BSG-MDS practical 4 Population Substructure

28/11/2023, submission deadline 05/12/2023

Pyry Satama Max de Visser

```
library(data.table)
library(MASS)
```

```
df <- fread("Chr21.dat")
df <- data.frame(df, header = TRUE)
df <- df[, (7:ncol(df))] # removed first 6 columns
```

1. How many variants are there in this database? What percentage of the data is missing?

```
n_individuals <- nrow(df)
n_SNPs <- ncol(df)
cat("Number of individuals:", n_individuals)
```

```
## Number of individuals: 203
```

```
cat("\nNumbre of SNPs:", n_SNPs)
```

```
##
## Numbre of SNPs: 138107
```

```
n_missing <- sum(is.na(df))
total_data_points <- n_individuals * n_SNPs
perc_missing <- (n_missing / total_data_points)*100
cat("\nPercentages missing data:", perc_missing, "%")
```

```
##
## Percentages missing data: 0 %
```

2. Compute the Manhattan distance matrix between the individuals (which is identical to the Minkowsky distance with parameter lambda = 1) using R function dist. Include a submatrix of dimension 5 by 5 with the distances between the first 5 individuals in your report.

```
manhattan_distance_matrix <- dist(df, method="manhattan")
submatrix <- dist(df[1:6, ], method="manhattan")
submatrix
```

```
##       1     2     3     4     5
```

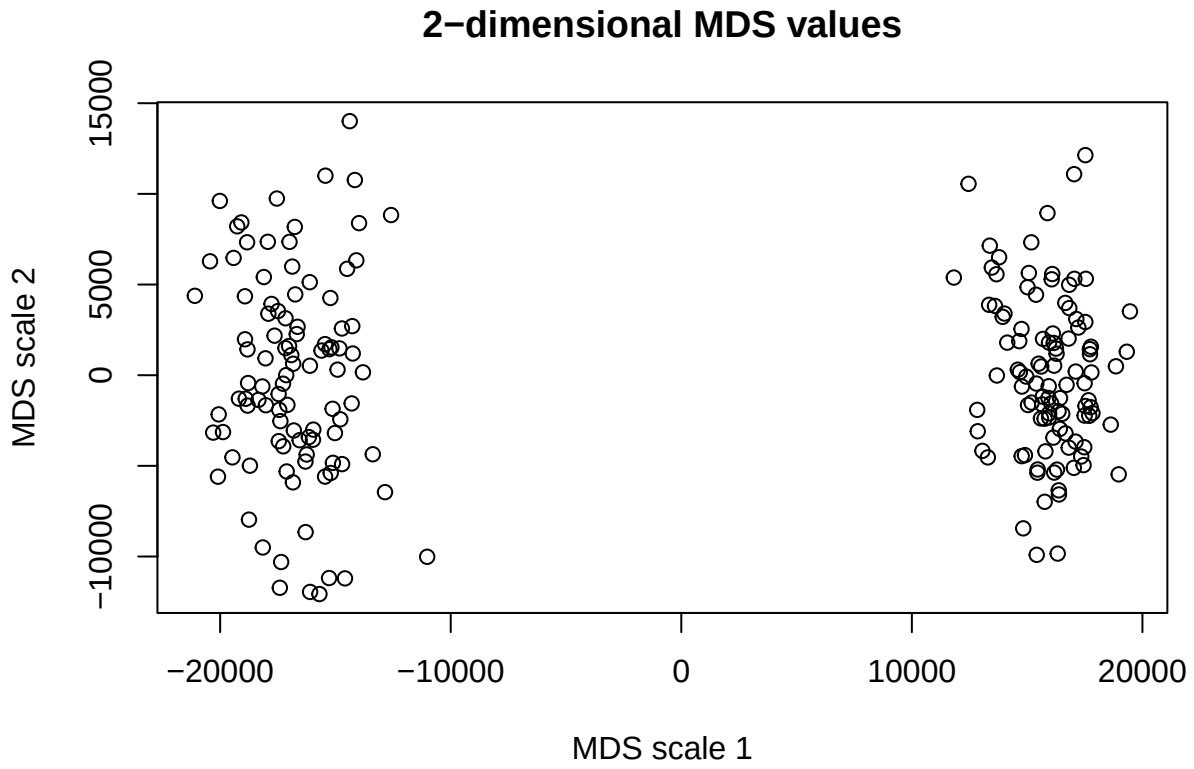
```
## 2 53495
## 3 55007 55372
## 4 58174 55995 54815
## 5 53794 55699 55683 59046
## 6 53675 52682 53790 57075 55565
```

3. How does the Manhattan distance relate to the allele sharing distance?

The Manhattan distance is inversely related to allele sharing distance

4. Apply metric multidimensional scaling (cmdscale) with two dimensions, $k = 2$, using the Manhattan distance matrix and include the map in your report. Do you think the data come from one homogeneous human population? If not, how many subpopulations do you think the data might come from, and how many individuals pertain to each suppopulation?

```r
mds <- cmdscale(manhattan_distance_matrix, k = 2, eig=TRUE) # metric MDS
points <- mds$points
plot(points[, 1], points[, 2], main="2-dimensional MDS values", xlab="MDS scale 1", ylab="MDS scale 2")
```



```r
n_sub1 <- points[points[, 1] > 10000]
n_sub2 <- points[points[, 1] < 10000]
cat("In the first subpopulation there are", length(n_sub1), "individuals")
```

```
## In the first subpopulation there are 208 individuals
```

```r
cat("\nIn the second subpopulation there are", length(n_sub2), "individuals")
```

```
##
## In the second subpopulation there are 198 individuals
```

By inspecting the scatterplot above, there is a clear visual pattern suggesting that the data contains 2 sub-populations, as a significant part of data points are clustering on either side, so we are not just talking about outliers. There are 208 individuals in the right cluster and 198 in the left cluster.

5. What is the goodness-of-fit of the two-dimensional approximation to your distance matrix? Explain which criterium you have used.

Calculating the stress value as a goodness-of-fit measure:

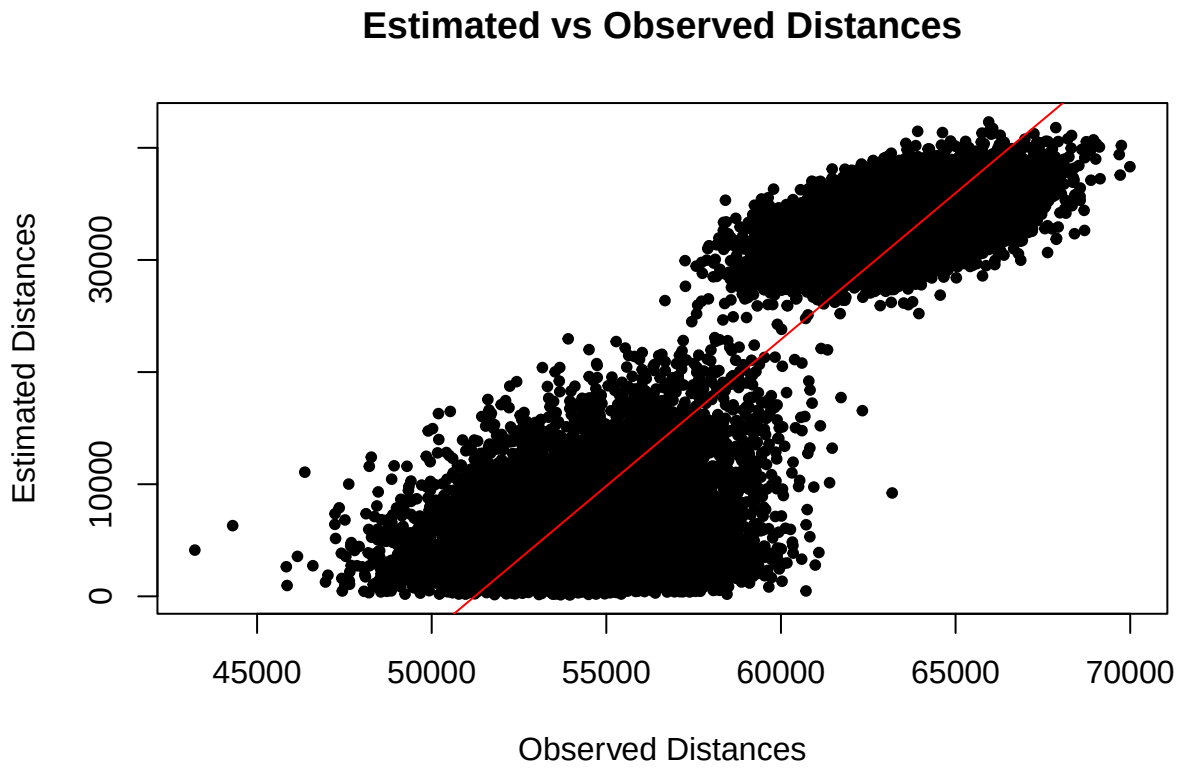
```r
mds2 <- cmdscale(manhattan_distance_matrix, k = 2)
mds_distances <- as.matrix(dist(mds2))
original_distances <- as.matrix(manhattan_distance_matrix)
sum_squared_differences <- sum((mds_distances - original_distances)^2)
sum_squared_original <- sum(original_distances^2)

stress <- sqrt(sum_squared_differences / sum_squared_original)

cat("Stress value:", stress)
```

```
## Stress value: 0.6786808
```

6. Make a plot of the estimated distances (according to your two-dimensional map of individuals) versus the observed distances. What do you observe? Regress estimated distances on observed distances and report the coefficient of determination of the regression (you can use the function lm).

```r
data <- data.frame(
  observed = as.vector(original_distances),
  estimated = as.vector(mds_distances)
)

# since the matrix is symmetric, remove upper triangle (and diagonal which only has values of 0)
data <- data[upper.tri(original_distances), ]

plot(data$observed, data$estimated, main = "Estimated vs Observed Distances",
     xlab = "Observed Distances", ylab = "Estimated Distances", pch = 20)

# Perform linear regression
model <- lm(estimated ~ observed, data = data)
```

```r
# Add regression line to the plot
abline(model, col = "red")
```



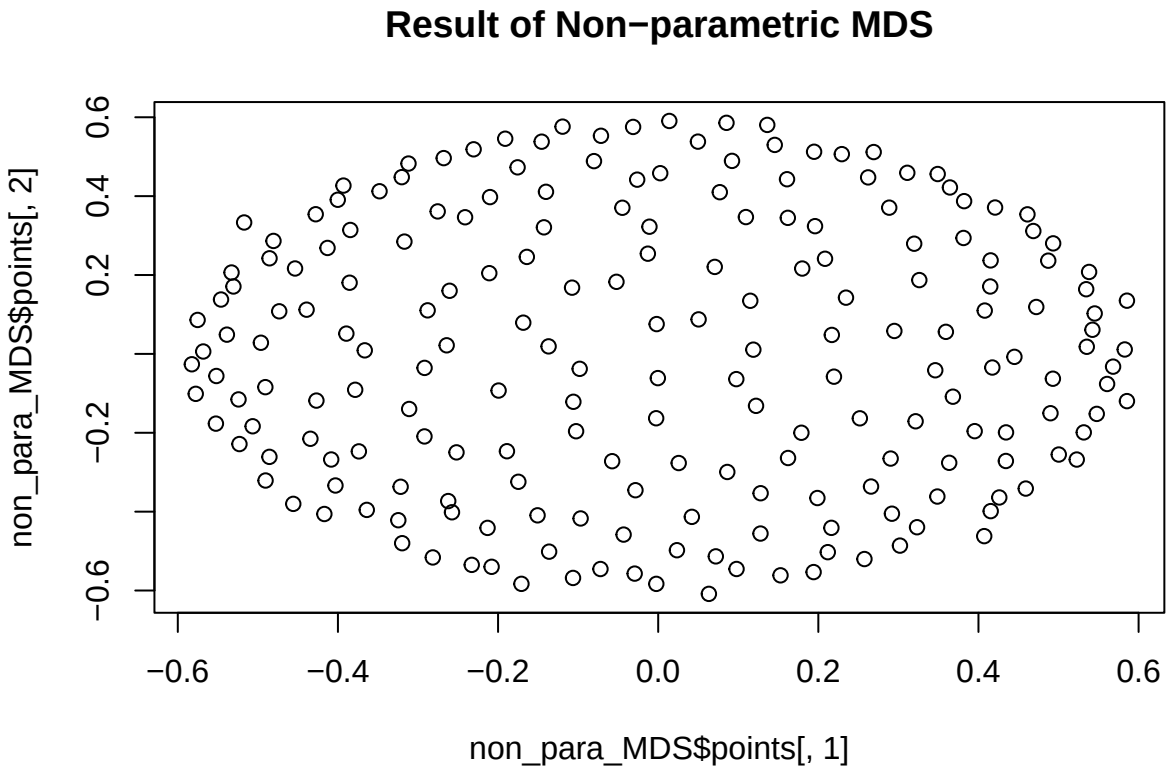

```r
cat("Coefficient of determination", summary(model)$r.squared)
```

```
## Coefficient of determination 0.8428413
```

The scatter plot indicates a positive linear relationship between estimated and observed distances. The estimated distances are on a smaller scale than the observed ones, implying that the two-dimensional MDS approximation compresses the distances.

7. We now try a (two-dimensional) non-metric multidimensional scaling using the isoMDs function that you will find in MASS library. We use a random initial configuration and, for the sake of reproducibility, make this random initial configuration with the instructions: set.seed(12345) and init <- scale(matrix(runif(m*n),ncol=m),scale=FALSE) where n represents the sample size and m represents the dimensionality of the solution. Make a plot of the two-dimensional solution. Do the results support that the data come from one homogeneous population?

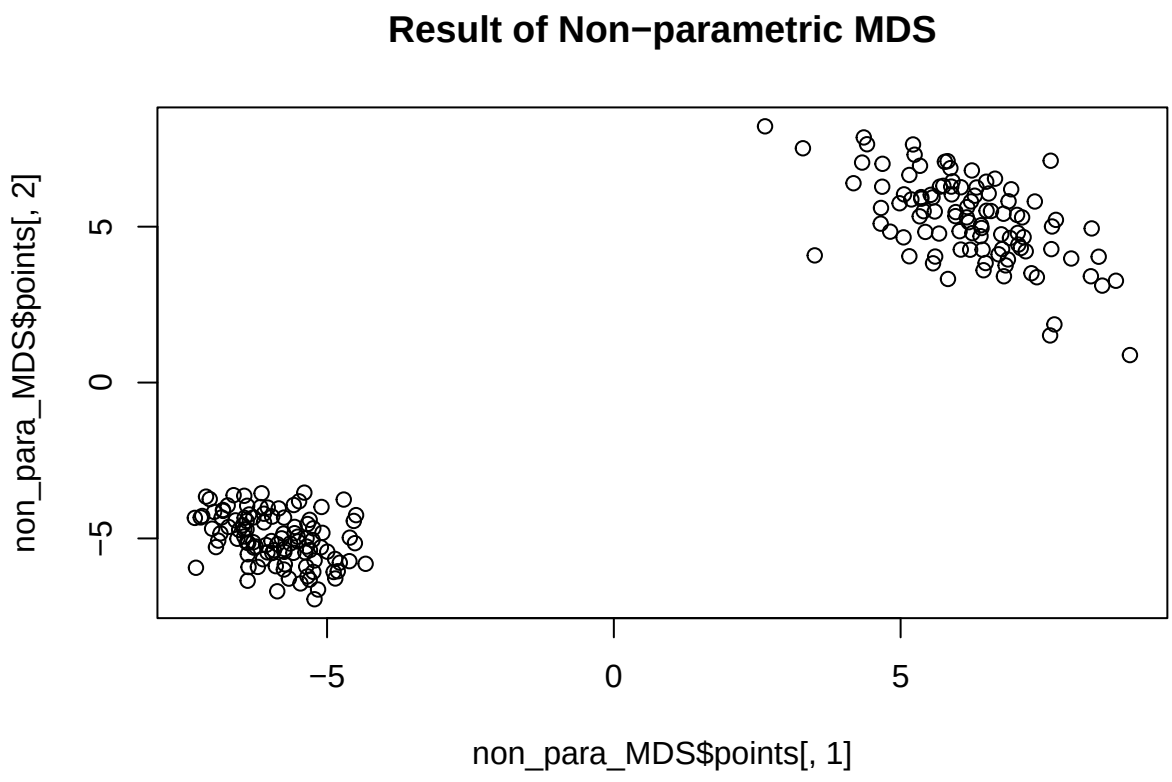
```
set.seed(12345)
m <- 2 # desired number of dimensions in the result
init <- scale(matrix(runif(m*n_individuals),ncol=m),scale=FALSE)
non_para_MDS <- isoMDS(manhattan_distance_matrix, y=init, k=m, trace=FALSE)
#stress <- non_para_MDS$stress
plot(non_para_MDS$points[,1], non_para_MDS$points[,2], main="Result of Non-parametric MDS")
```



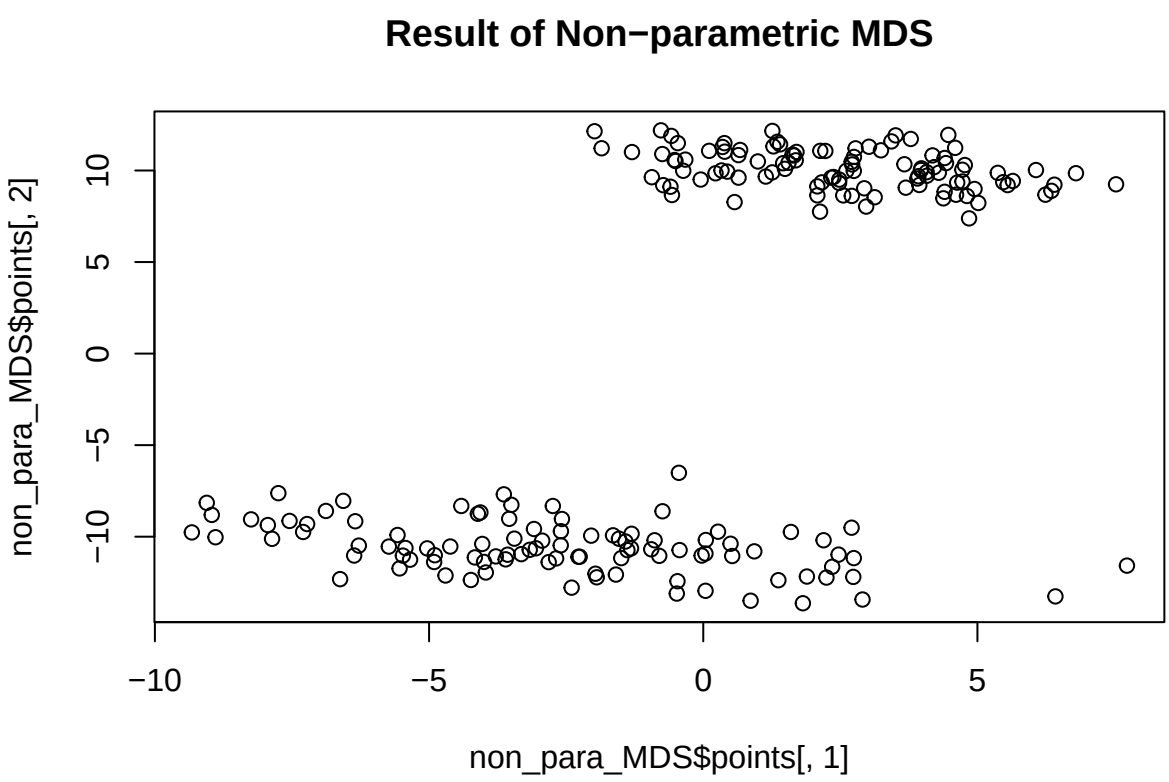
In the case of applying non-parametric MDS we do not observe any inherently clear clusters in the data, and as such the result supports that the data come from a homogeneous population.

8. Try some additional runs of the two-dimensional isoMDS with different initial configurations. Make a plot of the solutions and report the STRESS for each of them. What do you observe?

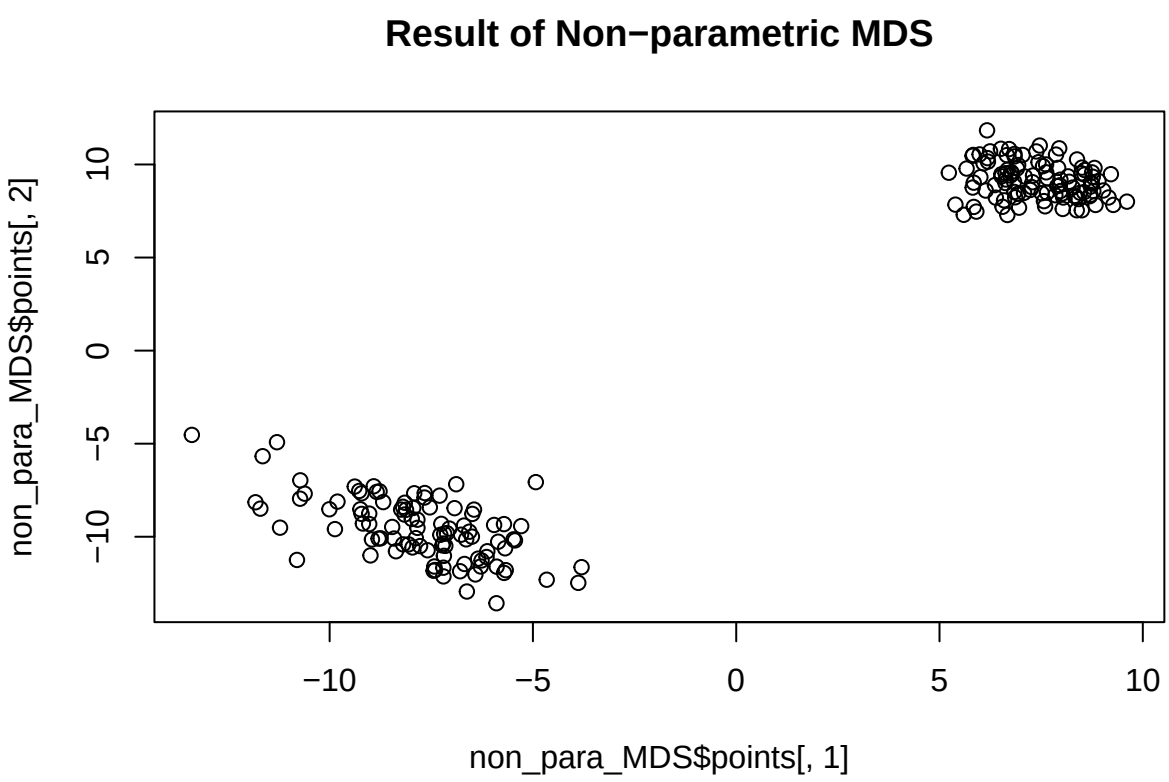
```
non_parametric_MDS <- function(distance_matrix, m, plot) {
  # m is number of dimensions in result
  init <- scale(matrix(runif(m*n_individuals),ncol=m),scale=FALSE)
  non_para_MDS <- isoMDS(manhattan_distance_matrix, y=init, k=m, trace=FALSE)
  stress <- non_para_MDS$stress
  if (plot) {
    plot(non_para_MDS$points[,1], non_para_MDS$points[,2], main="Result of Non-parametric MDS")
  }
  return (stress)
}

for (i in 1:5){
  s <- non_parametric_MDS(manhattan_distance_matrix, 2, plot=TRUE)
  cat("Stress for ", i,"th iteration: ", s, "\n")
}
```



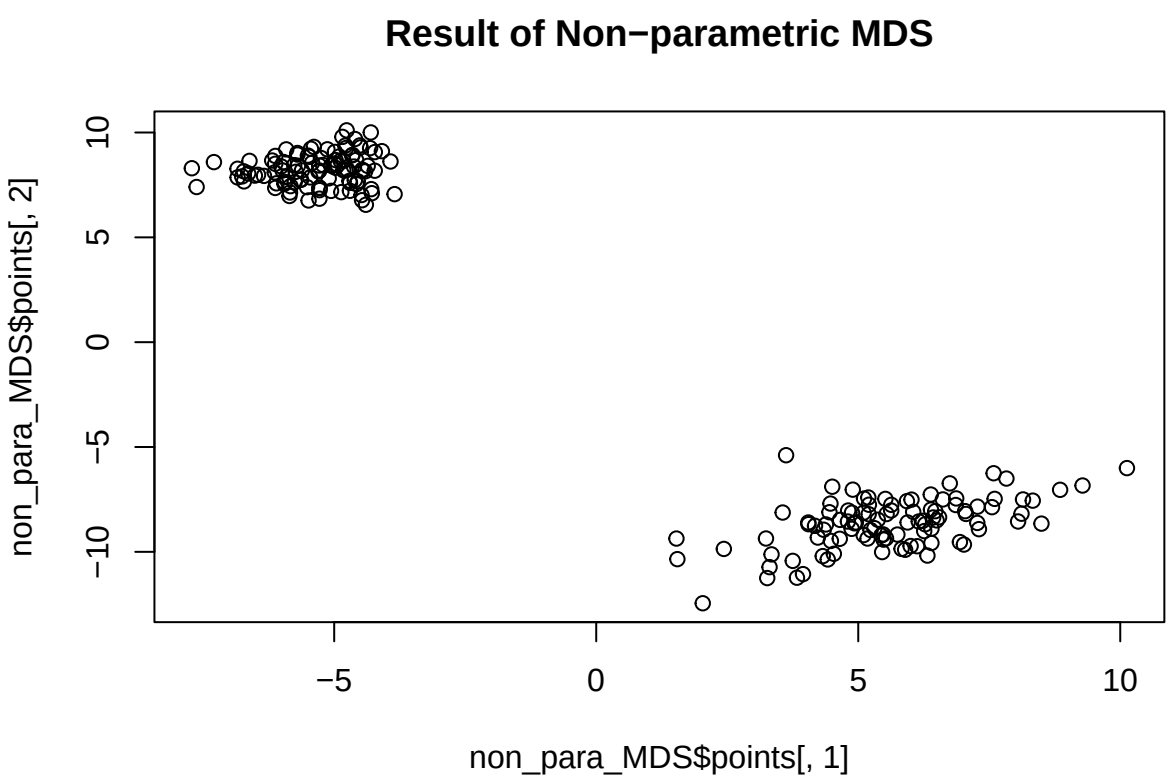

```
## Stress for  1 th iteration:  12.09879
```



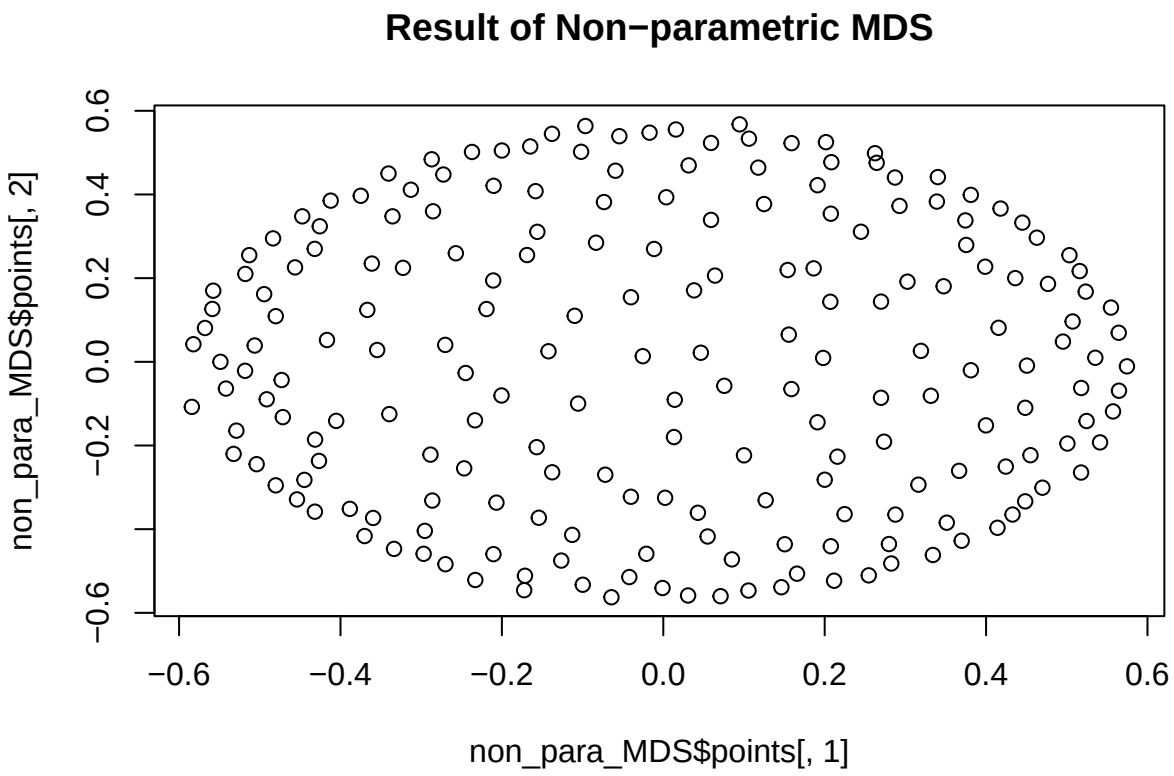

```
## Stress for  2 th iteration:  13.73254
```




```
## Stress for  3 th iteration:  11.85841
```




```
## Stress for  4 th iteration:  11.6411
```

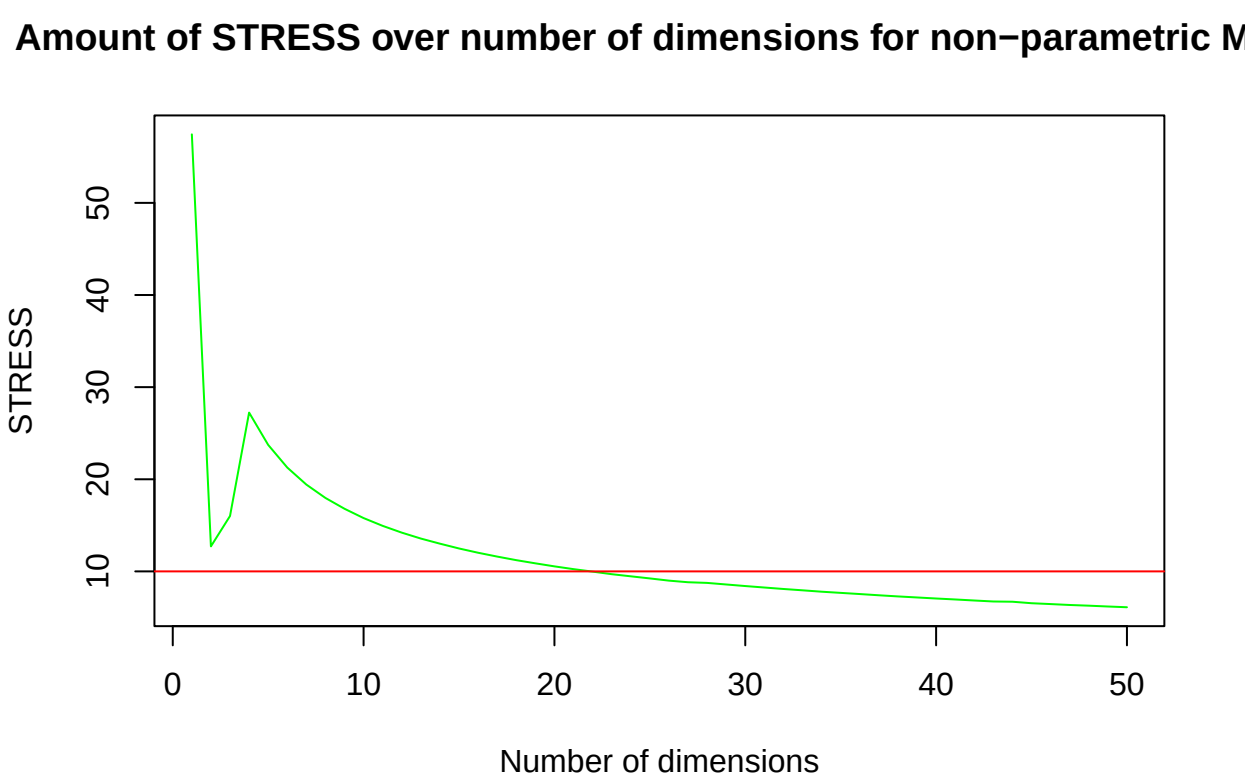



```
## Stress for  5 th iteration:  41.6868
```

There is a lot of variation between different plots which suggests that the dataset has multiple local minima, and the MDS algorithm finds different solutions depending on the initial configuration.

9. Compute the stress for a 1, 2, 3, . . . , 50-dimensional solution. How many dimensions are necessary to obtain a good representation with a stress below 10? Make a plot of the stress against the number of dimensions.

```
nth_stress <-  numeric(50)
for (i in 1:50){
  nth_stress[[i]] <- non_parametric_MDS(manhattan_distance_matrix, m=i, plot=FALSE)
}
plot(1:50, nth_stress, type="l", col="green", main="Amount of STRESS over number of dimensions for non-p
abline(h=10, col = "red")
```



To obtain a solution with a STRESS-value less than 10, the lowest number of dimensions we can reduce to is approximately 21-22. Any lower, and the STRESS-value is above 10, indicating a larger discrepancy between the reduced lower dimensional representation and the original representation.

10. Run the two-dimensional isoMDS a hundred times, each time using a different random initial configuration using the instructions above. Report the stress of the best and the worse run, and plot the corresponding maps. Compare your results to the metric MDS and comment on your findings.

```
run_isoMDS <- function(distance_matrix) {
  # Function to run non-parametric MDS
  init_config <- scale(matrix(runif(2 * n_individuals), ncol = 2), scale = FALSE)
  isoMDS_result <- isoMDS(distance_matrix, y = init_config, k = 2, trace = FALSE)
  return(list(points = isoMDS_result$points, stress = isoMDS_result$stress))
}

best_run <- list(stress = Inf)
worst_run <- list(stress = -Inf)

for (i in 1:100) {
  result <- run_isoMDS(manhattan_distance_matrix)
```

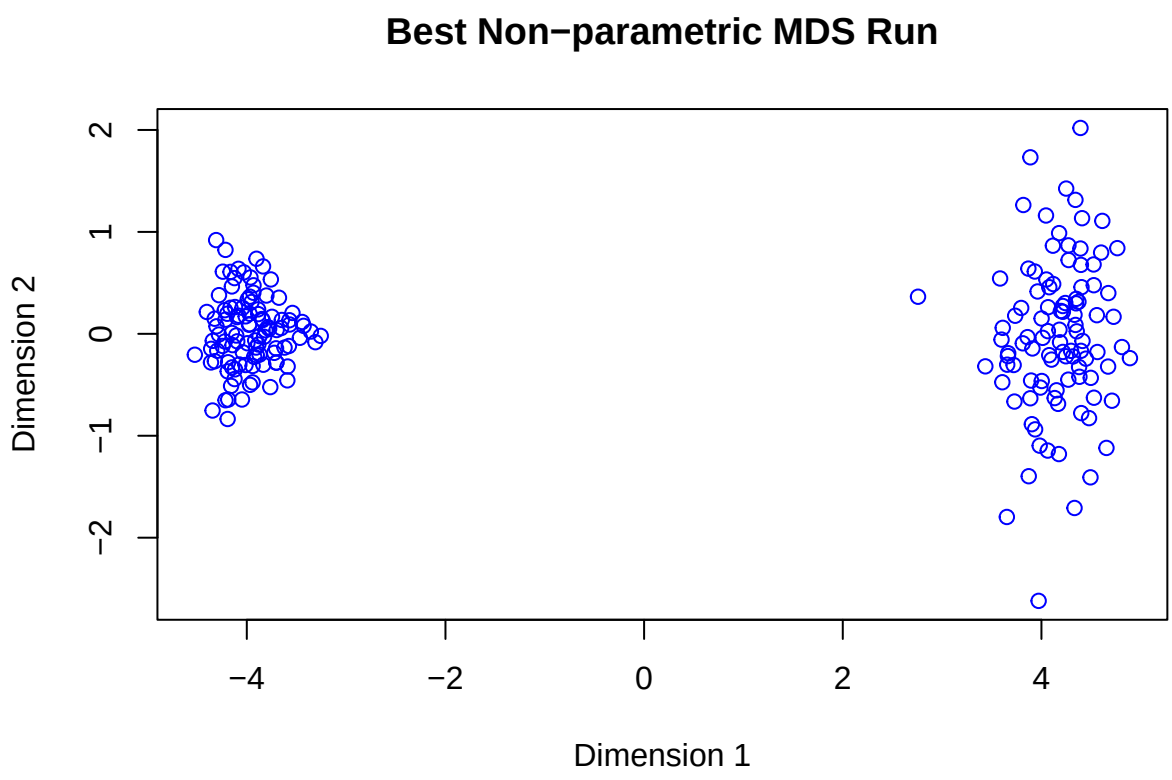
```r
  if (result$stress < best_run$stress) {
    best_run <- result
  }

  if (result$stress > worst_run$stress) {
    worst_run <- result
  }
}

cat("Best run stress: ", best_run$stress, "\n")
```

```
## Best run stress:  11.44118
```

```r
cat("Worst run stress: ", worst_run$stress, "\n")
```

```
## Worst run stress:  42.86588
```

```r
plot(best_run$points[,1], best_run$points[,2], main="Best Non-parametric MDS Run", xlab="Dimension 1", y
```




```r
plot(worst_run$points[,1], worst_run$points[,2], main="Worst Non-parametric MDS Run", xlab="Dimension 1
```

Metric MDS shows a clear pattern. Non-metric MDS with high stress values shows more variability and less defined structure in the plots due to the high sensitivity to initial configurations.

11. Compute the correlation matrix between the first two dimensions of the metric MDS and the two-dimensional solution of your best non-metric MDS. Comment your findings.

```
combined_df <- data.frame(Metric_MDS_x = mds$points[,1],
                          Metric_MDS_y = mds$points[,2],
                          Non_metric_MDS_x = best_run$points[,1],
                          Non_metric_MDS_y = best_run$points[,2])
correlation_matrix <- cor(combined_df, method="pearson")
correlation_matrix[, 1:2][3:4, ] # just to filter duplicate values
```

```
##                  Metric_MDS_x Metric_MDS_y
## Non_metric_MDS_x  -0.99743897 -0.009201842
## Non_metric_MDS_y   0.02530727 -0.361511851
```

There is a high correlation between the x-axis values of the metric and best non-metric MDS run. This supports the claim of there being two sub-populations in the data, as we now have two methods finding similar results; a significantly large distinction of two subpopulations in the data. Although the correlation between y-axis values in both approaches is comparatively low, our primary focus lies on the x-axis, where the discernment of the two subpopulations is notably pronounced.

Given the evident lack of a perfect linear relationship in both cases, it is anticipated that the correlation between x- and y-axis values is very low.