Data Programming with R

Karl Ho

9/18/2022

Table of contents

Preface

This course requires no prior experience in programming. Yet, if you have some programming experience (e.g. SPSS, Stata, HTML), it will be helpful. R is an interpreted languages. In other words, the programs do not need compilation but will run in an environment to get the outputs. In this course, that is RStudio.

All packages and accounts are free and supported by open sources. It is recommended students bring their own computers (not mobile device) running MacOS, Linux or Windows operating systems.

Recommended software and IDE's:

1. R version 4.2.1 or later (https://cran.r-project.org)
2. RStudio version 1.2.x (https://www.rstudio.com)
3. Text editor of own choice (e.g. Atom, Sublime Text, Ultraedit)

Recommended websites/accounts:

1. GitHub (https://github.com)
2. RStudio Cloud

1 Summary

This book is designed for the short course titled **Data Programming with R**, offered in the University of Texas at Dallas. It provides training for aspiring data scientists to program surrounding data using R. The primary goal is to build a comprehensive program preparing students in four major elements in Data Science:

1. Data collection
2. Data visualization
3. Data management
4. Data modeling

The major platform is R with the IDE (Integrated Development Environment) is Rstudio. However, the principles and applications can be used for other languages and platforms such as Python and Julia.

Part I

Introduction

2 Introduction

This chapter introduces the general principles for data programming or coding involving data. Data programming is a practice that works and evolves with data. Unlike the point-and-click approach, programming allows the user to manage most closely the data and process data in more effective manner. Programs are designed to be replicable, by user and collaborators. A data program can be developed and updated iteratively and incrementally. In other words, it is building on the culminated works without repeating the steps. It takes debugging, which is the process of identifying problems (bugs) but, in fact, updating the program in different situations or with different inputs when used in different contexts, including the programmer himself or herself working in future times.

2.1 Principles of Data Programming

Social scientists Gentzkow and Shapiro (2014) list out some principles for data programming.

1. Automation
    - For replicability (future-proof, for the future you)
2. Version Control
    - Allow evolution and updated edition
    - Use Git and GitHub
3. Directories/Modularity
    - Organize by functions and data chunks
4. Keys
    - Index variable (relational)
5. Abstraction
    - KISS (Keep in short and simple)
6. Documentation
    - Comments for communicating to later users

7. Management
    - Collaboration ready

2.2 Functionalities of Data Programs

A data program can provide or perform :

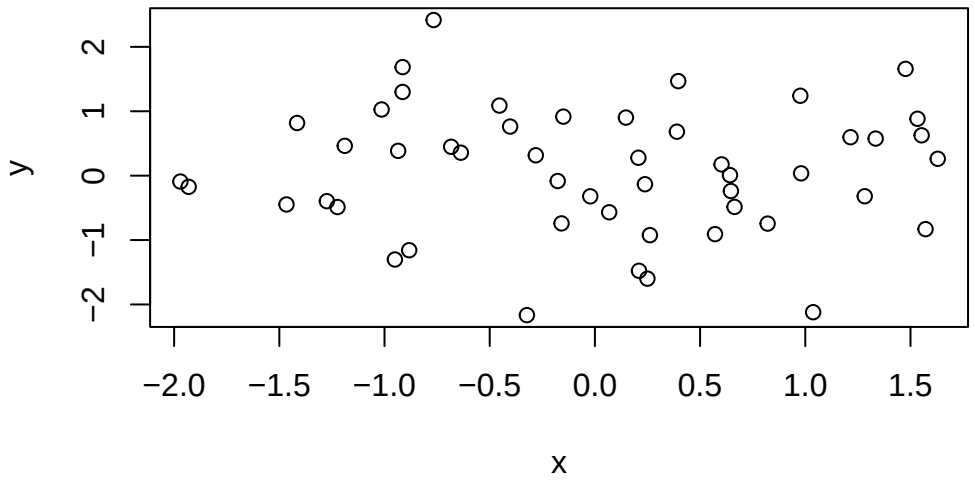
1. Data source
2. Documentation of data
3. Importing and exporting data
4. Management of data
5. Visualization of data
6. Data models

Sample R Programs:

R basics

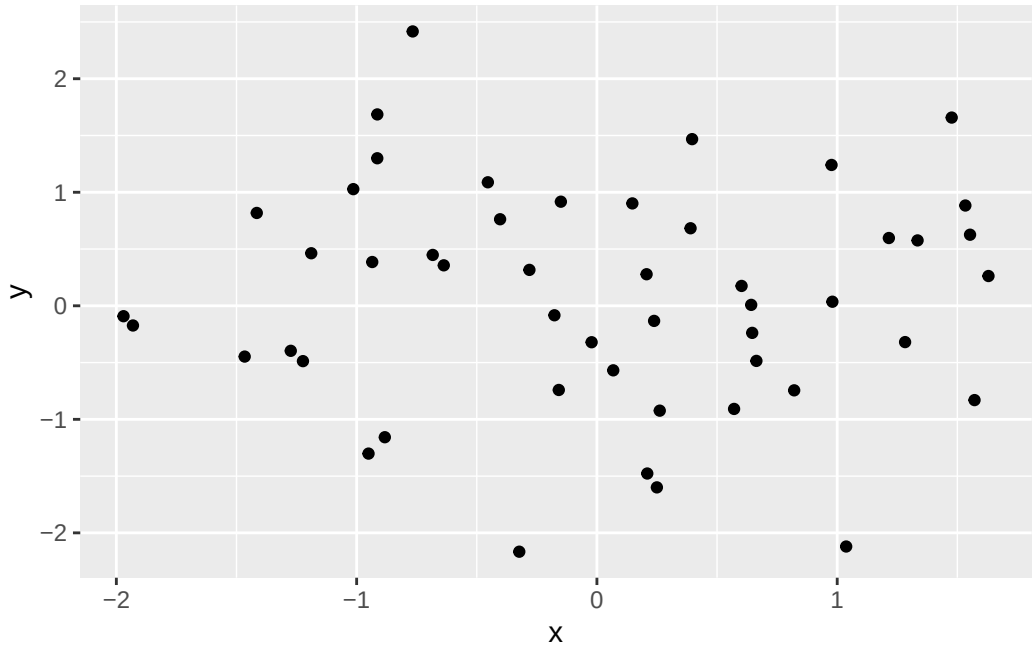
```
# Create variables composed of random numbers
x <-rnorm(50)
y = rnorm(x)

# Plot the points in the plane
plot(x, y)
```



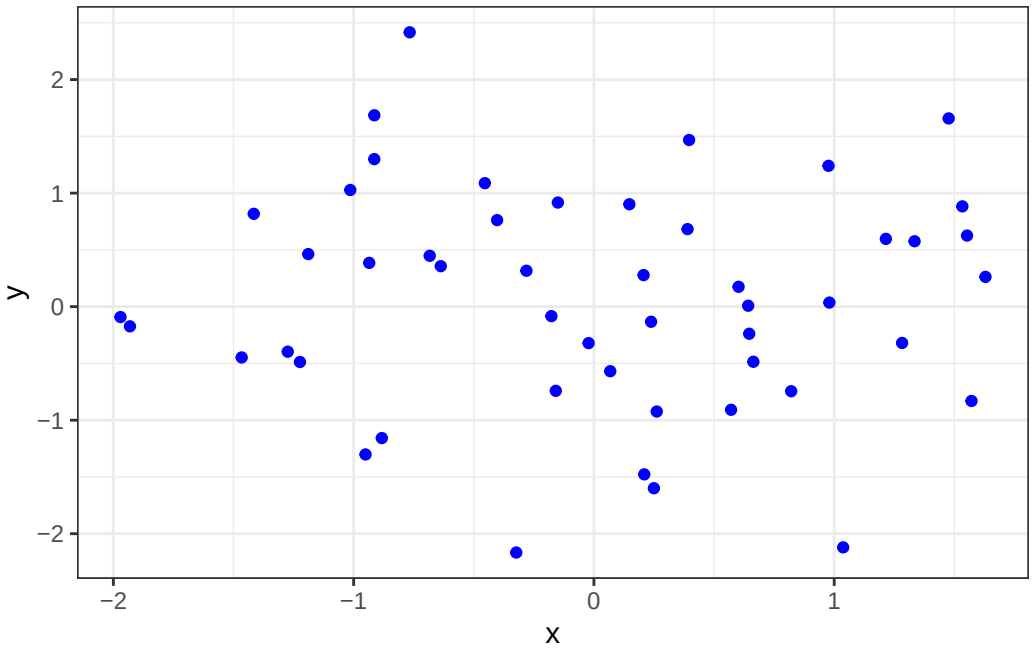
Using R packages

```
# Plot better, using the ggplot2 package
## Prerequisite: install and load the ggplot2 package
## install.packages("ggplot2")
library(ggplot2)
qplot(x,y)
```



More R Data Visualization

```
# Plot better better with ggplot2
library(ggplot2)
ggplot(,aes(x,y)) + theme_bw() + geom_point(col="blue")
```


2
1
0
−1
−2
y
−2
−1
0
1
x

3 R Basic operations

This chapter introduces basic R operations including installing packages, operators, loading libraries and writing functions.

3.1 Create a project

Before diving into R programming, it is highly recommended creating a project. It will organize all related objects and files under one roof. Version control can be applied for automatically saving history (log) and data objects. Projects can facilitate additional specialized works such as:

- Writing a book using Quarto or bookdown
- Creating a website or a blog using Quarto
  - Highly recommended to use this feature to create personal website
- Software development
- Data visualization using Shiny
- Manage data in database servers or other platform

However, to start R programming with a project helps better organize the process of data analytics. A program is not just running a procedure but performing more data science tasks that last more than just one time execution. It scales!

For more detail, consult Using RStudio Projects

3.2 Operators

3.2.1 Assignment operators

<- and = are both the assignment operator where as = must be used as top level.

```
a <- 0
b = 1
```

|> is the base R "pipe" operator, feeding value (arguement) into a function.

```
1:5 |> sum()
```

```
[1] 15
```

The other pipe operator %>% (package magrittr), which is more commonly used especially in tidyverse. Using %>%, the argument can be piped into the function without ().

```
library(magrittr)
1:5 %>% sum
```

```
[1] 15
```

3.2.2 Math operators

The common operators used in math are also applicable in the R environment.

- Arithmetic: +,-,*,/,^ (power)
- Logical: & (and), | (or), ! (Not)
- Relational: >,<, ==, >=, <=, !=

3.3 Package Management

3.3.1 Install packages

R comes with basic functions and demo datasets (e.g. mtcars, Titanic, iris). For additional functionalities or specialized functions, installing additional packages is needed. Use `install.packages("ISLR2")` for installation and it only needs be done the first time. However, for every new session, library() function is needed to call in (load) the package for remaining program.

3.3.2 Libraries

The `library()` function is used to load *libraries*, or groups of functions and data sets that are not included in the base `R` distribution. Basic functions that perform least squares linear regression and other simple analyses come standard with the base distribution, but more exotic functions require additional libraries. The *ISLR* book uses the `MASS` package, which is a very large collection of data sets and functions. The `ISLR2` package also includes the data sets associated with this book for demonstration.

```
library(MASS)
library(ISLR2)
```

```
Attaching package: 'ISLR2'
```

```
The following object is masked from 'package:MASS':

    Boston
```

If you receive an error message when loading any of these libraries, it likely indicates that the corresponding library has not yet been installed on your system. Some libraries, such as `MASS`, come with `R` and do not need to be separately installed on your computer. However, other packages, such as `ISLR2`, must be downloaded the first time they are used. This can be done directly from within `R`. For example, on a Windows system, select the `Install package` option under the `Packages` tab. After you select any mirror site, a list of available packages will appear. Simply select the package you wish to install and `R` will automatically download the package. Alternatively, this can be done at the `R` command line via `install.packages("ISLR2")`. This installation only needs to be done the first time you use a package. However, the `library()` function must be called within each `R` session.

3.3.3 Speed and organization

The general principle of using packages is: only load what you need! There are over 20,000 packages but the memory is limited. This article gives some comparison on different methods of using and loading packages or libraries. Again, using Project to manage your resources and it is advised to restart the R session (Session –> Restart R) to start with a clean slate for each project.

3.4 Writing Functions

As we have seen, R comes with many useful functions, and still more functions are available by way of R libraries. However, we will often be interested in performing an operation for which no function is available. In this setting, we may want to write our own function. For instance, below we provide examples of simple functions. The first one reads in the ISLR2 and MASS libraries, called LoadLibraries(). Before we have created the function, R returns an error if we try to call it.

```
LoadLibraries
```

```
Error in eval(expr, envir, enclos): object 'LoadLibraries' not found
```

```
LoadLibraries()
```

```
Error in LoadLibraries(): could not find function "LoadLibraries"
```

We now create the function. Note that the + symbols are printed by R and should not be typed in. The { symbol informs R that multiple commands are about to be input. Hitting *Enter* after typing { will cause R to print the + symbol. We can then input as many commands as we wish, hitting {*Enter*} after each one. Finally the } symbol informs R that no further commands will be entered.

```
LoadLibraries <- function() {
  library(ISLR2)
  library(MASS)
  print("The libraries have been loaded.")
}
```

Now if we type in LoadLibraries, R will tell us what is in the function.

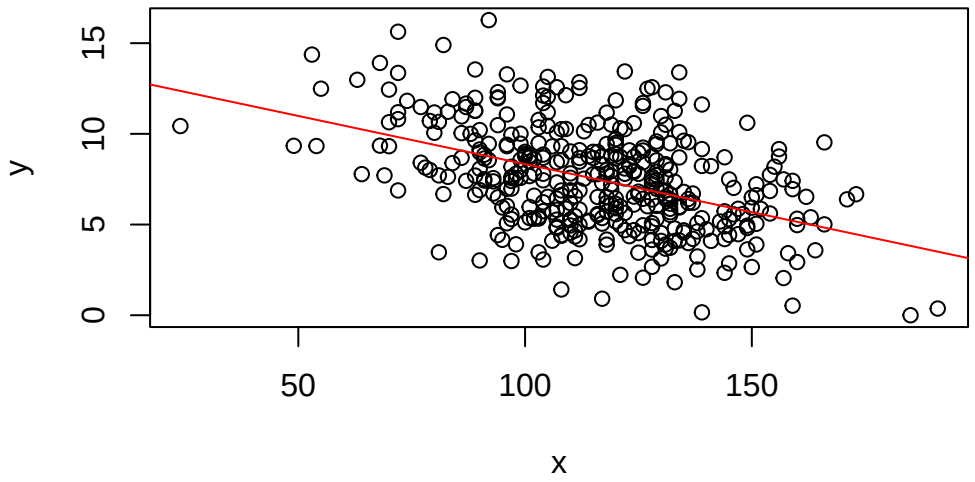
```
LoadLibraries()
```

```
[1] "The libraries have been loaded."
```

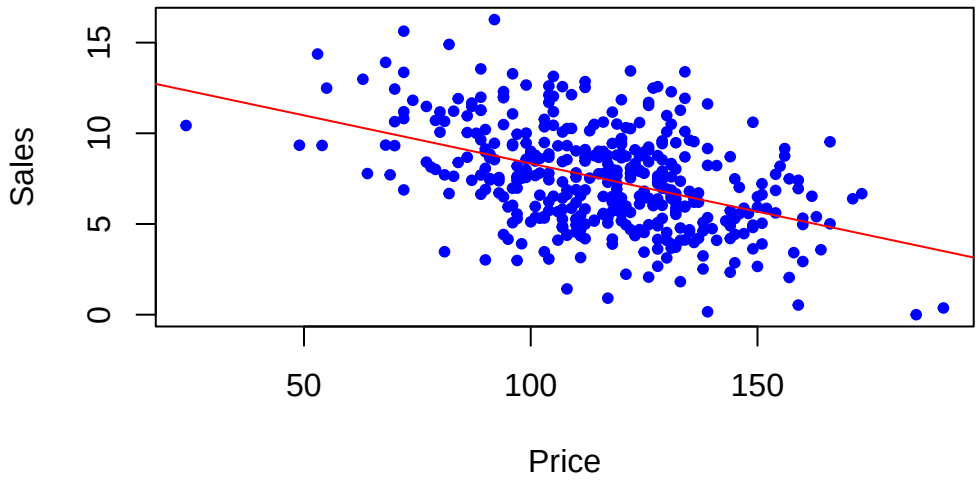
If we call the function, the libraries are loaded in and the print statement is output.

The following example demonstrates creating a function out of existing functions from different packages:

```r
###Writing R functions
## Combine the lm, plot and abline functions to create a one step regression fit plot func
regplot=function(x,y){
  fit=lm(y~x)
  plot(x,y)
  abline(fit,col="red")
}
attach(Carseats)
regplot(Price,Sales)
```




```r
## Allow extra room for additional arguments/specifications
regplot=function(x,y,...){
  fit=lm(y~x)
  plot(x,y,...)
  abline(fit,col="red")
}
regplot(Price,Sales,xlab="Price",ylab="Sales",col="blue",pch=20)
```



The following example creates a function to deal with package management:

```
# Create preload function
# Check if a package is installed.
# If yes, load the library
# If no, install package and load the library

preload<-function(x)
{
  x <- as.character(x)
  if (!require(x,character.only=TRUE))
  {
    install.packages(pkgs=x,  repos="http://cran.r-project.org")
    require(x,character.only=TRUE)
  }
}
```

Let's try preloading the package name **tidyverse** (be sure to wrap the name with double quotes " ":

3.5 Workshop 1

1. Create the following objects:
   - x <-rnorm(30)
   - y = rnorm(x)
2. Plot:

- histogram of y (hint: use the hist() function)
- Both x and y, using pch=20 (choose your own color using col=" ")

3. Check the environment
    1. Clean all objects using the following command:

```
rm(list=ls())
```

4. Alternatively, you can Ctrl+Shift+F10 (Mac: Command+Shift+0) to restart R session

3.5.1 Recommended R Resources:

- The R Journal
- Introduction to R by W. N. Venables, D. M. Smith and the R Core Team
- Introduction to R Seminar at UCLA
- Getting Started in Data Analysis using Stata and R by Data and Statistical Services, Princeton University

3.5.2 References:

Graham Williams 2011. *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge*

Part II

Communicative Programming using Quarto

4 Quarto

This chapter gives a brief introduction of communicative programming using Quarto. The idea is to incorporate data programming elements into a publishable document. Formats including PDF and HTML.

Part III

Data Collection

5 Data Collection with R

This chapter covers the basic methods of collecting data using R. Simple scrapping methods will be introduced like using *rvest* , *rtweet* and *academictwitteR*.

Part IV

Data Management

6 Data Management with R

This chapter focuses on the "Data" part in Data programming. In other words, we will cover methods of managing data not just locally but on cloud. It will cover the use of *sparklyr* to interact with Spark. Some basic concepts of relational database, database management systems and interfacing with database servers will be introduced.

Part V

Data Visualization

7 Data Visualization with R

7.1 What is Data Visualization?

Data visualization is to deliver a message from your data. It is like telling a story using the chart or data applications. Sometimes the data is huge or the story to too long to tell. Visualization provides an ability to comprehend huge amounts of data. The important information from more than a million measurements is immediately available.

Visualization often enables problems with the data to become immediately apparent. A visualization commonly reveals things not only about the data itself but also about the way it is collected. With an appropriate visualization, errors and artifacts in the data often jump out at you. For this reason, visualizations can be invaluable in quality control.

Visualization facilitates understanding of both large-scale and small-scale features of the data. It can be especially valuable in allowing the perception of patterns linking local features.

Visualization facilitates hypothesis formation, inviting further inquiries into building a theory (Colin Ware 2012). It is exploratory data anlaysis (EDA) but can also provide the tools for hypothesis confirmation.

7.2 Learn to read data

Edward Tufte is one of the earliest data scientists emphasizing visual thinking. He postulates that one should first learn to read data, before moving on to visualize. He suggests training the visual thinking, then preparing the educated eyes. His newest book is titled SEEING WITH FRESH EYES: MEANING, SPACE, DATA, TRUTH, vividly testifying his philosophy of connecting the human perception with the data message.

For Tufte, number one thing to learn about data visualization is to discard the default.

"If you're not doing something different, you're not doing anything at all." - Edward Tufte

Figure 7.1: **Edward Tufte**

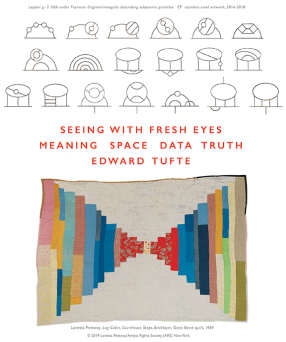


Figure 7.2: **Seeing with Fresh Eyes: Meaning, Space, Data, Truth**

7.3 References:

Graham Williams 2011. *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge*

Part VI

Data Modeling

8 Data Modeling with R

This chapter introduces methods of modeling data using R. Given the breadth of this topic, we will cover some basic machine learning models including:

- unsupervised learning
- supervised learning:
  - Classification
  - Regression
  - Tree-based models
- automated machine learning
  - H2O

9 What's next

This book will continue to incorporate new materials including new data science topics and developments.