Introduction to PCA

B. Liquet

Goal of Integrative Analysis:

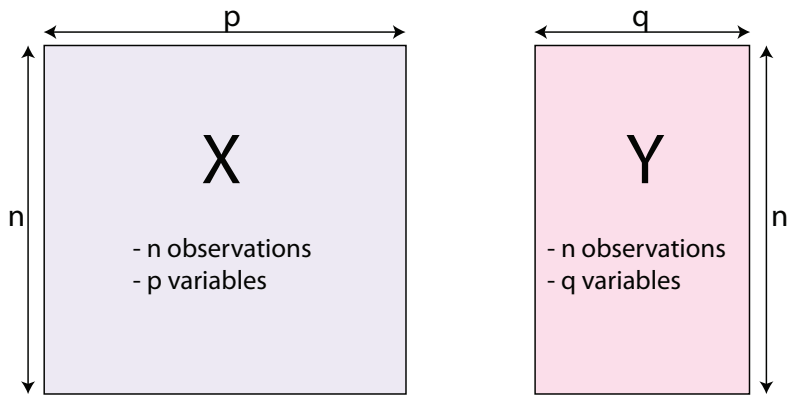
Wikipedia. **Data integration** “involves **combining data** residing in different sources and providing users with a unified view of these data. This process becomes significant in a variety of situations, which include both commercial and **scientific**".

System Biology. **Integrative Analysis:** Analysis of heterogeneous types of data from inter-platform technologies.

Goal. Combine multiple types of data:

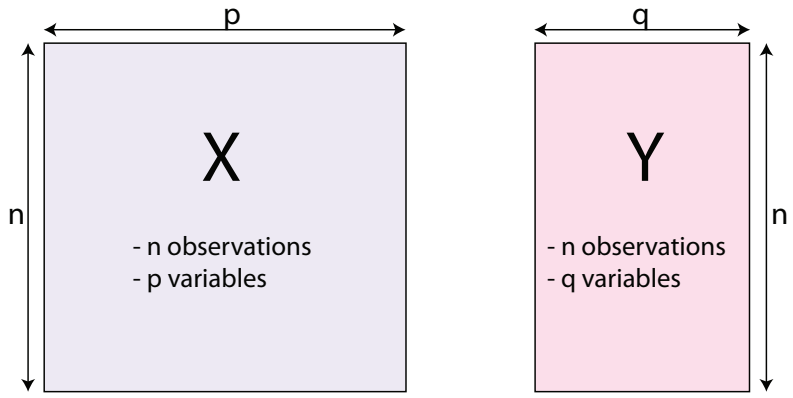
- Contribute to a better understanding of biological mechanism.
- Have the potential to improve the diagnosis and treatments of complex diseases.

Example: Data definition



- "Omics." **Y** matrix: gene expression, **X** matrix: SNP (single nucleotide polymorphism). Many others such as proteomic, metabolomic data.
- "neuroimaging". **Y** matrix: behavioral variables, **X** matrix: brain activity (e.g., EEG, fMRI, NIRS)
- "neuroimaging genetics." **Y** matrix: fMRI (Fusion of functional magnetic resonance imaging), **X** matrix: SNP
- "Ecology/Environment." **Y** matrix: Water quality variables , **X** matrix: Landscape variables

Questions



Some possible questions:

- How to investigate the relation of these two blocks of Data?
- How to dentify which of the p variables in X (OMICs data) are associated with the outcome Y (disease status or correlated continuous marker)
- Integrative Omics Analysis: Can we identify a subset of correlated genes and proteins?

Constraints

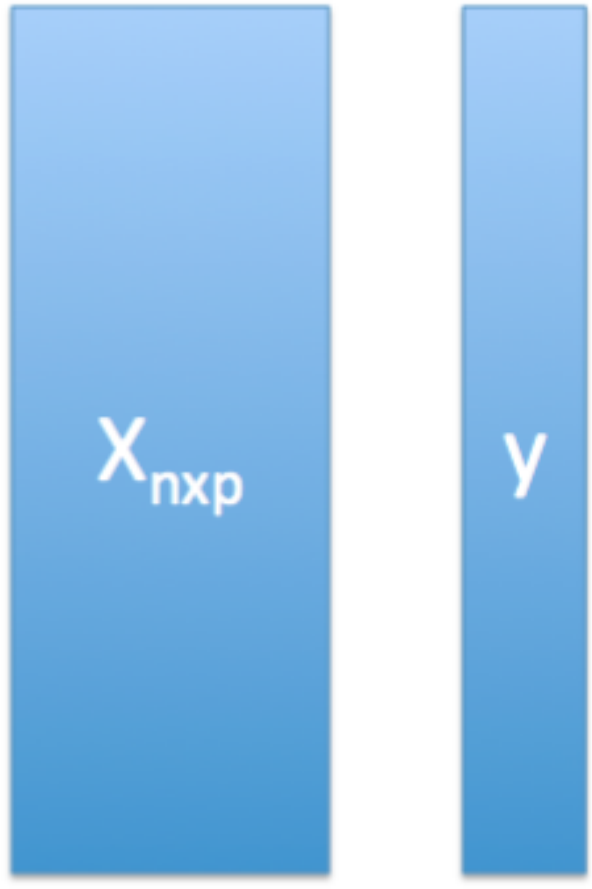
- The $n < p$ situation:
  - More predictors than observations

    $\Rightarrow$ numerically intractable statistical inferences
  - Data from multiple source.

Low-Dimensional Versus High-Dimensional

- The data set that you used to analyse in traditional statistics course is low-dimensional: $n >> p$
- Lots of the data sets coming out of modern biological techniques are high-dimensional: $n \simeq p$ or $n << p$.
- This poses statistical challenges! For example Linear model no longer applies.

Low Dimensional



High Dimensional

Dimension Reduction

Dimension Reduction methods is a popular approach to solve the high dimensional situation ($p >> n$).

Dimension reduction techniques summarize X into a lower dimension matrix.

Why Dimension Reduction?

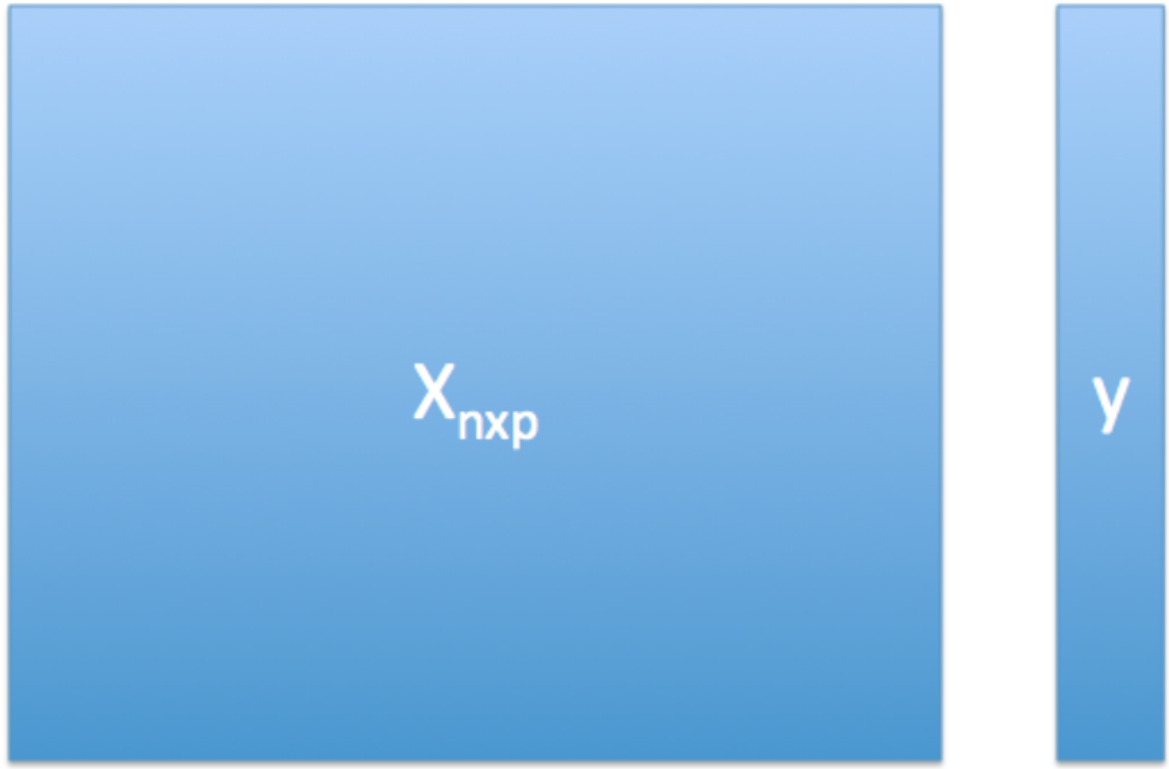
- Some features may be irrelevant
- We want to visualize high dimensional data
- High dimensional data often have high degrees of redundancy (correlation among features).

Dimension Reduction enables us to:

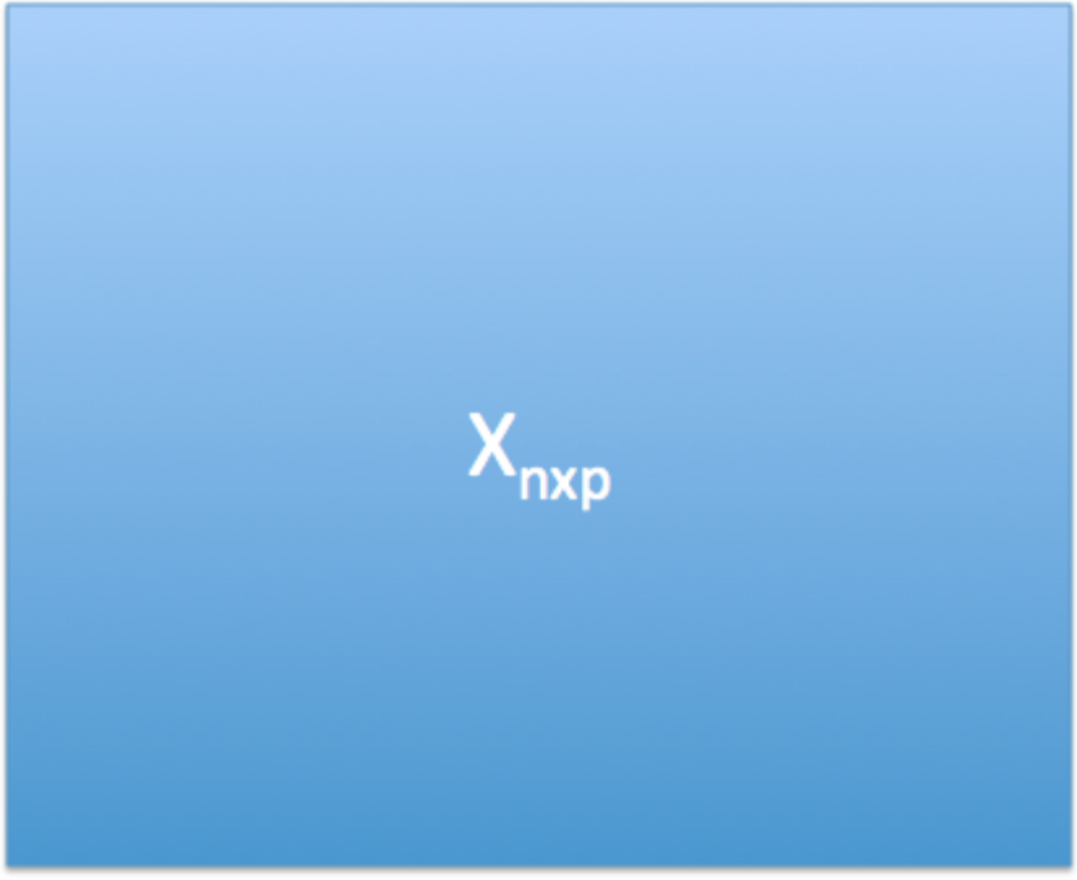
- Map the data into a new low-dimensional space, where important characteristics of the data are preserved.
- The new space often gives a (linear or non-linear) transformation of the original data.
- Visualization and analysis (clustering/prediction/...) is then performed in the new space.

Supervised Learning or Unsurvised Leraning ?

Supervised Learning



Unupervised Learning



- no response variable
- In biological applications, often $p >> n$.

Unsupervised Learning Examples

- Do genes/samples in microarray expression data form interesting groups?
- Can individuals' SNP profiles be used to learn about their ethnic/racial backgrounds?
- Can we find cancer subtypes based on gene/metabolic expression patterns?
- What's the best way to visualize a high dimensional genomic data?
- How to find "interesting patterns" in the data?

Principal Component Analysis

Principal Component Analysis, or PCA, is a well-known and widely used technique applicable to a wide variety of applications such as dimensionality reduction, data compression, feature extraction, and visualization.

The basic idea is to project a dataset from many correlated coordinates onto fewer uncorrelated coordinates called principal components while still retaining most of the variability present in the data.

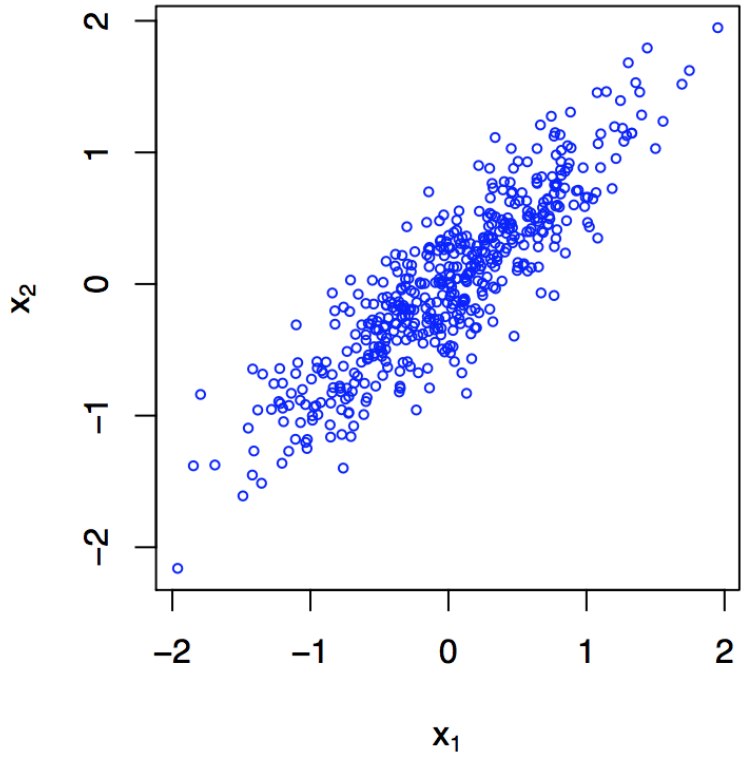
As an example, principal component analysis is commonly performed to account for population stratification in genome wide association study.

Definition: Principal Component Analysis (PCA)

- Data: n observations living in a p-dimensional space.
- Not all p dimensions are equally useful, especially when $p >> n$.
- Many are either completely redundant (correlated features) or uninformative (noise features).
- Need low-dimensional representation of the variables that captures most of the "information" in the data.
- To maximize the information retained, we need to minimize the redundancy, and to do this, we look for low-dimensional representations that capture most of the variation in the data.

PCA: The main Idea

What is a good 1-dim representation of the data?



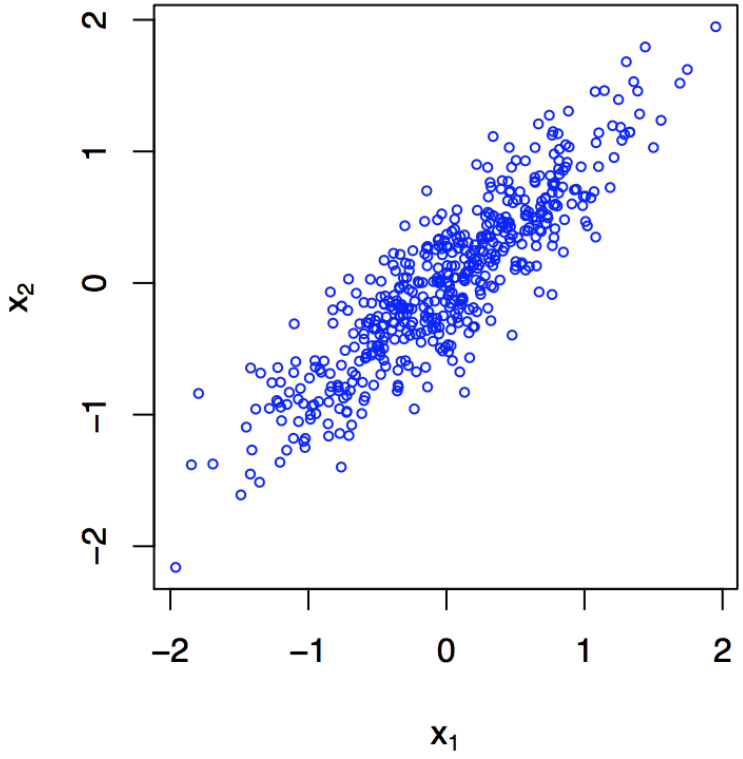
{ Use a linear combination of the variables; i.e. a weighted average of the variables.

$$\boldsymbol{c}_1 = w_1\boldsymbol{x}_1 + w_2\boldsymbol{x}_2$$

}

PCA: The main Idea

What is a good 1-dim representation of the data?



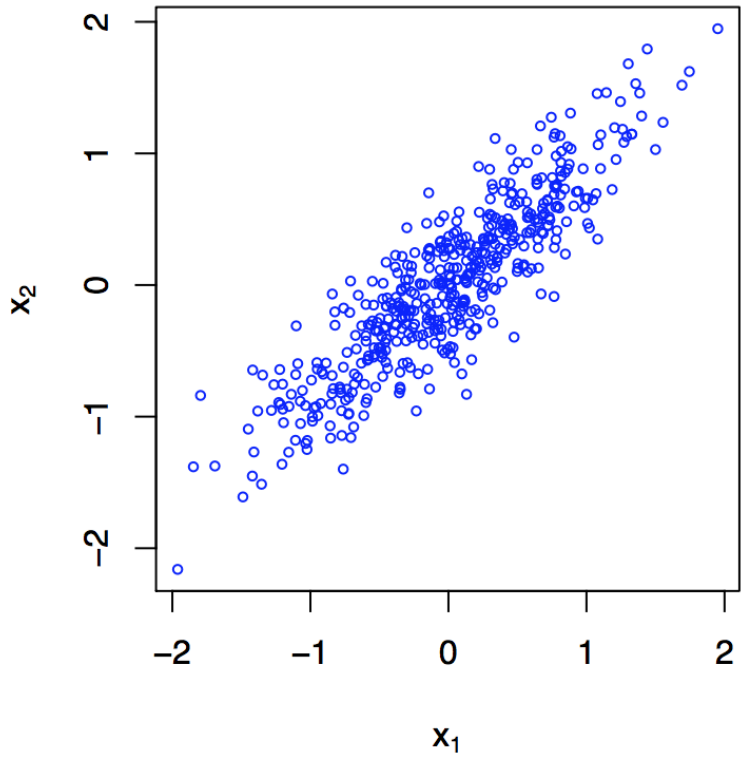
{ Use a linear combination of the variables; i.e. a weighted average of the variables.

$$\boldsymbol{c}_1 = w_1\boldsymbol{x}_1 + w_2\boldsymbol{x}_2$$

} { What is a good choice for the weights w_1 and w_2? }

The Criterion for Principal Components

In PCA, we try to find the direction with maximum variance



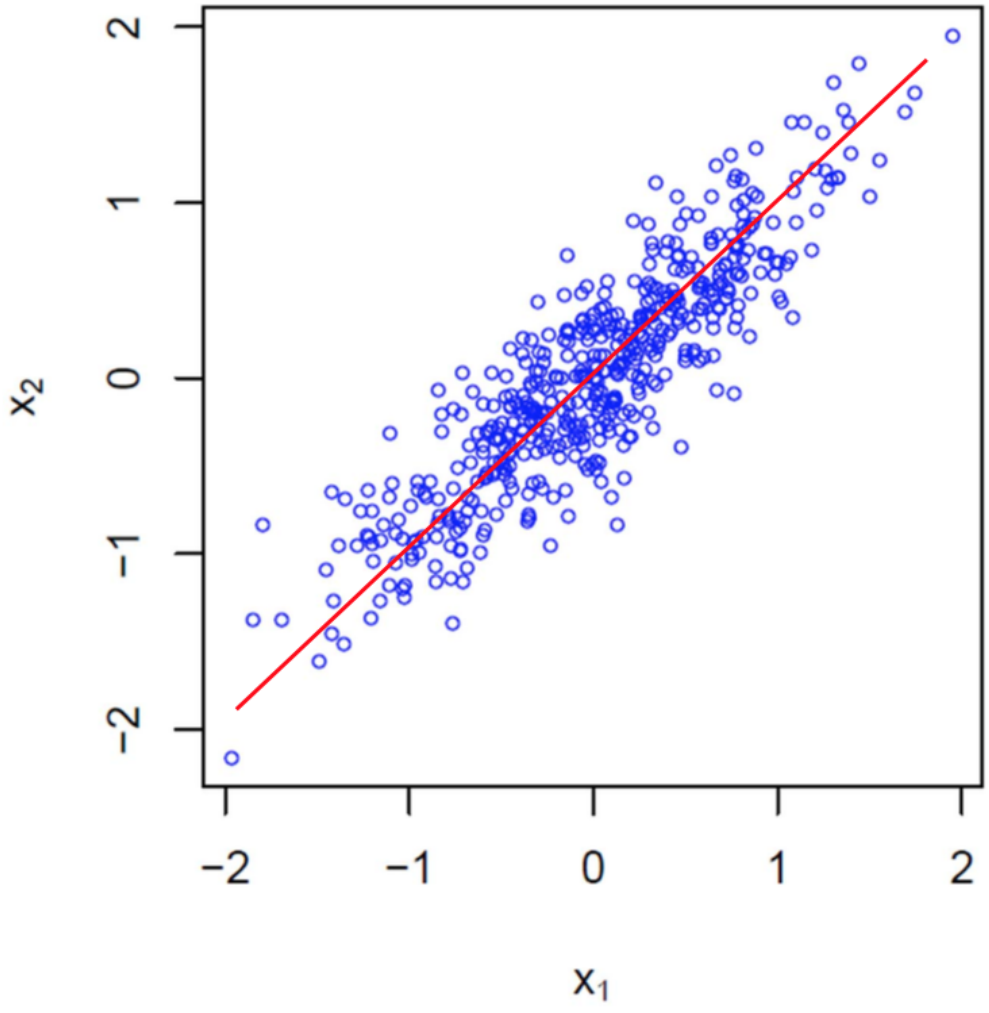
{ Formally, we seek the vector of weights $\boldsymbol{v} = (w_1, w_2)^T$ using the criterion

$$\max_{\|\boldsymbol{v}\|=1} Var(X\boldsymbol{v})$$

}

The 1-Dimensional PCA Solution

The interesting direction according to the PCA criterion is the one that captures the majority of the variance in the data.

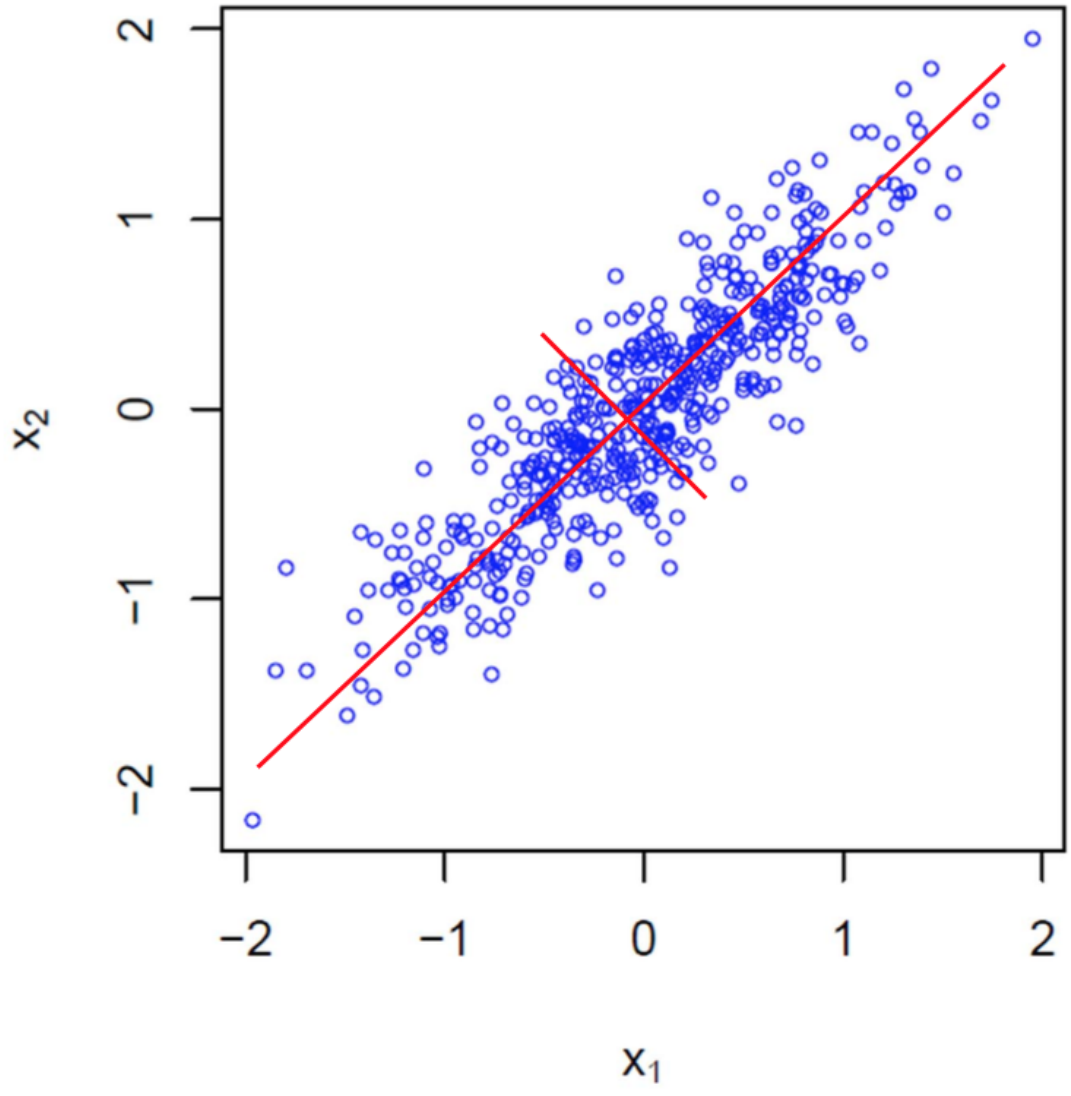


The 2-Dimensional PCA

- But, what if we need another direction.
- A systematic way to find additional principal components (PC's), is to choose subsequent linear combinations orthogonal/perpendicular to previous ones.
- This means that we want to choose v to be orthogonal to w, but to explain the majority of variability in the data.
- In the case of 2-dimensional data, there is only one choice!! This is always the case for the last PC.
- For $p > 2$, there are many orthogonal vectors to choose from, and we need to find the one that explains the maximum variation in the data, and is orthogonal to the first one.

The Full PCA Solution for 2 Dimensions

What is a good 1-dim representation of the data?



PCA: Summary

Seek the best directions in the data that account for most of the variability

→ principal components: artificial variables that are linear combinations of the original variables:

$$\underset{(n)}{\boldsymbol{c}} = \underset{(nxp)}{\boldsymbol{X}} \quad \underset{(p)}{\boldsymbol{v}}$$

- $\boldsymbol{c}$ is a linear combination of the elements of $\boldsymbol{X}$ having maximal variance
- $\boldsymbol{v}$ is called the associated loading vector
- For example, we get the first principal component with max. variance

$$\boldsymbol{c}_1 = w_1 \boldsymbol{x}_1 + \ldots + w_p \boldsymbol{x}_p$$

where $\boldsymbol{v}_1 = (w_1, \ldots, w_p)^T$.

PCA: Summary

- PCA $\Longrightarrow$ finds units vectors $v_1, \ldots, v_r$ that maximise the variance Xv under the constraint that v_{i+1} is orthogonal to $v_1, \ldots, v_i$
- Reduction dimension:
  p variables $\Longrightarrow$ r new variables $Xv_1, \ldots, Xv_r$ called Principal Components

The new PCs form a vectorial subspace of dimension $< p$

- Project the data on these new axes.

  $\rightarrow$ approximate representation of the data points in a lower dimensional space

- The first few PCs account for most of the variation in all the original variables

Cons:

- Interpretation difficult with very large number of variables
- Unsupervised approach

Objective function for PCA

Objective function:

$$\max_{\|\boldsymbol{v_h}\|=1} var(X_h \boldsymbol{v_h}), \qquad h = 1 \ldots H$$

Several ways of solving PCA:

1. Eigenvalue problem: $S\boldsymbol{v} = \lambda \boldsymbol{v}$; $\boldsymbol{c} = X\boldsymbol{v}$

   S = variance covariance matrix or correlation matrix if X is scaled

2. Singular Value Decomposition (SVD):

   $$X = UDV^T$$

   where the columns of $U(n \times r)$ and V $(p \times r)$ are orthonormal; $D = diag[d_1, \ldots, d_r]$ such that $d_1 \geqslant d_2 \geqslant \ldots \geqslant d_r > 0$ are the square root of the eigenvalue of $X^T X$.

Solution based on SVD Decomposition

- The column of $C = UD$ are the principal component;

- The columns of V are the corresponding loading.

- The variance of a principal component $\boldsymbol{c}_1$ is equal to its associated eigenvalue λ_1.

- The obtained eigenvalues λ_j are decreasing.

Singular Value Decomposition (SVD)

Singular Value Decomposition, or SVD, is a computational method often employed to calculate principal components for a dataset. Using SVD to perform PCA is efficient and numerically robust. (see https://intoli.com/blog/pca-and-svd/).

Let a matrix $M : p \times q$ of rank r:

$$M = U\Delta V^T = \sum_{l=1}^{r} \delta_l u_l v_l^T,$$

- $U = (u_l) : p \times p$ and $V = (v_l) : q \times q$ are two orthogonal matrices which contain the normalised left (resp. right) singular vectors
- $\Delta = \mathrm{diag}(\delta_1, \ldots, \delta_r, 0, \ldots, 0)$: the ordered singular values $\delta_1 \geqslant \delta_2 \geqslant \cdots \geqslant \delta_r > 0$.

Relation of PCA and SVD

- Let the data matrix X be of $n \times p$ size, where n is the number of samples and p is the number of variables.
- Let us assume that it is centered, i.e. column means have been subtracted and are now equal to zero.
- Then the $p \times p$ covariance matrix C is given by $C = X^T X/(n-1)$. It is a symmetric matrix and so it can be diagonalized:

$$C = VLV^T,$$

where V is a matrix of eigenvectors (each column is an eigenvector) and L is a diagonal matrix with eigenvalues λ_i in the decreasing order on the diagonal.

Relation of PCA and SVD

- The eigenvectors are called principal axes or principal directions of the data.
- Projections of the data on the principal axes are called principal components, also known as PC scores; these can be seen as new, transformed, variables.
- The j-th principal component is given by j-th column of XV. The coordinates of the i-th data point in the new PC space are given by the i-th row of XV.

Relation of PCA and SVD

- if we now perform singular value decomposition of X, we obtain a decomposition

$$X = U\Delta V^T,$$

From here one can easily see that

$$C = V\Delta^T U^T U\Delta V^T/(n-1) = V\frac{\Delta^2}{n-1}V^T,$$

meaning that right singular vectors V are principal directions and that singular values are related to the eigenvalues of covariance matrix via $\lambda_i = s_i^2/(n-1)$.

Principal components are given by $XV = U\Delta V^T V = U\Delta$.

Illustration PCA: GWAS

The dataset comes from the R package `bigstatr` . It is a subset of a Genome Wide Asssociation study.

> The variables are Single Nucleotide Polymorphism (SNPs) coded 0,1 amd 2 represented the number of rare allele at different locus (https://ghr.nlm.nih.gov/primer/genomicresearch/snp).

Principal Components Analysis (PCA) is a popular tool that has been used to infer population structure in genetic data for several decades.

Illustration PCA: R code

```
library(bigstatsr)
set.seed(1)

X <- big_attachExtdata()
n <- nrow(X)
X[1:2,1:3]
```

```
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    1    2    1
```

```
dim(X)
```

```
[1]  517 4542
```

Illustration PCA: R code

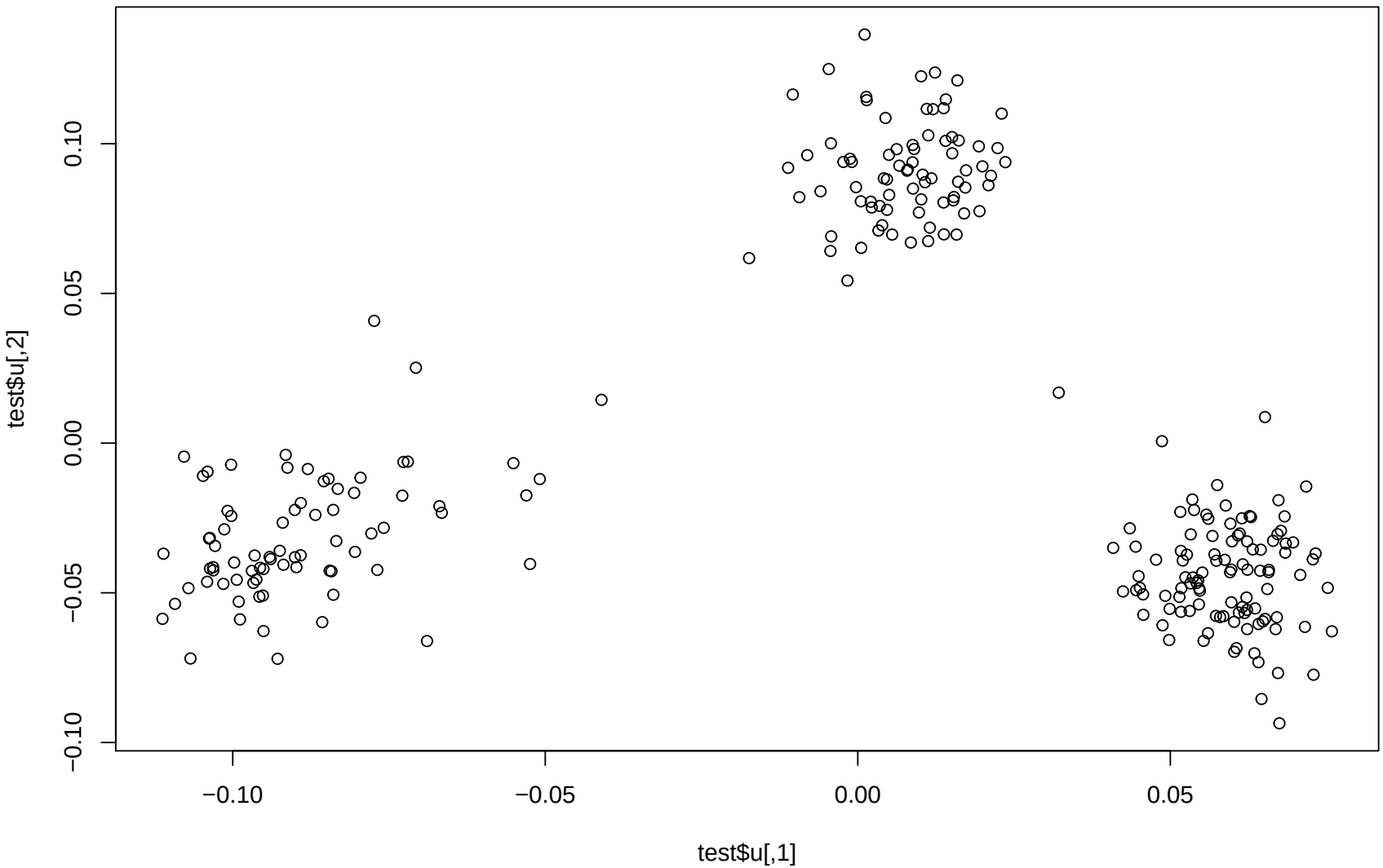
We only use only half of the data

```
ind <- sort(sample(n, n/2))
X.select <- X[ind,]
X.select <- as_FBM(X.select)
test <- big_SVD(X.select, fun.scaling = big_scale())#,
                    #ind.row = ind)
```

Projection

we project the sample on a lower dimensional space to see any structure

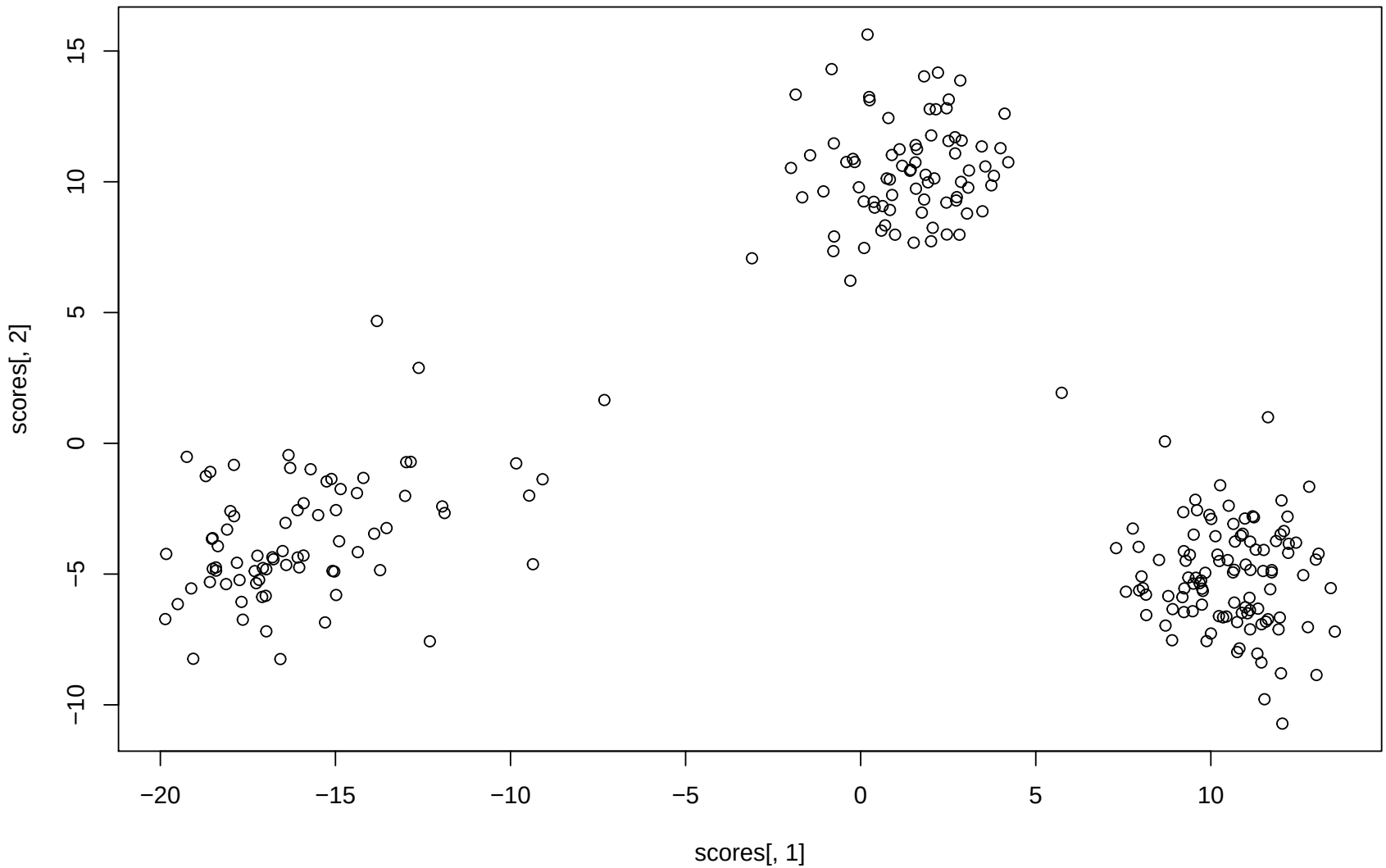
```
plot(test$u)
```



Projection

- A more realistic projection based on the scores

```
scores <- test$u %*% diag(test$d)
plot(scores[,2]~scores[,1])
```



classical pca

using the classical pca from R

```
pca <- prcomp(X[ind, ], center = TRUE, scale. = TRUE)
# same scaling
all.equal(test$center, pca$center)
```

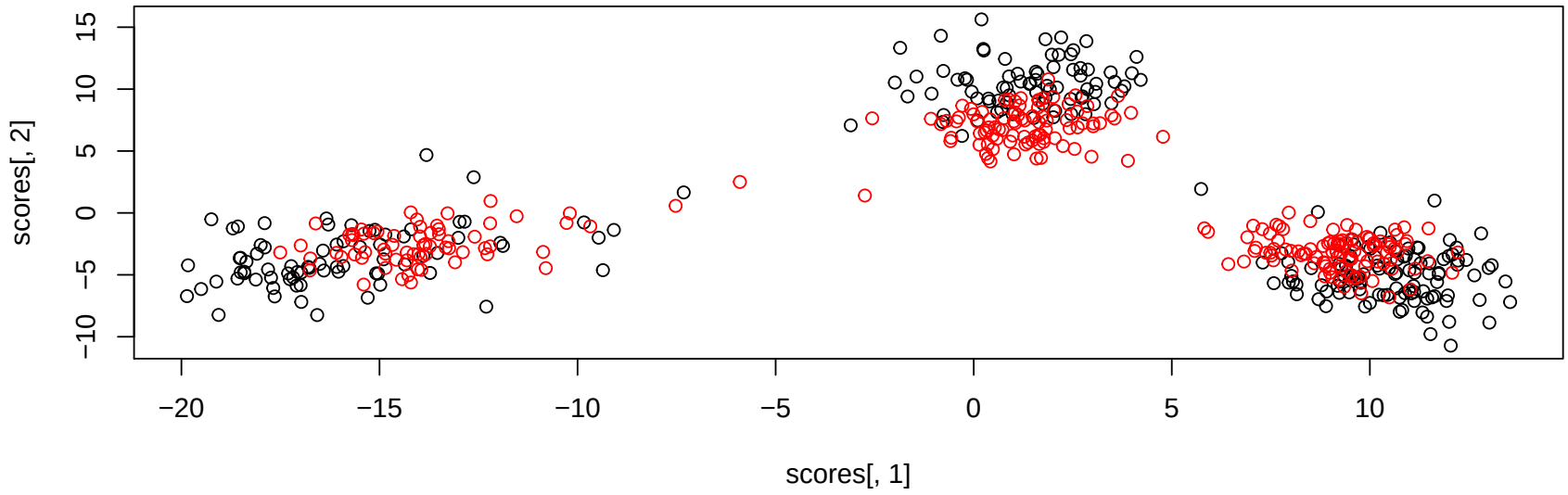
```
[1] TRUE
```

```
all.equal(test$scale,  pca$scale)
```

```
[1] TRUE
```

projecting on new data

```
# projecting on new data
ind2 <- setdiff(rows_along(X), ind)
scores.test2 <- predict(test, X, ind.row = ind2)
plot(scores[,2]~scores[,1])
points(scores.test2[,2]~scores.test2[,1],col="red")
```



PCs as covariates

Using top PCs as covariates corrects for stratification in GWAS. For example, for case control studies one can use the following model to detect variants which are related to a disease:

$$\text{logit}(P(Y = 1|SNP_j, PC_1, \ldots, PC_{20})) = \beta_0 + \beta_1 SNP_j + \gamma_1 PC_1 + \ldots + \gamma_{20} PC_{20}$$

SVD as a Compression/Dimension Reduction Tool

We start by reading an image and we perform SVDs on this image.

```
if (!"jpeg" %in% installed.packages())
  install.packages("jpeg")
# Read image file into an array with three channels
# (Red-Green-Blue, RGB)
liquet <- jpeg::readJPEG("liquet.jpeg")
r <- liquet[, , 1] ; g <- liquet[, , 2] ; b <- liquet[, , 3]
# Performs full SVD of each channel
liquet.r.svd <- svd(r) ; liquet.g.svd <- svd(g) ;
liquet.b.svd <- svd(b)
rgb.svds <- list(liquet.r.svd, liquet.g.svd, liquet.b.svd)
```

SVD as a Compression/Dimension Reduction Tool

These two functions will be needed to display an image stored in an RGB array:

```
# Function to display an image stored in an RGB array

plot.image <- function(pic, main = "") {
 h <- dim(pic)[1] ; w <- dim(pic)[2]
 plot(x = c(0, h), y = c(0, w), type = "n", xlab = "",
      ylab = "", main = main)
 rasterImage(pic, 0, 0, h, w)
}
```

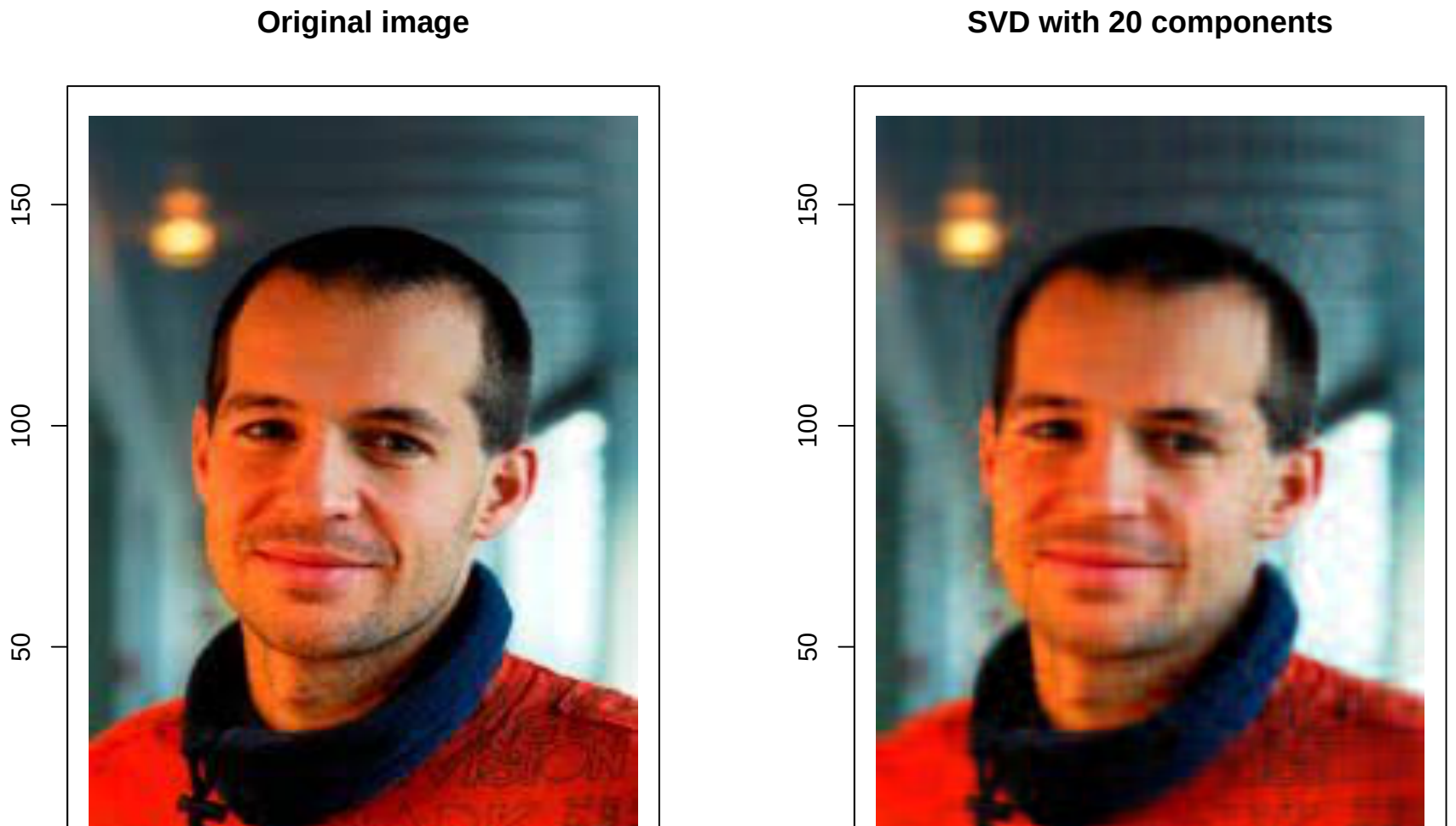
Function to compress an image via SVD of each channel

```
compress.image <- function(rgb.svds, nb.comp) {
 # nb.comp (number of components) should be less than min(d
 # i.e., 170 here
 svd.lower.dim <- lapply(rgb.svds, function(i)
   list(d = i$d[1:nb.comp],
   u = i$u[, 1:nb.comp],
   v = i$v[, 1:nb.comp]))

 img <- sapply(svd.lower.dim, function(i) {
    img.compressed <- i$u %*% diag(i$d) %*% t(i$v)
  }, simplify = 'array')
 img[img < 0] <- 0
 img[img > 1] <- 1
 return(list(img = img, svd.reduced = svd.lower.dim))
}
```

plot side-by-side the original and compressed images now

```
par(mfrow = c(1, 2))
plot.image(liquet, "Original image")
p <- 20 ; plot.image(compress.image(rgb.svds, p)$img,
                     paste("SVD with", p, "components"))
```



compression ?

As you can see, with 20 components (over 170 maximum), we can still recognize Benoit!

How much compression did we achieve with 20 components?

```
object.size(rgb.svds) # Original image
```

```
1740920 bytes
```

```
object.size(compress.image(rgb.svds, p)$svd.reduced)
```

```
207320 bytes
```

```
# Compressed image
```

Case Study: The `liver.toxicity` study

First, you have to install the R package `mixOmics` which is now availaible on Bioconductor

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("mixOmics", version = "3.8")
```

```
library(mixOmics)
```

Case Study: The `liver.toxicity` study

The `liver.toxicity` is a list in the package that contains:

- `gene`: a data frame with 64 rows and 3116 columns, corresponding to the expression levels of 3,116 genes measured on 64 rats.
- `clinic`: a data frame with 64 rows and 10 columns, corresponding to the measurements of 10 clinical variables on the same 64 rats.
- `treatment`: data frame with 64 rows and 4 columns, indicating the treatment information of the 64 rats, such as doses of acetaminophen and times of necropsy.
- `gene.ID`: a data frame with 3116 rows and 2 columns, indicating geneBank IDs of the annotated genes.

More details are available at `?liver.toxicity`.

Load the data

We first load the data from the package.

```
data(liver.toxicity)
X <- liver.toxicity$gene
```

Quick start

```
MyResult.pca <- pca(X)      # 1 Run the method
plotIndiv(MyResult.pca)     # 2 Plot the samples
```

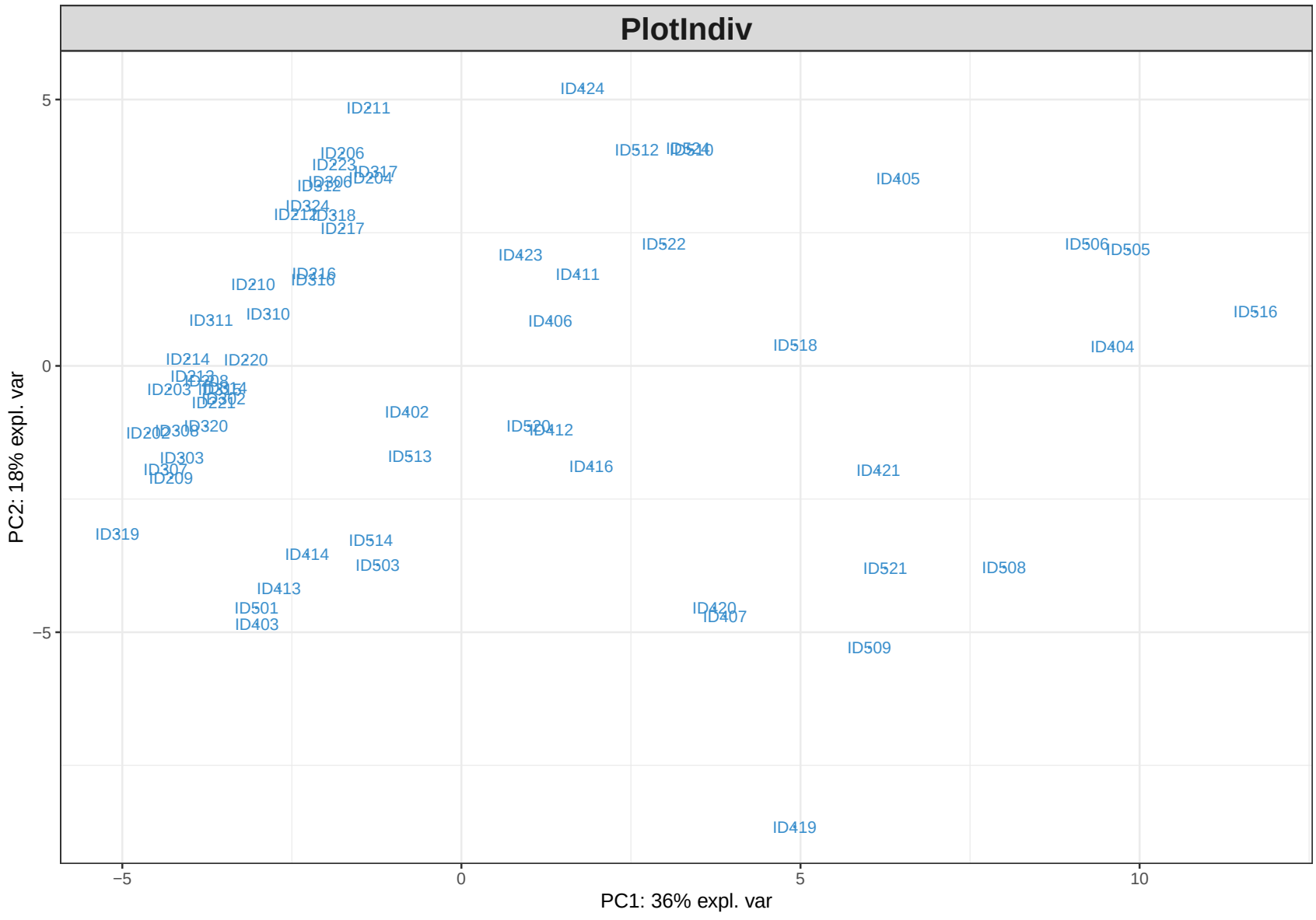
```
plotVar(MyResult.pca)       # 3 Plot the variables
```

If you were to run `pca` with this minimal code, you would be using the following default values:

- `ncomp =2`: the first two principal components are calculated and are used for graphical outputs;
- `center = TRUE`: data are centred (mean = 0)
- `scale = FALSE`: data are not scaled. If `scale = TRUE` standardizes each variable (variance = 1).

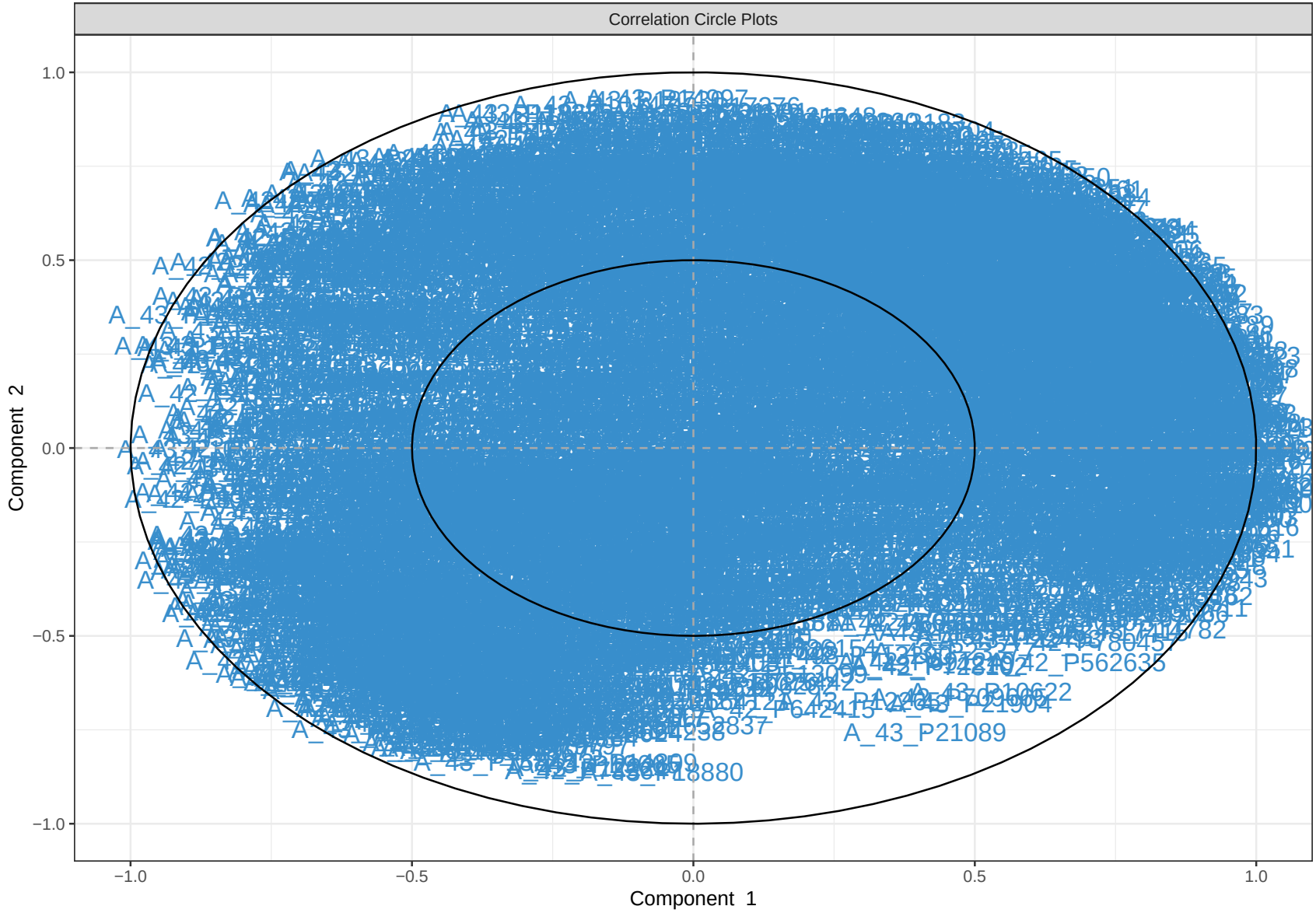
Plot the samples

```
plotIndiv(MyResult.pca)
```



Plot the variables

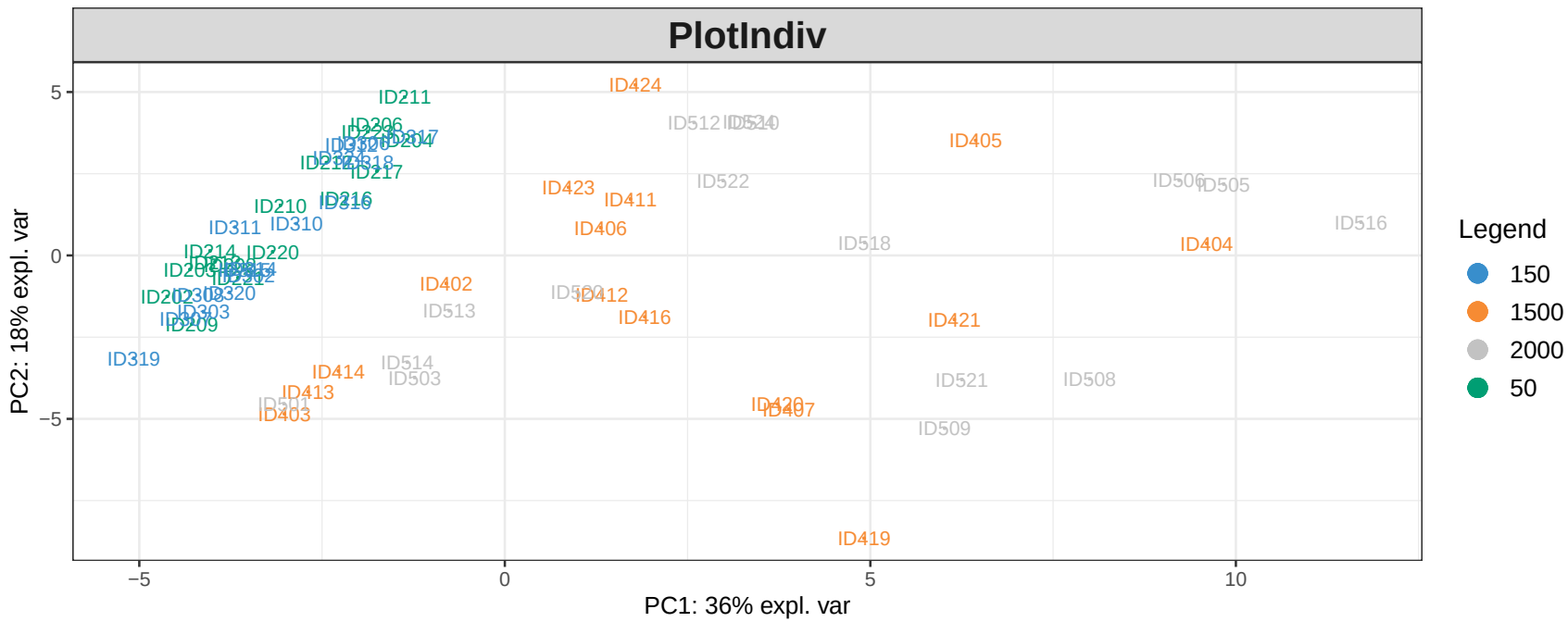
```
plotVar(MyResult.pca)
```



Customize plots

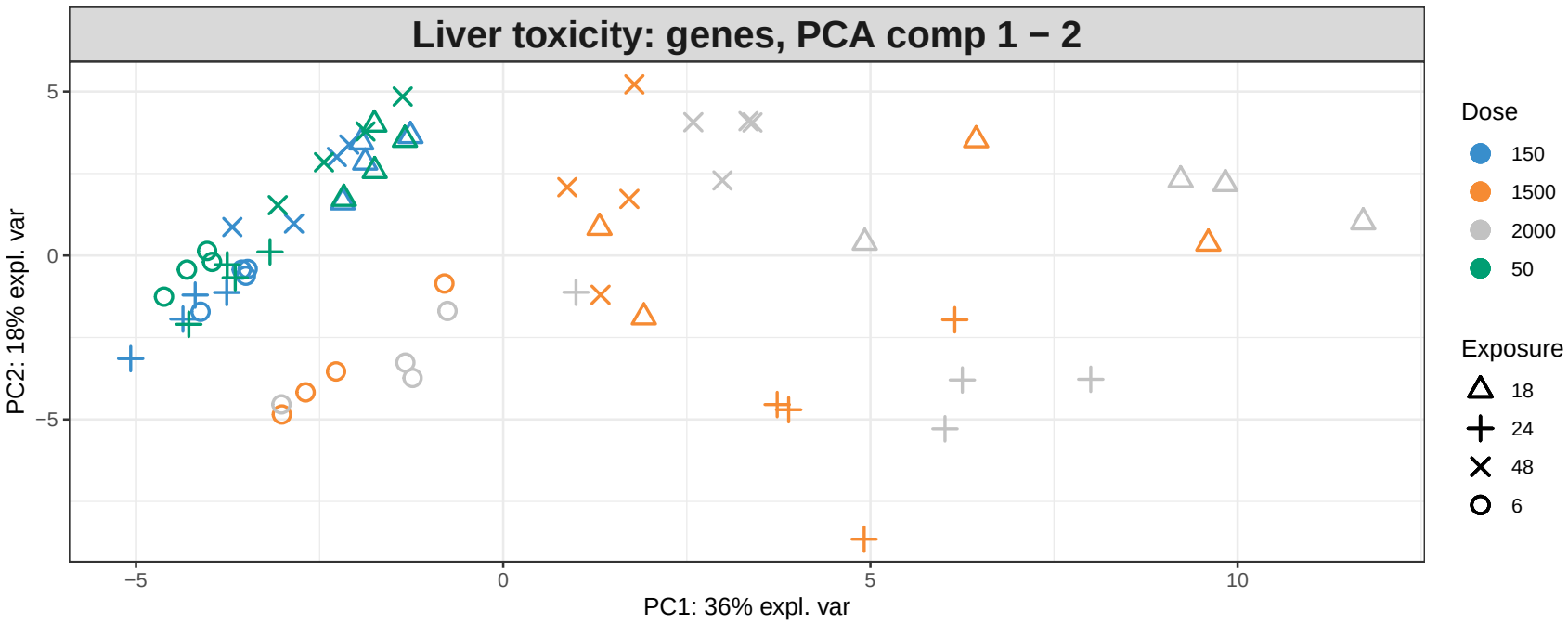
Here is an example where we include the sample groups information with the argument `group`:

```
plotIndiv(MyResult.pca,
          group = liver.toxicity$treatment$Dose.Group,
          legend = TRUE)
```



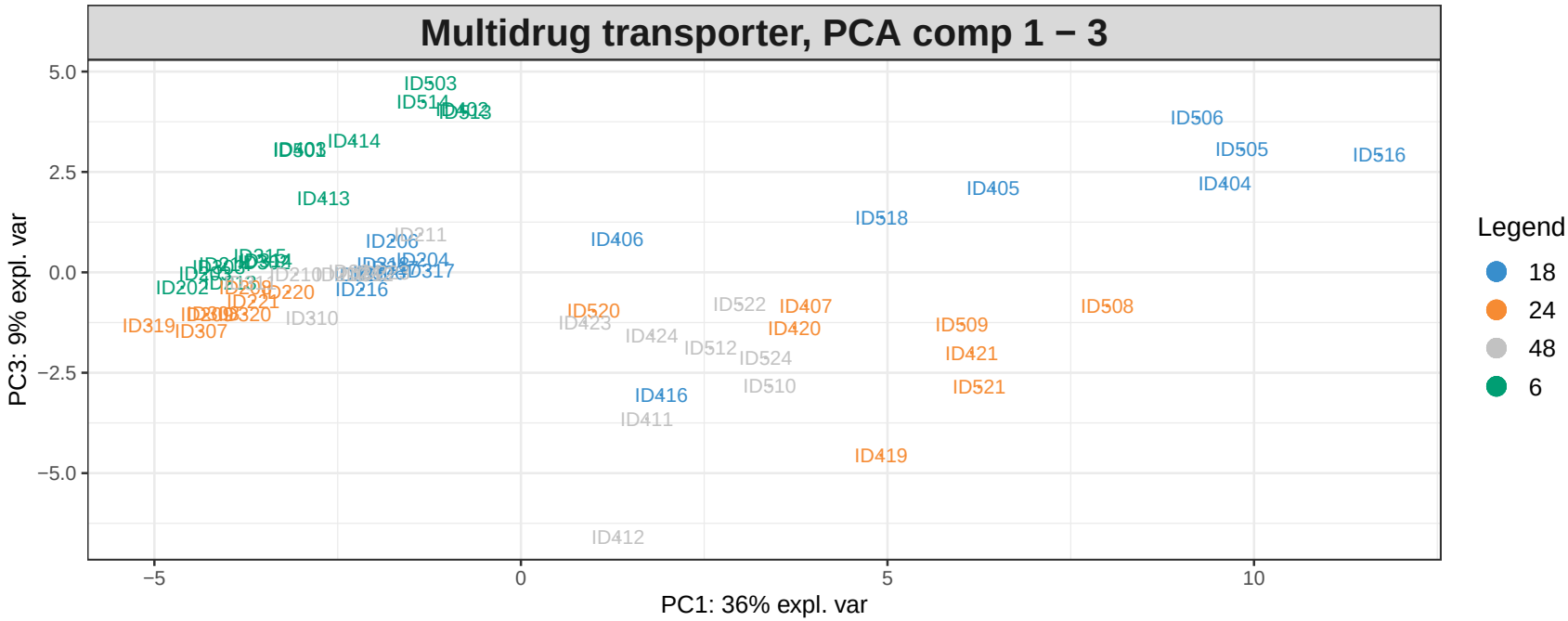
Customize plots: two factors displayed

```
plotIndiv(MyResult.pca, ind.names = FALSE,
          group = liver.toxicity$treatment$Dose.Group,
          pch = as.factor(liver.toxicity$treatment$Time.Group),
          legend = TRUE, title = 'Liver toxicity: genes, PCA comp 1 - 2',
          legend.title = 'Dose', legend.title.pch = 'Exposure')
```



second PCA with 3 components:

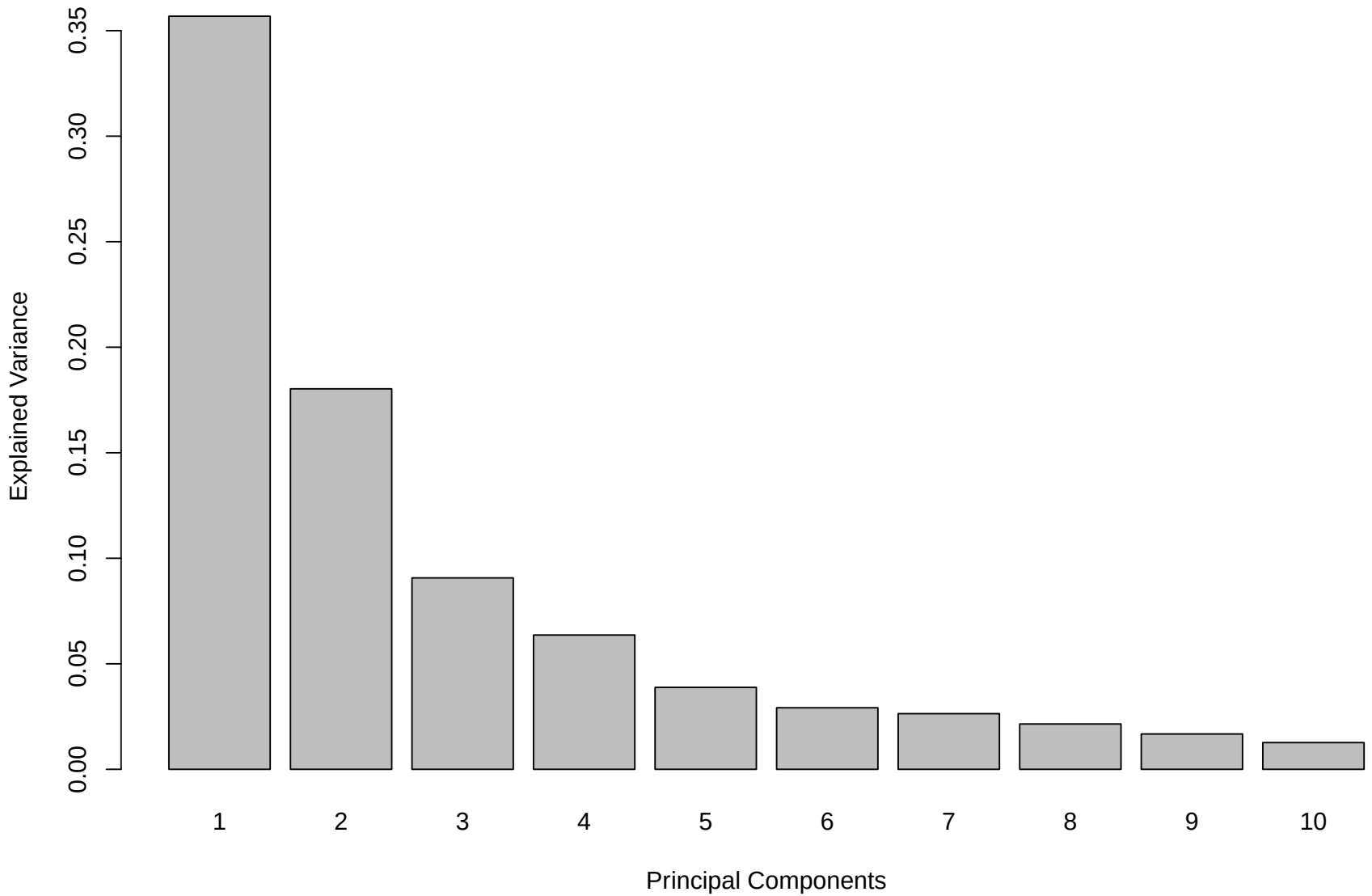
```
MyResult.pca2 <- pca(X, ncomp = 3)
plotIndiv(MyResult.pca2, comp = c(1,3), legend = TRUE,
          group = liver.toxicity$treatment$Time.Group,
          title = 'Multidrug transporter, PCA comp 1 - 3')
```



Here, the 3rd component on the y-axis clearly highlights a time of exposure effect.

Amount of variance explained and choice of number of components

```
MyResult.pca3 <- pca(X, ncomp = 10)
plot(MyResult.pca3)
```



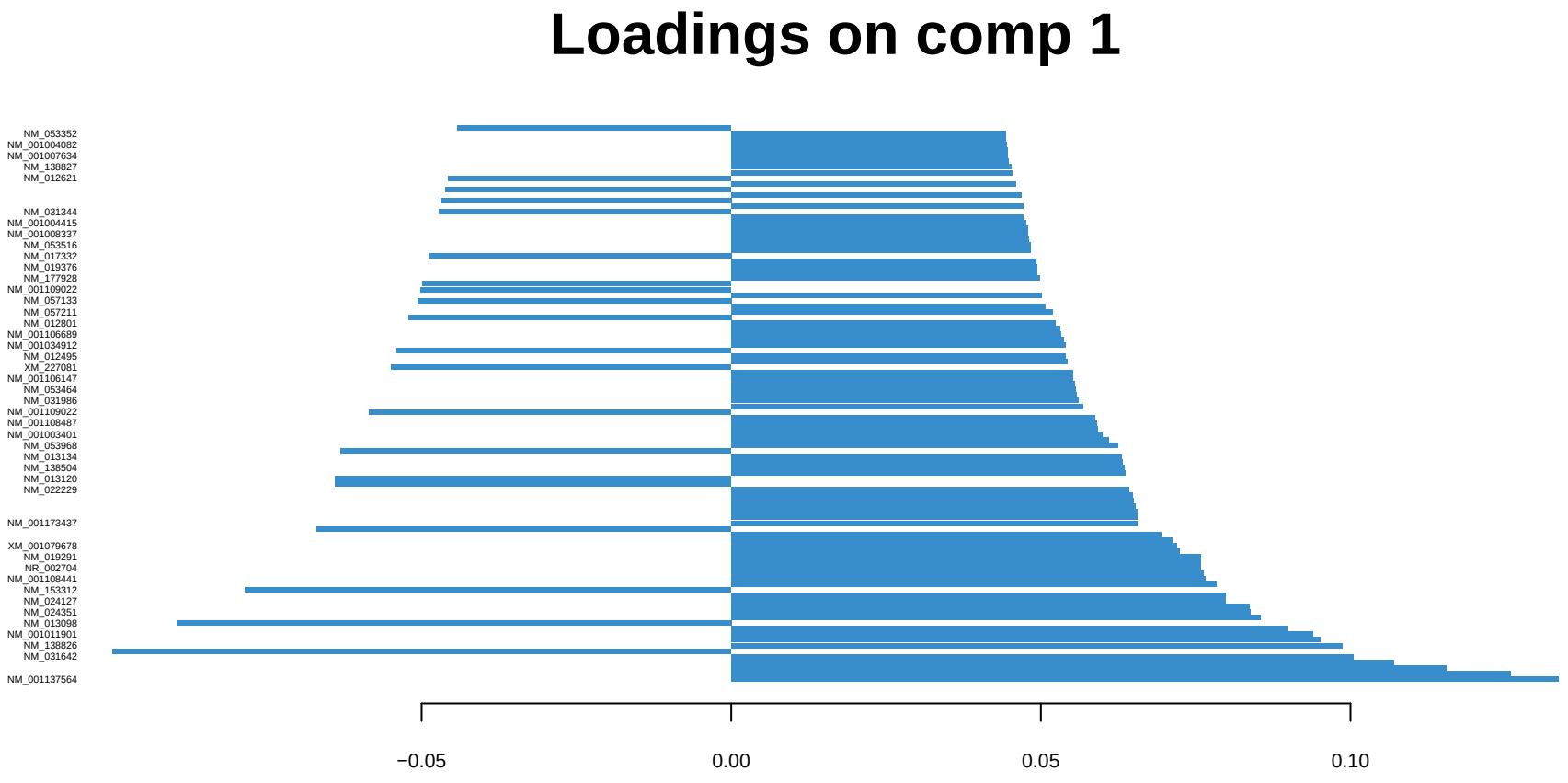
Other useful plots

We can also have a look at the variable coefficients in each component with the loading vectors.

```
# a minimal example
plotLoadings(MyResult.pca)
```

Other useful plots

```
# a customized example to only show the top 100 genes
# and their gene name
plotLoadings(MyResult.pca, ndisplay = 100,
             name.var = liver.toxicity$gene.ID[, "geneBank"],
             size.name = rel(0.3))
```



3 dimensions plots

```
plotIndiv(MyResult.pca2,
          group = liver.toxicity$treatment$Dose.Group,
          style="3d",legend = TRUE,
          title = 'Liver toxicity: genes, PCA comp 1 - 2 -
```

Take Home Message: PCA

- Dimension Reduction approach

- Unsupervised method

- create uncorrelated artificial variables called principal components

- The principal components are obtained so that their variance is maximised

Take Home Message: PCA

- Dimension Reduction approach

- Unsupervised method

- create uncorrelated artificial variables called principal components

- The principal components are obtained so that their variance is maximised

Difficult to interpret PC When there are too many variables.
How to make variable selection with PCA ?