Lecture 20

Cluster analysis – R code

MCB 416A/516A
Statistical Bioinformatics and Genomic Analysis

Prof. Lingling An
Univ of Arizona

Last time:

- Types of clustering in gene expression: objects to be clustered
- Application of clustering gene expression

Outline

- R code for cluster analysis
- An example of why we sometime need do data scaling before perform clustering

New libraries …

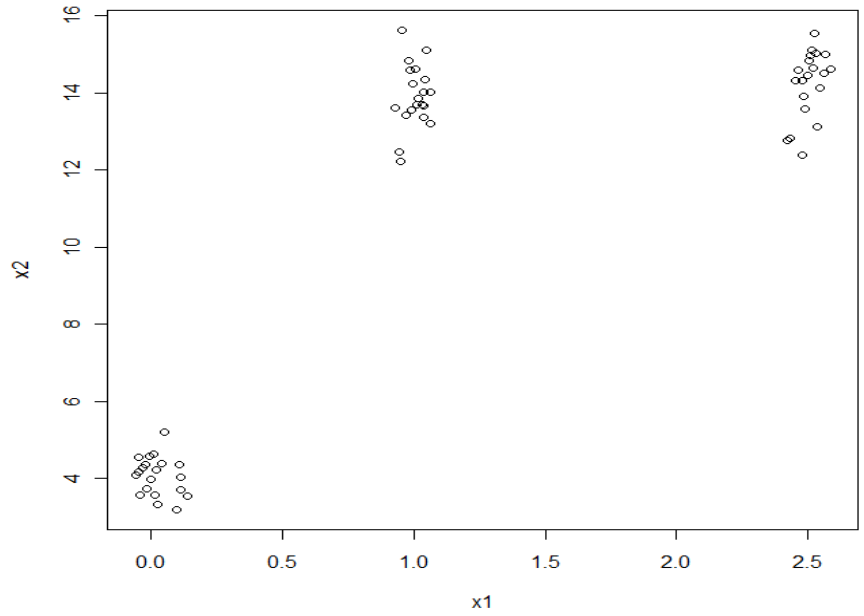
```
library(SAGx) ##  - for gap statistic

library(cluster) ##- for hierarchical
  cluster, k-mean cluster & silhouette width
library(Hmisc) ##- for error bar plot
```

Install “SAGx”:

```
source("http://bioconductor.org/biocLite.R")
biocLite("SAGx")
```

Why we need scaling data for cluster analysis?

Simple example:



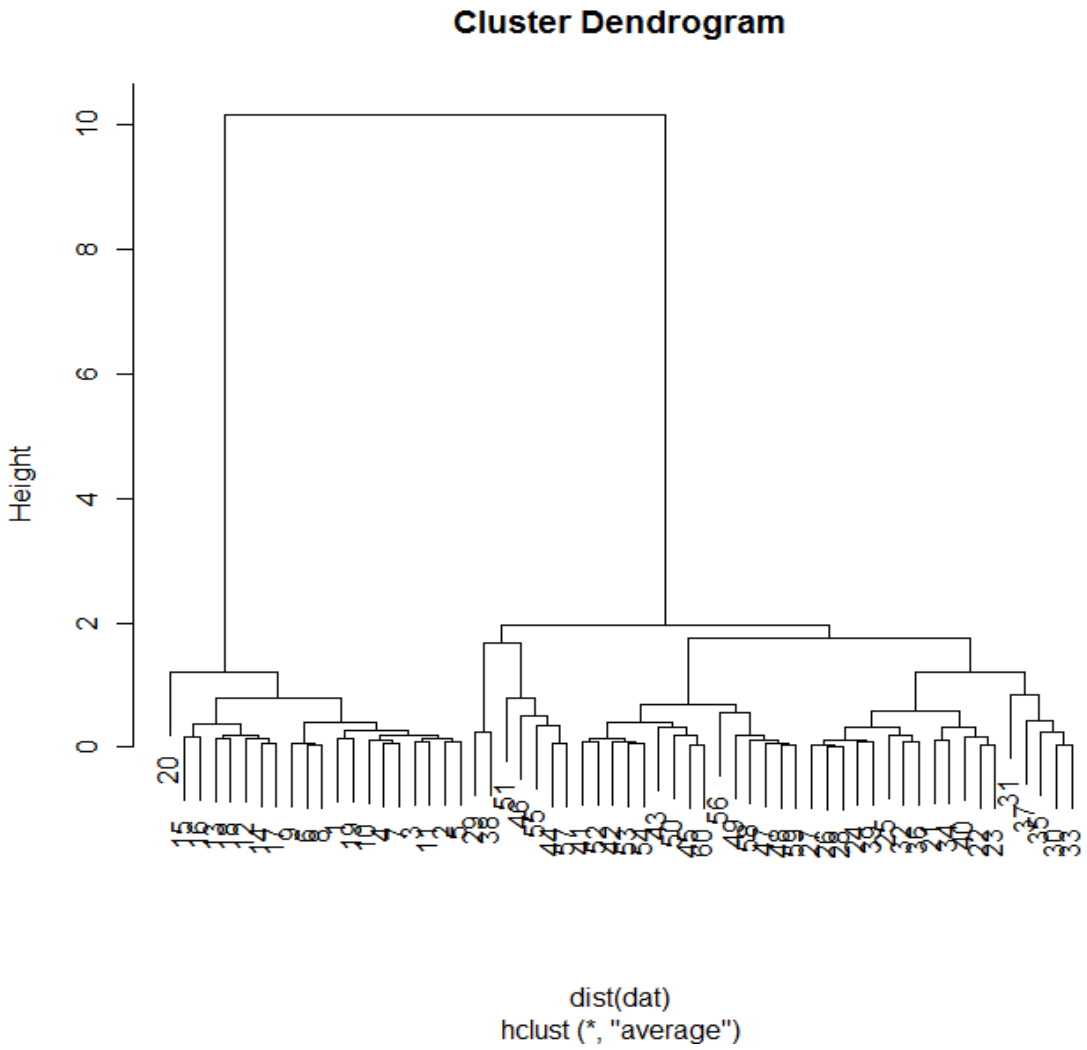
- Think of x1 as being: expression on array 1
- and x2 as being: expression on array 2
- Here, there are 60 “genes”

R code: generate data in previous plot

```
set.seed(7)          ## for random number later
x1 <- c(rnorm(20, sd=.05), rnorm(20, mean=1, sd=.05),
   rnorm(20, mean=2.5, sd=.05))
## x1 dimension with 3 clusters

x2 <- 4+c(rnorm(20, sd=0.5),
   rnorm(20, mean=10, sd=1.0), rnorm(20, mean=10, sd=1))
## x2 dimension with 3 clusters

plot(x1, x2)  ## scatter plot of the data
```

Just hierarchical clustering?

```
dat=cbind(x1,x2) ##  combine two vectors into one
   data matrix

hc0=hclust(dist(dat), method="ave")
   ## hierarchical clustering (hc) with average
   linkage

plot(hc0) ## plot the dendrogram of the hc result
```



2 clusters?

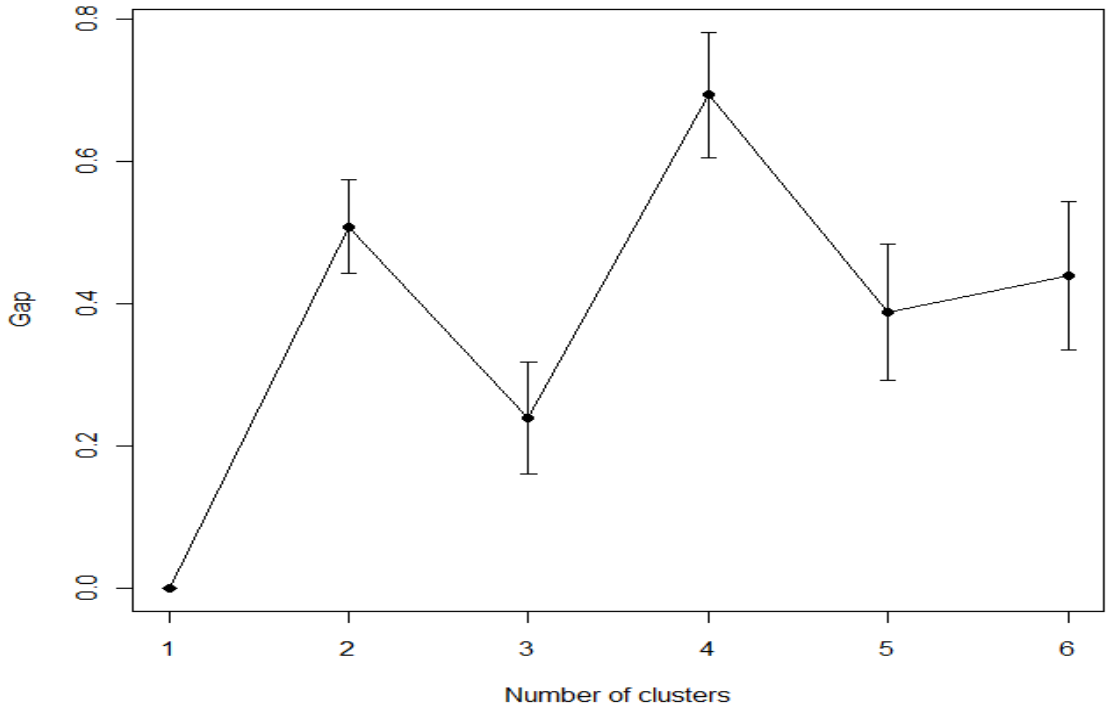
Check ***gap statistic*** and ***silhouette width***

Using gap statistic to determine k in HC - 1

```
k=6   ## check all possible numbers of clusters, 2~6
Gap=rep(0,k)        ## initialize the Gap statistics
se=rep(0,k) ## initialize the standard error


for (i in 2:k){
   mem=cutree(hc0, i)    ## get the cluster membership
   by using “cuttree” on the object of hier. cluster.
   result=gap(dat, class=mem)   ## get the gap
   statistics
   Gap[i]=result[1]      ## extract the gap stat values
   se[i]=result[2]       ## and the s.e. values
}
```

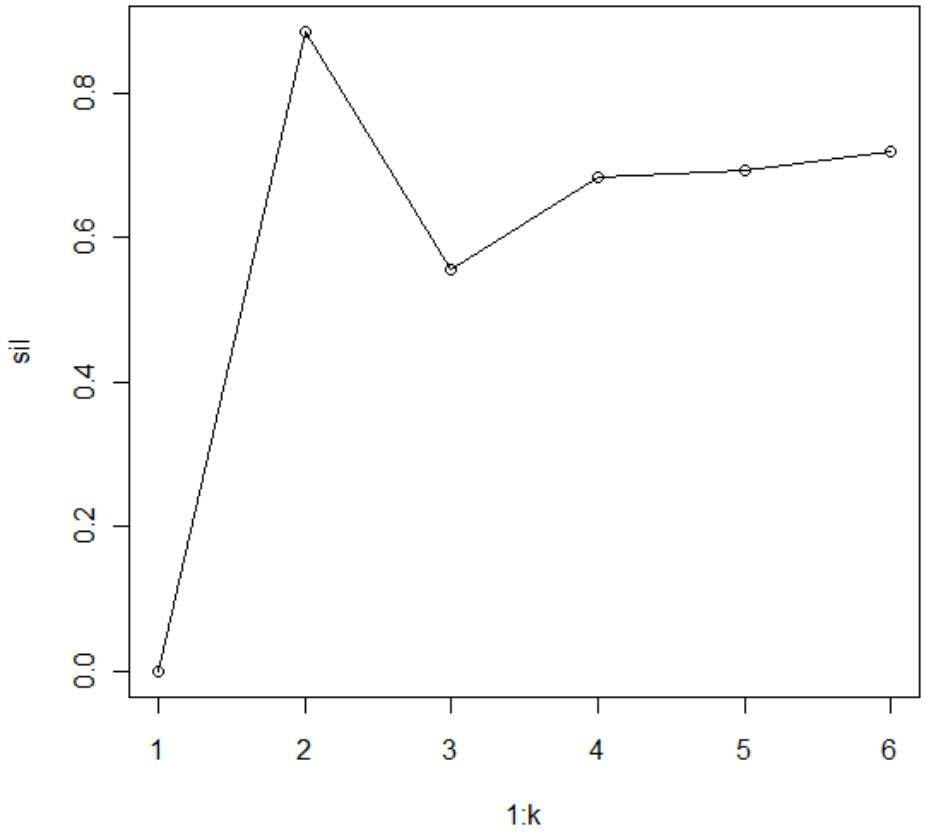
Using gap statistic to determine k in HC -2

```
errbar(1:k, Gap, Gap-se, Gap+se, xlab="Number of
   clusters")          ## error bar plot
lines(1:k, Gap) ## connect them
```



Using silhouette width to determine k in HC

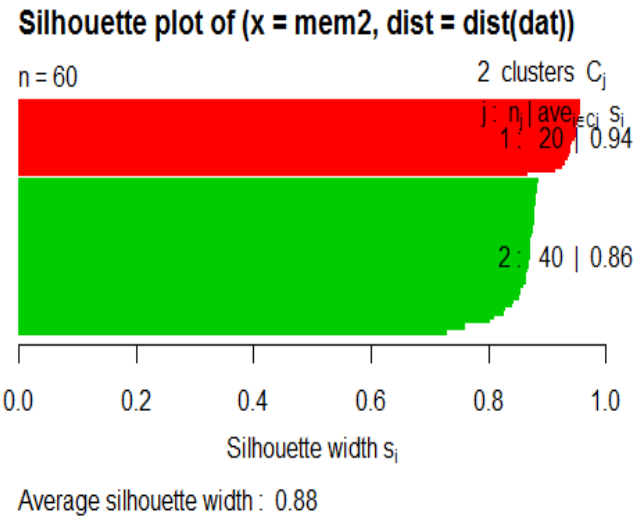
```
k=6
sil=rep(0,k)
for (i in 2:k){
   mem=cutree(hc0, i)
 aa=silhouette(mem,
   dist(dat))
   sil[i]=mean(aa[,3])
   }
plot(1:k, sil)
lines(1:k, sil)
```



silhouette width plot (to double check)

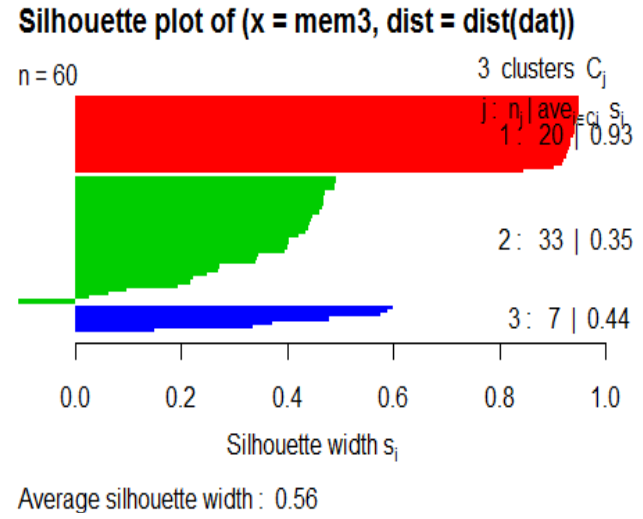
- ## K=2

```
mem2=cutree(hc0, 2)
aa=silhouette(mem2,
   dist(dat))
plot(aa, col=mem2+1)
```



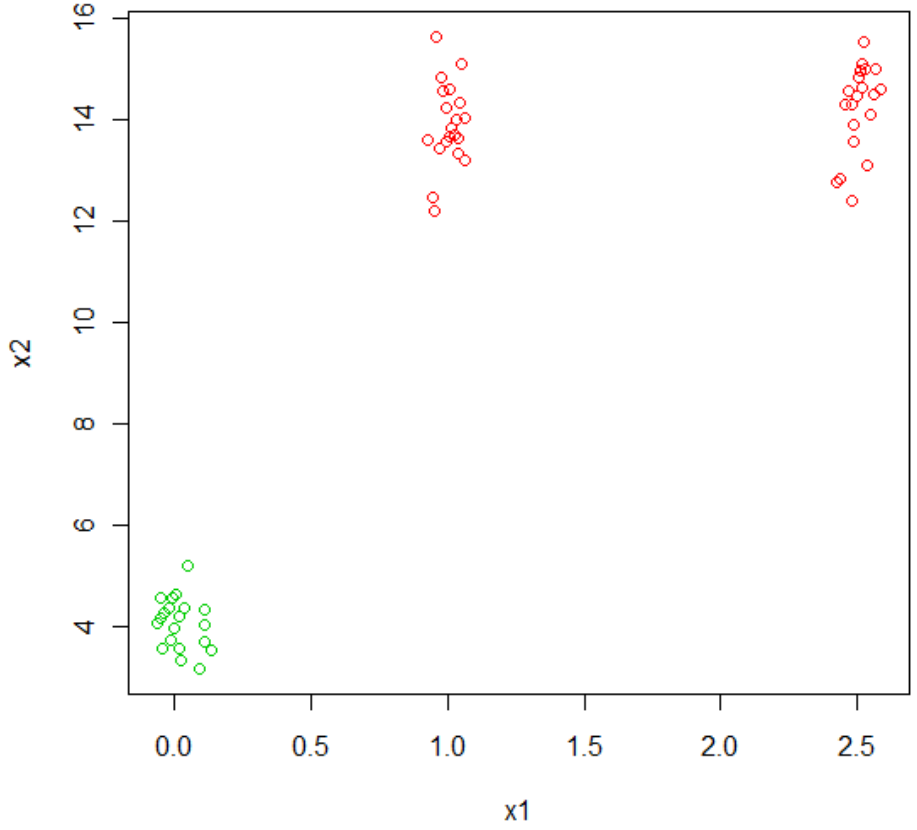
- ## K=3

```
mem3=cutree(hc0, 3)
aa=silhouette(mem3,
   dist(dat))
plot(aa, col=mem3+1)
```



If use k-mean cluster for this simple example…

```
## take a look at k=2

km=kmeans(dat, centers=2)
mem=km$cluster
plot(x1,x2, col=mem+1)
```

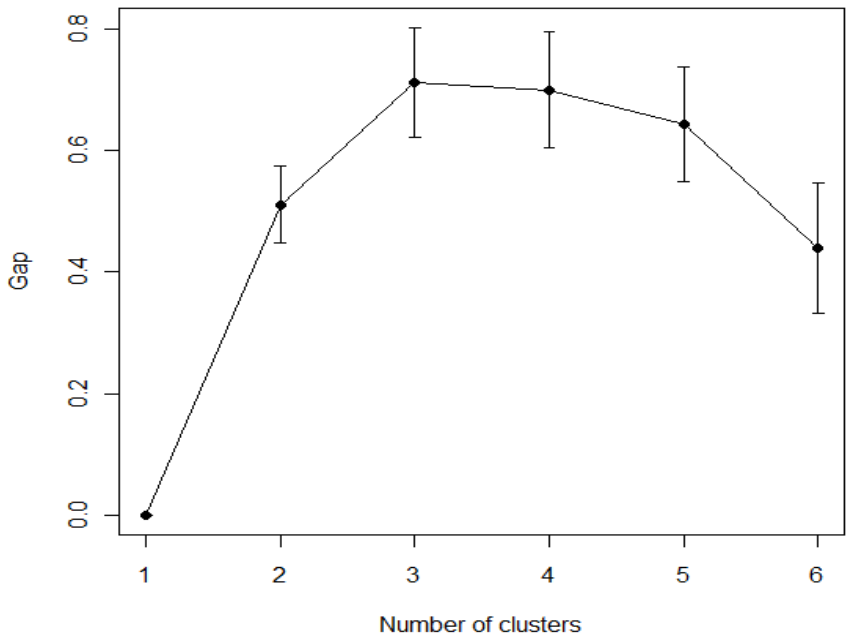


Use gap statistic to determine k in K-means -1

```
k=6   ## check all possible numbers of clusters, 2~6
Gap=rep(0,k)       ## initialize the Gap statistics
se=rep(0,k) ## initialize the standard error

for (i in 2:k){
   km=kmeans(dat, centers=i)
   mem=km$cluster
   result=gap(dat, class=mem)   ## get the gap
   statistics
   Gap[i]=result[1]       ## extract the gap stat values
   se[i]=result[2]        ## and the s.e. values
}
```

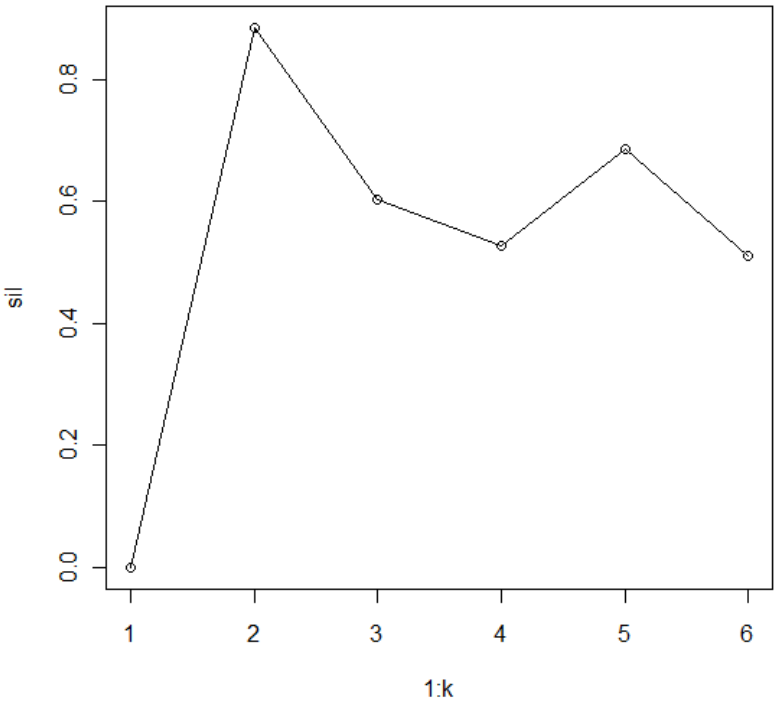
Use gap statistic to determine k in K-means -2

```
errbar(1:k, Gap, Gap-se, Gap+se, xlab="Number of
  clusters")          ## error bar plot
lines(1:k, Gap) ## connect them
```



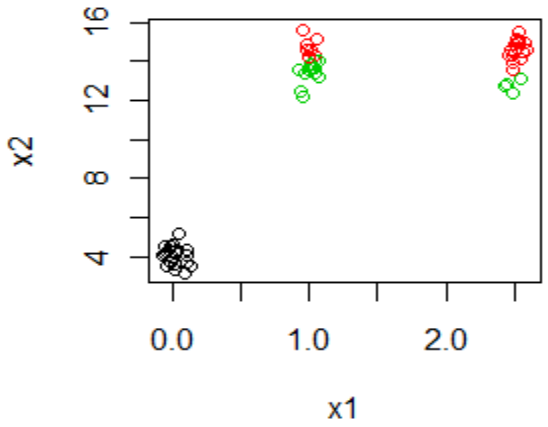
Use silhouette width to determine k in K-means

```
k=6
sil=rep(0,k)
for (i in 2:k){
   km=kmeans(dat, centers=i)
   mem=km$cluster
  aa=silhouette(mem,
   dist(dat))
   sil[i]=mean(aa[,3])
   }
plot(1:k, sil)
lines(1:k, sil)
```

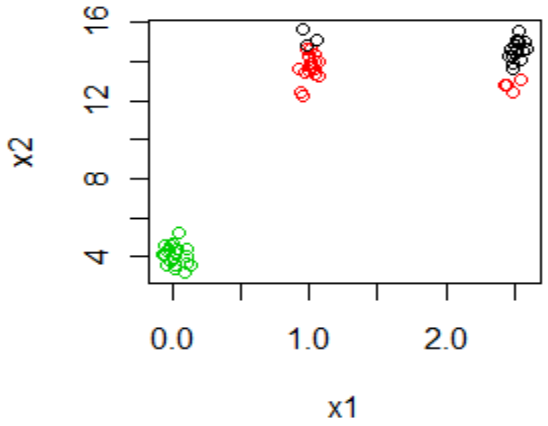


How about just let k=3 for both HC and K-means, and check …

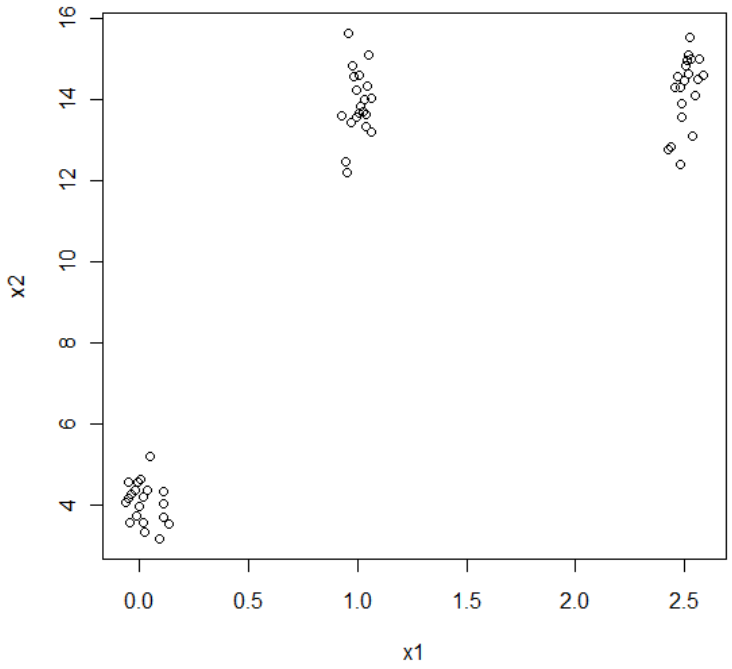
```
hc0=hclust(dist(dat))
mem=cutree(hc0, 3)
plot(x1,x2, col=mem)
```




```
km=kmeans(dat,3)
mem=km$cluster
plot(x1,x2, col=mem)
```

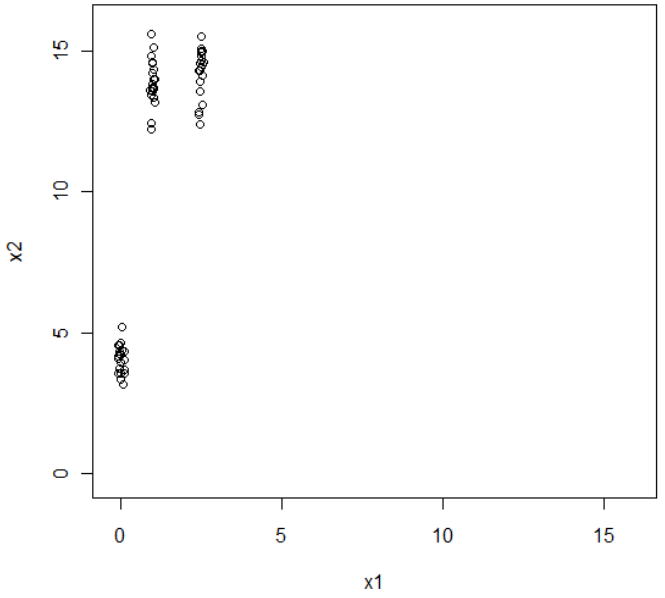


What? revisit the data …



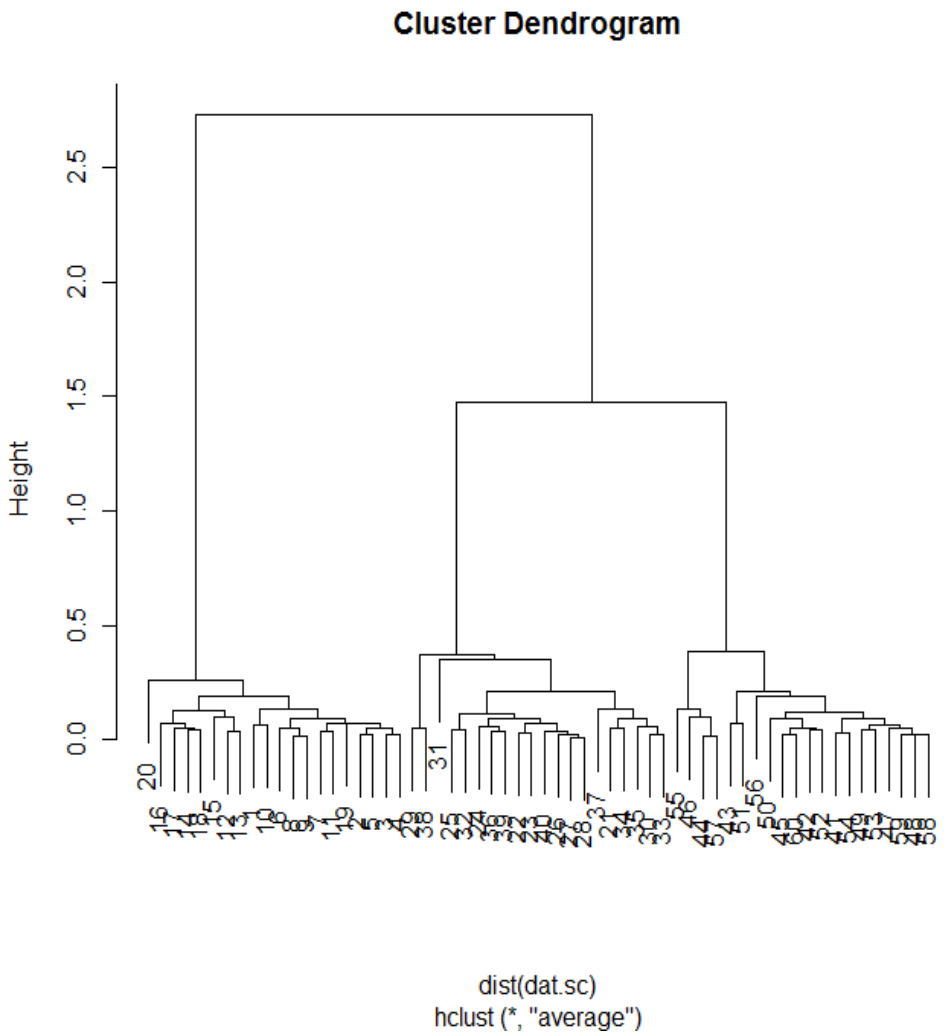
plot(x1,x2)

plot(x1,x2, xlim=c(-0.2, 16), ylim=c(-0.2, 16))



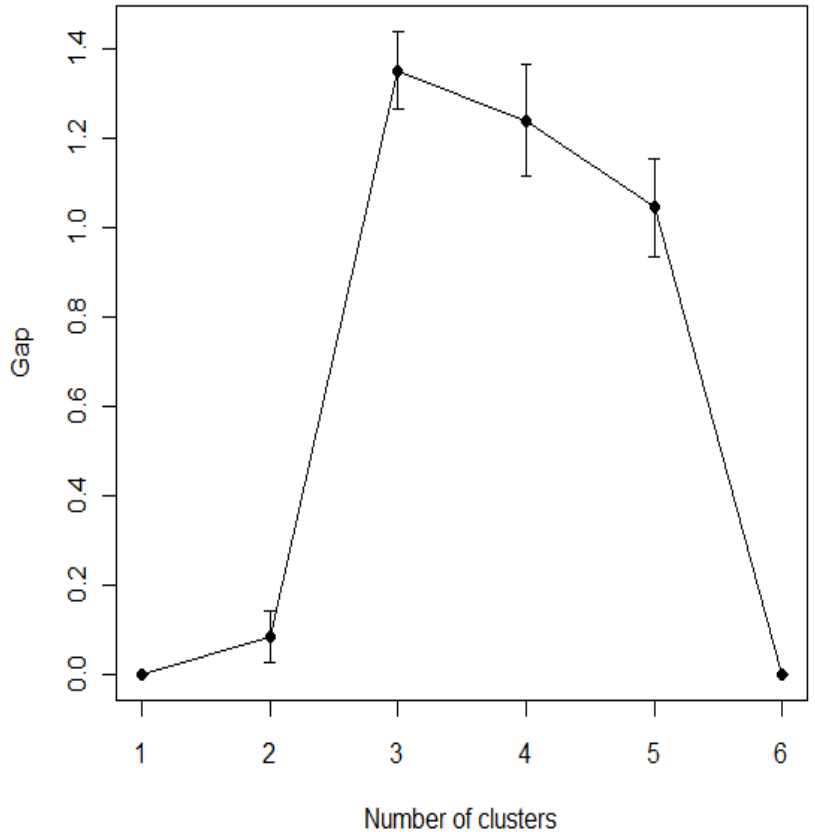
Need Scaling…

```
x1.sc <- x1/sd(x1)
x2.sc <- x2/sd(x2)
dat.sc <-
   cbind(x1.sc, x2.sc)


hc1=hclust(dist(dat.sc)
   , "ave")
plot(hc1)
```



For the scaled data, perform HC and gap

```
K=6; Gap=rep(0,k); se=rep(0,k)

for (i in 2:k){
   mem=cutree(hc1, i)
   result=gap(dat.sc,class=mem)
   Gap[i]=result[1]
   se[i]=result[2]
}
errbar(1:k, Gap, Gap-se, Gap
   +se, xlab="Number of
   clusters")
lines(1:k, Gap)
```

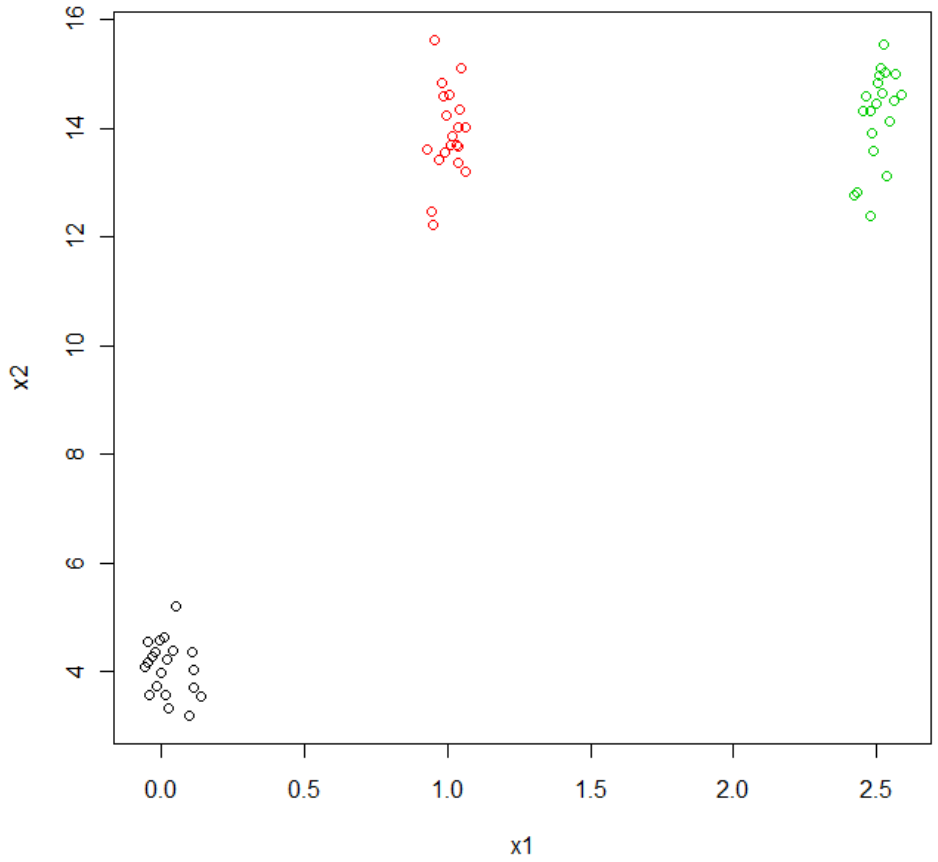


Error in gap(dat.sc, class = mem) :
Singleton clusters not allowed

=> try smaller k

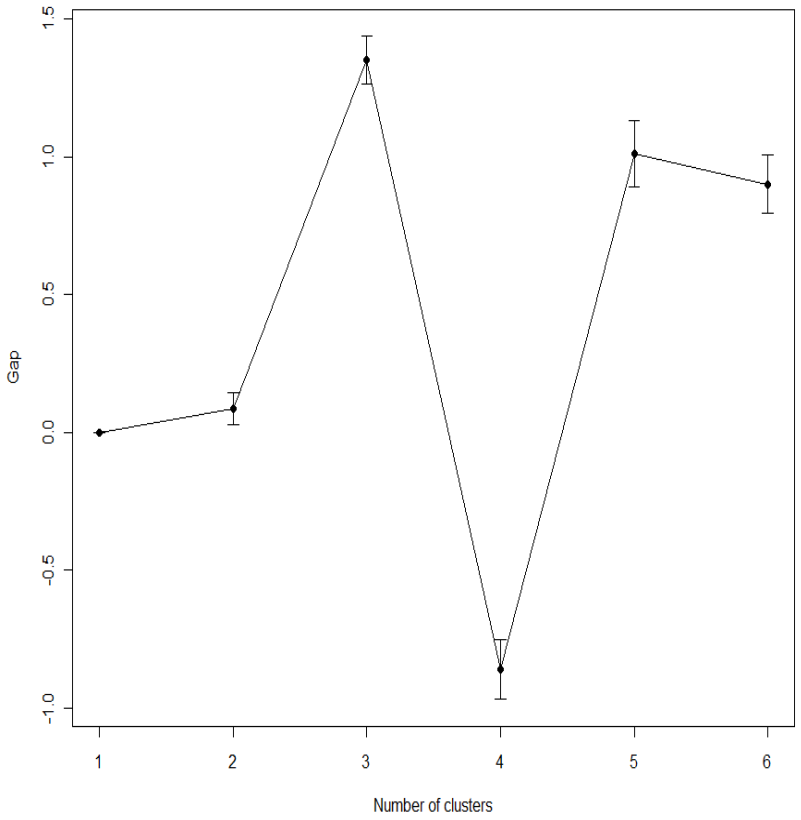
Scatter Plot of the original data using new cluster membership

```
mem3=cutree(hc1, 3)
plot(x1,x2, col=mem3)
```



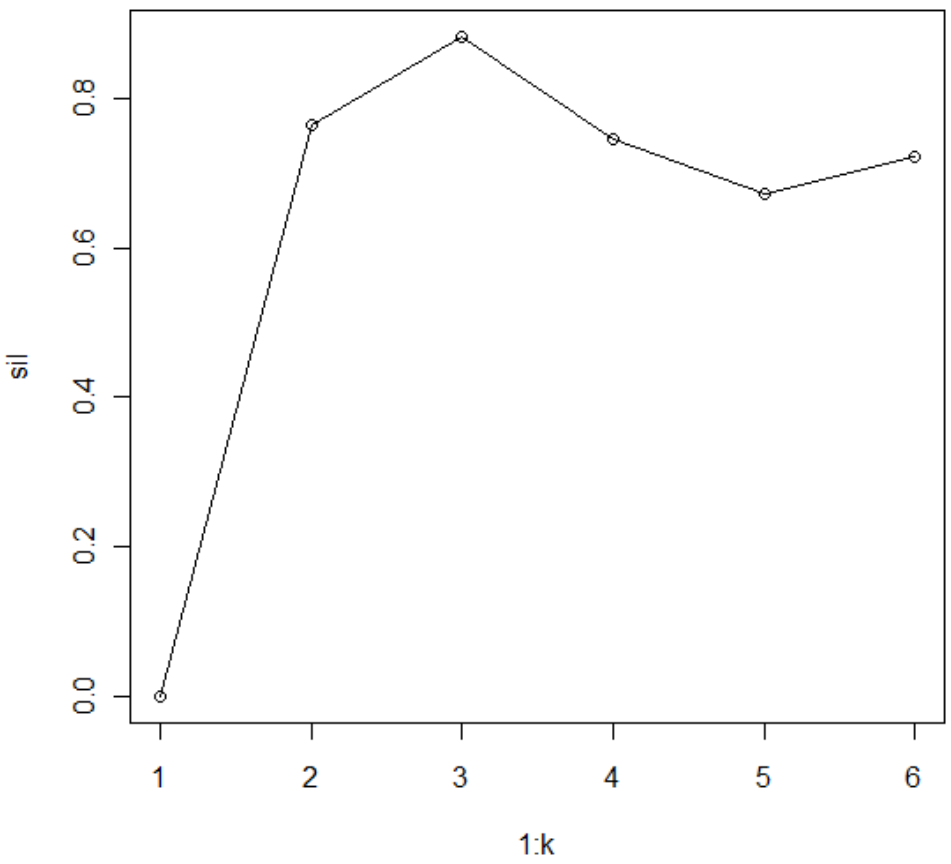
For the scaled data, perform Kmeans and gap …

```
k=6; Gap=rep(0,k); se=rep(0,k)

for (i in 2:k){
    km=kmeans(dat.sc, centers=i)
    mem=km$cluster
    result=gap(dat.sc, class=mem)
    Gap[i]=result[1]
    se[i]=result[2]
}
errbar(1:k, Gap, Gap-se, Gap
    +se, xlab="Number of
    clusters")
lines(1:k, Gap)
```



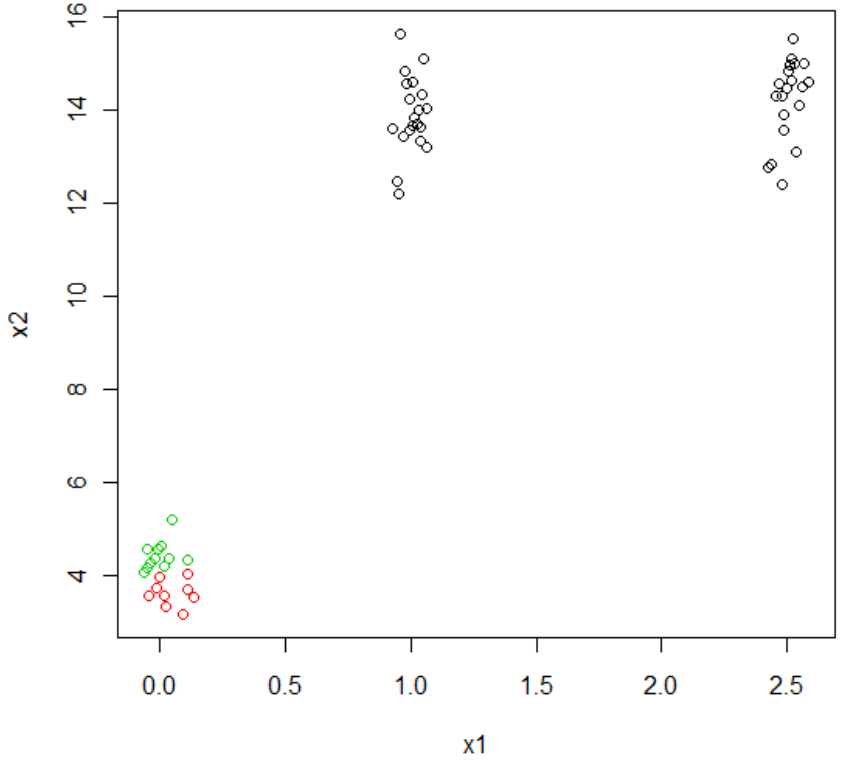
For the scaled data, perform Kmeans and silhouette width

```
k=6
sil=rep(0,k)
for (i in 2:k){
    km=kmeans(dat.sc, centers
    mem=km$cluster
 aa=silhouette(mem,
    dist(dat.sc))
    sil[i]=mean(aa[,3])
    }
plot(1:k, sil)
lines(1:k, sil)
```



Curious about kmeans result …

```
km=kmeans(dat.sc,
   centers=3)
mem3=km$cluster

plot(x1, x2,
   col=mem3)
```



What????

K-means is sensitive to seeds initialization

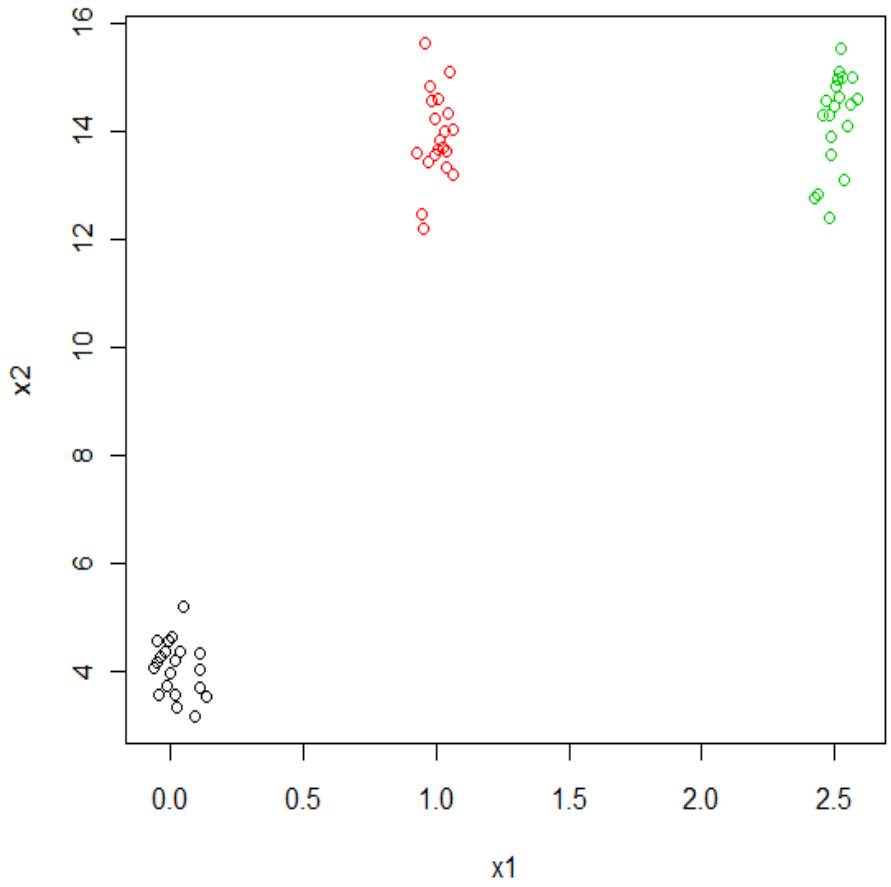
Recommend: Hybrid … (hierarchical+kmeans)

Use the seeds info from HC result

```
hc=hclust(dist(dat.sc),
   method="ave")
   mem=cutree(hc, 3)
c1=tapply(x1.sc, mem, mean)
c2=tapply(x2.sc, mem, mean)
```

$ then do kmeans clustering:

```
km=kmeans(dat.sc,
   centers=cbind(c1, c2))
plot(x1, x2, col=km
   $cluster)
```



Another way for visualizing hierarchical clustering result on high dimensional data

Heat map:
using "liver.brain" dataset as an example
http://cals.arizona.edu/~anling/MCB516/data_lecture13/

Packages needed

```
library(affy)
library(limma) ##- linear model for DE
library(Heatplus)## - for heat map view
library(gplots) ## - for color change in
  heatmap plot


## you may need to install “Heatplus” first:
source("http://bioconductor.org/biocLite.R")
biocLite("Heatplus")
```

Read data

Download the 8 .CEL files and dataset from:

http://cals.arizona.edu/~anling/MCB516/data_lecture13/

Save it to a folder;

Change the working directory to the folder where your data are saved

Read in data:

```
> raw = ReadAffy()
```

Preprocess data:

```
> eset=rma(raw)
```

Construct a design matrix

- construct a design matrix for this experiment

```
> f = factor(c(1,1,2,2,3,3,4,4), labels=c("brain",
"f.brain", "f.liver", "liver" ))
```

Or use:

```
f = factor(c("brain", "brain", "f.brain", "f.brain",
"f.liver", "f.liver", "liver", "liver"))
```

```
>design = model.matrix(~ 0 + f)
> design
>colnames(design) = c("brain", "f.brain", "f.liver",
"liver")
```

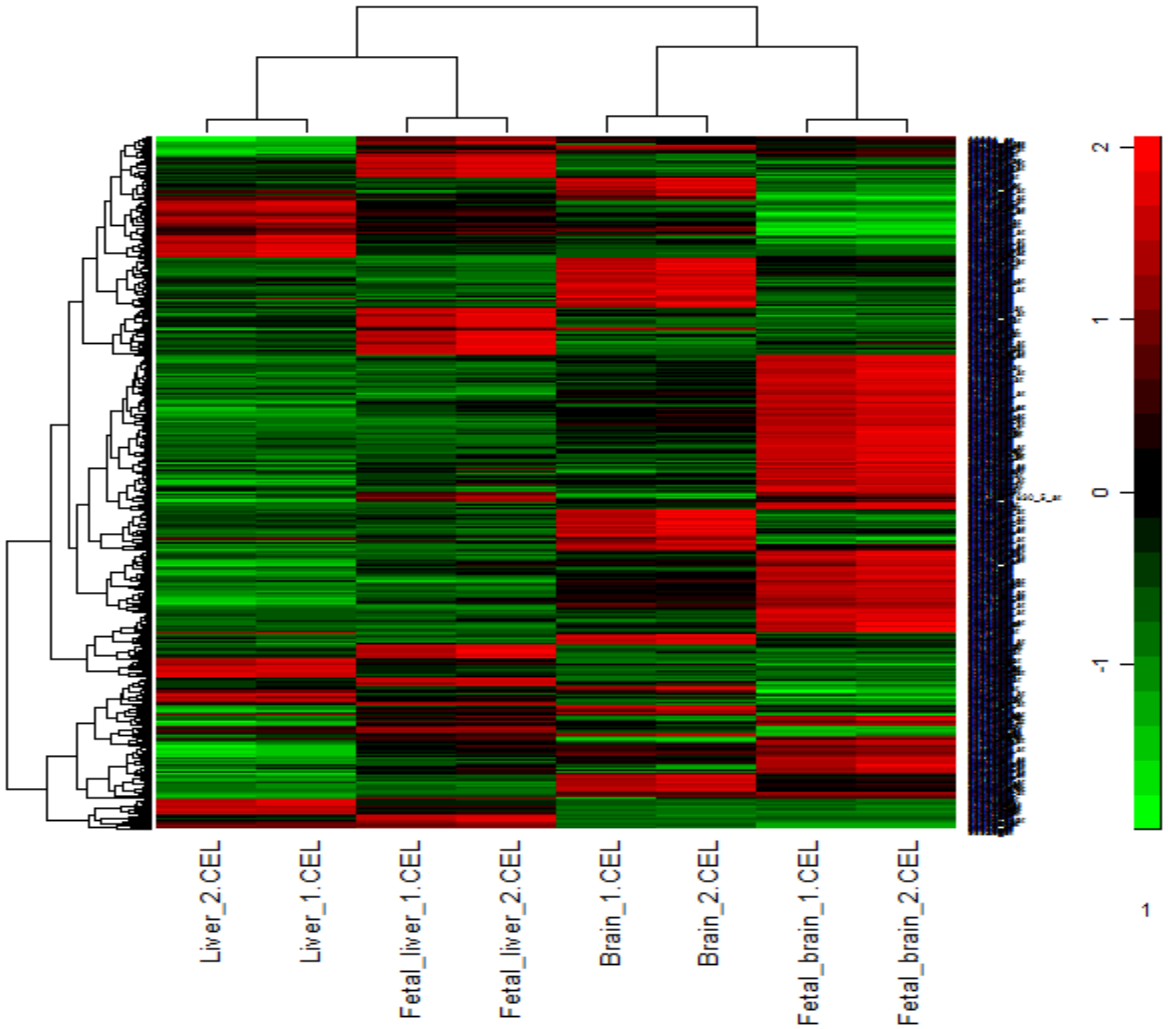
Linear fitting

```
fit = lmFit(eset, design)
contrast.matrix = makeContrasts(f.brain-brain,
   f.liver -liver, levels = design)
fit2 = contrasts.fit(fit, contrast.matrix)
fit2 = eBayes(fit2)
```

```
## select genes with adjusted p-value < 0.0001
p.bh = p.adjust(fit2$F.p.value, method = "BH")

select = (p.bh < 0.0001)

table(select) ## how many genes are selected for
  further analysis
```

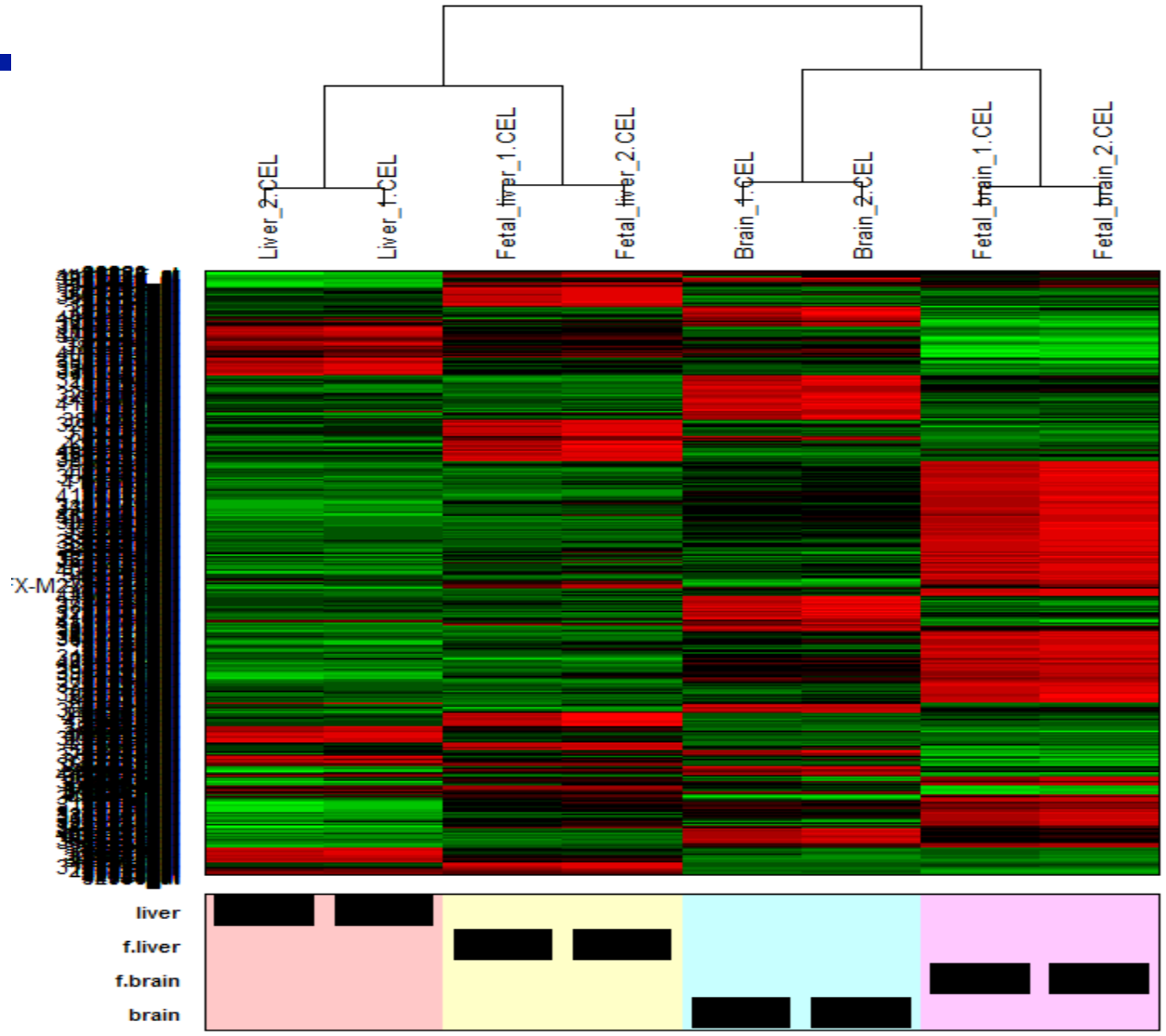
"enhanced heatmap"

```
library(Heatplus)

library(help=Heatplus)
vignette("Heatplus")

library(gplots)
newdata=exprs(eset)[select,]
heatmap_2(newdata,legend=4)
heatmap_2(newdata,legend=4, col=greenred(20),
   legfrac=10)
```

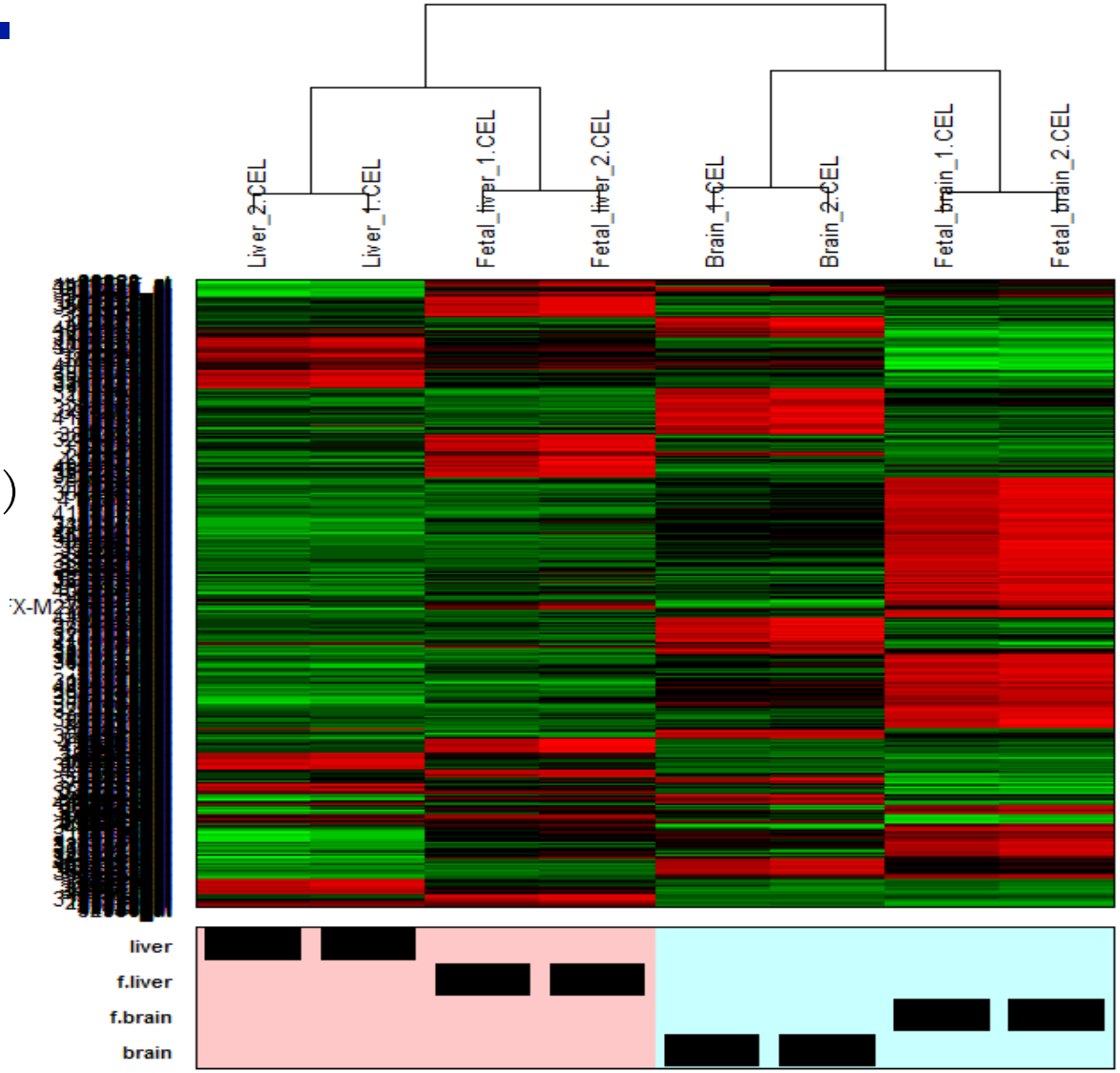

Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
2
1
0
-1

Change the grouping of columns:

```
# create a hierarchical tree "by hand" and cut it
hc= hclust(dist(t(newdata)))
c4=cutree(hc,k=4)
addvar= design
heatmap_plus(newdata, clus=c4, addvar=addvar,
   col=greenred(20))
```


Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
liver
f.liver
f.brain
brain

```
c2=cutree(hc,  k=2)
heatmap_plus(newdata,
addvar=addvar,
clus=c2,col=greenred(20))
```



"Scale" parameter of heatmap

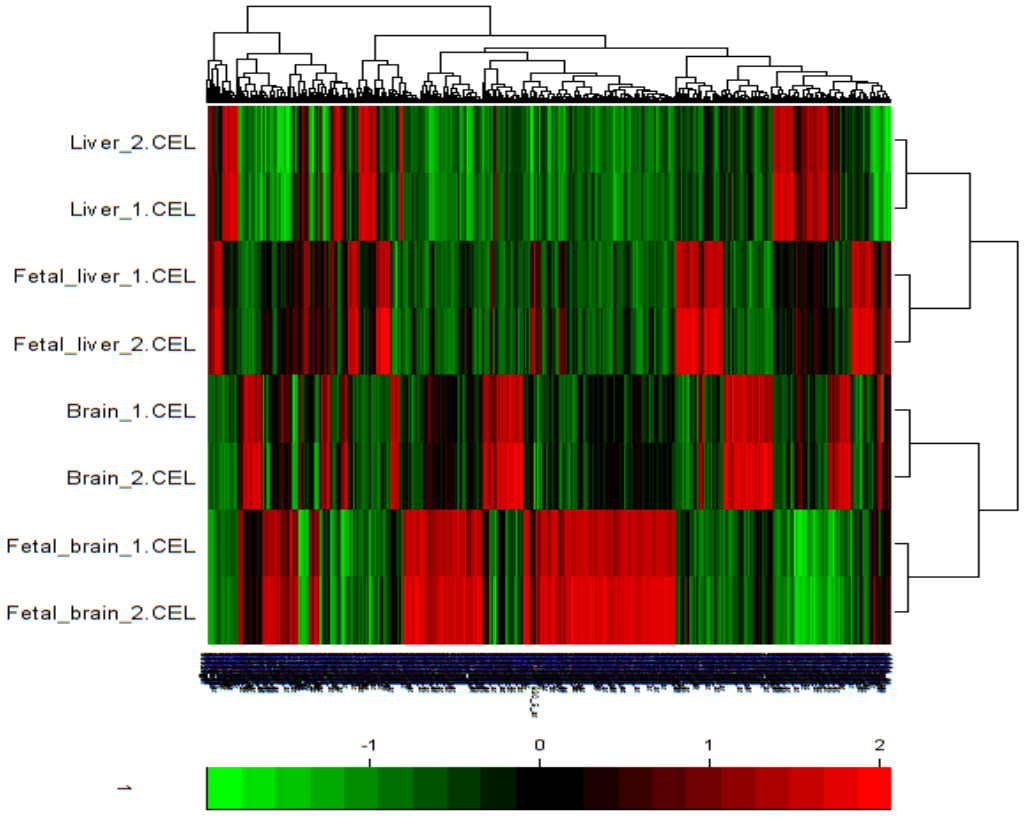
default value is “row”

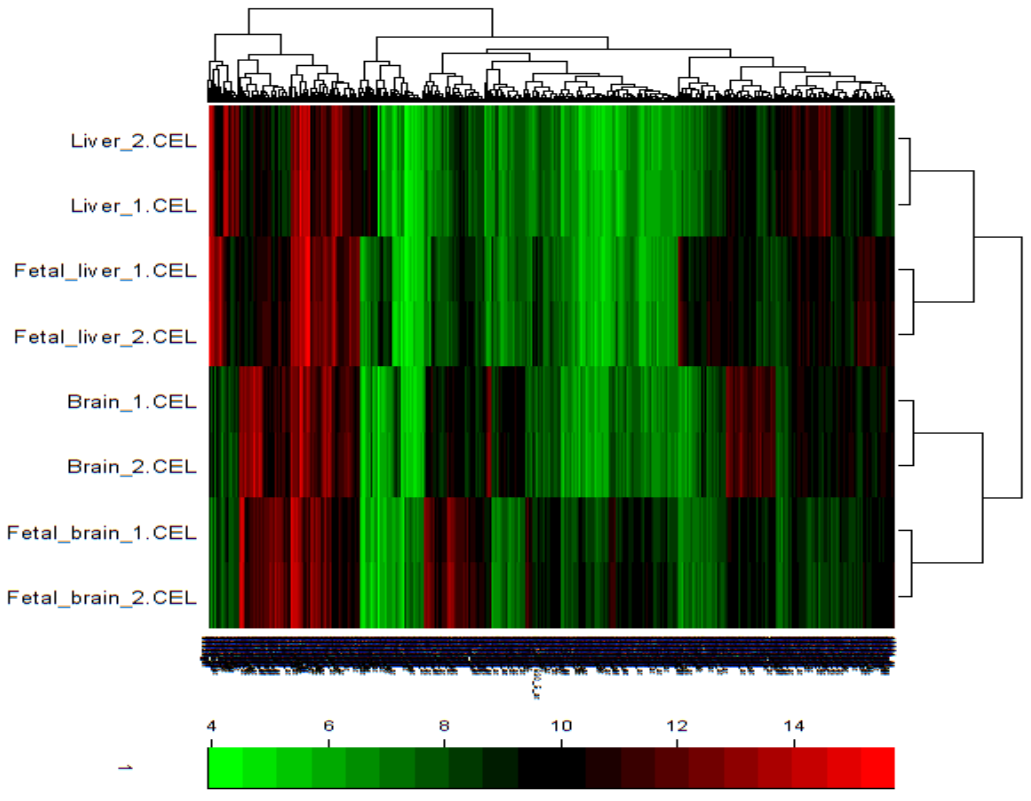
only impacts the colors of the image.

It does not affect the dendrograms.

```
heatmap_2(newdata, scale="row",legend=4,
   col=greenred(20)) ### we already saw it
```

```
heatmap_2(newdata, scale="none", legend=4,
   col=greenred(20))
```

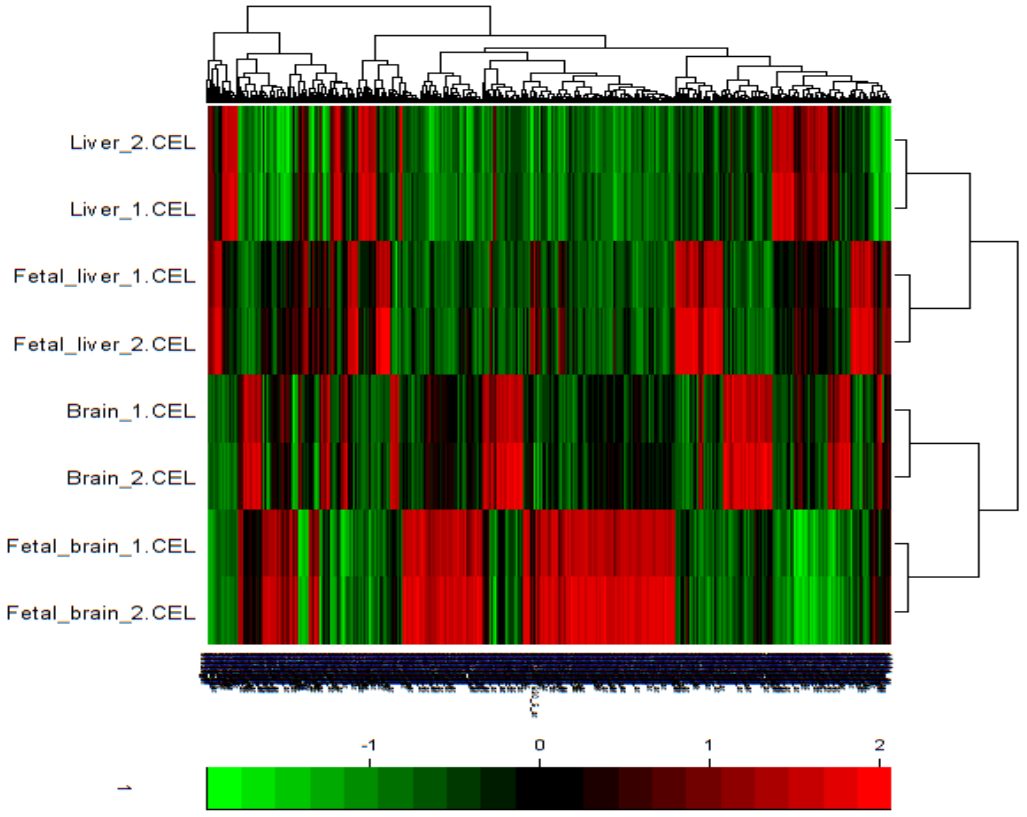

Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
-1
0
1
2

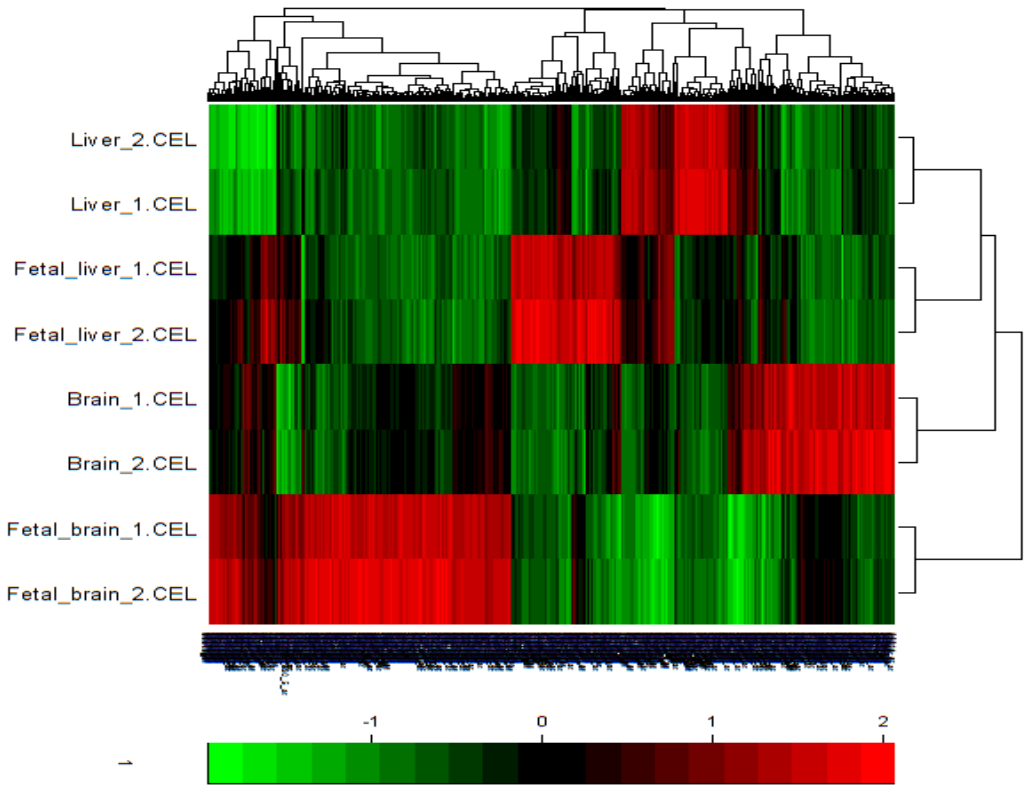

Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
4
6
8
10
12
14

But scaling the data …

#These two codes generate different dendrograms

```
heatmap_2(newdata, scale="row", legend=4,
   col=greenred(20) )  ### we already saw it


heatmap_2(t(scale(t(newdata))), scale="none",
   legend=4, col=greenred(20) )
```


Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
-1
0
1
2


Liver_2.CEL
Liver_1.CEL
Fetal_liver_1.CEL
Fetal_liver_2.CEL
Brain_1.CEL
Brain_2.CEL
Fetal_brain_1.CEL
Fetal_brain_2.CEL
-1
0
1
2

If you focus on the absolute distance among genes, use:

- `heatmap_2(newdata, scale="none", legend=4, col=greenred(20))`

If you're interested in the expression pattern of genes across experiments, use

- `heatmap_2(t(scale(t(newdata))), scale="none", legend=4, col=greenred(20) )`