Key for (the last) Long Problem Set 9

The file “MetalMix.RData” contains three objects: a dataframe with the name “metals” that contains absorbance values for 27 solutions at eight wavelengths, a vector with the name “score.labels” for identifying points on a scores plot, and vector with the name “rot.labels” for identifying points on a loadings plot. The 27 solutions were prepared by diluting stock solutions of 0.10 M $Cu(NO_3)_2$, 0.10 M $Ni(NO_3)_2$, and 0.10 M $Co(NO_3)_2$ to create

- three pure solutions (one each of Cu^{2+}, Ni^{2+}, and Co^{2+})
- eighteen binary mixtures (six each of Cu^{2+} and Ni^{2+}, Cu^{2+} and Co^{2+}, and Ni^{2+} and Co^{2+})
- six ternary mixtures (each containing all three metal ions)

This is the same data used for LPS08 and you must complete that long problem set before you can begin work on this long problem set. Using this data and a separate handout that identifies the ions present in each solution (which is available to you when you turn in your answers to LPS08), answer the following questions

(1). Complete a cluster analysis of the data in the dataframe “metals” using the clustering methods single, complete, and average. You do not need to center and scale the data. Examine the resulting dendrograms and discuss their individual success and/or lack of success in identifying meaningful clusters among the 27 samples.

Answer. Figure 1 shows the dendrogram using the nearest neighbors algorithm (single), which does a good job of clustering together samples from the three sets of binary mixtures. Three of the ternary mixtures cluster together, but three cluster with the binary mixtures; based on the scores plot from LPS08, these three ternary samples are enriched in Cu^{2+} (sample 21), Ni^{2+} (sample 12), and Co^{2+} (sample 26) and each clusters with its nearest binary mixture. The pure solutions are the last samples to cluster, which is not surprising as the scores plot shows that they are at the greatest distance from other samples.

```
load("MetalMix.RData")
metals.labels = c("1:Co", "2:CoCu", "3:CoCu", "4:CoNi", "5:CuNi",
                  "6:CuNi", "7:CoCuNi", "8:Ni", "9:CoCu", "10:CoNi",
                  "11:CuNi", "7:CoCuNi", "13:CoCu", "14:CoNi",
                  "15:CuNi", "7:CoCuNi", "17:CoCu", "18:CuNi",
                  "7:CoCuNi", "20:CoNi", "7:CoCuNi", "22:CoNi",
                  "23:CoCu", "24:CuNi", "25:Cu", "26:CoCuNi", "27:CoNi")
metals.dist = dist(metals)
```

```
metals.single = hclust(metals.dist, method = "single")
plot(metals.single, labels = metals.labels, hang = -1)
```

Figure 2 shows that the dendrogram using the furthest neighbors algorithm (complete) is less successful at forming clusters of binary mixtures and ternary mixtures. Instead, this algorithm identifies samples with higher concentrations of Cu^{2+} (upper right corner of scores plot from LPS08), with higher concentrations of Co^{2+} (upper left corner of scores plot), and with higher concentrations of Ni^{2+} (bottom middle of the scores plot). The three ternary samples with roughly equal amounts of Co^{2+}, Cu^{2+}, and $^{Ni2+}$ cluster with nickel; however, this likely is an artifact of the data.

```
metals.complete = hclust(metals.dist, method = "complete")
plot(metals.complete, labels = metals.labels, hang = -1)
```

Finally, Figure 3 shows that the dendrogram using the mean distance algorithm (average) is the least successful at forming logical clusters as it neither gathers together the binary mixtures nor solutions that are enriched in one of the pure solutions. Interestingly, the pure solution of Ni^{2+} is the most unique solution when using this algorithm.

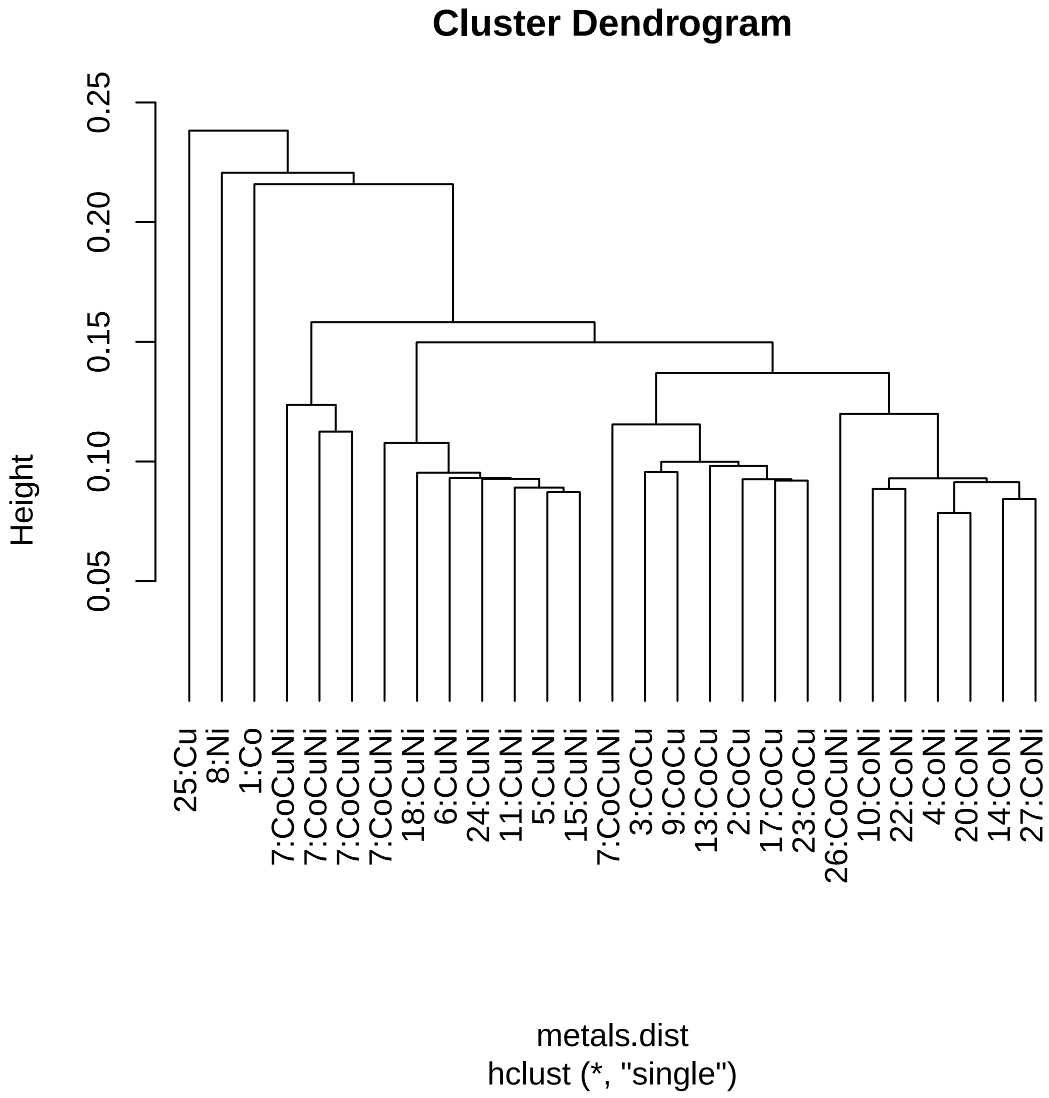

Figure 1: Clustering of samples based on the nearest neighbor method.

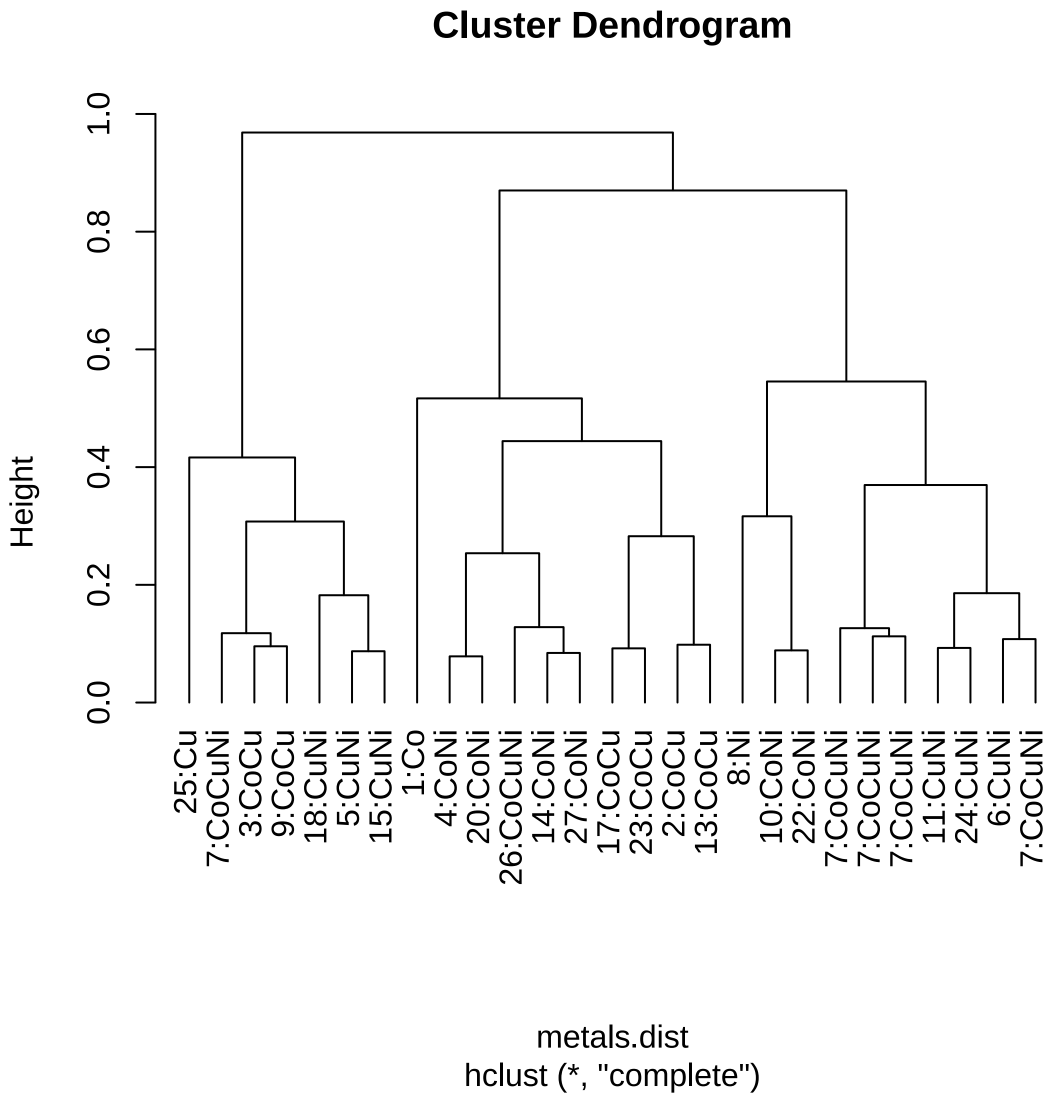

Figure 2: Clustering of samples based on the furthest neighbor method.

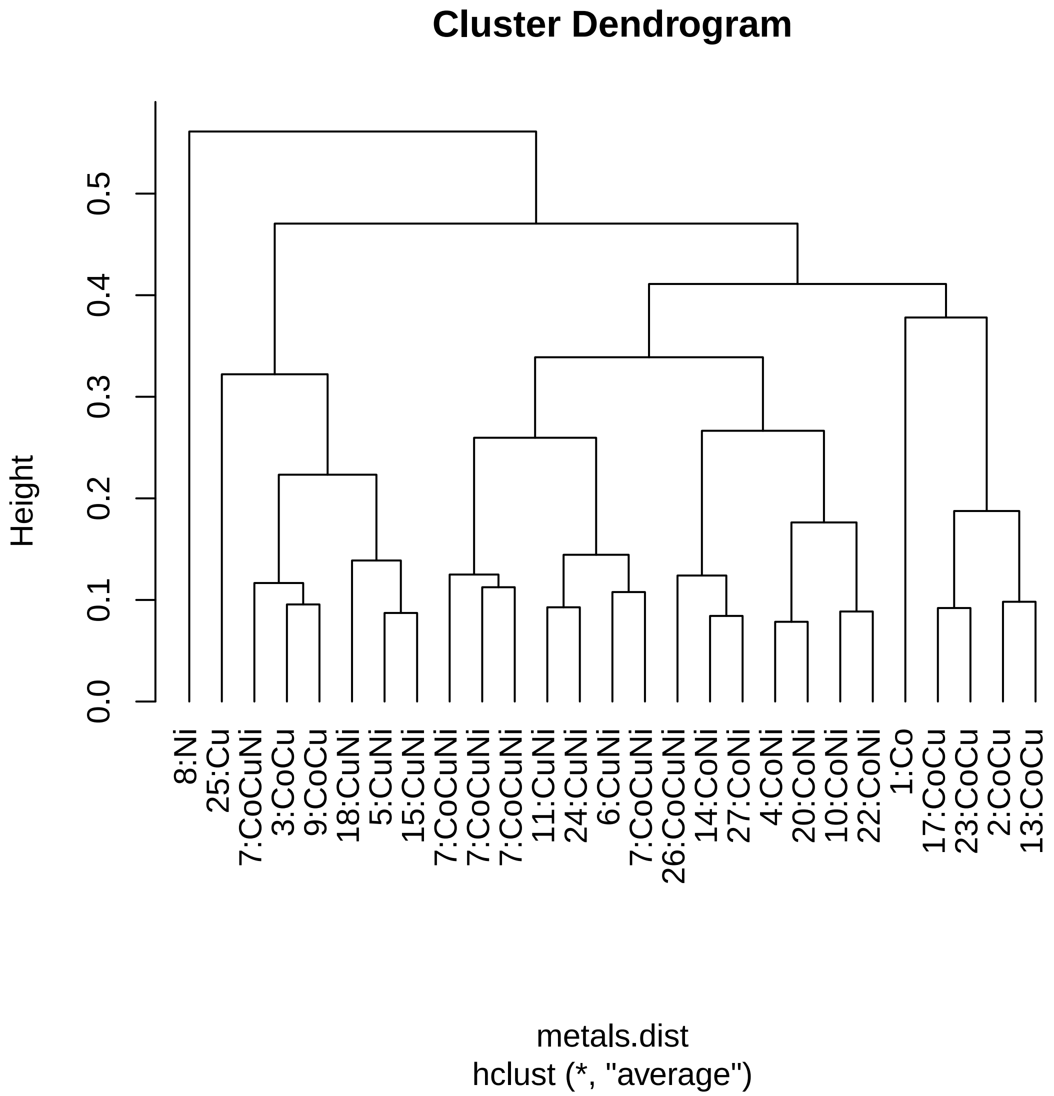

Figure 3: Clustering of samples based on the average method.

```r
metals.average = hclust(metals.dist, method = "average")
plot(metals.average, labels = metals.labels, hang = -1)
```

(2). Transpose the data in the dataframe “metals” and complete a cluster analysis using the clustering methods single, complete, and average. You do not need to center and scale the data. Examine the resulting dendrograms and discuss their individual success and/or lack of success in identifying meaningful clusters among the eight wavelengths.

Answer. Figure 4 shows the dendrogram using the nearest neighbors algorithm (single), which does a good job of clustering together the two wavelengths strongly associated with Co^{2+} (470 nm and 510 nm) and the two wavelengths strongly associated with Cu^{2+} (790 nm and 900 nm). It also identifies that 400 nm is the most unique wavelength in the loadings plot. The clustering of the remaining three wavelengths are not particularly informative, although solutions of Cu^{2+} do absorb strongly at 680 nm, which clusters here with 790 nm and 900 nm.

```r
wave = t(metals)
wave.labels = c(900,790, 680, 600, 510, 470, 400, 360)
```

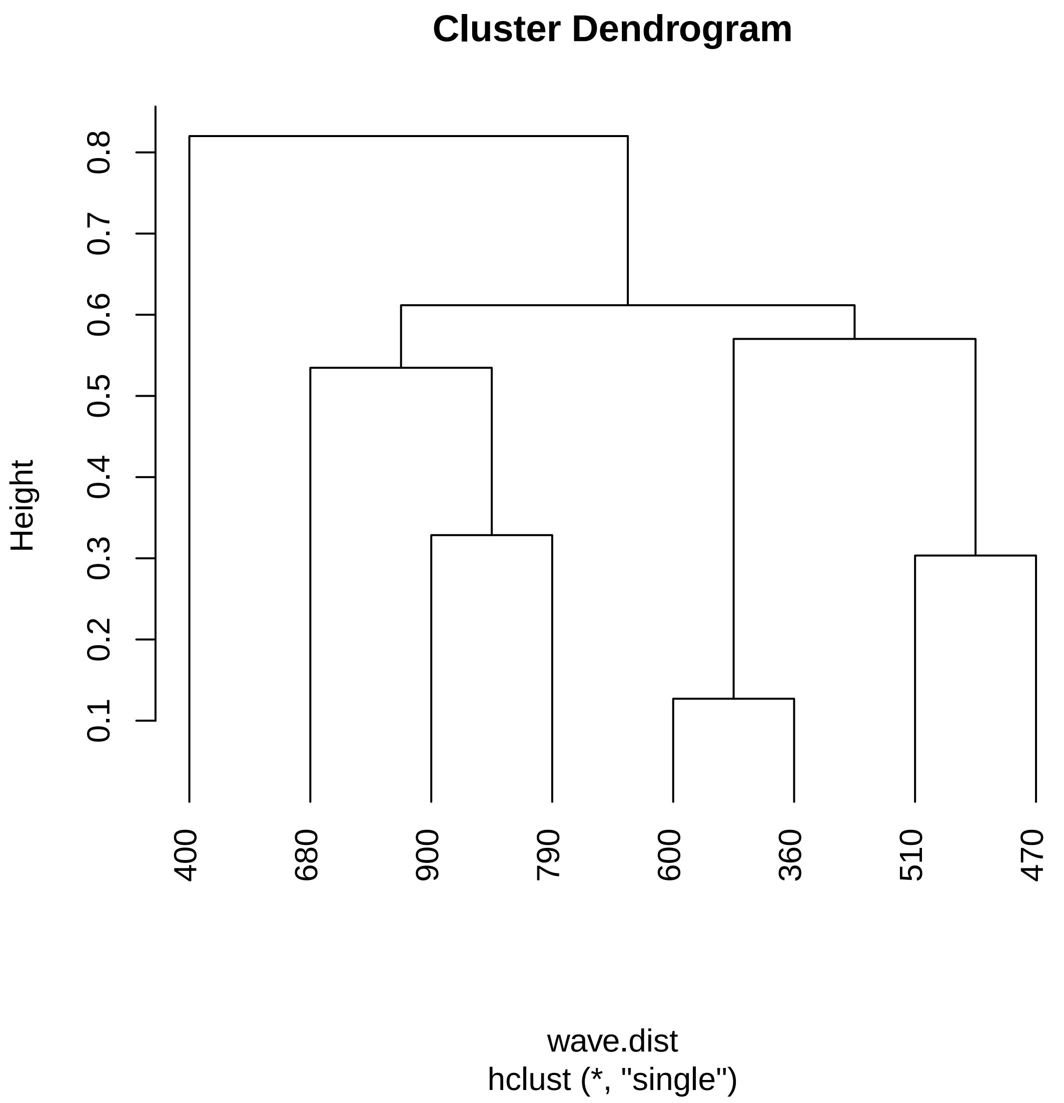

Figure 4: Clustering of wavelengths based on the nearest neighbor method.

```
wave.dist = dist(wave)
```

```
wave.single = hclust(wave.dist, method = "single")
plot(wave.single, labels = wave.labels, hang = -1)
```

Figure 5 showns that the dendrogram using the furthest neighbors algorithm (complete) does a good job of clustering together the two wavelengths most strongly associated with Co^{2+} (470 nm and 510 nm) and the two wavelengths most strongly associated with Cu^{2+} (790 nm and 900 nm), and it also clusters together the three wavelengths in the middle of the loadings plot: 360 nm, 600 nm, and 680 nm. It also identifies that 400 nm is the most unique wavelength in the loadings plot.

```
wave.complete = hclust(wave.dist, method = "complete")
plot(wave.complete, labels = wave.labels, hang = -1)
```

Finally, Figure 6 shows that the dendrogram using the mean distance algorithm (average) is nearly identical to that obtained using the nearest neighbor algorithm, differing only in the height of each cluster.

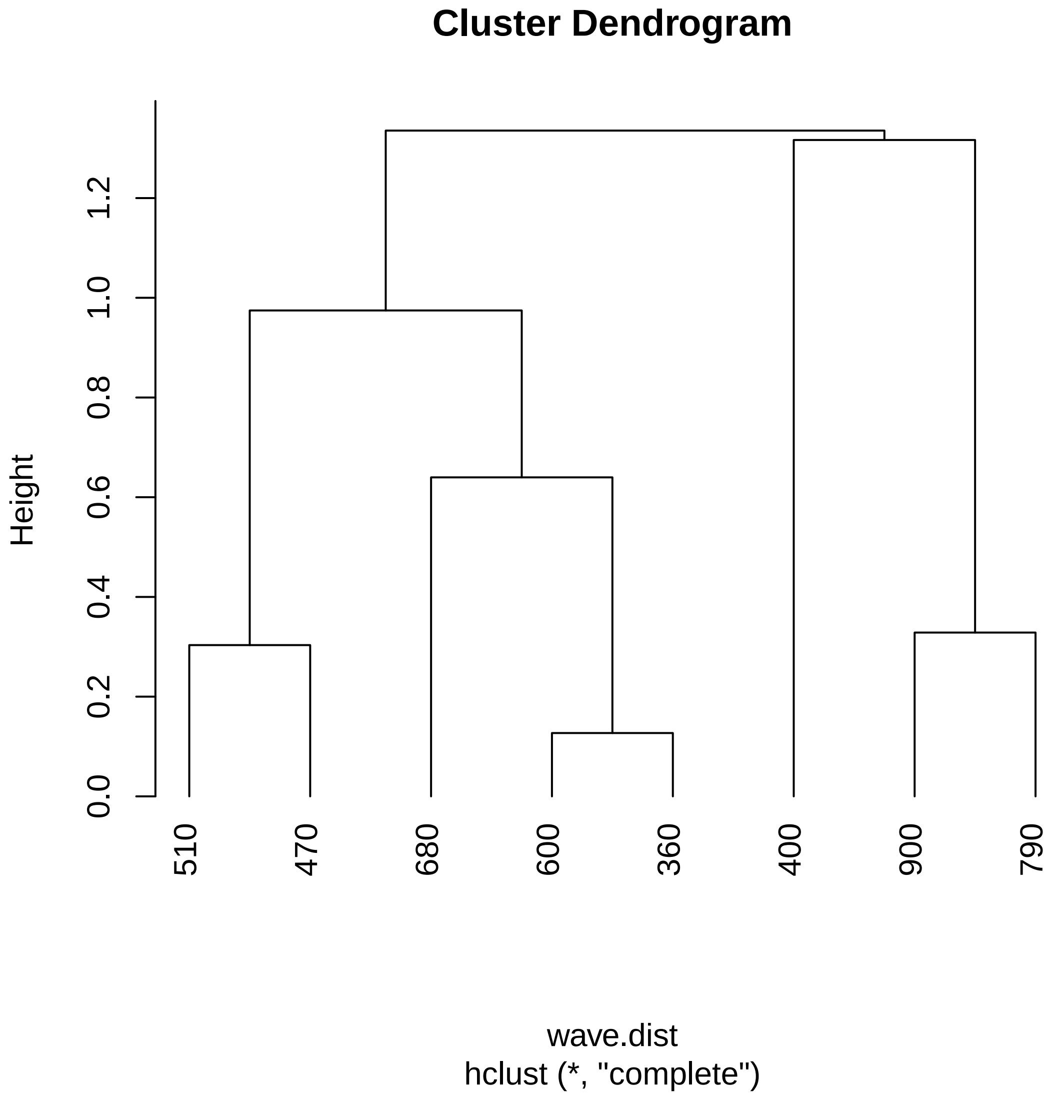

Figure 5: Clustering of wavelengths based on the furthest neighbor method.

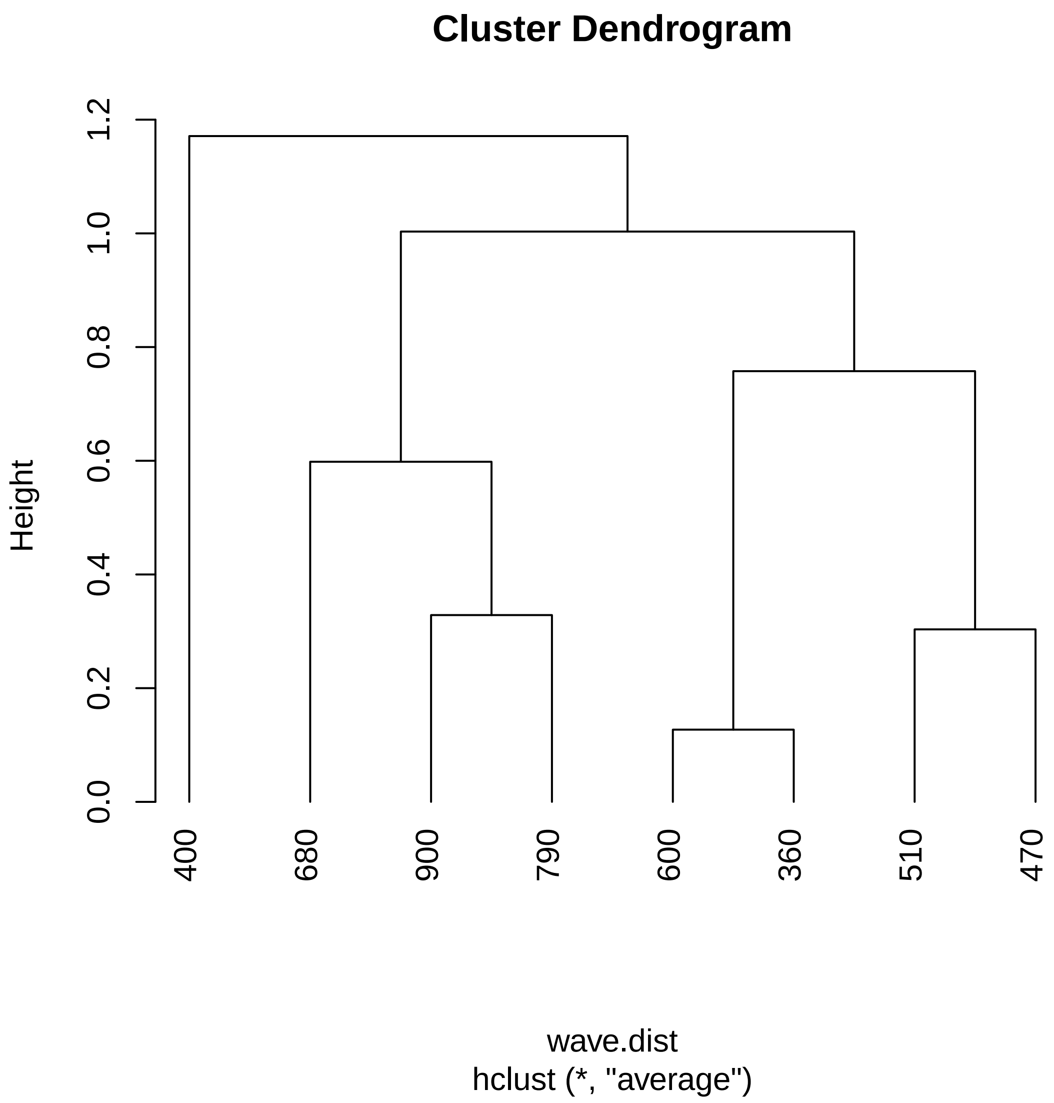

Figure 6: Clustering of wavelengths based on the average method.

```
wave.average = hclust(wave.dist, method = "average")
plot(wave.average, labels = wave.labels, hang = -1)
```