

Package ‘covid19.analytics’

July 22, 2025

Type Package

Title Load and Analyze Live Data from the COVID-19 Pandemic

Version 2.1.3.3

Date 2023-10-14

Author Marcelo Ponce [aut, cre], Amit Sandhel [ctb]

Maintainer Marcelo Ponce <m.ponce@utoronto.ca>

Description

Load and analyze updated time series worldwide data of reported cases for the Novel Coronavirus Disease (COVID-19) from different sources, including the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE) data repository <<https://github.com/CSSEGISandData/COVID-19>>, ``Our World in Data" <<https://github.com/owid/>> among several others. The datasets reporting the COVID-19 cases are available in two main modalities, as a time series sequences and aggregated data for the last day with greater spatial resolution. Several analysis, visualization and modelling functions are available in the package that will allow the user to compute and visualize total number of cases, total number of changes and growth rate globally or for an specific geographical location, while at the same time generating models using these trends; generate interactive visualizations and generate Susceptible-Infected-Recovered (SIR) model for the disease spread.

Imports readxl, ape, rentrez, curl, plotly, htmlwidgets, deSolve, gplots, pheatmap, shiny, shinydashboard, shinycssloaders, DT, dplyr, collapsibleTree

Suggests knitr, devtools, roxygen2, markdown, rmarkdown, testthat

License GPL (>= 2)

URL <https://mponce0.github.io/covid19.analytics/>

BugReports <https://github.com/mponce0/covid19.analytics/issues>

RoxygenNote 7.1.0

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2023-10-15 21:50:11 UTC

Contents

c19.fasta.data	3
c19.genomic.data	3
c19.NPs.data	4
c19.NP_fasta.data	4
c19.OWID.data	5
c19.ptree.data	5
c19.refGenome.data	6
consistency.check	6
covid19.Canada.data	7
covid19.data	8
covid19.genomic.data	9
covid19.JHU.data	9
covid19.testing.data	10
covid19.Toronto.data	11
covid19.Toronto_city.data	12
covid19.Toronto_OD.data	13
covid19.URL_csv.data	13
covid19.US.data	14
covid19.vaccination	15
covid19dashboard	15
covid19Explorer	16
data.checks	16
estimateRRs	17
generate.SIR.model	17
geographicalRegions	19
getFile	19
growth.rate	20
integrity.check	21
itrends	21
live.map	22
movingFn	23
mrollingRates	24
mtrends	24
nullify.data	25
OWID.repos	25
pandemics.data	26
pandemics.loaddata	26
plt.SIR.model	27
preProcessingData	27
red.devel.ver	28
report.summary	28
report.Tor	29
rollingRate	30
single.trend	30
sweep.SIR.models	31
totals.plt	32

<i>c19.fasta.data</i>	3
tots.per.location	33
X.covid19.genomic.data	34
Index	35

<i>c19.fasta.data</i>	<i>function to obtain FASTA sequence of the SARS-CoV-2 virus</i>
-----------------------	--

Description

function to obtain FASTA sequence of the SARS-CoV-2 virus

Usage

```
c19.fasta.data(src = "livedata")
```

Arguments

src	argument to indicate where the data is being retrieved from
-----	---

<i>c19.genomic.data</i>	<i>function to obtain genomic data from SARS-CoV-2019</i>
-------------------------	---

Description

function to obtain genomic data from SARS-CoV-2019

Usage

```
c19.genomic.data(src = "livedata", accOnly = TRUE)
```

Arguments

src	argument to indicate what sources are going to be used for retrieving the data: "livedata", "repo" or "local" 'livedata' will access NCBI servers to acquire the latest possible data, this may incur in significant longer times 'repo' will access an updated replica of the data from a github repository (faster but not necessarily upto the latest updates) 'local' will access previously archived records within teh package (fastest but not updated)
accOnly	boolean indicator for getting only accession codes or whole records

Value

a list containing reference genome, annotation data, nucleotides, proteins and list of SRA runs

c19.NPs.data	<i>function to obtain data for nucleotides or proteins from SARS-CoV-2</i>
--------------	--

Description

function to obtain data for nucleotides or proteins from SARS-CoV-2

Usage

```
c19.NPs.data(
  src = "livedata",
  DB = "nucleotide",
  max.nr.recs = NULL,
  accOnly = TRUE
)
```

Arguments

src	origin for the data source: "livedata", "repo", "local"
DB	database
max.nr.recs	maximun number of records to retrieve, there are limitations in the fns and server sides
accOnly	boolean indicator for getting only accession codes or whole records

c19.NP_fasta.data	<i>function to obtain FASTA seqs for nucleotides or proteins from SARS-CoV-2</i>
-------------------	--

Description

function to obtain FASTA seqs for nucleotides or proteins from SARS-CoV-2

Usage

```
c19.NP_fasta.data(src = "repo", target = "nucleotide")
```

Arguments

src	origin for the data source: "livedata" OR "repo"
target	"nucleotide", "protein" or "codingRegion"

c19.OWID.data	<i>function to read data from OWID repos</i>
---------------	--

Description

function to read data from OWID repos

Usage

```
c19.OWID.data(repo, disclaimer = TRUE)
```

Arguments

repo	URL for reading the data
disclaimer	indicate whether the information about the source of the data is disclosed

Value

datafram object

c19.ptree.data	<i>function to obtain "Tree of complete SARS-CoV-2 Sequences as obtained from NCBI"</i>
----------------	---

Description

function to obtain "Tree of complete SARS-CoV-2 Sequences as obtained from NCBI"

Usage

```
c19.ptree.data(src = "livedata")
```

Arguments

src	argument to indicate where the data is being retrieved from
-----	---

c19.refGenome.data	<i>function to obtain sequencing data from NCBI Reference: https://www.ncbi.nlm.nih.gov/nuccore/NC_045512.2</i>
--------------------	--

Description

function to obtain sequencing data from NCBI Reference: https://www.ncbi.nlm.nih.gov/nuccore/NC_045512.2

Usage

```
c19.refGenome.data(src = "livedata", graphics.ON = TRUE)
```

Arguments

src	data origin source: 'livedata', 'repo', 'local'
graphics.ON	flag to activate/deactivate graphical output

Examples

```
## Not run:
# obtain covid19's genomic data
covid19.gen.seq <- c19.refGenome.data()
# display the actual RNA seq
covid19.gen.seq$NC_045512.2

## End(Not run)
```

consistency.check	<i>function that determines whether there are consistency issues within the data, such as, anomalies in the cumulative quantities of the data as reported by JHU/CCSEGIS</i>
-------------------	--

Description

function that determines whether there are consistency issues within the data, such as, anomalies in the cumulative quantities of the data as reported by JHU/CCSEGIS

Usage

```
consistency.check(
  data,
  n0 = 5,
  nf = ncol(data),
  datasetName = "",
  disclose = FALSE,
  details = TRUE
)
```

Arguments

<code>data</code>	dataset to analyze
<code>n0</code>	column where the cumulative data begins
<code>nf</code>	column where the cumulative data ends
<code>datasetName</code>	optional argument to display the name of the dataset
<code>disclose</code>	boolean flag to indicate whether index of problematic entries are returned
<code>details</code>	optional argument to specify whether to show details about the records where inconsistencies were detected

<code>covid19.Canada.data</code>	<i>function to import data for Canada as reported by Health Canada https://health-infobase.canada.ca/src/data/covidLive/covid19.csv</i>
----------------------------------	--

Description

function to import data for Canada as reported by Health Canada <https://health-infobase.canada.ca/src/data/covidLive/covid19.csv>

Usage

```
covid19.Canada.data(  
  data.fmt = "TS",  
  local.data = FALSE,  
  debrief = FALSE,  
  acknowledge = FALSE  
)
```

Arguments

<code>data.fmt</code>	"TS" for TimeSeries of cumulative cases or "original" for the data as original reported
<code>local.data</code>	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
<code>debrief</code>	boolean specifying whether information about the read data is going to be displayed in screen
<code>acknowledge</code>	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe with the latest data reported by "OpenData Toronto" for the city of Toronto, ON - Canada

covid19.data	<i>function to read "live" data from reported covid19 cases</i>
--------------	---

Description

function to read "live" data from reported covid19 cases

Usage

```
covid19.data(
  case = "aggregated",
  local.data = FALSE,
  debrief = FALSE,
  acknowledge = FALSE
)
```

Arguments

case	a string indicating the category of the data, possible values are: "aggregated" : latest number of cases *aggregated* by country, "ts-confirmed" : time data of confirmed cases, "ts-deaths" : time series data of fatal cases, "ts-recovered" : time series data of recovered cases, "ts-ALL" : all time series data combined, "ts-confirmed-US" : time series data of confirmed cases for the United States, "ts-deaths-US" : time series data of fatal cases for the United States, "ts-dep-confirmed" : time series data of confirmed cases as originally reported (depricated), "ts-dep-deaths" : time series data of deaths as originally reported (depricated), "ts-dep-recovered" : time series data of recovered cases as originally reported (depricated), "ALL": all of the above "ts-Toronto" : data for the City of Toronto, ON - Canada
local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
debrief	boolean specifying whether information about the read data is going to be displayed in screen
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe (or a list in the case of "ALL") with the daily worldwide indicated type of data per country/region/city

Examples

```
# reads all possible datasets, returnin a list
covid19.all.datasets <- covid19.data("ALL")

# reads the latest aggregated data
```

```
covid19.ALL.agg.cases <- covid19.data("aggregated")
# reads time series data for casualties
covid19.TS.deaths <- covid19.data("ts-deaths")
```

covid19.genomic.data *main master (wrapper) function to obtain different types of genomic data for the SARS-CoV-2 virus*

Description

main master (wrapper) function to obtain different types of genomic data for the SARS-CoV-2 virus

Usage

```
covid19.genomic.data(
  type = "genome",
  src = "livedata",
  graphics.ON = TRUE,
  accOnly = TRUE
)
```

Arguments

type	type of data to retrieve, options are: 'genome', 'genomic', 'fasta', 'nucleotide', 'protein', 'ptree'
src	source of the data: "livedata", "repo" or "local"
graphics.ON	boolean option for display associated graphics
accOnly	boolean indicator for getting only accession codes or whole records

covid19.JHU.data *function to read "live" data as reported by JHU's CCSE repository*

Description

function to read "live" data as reported by JHU's CCSE repository

Usage

```
covid19.JHU.data(
  case = "aggregated",
  local.data = FALSE,
  debrief = FALSE,
  acknowledge = FALSE
)
```

Arguments

case	a string indicating the category of the data, possible values are: "aggregated" : latest number of cases *aggregated* by country, "ts-confirmed" : time data of confirmed cases, "ts-deaths" : time series data of fatal cases, "ts-recovered" : time series data of recovered cases, "ts-ALL" : all time series data combined, "ts-confirmed-US" : time series data of confirmed cases for the United States, "ts-deaths-US" : time series data of fatal cases for the United States, "ts-dep-confirmed" : time series data of confirmed cases as originally reported (deprecated), "ts-dep-deaths" : time series data of deaths as originally reported (deprecated), "ts-dep-recovered" : time series data of recovered cases as originally reported (deprecated), "ALL": all of the above "Toronto" : data for the City of Toronto, ON - Canada
local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
debrief	boolean specifying whether information about the read data is going to be displayed in screen
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe (or a list in the case of "ALL") with the daily worldwide indicated type of data per country/region/city

Examples

```
# reads all possible datasets, return in a list
covid19.all.datasets <- covid19.data("ALL")

# reads the latest aggregated data
covid19.ALL.agg.cases <- covid19.data("aggregated")
# reads time series data for casualties
covid19.TS.deaths <- covid19.data("ts-deaths")
```

covid19.testing.data *function to read data related to covid19 Testing from OWID repo*

Description

function to read data related to covid19 Testing from OWID repo

Usage

```
covid19.testing.data(tgt = "testing", disclaimer = TRUE)
```

Arguments

tgt	selects between time series data ('testing') and details and overall view of the data ("testing.details")
disclaimer	indicates whether the information about the source of the data is disclosed

Value

dataframe containing list of countries (Entity) and testing data

covid19.Toronto.data *function to import data from the city of Toronto, ON - Canada as reported by the City of Toronto OR Open Data Toronto*

Description

function to import data from the city of Toronto, ON - Canada as reported by the City of Toronto OR Open Data Toronto

Usage

```
covid19.Toronto.data(
  origin = "OD",
  data.fmt = "TS",
  local.data = FALSE,
  debrief = FALSE,
  OLD.fmt = FALSE,
  acknowledge = FALSE
)
```

Arguments

origin	select between the "City of Toronto" ('city') OR "Open Data Toronto" ('OD')
data.fmt	"TS" for TimeSeries of cumulative cases or "original" for the data as reported in the google-document with multiple sheets
local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
debrief	boolean specifying whether information about the read data is going to be displayed in screen
OLD.fmt	boolean flag to specify if the data is being read in an old format
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe (or a list in the case of "original") with the latest data reported for the city of Toronto, ON - Canada

```
covid19.Toronto_city.data
```

```
function to import data from the city of Toronto, ON - Canada as  
reported by the City of Toronto https://www.toronto.ca/home/covid-  
19/covid-19-pandemic-data/
```

Description

function to import data from the city of Toronto, ON - Canada as reported by the City of Toronto <https://www.toronto.ca/home/covid-19/covid-19-pandemic-data/>

Usage

```
covid19.Toronto_city.data(  
  data.fmt = "TS",  
  local.data = FALSE,  
  debrief = FALSE,  
  OLD.fmt = FALSE,  
  acknowledge = FALSE  
)
```

Arguments

<code>data.fmt</code>	"TS" for TimeSeries of cumulative cases or "original" for the data as reported in the google-document with multiple sheets
<code>local.data</code>	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
<code>debrief</code>	boolean specifying whether information about the read data is going to be displayed in screen
<code>OLD.fmt</code>	boolean flag to specify if the data is being read in an old format
<code>acknowledge</code>	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe (or a list in the case of "original") with the latest data reported for the city of Toronto, ON - Canada

covid19.Toronto_OD.data

function to import data from the city of Toronto, ON - Canada as reported by Open Data Toronto <https://open.toronto.ca/dataset/covid-19-cases-in-toronto/> This dataset is updated WEEKLY.

Description

function to import data from the city of Toronto, ON - Canada as reported by Open Data Toronto <https://open.toronto.ca/dataset/covid-19-cases-in-toronto/> This dataset is updated WEEKLY.

Usage

```
covid19.Toronto_OD.data(  
  data.fmt = "TS",  
  local.data = FALSE,  
  debrief = FALSE,  
  acknowledge = FALSE  
)
```

Arguments

data.fmt	"TS" for TimeSeries of cumulative cases or "original" for the data as original reported
local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
debrief	boolean specifying whether information about the read data is going to be displayed in screen
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

a dataframe with the latest data reported by "OpenData Toronto" for the city of Toronto, ON - Canada

covid19.URL_csv.data *function to read CSV from URLs or local replicas*

Description

function to read CSV from URLs or local replicas

Usage

```
covid19.URL_csv.data(
  local.data = FALSE,
  acknowledge = FALSE,
  srcURL = "",
  srcName = "",
  locFileName = NA,
  locVarName = NA
)
```

Arguments

local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.
srcURL	URL from where to obtain the data
srcName	name of the source
locFileName	name of the file to read from local repo
locVarName	name of the variable loaded from local file

Value

data as oriignally obtained from the URL src

covid19.US.data	<i>function to read the TimeSeries US detailed data</i>
-----------------	---

Description

function to read the TimeSeries US detailed data

Usage

```
covid19.US.data(local.data = FALSE, debrief = FALSE, acknowledge = FALSE)
```

Arguments

local.data	boolean flag to indicate whether the data will be read from the local repo, in case of connectivity issues or data integrity
debrief	boolean specifying whether information about the read data is going to be displayed in screen
acknowledge	boolean flag to indicate that the user acknowledges where the data is coming from. If FALSE, display data acquisition messages.

Value

TimeSeries dataframe with data for the US

covid19.vaccination	<i>function to read data related to covid19 vaccinations</i>
---------------------	--

Description

function to read data related to covid19 vaccinations

Usage

```
covid19.vaccination(tgt = "global", data.fmt = "orig", disclaimer = TRUE)
```

Arguments

tgt	selects data type: 'global','us','country','locations'
data.fmt	selects the format of the data, options are: 'orig' (original as reported by the OWID repo)
disclaimer	indicates whether the information about the source of the data is disclosed

covid19dashboard	<i>covid19.analytics explorer dashboard</i>
------------------	---

Description

covid19.analytics explorer dashboard

Usage

```
covid19dashboard(locln = NULL)
```

Arguments

locln	geographical location to use as default
-------	---

Value

list with shinyApp UI and server

covid19Explorer	<i>covid19.analytics explorer dashboard</i>
-----------------	---

Description

covid19.analytics explorer dashboard

Usage

```
covid19Explorer(locn = NULL)
```

Arguments

locn	geographical location to use as default
------	---

data.checks	<i>function to check for data integrity and data consistency</i>
-------------	--

Description

function to check for data integrity and data consistency

Usage

```
data.checks(
  data,
  n0 = 5,
  nf = ncol(data),
  datasetName = "",
  details = TRUE,
  disclose = FALSE
)
```

Arguments

data	dataset to analyze
n0	column where the cumulative data begins
nf	column where the cumulative data ends
datasetName	optional argument to display the name of the dataset
details	optional argument to specify whether to show details about the records where inconsistencies were detected
disclose	boolean flag to indicate whether index of problematic entries are returned

estimateRRs	<i>estimate rolling rates for a given geographical location for an specific TS data</i>
-------------	---

Description

estimate rolling rates for a given geographical location for an specific TS data

Usage

```
estimateRRs(
  data = NULL,
  geo.loc = NULL,
  period = NULL,
  graphics.ON = TRUE,
  splitG = TRUE
)
```

Arguments

data	time series dataset to consider
geo.loc	country/region to analyze
period	length of window
graphics.ON	boolean flag to activate/deactivate graphical output
splitG	boolean flag for having the graphical output separated or not

Examples

```
# the following examples take longer than 10 sec, and triggers CRAN checks
## Not run:
estimateRRs(covid19.data("TS-all"), geo.loc='Peru', period=7)
estimateRRs(covid19.data("TS-all"),
  geo.loc=c('Peru','Argentina','Uruguay','US','Spain','Japan'), period=7)

## End(Not run)
```

generate.SIR.model	<i>function to generate a simple SIR (Susceptible-Infected-Recovered) model based on the actual data of the covid19 cases</i>
--------------------	---

Description

function to generate a simple SIR (Susceptible-Infected-Recovered) model based on the actual data of the covid19 cases

Usage

```

generate.SIR.model(
  data = NULL,
  geo.loc = "Hubei",
  t0 = NULL,
  t1 = NULL,
  deltaT = NULL,
  tfinal = 90,
  fatality.rate = 0.02,
  tot.population = 1.4e+09,
  staticPlt = TRUE,
  interactiveFig = FALSE,
  add.extras = FALSE
)

```

Arguments

<code>data</code>	time series dataset to consider
<code>geo.loc</code>	country/region to analyze
<code>t0</code>	initial period of time for data consideration
<code>t1</code>	final period of time for data consideration
<code>deltaT</code>	interval period of time from <code>t0</code> , ie. number of days to consider since <code>t0</code>
<code>tfinal</code>	total number of days
<code>fatality.rate</code>	rate of causality, default value of 2 percent
<code>tot.population</code>	total population of the country/region
<code>staticPlt</code>	optional flag to activate/deactive plotting of the data and the SIR model generated
<code>interactiveFig</code>	optional flag to activate/deactive the generation of an interactive plot of the data and the SIR model generated
<code>add.extras</code>	boolean flag to add extra indicators, such as, the "force of infection" and time derivatives

Examples

```

data <- covid19.data("ts-confirmed")
generate.SIR.model(data,"Hubei", t0=1,t1=15)
generate.SIR.model(data,"Germany", tot.population=83149300)
generate.SIR.model(data,"Uruguay", tot.population=3500000)
generate.SIR.model(data,"Canada", tot.population=37590000, add.extras=TRUE)

```

geographicalRegions	<i>function to define continents and its constituent countries</i>
---------------------	--

Description

function to define continents and its constituent countries

Usage

```
geographicalRegions(cont = NULL)
```

Arguments

cont	optional argumetn, to specify a particular continent; if no argument is given then it returns all the continents and countries for each
------	---

Value

list with the composition of continents

getFile	<i>auxiliary function to download files in a protected fashion, i.e. against errors in the downloading procedure</i>
---------	--

Description

auxiliary function to download files in a protected fashion, i.e. against errors in the downloading procedure

Usage

```
getFile(url = NULL, fileName = NULL)
```

Arguments

url	resource's URL on the web
fileName	name of the resource (file) to download keywords internal

growth.rate	<i>function to compute daily changes and "Growth Rates" per location; "Growth Rates" defined as the ratio between changes in consecutive days</i>
-------------	---

Description

function to compute daily changes and "Growth Rates" per location; "Growth Rates" defined as the ratio between changes in consecutive days

Usage

```
growth.rate(
  data0,
  geo.loc = NULL,
  stride = 1,
  info = "",
  staticPlt = TRUE,
  interactiveFig = FALSE,
  interactive.display = TRUE
)
```

Arguments

data0	data.frame with <i>*time series*</i> data from covid19
geo.loc	list of locations
stride	how frequently to compute the growth rate in units of days
info	additional information to include in plots' title
staticPlt	boolean flag to indicate whether static plots would be generated or not
interactiveFig	boolean flag to indicate whether interactive figures would be generated or not
interactive.display	boolean flag to indicate whether the interactive plot will be displayed (pushed) to your browser

Value

a list containing two dataframes: one reporting changes on daily basis and a second one reporting growth rates, for the indicated regions

Examples

```
###\donttest{
# read data for confirmed cases
data <- covid19.data("ts-confirmed")
# compute changes and growth rates per location for all the countries
# growth.rate(data)
```

```
# compute changes and growth rates per location for 'Italy'
growth.rate(data,geo.loc="Italy")
# compute changes and growth rates per location for 'Italy' and 'Germany'
growth.rate(data,geo.loc=c("Italy","Germany"))
###}
```

integrity.check	<i>function that determines whether there are integrity issues within the datasets or changes to the structure of the data as reported by JHU/CCSEGIS</i>
-----------------	---

Description

function that determines whether there are integrity issues within the datasets or changes to the structure of the data as reported by JHU/CCSEGIS

Usage

```
integrity.check(data, datasetName = "", disclose = FALSE, recommend = TRUE)
```

Arguments

data	dataset to analyze
datasetName	optional argument to display the name of the dataset
disclose	boolean flag to indicate whether index of problematic entries are returned
recommend	optional flag to recommend further actions

itrends	<i>function to visualize trends in daily changes in time series data interactively</i>
---------	--

Description

function to visualize trends in daily changes in time series data interactively

Usage

```
itrends(
  ts.data = NULL,
  geo.loc = NULL,
  with.totals = FALSE,
  fileName = NULL,
  interactive.display = TRUE
)
```

Arguments

<code>ts.data</code>	time series dataset to process
<code>geo.loc</code>	geographical location, country/region or province/state to restrict the analysis to
<code>with.totals</code>	a boolean flag to indicate whether the global totals should be displayed with the records for the specific location
<code>fileName</code>	file where to save the HTML version of the interactive figure
<code>interactive.display</code>	boolean flag to indicate whether the interactive plot will be displayed (pushed) to your browser

<code>live.map</code>	<i>function to map cases in an interactive map</i>
-----------------------	--

Description

function to map cases in an interactive map

Usage

```
live.map(
  data = covid19.data(),
  select.projctn = TRUE,
  projctn = "orthographic",
  title = "",
  no.legend = FALSE,
  szRef = 0.2,
  fileName = NULL,
  interactive.display = TRUE
)
```

Arguments

<code>data</code>	data to be used
<code>select.projctn</code>	argument to activate or deactivate the pulldown menu for selecting the type of projection
<code>projctn</code>	initial type of map-projection to use, possible values are: "equirectangular" "mercator" "orthographic" "natural earth" "kavrayskiy7" "miller" "robinson" "eckert4" "azimuthal equal area" "azimuthal equidistant" "conic equal area" "conic conformal" "conic equidistant" "gnomonic" "stereographic" "mollweide" "hammer" "transverse mercator" "albers usa" "winkel tripel" "aitoff" "sinusoidal"
<code>title</code>	a string with a title to add to the plot
<code>no.legend</code>	parameter to turn off or on the legend on the right with the list of countries

szRef numerical value to use as reference, to scale up the size of the bubbles in the map, from 0 to 1 (smaller value → larger bubbles)

fileName file where to save the HTML version of the interactive figure

interactive.display boolean argument for enabling or not displaying the figure

Examples

```
## Not run:
# retrieve aggregated data
data <- covid19.data("aggregated")
# interactive map of aggregated cases -- with more spatial resolution
live.map(data)

# interactive map of the time series data of the confirmed cases
# with less spatial resolution, ie. aggregated by country
live.map(covid19.data("ts-confirmed"))

## End(Not run)
```

movingFn	<i>generic fn that computes the "fn" on a moving window</i>
----------	---

Description

generic fn that computes the "fn" on a moving window

Usage

```
movingFn(x, fn = mean, period = length(x), direction = "forward")
```

Arguments

x a numeric vector

fn a function to be applied/computed, default is set to mean()

period size of the "moving window", default set to the length of the vector

direction type of moving average to consider: "forward", "centered", "backward"; ie. whether the window computation is ("centered" / "forward" / "backward") wrt the data series

Value

a vector with the 'moving operation' applied to the x vector

mrollingRates	<i>function to compute a rolling fn (rate) of multiple quantities from TS data, eg. fatality and recovery rate</i>
---------------	--

Description

function to compute a rolling fn (rate) of multiple quantities from TS data, eg. fatality and recovery rate

Usage

```
mrollingRates(data = NULL, geo.loc = NULL, fn = mean, period)
```

Arguments

data	time series dataset to consider
geo.loc	country/region to analyze
fn	function to compute rolling
period	length of window

mtrends	<i>function to visualize different indicators for trends in daily changes of cases reported as time series data, for mutiple (or single) locations</i>
---------	--

Description

function to visualize different indicators for trends in daily changes of cases reported as time series data, for mutiple (or single) locations

Usage

```
mtrends(data, geo.loc = NULL, confBnd = TRUE, info = "")
```

Arguments

data	data.frame with <i>*time series*</i> data from covid19
geo.loc	list of locations
confBnd	flag to activate/deactivate drawing of confidence bands base on a moving average window
info	additional info to display in the plot

Examples

```
# triggers CRAN checks for timing
## Not run:
ts.data <- covid19.data("ts-confirmed")
mtrends(ts.data, geo.loc=c("Canada","Ontario","Uruguay","Italy"))

## End(Not run)
```

nullify.data	<i>remove inconsistencies from data by removing 'suspicious' entries</i>
--------------	--

Description

remove inconsistencies from data by removing 'suspicious' entries

Usage

```
nullify.data(data, stringent = FALSE)
```

Arguments

data	dataset to process
stringent	only return records with "complete cases"

OWID.repos	<i>function to define the OWID repos URLs</i>
------------	---

Description

function to define the OWID repos URLs

Usage

```
OWID.repos(tgt)
```

Arguments

tgt	target case: 'VAC.global','VAC.us','VAC.country','VAC.locations','testing','testing.details'
-----	--

Value

URL

pandemics.data	<i>function to retrieve historical pandemics data</i>
----------------	---

Description

function to retrieve historical pandemics data

Usage

```
pandemics.data(acknowledge = TRUE, show = FALSE, tgt = "pandemics")
```

Arguments

acknowledge	displays details on the data sources
show	displays data
tgt	which data set to read – options are 'pandemics' OR 'pandemics_vaccines'

Value

data.frame

pandemics.loaddata	<i>internal function to retrieve historical data on pandemics</i>
--------------------	---

Description

internal function to retrieve historical data on pandemics

Usage

```
pandemics.loaddata(  
  tgt.file = "pandemics.RDS",  
  acknowledge = TRUE,  
  show = FALSE,  
  src.descr = ""  
)
```

Arguments

tgt.file	which data set to read
acknowledge	displays details on the data sources
show	displays data
src.descr	description of the source of the data

Value

data.frame

plt.SIR.model	<i>function to plot the results from the SIR model fn</i>
---------------	---

Description

function to plot the results from the SIR model fn

Usage

```
plt.SIR.model(
    SIR.model,
    geo.loc = "",
    interactiveFig = FALSE,
    fileName = NULL,
    interactive.display = TRUE,
    add.extras = TRUE
)
```

Arguments

SIR.model	model resulting from the generate.SIR.model() fn
geo.loc	optional string to specify geographical location
interactiveFig	optional flag to activate interactive plot
fileName	file where to save the HTML version of the interactive figure
interactive.display	boolean flag to indicate whether the interactive plot will be displayed (pushed) to your browser
add.extras	boolean flag to add extra indicators, such as, the "force of infection" and time derivatives

preProcessingData	<i>auxiliary function to pre-process data per geographical location</i>
-------------------	---

Description

auxiliary function to pre-process data per geographical location

Usage

```
preProcessingData(data0, geo.loc)
```

Arguments

data0	data set
geo.loc	geographical location, can be a country, region, province or city #@keywords internal

red.devel.ver	<i>function to redirect users to install devel version from github</i>
---------------	--

Description

function to redirect users to install devel version from github

Usage

```
red.devel.ver(fileRDS, force = TRUE)
```

Arguments

fileRDS	missing data file
force	boolean flag to force stoping the code

report.summary	<i>function to summarize the current situation, will download the latest data and summarize the top provinces/cities per case</i>
----------------	---

Description

function to summarize the current situation, will download the latest data and summarize the top provinces/cities per case

Usage

```
report.summary(
  cases.to.process = "ALL",
  Nentries = 10,
  geo.loc = NULL,
  graphical.output = TRUE,
  saveReport = FALSE
)
```

Arguments

cases.to.process	which data to process: "TS" –time series–, "AGG" –aggregated– or "ALL" –time series and aggregated–
Nentries	number of top cases to display
geo.loc	geographical location to process
graphical.output	flag to deactivate graphical output
saveReport	flag to indicate whether the report should be saved in a file

Examples

```
# triggers CRAN checks for timing
## Not run:
# displaying top 10s
report.summary()

# get the top 20
report.summary(Nentries=20,graphical.output=FALSE)

# specify a location
report.summary(geo.loc="NorthAmerica")

## End(Not run)
```

report.Tor

function to obtain main indicators from Toronto data

Description

function to obtain main indicators from Toronto data

Usage

```
report.Tor(
  colTgts = c("Source.of.Infection", "Age.Group", "Client.Gender", "Outcome",
    "Neighbourhood.Name"),
  report = TRUE,
  staticPlt = TRUE,
  horiz.plts = 4,
  vert.plts = 3,
  same.Yaxis = TRUE,
  interactiveFig = FALSE,
  interactive.display = TRUE
)
```

Arguments

colTgts	optional argument to indicate which columns from the Toronto data to process
report	optional argument indicating whether a report will be printed to the screen
staticPlt	optional argument to indicate whether the 'static' graphical output is wanted or not
horiz.plts	number of plots in the horizontal direction
vert.plts	number of plots in the vertical direction
same.Yaxis	graphical argument to indicate if plots will use same y-axis
interactiveFig	boolean flag to indicate whether interactive figures would be generated or not

`interactive.display`

boolean flag to indicate whether the interactive plot will be displayed (pushed) to your browser

Value

list with statistics by selected as indicated in `colTgts`

`#@export`

<code>rollingRate</code>	<i>function to compute a rolling fn of a TS data</i>
--------------------------	--

Description

function to compute a rolling fn of a TS data

Usage

```
rollingRate(data, fn = mean, period = NULL)
```

Arguments

<code>data</code>	TS data
<code>fn</code>	function to compute rolling
<code>period</code>	length of window

Value

a vector with rolling values

<code>single.trend</code>	<i>function to visualize different indicators for trends in daily changes of cases reported as time series data</i>
---------------------------	---

Description

function to visualize different indicators for trends in daily changes of cases reported as time series data

Usage

```
single.trend(ts.data, confBnd = TRUE, info = "")
```

Arguments

ts.data	time series data
confBnd	optional argument to remove the drawing of a confidence band
info	additional information to display in plots

Examples

```

tor.data <- covid19.Toronto.data()
single.trend(tor.data[tor.data$status=="Active Cases",])

ts.data <- covid19.data("ts-confirmed")
ont.data <- ts.data[ ts.data$Province.State == "Ontario",]
single.trend(ont.data)

single.trend(ts.data[ ts.data$Country.Region=="Italy",])

```

sweep.SIR.models	<i>function to perform a sweep of models and generate values of R0</i>
------------------	--

Description

function to perform a sweep of models and generate values of R0

Usage

```

sweep.SIR.models(
  data = NULL,
  geo.loc = "Hubei",
  t0_range = 15:20,
  t1 = NULL,
  deltaT = NULL,
  tfinal = 90,
  fatality.rate = 0.02,
  tot.population = 1.4e+09
)

```

Arguments

data	time series dataset to consider
geo.loc	country/region to analyze
t0_range	range of initial date for data consideration
t1	final period of time for data consideration
deltaT	interval period of time from t0, ie. number of days to consider since t0
tfinal	total number of days
fatality.rate	rate of causality, default value of 2 percent
tot.population	total population of the country/region

Examples

```
# read TimeSeries data
TS.data <- covid19.data("TS-confirmed")
# select a location of interest, eg. France
# France has many entries, just pick "la France"
France.data <- TS.data[ (TS.data$Country.Region == "France") & (TS.data$Province.State == ""),]
# sweep values of R0 based on range of dates to consider for the model
ranges <- 15:20
deltaT <- 20
params_sweep <- sweep.SIR.models(data=France.data,geo.loc="France", t0_range=ranges, deltaT=deltaT)
# obtain the R0 values from the parameters
R0s <- unlist(params_sweep["R0",])
# nbr of infected cases
FR.infs<- preProcessingData(France.data,"France")
# average per range
# define ranges
lst.ranges <- lapply(ranges, function(x) x:(x+deltaT))
# compute averages
avg.FR.infs <- lapply(lst.ranges, function(x) mean(FR.infs[x]))
# plots
plot(R0s, type='b')
# plot vs average number of infected cases
plot(avg.FR.infs, R0s, type='b')
```

`totals.plt`

function to plot total number of cases per day for different groups

Description

function to plot total number of cases per day for different groups

Usage

```
totals.plt(
  data0 = NULL,
  geo.loc0 = NULL,
  one.plt.per.page = FALSE,
  log.plt = TRUE,
  with.totals = FALSE,
  interactive.fig = TRUE,
  fileName = NULL,
  interactive.display = TRUE
)
```

Arguments

`data0` time series dataset to process, default all the possible cases: 'confirmed' and 'deaths' for all countries/regions

geo.loc0 geographical location, country/region or province/state to restrict the analysis to
 one.plt.per.page boolean flag to have one plot per figure
 log.plt include a log scale plot in the static plot
 with.totals a boolean flag to indicate whether the totals should be displayed with the records
 for the specific location
 interactive.fig switch to turn off/on an interactive plot
 fileName file where to save the HTML version of the interactive figure
 interactive.display boolean argument for enabling or not displaying the interactive figure

Examples

```

# retrieve time series data
TS.data <- covid19.data("ts-ALL")

# static and interactive plot
totals.plt(TS.data)

```

tots.per.location	<i>function to compute totals per location</i>
-------------------	--

Description

function to compute totals per location

Usage

```

tots.per.location(
  data,
  geo.loc = NULL,
  confBnd = FALSE,
  nbr.plts = 1,
  info = ""
)

```

Arguments

data data.frame with **time series** data from covid19
 geo.loc list of locations
 confBnd flag to activate/deactivate drawing of confidence bands base on a moving aver-
 age window
 nbr.plts parameter to control the number of plots to display per figure
 info additional info to display in plots' titles

Value

a list or dataframe with totals per specified locations and type of case

Examples

```
# read data for confirmed cases
data <- covid19.data("ts-confirmed")
# compute totals per location for all the countries

tots.per.location(data)

# compute totals per location for 'Italy'
tots.per.location(data,geo.loc="Italy")
# compute totals per location for 'Italy' and 'Germany'
tots.per.location(data,geo.loc=c("Italy","Germany"))
```

X.covid19.genomic.data

*function to obtain sequencing data from NCBI Reference:
https://www.ncbi.nlm.nih.gov/nucore/NC_045512.2*

Description

function to obtain sequencing data from NCBI Reference: https://www.ncbi.nlm.nih.gov/nucore/NC_045512.2

Usage

```
X.covid19.genomic.data(graphics.ON = TRUE)
```

Arguments

graphics.ON flag to activate/deactivate graphical output

Examples

```
# obtain covid19's genomic data
covid19.gen.seq <- covid19.genomic.data()
# display the actual RNA seq
covid19.gen.seq$NC_045512.2
```

Index

c19.fasta.data, [3](#)
c19.genomic.data, [3](#)
c19.NP.fasta.data, [4](#)
c19.NPs.data, [4](#)
c19.OWID.data, [5](#)
c19.ptree.data, [5](#)
c19.refGenome.data, [6](#)
consistency.check, [6](#)
covid19.Canada.data, [7](#)
covid19.data, [8](#)
covid19.genomic.data, [9](#)
covid19.JHU.data, [9](#)
covid19.testing.data, [10](#)
covid19.Toronto.data, [11](#)
covid19.Toronto_city.data, [12](#)
covid19.Toronto_OD.data, [13](#)
covid19.URL_csv.data, [13](#)
covid19.US.data, [14](#)
covid19.vaccination, [15](#)
covid19dashboard, [15](#)
covid19Explorer, [16](#)

data.checks, [16](#)

estimateRRs, [17](#)

generate.SIR.model, [17](#)
geographicalRegions, [19](#)
getFile, [19](#)
growth.rate, [20](#)

integrity.check, [21](#)
itrends, [21](#)

live.map, [22](#)

movingFn, [23](#)
mrollingRates, [24](#)
mtrends, [24](#)

nullify.data, [25](#)

OWID.repos, [25](#)

pandemics.data, [26](#)
pandemics.loaddata, [26](#)
plt.SIR.model, [27](#)
preProcessingData, [27](#)

red.devel.ver, [28](#)
report.summary, [28](#)
report.Tor, [29](#)
rollingRate, [30](#)

single.trend, [30](#)
sweep.SIR.models, [31](#)

totals.plt, [32](#)
tots.per.location, [33](#)

X.covid19.genomic.data, [34](#)