GFreya′ R for Statistics

DS Glanzsche & Freya[1]

[1] A thank you or further information

Contents

Preface

For my future human Wife and our future biological daughters.

For my Divine Wife Freya the Goddess, and our daughters Catenary, Solreya, Mithra, Iyzumrae and Zefir.

For Lucrif and Znane too along with all the 8 Queens (Mischkra, Caldraz, Zalsvik, Zalsimourg, Hamzst, Lasthrim).

To Nature(Kala, Kathmandu, Big Tree, Sentinel, Aokigahara, Hoia Baciu, Jacob's Well, Mt Logan, etc) and my family Berlin: I have served, I will be of service.

To my current mentor Albert Silverberg and previous mentor Lucretia Merces.

To my dogs who always accompany me working in Valhalla Projection, go to Puncak Bintang or Kathmandu: Kecil, Browni Bruncit, Sweden Sexy, Cambridge Klutukk, Milan keng-keng, Piano Bludut, Barron and more will be adopted. To my cat who guard the home while I'm away with my dogs: London.

The one who moves a mountain begins by carrying away small stones - Confucius

A book for learning Statistics with R programming language that I am learning from zero. Helped by Freya the Goddess, Berlin, and Sentinel.

Figure 1: *FreyaCompass, I am inspired by Captain America who always bring compass with the love of his life’ picture, thus I created this, then proven by action, to let go of power and immortality for true love. Feels like an antique vintage magical compass, like a modem that connect internet to the world, this compass connects me on this planet to her in Valhalla.*

Figure 2: *Freya, thank you for everything, I am glad I marry you and I could never have done it without you.*

Date 22 Decembre 2021
Joyeux Noel my Femme, Freya
С Рождеством моя жена,
Фрея
Freya
Glanz
P.S
Hope you like the Pink Rose today.

Figure 3: *I paint her 3 days before Christmas in 2021.*

For critics and comments on the book can be sent through email to: *dsglanzsche@gmail.com*.

Chapter 1

Introduction and Installation of R

An opportunity missed is an opportunity wasted! - ***Seed (Suikoden II)***

THIS book is written on February 21st, 2025. Since 2022 we have been focusing on creating C++ codes for simulation and computation for Mathematics and Physics problems, they are all good, fast, but then we want to open a new horizon of knowledge, we read a book about R [1], it is said that we can do deep statistical analysis faster with R, given that the packages are already mature and the support is enormous, there is already a book series called 'The R book series' that can help statisticians and practitioners all over the world. I personally only know **Armadillo** library in C++ language that can compute mean, standard deviation, but then I think that basic statistics is not enough. If we want to do more with the data, e.g generalized linear models, generalized additive models, mixed-effects models, non-linear regression, time series analysis, multivariate statistics, survival analysis, then we can count on R language. R can produce beautiful plot and simulation with refined statistical analysis.

All the codes, CSV and book is available on this github' repository:
https://github.com/glanzkaiser/GFreya-R-for-Statistics

i. Introduction

[R*] The choice between R and C++ depends on your specific needs and the context in which you're working. If you want to focus on data science and data analysis use R. If you want to code embedded system, a micro controller, create game engines, create PC game (like GTA V, Skyrim, Quake 3, Doom 3, Assassin's Creed), desktop app then we use C++.

[R*] **The Pros of R**

1. Statistical Analysis: R is specifically designed for statistics and data analysis, making it ideal for data scientists and statisticians.
2. It has a vast collection of packages (like ggplot2, dplyr, and tidyverse) that simplify data manipulation and visualization.
3. R is generally easier to learn for beginners, especially those focused on data analysis.
4. There is a strong community around R, particularly in academia and research.

The Cons of R

1. R can be slower than C++ for computationally intensive tasks because it’s an interpreted language.

2. R abstracts many details away from the user, which can be limiting for low-level programming needs.

[R*] **The Pros of C++**

1. C++ is a compiled language, which typically results in faster execution times, making it suitable for performance-critical applications.

2. It offers more control over system resources and memory management, which is beneficial for system-level programming or applications requiring optimization.

3. C++ can be used for a wide range of applications beyond data analysis, including game development, systems programming, and application development.

The Cons of C++

1. C++ has a steeper learning curve than R, particularly due to its syntax and concepts like pointers and memory management.

2. While there are libraries available (like Armadillo and Eigen), C++ is not as tailored for statistical analysis as R.

Choose C++ if you need high performance, are developing complex systems, or require fine control over system resources.

ii. Download and Installing R

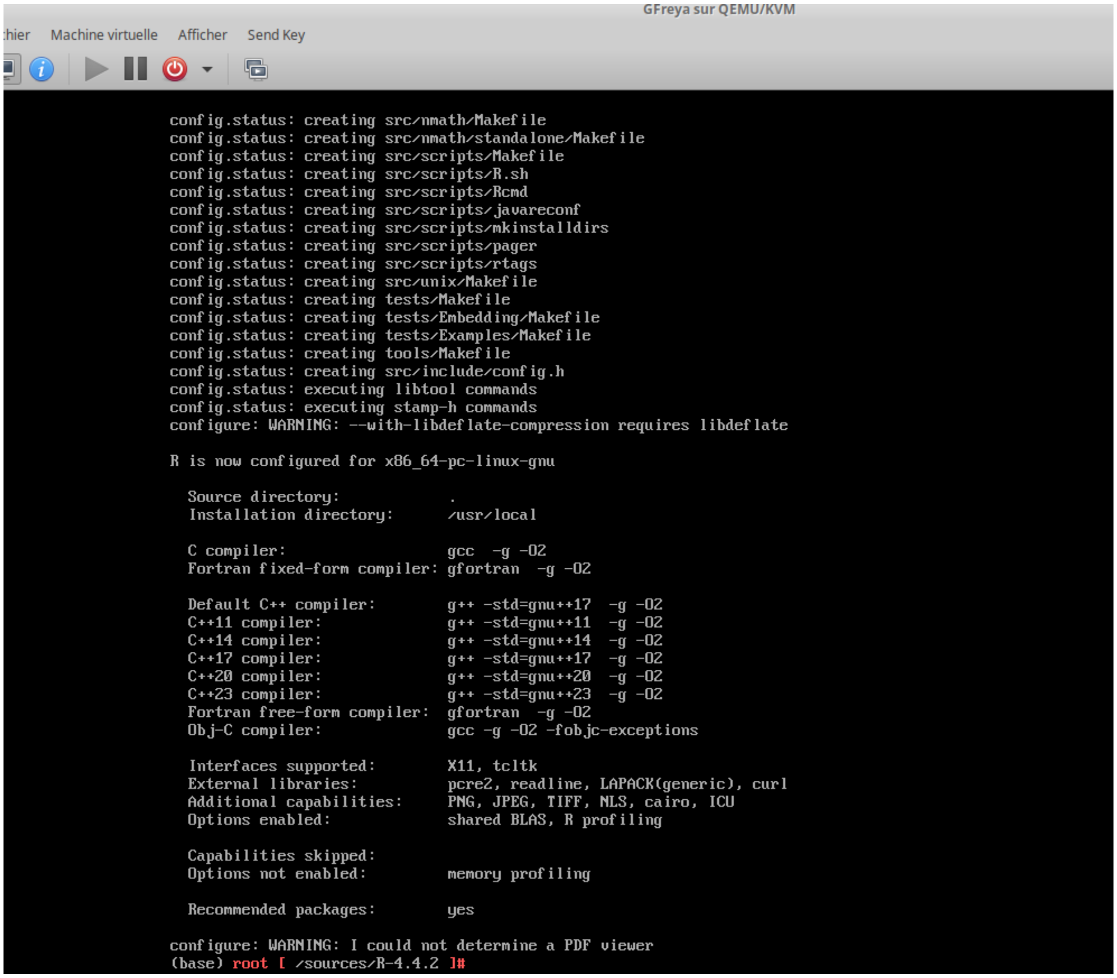
We are going to use **GFreya OS 1.8**, it is built based on Linux From Scratch and Beyond Linux From Scratch version 11.0 System V.

[R*] First download the newest R tarball from this link:
https://cran.r-project.org/src/base/R-4/R-4.4.2.tar.gz

we also have the tarball, you can check the github repo for this book here:
https://github.com/glanzkaiser/GFreya-R-for-Statistics/blob/main/Source%20Codes/R-4.4.2.tar.gz

[R*] After you download and then open terminal and type at the directory containing the downloaded R and type:

```
tar -xvf R-4.4.2.tar.gz
cd R-4.4.2
./configure
make
make install
```



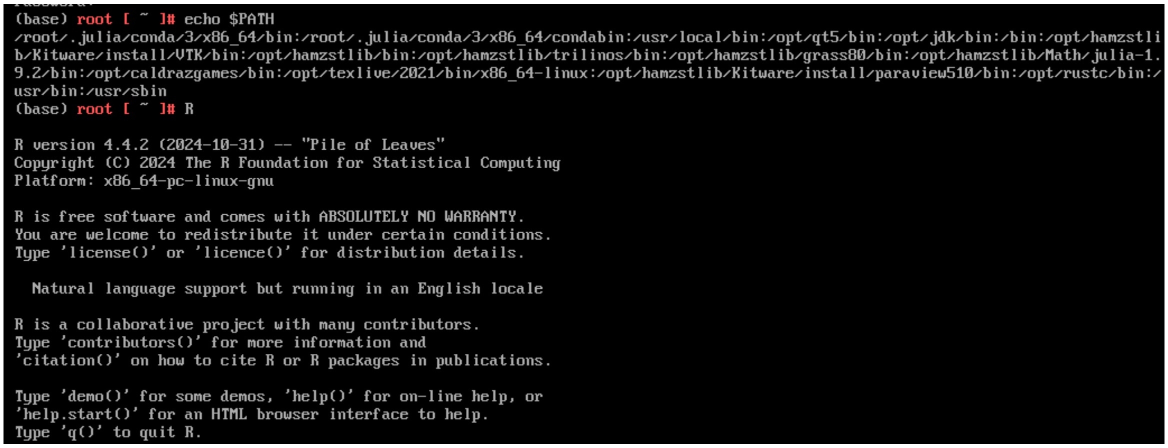
```
config.status: creating src/nmath/Makefile
config.status: creating src/nmath/standalone/Makefile
config.status: creating src/scripts/Makefile
config.status: creating src/scripts/R.sh
config.status: creating src/scripts/Rcmd
config.status: creating src/scripts/javareconf
config.status: creating src/scripts/mkinstalldirs
config.status: creating src/scripts/pager
config.status: creating src/scripts/rtags
config.status: creating src/unix/Makefile
config.status: creating tests/Makefile
config.status: creating tests/Embedding/Makefile
config.status: creating tests/Examples/Makefile
config.status: creating tools/Makefile
config.status: creating src/include/config.h
config.status: executing libtool commands
config.status: executing stamp-h commands
configure: WARNING: --with-libdeflate-compression requires libdeflate

R is now configured for x86_64-pc-linux-gnu

  Source directory:            .
  Installation directory:      /usr/local

  C compiler:                  gcc  -g -O2
  Fortran fixed-form compiler: gfortran  -g -O2

  Default C++ compiler:        g++ -std=gnu++17  -g -O2
  C++11 compiler:              g++ -std=gnu++11  -g -O2
  C++14 compiler:              g++ -std=gnu++14  -g -O2
  C++17 compiler:              g++ -std=gnu++17  -g -O2
  C++20 compiler:              g++ -std=gnu++20  -g -O2
  C++23 compiler:              g++ -std=gnu++23  -g -O2
  Fortran free-form compiler:  gfortran  -g -O2
  Obj-C compiler:              gcc -g -O2 -fobjc-exceptions

  Interfaces supported:        X11, tcltk
  External libraries:          pcre2, readline, LAPACK(generic), curl
  Additional capabilities:     PNG, JPEG, TIFF, NLS, cairo, ICU
  Options enabled:             shared BLAS, R profiling

  Capabilities skipped:
  Options not enabled:         memory profiling

  Recommended packages:        yes

configure: WARNING: I could not determine a PDF viewer
(base) root [ /sources/R-4.4.2 ]#
```

Figure 1.1: *If the **./configure** runs smoothly it will look like this.*

By default it is installed in **/usr/local/bin**, now you need to do one more important thing so you can call R from any directory.

Add the installation path of R to the $PATH environment variable
in GFreya OS go to root **cd /**
vim export

then press Esc and type **:wq** to save the contents and then quit, to quit without saving type **:q**, if you made a mistake and want to quit without saving type: **:q!** (these are some shell scripts when editing using vim editor).



```
(base) root [ ~ ]# echo $PATH
/root/.julia/conda/3/x86_64/bin:/root/.julia/conda/3/x86_64/condabin:/usr/local/bin:/opt/qt5/bin:/opt/jdk/bin:/bin:/opt/hamzstli
b/Kitware/install/VTK/bin:/opt/hamzstlib/bin:/opt/hamzstlib/trilinos/bin:/opt/hamzstlib/grass80/bin:/opt/hamzstlib/Math/julia-1.
9.2/bin:/opt/caldrazgames/bin:/opt/texlive/2021/bin/x86_64-linux:/opt/hamzstlib/Kitware/install/paraview510/bin:/opt/rustc/bin:/
usr/bin:/usr/sbin
(base) root [ ~ ]# R

R version 4.4.2 (2024-10-31) -- "Pile of Leaves"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
```

Figure 1.2: *Add the* ***/usr/local/bin*** *to the PATH environment variable to be able to run R from anywhere.*

add /usr/local/bin in PATH, then restart the computer and then check by typing in terminal:
echo $PATH
then you can now call R by typing:
R

[R*] How to install for Unix-like system can be seen from here:
https://cran.r-project.org/doc/manuals/r-devel/R-admin.html

[R*] **When Opening R**
Below the header you will see a blank line with a > symbol in the left hand margin. This is called the prompt. When working, you will sometimes see + at the left-hand side of the screen instead of >. This means that the last command you typed is incomplete.

To view the list of the already installed packages on your computer, type :
installed.packages()

If you want to update all installed R packages, type :
update.packages()

To update specific installed packages, say readr and ggplot2, type:
update.packages(oldPkgs = c('readr', 'ggplot2'))

To install a package, e.g. ggplot2, type:
install.packages('ggplot2')

you can then choose the CRAN (Comprehensive R Archive Network). mirror by typing a number representing which location for the mirror.

We can use the same function to install several R packages at once. In this case, we need to apply first the c() function to create a character vector containing all the desired packages as its items:
install.packages(c('readr', 'ggplot2', 'tidyr'))

Above, we’ve installed three R packages: the already-familiar readr, ggplot2 (for data visualization), and tidyr (for data cleaning).

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures. **install.packages(’tidyverse’)**

Chapter 2

Data Visualization

You don't need qualifications to make a difference. - ***Yun (Suikoden III)***

WE will start with a simple plotting then learning some basic and formulas in statistics and probability to create deep and more complex with more meaningful data visualization.

All the codes, CSV and book is available on this github' repository:
https://github.com/glanzkaiser/GFreya-R-for-Statistics

i. Scatter Plot with ggplot2

[R*] We will use CSV from the github' repository:
https://github.com/glanzkaiser/GFreya-R-for-Statistics/CSV/insurance.csv

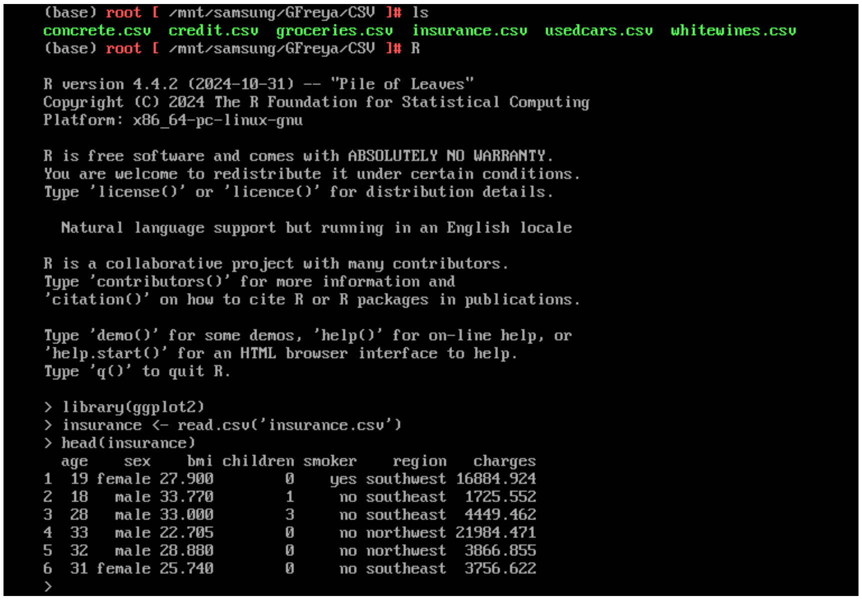
put this CSV in the working directory.

[R*] To open the desktop environment of GFreya OS, type:
startx

[R*] Open R from the working directory, from the current working directory open the terminal and type:
R

Load the necessary library:
library(ggplot2)

To import the data and look at the first six rows **insurance <- read.csv('insurance.csv')**



```
(base) root [ /mnt/samsung/GFreya/CSV ]# ls
concrete.csv  credit.csv  groceries.csv  insurance.csv  usedcars.csv  whitewines.csv
(base) root [ /mnt/samsung/GFreya/CSV ]# R

R version 4.4.2 (2024-10-31) -- "Pile of Leaves"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> library(ggplot2)
> insurance <- read.csv('insurance.csv')
> head(insurance)
  age    sex    bmi children smoker    region   charges
1  19 female 27.900        0    yes southwest 16884.924
2  18   male 33.770        1     no southeast  1725.552
3  28   male 33.000        3     no southeast  4449.462
4  33   male 22.705        0     no northwest 21984.471
5  32   male 28.880        0     no northwest  3866.855
6  31 female 25.740        0     no southeast  3756.622
>
```

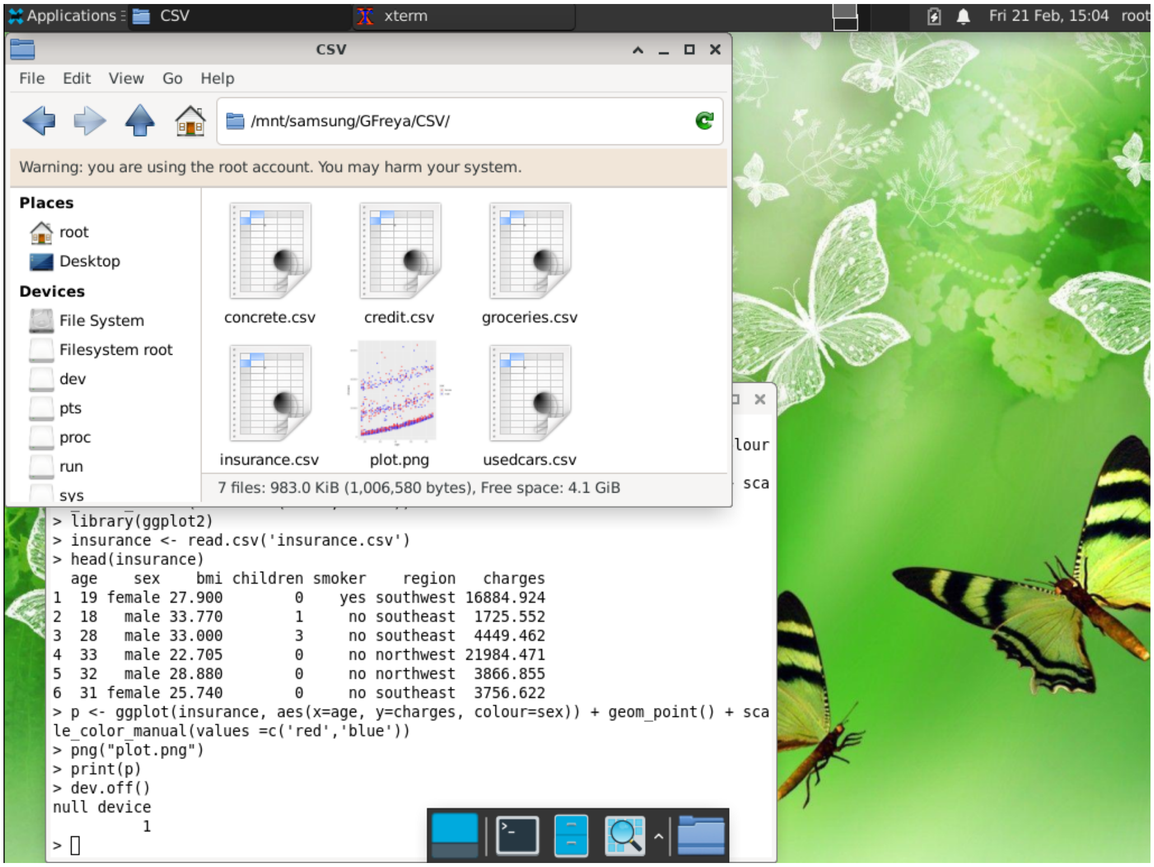
Figure 2.1: *To look at the top 6 rows of the data from CSV file.*

p <- ggplot(insurance, aes(x=age, y=charges, colour=sex)) + geom_point() + scale_color_manual(values =c('red', 'blue'))

Geoms are the geometric objects (points, lines, bars, etc.) that can be placed on a graph. They are added using functions that start with **geom_**.

In **ggplot2** graphs, functions are chained together using the + sign to build a final plot.

To save the plot as png, type:
png("plot.png")
print(p)
dev.off()



Warning: you are using the root account. You may harm your system.

7 files: 983.0 KiB (1,006,580 bytes), Free space: 4.1 GiB

```
> library(ggplot2)
> insurance <- read.csv('insurance.csv')
> head(insurance)
  age    sex    bmi children smoker    region   charges
1  19 female 27.900        0    yes southwest 16884.924
2  18   male 33.770        1     no southeast  1725.552
3  28   male 33.000        3     no southeast  4449.462
4  33   male 22.705        0     no northwest 21984.471
5  32   male 28.880        0     no northwest  3866.855
6  31 female 25.740        0     no southeast  3756.622
> p <- ggplot(insurance, aes(x=age, y=charges, colour=sex)) + geom_point() + sca
le_color_manual(values =c('red','blue'))
> png("plot.png")
> print(p)
> dev.off()
null device
          1
> 
```

Figure 2.2: *The process to plot the scatter plot with the x axis representing the age, the y axis representing the insurance charges and the color to separate male and female.*

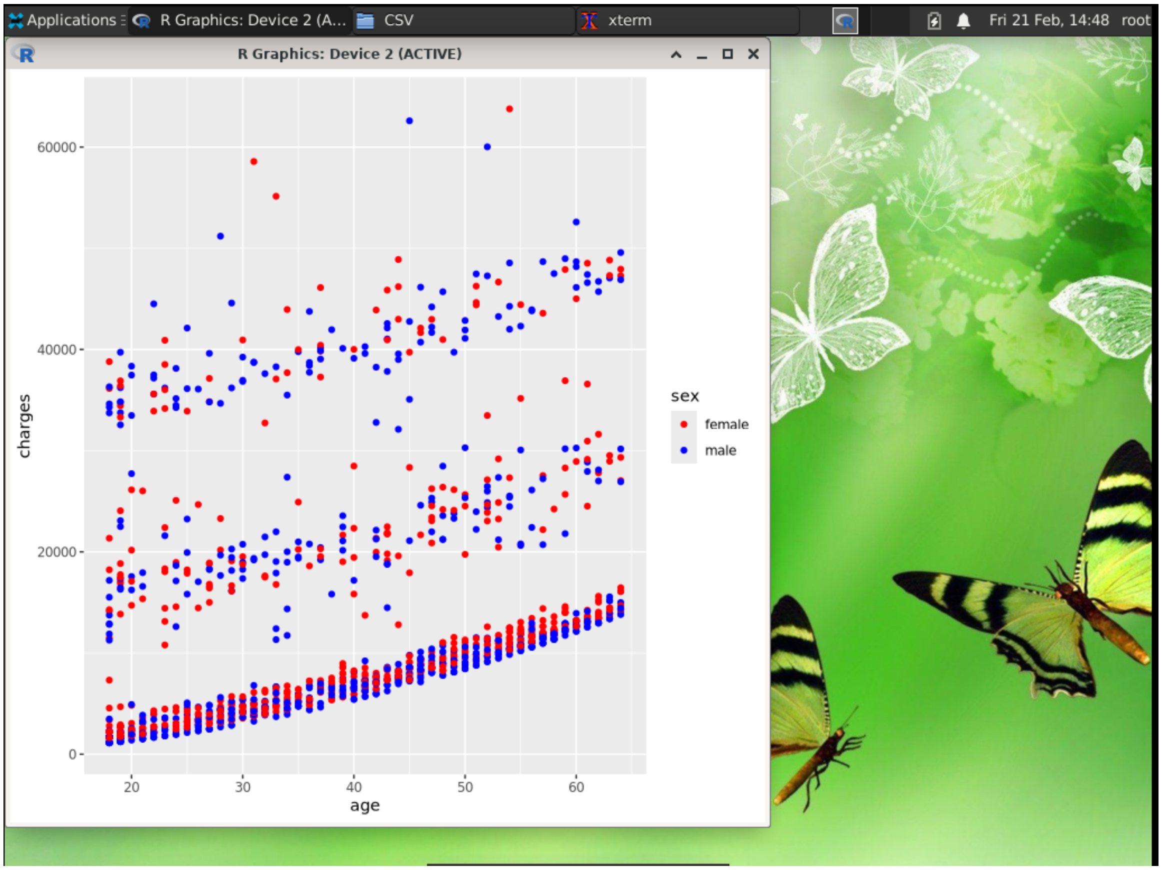


Figure 2.3: .

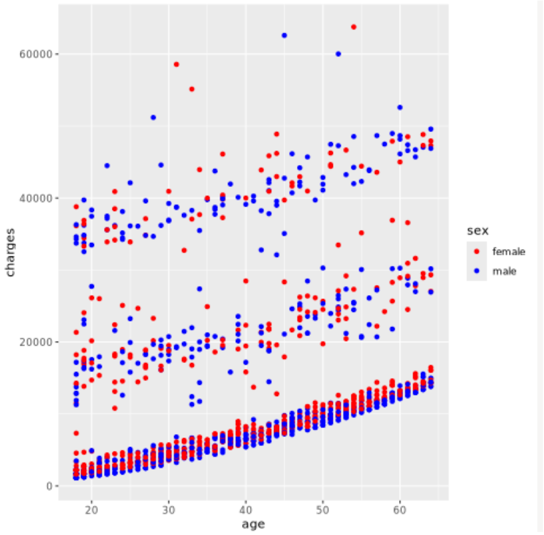


Figure 2.4: *The full picture.*

The whole code:

```
library(ggplot2)

insurance <- read.csv(’insurance.csv’)
p <- ggplot(insurance, aes(x=age, y=charges, colour=sex)) +
    geom_point() + scale_color_manual(values =c(’red’, ’blue’))

png("plot.png")
print(p)
dev.off()
```

R Code 1: *the first plot with ggplot2 (ch2-scatterplot.R)*

ii. Two Scatter Plots for Comparing with ggplot2

In thise section, we will create two scatter plots, more complex graph to explores the relationship between smoking, obesity, age, and medical costs using the data from the Medical Insurance Costs dataset / **insurance.csv**.

I won’t talk too many details and for further explanation you can read this book [2].

[R*] We will use CSV from the github’ repository:
https://github.com/glanzkaiser/GFreya-R-for-Statistics/CSV/insurance.csv

put this CSV in the working directory.

[R*] To open the desktop environment of GFreya OS, type:
startx

[R*] Open R from the working directory, from the current working directory open the terminal and type:
R

Load the necessary library:
library(ggplot2)

To import the data and look at the first six rows **insurance <- read.csv(’insurance.csv’)**

(alternative way to load / read **insurance.csv**) If you want to learn, there is this url that contains **insurance.csv**, it is on:
https://raw.githubusercontent.com/dataspelunking/MLwR/master
/Machine%20Learning%20with%20R%20(3rd%20Ed.)/Chapter06/insurance.csv

To obtain the insurance CSV data from a url page, type:
url <- "https://tinyurl.com/mtktm8e5"
insurance <- read.csv(url)

Now, beware that the column title for the last column is **expenses** instead of **charges**, so adjust that, or for the better, just stick with manually read the already available **insurance.csv** from the github’ repository of this book.

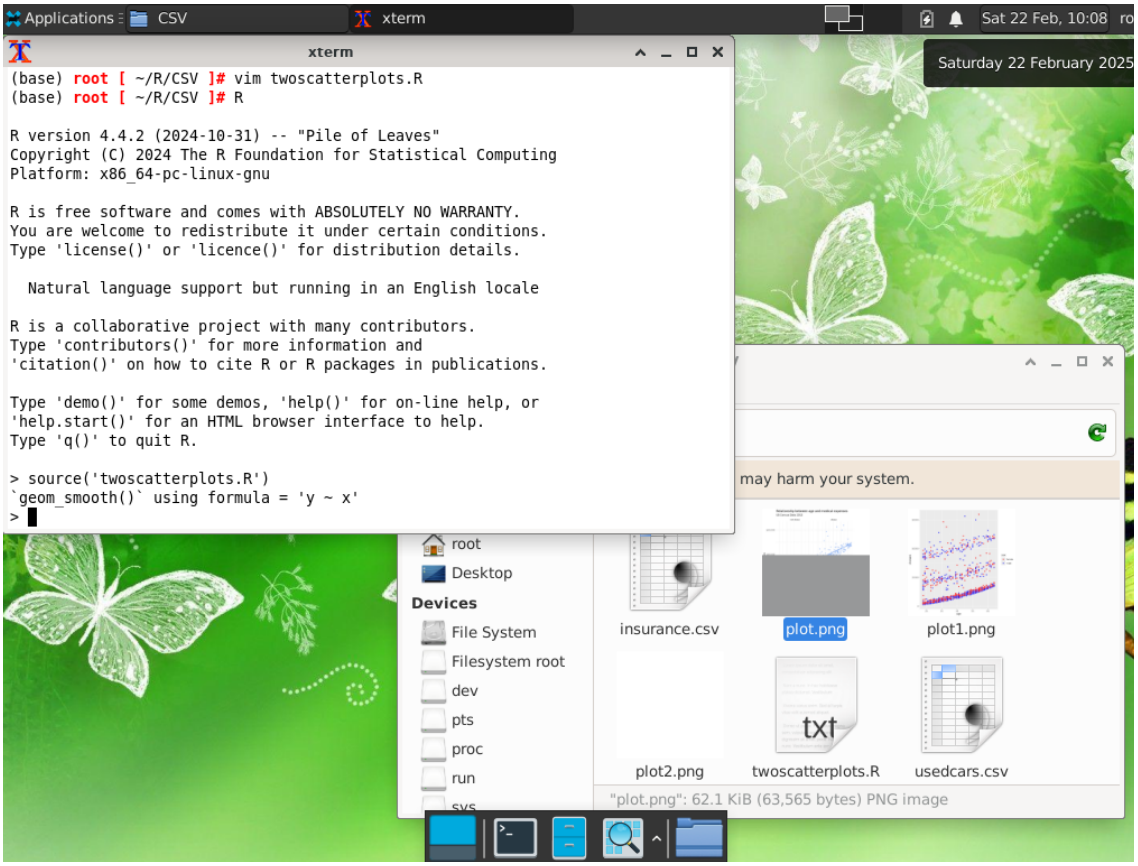
[R*] Now, without a lot of wasting time, you already know how to make a simple scatter plot, I will show the whole code to produce the plot in this section:

```
library(ggplot2)

insurance <- read.csv('insurance.csv')

# create an obesity variable
insurance$obese <- ifelse(insurance$bmi >= 30,"obese", "not
    obese")

p <- ggplot(data = insurance,mapping = aes(x = age,y = charges
    ,color = smoker)) + geom_point(alpha = .5) + geom_smooth(
    method = "lm", se = FALSE) +
scale_x_continuous(breaks = seq(0, 70, 10)) +
    scale_y_continuous(breaks = seq(0, 60000, 20000), label =
    scales::dollar) +
scale_color_manual(values = c("indianred3","cornflowerblue")) +
     facet_wrap(~obese) +
labs(title = "Relationship between age and medical expenses",
subtitle = "US Census Data 2013",
caption = "source: https://github.com/stedy/Machine-Learning-
    with-R-datasets/",
x = " Age (years)",
y = "Medical Expenses",
color = "Smoker?") +
theme_minimal()

png('plot.png')
print(p)
dev.off()
```

R Code 2: *two scatter plots with ggplot2 (ch2-twoscatterplots.R)*

Theme functions (which start with **theme_**) control background colors, fonts, grid-lines,legend placement, and other non-data related features of the graph.

Now, instead of typing one by one in the R console, we can be smart and open a text editor or vim editor or a notepad++ then just copy the whole codes above and save it with extension of **.R** and then we can use the source function, so put this **twoscatterplots.R** along with the CSV file if you wish to load it offline / from localhost then open the terminal at the current working directory and type:
R
source('twoscatterplots.R')

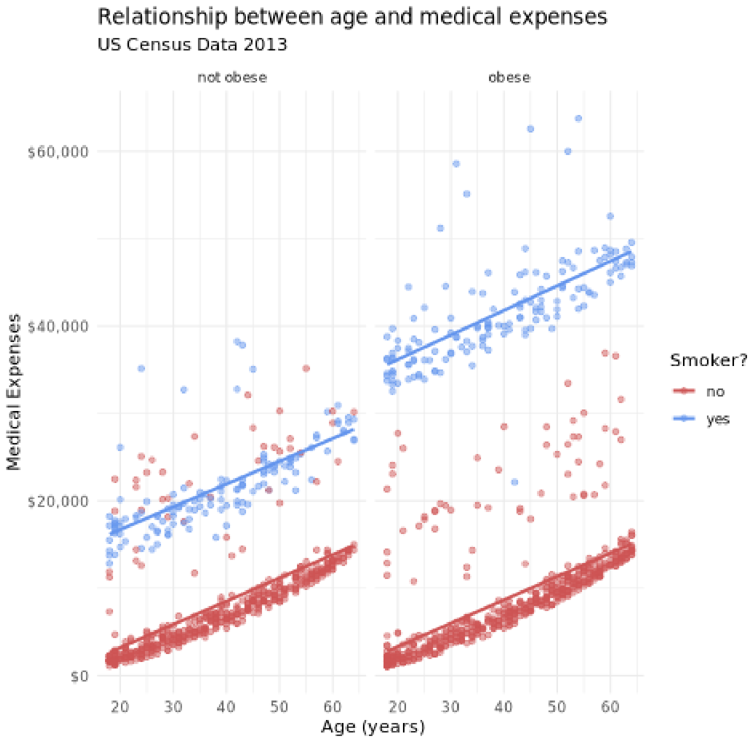


```
(base) root [ ~/R/CSV ]# vim twoscatterplots.R
(base) root [ ~/R/CSV ]# R

R version 4.4.2 (2024-10-31) -- "Pile of Leaves"
Copyright (C) 2024 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> source('twoscatterplots.R')
`geom_smooth()` using formula = 'y ~ x'
> 
```

may harm your system.

"plot.png": 62.1 KiB (63,565 bytes) PNG image

Figure 2.5: *Nothing comes in an instant.*



source: https://github.com/stedy/Machine-Learning-with-R-datasets/

Figure 2.6: *We can see that smokers and obese patients have higher medical charges / expenses.*

iii. Univariate Graphs for Categorical Variables: Bar Chart with ggplot2 and dplyr

Univariate graphs plots the distribution of data from a single variable. The variable can be categorical (e.g., race, sex, political affiliation) or quantitative (e.g., age, weight, income).

In this section we will plot a bar chart from the dataset Marriage that contains the marriage records of 98 individuals in Mobile County, Alabama (from the package **mosaicData**).

Pie charts are controversial in statistics. If your goal is to compare the frequency of categories, you are better off with bar charts (humans are better at judging the length of bars than the volume of pie slices).

[R*] We want to create a descending bar chart as it is easier to gain the knowledge from the data, most people’ brain work better by ordering. It is often helpful to sort the bars by frequency.

The **reorder** function is used to sort the categories by the frequency. The option **stat="identity"** tells the plotting function not to calculate counts, because they are supplied directly.

The minus sign in **reorder(race, -pct)** is used to order the bars in descending order.

```
# simple bar chart
library(ggplot2)
```

```
data(Marriage, package = "mosaicData")

# calculate number of participants in each race category
library(dplyr)

plotdata <- Marriage %>% count(race) %>% mutate(pct = n / sum(
    n), pctlabel = paste0(round(pct*100), "%"))

# plot the bars as percentages,
# in decending order with bar labels
p <- ggplot(plotdata, aes(x = reorder(race, -pct), y = pct)) +
geom_bar(stat="identity", fill="navyblue", color="black") +
geom_text(aes(label = pctlabel), vjust=-0.25) +
scale_y_continuous(labels = scales::percent) +
labs(x = "Race", y = "Percent", title = "Participants by race")

print(p)
```

R Code 3: *barchart with descending order (ch2-barchart.R)*

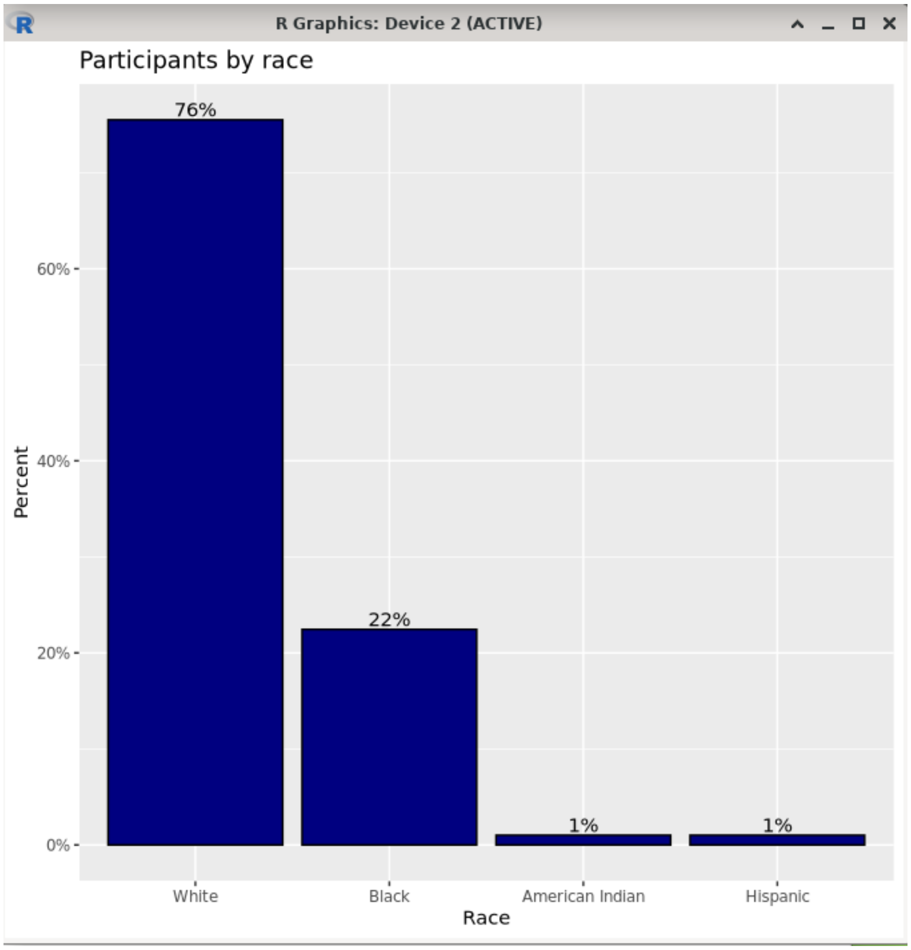


Figure 2.7: *The bar chart with rotated label.*

[R*] To solve a problem where category labels may overlap, we usually rotate the labels. Below is the code to rotate the label counterclockwise 45 degree.

```
# simple bar chart
library(ggplot2)

data(Marriage, package = "mosaicData")

# bar chart with rotated labels
p <- ggplot(Marriage, aes(x=officialTitle)) +
geom_bar(fill="navyblue", color="black") +
labs(x = "", y = "Frequency", title = "Marriages by officiate")
     +
theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```

R Code 4: *barchart with rotated label (ch2-barchart-rotatedlabels.R)*

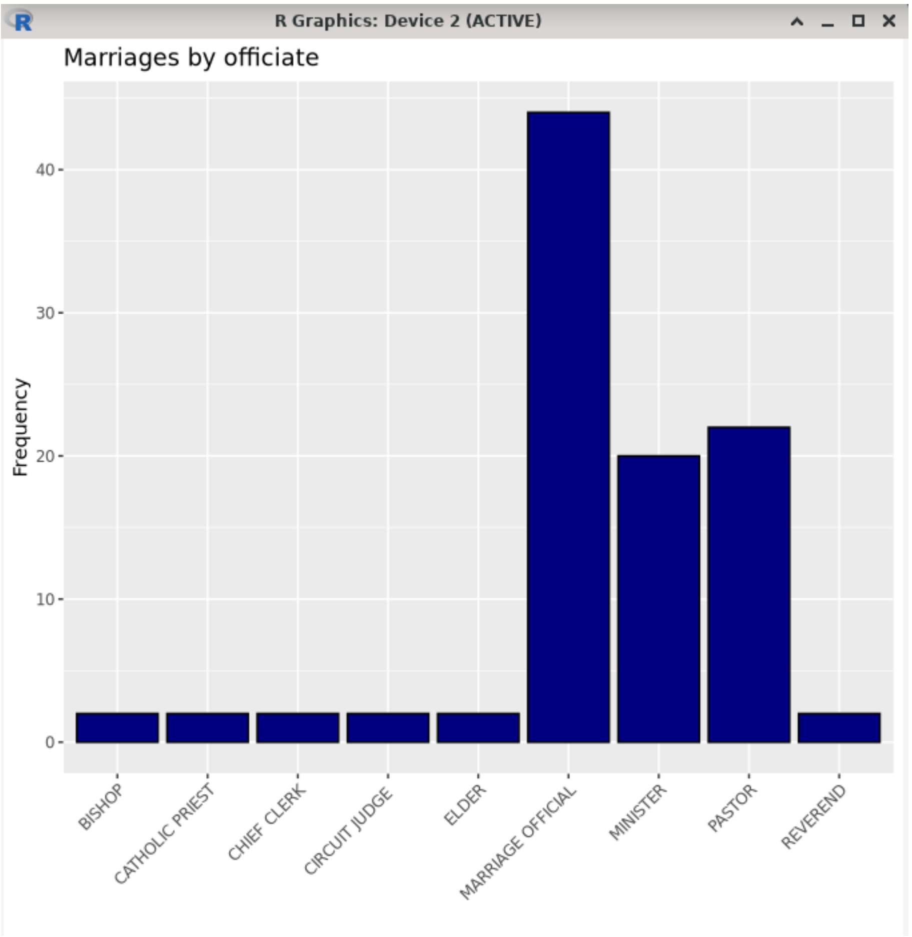


Figure 2.8: *The bar chart with rotated label.*

iv. Univariate Graphs for Categorical Variables: Tree Map with ggplot2

[R*] An alternative to a pie chart is a tree map. Unlike pie charts, it can handle categorical variables that have many levels. It is often used in The Economists magazine.

```
# simple bar chart
library(ggplot2)

data(Marriage, package = "mosaicData")

# bar chart with rotated labels
p <— ggplot(Marriage, aes(x=officialTitle)) +
geom_bar(fill="navyblue", color="black") +
labs(x = "", y = "Frequency", title = "Marriages by officiate")
     +
theme(axis.text.x = element_text(angle = 45, hjust = 1))

print(p)
```

R Code 5: *barchart with rotated label (ch2-treemap.R)*

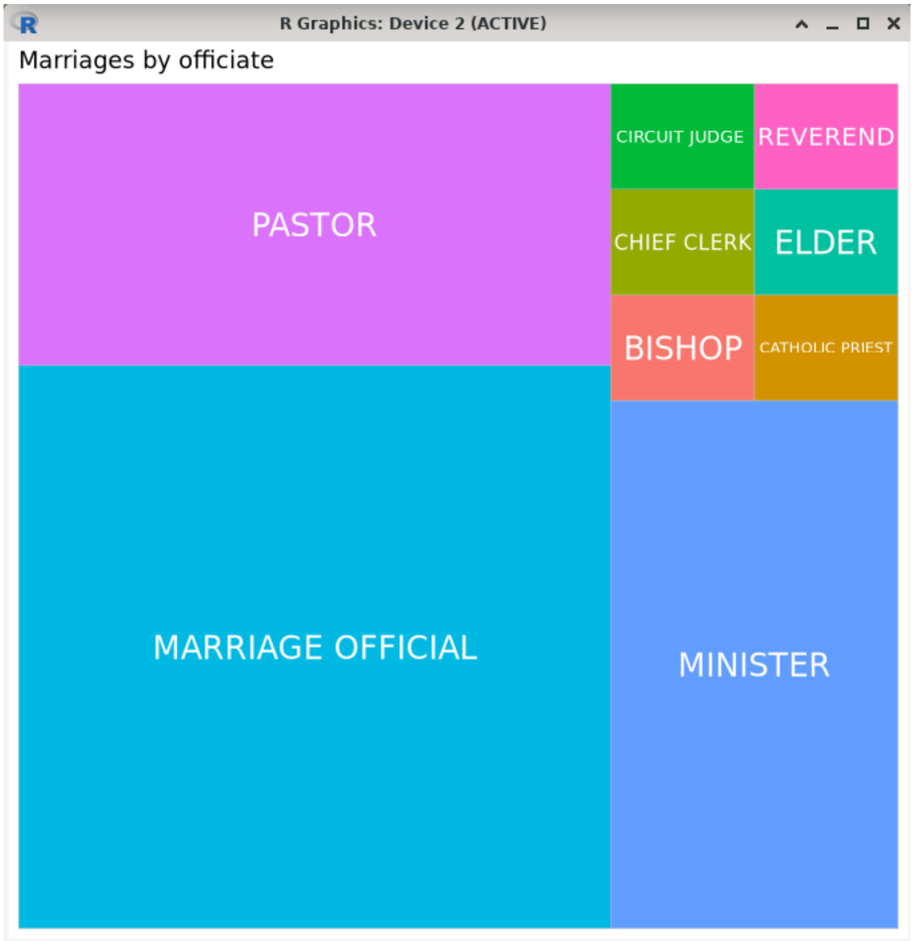


Figure 2.9: *The tree map of marriage officials with labels.*

v. Univariate Graphs for Quantitative Variables: Histogram with ggplot2

In the **Marriage** dataset, age is quantitative variable. The distribution of a single quantitative variable is typically plotted with a histogram, kernel density plot, or dot plot. In this section we will create a histogram.

Histograms [2] are the most common approach to visualizing a quantitative variable. In a histogram, the values of a variable are typically divided up into adjacent, equal width ranges (called bins), and the number of observations in each bin is plotted with a vertical bar.

One of the most important histogram options is bins, which controls the number of bins into which the numeric variable is divided (i.e., the number of bars in the plot). The default is 30, but it is helpful to try smaller and larger numbers to get a better impression of the shape of the distribution.

[R*] The first histogram is a simple histogram with **binwidth = 5**.

```
library(ggplot2)

data(Marriage, package = "mosaicData")

# displays the data with binwidth that are 5 years wide
p <— ggplot(Marriage, aes(x = age)) +
geom_histogram(fill = "navyblue", color = "white", binwidth =
    5) +
labs(title = "Participants by age", subtitle = "binwidth = 5
    years", x = "Age")

print(p)
```

R Code 6: *histogram with binwidth 5 (ch2-histogram.R)*

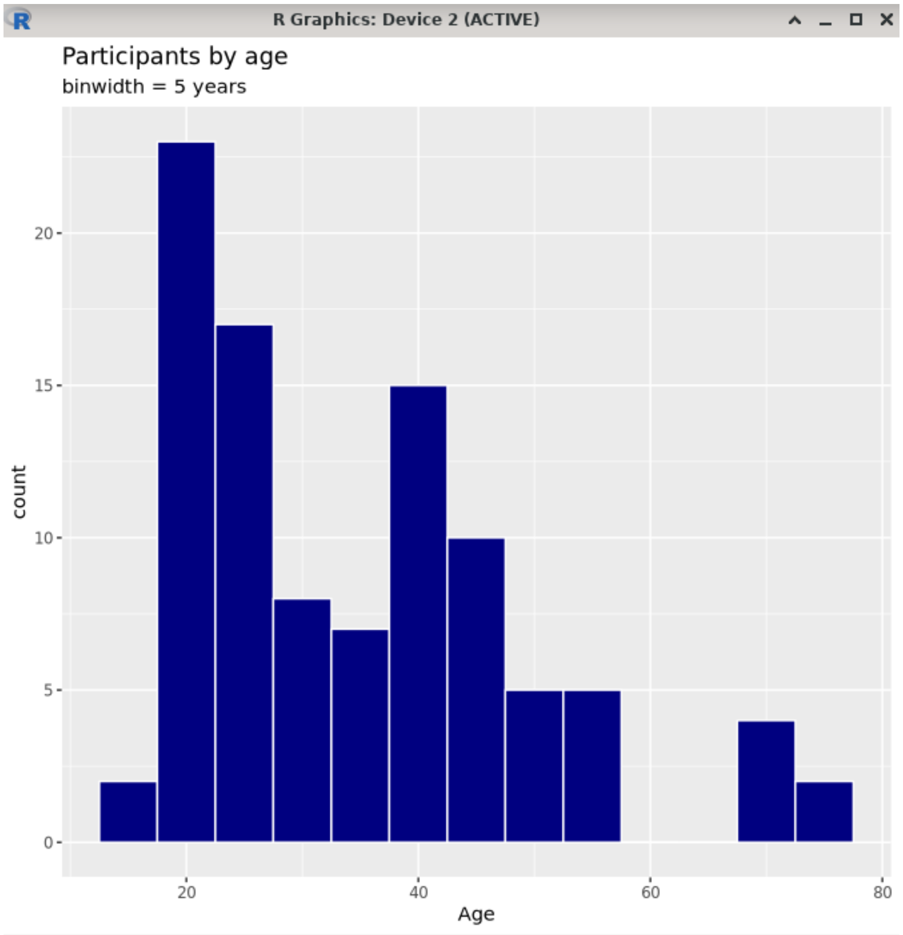


Figure 2.10: *The histogram with binwidth=5.*

[R*] The second histogram plot the histogram with percentages on the *y*-axis.

```
library(ggplot2)

data(Marriage, package = "mosaicData")

# plot the histogram with percentages on the y—axis
library(scales)

p <— ggplot(Marriage, aes(x = age, y = after_stat(count/sum(
    count)))) +
geom_histogram(fill = "navyblue", color = "white", binwidth =
    5) +
labs(title="Participants by age", y = "Percent", x = "Age") +
scale_y_continuous(labels = percent)

print(p)
```

R Code 7: *histogram with percentages on y axis (ch2-histogram.R)*

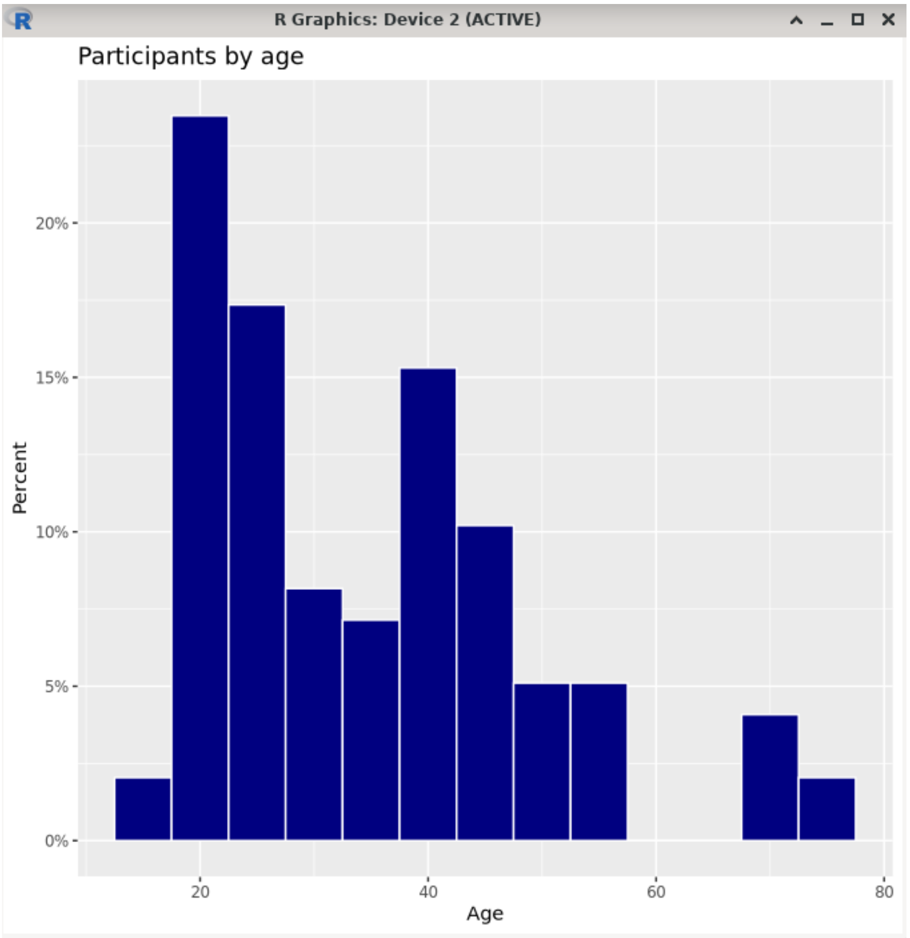
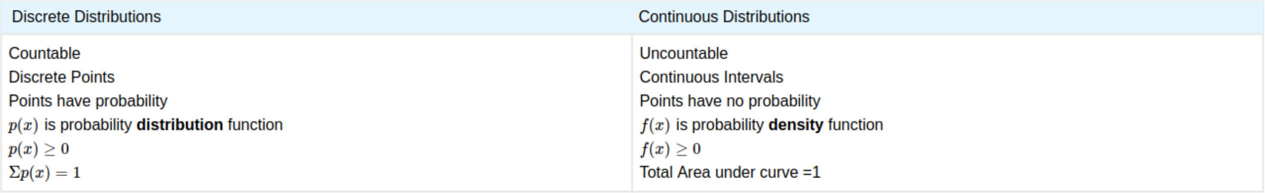


Figure 2.11: *The histogram with percentages on the y-axis.*

vi. Univariate Graphs for Quantitative Variables: Kernel Density Plot with ggplot2

An alternative to a histogram is the kernel density plot. Technically, kernel density estimation is a nonparametric method for estimating the probability density function of a continuous random variable.

A continuous random variable is a random variable that has only continuous values. Continuous values are uncountable and are related to real numbers. Examples: time, age, miles per gallon for a certain car.



Discrete Distributions	Continuous Distributions
Countable Discrete Points Points have probability $p(x)$ is probability **distribution** function $p(x) \geq 0$ $\Sigma p(x) = 1$	Uncountable Continuous Intervals Points have no probability $f(x)$ is probability **density** function $f(x) \geq 0$ Total Area under curve =1

Figure 2.12: *The similarities and differences between discrete and continuous distributions.*

In this section, we are trying to draw a smoothed histogram, where the area under the curve equals to one.

[R*] For this kernel density plot, the degree of smoothness is controlled by the bandwidth parameter **bw**. To find the default value for a particular variable, use the **bw.nrd0** function. Values that are larger will result in more smoothing, while values that are smaller will produce less smoothing.

```
library(ggplot2)

data(Marriage, package = "mosaicData")

p <- ggplot(Marriage, aes(x = age)) +
geom_density(fill = "navyblue", bw = 2) +
labs(title = "Participants by age",subtitle = "bandwidth = 2")

# default bandwidth for the age variable
# choosing a value that is less than bw.nrd0(Marriage$age) will
    resulting in less smoothing and more detail
bw.nrd0(Marriage$age)

png('plot.png')
print(p)
dev.off()
```

R Code 8: *kernel density plot with bandwidth 2 (ch2-kerneldensityplot.R)*

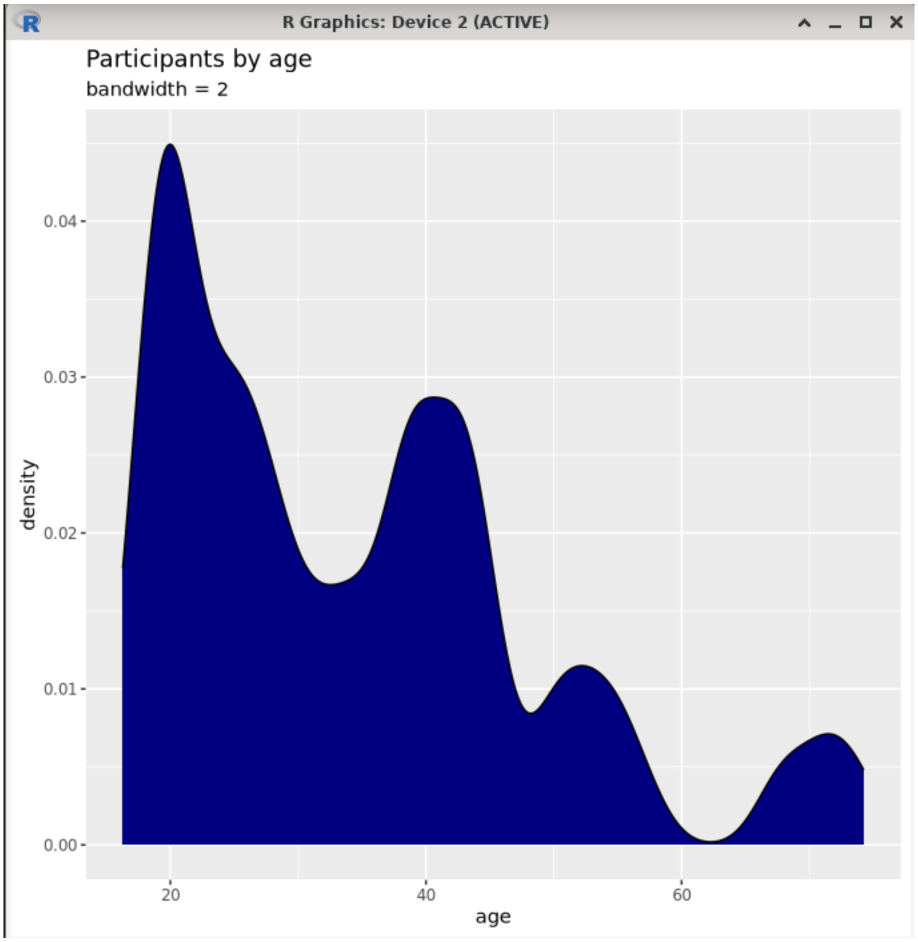


Figure 2.13: *The kernel density map with bandwidth = 2.*

Chapter 3

Classical Tests

Chapter 4

Statistical Modelling

Chapter 5

Regression

Chapter 6

Analysis of Variance

Chapter 7

Analysis of Covariance

Chapter 8

Generalized Linear Models

Chapter 9

Generalized Additive Models

Chapter 10

Non-linear Regression

Chapter 11

Tree Models

Chapter 12

Time Series Analysis

Chapter 13

Multivariate Statistics

Chapter 14

Spatial Statistics

Chapter 15

Survival Analysis

Chapter 16

Packages Needed to be Installed

These are packages that are used in this book:

1. ggplot2
2. mosaicData
3. dplyr
4. treemapify

Bibliography

[1] Crawley, Michael J., The R Book, John Wiley & Sons, England, 2007.

[2] Kabacoff, Robert, Modern Data Visualization with R, CRC Press, Boca Raton, USA, 2024.

[3] Lantz, Brett, Machine Learning with R 4th Edition, Packt, 2023.