Lecture 1.7 – Matrices and Arrays

Specific Learning Objectives:

1.1.9 – Create vectors, arrays, matrices, lists, and data frames.

1.1.10 – Understand vectors and vectorized calculations.

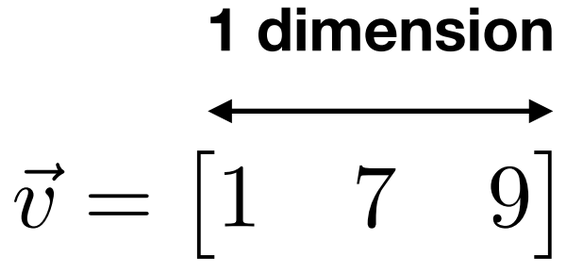
1.1.11 – Understand the data classes of R.

1.1.12 – Learn how to index vectors, arrays, matrices, lists, and data frames.

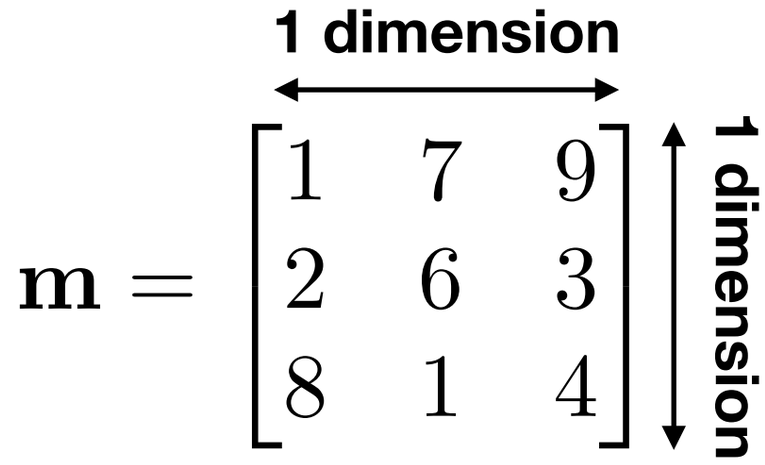
Question

- What's the difference between a vector, a matrix, and an array?

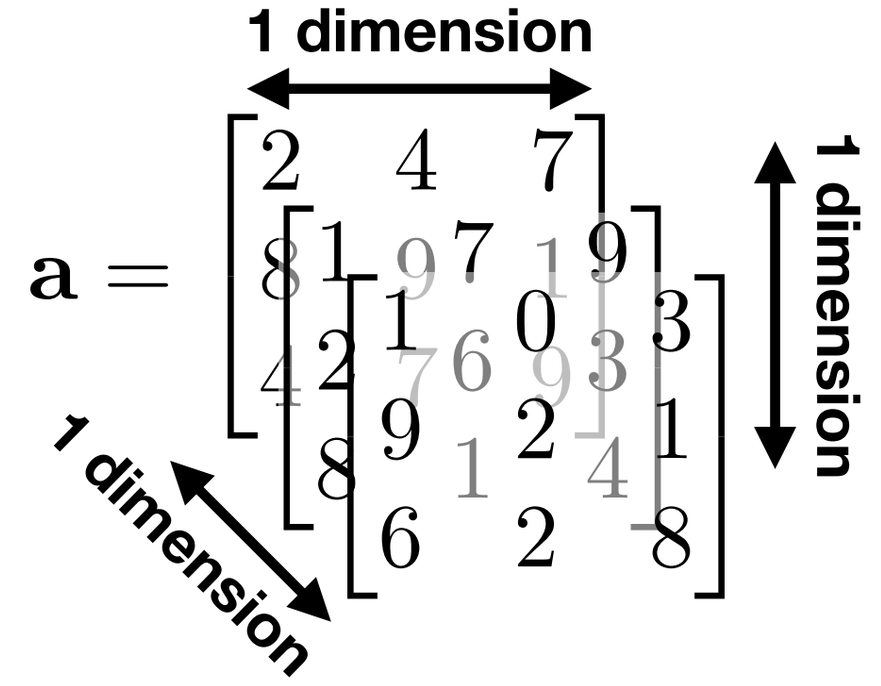
  - **Vectors** can be 1 or more elements in length, but must be 1-dimensional.



  - **Matrices** are 2-dimensional.



  - **Arrays** are *n*-dimensional.



Example: 3-D array

Matrix data

- **Matrices** are just several vectors stored together!

$$\vec{v} = \begin{bmatrix} 1 & 7 & 9 \end{bmatrix} \qquad \mathbf{m} = \begin{bmatrix} 1 & 7 & 9 \\ 2 & 6 & 3 \\ 8 & 1 & 4 \end{bmatrix}$$

vector 1 (v1)
vector 2 (v2)
vector 3 (v3)

- Create a matrix in R with **`matrix()`**:

```
> v1 <- c(1,7,9)
> v2 <- c(2,6,3)
> v3 <- c(8,1,4)
> m <- matrix(c(v1,v2,v3), nrow=3, byrow=TRUE)
> m
     [,1] [,2] [,3]
[1,]    1    7    9   = v1
[2,]    2    6    3   = v2
[3,]    8    1    4   = v3
```

Creating Matrices

- Creating matrices with **`matrix()`** provides many options!

```
> v1 <- c(1,7,9)
> v2 <- c(2,6,3)
> v3 <- c(8,1,4)
```

Number of rows the matrix should be

```
> m1 <- matrix(c(v1,v2,v3), nrow=3, byrow=TRUE)
> m1
     [,1] [,2] [,3]
[1,]    1    7    9
[2,]    2    6    3
[3,]    8    1    4
```

Construct the matrix by row (TRUE) or by column (FALSE)

```
> m2 <- matrix(c(v1,v2,v3), nrow=3, byrow=FALSE)
> m2
     [,1] [,2] [,3]
[1,]    1    2    8
[2,]    7    6    1
[3,]    9    3    4
```

By column is default behavior

Matrix dimensions

- Several functions help with determining the size of matrices.

```
> big.m <- cbind(m1,m2)
> big.m
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    7    9    1    2    8
[2,]    2    6    3    7    6    1
[3,]    8    1    4    9    3    4
```

- **`length()`** will give you the total number of elements

```
> length(big.m)
[1] 18
```

- **`dim()`** will give you the dimensions of the matrix:

```
> dim(big.m)
[1] 3 6
```

```
> dim(big.m)[2]
[1] 6
```

- **`nrow()`** will give you the number of rows:

```
> nrow(big.m)
[1] 3
```

- **`ncol()`** will give you the number of columns:

```
> ncol(big.m)
[1] 6
```

Check Your Understanding

You have four vectors in your environment:

```
v1 <- c(1, 1, 1)        v3 <- c(3, 3, 3)

v2 <- rep(2, 3)         v4 <- rep(4, 3)
```

You run the following line to create a matrix:

```
my.matrix <- matrix(c(v1,v2,v3,v4), nrow=4, byrow=FALSE)
```

Which describes the output matrix `my.matrix`?

a)

```
> matrix(c(v1,v2,v3,v4),nrow=3,byrow=FALSE)
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    1    2    3    4
[3,]    1    2    3    4
```

c)

```
> matrix(c(v1,v2,v3,v4),nrow=3,byrow=TRUE)
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    2
[2,]    2    2    3    3
[3,]    3    4    4    4
```

Correct

b)

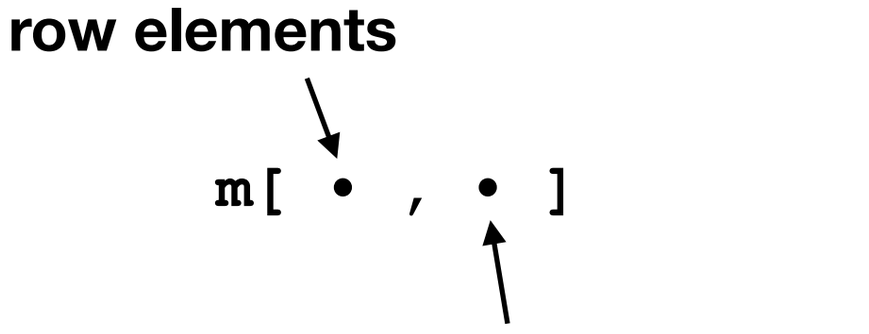
```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    2    4
[3,]    1    3    4
[4,]    2    3    4
```

d)

```
> matrix(c(v1,v2,v3,v4),nrow=4,byrow=TRUE)
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    2    2    2
[3,]    3    3    3
[4,]    4    4    4
```

Indexing a Matrix

- **Indexing a matrix:** use the position of both row and column to pick out an element.



(Leave column spot empty to grab everything in that column)

Indexing by row: `m[1, ]`

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

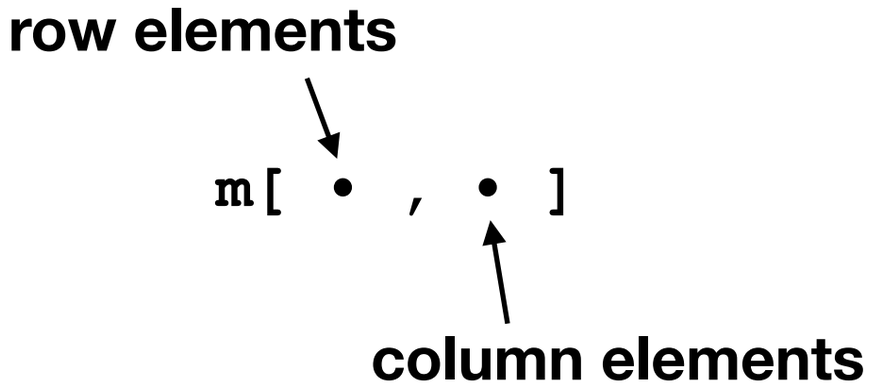
`m[1, ]` = **v1**

`m[2, ]` = **v2**

`m[3, ]` = **v3**

Indexing a Matrix

- **Indexing a matrix:** use the position of both row and column to pick out an element.



Indexing by column: `m[ ,1]`

What are `m[ ,1]`, `m[ ,2]`, and `m[ ,3]`?

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

Indexing a single element:

`m[ 2, 3] =`

3

Subsetting Matrices

- Similar to vectors, you can subset info in a matrix by specifying which rows and/or columns you want.

```
> big.m <- cbind(m1,m2)
> big.m
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    7    9    1    2    8
[2,]    2    6    3    7    6    1
[3,]    8    1    4    9    3    4
```

  - Subset a number of columns or rows with positive integers:

```
> big.m[,c(1,3,4)]
     [,1] [,2] [,3]
[1,]    1    9    1
[2,]    2    3    7
[3,]    8    4    9
```

Columns 1, 3, & 4

```
> big.m[,1:4]
     [,1] [,2] [,3] [,4]
[1,]    1    7    9    1
[2,]    2    6    3    7
[3,]    8    1    4    9
```

Columns 1 through 4

  - Remove a number of columns or rows with negative integers:

```
> big.m[-1,]
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    2    6    3    7    6    1
[2,]    8    1    4    9    3    4
```

All rows except 1

Check Your Understanding

You run the following line to create a matrix:

```
my.matrix2 <- matrix(seq(1,21), nrow=7, byrow=TRUE)
```

Which line of code will subset only two-digit numbers (those greater than 9)?

Correct

a) `my.matrix2[4:7, ]`

```
> my.matrix2[4:7,]
     [,1] [,2] [,3]
[1,]   10   11   12
[2,]   13   14   15
[3,]   16   17   18
[4,]   19   20   21
```

b) `my.matrix2[4:7, 3]`

```
> my.matrix2[4:7,3]
[1] 12 15 18 21
```

c) `my.matrix2[>9]`

```
> my.matrix2[>9]
Error: unexpected '>' in "my.matrix2[>"
```

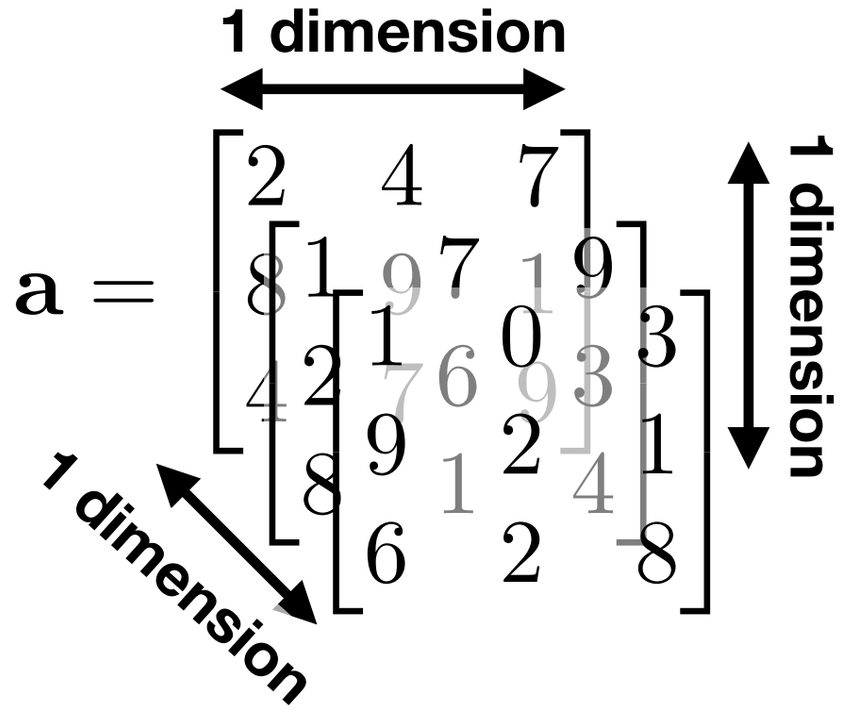
d) `my.matrix2[ , 4:7]`

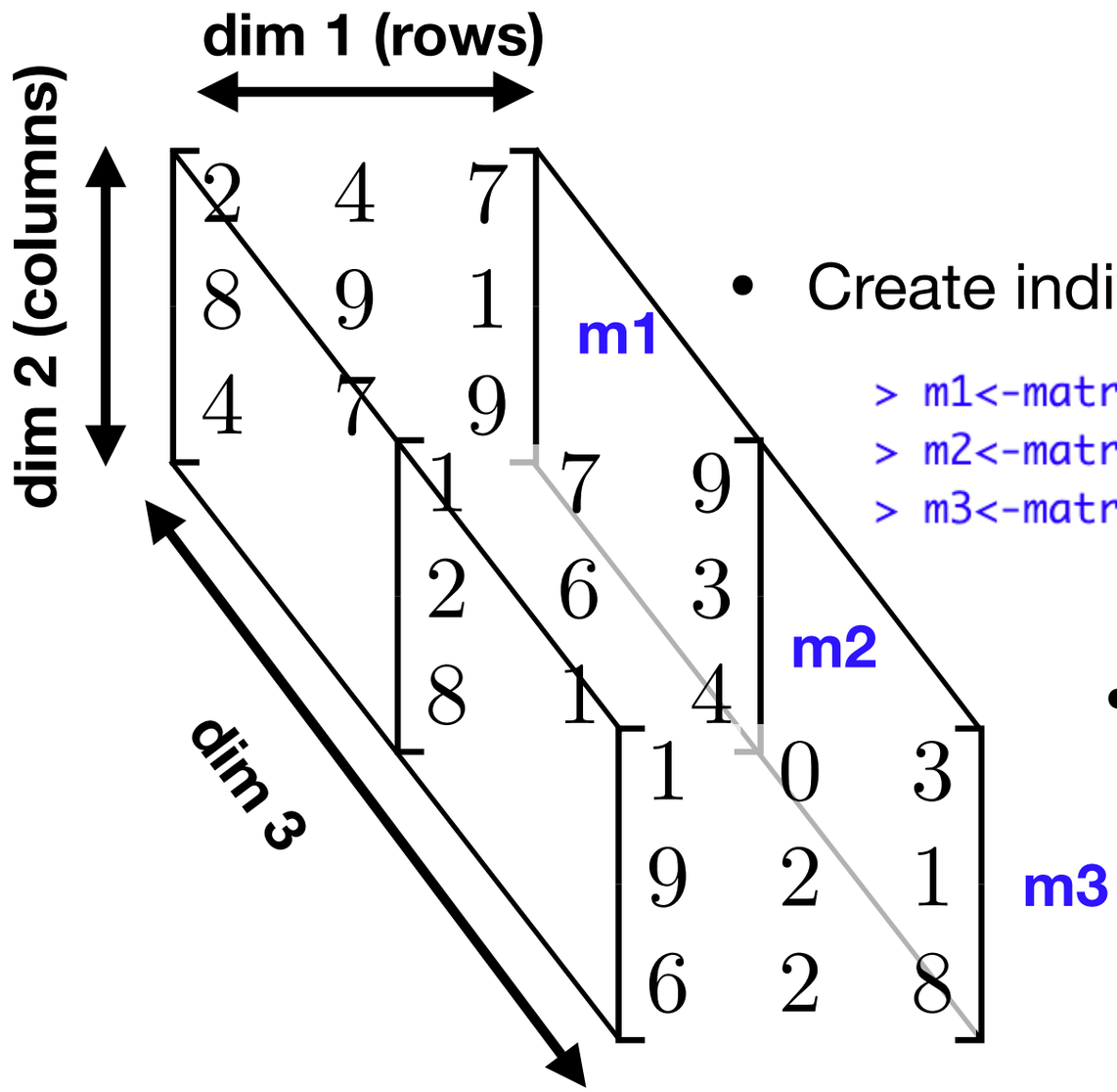
```
> my.matrix2[,4:7]
Error in my.matrix2[, 4:7] :
subscript out of bounds
```

Multidimensional Arrays

- **Arrays** are *n*-dimensional and numeric.

Example: 3-D array





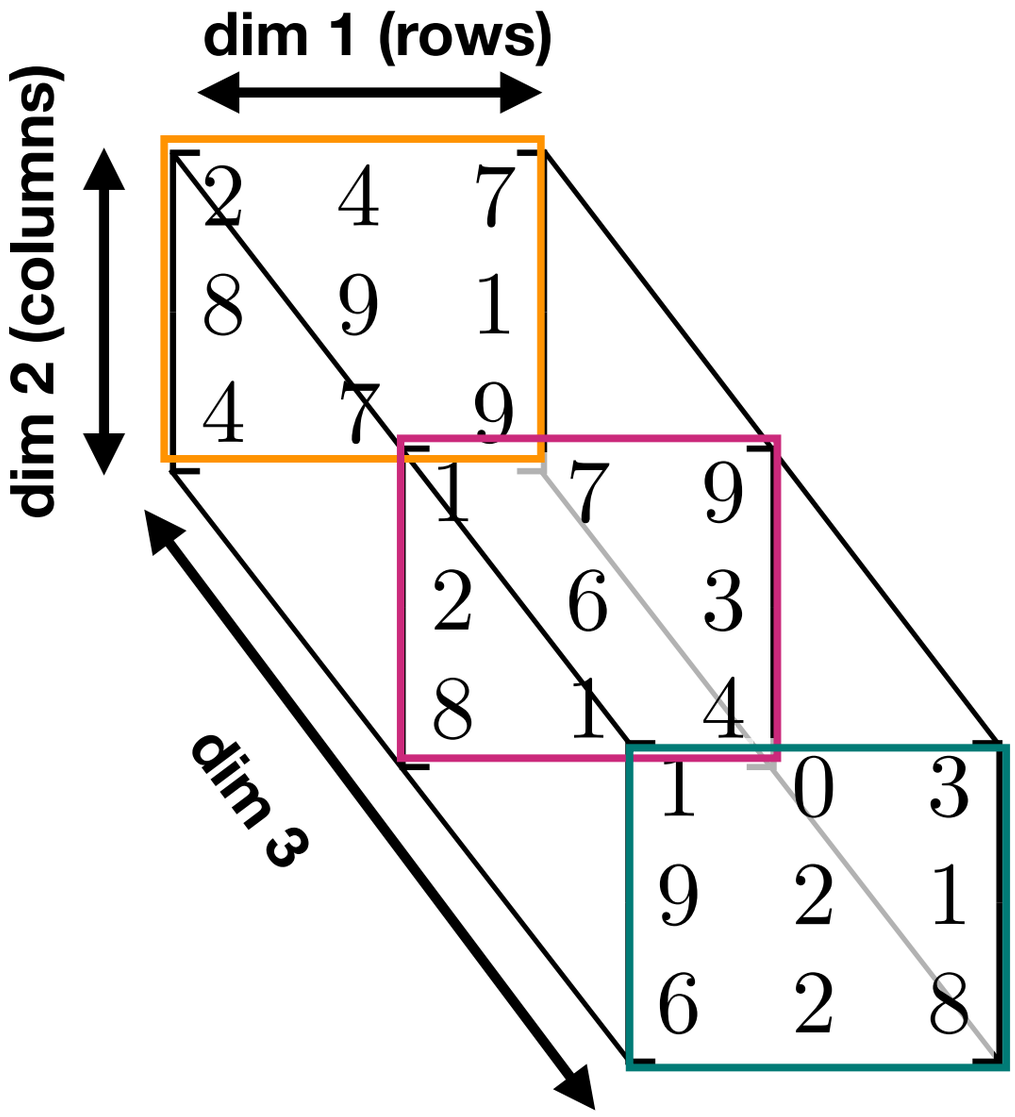
- Create individual matrices:

```
> m1<-matrix(c(2,4,7,8,9,1,4,7,9), nrow=3, byrow=TRUE)
> m2<-matrix(c(1,7,9,2,6,3,8,1,4), nrow=3, byrow=TRUE)
> m3<-matrix(c(1,0,3,9,2,1,6,2,8), nrow=3, byrow=TRUE)
```

- Create a 3D array:

```
> a1 <- array(c(m1,m2,m3),dim=c(3,3,3))
```

Indexing Multidimensional Arrays




```
> a1
, , 1

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

, , 2

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

, , 3

     [,1] [,2] [,3]
[1,]    1    0    3
[2,]    9    2    1
[3,]    6    2    8
```

Check Your Understanding

```
> a1
, , 1

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

, , 2

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

, , 3

     [,1] [,2] [,3]
[1,]    1    0    3
[2,]    9    2    1
[3,]    6    2    8
```

column

row

Which position in a1 hold the value 0?

a) `a1[1, 1, 1]`

b) `a1[1, 2, 3]`

Correct

c) `a1[2, 1, 3]`

d) `a1[1, 2, 1]`

Which positions in `a1` hold the value 7?

Write down your answer!

Action Items

1. **Complete assignments 1.8 and 1.9.**

2. **Read Davies Ch. 5 for next time.**