STA 235H - Bootcamp

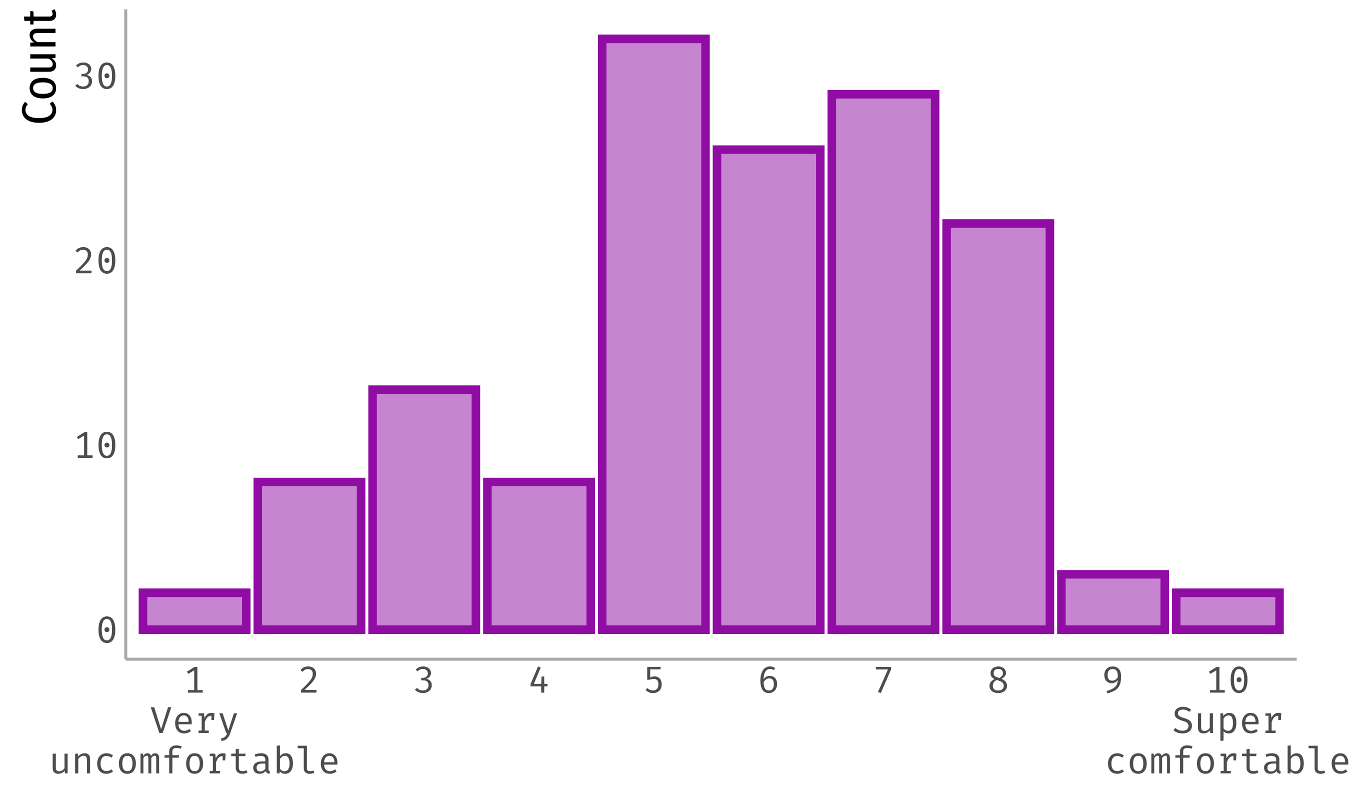
Fall 2023

McCombs School of Business, UT Austin

Bootcamp Agenda

- What do we need?
  - Quick look into R and RStudio
  - RScript format
- Refresher from the tidyverse:
  - Data wrangling
  - Plots and figures
  - Regressions

How comfortable are you with R?

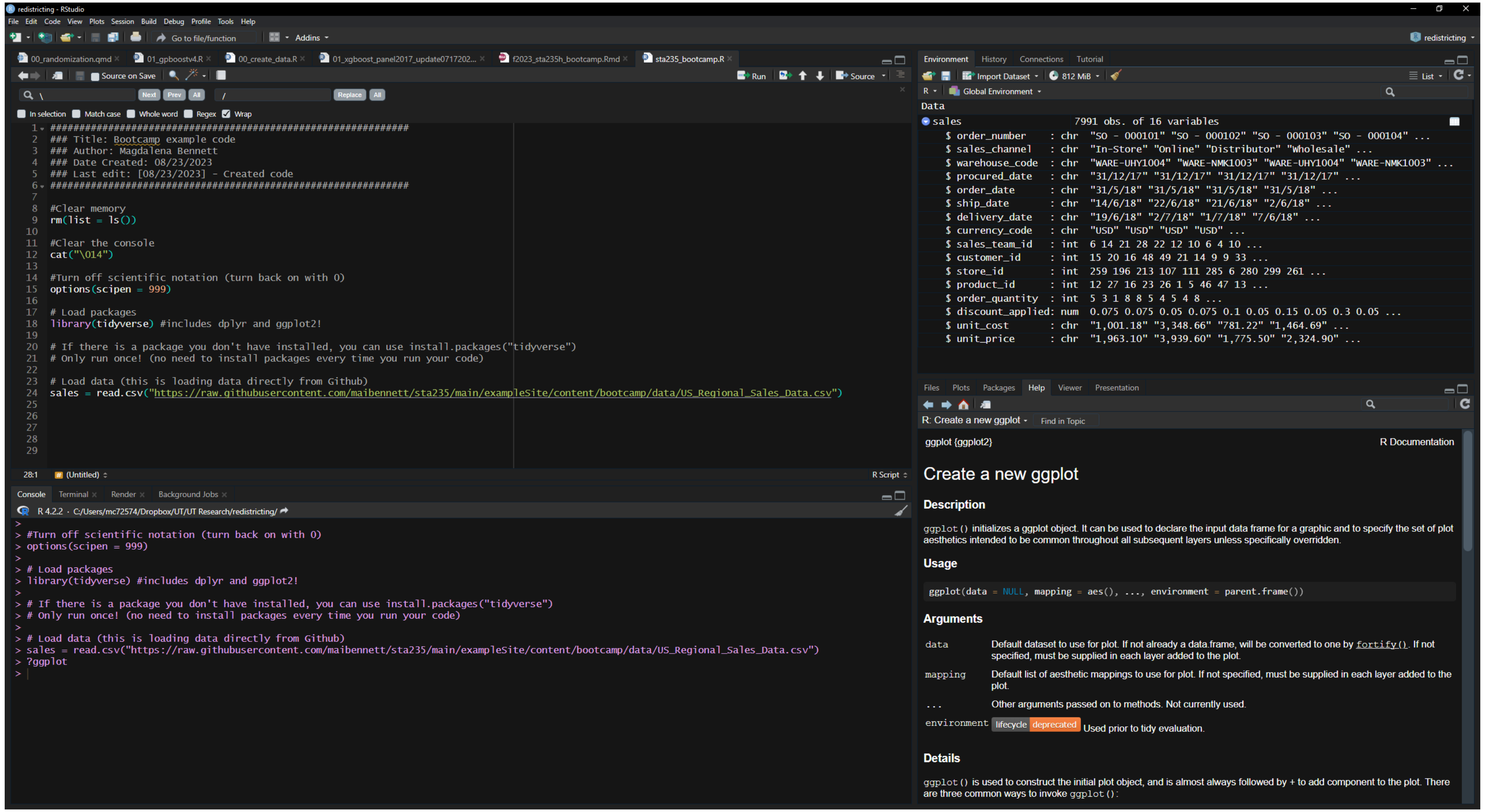


R for coding

R is the programming language we will use for statistical analysis

RStudio is the IDE (Integrated Development Environment) we will use to run R on our computers.

Let's look at RStudio


redistricting - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
redistricting
00_randomization.qmd
01_gpboostv4.R
00_create_data.R
01_xgboost_panel2017_update0717202...
f2023_sta235h_bootcamp.Rmd
sta235_bootcamp.R
Source on Save
Run
Source
Next Prev All
Replace All
In selection Match case Whole word Regex Wrap
28:1
(Untitled)
R Script
Console Terminal Render Background Jobs
R 4.2.2 · C:/Users/mc72574/Dropbox/UT/UT Research/redistricting/
Environment History Connections Tutorial
Import Dataset
812 MiB
List
R
Global Environment
Files Plots Packages Help Viewer Presentation
R: Create a new ggplot
Find in Topic

```r
###########################################################################
### Title: Bootcamp example code
### Author: Magdalena Bennett
### Date Created: 08/23/2023
### Last edit: [08/23/2023] - Created code
###########################################################################

#Clear memory
rm(list = ls())

#Clear the console
cat("\014")

#Turn off scientific notation (turn back on with 0)
options(scipen = 999)

# Load packages
library(tidyverse) #includes dplyr and ggplot2!

# If there is a package you don't have installed, you can use install.packages("tidyverse")
# Only run once! (no need to install packages every time you run your code)

# Load data (this is loading data directly from Github)
sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")
```

```
>
> #Turn off scientific notation (turn back on with 0)
> options(scipen = 999)
>
> # Load packages
> library(tidyverse) #includes dplyr and ggplot2!
>
> # If there is a package you don't have installed, you can use install.packages("tidyverse")
> # Only run once! (no need to install packages every time you run your code)
>
> # Load data (this is loading data directly from Github)
> sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")
> ?ggplot
>
```

```
Data
sales                      7991 obs. of 16 variables
   $ order_number    : chr  "SO - 000101" "SO - 000102" "SO - 000103" "SO - 000104" ...
   $ sales_channel   : chr  "In-Store" "Online" "Distributor" "Wholesale" ...
   $ warehouse_code  : chr  "WARE-UHY1004" "WARE-NMK1003" "WARE-UHY1004" "WARE-NMK1003" ...
   $ procured_date   : chr  "31/12/17" "31/12/17" "31/12/17" "31/12/17" ...
   $ order_date      : chr  "31/5/18" "31/5/18" "31/5/18" "31/5/18" ...
   $ ship_date       : chr  "14/6/18" "22/6/18" "21/6/18" "2/6/18" ...
   $ delivery_date   : chr  "19/6/18" "2/7/18" "1/7/18" "7/6/18" ...
   $ currency_code   : chr  "USD" "USD" "USD" "USD" ...
   $ sales_team_id   : int  6 14 21 28 22 12 10 6 4 10 ...
   $ customer_id     : int  15 20 16 48 49 21 14 9 9 33 ...
   $ store_id        : int  259 196 213 107 111 285 6 280 299 261 ...
   $ product_id      : int  12 27 16 23 26 1 5 46 47 13 ...
   $ order_quantity  : int  5 3 1 8 8 5 4 5 4 8 ...
   $ discount_applied: num  0.075 0.075 0.05 0.075 0.1 0.05 0.15 0.05 0.3 0.05 ...
   $ unit_cost       : chr  "1,001.18" "3,348.66" "781.22" "1,464.69" ...
   $ unit_price      : chr  "1,963.10" "3,939.60" "1,775.50" "2,324.90" ...
```

ggplot {ggplot2} R Documentation

Create a new ggplot

Description

`ggplot()` initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

Usage

```r
ggplot(data = NULL, mapping = aes(), ..., environment = parent.frame())
```

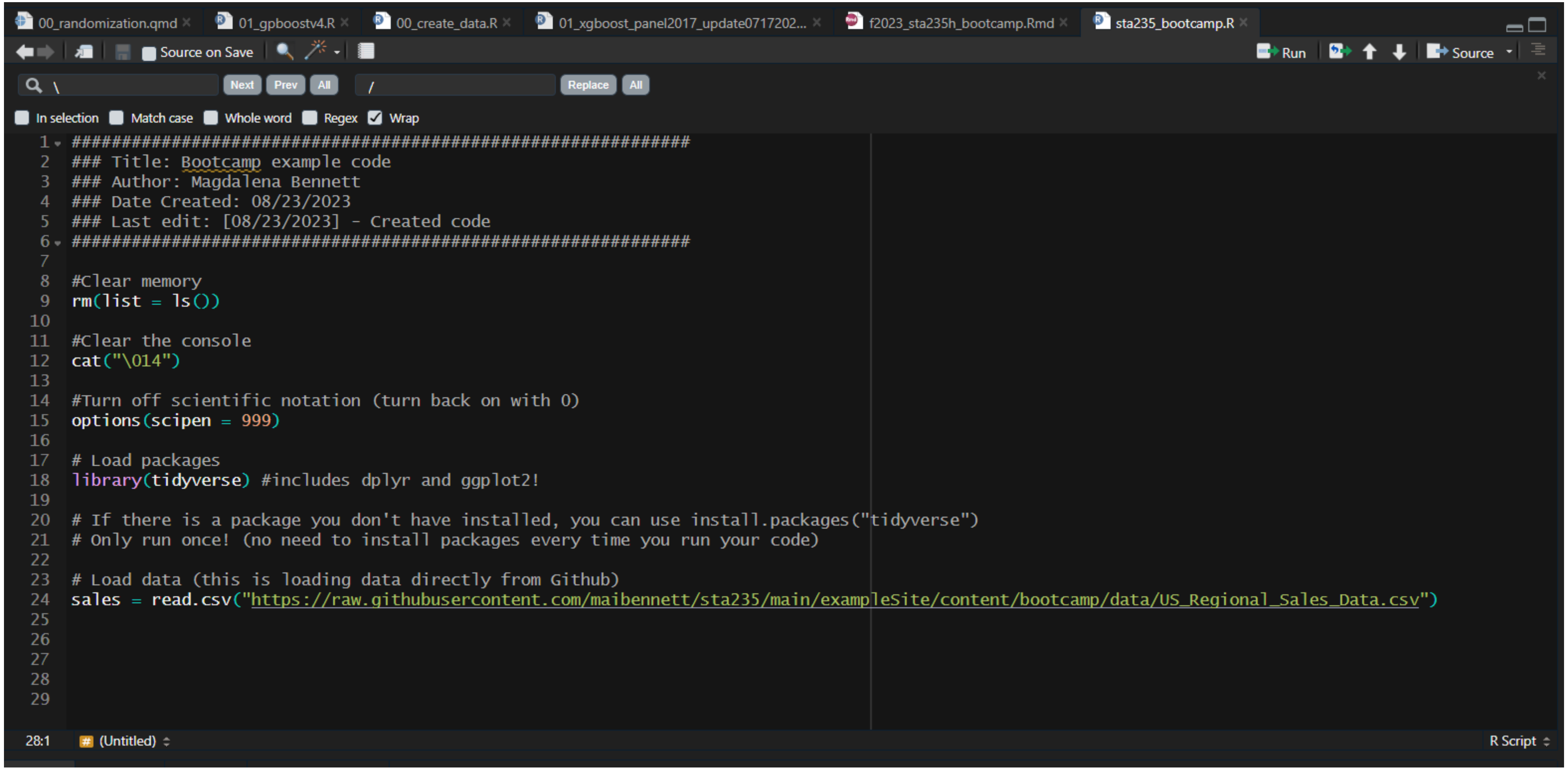
Arguments

`data`	Default dataset to use for plot. If not already a data.frame, will be converted to one by `fortify()`. If not specified, must be supplied in each layer added to the plot.
`mapping`	Default list of aesthetic mappings to use for plot. If not specified, must be supplied in each layer added to the plot.
`...`	Other arguments passed on to methods. Not currently used.
`environment`	lifecycle deprecated Used prior to tidy evaluation.

Details

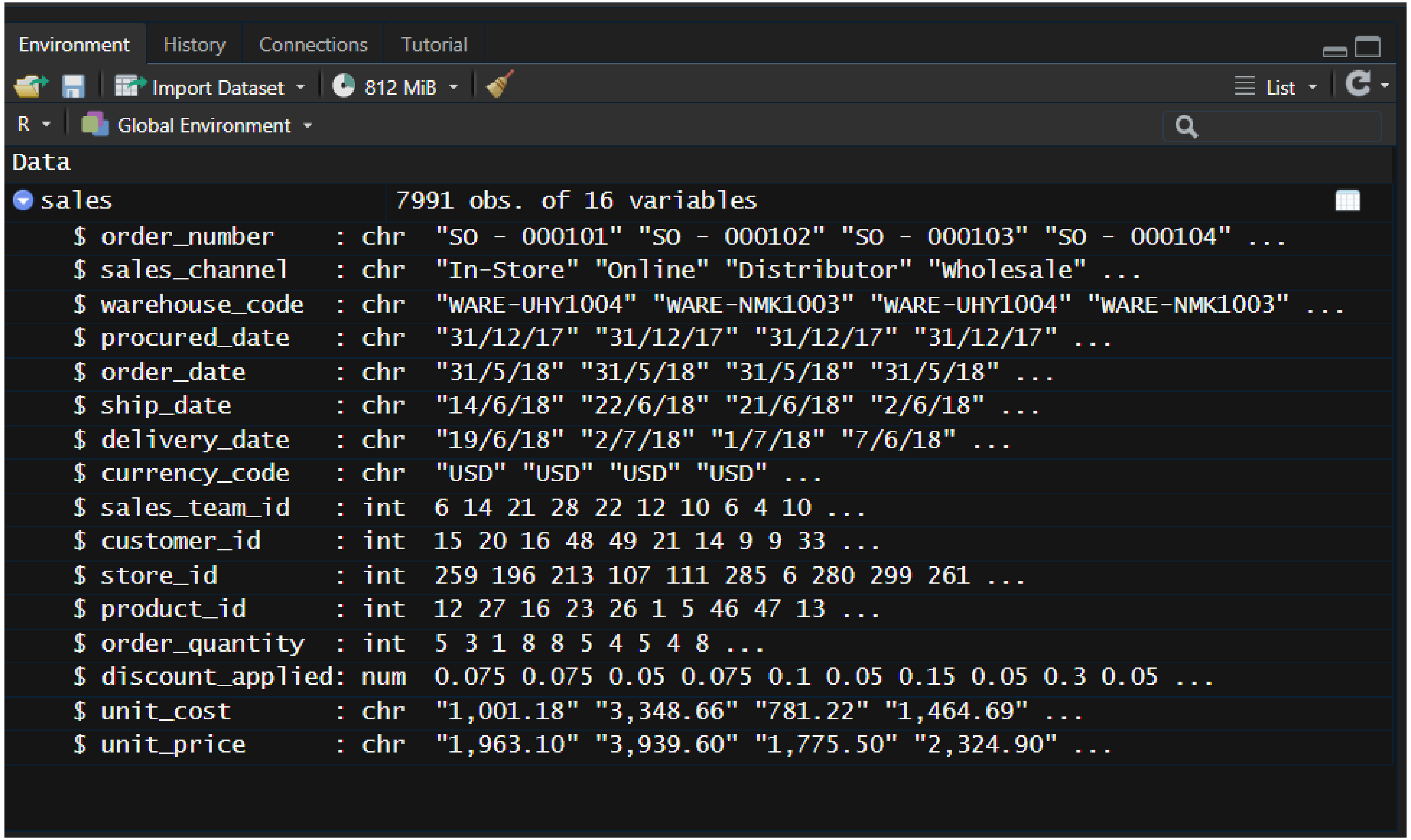
`ggplot()` is used to construct the initial plot object, and is almost always followed by + to add component to the plot. There are three common ways to invoke `ggplot()`:

Let's look at RStudio - Script



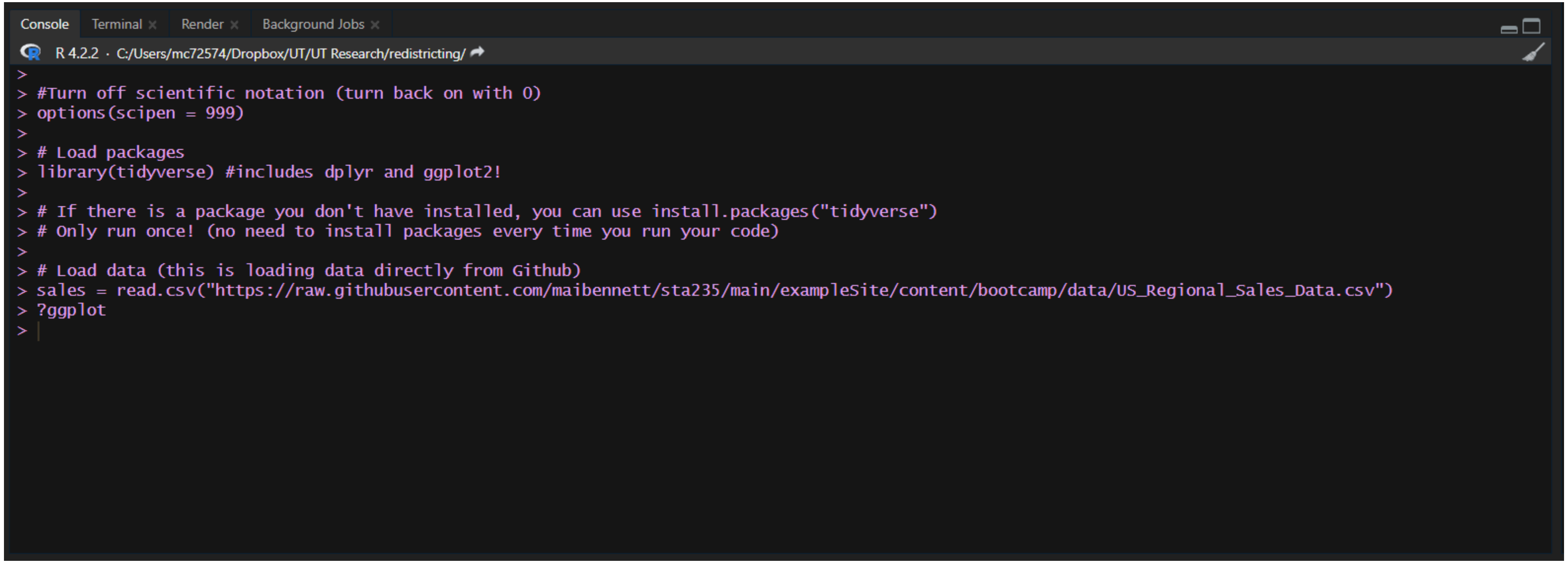
```r
##############################################################
### Title: Bootcamp example code
### Author: Magdalena Bennett
### Date Created: 08/23/2023
### Last edit: [08/23/2023] - Created code
##############################################################

#Clear memory
rm(list = ls())

#Clear the console
cat("\014")

#Turn off scientific notation (turn back on with 0)
options(scipen = 999)

# Load packages
library(tidyverse) #includes dplyr and ggplot2!

# If there is a package you don't have installed, you can use install.packages("tidyverse")
# Only run once! (no need to install packages every time you run your code)

# Load data (this is loading data directly from Github)
sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")
```

Let's look at RStudio - Environment



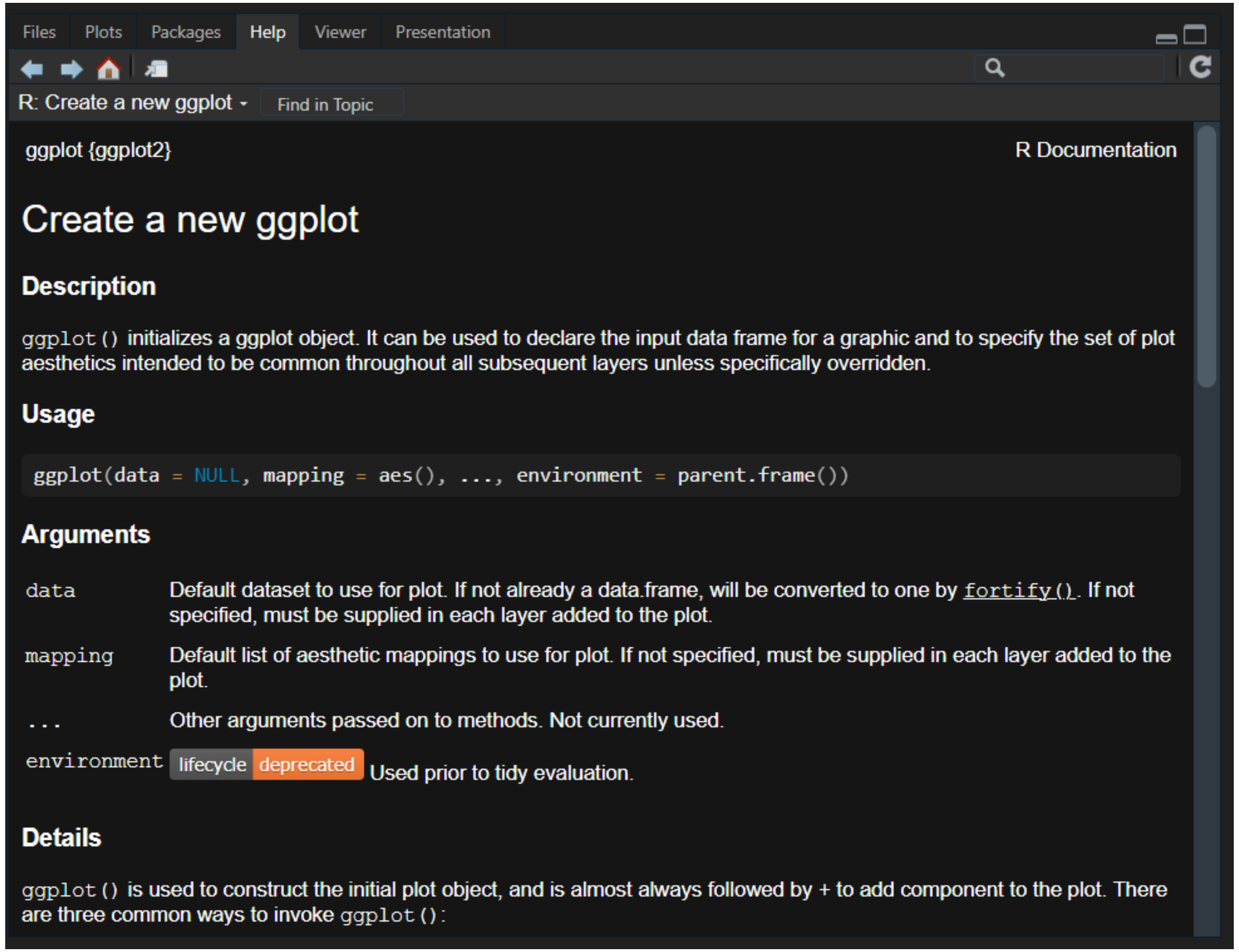
```
Data
sales                    7991 obs. of 16 variables
    $ order_number    : chr  "SO - 000101" "SO - 000102" "SO - 000103" "SO - 000104" ...
    $ sales_channel   : chr  "In-Store" "Online" "Distributor" "Wholesale" ...
    $ warehouse_code  : chr  "WARE-UHY1004" "WARE-NMK1003" "WARE-UHY1004" "WARE-NMK1003" ...
    $ procured_date   : chr  "31/12/17" "31/12/17" "31/12/17" "31/12/17" ...
    $ order_date      : chr  "31/5/18" "31/5/18" "31/5/18" "31/5/18" ...
    $ ship_date       : chr  "14/6/18" "22/6/18" "21/6/18" "2/6/18" ...
    $ delivery_date   : chr  "19/6/18" "2/7/18" "1/7/18" "7/6/18" ...
    $ currency_code   : chr  "USD" "USD" "USD" "USD" ...
    $ sales_team_id   : int  6 14 21 28 22 12 10 6 4 10 ...
    $ customer_id     : int  15 20 16 48 49 21 14 9 9 33 ...
    $ store_id        : int  259 196 213 107 111 285 6 280 299 261 ...
    $ product_id      : int  12 27 16 23 26 1 5 46 47 13 ...
    $ order_quantity  : int  5 3 1 8 8 5 4 5 4 8 ...
    $ discount_applied: num  0.075 0.075 0.05 0.075 0.1 0.05 0.15 0.05 0.3 0.05 ...
    $ unit_cost       : chr  "1,001.18" "3,348.66" "781.22" "1,464.69" ...
    $ unit_price      : chr  "1,963.10" "3,939.60" "1,775.50" "2,324.90" ...
```

Let's look at RStudio - Console


Console
Terminal
Render
Background Jobs
R 4.2.2 · C:/Users/mc72574/Dropbox/UT/UT Research/redistricting/

```
>
> #Turn off scientific notation (turn back on with 0)
> options(scipen = 999)
>
> # Load packages
> library(tidyverse) #includes dplyr and ggplot2!
>
> # If there is a package you don't have installed, you can use install.packages("tidyverse")
> # Only run once! (no need to install packages every time you run your code)
>
> # Load data (this is loading data directly from Github)
> sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")
> ?ggplot
>
```

Let's look at RStudio - Help and others



ggplot {ggplot2} R Documentation

Create a new ggplot

Description

`ggplot()` initializes a ggplot object. It can be used to declare the input data frame for a graphic and to specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

Usage

```
ggplot(data = NULL, mapping = aes(), ..., environment = parent.frame())
```

Arguments

`data`	Default dataset to use for plot. If not already a data.frame, will be converted to one by `fortify()`. If not specified, must be supplied in each layer added to the plot.
`mapping`	Default list of aesthetic mappings to use for plot. If not specified, must be supplied in each layer added to the plot.
`...`	Other arguments passed on to methods. Not currently used.
`environment`	lifecycle deprecated Used prior to tidy evaluation.

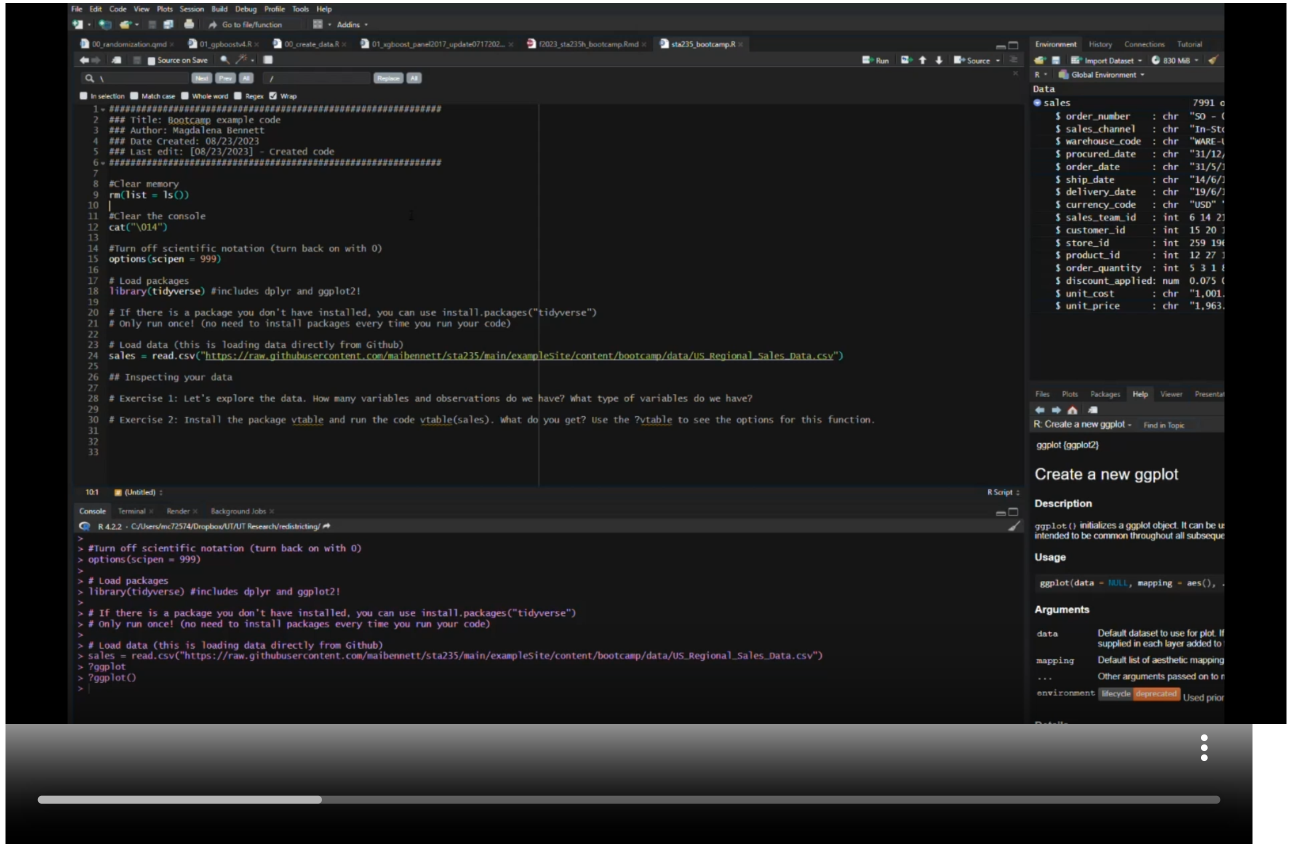
Details

`ggplot()` is used to construct the initial plot object, and is almost always followed by + to add component to the plot. There are three common ways to invoke `ggplot()`:

Useful basic commands

- `install.packages("name")`: Installs the package "name" on your computer. You only need to run this once!
- `library(name)`: Loads the package "name" on your current session. You should do this at the top of every script and only include packages you will use (to avoid confusion)
- `?function`: Opens the help file for `function` (if there is more than one `function` -- e.g. different libraries -- you can choose which one you open).

Also... don't restore RData into a new session!



```r
##################################################################
### Title: Bootcamp example code
### Author: Magdalena Bennett
### Date Created: 08/23/2023
### Last edit: [08/23/2023] - Created code
##################################################################

#Clear memory
rm(list = ls())

#Clear the console
cat("\014")

#Turn off scientific notation (turn back on with 0)
options(scipen = 999)

# Load packages
library(tidyverse) #includes dplyr and ggplot2!

# If there is a package you don't have installed, you can use install.packages("tidyverse")
# Only run once! (no need to install packages every time you run your code)

# Load data (this is loading data directly from Github)
sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")

## Inspecting your data

# Exercise 1: Let's explore the data. How many variables and observations do we have? What type of variables do we have?

# Exercise 2: Install the package vtable and run the code vtable(sales). What do you get? Use the ?vtable to see the options for this function.
```

```
R 4.2.2 · C:/Users/mc72574/Dropbox/UT/UT Research/redistricting/
>
> #Turn off scientific notation (turn back on with 0)
> options(scipen = 999)
>
> # Load packages
> library(tidyverse) #includes dplyr and ggplot2!
>
> # If there is a package you don't have installed, you can use install.packages("tidyverse")
> # Only run once! (no need to install packages every time you run your code)
>
> # Load data (this is loading data directly from Github)
> sales = read.csv("https://raw.githubusercontent.com/maibennett/sta235/main/exampleSite/content/bootcamp/data/US_Regional_Sales_Data.csv")
> ?ggplot
> ?ggplot()
>
```

```
Data
sales                                7991 o
    $ order_number    : chr  "SO - (
    $ sales_channel   : chr  "In-Sto
    $ warehouse_code  : chr  "WARE-U
    $ procured_date   : chr  "31/12/
    $ order_date      : chr  "31/5/1
    $ ship_date       : chr  "14/6/1
    $ delivery_date   : chr  "19/6/1
    $ currency_code   : chr  "USD" '
    $ sales_team_id   : int  6 14 21
    $ customer_id     : int  15 20 1
    $ store_id        : int  259 196
    $ product_id      : int  12 27 1
    $ order_quantity  : int  5 3 1 8
    $ discount_applied: num  0.075 0
    $ unit_cost       : chr  "1,001.
    $ unit_price      : chr  "1,963.
```

ggplot {ggplot2}

Create a new ggplot

Description

ggplot() initializes a ggplot object. It can be us
intended to be common throughout all subseque

Usage

```
ggplot(data = NULL, mapping = aes(), .
```

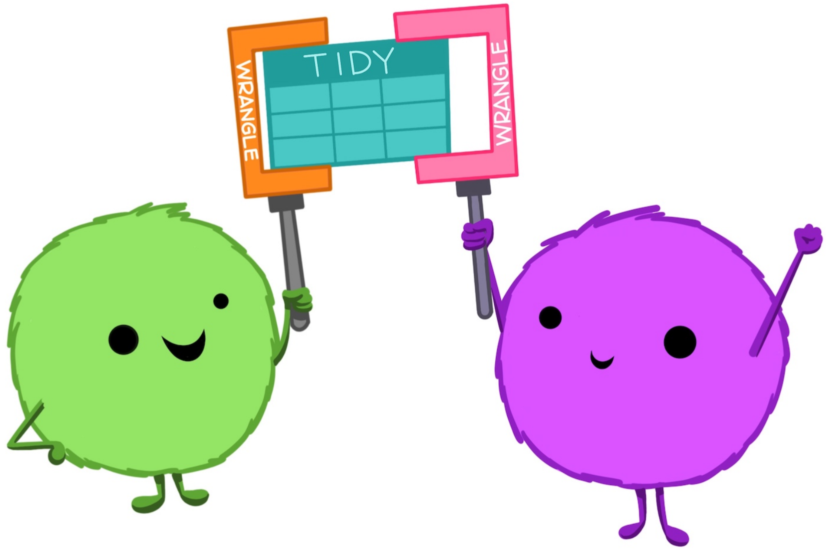
Arguments

data	Default dataset to use for plot. If supplied in each layer added to
mapping	Default list of aesthetic mapping
...	Other arguments passed on to m
environment	lifecycle deprecated Used prior

Let's go to R

Data Wrangling

- Most times we need to **transform**, **clean**, and **structure** data for analysis.
- Examples of data wrangling would be dropping missing observations, merging different datasets, identifying outliers, etc.
- **R can help us do that!**



Into the tidyverse

- For data wrangling, we will use the tidyverse: Collection of packages that follow a similar design structure (e.g. dplyr, ggplot2)
- It works through pipes: %>%
  - Concatenates functions!

Useful functions for wrangling

- `mutate(var = var1 + var2)`: Creates a new variable or replaces an existing one. It takes as an argument the name of the variable and what you want that variable to be.
- `filter(var == 1)`: Subsets your data according to a logic statement. Remember that logic statements use "==" instead of "="!
- `group_by(var1, var2)`: Used to group observations by values of different variables. You can use it either to create a variable with values at the group level, or to summarize your dataset by group.
- `select(var1, var2)`: Select specific variables from the dataset (drop the others). In case you want to drop instead of keeping variables, you can use `select(-var1, -var2)`
- `rename(var_new = var_old)`: The name says it all. Used to rename variables.

Other useful functions

- `is.na(var)`: logic function that returns TRUE if the observation is a missing value (NA) or FALSE in another case.
- `ifelse(logic_statement, val1, val2)`: Very useful function to create conditional values.
- `!(logic_statement)`: The exclamation point acts as a negation. If you want to invert a logic statement, use this (e.g. `!is.na(var)` will return TRUE if the obs of `var` is NOT missing and FALSE if it's missing).
- `table(var)`: Tabulates the different values of a variable

Let's go to R

Plotting in R

- Plotting your data is a very intuitive way to see what's going on.
- It's also useful to convey complex analysis!
- Make sure your plots are always informative and they tell the story you want to highlight.

General structure of ggplot

- `ggplot()` works in **"layers"**:
  - You can provide different geometries and "add" them to your plot (same with themes!)
- You always start with `ggplot(data = d, aes(x = var1, y = var2, color = var3))`, depending on what you want to do:
  - `aes()` stands for aesthetics, and it tells which variables you want to use and how. Sometimes you need one variable (e.g. histogram), sometimes you need two (e.g. scatter plot), or even three or more! (e.g. scatter plot for different groups)
- You can provide `aes()` in the `ggplot()` function (as seen above), or also in each geometric layer:

  e.g. `ggplot(data = d) + geom_point(aes(x = var1, y = var2))`

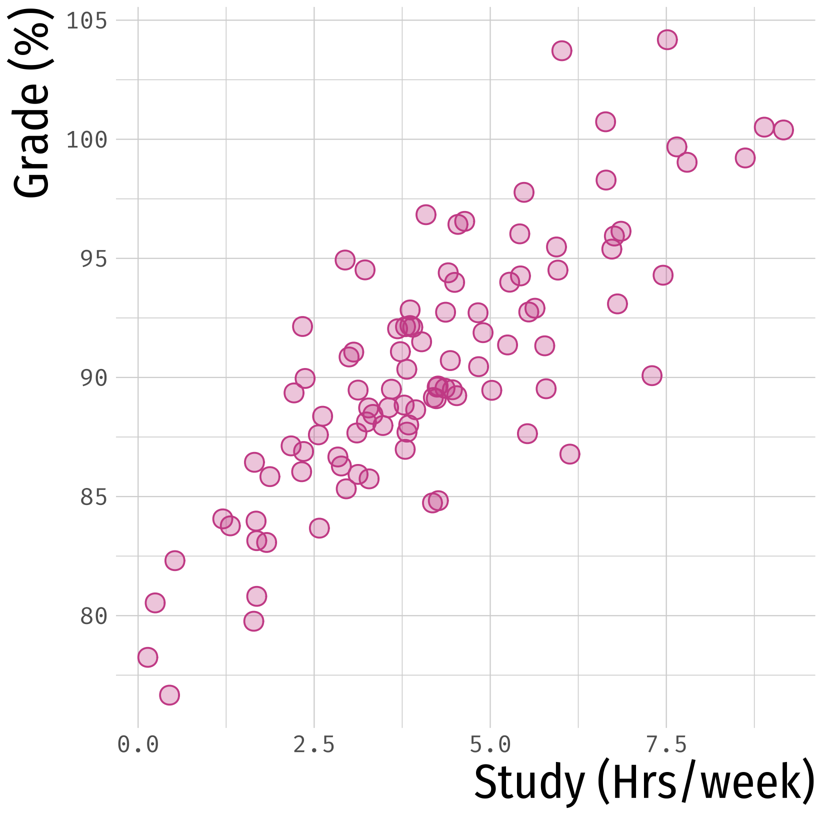
General structure of ggplot

- Some common geometries that are useful:
  - `geom_point()`: Creates a scatter plot
  - `geom_line()`: Creates a line plot
  - `geom_histogram()` or `geom_density()`: Creates a histogram or a density plot for your data!
  - `geom_smooth()`: Creates a smooth function that goes through your data. By default, it uses a loess or gam function, depending on the size of the data. Use `method = "lm"`as an argument if you want to fit a regression line!
- Finally, looks are also important!
  - `theme()` allows you to play around with every aspect of your plot (e.g. font size, grid lines, etc.)
  - Using a pre-packages theme can be useful, too. I personally like `theme_minimal()` or the `theme_ipsum_rc()` from the `hrbrthemes` package.

Let's go to R

Regression Analysis

- Regressions help us **quantify the relationship** between different variables.
- In R, we can get **many important insights** from regression analysis!

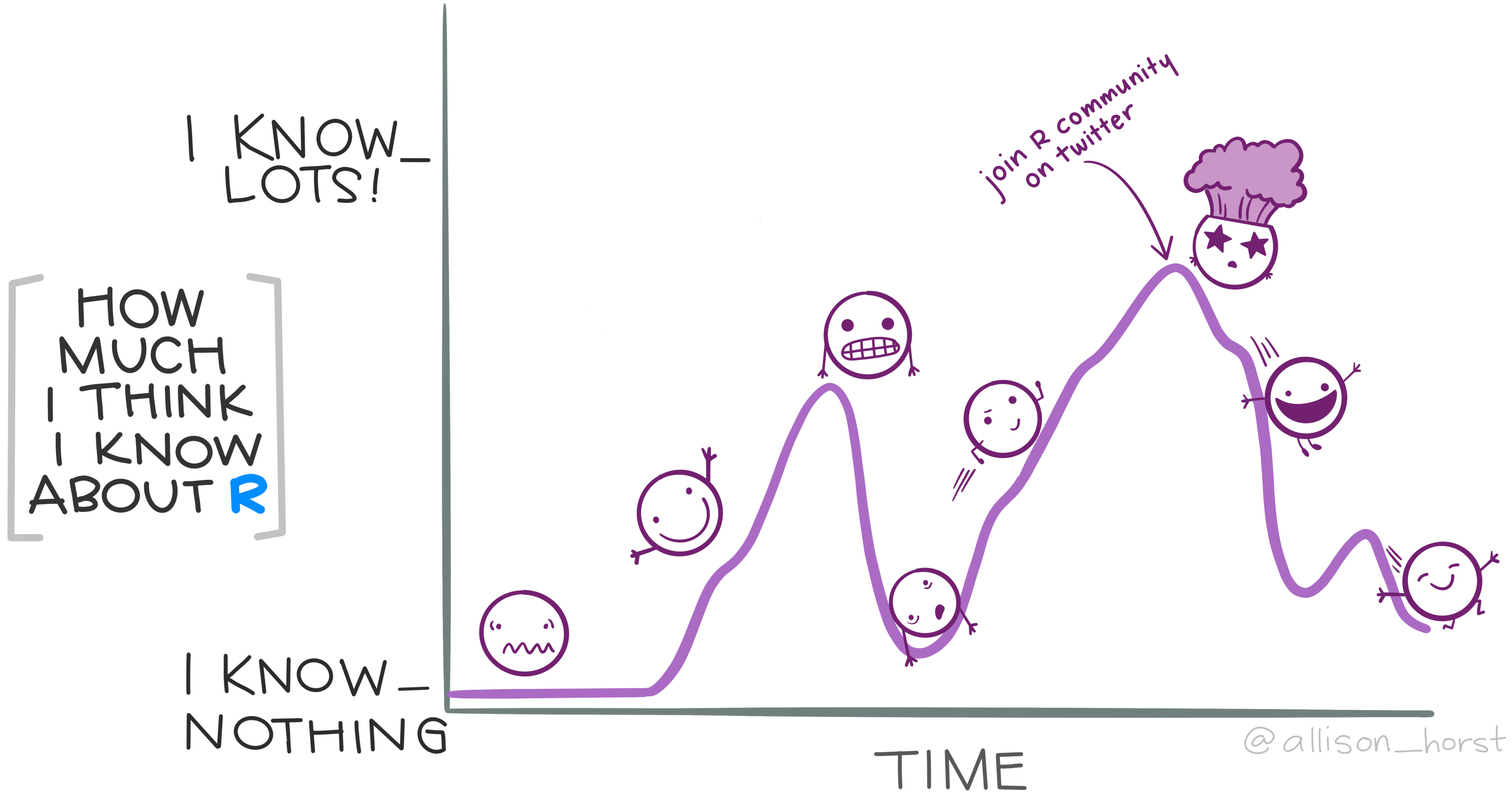


Regressions in R

- The main command to do regressions is `lm(y ~ x1 + x2, data = d)`, where `y` is our outcome of interest and `x1` and `x2` are regressors.
- For convenience, we can store the regression in a separate object (e.g. `lm1 = lm(y ~ x1 + x2, data = d)`), so we can later manipulate it:
  - `summary(lm1)`: Provides a summary table of the results (including estimates, standard errors, and p-values).
  - `lm1$coefficients`: Recovers the exact estimated coefficients (e.g. useful if you want to use them later).
  - `summary(lm1)$coefficients`: Matrix of results. Includes columns for the estimates betas, standard errors, t-stats, and p-values.

Let's go to R

R is useful and fun!


I KNOW LOTS!
HOW MUCH I THINK I KNOW ABOUT R
I KNOW NOTHING
TIME
join R community on twitter
@allison_horst