Assignment 2

Sebastian Veuskens

Exercise 1

Find the optimal number of clusters with the gap test statistic from the lecture.

Include the gapnc function:

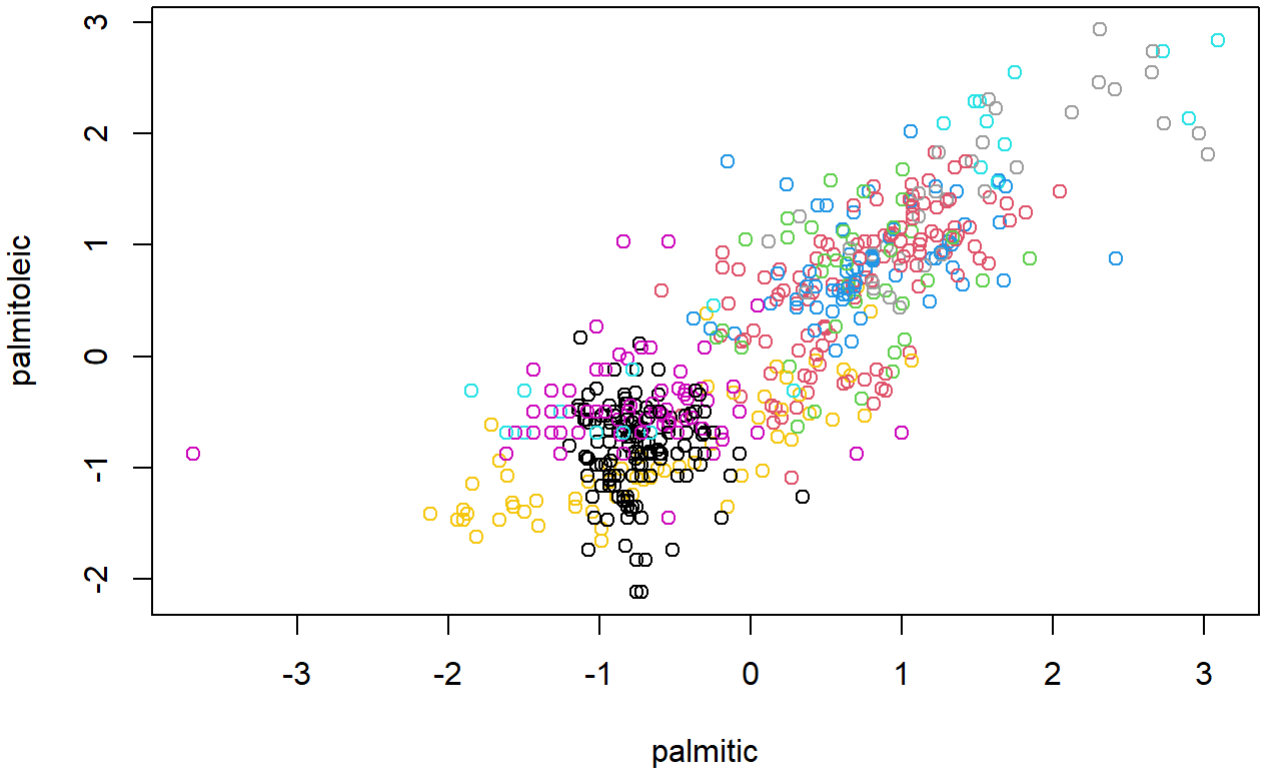
```
require(cluster)

gapnc <- function(data,FUNcluster=kmeans,
                  K.max=10, B = 100, d.power = 2,
                  spaceH0 ="scaledPCA",
                  method ="globalSEmax", SE.factor = 2,...){
# As in original clusGap function the ... arguments are passed on
# to the clustering method FUNcluster (kmeans).
# Run clusGap
    gap1 <- clusGap(data,kmeans,K.max, B, d.power,spaceH0,...)
# Find optimal number of clusters; note that the method for
# finding the optimum and the SE.factor q need to be specified here.
    nc <- maxSE(gap1$Tab[,3],gap1$Tab[,4],method, SE.factor)
# Re-run kmeans with optimal nc.
    kmopt <- kmeans(data,nc,...)
    out <- list()
    out$gapout <- gap1
    out$nc <- nc
    out$kmopt <- kmopt
    out
}
```

The best number of clusters according to the gap statistics with global SE max as a method is K=18 clusters

```
olive = read.table("data/oliveoil.dat", header = TRUE)
solive = scale(olive[,3:10])

set.seed(12345)

cgolive <- gapnc(solive, K.max = 20, SE.factor = 2)
print(cgolive$nc)
```

```
## [1] 18
```

```
plot(solive, col=cgolive$kmopt$cluster)
```



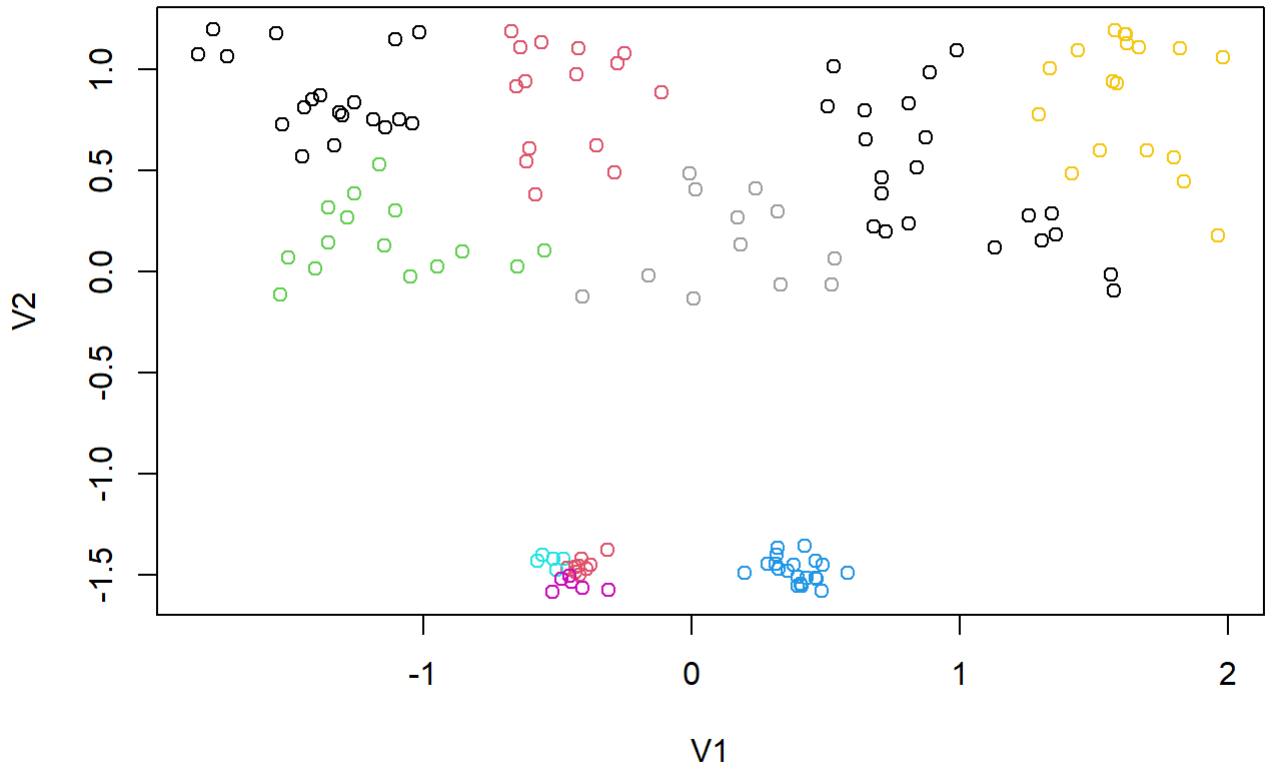
The best number of clusters according to the gap statistics with global SE max as a method is K=10 clusters.

The number of clusters is highly volatile and depends strongly on the random seed. Thus, the K_10 number of clusters should be treated with caution.

```
artificial2 <- read.table("data/clusterdata2.dat", header = FALSE)
sartificial2 <- scale(artificial2)

set.seed(12345)

cgartificial2 <- gapnc(sartificial2, K.max = 20, SE.factor = 2)
print(cgartificial2$nc)
```

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

```
plot(sartificial2, col=cgartificial2$kmopt$cluster)
```

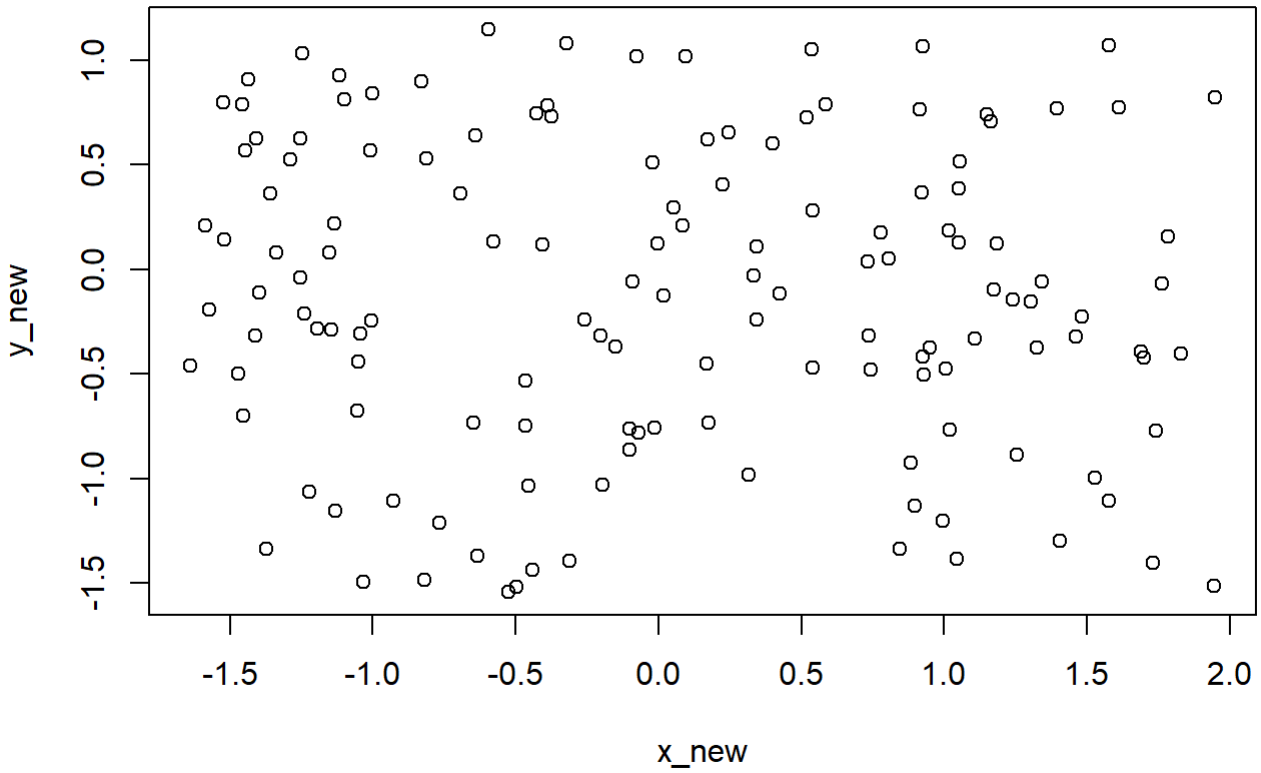


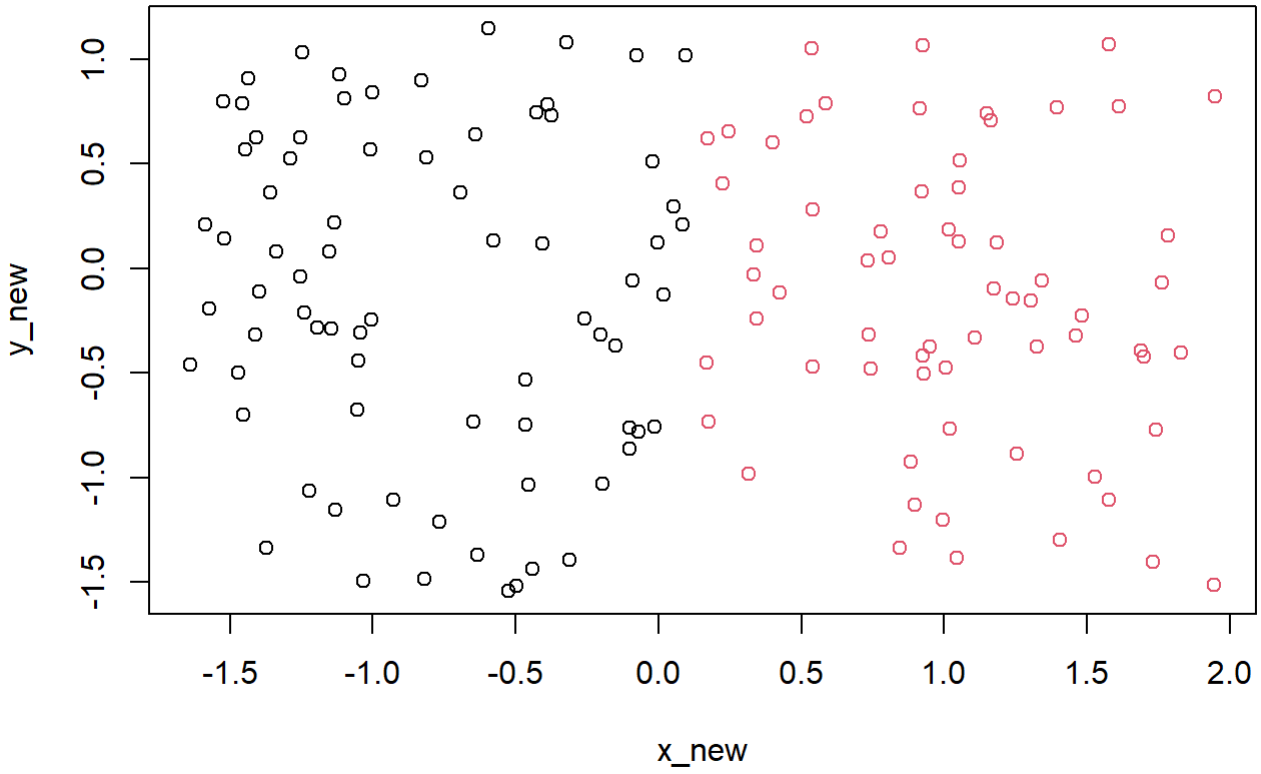
Create uniform random dataset

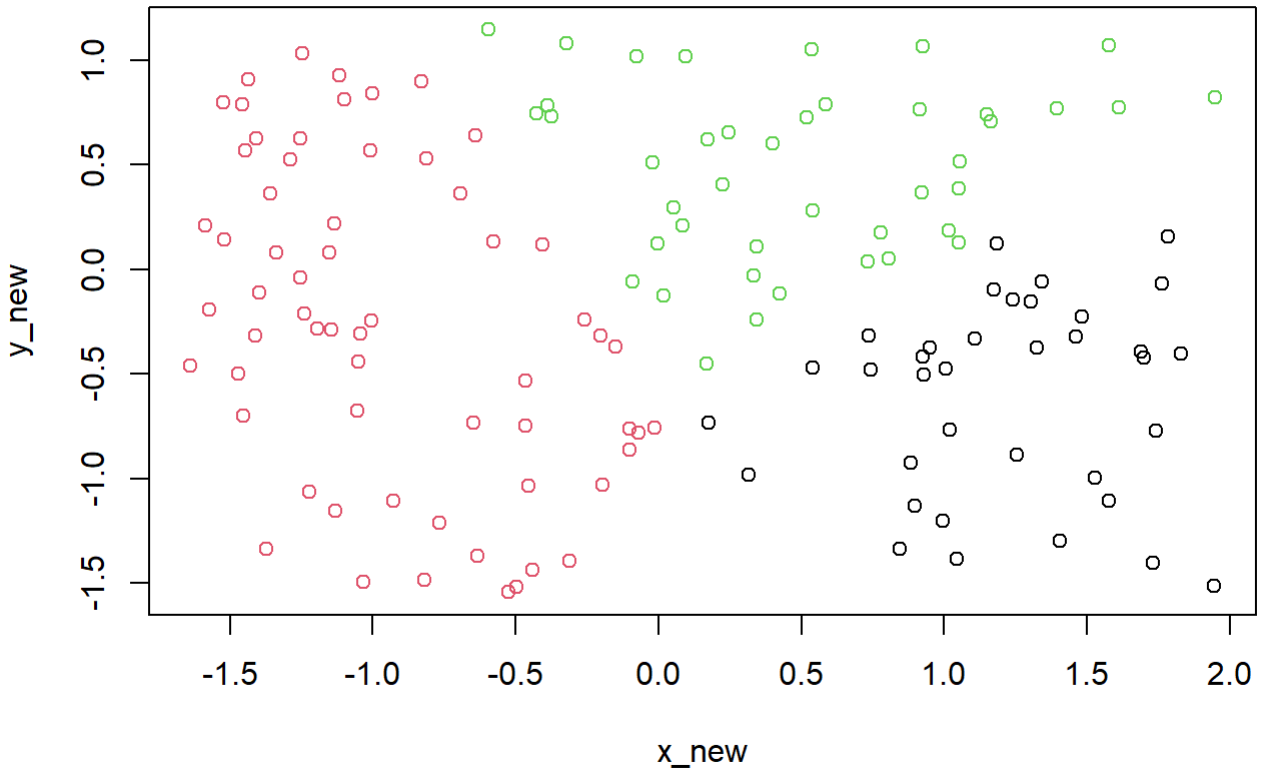
Create a new dataset within in the borders of the “old” dataset *clusterdata2*. It consists of data uniformly distributed within the borders (min and max values) of the “old” dataset and has the same number of elements.

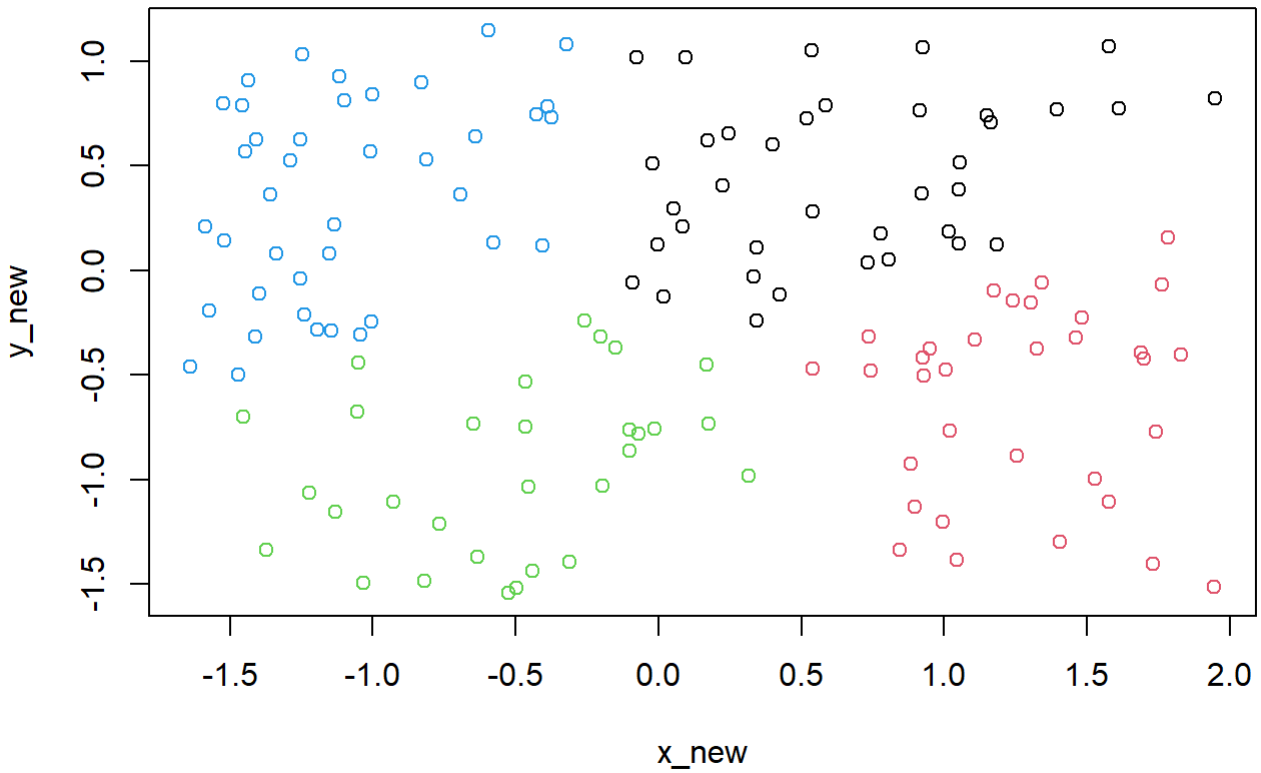
```
x_old = sartificial2[,1]
y_old = sartificial2[,2]
n = length(x_old)
x_new = runif(n, min(x_old), max(x_old))
y_new = runif(n, min(y_old), max(y_old))
data_unif = data.frame(x_new, y_new)
```

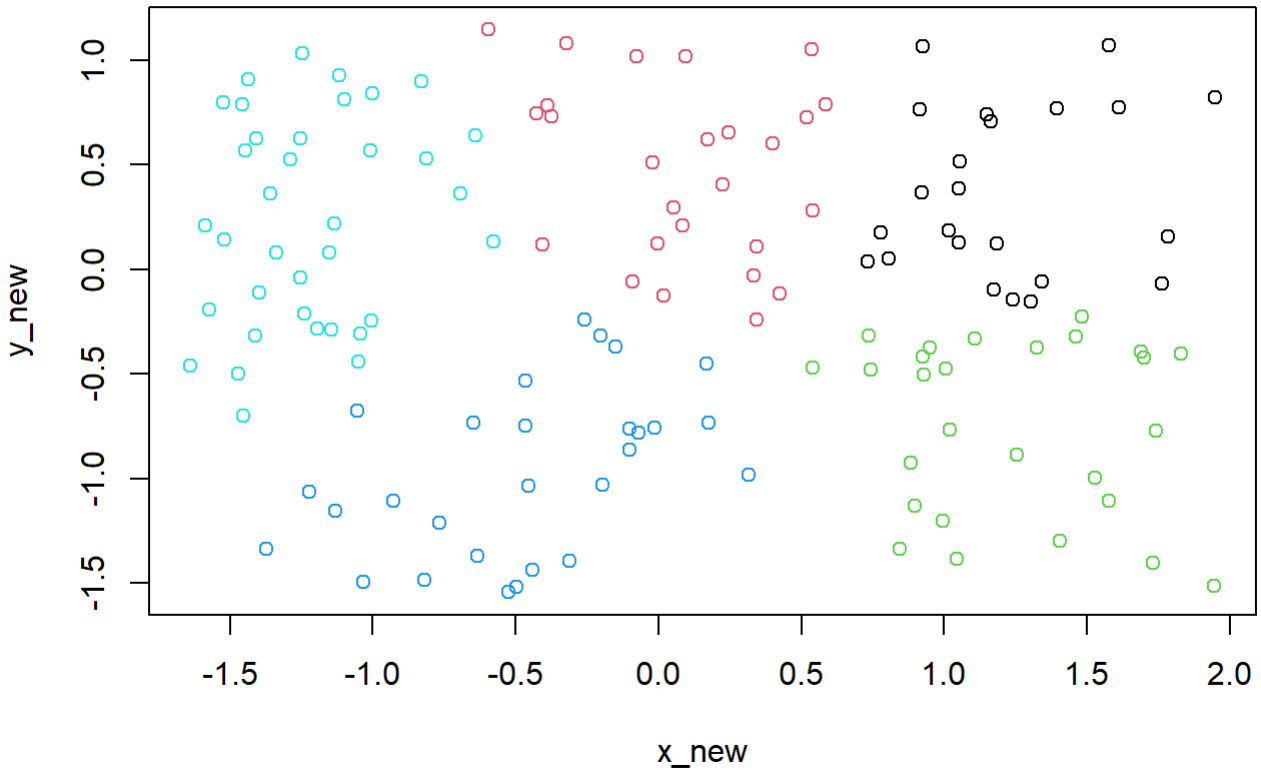
```
set.seed(12345)

se_unif <- numeric(0)
se <- numeric(0)
kclusterings_unif = list()
kclusterings = list()
for (k in 1:10) {
    kclusterings_unif[[k]] <- kmeans(data_unif, k)
    se_unif[k] <- log(kclusterings_unif[[k]]$tot.withinss)
    kclusterings[[k]] <- kmeans(sartificial2, k)
    se[k] <- log(kclusterings[[k]]$tot.withinss)

    plot(data_unif, col=kclusterings_unif[[k]]$cluster)
}
```

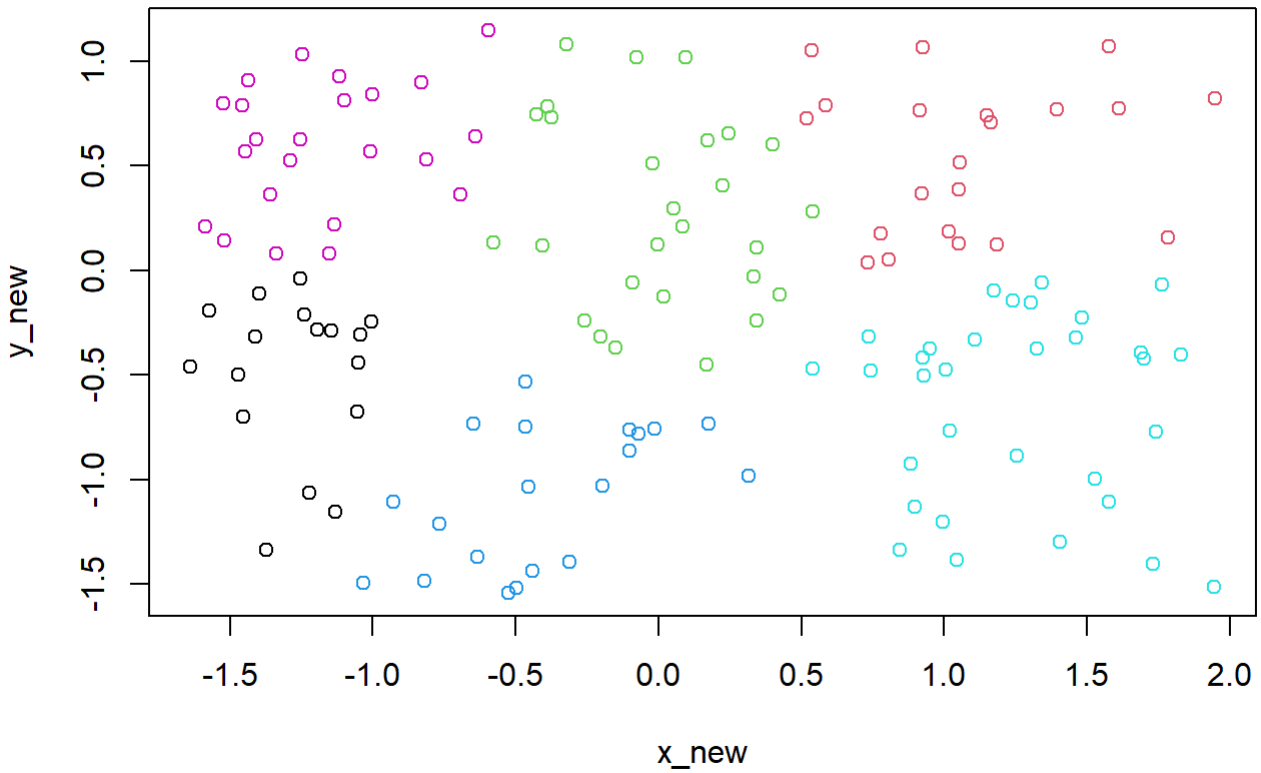

y_new
x_new
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0

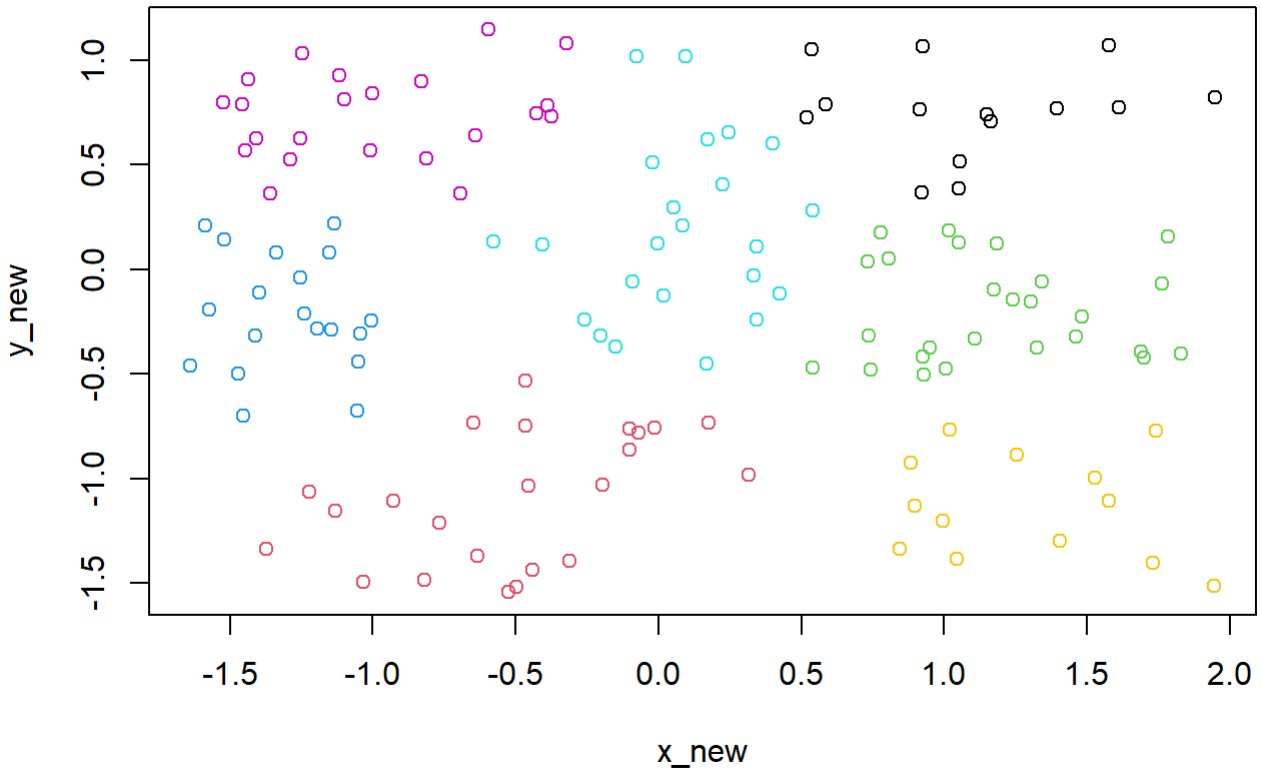

y_new
x_new
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0

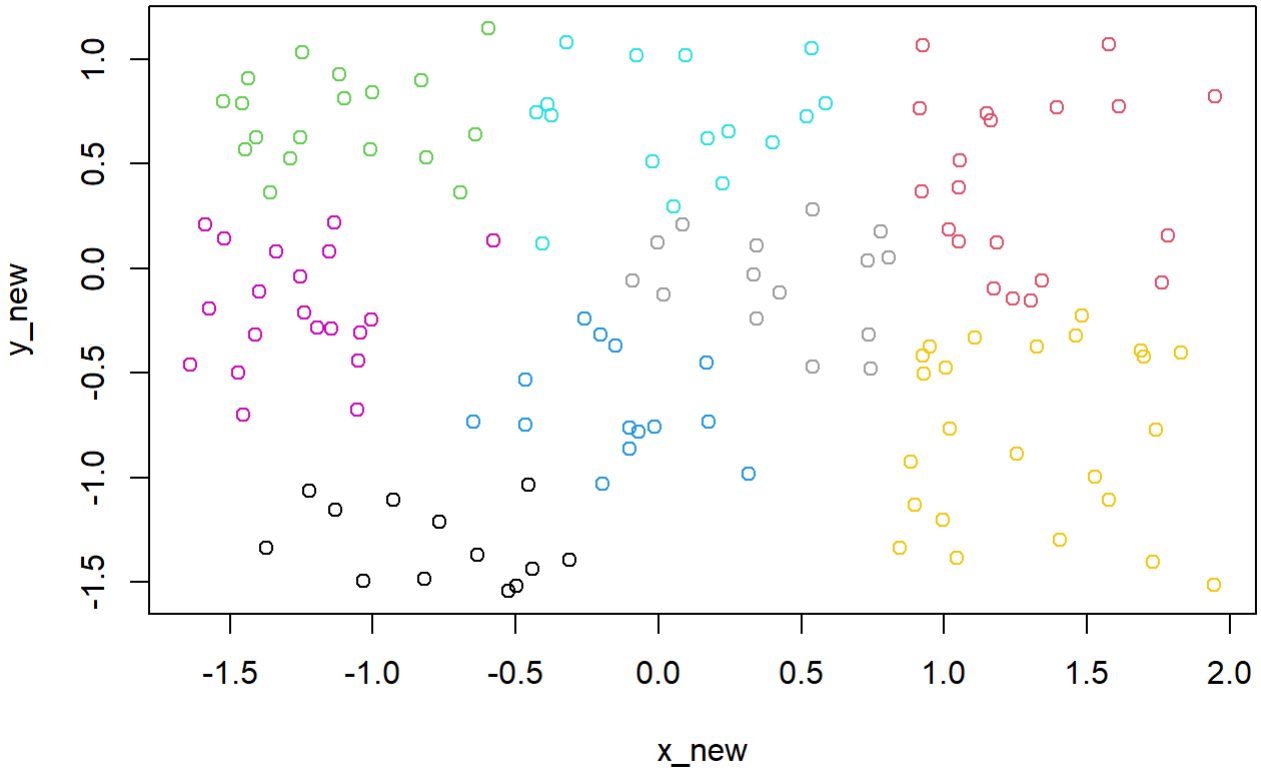

y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new

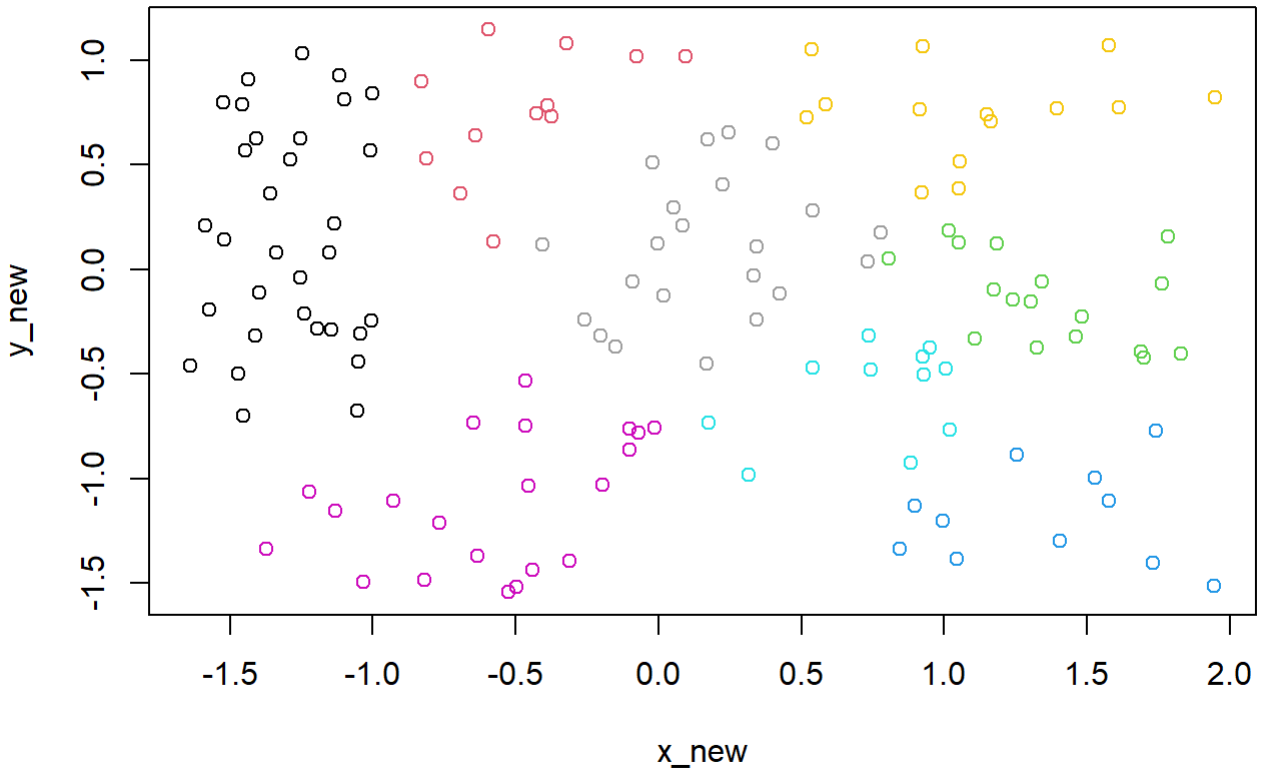

y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new

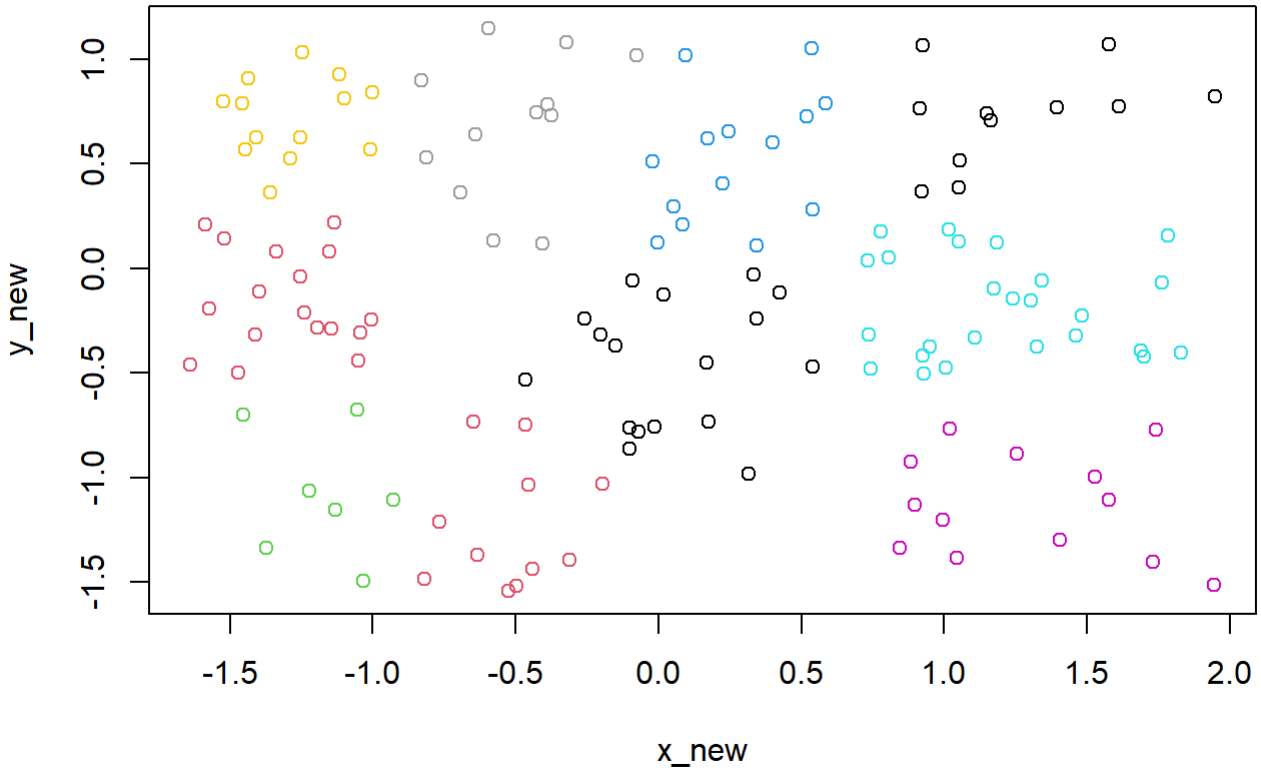

y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new


y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new

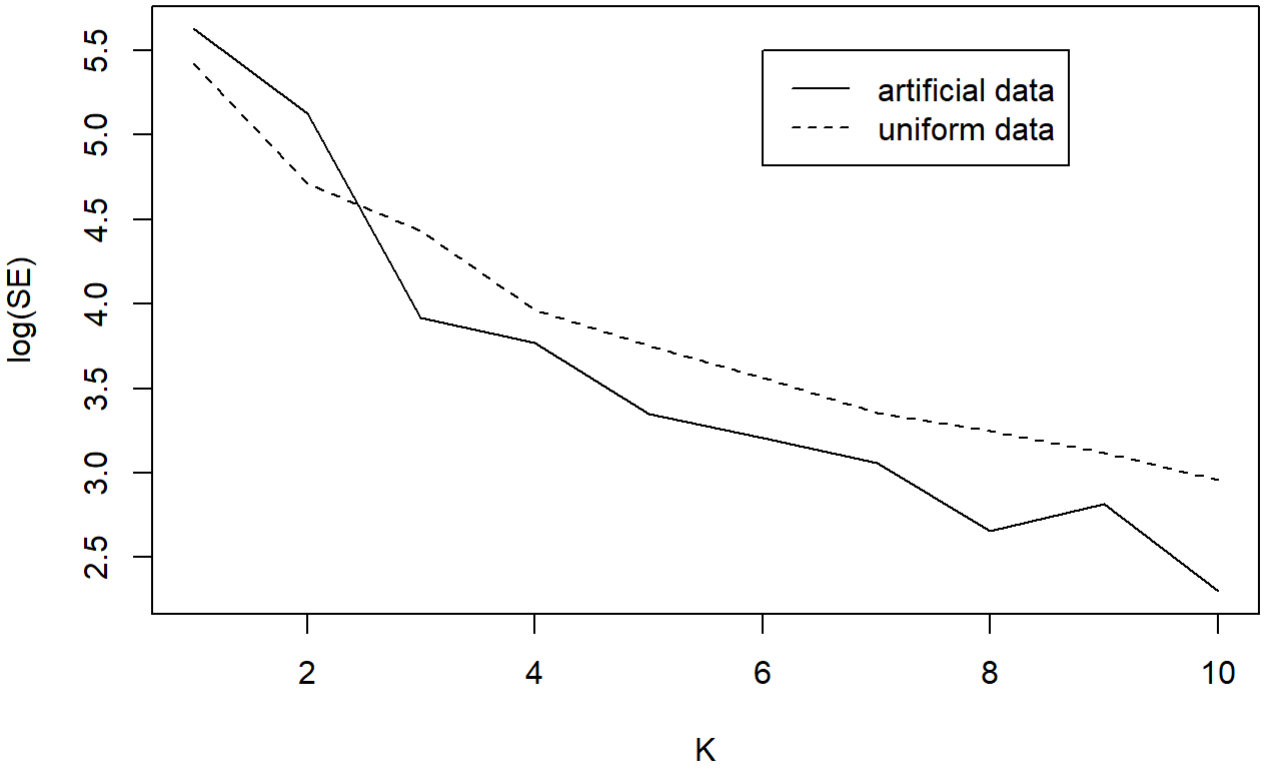

y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new


y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new


y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new


y_new
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
2.0
x_new

```
plot(se, type="l", xlab="K", ylab="log(SE)")
lines(se_unif, lty=2)

legend(6, 5.5, legend=c("artificial data", "uniform data"), lty=1:2)
```



Except for 3 or less clusters, there seems to be some kind of cluster structure in the artificial 2 dataset since its withinss is permanently lower than for uniformly randomly distributed data.

Exercise 2

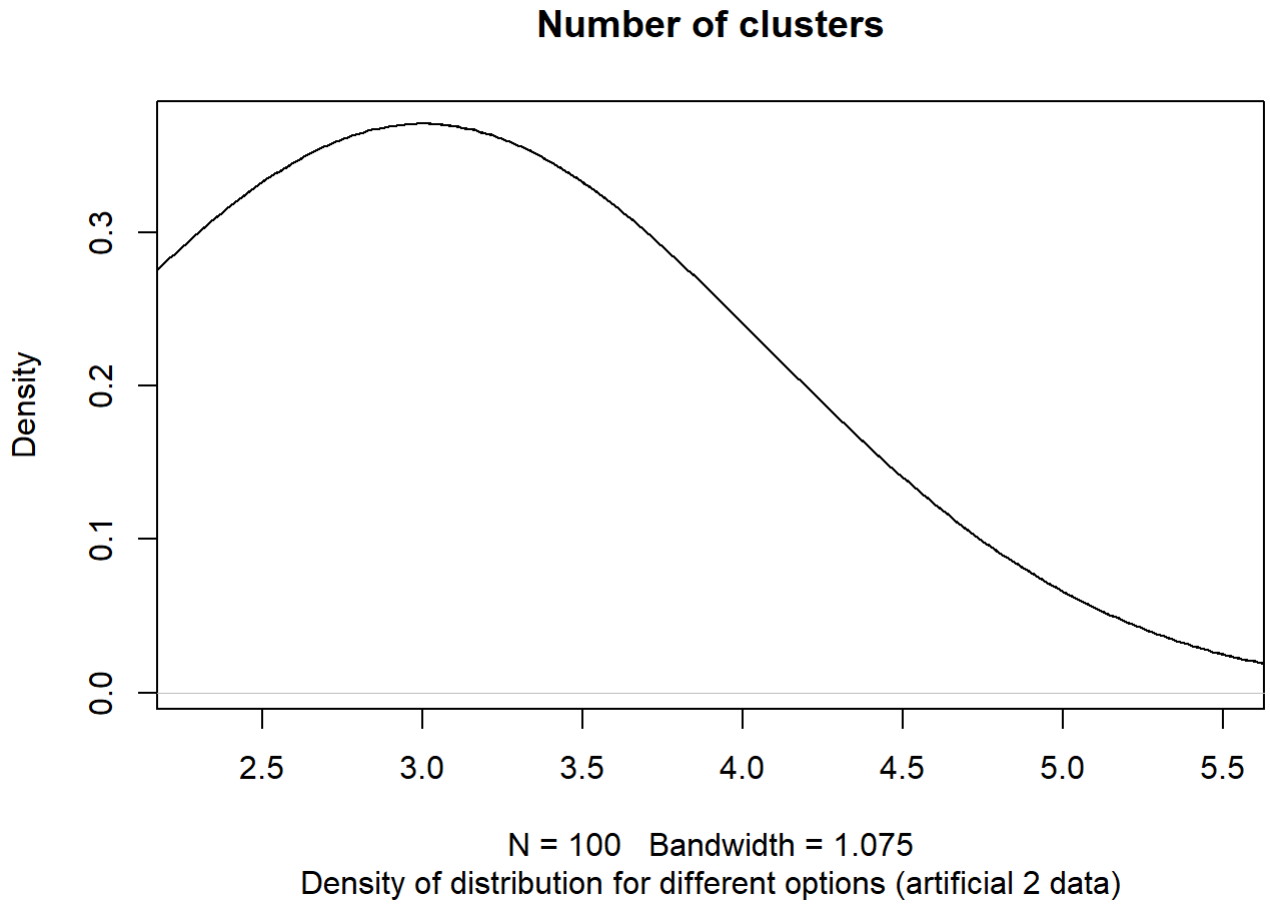
Choose for 100 different, but similarly distributed datasets the best number of clusters. Best refers here to the gap statistics with four different options for the clusGap/gapnc function.

Artificial 2 dataset

```
library(sn)

set.seed(12345)

num_clusters <- list()

for (k in 1:100) {
    v1 <- c(rnorm(50,0,1), rsn(70,5,1,8), rnorm(30,6,1))
    v2 <- c(rnorm(50,0,1), rsn(70,0,1,8), 8+rt(30,5))
    clusterdata <- cbind(v1,v2)

    num_clusters[["SE1 HOPC"]][k] <- gapnc(clusterdata, nstart=10, spaceH0 ="scaledPCA", SE.f
actor = 1)$nc
    num_clusters[["SE1 HOOr"]][k] <- gapnc(clusterdata, nstart=10, spaceH0 ="original", SE.fa
ctor = 1)$nc
    num_clusters[["SE2 HOPC"]][k] <- gapnc(clusterdata, nstart=10, spaceH0 ="scaledPCA", SE.f
actor = 2)$nc
    num_clusters[["SE2 HOOr"]][k] <- gapnc(clusterdata, nstart=10, spaceH0 ="original", SE.fa
ctor = 2)$nc
}
```

Display the densities of the distribution of the number of clusters for each option. It is visible that for all options the optimal number of clusters is 3 and thus the “correct” underlying number of clusters.

```
plot(density(num_clusters[["SE1 HOPC"]]), lty=1, xlim = c(2.3, 5.5), main="Number of cluster
s",
             sub="Density of distribution for different options (artificial 2 data)")
lines(density(num_clusters[["SE1 HOOr"]]), lty=2)
lines(density(num_clusters[["SE2 HOPC"]]), lty=3)
lines(density(num_clusters[["SE2 HOOr"]]), lty=4)

legend(4.5, 1.75, legend = c("SE:1 HO:PC", "SE:1 HO:Or", "SE:2 HO:PC", "SE:2 HO:Or"), lty=1:
4)
```

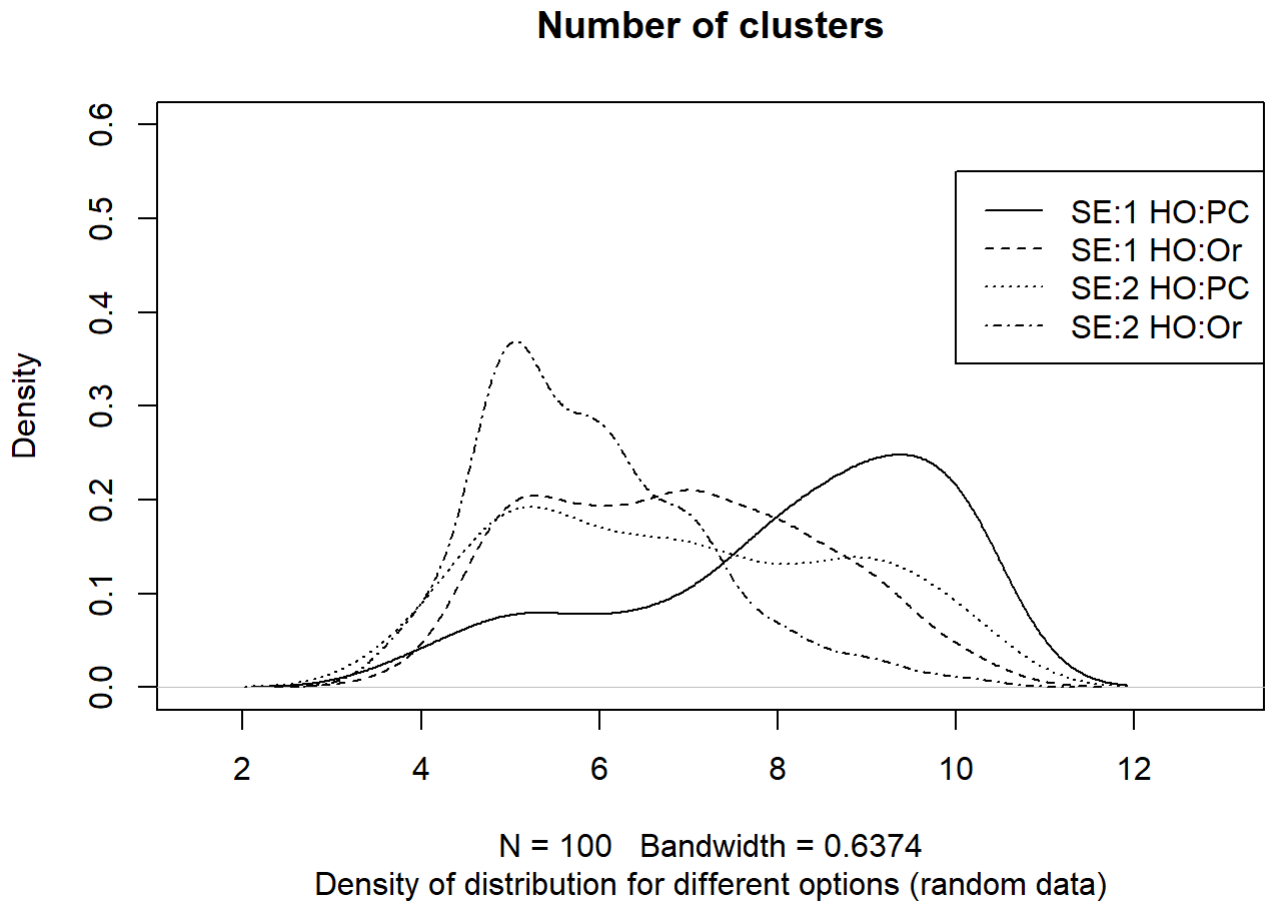


Random dataset

```
set.seed(12345)

num_clusters <- list()

for (k in 1:100) {
    v1 <- c(rnorm(50,0,1), rnorm(100,20,5), rnorm(150,6,2), rnorm(200, -3, 2))
    v2 <- c(rnorm(50,0,1), rnorm(100,13,3), rnorm(150,30,5), rnorm(200, 40, 6))
    v3 <- c(rnorm(50,-4,5), rnorm(100,5,2), rnorm(150,6,3), rnorm(200, 3, 2))
    v4 <- c(rnorm(50,0,1), rnorm(100,5,5), rnorm(150,14,3), rnorm(200, 17, 4))
    clusterdata <- cbind(v1,v2, v3, v4)

    num_clusters[["SE1 HOPC"]][k] <- gapnc(clusterdata, spaceH0 ="scaledPCA", SE.factor = 1)
$nc
    num_clusters[["SE1 HOOr"]][k] <- gapnc(clusterdata, spaceH0 ="original", SE.factor = 1)$n
c
    num_clusters[["SE2 HOPC"]][k] <- gapnc(clusterdata, spaceH0 ="scaledPCA", SE.factor = 2)
$nc
    num_clusters[["SE2 HOOr"]][k] <- gapnc(clusterdata, spaceH0 ="original", SE.factor = 2)$n
c
}
```

Display the densities of the distribution of the number of clusters for each option. It is visible that the best results are achieved in this case with **SE = 2** and **space HO = original**. With one ore both of the other options present, the gap statistic for the optimal K tends to overestimate the number of clusters K in this case.

```
plot(density(num_clusters[["SE1 HOPC"]]), xlim = c(1.5, 13), ylim = c(0, 0.6), main="Number o
f clusters",
             sub="Density of distribution for different options (random data)")
lines(density(num_clusters[["SE1 HOOr"]]), lty=2)
lines(density(num_clusters[["SE2 HOPC"]]), lty=3)
lines(density(num_clusters[["SE2 HOOr"]]), lty=4)

legend(10, 0.55, legend = c("SE:1 HO:PC", "SE:1 HO:Or", "SE:2 HO:PC", "SE:2 HO:Or"), lty=1:4)
```



Exercise 4

The correlation distance is computed by the function mentioned in the handwritten exercise sheet.

```
x_1 = c(1, 4, 5, 4, 2, 1, 1, 4)
x_2 = c(2, 3, 2, 2, 3, 3, 3, 3)
x_3 = c(7, 11, 11, 12, 9, 8, 8, 12)

corr_dist <- function(x, y) {
    return(0.5 * (1 - cor(x, y)))
}

print(corr_dist(x_1, x_2))
```

```
## [1] 0.6447072
```

```
print(corr_dist(x_1, x_3))
```

```
## [1] 0.03577897
```

```
print(corr_dist(x_1, x_2) > corr_dist(x_1, x_3))
```

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

The correlation distance between x_1 and x_2 is bigger than the distance between x_1 and x_3.

Assignment 2

06 October 2023 14:54

(3)

Note (1): $\operatorname{argmax}_L(GAP(L))$ is independent of q

$\Rightarrow k^*$ independent of q

Note (2): Assume $GAP(k) > GAP(k^*) - 1 \cdot s_{k^*}$ for any k

$\Rightarrow \quad GAP(k) > GAP(k^*) - 2 \cdot s_{k^*}$, since $s_{k^*} \geq 0$

Note (3): Assume $k_{0,1}$ is given

$\Rightarrow \quad GAP(k_{0,1}) \overset{\text{Def. of } k_{0,1}}{\geq} GAP(k^*) - 1 \cdot s_{k^*}^* \overset{\text{Note (2)}}{\geq} GAP(k^*) - 2 s_{k^*}$

Thus, $k_{0,1}$ also fulfills the condition for the inequality for $q=2$ $\boxed{GAP(k_{0,2}) > GAP(k^*) - 2 s_{k^*}}$

Since $k_{0,2}$ is the smallest k, that fulfills this inequality and since $k_{0,1}$ fulfills this inequality for $q=2$, it is always true that

$$\boxed{k_{0,2} \leq k_{0,1}}$$

(4)

The correlation dissimilarity has the property that it tends to assign two observations close to each other if they are positively correlated (if one is big/small, then the other one is big/small and vice versa).

Definition: ($\operatorname{cor}(x_i, x_j)$: correlation of x_i and x_j)

$$\boxed{d_c(x_i, x_j) = \frac{1}{2}\left(1 - \operatorname{cor}(x_i, x_j)\right)}$$

(one could also choose $d(x,y) = 1 - |\operatorname{cor}(x,y)|$, but in this case we choose d_c)

Proof of dissimilarity:

(1) $d(x,y) \overset{!}{=} d(y,x) \overset{!}{\geq} 0$:

$$\frac{1}{2}\left(1 - \operatorname{cor}(x_i, x_j)\right) = \frac{1}{2}\left(1 - \operatorname{cor}(x_j, x_i)\right)$$

(since correlation is symmetric) □

and $\operatorname{cor}(x_i, x_j) \in [-1, 1] \quad \forall x_i, x_j \in \mathbb{R}^p$

$\Rightarrow \operatorname{cor}(x_i, x_j) \leq 1$

$\Rightarrow 1 - \operatorname{cor}(x_i, x_j) \geq 0$ □

(2) $d(x,x) \overset{!}{=} 0 \quad \forall x \in \mathbb{R}^p$:

$$d(x_i, x_i) = \frac{1}{2}\left(1 - \operatorname{cor}(x_i, x_i)\right) = \frac{1}{2}(1 - 1) = \frac{1}{2} \cdot 0 = 0$$ □

(3) See in R code for computation