APSTA-GE 2011. Proj. #1

January 2025: Yael Beshaw

```
# likely useful libraries and initial seed set.
library(cluster)
library(foreign)
library(gtools)
library(NbClust)
library(ggplot2)
library(foreign)
library(haven)
library(caret)
library(klaR)
library(ggdendro)
library(GGally)
library(knitr)
library(gridExtra)
library(factoextra)
library(tibble)
library(phyclust)

set.seed(2011)
```

The objective of a cluster analysis is knowledge discovery – somehow, by identifying groups in the data, you learn something interesting about the substantive area being explored.

You will look for potential clusters in the Australian Leptograpsus Crabs data. As you know from the handouts, 200 crab specimens were collected at Fremantle, Western Australia in the mid-1970s (Campbell and Mahon, 1974). Each specimen has measurements on: frontal lip (FL), rear width (RW), length of midline of the carapace (CL), maximum width of carapace (CW), and body depth (BD), all in millimeters.

```
crabs <- read_dta("/Users/yaelbeshaw/Downloads/crabs.dta")
```

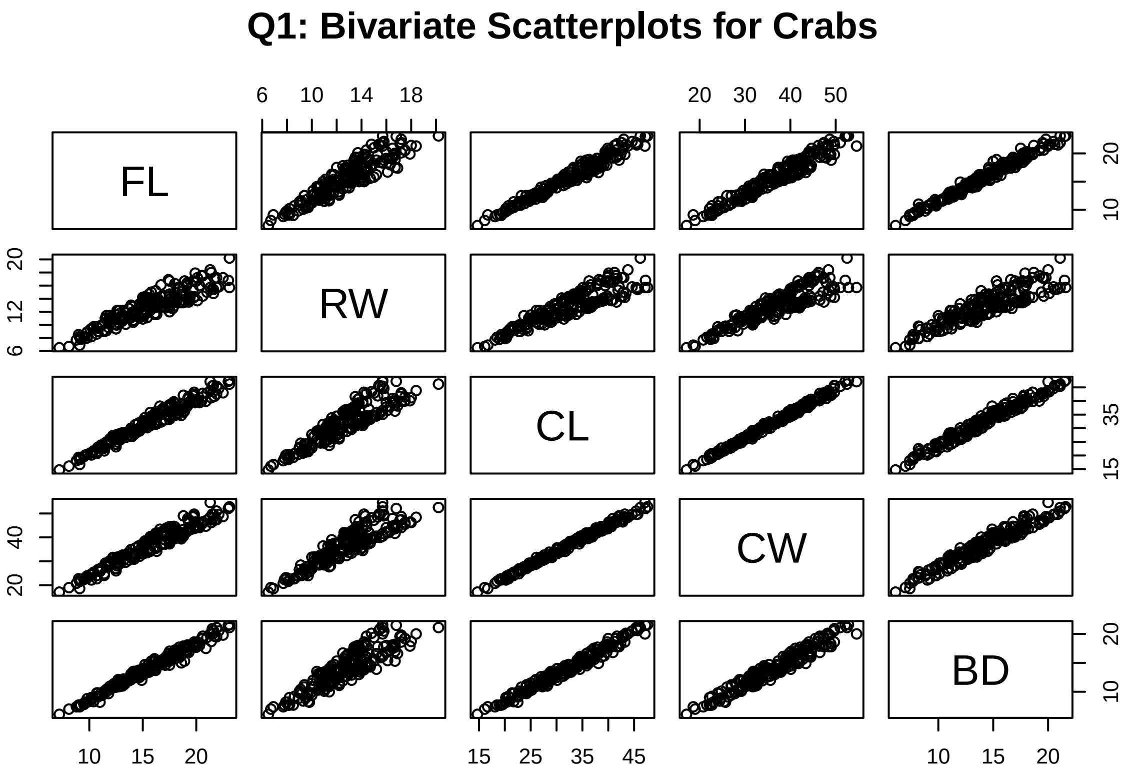
You also know the sex and species of these crabs – these are the demographics you will explore *after* clustering.

We have sometimes referred to this as classifying "blindly" since you don't know the labels, but you assess your clustering in part by its ability to separate crabs in a manner consistent with those labels.

#Q1.

First, explore the five features using bivariate plots. You should explore the need to transform or rescale the measurements. Make a recommendation based on those bivariate plots.

```
pairs(crabs[, 4:8], main = "Q1: Bivariate Scatterplots for Crabs")
```



```r
# exploring transforrmations and rescaling
summary(crabs[, c(4:8)])
```

```
##        FL              RW              CL              CW       
##  Min.   : 7.20   Min.   : 6.50   Min.   :14.70   Min.   :17.10  
##  1st Qu.:12.90   1st Qu.:11.00   1st Qu.:27.27   1st Qu.:31.50  
##  Median :15.55   Median :12.80   Median :32.10   Median :36.80  
##  Mean   :15.58   Mean   :12.74   Mean   :32.11   Mean   :36.41  
##  3rd Qu.:18.05   3rd Qu.:14.30   3rd Qu.:37.23   3rd Qu.:42.00  
##  Max.   :23.10   Max.   :20.20   Max.   :47.60   Max.   :54.60  
##        BD       
##  Min.   : 6.10  
##  1st Qu.:11.40  
##  Median :13.90  
##  Mean   :14.03  
##  3rd Qu.:16.60  
##  Max.   :21.60
```

Based on the bivariate plot and assessing the summary statistics, I would recommend re-scaling the measurements. We see that the variables are on different scales with varying ranges of their minimums and maximums. We see this reflected in the bivariate scatterplots as each plot is on a different scale, making it quite difficult for us to be able to properly compare and make accurate assesments about what we see.

After making your assessment, in order to save time, we have decided (for you) that you should standardize the measurements (the usual z-score transform). The simplest way to standardize is to make a NEW crabs dataframe as follows:

```r
crabs.stdz <- crabs
crabs.stdz[, 4:8] <- scale(crabs[, 4:8])
```

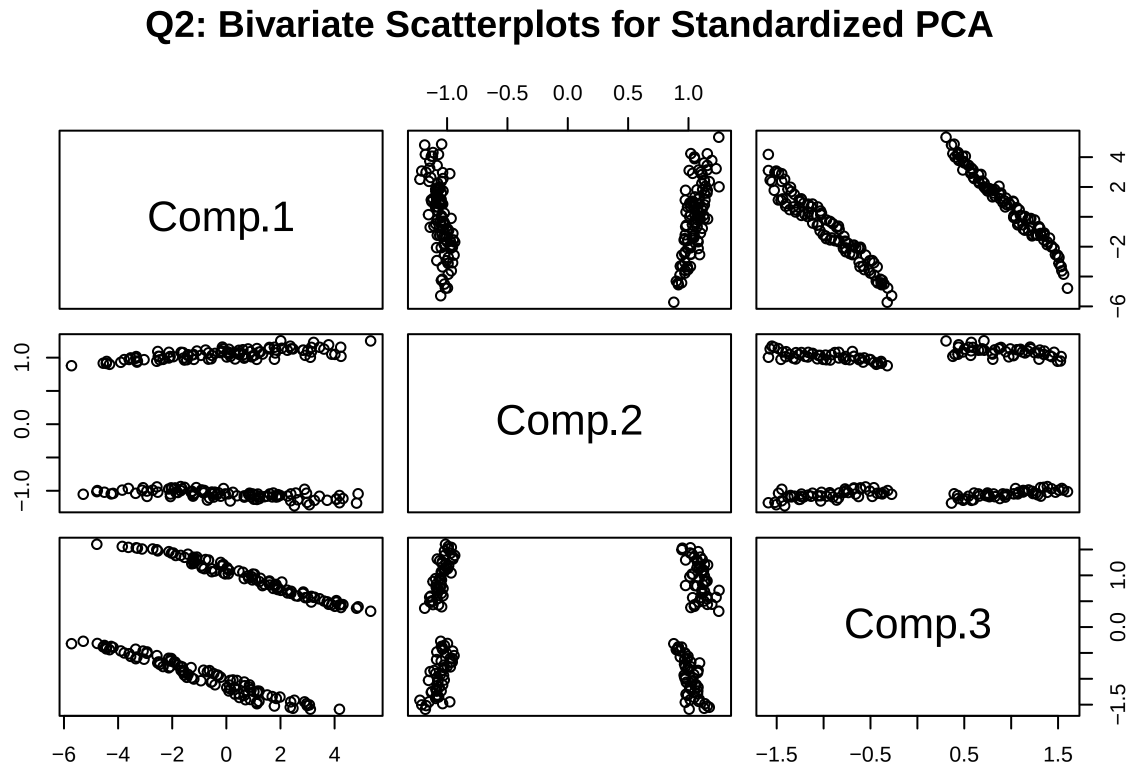
From this point forward, we use crabs.stdz in our analysis (not crabs).

#Q2.

You should also examine bivariate plots using principal components on the standardized version of the data, as these might reveal the clusters better. Do the actual clustering on the raw (standardized) measures, not the principal components.

If you see fairly well separated clusters, particularly if they are 'stringy,' you can use single linkage hierarchical clustering; otherwise, use centroid linkage [justify your choice in your writeup, but only choose ONE method].

```r
# Examine bivariate plots using principal components on the standardized
# version
pc.crabs <- princomp(crabs.stdz, cor = TRUE)$scores
pairs(pc.crabs[, 1:3], col = 1, main = "Q2: Bivariate Scatterplots for Standardized PCA")
```



Based on the results of the bivariate plots using principal components on the standardized version of the data, we see that single linkage hierarchical clustering is the best method to implement here. We see very distinct and "stringy" clusters on the first three PCA's.

We will assume that Euclidean distance (not squared, also known as L_2 norm) is appropriate for these data.

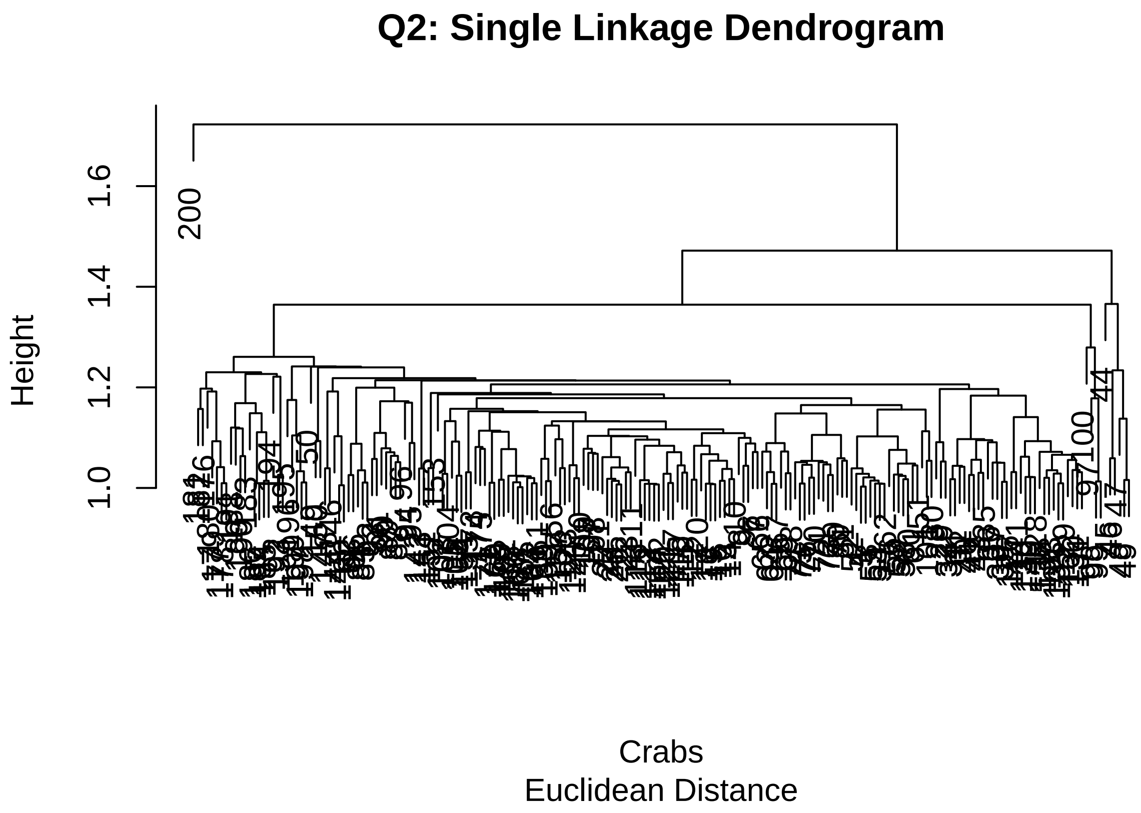
```r
# Single Linkage Hierarchical Clustering

# Euclidean distance
crabs.stdz_dist <- dist(crabs.stdz, method = "euclidean")

```

```
# single linkage hierarchical clustering
hcl.crabz <- hclust(crabs.stdz_dist, meth = "single")
```

#Q3.

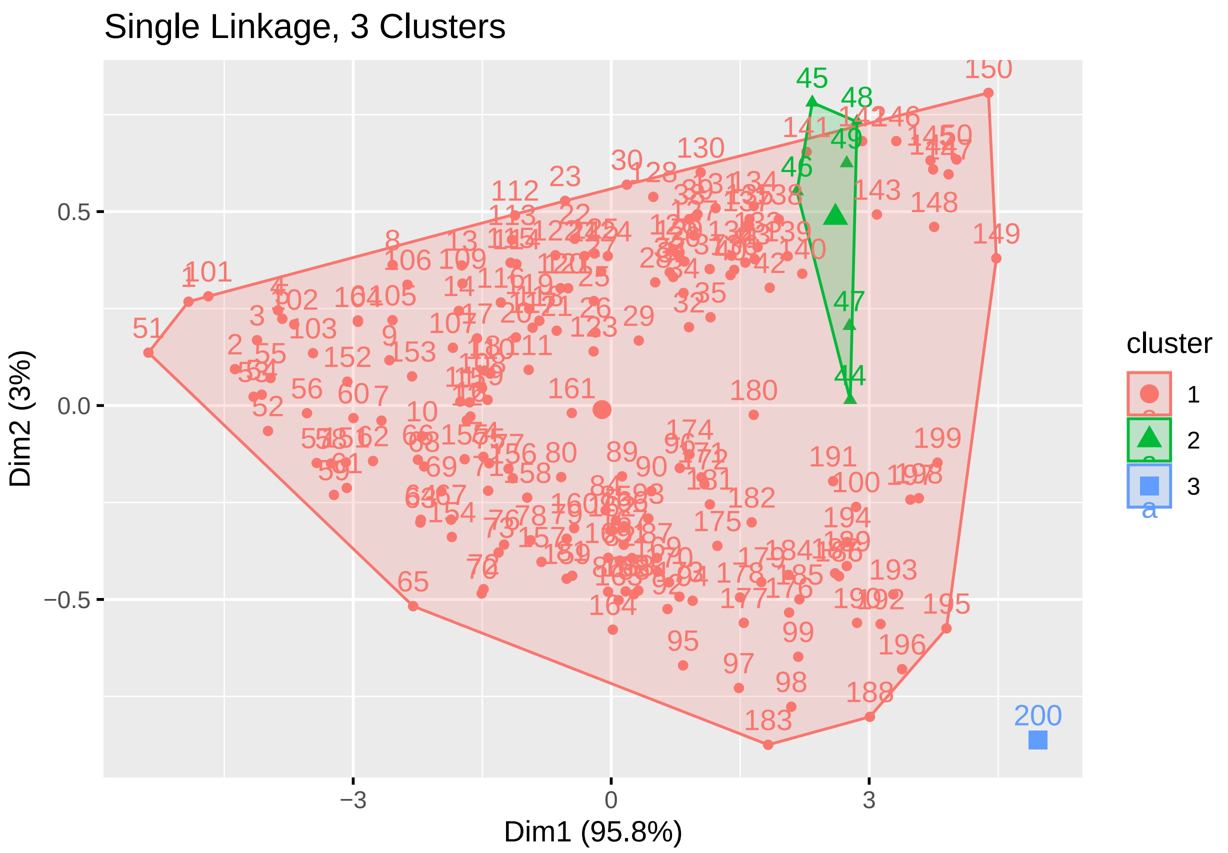
Choose the number of clusters (you think provide good separation between groups and homogeneity within) by *examining the dendrogram* and evaluating several alternative 'cut points' for the number of clusters.

```
# dendrogram
plot(hcl.crabz, main = "Q2: Single Linkage Dendrogram", xlab = "Crabs", sub = "Euclidean Distance")
```

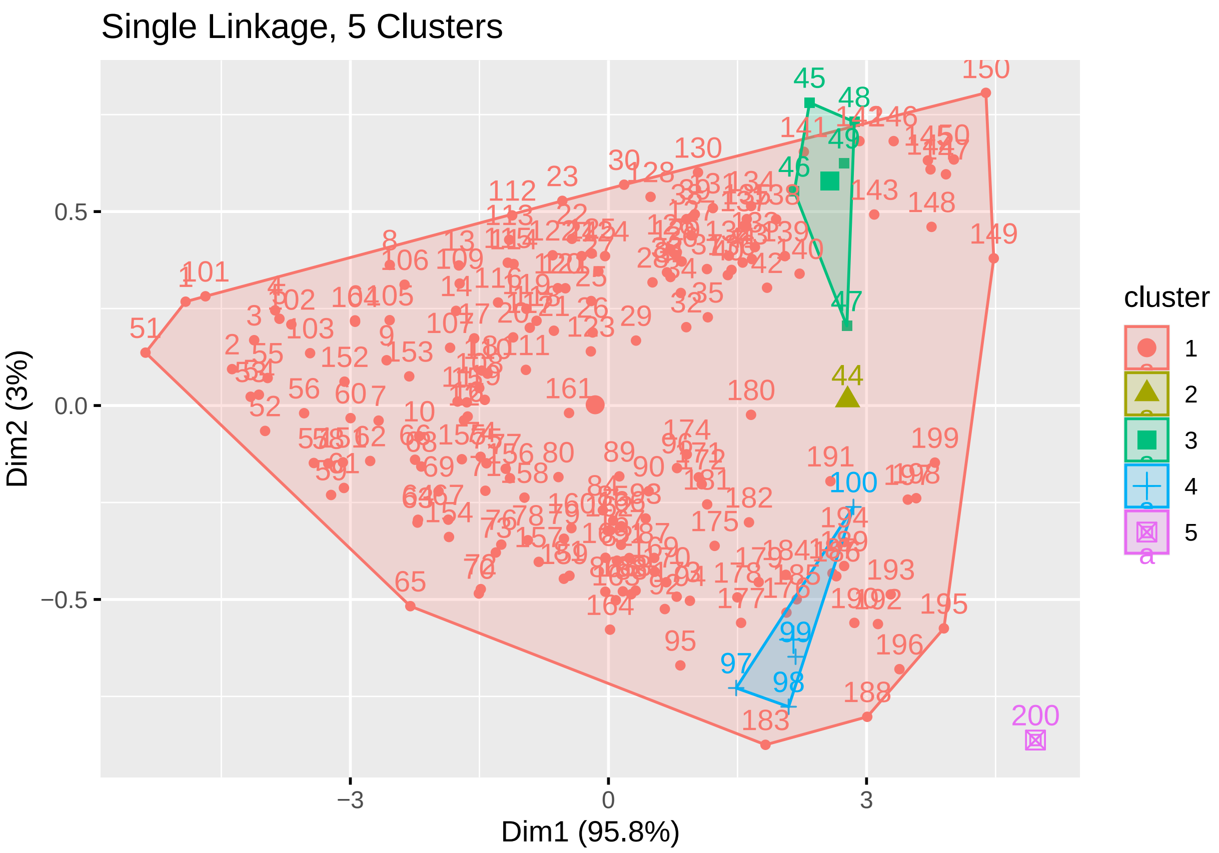


Based on the dendogram, it seems that three clusters is a good cut off point. This is due to the large heights we observe between potential clusters until we get past the third "row".

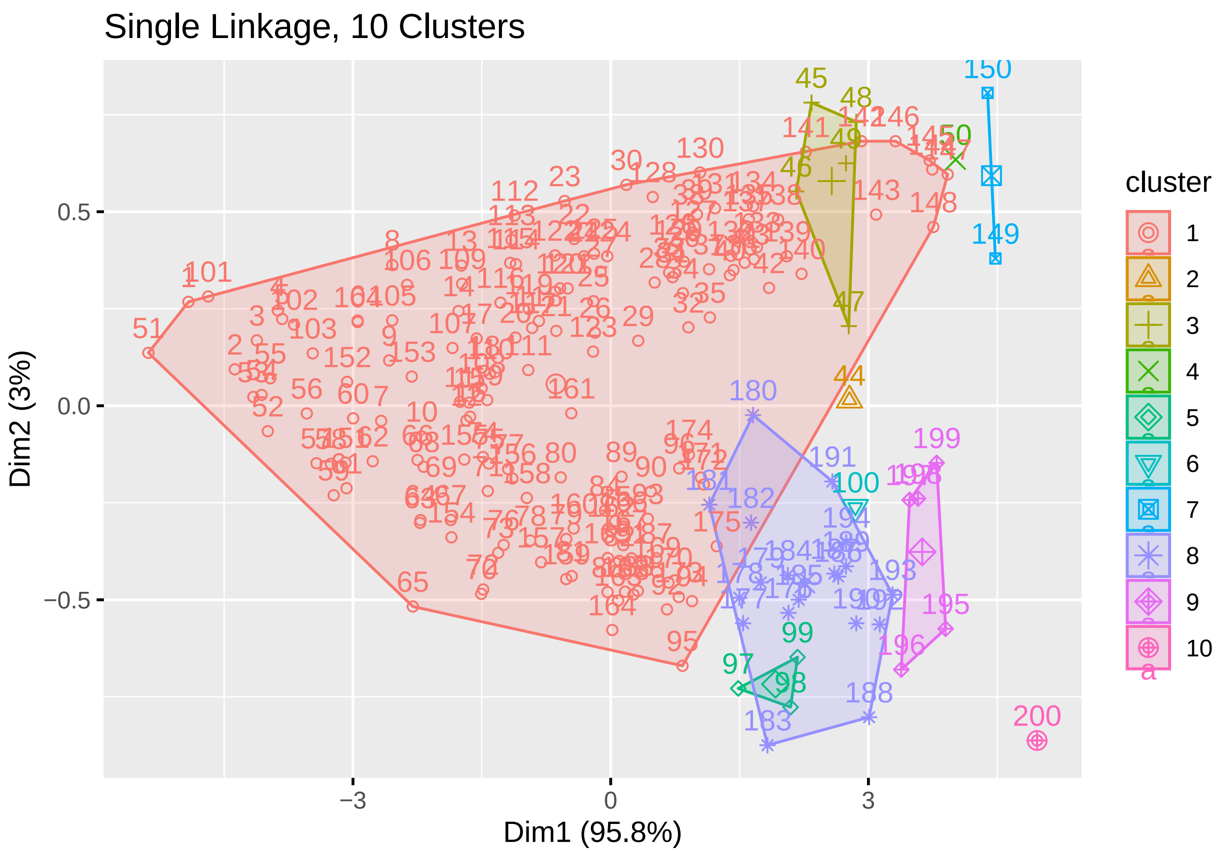
```
# evaluating alternative cutpoints for the number of clusters
factoextra::fviz_cluster(list(data = crabs.stdz, cluster = cutree(hcl.crabz, 3)),
    choose.vars = c(4:8), main = "Single Linkage, 3 Clusters")
```



```
factoextra::fviz_cluster(list(data = crabs.stdz, cluster = cutree(hcl.crabz, 5)),
    choose.vars = c(4:8), main = "Single Linkage, 5 Clusters")
```



```
factoextra::fviz_cluster(list(data = crabs.stdz, cluster = cutree(hcl.crabz, 10)),
    choose.vars = c(4:8), main = "Single Linkage, 10 Clusters")
```



#Q4.

Now, determine the optimal number of clusters based on a criterion: compute the ratio C(g)=(Σmsb)/(Σmsw) and choose the g such that C(g) is maximized (DISPLAY YOUR RESULTS IN A TABLE OR PLOT). We will use package, NbClust, which will compute C(g) - as index 'ch'.

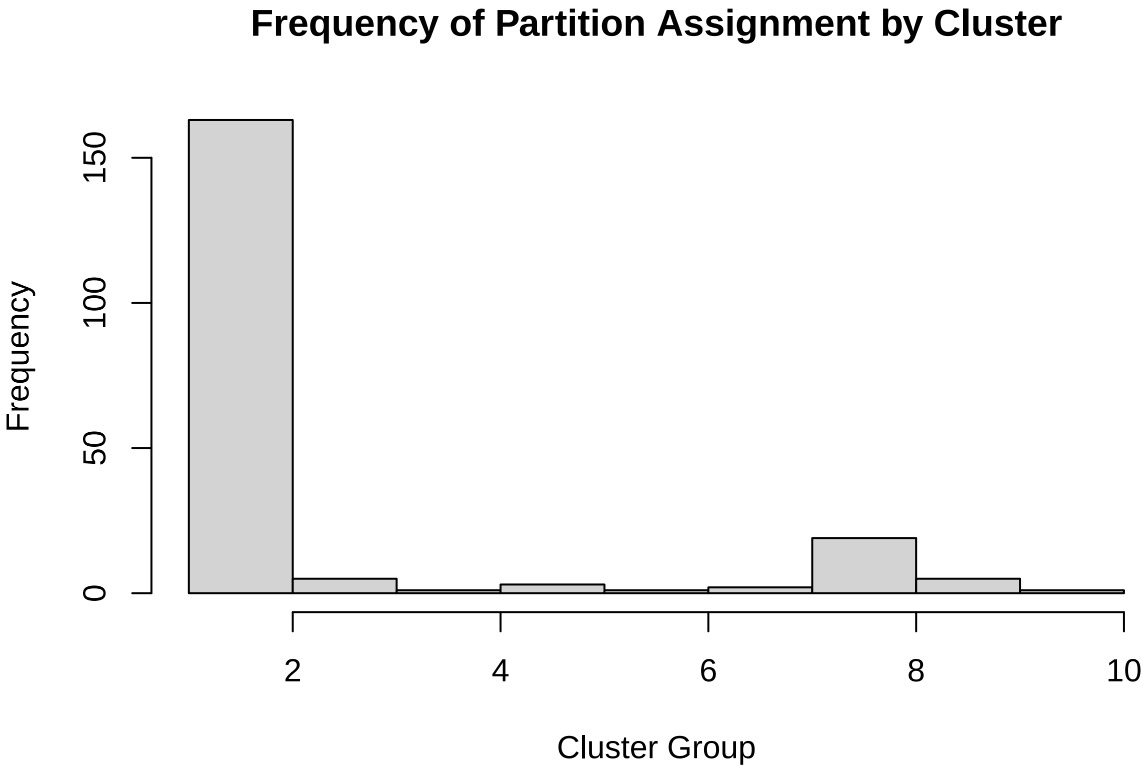
```
# compute the ratio
optimal.crabz <- NbClust(crabs.stdz, min.nc = 2, max.nc = 10, method = "single",
    index = "ch")
# table of C(g) results
cg_index <- optimal.crabz[["All.index"]]
cg_table <- data.frame(Cluster = 2:10, Value = cg_index)
cg_table <- as_tibble(cg_table)
cg_table
```

```
## # A tibble: 9 x 2
##   Cluster Value
##     <int> <dbl>
## 1       2  2.96
## 2       3  8.61
## 3       4  5.73
## 4       5  7.62
## 5       6  6.07
## 6       7  7.84
## 7       8 10.0
## 8       9  9.56
## 9      10 10.1
```

```
# table of Parition Results
partition_single <- optimal.crabz[["Best.partition"]]
table(partition_single)
```

```
## partition_single
##   1   2   3   4   5   6   7   8   9  10
## 162   1   5   1   3   1   2  19   5   1
```

```
# plot Partition Results as Frequency
hist(partition_single, main = "Frequency of Partition Assignment by Cluster", xlab = "Cluster Group",
    )
```



Based on the criterion and utilizing single linkage clustering, the optimal number of clusters is 10 as it has the greatest index value of 10.0618.

#Q5.

As a comparison approach, redo the analysis using k-means clustering. To be consistent use the NbClust package, and extract the `Best.partition` for the result. Use NbClust to SEARCH FOR optimal number of clusters for this method, again determined by C(g).

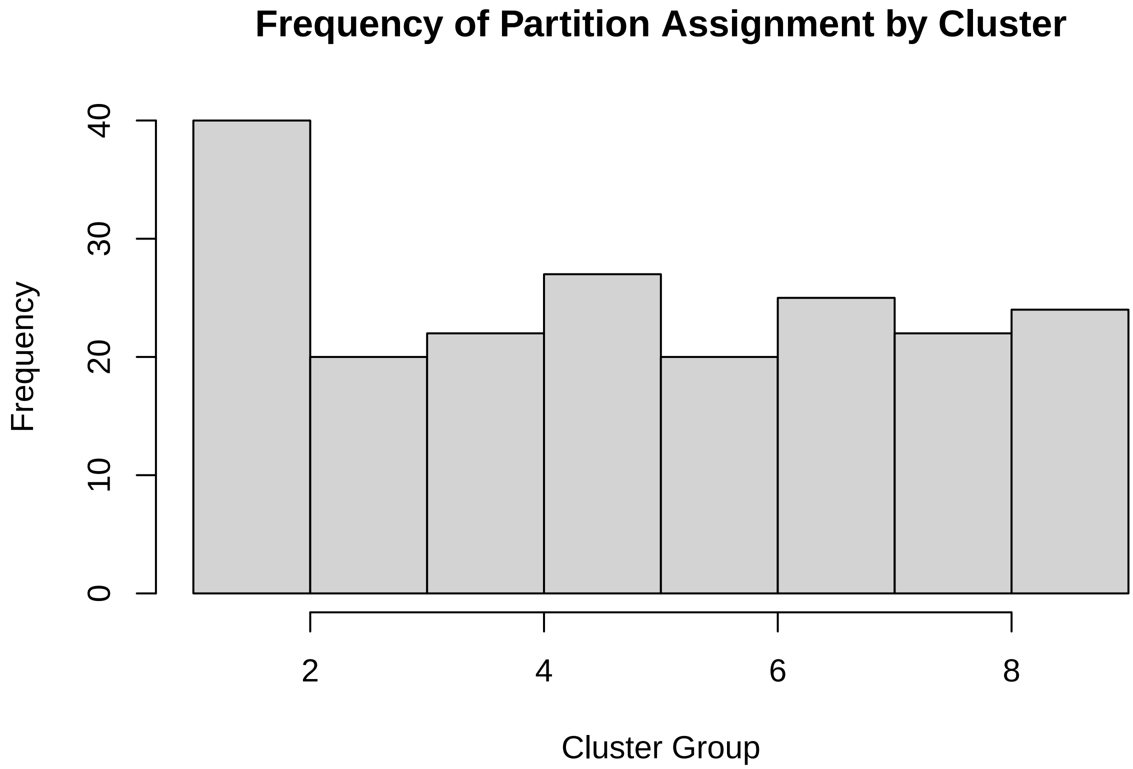
```
optimal.crabz_kmeans <- NbClust(crabs.stdz, min.nc = 2, max.nc = 10, method = "kmeans",
    index = "ch")
# table of C(g) rresults
cg_index_kmeans <- optimal.crabz_kmeans[["All.index"]]
cg_table_kmeans <- data.frame(Cluster = 2:10, Value = cg_index_kmeans)
cg_table_kmeans <- as_tibble(cg_table_kmeans)
cg_table_kmeans
```

```
## # A tibble: 9 x 2
##   Cluster Value
##     <int> <dbl>
## 1       2  576.
## 2       3  732.
## 3       4  854.
## 4       5  980.
## 5       6 1059.
## 6       7 1122.
## 7       8 1105.
## 8       9 1187.
## 9      10 1183.
```

```
# table of Parition Results
partition_kmeans <- optimal.crabz_kmeans[["Best.partition"]]
table(partition_kmeans)
```

```
## partition_kmeans
##  1  2  3  4  5  6  7  8  9
## 20 20 20 22 27 20 25 22 24
```

```
# plot Partition Results as Frequency
hist(partition_kmeans, main = "Frequency of Partition Assignment by Cluster", xlab = "Cluster Group",
    )
```



Based on the criterion and utilizing kmeans clustering, the optimal number of clusters is 9 as it has the greatest index value of 1187.3938.

#Q6.

Compare the results from these last two methods, e.g., optimal using C(g) and centroid or single linkage depending on your prior choice and the optimal k-means result.

i) Use a crosstab comparison.

```
# optimal single
crabz_single.10 <- cutree(hclust(crabs.stdz_dist, meth = "single"), 10)

# optimal kmeans
crabz_kmeans.9 <- kmeans(crabs.stdz, 9)

# compare
compare <- xtabs(~crabz_single.10 + crabz_kmeans.9$cluster)
compare
```

```
##                crabz_kmeans.9$cluster
## crabz_single.10  1  2  3  4  5  6  7  8  9
##              1  20 18 21 18 20 21 18  8 18
##              2   0  0  0  0  0  0  0  1  0
##              3   0  0  0  0  0  0  0  5  0
##              4   0  0  0  0  0  0  0  1  0
##              5   0  0  0  0  0  0  0  3  0
##              6   0  0  0  0  0  0  0  1  0
##              7   0  0  0  0  0  0  0  2  0
##              8   0  0  7  5  0  0  6  1  0
##              9   0  0  0  0  0  0  0  5  0
##              10  0  0  0  0  0  0  0  1  0
```

ii) State the maximal agreement between methods (and justify using the crosstab).

```
rand_index <- phyclust::RRand(crabz_single.10, crabz_kmeans.9$cluster)
rand_index
```

```
##     Rand   adjRand   Eindex
## 0.377035 0.009486 0.144970
```

The agreement between the methods is low with Rand at 0.377 and adjRand at 0.009. This indicates that while there is some agreement, once we adjust, there is almost no agreement between these methods.

iii) Evaluate the distribution of the known demographics (sex, species) for the k-means cluster solution (you can use a crosstab here as well). Do the clusters seem to divide in a manner consistent with demographic differences? Justify your answer by comparing the frequency distribution of demographics within each cluster.

```
# compare sex and species
crabs$kmeans_cluster <- crabz_kmeans.9$cluster

sex_compare <- xtabs(~sex + kmeans_cluster, data = crabs)
sex_compare
```

```
##    kmeans_cluster
## sex  1  2  3  4  5  6  7  8  9
##   1 10  9 14 11 10 11 12 14  9
##   2 10  9 14 12 10 10 12 14  9
```

```
species_compare <- xtabs(~species + kmeans_cluster, data = crabs)
species_compare
```

```
##        kmeans_cluster
## species  1  2  3  4  5  6  7  8  9
##       1 10  8 14 11 10 11 12 14 10
##       2 10 10 14 12 10 10 12 14  8
```

The clusters seem to divide in a way consistent with the differences. There is an even split across the board in sex and species which is also seen in the frequency distribution of the variables.