Package 'WhiteLabRt'

July 31, 2024

Type Package

Title Novel Methods for Reproduction Number Estimation, Back-Calculation, and Forecasting

Version 1.0

Date 2024-07-23

Maintainer Chad Milando <cmilando@bu.edu>

Description A collection of functions related to novel methods for estimating R(t), created by the lab of Professor Laura White. Currently implemented methods include two-step Bayesian back-calculation and now-casting for line-list data with missing reporting delays, adapted in 'STAN' from Li (2021) <doi:10.1371/journal.pcbi.1009210>, and calculation of time-varying reproduction number assuming a flux between various adjacent states, adapted into 'STAN' from Zhou (2021) <doi:10.1371/journal.pcbi.1010434>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Biarch true

Depends R (>= 3.4.0)

Imports methods, Rcpp (>= 0.12.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), rstantools (>= 2.4.0)

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), RcppParallel (>= 5.0.1), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make

Suggests knitr, rmarkdown

Language en-US

LazyData true

VignetteBuilder knitr

NeedsCompilation yes

Author Chad Milando [aut, cre],
Tenglong Li [ctb],
Zhenwei Zhou [ctb],
Laura White [ctb]

Repository CRAN

Date/Publication 2024-07-31 10:10:28 UTC

Contents

`WhiteLabRt-package` *The 'WhiteLabRt' package.*

Description

A collection of functions related to novel methods for estimating reproduction number, R(t), created by the lab of Professor Laura White at Boston University School of Public Health.

Currently implemented methods include (1) Temporal R(t) estimation: Two-step Bayesian back and nowcasting for linelist data with missing reporting delays, adapted in 'STAN' from Li et. al. 2021, and (2) Spatial R(t) estimation: Calculating time-varying reproduction number, R(t), assuming a flux of infectors between various adjacent states, in 'STAN' from Zhou et. al. 2021.

Author(s)

Maintainer: Chad Milando <cmilando@bu.edu>

Other contributors:

- Tenglong Li [contributor]
- Zhenwei Zhou [contributor]
- Laura White [contributor]

References

Stan Development Team (NA). RStan: the R interface to Stan. R package version 2.32.6. https://mc-stan.org

---

`convert_to_linelist` *Convert Case Counts to a Line List*

Description

This function takes a data frame of case counts and expands it into a line list format, which is often used for epidemiological analysis. The function validates input data, manages missingness, and assumes additional generation times based on the specified reporting function.

Usage

```
convert_to_linelist(
  caseCounts,
  reportF_missP,
  reportF = NULL,
  reportF_args = NULL
)
```

Arguments

`caseCounts`	A data frame with columns `date`, `cases`, and `location`. The data frame must meet several criteria: - It should only contain data for one location. - Dates must be in Date format. - Case numbers must be non-negative integers. - No missing values are allowed in the necessary columns.
`reportF_missP`	A numeric probability between 0 and 1 (exclusive) indicating the proportion of missing onset dates. It throws an error if it is out of bounds or not numeric.
`reportF`	A function used to simulate the delay from case reporting to case onset. Defaults to a negative binomial distribution function (`rnbinom`) if NULL.
`reportF_args`	A list of additional arguments to pass to `reportF`. Defaults to `list(size = 3, mu = 9)` when `reportF` is NULL.

Details

The function stops and sends error messages for various data integrity issues, such as incorrect data types, negative cases, or missing required columns. It also assumes that the input data is for only one location and handles NA generation according to `reportF_missP`.

Value

A data frame in line list format, where each row corresponds to a case report. The data frame includes columns for the report date, the delay from report to onset, the onset date, weekend indicator, report interval in days from the first report, and week interval. The returned data frame has additional attributes set, including `min_day` and the class `lineList`.

Examples

```
data("sample_dates")
data("sample_location")
data("sample_cases")
case_Counts <- create_caseCounts(sample_dates, sample_location, sample_cases)
line_list <- convert_to_linelist(case_Counts, reportF_missP = 0.5)
```

`create_caseCounts` *Create a Case Counts Data Frame*

Description

This function constructs a data frame from vectors representing dates, locations, and case numbers, ensuring that all input vectors meet specific data integrity requirements. It checks for the correct data types, non-negative case numbers, and uniformity in vector lengths. The function also ensures no missing values are present and that all data pertain to a single location.

Usage

```
create_caseCounts(date_vec, location_vec, cases_vec)
```

Arguments

`date_vec`	A vector of dates corresponding to case reports; must be of type Date.
`location_vec`	A character vector representing the location of the case reports; all entries must refer to the same location.
`cases_vec`	A numeric vector representing the number of cases reported on each date; values must be non-negative integers.

Details

The function performs several checks to ensure the integrity of the input: - It verifies that all vectors have the same length. - It confirms that there are no negative numbers in `cases_vec`. - It checks for and disallows any missing values in the data frame. It throws errors if any of these conditions are not met, indicating that the input vectors are not appropriately formatted or contain invalid data.

Value

A data frame named `caseCounts` with columns `date`, `cases`, and `location`. Each row corresponds to a unique report of cases on a given date at a specified location. The data frame is assigned a class attribute of `caseCounts`.

Examples

```
data("sample_dates")
data("sample_location")
data("sample_cases")
case_Counts = create_caseCounts(sample_dates, sample_location, sample_cases)
```

`create_linelist` *Create a Line List from Report and Onset Dates*

Description

This function constructs a line list data frame using vectors of report and onset dates.

Usage

```
create_linelist(report_dates, onset_dates)
```

Arguments

`report_dates`	A vector of dates representing when cases were reported; must be of type Date.
`onset_dates`	A vector of dates representing when symptoms onset occurred; must be of type Date. This vector can contain NA values, but not exclusively or none at all.

Details

The function ensures the following: - The length of `report_dates` and `onset_dates` must be equal. - There should be no NA values in `report_dates`. - `onset_dates` must contain some but not all NA values. - Each non-NA onset date must be earlier than or equal to its corresponding report date. If any of these conditions are violated, the function will stop with an error message. Additionally, the function calculates the delay in days between onset and report dates, identifies weekends, and calculates reporting and week intervals based on the earliest date.

Value

A data frame with the following columns: report_dates, delay_int, onset_dates, is_weekend, report_int, and week_int. This data frame is ordered by report_dates and assigned a class attribute of `lineList`.

Examples

```
data("sample_onset_dates")
data("sample_report_dates")
line_list <- create_linelist(sample_report_dates, sample_onset_dates)
```

out_list_demo *Sample back-calculation output*

Description

an example backnow output object from run_backnow()

Usage

```
out_list_demo
```

Format

An backnow object

list list

plot.backnow *Plot Estimates or Reproduction Numbers*

Description

This function plots estimates of case numbers or reproduction numbers (r(t)) based on the provided object. It can handle two types of plots: 'est' for estimated case numbers over time, and 'rt' for estimated reproduction numbers over time.

Usage

```
## S3 method for class 'backnow'
plot(x, plottype, ...)
```

Arguments

x	An object containing the necessary data for plotting. This object should have specific structure depending on the plottype: - For plottype = 'est', x should contain report_date, report_cases, est_back_date, and est_back, where est_back is expected to be a matrix with three rows representing the lower bound, estimate, and upper bound. - For plottype = 'rt', x should contain est_rt_date and est_rt, with est_rt formatted similarly to est_back.
plottype	A character string specifying the type of plot to generate. Valid options are 'est' for case estimates and 'rt' for reproduction numbers.
...	Additional arguments passed to the plot function.

Details

Depending on the plottype: - 'est': Plots the reported cases over time with a polygon representing the uncertainty interval and a line showing the central estimate. - 'rt': Plots the reproduction number over time with a similar style.

Value

a plot object for an object of class backnow

Examples

```
data("sample_onset_dates")
data("sample_report_dates")
line_list <- create_linelist(sample_report_dates, sample_onset_dates)
sip <- si(14, 4.29, 1.18)
results <- run_backnow(
  line_list,
  sip = sip, chains = 1)
plot(results, "est")
```

---

plot.caseCounts *Plot Case Counts Over Time*

Description

This function plots the number of cases over time from a data frame object. If the data frame contains multiple locations, a specific location must be specified. The plot displays the total number of cases against dates and annotates one of the earliest points with the location name.

Usage

```
## S3 method for class 'caseCounts'
plot(x, loc = NULL, ...)
```

Arguments

x	A data frame containing the case counts with at least two columns: date and cases. The data frame may optionally include a location column, which is required if multiple locations are present.
loc	An optional string specifying the location to filter the case counts by. If loc is provided and location column exists in x, the plot will only show data for the specified location. If multiple locations are present and loc is not specified, the function will stop with an error.
...	Additional arguments passed to the plot function.

Details

If the `location` column is present in `x` and contains multiple unique values, the `loc` parameter must be specified to indicate which location's data to plot. The function adds a text annotation to the plot, labeling one of the earliest points with the specified location's name.

Value

a plot object for an object of class `caseCounts`

Examples

```
data("sample_dates")
data("sample_location")
data("sample_cases")
case_Counts = create_caseCounts(sample_dates, sample_location, sample_cases)
plot(case_Counts)
```

`run_backnow` *Run Back Calculation and Estimate Reproduction Numbers*

Description

This function performs a back-calculation based on provided epidemic case count data, estimating the time distribution of infections and reproduction numbers (r(t)). It utilizes extensive input checks and parameter validation to ensure robust model execution.

Usage

```
run_backnow(
  input,
  sip,
  NB_maxdelay = as.integer(20),
  window_size = as.integer(7),
  ...
)
```

Arguments

`input`	A 'lineList' data.frame from `create_linelist` or `convert_to_linelist`.
`sip`	Vector of numeric values specifying the serial interval probabilities.
`NB_maxdelay`	Integer, the maximum delay for the negative binomial distribution used in modeling.
`window_size`	Integer, the number of days of the R(t) averaging window.
`...`	Additional arguments passed to rstan::sampling()

Details

The function ensures input data is of the correct class and processes it accordingly. It handles different input classes by either converting `caseCounts` to `lineList` or directly using `lineList`. The function stops with an error if the input doesn't meet expected standards. It performs simulations to estimate both the back-calculation of initial infections and reproduction numbers over time, while checking and adjusting for potential NA values and ensuring that all conditions for the model parameters are met. Output includes estimates of initial infections and reproduction numbers along with diagnostic statistics.

Value

an object of class `backnow`

Examples

```
data("sample_onset_dates")
data("sample_report_dates")
line_list <- create_linelist(sample_report_dates, sample_onset_dates)
sip <- si(14, 4.29, 1.18)
results <- run_backnow(
  line_list,
  sip = sip, chains = 1)
```

`sample_cases` *Sample cases*

Description

Sample of aggregated case counts from a single location.

Usage

```
sample_cases
```

Format

A vector of length 80.

Cases Numeric

sample_dates *Sample dates*

Description

Sample of case report dates from a single location.

Usage

```
sample_dates
```

Format

A vector of length 80.

Dates Dates of reported aggregated cases

sample_location *Sample location*

Description

An vector of a single location to accompany `sample_dates` and `sample_cases`. This is to emphasize that linelist functions are for a single location

Usage

```
sample_location
```

Format

A vector of length 80.

Location Character

sample_multi_site *Sample multi site*

Description

A matrix of daily aggregated cases from two sites (Hoth and Tatooine).

Usage

```
sample_multi_site
```

Format

A data.table data frame with 80 rows and 2 variables:

Cases for Site1 Number of aggregated cases for Site 1

Cases for Site1 Number of aggregated cases for Site 2

sample_m_hier *Sample hierachical model output*

Description

an example `rstan` output object from spatialrt()

Usage

```
sample_m_hier
```

Format

An rstan object

list list

sample_onset_dates *Sample onset dates*

Description

Line-list data, onset dates, with some missing.

Usage

```
sample_onset_dates
```

Format

A vector of length 6380, one value per case.

Date Date of onset

sample_report_dates *Sample report dates*

Description

Line-list data, case report dates.

Usage

```
sample_report_dates
```

Format

A vector of length 6380, one value per case.

Date Date of report

si *Calculate a Serial Interval Distribution*

Description

This function computes the probability distribution function (PDF) of the serial interval using a gamma distribution with specified shape and rate parameters. The serial interval is defined as the time between successive cases in a chain of transmission. This implementation generates a discrete PDF at the *daily* level.

Usage

```
si(ndays, shape, rate, leading0 = TRUE)
```

Arguments

ndays	Integer, the number of days over which to calculate the serial interval distribution.
shape	Numeric, the shape parameter of the gamma distribution.
rate	Numeric, the rate parameter of the gamma distribution.
leading0	Logical, should a leading 0 be added to indicate t0?

Details

The function uses the pgamma function to calculate cumulative probabilities for each day up to ndays and then differences these to get daily probabilities. The resulting probabilities are normalized to sum to 1, ensuring that they represent a valid probability distribution.

Value

Numeric vector representing the serial interval probabilities for each of the first ndays days. The probabilities are normalized so that their sum is 1.

Examples

```
si(ndays = 14, shape = 4.29, rate = 1.18)
```

spatialRt *Run Spatial R(t) estimation and Estimate Reproduction Numbers*

Description

This function calculates R(t) that arises from transfer of infectors between different states. There are different flavors of the model, but the base version calculates a weekly R(t) within each state.

Usage

```
spatialRt(report_dates, case_matrix, transfer_matrix, sip, v2 = FALSE, ...)
```

Arguments

report_dates	A vector of reporting dates
case_matrix	A matrix of cases, defined by integers
transfer_matrix	A matrix that defines how infectors flow between states. Each row of the transfer matrix must sum to 1.
sip	Vector of numeric values specifying the serial interval probabilities.
v2	a flag indicating FALSE if the base algorithm is to be used, and TRUE if the experimental algorithm is desired. The experimental version contains a non-centered parameterization, an AR1 process, and partial pooling across states.
...	Additional arguments passed to rstan::sampling()

Value

An rstan object.

Examples

```
data("sample_multi_site")
data("transfer_matrix")
Y <- matrix(integer(1), nrow = nrow(sample_multi_site), ncol = 2)
for(i in 1:nrow(Y)) {
  for(j in c(2, 3)) {
    Y[i,j-1] <- as.integer(sample_multi_site[i,j])
  }
}
all(is.integer(Y))
sip <- si(14, 4.29, 1.18, leading0 = FALSE)
sample_m_hier <- spatialRt(report_dates = sample_multi_site$date,
case_matrix = Y,
transfer_matrix = transfer_matrix,
v2 = FALSE,
sip = sip, chains = 1)
```

`transfer_matrix` *Transfer matrix*

Description

A matrix of daily transfers between two sites (Hoth and Tatooine). Each row sums to 1.

Usage

```
transfer_matrix
```

Format

A data.table data frame with 160 rows and 2 variables:

Transfer to Site1 Fraction of transfer to Site 1

Transfer to Site2 Fraction of transfer to Site 2

Index