umap

Tomohiro Takahashi

2020/11/4

```
library(umap)
library(vegan)
```

```
## Loading required package: permute
```

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

```
## This is vegan 2.5-6
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following object is masked from 'package:vegan':
##
##     diversity
```

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

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

データファイルを確認する Check data files

LPS刺激実験により、形・動き解析に足るトラッキングができ、かつ、トランスクリプトーム解析に足るデータが取れた１７細胞を対象とする。

１７細胞のそれぞれのデータは以下の２ファイルとして固定する。

- "LPS_only_shapemovement.csv"
- "LPS_only_transcriptome.csv"

作業手順は以下の通り。

データファイルは２つある。

There are two data files.

１つは細胞の形・動き特徴量を納めたファイル。２４データ行とヘッダー行、４１列。 冒頭１列は実験ID、次の２行は実験条件と実験者がつけたラベル、残り３８行が特徴量の行。

One file is on the features of shape and movement. It has 25 rows with 24 sample data rows and one header row. It has 41 columns; the left-most column is "experiment-id". The next two columns are an experimental condition and a label by the data-provider. The rests, 38 columns, are the quantitative features.

もう一つは、細胞の遺伝子発現量を納めたファイル。２万超行、５３列。ヘッダー行と遺伝子行。ヘッダー行は実験ID。５３列のうち冒頭１列は遺伝子名、残りが５２細胞。

The other file is on the gene expression of cells. It has more than 20,000 rows and 53 columns. The header row with many gene rows. The header row is for experiment-id. The left-most among 53 columns is gene ids. The rests, 52 columns, are 52 cells.

オリジナルデータを整理する Clean-up the original data

細胞同士のSimilarity/Dissimilarityの距離行列を、形・動き情報と、遺伝子発現情報とからそれぞれ作り、２つの距離行列の間に相関があるかどうかを調べるのが目的である。

The goal of this analysis is to evaluate correlation between two distance matrices; one matrix is a similarity/dissimilarity matrix of cells based on their shape and movement; the other matrix is a sim/dissim matrix of cells based on their gene expression profile.

したがって、２つのファイルの細胞の対応を取る必要がある。

Therefore, the collesponding cells between two files should be checked.

形・動きファイルの細胞数は24、遺伝子発現ファイルの細胞数は52。

The number of cells in the shape/movement file is 24 and one in the expression file is 52.

遺伝子発現ファイルにはあるが、形・動きファイルにはないexperimen-idが相当数存在する。これは、形・動き解析処理がうまく回らず、形・動きデータがない細胞に相当する。

There are many experiment-ids that exist in the gene expression file but not in the shape/movement file. These ids represent the cells whose shape/movement analysis was unsuccessful and subsequently no shape/movement data are available.

それとは別に、形・動きファイルのexperiment-idには２重重複IDが２つ(計４つ)あり、それに対応するexperiment-idは、遺伝子発現ファイルには１列ずつしかない。

Besides this partial missing ids, there are two ids that appeared twice in the shape/movement file, but only once in the gene expression file.

この不整合はデータ提供者にチェックするべきである。実際、確認すると、１つの実験から、１細胞トランスクリプトームデータは１つ提供され、同じ実験に対応する動画データには２つの細胞が撮影されているためであると判明した。

This id-discrepancy is critical for data-analysis and it should be clarified. Actually this discrepancy was asked to Yusri-san and he answered that there were two experimens that provided one set of single-cell transcriptome data but whose movies had two cells.

１対１対応の取れない細胞は解析から除外すべきであるから、結局、20細胞が解析の対象となる。

No cells that does not have both shape/movement and gene expression data should be excluded from the analysis. Eventually 20 cells remained for the analysis.

入力用ファイルは、この２０細胞用に準備しなおすこととする。 また、実験条件と実験者の提供情報により、細胞が４タイプに分けられるので、その情報カラム"label"を形・動きファイルに追加する。

The input files were remade for these 20 target cells. One column "label" was inserted into the shape/movement file that represented the types specified by experiment condition and information provided by the wet team.

さらにLPS刺激のみの細胞に絞る

２つのファイル名は以下の通り：

The name of two files are as below:

- "LPS_only_shapemovement.csv"
- "LPS_only_transcriptome.csv"

データの読み込み

```
d_x<-read.csv("LPSonly_shapemovement.csv")
d_y<-read.csv("LPSonly_transcriptome.csv")
```

```
cell_type <- d_x[,4] # experimental subtype is registered
data_x<-d_x[,c(-1,-2,-3,-4)] # non-feature columns are removed from shape-movement data
data_y <- d_y[,-1] # gene-name column is removed
data_y <- t(data_y) # cell ids should be rows
n.cell <- length(cell_type) # number of cells

dist_mat_x <- as.matrix(dist(data_x))
dist_mat_y <- as.matrix(dist(data_y))
```

細胞数

```
n.cell
```

```
## [1] 17
```

遺伝子数

```
length(data_y[1,])
```

```
## [1] 24421
```

データQC

１７細胞すべてにおいて、発現量がゼロである遺伝子は、解析対象から外す。

```
gene_non0s <- which(apply(data_y,2,sum)!=0)
```

少なくとも１個の細胞で正の発現量のある遺伝子の数は

```
length(gene_non0s)
```

```
## [1] 10997
```

この遺伝子を「対象遺伝子」と呼ぶことにする。

以下に、対象遺伝子の名称、対象遺伝子のみの遺伝子発現データオブジェクトを作る。

```
gene_non0s <- which(apply(data_y,2,sum)!=0)

gene.id <- d_y[,1]
gene.id.non0s <- gene.id[gene_non0s]

data_y_non0s <- data_y[,gene_non0s]

data_y <- data_y_non0s
```

形動き全体と対象全遺伝子発現との間の関連検定と相関係数を算出する

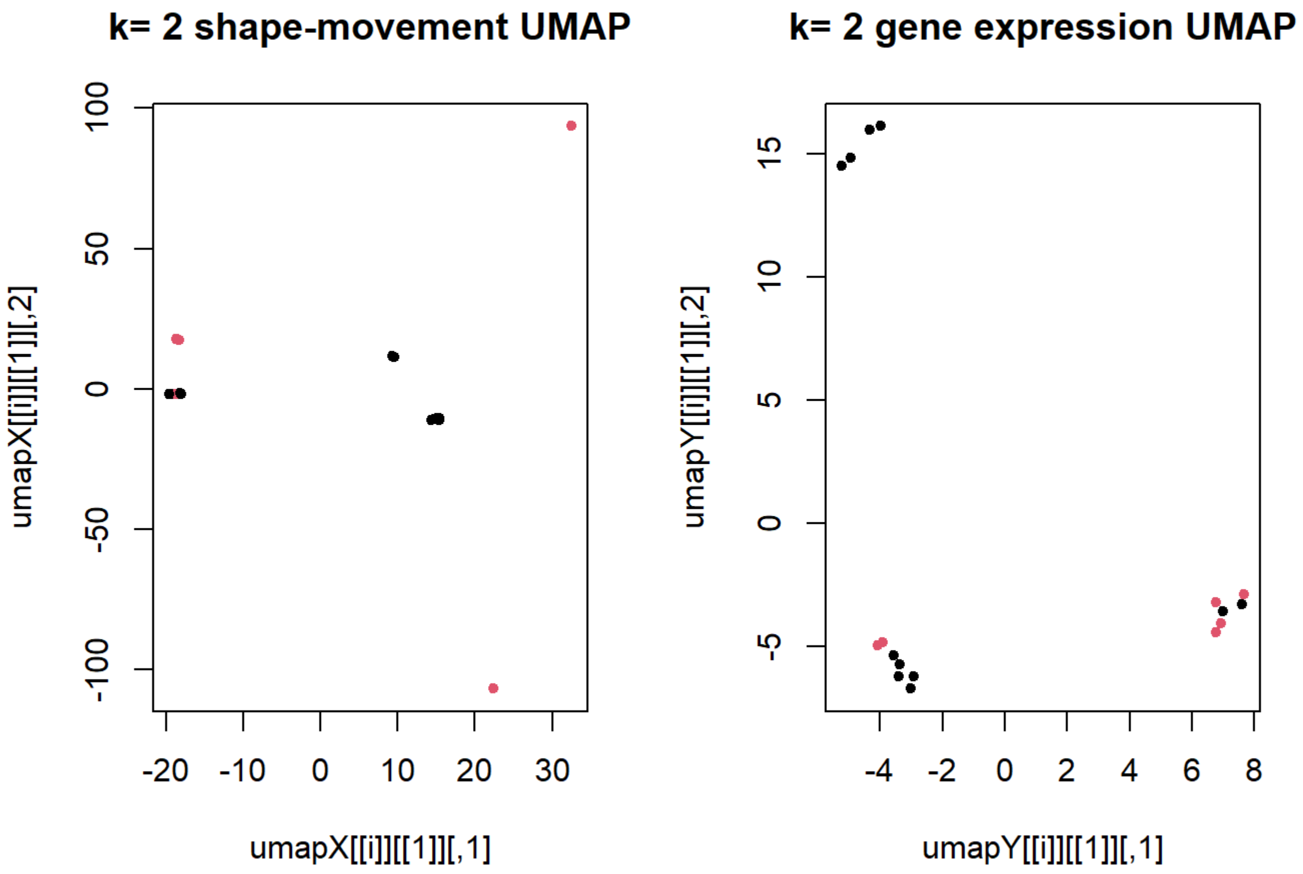
knn グラフの作成

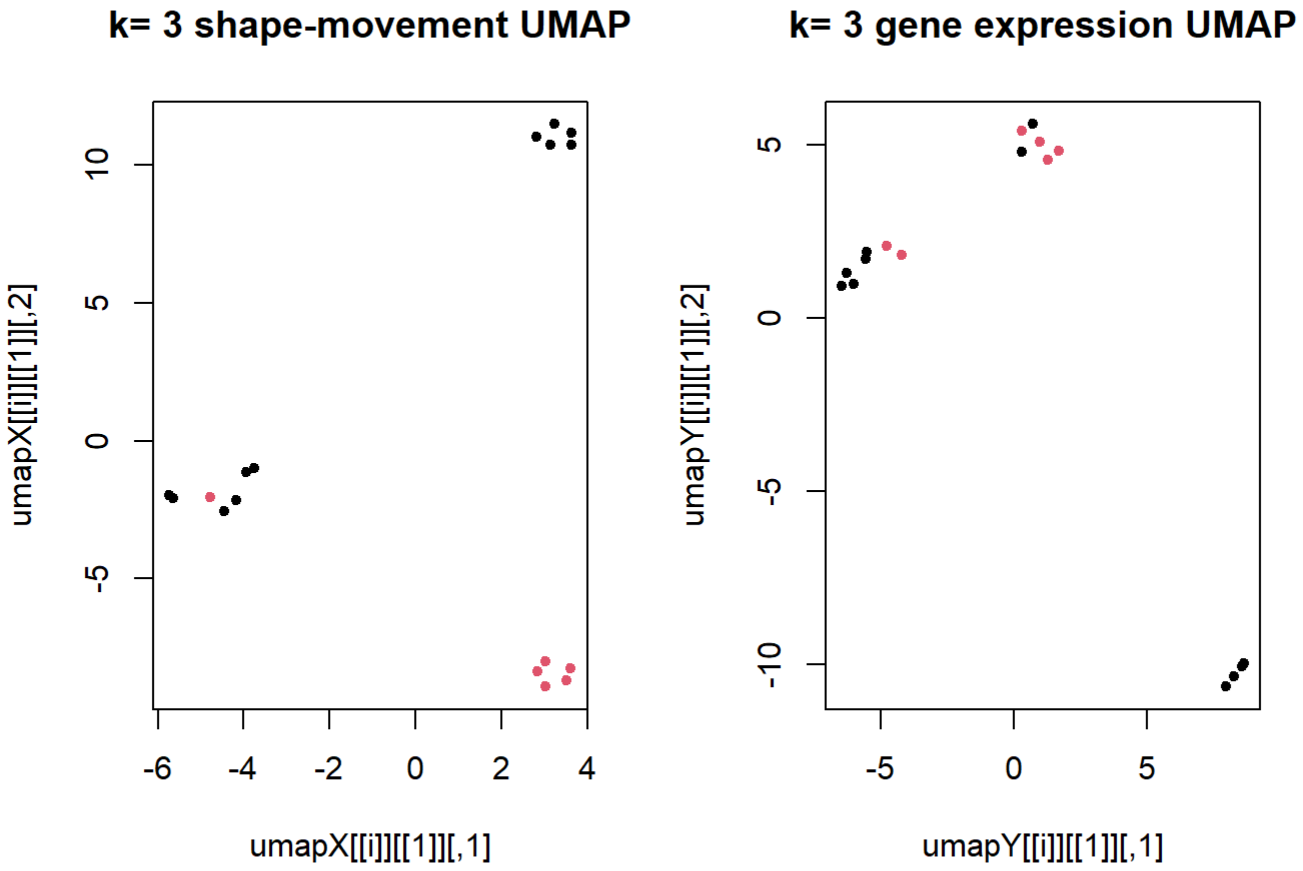
- 方法
- １７細胞のknn グラフを形動き情報から作る
- １７細胞のknn グラフを対象全遺伝子発現情報から作る
- 形動きknn グラフのk (k.sm)は、１７細胞のknnグラフが連結となるようなkの下限に近いものを採用する
- 遺伝子発現knn グラフのk (k.exp)は、１７細胞のknnグラフが連結となるようなkの下限に近いものを採用する

```
ks <- 2:n.cell # all possible k values
umapX <- umapY <- list() # Stocker of output of umap with multiple k values
for(i in 1:length(ks)){
  k <- ks[i] # k value of this time
  umap_x<-umap(data_x,n_neighbors=k) # umapping with specified k value
  umap_y<-umap(data_y,n_neighbors=k)
  umapX[[i]] <- umap_x # save the output in the stocker object
  umapY[[i]] <- umap_y
}
```

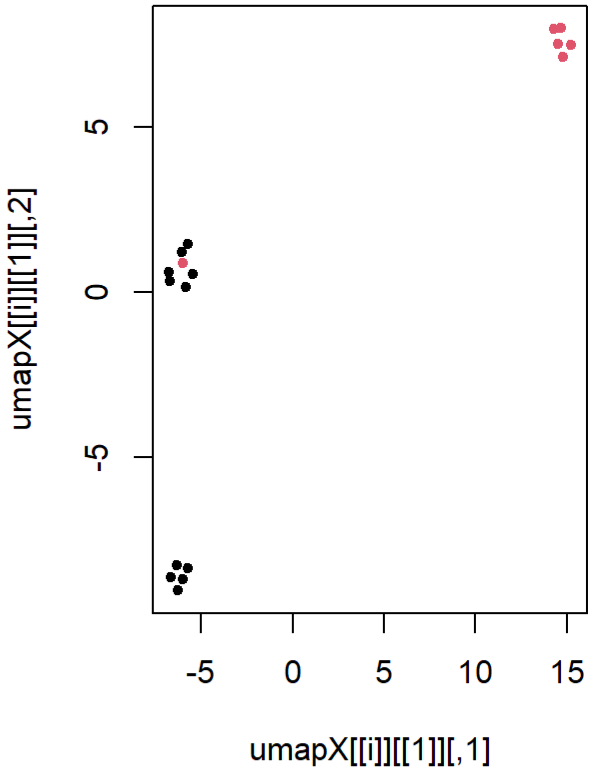
```
## Warning: failed creating initial embedding; using random embedding instead

## Warning: failed creating initial embedding; using random embedding instead

## Warning: failed creating initial embedding; using random embedding instead
```

```
par(mfcol=c(1,2))
for(i in 1:length(ks)){
  # output of function umap() is a list and its first element is 2D coordinates of umap
  plot(umapX[[i]][[1]],col=cell_type,pch=20,main=paste("k=",ks[i],"shape-movement UMAP"))
  plot(umapY[[i]][[1]],col=cell_type,pch=20,main=paste("k=",ks[i],"gene expression UMAP"))
}
```

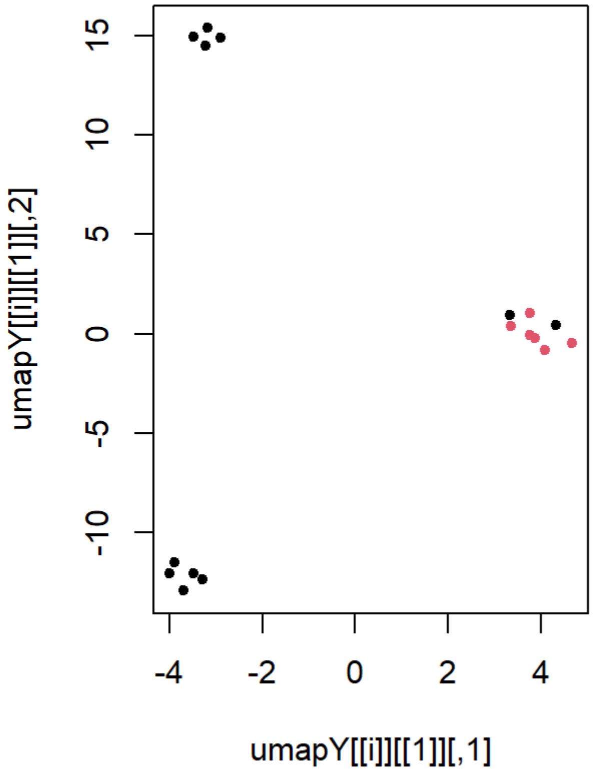

k= 2 shape-movement UMAP
umapX[[i]][[1]][,1]
umapX[[i]][[1]][,2]
-20
-10
0
10
20
30
-100
-50
0
50
100
k= 2 gene expression UMAP
umapY[[i]][[1]][,1]
umapY[[i]][[1]][,2]
-4
-2
0
2
4
6
8
-5
0
5
10
15


k= 3 shape-movement UMAP
umapX[[i]][[1]][,1]
umapX[[i]][[1]][,2]
-6
-4
-2
0
2
4
-5
0
5
10
k= 3 gene expression UMAP
umapY[[i]][[1]][,1]
umapY[[i]][[1]][,2]
-5
0
5
-10
-5
0
5

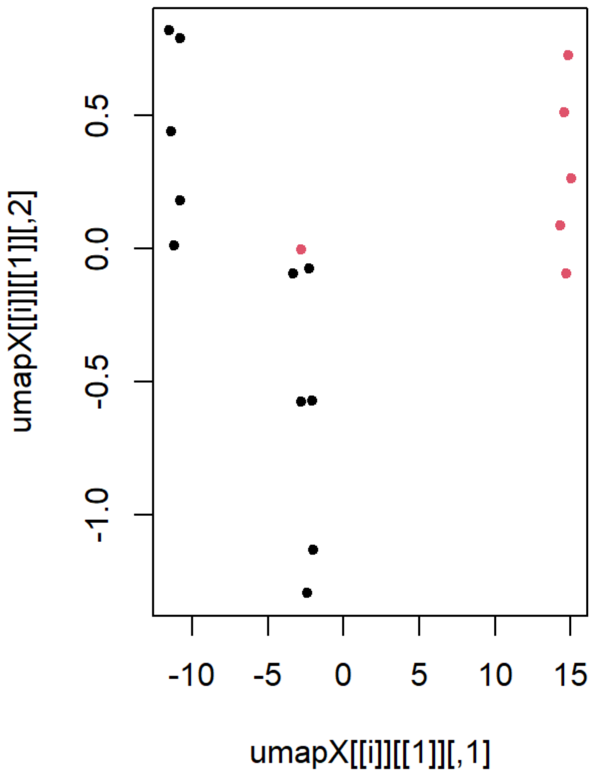
k= 4 shape-movement UMAP



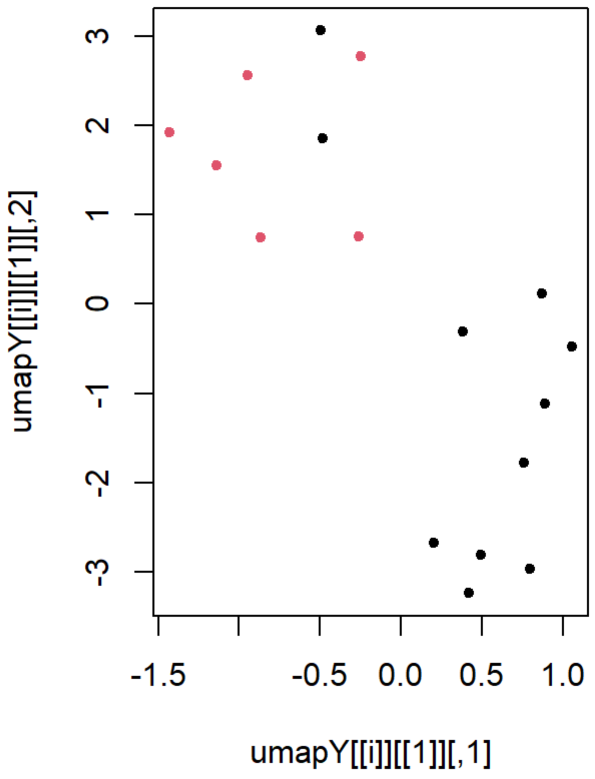
k= 4 gene expression UMAP

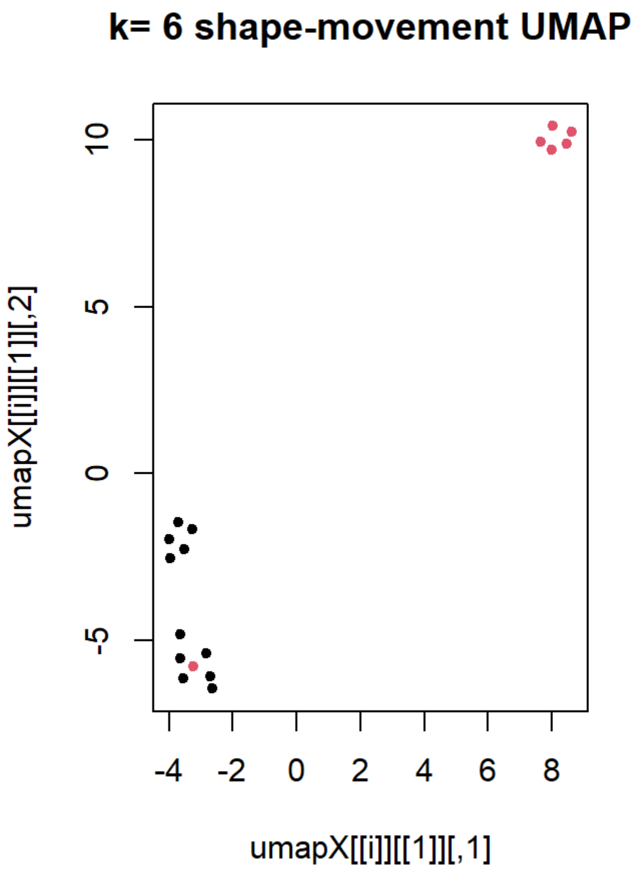


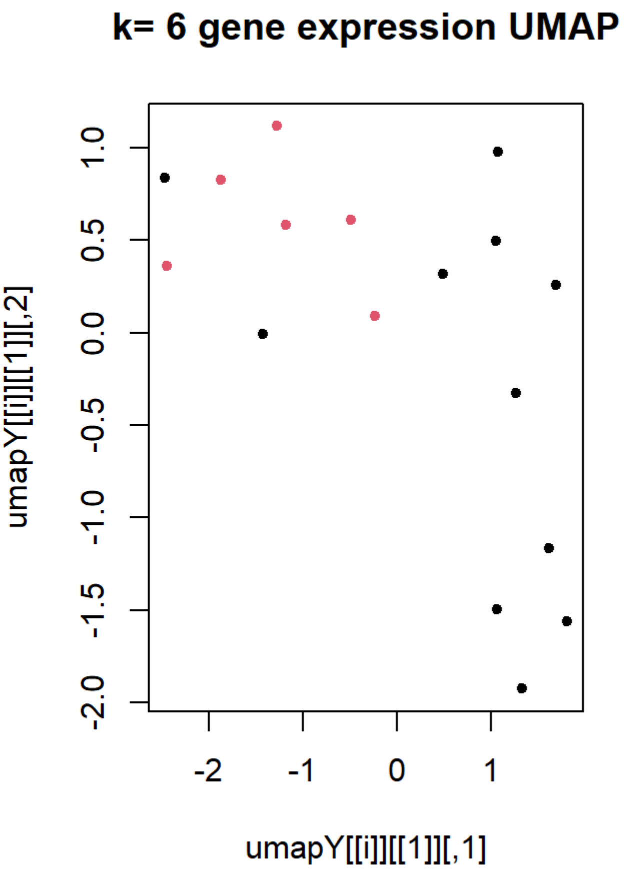
k= 5 shape-movement UMAP

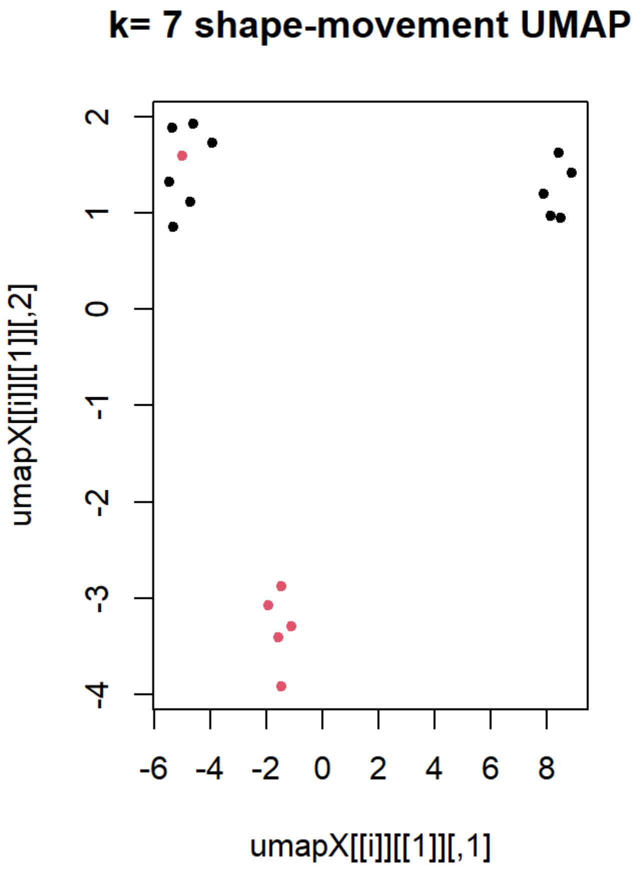


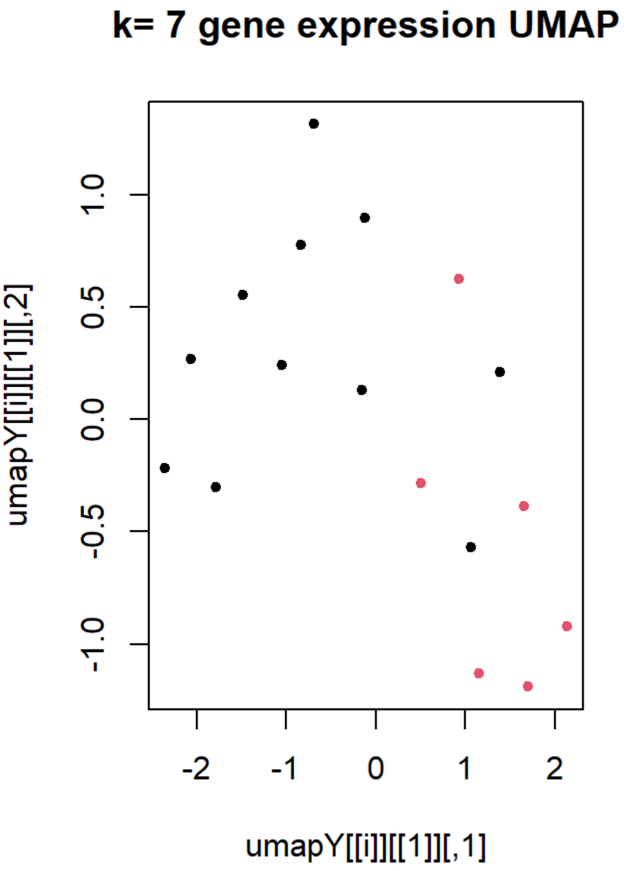
k= 5 gene expression UMAP



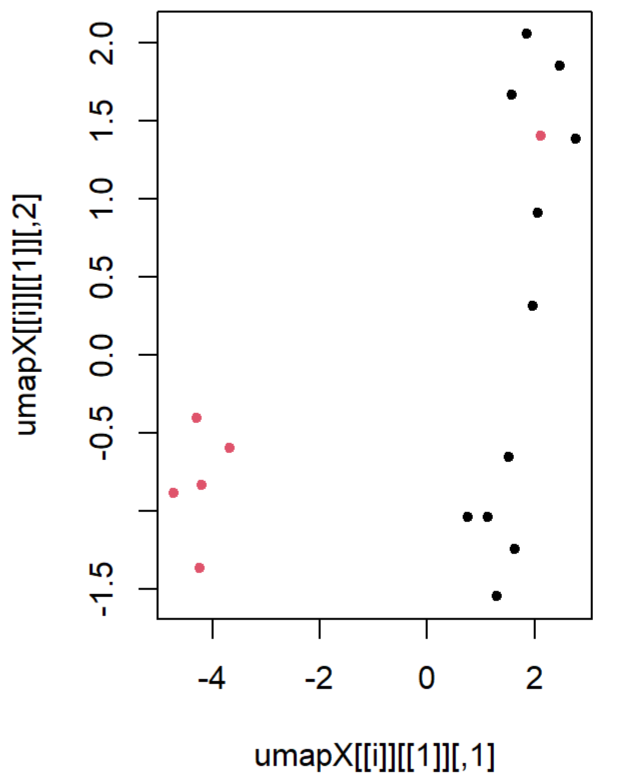

k= 6 shape-movement UMAP
umapX[[i]][[1]][,2]
10
5
0
-5
-4
-2
0
2
4
6
8
umapX[[i]][[1]][,1]


k= 6 gene expression UMAP
umapY[[i]][[1]][,2]
1.0
0.5
0.0
-0.5
-1.0
-1.5
-2.0
-2
-1
0
1
umapY[[i]][[1]][,1]

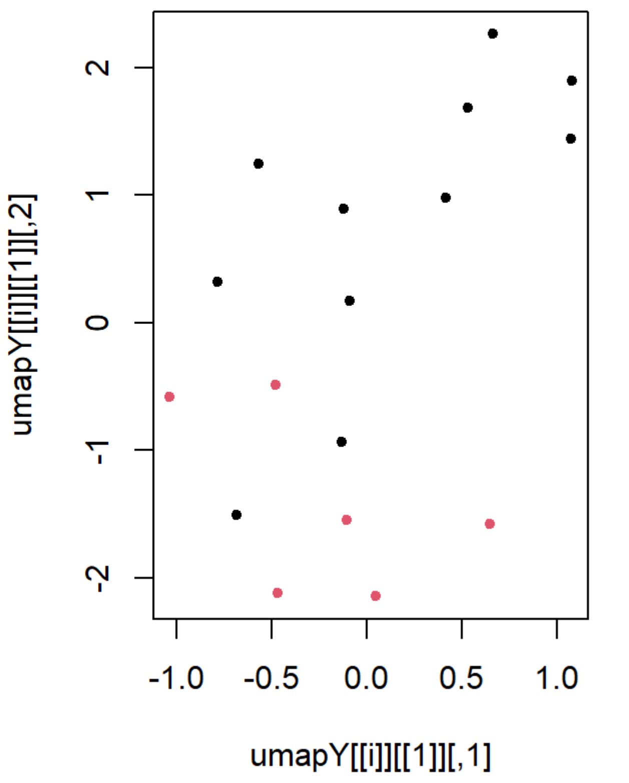

k= 7 shape-movement UMAP
umapX[[i]][[1]][,2]
2
1
0
-1
-2
-3
-4
-6
-4
-2
0
2
4
6
8
umapX[[i]][[1]][,1]


k= 7 gene expression UMAP
umapY[[i]][[1]][,2]
1.0
0.5
0.0
-0.5
-1.0
-2
-1
0
1
2
umapY[[i]][[1]][,1]

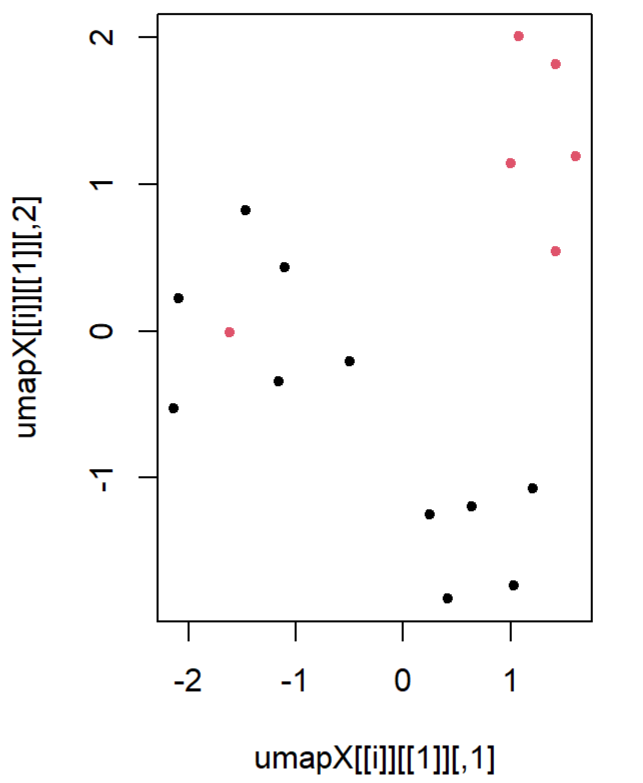
k= 8 shape-movement UMAP



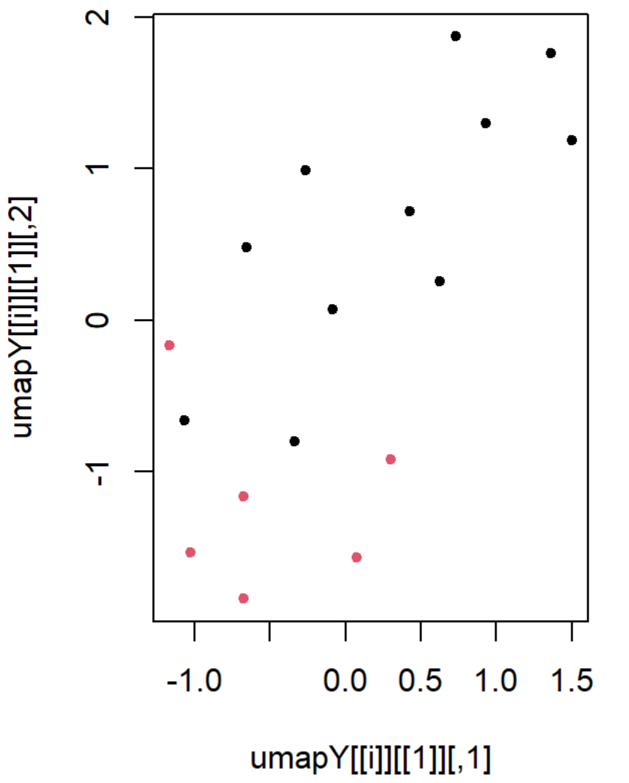
k= 8 gene expression UMAP

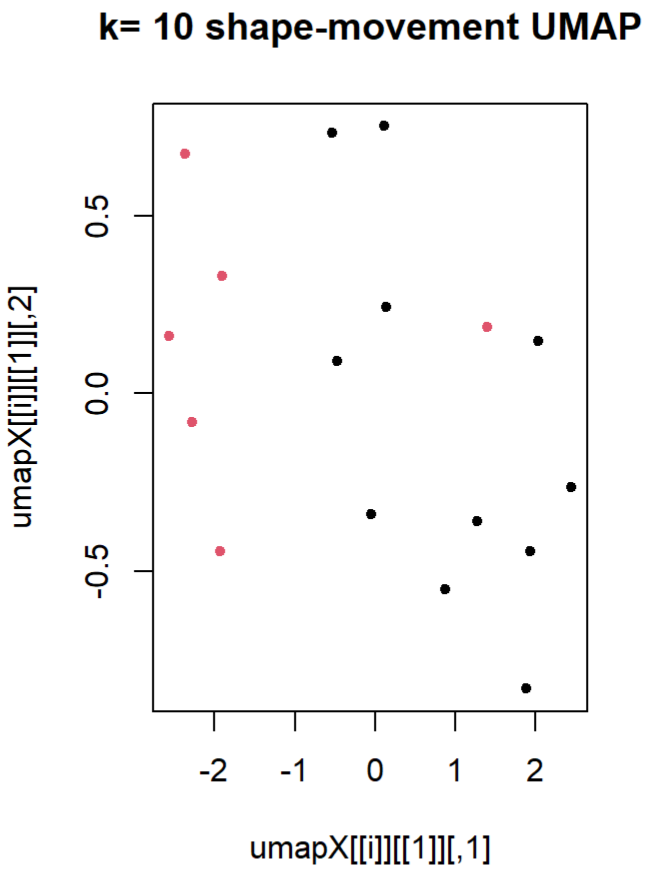


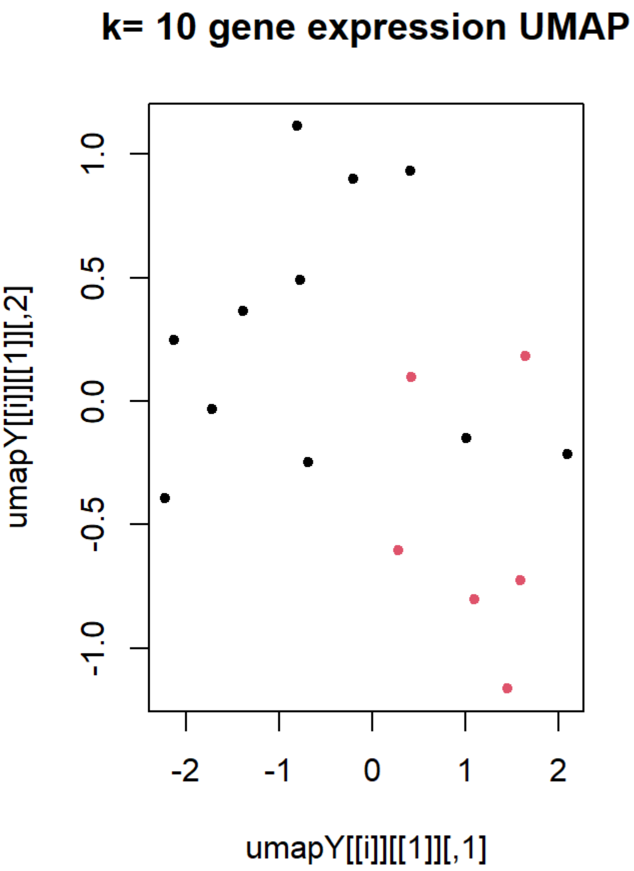
k= 9 shape-movement UMAP

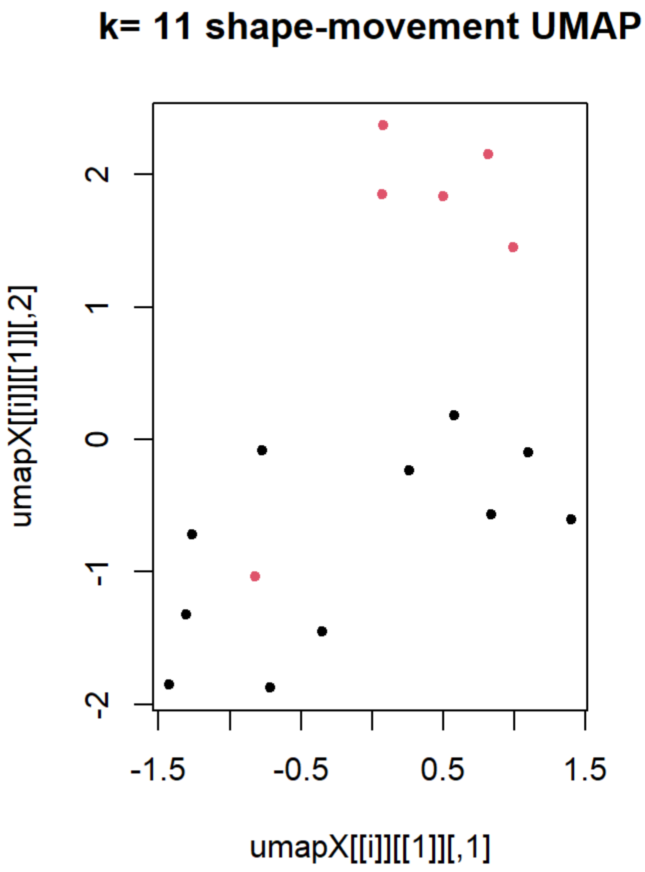


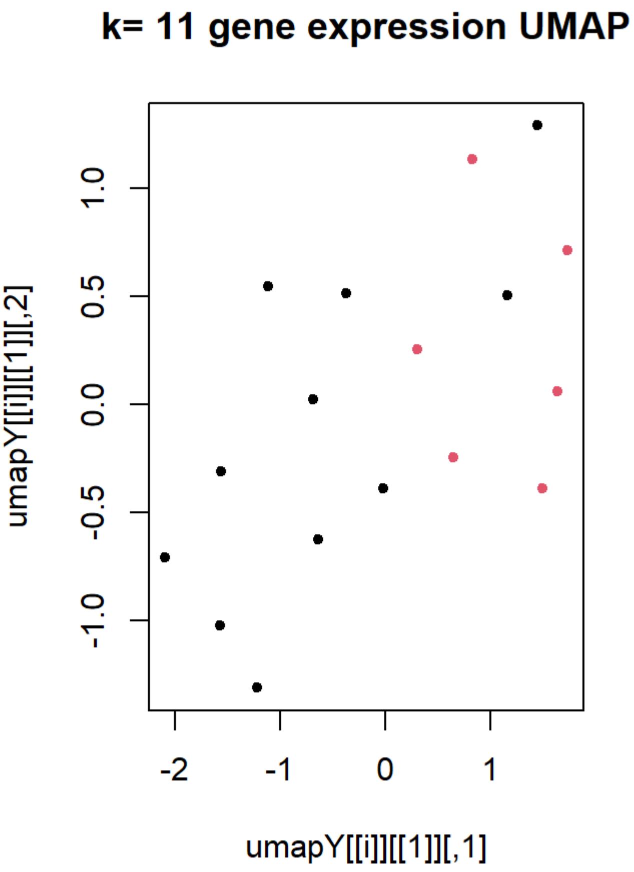
k= 9 gene expression UMAP

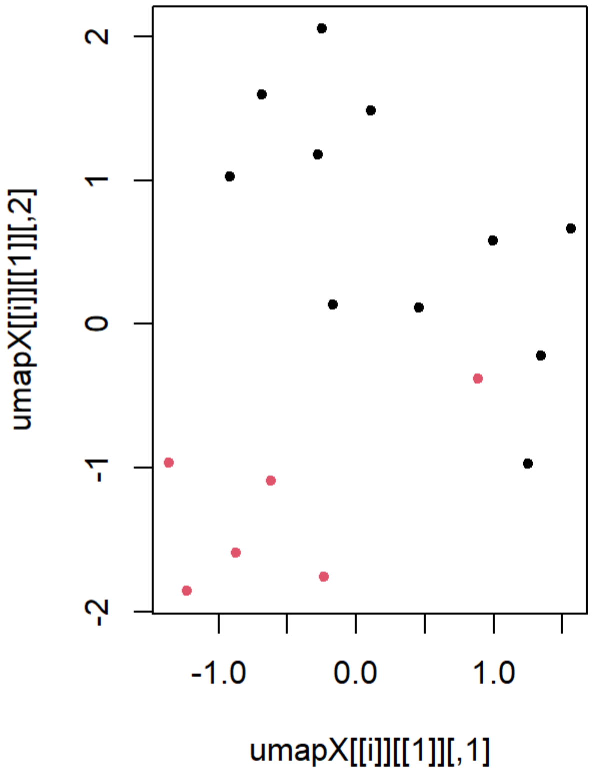



k= 10 shape-movement UMAP
umapX[[i]][[1]][,2]
0.5
0.0
-0.5
-2
-1
0
1
2
umapX[[i]][[1]][,1]

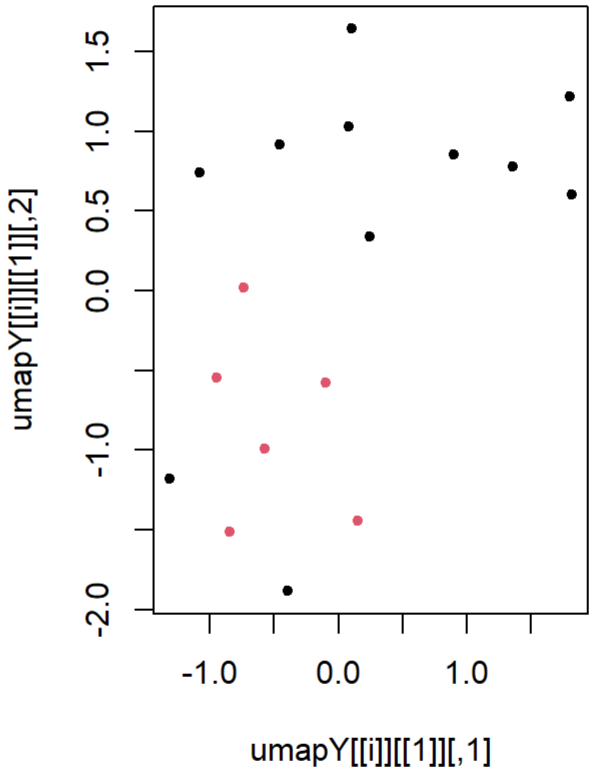

k= 10 gene expression UMAP
umapY[[i]][[1]][,2]
1.0
0.5
0.0
-0.5
-1.0
-2
-1
0
1
2
umapY[[i]][[1]][,1]


k= 11 shape-movement UMAP
umapX[[i]][[1]][,2]
2
1
0
-1
-2
-1.5
-0.5
0.5
1.5
umapX[[i]][[1]][,1]

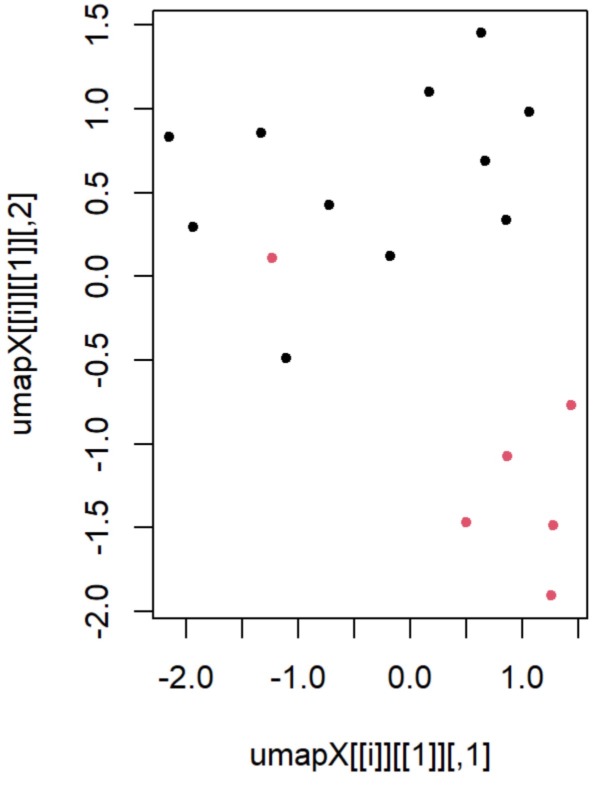

k= 11 gene expression UMAP
umapY[[i]][[1]][,2]
1.0
0.5
0.0
-0.5
-1.0
-2
-1
0
1
umapY[[i]][[1]][,1]



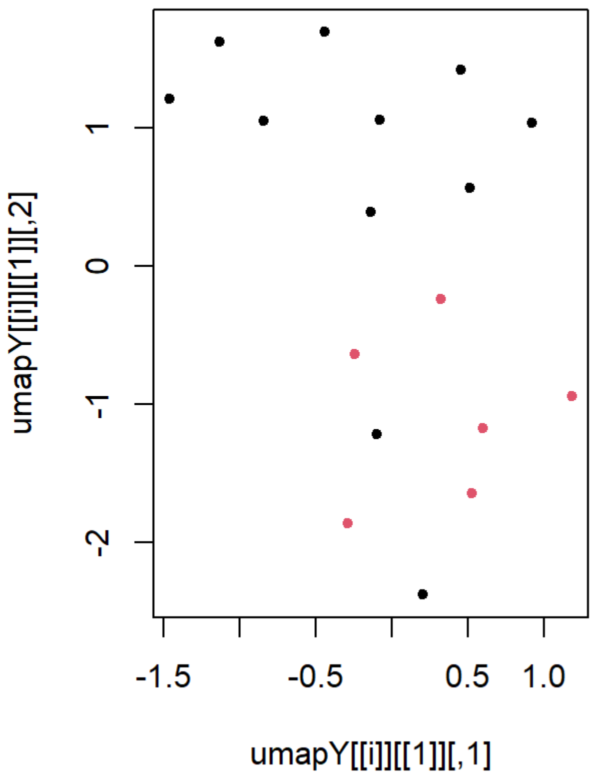
k= 12 shape-movement UMAP



k= 12 gene expression UMAP

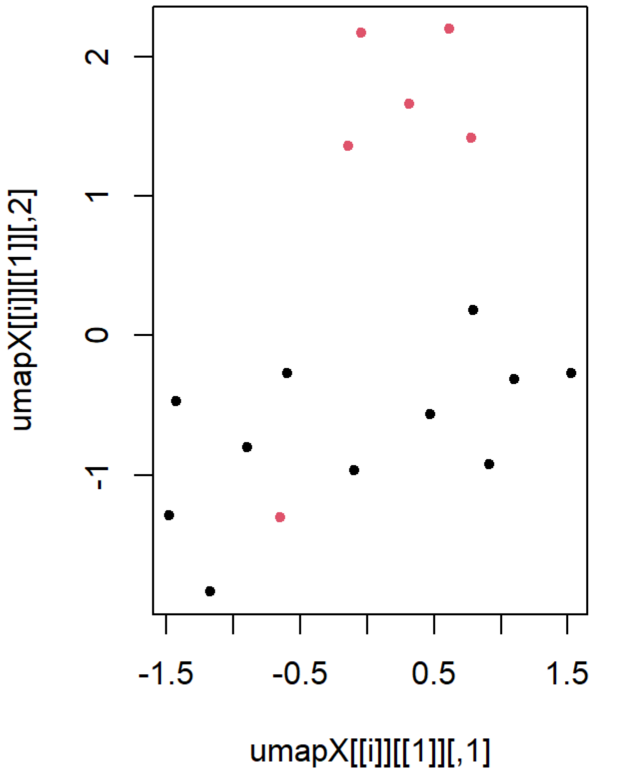


k= 13 shape-movement UMAP

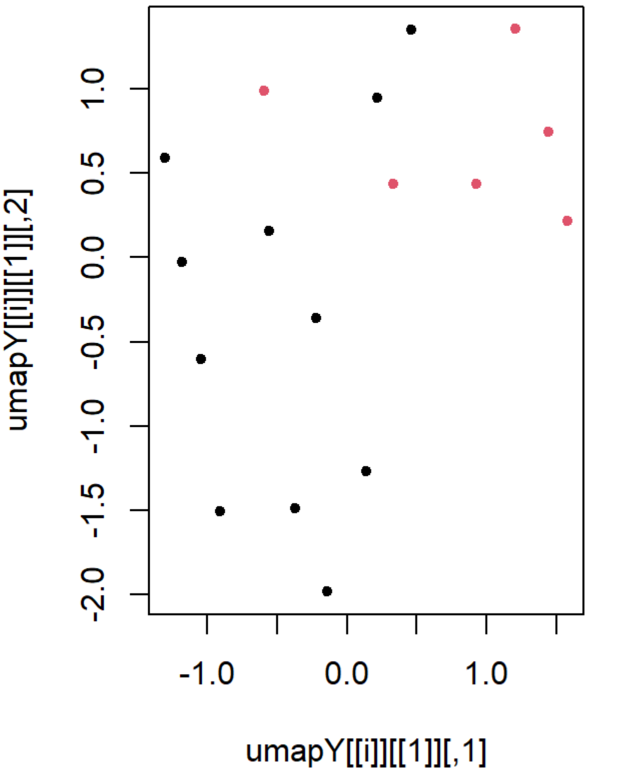


k= 13 gene expression UMAP

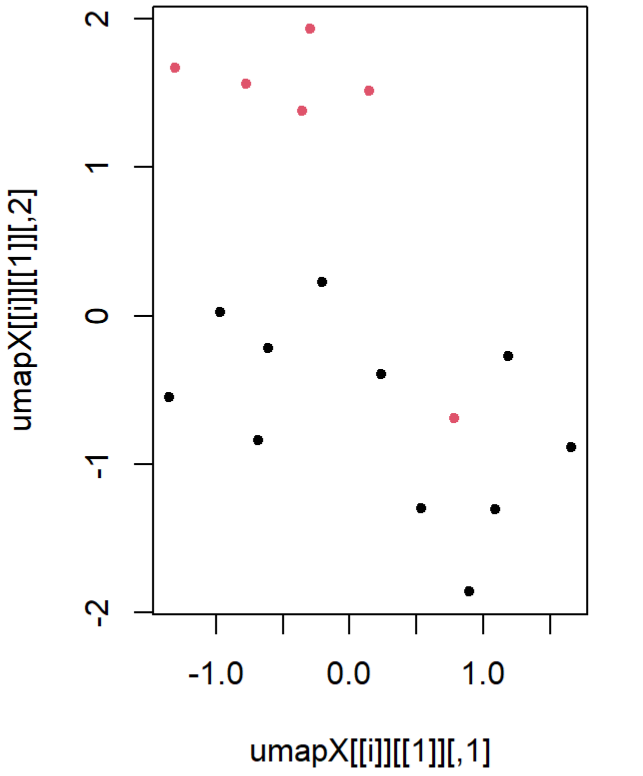
k= 14 shape-movement UMAP



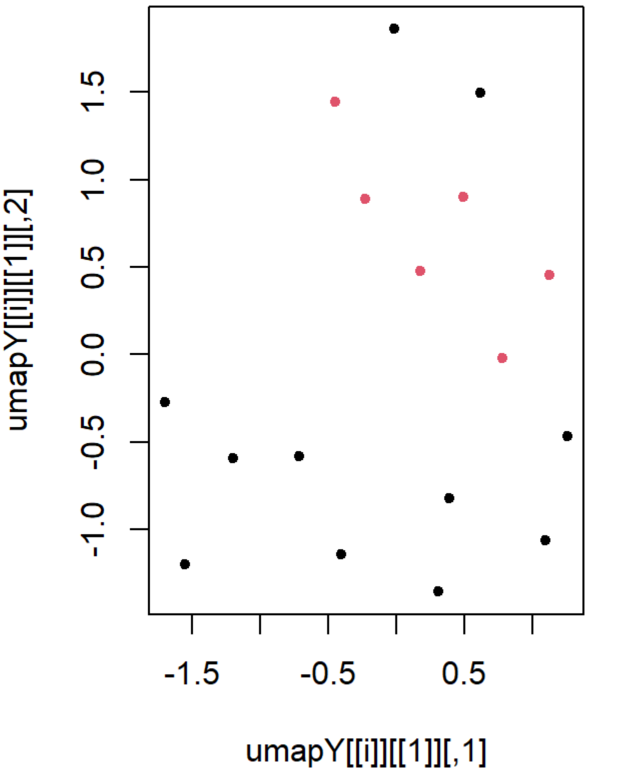
k= 14 gene expression UMAP

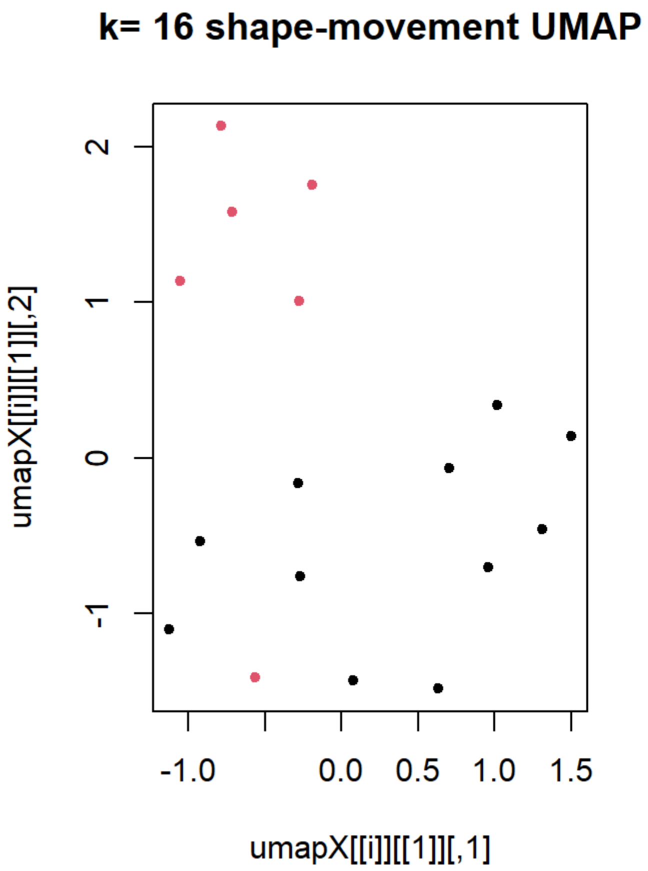


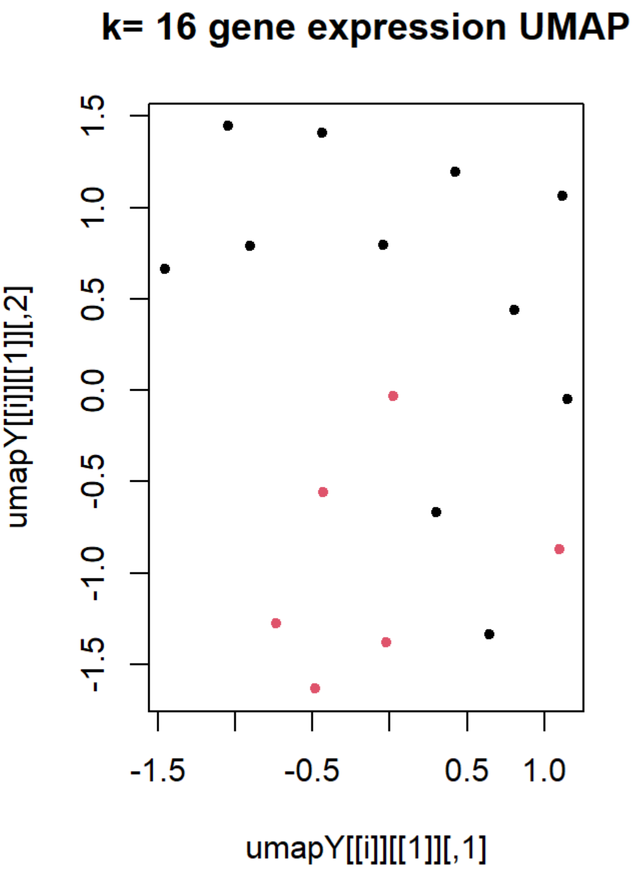
k= 15 shape-movement UMAP

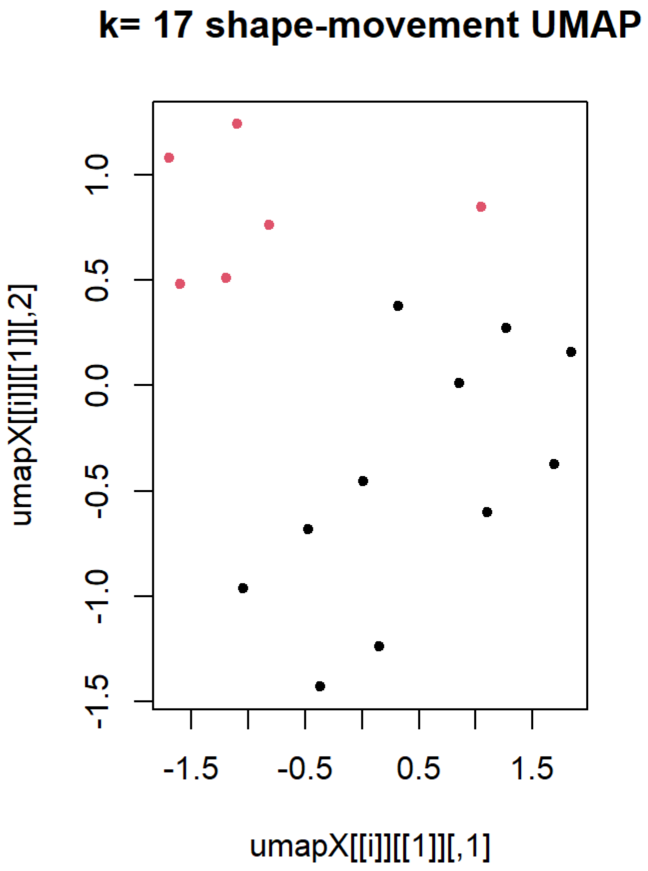


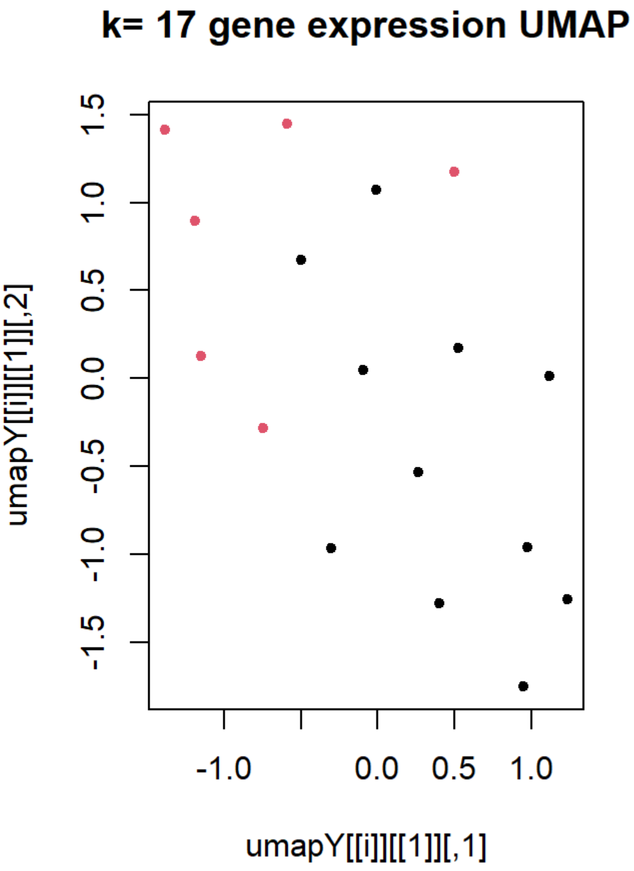
k= 15 gene expression UMAP




k= 16 shape-movement UMAP
umapX[[i]][[1]][,2]
2
1
0
-1
-1.0
0.0
0.5
1.0
1.5
umapX[[i]][[1]][,1]


k= 16 gene expression UMAP
umapY[[i]][[1]][,2]
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-0.5
0.5
1.0
umapY[[i]][[1]][,1]


k= 17 shape-movement UMAP
umapX[[i]][[1]][,2]
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.5
-0.5
0.5
1.5
umapX[[i]][[1]][,1]


k= 17 gene expression UMAP
umapY[[i]][[1]][,2]
1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
-1.0
0.0
0.5
1.0
umapY[[i]][[1]][,1]

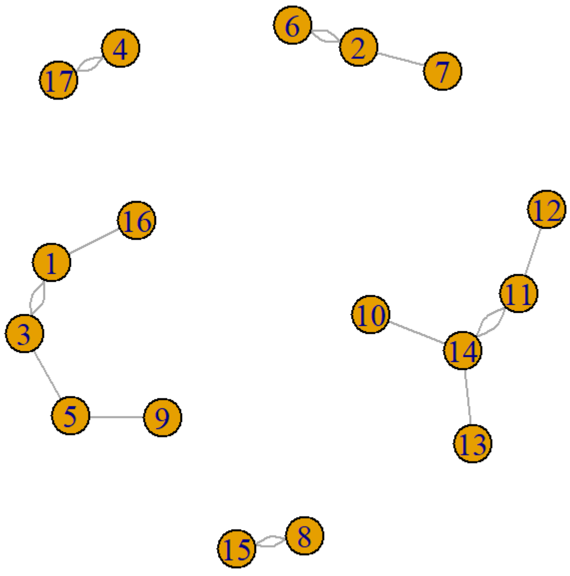
```
graphX <- graphY <- list() # Stockers of graph objects
GraphDistMatX <- GraphDistMatY <- list() # Stockers of graph-distance matrices
for(i in 1:length(ks)){
  # knn element of umap() function output has information of knn-graph and its edge length
  knn_x <- umapX[[i]]$knn
  knn_y <- umapY[[i]]$knn
  k <- ks[i]
  # knn_x has two elements.
  # The 1st element provides which samples are 1st to k-th neighbors
  # The 2nd element provids the length of corresponding edges

  # The following loop make a list of edges in the shape of 2-column matrix
  edge.listX <- edge.listY <- matrix(0,0,2)
  for(j in 1:length(knn_x[[1]][,1])){
    #print(rep(j,k))
    #print(knn_x[[1]][j,1:k])
    # To remove loops (edges starting from one node and ending to the self),
    # [j,2:k] is used rather than [j,1:k]
    edge.listX <- rbind(edge.listX,cbind(rep(j,k-1),knn_x[[1]][j,2:k]))
    edge.listY <- rbind(edge.listY,cbind(rep(j,k-1),knn_y[[1]][j,2:k]))
  }
  # graph.edgelist() function makes a graph object from an edge list.
  # "directed = FALSE" indicates the graph object should be undirected.
  gx <- graph.edgelist(edge.listX,directed=FALSE)
  gy <- graph.edgelist(edge.listY,directed=FALSE)
  # distances() function returns graph disatnce of all node pairs.
  # weights provides every edge length.
  # edge length information is in the 2nd element of knn

  dist.gX <- distances(gx,weights = c(t(knn_x[[2]][,2:k])))
  dist.gY <- distances(gy,weights = c(t(knn_y[[2]][,2:k])))
  graphX[[i]] <- gx
  graphY[[i]] <- gy
  GraphDistMatX[[i]] <- dist.gX
  GraphDistMatY[[i]] <- dist.gY
}
```

knnグラフを描く。

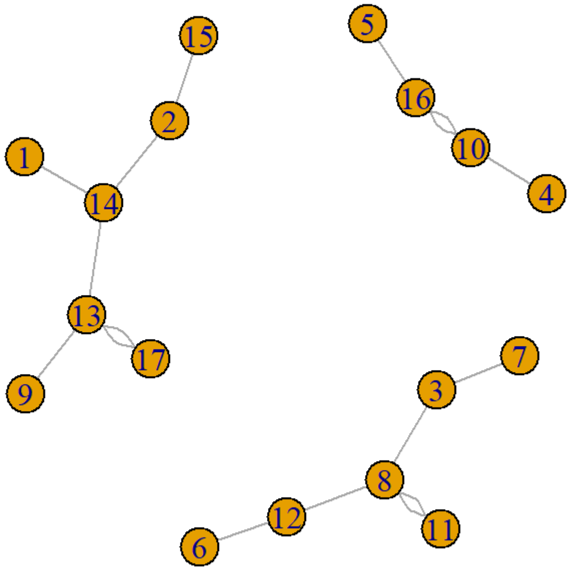
Draw knn graphs.

```
for(i in 1:length(ks)){
  plot(graphX[[i]],main = paste("knn-graph of shape/move, k=", ks[i]))
  plot(graphY[[i]],main = paste("knn-graph of gene expression, k=", ks[i]))
}
```

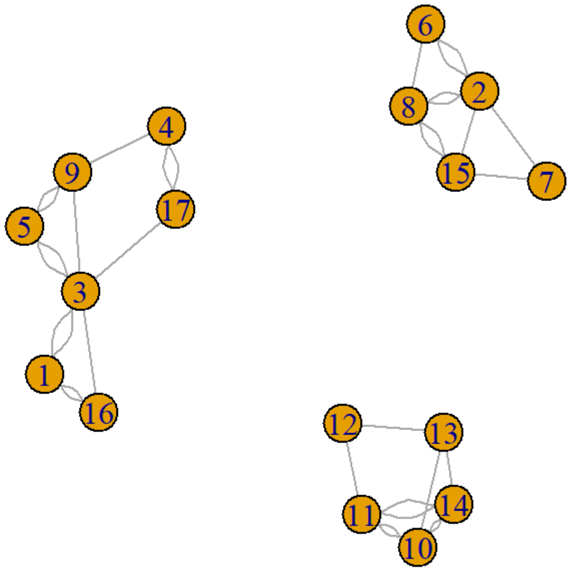
knn-graph of shape/move, k= 2



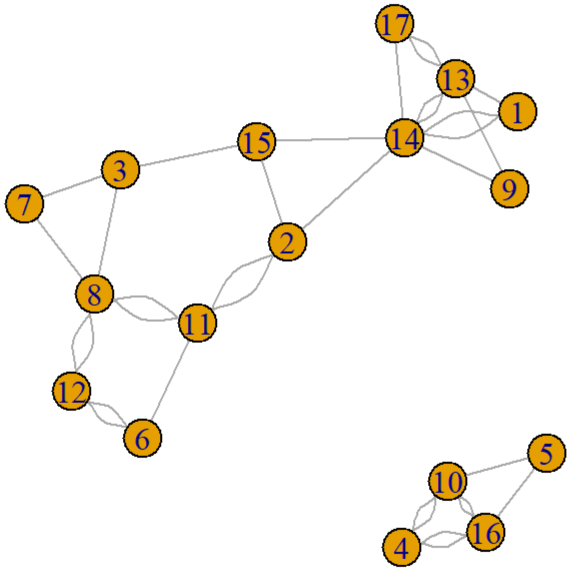
knn-graph of gene expression, k= 2



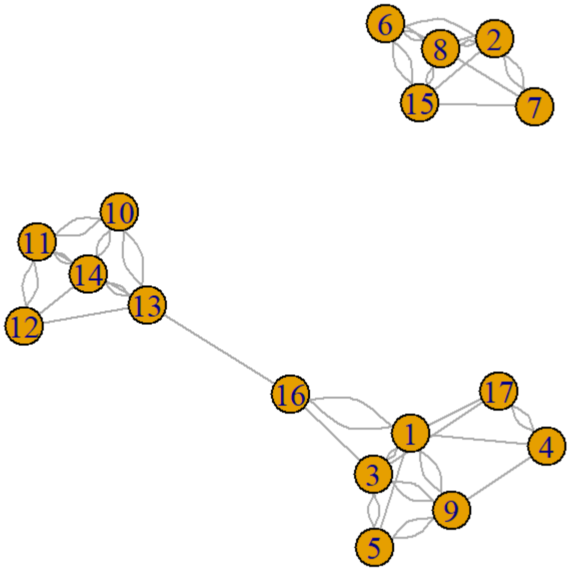
knn-graph of shape/move, k= 3



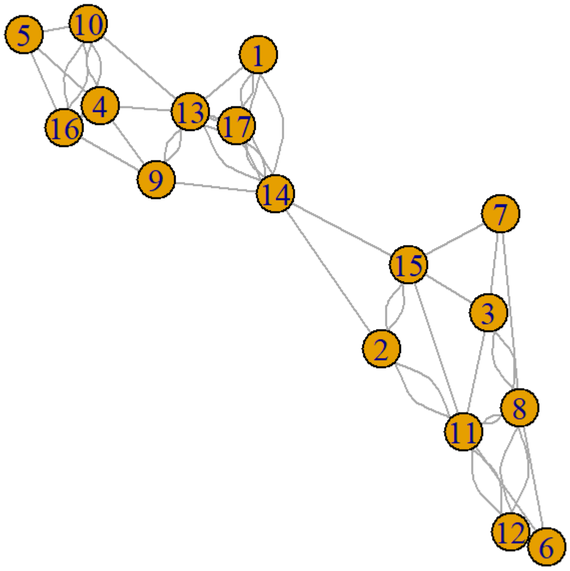
knn-graph of gene expression, k= 3



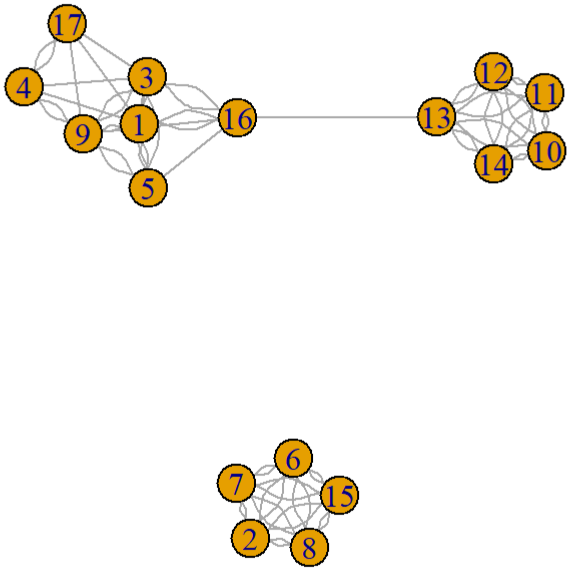
knn-graph of shape/move, k= 4



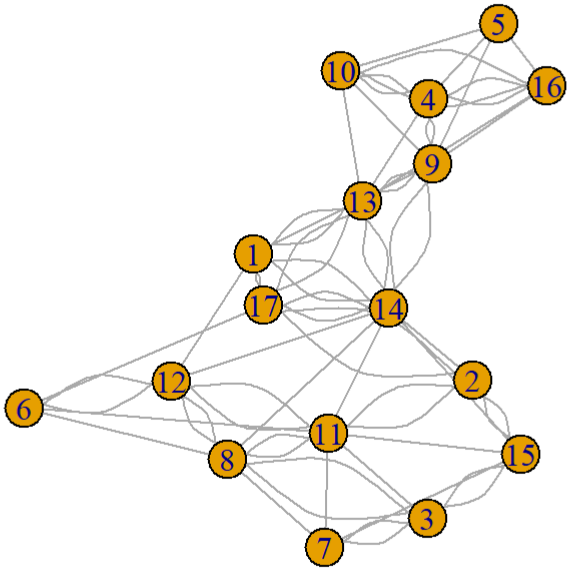
knn-graph of gene expression, k= 4



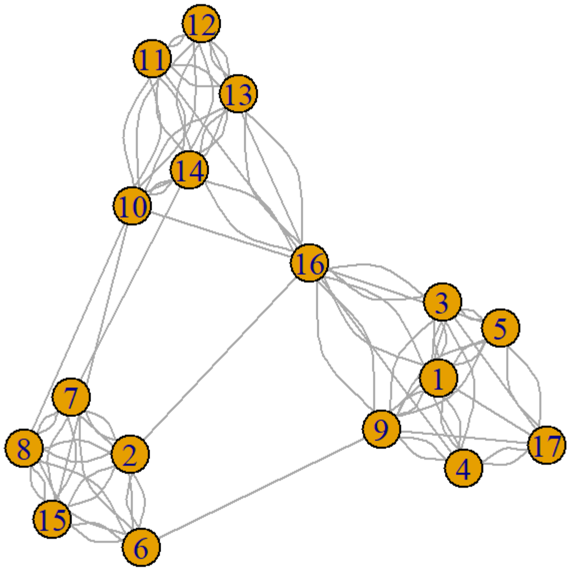
knn-graph of shape/move, k= 5



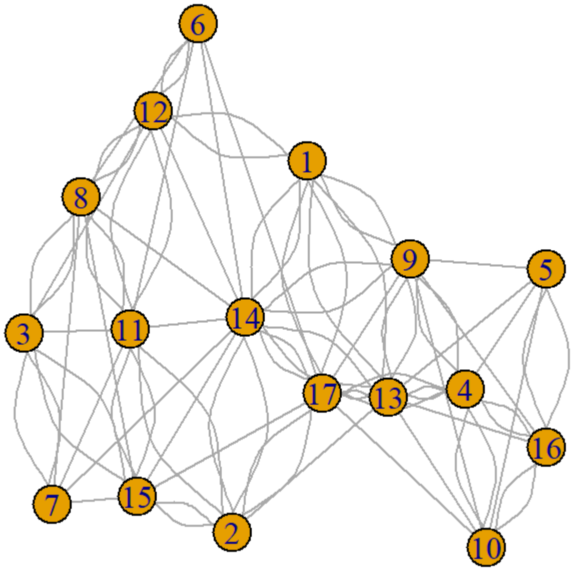
knn-graph of gene expression, k= 5



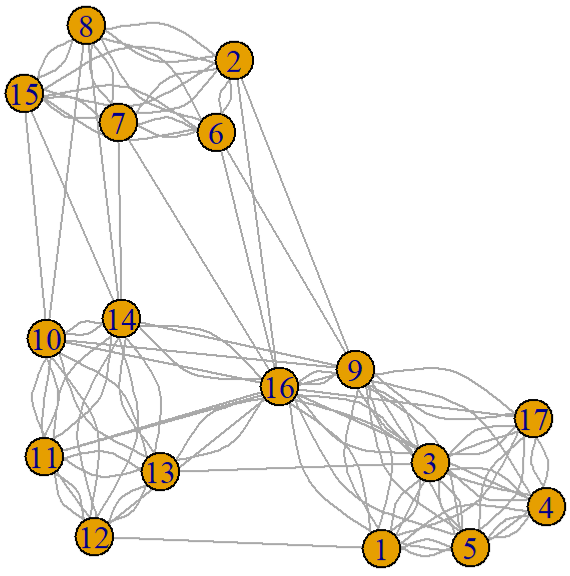
knn-graph of shape/move, k= 6



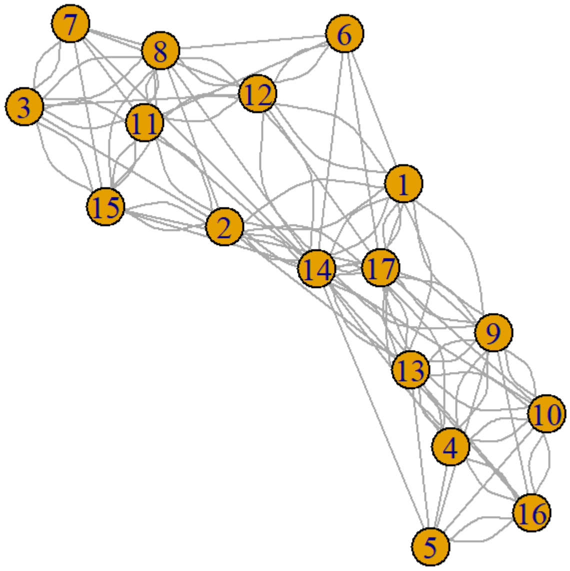
knn-graph of gene expression, k= 6



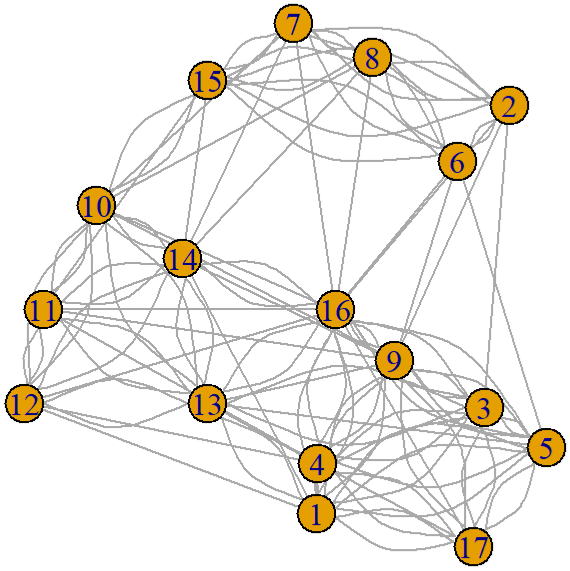
knn-graph of shape/move, k= 7



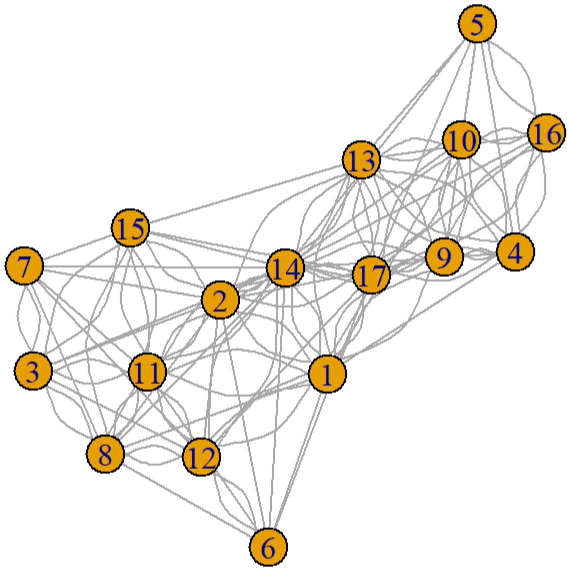
knn-graph of gene expression, k= 7



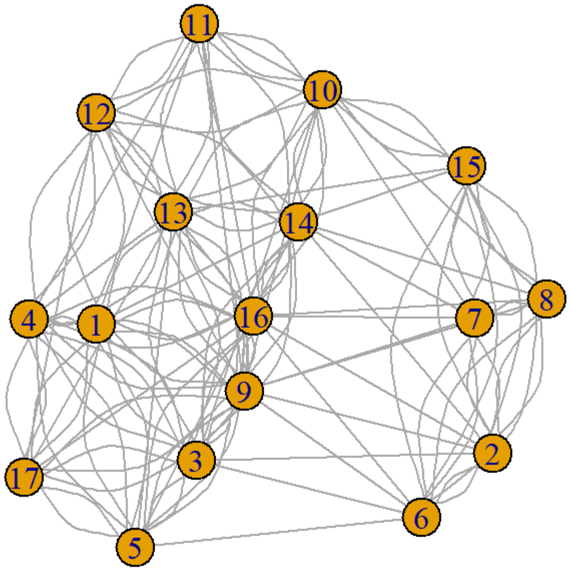
knn-graph of shape/move, k= 8



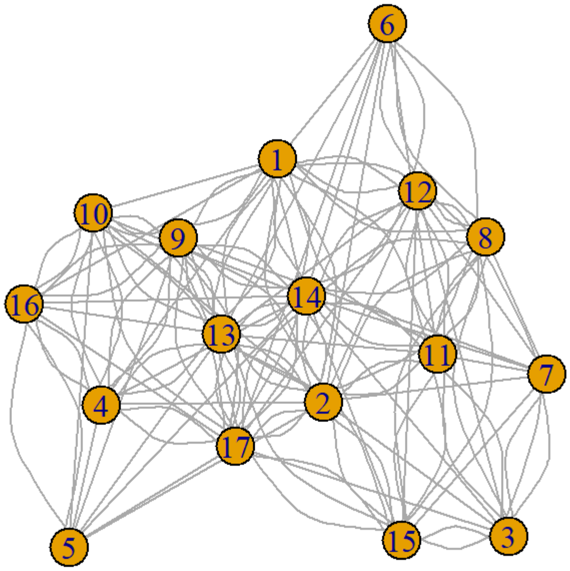
knn-graph of gene expression, k= 8

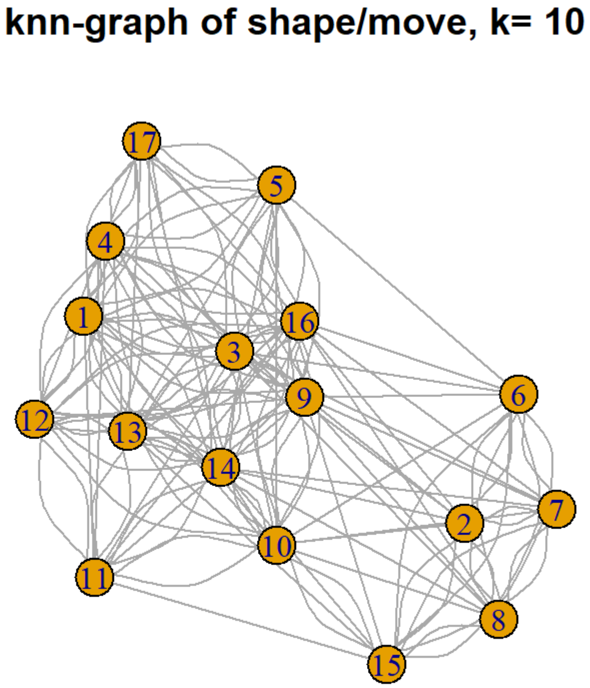


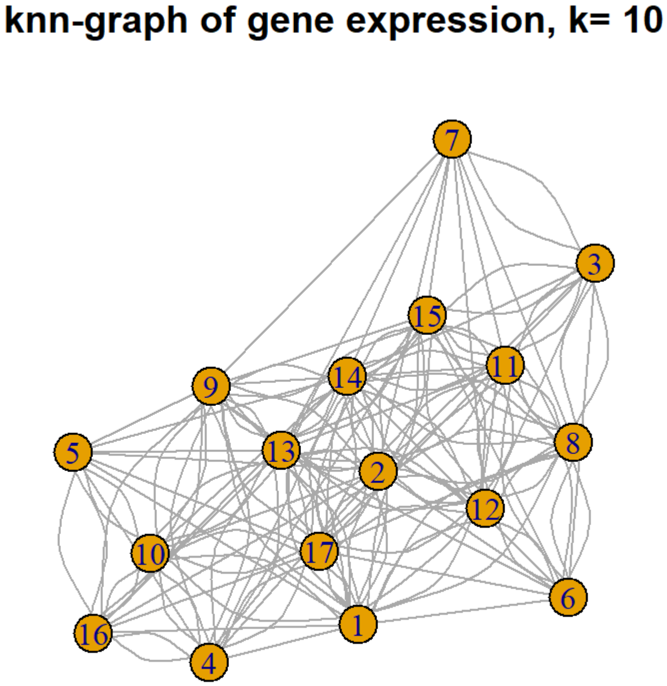
knn-graph of shape/move, k= 9



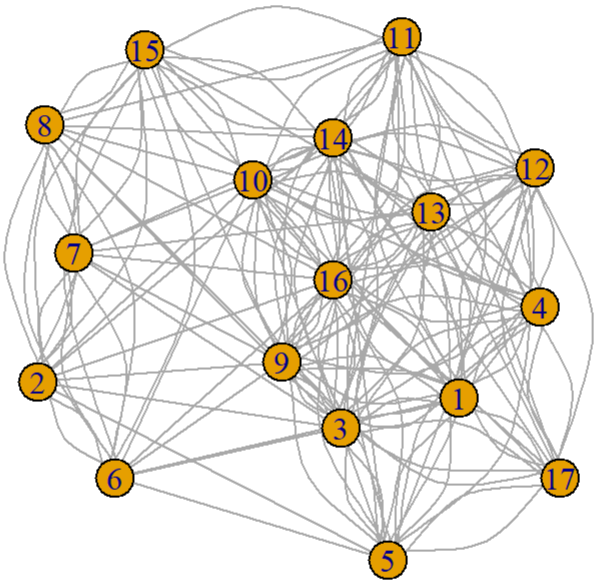
knn-graph of gene expression, k= 9



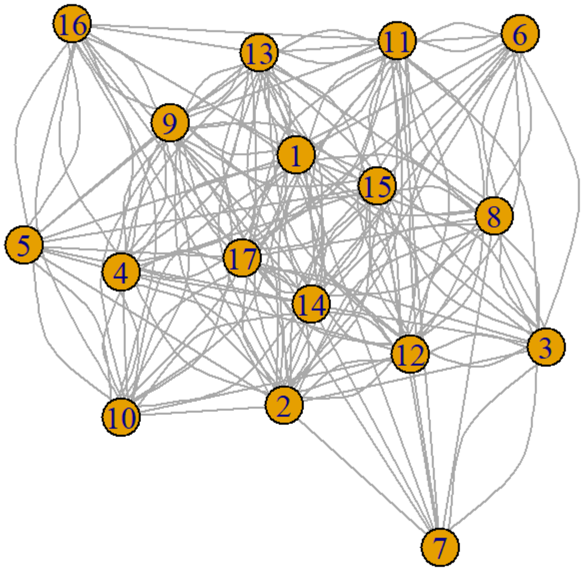

knn-graph of shape/move, k= 10
17
5
4
1
16
3
9
6
12
13
14
7
2
10
11
8
15


knn-graph of gene expression, k= 10
7
3
15
11
9
14
8
5
13
2
12
10
17
6
16
1
4

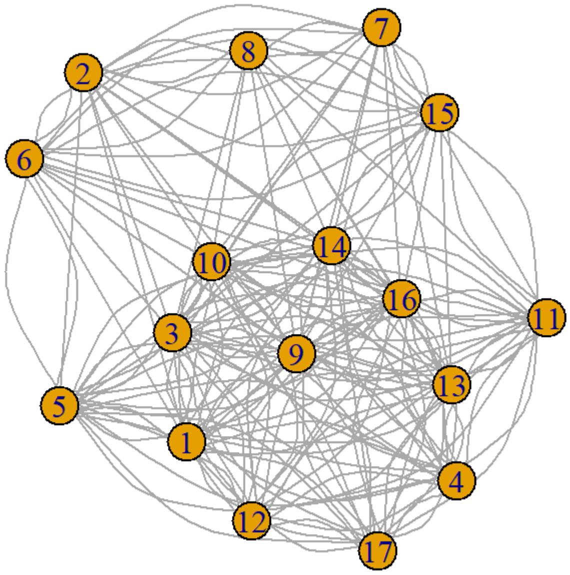
knn-graph of shape/move, k= 11



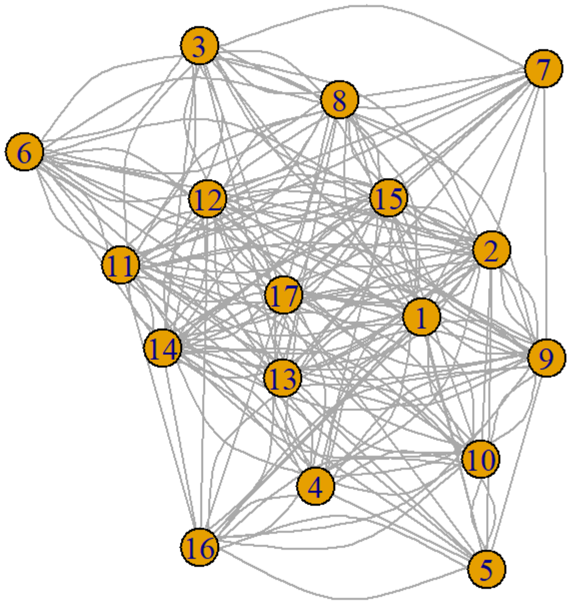
knn-graph of gene expression, k= 11

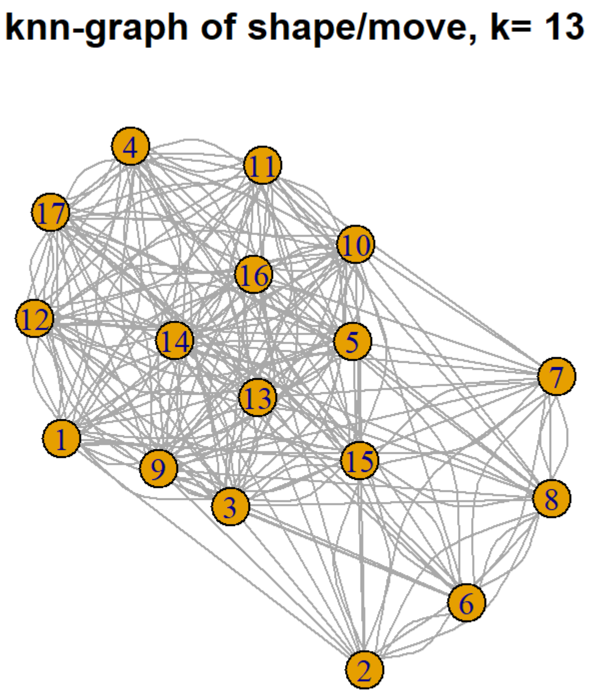


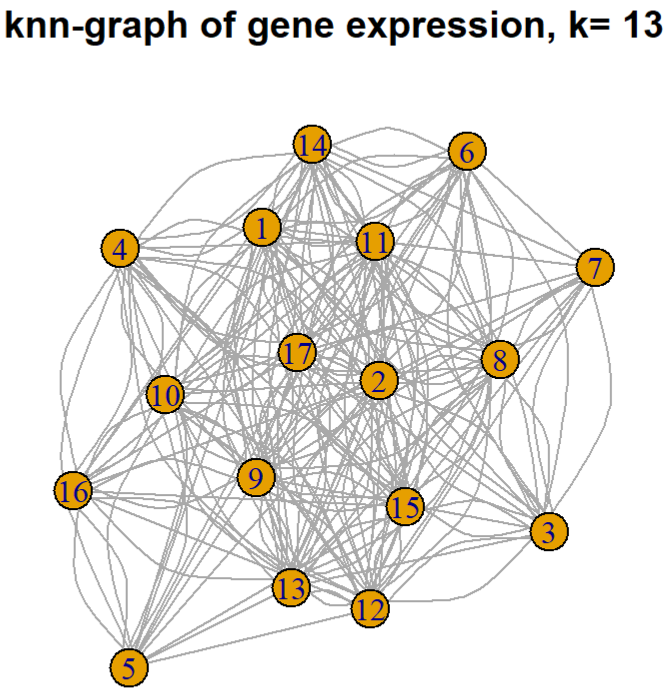
knn-graph of shape/move, k= 12



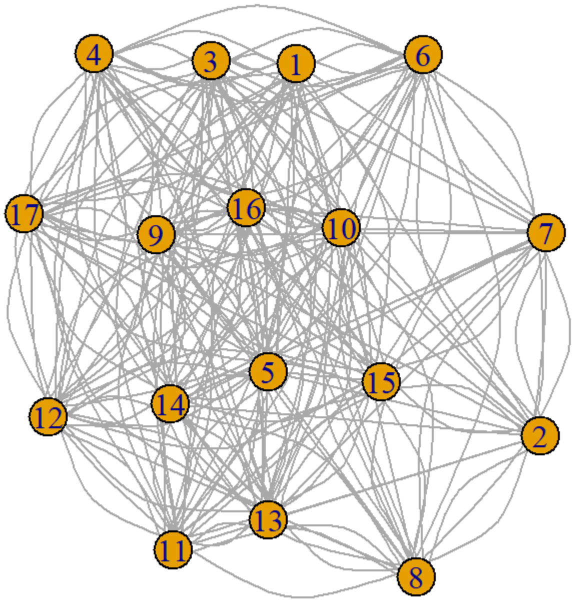
knn-graph of gene expression, k= 12



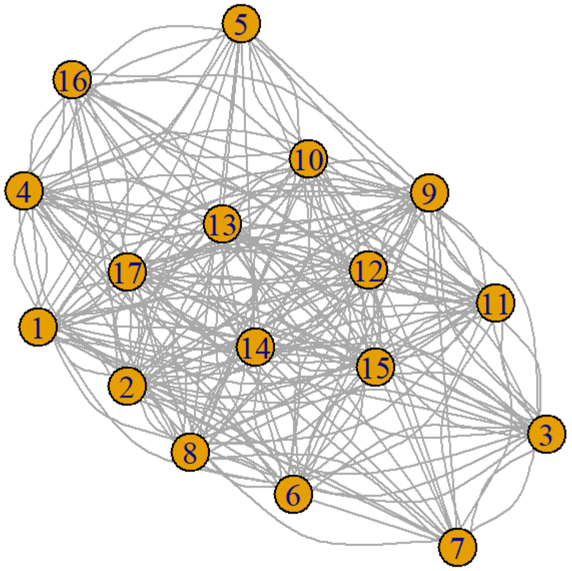

knn-graph of shape/move, k= 13
4
11
17
10
16
12
14
5
7
13
1
9
15
3
8
6
2


knn-graph of gene expression, k= 13
14
6
1
11
4
7
17
2
8
10
9
16
15
3
13
12
5

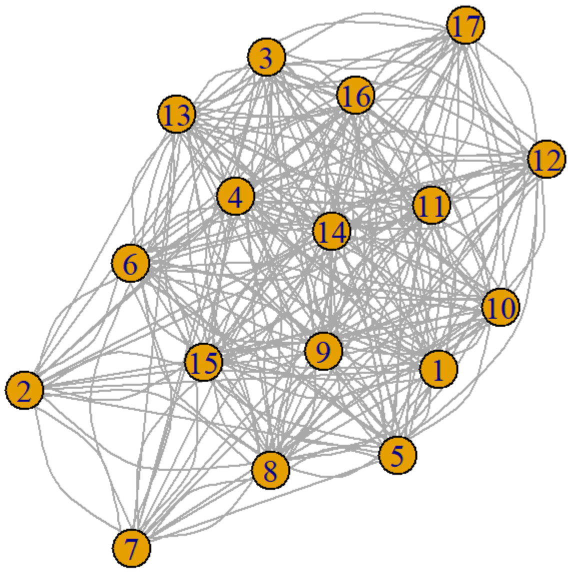
knn-graph of shape/move, k= 14



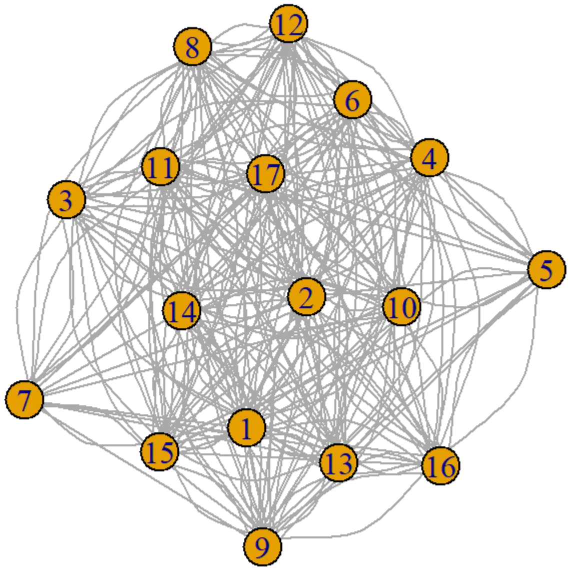
knn-graph of gene expression, k= 14



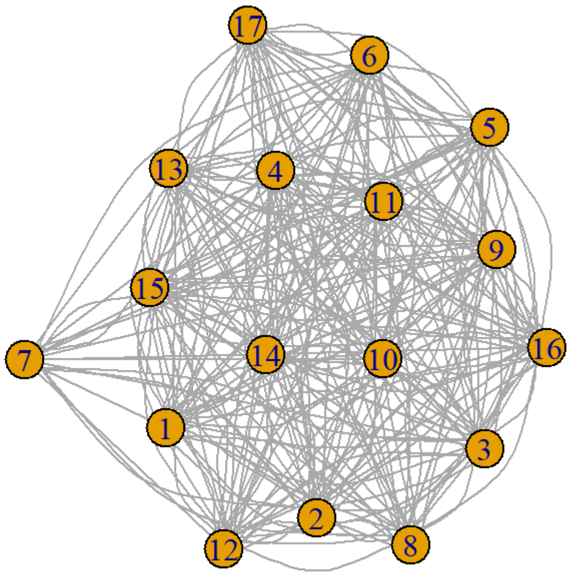
knn-graph of shape/move, k= 15



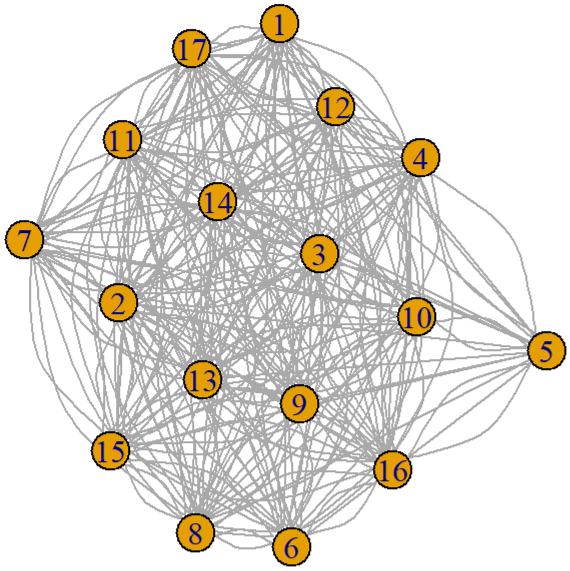
knn-graph of gene expression, k= 15

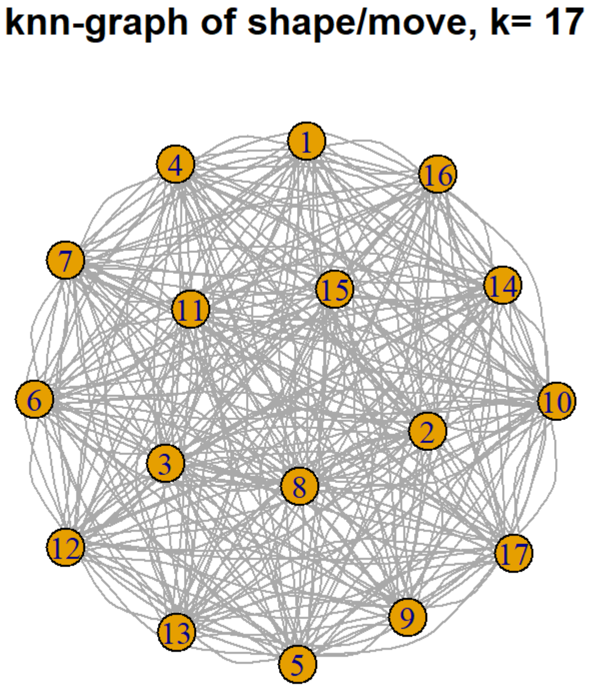


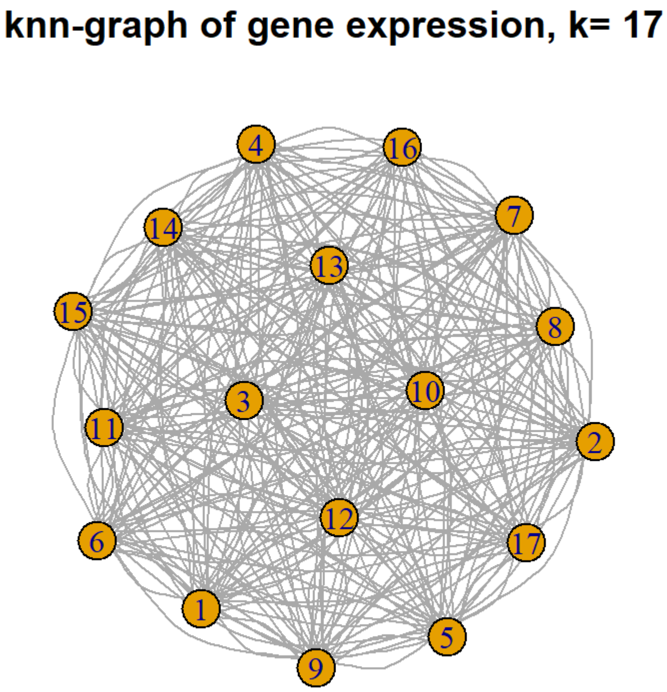
knn-graph of shape/move, k= 16



knn-graph of gene expression, k= 16




knn-graph of shape/move, k= 17
1
4
16
7
15
14
11
6
10
2
3
8
12
17
9
13
5


knn-graph of gene expression, k= 17
4
16
14
7
13
15
8
3
10
11
2
12
6
17
1
5
9

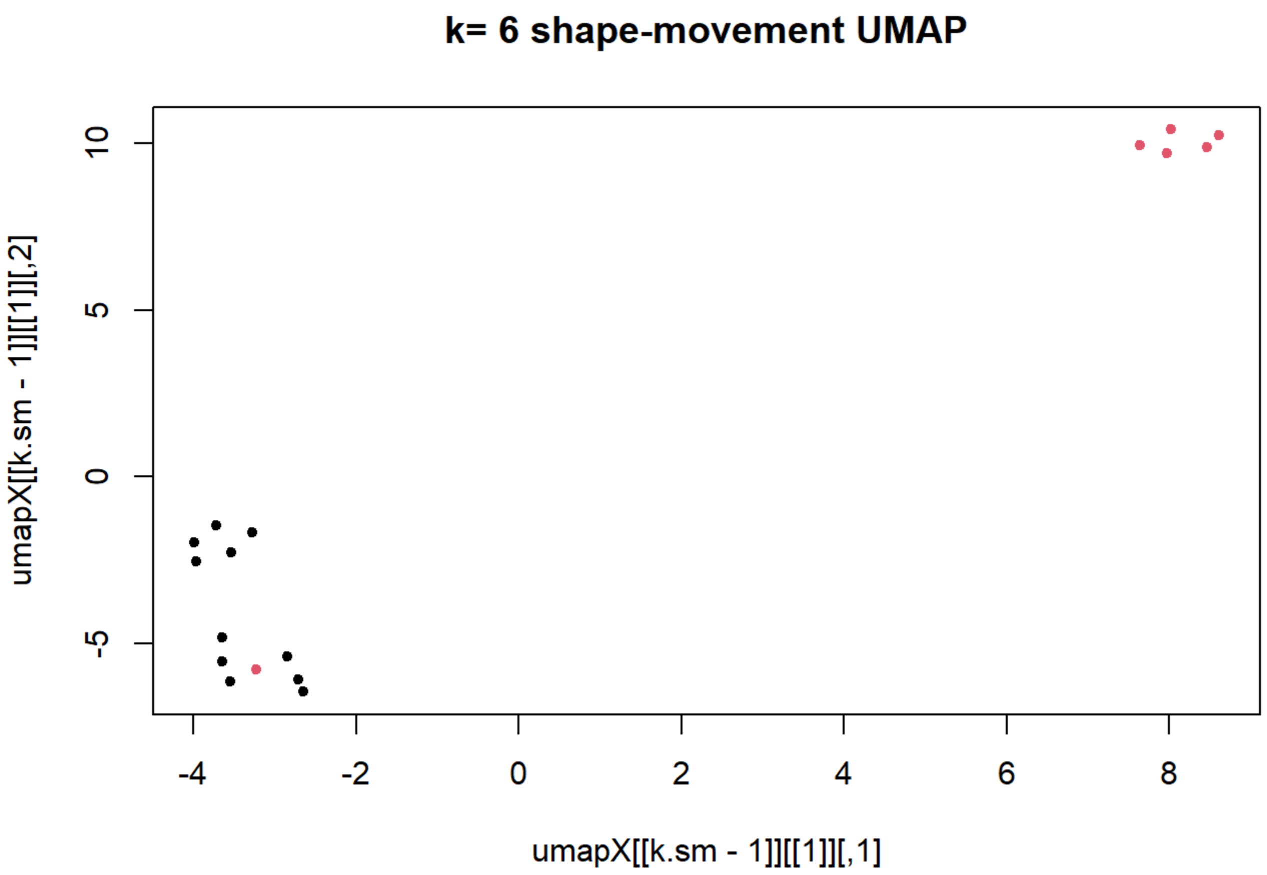
knn グラフの連結性から

```
k.sm <- 6
k.exp <- 4
```

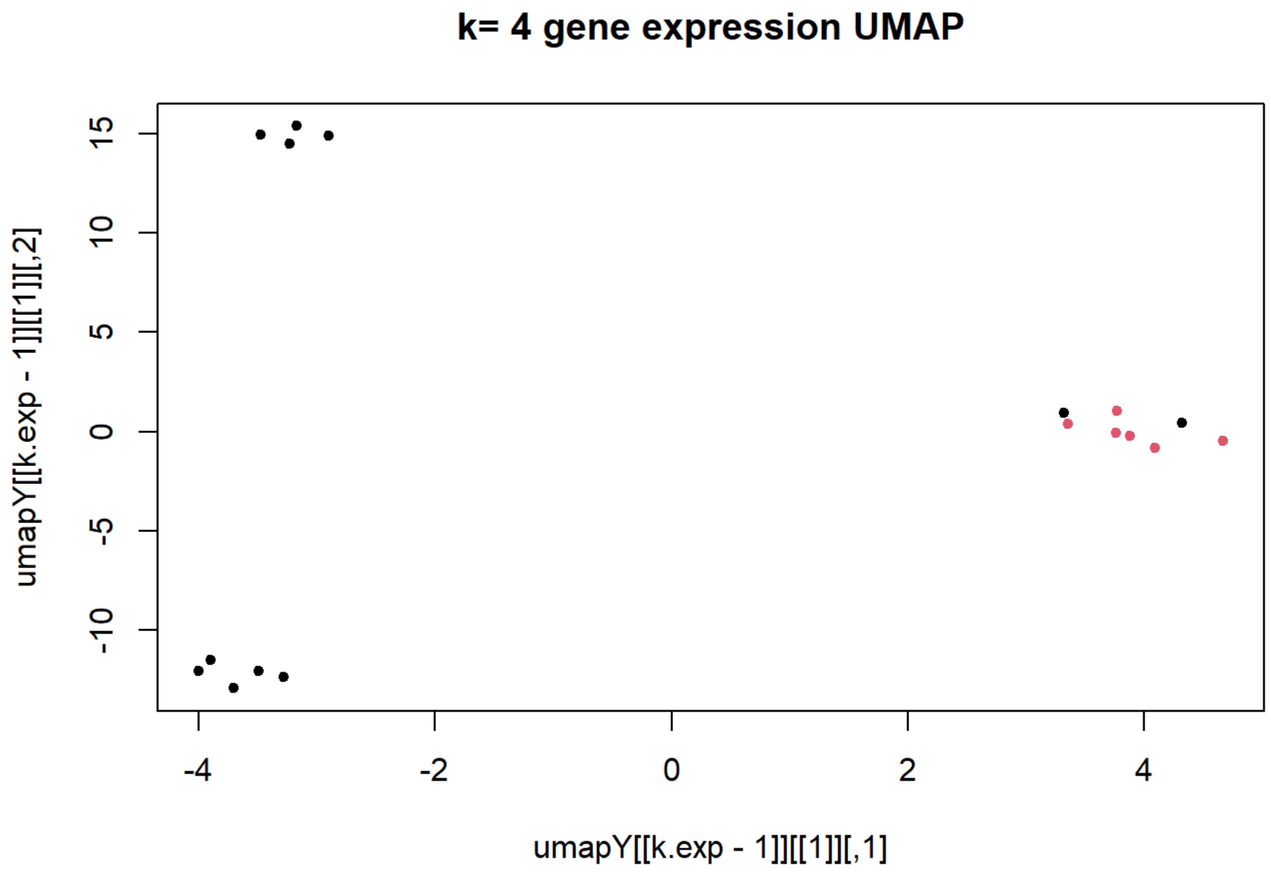
とする。

全対象遺伝子に対してMantelテストを実施する

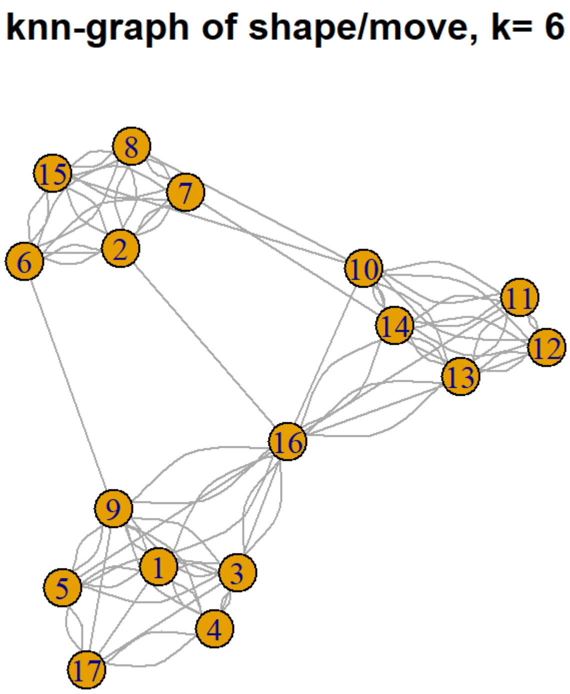
```
plot(umapX[[k.sm-1]][[1]],col=cell_type,pch=20,main=paste("k=",k.sm,"shape-movement UMAP"))
```



```
plot(umapY[[k.exp-1]][[1]],col=cell_type,pch=20,main=paste("k=",k.exp,"gene expression UMAP"))
```



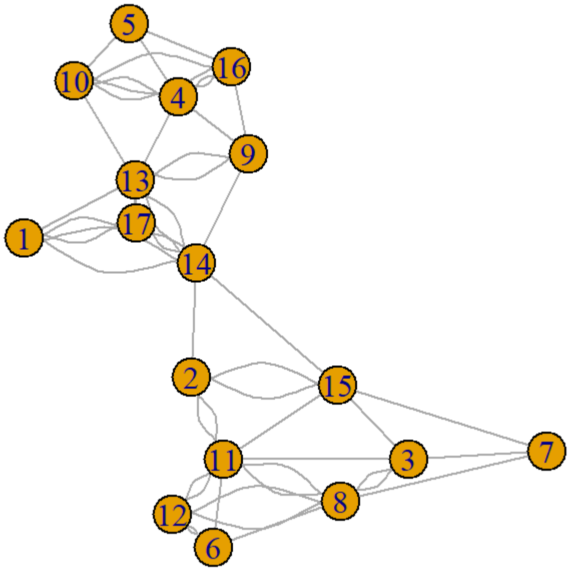
```
plot(graphX[[k.sm-1]],main = paste("knn-graph of shape/move, k=", k.sm))
```



knn-graph of shape/move, k= 6

```
plot(graphY[[k.exp-1]],main = paste("knn-graph of gene expression, k=", k.exp))
```

knn-graph of gene expression, k= 4



```
Mantel.out.6.4<-mantel(GraphDistMatX[[k.sm-1]],GraphDistMatY[[k.exp-1]],method="spearman",permutat
ions=10000,na.rm=TRUE)
Mantel.out.6.4
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## mantel(xdis = GraphDistMatX[[k.sm - 1]], ydis = GraphDistMatY[[k.exp -      1]], method = "spea
rman", permutations = 10000, na.rm = TRUE)
##
## Mantel statistic r: 0.1832
##       Significance: 0.040096
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.124 0.169 0.209 0.268
## Permutation: free
## Number of permutations: 10000
```

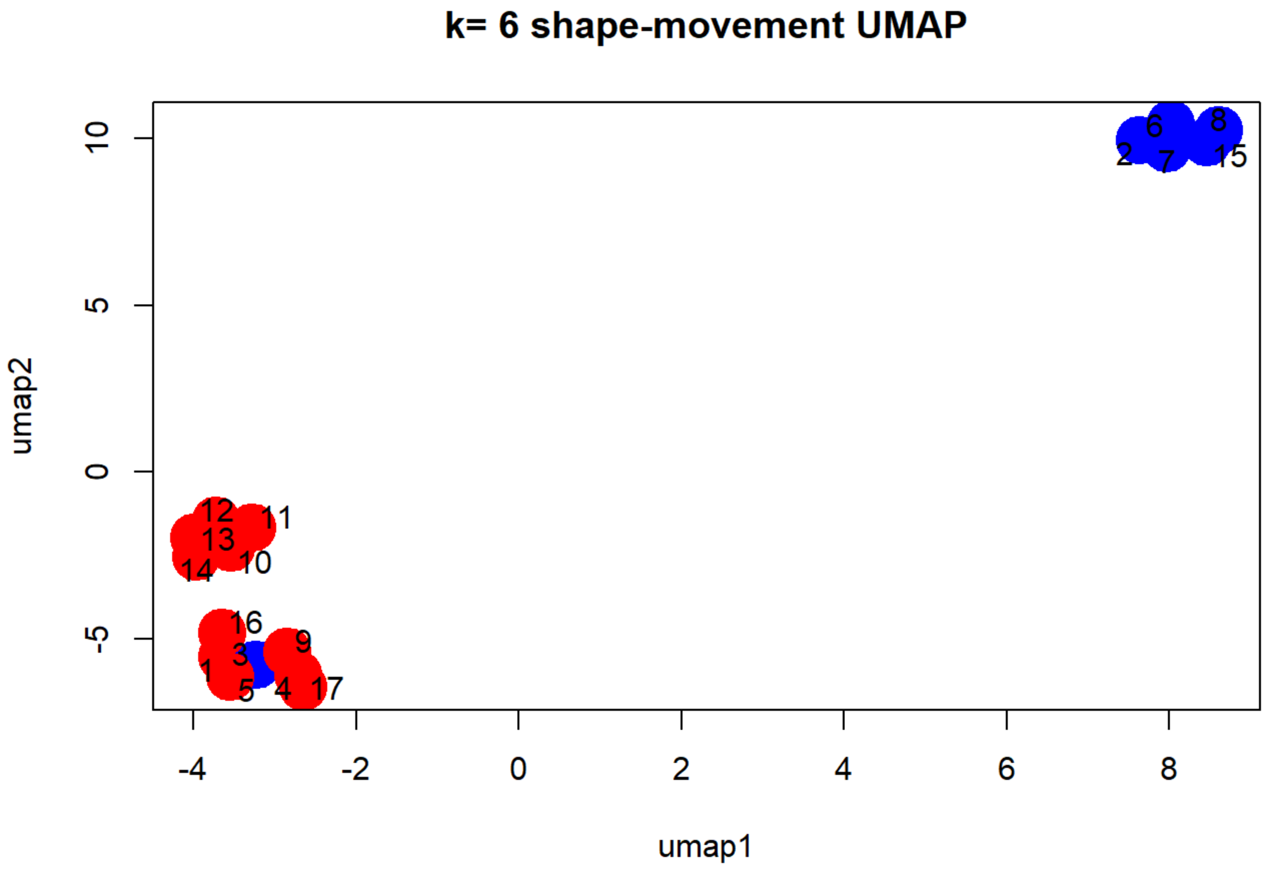
視認ラベル "resting"を赤、"moving"を青で表示する。

```
# dot/vertex color is specified with cell types specified by wet group
library(maptools)
```

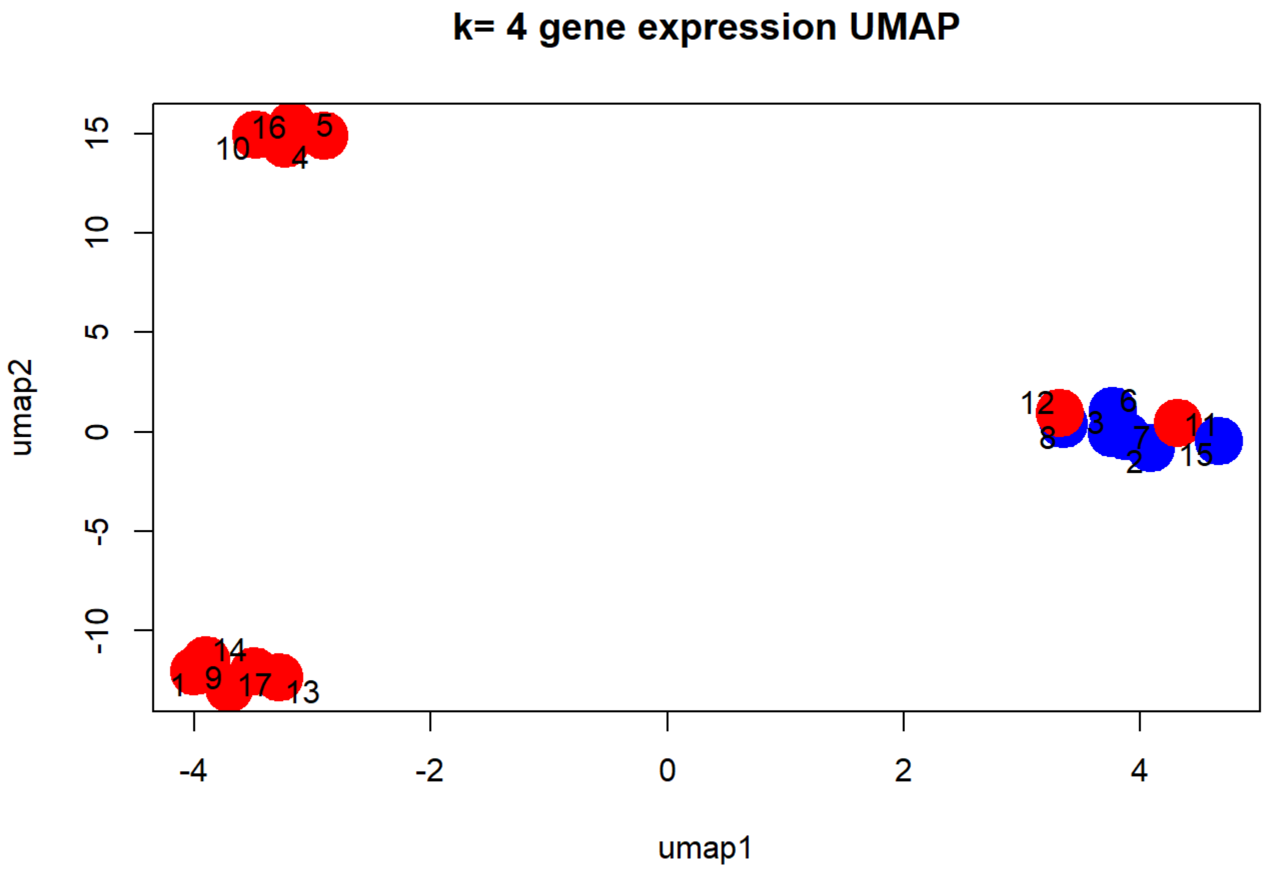
```
## Loading required package: sp
```

```
## Checking rgeos availability: FALSE
##      Note: when rgeos is not available, polygon geometry     computations in maptools depend on
gpclib,
##      which has a restricted licence. It is disabled by default;
##      to enable gpclib, type gpclibPermit()
```

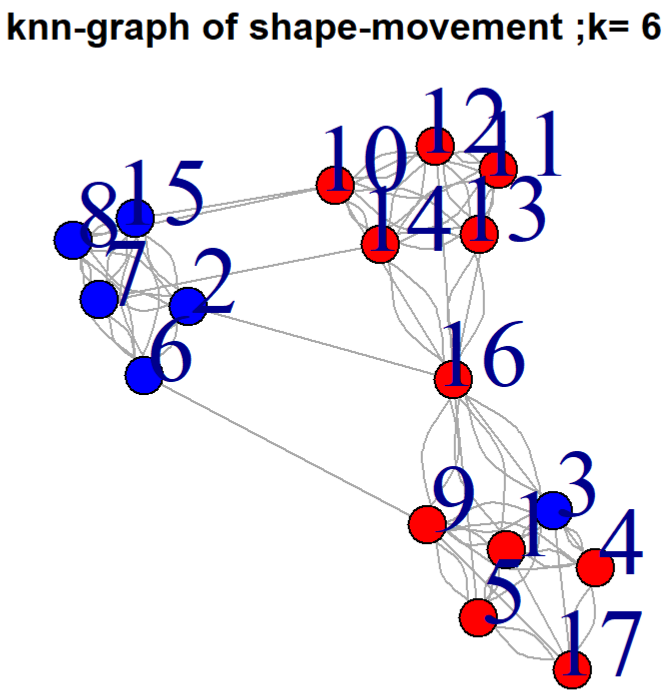
```
graph.color <- cell_type
graph.color[which(graph.color==1)] <- "red"
graph.color[which(graph.color==2)] <- "blue"


plot(umapX[[k.sm-1]][[1]],col=graph.color,pch=20,cex=5,main=paste("k=",k.sm,"shape-movement UMAP"
),xlab="umap1",ylab="umap2")
pointLabel(x=umapX[[k.sm-1]][[1]][,1], y=umapX[[k.sm-1]][[1]][,2], labels=as.character(1:n.cell))
```



```
plot(umapY[[k.exp-1]][[1]],col=graph.color,pch=20,cex=5,main=paste("k=",k.exp,"gene expression UMA
P"),xlab="umap1",ylab="umap2")
pointLabel(x=umapY[[k.exp-1]][[1]][,1], y=umapY[[k.exp-1]][[1]][,2], labels=as.character(1:n.cel
l))
```

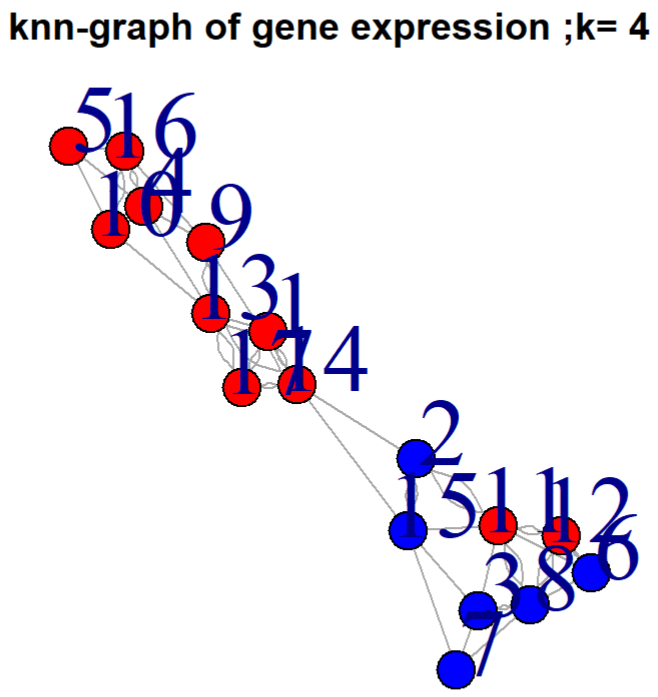


```
plot(graphX[[k.sm-1]],vertex.color=graph.color,main=paste("knn-graph of shape-movement",";k=",k.sm),vertex.label.dist=2,vertex.label.cex=3)
```



knn-graph of shape-movement ;k= 6

```
plot(graphY[[k.exp-1]],vertex.color=graph.color,main=paste("knn-graph of gene expression",";k=",k.exp),vertex.label.dist=2,vertex.label.cex=3)
```

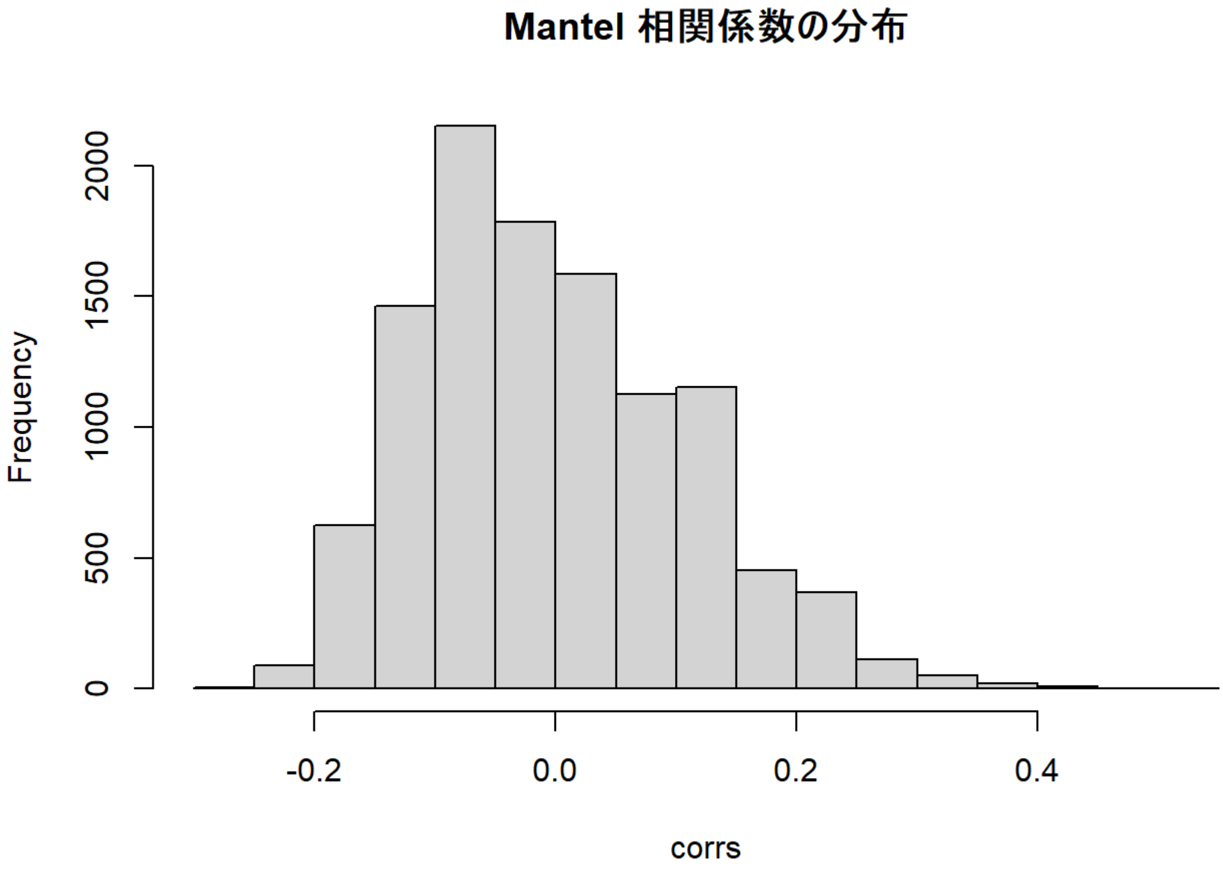


対象遺伝子の個別関連検定と相関係数

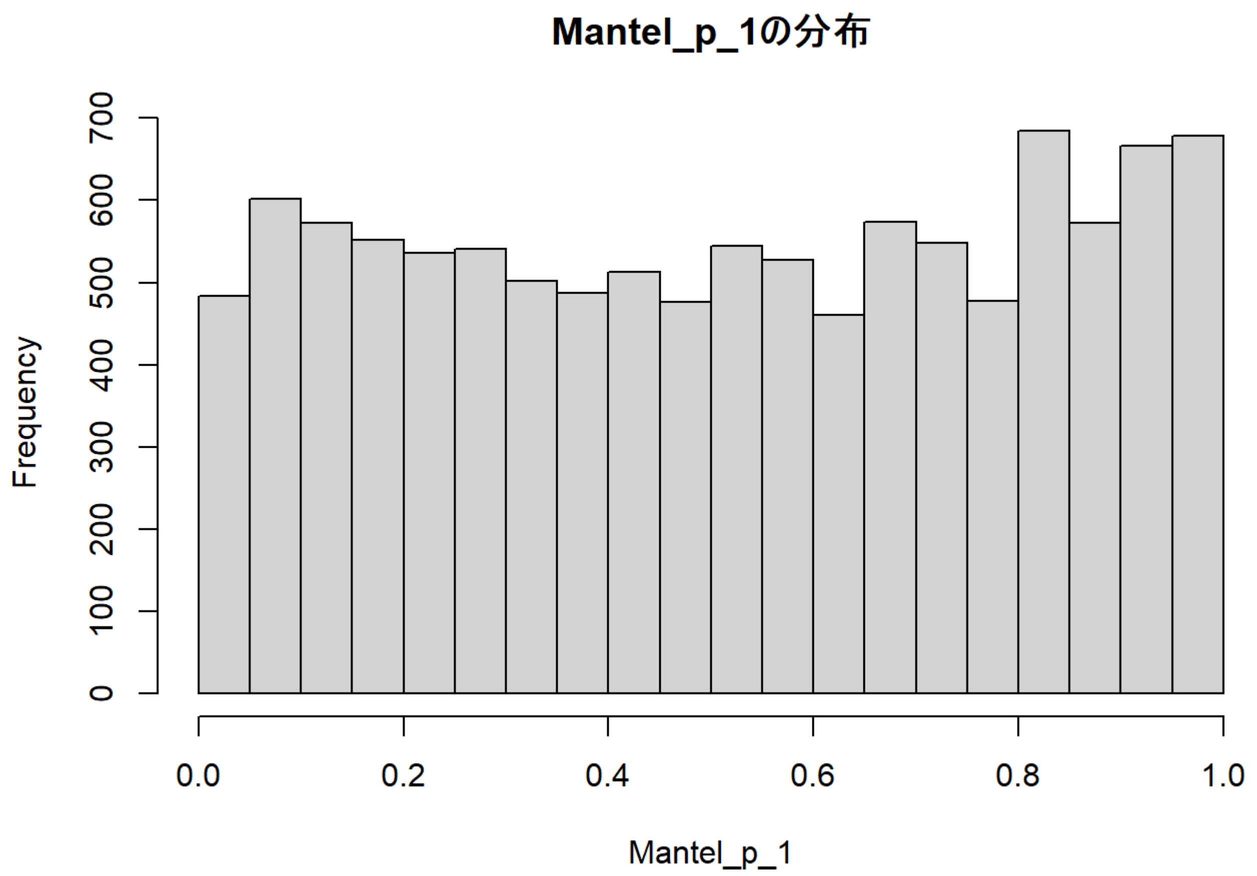
- 方法
- 形動きknn グラフは、全対象遺伝子とのMantelテストに使用したものと同じものを用いる
- 個々の遺伝子のknn グラフは、１変数であるから鎖グラフとする
- 全対象遺伝子につき、Mantel相関係数と、パーミュテーション回数1000での暫定的なMantel_p値(Matel-p_1)を出す
- Mantel_p_1 < 0.003 である遺伝子については、上記の方法で算出したMantel-p値が不正確であり、さらに小さなMantel-p値である可能性が高いと判断し、再度、パーミュテーション回数を100000に増やして、Mantel_p値を算出し直す。これをMantel_p_2とする
- Mantel_p値(Mantel_p_2値がある遺伝子はそれを、ない遺伝子はMantel_p_1値をそれとする)のセットが得られるので、それに対してFDR法(Benjamini, Hochberg)で補正し、それをFDR-q値とする

全対象遺伝子のMantel相関係数とMantel_p_1の分布を確認する。

```
hist(corrs,main="Mantel 相関係数の分布")
```



```
hist(Mantel_p_1,main="Mantel_p_1の分布")
```



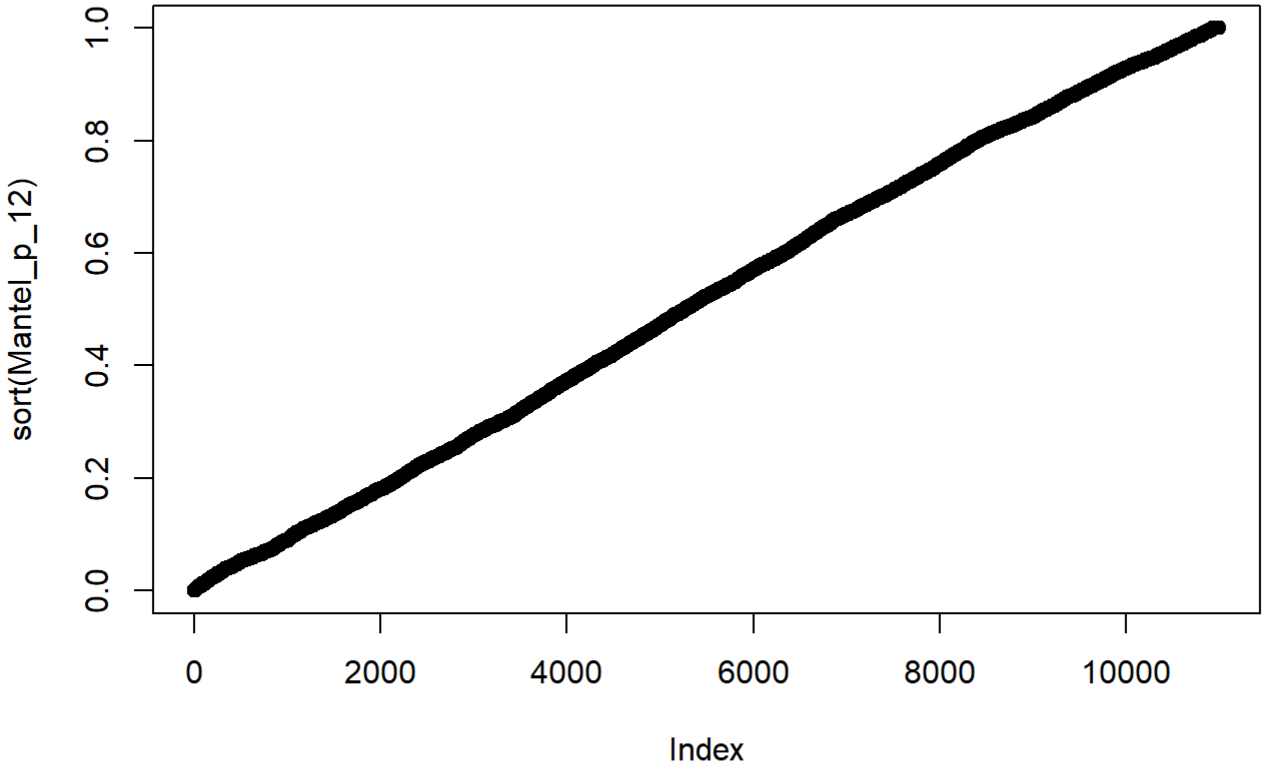
Mantel_p_1 < 0.002 の遺伝子につき、Mantel_p_2を計算する

```
selected_genes <- which(Mantel_p_1 < 0.002)
length(selected_genes)
```

```
## [1] 17
```

```
Mantel_p_2 <- rep(0,length(selected_genes))
dist_mat_x <- GraphDistMatX[[k.sm-1]]
for(i in 1:length(selected_genes)){
  this.gene <- data_y[,selected_genes[i]]
  if(sum(this.gene)==0){

  }else{
    dist_mat_y_gene <- as.matrix(dist(this.gene))
  tmp.result<-mantel(dist_mat_x,dist_mat_y_gene,method="spearman",permutations=100000,na.rm=TRUE)
  #corrs[i] <- tmp.result$statistic
  Mantel_p_2[i] <- tmp.result$signif
  }
}
```

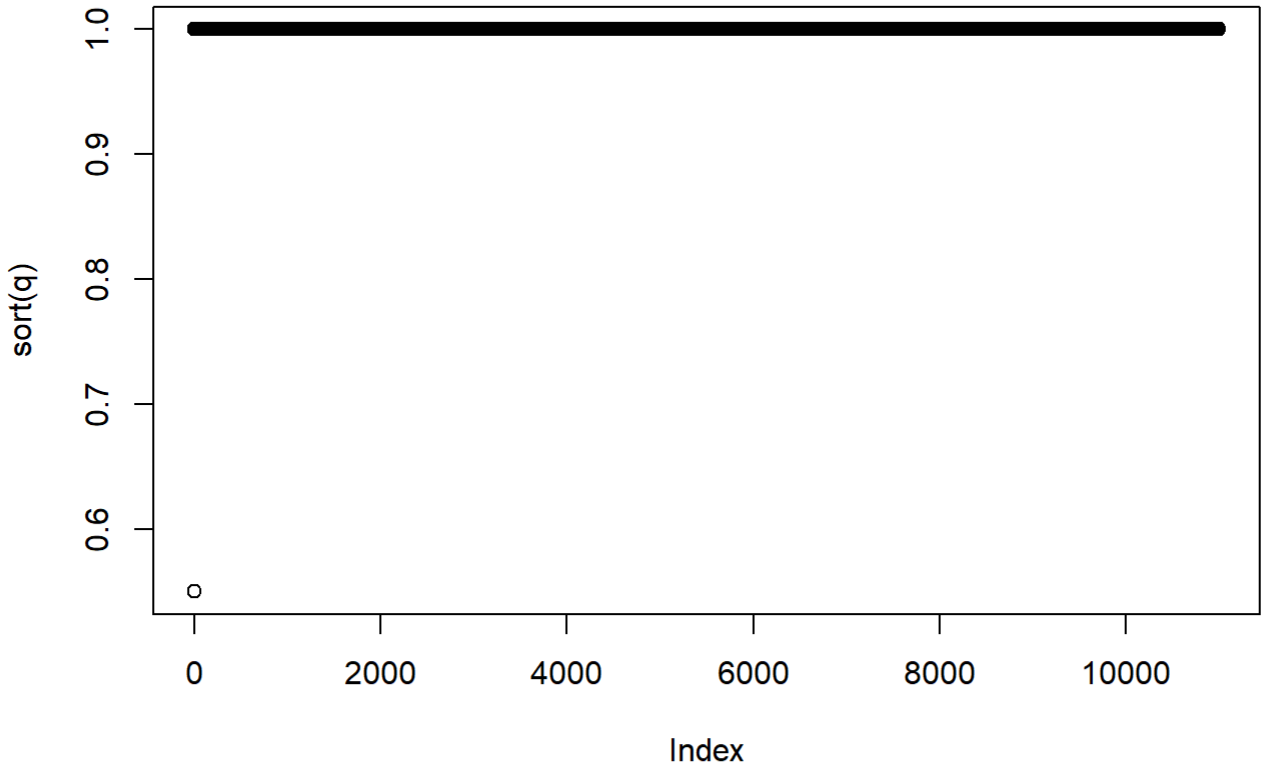
```
Mantel_p_12 <- Mantel_p_1
Mantel_p_12[selected_genes] <- Mantel_p_2
plot(sort(Mantel_p_12))
```



FDR

```
q <- p.adjust(Mantel_p_12)
```

```
plot(sort(q))
```



結果出力

```
ret <- data.frame(gene.id=gene.id.non0s,Mantel_p_12=Mantel_p_12,Mantel_p_1 = Mantel_p_1, q.FDR=q,corrs=corrs)

write.table(ret,file="outLPSonly2.txt",sep="¥t")
```