# compgen2021: Week 2 exercises

Irem YUCEL

## Exercises for Week2

For this set of exercises we will be using the expression data shown below:

```
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")
mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)
```

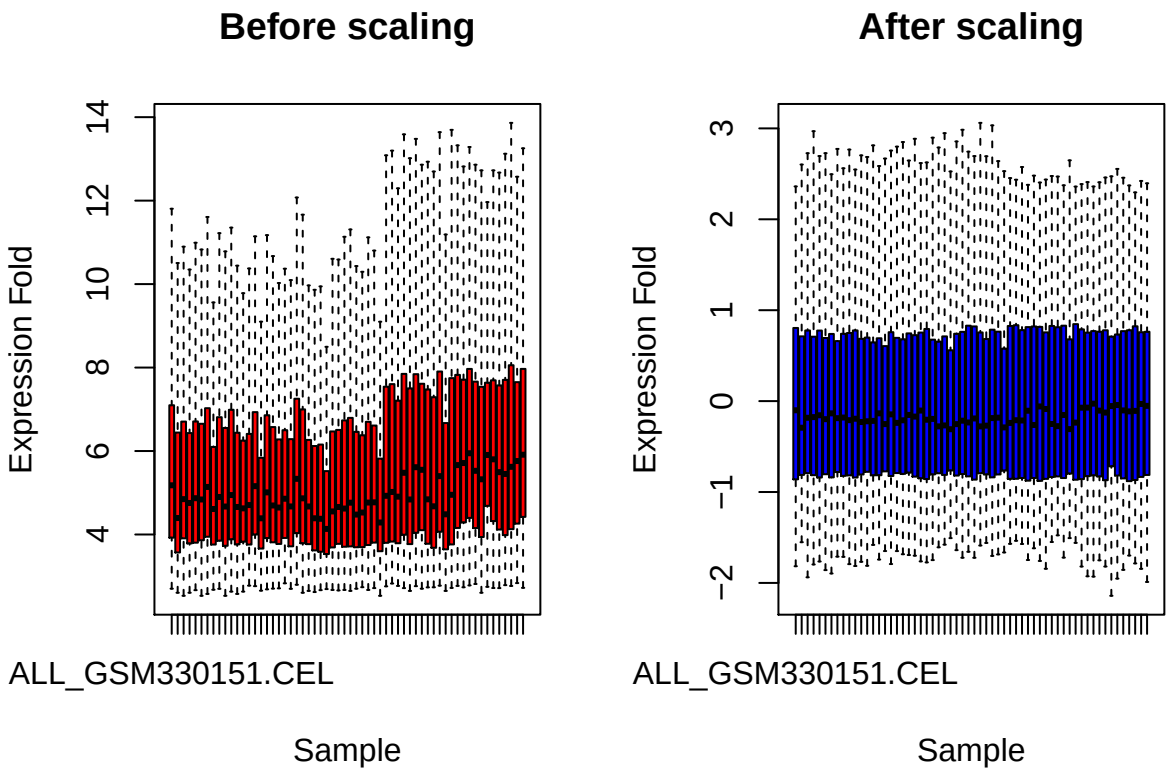
### Clustering

1. We want to observe the effect of data transformation in this exercise. Scale the expression matrix with the `scale()` function. In addition, try taking the logarithm of the data with the `log2()` function prior to scaling. Make box plots of the unscaled and scaled data sets using the `boxplot()` function. [Difficulty: **Beginner/Intermediate**]

**solution:** put your text here

```
#Log Transform and scale
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")
mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

scaled_mat <- scale(mat)

mat_log2 <- log2(mat+1)

scaled_mat_log2 <- scale(mat_log2)

par(mfrow=c(1,2))

boxplot(mat,
        main="Before scaling",
```

```
        xlab="Sample",
        ylab="Expression Fold",
        col='red',
        outline = FALSE)
boxplot(scaled_mat_log2,
        main="After scaling",
        xlab="Sample",
        ylab="Expression Fold",
        col='blue',
        outline = FALSE)
```

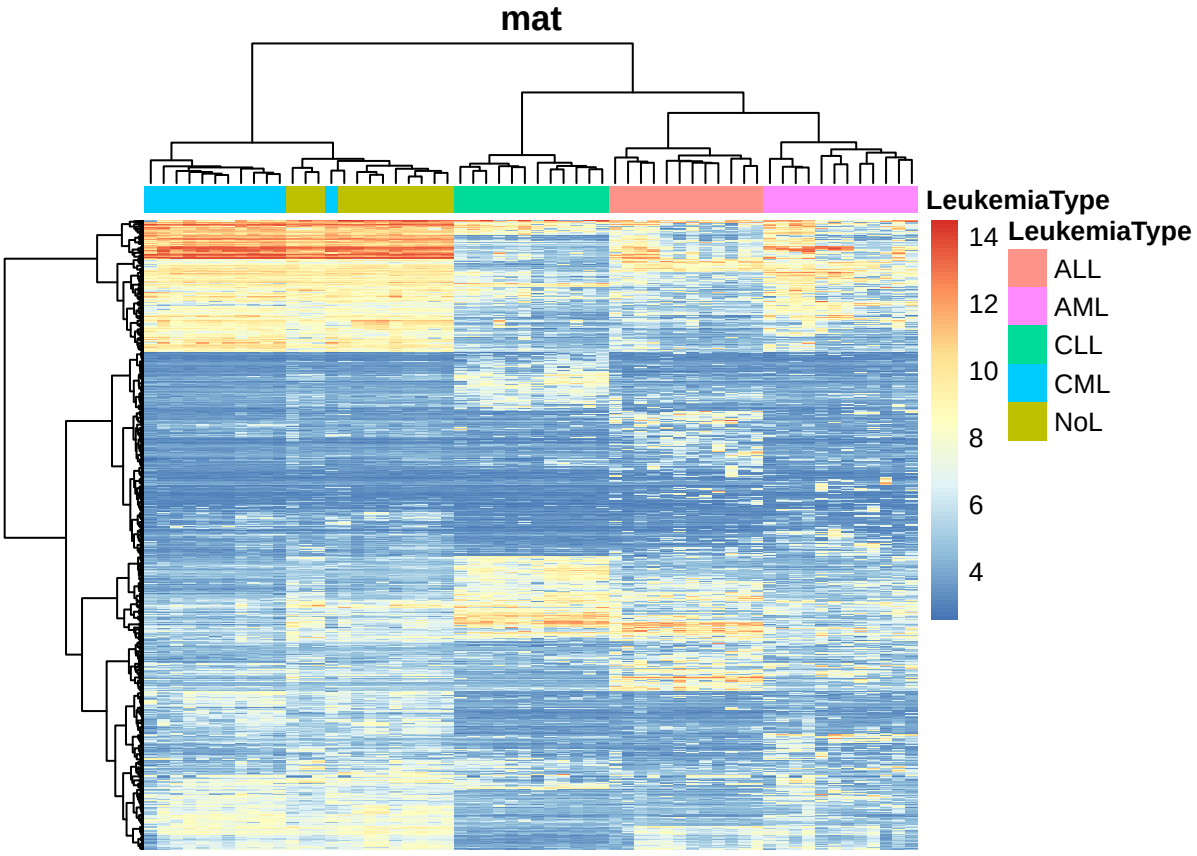



2. For the same problem above using the unscaled data and different data transformation strategies, use the `ward.d` distance in hierarchical clustering and plot multiple heatmaps. You can try to use the `pheatmap` library or any other library that can plot a heatmap with a dendrogram. Which data-scaling strategy provides more homogeneous clusters with respect to disease types? [Difficulty: **Beginner/Intermediate**]

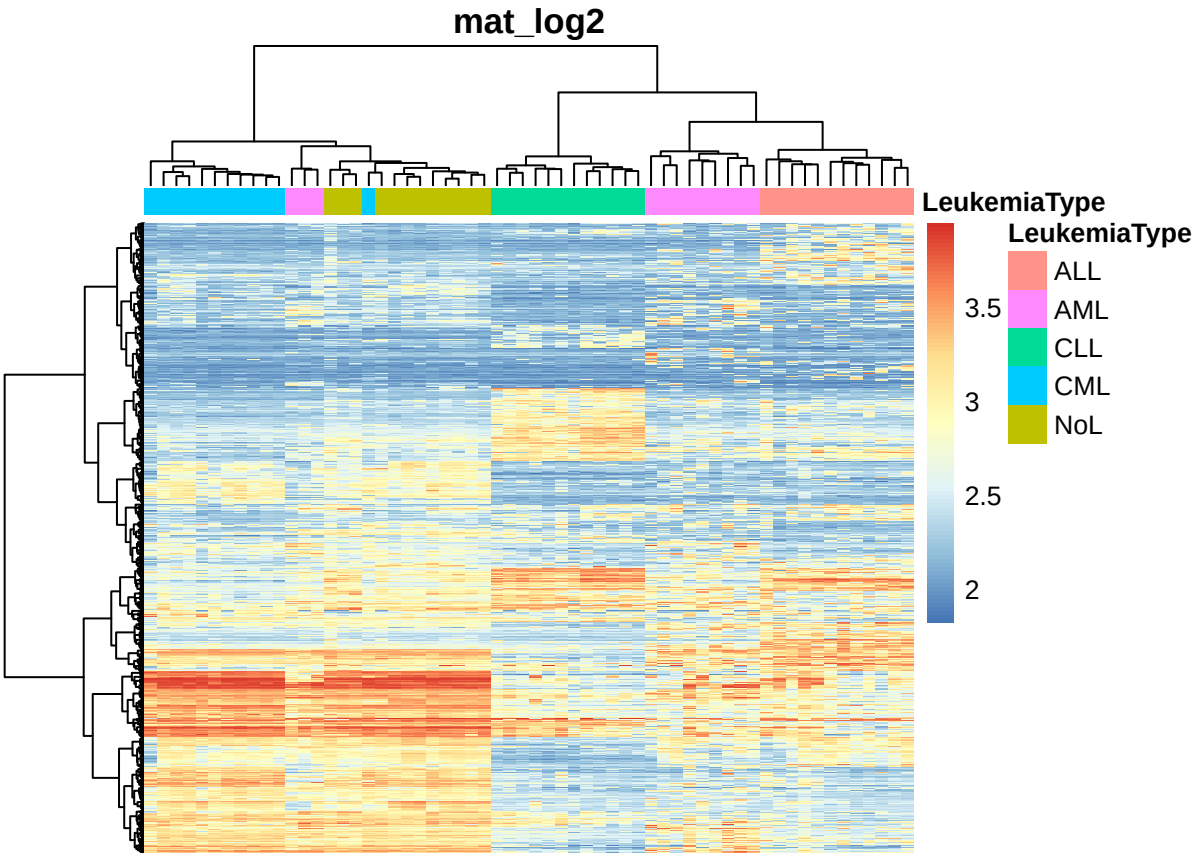
**solution:** Heatmaps after scaling have more homogeneous clusters with respect to disease types. Applying the log transform with out scaling seems to be the worst in terms of cluster homogeneity

```
library(pheatmap)

expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")
mat=readRDS(expFile)

annotation_col = data.frame(
```

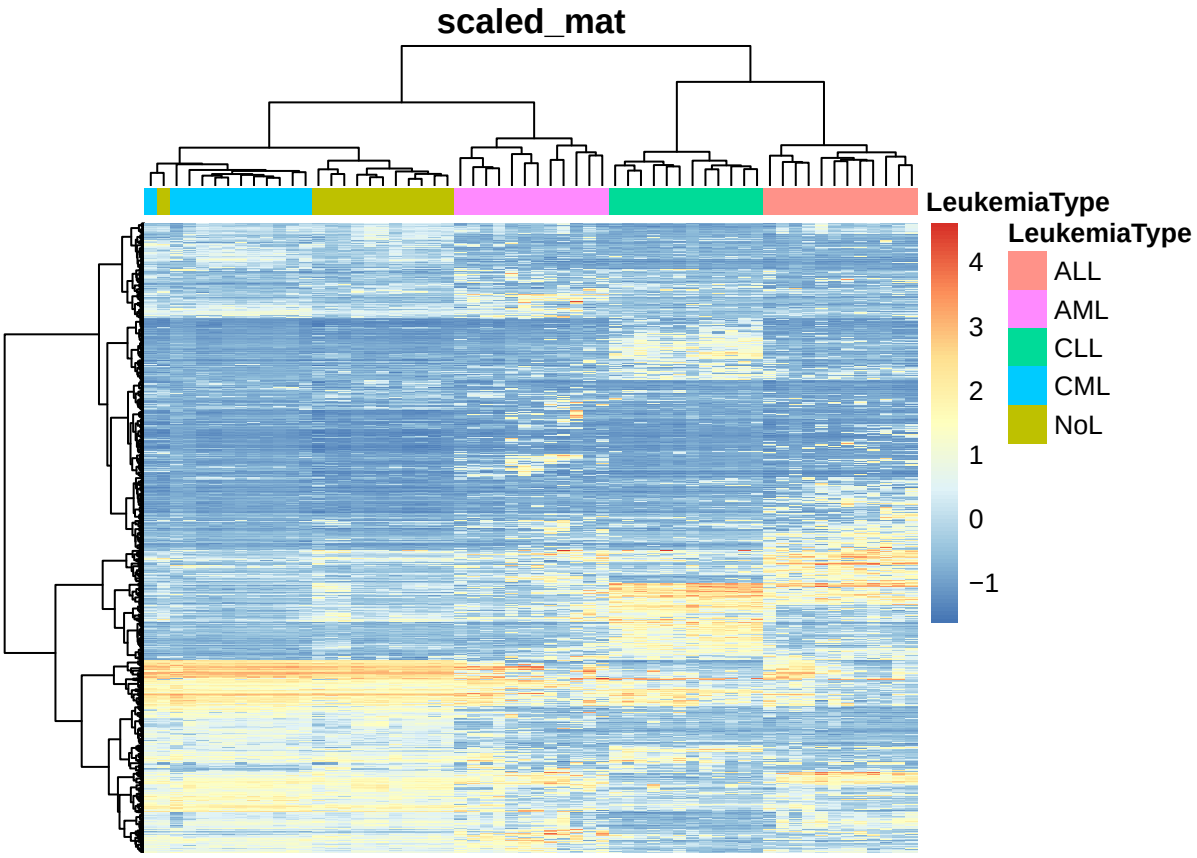
```
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

par(mfrow=c(2,2))
pheatmap(mat,
         main="mat",
         show_rownames=F,
         show_colnames=F,
         annotation_col=annotation_col,
         scale = "none",
         clustering_method="ward.D2",
         clustering_distance_cols="euclidean")
```



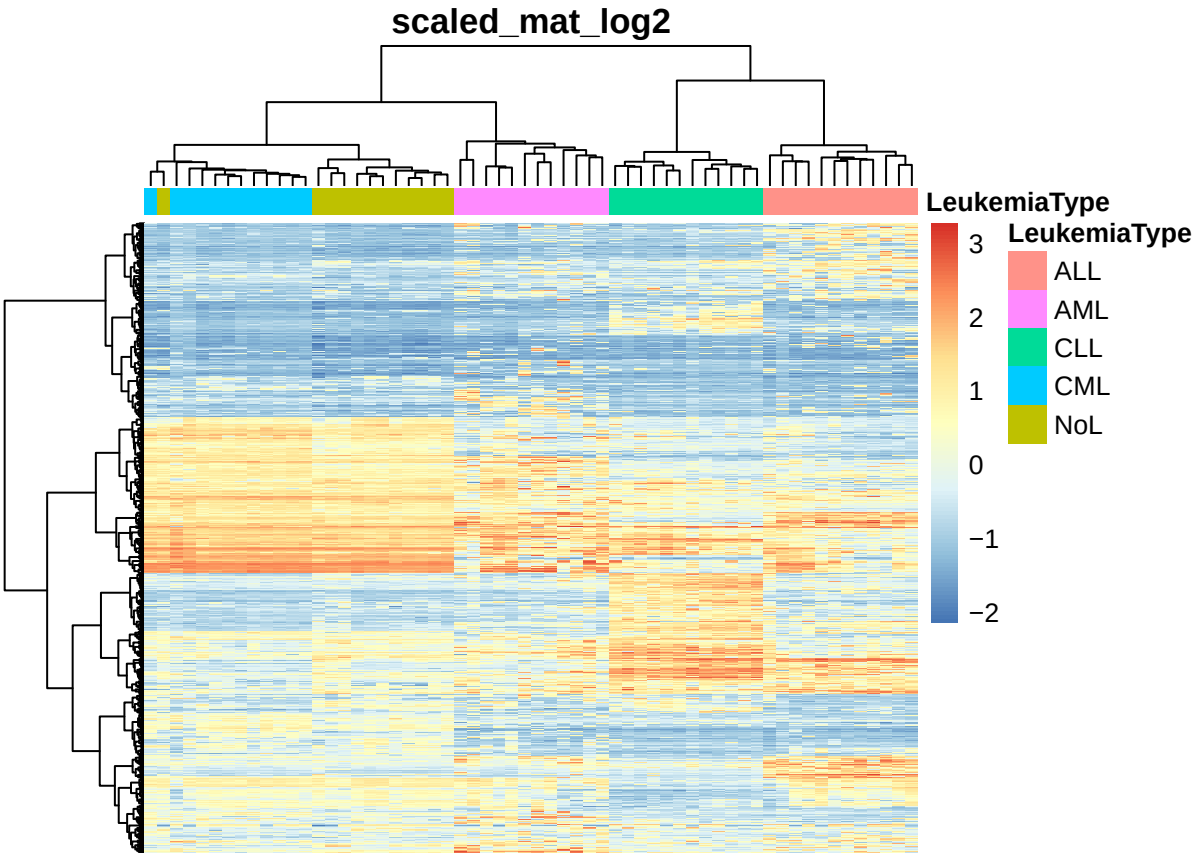

```
pheatmap(mat_log2,
         main="mat_log2",
         show_rownames=F,
         show_colnames=F,
         annotation_col=annotation_col,
         scale = "none",
         clustering_method="ward.D2",
         clustering_distance_cols="euclidean")
```




```
pheatmap(scaled_mat,
         main="scaled_mat",
         show_rownames=F,
         show_colnames=F,
         annotation_col=annotation_col,
         scale = "none",
         clustering_method="ward.D2",
         clustering_distance_cols="euclidean")
```




```
pheatmap(scaled_mat_log2,
         main="scaled_mat_log2",
         show_rownames=F,
         show_colnames=F,
         annotation_col=annotation_col,
         scale = "none",
         clustering_method="ward.D2",
         clustering_distance_cols="euclidean")
```



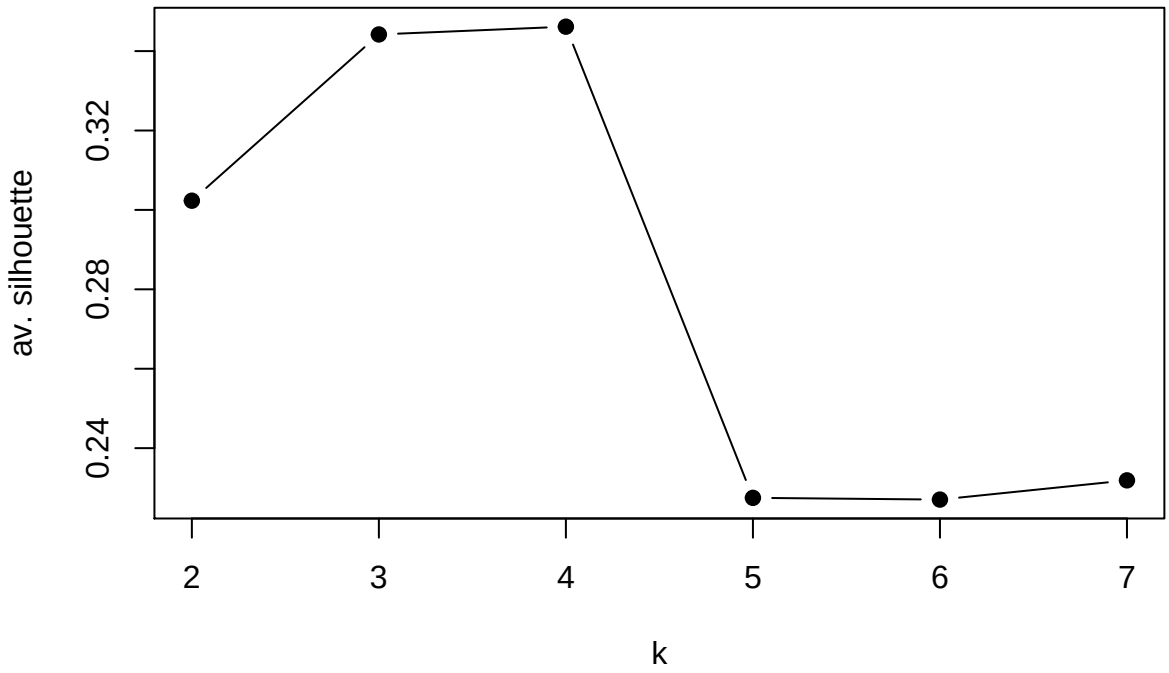

3. For the transformed and untransformed data sets used in the exercise above, use the silhouette for deciding number of clusters using hierarchical clustering. [Difficulty: **Intermediate/Advanced**]

**solution:** For each of the data sets, k = 4 seems the best number of clusters.

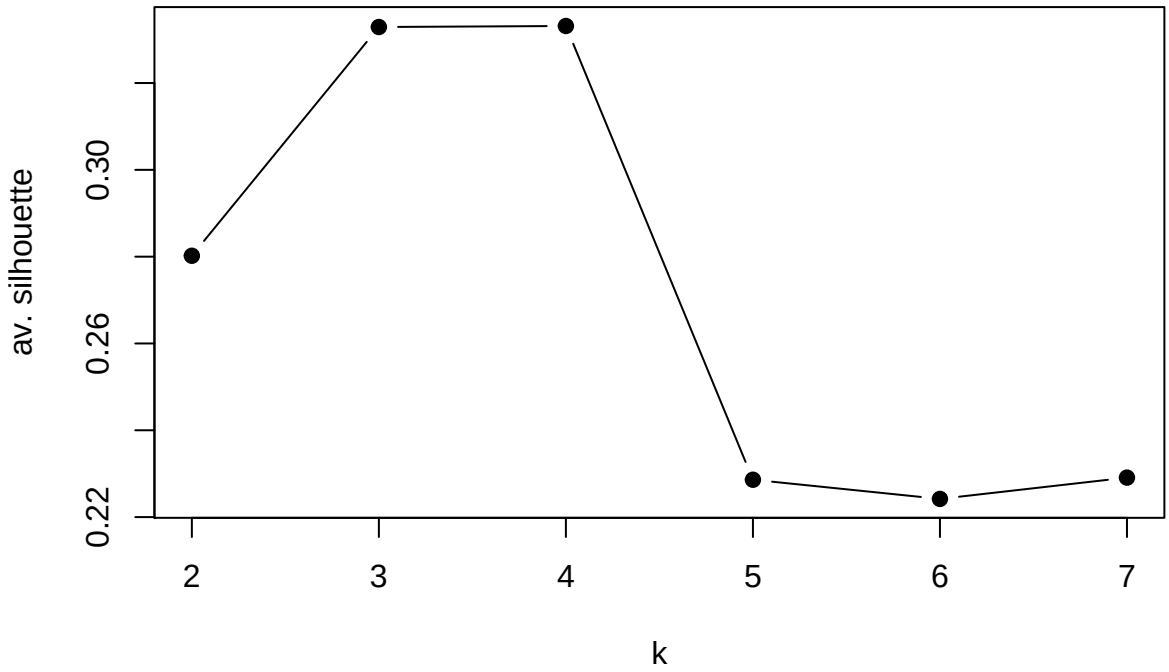
```
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 4.0.5
```

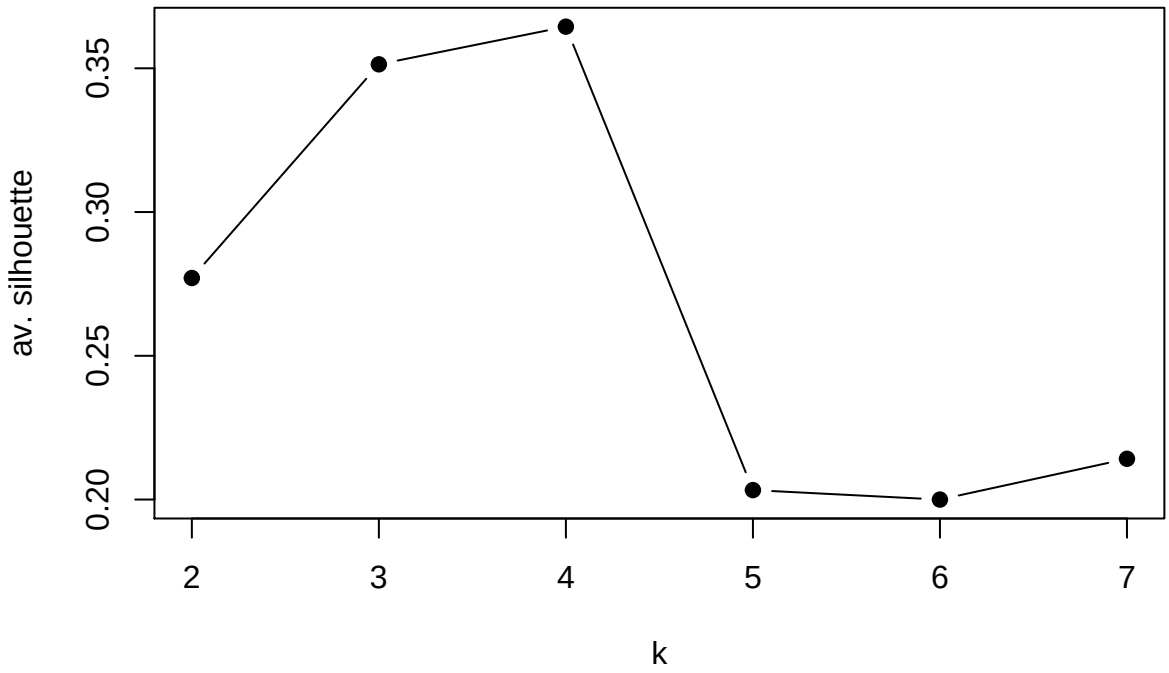
```
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")
mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

Ks=sapply(2:7,
    function(i)
      summary(silhouette(pam(t(mat),k=i)))$avg.width)
plot(2:7,Ks,xlab="k",ylab="av. silhouette",type="b",pch=19)
```



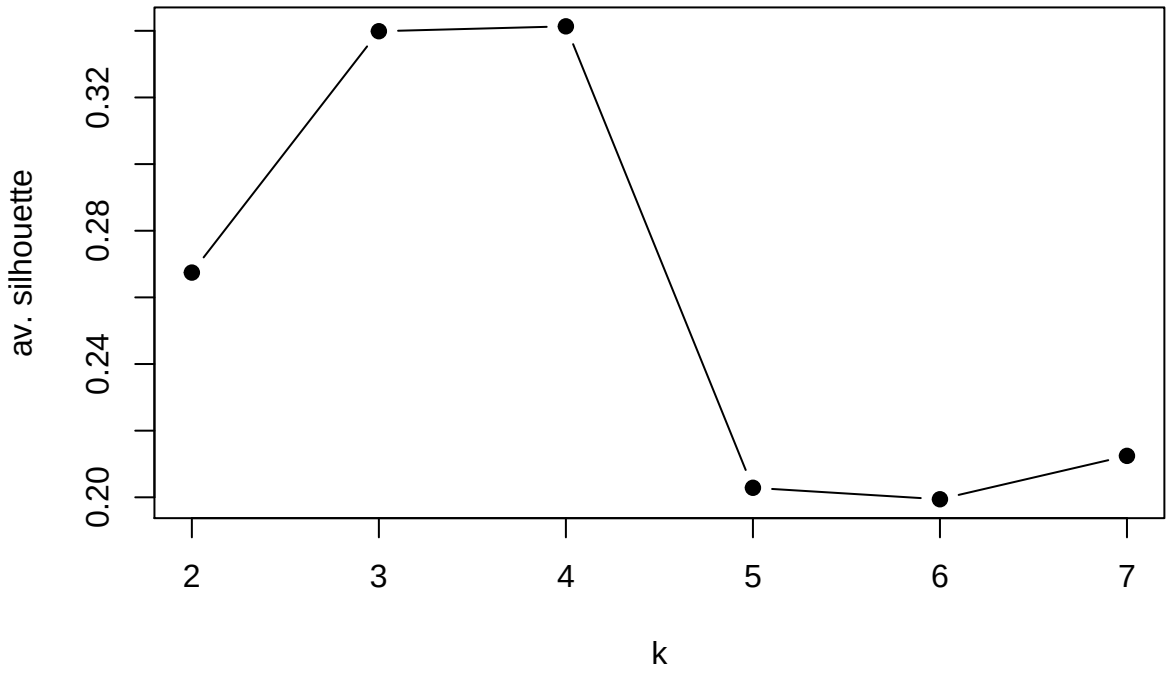

```
Ks=sapply(2:7,
    function(i)
      summary(silhouette(pam(t(mat_log2),k=i)))$avg.width)
plot(2:7,Ks,xlab="k",ylab="av. silhouette",type="b",pch=19)
```




```
Ks=sapply(2:7,
    function(i)
      summary(silhouette(pam(t(scaled_mat),k=i)))$avg.width)
plot(2:7,Ks,xlab="k",ylab="av. silhouette",type="b",pch=19)
```



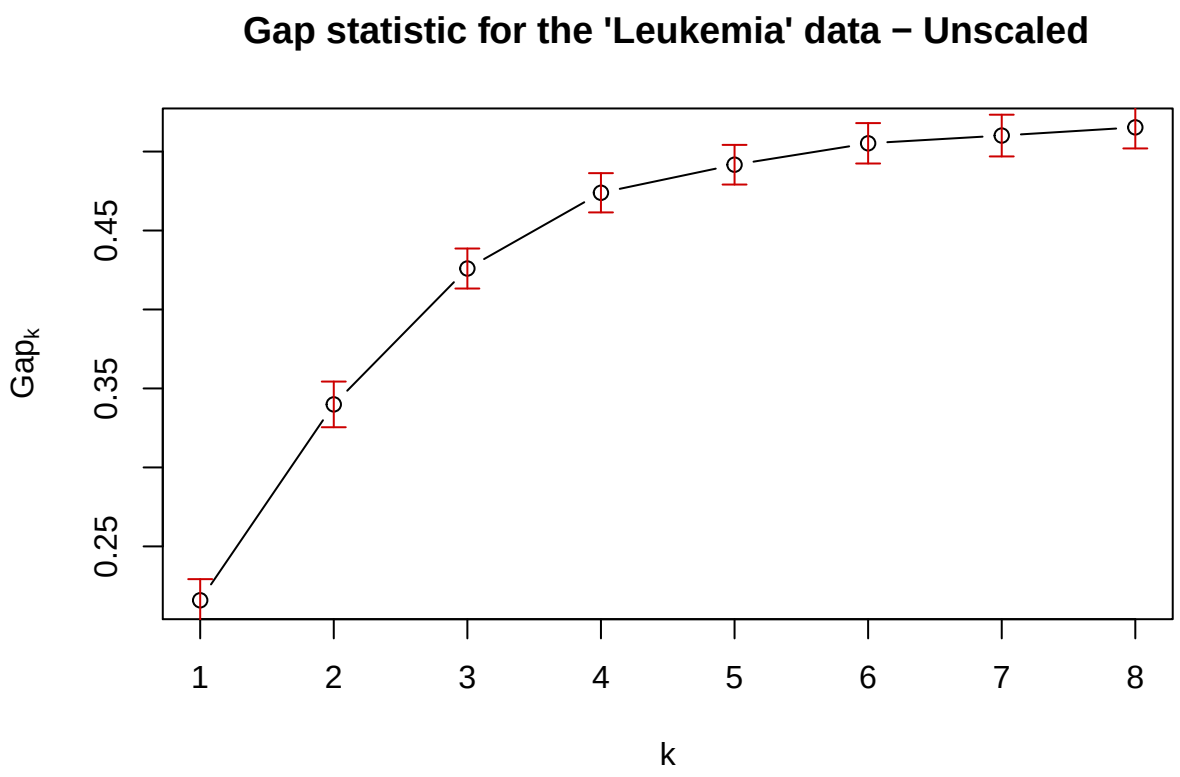

```
Ks=sapply(2:7,
    function(i)
      summary(silhouette(pam(t(scaled_mat_log2),k=i)))$avg.width)
plot(2:7,Ks,xlab="k",ylab="av. silhouette",type="b",pch=19)
```




4. Now, use the Gap Statistic for deciding the number of clusters in hierarchical clustering. Is the same number of clusters identified by two methods? Is it similar to the number of clusters obtained using the k-means algorithm in the unsupervised learning chapter. [Difficulty: **Intermediate/Advanced**]

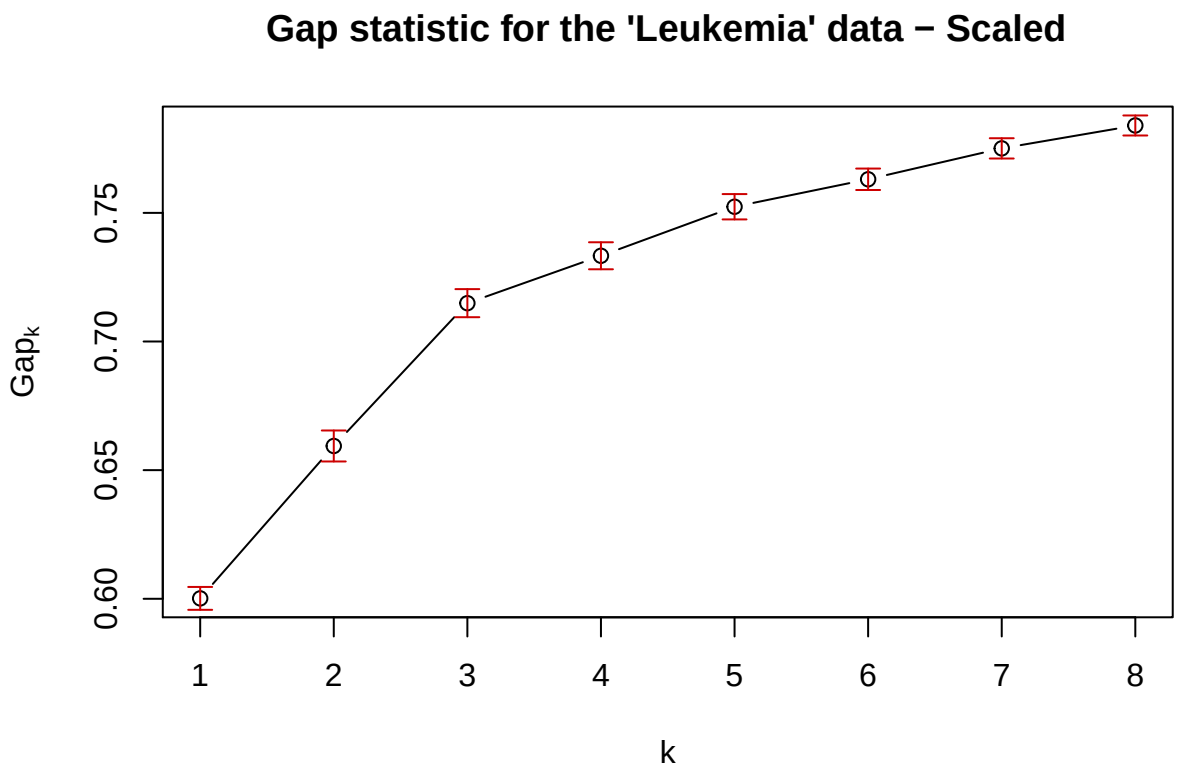
**solution:** The number of clusters identified are different in 2 methods. It is 8 in Hierarchial

```r
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")

mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

# define the clustering function
hclu <- function(x,k) {
  d=dist(x,method = "euclidean")
  hcl=hclust(d,method="ward.D")
  clu.k=cutree(hcl,k=k)
  list(cluster = clu.k)
}

par(mfrow=c(1,1))
# calculate the gap statistic for scaled and unscaled data
hclu.gap= clusGap(t(mat_log2), FUN = hclu, K.max = 8,B=50)

# plot the gap statistic accross k values
plot(hclu.gap, main = "Gap statistic for the 'Leukemia' data - Unscaled")
```




```r
# calculate the gap statistic for scaled and unscaled data
scaled.hclu.gap= clusGap(scaled_mat_log2, FUN = hclu, K.max = 8,B=50)
```

```
# plot the gap statistic accross k values
plot(scaled.hclu.gap, main = "Gap statistic for the 'Leukemia' data - Scaled")
```



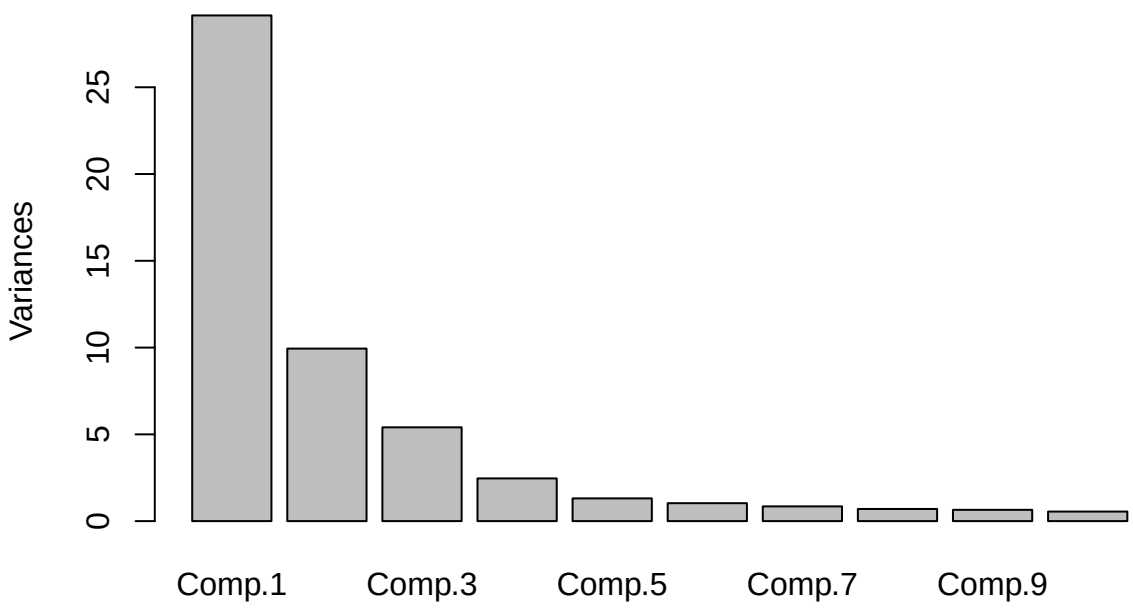

**Dimension reduction**

We will be using the leukemia expression data set again. You can use it as shown in the clustering exercises.

1. Do PCA on the expression matrix using the `princomp()` function and then use the `screeplot()` function to visualize the explained variation by eigenvectors. How many top components explain 95% of the variation? [Difficulty: **Beginner**]

**solution:** First 25 components explain 95.57% of the variation

```
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")

mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

pr = princomp(scale(mat))
screeplot(pr,
          npcs = 10,
          main = "Screeplot of the first 10 Principal Components")
```




```
summary(pr)
```

```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3     Comp.4     Comp.5
## Standard deviation     5.3980464 3.1529029 2.32472774 1.56870823 1.14416985
## Proportion of Variance 0.4861346 0.1658458 0.09016281 0.04105515 0.02184058
## Cumulative Proportion  0.4861346 0.6519803 0.74214316 0.78319830 0.80503889
##                           Comp.6     Comp.7     Comp.8     Comp.9     Comp.10
## Standard deviation     1.0163041 0.92085568 0.83618790 0.80637087 0.741512765
## Proportion of Variance 0.0172318 0.01414707 0.01166517 0.01084808 0.009173193
## Cumulative Proportion  0.8222707 0.83641776 0.84808292 0.85893101 0.868104199
##                           Comp.11     Comp.12    Comp.13     Comp.14
## Standard deviation     0.734978612 0.684352550 0.65828459 0.636403547
## Proportion of Variance 0.009012238 0.007813454 0.00722954 0.006756915
## Cumulative Proportion  0.877116437 0.884929891 0.89215943 0.898916345
##                           Comp.15     Comp.16     Comp.17    Comp.18
## Standard deviation     0.626302672 0.615372546 0.600370341 0.56906486
## Proportion of Variance 0.006544128 0.006317707 0.006013423 0.00540265
## Cumulative Proportion  0.905460473 0.911778180 0.917791603 0.92319425
##                           Comp.19     Comp.20     Comp.21     Comp.22
## Standard deviation     0.549322382 0.524753964 0.509814376 0.489873632
## Proportion of Variance 0.005034286 0.004594039 0.004336181 0.004003607
## Cumulative Proportion  0.928228538 0.932822578 0.937158759 0.941162365
##                           Comp.23     Comp.24     Comp.25    Comp.26
## Standard deviation     0.482433327 0.473364327 0.462804890 0.44960365
## Proportion of Variance 0.003882915 0.003738301 0.003573379 0.00337243
## Cumulative Proportion  0.945045280 0.948783581 0.952356961 0.95572939
##                           Comp.27     Comp.28     Comp.29    Comp.30
## Standard deviation     0.439205787 0.424763283 0.421822014 0.39135109
## Proportion of Variance 0.003218247 0.003010074 0.002968532 0.00255515
## Cumulative Proportion  0.958947638 0.961957712 0.964926244 0.96748139
```

```
##                            Comp.31     Comp.32     Comp.33     Comp.34
## Standard deviation     0.382994366 0.373072389 0.364965083 0.355318345
## Proportion of Variance 0.002447192 0.002322039 0.002222214 0.002106292
## Cumulative Proportion  0.969928586 0.972250625 0.974472839 0.976579130
##                            Comp.35     Comp.36     Comp.37     Comp.38
## Standard deviation     0.338241345 0.326528505 0.322846728 0.316324586
## Proportion of Variance 0.001908695 0.001778793 0.001738906 0.001669357
## Cumulative Proportion  0.978487826 0.980266619 0.982005525 0.983674882
##                            Comp.39     Comp.40     Comp.41     Comp.42
## Standard deviation     0.308522560 0.296659654 0.287414886 0.279974690
## Proportion of Variance 0.001588024 0.001468251 0.001378167 0.001307738
## Cumulative Proportion  0.985262906 0.986731156 0.988109323 0.989417061
##                            Comp.43     Comp.44      Comp.45      Comp.46
## Standard deviation     0.250015853 0.245318258 0.2348965297 0.2282002770
## Proportion of Variance 0.001042842 0.001004021 0.0009205269 0.0008687916
## Cumulative Proportion  0.990459903 0.991463925 0.9923844514 0.9932532430
##                             Comp.47      Comp.48      Comp.49      Comp.50
## Standard deviation     0.2246585841 0.2127630877 0.2052917511 0.1851652001
## Proportion of Variance 0.0008420334 0.0007552241 0.0007031148 0.0005720079
## Cumulative Proportion  0.9940952763 0.9948505004 0.9955536152 0.9961256231
##                            Comp.51      Comp.52      Comp.53      Comp.54
## Standard deviation     0.1791512892 0.1739143228 0.1657289040 0.1652361222
## Proportion of Variance 0.0005354552 0.0005046078 0.0004582261 0.0004555051
## Cumulative Proportion  0.9966610783 0.9971656861 0.9976239121 0.9980794173
##                            Comp.55      Comp.56      Comp.57      Comp.58
## Standard deviation     0.1570079504 0.1543267078 0.1381601383 0.1305610759
## Proportion of Variance 0.0004112695 0.0003973429 0.0003184555 0.0002843876
## Cumulative Proportion  0.9984906868 0.9988880297 0.9992064852 0.9994908728
##                            Comp.59      Comp.60
## Standard deviation     0.1283559255 0.1184982639
## Proportion of Variance 0.0002748623 0.0002342649
## Cumulative Proportion  0.9997657351 1.0000000000
```

2. Our next tasks are removing the eigenvectors and reconstructing the matrix using SVD, then we need to calculate the reconstruction error as the difference between the original and the reconstructed matrix. HINT: You have to use the `svd()` function and equalize eigenvalue to 0 for the component you want to remove. [Difficulty: **Intermediate/Advanced**]

```
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")

mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

svd_mat=svd(scale(mat)) #Take SVD
S =  diag(svd_mat$d) # S the diagonal matrix

#Reconstruct the matrix:
mat_reconstruct = svd_mat$u %*% S %*% t(svd_mat$v)
```

```
#Remove the 1st eigenvector
S[1,1] <- 0

#Reconstruct the matrix:
mat_reconstruct2 = svd_mat$u %*% S %*% t(svd_mat$v)

#Calculate the error:
e = mat_reconstruct2 - mat_reconstruct
```

3. Produce a 10-component ICA from the expression data set. Remove each component and measure the reconstruction error without that component. Rank the components by decreasing reconstruction-error. [Difficulty: **Advanced**]

**solution:** Removing the component 5 resulted the highest reconstruction error. The ranks is 5 9 8 3 7 6 2 4 10 1

```
library(fastICA)
expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")

mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

ica.res=fastICA(t(mat),n.comp=10)
ica.rescopy=ica.res

errors=c()
#Found that RMSE could be used here for the reconstruction error.

for(i in 1:10){
  ica.rescopy$S[,i]=0 #In each iteration, set the ith component to 0
  errors[i]=sqrt(mean((ica.rescopy$X-(ica.rescopy$S %*% ica.rescopy$A))^2)) #Calculate the error with R
  ica.rescopy=ica.res #Reset the copy for the next iteration
}
print(errors)
```

```
##  [1] 0.8377961 0.8251082 1.2700401 0.8629688 0.8117679 0.8091644 0.8670067
##  [8] 0.8597112 0.8642346 0.8001036
```

```
rank(-errors)
```

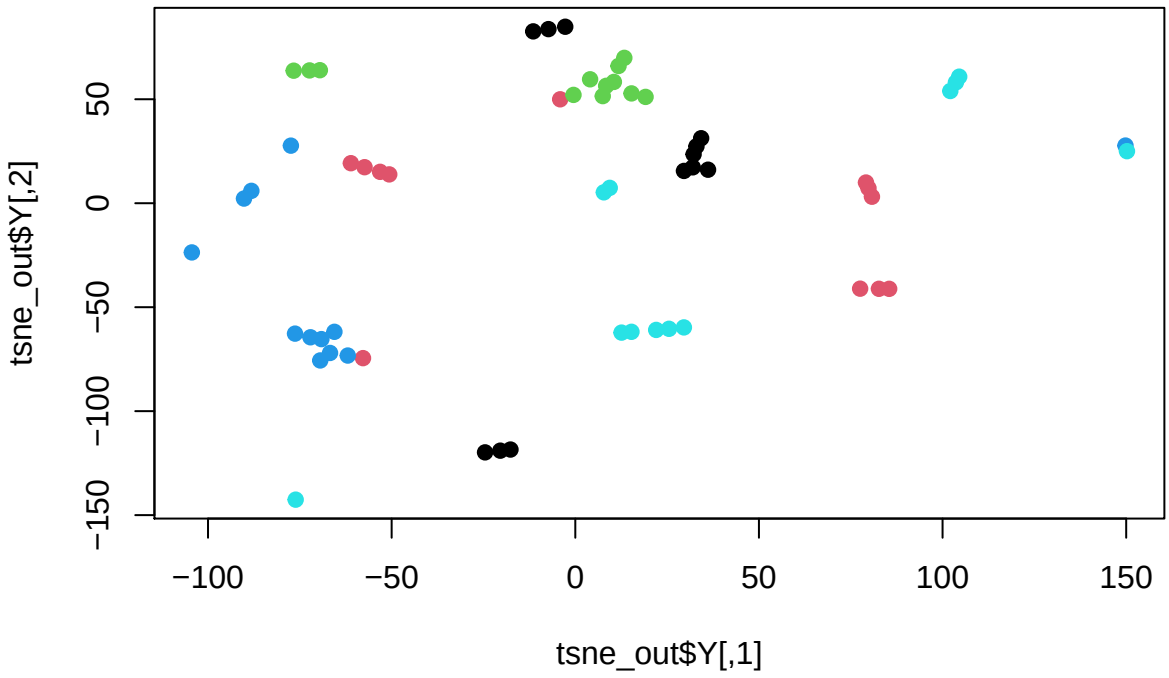
```
##  [1]  6  7  1  4  8  9  2  5  3 10
```

```
which.max(errors)
```

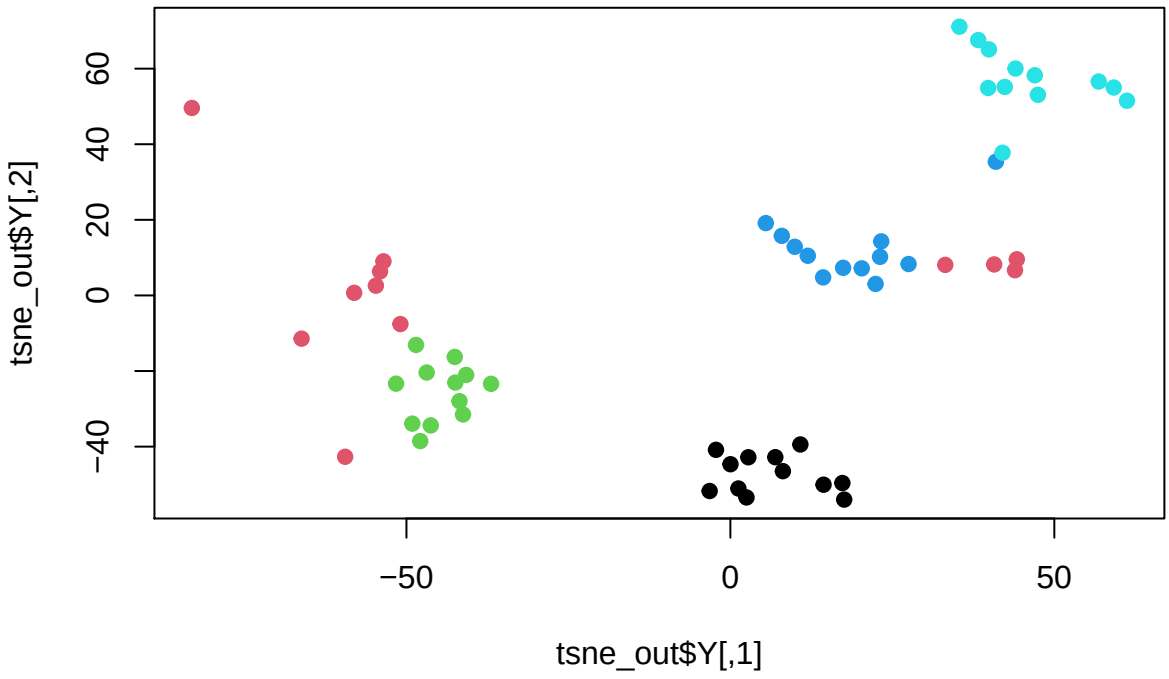
```
## [1] 3
```

4. In this exercise we use the `Rtsne()` function on the leukemia expression data set. Try to increase and decrease perplexity t-sne, and describe the observed changes in 2D plots. [Difficulty: **Beginner**] When the perplexity is above 10, the separation is clear between the samples.

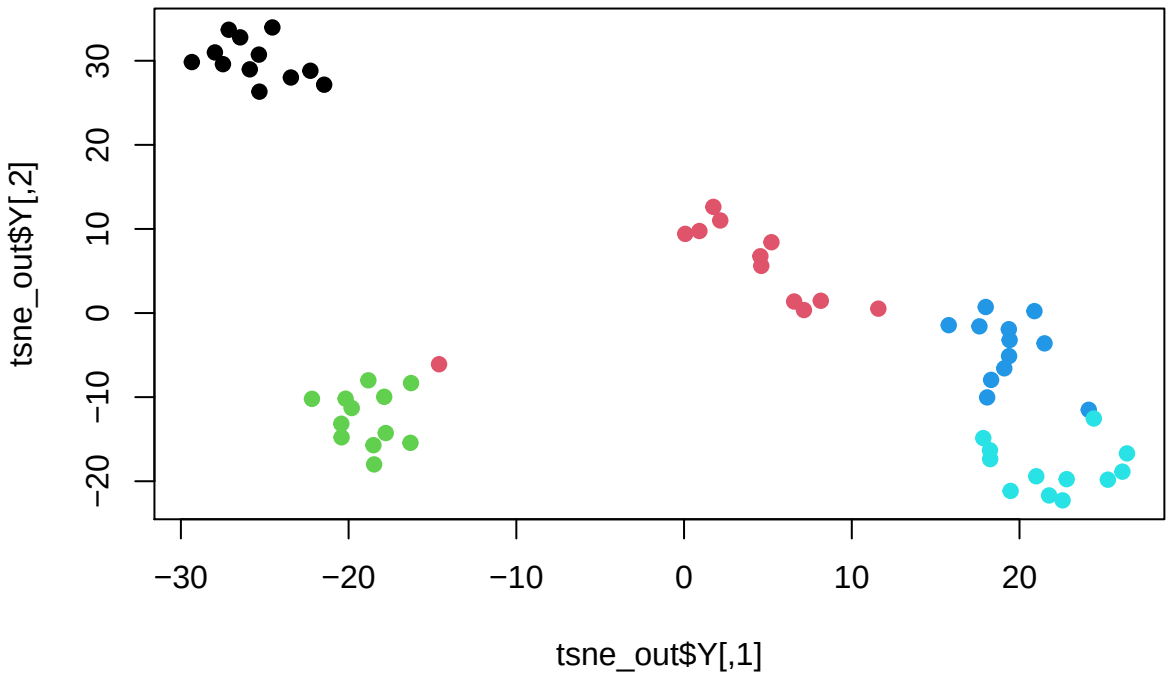
```
library(Rtsne)

expFile=system.file("extdata",
                    "leukemiaExpressionSubset.rds",
                    package="compGenomRData")
mat=readRDS(expFile)

annotation_col = data.frame(
                    LeukemiaType =substr(colnames(mat),1,3))
rownames(annotation_col)=colnames(mat)

annotation_col = data.frame(
  LeukemiaType =substr(colnames(mat),1,3))

tsne_out <- Rtsne(t(mat),perplexity = 1)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```



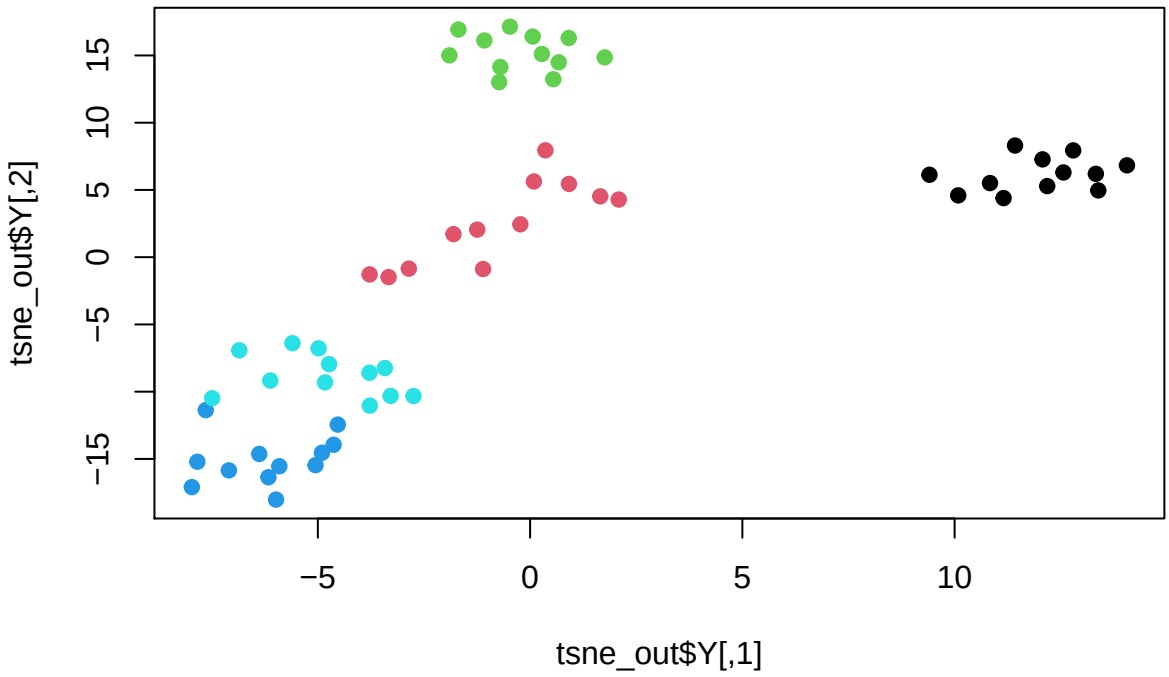

```
tsne_out <- Rtsne(t(mat),perplexity = 3)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```



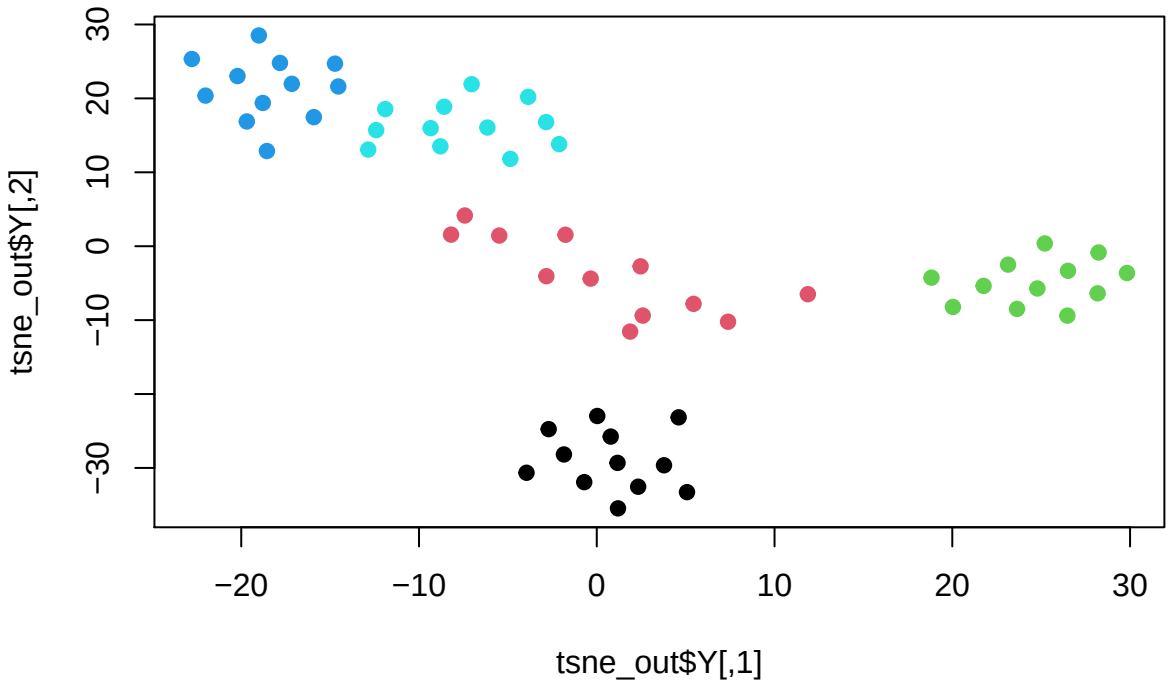

```
tsne_out <- Rtsne(t(mat),perplexity = 5)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```



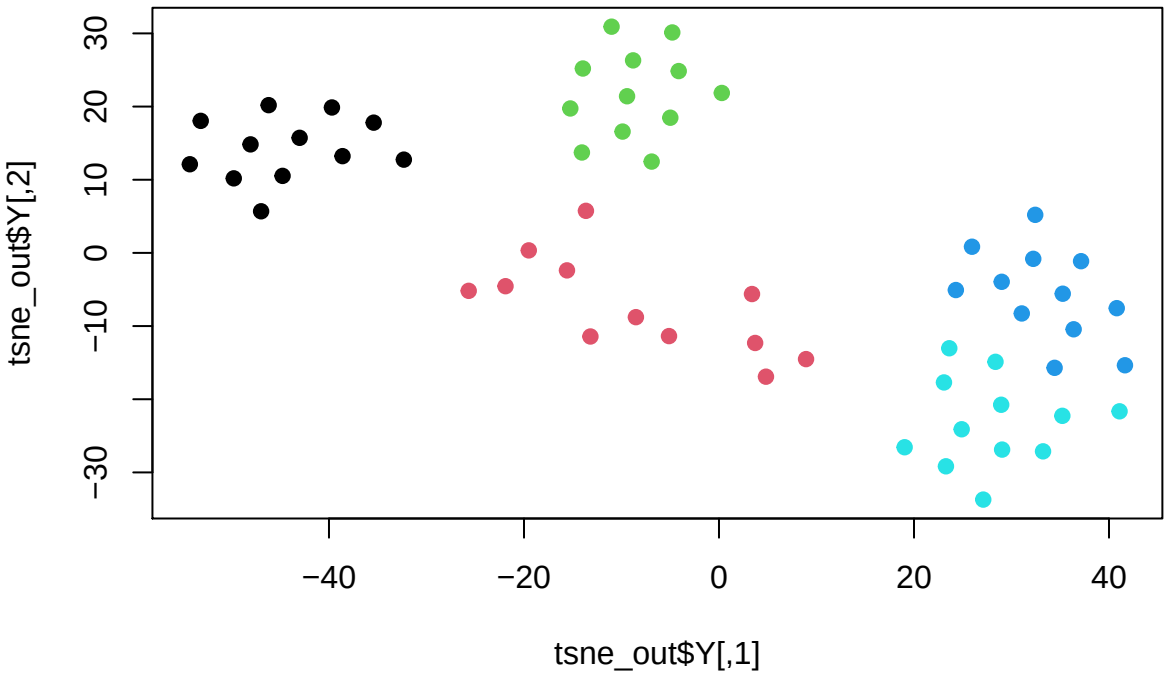

```
tsne_out <- Rtsne(t(mat),perplexity = 10)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```




```
tsne_out <- Rtsne(t(mat),perplexity = 15)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```




```
tsne_out <- Rtsne(t(mat),perplexity = 17)
plot(tsne_out$Y,col=as.factor(annotation_col$LeukemiaType),
     pch=19)
```


tsne_out$Y[,2]
30
20
10
0
−10
−30
−40
−20
0
20
40
tsne_out$Y[,1]