Package ‘PanelMatch’

February 28, 2020

Type Package

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

Version 1.0.0

Date 2020-02-10

Description Implements a set of methodological tools that enable researchers to apply matching methods to time-series cross-sectional data. Imai, Kim, and Wang (2018) <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 is done, both short-term and long-term average treatment effects for the treated can be estimated with standard errors. The package also offers a visualization technique that allows researchers to assess the quality of matches by examining the resulting covariate balance.

License GPL (>= 3)

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

Depends R (>= 2.14.0)

SystemRequirements C++11

LinkingTo RcppArmadillo, Rcpp, RcppEigen

Encoding UTF-8

LazyData true
Description

Implements a set of methodological tools that enable researchers to apply matching methods to time-series cross-sectional data. Imai, Kim, and Wang (2018) 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 is done, both short-term and long-term average treatment effects for the treated can be estimated with standard errors. The package also offers a visualization technique that allows researchers to assess the quality of matches by examining the resulting covariate balance.

Details

Package: PanelMatch
Type: Package
Version: 0.0.1-
Date: 2018-03-01
License: GPL (>= 3)

Author(s)
In Song Kim <insong@mit.edu>, Erik Wang <haixiao@Princeton.edu>, Adam Rauh <adamrauh@mit.edu>, and Kosuke Imai <imai@harvard.edu>
Maintainer: In Song Kim <insong@mit.edu>

References

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

Usage
balance_scatter(non_refined_set, refined_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
non_refined_set
a matched.set object produced by setting 'refinement.method' to "none" in 'PanelMatch'
refined_list
a list of one or two matched.set objects
xlim
xlim of the scatter plot. This is the same as the xlim argument in plot
**balance_scatter**

- `ylim` ylimit of the scatter plot. This is the same as the `ylim` argument in `plot`.
- `main` title of the scatter plot. This is the same as the `main` argument in `plot`.
- `pchs` one or two `pch` indicators for the symbols on the scatter plot. See `plot` for more information.
- `covariates` variables for which balance is displayed.
- `data` the same time series cross sectional data set used to create the matched sets.
- `x.axis.label` x axis label.
- `y.axis.label` y axis label.
- ... optional arguments to be passed to `plot`.

**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 <adamrauh@mit.edu>, and Kosuke Imai <imai@harvard.edu>

**Examples**

```r
# get a matched set without refinement
sets0 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "none",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    size.match = 5, qoi = "att",
                    outcome.var = "y",
                    lead = 0:4, forbid.treatment.reversal = FALSE)

# get a matched set with refinement using CBPS.match, setting the
# size of matched set to 5
sets1 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "mahalanobis",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    size.match = 5, qoi = "att",
                    outcome.var = "y",
                    lead = 0:4, forbid.treatment.reversal = FALSE)

# get another matched set with refinement using CBPS.weight
sets2 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "ps.weight",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    size.match = 10, qoi = "att",
```

---

**balance_scatter**

- `ylim` ylimit of the scatter plot. This is the same as the `ylim` argument in `plot`.
- `main` title of the scatter plot. This is the same as the `main` argument in `plot`.
- `pchs` one or two `pch` indicators for the symbols on the scatter plot. See `plot` for more information.
- `covariates` variables for which balance is displayed.
- `data` the same time series cross sectional data set used to create the matched sets.
- `x.axis.label` x axis label.
- `y.axis.label` y axis label.
- ... optional arguments to be passed to `plot`.

**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 <adamrauh@mit.edu>, and Kosuke Imai <imai@harvard.edu>

**Examples**

```r
# get a matched set without refinement
sets0 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "none",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    size.match = 5, qoi = "att",
                    outcome.var = "y",
                    lead = 0:4, forbid.treatment.reversal = FALSE)

# get a matched set with refinement using CBPS.match, setting the
# size of matched set to 5
sets1 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "mahalanobis",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    size.match = 5, qoi = "att",
                    outcome.var = "y",
                    lead = 0:4, forbid.treatment.reversal = FALSE)

# get another matched set with refinement using CBPS.weight
sets2 <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                    treatment = "dem", refinement.method = "ps.weight",
                    data = dem, match.missing = FALSE,
                    covs.formula = ~ I(lag(y, 1:4)) + I(lag(tradewb, 1:4)),
                    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(non_refined_set = sets0$att,
    refined_list = list(sets1$att, sets2$att),
    data = dem,
    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")


dem

County-level Democracy indicator

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

Format
A dataframe containing 9384 rows and 3 variables

Details
• wbcode2. World Bank country ID
• year. year (1960–2010)
• dem. binary indicator of democracy as defined in Acemoglu et al.
• y log of GDP per capita in 2000 constant dollars (multiplied by 100)
• tradewb Exports plus Imports as a share of GDP from World Bank

Source
DisplayTreatment visualizes the treatment distribution across units and time in a panel dataset.

Usage

```
DisplayTreatment(unit.id, time.id, treatment, data, color.of.treated = "red",
                  color.of.untreated = "blue",
                  title = "Treatment Distribution Across Units and Time", xlab = "Time",
                  ylab = "Unit", x.size = 10, y.size = 5, legend.position = "none",
                  x.angle = 45, y.angle = NULL, legend.labels = c("not treated", "treated"),
                  decreasing = FALSE, matched.set = NULL,
                  show.set.only = FALSE, hide.x.axis.label = FALSE,
                  hide.y.axis.label = FALSE, gradient.weights = FALSE, dense.plot = FALSE)
```

Arguments

- **unit.id**: Name of the unit identifier variable as a character string.
- **time.id**: Name of the time identifier variable as a character string.
- **treatment**: Name of the treatment variable as a character string.
- **data**: 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.
- **color.of.treated**: Color of the treated observations provided as a character string (this includes hex values). Default is red.
- **color.of.untreated**: Color of the untreated observations provided as a character string (this includes hex values). Default is blue.
- **title**: Title of the plot provided as character string.
- **xlab**: Character label of the x-axis.
- **ylab**: Character label of the y-axis.
- **x.size**: Numeric size of the text for xlab or x axis label. Default is 10. Assign x.size = NULL to use built in ggplot2 method of determining label size. When the length of the time period is long, consider setting to NULL and adjusting size and ratio of the plot.
- **y.size**: Numeric size of the text for ylab or y axis label. Default is 5. Assign y.size = NULL to use built in ggplot2 method of determining label size. When the number of units is large, consider setting to NULL and adjusting size and ratio of the plot.
DisplayTreatment

- `legend.position`: Position of the legend. Provide this according to ggplot2 standards.
- `x.angle`: Angle (in degrees) of the tick labels for x-axis.
- `y.angle`: Angle (in degrees) of the tick labels for y-axis.
- `legend.labels`: Character vector of length two describing the labels of the legend to be shown in the plot. ggplot2 standards are used.
- `decreasing`: Logical. Determines if display order should be increasing or decreasing by the amount of treatment received. Default is `decreasing = FALSE`.
- `matched.set`: a matched.set object (optional) containing a single treated unit and a set of matched controls. If provided, this set will be highlighted on the resulting plot.
- `show.set.only`: logical. If TRUE, only the treated unit and control units contained in the provided matched.set object will be shown on the plot. Default is FALSE. If no matched.set is provided, then this argument will have no effect.
- `hide.x.axis.label`: logical. If TRUE, x axis labels are not shown. Default is FALSE.
- `hide.y.axis.label`: logical. If TRUE, y axis labels are not shown. Default is FALSE.
- `gradient.weights`: logical. If TRUE, the "darkness"/shade of units in the provided matched.set 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 matched.set is provided.
- `dense.plot`: 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), 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 <adamrauh@mit.edu>, and Kosuke Imai <imai@harvard.edu>

**Examples**

```r
DisplayTreatment(unit.id = "wbcode2",
                 time.id = "year", legend.position = "none",
                 xlab = "year", ylab = "Country Code",
                 treatment = "dem", data = dem)
```
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, verbose = TRUE, plot = FALSE, reference.line = TRUE, legend = TRUE, ylab = "SD", ...)

Arguments

matched.sets A matched.set object
data The time series cross sectional data set (as a data.frame object) used to produce the matched.set object. This data set should be identical to the one passed to PanelMatch and PanelEstimate to ensure consistent results.
covariates a character vector, specifying the names of the covariates for which the user is interested in calculating balance.
use.equal.weights logical. If set to TRUE, 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.
verbose logical. When TRUE, the function will return more information about the calculations/results. When FALSE, a more compact version of the results/calculations are returned.
plot logical. When TRUE, a plot showing the covariate balance calculation results will be shown. When FALSE, no plot is made, but the results of the calculations are returned. default is FALSE
reference.line logical indicating whether or not a horizontal line should be present on the plot at y = 0. Default is TRUE.
legend logical indicating whether or not a legend identifying the variables should be included on the plot. Default is TRUE.
ylab Label for y axis. Default is "SD". This is the same as the ylab argument to plot.
... Additional graphical parameters to be passed to the plot function in base R.
Examples

```r
# add some additional data to data set for demonstration purposes
dem$rdata <- runif(runif(nrow(dem)))
pm.obj <- PanelMatch(lead = 0:3, lag = 4, time.id = "year", unit.id = "wbcode2", treatment = "dem",
                      outcome.var = "y", refinement.method = "mahalanobis",
                      data = dem, match.missing = TRUE,
                      covs.formula = ~ tradewb + rdata + I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
                      size.match = 5, qoi = "att")
get_covariate_balance(pm.obj$att, dem, covariates = c("tradewb", "rdata"),
                      ylim = c(-2,2))
get_covariate_balance(pm.obj$att, dem, covariates = c("tradewb", "rdata"),
                      plot = TRUE, ylim = c(-2,2))
```

**Description**

matched_set is a constructor for the matched.set class.

**Usage**

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

**Arguments**

- **matchedsets**
  - a list of treated units and matched control units. Each element in the list should be a vector of control unit ids.

- **id**
  - A vector containing the ids of treated units

- **t**
  - A vector containing the times of treatment for treated units.

- **L**
  - integer specifying the length of the lag window used in matching

- **t.var**
  - string specifying the time variable

- **id.var**
  - string specifying the unit id variable

- **treatment.var**
  - 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:

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

Author(s)

Adam Rauh <adamrauh@mit.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), 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. See the included code examples for more about how this functionality works.
Usage

PanelEstimate(sets, number.iterations = 1000, df.adjustment = FALSE, confidence.level = 0.95, data)

Arguments

- **sets**: A PanelMatch object attained via the PanelMatch function.
- **number.iterations**: An integer value indicating the number of bootstrap iterations. The default is 1000.
- **df.adjustment**: A logical value indicating whether or not a degree-of-freedom adjustment should be performed for the standard error calculation. The default is FALSE.
- **confidence.level**: A numerical value specifying the confidence level and range of interval estimates for statistical inference. The default is .95.
- **data**: The same time series cross sectional data set provided to the PanelMatch function used to produce the matched sets

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
- **bootstrapped.estimates**: the bootstrapped point estimate values
- **bootstrap.iterations**: the number of iterations used in bootstrapping
- **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

Author(s)

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

References

Imai, Kosuke, In Song Kim, and Erik Wang (2018)
Examples

```r
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                          treatment = "dem", refinement.method = "mahalanobis",
                          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 = TRUE)
PE.results <- PanelEstimate(sets = PM.results, data = dem)
```

Description

Create refined/weighted sets of treated and control units

Usage

```r
PanelMatch(lag, time.id, unit.id, treatment, refinement.method,
           size.match = 10, data, match.missing = TRUE, covs.formula = NULL,
           verbose = FALSE, qoi, lead = 0, outcome.var,
           exact.match.variables = NULL, forbid.treatment.reversal = FALSE,
           matching = TRUE, listwise.delete = FALSE,
           use.diagonal.variance.matrix = 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.
- **treatment**: 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).
- **refinement.method**: 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", "ps.msm.weight", "CBPS.msm.weight", or "none". The first three methods will use the size.match argument to create sets of at most size.match closest control units. Choosing "none" will assign equal weights to all control units in each matched set.
size.match  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 refinement.method that includes "weight" in the name).

data  A data.frame 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.

match.missing  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.

covs.formula  One sided formula object indicating which variables should be used for matching and refinement. Argument is not needed if refinement.method 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, I(lag(x, 1:4)) 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 I() function, such as I(x^2) are also permitted. The variables specified in this formula are used to define the similarity/distances between units.

verbose  option to include more information about the matched.set object calculations, like the distances used to create the refined sets and weights.

qoi  quantity of interest: att (average treatment effect on treated units), atc (average treatment effect on control units), ate (average treatment effect). Note that the qoi for MSM methods will give the estimated average treatment effect of being treated for a chosen lead time periods. This differs slightly from the non-MSM methods, where treatment reversal is permitted.

lead  integer sequence specifying the lead window, for which qoi point estimates (and standard errors) will ultimately be produced. Default is 0 (which corresponds to contemporaneous treatment effect).

outcome.var  A character string identifying the outcome variable.

exact.match.variables  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.

forbid.treatment.reversal  Logical indicating whether or not it is permissible for treatment to reverse in the specified lead window. This must be set to TRUE for MSM methods. 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.

matching  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.
listwise.delete

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.

use.diagonal.variance.matrix

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 refinement.method = mahalanobis and will have no impact otherwise.

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" 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 wiki page for more information about these objects: https://github.com/insongkim/PanelMatch/wiki/Matched-Set-Objects. The PanelMatch object also has some additional attributes:

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

Author(s)

Adam Rauh <adamrauh@mit.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 (2018)
Examples

```r
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
treatment = "dem", refinement.method = "mahalanobis",
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)
#not including any lagged variables
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
treatment = "dem", refinement.method = "mahalanobis",
data = dem, match.missing = TRUE,
covs.formula = ~ tradewb,
size.match = 5, qoi = "att",
outcome.var = "y", lead = 0:4, forbid.treatment.reversal = FALSE)
# Running multiple configurations at once
list.of.results = PanelMatch(lag = list(4,3),
time.id = list("year", "year"),
unit.id = list("wbcode2", "wbcode2"),
treatment = list("dem", "dem"),
refinement.method = list("mahalanobis", "ps.weight"),
data = dem,
match.missing = list(TRUE, TRUE),
covs.formula = list(~ I(lag(tradewb, 1:4)) + I(lag(y, 1:4)),
~ I(lag(tradewb, 1:4)) + I(lag(y, 1:4))),
size.match = list(5,5),
qoi = list("att", "att"),
outcome.var = list("y", "y"),
lead = list(0:4, 0:3),
forbid.treatment.reversal = list(FALSE, FALSE),
verbose = list(FALSE, FALSE),
listwise.delete = list(FALSE,FALSE),
use.diagonal.variance.matrix = list(TRUE, NULL),
exact.match.variables = list(NULL, NULL),
matching = list(TRUE, TRUE))
```

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

```r
## 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 `matched.set` object
- **...**: optional arguments to be passed to `hist`
- **border**: default is NA. This is the same argument as the standard argument for `hist`
- **col**: default is "grey". This is the same argument as the standard argument for `hist`
- **ylab**: default is "Frequency of Size". This is the same argument as the standard argument for `hist`
- **xlab**: default is "Matched Set Size". This is the same argument as the standard argument for `hist`
- **lwd**: default is NULL. This is the same argument as the standard argument for `hist`
- **main**: default is "Distribution of Matched Set Sizes". This is the same argument as the standard argument for `hist`
- **freq**: default is TRUE. See `freq` argument in `hist` 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 x = 0. If TRUE, empty sets will be included as normal sets.

Examples

```r
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                          treatment = "dem", refinement.method = "mahalanobis",
                          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  
Plot point estimates and standard errors from a PanelEstimate calculation.

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

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

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`
- `...`: Additional optional arguments to be passed to `plot`.

Examples

```r
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
                          treatment = "dem", refinement.method = "mahalanobis",
                          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)
PE.results <- PanelEstimate(sets = PM.results, data = dem)
plot(PE.results)
```
print.matched.set  \textit{Print} matched.set \textit{objects.}

\textbf{Description}

Print matched.set objects.

\textbf{Usage}

\begin{verbatim}
## S3 method for class 'matched.set'
print(x, ..., verbose = FALSE)
\end{verbatim}

\textbf{Arguments}

- \texttt{x}  
a matched.set object  
- \texttt{...}  
  additional arguments to be passed to \texttt{print}  
- \texttt{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

\textbf{Examples}

\begin{verbatim}
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
treatment = "dem", refinement.method = "mahalanobis",
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)
\end{verbatim}

summary.matched.set \textit{Summarize information about a matched.set object and the matched sets contained within them.}

\textbf{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 \texttt{summary} arguments.

\textbf{Usage}

\begin{verbatim}
## S3 method for class 'matched.set'
summary(object, ..., verbose = TRUE)
\end{verbatim}
**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

```r
## 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
- `...`: optional additional arguments. Currently, no additional arguments are supported.

Examples

```r
PM.results <- PanelMatch(lag = 4, time.id = "year", unit.id = "wbcode2",
    treatment = "dem", refinement.method = "mahalanobis",
    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)

PE.results <- PanelEstimate(sets = PM.results, data = dem)
summary(PE.results)
```
Index

*Topic dataset
dem, 5

*Topic package
  PanelMatch-package, 2
  balance_scatter, 3
dem, 5
  DisplayTreatment, 6
  get_covariate_balance, 8
  matched_set, 9
  PanelEstimate, 10
  PanelMatch, 12
  PanelMatch-package, 2
  plot.matched.set, 15
  plot.PanelEstimate, 17
  print.matched.set, 18
  summary.matched.set, 18
  summary.PanelEstimate, 19