Hierarchical clustering

Rebecca C. Steorts, Duke University

STA 325, Chapter 10 ISL

globally set figure width and height and cache

```
knitr::opts_chunk$set(fig.width=5, fig.height=4,
                      cache=TRUE)
```

Getting Sweave Working in R (Datathon project)

1. Open R Studio
2. Go to RStudio (click Preferences)
3. Go to Sweave
4. Weave Rnw files using knitr (not Sweave)
5. Hit Apply!

Check: See that you're files are caching.

Agenda

- K-means versus Hierarchical clustering
- Agglomerative vs divisive clustering
- Dendogram (tree)
- Hierarchical clustering algorithm
- Single, Complete, and Average linkage
- Application to genomic (PCA versus Hierarchical clustering)

From K-means to Hierarchical clustering

Recall two properties of K-means clustering:

1. It fits exactly K clusters (as specified)
2. Final clustering assignment depends on the chosen initial cluster centers

- Assume pairwise dissimilarites d_{ij} between data points.
- Hierarchical clustering produces a consistent result, without the need to choose initial starting positions (number of clusters).

Catch: choose a way to measure the dissimilarity between groups, called the linkage

- Given the linkage, hierarchical clustering produces a sequence of clustering assignments.
- At one end, all points are in their own cluster, at the other end, all points are in one cluster

Agglomerative vs divisive clustering

Agglomerative (i.e., bottom-up):

- Start with all points in their own group
- Until there is only one cluster, repeatedly: merge the two groups that have the smallest dissimilarity

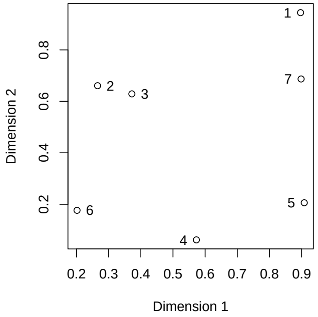
Divisive (i.e., top-down):

- Start with all points in one cluster
- Until all points are in their own cluster, repeatedly: split the group into two resulting in the biggest dissimilarity

Agglomerative strategies are simpler, we'll focus on them. Divisive methods are still important, but you can read about these on your own if you want to learn more.

Simple example

Given these data points, an agglomerative algorithm might decide on a clustering sequence as follows:



Step 1: $\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\}$;

Step 2: $\{1\},\{2,3\},\{4\},\{5\},\{6\},\{7\}$;

Step 3: $\{1,7\},\{2,3\},\{4\},\{5\},\{6\}$;

Step 4: $\{1,7\},\{2,3\},\{4,5\},\{6\}$;

Step 5: $\{1,7\},\{2,3,6\},\{4,5\}$;

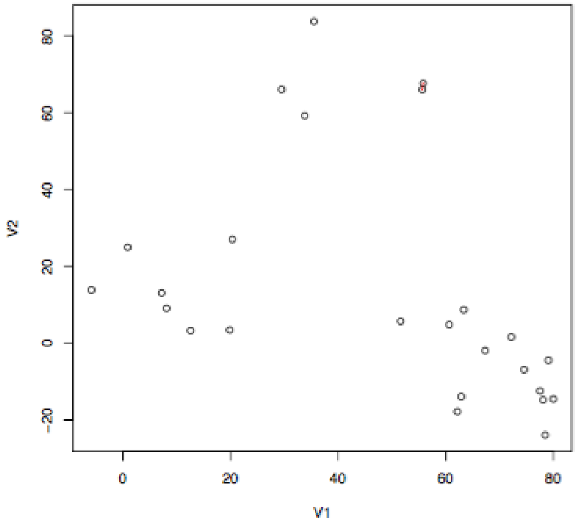
Step 6: $\{1,7\},\{2,3,4,5,6\}$;

Step 7: $\{1,2,3,4,5,6,7\}$.

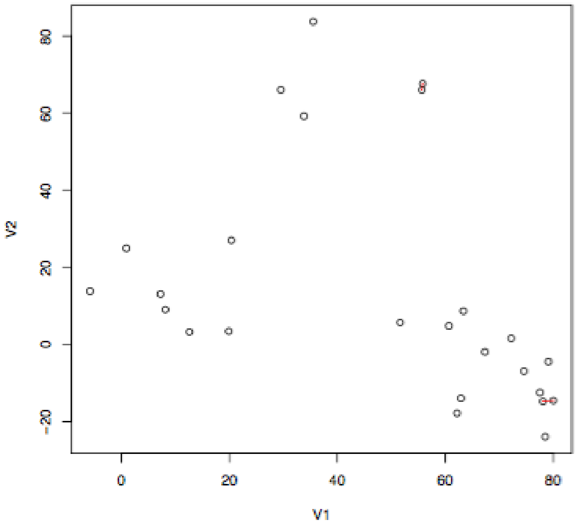
Algorithm

1. Place each data point into its own singleton group.
2. Repeat: iteratively merge the two closest groups
3. Until: all the data are merged into a single cluster

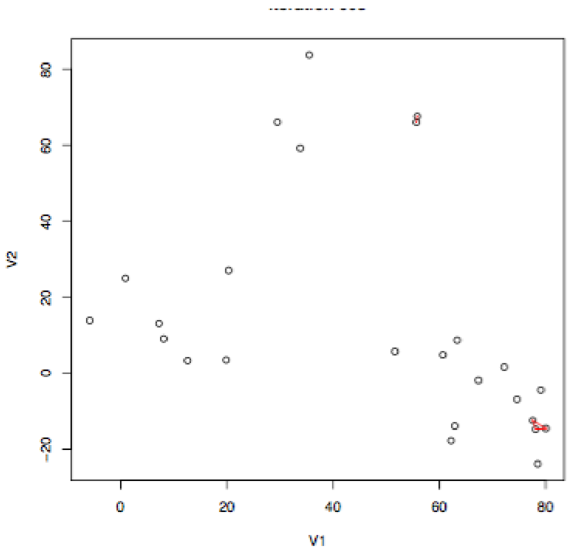
Example



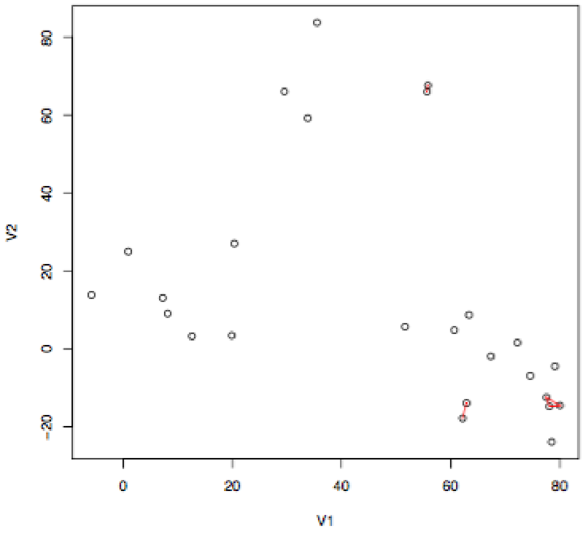
Iteration 2



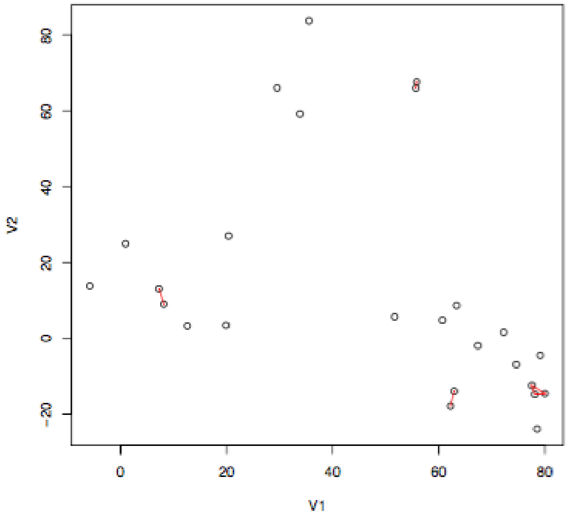
Iteration 3



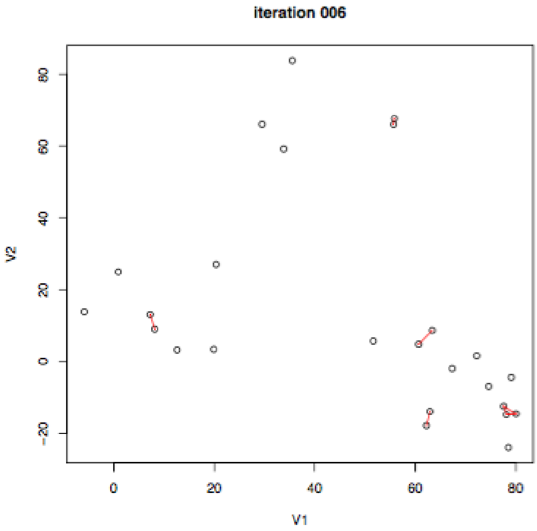
Iteration 4



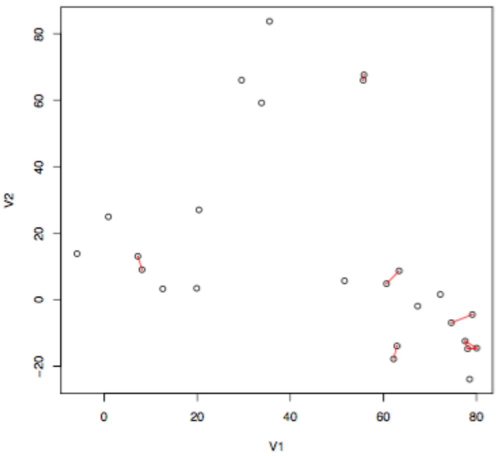
Iteration 5



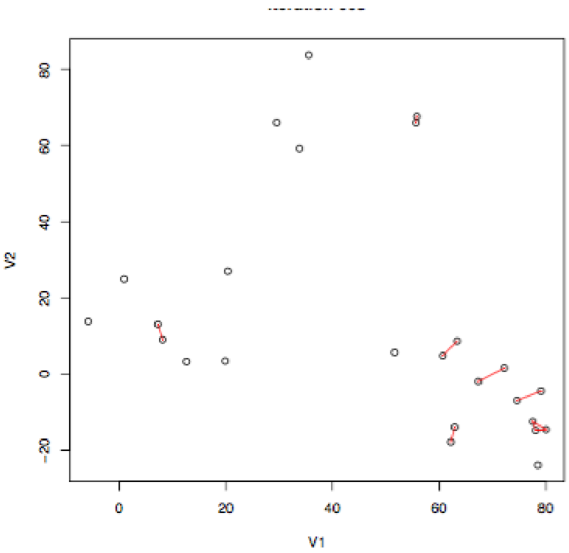
Iteration 6



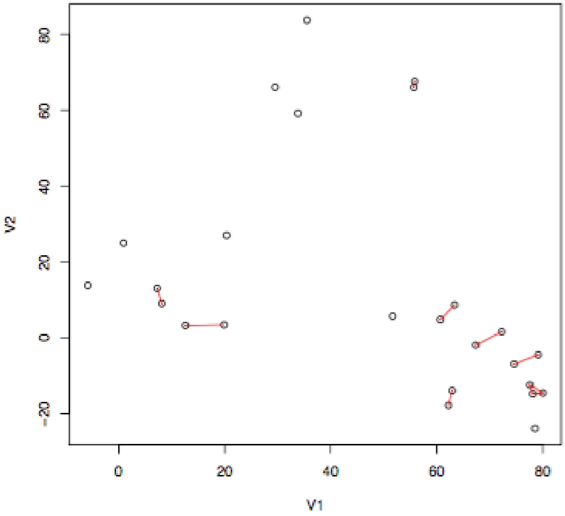
Iteration 7



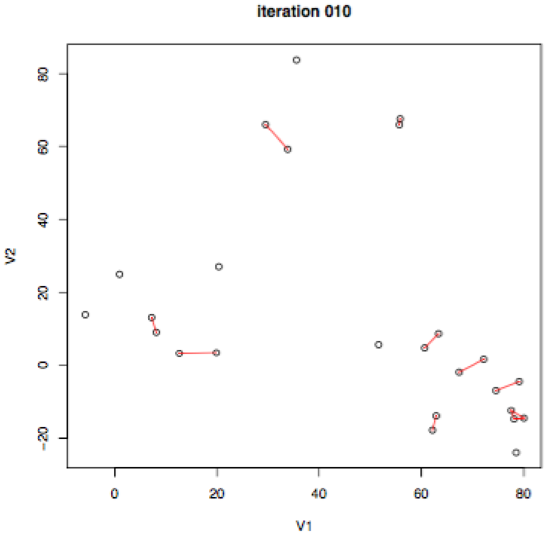
Iteration 8



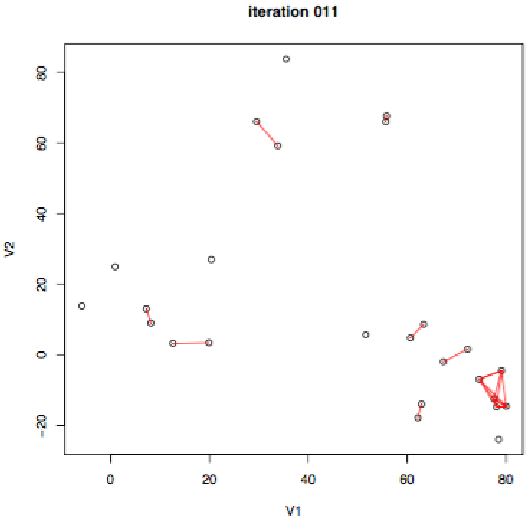
Iteration 9



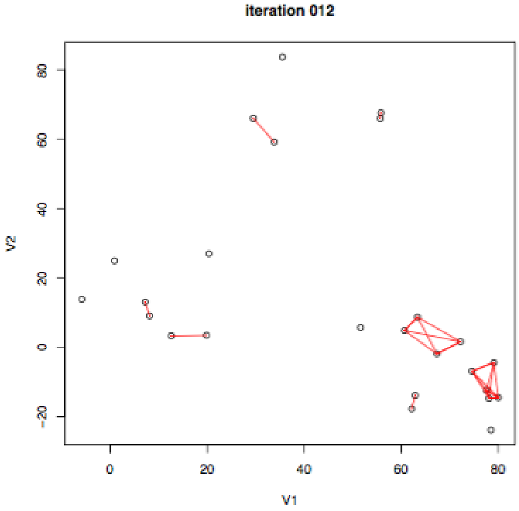
Iteration 10



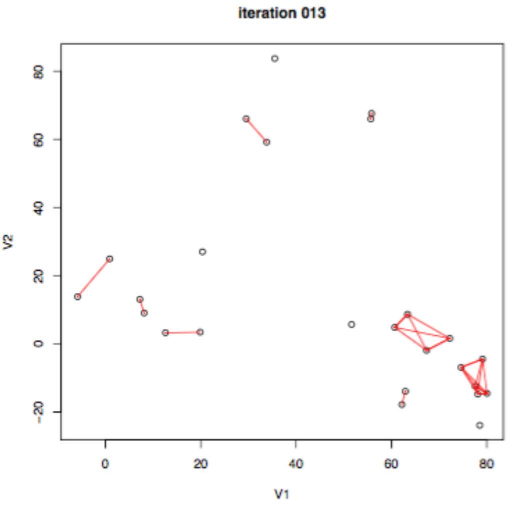
Iteration 11



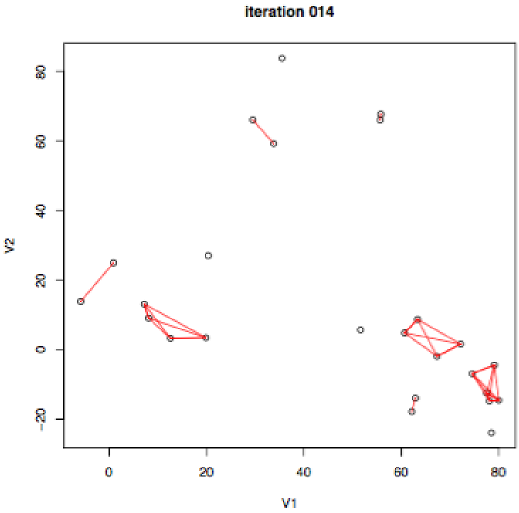
Iteration 12



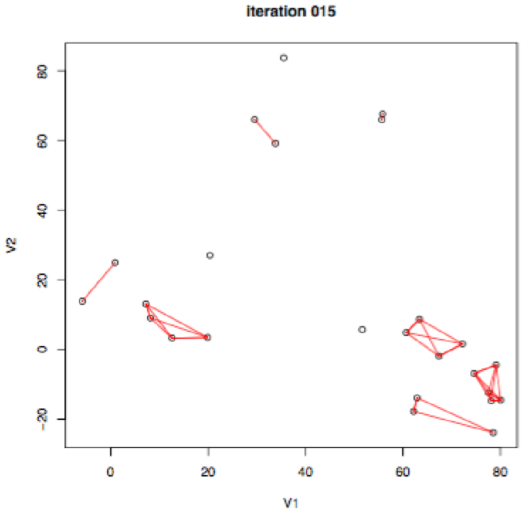
Iteration 13



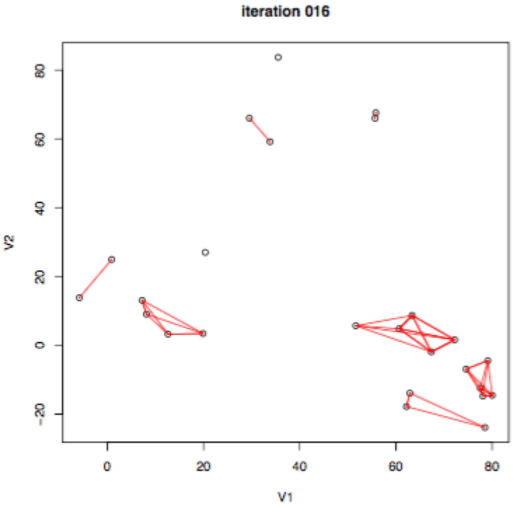
Iteration 14



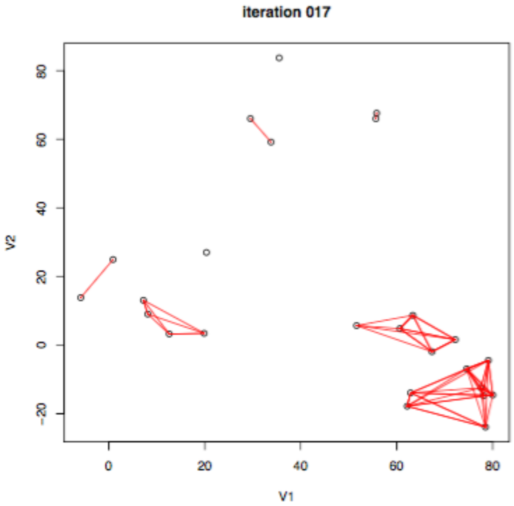
Iteration 15



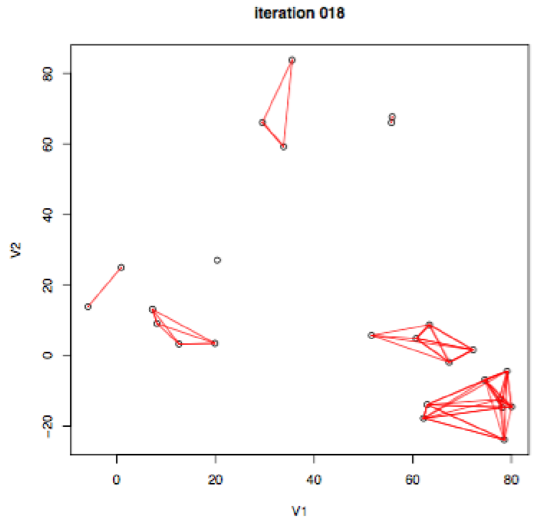
Iteration 16



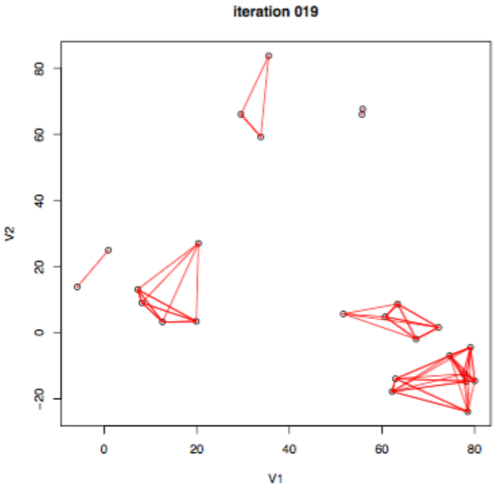
Iteration 17



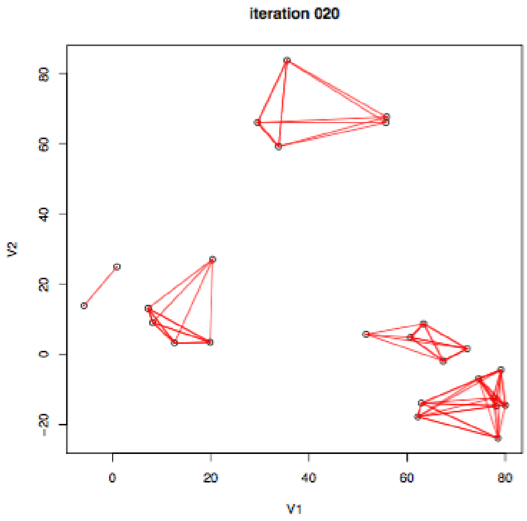
Iteration 18



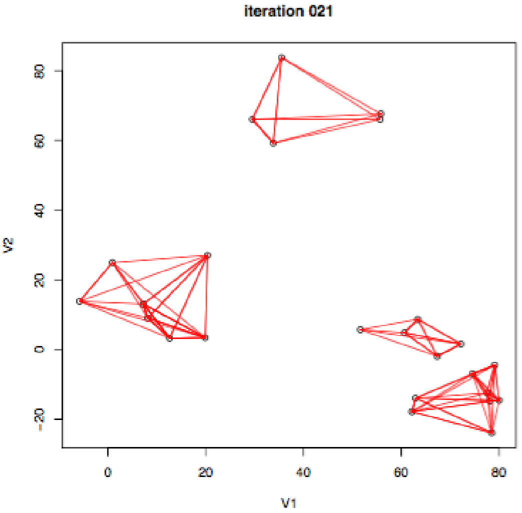
Iteration 19



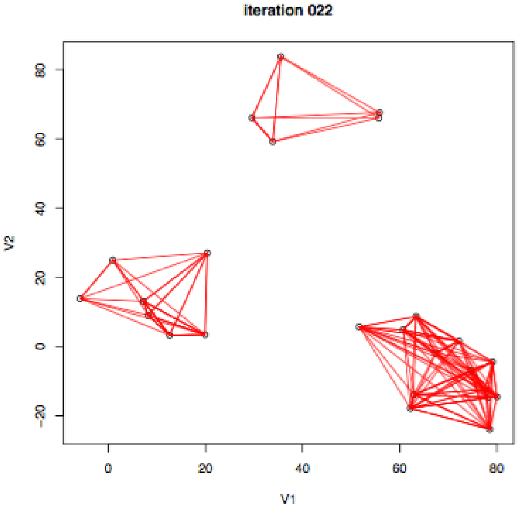
Iteration 20



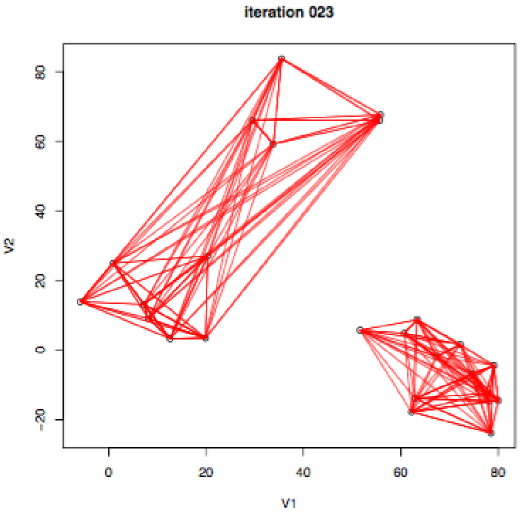
Iteration 21



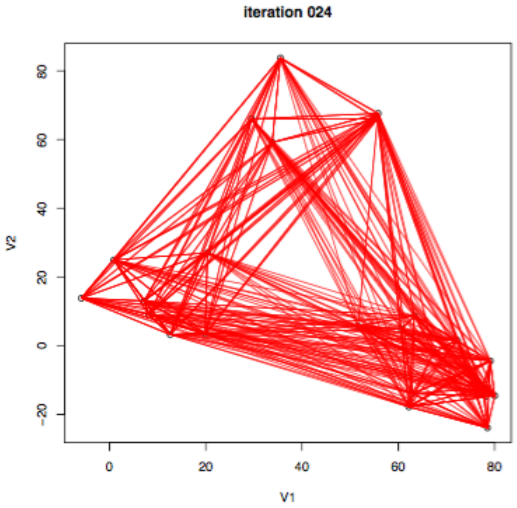
Iteration 22



Iteration 23



Iteration 24



Clustering

Suppose you are using the above algorithm to cluster the data points in groups.

- How do you know when to stop?
- How should we compare the data points?

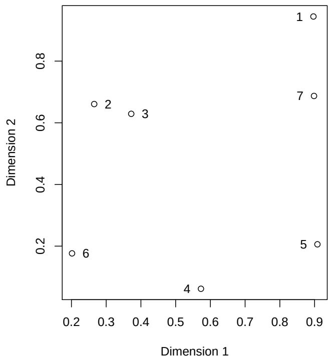
Let's investigate this further!

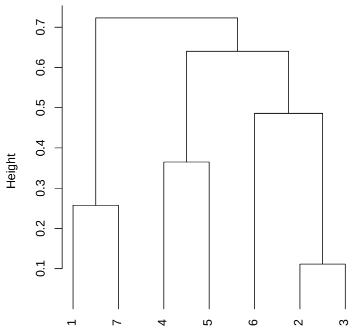
Agglomerative clustering

- Each level of the resulting tree is a segmentation of the data
- The algorithm results in a sequence of groupings
- It is up to the user to choose a “natural” clustering from this sequence

Dendogram

We can also represent the sequence of clustering assignments as a dendrogram:





Note that cutting the dendrogram horizontally partitions the data points into clusters

Dendogram

- Agglomerative clustering is monotonic
- The similarity between merged clusters is monotone decreasing with the level of the merge.[1]
- Dendrogram: Plot each merge at the (negative) similarity between the two merged groups
- Provides an interpretable visualization of the algorithm and data
- Useful summarization tool, part of why hierarchical clustering is popular

[1]A function that is increasing or decreasing at some point is called monotone at that point.

Group similarity

Given a distance similarity measure (say, Eucliclean) between points, the user has many choices on how to define intergroup similarity.

1. Single linkage: the similiarity of the closest pair

$$d_{SL}(G,H) = \min_{i \in G, j \in H} d_{i,j}$$

2. Complete linkage: the similarity of the furthest pair

$$d_{CL}(G,H) = \max_{i \in G, j \in H} d_{i,j}$$

3. Group-average: the average similarity between groups

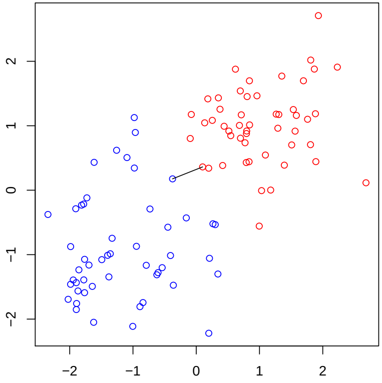
$$d_{GA} = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{j \in H} d_{i,j}$$

Single Linkage

In single linkage (i.e., nearest-neighbor linkage), the dissimilarity between G, H is the smallest dissimilarity between two points in opposite groups:

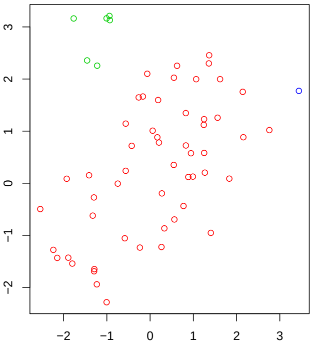
$$d_{\text{single}}(G, H) = \min_{i \in G, j \in H} d_{ij}$$

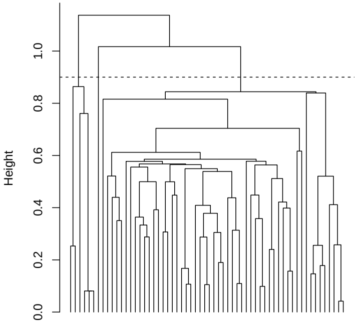
Example (dissimilarities d_{ij} are distances, groups are marked by colors): single linkage score $d_{\text{single}}(G, H)$ is the distance of the closest pair



Single Linkage Example

Here $n = 60$, $X_i \in \mathbb{R}^2$, $d_{ij} = \|X_i - X_j\|_2$. Cutting the tree at $h = 0.9$ gives the clustering assignments marked by colors





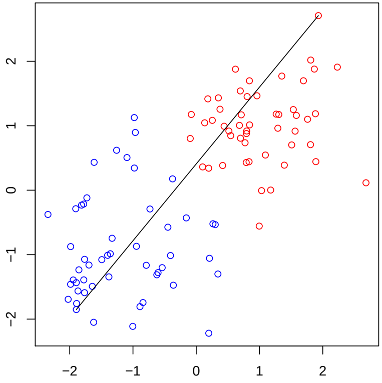
Cut interpretation: for each point X_i, there is another point X_j in its cluster with $d_{ij} \leq 0.9$

Complete Linkage

In complete linkage (i.e., furthest-neighbor linkage), dissimilarity between G, H is the largest dissimilarity between two points in opposite groups:

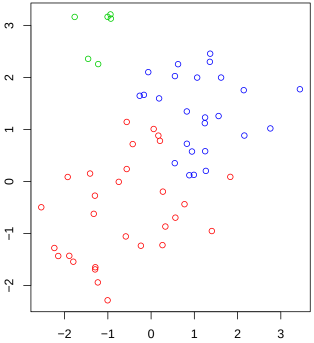
$$d_{\text{complete}}(G, H) = \max_{i \in G,\, j \in H} d_{ij}$$

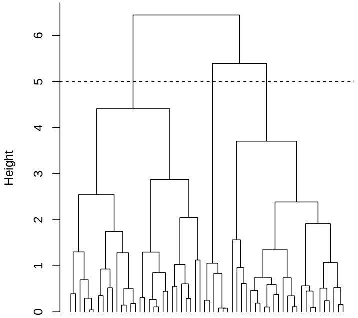
Example (dissimilarities d_{ij} are distances, groups are marked by colors): complete linkage score $d_{\text{complete}}(G, H)$ is the distance of the furthest pair



Complete Linkage Example

Same data as before. Cutting the tree at $h = 5$ gives the clustering assignments marked by colors





Cut interpretation: for each point X_i, every other point X_j in its cluster satisfies $d_{ij} \leq 5$

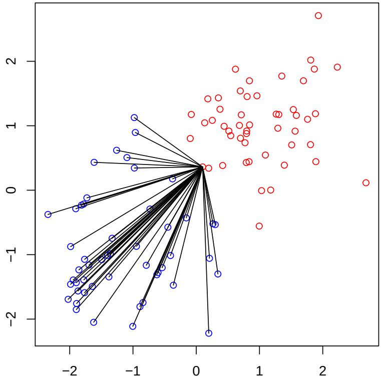
Average Linkage

In average linkage, the dissimilarity between G, H is the average dissimilarity over all points in opposite groups:

$$d_{\text{average}}(G, H) = \frac{1}{n_G \cdot n_H} \sum_{i \in G, j \in H} d_{ij}$$

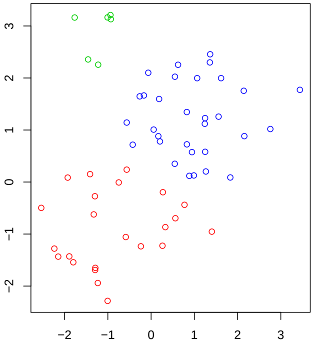
Example (dissimilarities d_{ij} are distances, groups are marked by colors): average linkage score $d_{\text{average}}(G, H)$ is the average distance across all pairs

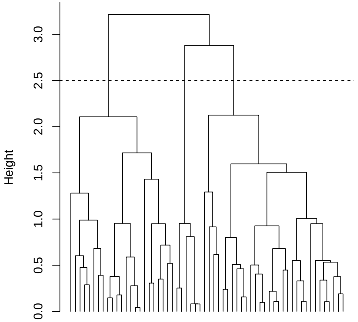
(Plot here only shows distances between the blue points and one red point)



Average linkage example

Same data as before. Cutting the tree at $h = 2.5$ gives clustering assignments marked by the colors





Cut interpretation: there really isn't a good one!

Properties of intergroup similarity

- Single linkage can produce “chaining,” where a sequence of close observations in different groups cause early merges of those groups
- Complete linkage has the opposite problem. It might not merge close groups because of outlier members that are far apart.
- Group average represents a natural compromise, but depends on the scale of the similarities. Applying a monotone transformation to the similarities can change the results.

Things to consider

- Hierarchical clustering should be treated with caution.
- Different decisions about group similarities can lead to vastly different dendrograms.
- The algorithm imposes a hierarchical structure on the data, even data for which such structure is not appropriate.

Application on genomic data

- Unsupervised methods are often used in the analysis of genomic data.
- PCA and hierarchical clustering are very common tools. We will explore both on a genomic data set.
- We illustrate these methods on the NCI60 cancer cell line microarray data, which consists of 6,830 gene expression measurements on 64 cancer cell lines.

Background on gene expression levels

A gene is a stretch of DNA inside the cell that tells the cell how to make a specific protein.

All cells in the body contain the same genes, but they do not always make the same proteins in the same quantities

The genes have different expression levels in different cell types, and cells can regulate gene expression levels in response to their environment.

Background on gene expression levels

Different types of cells thus have different expression profiles.

Many diseases, including cancer, fundamentally involve breakdowns in the regulation of gene expression.

The expression profile of cancer cells becomes abnormal, and different kinds of cancers have different expression profiles.[2]

[2] The exception are red blood cells, which do not contain DNA.

Gene expression data

Our data are gene expression measurements from cells drawn from 64 different tumors (from 64 different patients).

In each case, a device called a microarray (or gene chip) measured the expression of each of 6830 distinct genes.

Class types of gene expression data

The cells mostly come from known cancer types, so there are classes, in addition to the measurements of the expression levels.

The classes are breast, cns (central nervous system), colon, leukemia, melanoma, nsclc (non-small- cell lung cancer), ovarian, prostate, renal, K562A, K562B, MCF7A, MCF7D (those four are laboratory tumor cultures) and unknown.

Application on gene expression data

```r
library(ElemStatLearn)
data(nci)
head(nci)
```

```
##           CNS      CNS    CNS        RENAL BREAST    CNS
## [1,]  0.300 0.679961  0.940  2.800000e-01  0.485  0.310
## [2,]  1.180 1.289961 -0.040 -3.100000e-01 -0.465 -0.030
## [3,]  0.550 0.169961 -0.170  6.800000e-01  0.395 -0.100
## [4,]  1.140 0.379961 -0.040 -8.100000e-01  0.905 -0.460
## [5,] -0.265 0.464961 -0.605  6.250000e-01  0.200 -0.205
## [6,] -0.070 0.579961  0.000 -1.387779e-17 -0.005 -0.540
##         NSCLC  NSCLC  RENAL  RENAL  RENAL  RENAL  RENAL
## [1,]  0.460  0.760  0.270 -0.450 -0.030  0.710 -0.360 -
## [2,]  0.000  1.490  0.630 -0.060 -1.120  0.000 -1.420 -
## [3,]  1.150  0.280 -0.360  0.150 -0.050  0.160 -0.030 -
## [4,] -1.400  0.100 -1.040 -0.610  0.000 -0.770 -2.280 -
## [5,] -0.005 -0.525  0.015 -0.395 -0.285  0.045  0.135 -
## [6,] -0.700  0.360 -0.040  0.150 -0.250 -0.160 -0.320
```

Task 1: Hierarchical Clustering to NCI60 Data

Produce dendrograms using single-link clustering and complete-link clustering, and average link clustering. Make sure the figures are legible. (Try the cex=0.5 option to plot.)

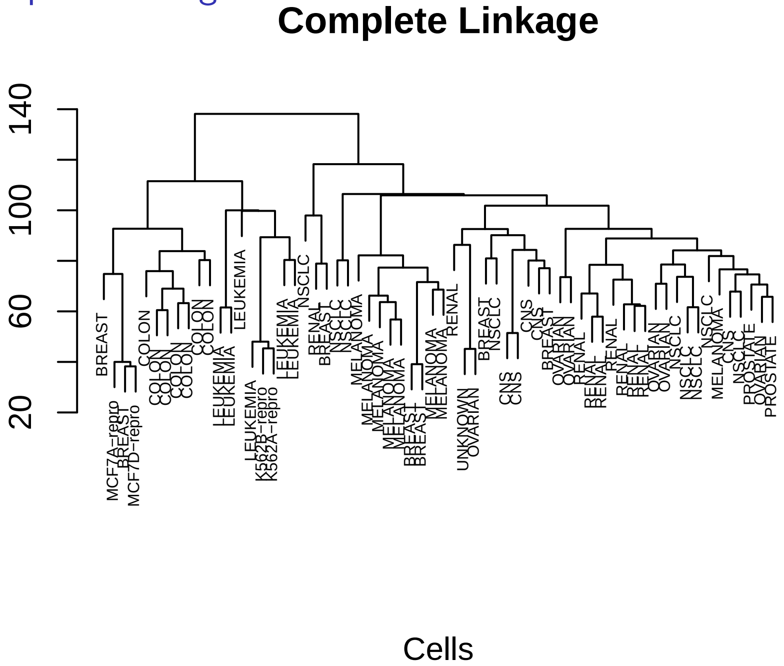
Remark: hclust needs a matrix of distances, handily produced by the dist function, so let's make sure to remember to use this.

Creating distances

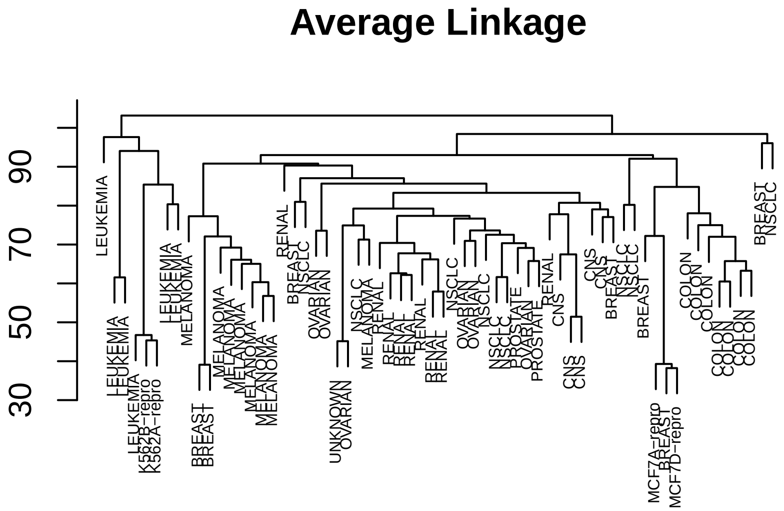
```r
# does euc. dist by default
nci.dist <- dist(nci.t)
head(nci.dist)
```

```
## [1] 51.43823 65.93815 79.87886 92.65185 80.36544 81.031
```

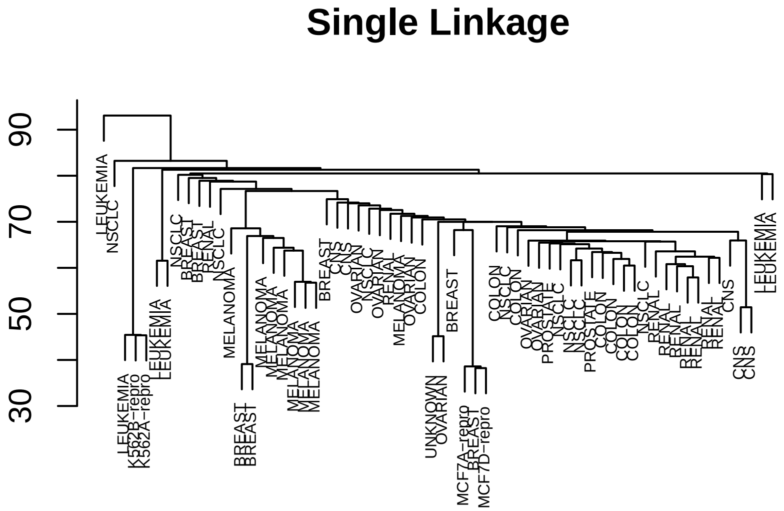
Complete Linkage



Average Linkage



Single Linkage



Task 2

Which cell classes seem are best captured by each clustering method?

1. Complete-linkage has nice sub-trees for COLON, LEUKEMIA, MELANOMA, and RENAL.
2. Average linkage has similar resuls to complete linkage.
3. Single-linkage is good with RENAL and decent with MELANOMA (though confused with BREAST). There are little sub-trees of cells of the same time, like COLON or CNS, but mixed together with others.

Complete and average linkage versus single linkage

- Typically, single linkage will tend to yield trailing clusters: very large clusters onto which individual observations attach one-by-one.
- On the other hand, complete and average linkage tend to yield more balanced, attractive clusters
- For this reason, complete and average linkage are generally preferred to single linkage.

Task 3: Height and sum of the within-cluster sums of squares relationship

The hclust command returns an object whose height attribute is the sum of the within-cluster sums of squares. How many clusters does this suggest we should use, according to complete linkage?

Height and sum of the within-cluster sums of squares relationship

```
nci.complete = hclust(nci.dist, method="complete")
length(nci.complete$height)
```

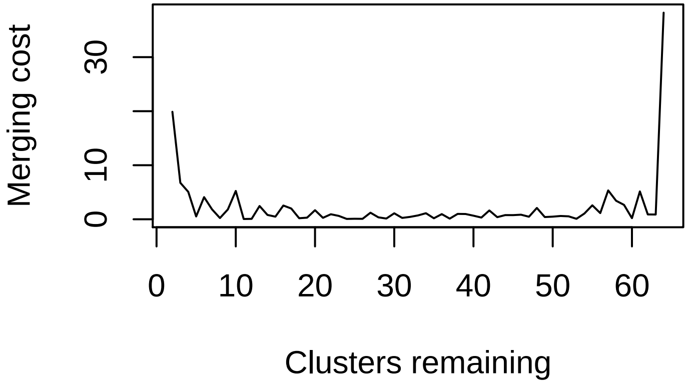
```
## [1] 63
```

```
nci.complete$height[1]
```

```
## [1] 38.23033
```

There are 63 “heights”, corresponding to the 63 joinings, i.e., not including the height (sum-of-squares) of zero when we have 64 clusters, one for each data point. Since the merging costs are differences, we want to add an initial 0 to the sequence.

Optimal number of clusters



The heuristic is to stop joining clusters when the cost of doing so goes up a lot.

This suggests using only 10 clusters, since going from 10 clusters to 9 is very expensive. (Alternately, use 63 clusters; but that would be silly.)

Task 4

Suppose you did not know the cell classes. Can you think of any reason to prefer one clustering method over another here?