Package: miapack (via r-universe)

May 27, 2026

Type Package

Title Marginalization over Incomplete Auxiliaries

Version 0.1.0

Maintainer Sean McGrath <sean.mcgrath514@gmail.com>

Description Implements methods to estimate conditional outcome means in settings with missingness-not-at-random and incomplete auxiliary variables. Specifically, this package implements the marginalization over incomplete auxiliaries (MIA) method proposed by Mathur et al. (2026) <doi:10.13140/RG.2.2.30750.19524>. The package supports continuous and binary outcomes, and supports auxiliary variables that are normal, binary, and categorical.

License GPL (>=3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

URL https://github.com/stmcg/miapack

BugReports https://github.com/stmcg/miapack/issues

Imports boot, nnet, progress

Depends R (>= 2.10)

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

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

Date/Publication 2026-02-26 22:14:45 UTC

RemoteUrl https://github.com/stmcg/miapack

RemoteRef HEAD

RemoteSha a0123c070c00967138887be34c67d905c558855d

Contents

`dat.sim` *Simulated data set*

Description

This data set was simulated to reflect a setting with missingness-not-at-random and an incomplete auxiliary variable.

Usage

```
dat.sim
```

Format

A data frame that contains 9,297 rows and the following columns:

`Y` A continuous outcome variable.

`X1` A binary predictor variable.

`X2` A binary predictor variable.

`W` A binary auxiliary variable.

Details

Variable dependencies: The underlying values of the variables were generated as follows:

- `X1` is generated independently.
- `X2` depends on `X1`.
- `W` depends on `X2`.
- `Y` depends on `X1`, `X2`, and their interaction.

Missingness patterns: The missingness patterns were generated as follows:

- Missingness in `X1`, `X2`, and `Y` depends on the underlying (potentially unobserved) values of `W`.
- Missingness in `W` is generated independently.
- Rows where all variables are missing are removed from the dataset.

See Also

`mia`

get_CI *Bootstrap-based confidence intervals for MIA*

Description

This function applies nonparametric bootstrap to construct confidence intervals around the conditional mean estimates obtained by mia. This function is a wrapper for the boot and boot.ci functions from the **boot** package.

Usage

```
get_CI(
  mia_res,
  n_boot = 1000,
  type = "bca",
  conf = 0.95,
  boot_args = list(),
  boot.ci_args = list(),
  show_progress = TRUE
)
```

Arguments

mia_res	Output from the mia function.
n_boot	Numeric scalar specifying the number of bootstrap replicates to use
type	Character string specifying the type of confidence interval. The options are "norm", "basic", "perc", and "bca".
conf	Numeric scalar specifying the level of the confidence interval. The default is 0.95.
boot_args	A list of additional arguments to pass to the boot function. Note that this includes parallelization options.
boot.ci_args	A list of additional arguments to pass to the boot.ci function
show_progress	Logical scalar indicating whether to show a progress bar during bootstrap. Default is TRUE. The progress bar will not be displayed when parallelization is used.

Value

An object of class "mia_ci". This object is a list with the following elements:

ci_1	An object of class "boot.ci" which contains the output of the boot.ci function applied for the confidence interval around the mean under X_values_1 in mia.
ci_2	An object of class "boot.ci" which contains the output of the boot.ci function applied for the confidence interval around the mean under X_values_2 in mia (if applicable).

ci_contrast	An object of class "boot.ci" which contains the output of the boot.ci function applied for the confidence interval around the contrast between mean under X_values_1 versus X_values_2 in mia (if applicable).
bres	An object of class "boot" which contains the output of the boot function. Users can access the bootstrap replicates through the element t in this object.
...	additional elements

Mathur MB, Seaman S, Zhang W, McGrath S, Shpitser I. (2026). *Estimating conditional means under missingness-not-at-random with incomplete auxiliary variables*. doi.org/10.13140/RG.2.2.30750.19524.

Examples

```
set.seed(1234)
res <- mia(data = dat.sim,
           X_names = c("X1", "X2"),
           X_values_1 = c(0, 1), X_values_2 = c(0, 0),
           Y_model = Y ~ W + X1 + X2, W_model = W ~ X1 + X2)
res_ci <- get_CI(mia_res = res, n_boot = 50, type = 'perc')
res_ci

## Example with parallelization
res_par <- get_CI(res, n_boot = 100, type = 'perc',
                  boot_args = list(parallel = "snow", ncpus = 2))
```

---

mia *MIA Method*

Description

This function implements the marginalization over incomplete auxiliaries (MIA) method (Mathur et al. 2026). For an outcome variable Y, predictor variable X, and auxiliary variable W, this function estimates the conditional outcome mean identified by

$$\mu_{\text{MIA}}(x) = \int_w E[Y|X = x, W = w, M = 1]p(w|X = x, R_W = R_X = 1)dw.$$

where R_W and R_X are indicators of non-missing values of W and X, respectively, and M is an indicator of a complete case pattern (i.e., Y, X, and W are non-missing). The function supports estimating the identifying functionals of $\mu_{\text{MIA}}(x_1)$ and $\mu_{\text{MIA}}(x_2)$ as well as contrasts between them (differences, ratios).

Usage

```
mia(
  data,
  X_names,
  X_values_1,
  X_values_2 = NULL,
  contrast_type,
  Y_model,
  Y_type,
  W_model,
  W_type,
  n_mc = 10000,
  return_simulated_data = FALSE
)
```

Arguments

`data`	Data frame containing the observed data.
`X_names`	Vector of character strings specifying the name(s) of the predictor variable(s) X.
`X_values_1`	Numeric vector specifying the value of the predictor variable(s) X, i.e. x_1 in $\mu_{\text{MIA}}(x_1)$.
`X_values_2`	(Optional) Numeric vector specifying an additional value of the predictor variable(s) X, i.e. x_2 in $\mu_{\text{MIA}}(x_2)$.
`contrast_type`	(Optional) Character string specifying the type of contrast to use when comparing $\mu_{\text{MIA}}(x_1)$ and $\mu_{\text{MIA}}(x_2)$. Options are `"difference"`, `"ratio"`, and `"none"`.
`Y_model`	Formula for the outcome model.
`Y_type`	(Optional) Character string specifying the "type" of the outcome variable. Options are `"binary"` and `"continuous"`. If this is not supplied, the type will be inferred from the corresponding column in `data`.
`W_model`	Formula for the auxiliary variable model. If the auxiliary variable is multivariate, this argument should be a list of model formulas, one for each component. The components will be simulated in the order they appear in the list.
`W_type`	(Optional) Vector of character strings specifying the "type" of each auxiliary variable. Options are `"binary"`, `"categorical"`, and `"normal"`. If this is not supplied, the type will be inferred from the corresponding column in `data`.
`n_mc`	Integer specifying the number of Monte Carlo samples to use.
`return_simulated_data`	Logical scalar indicating whether to return the simulated data set(s) containing the predictors and simulated auxiliary variable. Setting this argument to `TRUE` can substantially increase the size of the returned object, particularly when `n_mc` is large. The default is `FALSE`.

Details

Estimation algorithm:

Step 1: One fits a model for the conditional outcome mean $E[Y|X = x, W = w, M = 1]$ and the conditional density of the auxiliary variables $p(w|X = x, R_W = R_X = 1)$. When W is multivariate, i.e., $W = (W_1, \ldots, W_p)^\top$, one uses the decomposition

$$p(w|X = x, R_W = R_X = 1) = \prod_{j=1}^{p} p(w_j|X = x, w_1, \ldots, w_{j-1}, R_W = R_X = 1)$$

and fits models for the components $p(w_j|X = x, w_1, \ldots, w_{j-1}, R_W = R_X = 1)$.

Step 2: Monte Carlo integration is used to compute the integral in the identifying functional for $\mu_{\text{MIA}}(x)$ based on the fitted models in the first step. More specifically, for iteration i, the following algorithm is performed. The value of W is first simulated from its estimated conditional distribution. When W is multivariate, the components of W are simulated sequentially from their fitted models. That is, W_1 is simulated conditional on x, W_2 is simulated conditional on x, W_1, and so on. Then, the mean of Y is estimated conditional on x, W. Finally, the average of the estimated means (across all iterations i) is taken as the estimate of $\mu_{\text{MIA}}(x)$.

Value

An object of class "mia". This object is a list with the following elements:

`mean_est_1`	conditional outcome mean estimate under `X_values_1`
`mean_est_2`	conditional outcome mean estimate under `X_values_2`
`contrast_est`	contrast of conditional outcome mean estimates between `X_values_1` and `X_values_2`
`fit_W`	a list of fitted model(s) for W
`fit_Y`	fitted model for Y
`simulated_data`	a list, where the first element is the simulated data set under `X_values_1` and the second element is the simulated data set under `X_values_2`. The simulated data sets contain the predictors and simulated auxiliary variable. This element is set to `NULL` unless `return_simulated_data` is set to `TRUE`.
`...`	additional elements

References

Mathur MB, Seaman S, Zhang W, McGrath S, Shpitser I. (2026). *Estimating conditional means under missingness-not-at-random with incomplete auxiliary variables*. doi.org/10.13140/RG.2.2.30750.19524.

See Also

`print.mia`, `get_CI`

Examples

```
set.seed(1234)
mia(data = dat.sim,
    X_names = c("X1", "X2"),
    X_values_1 = c(0, 1), X_values_2 = c(0, 0),
    Y_model = Y ~ W + X1 + X2, W_model = W ~ X1 + X2)
```

`print.mia` *Print method for objects of class "mia"*

Description

Print method for objects of class "mia"

Usage

```
## S3 method for class 'mia'
print(x, digits = 4, ...)
```

Arguments

`x`	Object of class "mia".
`digits`	Integer specifying the number of decimal places to display.
`...`	Other arguments (ignored).

Value

No value is returned.

See Also

`mia`

Examples

```
res <- mia(data = dat.sim,
           X_names = c("X1", "X2"),
           X_values_1 = c(0, 1), X_values_2 = c(0, 0),
           Y_model = Y ~ W + X1 + X2, W_model = W ~ X1 + X2)
print(res)
```

print.mia_ci *Print method for objects of class "mia_ci"*

Description

Print method for objects of class "mia_ci"

Usage

```
## S3 method for class 'mia_ci'
print(x, digits = 4, ...)
```

Arguments

x	Object of class "mia_ci".
digits	Integer specifying the number of decimal places to display.
...	Other arguments (ignored).

Value

No value is returned.

See Also

get_CI

Examples

```
set.seed(1234)
res <- mia(data = dat.sim,
           X_names = c("X1", "X2"),
           X_values_1 = c(0, 1), X_values_2 = c(0, 0),
           Y_model = Y ~ W + X1 + X2, W_model = W ~ X1 + X2)
res_ci <- get_CI(res, n_boot = 100, type = 'perc')
print(res_ci)
```

Index