MDS Assessment

Roger Bukuru

2024-02-18

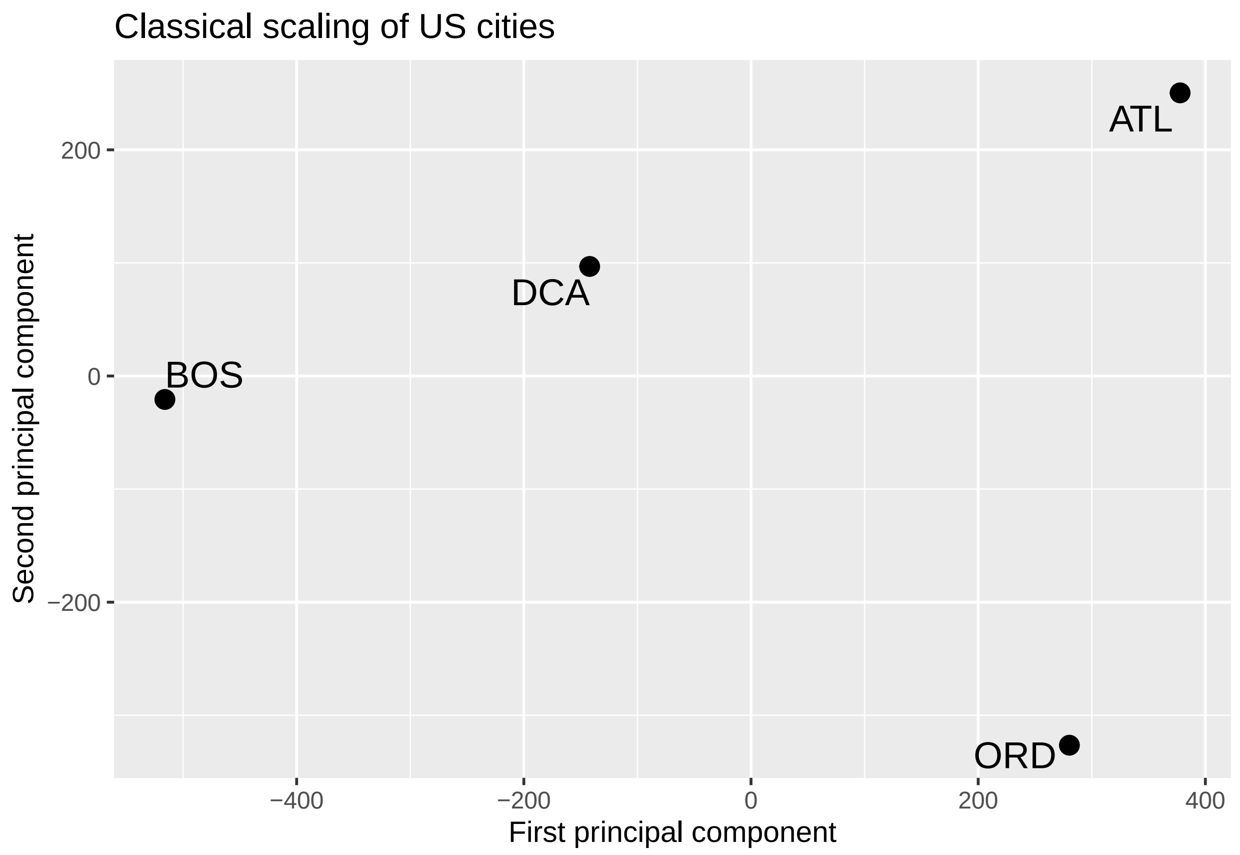
Question 1

For this exercise we will be using the cities data that is found in the psych package in R. Run the following lines of code to obtain the dissimilarity matrix that we will work with.

a) Using the first 4 (four) cities, ATL, BOS, ORD and DCA. Perform the classical scaling algorithm by first principals to obtain the principals coordinates. Plot the principal coordinates in 2 dimensions. You can use R to create matrices and perform the Eigen value decomposition.

```r
rm(list = ls())
library(psych)
library(psychTools)
library(tibble)
library(dplyr)
library(ggplot2)
library(ggrepel)
data(cities)

classical_scaling_first_principals = function(noOfCities = 4,
    rotateMap = FALSE) {
    rotateFactor = -1
    if (!rotateMap) {
        rotateFactor = 1
    }
    proximity_matrix = as.matrix(cities[1:noOfCities, 1:noOfCities])

    # A = -0.5 * sqrt(proximity_matrix)
    A = (-1/2) * proximity_matrix^2

    # B = HAH H = I-1/n11^1
    n = nrow(A)
    H = diag(1, nrow = n) - 1/n * matrix(1, nrow = n, ncol = ncol(A))
    B = H %*% A %*% H

    # Objective function Y = Eigenvectors %*%
    # diag(eigenvalues)
    evd = eigen(B)
    Y = evd$vectors %*% diag(sqrt(evd$values))  # Principal components

    principal_comp = Y %>%
        as.tibble() %>%
        mutate(city = colnames(proximity_matrix))
    # First two components, represent comps with the most
    # variance in representing the data with comp 1 having
    # the most variance etc.
    two_dim_plot = ggplot(principal_comp, mapping = aes(rotateFactor *
        V1, rotateFactor * V2)) + geom_point(size = 3) + geom_text_repel(aes(label = city),
        size = 5) + labs(x = "First principal component", y = "Second principal component",
        title = "Classical scaling of US cities")
    coord_equal()

    two_dim_plot

}
```

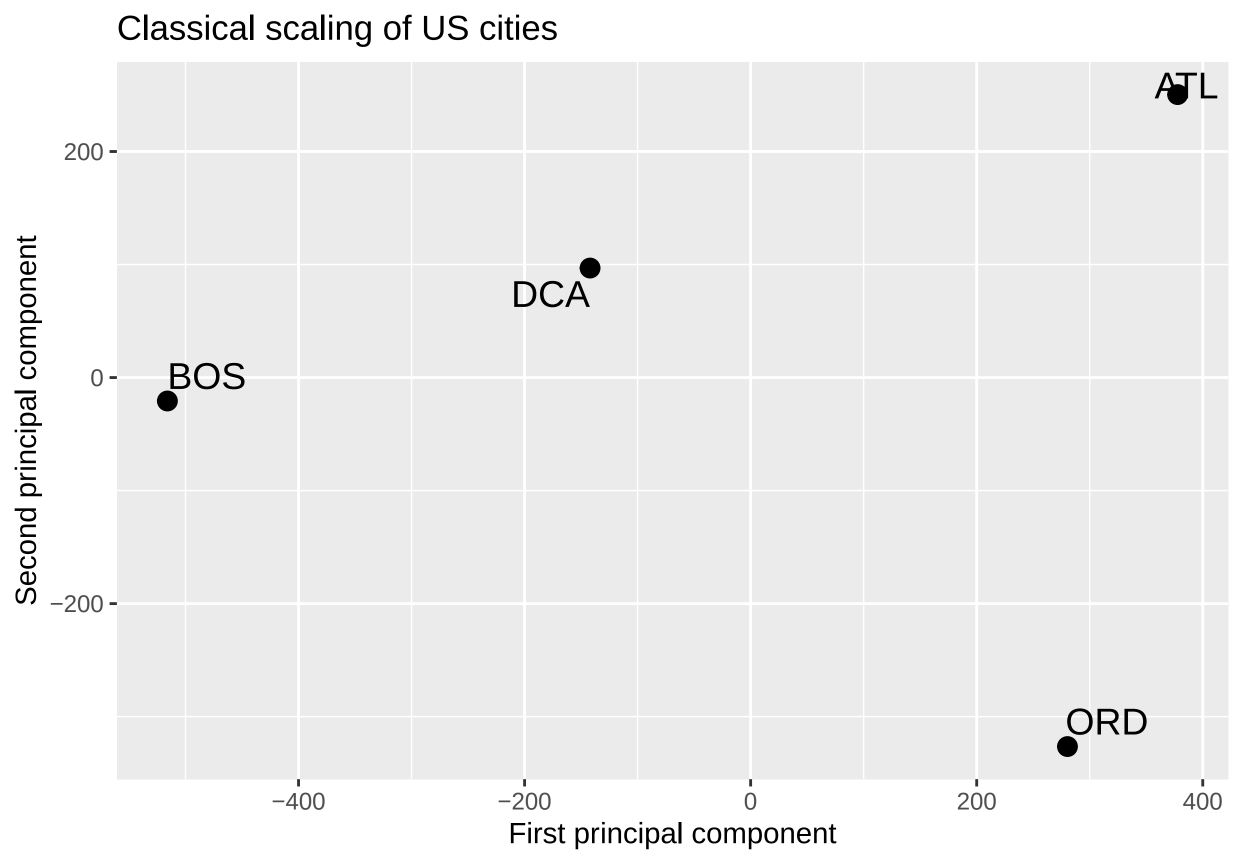
```
classical_scaling_first_principals()
```



b) Use the function cmdscale in R to check check your answer in part (a).

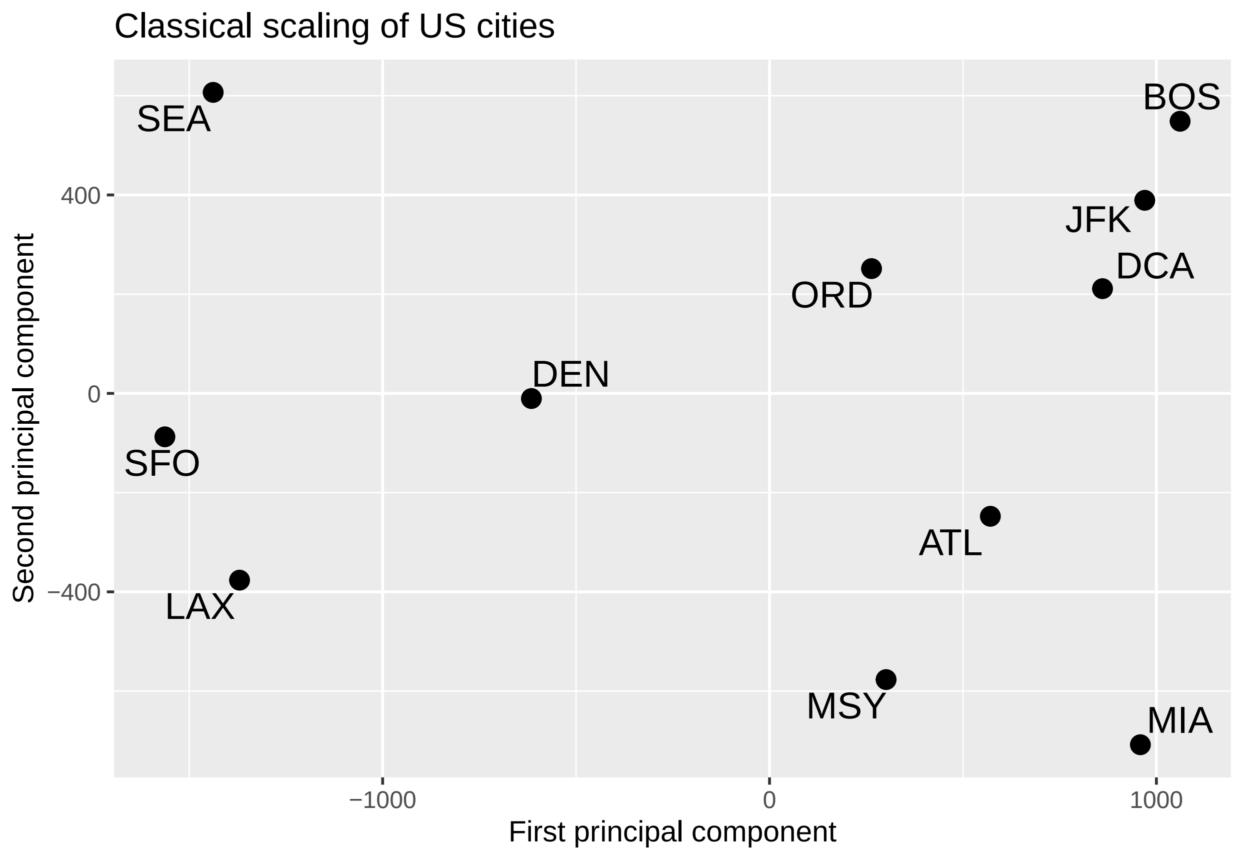
```
proximity_matrix = as.matrix(cities[1:4, 1:4])
evd = cmdscale(proximity_matrix, eig = TRUE, list. = TRUE)
Y = evd$points
principal_comp = Y %>%
    as.tibble() %>%
    mutate(city = colnames(proximity_matrix))
```

```
ggplot(principal_comp, aes(V1, V2)) + geom_point(size = 3) +
    geom_text_repel(aes(label = city), size = 5) + labs(x = "First principal component",
    y = "Second principal component", title = "Classical scaling of US cities") +
    coord_equal()
```



c) Perform the classical scaling algorithm to the entire dissimilarity matrix and obtain the principal coordinates. Plot the resulting coordinates in 2 dimensions.

```
classical_scaling_first_principals(noOfCities = 11, rotateMap = TRUE)
```

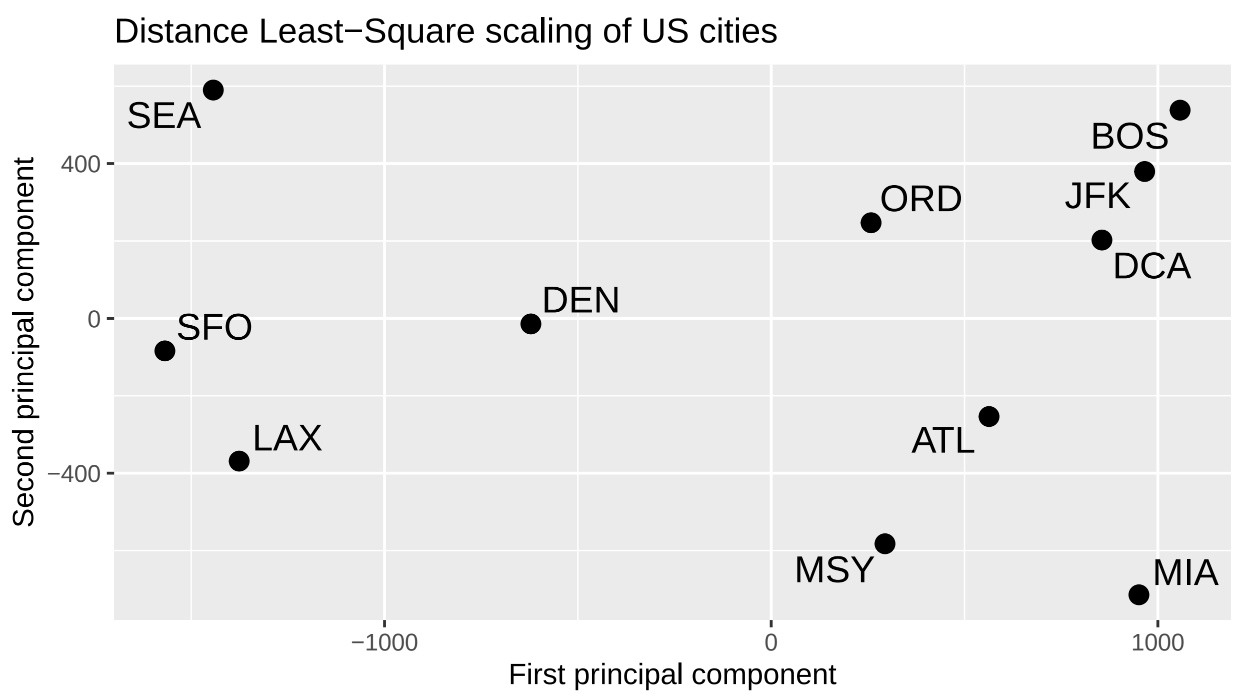


Question 2

a) (i) Use the mds() function – in the smacof package – to perform metric multidimensional scaling. Locate the cities in t=2 dimensions, starting with a classical scaling solution as the initial configuration.

From from first principles

```
# Step 1: Dissimilarity matrix
proximity_matrix = as.matrix(cities)
# Step 2: Initial config points y1....yn using classical
# scaling
low_dim_points = cmdscale(proximity_matrix)


stress_fun = function(y, dissimilarity_matrix, principalComp = 2) {
    y_mat = matrix(y, ncol = principalComp)  # points in lower-dim
    d_mat = as.matrix(dist(y_mat))  # distance of points in lower-dim
    # high dim transform fun = identity function i.e
    # maintains the dissimilarities from higher-dim as
    # distances
    weight_mat = 1/sum(dissimilarity_matrix[lower.tri(dissimilarity_matrix)]^2)
    error_mat = ((d_mat - dissimilarity_matrix)^2)[lower.tri(dissimilarity_matrix)]
    weight_mat * sum(error_mat)
}

min_stress_fun = optim(low_dim_points, stress_fun, dissimilarity_matrix = proximity_matrix)

optimal_distance_scaling_points = matrix(min_stress_fun$par,
    ncol = 2)

fitted_configs = optimal_distance_scaling_points %>%
    as.tibble() %>%
    mutate(city = colnames(proximity_matrix))


ggplot(fitted_configs, aes(-V1, -V2), alpha = 0.5, color = "blue",
    size = 10) + geom_point(size = 3) + geom_text_repel(aes(label = city),
    size = 5) + labs(x = "First principal component", y = "Second principal component",
    title = "Distance Least-Square scaling of US cities") + coord_equal()
```

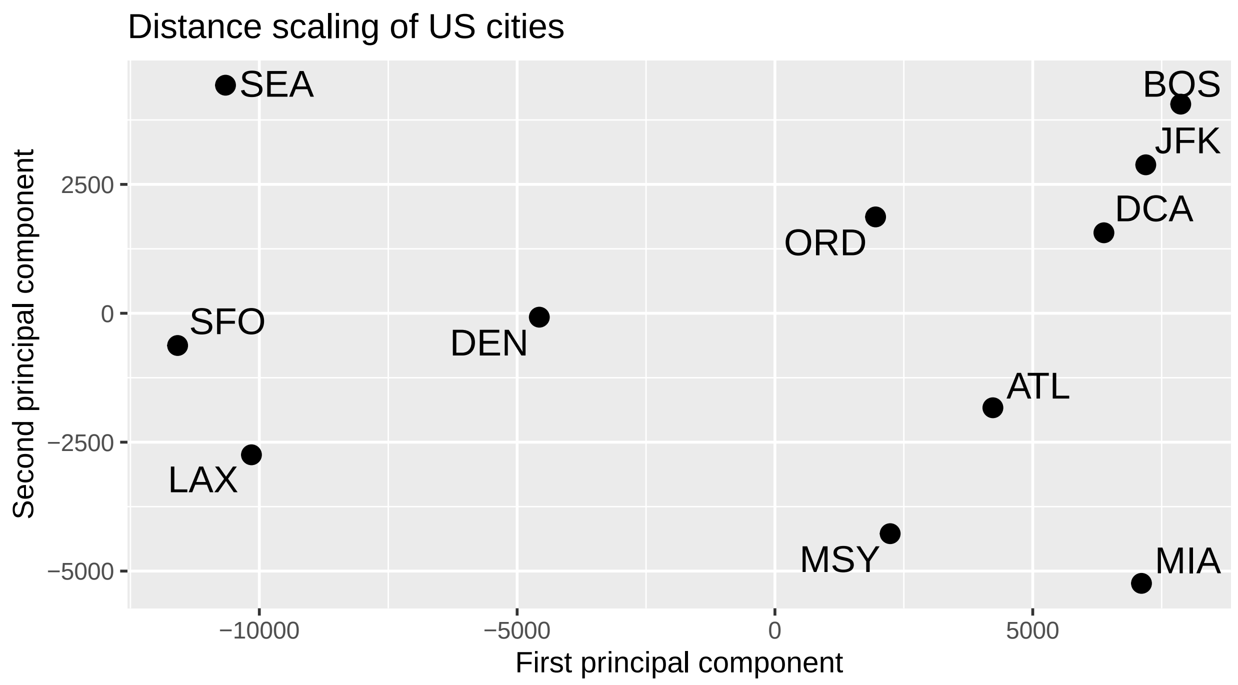


```
min_stress_fun$value
```

```
## [1] 3.331485e-06
```

Using mds function

```r
library(smacof)

weight_mat = matrix(0, ncol = 11, nrow = 11)
constant_weight = 1/sum(proximity_matrix[lower.tri(proximity_matrix)]^2)
weight_mat[lower.tri(weight_mat)] = constant_weight


metric_mds = mds(delta = proximity_matrix, ndim = 2, init = "torgerson",
    type = "ratio", weightmat = weight_mat)

fitted_configs_2 = metric_mds$conf %>%
    as.tibble() %>%
    mutate(city = colnames(proximity_matrix))

ggplot(fitted_configs_2, aes(-D1, -D2), alpha = 0.5, color = "blue",
    size = 10) + geom_point(size = 3) + geom_text_repel(aes(label = city),
    size = 5) + labs(x = "First principal component", y = "Second principal component",
    title = "Distance scaling of US cities") + coord_equal()
```



(ii) Which of the 11 cities has the poorest fit? Explain your answer.

Per Point Stress Points

```r
# metric_mds$stress
metric_mds$spp
```

```
##        ATL        BOS        ORD        DCA        DEN        LAX        MIA
##  2.6440777  7.9040554  1.7018267  0.7010979  1.5993119 20.1502872 12.2196378
##        JFK        SEA        SFO        MSY
##  2.3776837 33.4778124 13.6394028  3.5848066
```

Observing the per point stress points we note that SEA is the city with the poorest fit as it has the highest stress point of 33.48%, this means that it has been matched the poorest relative to it's original dissimilarities as it has the highest contribution to the stress value of 0.00206447. Could indicate an outlier or it's dissimilarities with the other cities might be inconsistent with the rest of the data

(iii) Plot the 2D multidimensional scaling configuration and compare this the locations of the cities on a map from the atlas.

From the map below we observe that the the low-dimensional map is a good representation of the Atlas mapping indicating the actual positions of the cities. We further support this with the relavtively low stress value of 0.00206447, indicating that the points in the low-dim have been matched relatively well with their original dissimilarities.

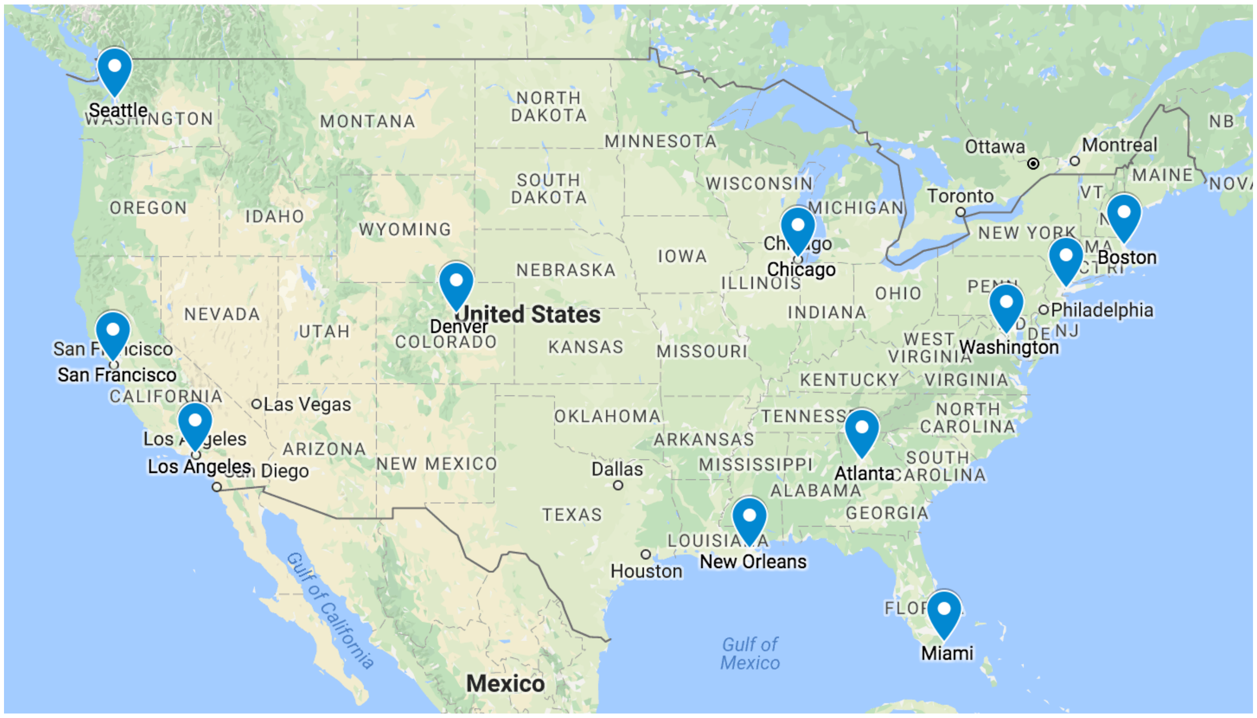
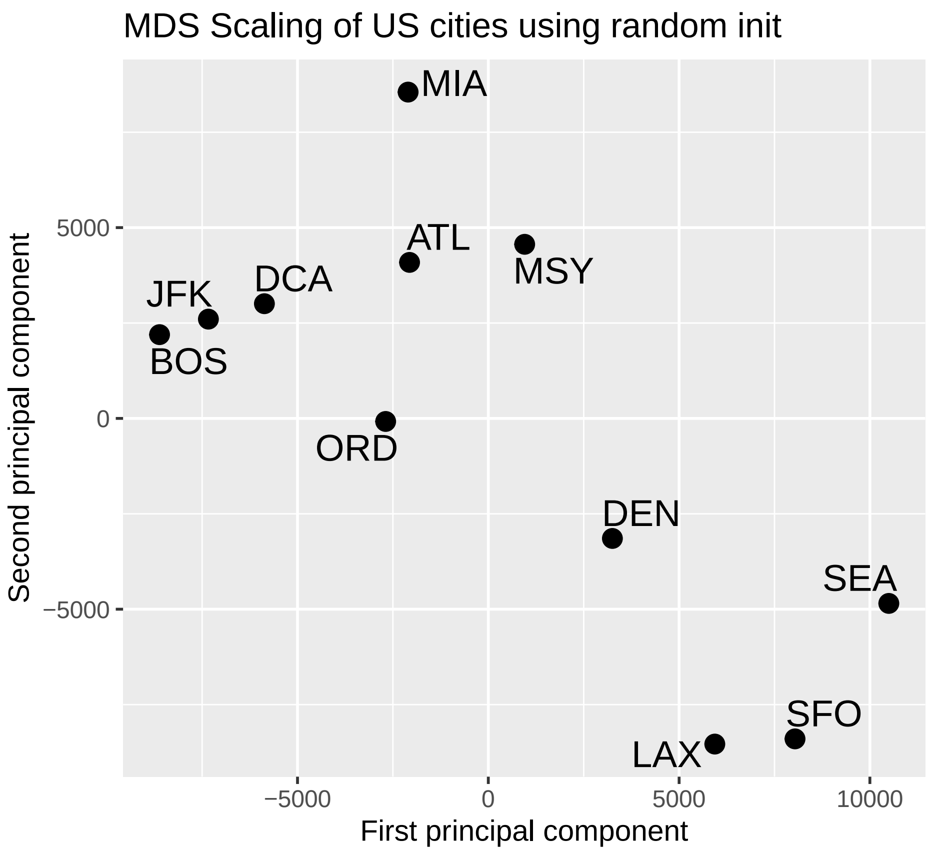


Figure 1: US Cities Map

b)Repeat the analysis in part (a) using a random starting solution. How does the different initial configuration change the location of the cities in the 2D configuration?

```r
metric_mds_random = mds(delta = proximity_matrix, ndim = 2, init = "random",
    type = "ratio", weightmat = weight_mat)
```

```r
fitted_configs_random = metric_mds_random$conf %>%
    as.tibble() %>%
    mutate(city = colnames(proximity_matrix))
ggplot(fitted_configs_random, aes(-D1, -D2), alpha = 0.5, color = "blue",
    size = 10) + geom_point(size = 3) + geom_text_repel(aes(label = city),
    size = 5) + labs(x = "First principal component", y = "Second principal component",
    title = "MDS Scaling of US cities using random init") + coord_equal()
```



```r
metric_mds_random$stress
```

```
## [1] 0.06024798
```

Using a random starting point we note that the resulting fit is quite poor. SEA has the poorest fit as firstly it's in the wrong direction and secondly it's closer to LAX and SFO hen it should be. The cities on the east are all poorly represented as well. This shows that a random starting point is not effective. We note as well that the stress value of 0.06023268 is higher than when we use classical scaling as the init config.

Notes

a) What do you understand about multidimensional scaling

Multidimensional scaling (MDS) is a statistical technique used for analyzing similarity or dissimilarity data. It's a method that allows researchers to visualize the level of similarity of individual cases of a dataset in a multidimensional space. The main idea behind MDS is to translate similarities or distances among data points into a geometric map, where each item is represented as a point in a low-dimensional space, typically two or three dimensions for easy visualization.

The process of MDS involves several key steps:

1. **Distance Matrix**: First, it starts with a distance matrix representing the dissimilarities between pairs of items. These dissimilarities can be based on direct measurements or derived from other data forms, such as correlation coefficients or other metrics.
2. **Dimensionality Reduction**: MDS seeks to place each item in an N-dimensional space (where N is small, usually 2 or 3) such that the distances between points in this space match the given dissimilarities as closely as possible. This is achieved through an optimization process that iteratively adjusts the positions of points to minimize the difference between the original dissimilarities and the distances in the constructed space.
3. **Visualization and Interpretation**: The resulting geometric map allows for visual inspection of the data's structure. Points that are close together indicate similar items, whereas points that are far apart indicate dissimilar items. This visualization can uncover patterns, groupings, or relationships within the data that might not be apparent from the raw data alone.

There are two main types of MDS:

- **Classical (Metric) MDS**: Assumes that the dissimilarity data are metric (i.e., they can be treated as distances in a geometric sense) and works by preserving the original distances as accurately as possible.
- **Nonmetric MDS**: Used when the dissimilarity data are ordinal. It attempts to preserve the rank order of the distances rather than their exact values, making it more suitable for data where precise distances are not available or meaningful.

MDS is widely used in various fields, including psychology (for perceptual mapping), marketing (for brand positioning analyses), sociology, and bioinformatics (for genetic and species relationship mapping), among others. It's a powerful tool for exploratory data analysis, allowing researchers to uncover underlying patterns in complex datasets by providing a visual representation of the relationships among data points.

b) does classical scaling fall part of metric MDS

Yes, Classical Scaling, often referred to as Classical Multidimensional Scaling (CMDS) or Torgerson Scaling, is a type of Metric Multidimensional Scaling (MDS). Metric MDS refers to a family of techniques where the goal is to place items in a low-dimensional space such that the distances between items in this space reflect the dissimilarities measured in the original high-dimensional space as closely as possible, under the constraint that these dissimilarities are treated as metric distances.

Classical Scaling specifically aims to preserve the pairwise Euclidean distances between points in the lower-dimensional representation as accurately as possible, based on the original dissimilarity matrix. It does so by using eigenvalue decomposition on a transformed version of the dissimilarity matrix, allowing for the reconstruction of points in a new space that best approximates the original distances.

The process involves:

1. **Creating a Dissimilarity Matrix**: Starting with a matrix that represents the dissimilarities (often distances) between pairs of items.
2. **Double Centering**: Transforming this matrix into a matrix of scalar products using double centering (subtracting the mean row and column values and adding the overall mean).

3. **Eigenvalue Decomposition**: Performing an eigenvalue decomposition on this double-centered matrix to find the principal components.
4. **Dimensionality Reduction**: Selecting the top n components to represent the data in a lower-dimensional space, aiming to preserve the original distances as much as possible.

Classical MDS is particularly useful when the dissimilarities are Euclidean distances or can be interpreted as such, making it a powerful tool for visualizing the structure of data in a way that is easy to understand. It's part of the broader category of metric MDS because it directly utilizes the metric properties of the dissimilarity measures.

c) After performing classical MDS, and we only plot the first two components what does this mean ?

Classical Multidimensional Scaling (MDS) is a technique used to reduce the dimensionality of data while preserving the distances between points as accurately as possible. When you perform classical MDS and plot only the first two components, you are essentially doing the following:

1. **Dimensionality Reduction**: Classical MDS starts with a matrix of distances (or dissimilarities) between pairs of items in a dataset. The goal is to represent these items as points in a lower-dimensional space (in this case, a 2D space) in such a way that the distances between points in this new space approximate the original distances as closely as possible.

2. **Capturing the Most Variance**: The first two components in MDS are the dimensions that capture the most variance in the dataset, based on the original distance matrix. By plotting these two components, you are visualizing the data in a way that emphasizes the most significant relationships between points according to the criteria used (usually Euclidean distance, but other metrics can be used).

3. **Interpretation**: Plotting only the first two components provides a simplified view of the dataset that can be easily visualized and interpreted. This can reveal patterns, clusters, or relationships that might not be apparent in the higher-dimensional space. For instance, points that are close together in the plot are similar to each other according to the distance metric used, while points that are far apart are dissimilar.

4. **Limitation in Detail**: While plotting the first two components helps in visualizing and interpreting complex datasets, it also means that some information is lost. The two components might not capture all the nuances of the relationships between items in the dataset. The amount of variance these two components capture varies depending on the dataset. In some cases, they might capture a substantial portion of the total variance, making the 2D representation quite informative. In other cases, they might capture a smaller portion, meaning that important relationships might not be fully represented in the plot.

5. **Summary**: Plotting the first two components after classical MDS is a useful technique for exploring and visualizing the structure of high-dimensional data in a more interpretable two-dimensional form. It helps in identifying broad patterns and relationships but should be complemented with other analyses or plots of additional components if a more comprehensive understanding of the data's structure is necessary.

In summary, when you plot only the first two components after performing classical MDS, you're simplifying and visualizing the dataset to make it easier to identify and interpret the most significant patterns and relationships, while acknowledging that some details and nuances may not be captured in this reduced representation.