Spectral Graph Clustering using `gmmase`

JHU Team

2017-09-17

Contents

Given a (possibly directed) (possibly weighted) graph $G = (V, E)$, the `gmmase` package does

1. do a *pass-to-rank* for a weighted graph (`PTR`, no-op for an unweighted graph),
2. do a *graph spectral embedding* (`ASE` or `LSE`[1]) with a *diagonal augmentation*,
3. do a *dimension reduction* (`ZG`[2]) and merge left and right vectors (no-op for an undirected graph),
4. cluster vertices (`GMM`[3] or `Kmeans`).

1 Connectome Data

This vignette demo uses a connectome data[4] with 123 vertices and 2740 edges.

```
library(gmmase)
suppressPackageStartupMessages(library(igraph))

data("akira")
summary(akira)
```

```
# IGRAPH 1770d3c DNW- 123 2740 --
# + attr: name (v/c), weight (e/n)
```

```
knitr::kable(as.matrix(akira[])[1:10,1:10], digits=2)
```

	CN1_FB5	CN2	CN3	CN4_FB12	CN5_FB13	CN6_FB16	CN7	CN8_FB21	CN9	CN10
CN1_FB5	0.00	0.00	0.00	0.00	0.00	0.00	0.19	0	0	0
CN2	0.00	0.00	0.28	0.21	0.00	0.00	0.00	0	0	0
CN3	0.00	0.25	0.00	0.10	0.23	0.10	0.00	0	0	0
CN4_FB12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0	0	0
CN5_FB13	0.00	0.00	0.00	0.00	0.00	0.31	0.00	0	0	0
CN6_FB16	0.00	0.00	0.00	0.00	0.80	0.00	0.00	0	0	0

[1] D.L. Sussman, M. Tang, D.E. Fishkind, and C.E. Priebe, A consistent adjacency spectral embedding for stochastic blockmodel graphs, Journal of the American Statistical Association, Vol. 107, No. 499, pp. 1119-1128, 2012.

[2] M. Zhu, and A. Ghodsi, Automatic dimensionality selection from the scree plot via the use of profile likelihood. Computational Statistics and Data Analysis, Vol. 51, 918–930, 2006.

[3] MCLUST Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation, Technical Report no. 597, Department of Statistics, University of Washington, June 2012.

[4] K. Eichler, F. Li, A. L. Kumar, Y. Park, I. Andrade, C. Schneider-Mizell, T. Saumweber, A. Huser, D. Bonnery, B. Gerber, R. D. Fetter, J. W. Truman, C. E. Priebe, L. F. Abbott, A. Thum, M. Zlatic, and A. Cardona, "The complete connectome of a learning and memory center in an insect brain," Nature, no. 548, pp. 175-182, 2017.

	CN1_FB5	CN2	CN3	CN4_FB12	CN5_FB13	CN6_FB16	CN7	CN8_FB21	CN9	CN10
CN7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0	0	0
CN8_FB21	0.00	0.06	0.14	0.00	0.00	0.10	0.00	0	0	0
CN9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0	0	0
CN10	0.13	0.81	0.00	0.00	1.48	2.28	0.00	0	0	0

```
# take induced subgraph using only CN neurons
cn <- grep("CN", V(akira)$name)
g.cn <- induced_subgraph(akira, cn)
```

1.1 gmmase

```
out <- gmmase(g.cn, dmax = 20, embed = "ASE", clustering = "GMM", verbose=FALSE)
```

```
# 1. Finding an lcc...
# IGRAPH fd4bbd3 DNW- 43 515 --
# + attr: name (v/c), weight (e/n)
# 2. Passing-to-rank...
# IGRAPH fd4bbd3 DNW- 43 515 --
# + attr: name (v/c), weight (e/n)
# 3. Embedding the graph into dmax = 20...
# 4. Finding an elbow (dimension reduction)..., use dhat =  2
# 5. Clustering vertices..., Khat =  4
# ----------------------------------------------------
# Gaussian finite mixture model fitted by EM algorithm
# ----------------------------------------------------
#
# Mclust EEV (ellipsoidal, equal volume and shape) model with 4 components:
#
#  log.likelihood  n df       BIC       ICL
#        74.86953 43 47 -27.03735 -27.88816
#
# Clustering table:
#  1  2  3  4
# 11  8  9 15
```

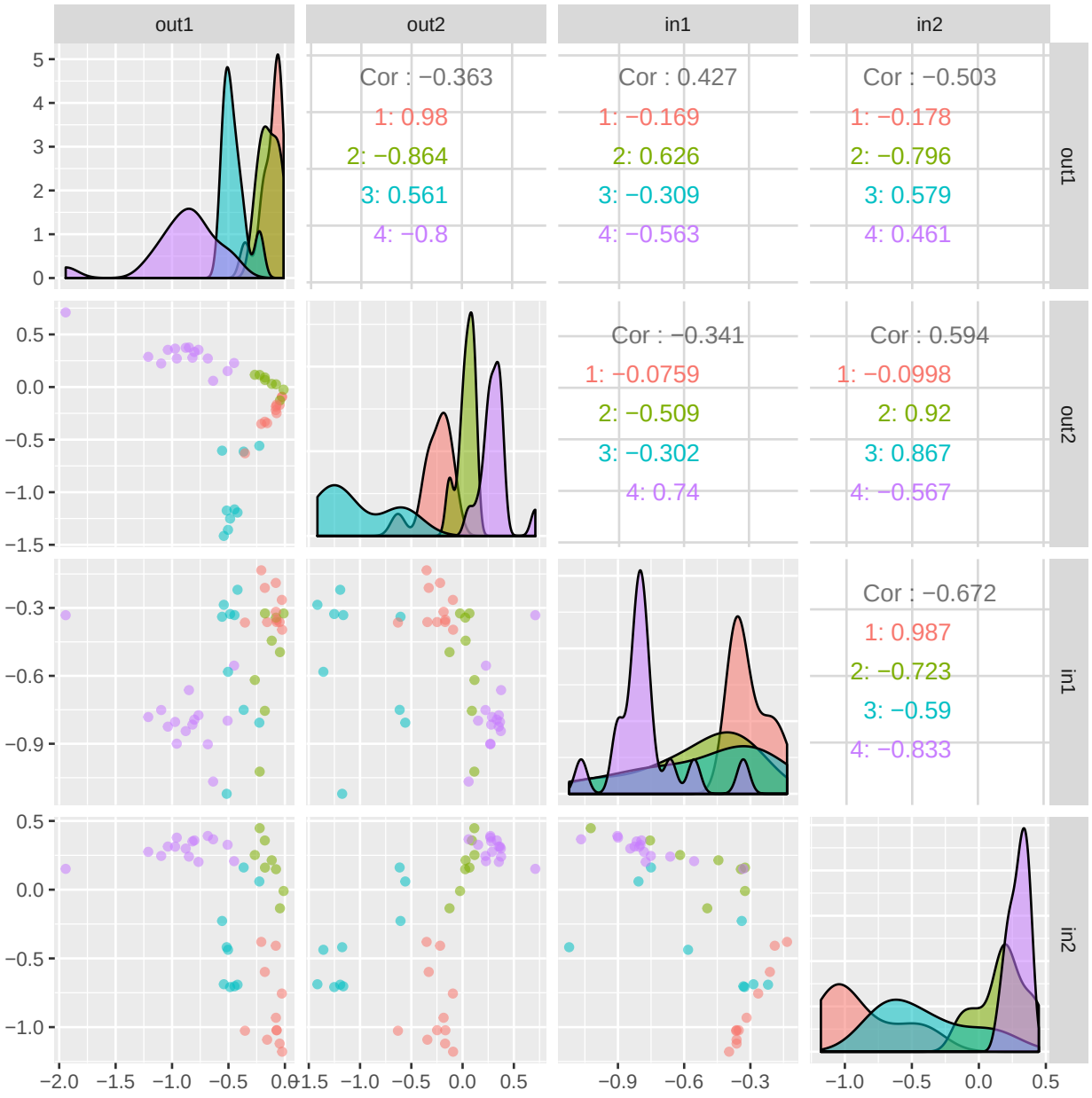
Now, we are plotting a paired scatter plot colored by the clustering labels.

```
g <- out$g
mc <- out$mc
Xhat <- mc$data
dhat <- ncol(Xhat)/2
Khat <- mc$G
colnames(Xhat) <- paste0(rep(c("out","in"),each=2), 1:2)
class <- mc$classification
df <- data.frame(Xhat, cluster=factor(class))

library(ggplot2)
library(GGally)
```

```
#
# Attaching package: 'GGally'
```

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

```
ggpairs(df, columns=1:ncol(Xhat), mapping=aes(color=cluster, alpha=0.5))
```



1.2 co-clustering

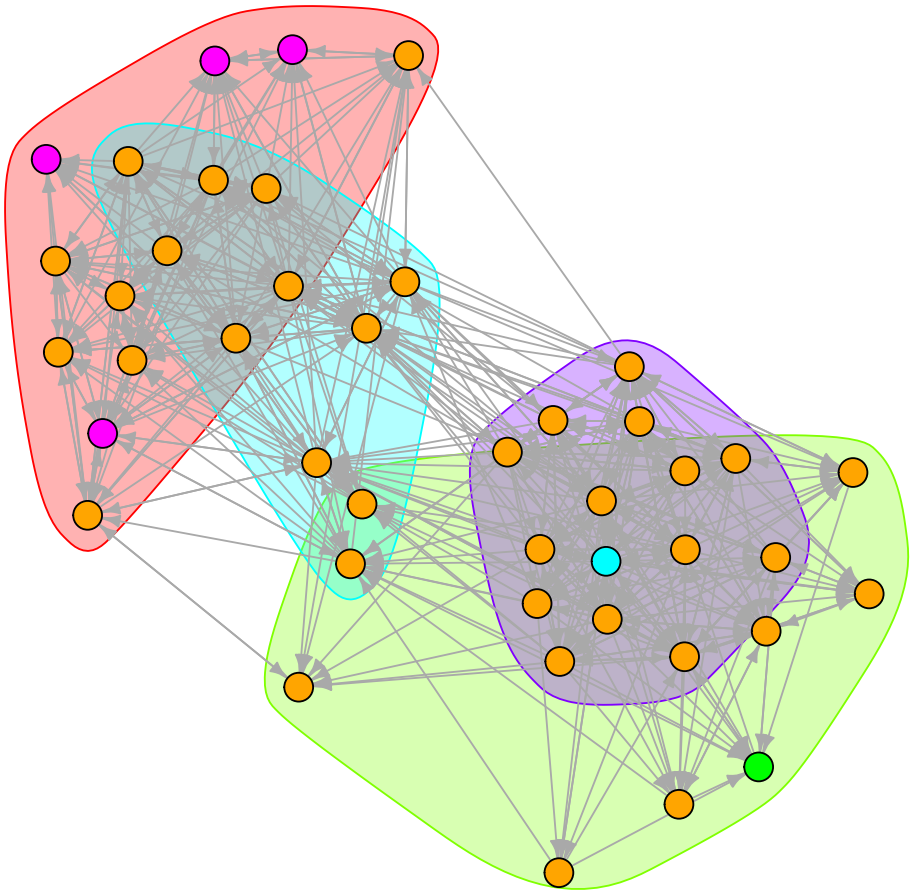
The data provider informed us from their beahvioral experiments that following neurons should be clustered onto their own groups.

- `CN4`, `CN12` `CN18`, and `CN41`,
- `CN19`,
- `CN40`.

```r
v1 <- c(4,12,18,41)
v2 <- 40
v3 <- 19
vcc<- c(v1,v2,v3)
df2 <- data.frame(name=V(g)$name, cluster=class)
df2[vcc,]
```

```
#           name cluster
# 4     CN4_FB12       1
# 12   CN12_FB25       1
# 18   CN18_FB30       1
# 41 MBONm1_CN41       1
# 40 MBONc1_CN40       4
# 19        CN19       2
```

```r
V(g)$color <- "orange"
V(g)$color[v1] <- "magenta"
V(g)$color[v2] <- "cyan"
V(g)$color[v3] <- "green"

plot(g, mark.groups=lapply(1:Khat, function(x) which(class==x)),
     edge.arrow.size=0.5, vertex.label=NA, #vertex.label.cex=0.8,
     vertex.size=7)#, vertex.color=rainbow(3, alpha=.5)[class])
```



This shows that we obtain the desired clustering!

2 MNIST Data[5]

There are 42000 training images of 10 digits (from 0 to 9), and we randomly select 1000 of them for our inference task.

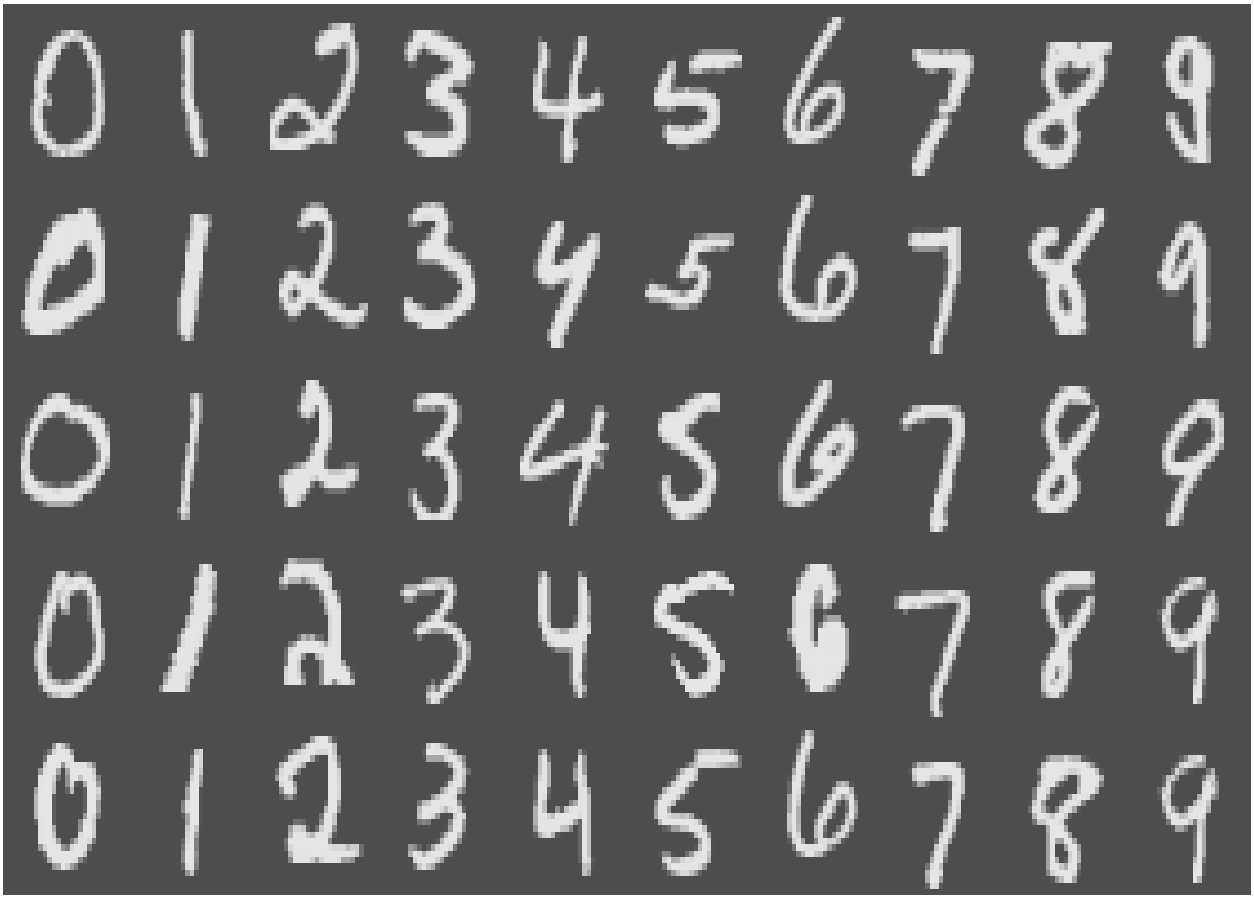
```
data(smnist)
slab <- smnist$slab
strain <- smnist$strain
(tab <- table(slab))
```

[5] http://yann.lecun.com/exdb/mnist/

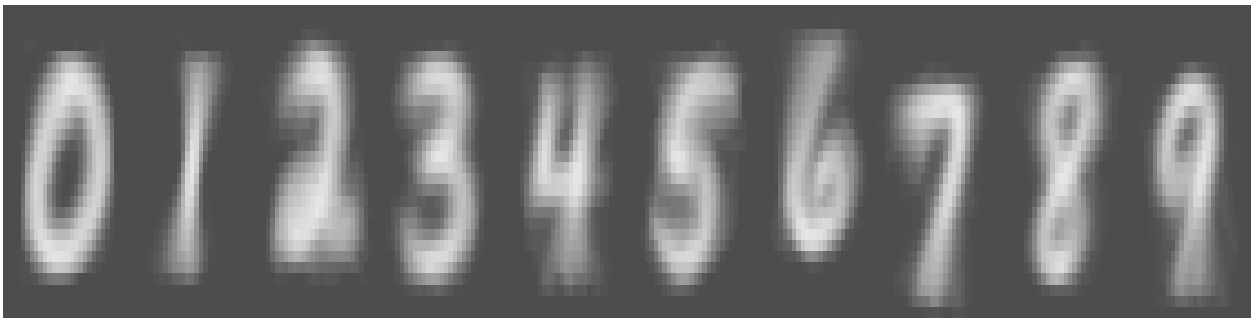
```
## slab
##   0   1   2   3   4   5   6   7   8   9
##  99 105 105 117  97  76  93 102  90 116
```

```
numTrain <- length(slab)

# Create a 28*28 matrix with pixel color values
res <- sqrt(ncol(strain))
glist <- lapply(1:numTrain, function(x) matrix(unlist(strain[x,]),byrow=T,ncol=res))
```

The following shows five random slections of each digit from the sampled data.



And, followings are averaged images of each digit from the sampled data.



2.1 Graph from Correlation Coefficients

We use Pearson correlation coefficient between two images as a similarity measure.

```
corrmat <- calcCorr(as.matrix(strain), useCorr=TRUE, recalc=TRUE); range(corrmat)
```

```
# [1] -0.1241095  0.9777664
```

```
#cor.eps <- median(as.vector(corrmat)); cat("threshold = ", cor.eps, "\n")
cor.eps <- quantile(as.vector(corrmat),0.75); cat("threshold = ", cor.eps, "\n")
```

```
# threshold =  0.380978
```

```
corrmat[corrmat<cor.eps] <- 0
g <- graph.adjacency(abs(corrmat), mode="undirected", weighted=TRUE); summary(g)
```

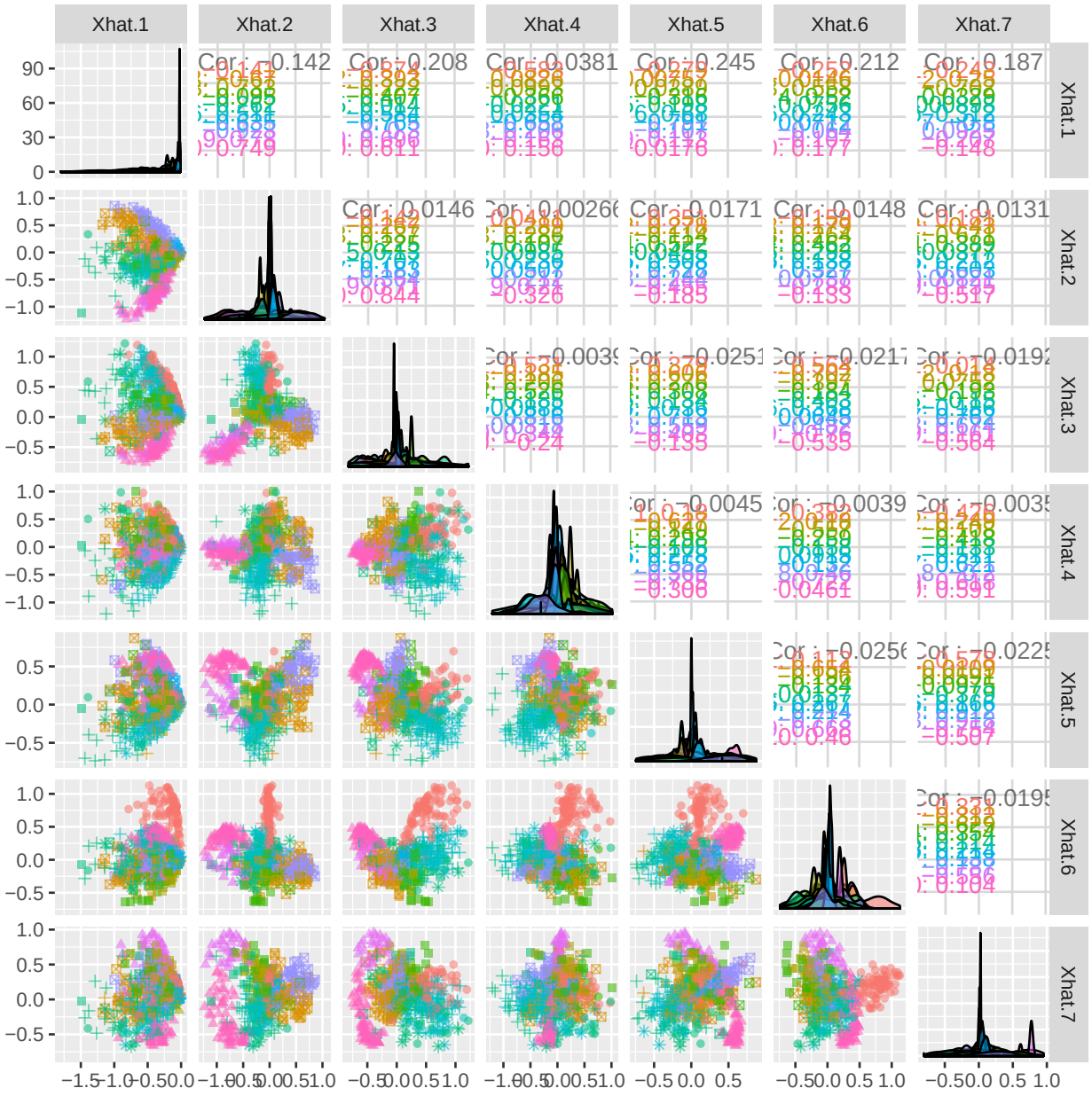
```
# IGRAPH 51fb0f2 U-W- 1000 125000 --
# + attr: weight (e/n)
```

```
#library(RColorBrewer)
#rf <- colorRampPalette(rev(brewer.pal(11,'Spectral')))
#r <- rf(10)
#V(g)$color <- r[slab+1]
#plot(g, layout=layout.spring,
#     edge.arrow.size=0.5, vertex.label=NA, #vertex.label.cex=0.8,
#     vertex.size=3)#, vertex.color=rainbow(3, alpha=.5)[class])


out1 <- gmmase(g, dmax = 100, embed = "ASE", Kmax = 10, clustering = "GMM", verbose=FALSE)
```

```
# 1. Finding an lcc...
# IGRAPH 387f265 U-W- 1000 125000 --
# + attr: weight (e/n)
# 2. Passing-to-rank...
# IGRAPH 387f265 U-W- 1000 125000 --
# + attr: weight (e/n)
# 3. Embedding the graph into dmax = 100...
# 4. Finding an elbow (dimension reduction)..., use dhat =  7
# 5. Clustering vertices..., Khat =  10
# ----------------------------------------------------
# Gaussian finite mixture model fitted by EM algorithm
# ----------------------------------------------------
#
# Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 10 components:
#
#  log.likelihood    n  df      BIC     ICL
#        1723.857 1000 359 967.8304 881.793
#
# Clustering table:
#   1   2   3   4   5   6   7   8   9  10
#  92 160  73  89 179 132  62 107  37  69
```

```
#out1 <- gmmase(g, dmax = 100, embed = "LSE", Kmax = 10, clustering = "GMM", verbose=FALSE)
#out1 <- gmmase(g, dmax = 100, embed = "ASE", Kmax = 10, clustering = "Kmeans", verbose=FALSE)

Xhat1 <- out1$mc$data
Yhat <- out1$Y
```

```
df2 <- data.frame(Xhat=Xhat1, lab=as.factor(slab), cluster=as.factor(Yhat))
ggpairs(df2, columns=1:(ncol(df2)-2), mapping=aes(color=cluster, shape=lab, alpha=0.5))
```



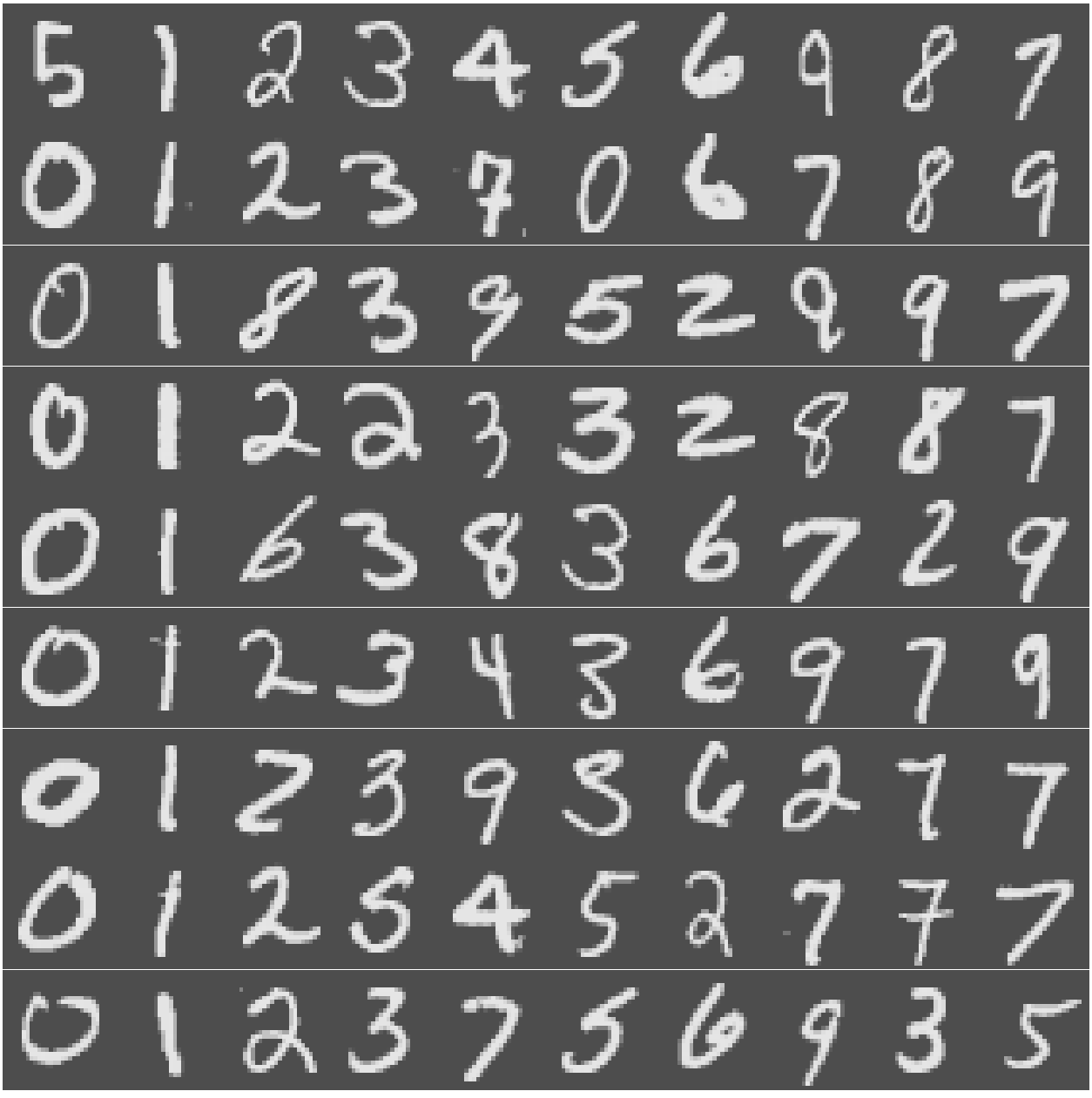
Now we plot nine random images of the cluster that contains each digist the most.

	1	2	3	4	5	6	7	8	9	10
0	82	0	0	1	8	6	2	0	0	0
1	0	0	4	0	1	0	0	0	33	67
2	1	6	45	23	11	3	15	1	0	0
3	1	5	1	1	28	76	4	1	0	0
4	0	45	1	4	3	0	7	37	0	0
5	5	2	12	2	12	36	3	4	0	0
6	2	0	3	55	15	0	17	1	0	0
7	0	50	3	0	13	0	7	25	4	0

	1	2	3	4	5	6	7	8	9	10
8	0	8	4	0	66	11	0	0	0	1
9	1	44	0	3	22	0	7	38	0	1

```
# ARI for Khat = 10  is  0.31
```

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



2.2 Graph from k-nearest neighbors

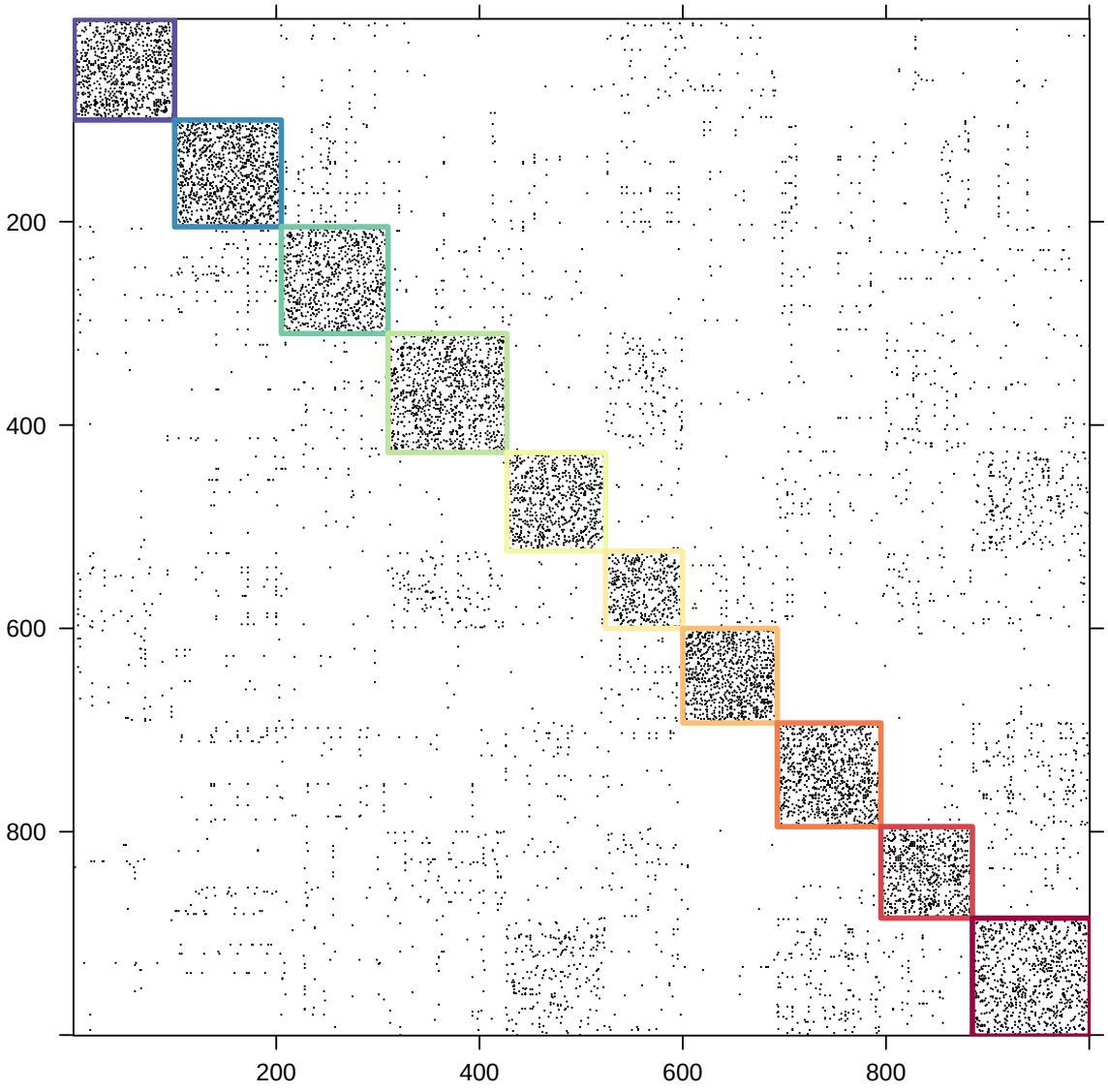
From a given point i, it is connected to the k other points that are closest to it in the Euclidean space.

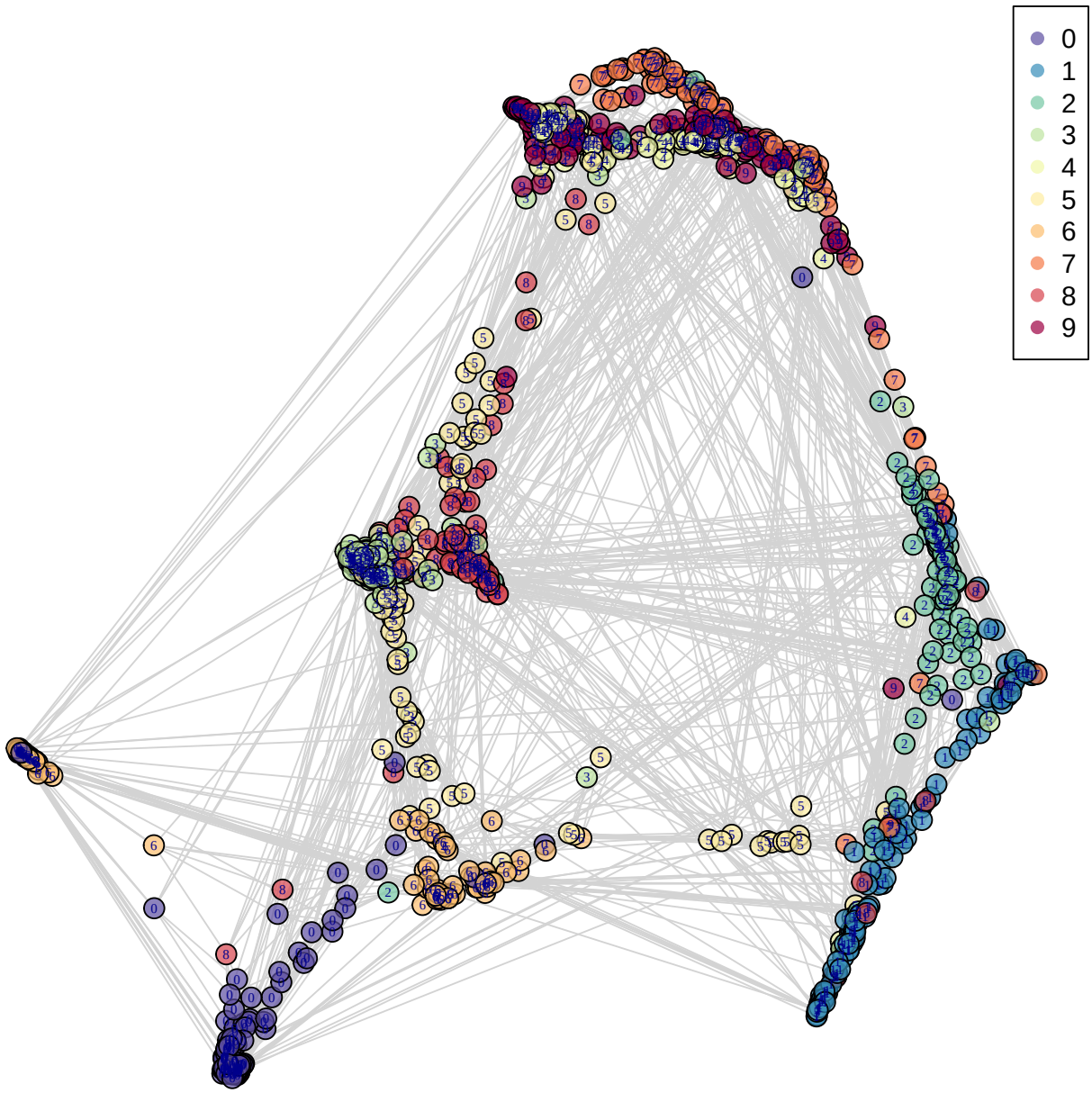
```
D <- as.matrix(dist(strain))
kvec <- 3:9
ariv <- dhatv <- khatv <- rep(0,length(kvec))
for (k in 1:length(kvec)) {
    A <- matrix(0,numTrain,numTrain)
    for (i in 1:numTrain) {
        nn <- order(D[i,])[1:kvec[k]]
        A[i,nn] <- A[nn,i] <- 1
    }
    diag(A) <- 0

    g.knn <- graph.adjacency(A, mode="undirected"); summary(g.knn)
    out2 <- gmmase(g.knn, dmax = 20, embed = "ASE", Kmax = 10, clustering = "GMM",
                   verbose=FALSE, doplot=FALSE)
    dhatv[k] <- out2$elb
    Yhat2 <- out2$Y
    khatv[k] <- max(Yhat2)
    ariv[k] <- adjustedRandIndex(slab, Yhat2)
    cat("k = ", kvec[k], ", dhat = ", dhatv[k], ", Khat = ", khatv[k], ", ari = ", ariv[k], "\n")
}
```

k	dhat	khat	ari
3	17	10	0.24
4	8	10	0.27
5	7	10	0.27
6	10	10	0.36
7	9	10	0.40
8	9	10	0.38
9	10	10	0.42

Let's see the example run of $argmax_k ARI = $ 9.

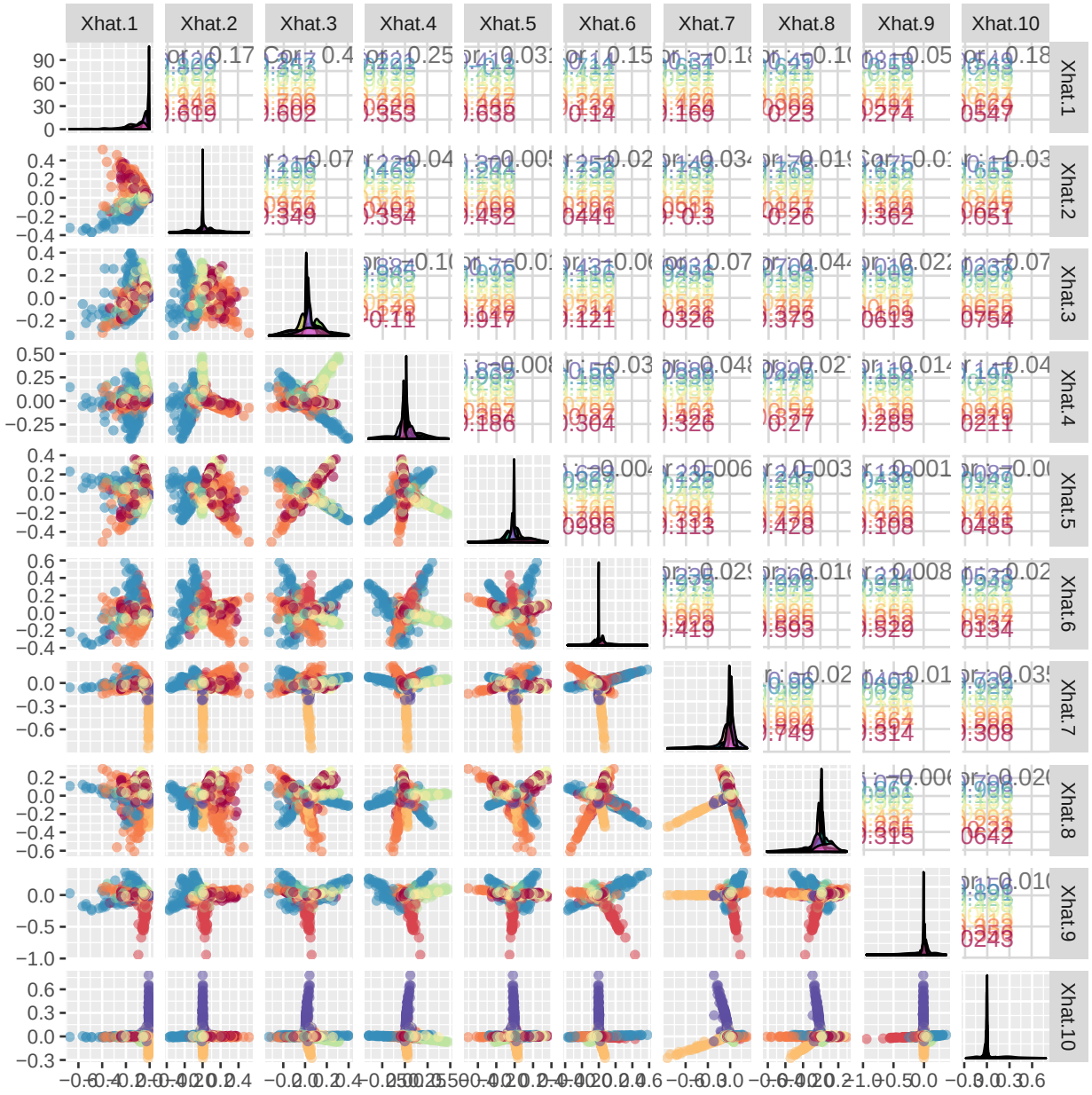
```
# IGRAPH 47a08fb U--- 1000 5797 --
```


200
400
600
800
200
400
600
800




```
# 1. Finding an lcc...
# IGRAPH 46575c3 U--- 1000 5797 --
# + attr: color (v/c)
# 3. Embedding the graph into dmax = 20...
# 4. Finding an elbow (dimension reduction)..., use dhat =  10
# 5. Clustering vertices..., Khat =  10
# ----------------------------------------------------
# Gaussian finite mixture model fitted by EM algorithm
# ----------------------------------------------------
#
# Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 10 components:
#
#  log.likelihood    n  df      BIC      ICL
#        27414.33 1000 659 50276.46 50252.37
```

```
#
# Clustering table:
#   1   2   3   4   5   6   7   8   9  10
# 164 100 103  75  84  81  88  98  83 124
```



	1	2	3	4	5	6	7	8	9	10
0	0	0	6	74	0	0	19	0	0	0
1	105	0	0	0	0	0	0	0	0	0
2	24	0	5	0	5	20	4	1	0	46
3	3	0	0	0	11	5	21	75	0	2
4	3	9	1	0	1	32	0	0	39	12
5	10	1	2	0	1	8	31	19	0	4
6	4	0	87	0	0	0	2	0	0	0
7	7	53	0	0	0	1	0	0	2	39

	1	2	3	4	5	6	7	8	9	10
8	5	2	0	1	66	3	9	3	0	1
9	3	35	2	0	0	12	2	0	42	20

```
# ARI for Khat = 10  is  0.42
```

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