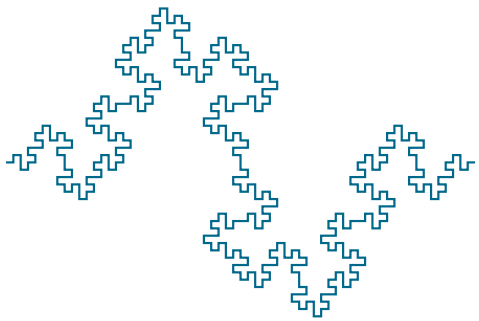
# High-Throughput Sequencing Course

## Unsupervised Learning

Biostatistics and Bioinformatics



Summer 2017

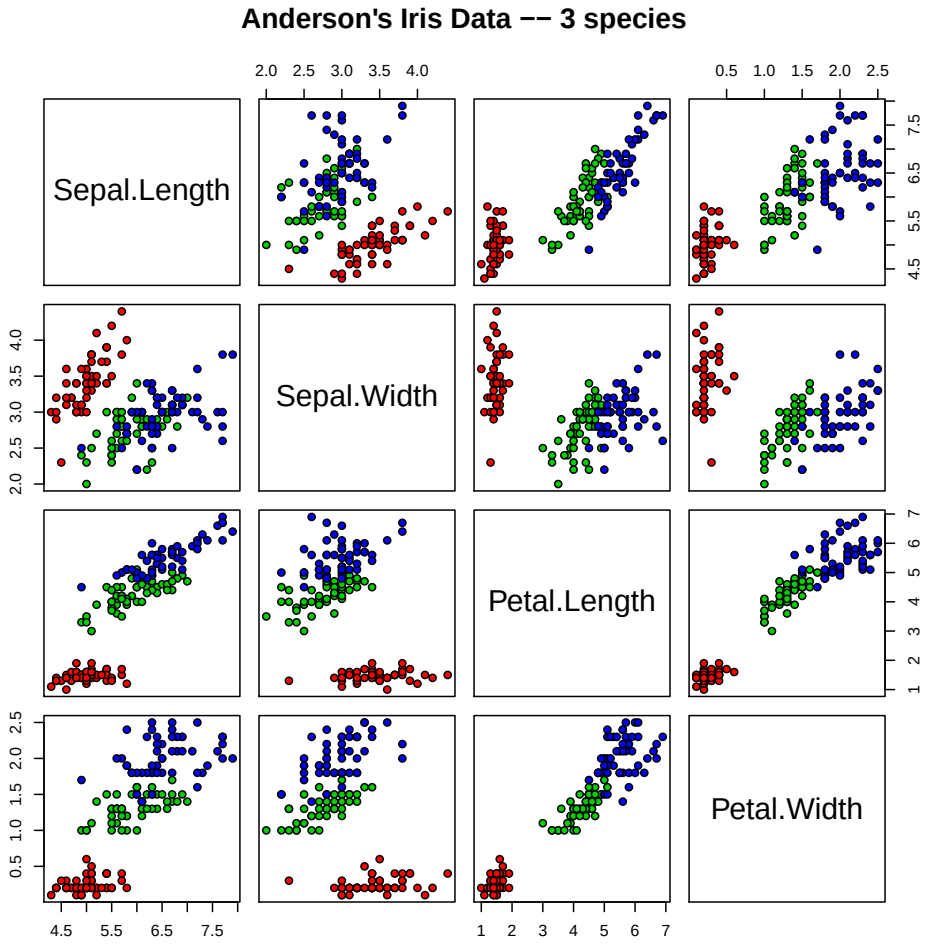
Duke University
School of Medicine

DUKE BIOSTATISTICS & BIOINFORMATICS

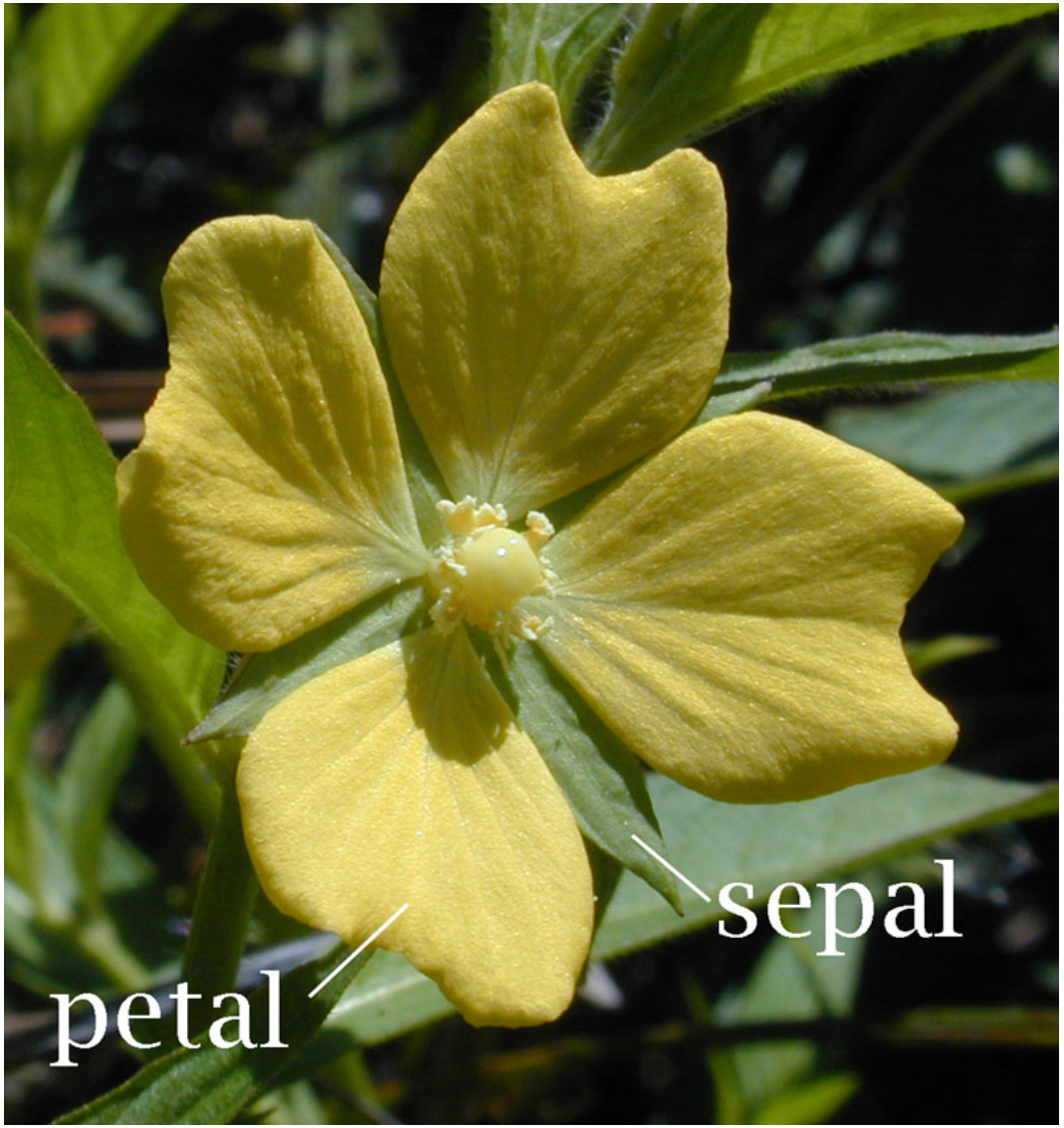
# Scope

- Let $X$ denote the genetic/genomic profile of a sample
- Often we would like to discover groups, clusters or outliers based on the genetic profiles of the samples
- These are *unsupervised* methods in the sense that the algorithm knows nothing about the grouping/clustering
- The method is only aware of the genetic profile ($X$) and not the outcome $Y$

# Fisher's Iris Data

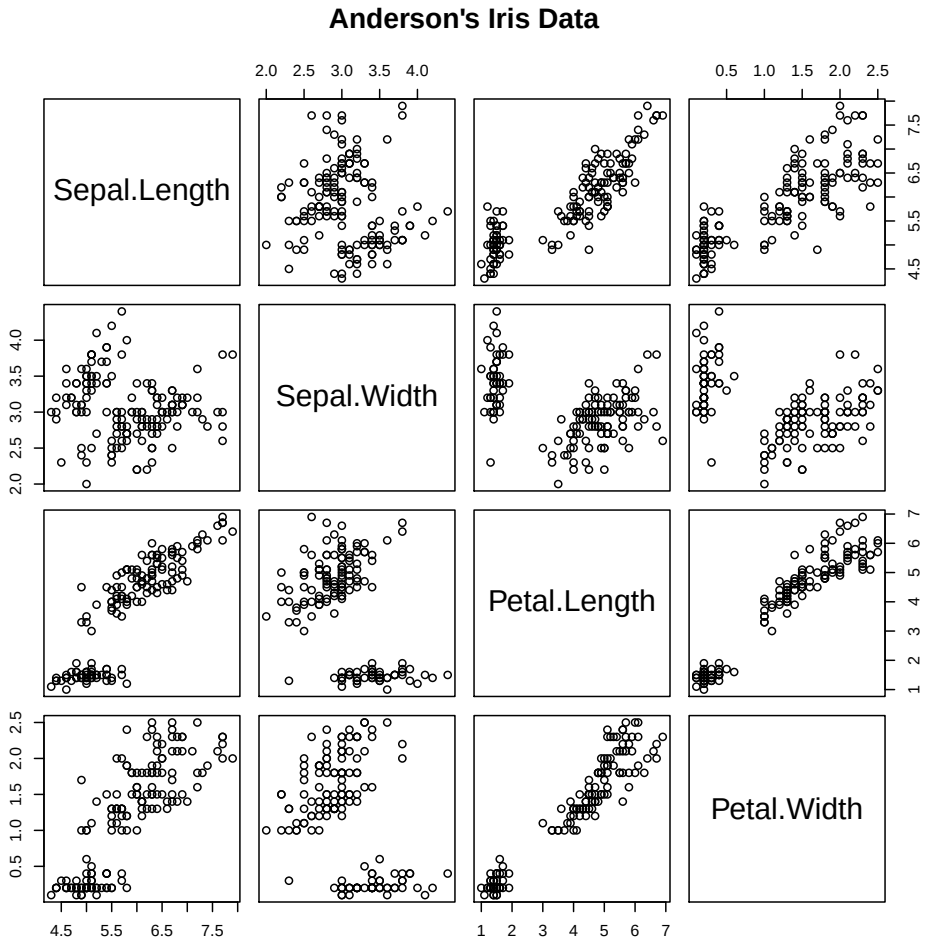


# On Petals and Sepals

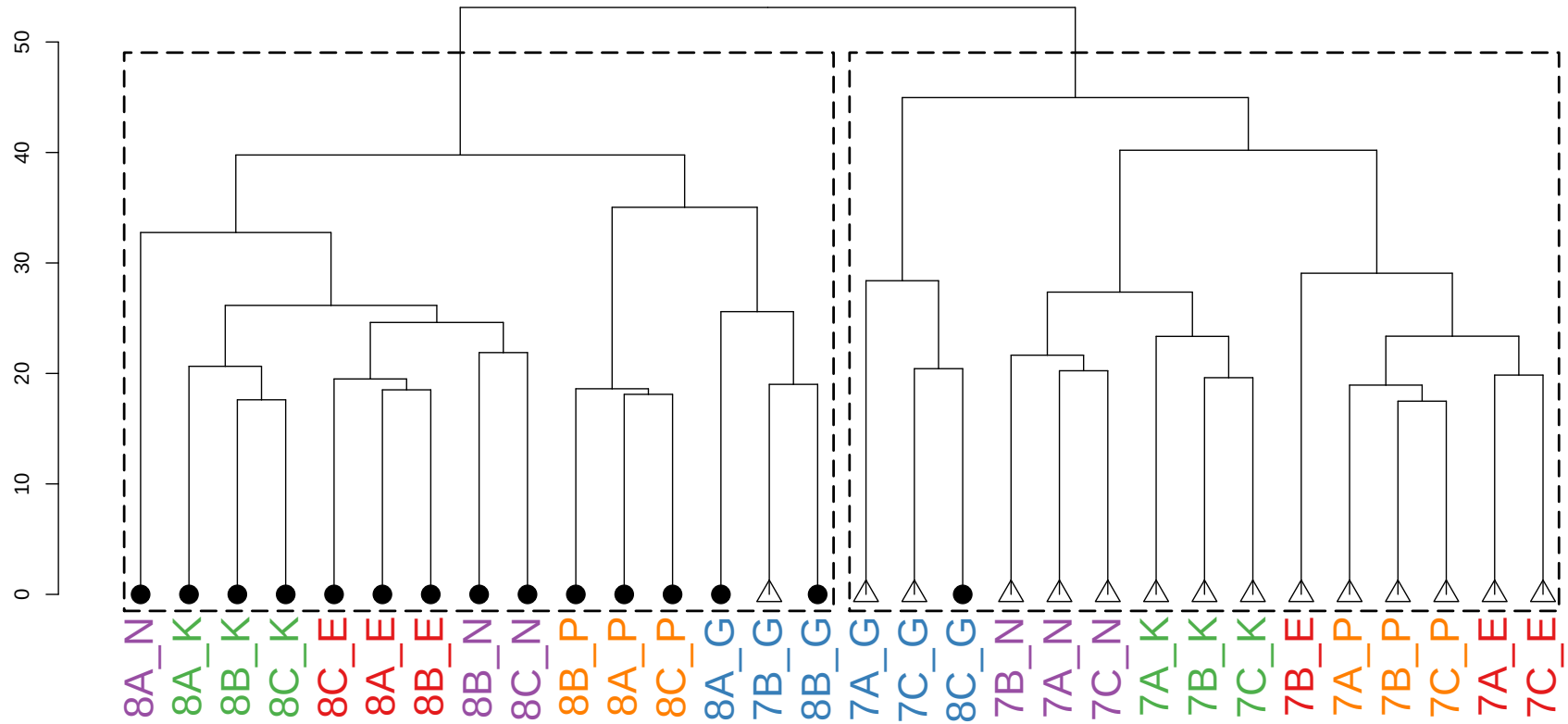



https://en.wikipedia.org/wiki/Sepal

# Fisher's Iris Data



# 2015 Data: Agglomerative Hierarchical Clustering



# A Self-fulfilling Prophecy

- Statistical methods for unsupervised learning guarantee one thing
- They will return a clustering of your data
- What they do not guarantee and are invariably unable to verify, is the biological relevance or reproducibility of the clustering
- In light of this Self-fulfilling Prophecy, these methods should be used with utmost care

# Methods to be Discussed

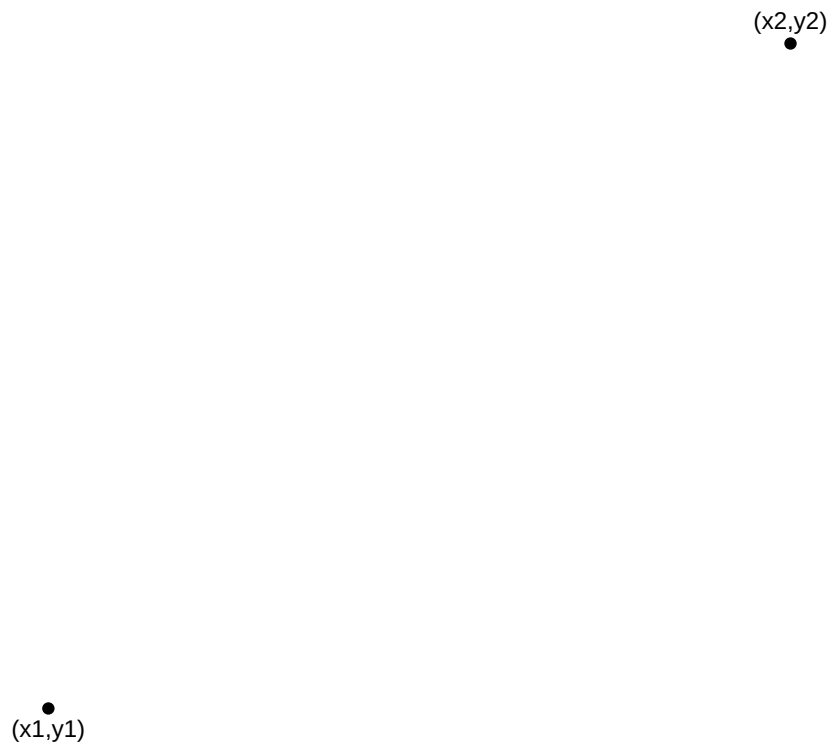
- There are many methods for unsupervised class discovery.
- We will consider three types of methods:
  - Hierarchical Clustering
  - $k$-means Clustering
  - Ordination Methods (e.g., Multi-Dimensional Scaling (MDS) and Principal Components (PC))
- Note that there are many variations of these methods
- Most mathematical details will be left out
- We focus on discovering classes among samples (not genes)

# Distance between Two Points

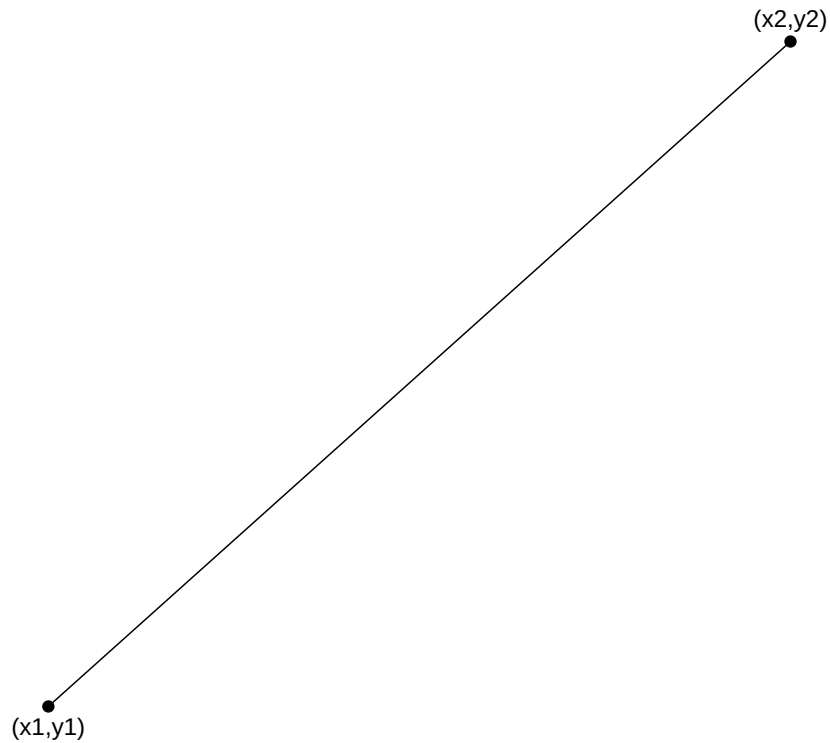
- Many class discover methods aim to quantify the similarity (or dissimilarity) among patients
- For each patient, the vector of gene expression can be thought of a "point" in an $m$-dimensional space
- For many class discovery methods, one has to be able to quantify the "distance" between two points (the expression profiles between two individuals)
- A common distance measure is the Euclidean distance

# Distance (Two points on the plane)

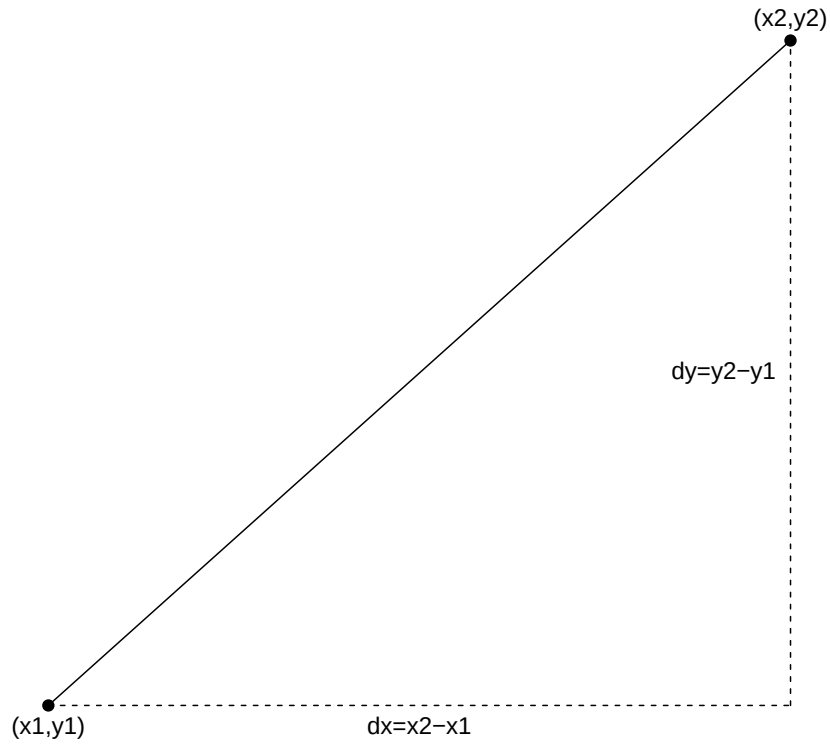
# Distance (Coordinates)



# Distance



# Distance (horizontal/vertical shifts)



# Pythagorean Theorem (on the plane)

- According to the Pythagorean theorem

$$h^2 = \mathrm{d}x^2 + \mathrm{d}y^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2$$

- $h$ is called the hypotenuse
- The distance between $(x_1, y_1)$ and $(x_2, y_2)$ is given by

$$h = \sqrt{\mathrm{d}x^2 + \mathrm{d}y^2} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

# Pythagorean Theorem (on the plane)

- Can be extended to higher dimensions
- In a three-dimensional space the distance between $(x_1, y_1, z_1)$ and $(x_2, y_2, z_2)$ is given by

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

- For any given dimension, the distance is obtained as the square root of the sum of the square of the coordinate-wise differences

# Golub *et al* Leukemia Data

- 47 patients with acute lymphoblastic leukemia (ALL)
- 25 patients with acute myeloid leukemia (AML)
- Platform: Affymetrix Hgu6800
- 7129 probe sets
- Golub *et al.* (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science, Vol. 286:531-537.

# Golub *et al* Leukemia Data

Expression data from first three features and 5 patients

```
dim(exprs(Golub_Merge))
```

```
## [1] 7129   72
```

```
exprs(Golub_Merge)[1:3, 1:5]
```

```
##                  39   40   42   47   48
## AFFX-BioB-5_at -342  -87   22 -243 -130
## AFFX-BioB-M_at -200 -248 -153 -218 -177
## AFFX-BioB-3_at   41  262   17 -163  -28
```

# Golub *et al* Leukemia Data: Distance

## Expression vector for patients 39 and 40

```
x <- exprs(Golub_Merge)[, "39"]
y <- exprs(Golub_Merge)[, "40"]
```

## Lengths of these vectors

```
length(x)
```

```
## [1] 7129
```

```
length(y)
```

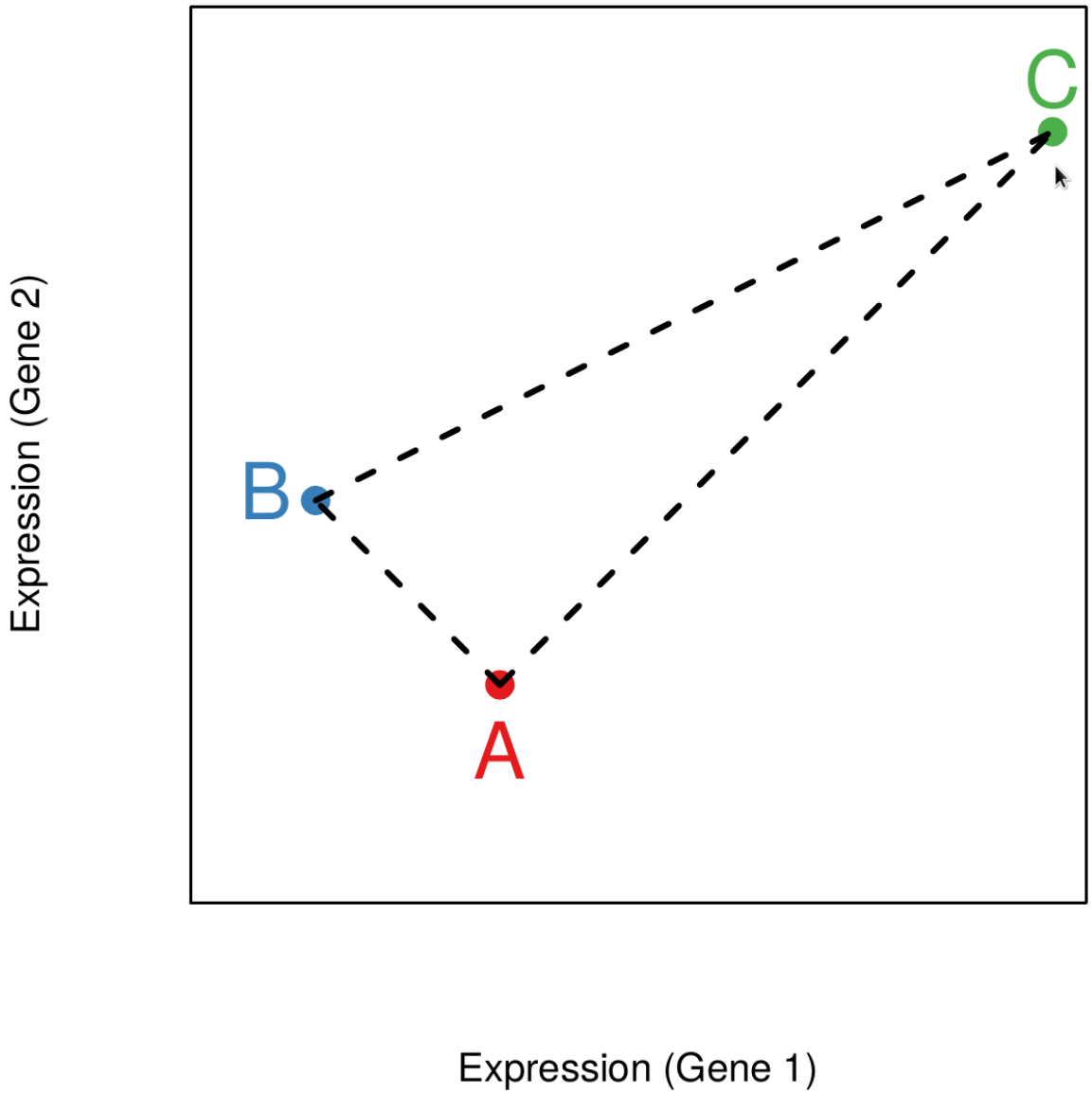
```
## [1] 7129
```

## Distance between these two vectors

```
sqrt(sum((x - y)^2))
```

```
## [1] 101530.8
```

# Relative Distance (From CST 2011 Paper)



# Dissimilarity matrix

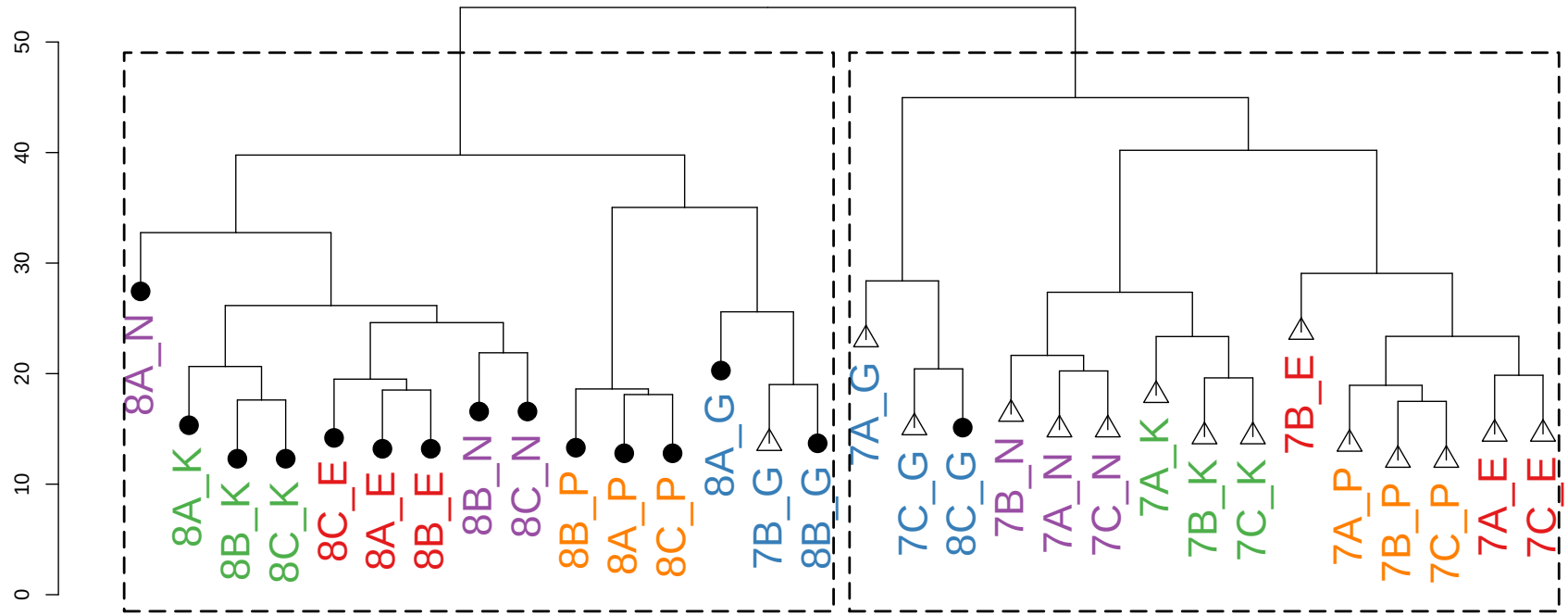
- Use pairwise distances to quantify similarity (or dissimilarity) among patients
- Construct a matrix containing all pairwise distances
- Take the first three patients in the Golub data set

```
dist(t(exprs(Golub_Merge[, 1:3])))
```

```
##           39        40
## 40 101530.75
## 42  94405.04  89502.29
```

- Patient 42 is more similar (closer) to patient 39 than patient 40 (distance of 94405.04 vs 101530.75)
- Patient 39 is more similar (closer) to 42 than patient 40 (distance of 94405.04 vs 101530.75)

# 2015 Data: Agglomerative Hierarchical Clustering



# Clusters

- Let $c_1, c_2, \ldots, c_n$ denote the $n$ samples
- Define a cluster to be a set of patients
  - $(c_1)$ is a cluster with one member: $c_1$
  - $(c_1, c_3)$ is a cluster of two members: $c_1$ and $c_3$
  - $(c_1, c_2, c_3)$ is a cluster of three members of $c_1, c_2$ and $c_3$
- Note that $c_1$ and $(c_1)$ are different entities

# Notion of a Linkage

- The distance measure quantified the distance between two points
- In clustering, you need to think about the criterion to link (merge) the clusters
- maximum distance (aka complete linkage)
- average distance (aka average linkage)
- minimum distance (aka single linkage)

# Agglomerative Hierarchical Clustering

- Agglomerate: To form clusters
- Let each of the $n$ points be its own cluster ($n$ clusters each with one single member)
- Find the pair of clusters that is most similar
- Merge these two
- Now you have $n - 1$ clusters (1 cluster with two members and $n - 2$ clusters each with a single member)
- Compute the similarities between the $n - 2$ "old" clusters with the new cluster
- Repeat the last two steps until all members have been merged into a single cluster.

# Clustering Cities by Distances

| | ATL | BOS | ORD | DCA |
|---|---|---|---|---|
| ATL | 0 | 934 | 585 | 542 |
| BOS | 934 | 0 | 853 | 392 |
| ORD | 585 | 853 | 0 | 598 |
| DCA | 542 | 392 | 598 | 0 |

# Clustering Cities by Distances (Single Linkage)

| | ATL | BOS | ORD | DCA |
|---|---|---|---|---|
| ATL | 0 | 934 | 585 | 542 |
| BOS | 934 | 0 | 853 | 392 |
| ORD | 585 | 853 | 0 | 598 |
| DCA | 542 | 392 | 598 | 0 |

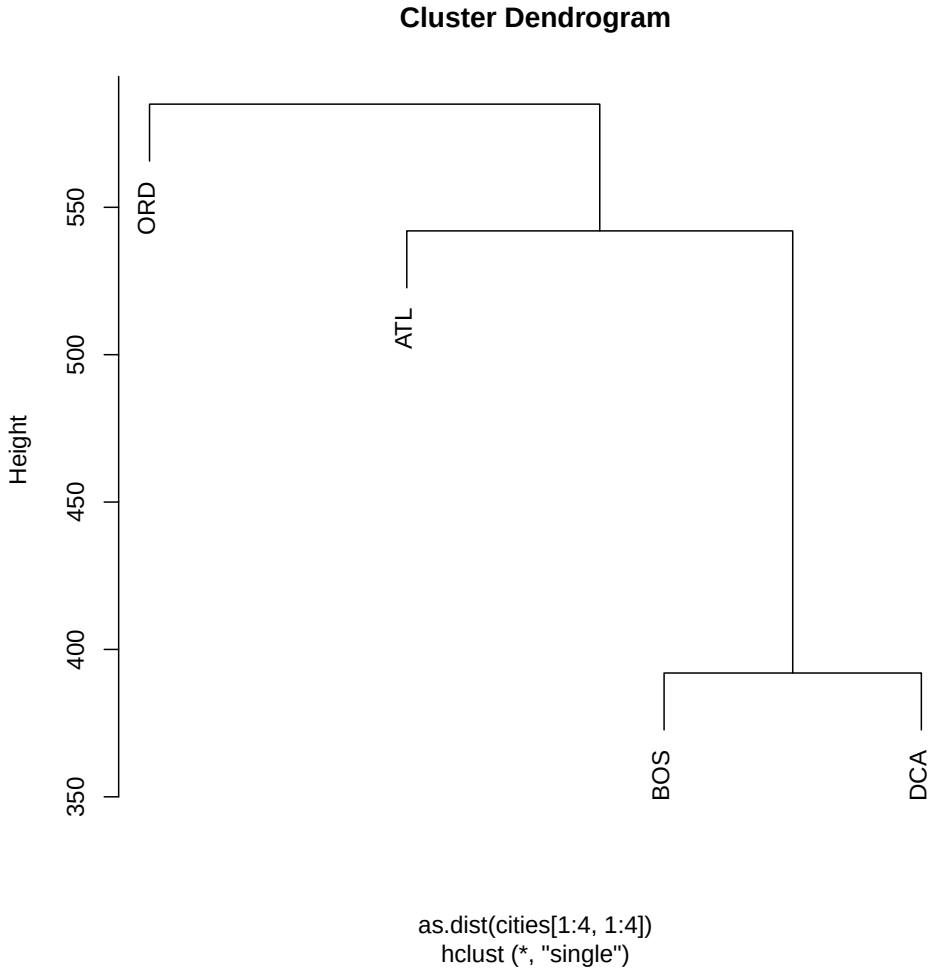
| | DCA-BOS | ATL | ORD |
|---|---|---|---|
| DCA-BOS | 0 | 542 | 598 |
| ATL | 542 | 0 | 585 |
| ORD | 598 | 585 | 0 |

# Clustering Cities by Distances (Single Linkage)

| | DCA-BOS | ATL | ORD |
|---|---|---|---|
| DCA-BOS | 0 | 542 | 598 |
| ATL | 542 | 0 | 585 |
| ORD | 598 | 585 | 0 |

| | DCA-BOS-ATL | ORD |
|---|---|---|
| DCA-BOS-ATL | 0 | 585 |
| ORD | 585 | 0 |

# Four Airports (Single linkage)



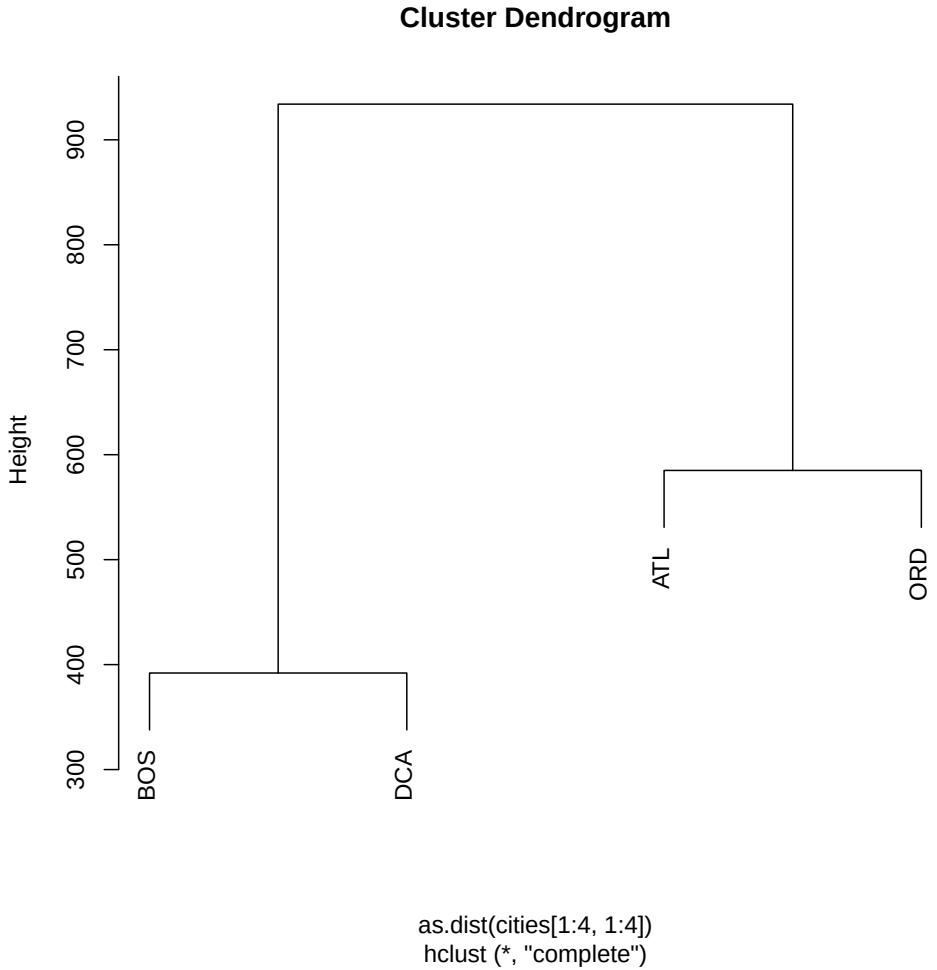
# Clustering Cities by Distances (complete linkage)

| | ATL | BOS | ORD | DCA |
|---|---|---|---|---|
| ATL | 0 | 934 | 585 | 542 |
| BOS | 934 | 0 | 853 | 392 |
| ORD | 585 | 853 | 0 | 598 |
| DCA | 542 | 392 | 598 | 0 |

| | DCA-BOS | ATL | ORD |
|---|---|---|---|
| DCA-BOS | 0 | 934 | 853 |
| ATL | 934 | 0 | 585 |
| ORD | 853 | 585 | 0 |

| | DCA-BOS | ATL-ORD |
|---|---|---|
| DCA-BOS | 0 | 934 |
| ATL-ORD | 934 | 0 |

# Four Airports (complete linkage)



# Four Airports (side by side)

| | ATL | BOS | ORD | DCA |
|---|---|---|---|---|
| ATL | 0 | 934 | 585 | 542 |
| BOS | 934 | 0 | 853 | 392 |
| ORD | 585 | 853 | 0 | 598 |
| DCA | 542 | 392 | 598 | 0 |

| | DCA-BOS | ATL | ORD |
|---|---|---|---|
| DCA-BOS | 0 | 934 | 853 |
| ATL | 934 | 0 | 585 |
| ORD | 853 | 585 | 0 |

| | DCA-BOS | ATL-ORD |
|---|---|---|
| DCA-BOS | 0 | 934 |
| ATL-ORD | 934 | 0 |

Table: Complete Linkage

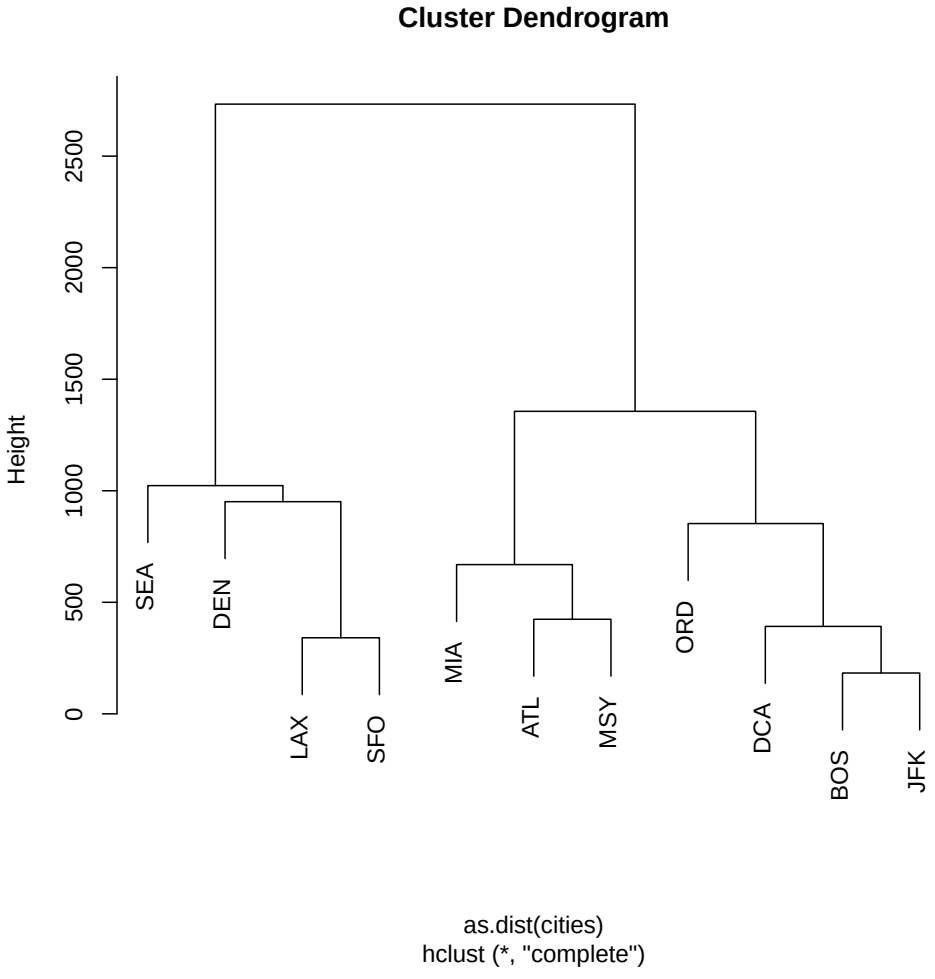
| | ATL | BOS | ORD | DCA |
|---|---|---|---|---|
| ATL | 0 | 934 | 585 | 542 |
| BOS | 934 | 0 | 853 | 392 |
| ORD | 585 | 853 | 0 | 598 |
| DCA | 542 | 392 | 598 | 0 |

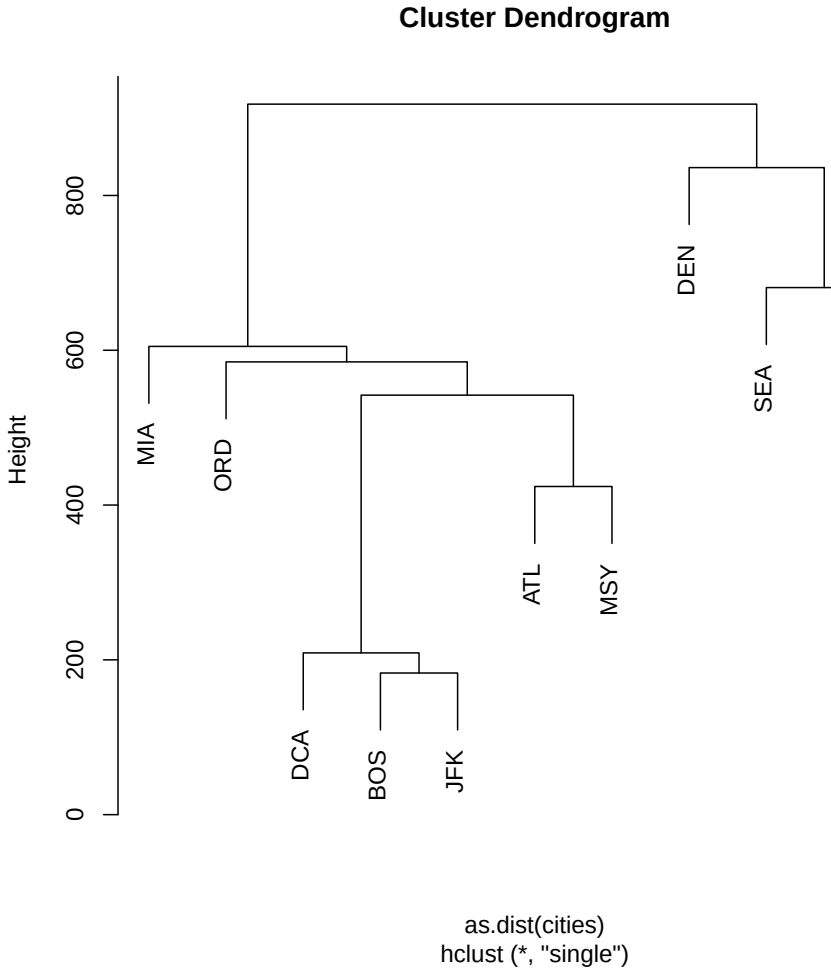
| | DCA-BOS | ATL | ORD |
|---|---|---|---|
| DCA-BOS | 0 | 542 | 598 |
| ATL | 542 | 0 | 585 |
| ORD | 598 | 585 | 0 |

| | DCA-BOS-ATL | ORD |
|---|---|---|
| DCA-BOS-ATL | 0 | 585 |
| ORD | 585 | 0 |

Table: Single Linkage

# All Airports (comparison)



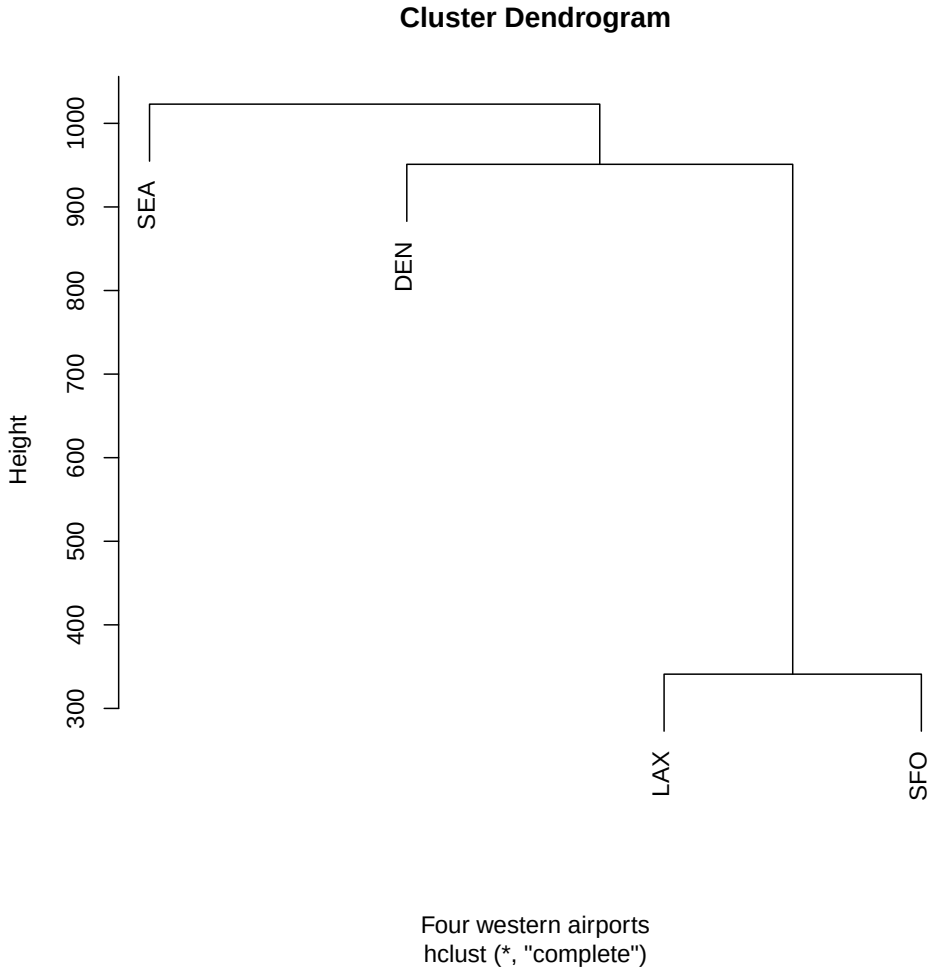



# Western Airports: Exercise

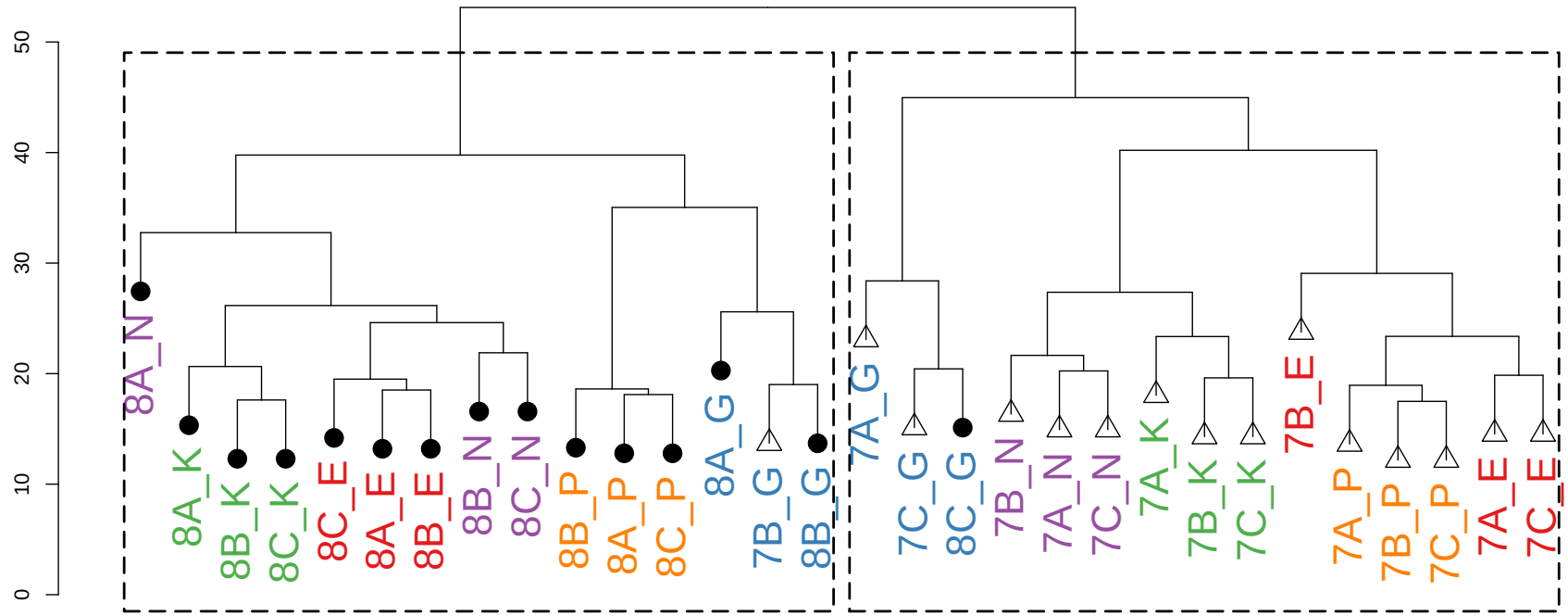
Carry out hierarchical clustering with complete linkage

```
##      DEN LAX  SEA SFO
## DEN    0 836 1023 951
## LAX  836   0  957 341
## SEA 1023 957    0 681
## SFO  951 341  681   0
```

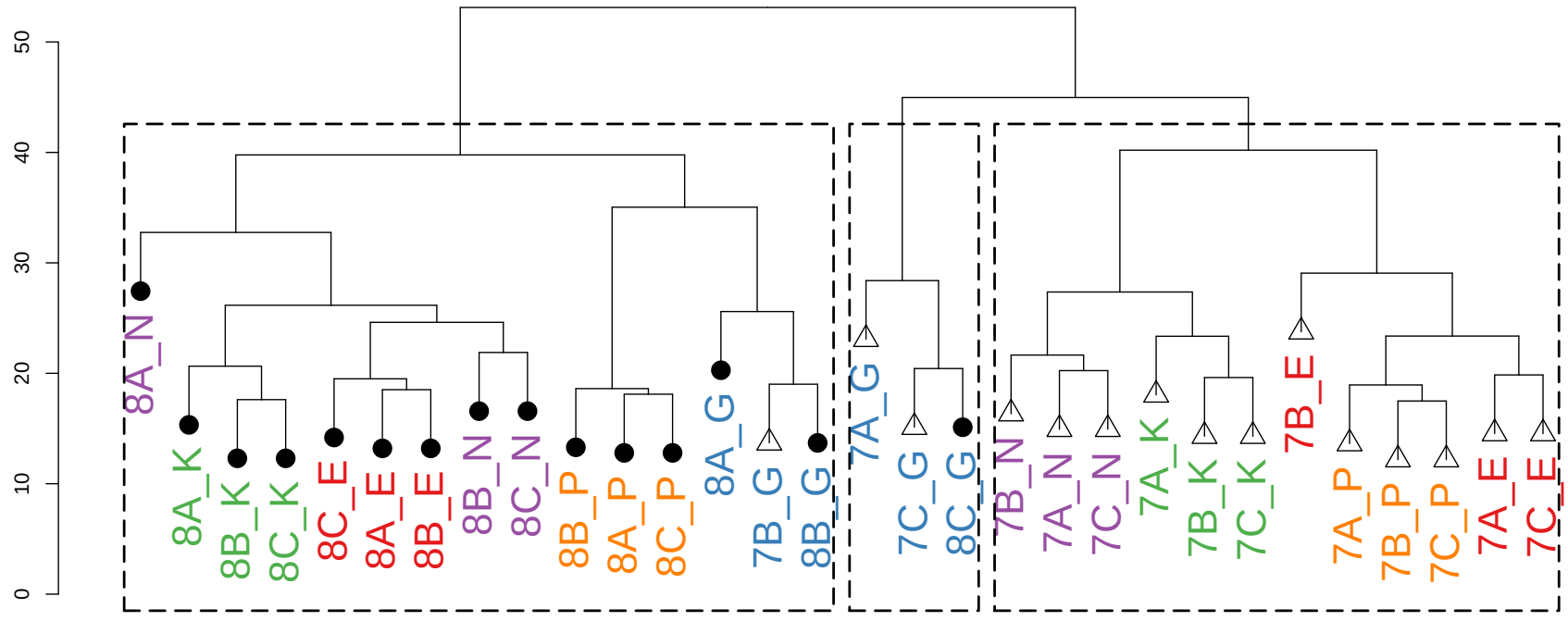
# Western Airports: Solution



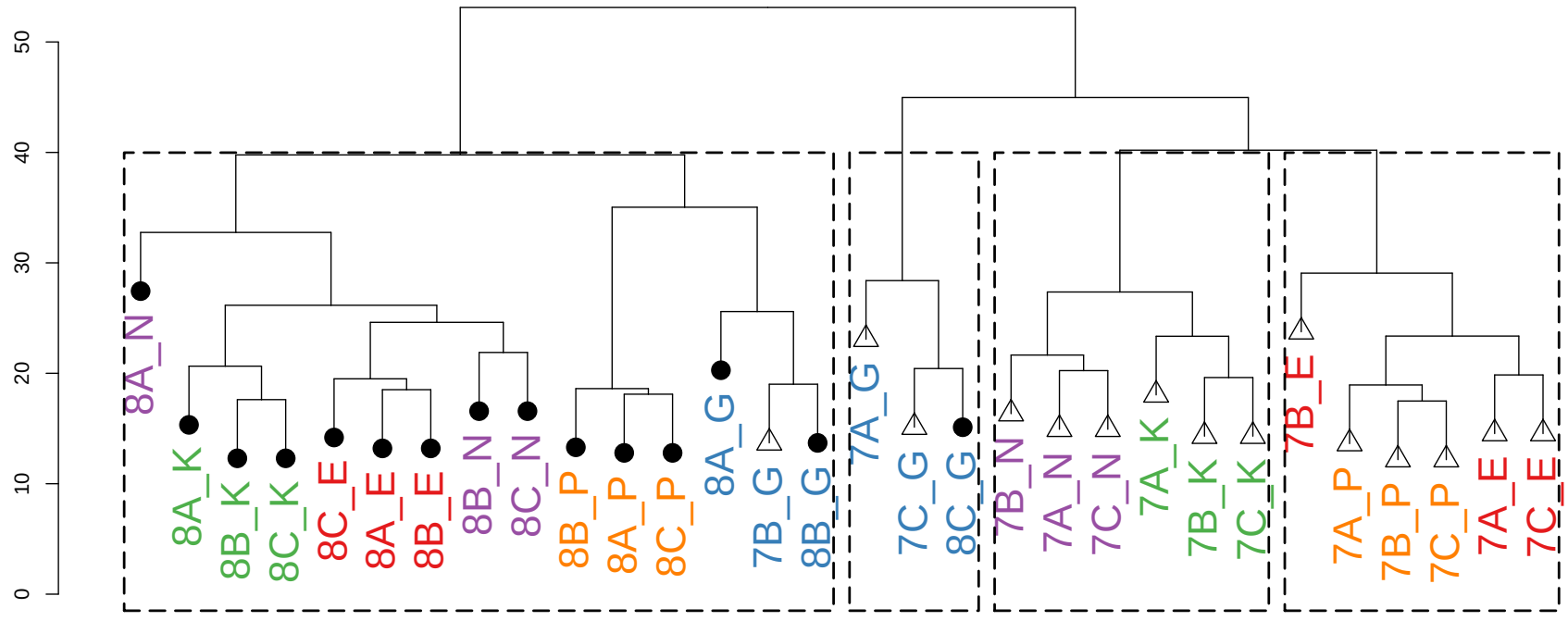
# 2015 Data: Agglomerative Hierarchical Clustering Complete Linkage



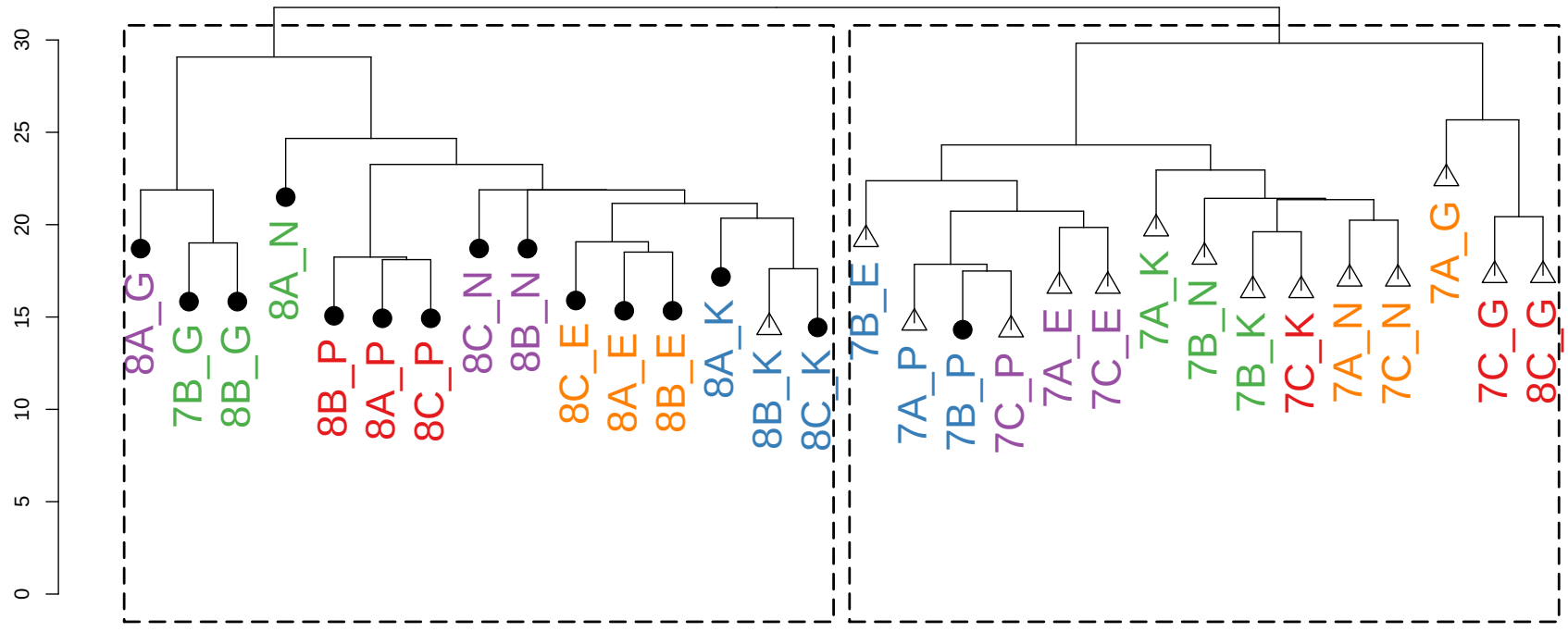
# 2015 Data: Agglomerative Hierarchical Clustering Complete Linkage



# 2015 Data: Agglomerative Hierarchical Clustering Complete Linkage



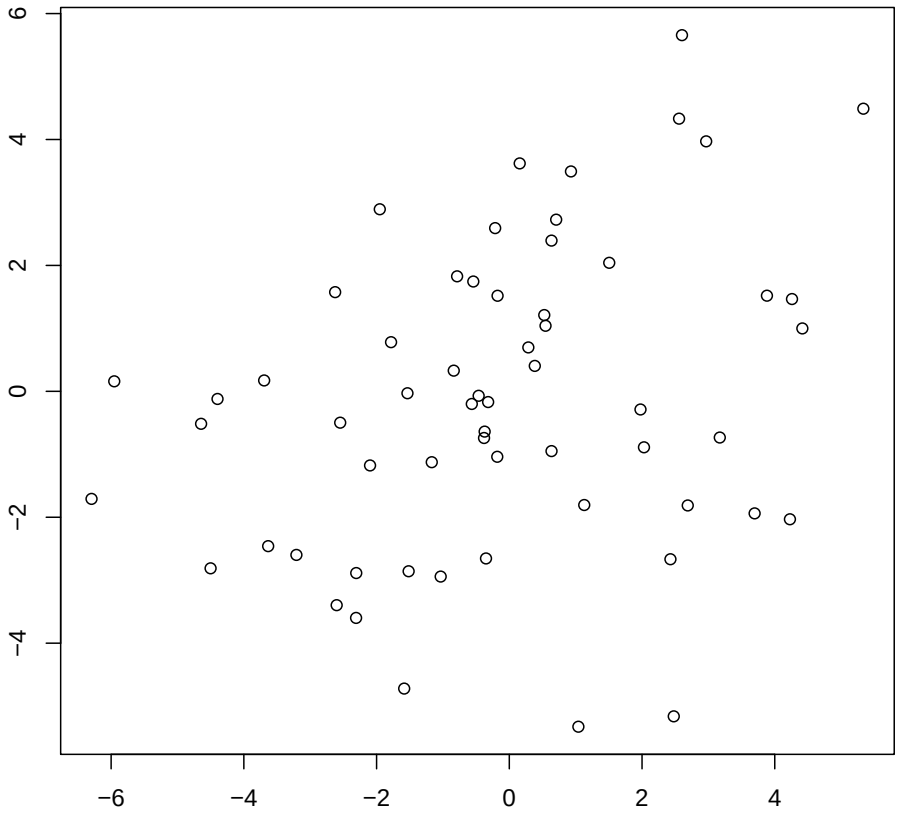
# 2015 Data: Agglomerative Hierarchical Clustering Single Linkage



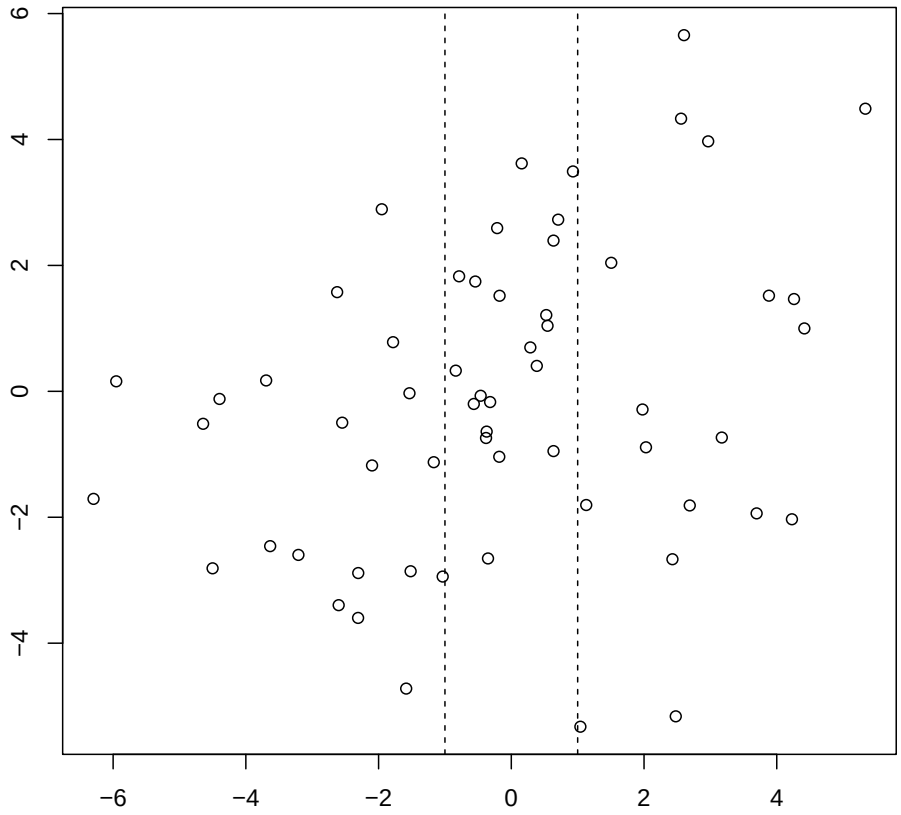
# $k$-means Clustering

- Specify a number of potential clusters ($k$)
- Split of the data (either randomly or based on some previous results) into $k$ partitions
- Compute the mean (aka centroid) for each partition
- For the first point (sample) determine the *nearest* centroid
- The closeness is typically quantified using the Euclidean distance
- Assign that point to that center
- Repeat for points 2 through $n$
- Assess the fit using the intra-cluster variance
- Repeat as needed.

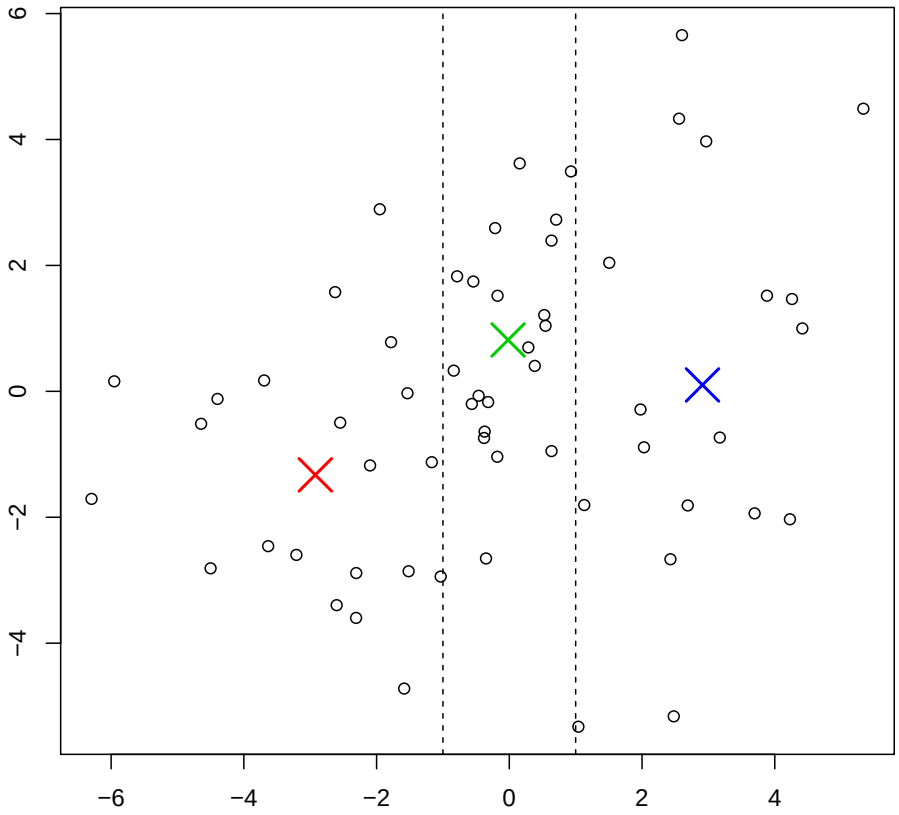
# $k$-means clustering: Data



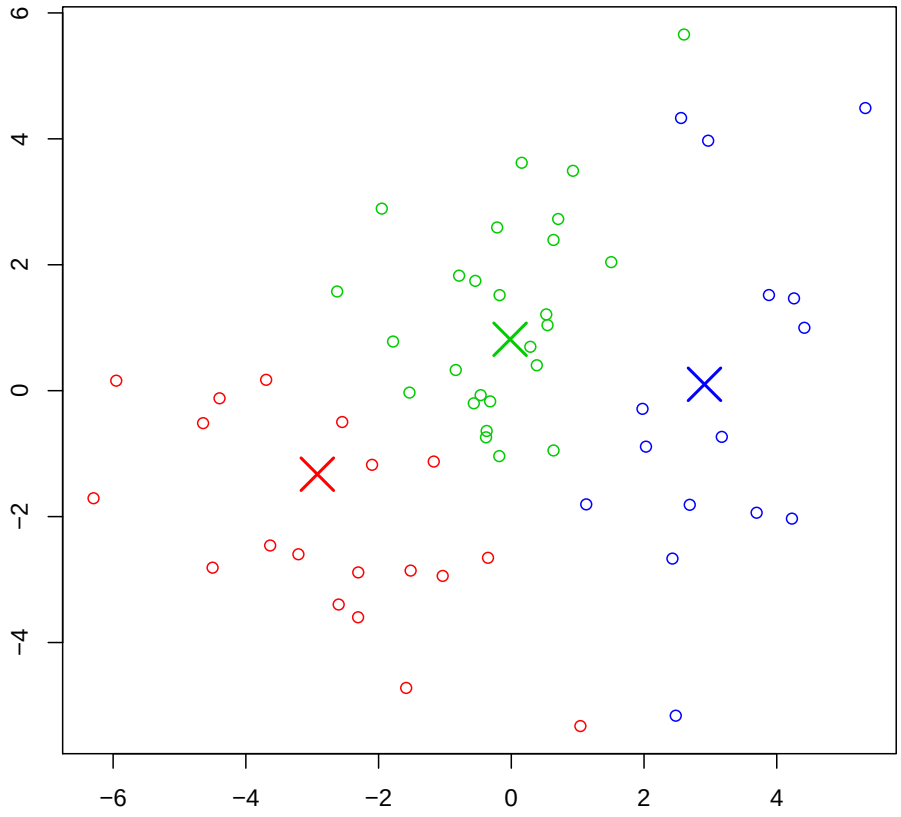
# $k$-means clustering: Initial Clusters



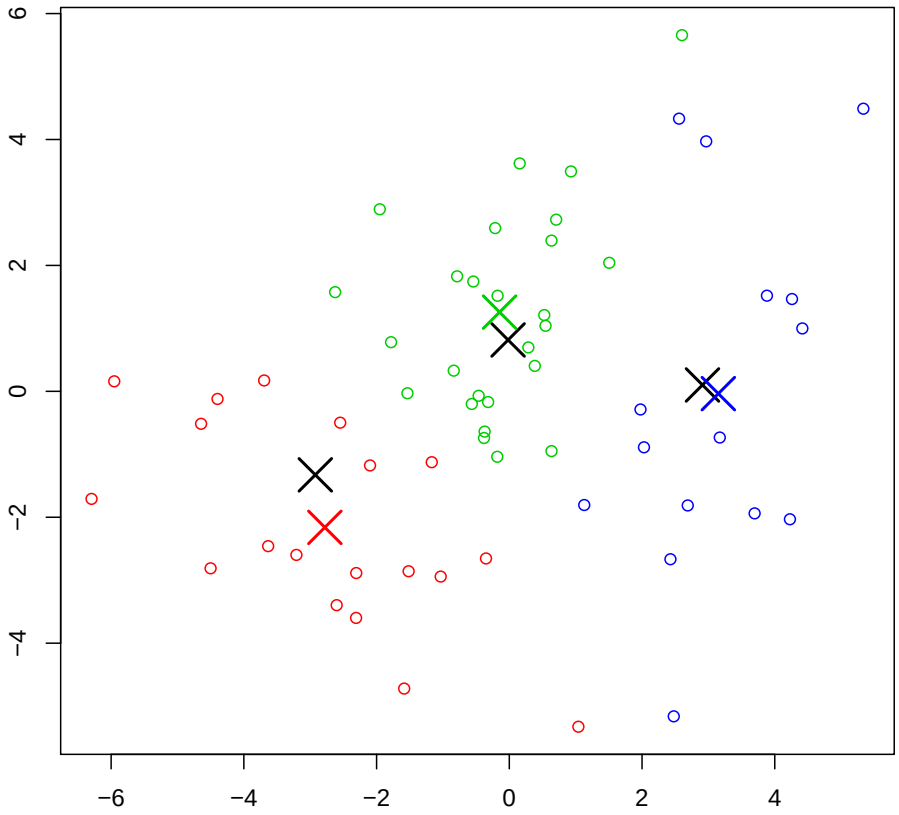
# $k$-means clustering: Initial Centers



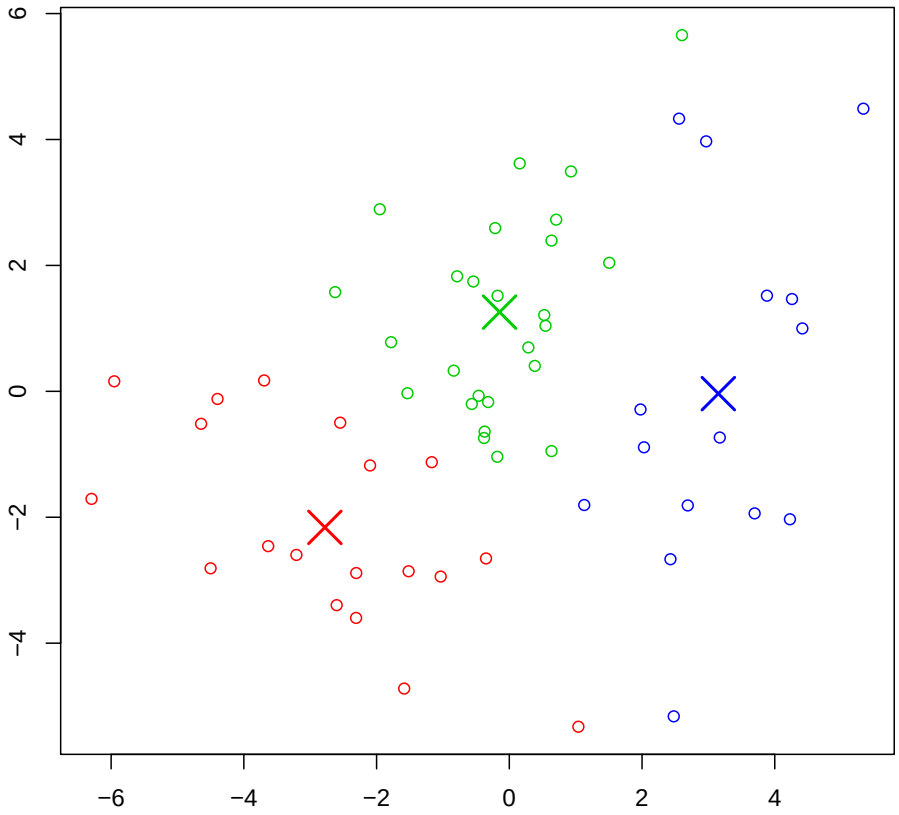
# $k$-means clustering: Label points according to centers



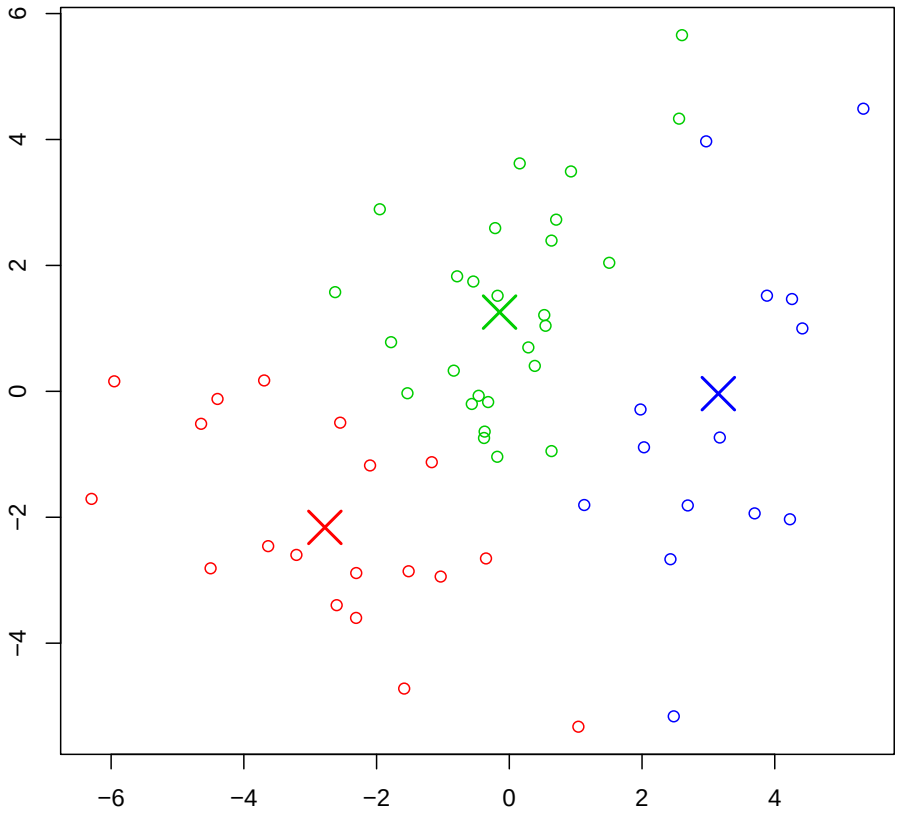
# $k$-means clustering: Update Centers



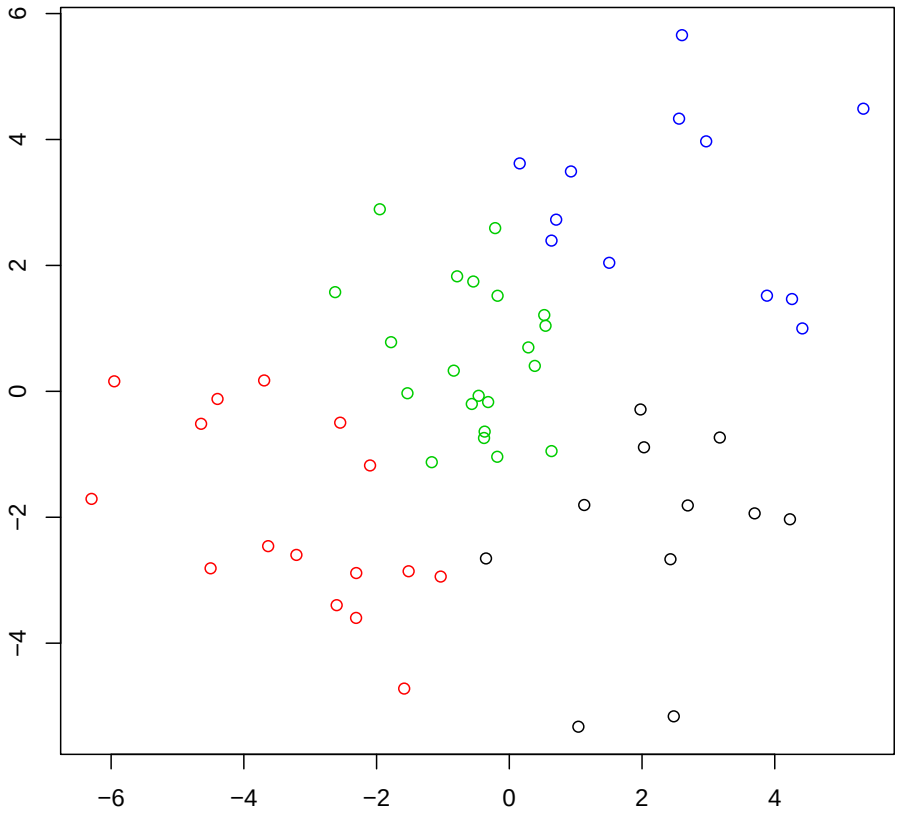
# $k$-means clustering: Update Centers



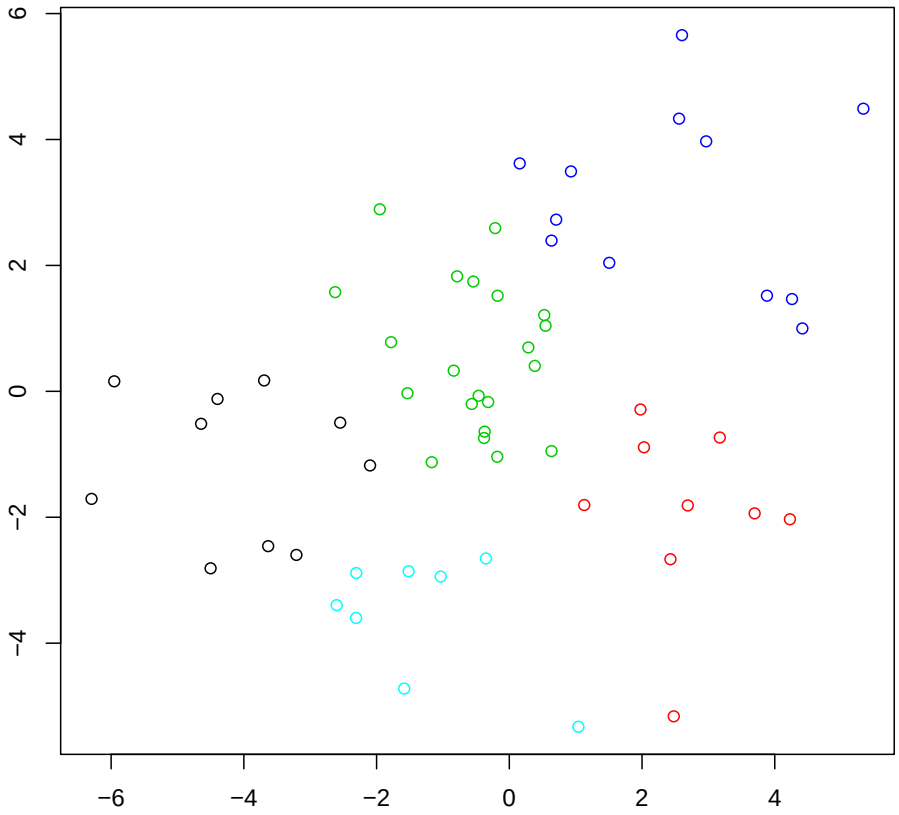
# $k$-means clustering: Update Points



# Why not 4 clusters?



# Why not 5 clusters?



# $k$-MEANS

- This is an example of *non-hierarchical* clustering
- Need to specify the number of clusters up front
- Need to specify (deterministically or randomly) the centers of the clusters up front
- Results are sensitive to the choice of $k$ and initial partitions
- Note: All the data points were simulated from a single cluster!

# Dimension reduction

- Genome-wide profiling platforms are high-dimensional ($m$ is large)
- Visualization beyond $m = 3$ not possible (for mortals)
- Representing the data by a lower dimensional format without losing too much information is desired.
- Two guiding principles:
  - Keep variables with highest variability
  - Reduce redundancy

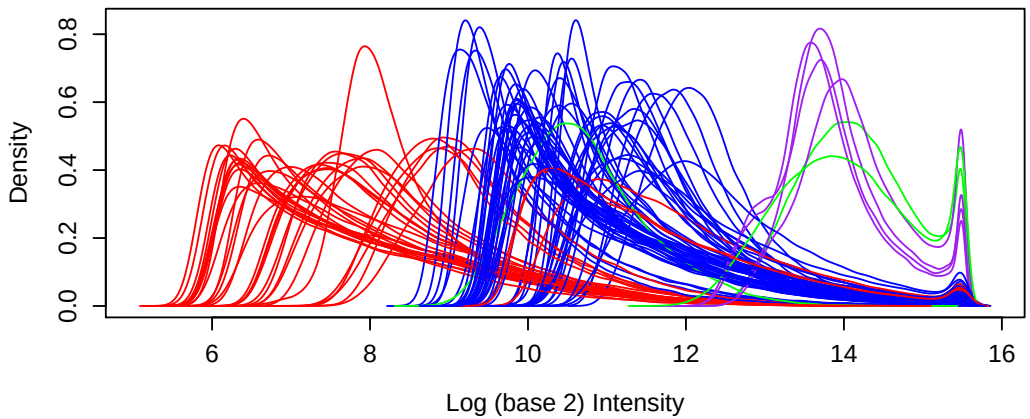
# Multi-Dimensional Scaling (MDS)

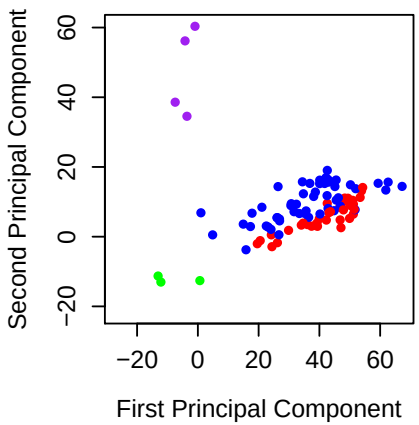
- Compute the dissimilarity matrix based on a distance measure
- Project the points into a lower dimensional space (say 2D or 3D) while preserving the similarity matrix
- PCA is a related (and in a sense equivalent method to MDS)
- Project the points into a lower dimensional space where the new variables are linear combinations of the original variables
- The new variables are chosen so as to have maximum variance and to be uncorrelated.

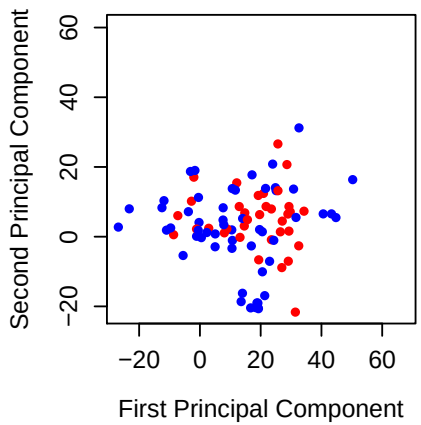
# Batch Effect Discovery

- The MDS method is very useful for detecting batch effects
- Batch effects tend to be stronger that biological effects
- They also affect most probe sets (the biological effect may only be captured by a few)
- This can be an effective weapon in your QC arsenal (this is how I start any new analysis)

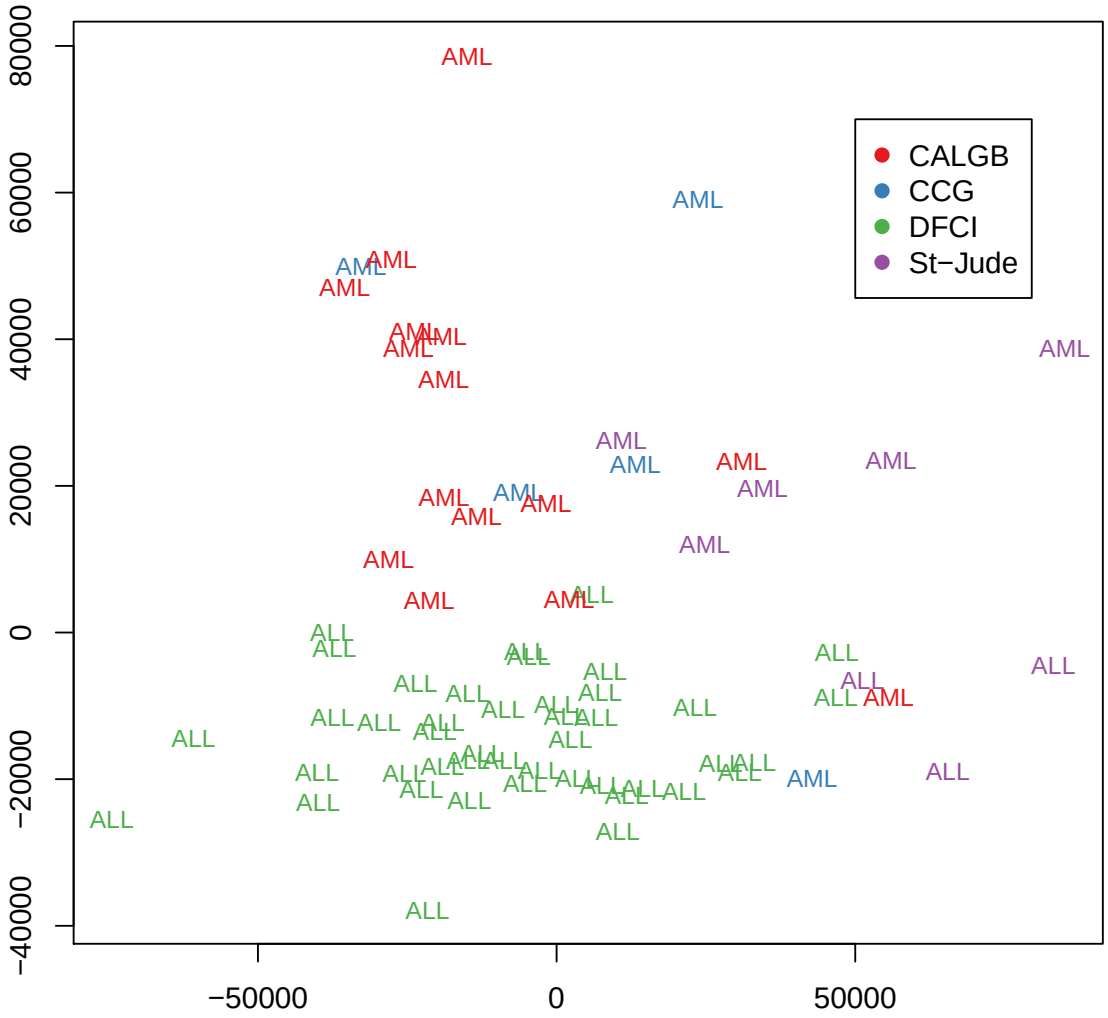
# From CCR 2008 Paper



# ALL/AML Data


CALGB
CCG
DFCI
St−Jude
80000
60000
40000
20000
0
−20000
−40000
−50000
0
50000

# Semi-supervised Learning

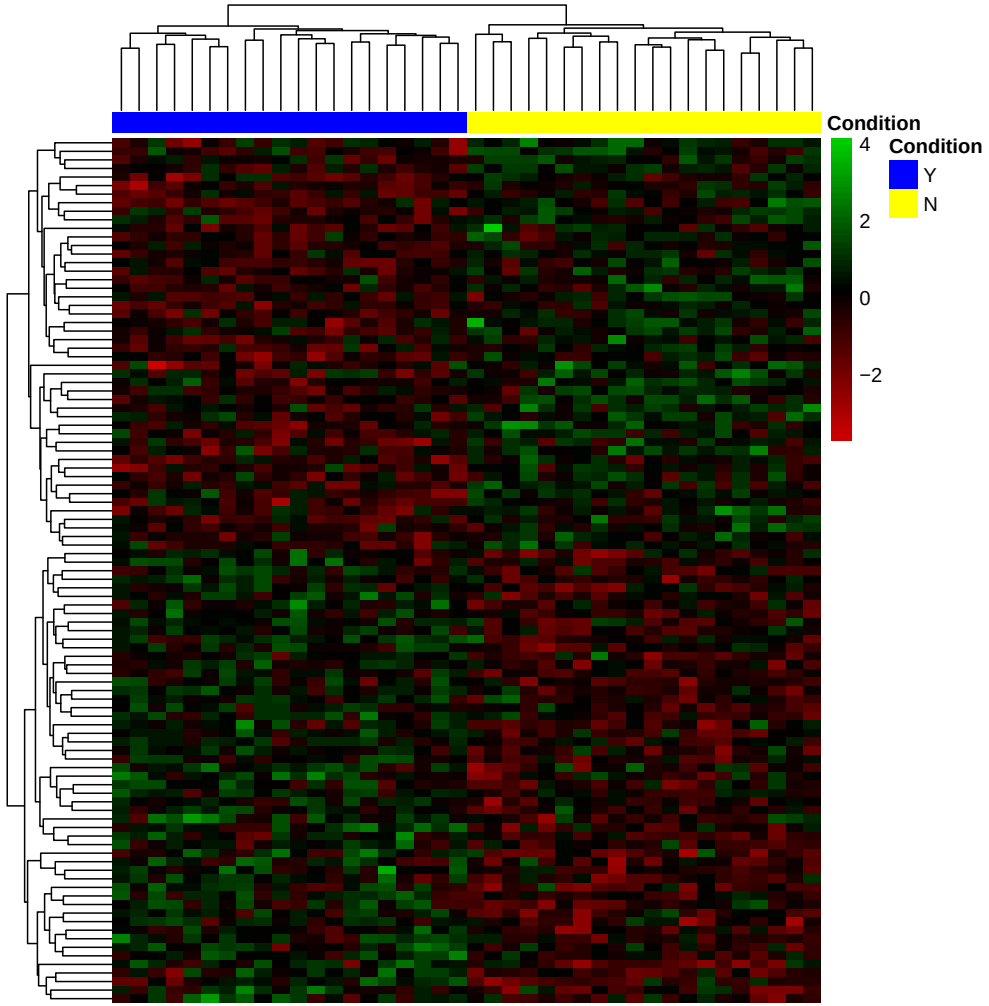
- Heatmap illustration:
  - Select a panel of probe-sets based on the two-sample $t$-test
  - Carry out hierarchical clustering with respect to the patients (the columns)
  - Carry out hierarchical clustering with respect to the probe sets in the panel (the rows)
  - Present the results using a heatmap
- Some consider this an *unsupervised* analysis as the hierarchical clustering algorithm is unaware of the classes
- This is not an accurate assessment: It is semi-supervised in the sense that we are picking genes based on the phenotype
- A procedure is *unsupervised* if the class info is only used for annotation

# R Code to simulate Heatmap

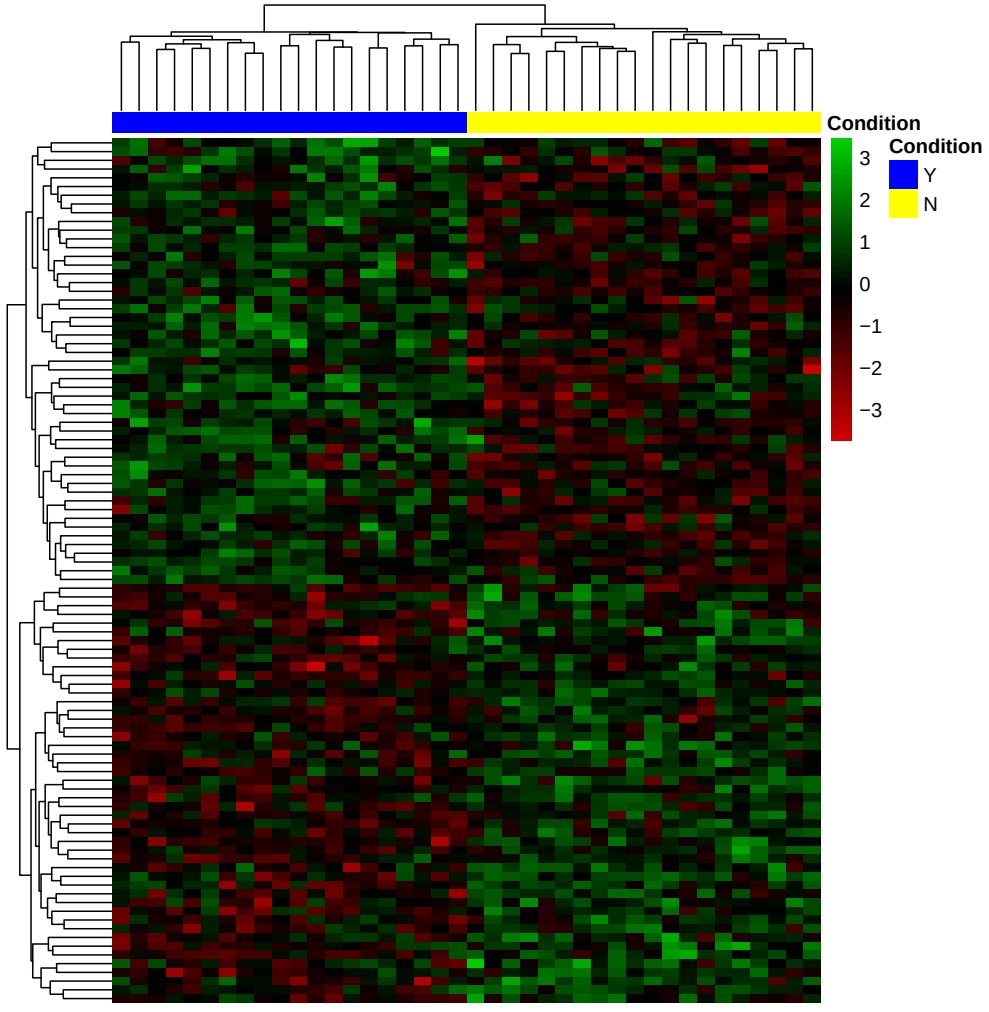
```
simulate.noise.heatmap = function(n, m, alpha) {
    # Simulate Expression Matrix
    EXPRS = matrix(rnorm(2 * n * m), m, 2 * n)
    grp = factor(rep(0:1, c(n, n)))
    rownames(EXPRS) = paste("Gene", 1:m, sep = "")
    colnames(EXPRS) = paste("patient id", 1:(2 * n), sep = "")

    # Get the two sample t-statistics
    pvals = rowttests(EXPRS, grp)$p.value
    topgenes = which(pvals < alpha)
    EXPRS = EXPRS[topgenes, ]
    annodat = data.frame(Condition = ifelse(grp == 0, "N", "Y"), row.names = colnames(EXPRS))
    pheatmap(EXPRS, border_color = NA, show_rownames = FALSE, show_colnames = FALSE,
        annotation_col = annodat, color = colorRampPalette(c("red3", "black",
            "green3"))(50), annotation_colors = list(Condition = c(Y = "blue",
            N = "yellow")))
    return(length(topgenes))
}
```

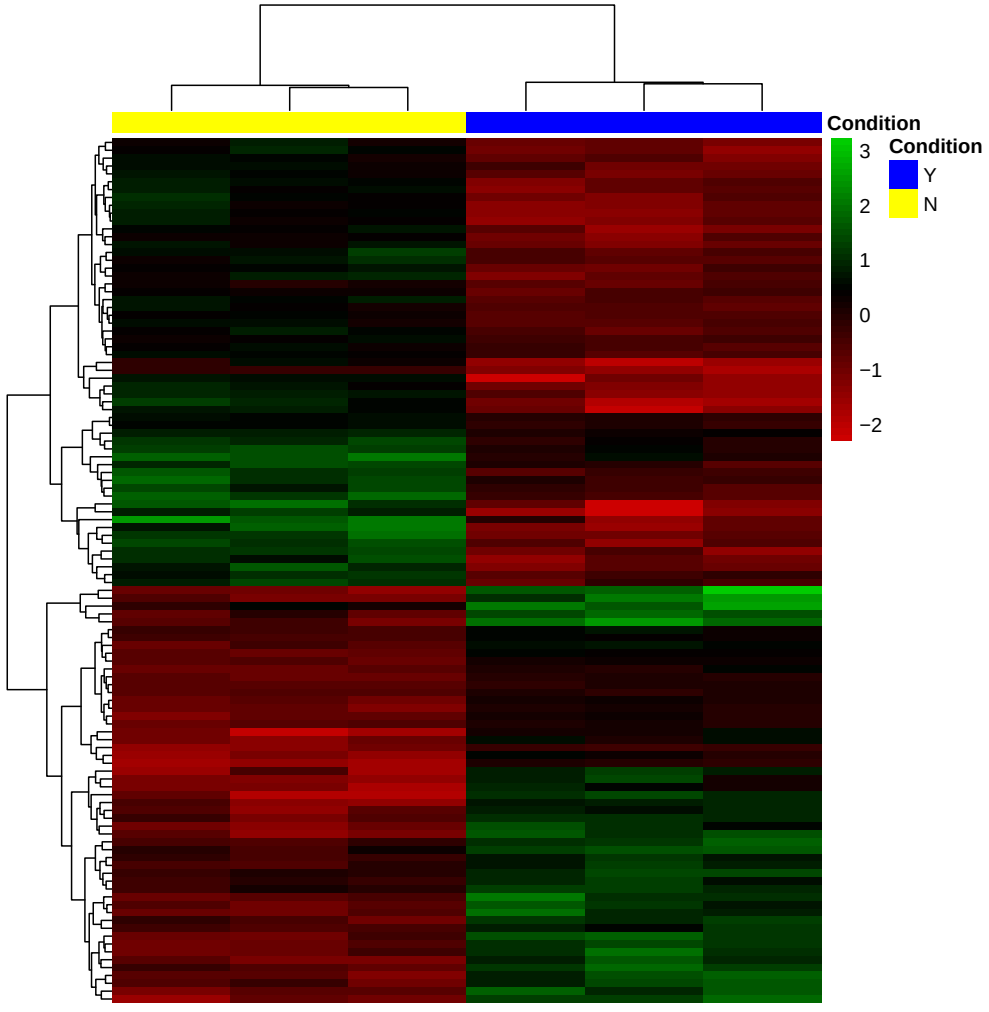
# Heatmap Example: $m = 20,000, n = 20,\ \alpha = 0.005$



# Heatmap Example: $m = 40,000, n = 20,\ \alpha = 0.0025$



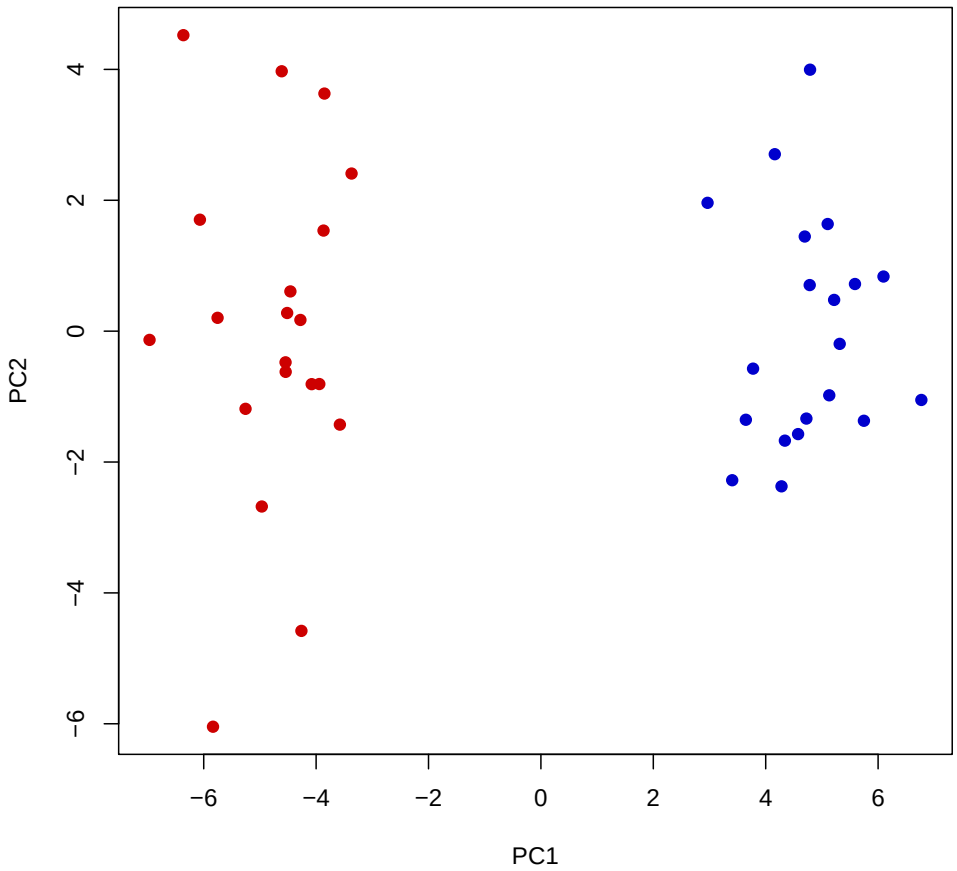
# Heatmap Example: $m = 20,000, n = 3,\ \alpha = 0.005$



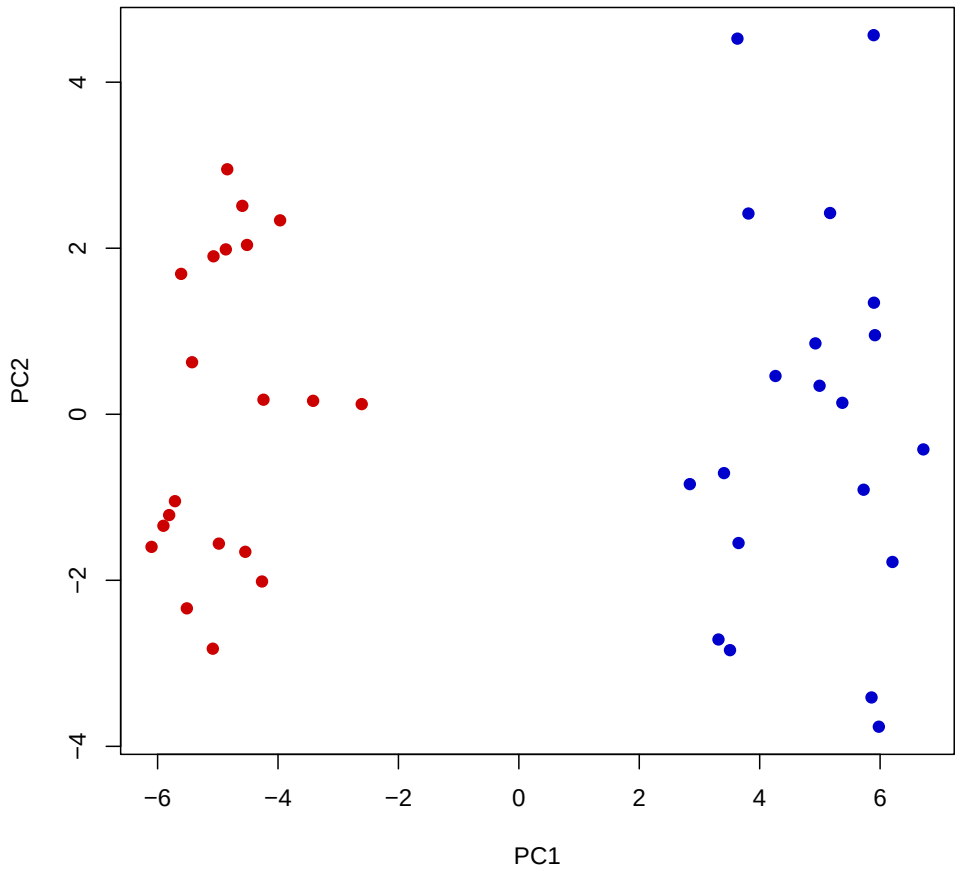
# R Code to simulate PC

```r
simulate.noise.PC = function(n, m, alpha) {
    # Simulate Expression Matrix
    EXPRS = matrix(rnorm(2 * n * m), m, 2 * n)
    grp = factor(rep(0:1, c(n, n)))
    # Get the two sample t-statistics
    pvals = rowttests(EXPRS, grp)$p.value
    topgenes = which(pvals < alpha)
    EXPRS = EXPRS[topgenes, ]
    annodat = data.frame(Condition = ifelse(grp == 0, "N", "Y"), row.names = colnames(EXPRS))
    PC = cmdscale(dist(t(EXPRS)))
    plot(PC, xlab = "PC1", ylab = "PC2", col = ifelse(grp == 0, "red3", "blue3"),
        pch = 19)
    return(length(topgenes))
}
```

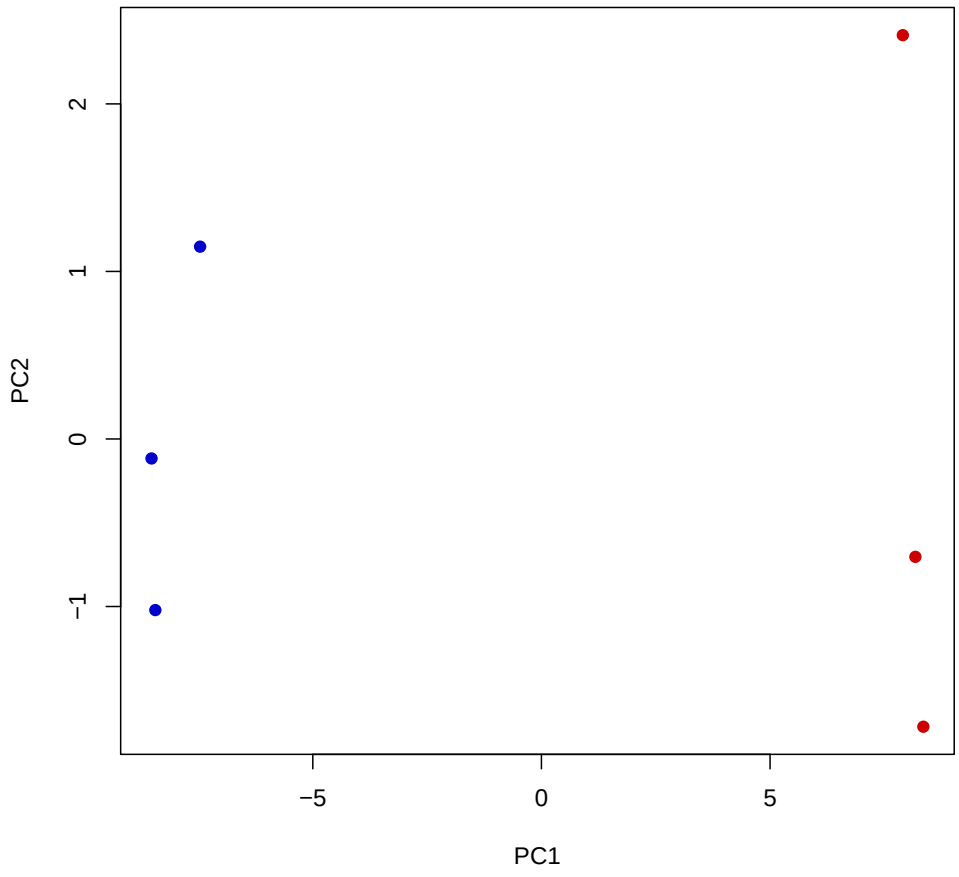
# Heatmap Example: $K = 20000, n = 20, \alpha = 0.005$



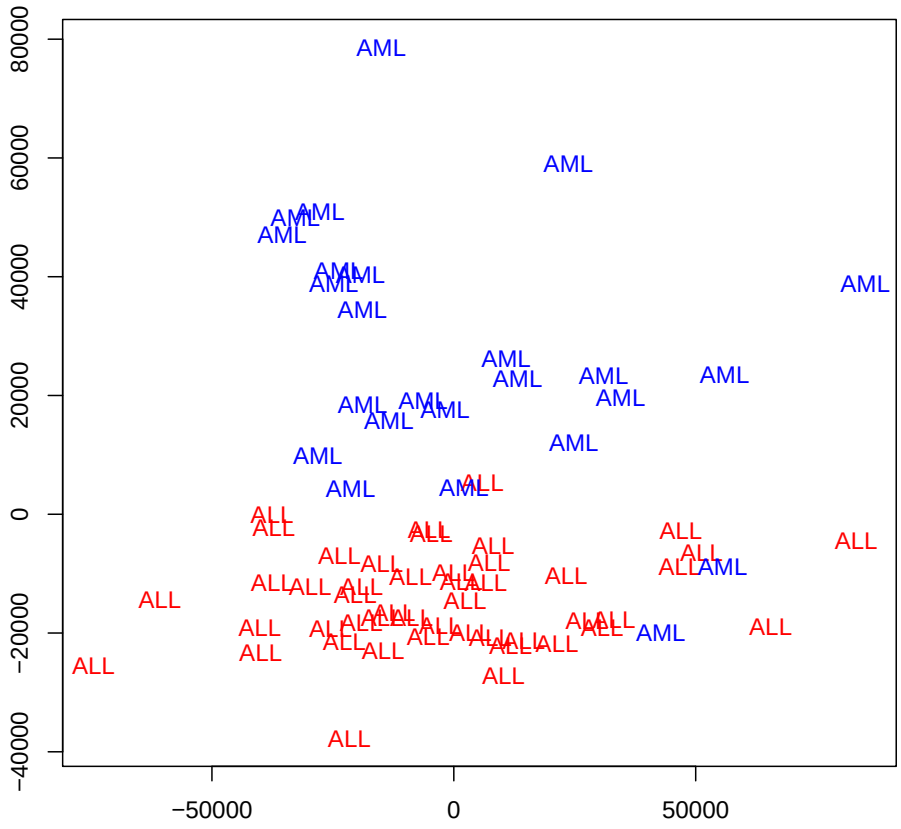
# Heatmap Example: $K = 40000, n = 20, \alpha = 0.0025$



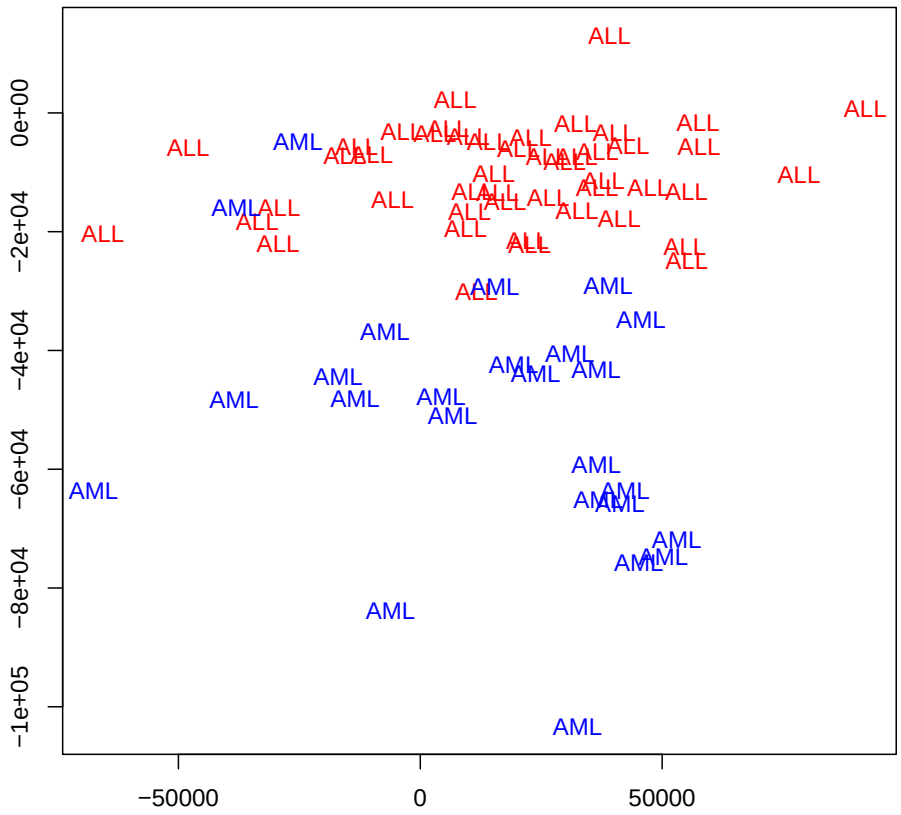
# Heatmap Example: $K = 20000, n = 3, \alpha = 0.005$



# MDS for Golub Data



# PCA for Golub Data



# Preserving The Distances

- Extract and standardize expression matrix for Golub data set

```
scexpdat = scale(t(exprs(Golub_Merge)))
dim(scexpdat)

## [1]   72 7129
```

- Check means for the first 4 genes

```
apply(scexpdat[, 1:4], 2, mean)

## AFFX-BioB-5_at AFFX-BioB-M_at AFFX-BioB-3_at AFFX-BioC-5_at
##  -7.841417e-17  -4.460287e-18   1.491832e-17  -5.051177e-17
```

- Check standard deviations for the first 4 genes

```
apply(scexpdat[, 1:4], 2, sd)

## AFFX-BioB-5_at AFFX-BioB-M_at AFFX-BioB-3_at AFFX-BioC-5_at
##              1              1              1              1
```

# Preserving The Distances

- Check distance among the first three patients

```
dist(scexpdat[1:3, ])

##          39       40
## 40 125.3402
## 42 118.1911 125.0390
```

- Calculate MDS $d = 2$

```
MDS = cmdscale(dist(scexpdat), 2)
dist(MDS[1:3, ])

##           39        40
## 40  4.644939
## 42 29.665656 34.287630
```

- Calculate MDS $d = 3$

```
MDS = cmdscale(dist(scexpdat), 3)
dist(MDS[1:3, ])

##           39        40
## 40  9.293559
## 42 45.719192 54.869668
```

# Preserving The Distances

- Check distance among the first three patients

```
dist(scexpdat[1:3, ])

##          39       40
## 40 125.3402
## 42 118.1911 125.0390
```

- Calculate MDS $d = 20$

```
MDS = cmdscale(dist(scexpdat), 3)
dist(MDS[1:3, ])

##           39        40
## 40  9.293559
## 42 45.719192 54.869668
```

- Calculate MDS $d = 45$

```
MDS = cmdscale(dist(scexpdat), 45)
dist(MDS[1:3, ])

##          39       40
## 40 124.9860
## 42 113.3668 121.7808
```

# Reminder: A Self-fulfilling Prophecy

- Statistical method for unsupervised learning guarantee one thing
- They will return a clustering of your data
- What they do not guarantee and are invariably unable to verify, is the biological relevance or reproducibility of the clustering
- In light of this Self-fulfilling Prophecy, these methods should be used with utmost care