In this exercise denote observations by $x_i \in \mathbb{R}^d$ and variables by $x_{(j)} \in \mathbb{R}^n$. Then data matrix $X \in \mathbb{R}^{n \times d}$ can be written as

$$X = \begin{pmatrix} x_1^\top \\ x_2^\top \\ \vdots \\ x_n^\top \end{pmatrix} = \begin{pmatrix} x_{(1)} & x_{(2)} & \dots & x_{(d)} \end{pmatrix}.$$

Demo Problem 1: Principal Component Analysis

Upload the file `decathlon.txt` into your R workspace. The file contains the results of 48 decathletes from 1973. Familiarize yourself with the data and perform the covariance matrix based PCA transformation. Conduct the analysis without the variables: points, height and weight.

a) Familiarize yourself with the function `princomp`. Visualize the original data.

b) How much of the variation of the original data is explained by k principal components, where $k = 1, 2, \dots, 10$.

c) Choose a sufficient amount of principal components and try to interpret them. Visualize the scores of the observations with respect to the first two principal components.

d) Calculate the sample mean and covariance matrix from the score matrix.

Solution

First we read the data.

```
decat <- read.table("../data/decathlon.txt", header = TRUE, sep = "\t",
                    row.names = 1)
```

Data includes results of 48 decathletes ("kymmenottelija" in Finnish). We remove variables `Points`, `Height` and `Weight` from the analysis.

```
decat <- decat[, -c(1, 12, 13)]
head(decat)
```

```
##          R100m Long_jump Shot_put High_jump R400m Hurdles Discus_throw
## Skowrone   853       931      725       857   838     903          772
## Hedmark    853       853      814       769   833     914          855
## Le_Roy     879       951      799       779   838     881          819
## Zeilbaue   826       931      793       865   875     891          729
## Zigert     879       840      924       857   788     892          866
## Bennett    905       859      647       779   938     859          651
##          Pole_vault Javelin R1500m
## Skowrone        981     818    528
## Hedmark         884     975    438
## Le_Roy         1028     758    408
## Zeilbaue        909     774    543
## Zigert          920     671    497
## Bennett        1028     794    661
```

```
dim(decat)
```

```
## [1] 48 10
```

a) First we visualize the original data. One way to visualize the data is a pairwise scatter plot. It is hard to get any sense from Figure 1.

```
pairs(decat, gap = 0, upper.panel = NULL)
```

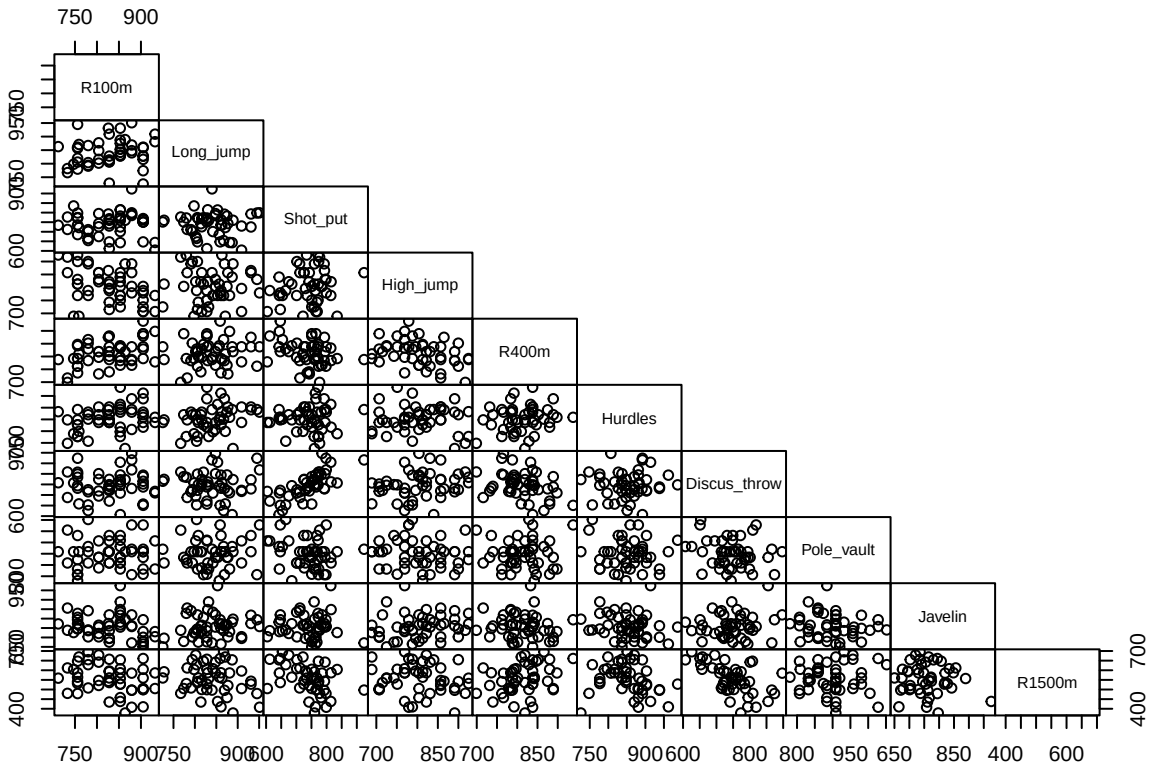

Figure 1: Pairwise scatter plots of all variables.

Figure 2 shows just one of the scatter plots. In this particular scatter plot, points are replaces with names of decathletes.

```
plot(decat$R100m, decat$R400m, xlab = "Running 100m", ylab = "Running 400m",
     type = "n")
text(decat$R100m, decat$R400m, labels = rownames(decat), cex = 0.5)
```

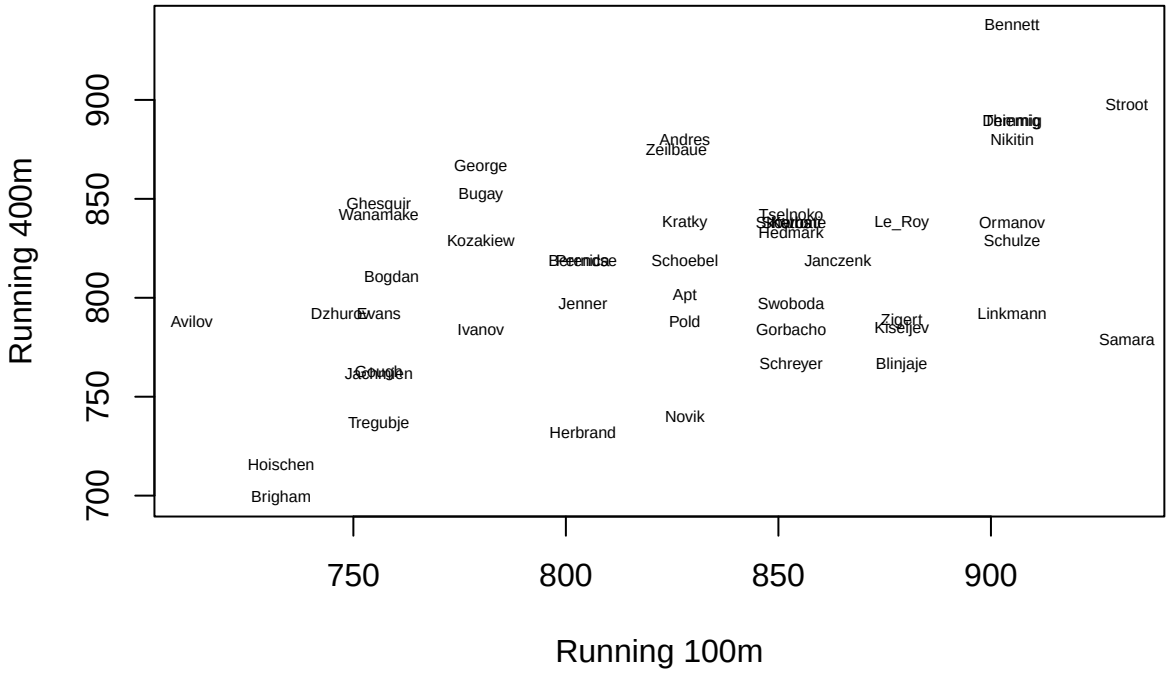

Figure 2: Scatter plot of two specific variables.

Next we familiarize ourselves with `princomp` function. Good place to start is the help pages that you can see with the command `?princomp`.

Now let us perform principal component analysis with decathlon data. We set argument `cor` as `FALSE`. This means that we perform PCA with covariance matrix.

```
decat_pca <- princomp(decat, cor = FALSE)
names(decat_pca)
```

```
## [1] "sdev"     "loadings" "center"   "scale"    "n.obs"    "scores"   "call"
```

Function `princomp` returns an object of class `princomp`, that is essentially a list of objects. Help pages give short explanations for each returned object. Let us familiarize ourselves with all the returned objects.

- Scores, that is, transformed variables are given by

$$Y = (X - 1_n \bar{x}^\top)G,$$

where G is the matrix of eigenvectors of the sample covariance and $\bar{x}$ is the sample mean vector. That is observations observations y_i can be written as

$$y_i = G^\top (x_i - \bar{x}), \quad i \in \{1, \ldots, n\},$$

and variables $y_{(j)}$ can be written as

$$y_{(j)} = (X - 1_n \bar{x}^\top) g_{(j)}, \quad j \in \{1, \ldots, d\}.$$

```
score <- decat_pca$scores
class(score)
```

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

```
head(score)
```

```
##              Comp.1    Comp.2      Comp.3     Comp.4     Comp.5    Comp.6
## Skowrone   33.64990  32.80843   5.2886457 -129.02080 -41.734435 -57.39629
## Hedmark   180.86806  25.21787 136.4300139 -105.56085 103.258919 -36.35881
## Le_Roy    133.06995 141.82592 -66.6008641 -116.89564  22.224933 -60.43915
## Zeilbaue   30.67033  16.85475  35.1390736  -46.63993 -69.713661 -67.06956
## Zigert    189.55708  56.41780 -77.9969270   95.55713 -22.944726 -80.98631
## Bennett  -203.44361  75.84875   0.4713267 -101.01300   8.417874 -34.76053
##              Comp.7     Comp.8     Comp.9    Comp.10
## Skowrone  40.26011  -4.411489  21.329168   3.937407
## Hedmark   14.37754  86.014441   1.802938 -23.795102
## Le_Roy    15.72299 -45.063628 -10.474568 -29.040083
## Zeilbaue  42.25378 -37.916406 -55.128784 -14.342753
## Zigert    57.07945  -8.271583 -18.848344  39.267980
## Bennett  100.60549  24.896091  -5.119935   1.514836
```

```
dim(score)
```

```
## [1] 48 10
```

Note that the function `princomp` uses divisor n instead of $n-1$ while calculating the sample covariance. Thus we can calculate scores manually in the following way.

```
n <- nrow(decat)
e <- eigen((n-1) / n * cov(decat))
score_manual <- as.matrix(sweep(decat, 2, colMeans(decat), "-")) %*% e$vectors

# Principal components are only unique up to sign
all(abs(round(score_manual, 2)) == abs(round(score, 2)))
```

```
## [1] TRUE
```

- Standard deviations of each principal component `sdev`. Remember that these are just the square roots of eigenvalues of the sample covariance matrix corresponding to original data set.

```
decat_pca$sdev
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 102.99508  83.83611  63.99022  63.48050  58.06222  47.35445  43.07928  39.76028
##    Comp.9   Comp.10
##  30.35520  28.98388
```

```
sqrt(e$values)
```

```
##  [1] 102.99508  83.83611  63.99022  63.48050  58.06222  47.35445  43.07928
##  [8]  39.76028  30.35520  28.98388
```

- Loadings, that is, eigenvector matrix G corresponding sample covariance matrix. Notice that this is not a matrix object but a special object of class loadings. Values very close to zero are showed as empty when the object is printed.

```
load <- decat_pca$loadings
class(load)
```

```
## [1] "loadings"
```

```
load
```

```
##
## Loadings:
##              Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
## R100m                 0.606  0.135                0.255  0.415
## Long_jump             0.175  0.220 -0.416 -0.234 -0.377 -0.176 -0.712
## Shot_put      0.462                0.340  0.183 -0.471  0.147        -0.598
## High_jump     0.233 -0.456 -0.113 -0.150 -0.649  0.132  0.494
## R400m        -0.228  0.276  0.277               -0.111  0.573        -0.175
## Hurdles               0.321  0.184        -0.510 -0.395 -0.264  0.598  0.107
## Discus_throw  0.516                0.155  0.214 -0.226  0.211         0.731
## Pole_vault   -0.139  0.145 -0.704 -0.511  0.177 -0.321  0.201  0.168
## Javelin       0.116 -0.305  0.546 -0.566  0.385         0.102  0.318
## R1500m       -0.609 -0.317  0.125  0.285        -0.479  0.193         0.230
##              Comp.10
## R100m         0.605
## Long_jump
## Shot_put      0.187
## High_jump     0.136
## R400m        -0.647
## Hurdles
## Discus_throw -0.170
## Pole_vault
## Javelin       0.102
## R1500m        0.334
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9
## SS loadings       1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.1    0.1    0.1    0.1    0.1    0.1    0.1    0.1    0.1
## Cumulative Var    0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9
##                Comp.10
## SS loadings        1.0
## Proportion Var     0.1
## Cumulative Var     1.0
```

```
# One can access elements of loadings object similarly to a matrix
load[1, 1]
```

```
## [1] 0.01446843
```

```
# One can transform loadings object to a matrix
load_mat <- load[]
class(load_mat)
```

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

- Mean vector corresponding to original data set.

```
decat_pca$center
```

```
##        R100m    Long_jump     Shot_put    High_jump        R400m      Hurdles
##     828.1875     840.1875     740.7708     805.8542     813.5000     852.8750
## Discus_throw   Pole_vault      Javelin       R1500m
##     747.4583     900.2708     760.0208     554.6250
```

- Scales are relevant when `cor = TRUE`. Actually, in the case `cor = TRUE`, scales are equal to the standard deviations of original variables $x_{(j)}$, $j \in \{1, \ldots, d\}$. This is due to the fact that (sample) correlation matrix is equal to the (sample) covariance matrix of standardized variables $x_{(j)}/\text{std}(x_{(j)})$.

```
princomp(decat, cor = TRUE)$scale
```

```
##        R100m    Long_jump     Shot_put    High_jump        R400m      Hurdles
##     58.68158     50.19738     61.18014     64.12650     49.28066     53.63714
## Discus_throw   Pole_vault      Javelin       R1500m
##     61.62993     62.38280     63.26745     75.86958
```

```
sqrt((n - 1) / n * apply(decat, 2, var))
```

```
##        R100m    Long_jump     Shot_put    High_jump        R400m      Hurdles
##     58.68158     50.19738     61.18014     64.12650     49.28066     53.63714
## Discus_throw   Pole_vault      Javelin       R1500m
##     61.62993     62.38280     63.26745     75.86958
```

b) Quantity

$$\text{Tr}(\Sigma) = \sum_{i=1}^{d} \Sigma_{ii} = \sum_{i=1}^{d} \lambda_i$$

gives one way to measure multivariate scatter. Then proportion of total variation

$$\frac{\sum_{i=1}^{k} \lambda_i}{\text{Tr}(\Sigma)}$$

measures amount of information retained in keeping k first principal components.

Proportion of total variation explained by the first k principal can be seen straight away with the `summary` function. See the `Cumulative Proportion` row in the `summary`.

```
summary(decat_pca)
```

```
## Importance of components:
##                             Comp.1     Comp.2     Comp.3     Comp.4      Comp.5
## Standard deviation     102.9950759 83.8361146 63.9902194 63.4804991 58.06221588
## Proportion of Variance   0.2900506  0.1921778  0.1119613  0.1101848  0.09217818
## Cumulative Proportion    0.2900506  0.4822284  0.5941898  0.7043745  0.79655273
##                             Comp.6      Comp.7      Comp.8      Comp.9
## Standard deviation     47.3544471 43.07927681 39.76028470 30.35519704
## Proportion of Variance  0.0613144  0.05074318  0.04322549  0.02519457
## Cumulative Proportion   0.8578671  0.90861031  0.95183579  0.97703037
##                            Comp.10
## Standard deviation     28.98388394
## Proportion of Variance  0.02296963
## Cumulative Proportion   1.00000000
```

One can also calculate proportions of variance manually.

```
vars <- decat_pca$sdev^2
var_prop <- vars / sum(vars)
var_prop_cum <- cumsum(var_prop)

var_prop # Proportion of Variance
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 0.29005064 0.19217779 0.11196134 0.11018477 0.09217818 0.06131440 0.05074318
##     Comp.8     Comp.9    Comp.10
```

```
## 0.04322549 0.02519457 0.02296963
```

```
var_prop_cum # Cumulative Proportion
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 0.2900506 0.4822284 0.5941898 0.7043745 0.7965527 0.8578671 0.9086103 0.9518358
##    Comp.9   Comp.10
## 0.9770304 1.0000000
```

Figure 3 shows so called scree plot. Scree plot can be used as a tool for choosing sufficient number of components.

```
plot(decat_pca, las = 2, main = NULL)
```

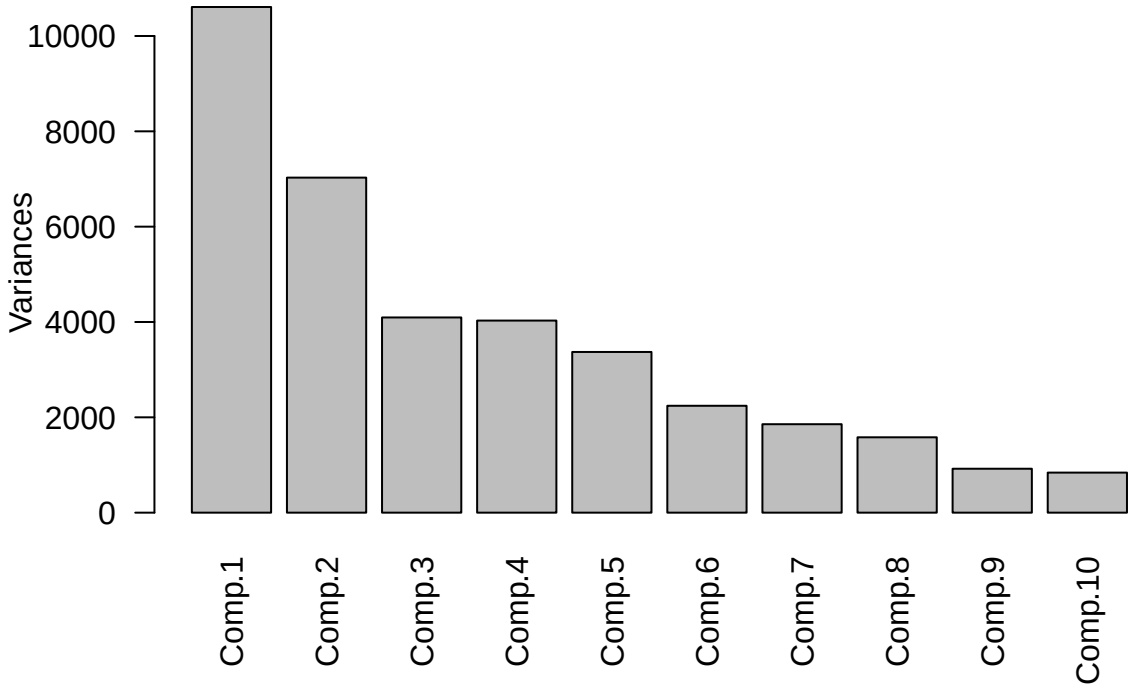

Figure 3: Scree plot.

Figure 4 can be also useful for choosing how many principal components to use.

```
plot(var_prop_cum, type = "b", pch = 21, lty = 3, bg = "skyblue", cex = 1.5,
     ylim = c(0, 1), xlab = "Principal component",
     ylab = "Cumulative proportion of variance explained",
     xaxt = "n", yaxt = "n")
axis(1, at = 1:10)
axis(2, at = 0:10 / 10, las = 2)
```

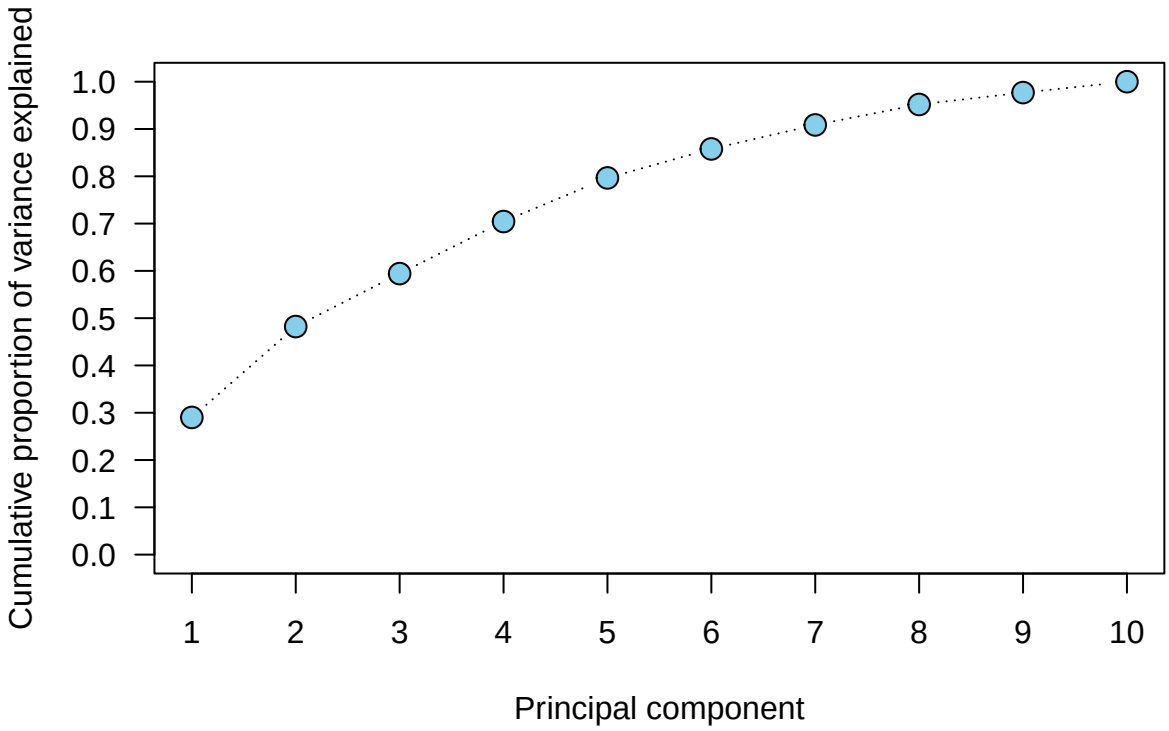

Figure 4: Cumulative proportion of variance explained by k principal components.

There are many more or less heuristic methods for choosing number of principal components, for example,

- include enough components to explain $x\%$ of total variation, where $x\%$ can chosen to be, e.g. 90%,
- Kaiser criterion: include those principal components whose eigenvalues are larger than average,
- elbow method,

among many others.

c) We choose four first principal components and try to interpret them. Together the first four components explain approximately 70% of variation in the original data.

Below snippet of code plots first two principal components and corresponding loadings. Notice that there are two coordinate systems, one for the principal components and other for the loadings.

```
pc12 <- score[, 1:2]
load12 <- load[, 1:2]
pc_axis <- c(-max(abs(pc12)), max(abs(pc12)))
ld_axis <- c(-0.8, 0.8)

plot(pc12, xlim = pc_axis, ylim = pc_axis, pch = 21, bg = "tomato", cex = 1.25,
     xlab = paste0("PC 1 (", round(100 * var_prop[1], 2), "%)"),
     ylab = paste0("PC 2 (", round(100 * var_prop[2], 2), "%)"))
par(new = T)
plot(load12, axes = F, type = "n", xlab = "", ylab = "", xlim = ld_axis,
     ylim = ld_axis)
```

```
axis(3)
axis(4)
arrows(0, 0, load12[, 1], load12[, 2], length = 0.05)
text(load12[, 1], load12[, 2], rownames(load12), pos = 3, cex = 0.5)
abline(h = 0, lty = 3)
abline(v = 0, lty = 3)
```

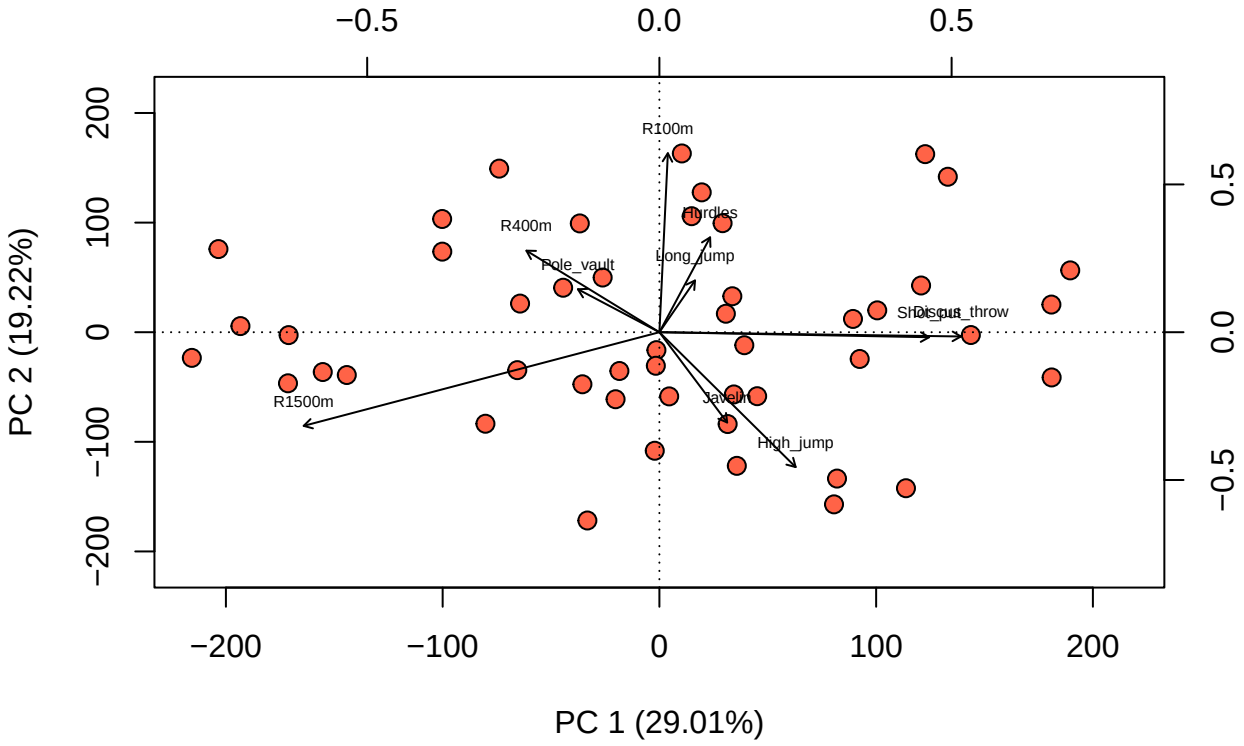

Figure 5: Biplot of scores and loadings.

You can try below command by yourself. It gives similar results to above plot.

```
biplot(decat_pca)
```

Loadings and biplots similar to Figure 5 can be used to interpret principal components. More precisely, we want to determine which variables contribute to components. Note that often expert knowledge is needed for interpreting principal components and interpretations are subjective.

Based on Figure 5, it seems that variables `Discus_throw` and `Shot_put` have significance positive contributions to the first principal component. On the other hand, `R1500m` has significant negative contribution to the first component. Thus first principal component tells that decathletes who are good at running long distances are very different compared to decathletes that are good at discus throw and shot put. Consequently, we could interpret first principal component as strength or bulkiness.

Similarly, we can interpret the second component. Variables such as `R100m`, `Hurdes` and `R400m` have significant positive contributions to the second principal component. On the other hand, variables such as `High_jump`, `R1500m` and `Javelin` have significant negative contributions to the second component. Thus the second principal component can be possibly interpreted as speed.

Interpretations for the third and fourth components are trickier. They can be maybe interpreted as techniques required for specific sports. You can look at the loading yourself to determine which sports contribute to the third and fourth components.

d) Mean vector corresponding to principal components is a zero vector.

```
round(colMeans(score), 2)
```

```
##  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8  Comp.9 Comp.10
##       0       0       0       0       0       0       0       0       0       0
```

Principal components y_i are uncorrelated, thus the sample covariance matrix is a diagonal matrix.

```
round(cov(score), 2)
```

```
##           Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8 Comp.9
## Comp.1  10833.69    0.00    0.00    0.00    0.00    0.00    0.00    0.00   0.00
## Comp.2      0.00 7178.04    0.00    0.00    0.00    0.00    0.00    0.00   0.00
## Comp.3      0.00    0.00 4181.87    0.00    0.00    0.00    0.00    0.00   0.00
## Comp.4      0.00    0.00    0.00 4115.51    0.00    0.00    0.00    0.00   0.00
## Comp.5      0.00    0.00    0.00    0.00 3442.95    0.00    0.00    0.00   0.00
## Comp.6      0.00    0.00    0.00    0.00    0.00 2290.16    0.00    0.00   0.00
## Comp.7      0.00    0.00    0.00    0.00    0.00    0.00 1895.31    0.00   0.00
## Comp.8      0.00    0.00    0.00    0.00    0.00    0.00    0.00 1614.52   0.00
## Comp.9      0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00 941.04
## Comp.10     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00   0.00
##          Comp.10
## Comp.1      0.00
## Comp.2      0.00
## Comp.3      0.00
## Comp.4      0.00
## Comp.5      0.00
## Comp.6      0.00
## Comp.7      0.00
## Comp.8      0.00
## Comp.9      0.00
## Comp.10   857.94
```

Diagonal elements of covariance matrix are just the variances of principal components.

```
(n - 1) / n * diag(cov(score))
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 10607.9857  7028.4941  4094.7482  4029.7738  3371.2209  2242.4437  1855.8241
##     Comp.8     Comp.9    Comp.10
##  1580.8802   921.4380   840.0655
```

```
decat_pca$sdev^2
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 10607.9857  7028.4941  4094.7482  4029.7738  3371.2209  2242.4437  1855.8241
##     Comp.8     Comp.9    Comp.10
##  1580.8802   921.4380   840.0655
```

Demo Problem 2: Eigendecomposition of a Symmetric Matrix

Let A be a symmetric matrix with distinct eigenvalues. Show that the eigenvector matrix of A is orthogonal.

Solution

Let λ_i be the ith eigenvalue and v_i the corresponding eigenvector of a $p \times p$ matrix A. The goal is to show that $v_i^\top v_j = 0$, $i \neq j$. We can order the eigenvalues such that $\lambda_1 > \lambda_2 > \cdots > \lambda_p$. The eigenvalues and -vectors satisfy:

$$\begin{cases} Av_i = \lambda_i v_i \\ Av_j = \lambda_j v_j. \end{cases}$$

First, we multiply the first equation with $v_j^\top$ from the left side,

$$\begin{aligned} & v_j^\top A v_i = \lambda_i v_j^\top v_i \\ & v_j^\top A^\top v_i = \lambda_i v_j^\top v_i \\ & (Av_j)^\top v_i = \lambda_i v_j^\top v_i \\ & \lambda_j v_j^\top v_i = \lambda_i v_j^\top v_i \\ & \Rightarrow (\lambda_j - \lambda_i)\, v_j^\top v_i = 0. \end{aligned}$$

Since $\lambda_i \neq \lambda_j$, vectors v_j and v_i have to be orthogonal: $v_j^T v_i = 0$, $i \neq j$. Hereby, the eigenvector matrix V of A satisfies $VV^T = I$, if we choose the eigenvectors of A that have length 1.

Homework Problem 1: PCA for Simulated Data

Simulate 100 observations from bivariate normal distribution with parameters

$$\mu = \begin{pmatrix} 4 \\ 7 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 10 & 6 \\ 6 & 8 \end{pmatrix}.$$

a) Plot the data. Label the data points with the corresponding observation number.

b) Perform the covariance based PCA transformation to the data set.

c) Plot the score matrix. Use the same scale as in a) and label the data points with the corresponding observation number. Choose your scale (limits for the x- and y-axis) in a way that all the observations are visible in the figure.

d) Compare the plots of a) and c) and describe the differences.

e) Calculate the G and Y matrices without using any existing PCA functions. Note that the function `princomp` scales the covariance matrix with $1/n$ (instead of the usual $1/(n-1)$). Attach the R code to your solution.

f) Verify that the estimated scores and the loadings are equal (up to signs) in parts b) and e). *Hint: If parts b) and e) are done correctly, the scores and loadings should be the same up to heterogeneous sign changes.*

g) Plot the directions of the first and second principal component to the original data. The function `arrows` might be useful.