

DataCamp Covid-19 Project

DSC 205 Introduction to Advanced Data Science

Table of Contents

- [README](#)
- [Aspects of this project \(discussion\)](#)
- [From epidemic to pandemic](#)
- [Confirmed cases throughout the world](#)
- [China compared to the rest of the world](#)
- [Let's annotate!](#)
- [Adding a trend line to China](#)
- [And the rest of the world?](#)
- [Adding a logarithmic scale](#)
- [Which countries outside of China have been hit hardest?](#)
- [Plotting hardest hit countries as of Mid-March 2020](#)
- [References](#)

README

- Demo of ggplot2 using early pandemic COVID data
- The material covered comes from the following DataCamp courses:
 - [Data Visualization with ggplot2](#) (a graphics package for R)
 - [Intermediate Data Visualization with ggplot2](#)
 - [Data Manipulation with dplyr](#) (a table manipulation package in R)
 - [Introduction to Importing Data in R](#) (using `read.table`)
- To look at the whole project at once [see here](#).
- We'll revisit some topics in our gapminder project

Aspects of this project (discussion)

The table 1 lists some (**variable**) aspects of this project. In practice, you would revisit these before, during and after the project.

ASPECT	CONTENT	DECISIONS
Technical	R packages	Base R, ggplot2
Scientific	Data science pipeline	Data quality, presentation
Political	Policy	Funding, trust, future
Personal	Pro or Con	Do it or don't, dig deeper

Which aspects did you find interesting and why (not)?

Personal

- Alright - "we've talked about COVID a lot"

- These plots were new?
- I somewhat disliked this project

Technical

- Fun to create fancy graphs with little effort
- Is ggplot2 really necessary?
- dplyr - table manipulation (similar to SQL)
- linear modeling with the glm() family of functions

Scientific

- You should always try to go to the source of the data (In this case: JHU)
- Data quality in this area needs to be questioned
- Issues are: collection procedures and measures

Political

- Ongoing issue (with every new variant)

Example: getting the data for your own project

- I found this dataset via
 1. Google search of `datasets/confirmed_cases_worldwide.csv`
 2. The site ourworldindata.org uses JHU as the main data source
 3. Data import from [GitHub](#) - you need the raw source
- Or you can select the Jan-Mar 17, 2020 timeline and download the CSV data from the "[COVID-19 Data Explorer](#)", a graphical dashboard.
- From the "[COVID-19 Data Explorer](#)".

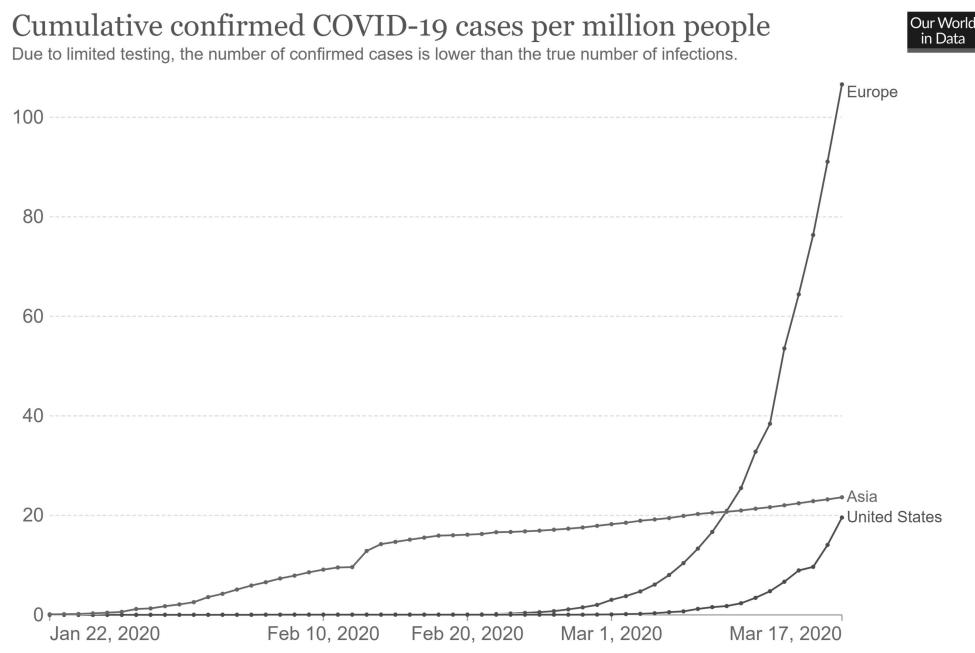


Figure 1: Cumulative cases (Asia, Europe, USA) Jan-March 17, 2020 (Source: OWID)

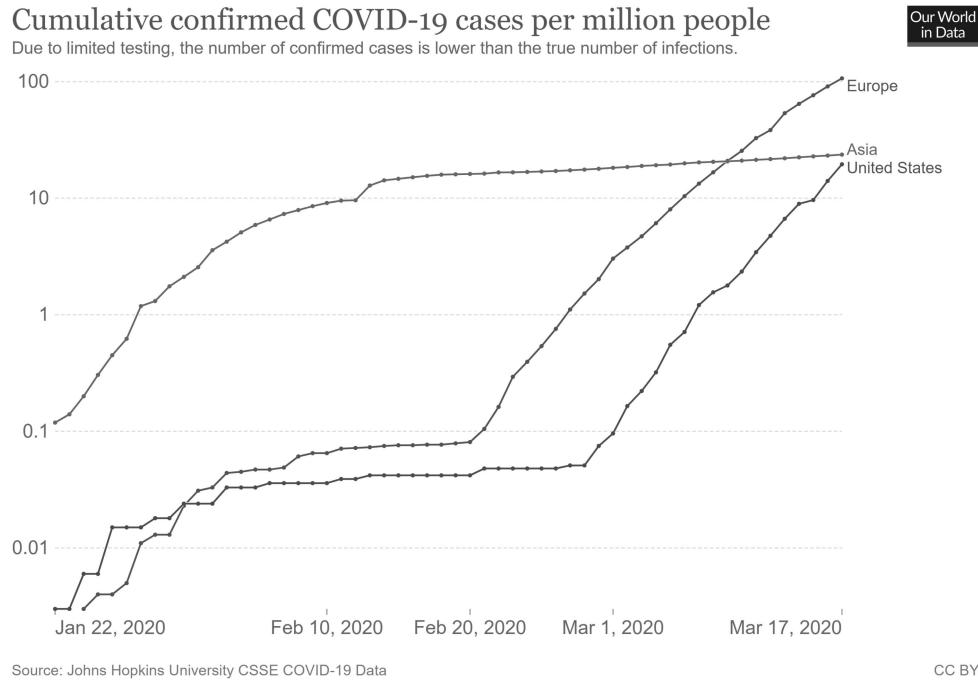


Figure 2: Cumulative cases (Asia, Europe, USA) Jan-March 17, 2020 (Source: OWID)

- [] What's wrong with just using such a nice dashboard solution?

From epidemic to pandemic

Load packages

- We load three different packages - for data import (`readr`), graphics (`ggplot2`) and table manipulation (`dplyr`).
- They're contained in the "Tidyverse" package but it's preferable to load them individually.
- [X]

Why is it better to load packages individually?

```
## Load the readr, ggplot2, and dplyr packages
library(readr)
library(ggplot2)
library(dplyr)
```

- Answer: performance and memory issues aside, packages rely on a limited supply of function names. When different packages use the same name, the function name of the older (less recently installed) package is masked - usually this is the base R function name. This can lead to confusion or error.

Reading data into a data frame using `readr::read_csv`

- We use `readr::read_csv` to read a (local) CSV file into a data frame
- [X]

What's a data frame again? Can you define it?

"A data frame is ... a list of variables of the same number of rows with unique row names, given class "data.frame". If no variables are included, the row names determine the number of rows." (`data.frame` help page)

Alternative definition: "A data frame is a table or a two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column." (tutorialspoint.com)

- [X] The command creates a number of substructures

```
## Read datasets/confirmed_cases_worldwide.csv into confirmed_cases_worldwide
confirmed_cases_worldwide <- read_csv("data/covid.csv")

## Try to fix tibble display problems in Emacs + ESS
## Source: https://github.com/emacs-ess/ESS/issues/810
options(crayon.enabled = FALSE)

## Print out confirmed_cases_worldwide
str(confirmed_cases_worldwide)
class(confirmed_cases_worldwide)
```

```
Error: 'data/covid.csv' does not exist in current working directory ('c:/Users/birkenkrahe').
Error in str(confirmed_cases_worldwide) :
  object 'confirmed_cases_worldwide' not found
Error: object 'confirmed_cases_worldwide' not found
```

`readr::read_csv`

"The goal of `readr` is to provide a fast and friendly way to read rectangular data from delimited files, such as comma-separated values (CSV) and tab-separated values (TSV). It is designed to parse many types of data found in the wild, while providing an informative problem report when parsing leads to unexpected results. If you are new to `readr`, the best place to start is the data import chapter in R for Data Science." ([online documentation](#))

- [X] Check the documentation for `read_csv` and its many options.
- [X] Test the claims made in the quote [1](#) by reading the Pima Indians diabetes data set (in GDrive - download [via Kaggle](#)).
- [X] Try to extract the ZIP file itself first, then the unzipped CSV file
- [X] After extraction, print the data structure and the first few lines (each block should have three statements).
- [X]

Read the archive ZIP file first.

```
pima_archive <- read_csv("data/archive.zip")
str(pima)
head(pima_archive)
```

```
Error: 'data/archive.zip' does not exist in current working directory ('c:/Users/birkenkra...  
Error in str(pima) : object 'pima' not found  
Error in head(pima_archive) : object 'pima_archive' not found
```

- [X]

Read the CSV file.

```
pima <- read_csv("data/pima-indians-diabetes.csv")  
str(pima)  
head(pima)
```

```
Error: 'data/pima-indians-diabetes.csv' does not exist in current working directory ('c:/U...  
Error in str(pima) : object 'pima' not found  
Error in head(pima) : object 'pima' not found
```

- [X]

For comparison, extract the archive and the CSV data using the Base R function `read.csv`.

Start with the CSV file "pima-indians-diabetes.csv".

```
pima_base <- read.csv("data/pima-indians-diabetes.csv")  
str(pima_base)  
head(pima_base)
```

```
Error in file(file, "rt") : cannot open the connection  
In addition: Warning message:  
In file(file, "rt") :  
  cannot open file 'data/pima-indians-diabetes.csv': No such file or directory  
Error in str(pima_base) : object 'pima_base' not found  
Error in head(pima_base) : object 'pima_base' not found
```

- []

Convince yourself that the Base R function cannot read the ZIP file (leave `:results silent`, otherwise Emacs will get confused by spurious characters in the ZIP file).

```
pima_base_archive <- read.csv("data/archive.zip")  
str(pima_base_archive)  
head(pima_base_archive)
```

Reading data into a data frame using Base R's `read.csv`

- []

We use `read.csv` to read a (local) CSV file into a data frame.

```
## Read data into cases
cases <- read.csv("data/covid.csv")

## Print out confirmed_cases_worldwide
str(cases)
class(cases)
```

```
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'data/covid.csv': No such file or directory
Error in str(cases) : object 'cases' not found
Error: object 'cases' not found
```

utils::read.table

The `read.csv` command belongs to the `read.table` family of functions as calling the help function will tell you.

"Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file."

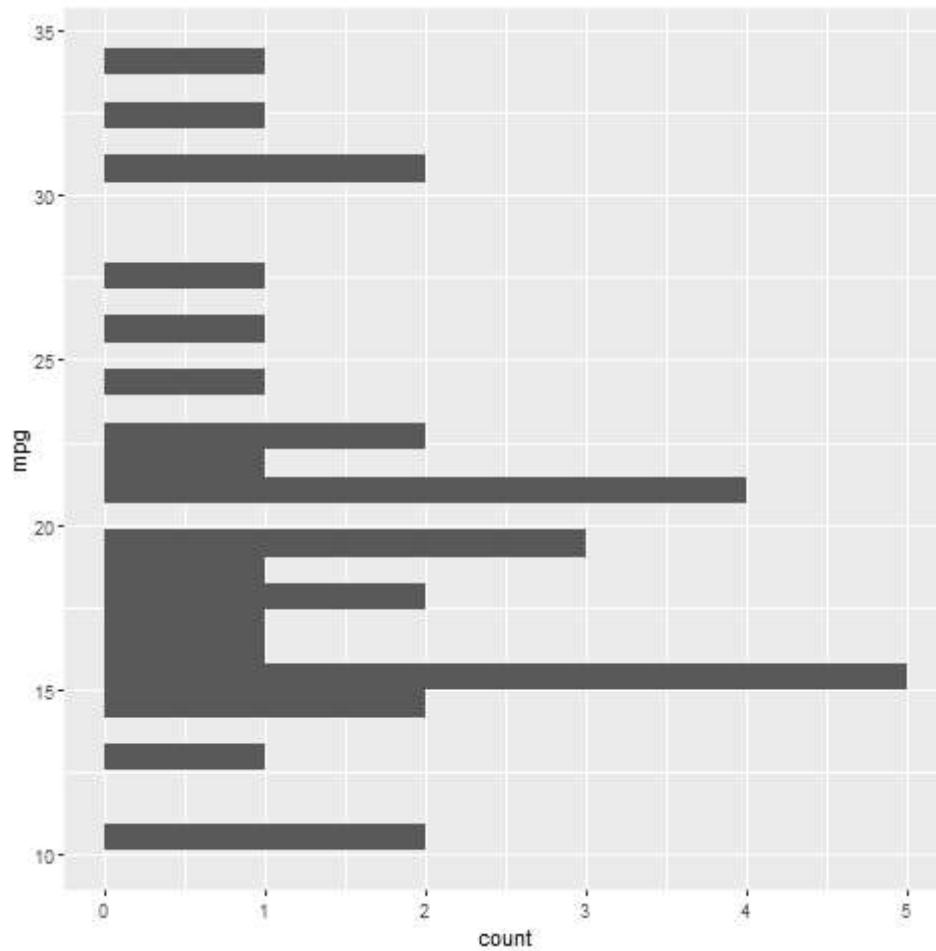
- Lines and fields are synonyms for rows and columns, resp.
- Cases or records, and variables or vectors are the corresponding names for the data structure (data frame)
- It is often important to distinguish between data in the real world (usually the result of real observations) and their representation by a machine

Confirmed cases throughout the world

Basics: data and layout (aes and geom)

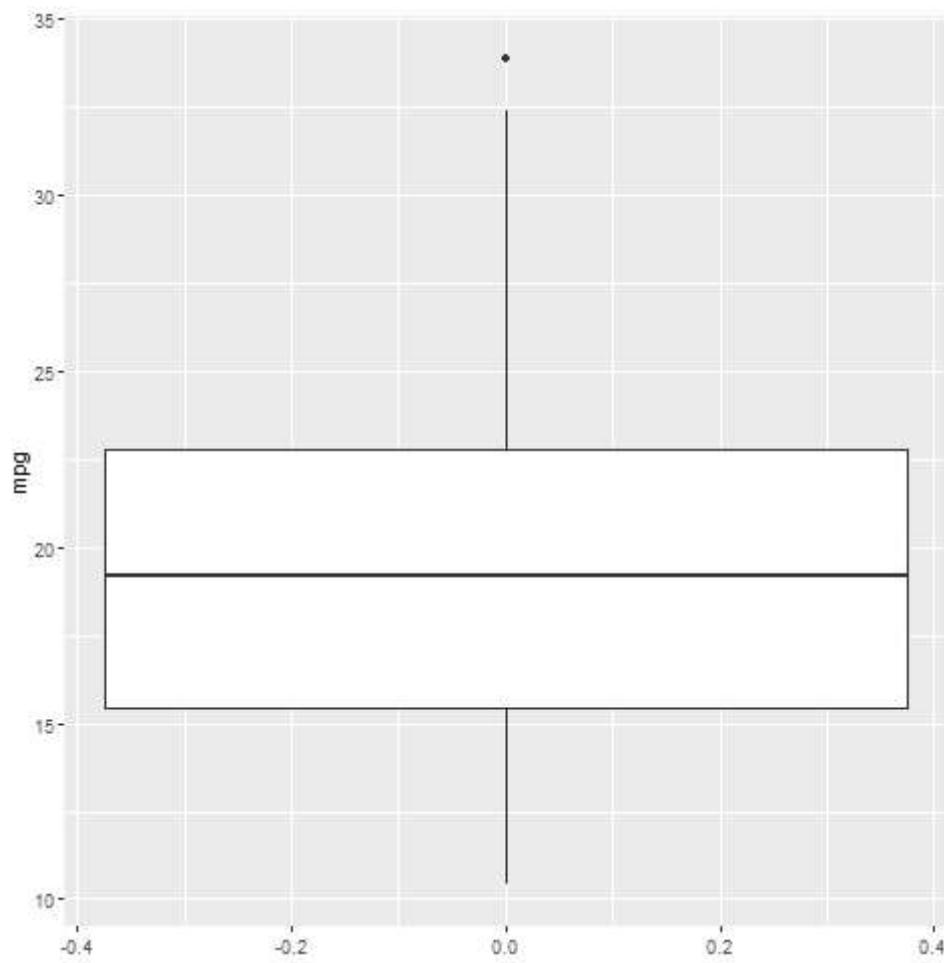
- To get this plot from the downloaded data, the `aes` argument has to be adapted accordingly.
- Remember: `aes[thetics]` means data, as in `x` and `y` for 2d plots, while `geom[etry]` means layout
- [] What does `ggplot` do if only one argument is given to `aes`?
- [] Answer: Depends on the layout function! `geom_hist` works (counts occurrences), and so does `geom_boxplot`, but `geom_point()` does not. A few examples:
 - Histogram: what is the default binwidth? Change it to 5. Tip: `binwidth=5` is a layout parameter.

```
ggplot(data=mtcars, aes(y=mpg)) +
  geom_histogram()
```



- Boxplot: without `scale_x_discrete()`, the x-axis would show a scale [-0.5,0.5].

```
ggplot(data=mtcars, aes(y=mpg)) +  
  geom_boxplot()
```



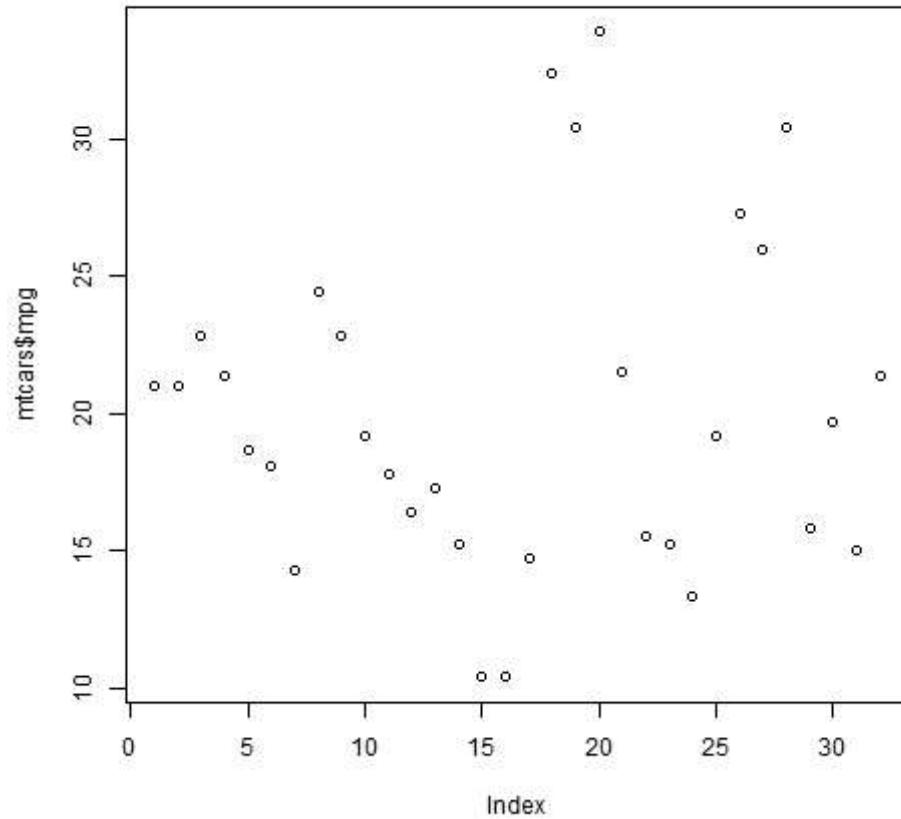
base::plot is generic

- [X]

Compare this with `base::plot`, which is a generic function capable of adapting to different data structures.

Plot the variable `mpg` of the data frame `mtcars`.

```
plot(mtcars$mpg)
```



- [X]

To see which data structures `plot` can digest, check `methods(plot)`.

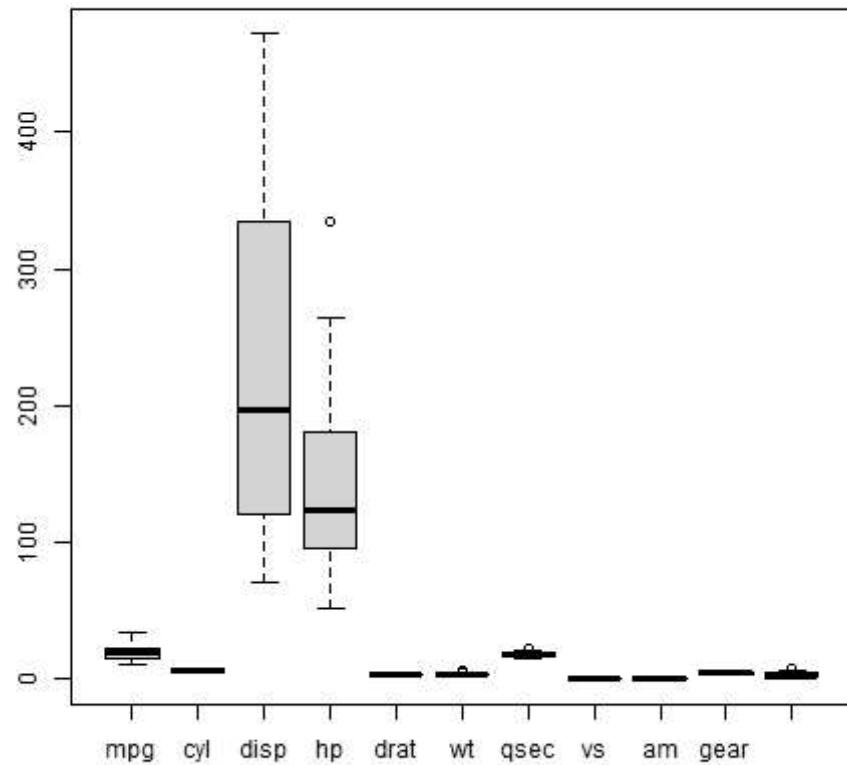
```
methods(plot)
```

```
[1] plot,ANY-method      plot,color-method    plot.acf*
[4] plot.data.frame*    plot.decomposed.ts*  plot.default
[7] plot.dendrogram*    plot.density*       plot.ecdf
[10] plot.factor*        plot.formula*       plot.function
[13] plot.ggpplot*       plot.gtable*        plot.hcl_palettes*
[16] plot.hclust*         plot.histogram*     plot.HoltWinters*
[19] plot.isoreg*         plot.lm*            plot.medpolish*
[22] plot.mlm*            plot.ppr*           plot.prcomp*
[25] plot.princomp*       plot.profile.nls*   plot.R6*
[28] plot.raster*          plot.spec*          plot.stepfun
[31] plot.stl*             plot.table*         plot.trans*
[34] plot.ts               plot.tskernel*      plot.TukeyHSD*
see '?methods' for accessing help and source code
```

- [X]

The `boxplot` function will display every available numerical variable. Make a boxplot of `mtcars`.

```
boxplot(mtcars)
```

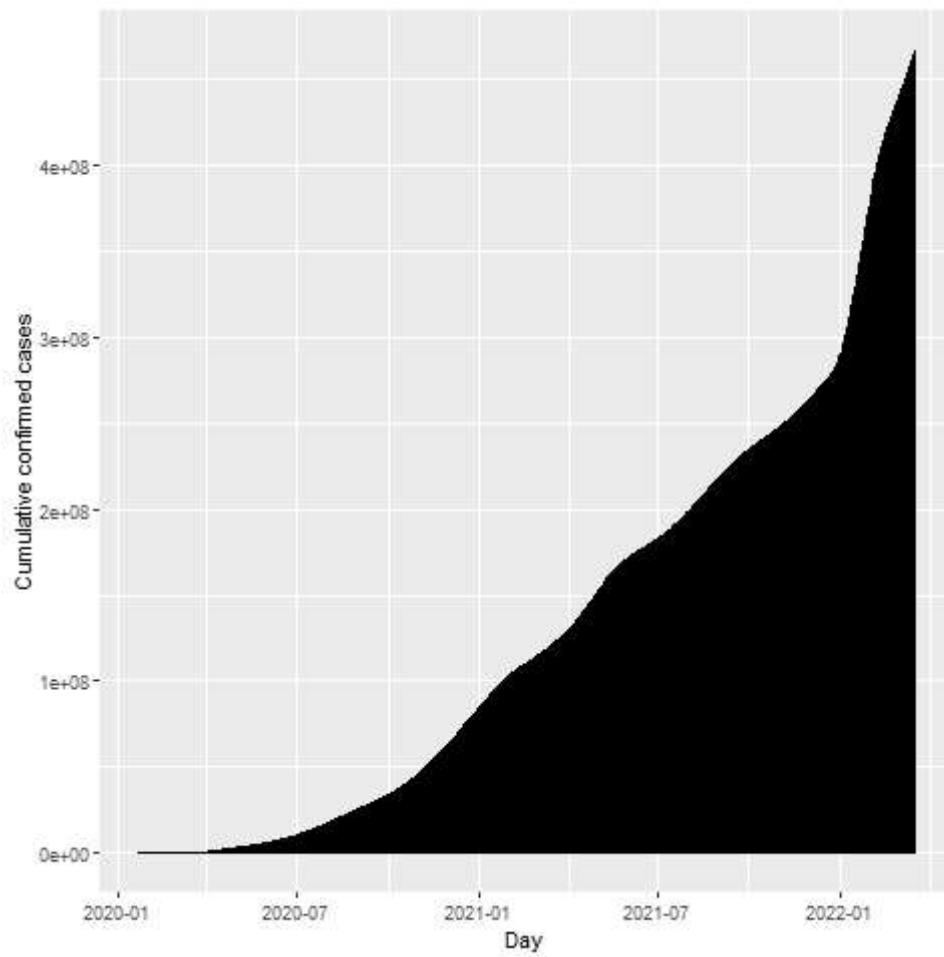


Plotting a line graph straight from the full data set

- [X]

Let's first run this code in 1 using the full, current data set.

```
## Draw a line plot of cumulative cases vs. date
## Label the y-axis
ggplot(
  confirmed_cases_worldwide,
  aes(x=Day, y=`Total confirmed cases of COVID-19`)) +
  geom_line() +
  ylab("Cumulative confirmed cases")
```

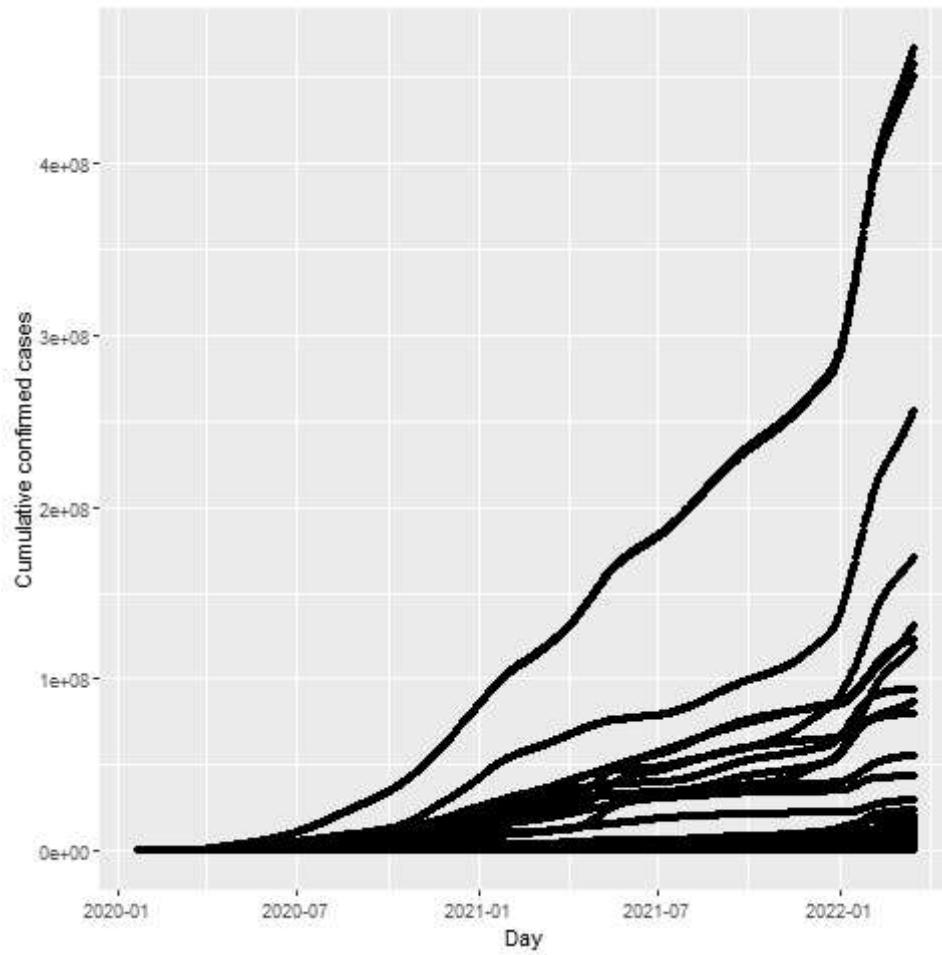


Plot points not lines

- [X]

If you change `geom_line()` to `geom_point()`, you see the individual lines (for each entity, or country): the cumulative case line is the enveloping line for all of them.

```
## Draw a line plot of cumulative cases vs. date
## Label the y-axis
ggplot(confirmed_cases_worldwide,
       aes(x=Day,
           y=`Total confirmed cases of COVID-19`))+  
  geom_point() +  
  ylab("Cumulative confirmed cases")
```



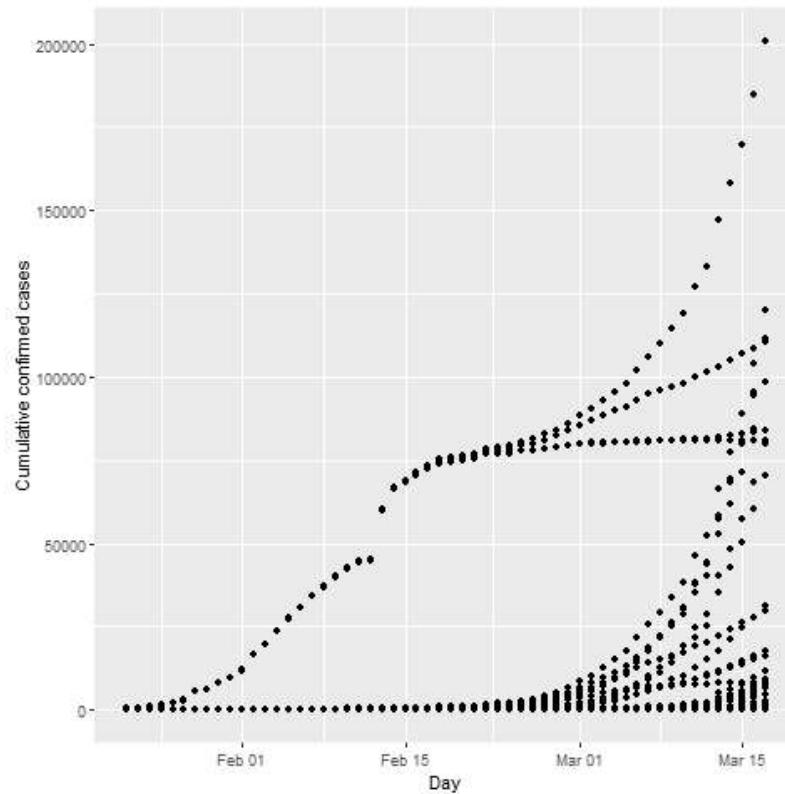
Limit the data set by filtering

- [X]

To narrow the data to the day range covered by the DataCamp project, you can use `dplyr::filter` applied to the `Day` variable. This function filters all values for which the argument is TRUE.

- Use `geom_point` for the plot layout.
- Pipe the data set into `filter()`
- Filter those records earlier than March 18, 2020

```
## Draw a point plot of cases vs. date
## Label the y-axis
confirmed_cases_worldwide %>%
  filter(Day < "2020-03-18") %>%
  ggplot(
    aes(
      x=Day,
      y=`Total confirmed cases of COVID-19`)) +
  geom_point() +
  ylab("Cumulative confirmed cases")
```



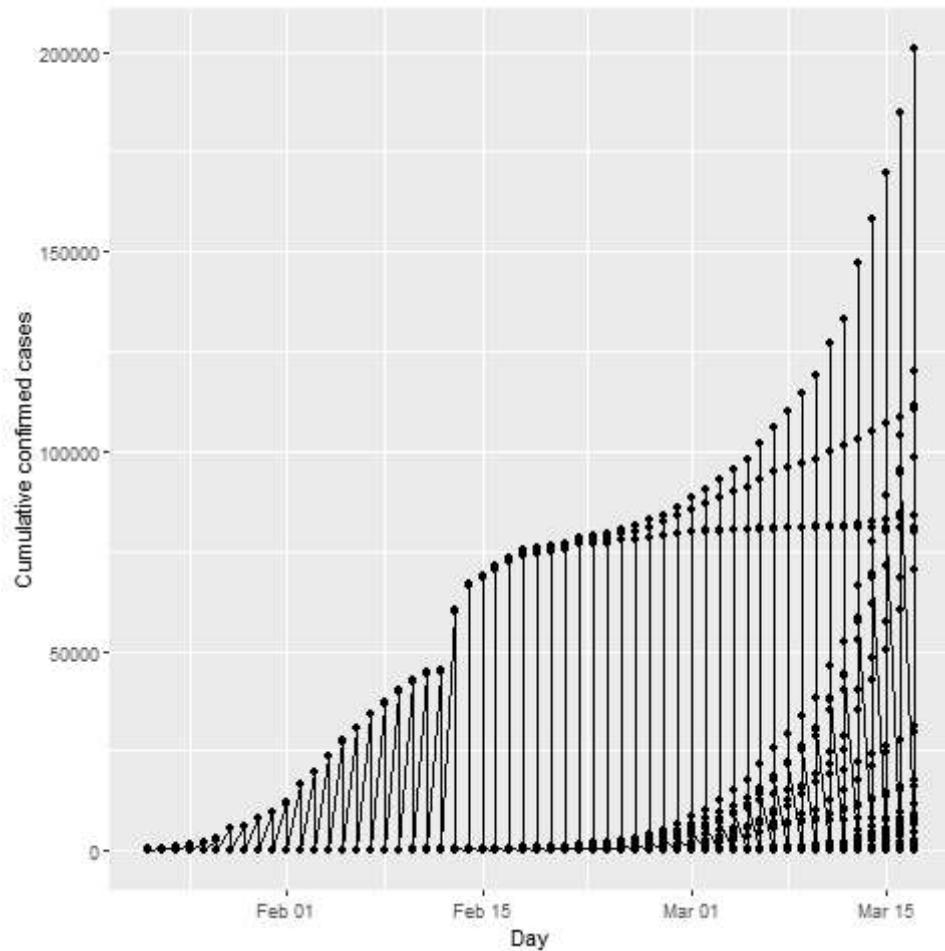
Use both lines and points

- [X]

Experiment with mixing both point and line layout: use both layouts in the same plot! Remember that layouts are layered.'

Below is the code from above. Alter it accordingly and run it.

```
## Draw a plot of cumulative cases vs. date
## Label the y-axis
confirmed_cases_worldwide %>%
  filter(Day < "2020-03-18") %>%
  ggplot(
    aes(
      x=Day,
      y=`Total confirmed cases of COVID-19`)) +
  geom_line() +
  geom_point() +
  ylab("Cumulative confirmed cases")
```

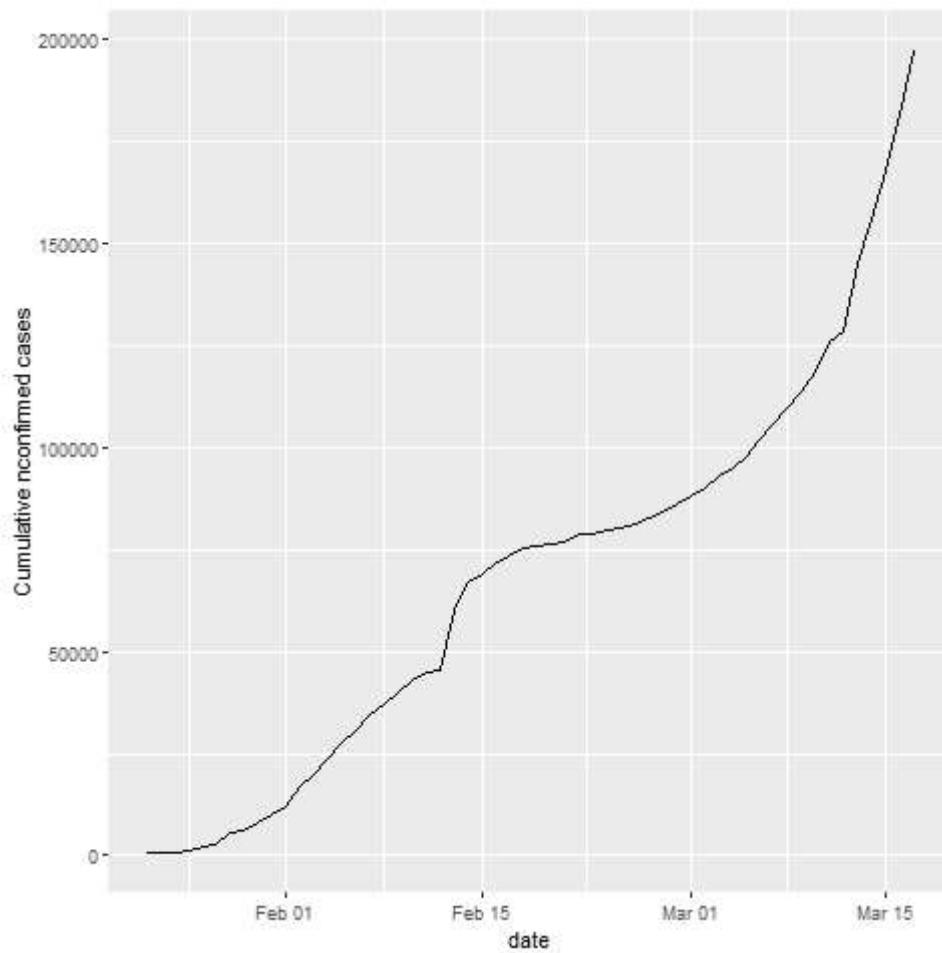


Compare with the DataCamp plot

We're now using the DataCamp data sets.

```
confirmed_cases_worldwide <- read_csv('data/confirmed_cases_worldwide.csv')

## Draw a line plot of cumulative cases vs. date
## Label the y-axis
ggplot(confirmed_cases_worldwide,
       aes(y=cum_cases, x=date)) +
  geom_line() +
  ylab('Cumulative nconfirmed cases')
```



- [] How can you remove the shading under the curve in our plot from the code block 1 above? (Extra credit question!)

Emacs tip

- []

Emacs info: you can change the HTML and screen layout of a plot with meta data - e.g. `#+attr_html: :width 400px` would restrict the width of the following inline image to 400px.

Try that with the last inline image - set the width to 200px.

To open/close inline images, use the key sequence C-c C-x C-v (or the command M-x org-toggle-inline-images).

Further reading

- [ggplot2 line plot : Quick start guide \(2018\)](#)
- [Create a line graph with ggplot \(2020\)](#)

China compared to the rest of the world

- [X]

Load data.

```
## Read in datasets/confirmed_cases_china_vs_world.csv
confirmed_cases_china_vs_world <-
  read_csv('data/confirmed_cases_china_vs_world.csv')
```

Error: 'data/confirmed_cases_china_vs_world.csv' does not exist in current working directory

Glimpse of the data

- [X]

What does dplyr::glimpse do?

Same thing (almost) as str but "it tries to show you as much data as possible" (documentation). Run glimpse on the data frame mtcars and compare with str.

```
glimpse(mtcars)
str(mtcars)
```

```
Rows: 32
Columns: 11
$ mpg <
  21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8, ...
$ cyl <
  6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 4, 4, 4, 4, 8, ...
$ disp <
  160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 140.8, 160.0, ...
$ hp <
  110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, 180, 180, ...
$ drat <
  3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69, 3.92, 3.92, 3.92, ...
$ wt <
  2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3.150, 3.080, ...
$ qsec <
  16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84, 20.00, 22.90, 18.30, ...
$ vs <
  0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, ...
$ am <
  1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, ...
$ gear <
  4, 4, 4, 3, 3, 3, 4, 4, 4, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3, ...
$ carb <
  4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, 1, 1, 2, ...
'data.frame':   32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
```

```
$ am  : num  1 1 1 0 0 0 0 0 0 ...
$ gear: num  4 4 4 3 3 3 3 4 4 4 ...
$ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

- [X]

Glimpse at the Chinese COVID cases. The output is slightly different.

```
glimpse(confirmed_cases_china_vs_world)
```

```
Error in glimpse(confirmed_cases_china_vs_world) :
object 'confirmed_cases_china_vs_world' not found
```

Aesthetics inside a geometry of the plot

- [] What's the effect of putting aes into the geometry instead of into the ggplot call?
- [] Answer: The geometry is responsible for the drawing - putting the aes in a geom function means that we only draw on the data specified in the geom function call.

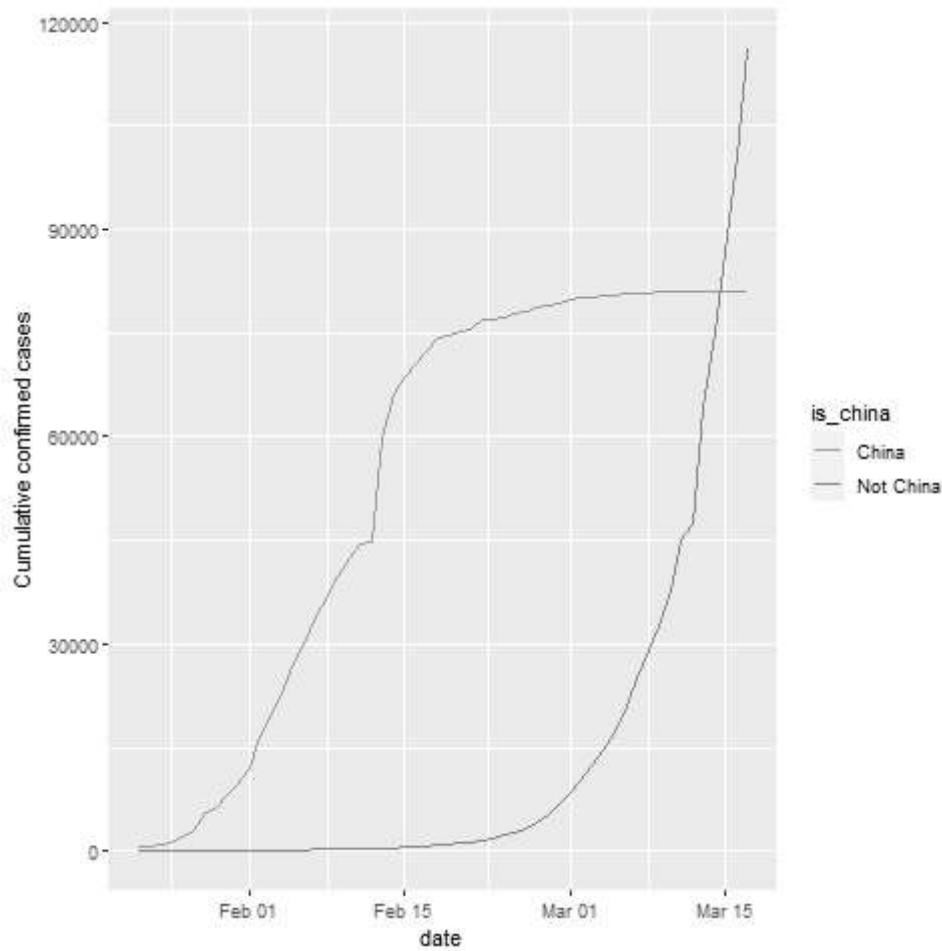
Plot China data

- []

Line plot of China data and store the plot in an object.

```
## Draw a line plot of cumulative cases vs. date, grouped and colored by is_china
## Define aesthetics within the line geom
plt_cum_confirmed_cases_china_vs_world <-
  ggplot(confirmed_cases_china_vs_world) +
  geom_line(aes(x=date, y=cum_cases, group = is_china, color=is_china)) +
  ylab("Cumulative confirmed cases")

## See the plot
plt_cum_confirmed_cases_china_vs_world
```



- The attribute `group` removes the default grouping and splits the data into Chinese and non-Chinese data (using the variable `$Entity` in the original data frame)
- None of these changes are permanent - they are only valid for the creation of the graphical object (better: save data in a separate data structure)
- The attribute `color` colors both categorical variables of the underlying data set
- [X]

What kind of R object is the plot? Check class and structure.

```
class(plt_cum_confirmed_cases_china_vs_world)
str(plt_cum_confirmed_cases_china_vs_world)
```

```
Error: object 'plt_cum_confirmed_cases_china_vs_world' not found
Error in str(plt_cum_confirmed_cases_china_vs_world) :
  object 'plt_cum_confirmed_cases_china_vs_world' not found
```

Let's annotate!

- You've already seen annotation in the test data analysis when we drew vertical lines for the average values.
- We want to annotate by drawing vertical lines over dates and giving them text labels.

Date and text data frame

- [X]

`who_events` is a data frame with two variables. Create this data frame using these data - not as a "tribble" but using the Base R tools that you already know. Call it `df`.

```
"2020-01-30" "Global health\nemergency declared"
"2020-03-11" "Pandemic\ndeclared"
"2020-02-13" "China reporting\nchange"
```

```
df <- data.frame(
  date = c("2020-01-30",
          "2020-03-11",
          "2020-02-13"),
  event = c("Global health\nemergency declared",
            "Pandemic\ndeclared",
            "China reporting\nchange"))
df
```

	date	event
1	2020-01-30	Global health\nemergency declared
2	2020-03-11	Pandemic\ndeclared
3	2020-02-13	China reporting\nchange

- The functions `geom_vline` and `geom_text` are drawn using the data frame and their own attributes. The Base R equivalents are `abline` (drawing a line from a to b, horizontally or vertically), and `text`.

DataCamp plot

- The DataCamp example uses a data frame formatted as a "tribble" and created with the `tribble` function instead of `data.frame`.
- The `dplyr::mutate` function attaches a new variable `date` to the data frame.

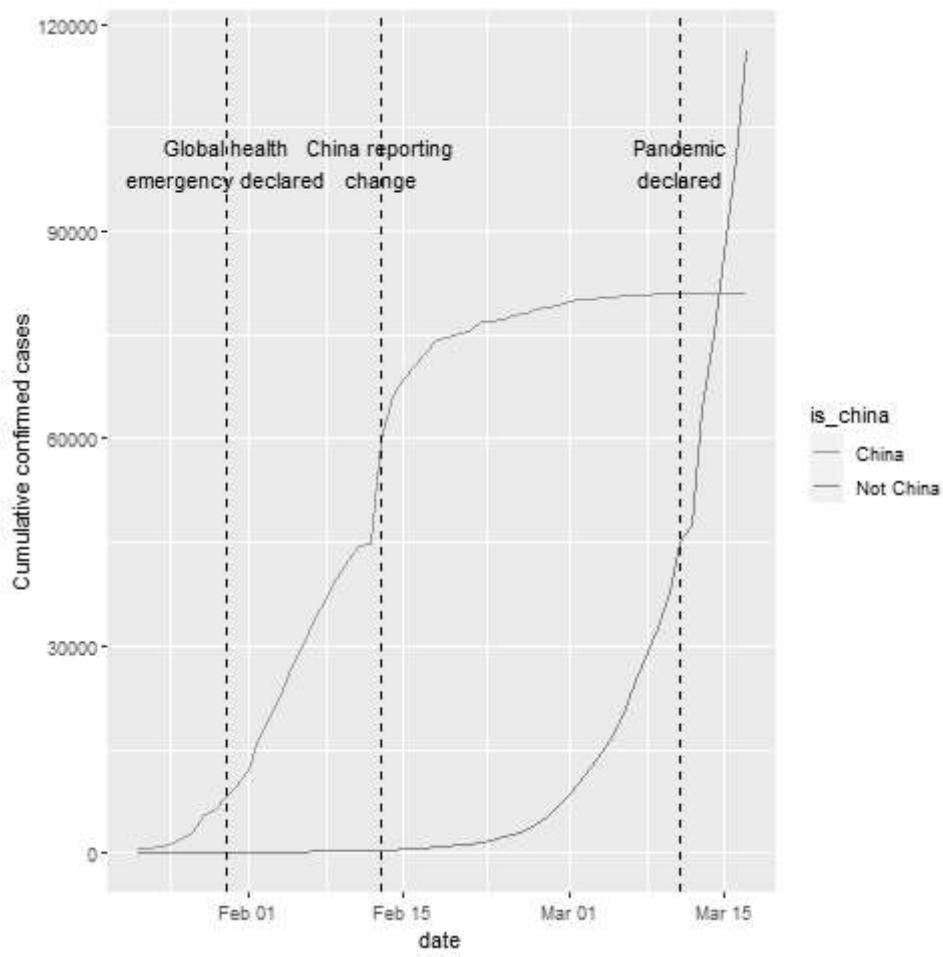
```
who_events <- tribble(
  ~ date, ~ event,
  "2020-01-30", "Global health\nemergency declared",
  "2020-03-11", "Pandemic\ndeclared",
  "2020-02-13", "China reporting\nchange"
) %>%
  mutate(date = as.Date(date))
who_events
```

```
# A tibble: 3 x 2
  date      event
  <date>    <chr>
1 2020-01-30 "Global health\nemergency declared"
2 2020-03-11 "Pandemic\ndeclared"
3 2020-02-13 "China reporting\nchange"
```

- [X]

Annotate the stored plot. I have broken the lines differently than DataCamp to clarify the layers and arguments within each layer.

```
## Using who_events, add vertical dashed lines with an xintercept at date
## and text at date, labeled by event, and at 100000 on the y-axis
plt_cum_confirmed_cases_china_vs_world +
  geom_vline(
    data=who_events,
    aes(xintercept=date),
    linetype="dashed") +
  geom_text(
    data = who_events,
    aes(x=date, label=event),
    y=1e5)
```



Adding a trend line to China

- You can use dplyr::filter to filter data set lines / rows that fulfil a set of logical conditions.

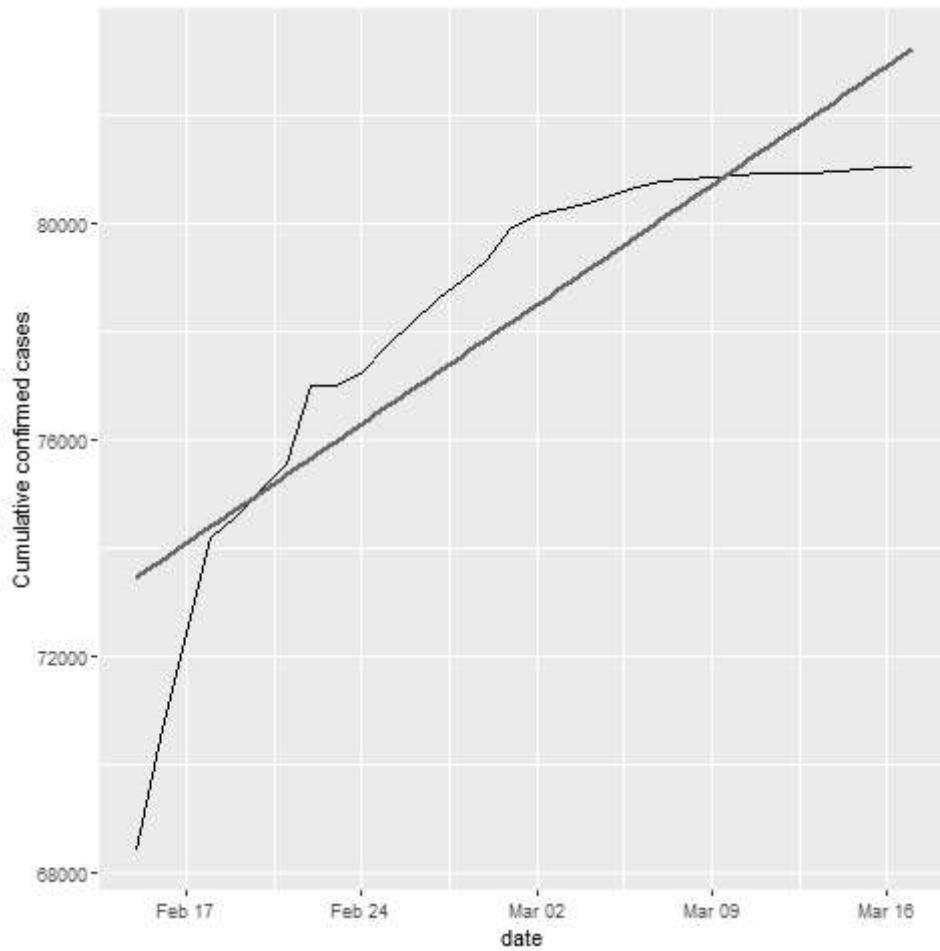
```
## Filter for China, from Feb 15
china_after_feb15 <- confirmed_cases_china_vs_world %>%
  filter(is_china == "China" & date >= '2020-02-15')
```

- `geom_smooth` implements a linear model.

"Calculation is performed by the (currently undocumented) `predictdf` generic and its methods. For most methods the standard error bounds are computed using the `predict()` method - the exceptions are `loess` which uses a t-based approximation, and `glm` where the normal confidence interval is constructed on the link scale, and then back-transformed to the response scale." ([ggplot2 doc](#))

- [X] what does the attribute `se=FALSE` mean? (Answer: [see doc](#))
- For more information and a demo, see `?predict` and run the `glm()` demo with examples.
- Better: separate modeling and creation of pretty plots. Understand functions like `glm()` before blindly applying them (needlessly, in this case - the linear modeling adds nothing to the plot).
- Trend plot with `lm`

```
## Using china_after_feb15, draw a line plot cum_cases vs. date
## Add a smooth trend line using linear regression, no error bars
ggplot(china_after_feb15, aes(x=date, y=cum_cases)) +
  geom_line() +
  geom_smooth(method='lm', se=FALSE) +
  ylab("Cumulative confirmed cases")
```



And the rest of the world?

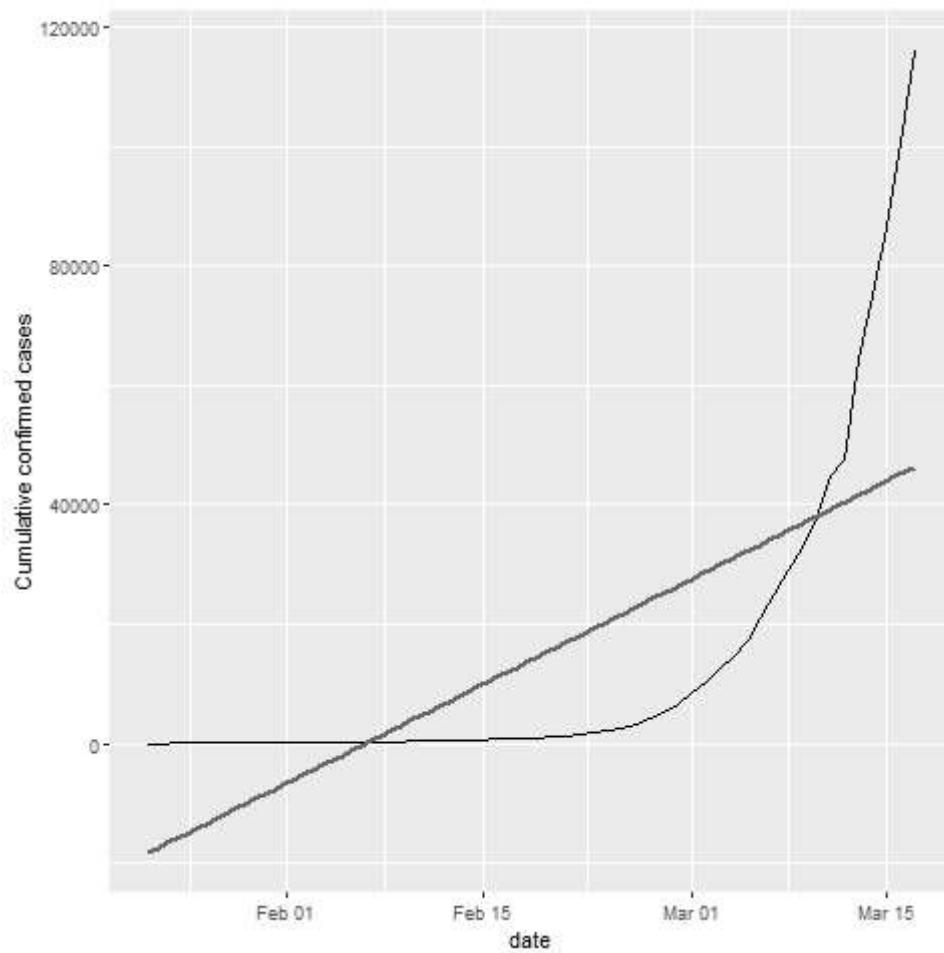
- [X]

Same thing for the rest of the world.

```
## Filter confirmed_cases_china_vs_world for not China
not_china <- confirmed_cases_china_vs_world %>%
  filter(is_china == 'Not China')

## Using not_china, draw a line plot cum_cases vs. date
# Add a smooth trend line using linear regression,
plt_not_china_trend_lin <- ggplot(not_china, aes(x=date, y=cum_cases)) +
  geom_line() +
  geom_smooth(method='lm', se=FALSE) +
  ylab("Cumulative confirmed cases")

## See the result
plt_not_china_trend_lin
```

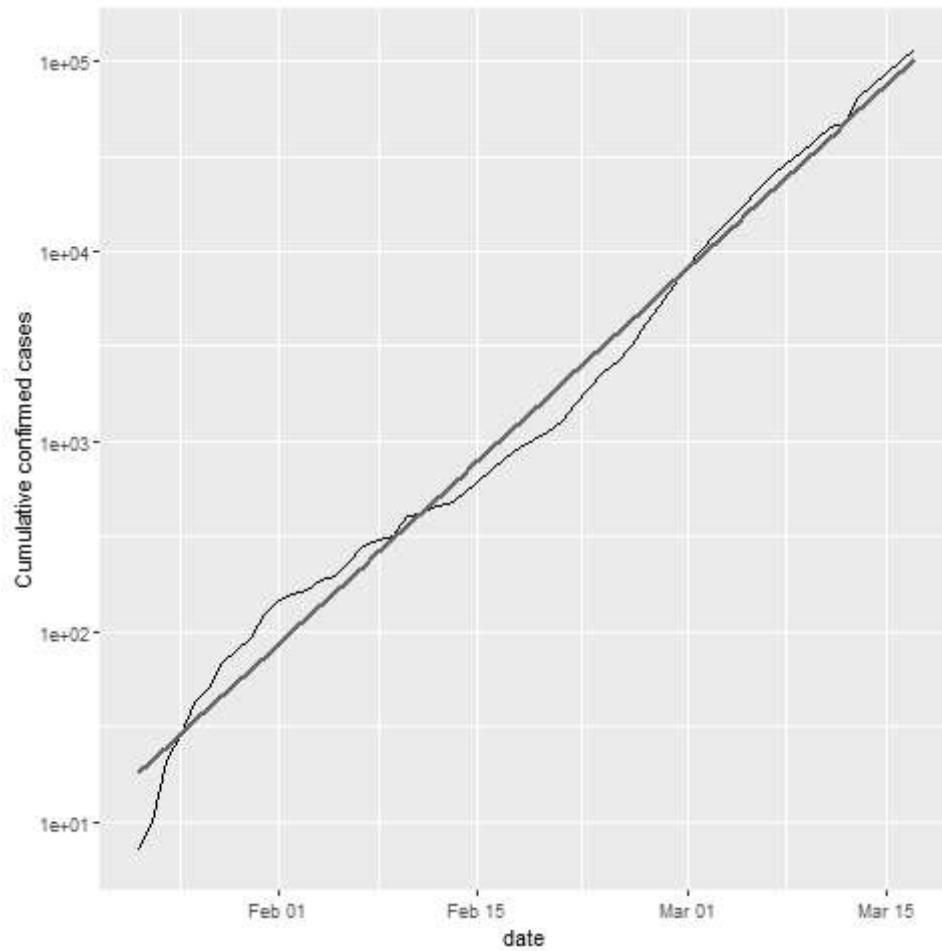


Adding a logarithmic scale

- [] What does the image suggest to not data-literate viewers?
- [] What does it mean "we get a much closer fit of the data?"
- []

What do you think of the graph (in terms of communication)?

```
## Modify the plot to use a logarithmic scale on the y-axis
plt_not_china_trend_lin +
  scale_y_log10()
```



Which countries outside of China have been hit hardest?

- New input data.

```
## Run this to get the data for each country
confirmed_cases_by_country <-
  read_csv("data/confirmed_cases_by_country.csv")
```

- Glimpse at the data

```
glimpse(confirmed_cases_by_country)
```

```
Error: 'data/confirmed_cases_by_country.csv' does not exist in current working directory (
Error in glimpse(confirmed_cases_by_country) :
  object 'confirmed_cases_by_country' not found
```

- [] What does dplyr::group_by do?
- []

What does dplyr::summarize do?

```
## Group by country, summarize to calculate total cases, find the top 7
top_countries_by_total_cases <- confirmed_cases_by_country %>%
  group_by(country) %>%
  summarize(total_cases = max(cum_cases)) %>%
  top_n(7)

## See the result
top_countries_by_total_cases
```

```
Error in group_by(., country) :
  object 'confirmed_cases_by_country' not found
Error: object 'top_countries_by_total_cases' not found
```

Plotting hardest hit countries as of Mid-March 2020

- [] Why do we move the aesthetic mapping back into the main function?
- New input data.

```
## Run this to get the data for the top 7 countries
confirmed_cases_top7_outside_china <-
  read_csv('data/confirmed_cases_top7_outside_china.csv')
```

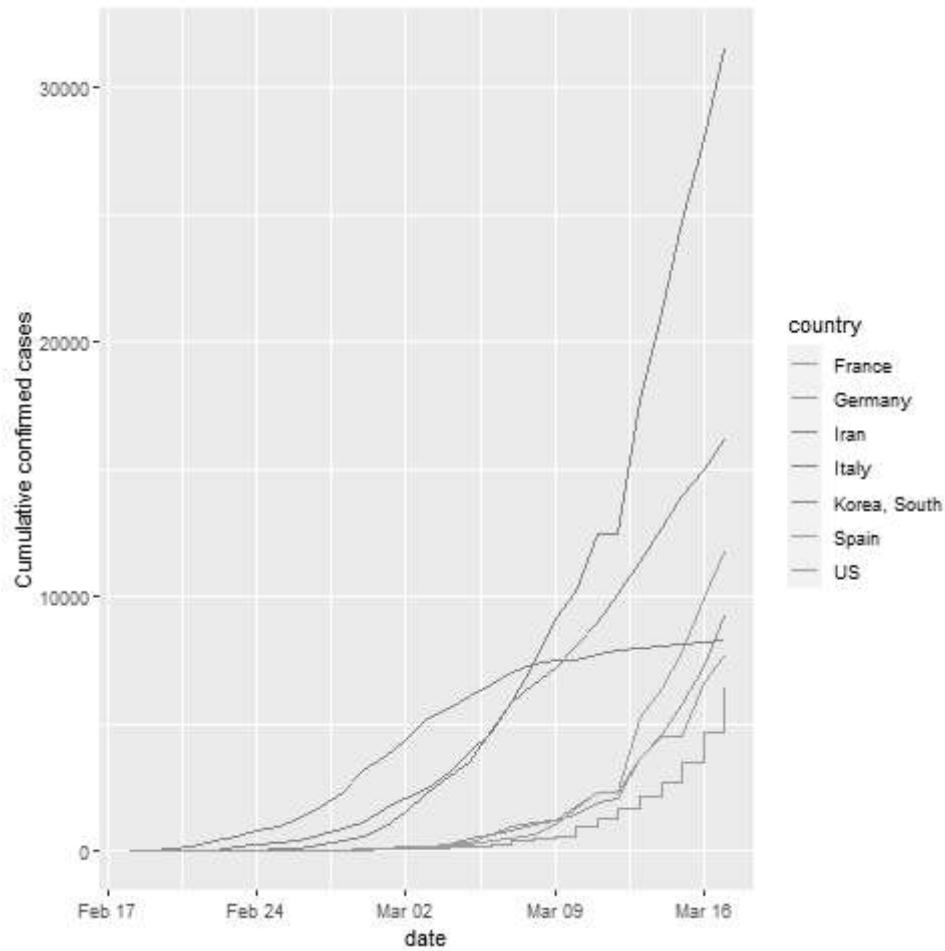
- Glimpse.

```
glimpse(confirmed_cases_top7_outside_china)
```

```
Error: 'data/confirmed_cases_top7_outside_china.csv' does not exist in current working dir
Error in glimpse(confirmed_cases_top7_outside_china) :
  object 'confirmed_cases_top7_outside_china' not found
```

- Plot

```
## Using confirmed_cases_top7_outside_china, draw a line plot of
## cum_cases vs. date, grouped and colored by country
ggplot(confirmed_cases_top7_outside_china, aes(x=date, y=cum_cases, group=country, color=country)) +
  geom_line() +
  ylab('Cumulative confirmed cases')
```



References

- Wickham H, Hester J, Bryan J (2022). `readr`: Read Rectangular Text Data. <https://readr.tidyverse.org>, <https://github.com/tidyverse/readr>.
- R Core Team (2021). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.r-project.org/>.

Author: DataCamp Covid-19 Project / M Birkenkrahe

Created: 2022-03-28 Mon 19:53

[Validate](#)