

# Package ‘PanelMatch’

June 4, 2024

**Type** Package

**Title** Matching Methods for Causal Inference with Time-Series  
Cross-Sectional Data

**Version** 2.2.0

**Date** 2024-06-02

**Description** Implements a set of methodological tools that enable researchers to apply matching methods to time-series cross-sectional data. Imai, Kim, and Wang (2023) <<http://web.mit.edu/insong/www/pdf/tscs.pdf>> proposes a nonparametric generalization of the difference-in-differences estimator, which does not rely on the linearity assumption as often done in practice. Researchers first select a method of matching each treated observation for a given unit in a particular time period with control observations from other units in the same time period that have a similar treatment and covariate history. These methods include standard matching methods based on propensity score and Mahalanobis distance, as well as weighting methods. Once matching and refinement is done, treatment effects can be estimated with standard errors. The package also offers diagnostics for researchers to assess the quality of their results.

**License** GPL (>= 3)

**Imports** Rcpp (>= 0.12.5), data.table, ggplot2, CBPS, stats, graphics,  
MASS, Matrix, doParallel, foreach, methods

**Depends** R (>= 2.14.0)

**LinkingTo** RcppArmadillo, Rcpp, RcppEigen

**Encoding** UTF-8

**LazyData** true

**BugReports** <https://github.com/insongkim/PanelMatch/issues>

**RoxygenNote** 7.3.1

**Suggests** knitr, rmarkdown, testthat (>= 2.1.0)

**VignetteBuilder** knitr

**NeedsCompilation** yes

**Author** In Song Kim [aut, cre],  
Adam Rauh [aut],  
Erik Wang [aut],  
Kosuke Imai [aut]

**Maintainer** In Song Kim <insong@mit.edu>

**Repository** CRAN

**Date/Publication** 2024-06-04 09:47:42 UTC

## R topics documented:

balance_scatter . . . . .	2
dem . . . . .	4
DisplayTreatment . . . . .	5
enforce_lead_restrictions . . . . .	7
get_covariate_balance . . . . .	8
get_set_treatment_effects . . . . .	10
handle_moderating_variable . . . . .	11
matched_set . . . . .	12
PanelEstimate . . . . .	13
PanelMatch . . . . .	16
perunitSum . . . . .	19
perunitSum_Dit . . . . .	20
placebo_test . . . . .	21
plot.matched.set . . . . .	22
plot.PanelEstimate . . . . .	24
print.matched.set . . . . .	25
summary.matched.set . . . . .	26
summary.PanelEstimate . . . . .	27
<b>Index</b>	<b>28</b>

---

balance_scatter	<i>balance_scatter</i>
-----------------	------------------------

---

## Description

Visualizing the standardized mean differences for covariates via a scatter plot.

## Usage

```
balance_scatter(  
  matched_set_list,  
  xlim = c(0, 0.8),  
  ylim = c(0, 0.8),  
  main = "Standardized Mean Difference of Covariates",  
  pchs = c(2, 3),  
  covariates,  
  data,  
  x.axis.label = "Before refinement",  
  y.axis.label = "After refinement",  
  ...  
)
```

## Arguments

<code>matched_set_list</code>	a list of one or more <code>matched.set</code> objects
<code>xlim</code>	<code>xlim</code> of the scatter plot. This is the same as the <code>xlim</code> argument in <code>plot()</code>
<code>ylim</code>	<code>ylim</code> of the scatter plot. This is the same as the <code>ylim</code> argument in <code>plot()</code>
<code>main</code>	title of the scatter plot. This is the same as the <code>main</code> argument in <code>plot()</code>
<code>pchs</code>	one or more <code>pch</code> indicators for the symbols on the scatter plot. You should specify a <code>pch</code> symbol for each <code>matched.set</code> you specify in <code>matched_set_list</code> . See <code>plot()</code> for more information
<code>covariates</code>	variables for which balance is displayed
<code>data</code>	the same time series cross sectional data set used to create the matched sets.
<code>x.axis.label</code>	x axis label
<code>y.axis.label</code>	y axis label
<code>...</code>	optional arguments to be passed to <code>plot()</code>

## Details

`balance_scatter` visualizes the standardized mean differences for each covariate. Although users can use the scatter plot in a variety of ways, it is recommended that the x-axis refers to balance for covariates before refinement, and y-axis refers to balance after refinement. Users can utilize parameters powered by `plot()` in base R to further customize the figure.

## Author(s)

In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, Adam Rauh <amrauh@umich.edu>, and Kosuke Imai <imai@harvard.edu>

## Examples

```

dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
# get a matched set without refinement
sets0 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "none",
  data = dem.sub, match.missing = FALSE,
  size.match = 5, qoi = "att",
  outcome.var = "y",
  lead = 0:4, forbid.treatment.reversal = FALSE)

# get a matched set with refinement using propensity score matching, setting the
# size of matched set to 5
sets1 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem.sub, match.missing = FALSE,
  covs.formula = ~ tradewb,
  size.match = 5, qoi = "att",
  outcome.var = "y",
  lead = 0:4, forbid.treatment.reversal = FALSE)

# get another matched set with refinement using propensity score weighting
sets2 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.weight",
  data = dem.sub, match.missing = FALSE,
  covs.formula = ~ tradewb,
  size.match = 10, qoi = "att",
  outcome.var = "y",
  lead = 0:4, forbid.treatment.reversal = FALSE)

# use the function to produce the scatter plot
balance_scatter(matched_set_list = list(sets0$att, sets1$att, sets2$att),
  data = dem.sub,
  covariates = c("y", "tradewb"))

# add legend
legend(x = 0, y = 0.8,
  legend = c("mahalanobis",
    "PS weighting"),
  y.intersp = 0.65,
  x.intersp = 0.3,
  xjust = 0,
  pch = c(1, 3), pt.cex = 1,
  bty = "n", ncol = 1, cex = 1, bg = "white")

```

**Description**

A dataset containing the democracy indicator for 184 countries from 1960 to 2010

**Format**

A data.frame containing 9384 rows and 3 variables

**Details**

- wbcodes. World Bank country ID. Integer.
- year. year (1960–2010). Integer.
- dem. binary indicator of democracy as defined in Acemoglu et al (2019).
- y log of GDP per capita in 2000 constant dollars (multiplied by 100). Numeric.
- tradewb Exports plus imports as a share of GDP from World Bank. Numeric.

**Source**

Acemoglu, Daron, Suresh Naidu, Pascual Restrepo, and James A Robinson. “Democracy does cause growth.” *Journal of Political Economy*.

---

DisplayTreatment	<i>DisplayTreatment</i>
------------------	-------------------------

---

**Description**

DisplayTreatment visualizes the treatment distribution across units and time in a panel data set

**Usage**

```
DisplayTreatment(  
  unit.id,  
  time.id,  
  treatment,  
  data,  
  color.of.treated = "red",  
  color.of.untreated = "blue",  
  title = "Treatment Distribution \n Across Units and Time",  
  xlab = "Time",  
  ylab = "Unit",  
  x.size = NULL,  
  y.size = NULL,  
  legend.position = "none",  
  x.angle = NULL,  
  y.angle = NULL,  
  legend.labels = c("not treated", "treated"),  
  decreasing = FALSE,
```

```

    matched.set = NULL,
    show.set.only = FALSE,
    hide.x.tick.label = FALSE,
    hide.y.tick.label = FALSE,
    gradient.weights = FALSE,
    dense.plot = FALSE
  )

```

### Arguments

<code>unit.id</code>	Name of the unit identifier variable as a character string
<code>time.id</code>	Name of the time identifier variable as a character string
<code>treatment</code>	Name of the treatment variable as a character string
<code>data</code>	data.frame that contains the time series cross sectional data used for matching and estimation. Unit and time data must be integers. Time data must also be formatted as sequential integers that increase by one.
<code>color.of.treated</code>	Color of the treated observations provided as a character string (this includes hex values). Default is red.
<code>color.of.untreated</code>	Color of the untreated observations provided as a character string (this includes hex values). Default is blue.
<code>title</code>	Title of the plot provided as character string
<code>xlab</code>	Character label of the x-axis
<code>ylab</code>	Character label of the y-axis
<code>x.size</code>	Numeric size of the text for xlab or x axis tick labels. Assign <code>x.size = NULL</code> to use built in <code>ggplot2</code> method of determining label size. When the length of the time period is long, consider setting to <code>NULL</code> and adjusting size and ratio of the plot.
<code>y.size</code>	Numeric size of the text for ylab or y axis tick labels. Assign <code>y.size = NULL</code> to use built in <code>ggplot2</code> method of determining label size. When the number of units is large, consider setting to <code>NULL</code> and adjusting size and ratio of the plot.
<code>legend.position</code>	Position of the legend. Provide this according to <code>ggplot2</code> standards.
<code>x.angle</code>	Angle (in degrees) of the tick labels for x-axis
<code>y.angle</code>	Angle (in degrees) of the tick labels for y-axis
<code>legend.labels</code>	Character vector of length two describing the labels of the legend to be shown in the plot. <code>ggplot2</code> standards are used.
<code>decreasing</code>	Logical. Determines if display order should be increasing or decreasing by the amount of treatment received. Default is <code>decreasing = FALSE</code> .
<code>matched.set</code>	a <code>matched.set</code> object (optional) containing a single treated unit and a set of matched controls. If provided, this set will be highlighted on the resulting plot.
<code>show.set.only</code>	logical. If <code>TRUE</code> , only the treated unit and control units contained in the provided <code>matched.set</code> object will be shown on the plot. Default is <code>FALSE</code> . If no <code>matched.set</code> is provided, then this argument will have no effect.

<code>hide.x.tick.label</code>	logical. If TRUE, x axis tick labels are not shown. Default is FALSE.
<code>hide.y.tick.label</code>	logical. If TRUE, y axis tick labels are not shown. Default is FALSE.
<code>gradient.weights</code>	logical. If TRUE, the "darkness"/shade of units in the provided <code>matched.set</code> object will be displayed according to their weight. Control units with higher weights will appear darker on the resulting plot. Control units with lower weights will appear lighter. This argument has no effect unless a <code>matched.set</code> is provided.
<code>dense.plot</code>	logical. if TRUE, lines between tiles are removed on resulting plot. This is useful for producing more readable plots in situations where the number of units and/or time periods is very high.

### Value

`DisplayTreatment` returns a treatment variation plot (using `ggplot2` `geom_tile()` or `geom_raster()`), which visualizes the variation of treatment across unit and time.

### Author(s)

In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, Adam Rauh <amrauh@umich.edu>, and Kosuke Imai <imai@harvard.edu>

### Examples

```
DisplayTreatment(unit.id = "wbcode2",
                 time.id = "year", legend.position = "none",
                 xlab = "year", ylab = "Country Code",
                 treatment = "dem", data = dem)
```

---

`enforce_lead_restrictions`

*enforce\_lead\_restrictions check treatment and control units for treatment reversion in the lead window. Treated units must stay treated and control units must stay in control (according to the specified qoi)*

---

### Description

`enforce_lead_restrictions` check treatment and control units for treatment reversion in the lead window. Treated units must stay treated and control units must stay in control (according to the specified qoi)

**Usage**

```
enforce_lead_restrictions(
  matched_sets,
  ordered.data,
  max.lead,
  t.var,
  id.var,
  treatment.var
)
```

**Arguments**

matched_sets	matched.set object
ordered.data	parsed data as data.frame object
max.lead	The largest lead value (e.g. the biggest F)
t.var	string specifying the time variable
id.var	string specifying the unit id variable
treatment.var	string specifying the treatment variable.

**Value**

matched.set object with the matched sets that meet the conditions

---

get\_covariate\_balance *Calculate covariate balance*

---

**Description**

Calculate covariate balance for user specified covariates across matched sets. Balance is assessed by taking the average of the difference between the values of the specified covariates for the treated unit(s) and the weighted average of the control units across all matched sets. Results are standardized and are expressed in standard deviations. Balance is calculated for each period in the specified lag window.

**Usage**

```
get_covariate_balance(
  matched.sets,
  data,
  covariates,
  use.equal.weights = FALSE,
  plot = FALSE,
  reference.line = TRUE,
  legend = TRUE,
  ylab = "SD",
```



```

    include.treatment.period = TRUE,
    legend.position = "topleft",
    ...
)

```

### Arguments

<code>matched.sets</code>	A <code>matched.set</code> object
<code>data</code>	The time series cross sectional data set (as a <code>data.frame</code> object) used to produce the <code>matched.set</code> object. This data set should be identical to the one passed to <code>PanelMatch()</code> and <code>PanelEstimate()</code> to ensure consistent results.
<code>covariates</code>	a character vector, specifying the names of the covariates for which the user is interested in calculating balance.
<code>use.equal.weights</code>	logical. If set to <code>TRUE</code> , then equal weights will be assigned to control units, rather than using whatever calculated weights have been assigned. This is helpful for assessing the improvement in covariate balance as a result of refining the matched sets.
<code>plot</code>	logical. When <code>TRUE</code> , a plot showing the covariate balance calculation results will be shown. When <code>FALSE</code> , no plot is made, but the results of the calculations are returned. default is <code>FALSE</code>
<code>reference.line</code>	logical indicating whether or not a horizontal line should be present on the plot at <code>y = 0</code> . Default is <code>TRUE</code> .
<code>legend</code>	logical indicating whether or not a legend identifying the variables should be included on the plot. Default is <code>TRUE</code> .
<code>ylab</code>	Label for y axis. Default is "SD". This is the same as the <code>ylab</code> argument to <code>plot()</code> .
<code>include.treatment.period</code>	logical. Default is <code>TRUE</code> . When <code>TRUE</code> , covariate balance measures for the period during which treatment occurs is included. These calculations are not included when <code>FALSE</code> . Users may wish to leave this period off in some circumstances. For instance, one would expect covariate balance to be poor during this period when treatment is continuous and a lagged outcome is included in the refinement formula.
<code>legend.position</code>	position of legend. See documentation for <code>graphics::legend</code> . Default is "topleft"
<code>...</code>	Additional graphical parameters to be passed to the <code>plot</code> function in base R.

### Examples

```

dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
#add some additional data to data set for demonstration purposes
dem.sub$rdata <- runif(runif(nrow(dem.sub)))
pm.obj <- PanelMatch(lead = 0:3, lag = 4, time.id = "year", unit.id = "wbcode2", treatment = "dem",
  outcome.var = "y", refinement.method = "ps.match",
  data = dem.sub, match.missing = TRUE,

```

```

covs.formula = ~ tradewb + rdata + I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
size.match = 5, qoi = "att")
get_covariate_balance(pm.obj$att, dem.sub, covariates = c("tradewb", "rdata"),
ylim = c(-2,2))

```

---

```

get_set_treatment_effects
      get_set_treatment_effects

```

---

## Description

Calculates the treatment effect size at the matched set level

## Usage

```
get_set_treatment_effects(pm.obj, data, lead)
```

## Arguments

<code>pm.obj</code>	an object of class <code>PanelMatch</code>
<code>data</code>	data.frame with the time series cross sectional data used for matching, refinement, and estimation
<code>lead</code>	integer (or integer vector) indicating the time period(s) in the future for which the treatment effect size will be calculated. Calculations will be made for the period $t + \text{lead}$ , where $t$ is the time of treatment. If more than one lead value is provided, then calculations will be performed for each value.

## Details

Calculate the size of treatment effects for each matched set.

## Value

a list equal in length to the number of lead periods specified to the `lead` argument. Each element in the list is a vector of the matched set level effects.

## Examples

```

dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ I(lag(tradewb, 1:4)),
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE,
  placebo.test = FALSE)

```

```
set.effects <- get_set_treatment_effects(pm.obj = PM.results, data = dem.sub, lead = 0)
```

---

handle\_moderating\_variable

*handle\_moderating\_variable handles moderating variable calculations: In practice, this just involves slicing the data up according to the moderator, calling PanelEstimate() and putting everything back together This function creates the sets of objects on which PanelEstimate() will be called. It identifies the set of valid values the moderating variable can take on.*

---

### Description

handle\_moderating\_variable handles moderating variable calculations: In practice, this just involves slicing the data up according to the moderator, calling PanelEstimate() and putting everything back together This function creates the sets of objects on which PanelEstimate() will be called. It identifies the set of valid values the moderating variable can take on.

### Usage

```
handle_moderating_variable(
  ordered.data,
  att.sets,
  atc.sets,
  PM.object,
  moderator,
  unit.id,
  time.id,
  qoi.in
)
```

### Arguments

ordered.data	data.frame
att.sets	matched.set object for the ATT or ART
atc.sets	matched.set object for the ATC
PM.object	PanelMatch object
moderator	string specifying the name of the moderating variable
unit.id	string specifying the unit id variable
time.id	string specifying the time id variable
qoi.in	string specifying the QOI

### Value

Character vector of valid moderating variable values

---

<code>matched_set</code>	<i>matched_set</i>
--------------------------	--------------------

---

### Description

`matched_set` is a constructor for the `matched.set` class.

### Usage

```
matched_set(matchedsets, id, t, L, t.var, id.var, treatment.var)
```

### Arguments

<code>matchedsets</code>	a list of treated units and matched control units. Each element in the list should be a vector of control unit ids.
<code>id</code>	A vector containing the ids of treated units
<code>t</code>	A vector containing the times of treatment for treated units.
<code>L</code>	integer specifying the length of the lag window used in matching
<code>t.var</code>	string specifying the time variable
<code>id.var</code>	string specifying the unit id variable
<code>treatment.var</code>	string specifying the treatment variable.

The constructor function returns a `matched.set` object. `matched.set` objects are a modified lists. Each element in the list is a vector of ids corresponding to the control unit ids in a matched set. Additionally, these vectors might have additional attributes – "weights". These correspond to the weights assigned to each control unit, as determined by the specified refinement method. Each element in the list also has a name, which corresponds to the unit id of the treated unit and time of treatment, concatenated together and separated by a period. `matched.set` objects also have a number of methods defined: `summary`, `plot`, and ``[``. `matched.set` objects can be modified manually as long as these conventions (and conventions about other attributes) are maintained. It is important to note that `matched.set` objects are distinct from `PanelMatch` objects. `matched.set` objects are often contained within `PanelMatch` objects.

### Details

Users should never need to use this function by itself. See below for more about `matched.set` objects.

### Value

`matched.set` objects have additional attributes. These reflect the specified parameters when using the `PanelMatch` function:

<code>lag</code>	an integer value indicating the length of treatment history to be used for matching. Treated and control units are matched based on whether or not they have exactly matching treatment histories in the lag window.
------------------	--

<code>t.var</code>	time variable name, represented as a character/string
<code>id.var</code>	unit id variable name, represented as a character/string
<code>treatment.var</code>	treatment variable name, represented as a character/string
<code>class</code>	class of the object: should always be "matched.set"
<code>refinement.method</code>	method used to refine and/or weight the control units in each set.
<code>covs.formula</code>	One sided formula indicating which variables should be used for matching and refinement
<code>match.missing</code>	Logical variable indicating whether or not units should be matched on the patterns of missingness in their treatment histories
<code>max.match.size</code>	Maximum size of the matched sets after refinement. This argument only affects results when using a matching method

**Author(s)**

Adam Rauh <amrauh@umich.edu>, In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, and Kosuke Imai <imai@harvard.edu>

---

PanelEstimate

*PanelEstimate*


---

**Description**

PanelEstimate estimates a causal quantity of interest, including the average treatment effect for treated or control units (att and atc, respectively), the average effect of treatment reversal on reversed units (art), or average treatment effect (ate), as specified in PanelMatch(). This is done by estimating the counterfactual outcomes for each treated unit using matched sets. Users will provide matched sets that were obtained by the PanelMatch function and obtain point estimates via a weighted average computation with weighted bootstrap standard errors. Point estimates and standard errors will be produced for each period in the lead window specified by the lead argument from PanelMatch(). Users may run multiple estimations by providing lists of each argument to the function. However, in this format, every argument must be explicitly specified in each configuration and must adhere to the same data types/structures outlined below.

**Usage**

```
PanelEstimate(
  sets,
  data,
  number.iterations = 1000,
  df.adjustment = FALSE,
  confidence.level = 0.95,
  moderator = NULL,
  se.method = "bootstrap",
  pooled = FALSE,
```

```

include.placebo.test = FALSE,
parallel = FALSE,
num.cores = 1
)

```

### Arguments

<code>sets</code>	A PanelMatch object attained via the PanelMatch() function.
<code>data</code>	The same time series cross sectional data set provided to the PanelMatch() function used to produce the matched sets.
<code>number.iterations</code>	If using bootstrapping for calculating standard errors, this is the number of bootstrap iterations. Provide as integer. If <code>se.method</code> is not equal to "bootstrap", this argument has no effect.
<code>df.adjustment</code>	A logical value indicating whether or not a degree-of-freedom adjustment should be performed for the standard error calculation. The default is FALSE. This parameter is only available for the bootstrap method of standard error calculation.
<code>confidence.level</code>	A numerical value specifying the confidence level and range of interval estimates for statistical inference. The default is .95.
<code>moderator</code>	The name of a moderating variable, provided as a character string. If a moderating variable is provided, the returned object will be a list of PanelEstimate objects. The names of the list will reflect the different values of the moderating variable. More specifically, the moderating variable values will be converted to syntactically proper names using <code>make.names()</code> .
<code>se.method</code>	Method used for calculating standard errors, provided as a character string. Users must choose between "bootstrap", "conditional", and "unconditional" methods. Default is "bootstrap". "bootstrap" uses a block bootstrapping procedure to calculate standard errors. The conditional method calculates the variance of the estimator, assuming independence across units but not across time. The unconditional method also calculates the variance of the estimator analytically, but makes no such assumptions about independence across units. When the quantity of interest is "att", "atc", or "art", all methods are available. Only "bootstrap" is available for the <code>ate</code> . If <code>pooled</code> argument is TRUE, then only bootstrap is available.
<code>pooled</code>	Logical. If TRUE, estimates and standard errors are returned for treatment effects pooled across the entire lead window. Only available for <code>se.method = "bootstrap"</code>
<code>include.placebo.test</code>	Logical. If TRUE, a placebo test is run and returned in the results. The placebo test uses the same specifications for calculating standard errors as the main results. That is, standard errors are calculated according to the user provided <code>se.method</code> and <code>confidence.level</code> arguments (and, if applicable, parallelization specifications). If these are invalid for some reason, an error will be thrown.
<code>parallel</code>	Logical. If TRUE and <code>se.method = "bootstrap"</code> , bootstrap procedure will be parallelized. Default is FALSE. If <code>se.method</code> is not set to bootstrap, this option does nothing.

`num.cores` Integer. Specifies the number of cores to use for parallelization. If `se.method = ``bootstrap``` and `parallel = TRUE`, then this option will take effect. Otherwise, it will do nothing.

### Value

PanelEstimate returns a list of class ‘PanelEstimate’ containing the following components:

`estimates` the point estimates of the quantity of interest for the lead periods specified

`se.method` The method used to calculate standard errors. This is the same as the argument provided to the function.

`bootstrapped.estimates` the bootstrapped point estimate values, when applicable

`bootstrap.iterations` the number of iterations used in bootstrapping, when applicable

`method` refinement method used to create the matched sets from which the estimates were calculated

`lag` See PanelMatch() argument lag for more information.

`lead` The lead window sequence for which PanelEstimate() is producing point estimates and standard errors.

`confidence.level` the confidence level

`qoi` the quantity of interest

`matched.sets` the refined matched sets used to produce the estimations

`standard.error` the standard error(s) of the point estimates

`pooled` Logical indicating whether or not estimates were calculated for individual lead periods or pooled.

`placebo.test` if `include.placebo.test = TRUE`, a placebo test is conducted using `placebo_test()` and returned as a list. See documentation for `placebo_test()` for more about each individual item.

### Author(s)

In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, Adam Rauh <amrauh@umich.edu>, and Kosuke Imai <imai@harvard.edu>

### References

Imai, Kosuke, In Song Kim, and Erik Wang (2023)

### Examples

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                        treatment = "dem", refinement.method = "ps.match",
                        data = dem.sub, match.missing = TRUE, covs.formula = ~ tradewb,
```

```

      size.match = 5, qoi = "att", outcome.var = "y",
      lead = 0:4, forbid.treatment.reversal = TRUE)
PE.results <- PanelEstimate(sets = PM.results, data = dem.sub, se.method = "unconditional")

```

---

PanelMatch

*PanelMatch*


---

### Description

Create refined/weighted sets of treated and control units

### Usage

```

PanelMatch(
  lag,
  time.id,
  unit.id,
  treatment,
  outcome.var,
  refinement.method,
  data,
  qoi,
  size.match = 10,
  match.missing = TRUE,
  covs.formula = NULL,
  lead = 0,
  verbose = FALSE,
  exact.match.variables = NULL,
  forbid.treatment.reversal = FALSE,
  matching = TRUE,
  listwise.delete = FALSE,
  use.diagonal.variance.matrix = FALSE,
  restrict.control.period = NULL,
  placebo.test = FALSE
)

```

### Arguments

lag	An integer value indicating the length of treatment history periods to be matched on
time.id	A character string indicating the name of the time variable in the data. This data currently must be formatted as sequential integers.
unit.id	A character string indicating the name of unit identifier in the data. This data must be integer.



<code>treatment</code>	A character string indicating the name of the treatment variable in the data. The treatment must be a binary indicator variable (integer with 0 for the control group and 1 for the treatment group).
<code>outcome.var</code>	A character string identifying the outcome variable.
<code>refinement.method</code>	A character string specifying the matching or weighting method to be used for refining the matched sets. The user can choose "mahalanobis", "ps.match", "CBPS.match", "ps.weight", "CBPS.weight", or "none". The first three methods will use the <code>size.match</code> argument to create sets of at most <code>size.match</code> closest control units. Choosing "none" will assign equal weights to all control units in each matched set.
<code>data</code>	A <code>data.frame</code> object containing time series cross sectional data. Time data must be sequential integers that increase by 1. Unit identifiers must be integers. Treatment data must be binary.
<code>qoi</code>	quantity of interest, provided as a string: <code>att</code> (average treatment effect on treated units), <code>atc</code> (average treatment effect of treatment on the control units) <code>art</code> (average effect of treatment reversal for units that experience treatment reversal), or <code>ate</code> (average treatment effect).
<code>size.match</code>	An integer dictating the number of permitted closest control units in a matched set after refinement. This argument only affects results when using a matching method ("mahalanobis" or any of the refinement methods that end in ".match"). This argument is not needed and will have no impact if included when a weighting method is specified (any <code>refinement.method</code> that includes "weight" in the name).
<code>match.missing</code>	Logical variable indicating whether or not units should be matched on the patterns of missingness in their treatment histories. Default is TRUE. When FALSE, neither treated nor control units are allowed to have missing treatment data in the lag window.
<code>covs.formula</code>	One sided formula object indicating which variables should be used for matching and refinement. Argument is not needed if <code>refinement.method</code> is set to "none" If the user wants to include lagged variables, this can be done using a function, "lag()", which takes two, unnamed, positional arguments. The first is the name of the variable which you wish to lag. The second is the lag window, specified as an integer sequence in increasing order. For instance, <code>I(lag(x, 1:4))</code> will then add new columns to the data for variable "x" for time t-1, t-2, t-3, and t-4 internally and use them for defining/measuring similarity between units. Other transformations using the <code>I()</code> function, such as <code>I(x^2)</code> are also permitted. The variables specified in this formula are used to define the similarity/distances between units.
<code>lead</code>	integer sequence specifying the lead window, for which <code>qoi</code> point estimates (and standard errors) will ultimately be produced. Default is 0 (which corresponds to contemporaneous treatment effect).
<code>verbose</code>	option to include more information about the <code>matched.set</code> object calculations, like the distances used to create the refined sets and weights.
<code>exact.match.variables</code>	character vector giving the names of variables to be exactly matched on. These

	should be time invariant variables. Exact matching for time varying covariates is not currently supported.
<code>forbid.treatment.reversal</code>	Logical. For the ATT, it indicates whether or not it is permissible for treatment to reverse in the specified lead window. This is defined analogously for the ART. It is not valid for the ATC or ATE. When set to TRUE, only matched sets for treated units where treatment is applied continuously in the lead window are included in the results. Default is FALSE.
<code>matching</code>	logical indicating whether or not any matching on treatment history should be performed. This is primarily used for diagnostic purposes, and most users will never need to set this to FALSE. Default is TRUE.
<code>listwise.delete</code>	TRUE/FALSE indicating whether or not missing data should be handled using listwise deletion or the package's default missing data handling procedures. Default is FALSE.
<code>use.diagonal.variance.matrix</code>	TRUE/FALSE indicating whether or not a regular covariance matrix should be used in mahalanobis distance calculations during refinement, or if a diagonal matrix with only covariate variances should be used instead. In many cases, setting this to TRUE can lead to better covariate balance, especially when there is high correlation between variables. Default is FALSE. This argument is only necessary when <code>refinement.method = mahalanobis</code> and will have no impact otherwise.
<code>restrict.control.period</code>	(optional) integer specifying the number of pre-treatment periods that treated units and potentially matched control units should be non-NULL and in the control state. For instance, specifying 4 would mean that the treatment history cannot contain any missing data or treatment from t-4 to t.
<code>placebo.test</code>	logical TRUE/FALSE. indicates whether or not you want to be able to run a placebo test. This will add additional requirements on the data – specifically, it requires that no unit included in the matching/refinement process can have missing outcome data over the lag window. Additionally, you should not use the outcome variable in refinement when <code>placebo.test = TRUE</code> .

## Details

PanelMatch identifies a matched set for each treated observation. Specifically, for a given treated unit, the matched set consists of control observations that have an identical treatment history up to a number of lag time periods. Researchers must specify `lag`. A further refinement of the matched set may be performed by setting a maximum size of each matched set, `size.match` (the maximum number of control units that can be matched to a treated unit). Users can also specify covariates that should be used to identify similar control units and a method for defining similarity/distance between units. This is done via the `covs.formula` and `refinement.method` arguments, respectively, which are explained in more detail below.

## Value

PanelMatch() returns an object of class "PanelMatch". This is a list that contains a few specific elements: First, a `matched.set` object(s) that has the same name as the provided `qoi` if the `qoi` is

"att", "art", or "atc". If `qoi = "ate"` then two `matched.set` objects will be attached, named "att" and "atc." Please consult the documentation for `matched.set()` to read more about the structure and usage of `matched.set` objects. Also, see the vignette page about `matched.set` objects for more information about these objects: `vignette("matched_set_objects", package = "PanelMatch")`. The `PanelMatch` object also has some additional attributes:

<code>qoi</code>	The <code>qoi</code> specified in the original function call
<code>lead</code>	the lead window specified in the original function call
<code>forbid.treatment.reversal</code>	logical value matching the <code>forbid.treatment.reversal</code> parameter provided in the function call.
<code>outcome.var</code>	character string matching the outcome variable provided in the original function call.

### Author(s)

Adam Rauh <amrauh@umich.edu>, In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, and Kosuke Imai <imai@harvard.edu>

### References

Imai, Kosuke, In Song Kim, and Erik Wang (2023)

### Examples

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ tradewb,
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
```

---

perunitSum	<i>perunitSum</i> This is a low level function that is used to calculate a value associated with each unit. This value is a weighted summation of the dependent variable, based on the Wit values discussed in Imai et al. (2023)
------------	---

---

### Description

`perunitSum` This is a low level function that is used to calculate a value associated with each unit. This value is a weighted summation of the dependent variable, based on the Wit values discussed in Imai et al. (2023)

**Usage**

```
perunitSum(udf, lead.in, dependent.in, qoi.in)
```

**Arguments**

udf	data.frame
lead.in	integer. A particular lead value
dependent.in	string specifying the dependent variable name
qoi.in	string specifying the QOI

**Value**

Named vector containing the per-unit sums.

---

perunitSum_Dit	<i>perunitSum_Dit Similar to perunitSum, this is a low level helper function for calculating specific values defined in Imai et al. (2023). This focuses on Dit rather than Wit</i>
----------------	---

---

**Description**

perunitSum\_Dit Similar to perunitSum, this is a low level helper function for calculating specific values defined in Imai et al. (2023). This focuses on Dit rather than Wit

**Usage**

```
perunitSum_Dit(udf, qoi.in)
```

**Arguments**

udf	data.frame
qoi.in	string specifying the QOI

**Value**

Named vector containing the per-unit sums.

---

placebo\_test                      *placebo\_test*

---

### Description

Calculates results for a placebo test

### Usage

```
placebo_test(
  pm.obj,
  data,
  lag.in = NULL,
  number.iterations = 1000,
  confidence.level = 0.95,
  plot = FALSE,
  se.method = "bootstrap",
  parallel = FALSE,
  num.cores = 1,
  ...
)
```

### Arguments

<code>pm.obj</code>	an object of class <code>PanelMatch</code>
<code>data</code>	<code>data.frame</code> with the original data
<code>lag.in</code>	integer indicating earliest the time period(s) in the future for which the placebo test change in outcome will be calculated. Calculations will be made over the period $t - \max(\text{lag})$ to $t-2$ , where $t$ is the time of treatment. The results are similar to those returned by <code>PanelEstimate()</code> , except $t-1$ is used as the period of comparison, rather than the lead window. If not specified, the placebo test is conducted for periods from $t - \max(\text{lag})$ to $t-2$ .
<code>number.iterations</code>	integer specifying the number of bootstrap iterations
<code>confidence.level</code>	confidence level for the calculated standard error intervals
<code>plot</code>	logical indicating whether or not a plot should be generated, or just return the raw data from the calculations
<code>se.method</code>	character string describing the type of standard error to be used. Valid inputs include "bootstrap", "conditional" and "unconditional". When the QOI is ATE, only bootstrap can be used.
<code>parallel</code>	Logical. If TRUE and <code>se.method = "bootstrap"</code> , bootstrap procedure will be parallelized. Default is FALSE. If <code>se.method</code> is not set to bootstrap, this option does nothing.

num.cores        Integer. Specifies the number of cores to use for parallelization. If `se.method = ``bootstrap``` and `parallel = TRUE`, then this option will take effect. Otherwise, it will do nothing.

...                extra arguments to be passed to `plot()`

### Details

Calculate the results of a placebo test, looking at the change in outcome at time =  $t-1$ , compared to other pre-treatment periods in the lag window.

### Value

list with 2 or 3 elements: "estimates", which contains the point estimates for the test, "standard.errors" which has the standard errors for each period and optionally "bootstrapped.estimates", containing the bootstrapped point estimates for the test for each specified lag window period.

### Examples

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "mahalanobis",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ tradewb,
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE,
  placebo.test = TRUE)
placebo_test(PM.results, data = dem.sub, se.method = "unconditional", plot = FALSE)
```

---

plot.matched.set        *Plot the distribution of the sizes of matched sets.*

---

### Description

A plot method for creating a histogram of the distribution of the sizes of matched sets. This method accepts all standard optional `hist` arguments via the `...` argument. By default, empty matched sets (treated units that could not be matched with any control units) are noted as a vertical bar at  $x = 0$  and not included in the regular histogram. See the `include.empty.sets` argument for more information about this.

### Usage

```
## S3 method for class 'matched.set'
plot(
  x,
  ...,
```

```

border = NA,
col = "grey",
ylab = "Frequency of Size",
xlab = "Matched Set Size",
lwd = NULL,
main = "Distribution of Matched Set Sizes",
freq = TRUE,
include.empty.sets = FALSE
)

```

### Arguments

x	a <code>matched.set</code> object
...	optional arguments to be passed to <code>hist()</code>
border	default is NA. This is the same argument as the standard argument for <code>hist()</code>
col	default is "grey". This is the same argument as the standard argument for <code>hist()</code>
ylab	default is "Frequency of Size". This is the same argument as the standard argument for <code>hist()</code>
xlab	default is "Matched Set Size". This is the same argument as the standard argument for <code>hist()</code>
lwd	default is NULL. This is the same argument as the standard argument for <code>hist()</code>
main	default is "Distribution of Matched Set Sizes". This is the same argument as the standard argument for <code>hist()</code>
freq	default is TRUE. See <code>freq</code> argument in <code>hist()</code> function for more.
include.empty.sets	logical value indicating whether or not empty sets should be included in the histogram. default is FALSE. If FALSE, then empty sets will be noted as a separate vertical bar at <code>x = 0</code> . If TRUE, empty sets will be included as normal sets.

### Examples

```

dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem, match.missing = TRUE,
  covs.formula = ~ I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
plot(PM.results$att)
plot(PM.results$att, include.empty.sets = TRUE)

```

---

plot.PanelEstimate	<i>Plot point estimates and standard errors from a PanelEstimate calculation.</i>
--------------------	---

---

## Description

The plot.PanelEstimate method takes an object returned by the PanelEstimate function and plots the calculated point estimates and standard errors over the specified lead time period. The only mandatory argument is an object of the PanelEstimate class.

## Usage

```
## S3 method for class 'PanelEstimate'
plot(
  x,
  ylab = "Estimated Effect of Treatment",
  xlab = "Time",
  main = "Estimated Effects of Treatment Over Time",
  ylim = NULL,
  pch = NULL,
  cex = NULL,
  bias.corrected = FALSE,
  ...
)
```

## Arguments

x	a PanelEstimate object
ylab	default is "Estimated Effect of Treatment." This is the same argument as the standard argument for plot()
xlab	default is "Time". This is the same argument as the standard argument for plot()
main	default is "Estimated Effects of Treatment Over Time". This is the same argument as the standard argument for plot
ylim	default is NULL. This is the same argument as the standard argument for plot()
pch	default is NULL. This is the same argument as the standard argument for plot()
cex	default is NULL. This is the same argument as the standard argument for plot()
bias.corrected	logical indicating whether or not bias corrected estimates should be plotted Default is FALSE. This argument only applies for standard errors calculated with the bootstrap.
...	Additional optional arguments to be passed to plot().



**Examples**

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "mahalanobis",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ tradewb,
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
PE.results <- PanelEstimate(sets = PM.results, data = dem.sub, se.method = "unconditional")
plot(PE.results)
```

---

```
print.matched.set      Print matched.set objects.
```

---

**Description**

Print matched.set objects.

**Usage**

```
## S3 method for class 'matched.set'
print(x, ..., verbose = FALSE)
```

**Arguments**

x	a matched.set object
...	additional arguments to be passed to print
verbose	logical indicating whether or not output should be printed in expanded/raw list form. The verbose form is not recommended unless the data set is small. Default is FALSE

**Examples**

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem, match.missing = TRUE,
  covs.formula = ~ I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
print(PM.results$att)
```

---

summary.matched.set     *Summarize information about a matched.set object and the matched sets contained within them.*

---

### Description

A method for viewing summary data about the sizes of matched sets and metadata about how they were created. This method accepts all standard summary arguments.

### Usage

```
## S3 method for class 'matched.set'
summary(object, ..., verbose = TRUE)
```

### Arguments

object	a matched.set object
...	Optional additional arguments to be passed to the summary function
verbose	Logical value specifying whether or not a longer, more verbose summary should be calculated and returned. Default is TRUE.

### Value

list object with either 5 or 1 element(s), depending on whether or not verbose is set to TRUE or not.

overview	A data.frame object containing information about the treated units (unit id, time of treatment), and the number of matched control units with weights zero and above.
set.size.summary	a summary object summarizing the minimum, maximum, and IQR of matched set sizes
number.of.treated.units	The number of unit, time pairs that are considered to be "treated" units
num.units.empty.set	The number of units treated at a particular time that were not able to be matched to any control units
lag	The size of the lag window used for matching on treatment history. This affects which treated and control units are matched.

### Examples

```
dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.match",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
```

```

      size.match = 5, qoi = "att",
      outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
summary(PM.results$att)

```

---

summary.PanelEstimate *Get summaries of PanelEstimate objects/calculations*

---

### Description

summary.PanelEstimate takes an object returned by PanelEstimate, and returns a summary table of point estimates and confidence intervals

### Usage

```

## S3 method for class 'PanelEstimate'
summary(object, verbose = TRUE, bias.corrected = FALSE, ...)

```

### Arguments

object	A PanelEstimate object
verbose	logical indicating whether or not output should be printed in an expanded form. Default is TRUE
bias.corrected	logical indicating whether or not bias corrected estimates should be provided. Default is FALSE. This argument only applies for standard errors calculated with the bootstrap.
...	optional additional arguments. Currently, no additional arguments are supported.

### Examples

```

dem.sub <- dem[dem[, "wbcode2"] <= 100, ]
# create subset of data for simplicity
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
  treatment = "dem", refinement.method = "ps.weight",
  data = dem.sub, match.missing = TRUE,
  covs.formula = ~ tradewb,
  size.match = 5, qoi = "att",
  outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
PE.results <- PanelEstimate(sets = PM.results, data = dem.sub, se.method = "unconditional")
summary(PE.results)

```

# Index

## \* dataset

dem, [4](#)

balance\_scatter, [2](#)

dem, [4](#)

DisplayTreatment, [5](#)

enforce\_lead\_restrictions, [7](#)

get\_covariate\_balance, [8](#)

get\_set\_treatment\_effects, [10](#)

handle\_moderating\_variable, [11](#)

matched\_set, [12](#)

PanelEstimate, [13](#)

PanelMatch, [16](#)

perunitSum, [19](#)

perunitSum\_Dit, [20](#)

placebo\_test, [21](#)

plot.matched.set, [22](#)

plot.PanelEstimate, [24](#)

print.matched.set, [25](#)

summary.matched.set, [26](#)

summary.PanelEstimate, [27](#)