Introduction to computational programming

Appendix 3

Matrix review and use in *R*

David M. Rosenberg
University of Chicago
Committee on Neurobiology

Version control information:
Last changed date: 2009-11-09 15:10:33 -0600 (Mon, 09 Nov 2009)
Last changes revision:213
Version: Revision 213
Last changed by: David M. Rosenberg

November 10, 2009

Overview

This guide is intended to serve as a review of matrices and linear algebra and an introduction to their use in *R*.

1 Matrix review

1.1 Linear systems of equations

Consider the following system of equations.

$$\begin{aligned} 3a + 2b &= -1 \\ a - b &= 3 \end{aligned}$$

A more compact form of this system can be written (by omitting everything but the coefficients) as

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

This matrix, A, would be referred to as a 2×3 matrix, since it has 2 rows and three columns. It is easy to solve a system of linear equations in this form (see **??**).

1.2 Special matrices

There are several *special* types of matrices with which you should be familiar.

- **Square Matrix -** A square matrix is a matrix with an equal number of rows and columns.
- **Augmented square matrix -** A matrix with n rows and $n+1$ columns (such as the first example shown above). This form is often used to represent a simultaneous system of linear equations.
- **Identity matrix -** The identity matrix (abbreviated I_m) is an $m \times m$ matrix that contains only zeros and ones with ones on the top-left to bottom-right diagonal (this is called the *main diagonal*). The following are examples of identity matrices

$$I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  Identity matrices are special in that $\forall\ k \times k$ square matrices A, $AI_k = A$ (more on matrix multiplication later).
- **Diagonal matrix -** A diagonal matrix is a square matrix for which all entries not on the main diagonal are zero.
- **Upper/lower triangular matrix -** An *upper triangular matrix* is a square $k \times k$ matrix where $\forall m, n \in \mathbb{Z} : m > n \quad a_{mn} = 0$. Similarly a *lower triangular matrix* is a square $k \times k$ matrix for which $\forall m, n \in \mathbb{Z} : m < n \quad a_{mn} = 0$.

$$\begin{bmatrix} 1 & 5 & 16 \\ 0 & 2 & -5 \\ 0 & 0 & 2 \end{bmatrix} \qquad \begin{bmatrix} 0 & 0 & -3 \\ 0 & 1 & -1 \\ 14 & 3 & 5 \end{bmatrix}$$

- **Row-echelon form -** A matrix in *row-echelon* form is an augmented square matrix where $\forall m, n \in \mathbb{Z} : m > n \quad a_{mn} = 0$ and $\forall m, n \in \mathbb{Z} : m = n \quad a_{mn} = 1$.

$$\begin{bmatrix} 1 & 3 & 2 & 1 \\ 0 & 1 & -1 & 9 \\ 0 & 0 & 1 & -14 \end{bmatrix}$$

- **Reduced row-echelon form -** Reduced row-echelon form is a subset of row-echelon for with the additional constraint that, (for a $k \times k + 1$ matrix) $\forall m, n \in \mathbb{Z} : m \neq n$ and $n \neq k+1 \quad a_{mn} = 0$.

$$\begin{bmatrix} 1 & 0 & 0 & 16 \\ 0 & 1 & 0 & -5 \\ 0 & 0 & 1 & -14 \end{bmatrix}$$

1.3 Operations on matrices

There are five "basic" operations which can be performed on matrices.

1. **Scalar addition** A scalar can be added to any matrix. The result is the matrix of scalar by element sums.
$$3 + \begin{bmatrix} 1 & 3 \\ 5 & 7 \end{bmatrix} = \begin{bmatrix} 4 & 6 \\ 8 & 10 \end{bmatrix}$$

2. **Matrix addition** Two matrices can be added if and only if they are of equivalent dimensions. The sum of two matrices is the matrix of elementwise sums.
$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix} = \begin{bmatrix} 1+2 & 2+4 \\ 3+6 & 4+8 \end{bmatrix} = \begin{bmatrix} 3 & 6 \\ 9 & 12 \end{bmatrix}$$

3. **Scalar multiplication** Any matrix can be multiplied by a scalar to yield a matrix of scalar by element products.
$$3 \cdot \begin{bmatrix} 1 & 3 \\ 5 & 7 \end{bmatrix} = \begin{bmatrix} 3 & 9 \\ 15 & 21 \end{bmatrix}$$

4. **Matrix multiplication** A $m \times n$ matrix A and a $n \times l$ matrix B can be multiplied together to yield a $m \times l$ matrix as follows:
$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix} = \begin{bmatrix} 1 \cdot 2 + 2 \cdot 6 & 1 \cdot 4 + 2 \cdot 8 \\ 3 \cdot 2 + 4 \cdot 6 & 3 \cdot 4 + 4 \cdot 8 \end{bmatrix} = \begin{bmatrix} 14 & 18 \\ 24 & 48 \end{bmatrix}$$

5. **Determinant** The *determinant* of a matrix A is the sum the products of the "top-left" to "bottom-right" diagonals minus the products of the the "top-right" to "bottom-left" diagonals. For a 2×2 matrix
$$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$$
$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - bdi - afh$$

1.4 Matrix transformations

There are three basic "elementary row operations" for modifying a (square) matrix.

- **Row-switching operations** Swap two rows of a matrix. This reverses the sign of the matrix determinant.
$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \Rightarrow \begin{bmatrix} a & b & c \\ g & h & i \\ d & e & f \end{bmatrix}$$

- **Row-multiplication operations** Multiply a row of a matrix by a constant. The effect of multiplying a single row of A by k causes the determinant of A to increase by a factor of k as well. $k|A| = |A'|$.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \Rightarrow \begin{bmatrix} a & b & c \\ kg & kh & ki \\ d & e & f \end{bmatrix}$$

- **Linear combinations of rows -** Given a matrix A with row vectors $a_m = (a_{m1}, a_{m2}, \ldots)$ and $a_n = (a_{n1}, a_{n2}, \ldots)$, one of rows m and n can be replaced with a linear combination of a_m and a_n. This does not change the determinant of the matrix.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \Rightarrow \begin{bmatrix} a & b & c \\ 3a-g & 3b-h & 3c-i \\ d & e & f \end{bmatrix}$$

Basic row operations can be used to transform a matrix into row-(echelon) form. For an augmented square matrix representing a set of simultaneous linear equations, this corresponds to solving the linear system by *Gauss-Jordan* elimination.

1.5 Matrix differential equations

Consider the system of differential equations

$$\frac{dy}{dx} = 3y - 4z$$
$$\frac{dz}{dx} = 4y - 7z$$

with initial conditions $y(0) = 1$ and $z(0) = 3$

$$\begin{aligned} \begin{bmatrix} y'(x) \\ z'(x) \end{bmatrix} &= \begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} \\ \det\left(\begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) &= 0 \\ (3-\lambda)(-7-\lambda) + 16 &= 0 \\ \lambda^2 + 4\lambda - 5 &= 0 \\ \lambda &\in \{1, -5\} \end{aligned}$$

$$\begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda_1 \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$
$$\begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$
$$\alpha = 2\beta$$
$$\begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda_2 \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$
$$\begin{bmatrix} 3 & -4 \\ 4 & -7 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = -5 \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$
$$2\alpha = \beta$$

$$\begin{bmatrix} y \\ z \end{bmatrix} = Ae^{\lambda_1 x} v_1 + Be^{\lambda_2 x} v_2$$
$$y = 2Ae^{x} + Be^{-5x}$$
$$z = Ae^{x} + 2Be^{-5x}$$
$$1 = 2A + B$$
$$3 = A + 2B$$
$$B = \frac{5}{3}$$
$$A = -\frac{1}{3}$$
$$y = -\frac{2}{3}e^{x} + \frac{5}{3}e^{-5x}$$
$$z = -\frac{2}{3}e^{x} + \frac{10}{3}e^{-5x}$$

Solving this system using sage:

```
x = var('x')
y = function('y', x)
z = function('z', x)
DE1 = diff(y, x) == 3 * y - 4 * z
DE2 = diff(z, x) == 4 * y - 7 * z
iVals = [0, 1, 3]
desolve_system([DE1, DE2], [y, z], iVals)
```

$$\left[y\left(x\right) = \frac{5}{3}\,e^{-5\,x} - \frac{2}{3}\,e^{x}, z\left(x\right) = \frac{10}{3}\,e^{-5\,x} - \frac{1}{3}\,e^{x}\right]$$

2 Creating matrices in *R*

1. `matrix()` - matrices are created in *R* using the `matrix()` function. The number of rows (or columns) can be specified using the `nrow=` and `ncol=` arguments. By default, values created as such are placed in the matrix in *column-major* format; you can change this with the `byrow=TRUE` parameter.

```
> M <- matrix(1:9, nrow=3);
> N <- matrix(1:9, nrow=3, byrow=TRUE);
> M

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

> N

     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9

> vec <- sample(1:100, 20);
> vec

 [1] 59 19  7 53 17 50 41 80 66 55 60 62 92 88 77 15 58 29  6 72

> O <- matrix(vec, nrow=4, byrow=TRUE);
> O

     [,1] [,2] [,3] [,4] [,5]
[1,]   59   19    7   53   17
[2,]   50   41   80   66   55
[3,]   60   62   92   88   77
[4,]   15   58   29    6   72

> O[1,];

[1] 59 19  7 53 17

> O[,2];

[1] 19 41 62 58

> O[3,4];

[1] 88
```

Using sage:

```
A = matrix(QQ, [[1, 2, -3], [4 ,-5, -6], [7, 8, 19]])
A
```

$$\begin{pmatrix} 1 & 2 & -3 \\ 4 & -5 & -6 \\ 7 & 8 & 19 \end{pmatrix}$$

2. `diag()` - Diagonal matrices can be created using the `diag()` function. Identity matrices can be created using the function as well.

```
> I2 <- diag(1, nrow=2);
> I2;

     [,1] [,2]
[1,]    1    0
[2,]    0    1

> I5 <- diag(1, nrow=5);
> I5;

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1

> diag(c(1, 2, 4), nrow=3);

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    2    0
[3,]    0    0    4
```

3. `cbind()` - matrices can be “joined” columnwise using the `cbind()` command.
4. `rbind()` - matrices can be “joined” rowwise using the `rbind()` command.

```
> cbind(M, N);

     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    4    7    1    2    3
[2,]    2    5    8    4    5    6
[3,]    3    6    9    7    8    9

> rbind(M, N);

     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
[4,]    1    2    3
[5,]    4    5    6
[6,]    7    8    9
```

2.1 Basic manipulation

1. `det()` Calculate the determinant of a matrix.

```
> set.seed(12345);
> M <- matrix(sample((1:10) - 5, 9), nrow=3);
> M
```

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

> det(M);

[1] 2
```

```
A.determinant()
```

-484

2. `+` `-` Add either a scalar to a matrix *or* add two matrices together. *WARNING:*

```
> N <- matrix(sample((1:10) - 5, 9), nrow=3);
> N;

     [,1] [,2] [,3]
[1,]    5    1    0
[2,]   -4    4   -1
[3,]   -3    3    2

> 3 + N;

     [,1] [,2] [,3]
[1,]    8    4    3
[2,]   -1    7    2
[3,]    0    6    5

> M + N;     ## Probably not what you want

     [,1] [,2] [,3]
[1,]    8    5   -3
[2,]    1    2   -2
[3,]   -1   -1    3

> matrix(M + N, nrow=3, byrow=TRUE);

     [,1] [,2] [,3]
[1,]    8    1   -1
[2,]    5    2   -1
[3,]   -3   -2    3
```

```
3 * A
```

$$\begin{pmatrix} 3 & 6 & -9 \\ 12 & -15 & -18 \\ 21 & 24 & 57 \end{pmatrix}$$

3. `%*%` - Calculate the matrix product of two matrices.

```
> M %*% N;
```

```
     [,1] [,2] [,3]
[1,]    8   10  -10
[2,]   36   -6    0
[3,]   23  -11    6
```

```
> N %*% M;
```

```
     [,1] [,2] [,3]
[1,]   20   18  -16
[2,]    6  -20    7
[3,]   10  -26    8
```

```
B = matrix(QQ, [[1,3, -5],[2, -4, -16], [1, -1, 11]])
A * B;
```

$$\begin{pmatrix} 2 & -2 & -70 \\ -12 & 38 & -6 \\ 42 & -30 & 46 \end{pmatrix}$$

4. `%/%` - Inverse matrix product.

```
> O <- N %/% M;
> O;
```

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

```
> O %*% M;
```

```
     [,1] [,2] [,3]
[1,]    3    4   -3
[2,]  -11   -4    6
[3,]   -7  -14    9
```

5. `t()` - Transpose a matrix.

```
> M;
```

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

```
>  t(M);
```

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

```
A.transpose()
```

$$\begin{pmatrix} 1 & 4 & 7 \\ 2 & -5 & 8 \\ -3 & -6 & 19 \end{pmatrix}$$

6. `library(Matrix)` - The `library()` command loads an *R* package - a collection of functions and other *R* objects. The *Matrix* package contains a large number of functions for the efficient manipulation of matrices.

2.2 Letting *R* do the heavy lifting

Using *R* invert a matrix

Not all matrices have an inverse. A matrix is *invertible* if and only if the determinant of the matrix is nonzero. The inverse of a square matrix A is the matrix A^{-1} such that $AA^{-1} = I$. *R* can calculate the inverse of a matrix as follows.

```
> A <- matrix(c(1, 5, 2, 8), nrow=2, byrow=TRUE);
> Ainv <- solve(A);
> A

     [,1] [,2]
[1,]    1    5
[2,]    2    8

> Ainv

     [,1] [,2]
[1,]   -4  2.5
[2,]    1 -0.5

> A %*% Ainv

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

Using sage instead

```
A = matrix(QQ, [[1, 5], [2, 8]]);
A.inverse();
```

$$\begin{pmatrix} -4 & \frac{5}{2} \\ 1 & -\frac{1}{2} \end{pmatrix}$$

Similarly, the matrix equation $\mathbf{A} \cdot X = B$ for a given matrix A and column vector B can be solved for vector X using *R* as shown below. This method fails when the matrix is not invertible.

```
> A
```

```
      [,1] [,2]
[1,]     1    5
[2,]     2    8

> B <- matrix(c(3,4), nrow=2);
> x <- solve(A, B);
> x;

      [,1]
[1,]    -2
[2,]     1

> A %*% x;

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

Using sage ...

```
B = matrix(QQ, [[3], [4]])
A.solve_right(B)
```

$$\begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

This corresponds to solution to the simultaneous system of linear equations described in the beginning. Moreover, the "failure" which occurs when the determinant is 0 (i.e. A is not invertible) implies that the equation lacks a unique solution (i.e. the system of equations is underdetermined).