**Quickstart**

# install the package

install.packages("COVID19")

# load the package

library("COVID19")

# additional packages to replicate the examples

library("ggplot2")

library("directlabels")

**Data**

The data are retrieved with the covid19 function. By default, it downloads worldwide data by country, and prints the corresponding data sources.

x <- covid19()

To hide the data sources use verbose = FALSE

x <- covid19(verbose = FALSE)

A table with several columns is returned: cumulative number of confirmed cases, tests, recovered, deaths, daily number of hospitalized, intensive therapy, patients requiring ventilation, policy measures, geographic information, population, and external identifiers to easily extend the dataset with additional sources.

## [1] "id" "date"

## [3] "tests" "confirmed"

## [5] "recovered" "deaths"

## [7] "hosp" "vent"

## [9] "icu" "population"

## [11] "school\_closing" "workplace\_closing"

## [13] "cancel\_events" "gatherings\_restrictions"

## [15] "transport\_closing" "stay\_home\_restrictions"

## [17] "internal\_movement\_restrictions" "international\_movement\_restrictions"

## [19] "information\_campaigns" "testing\_policy"

## [21] "contact\_tracing" "stringency\_index"

## [23] "iso\_alpha\_3" "iso\_alpha\_2"

## [25] "iso\_numeric" "currency"

## [27] "administrative\_area\_level" "administrative\_area\_level\_1"

## [29] "administrative\_area\_level\_2" "administrative\_area\_level\_3"

## [31] "latitude" "longitude"

## [33] "key\_apple\_mobility" "key\_google\_mobility"

**Clean data**

By default, the raw data are cleaned by filling missing dates with NA values. This ensures that all locations share the same grid of dates and no single day is skipped. Then, NA values are replaced with the previous non-NA value or 0.

*Example*: plot confirmed cases by country.

ggplot(data = x, aes(x = date, y = confirmed)) +

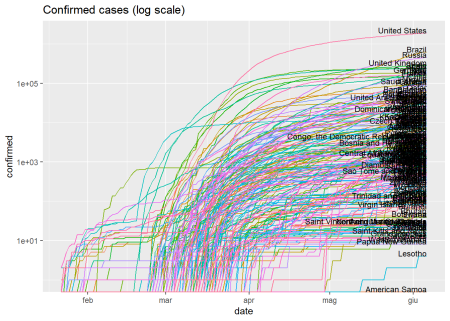
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Confirmed cases (log scale)")



*Example*: plot confirmed cases by country as fraction of total population.

ggplot(data = x, aes(x = date, y = confirmed/population)) +

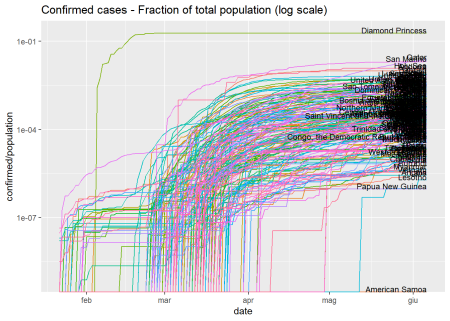
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Confirmed cases - Fraction of total population (log scale)")



**Raw Data**

Filling the data with the previous non-missing data is not always recommended, especially when computing ratios or dealing with more sophisticated analysis other than data visualization. The raw argument allows to skip data cleaning and retrieve the raw data as-is, without any preprocessing.

x <- covid19(raw = TRUE, verbose = FALSE)

The package relies upon publicly available data from multiple sources that do not always agree, e.g. number of confirmed cases greater then number of tests, decreasing cumulative counts, etc. [COVID-19 Data Hub](https://covid19datahub.io/) can spot misalignments between data-sources and automatically inform authorities of possible errors.

*Example*: plot confirmed cases by country as fraction of tests.

ggplot(data = x, aes(x = date, y = confirmed/tests)) +

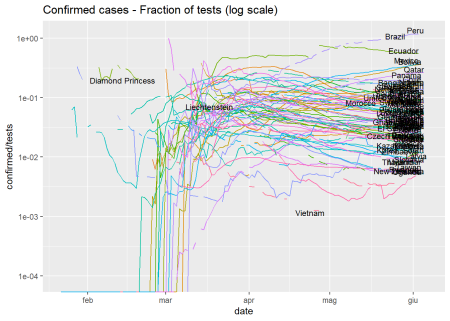
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Confirmed cases - Fraction of tests (log scale)")



*Example*: plot mortality rates by country.

ggplot(data = x, aes(x = date, y = deaths/confirmed)) +

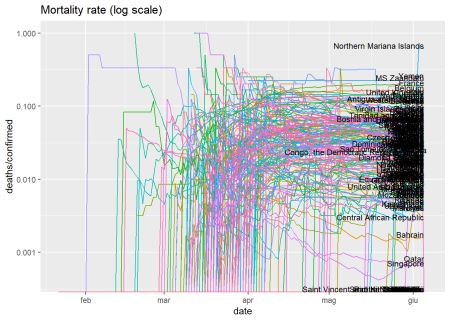
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Mortality rate (log scale)")



**Vintage data**

Retrieve the snapshot of the dataset that was generated at the end date instead of using the latest version. This option ensures reproducibility of the results and keeps track of possible changes made by the data providers. Example: retrieve vintage data on 2020-06-02.

# retrieve vintage data on 2020-06-02

x <- covid19(end = "2020-06-02", vintage = TRUE, verbose = FALSE)

*Example*: compare with the latest data for United Kingdom.

# retrieve latest data

y <- covid19(verbose = FALSE)

# add type

x$type <- "vintage"

y$type <- "latest"

# bind and filter

x <- rbind(x, y)

x <- x[x$iso\_alpha\_3=="GBR",]

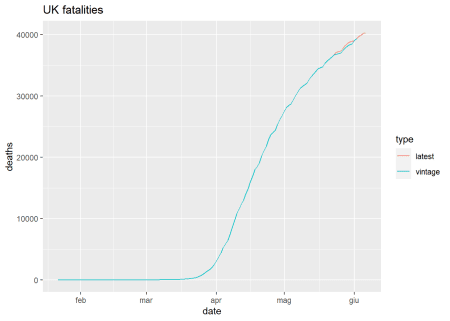
# plot

ggplot(data = x, aes(x = date, y = deaths)) +

geom\_line(aes(color = type)) +

theme(legend.position = "right") +

ggtitle("UK fatalities")



**Administrative areas**

The argument country specifies a vector of case-insensitive country names (alpha-2, alpha-3, numeric) to retrieve.

# load data for United States, Italy, and Switzerland

x <- covid19(c("United States", "ITA", "ch"), verbose = FALSE)

*Example*: plot the mortality rate.

ggplot(data = x, aes(x = date, y = deaths/confirmed)) +

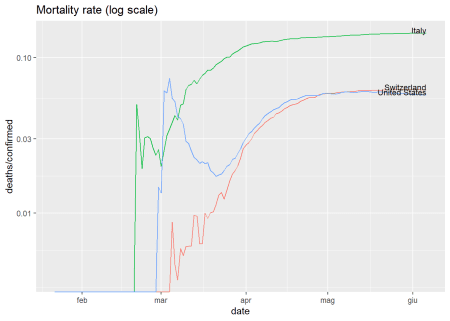
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Mortality rate (log scale)")



The data are available at different levels of granularity: admin area level 1 (administrative area of top level, usually countries), admin area level 2 (usually states, regions, cantons), admin area level 3 (usually cities, municipalities). The granularity of the data is specified by the argument level. *Example*: retrieve data for Italian regions and plot the mortality rate.

# italy admin area level 2

x <- covid19("ITA", level = 2, verbose = FALSE)

# plot

ggplot(data = x, aes(x = date, y = deaths/confirmed)) +

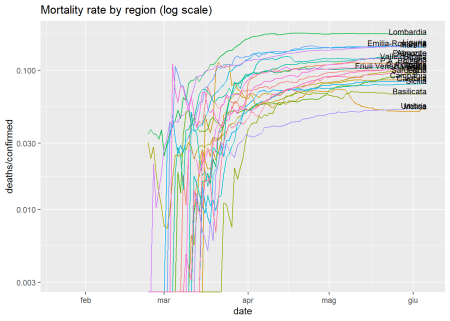
geom\_line(aes(color = id)) +

geom\_dl(aes(label = administrative\_area\_level\_2), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("Mortality rate by region (log scale)")



The package allows for cross-country comparison. *Example*: plot mortality rates for Italian regions and Swiss cantons. Depending on the country, city-level data (level 3) are also supported.

# italy and switzerland admin area level 2

x <- covid19(c("ITA","CHE"), level = 2, verbose = FALSE)

# plot

ggplot(data = x, aes(x = date, y = deaths/confirmed)) +

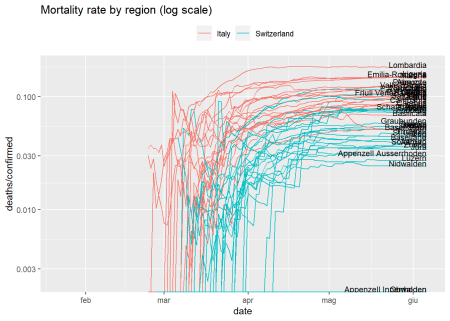
geom\_line(aes(color = administrative\_area\_level\_1, group = administrative\_area\_level\_2)) +

geom\_dl(aes(label = administrative\_area\_level\_2), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "top", legend.title = element\_blank()) +

ggtitle("Mortality rate by region (log scale)")



**Policy measures**

National-level policies are obtained by Oxford Covid-19 Government Response Tracker. Policies for admin areas level 2 and 3 are inherited from national-level policies.

*Example*: load US data, detect changes in the testing policy, and plot them together with the mortality rate.

# US data

x <- covid19("USA", verbose = FALSE)

# detect changes in testing policy

testing\_policy\_dates <- x$date[diff(x$testing\_policy)!=0]

# plot mortality rate and changes in testing policy

ggplot(data = x, aes(x = date, y = deaths/confirmed)) +

geom\_line(aes(color = id)) +

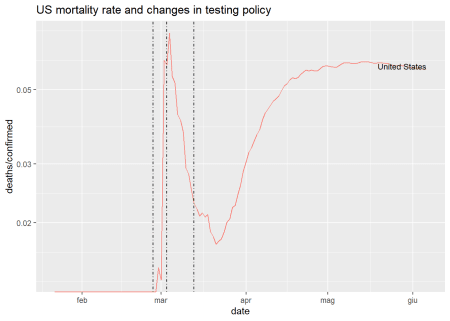
geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

geom\_vline(xintercept = testing\_policy\_dates, linetype = 4) +

scale\_y\_continuous(trans = 'log10') +

theme(legend.position = "none") +

ggtitle("US mortality rate and changes in testing policy")



**World Bank Open Data**

The dataset can be extended with [World Bank Open Data](https://data.worldbank.org/) via the argument wb, a character vector of indicator codes. The codes can be found by inspecting the corresponding URL.

*Example*: extend the dataset with the World Bank indicators [NY.GDP.MKTP.CD](http://NY.GDP.MKTP.CD) (GDP) and SH.MED.BEDS.ZS (hospital beds per 1,000 people).

# download worldwide data + GDP + hospital beds (per 1,000 people)

wb <- c("gdp" = "[NY.GDP.MKTP.CD](http://NY.GDP.MKTP.CD)", "hosp\_beds" = "SH.MED.BEDS.ZS")

x <- covid19(wb = wb, raw = TRUE, verbose = FALSE)

*Example*: plot the mortality rate in function of the number of hospital beds.

ggplot(data = x, aes(x = hosp\_beds, y = deaths/confirmed)) +

geom\_line(aes(color = id)) +

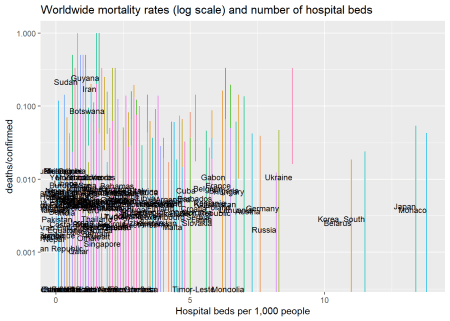
geom\_dl(aes(label = administrative\_area\_level\_1), method = list("last.points", cex = .75, hjust = 1, vjust = 0)) +

scale\_y\_continuous(trans = 'log10') +

xlab("Hospital beds per 1,000 people") +

theme(legend.position = "none") +

ggtitle("Worldwide mortality rates (log scale) and number of hospital beds")



**Google Mobility Reports**

The dataset can be extended with [Google Mobility Reports](https://www.google.com/covid19/mobility/) via the argument gmr, the url to the Google CSV file.

# at the time of writing, the CSV is available at:

gmr <- "<https://www.gstatic.com/covid19/mobility/Global_Mobility_Report.csv>"

x <- covid19("ITA", gmr = gmr, raw = TRUE, verbose = FALSE)

*Example*: detect changes in the Italian transport policy, and plot them together with the mortality rate, percentage of confirmed cases, and Google mobility indicators. Depending on the country, regional or city-level mobility data are also supported.

# detect changes in transport policy

transport\_dates <- x$date[diff(x$transport\_closing)!=0]

# plot

ggplot(x, aes(x = date)) +

geom\_line(aes(y = confirmed/tests\*100, color = "Confirmed/Tested"), size = 1.2) +

geom\_line(aes(y = deaths/confirmed\*100, color = "Deaths/Confirmed"), size = 1.2) +

geom\_line(aes(y = residential\_percent\_change\_from\_baseline, color = "Residential")) +

geom\_line(aes(y = workplaces\_percent\_change\_from\_baseline, color = "Workplaces")) +

geom\_line(aes(y = transit\_stations\_percent\_change\_from\_baseline, color = "Transit Stations")) +

geom\_line(aes(y = parks\_percent\_change\_from\_baseline, color = "Parks")) +

geom\_line(aes(y = grocery\_and\_pharmacy\_percent\_change\_from\_baseline, color = "Grocery and Pharmacy")) +

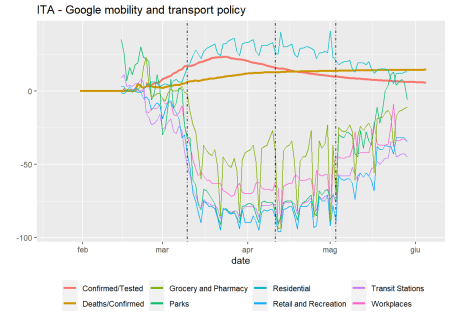
geom\_line(aes(y = retail\_and\_recreation\_percent\_change\_from\_baseline, color = "Retail and Recreation")) +

geom\_vline(xintercept = transport\_dates, linetype = 4) +

ylab("") +

theme(legend.position = "bottom", legend.title = element\_blank()) +

ggtitle("ITA - Google mobility and transport policy")



**Apple Mobility Reports**

The dataset can be extended with [Apple Mobility Reports](https://www.apple.com/covid19/mobility) via the argument amr, the url to the Apple CSV file. Depending on the country, regional or city-level mobility data are also supported.

# at the time of writing, the CSV is available at:

amr <- "<https://covid19-static.cdn-apple.com/covid19-mobility-data/2009HotfixDev19/v3/en-us/applemobilitytrends-2020-06-03.csv>"

x <- covid19("ITA", amr = amr, raw = TRUE, verbose = FALSE)

*Example*: detect changes in the Italian transport policy, and plot them together with the mortality rate, percentage of confirmed cases, and Apple mobility indicators.

# detect changes in transport policy

transport\_dates <- x$date[diff(x$transport\_closing)!=0]

# plot

ggplot(x, aes(x = date)) +

geom\_line(aes(y = confirmed/tests\*100, color = "Confirmed/Tested"), size = 1.2) +

geom\_line(aes(y = deaths/confirmed\*100, color = "Deaths/Confirmed"), size = 1.2) +

geom\_line(aes(y = driving-100, color = "Driving")) +

geom\_line(aes(y = walking-100, color = "Walking")) +

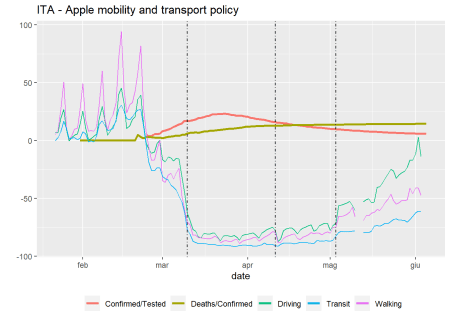
geom\_line(aes(y = transit-100, color = "Transit")) +

geom\_vline(xintercept = transport\_dates, linetype = 4) +

ylab("") +

theme(legend.position = "bottom", legend.title = element\_blank()) +

ggtitle("ITA - Apple mobility and transport policy")



**Analysis at scale**

The structure of the dataset makes it easy to replicate the anlysis for multiple countries, states, and cities, by using the dplyr package.

library("dplyr")

Load the data.

x <- covid19(raw = TRUE, verbose = FALSE)

Define the function to apply to each group (e.g. country), must return a data frame. For example, the function could compute the \(R\_0\) for each country or region. For the sake of simplicity, the following example is limited to extract the latest number of fatalities for each group.

f <- function(x, key){

# x: the subset of rows for the given group

# key: row with one column per grouping variable that identifies the group

# code here...

example <- tail(x$deaths, 1)

# return data.frame

data.frame(id = key$id, example = example)

}

Define the groups, apply the function to each group, and bind the results.

y <- x %>%

group\_by(id) %>% # group by location id

group\_map(f) %>% # apply the function to each group

bind\_rows() # bind the results

Print the first rows.

head(y, 5)

## id example

## 1 AFG 309

## 2 AGO 4

## 3 ALB 33

## 4 AND 51

## 5 ARE 274