Package: fixes (via r-universe)

October 20, 2026

Type Package

Title Staggered DiD Tools: Event Studies and ATT Aggregation

Version 0.12.0

Date 2026-07-05

Description Provides tools for difference-in-differences (DiD) estimation and visualization with staggered adoption. Includes run_es() for event-study curves (dynamic effects by relative time) and calc_att() for aggregated ATT estimation (overall, by cohort, by calendar time). Supports multiple modern estimators: Callaway-Sant'Anna (2021), Sun-Abraham (2021), Borusyak-Jaravel-Spiess (2024), Wooldridge TWM, and Deb et al. FLEX.

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, fixest, broom, tibble, rlang, Rcpp, scales, parallel

License MIT + file LICENSE

Encoding UTF-8

Suggests knitr, rmarkdown, haven, testthat, plotly, tidyr, did, didimputation, modelsummary, tinytable, lpSolveAPI, Rglpk, TruncatedNormal, Matrix, pracma, HonestDiD

VignetteBuilder knitr

URL https://github.com/yo5uke/fixes, https://yo5uke.com/fixes/

BugReports https://github.com/yo5uke/fixes/issues

Config/roxygen2/version 8.0.0

Config/roxygen2/markdown TRUE

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Yosuke Abe [aut, cre]

Maintainer Yosuke Abe <yosuke.abe0507@gmail.com>

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

Date/Publication 2026-07-04 16:10:02 UTC

RemoteUrl https://github.com/cran/fixes

RemoteRef HEAD

RemoteSha cca8a41479d29b223149ffd519c431f93a00c9ea

Contents

autoplot.es_result . . . 2
calc_att . . . 3
compute_contamination_weights . . . 5
glance.att_result . . . 7
glance.did_result . . . 8
honest_sensitivity . . . 8
plot_att_gt . . . 10
plot_contamination_weights . . . 11
plot_es . . . 12
plot_es_interactive . . . 14
plot_honest . . . 15
run_did . . . 16
run_es . . . 18
tidy.att_result . . . 22
tidy.did_result . . . 23

Index **24**

`autoplot.es_result` *Autoplot for event-study results*

Description

S3 method that plots an `es_result` (from `run_es()`). It forwards arguments to `plot_es()`.

Usage

```
## S3 method for class 'es_result'
autoplot(object, ci_level = 0.95, type = "ribbon", ...)
```

Arguments

`object`	An `es_result` returned by `run_es()`.
`ci_level`	Confidence level (numeric, e.g., 0.95). Passed to `plot_es()`.
`type`	Plot type: `"ribbon"` (default) or `"errorbar"`. Passed to `plot_es()`.
`...`	Additional arguments forwarded to `plot_es()`.

Value

A ggplot object.

Examples

```
# res <- run_es(...)
# ggplot2::autoplot(res, ci_level = 0.95, type = "ribbon")
```

`calc_att` *Compute ATT Aggregations for Staggered Adoption Designs*

Description

Estimates average treatment effects on the treated (ATT) using either the Callaway-Sant'Anna (2021) or Borusyak-Jaravel-Spiess (2024) estimator, aggregated to a single summary effect (`"simple"`), per-cohort effects (`"by_cohort"`), or per calendar-time effects (`"by_time"`).

Usage

```
calc_att(
  data,
  outcome,
  treatment = NULL,
  time,
  timing,
  fe = NULL,
  covariates = NULL,
  cluster = NULL,
  weights = NULL,
  interval = 1,
  time_transform = FALSE,
  unit = NULL,
  estimator = c("cs", "bjs"),
  aggregation = c("simple", "by_cohort", "by_time"),
  control_group = c("nevertreated", "notyettreated"),
  anticipation = 0L,
  conf.level = 0.95,
  vcov = "HC1",
  vcov_args = list()
)
```

Arguments

`data`	A data.frame containing panel data.
`outcome`	Unquoted outcome variable (name or expression, e.g., `log(y)`).
`treatment`	Unused; reserved for future use.

time	Unquoted calendar time variable (numeric).
timing	Unquoted column giving each unit's first treatment period (NA = never treated).
fe	Ignored (CS and BJS absorb fixed effects internally).
covariates	Ignored; reserved for future use.
cluster	Ignored; reserved for future use.
weights	Ignored; reserved for future use.
interval	Numeric time spacing (default 1; informational only).
time_transform	Logical; if TRUE, creates consecutive integer time within unit via dplyr::dense_rank(). Requires unit.
unit	Unquoted unit identifier (required).
estimator	Estimation strategy: "cs" (Callaway-Sant'Anna 2021, default) or "bjs" (Borusyak-Jaravel-Spiess 2024).
aggregation	Aggregation type: "simple" (overall ATT, default), "by_cohort" (one ATT per treatment cohort), or "by_time" (one ATT per calendar time period).
control_group	For estimator = "cs": comparison group, "nevertreated" (default) or "notyettreated".
anticipation	For estimator = "cs": number of anticipation periods before treatment (non-negative integer, default 0L).
conf.level	Numeric confidence level(s) (default 0.95). Multiple levels are supported, e.g., c(0.90, 0.95).
vcov	Ignored (SE is analytical for CS; approximate for BJS).
vcov_args	Ignored.

Details

This function complements run_es(): use run_es() when you want a full event-study curve (dynamic effects by relative time), and calc_att() when you want aggregated ATT estimates that collapse the time dimension.

The result has broom::tidy() and broom::glance() methods, so it can be passed to modelsummary::modelsummary() (rendered with tinytable) to build a publication table — including stacking several estimators (e.g. run_did() TWFE, CS, and BJS) side by side in one table.

Value

A data.frame of class "att_result" with columns:

group Cohort or calendar time (NA for "simple").

estimate ATT point estimate.

std.error Standard error.

statistic t-statistic (estimate / std.error).

p.value Two-sided p-value (normal approximation).

conf_low_XX, conf_high_XX CI bounds for each conf.level.

Attributes: aggregation, estimator, conf.level, N, N_units, N_treated, N_nevertreated, control_group (CS only), att_gt (CS raw ATT(g,t) table), tau_it (BJS unit-time effects table).

Aggregation formulas (CS estimator)

- **simple**: $\theta = \sum_g (n_g / n_{treated}) \cdot \overline{ATT(g,\cdot)}$ where $\overline{ATT(g,\cdot)}$ is the mean over post-treatment periods.
- **by_cohort**: $\theta(g) = \overline{ATT(g,\cdot)}$ per cohort.
- **by_time**: $\theta(t) = \sum_{g \le t} w(g,t) \cdot ATT(g,t)$ with $w(g,t) = n_g / \sum_{g' \le t} n_{g'}$.

Standard errors (BJS estimator)

BJS SEs are approximate (naive sample variance of unit-time effects). Cluster-robust SEs for BJS aggregations are planned for a future release.

See Also

`run_es()` for event-study (dynamic) estimates.

---

compute_contamination_weights

Contamination Weights for TWFE Event-Study Coefficients (Sun-Abraham 2021)

Description

Estimates the contamination weights $\omega_{e,\ell}^{\ell''}$ that decompose each TWFE event-study coefficient $\hat{\mu}_{\ell''}$ into a linear combination of cohort-specific ATTs (CATTs):

$$\hat{\mu}_{\ell''} = \sum_{(e,\ell)} \omega_{e,\ell}^{\ell''} \cdot \widehat{CATT}_{e,\ell}$$

(Sun and Abraham 2021, Equation 20).

The weights are obtained via the OVB auxiliary regression (Eq. 12): for each cohort-period CATT cell (e,ℓ), regress the cohort-specific indicator $1\{E_i = e\} \cdot 1\{t - E_i = \ell\}$ on all cohort-aggregated relative-time indicators $D_{i,t}^{\ell''}$ and two-way fixed effects. The resulting regression coefficient on $D_{i,t}^{\ell''}$ is $\omega_{e,\ell}^{\ell''}$.

Usage

```
compute_contamination_weights(
  data,
  time,
  timing,
  unit,
  fe = NULL,
  baseline = -1L
)
```

Arguments

`data`	A data.frame with one row per unit-period (balanced panel).
`time`	Unquoted name of the calendar time variable (numeric).
`timing`	Unquoted name of the first-treatment-period variable; `NA` marks never-treated units.
`unit`	Unquoted name of the unit identifier.
`fe`	One-sided fixed-effects formula, e.g. `~ id + year`. When `NULL` (default), falls back to `~ <unit> + <time>` with a warning.
`baseline`	Integer reference (baseline) period excluded from the TWFE specification (default `-1L`). Must match the `baseline` argument used in `run_es()`.

Value

An object of class `c("sa_contamination_weights", "data.frame")` with one row per `(catt_cohort, catt_period, tw` triple, and columns:

`catt_cohort` Cohort e (first treatment period).

`catt_period` Relative event time ℓ of the CATT.

`twfe_period` Relative event time ℓ'' of the TWFE coefficient being decomposed.

`weight` Contamination weight $\omega_{e,\ell}^{\ell''}$.

`is_own` Logical; `TRUE` when `catt_period == twfe_period`.

Attributes: `baseline`, `cohorts`, `cohort_sizes`, `incl_periods`.

Interpretation

- **Own-period cell** (`catt_period == twfe_period`): $\omega_{e,\ell}^{\ell}$ represents the weight the TWFE estimator places on $CATT_{e,\ell}$. Under the SA IW estimator these equal the cohort-size weights $n_e / \sum n_{e'}$.
- **Cross-period cell** (`catt_period != twfe_period`): Any non-zero weight indicates contamination: the TWFE coefficient $\hat{\mu}_{\ell''}$ also picks up treatment effects from period $\ell \neq \ell''$.
- **Verification**: the OVB identity (property iii) holds exactly, so $\hat{\mu}_{\ell''} = \sum_{(e,\ell)} \omega_{e,\ell}^{\ell''} \cdot \widehat{CATT}_{e,\ell}$ up to floating-point precision.

References

Sun, L. and Abraham, S. (2021). Estimating dynamic treatment effects in event studies with heterogeneous treatment effects. *Journal of Econometrics*, 225(2), 175–199.

See Also

`plot_contamination_weights()`, `run_es()`

Examples

```
## Not run:
# Estimate contamination weights
cw <- compute_contamination_weights(
  data     = panel_data,
  time     = year,
  timing   = first_treat,
  unit     = id,
  fe       = ~ id + year,
  baseline = -1L
)
print(cw)
plot_contamination_weights(cw)

## End(Not run)
```

`glance.att_result` *Glance at an* `att_result` *object*

Description

Returns a one-row summary of an `calc_att()` result for the goodness-of-fit block of a `modelsummary::modelsummary()` table.

Usage

```
glance.att_result(x, ...)
```

Arguments

`x`	An `att_result` object returned by `calc_att()`.
`...`	Unused; for S3 generic compatibility.

Value

A one-row data frame with `nobs`, `n.units`, `n.treated`, `estimator`, and `aggregation`.

Examples

```
## Not run:
att <- calc_att(df, outcome = y, time = year, timing = g, unit = id)
broom::glance(att)

## End(Not run)
```

glance.did_result *Glance at a* did_result *object*

Description

Returns a single-row summary of model-level statistics from a run_did() result. Delegates to broom::glance.fixest() which provides nobs, r.squared, adj.r.squared, within.r.squared, AIC, BIC, and related statistics.

Usage

```
glance.did_result(x, ...)
```

Arguments

x	A did_result object returned by run_did().
...	Additional arguments passed to broom::glance.fixest().

Value

A one-row data frame of model-level statistics.

Examples

```
## Not run:
res <- run_did(df, outcome = y, treatment = D, fe = ~ id + year)
broom::glance(res)

## End(Not run)
```

honest_sensitivity *Honest sensitivity analysis for parallel-trends violations*

Description

Robust inference and sensitivity analysis for event-study / DiD designs following Rambachan and Roth (2023). Instead of assuming parallel trends holds exactly, it asks how large a violation of parallel trends would have to be before the causal conclusion changes, returning confidence sets for a post-treatment effect under a sequence of restrictions on the possible difference in trends.

Two restriction families are supported:

- "relative_magnitude" ($\Delta^{RM}(\bar{M})$): post-treatment violations are at most $\bar{M}$ times the largest pre-treatment violation (Rambachan & Roth 2023, Section 2.4.1).
- "smoothness" ($\Delta^{SD}(M)$): the difference in trends deviates from linearity by at most M per period (Section 2.4.3).

Inference uses the Andrews-Roth-Pakes (ARP) conditional moment-inequality test (Section 3.2.1), which is uniformly valid and recommended for general restriction sets.

Usage

```
honest_sensitivity(
  object = NULL,
  type = c("relative_magnitude", "smoothness"),
  Mvec = NULL,
  l_vec = NULL,
  alpha = 0.05,
  gridPoints = 1000L,
  betahat = NULL,
  sigma = NULL,
  numPrePeriods = NULL,
  numPostPeriods = NULL
)
```

Arguments

`object`	An `es_result` from `run_es()` (with `estimator = "twfe"`, i.e. classic or `method = "sunab"`, which carry the event-study coefficient covariance). Alternatively, supply `betahat` and `sigma` directly (see below) for any estimator.
`type`	Restriction family: `"relative_magnitude"` (default) or `"smoothness"`.
`Mvec`	Numeric vector of restriction parameters. For `"relative_magnitude"` these are $\bar{M}$ values (default `seq(0, 2, by = 0.5)`); for `"smoothness"` these are M values (default a data-driven sequence from 0 to the largest pre-period SD).
`l_vec`	Numeric weight vector over post-treatment periods defining the target $\theta = l'\tau_{post}$. Defaults to the first post-treatment period.
`alpha`	Significance level (default `0.05` for 95% confidence sets).
`gridPoints`	Number of grid points for test inversion (default `1000`).
`betahat`	Optional event-study coefficient vector (pre then post, excluding the reference period), ordered by relative time. Required when `object` does not carry an `es_vcov` attribute.
`sigma`	Optional covariance matrix of `betahat`.
`numPrePeriods, numPostPeriods`	Optional integer counts; inferred from `object` or from `betahat`/`l_vec` when omitted.

Value

A `data.frame` of class `"honest_result"` with one row per restriction value plus the original (parallel-trends) confidence interval, with columns `M`, `lb`, `ub`, `method`, and `type`. The breakdown value (largest restriction at which the robust CI still excludes 0) is stored in `attr(., "breakdown")`.

References

Rambachan, A. and Roth, J. (2023). A More Credible Approach to Parallel Trends. *Review of Economic Studies*, 90(5), 2555-2591.

See Also

run_es(), plot_honest()

---

plot_att_gt *Plot the ATT(g,t) matrix from a Callaway-Sant'Anna event study*

Description

Visualises the full cohort-by-period ATT(g,t) matrix stored in the att_gt attribute of an es_result object produced by run_es(estimator = "cs"). Two display styles are available:

- "heatmap": a tile plot with calendar time t on the x-axis and cohort g on the y-axis, colour-filled by the point estimate. Cells whose pointwise confidence interval excludes zero are marked with a filled dot; cells that are simultaneously significant (when bootstrap data are available) are additionally marked with an open diamond.
- "facet": one panel per cohort showing ATT(g,t) over calendar time t with a pointwise confidence ribbon, mirroring the style of plot_es(). A lighter simultaneous CI ribbon is overlaid when bootstrap data are available.

Both types draw a vertical dashed line at $t = g$ (treatment onset) for each cohort.

Usage

```
plot_att_gt(
  x,
  type = c("heatmap", "facet"),
  ci_level = 0.95,
  zero_line = TRUE,
  theme = c("bw", "minimal", "classic"),
  color = "#B25D91FF",
  fill = "#B25D91FF",
  alpha = 0.2
)

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

Arguments

x	An es_result object returned by run_es(estimator = "cs"), or an att_gt_result data frame produced by extracting attr(x, "att_gt") and giving it class "att_gt_result".
type	"heatmap" (default) or "facet".
ci_level	Confidence level for pointwise intervals (default 0.95).
zero_line	Logical; draw a horizontal reference line at zero in the "facet" display (default TRUE).
theme	One of "bw" (default), "minimal", or "classic".

color	Line and point colour used in the "facet" display (default "#B25D91FF", matching plot_es()).
fill	Ribbon fill colour in the "facet" display (default "#B25D91FF").
alpha	Ribbon transparency in the "facet" display (default 0.2).
object	An att_gt_result object (extracted from an es_result via attr(result, "att_gt")).
...	Passed to plot_att_gt.

Value

A ggplot2::ggplot() object.

Bootstrap annotation

When attr(x, "bootstrap") is present (i.e., run_es() was called with bootstrap = TRUE), both plot types add simultaneous inference overlays sourced from the (g,t)-level bootstrap object.

See Also

plot_es(), run_es()

Examples

```
## Not run:
cs_result <- run_es(data = mydata, outcome = y, time = year,
                    timing = g, unit = id, fe = ~id + year,
                    staggered = TRUE, estimator = "cs")
plot_att_gt(cs_result)
plot_att_gt(cs_result, type = "facet")

## End(Not run)
```

---

plot_contamination_weights

Plot Contamination Weights as a Tile Heatmap

Description

Creates a ggplot2 tile heatmap of the contamination weights returned by compute_contamination_weights().

Each cell at position (twfe_period, catt_label) shows the weight $\omega_{e,\ell}^{\ell''}$: how much of $CATT_{e,\ell}$ leaks into the TWFE coefficient $\hat{\mu}_{\ell''}$.

- **Diagonal cells** (catt_period == twfe_period): own-period weights (sum across cohorts should be ≈ 1).
- **Off-diagonal cells**: cross-period contamination (ideally close to zero under treatment effect homogeneity).

Usage

```
plot_contamination_weights(
  x,
  limit_abs = NULL,
  midpoint = 0,
  low = "#2166AC",
  mid = "white",
  high = "#B2182B",
  theme = c("bw", "minimal", "classic"),
  show_values = FALSE,
  value_digits = 2L
)
```

Arguments

x	An sa_contamination_weights object from compute_contamination_weights().
limit_abs	Numeric; symmetric colour scale limit [-limit, limit]. Defaults to the maximum absolute weight (rounded up to one decimal).
midpoint	Numeric; midpoint of the diverging colour scale (default 0).
low	Colour for negative weights (default "#2166AC").
mid	Colour for zero weight (default "white").
high	Colour for positive weights (default "#B2182B").
theme	Character; "bw" (default), "minimal", or "classic".
show_values	Logical; overlay weight values in each tile (default FALSE).
value_digits	Integer; decimal digits when show_values = TRUE (default 2L).

Value

A ggplot2::ggplot() object.

See Also

compute_contamination_weights()

---

plot_es *Plot event-study results with ribbons or error bars*

Description

Plot event-study results with ribbons or error bars

Usage

```
plot_es(
  data,
  ci_level = 0.95,
  type = "ribbon",
  vline_val = 0,
  vline_color = "#000",
  hline_val = 0,
  hline_color = "#000",
  linewidth = 1,
  pointsize = 2,
  alpha = 0.2,
  barwidth = 0.2,
  color = "#B25D91FF",
  fill = "#B25D91FF",
  theme_style = "bw",
  show_simultaneous = FALSE
)
```

Arguments

`data`	An object of class `es_result` returned by `run_es()`.
`ci_level`	Confidence level to display (e.g., 0.95).
`type`	One of `"ribbon"` (default) or `"errorbar"`.
`vline_val`, `hline_val`	Numeric locations for vertical/horizontal reference lines (default 0).
`vline_color`, `hline_color`	Colors for reference lines.
`linewidth`, `pointsize`, `alpha`, `barwidth`	Styling parameters for lines/points/bands/bars.
`color`, `fill`	Optional, override line/point color and ribbon fill.
`theme_style`	One of `"bw"`, `"minimal"`, or `"classic"` for ggplot theme.
`show_simultaneous`	Logical; if `TRUE`, overlays the simultaneous bootstrap CI (lighter band, alpha 0.15) alongside the standard pointwise CI (alpha 0.3), with a legend distinguishing the two bands. Requires `bootstrap = TRUE` in the originating `run_es()` call. Default `FALSE`.

Value

A `ggplot` object.

Examples

```
# Assuming `res <- run_es(...)`
# p <- plot_es(res, ci_level = 0.95, type = "ribbon")
# print(p)
```

plot_es_interactive *Interactive event-study plot with hover details*

Description

Creates an interactive plotly visualization of event study results with hover-over displays showing coefficients, confidence intervals, and other details.

Usage

```
plot_es_interactive(
  data,
  ci_level = 0.95,
  vline_val = 0,
  hline_val = 0,
  vline_color = "#000",
  hline_color = "#000",
  color = "#B25D91FF",
  fill = "#B25D91FF",
  alpha = 0.2,
  linewidth = 2,
  markersize = 8,
  show_ribbon = TRUE,
  show_simultaneous = FALSE,
  height = NULL,
  width = NULL
)
```

Arguments

data	An object of class es_result returned by run_es().
ci_level	Confidence level to display (e.g., 0.95). Default is 0.95.
vline_val	Numeric location for vertical reference line (default 0).
hline_val	Numeric location for horizontal reference line (default 0).
vline_color	Color for vertical reference line (default "#000").
hline_color	Color for horizontal reference line (default "#000").
color	Point and line color (default "#B25D91FF").
fill	Ribbon/band fill color (default "#B25D91FF").
alpha	Ribbon transparency (default 0.2).
linewidth	Line width (default 2).
markersize	Marker size (default 8).
show_ribbon	Logical; if TRUE, shows confidence interval as a ribbon band (default TRUE).

show_simultaneous
: Logical; if TRUE, overlays a second (lighter) ribbon for the simultaneous bootstrap CI and extends the hover tooltip with simultaneous CI bounds. Requires bootstrap = TRUE in the originating run_es() call. Default FALSE.

height
: Plot height in pixels (default NULL for auto).

width
: Plot width in pixels (default NULL for auto).

Details

The hover tooltip displays:

- Relative time to treatment
- Point estimate (coefficient)
- Confidence interval bounds
- Standard error
- P-value
- Simultaneous CI bounds (when show_simultaneous = TRUE)

Value

A plotly object that can be displayed interactively.

Examples

```
## Not run:
# Assuming res <- run_es(...)
plot_es_interactive(res)
plot_es_interactive(res, ci_level = 0.99, show_ribbon = FALSE)
plot_es_interactive(res, show_simultaneous = TRUE)

## End(Not run)
```

---

plot_honest *Plot a honest sensitivity analysis*

Description

Visualises the output of honest_sensitivity(): robust confidence intervals for the target effect under progressively weaker parallel-trends restrictions (increasing M or $\bar{M}$), alongside the original confidence interval that assumes parallel trends holds exactly. This is the "top-down" sensitivity plot of Rambachan and Roth (2023).

Usage

```
plot_honest(x, ...)

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

Arguments

x	A honest_result object from honest_sensitivity().
...	Unused.
object	A honest_result object.

Value

A ggplot object.

See Also

honest_sensitivity()

---

run_did *Run a basic two-way fixed-effects DiD model*

Description

Estimates a classic difference-in-differences model of the form outcome ~ D_it | fe using fixest::feols().

There are two ways to supply the treatment indicator:

Option A — pre-built D_it (maximum flexibility):

```
df$D <- as.integer(df$treated & df$year >= 2006)
run_did(df, outcome = y, treatment = D, fe = ~ id + year)
```

Option B — timing-based construction (convenience; consistent with run_es() and calc_att()):

```
run_did(df, outcome = y, treatment = treated, time = year, timing = 2006,
        fe = ~ id + year)
```

Here treatment is a binary group indicator (1 = treated unit, 0 = control), time is the calendar-time variable, and timing is the scalar treatment onset period. Internally D_it = treatment * (time >= timing) is constructed automatically. For staggered-adoption settings use calc_att(); for dynamic event-study estimates use run_es().

Usage

```
run_did(
  data,
  outcome,
  treatment,
  timing = NULL,
  fe = NULL,
  unit = NULL,
  time = NULL,
```

```
  covariates = NULL,
  cluster = NULL,
  weights = NULL,
  conf.level = 0.95,
  vcov = "HC1",
  vcov_args = list()
)
```

Arguments

data	A data.frame (panel format).
outcome	Unquoted outcome variable or expression (e.g., log(y)).
treatment	Unquoted column name. When timing = NULL (default): a pre-built binary D_it indicator (1 = treated unit-time, 0 = otherwise). When timing is provided: a binary group indicator (1 = treated unit, 0 = control unit; constant within units).
timing	Numeric scalar. When provided, D_it is constructed as treatment * (time >= timing). Requires time to be specified. Default NULL (user supplies pre-built D_it via treatment).
fe	One-sided formula specifying fixed effects, e.g. ~ id + year. If NULL and both unit and time are supplied, fe is auto-inferred as ~ unit + time. If NULL and neither is supplied, a pooled OLS model is estimated (with a message).
unit	Unquoted unit identifier column (for metadata and fe auto-inference).
time	Unquoted time variable column. Used for (a) fe auto-inference and (b) D_it construction when timing is provided.
covariates	One-sided formula of additional controls, e.g. ~ x1 + x2.
cluster	Clustering specification: a one-sided formula (~ id), a single character column name, or a numeric vector of length nrow(data). When cluster is specified and vcov is the default "HC1", cluster-robust standard errors are used automatically.
weights	Observation weights (formula or numeric vector).
conf.level	Confidence level(s) for CIs. Scalar or vector (e.g., c(0.90, 0.95)). Default 0.95.
vcov	VCOV type string passed to fixest::vcov(). Default "HC1". Ignored in favour of cluster-robust SE when cluster is supplied and vcov is left at its default "HC1".
vcov_args	Named list of additional arguments forwarded to fixest::vcov().

Value

A did_result object (named list) with elements:

estimates Data frame with the treatment coefficient: term, estimate, std.error, statistic, p.value, and conf_low_XX/conf_high_XX for each conf.level entry.

model The underlying fixest model object.

Attributes: call, formula_str, outcome, treatment, timing, fe, vcov_type, cluster_vars, conf.level, N, N_units, N_treated, unit, time. N, N_units, and N_treated describe the *estimation* sample (after fixest::feols() drops rows with missing values), matching nobs(model) and broom::glance(), not nrow(data).

Examples

```
## Not run:
# Option A: pre-built D_it
df$D <- as.integer(df$treated & df$year >= 2006)
res <- run_did(df, outcome = y, treatment = D, fe = ~ id + year)

# Option B: timing-based construction
res <- run_did(df, outcome = y, treatment = treated, time = year,
               timing = 2006, fe = ~ id + year)

# Cluster-robust SE
res <- run_did(df, outcome = y, treatment = D, fe = ~ id + year,
               cluster = ~ id)

print(res)
broom::tidy(res)
broom::glance(res)
# modelsummary::modelsummary(res)

## End(Not run)
```

---

run_es *Event Study Estimation for Panel Data*

Description

Runs an event study regression on panel data, supporting both classic (universal timing) and staggered (unit-varying timing via sunab). The function builds the design (lead/lag factor or sunab), estimates with fixest::feols(), and returns a tidy table with metadata.

Usage

```
run_es(
  data,
  outcome,
  treatment = NULL,
  time,
  timing,
  fe = NULL,
  lead_range = NULL,
  lag_range = NULL,
  covariates = NULL,
```

```
  cluster = NULL,
  weights = NULL,
  baseline = -1L,
  interval = 1,
  time_transform = FALSE,
  rel_time = NULL,
  unit = NULL,
  staggered = FALSE,
  method = c("classic", "sunab"),
  estimator = c("twfe", "cs", "sa", "bjs", "twm", "flex"),
  control_group = c("nevertreated", "notyettreated"),
  anticipation = 0L,
  conf.level = 0.95,
  vcov = "HC1",
  vcov_args = list(),
  bootstrap = FALSE,
  B = 999L,
  alpha = 0.05,
  boot_seed = NULL,
  group = NULL,
  trends = FALSE
)
```

Arguments

`data`	A data.frame containing panel data.
`outcome`	Unquoted outcome (name or expression, e.g., `log(y)`).
`treatment`	Unquoted treatment indicator (0/1 or logical). Used only when `method = "classic"`.
`time`	Unquoted time variable (numeric or Date).
`timing`	For `classic`: a numeric/Date (universal) or a variable (unquoted) if `staggered = TRUE`. For `sunab`: an unquoted variable with adoption time. For `estimator = "cs"`: unquoted column giving each unit's first treatment period (`NA` = never treated). **Convention for staggered estimators (cs, sa, bjs, twm, flex, classic with `staggered = TRUE`):** `NA` in the `timing` column means the unit is *never treated* and will be used as a control. This follows the same convention as `did::att_gt()`, `fixest::sunab()`, and `didimputation`. If `NA` instead represents *missing* treatment timing for an otherwise treated unit, those observations will be silently absorbed into the control group, which is almost certainly wrong. For `estimator = "twfe"` with `staggered = TRUE`, a warning is emitted when units with `treatment = 1` also have `timing = NA`.
`fe`	One-sided fixed-effects formula, e.g., `~ id + year`. Can be `NULL` for no fixed effects. Ignored when `estimator = "cs"`.
`lead_range, lag_range`	Integers for pre/post windows. If `NULL`, determined automatically.
`covariates`	One-sided formula of additional controls, e.g., `~ x1 + log(x2)`.
`cluster`	Cluster specification (one-sided formula like `~ id + year`, a single character column name, or a vector of length `nrow(data)`).

weights	Observation weights (a name/one-sided formula or a numeric vector of length nrow(data)).
baseline	Integer baseline period (default -1); reference period excluded from results for both "classic" and "sunab" methods.
interval	Numeric spacing of the time variable (default 1; ignored internally for Dates).
time_transform	Logical; if TRUE, creates consecutive integer time within unit.
rel_time	Optional unquoted column holding a **pre-built event time** (time relative to treatment), e.g. the Time_to_Treatment produced by paneltools/fect's get.cohort(). When supplied, run_es() uses this column verbatim as the event-study factor — i(rel_time, treatment, ref = baseline) — instead of computing relative time internally from time and timing. Never-treated / control rows should be NA. Requires estimator = "twfe" and method = "classic"; cannot be combined with staggered = TRUE or time_transform = TRUE. This normalises the older time = event_time, timing = 0 idiom. See **Details** for how this differs from the calendar-difference and time_transform conventions.
unit	Unit identifier variable (required when estimator = "cs" or time_transform = TRUE); also used for metadata when supplied.
staggered	Logical; if TRUE, timing is a variable (classic) or is used by sunab.
method	Either "classic" or "sunab" (default: "classic").
estimator	Estimation strategy: "twfe" (default, existing fixest-based paths), "cs" for the Callaway-Sant'Anna (2021) group-time ATT estimator, or "sa" for the Sun-Abraham (2021) interaction-weighted estimator.
control_group	For estimator = "cs": comparison group, either "nevertreated" (default) or "notyettreated".
anticipation	For estimator = "cs": number of anticipation periods before treatment (non-negative integer, default 0L).
conf.level	Numeric vector of confidence levels (default 0.95).
vcov	VCOV type passed to fixest::vcov() or used via broom::tidy(vcov = ...). Default "HC1".
vcov_args	List of additional arguments forwarded to fixest::vcov().
bootstrap	Logical; if TRUE and estimator = "cs", compute simultaneous confidence bands via the multiplier bootstrap (Algorithm 1, Callaway and Sant'Anna 2021). Adds conf_low_sim and conf_high_sim columns to the result and stores the full (g,t)-level bootstrap object as attr(result, "bootstrap"). Default FALSE.
B	Integer number of bootstrap draws (default 999). Used only when bootstrap = TRUE and estimator = "cs".
alpha	Significance level for the simultaneous band (default 0.05). Note: this is independent of conf.level, which governs the pointwise delta-method CIs.
boot_seed	Integer seed for the bootstrap RNG; NULL (default) does not set a seed. Pass an integer for reproducible results.
group	Unquoted group identifier for estimator = "flex" only. Identifies which treatment group (cohort) each observation belongs to in a repeated cross-section design (R_{ig} in Deb et al. 2024). Each group must map to exactly one value of timing (or NA for never-treated groups). Not used by other estimators.

`trends` Logical; for `estimator = "twm"` only. When `TRUE`, adds cohort-specific linear trend regressors $d_g \cdot t$ to the Procedure 5.1 regression (Wooldridge 2025, Section 8), allowing each cohort's counterfactual trend to deviate linearly from the common time trend. Requires at least 2 pre-treatment periods per cohort. Default `FALSE`.

Details

How relative (event) time is defined. `run_es()` can build the event-time axis in three different ways, and in sparse or unbalanced panels (e.g. monthly surveys observed only a few times a year) they do **not** coincide:

Method	Definition	How to invoke
Calendar difference (default)	`(time - timing) / interval`	`staggered = TRUE` (or scalar `timing`)
Within-unit rank	`dense_rank(time)` within each unit	`time_transform = TRUE`
Pre-built event time	supplied column, used verbatim	`rel_time = <col>`

In a balanced panel all three agree. When observation spacing is irregular they diverge: for a unit observed at periods `{1, 5, 9, 20}` treated at 20, the calendar difference puts period 9 at `9 - 20 = -11`, whereas a global observed-period grid (as used by `paneltools/fect`'s `get.cohort()`) places it two observed waves before treatment (`-2`). To reproduce an analysis built on such a grid, pass that grid's event-time column via `rel_time` so the design matches `fixest::feols(y ~ i(rel_time, treatment, ref) | fe)` exactly. `calc_att()` uses the same calendar-difference convention as the default here.

Value

A `data.frame` of class `"es_result"` with columns:

- `term`, `estimate`, `std.error`, `statistic`, `p.value`
- `conf_low_XX`, `conf_high_XX` (for each requested `conf.level`)
- `relative_time` (integer; 0 = event), `is_baseline` (logical; marks the reference period)

Attributes include: `lead_range`, `lag_range`, `baseline`, `interval`, `call`, `model_formula`, `conf.level`, `N`, `N_units`, `N_treated`, `N_nevertreated`, `fe`, `vcov_type`, `cluster_vars`, `staggered`, `sunab_used`.

Key Features

- One-step event study: specify outcome, treatment, time, timing, fixed effects, and (optionally) covariates.
- Switch between Classic (factor expansion) and Staggered-SAFE (`method = "sunab"`).
- Flexible clustering, weights, and VCOV choices (e.g., `vcov = "HC1" | "HC3" | "CR2" | "iid" ...`).
- Automatic lead/lag window detection and customizable baseline period.
- Returns an `"es_result"` object compatible with `print()` and `autoplot()`.

`tidy.att_result` *Tidy an* `att_result` *object*

Description

Returns a tidy data frame of the aggregated ATT estimates from a `calc_att()` result, in the column format expected by `modelsummary::modelsummary()` (and therefore renderable with `tinytable::tt()`). This makes it easy to put overall, by-cohort, or by-time ATTs into a publication table, and to stack several estimators (e.g. `run_did()` TWFE, CS, and BJS) side by side in one table.

The `term` column is derived from the aggregation type so that rows align across models:

- `aggregation = "simple"` -> `"ATT"`
- `aggregation = "by_cohort"` -> `"Cohort <g>"`
- `aggregation = "by_time"` -> `"Time <t>"`

Usage

```
## S3 method for class 'att_result'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

`x`	An `att_result` object returned by `calc_att()`.
`conf.int`	Logical; add `conf.low`/`conf.high` columns? Default `FALSE`.
`conf.level`	Confidence level for `conf.int` (normal approximation). Default `0.95`.
`...`	Unused; for S3 generic compatibility.

Value

A data frame with columns `term`, `estimate`, `std.error`, `statistic`, `p.value` (and optionally `conf.low`, `conf.high`).

Examples

```
## Not run:
att <- calc_att(df, outcome = y, time = year, timing = g, unit = id,
                estimator = "cs", aggregation = "by_cohort")
broom::tidy(att)

# Compare estimators in one table:
twfe <- run_did(df, outcome = y, treatment = D, fe = ~ id + year)
cs   <- calc_att(df, outcome = y, time = year, timing = g, unit = id,
                 estimator = "cs")
bjs  <- calc_att(df, outcome = y, time = year, timing = g, unit = id,
                 estimator = "bjs")
modelsummary::modelsummary(list(TWFE = twfe, CS = cs, BJS = bjs))
```

```
## End(Not run)
```

`tidy.did_result` *Tidy a* `did_result` *object*

Description

Returns a tidy data frame of model coefficients from a `run_did()` result. Delegates to `broom::tidy.fixest()` on the underlying `fixest` model so that all regressors (treatment and covariates) appear in the output — the format expected by `modelsummary::modelsummary()`.

Usage

```
## S3 method for class 'did_result'
tidy(x, conf.int = FALSE, conf.level = 0.95, ...)
```

Arguments

`x`	A `did_result` object returned by `run_did()`.
`conf.int`	Logical; add `conf.low`/`conf.high` columns? Default `FALSE`.
`conf.level`	Confidence level for `conf.int`. Default `0.95`.
`...`	Additional arguments passed to `broom::tidy.fixest()`.

Value

A data frame with columns `term`, `estimate`, `std.error`, `statistic`, `p.value` (and optionally `conf.low`, `conf.high`).

Examples

```
## Not run:
res <- run_did(df, outcome = y, treatment = D, fe = ~ id + year)
broom::tidy(res)
broom::tidy(res, conf.int = TRUE)

## End(Not run)
```

Index

autoplot.att_gt_result *(*plot_att_gt*)*, 10
autoplot.es_result, 2
autoplot.honest_result *(*plot_honest*)*, 15

broom::glance.fixest(), *8*
broom::tidy.fixest(), *23*

calc_att, 3
calc_att(), *7*, *16*, *22*
compute_contamination_weights, 5
compute_contamination_weights(), *11, 12*

fixest::feols(), *16*, *18*

ggplot2::ggplot(), *11, 12*
glance.att_result, 7
glance.did_result, 8

honest_sensitivity, 8
honest_sensitivity(), *15, 16*

plot_att_gt, 10, *11*
plot_contamination_weights, 11
plot_contamination_weights(), *6*
plot_es, 12
plot_es(), *2*, *10, 11*
plot_es_interactive, 14
plot_honest, 15
plot_honest(), *10*

run_did, 16
run_did(), *4*, *8*, *22, 23*
run_es, 18
run_es(), *2*, *4–6*, *9–11*, *13–16*

tidy.att_result, 22
tidy.did_result, 23