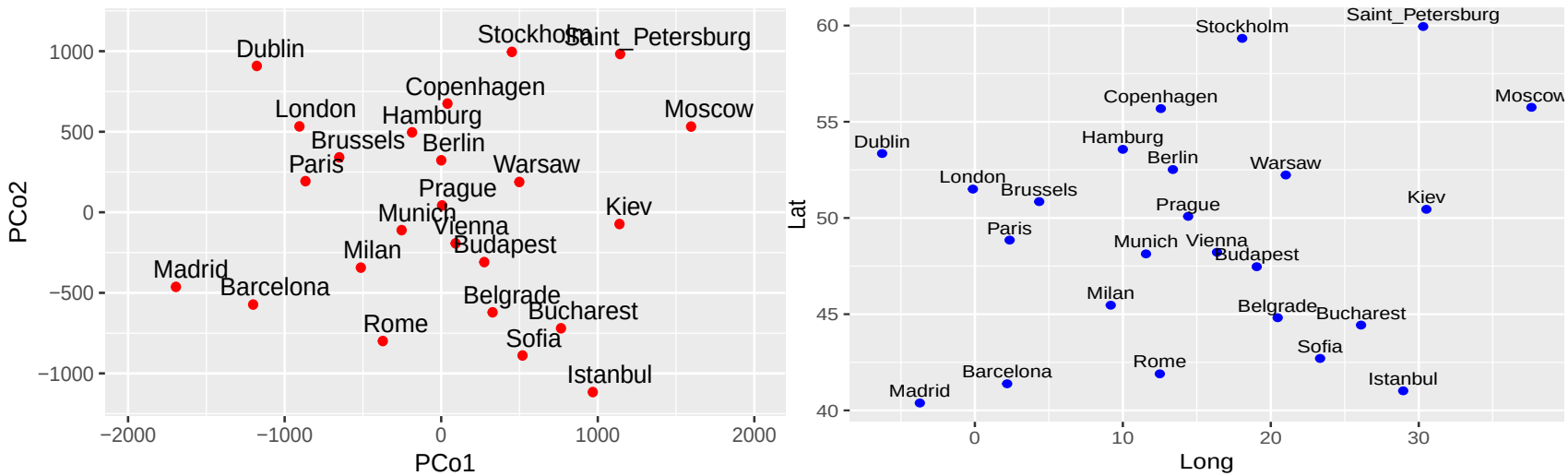


Fig. 4: Two-dimensional approximation (left) vs. True latitudes and longitudes (right).

A psychological experiment: color confusion

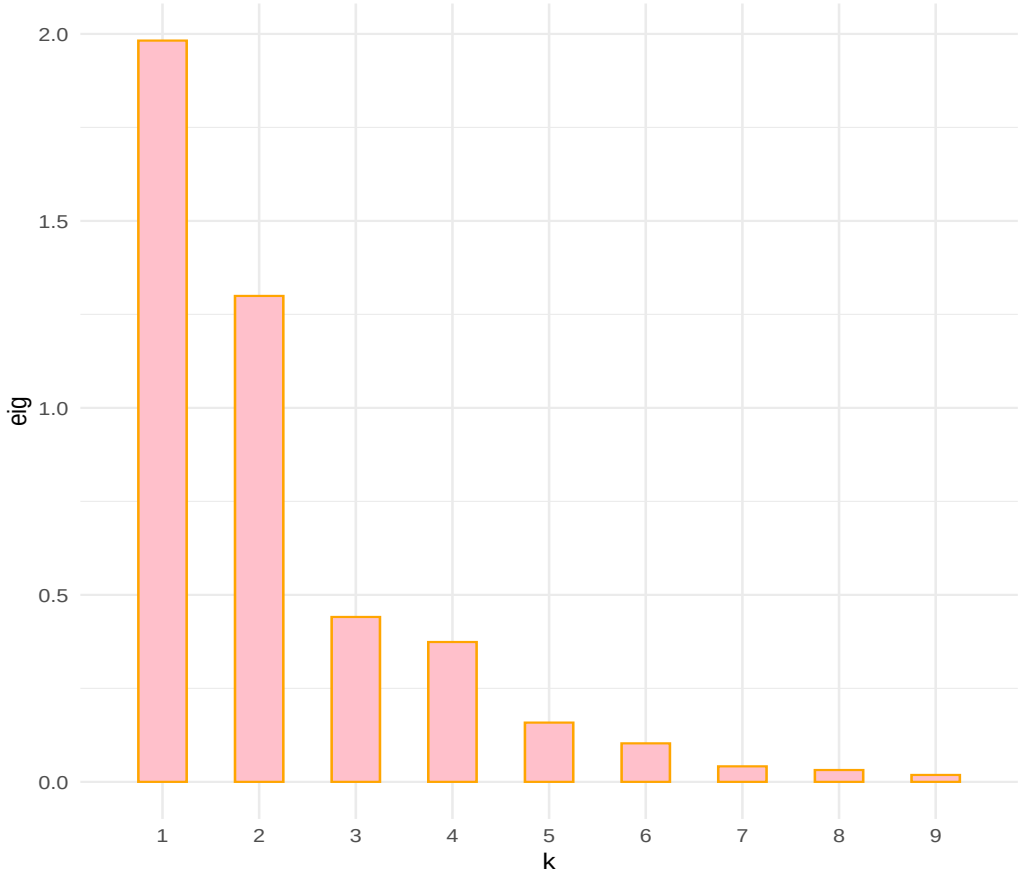
```
ekm=read.table("data/eckman.txt",header=TRUE)
rownames(ekm) = colnames(ekm)
disekm = 1 - ekm - diag(1, ncol(ekm))
disekm[1:5, 1:5]
```

```
##      w434 w445 w465 w472 w490
## w434 0.00 0.14 0.58 0.58 0.82
## w445 0.14 0.00 0.50 0.56 0.78
## w465 0.58 0.50 0.00 0.19 0.53
## w472 0.58 0.56 0.19 0.00 0.46
## w490 0.82 0.78 0.53 0.46 0.00
```

```
disekm = as.dist(disekm)
```

A psychological experiment: color confusion

```
mdsekm = cmdscale(disekm, eig = TRUE)
plotbar(mdsekm)
```

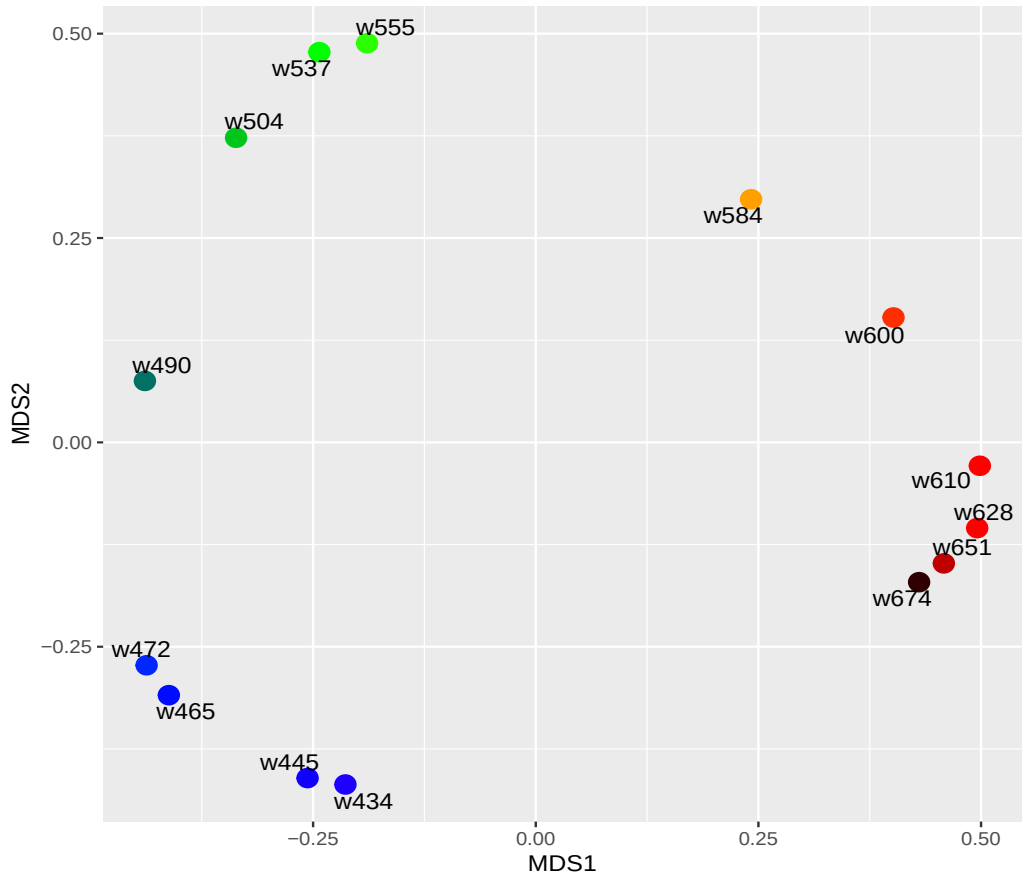


A psychological experiment: color confusion

```
dfekm = as_tibble(mdsekm$points[,1:2])%>%
  setNames(paste0("MDS", 1:2)) %>%
  mutate(
    name = rownames(ekm),
    rgb = photobiology::w_length2rgb(
          as.numeric(sub("w", "", name))))
```

A psychological experiment: color confusion

```
library("ggrepel")
ggplot(dfekm, aes(x = MDS1, y = MDS2)) +
  geom_point(col = dfekm$rgb, size = 4) +
  geom_text_repel(aes(label = name)) + coord_fixed()
```



How does MDS work?

MDS finds points in a lower-dim Euclidean space that have approximately the same distances as those in the full space.

From Data X to Distances D

- Given a Euclidean distance D between the observation (rows), i.e.
$$d_{i,j} = \sqrt{(x_{i1} - x_{j1})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$
- Call $B = XX'$. If $D \bullet D$ is the matrix of squared distances between rows of X in the euclidean coordinates, we can show that
$$-\frac{1}{2} HD \bullet DH = B,$$
where
$$H = I - \frac{1}{n} 11'$$
is the centering matrix.

Proof by computation

```
load("data/citiesX.RData")
X=Xcoord
DdotD=as.matrix(dist(X)^2)
```

Let's check the centering property of the matrix H:

```
X=Xcoord
H=diag(rep(1,20))-(1/20)*matrix(1,20,20)
Xc=sweep(X,2,apply(X,2,mean))
Xc[1:2,]
HX=H%*%X
HX[1:2,]
apply(HX,2,mean)
```

Relative distances do not depend on the point of origin of the data:

```
B1=-0.5*H%*%DdotD%*%H
B2=HX%*%t(HX)
max(abs(B1-B2))
```

Backward from Distances D to Data X

Given an $n \times n$ matrix of distances D, one can solve for points achieving these distances by:

1. Double centering the interpoint distance squared matrix: $B = -\frac{1}{2} HD \bullet DH$.
2. Diagonalizing B: $B = U \Lambda U^T$.
3. Extracting $\tilde{X}$: $\tilde{X} = U \Lambda^{1/2}$.

Classical MDS algorithm

- We can go backwards from a matrix $D \bullet D$ to X by taking the eigendecomposition of B.
- We can also choose how many coordinates, or columns we want for the X matrix, in much the same way that PCA provides the best rank r approximation for data.
  - We actually have the choice between using the singular value decomposition of HX or the eigendecomposition of $HX(HX)^t$.
- The algorithm gives us the coordinates of points that have approximately the same distances as those provided by D matrix.

Robustness

Robust methods are not too influenced by a few outliers.

- For example, the median does not change even if we change 20% its points by perturbing them, one has to change 50% an arbitrary amount. We say its **breakdown point** (0.5) is very high and that is a robust estimate of *location*.
- On the other hand, to change the mean value by arbitrary amount, it is enough to change one observation, so for n observations the breakdown point is $1/n$, which can be quite small.

Robust MDS

- MDS minimizes the difference between the squared distances as given by $D \bullet D$ and the squared distances between the points with their new coordinates.
- We need procedures that are less dependent on the actual values of the distances but still take into account the distances' relative rankings.
- Methods based on ranks are much more robust than those based on the actual values in general, and many nonparametric tests are based on a reduction of data to their ranks.

Non metric Multidimensional Scaling

```
res=metaMDS(as.dist(1-ekm))
col14=rainbow(14)
NMDS = data.frame(PCo1 = res$points[,1], PCo2 = res$points[,2])

ggplot(data = NMDS, aes(PCo1, PCo2)) +
  geom_point(size=4,color = col14)
```

NMDS

Robust ordination (NMDS for short) attempts to embed the points in a new space such that the *order* of the reconstructed distances in the new map is the same as the ordering of the original distance matrix.

- It looks for a transformation f of the given dissimilarities in d and a set of coordinates in a low dimensional space (the map) such that the distance in this new map is $\tilde{d}$ and $f(d) \approx \tilde{d}$.
- The quality of the approximation can be measured by

$$STRESS^2 = \frac{\sum\{f(d) - \tilde{d}\}^2}{\sum d^2}.$$

NMDS

- NMDS is not sequential: we specify the underlying dimensionality at the outset.
- The optimization is run to maximize the reconstruction of the distances according to that number.
- There is no notion of percentage of variation explained by individual axes as provided in PCA. However, we can make a similar screeplot by running the program for all the successive values of $k = 1, 2, \ldots$ and looking at how well the `STRESS` drops.

NMDS

Several replicates at each dimension are run to evaluate the stability of the STRESS.

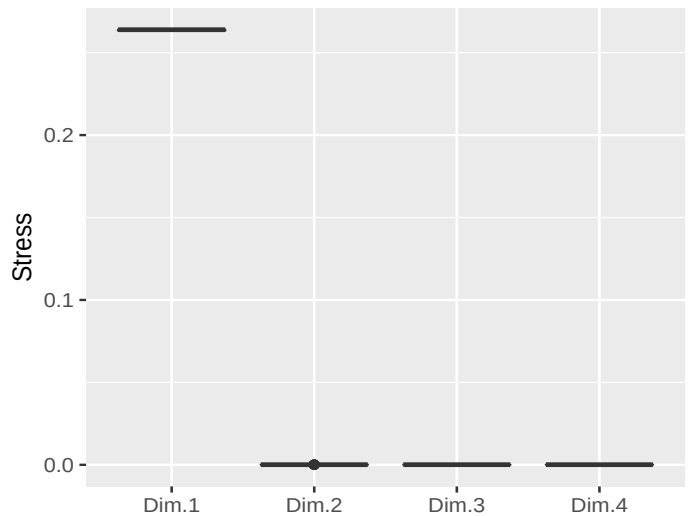
```
nmds.stress<-function(x,sim=20,kmax=4) {
  stressp=matrix(0,nrow=sim,ncol=kmax)
  for  (i in 1:kmax)
  stressp[,i]=replicate(sim,metaMDS(x, autotransform=F, k=i)$stress)
  return(stressp)
}
bootscree=nmds.stress(distEuroN,sim=100)

colnames(bootscree)=paste("Dim",1:4,sep=" ")
dfm=melt(data.frame(bootscree),variable_name="stress")
```

NMDS

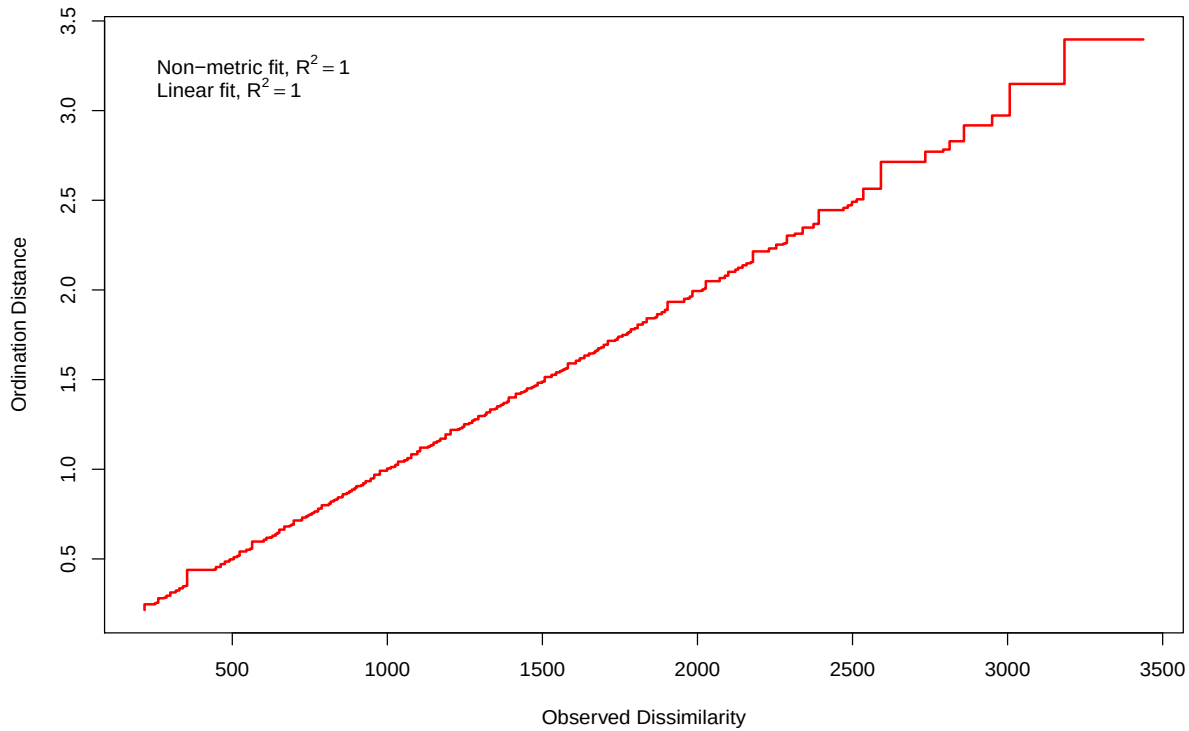
We see that the STRESS drops dramatically after the first axis, thus indicating that a one dimensional solution may be appropriate here. But it's easier to use two dimensions because it allows us to keep the labels from overlapping.

```
ggplot(dfm,aes(x = stress,y = value)) +
  geom_boxplot() + xlab("") + ylab("Stress")
```



NMDS: the Shepard plot

```
### Compare the distances from the approximations.
out=monoMDS(distEuroN,k=2,autotransform=FALSE)
stressplot(out,distEuroN,pch="")
```



NMDS vs MDS

```
set.seed(499245)

## NMDS
out2=monoMDS(distEuroN,k=2,autotransform=FALSE)
###We flip the coordinate signs.
out2$points[,1]=-(out2$points[,1])
out2$points[,2]=-(out2$points[,2])
NMDSout = data.frame(NMDS1 = out2$points[,1], NMDS2 = out2$points[,2],
        labs=rownames(out2$points))
gn <- ggplot(data = NMDSout, aes(NMDS1, NMDS2,label=labs) ) +
               geom_text_repel(aes(NMDS1, NMDS2, label = labs)) +
               xlim(-2, 1.8)+ylim(-1.5,1.5)+theme_bw()

## MDS
MDSeur$PCo1<- (-MDSeur$PCo1)
MDSeur$PCo2<- (-MDSeur$PCo2)
g <- ggplot(data = MDSeur, aes(PCo1,PCo2,label=labs) ) +
         geom_text_repel(aes(PCo1, PCo2, label = labs)) +
         xlim(-2200, 2300)+ylim(-1500,1500)+theme_bw()
```

NMDS vs MDS

```
library(cowplot)
plot_grid(gn, g, labels = c("NMDS","MDS"))
```

Different distances give different MDS projections

- Example of all the choices

Contiguous or supplementary information

- Many programs and workflows in biological sequence analysis or assays separate the environmental and contextual information they call **metadata** from the assays or sequence read numbers.
- We discourage this practice as the exact connections between the samples and covariates are important.
- Extra information about sample batches, dates of measurement, different protocols are often *misnamed* metadata. This information is actually *real* data that needs to be integrated into the analyses.
- Best practices in data analyses are sequential and that it is better to analyse data as they are collected to adjust for severe problems in the experimental design *as they occur*.

Known batches in data

We show an example on bacterial abundance data from *Phylochip* microarrays.

- The experiment was designed to detect differences between a group of healthy rats and a group who had Irritable Bowel Disease.
- This example shows a case where the nuisance batch effects become apparent in the analysis of experimental data.

Known batches in data

- When data collection started on this project, days 1 and 2 were delivered.
  - This showed a definite *day* effect.
- When investigating the source of this effect, we found that both the protocol and the array were different in days 1 and 2. This leads to uncertainty in the source of variation, we call this **confounding** of effects.

Phylochip data

```
IBDchip = readRDS("data/vsn28Exprd.rds")
library("ade4")
library("factoextra")
library("sva")

class(IBDchip)
```

```
## [1] "matrix"
```

```
dim(IBDchip)
```

```
## [1] 8635   28
```

```
# tail(IBDchip[,1:3])
summary(IBDchip[nrow(IBDchip),])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   1.857   2.000   3.000
```

Phylochip data

The data are normalized abundance measurements of 8634 taxa measured on 28 samples.

We use a rank-threshold transformation, giving the top 3000 most abundant tax scores from 3000 to 1, and letting the 5634 least abundant all have a score of 1.

We separate out the assay data from the day variable which should be considered a factor:

```
assayIBD = IBDchip[-nrow(IBDchip), ]
day = factor(IBDchip[nrow(IBDchip), ])
```

Screeplot

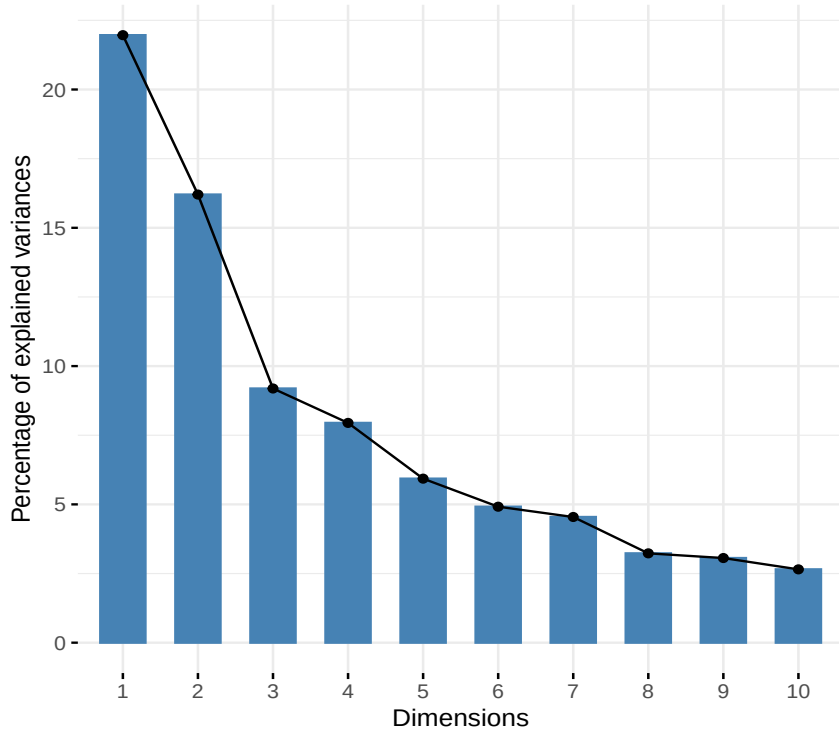


Fig. 5: The screeplot shows us that the phenomenon can be usefully represented in two dimensions.

Robust PCA

- Instead of using the continuous, normalized data, we use a robust analysis replacing the values by their ranks.
- Low abundances at noise level occur for species that are not really present, of which there are more than half. A large jump in rank for these observations would be meaningless and could easily occur without any meaningful effect. Thus we create a large number of ties at zero.
- The lower values are considered ties encoded as a threshold chosen to reflect the number of expected taxa thought to be present.

Robust PCA

```
rankthreshPCA = function(x, threshold = 3000) {
  ranksM = apply(x, 2, rank)
  ranksM[ranksM < threshold] = threshold
  ranksM = threshold - ranksM
  dudi.pca(t(ranksM), scannf = FALSE, nf = 2)
}
pcaDay12 = rankthreshPCA(assayIBD[,day!=3])
day12 = day[day!=3]
rtPCA1 <- fviz(pcaDay12, element="ind", axes=c(1,2), geom=c("point","text"),
               habillage = day12, repel = TRUE, palette = "Dark2",
               addEllipses = TRUE, ellipse.type = "convex") + ggtitle("") +
  coord_fixed()
rtPCA1
```

Robust PCA

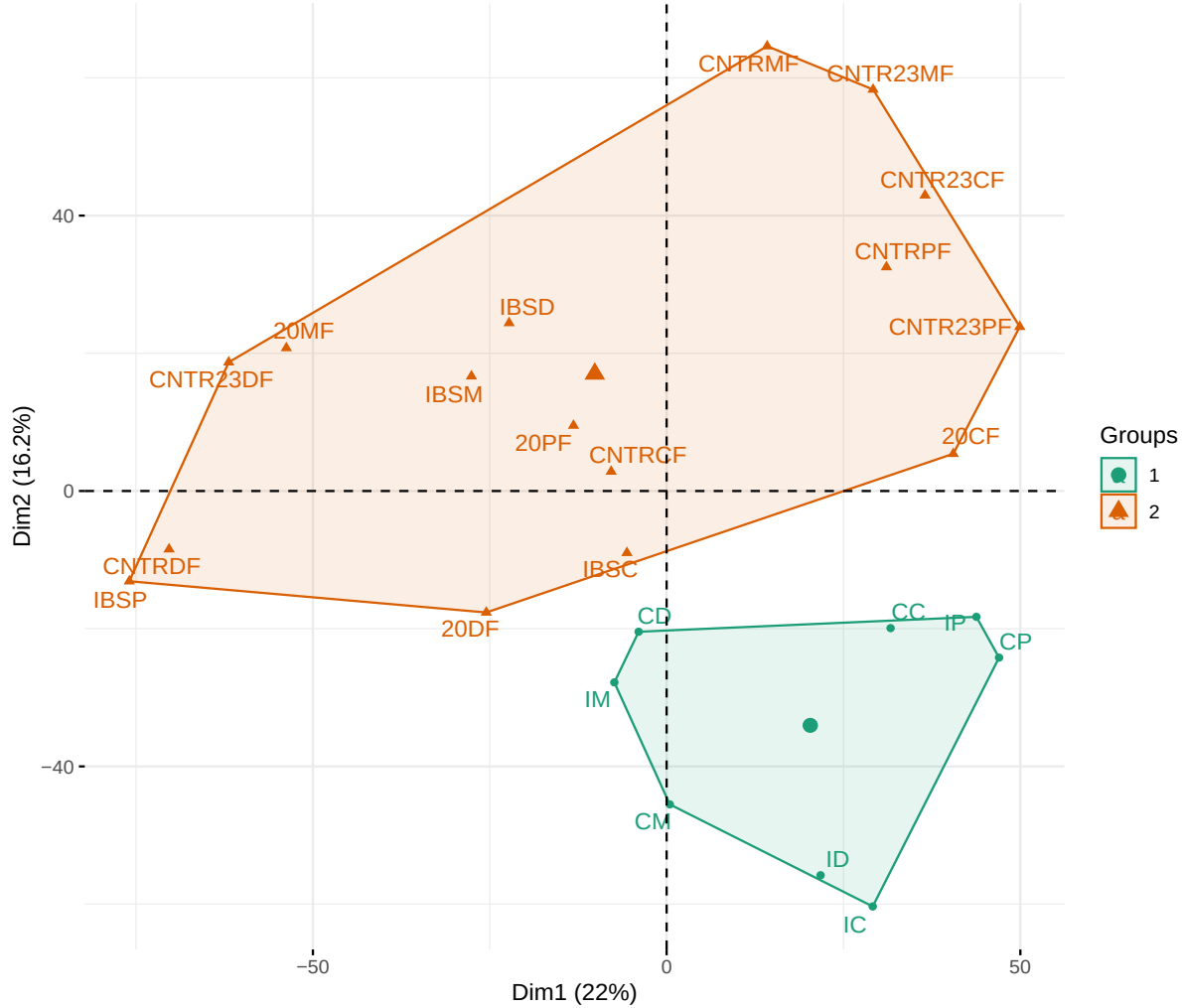


Fig. 6: We have used colors to identify the different days and have kept the sample labels as well. We have also added convex hulls for each day. The group mean is identified as the point with the larger symbol (circle, triangle or square).

Phylochip data

There were two different protocols used (protocol 1 on day 1, protocol 2 on day 2) *and* unfortunately two different provenances for the arrays used on those two days (array 1 on day 1, array 2 on day 2).

A third set of data of four samples had to be collected to deconvolve the confounding effect. Array 2 was used with protocol 1 on Day 3.

```
pcaDay123 = rankthreshPCA(assayIBD)
rtPCA2 <- fviz(pcaDay123, element="ind", axes=c(1,2), geom=c("point","text"),
               habillage = day, repel=TRUE, palette = "Dark2",
               addEllipses = TRUE, ellipse.type = "convex") + ggtitle("") +
  coord_fixed()
rtPCA2
```

Phylochip data

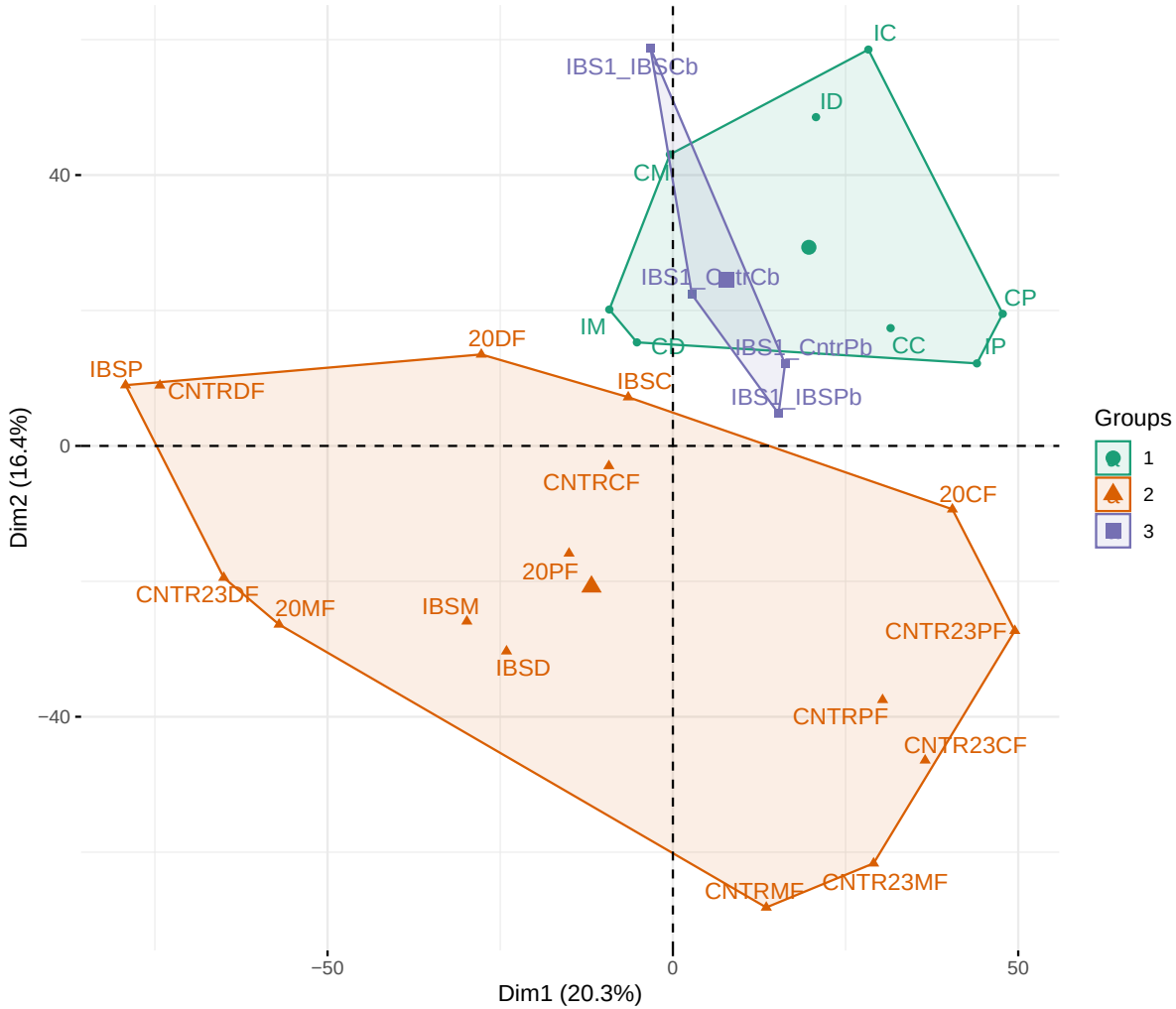


Fig. 7: Group 3 overlaps heavily with group 1 indicating that it was the protocol change on Day 2 which created the variability.

Phylochip data

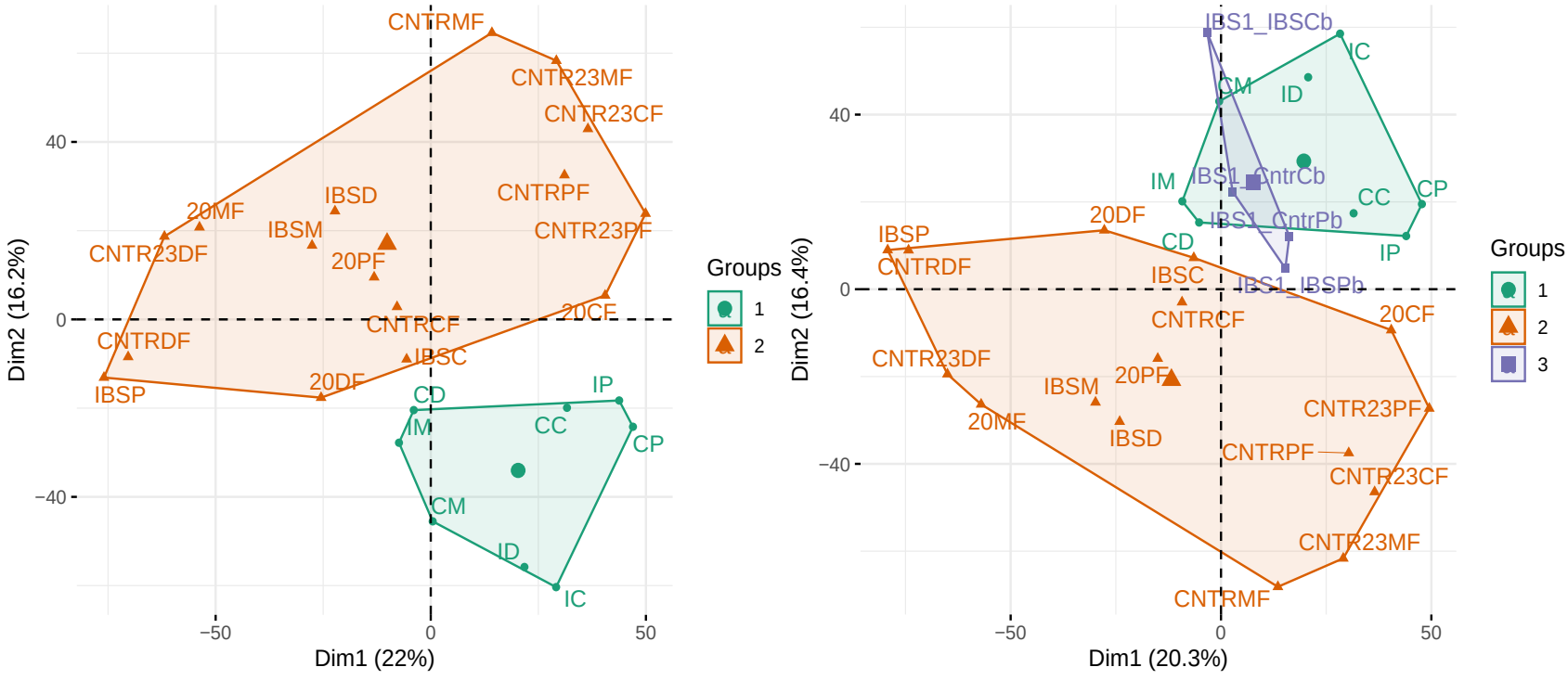
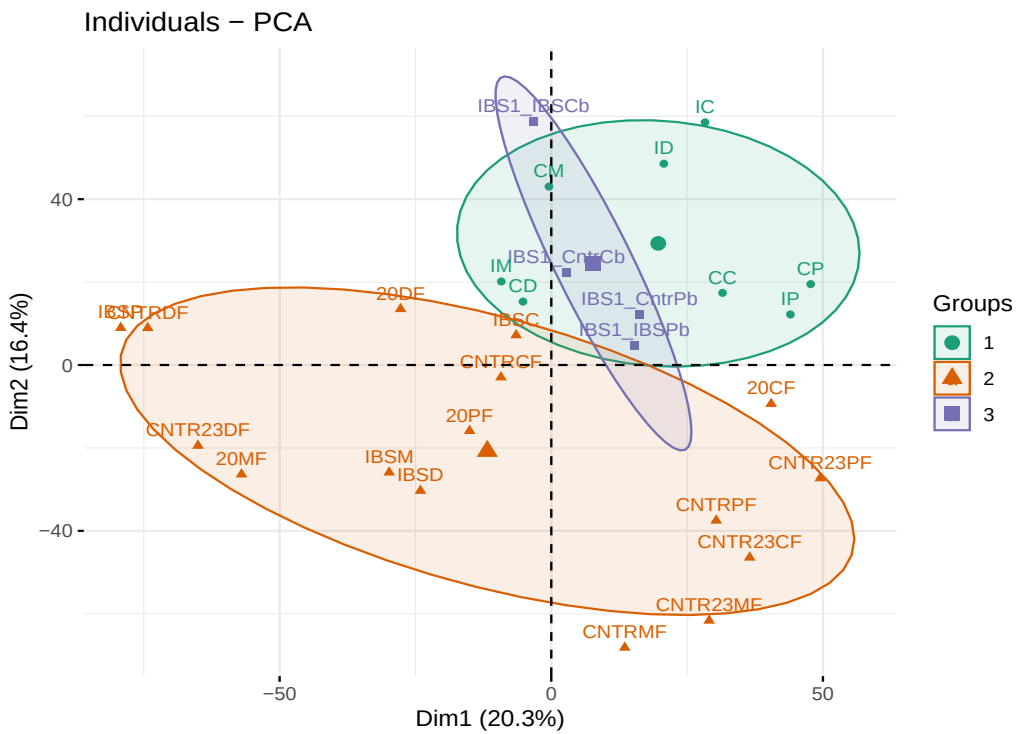


Fig. 8: When comparing the three day analysis to that of the first two days, we notice the inversion of signs in the coordinates on the second axis: this has no biological relevance.

Question

In which situation would it be preferable to make confidence ellipses around the group means using the following code?

```
fviz_pca_ind(pcaDay123, habillage = day, labelsize = 3,
  palette = "Dark2", addEllipses = TRUE, ellipse.level = 0.69)
```



Screeplot

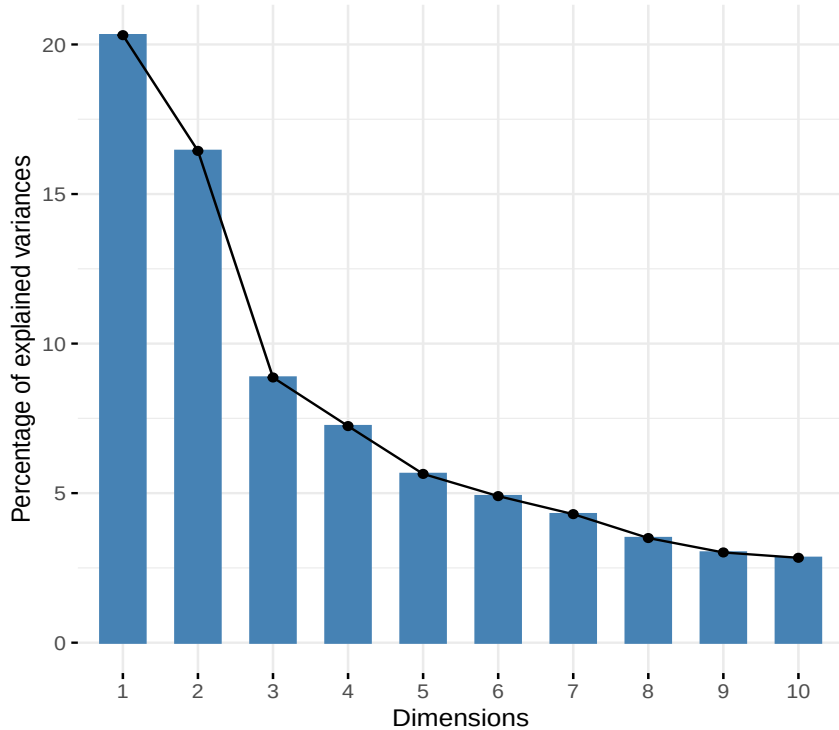


Fig. 9: The eigenvalue screeplot the case of 3 groups is extremely similar to that with two groups.

Removing batch effects

- Through the combination of the continuous measurements from `assayIBD` and the **supplementary** batch number as a factor, the PCA map has provided an invaluable investigation tool. This is a good example of the use of **supplementary points**.
- We can decide to re-align the three groups by subtracting the group means so that all the batches are centered on the origin.
- A slightly more effective way is to use the `ComBat` function available in the **sva**. This function uses a similar, but slightly more sophisticated Empirical Bayes mixture approach (Leek,2010).

Removing batch effects

We can see it's effect on the data by redoing our robust PCA.

```
model0 = model.matrix(~1,day)
combatIBD = ComBat(dat=assayIBD, batch=day, mod=model0)

pcaDayBatRM = rankthreshPCA(combatIBD)
fviz(pcaDayBatRM, element = "ind", geom = c("point", "text"),
  habillage = day, repel=TRUE, palette = "Dark2", addEllipses = TRUE,
  ellipse.type = "convex", axes =c(1,2)) + coord_fixed() + ggtitle("")
```

Removing batch effects

```
## Standardizing Data across genes
```

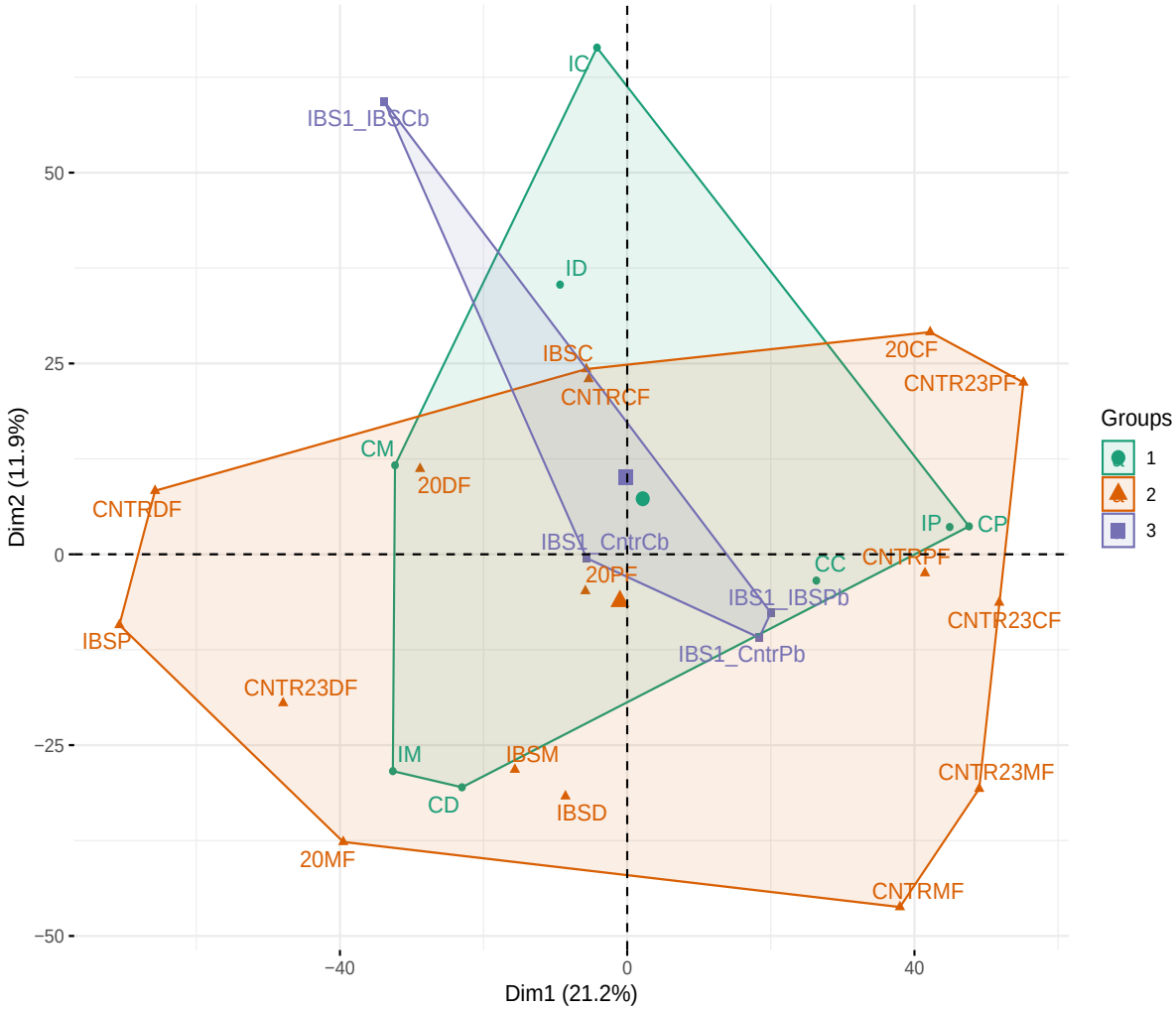


Fig. 10: The modified data with the batch effects removed now show three batch-groups heavily overlapping and centered almost at the origin.

Hybrid data and Bioconductor containers

A rational way of combining the batch and treatment information into compartments of a composite object is to use the *SummarizedExperiment* classes.

- These include special *slots* for the assay(s) where rows represent features of interest (e.g., genes, transcripts, exons, etc.) and columns represent samples.
- Supplementary information about the features can be stored in a *DataFrame* object, accessible using the function `rowData`. Each row of the *DataFrame* provides information on the feature in the corresponding row of the *SummarizedExperiment* object.
- A confusing notational similarity occurs here, in the *SummarizedExperiment* framework a *DataFrame* is not the same as a `data.frame`.

Hybrid data and Bioconductor containers

For example, we insert the two covariates day and treatment in the `colData` object and combine it with assay data in a new *SummarizedExperiment* object.

```
library("SummarizedExperiment")
sampletypes = c("IBS","CTL")
status = c(1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
           2, 2, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)
colData = DataFrame(day=day, treatment=factor(sampletypes[status]))
chipse = SummarizedExperiment(assays=list(abund = assayIBD),
                              colData=colData)
```

This is the best way to keep all the relevant data together.

It will also enable you to quickly filter the data while keeping all the information aligned properly.

Example from single cell experiment

We use an example of hybrid data container from single cell experiments (see Bioconductor workflow in Perraudeau, 2017 for more details).

```
corese = readRDS("data/normse.rds")
norm = assays(corese)$normalizedValues
```

- After the pre-processing and normalization steps prescribed in the workflow, we retain the 1,000 most variable genes measured on 747 cells.
- Columns of the *DataFrame* represent different attributes of the features of interest, e.g., gene or transcript IDs, etc.

Screeplot

```
respca = dudi.pca(t(norm), nf = 3, scannf = FALSE)
fviz_eig(respca, ncp=15)
```

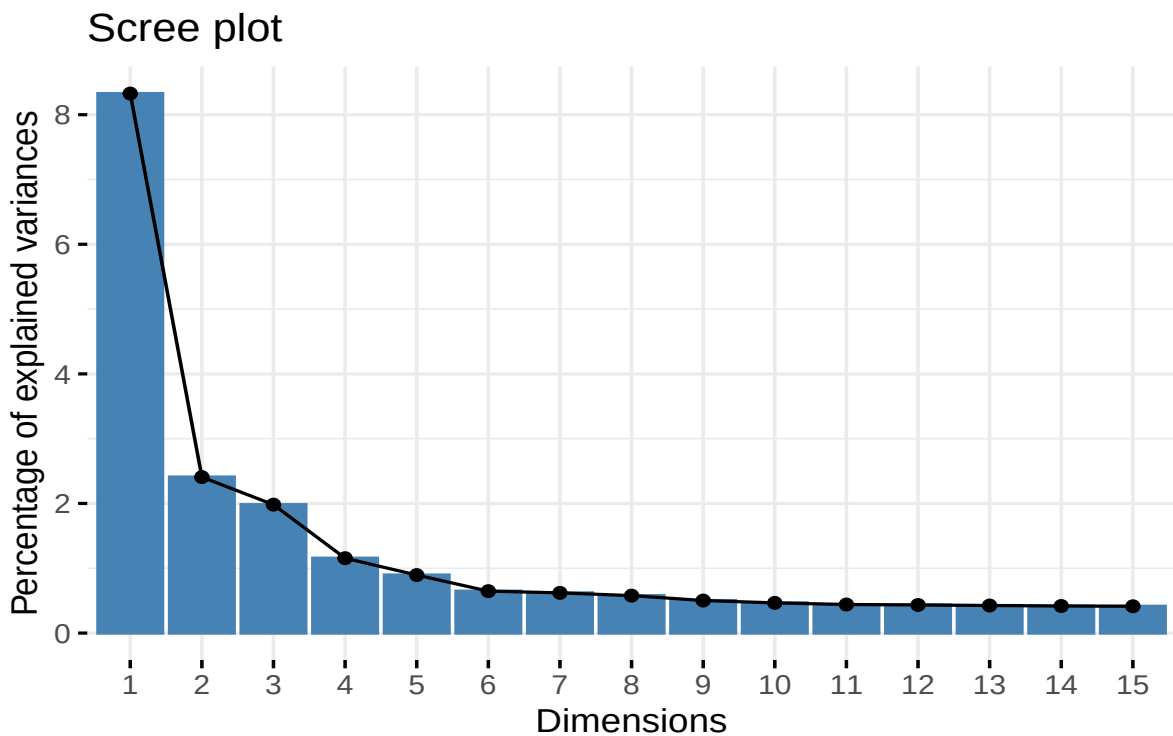


Fig. 11: Screeplot of the PCA of the normalized data.

3D plot

Since the screeplot shows us that we must not dissociate axes 2 and 3, we will make a three dimensional plot with the **rgl** package.

```
library("rgl")
PCS = respca$li[, 1:3]
batch = colData(corese)$Batch
plot3d(PCS,aspect=sqrt(c(84,24,20)),
       col=col_clus[batch])
plot3d(PCS,aspect=sqrt(c(84,24,20)),
       col = col_clus[as.character(publishedClusters)])
```

3D plot

With plotly:

```
library(plotly)
p <- plot_ly(PCS,x=~Axis1,y=~Axis2,z=~Axis3,color=batch) %>% add_markers()
```

Correspondence analysis

When the data are counts from cross tabulating categorical variables, we cannot use PCA.

What is a contingency table?

	black	*blue*	*green*	*grey*	*orange*	*purple*	*white*
quiet	27700	21500	21400	8750	12200	8210	25100
angry	29700	15300	17400	7520	10400	7100	17300
clever	16500	12700	13200	4950	6930	4160	14200
depressed	14800	9570	9830	1470	3300	1020	12700
happy	193000	83100	87300	19200	42200	26100	91500
lively	18400	12500	13500	6590	6210	4880	14800
perplexed	1100	713	801	189	233	152	1090
virtuous	1790	802	1020	200	247	173	1650

Categorical data representations

Sample by mutation matrix (the long version representation):

Patient	*Mutation*1	*Mutation*2	*Mutation*3	...
*AHY*789	0	0	0	
*AHX*717	1	0	1	
*AHX*543	1	0	0	

are transformed into a Mutation $\times$ Mutation matrix:

	*Mutation*1	*Mutation*2	*Mutation*3	...
*Mutation*1	853	29	10	
*Mutation*2	29	853	52	
*Mutation*3	10	52	853	

Example: hair color, eye color

```
require(ade4)
HairColor=HairEyeColor[,,2]
HairColor
```

```
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    36    9     5     2
##   Brown    66   34    29    14
##   Red      16    7     7     7
##   Blond     4   64     5     8
```

```
chisq.test(HairColor)
```

```
##
##  Pearson's Chi-squared test
##
## data:  HairColor
## X-squared = 106.66, df = 9, p-value < 2.2e-16
```

Conclusion:

Hair color and eye color are not independent: the categories show a pattern of dependency. What can we say about them?

Independence: computationally

```
rowsums=as.matrix(apply(HairColor,1,sum))
rowsums
```

```
##        [,1]
## Black    52
## Brown   143
## Red      37
## Blond    81
```

```
colsums=as.matrix(apply(HairColor,2,sum))
colsums
```

```
##        [,1]
## Brown   122
## Blue    114
## Hazel    46
## Green    31
```

Independence: computationally

```
HCexp=rowsums%*%t(colsums)/sum(colsums)
# Here is actually how the chisquare is computed
sum((HairColor - HCexp)^2/HCexp)
```

```
## [1] 106.6637
```

```
round(t(HairColor-HCexp))
```

```
##         Hair
## Eye      Black Brown Red Blond
##   Brown     16    10   2   -28
##   Blue     -10   -18  -6    34
##   Hazel     -3     8   2    -7
##   Green     -3     0   3     0
```

Independence: computationally

```
require(vcd)
# mosaicplot(HairColor,shade=TRUE,las=1,type="pearson",cex.axis=0.7,main="")
mosaic(HairColor,shade=TRUE)
```

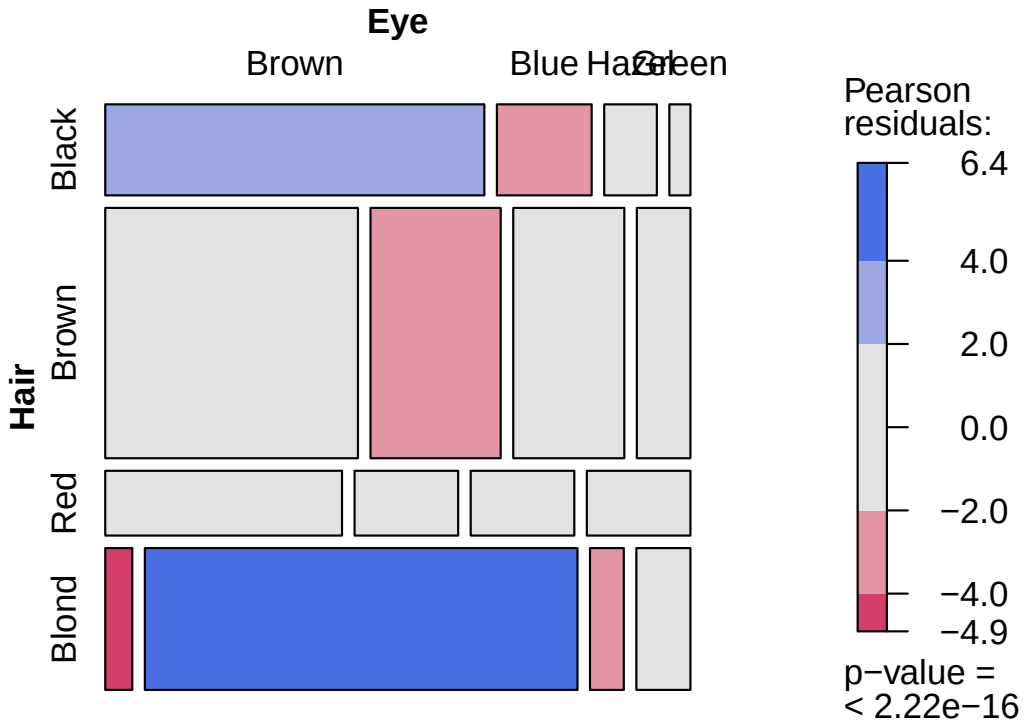


Fig. 12: Visualization of the departure from independence.

Independence: mathematically

Suppose we are comparing two categorical variables, (hair color, eye color).

For a $I \times J$ contingency table with a total sample size of $n = \sum_{i=1}^{I} \sum_{j=1}^{J} n_{ij}$, let n_{i+} be the sum of the i-th row, n_{+j} sum of the j-th column, and

$$\hat{\mu}_{ij} \doteq \frac{n_{i+}}{n} \times \frac{n_{+j}}{n} \times n.$$

The departure from independence is measured by the χ^2 **statistic**

$$\mathcal{X}^2 = \sum_{i,j} \left\{ \frac{(n_{ij} - \hat{\mu}_{ij})^2}{\hat{\mu}_{ij}} \right\}.$$

Correspondence analysis implementations

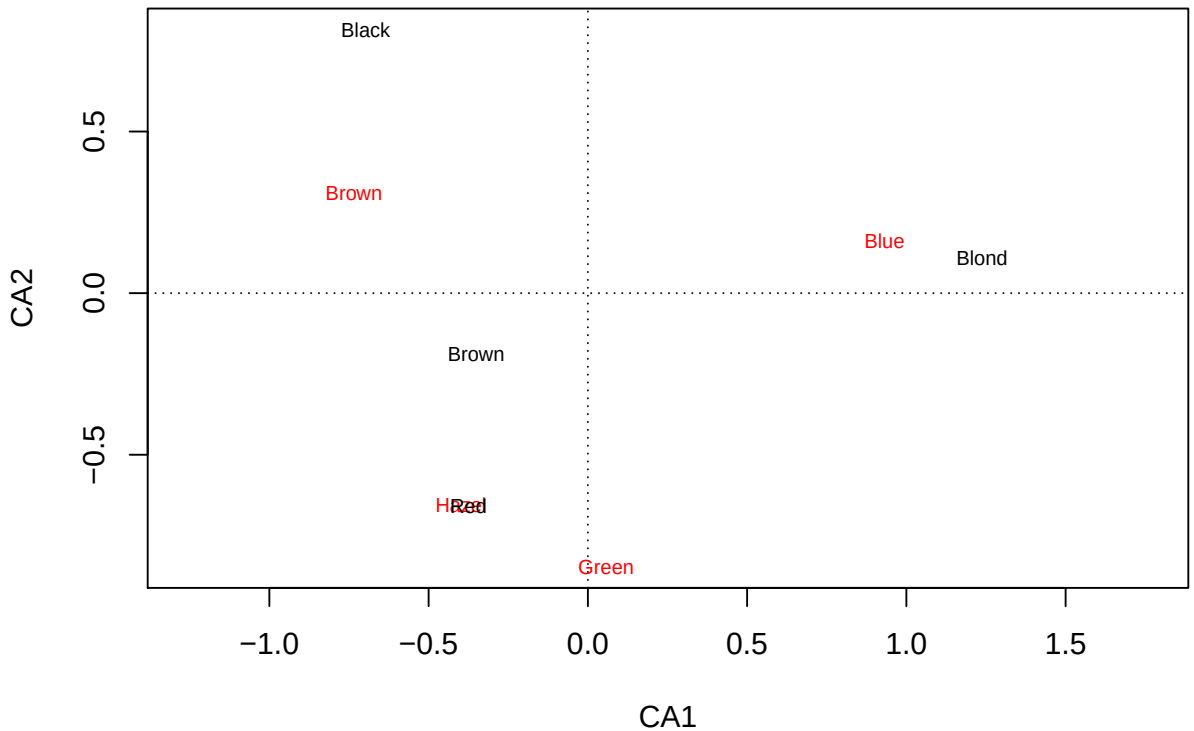
Many implemetations:

- `dudi.coa` in ade4
- `CCA` in vegan
- `ordinate` in phyloseq

```
library("ade4")
HairColor = HairEyeColor[,,2]
chisq.test(HairColor)
```

CCA in vegan

```
library(vegan)
res.ca=vegan::cca(HairColor)
plot(res.ca,scaling=3)
```



Example: Plato's sentence endings

TABLE 1

Percentage Distribution of Sentence Endings

Type of Ending	Π_0 Rep.	Π_1 Laws	$\hat{s}_t$ = Natural Log of Ratio	Crit.	Phil.	Pol.	Soph.	Tim
⏑ ⏑ ⏑ ⏑ ⏑ .	1·1	2·4	0·779	3·3	2·5	1·7	2·8	2·4
– ⏑ ⏑ ⏑ ⏑ ⏑ .	1·6	3·8	0·867	2·0	2·8	2·5	3·6	3·9
⏑ – ⏑ ⏑ ⏑ ⏑ .	1·7	1·9	0·113	2·0	2·1	3·1	3·4	6·0
⏑ ⏑ – ⏑ ⏑ ⏑ .	1·9	2·6	0·315	1·3	2·6	2·6	2·6	1·8
⏑ ⏑ ⏑ – ⏑ ⏑ .	2·1	3·0	0·358	6·7	4·0	3·3	2·4	3·4
⏑ ⏑ ⏑ ⏑ – ⏑ .	2·0	3·8	0·642	4·0	4·8	2·9	2·5	3·5
– – ⏑ ⏑ ⏑ ⏑ .	2·1	2·7	0·255	3·3	4·3	3·3	3·3	3·4
– ⏑ – ⏑ ⏑ ⏑ .	2·2	1·8	−0·199	2·0	1·5	2·3	4·0	3·4
⏑ – – ⏑ ⏑ ⏑ .	2·8	0·6	−1·541	1·3	0·7	0·4	2·1	1·7
– ⏑ ⏑ – ⏑ ⏑ .	4·6	8·8	0·647	6·0	6·5	4·0	2·3	3·3
⏑ – ⏑ ⏑ – ⏑ .	3·3	3·4	0·030	2·7	6·7	5·3	3·3	3·4
⏑ ⏑ – ⏑ ⏑ – .	2·6	1·0	−0·956	2·7	0·6	0·9	1·6	2·2
⏑ ⏑ ⏑ – ⏑ – .	4·6	1·1	−1·430	2·0	0·7	1·0	3·0	2·7
⏑ ⏑ ⏑ ⏑ – – .	2·6	1·5	−0·548	2·7	3·1	3·1	3·0	3·0
⏑ ⏑ ⏑ – – – .	4·4	3·0	−0·385	3·3	1·9	3·0	3·0	2·2
⏑ ⏑ – ⏑ – – .	2·5	5·7	0·824	6·7	5·4	4·4	5·1	3·9
– – – ⏑ ⏑ ⏑ .	2·9	4·2	0·372	2·7	5·5	6·9	5·2	3·0
– – – ⏑ ⏑ – .	3·0	1·4	−0·761	2·0	0·7	2·7	2·6	3·3
– – – ⏑ – ⏑ .	3·4	1·0	−1·224	0·7	0·4	0·7	2·3	3·3
– ⏑ – – ⏑ – .	2·0	2·3	0·140	2·0	1·2	3·4	3·7	3·3
– ⏑ – ⏑ – – .	6·4	2·4	−0·982	1·3	2·8	1·8	2·1	3·0
– ⏑ ⏑ – – – .	4·2	0·6	−1·946	4·7	0·7	0·8	3·0	2·8
⏑ ⏑ – – – – .	2·8	2·9	0·039	1·3	2·6	4·6	3·4	3·0
⏑ – ⏑ – – – .	4·2	1·2	−1·253	2·7	1·3	1·0	1·3	3·3
⏑ – – ⏑ – – .	4·8	8·2	0·536	5·3	5·3	4·5	4·6	3·0
⏑ – – – ⏑ – .	2·4	1·9	−0·231	3·3	3·3	2·5	2·5	2·2
⏑ – – – – – .	3·5	4·1	0·157	2·0	3·3	3·8	2·9	2·4
– ⏑ – – – – .	4·0	3·7	−0·077	4·7	3·3	4·9	3·5	3·0
– – ⏑ – – – .	4·1	2·1	−0·668	6·0	2·3	2·1	4·1	6·4
– – – ⏑ – – .	4·1	8·8	0·765	2·0	9·0	6·8	4·7	3·8
– – – – ⏑ – .	2·0	3·0	0·405	3·3	2·9	2·9	2·6	2·2
– – – – – .	4·2	5·2	0·215	4·0	4·9	7·3	3·4	1·8
Number of sentences .	3,778	3,783	—	150	958	770	919	762

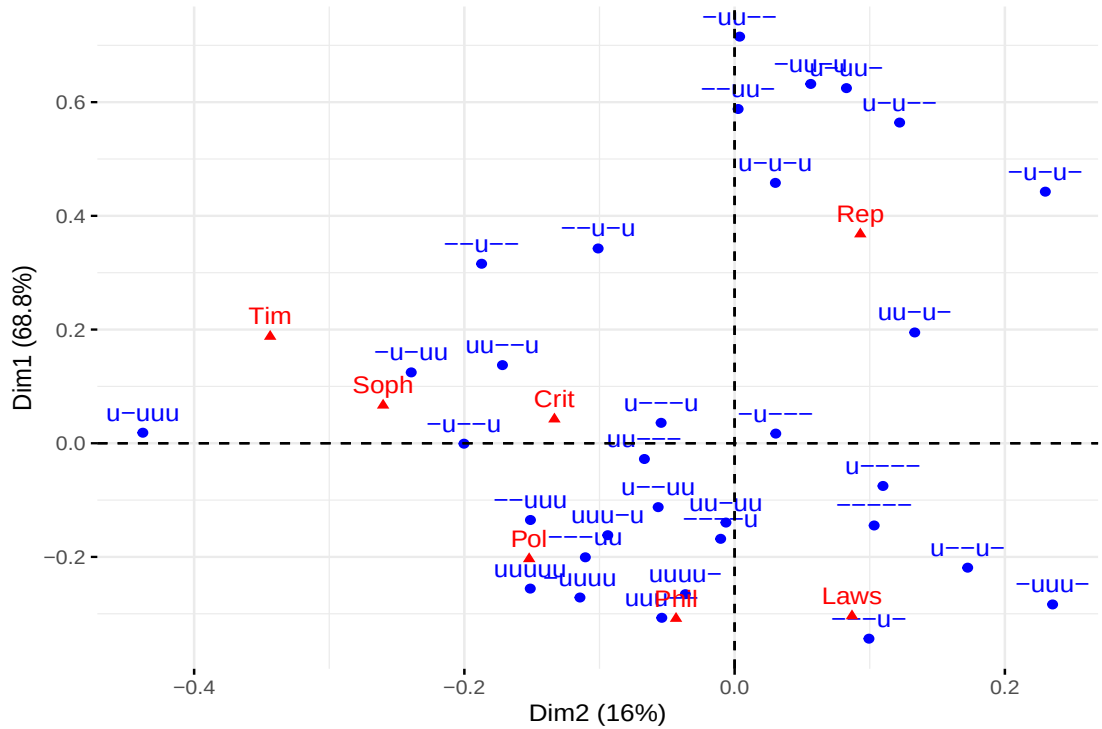
There are minor errors in the third decimal place of the scores $\hat{s}_t$.

Fig. 13: Table from Brandwood and Cox

The date Plato wrote these various 'books' is not known. We use these pattern frequencies as the data.

Example: Plato's sentence endings

```
platof=read.table("data/platof.txt",header=T)
# platof[1:8,]
res.plato=dudi.coa(platof,scannf=FALSE,nf=2)
fviz_ca_biplot(res.plato, axes=c(2, 1)) + ggtitle("")
```



Questions

- From the biplot in Figure 9.33 can you guess at the chronological order of Plato's works?
- Which sentence ending did Plato use more frequently early in his life?

More about gradients

The Boomer Lake example: Arch Effect for Ecologists

Two species count matrices

```
load("data/lakes.RData")
lakelike[1:2,1:2]
```

```
##      plant1 plant2
## loc1      6      4
## loc2      4      5
```

```
lakelikeh[1:2,1:2]
```

```
##      plant1 plant2
## loc1      6      4
## loc2      4      5
```

```
reslake =dudi.coa(lakelike,scannf=FALSE,nf=2)
reslakeh=dudi.coa(lakelikeh,scannf=FALSE,nf=2)

reslake2 =dudi.pca(lakelike,scannf=FALSE,nf=2)
reslakeh2=dudi.pca(lakelikeh,scannf=FALSE,nf=2)
```

PCA on the first data set

```
fviz_pca_ind(reslake2,repel=TRUE)+ggtitle("")
```

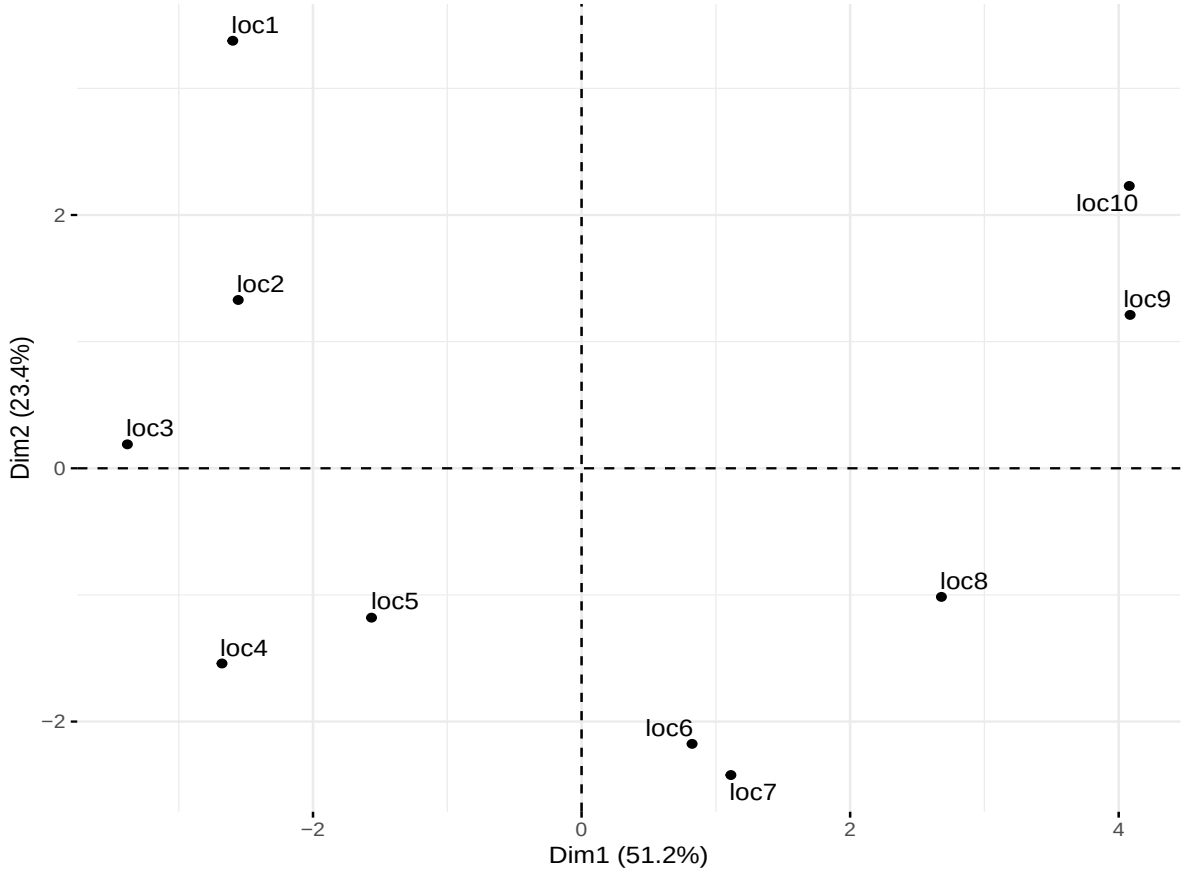


Fig. 14: Ordination with PCA

PCA on the first data set

```
fviz_pca_biplot(reslake2,repel=TRUE)+ggtitle("")
```

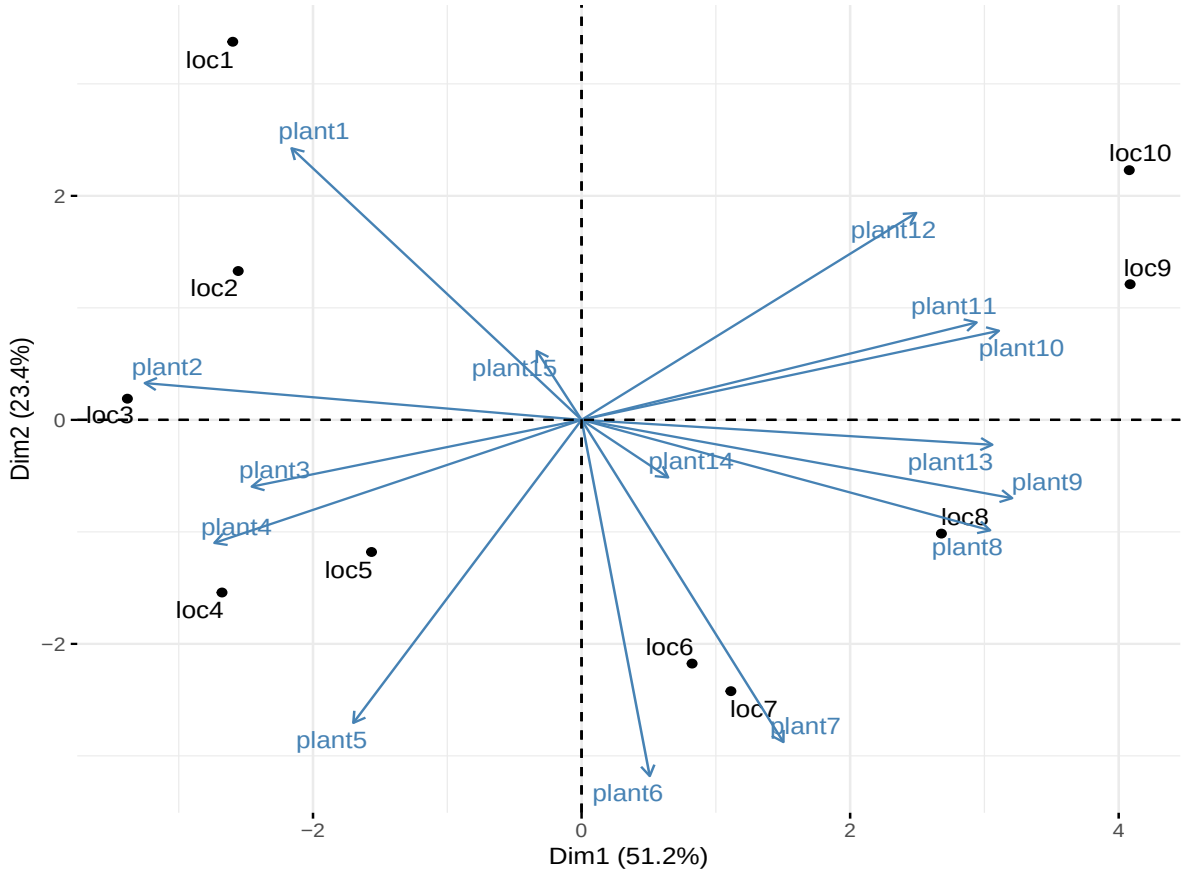


Fig. 15: PCA biplot

Correspondence analysis on the first data set

```
fviz_ca_row(reslake,repel=TRUE)+ggtitle("")
```

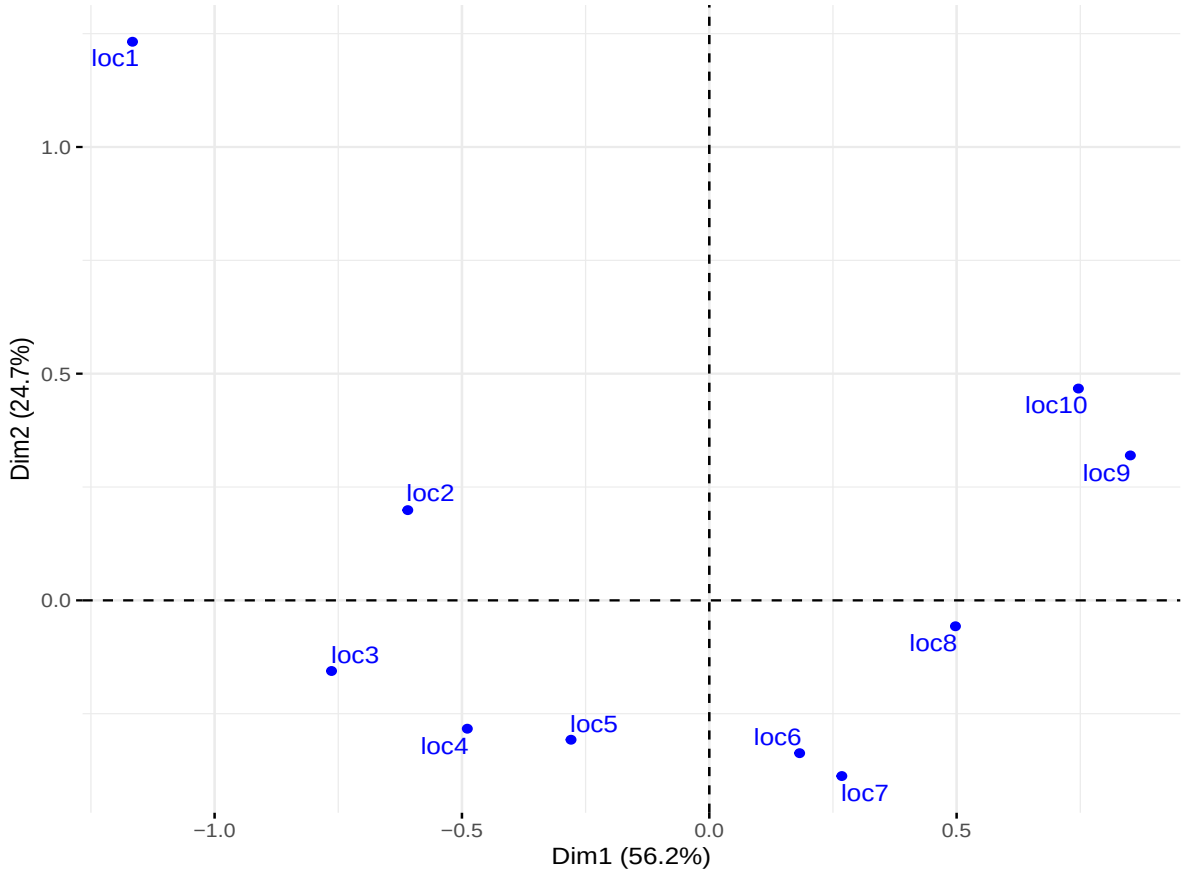


Fig. 16: Ordination with correspondence analysis

Correspondence analysis on the first data set

```
fviz_ca_biplot(reslake,repel=TRUE)+ggtitle("")
```

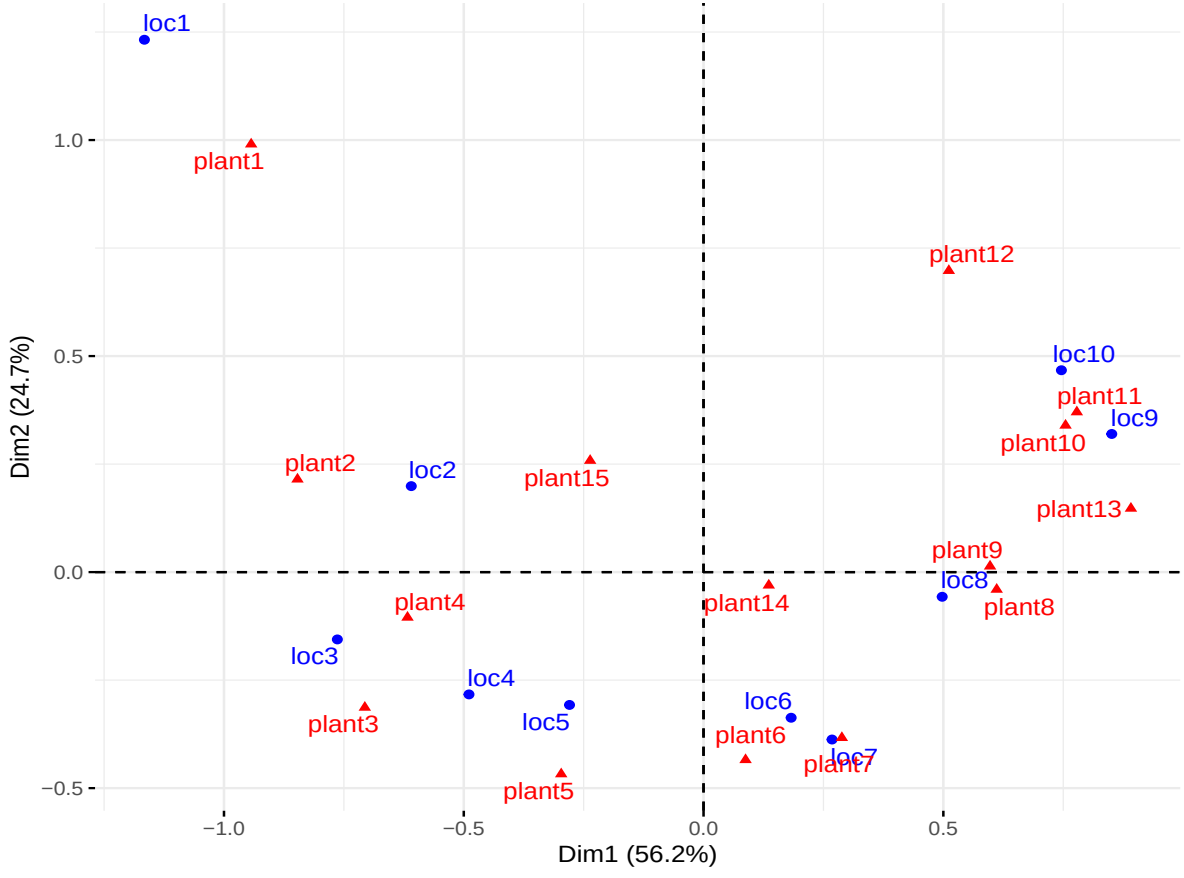


Fig. 17: Correspondence analysis biplot

Exercises: PCA and CA on the second data set

You can use the `fviz_pca`, `fviz_ca` in the **factoextra** package or the `scatter` function **ade4** package. More examples are available at sthda.

```
scatter(reslakeh2)
# s.label(reslakeh2$li)

scatter(reslakeh)
# s.label(reslakeh$li)
# s.label(reslakeh2$li)
```

Summary of multivariate analysis

- Continuous variables: PCA (`dudi.pca`)
- Count data or binary data in contingency tables: Correspondence Analysis (`dudi.coa`)
- Distances between objects (samples): MDS or PCoA (`cmdscale`)

Graphical and interpretational aides

- Screeplot
- Biplot