ELEMENTS OF DATA SCIENCE AND STATISTICAL LEARNING

SPRING 2020

Week 9

OUTLINE

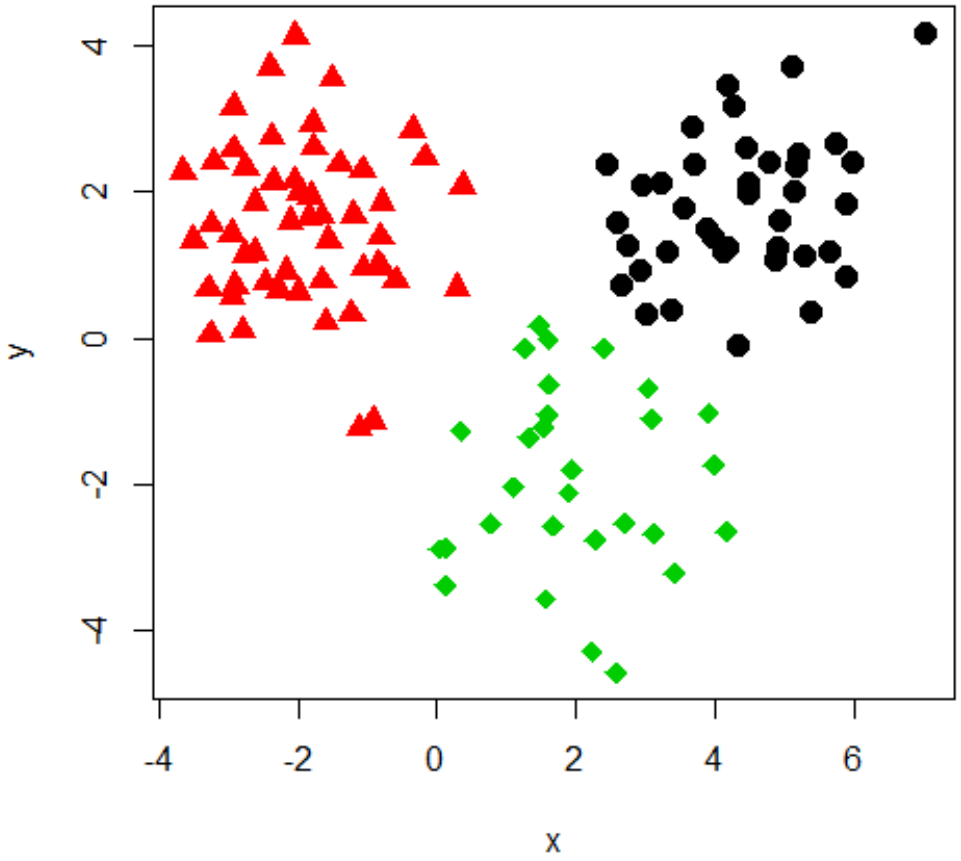
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

TEST DATASET

- Let us consider again our artificial example. We will use it to develop some intuition (and code)

```
x=c(rnorm(30,mean=2),rnorm(50,mean=-2),rnorm(40,mean=4))
y=c(rnorm(30,mean=-2),rnorm(50,mean=2),rnorm(40,mean=2))
kf=kmeans(cbind(x,y),3,nstart=10)
plot(x,y,pch=15+kf$cluster,col=kf$cluster,cex=1.5)
l.cl=c(rep("A",30),rep("B",50),rep("C",40))
```



OUTLINE

Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

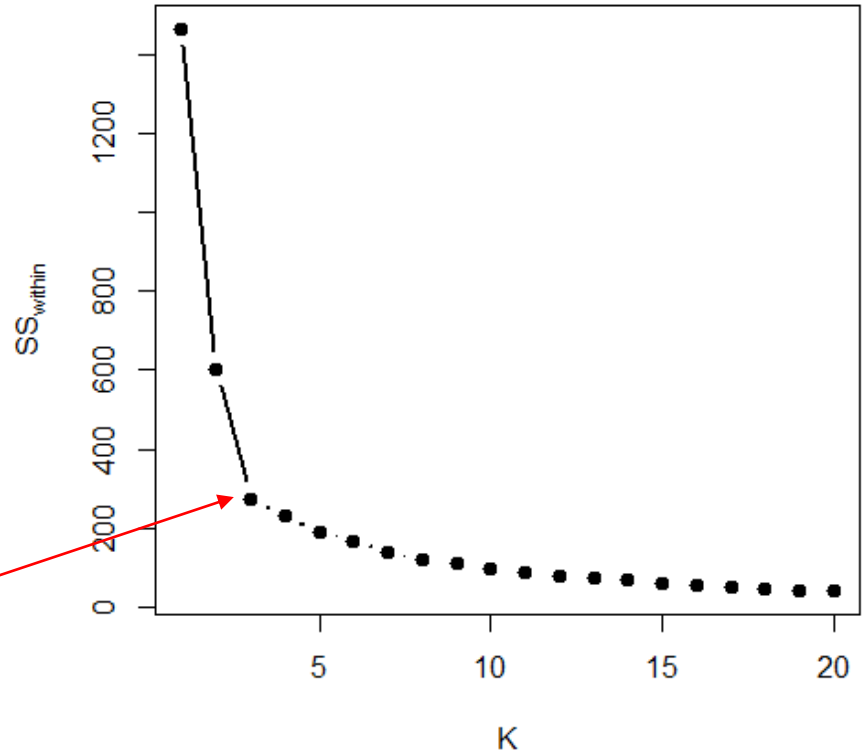
- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

WITHIN-CLUSTER VARIATION

- Within-cluster variation $W = \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - \bar{x}_k\|_2^2$ ← (sum of squares): $\boldsymbol{v} = (v_1, \ldots, v_p) \Rightarrow \|\boldsymbol{v}\|_2 = \sqrt{\sum_{j=1}^{p} v_p^2}$
  - Where the cluster centers are defined as $\bar{x}_k = \frac{1}{n_k} \sum_{C(i)=k} x_i$
  - Data points x_i are considered here to be multi-dimensional , i.e. each point = a vector of observations for p variables

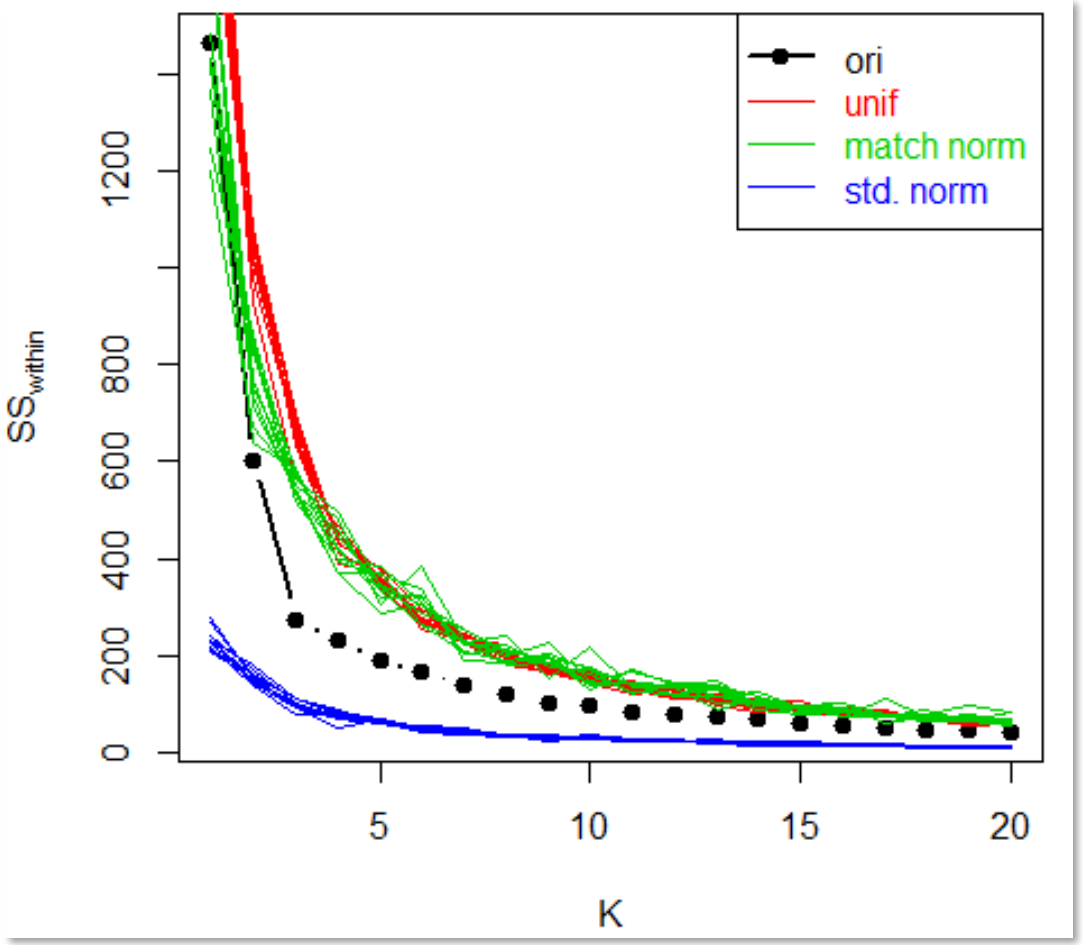
```
w=numeric(20)
for ( k in 1:20 ) {
  kf=kmeans(cbind(x,y),k,nstart=10)
  w[k] = kf$tot.withinss
}
plot(1:20,w,type="b",lwd=2,pch=19,xlab="K",
       ylab=expression(SS[within]))
```

elbow



WITHIN CLUSTER VARIATION ON RANDOM DATA

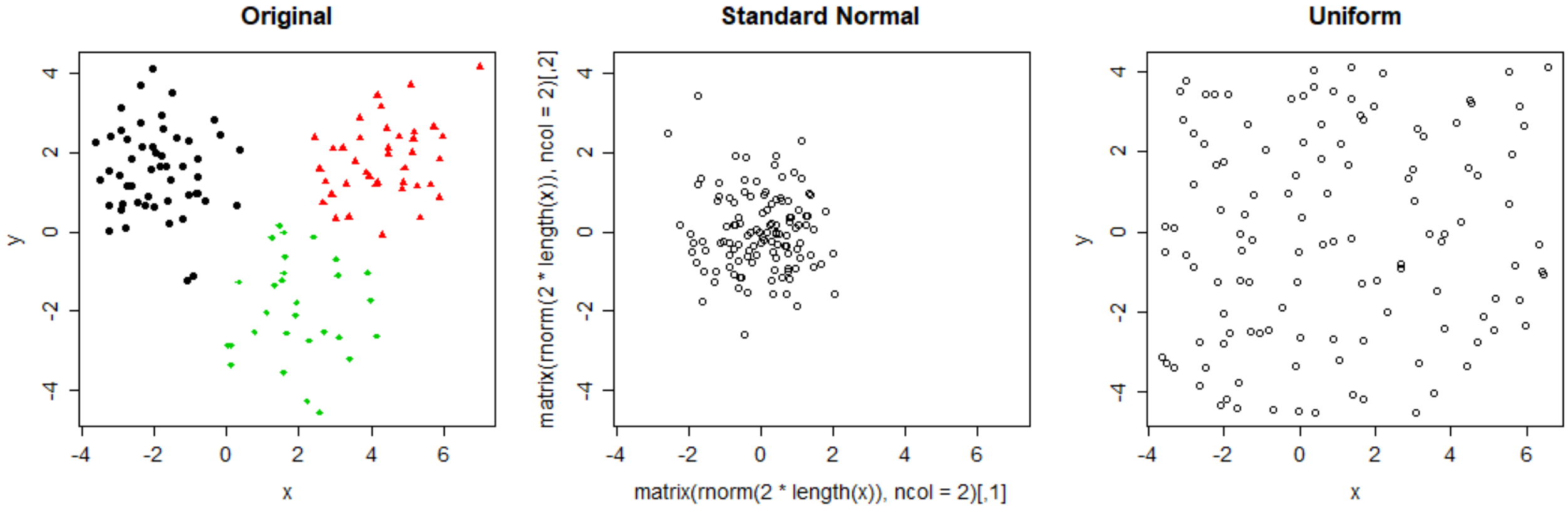
- How striking is the elbow?
- Striking compared to what?
- For example, results of the same algorithm (K-means clustering) applied to data that is *known* to have *no clusters*
  - Random sample from normal (standard? matching mean and standard deviation of x and y?) or uniform distribution
- Of course, distribution we sample has to be (at least roughly) representative of the data we work with
  - E.g. standard normal has much lower variance than our simulated dataset
- Random sample of uniform distribution bounded by range of every attribute is commonly used alternative
- In this case total within cluster variance decreases faster for original data as compared to random sample
  - Similarly to normal with matching mean and standard deviation



CODE TO GENERATE PREVIOUS PLOT

```
plot(1:20,w,type="b",lwd=2,pch=19,xlab="K",ylab=expression(SS[within]))
for ( j in 1:3 ) {
  for ( i in 1:10 ) {
    wrnd = numeric()
    for ( k in 1:20 ) {
      xyTmp <- apply(cbind(x,y),2,function(z)runif(length(z),min(z),max(z)))
      if ( j == 2 ) {
        xyTmp <- cbind(rnorm(length(x),mean(x),sd(x)),rnorm(length(y),mean(y),sd(y)))
      }
      if ( j == 3 ) {
        xyTmp <- matrix(rnorm(2*length(x)),ncol=2)
      }
      wrnd[k] = kmeans(xyTmp,k,nstart=10)$tot.withinss
    }
    points(wrnd,type="l",col=1+j)
  }
}
legend("topright", c("ori","unif","match norm","std. norm"), col=1:4, text.col=1:4,
pch=c(19,NA,NA,NA), lty=1, lwd=c(2,1,1,1))
```

ORIGINAL, NORMAL AND UNIFORM EXAMPLES



- Sample from bivariate standard normal is obviously much tighter than that in the "original" data
- Uniform sample is more representative of "the same data without clusters"

CODE TO GENERATE PREVIOUS PLOTS

```
old.par <- par(mfrow=c(1,3),ps=16)
plot(x,y,pch=19, col=kf$cluster, main="Original")
plot(matrix(rnorm(2*length(x)),ncol=2), xlim=range(x), ylim=range(y), main="Standard
Normal")
plot(apply(cbind(x,y),2,function(x)runif(length(x),min(x),max(x))),main="Uniform")
par(old.par)
```

- Adding a plot of random sample from normal distribution with mean and variance matching those of "x" and "y" is left as an exercise for the reader

OUTLINE

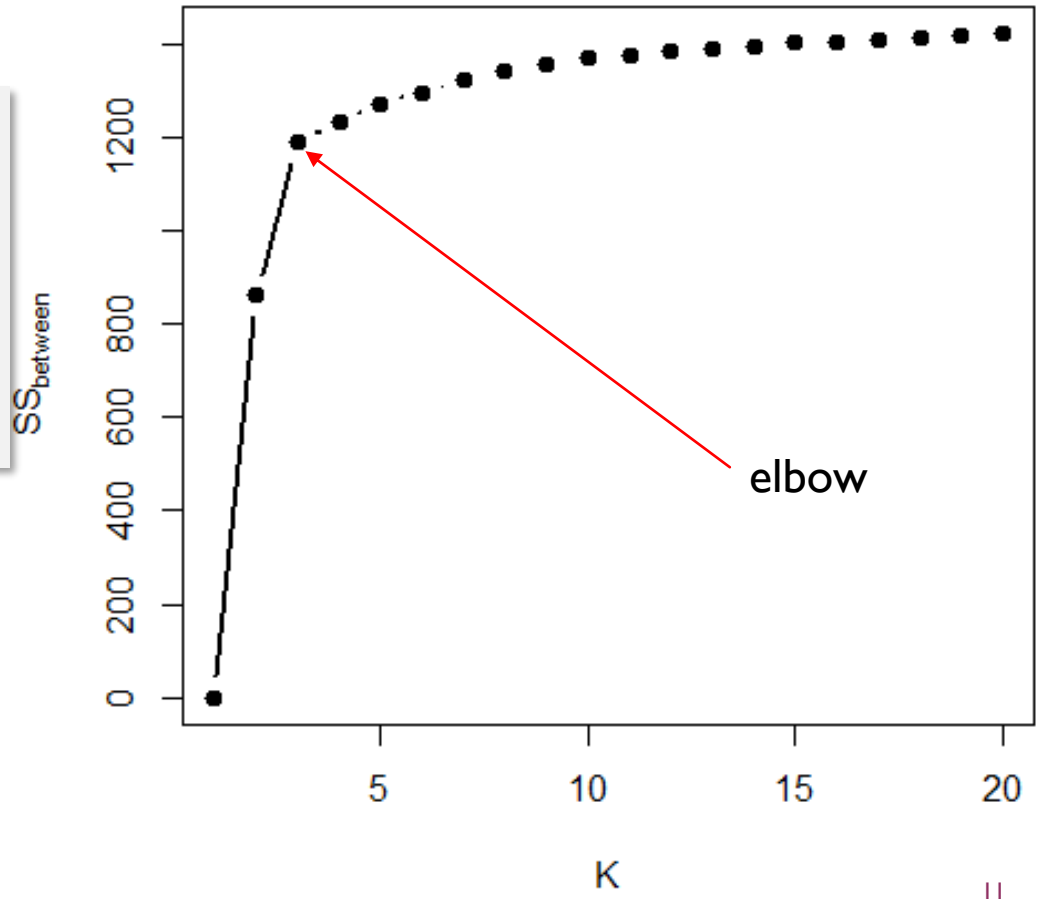
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

BETWEEN-CLUSTER VARIATION

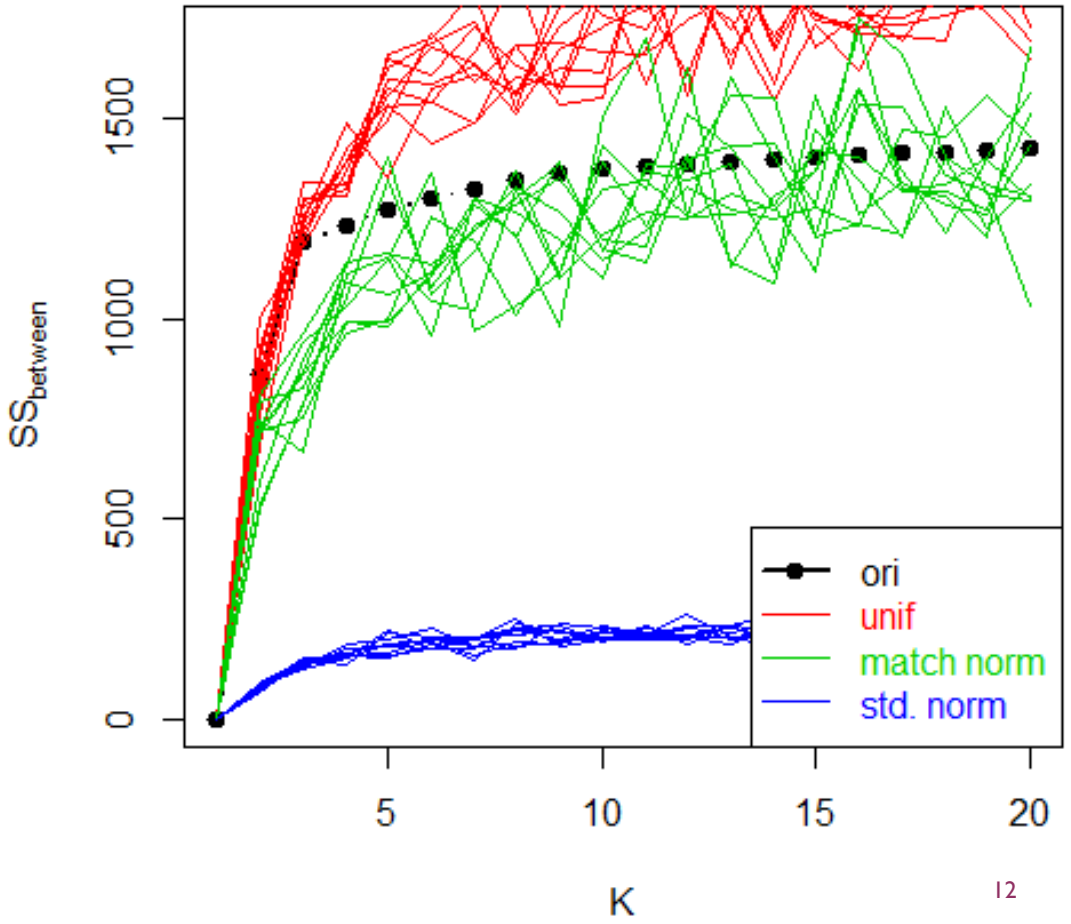
- Between-cluster variation, $B = \sum_{k=1}^{K} n_k \, \|\bar{x}_k - \bar{x}\|_2^2$
- Note that the *total sum of squares* $T = \sum_{i=1}^{N} \|x_i - \bar{x}\|_2^2 = B + W$

```
btw=numeric(20)
for ( k in 1:20 ) {
  kf=kmeans(cbind(x,y),k,nstart=100)
  btw[k] = kf$betweenss
}
plot(1:20,btw,type="b",lwd=2,pch=19,xlab="K",
    ylab=expression(SS[between]))
```



BETWEEN-CLUSTER VARIATION ON RANDOM DATA

- We know now that standard normal is a poor comparison for our simulated data with three clusters
- For samples from uniform distribution, between-cluster sum of squares also increases rapidly for the first few clusters
- And even to higher levels on average than what is observed for higher values of K
- K-means results for random samples from normal distribution with mean and variance matching those of the original data achieve lower between-cluster variation than for those drawn from uniform distribution



CODE FOR THE PREVIOUS PLOT

```
plot(1:20,btw,type="b",lwd=2,pch=19,xlab="K",ylab=expression(SS[between]),ylim=range(btw)*1.2)
for ( j in 1:3 ) {
  for ( i in 1:10 ) {
    btwrnd = numeric()
    for ( k in 1:20 ) {
      xyTmp <- apply(cbind(x,y),2,function(z)runif(length(z),min(z),max(z)))
      if ( j == 2 ) {
        xyTmp <- cbind(rnorm(length(x),mean(x),sd(x)),rnorm(length(y),mean(y),sd(y)))
      }
      if ( j == 3 ) {
        xyTmp <- matrix(rnorm(2*length(x)),ncol=2)
      }
      btwrnd[k] = kmeans(xyTmp,k,nstart=10)$betweenss
    }
    points(btwrnd,type="l",col=1+j)
  }
}
legend("bottomright", c("ori","unif","match norm","std. norm"), col=1:4, text.col=1:4,
pch=c(19,NA,NA,NA), lty=1, lwd=c(2,1,1,1))
```

OUTLINE

Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

CH INDEX

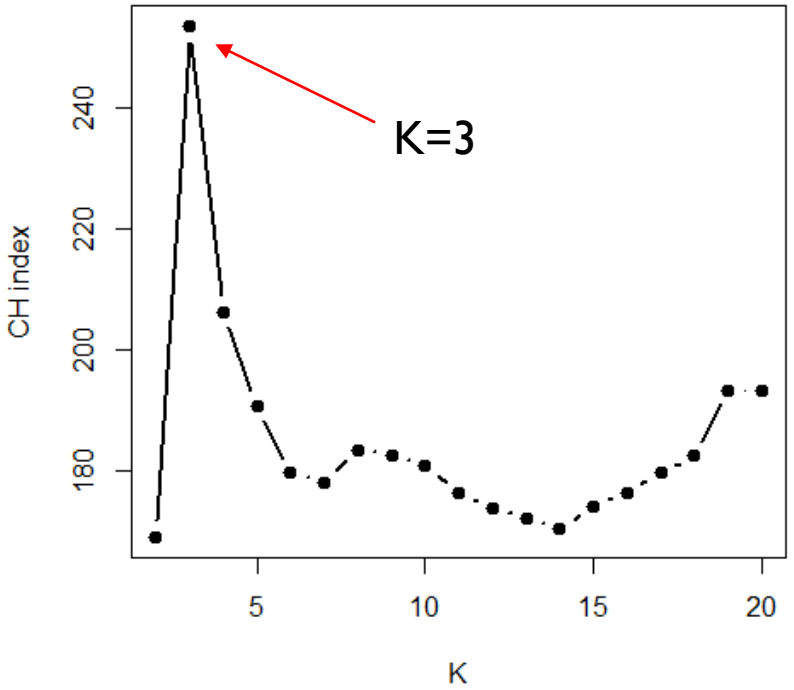
- Consider a score that attempts to strike a balance between within- and between-cluster sums of squares:

$$\mathrm{CH}(K) = \frac{B(K)/(K-1)}{W(K)/(n-K)}$$

compare this to F-statistic (e.g. for linear regression it's Eq(3.23)in ISLR): $F = \frac{(TSS-RSS)/p}{RSS/(n-p-1)}$

  - Called CH index [Calinski & Harabasz (1974),"A dendrite method for cluster analysis"]
  - Calculate in a (reasonable) range of K values, pick K that maximizes the index:

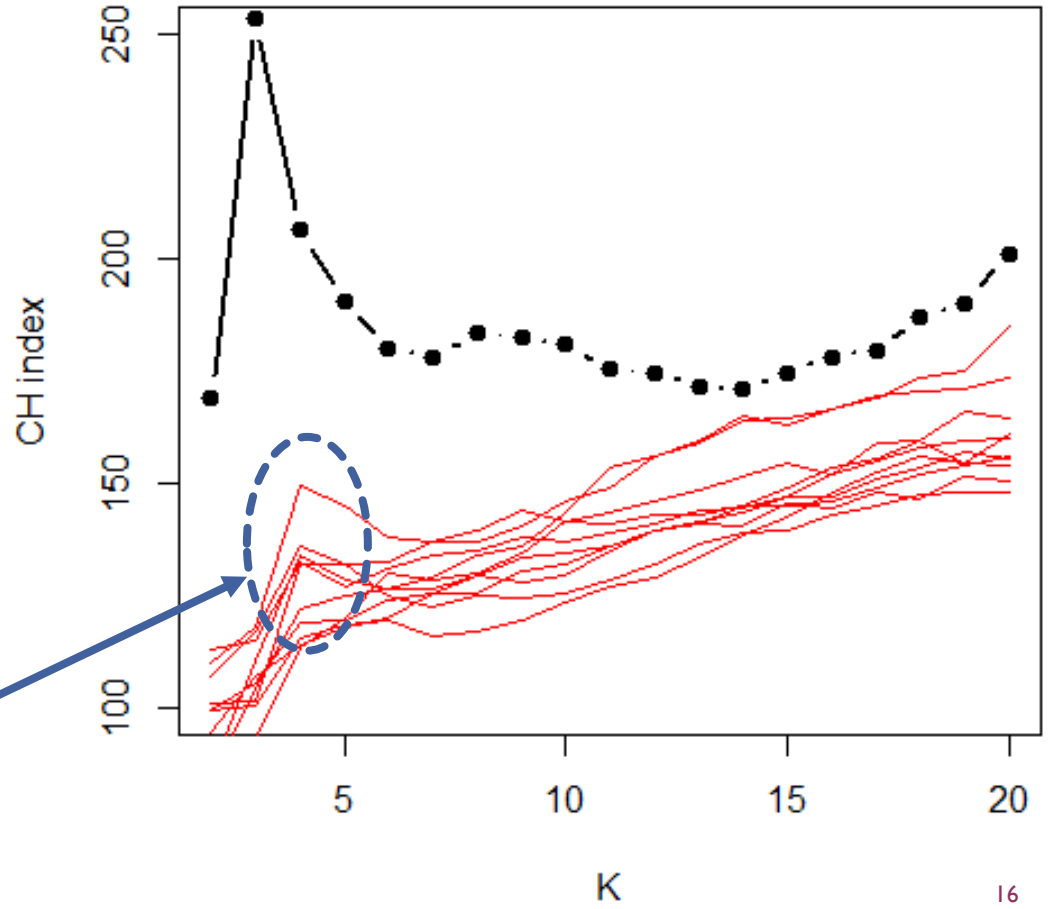
```
chidx=numeric(20)
for ( k in 2:20 ) {
  kf=kmeans(cbind(x,y),k,nstart=100)
  chidx[k] = (kf$betweenss/(k-1))/
         (kf$tot.withinss/(length(x)-k))
}
# undefined at k=1, so start plotting at 2:
plot(2:20,chidx[-1],type="b", lwd=2, pch=19, xlab="K",
       ylab="CH index")
```



CH-INDEX ON RANDOM DATA

- CH-index achieves much higher values on original as compared to random data from uniform distribution with the same ranges
- Tends to gradually increase with the increase of K on random data
- Value of K corresponding to the maximum in CH-index does not imply that this is the number of "real" clusters
  - Can achieve well defined local maxima on random data as well

Local maxima on random samples from uniform distribution



CODE FOR THE PLOT ABOVE

```
plot(2:20,chidx[-1],type="b", lwd=2,pch=19,xlab="K", ylab="CH index",ylim=c(100,250))
for ( i in 1:10 ) {
  chrnd = numeric()
  xyTmp = apply(cbind(x,y),2,function(x)runif(length(x),min(x),max(x)))
  for ( k in 2:20 ) {
    krnd = kmeans(xyTmp,k,nstart=100)
    chrnd[k] = (krnd$betweenss/(k-1))/(krnd$tot.withinss/(length(x)-k))
  }
  points(chrnd,type="l",col="red")
}
```

- Notice that each random sample from uniform distribution was clustered by K-means in 2 through 20 clusters, as opposed to drawing new sample for each value of K as before

OUTLINE

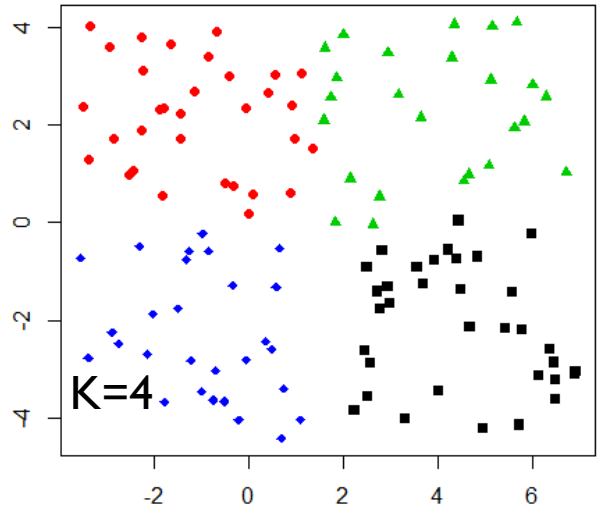
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

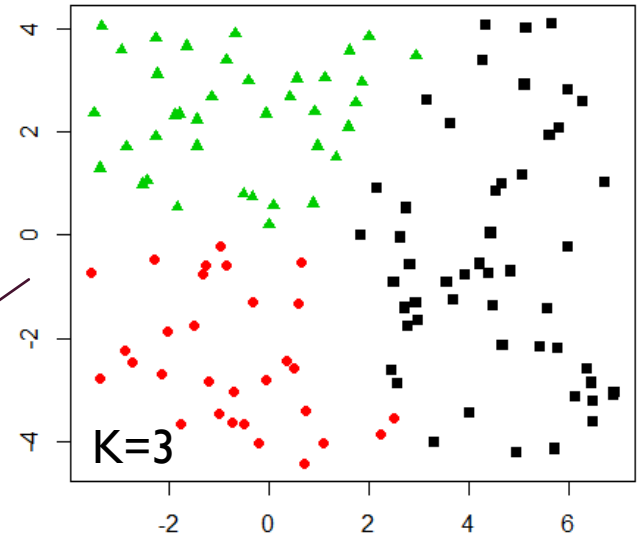
- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

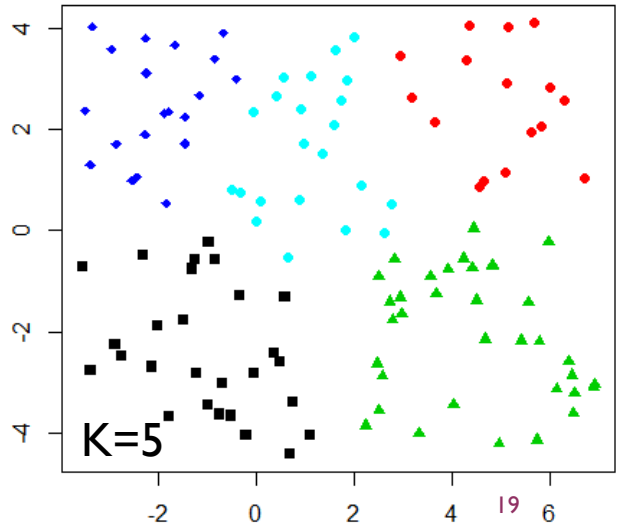
GAP STATISTICS

- Idea: in order to quantify the "significance", we need to compare against some "null distribution"!
  - What is "null"?
  - No clusters = the points are distributed randomly and uniformly over an encapsulating box
  - Let's try it. Are those real clusters? (remember, this is how *random* points "cluster"!)

```
m.new=apply(cbind(x,y),2,function(x) {
     runif(length(x),min=min(x),max=max(x))
})
m.new[1:3,]
                x          y
[1,]  1.1030387 -4.034728
[2,] -2.8717777 -2.246960
[3,] -0.1973551 -4.041795
for ( k in 2:5 ) {
  kf=kmeans(m.new,k,nstart=100)
  plot(m.new,pch=14+kf$cluster,col=kf$cluster,main=paste("K =",k))
}
```







GAP STATISTICS: CONTINUED

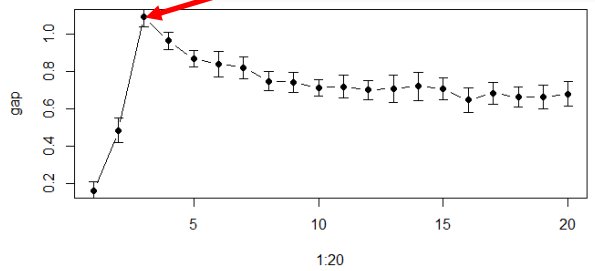
- Let us use the ratio of the "null" within-cluster sum of squares $W_{unif}(K)$ (computed from uniformly scattered points) to the actual within-cluster SS, W(K) (computed on real data) as the measure of "goodness of clustering"
  - With no cluster structure we expect the clusters to be "bad" and wide ➔ large W_{unif} (but still decreases with K!)
  - If there are true clusters in the data, we expect them to be "tighter" than those "seen" in uniform random data, so $W_{unif}(K)/W(K)$ is large.
  - Since we are computing a quantity (W_{unif}) from random data, it is a good idea to perform a few resamplings and take the average, so below we consider W_{unif} to be such average, with standard error s(K)
  - It is convenient to take log: $\text{Gap}(K) = \log W_{unif}(K) - \log W(K)$
  - We define the optimal K as the value where the gap reaches its first maximum *up to the sampling error*:
    - $K_{\text{opt}} = \min\{K \in \{1 \dots K_{\max}\}: \text{Gap}(K) \geq \text{Gap}(K+1) - s(K+1)\}$

GAP STATISTICS: CODE

- Let's develop the code for gap statistics:

```
# takes matrix of observations of p variables (points in
# p-dimensional space: a row=point), generates the same
# number of p-dimensional points randomly and uniformly
# scattered in the bounding box:
lw.unif=function(m,K,N=20,...) {
  w=numeric(N)
  for ( i in 1:N ) {
    m.new=apply(m,2,function(x) {
                  runif(length(x),min=min(x),max=max(x))
                })
    # ellipsis allows caller
    # to pass through any additional arguments:
    w[i] = kmeans(m.new,K,...)$tot.withinss
  }
  return( list(LW=mean(log(w)),SE=sd(log(w))*sqrt(1+1/N)) )
}
```

```
# computes the gap and the LogW_unif SE
# for different K:
gap = numeric(20)
se = numeric(20)
for ( k in 1:20 ) {
  kf=kmeans(cbind(x,y),k,nstart=100)
  sim = lw.unif(cbind(x,y),k,nstart=100)
  gap[k] = sim$LW - log(kf$tot.withinss)
  se[k] = sim$SE
}
plot(1:20, gap, pch=19, type="b")
arrows(1:20, gap-se, 1:20, gap+se,
       length=0.05, angle=90, code=3)
# find optimal K:
min(which(gap[-length(gap)]>=(gap-se)[-1]))
[1] 3
```



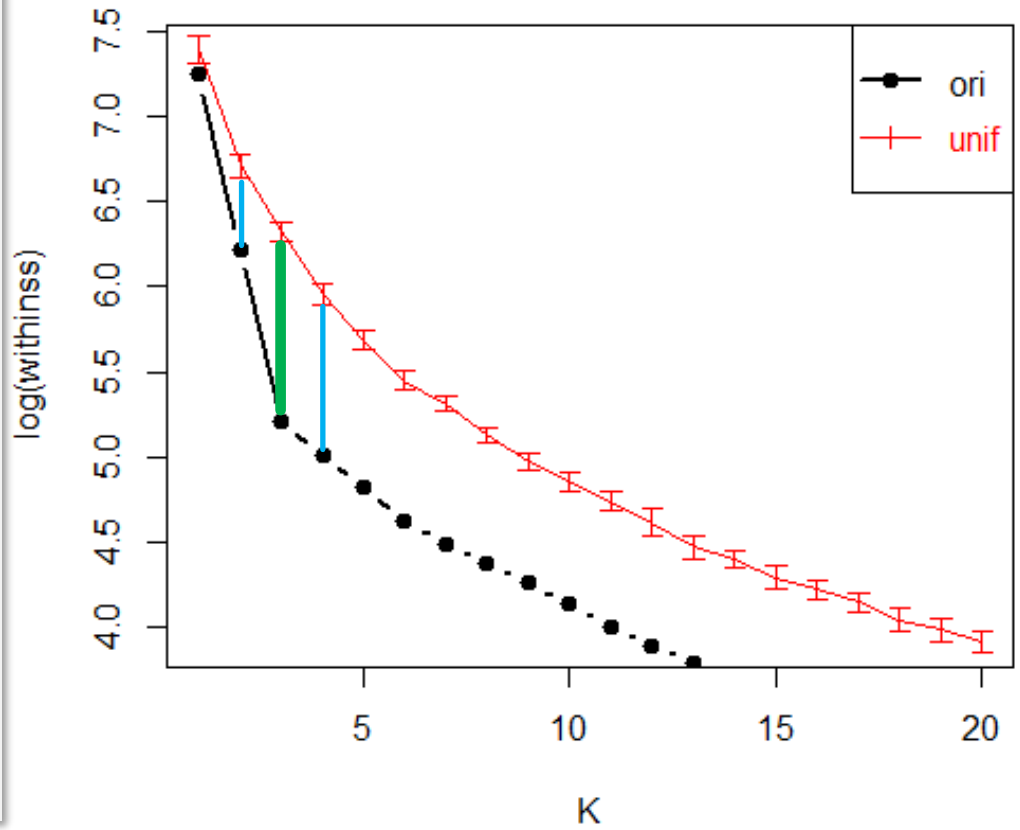
Note: gap statistics is implemented in R packages `cluster`, `lga` and `SAGx`

Try to use `nstart=1` above and understand the effect of it

ORIGINAL AND NULL WITHIN-CLUSTER SUM OF SQUARES

```
se = numeric(20)
rnd.logw = numeric(20)
ori.logw = numeric(20)
for ( k in 1:20 ) {
  kf = kmeans(cbind(x,y),k,nstart=100)
  sim = lw.unif(cbind(x,y),k,nstart=100)
  rnd.logw[k] = sim$LW
  ori.logw[k] = log(kf$tot.withinss)
  se[k] = sim$SE
}
plot(1:20, rnd.logw, type="l", lwd=1, col="red",
     xlab="K", ylab="log(withinss)")
arrows(1:20, rnd.logw-se, 1:20, rnd.logw+se,
       length=0.05, angle=90, code=3, col=2)
points(ori.logw, type="b", pch=19, lwd=2)
legend("topright",c("ori","unif"),text.col=1:2,
       col=1:2,pch=c(19,3),lty=1,lwd=c(2,1))
```

Try with nstart=1:



- According to gap statistics, the "best" number of clusters is where the gap between original and null is the largest

OUTLINE

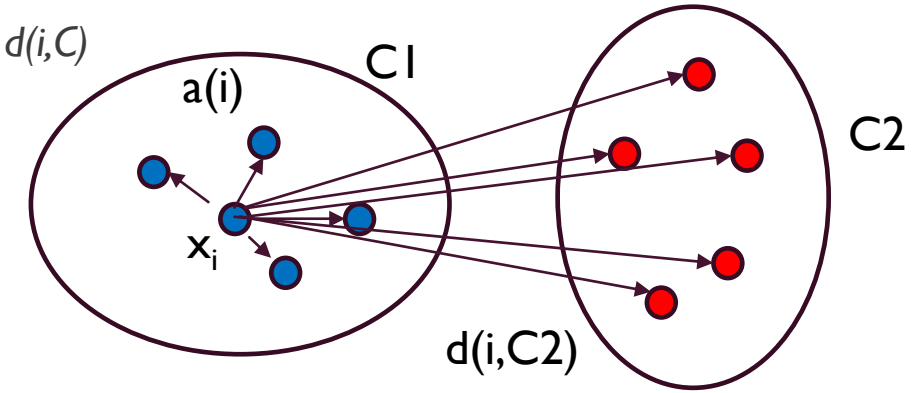
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

SILHOUETTE

- Consider a point x_i. *Silhouette* is defined for each individual point in the following way:
  - Let a(i) be the average dissimilarity (i.e. distance) between x_i and all other points of the cluster *C(i)* to which x_i belongs
  - If x_i is in its own cluster (with no other points in it), then the silhouette is *s(i)=0*; otherwise
  - For each other cluster *C* (i,e. excluding *C(i)*) let *d(i,C)* be the average distance from x_i to all observations in C
  - Define the "distance to the closest cluster" as $b(i)=\min_C d(i,C)$
  - Define silhouette as

$$s(i) = \frac{b(i)-a(i)}{\max(a(i),b(i))}$$

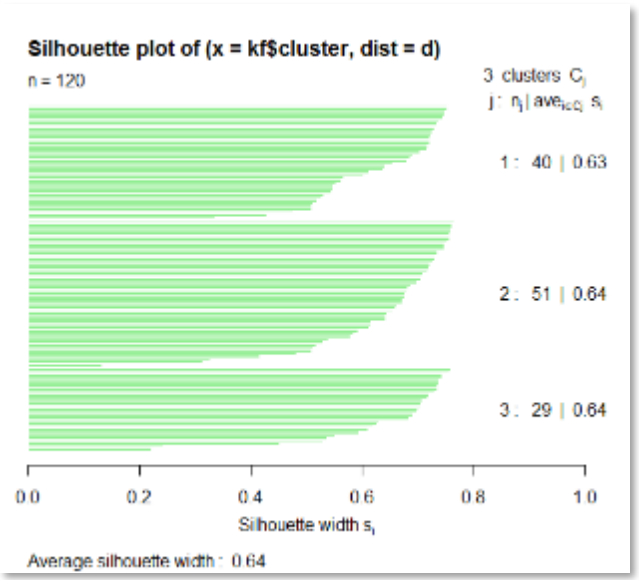


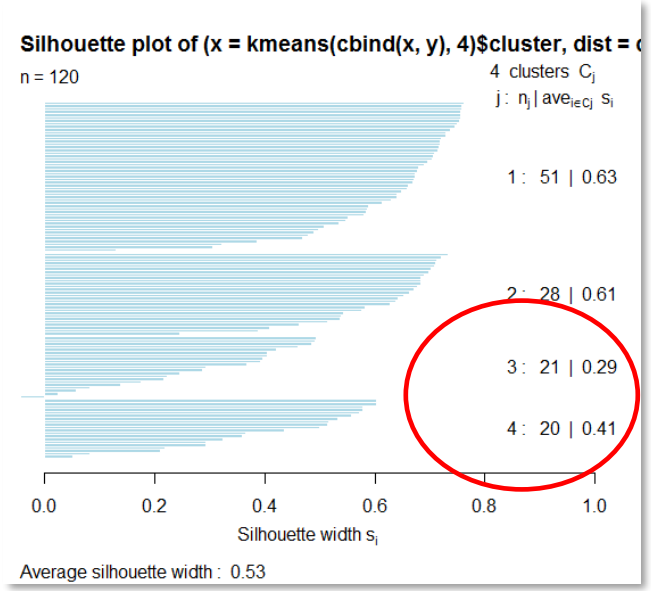
- Properties:
  - Normalized! (always in the range [-1,1])
  - If "average width" a(i) is small and the distance to the closest cluster b(i) is large, then s(i) → 1 (i.e. 1 is good!)
  - If s(i) is small, (i.e. much less than 1), then the point is "between clusters" (or in its own cluster): the average distance to the other members of the same cluster is not much smaller in this case than the average distance to other cluster(s)
  - If s(i) is negative (a(i) > b(i)), the point is seriously misclassified: it's closer, on average, to another cluster than to other members of the cluster it was assigned to

SILHOUETTE: R CODE

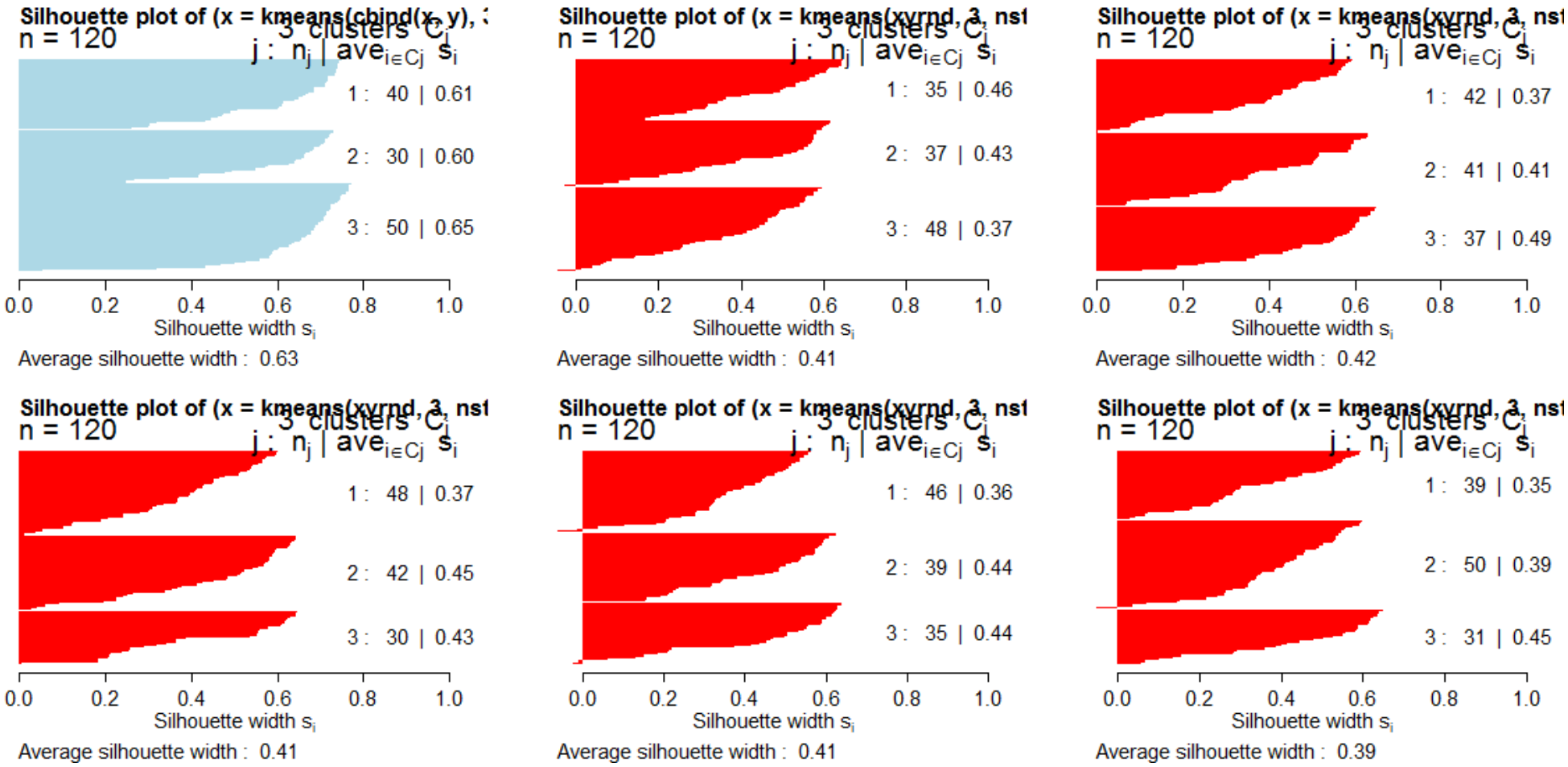
- R has a function for calculating the silhouette (in package `cluster`)
  - Watch for average silhouette (cf. all average silhouettes > 0.6 at K=3 (green) and two much weaker clusters at K=4 (blue))

```
kf=kmeans(cbind(x,y),3)
d=dist(cbind(x,y))
sl=silhouette(kf$cluster,d)
summary(sl)
Silhouette of 120 units in 3 clusters ...
 Cluster sizes and average silhouette widths:
       40        51        29
0.6316847 0.6408889 0.6410914
Individual silhouette widths:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1327  0.5657  0.6812  0.6379  0.7287  0.7635
plot(sl,col="lightgreen")
plot(silhouette(kmeans(cbind(x,y),4)$cluster,d),
     col='lightblue')
```





SILHOUETTES OF RANDOM DATA



- Wider silhouette for the original data as compared to random uniform

CODE FOR THE PREVIOUS PLOTS

```
old.par = par(mfrow=c(2,3),ps=16)
d=dist(cbind(x,y))
plot(silhouette(kmeans(cbind(x,y),3,nstart=10)$cluster,d),col='lightblue')
for ( iRnd in 1:5 ) {
    xyrnd = apply(cbind(x,y),2,function(x)runif(length(x),min(x),max(x)))
    drnd = dist(xyrnd)
    plot(silhouette(kmeans(xyrnd,3,nstart=10)$cluster,drnd),col='red')
}
par(old.par)
```

OUTLINE

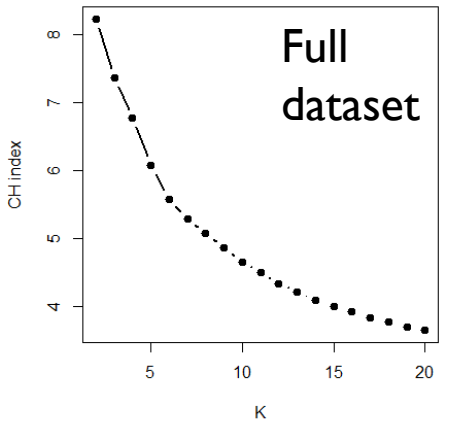
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

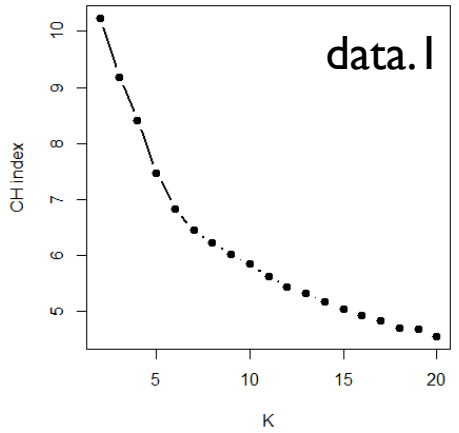
- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

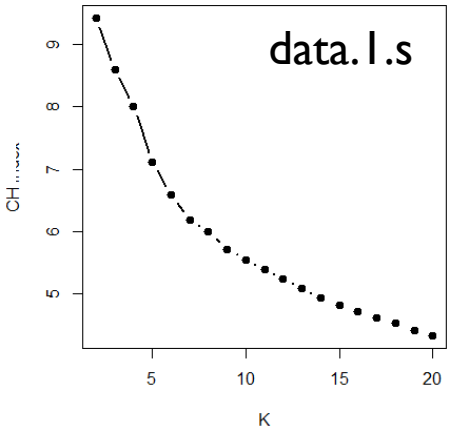
REAL DATA EXAMPLE

- Let us consider again the NCI60 dataset introduced last week
  - 64 tumor samples, expression levels of 6380 genes are measured in each sample

```
> library(ISLR)
> data(NCI60)
> rownames(NCI60$data)=NCI60$labs
# why >1, and not >2 or >0.5 ?
> data.1 = NCI60$data[,apply(NCI60$data,2,sd)>1]
> data.1.s = scale(data.1)
# try k-means first:
> w=numeric(20)
> for ( k in 2:20 ) {
  # or full, unfiltered NCI60$data; or data.1.s:
  kf=kmeans(data.1,k,nstart=100)
  w[k] = (kf$betweenss/    # CH index
    (k-1))/(kf$tot.withinss/(length(kf$cluster)-k))
  }
> plot(2:20,w[-1],type="b",lwd=2,pch=19,xlab="K",
      ylab="CH index")
```







! Gap statistics is similarly unconvincing (you can try it!)!

CLUSTER STABILITY

- How stable are the clusters as we wiggle parameters/rescale the data etc?
- Build a "contingency table" between the clustering results (i.e. count how many observations are in cluster i in one run and in cluster j in the other); if the two clustering runs are perfectly consistent, do we expect that all observations in, say, cluster 2 of one run to be also in cluster 2 of the other (i.e. contingency table perfectly diagonal)? Not so fast…
  - Caveat: cluster IDs can be different across different clustering runs, need to rearrange the table
  - Problem: given matrix A, find permutation p: [1…n] →[1…n] that maximizes the trace of the rearranged matrix, $\sum_i A[i, p(i)]$
  - "Hungarian algorithm" provides the solution, but it is not a simple one… But there is a library!

```
matrix.sort <- function(m) {
    require(clue)
    p = solve_LSAP(m, maximum=T)  # find the permutation…
    m[, p] # and apply it!
}
```

7	1	2	7	2	1
2	0	12	2	12	0
0	9	3	0	3	9

```
cmp.shortcut = function(K,...) {
   matrix.sort(table(
      FULL=kmeans(NCI60$data,K,...)$cluster,
      SCALED.SUBSET=
         kmeans(data.1.s,K,...)$cluster))
}
cmp.shortcut(4,nstart=100)
```

```
     SCALED.SUBSET
FULL  1  2  3  4
   1  8  0  0  0
   4  0 30  0  0
   3  0  0  9  0
   2  0  0  0 17
```

```
cmp.shortcut(5,nstart=100)
 SCALED.SUBSET
FULL  1  2  3  4  5
   2  4  0  0  0  0
   3  0  8  0  0  0
   1  2  0 13  0  5
   5  0  0  0  9  0
   4  0  0  0  0 23
```

Try also with `nstart=1`

IS HIERARCHICAL BETTER?

- Functions for within/between cluster variance:

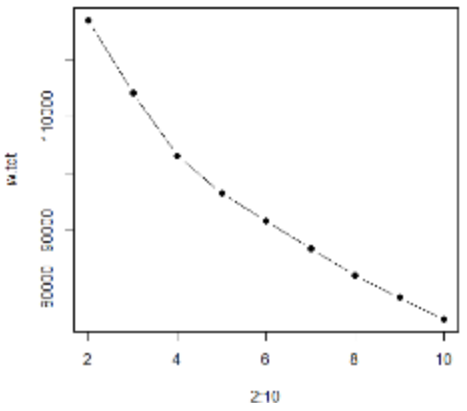
```
within=function(d,clust) {
  w=numeric(length(unique(clust)))
  for ( i in sort(unique(clust)) ) {
    members = d[clust==i,,drop=F]
    centroid = apply(members,2,mean)
    members.diff = sweep(members,2,centroid)
    w[i] = sum(members.diff^2)
  }
  return(w)
}
```

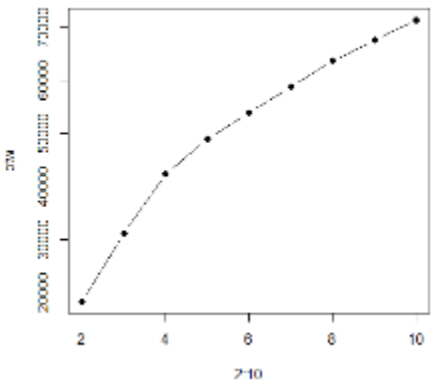
```
between=function(d,clust) {
  b=0
  total.mean = apply(d,2,mean)
  for ( i in sort(unique(clust)) ) {
    members = d[clust==i,,drop=F]
    centroid = apply(members,2,mean)
    b = b + nrow(members)*
        sum( (centroid-total.mean)^2 )
  }
  return(b)
}
```

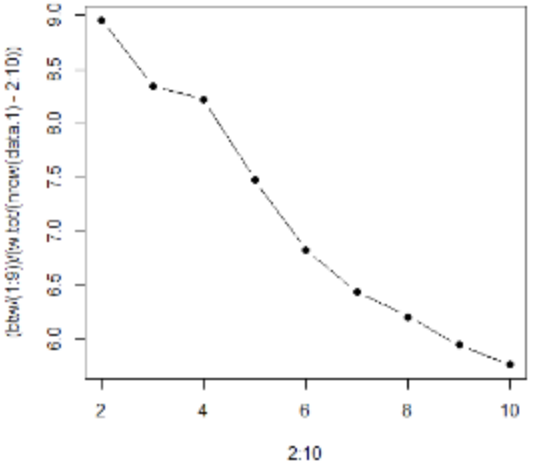
EXAMINING DIFFERENT NUMBERS OF HIERARCHICAL CLUSTERS

- The clear cluster structure is missing from the hierarchical clustering result as well…

```
data.1=NCI60$data[,apply(NCI60$data,2,sd)>1]
dd.1=dist(data.1)
hw.1=hclust(dd.1,method="ward.D2")
w.tot=numeric(9)
btw=numeric(9)
for ( k in 2:10 ) {
  clust = cutree(hw.1,k=k)
  w = within(data.1,clust)
  w.tot[k-1]=sum(w)
  btw[k-1] = between(data.1,clust)
}
plot(2:10,w.tot,pch=19,type="b")
plot(2:10,btw,pch=19,type="b")
plot(2:10,(btw/(1:9))/
   (w.tot/(nrow(data.1)-2:10)),pch=19,type="b")
```







OUTLINE

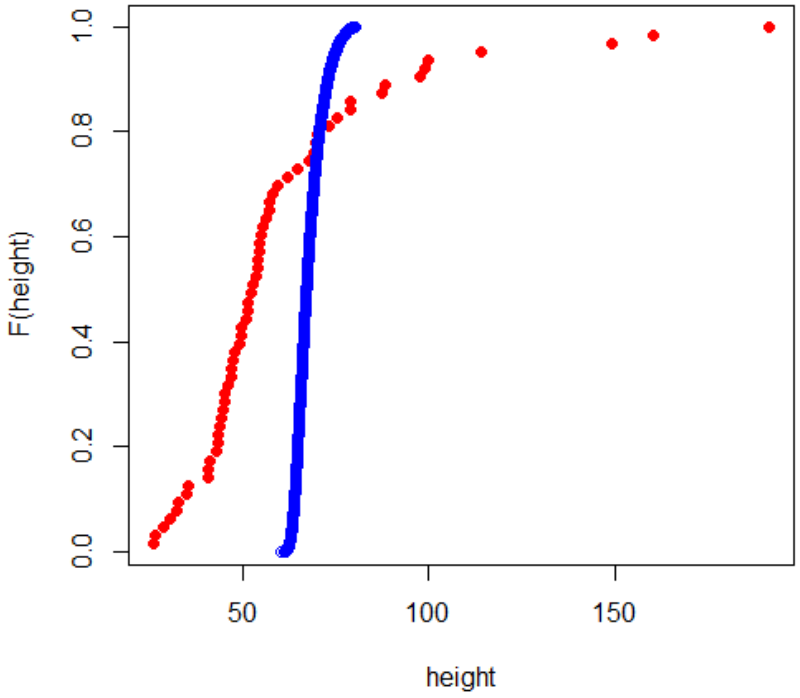
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

BRUTE FORCE RANDOMIZATION

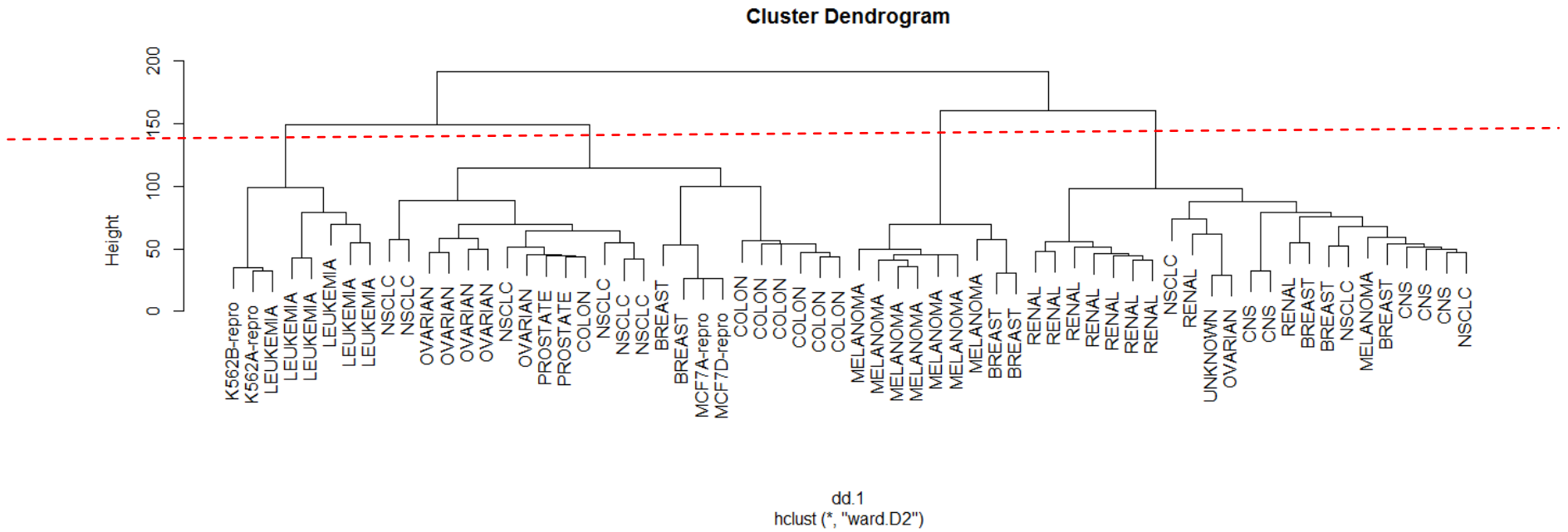
- Let us randomize expression levels across each sample and repeat clustering
- All the "similarities" between samples we are now observing are due to random chance
- What is the distribution of the distances between samples/clusters and how does it compare to the original data?

```
ori.heights = hw.1$height
rnd.heights = numeric()
for ( i.sim in 1:100 ) {
   data.rnd <-apply(data.1,2,sample)
   hw.rnd=hclust(dist(data.rnd),method="ward.D2")
   rnd.heights <- c(rnd.heights,hw.rnd$height)
}
plot(ori.heights,rank(ori.heights)/length(ori.heights),
     col="red",xlab="height",ylab="F(height)",pch=19)
points(rnd.heights,rank(rnd.heights)/length(rnd.heights),
     col="blue")
```

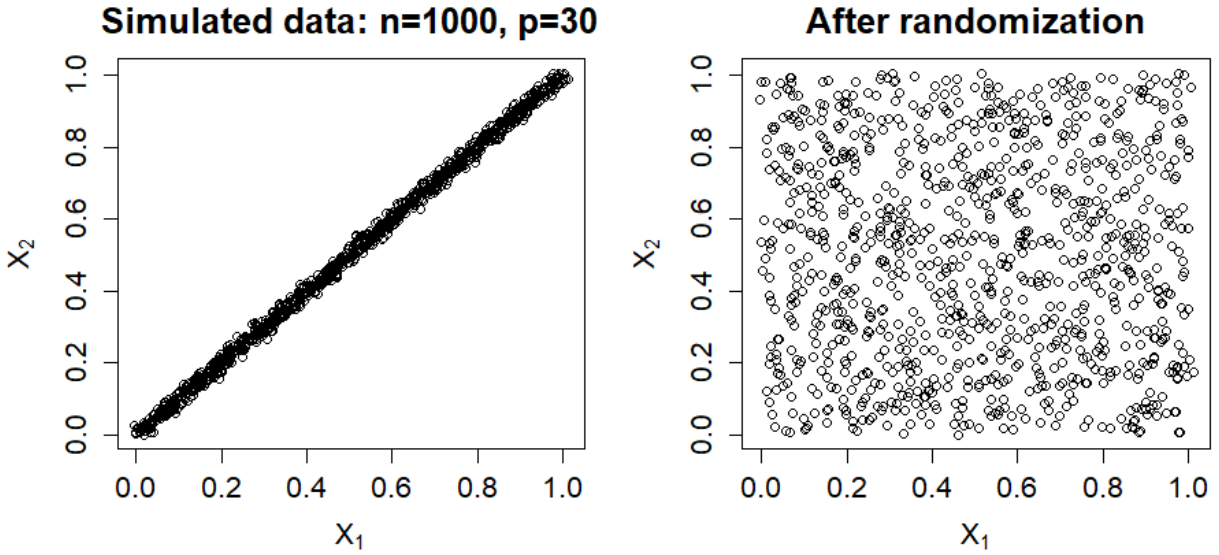


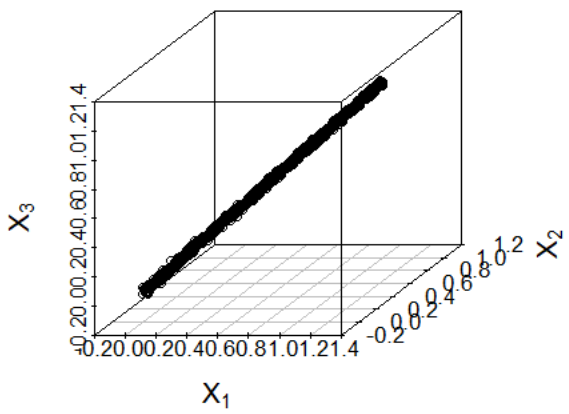
NCI60 CLUSTERS

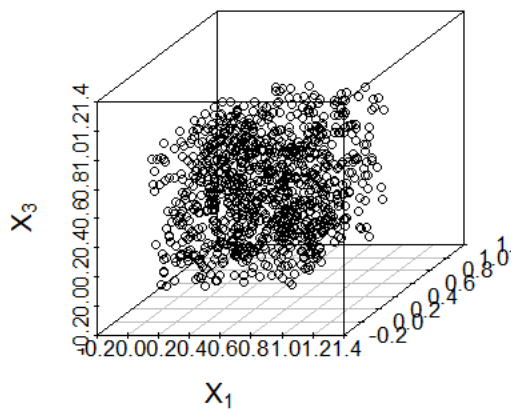
- About 4 top level clusters *might* be real
  - Brute force resampling can be too optimistic!
- Many other methods exist, including bootstrap of variables (genes in our example) or multilevel bootstrap
  - See package `pvclust`

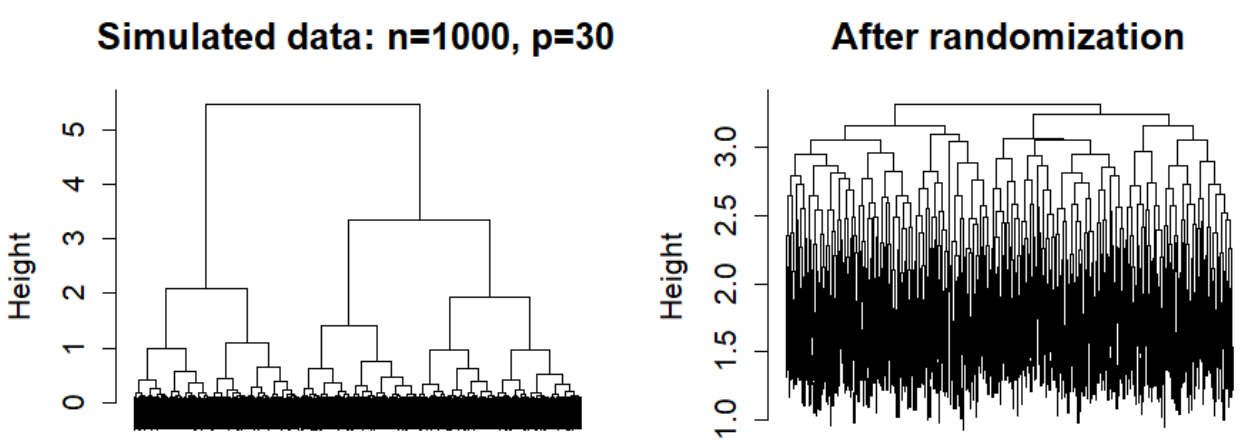


RE: EXCESSIVE OPTIMISM OF BRUTE FORCE RANDOMIZATION

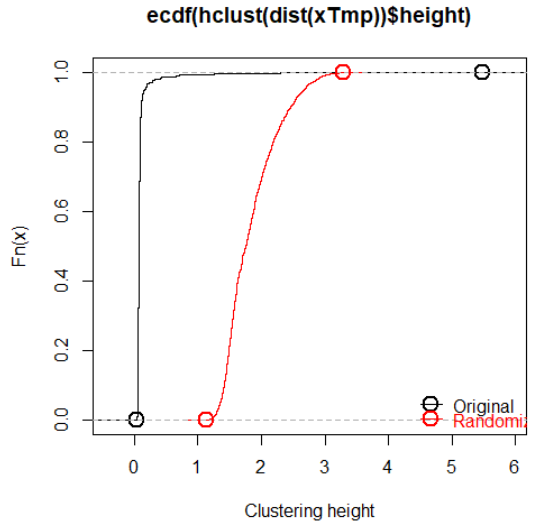








- In the presence of strong correlation, clustering results by brute force randomization will be remarkably different from those on the original data
- Even when no true clusters are present in the original data

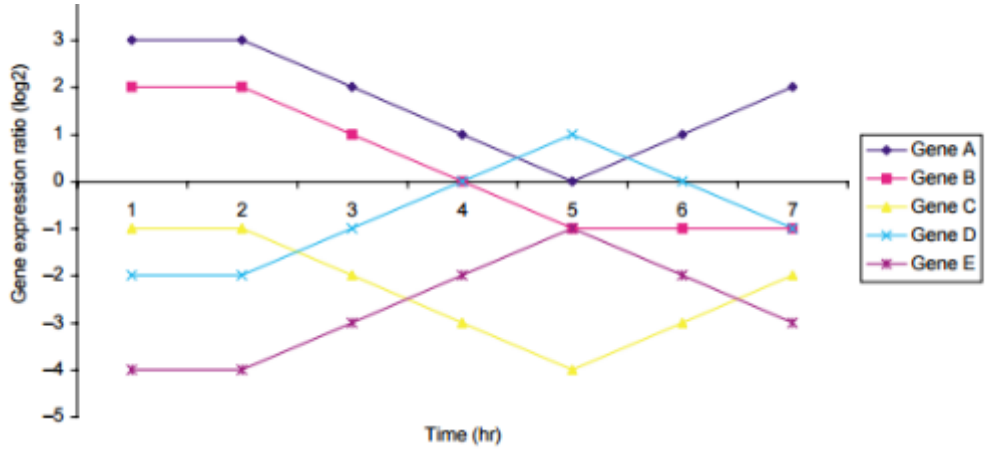


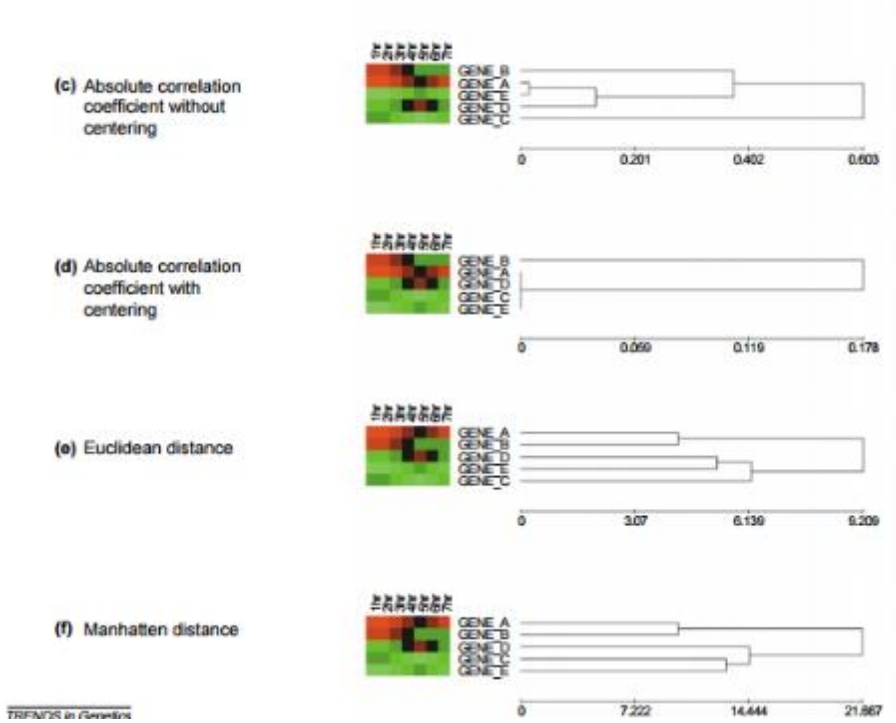
CODE FOR THE PREVIOUS PLOTS

```
n <- 1000
p <- 30
xTmp <- matrix(runif(n),nrow=n,ncol=p)+rnorm(n*p,sd=0.01)
xTmpRnd <- apply(xTmp,2,sample)
old.par <- par(mfrow=c(1,2),ps=16,mar=c(5,5,3,1))
plot(xTmp,xlab=expression(X[1]),ylab=expression(X[2]),main="Simulated data: n=1000, p=30")
plot(xTmpRnd,xlab=expression(X[1]),ylab=expression(X[2]),main="After randomization")
par(old.par)
old.par <- par(mfrow=c(1,2),ps=16)
scatterplot3d(xTmp[,1:3],xlab=expression(X[1]),ylab=expression(X[2]),zlab=expression(X[3]),main="Simulated
data: n=1000, p=30")
scatterplot3d(xTmpRnd[,1:3],xlab=expression(X[1]),ylab=expression(X[2]),zlab=expression(X[3]),main="After
randomization")
par(old.par)
old.par <- par(mfrow=c(1,2),ps=16)
plot(hclust(dist(xTmp)),labels=FALSE,main="Simulated data: n=1000, p=30")
plot(hclust(dist(xTmpRnd)),labels=FALSE,main="After randomization")
par(old.par)
plot(ecdf(hclust(dist(xTmp))$height),xlab="Clustering height")
points(range(hclust(dist(xTmp))$height),y=c(0,1),cex=2,lwd=2)
plot(ecdf(hclust(dist(xTmpRnd))$height),add=TRUE,col=2)
points(range(hclust(dist(xTmpRnd))$height),y=c(0,1),cex=2,lwd=2,col=2)
legend("bottomright",c("Simulated","Randomized"),col=1:2,text.col=1:2,bty="n",pch=1,pt.cex=2,pt.lwd=2,lty=1)
```

NOTE ON DISTANCES

- From: Leung, Cavalieri, Fundamentals of cDNA microarray data analysis, Trends in Genetics, 19:2003, p.649





OUTLINE

Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

"AIRWAY" DATASET

- Another gene expression experiment
  - Can be downloaded as library/dataset from Bioconductor (another large repository of R packages, biology-oriented)
  - 4 cell lines (lung airway, asthma), 2 conditions: untreated (control) and treated with a drug

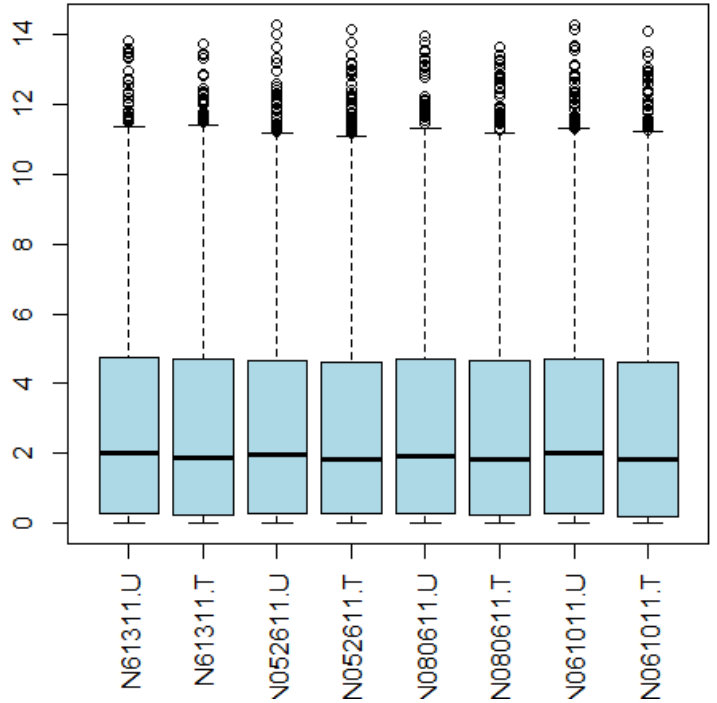
```
source("http://bioconductor.org/biocLite.R")
biocLite("airway")
library(airway)
data(airway)
airway
class: SummarizedExperiment
dim: 64102 8
exptData(1): ''
assays(1): counts
rownames(64102): ENSG00000000003 ENSG00000000005 ...
rowData metadata column names(0):
colnames(8): SRR1039508 SRR1039509 ...
colData names(9): SampleName cell ...
```

```
> colData(airway)[1:3,1:4]
DataFrame with 3 rows and 4 columns
           SampleName     cell      dex    albut
             <factor> <factor> <factor> <factor>
SRR1039508 GSM1275862   N61311    untrt    untrt
SRR1039509 GSM1275863   N61311      trt    untrt
SRR1039512 GSM1275866  N052611    untrt    untrt
> assay(airway)[1:3,1:3]
                SRR1039508 SRR1039509 SRR1039512
ENSG00000000003        679        448        873
ENSG00000000005          0          0          0
ENSG00000000419        467        515        621
```

PREPARE DATA

- Many genes are in fact not measured (count = 0 or extremely low) : remove unmeasured genes
- If we "count longer", we will count proportionally more events per gene. Need to normalize the counts by the total number of counts in each sample
  - Better methods do exist!
- Distribution of expression levels is log-normal. Take the log.

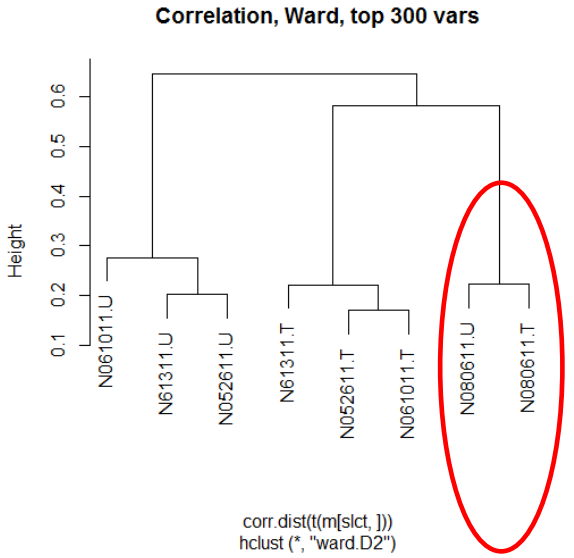
```
> m = assay(airway)
# use cell line/treatment to rename columns (samples):
> colnames(m)= paste(colData(airway)$cell,
        ifelse(colData(airway)$dex=="untrt","U","T"),sep=".")
> labels=colnames(m)
> m = m[ apply(m,1,sum) > 5, ]  # cuts m from 64K rows (genes) to 24K
> total.counts=apply(m,2,sum)
> m = sweep(m,2,total.counts/1000000,FUN="/") # normalize per MILLION cnts
> m=log2(m+1) # regularized log
> boxplot(m,col="lightblue",las=3)
```

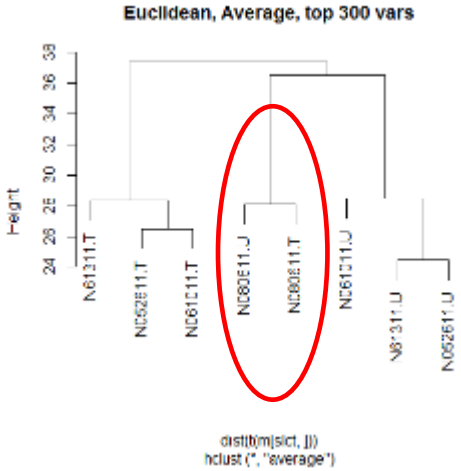


CLUSTER SAMPLES

- Let us consider the genes ("variables") with largest variance across the "experimental observations" (samples) and cluster samples on those genes
  - Not shown: we could cluster on full set of variables, the results would be less stable but still strange

```
# correlation distance (Note the t(): we are going to use the
# same contract as dist() which calculates pairwise
# distances between rows (observations)
corr.dist=function(x) { as.dist(1-cor(t(x))) }
vars=apply(m,1,var)
slct=order(vars,decreasing=T)[1:300]
plot(hclust(corr.dist(t(m[slct,])),method="ward.D2"),
   main="Correlation, Ward, top 300 vars")
plot(hclust(dist(t(m[slct,])),method="average"),
        main="Euclidean, Average, top 300 vars")
```





OUTLINE

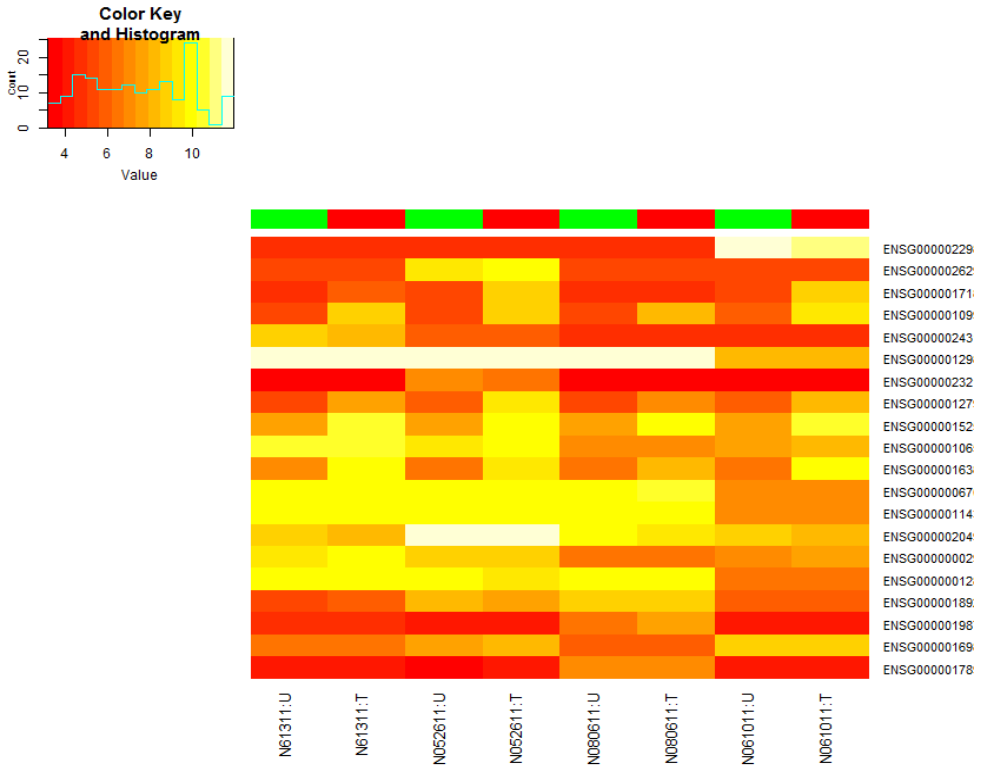
Unsupervised learning, continued: how many clusters? what are the measures of "goodness of clustering"?

- Within-cluster variation
- Between-cluster variation
- CH-index
- Gap statistics
- Silhouette
- NCI60 example
  - Brute force permutation
- "Airway" dataset
  - Heatmap

INTRODUCING HEATMAP

- Simplest heatmap
  - Order of rows/columns the same as in the data table (must use Rowv=F and Colv=F); values are represented with color gradient (see the legend)
  - Note how we set color code for untreated/treated (green/red) on top (ColSideColors)

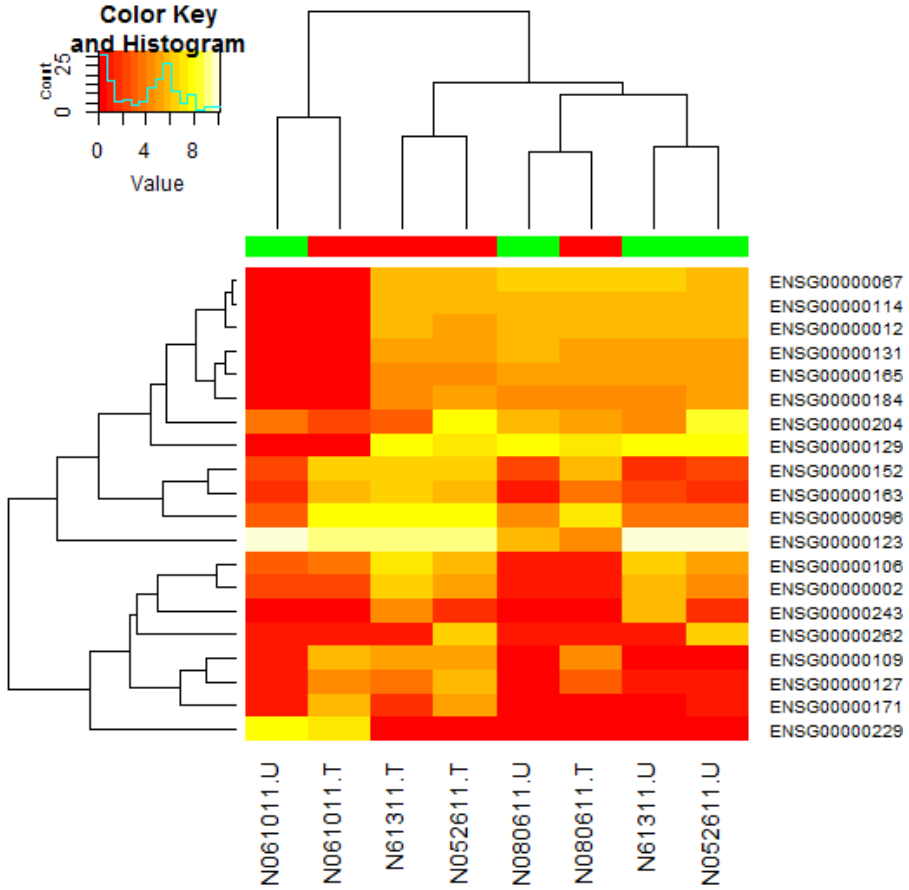
```
library(gplots)
slct=order(vars,decreasing=T)[1:20]
heatmap.2(m[slct,],trace="none",labCol=labels,
    ColSideColors=rep(c("green","red"),4),
    margins=c(7,7),Rowv=F,Colv=F)
```



CLUSTERED HEATMAP

- Heatmap can also automatically cluster both rows AND columns
- Two-way clustering: samples are clustered according to the similarity of expression patterns across (considered) genes in each sample
- Genes are clustered according to the similarity of their expression patterns across the samples

```
heatmap.2(m[slct,],trace="none",
  labCol=labels,
  ColSideColors=rep(c("green","red"),4),
  margins=c(7,7))
```



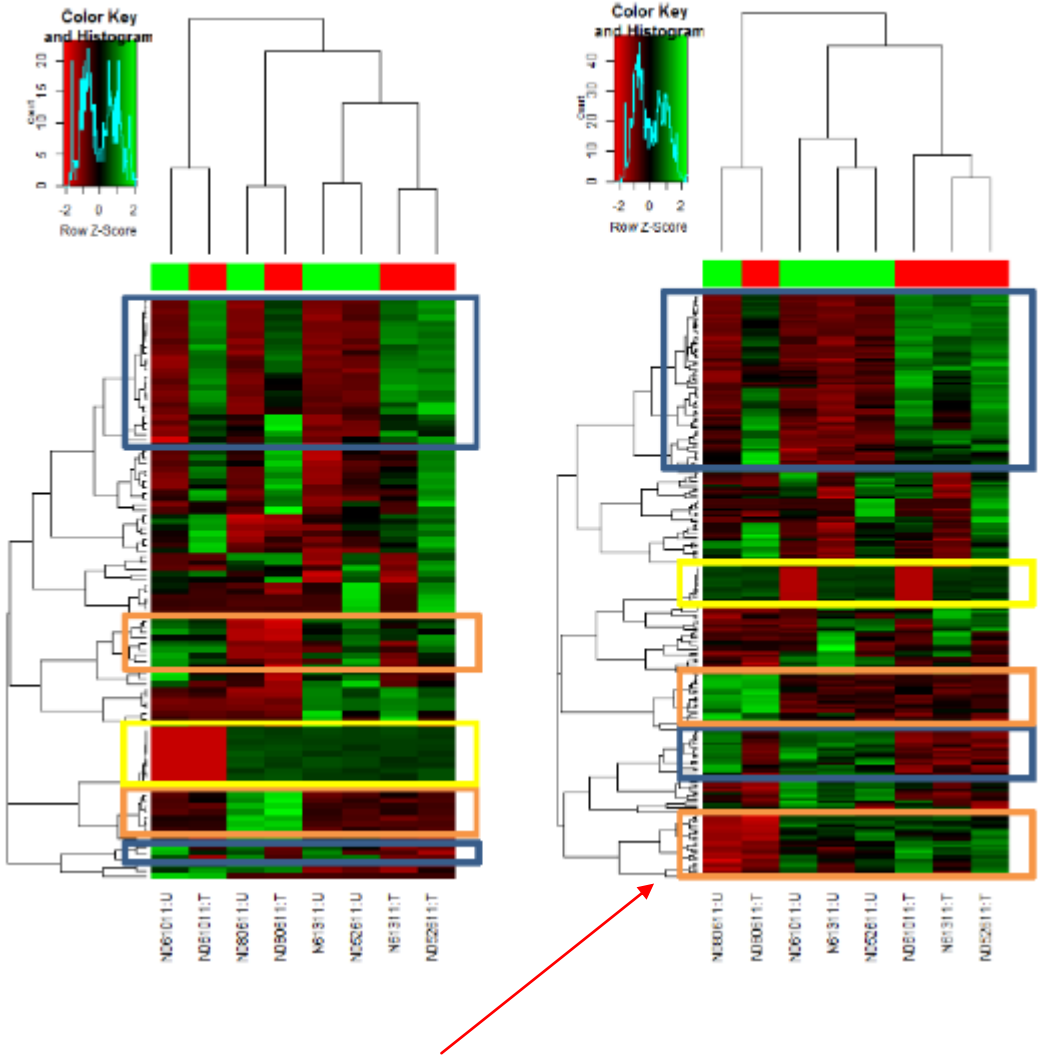
CONSIDERING LARGER SET

- Let us (a) consider the larger set of genes (comparable to what we used before), (b) use the same clustering distance (heatmap allows us to do that!) , and (c) enhance visualization:

```
# generates a vector of n colors transitioning from red, through black,
# to green
redgreen <- function(n) {
     c( hsv(h=0/6, v=seq(1,0,length=n/2) ),hsv(h=2/6, v=seq(0,1,length=n/2) ) )
}
# try 100 highest-variance genes first
slct=order(vars,decreasing=T)[1:100]
# use correlation distance, center and scale rows, use red-green palette,
# and turn off gene labels for now - there are too many to be readable anyway:
heatmap.2(m[slct,],trace="none",labCol=labels, labRow=F,
            col=redgreen(100),scale="row",
            ColSideColors=rep(c("green","red"),4),margins=c(7,7),distfun=corr.dist)
#now try 200 highest-variance genes
slct=order(vars,decreasing=T)[1:200]
heatmap.2(m[slct,],trace="none",labCol=labels,labRow=F,
            col=redgreen(100),scale="row",
            ColSideColors=rep(c("green","red"),4),margins=c(7,7),distfun=corr.dist)
```

DISTINCT PATTERS OF EXPRESSION

- We observe the same effect: one cell line starts clustering away when we consider 200 genes
- We clearly see a few different gene clusters with specific expression patterns.
  - Some distinct clusters are outlined with colored rectangles, note different expression patterns across samples
  - Note the cluster that differentiates the outlier sample from the rest (orange boxes)
  - Upon close examination, we would discover that downregulated genes in this cluster are located… on Y-chromosome!
  - In *contrast to what's claimed in the publication*, one cell line comes from a female!



SUMMARY

- The question "how many clusters" is a *very difficult* one:
  - Just like everything else we have seen so far: the question is not only "how many clusters are truly there" but also "how many clusters we can possibly distinguish with the data in hand" ! (cf. : "what's the true underlying dependence" vs "what we can possibly do with available data" – for instance should we even try higher order polynomial regression, or that would give us too many variables and lead to overfitting for the small dataset we have?)
  - There are many metrics and statistics that can (and should) be examined (again, similar to the large number of statistics and diagnostics we may want to look at when performing model selection)
- The questions of "how many clusters" and "how good are the clusters" are closely related (when the "goodness" metrics are clear and telling, we are also likely to know how many clusters are present!)
- Unsupervised learning (e.g. clustering) is very important and useful for understanding the structure of the data, sample QC, and for discovery/hypothesis generation…
  - …but its results have to be interpreted ***very*** cautiously (e.g. ISLR Ch.10.3.3)
- Another potential use: a *recommender system* (e.g. find customers who are "similar" to customer cluster X and send them offers that other members of their cluster were responding to), *anomaly detection* etc