Multidimensional Scaling I

Mike Strube

October 11, 2018

1 Preliminaries

In this section, the RStudio workspace and console panes are cleared of old output, variables, and other miscellaneous debris. Packages are loaded and any required data files are retrieved.

```
options(replace.assign = TRUE, width = 65, digits = 4, scipen = 4, fig.width = 4,
    fig.height = 4)
# Clear the workspace and console.
rm(list = ls(all = TRUE))
cat("\f")
```

```
# Turn off showing of significance asterisks.
options(show.signif.stars = F)
# Set the contrast option; important for ANOVAs.
options(contrasts = c("contr.sum", "contr.poly"))
how_long <- Sys.time()
set.seed(123)
library(knitr)
```

```
library(psych)
library(ggplot2)

##
## Attaching package:  'ggplot2'
## The following objects are masked from 'package:psych':
##
##    %+%, alpha

library(MASS)
library(sciplot)
library(ggplot2)
library(vegan)

## Warning:  package 'vegan' was built under R version 3.5.1
## Loading required package:  permute
## Warning:  package 'permute' was built under R version 3.5.1
## Loading required package:  lattice
## This is vegan 2.5-2

library(smacof)
```

```
## Warning:  package 'smacof' was built under R version 3.5.1
## Loading required package:  plotrix
##
## Attaching package:  'plotrix'
## The following object is masked from 'package:psych':
##
##    rescale
##
## Attaching package:  'smacof'
## The following object is masked from 'package:base':
##
##    transform
```

```
library(ape)
library(ade4)
```

```
## Warning:  package 'ade4' was built under R version 3.5.1
```

```
library(ecodist)
```

```
## Warning:  package 'ecodist' was built under R version 3.5.1
##
## Attaching package:  'ecodist'
## The following object is masked from 'package:vegan':
##
##    mantel
```

```
library(scatterplot3d)
```

1.1 Data Files

We will use distances between European cities for the initial example of metric MDS. The Cars data set will be used for the second example. It contains pairwise rankings, with lower numbers indicating pairs of cars that were viewed as being more similar to each other.

```
# Get the drug use data from the working directory.
setwd("C:\\Courses\\Psychology 516\\PowerPoint\\2018")
Cities <- read.table("cities.csv", sep = ",", header = TRUE)
Cities_Names <- as.vector(Cities[, 1])
Cities_Matrix <- as.matrix(Cities[, 2:ncol(Cities)])
Cities_Dist <- as.dist(Cities_Matrix)
Cities_Dist
```

```
##        Athens Berlin Dublin London Madrid Paris Rome
## Berlin   1119
## Dublin   1777    817
## London   1486    577    291
## Madrid   1475   1159    906    783
## Paris    1303    545    489    213    652
## Rome      646    736   1182    897    856   694
## Warsaw   1013    327   1135    904   1483   859  839
```

```
Cars <- read.table("car_dissim.csv", sep = ",", header = TRUE)
Cars_Names <- as.vector(Cars[, 1])
```

```
Cars_Matrix <- as.matrix(Cars[, 2:ncol(Cars)])
Cars_Dist <- as.dist(Cars_Matrix)
Cars_Dist
```

```
##          BMW Ford Infiniti Jeep Lexus Chrysler Mercedes Saab
## Ford      34
## Infiniti   8   24
## Jeep      31    2       25
## Lexus      7   26        1   27
## Chrysler  43   14       35   15    37
## Mercedes   3   28        5   29     4       42
## Saab      10   18       20   17    13       36       19
## Porsche    6   39       41   38    40       45       32   21
## Volvo     33   11       22   12    23        9       30   16
##          Porsche
## Ford
## Infiniti
## Jeep
## Lexus
## Chrysler
## Mercedes
## Saab
## Porsche
## Volvo         44
```

2 Available Metric Methods

There are a number of packages in R that can conduct metric MDS. Following are several examples. Some have the advantage of making it relatively easy to get the stress values and coordinates for plotting. Some allow specifying the particular number of dimensions to extract.

2.1 cmdscale( )

The cmdscale( ) function comes as part of the base R installation.

```
mds_1 <- cmdscale(Cities_Dist, k = 1, eig = TRUE, add = FALSE, x.ret = FALSE)
mds_1
```

```
## $points
##           [,1]
## Athens 1011.09
## Berlin   76.52
## Dublin -714.65
## London -432.23
## Madrid -406.71
## Paris  -274.12
## Rome    368.46
## Warsaw  371.64
##
## $eig
## [1]  2.240e+06  1.131e+06  1.108e+04  2.503e+02 -2.626e-10
```

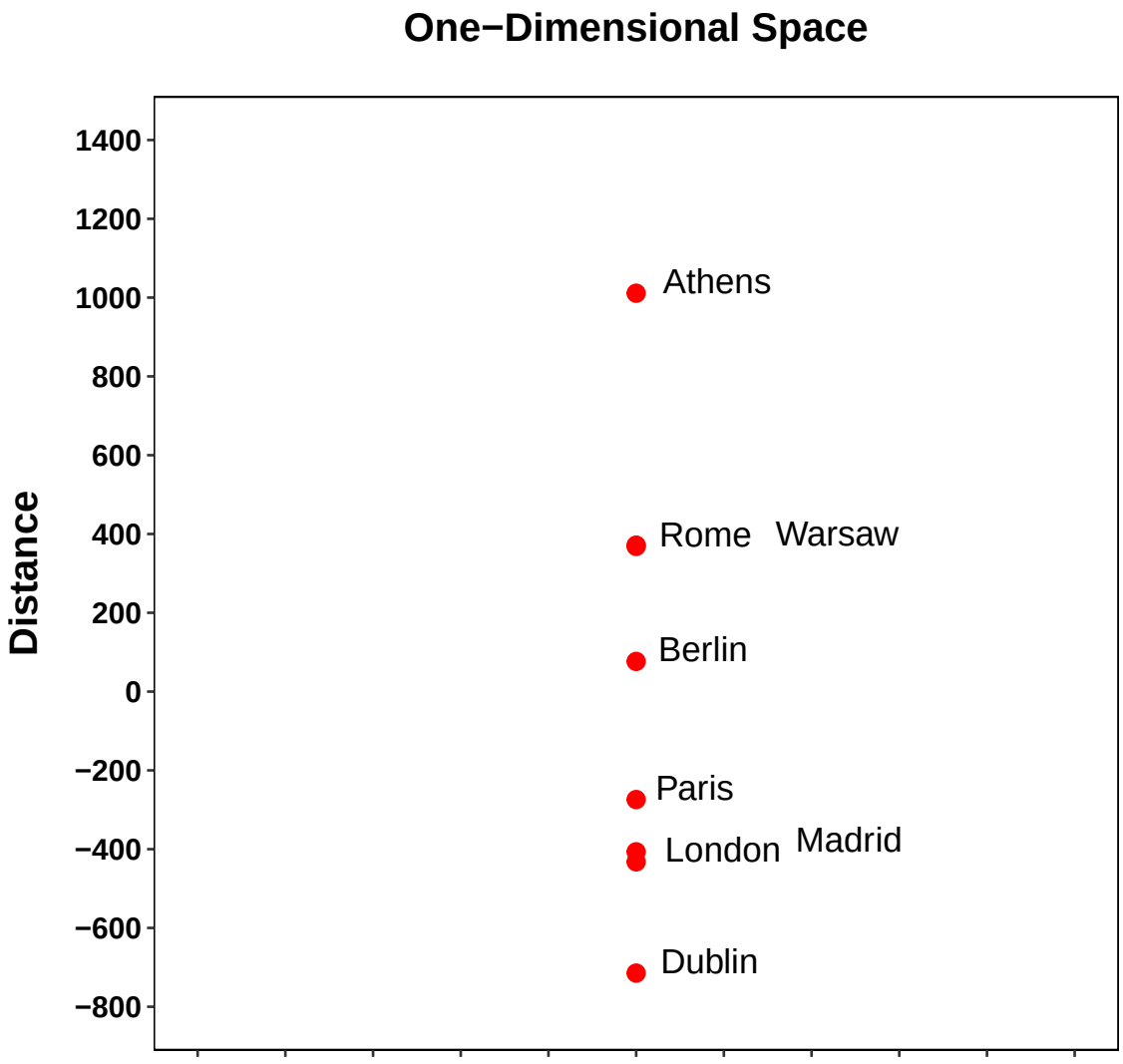
```
## [6] -4.600e+01 -1.652e+03 -5.432e+04
##
## $x
## NULL
##
## $ac
## [1] 0
##
## $GOF
## [1] 0.6514 0.6622
```

```
mds_2 <- cmdscale(Cities_Dist, k = 2, eig = TRUE, add = FALSE, x.ret = FALSE)
mds_2
```

```
## $points
##           [,1]    [,2]
## Athens 1011.09 -239.30
## Berlin   76.52  375.28
## Dublin -714.65  183.71
## London -432.23  113.89
## Madrid -406.71 -688.50
## Paris  -274.12  -27.98
## Rome    368.46 -289.71
## Warsaw  371.64  572.61
##
## $eig
## [1]  2.240e+06  1.131e+06  1.108e+04  2.503e+02 -2.626e-10
## [6] -4.600e+01 -1.652e+03 -5.432e+04
##
## $x
## NULL
##
## $ac
## [1] 0
##
## $GOF
## [1] 0.9804 0.9966
```

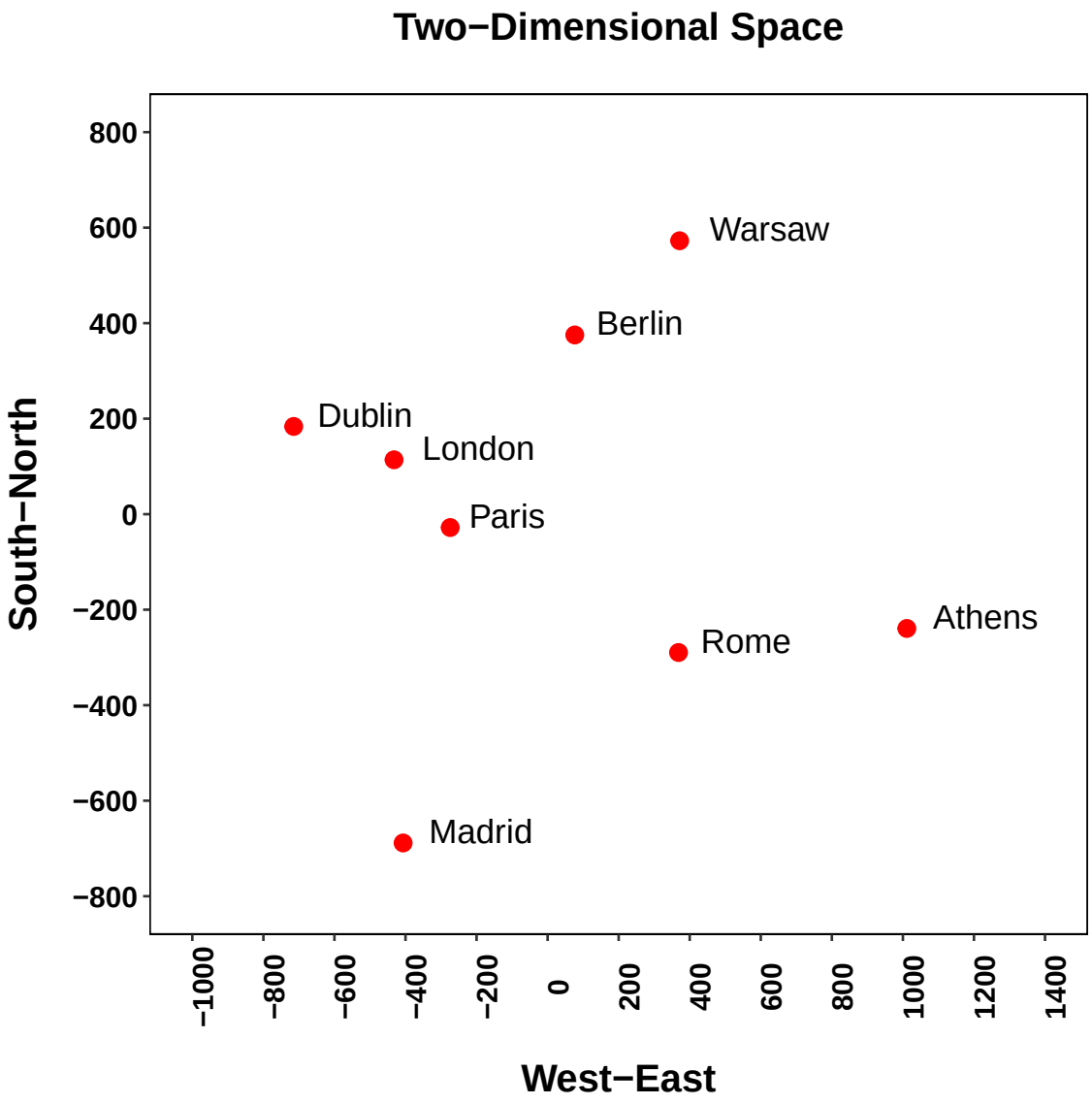
```
plot_data <- as.data.frame(mds_1$points)
names(plot_data) <- c("D")
plot_data$Name <- c("Athens", "Berlin", "Dublin", "London", "           Madrid",
    "Paris", "Rome", "         Warsaw")
plot_data$x <- 0

ggplot(plot_data, aes(x = x, y = D)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-800,
    1400, 200))) + scale_x_continuous(breaks = c(seq(-1000, 1000,
    200))) + geom_text(aes(label = Name), hjust = -0.25, vjust = 0,
    size = 5) + coord_cartesian(xlim = c(-1000, 1000), ylim = c(-800,
    1400)) + xlab("") + ylab("Distance") + theme(text = element_text(size = 14,
    family = "sans", color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_blank(), axis.title.x = element_text(margin = margi
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
```

```
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("One-Dimensional Space")
```




```
plot_data <- as.data.frame(mds_2$points)
names(plot_data) <- c("W_E", "S_N")
plot_data$Name <- row.names(plot_data)

ggplot(plot_data, aes(x = W_E, y = S_N)) + geom_point(shape = 19,
```

```
    size = 3, color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-800,
    800, 200))) + scale_x_continuous(breaks = c(seq(-1000, 1400, 200))) +
    geom_text(aes(label = Name), hjust = -0.25, vjust = 0, size = 5) +
    coord_cartesian(xlim = c(-1000, 1400), ylim = c(-800, 800)) +
    xlab("West-East") + ylab("South-North") + theme(text = element_text(size = 14,
    family = "sans", color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Two-Dimensional Space")
```



2.2 wcmdscale( )

The wcmdcmdscale( ) function is part of the vegan package.

```
mds_3 <- wcmdscale(Cities_Dist, k = 1, add = FALSE, x.ret = FALSE,
    w = rep(1, 8))
mds_3

##           [,1]
## Athens 1011.09
## Berlin   76.52
## Dublin -714.65
## London -432.23
## Madrid -406.71
```

```
## Paris  -274.12
## Rome    368.46
## Warsaw  371.64

mds_4 <- wcmdscale(Cities_Dist, k = 2, add = FALSE, x.ret = FALSE,
    w = rep(1, 8))
mds_4

##           [,1]    [,2]
## Athens 1011.09 -239.30
## Berlin   76.52  375.28
## Dublin -714.65  183.71
## London -432.23  113.89
## Madrid -406.71 -688.50
## Paris  -274.12  -27.98
## Rome    368.46 -289.71
## Warsaw  371.64  572.61

mds_4 <- wcmdscale(Cities_Dist, k = 2, add = FALSE, x.ret = TRUE,
    w = rep(1, 8))
mds_4

## Call: wcmdscale(d = Cities_Dist, k = 2, add = FALSE,
## x.ret = TRUE, w = rep(1, 8))
##
##           Inertia Rank
## Total     3326899
## Real      3382919    4
## Imaginary  -56020    3
##
## Results have 8 points, 2 axes
##
## Eigenvalues:
## [1] 2240139 1131446   11084     250     -46   -1652  -54323
##
## Weights: Constant
```

2.3 pco( )

The pco( ) function is part of the ecodist package.

```
Cities_Dist_2 <- dist(Cities_Matrix, method = "euclidean")
mds_5 <- pcoscaled(Cities_Dist_2)
mds_5

##         C1       C2        C3       C4       C5        C6
## 1  1.52920 -0.35825 -0.256972  0.19740 -0.09499 -0.021392
## 2  0.02126  0.66126  0.101808 -0.13359 -0.03568 -0.161277
## 3 -0.88923 -0.08258 -0.464711  0.13622  0.15108 -0.023249
## 4 -0.79715  0.11473  0.023283  0.17561 -0.06903  0.024587
## 5 -0.32769 -0.94880 -0.001369 -0.34733 -0.03579 -0.006023
## 6 -0.61484  0.06146  0.295618  0.12420 -0.14244  0.056848
## 7  0.56050 -0.17214  0.489583  0.08539  0.20031  0.010591
## 8  0.51794  0.72432 -0.187241 -0.23791  0.02655  0.119915
```

```
##          C7
## 1  0.001781
## 2  0.003845
## 3  0.035809
## 4 -0.097041
## 5 -0.008799
## 6  0.074377
## 7 -0.007239
## 8 -0.002733
```

2.4 pcoa( )

The pcoa( ) function is part of the ape package.

```
mds_6 <- pcoa(Cities_Dist, correction = "none", rn = NULL)
mds_6

## $correction
## [1] "none" "1"
##
## $note
## [1] "No correction was applied to the negative eigenvalues"
##
## $values
##   Eigenvalues Relative_eig Rel_corr_eig Broken_stick
## 1   2240138.7   0.67334138      0.61893      0.40833
## 2   1131445.5   0.34009015      0.31986      0.24167
## 3     11084.4   0.00333176      0.01764      0.15833
## 4       250.3   0.00007524      0.01472      0.10278
## 5         0.0   0.00000000      0.01464      0.06111
## 6       -46.0  -0.00001383      0.01421      0.02778
## 7     -1651.5  -0.00049641      0.00000      0.00000
## 8    -54322.6  -0.01632829      0.00000      0.00000
##   Cum_corr_eig Cumul_br_stick
## 1       0.6189         0.4083
## 2       0.9388         0.6500
## 3       0.9564         0.8083
## 4       0.9712         0.9111
## 5       0.9858         0.9722
## 6       1.0000         1.0000
## 7       1.0000         1.0000
## 8       1.0000         1.0000
##
## $vectors
##         Axis.1  Axis.2  Axis.3   Axis.4
## Athens 1011.09 -239.30 -16.817  1.32373
## Berlin   76.52  375.28 -77.598 -0.60503
## Dublin -714.65  183.71   6.244 -8.97632
## London -432.23  113.89   6.513  5.89108
## Madrid -406.71 -688.50 -26.547  0.03985
## Paris  -274.12  -27.98  31.981  9.33899
## Rome    368.46 -289.71  43.861 -6.75621
## Warsaw  371.64  572.61  32.364 -0.25610
```

```
##
## $trace
## [1] 3326899
##
## attr(,"class")
## [1] "pcoa"
```

2.5 dudi.pco( )

The dudi.pco( ) function is part of the ade4 package.

```
mds_7 <- dudi.pco(Cities_Dist, row.w = "uniform", scannf = FALSE,
    nf = 2, full = FALSE, tol = 0.0000001)

## Warning in dudi.pco(Cities_Dist, row.w = "uniform", scannf = FALSE, nf = 2, :  Non euclidean
distance

mds_7

## Duality diagramm
## class: pco dudi
## $call: dudi.pco(d = Cities_Dist, row.w = "uniform", scannf = FALSE,
##     nf = 2, full = FALSE, tol = 0.0000001)
##
## $nf: 2 axis-components saved
## $rank: 4
## eigen values: 280000 141400 1386 31.29
##   vector length mode    content
## 1 $cw    4      numeric column weights
## 2 $lw    8      numeric row weights
## 3 $eig   4      numeric eigen values
##
##   data.frame nrow ncol content
## 1 $tab       8    4    modified array
## 2 $li        8    2    row coordinates
## 3 $l1        8    2    row normed scores
## 4 $co        4    2    column coordinates
## 5 $c1        4    2    column normed scores
## other elements: NULL
```

2.6 smacofSym( )

The smacofSym( ) function is part of the smacof package.

```
mds_8 <- smacofSym(Cities_Dist, ndim = 2, verbose = FALSE, type = "interval")
mds_8

##
## Call:
## smacofSym(delta = Cities_Dist, ndim = 2, type = "interval", verbose = FALSE)
##
## Model: Symmetric SMACOF
## Number of objects: 8
```

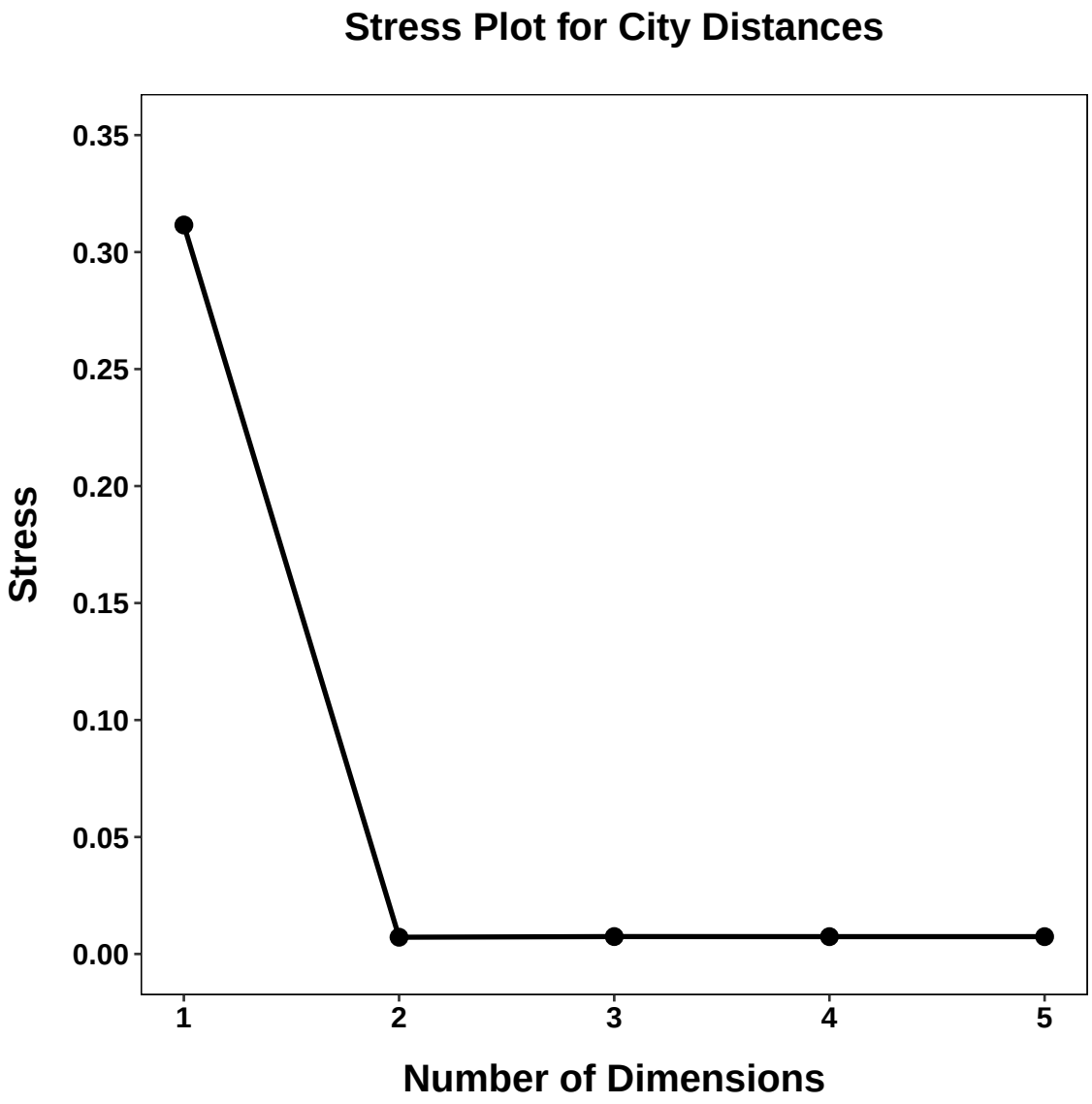
```
## Stress-1 value: 0.007
## Number of iterations: 5

mds_8b <- smacofSym(Cities_Dist, ndim = 2, verbose = FALSE, type = "ratio")
mds_8b

##
## Call:
## smacofSym(delta = Cities_Dist, ndim = 2, type = "ratio", verbose = FALSE)
##
## Model: Symmetric SMACOF
## Number of objects: 8
## Stress-1 value: 0.008
## Number of iterations: 4
```

2.6.1 Stress Plot

Stress plot for up to 4 dimensions. Here we run the smacofSym( ) function iteratively for 1 to 4 dimensions in order to get the stress values for plotting.

```
# Create a matrix to hold the stress values.
mds_stress <- matrix(NA, nrow = 5, ncol = 2)
for (i in 1:5) {
    mds_8 <- smacofSym(Cities_Dist, ndim = i, verbose = FALSE, type = "interval")
    mds_stress[i, 1] <- i
    mds_stress[i, 2] <- mds_8$stress
}
```

```
plot_data <- as.data.frame(mds_stress)
names(plot_data) <- c("D", "Stress")

ggplot(plot_data, aes(x = D, y = Stress)) + geom_point(shape = 19,
    size = 3, color = "black", na.rm = TRUE) + geom_line(size = 1) +
    scale_y_continuous(breaks = c(seq(0, 0.35, 0.05))) + scale_x_continuous(breaks = c(seq(1,
    5, 1))) + coord_cartesian(xlim = c(1, 5), ylim = c(0, 0.35)) +
    xlab("Number of Dimensions") + ylab("Stress") + theme(text = element_text(size = 14,
    family = "sans", color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 0), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Stress Plot for City Distances")
```



2.6.2 Shepard Plot

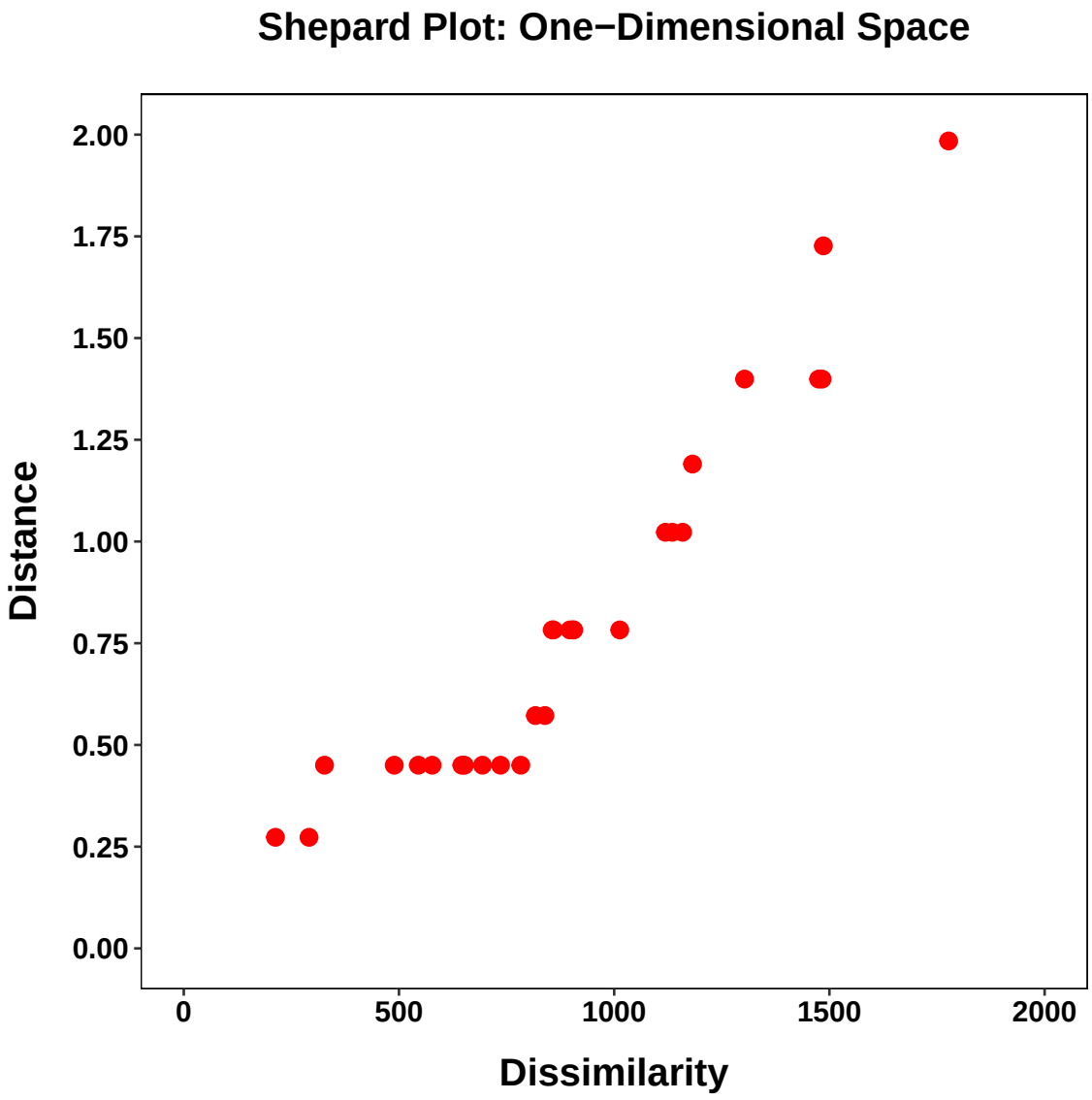
A Shepard plot pairs the actual distances (called disparities) with the distances estimated from the MDS solution. An excellent fit should produce a nearly diagonal line. Shepard plots for up to 4 dimensions.

```
mds_8 <- smacofSym(Cities_Dist, ndim = 1, verbose = FALSE, type = "interval")
Cities_1_fits <- Shepard(Cities_Dist, mds_8$conf)
mds_8 <- smacofSym(Cities_Dist, ndim = 2, verbose = FALSE, type = "interval")
Cities_2_fits <- Shepard(Cities_Dist, mds_8$conf)
mds_8 <- smacofSym(Cities_Dist, ndim = 3, verbose = FALSE, type = "interval")
Cities_3_fits <- Shepard(Cities_Dist, mds_8$conf)
mds_8 <- smacofSym(Cities_Dist, ndim = 4, verbose = FALSE, type = "interval")
Cities_4_fits <- Shepard(Cities_Dist, mds_8$conf)
```

```
mds_8$conf
```

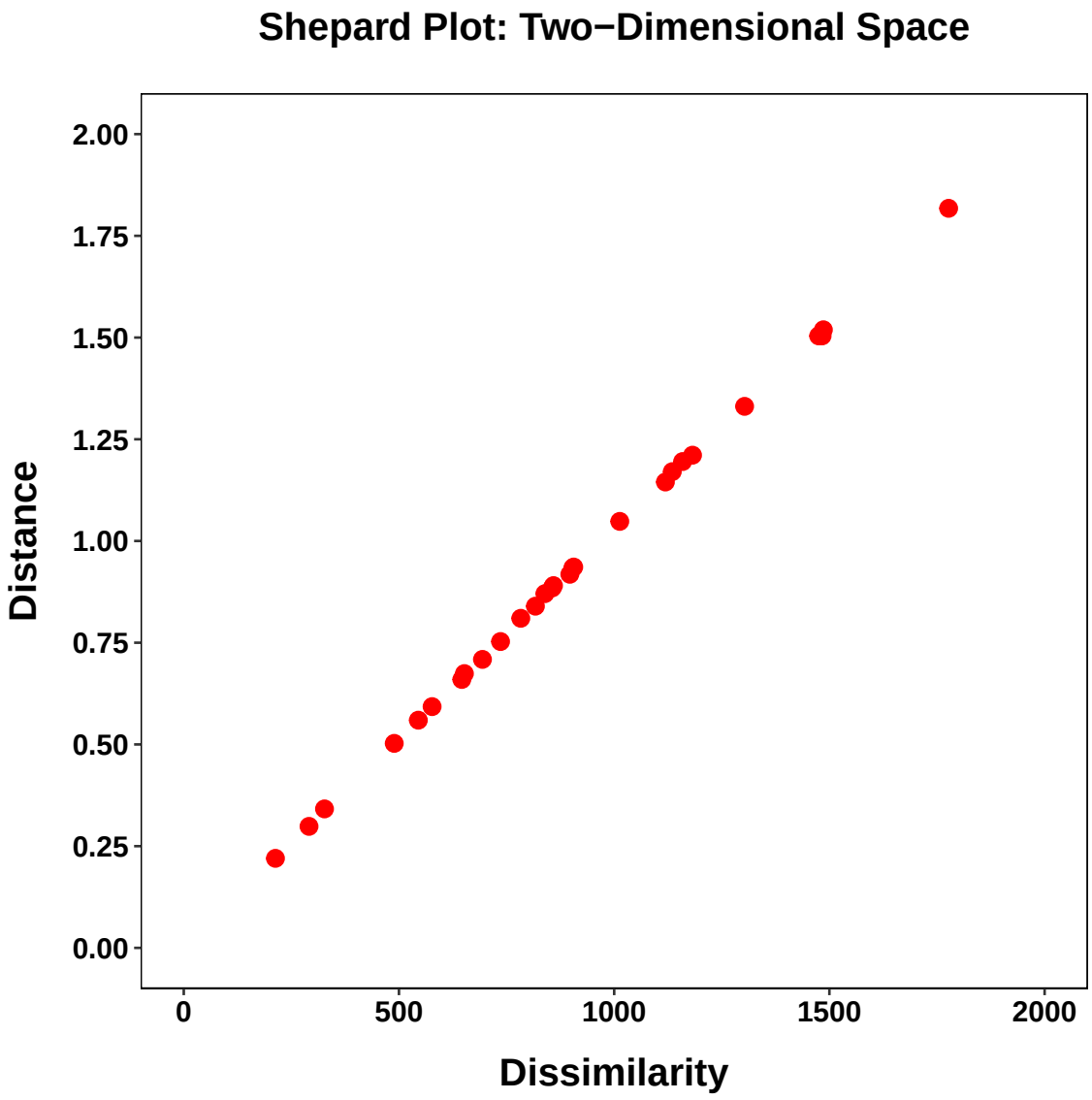
```
##              D1       D2        D3          D4
## Athens  1.03277 -0.24182 -0.017088  0.00135945
## Berlin  0.08354  0.39023 -0.076468 -0.00061133
## Dublin -0.73174  0.18724  0.006095 -0.00937435
## London -0.44127  0.11654  0.006192  0.00614096
## Madrid -0.41294 -0.69342 -0.026475  0.00006271
## Paris  -0.27868 -0.03252  0.032850  0.00967495
## Rome    0.37723 -0.29783  0.045115 -0.00699100
## Warsaw  0.37109  0.57158  0.029780 -0.00026139
```

```
plot_data <- as.data.frame(Cities_1_fits)

ggplot(plot_data, aes(x = x, y = yf)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(0,
    2, 0.25))) + scale_x_continuous(breaks = c(seq(0, 2000, 500))) +
    coord_cartesian(xlim = c(0, 2000), ylim = c(0, 2)) + xlab("Dissimilarity") +
    ylab("Distance") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 0), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Shepard Plot: One-Dimensional Space")
```



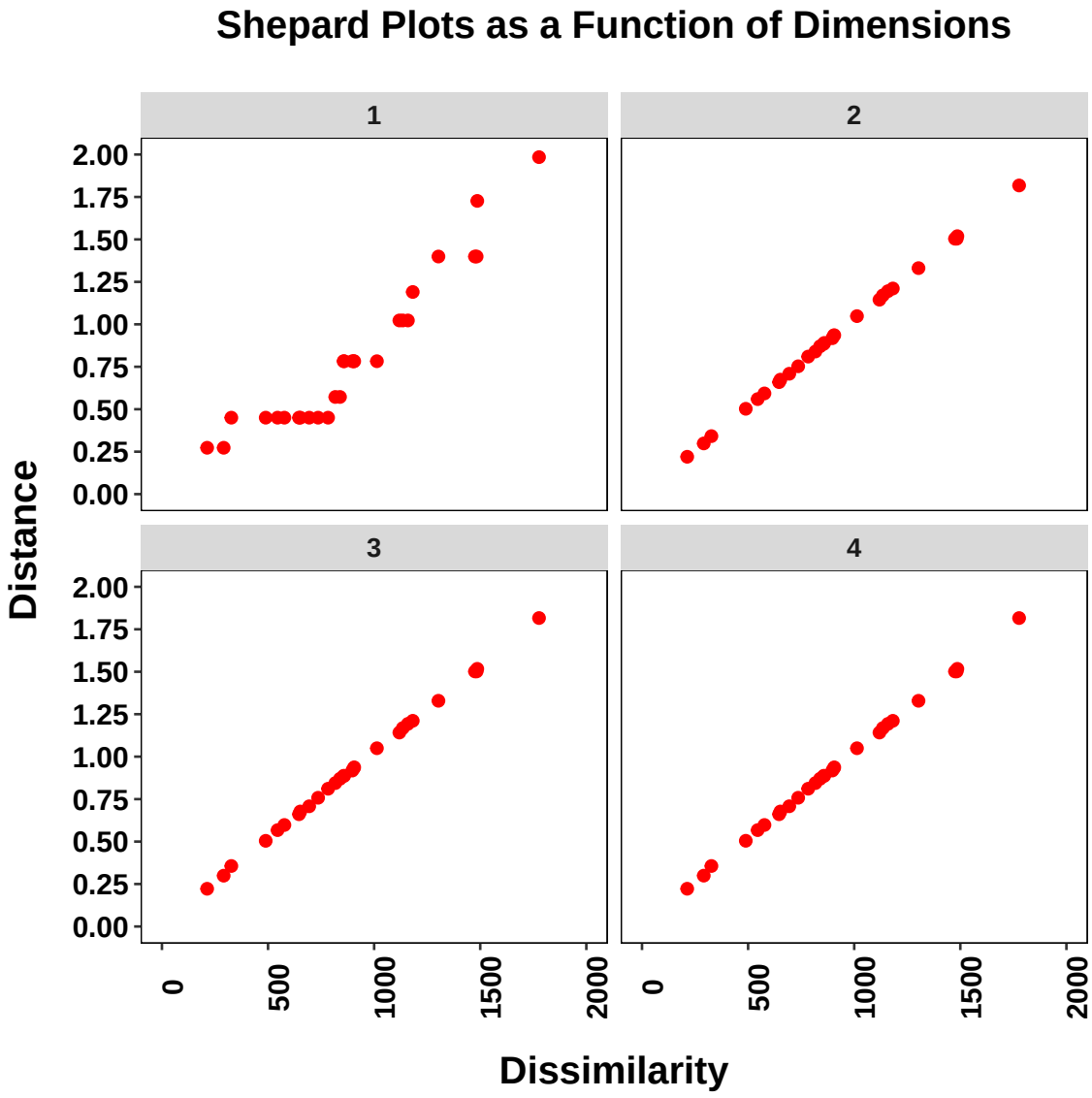

```
plot_data <- as.data.frame(Cities_2_fits)

ggplot(plot_data, aes(x = x, y = yf)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(0,
    2, 0.25))) + scale_x_continuous(breaks = c(seq(0, 2000, 500))) +
    coord_cartesian(xlim = c(0, 2000), ylim = c(0, 2)) + xlab("Dissimilarity") +
    ylab("Distance") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 0), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
```

```
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Shepard Plot: Two-Dimensional Space")
```




```
plot_data <- as.data.frame(Cities_1_fits)
plot_data <- rbind(plot_data, Cities_2_fits)
plot_data <- rbind(plot_data, Cities_3_fits)
plot_data <- rbind(plot_data, Cities_4_fits)
plot_data$D <- c(rep(1, 28), rep(2, 28), rep(3, 28), rep(4, 28))

p <- ggplot(plot_data, aes(x = x, y = yf)) + geom_point(shape = 19,
    size = 2, color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(0,
```

```
    2, 0.25))) + scale_x_continuous(breaks = c(seq(0, 2000, 500))) +
    coord_cartesian(xlim = c(0, 2000), ylim = c(0, 2)) + xlab("Dissimilarity") +
    ylab("Distance") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Shepard Plots as a Function of Dimensions")
p + facet_wrap(~D, nrow = 2)
```



3 Non-Metric Methods

The car rank data can be used to demonstrate non-metric methods. In non-metric MDS, the lower dimensional solution only needs to preserve rank order. The smacofSym( ) can be used for non-metric MDS, but we need to specifically request this in the function call.

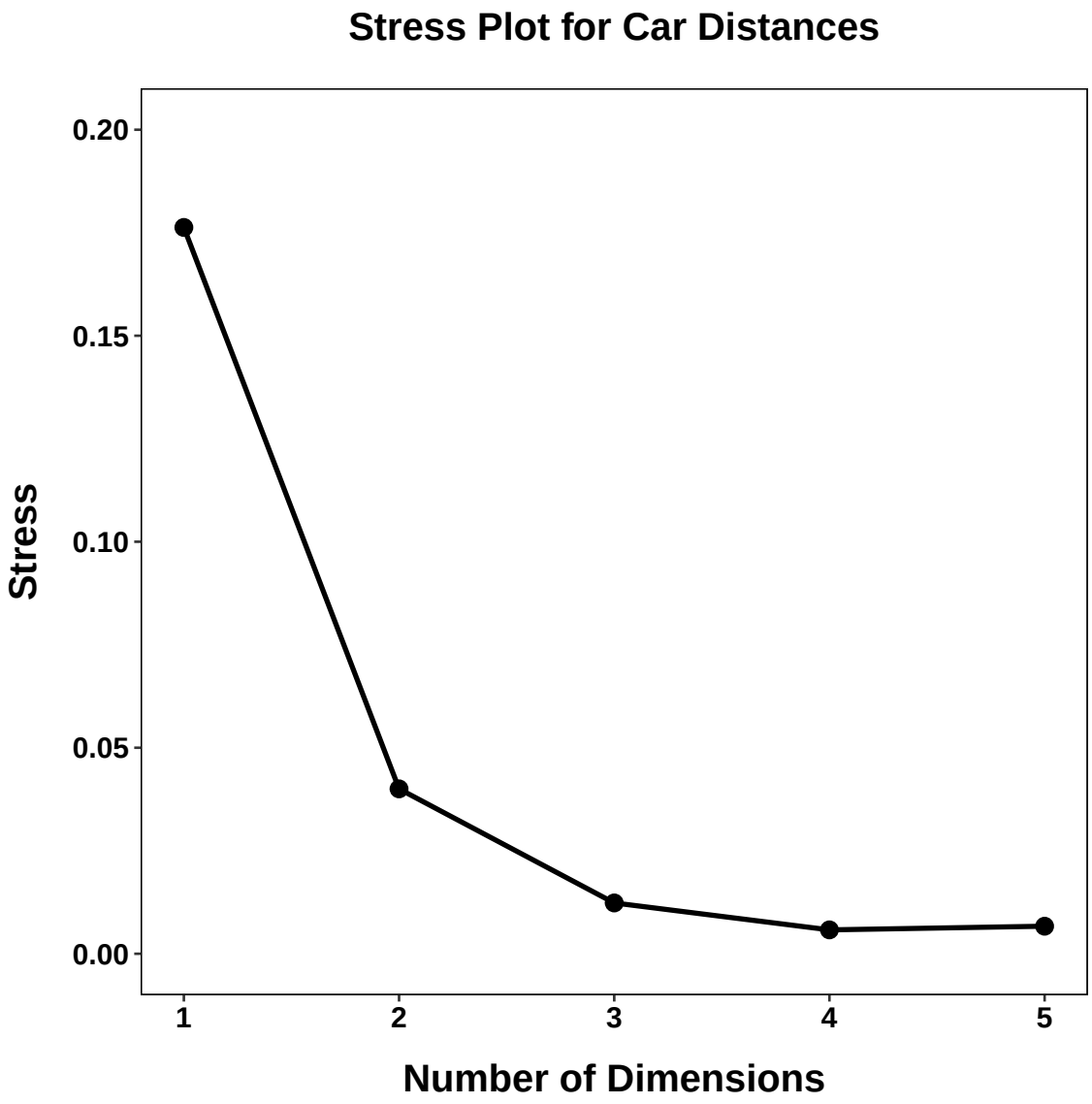
3.0.3 Stress Plot

Stress plot for up to 4 dimensions. The stress plot can help us determine the best number of dimensions to describing the data.

```
mds_stress <- matrix(NA, nrow = 5, ncol = 2)
for (i in 1:5) {
```

```
    mds_9 <- smacofSym(Cars_Dist, ndim = i, verbose = FALSE, type = "ordinal")
    mds_stress[i, 1] <- i
    mds_stress[i, 2] <- mds_9$stress
}
```

```
plot_data <- as.data.frame(mds_stress)
names(plot_data) <- c("D", "Stress")

ggplot(plot_data, aes(x = D, y = Stress)) + geom_point(shape = 19,
    size = 3, color = "black", na.rm = TRUE) + geom_line(size = 1) +
    scale_y_continuous(breaks = c(seq(0, 0.2, 0.05))) + scale_x_continuous(breaks = c(seq(1,
    5, 1))) + coord_cartesian(xlim = c(1, 5), ylim = c(0, 0.2)) +
    xlab("Number of Dimensions") + ylab("Stress") + theme(text = element_text(size = 14,
    family = "sans", color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 0), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Stress Plot for Car Distances")
```

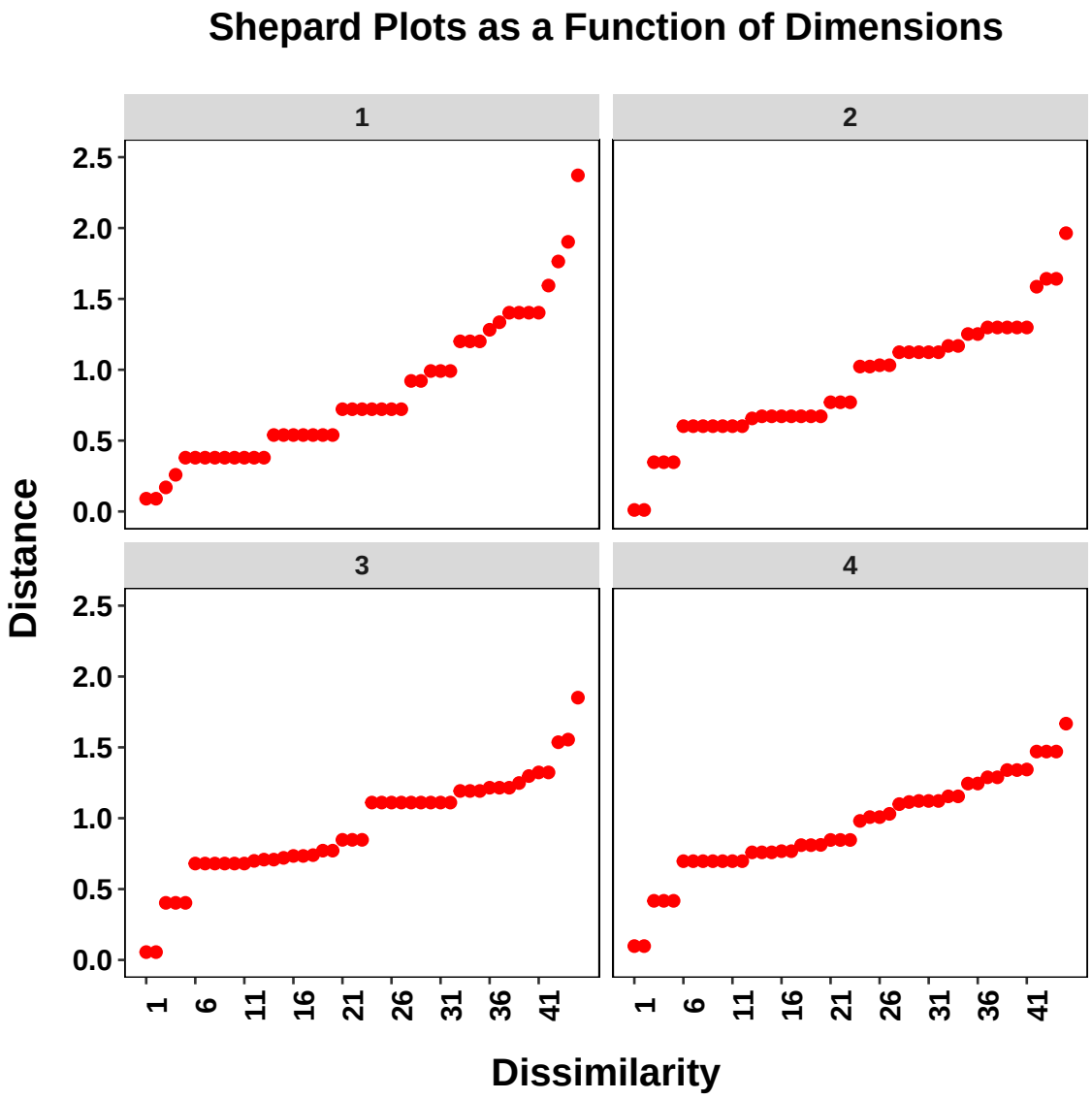


3.1 Shepard Plots

Shepard plots for up to 4 dimensions.

```
mds_9 <- smacofSym(Cars_Dist, ndim = 1, verbose = FALSE, type = "ordinal")
Cars_1_fits <- Shepard(Cars_Dist, mds_9$conf)
mds_9 <- smacofSym(Cars_Dist, ndim = 2, verbose = FALSE, type = "ordinal")
Cars_2_fits <- Shepard(Cars_Dist, mds_9$conf)
mds_9 <- smacofSym(Cars_Dist, ndim = 3, verbose = FALSE, type = "ordinal")
Cars_3_fits <- Shepard(Cars_Dist, mds_9$conf)
mds_9 <- smacofSym(Cars_Dist, ndim = 4, verbose = FALSE, type = "ordinal")
Cars_4_fits <- Shepard(Cars_Dist, mds_9$conf)
```

```
plot_data <- as.data.frame(Cars_1_fits)
plot_data <- rbind(plot_data, Cars_2_fits)
plot_data <- rbind(plot_data, Cars_3_fits)
plot_data <- rbind(plot_data, Cars_4_fits)
plot_data$D <- c(rep(1, 45), rep(2, 45), rep(3, 45), rep(4, 45))

p <- ggplot(plot_data, aes(x = x, y = yf)) + geom_point(shape = 19,
    size = 2, color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(0,
    2.5, 0.5))) + scale_x_continuous(breaks = c(seq(1, 45, 5))) +
    coord_cartesian(xlim = c(1, 45), ylim = c(0, 2.5)) + xlab("Dissimilarity") +
    ylab("Distance") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Shepard Plots as a Function of Dimensions")
p + facet_wrap(~D, nrow = 2)
```

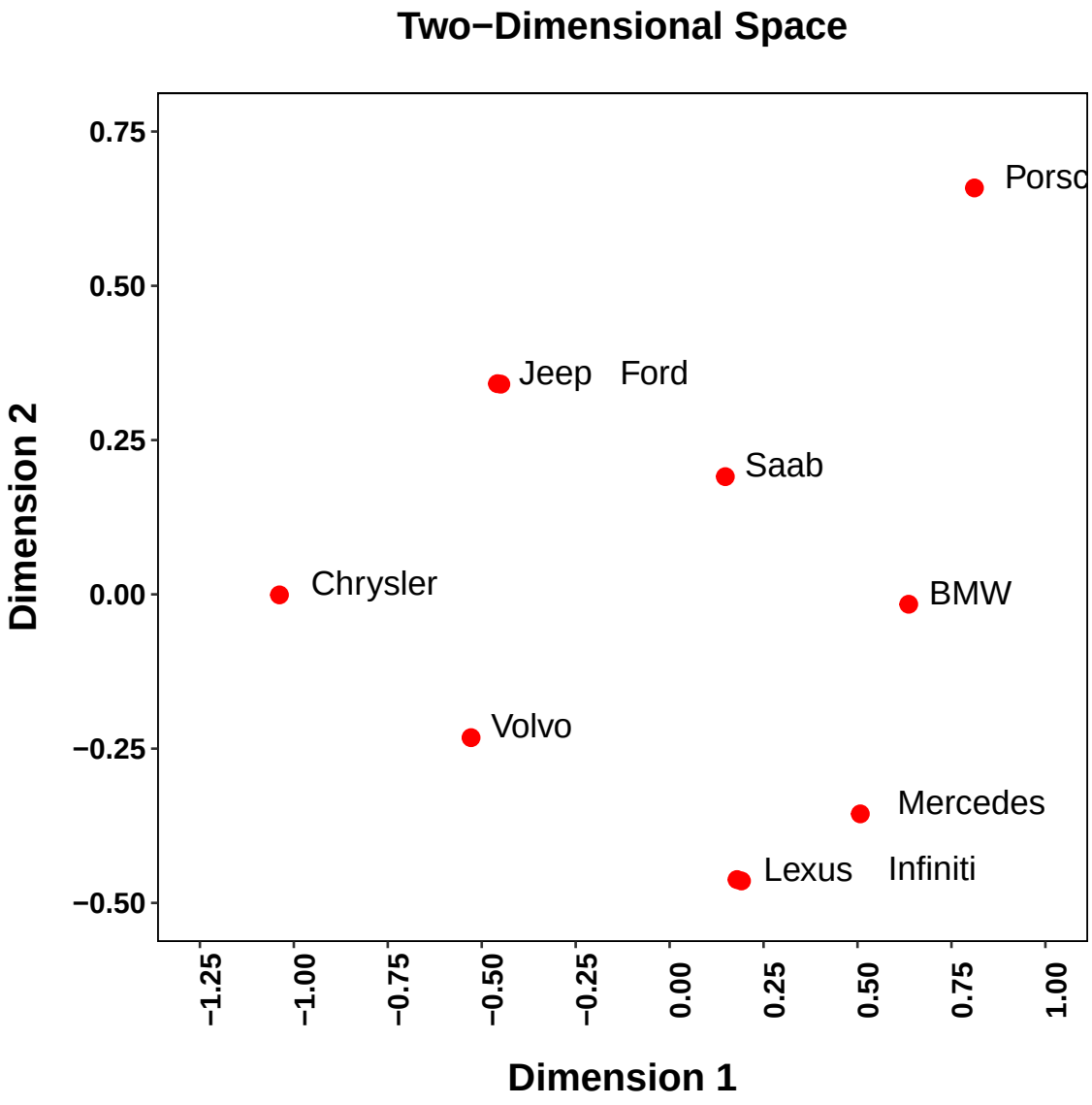


3.2 Scatterplots

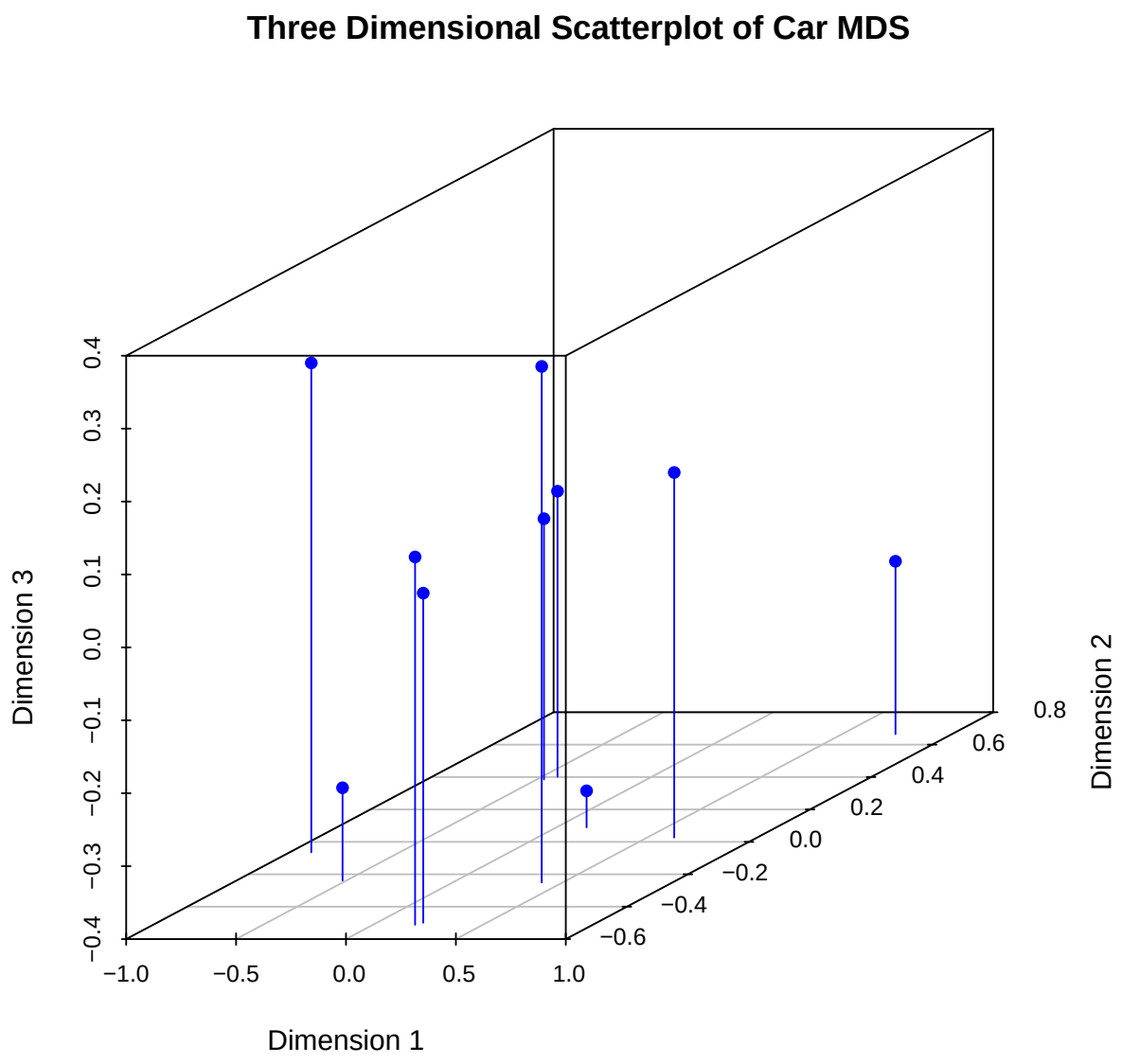
```
mds_9 <- smacofSym(Cars_Dist, ndim = 2, verbose = FALSE, type = "ordinal")
```

```
plot_data <- as.data.frame(mds_9$conf)
names(plot_data) <- c("D1", "D2")
plot_data$Name <- c("BMW", "          Ford", "           Infiniti",
    "Jeep", "Lexus", "Chrysler", "Mercedes", "Saab", "Porsche", "Volvo")

ggplot(plot_data, aes(x = D1, y = D2)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-0.5,
    0.75, 0.25))) + scale_x_continuous(breaks = c(seq(-1.25, 1, 0.25))) +
```

```
    geom_text(aes(label = Name), hjust = -0.25, vjust = 0, size = 5) +
    coord_cartesian(xlim = c(-1.25, 1), ylim = c(-0.5, 0.75)) + xlab("Dimension 1") +
    ylab("Dimension 2") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Two-Dimensional Space")
```

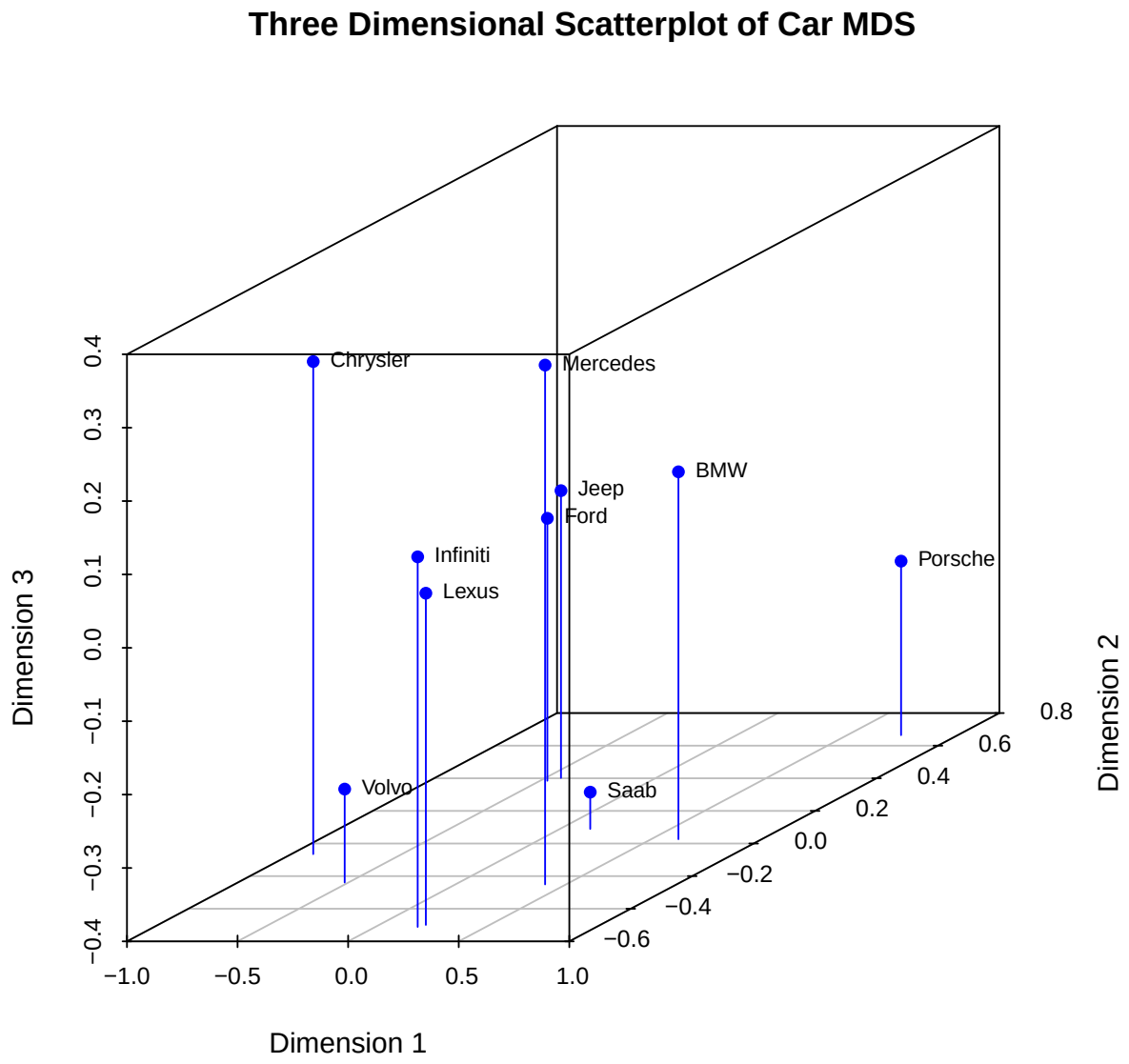


```r
mds_9 <- smacofSym(Cars_Dist, ndim = 3, verbose = FALSE, type = "ordinal")

# Save the MDS coordinates as a dataframe.
mds_points <- as.data.frame(mds_9$conf)

scatterplot3d(mds_points$D1, mds_points$D2, mds_points$D3, color = "blue",
    pch = 16, type = "h", main = "Three Dimensional Scatterplot of Car MDS",
    xlab = "Dimension 1", ylab = "Dimension 2", zlab = "Dimension 3")
```




```r
with(mds_points, {
    s3d <- scatterplot3d(mds_points$D1, mds_points$D2, mds_points$D3,
        color = "blue", pch = 16, type = "h", main = "Three Dimensional Scatterplot of Car MDS",
        xlab = "Dimension 1", ylab = "Dimension 2", zlab = "Dimension 3")
```

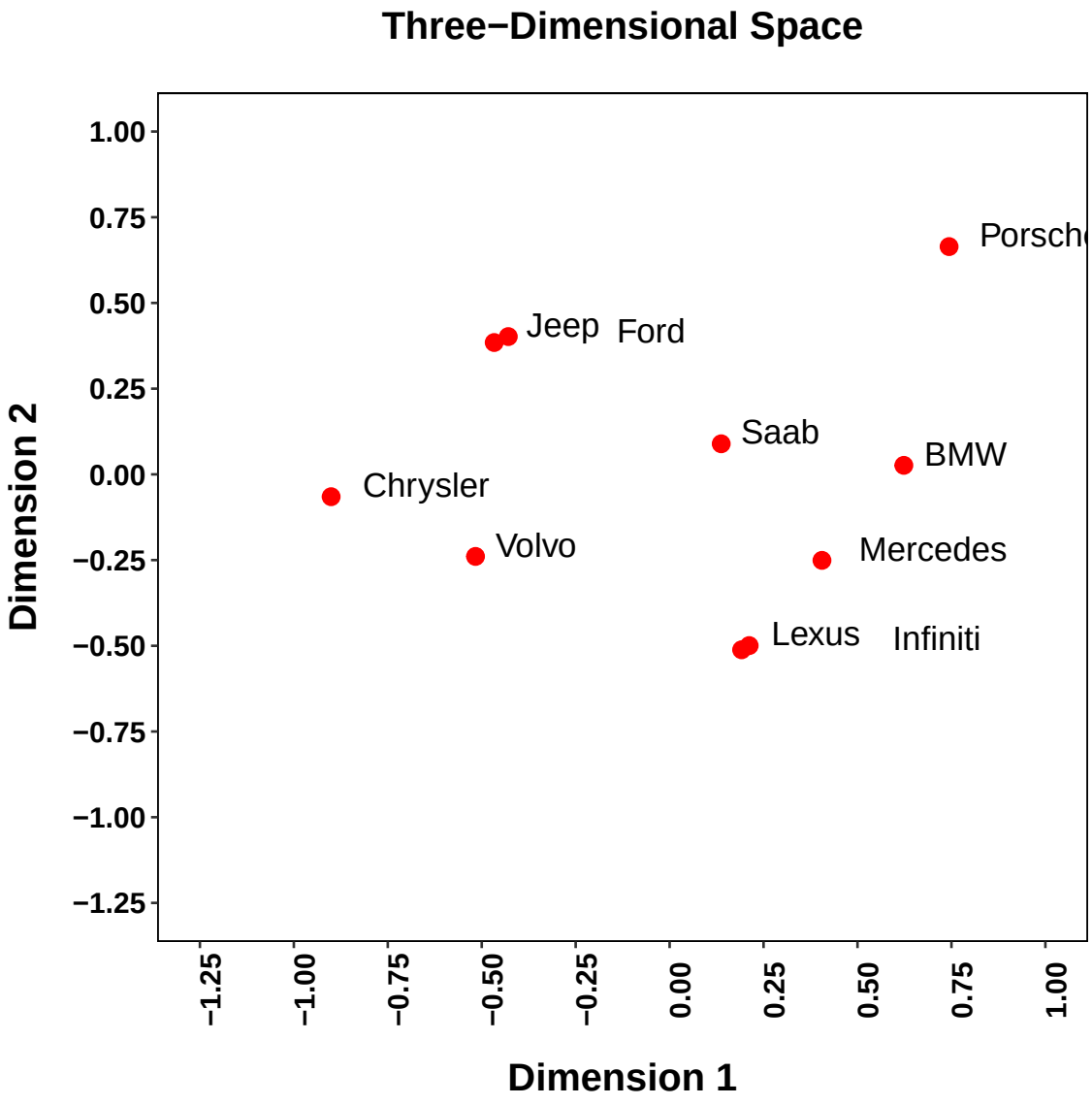
```
    s3d.coords <- s3d$xyz.convert(mds_points$D1, mds_points$D2, mds_points$D3)
    text(s3d.coords$x, s3d.coords$y, labels = row.names(mds_points),
        cex = 0.75, pos = 4)
})
```




```
plot_data <- as.data.frame(mds_9$conf)
names(plot_data) <- c("D1", "D2", "D3")
plot_data$Name <- c("BMW", "          Ford", "           Infiniti",
    "Jeep", "Lexus", "Chrysler", "Mercedes", "Saab", "Porsche", "Volvo")
```

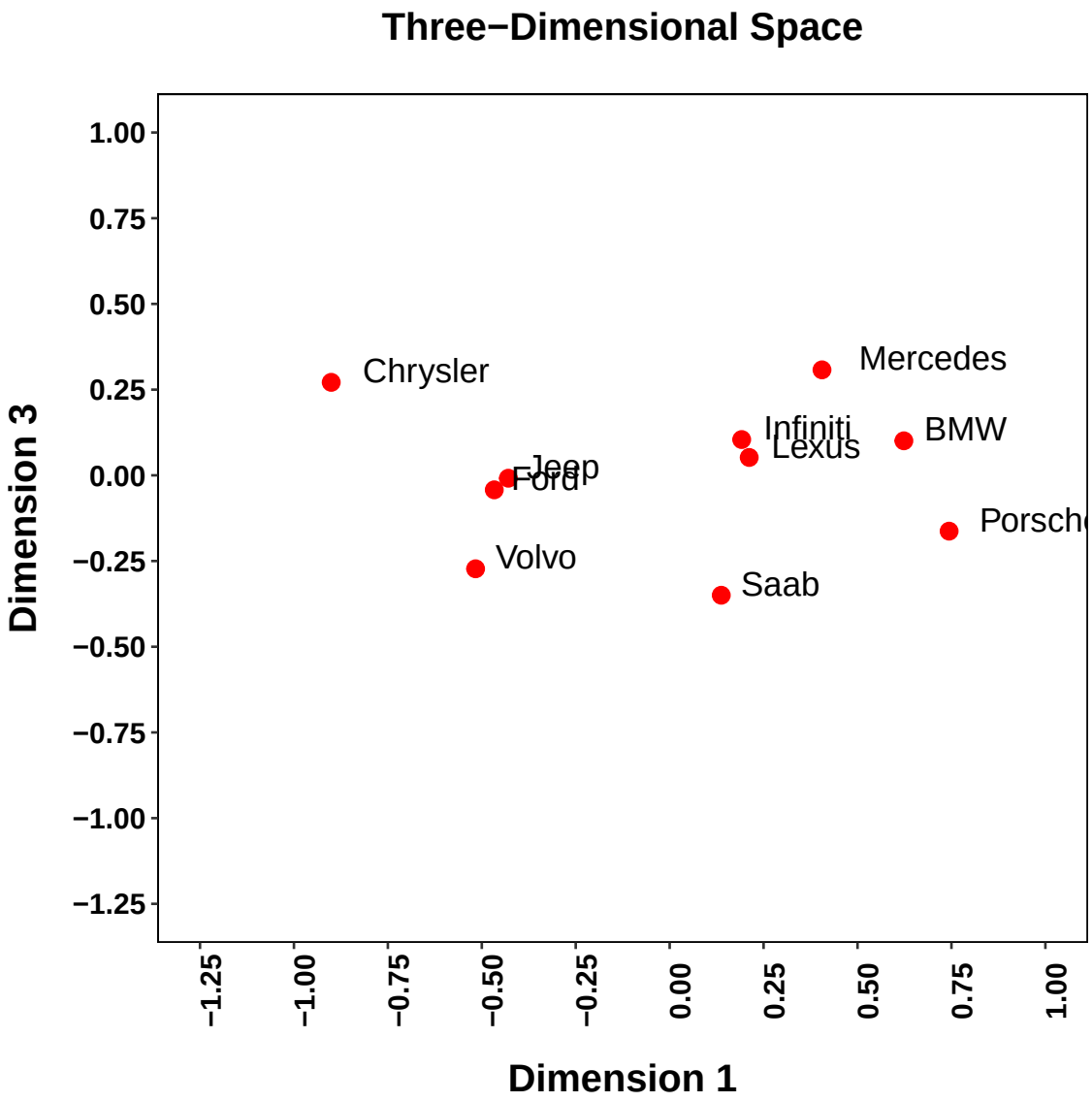
```
ggplot(plot_data, aes(x = D1, y = D2)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-1.25,
    1, 0.25))) + scale_x_continuous(breaks = c(seq(-1.25, 1, 0.25))) +
```

```
    geom_text(aes(label = Name), hjust = -0.25, vjust = 0, size = 5) +
    coord_cartesian(xlim = c(-1.25, 1), ylim = c(-1.25, 1)) + xlab("Dimension 1") +
    ylab("Dimension 2") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Three-Dimensional Space")
```



```
plot_data <- as.data.frame(mds_9$conf)
names(plot_data) <- c("D1", "D2", "D3")
plot_data$Name <- c("BMW", "Ford", "Infiniti", "Jeep", "Lexus", "Chrysler",
    "Mercedes", "Saab", "Porsche", "Volvo")
```

```
ggplot(plot_data, aes(x = D1, y = D3)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-1.25,
    1, 0.25))) + scale_x_continuous(breaks = c(seq(-1.25, 1, 0.25))) +
    geom_text(aes(label = Name), hjust = -0.25, vjust = 0, size = 5) +
    coord_cartesian(xlim = c(-1.25, 1), ylim = c(-1.25, 1)) + xlab("Dimension 1") +
    ylab("Dimension 3") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Three-Dimensional Space")
```




```
plot_data <- as.data.frame(mds_9$conf)
names(plot_data) <- c("D1", "D2", "D3")
plot_data$Name <- c("BMW", "Ford", "Infiniti", "Jeep", "Lexus", "Chrysler",
    "Mercedes", "Saab", "Porsche", "Volvo")
```

```
ggplot(plot_data, aes(x = D2, y = D3)) + geom_point(shape = 19, size = 3,
    color = "red", na.rm = TRUE) + scale_y_continuous(breaks = c(seq(-1.25,
    1, 0.25))) + scale_x_continuous(breaks = c(seq(-1.25, 1, 0.25))) +
    geom_text(aes(label = Name), hjust = -0.25, vjust = 0, size = 5) +
    coord_cartesian(xlim = c(-1.25, 1), ylim = c(-1.25, 1)) + xlab("Dimension 2") +
    ylab("Dimension 3") + theme(text = element_text(size = 14, family = "sans",
    color = "black", face = "bold"), axis.text.y = element_text(colour = "black",
    size = 12, face = "bold"), axis.text.x = element_text(colour = "black",
```

```
    size = 12, face = "bold", angle = 90), axis.title.x = element_text(margin = margin(15,
    0, 0, 0), size = 16), axis.title.y = element_text(margin = margin(0,
    15, 0, 0), size = 16), axis.line.x = element_blank(), axis.line.y = element_blank(),
    plot.title = element_text(size = 16, face = "bold", margin = margin(0,
        0, 20, 0), hjust = 0.5), panel.background = element_rect(fill = "white",
        linetype = 1, color = "black"), panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(), plot.background = element_rect(fill = "white"),
    plot.margin = unit(c(1, 1, 1, 1), "cm"), legend.position = "bottom",
    legend.title = element_blank()) + ggtitle("Three-Dimensional Space")
```