ISyE 3770, Spring 2024
Statistics and Applications

Introduction to R

Instructor: Jie Wang
H. Milton Stewart School of
Industrial and Systems Engineering
Georgia Tech

jwang3163@gatech.edu
Office: ISyE Main 445

Outline

- R download and Installation
- Basic Operations in R
  - Getting help
  - Arithmetic Operators
  - Numbers and Variables
  - Built-in Functions
  - Assignment and Removement
- Statistical Applications
  - Data Summaries
  - Box-plot and histogram
  - Time series
  - Pie chart, bar chart, stem-and-leaf
  - Scatter plot

Download and Installation

R can be downloaded from the Comprehensive R Archive Network (CRAN) mirror site for free:

http://cran.r-project.org/

In "Download and Install R" section, click to choose between Linux, Mac OS and Windows, then follow the (rather different) instructions

RStudio is a free and open source integrated development environment for R. It can be downloaded from:

http://www.rstudio.com/

Package Installation

Part of the reason R has become so popular is the vast array of packages available at the CRAN repositories.

In the last few years, the number of packages has grown exponentially!

Suppose you want to install the ggplot2 package

```
> install.packages("ggplot2")
```

In order to use functions in this package, type

```
> library(ggplot2)
```

Basic Operations in R
Getting Help

- If you know the name of the function you want help with, type a question mark ? followed by the name of the function, or use help()
  - ?read.table
  - help(read.table)
- If you only know the subject on which you want help, use help.search(" ")
  - help.search("data input")
- If you are connected to the internet, you can type CRAN in Google and search for the help you need at CRAN.

Basic Operations in R

Arithmetic Operations

Some examples:

```
> 7*3
[1] 21
```

Note: The counter [1] is included to number the first entry on that line of output. When variables with lots of entries are printed, it is easier to find a specific entry.

```
> c(1,2,3)
[1] 1 2 3
> 10:50
[1] 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
[22] 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
> seq(from=1,to=10,by=0.5) or seq(1,10,0.5)
 [1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0  8.5
9.0  9.5 10.0
```

Basic Operations in R
Numbers and Variables

Numbers with Exponents

For very big numbers or very small numbers R uses the following scheme:

- **1.2e3 means 1200 because the e3 means 'move the decimal point 3 places to the right';**
- **1.2e-2 means 0.012 because the e-2 means 'move the decimal point 2 places to the left';**
- **3.9+4.5i is a complex number with real (3.9) and imaginary (4.5) parts, and i is the square root of −1.**

Basic Operations in R
Variable Names

- Variable names in R are case-sensitive so x is not the same as X;
- Variable names should not begin with numbers (e.g. 1x) or symbols (e.g. %x);
- Variable names should not contain blank spaces: use back.pay/back_pay (not back pay).

Basic Operations in R
Built-in Functions

Log and Exp

The log function gives logs to the base e = 2.718282.

```
> log(10)
[1] 2.302585
> exp(1)
[1] 2.718282
```

If you want logs to the base 10, then there is a separate function.

```
> log10(6)
[1] 0.7781513
```

Suppose you want log to base 3 of 9.

```
> log(9, base = 3)
[1] 2
```

Basic Operations in R
Assignment and Removement

Assign Values and Remove Variables

> x <- 3

where the gets arrow "<-" is a combination of the less than (<) and minus (-) signs. You could also write

> 3 -> x

to express that "3 is assigned to x."

Or

>x=3

The rm command may be used to explicitly remove a variable. For example, rm(x) removes the variable x.

Statistical Applications
Data Summaries

Mean and Variance

Let's randomly generate some data from a normal distribution.

```
> x = rnorm(100, mean = 3, sd = 3)
```

Note that x is a vector with 100 random numbers. We can calculate the sample mean using

```
> mean(x)
```

We can also calculate the sample variance by

```
> var(x)
```

Accordingly, the sample standard deviation is

```
> sqrt(var(x))
```

or

```
> sd(x)
```

Statistical Applications
Data Summaries

Quantiles

Again, randomly generate some data from a normal distribution.

```
> x = rnorm(100, mean = 3, sd = 3)
```

We can calculate the sample quantiles using

```
> quantile(x)
```

which gives us the 0%(min), 25%(Lower quantile), 50%(median), 75%(Upper quantile), 100%(max).

We can also calculate a specific quantile by

```
> quantile(x,0.3)
```

```
> quantile(x,seq(0,1,0.2))
```

Statistical Applications
Boxplots

Randomly generate 100 normally distributed numbers.

```
> x = rnorm(100, mean = 3, sd = 3)
```

To get a boxplot, simply use

```
> boxplot(x)
```

Statistical Applications
Histogram

Randomly generate 100 normally distributed numbers.

```
> x = rnorm(100, mean = 3, sd = 3)
```

The histogram can be easily obtained by

```
> hist(x)
```

Add title and x label

```
> hist(x, main = "Distribution of x", xlab = "x")
```

Specify the the number of breaks.

```
> hist(x, breaks = 12)
```

Vary the size of the domain using the xlim option.

```
> hist(x, xlim = c(-2, 2))
```

Add some colors to the plot

```
> hist(x, border = "blue", col = "green")
```

Statistical Applications
Histogram

We can try to “add” the box plot to the histogram.

```
> hist(x)
> boxplot(x, horizontal = TRUE, at = median(x), add =
TRUE, axes = FALSE)
```

Statistical Applications
Probability Plots

We can use histograms to find densities

```
> hist(x, prob = TRUE)
> lines(density(x))
```

We can also use scatter plots to visualize different densities of various distributions.

```
> x = seq(from = -5, to = 5, by = 0.1)
> y = dnorm(x, mean = 0, sd = 2)
> plot(x, y, "l")
```

For t distribution, we plot its cdf by

```
> x = seq(-5, 5, 0.1)
> y = pt(x, df = 5)
> plot(x, y, "l")
```

Statistical Applications
Pie Chart & Stem-and-Leaf Diagram

We can use pie() to construct pie charts

```
> slices <- c(25, 43, 32)
> lbls <- c("Cumin", "Saffron", "Ginger")
> pie(slices, labels = lbls, main="Sales of Spices")
```

We can use stem() to construct pie charts

```
> x <- c(33,28,16,35,11,44,33,38)
> stem(x)
```

Statistical Applications
Pareto Chart

First, we need to install “qcc” package

```
> library(qcc)
```

Then, we can use pareto.chart() to construct the pareto chart

```
> defect <- c(80, 27, 66, 94, 33)
> names(defect) <- c("price code", "schedule date", "supplier code", "contact num.", "part num.")
> pareto.chart(defect, ylab = "Error frequency")
```

Statistical Applications
Bar Chart

We can use barplot() to construct bar chart

```
> sales <- c(25,43,32)
> barplot(sales, main="Sales of Spices", horiz=TRUE,
names.arg=c("Cumin", "Saffron", "Ginger"))
> barplot(sales, main="Sales of Spices", horiz=FALSE,
names.arg=c("Cumin", "Saffron", "Ginger"))
```

Statistical Applications
Time Series Plot

We can use ts() to convert an array into a time series object and then use plot() or plot.ts() to construct a time series plot

```
> seq <- seq(from = 1, to = 100, by = 1) + 10
> y <- seq + rnorm(100, sd = 5)
> timeseries <- ts(y, start=c(2000, 1), frequency = 4)
> plot(timeseries)
```

Statistical Applications
Normal Probability Plot

We can use qqnorm() to construct the normal probability plot (also called QQ plot when the hypothesized distribution is normal distribution)

```
> x = rnorm(100, mean = 3, sd = 3)
> qqnorm(x)
```

We can use qqline() to add a straight line

```
> qqline(x)
```

When data is not normal

```
> x<-rexp(1000,1)
> qqnorm(x)
> qqline(x)
```

Statistical Applications
Scatter Plots

Scatter plots provide a graphical view of the relationship between two sets of numbers.

```
> x = rnorm(100, mean = 3, sd = 3)
> err = rnorm(100, mean = 0, sd = 1)
> y = 3 * x + err
> plot(x, y)
```

Statistical Applications
Scatter Plots for Multi-variables

Use the Iris dataset as an example

The dataset is contained in R

```
> iris
```

Use the first 4 columns (sepal length/width, petal length/width) to construct a scatter plot

```
> plot(iris[1:4])
```