

Package: pss (via r-universe)

October 19, 2026

Title Propensity Score Sampling for quasi-experimental studies with an external comparison group

Version 0.0.0.9000

Description IN DEVELOPMENT! Propensity score sampling is a method for finding a matched comparison group in a quasi-experimental study where an 'external' comparison cohort is being used. By 'external', we mean a cohort for which the intervention is not available to anyone (for example, a different time or region from the treatment group).

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Depends R (>= 3.5)

LazyData true

Config/roxygen2/version 8.1.0

Imports dplyr, ggplot2, rlang, smd, tidyr, tidyselect

Suggests knitr, rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs libicu-dev

Repository <https://racheloughton.r-universe.dev>

Date/Publication 2026-10-19 12:36:11 UTC

RemoteUrl <https://github.com/RachelOughton/pss>

RemoteRef HEAD

RemoteSha 9c2761b8f87c6685295b2ce69c0d0e07b7fd88c0

Contents

candidates	2
dropped_df	4

eg_data	5
expand_matched_df	6
matched_sample	7
plot_smd	9
propscores	11
pss_check_data	14
Index	15

candidates	<i>Create a summary table for a vector of seeds</i>
------------	---

Description

Create a summary table for a vector of seeds

Usage

```
candidates(  
  df,  
  propscores,  
  seeds,  
  smd_cols,  
  smd_order,  
  replace = TRUE,  
  downsample = FALSE,  
  drop_int = FALSE,  
  use_attr = TRUE,  
  cov_cols = NULL,  
  arm_col = NULL,  
  intervention_level = NULL  
)
```

Arguments

df	A data frame from which to generate matched samples
propscores	A data frame (created by propscores() , or with the same format) giving a propensity score (in column PropScore) for every covariate being matched on
seeds	A vector of random seeds to be used
smd_cols	A character vector of columns for which to assess balance. This can be the same as cov_cols but it doesn't have to be.
smd_order	An integer denoting the highest order of interaction for which to calculate the standardised mean difference (SMD)
replace	logical. Passed to matched_sample() . Should the comparison cases be sampled with replacement?

downsample	logical. Passed to <code>matched_sample()</code> . Should the treatment cases be down-sampled when there aren't enough comparison cases? Only one of <code>replace</code> and <code>downsample</code> should be TRUE.
drop_int	logical. Passed to <code>matched_sample()</code> . Should cases in the intervention/treated group be dropped if there are no equivalent cases in the comparison cohort to sample from?
use_attr	logical. Should the attributes of <code>propscores</code> (set by <code>propscores()</code>) be used to specify <code>cov_cols</code> , <code>arm_col</code> and <code>intervention_level</code> ? Default TRUE
cov_cols	(only specify if <code>use_attr == FALSE</code>) A character vector containing the column names of the covariates to be matched on. These should all be factor / categorical data.
arm_col	(only specify if <code>use_attr == FALSE</code>) The name of the column indicating which rows are treated cases and which are comparison cases. These should be factor or character, with only two levels / options.
intervention_level	(only specify if <code>use_attr == FALSE</code>) The value in the <code>arm_col</code> for the treatment cases

Value

`candidates()` returns a data frame with a row for each random seed and columns

- seed
- n_int: the number of treated / intervention cases included in the matched sample for that seed
- n_comp: the number of comparison cases included in the matched sample for that seed
- SMD_...: column for each covariate and covariate interaction, giving the calculated SMD, found using `smd::smd()`

See Also

`matched_sample()`, `expand_matched_df()`

Examples

```
## First use `propscores` to create a propensity score data frame
## This example uses the joint approach and specifies p(Z=1) such
## that the expected size of the comparison group is 300
```

```
eg_j_exp300 <- propscores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_expectedN(300)
)
```

```
## Now we can use `eg_j_exp300` to find a matched comparison group
```

```
## for each seed in the vector `seeds`. [candidates()] calls
## [matched_sample()] each time it does this, and we specify that
## it should downsample the intervention cases (`downsample = TRUE`)
## rather than sampling from the comparison cases with replacement.
## In the output of [candidates()] we include SMD values for
## four variables and include interactions up to order 3. Because
## `suspect_age` is numeric it will not be included in the interactions.

candidates(
  df = eg_data,
  propscores = eg_j_exp300,
  seeds = 1:20,
  smd_cols = c("suspect_age", "category", "risk_ass", "sus_age_bin"),
  smd_order = 3,
  downsample = TRUE,
  replace = FALSE
)
```

dropped_df	<i>Display the treatment cases that will be dropped under a particular matched sample</i>
------------	---

Description

Display the treatment cases that will be dropped under a particular matched sample

Usage

```
dropped_df(matched_df, use_attr = TRUE, cov_cols = NULL)
```

Arguments

matched_df	A data frame output by <code>matched_sample()</code> . This should contain columns for <code>arm_col</code> , all elements of <code>cov_cols</code> and <code>include</code> . It must have the <code>Arm</code> column with levels "Intervention" and "Comparison". It may also contain other columns.
use_attr	logical. Should the attributes of <code>matched_df</code> (set by <code>matched_sample()</code>) be used to specify <code>cov_cols</code> ?
cov_cols	(only specify if <code>use_attr == FALSE</code>) A character vector containing the column names of the covariates to be matched on. These should all be factor / categorical data.

Value

`dropped_df()` returns a data frame with the following columns

- A column for each of `cov_cols`

- `n_dropped`: how many treated / intervention cases with this combination of `cov_cols` have been dropped / downsampled
- `n_kept`: how many treated / intervention cases with this combination of `cov_cols` have been kept The rows will be ordered in descending order of `n_dropped`

See Also

[matched_sample\(\)](#)

Examples

```
## First use [propscores()] to create a propensity score data frame
## This example uses the joint approach and specifies p(Z=1) such
## that the expected size of the comparison group is 300

eg_j_exp300 <- propscores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_expectedN(300)
)

## Now we can use `eg_j_exp300` to find a matched comparison group
## In this version we sample from the comparison group with replacement
## and do not drop any intervention cases for which there are no
## comparison cases to sample

match_j_exp300_replace <- matched_sample(
  df = eg_data,
  propscores = eg_j_exp300,
  seed = 20,
  replace = TRUE,
  downsample = FALSE,
  drop_int = FALSE
)

## Finally we use [dropped_df()] to summarise the treated / intervention
## cases that have been dropped or downsampled
## included `include` times

match_j_exp300_replace |> dropped_df()
```

Description

An anonymised and slightly altered version of data from a study investigating forensic marking to protect victims of domestic abuse in the UK

Usage

```
eg_data
```

Format

eg_data:

A data frame with 999 rows and 20 columns:

FM_equip Whether the victim was given forensic marking equipment

warning_YN Whether the perpetrator was warned about the use of forensic marking equipment (can only be Y if FM_equip is TRUE)

princ_crime The principal crime committed at the reference incident, according to the home office lookup table ...

expand_matched_df	<i>Expand a matched comparison data frame</i>
-------------------	---

Description

Expand a matched comparison data frame

Usage

```
expand_matched_df(matched_df)
```

Arguments

matched_df A data frame output by [matched_sample\(\)](#). This should contain columns for arm_col, all elements of cov_cols and include. It can also contain other columns.

Value

[expand_matched_df\(\)](#) returns a data frame with all the columns of matched_df except include. Each row (row i) is now included include[i] times. That is, rows with matched_df\$include[i]==0 are omitted from the returned data frame, and rows with matched_df\$include[i] > 1 are included multiple times.

See Also

[matched_sample\(\)](#)

Examples

```
## First use `propscores` to create a propensity score data frame
## This example uses the joint approach and specifies p(Z=1) such
## that the expected size of the comparison group is 300

eg_j_exp300 <- propscores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_expectedN(300)
)

## Now we can use `eg_j_exp300` to find a matched comparison group
## In this version we sample from the comparison group with replacement
## and do not drop any intervention cases for which there are no
## comparison cases to sample

match_j_exp300_replace <- matched_sample(
  df = eg_data,
  propscores = eg_j_exp300,
  seed = 20,
  replace = TRUE,
  downsample = FALSE,
  drop_int = FALSE
)

## Finally we use `expand_matched_df` to expand this so that each row is
## included `include` times

match_j_exp300_replace |> expand_matched_df()
```

matched_sample	<i>Form a matched sample</i>
----------------	------------------------------

Description

`matched_sample()` uses propensity score sampling to generate a matched sample.

Usage

```
matched_sample(
  df,
  propscores,
  seed,
  replace = TRUE,
  downsample = FALSE,
```

```

drop_int = FALSE,
use_attr = TRUE,
cov_cols = NULL,
arm_col = NULL,
intervention_level = NULL
)

```

Arguments

df	A data frame containing treatment group and comparison cohort data. These should
propscores	A data frame with a column for each variable in cov_cols and a PropScore column giving the estimated propensity score for each combination of their levels. This should (usually) have been created by <code>propscores()</code> using df as input.
seed	The random seed to be used, for reproducibility
replace	logical. Should the comparison cases be sampled with replacement?
downsample	logical. Should the treatment cases be downsampled when there aren't enough comparison cases? Only one of replace and downsample should be TRUE.
drop_int	logical. Should cases in the intervention/treated group be dropped if there are no equivalent cases in the comparison cohort to sample from?
use_attr	logical. Should the attributes of propscores (set by <code>propscores()</code>) be used to specify cov_cols, arm_col and intervention_level?
cov_cols	(only specify if use_attr == FALSE) A character vector containing the column names of the covariates to be matched on. These should all be factor / categorical data.
arm_col	(only specify if use_attr == FALSE) The name of the column indicating which rows are treated cases and which are comparison cases. These should be factor or character, with only two levels / options.
intervention_level	(only specify if use_attr == FALSE) The value in the arm_col for the treatment cases

Value

`matched_sample()` returns a data frame that is an expanded form of df, with the same number of rows, in the same order, but with some extra columns added:

- PropScore: the propensity score for that row, from propscores
- include: how many copies of this row are included in the matched dataset. Zero means the row has been dropped. Values greater than one mean the row will be duplicated.
- seed: the random seed that was used. This will be the same for all rows and is included in the output for reproducibility.

The data frame can be expanded into a usable data frame (that is, with row i included include[i] times) using `expand_matched_df()`.

`matched_sample()` will also issue messages indicating:

- How many comparison cases have been repeated (and how many times) if `replace == TRUE`
- How many intervention cases have been downsampled, if `downsample == TRUE`
- How many (if any) intervention cases have been dropped because there are no similar comparison cases to sample from, if `drop_int == TRUE`
- How many (if any) intervention cases have been kept even though there are no similar comparison cases to sample from, if `drop_int == TRUE`

See Also

[expand_matched_df\(\)](#)

Examples

```
## First use `propscores` to create a propensity score data frame
## This example uses the joint approach and specifies p(Z=1) such
## that the expected size of the comparison group is 300

eg_j_exp300 <- propscores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_expectedN(300)
)

## Now we can use `eg_j_exp300` to find a matched comparison group
## In this example we use the attribute information in `propscores`,
## we sample from the comparison group with replacement
## and do not drop any intervention cases for which there are no
## comparison cases to sample
match_j_exp300_replace <- matched_sample(
  df = eg_data,
  propscores = eg_j_exp300,
  seed = 20,
  replace = TRUE,
  downsample = FALSE,
  drop_int = FALSE
)
```

plot_smd

Plots standardised mean difference (SMD) for each covariate and interaction in a [candidates\(\)](#) data frame. If a threshold value for the SMD is given (via `smd_thresh`) then the output for those seeds with any SMD value exceeding the threshold are greyed, and a message is generated listing the successful seeds.

Description

Plots standardised mean difference (SMD) for each covariate and interaction in a `candidates()` data frame. If a threshold value for the SMD is given (via `smd_thresh`) then the output for those seeds with any SMD value exceeding the threshold are greyed, and a message is generated listing the successful seeds.

Usage

```
plot_smd(summary_df, smd_thresh = NULL)
```

Arguments

<code>summary_df</code>	A data frame generated by <code>candidates()</code> , containing diagnostic summaries of pss matched samples for a vector of seeds
<code>smd_thresh</code>	Numerical. The threshold above which the SMD is considered 'high'. If NULL, no threshold is used and all lines are the same

Value

A plot with covariates and covariate interactions on the x axis and SMD on the y axis. Lines connect SMD values for the same seed.

See Also

`candidates()`

Examples

```
## First use `propscores` to create a propensity score data frame
## This example uses the joint approach and specifies p(Z=1) such
## that the expected size of the comparison group is 300

eg_j_exp300 <- propscores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_expectedN(300)
)

## Now we can use [candidates()] to tabulate the SMD values for
## four variables and include interactions up to order 3. Because
## `suspect_age` is numeric it will not be included in the interactions.

sdf20 = candidates(
  df = eg_data,
  propscores = eg_j_exp300,
  seeds = 1:20,
  smd_cols = c("suspect_age", "category", "risk_ass", "sus_age_bin"),
  smd_order = 3,
```

```

    downsample = TRUE,
    replace = FALSE
  )

  ## Finally we use [plot_smd()] to display the SMD values for each
  ## covariate and interaction. If a numeric value is given for `smd_thresh`
  ## then points and lines for seeds for which at lease one SMD values
  ## is above the threshold are plotted with transparency `alpha = 0.5`.

  plot_smd(sdf20, smd_thresh = 0.15)

```

propcores	<i>Propensity score data frame</i>
-----------	------------------------------------

Description

`propcores()` generates a data frame of propensity score estimates from study / trial data, to be used in the propensity score sampling method, using the Bayes Theorem factorisation of the propensity score

$$p(Z = 1 \mid x) = \frac{p(x \mid Z = 1)p(Z = 1)}{p(x)}$$

Of the functions listed, only `propcores()` should be used directly.

`.props_fun()` is called by `propcores()` to find the proportions of the data within each category for each arm. One of `pz1_estimate()`, `pz1_rescale()` or `pz1_expectedN()` should be given (with its own argument `n_pz1`) as the `pz1_fn` argument, to determine which method is used to specify $p(Z = 1)$.

Usage

```

propcores(df, cov_cols, arm_col, intervention_level, cov_dist, pz1_fn)

.props_fun(df, cov_cols, cov_list)

pz1_estimate(n_pz1)

pz1_rescale(n_pz1)

pz1_expectedN(n_pz1)

```

Arguments

<code>df</code>	A data frame containing treatment group and comparison cohort data
<code>cov_cols</code>	A character vector containing the column names of the covariates to be matched on. These should all be factor / categorical data.

<code>arm_col</code>	The name of the column indicating which rows are treated cases and which are comparison cases. These should be factor or character, with only two levels / options.
<code>intervention_level</code>	The value in the <code>arm_col</code> for the treatment cases
<code>cov_dist</code>	Either "joint" for fully joint distribution, "marginal" for fully marginal distribution, or a list of character vectors indicating groups of covariates that should be considered jointly.
<code>pz1_fn</code>	One of the functions used to specify $p(Z=1)$: <code>pz1_rescale()</code> , <code>pz1_estimate()</code> or <code>pz1_expectedN()</code> . Each of these has one numerical argument, <code>n_pz1</code> , whose meaning depends on the function being used.
<code>cov_list</code>	An argument to <code>[.props_fun()]</code> . This should be a list of character vectors. The elements of the list determine which groups of covariates are treated jointly. If <code>cov_dist</code> is "marginal" or "joint" then this is created automatically from <code>cov_cols</code> . If <code>cov_dist</code> is a list of groups of covariates to be treated jointly then the covariates to be treated marginally are filled in automatically.
<code>n_pz1</code>	An argument to <code>pz1_fn</code> . Its meaning depends on the function used for <code>pz1_fn</code> .

Value

`propcores()` returns a data frame with a row for each combination of values of `cov_cols` and the following columns

- a column for each element of `cov_cols`
- `px_int`, the estimate of $p(x|Z = 1)$
- `px_comp`, the estimate of $p(x)$
- `count_comp`, the number of cases in the comparison cohort
- `count_int`, the number of cases in the intervention cohort
- `PropScore`, which contains the estimated propensity score
- Some further columns, depending on which method is used

The data frame is assigned attributes `cov_cols`, `arm_col` and `intervention_level`, storing the information for use in subsequent functions such as `matched_sample()` (if `use_attr` is set to `TRUE`).

`.props_fun` returns a function that creates a data frame containing `px_int` ($p(x|Z = 1)$), `px_comp` $p(x)$ and `ratio` (`px_int / px_comp`) for every combination of the levels of the covariates in `cov_cols`, using the approach specified via `cov_dist`.

`pz1_estimate` returns the estimate of $p(Z = 1)$ using the 'estimate' method (eg. a value provided by an expert). The argument is used as the estimate.

`pz1_rescale` finds the value of $p(Z = 1)$ such that the largest propensity score will be `n_pz1` (the argument given to `pz1_fn`)

`pz1_expectedN` finds the value of $p(Z = 1)$ such that the expected size of the matched comparison group will be `n_pz1` (the argument to `pz1_fn`).

Functions

- `.props_fun()`: Find table of estimates of $p(x)$, $p(x|Z = 1)$ and their ratio using fully joint approach
- `pz1_estimate()`: Specify $p(Z = 1)$ using the estimate method. The argument given is the estimate used
- `pz1_rescale()`: Specify $p(Z = 1)$ using the rescale method. The argument given is the value the largest estimated propensity score should take
- `pz1_expectedN()`: Specify $p(Z = 1)$ using the expectedN method. The argument given should be the expected size of the matched comparison group

Examples

```
# Creates a propensity score data frame from `eg_data`
# using the joint approach, and rescaling so that the
# highest propensity score is 1.
propcores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "joint",
  pz1_fn = pz1_rescale(1)
)

# Creates a propensity score data frame from `eg_data`
# using the fully marginal approach, and setting p(Z=1)
# so that the expected size of the matched comparison
# group will be 300
propcores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "marginal",
  pz1_fn = pz1_expectedN(300)
)

# Creates a propensity score data frame from `eg_data`
# for which "risk_ass" and "category" are considered
# jointly and "sus_age_bin" marginally, and setting p(Z=1)
# so that the expected size of the matched comparison
# group will be 300
propcores(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention",
  cov_dist = "marginal",
  pz1_fn = pz1_expectedN(300)
)
```

pss_check_data	<i>Check data</i>
----------------	-------------------

Description

pss_check_data checks that the data frame provided is suitable. Will return either a very slightly cleaned up version, or an error message. The function does the following:

- checks the arm_col column has two unique values, one of which is intervention_level
- adds an Arm column, with two levels ("Intervention" and "Comparison")
- checks the columns named in cov_cols are present and in a suitable form (ideally factor, but if character it will coerce to factor and give a warning)
- Removes rows for which any of cov_cols or arm_col is missing.

Usage

```
pss_check_data(df, cov_cols, arm_col, intervention_level)
```

Arguments

df	A data frame containing treatment group and comparison cohort data
cov_cols	A character vector containing the column names of the covariates to be matched on. These should all be factor / categorical data.
arm_col	The name of the column indicating which rows are treated cases and which are comparison cases. These should be factor or character, with only two levels / options.
intervention_level	The value in the arm_col for the treatment cases

Value

pss_check_data returns a (possibly slightly tidied up) data frame or an error message. The data frame will have an Arm column with levels Intervention and Comparison.

Examples

```
pss_check_data(
  df = eg_data,
  cov_cols = c("risk_ass", "category", "sus_age_bin"),
  arm_col = "Arm",
  intervention_level = "Intervention"
)
```

Index

* datasets

- eg_data, [5](#)
- .props_fun(propscores), [11](#)
- .props_fun(), [11](#)
- candidates, [2](#)
- candidates(), [3](#), [9](#), [10](#)
- dropped_df, [4](#)
- dropped_df(), [4](#)
- eg_data, [5](#)
- expand_matched_df, [6](#)
- expand_matched_df(), [3](#), [6](#), [8](#), [9](#)
- matched_sample, [7](#)
- matched_sample(), [2-8](#), [12](#)
- plot_smd, [9](#)
- propscores, [11](#)
- propscores(), [2](#), [3](#), [8](#), [11](#), [12](#)
- pss_check_data, [14](#)
- pz1_estimate(propscores), [11](#)
- pz1_estimate(), [11](#)
- pz1_expectedN(propscores), [11](#)
- pz1_expectedN(), [11](#)
- pz1_rescale(propscores), [11](#)
- pz1_rescale(), [11](#)
- smd: : smd(), [3](#)