Matrix Operations in R — A Minimal Introduction

James H. Steiger

Department of Psychology and Human Development
Vanderbilt University

Matrix Operations in R — A Minimal Introduction

Matrix Algebra in R

Preliminary Comments

- This is a very basic introduction
- For some more challenging basics, you might examine Chapter 5 of *An Introduction to R*, the manual available from the *Help PDF Manuals* menu selection in the R program

Defining a Matrix in R

- Suppose you wish to enter, then view the following matrix **A** in R

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

- You would use the R commands:

```
> A <- matrix(c(1,3,2,4),2,2)
> A

     [,1] [,2]
[1,]    1    2
[2,]    3    4
```

- Note that the numbers are, by default, entered into the matrix *columnwise*, i.e., by column.

Defining a Matrix in R

- You can enter the numbers by row, simply by adding an optional input variable
- Here are the R commands:

```
> A <- matrix(c(1,2,3,4),2,2,byrow=TRUE)
> A
     [,1] [,2]
[1,]    1    2
[2,]    3    4
```

Entering a Column Vector

- To enter a $p \times 1$ column vector, simply enter a $p \times 1$ matrix

```
> a <- matrix(c(1,2,3,4),4,1)
> a
     [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4
```

- Row vectors are, likewise, entered as $1 \times q$ matrices

Extracting Individual Elements

- Individual elements of a matrix are referred to by their subscripts
- For example, consider a matrix correlation matrix **R** given below
- To extract element $R_{3,1}$, we simply request `R[3,1]`

	1	2	3	4
1	1.00	0.40	0.30	0.30
2	0.40	1.00	0.20	0.20
3	0.30	0.20	1.00	0.30
4	0.30	0.20	0.30	1.00

```
> R[3,1]

[1] 0.3
```

Extracting a Row of a Matrix

- To get an entire row of a matrix, you name the row and leave out the column
- For example, in the matrix R below, to get the first row, just enter `R[1,]`

	1	2	3	4
1	1.00	0.40	0.30	0.30
2	0.40	1.00	0.20	0.20
3	0.30	0.20	1.00	0.30
4	0.30	0.20	0.30	1.00

```
> R[1,]
```

```
[1] 1.0 0.4 0.3 0.3
```

Extracting a Column of a Matrix

- To get an entire column of a matrix, you name the column and leave out the row
- For example, in the matrix R below, to get the first column, just enter `R[,1]`

	1	2	3	4
1	1.00	0.40	0.30	0.30
2	0.40	1.00	0.20	0.20
3	0.30	0.20	1.00	0.30
4	0.30	0.20	0.30	1.00

```
> R[,1]

[1] 1.0 0.4 0.3 0.3
```

Extracting Several Rows and/or Columns

Example (Extracting Several Rows and/or Columns)

Examine the following examples to see how we can extract any specified range of rows and/or columns

	1	2	3	4
1	1.00	0.40	0.30	0.30
2	0.40	1.00	0.20	0.20
3	0.30	0.20	1.00	0.30
4	0.30	0.20	0.30	1.00

```
> R[1:3,]
     [,1] [,2] [,3] [,4]
[1,]  1.0  0.4  0.3  0.3
[2,]  0.4  1.0  0.2  0.2
[3,]  0.3  0.2  1.0  0.3
> R[1:3,2:4]
     [,1] [,2] [,3]
[1,]  0.4  0.3  0.3
[2,]  1.0  0.2  0.2
[3,]  0.2  1.0  0.3
```

Joining Rows

- On occasion, we need to build up matrices from smaller parts
- You can combine several matrices with the same number of columns by joining them as rows, using the `rbind()` command
- Here is an example

Joining Rows

Example (Joining Rows)

```
> A <- matrix(c(1,3,3,9,6,5),2,3)
> B <- matrix(c(9,8,8,2,9,0),2,3)
> A
     [,1] [,2] [,3]
[1,]    1    3    6
[2,]    3    9    5
> B
     [,1] [,2] [,3]
[1,]    9    8    9
[2,]    8    2    0
> rbind(A,B)
     [,1] [,2] [,3]
[1,]    1    3    6
[2,]    3    9    5
[3,]    9    8    9
[4,]    8    2    0
> rbind(B,A)
     [,1] [,2] [,3]
[1,]    9    8    9
[2,]    8    2    0
[3,]    1    3    6
[4,]    3    9    5
```

Joining Columns

- In similar fashion, you can combine several matrices with the same number of rows by joining them as columnss, using the `cbind()` command
- Here is an example

Joining Columns

Example (Joining Columns)

```
> A <- matrix(c(1,3,3,9,6,5),2,3)
> B <- matrix(c(9,8,8,2,9,0),2,3)
> A
     [,1] [,2] [,3]
[1,]    1    3    6
[2,]    3    9    5
> B
     [,1] [,2] [,3]
[1,]    9    8    9
[2,]    8    2    0
> cbind(A,B)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    3    6    9    8    9
[2,]    3    9    5    8    2    0
> cbind(B,A)
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    9    8    9    1    3    6
[2,]    8    2    0    3    9    5
```

Matrix Addition and Subtraction

Adding or subtracting matrices is natural and straightforward, as the example below shows

Example

```
> A <- matrix(c(1,3,3,9),2,2)
> B <- matrix(c(9,8,8,2),2,2)
> A
     [,1] [,2]
[1,]    1    3
[2,]    3    9
> B
     [,1] [,2]
[1,]    9    8
[2,]    8    2
> A+B
     [,1] [,2]
[1,]   10   11
[2,]   11   11
> A-B
     [,1] [,2]
[1,]   -8   -5
[2,]   -5    7
```

Scalar Multiplication

To multiply a matrix by a scalar, simply use the multiplication symbol * For example,

Example (Scalar Multiplication)

```
> A
     [,1] [,2]
[1,]    1    3
[2,]    3    9
> 3*A
     [,1] [,2]
[1,]    3    9
[2,]    9   27
```

Matrix Multiplication

Matrix multiplication uses the %*% command

Example (Matrix Multiplication)

```
> A
     [,1] [,2]
[1,]    1    3
[2,]    3    9
> B
     [,1] [,2]
[1,]    9    8
[2,]    8    2
> A %*% B
     [,1] [,2]
[1,]   33   14
[2,]   99   42
> B %*% A
     [,1] [,2]
[1,]   33   99
[2,]   14   42
```

Matrix Transposition

To transpose a matrix, use the `t()` command

Example (Transposing a matrix)

```
> A
     [,1] [,2] [,3]
[1,]    1    3    6
[2,]    3    9    5
> B
     [,1] [,2] [,3]
[1,]    9    8    9
[2,]    8    2    0
> t(A)
     [,1] [,2]
[1,]    1    3
[2,]    3    9
[3,]    6    5
> t(B)
     [,1] [,2]
[1,]    9    8
[2,]    8    2
[3,]    9    0
```

Matrix Inversion

- To invert a square matrix, use the `solve()` command
- In the example below, we illustrate a common problem — numbers that are really zero are only very close to zero due to rounding error
- When we compute the product $\mathbf{A}\mathbf{A}^{-1}$, we should get the identity matrix $\mathbf{I}$, but instead we see that the off-diagonal elements are not quite zero.
- To cure this problem, you can use the `zapsmall()` function

Matrix Inversion

Example (Inverting a matrix)

```
> A
     [,1] [,2] [,3]
[1,]    1    9    9
[2,]    3    6    1
[3,]    3    5    8
> solve(A)
            [,1]       [,2]       [,3]
[1,] -0.24855491  0.1560694  0.2601156
[2,]  0.12138728  0.1098266 -0.1502890
[3,]  0.01734104 -0.1271676  0.1213873
> A %*% solve(A)
              [,1]          [,2]         [,3]
[1,]  1.000000e+00  0.000000e+00 0.000000e+00
[2,] -4.510281e-17  1.000000e+00 1.387779e-17
[3,] -2.775558e-17 -2.220446e-16 1.000000e+00
> zapsmall(  A %*% solve(A) )
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

Manipulating Diagonal Matrices

Extracting Diagonal Elements

- In many situations in multivariate statistics, we need to perform operations involving the diagonal elements of a matrix, or diagonal matrices, or both.
- R has a surprisingly versatile function, `diag`, that can perform several of the most important operations.
- Consider the symmetric correlation matrix defined below:

```
> Rxx <- matrix(c(1.0, 0.5, 0.4,
+                 0.5, 1.0, 0.3,
+                 0.4, 0.3, 1.0
+                 ),3,3)
> Rxx

     [,1] [,2] [,3]
[1,]  1.0  0.5  0.4
[2,]  0.5  1.0  0.3
[3,]  0.4  0.3  1.0
```

Manipulating Diagonal Matrices

Extracting Diagonal Elements

- Suppose we wished to extract the diagonal entries of $\mathbf{R_xx}$.
- If the `diag` command is applied to a matrix, it extracts the diagonal entries in a vector.

```
> diag(Rxx)

[1] 1 1 1
```

- On the other hand, if you apply the `diag` function to a vector, the result is a diagonal matrix with diagonal entries equal to the elements of the vector.

```
> d<- diag(Rxx)
> diag(d)

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

Manipulating Diagonal Matrices

Extracting the Diagonal into a Diagonal Matrix

- On several occasions we will want to extract the diagonal entries of a matrix, and creat a diagonal matrix composed of those elements.
- This can be accomplished directly as follows:

```
> D <- diag(diag(Rxx))
> D
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

Manipulating Diagonal Matrices

Extracting the Diagonal into a Diagonal Matrix

- An odd but useful variation on the `diag` command allows one to create an identity matrix of any order.
- To create a $p \times p$ identity matrix, simply enter the integer p as input to the `diag` function, as demonstrated below.

```
> diag(4)
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
```