Package ‘staggered’

July 23, 2025

Title Efficient Estimation Under Staggered Treatment Timing

Version 1.2.2

Description
Efficiently estimates treatment effects in settings with randomized staggered rollouts, using tools proposed by Roth and Sant'Anna (2023) <doi:10.48550/arXiv.2102.01291>.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports data.table, purrr, Rcpp, RcppEigen, magrittr, MASS, stats

LinkingTo Rcpp, RcppEigen

Depends R (>= 3.5.0)

NeedsCompilation yes

Author Jonathan Roth [aut],
Pedro H.C. Sant'Anna [aut, cre]

Maintainer Pedro H.C. Sant'Anna <pedrohcgs@gmail.com>

Repository CRAN

Date/Publication 2025-01-09 19:00:02 UTC

Contents

`balance_checks` *Calculate balance checks in staggered rollout designs*

Description

This functions calculates Wald-tests for balance in staggered rollout designs proposed by Roth and Sant'Anna.

Usage

```
balance_checks(
  df,
  i = "i",
  t = "t",
  g = "g",
  y = "y",
  estimand = NULL,
  A_0_list = NULL,
  eventTime = 0,
  use_DiD_A0 = NULL,
  use_last_treated_only = FALSE,
  compute_fisher = FALSE,
  num_fisher_permutations = 500,
  return_full_vcv = FALSE,
  skip_data_check = FALSE,
  seed = NULL
)
```

Arguments

`df` A data frame containing panel data with the variables y (an outcome), i (an individual identifier), t (the period in which the outcome is observe), g (the period in which i is first treated, with Inf denoting never treated)

`i` The name of column containing the individual (cross-sectional unit) identifier. Default is "i".

`t` The name of the column containing the time periods. Default is "t".

`g` The name of the column containing the first period when a particular observation is treated, with Inf denoting never treated. Default is "g".

`y` The name of the column containing the outcome variable. Default is "y".

`estimand` The estimand to be calculated: "simple" averages all treated (t,g) combinations with weights proportional to N_g; "cohort" averages the ATEs for each cohort g, and then takes an N_g-weighted average across g; "calendar" averages ATEs for each time period, weighted by N_g for treated units, and then averages across time. "eventstudy" returns the average effect at the ”event-time” given in the parameter EventTime. The parameter can be left blank if a custom parameter is provided in A_theta_list. The argument is not case-sensitive.

`A_0_list` This parameter allow for specifying the matrices used to construct the Xhat vector of pre-treatment differences. If left NULL, the default is to use the scalar set of controls used in Callaway and Sant'Anna. If use_DiD_A0 = FALSE, then it uses the full vector possible comparisons of (g,g') in periods t<g,g'.

`eventTime` If using estimand = "eventstudy", specify what eventTime you want the event-study parameter for. The default is 0, the period in which treatment occurs. If a vector is provided, estimates are returned for all the event-times in the vector.

`use_DiD_A0` If this parameter is true, then Xhat corresponds with the scalar used by Callaway and Sant'Anna, so the Callaway and Sant'Anna estimator corresponds with beta=1. If it is false, the Xhat is a vector with all possible comparisons of pairs of cohorts before either is treated. The latter option should only be used when the number of possible comparisons is small relative to sample size.

`use_last_treated_only`
If true, then A_0_list and A_theta_list are created to only make comparisons with the last treated cohorts (as suggested by Sun and Abraham), rather than using not-yet-treated units as comparisons. If set to TRUE (and use_DiD_A0 = TRUE), then beta=1 corresponds with the Sun and Abraham estimator.

`compute_fisher` If true, computes a Fisher Randomization Test using the studentized estimator.

`num_fisher_permutations`
The number of permutations to use in the Fisher Randomization Test (if compute_fisher = TRUE). Default is 500.

`return_full_vcv`
If this is true, then the function returns a list containing the full variance-covariance matrix for all Xhats.

`skip_data_check`
If true, skips checks that the data is balanced and contains the colums i,t,g,y. Used in internal recursive calls to increase speed, but not recommended for end-user.

`seed` Set seed for permutations

Value

resultsDF A data.frame containing: estimate (the point estimate), se (the standard error), and se_neyman (the Neyman standard error). If a vector-valued eventTime is provided, the data.frame contains multiple rows for each eventTime and an eventTime column. If return_full_vcv = TRUE and estimand = "eventstudy", the function returns a list containing resultsDF and the full variance covariance for the event-study estimates (vcv) as well as the Neyman version of the covariance matrix (vcv_neyman). (If return_matrix_list = TRUE, it likewise returns a list containing lists of matrices used in the vcv calculation.)

References

Roth, Jonatahan, and Sant'Anna, Pedro H. C. (2021), 'Efficient Estimation for Staggered Rollout Designs', arXiv: 2102.01291, `https://arxiv.org/abs/2102.01291`.

Examples

```
set.seed(1234)
# load the officer data and subset it
df <- pj_officer_level_balanced
group_random <- sample(unique(df$assigned), 3)
df <- df[df$assigned %in% group_random,]
# Calculate balance checks for simple aggregation
balance_checks(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "simple")
# Calculate balance checks for the cohort weighted average
balance_checks(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "cohort")
# Calculate balance checks for the calendar weighted average
balance_checks(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "calendar")
# Calculate balance checks for event-study aggregation for the first 24 months
# (month 0 is instantaneous effect)
event_bal_checks <- balance_checks(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "eventstudy",
  eventTime = 0:23)
head(event_bal_checks)
```

---

compute_balance_test *Wald-test if Xhat is statistically different from zero*

Description

balance_check computes the Wald test-statistic (F-test) for the null that E[X]=0

Usage

```
compute_balance_test(Ybar_g_list, A_0_list, S_g_list, N_g_list)
```

Arguments

`Ybar_g_list`	Ybar_g_list
`A_0_list`	A_0_list
`S_g_list`	S_g_list
`N_g_list`	N_g_list

Value

Wald_test_Xhat Wald-test statistic for the balance test.

`compute_Betastar` *Plug-in efficient Beta hat*

Description

`compute_Betastar` computes the plug-in efficient betahat

Usage

```
compute_Betastar(
  Ybar_g_list,
  A_theta_list,
  A_0_list,
  S_g_list,
  N_g_list,
  Xvar_list = NULL
)
```

Arguments

`Ybar_g_list`	Ybar_g_list
`A_theta_list`	A_theta_list
`A_0_list`	A_0_list
`S_g_list`	S_g_list
`N_g_list`	N_g_list
`Xvar_list`	Xvar_list

Value

betastar Vector of plug-in efficient betahat estimates.

`compute_g_level_summaries` *Calculate group level summary statistics*

Description

This function computes the mean-vector and covariance matrix of the outcomes for each cohort, where a cohort g is a group of units first treated in period g

Usage

```
compute_g_level_summaries(df, is_balanced = TRUE)
```

Arguments

`df`	A data table containing panel data with the variables y (an outcome), i (an individual identifier), t (the period in which the outcome is observe), g (the period in which i is first treated, with Inf denoting never treated)
`is_balanced`	If true, the df has previously been balanced so this does not need to be done internally.

Value

Y_bar_list A list of the means of the outcomes for each cohort g

S_g_list A list of covariance matrices for the outcomes for each cohort g

N_g_DT A data table of the number of observations for each cohort g

g_list A list of when the cohorts were first treated

t_list A list of the the time periods for the outcome. The vector of outcomes corresponds with this order.

`compute_Xhat` *Compute Xhat of pre-treatment differences*

Description

`compute_Xhat` computes the vector Xhat of pre-treatment differences given the list of cohort means Ybar_g_list and the list of matrices A_0_list

Usage

```
compute_Xhat(Ybar_g_list, A_0_list)
```

Arguments

`Ybar_g_list`	Ybar_g_list
`A_0_list`	A_0_list

Value

Xhat the vector Xhat of pre-treatment differences to be used as regressors

`create_A0_list` *create_A0_list*

Description

`create_A0_list` creates the list of A_0 matrices for Xhat corresponding with all possible comparisons of cohorts before they are treated

Usage

```
create_A0_list(g_list, t_list)
```

Arguments

`g_list`	g_list
`t_list`	t_list

Value

A0_list list of A_0 matrices for Xhat corresponding with all possible comparisons of cohorts before they are treated

`pj_officer_level_balanced` *Procedural Justice Training Program in the Chicago Police Department*

Description

Data from a large-scale procedural justice training program in the Chicago Police Department analyzed by Wood, Tyler, Papachristos, Roth and Sant'Anna (2020) and Roth and Sant'Anna (2021). The data contains a balanced panel of 7,785 police officers in Chicago who were randomly given a procedural justice training on different dates, and who remained in the police force throughout the study period (from January 2011 to December 2016).

Usage

```
pj_officer_level_balanced
```

Format

A data frame with 560520 observations (7,785 police officers and 72 months) and 12 variables:

uid identifier for the police officer

month month and year of the observation

assigned month-year of first training assignment

appointed appointment date

resigned Date the police officer resigned. NA if he/she did not resigned by the time data was collected

birth_year Officer's year of birth

assigned_exact Exact date of first training assignment

complaints Number of complaints (setlled and sustained)

sustained Number of sustained complaints

force Number of times force was used

period Time period: 1 - 72

first_trained Time period first exposed to treatment (Treatment cohort/group)

Source

Wood, Tyler, Papachristos, Roth and Sant'Anna (2020) and Roth and Sant'Anna (2021).

References

Roth, Jonatahan, and Sant'Anna, Pedro H. C. (2021), 'Efficient Estimation for Staggered Rollout Designs', arXiv: 2102.01291, `https://arxiv.org/abs/2102.01291`.

Wood, George, Tyler, Tom R., Papachristos, Andrew P., Roth, Jonathan and Sant'Anna, Pedro H. C. (2020), 'Revised findings for "Procedural justice training reduces police use of force and complaints against officers", doi:10.31235/osf.io/xf32m.

`staggered` *Calculate the efficient adjusted estimator in staggered rollout designs*

Description

This functions calculates the efficient estimator for staggered rollout designs proposed by Roth and Sant'Anna.

Usage

```
staggered(
  df,
  i = "i",
  t = "t",
  g = "g",
  y = "y",
  estimand = NULL,
  A_theta_list = NULL,
  A_0_list = NULL,
  eventTime = 0,
  beta = NULL,
  use_DiD_A0 = ifelse(is.null(A_0_list), TRUE, FALSE),
  return_full_vcv = FALSE,
  use_last_treated_only = FALSE,
  compute_fisher = FALSE,
  num_fisher_permutations = 500,
  skip_data_check = FALSE
)
```

Arguments

`df` A data frame containing panel data with the variables y (an outcome), i (an individual identifier), t (the period in which the outcome is observe), g (the period in which i is first treated, with Inf denoting never treated)

`i` The name of column containing the individual (cross-sectional unit) identifier. Default is "i".

`t` The name of the column containing the time periods. Default is "t".

`g` The name of the column containing the first period when a particular observation is treated, with Inf denoting never treated. Default is "g".

`y` The name of the column containing the outcome variable. Default is "y".

`estimand` The estimand to be calculated: "simple" averages all treated (t,g) combinations with weights proportional to N_g; "cohort" averages the ATEs for each cohort g, and then takes an N_g-weighted average across g; "calendar" averages ATEs for each time period, weighted by N_g for treated units, and then averages across time. "eventstudy" returns the average effect at the ”event-time” given in the parameter EventTime. The parameter can be left blank if a custom parameter is provided in A_theta_list. The argument is not case-sensitive.

`A_theta_list` This parameter allows for specifying a custom estimand, and should be left as NULL if estimand is specified. It is a list of matrices A_theta_g so that the parameter of interest is sum_g A_theta_g Ybar_g, where Ybar_g = 1/N sum_i Y_i(g)

`A_0_list` This parameter allow for specifying the matrices used to construct the Xhat vector of pre-treatment differences. If left NULL, the default is to use the scalar set of controls used in Callaway and Sant'Anna. If use_DiD_A0 = FALSE, then it uses the full vector possible comparisons of (g,g') in periods t<g,g'.

`eventTime` If using estimand = "eventstudy", specify what eventTime you want the event-study parameter for. The default is 0, the period in which treatment occurs. If a vector is provided, estimates are returned for all the event-times in the vector.

`beta` A coefficient to use for covariate adjustment. If not specified, the plug-in optimal coefficient is used. beta =0 corresponds with the simple difference-in-means. beta = 1 corresponds with the Callaway and Sant'Anna estimator when using the default value of use_DiD_A0 = TRUE.

`use_DiD_A0` If this parameter is true, then Xhat corresponds with the scalar used by Callaway and Sant'Anna, so the Callaway and Sant'Anna estimator corresponds with beta=1. If it is false, the Xhat is a vector with all possible comparisons of pairs of cohorts before either is treated. The latter option should only be used when the number of possible comparisons is small relative to sample size.

`return_full_vcv`
If this is true and estimand = "eventstudy", then the function returns a list containing the full variance-covariance matrix for the event-plot estimates in addition to the usual dataframe with the estimates

`use_last_treated_only`
If true, then A_0_list and A_theta_list are created to only make comparisons with the last treated cohorts (as suggested by Sun and Abraham), rather than using not-yet-treated units as comparisons. If set to TRUE (and use_DiD_A0 = TRUE), then beta=1 corresponds with the Sun and Abraham estimator.

`compute_fisher` If true, computes a Fisher Randomization Test using the studentized estimator.

`num_fisher_permutations`
The number of permutations to use in the Fisher Randomization Test (if compute_fisher = TRUE). Default is 500.

`skip_data_check`
If true, skips checks that the data is balanced and contains the colums i,t,g,y. Used in internal recursive calls to increase speed, but not recommended for end-user.

Value

resultsDF A data.frame containing: estimate (the point estimate), se (the standard error), and se_neyman (the Neyman standard error). If a vector-valued eventTime is provided, the data.frame contains multiple rows for each eventTime and an eventTime column. If return_full_vcv = TRUE and estimand = "eventstudy", the function returns a list containing resultsDF and the full variance covariance for the event-study estimates (vcv) as well as the Neyman version of the covariance matrix (vcv_neyman).

References

Roth, Jonatahan, and Sant'Anna, Pedro H. C. (2021), 'Efficient Estimation for Staggered Rollout Designs', arXiv: 2102.01291, https://arxiv.org/abs/2102.01291.

Examples

```
set.seed(1234)
```

```
# load the officer data and subset it
df <- pj_officer_level_balanced
group_random <- sample(unique(df$assigned), 3)
df <- df[df$assigned %in% group_random,]
# Calculate efficient estimator for the simple weighted average
staggered(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "simple")
# Calculate efficient estimator for the cohort weighted average
staggered(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "cohort")
# Calculate efficient estimator for the calendar weighted average
staggered(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "calendar")
# Calculate event-study coefficients for the first 24 months
# (month 0 is instantaneous effect)
eventPlotResults <- staggered(df = df,
  i = "uid",
  t = "period",
  g = "first_trained",
  y = "complaints",
  estimand = "eventstudy",
  eventTime = 0:23)
head(eventPlotResults)
```

staggered_cs *Calculate the Callaway & Sant'Anna (2021) estimator for staggered rollouts*

Description

This functions calculates the Callaway & Sant'Anna (2021) estimator for staggered rollout designs using not-yet-treated units (including never-treated, if available) as controls.

Usage

```
staggered_cs(
    df,
    i = "i",
    t = "t",
    g = "g",
    y = "y",
    estimand = NULL,
    A_theta_list = NULL,
    A_0_list = NULL,
    eventTime = 0,
    return_full_vcv = FALSE,
    compute_fisher = FALSE,
    num_fisher_permutations = 500,
    skip_data_check = FALSE
  )
```

Arguments

`df`
A data frame containing panel data with the variables y (an outcome), i (an individual identifier), t (the period in which the outcome is observe), g (the period in which i is first treated, with Inf denoting never treated)

`i`
The name of column containing the individual (cross-sectional unit) identifier. Default is "i".

`t`
The name of the column containing the time periods. Default is "t".

`g`
The name of the column containing the first period when a particular observation is treated, with Inf denoting never treated. Default is "g".

`y`
The name of the column containing the outcome variable. Default is "y".

`estimand`
The estimand to be calculated: "simple" averages all treated (t,g) combinations with weights proportional to N_g; "cohort" averages the ATEs for each cohort g, and then takes an N_g-weighted average across g; "calendar" averages ATEs for each time period, weighted by N_g for treated units, and then averages across time. "eventstudy" returns the average effect at the ”event-time” given in the parameter EventTime. The parameter can be left blank if a custom parameter is provided in A_theta_list. The argument is not case-sensitive.

`A_theta_list`
This parameter allows for specifying a custom estimand, and should be left as NULL if estimand is specified. It is a list of matrices A_theta_g so that the parameter of interest is sum_g A_theta_g Ybar_g, where Ybar_g = 1/N sum_i Y_i(g)

`A_0_list`
This parameter allow for specifying the matrices used to construct the Xhat vector of pre-treatment differences. If left NULL, the default is to use the scalar set of controls used in Callaway and Sant’Anna. If use_DiD_A0 = FALSE, then it uses the full vector possible comparisons of (g,g’) in periods t<g,g’.

`eventTime`
If using estimand = "eventstudy", specify what eventTime you want the eventstudy parameter for. The default is 0, the period in which treatment occurs. If a vector is provided, estimates are returned for all the event-times in the vector.

`return_full_vcv`
: If this is true and estimand = "eventstudy", then the function returns a list containing the full variance-covariance matrix for the event-plot estimates in addition to the usual dataframe with the estimates

`compute_fisher`
: If true, computes a Fisher Randomization Test using the studentized estimator.

`num_fisher_permutations`
: The number of permutations to use in the Fisher Randomization Test (if compute_fisher = TRUE). Default is 500.

`skip_data_check`
: If true, skips checks that the data is balanced and contains the colums i,t,g,y. Used in internal recursive calls to increase speed, but not recommended for enduser.

Value

resultsDF A data.frame containing: estimate (the point estimate), se (the standard error), and se_neyman (the Neyman standard error). If a vector-valued eventTime is provided, the data.frame contains multiple rows for each eventTime and an eventTime column. If return_full_vcv = TRUE and estimand = "eventstudy", the function returns a list containing resultsDF and the full variance covariance for the event-study estimates (vcv) as well as the Neyman version of the covariance matrix (vcv_neyman).

References

Callaway, Brantly, and Sant'Anna, Pedro H. C. (2021), 'Difference-in-Differences with Multiple Time Periods', Journal of Econometrics, doi:10.1016/j.jeconom.2020.12.001.

Examples

```
# Load some libraries
set.seed(1234)
# load the officer data and subset it
df <- pj_officer_level_balanced
group_random <- sample(unique(df$assigned), 3)
df <- df[df$assigned %in% group_random,]
# We modify the data so that the time dimension is named t,
# the period of treatment is named g,
# the outcome is named y,
# and the individual identifiers are named i
# (this allow us to use default arguments on \code{staggered_cs}).
oldnames <- c("period", "complaints", "first_trained", "uid")
names(df)[match(oldnames, names(df))] <- c("t", "y", "g", "i")
# Calculate Callaway and Sant'Anna estimator for the simple weighted average
staggered_cs(df = df, estimand = "simple")
# Calculate Callaway and Sant'Anna estimator for the cohort weighted average
staggered_cs(df = df, estimand = "cohort")
# Calculate Callaway and Sant'Anna estimator for the calendar weighted average
staggered_cs(df = df, estimand = "calendar")
# Calculate Callaway and Sant'Anna event-study coefficients for the first 24 months
# (month 0 is instantaneous effect)
eventPlotResults <- staggered_cs(df = df, estimand = "eventstudy", eventTime = 0:23)
head(eventPlotResults)
```

---

staggered_sa *Calculate the Sun & Abraham (2020) estimator for staggered rollouts*

Description

This functions calculates the Sun & Abraham (2020) estimator for staggered rollout designs using last-treated-treated units (never-treated, if availabe) as controls.

Usage

```
staggered_sa(
  df,
  i = "i",
  t = "t",
  g = "g",
  y = "y",
  estimand = NULL,
  A_theta_list = NULL,
  A_0_list = NULL,
  eventTime = 0,
  return_full_vcv = FALSE,
  compute_fisher = FALSE,
  num_fisher_permutations = 500,
  skip_data_check = FALSE
)
```

Arguments

df	A data frame containing panel data with the variables y (an outcome), i (an individual identifier), t (the period in which the outcome is observe), g (the period in which i is first treated, with Inf denoting never treated)
i	The name of column containing the individual (cross-sectional unit) identifier. Default is "i".
t	The name of the column containing the time periods. Default is "t".
g	The name of the column containing the first period when a particular observation is treated, with Inf denoting never treated. Default is "g".
y	The name of the column containing the outcome variable. Default is "y".

`estimand` The estimand to be calculated: "simple" averages all treated (t,g) combinations with weights proportional to N_g; "cohort" averages the ATEs for each cohort g, and then takes an N_g-weighted average across g; "calendar" averages ATEs for each time period, weighted by N_g for treated units, and then averages across time. "eventstudy" returns the average effect at the ”event-time” given in the parameter EventTime. The parameter can be left blank if a custom parameter is provided in A_theta_list. The argument is not case-sensitive.

`A_theta_list` This parameter allows for specifying a custom estimand, and should be left as NULL if estimand is specified. It is a list of matrices A_theta_g so that the parameter of interest is sum_g A_theta_g Ybar_g, where Ybar_g = 1/N sum_i Y_i(g)

`A_0_list` This parameter allow for specifying the matrices used to construct the Xhat vector of pre-treatment differences. If left NULL, the default is to use the scalar set of controls used in Callaway and Sant'Anna. If use_DiD_A0 = FALSE, then it uses the full vector possible comparisons of (g,g') in periods t<g,g'.

`eventTime` If using estimand = "eventstudy", specify what eventTime you want the event-study parameter for. The default is 0, the period in which treatment occurs. If a vector is provided, estimates are returned for all the event-times in the vector.

`return_full_vcv`
If this is true and estimand = "eventstudy", then the function returns a list containing the full variance-covariance matrix for the event-plot estimates in addition to the usual dataframe with the estimates

`compute_fisher` If true, computes a Fisher Randomization Test using the studentized estimator.

`num_fisher_permutations`
The number of permutations to use in the Fisher Randomization Test (if compute_fisher = TRUE). Default is 500.

`skip_data_check`
If true, skips checks that the data is balanced and contains the colums i,t,g,y. Used in internal recursive calls to increase speed, but not recommended for end-user.

Value

resultsDF A data.frame containing: estimate (the point estimate), se (the standard error), and se_neyman (the Neyman standard error). If a vector-valued eventTime is provided, the data.frame contains multiple rows for each eventTime and an eventTime column. If return_full_vcv = TRUE and estimand = "eventstudy", the function returns a list containing resultsDF and the full variance covariance for the event-study estimates (vcv) as well as the Neyman version of the covariance matrix (vcv_neyman).

References

Sun, Liyang, and Abraham, Sarah (2020), 'Estimating dynamic treatment effects in event studies with heterogeneous treatment effects', Forthcoming at the Journal of Econometrics, doi:10.1016/j.jeconom.2020.09.006.

Examples

```
set.seed(1234)
# load the officer data and subset it
df <- pj_officer_level_balanced
group_random <- sample(unique(df$assigned), 3)
df <- df[df$assigned %in% group_random,]
# We modify the data so that the time dimension is named t,
# the period of treatment is named g,
# the outcome is named y,
# and the individual identifiers are named i
#  (this allow us to use default arguments on \code{staggered_cs}).
oldnames <- c("period", "complaints", "first_trained", "uid")
names(df)[match(oldnames, names(df))] <- c("t", "y", "g", "i")
# Calculate Sun and Abraham estimator for the simple weighted average
staggered_sa(df = df, estimand = "simple")
# Calculate Sun and Abraham estimator for the cohort weighted average
staggered_sa(df = df, estimand = "cohort")
# Calculate Sun and Abraham estimator for the calendar weighted average
staggered_sa(df = df, estimand = "calendar")
# Calculate Sun and Abraham event-study coefficients for the first 24 months
# (month 0 is instantaneous effect)
eventPlotResults <- staggered_sa(df = df, estimand = "eventstudy", eventTime = 0:23)
head(eventPlotResults)
```

Index