UNIVERSITY OF
NORTHERN BRITISH COLUMBIA

Introduction to R Programming

Tutorial

Siraj Ul Islam

Research Associate and Adjunct Professor

Environmental Science and Engineering Program

University of Northern British Columbia (UNBC)

THE R LOGO IS © 2016, THE R FOUNDATION

Brief Biography

Instructor: Siraj Ul Islam **Email:** sirajul.islam@unbc.ca **Office:** 4-256

- QAU: MSc, Physics, PAKISTAN
- QAU: MPhil, Computational Physics, PAKISTAN
- GCISC: Scientific Officer (Climate Modeling), PAKISTAN
- ITCP: Junior Associate (Earth System Science), ITALY
- UNBC: PhD, Climate Modeling/Dynamics, CANADA
- UNBC: PDF, Hydrological Modeling, Analysis
- UNBC: Adjunct Professor, Environmental Science
- UNBC: Research Associate, Hydrological-Water Temperature Modeling
- UNBC: Instructor, Environmental Science

Outline

- R and RStudio Environments
- R Basics
  - objects, elements arithmetic
  - packages, functions, examples
- Plotting
  - scatter, line, histograms
  - examples, storing plots
- Looping
  - for, if loops
- Read Write Data
- Sample Data Analysis
  - linear model, trend
  - Basic statistics

THE R LOGO IS © 2016, THE R FOUNDATION

R Programming

- R is a programming language developed by Ross Ihaka and Robert Gentleman in 1993.

- It is an open source computing package used by not only academic, but many companies also use R programming including Uber, Google, Airbnb, Facebook

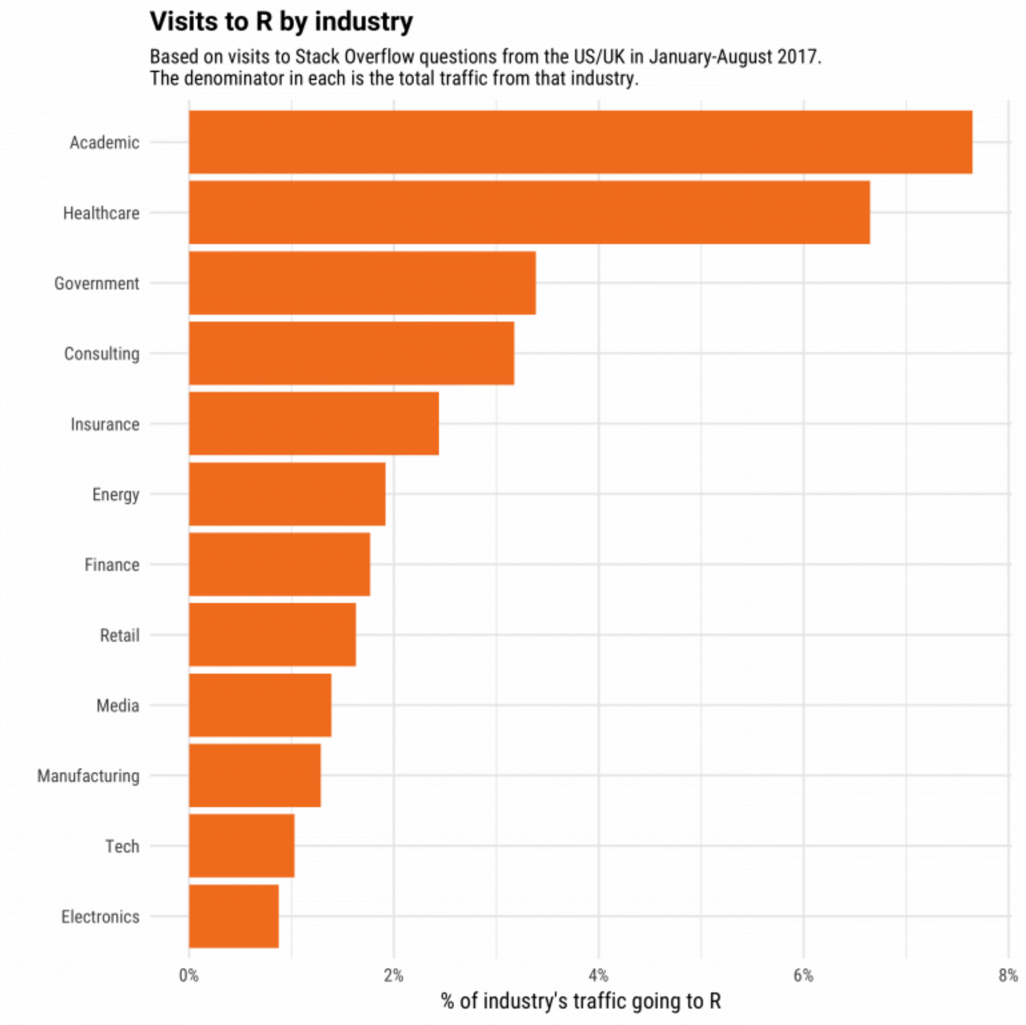


Figure Source: https://www.guru99.com/r-programming-introduction-basics.html

R Environment

R code is stored with .R file extension such as hello.R

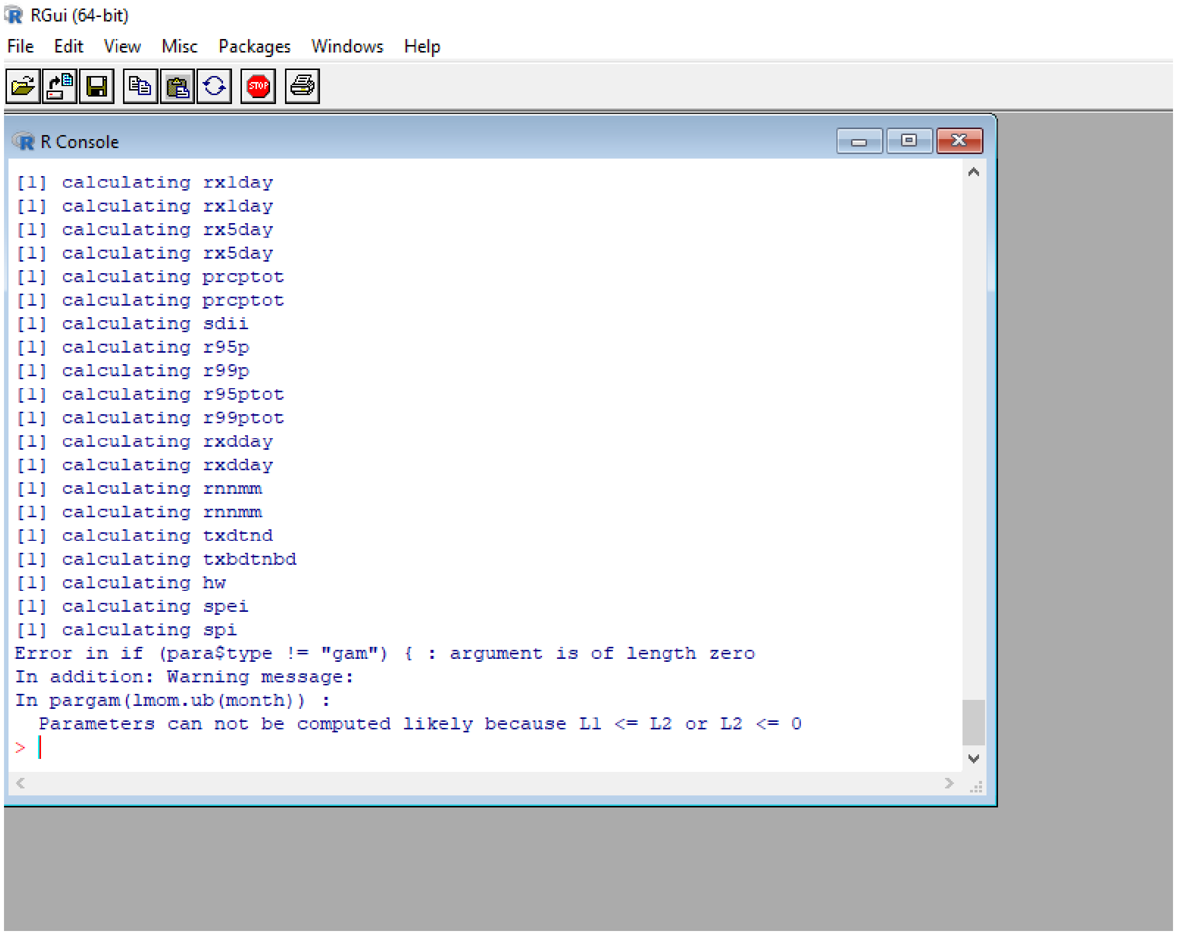
```
#hello.R
x <- "hello world"
print(x)
```

To run R code

```
source( "hello.R" )
```

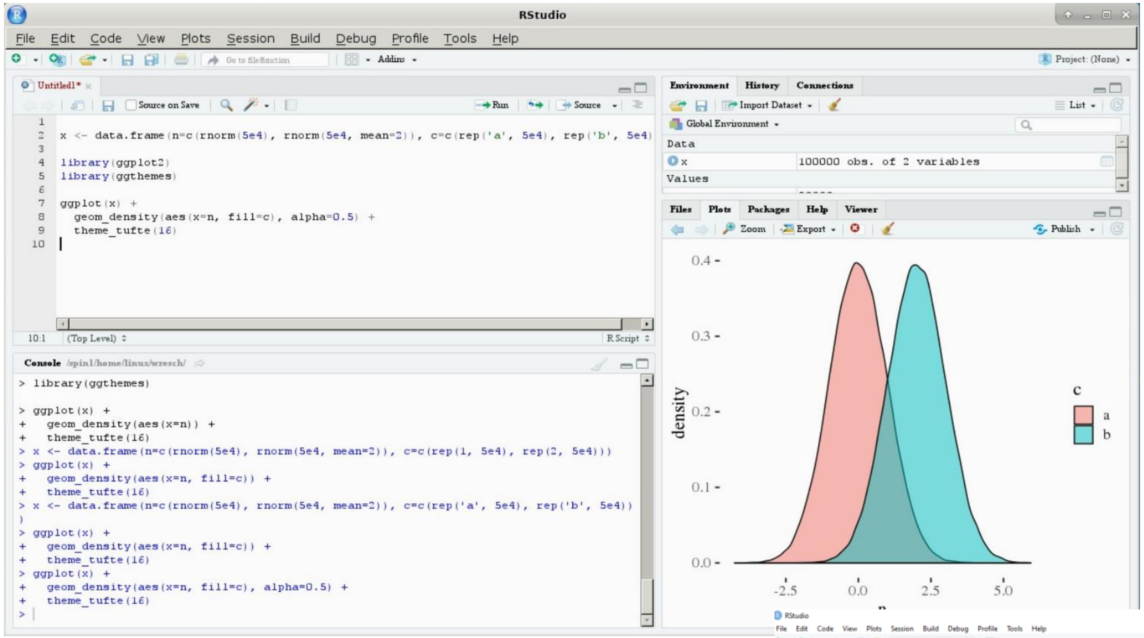
Comments are included in the code as

```
# A script to analyze data
# author: Siraj
# date: 11/3/2020
```



```
[1] calculating rx1day
[1] calculating rx1day
[1] calculating rx5day
[1] calculating rx5day
[1] calculating prcptot
[1] calculating prcptot
[1] calculating sdii
[1] calculating r95p
[1] calculating r99p
[1] calculating r95ptot
[1] calculating r99ptot
[1] calculating rxdday
[1] calculating rxdday
[1] calculating rnnmm
[1] calculating rnnmm
[1] calculating txdtnd
[1] calculating txbdtnbd
[1] calculating hw
[1] calculating spei
[1] calculating spi
Error in if (para$type != "gam") { : argument is of length zero
In addition: Warning message:
In pargam(lmom.ub(month)) :
  Parameters can not be computed likely because L1 <= L2 or L2 <= 0
> 
```

Rstudio Environment



```
x <- data.frame(n=c(rnorm(5e4), rnorm(5e4, mean=2)), c=c(rep('a', 5e4), rep('b', 5e4)

library(ggplot2)
library(ggthemes)

ggplot(x) +
  geom_density(aes(x=n, fill=c), alpha=0.5) +
  theme_tufte(16)
```

```
Data
x            100000 obs. of 2 variables
Values
```

```
> library(ggthemes)

> ggplot(x) +
+   geom_density(aes(x=n)) +
+   theme_tufte(16)
> x <- data.frame(n=c(rnorm(5e4), rnorm(5e4, mean=2)), c=c(rep(1, 5e4), rep(2, 5e4)))
> ggplot(x) +
+   geom_density(aes(x=n, fill=c)) +
+   theme_tufte(16)
> x <- data.frame(n=c(rnorm(5e4), rnorm(5e4, mean=2)), c=c(rep('a', 5e4), rep('b', 5e4))
)
> ggplot(x) +
+   geom_density(aes(x=n, fill=c)) +
+   theme_tufte(16)
> ggplot(x) +
+   geom_density(aes(x=n, fill=c), alpha=0.5) +
+   theme_tufte(16)
> |
```

R language

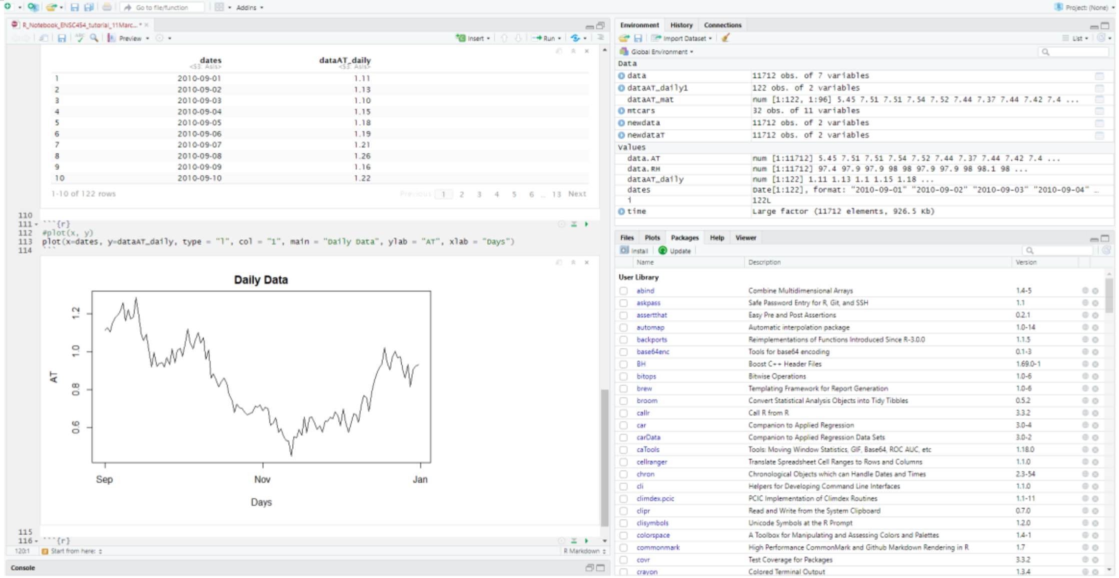
https://www.r-project.org/about.html

Open source

R Studio

https://www.rstudio.com/

Open source



	dates	dataAT_daily
1	2010-09-01	1.11
2	2010-09-02	1.13
3	2010-09-03	1.10
4	2010-09-04	1.15
5	2010-09-05	1.18
6	2010-09-06	1.19
7	2010-09-07	1.21
8	2010-09-08	1.26
9	2010-09-09	1.16
10	2010-09-10	1.22

1-10 of 122 rows Previous 1 2 3 4 5 6 ... 13 Next

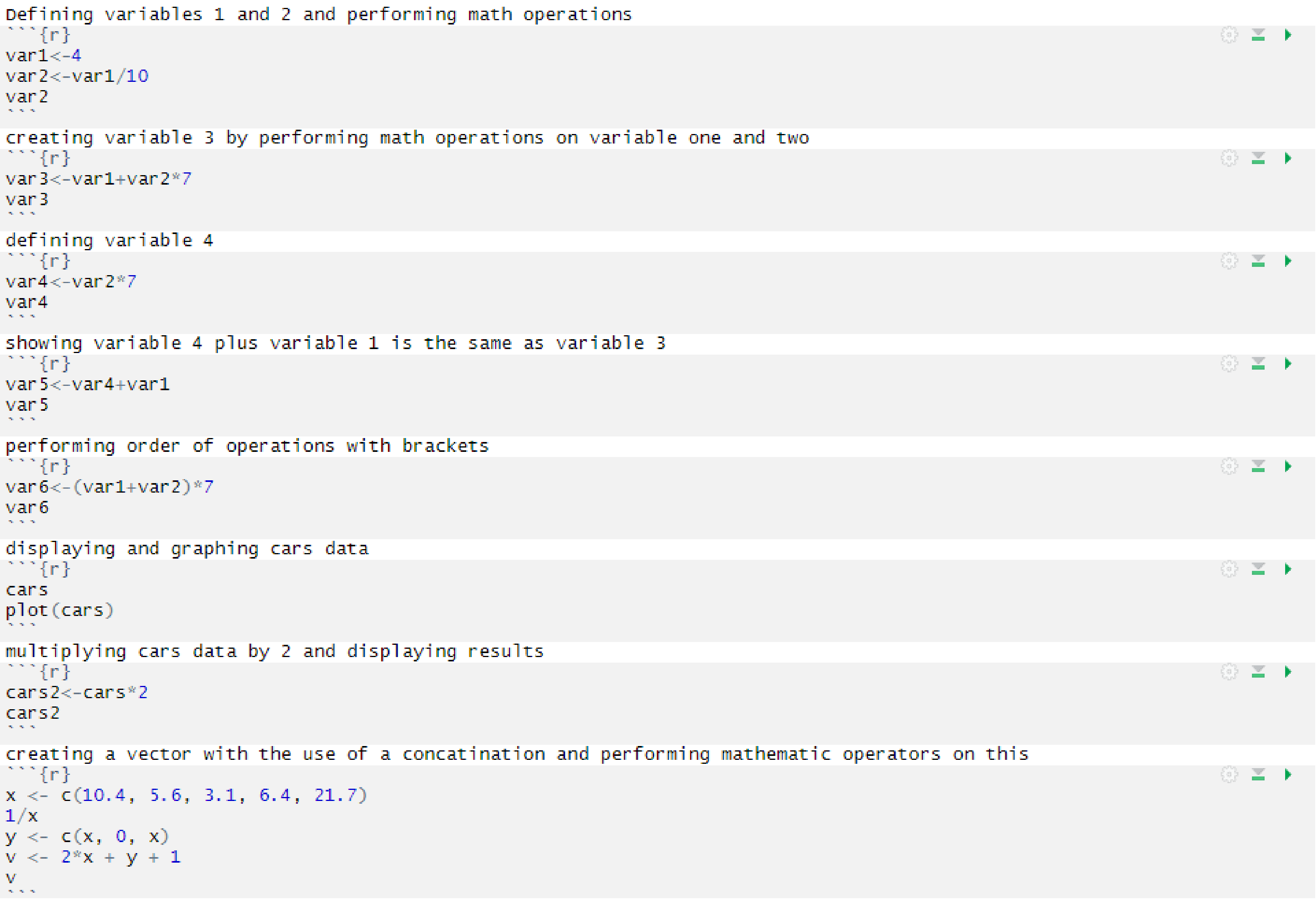
~~~
```{r}
#plot(x, y)
plot(x=dates, y=dataAT_daily, type = "l", col = "1", main = "Daily Data", ylab = "AT", xlab = "Days")
```
~~~

```
Data
data          11712 obs. of 7 variables
dataAT_daily1 122 obs. of 2 variables
dataAT_mat    num [1:122, 1:96] 5.45 7.51 7.51 7.54 7.52 7.44 7.37 7.44 7.42 7.4 ...
mtcars        32 obs. of 11 variables
newdata       11712 obs. of 2 variables
newdataT      11712 obs. of 2 variables
Values
data.AT       num [1:11712] 5.45 7.51 7.51 7.54 7.52 7.44 7.37 7.44 7.42 7.4 ...
data.RH       num [1:11712] 97.4 97.9 97.9 98 98 97.9 97.9 98 98.1 98 ...
dataAT_daily  num [1:122] 1.11 1.13 1.1 1.15 1.18 ...
dates         Date[1:122], format: "2010-09-01" "2010-09-02" "2010-09-03" "2010-09-04" ...
i             122L
time          Large factor (11712 elements, 926.5 Kb)
```

	Name	Description	Version
	User Library		
□	abind	Combine Multidimensional Arrays	1.4-5
□	askpass	Safe Password Entry for R, Git, and SSH	1.1
□	assertthat	Easy Pre and Post Assertions	0.2.1
□	automap	Automatic interpolation package	1.0-14
□	backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.5
□	base64enc	Tools for base64 encoding	0.1-3
□	BH	Boost C++ Header Files	1.69.0-1
□	bitops	Bitwise Operations	1.0-6
□	brew	Templating Framework for Report Generation	1.0-6
□	broom	Convert Statistical Analysis Objects into Tidy Tibbles	0.5.2
□	callr	Call R from R	3.3.2
□	car	Companion to Applied Regression	3.0-4
□	carData	Companion to Applied Regression Data Sets	3.0-2
□	caTools	Tools: Moving Window Statistics, GIF, Base64, ROC AUC, etc	1.18.0
□	cellranger	Translate Spreadsheet Cell Ranges to Rows and Columns	1.1.0
□	chron	Chronological Objects which can Handle Dates and Times	2.3-54
□	cli	Helpers for Developing Command Line Interfaces	1.1.0
□	climdex.pcic	PCIC Implementation of Climdex Routines	1.1-11
□	clipr	Read and Write from the System Clipboard	0.7.0
□	clisymbols	Unicode Symbols at the R Prompt	1.2.0
□	colorspace	A Toolbox for Manipulating and Assessing Colors and Palettes	1.4-1
□	commonmark	High Performance CommonMark and Github Markdown Rendering in R	1.7
□	covr	Test Coverage for Packages	3.3.2
□	crayon	Colored Terminal Output	1.3.4

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Rstudio Notebooks



~~~~
Defining variables 1 and 2 and performing math operations
```{r}
var1<-4
var2<-var1/10
var2
```
creating variable 3 by performing math operations on variable one and two
```{r}
var3<-var1+var2*7
var3
```
defining variable 4
```{r}
var4<-var2*7
var4
```
showing variable 4 plus variable 1 is the same as variable 3
```{r}
var5<-var4+var1
var5
```
performing order of operations with brackets
```{r}
var6<-(var1+var2)*7
var6
```
displaying and graphing cars data
```{r}
cars
plot(cars)
```
multiplying cars data by 2 and displaying results
```{r}
cars2<-cars*2
cars2
```
creating a vector with the use of a concatination and performing mathematic operators on this
```{r}
x <- c(10.4, 5.6, 3.1, 6.4, 21.7)
1/x
y <- c(x, 0, x)
v <- 2*x + y + 1
v
```
~~~~

Objects in R

- types of objects: vector, array, matrix, data.frame
- attributes
  - mode: numeric, character, logical
  - length: number of elements in object
- Creation
  - assign a value
  - create a blank object

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Naming Convention

- must start with a letter (A-Z or a-z)
- variable names are case sensitive (S is not same as s)
- variable names starting with either numbers (e.g. 2Y) or symbols (e.g. %Y) is not allowed
- variable names should not contain blank spaces
- can contain letters, digits (0-9), and/or periods “.”

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Objects and Assignments

- “<-” used to indicate assignment

```
x<-c(1,2,3,4,5,6,7)
x<-c(1:7)
x<-1:7

A <- c(1,2,3,4,5,6,7) or A <- 1:7
```

- list objects

```
ls()
```

- remove objects

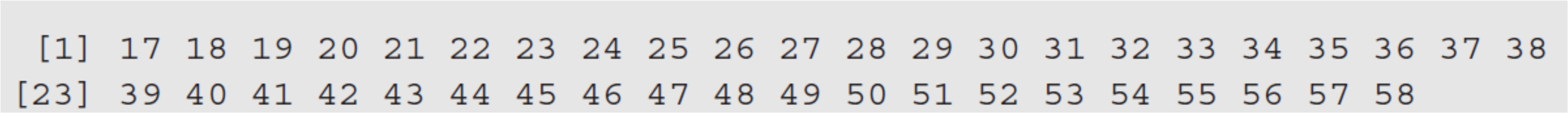
```
rm()
```

- *note: as of version 1.4 “=“ is also a valid assignment operator*

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

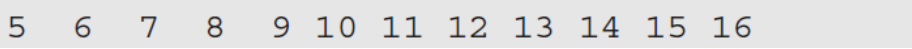
Objects Sequencing/Arithmetic

```
17:58
```



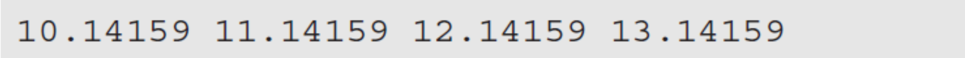
```
 [1] 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
[23] 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
```

```
5:(2*3 + 10)
```



```
5  6  7  8  9 10 11 12 13 14 15 16
```

`(7:10) + pi` # pi is a stored constant



```
10.14159 11.14159 12.14159 13.14159
```

Objects can be joined together (i.e. concatenated) with the c function. For example

`s <- c(x, a)` # x and a are vectors.

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Extract Elements of Objects

```
x <- c(0, 2, 4, 6, 8, 10)
x[3]  # access 3rd element
4
x[c(2, 4)]  # access 2nd and 4th element
2 6
```

select range of elements

```
x[1:3]
```

select all but one element

```
x[-3]
```

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Object: Matrix

- a matrix is a vector with an additional attribute (dim) that defines the number of columns and rows
- only one mode (numeric, character or logical) allowed
- can be created using `matrix()`

```
x<-matrix(data=0,nrow=2,ncol=2)
```

or

```
x<-matrix(0,2,2)
```

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Loading Packages

- Packages are the fundamental units of R code. Many packages exist for data analysis and plotting. To activate the package, use the following command.

```
library(stats)
library(ggplot2)
```

- The most powerful packages is the “ggplot2”.

Functions

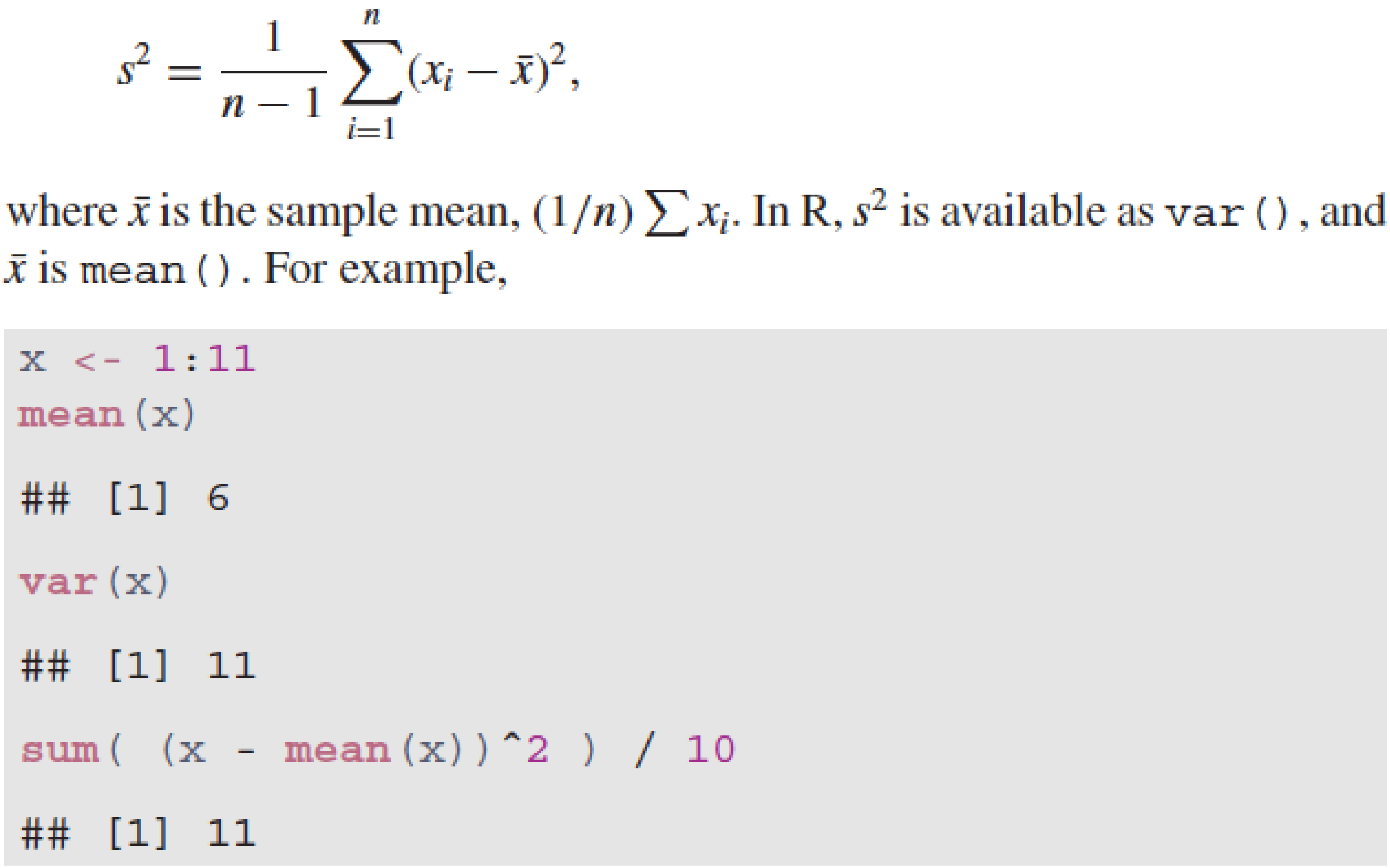
- In R, actions can be performed on objects using functions.
- parentheses () are used to specify that a function is being called.

```
mean(x)
```

```
source(“my_function.R”)
my_mean(x)
```

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Example Functions



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

where $\bar{x}$ is the sample mean, $(1/n)\sum x_i$. In R, s^2 is available as `var()`, and $\bar{x}$ is `mean()`. For example,

```
x <- 1:11
mean(x)

## [1] 6

var(x)

## [1] 11

sum( (x - mean(x))^2 ) / 10

## [1] 11
```

Practice Examples

Create a vector filled with random numbers

```
U1 <- rnorm(30)
```

Create a 30 x 30 matrix i.e. (30 rows and 30 columns)

```
mymat <- matrix(nrow=30, ncol=30)
```

Just show the upper left 10x10 chunk

```
mymat[1:10, 1:10]
```

Calculate mean

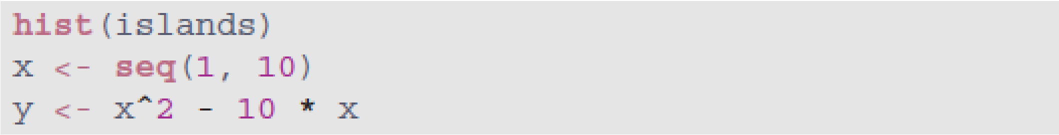
```
M = sum(xx)/length(xx)
```

Covert temperature Fahrenheit to Celsius

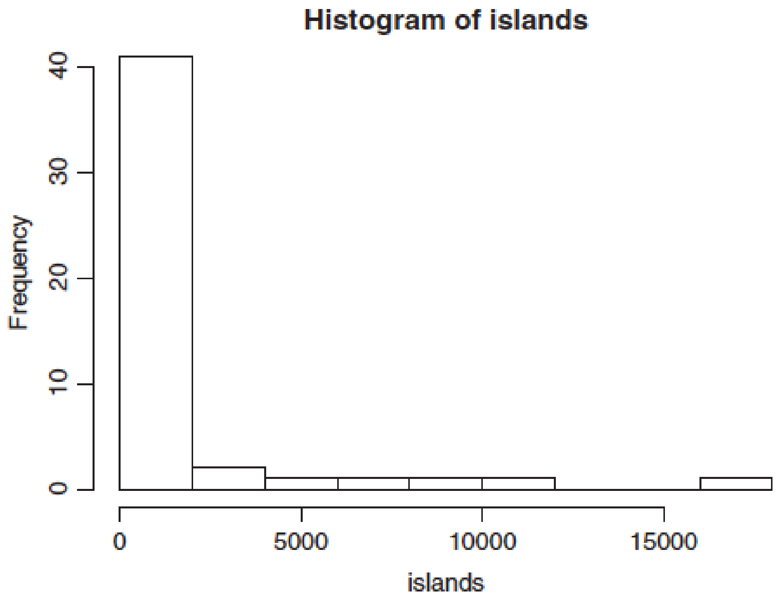
```
temp_C <- (temp_F - 32) * 5 / 9
```

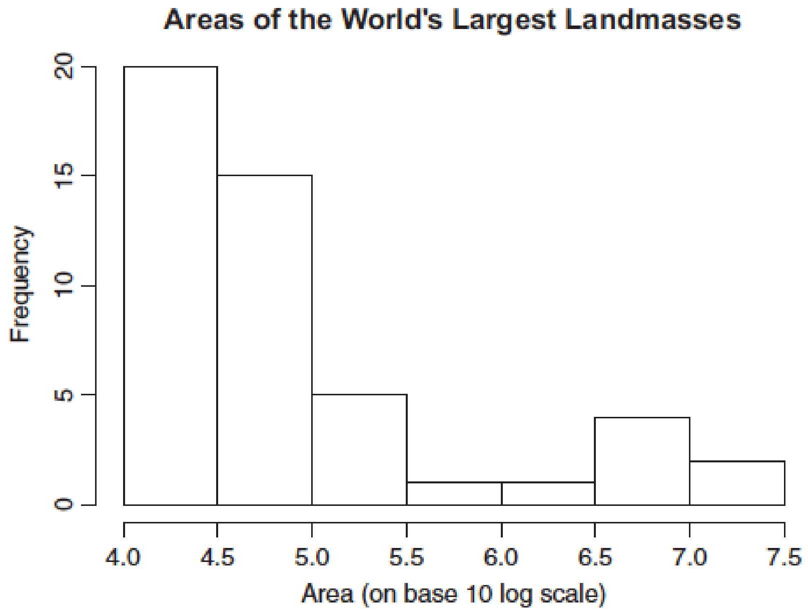
Source: www.datacamp.com

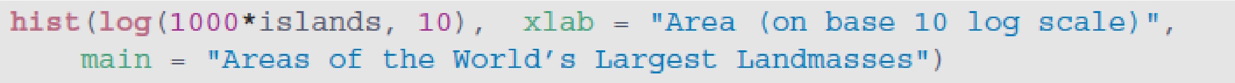
Plotting: Histograms



```
hist(islands)
x <- seq(1, 10)
y <- x^2 - 10 * x
```







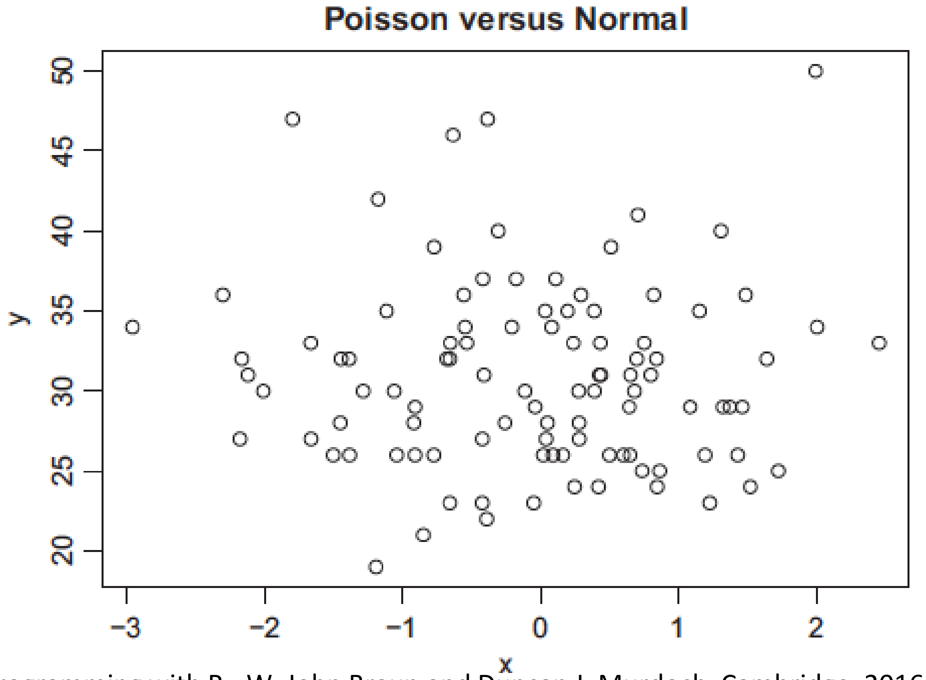
```
hist(log(1000*islands, 10),  xlab = "Area (on base 10 log scale)",
    main = "Areas of the World's Largest Landmasses")
```

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Scatter Plot

```
x <- rnorm(100)          # assigns 100 random normal observations to x
y <- rpois(100, 30)      # assigns 100 random Poisson observations
                         # to y; mean value is 30
mean(y)                  # the resulting value should be near 30

## [1] 30.91
```

```
plot(x, y, main = "Poisson versus Normal")
```



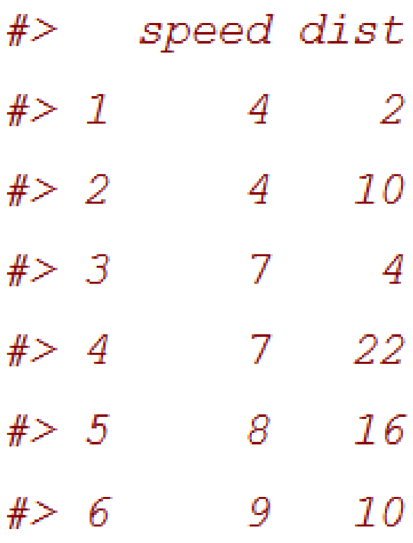
Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Plotting Example Data

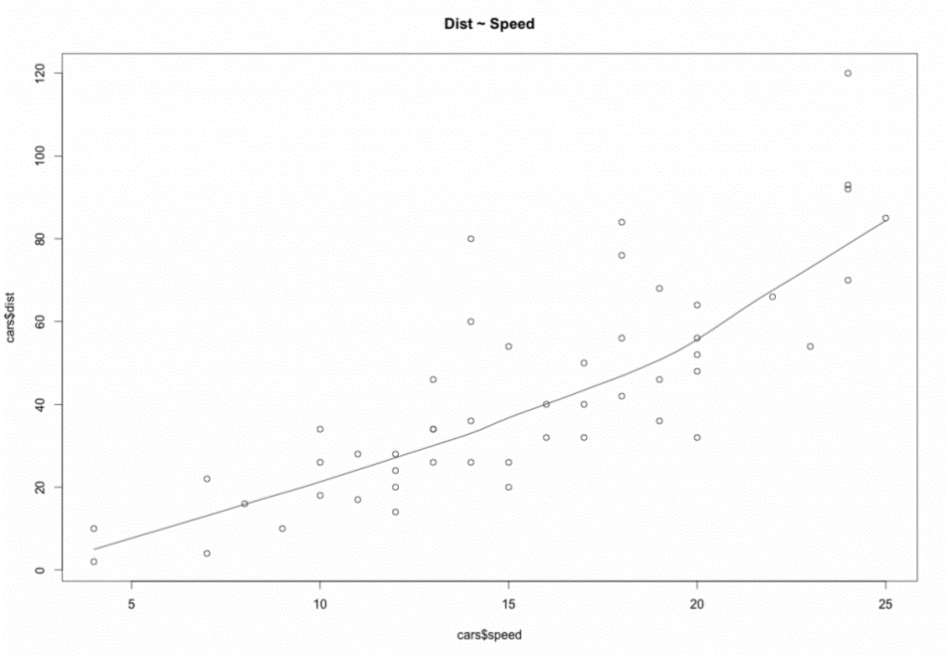
- cars is a standard built-in dataset, that makes it convenient to show analysis in a simple and easy to understand fashion.
- It consists of 50 observations(rows) and 2 variables (columns) – dist and speed. Lets print out the first six observations here.

```
head(cars)
```

```
R <- cor(cars$speed, cars$dist)
```



```
#>   speed dist
#> 1     4    2
#> 2     4   10
#> 3     7    4
#> 4     7   22
#> 5     8   16
#> 6     9   10
```



Source: https://www.machinelearningplus.com/machine-learning

Saving Plots

R can save plots in different formats e.g. “PNG”, and “PDF”.

to save a graph to PNG file. First set resolution in ppi unit.

```
ppi <- 300
```

width and height are in inch.

```
png("fig_1.png", width = 6 * ppi, height = 3 *
ppi, res = ppi)
plot(nvec,x,type="l")
dev.off()
```

to save a PDF file.

```
pdf("fig_1.pdf")
plot(nvec,x,type="l")
dev.off()
```

Source: Nobuhiko ENDO, Japan Agency for Marine-Earth Science and Technology

Logics and Automation: Looping

Looping allows to run the command many times. For example, if one needs to print a sentence, saying “I like Prince George”, it can be simply done by the command

```
print(“I like Prince George”)
```

Now, if this sentence is required to repeat 10 times, one simple way is to repeat the above command 10 times but the “smarter” way is to use a specific loop called “for” loop.

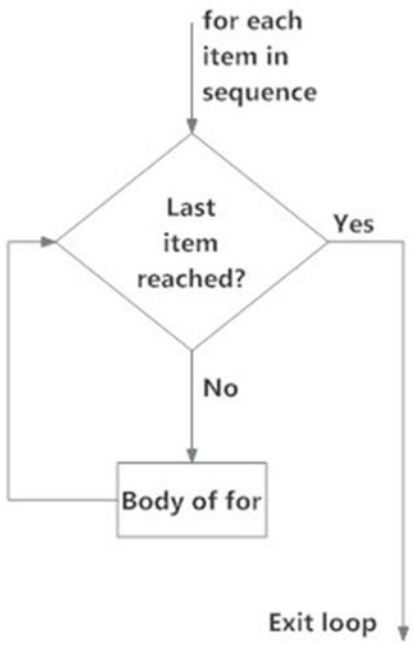
“for” Loop in R

Conceptually, a loop is a way to repeat a sequence of instructions under certain conditions. They allow users to automate parts of the code that need of repetition.

```
for (i in 1:10)
{
print(“I love Prince George”)
}
```

Lets print numbers 1, 4, 9, 16, 25

```
for(i in 1:5){print(i^2)}
```



“for” Loop in R

If we want to store the above solution in a vector for future use, we can do as below

```
 y<-numeric(5)
 x<-c(-3, 6, 2, 5, 9)
 m<-0
 for(i in x){
 m<- m+1
 y[m]<-i^2
}
```

Nested “For loop”

Lets say x=[2, 3,-2,4,5] and y=[-1, 2, 3,5,6]. Calculate their cross-product which should be a matrix.

```
z = matrix(nrow=5,ncol=5)
x<- c(2,3,-2,4,5);
y<- c(-1,2,3,5,6)
 for (i in 1:5){
    for(j in 1:5){
    z[i,j]=x[i]*y[j];
    }
 }
```

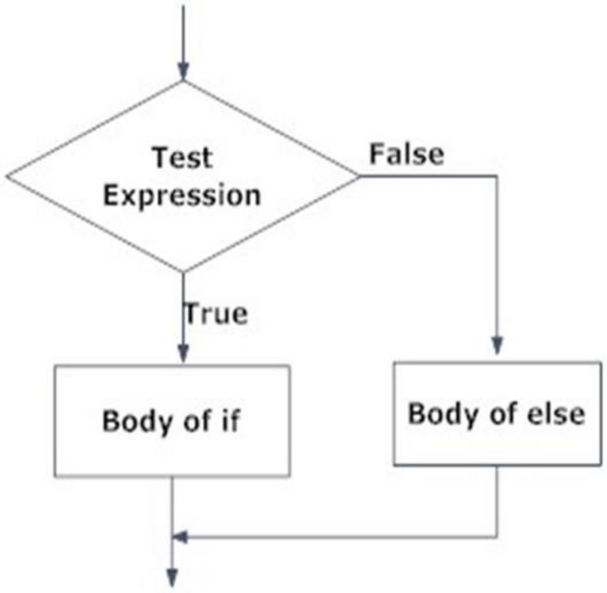
“if” loop

Decision making is an important part of programming. For example, one plans to fish if it is sunny. Intuitionally we have the below command

If (weather = sunny) {fishing}

The syntax of if statement is:

```
if (condition) {
    statement
}
```



“if … else” statement

```
x <- 5
if(x > 0){
   print("Positive number")
}
```

Output
"Positive number"

```
x <- -5
if(x > 0){
   print("Non-negative number")
} else {
   print("Negative number")
}
```

Output: "Negative number"

Basic Logical Operations

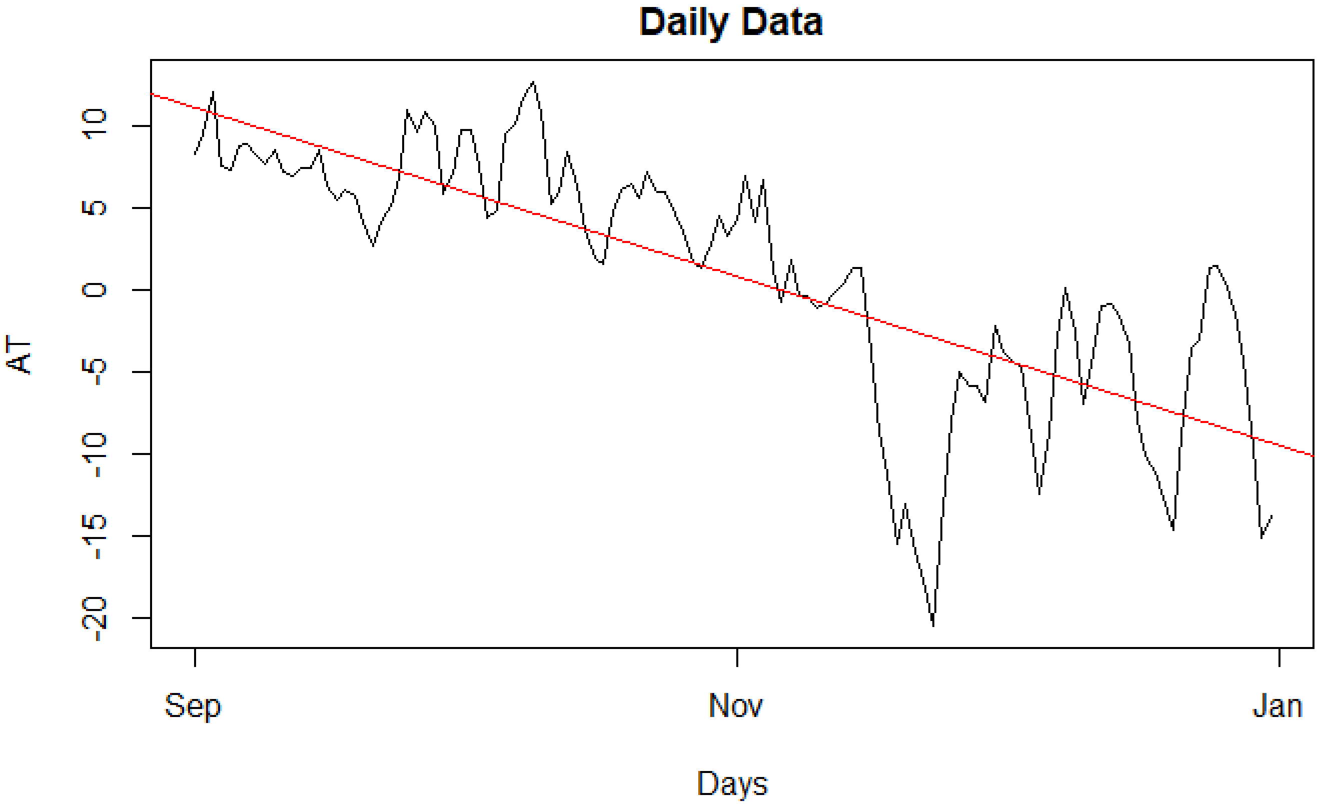
Operator	Description
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
= =	Equal to exactly
! =	Not equal to

Analysis of Sample Data

```
data <- read.csv("sample_weather_data.csv", header = TRUE)
head(data)

data.AT <- data$AirTC
data.RH <- data$RH

length(data.AT)

#matrix conversion
dataAT_mat<- matrix(data.AT, nrow=96, ncol=122)

#emptry matrix for new data
dataAT_daily<- matrix(nrow=122, ncol=0)

for(i in 1:122){
dataAT_daily[i]<- mean(dataAT_mat[,i])
}
dataAT_daily
```

Trend Analysis

```
trnd <- lm(dataAT_daily~dates)
plot(x=dates, y=dataAT_daily, type = "l", col = "1", main = "Daily Data",
ylab = "AT", xlab = "Days")
abline(trnd, col="red")
```



Data Statistics

Correlation

```
CR <-cor(dataAT_daily,dataRH_daily)
```

Variance

```
VR <-var(dataAT_daily)
```

Standard Deviation

```
SD <-sd(dataAT_daily)
```

Mean

```
MN<- mean(dataAT_daily)
```

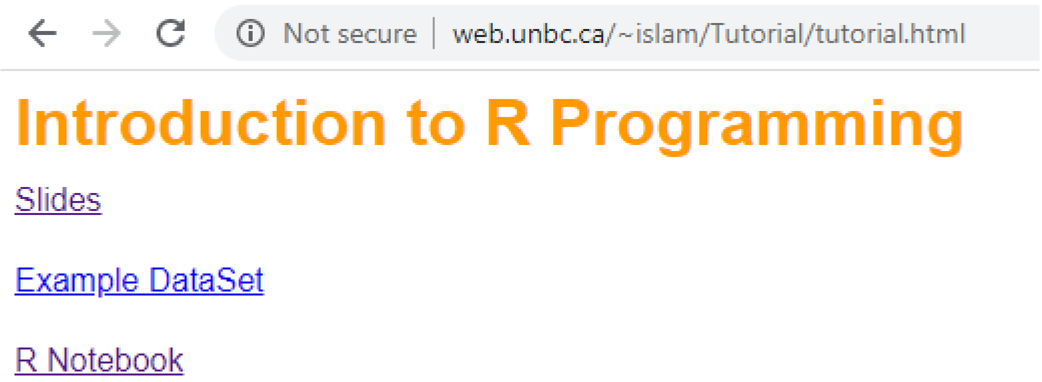
Coefficient of Variation

```
CV<- SD/MN
```

Hands-on Training

Let's start coding in R using a sample weather data file. Both Rstudio notebook (.rmd) and sample data files (.csv) are available on my website at: http://web.unbc.ca/~islam

Click on the “Tutorial” tab on the right.



web.unbc.ca/~islam/Tutorial/tutorial.html

Introduction to R Programming

Slides

Example DataSet

R Notebook