Package ‘corona’

October 12, 2022

Title Coronavirus ('Rona') Data Exploration
Version 0.3.0
Depends R (>= 3.5.0), plyr
Imports gganimate, ggplot2, gridExtra, qicharts2, reshape2
Maintainer Jo van Schalkwyk <jvanschalkwyk@gmail.com>
Description Manipulate and view coronavirus data and other societally relevant data at a basic level.
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.1.0
NeedsCompilation no
Author Jo van Schalkwyk [aut, cre] (<https://orcid.org/0000-0002-0082-5243>)
Repository CRAN
Date/Publication 2020-09-23 09:30:03 UTC

R topics documented:

  allo ................................................................. 2
  citymap ............................................................. 3
  entry ................................................................. 4
  corona ............................................................... 5
  corona_all ......................................................... 5
  corona_citymap ................................................... 6
  corona_converge .................................................. 6
  corona_country .................................................... 7
  corona_dowjones .................................................. 8
  corona_life ........................................................ 8
  corona_lockdown .................................................. 9
  corona_metabolism ............................................... 10
  corona_monty ..................................................... 11
  corona_rabbits ................................................... 11
allo

Allometric scaling data.

Description

Used to introduce power laws.

Usage

allo

Format

A data frame with 455 rows.

Species

Mass

Temperature

MR Metabolic rate

AvgMass

Q10SMR

Reference

Source

citymap

Citymapper data.

Description

These are a bit unusual in that each country has a column.

Usage
citymap

Format

A data frame with 108 rows.

Date
Australia
Austria
Belgium
Brazil
Canada
Denmark
France
Germany
Italy
Japan
Mexico
Netherlands
Portugal
Russia
Singapore
South.Korea
Spain
Sweden
Turkey
United.Kingdom
United.States

Source

https://citymapper.com/cmi/about
Country data from Our World In Data.

Description
Country data from Our World In Data.

Usage
cntry

Format
A data frame with 17,013 rows (current)

iso_code   ISO 3-letter country code
location   Text name of country
population
continent
population_density
median_age
aged_65_older
aged_70_older
gdp_per_capita
extreme_poverty
cvd_death_rate
diabetes_prevalence
female_smokers
male_smokers
handwashing_facilities
hospital_beds_per_thousand
life_expectancy
alias      Alias country name, shorter
lowstart   Start of `summer` viral respiratory low
lowend     End of respiratory low. Sketchy at present.

Source
Description

Try `?corona` for help. For most functions, saying `pdf=TRUE` will write a PDF to `images/`. If you wish to print to PDF, you need to `setwd()` to a directory that contains an `images/` directory that can be written to, or this will fail. Individual examples are also available. Try e.g. `?corona_rabbits` or `?corona_country` The results of `corona_life()` will depend on how your system handles animated GIF files.

Usage

corona()

Examples

corona_rabbits ()
corona_monty ()
corona_country ('France')
corona_vienna ()
corona_totals ()
country_dead ()
corona_converge ()
corona_metabolism ()
corona_citymap ()
corona_dowjones ()

corona_all

Generate all Figures

Description

For the book `Rona` (printing to PDF) work through and generate PDFs for all examples.

Usage

corona_all()
corona_citymap  
*Plot citymapper data against COVID-19 diagnoses, over time*

**Description**

Requires ggplot2, plyr and the data frames lock, owid, citymap. Multiple, select frames are plotted.

**Usage**

```r
corona_citymap(pdf = FALSE, FewCities = NULL, cols = 4)
```

**Arguments**

- `pdf` = TRUE writes to PDF, default FALSE
- `FewCities` a c() list of city names from the city options. Default is all.
- `cols` Number of columns in output, default is 4

**Examples**

```r
corona_citymap(cols=4);
```

corona_converge  
*Create various statistical distributions*

**Description**

Build a normal or log-normal distribution from simple components. Large numbers e.g. n=1e6 will take some time to run.

**Usage**

```r
corona_converge(
  n = 1e+05,
  method = "add",
  runs = 7,
  pdf = FALSE,
  xscale = 1,
  bins = 64,
  log = FALSE
)
```
Arguments

- **n**: is the number of samples
- **method**: is either 'multiply' or 'add'
- **runs**: number of iterations (default 7)
- **pdf**: defaults to FALSE
- **xscale**: a scaling factor, can use values < 1.0 to magnify (x) e.g. 0.4
- **bins**: defaults to 64
- **log**: take logarithm of values (for 'multiply')

Examples

```r
corona_converge( n=10000, method='multiply', xscale=0.4, bins=128, runs=5 )
```

Description

The daily case rate is also shown as a smoothed curve. The smoothed death incidence is MULTI-
PLIED x5 to highlight its relationship to the incidence curve. See grown-up documentation (LyX)

Usage

```r
corona_country(country, pdf = FALSE, smooth = TRUE, deaths = TRUE)
```

Arguments

- **country**: no default
- **pdf**: defaults to FALSE. If TRUE, writes to country_name_new.pdf i.e. 'new.pdf' is appended to formal country name. If the country name contains spaces ’ ’ they are changed to underscores ”
- **smooth**: default TRUE show smoothed (red) curve
- **deaths**: default TRUE show deaths

Examples

```r
corona_country('United States');
corona_country('Taiwan');
```
corona_dowjones  

**Plot Dow-Jones Closing data**

**Description**
Assumes the existence of the data frame djia, part of corona data.

**Usage**
corona_dowjones(pdf = FALSE)

**Arguments**

- **pdf** : will not print to PDF

**Examples**
corona_dowjones()

---

**corona_life**  

**Animate Conway's Game of Life**

**Description**
The canvas (arena) wraps around vertically and horizontally! Execution will take some time. Results will be viewed differently depending on your system's default viewer for animated GIF files.

**Usage**
corona_life(
    pattern = "soup",
    side = 50,
    steps = 100,
    density = 0.3,
    filename = NULL,
    wrap = TRUE,
    fps = 20,
    pause = 10
)
corona_lockdown

Arguments

- **pattern**: Defaults to 'soup' but there are many other well-known options: blinker tetromino tetromino toad bee hive beacon clock pulsar pentadecathlon galaxy spaceship glider gun pheptomino switchengine conway acorn rabbits boring static patterns: block snake eater
- **side**: The number of elements on the area’s side (width or height)
- **steps**: The number of frames
- **density**: 0.0–1 The density of the initial, random items ('soup')
- **filename**: writes to this file name e.g. foo.gif (NULL for current GIF device)
- **wrap**: Wrap around
- **fps**: Frames per second
- **pause**: Initial pause

Examples

```r
## Not run:
corona_life( filename='animation.gif', side=50, steps=500, density=0.2 )
corona_life( side=100, steps=1000, pattern='tetromino', wrap=FALSE )
corona_life( side=30, steps=120, pattern='spaceship' )
corona_life( side=100, steps=400, pattern='switchengine' )
corona_life( side=20, steps=30, pattern='clock' )
corona_life( side=20, steps=30, pattern='galaxy' )
corona_life( side=100, steps=200, pattern='glidergun' )
corona_life( side=45, steps=130, pattern='conway', fps=8, pause=40 )
## End(Not run)
```

corona_lockdown  

*Draw multiple smoothed graphs of new daily cases, with lockdown date, if present*

Description

By default limited to countries with population > 4M, and over 200 cases. This may take over 5s to run, depending on your hardware.

Usage

```r
corona_lockdown(
  pdf = FALSE,
  minpeople = 4e+06,
  mincases = 200,
  cols = 7,
  striptextsize = 10,
  textsize = 10,
  legendx = 0.94,
  legendy = 0.02
)
```
Arguments

- **pdf**: print to PDF
- **minpeople**: Minimum population for the country
- **mincases**: Minimum number of COVID-19 cases
- **cols**: Number of columns to display, default = 7
- **stripedtextsize**: size of text in country names
- **textsize**: Size of text header
- **legendx**: X position of legend
- **legendy**: Y position of legend

Examples

```r
## Not run:
corona_lockdown( cols=14 )
## End(Not run)
```

---

corona_metabolism  
*Allometric scaling of metabolic rates*

Description

Log-log plot of mammalian weights (grams) against metabolic rates. The PDF file is allometry.pdf.

Usage

```r
corona_metabolism(pdf = FALSE, base = 10)
```

Arguments

- **pdf**: will not print to PDF
- **base**: base for logarithms, default 10

Examples

```r
corona_metabolism() 
```
corona_monty

Description
A Monte Carlo simulation of the Monty Hall problem

Usage
corona_monty(runs = 100)

Arguments
- runs: specifies the number of parallel simulations, default=100.

Examples
corona_monty(runs=10000)

corona_rabbits
Demonstrate (graph) exponential growth of rabbit population:

Description
For finer details, see the LyX/PDF documentation.

Usage
corona_rabbits(topyear = 6, pdf = FALSE)

Arguments
- topyear: is last year, defaults to 6
- pdf: Will not print to PDF if FALSE (the default)

Examples
corona_rabbits(topyear=10)
corona_totals

Plot total cases over time for a selected country.

Description
Defaults to Italy, as this was our demonstration. Add a linear regression by specifying smooth=TRUE.

Usage

```r
corona_totals(
  country = "Italy",
  daystart = 60,
  dayend = 76,
  pdf = FALSE,
  log = FALSE,
  smooth = FALSE,
  prefix = ""
)
```

Arguments

- **country**: Text name of country (in owid frame)
- **daystart**: first day
- **dayend**: last day to plot
- **pdf**: TRUE will print value
- **log**: TRUE will take base 10 logarithm of y-axis values
- **smooth**: TRUE will try to fit linear model (use with logarithm)
- **prefix**: defaults to ""; a text value will be prefixed to PDF name after country_. name.

Examples

```r
corona_totals( country='Italy', daystart=60, dayend=76, log=TRUE, smooth=TRUE )
corona_totals( country='United Kingdom', log=TRUE, smooth=TRUE )
```

corona_trends

Plot Google Trends data for searches involving the word 'coronavirus'.

Description

Just plot the lines.

Usage

```r
corona_trends(pdf = FALSE)
```
corona_vienna

Arguments

pdf default FALSE will not print the PDF file

Examples

corona_trends()

corona_vienna

Plot Semmelweis’ original data from Vienna.

Description
First simply 'plots the dots'; subsequently draws a run chart with a transition at the point where he instituted hand-washing.

Usage

corona_vienna(pdf = FALSE)

Arguments

pdf default FALSE will not print the two PDF files: semmelweis_plot.pdf semmelweis_run.pdf

Examples

corona_vienna()

country_dead

Plot country deaths by week, with various adjustments:

Description
Assumes the existence of the data frame stmf containing relevant iso_codes for countries. The unusual codes GBRTENW and GBR_SCO represent England+Wales and Scotland. You can obtain a list of countries by country_dead('?'), forcing a diagnostic error!

Usage

country_dead(country = "England+Wales", pdf = FALSE, save = FALSE)

Arguments

country Country name
pdf default FALSE will not print to PDF
save Do we save the data as a CSV
Details

The columns in the frame stmf are just 'iso_code', 'Year', 'Week', and 'Deaths'.

Draws three graphs:

1. Raw data with a linear regression line, over n years;
2. Data with secular adjustment;
3. Data adjusted for a 'summer baseline' using the "other n years of data" after secular adjustment.

Examples

country_dead('New Zealand')

djia

Historical Dow Jones Industrial Average prices.

Description

Historical Dow Jones Industrial Average prices.

Usage

djia

Format

A data frame with 110 rows (current)

Date  Date of transaction—excludes weekends etc
Open  Opening average
High  Maximum over the day
Low   Minimum
Close Closing price

Source

https://www.wsj.com/market-data/quotes/index/DJIA/historical-prices
**gt**

*Google trends search for 'coronavirus'.*

**Description**

Google trends search for 'coronavirus'.

**Usage**

`gt`

**Format**

A data frame with 155 rows (current)

- **Date**  Date in format YYYY-MM-DD
- **Day** Coronavirus 'interest' as percentage of maximum count

**Source**

[https://trends.google.com/trends/](https://trends.google.com/trends/)

---

**life**

*The game of life.*

**Description**

This specifies initial conditions, using a clumsy storage format as below.

**Usage**

`life`

**Format**

A data frame with 213 rows.

- **x**  x co-ordinate of an active cell
- **y**  y co-ordinate

- **pattern** A name like 'blinker' — will be common to several rows, specifying a Game of Life pattern

**Source**

(internal generation)
**Description**

Approximate dates of full lockdown in various countries.

**Usage**

lock

**Format**

A data frame with 110 rows (current)

- **iso_code** Country
- **Lockdown** Date of lockdown YYYY-MM-DD
- **nature** Text description: national | partial | advice | empty (none)

**Source**

Various data sources.

---

**owid**

Wide-ranging data from Our World In Data. I only use a tiny part.

**Description**

Wide-ranging data from Our World In Data. I only use a tiny part.

**Usage**

owid

**Format**

A data frame with 17,013 rows (current)

- **iso_code** ISO 3-letter country code
- **date** Date for this row of data
- **total_cases** total cases to date
- **new_cases** new cases
- **total_deaths** eponymous
- **new_deaths**
total_tests  Recorded tests in toto
new_tests  Eponymous
tests_units
stringency_index  How severe the lockdown was

Source


stmf  Deaths, by week, for various countries.

Description

Deaths, by week, for various countries.

Usage

stmf

Format

A data frame with 22678 rows.

iso_code  Normally a 3-character country code e.g. NZL, AUT. England+Wales=GBR_TENW, Scotland=GBR_SCO
Year  YYYY
Week  Week within that year, 1=1st
Deaths  Number of deaths in that week
X

Source

https://www.ons.gov.uk/peoplepopulationandcommunity/birthsdeathsandmarriages/deaths/datasets/weeklyprovisionalfiguresondeathsregisteredinenglandandwales
Semmelweis’ data on Deaths of parturients in Vienna

Description
Semmelweis’ data on Deaths of parturients in Vienna

Usage
vienna

Format
A data frame with 98 rows

- **date**: Date of the start of each month YYYY-MM-01
- **births**: Number of births during that month
- **deaths**: Number of maternal deaths during that month

Source
Index

* Benford
  * Benford_converge, 6
* Carlo
  * Carlo_monty, 11
* Conway
  * Conway_life, 8
* Figures
  * Figures_all, 5
* Google
  * Google_trends, 12
* Hall
  * Hall_monty, 11
* Kleiber
  * Kleiber_metabolism, 10
* Monte
  * Monte_monty, 11
* Monty
  * Monty_monty, 11
* Nanny
  * Nanny_corona, 5
  * Nanny_all, 5
* PDF
  * PDF_all, 5
* Rona
  * Rona_corona, 5
  * Rona_all, 5
* Semmelweis
  * Semmelweis_viena, 13
* Trends
  * Trends_trends, 12
* Vienna
  * Vienna_viena, 13
* allometric
  * allometric_metabolism, 10
* allometry
  * allometry_metabolism, 10
* animation
  * animation_life, 8
* average
  * average_dowjones, 8
* book
  * book_all, 5
* cases
  * cases_totals, 12
* central
  * central_converge, 6
* citymapper
  * citymapper_citymap, 6
* citymap
  * citymap_citymap, 6
* coronavirus
  * coronavirus_monty, 11
  * coronavirus_trends, 12
* corona
  * corona_corona, 5
  * corona_all, 5
  * corona_citymap, 6
  * corona_converge, 6
  * corona_country, 7
  * corona_dowjones, 8
  * corona_life, 8
  * corona_lockdown, 9
  * corona_metabolism, 10
  * corona_monty, 11
  * corona_rabbits, 11
  * corona_totals, 12
  * corona_trends, 12
  * corona_viena, 13
  * country_dead, 13
* countries
  * countries_citymap, 6
* country
  * country_country, 7
  * country_totals, 12
* daily
  * daily_citymap, 6
* datasets
  * datasets_allo, 2
citymap, 3
cntry, 4
djia, 14
gt, 15
life, 15
lock, 16
owid, 16
stmf, 17
vienna, 18

* deaths
  country_dead, 13
* dow
  corona_dowjones, 8
* exponent
  corona_metabolism, 10
* frames
  corona_life, 8
* game
  corona_life, 8
* industrial
  corona_dowjones, 8
* jones
  corona_dowjones, 8
* law
  corona_converge, 6
* life
  corona_life, 8
* limit
  corona_converge, 6
* lockdown
  corona_lockdown, 9
* log-normal
  corona_converge, 6
* lognormal
  corona_converge, 6
* normal
  corona_converge, 6
* of
  corona_life, 8
* print
  corona_all, 5
* quarters
  corona_metabolism, 10
* rabbits
  corona_rabbits, 11
* rates
  corona_citymap, 6
* scaling
  corona_metabolism, 10
* simulation
  corona_monty, 11
* single
  corona_country, 7
* smoothed
  corona_lockdown, 9
* theorem
  corona_converge, 6
* thirds
  corona_metabolism, 10
* three
  corona_metabolism, 10
* total
  corona_totals, 12
* two
  corona_metabolism, 10

allo, 2
citymap, 3
cntry, 4
corona, 5
corona_all, 5
corona_citymap, 6
corona_converge, 6
corona_country, 7
corona_dowjones, 8
corona_life, 8
corona_lockdown, 9
corona_metabolism, 10
corona_monty, 11
corona_rabbits, 11
corona_totals, 12
corona_trends, 12
corona_vienna, 13
country_dead, 13
djia, 14
gt, 15
life, 15
lock, 16
owid, 16
stmf, 17
vienna, 18