Noah L. Schrick - 1492657

Hierarchical clustering (upgma and linkage matrix). The goal of the lab is to get experience using hierarchical clustering, better understand how trees are constructed, and how R's hclust object works. We will also illustrate WGCNA clustering and multidimensional scaling. Include plots in your Word doc report and use a color to indicate your answers.

A. Background on the inner workings of hclust. hclust is an R function that takes a distance matrix as input and performs the UPGMA merging algorithm when you specify `method="average"` (average linkage); hclust does other linkages too. It returns the tree encoded as a 2-column linkage matrix, described more below. The hclust object can then be plotted with the plot function, which knows how to plot a tree given an hclust object.

Recall the distance matrix below from lecture. We manually constructed its UPGMA tree during class: (((A-B)-C)-(D-E))-F.

	A	B	C	D	E	F
A	x	2	4	6	6	8
B		x	4	6	6	8
C			x	6	6	8
D				x	3	8
E					x	8
F						x

1. Based on what we found in lecture, write a 2-column linkage matrix in Word using the following hclust rules.

a. Label the object A, B, C, D, E, F as negative numbers -1, -2, -3, -4, -5, -6.

b. Put the closest pair of objects in the first row of the matrix (this merged node will be referenced as 1 later in the merge matrix). Since the closest pair are A and B, the first row is

```
-1 -2
```

c. Next we will merge AB (1) with C (-3), and this merger will be referenced as +2 later in the merge matrix).

```
-1 -2
 1 -3
```

d. Repeat until you finish the tree.

Show the full merge matrix and the heights of each row in the merge matrix.

	A	B	C	D	E	F
A	x	**2**	4	6	6	8
B		x	4	6	6	8
C			x	6	6	8
D				x	3	8
E					x	8
F						x

d(AB,C) = (d(A,C) + d(B,C))/|AB||C| = (4 + 4)/(2*1) = 4
d(AB,D) = (d(A,D) + d(B,D))/|AB||D| = (6 + 6)/(2*1) = 6
d(AB,E) = (d(A,E) + d(B,E))/|AB||E| = (6 + 6)/(2*1) = 6
d(AB,F) = (d(A,F) + d(B,F))/|AB||F| = (8 + 8)/(2*1) = 8

Merge -1 and -2 => (+1) (Merge A and B)
Height of 2

	AB	C	D	E	F
AB	x	4	6	6	8
C		x	6	6	8
D			x	**3**	8
E				x	8
F					x

d(AB,DE) = (d(A,D) +d(A,E) + d(B,D) +d(B,E))/|AB||DE| = (6+6+6+6)/(2*2) = 6
d(C,DE) = (d(C,D) + d(C,E))/|C||DE| = (6 + 6)/(1*2) = 6
d(DE,F) = (d(D,F) + d(E,F))/|DE||F| = (8 + 8)(2*1) = 8

Merge -3 and -4 => (+1) (Merge D and E)
Height of 3

	AB	C	DE	F
AB	x	**4**	6	8
C		x	6	8
DE			x	8
F				x

d(ABC,DE) = (d(A,D) +d(A,E) + d(B,D) +d(B,E) +d(C,D) +d(C,E))/|ABC||DE| = (6+6+6+6+6+6)/(3*2) = 6
d(ABC,F) = (d(A,F) +d(B,F) + d(C,F))/|ABC||F| = (8+8+8)/(3*1) = 8

Merge +1 with -3 => (+4) (Merge AB and C)
Height of 4

	ABC	DE	F
ABC	x	**6**	8
DE		x	8
F			x

d(ABCDE,F) = (d(A,F) +d(B,F) + d(C,F) +d(D,F) +d(E,F))/|ABCDE||F| = (8+8+8+8+8/(5*1) = 8

Merge +4 with +1 => (+3) (Merge ABC and DE)
Height of 6

	ABCDE	F
ABCDE	x	**8**
F		x

Merge +3 with -5 => (-2) (Merge ABCDE and F)
Height of 8

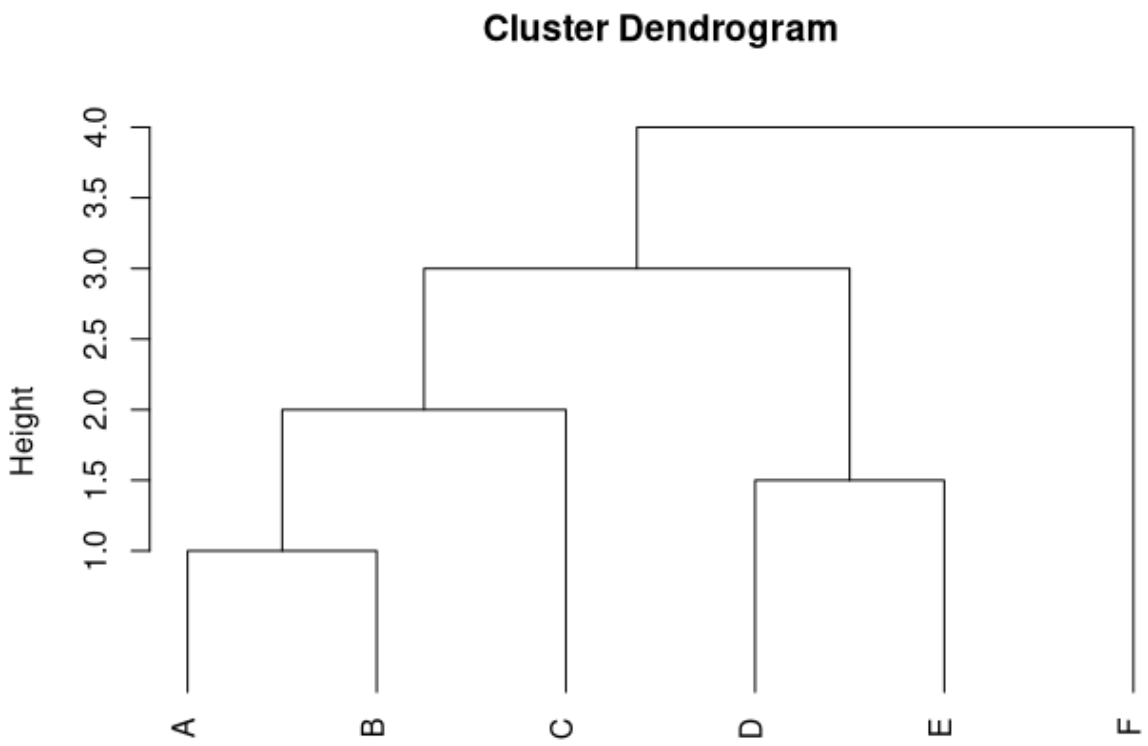
Linkage Matrix

-1	**-2**	**# (1) First Merge (A,B)**
1	**-3**	**# (2) Second Merge (AB, C)**
-4	**-5**	**# (3) Third Merge (D,E)**
2	**3**	**# (4) Fourth Merge (ABC, DE)**
4	**-6**	**# (5) Fifth Merge (ABCDE, F)**

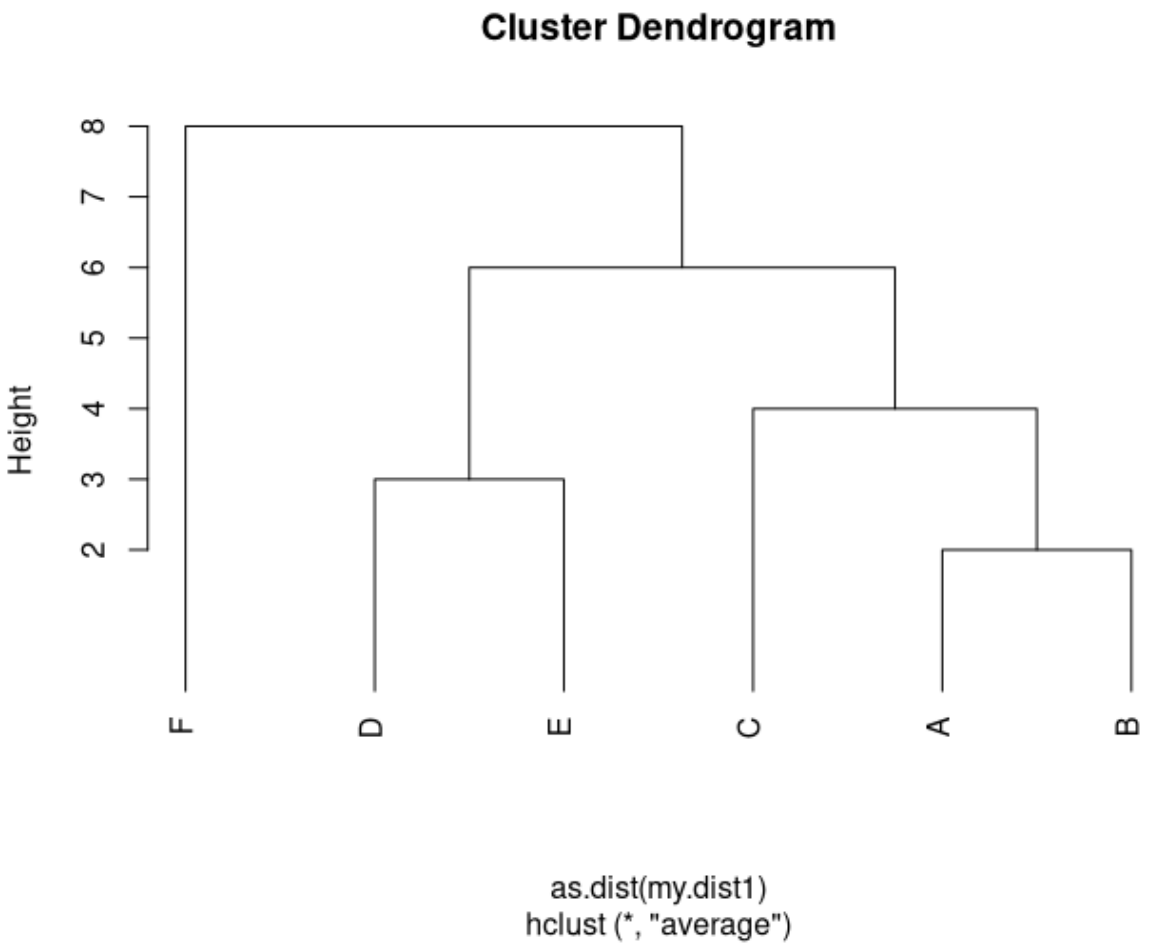
2. Create a script with the following *incomplete* code. Based on your UPGMA merging calculation and your linkage matrix above, fill in the blanks in the code below to plot the tree. Show the resulting plot.

```
# initialize an empty list that will contain fields of our hclust object
a <- list()
#    encoding rules:
#    negative numbers are leaves (A,B,...,E) -> (-1,-2,...,-5)
#    positive are merged clusters (defined by row number in $merge)

# each merged pair in a row
a$merge <- rbind(c(-1, -2),   # +1 (A-B)
                 c( 1, -3),   # +2 (A-B)-C
                 c(-4, -5),   # +3 (D-E)
                 c( 2,  3), # +4 ((A-B)-C)-(DE)
                 c( 4, -6))     # +5 (((A-B)-C)-(DE))-F

a$height <- c(1, 2, 1.5, 3, 4) # merge heights
a$order <- 1:6                 # order of leaves
a$labels <- LETTERS[1:6]       # labels of leaves as Letters
```

```
class(a) <- "hclust"          # force a to be hclust object
# plot the tree
# plot knows that a is an hclust object and plots accordingly
plot(a,hang=-1) # or use plot(as.dendrogram(a))
```



B. UPGMA from distance matrix computed by built-in **hclust** and the **WGCNA** (Weighted Gene Correlation Network Analysis) tool. Given a distance matrix and a linkage tree, WGCNA attempts to find the best clusters. Clusters are objects that are more closely related to each other than with objects outside the cluster (like a clade). Add the code below to your script to read in the distance matrix `distance_matrix.txt` (from Harvey) and plot the dendrogram. Show the dendrogram. **3.** View the distance matrix in Excel and paste it in the Word doc. Look up help on `read.table` and look at `my.dist1` to tell what `header=T` does. **4.** What does `header=T` do in this situation (see what my.dist1 looks like with header=T and header=F)?



3.)

A	B	C	D	E	F
999	2	4	6	6	8
999	999	4	6	6	8
999	999	999	6	6	8
999	999	999	999	3	8
999	999	999	999	999	8
999	999	999	999	999	999

4.)

Header in this situation adds the letters A, B, C, D, E, and F at the top of each column.

```
# Set the Working Directory First
```

```
my.dist1 <- read.table("distance_matrix.txt",sep="\t", header=T)
rownames(my.dist1) = colnames(my.dist1)
my.dist1[lower.tri(my.dist1)] = t(my.dist1)[lower.tri(my.dist1)]
hc1 <- hclust(as.dist(my.dist1), method="average")
plot(hc1,hang=-1)
```

Add the following code to your script to install and run WGCNA. **5.** Looking at `dynamicMods`, what leaves are clustered together?

5.)

A B C D E F
1 1 1 2 2 0

A, B, and C are clustered together, D and E are clustered together, and F is a singleton.

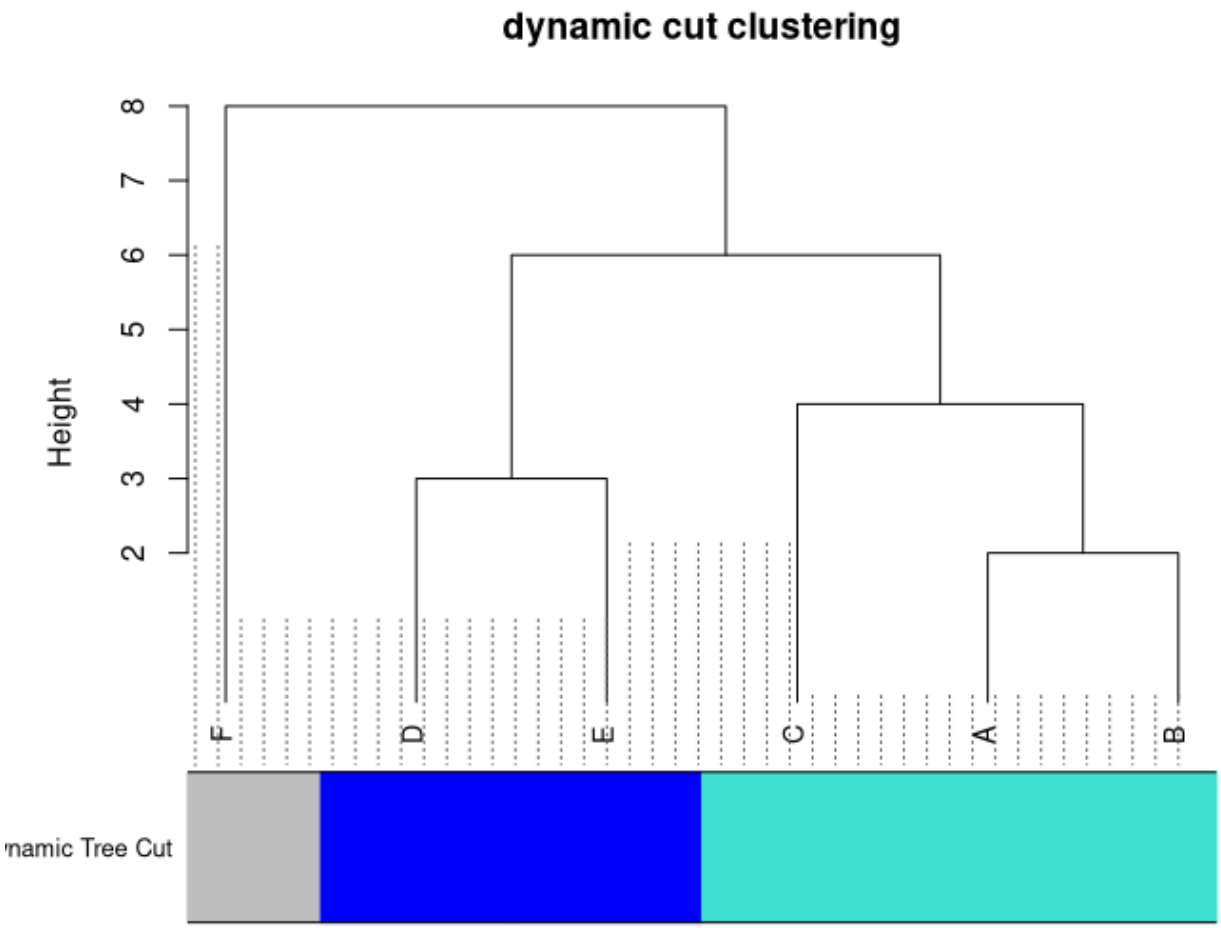
Installing WGCNA. For **Mac**, first install XCode from the App Store (if not installed already – try to run the commands below and see if it works). Installing XCode might also require you to upgrade your Mac OS. For **Windows**, you might need to install RTools outside of RStudio (it is a separate program). Download, unzip and install RTools. Then in RStudio, run `install.packages("devtools")`. Now (in Windows or Mac) try one of the following installation methods in RStudio.

```
    # method 1
    install.packages("WGCNA", dependencies=T)
    source("http://bioconductor.org/biocLite.R")
    biocLite(c("AnnotationDbi", "impute", "GO.db", "preprocessCore"))
    biocLite("WGCNA")
    install.packages("WGCNA")

    # method 2 (easier method)
    install.packages("BiocManager")
    library(BiocManager)
    BiocManager::install("WGCNA")

# load WGCNA and run on the hc1 dendrogram
cutree(hc1,k=3)  # hclust cluster labels with k=3 clusters
library(WGCNA)
dynamicMods = cutreeDynamic(dendro = hc1, distM = my.dist1,
                            deepSplit = 2, pamRespectsDendro = FALSE,
                            minClusterSize = 2)
names(dynamicMods) <- colnames(my.dist1)
```

Add the following code to show the clustered data. **6.** What do the colors at the bottom represent?

```
dynamicColors = labels2colors(dynamicMods) # colors might not work
```

```
table(dynamicColors)
# Plot the dendrogram and colors underneath
sizeGrWindow(8,6)
plotDendroAndColors(hc1, dynamicColors, "Dynamic Tree Cut",
                    dendroLabels = NULL, hang = -1,
                    addGuide = TRUE, #guideHang = 0.05,
                    main = "dynamic cut clustering")
```



6.)

The colors represent the clustering of leaf nodes. In this case, the cluster of A, B, and C are represented with turquoise, the cluster of D and E are represented with blue, and F is represented with grey.

C. Consider the following small distance matrix for four objects A-D. The objects might be taxa (sequences) or genes from an expression experiment. **7.** Do the UPGMA by hand, showing the new UPGMA distance matrices as you merge clusters (include your work on the Word document or upload a photo of handwritten work). **8.** Create the linkage matrix as you did in A.1, and list the

height of each branch. For example, the first row of the linkage matrix is -1, -2 with height .14.

	A	B	C	D
A	x	0.14	0.48	0.33
B		x	0.5	0.28
C			x	0.55
D				x

7.)

	A	B	C	D
A	x	0.14	0.48	0.33
B		x	0.5	0.28
C			x	0.55
D				x

$d(AB,C) = (d(A,C) + d(B,C))/|AB||C| = (0.48 + 0.50)/(2*1) = 0.49$
$d(AB,D) = (d(A,D) + d(B,D))/|AB||D| = (0.33 + 0.28)/(2*1) = 0.305$

	A B	C	D
AB	x	0.49	0.305
C		x	0.55
D			x

$d(ABD,C) = (d(A,C) + d(B,C) + d(D,C))/|AB||C| = (0.48 + 0.50 + 0.55)/(3*1) = 0.51$

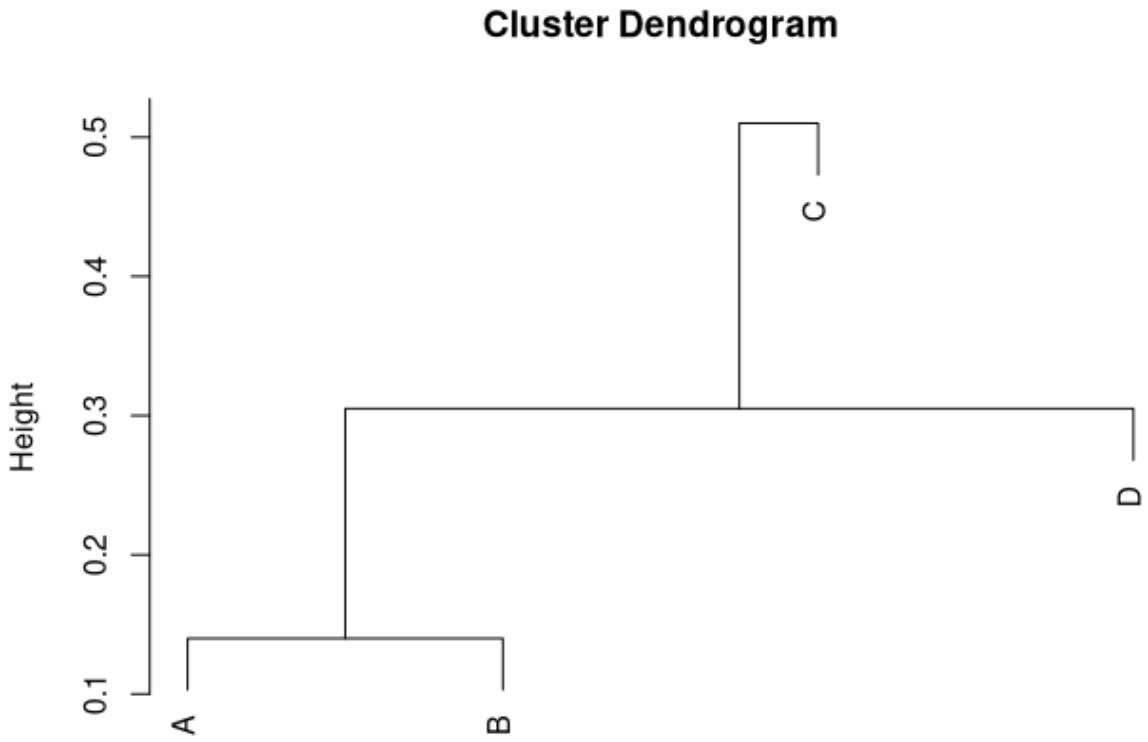
	ABD	C
ABD	x	0.51
C		x

8.)

Linkage Matrix

-1	**-2**	**# (1) First Merge (A,B)**	**Height = 0.14**
1	**-4**	**# (2) Second Merge (AB, D)**	**Height = 0.305**
2	**-3**	**# (3) Third Merge (ABD, C)**	**Height = 0.51**

9. Mirror the code you used in A.2 to plot the dendrogram for this new matrix.



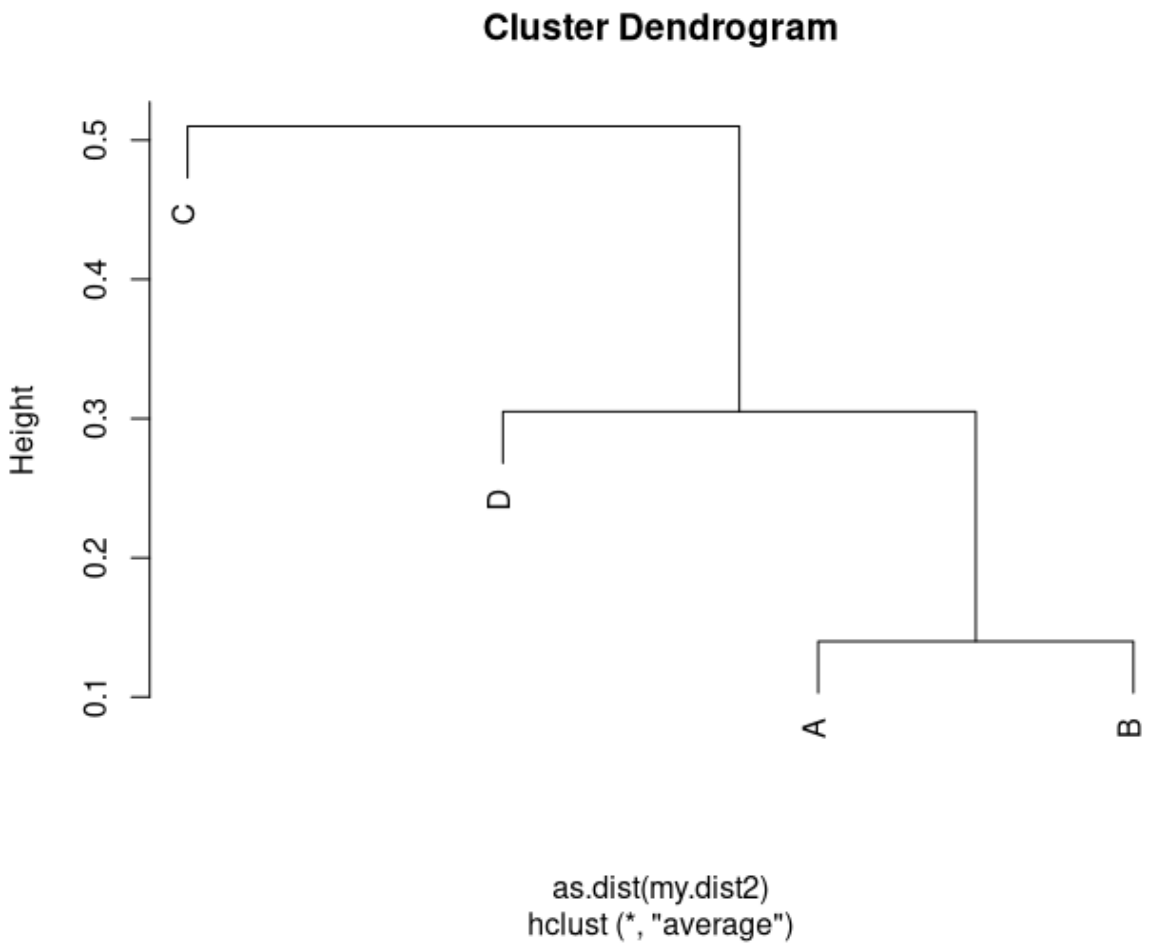

```
a <- list()
a$merge <- rbind(c(-1, -2),  # +1 (A-B)
                 c(1, -4),   # +2 (A-B)-D
                 c(2, -3))   # +3 ((A-B)-D)-C
```

a$height <- c(0.14, 0.305, 0.51) # merge heights
a$order <- 1:4 # order of leaves
a$labels <- LETTERS[1:4] # labels of leaves as Letters
class(a) <- "hclust" # force a to be hclust object
# plot the tree
# plot knows that a is an hclust object and plots accordingly
plot(a) # or use plot(as.dendrogram(a))

Use code below to verify your results by applying hclust to the distance matrix.

```
# create distance matrix
my.dist2 <- matrix(data=rep(0,16),ncol=4)
my.dist2[1,2]<-.14; my.dist2[1,3]<-.48; my.dist2[1,4]<-.33;
                    my.dist2[2,3]<-.50; my.dist2[2,4]<-.28;
                                        my.dist2[3,4]<-.55;
# make matrix symmetric
# make the lower triangle equal to the upper triangle
my.dist2[lower.tri(my.dist2)] = t(my.dist2)[lower.tri(my.dist2)]
diag(my.dist2)<-999 # big number to guarantee it's not the min
# compare with hclust
test <- hclust(as.dist(my.dist2), method="average") # average linkage, UPGMA
test$merge
test$height
test$order
test$labels<-LETTERS[1:4]
plot(test,hang=-1)
```

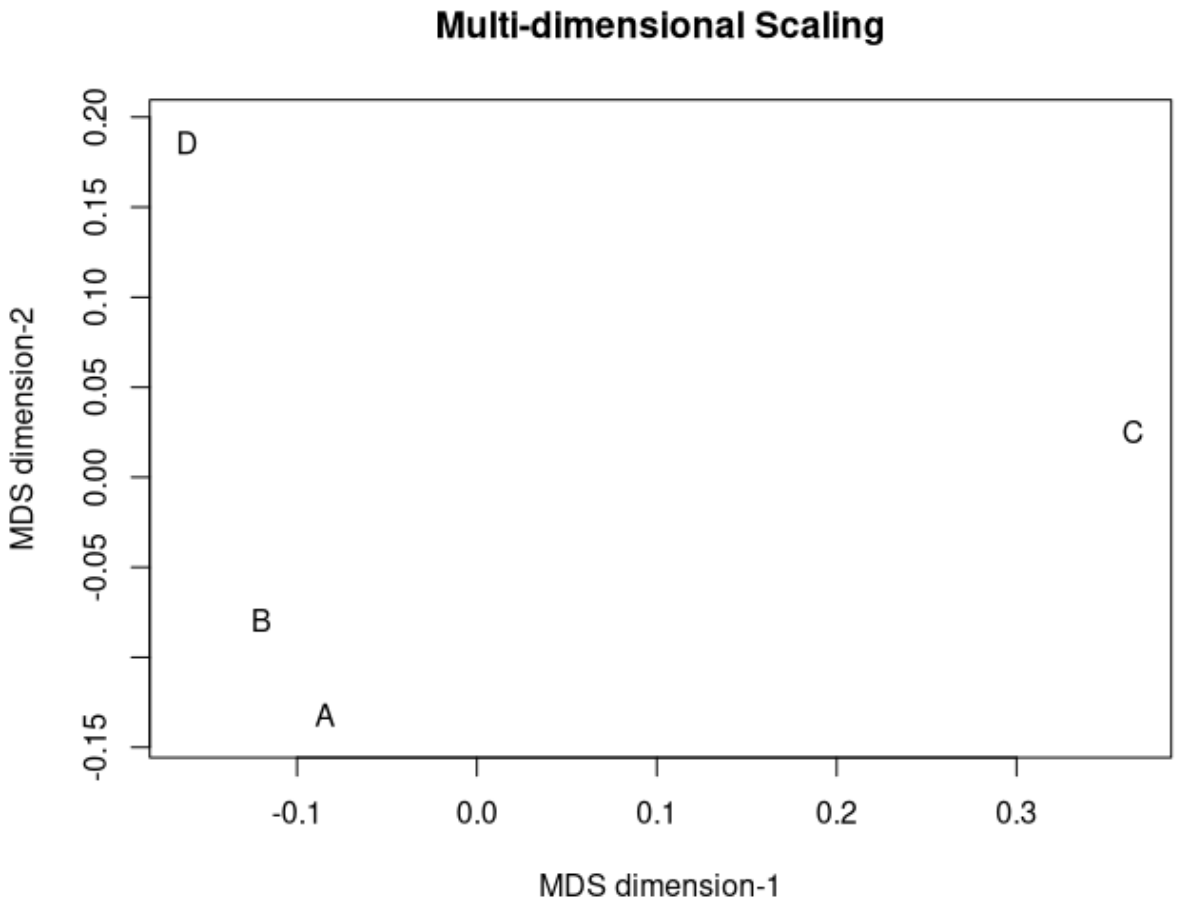
Results verified.
NOTE: Removed the "hang=-1" component. No particular reason other than personal aesthetics, since the original dendrogram had the C hang splitting the dendrogram. I could have restructured the ordering, but removing the hang was a very trivial workaround.



D. Multidimensional Scaling: MDS plot of a distance matrix. You can visualize the relationships between the objects in an abstract 2d or 3d space using multidimensional scaling (MDS) given the pairwise distance matrix between the objects. MDS is an optimization and visualization technique that creates an artificial space such that the distance between the objects in the space is close to the distances in the original distance matrix. **10.** Show the 2d MDS plot for my.dist2. How does this visualization change your perspective on how the objects are clustered?

```
## MDS, you might need to refresh (sweep) the plot window
locs<-cmdscale(as.dist(my.dist2), k=2) # k is number of mds components
x<-locs[,1]
y<-locs[,2]
# pch=NA hides plot symbols:
plot(x,y,main="Multi-dimensional Scaling",
            xlab="MDS dimension-1", ylab="MDS dimension-2", pch=NA,asp=1)
# pch specifies plot symbol; we want null because we are using letter next
```

```
# plot the text labels instead of symbols:
text(x,y,test$labels,cex=1)
```



10.)

Using MDS allows for viewing the objects' distance from each other with an easy-to-interpret relative scale in the 2 dimensional coordinate plane. In this example, we can see A and B are physically close to each other in the coordinate space, and C and D are farther apart.