Package 'eventstudyr'

October 1, 2026

Title Estimation and Visualization of Linear Panel Event Studies

Version 1.2.1

Description Estimates linear panel event study models. Plots coefficients following the recommendations in Freyaldenhoven et al. (2021) <doi:10.3386/w29170>. Includes sup-t bands, testing for key hypotheses, least wiggly path through the Wald region. Allows instrumental variables estimation following Freyaldenhoven et al. (2019) <doi:10.1257/aer.20180609>.

License MIT + file LICENSE

Encoding UTF-8

Imports car, data.table, dplyr, estimatr, fixest, ggplot2, MASS, rlang, pracma, stats, stringr

VignetteBuilder knitr

LazyData true

Suggests rmarkdown, knitr, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 3.5.0)

URL https://github.com/JMSLab/eventstudyr

BugReports https://github.com/JMSLab/eventstudyr/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Simon Freyaldenhoven [aut],
Christian Hansen [aut],
Jorge Pérez Pérez [aut],
Jesse Shapiro [aut],
Veli Andirin [aut],
Richard Calvo [aut],
Santiago Hermo [aut, cre],
Nathan Schor [aut],
Emily Wang [aut],
JMSLab [cph],
Ryan Kessler [cph]

Maintainer Santiago Hermo <santiago.hermo@monash.edu>

Repository CRAN

Date/Publication 2026-10-01 08:20:09 UTC

Contents

EventStudy 2
EventStudyPlot 6
example_data 8
TestLinear 9

Index **11**

`EventStudy` *Estimates Equation (2) in Freyaldenhoven et al. (2021)*

Description

`EventStudy` uses regression methods to estimate the effect of a policy on a given outcome.

Usage

```
EventStudy(
  estimator,
  data,
  outcomevar,
  policyvar,
  idvar,
  timevar,
  controls = NULL,
  proxy = NULL,
  proxyIV = NULL,
  FE = TRUE,
  TFE = TRUE,
  post,
  overidpost = 1,
  pre,
  overidpre = post + pre,
  normalize = -1 * (pre + 1),
  cluster = TRUE,
  anticipation_effects_normalization = TRUE,
  allow_duplicate_id = FALSE,
  avoid_internal_copy = FALSE,
  kernel = "estimatr"
)
```

Arguments

`estimator`	Accepts one of "OLS" or "FHS". If "OLS" is specified, implements Ordinary Least Squares. If "FHS" is specified, implements Instrumental Variables (IV) estimator proposed in Freyaldenhoven Hansen Shapiro (FHS, 2019).
`data`	Data frame containing the variables of interest.
`outcomevar`	Character indicating column of outcome variable y.
`policyvar`	Character indicating column of policy variable z.
`idvar`	Character indicating column of units.
`timevar`	Character indicating column of time periods.
`controls`	Optional character vector indicating a set of control variables q.
`proxy`	Character indicating column of variable that is thought to be affected by the confound but not by the policy. Should be specified if and only if estimator is specified as "FHS".
`proxyIV`	Character of column to be used as an instrument. Should be specified if and only if estimator is specified as "FHS". If NULL, defaults to the strongest lead of the policy variable based on the first stage.
`FE`	Logical indicating whether unit fixed-effects should be included. Defaults to TRUE.
`TFE`	Logical indicating whether time fixed-effects should be included. Defaults to TRUE.
`post`	Whole number indicating the number of periods in the past before which the past values of the policy are not supposed to affect the value of the outcome. Corresponds to M in equation (2) of Freyaldenhoven et al. (2021).
`overidpost`	Optional whole number indicating the number of event times after "post" to be included in estimation. Defaults to 1. Corresponds to L_M in equation (2) of Freyaldenhoven et al. (2021).
`pre`	Whole number indicating the number of periods in the future after which the future values of the policy are not supposed to affect the value of the outcome today. Corresponds to G in equation (2) of Freyaldenhoven et al. (2021).
`overidpre`	Optional whole number indicating the number of event times earlier than -"pre" to be included in estimation. Defaults to "post" + "pre". Corresponds to L_G in equation (2) of Freyaldenhoven et al. (2021).
`normalize`	Specifies the event-time coefficient to be normalized. Defaults to - pre - 1.
`cluster`	Logical indicating whether to use clustered errors by units. If FALSE, will use unclustered heteroskedasticity-robust standard errors. Defaults to TRUE. Must be TRUE if FE is TRUE.
`anticipation_effects_normalization`	If set to TRUE, runs the default process and switches coefficient to be normalized to 0 when there are anticipation effects. If set to FALSE, does not make the switch. Defaults to TRUE.
`allow_duplicate_id`	If TRUE, the function estimates a regression where duplicated ID-time rows are weighted by their duplication count. If FALSE, the function raises an error if duplicate unit-time keys exist in the input data. Default is FALSE.

avoid_internal_copy
: If TRUE, the function avoids making an internal deep copy of the input data, and instead directly modifies the input data.table. Default is FALSE.

kernel
: Accepts one of "estimatr" or "fixest". If "estimatr" is specified, uses the estimatr package for estimation. If "fixest" is specified, uses the fixest package for estimation. Defaults to "estimatr" (deprecated - will change to "fixest" in a future release).

Value

A list that contains, under "output", the estimation output as an lm_robust object, and under "arguments", the arguments passed to the function.

Examples

```
# A minimal example
eventstudy_model <-
  EventStudy(
    estimator = "OLS",
    data = example_data,
    outcomevar = "y_base",
    policyvar = "z",
    idvar = "id",
    timevar = "t",
    pre = 0, post = 3,
    normalize = -1,
    kernel = "fixest"
  )

### Access estimated model
eventstudy_model$output

summary(eventstudy_model$output)

### data.frame of estimates
fixest::coeftable(eventstudy_model$output) # for kernel='fixest'
# estimatr::tidy(eventstudy_model$output) # for kernel='estimatr'

### Access arguments
eventstudy_model$arguments

# A dynamic OLS model with anticipation effects and controls
eventstudy_model_dyn <-
  EventStudy(
    estimator = "OLS",
    data = example_data,
    outcomevar = "y_base",
    policyvar = "z",
    idvar = "id",
    timevar = "t",
    controls = "x_r",
    FE = TRUE, TFE = TRUE,
```

```
    post = 3, overidpost = 5,
    pre  = 2, overidpre  = 4,
    normalize = - 3,
    cluster = TRUE,
    kernel = "fixest",
    anticipation_effects_normalization = TRUE
  )

summary(eventstudy_model_dyn$output)

# A static model
eventstudy_model_static <-
  EventStudy(
    estimator = "OLS",
    data = example_data,
    outcomevar = "y_jump_m",
    policyvar = "z",
    idvar = "id",
    timevar = "t",
    FE = TRUE, TFE = TRUE,
    post = 0, overidpost = 0,
    pre  = 0, overidpre  = 0,
    cluster = TRUE,
    kernel = "fixest"
  )

summary(eventstudy_model_static$output)

# A dynamic model with an unbalanced panel
data_unbal <- example_data[1:(nrow(example_data)-1),]  # drop last row to make unbalanced

eventstudy_model_unbal <-
 EventStudy(
    estimator = "OLS",
    data = data_unbal,
    outcomevar = "y_base",
    policyvar = "z",
    idvar = "id",
    timevar = "t",
    pre = 0, post = 3,
    normalize = -1,
    kernel = "fixest"
  )

summary(eventstudy_model_unbal$output)

# A dynamic model estimated using IV
eventstudy_model_iv <-
  EventStudy(
    estimator = "FHS",
    data = example_data,
    outcomevar = "y_base",
    policyvar = "z",
```

```
    idvar = "id",
    timevar = "t",
    proxy = "x_r",
    FE = TRUE, TFE = TRUE,
    post = 2, overidpost = 1,
    pre  = 0, overidpre  = 3,
    normalize = -1,
    cluster = TRUE,
    kernel = "fixest"
  )

summary(eventstudy_model_iv$output)
```

`EventStudyPlot` *Creates an Event-Study Plot Following the Suggestions in Freyaldenhoven et al. (2021)*

Description

`EventStudyPlot` takes the output from `EventStudy()` and combines it with additional optional arguments to facilitate constructing an Event-Study Plot.

Usage

```
EventStudyPlot(
  estimates,
  xtitle = "Event time",
  ytitle = "Coefficient",
  ybreaks = NULL,
  conf_level = 0.95,
  supt = 0.95,
  num_sim = 1000,
  add_mean = FALSE,
  pre_event_coeffs = TRUE,
  post_event_coeffs = TRUE,
  add_zero_line = TRUE,
  smpath = FALSE
)
```

Arguments

`estimates`	The output from calling `EventStudy()`. Should be a list of length 2.
`xtitle`	The title for the x-axis. Should be a string. Defaults to "Event time".
`ytitle`	The title for the y-axis. Should be a string. Defaults to "Coefficient".
`ybreaks`	A vector containing the desired breaks for the y-axis. Defaults to NULL, which means the breaks are computed automatically. If custom breaks are selected with the `add_mean` argument set to TRUE, then the breaks must include zero.

`conf_level` Confidence level used for confidence interval expressed as a real number between 0 and 1, inclusive. Defaults to 0.95.

`supt` The confidence level used for obtaining the sup-t bands critical value. Should be a real number between 0 and 1, inclusive. Defaults to .95. Sup-t bands are simulation-based, so you must set a seed if you would like your sup-t band results to be reproducible (see examples).

`num_sim` The number of simulations used in generating the sup-t bands. Should be a natural number. Defaults to 1000.

`add_mean` Adds the mean of the dependent variable in the period used for normalization. Should be TRUE or FALSE. Defaults to FALSE.

`pre_event_coeffs` If TRUE, uses pre and overidpre from estimates to test for pre-trends. Should be TRUE or FALSE. Defaults to TRUE.

`post_event_coeffs` If TRUE, uses post and overidpost from estimates to test for leveling-off. Should be TRUE or FALSE. Defaults to TRUE.

`add_zero_line` Whether or not to plot a dashed horizontal line at y = 0. Should be TRUE or FALSE. Defaults to TRUE, meaning the line is plotted.

`smpath` Plot smoothest path of confounder that rationalizes event study coefficients. Should be TRUE or FALSE. Defaults to FALSE.

Value

The Event-Study plot as a ggplot2 object.

Examples

```
#

# Minimal examples
### OLS

estimates_ols <- EventStudy(
   estimator = "OLS",
   data = example_data,
   outcomevar = "y_smooth_m",
   policyvar = "z",
   idvar = "id",
   timevar = "t",
   controls = "x_r",
   FE = TRUE, TFE = TRUE,
   post = 3, overidpost = 5,
   pre = 2,  overidpre = 4,
   normalize = - 3
)

plt_ols <- EventStudyPlot(estimates = estimates_ols)
plt_ols
```

```
### IV

estimates_fhs <- EventStudy(
   estimator = "FHS",
   data = example_data,
   outcomevar = "y_smooth_m",
   policyvar = "z",
   idvar = "id",
   timevar = "t",
   proxy = "x_r",
   post = 2, overidpost = 1,
   pre = 0,  overidpre = 3,
   normalize = -1
)

plt_fhs <- EventStudyPlot(estimates = estimates_fhs)
plt_fhs

# Optional arguments

### Change x- and y-axis titles and set ybreaks
EventStudyPlot(estimates = estimates_ols,
               xtitle = "Relative time", ytitle = "",
               ybreaks = seq(-2, 1, 0.5))

### Add smoothest path
EventStudyPlot(estimates = estimates_ols, smpath = TRUE)

### Add y-mean to y-axis and line y = 0
EventStudyPlot(estimates = estimates_ols, add_mean = TRUE,
               add_zero_line = TRUE)

### Do not plot supt bands
EventStudyPlot(estimates = estimates_ols, supt = NULL)

### Setting seed prior to plotting sup-t bands
set.seed(1234)
EventStudyPlot(estimates = estimates_ols)

# Modify plots using ggplot2 functions
library(ggplot2)

### Change color of dots, horizontal line, and theme
plt_ols +
  geom_point(color = "red") +
  geom_hline(color = "gray", yintercept = 0) +
  theme_light() +
  theme(panel.grid.minor.x = element_blank())
```

example_data — *Sample dataset obtained from the replication archive for Freyaldenhoven et al. (2021)*

Description

Sample dataset obtained from the replication archive for Freyaldenhoven et al. (2021)

Usage

```
example_data
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 2000 rows and 12 columns.

Source

Dataset in .dta format can be found in the .zip archive in https://data.nber.org/data-appendix/w29170/

TestLinear — *Perform Tests of Linear Hypotheses*

Description

`TestLinear` tests linear restrictions on coefficients.

Usage

```
TestLinear(
  estimates,
  test = NA,
  test_name = "User Test",
  pretrends = TRUE,
  leveling_off = TRUE
)
```

Arguments

`estimates`	A list of length 2 containing estimation results and model information. Should be an output of `EventStudy()`.
`test`	The hypothesis to be estimated. Accepts inputs that can be passed to hypothesis.matrix argument in `car::linearHypothesis()`.
`test_name`	Name for test input by user. Defaults to "User Test."
`pretrends`	If TRUE, uses pre and overidpre from estimates to test for pre-trends. Defaults to TRUE.

`leveling_off` If TRUE, uses post and overidpost from estimates to test for leveling-off. Defaults to TRUE.

Value

A data frame containing the F-statistic and p-value for the specified test(s).

Examples

```
estimates <- EventStudy(estimator = "OLS", data = example_data, outcomevar = "y_base",
                        policyvar = "z", idvar = "id", timevar = "t",
                        controls = "x_r", FE = TRUE, TFE = TRUE,
                        post = 3, pre = 2, overidpre = 4, overidpost = 5,
                  normalize = -3, cluster = TRUE, anticipation_effects_normalization = TRUE)

TestLinear(
    estimates,
    test = "z_fd_lag1 = z_fd",
    test_name = "Hypothesis Test",
    pretrends = TRUE,
    leveling_off = TRUE
   )
```

Index

∗ **datasets**
example_data, 9

EventStudy, 2
EventStudy(), *6*, *9*
EventStudyPlot, 6
example_data, 8

TestLinear, 9