Principal Component Analysis (PCA)

Goal of Integrative Analysis:

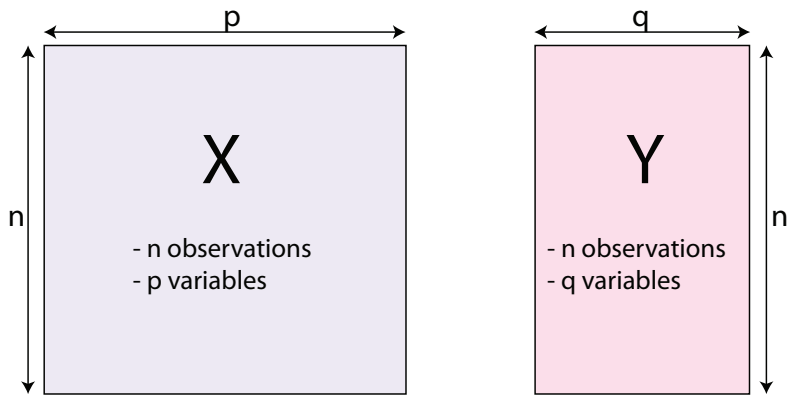
Wikipedia. **Data integration** "involves **combining data** residing in different sources and providing users with a unified view of these data. This process becomes significant in a variety of situations, which include both commercial and **scientific**".

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

Goal. Combine multiple types of data:

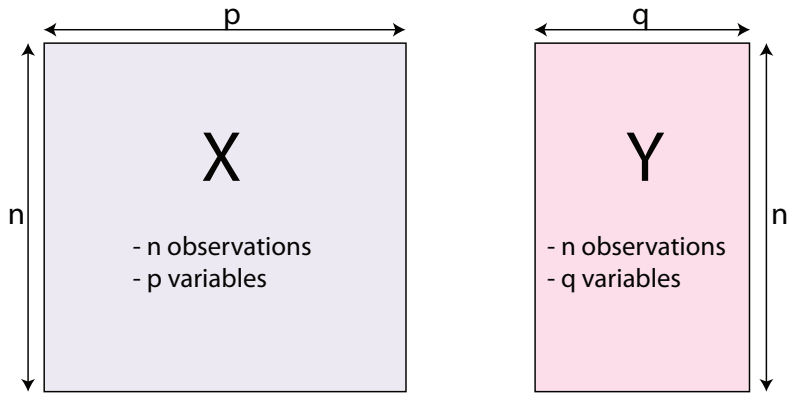
- Contribute to a better understanding of biological mechanism.
- Have the potential to improve the diagnosis and treatments of complex diseases.

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

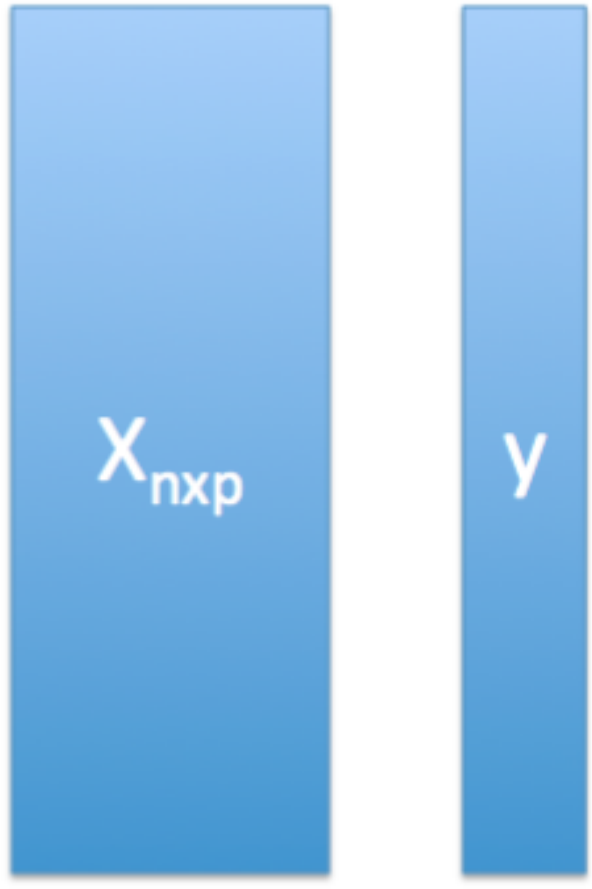
- The $n < p$ situation:
  - More predictors than observations

    $\Rightarrow$ 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.

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.

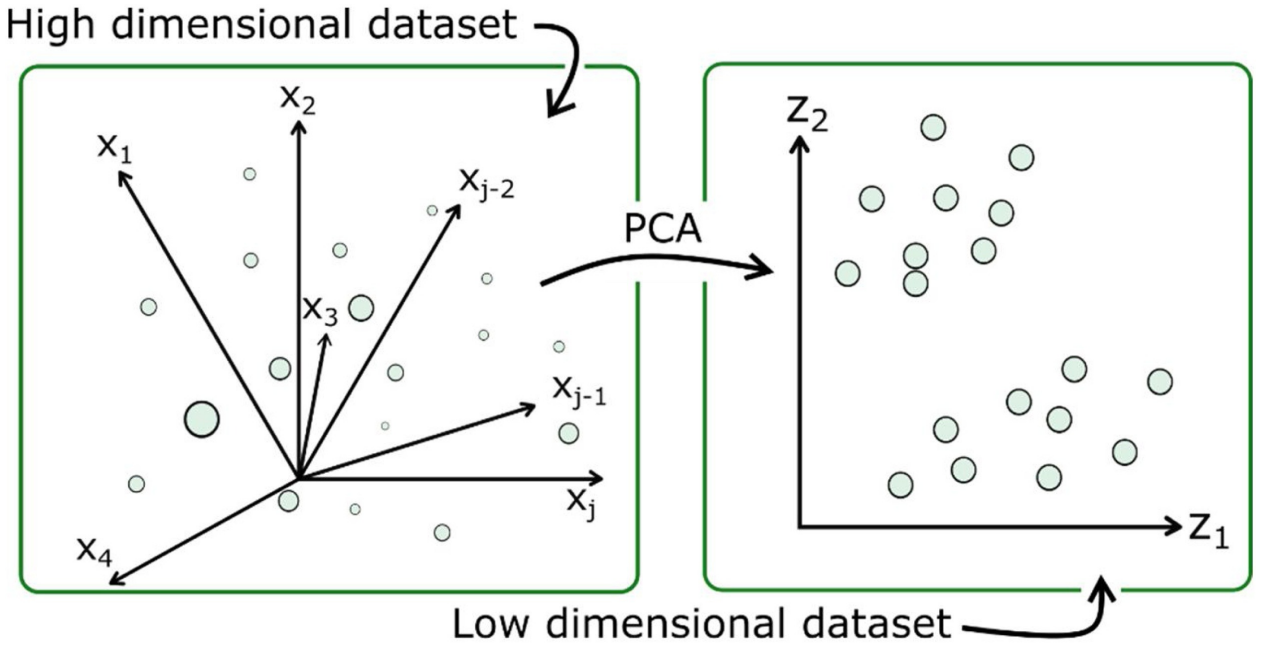
As an example, principal component analysis is commonly performed to account for population stratification in genome wide association study.

Principal Component Analysis

- Data is unlabelled and is denoted via $\mathcal{D} = \{x^{(1)}, \ldots, x^{(n)}\}$ where each sample or observation $x^{(j)}$ is a p dimensional vector of features in Euclidean space.
- Not all p dimensions of the data are equally useful.
- Especially the case in the presence of high dimensional data (large p).
- Many features may be either completely redundant or uninformative.
- These cases are referred to as **correlated features** or **noise features**
- PCA is a well-known and widely used **dimensionality reduction technique** for a wide variety of applications such as *data compression*, *feature extraction*, and *visualization*.

Principal Component Analysis

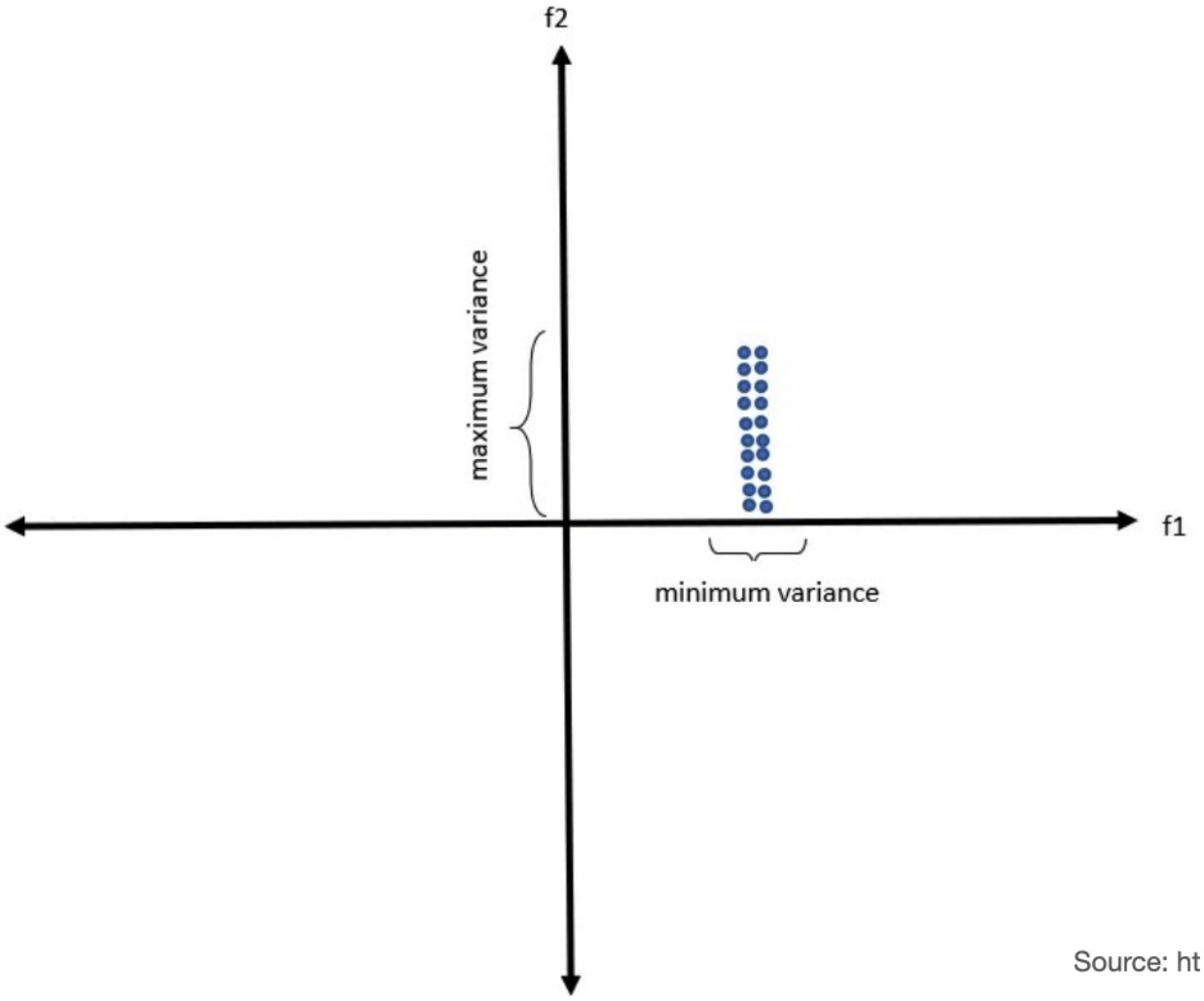
- Basic idea of PCA is to project each point of $\mathcal{D}$ which has many correlated coordinates onto fewer coordinates called Principal components which are uncorrelated.



- This is done while still retaining most of the variability present in the data.
- PCA offers a low-dimensional representation of the features that attempts to capture the most important information from the data.

Principle of PCA

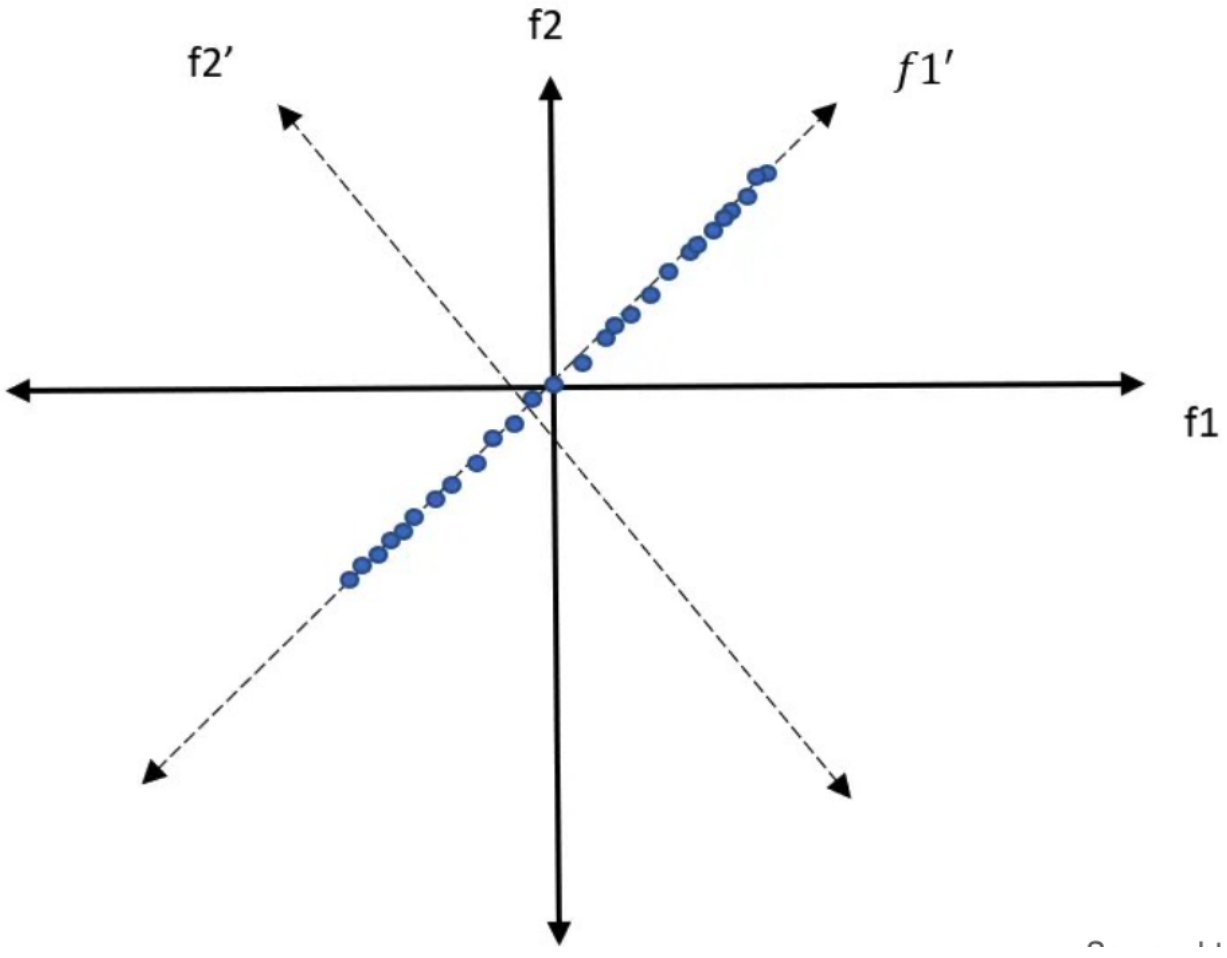
Suppose we have set of data points $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^2$ denoted as blue dots.



Source: ht

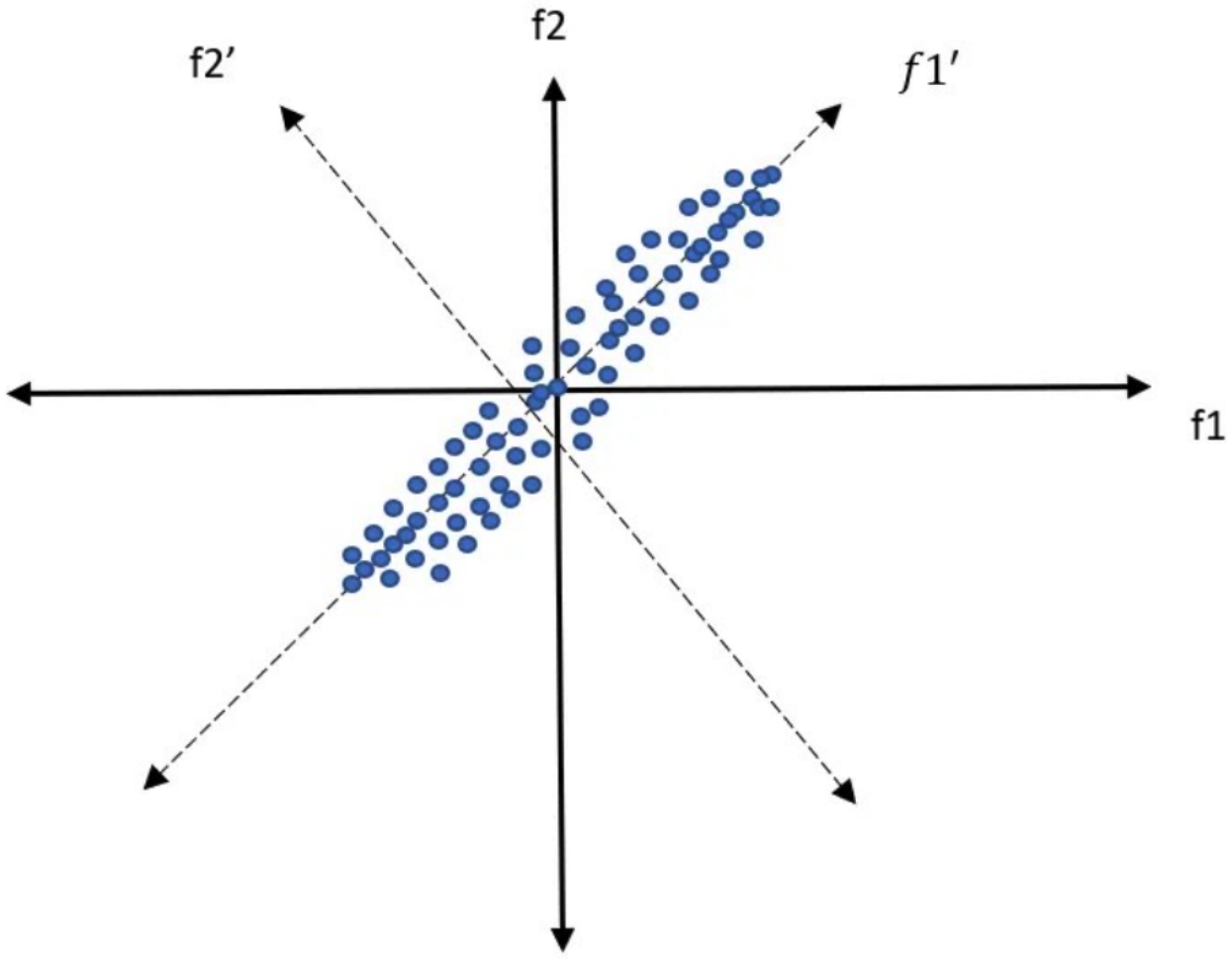
Principle of PCA

Suppose we have set of data points $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^2$ denoted as blue dots.



Principle of PCA

Suppose we have set of data points $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^2$ denoted as blue dots.



Principle of PCA

Suppose we have set of data points $x^{(1)}, x^{(2)}, \ldots, x^{(n)} \in \mathbb{R}^2$ denoted as blue dots.

Principle of PCA

Input data for PCA

- PCA uses the de-meaned data from the centered data matrix X

$$X = \begin{bmatrix} | & & | \\ x_{(1)} & \dots & x_{(p)} \\ | & & | \end{bmatrix} \quad \text{with} \quad x_{(i)} = \begin{bmatrix} x_i^{(1)} \\ \vdots \\ x_i^{(n)} \end{bmatrix}$$

- PCA uses a linear combination of these columns to arrive at the vectors of the new features $\widetilde{x}_{(1)}, \dots, \widetilde{x}_{(m)}$ called

$$\widetilde{x}_{(i)} = v_{i,1} \begin{bmatrix} | \\ x_{(1)} \\ | \end{bmatrix} + v_{i,2} \begin{bmatrix} | \\ x_{(2)} \\ | \end{bmatrix} + \dots + v_{i,p} \begin{bmatrix} | \\ x_{(p)} \\ | \end{bmatrix} \quad \text{for} \quad i = 1, \dots, m,$$

How it works ?

- The first principal component (PC) can be defined as a direction that maximizes the variance of the projected data.
- The subsequent PC can be taken as a direction orthogonal to the first PC that maximize the variance of the projected data.
- PCs are weighted, linear combinations of the original variables
  - Weights reveal how different variables are loaded into the PCs
- Goal: We want a small number of PCs to explain most of the information / variance in the data

First principal component:

$$\widetilde{x}_{(1)} = v_{1,1}x_{(1)} + v_{2,1}x_{(2)} + \cdots + v_{p,1}x_{(p)}$$

- $v_{i,1}$ are the weights indicating the contributions of each variable $i \in 1, \ldots, p$
  - Weights are normalized $\sum_{i=1}^{p} v_{i1}^2 = 1$
  - $v_1 = (v_{i,1}, v_{i,1}, \ldots, v_{i,1})$ is the loading vector for PC1
  - $\widetilde{x}_{(1)}$ is a linear combination of the p variables that has the largest variance

Second principal component

$$\widetilde{x}_{(2)} = v_{1,2}x_{(1)} + v_{2,2}x_{(2)} + \cdots + v_{p,2}x_{(p)}$$

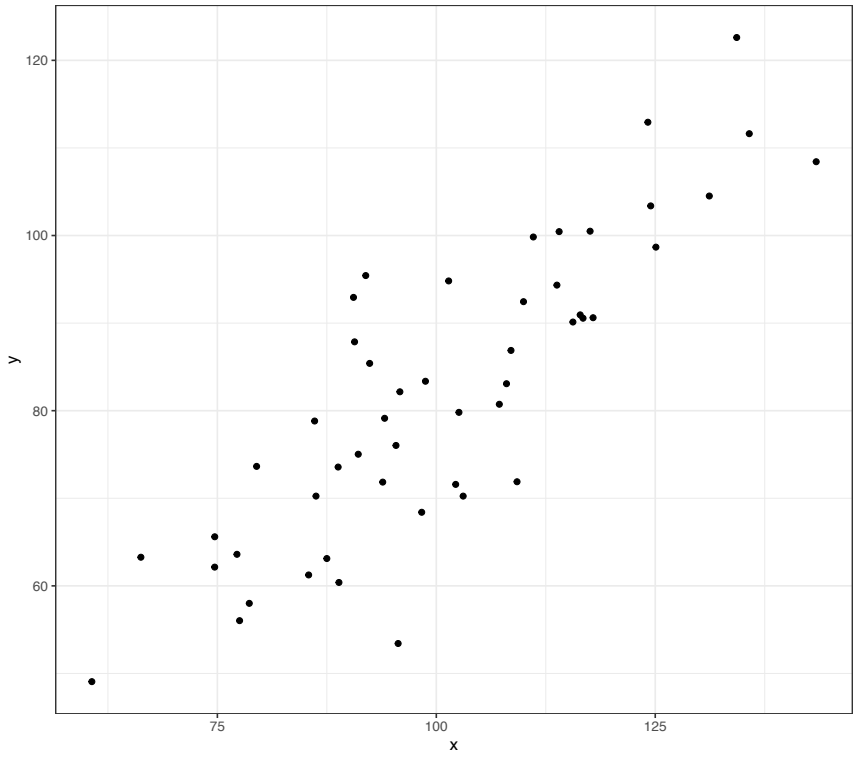
- $v_{i,2}$ are the weights indicating the contributions of each variable $i \in 1, \ldots, p$
  - Weights are normalized $\sum_{i=1}^{p} v_{i2}^2 = 1$
  - $v_2 = (v_{i,2}, v_{i,2}, \ldots, v_{i,2})$ is the loading vector for PC2
  - $\widetilde{x}_{(2)}$ is a linear combination of the p variables that has the largest variance
  - Subject to constraint it is uncorrelated with $\widetilde{x}_{(1)}$
- We repeat this process to create m principal components with $(m \leqslant p)$

Component score, loading vector

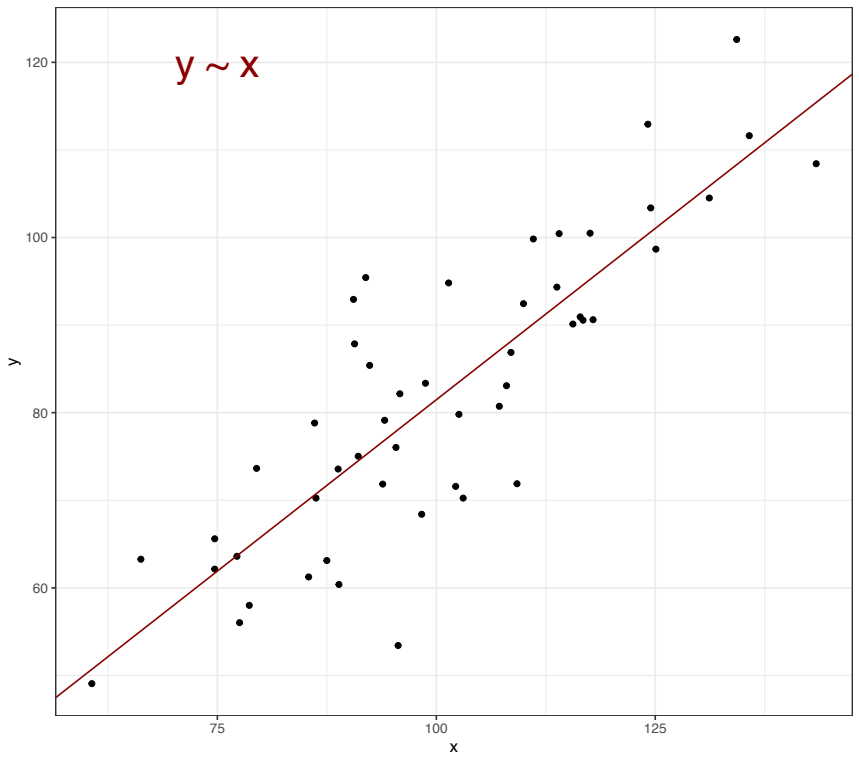
- each new n dimensional vector, $\widetilde{x}_{(i)}$, is a linear combination of the original features.
- $\widetilde{x}_{(i)} = Xv_i$ where $v_i = (v_{i,1}, \ldots, v_{i,p})$ is called the loading vector for the new artificial feature (known as component score) i

$$\underbrace{\begin{bmatrix} | & & | \\ \widetilde{x}_{(1)} & \ldots & \widetilde{x}_{(m)} \\ | & & | \end{bmatrix}}_{\substack{\widetilde{X}_{n\times m} \\ \text{Reduced data}}} = \underbrace{\begin{bmatrix} | & & & | \\ x_{(1)} & \ldots & \ldots & x_{(p)} \\ | & & & | \end{bmatrix}}_{\substack{X_{n\times p} \\ \text{Original de-meaned data}}} \times \underbrace{\begin{bmatrix} | & & | \\ | & & | \\ v_1 & \ldots & v_m \\ | & & | \\ | & & | \end{bmatrix}}_{\substack{\widetilde{V}_{p\times m} \\ \text{Matrix of loading vectors}}}.$$

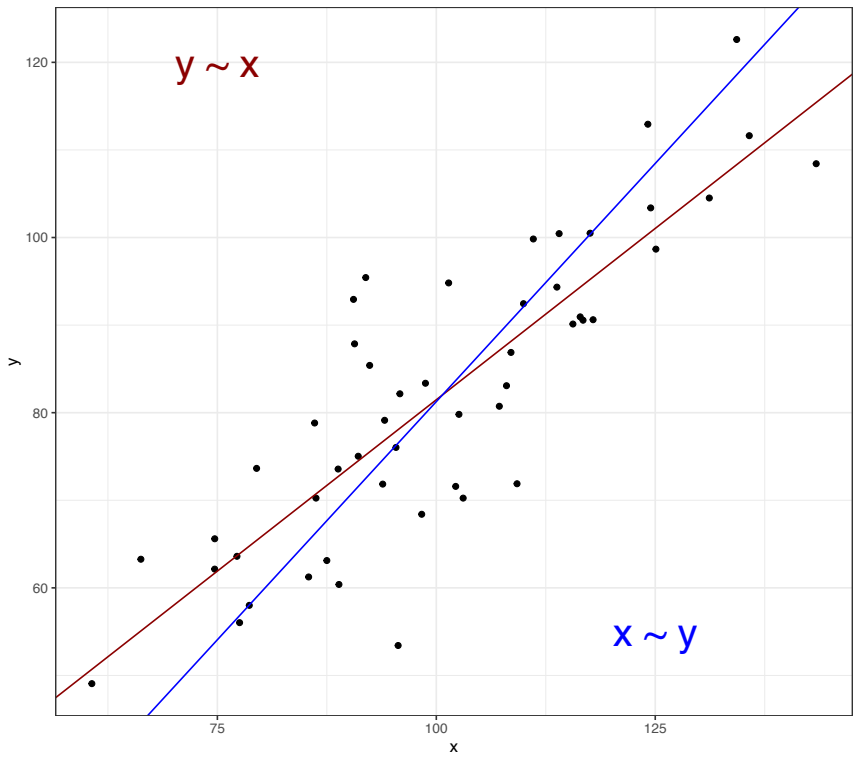
Visualizing PCA in two dimensions



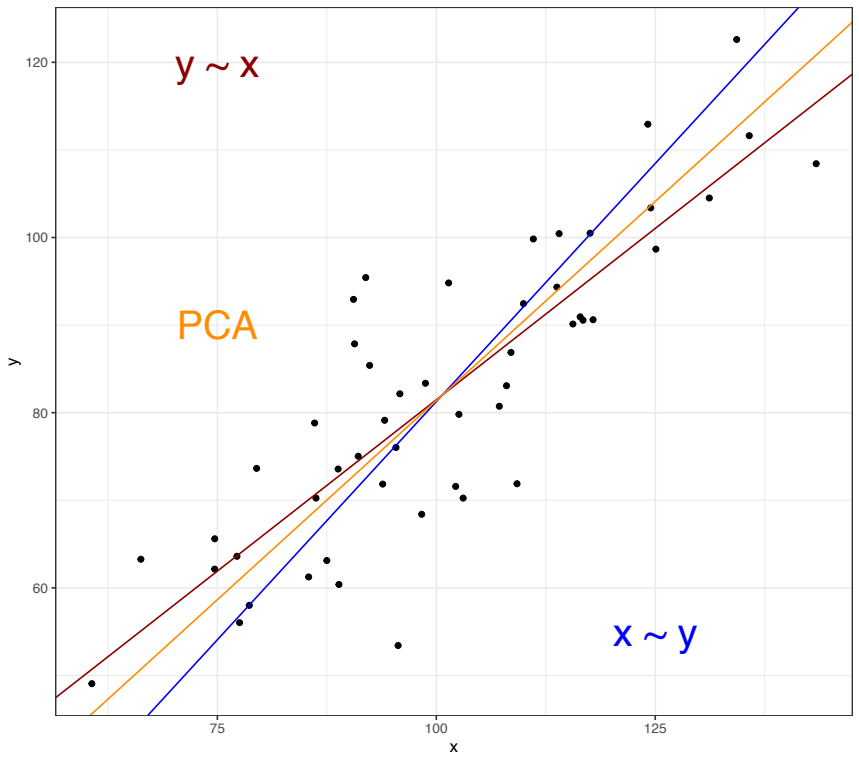
Visualizing PCA in two dimensions



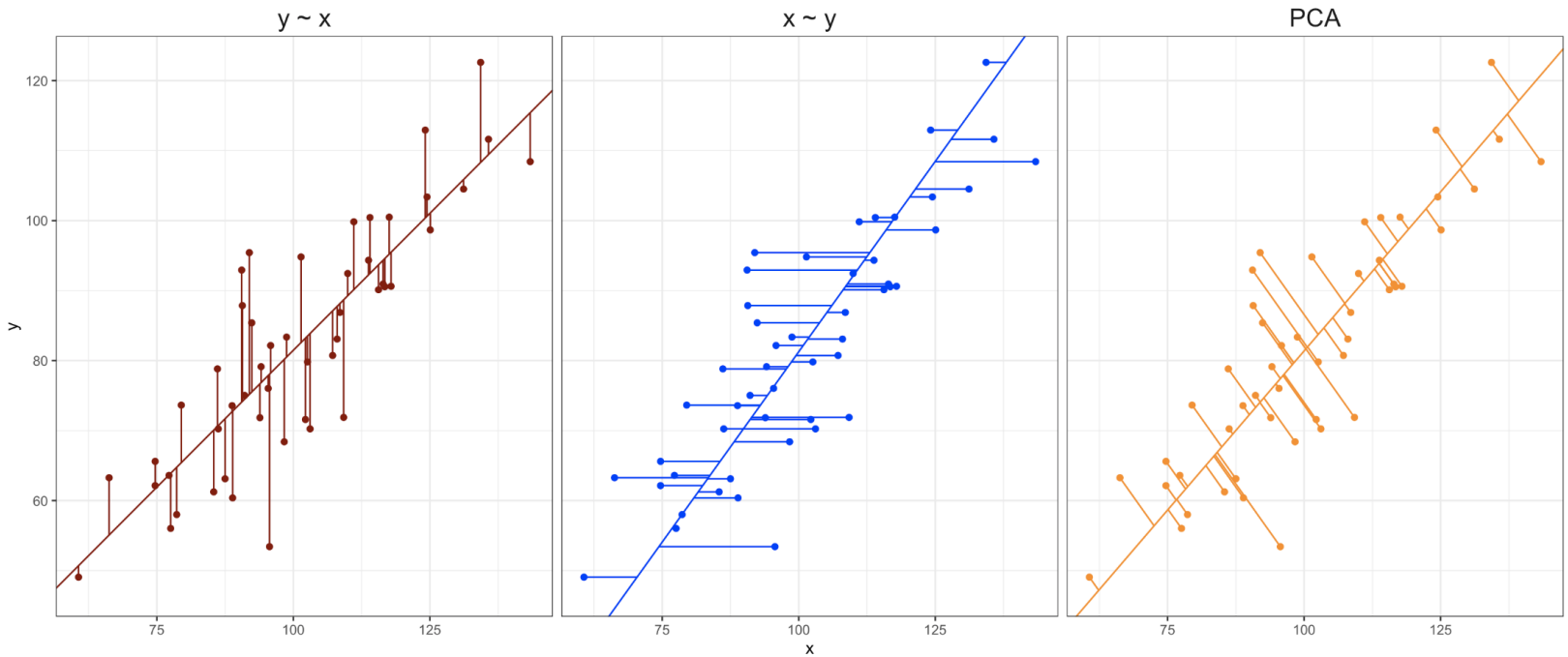
Visualizing PCA in two dimensions



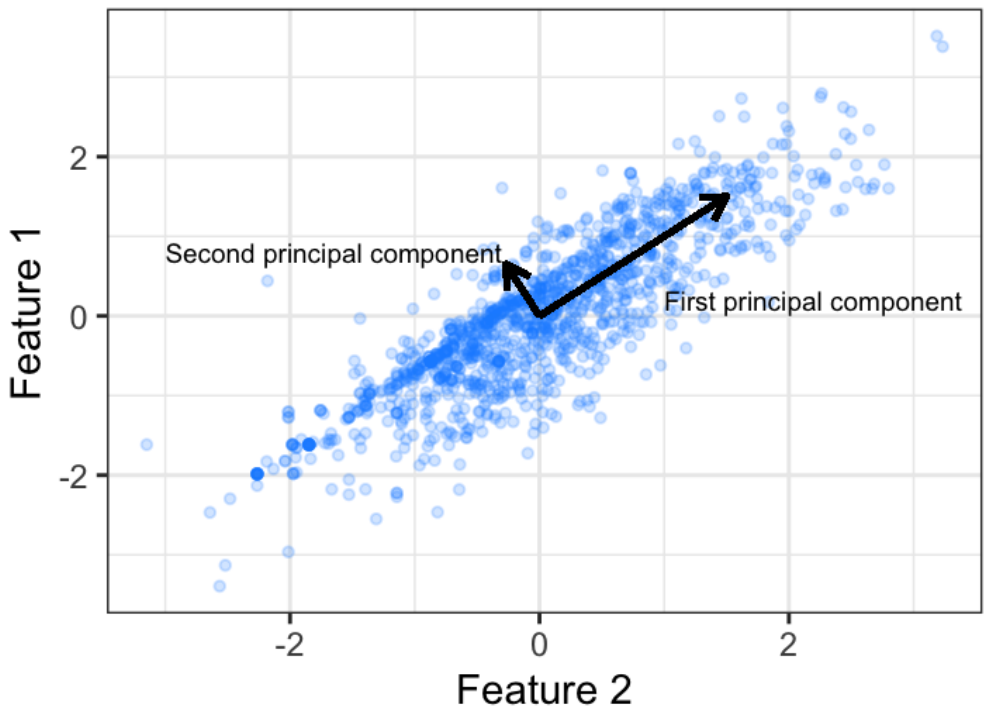
Visualizing PCA in two dimensions



Visualizing PCA in two dimensions



Searching for variance in orthogonal directions



Derivation of PCA

- PCA aims to project the data in the directions with maximum variance.
- Since $\widetilde{x}_{(i)} = Xv_i$ we formulate this by maximizing the sample variance of the components of $\widetilde{x}_{(i)}$.
- $\widetilde{x}_{(i)}$ is a 0 mean vector, its sample variance is $\widetilde{x}_{(i)}^\top \widetilde{x}_{(i)}/n$.
- Thus,

$$\text{Sample variance of component } i = \frac{1}{n} v_i^\top X^\top X v_i = v_i^\top S v_i,$$

  where S is the sample covariance of the data.
- It turns out that the loading vectors $v_1, \ldots, v_m$ is the normed eigenvectors associated with eigenvalues of the sample covariance matrix S
- As S is symmetric and positive semi-definite, the eigenvalues of S are real and non-negative, a fact which allows us to order them via $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_p \geqslant 0$.

- We then pick the loading vector v_i to be a normed eigenvector associated with λ_i, namely,

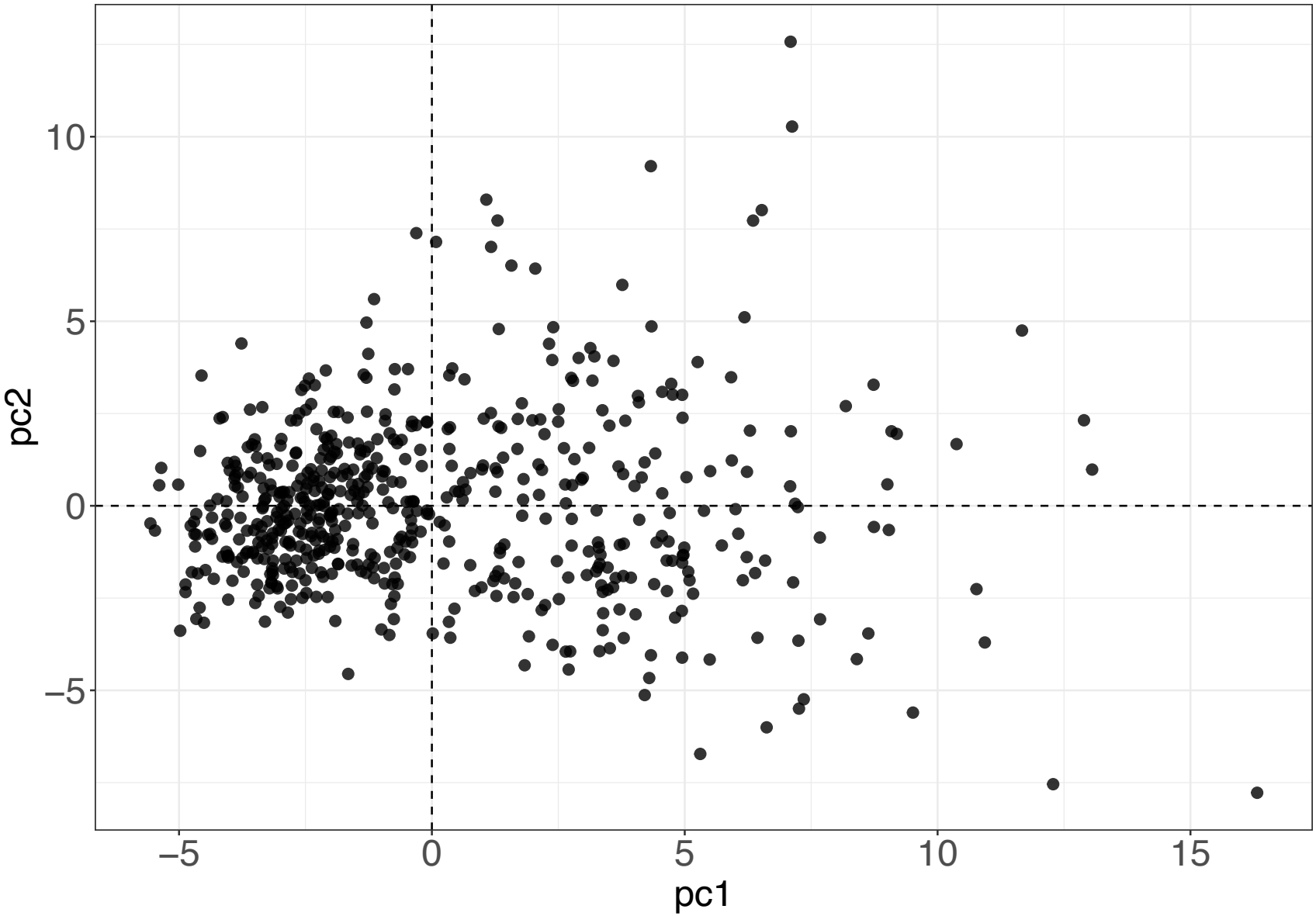
$$Sv_i = \lambda_i v_i,$$

- The first loading vector is associated with the highest eigenvalue; the second is associated with the second highest eigenvalue; and so fourth.
- The symmetry of S means that its eigenvectors are orthogonal and hence $\widetilde{V}$ is a matrix with orthonormal columns.
- Proof in chapter 2 of our textbook `Mathematical Engineering Of Deep Learning`

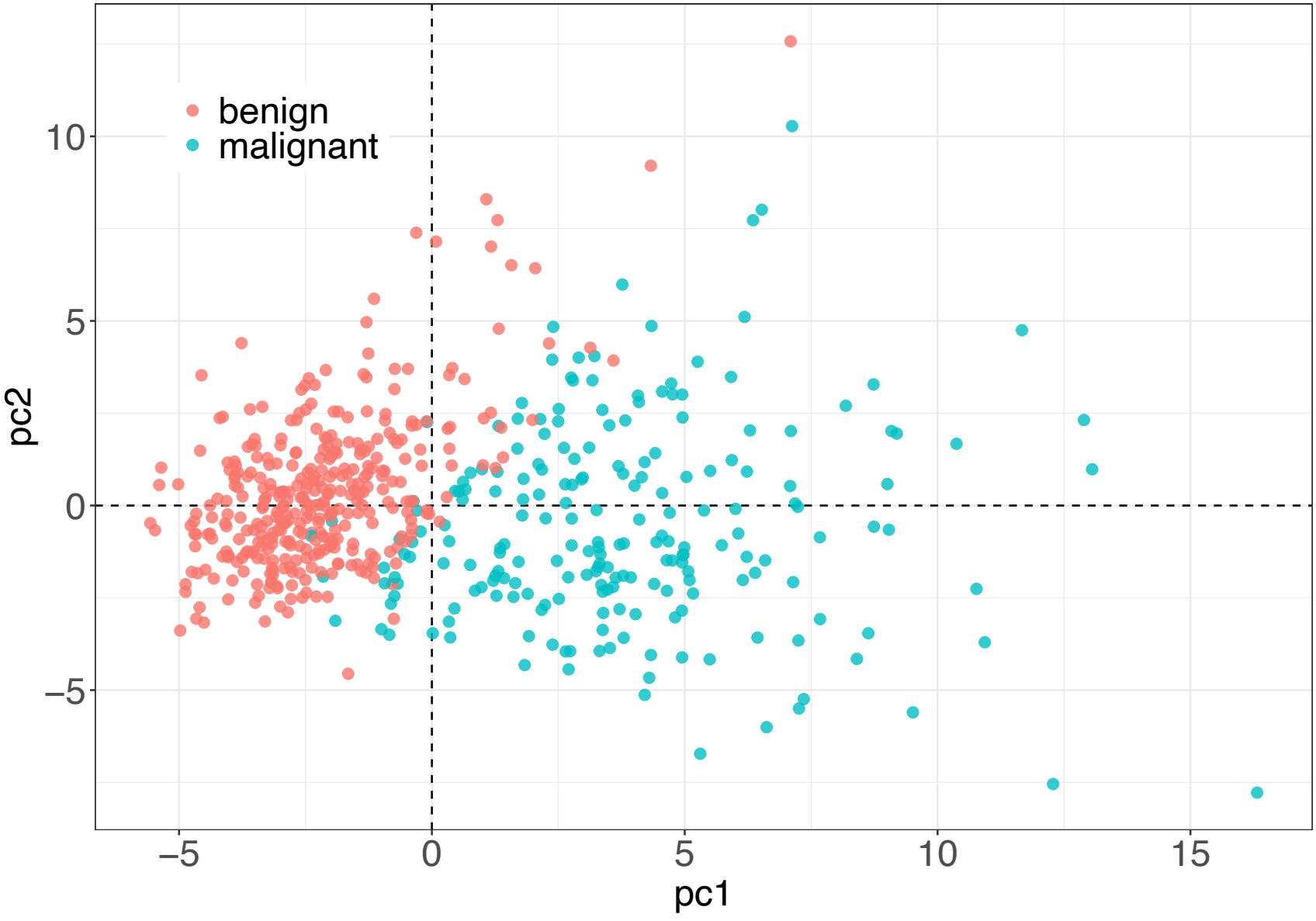
PCA on Wisconsin breast cancer data

- Wisconsin breast cancer data: $p = 30$ and $n = 569$.
- Aim to visualize this data using PCA we set $m = 2$

```
##         id diagnosis radius_mean texture_mean perimeter_mean area_me
## 1   842302         M       17.99        10.38          122.80    1001
## 2   842517         M       20.57        17.77          132.90    1326
## 3 84300903         M       19.69        21.25          130.00    1203
## 4 84348301         M       11.42        20.38           77.58     386
## 5 84358402         M       20.29        14.34          135.10    1297
## 6   843786         M       12.45        15.70           82.57     477
```


10
5
0
−5
pc2
−5
0
5
10
15
pc1

- Color the points based on the labels benign vs. malignant, a useful pattern emerges ?



Variance explained

- Strength of each component: proportion of variance explained (PVE) by each one.
- The total variance present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^{p} \text{Var}\left(x_{(j)}\right) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2,$$

- The variance explained by the *i* th principal component is

$$\text{Var}\left(\widetilde{x}_{(i)}\right) = \frac{1}{n} \sum_{j=1}^{n} \widetilde{x}_{jm}^2.$$

- It can be shown that $\sum_{j=1}^{p} \text{Var}\left(x_{(j)}\right) = \sum_{j=1}^{r} \text{Var}\left(\widetilde{x}_{(j)}\right)$, with $r = \min(n-1, p)$.
- Therefore, the PVE of the *j*-th principal component is given by:

$$\frac{\text{Var}\left(\widetilde{x}_{(i)}\right)}{\text{total variance}}.$$

How many principal components should we use?

- Eigenvalues λ_j for $j \in 1, \ldots, p$ indicate the variance explained by each component
- $PEV_j = \lambda_j / \sum_{i=1}^{r} \lambda_i$ equals proportion of variance explained by PCj
- Can compute the cumulative proportion of variance explained (CVE) with p^* components:

$$\text{CVE}_{p^*} = \frac{\sum_{j=1}^{p*} \lambda_j}{\sum_{i=1}^{m} \lambda_i}$$

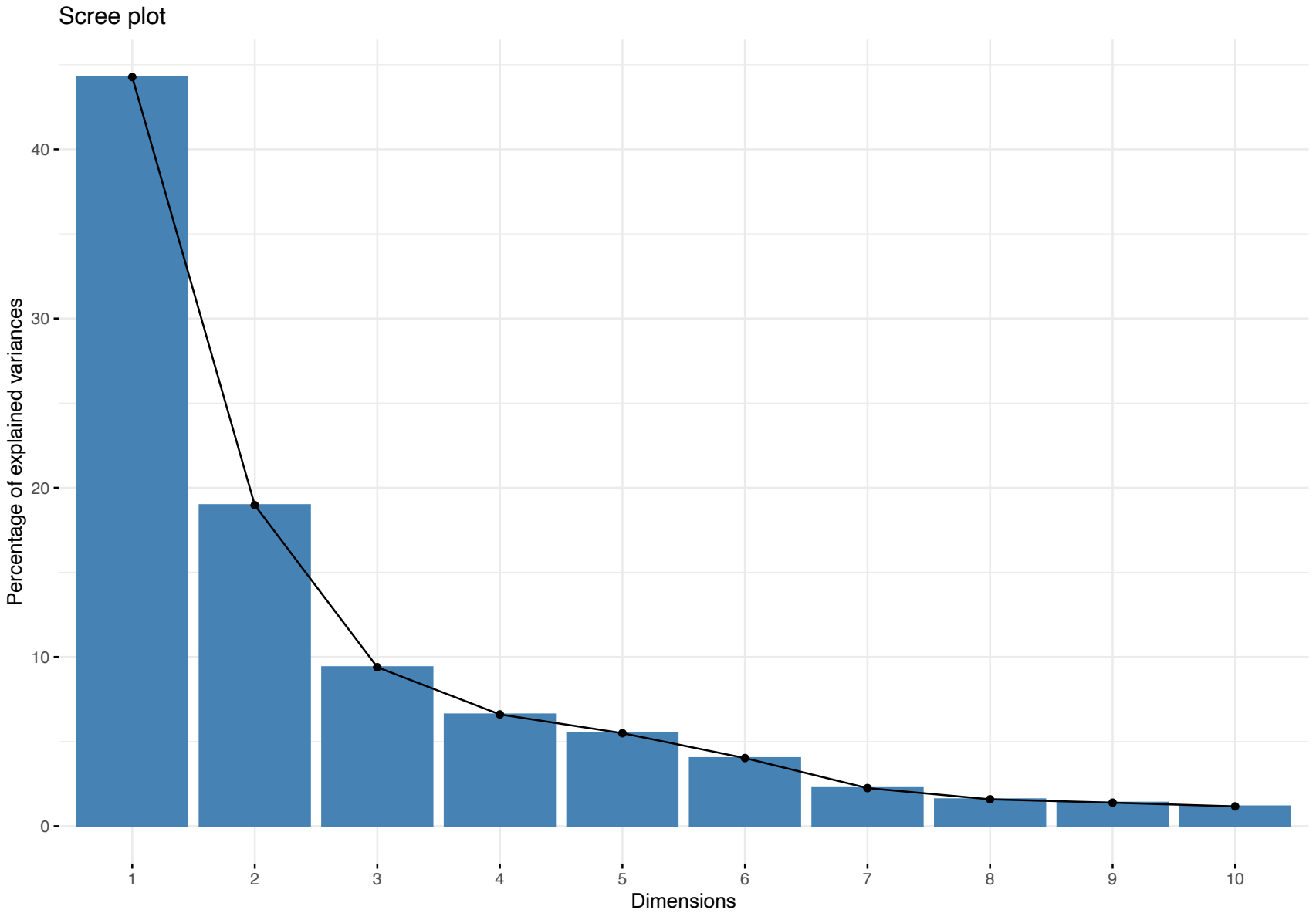
- Can use **scree plot** to plot eigenvalues and guide choice for $p^* < p$ by looking for "elbow'' (rapid to slow change)

PCA on Wisconsin breast cancer data

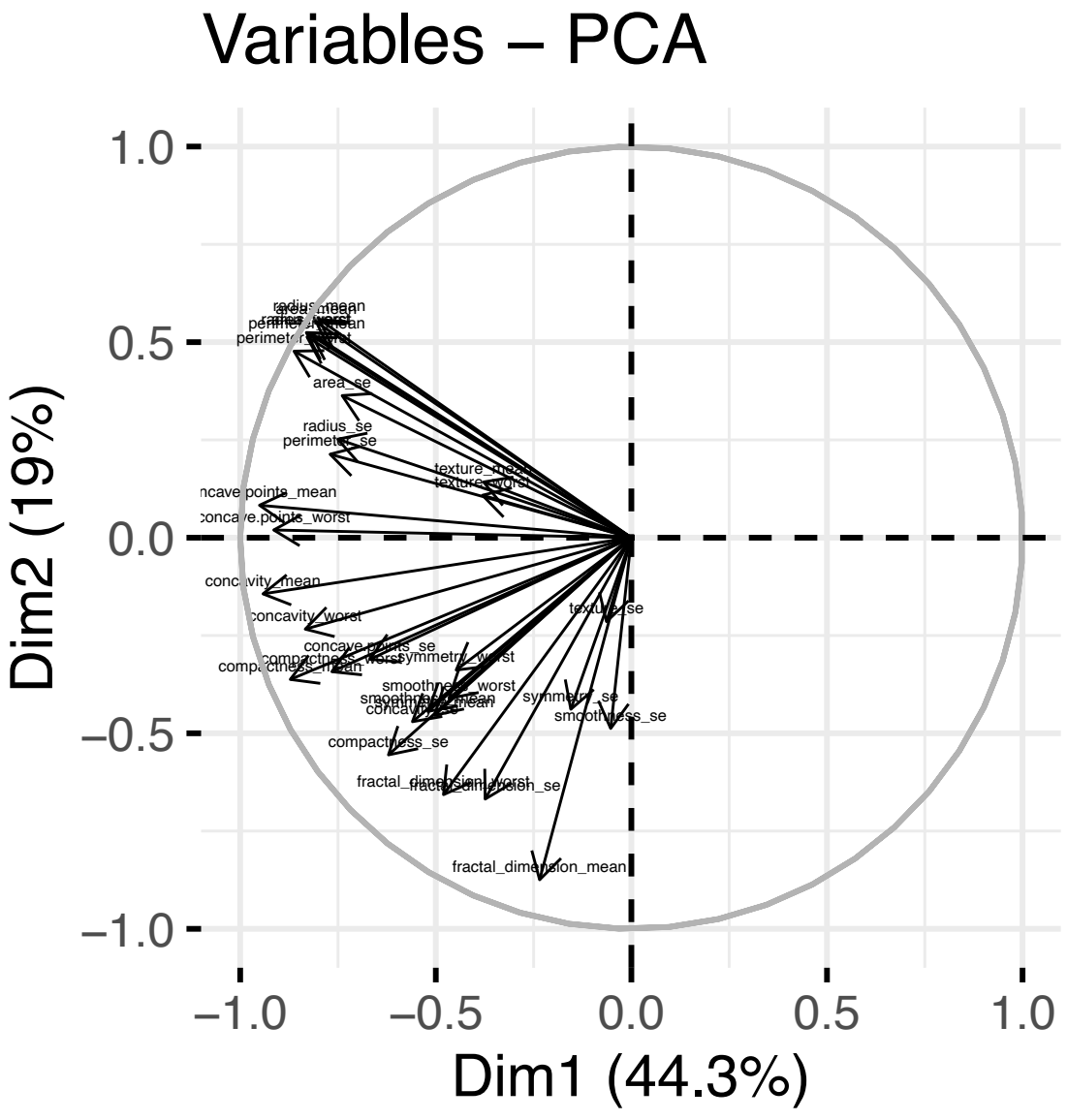
- Variance explained

```
##        Eigenvalue       PEV     CPEV
## comp 1  13.281608 44.272026 44.27203
## comp 2   5.691355 18.971182 63.24321
## comp 3   2.817949  9.393163 72.63637
## comp 4   1.980640  6.602135 79.23851
## comp 5   1.648731  5.495768 84.73427
## comp 6   1.207357  4.024522 88.75880
```

Proportion of variance explained

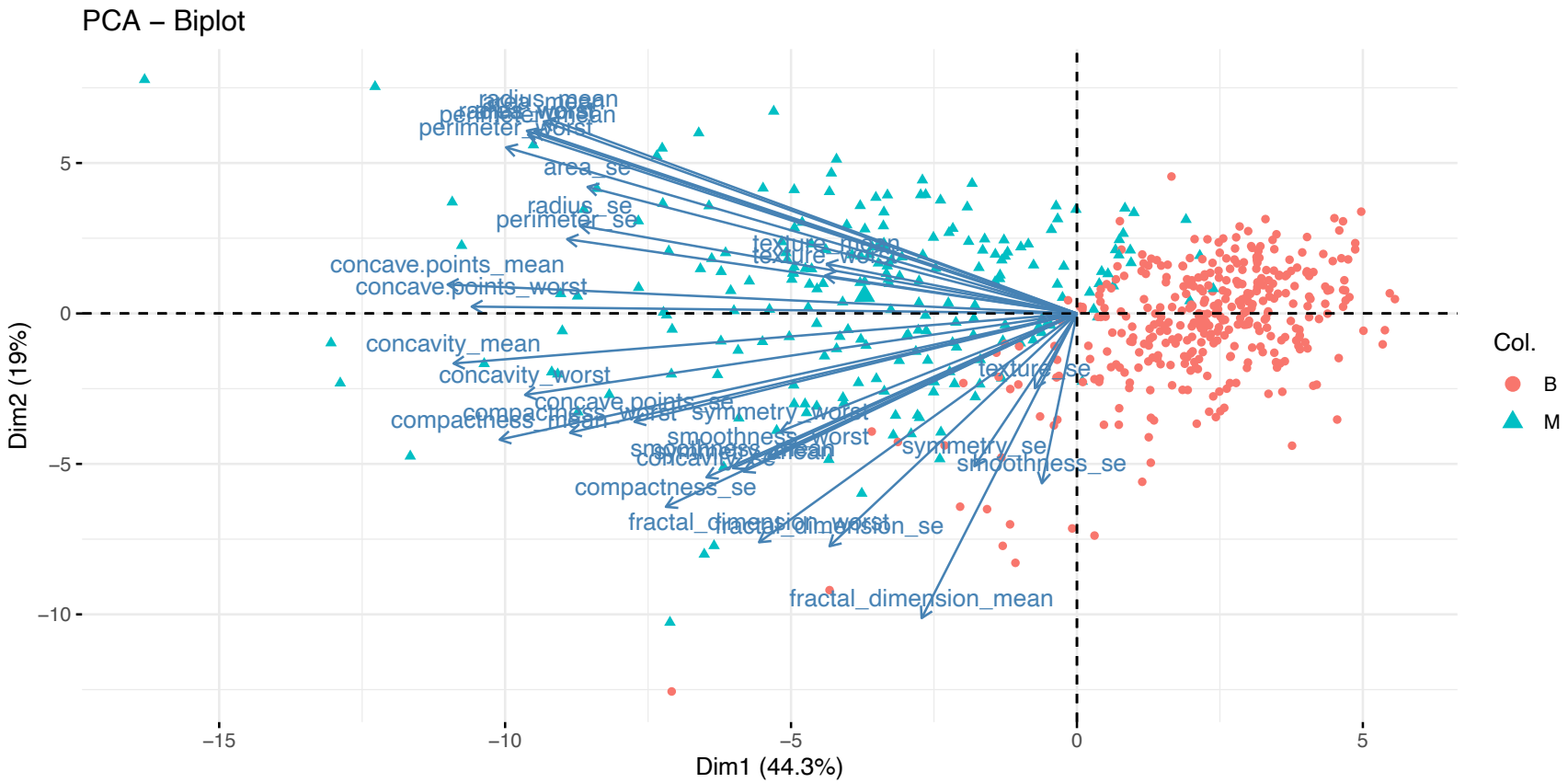


Projection of variables



Biplot: both the space of observations and the space of variables

- Arrows represent the directions of the original variables



PCA Through SVD

- Any $n \times p$ dimensional matrix X of rank r can be represented as

$$X = U\Delta V^\top = \sum_{i=1}^{r} \delta_i \, u_i \, v_i^\top, \text{with} \quad \Delta = \operatorname{diag}(\delta_1, \ldots, \delta_r), \quad \text{and} \quad \delta_i > 0.$$

- $n \times r$ matrix U and the $p \times r$ matrix V are both with orthonormal columns denoted u_i and v_i respectively for $i = 1, \ldots, r$.
- Columns are called the left and right **singular vectors** respectively.
- δ_i in the $r \times r$ diagonal matrix Δ are called **singular values** and are ordered as $\delta_1 \geqslant \delta_2 \geqslant \cdots \geqslant \delta_r > 0$.

- SVD representation of the sample covariance:

$$S = \frac{1}{n} \underbrace{V\Delta^\top U^\top}_{X^\top} \underbrace{U\Delta V^\top}_{X} = \frac{1}{n} V\Delta^2 V^\top, \quad \text{with} \quad \Delta^2 = \operatorname{diag}(\delta_1^2, \ldots, \delta_r^2).$$

Here the fact that U has orthonormal columns implies $U^\top U$ is the $r \times r$ identity matrix and hence it cancels out:

$$S = \sum_{i=1}^{r} \frac{\delta_i^2}{n} v_i v_i^\top.$$

$$S = \sum_{i=1}^{r} \frac{\delta_i^2}{n} v_i v_i^\top.$$

- Using the Spectral decomposition of S

$$S = \widetilde{V}^\top \Lambda \widetilde{V} = \sum_{i=1}^{r} \lambda_i v_i v_i^\top.$$

- Thus, $\lambda_i = \delta_i^2 / n$ and the loading vectors in spectral decomposition are the right singular vectors in SVD: $\widetilde{V} = V$.
- Further, to obtain the data matrix of principal components, $\widetilde{X}$ we set $\widetilde{X} = XV$.
- Using the SVD, PCA can be represented:

$$\widetilde{X} = \underbrace{U \Delta V^\top}_{X} V = U\Delta = \begin{bmatrix} | & | & & | \\ \delta_1 u_1 & \delta_2 u_2 & \dots & \delta_r u_r \\ | & | & & | \end{bmatrix}.$$

- Each column of the reduced data matrix $\widetilde{X}$ is a left singular vector u_i stretched by the singular value δ_i.

SVD for Compression

- The singular value decomposition can also be viewed as a means for compressing any matrix X.
- A rank $m < r$ approximation of X is,

$$\widehat{X} = \sum_{i=1}^{m} \delta_i \, u_i \, v_i^\top \approx X, \qquad \text{where} \qquad X - \widehat{X} = \sum_{i=m+1}^{r} \delta_i \, u_i \, v_i^\top.$$

- The rank of $\widehat{X}$ is m and since one often uses m significantly smaller than r, this is called a low rank approximation.

- For small enough δ_{m+1} the approximation error is negligible since the summation of rank one matrices $\delta_i \, u_i \, v_i^\top$ for $i = m + 1, \ldots, r$ is small.

- The number of values used in this representation of $\widehat{X}$ is $m \times (1 + n + p)$ and for small m this number is generally much smaller than $n \times p$ which is the number of values in X.

- Hence this may viewed as a compression method.

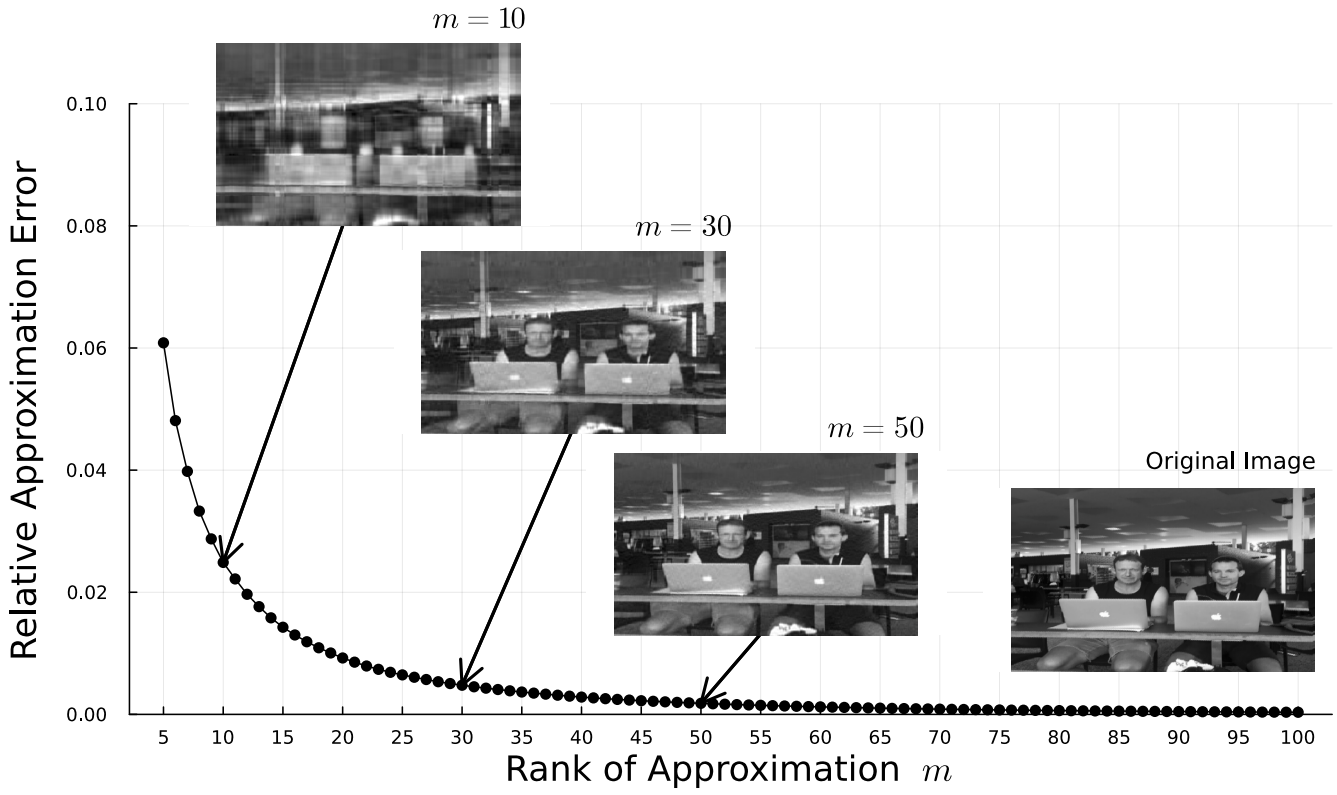
SVD in action for compression

- We seek to have the best rank m approximation in terms of minimization of $||X - \widehat{X}||_F$.
- Frobenious norm noted $||A||_F$: square root of the sum of the squared elements of the matrix A
- Low rank approximations established by **Eckart–Young-Mirsky theorem**.

$$\min_{\widehat{X} \text{ of rank } m} \left\| X - \widehat{X} \right\|_F^2 = \left\| X - \sum_{i=1}^{m} \delta_i \, u_i v_i^\top \right\|_F^2 = \sum_{i=m+1}^{r} \delta_i^2 .$$

- Consider a simple visual example with a 353×469 monochrome (grayscale) image appearing this is X.

Compression using SVD



- The original image uses $353 \times 469 = 165,557$ values while the $m = 50$ approximation only uses $50 \times (1 + 353 + 469) = 41,150$ values.
- That is the approximation yields $\widehat{X}$ which is compressed to about 25% of the size of X and looks very similar.

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)
}
```

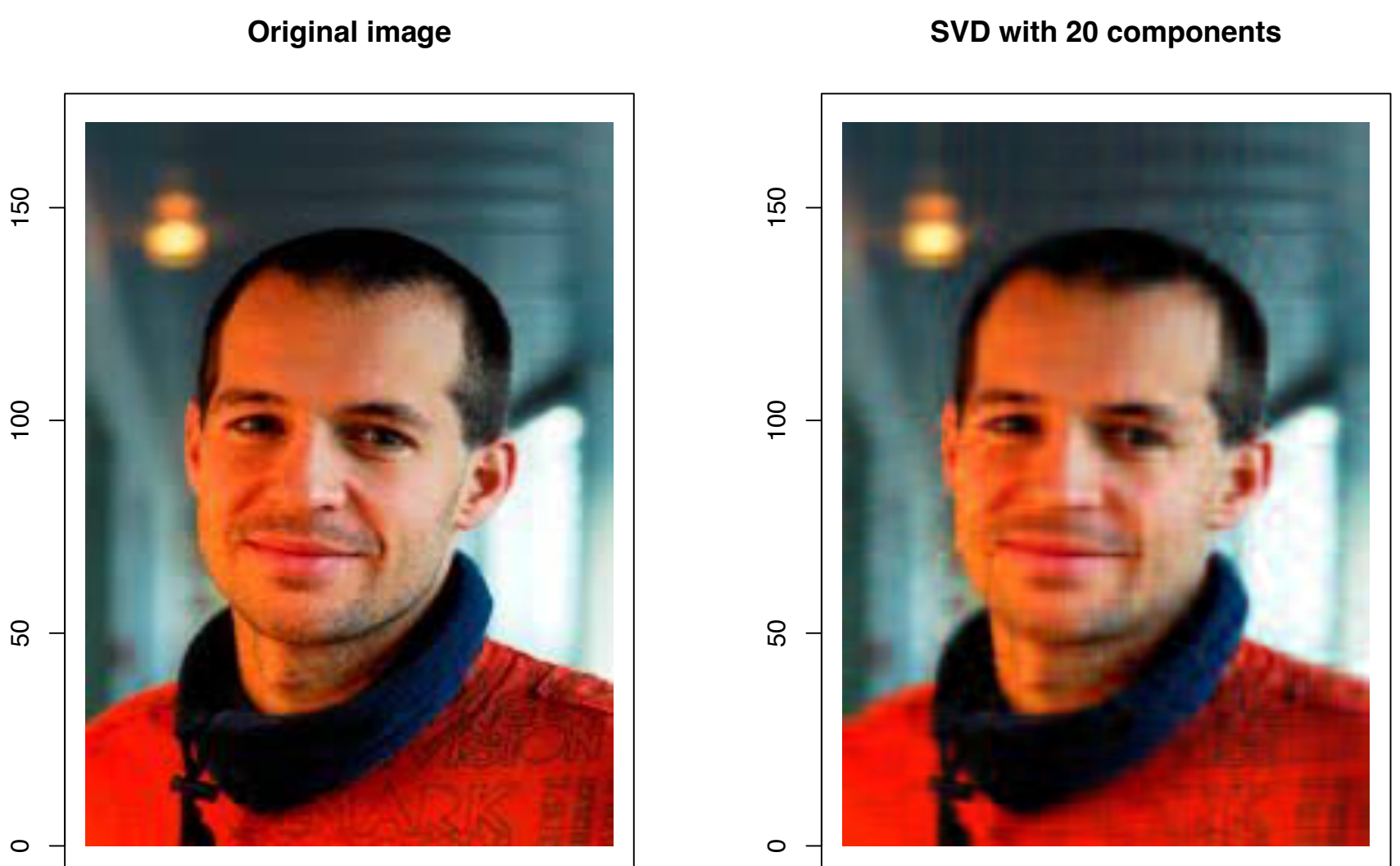
Function to compress an image via SVD of each channel

```r
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

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
```

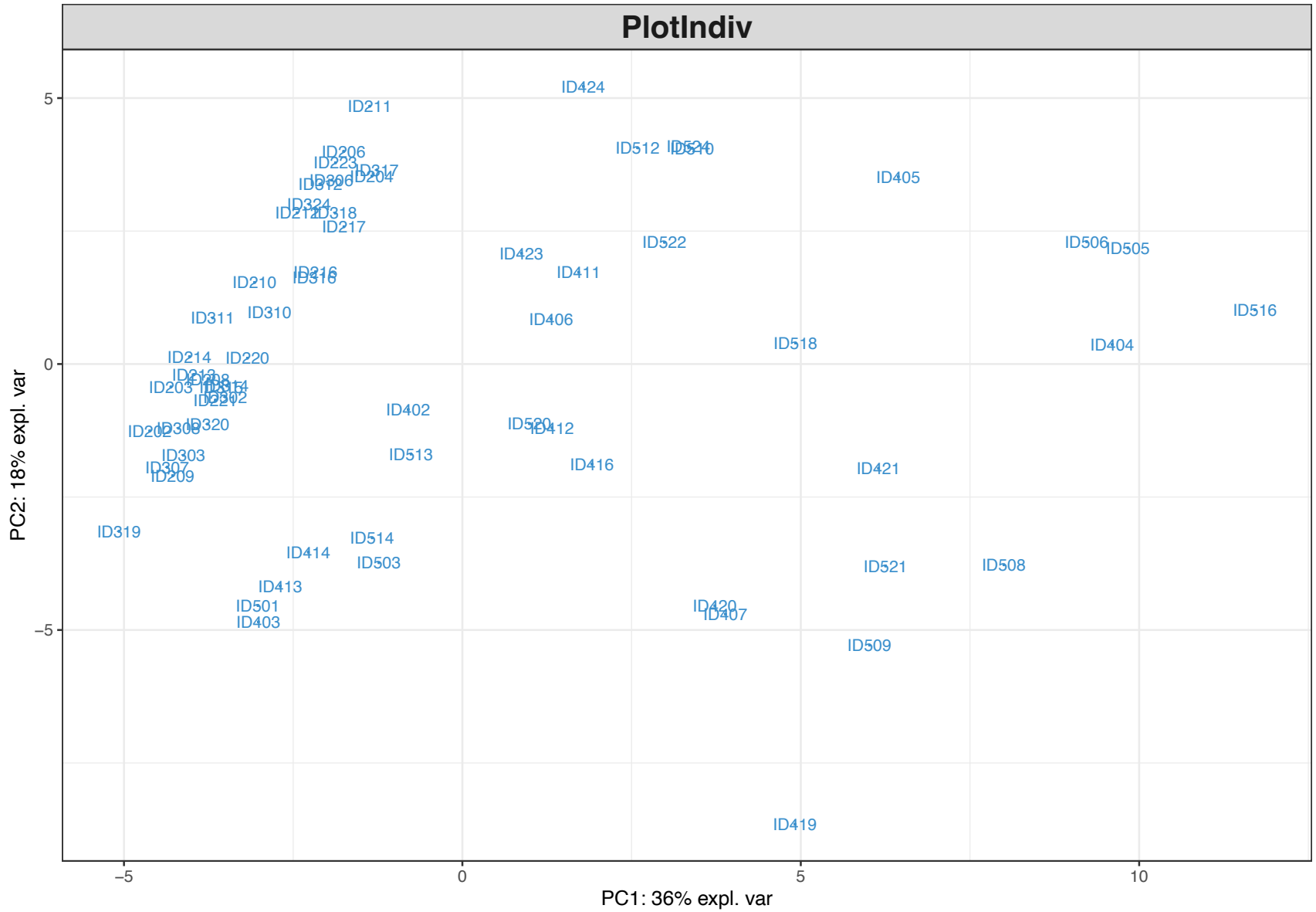
```
plotVar(MyResult.pca)       # 3 Plot the variables
```

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).

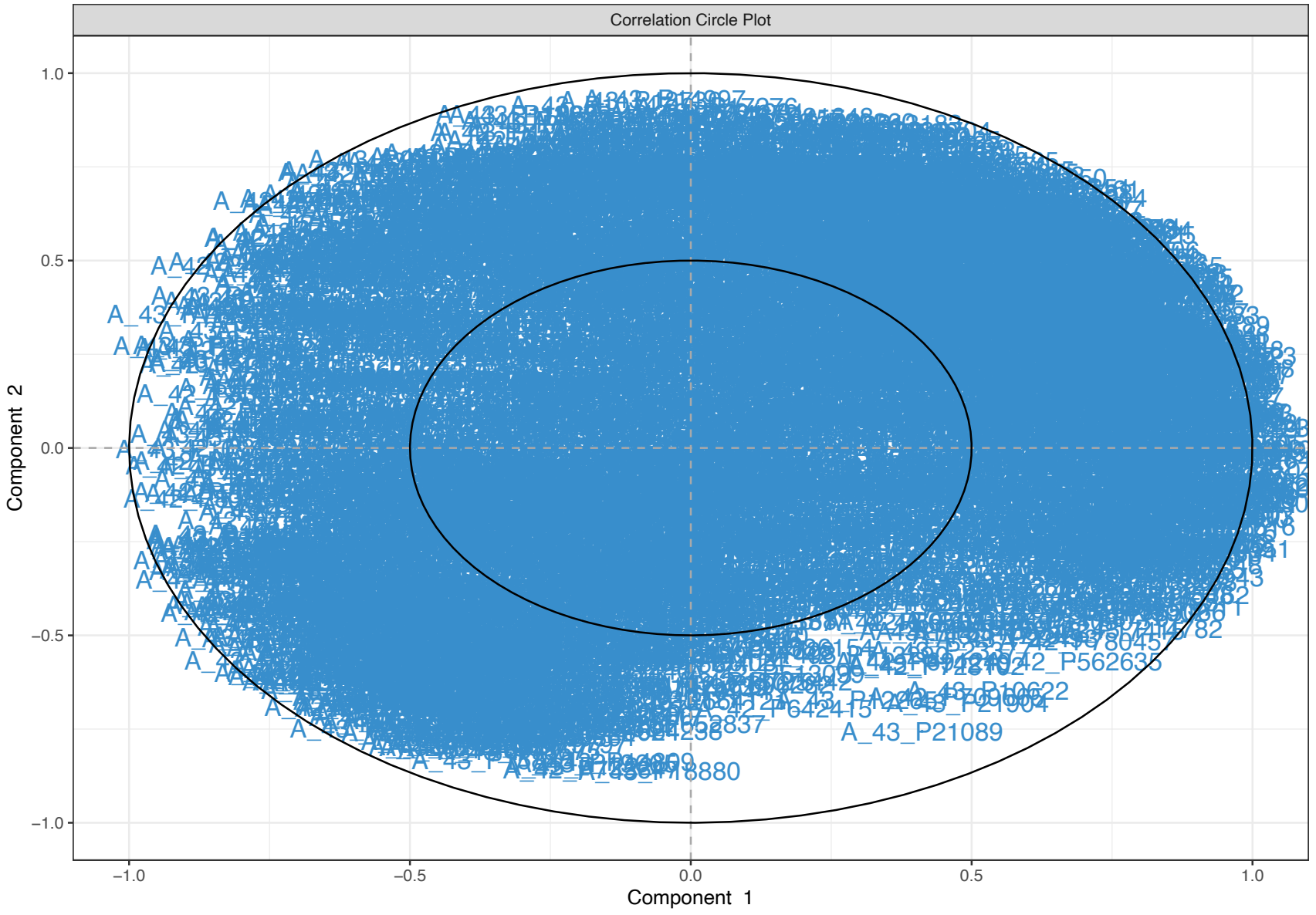
Plot the samples

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



Plot the variables

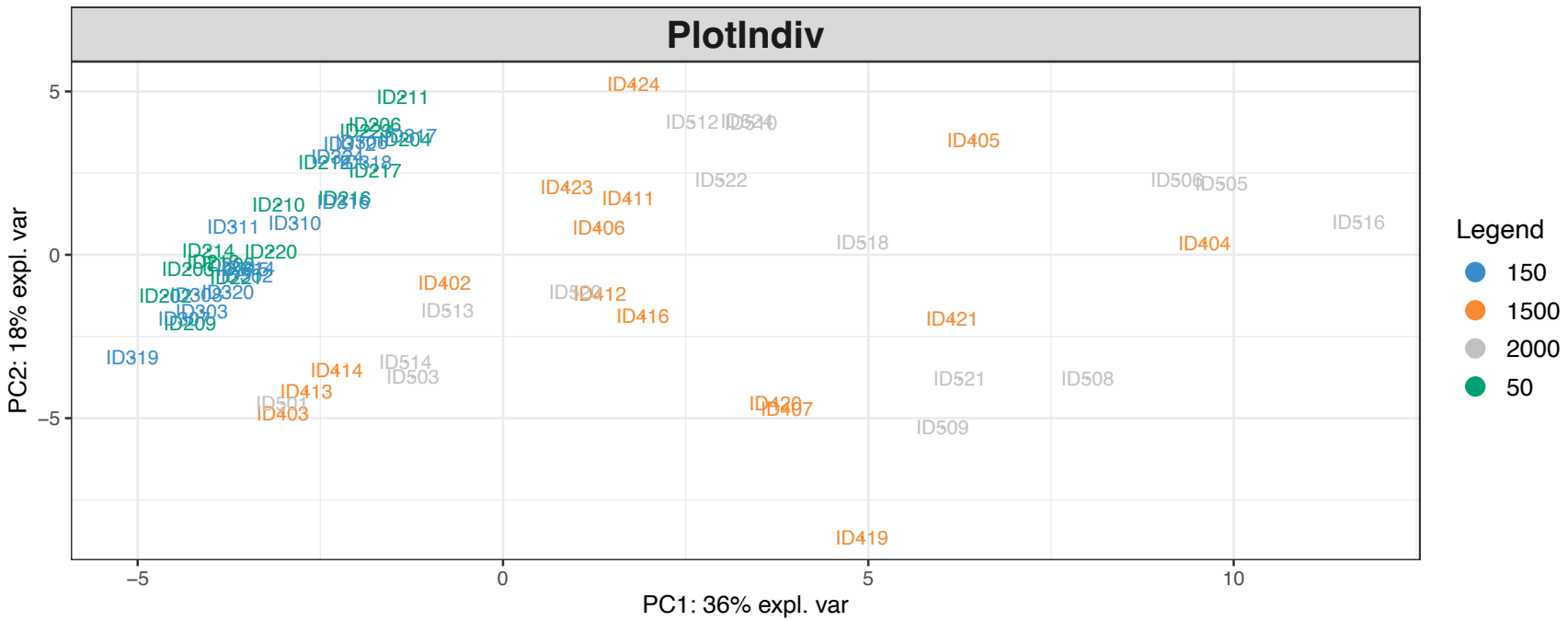
```
plotVar(MyResult.pca)
```



Customize plots

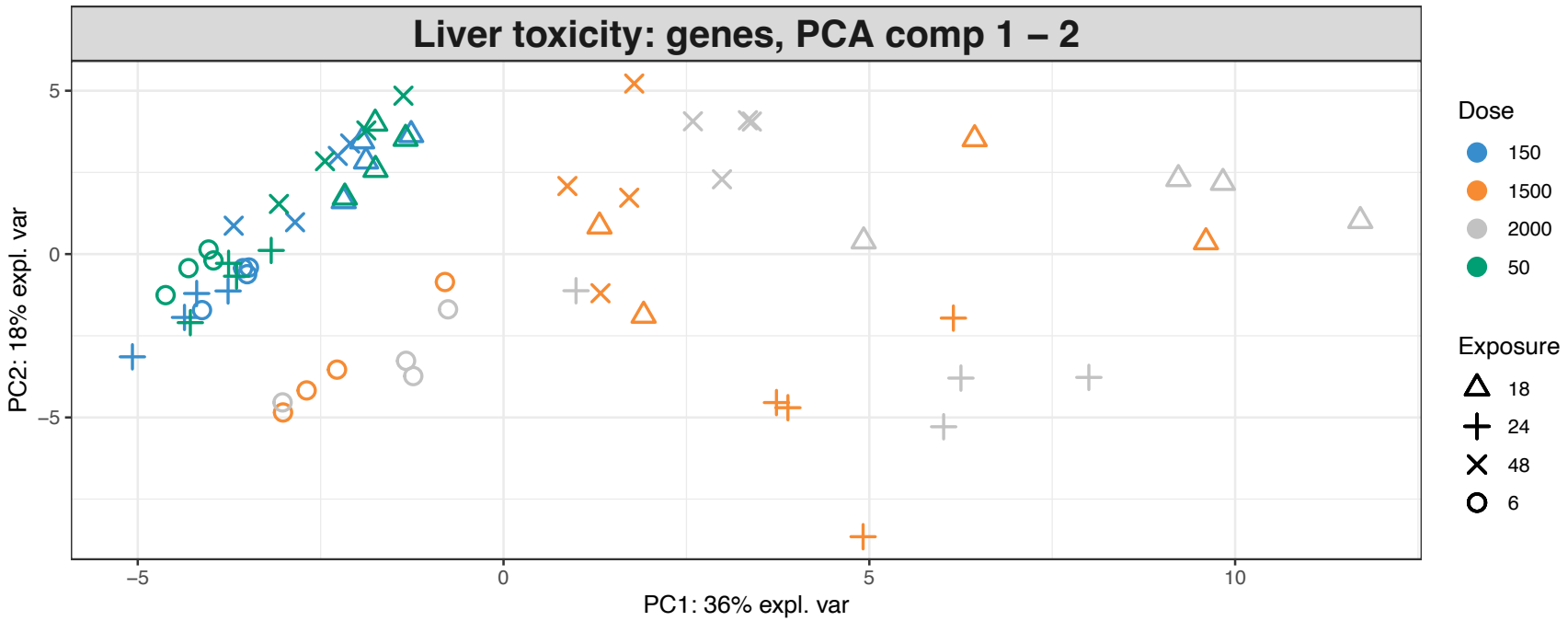
Here is an example where we include the sample groups information with the argument `group`:

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



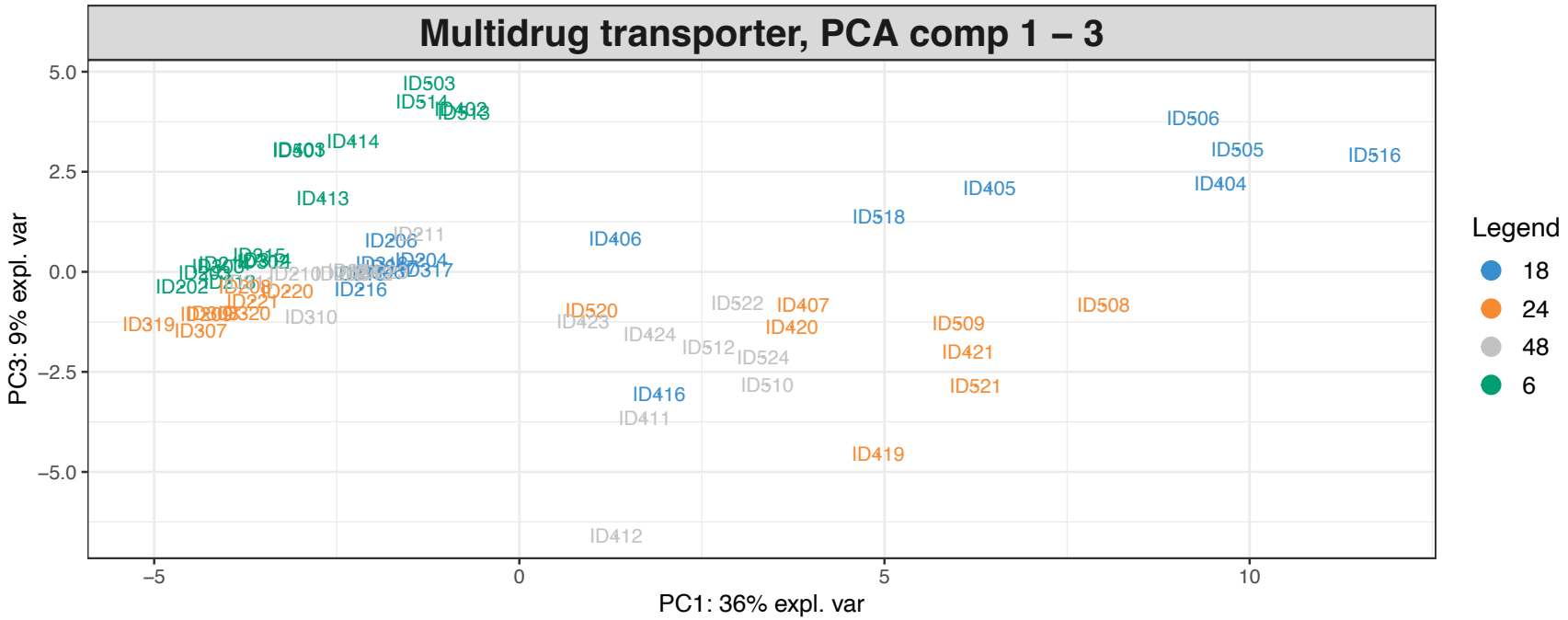
Customize plots: two factors displayed

```
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:

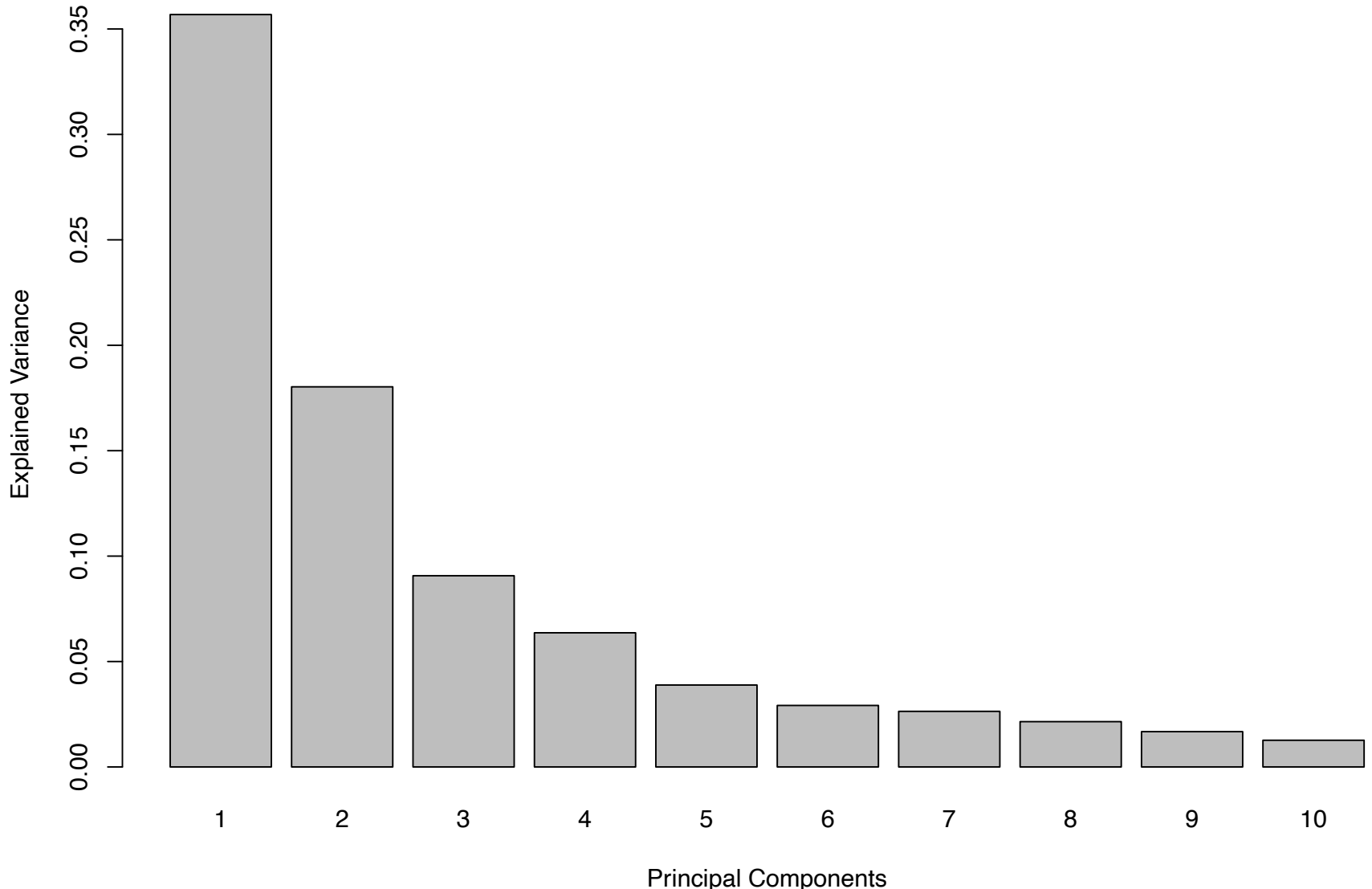
```
MyResult.pca2 <- pca(X, ncomp = 3)
plotIndiv(MyResult.pca2, comp = c(1,3), legend = TRUE,
          group = liver.toxicity$treatment$Time.Group,
          title = 'Multidrug transporter, PCA comp 1 - 3')
```



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)
```



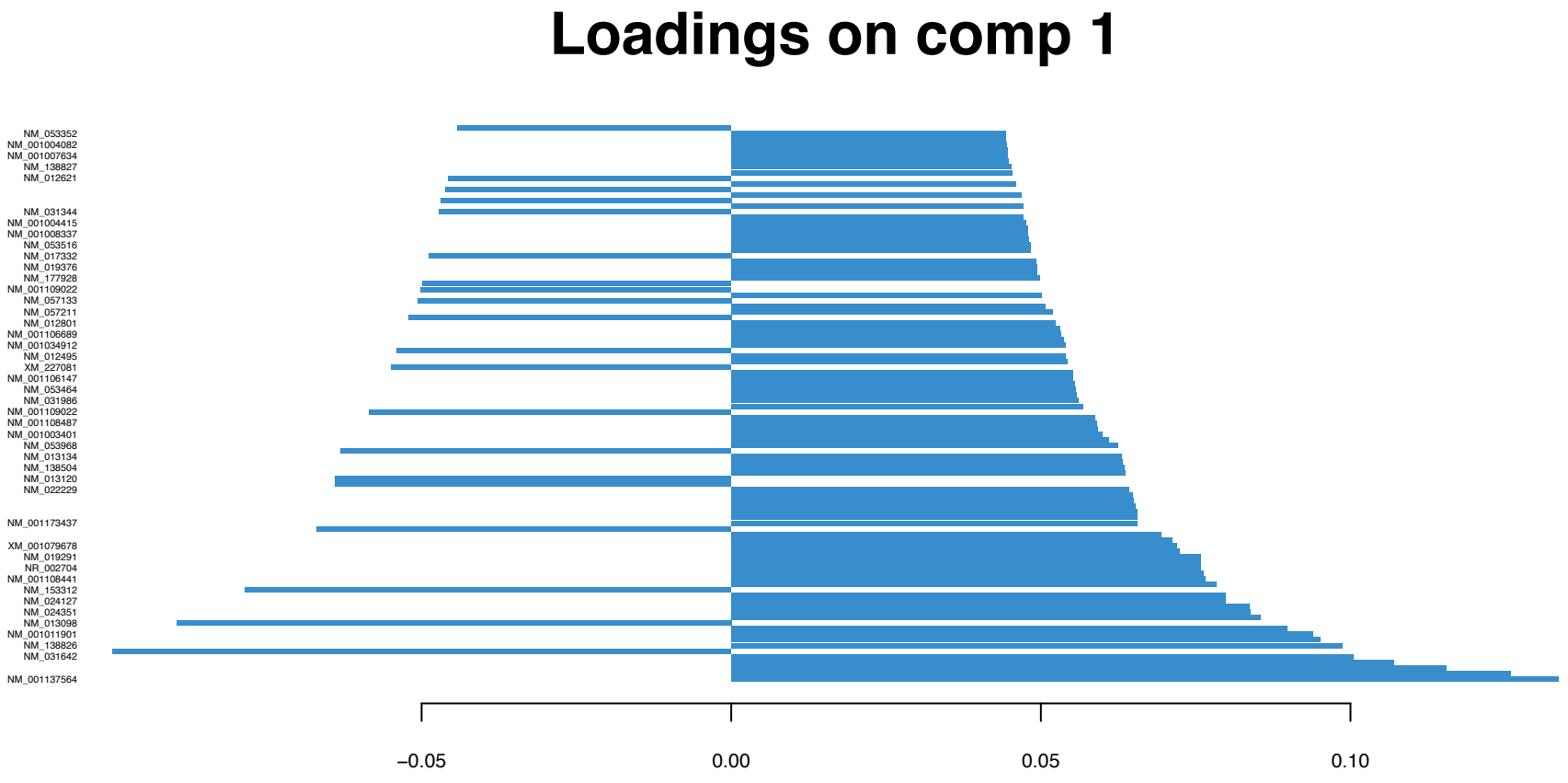
Other useful plots

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

- PCA can ve viewed as a linear autoencoder