Exploring, Visualizing, and Modeling Big Data with R

Okan Bulut
Christopher Desjardins

2019-04-01

Contents

Chapter 1

Exploring, Visualizing, and Modeling Big Data with R

Big Data in R

Explore, Visualize, and Model Big Data

Working with **BIG DATA** requires a particular suite of data analytics tools and advanced techniques, such as machine learning (ML). Many of these tools are readily and freely available in R. This full-day session will provide participants with a hands-on training on how to use data analytics tools and machine learning methods available in R to explore, visualize, and model big data.

The first half of our training session will focus on organizing (manipulating and summarizing) and visualizing (both statically and dynamically) big data in R. The second half will involve a series of short lectures on ML techniques (decision trees, support vector machines, and k-nearest neighbors), as well as hands-on demonstrations applying these methods in R. Examples will be drawn from the OECD's Programme for

International Student Assessment (PISA). As participants, you will get opportunities to work through several hands-on lab sessions throughout the day.

1.1 Who we are

1. **Okan Bulut – University of Alberta**
    - Associate Professor of educational measurement and psychometrics at the University of Alberta
    - 10+ years using the R programming language for data analysis and visualization
    - Specialized in the analysis and visualization of big data (mostly from large-scale assessments)
    - 5+ years teaching courses and workshops on statistics, psychometrics, and programming with R
    - https://github.com/okanbulut and <www.okanbulut.com>
    - **E-mail:** bulut@ualberta.ca
2. **Christopher D. Desjardins – University of Minnesota**
    - Research Associate at the Research Methodology Consulting Center.
    - R user since 2006 and contributer since 2009.
    - https://github.com/cddesja and https://cddesja.github.io/
    - **E-mail:** cdesjard@umn.edu

We also co-authored:

- two R packages – profileR for profile analysis of multivariate data and hemp for psychometric analysis of assessment data
- a recent book titled Handbook of Educational Measurement and Psychometrics Using R

Chapter 2

Introduction

2.1 What is big data?

When we handle a data-related problem, how do we know that we are actually dealing with "big data"? What is "big data"? What characteristics make a dataset big? The following three characteristics (three Vs of big data, source: TechTarget) can help us define the size of data:

1. **Volume**: The number of rows or cases (e.g., students) and the number of columns or variables (e.g., age, gender, student responses, response times)
2. **Variety**: Whether there are secondary sources or data that expand the existing data even further
3. **Velocity**: Whether real-time data are being used

2.2 Why is big data important?

Nowadays nearly every private and public sector of industry, commerce, health, education, and so forth are talking about big data. Data is a strategic and valuable asset when we know which questions we want to answer (see Bernard Marr's article titled Big Data: Too Many Answers, Not Enough Questions). Therefore, it is very important to identify the right questions at the beginning of data collection. More data with appropriate questions can yield quality answers that we can use for better decision-making. However, too much data without any purpose may obfuscate the truth.

Currently big data is used to better understand customers and their behaviors and preferences. Consider Netflix – one of the world's leading subscription services for watching movies and TV shows online. They use big data – such as customers' ratings for each movie and TV show and when customers subscribe/unsubscribe – to make better recommendations for existing customers and to convince more customers to subscribe. Target, a big retailer in the US, implements data mining techniques to predict pregnancies of their shoppers and send them a sale booklet for baby clothes, cribs, and diapers (see this interesting article). Car insurance companies analyze big data from their customers to understand how well their customers actually drive and how much they need to charge each customer to make a profit.

In education, there is no shortage of big data. Student records, teacher observations, assessment results, and other student-related databases make tons of information available to researchers and practitioners. With the advent of new technologies such as facial recognition software and biometric signals, now we get access to a variety of visual and audio data on students. In the context of educational testing and psychometrics, big data can help us to assess students more accurately, while continuously monitoring their progress via learning analytics. We can use log data and response times to understand students' engagement with the test, whether they were cheating, and whether they had pre-knowledge of the items presented on the test.

Figure 2.1: **Figure 1.** Three Vs of big data

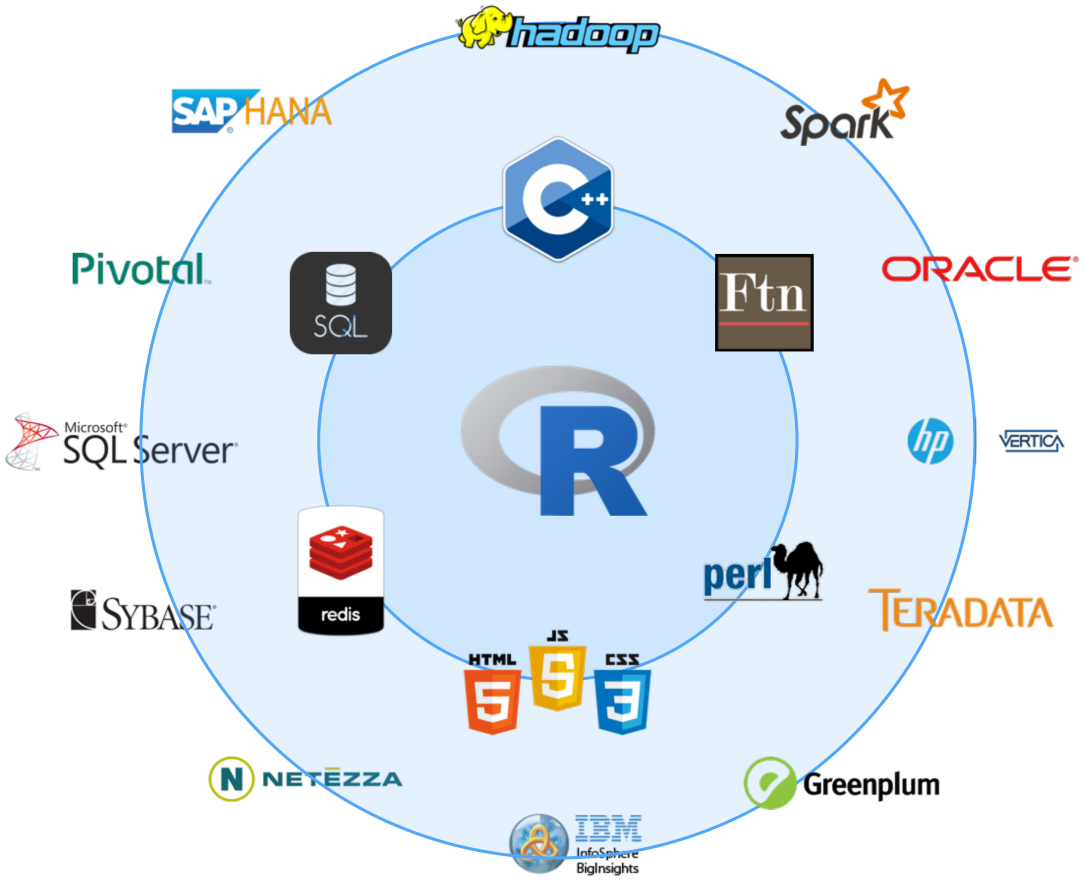


Figure 2.2: **Figure 2.** Other big data programs integrated with R

2.3 How do we analyze big data?

Big data analysis often begins with reading and then extracting the data. First, we need to read the data into a software program – such as R – and then manage it properly. Second, we need to extract a subset, sample, or summary from the big data. Due to its size, even a subset of the big data might itself be quite large. Third, we need to repeat computation (e.g., fitting a model) for many subgroups of the data (e.g., for each individual or by larger groups that combine individuals based on a particular characteristic). Therefore, we need to use the right tools for our data operations. For example, we may need to store big data in a data warehouse (either a local database or a cloud system) and then pass subsets of data from the warehouse to the local machine where we are analyzing the data.

R, maintained by the R Core Team, has its packages (collect of R functions) available on this The Comprehensive R Archive Network (CRAN). It used to be considered an *inadequate* programming language for big data (see Douglass Merril's article from 2012). Fortunately, today's R, with the help of RStudio and many data scientists, is capable of running most analytic tasks for big data either alone or with the help of other programs and programming languages, such as Spark, Hadoop, SQL, and C++ (see Figure 2). R is an amazing data science programming tool, it has a myriad statistical techniques available, and can readily translate the results of our analyses into colourful graphics. There is no doubt that R is one of the most preferred programming tool for statisticians, data scientists, and data analysts who deal with big data on a daily basis.

Some general suggestions on big data analysis include:

1. Obtain a strong computer (multiple and faster CPUs, more memory)
2. If memory is a problem, access the data differently or split up the data
3. Preview a subset of big data using a program, **not** the entire raw data.
4. Visualize either a subset of data or a summary of the big data, **not** the entire raw data.
5. Calculate necessary summary statistics manually, **not** for all variables in big data.
6. Delay computationally expensive operations (e.g., those that require large memory) until you actually need them.
7. Consider using parallel computing – parallel and foreach packages and cloud computing
8. Profile big tasks (in R) to cut down on computational time

```
start_time <- proc.time()

# Do all of your coding here

end_time <- proc.time()
end_time - start_time

# Alternatively,

system.time({

 # Do all of your coding here

})
```

During this training session, we will follow these steps and demonstrate how each one helps us explore, visualize, and model big data in R.

2.4 Additional resources

There are dozens of online resources and books on big data analysis. Here are a few of them that we recommend you check out:

- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). An introduction to statistical learning with applications in R. New York, NY: Springer. (Freely available from the authors' website: http://www-bcf.usc.edu/~gareth/ISL/index.html)
- Grolemund, G., & Wickham, H. (2016). R for data science. Sebastopol, CA: O'Reilly Media, Inc. (Freely available from the authors' website: http://r4ds.had.co.nz/)
- Baumer, B. S., Kaplan, D. T., & Horton, N. J. (2017). Modern data science with R. Boca Raton, FL: CRC Press.
- Romero, C., Ventura, S., Pechenizkiy, M., & Baker, de, R. S. J. (Eds.) (2011). Handbook of educational data mining. (Chapman and Hall/CRC data mining and knowledge discovery series). Boca Raton: CRC Press.
- DataCamp: https://www.datacamp.com/tracks/big-data-with-r
- RStudio: https://www.rstudio.com/resources/webinars/working-with-big-data-in-r/

2.5 PISA dataset

In this training session, we will use the 2015 administration of the OECD's Programme for International Student Assessment (PISA). PISA is a large-scale, international assessment that involves students, parents, teachers, and school principals from all over the world as participants. Every three years, PISA tests 15-year-old students from all over the world in reading, mathematics and science. The tests are designed to gauge how well the students master key subjects in order to be prepared for real-life situations in the adult world.

In addition to assessing students' competencies, PISA also aims to inform educational policies and practices for the participating countries and economies by providing additional information obtained from students, parents, teachers, and school principals through the questionnaires. Students complete a background questionnaire with questions about themselves, their family and home, and their school and learning experiences. School principals complete a questionnaire about the system and learning environment in schools. In some countries, teachers and parents also complete optional questionnaires to provide more information on their perceptions and expectations regarding students. In this training session, we specifically focus on the assessment data and the background questionnaire that all participating students are required to complete.

The 2015 administration of PISA involves approximately 540,000 15-year-old students from 72 participating countries and economies. During this training session, we will sometimes use the entire dataset or take a subset of the PISA dataset to demonstrate the methods used for exploring, visualizing, and modeling big data.

Chapter 3

Visualizing big data

One of the most effective ways to explore big data, interpret the variables, and communicate the results obtained from big data analysis to various audience is **data visualization**. When we deal with big data, we can benefit from data visualizations in many ways, such as:

- understanding the distributional characteristics of variables,
- detecting data entry issues,
- identifying outliers in the data,
- understanding relationships among variables,
- selecting suitable variables for data analysis (a.k.a., feature extraction),
- examining the outcomes of predictive models (e.g., accuracy and overfit), and
- communicating the results to various audience.

Developing effective visualizations requires identifying the goals and design of data analysis clearly. Sometimes we may already know the answers for some questions about the data; in other cases, we may want to explore further and understand the data in order to generate better insights into the next steps of data analysis. In this process, we need to consider many elements, such as types of variables to be used, axes, labels, legends, colors, and so on. Furthermore, if we aim to present the visualization to a particular audience, then we also need to consider the usability and interpretability of the visualization for the target audience.

The development of an effective data visualization typically includes the following steps:

1. Determine the goal of data visualization (e.g., exploring data, relationships, model outcomes)
2. Prepare the data (e.g., clean, organize, and transform data)
3. Identify the ideal visualization tool based on the goal of data visualization
4. Produce the visualization
5. Interpret the information in the visualization and present it to your target audience

Figure 3.1 shows some suggestions for visualizing data based on the type of variables and purpose of the visualization. In R, almost all of these visualizations can be created very easily, although preparing the data for these visualizations is sometimes quite tedious.

In this section of our session, we will review data visualization tools in R that can help us organize big data, interpret variables, and identify potential variables for predictive models. The first part will focus on data visualizations using the ggplot2 package. Furthermore, we will use other R packages (e.g., GGally, ggExtra, and ggalluvial) that expand the capabilities of `ggplot2` even further (also see http://www.ggplot2-exts.org/gallery/ for more extensions of `ggplot2`). In the second part, we will discuss web-based, interactive visualizations and dashboards using plotly.

As we review data visualization tools, we will also demonsrate how to use each visualization tool in R and produce sample plots and graphics using the **pisa** dataset. Furthermore, we will ask you to work on short

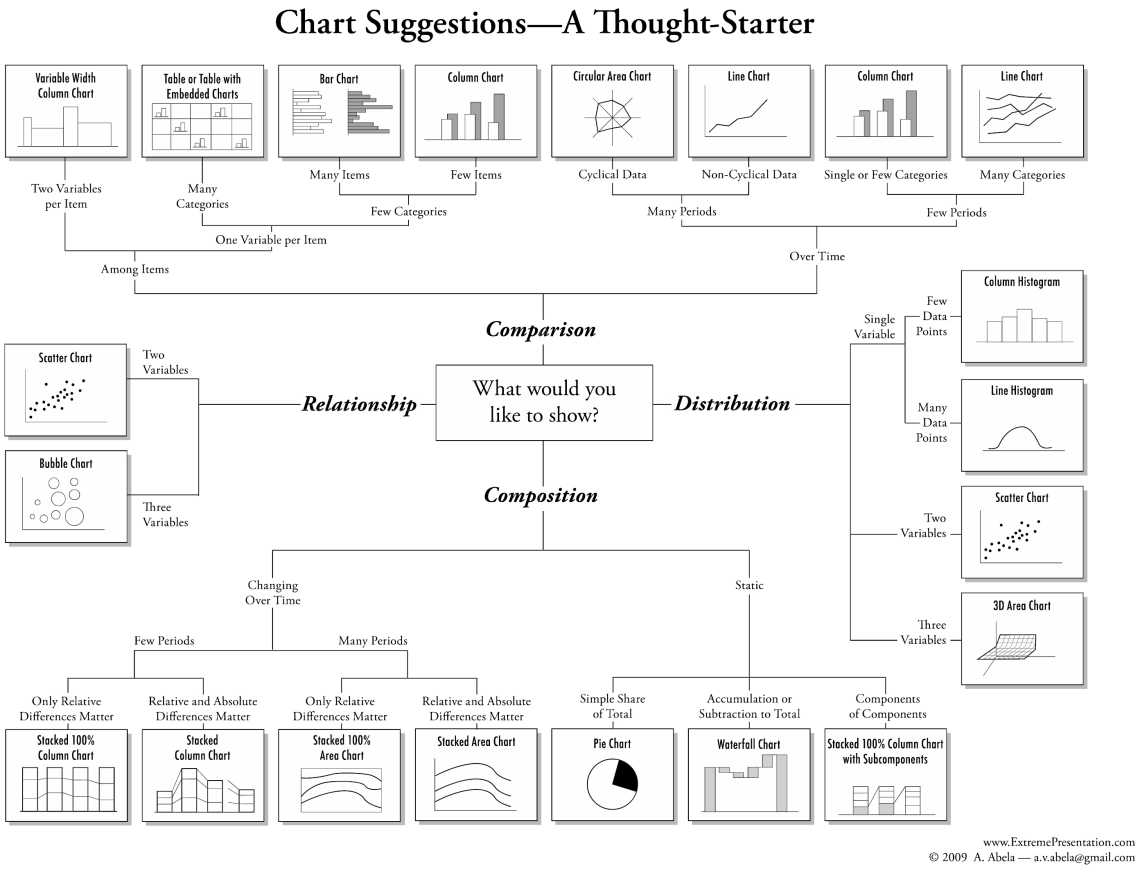


Figure 3.1: Chart suggestions (Source: <https://extremepresentation.com/>)

exercises where you will need to use the functions and packages presented in this section in order to generate your own plots and visualizations using the **pisa** dataset.

Before we begin, let's install and load all of the R packages that we will use in this section:

```
# Install and load the packages one by one.
install.packages("ggplot2")
install.packages("hexbin")
install.packages("GGally")
install.packages("ggExtra")
install.packages("ggalluvial")
install.packages("plotly")

library("ggplot2")
library("hexbin")
library("GGally")
library("ggExtra")
library("ggalluvial")
library("plotly")

# Or, just simply run the following to install and load all packages:
dataviz_packages <- c("ggplot2", "hexbin", "GGally", "ggExtra",
                      "ggalluvial", "plotly")
install.packages(dataviz_packages)
lapply(packages, require, character.only = TRUE)

# Load already installed packages
library("data.table")

# we will also use cowplot later in this session.
# Please install it but do not load it for now.
install.packages("cowplot")
```

3.1 Introduction to ggplot2

This section will demonstrate how to visualise your big data using `ggplot2` and other R packages that rely on `ggplot2`. We use 'ggplot2 because it is the most elegant and versatile visualization package in R. Also, it implements a simple grammar of graphics for building a variety of visualizations for either small or large data. This enables creating high-quality plots for publications and presentations easily, with minimal amounts of adjustments and tweaking.

A typical `ggplot2` template ranges from a few layers to many layers, depending on the complexity of the visualization of interest. Layers generate a plot and plot transformations within the plot. We can combine multiple layers using the + operator. Therefore, plots are built step by step by adding new elements in each layer. A simple `ggplot2` template is shown below:

```
ggplot(data = my_data,
       mapping = aes(x = var1, y = var2)) +
  geom_function()
```

where the `ggplot()` function uses the two variables (**var1** and **var2**) from a dataset (**my_data**), and draws a new plot based on a particular geom function (**geom_function**). Selecting the variables to be plotted is done through the aesthetic mapping (via the `aes()` function). Depending on the aesthetic mapping of

Table 3.1: Variables to be used in the data visualizataions

Variable	Description	Variable	Description
CNT	Country	BELONG	Sense of belonging to sc
OECD	OECD membership	EMOSUPS	Parents emotional suppo
CNTSTUID	Student ID	HOMESCH	ICT use outside of schoo
W_FSTUWT	Student weight in the PISA database	ENTUSE	ICT use outside of schoo
ST001D01T	Grade level	ICTHOME	ICT available at home
ST004D01T	Gender (female/male)	ICTSCH	ICT availability at schoo
ST011Q04TA	Possessing a computer at home	WEALTH	Family wealth
ST011Q05TA	Possessing educational software at home	PARED	Highest parental educati
ST011Q06TA	Having internet access at home	TMINS	Total learning time per
ST071Q02NA	Additional time spent for learning math	ESCS	Index of economic, socia
ST071Q01NA	Additional time spent for learning science	TDTEACH	Teacher-directed science
ST123Q02NA	Whether parents support educational efforts and achievements	IBTEACH	Inquiry based science ins
ST082Q01NA	Prefering working as part of a team to working alone	TEACHSUP	Teacher support in scien
ST119Q01NA	Wanting top grades in most or all courses	SCIEEFF	Science self-efficacy
ST119Q05NA	Wanting to the best student in class	math	Students math scores in
ANXTEST	Test anxiety	reading	Students reading scores
COOPERATE	Enjoying cooperation	science	Students science scores i

interest, we can split the plot, add colors by a group variable, change the labels for each axis, change the font size, and so on. The ‘`ggplot2` package offers many geom functions to draw different types of plots:

- `geom_point()` for scatter plots, dot plots, etc.
- `geom_boxplot()` for boxplots
- `geom_line()` for trend lines, time series, etc.

In addition, functions such as `theme_bw()` and `theme()` enable adjusting the theme elements (e.g., font size, font type, background colors) for a given plot. As we create plots in our examples, we will use some of these theme elements to make our plots look nicer.

An important caveat in visualizing big data is that the size of the dataset (*especially the number of rows*) and complexity level of the plot (e.g., additional lines, colors, facets) will influence how quickly and successfully `ggplot2` can render the desired plot. Nobody can absorb the meaning of thousands of data points presented on a single visualization. Therefore, in some cases we will need to find a way to cluster or reduce the magnitude of items to visualize before we render the visualization. Typically we can achieve this by:

- taking smaller samples from our big data, or
- summarizing our big data using categorical, group variables (e.g., gender, grade, year).

3.2 Marginal plots

We can use marginal plots to examine the distributions of individual variables in a large dataset. A typical marginal plot is a scatter plot that also has histograms or boxplots in the margins of the x- and y-axes. In this section, first we will create histograms and boxplots for the variables in the **pisa** dataset. Then, we will review other options where we will combine multiple variables and different types of plots in a single visualization.

To demonstrate data visualizations, we will first take a subset of the **pisa** dataset by selecting some countries and some variables of interest. The selected variables are shown below.

Here we filter our big data based on a list of countries (we called it `country`), create single scale scores for reading, math, and science by taking the mean of the plausible values for reading, math, and science in PISA 2015, and select the variables that we have just identified in Table 3.1.

```
country <- c("United States", "Canada", "Mexico", "B-S-J-G (China)", "Japan",
             "Korea", "Germany", "Italy", "France", "Brazil", "Colombia", "Uruguay",
             "Australia", "New Zealand", "Jordan", "Israel", "Lebanon")

dat <- pisa[, `:=` (
  math = rowMeans(pisa[, c(paste0("PV", 1:10, "MATH"))], na.rm = TRUE),
  reading = rowMeans(pisa[, c(paste0("PV", 1:10, "READ"))], na.rm = TRUE),
  science = rowMeans(pisa[, c(paste0("PV", 1:10, "SCIE"))], na.rm = TRUE)
  )][CNT %in% country,][,.(CNT, OECD, CNTSTUID, W_FSTUWT, sex, female,
                           ST001D01T, computer, software, internet,
                           ST011Q05TA, ST071Q02NA, ST071Q01NA, ST123Q02NA,
                           ST082Q01NA, ST119Q01NA, ST119Q05NA, ANXTEST,
                           COOPERATE, BELONG,  EMOSUPS, HOMESCH, ENTUSE,
                           ICTHOME, ICTSCH, WEALTH, PARED, TMINS, ESCS,
                           TEACHSUP, TDTEACH, IBTEACH, SCIEEFF,
                           math, reading, science)]
```

Next, we create additional variables by recoding some of the existing variables. The goal is to create some numerical variables out of the character variables in case we want to use them in the modeling stage.

```
# Let's create additional variables that we will use for visualizations
dat <- dat[, `:=` (
  # New grade variable
  grade = (as.numeric(sapply(ST001D01T, function(x) {
  if(x=="Grade 7") "7"
  else if (x=="Grade 8") "8"
  else if (x=="Grade 9") "9"
  else if (x=="Grade 10") "10"
  else if (x=="Grade 11") "11"
  else if (x=="Grade 12") "12"
  else if (x=="Grade 13") NA_character_
  else if (x=="Ungraded") NA_character_}))),
  # Total learning time as hours
  learning = round(TMINS/60, 0),
  # Regions for selected countries
  Region = (sapply(CNT, function(x) {
  if(x %in% c("Canada", "United States", "Mexico")) "N. America"
    else if (x %in% c("Colombia", "Brazil", "Uruguay")) "S. America"
    else if (x %in% c("Japan", "B-S-J-G (China)", "Korea")) "Asia"
    else if (x %in% c("Germany", "Italy", "France")) "Europe"
    else if (x %in% c("Australia", "New Zealand")) "Australia"
    else if (x %in% c("Israel", "Jordan", "Lebanon")) "Middle-East"
    }))
  )]
```

Now, let's see the number of rowns in the final dataset and how it looks like with all the selected variables.

```
# N count for the final dataset
dat[,.N] # 158,061 rows
```

```
## [1] 158061
```

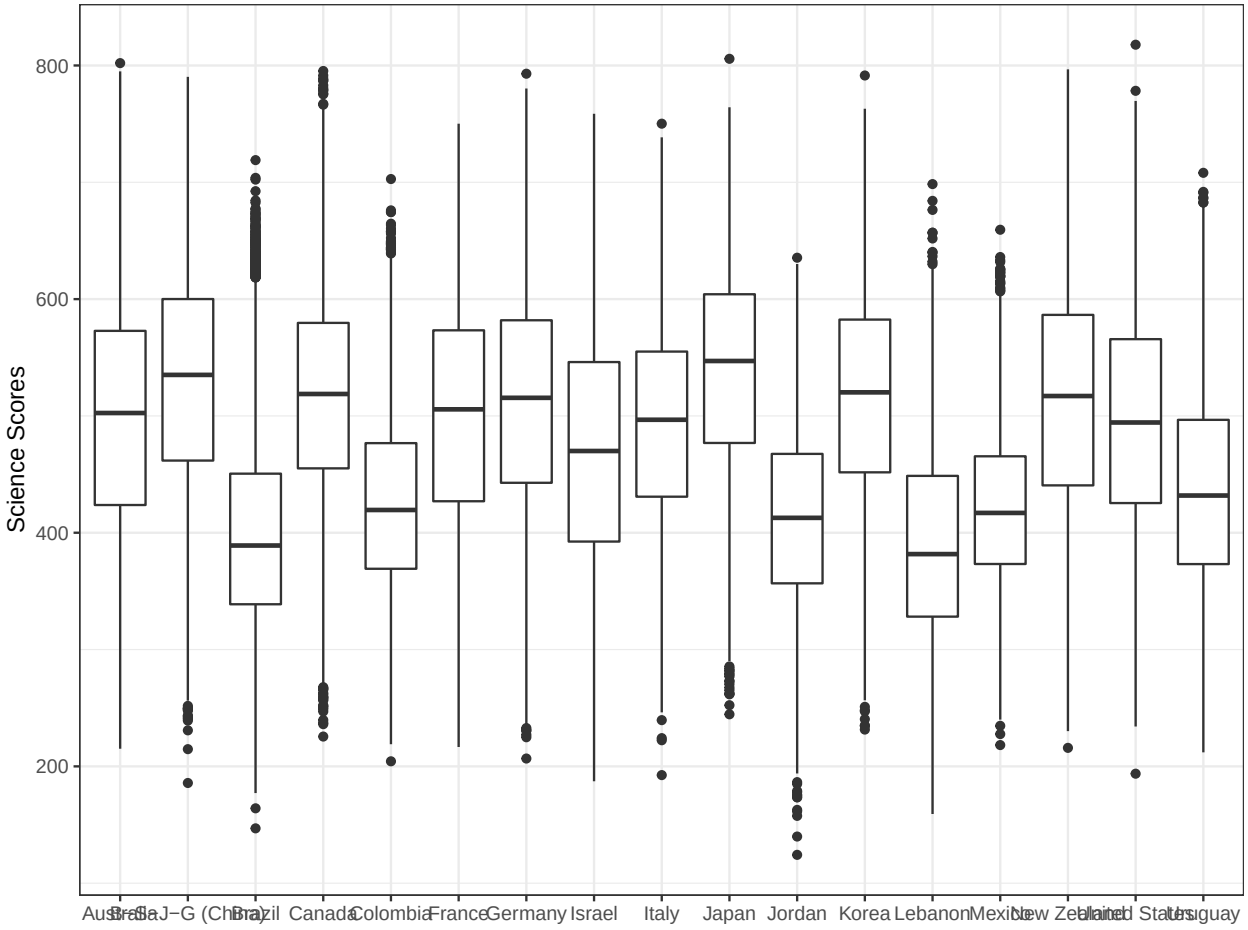
```
# Let's preview the final data
head(dat)
```

```
##          CNT OECD CNTSTUID W_FSTUWT    sex female ST001D01T computer
## 1: Australia  Yes  3610676 28.19991 Female      1  Grade 10        1
## 2: Australia  Yes  3611874 28.19991 Female      1  Grade 10        1
## 3: Australia  Yes  3601769 28.19991 Female      1  Grade 10        1
## 4: Australia  Yes  3605996 28.19991 Female      1  Grade 10        1
## 5: Australia  Yes  3608147 33.44862   Male      0  Grade 10        1
## 6: Australia  Yes  3610012 33.44862   Male      0  Grade 10        1
##    software internet ST011Q05TA ST071Q02NA ST071Q01NA     ST123Q02NA
## 1:        1        1        Yes          0          1       Disagree
## 2:        1        1        Yes          1          1          Agree
## 3:        1        1        Yes         NA         NA          Agree
## 4:        1        1        Yes          5          7 Strongly agree
## 5:        1        1        Yes          1          1          Agree
## 6:        1        1        Yes          2          2          Agree
##           ST082Q01NA     ST119Q01NA     ST119Q05NA ANXTEST COOPERATE
## 1:          Disagree          Agree Strongly agree -0.1522    0.2085
## 2:             Agree          Agree       Disagree  0.2594   -0.2882
## 3: Strongly disagree Strongly agree       Disagree  2.5493   -1.2109
## 4: Strongly disagree Strongly agree Strongly agree  0.2563    0.3950
## 5:             Agree          Agree       Disagree  0.4517   -1.3606
## 6:             Agree          Agree          Agree  0.5175    0.4252
##     BELONG EMOSUPS HOMESCH  ENTUSE ICTHOME ICTSCH  WEALTH PARED TMINS
## 1:  0.5073 -2.2547 -0.1686 -0.7369       4      5  0.0592    12  1400
## 2: -0.8021 -0.2511  0.0302 -0.1047       9      6  0.7605    12  1100
## 3: -2.4078 -1.9895  1.2836 -1.5403      11     10 -0.1220    11  1960
## 4: -0.3381  1.0991 -0.0498  0.0342      10      7  0.9314    15  2450
## 5: -0.5050 -1.3298 -0.3355  0.2309      NA      7  0.7905    15  1400
## 6: -0.0099 -0.4263  0.1567  0.6896      10      5  0.7054    15  1400
##       ESCS TEACHSUP TDTEACH IBTEACH SCIEEFF     math  reading  science
## 1:  0.4078       NA      NA      NA      NA 545.8999 586.5175 589.5787
## 2:  0.4500   0.3574  0.0615  0.2208 -0.4041 511.6101 570.8238 557.2042
## 3: -0.5889  -1.0718 -0.6102 -0.2198 -0.9003 478.6052 570.0345 569.4709
## 4:  0.6498   0.6375  0.7979 -0.0282  1.2395 506.0904 531.0690 529.0353
## 5:  0.7675   0.8213  0.1990  1.1477 -0.0746 481.8569 506.4988 504.2148
## 6:  1.1151       NA      NA      NA      NA 455.0202 456.4882 472.6377
##    grade learning    Region
## 1:    10       23 Australia
## 2:    10       18 Australia
## 3:    10       33 Australia
## 4:    10       41 Australia
## 5:    10       23 Australia
## 6:    10       23 Australia
```

We want to see the distributions of the science scores across the 17 countries in our final dataset. The first line with `ggplot()` creates a layout for our figure, the second line draws a box plot using `geom_boxplot()`, the fourth line with `labs()` creates labels of the axes, and the last line with `theme_bw()` removes the default theme with a grey background and activates the dark-on-light `ggplot2` theme – which is much better for publications and presentations (see https://ggplot2.tidyverse.org/reference/ggtheme.html for a complete list of themes available in `ggplot2`).

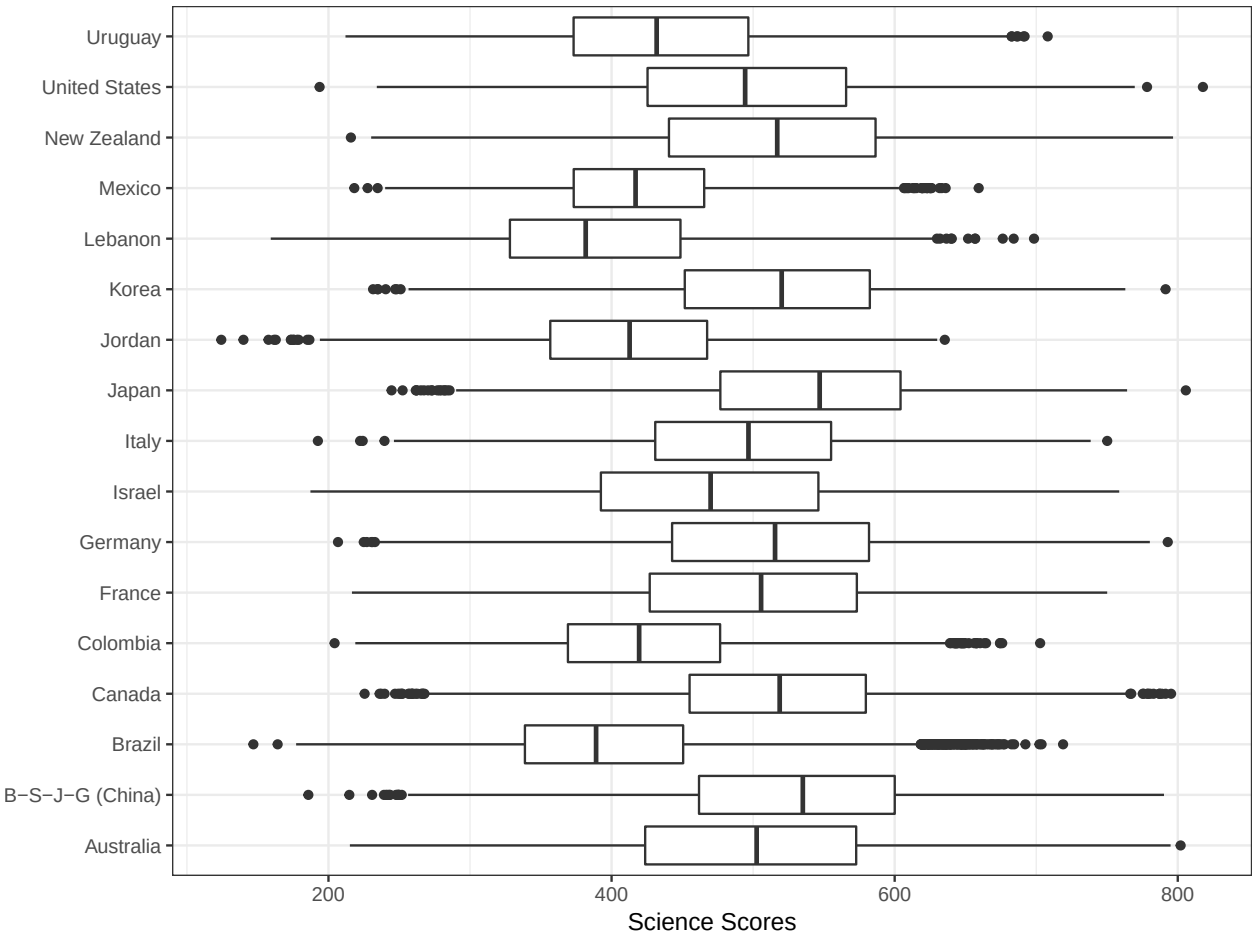
```
ggplot(data = dat, mapping = aes(x = CNT, y = science)) +
  geom_boxplot() +
```

```
  labs(x=NULL, y="Science Scores") +
  theme_bw()
```



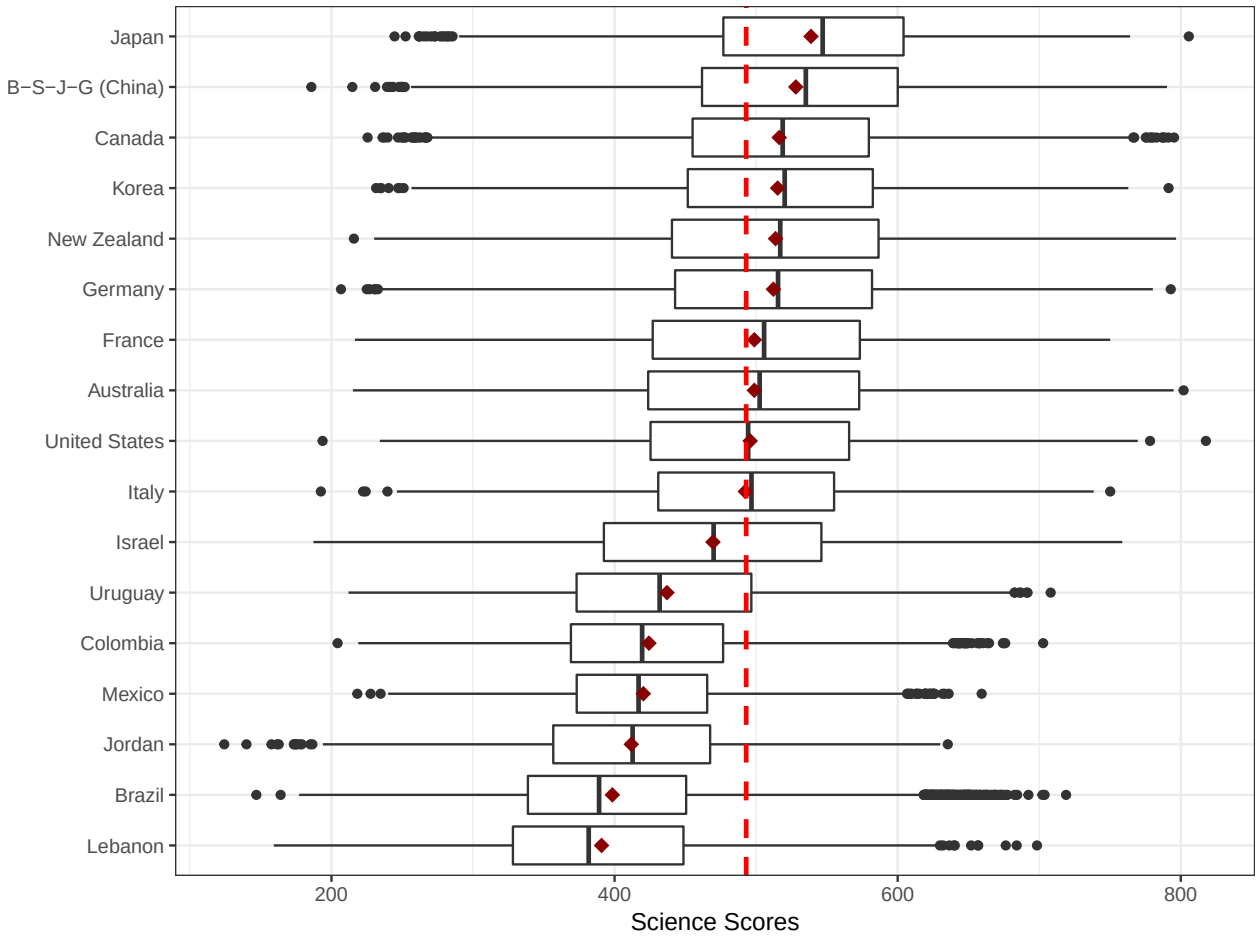
The resulting plot is not necessarily nice because all the country names on the x-axis seem to be squeezed together and thus some of the country names are not visible on the x-axis. To correct this, we may want to flip the coordinates of the plot and use country names on the y-axis instead. The `coord_flip()` function allows us to achieve that very easily.

```
ggplot(data = dat,
       mapping = aes(x = CNT, y = science)) +
  geom_boxplot() +
  labs(x=NULL, y="Science Scores") +
  coord_flip() +
  theme_bw()
```



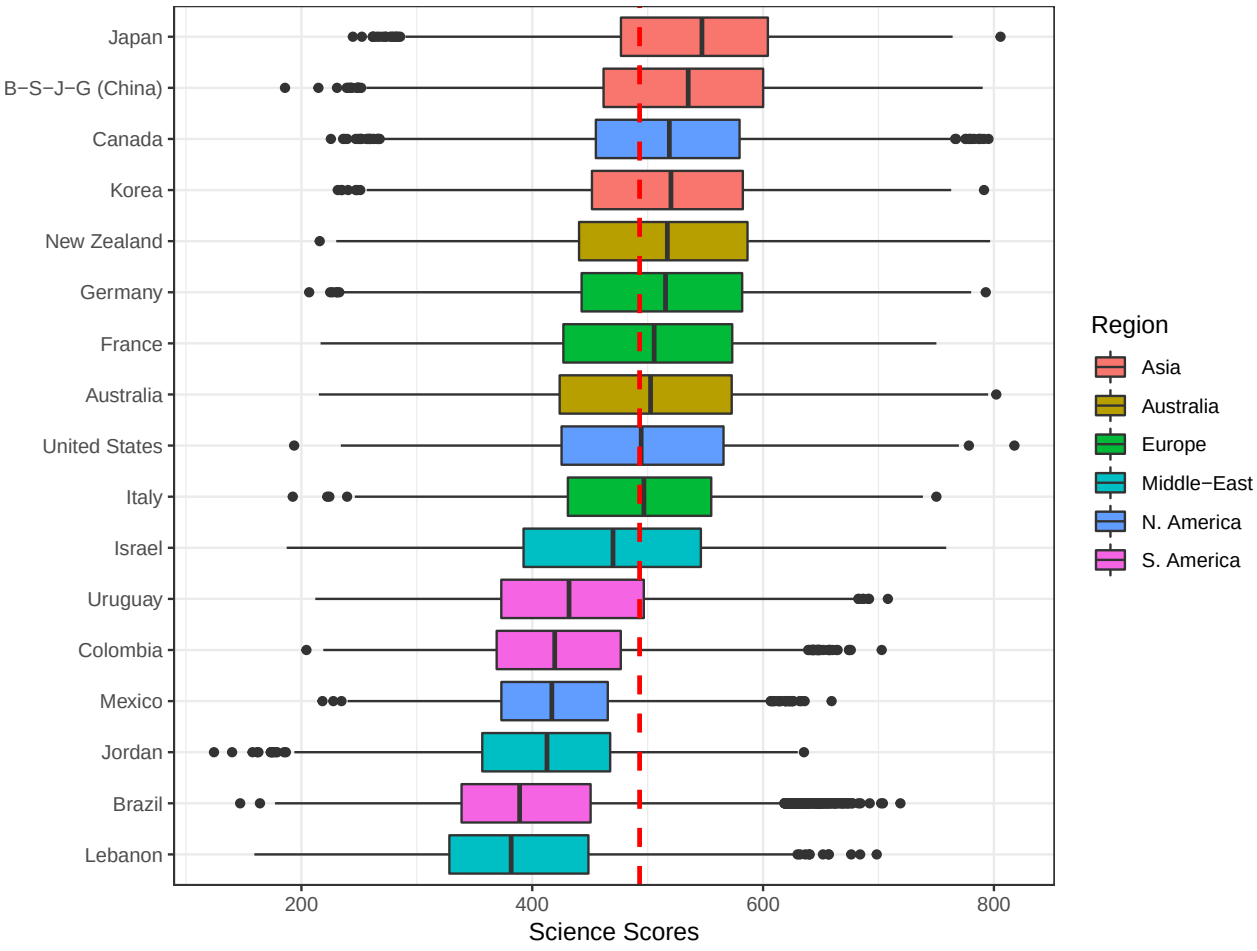
Given that the average science score in PISA 2015 was 493 across all participating countries (see PISA 2015 Results in Focus for more details), we can add a reference line into our plot to identify the average score, which would then allow us to visually examine which countries are above or below the average score. To achieve this, we use `geom_hline()` function and specify where it should intersect the plot (i.e., `yintercept = 493`). We also want the reference line to be a red, dashed-line with a thickness level of 1 – to make it more visible in the plot. Finally, to faciliate the interpretation of the plot, we want the boxplots to be ordered based on the average scores for each country and thus we add `reorder(CNT, science)` into the mapping.

```
means <- aggregate(science ~  CNT, data = dat, mean)

ggplot(data = dat,
       mapping = aes(x = reorder(CNT, science), y = science)) +
  geom_boxplot() +
  stat_summary(fun.y=mean, colour="darkred", geom="point",
               shape=18, size=3,show_guide = FALSE) +
  labs(x=NULL, y="Science Scores") +
  coord_flip() +
  geom_hline(yintercept = 493, linetype="dashed", color = "red", size = 1) +
  theme_bw()
```



Now let's add some colors to our figure based on the region where each country is located. In order to do this, we use the region variable to fill the boxplots with color, using `fill = Region`.

```
ggplot(data = dat,
       mapping = aes(x = reorder(CNT, science), y = science, fill = Region)) +
  geom_boxplot() +
  labs(x=NULL, y="Science Scores") +
  coord_flip() +
  geom_hline(yintercept = 493, linetype="dashed", color = "red", size = 1) +
  theme_bw()
```



3.2.1 Exercise

Create a plot of math scores over countries with different colors based on region. You need to modify the R code below by replacing `geom_boxplot()` with:

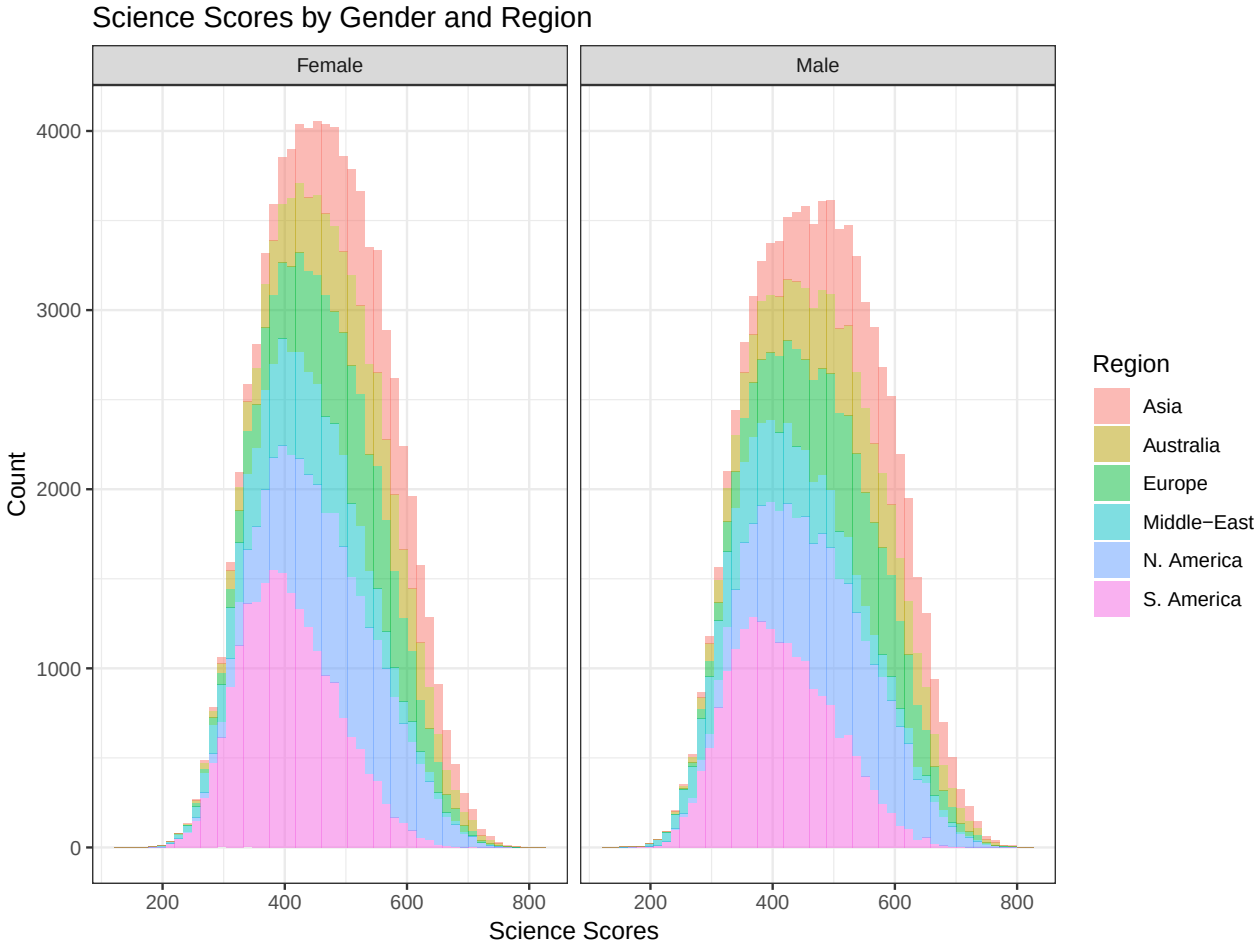
- `geom_point(aes(color = Region))`, and then
- `geom_violin(aes(color = Region))`.

How long did it take to create both plots? Which one is a better way to visualize this type of data?

```
ggplot(data = dat,
       mapping = aes(x = reorder(CNT, science), y = science, fill = Region)) +
  geom_boxplot() +
  labs(x=NULL, y="Science Scores") +
  coord_flip() +
  geom_hline(yintercept = 493, linetype="dashed", color = "red", size = 1) +
  theme_bw()
```

We can also create histograms (or density plots) for a particular variable and split the plot into multiple plots by using a categorial, group variable. In the following example, we use `x = Region` in the mapping in order to identify different regions in the distribution of the science scores. In addition, we use `facet_grid(. ~ ST004D01T)` to generate separate histograms by gender.

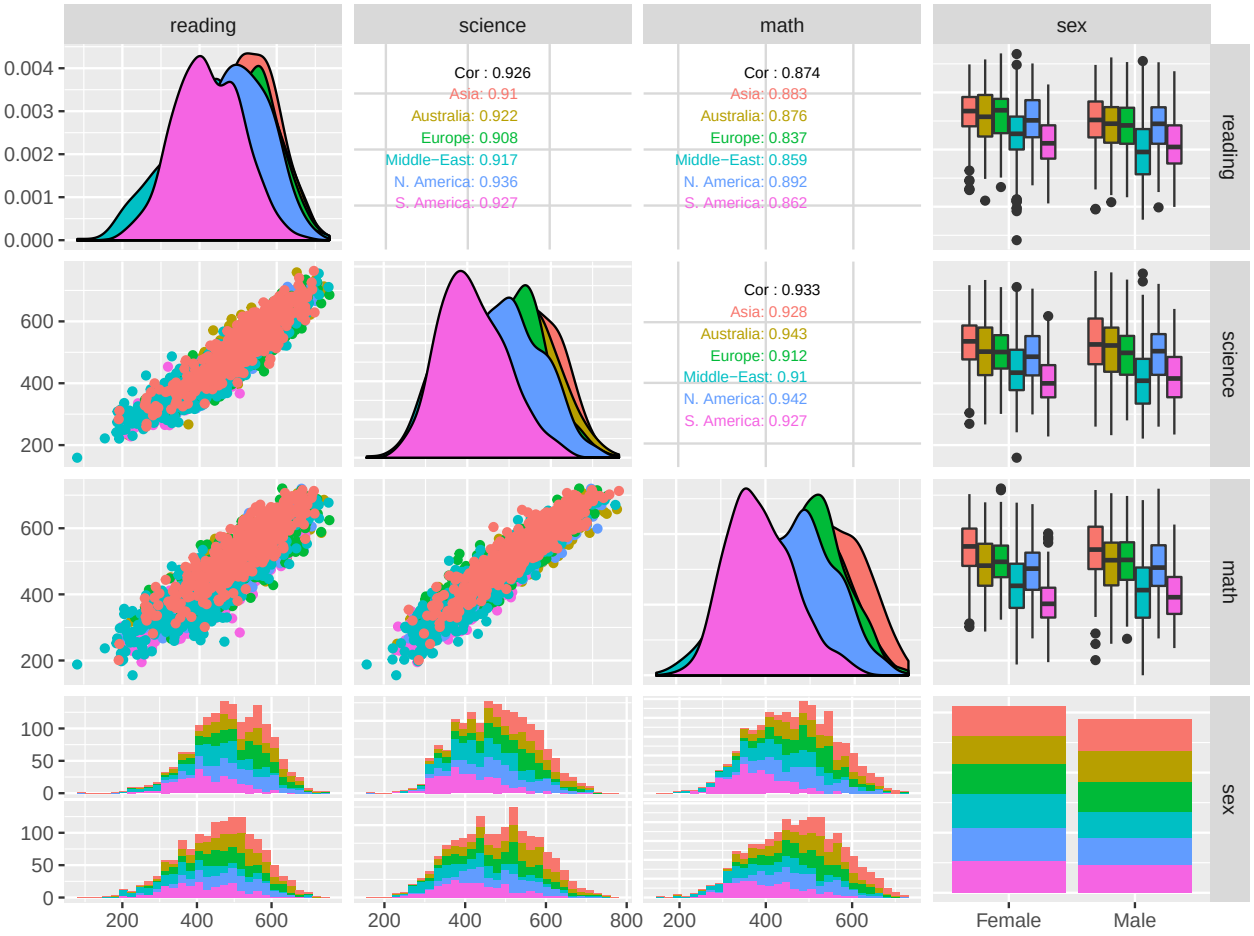
```
ggplot(data = dat,
       mapping = aes(x = science, fill = Region)) +
  geom_histogram(alpha = 0.5, bins = 50) +
  labs(x = "Science Scores", y = "Count",
       title = "Science Scores by Gender and Region") +
  facet_grid(. ~ sex) +
  theme_bw()
```



If we are interested in visualizing multiple variables, plotting each variable individually can be time consuming. Therefore, we an use the `ggpairs()` function from the `GGally` package to build a more complex, diagnostic plot for multiple variables. In the following example, we plot reading, science, and math scores as well as gender (i.e., ST004D01T) in the same plot. Because our dataset is quite large, plotting all the data points would result in a highly complex plot where most data points overlap. Therefore, we will take a random sample of 500 cases from each region defined in the data, save this smallar dataset as `dat_small`, and use this dataset inside the `ggpairs` function. We colorize each variable by region (using `mapping = aes(color = Region)`). The resulting plot shows density plots for the continuous variables (by region), a stacked bar chart for gender, and box plots for the continous variables by region and gender.

```
# Random sample of 500 students from each region
dat_small <- dat[,.SD[sample(.N, min(500,.N))], by = Region]
#
ggpairs(data = dat_small,
        mapping = aes(color = Region),
        columns = c("reading", "science", "math", "sex"),
```

```
        upper = list(continuous = wrap("cor", size = 2.5))
        )
```



3.3 Conditional plots

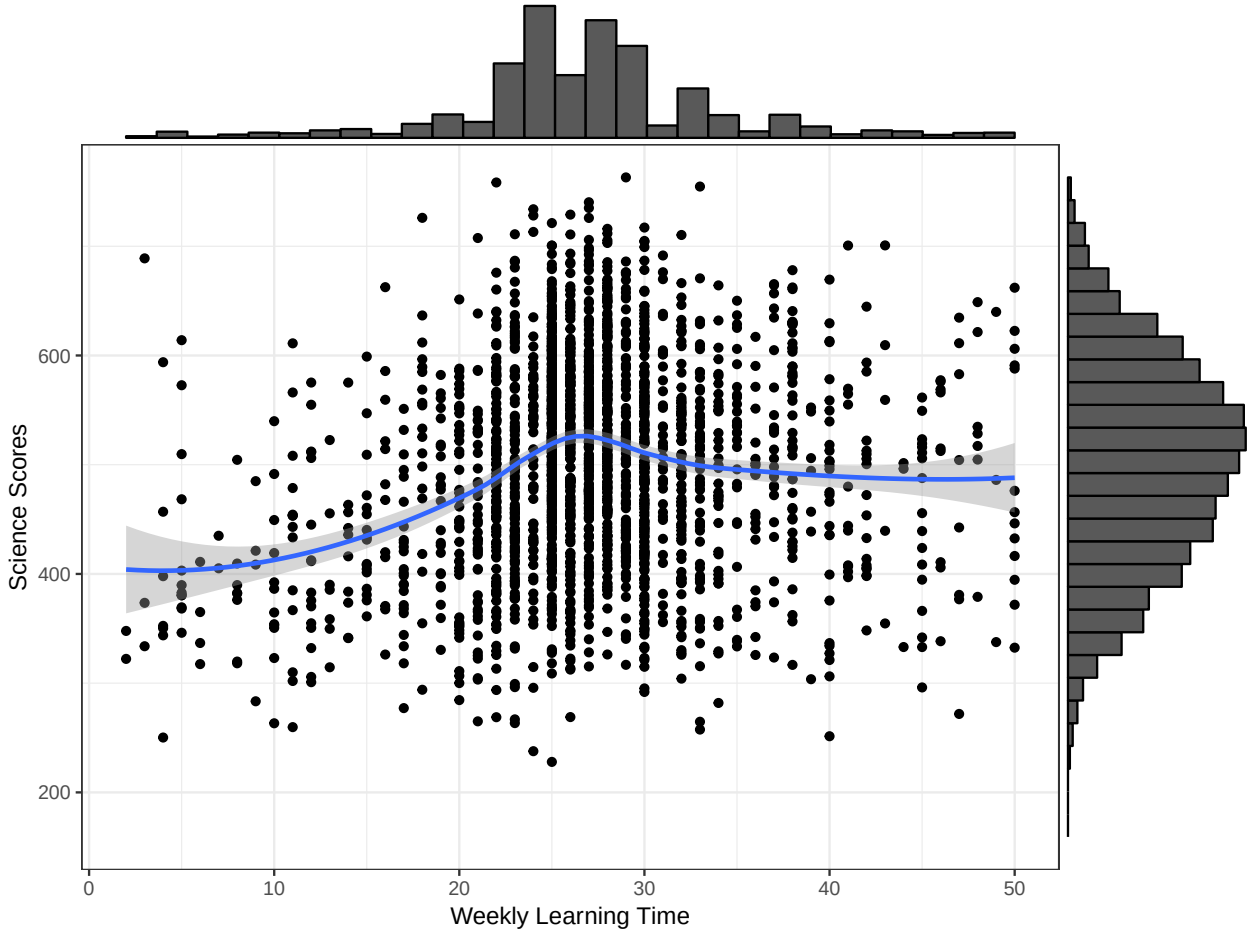
When we deal with continuous variables, an effective way to understand the relationship between the variables is to produce conditional plots, such as scatterplots, dotplots, and bubble charts.

Simple scatterplots in R can be created using `plot(var1, var2, data = name_of_dataset)`. Using the extended capabilities of `ggplot2` via the `ggExtra` package, we can combine histograms and density plots with scatterplots and visualize them together.

In the following example, we first create a scatterplot of learning time per week and science scores using `ggplot()`. We use `geom_point()` to draw a plot with points and `geom_smooth(method = "loess")` to add a regression line with loess smoothing (i.e., Locally Estimated Scatterplot Smoothing). We save this result as `p1` and then pass it to `ggMarginal()` to transform the plot into a marginal scatterplot. Inside `ggMarginal()`, we use `type = "histogram"` to create histograms for learning time per week and science scores on the x and y axes of the plot. Note that as the plot is created, you may see some warning messages, such as "Removed 750 rows containing missing values", because the variables have some missing rows in the dataset.

```
p1 <- ggplot(data = dat_small,
             mapping = aes(x = learning, y = science)) +
```

```
  geom_point() +
  geom_smooth(method = "loess") +
  labs(x = "Weekly Learning Time", y = "Science Scores") +
  theme_bw() +
  theme(legend.position = "bottom",
        legend.title = element_blank())

# Replace "histogram" with "boxplot" or "density" for other types
ggMarginal(p1, type = "histogram")
```



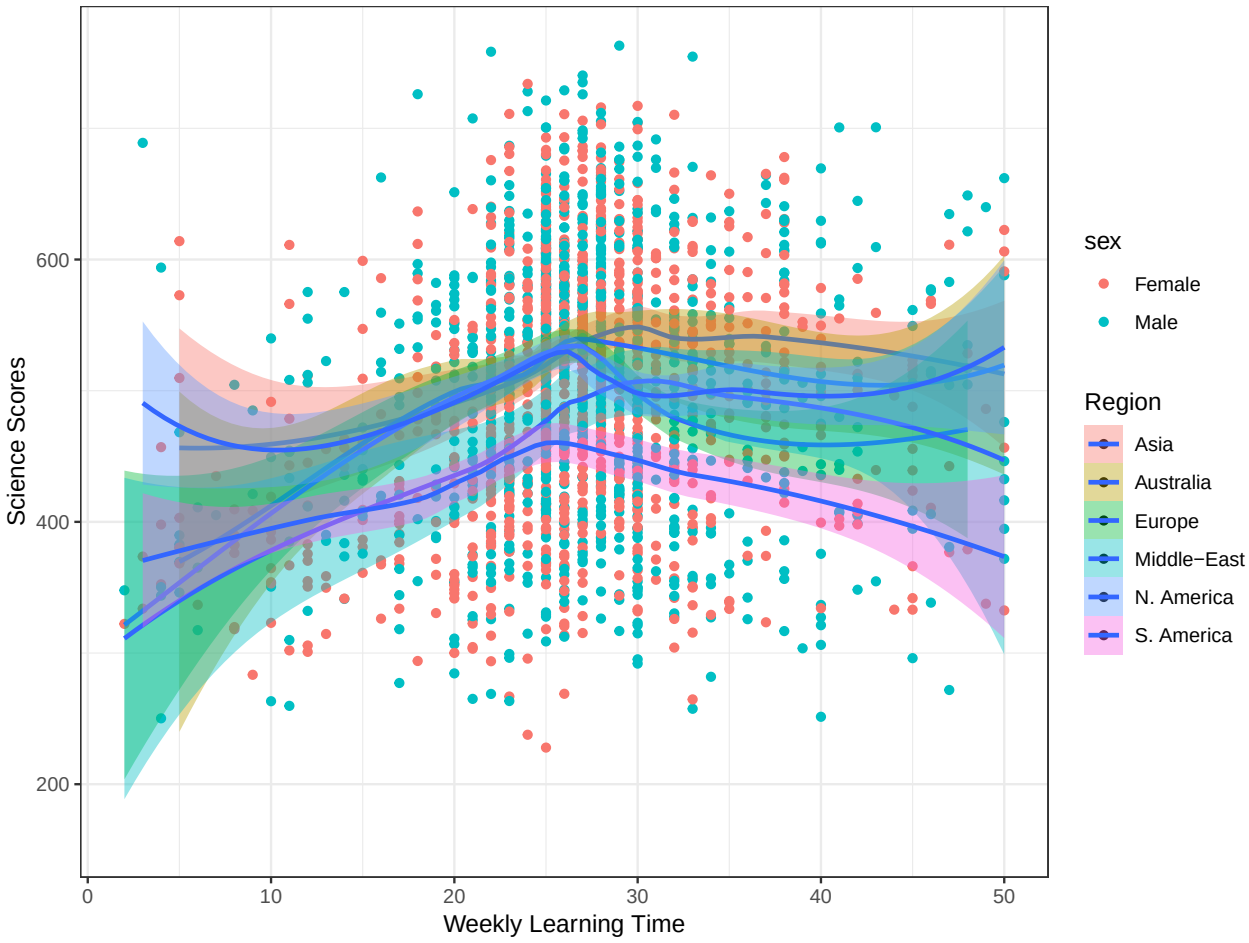
We can also distinguish male and female students in the plot and create a scatterplot of learning time and science scores with densities by gender. To achieve this, we add `colour = sex` into the mapping of `ggplot` and change the type of plot to `type = "density"` in `ggMarginal`. In addition, we use `groupColour = TRUE, groupFill = TRUE` inside `ggMarginal()` to use separate colors for each gender in the density plots.

```
p2 <- ggplot(data = dat_small,
             mapping = aes(x = learning, y = science,
                           colour = sex)) +
  geom_point() +
  geom_smooth(method = "loess") +
  labs(x = "Weekly Learning Time", y = "Science Scores") +
  theme_bw() +
  theme(legend.position = "bottom",
        legend.title = element_blank())
```

```
ggMarginal(p2, type = "density", groupColour = TRUE, groupFill = TRUE)
```

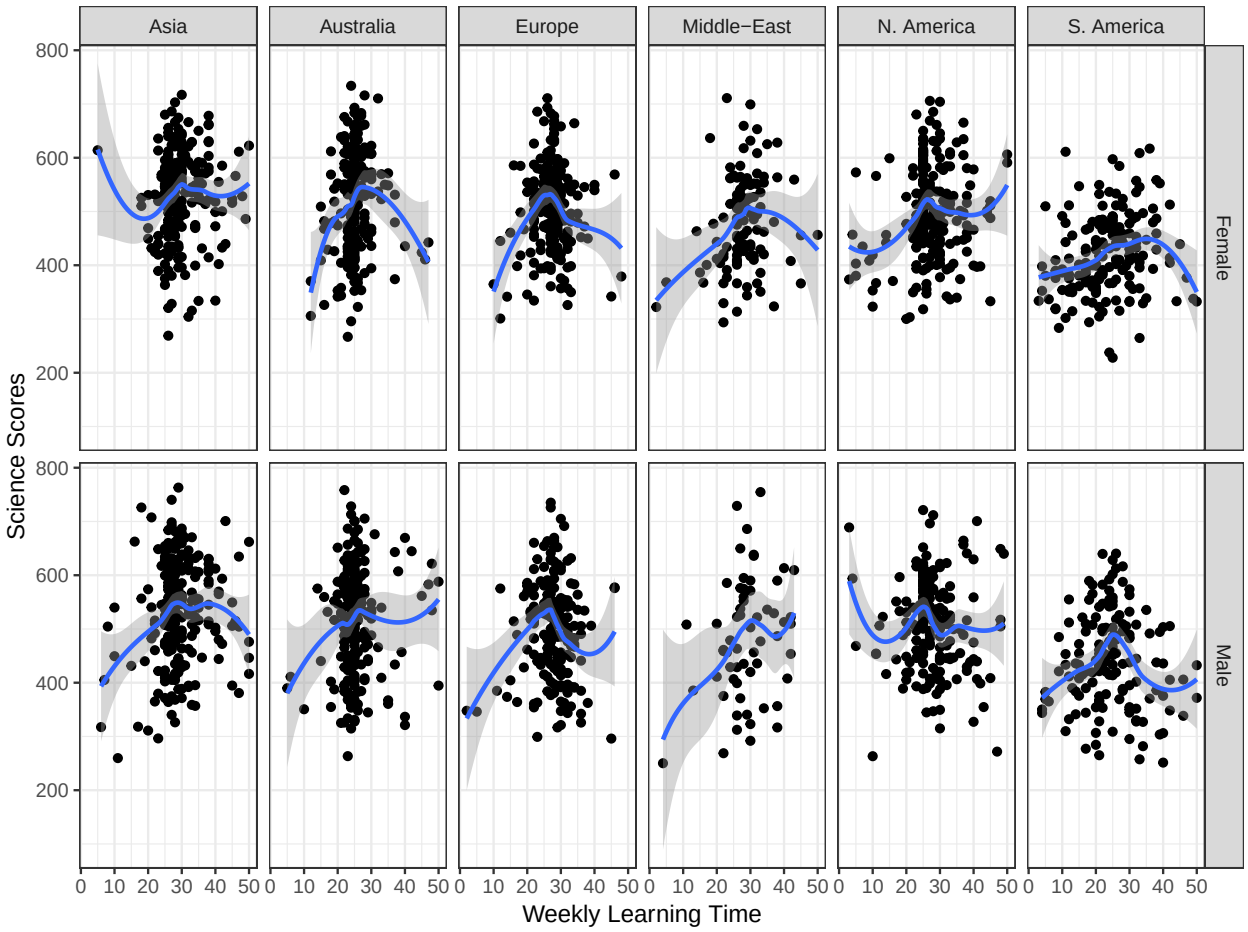
Now let's incorporate more variables into the plot. This time we are not going to use marginal plots. Instead, we will create a regular scatterplot but add other layers to represent additional variables. In the following example, we examine the relationship between students' weekly learning time (learning) and science scores (science) across regions (region) and gender (ST004D01T). Adding `fill = Region` into the mapping will allow us to draw regression lines by regions, while adding `aes(colour = sex)` into `geom_point()` will allow us to use different colors for male and female students in the plot.

```
ggplot(data = dat_small,
       mapping = aes(x = learning, y = science, fill = Region)) +
  geom_point(aes(colour = sex)) +
  geom_smooth(method = "loess") +
  labs(x = "Weekly Learning Time", y = "Science Scores") +
  theme_bw()
```



The resulting scatterplot is nice but it is hard to compare the results clearly between gender groups and regions. To improve the interpretability of the plot, we will use the faceting option. This will allow us to split the scatterplot into multiple plots based on gender and region. In the following example, we examine the relationship between students' learning time and science scores across regions and gender. We use `facet_grid(sex ~ Region)` to split the plots into multiple rows based on gender and multiple columns based on region.

```
ggplot(data = dat_small,
       mapping = aes(x = learning, y = science)) +
  geom_point() +
  geom_smooth(method = "loess") +
  labs(x = "Weekly Learning Time", y = "Science Scores") +
  theme_bw() +
  theme(legend.title = element_blank()) +
  facet_grid(sex ~ Region)
```

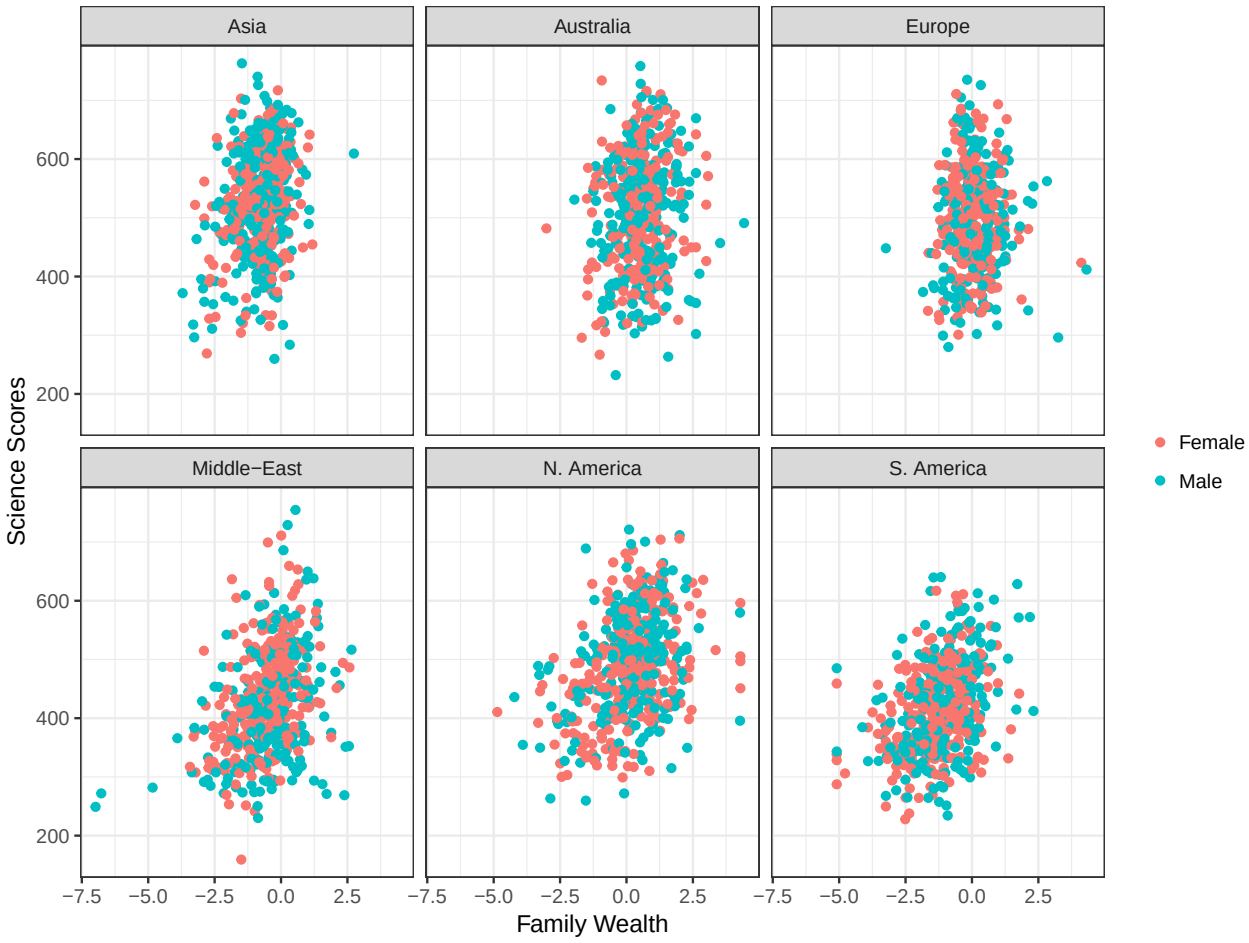


3.3.1 Exercise

Create a scatterplot of socio-economic status (ESCS) and math scores (math) acorss regions (region) and gender (sex). Use `geom_smooth(method = "lm")` to draw linear regression lines (instead of loess smoothing). Do you think that the relationship between ESCS and math changes across gender groups and regions?

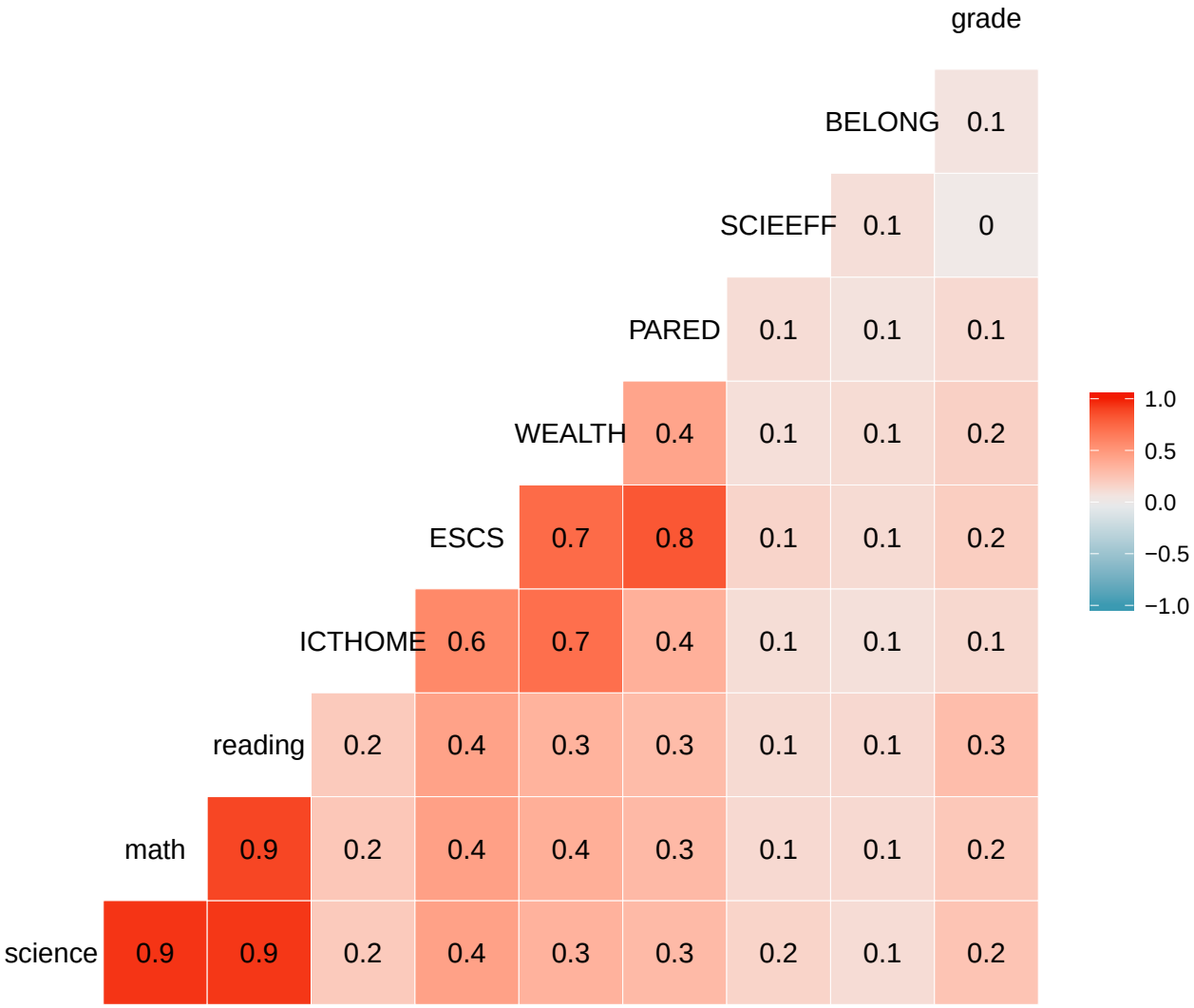
For a simpler examination of variables of two continuous variables, we could just create a scatterplot matrix. In the following plot, we will create a scatterplot matrix of family wealth (WEALTH) and science scores (science) by gender (sex) and region (region). We will use region for facetting and gender for coloring the data points.

```
ggplot(data = dat_small,
       mapping = aes(x = WEALTH, y = science)) +
  geom_point(aes(color = sex)) +
  facet_wrap( ~ Region) +
  labs(x = "Family Wealth", y = "Science Scores") +
  theme_bw() +
  theme(legend.title = element_blank())
```



A more effective way for identifying related variables in the dataset for further statistical analyses (also known as feature extraction) is to create a correlation matrix plot. The `ggcorr()` function from the `GGally` package provides a simple way to make a correlation matrix plot. In the following example, we will create a correlation matrix plot for science, math, reading, emotional support, test anxiety, SES, family wealth, highest parental education, and grade level.

```
ggcorr(data = dat[,.(science, math, reading, ICTHOME, ESCS,
                     WEALTH, PARED, SCIEEFF, BELONG, grade)],
       method = c("pairwise.complete.obs", "pearson"),
       label = TRUE, label_size = 4)
```



Let's see the relationship between gender and science scores across all countries. First, we need to find the average science scores by country and gender.

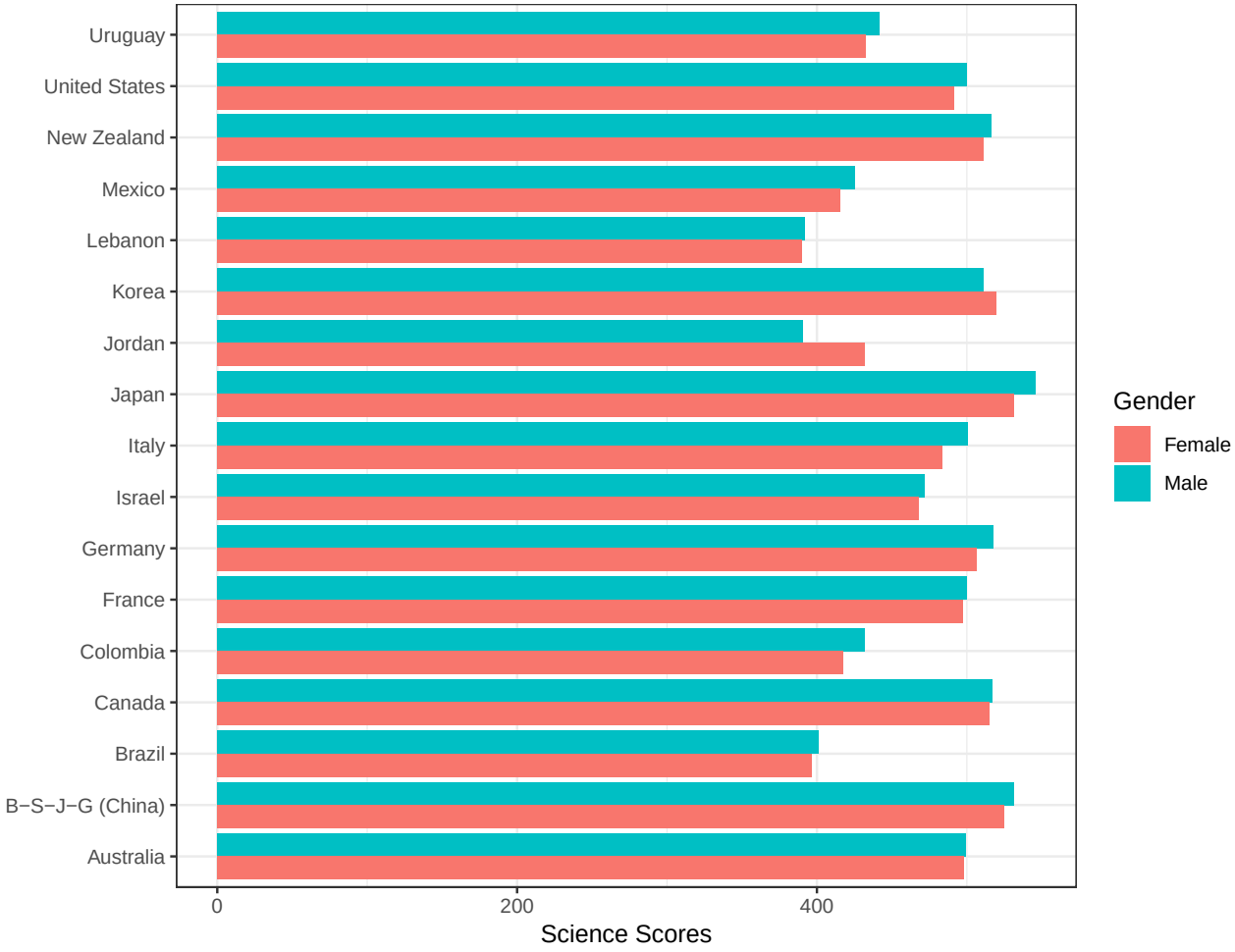
```
science_summary <- dat[, .(Science = mean(science, na.rm = TRUE), Freq = .N),
                by = c("sex", "CNT")]

head(science_summary)
```

```
##       sex       CNT  Science  Freq
## 1: Female Australia 498.0377  7163
## 2:   Male Australia 499.4419  7367
## 3:   Male    Brazil 400.8134 11068
## 4: Female    Brazil 396.2647 12073
## 5: Female    Canada 515.3443 10022
## 6:   Male    Canada 517.2765 10036
```

Next, we can create a bar graph summarizing the average science performance by gender and country.

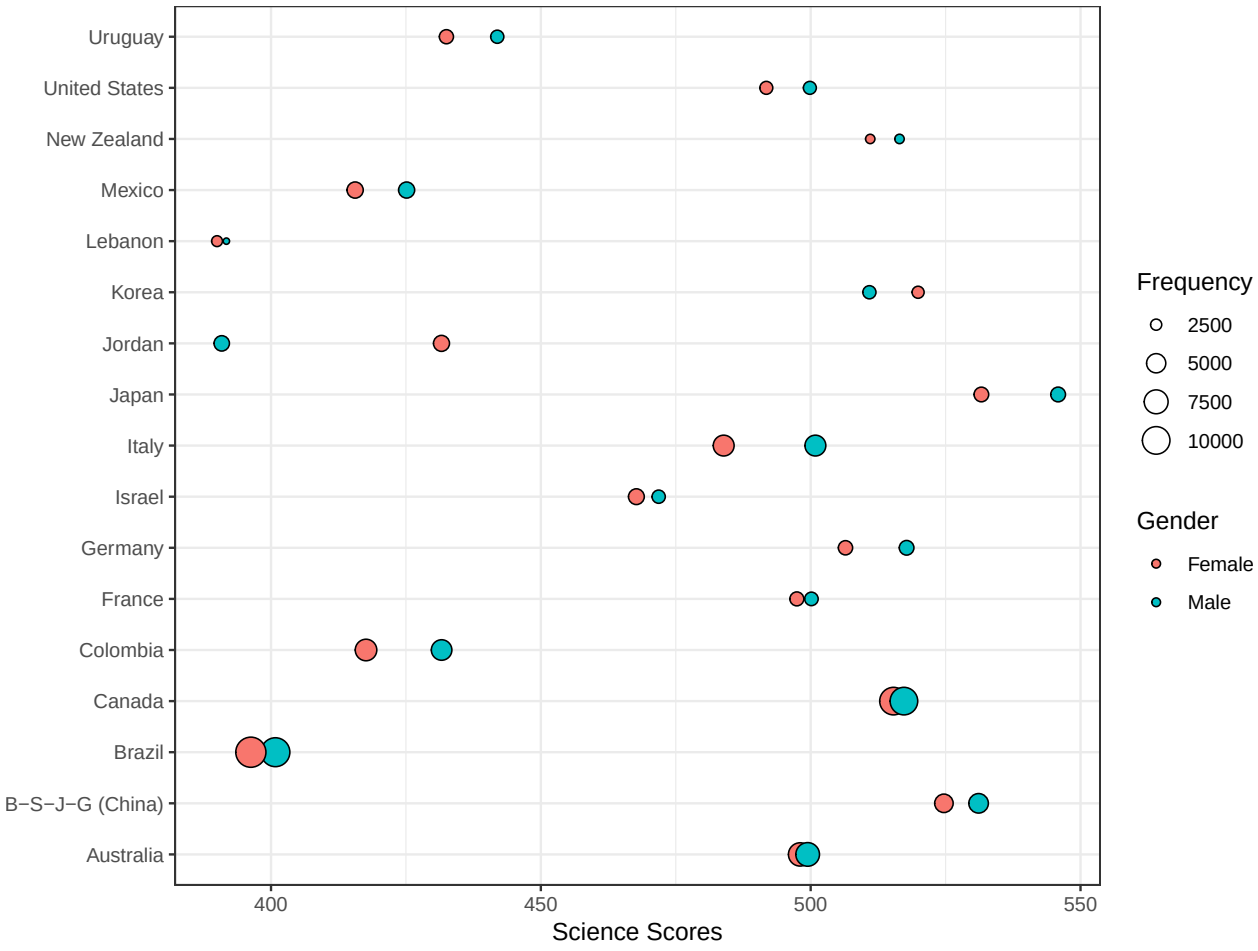
```
ggplot(data = science_summary,
       mapping = aes(x = CNT, y = Science, fill = sex)) +
  geom_bar(stat = "identity", position = "dodge") +
  coord_flip() +
```

```
  labs(x = "", y = "Science Scores", fill = "Gender") +
  theme_bw()
```



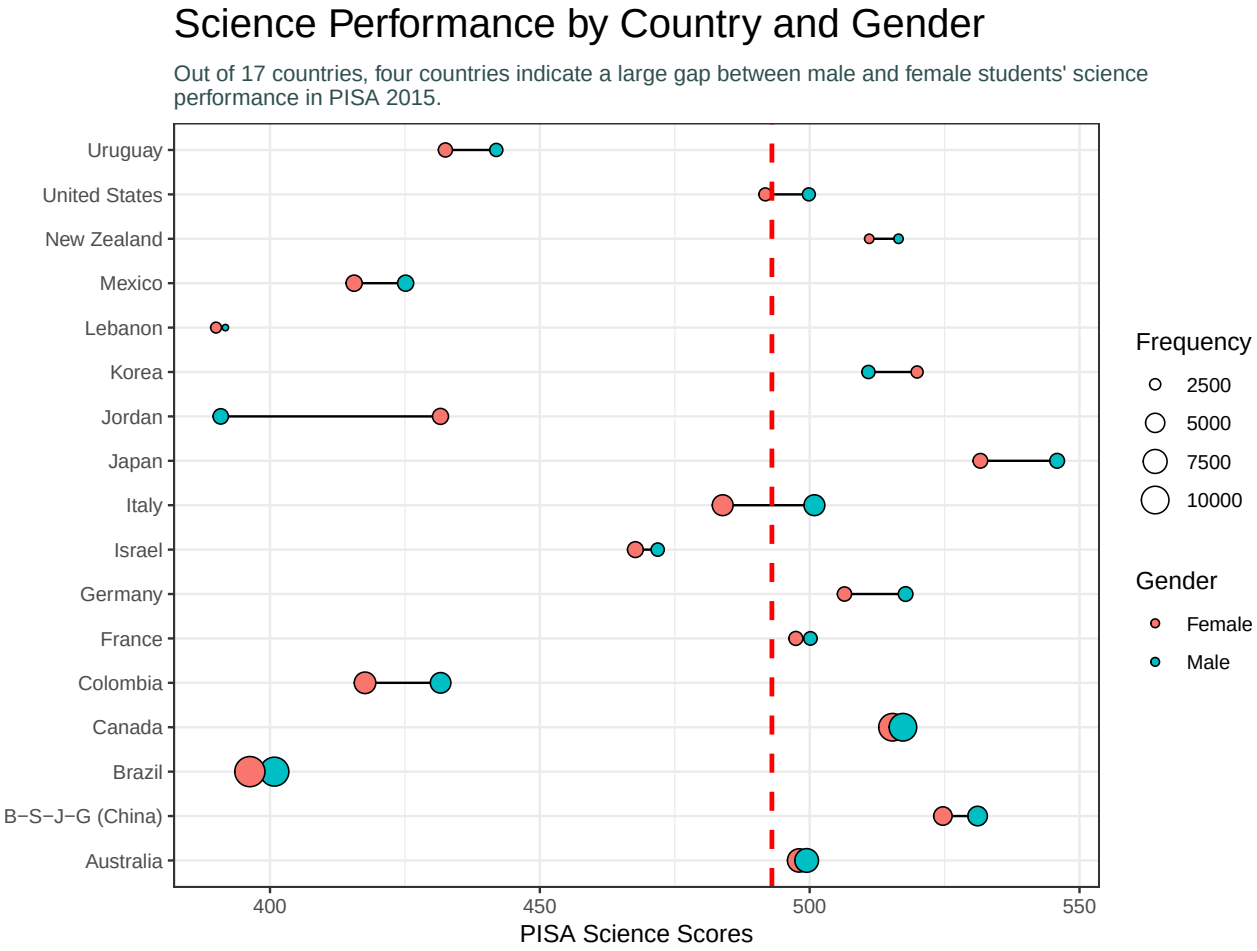
Despite their easiness and simplicity, bar graphs are not necessarily visually appealing. Let's create a bubble chart to visualize the same information in a different way. A bubble chart is essentially a weighted scatterplot where a third variable determines the size of the dots in the plot. In the following bubble chart, we use *Freq* (i.e., number of students from each country) to determine the size of the dots in the plot, using `size = Freq`.

```
ggplot(data = science_summary,
       mapping = aes(x = CNT, y = Science, size = Freq, fill = sex)) +
  geom_point(shape = 21) +
  coord_flip() +
  theme_bw() +
  labs(x = NULL, y = "Science Scores", fill = "Gender",
       size = "Frequency")
```



We can also create a dot plot, which is very similar to the buble chart when one of the variables is categorical, to convey the same information even more effectively. As you will see, this is a more polished version of the buble chart with additional titles and subtitles.

```
ggplot(data = science_summary, mapping = aes(x = CNT, y = Science, fill = sex)) +
    geom_line(aes(group = CNT)) + geom_point(aes(size = Freq), shape = 21) +
    geom_hline(yintercept = 493, linetype = "dashed", color = "red", size = 1) +
    labs(x = NULL, y = "PISA Science Scores", fill = "Gender", size = "Frequency",
        title = "Science Performance by Country and Gender", subtitle = "Out of 17 countries, four coun
    coord_flip() + theme_bw() + theme(plot.title = element_text(size = 18, margin = margin(b = 10)),
    plot.subtitle = element_text(size = 10, color = "darkslategrey"))
```



A slightly more complex plot, which is also known as *alluvial plot*, can be used to summarize relationships between multiple categorical variables. In the following example, we use region (Region), gender (ST004D01T), and a survey item regarding whether parents support educational efforts and achievements (ST123Q02NA). We create a new dataset called `dat_alluvial` to have frequency counts by region, gender, and our survey item. Because the survey item includes missing values, we label them as "missing" and then recode this variable as a factor with reordered levels.

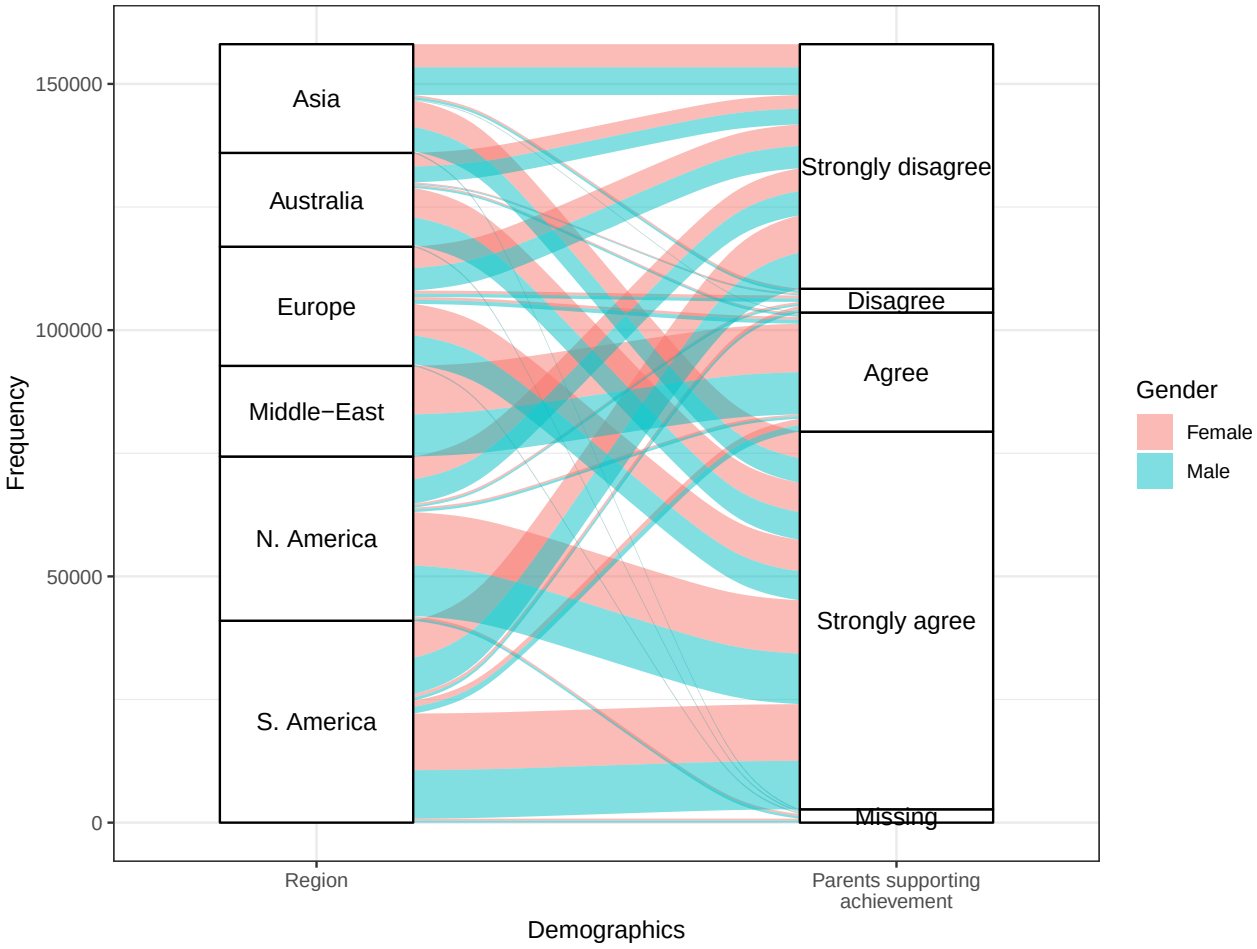
```
dat_alluvial <- dat[, .(Freq = .N), by = c("Region", "sex", "ST123Q02NA")][
  , ST123Q02NA := as.factor(ifelse(ST123Q02NA == "", "Missing", ST123Q02NA))]

levels(dat_alluvial$ST123Q02NA) <- c("Strongly disagree", "Disagree", "Agree",
                                     "Strongly agree", "Missing")

head(dat_alluvial)
```

```
##       Region    sex        ST123Q02NA Freq
## 1: Australia Female          Disagree  232
## 2: Australia Female Strongly disagree 2773
## 3: Australia Female    Strongly agree 5981
## 4: Australia   Male Strongly disagree 3209
## 5: Australia   Male    Strongly agree 5626
## 6: Australia   Male           Missing  186
```

Unlike the previous visualizations, there is a new layer called `geom_alluvium()`, which allows creating an alluvial plot using the `ggplot()` function. We use `aes(fill = ST004D01T)` inside `geom_alluvium()` to

differentiate the frequencies by gender.

```
# StatStratum <- StatStratum
ggplot(data = dat_alluvial,
       aes(axis1 = Region, axis2 = ST123Q02NA, y = Freq)) +
  scale_x_discrete(limits = c("Region", "Parents supporting\nachievement"),
                   expand = c(.1, .05)) +
  geom_alluvium(aes(fill = sex)) +
  geom_stratum() +
  geom_text(stat = "stratum", label.strata = TRUE) +
  labs(x = "Demographics", y = "Frequency", fill = "Gender") +
  theme_bw()
```



3.3.2 Exercise

Create an alluvial plot for the survey item (ST119Q01NA) of whether students want top grades in most or all courses by region (Region) and gender (ST004D01T). Below we create the summary dataset (dat_alluvial2) for this plot. Use this dataset to draw the alluvial plot plot. How should we interpret the plot (e.g., for each region)?

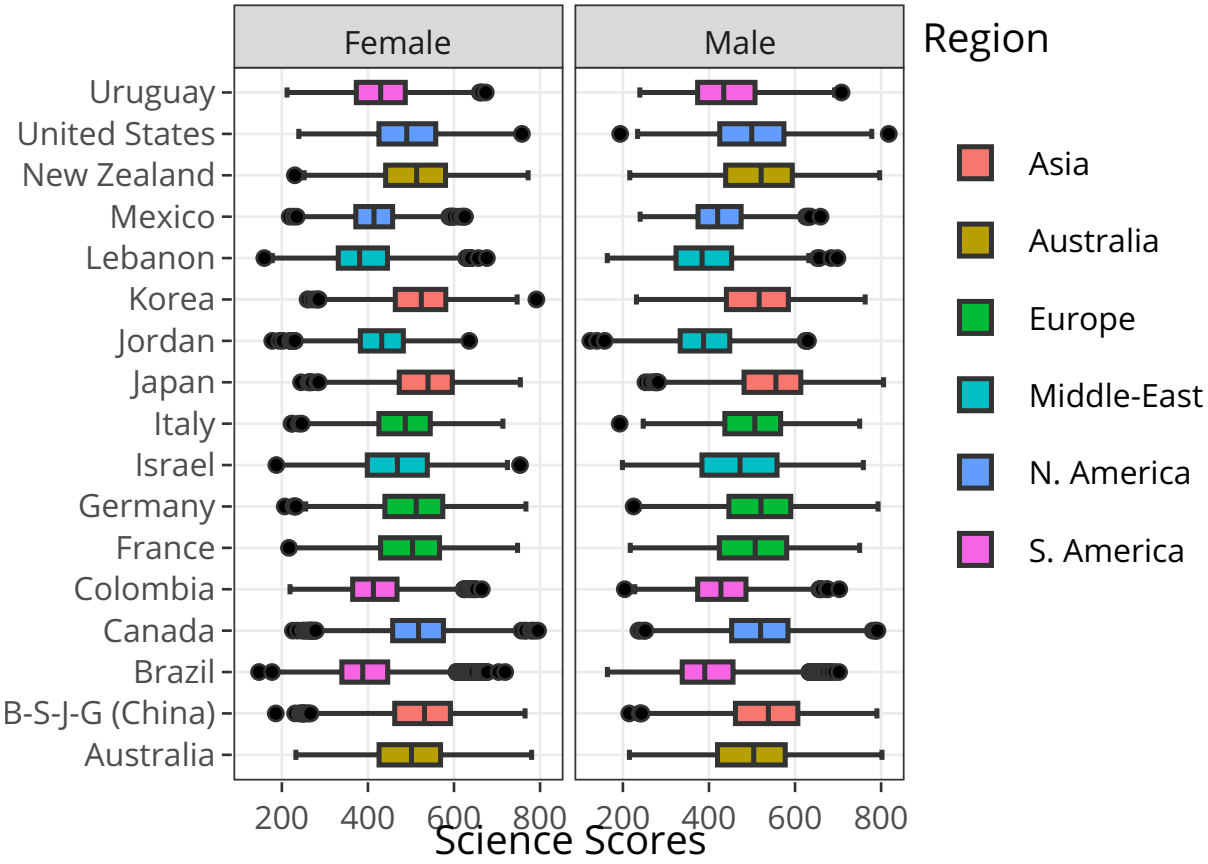
```
dat_alluvial2 <- dat[, .(Freq = .N), by = c("Region", "sex", "ST119Q01NA")][
  , ST119Q01NA := as.factor(ifelse(ST119Q01NA == "", "Missing", ST119Q01NA))]

levels(dat_alluvial2$ST119Q01NA) <- c("Strongly disagree", "Disagree", "Agree",
```

```
                                    "Strongly agree", "Missing")
```

3.4 Interactive plots with plotly

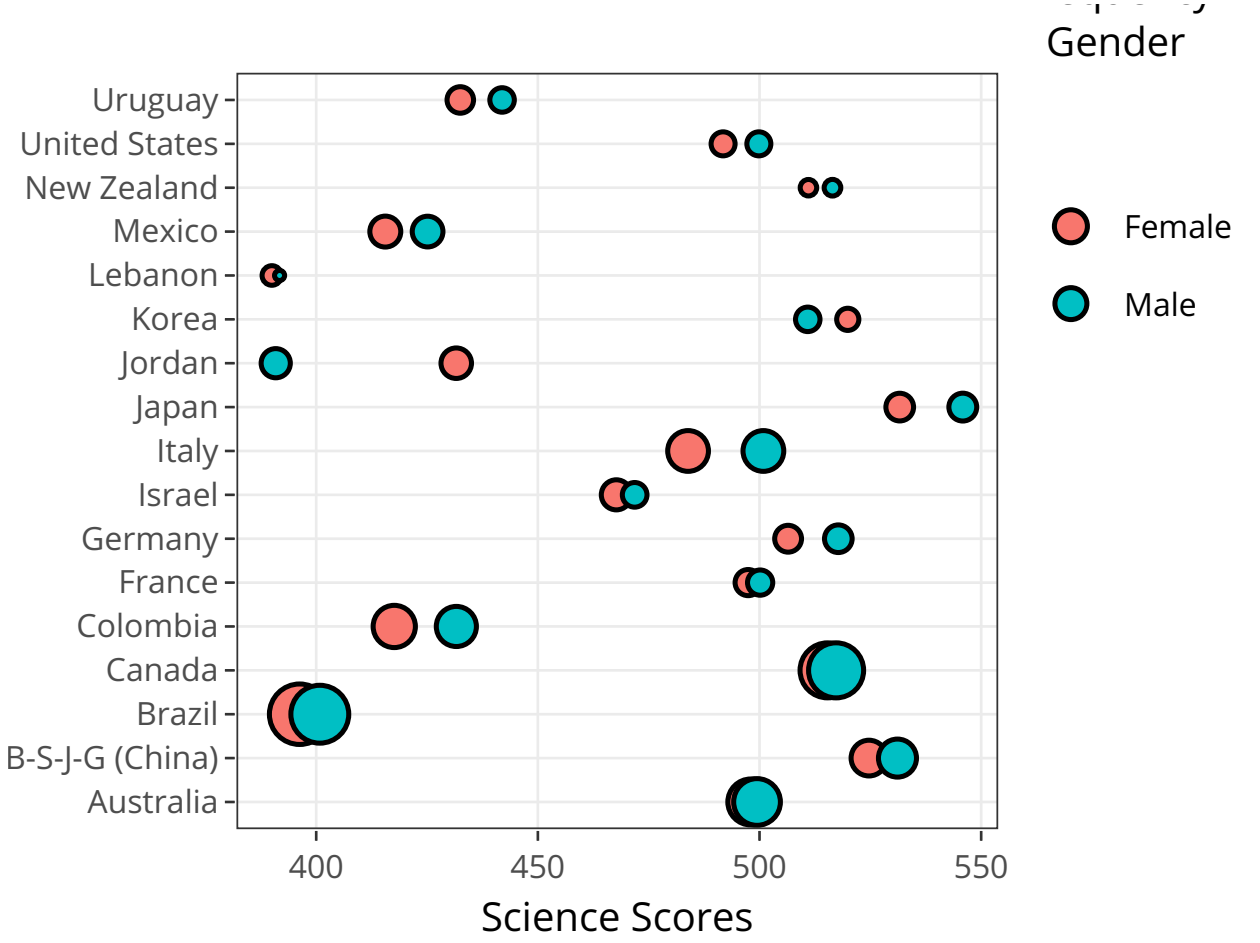
Using the plotly package, we can make more interactive visualizations. The ggplotly function from the plotly package transforms a ggplot2 plot into an interactive plot in the HTML format. In the following example, we first save the plot as p3 and then insert this plot into the plotly function, in order to generate an interactive plot. As we hover the pointer over the plot area, the plot shows the min, max, q1, q3, and median values.

```
p3 <- ggplot(data = dat,
             mapping = aes(x = CNT, y = science, fill = Region))+
  geom_boxplot() +
  facet_grid(. ~ sex)+
  labs(x = NULL, y = "Science Scores", fill = "Region") +
  coord_flip() +
  theme_bw()

ggplotly(p3)
```



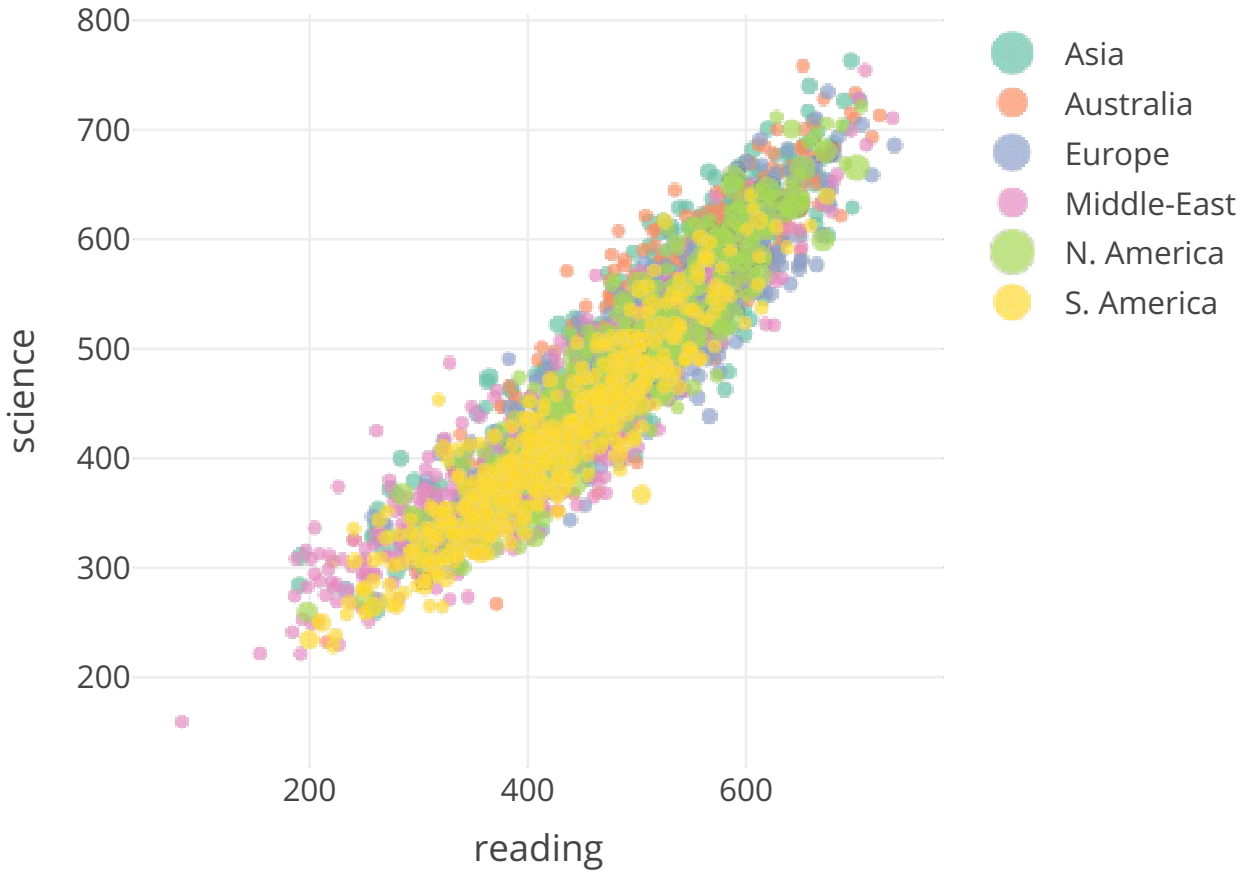

```
p4 <- ggplot(data = science_summary,
       mapping = aes(x = CNT, y = Science, size = Freq, fill = sex)) +
  geom_point(shape = 21) +
```

```
  coord_flip() +
  theme_bw() +
  labs(x = NULL, y = "Science Scores", fill = "Gender",
       size = "Frequency")

ggplotly(p4)
```



We can also use the `plot_ly` function to create interactive visualizations, without using 'ggplot2. In the following example, we create a scatterplot of reading scores and science scores where the color of the dots will be based on region and the size of the dots will be based on student weight in the PISA database. Because the resulting figure is interactive, we can click on the legend and hide some regions as we review the plot. In addition, we add a hover text (`text = ~paste("Reading: ", reading, '<br>Science:', science)`) into the plot. As we hover on the plot, it will show us a label with reading and science scores.

```
plot_ly(data = dat_small,
        x = ~reading, y = ~science, color = ~Region,
        size = ~W_FSTUWT,
        type = "scatter",
        text = ~paste("Reading: ", reading, '<br>Science:', science))
```



3.4.1 Exercise

Replicate the science-by-region histogram below as a density plot and use`plotly` to make it interactive. You will need to replace `geom_histogram(alpha = 0.5, bins = 50)` with `geom_density(alpha = 0.5)`. Repeat the same process by changing `alpha = 0.5` to `alpha = 0.8`. Which version is better for examining the science score distribution?

```
ggplot(data = dat,
       mapping = aes(x = science, fill = Region)) +
  geom_histogram(alpha = 0.5, bins = 50) +
  labs(x = "Science Scores", y = "Count",
       title = "Science Scores by Gender and Region") +
  facet_grid(. ~ ST004D01T) +
  theme_bw()
```

3.5 Customizing visualizations

- cowplot
- ggsave

3.6 Web-based visualizations and dashboards

...

3.7 Lab

...