Clustering with Gene Expression Data

Utah State University – Spring 2014

STAT 5570: Statistical Bioinformatics

Notes 2.3

References

- Chapters 10 & 13 of Bioconductor Monograph (course text)
- Chapter 12 of Johnson & Wichern's Applied Multivariate Statistical Analysis, 5th ed.

Visualizing Distances

- Last time: look at “distances” between genes
- Why cluster and why visualization?

  To look for genes with similar - function

  To look for samples with similar - expression profiles
- How visualization?

  scatterplots, trees, “heat”-maps, etc.

Gene expression “vectors”

- For each gene, expression level is estimated on each array
- For many arrays, think of gene expression as a vector
- With many vectors, look at which ones are “close together,” or grouped in “clusters”

Main elements of clustering

- Distance measure

  → get a matrix of pairwise distances
- Number of clusters

  → know when to stop; often subjective
- Criterion

  → know what makes a “good” cluster
- Search strategy

  → how to optimize criterion

Types of search strategies

- Partitioning

  map vectors to clusters (max. some criterion)

- Hierarchical

  construct a “tree” showing similarities between groups

  - Divisive

    top-down (from single cluster to more; like forward sel.)

  - Agglomerative

    bottom-up (from singletons to one cluster; like backward sel.)

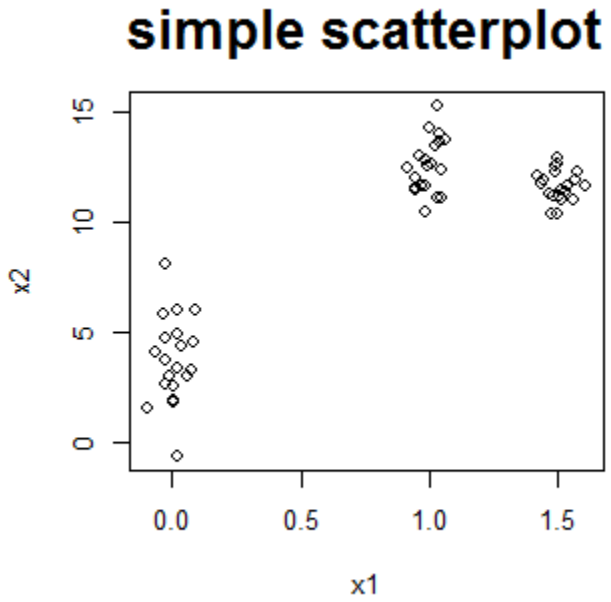
- Hybrid

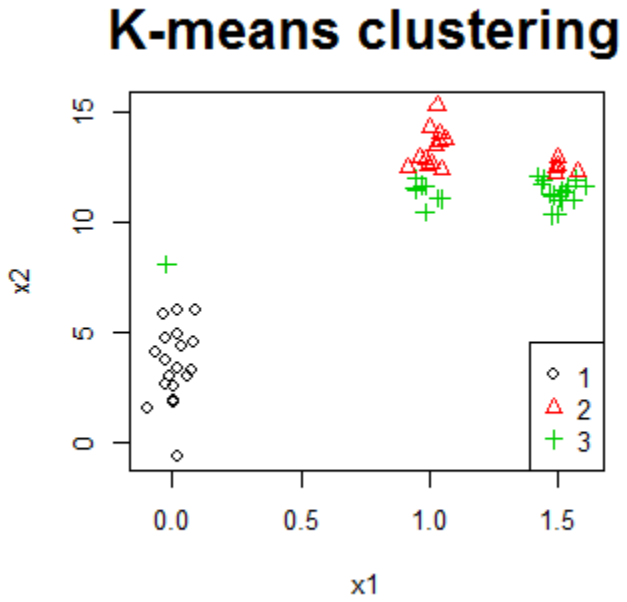
  alternate divis. / agglom. steps to choose # “children” at each step (like stepwise sel.)

Partitioning example: K-means clustering

- Search strategy: (partitioning)
  - Select K initial “centroids”- (or “mediods” for PAM)
  - Assign each vector to be clustered with nearest centroid
  - Recalculate K centroids
  - Repeat assignment and centroid calculation until “convergence”
- What are the [possibly subjective] choices here?
  - K – how many clusters are there?
  - initial centroids

Example – what's the problem here?





How to "fix" this?

Think of x1 as being: expression on array 1

and x2 as being: expression on array 2

Here, there are 60 "genes"

A simple K-means example in R

```
set.seed(123)
x1 <- c(rnorm(20,sd=.05),
        rnorm(20,mean=1,sd=.05),
        rnorm(20,mean=1.5,sd=.05))
x2 <- 4+c(rnorm(20,sd=2),
          rnorm(20,mean=8,sd=1.5),
          rnorm(20,mean=8,sd=1))

par(mfrow=c(2,2))
plot(x1,x2,main='simple scatterplot',cex.main=2)

k.fit <- kmeans(x=cbind(x1,x2),centers=3)
plot(x1,x2, main='K-means clustering',cex.main=2,
     pch=k.fit$cluster,col=k.fit$cluster)
legend('bottomright', legend=c(1:3),pch=1:3,col=1:3)
```

Concerns with partitioning methods

- Normalization

  need same scale for all dimensions (arrays)
- Choice of k

  best if chosen based on “expert” knowledge
- Optimality

  best 2-cluster arrangement not necessarily a subset of best 3-cluster arrangement –

  → need for justified choice of k

Hierarchical example: agnes

- Agglomerative Nesting – (hierarchical clustering)

- Start with "singleton" clusters –
  each vector is its own group
- Find the two "closest" vectors and "merge" them –
  distance usually Euclidean; form a group
- Then recalculate distances:
  Linkage – distance between groups
  - Average linkage –
    distance is average of dissimilarities between groups
  - Single linkage –
    distance is dissimilarity between "nearest neighbors"

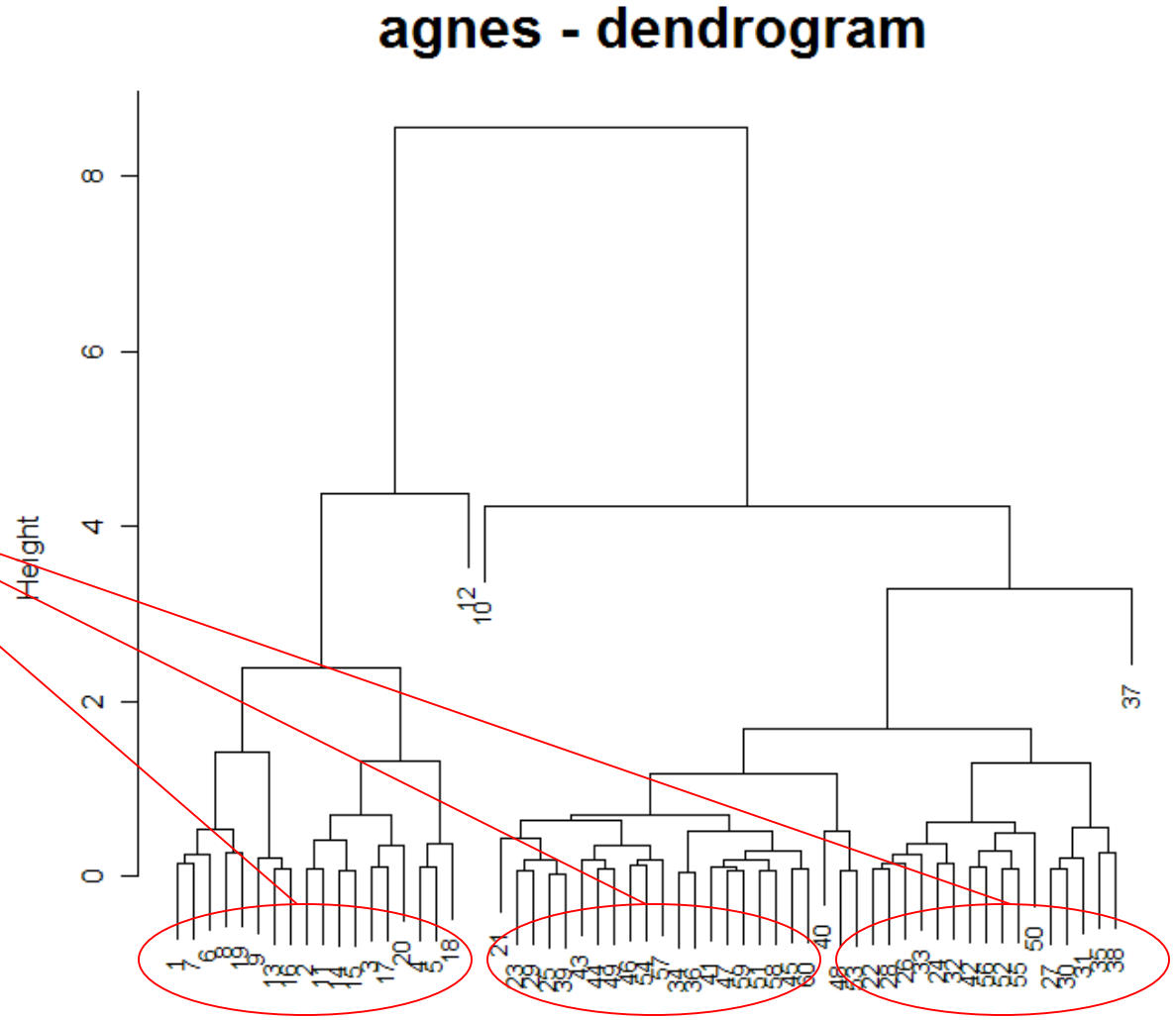
Hierarchical example: diana

- Divisive Analysis Clustering

- 1. All genes start out in same cluster
- 2. Find “best” split to form two new clusters
  - “best” – maximize “distance” between new clusters
  - “distance” between new clusters: **linkage** - average, single (nearest neighbor), etc.
- 3. Repeat step 2 until each gene is its own cluster
- (Same with samples)
- Visualize with a dendrogram (tree)

Problem here?

wrong clusters



measures clustering structure of data (0 to 1); only useful to compare same-size data sets

Visualization of hierarchical clustering

- Create a “tree” diagram indicating grouped vectors -

  dendrogram

- What should vertical height represent?

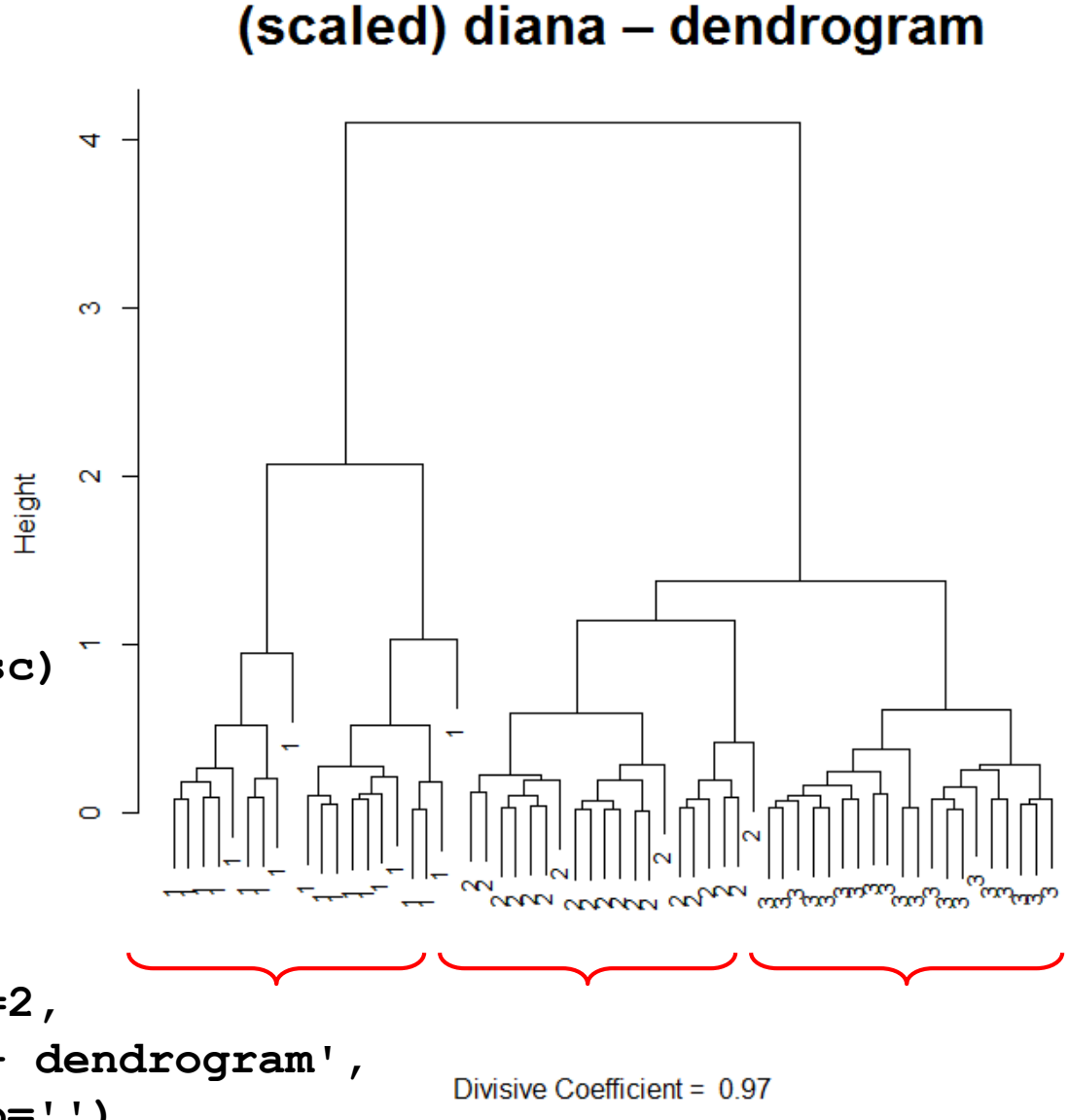
  - “distance” between groups

```
library(cluster)
ag.fit <- agnes(cbind(x1,x2))
par(mfrow=c(1,1))
plot(ag.fit,which.plots=2,main='agnes - dendrogram',
  cex=.8,cex.main=2,xlab='')
```

Re-scaling changes:

cluster assignments

```
x1.sc <- x1/sd(x1)
x2.sc <- x2/sd(x2)
x.sc <- cbind(x1.sc,x2.sc)
rownames(x.sc) <-
 c(rep(1,20),
   rep(2,20),
   rep(3,20))
di.fit <- diana(x.sc)
plot(di.fit,which.plots=2,
  main='(scaled) diana - dendrogram',
  cex=.8,cex.main=2,xlab='')
```

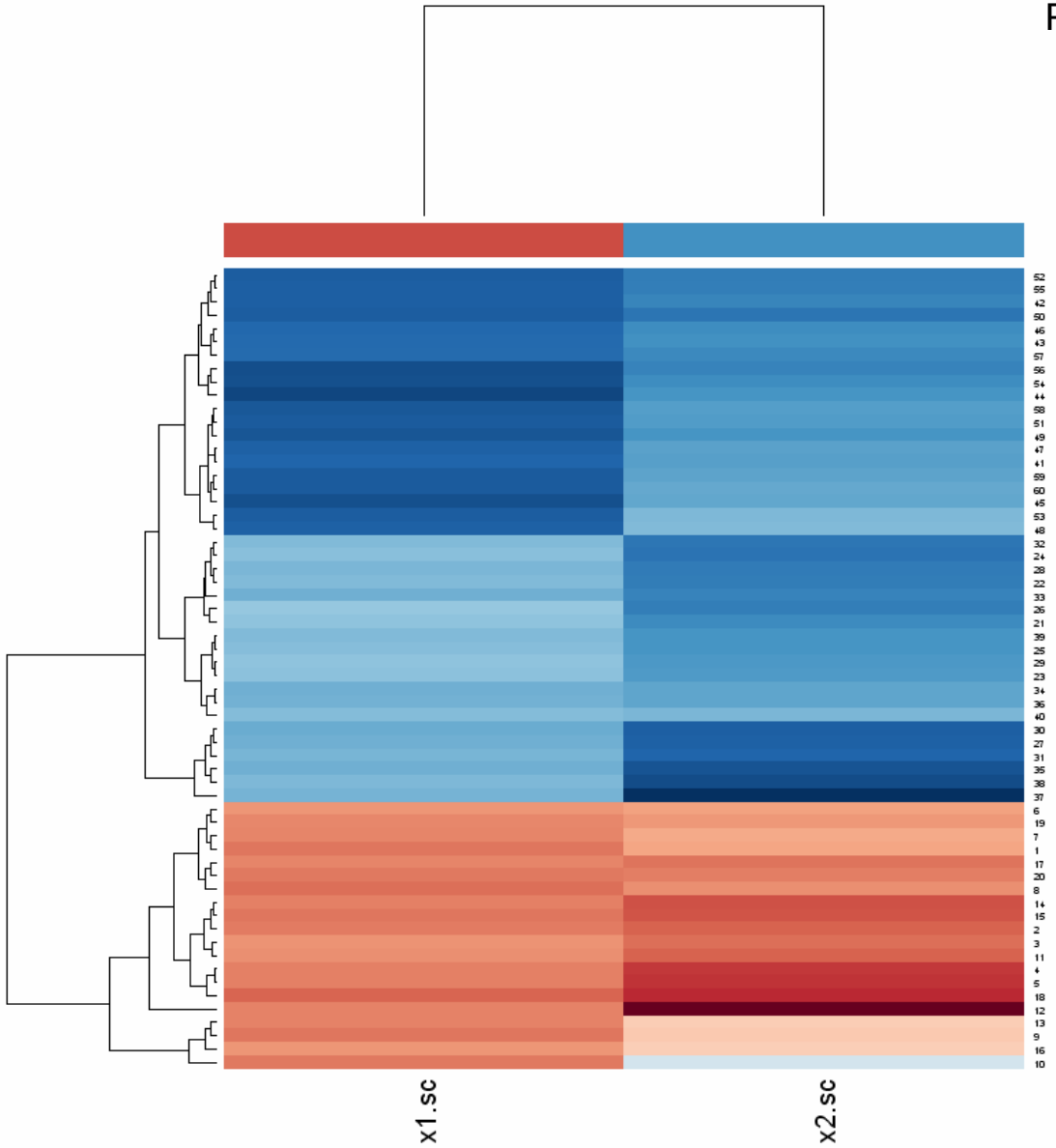


Another way to visualize hierarchical clustering

- Heat map

  also called a - false color image

- Consider data arranged in a matrix with columns and rows ordered according to "similarity" - (to show structure)
  - Think of cols. for arrays and rows for genes, maybe
  - "Similarity" based on hierarchical clustering, maybe
- Transform matrix values to color scale – to visualize the structure created by clustering



Recall: x1 as expression on array 1
x2 as expression on array 2
60 “genes”

Look for:

- distinct “blocks”

What do these mean?

hopefully, meaningful differences

Heat maps in R

```
library(cluster)
library(RColorBrewer)
set.seed(123)
x1 <- c(rnorm(20,sd=.05),
        rnorm(20,mean=1,sd=.05),
        rnorm(20,mean=1.5,sd=.05))
x2 <- 4+c(rnorm(20,sd=2),
          rnorm(20,mean=8,sd=1.5),
          rnorm(20,mean=8,sd=1))
x1.sc <- x1/sd(x1)
x2.sc <- x2/sd(x2)
hmcol <- colorRampPalette(brewer.pal(10,"RdBu"))(256)
csc <- c(hmcol[50],hmcol[200])
heatmap(cbind(x1.sc,x2.sc),scale="column",col=hmcol,
        ColSideColors=csc,cexCol=1.5,cexRow=.5)
```

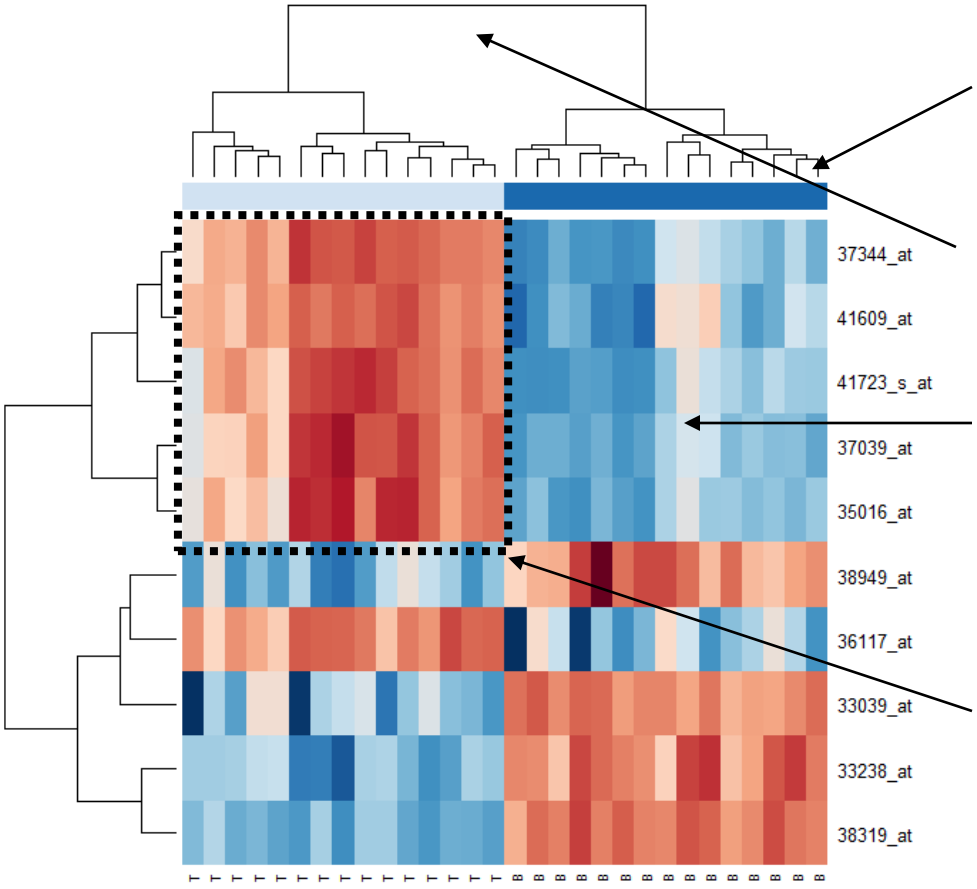
this heatmap function applies dendrogram equivalent to diana (divisive analysis clustering); default is scale="row"

Recall: ALL Data

- preprocessed gene expression data (RMA)
- Chiaretti et al., Blood (2004) 103(7)
- 12625 genes (hgu95av2 Affymetrix GeneChip)
- 128 samples (arrays)
- phenotypic data on all 128 patients, including:
  - 95 B-cell cancer
  - 33 T-cell cancer

- Just as an example, look at a subset here:
  - 12 genes (Unit 3 will look at finding such ‘candidates’)
    - 10 with known common function (apoptosis)
    - 2 unknown
  - 30 arrays (15 B-cell and 15 T-cell; patients 81-110)

Heatmap with gene expression data



Can add column and row side colors to facilitate interpretation

Dendrograms on columns and rows
Here, using agglomerative nesting

Color represents relative expression level, but on what scale?
Here, scale is red (low) to blue (high); light colors are medium

Look for: distinct blocks
- especially meaningful differences

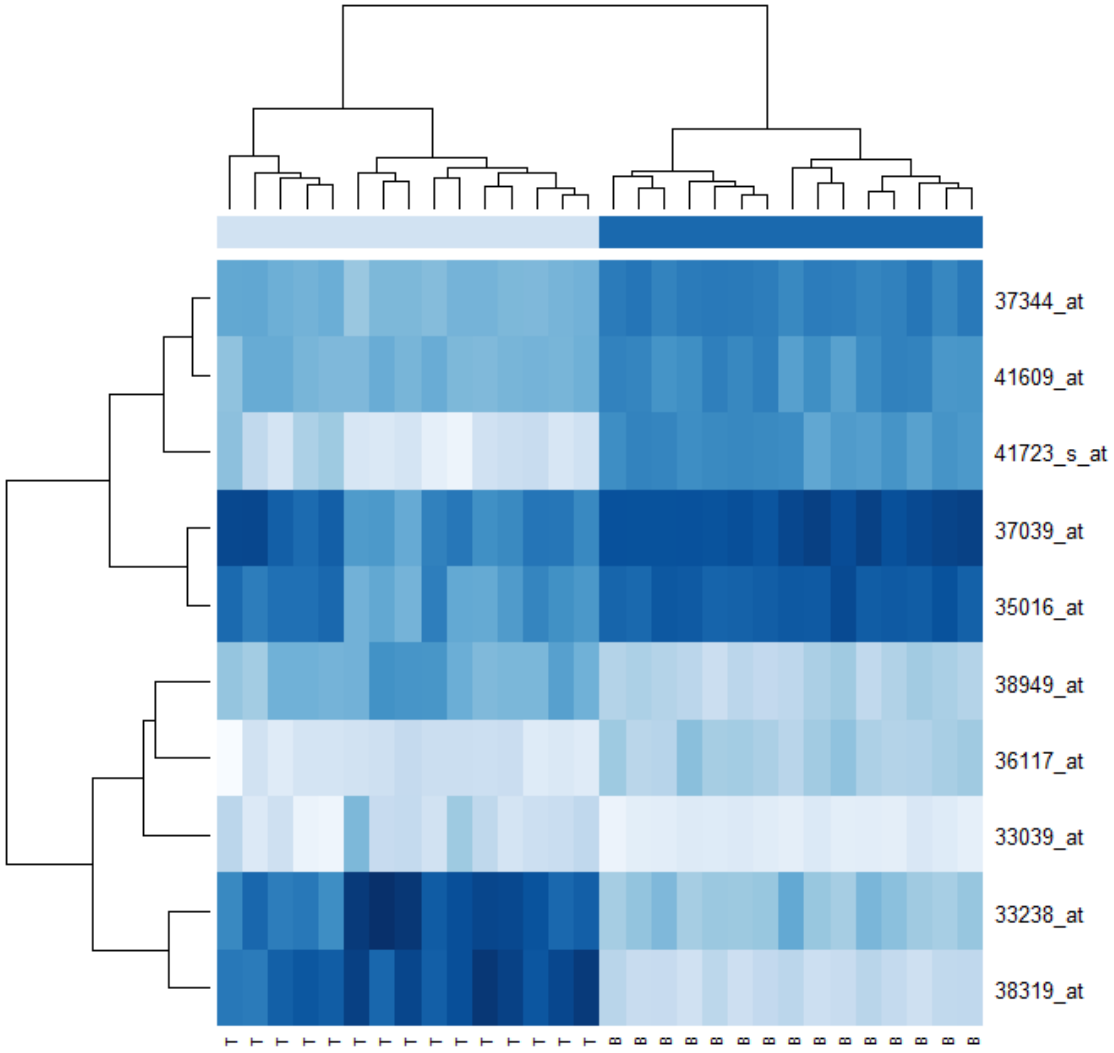
Ask: what distance measure, and what linkage method?
Here, Euclidean distance with complete (farthest neighbors) linkage

Why such distinct blocks here?
- genes selected for differentiation

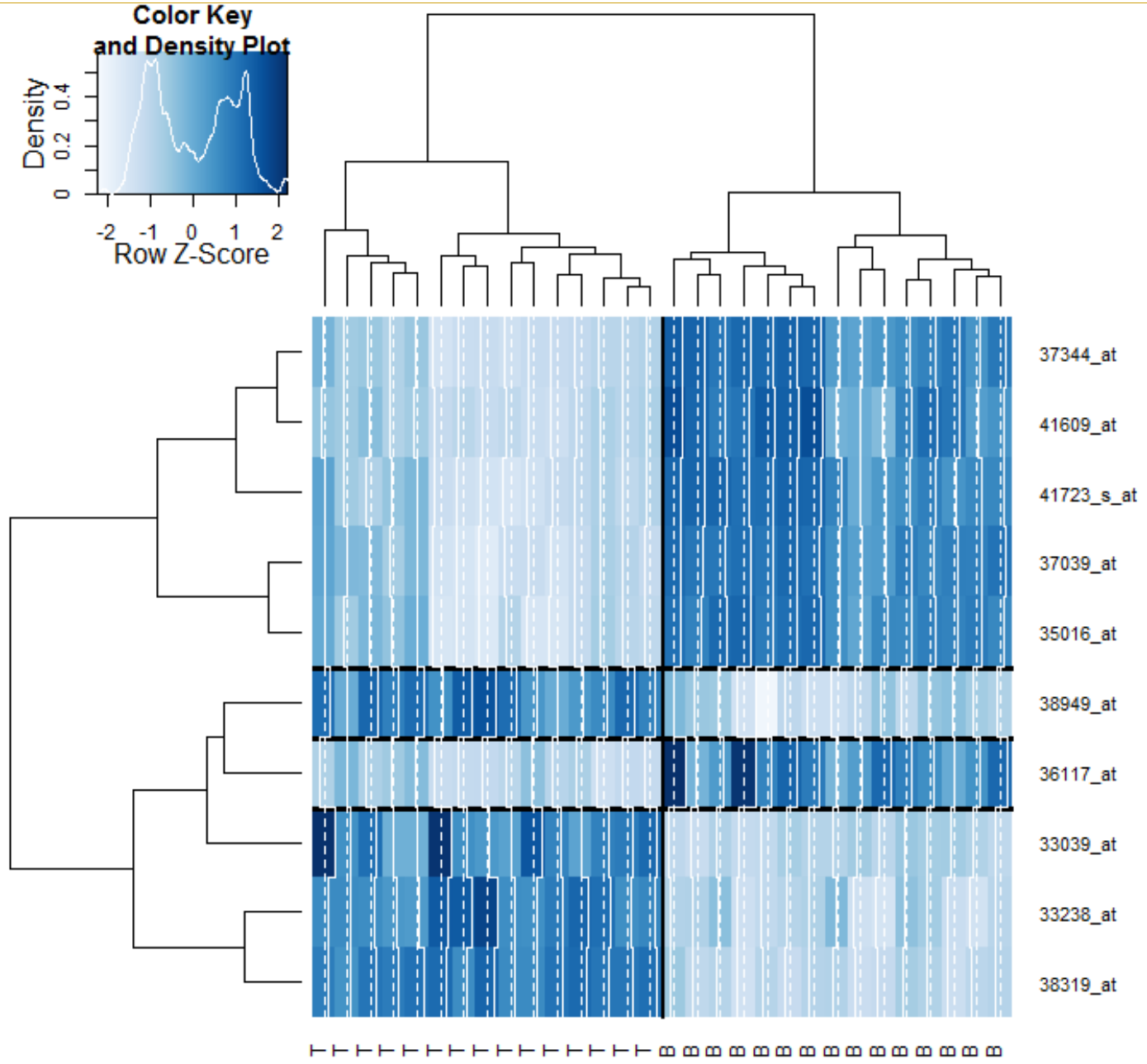
```
library(affy); library(ALL); data(ALL)
gn <- featureNames(ALL) # ALL is an ExpressionSet object

# Choose a subset of genes (How in Unit 3)
gn.list <- c("37039_at","41609_at","33238_at","37344_at",
             "33039_at","41723_s_at","38949_at","35016_at",
             "38319_at","36117_at")
t <- is.element(gn,gn.list)
small.eset <- exprs(ALL)[t,c(81:110)]

# Define color scheme
hmcol <- colorRampPalette(brewer.pal(10,"RdBu"))(256)

## Columns
# Here, the first 15 are B-cell patients; rest are T-cell
cell <- c(rep('B',15),rep('T',15))
csc <- rep(hmcol[50],ncol(small.eset))
csc[cell=='B'] <- hmcol[200]
colnames(small.eset) <- cell
```

```
## Rows
rownames(small.eset)
# [1] "33039_at"   "33238_at"   "35016_at"   "36117_at"
# [5] "37039_at"   "37344_at"   "38319_at"   "38949_at"
# [9] "41609_at"   "41723_s_at"

## (Could add rsc [row-side colors] similar to csc
## if we knew functional differences among genes)

## Make heatmap
heatmap(small.eset,scale="row",col=hmcol, ColSideColors=csc)
```

Alternative color scheme



```
# Use alternative color scheme
hmcol2 <- colorRampPalette(brewer.pal(10,"Blues"))(256)

csc <- rep(hmcol2[50],ncol(small.eset))
csc[cell=='B'] <- hmcol2[200]
heatmap(small.eset,scale="col", col=hmcol2,
  ColSideColors=csc)
```

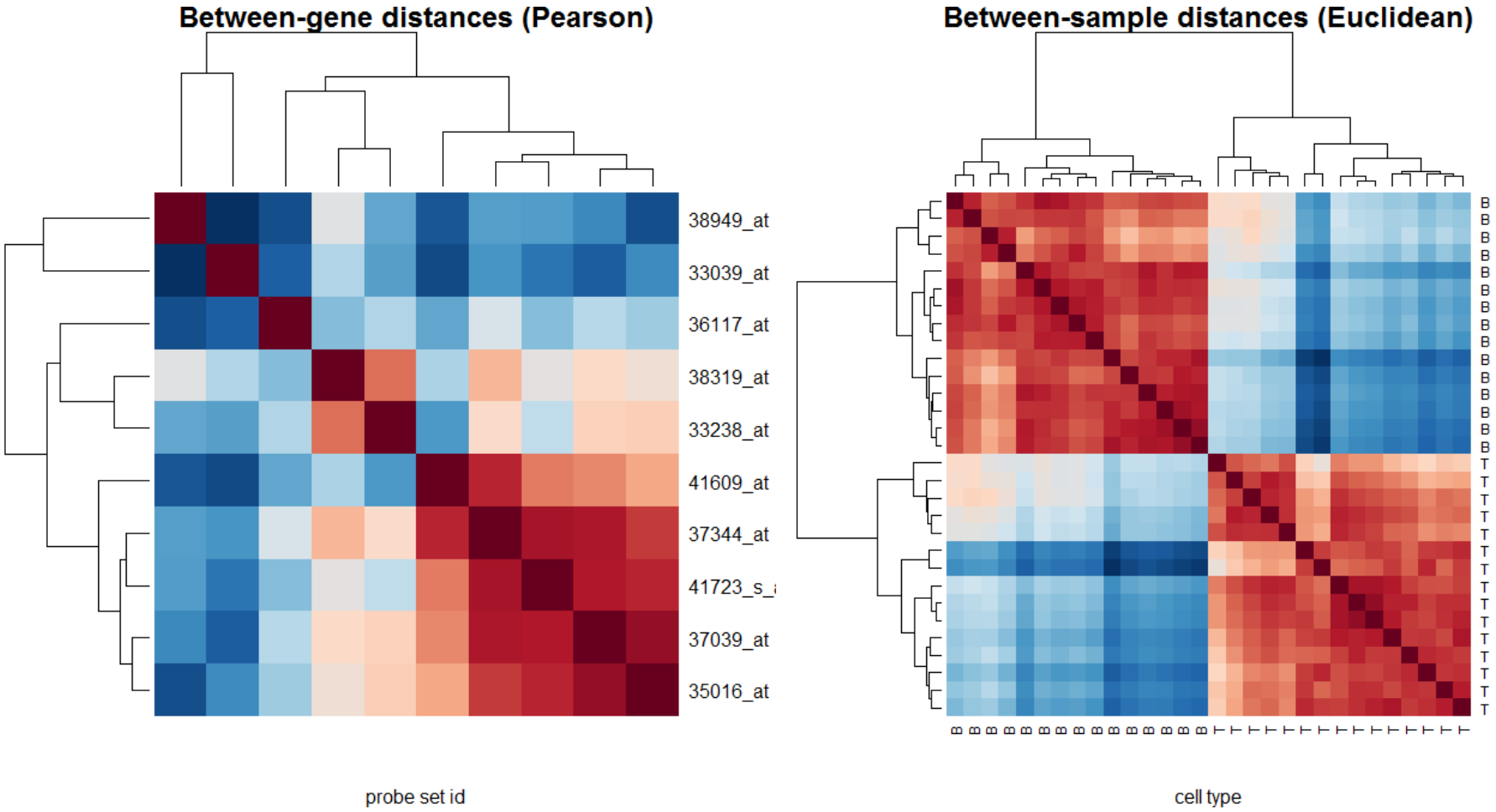

Color Key
and Density Plot
Density
0
0.2
0.4
-2
-1
0
1
2
Row Z-Score
37344_at
41609_at
41723_s_at
37039_at
35016_at
38949_at
36117_at
33039_at
33238_at
38319_at

```
# Look at 'enhanced' heatmap
### The white dashed lines show the average for each column
### (on the left-to-right scale of the key),
##  whereas the white solid lines show the deviations from
##  the average within arrays.

library(gplots)
heatmap.2(small.eset, col=hmcol2, scale="row",
       margin=c(5, 5), cexRow= 0.8,
       # level trace
       tracecol='white',
       # color key and density info
       key = TRUE, keysize = 1.5, density="density",
       # manually accentuate blocks
       colsep=c(15), rowsep=c(5,6,7), sepcolor='black')
```

Alternative use for heat maps

- Visualize “distances” between –
  arrays (samples)
- Recall defining distance between genes and between arrays – get a distance matrix
- Run heatmap on this distance matrix -
  often called false color image

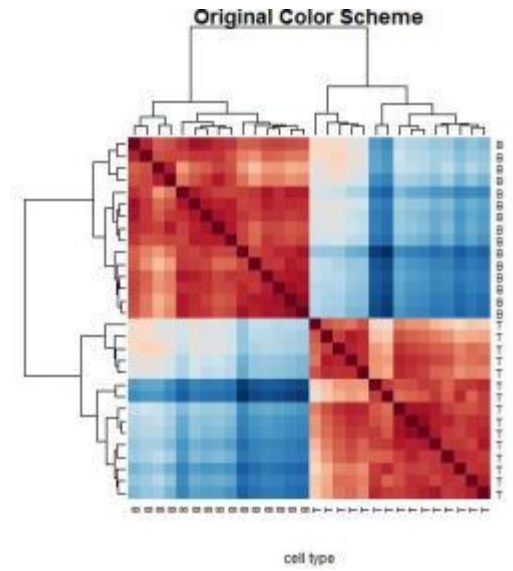
Look for: “distinct” blocks

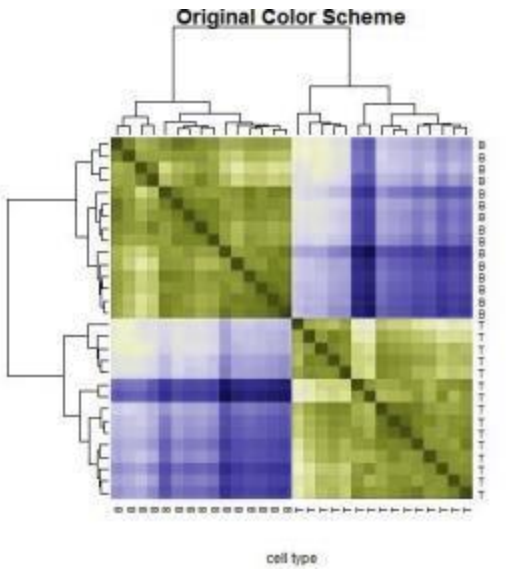


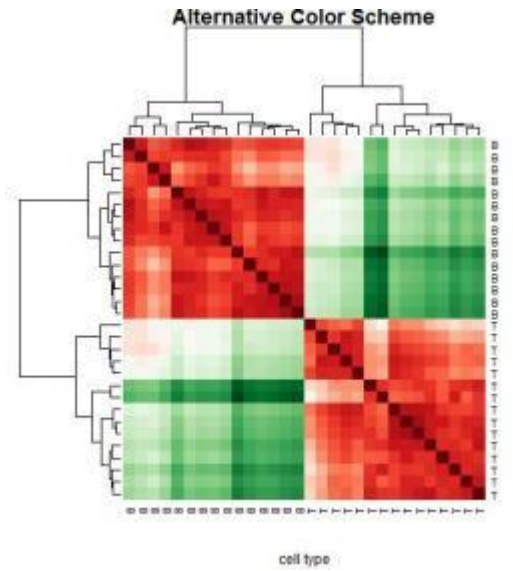
Note: bottom 3-4 genes may share common function

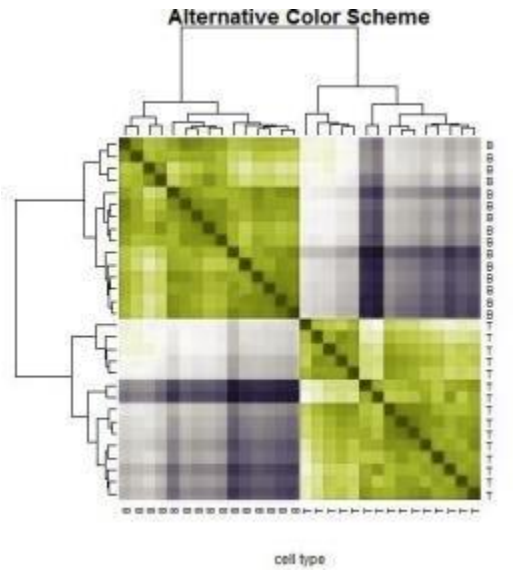
False color images in Bioconductor

```
library(bioDist)

d.gene.cor <- cor.dist(small.eset)
heatmap(as.matrix(d.gene.cor),sym=TRUE,col=hmcol,
        main='Between-gene distances (Pearson)',
        xlab='probe set id',labCol=NA)

d.sample.euc <- euc(t(small.eset))
heatmap(as.matrix(d.sample.euc),sym=TRUE,col=hmcol,
        main='Between-sample distances (Euclidean)',
        xlab='cell type')
```









(deuteranopia)

Eventually, want to visually display "significant" results.

Need to be able to "tell a story"

A poor color scheme (like red/green) could prevent a large percentage of the audience from gaining any information from the visualization.

```
# Consider alternative color scheme

red.hmcol <- colorRampPalette(brewer.pal(9,"Reds"))(256)
green.hmcol <- colorRampPalette(brewer.pal(9,"Greens"))(256)
use.hmcol <- c(rev(red.hmcol),green.hmcol)

jpeg('C:\\folder\\orig.jpg')
heatmap(as.matrix(d.sample.euc),sym=TRUE,col=hmcol,
  main='Original Color Scheme',xlab='cell type')
dev.off()

jpeg('C:\\folder\\alt.jpg')
heatmap(as.matrix(d.sample.euc),sym=TRUE,col=use.hmcol,
  main='Alternative Color Scheme',xlab='cell type')
dev.off()

# Upload jpeg files to:
#   http://www.vischeck.com/vischeck/vischeckImage.php
```

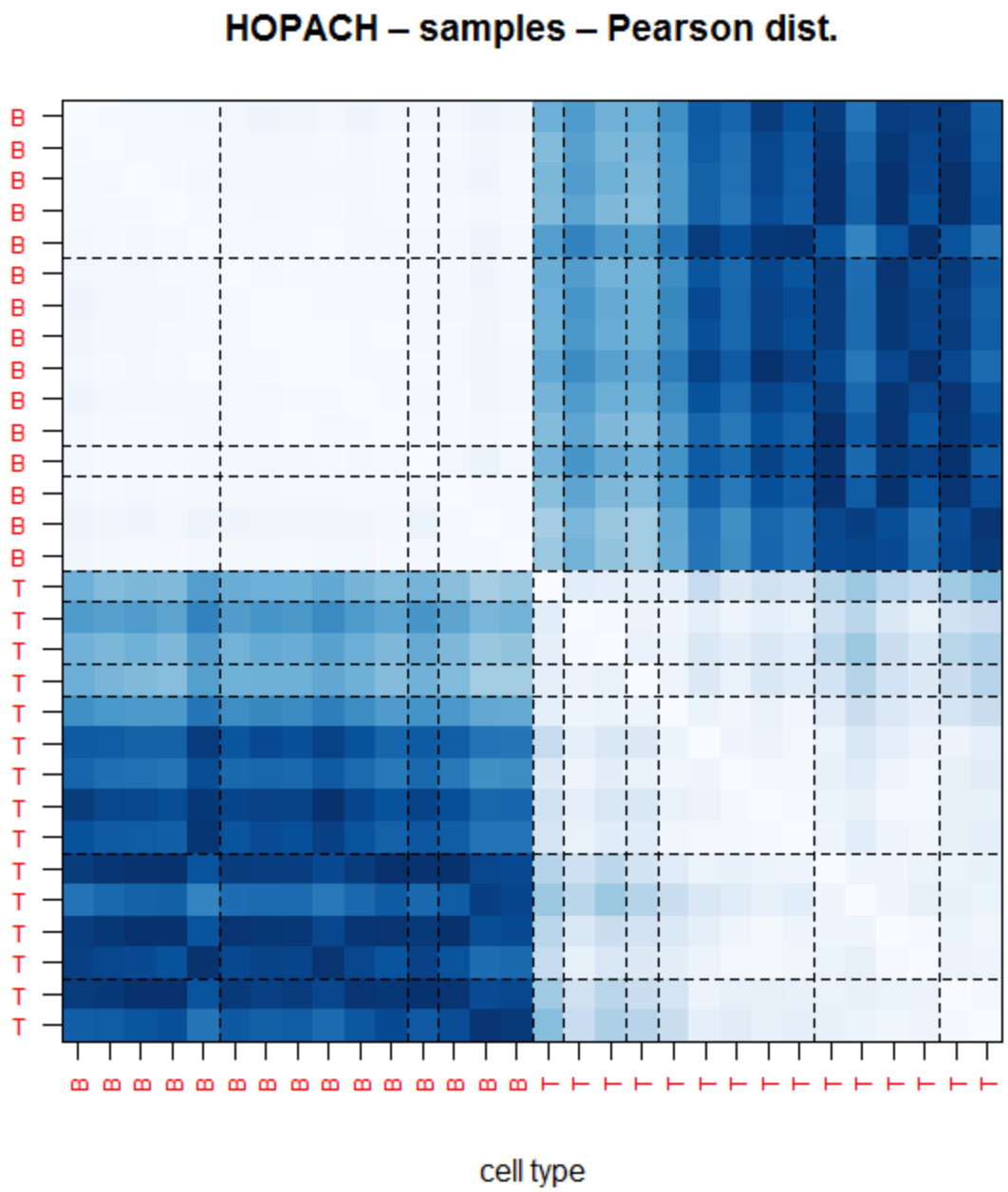
Hybrid example: HOPACH

- Hierarchical Ordered Partitioning And Collapsing Hybrid

- Iterative procedure:
  - Partition (into “child” clusters)
    - determine the appropriate number of “children” (k=2,…,9) using PAM (“k-Mediods”)
    - choose k minimizer of “median split silhouette” MSS, a measure of cluster heterogeneity
  - Order

    child clusters by distance of medioids from “uncle” mediods
  - Collapse

    collapse two clusters if doing so will improve MSS



Look for:

distinct blocks

Dotted lines:

“breaks” between clusters

Median split silhouette (MSS)

- Want to find small clusters inside big clusters
- For each vector, measure how well matched it is to the other vectors in the cluster compared to if it were switched to the next closest cluster – call it the vector's silhouette (scale from -1 to 1)
- Split each of K clusters into sub-clusters -
  (# chosen at each cluster to maximize average silhouette)
- Re-evaluate each vector's silhouette (relative to other siblings), and take median within parent cluster –
  call it the split silhouette
- Take median across all parent clusters (MSS) –
  it is a measure of overall homogeneity within clusters
  when K clusters are used

```
library(affy); library(ALL); data(ALL)
gn <- featureNames(ALL)

# Choose a subset (same as slide 21)
gn.list <- c("37039_at","41609_at","33238_at","37344_at",
             "33039_at","41723_s_at","38949_at","35016_at",
             "38319_at","36117_at")
t <- is.element(gn,gn.list)
small.eset <- exprs(ALL)[t,c(81:110)]
cell <- c(rep('B',15),rep('T',15))
colnames(small.eset) <- cell

# HOPACH
library(hopach)  # dimension to be clustered should be rows
array.hop <- hopach(t(small.eset),d='cor')
```

```
# Look at num. of clusters
array.hop$clust$k
# 10

# Look at sizes of clusters
table(array.hop$clust$sizes)
# 1 2 3 4 5 6
# 3 2 1 1 2 1

# Look at graph
library(RColorBrewer); library(bioDist)
hmcol <- colorRampPalette(brewer.pal(10,"Blues"))(256)
d.mat.cor <- as.matrix(cor.dist(t(small.eset)))

dplot(d.mat.cor,array.hop,lab=colnames(small.eset),
  showclust=T, col=hmcol, xlab='cell type',
  main='HOPACH - samples - Pearson dist.')
```

Inference with clusters

- Are the clusters “real” or just due to noise?

- Look at bootstrapping
  - put vectors as rows in a matrix
  - sample the n columns with replacement, n times
  - run hopach on resampled matrix
    - find final mediods
    - assign original vectors to nearest final mediods
  - for each vector, calculate percentage of resamplings that result in same cluster assignments

- Visualization of bootstrapping –

  estimate of membership of vectors in clusters

Bootstrapping HOPACH in R

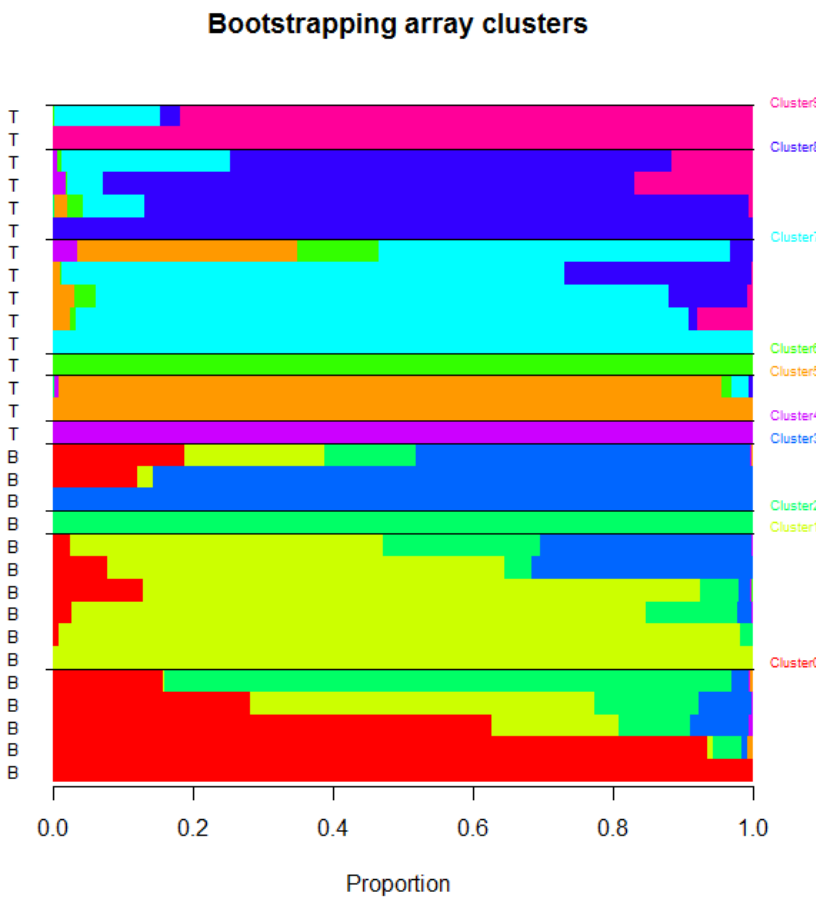
```
array.hop <- hopach(t(small.eset),d='cor')

bobj <- boothopach(t(small.eset),array.hop, B=1000)

bootplot(bobj,array.hop,ord="bootp",showclusters=TRUE,
         main='Bootstrapping array clusters')
```

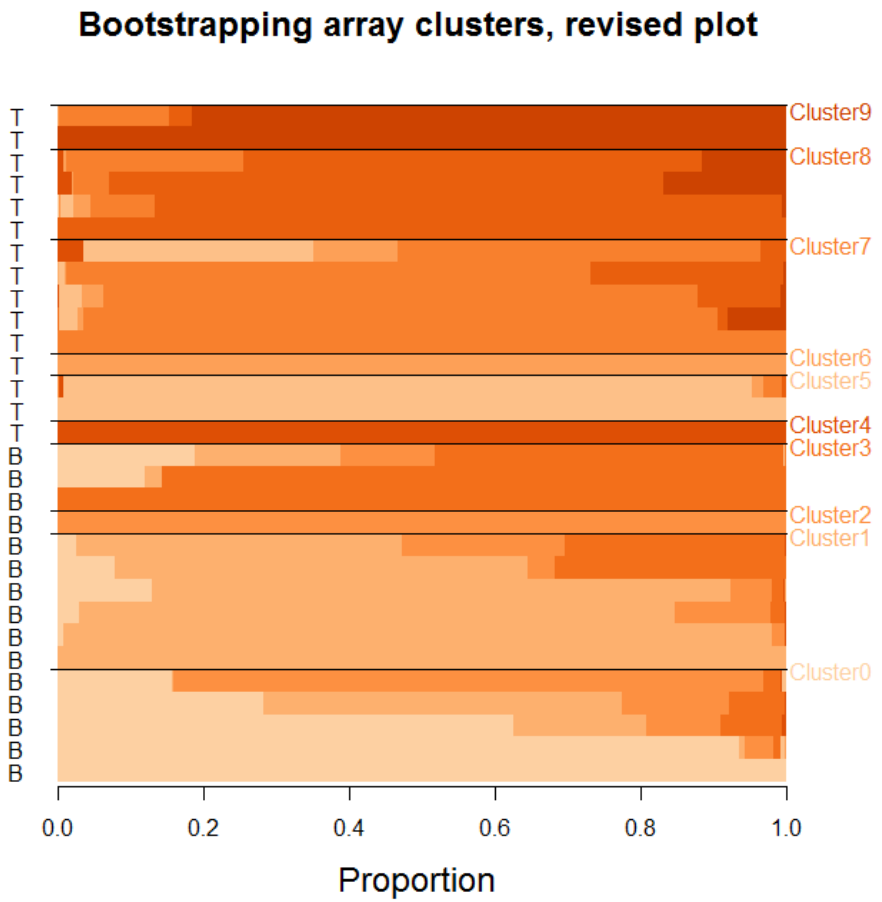
Interpretation:

For each vector (here, array), in what proportion of the resampled data sets was the vector -
"assigned" to which cluster?

Gives indication of - "confidence" of cluster membership

bootplot – default and revised





Look at bootplot function (portions)

```
function (bootobj, hopachobj, ord = "bootp", main = NULL,
  labels = NULL, showclusters = TRUE, ...)
{
    p <- nrow(bootobj)
    k <- ncol(bootobj)
    # ...
    colors <- rainbow(k)
    colors <- c(colors[seq(1, k, by = 2)], colors[seq(2, k, by = 2)])
    # ...
    barplot(t(bootobj), ylim = c(1, p), border = FALSE,
      space = 0, horiz = TRUE, names.arg = labels[ordering],
      las = 1, main = main, cex.names = 0.75, legend.text = FALSE,
      col = colors, axisnames = shownames, xlab = "Proportion", ...)
    if (showclusters) {
        abline(h = cumsum(hopachobj$clust$sizes))
        mtext(colnames(bootobj), outer = TRUE, side = 4,
            at = cumsum(hopachobj$clust$sizes)/p * 0.7 + 0.16,
            line = -2, col = colors, las = 1, cex = 0.6)
    }
}
```

Summary

- Clustering methods for microarray data
  - partitioning
  - hierarchical
  - hybrid
- Visualization of clustering
  - dendrograms
  - heat maps
  - bootstrapped “confidence” of cluster memberships (seed?)