Package: flexMatch (via r-universe)

October 21, 2026

Title Flexible Matching for Stacked Difference-in-Differences

Version 0.1.0

Description Tools for building stacked difference-in-differences samples in staggered-adoption settings. The package labels valid treated and control candidates, performs cross-sectional matching, expands matched units into event-time stacks, and returns diagnostics for balance assessment and visualization.

License MIT + file LICENSE

Depends R (>= 4.1.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports data.table, ggplot2, MatchIt, scales

Suggests fixest, knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/avkazakovgit/flexMatch

BugReports https://github.com/avkazakovgit/flexMatch/issues

VignetteBuilder knitr

NeedsCompilation no

Author Aleksandr Kazakov [aut, cre, cph]

Maintainer Aleksandr Kazakov <avkazakov@proton.me>

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

Date/Publication 2026-10-21 15:40:08 UTC

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

RemoteRef HEAD

RemoteSha 0c66e5bc73a3fe9ac38d913a8161a34b25f2624e

Contents

assess_match_quality 2
expand_matched_stacks 3
fit_event_study 4
flexmatch 5
flexmatch_example 8
label_candidates 9
match_candidates 11
plot_event_dynamics 12

Index **14**

`assess_match_quality` *Assess Matching Quality*

Description

Assess Matching Quality

Usage

```
assess_match_quality(
  match_res,
  stack_candidates,
  dt,
  match_vars,
  id_col,
  time_col
)
```

Arguments

`match_res`	List returned by `match_candidates()`.
`stack_candidates`	Candidate table returned by `label_candidates()`.
`dt`	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
`match_vars`	Character vector of covariates used for matching. These variables are measured at the period defined by `match_by` and are passed into `MatchIt::matchit()`.
`id_col`	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
`time_col`	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.

Value

A list with balance_table and optional numeric_balance_panel and categorical_balance_panel ggplot objects. Threshold labels in the table are descriptive heuristics, not statistical tests of exchangeability.

Examples

```
data("flexmatch_example")
d <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
c <- label_candidates(d, "unit_id", "year", "event", 1, 1, 1)
m <- match_candidates(d, c, "unit_id", "year", "size", method = "nearest")
q <- assess_match_quality(m, c, d, "size", "unit_id", "year")
q$balance_table
print(q$numeric_balance_panel)
```

`expand_matched_stacks` *Expand Matched Cross-Section to Full Stacks*

Description

Expand Matched Cross-Section to Full Stacks

Usage

```
expand_matched_stacks(
  dt,
  matched_data,
  id_col,
  time_col,
  k_pre,
  k_post,
  match_by = -1,
  include_all_years = FALSE
)
```

Arguments

`dt`	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
`matched_data`	Matched cross section returned by `match_candidates()`.
`id_col`	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
`time_col`	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.
`k_pre`	Integer. Required number of pre-treatment periods. These periods are retained in the standard event window and are also used when checking that treated and control candidates have enough clean history.

`k_post` Integer. Required number of post-treatment periods. These periods are retained in the standard event window and are also used when checking whether recent or upcoming events contaminate a control window.

`match_by` Integer. Relative period used for matching covariates. Default `-1`. For example, `-1` matches on the period immediately before the event and `-2` matches on two periods before the event.

`include_all_years` Logical. If `FALSE`, keep only periods in the standard event window from `T-k_pre` through `T+k_post`. If `TRUE`, keep every observed period for each matched unit and collapse periods before the window to `-k_pre-1` and after the window to `k_post+1`.

Value

A data.table with one row per matched unit, matched set, and retained period. It includes stack_id, cohort_id, rel_time, treat, post, and weights. The original unit column name is stored as an attribute for clustering.

Examples

```
data("flexmatch_example")
d <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
c <- label_candidates(d, "unit_id", "year", "event", 1, 1, 1)
m <- match_candidates(d, c, "unit_id", "year", "size", method = "nearest")
s <- expand_matched_stacks(d, m$matched_cross_section, "unit_id", "year", 1, 1)
table(s$rel_time)
```

`fit_event_study` *Fit an Event-Study Model with fixest*

Description

Convenience wrapper around `fixest::feols()` for the stacked dataset returned by `flexmatch()`.

Usage

```
fit_event_study(final_data, outcome, time_col, ref = -1, cluster = NULL, ...)
```

Arguments

`final_data` Analysis-ready stacked data from `flexmatch()`.

`outcome` Character scalar naming the outcome column.

`time_col` Character scalar naming the calendar-time fixed effect column.

`ref` Reference period for `i(rel_time, treat)`. Default is `-1`.

`cluster` Character vector of clustering variables. By default, cluster by the original unit identifier stored by `expand_matched_stacks()`. Supply this explicitly for other data.

`...` Additional arguments passed to `fixest::feols()`.

Details

This is an illustrative regression wrapper. Identification and the interpretation of pooled event coefficients require appropriate parallel-trends, anticipation, treatment-effect, and weighting assumptions. Repeated units are clustered together by default. Pass estimator options through ...; do not supply a second weights argument.

Value

A fixest model with matching weights, unit-within-set fixed effects, matched-set-by-period fixed effects, and clustered standard errors.

Examples

```
if (requireNamespace("fixest", quietly = TRUE)) {
  data("flexmatch_example")
  d <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
  r <- flexmatch(d, "unit_id", "year", "event", 1, 1, 1, "size",
    method = "nearest", assess_quality = FALSE, plot_events = FALSE,
    verbose = FALSE)
  fit <- fit_event_study(r$final_expanded_data, "outcome", "year")
  coef(fit)
}
```

`flexmatch` *Flexmatch: Automated Stacked Difference-in-Differences with Flexible Matching Logic*

Description

`flexmatch` orchestrates a complete Stacked Difference-in-Differences (DiD) pipeline. The package is designed for staggered-adoption panel settings where users want to:

1. identify valid treated and control observations around treatment events,
2. match treated and control units on covariates measured at a chosen relative time,
3. expand the matched sample into event-time stacks for estimation, and
4. inspect diagnostics before fitting an event-study model.

`flexmatch()` is the main high-level wrapper. It builds clean treatment stacks, matches treated units to valid controls, expands the matched sample into event-time panels, and returns diagnostics for review. The package also ships with `flexmatch_example`, a synthetic unbalanced panel dataset that lets users try different matching and expansion options immediately.

Usage

```
flexmatch(
  dt,
  id_col,
  time_col,
  event_col,
  k_pre,
  k_post,
  k_freeze,
  match_vars,
  match_exact = NULL,
  match_by = -1,
  include_all_years = FALSE,
  treated_type = "first",
  control_type = "all",
  assess_quality = TRUE,
  plot_events = TRUE,
  verbose = TRUE,
  ...
)
```

Arguments

`dt`	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
`id_col`	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
`time_col`	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.
`event_col`	Character. The name of the 0/1 dummy indicating the start of treatment. The variable should equal 1 only in the period when a treatment event begins.
`k_pre`	Integer. Required number of pre-treatment periods. These periods are retained in the standard event window and are also used when checking that treated and control candidates have enough clean history.
`k_post`	Integer. Required number of post-treatment periods. These periods are retained in the standard event window and are also used when checking whether recent or upcoming events contaminate a control window.
`k_freeze`	Integer. Additional cool-down periods used to keep treated units out of the control pool. Increasing this value makes the control pool stricter because recently treated units stay "dirty" for longer after the event window ends.
`match_vars`	Character vector of covariates used for matching. These variables are measured at the period defined by `match_by` and are passed into `MatchIt::matchit()`.
`match_exact`	Optional character vector of exact-match covariates. Use this when matched units must agree exactly on selected dimensions such as industry or region.

`match_by` Integer. Relative period used for matching covariates. Default `-1`. For example, `-1` matches on the period immediately before the event and `-2` matches on two periods before the event.

`include_all_years` Logical. If `FALSE` (default), stacks keep only the standard event window from `T-k_pre` through `T+k_post`. If `TRUE`, each matched unit keeps all periods available in the input data; periods before the standard window are collapsed to `-k_pre-1` and periods after the window are collapsed to `k_post+1`.

`treated_type` Character controlling which treatment events are allowed to define stacks.

- `"first"` keeps only the first treatment event observed for each unit.
- `"all"` allows later treatment events too, provided they satisfy the clean-window rules.

`control_type` Character controlling which observations can act as controls.

- `"all"` uses any observation with a clean event window.
- `"nyt"` uses only eventually-treated units observed before their first treatment.
- `"nt"` uses observations that are not currently contaminated, including eventually-treated units.
- `"nevert"` restricts controls to units that are never treated in the full sample.

`assess_quality` Logical. If `TRUE`, generate covariate balance diagnostics. This returns balance tables and numeric/categorical balance plots when possible.

`plot_events` Logical. If `TRUE`, generate the event dynamics plot. This is useful for visually checking event timing and matched-sample coverage.

`verbose` Logical. If `TRUE`, print progress updates.

`...` Additional arguments passed to `MatchIt::matchit()`. Typical options include `method`, `ratio`, `replace`, and `caliper`.

Value

A list containing the stacked analysis data, matched cross section, candidate pool, diagnostics, optional plots, and the original `matchit` object. Important components are:

- `final_expanded_data`: the analysis-ready stacked panel.
- `matched_cross_section`: matched treated/control observations at the matching period.
- `stack_candidates`: all observations that passed the temporal eligibility rules.
- `diagnostics`: counts of candidates, matches, unmatched treated units, and rows dropped for missing covariates.
- `quality_stats`: balance outputs if `assess_quality = TRUE`.
- `event_plot`: the event-dynamics plot if `plot_events = TRUE`.

Examples

```
data("flexmatch_example")
demo_panel <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
results <- flexmatch(demo_panel, "unit_id", "year", "event",
  k_pre = 1, k_post = 1, k_freeze = 1, match_vars = "size",
  method = "nearest", assess_quality = TRUE, plot_events = TRUE,
  verbose = FALSE)
head(results$final_expanded_data)
results$quality_stats$balance_table
print(results$event_plot)
```

---

flexmatch_example *Example Panel Dataset for* `flexmatch()`

Description

A large unbalanced unit-year panel dataset shipped with the package to demonstrate treatment timing, matching, and stack expansion options.

Usage

```
flexmatch_example
```

Format

A data frame with 65000 rows and 8 variables:

unit_id Integer unit identifier.

year Calendar year.

industry Industry category used for exact matching examples.

region Geographic region label.

size Continuous covariate for matching.

leverage Continuous covariate for matching.

event Binary indicator equal to 1 in periods when treatment starts.

outcome Example outcome variable with treatment-driven dynamics.

Details

The dataset was designed to let users experiment with:

- `treated_type` = `"first"` versus `"all"`,
- `control_type` = `"all"`, `"nyt"`, `"nt"`, and `"nevert"`,
- exact matching with `match_exact`,
- standard event windows versus `include_all_years` = `TRUE`, and

- downstream estimation with `fit_event_study()`.

The dataset has 10,000 units observed between 2013 and 2022, but units enter and exit the panel at different dates, so histories range from 3 to 10 years. Units are assigned deterministically to treatment patterns including never treated, once treated, late treated, sparse repeated treatment, and frequent treatment. This makes the dataset useful for illustrating the practical differences between `treated_type`, `control_type`, `match_exact`, and `include_all_years`.

Examples

```
data("flexmatch_example")
head(flexmatch_example)
with(flexmatch_example, table(event))
```

`label_candidates` *Label Treatment and Control Stack Candidates*

Description

Identifies valid stack candidates while respecting treatment timing, clean windows, and exact matching period lookup.

Usage

```
label_candidates(
  dt,
  id_col,
  time_col,
  event_col,
  k_pre,
  k_post,
  k_freeze,
  match_by = -1,
  treated_type = c("first", "all"),
  control_type = c("all", "nyt", "nt", "nevert")
)
```

Arguments

`dt`	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
`id_col`	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
`time_col`	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.
`event_col`	Character. The name of the 0/1 dummy indicating the start of treatment. The variable should equal 1 only in the period when a treatment event begins.

`k_pre` Integer. Required number of pre-treatment periods. These periods are retained in the standard event window and are also used when checking that treated and control candidates have enough clean history.

`k_post` Integer. Required number of post-treatment periods. These periods are retained in the standard event window and are also used when checking whether recent or upcoming events contaminate a control window.

`k_freeze` Integer. Additional cool-down periods used to keep treated units out of the control pool. Increasing this value makes the control pool stricter because recently treated units stay "dirty" for longer after the event window ends.

`match_by` Integer. Relative period used for matching covariates. Default `-1`. For example, `-1` matches on the period immediately before the event and `-2` matches on two periods before the event.

`treated_type` Character controlling which treatment events are allowed to define stacks.

- `"first"` keeps only the first treatment event observed for each unit.
- `"all"` allows later treatment events too, provided they satisfy the clean-window rules.

`control_type` Character controlling which observations can act as controls.

- `"all"` uses any observation with a clean event window.
- `"nyt"` uses only eventually-treated units observed before their first treatment.
- `"nt"` uses observations that are not currently contaminated, including eventually-treated units.
- `"nevert"` restricts controls to units that are never treated in the full sample.

Details

Period distances use the ordered periods present anywhere in the panel. Missing calendar periods across the entire input cannot be inferred. Controls must have a clean full window; treated units must have clean pre-event observations and no additional event inside the post-event window. All unit/time pairs must be unique.

Value

A data.table with eligible unit/cohort rows, a treatment flag, and the period at which matching covariates are observed.

Examples

```
data("flexmatch_example")
d <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
candidates <- label_candidates(d, "unit_id", "year", "event", 1, 1, 1)
table(candidates$treat)
```

match_candidates *Cross-Sectional Matching for Stacked DiD*

Description

Performs exact cohort matching plus flexible 1-to-K matching via `MatchIt`.

Usage

```
match_candidates(
  dt,
  stack_candidates,
  id_col,
  time_col,
  match_vars,
  match_exact = NULL,
  ...
)
```

Arguments

`dt`	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
`stack_candidates`	Candidate table returned by `label_candidates()`.
`id_col`	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
`time_col`	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.
`match_vars`	Character vector of covariates used for matching. These variables are measured at the period defined by `match_by` and are passed into `MatchIt::matchit()`.
`match_exact`	Optional character vector of exact-match covariates. Use this when matched units must agree exactly on selected dimensions such as industry or region.
`...`	Additional arguments passed to `MatchIt::matchit()`. Typical options include `method`, `ratio`, `replace`, and `caliper`.

Details

Matching is exact on the event cohort period, in addition to any exact covariates supplied. Matching with replacement can reuse a unit in different matched sets. Rows with missing matching covariates are removed. The methods 'cem' and 'subclass' are rejected because their pooled calls ignore the required exact-cohort argument. For method = 'exact', the cohort and exact covariates are included directly in the matching formula.

Value

A list with matched_cross_section, the original MatchIt match_object, and diagnostics for missing matching covariates.

Examples

```
data("flexmatch_example")
d <- flexmatch_example[flexmatch_example$unit_id <= 100, ]
c <- label_candidates(d, "unit_id", "year", "event", 1, 1, 1)
m <- match_candidates(d, c, "unit_id", "year", "size", method = "nearest")
head(m$matched_cross_section)
```

---

plot_event_dynamics *Plot Event Dynamics (Lasagna Plot with Overlay)*

Description

Plot Event Dynamics (Lasagna Plot with Overlay)

Usage

```
plot_event_dynamics(
  dt,
  dt_matched = NULL,
  id_col,
  time_col,
  event_col,
  n_bins = 4L,
  sort_by_intensity = FALSE,
  show_data = NULL
)
```

Arguments

dt	A `data.table` (or data.frame) containing the panel data. Each row should be one unit-period observation.
dt_matched	Optional matched cross section used to mark selected units.
id_col	Character. The name of the unique identifier column. This identifies the panel unit, such as a firm, county, school, or person.
time_col	Character. The name of the time identifier column. This can be a year, quarter, date, or any sortable period variable.
event_col	Character. The name of the 0/1 dummy indicating the start of treatment. The variable should equal 1 only in the period when a treatment event begins.
n_bins	Positive integer number of treatment-intensity categories.
sort_by_intensity	Logical; order units by treatment intensity.

`show_data` Optional character scalar naming a variable in `dt`. When supplied, periods with non-missing data for that variable are shown as a light gray background layer, and unit ordering also uses the first and last periods with observed data before treatment-intensity tie-breaks.

Value

A list containing a ggplot object in plot and the unit order_table.

Examples

```
data("flexmatch_example")
d <- flexmatch_example[flexmatch_example$unit_id <= 50, ]
p <- plot_event_dynamics(d, id_col = "unit_id", time_col = "year",
  event_col = "event", show_data = "outcome")
print(p$plot)
```

Index

∗ **datasets**
    flexmatch_example, 8

assess_match_quality, 2

expand_matched_stacks, 3
expand_matched_stacks(), *4*

fit_event_study, 4
flexmatch, 5
flexmatch_example, 8

label_candidates, 9
label_candidates(), *2*, *11*

match_candidates, 11
match_candidates(), *2*, *3*

plot_event_dynamics, 12