Package: epidist (via r-universe)

March 7, 2025

Title Estimate Epidemiological Delay Distributions With brms

Version 0.3.0

Description Understanding and accurately estimating epidemiological delay distributions is important for public health policy. These estimates influence epidemic situational awareness, control strategies, and resource allocation. This package provides methods to address the key challenges in estimating these distributions, including truncation, interval censoring, and dynamical biases. These issues are frequently overlooked, resulting in biased conclusions. Built on top of 'brms', it allows for flexible modelling including time-varying spatial components and partially pooled estimates of demographic characteristics.

License MIT + file LICENSE

URL https://epidist.epinowcast.org/, https://github.com/epinowcast/epidist/

BugReports https://github.com/epinowcast/epidist/issues/

Depends R (>= 4.0.0)

Imports brms, checkmate, cli, dplyr, lubridate, primarycensored, purrr, rstan (>= 2.26.0), stats, tibble, tidyr

Suggests bayesplot, BH (>= 1.66.0), bookdown, cmdstanr, CodeDepends, collapse, ggplot2, knitr, loo, marginaleffects, modelr, patchwork, pkgdown, posterior, priorsense, Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), roxyglobals, testthat (>= 3.0.0), tidybayes, usethis

Remotes stan-dev/cmdstanr

Config/Needs/website r-lib/pkgdown, epinowcast/enwtheme

Config/testthat/edition 3

Config/roxyglobals/filename globals.R

Config/roxyglobals/unique FALSE

LazyData true

VignetteBuilder knitr

Encoding UTF-8

Roxygen list(markdown = TRUE, roclets = c(``collate", ``namespace",
``rd", ``roxyglobals::global_roclet"))

RoxygenNote 7.3.2

Config/pak/sysreqs make libicu-dev

Repository https://epinowcast.r-universe.dev

RemoteUrl https://github.com/epinowcast/epidist

RemoteRef v0.3.0

RemoteSha 051d0cf18ef74dbc97e9e6d94b9040b457b7594d

Contents

add_mean_sd *Add natural scale mean and standard deviation parameters*

Description

Add natural scale mean and standard deviation parameters

Usage

```
add_mean_sd(data, ...)
```

Arguments

`data`	A dataframe of distributional parameters.
`...`	Additional arguments for method.

See Also

Other postprocess: `add_mean_sd.default()`, `add_mean_sd.gamma_samples()`, `add_mean_sd.lognormal_samples()`, `add_mean_sd.weibull_samples()`, `predict_delay_parameters()`

add_mean_sd.default *Default method for add natural scale parameters*

Description

Default method for add natural scale parameters

Usage

```
## Default S3 method:
add_mean_sd(data, ...)
```

Arguments

`data`	A dataframe of distributional parameters.
`...`	Additional arguments for method.

See Also

Other postprocess: `add_mean_sd()`, `add_mean_sd.gamma_samples()`, `add_mean_sd.lognormal_samples()`, `add_mean_sd.weibull_samples()`, `predict_delay_parameters()`

add_mean_sd.gamma_samples

Add natural scale mean and standard deviation parameters for a Gamma model

Description

Again, mu and shape here are the distributional parameters of brms.

Usage

```
## S3 method for class 'gamma_samples'
add_mean_sd(data, ...)
```

Arguments

data	A dataframe of distributional parameters.
...	Additional arguments for method.

See Also

Other postprocess: add_mean_sd(), add_mean_sd.default(), add_mean_sd.lognormal_samples(), add_mean_sd.weibull_samples(), predict_delay_parameters()

add_mean_sd.lognormal_samples

Add natural scale mean and standard deviation parameters for a lognormal model

Description

Note that the input parameters here are mu and sigma, corresponding to the distributional parameters used by brms for the lognormal family.

Usage

```
## S3 method for class 'lognormal_samples'
add_mean_sd(data, ...)
```

Arguments

data	A dataframe of distributional parameters.
...	Additional arguments for method.

See Also

Other postprocess: add_mean_sd(), add_mean_sd.default(), add_mean_sd.gamma_samples(), add_mean_sd.weibull_samples(), predict_delay_parameters()

add_mean_sd.weibull_samples

Add natural scale mean and standard deviation parameters for a Weibull model

Description

Note that the input parameters here are mu and shape, corresponding to the distributional parameters used by brms for the weibull family.

Usage

```
## S3 method for class 'weibull_samples'
add_mean_sd(data, ...)
```

Arguments

data	A dataframe of distributional parameters.
...	Additional arguments for method.

See Also

Other postprocess: add_mean_sd(), add_mean_sd.default(), add_mean_sd.gamma_samples(), add_mean_sd.lognormal_samples(), predict_delay_parameters()

assert_epidist

Validation for epidist objects

Description

Validation for epidist objects

Usage

```
assert_epidist(data, ...)
```

Arguments

data	Object to validate
...	Additional arguments

Value

NULL invisibly

assert_epidist.epidist_aggregate_data

Assert validity of `epidist_aggregate_data` *objects*

Description

Assert validity of `epidist_aggregate_data` objects

Usage

```
## S3 method for class 'epidist_aggregate_data'
assert_epidist(data, ...)
```

Arguments

`data`	An object to check
`...`	Additional arguments

See Also

Other aggregate_data: `as_epidist_aggregate_data()`, `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.default()`, `as_epidist_aggregate_data.epidist_linelist_data()`, `is_epidist_aggregate_data()`, `new_epidist_aggregate_data()`

Examples

```
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_aggregate_data() |>
  assert_epidist()
```

assert_epidist.epidist_linelist_data

Assert validity of `epidist_linelist_data` *objects*

Description

Assert validity of `epidist_linelist_data` objects

Usage

```
## S3 method for class 'epidist_linelist_data'
assert_epidist(data, ...)
```

Arguments

`data`	An object to check for validity.
`...`	Additional arguments

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.default()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `is_epidist_linelist_data()`, `new_epidist_linelist_data()`

`as_epidist_aggregate_data` *Create an epidist_aggregate_data object*

Description

Creates an epidist_aggregate_data object from various input formats. This is useful when working with pre-aggregated data where each row represents multiple identical observations. See the specific methods for details on supported input formats and usage examples.

Usage

```
as_epidist_aggregate_data(data, ...)
```

Arguments

`data`	The data to convert
`...`	Additional arguments passed to methods

See Also

Other aggregate_data: `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.default()`, `as_epidist_aggregate_data.epidist_linelist_data()`, `assert_epidist.epidist_aggregate_data()`, `is_epidist_aggregate_data()`, `new_epidist_aggregate_data()`

as_epidist_aggregate_data.data.frame

Create an epidist_aggregate_data object from a data.frame

Description

This method takes a data.frame containing event dates (primary/secondary event dates and observation date) along with counts and creates an `epidist_aggregate_data` object. This format is useful when working with pre-aggregated data where each row represents multiple identical observations with the count stored in a specified column. Internally it makes use of `as_epidist_linelist_data.data.frame()` to convert the data to a linelist format before adding the count column. See the other methods for other data input options.

Usage

```
## S3 method for class 'data.frame'
as_epidist_aggregate_data(
  data,
  n = NULL,
  pdate_lwr = NULL,
  sdate_lwr = NULL,
  pdate_upr = NULL,
  sdate_upr = NULL,
  obs_date = NULL,
  ...
)
```

Arguments

`data` A data.frame containing line list data

`n` A character string giving the name of the column containing the counts for each row. If `NULL` then the column `n` must be present in the data.

`pdate_lwr` A string giving the column of `data` containing the primary event lower bound as a datetime. Defaults to `NULL` which assumes that the variable `pdate_lwr` is present.

`sdate_lwr` A string giving the column of `data` containing the secondary event lower bound as a datetime. Defaults to `NULL` which assumes that the variable `sdate_lwr` is present.

`pdate_upr` A string giving the column of `data` containing the primary event upper bound as a datetime. If this column exists in the data it will be used, otherwise if not supplied then the value of `pdate_lwr` + 1 day is used.

`sdate_upr` A string giving the column of `data` containing the secondary event upper bound as a datetime. If this column exists in the data it will be used, otherwise if not supplied then the value of `sdate_lwr` + 1 day is used.

obs_date	A string giving the column of data containing the observation time as a date-time. Optional, if not supplied then the maximum of sdate_upr is used.
...	Additional arguments passed to methods

See Also

Other aggregate_data: as_epidist_aggregate_data(), as_epidist_aggregate_data.default(), as_epidist_aggregate_data.epidist_linelist_data(), assert_epidist.epidist_aggregate_data(), is_epidist_aggregate_data(), new_epidist_aggregate_data()

Examples

```
sierra_leone_ebola_data |>
  dplyr::count(date_of_symptom_onset, date_of_sample_tested) |>
  as_epidist_aggregate_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested",
    n = "n"
  )
```

as_epidist_aggregate_data.default

Create an epidist_aggregate_data object from vectors of event times

Description

This method takes vectors of event times (primary/secondary event times and observation time) along with counts and creates an epidist_aggregate_data object. This format is useful when working with pre-aggregated data where each row represents multiple identical observations with the count stored in the n column. Internally it makes use of as_epidist_linelist_data.default() to convert the data to a linelist format before adding the count column. See the other methods for other data input options.

Usage

```
## Default S3 method:
as_epidist_aggregate_data(
  data,
  n = NULL,
  ptime_upr = NULL,
  stime_lwr = NULL,
  stime_upr = NULL,
  obs_time = NULL,
  ...
)
```

Arguments

`data` Numeric vector giving lower bounds for primary times.

`n` An integerish vector containing the counts for each row. Must be the same length as the input data vector.

`ptime_upr` Numeric vector giving upper bounds for primary times.

`stime_lwr`, `stime_upr`
Numeric vectors giving lower and upper bounds for secondary times.

`obs_time` Numeric vector giving observation times.

`...` Additional columns to add to the epidist_linelist_data object.

See Also

Other aggregate_data: `as_epidist_aggregate_data()`, `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.epidist_linelist_data()`, `assert_epidist.epidist_aggregate_data()`, `is_epidist_aggregate_data()`, `new_epidist_aggregate_data()`

Examples

```
as_epidist_aggregate_data(
  data = c(1, 2, 3),
  ptime_upr = c(2, 3, 4),
  stime_lwr = c(3, 4, 5),
  stime_upr = c(4, 5, 6),
  obs_time = c(5, 6, 7),
  n = c(1, 2, 3)
)
```

`as_epidist_aggregate_data.epidist_linelist_data`
Convert linelist data to aggregate format

Description

This method takes an `epidist_linelist_data` object (see `as_epidist_linelist_data()`) and aggregates it by counting unique combinations of the required time variables (primary/secondary event times and observation time) and any additional variables specified in by. The result is a more compact representation of the same data where each row represents multiple identical observations with the count stored in the `n` column.

Usage

```
## S3 method for class 'epidist_linelist_data'
as_epidist_aggregate_data(data, by = NULL, ...)
```

Arguments

`data` The data to convert

`by` Character vector of additional variables to stratify by, beyond the required time variables.

`...` Additional arguments passed to methods

See Also

Other aggregate_data: `as_epidist_aggregate_data()`, `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.default()`, `assert_epidist.epidist_aggregate_data()`, `is_epidist_aggregate_da` `new_epidist_aggregate_data()`

Examples

```
# Default stratification by required time variables only
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_aggregate_data()

# Additional stratification by other variables
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_aggregate_data(by = "age")
```

`as_epidist_latent_model` *Convert an object to an* `epidist_latent_model` *object*

Description

Creates an `epidist_latent_model` object from various input formats. This enables fitting latent variable models for epidemiological delays using `epidist()`, as described in Park et al. (2024) and Charniga et al. (2024) The latent model approach accounts for double interval censoring and right truncation in delay data.

Usage

```
as_epidist_latent_model(data, ...)
```

Arguments

`data` An object to be converted to the class `epidist_latent_model`

`...` Additional arguments passed to methods.

References

- Park et al. (2024)
- Charniga et al. (2024)

See Also

Other latent_model: `as_epidist_latent_model.epidist_aggregate_data()`, `as_epidist_latent_model.epidist_li`
`epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `is_epidist_latent_model()`, `new_epidist_latent_model()`

`as_epidist_latent_model.epidist_aggregate_data`
The latent model method for `epidist_aggregate_data` *objects*

Description

This method converts aggregate data to a latent model format by first converting it to linelist format using `as_epidist_linelist_data.epidist_aggregate_data()` and then passing it to `as_epidist_latent_model.epidist_linelist_data()`. This ensures that the counts in the aggregate data are properly expanded into individual observations before fitting the latent model.

Usage

```
## S3 method for class 'epidist_aggregate_data'
as_epidist_latent_model(data, ...)
```

Arguments

`data`	An epidist_aggregate_data object
`...`	Not used in this method.

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_linelist_data()`, `epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `is_epidist_latent_model()`, `new_epidist_latent_model()`

Examples

```
sierra_leone_ebola_data |>
  dplyr::count(date_of_symptom_onset, date_of_sample_tested) |>
  as_epidist_aggregate_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested",
    n = "n"
  ) |>
  as_epidist_latent_model()
```

as_epidist_latent_model.epidist_linelist_data

The latent model method for `epidist_linelist_data` *objects*

Description

This method takes an `epidist_linelist_data` object and converts it to a format suitable for fitting latent variable models. It calculates key variables needed for the latent variable method described in Park et al. (2024) and Charniga et al. (2024). This approach adjusts for double interval censoring and right truncation in the data.

Usage

```
## S3 method for class 'epidist_linelist_data'
as_epidist_latent_model(data, ...)
```

Arguments

`data`	An `epidist_linelist_data` object containing individual-level observations with primary and secondary event times. See `as_epidist_linelist_data()` for details on creating this object.
`...`	Not used in this method.

References

- Park et al. (2024)
- Charniga et al. (2024)

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_aggregate_data()`, `epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `is_epidist_latent_model()`, `new_epidist_latent_model()`

Examples

```
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_latent_model()
```

`as_epidist_linelist_data` *Create an epidist_linelist_data object*

Description

Creates an epidist_linelist_data object from various input formats. This is useful when working with individual-level data where each row represents a single observation with primary and secondary event times. See the specific methods for details on supported input formats and usage examples.

Usage

```
as_epidist_linelist_data(data, ...)
```

Arguments

`data`	The data to convert
`...`	Additional arguments passed to methods

See Also

Other linelist_data: `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.default()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `assert_epidist.epidist_linelist_data()`, `is_epidist_linelist_data()`, `new_epidist_linelist_data()`

`as_epidist_linelist_data.data.frame` *Create an epidist_linelist_data object from a data frame with event dates*

Description

This method takes a data.frame containing event dates (primary/secondary event dates and observation date) and creates an `epidist_linelist_data` object. This format is useful when working with individual-level data where each row represents a single observation. Internally it converts dates to numeric times relative to the earliest primary event date and uses `as_epidist_linelist_data.default()` to create the final object. See the other methods for other data input options.

Usage

```
## S3 method for class 'data.frame'
as_epidist_linelist_data(
  data,
  pdate_lwr = NULL,
  sdate_lwr = NULL,
  pdate_upr = NULL,
  sdate_upr = NULL,
  obs_date = NULL,
  ...
)
```

Arguments

`data` A data.frame containing line list data

`pdate_lwr` A string giving the column of `data` containing the primary event lower bound as a datetime. Defaults to `NULL` which assumes that the variable `pdate_lwr` is present.

`sdate_lwr` A string giving the column of `data` containing the secondary event lower bound as a datetime. Defaults to `NULL` which assumes that the variable `sdate_lwr` is present.

`pdate_upr` A string giving the column of `data` containing the primary event upper bound as a datetime. If this column exists in the data it will be used, otherwise if not supplied then the value of `pdate_lwr` + 1 day is used.

`sdate_upr` A string giving the column of `data` containing the secondary event upper bound as a datetime. If this column exists in the data it will be used, otherwise if not supplied then the value of `sdate_lwr` + 1 day is used.

`obs_date` A string giving the column of `data` containing the observation time as a datetime. Optional, if not supplied then the maximum of `sdate_upr` is used.

`...` Additional arguments passed to methods

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.default()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `assert_epidist.epidist_linelist_data()`, `is_epidist_linelist_data()`, `new_epidist_linelist_data()`

Examples

```
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  )
```

---

as_epidist_linelist_data.default

Create an epidist_linelist_data object from vectors of event times

---

Description

This method takes vectors of event times (primary/secondary event times and observation time) and creates an `epidist_linelist_data` object. This format is useful when working with individual-level data where each row represents a single observation. See the other methods for other data input options.

Usage

```
## Default S3 method:
as_epidist_linelist_data(
  data,
  ptime_upr = NULL,
  stime_lwr = NULL,
  stime_upr = NULL,
  obs_time = NULL,
  ...
)
```

Arguments

`data` Numeric vector giving lower bounds for primary times.

`ptime_upr` Numeric vector giving upper bounds for primary times.

`stime_lwr`, `stime_upr`
Numeric vectors giving lower and upper bounds for secondary times.

`obs_time` Numeric vector giving observation times.

`...` Additional columns to add to the epidist_linelist_data object.

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `assert_epidist.epidist_linelist_data()`, `is_epidist_linelist_data()`, `new_epidist_linelist_data()`

Examples

```
as_epidist_linelist_data(
  data = c(1, 2, 3),
  ptime_upr = c(2, 3, 4),
  stime_lwr = c(3, 4, 5),
  stime_upr = c(4, 5, 6),
  obs_time = c(5, 6, 7)
)
```

as_epidist_linelist_data.epidist_aggregate_data

Convert aggregate data to linelist format

Description

This method expands an `epidist_aggregate_data` object into individual observations by uncounting the `n` column, then converts it to linelist format using `as_epidist_linelist_data.data.frame()`.

Usage

```
## S3 method for class 'epidist_aggregate_data'
as_epidist_linelist_data(data, ...)
```

Arguments

`data`	The data to convert
`...`	Additional arguments passed to methods

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.default()`, `assert_epidist.epidist_linelist_data()`, `is_epidist_linelist_data(`
`new_epidist_linelist_data()`

Examples

```
sierra_leone_ebola_data |>
  dplyr::count(date_of_symptom_onset, date_of_sample_tested) |>
  as_epidist_aggregate_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested",
    n = "n"
  ) |>
  as_epidist_linelist_data()
```

as_epidist_marginal_model

Convert an object to an `epidist_marginal_model` *object*

Description

Creates an `epidist_marginal_model` object from various input formats. This enables fitting marginal models for epidemiological delays using `epidist()`. The marginal model approach uses the likelihood from the primarycensored package to efficiently handle censoring in both primary and secondary events as well as truncation due to observation times. See the specific methods `as_epidist_marginal_model.epidist_linelist_data()` and `as_epidist_marginal_model.epidist_aggregate_dat`
for details on supported input formats and usage examples.

Usage

```
as_epidist_marginal_model(data, ...)
```

Arguments

`data` An object to be converted to the class `epidist_marginal_model`

`...` Additional arguments passed to methods.

See Also

Other marginal_model: `as_epidist_marginal_model.epidist_aggregate_data()`, `as_epidist_marginal_model.epi`
`epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model()`, `is_epidist_marginal_model()`, `new_epidist_marginal_model()`

`as_epidist_marginal_model.epidist_aggregate_data`

The marginal model method for `epidist_aggregate_data` *objects*

Description

This method converts aggregate data to a marginal model format by passing it to `as_epidist_marginal_model.epidist_li`
with the `n` column used as weights. This ensures that the likelihood is weighted by the counts in the aggregate data.

Usage

```
## S3 method for class 'epidist_aggregate_data'
as_epidist_marginal_model(data, obs_time_threshold = 2, ...)
```

Arguments

`data` An `epidist_aggregate_data` object

`obs_time_threshold`
Ratio used to determine threshold for setting relative observation times to Inf. Observation times greater than `obs_time_threshold` times the maximum delay will be set to Inf to improve model efficiency by reducing the number of unique observation times. Default is 2.

`...` Not used in this method.

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_linelist_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model()`, `is_epidist_marginal_model()`, `new_epidist_marginal_model()`

Examples

```
sierra_leone_ebola_data |>
  dplyr::count(date_of_symptom_onset, date_of_sample_tested) |>
  as_epidist_aggregate_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested",
    n = "n"
  ) |>
  as_epidist_marginal_model()
```

`as_epidist_marginal_model.epidist_linelist_data` *The marginal model method for* `epidist_linelist_data` *objects*

Description

This method converts linelist data to a marginal model format by calculating delays between primary and secondary events, along with observation times and censoring windows. The likelihood used is imported from the primarycensored package which handles censoring in both primary and secondary events as well as truncation due to observation times. In principle, this method should be more accurate and more computationally efficient than the latent model (`as_epidist_latent_model()`) approach in most settings except when the number of unique strata approaches the number of observations.

Usage

```
## S3 method for class 'epidist_linelist_data'
as_epidist_marginal_model(data, obs_time_threshold = 2, weight = NULL, ...)
```

Arguments

`data` An `epidist_linelist_data` object

`obs_time_threshold` Ratio used to determine threshold for setting relative observation times to Inf. Observation times greater than `obs_time_threshold` times the maximum delay will be set to Inf to improve model efficiency by reducing the number of unique observation times. Default is 2.

`weight` A column name to use for weighting the data in the likelihood. Default is NULL. Internally this is used to define the 'n' column of the returned object.

`...` Not used in this method.

Details

When a formula is specified in `epidist()`, the data will be transformed using `epidist_transform_data_model.epidist_m` to prepare it for model fitting. This transformation summarises the data by counting unique combinations of delays, observation times, censoring windows and any variables in the model formula.

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_aggregate_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model()`, `is_epidist_marginal_model()`, `new_epidist_marginal_model()`

Examples

```
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_marginal_model()
```

`as_epidist_naive_model` *Convert an object to an* `epidist_naive_model` *object*

Description

Creates an `epidist_naive_model` object from various input formats. This enables fitting naive models for epidemiological delays using `epidist()`. The naive model approach ignores censoring and truncation in the data, using only the lower bounds of the intervals as point estimates. This is the simplest approach but may lead to biased estimates if there is substantial censoring or truncation in the data.

Usage

```
as_epidist_naive_model(data, ...)
```

Arguments

`data`	An object to be converted to the class `epidist_naive_model`
`...`	Additional arguments passed to methods.

See Also

Other naive_model: `as_epidist_naive_model.epidist_aggregate_data()`, `as_epidist_naive_model.epidist_line`
`epidist_formula_model.epidist_naive_model()`, `epidist_transform_data_model.epidist_naive_model()`, `is_epidist_naive_model()`, `new_epidist_naive_model()`

as_epidist_naive_model.epidist_aggregate_data

The naive model method for epidist_aggregate_data *objects*

Description

This method converts aggregate data to a naive model format by passing it to as_epidist_naive_model.epidist_linelist with the n column used as weights. This ensures that the likelihood is weighted by the counts in the aggregate data.

Usage

```
## S3 method for class 'epidist_aggregate_data'
as_epidist_naive_model(data, ...)
```

Arguments

data	An epidist_aggregate_data object.
...	Additional arguments passed to methods.

See Also

Other naive_model: as_epidist_naive_model(), as_epidist_naive_model.epidist_linelist_data(), epidist_formula_model.epidist_naive_model(), epidist_transform_data_model.epidist_naive_model(), is_epidist_naive_model(), new_epidist_naive_model()

Examples

```
sierra_leone_ebola_data |>
  dplyr::count(date_of_symptom_onset, date_of_sample_tested) |>
  as_epidist_aggregate_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested",
    n = "n"
  ) |>
  as_epidist_naive_model()
```

as_epidist_naive_model.epidist_linelist_data

The naive model method for epidist_linelist_data *objects*

Description

This method converts linelist data to a naive model format by calculating delays between primary and secondary events to enable model fitting in epidist(). If the input data contains an n column (e.g. from aggregated data), the likelihood will be weighted by these counts.

Usage

```
## S3 method for class 'epidist_linelist_data'
as_epidist_naive_model(data, weight = NULL, ...)
```

Arguments

`data`	An `epidist_linelist_data` object.
`weight`	A column name to use for weighting the data in the likelihood. Default is NULL. Internally this is used to define the 'n' column of the returned object.
`...`	Additional arguments passed to methods.

Details

When a formula is specified in `epidist()`, the data will be transformed using `epidist_transform_data_model.epidist_n` to prepare it for model fitting. This transformation summarises the data by counting unique combinations of delays and any variables in the model formula.

The naive model is the simplest approach but ignores censoring and truncation in the data by using only lower bounds as point estimates. For data with substantial censoring or truncation, consider using `as_epidist_latent_model()` or `as_epidist_marginal_model()` which properly account for these features.

See Also

Other naive_model: `as_epidist_naive_model`(), `as_epidist_naive_model.epidist_aggregate_data`(), `epidist_formula_model.epidist_naive_model`(), `epidist_transform_data_model.epidist_naive_model`(), `is_epidist_naive_model`(), `new_epidist_naive_model`()

Examples

```
sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_naive_model()
```

`epidist` *Fit epidemiological delay distributions using a* `brms` *interface*

Description

Fit epidemiological delay distributions using a `brms` interface

Usage

```
epidist(
  data,
  formula = mu ~ 1,
  family = lognormal(),
  prior = NULL,
  merge_priors = TRUE,
  fn = brms::brm,
  ...
)
```

Arguments

`data` An object with class corresponding to an implemented model.

`formula` An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.

`family` A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.

`prior` One or more `brmsprior` objects created by `brms::set_prior()` or related functions. These priors are passed to `epidist_prior()` in the `prior` argument. Some models have default priors that are automatically added (see `epidist_model_prior()`). These can be merged with user-provided priors using the `merge_priors` argument.

`merge_priors` If `TRUE` then merge user priors with default priors, if `FALSE` only use user priors. Defaults to `TRUE`. This may be useful if the built in approaches for merging priors are not flexible enough for a particular use case.

`fn` The internal function to be called. By default this is `brms::brm()` which performs inference for the specified model. Other options are `brms::make_stancode()` which returns the Stan code for the specified model, or `brms::make_standata()` which returns the data passed to Stan. These two later options may be useful for model debugging and extensions.

`...` Additional arguments passed to `fn` method.

Examples

```
fit <- sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_aggregate_data() |>
  as_epidist_marginal_model() |>
  epidist(chains = 2, cores = 2, refresh = ifelse(interactive(), 250, 0))

summary(fit)
```

epidist_diagnostics *Diagnostics for* `epidist_fit` *models*

Description

This function computes diagnostics to assess the quality of a fitted model. When the fitting algorithm used is `"sampling"` (HMC) then the output of `epidist_diagnostics` is a `data.frame` containing:

- `time`: the total time taken to fit all chains
- `samples`: the total number of samples across all chains
- `max_rhat`: the highest value of the Gelman-Rubin statistic
- `divergent_transitions`: the total number of divergent transitions
- `per_divergent_transitions`: the proportion of samples which had divergent transitions
- `max_treedepth`: the highest value of the treedepth HMC parameter
- `no_at_max_treedepth`: the number of samples which attained the `max_treedepth`
- `per_at_max_treedepth`: the proportion of samples which attained the `max_treedepth`

Usage

```
epidist_diagnostics(fit)
```

Arguments

`fit`	A fitted model of class `epidist_fit`

Details

When the fitting algorithm is not `"sampling"` (see `brms::brm()` for other possible algorithms) then diagnostics are yet to be implemented.

Examples

```
fit <- sierra_leone_ebola_data |>
  as_epidist_linelist_data(
    pdate_lwr = "date_of_symptom_onset",
    sdate_lwr = "date_of_sample_tested"
  ) |>
  as_epidist_aggregate_data() |>
  as_epidist_marginal_model() |>
  epidist(chains = 2, cores = 2, refresh = ifelse(interactive(), 250, 0))
epidist_diagnostics(fit)
```

epidist_family *Define* epidist *family*

Description

This function is used within epidist() to create a model specific custom brms family object. This custom family is passed to brms. It is unlikely that as a user you will need this function, but we export it nonetheless to be transparent about what happens inside of a call to epidist().

Usage

```
epidist_family(data, family = lognormal(), ...)
```

Arguments

data	An object with class corresponding to an implemented model.
family	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see brmsfamily(). Commonly used, such as lognormal(), are also reexported as part of epidist.
...	Additional arguments passed to fn method.

See Also

Other family: epidist_family_model(), epidist_family_model.default(), epidist_family_param(), epidist_family_param.default()

epidist_family_model *The model-specific parts of an* epidist_family() *call*

Description

The model-specific parts of an epidist_family() call

The model-specific parts of an epidist_formula() call

Usage

```
epidist_family_model(data, family, ...)

epidist_formula_model(data, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	Output of a call to `brms::brmsfamily()` with additional information as provided by `.add_dpar_info()`
`...`	Additional arguments passed to `fn` method.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.

See Also

Other family: `epidist_family()`, `epidist_family_model.default()`, `epidist_family_param()`, `epidist_family_param.default()`

Other formula: `epidist_formula()`, `epidist_formula_model.default()`

`epidist_family_model.default`

Default method for defining a model specific family

Description

Default method for defining a model specific family

Usage

```
## Default S3 method:
epidist_family_model(data, family, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	Output of a call to `brms::brmsfamily()` with additional information as provided by `.add_dpar_info()`
`...`	Additional arguments passed to `fn` method.

See Also

Other family: `epidist_family()`, `epidist_family_model()`, `epidist_family_param()`, `epidist_family_param.defa`

epidist_family_model.epidist_latent_model

Create the model-specific component of an `epidist` *custom family*

Description

Create the model-specific component of an `epidist` custom family

Usage

```
## S3 method for class 'epidist_latent_model'
epidist_family_model(data, family, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	Output of a call to `brms::brmsfamily()` with additional information as provided by `.add_dpar_info()`
`...`	Additional arguments passed to method.

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_aggregate_data()`, `as_epidist_latent_model.epidist_linelist_data()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `is_epidist_latent_model()`, `new_epidist_latent_model()`

epidist_family_model.epidist_marginal_model

Create the model-specific component of an `epidist` *custom family*

Description

Create the model-specific component of an `epidist` custom family

Usage

```
## S3 method for class 'epidist_marginal_model'
epidist_family_model(data, family, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	Output of a call to `brms::brmsfamily()` with additional information as provided by `.add_dpar_info()`
`...`	Additional arguments passed to method.

See Also

Other marginal_model: as_epidist_marginal_model(), as_epidist_marginal_model.epidist_aggregate_data(), as_epidist_marginal_model.epidist_linelist_data(), epidist_formula_model.epidist_marginal_model(), epidist_transform_data_model.epidist_marginal_model(), is_epidist_marginal_model(), new_epidist_marginal_model()

epidist_family_param *Reparameterise an* epidist *family to align* brms *and Stan*

Description

Reparameterise an epidist family to align brms and Stan

Usage

```
epidist_family_param(family, ...)
```

Arguments

family	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see brmsfamily(). Commonly used, such as lognormal(), are also reexported as part of epidist.
...	Additional arguments passed to fn method.

See Also

Other family: epidist_family(), epidist_family_model(), epidist_family_model.default(), epidist_family_param.default()

epidist_family_param.default *Default method for families which do not require a reparameterisation*

Description

This function extracts the Stan parameterisation for a given brms family by creating a dummy model and parsing its Stan code. It looks for the log probability density function (lpdf) call in the Stan code and extracts the parameter order used by Stan. This is needed because brms and Stan may use different parameter orderings for the same distribution.

Usage

```
## Default S3 method:
epidist_family_param(family, ...)
```

Arguments

family A brms family object containing at minimum a `family` element specifying the distribution family name.

... Additional arguments passed to methods (not used)

Details

The function works by:

1. Creating a minimal dummy model using the specified family
2. Extracting the Stan code for this model
3. Finding the lpdf function call for the family
4. Parsing out the parameter ordering used in Stan
5. Adding this as the `param` element to the family object

Value

The input family object with an additional `param` element containing the Stan parameter ordering as a string

See Also

Other family: `epidist_family()`, `epidist_family_model()`, `epidist_family_model.default()`, `epidist_family_param()`

`epidist_family_prior` *Family specific prior distributions*

Description

This function contains `brms` prior distributions which are specific to particular likelihood families e.g. `brms::lognormal()`.

Usage

```
epidist_family_prior(family, ...)
```

Arguments

family A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.

... Additional arguments passed to `fn` method.

See Also

Other prior: `epidist_family_prior.default()`, `epidist_family_prior.lognormal()`, `epidist_model_prior()`, `epidist_model_prior.default()`, `epidist_prior()`

`epidist_family_prior.default`
Default family specific prior distributions

Description

By default, we do not return any family specific prior distributions.

Usage

```
## Default S3 method:
epidist_family_prior(family, formula, ...)
```

Arguments

`family`	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.
`...`	Additional arguments passed to `fn` method.

See Also

Other prior: `epidist_family_prior()`, `epidist_family_prior.lognormal()`, `epidist_model_prior()`, `epidist_model_prior.default()`, `epidist_prior()`

epidist_family_prior.lognormal

Family specific prior distributions for the lognormal family

Description

We suggest priors to overwrite the brms defaults for the lognormal family.

Usage

```
## S3 method for class 'lognormal'
epidist_family_prior(family, formula, ...)
```

Arguments

family	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see brmsfamily(). Commonly used, such as lognormal(), are also reexported as part of epidist.
formula	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter mu, and may optionally be provided for other distributional parameters.
...	Additional arguments passed to fn method.

See Also

Other prior: epidist_family_prior(), epidist_family_prior.default(), epidist_model_prior(), epidist_model_prior.default(), epidist_prior()

epidist_formula

Define a model specific formula

Description

This function is used within epidist() to create the formula object passed to brms. It is unlikely that as a user you will need this function, but we export it nonetheless to be transparent about what exactly is happening inside of a call to epidist().

Usage

```
epidist_formula(data, family, formula, ...)
```

Arguments

`data` An object with class corresponding to an implemented model.

`family` A description of the response distribution and link function to be used in the model created using `epidist_family()`.

`formula` An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.

`...` Additional arguments passed to `fn` method.

See Also

Other formula: `epidist_family_model()`, `epidist_formula_model.default()`

`epidist_formula_model.default` *Default method for defining a model specific formula*

Description

Default method for defining a model specific formula

Usage

```
## Default S3 method:
epidist_formula_model(data, formula, ...)
```

Arguments

`data` An object with class corresponding to an implemented model.

`formula` An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.

`...` Additional arguments passed to `fn` method.

See Also

Other formula: `epidist_family_model()`, `epidist_formula()`

epidist_formula_model.epidist_latent_model

Define the model-specific component of an epidist *custom formula for the latent model*

Description

Define the model-specific component of an epidist custom formula for the latent model

Usage

```
## S3 method for class 'epidist_latent_model'
epidist_formula_model(data, formula, ...)
```

Arguments

data	An object with class corresponding to an implemented model.
formula	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter mu, and may optionally be provided for other distributional parameters.
...	Additional arguments passed to method.

See Also

Other latent_model: as_epidist_latent_model(), as_epidist_latent_model.epidist_aggregate_data(), as_epidist_latent_model.epidist_linelist_data(), epidist_family_model.epidist_latent_model(), epidist_model_prior.epidist_latent_model(), is_epidist_latent_model(), new_epidist_latent_model()

epidist_formula_model.epidist_marginal_model

Define the model-specific component of an epidist *custom formula for the marginal model*

Description

Define the model-specific component of an epidist custom formula for the marginal model

Usage

```
## S3 method for class 'epidist_marginal_model'
epidist_formula_model(data, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.
`...`	Additional arguments passed to method.

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_aggregate_data()`, `as_epidist_marginal_model.epidist_linelist_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model()`, `is_epidist_marginal_model()`, `new_epidist_marginal_model()`

`epidist_formula_model.epidist_naive_model` *Define the model-specific component of an* `epidist` *custom formula for the naive model*

Description

Define the model-specific component of an `epidist` custom formula for the naive model

Usage

```
## S3 method for class 'epidist_naive_model'
epidist_formula_model(data, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.
`...`	Additional arguments passed to method.

See Also

Other naive_model: `as_epidist_naive_model()`, `as_epidist_naive_model.epidist_aggregate_data()`, `as_epidist_naive_model.epidist_linelist_data()`, `epidist_transform_data_model.epidist_naive_model()`, `is_epidist_naive_model()`, `new_epidist_naive_model()`

`epidist_gen_log_lik` *Create a function to calculate the marginalised log likelihood for double censored and truncated delay distributions*

Description

This function creates a log likelihood function that calculates the marginal likelihood for a single observation by integrating over the latent primary and secondary event windows. Where analytical solutions exist in `primarycensored::dpcens()` these are used, otherwise the integration is performed numerically. `primarycensored::dpcens()` handles the double censoring and truncation of the delay distribution.

Usage

```
epidist_gen_log_lik(family)
```

Arguments

`family`	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.

Details

The marginal likelihood accounts for uncertainty in both the primary and secondary event windows by integrating over their possible values, weighted by their respective uniform distributions.

Value

A function that calculates the marginal log likelihood for a single observation. The prep object must have the following variables:

- `vreal1`: relative observation time
- `vreal2`: primary event window
- `vreal3`: secondary event window

See Also

`brms::log_lik()` for details on the brms log likelihood interface.

Other gen: `epidist_gen_posterior_epred()`, `epidist_gen_posterior_predict()`

epidist_gen_posterior_epred

Create a function to draw from the expected value of the posterior predictive distribution for a model

Description

This function creates a function that calculates the expected value of the posterior predictive distribution for a latent model. The returned function takes a `prep` argument (from brms) and returns posterior expected values. This is used internally by `brms::posterior_epred()` to calculate expected values for latent models.

Usage

```
epidist_gen_posterior_epred(family)
```

Arguments

`family`	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.

Value

A function that takes a prep argument from brms and returns a matrix of posterior expected values, with one row per posterior draw and one column per observation.

See Also

`brms::posterior_epred()` for details on how this is used within `brms`.

Other gen: `epidist_gen_log_lik`(), `epidist_gen_posterior_predict`()

epidist_gen_posterior_predict

Create a function to draw from the posterior predictive distribution for a double censored and truncated delay distribution

Description

This function creates a function that draws from the posterior predictive distribution for a latent model using `primarycensored::rpcens()` to handle censoring and truncation. The returned function takes a `prep` argument from `brms` and returns posterior predictions. This is used internally by `brms::posterior_predict()` to generate predictions for latent models.

Usage

```
epidist_gen_posterior_predict(family)
```

Arguments

`family`	A description of the response distribution and link function to be used in the model. Every family function has a link argument allowing users to specify the link function to be applied on the response variable. If not specified, default links are used. For details of all supported families see `brmsfamily()`. Commonly used, such as `lognormal()`, are also reexported as part of `epidist`.

Value

A function that takes a `prep` argument from brms and returns a matrix of posterior predictions, with one row per posterior draw and one column per observation. The `prep` object must have the following variables:

- `vreal1`: relative observation time
- `vreal2`: primary event window
- `vreal3`: secondary event window

See Also

`brms::posterior_predict()` for details on how this is used within `brms`, `primarycensored::rpcens()` for details on the censoring approach

Other gen: `epidist_gen_log_lik()`, `epidist_gen_posterior_epred()`

`epidist_model_prior` *Model specific prior distributions*

Description

This function contains `brms` prior distributions which are specific to particular `epidist` models e.g. the `latent_lognormal` model.

Usage

```
epidist_model_prior(data, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`...`	Additional arguments passed to `fn` method.

See Also

Other prior: `epidist_family_prior()`, `epidist_family_prior.default()`, `epidist_family_prior.lognormal()`, `epidist_model_prior.default()`, `epidist_prior()`

epidist_model_prior.default

Default model specific prior distributions

Description

By default, we do not return any model specific prior distributions.

Usage

```
## Default S3 method:
epidist_model_prior(data, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.
`...`	Additional arguments passed to `fn` method.

See Also

Other prior: `epidist_family_prior()`, `epidist_family_prior.default()`, `epidist_family_prior.lognormal()`, `epidist_model_prior()`, `epidist_prior()`

epidist_model_prior.epidist_latent_model

Model specific prior distributions for latent models

Description

Defines prior distributions for the latent model parameters `pwindow_raw` and `swindow_raw` which control the width of the observation windows.

Usage

```
## S3 method for class 'epidist_latent_model'
epidist_model_prior(data, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`formula`	An object of class stats::formula or brms::brmsformula (or one that can be coerced to those classes). A symbolic description of the model to be fitted. A formula must be provided for the distributional parameter `mu`, and may optionally be provided for other distributional parameters.
`...`	Additional arguments passed to `fn` method.

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_aggregate_data()`, `as_epidist_latent_model.epidist_linelist_data()`, `epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `is_epidist_latent_model()`, `new_epidist_latent_model()`

`epidist_prior` *Define custom prior distributions for epidist models*

Description

This function combines model specific prior distributions from `epidist_model_prior()`, family specific prior distributions from `epidist_family_prior()`, and user provided prior distributions into a single set of custom priors. Each element overwrites previous elements, such that user provided prior distributions have the highest priority. If a user prior distribution is provided which is not included in the model, a warning will be shown.

Usage

```
epidist_prior(
  data,
  family,
  formula,
  prior,
  merge = TRUE,
  enforce_presence = FALSE
)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	A description of the response distribution and link function to be used in the model created using `epidist_family()`.
`formula`	A symbolic description of the model to be fitted created using `epidist_formula()`.
`prior`	One or more `brmsprior` objects created by `brms::set_prior()` or related functions. These priors are passed to `epidist_prior()` in the `prior` argument. Some models have default priors that are automatically added (see `epidist_model_prior()`). These can be merged with user-provided priors using the `merge_priors` argument.

`merge` If `TRUE` then merge new priors with existing ones, if `FALSE` only use new priors. Defaults to `TRUE`. This may be useful if the built in approaches for merging priors are not flexible enough for a particular use case.

`enforce_presence`
If `TRUE` then only allow user priors that match existing default priors. If `FALSE` then allow user priors that are not present in the default set. Defaults to `FALSE`.

Details

Note that the matching of priors is imperfect as it does not use brms' internal prior matching functionality. For example, it cannot distinguish between a prior for all coefficients (class = "b") and a prior for a specific coefficient (class = "b" and coef specified).

Value

A `brmsprior` object containing the combined custom prior distributions.

See Also

Other prior: `epidist_family_prior()`, `epidist_family_prior.default()`, `epidist_family_prior.lognormal()`, `epidist_model_prior()`, `epidist_model_prior.default()`

`epidist_stancode` *Define model specific Stan code*

Description

This function is used within `epidist()` to create any custom Stan code which is injected into `brms` via the `stanvars` argument. It is unlikely that as a user you will need this function, but we export it nonetheless to be transparent about what exactly is happening inside of a call to `epidist()`.

Usage

```
epidist_stancode(data, ...)
```

Arguments

`data` An object with class corresponding to an implemented model.

`...` Additional arguments passed to `fn` method.

See Also

Other stan: `epidist_stancode.default()`

epidist_stancode.default

Default method for defining model specific Stan code

Description

Default method for defining model specific Stan code

Usage

```
## Default S3 method:
epidist_stancode(data, ...)
```

Arguments

data	An object with class corresponding to an implemented model.
...	Additional arguments passed to fn method.

See Also

Other stan: epidist_stancode()

epidist_transform_data

Transform data for an epidist model

Description

This function is used within epidist() to transform data before passing to brms. It is unlikely that as a user you will need this function, but we export it nonetheless to be transparent about what happens inside of a call to epidist().

Usage

```
epidist_transform_data(data, family, formula, ...)
```

Arguments

data	An object with class corresponding to an implemented model.
family	A description of the response distribution and link function to be used in the model created using epidist_family().
formula	A formula object created using epidist_formula().
...	Additional arguments passed to fn method.

See Also

Other transform_data: epidist_transform_data_model(), epidist_transform_data_model.default()

epidist_transform_data_model

The model-specific parts of an `epidist_transform_data()` *call*

Description

The model-specific parts of an `epidist_transform_data()` call

Usage

```
epidist_transform_data_model(data, family, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	A description of the response distribution and link function to be used in the model created using `epidist_family()`.
`formula`	A formula object created using `epidist_formula()`.
`...`	Additional arguments passed to `fn` method.

See Also

Other transform_data: `epidist_transform_data()`, `epidist_transform_data_model.default()`

epidist_transform_data_model.default

Default method for transforming data for a model

Description

Default method for transforming data for a model

Usage

```
## Default S3 method:
epidist_transform_data_model(data, family, formula, ...)
```

Arguments

`data`	An object with class corresponding to an implemented model.
`family`	A description of the response distribution and link function to be used in the model created using `epidist_family()`.
`formula`	A formula object created using `epidist_formula()`.
`...`	Additional arguments passed to `fn` method.

See Also

Other transform_data: `epidist_transform_data()`, `epidist_transform_data_model()`

`epidist_transform_data_model.epidist_marginal_model`
Transform data for the marginal model

Description

This method transforms data into the format required by the marginal model by:

1. Identifying required columns for the marginal model
2. Summarising the data by counting unique combinations of these columns and any variables in the model formula using `.summarise_n_by_formula()`
3. Converting the summarised data to a marginal model object using `new_epidist_marginal_model()`
4. Informing the user about any data aggregation that occurred using `.inform_data_summarised()`

Usage

```
## S3 method for class 'epidist_marginal_model'
epidist_transform_data_model(data, family, formula, ...)
```

Arguments

`data`	The data to transform
`family`	The epidist family object specifying the distribution
`formula`	The model formula
`...`	Additional arguments passed to methods

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_aggregate_data()`, `as_epidist_marginal_model.epidist_linelist_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `is_epidist_marginal_model()`, `new_epidist_marginal_mode`

epidist_transform_data_model.epidist_naive_model

Transform data for the naive model

Description

This method transforms data into the format required by the naive model by:

1. Identifying required columns for the naive model
2. Summarising the data by counting unique combinations of these columns and any variables in the model formula using `.summarise_n_by_formula()`
3. Converting the summarised data to a naive model object using `new_epidist_naive_model()`
4. Informing the user about any data aggregation that occurred using `.inform_data_summarised()`

Usage

```
## S3 method for class 'epidist_naive_model'
epidist_transform_data_model(data, family, formula, ...)
```

Arguments

`data`	The data to transform
`family`	The epidist family object specifying the distribution
`formula`	The model formula
`...`	Additional arguments passed to methods

See Also

Other naive_model: `as_epidist_naive_model()`, `as_epidist_naive_model.epidist_aggregate_data()`, `as_epidist_naive_model.epidist_linelist_data()`, `epidist_formula_model.epidist_naive_model()`, `is_epidist_naive_model()`, `new_epidist_naive_model()`

is_epidist_aggregate_data

Check if data has the `epidist_aggregate_data` *class*

Description

Check if data has the `epidist_aggregate_data` class

Usage

```
is_epidist_aggregate_data(data, ...)
```

Arguments

`data`	The data to convert
`...`	Additional arguments

See Also

Other aggregate_data: `as_epidist_aggregate_data()`, `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.default()`, `as_epidist_aggregate_data.epidist_linelist_data()`, `assert_epidist.epidist_aggregate_data()`, `new_epidist_aggregate_data()`

`is_epidist_latent_model`

Check if data has the `epidist_latent_model` *class*

Description

Check if data has the `epidist_latent_model` class

Usage

```
is_epidist_latent_model(data)
```

Arguments

`data`	An object

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_aggregate_data()`, `as_epidist_latent_model.epidist_linelist_data()`, `epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `new_epidist_latent_model()`

`is_epidist_linelist_data`

Check if data has the `epidist_linelist_data` *class*

Description

Check if data has the `epidist_linelist_data` class

Usage

```
is_epidist_linelist_data(data, ...)
```

Arguments

`data`	The data to convert
`...`	Additional arguments

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.default()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `assert_epidist.epidist_linelist_data()`, `new_epidist_linelist_data()`

is_epidist_marginal_model

Check if data has the `epidist_marginal_model` *class*

Description

Check if data has the `epidist_marginal_model` class

Usage

```
is_epidist_marginal_model(data)
```

Arguments

`data`	A `data.frame` containing line list data

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_aggregate_data()`, `as_epidist_marginal_model.epidist_linelist_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model(`
`new_epidist_marginal_model()`

is_epidist_naive_model

Check if data has the `epidist_naive_model` *class*

Description

Check if data has the `epidist_naive_model` class

Usage

```
is_epidist_naive_model(data)
```

Arguments

`data`	An object.

See Also

Other naive_model: `as_epidist_naive_model()`, `as_epidist_naive_model.epidist_aggregate_data()`, `as_epidist_naive_model.epidist_linelist_data()`, `epidist_formula_model.epidist_naive_model()`, `epidist_transform_data_model.epidist_naive_model()`, `new_epidist_naive_model()`

---

new_epidist_aggregate_data

Class constructor for `epidist_aggregate_data` *objects*

Description

Class constructor for `epidist_aggregate_data` objects

Usage

```
new_epidist_aggregate_data(data)
```

Arguments

`data`	A data.frame to convert

Value

An object of class `epidist_aggregate_data`

See Also

Other aggregate_data: `as_epidist_aggregate_data()`, `as_epidist_aggregate_data.data.frame()`, `as_epidist_aggregate_data.default()`, `as_epidist_aggregate_data.epidist_linelist_data()`, `assert_epidist.epidist_aggregate_data()`, `is_epidist_aggregate_data()`

Examples

```
df <- new_epidist_aggregate_data(data.frame())
class(df)
```

new_epidist_latent_model

Class constructor for `epidist_latent_model` *objects*

Description

Class constructor for `epidist_latent_model` objects

Usage

```
new_epidist_latent_model(data, ...)
```

Arguments

`data`	An object to be set with the class `epidist_latent_model`
`...`	Additional arguments passed to methods.

Value

An object of class `epidist_latent_model`

See Also

Other latent_model: `as_epidist_latent_model()`, `as_epidist_latent_model.epidist_aggregate_data()`, `as_epidist_latent_model.epidist_linelist_data()`, `epidist_family_model.epidist_latent_model()`, `epidist_formula_model.epidist_latent_model()`, `epidist_model_prior.epidist_latent_model()`, `is_epidist_latent_model()`

new_epidist_linelist_data

Class constructor for `epidist_linelist_data` *objects*

Description

Class constructor for `epidist_linelist_data` objects

Usage

```
new_epidist_linelist_data(data)
```

Arguments

`data`	A data.frame to convert

Value

An object of class `epidist_linelist_data`

See Also

Other linelist_data: `as_epidist_linelist_data()`, `as_epidist_linelist_data.data.frame()`, `as_epidist_linelist_data.default()`, `as_epidist_linelist_data.epidist_aggregate_data()`, `assert_epidist.epidist_linelist_data()`, `is_epidist_linelist_data()`

`new_epidist_marginal_model`

Class constructor for `epidist_marginal_model` *objects*

Description

Class constructor for `epidist_marginal_model` objects

Usage

```
new_epidist_marginal_model(data)
```

Arguments

`data`	A data.frame to convert

Value

An object of class `epidist_marginal_model`

See Also

Other marginal_model: `as_epidist_marginal_model()`, `as_epidist_marginal_model.epidist_aggregate_data()`, `as_epidist_marginal_model.epidist_linelist_data()`, `epidist_family_model.epidist_marginal_model()`, `epidist_formula_model.epidist_marginal_model()`, `epidist_transform_data_model.epidist_marginal_model(`
`is_epidist_marginal_model()`

new_epidist_naive_model

Class constructor for `epidist_naive_model` *objects*

Description

Class constructor for `epidist_naive_model` objects

Usage

```
new_epidist_naive_model(data)
```

Arguments

`data`	An object to be set with the class `epidist_naive_model`.

Value

An object of class `epidist_naive_model`.

See Also

Other naive_model: `as_epidist_naive_model()`, `as_epidist_naive_model.epidist_aggregate_data()`, `as_epidist_naive_model.epidist_linelist_data()`, `epidist_formula_model.epidist_naive_model()`, `epidist_transform_data_model.epidist_naive_model()`, `is_epidist_naive_model()`

predict_delay_parameters

Extract samples of the delay distribution parameters

Description

Extract samples of the delay distribution parameters

Usage

```
predict_delay_parameters(fit, newdata = NULL, ...)

predict_dpar(fit, newdata = NULL, ...)
```

Arguments

`fit`	A model fit with `epidist()`.
`newdata`	An optional data.frame for which to evaluate predictions. If `NULL` (default), the original data of the model is used. `NA` values within factors (excluding grouping variables) are interpreted as if all dummy variables of this factor are zero. This allows, for instance, to make predictions of the grand mean when using sum coding. `NA` values within grouping variables are treated as a new level.
`...`	Additional arguments passed to `brms::prepare_predictions()`.

See Also

Other postprocess: `add_mean_sd()`, `add_mean_sd.default()`, `add_mean_sd.gamma_samples()`, `add_mean_sd.lognormal_samples()`, `add_mean_sd.weibull_samples()`

`sierra_leone_ebola_data` *Ebola linelist data from Fang et al. (2016)*

Description

Linelist data for the Ebola virus collected in Sierra Leone. If you use this data in your work, please cite the corresponding paper.

Usage

```
sierra_leone_ebola_data
```

Format

A `tibble` with 8,358 rows and 8 columns:

id Unique identification number for the case

name Name as character, omitted

age Age as numeric

sex Sex as character, either "F", "M" or NA

date_of_symptom_onset The date symptoms began

date_of_sample_tested The date the sample was tested

district The district (ADM2)

chiefdom The chiefdom (ADM3)

Source

https://www.pnas.org/doi/full/10.1073/pnas.1518587113

simulate_exponential_cases *Simulate exponential cases*

Description

This function simulates cases from an exponential distribution. The user may specify the rate parameter `r`, the sample size, and the upper bound of the survival time. If the rate parameter is 0, then this function defaults to the uniform distribution.

Usage

```
simulate_exponential_cases(r = 0.2, sample_size = 10000, seed, t = 30)
```

Arguments

`r`	The exponential growth rate parameter. Defaults to 0.2.
`sample_size`	The number of cases to simulate. Defaults to 10000.
`seed`	The random seed to be used in the simulation process.
`t`	Upper bound of the survival time. Defaults to 30.

Value

A `data.frame` with two columns: `case` (case number) and `ptime` (primary event time).

See Also

Other simulate: `simulate_gillespie()`, `simulate_secondary()`, `simulate_uniform_cases()`

simulate_gillespie *Simulate cases from a stochastic SIR model*

Description

This function simulates cases from an stochastic SIR model. The user may specify the initial epidemic growth rate r, the rate of recovery gamma γ, the initial number of infected cases I_0, and the total population size N.

Usage

```
simulate_gillespie(r = 0.2, gamma = 1/7, I0 = 50, N = 10000, seed)
```

Arguments

`r`	The initial epidemic growth rate. Defaults to 0.2.
`gamma`	The rate of recovery. Defaults to 1/7.
`I0`	The initial number of infected people. Defaults to 50.
`N`	The total population size. Defaults to 10000.
`seed`	The random seed to be used in the simulation process.

Value

A `data.frame` with two columns: `case` (case number) and `ptime` (primary event time).

See Also

Other simulate: `simulate_exponential_cases()`, `simulate_secondary()`, `simulate_uniform_cases()`

`simulate_secondary` *Simulate secondary events based on a delay distribution*

Description

This function simulates secondary events based on a given delay distribution. The input dataset should have the primary event times in a column named `ptime`.

Usage

```
simulate_secondary(data, dist = rlnorm, ...)
```

Arguments

`data`	A data frame with the primary event times.
`dist`	The delay distribution to be used. Defaults to `rlnorm()`.
`...`	Arguments to be passed to the delay distribution function.

Value

A `data.frame` that augments `data` with two new columns: `delay` (secondary event latency) and `stime` (the time of the secondary event).

See Also

Other simulate: `simulate_exponential_cases()`, `simulate_gillespie()`, `simulate_uniform_cases()`

simulate_uniform_cases

Simulate cases from a uniform distribution

Description

This function simulates cases from a uniform distribution, where the primary event times are uniformly distributed between 0 and t.

Usage

```
simulate_uniform_cases(sample_size = 1000, t = 60)
```

Arguments

sample_size	The number of cases to simulate.
t	Upper bound of the uniform distribution to generate primary event times.

Value

A data.frame with two columns: case (case number) and ptime (primary event time).

See Also

Other simulate: simulate_exponential_cases(), simulate_gillespie(), simulate_secondary()

Index