Package 'incidence'

June 19, 2025

Type Package

Title Compute, Handle, Plot and Model Incidence of Dated Events

Version 1.7.6

Description Provides functions and classes to compute, handle and visualise incidence from dated events for a defined time interval. Dates can be provided in various standard formats. The class 'incidence' is used to store computed incidence and can be easily manipulated, subsetted, and plotted. In addition, log-linear models can be fitted to 'incidence' objects using 'fit'. This package is part of the RECON (<https://www.repidemicsconsortium.org/>) toolkit for outbreak analysis.

Encoding UTF-8

License MIT + file LICENSE

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

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

RoxygenNote 7.3.2

Imports ggplot2 (>= 3.3.2), aweek (>= 0.2.0)

Suggests magrittr, outbreaks, testthat, vdiffr, knitr, rmarkdown, scales, cowplot

VignetteBuilder knitr

NeedsCompilation no

Author Thibaut Jombart [aut],
Zhian N. Kamvar [aut] (ORCID: <https://orcid.org/0000-0003-1458-7108>),
Rich FitzJohn [aut],
Tim Taylor [cre] (ORCID: <https://orcid.org/0000-0002-8587-7113>),
Jun Cai [ctb] (ORCID: <https://orcid.org/0000-0001-9495-1226>),
Sangeeta Bhatia [ctb],
Jakob Schumacher [ctb],
Juliet R.C. Pulliam [ctb] (ORCID: <https://orcid.org/0000-0003-3314-8223>)

Maintainer Tim Taylor <tim.taylor@hiddenelephants.co.uk>

Repository CRAN

Date/Publication 2025-06-19 21:50:02 UTC

Contents

`as.data.frame.incidence` *Conversion of incidence objects*

Description

These functions convert `incidence` objects into other classes.

Usage

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

as.incidence(x, ...)

## S3 method for class 'matrix'
as.incidence(
  x,
  dates = NULL,
  interval = NULL,
  standard = TRUE,
  isoweeks = standard,
  ...
)

## S3 method for class 'data.frame'
as.incidence(x, dates = NULL, interval = NULL, isoweeks = TRUE, ...)
```

```
## S3 method for class 'numeric'
as.incidence(x, dates = NULL, interval = NULL, isoweeks = TRUE, ...)
```

Arguments

x	An incidence object, or an object to be converted as incidence (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.
dates	A vector of dates, each corresponding to the (inclusive) lower limit of the bins.
interval	An integer indicating the time interval used in the computation of the incidence. If NULL, it will be determined from the first time interval between provided dates. If only one date is provided, it will trigger an error.
standard	A logical indicating whether standardised dates should be used. Defaults to TRUE.
isoweeks	Deprecated. Use standard.

Details

Conversion to incidence objects should only be done when the original dates are not available. In such case, the argument x should be a matrix corresponding to the $counts element of an incidence object, i.e. giving counts with time intervals in rows and named groups in columns. In the absence of groups, a single unnamed columns should be given. data.frame and vectors will be coerced to a matrix.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>, Rich Fitzjohn

See Also

the incidence() function to generate the 'incidence' objects.

Examples

```
## create fake data
data <- c(0,1,1,2,1,3,4,5,5,5,5,4,4,26,6,7,9)
sex <- sample(c("m","f"), length(data), replace=TRUE)

## get incidence per group (sex)
i <- incidence(data, groups = sex)
i
plot(i)

## convert to data.frame
as.data.frame(i)

## same, 'long format'
```

```
as.data.frame(i, long = TRUE)


## conversion from a matrix of counts to an incidence object
i$counts
new_i <- as.incidence(i$counts, i$dates)
new_i
all.equal(i, new_i)
```

`bootstrap` *Bootstrap incidence time series*

Description

This function can be used to bootstrap `incidence` objects. Bootstrapping is done by sampling with replacement the original input dates. See `details` for more information on how this is implemented.

Usage

```
bootstrap(x, randomise_groups = FALSE)
```

Arguments

`x`	An `incidence` object.
`randomise_groups`	A `logical` indicating whether groups should be randomised as well in the re-sampling procedure; respective group sizes will be preserved, but this can be used to remove any group-specific temporal dynamics. If `FALSE` (default), data are resampled within groups.

Details

As original data are not stored in `incidence` objects, the bootstrapping is achieved by multinomial sampling of date bins weighted by their relative incidence.

Value

An `incidence` object.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

See Also

find_peak to use estimate peak date using bootstrap

Examples

```
if (require(outbreaks) && require(ggplot2)) { withAutoprint({
  i <- incidence(fluH7N9_china_2013$date_of_onset)
  i
  plot(i)

  ## one simple bootstrap
  x <- bootstrap(i)
  x
  plot(x)

})}
```

`cumulate` *Compute cumulative 'incidence'*

Description

`cumulate` is an S3 generic to compute cumulative numbers, with methods for different types of objects:

Usage

```
cumulate(x)

## Default S3 method:
cumulate(x)

## S3 method for class 'incidence'
cumulate(x)
```

Arguments

`x`	An incidence object.

Details

- default method is a wrapper for `cumsum`
- `incidence` objects: computes cumulative incidence over time
- `projections` objects: same, for `projections` objects, implemented in the similarly named package; see `?cumulate.projections` for more information, after loading the package

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

See Also

The incidence() function to generate the 'incidence' objects.

Examples

```
dat <- as.integer(c(0,1,2,2,3,5,7))
group <- factor(c(1, 2, 3, 3, 3, 3, 1))
i <- incidence(dat, groups = group)
i
plot(i)

i_cum <- cumulate(i)
i_cum
plot(i_cum)
```

---

dim.incidence *Access various elements of an incidence object*

Description

Access various elements of an incidence object

Usage

```
## S3 method for class 'incidence'
dim(x)

get_interval(x, ...)

## Default S3 method:
get_interval(x, ...)

## S3 method for class 'incidence'
get_interval(x, integer = TRUE, ...)

get_n(x)

## Default S3 method:
get_n(x)

## S3 method for class 'incidence'
get_n(x)

get_timespan(x)

## Default S3 method:
```

```
get_timespan(x)

## S3 method for class 'incidence'
get_timespan(x)
```

Arguments

x	an incidence object.
...	Unused
integer	When TRUE (default), the interval will be converted to an integer vector if it is stored as a character in the incidence object.

Value

- dim() the dimensions in the number of bins and number of groups
- get_interval() if integer = TRUE: an integer vector, otherwise: the value stored in x$interval
- get_n() The total number of cases stored in the object
- get_timespan(): an integer denoting the timespan represented by the incidence object.

See Also

- get_counts() to access the matrix of counts
- get_dates() to access the dates on the right, left, and center of the interval.
- group_names() to access and possibly re-name the groups

Examples

```
set.seed(999)
dat <- as.Date(Sys.Date()) + sample(-3:50, 100, replace = TRUE)
x <- incidence(dat, interval = "month")

# the value stored in the interval element
get_interval(x)

# the numeric value of the interval in days
get_interval(x, integer = FALSE)

# the number of observations in the object
get_n(x)

# the length of time represented
get_timespan(x)

# the number of groups
ncol(x)

# the number of bins (intervals)
nrow(x)
```

`estimate_peak` *Estimate the peak date of an incidence curve using bootstrap*

Description

This function can be used to estimate the peak of an epidemic curve stored as `incidence`, using bootstrap. See bootstrap for more information on the resampling.

Usage

```
estimate_peak(x, n = 100, alpha = 0.05)
```

Arguments

`x`	An `incidence` object.
`n`	The number of bootstrap datasets to be generated; defaults to 100.
`alpha`	The type 1 error chosen for the confidence interval; defaults to 0.05.

Details

Input dates are resampled with replacement to form bootstrapped datasets; the peak is reported for each, resulting in a distribution of peak times. When there are ties for peak incidence, only the first date is reported.

Note that the bootstrapping approach used for estimating the peak time makes the following assumptions:

- the total number of event is known (no uncertainty on total incidence)
- dates with no events (zero incidence) will never be in bootstrapped datasets
- the reporting is assumed to be constant over time, i.e. every case is equally likely to be reported

Value

A list containing the following items:

- `observed`: the peak incidence of the original dataset
- `estimated`: the mean peak time of the bootstrap datasets
- `ci`: the confidence interval based on bootstrap datasets
- `peaks`: the peak times of the bootstrap datasets

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>, with inputs on caveats from Michael Höhle.

See Also

bootstrap for the bootstrapping underlying this approach and find_peak to find the peak in a single `incidence` object.

Examples

```
if (require(outbreaks) && require(ggplot2)) { withAutoprint({
  i <- incidence(fluH7N9_china_2013$date_of_onset)
  i
  plot(i)

  ## one simple bootstrap
  x <- bootstrap(i)
  x
  plot(x)

  ## find 95% CI for peak time using bootstrap
  peak_data <- estimate_peak(i)
  peak_data
  summary(peak_data$peaks)

  ## show confidence interval
  plot(i) + geom_vline(xintercept = peak_data$ci, col = "red", lty = 2)

  ## show the distribution of bootstrapped peaks
  df <- data.frame(peak = peak_data$peaks)
  plot(i) + geom_density(data = df,
                         aes(x = peak, y = 10 * ..scaled..),
                         alpha = .2, fill = "red", color = "red")

})}
```

`find_peak` *Find the peak date of an incidence curve*

Description

This function can be used to find the peak of an epidemic curve stored as an `incidence` object.

Usage

```
find_peak(x, pool = TRUE)
```

Arguments

`x`	An `incidence` object.
`pool`	If `TRUE` (default), any groups will be pooled before finding a peak. If `FALSE`, separate peaks will be found for each group.

Value

The date of the (first) highest incidence in the data.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>, Zhian N. Kamvar <zkamvar@gmail.com>

See Also

estimate_peak() for bootstrap estimates of the peak time

Examples

```
if (require(outbreaks) && require(ggplot2)) { withAutoprint({
  i <- incidence(fluH7N9_china_2013$date_of_onset)
  i
  plot(i)

  ## one simple bootstrap
  x <- bootstrap(i)
  x
  plot(x)

  ## find 95% CI for peak time using bootstrap
  find_peak(i)


  ## show confidence interval
  plot(i) + geom_vline(xintercept = find_peak(i), col = "red", lty = 2)

})}
```

---

fit *Fit exponential models to incidence data*

Description

The function fit fits two exponential models to incidence data, of the form: $log(y) = r * t + b$ where 'y' is the incidence, 't' is time (in days), 'r' is the growth rate, and 'b' is the origin. The function fit will fit one model by default, but will fit two models on either side of a splitting date (typically the peak of the epidemic) if the argument split is provided. When groups are present, these are included in the model as main effects and interactions with dates. The function fit_optim_split() can be used to find the optimal 'splitting' date, defined as the one for which the best average R2 of the two models is obtained. Plotting can be done using plot, or added to an existing incidence plot by the piping-friendly function add_incidence_fit().

Usage

```
fit(x, split = NULL, level = 0.95, quiet = FALSE)

fit_optim_split(
```

```
  x,
  window = x$timespan/4,
  plot = TRUE,
  quiet = TRUE,
  separate_split = TRUE
)

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

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

Arguments

x	An incidence object, generated by the function incidence(). For the plotting function, an incidence_fit object.
split	An optional time point identifying the separation between the two models. If NULL, a single model is fitted. If provided, two models would be fitted on the time periods on either side of the split.
level	The confidence interval to be used for predictions; defaults to 95%.
quiet	A logical indicating if warnings from fit should be hidden; FALSE by default. Warnings typically indicate some zero incidence, which are removed before performing the log-linear regression.
window	The size, in days, of the time window either side of the split.
plot	A logical indicating whether a plot should be added to the output (TRUE, default), showing the mean R2 for various splits.
separate_split	If groups are present, should separate split dates be determined for each group? Defaults to TRUE, in which separate split dates and thus, separate models will be constructed for each group. When FALSE, the split date will be determined from the pooled data and modelled with the groups as main effects and interactions with date.
...	currently unused.

Value

For fit(), a list with the class incidence_fit (for a single model), or a list containing two incidence_fit objects (when fitting two models). incidence_fit objects contain:

- $model: the fitted linear model
- $info: a list containing various information extracted from the model (detailed further)
- $origin: the date corresponding to day '0'

The $info item is a list containing:

- r: the growth rate
- r.conf: the confidence interval of 'r'

- pred: a data.frame containing predictions of the model, including the true dates (dates), their numeric version used in the model (dates.x), the predicted value (fit), and the lower (lwr) and upper (upr) bounds of the associated confidence interval.
- doubling: the predicted doubling time in days; exists only if 'r' is positive
- doubling.conf: the confidence interval of the doubling time
- halving: the predicted halving time in days; exists only if 'r' is negative
- halving.conf: the confidence interval of the halving time

For fit_optim_split, a list containing:

- df: a data.frame of dates that were used in the optimization procedure, and the corresponding average R2 of the resulting models.
- split: the optimal splitting date
- fit: an incidence_fit_list object containing the fit for each split. If the separate_split = TRUE, then the incidence_fit_list object will contain these splits nested within each group. All of the incidence_fit objects can be retrieved with get_fit().
- plot: a plot showing the content of df (ggplot2 object)

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>, Zhian N. Kamvar <zkamvar@gmail.com>.

See Also

the incidence() function to generate the 'incidence' objects. The get_fit() function to flatten incidence_fit_list objects to a list of incidence_fit objects.

Examples

```
if (require(outbreaks)) { withAutoprint({
 dat <- ebola_sim$linelist$date_of_onset

 ## EXAMPLE WITH A SINGLE MODEL

 ## compute weekly incidence
 i.7 <- incidence(dat, interval=7)
 plot(i.7)
 plot(i.7[1:20])

 ## fit a model on the first 20 weeks
 f <- fit(i.7[1:20])
 f
 names(f)
 head(get_info(f, "pred"))

 ## plot model alone (not recommended)
 plot(f)

 ## plot data and model (recommended)
```

```
 plot(i.7, fit = f)
 plot(i.7[1:25], fit = f)

## piping versions
if (require(magrittr)) { withAutoprint({
  plot(i.7) %>% add_incidence_fit(f)


  ## EXAMPLE WITH 2 PHASES
  ## specifying the peak manually
  f2 <- fit(i.7, split = as.Date("2014-10-15"))
  f2
  plot(i.7) %>% add_incidence_fit(f2)

  ## finding the best 'peak' date
  f3 <- fit_optim_split(i.7)
  f3
  plot(i.7) %>% add_incidence_fit(f3$fit)
})}

})}
```

`get_counts` *Get counts from an incidence object*

Description

Get counts from an incidence object

Usage

```
get_counts(x, groups = NULL)

## S3 method for class 'incidence'
get_counts(x, groups = NULL)
```

Arguments

`x`	an `incidence` object.
`groups`	if there are groups, use this to specify a group or groups to subset. Defaults to `NULL` indicating that all groups are returned.

Value

a matrix of counts where each row represents a date bin

Examples

```
if (require(outbreaks)) { withAutoprint({
  dat  <- ebola_sim$linelist$date_of_onset
  gend <- ebola_sim$linelist$gender
  i    <- incidence(dat, interval = "week", groups = gend)

  ## Use with an object and no arguments gives the counts matrix
  head(get_counts(i))

  ## Specifying a position or group name will return a matrix subset to that
  ## group
  head(get_counts(i, 1L))
  head(get_counts(i, "f"))

  ## Specifying multiple groups allows you to rearrange columns
  head(get_counts(i, c("m", "f")))

  ## If you want a vector, you can use drop
  drop(get_counts(i, "f"))
})}
```

`get_dates` *Retrieve dates from an incidence object*

Description

Retrieve dates from an incidence object

Usage

```
get_dates(x, ...)

## Default S3 method:
get_dates(x, ...)

## S3 method for class 'incidence'
get_dates(x, position = "left", count_days = FALSE, ...)
```

Arguments

`x`	an incidence object
`...`	Unused
`position`	One of "left", "center", "middle", or "right" specifying what side of the bin the date should be drawn from.
`count_days`	If `TRUE`, the result will be represented as the number of days from the first date.

Value

a vector of dates or numerics

Examples

```
set.seed(999)
dat <- as.Date(Sys.Date()) + sample(-3:50, 100, replace = TRUE)
x <- incidence(dat, interval = "month")
get_dates(x)
get_dates(x, position = "middle")
set.seed(999)
dat <- as.Date(Sys.Date()) + sample(-3:50, 100, replace = TRUE)
x <- incidence(dat, interval = "month")
get_dates(x)
get_dates(x, "center")
get_dates(x, "right")

# Return dates by number of days from the first date
get_dates(x, count_days = TRUE)
get_dates(incidence(-5:5), count_days = TRUE)
```

---

get_fit *Accessors for* incidence_fit *objects*

Description

Accessors for incidence_fit objects

Usage

```
get_fit(x)

## S3 method for class 'incidence_fit'
get_fit(x)

## S3 method for class 'incidence_fit_list'
get_fit(x)

get_info(x, what = "r", ...)

## S3 method for class 'incidence_fit'
get_info(x, what = "r", ...)

## S3 method for class 'incidence_fit_list'
get_info(x, what = "r", groups = NULL, na.rm = TRUE, ...)
```

Arguments

x	an incidence_fit or incidence_fit_list object.
what	the name of the item in the "info" element of the incidence_fit object.
...	currently unused.
groups	if what = "pred" and x is an incidence_fit_list object, then this indicates what part of the nesting hierarchy becomes the column named "groups". Defaults to NULL, indicating that no groups column will be added/modified.
na.rm	when TRUE (default), missing values will be excluded from the results.

Value

a list of incidence_fit objects.

Examples

```
if (require(outbreaks)) { withAutoprint({

 dat <- ebola_sim$linelist$date_of_onset

 ## EXAMPLE WITH A SINGLE MODEL

 ## compute weekly incidence
 sex <- ebola_sim$linelist$gender
 i.sex <- incidence(dat, interval = 7, group = sex)

 ## Compute the optimal split for each group separately
 fits  <- fit_optim_split(i.sex, separate_split = TRUE)

 ## `fits` contains an `incidence_fit_list` object
 fits$fit

 ## Grab the list of `incidence_fit` objects
 get_fit(fits$fit)

 ## Get the predictions for all groups
 get_info(fits$fit, "pred", groups = 1)

 ## Get the predictions, but set `groups` to "before" and "after"
 get_info(fits$fit, "pred", groups = 2)

 ## Get the reproduction number
 get_info(fits$fit, "r")

 ## Get the doubling confidence interval
 get_info(fits$fit, "doubling.conf")

 ## Get the halving confidence interval
 get_info(fits$fit, "halving.conf")
})}
```

group_names *extract and set group names*

Description

extract and set group names

Usage

```
group_names(x, value)

group_names(x) <- value

## Default S3 method:
group_names(x, value)

## Default S3 replacement method:
group_names(x) <- value

## S3 method for class 'incidence'
group_names(x, value = NULL)

## S3 replacement method for class 'incidence'
group_names(x) <- value
```

Arguments

x	an incidence() object.
value	character vector used to rename groups

Details

This accessor will return a

Value

an integer indicating the number of groups present in the incidence object.

Examples

```
i <- incidence(dates = sample(10, 100, replace = TRUE),
               interval = 1L,
               groups = sample(letters[1:3], 100, replace = TRUE))
i
group_names(i)

# change the names of the groups
group_names(i) <- c("Group 1", "Group 2", "Group 3")
```

```
i

# example if there are mistakes in the original data, e.g.
# something is misspelled
set.seed(50)
grps <- sample(c("child", "adult", "adlut"), 100, replace = TRUE, prob = c(0.45, 0.45, 0.05))
i <- incidence(dates = sample(10, 100, replace = TRUE),
               interval = 1L,
               groups = grps)
colSums(get_counts(i))

# If you change the name of the mis-spelled group, it will be merged with the
# correctly-spelled group
gname <- group_names(i)
gname[gname == "adlut"] <- "adult"
# without side-effects
print(ii <- group_names(i, gname))
colSums(get_counts(i))  # original still has three groups
colSums(get_counts(ii))
# with side-effects
group_names(i) <- gname
colSums(get_counts(i))
```

`incidence` *Compute incidence of events from a vector of dates.*

Description

This function computes incidence based on dates of events provided in various formats. A fixed interval, provided as numbers of days, is used to define time intervals. Counts within an interval always include the first date, after which they are labeled, and exclude the second. For instance, intervals labeled as 0, 3, 6, ... mean that the first bin includes days 0, 1 and 2, the second interval includes 3, 4 and 5 etc.

Usage

```
incidence(dates, interval = 1L, ...)

## Default S3 method:
incidence(dates, interval = 1L, ...)

## S3 method for class 'Date'
incidence(
  dates,
  interval = 1L,
  standard = TRUE,
  groups = NULL,
  na_as_group = TRUE,
  first_date = NULL,
```

```
  last_date = NULL,
  ...
)

## S3 method for class 'character'
incidence(
  dates,
  interval = 1L,
  standard = TRUE,
  groups = NULL,
  na_as_group = TRUE,
  first_date = NULL,
  last_date = NULL,
  ...
)

## S3 method for class 'integer'
incidence(
  dates,
  interval = 1L,
  groups = NULL,
  na_as_group = TRUE,
  first_date = NULL,
  last_date = NULL,
  ...
)

## S3 method for class 'numeric'
incidence(
  dates,
  interval = 1L,
  groups = NULL,
  na_as_group = TRUE,
  first_date = NULL,
  last_date = NULL,
  ...
)

## S3 method for class 'POSIXt'
incidence(
  dates,
  interval = 1L,
  standard = TRUE,
  groups = NULL,
  na_as_group = TRUE,
  first_date = NULL,
  last_date = NULL,
  ...
```

```
)

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

Arguments

dates	A vector of dates, which can be provided as objects of the class: integer, numeric, Date, POSIXct, POSIXlt, and character. (See Note about numeric and character formats)
interval	An integer or character indicating the (fixed) size of the time interval used for computing the incidence; defaults to 1 day. This can also be a text string that corresponds to a valid date interval: day, week, month, quarter, or year. (See Note).
...	Additional arguments passed to other methods (none are used).
standard	(Only applicable to Date objects) When TRUE (default) and the interval one of "week", "month", "quarter", or "year", then this will cause the bins for the counts to start at the beginning of the interval (See Note).
groups	An optional factor defining groups of observations for which incidence should be computed separately.
na_as_group	A logical value indicating if missing group (NA) should be treated as a separate group.
first_date, last_date	optional first/last dates to be used in the epicurve. When these are NULL (default), the dates from the first/last dates are taken from the observations. If these dates are provided, the observations will be trimmed to the range of [first_date, last_date].
x	An 'incidence' object.

Details

For details about the incidence class, see the dedicated vignette:
`vignette("incidence_class", package = "incidence")`

Value

An list with the class incidence, which contains the following items:

- **dates**: The dates marking the left side of the bins used for counting events. When standard = TRUE and the interval represents weeks, months, quarters, or years, the first date will represent the first standard date (See Interval specification, below).
- **counts**: A matrix of incidence counts, which one column per group (and a single column if no groups were used).
- **timespan**: The length of the period for which incidence is computed, in days.
- **interval**: The bin size. If it's an integer, it represents the number of days between each bin. It can also be a character, e.g. "2 weeks" or "6 months".

- **n**: The total number of cases.
- **weeks**: Dates in week format (YYYY-Www), where YYYY corresponds to the year of the given week and ww represents the numeric week of the year. This will be a produced from the function `aweek::date2week()`. Note that these will have a special `"week_start"` attribute indicating which day of the ISO week the week starts on (see Weeks, below).
- **isoweeks**: ISO 8601 week format YYYY-Www, which is returned only when ISO week-based weekly incidence is computed.

Note

Input data (`dates`):

- **Decimal (numeric) dates**: will be truncated with a warning
- **Character dates** should be in the unambiguous `yyyy-mm-dd` (ISO 8601) format. Any other format will trigger an error.

Interval specification (`interval`): If `interval` is a valid character (e.g. "week" or "1 month"), then the bin will start at the beginning of the interval just before the first observation by default. For example, if the first case was recorded on Wednesday, 2018-05-09:

- "week" : first day of the week (i.e. Monday, 2018-05-07) (defaults to ISO weeks, see "Week intervals", below)
- "month" : first day of the month (i.e. 2018-05-01)
- "quarter" : first day of the quarter (i.e. 2018-04-01)
- "year" : first day of the calendar year (i.e. 2018-01-01)

These default intervals can be overridden with `standard = FALSE`, which sets the interval to begin at the first observed case.

Week intervals:

As of *incidence* version 1.7.0, it is possible to construct standardized incidence objects standardized to any day of the week thanks to the `aweek::date2week()` function from the **aweek** package. The default state is to use ISO 8601 definition of weeks, which start on Monday. You can specify the day of the week an incidence object should be standardised to by using the pattern "{n} {W} weeks" where "{W}" represents the weekday in an English or current locale and "{n}" represents the duration, but this can be ommitted. Below are examples of specifying weeks starting on different days assuming we had data that started on 2016-09-05, which is ISO week 36 of 2016:

- interval = "2 monday weeks" (Monday 2016-09-05)
- interval = "1 tue week" (Tuesday 2016-08-30)
- interval = "1 Wed week" (Wednesday 2016-08-31)
- interval = "1 Thursday week" (Thursday 2016-09-01)
- interval = "1 F week" (Friday 2016-09-02)
- interval = "1 Saturday week" (Saturday 2016-09-03)
- interval = "Sunday week" (Sunday 2016-09-04)

It's also possible to use something like "3 weeks: Saturday"; In addition, there are keywords reserved for specific days of the week:

- interval = "week", standard = TRUE (Default, Monday)
- interval = "ISOweek" (Monday)

- interval = "EPIweek" (Sunday)
- interval = "MMWRweek" (Sunday)

The "EPIweek" specification is not strictly reserved for CDC epiweeks, but can be prefixed (or posfixed) by a day of the week: "1 epiweek: Saturday".

The `first_date` argument: Previous versions of *incidence* had the `first_date` argument override `standard = TRUE`. It has been changed as of *incidence* version 1.6.0 to be more consistent with the behavior when `first_date = NULL`. This, however may be a change in behaviour, so a warning is now issued once and only once if `first_date` is specified, but `standard` is not. To never see this warning, use `options(incidence.warn.first_date = FALSE)`.

The intervals for "month", "quarter", and "year" will necessarily vary in the number of days they encompass and warnings will be generated when the first date falls outside of a calendar date that is easily represented across the interval.

Author(s)

Thibaut Jombart, Rich Fitzjohn, Zhian Kamvar

See Also

The main other functions of the package include:

- `plot.incidence()`: Plot epicurves from an incidence object.
- `fit()`: Fit log-linear model to computed incidence.
- `fit_optim_split()`: Find the optimal peak of the epidemic and fits log-linear models on either side of the peak.
- `subset()`: Handling of `incidence` objects.
- `pool()`: Sum incidence over groups.
- `as.data.frame.incidence()`: Convert an `incidence` object to a `data.frame`.

The following vignettes are also available:

- `overview`: Provides an overview of the package's features.
- `customize_plot`: Provides some tips on finer plot customization.
- `incidence_class`: Details the content of the `incidence` class.

Examples

```
## toy example
incidence(c(1, 5, 8, 3, 7, 2, 4, 6, 9, 2))
incidence(c(1, 5, 8, 3, 7, 2, 4, 6, 9, 2), 2)

## example using simulated dataset
if(require(outbreaks)) { withAutoprint({
  onset <- outbreaks::ebola_sim$linelist$date_of_onset

  ## daily incidence
  inc <- incidence(onset)
```

```
  inc
  plot(inc)

  ## weekly incidence
  inc.week <- incidence(onset, interval = 7, standard = FALSE)
  inc.week
  plot(inc.week)
  plot(inc.week, border = "white") # with visible border

  # Starting on Monday
  inc.isoweek <- incidence(onset, interval = "isoweek")
  inc.isoweek

  # Starting on Sunday
  inc.epiweek <- incidence(onset, interval = "epiweek")
  inc.epiweek

  # Starting on Saturday
  inc.epiweek <- incidence(onset, interval = "saturday epiweek")
  inc.epiweek

  ## use group information
  sex <- outbreaks::ebola_sim$linelist$gender
  inc.week.gender <- incidence(onset, interval = 7,
                               groups = sex, standard = FALSE)
  inc.week.gender
  head(inc.week.gender$counts)
  plot(inc.week.gender, border = "grey90")
  inc.satweek.gender <- incidence(onset, interval = "2 epiweeks: saturday",
                                  groups = sex)
  inc.satweek.gender
  plot(inc.satweek.gender, border = "grey90")

})}

# Use of first_date
d <- Sys.Date() + sample(-3:10, 10, replace = TRUE)

# `standard` specified, no warning
di <- incidence(d, interval = "week", first_date = Sys.Date() - 10, standard = TRUE)

# warning issued if `standard` not specified
di <- incidence(d, interval = "week", first_date = Sys.Date() - 10)

# second instance: no warning issued
di <- incidence(d, interval = "week", first_date = Sys.Date() - 10)
```

---

incidence_pal1 *Color palettes used in incidence*

Description

These functions are color palettes used in incidence.

Usage

```
incidence_pal1(n)

incidence_pal1_light(n)

incidence_pal1_dark(n)
```

Arguments

n	a number of colors

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

Examples

```
plot(1:4, cex=8, pch=20, col = incidence_pal1(4),
     main = "palette: incidence_pal1")
plot(1:100, cex=8, pch=20, col = incidence_pal1(100),
     main ="palette: incidence_pal1")
plot(1:100, cex=8, pch=20, col = incidence_pal1_light(100),
     main="palette: incidence_pal1_light")
plot(1:100, cex=8, pch=20, col = incidence_pal1_dark(100),
     main="palette: incidence_pal1_dark")
```

---

plot.incidence *Plot function for incidence objects*

Description

This function is used to visualise the output of the `incidence()` function using the package `ggplot2`.
#'

Usage

```
## S3 method for class 'incidence'
plot(
  x,
  ...,
  fit = NULL,
  stack = is.null(fit),
  color = "black",
```

```
  border = NA,
  col_pal = incidence_pal1,
  alpha = 0.7,
  xlab = "",
  ylab = NULL,
  labels_week = !is.null(x$weeks),
  labels_iso = !is.null(x$isoweeks),
  show_cases = FALSE,
  n_breaks = 6
)

add_incidence_fit(p, x, col_pal = incidence_pal1)

## S3 method for class 'incidence_fit'
plot(x, ...)

## S3 method for class 'incidence_fit_list'
plot(x, ...)

scale_x_incidence(x, n_breaks = 6, labels_week = TRUE, ...)

make_breaks(x, n_breaks = 6L, labels_week = TRUE)
```

Arguments

x	An incidence object, generated by the function incidence().
...	arguments passed to ggplot2::scale_x_date(), ggplot2::scale_x_datetime(), or ggplot2::scale_x_continuous(), depending on how the $date element is stored in the incidence object.
fit	An 'incidence_fit' object as returned by fit().
stack	A logical indicating if bars of multiple groups should be stacked, or displayed side-by-side.
color	The color to be used for the filling of the bars; NA for invisible bars; defaults to "black".
border	The color to be used for the borders of the bars; NA for invisible borders; defaults to NA.
col_pal	The color palette to be used for the groups; defaults to incidence_pal1. See incidence_pal1() for other palettes implemented in incidence.
alpha	The alpha level for color transparency, with 1 being fully opaque and 0 fully transparent; defaults to 0.7.
xlab	The label to be used for the x-axis; empty by default.
ylab	The label to be used for the y-axis; by default, a label will be generated automatically according to the time interval used in incidence computation.
labels_week	a logical value indicating whether labels x axis tick marks are in week format YYYY-Www when plotting weekly incidence; defaults to TRUE.

labels_iso	(deprecated) This has been superceded by labels_iso. Previously:a logical value indicating whether labels x axis tick marks are in ISO 8601 week format yyyy-Www when plotting ISO week-based weekly incidence; defaults to be TRUE.
show_cases	if TRUE (default: FALSE), then each observation will be colored by a border. The border defaults to a white border unless specified otherwise. This is normally used outbreaks with a small number of cases. Note: this can only be used if stack = TRUE
n_breaks	the ideal number of breaks to be used for the x-axis labeling
p	An existing incidence plot.

Details

- plot() will visualise an incidence object using ggplot2
- make_breaks() calculates breaks from an incidence object that always align with the bins and start on the first observed incidence.
- scale_x_incidence() produces and appropriate ggplot2 scale based on an incidence object.

Value

- plot() a ggplot2::ggplot() object.
- make_breaks() a two-element list. The "breaks" element will contain the evenly-spaced breaks as either dates or numbers and the "labels" element will contain either a vector of weeks OR a ggplot2::waiver() object.
- scale_x_incidence() a **ggplot2** "ScaleContinuous" object.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com> Zhian N. Kamvar <zkamvar@gmail.com>

See Also

The incidence() function to generate the 'incidence' objects.

Examples

```
if(require(outbreaks) && require(ggplot2)) { withAutoprint({
  onset <- outbreaks::ebola_sim$linelist$date_of_onset

  ## daily incidence
  inc <- incidence(onset)
  inc
  plot(inc)

  ## weekly incidence
  inc.week <- incidence(onset, interval = 7)
  inc.week
  plot(inc.week) # default to label x axis tick marks with isoweeks
```

```
  plot(inc.week, labels_week = FALSE) # label x axis tick marks with dates
  plot(inc.week, border = "white") # with visible border

  ## use group information
  sex <- outbreaks::ebola_sim$linelist$gender
  inc.week.gender <- incidence(onset, interval = "1 epiweek", groups = sex)
  plot(inc.week.gender)
  plot(inc.week.gender, labels_week = FALSE)

  ## show individual cases at the beginning of the epidemic
  inc.week.8 <- subset(inc.week.gender, to = "2014-06-01")
  p <- plot(inc.week.8, show_cases = TRUE, border = "black")
  p

  ## update the range of the scale
  lim <- c(min(get_dates(inc.week.8)) - 7*5,
           aweek::week2date("2014-W50", "Sunday"))
  lim
  p + scale_x_incidence(inc.week.gender, limits = lim)

  ## customize plot with ggplot2
  plot(inc.week.8, show_cases = TRUE, border = "black") +
    theme_classic(base_size = 16) +
    theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5))

  ## adding fit
  fit <- fit_optim_split(inc.week.gender)$fit
  plot(inc.week.gender, fit = fit)
  plot(inc.week.gender, fit = fit, labels_week = FALSE)

})}
```

`pool` *Pool 'incidence' across groups*

Description

This function pools incidence across all groups of an `incidence` object. The resulting `incidence()` object will contains counts summed over all groups present in the input.

Usage

```
pool(x)
```

Arguments

`x`	An 'incidence' object.

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

See Also

The incidence() function to generate the 'incidence' objects.

Examples

```
dat <- as.integer(c(0,1,2,2,3,5,7))
group <- factor(c(1, 2, 3, 3, 3, 3, 1))
i <- incidence(dat, groups = group)
i
i$counts

## pool all groups
pool(i)
pool(i)$counts

## pool only groups 1 and 3
pool(i[,c(1,3)])
pool(i[,c(1,3)])$counts
```

---

subset.incidence *Subsetting 'incidence' objects*

Description

Two functions can be used to subset incidence objects. The function subset permits to retain dates within a specified range and, optionally, specific groups. 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 groups.

Usage

```
## S3 method for class 'incidence'
subset(x, ..., from = min(x$dates), to = max(x$dates), groups = TRUE)

## S3 method for class 'incidence'
x[i, j]
```

Arguments

x	An incidence object, generated by the function incidence().
...	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.

groups	(optional) The groups to retained, indicated as subsets of the columns of x$counts.
i	a subset of dates to retain
j	a subset of groups to retain

Author(s)

Thibaut Jombart <thibautjombart@gmail.com>

See Also

The incidence() function to generate the 'incidence' objects.

Examples

```
## example using simulated dataset
if(require(outbreaks)) { withAutoprint({
  onset <- ebola_sim$linelist$date_of_onset

  ## weekly incidence
  inc <- incidence(onset, interval = 7)
  inc
  inc[1:10] # first 10 weeks
  plot(inc[1:10])
  inc[-c(11:15)] # remove weeks 11-15
  plot(inc[-c(11:15)])
})}
```

Index