BSG - Homework 5 - Population substructure

HENRY QIU LO & MEYSAM ZAMANI

December 15, 2019

1. The file Chr21.dat contains genotype information of a set of individuals of unknown background. Load this data into the R environment with the fread instruction. The first six columns of the data matrix contain identifiers, sex and phenotype and are not needed. The remaining columns contain the allele counts (0, 1 or 2) for over 138.000 SNPs for one of the alleles of each SNP.

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

```
data <- fread(file="Chr21.dat")
data$SEX <- NULL
data$PHENOTYPE <- NULL
nrows <- nrow(data)
ncols <- ncol(data)
sum(is.na(data))/(nrows*ncols)*100
```

```
## [1] 0
```

```
nrows
```

```
## [1] 203
```

```
ncols
```

```
## [1] 138110
```

2. Compute the Manhattan distance matrix between the individuals (this may take a few minutes) using R function dist. Include a submatrix of dimension 5 by 5 with the distances between the first 5 individuals in your report.

```
manhattanDist <- as.matrix(dist(data[,5:ncols]))
manhattanDist[1:5,1:5]
```

```
##          1        2        3        4        5
## 1   0.0000 255.2234 259.5400 270.5993 256.3201
## 2 255.2234   0.0000 258.1163 262.9239 262.3643
## 3 259.5400 258.1163   0.0000 261.5473 260.7623
## 4 270.5993 262.9239 261.5473   0.0000 274.8418
## 5 256.3201 262.3643 260.7623 274.8418   0.0000
```

This is the submatrix of distances between the frst 5 individuals

3. The Manhattan distance (also known as the taxicab metric) is identical to the Minkowsky distance with parameter Landa = 1. How does the Manhattan distance relate to the allele sharing distance, where the latter is calculated as two minus the number of shared alleles?

Division of the Manhattan distance by the number of polymorphismsgives the allele-sharing distance

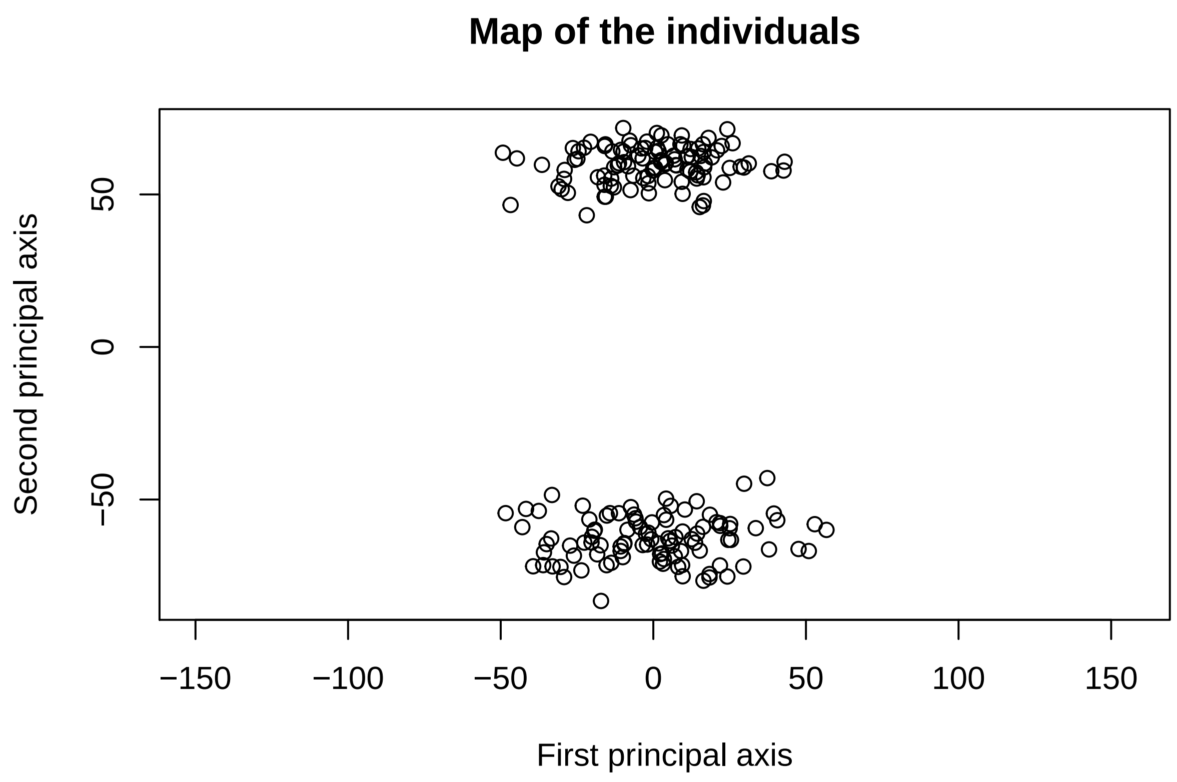
4. Apply metric multidimensional scaling using the Manhattan distance matrix to obtain a map of the individuals, and include your map in your report. Do you think the data come from one homogeneous human population? If not, how many subpopulations do you think the data might come from, and how many individuals pertain to each suppopulation?

Multidimensional scaling:

```r
muliDimScaling <- cmdscale(manhattanDist,k=nrows-1,eig=TRUE)
```

Map of the individuals:

```r
X <- muliDimScaling$points
plot(X[,2],X[,1], xlab="First principal axis", ylab="Second principal axis", main="Map of the individual
```



From the map we can observe the data seems to come from two different subpopulations, there are individuals concentrating in two different poles in the plot.

```r
positives <- sum(X[,1]>0)
negatives <- sum(X[,1]<0)
positives
```

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

```r
negatives
```

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

As they are very differentiated by the y axis, we can separate them by sign of their value in dimension one, we can see 104 individuals pertain one population and 99 individuals pertain to another.

5. Report the first 10 eigenvalues of the solution.

The first 10 eigen values are the following:

```r
eigenValues <- muliDimScaling$eig
head(eigenValues,10)
```

```
##  [1] 772590.24  90338.34  81999.06  78287.09  77424.73  73191.08  72052.16
## [8] 70292.23 68918.76 68059.62
```

6. Does a perfect representation of this n x n distance matrix exist, in n or fewer dimensions? Why so or not?

No, it cannot exist, because while we are reducing the dimensions we are losing more information, however if we increase so much the dimension to have more information we will be overfitting the model. This is why it is so difficult to obtain a perfect representation of that distance matrix, and this is why we can only do approximations to that distance matrix.

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

Two-dimensional approximation

```r
approx2d <- cmdscale(manhattanDist,k=2,eig=TRUE)
goodnessOfFit <- approx2d$GOF
```

The goodness of fit is given by the cmdscale method reults. The criterium used to check this is the quotient between the sum of the eigen values of two dimensions divided to the sum of eigen values of n-1 dimensions.

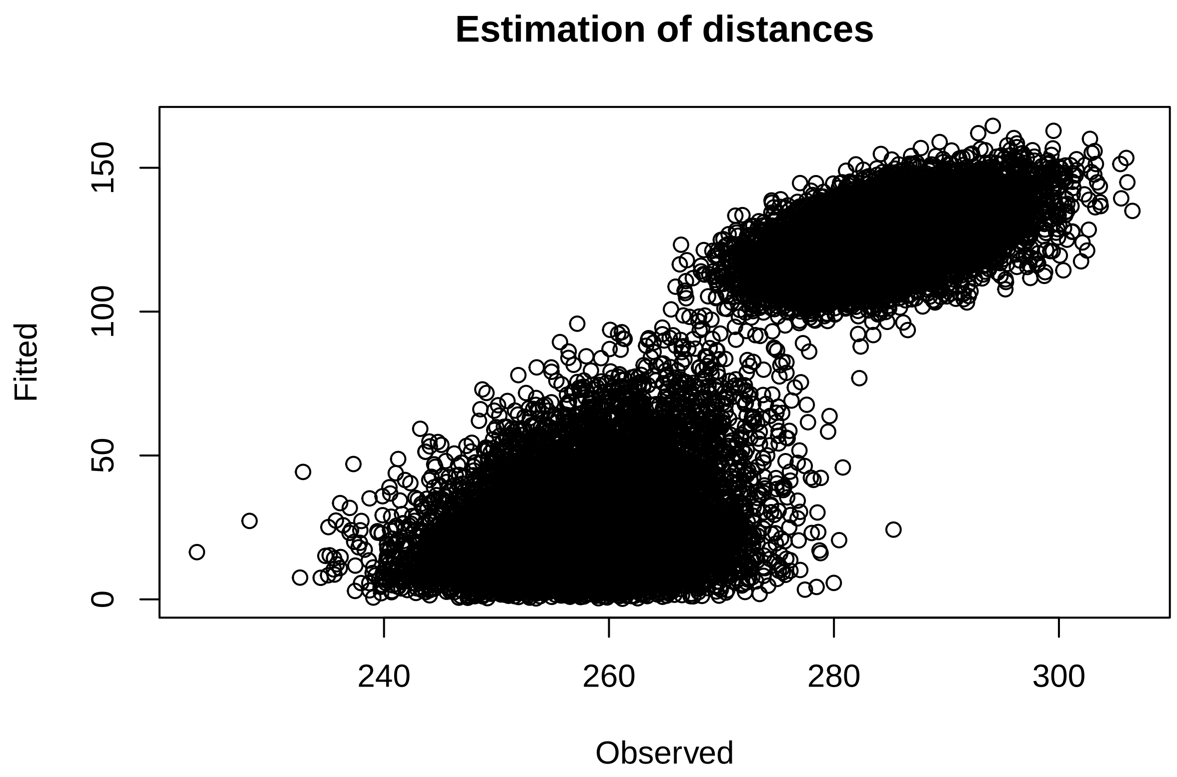
```r
goodnessOfFit
```

```
## [1] 0.1154384 0.1154384
```

We can see the GOF only has a value of 0.1154384, which is a very bad value, this is understandable because we have a lot of individuals and we are only considering 2 dimensions, the proportion of significant information explained is very small.

8. Make a plot of the estimated distances (according to your 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.

```r
Observed <- manhattanDist[lower.tri(manhattanDist)]
X <- approx2d$points
XDist <- as.matrix(dist(X))
Fitted <- XDist[lower.tri(XDist)]
plot(Observed,Fitted,main="Estimation of distances",xlab="Observed",ylab="Fitted")
```



```
#abline(lm(Observed ~ Fitted),col="blue")
```

What we can observe is the estimated distances are not fitting very well the observed distances. Which coincide with our results before. We can see on the plot two big “balls” of points instead of a slim line that would indicates a good fitting.

```
coefOfDetermination <- cor(Observed,Fitted)
coefOfDetermination
```

```
## [1] 0.9096407
```

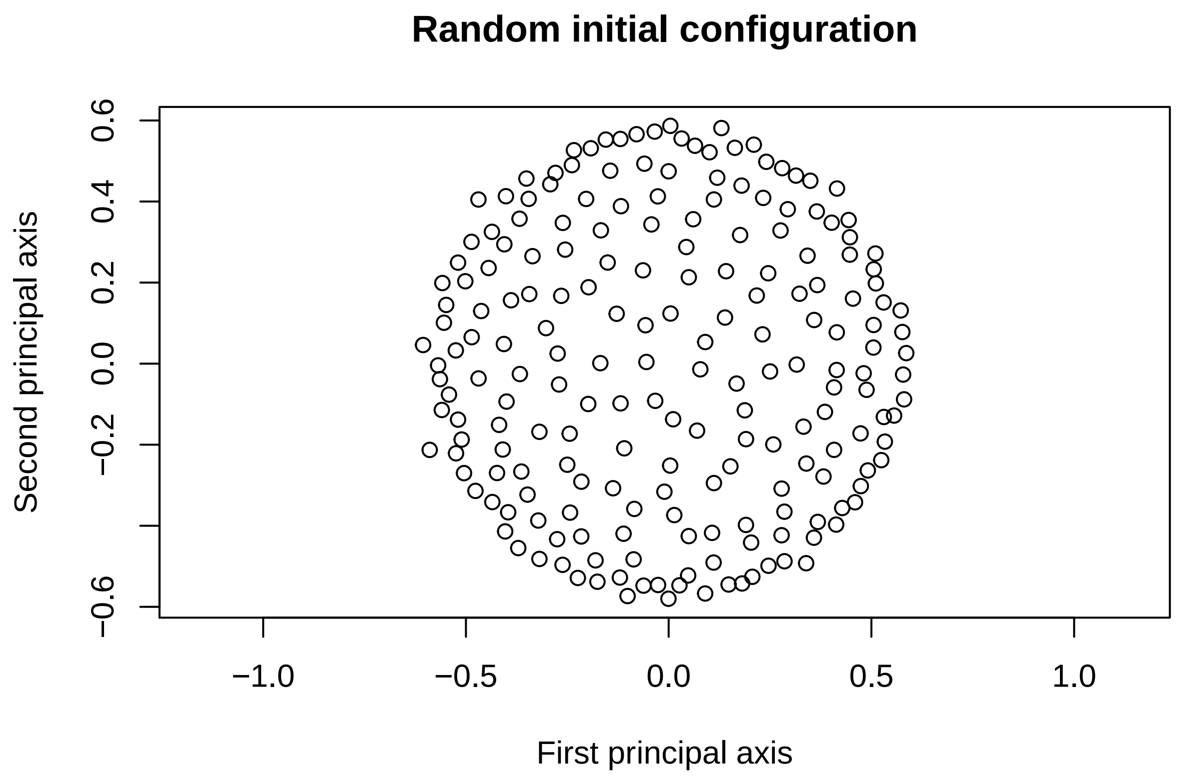
The coefficient of determination of the regression is 0.9096407.

9. We now try non-metric multidimensional scaling using the isoMDs instruction. We use a random initial con

guration. For the sake of reproducibility, make this random initial configuration with the instructions:

```
library(MASS)
```

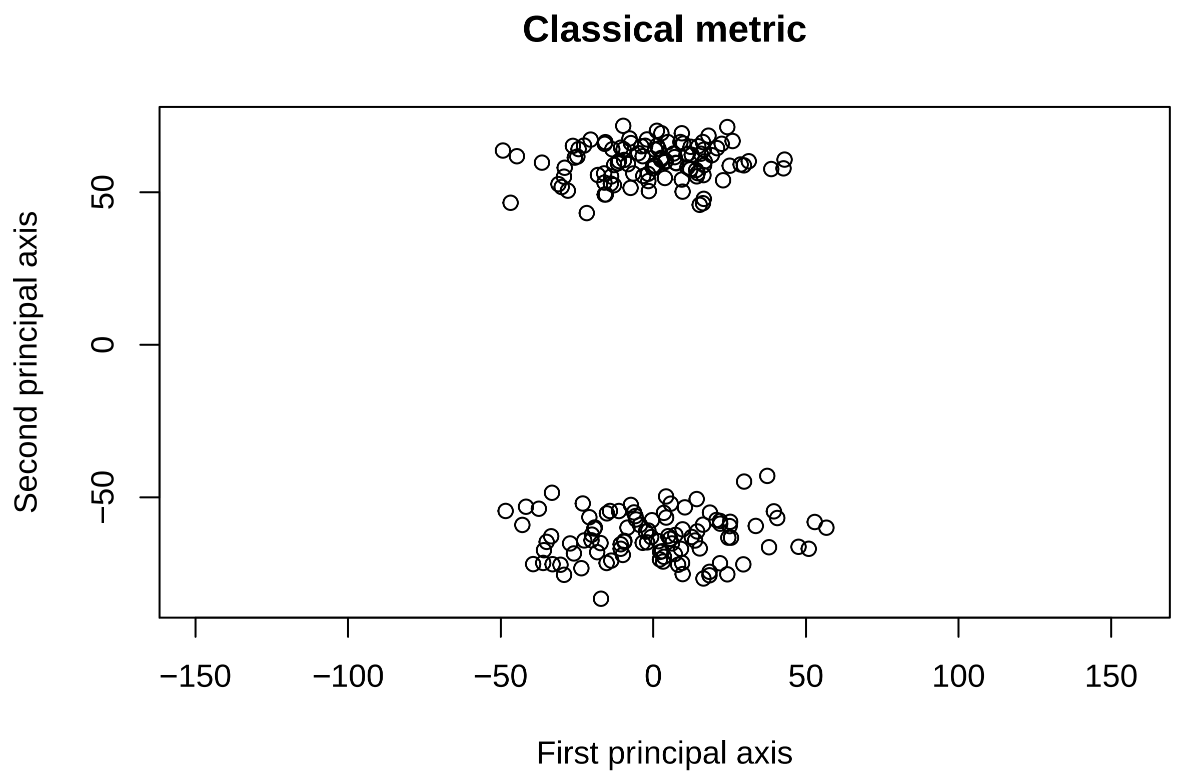
```
set.seed(12345)
init <- scale(matrix(runif(2*nrows),ncol=2),scale=FALSE)
sample <- isoMDS(manhattanDist,init,trace=FALSE)
plot(sample$points[,2],sample$points[,1], xlab="First principal axis", ylab="Second principal axis", ma
```



It seems that the distance between the individuals are simmilar, only observing the behaviour of the plot we can say it seems an homogeneous population. It is normal that data generated randomly have simmilar distances between them.

Using the classical metric without adding an initial configuration.

```
sample <- isoMDS(manhattanDist,trace=FALSE)
plot(sample$points[,2],sample$points[,1], xlab="First principal axis", ylab="Second principal axis", ma
```



It seems that the reulst is the same we saw before where individuals are separated in two different poles, which is natural because the classical solution use cmdscale with k = 2 to compute the initial configuration, which is the same we did before.

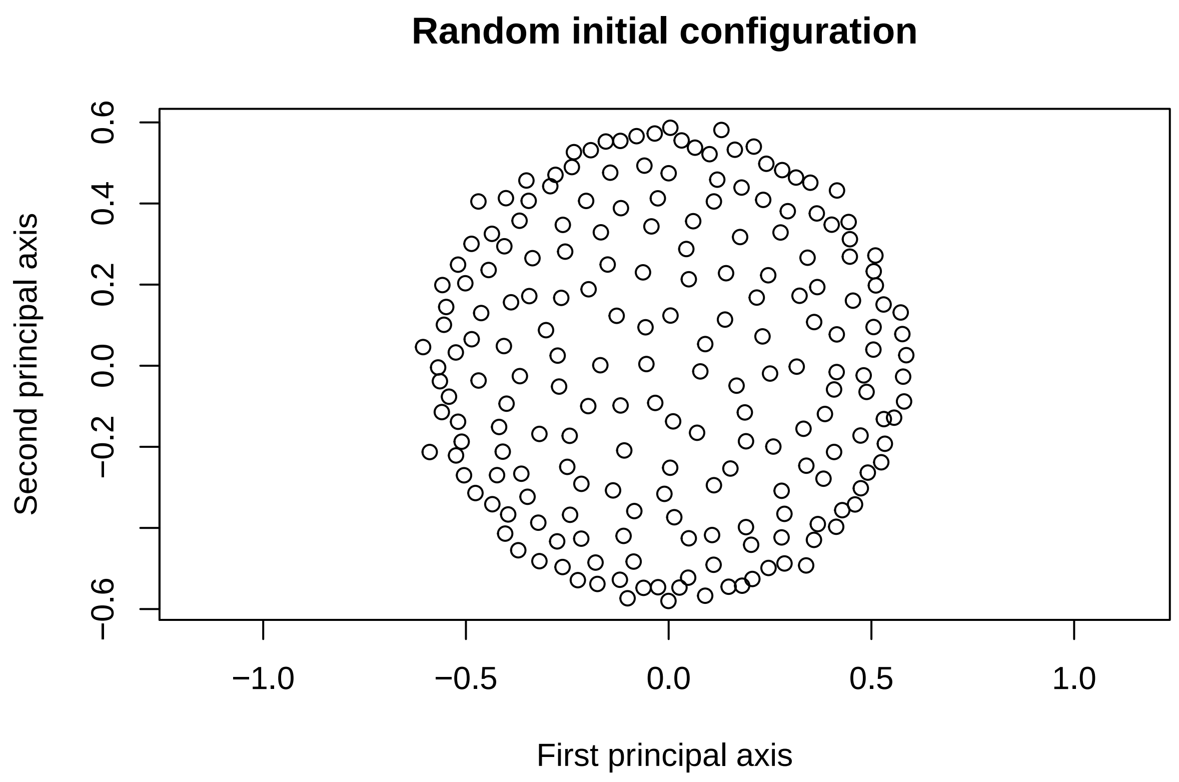
10. Set the seed of the random number generator to 123. Then run isoMDS a hundred times, each time using a different random initial configuration using the instructions above. Save the final stress-value and the coordinates of each run. Report the stress of the best run, and plot the corresponding map.

```r
set.seed(123)

best_run <- function(M) {
  initConf <- scale(matrix(runif(2*nrows),ncol=2),scale=FALSE)
  nmdsOut <- isoMDS(M,init,k=2,trace=FALSE)
  bestNmdsOut <- nmdsOut
  for (i in 2:100) {
    initConf <- scale(matrix(runif(2*nrows),ncol=2),scale=FALSE)
    nmdsOut <- isoMDS(M,init,k=2,trace=FALSE)
    if (nmdsOut$stress < bestNmdsOut$stress) {
      bestNmdsOut <- nmdsOut
    }
  }
  return(nmdsOut)
}

bestNmdsOut <- best_run(manhattanDist)
bestNmdsOut$stress
```

```
## [1] 41.65025
```

The best stress has value of 41.65025 which is really high.

```
XBestPoints <- bestNmdsOut$points
plot(XBestPoints[,2],XBestPoints[,1], xlab="First principal axis", ylab="Second principal axis", main="
```



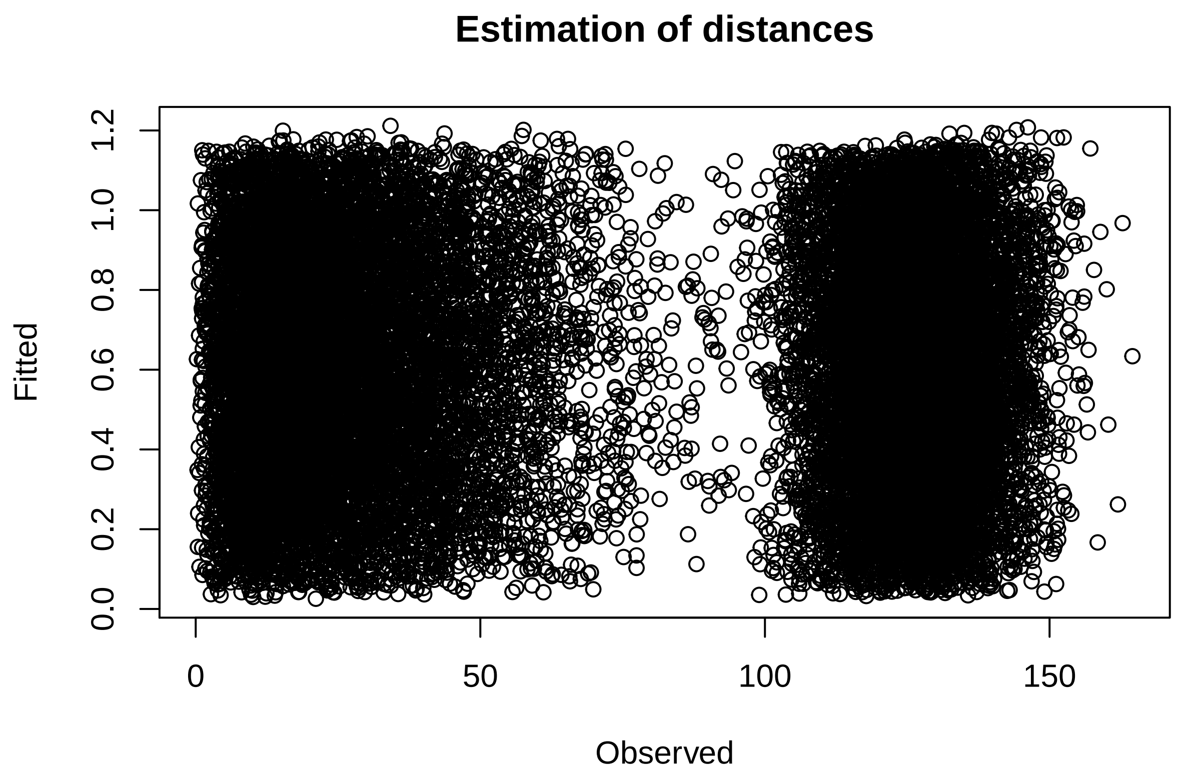
11. Make again a plot of the estimated distances (according to your map of individuals of the best run) versus the observed distances, now for the two-dimensional solution of non-metric MDS. Regress estimated distances on observed distances and report the coefficient of determination of the regression.

Two-dimensional solution of non-metric MDS

```
classicalMetric <- isoMDS(manhattanDist,k=2,trace=FALSE)
ClassicalMetricPoints <- classicalMetric$points
ClassicalMetricDist <- as.matrix(dist(ClassicalMetricPoints))
ClassicalMetricObserved <- ClassicalMetricDist[lower.tri(ClassicalMetricDist)]
```

Estimated distances of the best run

```
XBestDist <- as.matrix(dist(XBestPoints))
XBestFitted <- XBestDist[lower.tri(XBestDist)]
plot(ClassicalMetricObserved,XBestFitted,main="Estimation of distances",xlab="Observed",ylab="Fitted")
```



```
#abline(lm(ClassicalMetricObserved ~ XBestFitted),col="blue")
```

What we can observe is the estimated distances are not fitting very well the observed distances. Which coincide with our results before. We can see on the plot two big "balls" of points instead of a slim line that would indicatesa good fitting.

```
coefOfDet <- cor(ClassicalMetricObserved,XBestFitted)
coefOfDet
```

```
## [1] -0.002304815
```

The coefficient of determination of the regression is -0.002304815. Very low as we expected, because the samples generated randomly are completely different from the observed distance.

12. Compute the stress for a 1,2,3,4. . . . . . ,n-dimensional solution, always using the classical MDS solution as an initial configuration. How many dimensions are necessary to obtain a good representation with a stress below 5? Make a plot of the stress against the number of dimensions

```
dim_necessary <- function(M) {
  nmdsOut <- isoMDS(M,k=1,trace=FALSE)
  dimension <- 1
  stresses <- vector(length=nrows)
  stresses[1] = nmdsOut$stress
  dimensions <- vector(length=nrows)
  dimensions[1] = 1
  for (i in 2:ncols-1) {
    initConf <- scale(matrix(runif(2*nrows),ncol=2),scale=FALSE)
    nmdsOut <- isoMDS(M,k=i,trace=FALSE)
    if (nmdsOut$stress < 5 & dimension == 1) {
      dimension = i
    }
    stresses[i] <- nmdsOut$stress
    dimensions[i] <- i
  }
  result = list("stresses" = stresses,"dimensions" = dimensions, "dim" = dimension)
  return (result)
}
```

```
#dimNecessaryNmdsOut <- dim_necessary(manhattanDist)
#plot(dimNecessaryNmdsOut$stresses,dimNecessaryNmdsOut$dimensions,main="Stress against dimensions",xlab
```

```
#dim
```

The number of dimensions necessary to obtain a good representation with a stress below 5 is 36.

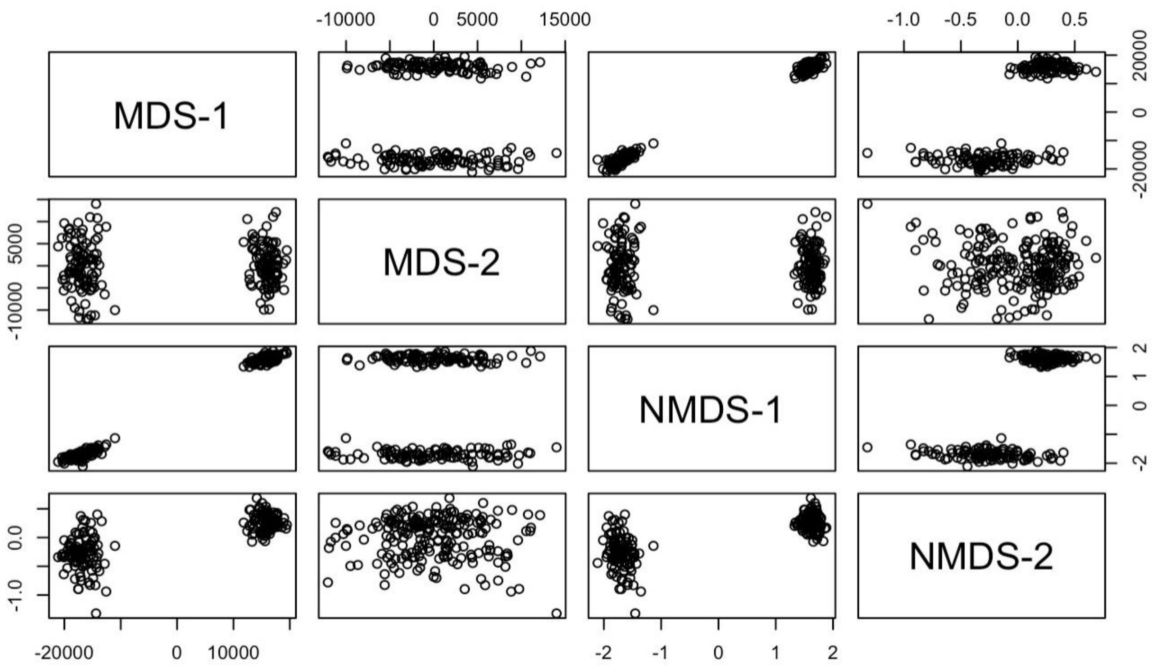
13. Compute the correlation matrix between the first two dimensions of a metric MDS and the two-dimensional solution of your best non-metric MDS. Make a scatterplot matrix of the 4 variables. Comment on your findings.

```
matrix <- data.frame(x1=Fitted,x2=XBestFitted)
corrMatrix <- cor(matrix)
```

```
      MDS-1   MDS-2   NMDS-1    NMDS-2
```

MDS-1 1.0000 0.0000 0.9974 0.7505 MDS-2 0.0000 1.0000 0.0115 -0.0545 NMDS-1 0.9974 0.0115 1.0000 0.7408 NMDS-2 0.7505 -0.0545 0.7408 1.0000

```
#H <- cbind(XBestPoints, corrMatrix[[1,98]])
#colnames(H) <- c("MDS-1","MDS-2","NMDS-1","NMDS-2")
#pairs(H)
```


MDS-1
MDS-2
NMDS-1
NMDS-2