Package ‘WMAP’

January 20, 2025

Title Weighted Meta-Analysis with Pseudo-Populations

Version 1.1.0

Description Implementation of integrative weighting approaches for multiple observational studies and causal inferences. The package features three weighting approaches, each representing a special case of the unified weighting framework, introduced by Guha and Li (2024) <doi:10.1093/biomtc/ujae070>, which includes an extension of inverse probability weights for data integration settings.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

LazyData true

Imports pkgcond, ggplot2, zeallot, caret, randomForest, forcats, utils, stats, grDevices

Depends R (>= 3.5.0)

NeedsCompilation no

Author Subharup Guha [aut, cre],
Mengqi Xu [aut],
Kashish Priyam [aut],
Yi Li [aut]

Maintainer Subharup Guha <s.guha@ufl.edu>

Repository CRAN

Date/Publication 2024-11-30 05:10:02 UTC

Contents

`balancing.weights` *Compute balancing weights using FLEXOR or other methods*

Description

This function calculates balancing weights based on the specified pseudo-population method.

Usage

```
balancing.weights(
  S,
  Z,
  X,
  method,
  naturalGroupProp,
  num.random = 40,
  gammaMin = 0.001,
  gammaMax = (1 - 0.001),
  seed = NULL,
  verbose = TRUE
)
```

Arguments

`S` Vector of factor levels representing the study memberships. Takes values in {1, ..., J}.

`Z` Vector of factor levels representing the group memberships. Takes values in {1, ..., K}.

`X` Covariate matrix of N rows and p columns.

`method` Pseudo-population method, i.e., weighting method. Take values in `FLEXOR`, `IC`, or `IGO`.

`naturalGroupProp` Relevant only for FLEXOR method: a fixed user-specified probability vector θ.

`num.random` Relevant only for FLEXOR method: number of random starting points of γ in the two-step iterative procedure. Default is 40.

`gammaMin` Relevant only for FLEXOR method: Lower bound for each γ_s in the two-step iterative procedure. Default is 0.001.

`gammaMax` Relevant only for FLEXOR method: Upper bound for each γ_s in the two-step iterative procedure. Default is 0.999.

`seed` Seed for random number generation. Default is `NULL`.

`verbose` Logical; Relevant only for FLEXOR method: if `TRUE` (default), displays progress messages during computation to the console. Set to `FALSE` to suppress these messages.

Value

An S3 list object with the following components:

wt.v N empirically normalized sample weights.

percentESS Percentage sample effective sample size (ESS) for the pseudo-population.

Examples

```
data(demo)
balancing.weights(S, Z, X, method = "IC", naturalGroupProp)
```

`causal.estimate` *Estimate causal effects using FLEXOR or other methods*

Description

This function estimates causal effects based on the specified pseudo-population method. The FLEXOR method involves an iterative two-step procedure.

Usage

```
causal.estimate(
  S,
  Z,
  X,
  Y,
  B = 100,
  method,
  naturalGroupProp = NULL,
  num.random = 40,
  gammaMin = 0.001,
  gammaMax = (1 - 0.001),
  seed = NULL,
  verbose = TRUE
)
```

Arguments

`S` Vector of factor levels representing the study memberships. Takes values in {1, ..., J}.

`Z` Vector of factor levels representing the group memberships. Takes values in {1, ..., K}.

`X` Covariate matrix of N rows and p columns.

`Y` Matrix of L outcomes, with dimensions $N \times L$.

`B` Number of bootstrap samples for variance estimation. Default is 100.

`method` Pseudo-population method, i.e., weighting method. Take values in `FLEXOR`, `IC`, or `IGO`.

`naturalGroupProp` Relevant only for FLEXOR method: a fixed user-specified probability vector θ.

`num.random` Relevant only for FLEXOR method: number of random starting points of γ in the two-step iterative procedure. Default is 40.

`gammaMin` Relevant only for FLEXOR method: Lower bound for each γ_s in the two-step iterative procedure. Default is 0.001.

`gammaMax` Relevant only for FLEXOR method: Upper bound for each γ_s in the two-step iterative procedure. Default is 0.999.

`seed` Seed for random number generation. Default is `NULL`.

`verbose` Logical; if `TRUE` (default), displays progress messages during computation to the console. Set to `FALSE` to suppress these messages.

Value

An S3 list object with the following components:

percentESS Percentage sample effective sample size (ESS) of the pseudo-population.

moments.ar An array of dimension $3 \times K \times L$, containing:

- Estimated means, standard deviations (SDs), and medians (dimension 1),
- For K groups (dimension 2),
- And L counterfactual outcomes (dimension 3).

otherFeatures.v Estimated mean group differences for L outcomes.

collatedMoments.ar An array of dimension $3 \times K \times L \times B$, containing:

- `moments.ar` of the bth bootstrap sample (dimensions 1–3),
- For B bootstrap samples (dimension 4).

collatedOtherFeatures.mt A matrix of dimension $L \times B$ containing:

- `otherFeatures.v` of the bth bootstrap sample (dimension 1),
- For B bootstrap samples (dimension 2).

collatedESS A vector of length Bcontaining percentage sample ESS for B bootstrap samples.

method Pseudo-population method, i.e., weighting method.

Examples

```
data(demo)
set.seed(1)
causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
```

demo *Demo Dataset*

Description

A dataset containing example data for demonstration purposes.

Usage

```
data(demo)
```

Format

An rda object, with 450 observations and the following variables:

S A vector of factor levels, representing the study memberships.

Z A vector of factor levels, representing the group memberships.

X A covariate matrix.

Y An outcome matrix.

naturalGroupProp The relative group prevalences of the larger natural population. Necessary only for FLEXOR weights; it should be skipped for IC and IGO weights.

groupNames Disease subtype names "IDC" or "ILC"

Details

Demo Dataset

Examples

```
data(demo)
```

get_weights *Extract sample weights*

Description

A generic function to extract sample weights from objects of class 'balancing_weights'

Usage

```
get_weights(object)
```

Arguments

object	An objects of class 'balancing_weights'.

Value

Empirically normalized sample weights.

Examples

```
data(demo)
output1 <- balancing.weights(S, Z, X, method = "IC", naturalGroupProp)
get_weights(output1)
```

mean_diff *Extract causal estimates (mean differences)*

Description

A generic function to extract mean differences for objects of class 'causal_estimates'.

Usage

```
mean_diff(object)
```

Arguments

object	An objects of class 'causal_estimates'.

Value

The mean differences between two groups

Examples

```
data(demo)
output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
mean_diff(output2)
```

percentESS *Extract percentage sample ESS*

Description

A generic function to extract percentage sample ESS for different object classes

Usage

```
percentESS(object)
```

Arguments

`object`	An objects of class 'balancing_weights' or 'causal_estimates'.

Value

Percentage sample effective sample size (ESS) for the pseudo-population.

Examples

```
data(demo)
output1 <- balancing.weights(S, Z, X, method = "IC", naturalGroupProp)
percentESS(output1)

output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
percentESS(output2)
```

plot.causal_estimates *Boxplot of percent ESS*

Description

Plot method for objects of class 'causal_estimates' to generate a boxplot of percent sample effective sample size (ESS) for a specific weighting method using bootstrap samples.

Usage

```
## S3 method for class 'causal_estimates'
plot(x, ...)
```

Arguments

`x` An object of class 'causal_estimates', the number of bootstrap samples `B` must be greater than 0.

`...` Additional arguments including:

y_limit The y-axis range. Default is `c(0,50)`.

color The boxplot color. Default is "red".

Value

A boxplot of percent sample ESS for a specific weighting method (`FLEXOR`, `IC`, or `IGO`)

Examples

```
data(demo)
set.seed(1)
output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
plot(output2)
```

print.balancing_weights

Print method for objects of class 'balancing_weights'

Description

Print method for objects of class 'balancing_weights'

Usage

```
## S3 method for class 'balancing_weights'
print(x, ...)
```

Arguments

`x` An object of class 'balancing_weights'

`...` Additional arguments affecting the printed results (so far no additional arguments are needed, so leave blank).

Value

Print values of the 'balancing_weights' object, including:

Weight length The total number of weights.

percentESS Percentage sample effective sample size (ESS) for the pseudo-population.

Examples

```
data(demo)
output1 <- balancing.weights(S, Z, X, method = "IC", naturalGroupProp)
print(output1)
```

---

print.causal_estimates *Print method for objects of class 'causal_estimates'*

Description

Print method for objects of class 'causal_estimates'

Usage

```
## S3 method for class 'causal_estimates'
print(x, ...)
```

Arguments

x An object of class 'causal_estimates'

... Additional arguments affecting the printed results (so far no additional arguments are needed, so leave blank).

Value

Print values of the 'causal_estimates' object, including:

Percentage sample ESS Percentage sample effective sample size (ESS) for the pseudo-population.

Mean differences The mean differences between two groups

Sigma ratios The ratios of standard deviations between two groups

Examples

```
data(demo)
output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
print(output2)
```

sigma_ratio *Extract sigma ratios*

Description

A generic function to extract the ratios of standard deviations for objects of class 'causal_estimates'.

Usage

```
sigma_ratio(object)
```

Arguments

`object`	An objects of class 'causal_estimates'.

Value

The ratios of standard deviations between two groups

Examples

```
data(demo)
output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
mean_diff(output2)
```

summary.balancing_weights *Summary method for objects of class 'balancing_weights'*

Description

Summary method for objects of class 'balancing_weights'

Usage

```
## S3 method for class 'balancing_weights'
summary(object, ...)
```

Arguments

`object`	An object of class 'balancing_weights'
`...`	Additional arguments affecting the summary produced (so far no additional arguments are needed, so leave blank).

Value

Printed summary of the ’balancing_weights’ object, including:

Weight length The total number of weights.

Weight distribution Statistical summary of weight values.

Percentage sample ESS Percentage sample effective sample size (ESS) for the pseudo-population.

Examples

```
data(demo)
output1 <- balancing.weights(S, Z, X, method = "IC", naturalGroupProp)
summary(output1)
```

---

summary.causal_estimates *Summary method for objects of class ’causal_estimates’*

Description

Summary method for objects of class ’causal_estimates’

Usage

```
## S3 method for class 'causal_estimates'
summary(object, ...)
```

Arguments

`object`	An object of class ’causal_estimates’
`...`	Additional arguments affecting the summary produced (so far no additional arguments are needed, so leave blank).

Value

Printed summary of the ’causal_estimates’ object, including:

Percentage sample ESS Percentage sample effective sample size (ESS) for the pseudo-population.

Mean differences with 95% CI (if `B > 0`) The mean differences between two groups with their corresponding 95% confidence intervals.

Sigma ratios with 95% CI (if `B > 0`) The ratios of standard deviations between two groups with their corresponding 95% confidence intervals.

Mean differences (if `B = 0`) The mean differences between two groups.

Sigma ratios (if `B = 0`) The ratios of standard deviations between two groups.

Examples

```
data(demo)
set.seed(1)
output2 <- causal.estimate(S, Z, X, Y, B = 5, method = "IC", naturalGroupProp)
summary(output2)
```

Index