Multidimensional Scaling

Ryan Mouw

March 11, 2021

1 Introduction

Multi-dimensional scaling (MDS) is used to create low-dimensional models of a set of objects, while attempting to maintain the distance relationship between each object as much as possible. MDS takes pairs of objects and creates a set of points in low-dimensional Euclidean space, with each point representing one of the objects in the set in such a way that the distance between the points in the model are as similar as possible to the corresponding objects in the set.

In this this paper we will create two MDS models and evaluate their accuracy and utility. The utility of any model is a complex thing to adress, so we will use a number of tools to evaluate the MDS models, such as Goodness of Fit (GOF) measurements, absolute error, eigenvalues, and a few different graphs.

2 Multi-Dimensional Scale Models

2.1 Matrix A (Tokyo Area)

We will create an MDS model of non-Euclidean distances between cities/ locations. In other words, we will measure the **air** distance between cities/locations on the surface of the Earth. We will do this with a set of relatively small distances (<300km) and a very large set that spans the globe. Then we will create MDS models (Models that use Euclidean distance) and evaluate how well the models accurately reflect the objects they are meant to represent.

Below is an image showing 6 cities/locations (Tokyo, Kofu, Asahi, Mt. Fuji, Yokohama, Chiba) around Tokyo and the non-Euclidean air distances between them.

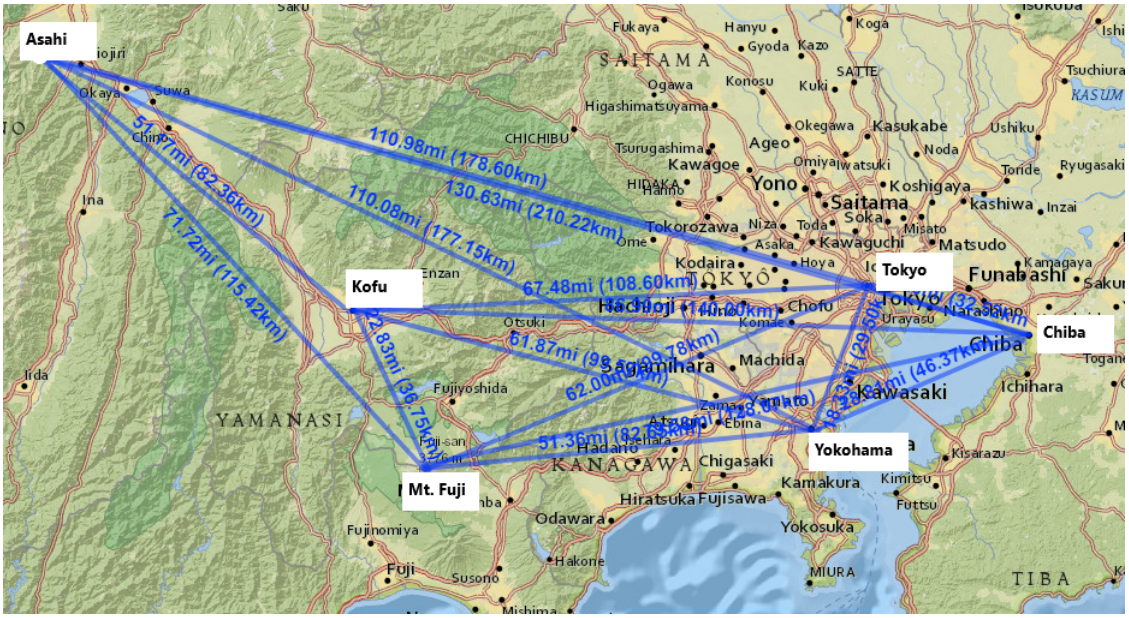


Figure 1: Greater Tokyo area (Local distances)

The non-Euclidean air distance between 6 locations in the greater Tokyo area is given in Matrix A. A_{ij} is the distance in kilometers between city/location i and city/location j.

	Tokyo	Kofu	Asahi	Mt Fuji	Yokohama	Chiba
Tokyo	0	109	179	100	29	32
Kofu	109	0	82	37	100	140
Asahi	179	82	0	115	177	210
Mt Fuji	100	37	115	0	83	128
Yokohama	29	100	177	83	0	46
Chiba	32	140	210	128	46	0

Figure 2: Matrix A (Local distances)

2.2 Matrix B (World Map)

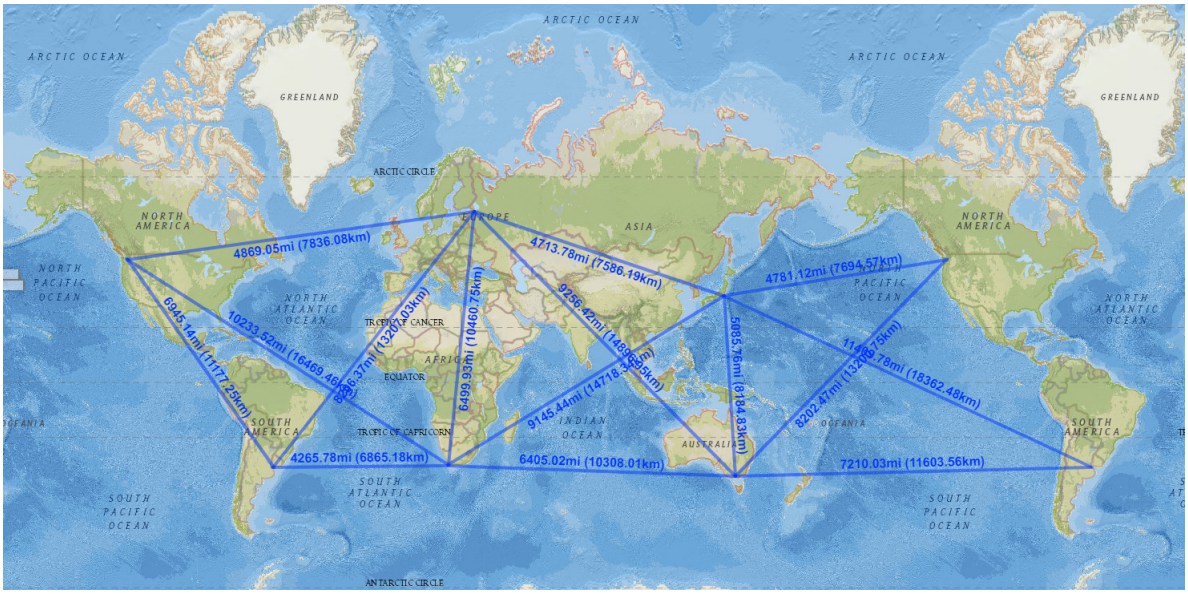


Figure 3: Continuous World Map with City Distances

The non-Euclidean air distance between 6 cities (Tokyo, Melbourne, Seattle, Buenos Aires, St. Petersburg, and Cape Town) around the globe is given in Matrix B. B_{ij} is the distance in kilometers between city i and city j.

	Tokyo	St Petersburg	Seattle	Cape Town	Buenos Aires	Melbourne
Tokyo	0	7586	7695	14718	18362	8185
St Petersburg	7586	0	7799	10461	13207	14897
Seattle	7695	7799	0	16469	11177	13200
Cape Town	14718	10461	16469	0	6865	10308
Buenos Aires	18362	13207	11177	6865	0	11603
Melbourne	8185	14897	13200	10308	11603	0

Figure 4: Matrix B (International distances)

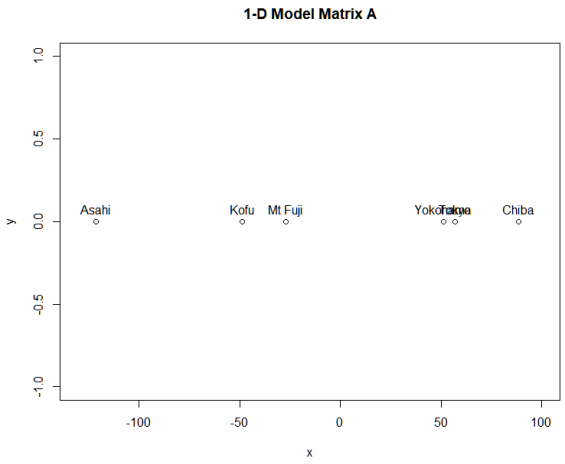
3 Model A, Model B, and cmdscale()

To create the MDS model we use the **cmdscale()** in R statistical programming language. cmdscale() outputs a set of coordinates representing the set of objects in whichever dimension specified in the function. For our purposes, $Model_{Ai}$ will represent the set of coordinates given from Matrix A, in the i^{th} dimension, and similarly with $Model_{Bi}$ and Matrix B.

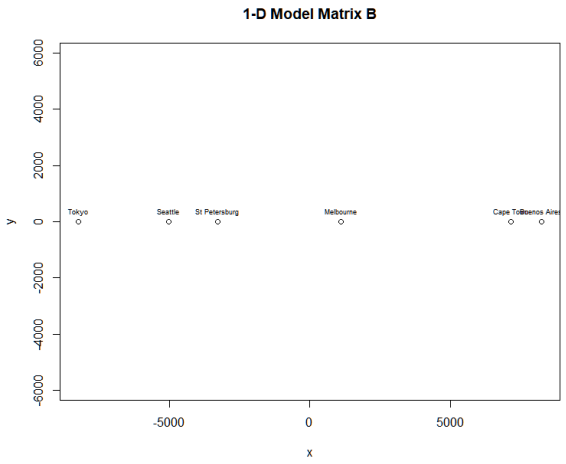
4 MDS Plots

Below are the MDS models of Matrix A ($Model_{A1}$) and Matrix B ($Model_{B1}$)

4.1 1-D Model Plot



(a) $Model_{A1}$

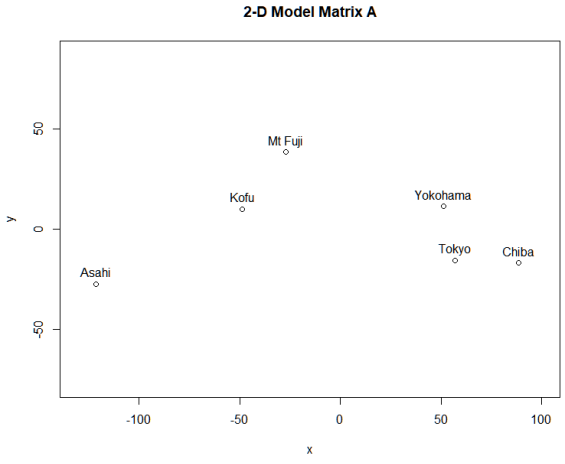


(b) $Model_{B1}$

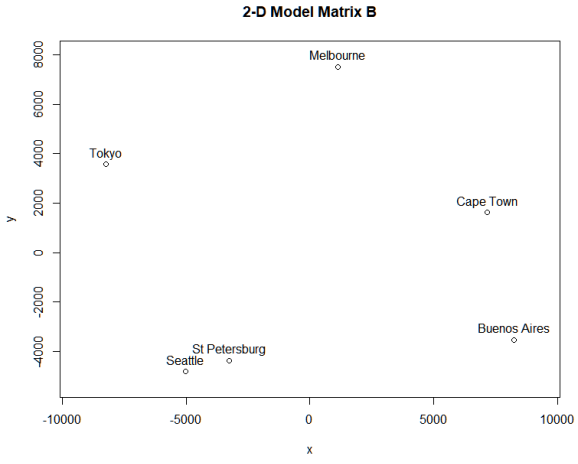
Like Matrix A, the 1-D model shows the farthest cities are Chiba and Asahi. The closest cities are Tokyo and Yokohama.

The 1-D model for Matrix B shows the closest cities to be Cape Town and Buenos Aires, which is correct, but makes some obvious errors by placing Seattle closer to St. Petersburg than Tokyo, which is incorrect. The 1-D model accurately shows that Buenos Aires and Tokyo are the farthest cities apart.

4.2 2-D Model Plot



(a) $Model_{A2}$

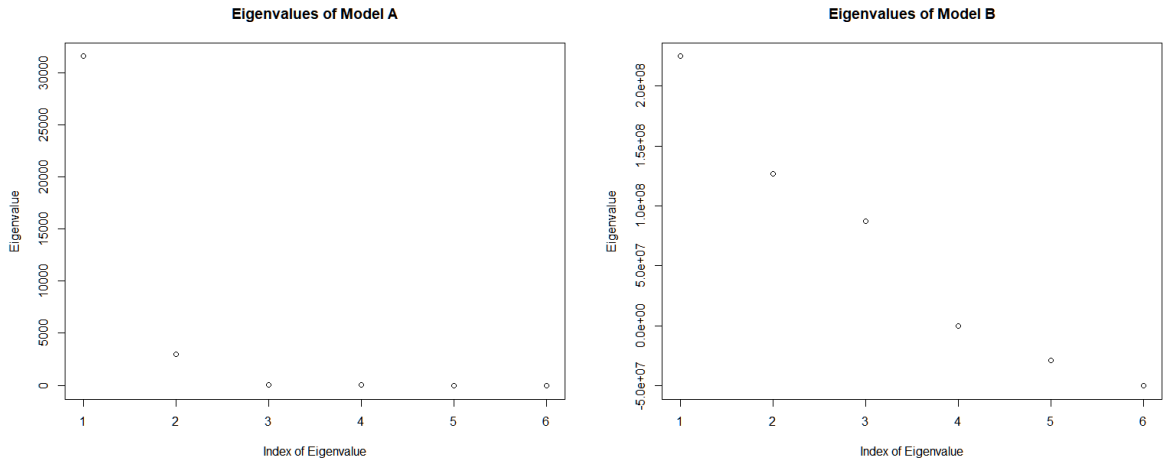


(b) $Model_{B2}$

It can be difficult to interpret how well the models capture the distance relationships between the objects. This is especially true in the 1-D model. To interpret how well the models works, we will first look to the Eigenvalues.

5 Eigenvalues

One way to glean information from the MDS process is to look at the eigenvalues of the distance matrix. Below are the plots of the eigenvalues from the distance matrix associated with $Model_A$ and $Model_B$.



(a) Eigenvalues of Model A

(b) Eigenvalues of Model B

The eigenvalues of Model A and Model B are listed below.

Model A) 3.154584e+04, 2.971829e+03, 6.152902e+01, 5.620774e+01, 3.304024e-12, -4.823948e+01

Model B) 2.247291e+08, 1.267970e+08, 8.741936e+07, -8.195639e-08, -2.847802e+07, -5.004324e+07

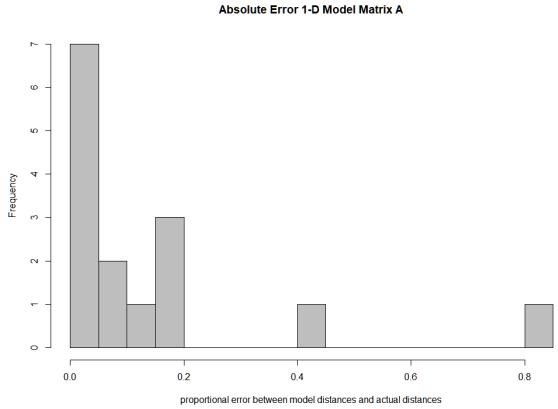
The first two eigenvalues of $Model_A$ are high relative to the rest, with the first being much higher than the second. This suggests that a 1-dimensional model will capture the object distance relationship well, but the addition of the second dimension will fit the data even better. The last eigenvalue of $Model_A$ is negative. This is an indication of non-Euclidean distances in the original data.

The set of Eigenvalues for $Model_B$ decrease at steadier rate compared to $Model_A$. Although, there is still a sizable gap between the eigenvalues for 3-D and 4-D models. Also, the Eigenvalues are negative after the third dimension. This would indicate that a 3D model would capture some aspects of the original data.

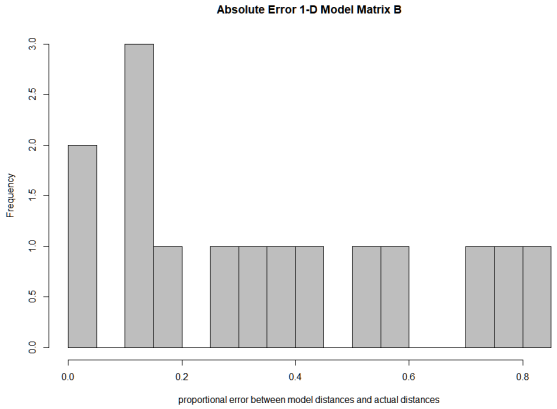
6 Absolute Error

Below are histograms showing the absolute error between the distance between the locations in Matrix A and Matrix B and their corresponding models.

6.1 1-D



(a) Absolute Error $Model_{A1}$



(b) Absolute Error $Model_{B1}$

The max error of $Model_{A1}$ is 23.24224 km and is between Yokohama and Tokyo.

The mean error of $Model_{A1}$ is 8.157484 km.

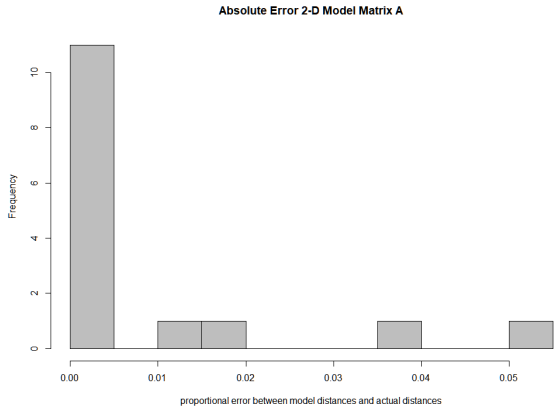
$Model_{A1}$ has mostly small error, with a single distance being almost completely misrepresented with 80% error. This agrees with the eigenvalue analysis of $Model_{A1}$ that indicated the 1-D model of Matrix A captures the distance relationship well.

The max error of $Model_{B1}$ is 10502.23 km and is between Buenos Aires and Cape Town.

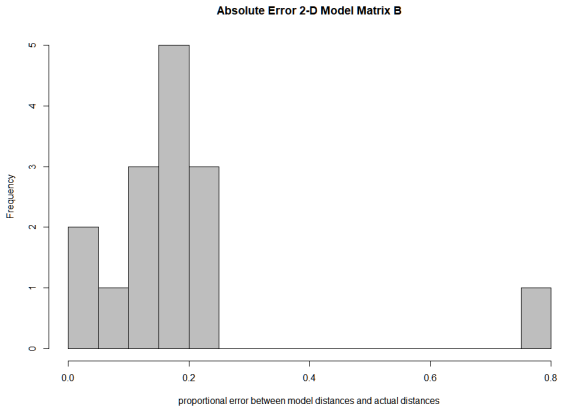
The mean error of $Model_{B1}$ is 3800.804 km.

$Model_{B1}$ has a more uniform spread of error, with three distances having nearly 80% error.

6.2 2-D



(a) Absolute Error $Model_{A2}$



(b) Absolute Error $Model_{B2}$

The max error of $Model_{A2}$ is 1.524695 km is between Yokohama and Tokyo.

The mean error of $Model_{A2}$ is 0.3660635 km.

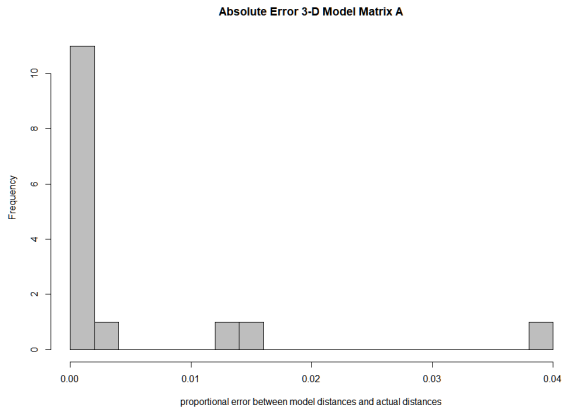
The 2-D model of Matrix A drastically lowered the max and mean error. This agrees with the information from the eigenvalue graph.

The max error of $Model_{B2}$ is 5985.221 km and is between Seattle and St. Petersburg, Russia.

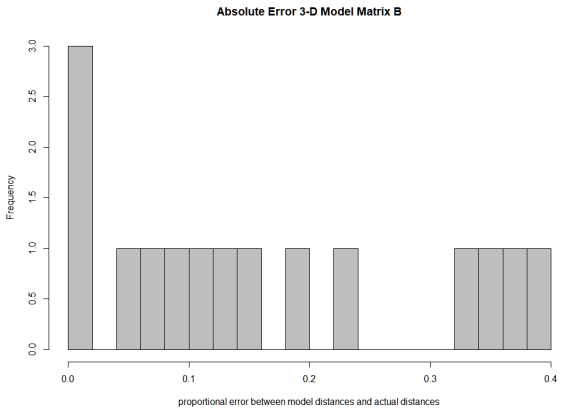
The mean error of $Model_{B2}$ is 1875.478 km.

The max and mean error difference between $Model_{B1}$ and $Model_{B2}$ was not as dramatic as $Model_{A1}$ to $Model_{A2}$. The max and mean fell by roughly half. This was also seen in the eigenvalue graph.

6.3 3-D



(a) Absolute Error $Model_{A3}$



(b) Absolute Error $Model_{B3}$

The max error of $Model_{A3}$ is 1.142773 km and is between Tokyo and Yokohama.

The mean error of $Model_{A3}$ is 0.2101351 km.

The difference between $Model_{A3}$ and $Model_{A2}$ is much less than the jump from 1-D to 2-D. This indicates that Matrix A may be best suited for an MDS model of 2-D.

The max error of $Model_{B3}$ is 2864.534 km and is between Buenos Aires and Cape Town.

The mean error of $Model_{B3}$ is 1486.843 km.

The max and mean error again fell by roughly half. This is significant because the next dimension involves negative Eigenvalues and indicates non-Euclidean distances. This again agrees with the Eigenvalue analysis indicates that Matrix B may be best modeled in 3-D.

Interestingly, with the single exception of $Model_{B2}$, the max absolute error was between the two locations that have the shortest non-Euclidean distance (Yokohama-Tokyo and Buenos Aires-Cape Town).

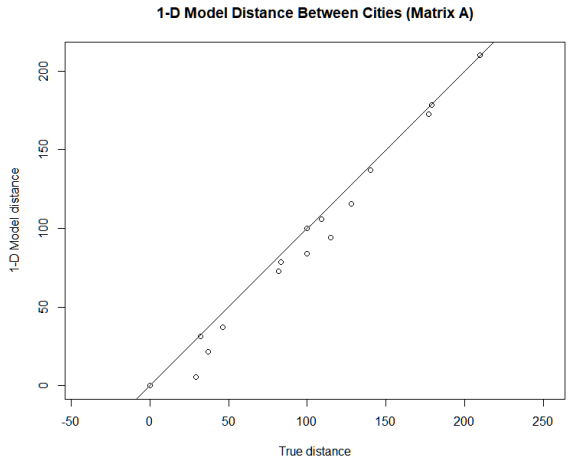
7 Distance Plots and Goodness of Fit

Next we will show plots with true distance between objects on the x-axis and the modeled distance on the y-axis. A perfect model would mean that the plots would only have points on the line y=x. Any point of the line y=x indicates error in the model.

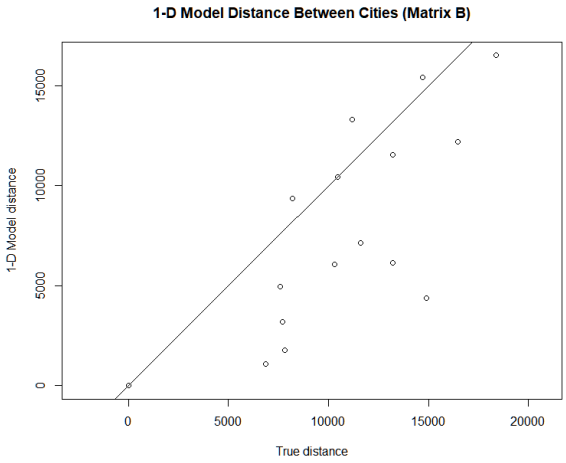
Below each plot is a measure called "Goodness of Fit" (GOF). GOF is a measurement of how well a model predicts a set of observations. In our example, we have observed the non-Euclidean distance between locations on the surface of the Earth. The GOF for our models is then a

number between 0 and 1 (1 being perfectly fit, 0 being no fit) that describes how well the MDS model distances fit the actual observed distances. Another more intuitive notion of GOF is the "skinniness" of the plot. A wide, randomly dispersed graph will have a low GOF, while a plot that follows a line (e.g. y=x) will have a high GOF. The GOF number for each of our models is below each plot.

7.1 1-D



(a) GOF: 0.9095307

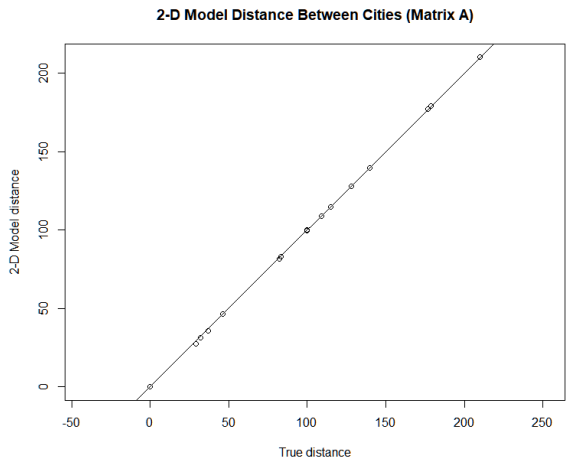


(b) GOF: 0.4342871

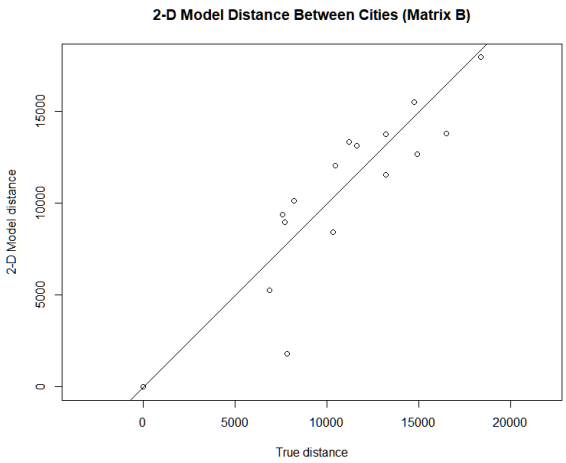
The distance graph of $Model_{A1}$ a relatively nice fit between the model distance and the observed distances. As you can see, the plot is "skinny" and lies almost exactly on the line $y = x$. This is also reflected with a GOF of .910. The Eigenvalues and absolute error also suggested that a 1-D model would capture the true distance relationships of Matrix A fairly well.

The distance graph of $Model_{B1}$ is a bit "wider" and the points do not all lie on the line $y = x$. This indicated error in the 1-D model, which was also indicated in the eigenvalue and absolute error analysis.

7.2 2-D



(a) GOF: 0.9952146

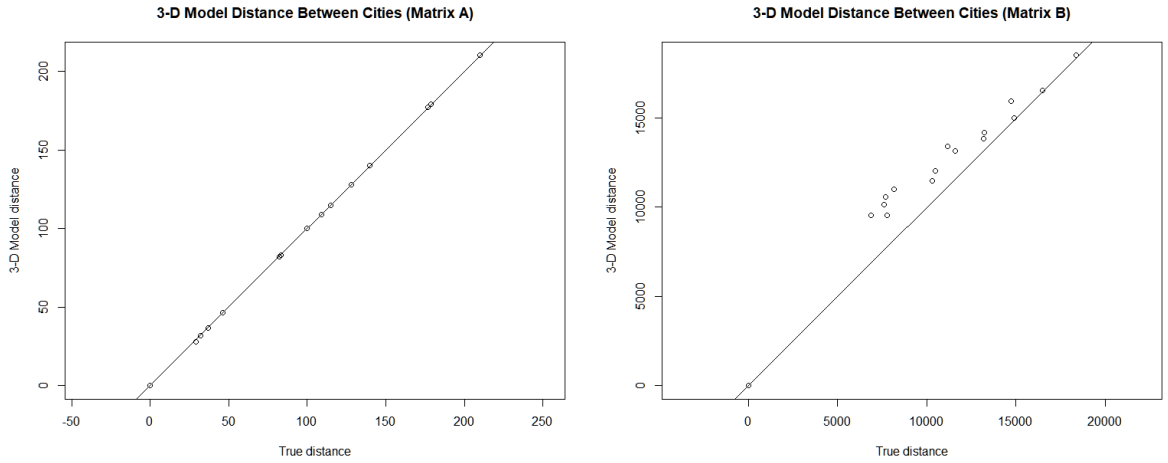


(b) GOF: 0.6793212

The distance graphs between $Model_{A1}$ and $Model_{A2}$ do not seem to change much, but the GOF changes significantly from .910 to .995, respectively. This is a nearly perfect fit between the model and the observed distances. The Eigenvalue and absolute error analysis also indicated that a 2-D model would properly capture the relationships within Matrix A.

The distance graphs between $Model_{B1}$ and $Model_{B2}$ also show much improvement in higher dimensions, with the GOF jumping from .434 to .679, respectively.

7.3 3-D



(a) GOF: 0.9969886

(b) GOF: 0.8482584

The distance graph for $Model_{A3}$ looks the same as $Model_{A2}$. This is confirmed by seeing that the GOF only changed from .995 to .997. This subtle change means that Matrix A can be modeled in 2-D nearly as well as 3-D.

The distance graph for $Model_{B3}$ improves significantly from the lower dimension. The graph looks "skinnier" and this is confirmed with a GOF jump from .679 to .848.

8 Tables

	$Model_{A1}$	$Model_{A2}$	$Model_{A3}$	$Model_{B1}$	$Model_{B2}$	$Model_{B3}$
max error	23.24224	1.524695	1.142773	10502.23	5985.221	2864.534
mean error	8.157484	0.3660635	0.2101351	3800.804	1875.478	1486.843
GOF	0.9095307	0.9952146	0.9969886	0.4342871	0.6793212	0.8482584

9 Conclusion

A model is only useful if it is simpler than the thing it is modeling. For this reason, we wish to choose MDS models of the lowest dimension possible. In this way we distill complicated distance relationships into easier to understand graphics and metrics.

For Matrix A, it seems that a 2-D MDS model is best. The 2-D model has a nearly perfect GOF (.995). From the Eigenvalue and absolute error analyis, we can see that creating a 3-D model of Matrix A would add little to accuracy, while complicating the model with another dimension.

Matrix B is prehaps best modelled using a different modeling system altogether. If MDS must be used, it seems best modeled in 3-D. The GOF is only .848, but this is the best possible fit using MDS. The negative Eigenvalues of the higher dimensions indicate non-Euclidean distances that cannot be handled by the cmdscale() function in R.

An interesting aspect of the error analysis revealed that in most cases the maximum error was between locations that have the shortest non-Euclidean distance. This is perhaps an effect of "stretching" those distances to minimize error elsewhere.

10 References

Figure 1,3 and location distances were found using interactive maps at;
https://mapmaker.nationalgeographic.org/

11 Code

The MDS models and matrix manipulation was done using R

```
    A <- matrix(1:36, nrow = 6, ncol = 6, dimnames =
    list(c("Tokyo", "Kofu", "Asahi", "Mt Fuji", "Yokohama", "Chiba"),
    c("Tokyo", "Kofu", "Asahi", "Mt Fuji", "Yokohama", "Chiba")))

for(i in 1:6) {
  A[i,i]=0
}
A[1,2]=109
A[2,1]=109
A[1,3]=179
A[3,1]=179
A[1,4]=100
A[4,1]=100
A[1,5]=29
A[5,1]=29
A[1,6]=32
A[6,1]=32
A[2,3]=82
A[3,2]=82
A[2,4]=37
A[4,2]=37
A[2,5]=100
A[5,2]=100
A[2,6]=140
A[6,2]=140
A[3,4]=115
A[4,3]=115
A[3,5]=177
A[5,3]=177
A[3,6]=210
```

```
A[6,3]=210
A[5,4]=83
A[4,5]=83
A[4,6]=128
A[6,4]=128
A[5,6]=46
A[6,5]=46
A

# 1D

model_1<-cmdscale(A, k=1, eig = TRUE)
x1<-model_1$points
model_1

plot(x1,x1*0, xlim=c(-130,100),xlab="x",ylab = "y", main = "1-D Model Matrix A")

text(x1,x1*0,labels=rownames(model_1$points), cex=1, pos=3)

dist_1 <- as.matrix(dist(model_1$points))
Error_1_1<-(A-dist_1)/A
Error_1_1

GOF_1_1<-model_1$GOF
GOF_1_1

Eig_1_1<-model_1$eig
Eig_1_1

plot(Eig_1_1, xlab="Index of Eigenvalue", ylab="Eigenvalue",
main="Eigenvalues of Model A")

plot(A,dist_1,xlab = "True distance", ylab = "1-D Model distance",
main = "1-D Model Distance Between Cities (Matrix A)", asp = 1)
abline(0,1)

hist_1_ob<-hist(abs(Error_1_1), breaks=15)
hist_1_ob$counts <- hist_1_ob$counts/2
plot(hist_1_ob, xlab="% error between model distances and actual distances",
main = "Absolute Error 1-D Model Matrix A" ,col = "gray")



# 2D
model_2<- cmdscale(A, k=2, eig = TRUE)


x2<-model_2$points[,1]
```

```
y2<-model_2$points[,2]

plot(x2,y2, xlim=c(-130,100), ylim = c(-40, 50),xlab="x",ylab = "y",
main = "2-D Model Matrix A", asp = 1)

text(x2,y2,labels=rownames(model_2$points), cex=1, pos=3)


dist_2 = as.matrix(dist(model_2$points))
Error_1_2<-(A-dist_2)/A
Error_1_2


GOF_1_2<-model_2$GOF
GOF_1_2

plot(A,dist_2,xlab = "True distance", ylab = "2-D Model distance",
main = "2-D Model Distance Between Cities (Matrix A)", asp = 1)
abline(0,1)

hist_2_ob<-hist(abs(Error_1_2), breaks=15)
hist_2_ob$counts<-hist_2_ob$counts/2
plot(hist_2_ob, xlab="% error between model distances and actual distances",
main = "Absolute Error 2-D Model Matrix A" ,col = "gray")


#3D
model_3<-cmdscale(A,k=3, eig = TRUE)
model_3


dist_3<-as.matrix(dist(model_3$points))
Error_1_3<-(A-dist_3)/A
Error_1_3


GOF_1_3<-model_3$GOF
GOF_1_3


plot(A,dist_3,xlab = "True distance", ylab = "3-D Model distance",
main = "3-D Model Distance Between Cities (Matrix A)", asp = 1)
abline(0,1)


hist_3_ob<-hist(abs(Error_1_3), breaks=15)
hist_3_ob$counts<-hist_3_ob$counts/2
plot(hist_3_ob, xlab="% error between model distances and actual distances",
```

```
main = "Absolute Error 3-D Model Matrix A" ,col = "gray")



## Larger map
B <- matrix(1:36, nrow = 6, ncol = 6, dimnames = list(c("Tokyo",
"St Petersburg", "Seattle", "Cape Town", "Buenos Aires", "Melbourne"),
c("Tokyo", "St Petersburg", "Seattle", "Cape Town", "Buenos Aires",
"Melbourne")))

for(i in 1:6) {
  B[i,i]=0
}
B[1,2]=7586
B[2,1]=7586
B[1,3]=7695
B[3,1]=7695
B[1,4]=14718
B[4,1]=14718
B[1,5]=18362
B[5,1]=18362
B[1,6]=8185
B[6,1]=8185
B[2,3]=7799
B[3,2]=7799
B[2,4]=10461
B[4,2]=10461
B[2,5]=13207
B[5,2]=13207
B[2,6]=14897
B[6,2]=14897
B[3,4]=16469
B[4,3]=16469
B[3,5]=11177
B[5,3]=11177
B[3,6]=13200
B[6,3]=13200
B[5,4]=6865
B[4,5]=6865
B[4,6]=10308
B[6,4]=10308
B[5,6]=11603
B[6,5]=11603


# 1D
model_2_1<-cmdscale(B,k=1, eig = TRUE)
```

```
plot(model_2_1$points[,1], model_2_1$points[,1]*0,xlab="x",ylab = "y",
main = "1-D Model Matrix B", asp = 1)
text(model_2_1$points[,1], model_2_1$points[,1]*0,
labels=rownames(model_2_1$points), cex=.55, pos=3)

dist_2_1<-as.matrix(dist(model_2_1$points))

Error_2_1<-(B-dist_2_1)/B
Error_2_1


GOF_2_1<-model_2_1$GOF
GOF_2_1

Eig_2_1<-model_2_1$eig
Eig_2_1

plot(Eig_2_1, xlab="Index of Eigenvalue", ylab="Eigenvalue",
main="Eigenvalues of Model B")

plot(B,dist_2_1,xlab = "True distance", ylab = "1-D Model distance",
main = "1-D Model Distance Between Cities (Matrix B)", asp = 1)
abline(0,1)

hist_2_1_ob<-hist(abs(Error_2_1), breaks = 15)
hist_2_1_ob$counts<-hist_2_1_ob$counts/2
plot(hist_2_1_ob, xlab="% error between model distances and actual distances",
main = "Absolute Error 1-D Model Matrix B" ,col = "gray")


#2-D
model_2_2<-cmdscale(B,k=2, eig = TRUE)
x2<-model_2_2$points[,1]
y2<-model_2_2$points[,2]

plot(x2,y2, xlim=c(min(x2)-500, max(x2)+500), ylim = c(min(y2)-500, max(y2)+500),xlab="x",yla

text(x2,y2,labels=rownames(model_2_2$points), cex=1, pos=3)

dist_2_2 = as.matrix(dist(model_2_2$points))

Error_2_2<-(B-dist_2_2)/B
Error_2_2

max_dist_2_2<-max(dist_2_2)

GOF_2_2<-model_2_2$GOF
GOF_2_2
```

```
plot(B,dist_2_2,xlab = "True distance", ylab = "2-D Model distance",
main = "2-D Model Distance Between Cities (Matrix B)", asp = 1)
abline(0,1)

hist_2_2_ob<-hist(abs(Error_2_2), breaks = 15)
hist_2_2_ob$counts<-hist_2_2_ob$counts/2
plot(hist_2_2_ob, xlab="% error between model distances and actual distances",
main = "Absolute Error 2-D Model Matrix B" ,col = "gray")


#3-D
model_2_3<-cmdscale(B,k=3, eig = TRUE)

dist_2_3 <- as.matrix(dist(model_2_3$points))

Error_2_3<- (B-dist_2_3)/B
Error_2_3

GOF_2_3<-model_2_3$GOF
GOF_2_3

plot(B,dist_2_3,xlab = "True distance", ylab = "3-D Model distance",
main = "3-D Model Distance Between Cities (Matrix B)", asp = 1)
abline(0,1)

hist_2_3_ob<-hist(abs(Error_2_3), breaks = 15)
hist_2_3_ob$counts<-hist_2_3_ob$counts/2
plot(hist_2_3_ob, xlab="% error between model distances and actual distances",
main = "Absolute Error 3-D Model Matrix B" ,col = "gray")


hist_1<-hist(A, breaks=10)
hist_2<-hist(dist_1, breaks=10)

max(abs(dist(model_2$points)-as.dist(A)))
mean(abs(dist(model_2$points)-as.dist(A)))

max(abs(dist(model_2_2$points)-as.dist(B)))
mean(abs(dist(model_2_2$points)-as.dist(B)))


max(abs(dist(model_3$points)-as.dist(A)))
mean(abs(dist(model_3$points)-as.dist(A)))

max(abs(dist(model_2_3$points)-as.dist(B)))
mean(abs(dist(model_2_3$points)-as.dist(B)))
```