CSI 2300: Intro to Data Science

Evelyn Pan

In-Class Exercise 23: Modeling – Clustering and Multidimensional Scaling

In this lecture, we're going to work with a mystery dataset and try to unlock some of the patterns it contains. The mystery dataset is from one of the mowater datasets, but it has been sub-sampled and some variables removed to protect its identity (and make it easier to work with). There's a lot to do, so get going!

1. Load and scale the dataset.

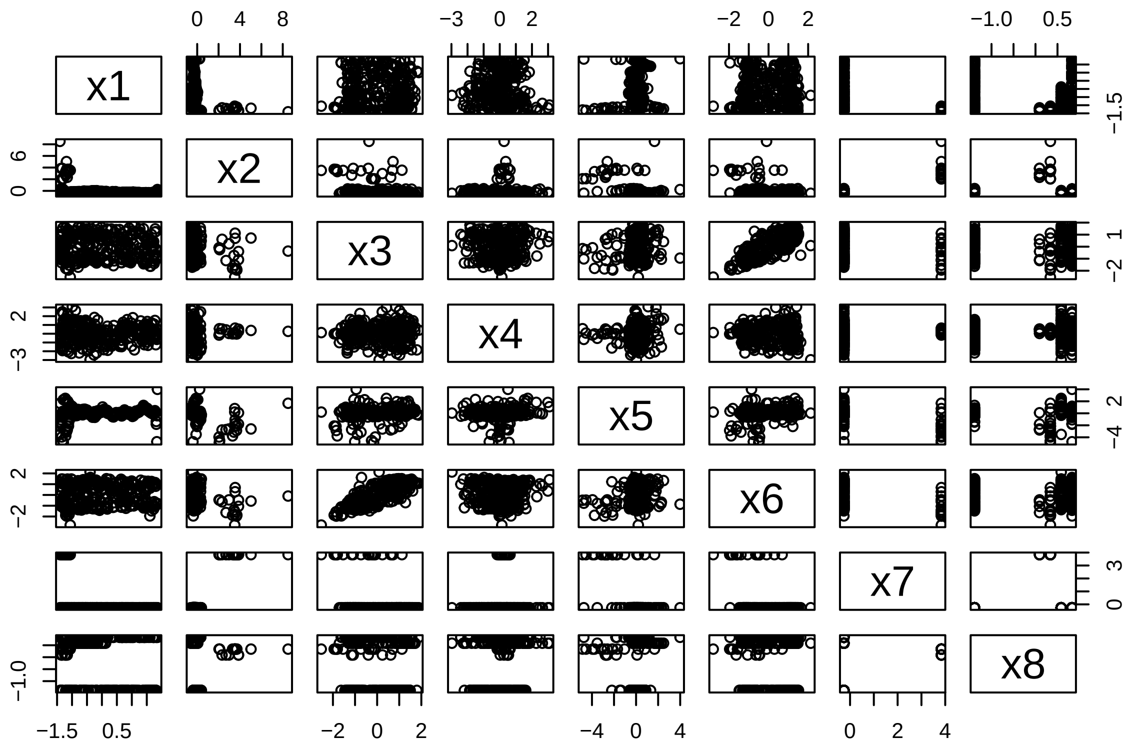
   - Load in the dataset you were given in the `rda` file. The data frame contained has the mysterious name "lecture23".
   - It's important to manage the scale of our variables in distance-based methods, otherwise variables that have a large range can dominate the distance calculations. Use `as.data.frame(scale(lecture23))` command to change each variable to have 0 mean and standard deviation of 1. Replace your data (`lecture23`) with the output of that scaling.
   - Check your work: run `sapply(lecture23, mean)` and `sapply(lecture23, sd)`, and you should get values near zero and one.

```
load("lecture_23_inclass.rda")
lecture23 = as.data.frame(scale(lecture23))

sapply(lecture23, mean)
#            x1            x2            x3            x4            x5
# -1.593488e-17 -8.547721e-17 -2.357681e-16  2.472828e-16 -4.291897e-15
#            x6            x7            x8
#  5.610808e-17  3.104121e-15 -4.308836e-17
sapply(lecture23, sd)
# x1 x2 x3 x4 x5 x6 x7 x8
#  1  1  1  1  1  1  1  1
```

2. You've been given "original space" data. Explore this data by plotting some of its variables. You may find the `pairs` function useful. Do you notice any clusters? It may be hard to tell (because this data has more than two variables), but how many clusters do you think you see (just give an estimate)?

```
pairs(lecture23)
```

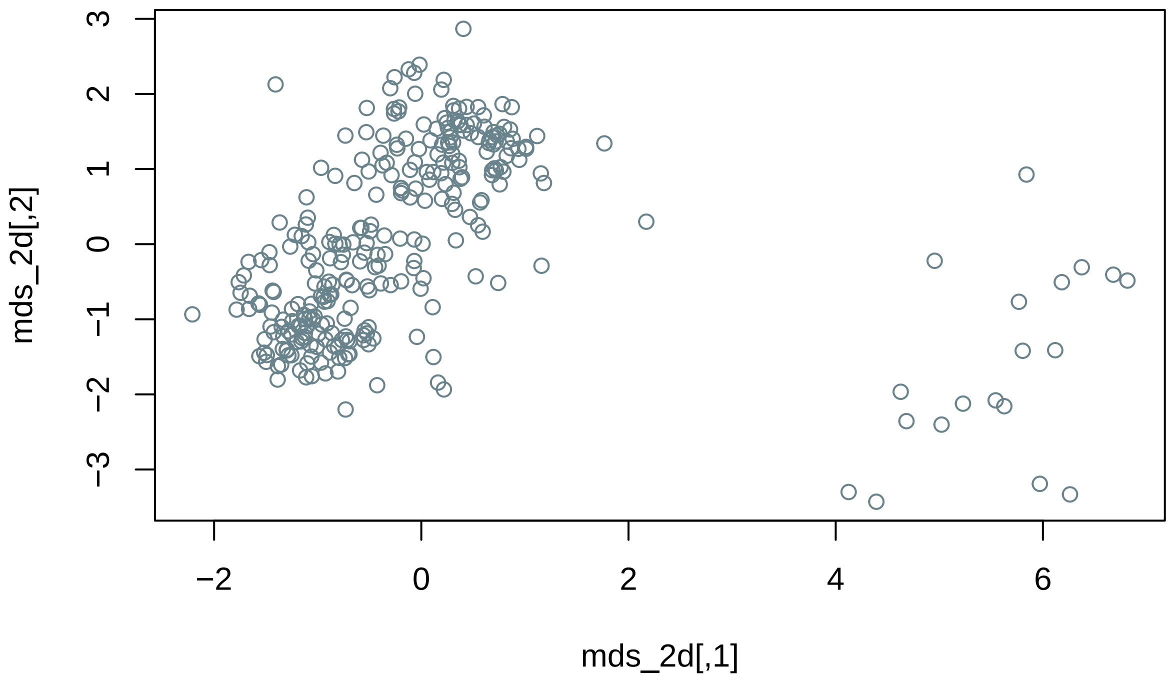


We noticed the 2x5 clusters.

3. Construct the pairwise distance matrix, and then use it to construct a two-dimensional visualization using multidimensional scaling.

    - Construct the pairwise distance matrix, and save it in a variable called `pairwise_dist`.
    - Use multidimensional scaling to produce a 2-dimensional view of the `pairwise_dist` matrix you just created. Let's call this 2-dimensional version `mds_2d`.
    - Then plot `mds_2d`. Do you see clusters? How many clusters do you see?

```
pairwise_dist = dist(lecture23)
#k = 2 means you take something that's 8D to 2D to scale/view it
mds_2d = cmdscale(pairwise_dist, k = 2)
plot(mds_2d, col = "lightblue4")
```



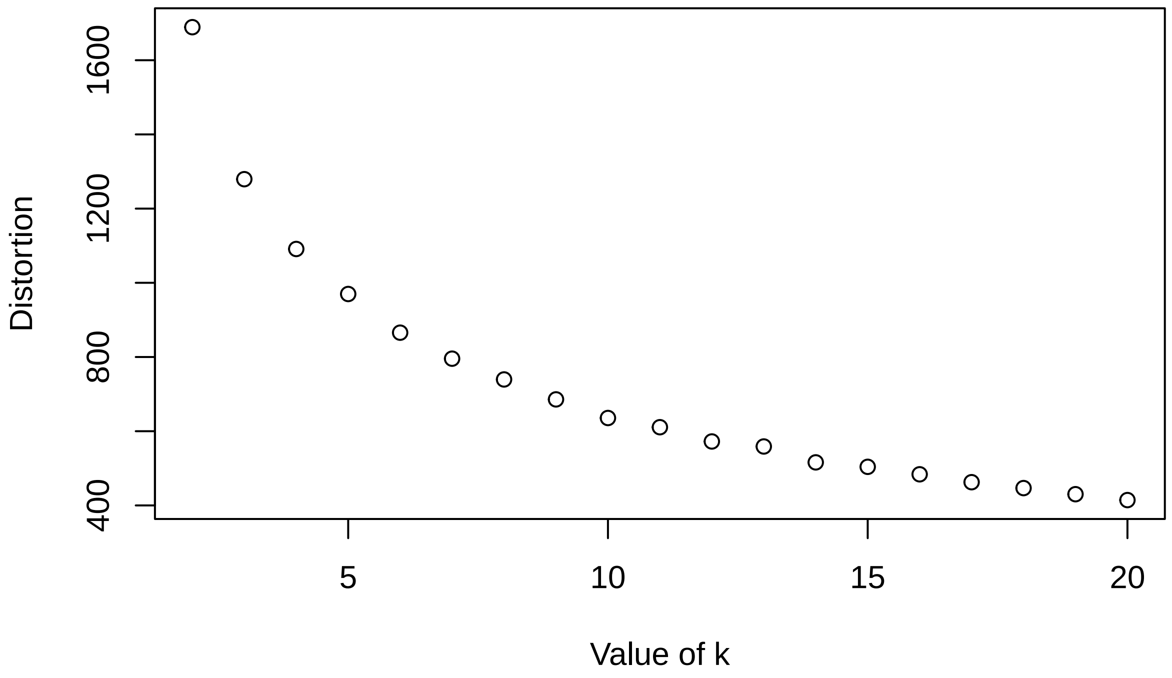
We see 2 clusters because the `k=2` command takes something that's a certain dimension to 2D to scale it.

4. Cluster the original-space data using k-means.

    - Cluster `lecture23` with each value of k from 2 up to 20.
    - For each k, record the distortion (available as the field `$tot.withinss` from the model).
    - Make a plot of the distortion as a function of k.
    - Based on this plot, how many clusters do you think there are in the data? Look for a low distortion, but good vertical separation between the distortion for your chosen k and the distortion for $k - 1$.

```
set.seed(1)
distortion <- NULL
#number of clusters from 2 to 20
k_vals <- 2:20
for (k in k_vals){
  m <- kmeans(lecture23, k, iter.max = 100, nstart = 20)
  distortion[k] <- m$tot.withinss
}
```

```r
plot(k_vals, distortion[k_vals],
     xlab = "Value of k", ylab = "Distortion")
```



There are 12 clusters since distortion usually goes down.

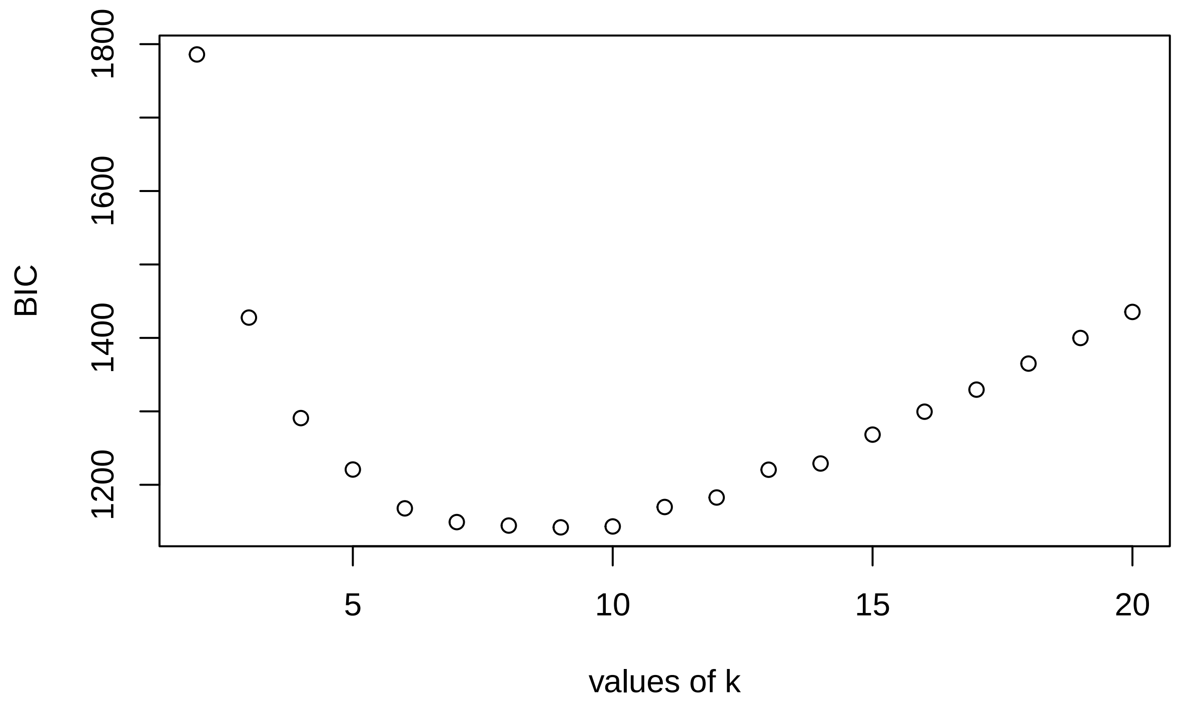
5. The distortion values tend to "just go down" as you add more clusters. We can use a *penalized* distortion with the BIC (Bayesian Information Criterion). The computation is:

$$\text{BIC}(k) = \text{distortion(k)} + (k - 1 + k \cdot d) \cdot \log(n)$$

where k is the number of clusters, d is the number of dimensions (variables), and n is the number of observations. The penalty is based on the number of parameters estimated ($k - 1$ "cluster priors" and $k \cdot d$ parameters for the cluster centers), and we multiply them by the BIC penalty $\log(n)$.

- Compute the BIC-penalized distortion according to the formula above and your distortion values from the previous question.
- Plot the BIC-penalized distortion as a function of k. What k would you choose based on that plot? Is it different than what you chose in the previous question?

```
num_parameters = (k_vals - 1) + k_vals * ncol(lecture23)
bic = distortion[k_vals]+num_parameters * log(nrow(lecture23))
plot(k_vals, bic, xlab = "values of k", ylab = "BIC")
```

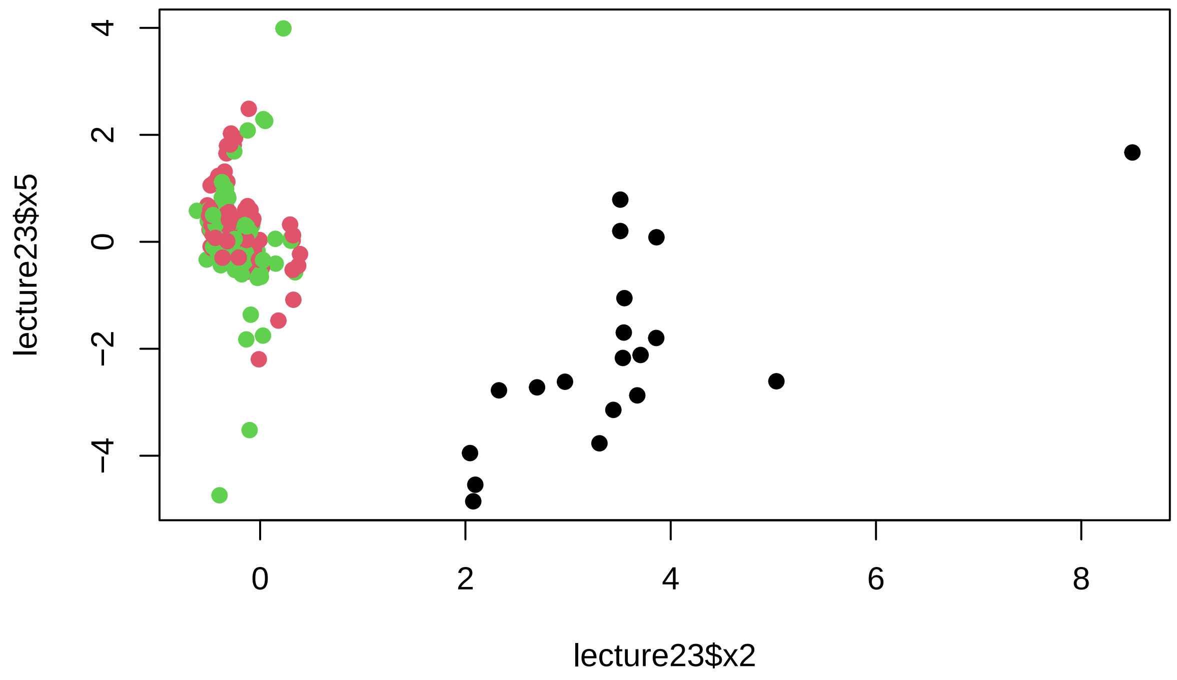


I would choose 8 because with the bic command in the console, you want to choose the one that is the lowest.

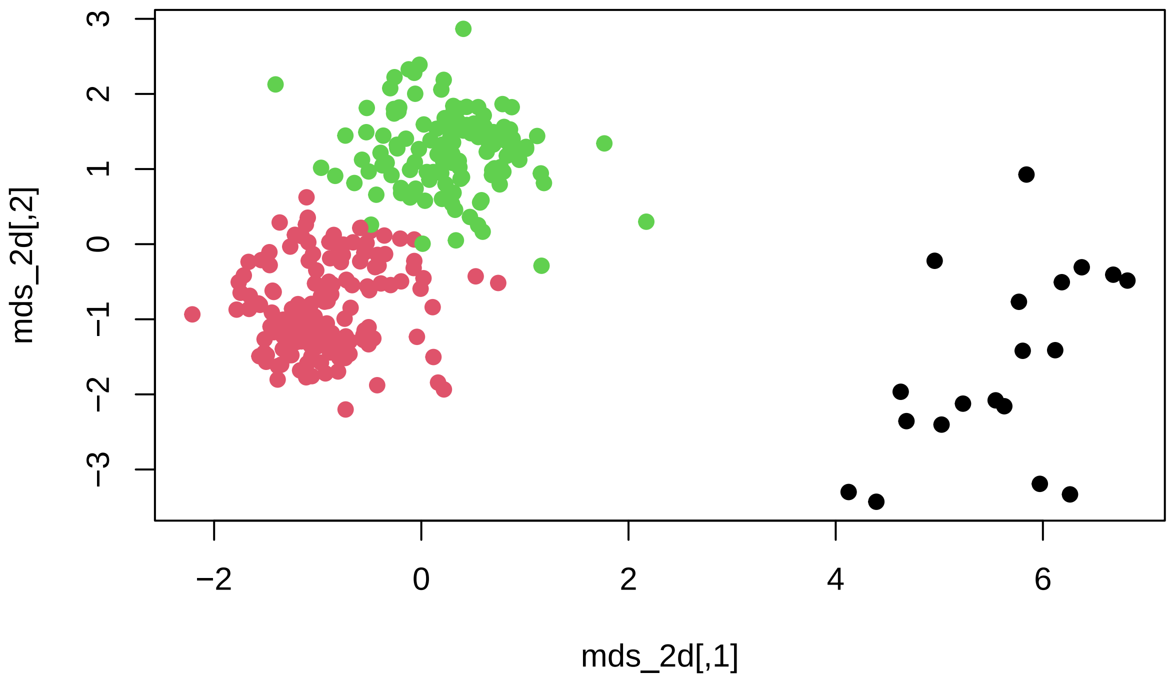
6. Let's visualize the k-means clustering. For ease of grading, we'll choose $k = 3$ (though you may have chosen a different k in the previous questions).

    - Cluster the `lecture23` data using k-means with $k = 3$, and save the model in a variable.
    - Plot `lecture23` variables `x2` and `x5`, and color the points using the cluster labels.
    - Plot the `mds_2d` data, coloring the points using the cluster labels.
    - Do the clusters look reasonable? Keep in mind that we are clustering with 8 variables, but plotting in 2 dimensions, so some clusters may appear to overlap – that's because we are not plotting all the dimensions (and the un-plotted dimensions get "collapsed" or "projected" down onto the two that we are plotting).

```
k3_model = kmeans(lecture23,  3, iter.max = 100, nstart = 100)
plot(lecture23$x2, lecture23$x5, pch = 19, col = k3_model$cluster)
```



```
plot(mds_2d, pch = 19, col = k3_model$cluster)
```



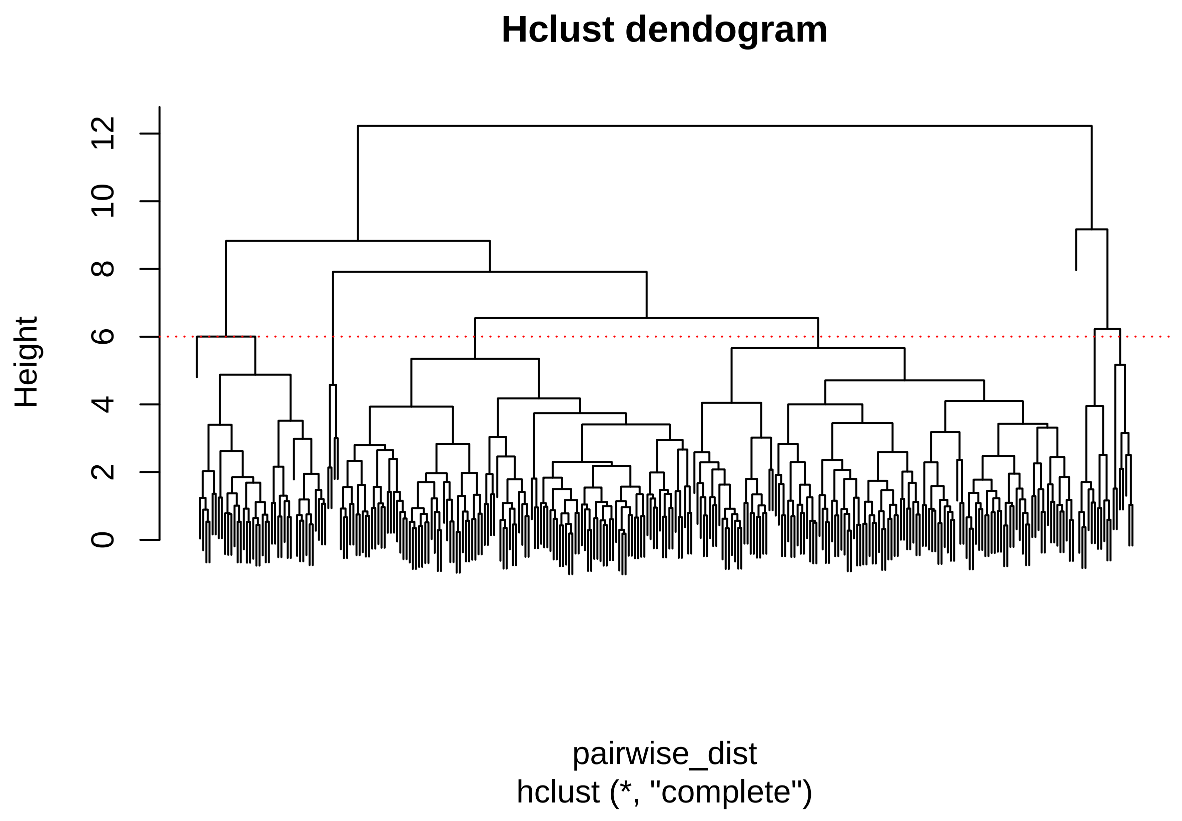
**The clusters look reasonable because

7. Apply hierarchical clustering to the pairwise distance data.

   - Use `hclust` to cluster the `pairwise_dist` matrix, producing a model which we'll call `hclust_model`.
   - Plot the model to obtain the dendrogram.
   - How many clusters do you think there are based on this dendrogram? A good splitting point of the dendrogram is a place where there is a large vertical separation between joined clusters.
   - For your chosen number of clusters, extract the clustering membership labels (one for each observation) from the hierarchical clustering model. You may find `cutree` useful here.
   - Plot the multidimensional scaling data and color the observations with its cluster membership you just extracted.

```
hclust_model = hclust(pairwise_dist)
plot(hclust_model, main = "Hclust dendogram", label = F)

hclust_k = 8
abline(h = hclust_model$height[nrow(lecture23) - hclust_k + 1],
       col = "red", lty = 3)
```

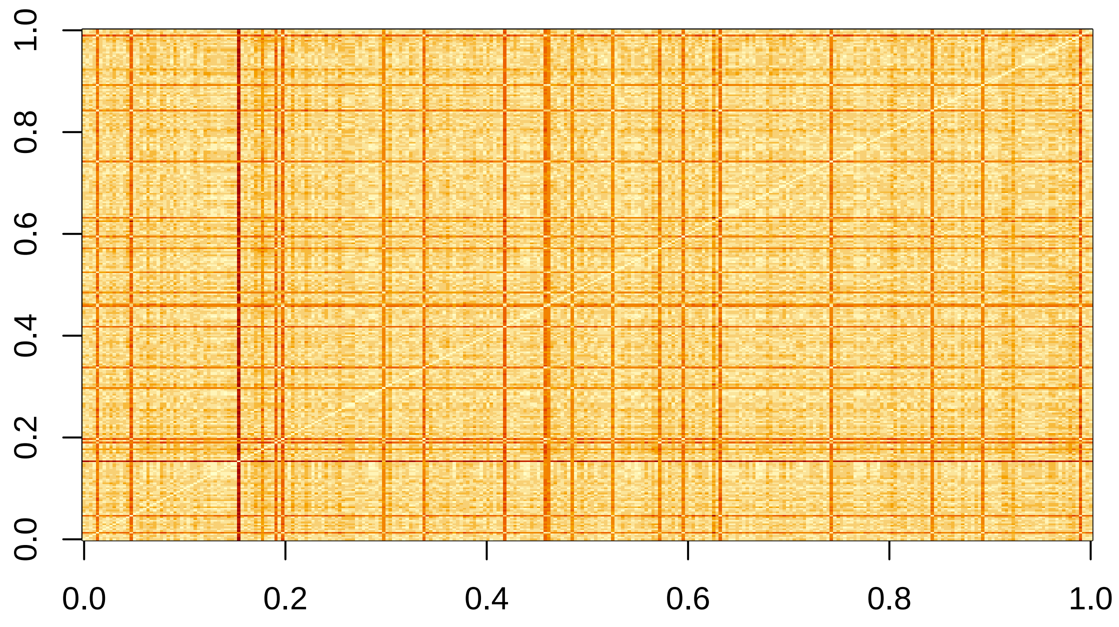


8. **Extra Credit** Let's look at the `pairwise_dist` distance matrix.

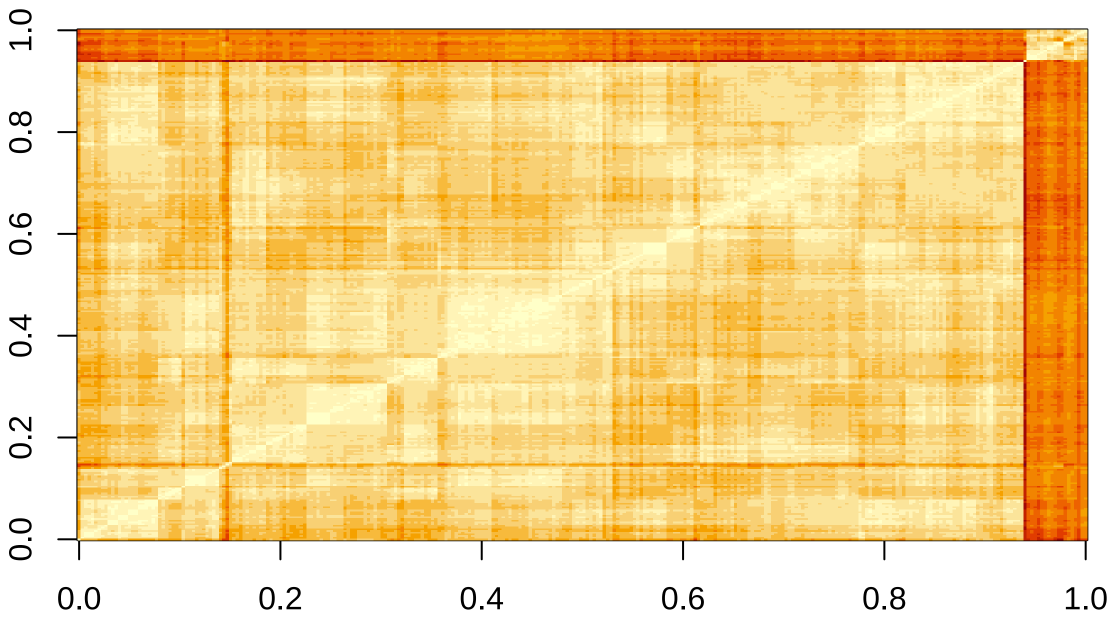
   - Create an image of the `pairwise_dist` matrix we created in question 3. You may find `as.matrix` useful here.
   - The hierarchical clustering model (`hclust_model`) we created in question 6 has an `$order` field, which is the order of the observations along the bottom of the dendrogram. Use this order to:
     - Reorder the `lecture23` observations according to the order found by `hclust`.
     - Create a pairwise distance matrix of the reordered observations.
     - Create an image of this reordered distance matrix.

   Does this second matrix appear to have more or less of a pattern than the first matrix? Do either of the matrices appear to show you clustering information?

```
dist_mat = as.matrix(pairwise_dist)
image(dist_mat)
```



```
reordered = lecture23[hclust_model$order,]
dist_mat_ord = as.matrix(dist(reordered))
image(dist_mat_ord)
```


0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0