Package ‘projections’

April 22, 2021

Title Project Future Case Incidence

Version 0.5.4

Description

Provides functions and graphics for projecting daily incidence based on past incidence, and estimates of the serial interval and reproduction number. Projections are based on a branching process using a Poisson-distributed number of new cases per day, similar to the model used for estimating R in ‘EpiEstim’ or in ‘earlyR’, and described by Nouvellet et al. (2017) <doi:10.1016/j.epidem.2017.02.012>. The package provides the S3 class ‘projections’ which extends ‘matrix’, with accessors and additional helpers for handling, subsetting, merging, or adding these objects, as well as dedicated printing and plotting methods.

Depends R (>= 3.5.0)

License MIT + file LICENSE

Encoding UTF-8

Suggests testthat, roxygen2, knitr, rmarkdown, outbreaks, magrittr, distrcrete, svglite, EpiEstim

Imports stats, utils, incidence (>= 1.4.1), ggplot2, methods

RoxygenNote 7.1.1

URL https://www.repidemicsconsortium.org/projections/

BugReports https://github.com/reconhub/projections/issues

Config/testthat/edition 3

NeedsCompilation no

Author Thibaut Jombart [aut, cre],

Pierre Nouvellet [aut],

Sangeeta Bhatia [ctb],

Zhian N. Kamvar [ctb],

Tim Taylor [ctb],

Stephane Ghozzi [ctb]

Maintainer Thibaut Jombart <thibautjombart@gmail.com>

Repository CRAN

Date/Publication 2021-04-22 09:30:05 UTC
as.matrix.projections

Conversion of projections objects

Description

These functions convert projections objects into other classes.

Usage

```r
## S3 method for class 'projections'
as.matrix(x, ...)

## S3 method for class 'projections'
as.data.frame(x, ..., long = FALSE)
```

Arguments

- **x**: An projections object, or an object to be converted as projections (see details).
- **...**: Further arguments passed to other functions (no used).
- **long**: A logical indicating if the output data.frame should be 'long', i.e. where a single column containing 'groups' is added in case of data computed on several groups.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

See Also

the `project` function to generate the 'projections' objects.
**Description**

This function builds a valid `projections` object from some input simulations and dates.

**Usage**

```r
build_projections(x, dates = NULL, cumulative = FALSE, order_dates = TRUE)
```

**Arguments**

- `x`: A matrix of simulated incidence stored as integers, where rows correspond to dates and columns to simulations.
- `dates`: A vector of dates containing one value per row in `x`; acceptable formats are: integer, `Date`, and `POSIXct`; if `NULL`, the time steps will be counted, with the first dates corresponding to 0.
- `cumulative`: A logical indicating if data represent cumulative incidence; defaults to `FALSE`.
- `order_dates`: A logical indicating whether the dates should be ordered, from the oldest to the most recent one; ‘TRUE’ by default.

**Author(s)**

Thibaut Jombart <thibautjombart@gmail.com>

**See Also**

- the `project` function to generate the 'projections' objects.

---

**cumulate.projections**  
**Compute cumulative projections**

**Description**

cumulate is an S3 generic to compute cumulative numbers defined in the package `incidence`. The method for `projections` objects turns predicted incidences into cumulative incidences over time.

**Usage**

```r
## S3 method for class 'projections'
cumulate(x)
```

**Arguments**

- `x`: A `projections` object.
get_dates

Author(s)
Thibaut Jombart <thibautjombart@gmail.com>

See Also
- The `project` function to generate the projections objects.

Examples

```r
if (require(distcrete) && require(incidence)) {

## simulate basic epicurve
dat <- c(0, 2, 2, 3, 3, 5, 5, 5, 6, 6, 6, 6)
i <- incidence(dat)

## example with a function for SI
si <- distcrete("gamma", interval = 1L,
    shape = 1.5,
    scale = 2, w = 0)
set.seed(1)
pred_1 <- project(i, runif(100, 0.8, 1.9), si, n_days = 30)
plot_1 <- plot(pred_1)

## cumulative predictions
pred_1_cum <- cumulate(pred_1)
pred_1_cum
plot(pred_1_cum)
}
```

---

get_dates

Access content projections objects

Description
These simple helper functions retrieve content from projections objects. They currently include:

Usage
```r
## S3 method for class 'projections'
get_dates(x, ...)
```

Arguments
- `x` A projections object.
- `...` Further arguments passed to methods; currently not used.
merge_add_projections

Details

• get_dates: get dates of the predictions.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

Examples

if (require(distcrete) && require(incidence)) {
  withAutoprint{
    ## prepare input: epicurve and serial interval
    dat <- c(0, 2, 2, 3, 3, 5, 5, 5, 6, 6, 6, 6)
    i <- incidence(dat)
    si <- distcrete("gamma", interval = 1L,
                     shape = 1.5,
                     scale = 2, w = 0)

    ## make predictions
    pred_1 <- project(i, 1.2, si, n_days = 30)
    pred_1

    ## retrieve content
    get_dates(pred_1)
    max(i$dates) # predictions start 1 day after last incidence
  }
}

merge_add_projections Add data of different projections objects

Description

This function adds counts from several ‘projections’ objects, making sure that they all use the same dates, adding rows of '0' where needed. Simulations (columns) are recycled when needed if some objects have less simulations than others. The same operation is implemented by the ‘+’ operator.

Usage

merge_add_projections(x)

## S3 method for class 'projections'
a + b
merge_projections

Merge a list of projections objects

Description

This function merges ‘projections’ objects, binding them by columns, making sure that they all use the same dates, adding rows of '0' where needed.
Usage

merge_projections(x)

Arguments

x A ‘list’ of ‘projections’ objects to be merged.

Author(s)

Thibaut Jombart

Examples

## generate toy data
dates <- Sys.Date() + c(0, 0, 2, 5, 6, 6, 7)
i <- incidence::incidence(dates)
si <- c(0.2, 0.5, 0.2, 0.1)
R0 <- 3.5

## make several projections objects
x <- lapply(1:10,
    function(j)
        project(x = i,
            si = si,
            R = R0,
            n_sim = 2 * j,
            R_fix_within = TRUE,
            n_days = j,
            model = "poisson"
))

## see all dimensions
lapply(x, dim)
merge_projections(x)

plot.projections

Plot projections objects

Description

The plot method of projections objects (output by the function project) shows quantiles of predicted incidence over time. The function add_projections can be used to add a similar plot to an existing incidence plot. This latter function is piping friendly (see examples).
Usage

```r
## S3 method for class 'projections'
plot(x, ylab = NULL, title = NULL, ...)

add_projections(
  p,
  x,
  quantiles = c(0.01, 0.05, 0.1, 0.5),
  ribbon = TRUE,
  boxplots = FALSE,
  palette = quantile_pal,
  quantiles_alpha = 1,
  linetype = 1,
  linesize = 0.5,
  ribbon_quantiles = NULL,
  ribbon_color = NULL,
  ribbon_alpha = 0.3,
  boxplots_color = "#47476b",
  boxplots_fill = "grey",
  boxplots_alpha = 0.8,
  outliers = TRUE
)
```

Arguments

- `x`: A projections object.
- `ylab`: An optional label for the y-axis. If missing will default to "predicted incidence" or, if cumulative, "predicted cumulative incidence".
- `title`: An optional title.
- `...`: Further arguments to be passed to `add_projections`.
- `p`: A previous incidence plot to which projections should be added.
- `quantiles`: A vector of quantiles to plot, automatically completed to be symmetric around the median.
- `ribbon`: A logical indicating if a ribbon should be drawn; defaults to TRUE.
- `boxplots`: A logical indicating if boxplots should be drawn.
- `palette`: A color palette to be used for plotting the quantile lines; defaults to `quantile_pal`.
- `quantiles_alpha`: A number used to control the transparency of the quantile lines, from 0 (full transparency) to 1 (full opacity); defaults to 1.
- `linetype`: An integer indicating the type of line used for plotting the quantiles; defaults to 1 for a plain line.
- `linesize`: An integer indicating the size of line used for plotting the quantiles; defaults to 0.5.
ribbon_quantiles
A vector of 2 quantiles to be used to determine the limits of the ribbon; if NULL (default); uses the most extreme quantiles if available; if quantiles are not provided, the daily range will be used.

ribbon_color
Any valid color, used for the ribbon.

ribbon_alpha
A number used to control the transparency of the ribbon, from 0 (full transparency) to 1 (full opacity); defaults to 0.3.

boxplots_color
Any valid color, used for the boxplot.

boxplots_fill
Any valid color, used for filling the boxplot.

boxplots_alpha
A number used to control the transparency of the boxplots, from 0 (full transparency) to 1 (full opacity); defaults to 0.8.

outliers
A logical indicating if outliers should be displayed alongside the boxplots; defaults to TRUE.

Author(s)
Thibaut Jombart <thibautjombart@gmail.com>

See Also
project to generate projections

Examples

```r
if (require(outbreaks) && require(distcrete) && require(incidence) && require(magrittr)) {

  si <- distcrete("gamma",
      interval = 1L,
      shape = 2.4,
      scale = 4.7,
      w = 0.5)

  i <- incidence(ebola_sim$linelist$date_of_onset)
  plot(i)

  ## add projections after the first 100 days, over 60 days
  set.seed(1)
  proj <- project(x = i[1:100], R = 1.4, si = si, n_days = 60)

  ## plotting projections: different options
  plot(proj)
  plot(proj, quantiles = c(.025, .5)) # 95% CI
  plot(proj, ribbon_color = "red", quantiles = FALSE) # range
  plot(proj, ribbon_color = "red", quantiles = FALSE,
       ribbon_quantiles = c(.025, .5))
  plot(proj, boxplots = TRUE, quantiles = FALSE, ribbon = FALSE)
```
print.projections

Print method for projections objects

Description

This method prints the content of projections objects.

Usage

## S3 method for class 'projections'
print(x, ...)

Arguments

x A projections object.

... further parameters to be passed to other methods (currently not used)

Author(s)

Thibaut Jombart (<thibautjombart@gmail.com>)
### project

**Project future incidence**

**Description**

This function simulates future incidence based on past incidence data, a selection of plausible reproduction numbers (R), and the distribution of the serial interval (time from primary onset to secondary onset).

**Usage**

```r
project(
  x,  
  R,  
  si,  
  n_sim = 100,  
  n_days = 7,  
  R_fix_within = FALSE,  
  model = c("poisson", "negbin"),  
  size = 0.03,  
  time_change = NULL,  
  instantaneous_R = FALSE  
)
```

**Arguments**

- `x` An incidence object containing daily incidence; other time intervals will trigger an error.
- `R` A vector of numbers representing plausible reproduction numbers; for instance, these can be samples from a posterior distribution using the 'earlyR' or 'EpiEstim' packages. If 'time_change' is provided, then it must be a 'vector' (for fixed values of R per time window) or a 'list' of vectors (for separate distributions of R per time window), with one element more than the number of dates in 'time_change'.
- `si` A function computing the serial interval, or a 'numeric' vector providing its mass function, starting a day 1, so that si[i] is the PMF for serial interval of 'i'. The model implicitly assumes that 'si[0] = 0'. For functions, we strongly recommend using the RECON package distcrete to obtain such distribution (see example).
- `n_sim` The number of epicurves to simulate. Defaults to 100.
- `n_days` The number of days to run simulations for. Defaults to 14.
- `R_fix_within` A logical indicating if R should be fixed within simulations (but still varying across simulations). If FALSE, R is drawn for every simulation and every time step. Fixing values within simulations favours more extreme predictions (see details).
model Distribution to be used for projections. Must be one of "poisson" or "negbin" (negative binomial process). Defaults to poisson

size size parameter of negative binomial distribution. Ignored if model is poisson

time_change an optional vector of times at which the simulations should use a different sample of reproduction numbers, provided in days into the simulation (so that day '1' is the first day after the input 'incidence' object); if provided, 'n' dates in 'time_change' will produce 'n+1' time windows, in which case 'R' should be a list of vectors of 'n+1' 'R' values, one per each time window.

instantaneous_R a boolean specifying whether to assume 'R' is the case reproduction number ('instantaneous_R = FALSE', the default), or the instantaneous reproduction number ('instantaneous_R = TRUE'). If 'instantaneous_R = FALSE' then values of 'R' at time 't' will govern the mean number of secondary cases of all cases infected at time 't', even if those secondary cases appear after 't'. In other words, 'R' will characterise onwards transmission from infectors depending on their date of infection. If 'instantaneous_R = TRUE' then values of 'R' at time 't' will govern the mean number of secondary cases made at time 't' by all cases infected before 't'. In other words, 'R' will characterise onwards transmission at a given time.

Details

The decision to fix R values within simulations (R_fix_within) reflects two alternative views of the uncertainty associated with R. When drawing R values at random from the provided sample, (R_fix_within set to FALSE), it is assumed that R varies naturally, and can be treated as a random variable with a given distribution. When fixing values within simulations (R_fix_within set to TRUE), R is treated as a fixed parameter, and the uncertainty is merely a consequence of the estimation of R. In other words, the first view is rather Bayesian, while the second is more frequentist.

Author(s)

Pierre Nouvellet (original model), Thibaut Jombart (bulk of the code), Sangeeta Bhatia (Negative Binomial model), Stephane Ghozzi (bug fixes time varying R)

Examples

```r
## example using simulated Ebola outbreak
if (require(outbreaks) &&
 require(distcrete) &&
 require(incidence) &&
 require(magrittr)) {

 si <- distcrete("gamma", interval = 1L,
 shape = 2.4,
 scale = 4.7,
 w = 0.5)

 i <- incidence(ebola_sim$linelist$date_of_onset)
 plot(i)
```
## projections after the first 100 days, over 60 days, fixed R to 2.1

```r
set.seed(1)
proj_1 <- project(x = i[1:100], R = 2.1, si = si, n_days = 60)
plot(proj_1)
```

## add projections to incidence plot
```r
plot(i[1:160]) %>% add_projections(proj_1)
```

## projections after the first 100 days, over 60 days, using a sample of R

```r
set.seed(1)
R <- rnorm(100, 1.8, 0.2)
hist(R, col = "grey", border = "white", main = "Distribution of R")
proj_2 <- project(x = i[1:100], R = R, si = si, n_days = 60)
```

## add projections to incidence plot
```r
plot(i[1:160]) %>% add_projections(proj_2)
```

## same with R constant per simulation (more variability)

```r
set.seed(1)
proj_3 <- project(x = i[1:100], R = R, si = si, n_days = 60,
                  R_fix_within = TRUE)
```

## add projections to incidence plot
```r
plot(i[1:160]) %>% add_projections(proj_3)
```

## time-varying R, 2 periods, R is 2.1 then 0.5

```r
set.seed(1)
proj_4 <- project(i,
                  R = c(2.1, 0.5),
                  si = si,
                  n_days = 60,
                  time_change = 40,
                  n_sim = 100)
plot(proj_4)
```

## time-varying R, 2 periods, separate distributions of R for each period

```r
set.seed(1)
R_period_1 <- runif(100, min = 1.1, max = 3)
R_period_2 <- runif(100, min = 0.6, max = .9)
proj_5 <- project(i,
                  R = list(R_period_1, R_period_2),
                  si = si,
```
Use the `plot` function to visualize the projections:

```r
plot(proj_5)
```

### Usage

```r
## S3 method for class 'projections'
summary(
  object,
  quantiles = c(0.025, 0.25, 0.5, 0.75, 0.975),
  mean = TRUE,
  sd = TRUE,
  min = TRUE,
  max = TRUE,
  ...
)
```

### Arguments

- `object`: A `projections` object to summarise
- `quantiles`: A numeric vector indicating which quantiles should be computed; ignored if `FALSE` or of length 0
- `mean`: a logical indicating of the mean should be computed
- `sd`: a logical indicating of the standard deviation should be computed
- `min`: a logical indicating of the minimum should be computed
- `max`: a logical indicating of the maximum should be computed
- `...`: only present for compatibility with the generic

```r
if (require(incidence))
  i <- incidence::incidence(as.Date('2020-01-23'))
  si <- c(0.2, 0.5, 0.2, 0.1)
  R0 <- 2
  p <- project(x = i, si = si, R = R0, n_sim = 2, R_fix_within = TRUE, n_days = 10, model = "poisson")
  summary(p)
```

### Description

This method summarises predicted epidemic trajectories contained in a `projections` object by days, deriving the mean, standard deviation, and user-specified quantiles for each day.

### Example

```r
summary(p)
```
### Description

Two functions can be used to subset projections objects. The operator "[" can be used as for matrices, using the syntax `x[i, j]` where 'i' is a subset of dates, and 'j' is a subset of simulations.

### Usage

```r
## S3 method for class 'projections'
x[i, j]
```  
```r
## S3 method for class 'projections'
subset(x, ..., from = NULL, to = NULL, sim = TRUE)
```

### Arguments

- `x`: An projections object, generated by the function `project`.
- `i`: A subset of dates to retain.
- `j`: A subset of groups to retain.
- `...`: Further arguments passed to other methods (not used).
- `from`: The starting date; data strictly before this date are discarded.
- `to`: The ending date; data strictly after this date are discarded.
- `sim`: (optional) The simulations to retained, indicated as subsets of the columns of `x`.

### Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

### See Also

The `project` function to generate the 'projections' objects.
Index

+.projections (merge_add_projections), 5
[.projections, 15

add_projections (plot.projections), 7
as.data.frame.projections
  (as.matrix.projections), 2
as.matrix.projections, 2

build_projections, 3

cumulate.projections, 3

get_dates, 4

merge_add_projections, 5
merge_projections, 6

plot.projections, 7
print.projections, 10
project, 2–4, 7, 9, 11, 15

subset.projections ([.projections), 15
summary.projections, 14