Practical 4

Statistical Genetics: Population Substructure

Anna Putina Marine Mauzeau

2023-12-04

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

```
data.geno <- fread("Chr21.dat")
data.geno <- data.geno[,-c(1:6)]
```

Question 1 :

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

```
NA.counter <- sum(is.na(data.geno))
NA.percentage <- NA.counter*100/prod(dim(data.geno))
```

There are 138106 variants in this data base.

The percentage of missing data in this database: 0 %.

Question 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.dist <- dist(data.geno, method = "manhattan")
sub.mat <- dist(data.geno[c(1:5),], method = "manhattan")
```

The submatrix of dimension 5 by 5 with the distances between the first 5 individuals:

```
print(sub.mat)
```

```
##       1     2     3     4
## 2 53495
## 3 55007 55372
## 4 58174 55995 54815
## 5 53794 55699 55683 59046
```

Question 3 :

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

Concerning 2 different individuals and 1 variant :

- if they have 2 shared alleles : the absolute value of the difference of their genetic data is 0 (0-0 or 1-1 or 2-2).

- if they have 1 shared allele : the absolute value of the difference of their genetic data is 1 (|2-1|, |1-2|, |0-1|, |1-0|).
- if they have 0 shared allele : the absolute value of the difference of their genetic data is 2 (|2-0| or |0-2|).

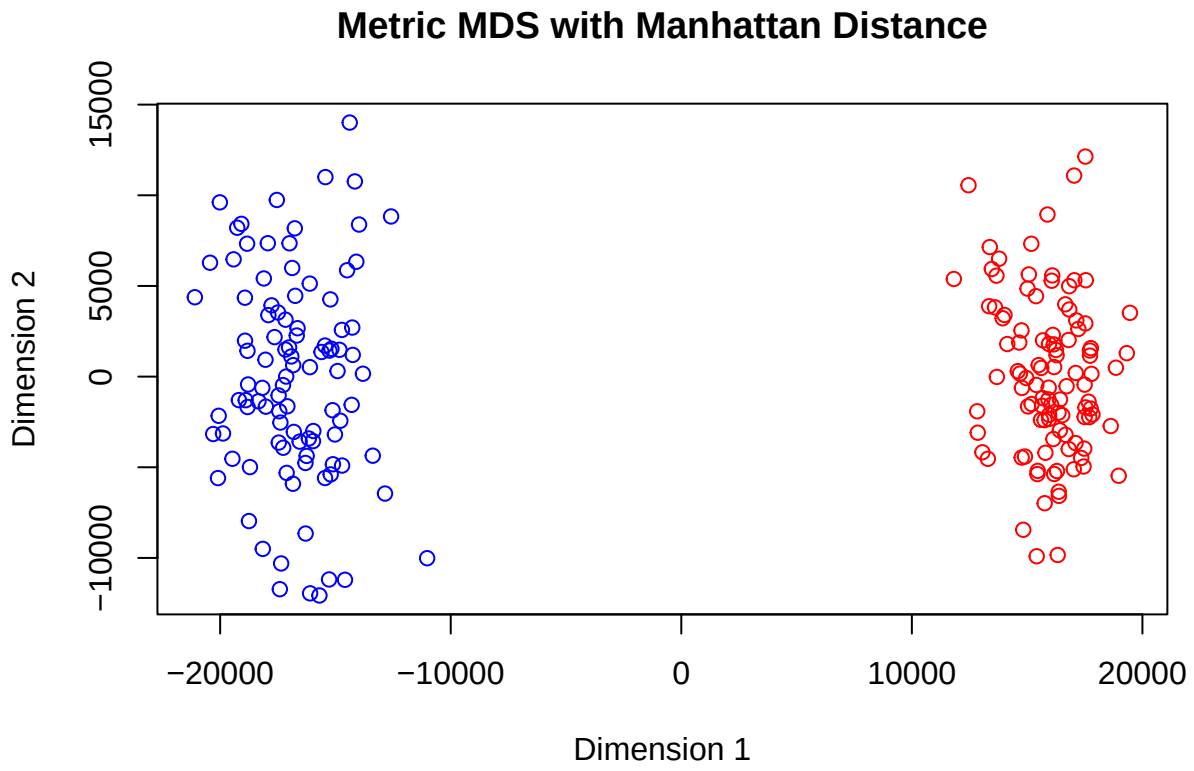
Thus, thanks to the definitions of Manhattan distance matrix and Allele sharing distance matrix, there is only a factor k between the 2 matrices :

Manhattan matrix = number of variants × allele sharing matrix = 138106 × allele sharing matrix.

Question 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?

```
mds.res <- cmdscale(manhattan.dist, list. = TRUE, x.ret = TRUE)

x <- mds.res$points[, 1]
y <- mds.res$points[, 2]
plot(x, y, col = ifelse(x > 0, "red", "blue"), xlab="Dimension 1",
           ylab = "Dimension 2", main = "Metric MDS with Manhattan Distance")
```



According to the plot above, the data doesn't seem to come from one homogeneous population. There clearly seem to be 2 subpopulations : we can easily identify 2 clusters.

```r
# To compute the number of individuals for each class
length(x[x > 0])
```

```
## [1] 104
```

```r
length(x[x < 0])
```

```
## [1] 99
```

There are 104 individuals in the red cluster, and 99 in the blue one.

Question 5 :

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

```r
mds.res$GOF[1]
```

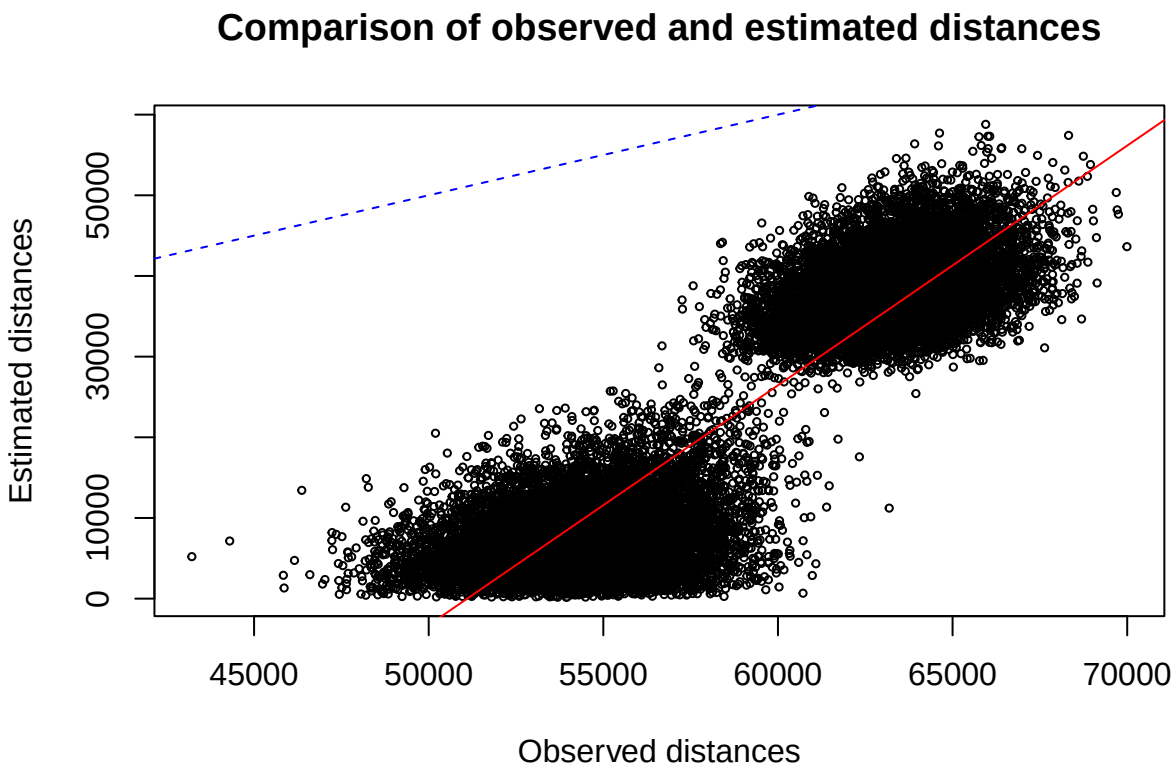
```
## [1] 0.1703581
```

We used criterium $g = \frac{\sum_{j=1}^{2} \lambda_j}{\sum_{j=1}^{203} |\lambda_j|}$. We get a goodness-of-fit equal to 0.1703581.

Question 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
estimated.dist <- dist(mds.res$points, method = "manhattan")

plot(manhattan.dist, estimated.dist, xlab = "Observed distances", ylab = "Estimated distances",
                     main = "Comparison of observed and estimated distances", cex = 0.5)

reg.lin <- lm(estimated.dist ~ manhattan.dist)

abline(reg.lin, col = "red")
abline(a = 0, b = 1, col = "blue", lty = 2)
```



We observe that there hardly seems to be a linear relationship between observed and estimated distances. The 2 clusters are still visible here.

```
summary(reg.lin)
```

```
##
## Call:
## lm(formula = estimated.dist ~ manhattan.dist)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -27903.9  -4377.2    166.1   4674.1  28636.7
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -1.519e+05  5.665e+02  -268.1   <2e-16 ***
## manhattan.dist  2.972e+00  9.589e-03   309.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6755 on 20501 degrees of freedom
## Multiple R-squared:  0.8241, Adjusted R-squared:  0.8241
## F-statistic: 9.605e+04 on 1 and 20501 DF,  p-value: < 2.2e-16
```

The coefficient of determination of the regression is quite high : 0.8241. It means that more than 82% of the data fit the model.

We have a linear dependance, but there is a systematic bias in under-estimating short distances.

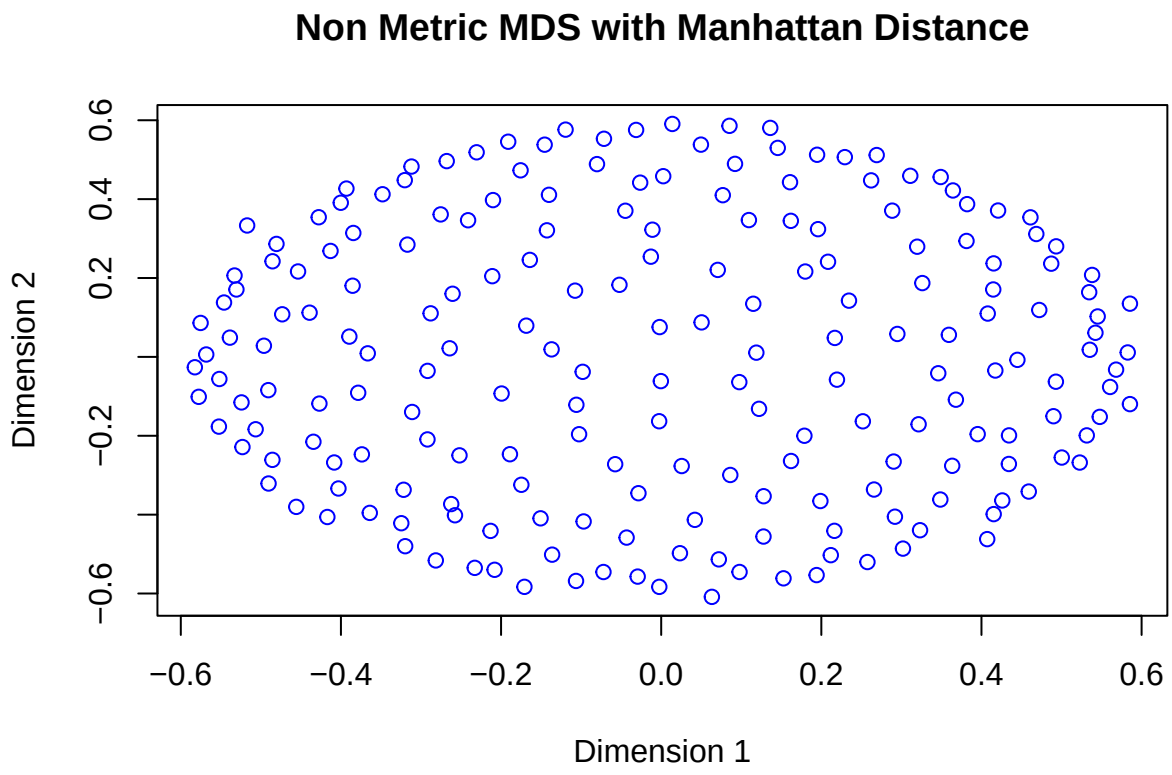
Question 7 :

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
n <- 203
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

```
## initial  value 43.001235
## iter   5 value 41.668593
## iter   5 value 41.629957
## iter   5 value 41.629319
## final  value 41.629319
## converged
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab = "Dimension 1", ylab = "Dimension 2")
```



Here, the results support that the data come from one homogeneous population : we cannot see any visible

cluster.

Question 8 :

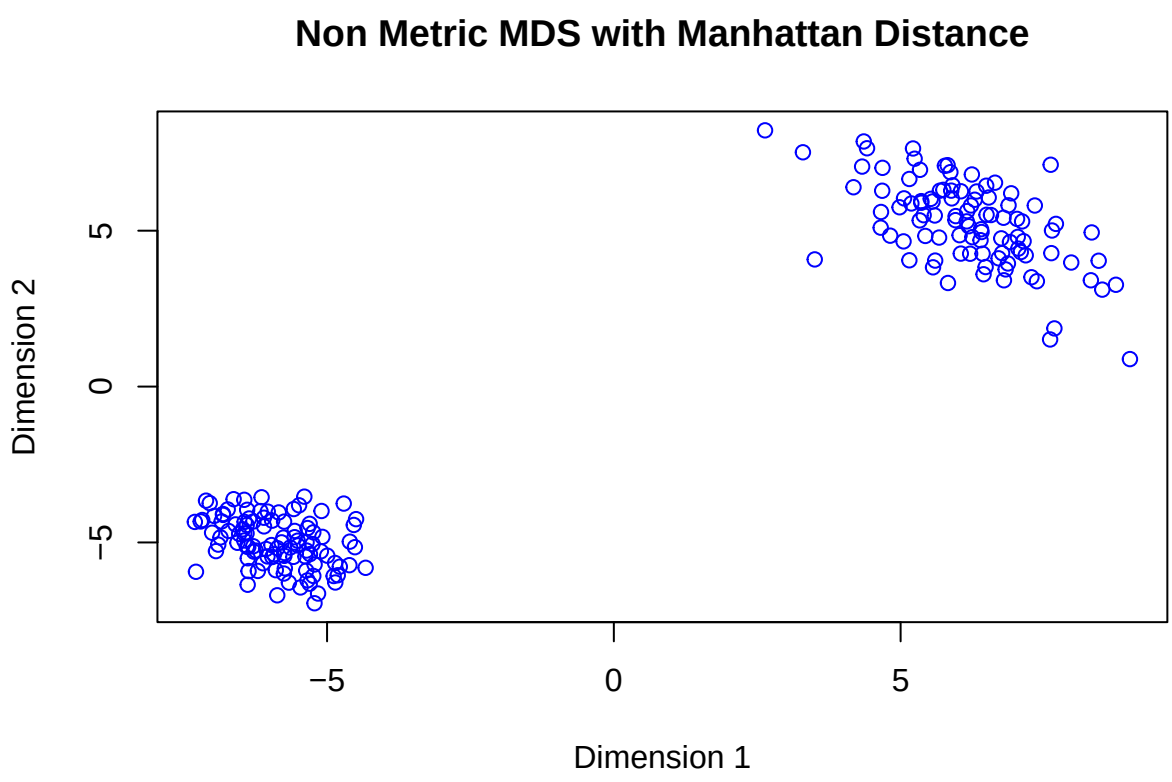
Try some additional runs of the two-dimensional isoMDS with different initial configurations. Make a plot of the solutions and report the STRESS for each of them. What do you observe?

```
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

The 2d attempt

```
## initial  value 42.943761
## iter   5 value 41.427197
## iter  10 value 40.002996
## iter  15 value 38.060966
## iter  20 value 29.659310
## iter  25 value 21.653732
## iter  30 value 17.463856
## iter  35 value 14.418036
## iter  40 value 13.070793
## iter  45 value 12.426749
## iter  50 value 12.098786
## final  value 12.098786
## stopped after 50 iterations
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab="Dimension 1", ylab = "Dimension 2")
```

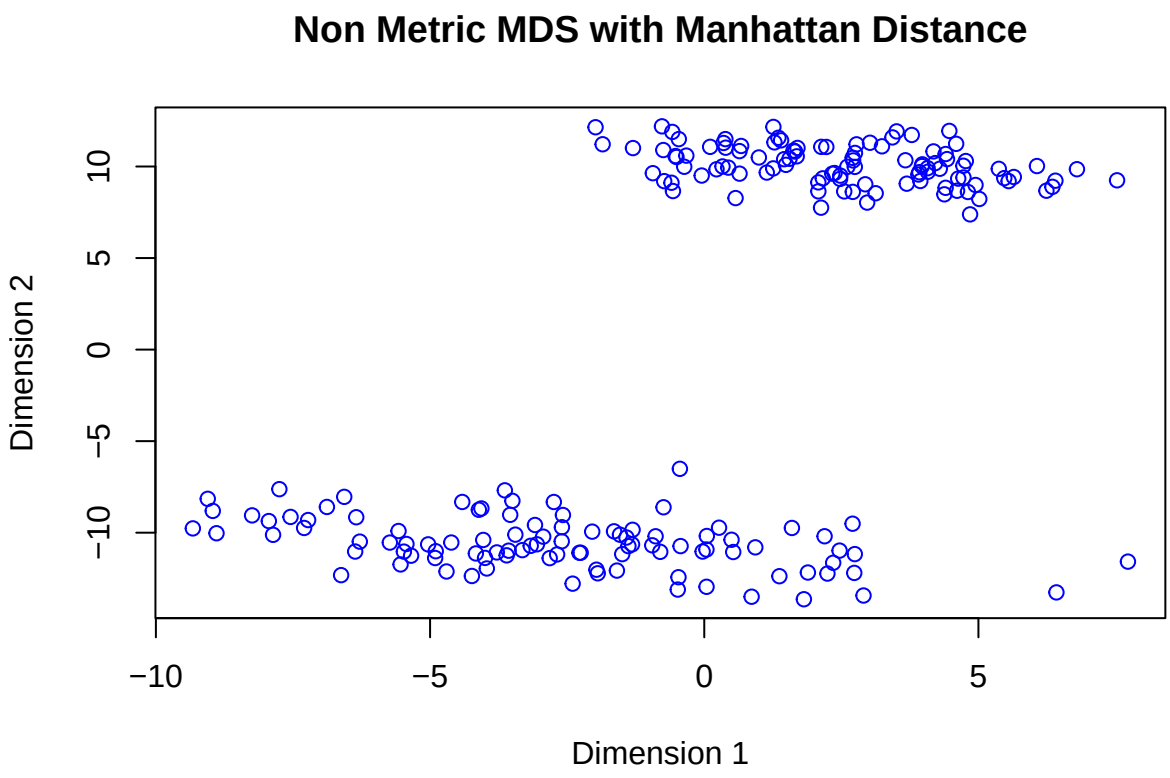



```
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

The 3d attempt

```
## initial  value 42.871770
## iter   5 value 41.643057
## iter  10 value 39.862693
## iter  15 value 39.086758
## iter  20 value 35.726119
## iter  25 value 28.329909
## iter  30 value 22.975542
## iter  35 value 19.446347
## iter  40 value 17.493380
## iter  45 value 15.388138
## iter  50 value 13.732536
## final  value 13.732536
## stopped after 50 iterations
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab="Dimension 1", ylab = "Dimension 2")
```

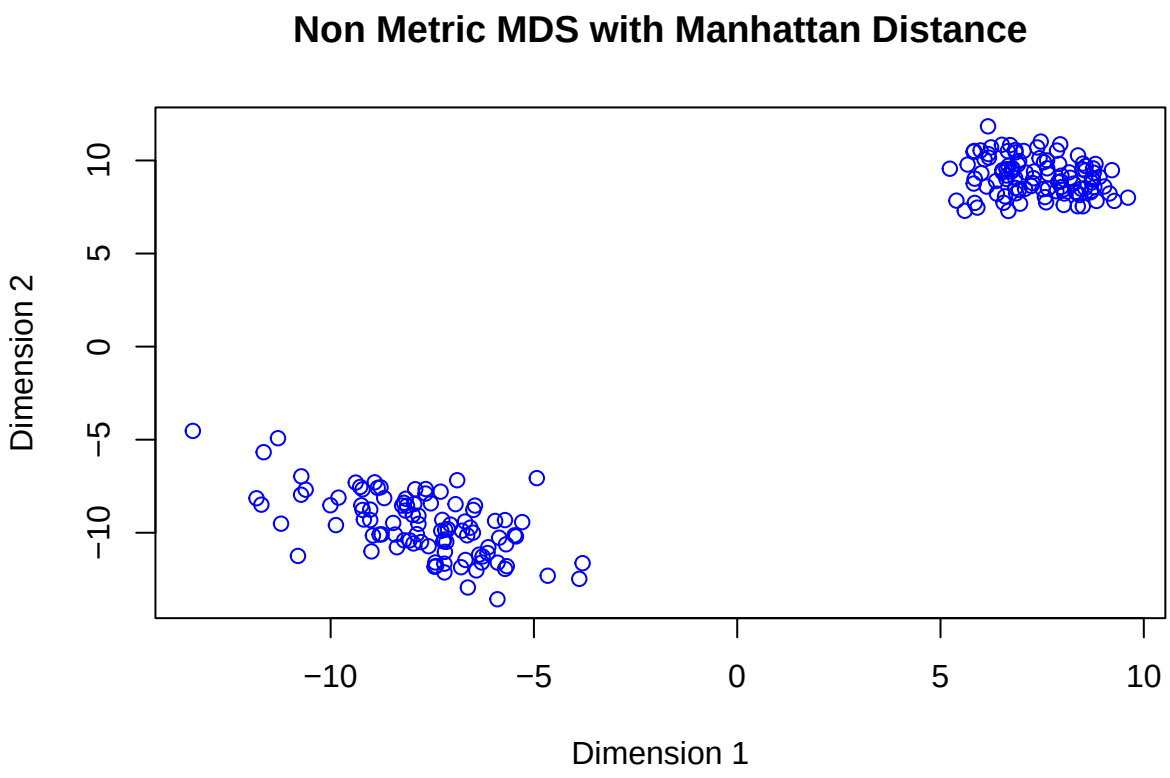



```
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

The 4th attempt

```
## initial  value 42.750591
## iter   5 value 41.580957
## iter  10 value 40.434989
## iter  15 value 38.226909
## iter  20 value 29.256158
## iter  25 value 19.959781
## iter  30 value 16.331637
## iter  35 value 13.632103
## iter  40 value 12.522541
## iter  45 value 12.111644
## final  value 11.858405
## converged
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab="Dimension 1", ylab = "Dimension 2")
```

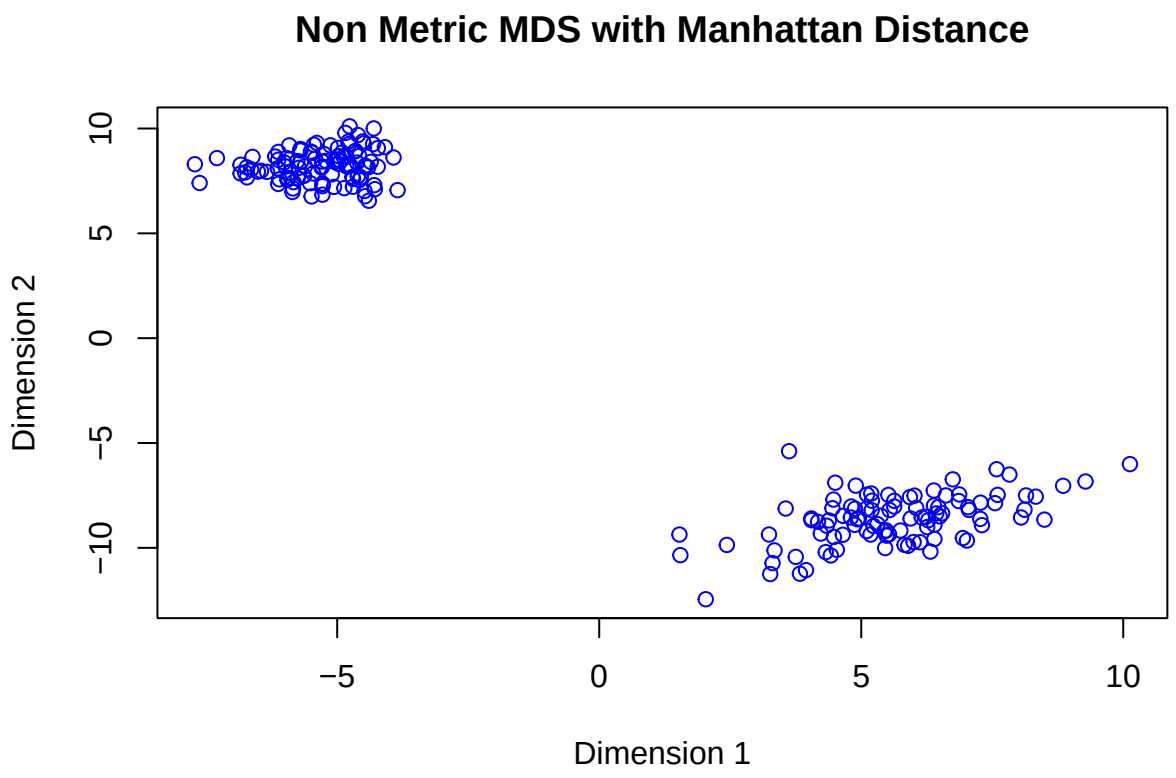



```
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

The 5th attempt

```
## initial  value 42.940739
## iter   5 value 41.589356
## iter  10 value 39.480641
## iter  15 value 35.914108
## iter  20 value 25.072993
## iter  25 value 17.130764
## iter  30 value 14.390847
## iter  35 value 12.968520
## iter  40 value 12.026409
## iter  45 value 11.666159
## final  value 11.641099
## converged
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab="Dimension 1", ylab = "Dimension 2")
```

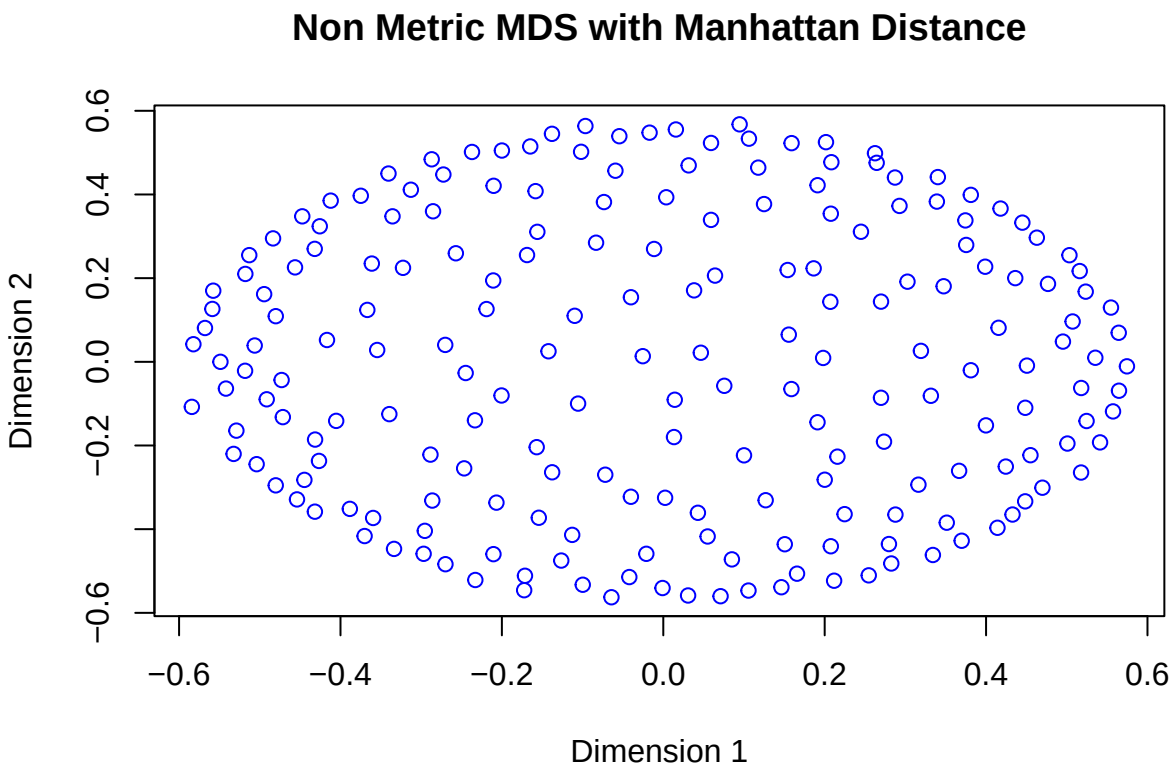



```
init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
nm.mds.res <- isoMDS(manhattan.dist, y = init)
```

The 6th attempt

```
## initial  value 42.741937
## final  value 41.686800
## converged
```

```
x <- nm.mds.res$points[, 1]
y <- nm.mds.res$points[, 2]
plot(x, y, col = "blue", main = "Non Metric MDS with Manhattan Distance",
                          xlab="Dimension 1", ylab = "Dimension 2")
```



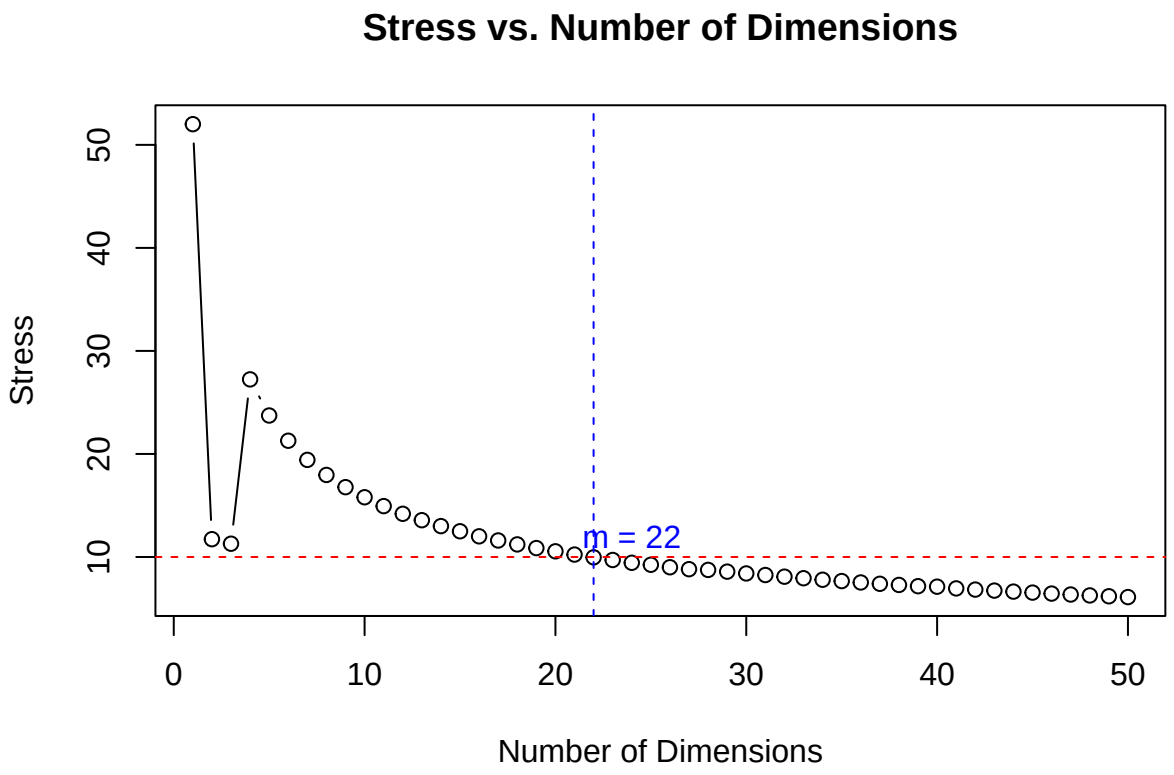
The plots we have obtained demonstrate a strong dependence of the results on the initial configurations. As observed, some configurations exhibit clusterization, while others do not, even for the same dataset.

Question 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.

```
set.seed(12345)
max_dimensions <- 50
stress_values <- numeric(max_dimensions)

for (m in 1:max_dimensions) {
  init <- scale(matrix(runif(m*n), ncol=m), scale=FALSE)
  nm.mds.res <- isoMDS(manhattan.dist, y = init, k = m, trace = FALSE)
  stress_values[m] <- nm.mds.res$stress
}

good_dimensions <- which(stress_values < 10)[1]

plot(1:max_dimensions, stress_values, type = "b",
     xlab = "Number of Dimensions", ylab = "Stress",
     main = "Stress vs. Number of Dimensions")

abline(h = 10, col = "red", lty = 2)
```

```r
abline(v = good_dimensions, col = "blue", lty = 2)
text(good_dimensions + 2, 12, paste("m =", good_dimensions), col = "blue")
```



22 dimensions are necessary to obtain a good representation with a stress below 10.

Question 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.

```r
set.seed(12345)
num_runs <- 100
m <- 2

best_stress <- Inf
worst_stress <- -Inf
best_run <- NULL
worst_run <- NULL

for (i in 1:num_runs) {
  init <- scale(matrix(runif(m * n), ncol = m), scale = FALSE)
  nm.mds.res <- isoMDS(manhattan.dist, y = init, trace = FALSE)
  current_stress <- nm.mds.res$stress
```

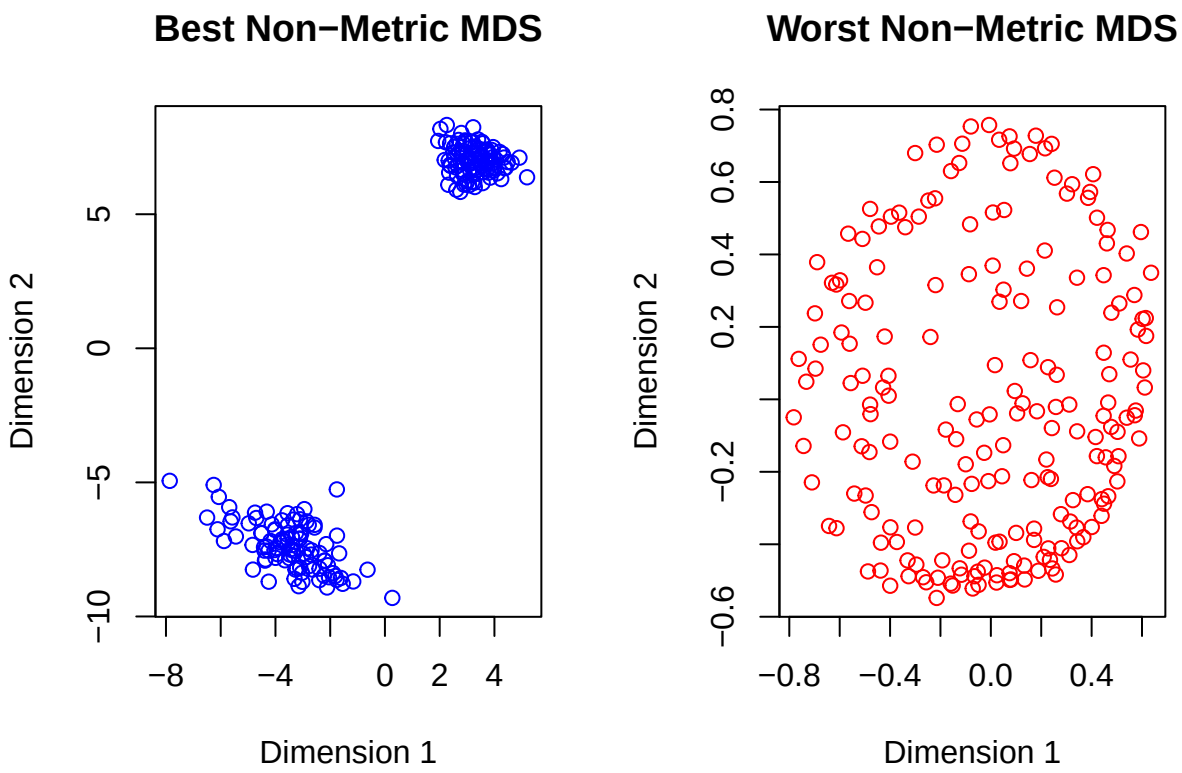
```r
  if (current_stress < best_stress) {
    best_stress <- current_stress
    best_run <- nm.mds.res$points
  }

  if (current_stress > worst_stress) {
    worst_stress <- current_stress
    worst_run <- nm.mds.res$points
  }
}

cat("Best Run Stress:", best_stress, "\n")
```

```
## Best Run Stress: 11.4837
```

```r
cat("Worst Run Stress:", worst_stress, "\n")
```

```
## Worst Run Stress: 42.97238
```

```r
par(mfrow = c(1, 2))

plot(best_run[, 1], best_run[, 2], col = "blue",
     main = "Best Non-Metric MDS", xlab = "Dimension 1", ylab = "Dimension 2")

plot(worst_run[, 1], worst_run[, 2], col = "red",
     main = "Worst Non-Metric MDS", xlab = "Dimension 1", ylab = "Dimension 2")
```



In the worst case scenario for Non-Metric MDS, clusterization is not observed. However, for both the best-performing Non-Metric MDS and Metric MDS, we observe clusterization. Consequently, Metric MDS appears to be more stable in these terms.

In the case of the same dataset and k=2, we noticed that for the Metric MDS the two clusters are segregated along only one dimension — either positive or negative x. On the contrary, in the case of Non-Metric MDS, we found that the clusters are separated along both dimensions.

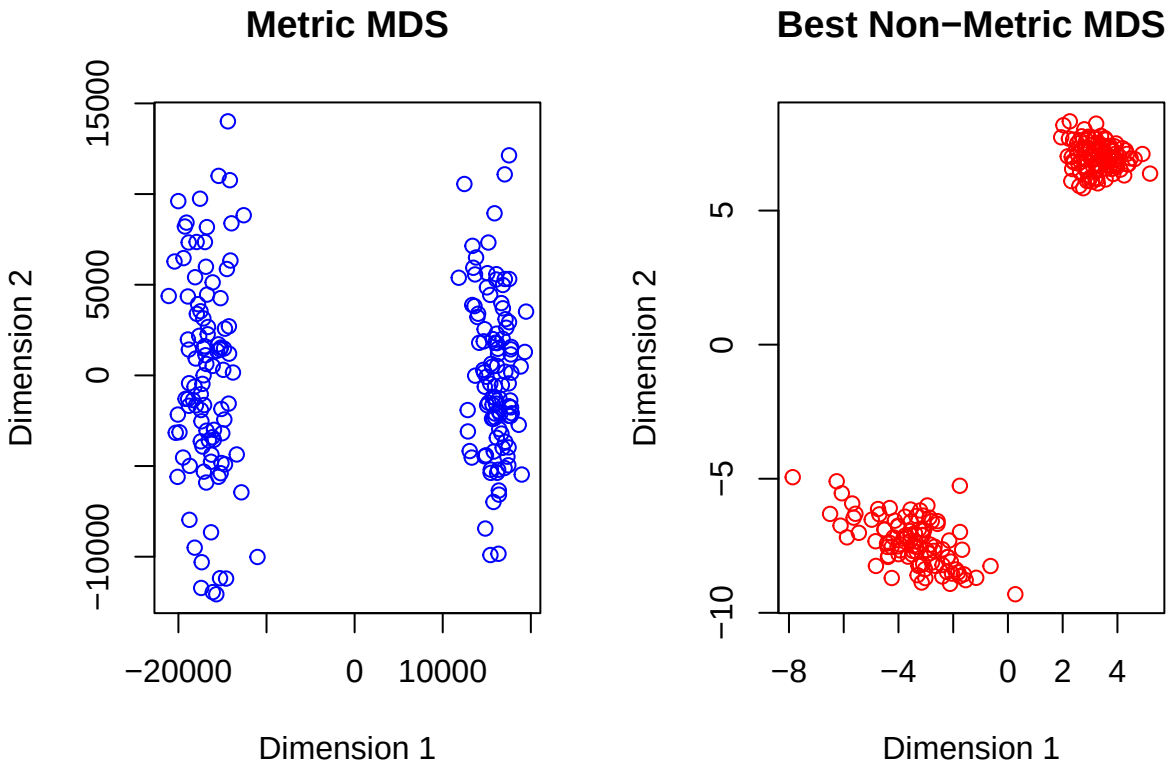
Question 11 :

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.

```
correlation_matrix <- cor(mds.res$points, best_run)
print(correlation_matrix)
```

```
##             [,1]       [,2]
## [1,]  0.95472789 0.99566188
## [2,] -0.06874971 0.02506743
```

```
par(mfrow = c(1, 2))

plot(mds.res$points[, 1], mds.res$points[, 2], col = "blue",
     main = "Metric MDS", xlab = "Dimension 1", ylab = "Dimension 2")

plot(best_run[, 1], best_run[, 2], col = "red",
     main = "Best Non-Metric MDS",
     xlab = "Dimension 1", ylab = "Dimension 2")
```



As mentioned earlier, we observe a correlation between dimensions in Metric MDS but not in Non-Metric MDS.

The difference in the dimensionality of clusters between Metric and Non-Metric MDS may be attributed to factors beyond the choice of the distance metric, especially considering that the same Manhattan metric was utilized in both cases. It's possible that other aspects of the algorithms, such as the underlying optimization procedures or the treatment of dissimilarities, contribute to the observed distinctions.

In this context, while Metric MDS relies on a distance metric to preserve original dissimilarities and might result in a one-dimensional separation in certain cases, Non-Metric MDS prioritizes maintaining the order of dissimilarities over their exact magnitudes, potentially leading to a more distributed separation across multiple dimensions. Further investigation into the specific intricacies of the algorithms could shed light on the observed differences in cluster dimensionality.