BSG-MDS practical 4 Statistical Genetics

Eliya Tiram and Ximena Moure

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

Load data

```
data <- fread("Chr21.dat")
genetic_data <- data[, -c(1:6), with=FALSE]
```

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

```
num_variants <- ncol(genetic_data)
missing_data_percentage <- 100 * sum(is.na(genetic_data)) /
  (nrow(genetic_data) * ncol(genetic_data))

cat("\nNum variants:",num_variants, "\n")
```

```
##
## Num variants: 138106
```

```
cat("\nPercentage missing:",missing_data_percentage, "\n")
```

```
##
## Percentage missing: 0
```

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

```
manhattan_distances <- as.matrix(dist(genetic_data, method = "manhattan"))
submatrix_5x5 <- manhattan_distances[1:5, 1:5]

submatrix_5x5
```

```
##       1     2     3     4     5
## 1     0 53495 55007 58174 53794
## 2 53495     0 55372 55995 55699
## 3 55007 55372     0 54815 55683
## 4 58174 55995 54815     0 59046
## 5 53794 55699 55683 59046     0
```

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

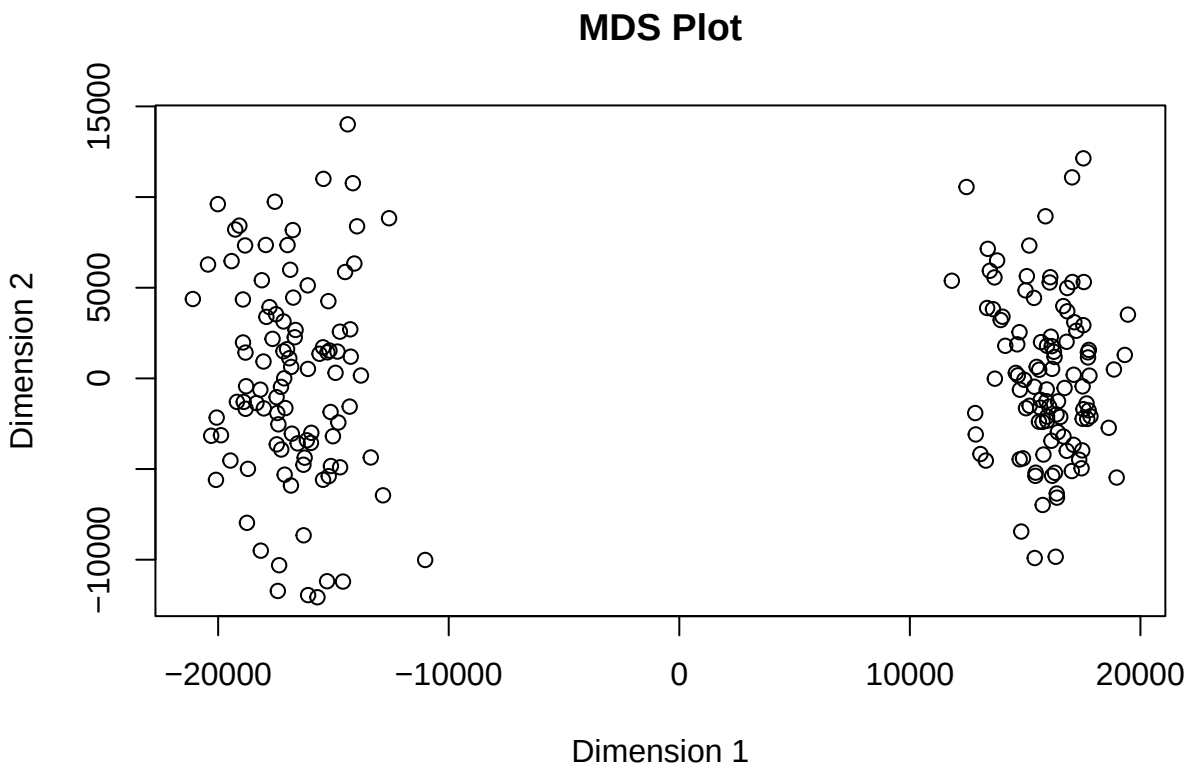
The Manhattan distance and the allele sharing distance are inversely related. If two individuals share more alleles across their genotypes (high allele sharing distance), they will have a smaller Manhattan distance because there will be fewer differences in their (0,1,2) genotype encodings. Conversely, if two individuals share fewer alleles (low allele sharing distance), the Manhattan distance will be larger due to more differences in their genotype encodings.

While Manhattan distance is a measure of dissimilarity, allele sharing distance is a measure of similarity.

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?

In the plot, we see two distinct clusters of points. This suggests that the individuals in this dataset likely come from at least two different subpopulations. The clusters are quite distinct and do not overlap, indicating that the genetic variation between the groups is substantial.

```
mds <- cmdscale(manhattan_distances, k = 2, eig = T)
mds_scaled <- as.data.frame(mds$points)
# Plot the map
plot(mds_scaled$V1,mds_scaled$V2, main = "MDS Plot",
     xlab = "Dimension 1", ylab = "Dimension 2")
```




```
threshold <- 0
num_subpop1 <- sum(mds$points[,1] < threshold)
num_subpop2 <- sum(mds$points[,1] >= threshold)
num_subpop1
```

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

```
num_subpop2
```

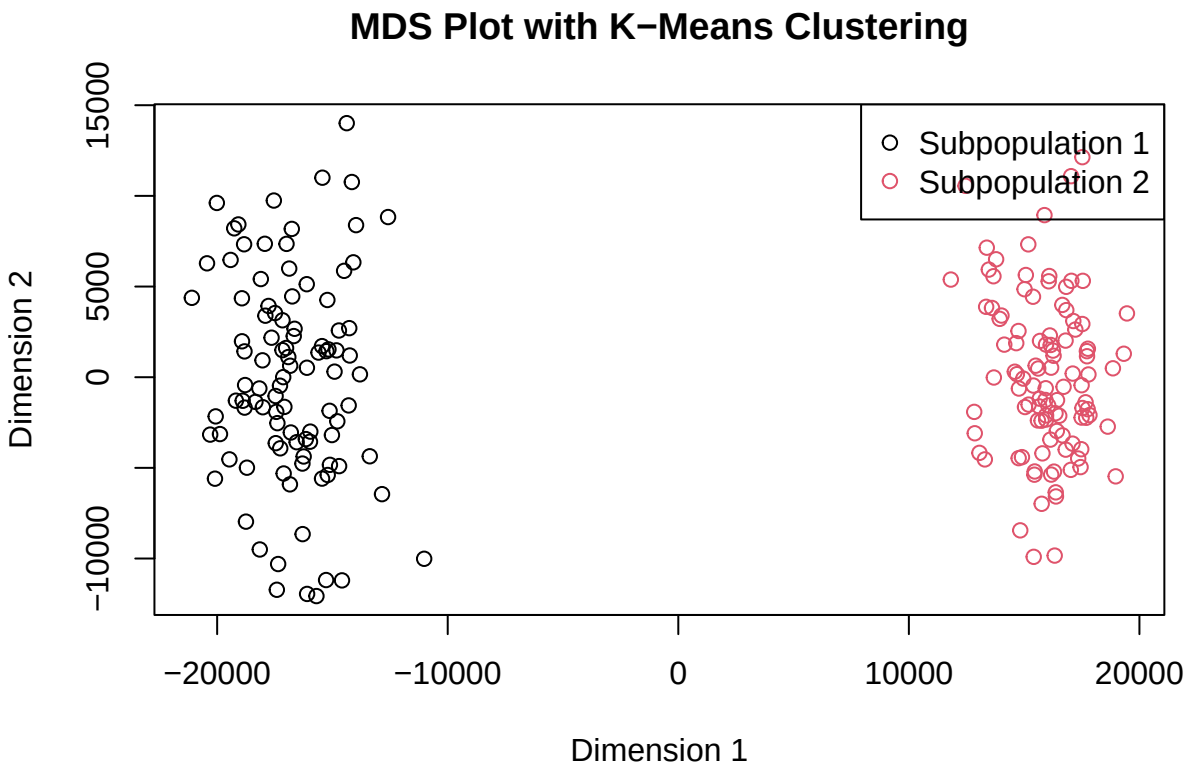
```
## [1] 104
```

We could also use kmeans to calculate the number of individuals in each subpopulation.

```
set.seed(42)
kmeans_result <- kmeans(mds$points, centers = 2)

# Count the number of individuals in each cluster
table(kmeans_result$cluster)
```

```
##
##   1   2
##  99 104
```

```r
plot(mds$points[, 1], mds$points[, 2], col = kmeans_result$cluster,
     xlab = "Dimension 1", ylab = "Dimension 2",
     main = "MDS Plot with K-Means Clustering")
legend("topright", legend = c("Subpopulation 1", "Subpopulation 2"),
       col = 1:2, pch = 1)
```



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

The goodness-of-fit measure (GOF) that we obtain from the mds$GOF is based on the proportion of variance accounted for by the chosen number of dimensions.

First Value (0.1703581): This represents the variance accounted for by the first dimension alone. It suggests that the first dimension captures about 17.03% of the total variance. Second Value (0.1703605): This represents the cumulative variance accounted for by both the first and second dimensions. Since the increase from the first to the second value is very small, it implies that the second dimension adds very little additional information beyond what is captured by the first dimension.

A cumulative goodness-of-fit of 17.04% is relatively low. This means that the two-dimensional solution does not capture a large portion of the variance in the original high-dimensional distance data.

```r
gof <-mds$GOF
cat("\nGOF:",gof, "\n")
```

```
##
## GOF: 0.1703581 0.1703605
```

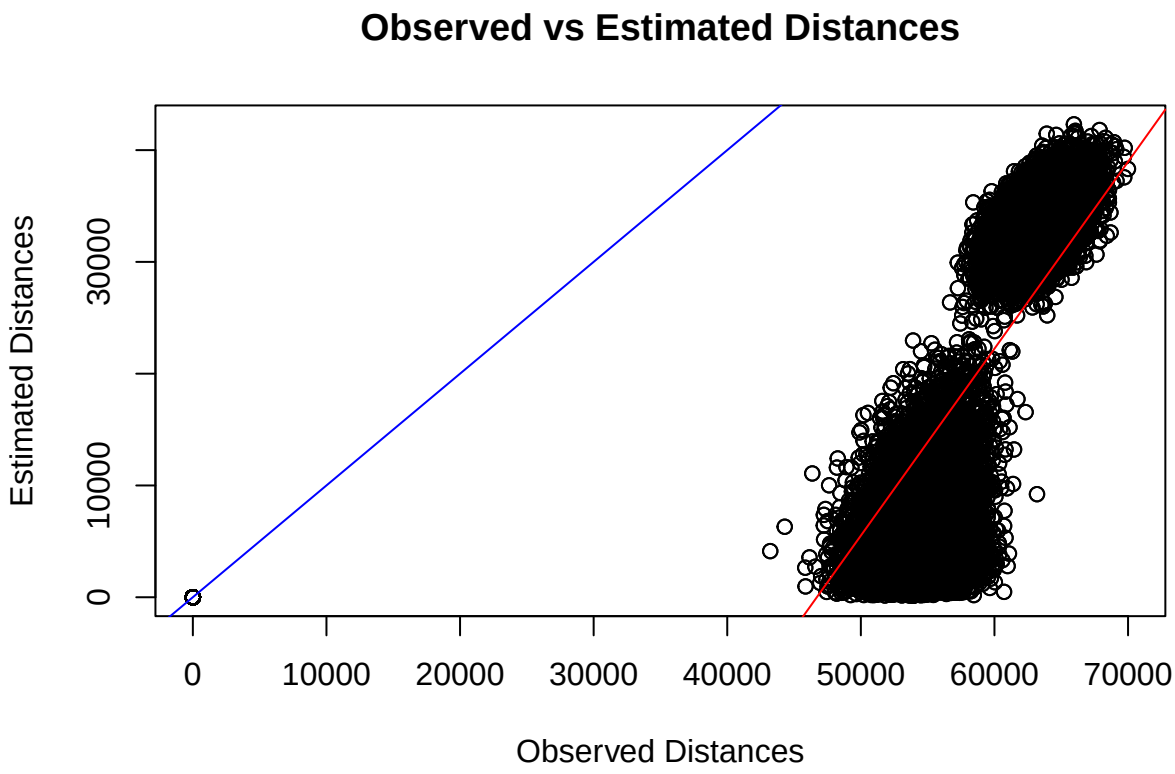
6. Make a plot of the estimated distances (according to your two-dimensional map of individuals) versus the observed distances. What do you observe? Regress estimated distances on observed distances and report the coefficient of determination of the regression (you can use the function lm).

The blue line indicates where the points would lie if the estimated distances perfectly matched the observed distances. It's clear that not all points lie on this line, suggesting that the MDS has not perfectly preserved the original distances.

There's a dense clustering of points towards the lower end of the observed distances, which spreads out as the observed distances increase. This indicates that the MDS might be compressing the distances, leading to a loss of information especially for larger distances.

An R-squared value of 0.583 suggests a moderate level of fit. It means that while there is a significant relationship between the observed and estimated distances, the two-dimensional MDS solution does not capture all the variability in the observed distances.

```
mds_distances <- as.matrix(dist(mds$points))

obs_dist <- as.vector(manhattan_distances)
est_dist <- as.vector(mds_distances)


plot(obs_dist, est_dist, xlab = "Observed Distances",
     ylab = "Estimated Distances",
     main = "Observed vs Estimated Distances")
abline(0, 1, col = "blue", lty = 1)

reg <- lm(est_dist ~ obs_dist)
abline(reg, col = "red")
```




```r
summary(reg)
```

```
##
## Call:
## lm(formula = est_dist ~ obs_dist)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -22953  -6609   1471   5872  78102
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.810e+04  4.106e+02  -190.2   <2e-16 ***
## obs_dist     1.672e+00  6.967e-03   240.0   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9064 on 41207 degrees of freedom
## Multiple R-squared:  0.583,  Adjusted R-squared:  0.583
## F-statistic: 5.761e+04 on 1 and 41207 DF,  p-value: < 2.2e-16
```

```r
r_squared <- summary(reg)$r.squared
print(paste("Coefficient of Determination (R-squared):", r_squared))
```

```
## [1] "Coefficient of Determination (R-squared): 0.582996481647993"
```

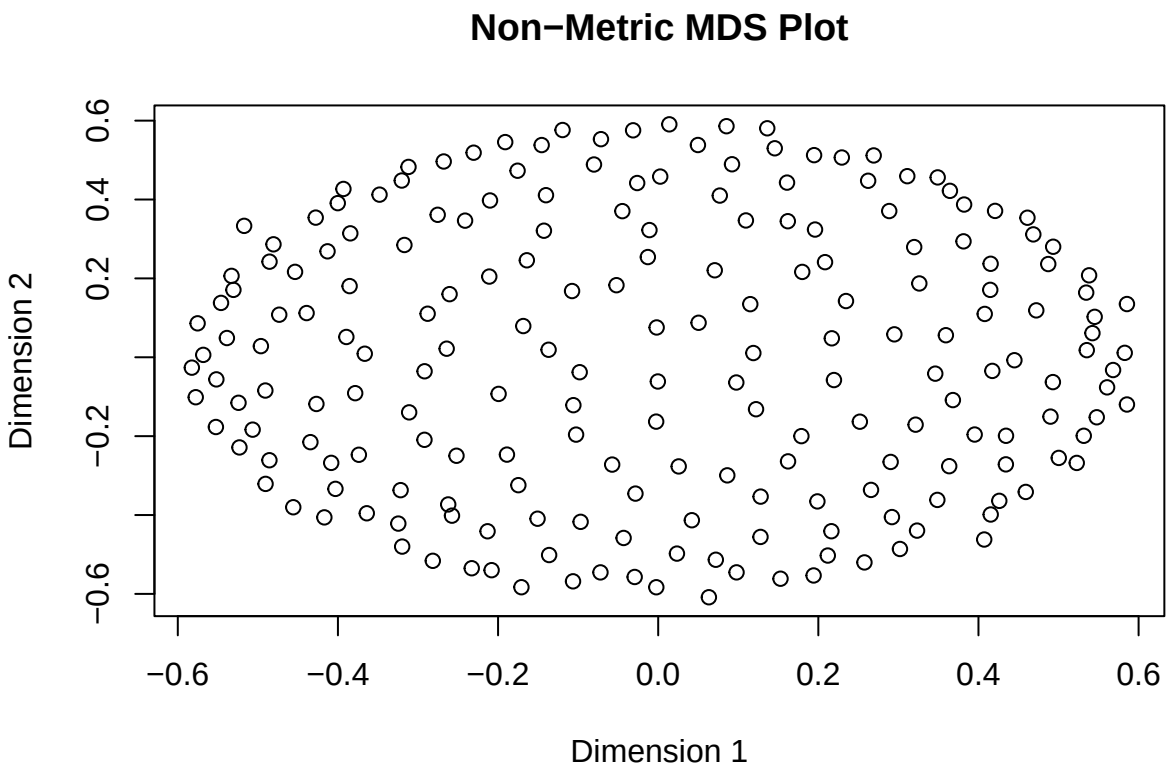
7. We now try a (two-dimensional) non-metric multidimensional scaling using the isoMDs function that you will find in MASS library. We use a random initial configuration and, for the sake of reproducibility, make this random initial configuration with the instructions: set.seed(12345) and init <- scale(matrix(runif(m*n),ncol=m),scale=FALSE) 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?

There are no obvious subgroups indicating separate populations, this can suggest that the individuals in the dataset might come from a single homogeneous population.

```r
set.seed(12345)
n <- nrow(genetic_data)
m <- 2
init <- scale(matrix(runif(m * n), ncol = m), scale = FALSE)
nonmetric_mds <- isoMDS(manhattan_distances, k = 2, 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
```

```r
# Plot the non-metric MDS map
plot(nonmetric_mds$points, main = "Non-Metric MDS Plot", xlab = "Dimension 1",
     ylab = "Dimension 2")
```



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?

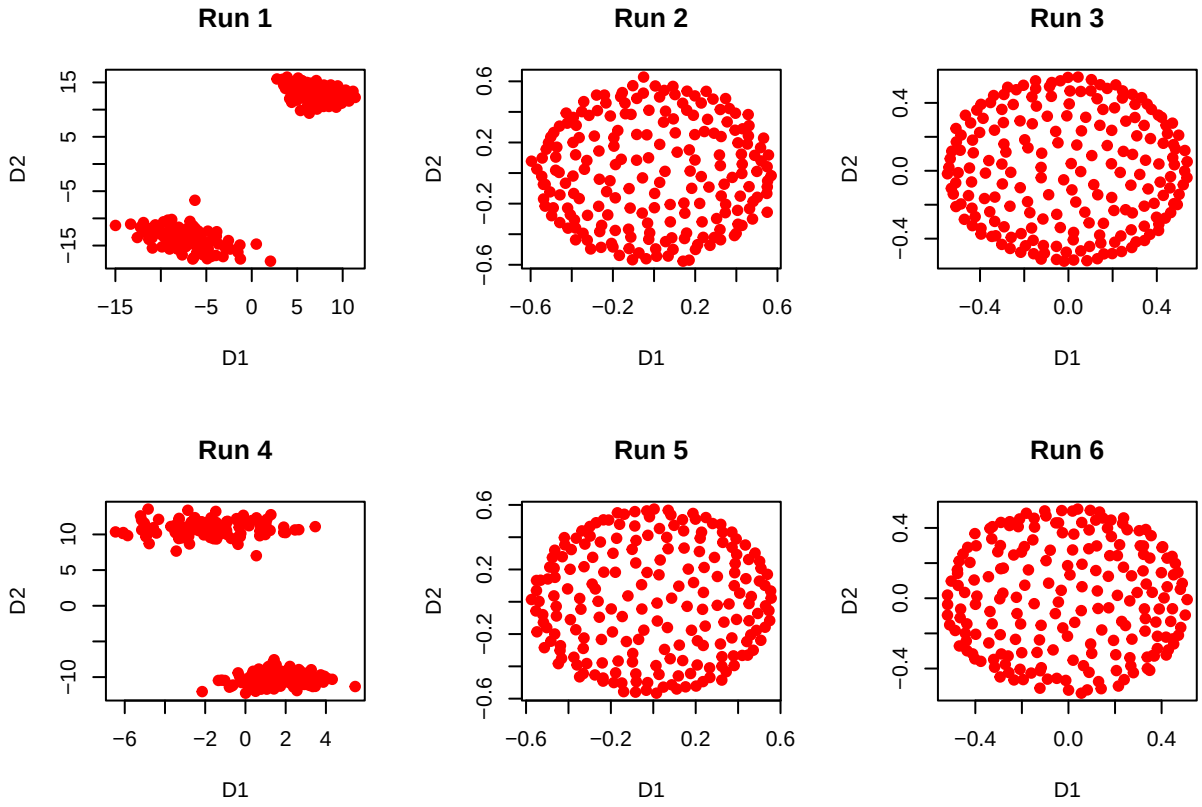
The variation in stress values implies that the initial configuration has a significant impact on the resulting MDS solution. The results suggest that the optimization process in isoMDS might be getting stuck in local minima, leading to suboptimal solutions in some runs.

From the plots we can observe that some of them show 1 population and others 2.

```
times <- 8
par(mfrow = c(2, 3))
stress_vals <- numeric(times) # To store stress values
mds_run_results <- list() # To store MDS results for each run

set.seed(123)
for (i in 1:times) {
  # Random initial configuration
  init <- scale(matrix(runif(2 * nrow(manhattan_distances)), ncol = 2),
                scale = FALSE)
  mds_run_results[[i]] <- isoMDS(manhattan_distances, k = 2, y = init)$points
  plot(mds_run_results[[i]], main = paste("Run", i), xlab = "D1", ylab = "D2",
       pch = 19, col = "red")
  stress_vals[i] <- isoMDS(manhattan_distances, k = 2, y = init)$stress
}
```

```
## initial  value 42.831731
## iter   5 value 41.451139
## iter  10 value 39.514546
## iter  15 value 38.318724
## iter  20 value 28.166914
## iter  25 value 19.664806
## iter  30 value 17.001873
## iter  35 value 14.418368
## iter  40 value 13.254431
## iter  45 value 12.600721
## iter  50 value 12.250950
## final  value 12.250950
## stopped after 50 iterations

## initial  value 42.831731
## iter   5 value 41.451139
## iter  10 value 39.514546
## iter  15 value 38.318724
## iter  20 value 28.166914
## iter  25 value 19.664806
## iter  30 value 17.001873
## iter  35 value 14.418368
## iter  40 value 13.254431
## iter  45 value 12.600721
## iter  50 value 12.250950
## final  value 12.250950
## stopped after 50 iterations
## initial  value 42.815019
## final  value 41.626067
## converged

## initial  value 42.815019
## final  value 41.626067
## converged
## initial  value 42.869420
## final  value 41.679113
## converged

## initial  value 42.869420
## final  value 41.679113
## converged
## initial  value 42.732521
## iter   5 value 41.653260
## iter  10 value 40.565860
## iter  15 value 34.646168
## iter  20 value 24.128095
## iter  25 value 17.671389
## iter  30 value 15.144808
## iter  35 value 13.439508
## iter  40 value 12.900325
## iter  45 value 12.522467
## final  value 12.429894
## converged
```

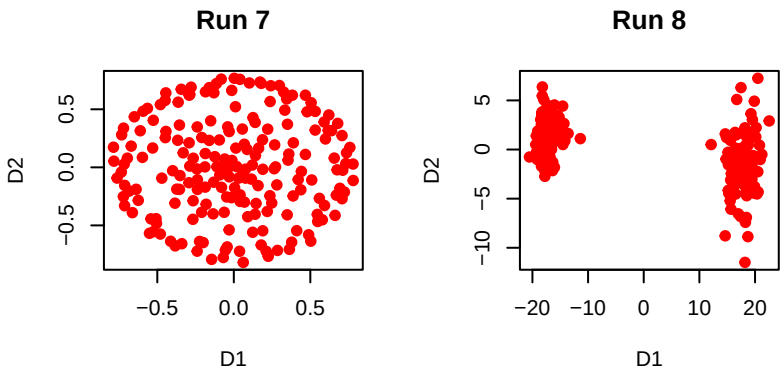
```
## initial  value 42.732521
## iter   5 value 41.653260
## iter  10 value 40.565860
## iter  15 value 34.646168
## iter  20 value 24.128095
## iter  25 value 17.671389
## iter  30 value 15.144808
## iter  35 value 13.439508
## iter  40 value 12.900325
## iter  45 value 12.522467
## final  value 12.429894
## converged
## initial  value 43.064402
## final  value 41.661111
## converged

## initial  value 43.064402
## final  value 41.661111
## converged
## initial  value 43.205674
## final  value 41.683537
## converged
```




```
## initial  value 43.205674
## final  value 41.683537
```

```
## converged
## initial  value 42.921608
## iter   5 value 41.443987
## iter  10 value 39.700992
## final  value 39.568095
## converged

## initial  value 42.921608
## iter   5 value 41.443987
## iter  10 value 39.700992
## final  value 39.568095
## converged
## initial  value 42.967078
## iter   5 value 41.537271
## iter  10 value 39.622920
## iter  15 value 37.183724
## iter  20 value 28.122611
## iter  25 value 22.466483
## iter  30 value 16.483062
## iter  35 value 13.874060
## iter  40 value 13.107436
## iter  45 value 12.319836
## iter  50 value 12.079692
## final  value 12.079692
## stopped after 50 iterations

## initial  value 42.967078
## iter   5 value 41.537271
## iter  10 value 39.622920
## iter  15 value 37.183724
## iter  20 value 28.122611
## iter  25 value 22.466483
## iter  30 value 16.483062
## iter  35 value 13.874060
## iter  40 value 13.107436
## iter  45 value 12.319836
## iter  50 value 12.079692
## final  value 12.079692
## stopped after 50 iterations
```

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




```
cat("Stress values: ", stress_vals, "\n")
```

```
## Stress values:  12.25095 41.62607 41.67911 12.42989 41.66111 41.68354 39.56809 12.07969
```

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.

```
stress_values <- numeric(50)  # to store stress values for each dimension
dimensions <- 1:50
cmdscale_result <- cmdscale(manhattan_distances, eig = TRUE,
                            k = max(dimensions))

for (k in dimensions) {
   init <- cmdscale_result$points[, 1:k, drop = FALSE]
  # Run non-metric MDS with varying dimensions
  mds_result <- isoMDS(manhattan_distances, k = k, y=init)
  # Store the stress value
  stress_values[k] <- mds_result$stress
}
```

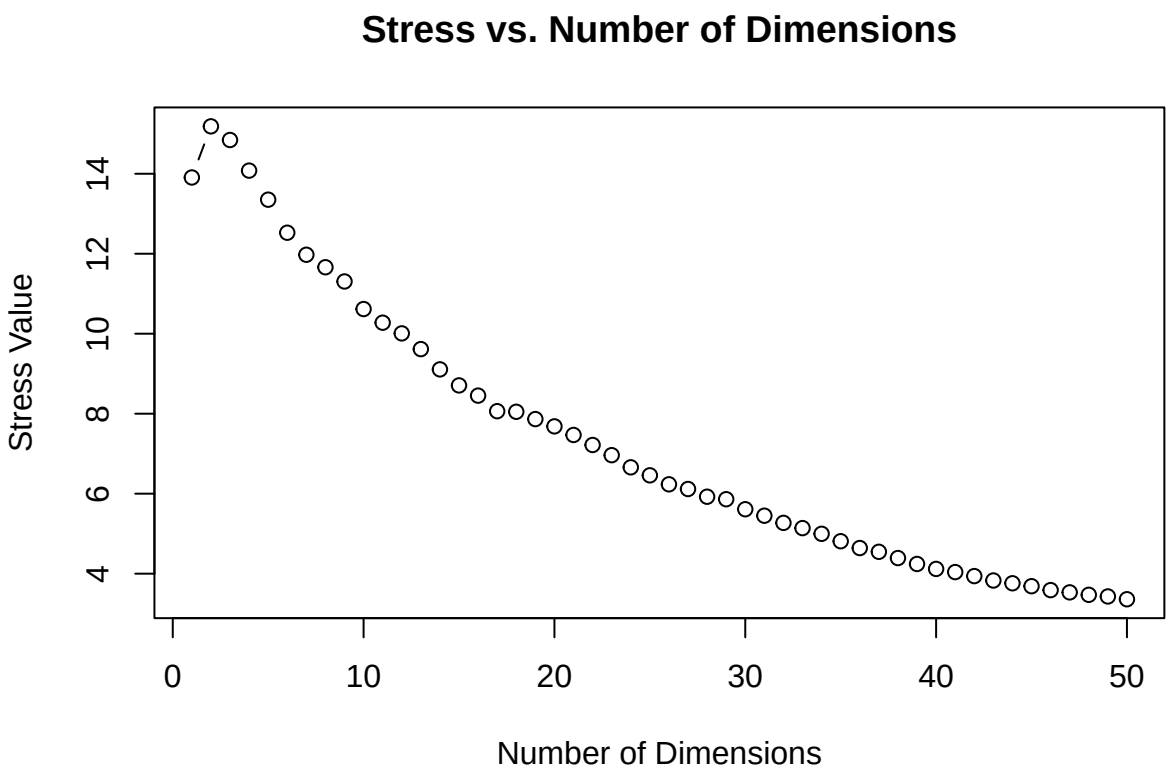
```
## initial  value 13.906634
## final  value 13.906634
```

```
## converged
## initial  value 15.185076
## final  value 15.185076
## converged
## initial  value 14.843856
## final  value 14.843856
## converged
## initial  value 14.078522
## final  value 14.078522
## converged
## initial  value 13.351683
## final  value 13.351683
## converged
## initial  value 12.525247
## final  value 12.525247
## converged
## initial  value 11.974448
## final  value 11.974448
## converged
## initial  value 11.662172
## final  value 11.662172
## converged
## initial  value 11.306433
## final  value 11.306433
## converged
## initial  value 10.617137
## final  value 10.617137
## converged
## initial  value 10.275785
## final  value 10.275785
## converged
## initial  value 10.008590
## final  value 10.008590
## converged
## initial  value 9.614630
## final  value 9.614630
## converged
## initial  value 9.108197
## final  value 9.108197
## converged
## initial  value 8.707142
## final  value 8.707142
## converged
## initial  value 8.451913
## final  value 8.451913
## converged
## initial  value 8.062375
## final  value 8.062375
## converged
## initial  value 8.047934
## final  value 8.047934
## converged
## initial  value 7.864988
## final  value 7.864988
```

```
## converged
## initial  value 7.683222
## final  value 7.683222
## converged
## initial  value 7.468312
## final  value 7.468312
## converged
## initial  value 7.218013
## final  value 7.218013
## converged
## initial  value 6.963368
## final  value 6.963368
## converged
## initial  value 6.659217
## final  value 6.659217
## converged
## initial  value 6.458941
## final  value 6.458941
## converged
## initial  value 6.233994
## final  value 6.233994
## converged
## initial  value 6.117082
## final  value 6.117082
## converged
## initial  value 5.924375
## final  value 5.924375
## converged
## initial  value 5.861823
## final  value 5.861823
## converged
## initial  value 5.610832
## final  value 5.610832
## converged
## initial  value 5.449734
## final  value 5.449734
## converged
## initial  value 5.269790
## final  value 5.269790
## converged
## initial  value 5.139134
## final  value 5.139134
## converged
## initial  value 4.996730
## final  value 4.996729
## converged
## initial  value 4.811151
## final  value 4.811151
## converged
## initial  value 4.640897
## final  value 4.640897
## converged
## initial  value 4.546486
## final  value 4.546486
```

```
## converged
## initial  value 4.390268
## final  value 4.390267
## converged
## initial  value 4.242612
## final  value 4.242612
## converged
## initial  value 4.118732
## final  value 4.118732
## converged
## initial  value 4.039682
## final  value 4.039682
## converged
## initial  value 3.939259
## final  value 3.939259
## converged
## initial  value 3.829421
## final  value 3.829421
## converged
## initial  value 3.759687
## final  value 3.759687
## converged
## initial  value 3.686426
## final  value 3.686426
## converged
## initial  value 3.588795
## final  value 3.588795
## converged
## initial  value 3.532327
## final  value 3.532327
## converged
## initial  value 3.471093
## final  value 3.471093
## converged
## initial  value 3.430112
## final  value 3.430112
## converged
## initial  value 3.360582
## final  value 3.360582
## converged
```

```r
# Find the number of dimensions needed for stress below 10
good_dims <- which(stress_values < 10)

# Plot stress values against number of dimensions
plot(dimensions, stress_values, type = "b", xlab = "Number of Dimensions",
     ylab = "Stress Value", main = "Stress vs. Number of Dimensions")
```




```
if (length(good_dims) > 0) {
  cat("Minimum dimensions needed for stress below 10:", min(good_dims), "\n")
} else {
  cat("No solution with stress below 10 found up to 50 dimensions.\n")
}
```

```
## Minimum dimensions needed for stress below 10: 13
```

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.

The plot for the best run shows a more defined structure with data points grouped together, indicating that the two-dimensional space may be capturing some meaningful relationships within the data.

The plot for the worst run shows the points dispersed in a circular pattern, which suggests that the two-dimensional representation is more or less random and does not capture the structure of the data effectively.

The substantial difference in stress values and the visual configurations imply that the initial configuration can significantly influence the quality of the isoMDS solution.

```
n_runs <- 100
n <- nrow(manhattan_distances)
stress_values <- numeric(n_runs)
```

```r
mds_configs <- list()

for (i in 1:n_runs) {
  set.seed(i)
  init <- scale(matrix(runif(2 * n), ncol = 2), scale = FALSE)
  mds_run <- isoMDS(manhattan_distances, k = 2, y = init)
  stress_values[i] <- mds_run$stress
  mds_configs[[i]] <- mds_run$points
}
```

```
## initial  value 43.476480
## iter   5 value 41.268332
## iter  10 value 37.760785
## iter  15 value 21.929601
## iter  20 value 13.589127
## iter  25 value 11.897315
## iter  30 value 11.562419
## final  value 11.458723
## converged
## initial  value 42.730109
## iter   5 value 41.644638
## iter   5 value 41.619687
## iter   5 value 41.619077
## final  value 41.619077
## converged
## initial  value 42.870609
## final  value 41.691233
## converged
## initial  value 42.586838
## iter   5 value 41.633134
## iter  10 value 40.245337
## iter  15 value 39.319827
## iter  20 value 36.290634
## iter  25 value 26.711707
## iter  30 value 19.980501
## iter  35 value 16.932226
## iter  40 value 14.886214
## iter  45 value 13.515145
## iter  50 value 12.707970
## final  value 12.707970
## stopped after 50 iterations
## initial  value 42.470579
## iter   5 value 40.756714
## iter  10 value 39.495865
## iter  15 value 38.748474
## iter  20 value 31.904209
## iter  25 value 22.085221
## iter  30 value 17.278974
## iter  35 value 14.984896
## iter  40 value 13.108781
## iter  45 value 12.362269
## iter  50 value 12.034551
## final  value 12.034551
```

```
## stopped after 50 iterations
## initial  value 43.008328
## iter   5 value 41.576829
## iter  10 value 39.820681
## iter  15 value 38.363454
## iter  20 value 30.824796
## iter  25 value 24.422433
## iter  30 value 19.044437
## iter  35 value 15.333543
## iter  40 value 14.142978
## iter  45 value 12.905919
## iter  50 value 11.980119
## final  value 11.980119
## stopped after 50 iterations
## initial  value 42.807330
## iter   5 value 41.615372
## iter  10 value 40.507798
## iter  15 value 38.613582
## iter  20 value 29.807125
## iter  25 value 21.367125
## iter  30 value 17.173046
## iter  35 value 14.966939
## iter  40 value 13.334531
## iter  45 value 12.777562
## iter  50 value 12.268977
## final  value 12.268977
## stopped after 50 iterations
## initial  value 42.472828
## final  value 41.681578
## converged
## initial  value 42.699737
## iter   5 value 41.179168
## iter  10 value 39.543362
## iter  15 value 37.545687
## iter  20 value 30.800576
## iter  25 value 23.970389
## iter  30 value 20.446982
## iter  35 value 16.063756
## iter  40 value 14.416071
## iter  45 value 13.769022
## iter  50 value 12.552065
## final  value 12.552065
## stopped after 50 iterations
## initial  value 42.973752
## iter   5 value 41.639055
## iter  10 value 40.260879
## iter  15 value 35.567509
## iter  20 value 24.786195
## iter  25 value 18.483805
## iter  30 value 14.582254
## iter  35 value 12.689924
## iter  40 value 12.119605
## iter  45 value 11.812621
## final  value 11.651843
```

```
## converged
## initial  value 43.076258
## iter   5 value 41.687414
## iter   5 value 41.667637
## iter   5 value 41.667637
## final  value 41.667637
## converged
## initial  value 43.101792
## iter   5 value 41.524444
## iter  10 value 39.537878
## iter  15 value 36.129771
## iter  20 value 26.486680
## iter  25 value 18.469821
## iter  30 value 15.889505
## iter  35 value 13.216294
## iter  40 value 12.168404
## iter  45 value 11.736687
## iter  45 value 11.725899
## iter  45 value 11.721639
## final  value 11.721639
## converged
## initial  value 42.742036
## iter   5 value 41.655183
## iter   5 value 41.616456
## iter   5 value 41.613727
## final  value 41.613727
## converged
## initial  value 42.495312
## final  value 41.685183
## converged
## initial  value 43.069818
## iter   5 value 41.422888
## iter  10 value 38.657742
## iter  15 value 26.794211
## iter  20 value 13.854413
## iter  25 value 11.909203
## iter  30 value 11.639829
## iter  35 value 11.563501
## iter  35 value 11.553229
## iter  35 value 11.549246
## final  value 11.549246
## converged
## initial  value 42.985195
## iter   5 value 41.644630
## iter  10 value 40.812109
## iter  15 value 39.179498
## iter  20 value 35.876378
## iter  25 value 29.748469
## iter  30 value 22.025481
## iter  35 value 18.912712
## iter  40 value 14.467677
## iter  45 value 13.563239
## iter  50 value 12.389604
## final  value 12.389604
```

```
## stopped after 50 iterations
## initial  value 42.844485
## iter   5 value 41.514837
## iter  10 value 39.967877
## final  value 39.557375
## converged
## initial  value 42.562431
## iter   5 value 41.680026
## iter   5 value 41.646746
## iter   5 value 41.646143
## final  value 41.646143
## converged
## initial  value 42.596361
## iter   5 value 41.686429
## iter   5 value 41.672262
## iter   5 value 41.672262
## final  value 41.672262
## converged
## initial  value 42.955560
## iter   5 value 41.673050
## iter   5 value 41.640438
## iter   5 value 41.639382
## final  value 41.639382
## converged
## initial  value 43.100856
## iter   5 value 41.624579
## iter  10 value 40.557626
## iter  15 value 35.609003
## iter  20 value 25.730742
## iter  25 value 18.331261
## iter  30 value 15.013823
## iter  35 value 12.761126
## iter  40 value 12.448575
## iter  45 value 12.095755
## iter  50 value 11.991133
## final  value 11.991133
## stopped after 50 iterations
## initial  value 43.018264
## iter   5 value 41.619205
## iter  10 value 40.716087
## iter  15 value 39.155994
## iter  20 value 35.591939
## iter  25 value 24.651643
## iter  30 value 18.739921
## iter  35 value 14.385684
## iter  40 value 12.778798
## iter  45 value 12.109447
## iter  50 value 11.806782
## final  value 11.806782
## stopped after 50 iterations
## initial  value 42.403779
## iter   5 value 41.411931
## iter  10 value 39.625587
## iter  15 value 37.951499
```

```
## iter  20 value 30.317770
## iter  25 value 21.089394
## iter  30 value 17.431094
## iter  35 value 14.685037
## iter  40 value 13.178631
## iter  45 value 12.407665
## iter  50 value 12.131795
## final  value 12.131795
## stopped after 50 iterations
## initial  value 42.888419
## final  value 41.650397
## converged
## initial  value 42.613753
## iter   5 value 41.566800
## iter  10 value 41.159080
## iter  10 value 41.155241
## iter  15 value 39.277177
## iter  20 value 27.574305
## iter  25 value 17.286774
## iter  30 value 12.574487
## iter  35 value 11.913885
## iter  40 value 11.632873
## iter  45 value 11.521905
## iter  45 value 11.511425
## iter  45 value 11.507148
## final  value 11.507148
## converged
## initial  value 42.526494
## iter   5 value 41.441624
## iter  10 value 38.074926
## iter  15 value 26.769797
## iter  20 value 18.049496
## iter  25 value 13.441432
## iter  30 value 11.981993
## iter  35 value 11.685274
## final  value 11.608196
## converged
## initial  value 42.669826
## iter   5 value 41.646460
## iter   5 value 41.605465
## iter   5 value 41.605150
## final  value 41.605150
## converged
## initial  value 43.047913
## final  value 41.671305
## converged
## initial  value 42.874070
## final  value 41.678608
## converged
## initial  value 42.622667
## iter   5 value 41.614446
## iter  10 value 40.139624
## iter  15 value 36.880188
## iter  20 value 27.247469
```

```
## iter  25 value 21.027512
## iter  30 value 14.596548
## iter  35 value 13.134958
## iter  40 value 11.971549
## iter  45 value 11.724502
## final  value 11.670962
## converged
## initial  value 42.933475
## iter   5 value 41.635178
## iter  10 value 40.725203
## iter  15 value 33.271297
## iter  20 value 21.586867
## iter  25 value 16.196337
## iter  30 value 12.914376
## iter  35 value 11.824997
## iter  40 value 11.612950
## iter  45 value 11.451936
## iter  45 value 11.444002
## iter  45 value 11.440252
## final  value 11.440252
## converged
## initial  value 42.782969
## iter   5 value 41.562559
## iter  10 value 40.495858
## iter  15 value 34.675327
## iter  20 value 24.221564
## iter  25 value 15.896258
## iter  30 value 12.873695
## iter  35 value 12.162108
## iter  40 value 11.759714
## final  value 11.714560
## converged
## initial  value 42.811331
## iter   5 value 41.565581
## iter  10 value 39.862451
## iter  15 value 34.218117
## iter  20 value 24.455534
## iter  25 value 15.354424
## iter  30 value 12.523297
## iter  35 value 11.883367
## iter  40 value 11.599975
## iter  45 value 11.459807
## iter  45 value 11.449415
## iter  45 value 11.446316
## final  value 11.446316
## converged
## initial  value 42.612513
## iter   5 value 41.614813
## final  value 41.526551
## converged
## initial  value 43.064454
## iter   5 value 41.617307
## iter  10 value 40.261896
## iter  15 value 35.205736
```

```
## iter  20 value 22.013602
## iter  25 value 14.591255
## iter  30 value 12.923742
## iter  35 value 11.937363
## iter  40 value 11.764264
## final  value 11.615340
## converged
## initial  value 42.902298
## iter   5 value 41.467899
## iter  10 value 39.596659
## iter  15 value 34.694948
## iter  20 value 23.330674
## iter  25 value 17.289846
## iter  30 value 14.343120
## iter  35 value 12.642517
## iter  40 value 12.199490
## final  value 11.843602
## converged
## initial  value 42.884616
## iter   5 value 41.655768
## iter  10 value 39.988455
## iter  15 value 38.896531
## iter  20 value 32.745048
## iter  25 value 22.892299
## iter  30 value 18.829794
## iter  35 value 15.093835
## iter  40 value 14.014505
## iter  45 value 12.684139
## iter  50 value 12.375831
## final  value 12.375831
## stopped after 50 iterations
## initial  value 43.469521
## iter   5 value 41.622530
## iter  10 value 41.054958
## iter  15 value 39.539515
## iter  20 value 39.037353
## iter  25 value 34.164359
## iter  30 value 26.381206
## iter  35 value 22.516125
## iter  40 value 19.861637
## iter  45 value 18.448551
## iter  50 value 17.266596
## final  value 17.266596
## stopped after 50 iterations
## initial  value 42.768277
## iter   5 value 41.142883
## iter  10 value 39.635529
## iter  15 value 39.406660
## iter  20 value 38.799575
## iter  25 value 30.353257
## iter  30 value 22.427203
## iter  35 value 18.025745
## iter  40 value 15.970394
## iter  45 value 13.892683
```

```
## iter  50 value 12.317544
## final  value 12.317544
## stopped after 50 iterations
## initial  value 42.793641
## final  value 41.638571
## converged
## initial  value 42.831409
## iter   5 value 41.608288
## iter  10 value 40.796559
## iter  15 value 26.837930
## iter  20 value 15.214020
## iter  25 value 12.846845
## iter  30 value 11.993583
## iter  35 value 11.781860
## final  value 11.743331
## converged
## initial  value 42.915826
## iter   5 value 41.698397
## final  value 41.631597
## converged
## initial  value 42.685068
## iter   5 value 41.109283
## iter  10 value 39.366107
## iter  15 value 37.260143
## iter  20 value 30.952066
## iter  25 value 24.440765
## iter  30 value 18.723985
## iter  35 value 16.359983
## iter  40 value 13.520740
## iter  45 value 12.771517
## iter  50 value 11.901840
## final  value 11.901840
## stopped after 50 iterations
## initial  value 43.199755
## iter   5 value 41.668231
## iter   5 value 41.647589
## iter   5 value 41.647106
## final  value 41.647106
## converged
## initial  value 42.877118
## final  value 41.678230
## converged
## initial  value 42.899863
## iter   5 value 41.625759
## iter  10 value 40.621682
## iter  15 value 39.513931
## iter  20 value 38.788965
## iter  25 value 33.602410
## iter  30 value 26.684882
## iter  35 value 21.866778
## iter  40 value 19.355709
## iter  45 value 17.245518
## iter  50 value 16.093080
## final  value 16.093080
```

```
## stopped after 50 iterations
## initial  value 42.625453
## iter   5 value 41.641186
## iter   5 value 41.612700
## iter   5 value 41.612472
## final  value 41.612472
## converged
## initial  value 42.416685
## iter   5 value 41.443033
## iter  10 value 39.263062
## iter  15 value 34.789072
## iter  20 value 26.654925
## iter  25 value 21.028607
## iter  30 value 18.422584
## iter  35 value 14.704770
## iter  40 value 13.585379
## iter  45 value 12.732956
## iter  50 value 12.038282
## final  value 12.038282
## stopped after 50 iterations
## initial  value 42.821169
## final  value 41.648802
## converged
## initial  value 42.827339
## iter   5 value 41.478793
## iter  10 value 39.475424
## iter  15 value 32.155615
## iter  20 value 18.150458
## iter  25 value 13.411664
## iter  30 value 12.016895
## iter  35 value 11.734218
## final  value 11.657089
## converged
## initial  value 42.942474
## final  value 41.650279
## converged
## initial  value 42.890495
## iter   5 value 41.695130
## iter   5 value 41.658214
## iter   5 value 41.658213
## final  value 41.658213
## converged
## initial  value 42.875381
## iter   5 value 41.637954
## iter  10 value 39.987648
## iter  15 value 39.102833
## iter  20 value 33.337708
## iter  25 value 23.441562
## iter  30 value 17.417023
## iter  35 value 13.329526
## iter  40 value 12.747893
## iter  45 value 11.981758
## iter  45 value 11.969829
## iter  45 value 11.964774
```

```
## final  value 11.964774
## converged
## initial  value 42.767667
## iter   5 value 41.473747
## iter  10 value 39.343515
## iter  15 value 34.510375
## iter  20 value 20.936721
## iter  25 value 15.112082
## iter  30 value 12.289944
## iter  35 value 11.773799
## iter  40 value 11.626368
## final  value 11.579512
## converged
## initial  value 42.395488
## iter   5 value 41.595813
## iter  10 value 23.093503
## iter  15 value 11.815345
## iter  20 value 11.527922
## final  value 11.471164
## converged
## initial  value 42.458343
## final  value 41.689053
## converged
## initial  value 42.811890
## iter   5 value 41.560086
## iter  10 value 40.420169
## iter  15 value 36.938075
## iter  20 value 27.171209
## iter  25 value 20.861517
## iter  30 value 16.057564
## iter  35 value 13.985092
## iter  40 value 12.633530
## final  value 12.250686
## converged
## initial  value 42.872265
## iter   5 value 41.641841
## iter  10 value 40.429727
## iter  15 value 37.281871
## iter  20 value 30.313680
## iter  25 value 22.932530
## iter  30 value 20.355586
## iter  35 value 16.643340
## iter  40 value 14.906676
## iter  45 value 13.989242
## iter  50 value 13.140381
## final  value 13.140381
## stopped after 50 iterations
## initial  value 43.171908
## iter   5 value 41.551962
## final  value 40.778096
## converged
## initial  value 43.006920
## final  value 41.996701
## converged
```

```
## initial  value 42.526613
## iter   5 value 41.583277
## iter  10 value 39.890051
## iter  15 value 38.769654
## iter  20 value 30.576307
## iter  25 value 21.502628
## iter  30 value 18.108108
## iter  35 value 15.065587
## iter  40 value 13.939428
## iter  45 value 13.477950
## iter  50 value 12.224117
## final  value 12.224117
## stopped after 50 iterations
## initial  value 43.070045
## iter   5 value 41.583522
## iter  10 value 40.614731
## iter  15 value 39.233736
## iter  20 value 34.214488
## iter  25 value 26.635969
## iter  30 value 21.442142
## iter  35 value 19.915145
## iter  40 value 19.465565
## iter  45 value 18.561460
## iter  50 value 18.024242
## final  value 18.024242
## stopped after 50 iterations
## initial  value 43.311610
## iter   5 value 41.647033
## iter   5 value 41.621347
## iter   5 value 41.619072
## final  value 41.619072
## converged
## initial  value 42.994852
## iter   5 value 41.478535
## iter  10 value 40.122911
## iter  15 value 37.521961
## iter  20 value 28.030229
## iter  25 value 18.989103
## iter  30 value 15.097485
## iter  35 value 13.298036
## iter  40 value 12.253593
## iter  45 value 11.915941
## iter  50 value 11.652214
## final  value 11.652214
## stopped after 50 iterations
## initial  value 42.602344
## iter   5 value 41.638147
## iter  10 value 40.979657
## iter  15 value 39.406701
## iter  20 value 37.220395
## iter  25 value 30.265997
## iter  30 value 22.020789
## iter  35 value 18.022181
## iter  40 value 15.225340
```

```
## iter  45 value 13.946005
## iter  50 value 13.098476
## final  value 13.098476
## stopped after 50 iterations
## initial  value 43.891358
## final  value 42.278186
## converged
## initial  value 42.896989
## iter   5 value 40.992941
## iter  10 value 38.946807
## iter  15 value 27.401239
## iter  20 value 19.606724
## iter  25 value 16.021573
## iter  30 value 13.823727
## iter  35 value 12.350858
## iter  40 value 12.138541
## final  value 11.847142
## converged
## initial  value 42.864893
## iter   5 value 41.442339
## iter  10 value 39.625321
## iter  15 value 38.698658
## iter  20 value 31.338727
## iter  25 value 21.221849
## iter  30 value 17.044420
## iter  35 value 13.997652
## iter  40 value 13.468229
## iter  45 value 12.537732
## iter  50 value 12.049540
## final  value 12.049540
## stopped after 50 iterations
## initial  value 43.248140
## iter   5 value 41.651378
## iter  10 value 40.005666
## iter  15 value 38.793423
## iter  20 value 31.221566
## iter  25 value 24.041223
## iter  30 value 16.804883
## iter  35 value 13.589993
## iter  40 value 12.431453
## iter  45 value 11.896455
## final  value 11.726861
## converged
## initial  value 43.216688
## iter   5 value 41.557870
## iter  10 value 39.733602
## iter  15 value 38.436283
## iter  20 value 27.891216
## iter  25 value 17.423836
## iter  30 value 13.624892
## iter  35 value 12.403101
## iter  40 value 11.856456
## final  value 11.677474
## converged
```

```
## initial  value 42.910161
## iter   5 value 41.607021
## iter  10 value 39.994695
## iter  15 value 39.236339
## iter  20 value 36.265376
## iter  25 value 29.224511
## iter  30 value 22.365581
## iter  35 value 18.387424
## iter  40 value 14.515853
## iter  45 value 13.596999
## iter  50 value 12.417988
## final  value 12.417988
## stopped after 50 iterations
## initial  value 42.462172
## iter   5 value 40.550779
## iter  10 value 39.013172
## iter  15 value 30.562416
## iter  20 value 20.144212
## iter  25 value 14.555347
## iter  30 value 12.332681
## iter  35 value 11.877732
## iter  40 value 11.765732
## iter  40 value 11.756369
## iter  40 value 11.752116
## final  value 11.752116
## converged
## initial  value 43.597737
## iter   5 value 41.675799
## iter   5 value 41.657825
## iter   5 value 41.656167
## final  value 41.656167
## converged
## initial  value 42.535938
## iter   5 value 41.490325
## iter  10 value 39.453762
## iter  15 value 30.048518
## iter  20 value 19.113665
## iter  25 value 13.449352
## iter  30 value 12.445708
## iter  35 value 11.921879
## iter  40 value 11.703651
## iter  40 value 11.692619
## iter  40 value 11.688169
## final  value 11.688169
## converged
## initial  value 42.967122
## iter   5 value 41.608609
## iter  10 value 40.047395
## iter  15 value 38.675927
## iter  20 value 30.981302
## iter  25 value 22.217781
## iter  30 value 17.624606
## iter  35 value 14.133732
## iter  40 value 13.233029
```

```
## iter  45 value 12.208657
## iter  50 value 11.789817
## final  value 11.789817
## stopped after 50 iterations
## initial  value 42.667541
## iter   5 value 41.589941
## iter  10 value 40.105925
## iter  15 value 37.498266
## iter  20 value 28.152393
## iter  25 value 22.719758
## iter  30 value 16.604189
## iter  35 value 13.998706
## iter  40 value 12.354651
## iter  45 value 11.986054
## final  value 11.766140
## converged
## initial  value 42.851739
## iter   5 value 41.670996
## iter   5 value 41.630941
## iter   5 value 41.630488
## final  value 41.630488
## converged
## initial  value 42.583132
## final  value 41.670080
## converged
## initial  value 42.717382
## iter   5 value 41.676260
## iter   5 value 41.660053
## iter   5 value 41.660053
## final  value 41.660053
## converged
## initial  value 42.608287
## final  value 41.871482
## converged
## initial  value 42.698499
## iter   5 value 41.511157
## iter  10 value 39.471089
## iter  15 value 34.953929
## iter  20 value 23.064420
## iter  25 value 16.251467
## iter  30 value 12.787155
## iter  35 value 12.012495
## iter  40 value 11.725187
## iter  45 value 11.554780
## iter  45 value 11.544171
## iter  45 value 11.540323
## final  value 11.540323
## converged
## initial  value 43.345558
## iter   5 value 41.630632
## iter  10 value 40.617006
## iter  15 value 39.184428
## iter  20 value 35.313898
## iter  25 value 29.367295
```

```
## iter  30 value 24.871772
## iter  35 value 21.908552
## iter  40 value 18.430480
## iter  45 value 16.056193
## iter  50 value 15.820066
## final  value 15.820066
## stopped after 50 iterations
## initial  value 43.013722
## iter   5 value 41.690712
## iter   5 value 41.656800
## iter   5 value 41.656371
## final  value 41.656371
## converged
## initial  value 43.185204
## iter   5 value 41.459870
## iter  10 value 39.624378
## iter  15 value 39.322523
## iter  20 value 38.494913
## iter  25 value 33.204563
## iter  30 value 27.975050
## iter  35 value 23.702319
## iter  40 value 21.789992
## iter  45 value 19.669208
## iter  50 value 17.639905
## final  value 17.639905
## stopped after 50 iterations
## initial  value 42.663875
## iter   5 value 41.282785
## iter  10 value 39.229686
## iter  15 value 34.521673
## iter  20 value 22.365901
## iter  25 value 16.220412
## iter  30 value 13.402801
## iter  35 value 12.536993
## iter  40 value 12.058200
## iter  45 value 11.843710
## final  value 11.690051
## converged
## initial  value 42.776544
## iter   5 value 41.378004
## iter  10 value 37.005584
## iter  15 value 23.311578
## iter  20 value 14.705090
## iter  25 value 12.516813
## iter  30 value 11.920437
## iter  35 value 11.682452
## iter  40 value 11.566771
## final  value 11.518264
## converged
## initial  value 42.878178
## iter   5 value 41.518884
## iter  10 value 39.621007
## iter  15 value 38.442799
## iter  20 value 30.208153
```

```
## iter  25 value 20.925216
## iter  30 value 16.253719
## iter  35 value 14.034275
## iter  40 value 12.559776
## iter  45 value 12.004385
## iter  50 value 11.837702
## final  value 11.837702
## stopped after 50 iterations
## initial  value 42.817052
## iter   5 value 41.679735
## iter   5 value 41.664321
## iter   5 value 41.664321
## final  value 41.664321
## converged
## initial  value 42.543398
## iter   5 value 41.564643
## iter  10 value 40.530519
## iter  15 value 37.180633
## iter  20 value 28.038027
## iter  25 value 22.730946
## iter  30 value 16.627144
## iter  35 value 14.835625
## iter  40 value 13.898645
## iter  45 value 12.588647
## final  value 12.535176
## converged
## initial  value 43.139904
## iter   5 value 40.818296
## iter  10 value 38.986792
## iter  15 value 32.632444
## iter  20 value 22.127773
## iter  25 value 16.932471
## iter  30 value 13.860933
## iter  35 value 12.771270
## iter  40 value 12.167785
## iter  45 value 11.868640
## iter  45 value 11.858278
## iter  45 value 11.854124
## final  value 11.854124
## converged
## initial  value 42.932869
## final  value 42.193687
## converged
## initial  value 42.706820
## final  value 41.658415
## converged
## initial  value 42.701924
## final  value 41.665962
## converged
## initial  value 43.226702
## iter   5 value 41.716055
## iter   5 value 41.678815
## iter   5 value 41.678811
## final  value 41.678811
```

```
## converged
## initial  value 42.536754
## iter   5 value 41.517928
## iter  10 value 39.543560
## iter  15 value 31.839431
## iter  20 value 19.445321
## iter  25 value 13.893240
## iter  30 value 12.203766
## iter  35 value 11.856291
## iter  40 value 11.684325
## iter  40 value 11.677901
## iter  40 value 11.674350
## final  value 11.674350
## converged
## initial  value 42.330188
## final  value 41.664230
## converged
## initial  value 42.798785
## final  value 41.953961
## converged
## initial  value 42.549451
## iter   5 value 41.448864
## iter  10 value 38.840831
## iter  15 value 27.693335
## iter  20 value 19.366618
## iter  25 value 13.594749
## iter  30 value 12.456203
## iter  35 value 12.042850
## iter  40 value 11.795681
## final  value 11.708398
## converged
## initial  value 42.958900
## final  value 41.685603
## converged
## initial  value 43.402013
## iter   5 value 41.177024
## iter  10 value 39.280300
## iter  15 value 34.197693
## iter  20 value 27.612757
## iter  25 value 21.045158
## iter  30 value 16.455420
## iter  35 value 13.884829
## iter  40 value 12.860746
## iter  45 value 12.381250
## final  value 11.849048
## converged
```

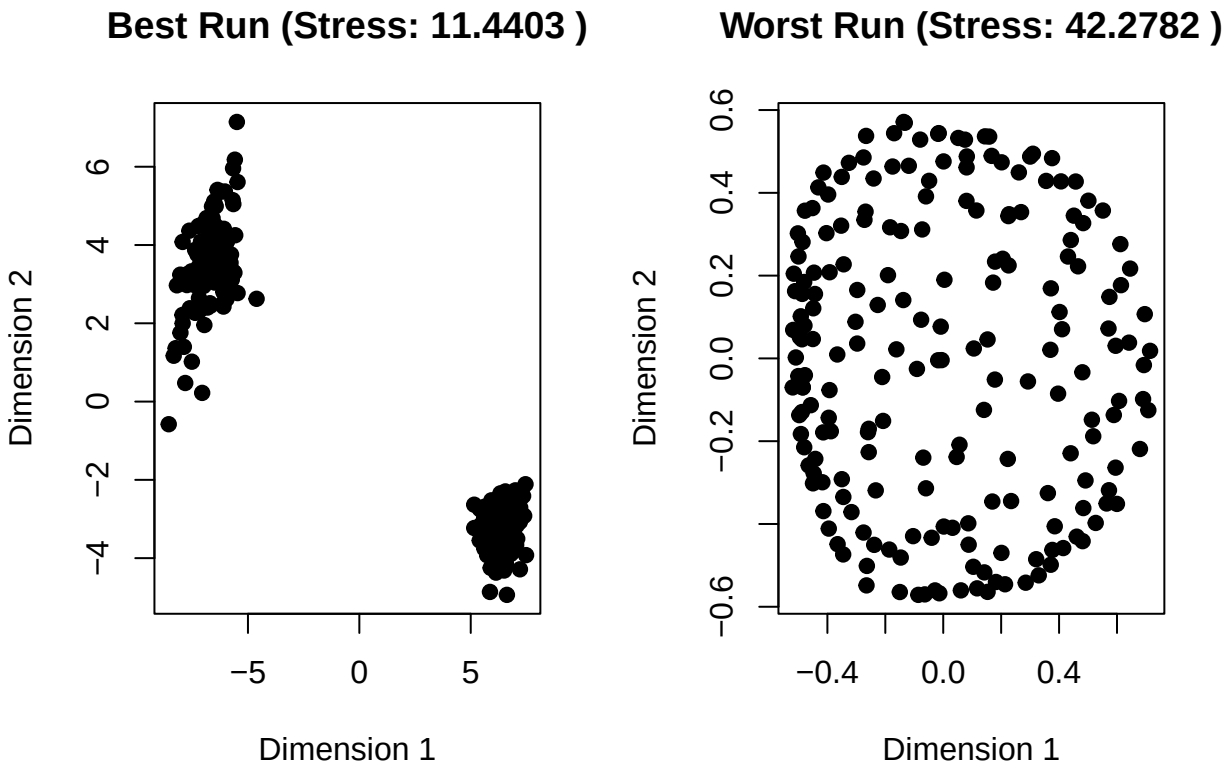
```
# best and worst runs
best_run <- which.min(stress_values)
worst_run <- which.max(stress_values)

cat("Best run stress:", stress_values[best_run], "\n")
```

```
## Best run stress: 11.44025
```

```r
cat("Worst run stress:", stress_values[worst_run], "\n")
```

```
## Worst run stress: 42.27819
```

```r
# Plot the best and worst MDS configurations
par(mfrow=c(1, 2))

plot(mds_configs[[best_run]][,1], mds_configs[[best_run]][,2],
     main = paste("Best Run (Stress:", round(stress_values[best_run], 4), ")"),
     xlab = "Dimension 1", ylab = "Dimension 2", pch = 19)

plot(mds_configs[[worst_run]][,1], mds_configs[[worst_run]][,2],
     main = paste("Worst Run (Stress:",
                  round(stress_values[worst_run], 4), ")"),
     xlab = "Dimension 1", ylab = "Dimension 2", pch = 19)
```




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

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.

The first dimension of metric MDS (Dimension 1) has a very high positive correlation with the first dimension of non-metric MDS. This indicates that both MDS methods are in strong agreement and are capturing a

similar pattern of variation along their respective first dimensions.

The first dimension of metric MDS also has a high negative correlation with the second dimension of non-metric MDS. A negative correlation means that as one dimension increases, the other decreases.

There is a low negative correlation between the second dimension of metric MDS and both dimensions of non-metric MDS. These low values suggest that the second dimension of the metric MDS does not correspond well to either dimension of the non-metric MDS, indicating dissimilarity in the patterns captured by this particular dimension across the two methods.

The strong correlations (positive and negative) indicate that while there is some agreement between the methods on the primary dimension of variation, there is a significant difference in how the secondary dimensions are represented.

```
# best non-metric MDS run
best_nonmetric_mds_points <- mds_configs[[best_run]]

cor_matrix <- cor(mds_scaled, best_nonmetric_mds_points)

print(cor_matrix)
```

```
##           [,1]       [,2]
## V1  0.99470782 -0.9646396
## V2 -0.01898338 -0.1089204
```

```
image(1:2, 1:2, as.matrix(cor_matrix), main="Correlation Matrix", xlab="Metric MDS Dimensions", ylab="N
axis(1, at=1:2, labels=c("Dimension 1", "Dimension 2"))
axis(2, at=1:2, labels=c("Dimension 1", "Dimension 2"), las=2)  # 'las=2' makes the y-axis labels perpe
```