Package 'SensIAT'

May 7, 2026

Title Sensitivity Analysis for Irregular Assessment Times

Version 0.3.0

Description Sensitivity analysis for trials with irregular and informative assessment times, based on a new influence function-based, augmented inverse intensity-weighted estimator.

Language en-US

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports assertthat, dplyr, generics, ggplot2, glue, KernSmooth, MASS, MAVE, methods, orthogonalsplinebasis, pracma, purrr, Rcpp (>= 1.0.12), rlang, splines, stats, survival, tibble, tidyr, utils

Suggests dfoptim, inline, ManifoldOptim, metR, progress, rmarkdown, spelling, testthat (>= 3.0.0), tidyverse

Config/testthat/edition 3

Depends R (>= 4.4.0)

LazyData true

LinkingTo Rcpp

SystemRequirements C++17

URL https://github.com/UofUEpiBio/SensIAT, https://uofuepibio.github.io/SensIAT/

BugReports https://github.com/UofUEpiBio/SensIAT/issues

NeedsCompilation yes

Author Andrew Redd [aut, cre] (ORCID: <https://orcid.org/0000-0002-6149-2438>),
Yujing Gao [aut],
Shu Yang [aut],
Bonnie Smith [aut],
Ravi Varadhan [aut],
Agatha Mallett [ctb, ctr],
Daniel Scharfstein [pdr, aut] (ORCID:

<https://orcid.org/0000-0001-7482-9653>),
University of Utah [cph]

Maintainer Andrew Redd <andrew.redd@hsc.utah.edu>

Repository CRAN

Date/Publication 2025-09-05 19:10:08 UTC

Contents

`add_terminal_observations` *Add Terminal Observations to a Dataset*

Description

This function adds terminal observations to a dataset. For each subject given by `id`, if that subject has less than the maximum number of observations, A row is added with the `end` time value, leaving all other variables as `NA`.

Usage

```
add_terminal_observations(
  data,
  id,
  time,
  end = max(pull(data, {
     {
         time
     }
 }))
)
```

Arguments

data	A data frame containing the dataset.
id	A variable in data that identifies the subject.
time	A variable in data that identifies the time of the observation.
end	The value to use for the time variable in the terminal observation. If end is less that the maximum in the dataset resulting data will be filtered such that time is less than or equal to end.

Value

A data frame with terminal observations added.

See Also

tidyr::complete()

Examples

```
exdata <- tibble::tibble(
  patient = rep(1:3, 3:5),
  day = c(0, 30, 60,
          0, 30, 60, 90,
          0, 30, 60, 90, 120),
  value = TRUE
)
add_terminal_observations(exdata, patient, day)
```

---

autoplot.SensIAT_fulldata_jackknife_results *Plot for Estimated Treatment Effect for* SensIAT_fulldata_jackknife_results *Objects*

Description

The horizontal and vertical axes represent the sensitivity parameter alpha for the control and treatment groups, respectively. The plot shows at each combination of alpha values zero if the 95% confidence interval contains zero, otherwise the bound of the confidence interval that is closest to zero.

Usage

```
## S3 method for class 'SensIAT_fulldata_jackknife_results'
autoplot(object, ..., include.rugs = NA)
```

Arguments

object	A SensIAT_fulldata_jackknife_results object.
...	Additional arguments passed to predict.
include.rugs	If TRUE, adds rugs to the plot. If FALSE, no rugs are added. When NA, rugs are added only if the number of distinct values of alpha_control and alpha_treatment is less than or equal to 10.

Examples

```
# Note: fitting the jackknife is computationally expensive,
#       so this example is here for reference.
## Not run:
full.object <-
    fit_SensIAT_fulldata_model(
        data = SensIAT_example_fulldata,
        trt = Treatment_group == 'treatment',
        outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        knots = c(60, 260, 460),
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6)
    )
jk.full.model <- jackknife(full.object, time = 180)
ggplot2::autoplot(jk.full.model)

## End(Not run)
```

---

autoplot.SensIAT_fulldata_model

Plot for Estimated Treatment Effect for SensIAT_fulldata_model *Objects*

Description

The horizontal and vertical axes represent the sensitivity parameter alpha for the control and treatment groups, respectively. The contour plot shows the estimated treatment effect at each combination of alpha values.

Usage

```
## S3 method for class 'SensIAT_fulldata_model'
autoplot(object, time, include.rugs = NA, ...)
```

Arguments

object	A SensIAT_fulldata_model object.
time	Time at which to plot the estimates.
include.rugs	If TRUE, adds rugs indicating the locations where the sensitivity was evaluated to the plot. If FALSE, no rugs are added. If NA, rugs are added only if the number of distinct values of alpha_control and alpha_treatment is less than or equal to 10.
...	Additional arguments passed to predict.

Value

A ggplot2 object.

Examples

```
full.object <-
    fit_SensIAT_fulldata_model(
        data = SensIAT_example_fulldata,
        trt = Treatment_group == 'treatment',
        outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        knots = c(60, 260, 460),
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6)
    )
ggplot2::autoplot(full.object, time = 180)
```

---

autoplot.SensIAT_withingroup_jackknife_results *Plot Estimates at Given Times for* SensIAT_withingroup_jackknife_results *Objects*

Description

Horizontal axis represents time, and the vertical axis represents the outcome from the model. Point plotted is the mean estimate, and the error bars show the 95% confidence interval using the variance estimated from the jackknife.

Usage

```
## S3 method for class 'SensIAT_withingroup_jackknife_results'
autoplot(object, width = NULL, ...)
```

Arguments

object	A SensIAT_withingroup_jackknife_results object produced from SensIAT_jackknife.
width	Width of the dodge for position, default is half the minimum distance between time evaluation points.
...	Ignored.

Value

A ggplot2 object.

Examples

```
# Note: fitting the jackknife is computationally expensive,
#       so this example is here for reference.
## Not run:
fitted <-
fit_SensIAT_within_group_model(
    group.data = SensIAT_example_data,
    outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
    alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
    id = Subject_ID,
    outcome = Outcome,
    time = Time,
    intensity.args=list(bandwidth = 30),
    knots = c(60,260,460),
    End = 830
)
jackknife.estimates <- SensIAT_jackknife(fitted, time = c(90, 180, 270, 360, 450))
ggplot2::autoplot(jackknife.estimates)

## End(Not run)
```

---

autoplot.SensIAT_within_group_model

Plot a SensIAT_within_group_model *Object*

Description

This creates a line plot for a SensIAT_within_group_model object. The horizontal axis represents time, and the vertical axis represents the expected marginal outcome given the sensitivity parameter alpha.

Usage

```
## S3 method for class 'SensIAT_within_group_model'
autoplot(object, ...)
```

Arguments

object	A SensIAT_within_group_model object.
...	currently ignored

Value

A ggplot2 object.

Examples

```
# Note: example takes a few seconds to run.

object <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = fit_SensIAT_single_index_fixed_bandwidth_model,
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        knots = c(60,260,460),
        End = 830,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        intensity.args=list(bandwidth=30)
    )
ggplot2::autoplot(object) +
    # Title not included
    ggplot2::ggtitle("SensIAT within group model") +
    # Nor are bounds on reasonable values of alpha
    ggplot2::geom_hline(yintercept = c(1.2, 3), linetype = "dotted", linewidth = 1.5)
```

---

compute_influence_terms

Compute Influence Terms

Description

This function computes the influence terms for the marginal outcome model sensitivity analysis. It is a generic function that can handle different types of outcome models.

Usage

```
compute_influence_terms(outcome.model, intensity.model, alpha, data, ...)

## Default S3 method:
compute_influence_terms(
  outcome.model,
  intensity.model,
```

```
  alpha,
  data,
  id,
  base,
  ...
)

## S3 method for class '`SensIAT::Single-index-outcome-model`'
compute_influence_terms(
  outcome.model,
  intensity.model,
  alpha,
  data,
  base,
  tolerance = .Machine$double.eps^(1/3),
  na.action = na.fail,
  id = NULL,
  time = NULL,
  ...
)
```

Arguments

`outcome.model`	The outcome model fitted to the data.
`intensity.model`	The intensity model fitted to the data.
`alpha`	A numeric vector representing the sensitivity parameter.
`data`	A data frame containing the observations.
`...`	Additional arguments passed to the method.
`id`	A variable representing the patient identifier.
`base`	A spline basis object.
`tolerance`	Numeric value indicating the tolerance for integration, default is `.Machine$double.eps^(1/3)`.
`na.action`	Function to handle missing values, default is `na.fail`.
`time`	Variable indicating the time variable in the data, by Default will be extracted from the intensity model response.

Methods (by class)

- `compute_influence_terms(default)`: Generic method, which throws a not implemented error.
- `compute_influence_terms(`SensIAT::Single-index-outcome-model`)`: Optimized method for the single index model.

compute_SensIAT_expected_values *Compute Conditional Expected Values based on Outcome Model*

Description

Compute Conditional Expected Values based on Outcome Model

Usage

```
compute_SensIAT_expected_values(
  model,
  alpha = 0,
  new.data = model.frame(model),
  ...
)

## S3 method for class 'lm'
compute_SensIAT_expected_values(model, alpha, new.data, ...)

## S3 method for class 'glm'
compute_SensIAT_expected_values(
  model,
  alpha,
  new.data,
  ...,
  y.max = NULL,
  eps = .Machine$double.eps
)

## S3 method for class 'negbin'
compute_SensIAT_expected_values(
  model,
  alpha,
  new.data,
  ...,
  y.max = NULL,
  eps = .Machine$double.eps^(1/4)
)
```

Arguments

model	An object representing the output of the outcome model.
alpha	The sensitivity parameter
new.data	Data to compute conditional means for, defaults to the model frame for the fitted model.

...	passed onto methods.
y.max	The maximum value of the outcome variable for the Poisson and Negative Binomial models. If omitted it is chosen from the quantile function for the distribution at 1-eps.
eps	The tolerance for the quantile function used to estimate y.max, default is .Machine$double.eps.

Details

Compute the conditional expectations needed for predictions in the models. Two additional values/expectations are computed:

- $E \big[Y(t) \exp \{ \alpha Y(t) \} | A(t)=1, \bar{O}(t) \big]$, returned as E_Yexp_alphaY, and
- $E \big[\exp \{ \alpha Y(t) \} \ | A(t)=1, \bar{O}(t) \big]$, returned as E_exp_alphaY.

For the methods shown here

Value

The new.data frame with additional columns alpha, E_Yexp_alphaY, and E_exp_alphaY appended.

Methods (by class)

- compute_SensIAT_expected_values(lm): (Gaussian) Linear Model method The stats::integrate method is used to compute the conditional expectations.
- compute_SensIAT_expected_values(glm): Generalized Linear Model method
- compute_SensIAT_expected_values(negbin): Negative Binomial Model method

Examples

```
model <- lm(mpg ~ as.factor(cyl)+disp+wt, data=mtcars)
compute_SensIAT_expected_values(model, alpha= c(-0.3, 0, 0.3), new.data = mtcars[1:5, ])
model <- glm(cyl ~ mpg+disp+wt, data=mtcars, family=poisson())
compute_SensIAT_expected_values(model, alpha= c(-0.3, 0, 0.3), new.data = mtcars[1:5, ]) |>
    dplyr::mutate('E(y|alpha)' = .data$E_Yexp_alphaY/.data$E_exp_alphaY)
```

---

fit_SensIAT_fulldata_model

Produce fitted model for group (treatment or control)

Description

Produces a fitted model that may be used to produce estimates of mean and variance for the given group.

Usage

```
fit_SensIAT_fulldata_model(data, trt, ...)

fit_SensIAT_within_group_model(
  group.data,
  outcome_modeler,
  id,
  outcome,
  time,
  knots,
  alpha = 0,
  End = NULL,
  intensity.args = list(),
  outcome.args = list(),
  influence.args = list(),
  spline.degree = 3,
  add.terminal.observations = TRUE
)
```

Arguments

data	the full data set.
trt	an expression that determine what is treated as the treatment. Everything not treatment is considered control.
...	common arguments passed to `fit_SensIAT_within_group_model`.
group.data	The data for the group that is being analyzed. Preferably passed in as a single `tibble` that internally is subsetted/filtered as needed.
outcome_modeler	function for fitting the outcome model. Called with a formula, data argument and `outcome.args` list.
id	The variable that identifies the patient.
outcome	The variable that contains the outcome.
time	The variable that contains the time.
knots	knot locations for defining the spline basis.
alpha	The sensitivity parameter.
End	The end time for this data analysis, we need to set the default value as the max value of the time.
intensity.args	A list of optional arguments for intensity model. See the Intensity Arguments section.
outcome.args	parameters as needed passed into the `outcome_modeler`. One special element may be `'model.modifications'` which, if present, should be a formula that will be used to modify the outcome model per, update.formula.
influence.args	A list of optional arguments used when computing the influence. See the Influence Arguments section.

`spline.degree` The degree of the spline basis.

`add.terminal.observations`
Logical indicating whether to add terminal observations to the data. If TRUE, data may not contain any `NA`s. if FALSE, data will be assumed to already include the terminal observations

Details

This function should be agnostic to whether it is being provided a treatment or control group.

Value

a list with class `SensIAT-fulldata-fitted-model` with two components, `control` and `treatment`, each of which is an independently fitted `SensIAT-within-group-fitted-model` fit with the fit_within_group_model function.

Should return everything needed to define the fit of the model. This can then be used for producing the estimates of mean, variance, and in turn treatment effect. For the full data model a list with two models one each for the treatment and control groups.

Functions

- `fit_SensIAT_fulldata_model()`: Fit the sensitivity analysis for both treatment and control groups.

Intensity Arguments

The `intensity.args` list may contain the following elements:

- `model.modifications` A formula that will be used to modify the intensity model from it's default, per update.formula.
- `kernel` The kernel function for the intensity model. Default is the Epanechnikov kernel.
- `bandwidth` The bandwidth for the intensity model kernel.

Influence Arguments

The `influence.args` list may contain the following elements:

- `method` The method for integrating, adaptive or fixed quadrature. Default is `'adaptive'`.
- `tolerance` The tolerance when using adaptive quadrature.
- `delta` The bin width for fixed quadrature.
- `resolution` alternative to `delta` by specifying the number of bins.
- `fix_discontinuity` Whether to account for the discontinuity in the influence at observation times.

Examples

```
model <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        End = 830,
        knots = c(60,260,460),
    )
```

---

fit_SensIAT_marginal_mean_model

Fit the Marginal Means Model

Description

Fit the Marginal Means Model

Usage

```
fit_SensIAT_marginal_mean_model(
  data,
  id,
  alpha,
  knots,
  outcome.model,
  intensity.model,
  spline.degree = 3L,
  ...
)
```

Arguments

data	Data for evaluation of the model. Should match the data used to fit the intensity and outcome models.
id	The subject identifier variable in the data. Lazy evaluation is used, so it can be a symbol or a string.
alpha	Sensitivity parameter, a vector of values.
knots	Location of spline knots. If a SplineBasis object is provided, it is used directly.
outcome.model	The observed effects model.
intensity.model	The assessment time intensity model.
spline.degree	The degree of the spline basis, default is 3 (cubic splines).
...	Additional arguments passed to compute_influence_terms.

Value

a list with the fitted model, including the coefficients and their variances for each alpha value.

Examples

```
# Note: example takes approximately 30 seconds to run.

library(survival)
library(dplyr)
library(splines)
# Create followup data with lags
# added variables `..prev_time..`, `..delta_time..` and `..prev_outcome..`
# have special interpretations when computing the influence.
data_with_lags <- SensIAT_example_data |>
    dplyr::group_by(Subject_ID) |>
    dplyr::mutate(
        ..prev_outcome.. = dplyr::lag(Outcome, default = NA_real_, order_by = Time),
        ..prev_time.. = dplyr::lag(Time, default = 0, order_by = Time),
       ..delta_time.. = Time - dplyr::lag(.data$Time, default = NA_real_, order_by = Time)
    )

# Create the observation time intensity model
intensity.model <-
    coxph(Surv(..prev_time.., Time, !is.na(Outcome)) ~ ..prev_outcome.. + strata(Visit),
    data = data_with_lags |> dplyr::filter(.data$Time > 0))

# Create the observed outcome model
outcome.model <-
    fit_SensIAT_single_index_fixed_coef_model(
        Outcome ~ ns(..prev_outcome.., df=3) + ..delta_time.. - 1,
        id = Subject_ID,
        data = data_with_lags |> filter(Time > 0))

# Fit the marginal outcome model
mm <- fit_SensIAT_marginal_mean_model(
    data = data_with_lags,
    id = Subject_ID,
    alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
    knots = c(60, 260, 460),
    intensity.model = intensity.model,
    time.vars = c('..delta_time..'),
    outcome.model = outcome.model)
```

`fit_SensIAT_single_index_fixed_coef_model`
Outcome Modeler for `SensIAT` *Single Index Model.*

Description

Outcome Modeler for SensIAT Single Index Model.

Usage

```
fit_SensIAT_single_index_fixed_coef_model(
  formula,
  data,
  kernel = "K2_Biweight",
  method = "nmk",
  id = ..id..,
  initial = NULL,
  ...
)

fit_SensIAT_single_index_fixed_bandwidth_model(
  formula,
  data,
  kernel = "K2_Biweight",
  method = "nmk",
  id = ..id..,
  initial = NULL,
  ...
)
```

Arguments

formula	The outcome model formula
data	The data to fit the outcome model to. Should only include follow-up data, i.e. time > 0.
kernel	The kernel to use for the outcome model.
method	The optimization method to use for the outcome model, either "optim", "nlminb", or "nmk".
id	The patient identifier variable for the data.
initial	Either a vector of initial values or a function to estimate initial values. If NULL (default), the initial values are estimated using the MAVE::mave.compute function.
...	Currently ignored, included for future compatibility.

Value

Object of class SensIAT::Single-index-outcome-model which contains the outcome model portion.

Functions

- fit_SensIAT_single_index_fixed_bandwidth_model(): for fitting with a fixed bandwidth

Examples

```
# A basic example using fixed intensity bandwidth.
object <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = fit_SensIAT_single_index_fixed_bandwidth_model,
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        knots = c(60,260,460),
        End = 830,
        intensity.args=list(bandwidth=30)
    )

# A basic example using variable bandwidth but with fixed first coefficient.
object.bw <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        knots = c(60,260,460),
        End = 830,
        intensity.args=list(bandwidth=30)
    )
```

---

fit_SensIAT_single_index_norm1coef_model

Single Index Model using MAVE and Optimizing Bandwidth.

Description

Single index model estimation using minimum average variance estimation (MAVE). A direction is estimated using MAVE, and then the bandwidth is selected by minimization of the cross-validated pseudo-integrated squared error. Optionally, the initial coefficients of the outcome model can be re-estimated by optimization on a spherical manifold. This option requires the ManifoldOptim package.

Usage

```
fit_SensIAT_single_index_norm1coef_model(
  formula,
  data,
  kernel = "K2_Biweight",
  mave.method = "meanMAVE",
  id = ..id..,
```

```
  bw.selection = c("ise", "mse"),
  bw.method = c("optim", "grid", "optimize"),
  bw.range = c(0.01, 1.5),
  reestimate.coef = 0,
  ...
)
```

Arguments

formula	The outcome model formula
data	The data to fit the outcome model to. Should only include follow-up data, i.e. time > 0.
kernel	The kernel to use for the outcome model.
mave.method	The method to use for the MAVE estimation.
id	The patient identifier variable for the data.
bw.selection	The criteria for bandwidth selection, either 'ise' for Integrated Squared Error or 'mse' for Mean Squared Error.
bw.method	The method for bandwidth selection, either 'optim' for using optimization or 'grid' for grid search.
bw.range	A numeric vector of length 2 indicating the range of bandwidths to consider for selection as a multiple of the standard deviation of the single index predictor.
reestimate.coef	number of iterations to go through.
...	Additional arguments to be passed to optim.

Value

Object of class SensIAT::Single-index-outcome-model which contains the outcome model portion.

---

jackknife *Perform Jackknife Resampling on an Object*

Description

Perform Jackknife Resampling on an Object

Usage

```
jackknife(object, ...)

## S3 method for class 'SensIAT_within_group_model'
jackknife(object, time, ...)

## S3 method for class 'SensIAT_fulldata_model'
jackknife(object, time, ...)
```

Arguments

object	An object to cross validate on.
...	Additional arguments passed to the method.
time	Time points for which to estimate the response.

Value

A data frame of the jackknife resampling results. For SensIAT objects, a tibble with columns alpha, time, jackknife_mean, and jackknife_var, where jackknife_mean is the mean of the jackknife estimates and jackknife_var is the estimated variances of the response at the given time points for the specified alpha values.

Methods (by class)

- jackknife(SensIAT_within_group_model): Perform jackknife resampling on a SensIAT_within_group_model object.
- jackknife(SensIAT_fulldata_model): Perform jackknife resampling on a SensIAT_fulldata_model object.

Examples

```
## Not run:
object <-
fit_SensIAT_within_group_model(
    group.data = SensIAT_example_data,
    outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
    alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
    id = Subject_ID,
    outcome = Outcome,
    time = Time,
    intensity.args=list(bandwidth = 30),
    knots = c(60,260,460),
    End = 830
)
jackknife.estimates <- SensIAT_jackknife(object, time = c(90, 180, 270, 360, 450))

## End(Not run)
```

---

predict.SensIAT_fulldata_model

Give the Marginal Mean Estimate and its Estimated Asymptotic Variance

---

Description

Give the marginal mean model estimate

Usage

```
## S3 method for class 'SensIAT_fulldata_model'
predict(object, time, ...)

## S3 method for class 'SensIAT_within_group_model'
predict(object, time, include.var = TRUE, ..., base = object$base)
```

Arguments

object	SensIAT_within_group_model object
time	Time points of interest
...	Currently ignored.
include.var	Logical. If TRUE, the variance of the outcome is also returned
base	A SplineBasis object used to evaluate the basis functions.

Value

If include.var is TRUE, a tibble with columns time, mean, and var is returned. otherwise if include.var is FALSE, only the mean vector is returned.

Functions

- predict(SensIAT_fulldata_model): For each combination of time and alpha estimate the mean response and variance for each group as well as estimate the mean treatment effect and variance.

Examples

```
model <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = fit_SensIAT_single_index_fixed_coef_model,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        id = Subject_ID,
        outcome = Outcome,
        time = Time,
        End = 830,
        knots = c(60,260,460),
    )
predict(model, time = c(90, 180))
```

`prepare_SensIAT_data` *Prepare Data for Sensitivity Analysis with Irregular Assessment Times*

Description

This function prepares the data for SensIAT analysis by transforming it into a format suitable for the SensIAT models.

Usage

```
prepare_SensIAT_data(
  data,
  id.var,
  time.var,
  outcome.var,
  End,
  add.terminal.observations = TRUE
)
```

Arguments

`data`	A data frame containing the data to be prepared.
`id.var`	The variable in `data` that identifies the subject.
`time.var`	The variable in `data` that identifies the time of the observation.
`outcome.var`	The variable in `data` that contains the outcome of interest.
`End`	The end time for the analysis. Observations with time greater than `End` will be filtered out.
`add.terminal.observations`	Logical indicating whether to add terminal observations to the data (`TRUE`), or terminal observations have already been added (`FALSE`).

Value

A data frame with the following transformations:

- Data filtered to time less than or equal to `End`.
- Observations are arranged by `id.var` and `time.var`.
- Terminal observations added if `add.terminal.observations` is `TRUE`, with `..time..` set to `End` and `..outcome..` set to `NA`, if the subject has less observations than the maximum number of observations.
- New variables created:
  - `..id..` aliases `id.var`,
  - `..time..` aliases `time.var`,
  - `..outcome..` aliases `outcome.var`,

- ..visit_number.. is the visit number within each subject derived from time.var,
- ..prev_outcome.., i.e. lag-outcome, the outcome from the previous visit,
- ..prev_time.., i.e. lag-time, the time from the previous visit,
- ..delta_time.., the difference in time between the current and previous visit.

Examples

```
prepare_SensIAT_data( SensIAT_example_data, Subject_ID, Time, Outcome, 830)

exdata <- tibble::tibble(ID=rep(1:2, c(3,5)),
                         Time=c(0, 30, 60,
                                0, 30, 60, 90, 120),
                         Outcome=floor(runif(8, 1, 100)))

prepare_SensIAT_data(exdata, ID, Time, Outcome, 120)
```

SensIAT_example_data *SensIAT Example Data*

Description

A simulated dataset for use in the SensIAT tutorial, testing and documentation.

Usage

```
SensIAT_example_data

SensIAT_example_fulldata
```

Format

SensIAT_example_data is a data frame with 779 rows and 4 variables consisting of 200 simulated patients. Each row in the data represents a visit for the patient. The columns are:

Subject_ID A unique identifier for each patient.

Visit The ordinal number of the visit for the patient. Baseline observation is 0.

Time The time of the visit in days, since baseline.

Outcome The outcome of interest.

SensIAT_example_fulldata is a data frame with 1614 rows and 5 variables consisting of 400 simulated patients, 200 for each treatment arm. Each row in the data represents a visit for the patient. The columns are:

Subject_ID A unique identifier for each patient.

Visit The ordinal number of the visit for the patient. Baseline observation is 0.

Time The time of the visit in days, since baseline.

Outcome The outcome of interest.

Treatment_group Treatment or control group.

Functions

- `SensIAT_example_fulldata`: A simulated dataset with both treatment and control groups.

Index