

Exploratory Statistical Data Analysis With R Software (ESDAR) Swayam Prabha

Lecture 5

Built-in Commands and Missing Values

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Slides can be downloaded from
<http://home.iitk.ac.in/~shalab/sp>



Built in commands

Some commands are readily available in R to compute mathematical functions.

How to use them and utilize them in computing various quantities.

Maximum

```
> max(8.2, 6.7, -4.3, 2)
[1] 8.2
```

```
R Console
> max(8.2, 6.7, -4.3, 2)
[1] 8.2
```

```
> max( c(8.2, 6.7, -4.3, 2) )
[1] 8.2
```

```
R Console
> max( c(8.2, 6.7, -4.3, 2) )
[1] 8.2
```

Minimum

```
> min(8.2, 6.7, -1.3, 2)
[1] -1.3
```

R Console

```
> min(8.2, 6.7, -1.3, 2)
[1] -1.3
```

```
> min( c(8.2, 6.7, -1.3, 2) )
[1] -1.3
```

R Console

```
> min( c(8.2, 6.7, -1.3, 2) )
[1] -1.3
```

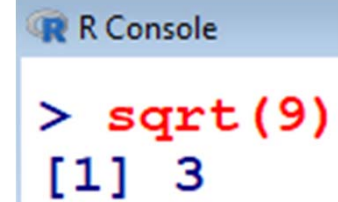
Overview Over Other Functions

<code>abs()</code>	Absolute value
<code>sqrt()</code>	Square root
<code>round()</code> , <code>floor()</code> , <code>ceiling()</code>	Rounding, up and down
<code>sum()</code> , <code>prod()</code>	Sum and product
<code>log()</code> , <code>log10()</code> , <code>log2()</code>	Logarithms
<code>exp()</code>	Exponential function
<code>sin()</code> , <code>cos()</code> , <code>tan()</code> , <code>asin()</code> , <code>acos()</code> , <code>atan()</code>	Trigonometric functions
<code>sinh()</code> , <code>cosh()</code> , <code>tanh()</code> , <code>asinh()</code> , <code>acosh()</code> , <code>atanh()</code>	Hyperbolic functions

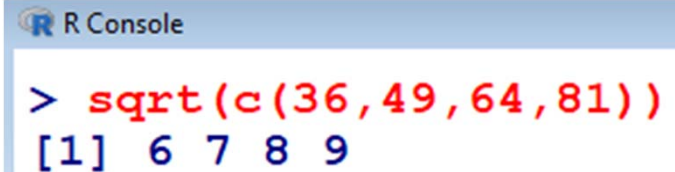
Examples

```
> sqrt(9)
[1] 3
```

```
> sqrt(c(36,49,64,81))
[1] 6 7 8 9
```



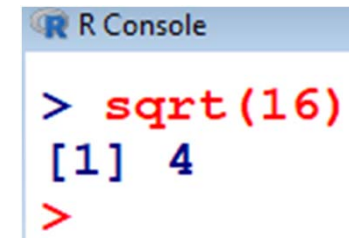
```
R Console
> sqrt(9)
[1] 3
```



```
R Console
> sqrt(c(36,49,64,81))
[1] 6 7 8 9
```

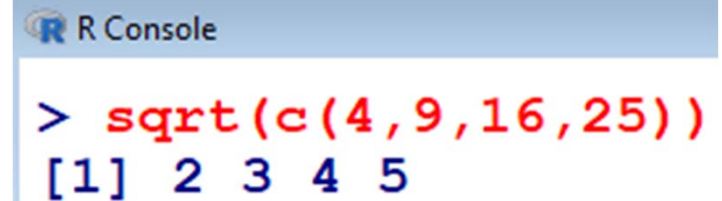
Examples

```
> sqrt(16)
[1] 4
```



```
R Console
> sqrt(16)
[1] 4
>
```

```
> sqrt(c(4,9,16,25))
[1] 2 3 4 5
```



```
R Console
> sqrt(c(4,9,16,25))
[1] 2 3 4 5
```

Examples

```
> sum(c(6,7,8,9))  
[1] 30
```

```
R Console  
> sum(c(6,7,8,9))  
[1] 30
```

```
> prod(c(6,7,8,9))  
[1] 3024
```

```
R Console  
> prod(c(6,7,8,9))  
[1] 3024
```

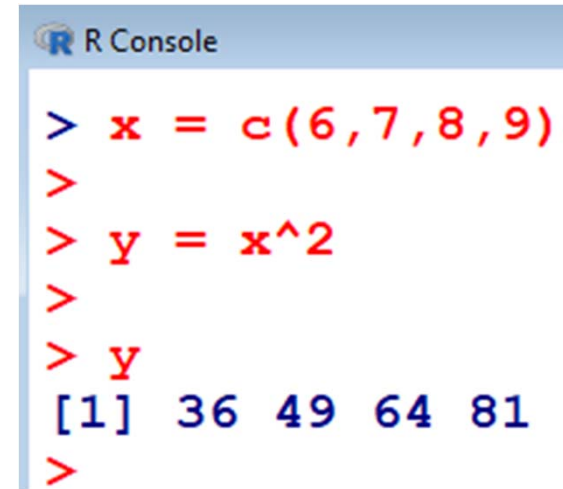

Assignments

An assignment can also be used to save values in variables:

```
> x = c(6,7,8,9)
```

```
> y = x^2
```

```
> y  
[1] 36 49 64 81
```



```
R Console  
> x = c(6,7,8,9)  
>  
> y = x^2  
>  
> y  
[1] 36 49 64 81  
>
```

Examples

To find sum of squares:

> **x** = c(6,7,8,9)

$$z = \sum_{i=1}^n x_i^2$$

Examples

To find sum of squares:

```
> x = c(6,7,8,9)
```

```
> z = sum(x^2)
```

```
> z
```

```
[1] 230
```

```
R Console  
> x = c(6,7,8,9)  
> z = sum(x^2)  
> z  
[1] 230  
>
```

Examples

To find sum of squares of deviation from mean

> **x = c(6,7,8,9)**

$$z = \sum_{i=1}^n (x_i - \bar{x})^2 = \sum_{i=1}^n x_i^2 - n\bar{x}^2$$

Examples

To find sum of squares of deviation from mean

```
> x = c(6,7,8,9)
```

$$z = \sum_{i=1}^n (x_i - \bar{x})^2 = \sum_{i=1}^n x_i^2 - n\bar{x}^2$$

```
> z = sum(x^2) - length(x) * mean(x)^2
```

```
> length(x)
```

```
[1] 4
```

```
> z
```

```
[1] 5
```

```
R Console
> x = c(6,7,8,9)
> z = sum(x^2) - length(x) * mean(x)^2
> length(x)
[1] 4
>
> z
[1] 5
>
```

Examples

To find sum of cross product:

```
> x1 = c(6,7,8,9)
```

```
> x2 = c(2,3,4,5)
```

$$6 \times 2 + 7 \times 3 + 8 \times 4 + 9 \times 5$$

```
> z = sum(x1*x2)
```

```
> z
```

```
[1] 110
```

```
R Console
> x1 = c(6,7,8,9)
> x2 = c(2,3,4,5)
> z = sum(x1*x2)
> z
[1] 110
```

Missing data

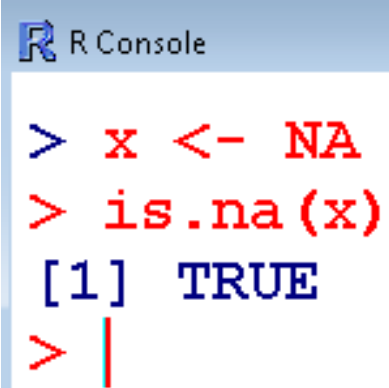
R represents missing observations through the data value **NA**

NA : Reserved word

Missing values can be detected in a data vector by using **is.na**

Create a data with NA.

```
> x <- NA      # assign NA to variable x
> is.na(x)     # is it missing?
[1] TRUE
```



```
R Console
> x <- NA
> is.na(x)
[1] TRUE
> |
```

Missing data

TRUE and **FALSE** are logical operators that are used to compare expressions.

TRUE and **FALSE** are reserved words.

T can also be used in place of **TRUE** .

F can also be used in place of **FALSE**.

TRUE and **FALSE** are not the same as **true** and **false** respectively.

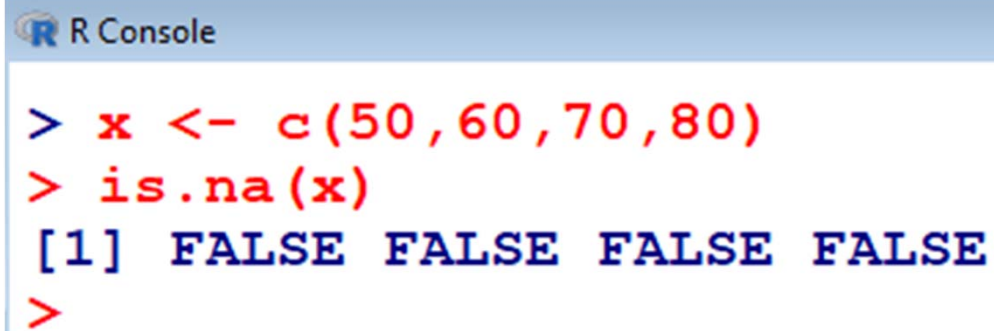
Missing data

How to know if any value is missing in a data vector?

```
> x <- c(50,60,70,80)
```

```
> is.na(x)
```

```
[1] FALSE FALSE FALSE FALSE
```



```
R Console  
> x <- c(50,60,70,80)  
> is.na(x)  
[1] FALSE FALSE FALSE FALSE  
>
```

Missing data

How to know if any value is missing in a data vector?

```
> x <- c(50,NA,70,80)
```

```
> is.na(x)
```

```
[1] FALSE TRUE FALSE FALSE
```

R Console

```
> x <- c(50,NA,70,80)
```

```
> is.na(x)
```

```
[1] FALSE TRUE FALSE FALSE
```

```
> .
```

Missing data

How to know if any value is missing in a data vector?

```
> x <- c(50,NA,NA,80)
```

```
> is.na(x)
```

```
[1] FALSE TRUE TRUE FALSE
```

R Console

```
> x <- c(50,NA,NA,80)
```

```
> is.na(x)
```

```
[1] FALSE TRUE TRUE FALSE
```

```
>
```

Example : How to work with missing data

```
> x <- c(50,60,NA,80) # data vector
```

```
> mean(x)
```

```
[1] NA
```

$$\frac{50 + 60 + \text{NA} + 80}{4}$$

```
> mean(x, na.rm = TRUE) # NAs can be removed
```

```
[1] 63.33333
```

$$\frac{50 + 60 + 80}{3} = 63.33$$

R Console

```
> x <- c(50,60,NA,80)
```

```
> mean(x)
```

```
[1] NA
```

```
> mean(x, na.rm=TRUE)
```

```
[1] 63.33333
```

```
>
```

Example : How to work with missing data

The null object, called **NULL**, is returned by some functions and expressions.

Note that **NA** and **NULL** are not the same.

Note that **NA** and **na** are not the same.

R Console

```
> x <- c(50,60,na,80)
Error: object 'na' not found
>
```

NA is a placeholder for something that exists but is missing.

NULL stands for something that never existed at all.