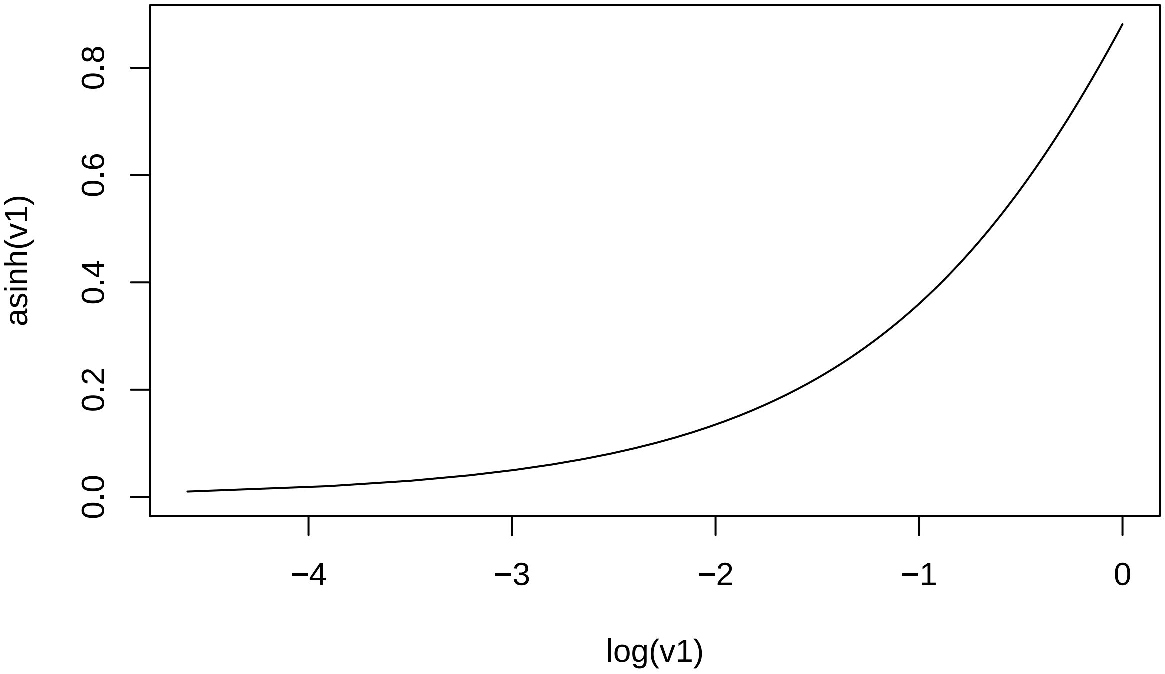
# MSMB-Chapter5-Clustering

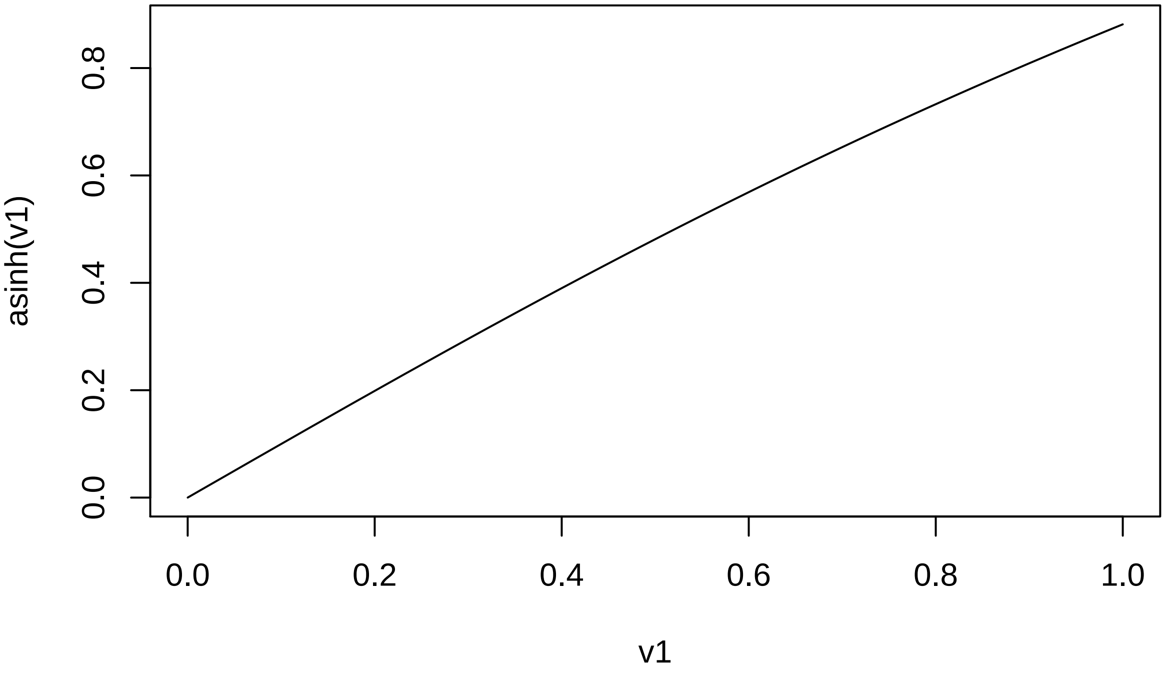
*Aleeza Gerstein*

*2019-11-06*

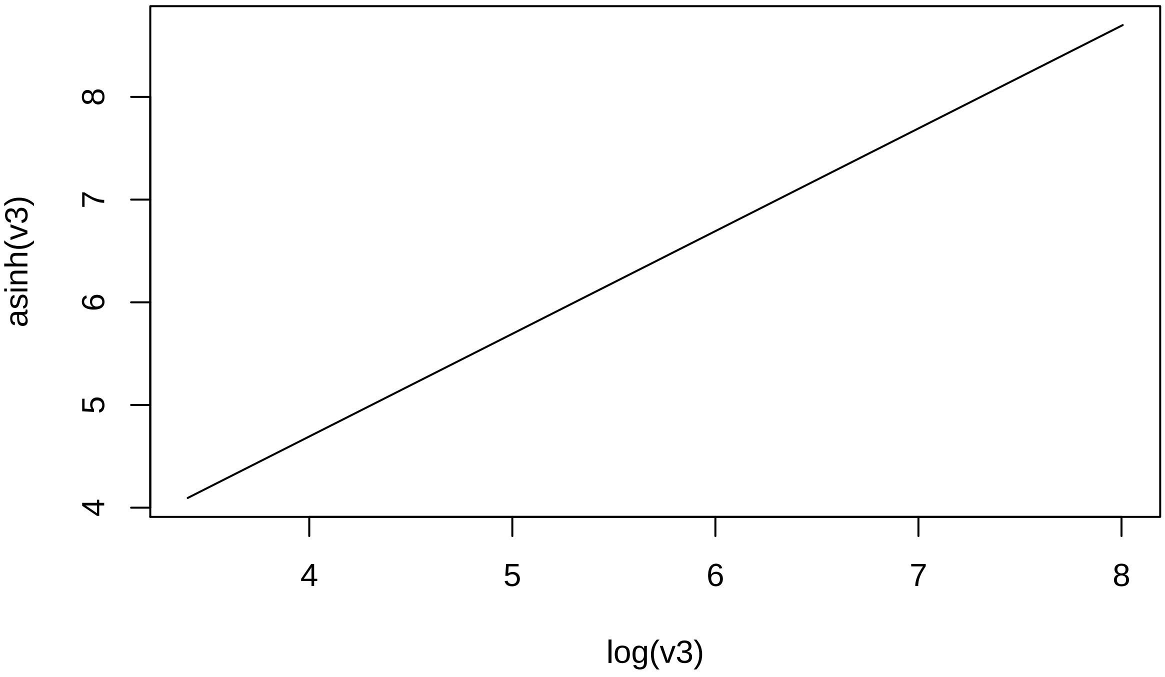
```
v1 <- seq(0, 1, length.out=100)
plot(log(v1), asinh(v1), type="l")
```



```
plot(v1, asinh(v1), type= "l")
```



```
v3 <- seq(30, 3000, length = 100)
plot(log(v3), asinh(v3), type="l")
```



## 5.5 Clustering examples: Flow cytometry

```
library("flowCore")
library("flowViz")
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following objects are masked from 'package:ncdfFlow':
##
##     densityplot, histogram, xyplot
```

```
fcsB <- read.FCS("../../data/Bendall_2011.fcs")
markersB <- readr::read_csv("../../data/Bendall_2011_markers.csv")
```

```
## Parsed with column specification:
## cols(
##   isotope = col_character(),
##   marker = col_character()
## )
```

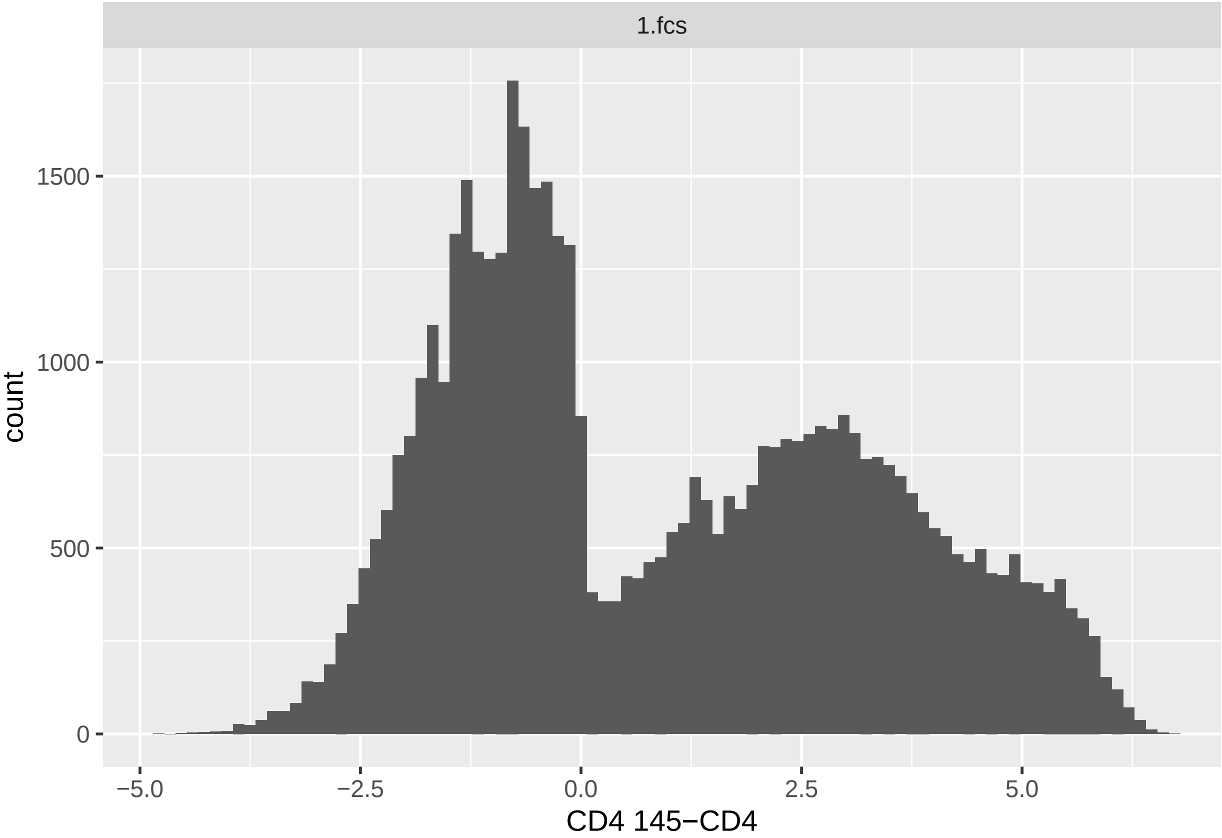
```
mt <- match(markersB$isotope, colnames(fcsB))
colnames(fcsB)[mt] <- markersB$marker

asinhtrsf <- arcsinhTransform(a = 0.1, b = 1)
fcsBT <- transform(fcsB, transformList(colnames(fcsB)[-c(1, 2, 4)], asinhtrsf))
```

```r
kf <- kmeansFilter("CD3all" = c("Pop1", "Pop2"), filterID = "myKmFilter")
fres <- flowCore::filter(fcsBT, kf)
summary(fres)
```

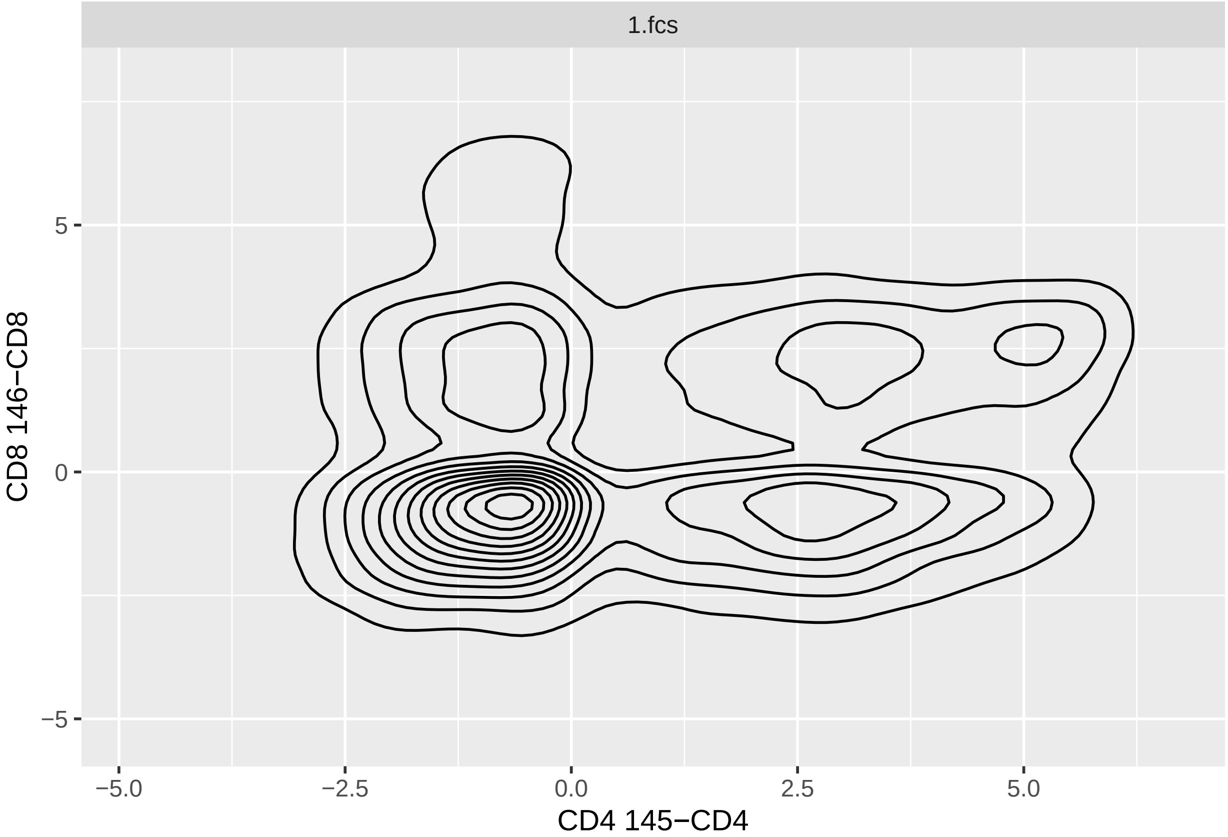
```
## Pop1: 33429 of 91392 events (36.58%)
## Pop2: 57963 of 91392 events (63.42%)
```

```r
fcsBT1 <- flowCore::split(fcsBT, fres, population = "Pop1")
fcsBT2 <- flowCore::split(fcsBT, fres, population = "Pop2")
```

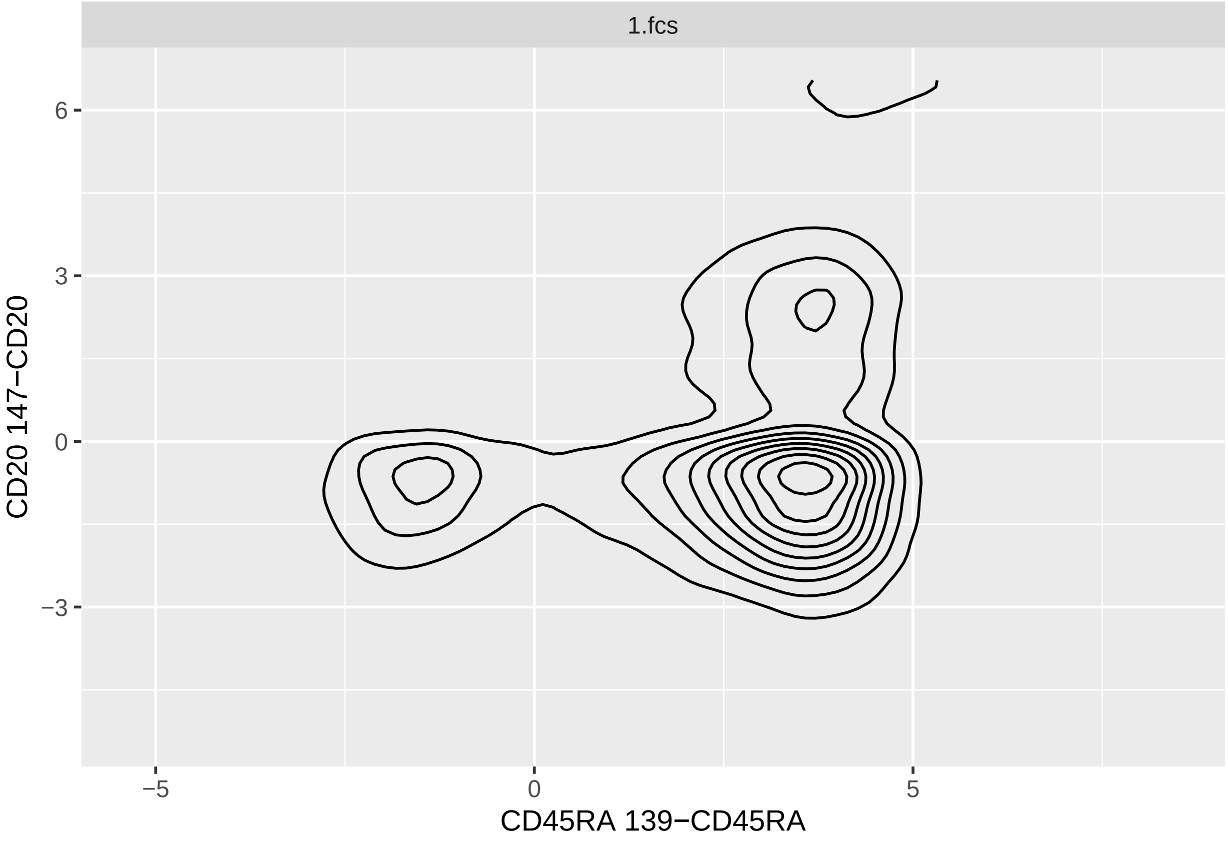
```r
library("ggcyto")
ggcd4cd8 <- ggcyto(fcsB, aes(x = CD4, y = CD8))
ggcd4 <- ggcyto(fcsB, aes(x = CD4))
ggcd8 <- ggcyto(fcsB, aes(x = CD8))
p1 <- ggcd4 +
  geom_histogram(bins = 60)
p1b <- ggcd8 +
  geom_histogram(bins = 60)
asinht <- arcsinhTransform(a = 0, b =1)
trans1 <- transformList(colnames(fcsB)[-c(1, 2, 4)], asinht)
fcsBT <- transform(fcsB, trans1)
p1t <- ggcyto(fcsBT, aes(x=CD4)) +
  geom_histogram(bins = 90)
p1t
```



```r
p2t <- ggcyto(fcsBT, aes(x=CD4, y = CD8)) +
  geom_density2d(colour="black")
p2t
```



```
p3t <- ggcyto(fcsBT, aes(x = CD45RA, y = CD20)) +
  geom_density2d(colour = "black")
p3t
```



### 5.5.3 Density-based clustering

```
mc5 <- exprs(fcsBT)[,c(15, 16, 19, 40, 33)] #why this order?
res5 <- dbscan::dbscan(mc5, eps = 0.65, minPts = 30)
mc5df <- data.frame(mc5, cluster = as.factor(res5$cluster))
table(mc5df$cluster)
```

```
##
##     0     1     2     3     4     5     6     7     8
## 75954  4031  5450  5310   259   257    63    25    43
```

```
mc6 <- exprs(fcsBT)[,c(15, 16, 19, 40, 25, 33)] #why this order?
res6 <- dbscan::dbscan(mc6, eps = 0.65, minPts = 20)
mc6df <- data.frame(mc6, cluster = as.factor(res6$cluster))
table(mc6df$cluster)
```

```
##
##     0     1     2     3     4     5     6
## 91068    34    61    20    67   121    21
```

```
res6_b <- dbscan::dbscan(mc6, eps = 0.45, minPts = 20)
mc6df_b <- data.frame(mc6, cluster = as.factor(res6_b$cluster))
table(mc6df_b$cluster)
```

```
##
##     0
## 91392
```

```
res6_c <- dbscan::dbscan(mc6, eps = 0.75, minPts = 20)
mc6df_c <- data.frame(mc6, cluster = as.factor(res6_c$cluster))
table(mc6df_c$cluster)
```

```
##
##     0     1     2     3     4     5     6     7     8     9    10    11
## 88650   412   384   733   170   599    20   167   143    27    28    16
##    12    13
##    23    20
```

Question 5.8

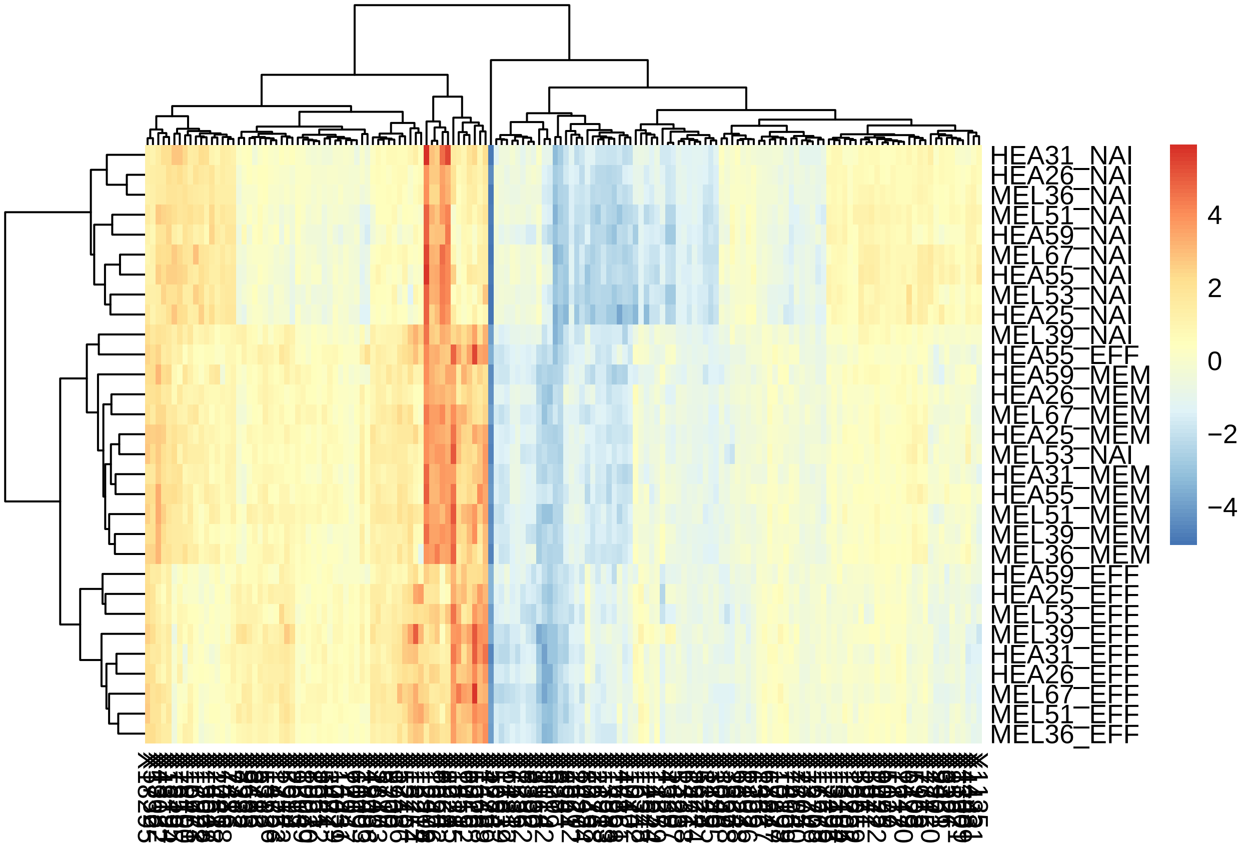
```
load("../../data/Morder.RData")
#without dendogram or reordering, Euclidean and Manhattan distances

library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

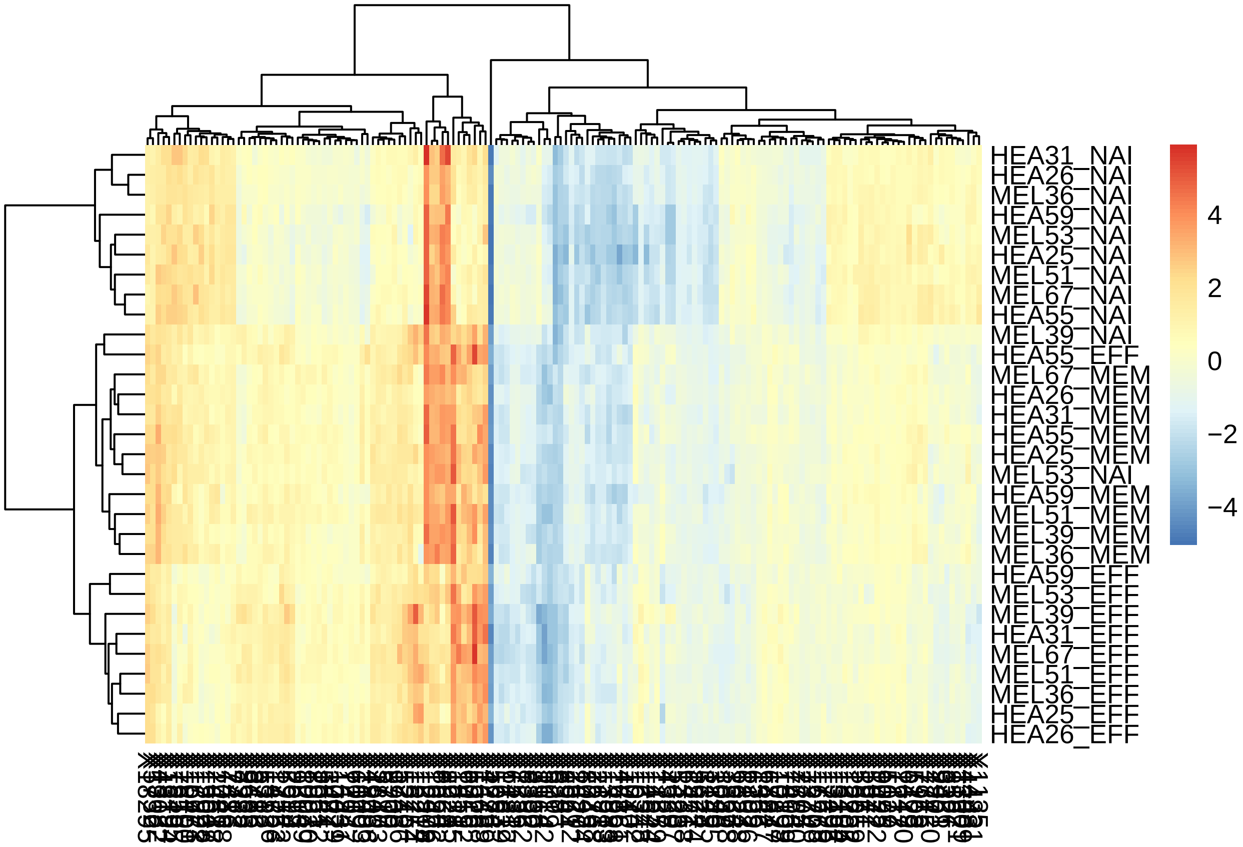
```
library(grid)
```

```
##
## Attaching package: 'grid'

## The following object is masked from 'package:mixtools':
##
##     depth
```

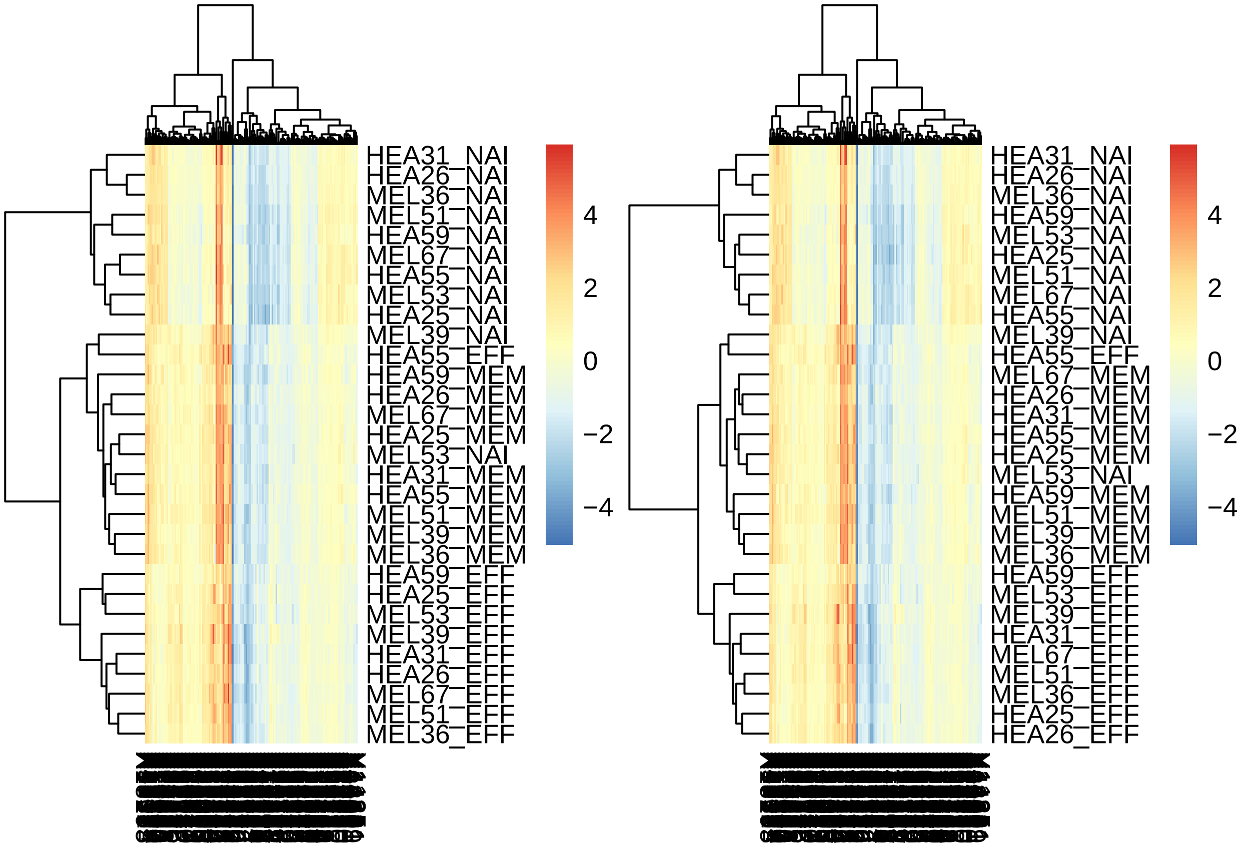
```
#I want to plot both on the same panel
#two ways of doing it
#https://www.biostars.org/p/128229/
#https://stackoverflow.com/questions/39590849/using-a-pheatmap-in-arrangegrob

plot_list <- list()
ph1 <- pheatmap(Morder)
```



```
ph2 <- pheatmap(Morder, clustering_distance_rows = "manhattan")
```



```
plot_list[[1]] <- ph1[[4]]
plot_list[[2]] <- ph2[[4]]

g <- grid.arrange(arrangeGrob(grobs= plot_list,ncol=2))
```



```
#g<-do.call(grid.arrange,plot_list)

#Question 5.9: which orderings do not match
#https://www.biostars.org/p/170614/
res1 <- Morder[c(ph1$tree_row[["order"]]),ph1$tree_col[["order"]]]
res2 <- Morder[c(ph2$tree_row[["order"]]),ph2$tree_col[["order"]]]
row.names(res1) == row.names(res2)
```

```
##  [1]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
## [12] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

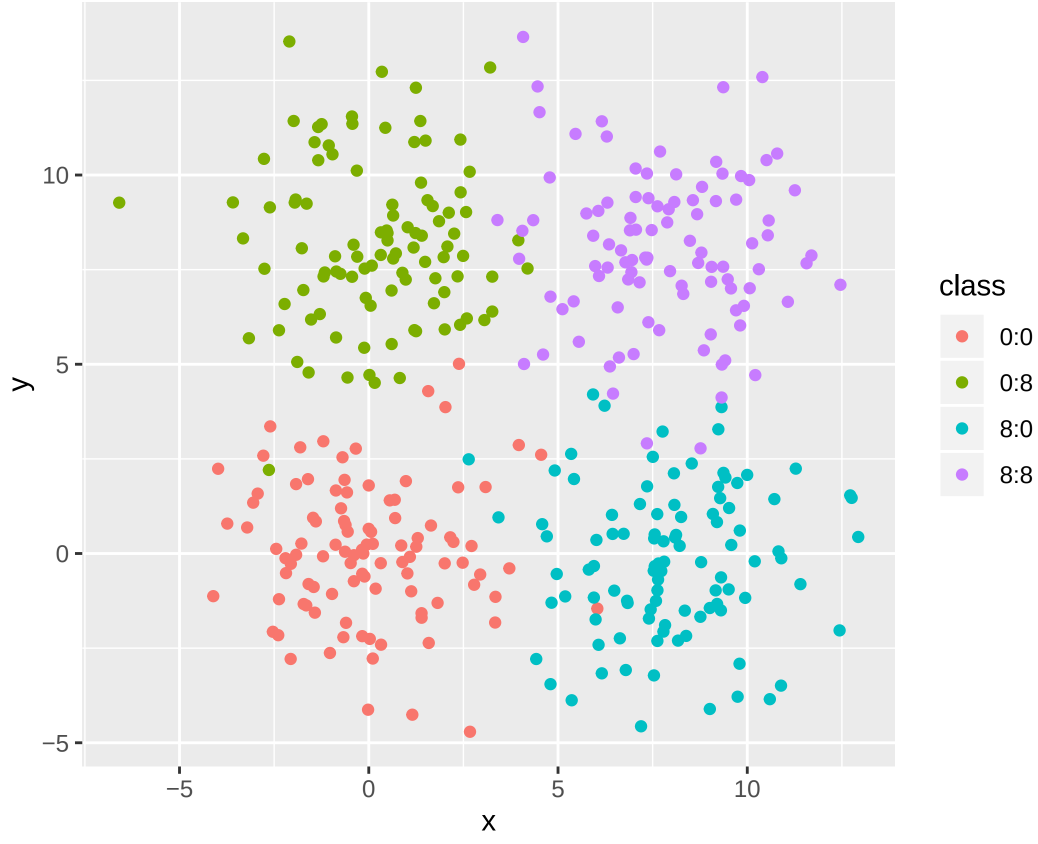
## here

Question 5.12

```
library(tidyverse)
simdat <- lapply(c(0, 8), function(mx){
  lapply(c(0, 8), function(my){
    tibble(x = rnorm(100, mean = mx, sd = 2),
           y = rnorm(100, mean = my, sd = 2),
           class = paste(mx, my, sep=":"))
  }) %>% bind_rows
}) %>% bind_rows
simdat
```

```
## # A tibble: 400 x 3
##         x       y class
##     <dbl>   <dbl> <chr>
##  1 -1.20   2.97   0:0
##  2 -2.53  -2.07   0:0
##  3 -2.79   2.58   0:0
##  4 -3.74   0.791  0:0
##  5  3.71  -0.392  0:0
##  6 -0.575  1.61   0:0
##  7  1.26   0.180  0:0
##  8  2.94  -0.554  0:0
##  9 -1.92  -0.0333 0:0
## 10 -0.172 -0.535  0:0
## # ... with 390 more rows
```

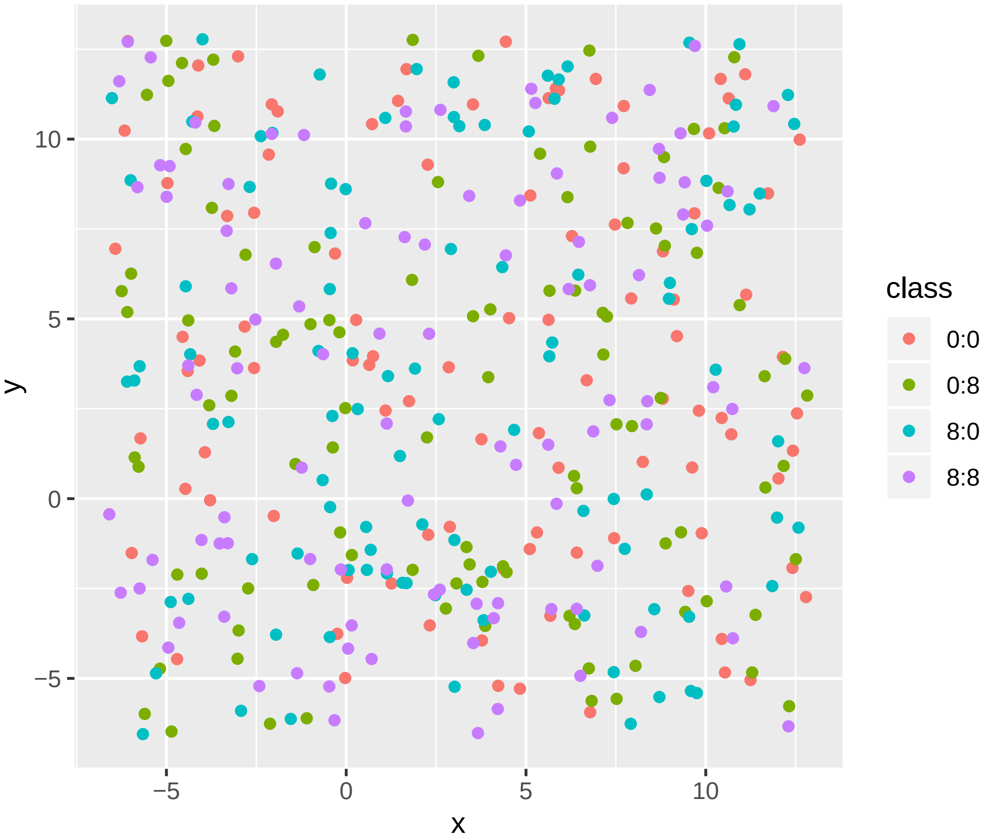
```
simdatxy <- simdat[, c("x", "y")]
ggplot(simdat, aes(x = x, y = y, col=class))+
  geom_point()+
  coord_fixed()
```



```
rsim1 <- range(c(simdat$x[1:100], simdat$y[1:100]))
rsim2 <- range(c(simdat$x[101:200], simdat$y[101:200]))
rsim3 <- range(c(simdat$x[201:300], simdat$y[201:300]))
rsim4 <- range(c(simdat$x[301:400], simdat$y[301:400]))

simdat_un <- lapply(c(0, 8), function(mx){
  lapply(c(0, 8), function(my){
    tibble(x = runif(100, range(simdat$x)[1], range(simdat$x)[2]),
           y = runif(100, range(simdat$x)[1], range(simdat$x)[2]),
           class = paste(mx, my, sep=":"))
  }) %>% bind_rows
}) %>% bind_rows
simdat_un
```

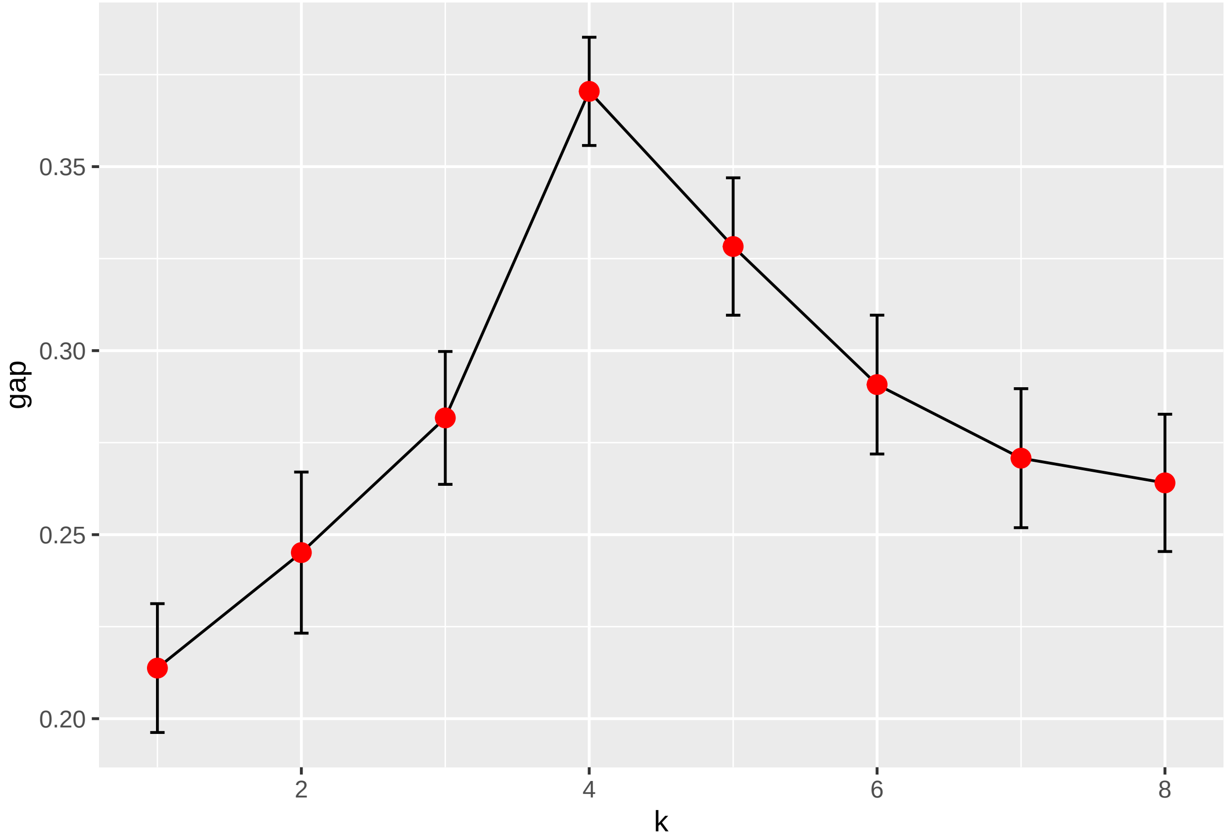
```
## # A tibble: 400 x 3
##          x      y class
##      <dbl>  <dbl> <chr>
##   1  5.31  -0.941 0:0
##   2  8.25   1.02  0:0
##   3 12.6    9.99  0:0
##   4 11.7    8.49  0:0
##   5  0.276  4.97  0:0
##   6 -4.55   4.50  0:0
##   7  5.12   8.43  0:0
##   8 11.1   11.8   0:0
##   9 -1.91  10.8   0:0
##  10  7.45  -1.10  0:0
## # ... with 390 more rows
```

```
ggplot(simdat_un, aes(x = x, y = y, col=class))+
  geom_point()+
  coord_fixed()
```



```
library(cluster)

pamfun <- function(x, k)list(cluster=pam(x, k, cluster.only = TRUE))
```

```
gss <- clusGap(simdatxy, FUN = pamfun, K.max = 8, B = 50, verbose = FALSE)

plot_gap <- function(x){
  gstab <- data.frame(x$Tab, k = seq_len(nrow(x$Tab)))
  ggplot(gstab, aes(k, gap)) +
    geom_line()+
    geom_errorbar(aes(ymax = gap + SE.sim,
                      ymin = gap - SE.sim), width = 0.1) +
    geom_point(size = 3, col = "red")
}

plot_gap(gss)
```



```
library("Hiiragi2013")
```

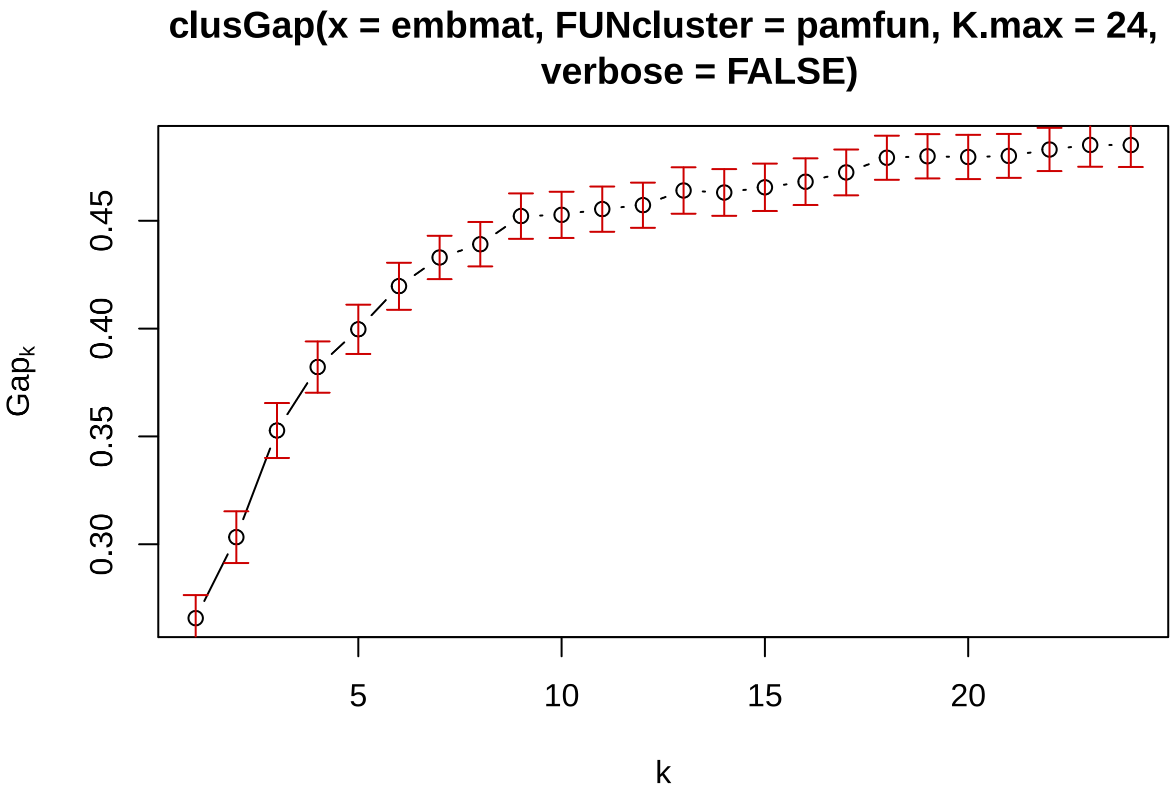
```
## Loading required package: affy

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:flowCore':
##
##     normalize

## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter,
##     Find, get, grep, grepl, intersect, is.unsorted, lapply, Map,
##     mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unsplit, which,
##     which.max, which.min

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

## Loading required package: boot

##
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':
##
##     melanoma

## Loading required package: clue

## Loading required package: genefilter

##
## Attaching package: 'genefilter'

## The following object is masked from 'package:readr':
##
##     spec

## Loading required package: geneplotter

## Loading required package: annotate

## Loading required package: AnnotationDbi

## Loading required package: stats4

## Loading required package: IRanges
## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:tidyr':
##
##     expand

## The following objects are masked from 'package:dplyr':
##
##     first, rename

## The following object is masked from 'package:base':
##
##     expand.grid

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:purrr':
##
##     reduce

## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice

##
## Attaching package: 'AnnotationDbi'

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: XML

##
## Attaching package: 'annotate'

## The following object is masked from 'package:flowCore':
##
##     journal

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:IRanges':
##
##     space

## The following object is masked from 'package:S4Vectors':
##
##     space

## The following object is masked from 'package:stats':
##
##     lowess

## Loading required package: gtools
```

```
##
## Attaching package: 'gtools'

## The following objects are masked from 'package:boot':
##
##     inv.logit, logit

## The following object is masked from 'package:mixtools':
##
##     ddirichlet

## Loading required package: KEGGREST

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:AnnotationDbi':
##
##     select

## The following object is masked from 'package:genefilter':
##
##     area

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: mouse4302.db

## Loading required package: org.Mm.eg.db

##

##

## Loading required package: RColorBrewer

## Loading required package: xtable
```

```
data("x")

selFeats = order(rowVars(Biobase::exprs(x)), decreasing = TRUE)[1:50]
embmat = t(Biobase::exprs(x)[selFeats, ])
embgap = clusGap(embmat, FUN = pamfun, K.max = 24, verbose = FALSE)
k1 = maxSE(embgap$Tab[, "gap"], embgap$Tab[, "SE.sim"])
k2 = maxSE(embgap$Tab[, "gap"], embgap$Tab[, "SE.sim"],
           method = "Tibs2001SEmax")
c(k1, k2)
```

```
## [1] 11  7
```

```
plot(embgap)
```



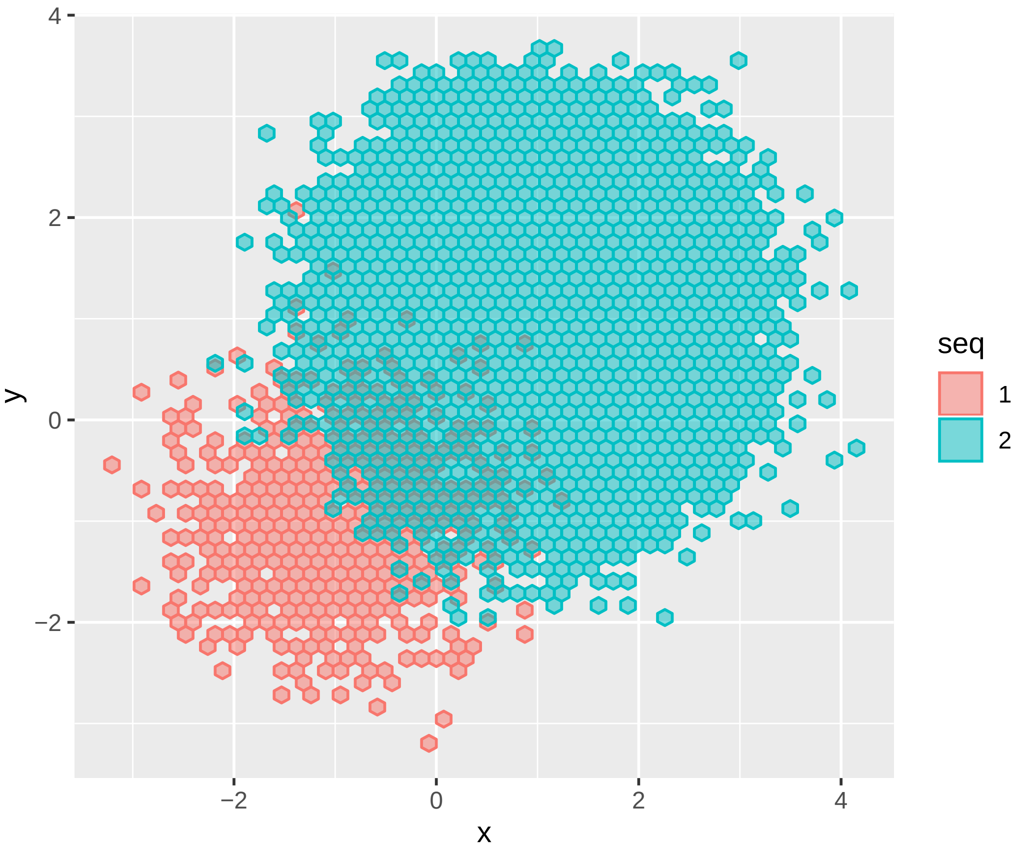
Question 5.15 Use all features in x #times out

```
embmat_full = t(Biobase::exprs(x))
embgap_full = clusGap(embmat_full, FUN = pamfun, K.max = 24, verbose = FALSE)
k1 = maxSE(embgap_full$Tab[, "gap"], embgap_full$Tab[, "SE.sim"])
k2 = maxSE(embgap_full$Tab[, "gap"], embgap_full$Tab[, "SE.sim"],
           method = "Tibs2001SEmax")
c(k1, k2)

plot(embgap)
```

```
clusterResampling = function(x, ngenes = 50, k = 2, B = 250,
                             prob = 0.67) {
  mat = Biobase::exprs(x)
  #draw a random resampling of 67% of the data without replacement
  #repeat B times
  ce = cl_ensemble(list = lapply(seq_len(B), function(b) {
    selSamps = sample(ncol(mat), size = round(prob * ncol(mat)),
                      replace = FALSE)
    submat = mat[, selSamps, drop = FALSE]
    #select the top n genes features by overall variance in the subset
    sel = order(rowVars(submat), decreasing = TRUE)[seq_len(ngenes)]
    submat = submat[sel,, drop = FALSE]
    #apply kmeans clustering
    pamres = pam(t(submat), k = k)
    #predict cluster memberships of the samples that were not in the subset with the cl_predict method
    pred = cl_predict(pamres, t(mat[sel, ]), "memberships")
    as.cl_partition(pred)
  }))
  #for each of B clusterings, measure the agreement with the consensus through the function cl_agreemen
  cons = cl_consensus(ce)
  ag = sapply(ce, cl_agreement, y = cons)
  list(agreements = ag, consensus = cons)
}

iswt <- (x$genotype == "WT")
cr1 <- clusterResampling(x[, x$Embryonic.day == "E3.25" & iswt])
cr2 <- clusterResampling(x[, x$Embryonic.day == "E3.5"  & iswt])
```

### 5.8 Clustering as a means for denoising

```
library("mixtools")
library("ggplot2")
seq1 = rmvnorm(n = 1e3, mu = -c(1, 1), sigma = 0.5 * diag(c(1, 1)))
seq2 = rmvnorm(n = 1e5, mu =  c(1, 1), sigma = 0.5 * diag(c(1, 1)))
twogr = data.frame(
  rbind(seq1, seq2),
  seq = factor(c(rep(1, nrow(seq1)),
                 rep(2, nrow(seq2))))
)
colnames(twogr)[1:2] = c("x", "y")
ggplot(twogr, aes(x = x, y = y, colour = seq,fill = seq)) +
  geom_hex(alpha = 0.5, bins = 50) + coord_fixed()
```



Question 5.16: Take the data seq1 and seq2 and cluster them into two groups according to distance from the

group center

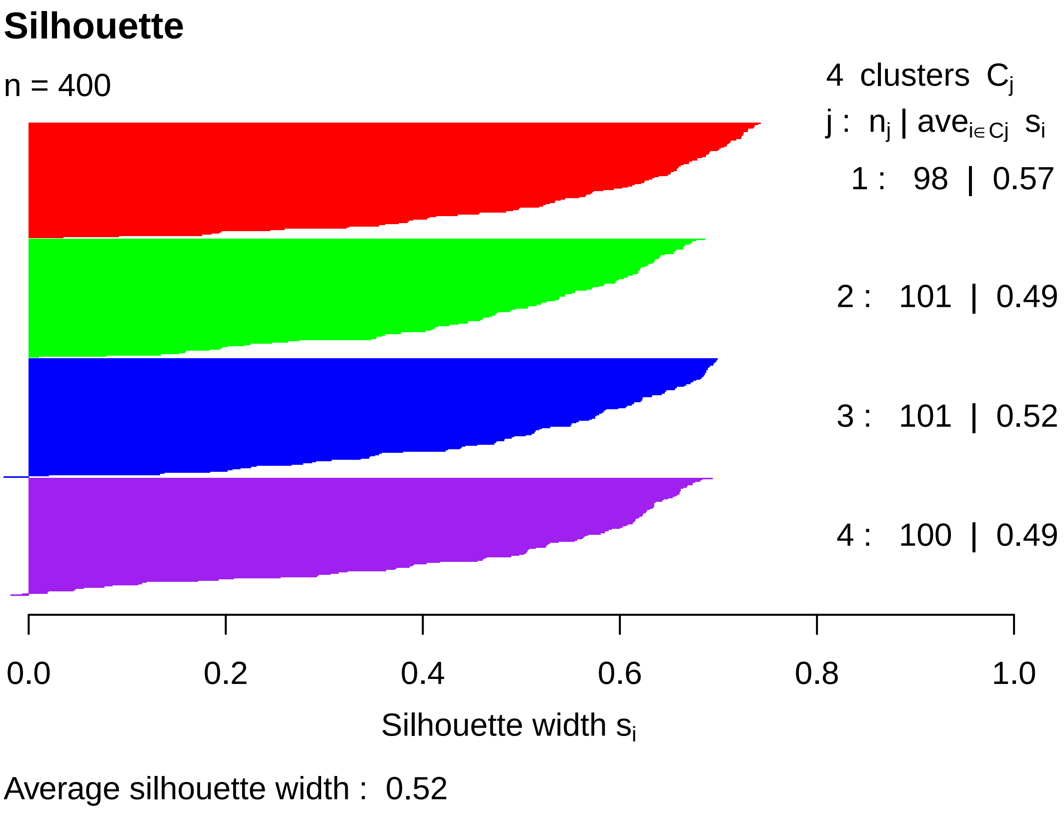
### Exercises

Exercise 5.1

```
library("cluster")
library("viridis")
```

```
## Loading required package: viridisLite
```

```
pam4 <- pam(simdatxy, 4)
sil <- silhouette(pam4, 8)
plot(sil, col=c("red", "green", "blue", "purple"), main = "Silhouette")
```

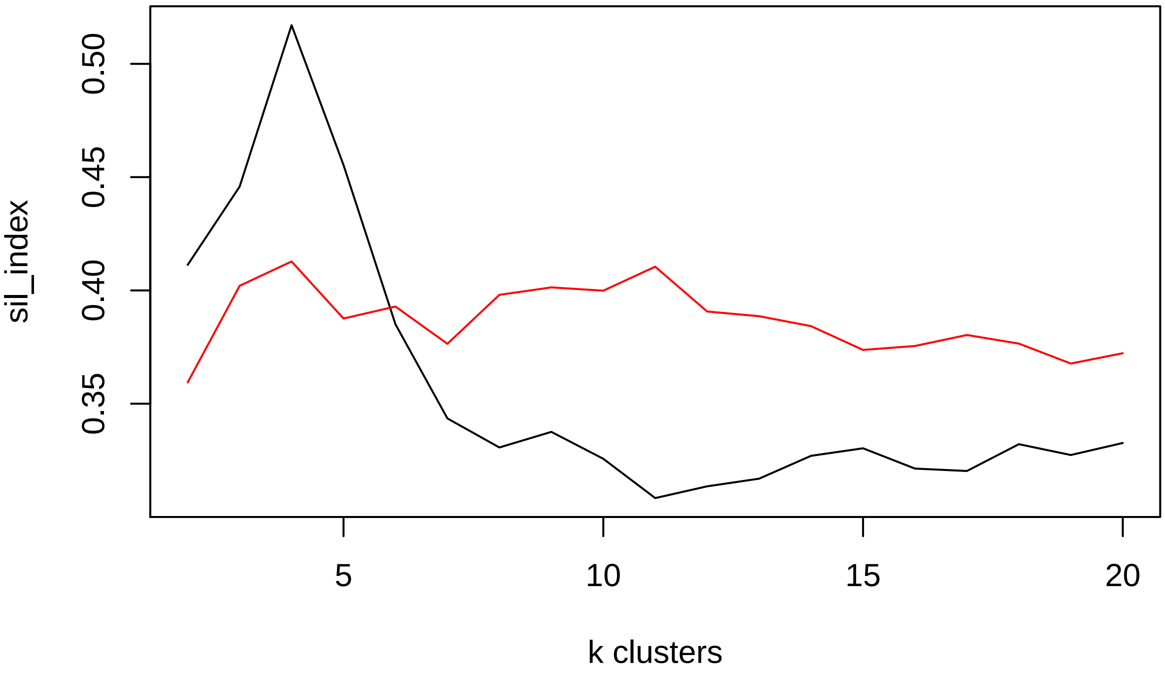


```
# coloursV <- viridis(10, direction=-1)
# for(k in 2:10)
#      plot(silhouette(pam(simdatxy, k=k)), main = paste("k = ",k), do.n.k=FALSE,
#           col = coloursV[1:k])

sil_index <- c()
for(k in 2:20){
  pam4 <- pam(simdatxy, k)
  sil <- silhouette(pam4, k)
  sil_index[k-1] <- mean(sil[,3])
}
```

```
sil_index_un <- c()
for(k in 2:20){
  pam4 <- pam(simdat_un, k)
  sil <- silhouette(pam4, k)
  sil_index_un[k-1] <- mean(sil[,3])
}
```

```
## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion

## Warning in data.matrix(x): NAs introduced by coercion
```

```
plot(2:20, sil_index, type="l", xlab = "k clusters")
points(2:20, sil_index_un, type="l", col="red")
```



Exercise 5.2

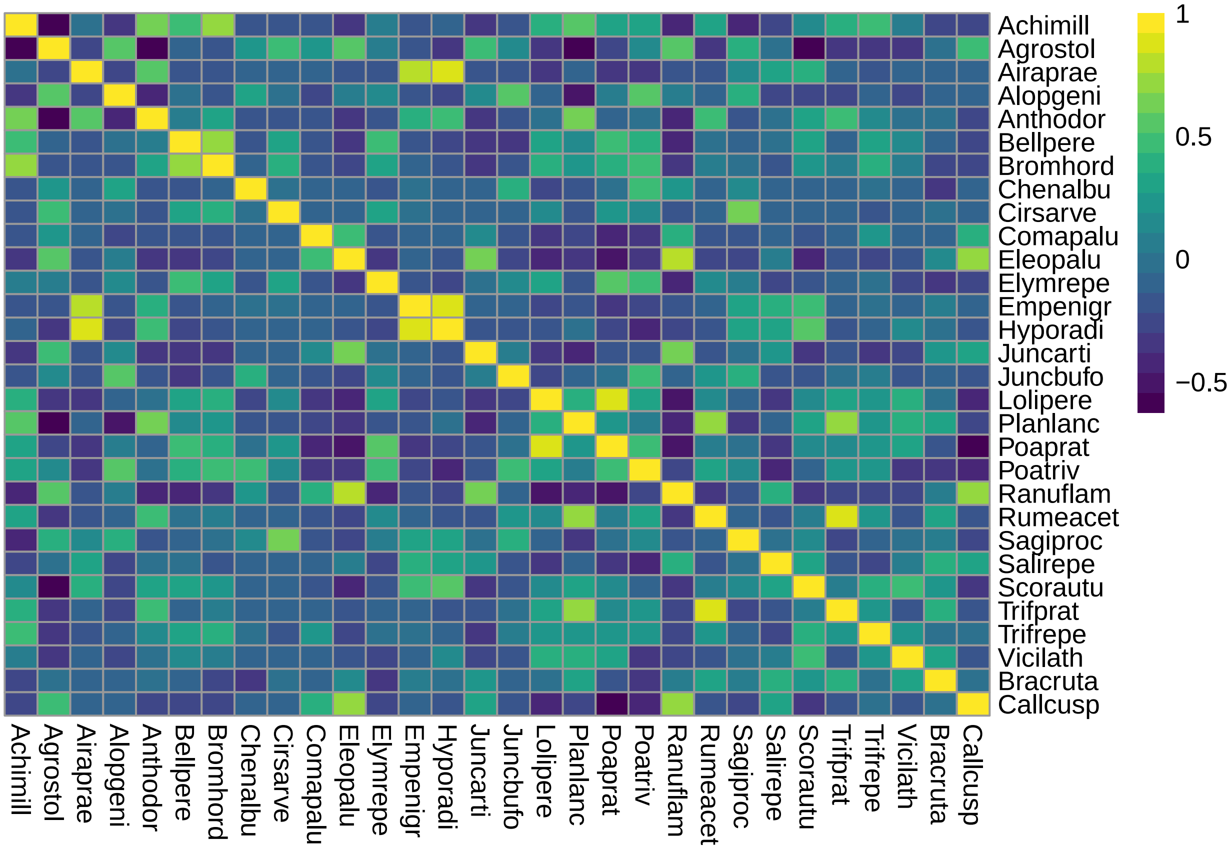
```
library(vegan)
```

```
## Loading required package: permute

##
## Attaching package: 'permute'

## The following object is masked from 'package:gtools':
##
##     permute

## The following object is masked from 'package:devtools':
##
##     check

## This is vegan 2.5-6
```

```
data(dune)
dune_cor = cor(dune)
symnum(dune_cor)
```

```
##          Ac Ag Ar Al An Bl Brm Ch Cr Cm Elp Ely Em H Jncr Jncb L Pl Pp Pt
## Achimill 1
## Agrostol ,  1
## Airaprae       1
## Alopgeni .  .     1
## Anthodor ,  .  .  .  1
## Bellpere .              1
## Bromhord ,              ,  1
## Chenalbu                       1
## Cirsarve    .           .  .      1
## Comapalu                             1
## Eleopalu .  .        .  .            .  1
## Elymrepe                .         .     .   1
## Empenigr       +     .                          1
## Hyporadi    .  *     .                          +  1
## Juncarti .  .        .  .  .            ,            1
## Juncbufo          .     .      .                          1
## Lolipere .  .  .        .  .         .  .   .        .         1
## Planlanc .  ,     .  .                  .            .         . 1
## Poaprat  .     .        .  .         .  .   .   .              +    1
## Poatriv        .  .     .  .   .     .  .   .      .      .         .  1
## Ranuflam .  .        .  .  .         .  ,   .        ,         . .  .
## Rumeacet .  .        .                                           ,     .
## Sagiproc .  .     .               ,             .         .      .
## Salirepe                                        .  .           .    .  .
## Scorautu    .  .        .               .       .  . .           .
## Trifprat .  .        .                                         . ,
## Trifrepe .  .              .                         .
## Vicilath    .                                                  . .     .
## Bracruta                       .            .                    .     .
## Callcusp    .                        .  ,            .         .    .  .
##          Rn Rm Sg Sl Sc Trfp Trfr V Brc Cl
## Achimill
## Agrostol
## Airaprae
## Alopgeni
## Anthodor
## Bellpere
## Bromhord
## Chenalbu
## Cirsarve
## Comapalu
## Eleopalu
## Elymrepe
## Empenigr
## Hyporadi
## Juncarti
## Juncbufo
## Lolipere
## Planlanc
## Poaprat
## Poatriv
## Ranuflam 1
## Rumeacet .  1
## Sagiproc       1
## Salirepe .        1
## Scorautu .        .  1
## Trifprat    +           1
## Trifrepe             .       1
## Vicilath             .            1
## Bracruta    .     .     .         . 1
## Callcusp ,           .                  1
## attr(,"legend")
## [1] 0 ' ' 0.3 '.' 0.6 ',' 0.8 '+' 0.9 '*' 0.95 'B' 1
```

```
pheatmap(dune_cor, cluster_rows = FALSE, cluster_cols = FALSE, color = viridis(20))
```



Exercise 5.3

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:permute':
##
##     how

## The following object is masked from 'package:purrr':
##
##     cross

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
data(spirals)
```