Matrices

João Vieira & Pedro Fonseca

Introduction to R Programming

April 2, 2020

What is a Matrix?

- A matrix with m rows and n columns is defined as:

$$A_{m,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

The *matrix* function

- To create a matrix use the *matrix* function:

  ```
  > A <- matrix(data, nrow, ncol, byrow)
  ```

- Only the *data* argument is mandatory.
- Provide the entries of the matrix as a vector to the data argument.
- With *nrow* you can provide the number of row and with *ncol* the number of columns.
- By default, *matrix* returns a lenght(data) $\times$ 1 matrix.

The *matrix* function: examples


```
> A <- matrix(data = c(1, 2, 3 ,4 ,5 , 6), nrow = 2,
ncol = 3)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> b <- c(1, 5 ,5, 7)  # Remark: R is case sensitive!
> B <- matrix(b, nrow = 2, ncol = 2)
> B
     [,1] [,2]
[1,]    1    5
[2,]    5    7
```

The *byrow* argument

- *byrow* is a logical argument (TRUE or FALSE), set to FALSE by default.
- If FALSE the output matrix is filled by columns, otherwise the matrix is filled by rows.

The *byrow* argument: example


```
> A <- matrix(data = c(1, 2, 3 ,4 ,5 , 6), nrow = 2,
ncol = 3)
> A
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> B <- matrix(data = c(1, 2, 3 ,4 ,5 , 6), nrow = 2,
ncol = 3, byrow = TRUE)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Argument Names

- Remember: Argument names are optional:

```
> matrix(data = c(1, 2, 3 ,4 ,5 , 6), nrow = 2,
ncol = 3)

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> matrix(c(1, 2, 3 ,4 ,5 , 6),  2,  3)

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

Recycling

- R *recycles* the *data* vector if its length is not compatible with the number of elements of the output matrix:

```
> matrix(1:5, nrow = 2, ncol = 3, byrow = TRUE)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    1
```

rbind and *cbind*

- If you have multiple vectors of equal length, you create matrix by binding them together with the *rbind* and *cbind* functions:

```
> rbind(1:3, 4:6)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

rbind and *cbind*: example


```
> set.seed(123)

> a <- 1:4
> b <- seq(from = 1, by = 0.5, length.out = 4)
> c <- rpois(4, lambda = 2)

> rbind(a, b, c)
  [,1] [,2] [,3] [,4]
a    1  2.0    3  4.0
b    1  1.5    2  2.5
c    1  3.0    2  4.0
```

rbind and *cbind*: example


```
> mymatrix <- cbind(c(1,4), c(2,5), c(3,6))
> mymatrix
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Dimensions

- Number of columns:

```
> ncol(mymatrix)
[1] 3
```

- Number of rows:

```
> nrow(mymatrix)
[1] 2
```

- Both:

```
> dim(mymatrix)
[1] 2 3
```

```
> dim(mymatrix)[1]
[1] 2
```

- Matrix rows and columns can be named:

```
> M <- matrix(c(1, 5, 3, 7), ncol = 2, byrow = TRUE)

> rownames(M) <- c("x1", "x2")
> colnames(M) <- c("y1", "y2")

> M
   y1 y2
x1  1  5
x2  3  7
```

Subsetting

- The first column:

```
> mymatrix[, 1]
[1] 1 4
```

- The first row:

```
> mymatrix[1, ]
[1] 1 2 3
```

- A specific element:

```
> mymatrix[2, 3]
[1] 6
```

Subsetting

- The first and third columns:

```
> mymatrix[, c(1,3)]
     [,1] [,2]
[1,]    1    3
[2,]    4    6
```

- The first and second rows:

```
> mymatrix[c(1, 2), ]
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

- The elements in the diagonal of the matrix:

```
> diag(mymatrix)
[1] 1 5
```

Subsetting

- To omit elements from a matrix, use negative indexes:
  - The first column:

```
> mymatrix[, -1]
     [,1] [,2]
[1,]    2    3
[2,]    5    6
```

  - The first row:

```
> mymatrix[-1, ]
[1] 4 5 6
```

  - A specific row and column:

```
> mymatrix[-2, -3]
[1] 1 2
```

Subsetting

- To omit elements from a matrix, use negative indexes:
  - The first and third columns:

```
> mymatrix[, -c(1,3)]
[1] 2 5
```

  - The first and third rows:

```
> mymatrix[-c(1,2), ]
     [,1] [,2] [,3]
```

Subsetting

- In a matrix with named columns or vectors, we can use the names to subset:

```
> M
   y1 y2
x1  1  5
x2  3  7

> M[, "y1"]
x1 x2
 1  3

> M["x2", "y1"]
[1] 3
```

Substituting

- Overwriting elements of a matrix:
  - The first column:

```
> mymatrix

     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6

> mymatrix[, 1] <- c(5, 5)
> mymatrix

     [,1] [,2] [,3]
[1,]    5    2    3
[2,]    5    5    6
```

Substituting

- Overwriting elements of a matrix:
  - The first row:

```
> mymatrix

     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6

> mymatrix[1,] <- 30:32
> mymatrix
     [,1] [,2] [,3]
[1,]   30   31   32
[2,]    5    5    6
```

Substituting

- Overwriting a specific element in a matrix:

```
> mymatrix[2,3] <- 60
> mymatrix
     [,1] [,2] [,3]
[1,]   30   31   32
[2,]    5    5   60
```

- Overwriting the diagonal:

```
> diag(mymatrix) <- rep(x = 0, times = 2)
> mymatrix
     [,1] [,2] [,3]
[1,]    0   31   32
[2,]    5    0   60
```

Summation, subtraction and multiplication of scalars


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

> 2*A
     [,1] [,2]
[1,]    2    6
[2,]    4    8

> A + 5
     [,1] [,2]
[1,]    6    8
[2,]    7    9
```

Adding and subtracting matrices


```
> A <- matrix(1:4, nrow = 2)
> B <- matrix(rep(1, times = 4), ncol = 2)

> A + B
     [,1] [,2]
[1,]    2    4
[2,]    3    5

> A - B
     [,1] [,2]
[1,]    0    2
[2,]    1    3
```

Element-wise multiplication and division


```
> A <- matrix(1:4, nrow = 2)
> C <- matrix(rep(2, times = 4), ncol = 2)

> A * C     # This is not matrix multiplication!
     [,1] [,2]
[1,]    2    6
[2,]    4    8

> A / C
     [,1] [,2]
[1,]  0.5  1.5
[2,]  1.0  2.0
```

Matrix Algebra

- Transpose matrix:

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

> t(A)
     [,1] [,2]
[1,]    1    2
[2,]    3    4
```

Matrix Algebra

- Inverse matrix:

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

> solve(A)
     [,1] [,2]
[1,]   -2  1.5
[2,]    1 -0.5
```

Matrix Algebra


```
> A <- rbind(c(1, 3), c(2, 4))
> B <- matrix(1:4, ncol = 2, byrow = TRUE)

> A
     [,1] [,2]
[1,]    1    3
[2,]    2    4

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

Matrix Algebra

- Determinant of a matrix

```
> A <- matrix(c(-1, 1, 2,
                -5, 3, 4,
                10, 8, -4),
              ncol = 3,
              byrow = TRUE)

> det(A)
[1] -76
```

Matrix Algebra

- Matrix multiplication:

```
> A %*% B
     [,1] [,2]
[1,]   10   14
[2,]   14   20
```

- Matrix multiplication is not commutative:

```
> B %*% A
     [,1] [,2]
[1,]    5   11
[2,]   11   25
```

Systems of Linear Equations

- The *solve* function can also be used to solve systems of linear equations.
- *solve(A, B)* computes X from equation $AX = B$
  - A: matrix with the coefficients of the equations
  - B: vector or matrix of the equation's right side
  - X: vector or matrix of unknowns
- The default value of B is an identity matrix. This is why *solve*(A) gives A^{-1}:

$$AX = I \Leftrightarrow X = A^{-1} \tag{1}$$

Systems of Linear Equations

- Example 1:
  - $5x = 10$, what is x?

```
> solve(5,10)
[1] 2
```

Systems of Linear Equations

- Example 2: $\begin{cases} 3x + 2y &= 8 \\ x + y &= 2 \end{cases}$ what is (x, y)?

```
> A <- matrix(c(3,1,2,1), nrow = 2, ncol = 2)
> A
     [,1] [,2]
[1,]    3    2
[2,]    1    1

> b  <- c(8,  2)

> solve(A,b)
[1]  4 -2
```

- Conclusion: $x = 4$ and $y = -2$.

Systems of Linear Equations

- Alternative solution to example 2:
  - $Ax = b \Leftrightarrow x = A^{-1}b$

```
> solve(A,b)
[1]  4 -2

> solve(A) %*% b
     [,1]
[1,]    4
[2,]   -2
```

Questions?

"One whose knowledge is confined to books and whose wealth is in the possession of others, can use neither his knowledge nor wealth when the need for them arises."

— Chanakya