Classical Multidimensional Scaling (MDS) - Mathematical and Practical Steps in R

Benjamin Grant Purzycki

Last Updated: January 10, 2025

MDS

MDS is a family of techniques designed to examine the relational structure of elements in a matrix, often with the goal of reducing the complex relationships between those elements and classes of those elements. In `R`, the following provides what the `cmdscale()` function performs in a step-by-step fashion.

While there are many options for this particular process, the steps involved in MDS are common across these options. Step 1 is to compute a distance matrix from your original data. Step 2 entails double centering this distance matrix. Step 3 is to eigendecompose the double-centered matrix. Step 4 involves selecting the principal components from the resulting eigendecomposition. Step 5 computes the coordinates from those components, and Step 6 entails plotting and interpretation. There are some appendices that dig a little deeper into the details.

Here's some simulated data to work with. It's a 5 $\times$ 3 matrix (e.g., a participant x variable matrix) of 15 values $\sim$ Normal$(0, 1)$. We'll start with this just to focus on the steps, then use a more intuitive example to emphasize interpretation.

```
set.seed(123)
data <- matrix(rnorm(15), nrow = 5)
```

You should get the following:

```
            [,1]       [,2]       [,3]
[1,] -0.56047565  1.7150650  1.2240818
[2,] -0.23017749  0.4609162  0.3598138
[3,]  1.55870831 -1.2650612  0.4007715
[4,]  0.07050839 -0.6868529  0.1106827
[5,]  0.12928774 -0.4456620 -0.5558411
```

Our analytical task is to examine the interrelationships between the elements. These relationships are represented as having dimensions; things can relate it different ways. It's randomly drawn data, so there shouldn't be much of any structure here.

Step 1: Compute the Distance Matrix

Let $\mathbf{X}$ be a matrix with n rows (samples) and p columns (features). The Euclidean distance matrix $\mathbf{D}$ is calculated as:

$$D_{ij} = \sqrt{\sum_{k=1}^{p}(X_{ik} - X_{jk})^2}$$

This equation finds the square root of the sum of squared distances between each element in each item. To do this in `R`, use the `dist()` function, but be sure to turn define it *as* a matrix, otherwise the subsequent steps that entail linear algebra won't work.

```
D <- as.matrix(dist(data))
```

This should yield the following 5×5 square distance matrix $\mathbf{D}$:

```
         1        2        3         4         5
1 0.000000 1.558507 3.748324 2.7215818 2.8831650
2 1.558507 0.000000 2.486119 1.2123746 1.3377306
3 3.748324 2.486119 0.000000 1.6227185 1.9051945
4 2.721582 1.212375 1.622719 0.0000000 0.7112539
5 2.883165 1.337731 1.905194 0.7112539 0.0000000
```

This matrix is a Euclidean distance matrix of the row items from our original data set. You'll notice the 0's on the diagonal (i.e., an item has a distance of 0 with itself). If our data were originally a participant x variable matrix, $\mathbf{D}$ would be a distance matrix of *individuals*, not variables. To get a distance matrix of *variables*, we would just need to transpose the original dataset before calculating the distances using the `t()` function (e.g., `D <- as.matrix(dist(t(data)))`).

Step 2: Double Center the Distance Matrix

To double center the distance matrix, we first compute the appropriate centering matrix $\mathbf{H}$, which is defined as:

$$\mathbf{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$$

where $\mathbf{I}_n$ is the $n \times n$ identity matrix and $\mathbf{1}_n$ is a column vector of ones of length n. The product of $\mathbf{1}_n\mathbf{1}_n^T$ is an $n \times n$ all-ones matrix. This matrix helps us convert a distance matrix into coordinates. See Appendix A for more details. In `R`, we need to first define these matrices with properties drawn from our original data, namely, the number of rows that determine the dimensions of $\mathbf{D}$:

```
n <- nrow(D)
H <- diag(n) - (1/n) * matrix(1, n, n)
```

H should look like this:

```
     [,1] [,2] [,3] [,4] [,5]
[1,]  0.8 -0.2 -0.2 -0.2 -0.2
[2,] -0.2  0.8 -0.2 -0.2 -0.2
[3,] -0.2 -0.2  0.8 -0.2 -0.2
[4,] -0.2 -0.2 -0.2  0.8 -0.2
[5,] -0.2 -0.2 -0.2 -0.2  0.8
```

In general, a centering matrix has a few notable properties. First, it has a mean of zero. Second, its rows and columns also have means of zeros. Third, if we multiply a vector or matrix by its corresponding centering matrix, the product will have a mean of zero. As such, it is a way to convert an original matrix to its centered state. The original matrix minus the mean of itself is equal to this product. Next, we compute the matrix **B** by applying the following transformation to the squared distance matrix:

$$\mathbf{B} = -\frac{1}{2}\mathbf{H}\mathbf{D}^2\mathbf{H}$$

```
B <- -0.5 * H %*% (D^2) %*% H
```

Matrix **B** should look like this:

```
          [,1]       [,2]       [,3]       [,4]       [,5]
[1,]  4.503403  1.2559888 -3.0920457 -1.2183576 -1.4489885
[2,]  1.255989  0.4375196 -1.1904151 -0.2827216 -0.2203717
[3,] -3.092046 -1.1904151  3.3624384  0.5980561  0.3219663
[4,] -1.218358 -0.2827216  0.5980561  0.4668893  0.4361337
[5,] -1.448988 -0.2203717  0.3219663  0.4361337  0.9112602
```

Step 3: Eigendecomposition of the Centered Matrix

Perform the eigendecomposition of the matrix **B**:

$$\mathbf{B} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$$

where **V** is the matrix of eigenvectors and **Λ** is the diagonal matrix of eigenvalues. If you use the `eigen()` function on `B`, you should get the following (rounded to fit):

```
eig <- eigen(B)

> eig

eigen() decomposition
$values
[1]  8.03 1.49 0.16 0.00 0.00

$vectors
      [,1]  [,2]  [,3] [,4]  [,5]
[1,] -0.73 -0.34  0.08 0.49 -0.32
[2,] -0.23  0.14  0.01 0.35  0.90
[3,]  0.58 -0.66  0.17 0.44  0.08
[4,]  0.18  0.24 -0.81 0.47 -0.17
[5,]  0.19  0.61  0.55 0.48 -0.24
```

Eigenvectors of a matrix are nonzero vectors where $\mathbf{Ax} = \lambda\mathbf{x}$. That is, multiplying a given matrix $\mathbf{A}$ by eigenvector $\mathbf{x}$ is the same as multiplying $\mathbf{x}$ by eigenvalue λ. See Appendix B for more technical details.

Step 4: Select Principal Components

To get the principal components, we select eigenvalues $\lambda_i > 0$ and their corresponding eigenvectors $\mathbf{v}_i$. Let $\mathbf{\Lambda}_+$ and $\mathbf{V}_+$ represent the matrices of the selected eigenvalues and eigenvectors, respectively.

```
posevalues <- eig$values[eig$values > 0]
posevectors <- eig$vectors[, eig$values > 0]

> posevalues
[1] 8.0253889 1.4939783 0.1621434

> posevectors
           [,1]       [,2]        [,3]
[1,] -0.7344692 -0.3404724  0.07704237
[2,] -0.2253083  0.1419459  0.01064654
[3,]  0.5815643 -0.6561226  0.17493920
[4,]  0.1840366  0.2426350 -0.81280530
[5,]  0.1941767  0.6120141  0.55017720
```

Step 5: Compute the Coordinates in Reduced Space

The coordinates in the lower-dimensional space are given by:

$$\mathbf{Y} = \mathbf{V}_{+}\mathbf{\Lambda}_{+}^{1/2}$$

where $\mathbf{\Lambda}_{+}^{1/2}$ is the diagonal matrix of square roots of the positive eigenvalues. If we want to reduce the dimensionality to k, we take only the first k columns of $\mathbf{Y}$, denoted as $\mathbf{Y}_k$:

$$\mathbf{Y}_k = \mathbf{V}_{+,k}\mathbf{\Lambda}_{+,k}^{1/2}$$

```
k <- 2 # dimensions
coordinates <- posevectors %*% diag(sqrt(posevalues))
coordinatesk <- coordinates[, 1:k]

> coordinates

           [,1]       [,2]         [,3]
[1,] -2.0806866 -0.4161540  0.031022676
[2,] -0.6382786  0.1734982  0.004287046
[3,]  1.6475203 -0.8019682  0.070442824
[4,]  0.5213595  0.2965689 -0.327292582
[5,]  0.5500853  0.7480550  0.221540036

> coordinatesk

           [,1]       [,2]
[1,] -2.0806866 -0.4161540
[2,] -0.6382786  0.1734982
[3,]  1.6475203 -0.8019682
[4,]  0.5213595  0.2965689
[5,]  0.5500853  0.7480550
```

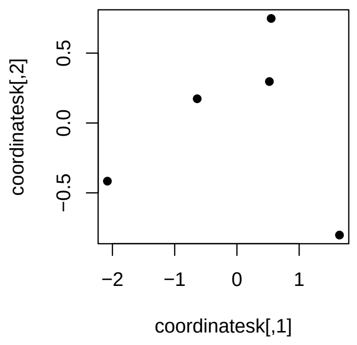

Figure 1: **MDS plot of randomly drawn data.**

Step 6: Plot the coordinates and interpret

Finally, we take the coordinates generated in the previous step and plot them. Figure 1 plots the data points. As the data were randomly generated, there is no clear structure and the axes–or dimensions–do not lend themselves to much interpretation.

Now that we have reviewed the steps, let's look at a common, intuitive example that lends itself to interpretation. Table 1 is a (literal) distance matrix of American cities.

	BOS	CHI	DC	DEN	LA	MIA	NY	SEA	SF
BOS	0	963	429	1949	2979	1504	206	2976	3095
CHI	963	0	671	996	2054	1329	802	2013	2142
DC	429	671	0	1616	2631	1075	233	2684	2799
DEN	1949	996	1616	0	1059	2037	1771	1307	1235
LA	2979	2054	2631	1059	0	2687	2786	1131	379
MIA	1504	1329	1075	2037	2687	0	1308	3273	3053
NY	206	802	233	1771	2786	1308	0	2815	2934
SEA	2976	2013	2684	1307	1131	3273	2815	0	808
SF	3095	2142	2799	1235	379	3053	2934	808	0

Table 1: **Distance matrix of American cities.**

Figure 2 plots the resulting coordinates from the MDS process. Here, the dimensions are clear; they are directions. The values along the axes are miles. You can see that Chicago (CHI) and Denver (DEV) are about 1,000 miles apart. Of course, the analysis doesn't know that these points are American cities and doesn't know the meaning of the dimensions (i.e., directions). It also doesn't know directions, so its solution is upside-down (in our perspective). The solution is nevertheless correct in terms of the relative distance of each city from every other city.

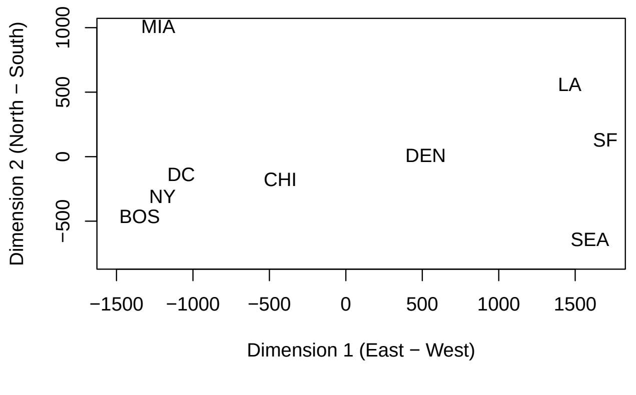

Figure 2: **MDS plot of American cities.**

Appendices

A Double Centering

Double centering is a relatively simple process and entails mean-centering a matrix. To illustrate, let's just take a simple 3×1 matrix and show that the first step is just mean-centering:

$$\mathbf{x} = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}$$
$$\mathbf{x} - \bar{\mathbf{x}} = \begin{bmatrix} -2 \\ 0 \\ 2 \end{bmatrix}$$

Because we typically have more elaborate matrices, we'll need more elaborate ways to compute this mean-centered version. By way of proof, we can see that the original equation $\mathbf{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$ produces the same solution:

$$\mathbf{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$$
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \end{bmatrix}$$
$$\mathbf{H} = \begin{bmatrix} 0.67 & -0.33 & -0.33 \\ -0.33 & 0.67 & -0.33 \\ -0.33 & -0.33 & 0.67 \end{bmatrix}$$
$$\mathbf{Hx} = \begin{bmatrix} 0.67 & -0.33 & -0.33 \\ -0.33 & 0.67 & -0.33 \\ -0.33 & -0.33 & 0.67 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \\ 2 \end{bmatrix}$$

So, we've mean-centered our matrix. Now, we'll apply another centering procedure. Now, given that distance matrix

$$\mathbf{D_x} = \begin{bmatrix} 0 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 0 \end{bmatrix}$$

we can calculate the double-centered matrix:

$$\mathbf{B} = -\frac{1}{2}\mathbf{H}\mathbf{D}^2\mathbf{H}$$
$$\mathbf{B} = -\frac{1}{2}\begin{bmatrix} 0.67 & -0.33 & -0.33 \\ -0.33 & 0.67 & -0.33 \\ -0.33 & -0.33 & 0.67 \end{bmatrix}\begin{bmatrix} 0 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 0 \end{bmatrix}^2\begin{bmatrix} 0.67 & -0.33 & -0.33 \\ -0.33 & 0.67 & -0.33 \\ -0.33 & -0.33 & 0.67 \end{bmatrix}$$
$$\mathbf{B} = \begin{bmatrix} 4 & 0 & -4 \\ 0 & 0 & 0 \\ -4 & 2 & 4 \end{bmatrix}$$

B Eigendecomposition

B.1 Determinants

A determinant is a particular value associated with square matrices. For simple 2×2 matrices, the determinant is calculated as follows:

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$
$$\det(\mathbf{A}) = ad - bc$$

For a simple 3×3 matrix, it is:

$$\mathbf{A} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$
$$\det(\mathbf{A}) = aei - afh - bdi + bfg + cdh - ceg$$

For any n-dimension square matrix, there are $n!$ products to calculate and can be generally calculated using $\det(\mathbf{A}) = \sum(-1)^{i+j}a_{ij}|Cij|$ where i and j are the row and column numbers, a_{ij} is a specific element of the matrix, and $|C_{ij}|$ is the determinant of the minor associated with element a_{ij}. The determinant's minor is a reduced determinant produced after deleting a specific row and column of the original matrix.

B.2 Eigendecomposition

Steps in the eigendecomposition of a matrix are as follows:

1. Find eigenvalues: $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$
2. Find eigenvectors: $(\mathbf{A} - \lambda_i\mathbf{I})\mathbf{v}_i = 0$
3. Form eigenvector matrix $\mathbf{V}$ and eigenvalue matrix $\mathbf{\Lambda}$
4. Confirm: $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathbf{T}}$

B.2.1 Step 1: Find eigenvalues

First, we must solve the characteristic polynomial equation $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$. For a 2×2 matrix:

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$\det(\mathbf{A} - \lambda\mathbf{I}) = \det \begin{bmatrix} a - \lambda & b \\ c & d - \lambda \end{bmatrix} = (a - \lambda)(d - \lambda) - bc$$

The polynomial equation is equivalent to:

$$\mathbf{A} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{1}$$

where v_1 and v_2 are elements in the unknown eigenvector and λ is an unknown eigenvalue. Here's a concrete, non-trivial (i.e., where the eigenvector is a non-zero vector) example. Let

$$\mathbf{A} = \begin{bmatrix} 4 & 3 \\ 2 & -1 \end{bmatrix}$$

So:

$$\begin{aligned} &\det(\mathbf{A} - \lambda\mathbf{I}) \\ &= \det \begin{bmatrix} 4 - \lambda & 3 \\ 2 & -1 - \lambda \end{bmatrix} \\ &= (4 - \lambda)(-1 - \lambda) - 2(3) \\ &= -4 - 4\lambda + \lambda + \lambda^2 - 6 \\ &= \lambda^2 - 3\lambda + 10 \end{aligned}$$

Here, λ values are the solutions to the quadratic equation:

$$\lambda = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Here, then, $a = 1$, $b = -3$, and $c = -10$. Solving this equation yields to eigenvalues: $\lambda_1 = 5$ and $\lambda_2 = -2$.

B.2.2 Step 2: Find eigenvectors

Using these eigenvalues, we now need to calculate the eigenvectors. We can use $(\mathbf{A} - \lambda_i\mathbf{I})\mathbf{v}_i = 0$, but we can also use Equation 1. For the sake of illustration, let's calculate them both ways. As we have two eigenvalues in this context, we will have two eigenvectors, $\mathbf{v_1}$ and $\mathbf{v_2}$. So:

$$
\begin{aligned}
(\mathbf{A} - \lambda_1\mathbf{I})\mathbf{v_1} &= 0 \\
\left(\begin{bmatrix} 4 & 3 \\ 2 & -1 \end{bmatrix} - 5\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)\mathbf{v_1} &= 0 \\
\left(\begin{bmatrix} 4 & 3 \\ 2 & -1 \end{bmatrix} - \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}\right)\mathbf{v_1} &= 0 \\
\begin{bmatrix} -1 & 3 \\ 2 & -6 \end{bmatrix}\mathbf{v_1} &= 0 \\
\begin{bmatrix} -1 & 3 \\ 2 & -6 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\end{aligned}
$$

Here, we have a system of two equations:

$$
\begin{aligned}
-v_1 + 3v_2 &= 0 \\
2v_1 - 6v_2 &= 0
\end{aligned}
$$

which we can simplify to:

$$
\begin{aligned}
v_1 &= 3v_2 \\
2v_1 &= 6v_2
\end{aligned}
$$

which are just multiples of each other. Taking $v_1 = 3v_2$, we can solve for $\mathbf{v_1}$ by letting $v_2 = t$, where t is a stand-in value :

$$
\begin{aligned}
\mathbf{v_1} &= \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\
\mathbf{v_1} &= \begin{bmatrix} v_1 \\ t \end{bmatrix} \\
\mathbf{v_1} &= \begin{bmatrix} 3t \\ t \end{bmatrix} \\
\mathbf{v_1} &= \begin{bmatrix} 3 \\ 1 \end{bmatrix}
\end{aligned}
$$

Again, we can also find the eigenvector using the alternative formulation in Equation 1:

$$\begin{aligned}
\mathbf{A}\begin{bmatrix}v_1\\v_2\end{bmatrix} &= \lambda\begin{bmatrix}v_1\\v_2\end{bmatrix}\\
\begin{bmatrix}4 & 3\\2 & -1\end{bmatrix}\begin{bmatrix}v_1\\v_2\end{bmatrix} &= 5\begin{bmatrix}v_1\\v_2\end{bmatrix}\\
\begin{bmatrix}4 & 3\\2 & -1\end{bmatrix}\begin{bmatrix}v_1\\v_2\end{bmatrix} &= \begin{bmatrix}5v_1\\5v_2\end{bmatrix}\\
\begin{bmatrix}4v_1+3v_2\\2v_1-1v_2\end{bmatrix} &= \begin{bmatrix}5v_1\\5v_2\end{bmatrix}\\
4v_1+3v_2 &= 5v_1\\
2v_1-v_2 &= 5v2
\end{aligned}$$

which once again brings us to $v_2 = 3v_2$. We still need to solve for $\lambda_2 = -2$. Following the steps correctly yields:

$$\mathbf{v_2} = \begin{bmatrix}1\\-2\end{bmatrix}$$

B.2.3 Step 3: Form eigenvector and eigenvalue matrices

So, to confirm, recall that $\mathbf{A} = \mathbf{V\Lambda V}^T$. As $\lambda_1 = 5$ and $\lambda_2 = -2$,

$$\mathbf{\Lambda} = \begin{bmatrix}5 & 0\\0 & -2\end{bmatrix}$$

whereas $\mathbf{V}$ is the eigenvector matrix:

$$\mathbf{V} = \begin{bmatrix}3 & 1\\1 & -2\end{bmatrix}$$

B.2.4 Step 4: Prove that $\mathbf{A = V\Lambda V^T}$

Here, then

$$\begin{aligned}
\mathbf{A} &= \mathbf{V\Lambda V}^T\\
\mathbf{A} &= \begin{bmatrix}3 & 1\\1 & -2\end{bmatrix}\begin{bmatrix}5 & 0\\0 & -2\end{bmatrix}\begin{bmatrix}3 & 1\\1 & -2\end{bmatrix}^T\\
\mathbf{A} &= \begin{bmatrix}4 & 3\\2 & -1\end{bmatrix}
\end{aligned}$$

B.2.5 Note on normalization

If we run the `eigen()` function in `R`, we'll get the same eigenvalues, but we'll get different eigenvalues. Part of the `R` output of the eigendecomposition of matrix $\mathbf{A}$ will look like this:

```
$values
[1]  5 -2

$vectors
[,1]       [,2]
[1,] 0.9486833 -0.4472136
[2,] 0.3162278  0.8944272
```

Why? First, note that they should be proportional to the values we calculated by hand (i.e., 3 to 1 is 0.95 to 0.32 as 1 to 2 is 0.45 to 0.89). In other words, the vectors have just been rescaled (i.e., they were normalized). Second, note that the sign of the values is reversed for v_2. This too can happen in the process of normalization.

Normalization puts eigenvectors into a standardized space so that their length is 1. We use the vector's norm to do this. One way to calculate the norm is to take the square root of the sum of squared elements (the Euclidean norm or L_2-norm):

$$||\mathbf{v}|| = \sqrt{v_1^2 + v_2^2 ... v_n^2} \qquad \text{if } \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ ... \\ v_n \end{bmatrix}$$

For example:

$$||\mathbf{v}|| = \sqrt{2} \qquad \text{if } \mathbf{v} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

We then use this root to normalize $\mathbf{v}$ by dividing each of its elements by the root:

$$\mathbf{v}_{\text{normalized}} = \frac{\mathbf{v}}{||\mathbf{v}||} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

To calculate the length of this normalized vector (and demonstrate it is equal to 1), we plug in the values into the distance equation:

$$||\mathbf{v}_{\text{normalized}}|| = \sqrt{\left(\frac{1}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2} = \sqrt{\frac{1}{2} + \frac{1}{2}} = 1$$

Applying this to eigenvectors $\mathbf{v_1}$ and $\mathbf{v_2}$ of matrix $\mathbf{A}$ from above, we get:

$$
\begin{aligned}
||\mathbf{v_1}|| &= \sqrt{3^2 + 1^2} \text{ ... if } \mathbf{v_1} = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
||\mathbf{v_1}|| &= 3.162278 \\
||\mathbf{v_1}_{\text{normalized}}|| &= \frac{1}{3.162278} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
||\mathbf{v_1}_{\text{normalized}}|| &= \begin{bmatrix} 0.9486833 \\ 0.3162278 \end{bmatrix}
\end{aligned}
$$

and for $\mathbf{v_2}$:

$$
\begin{aligned}
||\mathbf{v_2}|| &= \sqrt{1^2 - 2^2} \text{ ... if } \mathbf{v_2} = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \\
||\mathbf{v_2}|| &= 2.236068 \\
||\mathbf{v_2}_{\text{normalized}}|| &= \frac{1}{2.236068} \begin{bmatrix} 1 \\ -2 \end{bmatrix} \\
||\mathbf{v_2}_{\text{normalized}}|| &= \begin{bmatrix} 0.4472136 \\ -0.8944272 \end{bmatrix}
\end{aligned}
$$

So, we get the *values* from the `R` output, but the signs are different. We could have easily reversed the numbers (`R` appears to sort them) because what matters here is the relative distance between the two. The sign could also be reversed for each as the distance between the two values would be the same, just in a different direction. Ultimately, we're thinking of points, so the relative distance is what matters here. Flipping the column for $\mathbf{v_2}$, for example while leaving $\mathbf{v_1}$ alone would result in the same ultimate solution, but in different quadrants of a visualization.

B.2.6 Spectral decomposition

Earlier, we proved that we can use $\mathbf{B} = \mathbf{V\Lambda V}^T$ to eigendecompose a matrix, but also to reconstruct the original matrix when we know the eigenvalues and eigenvectors of said matrix. To show another way of this de- and re-composition process, take the following matrix:

$$
\mathbf{A} = \begin{bmatrix} 13 & -4 & 2 \\ -4 & 11 & -2 \\ 2 & -2 & 8 \end{bmatrix}
$$

Its eigenvalues are 17, 8, and 7 and the eigenvector matrix is:

$$\mathbf{V} = \begin{bmatrix} 0.75 & 0.67 & 0.00 \\ -0.60 & 0.67 & 0.45 \\ 0.30 & -0.33 & 0.89 \end{bmatrix}$$

Using $A = \sum \lambda_i v_i v_i^T$, we can spectrally reconstruct the original matrix.

$$A_1 = 17 \begin{bmatrix} 0.7453560 \\ -0.5962848 \\ 0.2981424 \end{bmatrix} \begin{bmatrix} 0.7453560 & -0.5962848 & 0.2981424 \end{bmatrix}$$

$$A_1 = \begin{bmatrix} 9.44 & -7.56 & 3.78 \\ -7.56 & 6.04 & -3.02 \\ 3.78 & -3.02 & 1.51 \end{bmatrix}$$

$$A_2 = 8 \begin{bmatrix} 0.6666667 \\ 0.6666667 \\ -0.3333333 \end{bmatrix} \begin{bmatrix} 0.6666667 & 0.6666667 & -0.3333333 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 3.56 & 3.56 & -1.78 \\ 3.56 & 3.56 & -1.78 \\ -1.78 & -1.78 & 0.89 \end{bmatrix}$$

$$A_3 = 7 \begin{bmatrix} 0.0000000 \\ 0.4472136 \\ 0.8944272 \end{bmatrix} \begin{bmatrix} 0.0000000 & 0.4472136 & 0.8944272 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0.00 & 0.00 & 0.00 \\ 0.00 & 1.40 & 2.80 \\ 0.00 & 2.80 & 5.60 \end{bmatrix}$$

Now, if we add A_1, A_2, and A_3 together, we get the original data set:

$$\begin{bmatrix} 9.44 & -7.56 & 3.78 \\ -7.56 & 6.04 & -3.02 \\ 3.78 & -3.02 & 1.51 \end{bmatrix} + \begin{bmatrix} 3.56 & 3.56 & -1.78 \\ 3.56 & 3.56 & -1.78 \\ -1.78 & -1.78 & 0.89 \end{bmatrix} + \begin{bmatrix} 0.00 & 0.00 & 0.00 \\ 0.00 & 1.40 & 2.80 \\ 0.00 & 2.80 & 5.60 \end{bmatrix}$$
$$= \begin{bmatrix} 13 & -4 & 2 \\ -4 & 11 & -2 \\ 2 & -2 & 8 \end{bmatrix}$$

References

Namboodiri NK. The mathematics of matrices. 2nd ed. London: John Wiley and Sons, Inc.; 1973.

Davis PJ. Matrix algebra: An introduction. London: Sage; 1984.

Weller SC, Romney AK. Metric scaling: Correspondence analysis. London: Sage; 1990.