Lab 6 Clustering Samples and Genes

March 19, 2013

```
> library(GEOquery)

GDS2724=getGEO("GDS2724", destdir=".")
#GDS2724=getGEO(filename="GDS2724.soft.gz")

> E1=exprs(GDS2724)
```

Cleaning up data: are there any genes that we don't have data on?

```
> max(rowSums(is.na(E1)))

[1] 15

> table(rowSums(is.na(E1)))

    0     1     2     3     4     5     6     7     8     9    10    11
52886   932   347   175   118    76    53    33    23     9     9     9
   12    14    15
    3     1     1

> library(genefilter)
> filter0=function(x) sum(is.na(x))<5
> E1=E1[genefilter(E1,filter0),]
```

1 Clustering Samples

Check out the phenotypic data. This is given to you from GEO when you download a dataset instead of individual samples. You may have to construct this for your own datasets.

```
> #These generic functions access the phenotypic data (e.g., covariates) and
> # meta-data (e.g., descriptions of covariates) associated with an experiment.
> pData(GDS2724)
```

```
             sample                protocol
GSM182018 GSM182018                 control
GSM182019 GSM182019                 control
GSM182020 GSM182020                 control
GSM182015 GSM182015           Alk knockdown
GSM182016 GSM182016           Alk knockdown
GSM182017 GSM182017           Alk knockdown
GSM182006 GSM182006        Bmi-1 knockdown
GSM182007 GSM182007        Bmi-1 knockdown
GSM182008 GSM182008        Bmi-1 knockdown
GSM182009 GSM182009       Mel-18 knockdown
GSM182010 GSM182010       Mel-18 knockdown
GSM182011 GSM182011       Mel-18 knockdown
GSM182012 GSM182012 Bmi-1 Mel-18 knockdown
GSM182013 GSM182013 Bmi-1 Mel-18 knockdown
GSM182014 GSM182014 Bmi-1 Mel-18 knockdown

GSM182018        Value for GSM182018: Wild Type, biological rep1; src: Human medul
GSM182019        Value for GSM182019: Wild Type, biological rep2; src: Human medul
GSM182020        Value for GSM182020: Wild Type, biological rep3; src: Human medul
GSM182015        Value for GSM182015: Alk shRNA, biological rep1; src: Human medul
GSM182016        Value for GSM182016: Alk shRNA, biological rep2; src: Human medul
GSM182017        Value for GSM182017: Alk shRNA, biological rep3; src: Human medul
GSM182006       Value for GSM182006: Bmi1 shRNA, biological rep1; src: Human medul
GSM182007       Value for GSM182007: Bmi1 shRNA, biological rep2; src: Human medul
GSM182008       Value for GSM182008: Bmi1 shRNA, biological rep3; src: Human medul
GSM182009      Value for GSM182009: Mel18 shRNA, biological rep1; src: Human medul
GSM182010      Value for GSM182010: Mel18 shRNA, biological rep2; src: Human medul
GSM182011      Value for GSM182011: Mel18 shRNA, biological rep3; src: Human medul
GSM182012 Value for GSM182012: Bmi1 Mel18 shRNA, biological rep1; src: Human medul
GSM182013 Value for GSM182013: Bmi1 Mel18 shRNA, biological rep2; src: Human medul
GSM182014 Value for GSM182014: Bmi1 Mel18 shRNA, biological rep3; src: Human medul
```

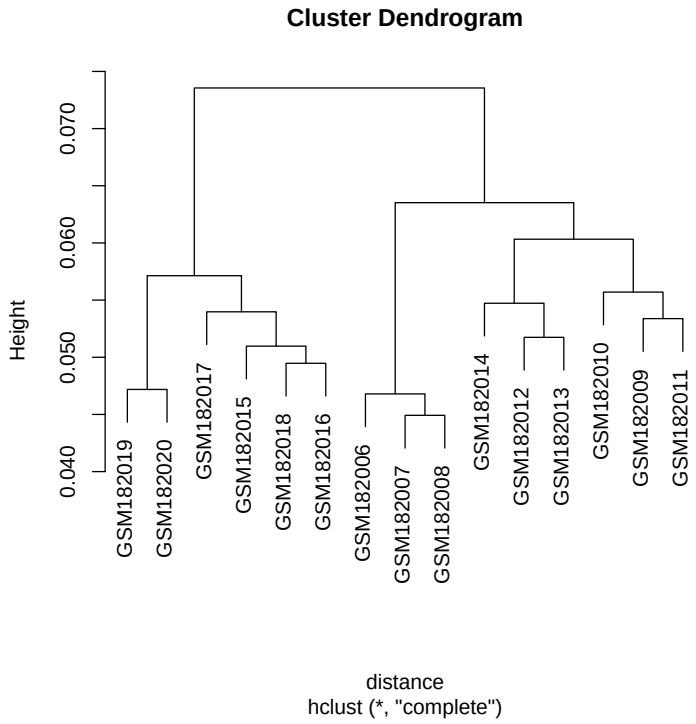
```
> sample.names=pData(GDS2724)[,"protocol"]
```

```
> library(cluster)
> library(hopach)
> dissimilarity <- 1 - cor(E1, use = "complete.obs")
> distance <- as.dist(dissimilarity)
> plot(hclust(distance))
> cluster1 <- hclust(distance)
> #h: numeric scalar or vector with heights where the tree should be cut.
> cluster1B=cutree(cluster1,h=0.02)
> table(cluster1B)
```

```
cluster1B
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
```

```
> d1=distancematrix(t(E1),d="cor")
```



Notice the use of function "t" for transposing a matrix. Why is matrix transposed here?

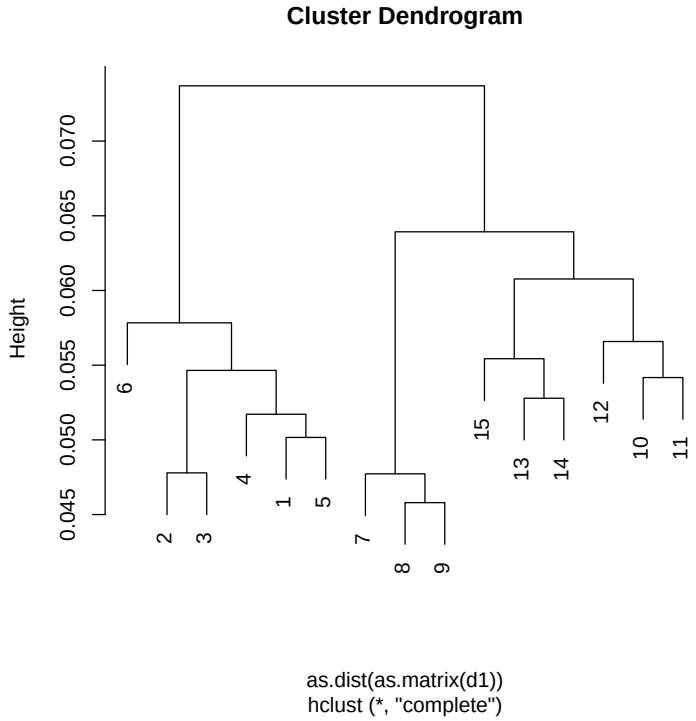
```
> sample.hobj$clust$k
```

```
[1] 8
```

```
> table(sample.hobj$clust$sizes)
```

```
1 2 3
4 1 3
```

```
> plot(hclust(as.dist(as.matrix(d1))))
> plot(hclust(as.dist(as.matrix(d1))), labels=sample.names)
```



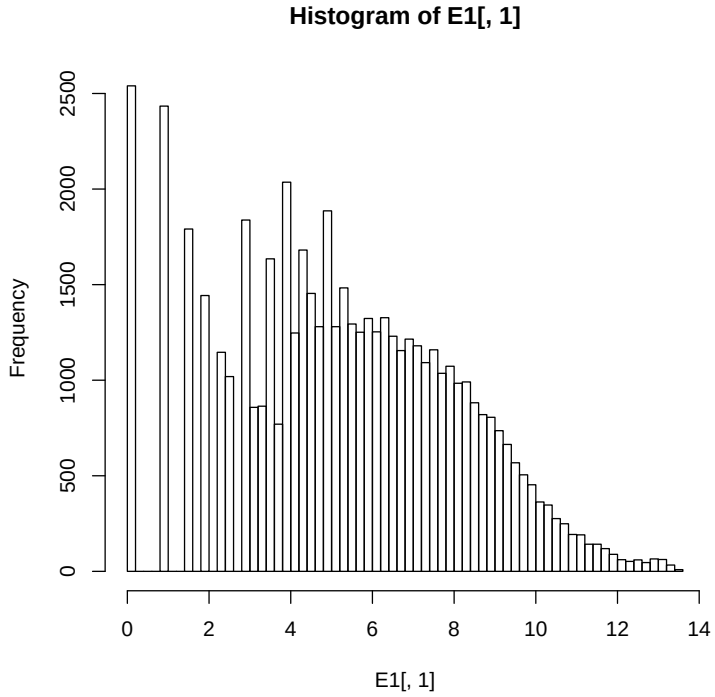
Now we select a subset of genes. As an example we choose genes that are "highly" expressed. The choice is usually rather arbitrary, but the distribution of expression levels will serve as a guide for what levels are "high" in a particular dataset.

```
> hist(E1[,1],50)
> #A function that returns a function with values for A, p and na.rm bound to
> # the specified values. The function takes a single vector, x, as an argument.
> # When the returned function is evaluated it returns TRUE if the proportion
> # of values in x that are larger than A is at least p.
> filter1=pOverA(.3,6)
> ## A gene's expression values has to be >6 for over 30% of the time
> E2=E1[genefilter(E1,filter1),]
> d2=distancematrix(t(E2),d="cor")
> gene.hobj <- hopach(t(E2), dmat = d2)
> gene.hobj$clust$k

[1] 5

> table(gene.hobj$clust$sizes)

3
5
```



To find help for plotting clustering results, first find out what function is associated with the wrapper "plot". Then you can get the appropriate help page.

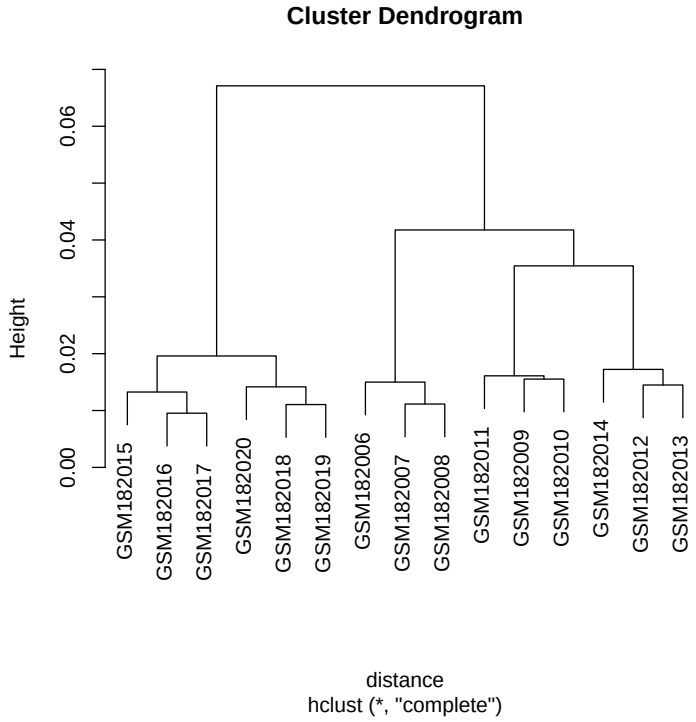
```
> methods("plot")
```

```
 [1] plot.aareg*         plot.acf*           plot.agnes*
 [4] plot.clusGap*       plot.cox.zph*       plot.data.frame*
 [7] plot.decomposed.ts* plot.default        plot.dendrogram*
[10] plot.density        plot.diana*         plot.ecdf
[13] plot.factor*        plot.formula*       plot.function
[16] plot.hclust*        plot.histogram*     plot.HoltWinters*
[19] plot.isoreg*        plot.lm             plot.medpolish*
[22] plot.mlm            plot.mona*          plot.partition*
[25] plot.ppr*           plot.prcomp*        plot.princomp*
[28] plot.profile.nls*   plot.silhouette*    plot.spec
[31] plot.spline*        plot.stepfun        plot.stl*
[34] plot.survfit*       plot.table*         plot.ts
[37] plot.tskernel*      plot.TukeyHSD       plot.xyVector*

   Non-visible functions are asterisked
```

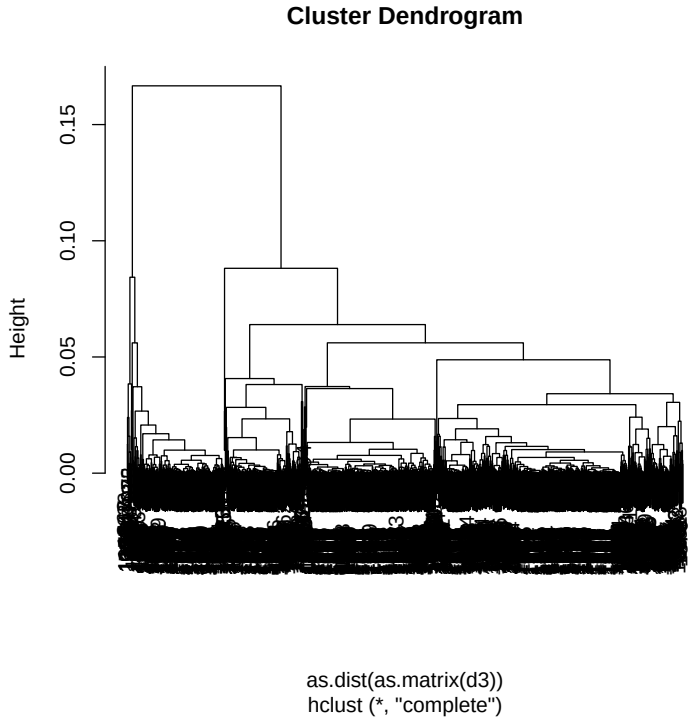
```
?plot.hclust
```

```
> dissimilarity <- 1 - cor(E2, use = "complete.obs")
> distance <- as.dist(dissimilarity)
> plot(hclust(distance))
```



2 Clustering Genes

```
> SDs=apply(E2,1,sd,na.rm=T)
> E3=E2[SDs>.5,]
> d3=distancematrix(E3,d="cosangle")
> cluster3=hclust(as.dist(as.matrix(d3)))
> plot(cluster3)
> plot(cluster3,labels=F)
> cluster3A=cutree(cluster3,h=.3)
```



You can also specify how many branches you want from cutting the tree. Read the help file to find out how to specify the number of clusters.

```
> table(cluster3A)

cluster3A
   1
1650

> E3.1=E3[which(cluster3A==1),]
> heatmap(E3.1)
> heatmap(E3.1,na.rm=T)
> heatmap(E3.1,na.rm=T,Colv=NA)
> #Colv determines if and how the column dendrogram should be reordered. Has
> # the same options as the Rowv argument above and additionally when x is a
> # square matrix, Colv = "Rowv" means that columns should be treated
> # identically to the rows (and so if there is to be no row dendrogram there
> # will not be a column one either).
> heatmap(E3.1,na.rm=T,Colv=NA,labCol=sample.names)
> cluster3C=cutree(cluster3,h=.1)
> table(cluster3C)

cluster3C
   1    2
 288 1362
```

```
> bigclusters=which(table(cluster3C)>=30)
> for (i in bigclusters){
+ subdata=E3[which(cluster3C==i),]
+ heatmap(subdata,na.rm=T,Colv=NA,labCol=sample.names)
+ }
> bigclusters=which(table(cluster4A)>=30)
> for (i in bigclusters){
+ subdata=E3[which(cluster4A==i),]
+ heatmap(subdata,na.rm=T,Colv=NA,labCol=sample.names)
+ }
```