Matrix Arithmetic

Statistics 427: R Programming

Module 11

2020

Other ways to multiply vectors

If `x` and `y` are vectors (of numbers), then `R` defines the product of `x*y` as the vector containing the elements of `x` each multiplied by the corresponding elements in vector `y`. This operation is called **elementwise** multiplication. Another kind of multiplication for vectors is called the *dot product* or *scalar product* that is useful in many applications in science and commerce.

Notation for vectors and matrices will be bold font **C**. When doing these calculations by hand, the **C** can be used for notation because trying to write a bold letter by hand does not work well. For lectures and this course, bold font **C** will be used to denote vectors and matrices.

Example with the usual methods

Suppose a population of a small mammal species is represented by the vector $\mathbf{n} = (\mathbf{n_1}, \mathbf{n_2}, \mathbf{n_3}, \mathbf{n_4})$, containing, respectively, the numbers of 0-year-olds, 1-year-olds, 2-year-olds, and animals 3 years old or older. Suppose the vector $\mathbf{f} = (\mathbf{f_1}, \mathbf{f_2}, \mathbf{f_3}, \mathbf{f_4})$ contains the average number of offspring born in the population per year to an average 0-year-old, 1-year-old, 2-year-old, 3-or-more-year-old, respectively. We will use **bolded** letters **n** and **f** to denote vectors in mathematical formulas. The average total number of offspring born in the population in a year is:

$$\text{total offspring} = \sum n_i f_i = n_1 f_1 + n_2 f_2 + n_3 f_3 + n_4 f_4$$

Example in `R`

The following result is a single number or a *scalar*. The calculation represented by `sum(n*f)` is the *dot* or *scalar* product of two vectors `n` and `f` of identical lengths.

```
n=c(49,36,28,22)
f=c(0,.2,2.3,3.8)
total.offspring=sum(n*f)
total.offspring
```

```
[1] 155.2
```

Dot product (scalar product)

In general if $\mathbf{x} = (\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_k})$ and $\mathbf{y} = (\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_k})$ are vectors with equal length, the symbol for a dot product is a centered dot and the general definition of the dot product is the scalar number resulting from the following:

$$\mathbf{x} \bullet \mathbf{y} = \mathbf{x_1 y_1} + \mathbf{x_2 y_2} + \cdots + \mathbf{x_k y_k}$$

The operator in R for the dot product is %*% (yes, with the percent symbols).

Example with dot product

```
n%*%f
```

```
      [,1]
[1,] 155.2
```

Uses of dot product I

If a vector $\mathbf{x}$ contains the number of different models of automobiles produced by a manufacturer and the vector $\mathbf{y}$ contains the profit per automobile for each model, then $\mathbf{x} \bullet \mathbf{y}$ gives the total profit for all models produced.

Uses of dot product II

Or if $\mathbf{a}$ is the vector of quantities that are recorded of a house and $\mathbf{b}$ contains the average contribution to the value of the house by a unit of each quantity in $\mathbf{a}$, then $\mathbf{b} \bullet \mathbf{a}$ is the estimated sale (appraisal) value of the house.

Uses of dot product III

In geometry, if $\mathbf{r}$ contains coordinates of a point in two dimensions, three dimensions, or any number of higher dimensions, and $\mathbf{s}$ contains the coordinates of another such point in the same coordinate system, and θ is the angle between the two line segments drawn from the origin to the points represented by $\mathbf{r}$ and $\mathbf{s}$, then

$$\cos\theta = \frac{\mathbf{r} \bullet \mathbf{s}}{\left(\sqrt{\mathbf{r} \bullet \mathbf{r}}\right)\left(\sqrt{\mathbf{s} \bullet \mathbf{s}}\right)}$$

Uses of dot product IV

```
cos(pi/4)
```

```
[1] 0.7071068
```

```
r=c(2,2) # line segments from (0,0) to r and to s
s=c(2,0) # form an angle pi/4 (45 deg)
(r%*%s)/(sqrt(r%*%r)*sqrt(s%*%s))
```

```
          [,1]
[1,] 0.7071068
```

Uses of dot product V

There is frequent occasion to perform and keep track of many dot products. For instance, let p_1 be the average yearly survival probability for 0-year-olds in the mammal population from earlier, and let p_2, p_3, and p_4 be the respective annual survival probabilities of 1-year-olds, 2-year-olds, and animals 3 years old or older. Define the vectors $\mathbf{p_1} = (\mathbf{p_1}, \mathbf{0}, \mathbf{0}, \mathbf{0})$, $\mathbf{p_2} = (\mathbf{0}, \mathbf{p_2}, \mathbf{0}, \mathbf{0})$, and $\mathbf{p_3} = (\mathbf{0}, \mathbf{0}, \mathbf{p_3}, \mathbf{p_4})$. Then after a year has passed, $\mathbf{p_1} \bullet \mathbf{n}$ $(= p_1 n_1)$ is the number of 1-year-olds, $\mathbf{p_2} \bullet \mathbf{n}$ is the number of 2-year-olds, and $\mathbf{p_3} \bullet \mathbf{n}$ is the number of animals 3 years or older. Use all of that with the original calculations we did and the four dot products for projecting the population in 1 year become the components for an operation called matrix multiplication.

Matrix Multiplication

A **matrix** is a rectangular array of numbers. Matrices are simple but some matrix operations are a bit unintuitive at first. In a way, matrix multiplication is a way to keep track of many dot products of vectors. We can make matrices from vectors with a few different methods. First up is `rbind()` and `cbind()` to bind vectors together in a matrix either by rows or columns, respectively.

General forms of `rbind()` and `cbind()`

`rbind(...,)` and `cbind(...,)`
`...`: generalized vectors or matrices with arguments separated by commas

`rbind()` and `cbind()` example

```
x1=c(3:6)
x2=c(10:13)
x3=c(-1:-4)
a=rbind(x1,x2,x3); a
```

```
   [,1] [,2] [,3] [,4]
x1    3    4    5    6
x2   10   11   12   13
x3   -1   -2   -3   -4
```

```
b=cbind(x1,x2,x3); b
```

```
     x1 x2 x3
[1,]  3 10 -1
[2,]  4 11 -2
[3,]  5 12 -3
[4,]  6 13 -4
```

Matrices I

The matrix **a** defined in the previous `R` example has three rows and four columns (therefore **a** is a 3×4 matrix), whereas the matrix **b** is a 4×2 matrix. the rows of **a** and the columns of **b** were labeled with the original vector names. However, matrix elements are actually referenced by their row and column numbers. With `R`, the elements can be selected out of a matrix using their row and column numbers using the index method (square brackets `[r,c]`).

Index method review I

```
a[2,3]
```

```
x2
12
```

```
b[4,3]
```

```
x3
-4
```

```r
a[2,3]+b[4,3]
```

```
x2
 8
```

Index method review II

```r
a[1,2:4]
```

```
[1] 4 5 6
```

```r
b[c(1,3),1:3]
```

```
     x1 x2 x3
[1,]  3 10 -1
[2,]  5 12 -3
```

```r
a[2,]
```

```
[1] 10 11 12 13
```

Matrices II

In R, a matrix differs from a data frame in that a matrix can only contain numerical elements, while a data frame can have categorical or numerical data.

The *matrix product* **AB** of a matrix **A** $(\mathbf{l} \times \mathbf{m})$ and a matrix **B** $(\mathbf{m} \times \mathbf{n})$ is defined if the number of the columns of **A** equals the number of rows of **B**. Think of **A** as a "stack" of vectors, each with m elements, in the form of rows:

$$\mathbf{A} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

Matrices III

Think of the second matrix in the product as a line of vectors each with m elements in the form of columns:

$$\mathbf{B} = \begin{bmatrix} b_1 & b_2 & \dots & b_n \end{bmatrix}$$

Matrices IV

The matrix **AB** is a matrix with l rows and n columns, consisting of all the pairwise dot products of the vectors of **A** and **B**. In other ways, the element in the ith row and the jth column of **AB** is the dot product of the ith row of **A** and the jth column of **B**.

$$\mathbf{AB} = \begin{bmatrix} a_1 \bullet b_1 & a_1 \bullet b_2 & \cdots & a_1 \bullet b_n \\ a_2 \bullet b_1 & a_2 \bullet b_2 & \cdots & a_2 \bullet b_n \\ \vdots & \vdots & \ddots & \vdots \\ a_l \bullet b_1 & a_l \bullet b_2 & \cdots & a_l \bullet b_n \end{bmatrix}$$

Matrices V

In general, with matrix algebra, the commutative property **does not hold**. That is $\mathbf{AB} \neq \mathbf{BA}$. The product $\mathbf{BA}$ could be a different-sized matrix or could be undefined if the rows of $\mathbf{B}$ and columns of $\mathbf{A}$ do not match.

Matrices VI

`R` handles matrix calculations well (of course it does!)

```
c=a%*%b; c
```

```
    x1   x2   x3
x1  86  212  -50
x2 212  534 -120
x3 -50 -120   30
```

```
d=b%*%a; d
```

```
     [,1] [,2] [,3] [,4]
[1,]  110  124  138  152
[2,]  124  141  158  175
[3,]  138  158  178  198
[4,]  152  175  198  221
```

Matrices VII

When discussing a matrix product $\mathbf{AB}$, saying that $\mathbf{A}$ is "postmultiplied" by $\mathbf{B}$ and that $\mathbf{B}$ is "premultiplied" by $\mathbf{A}$ helps to avoid confusion.

The associative property *does* hold in matrix multiplication.

$$\mathbf{ABC} = (\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$$

Additionally, when a matrix only has one row or one column, it is called, respectively, a *row vector* or a *column vector*

Matrices VIII

Scalar multiplication of a matrix is defined as a matrix, $\mathbf{A}$, multiplied by a scalar number, say x; the operation is denoted as $\mathbf{Ax}$ or $x\mathbf{A}$ and results in a matrix containing all the elements of $\mathbf{A}$, each individually multiplied by x. In `R`, this is done with the plain multiplication operation `*` (no % signs). Scalar multiplication **is** commutative.

Matrices IX

```
x=2
x*a
```

```
   [,1] [,2] [,3] [,4]
x1    6    8   10   12
x2   20   22   24   26
x3   -2   -4   -6   -8
```

Matrix addition and subtraction

Unlike matrix multiplication, *matrix addition* is defined elementwise, as we intuitively suppose it should be. In matrix addition, each element is defined for only two matrices with the same dimensions (same number of rows and columns). *Matrix subtraction* is similar to matrix addition, subtracting elements of one matrix from the corresponding elements in another matrix. The usual plus `+` and minus `-` signs in `R` work here.

Addition and subtraction example I

```
k=1:10
a=matrix(k,2,5); a
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

```
j=c(1,2)
b=matrix(j,2,5); b
```

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

Addition and subtraction example II

```
a+b
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    2    4    6    8   10
[2,]    4    6    8   10   12
```

```
a-b
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    0    2    4    6    8
[2,]    0    2    4    6    8
```

Reading a data file into a matrix

A file of data can be read into a matrix with the `matrix()` function. Suppose I have a file called `C:/Docs/mydata.txt` on my computer. If the file has only numeric data, the following statement would read the data into a matrix:

```
x=matrix(scan('C:/Docs/mydata.txt'),nrow=6,ncol=8,
byrow=T)
matrix(x,r,c,byrow=F)
```

`x`: object (data frame or vector)
`r`, `c`: number of rows and columns, respectively
`byrow=F`: F (default) means matrix is filled by columns, otherwise (T) by rows

`scan(file='',...)`
`file`: name of file to read data values from
`...`: more options

Wildlife population example I

There are three age classes of animals in a population: juveniles (< 1 yr), subadults (nonbreeding animals $1 - 2$ yrs), and breeding adults (≥ 2 yrs). The number of juveniles, subadults, and adults in the population at time t were denoted, respectively, as J_t, S_t, A_t. These age classes were projected one time unit (year) into the future with three equations:

$$J_{t+1} = fA_t$$

$$S_{t+1} = p_1 J_t$$

$$A_{t+1} = p_2 S_t + p_3 A_t$$

Wildlife population example II

The values p_1,p_2, and p_3 are the annual survival probabilities for individuals in the three age classes, and f is the average annual number of offspring produced by each adult (fecundity). Rewritten as dot products is

$J_{t+1} = 0J_t + 0S_t + fA_t$, which is the dot product of $(0, 0, f)$ and (J_t, S_t, A_t)
$S_{t+1} = p_1 J_t + 0S_t + 0A_t$, which is the dot product of $(p_1, 0, 0)$ and (J_t, S_t, A_t)
$A_{t+1} = 0J_t + p_2 S_t + p_3 A_t$, which is the dot product of $(0, p_2, p_3)$ and (J_t, S_t, A_t)

Wildlife population example III

Enter into matrices (well, one vector and one matrix… but… semantics)

$$\mathbf{n_t} = \begin{bmatrix} J_t \\ S_t \\ A_t \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & f \\ p_1 & 0 & 0 \\ 0 & p_2 & p_3 \end{bmatrix}$$

The column vector $\mathbf{n_{t+1}}$ of next year's age classes is found by matrix multiplication.

$$\mathbf{n_{t+1}} = \mathbf{M}\mathbf{n_t}$$

Wildlife population example IV

The Northern Spotted Owls have age class survival probabilities as follows: $p_1 = 0.11$, $p_2 = 0.71$, and $p_3 = 0.94$. Fecundity (average annual number of offspring produced by each adult), $f = 0.24$.

Entering the data into `R`, `rbind()` will be used to bind the vectors together by rows to create the matrix. Additionally, `n.time` and `n.ages` will be created as well; `n.time`= 20 years and `n.ages`= 3 for the three age classes.

```
p1=.11; p2=.71; p3=.94; f=.24
n.time=20; n.ages=3
M=rbind(c(0,0,f),c(p1,0,0),c(0,p2,p3))
N=matrix(0,n.time,n.ages)
```

Wildlife population example V

The initial starting values of the population age classes are the first row of **N**. Next is a loop to calculate for all twenty years.

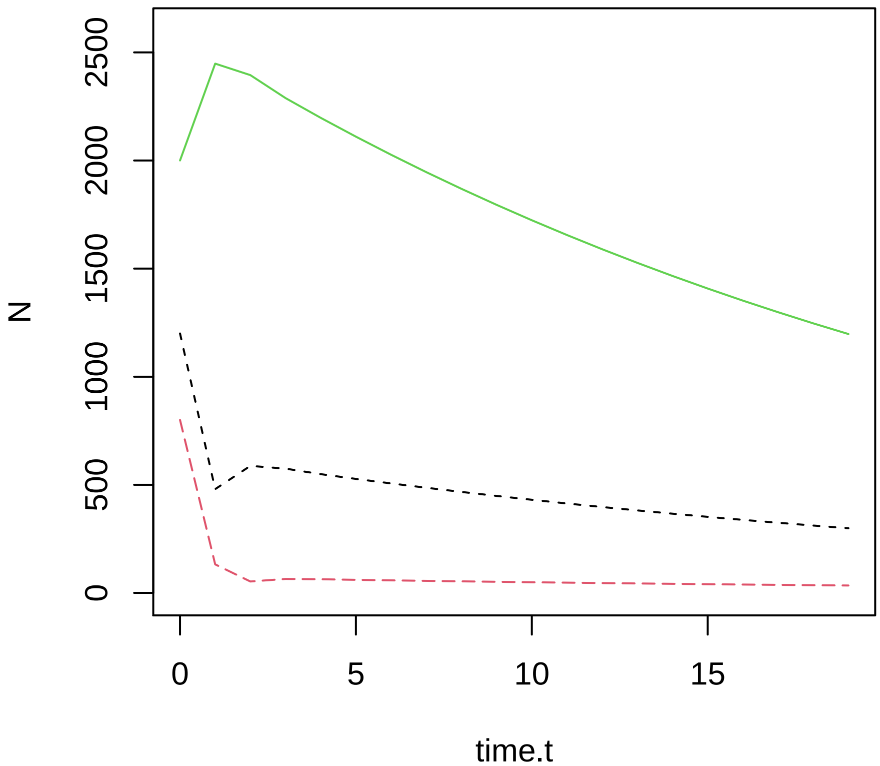
```
N[1,]=c(1200,800,2000)

for(t in 1:(n.time-1)){
  N[t+1,]=M%*%N[t,]
}
N
```

```
           [,1]      [,2]     [,3]
 [1,] 1200.0000 800.00000 2000.000
 [2,]  480.0000 132.00000 2448.000
 [3,]  587.5200  52.80000 2394.840
 [4,]  574.7616  64.62720 2288.638
 [5,]  549.2730  63.22378 2197.205
 [6,]  527.3291  60.42003 2110.261
 [7,]  506.4627  58.00620 2026.544
 [8,]  486.3705  55.71090 1946.136
 [9,]  467.0725  53.50076 1868.922
[10,]  448.5413  51.37798 1794.772
[11,]  430.7454  49.33955 1723.564
[12,]  413.6555  47.38199 1655.182
[13,]  397.2436  45.50210 1589.512
[14,]  381.4829  43.69679 1526.448
[15,]  366.3475  41.96312 1465.886
[16,]  351.8125  40.29822 1407.726
[17,]  337.8543  38.69938 1351.874
[18,]  324.4499  37.16397 1298.239
[19,]  311.5772  35.68948 1246.731
[20,]  299.2153  34.27350 1197.266
```

Wildlife population example VI

```
time.t=0:(n.time-1)
matplot(time.t,N,type='l',lty=c(2,5,1),ylim=c(0,2600))
```



The Matrix

Red or blue?