Package ‘HOIFCar’

February 2, 2026

Type Package

Title Covariate Adjustment in RCT by Higher-Order Influence Functions

Version 1.1.1

Maintainer Xinbo Wang `<cinbo_w@sjtu.edu.cn>`

Description

Estimates treatment effects using covariate adjustment methods in Randomized Clinical Trials (RCT) motivated by higher-order influence functions (HOIF). Provides point estimates, oracle bias, variance, and approximate variance for HOIF-adjusted estimators. For methodology details, see Zhao et al. (2024) <doi:10.48550/arXiv.2411.08491> and Gu et al. (2025) <doi:10.48550/arXiv.2512.20046>.

Suggests mvtnorm

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Imports brglm2, BB, parallel, doParallel, foreach, glmnet, stats, MASS

Author Sihui Zhao [aut],
Xinbo Wang [cre, aut],
Yujia Gu [aut],
Wei Ma [ctb],
Liu Liu [ctb]

Repository CRAN

Date/Publication 2026-02-02 14:40:19 UTC

Contents

`esti_mean_treat` *Estimate treatment effect and the corresponding variance estimation on the treatment arm using different covariate adjustment methods.*

Description

Implements a unified framework for comparing covariate adjustment method for completely randomized experiments under randomization-based framework.

Usage

```
esti_mean_treat(X, Y, A, H = NULL)
```

Arguments

`X`	The n by p covariates matrix.
`Y`	Vector of n dimensional observed response.
`A`	Vector of n dimensional treatment assignment.
`H`	The n by n hat projection matrix corresponding to X.

Value

A list with two named vectors:

point_est Point estimates for all estimators:

- `unadj`: Unadjusted estimator
- `db`: Debiased estimator (Lu et al., 2023)
- `adj2c`: HOIF-inspired debiased estimator (Zhao et al., 2024), the same as `db`
- `adj2`: HOIF-motivated adjusted estimator (Zhao et al., 2024)
- `adj3`: Bias-free adjusted estimator based on `adj2`
- `lin`: Covariate-adjusted estimator (Lin, 2013)
- `lin_db`: Debiased estimator with population leverage scores (Lei, 2020)

var_est Variance estimates corresponding to each estimator:

- `unadj`: Variance estimate for unadjusted estimator
- `db`: Variance estimate for debiased estimator (Lu et al., 2023)
- `adj2c`: Variance for `adj2c`, using formulas given in (Lu et al., 2023)
- `adj2c_v2`: Conservative variance for `adj2c` (Zhao et al., 2024)
- `adj2`: Variance for `adj2`, with formulas motivated by (Lu et al., 2023)
- `adj2_v2`: Conservative variance for `adj2` (Zhao et al., 2024)
- `adj3`: Variance for `adj3`, with formulas motivated by (Lu et al., 2023)
- `adj3_v2`: Conservative variance for `adj3` (Zhao et al., 2024)
- `lin`: HC3-type variance for Lin's (2013) estimator
- `lin_db`: HC3-type variance for Lei's (2020) estimator

References

Lin, W. (2013). *Agnostic notes on regression adjustments to experimental data: Reexamining Freedman's critique. The Annals of Statistics, Vol. 7(1), 295–318*, doi:10.1214/12AOAS583.
Lei, L. and Ding, P. (2020) *Regression adjustment in completely randomized experiments with a diverging number of covariates. Biometrika, Vol. 108(4), 815–828*, doi:10.1093/biomet/asaa103.
Lu, X., Yang, F. and Wang, Y. (2023) *Debiased regression adjustment in completely randomized experiments with moderately high-dimensional covariates. arXiv preprint, arXiv:2309.02073*, doi:10.48550/arXiv.2309.02073.
Zhao, S., Wang, X., Liu, L. and Zhang, X. (2024) *Covariate Adjustment in Randomized Experiments Motivated by Higher-Order Influence Functions. arXiv preprint, arXiv:2411.08491*, doi:10.48550/arXiv.2411.08491.

Examples

```
set.seed(100)
n <- 500
p <- n * 0.3
beta <- runif(p, -1 / sqrt(p), 1 / sqrt(p))

X <- mvtnorm::rmvt(n, sigma = diag(1, p), df = 3)
Y1 <- as.numeric(X %*% beta)
Y0 <- rep(0, n)

pi1 <- 2/3
n1 <- ceiling(n * pi1)
ind <- sample(n, size = n1)
A <- rep(0, n)
A[ind] <- 1
Y <- Y1 * A + Y0 * (1 - A)
Xc <- scale(X, scale = FALSE)
H <- Xc %*% MASS::ginv(t(Xc) %*% Xc) %*% t(Xc)

result_ls <- esti_mean_treat(X, Y, A, H)
point_est <- result_ls$point_est
var_est <- result_ls$var_est
print(paste0('True mean treat:', round(mean(Y1), digits = 3), '.'))
print('Absolute bias:')
print(abs(point_est - mean(Y1)))
print('Estimate variance:')
print(var_est)
```

`fit.adj2.adj2c.CAR` *Covariate-Adjusted Treatment Effect Estimation under Covariate-Adaptive randomization*

Description

Implements HOIF-inspired debiased estimators for average treatment effect (ATE) or treatment effect on the treatment/control arm with variance estimation using estimated asymptotic variance. Designed for randomized experiments with moderately high-dimensional covariates.

Usage

```
fit.adj2.adj2c.CAR(Y, X, S, A, intercept = TRUE, pi1 = NULL, target = "ATE")
```

Arguments

`Y`	Numeric vector of length n containing observed responses.
`X`	Numeric matrix (n x p) of covariates. Centering is required. May include intercept column.
`S`	Vector of length n denoting strata used in randomization procedure. Either a factor or an integer-valued numeric vector indexed from 1 to K.
`A`	Binary vector of length n indicating treatment assignment (1 = treatment, 0 = control).
`intercept`	Logical. If TRUE (default), X already contains intercept. Set FALSE if X does not contain intercept.
`pi1`	The assignment probability for the randomization assignment. If ‘NULL‘ (the default), the empirical assignment probability is used. Should be a vector with length K (Number of strata).
`target`	A character string specifying the target estimand. Must be one of: - ‘"ATE"‘ (default): Average Treatment Effect (difference between treatment and control arms). - ‘"EY1"‘: Expected outcome under treatment (estimates the effect for the treated group). - ‘"EY0"‘: Expected outcome under control (estimates the effect for the control group).

Value

A list containing two named vectors, including point estimates and variance estimates:

tau_vec Point estimates:

- `adj2`: Point estimation of the HOIF-inspired debiased estimator (Gu et al., 2025).
- `adj2c`: Same as `adj2`, but incorporating the centering step from Zhao et al. (2024) and Lu et al. (2023).

var_vec Variance estimates:

- `adj2`: Variance for `adj2` via the sample variance of its asymptotic variance formula.
- `adj2c`: Variance for `adj2c` via the sample variance of its asymptotic variance formula.

References

Gu, Y., Liu, L. and Ma, W. (2025) *Assumption-lean covariate adjustment under covariate adaptive randomization when* $p = o\,(n)$*. arXiv preprint, arXiv:2512.20046*, doi:10.48550/arXiv.2512.20046.

Lu, X., Yang, F. and Wang, Y. (2023) *Debiased regression adjustment in completely randomized experiments with moderately high-dimensional covariates. arXiv preprint, arXiv:2309.02073*, doi:10.48550/arXiv.2309.02073.

Zhao, S., Wang, X., Liu, L. and Zhang, X. (2024) *Covariate Adjustment in Randomized Experiments Motivated by Higher-Order Influence Functions. arXiv preprint, arXiv:2411.08491*, doi:10.48550/arXiv.2411.08491.

Examples

```
set.seed(120)
alpha0 <- 0.1;
n <- 400;
S <- as.factor(sample(c("0-30","31-50",">50"),n,replace = TRUE,prob=c(0.2,0.4,0.4)))
ns_min = min(table(S))

p0 <- ceiling(ns_min * alpha0)
beta0_full <- 1 / (1:p0) ^ (1 / 2) * (-1) ^ c(1:p0)
beta <- beta0_full / norm(beta0_full,type='2')

Sigma_true <- matrix(0, nrow = p0, ncol = p0)
for (i in 1:p0) {
  for (j in 1:p0) {
    Sigma_true[i, j] <- 0.1 ** (abs(i - j))
  }
}

X <- mvtnorm::rmvt(n, sigma = Sigma_true, df = 3)

lp0 <- X %*% beta
delta_X <- 1  -  1/4 * X[, 2] -  1/8 * X[, 3]
lp1 <- lp0 + delta_X

Y0 <- lp0 + rnorm(n)
Y1 <- lp1 + rnorm(n)


pi1 <- 1 / 2

# We use stratified block randomization as an example. Simple randomization
# is also valid by setting S = rep(1,n) and A = rbinom(n,1,pi1)

sbr <- function(S,nA,p,block_size=10){
  N <- length(S)
  B <- block_size
  A <- rep(0,N)
  nS <- length(unique(S))
  for(s in 1:nS){
    ind_s <- which(S==s)
    n_s <- length(ind_s)
    A_s <- rep(0,n_s)
    numB <- floor(n_s/B)
    rem <- n_s - numB*B
```

```
    size_A <- B*p[s]
    if(numB==0){
      size_rem = floor(rem*p[s])
      size_rem[1] = rem - sum(size_rem[-1])
      A_s[(B*numB+1):n_s] <- sample(rep(0:(nA-1),size_rem),size=rem,replace = FALSE)
    }else{
      for(i in 1:numB){
        A_s[(B*(i-1)+1):(B*i)] <- sample(rep(0:(nA-1),size_A),size=B,replace = FALSE)
      }
      if(rem>0){
        size_rem = floor(rem*p[s])
        size_rem[1] = rem - sum(size_rem[-1])
        A_s[(B*numB+1):n_s] <- sample(rep(0:(nA-1),size_rem),size=rem,replace = FALSE)
      }
    }
    A[ind_s] <- A_s
  }
  return(A)
}


A <- sbr(as.numeric(S),2,rep(pi1,3),block_size = 4)

Y <- A * Y1 + (1 - A) * Y0

Xc <- cbind(1, scale(X, scale = FALSE))
result.adj2.adj2c.car.ate.ls <- fit.adj2.adj2c.CAR(Y, Xc,S, A, intercept = TRUE,
                                                    target = 'ATE')
result.adj2.adj2c.car.ate.ls
result.adj2.adj2c.car.treat.ls <- fit.adj2.adj2c.CAR(Y, Xc,S, A, intercept = TRUE,
                                                      target = 'EY1')
result.adj2.adj2c.car.treat.ls
result.adj2.adj2c.car.control.ls <- fit.adj2.adj2c.CAR(Y, Xc,S, A, intercept = TRUE,
                                                        target = 'EY0')
result.adj2.adj2c.car.control.ls
```

`fit.adj2.adj2c.Random` *Covariate-Adjusted Treatment Effect Estimation under the Randomization-based Framework*

Description

Implements the (HOIF-inspired) debiased estimators for average treatment effect (ATE) or treatment effect on the treatment/control arm with variance estimation using asymptotic-variance. Designed for randomized experiments with moderately high-dimensional covariates.

Usage

```
fit.adj2.adj2c.Random(Y, X, A, pi1 = NULL, target = "ATE")
```

Arguments

`Y`	Numeric vector of length n containing observed responses.
`X`	Numeric matrix (n x p) of covariates. Centering is required. Intercept term can include or not.
`A`	Binary vector of length n indicating treatment assignment (1 = treatment, 0 = control).
`pi1`	Default is NULL. The assignment probability for the randomization assignment.
`target`	A character string specifying the target estimand. Must be one of: - ‘"ATE"‘ (default): Average Treatment Effect (difference between treatment and control arms). - ‘"EY1"‘: Expected outcome under treatment (estimates the effect for the treated group). - ‘"EY0"‘: Expected outcome under control (estimates the effect for the control group).

Value

A list containing three named vectors, including point estimates and variance estimates:

tau_vec Point estimates:

- `adj2`: Point estimation of the HOIF-inspired debiased estimator given by Zhao et al.(2024).
- `adj2c`: Point estimation of the debiased estimator given by Lu et al. (2023), which is also the HOIF-inspired debiased estimator given by Zhao et al.(2024).

var_vec_v1 Variance estimates for adj2 and adj2c, with formulas inspired by Lu et al. (2023).:

- `adj2`: Variance for `adj2`.
- `adj2c`: Variance for `adj2c`.

var_vec_v2 Variance estimates for adj2 and adj2c, with formulas given in Zhao et al. (2024), which is more conservative.

- `adj2`: Variance for `adj2`.
- `adj2c`: Variance for `adj2c`.

References

Lu, X., Yang, F. and Wang, Y. (2023) *Debiased regression adjustment in completely randomized experiments with moderately high-dimensional covariates. arXiv preprint, arXiv:2309.02073*, doi:10.48550/arXiv.2309.02073.
Zhao, S., Wang, X., Liu, L. and Zhang, X. (2024) *Covariate Adjustment in Randomized Experiments Motivated by Higher-Order Influence Functions. arXiv preprint, arXiv:2411.08491*, doi:10.48550/arXiv.2411.08491.

Examples

```
set.seed(100)
n <- 500
p <- n * 0.3
beta <- runif(p, -1 / sqrt(p), 1 / sqrt(p))

X <- mvtnorm::rmvt(n, sigma = diag(1, p), df = 3)
```

```
Y1 <- as.numeric(X %*% beta)
Y0 <- as.numeric(X %*% beta - 1)

pi1 <- 2/3
n1 <- ceiling(n * pi1)
ind <- sample(n, size = n1)
A <- rep(0, n)
A[ind] <- 1
Y <- Y1 * A + Y0 * (1 - A)

Xc <- cbind(1, scale(X, scale = FALSE))
result.adj2.adj2c.random.ate.ls <- fit.adj2.adj2c.Random(Y, Xc, A, target = 'ATE')
result.adj2.adj2c.random.ate.ls

result.adj2.adj2c.random.treat.ls <- fit.adj2.adj2c.Random(Y, Xc, A, target = 'EY1')
result.adj2.adj2c.random.treat.ls

result.adj2.adj2c.random.control.ls <- fit.adj2.adj2c.Random(Y, Xc, A, target = 'EY0')
result.adj2.adj2c.random.control.ls
```

`fit.adj2.adj2c.Super` *Covariate-Adjusted Treatment Effect Estimation under the Super-Population Framework*

Description

Implements HOIF-inspired debiased estimators for average treatment effect (ATE) or treatment effect on the treatment/control arm with variance estimation using influence function-based and asymptotic-variance. Designed for randomized experiments with moderately high-dimensional covariates.

Usage

```
fit.adj2.adj2c.Super(
  Y,
  X,
  A,
  intercept = TRUE,
  pi1 = NULL,
  target = "ATE",
  lc = FALSE
)
```

Arguments

`Y`	Numeric vector of length n containing observed responses.
`X`	Numeric matrix (n x p) of covariates. Centering is required. May include intercept column.

`A`	Binary vector of length n indicating treatment assignment (1 = treatment, 0 = control).
`intercept`	Logical. If TRUE (default), X already contains intercept. Set FALSE if X does not contain intercept.
`pi1`	The assignment probability for the randomization assignment. If ‘NULL‘ (the default), the empirical assignment probability is used.
`target`	A character string specifying the target estimand. Must be one of: - ‘"ATE"‘ (default): Average Treatment Effect (difference between treatment and control arms). - ‘"EY1"‘: Expected outcome under treatment (estimates the effect for the treated group). - ‘"EY0"‘: Expected outcome under control (estimates the effect for the control group).
`lc`	Default is FALSE. If TRUE, then performs linear calibration to achieve efficiency gain using $\hat{\mu}_0(X_i)$ and $\hat{\mu}_1(X_i)$.

Value

A list containing three named vectors, including point estimates and variance estimates:

tau_vec Point estimates:

- `adj2`: Point estimation of the HOIF-inspired debiased estimator (Zhao et al., 2024).
- `adj2c`: Point estimation of the the HOIF-inspired debiased estimator (Zhao et al., 2024), which is also the debiased estimator given by Lu et al. (2023).

var_infl_vec Influence function-based variance estimates:

- `adj2`: Variance for `adj2` via the sample variance of its influence function formula.
- `adj2c`: Variance for `adj2c` via the sample variance of its influence function formula.

var_rb_vec Variance estimates inspired by Bannick et al. (2025):

- `adj2`: Variance for `adj2` following the asymptotic variance given by Bannick et al. (2025).
- `adj2c`: Variance for `adj2c` following the asymptotic variance given by Bannick et al. (2025).

References

Bannick, M. S., Shao, J., Liu, J., Du, Y., Yi, Y. and Ye, T. (2025) *A General Form of Covariate Adjustment in Clinical Trials under Covariate-Adaptive Randomization. Biometrika, Vol. xx(x), 1-xx*, doi:10.1093/biomet/asaf029.
Lu, X., Yang, F. and Wang, Y. (2023) *Debiased regression adjustment in completely randomized experiments with moderately high-dimensional covariates. arXiv preprint, arXiv:2309.02073*, doi:10.48550/arXiv.2309.02073.
Zhao, S., Wang, X., Liu, L. and Zhang, X. (2024) *Covariate Adjustment in Randomized Experiments Motivated by Higher-Order Influence Functions. arXiv preprint, arXiv:2411.08491*, doi:10.48550/arXiv.2411.08491.

Examples

```
set.seed(120)
alpha0 <- 0.1;
n <- 400;

p0 <- ceiling(n * alpha0)
beta0_full <- 1 / (1:p0) ^ (1 / 2) * (-1) ^ c(1:p0)
beta <- beta0_full / norm(beta0_full,type='2')

Sigma_true <- matrix(0, nrow = p0, ncol = p0)
for (i in 1:p0) {
  for (j in 1:p0) {
    Sigma_true[i, j] <- 0.1 ** (abs(i - j))
  }
}

X <- mvtnorm::rmvt(n, sigma = Sigma_true, df = 3)

lp0 <- X %*% beta
delta_X <- 1  -  1/4 * X[, 2] -  1/8 * X[, 3]
lp1 <- lp0 + delta_X

Y0 <- lp0 + rnorm(n)
Y1 <- lp1 + rnorm(n)


pi1 <- 1 / 2
A <- rbinom(n, size = 1, prob = pi1)
Y <- A * Y1 + (1 - A) * Y0

Xc <- cbind(1, scale(X, scale = FALSE))
result.adj2.adj2c.sp.ate.ls <- fit.adj2.adj2c.Super(Y, Xc, A, intercept = TRUE,
                                                     target = 'ATE', lc = TRUE)
result.adj2.adj2c.sp.ate.ls
result.adj2.adj2c.sp.treat.ls <- fit.adj2.adj2c.Super(Y, Xc, A, intercept = TRUE,
                                                       target = 'EY1', lc = TRUE)
result.adj2.adj2c.sp.treat.ls
result.adj2.adj2c.sp.control.ls <- fit.adj2.adj2c.Super(Y, Xc, A, intercept = TRUE,
                                                         target = 'EY0', lc = TRUE)
result.adj2.adj2c.sp.control.ls
```

`fit.JASA` *Covariate Adjustment via JAckknife Score-based Adjustment (JASA) for Generalized Linear Working Models*

Description

Implements the Jackknife Score-Based Adjustment (JASA) method and its calibration for covariate adjustment in simple randomized experiments where covariate dimension p may be large relatively to sample size n. Handles Continuous, Binary, and Poisson outcomes.

Usage

```
fit.JASA(
  Y,
  X,
  A,
  family = "gaussian",
  pi1 = NULL,
  is.parallel = FALSE,
  core_num = 4,
  opt_obj = c("beta", "mu")[1]
)
```

Arguments

`Y`	Numeric vector of outcome values.
`X`	Matrix of centered baseline covariates (may include intercept). Dimensions: n x p.
`A`	Binary treatment vector (0 = control, 1 = treatment). Assignment is assumed to follow simple randomization.
`family`	GLM family specification: `"gaussian"`, `"binomial"`, or `"poisson"`. Default: `"gaussian"`.
`pi1`	The assignment probability for the simple randomization. If NULL (default), the empirical assignment probability is used.
`is.parallel`	Boolean for parallelization. Default: FALSE.
`core_num`	Number of cores for parallel computing (default is 4 if is.parallel = TRUE).
`opt_obj`	Ways to optimization: 'beta' (GLM coefficients) or 'mu' (expected outcomes). Default: 'beta'.

Value

List with components:

`tau_vec`	Named vector of average treatment effect estimates (`JASA`, `JASA-cal`)
`tau0_vec`	Treatment effect estimates on the control group
`tau1_vec`	Treatment effect estimates on the treatment group
`var_tau_vec`	Variance estimates for average treatment effect estimates
`var_tau0_vec`	Variance estimates for treatment effect estimates on the control group
`var_tau1_vec`	Variance estimates for treatment effect estimates on the treatment group
`y_hat_mat`	Matrix of predicted outcomes (columns: control, treatment) using varied Leave-One-Out strategy.
`obj_value_mat`	Matrix of objective values from optimization (columns: control, treatment)

Examples

```
generate_data_SR <- function(n, family, pi1, p_n_ratio = 0.05, seed = 123){
  set.seed(seed)
  alpha0 <- 0.15
  p0 <- ceiling(round(n * alpha0))
  beta0_full <- 1/(1:p0)^(1/4)*(-1)^c(1:p0)
  Sigma_true <- matrix(0, nrow = p0, ncol = p0)
  for (i in 1:p0) {
    for (j in 1:p0) {
      Sigma_true[i, j] <- 0.1 ** (abs(i - j))
    }
  }

  if(family != 'poisson'){
    X <- mvtnorm::rmvt(n, sigma = Sigma_true, df = 3)
  }else{
    X0 <- mvtnorm::rmvt(n, sigma = Sigma_true, df = 3)
    X <- pmin(pmax(X0, -3), 3)
    rm(X0)
  }

  beta <- beta0_full / norm(beta0_full,type='2')
  lp0 <- X %*% beta

  delta_X <- 1 - 1/2 * pmin(X[, 1]^2, 5) + 1/4 * X[, 1:10] %*% beta[1:10]
  lp1 <- lp0 + delta_X


  if (family == 'binomial') {
    r0 <- plogis(2 * lp0)
    r1 <- plogis(2 * lp1)
    Y1 <- rbinom(n, size=1, prob=r1)
    Y0 <- rbinom(n, size=1, prob=r0)
  }else if(family == 'poisson'){
    # quantile(lp1);quantile(lp0)
    lp1_tran <- pmin(lp1, 4)
    lp0_tran <- pmin(lp0, 4)
    r1 <- exp(lp1_tran)
    r0 <- exp(lp0_tran)

    Y1 <- rpois(n,r1)
    Y0 <- rpois(n,r0)
  }else if(family == 'gaussian'){
    r1 <- lp1;
    r0 <- lp0
    Y1 <- r1 + rnorm(n)
    Y0 <- r0 + rnorm(n)
  }

  A <- rbinom(n, size=1, prob=pi1)
  Y <- A * Y1 + (1 - A) * Y0
```

```
  p <- ceiling(round(n * p_n_ratio))
  if(p > ncol(X)){
    if(family != 'poisson'){
      X_noise <- rmvt(n, sigma = diag(p - ncol(X)), df = 3)
    }else{
      X0_noise <- rmvt(n, sigma = diag(p - ncol(X)), df = 3)
      X_noise <- pmin(pmax(X0_noise, -3), 3)
    }
    X_obs <- cbind(X, X_noise)
  }else{
    X_obs <- X[, 1:p, drop = FALSE]
  }

  data_ls <- list(
    X = X_obs, Y = Y, A = A,
    Y1 = Y1, Y0 = Y0,
    r1 = r1, r0 = r0
  )
  return(data_ls)
}


n <- 400; pi1 <- 1/3

family <- 'gaussian'; p_n_ratio <- 0.05
data_ls <- generate_data_SR(n, family, pi1, p_n_ratio)
X <- data_ls$X;
A <- data_ls$A
Y <- data_ls$Y

## Not run:
Xc <- scale(X, scale = FALSE)
Xc_aug <- cbind(1, Xc)
result.jasa.ls <- fit.JASA(Y, Xc_aug, A, family, opt_obj = 'beta')
result.jasa.ls$tau_vec
result.jasa.ls$var_tau_vec


family <- 'poisson'; p_n_ratio <- 0.05
data_ls <- generate_data_SR(n, family, pi1, p_n_ratio)
X <- data_ls$X;
A <- data_ls$A
Y <- data_ls$Y

Xc <- scale(X, scale = FALSE)
Xc_aug <- cbind(1, Xc)
result.jasa.ls <- fit.JASA(Y, Xc_aug, A, family, opt_obj = 'mu')
result.jasa.ls$tau_vec
result.jasa.ls$var_tau_vec


family <- 'binomial'; p_n_ratio <- 0.05
data_ls <- generate_data_SR(n, family, pi1, p_n_ratio)
```

```
X <- data_ls$X;
A <- data_ls$A
Y <- data_ls$Y

Xc <- scale(X, scale = FALSE)
Xc_aug <- cbind(1, Xc)
result.jasa.ls <- fit.JASA(Y, Xc_aug, A, family, opt_obj = 'beta')
result.jasa.ls$tau_vec
result.jasa.ls$var_tau_vec

## End(Not run)
```

`get_oracle_bias_var_adj2c` *Estimate the oracle bias, the exact variance and approximated variance of the debiased estimator tau_adj2c inspired by HOIF (Zhao et al.(2024)).*

Description

Estimate the oracle bias, the exact variance and approximated variance of the debiased estimator tau_adj2c inspired by HOIF (Zhao et al.(2024)).

Usage

```
get_oracle_bias_var_adj2c(X, Y1, n1 = NULL)
```

Arguments

`X`	The n by p covariates matrix.
`Y1`	Vector of n dimensional potential response Y(1).
`n1`	The number of subjects in the treatment group.

Value

A list of oracle bias and variance of the debised adjusted estimator tau_adj2c.

`bias_adj2c`	The oracle bias of the debiased estimator tau_adj2c.
`variance_exact_adj2c`	The oracle exact bias of the debiased estimator tau_adj2c.
`variance_approx_adj2c`	The oracle approximated variance of the debiased estimator tau_adj2c which omits the term of order o(1/n).
`variance_unadj`	The oracle variance of the unadjusted estimator.

References

Zhao, S., Wang, X., Liu, L., & Zhang, X. (2024). Covariate adjustment in randomized experiments motivated by higher-order influence functions. arXiv preprint. https://arxiv.org/abs/2411.08491.

Examples

```
NULL
```

`get_oracle_bias_var_adj_2_3` *Estimate the oracle bias, the exact variance and approximated variance of the debiased estimator and the bias-free estimator motivated by HOIF (Zhao et al.(2024)).*

Description

Estimate the oracle bias, the exact variance and approximated variance of the debiased estimator and the bias-free estimator motivated by HOIF (Zhao et al.(2024)).

Usage

```
get_oracle_bias_var_adj_2_3(X, Y1, n1 = NULL)
```

Arguments

`X`	The n by p covariates matrix.
`Y1`	Vector of n dimensional potential response Y(1).
`n1`	The number of subjects in the treatment group.

Value

A list of oracle bias and variance of the adjusted estimator motivated by HOIF and the bias-free estimator.

`bias_adj2`	The oracle bias of the estimator tau_adj2.
`variance_exact_adj2`	The oracle exact variance of the estimator tau_adj2.
`variance_approx_adj2`	The oracle approximated variance of the estimator tau_adj2 which omits the term of order o(1/n).
`variance_exact_adj3`	The oracle exact variance of the bias-free estimator tau_adj3.
`variance_unadj`	The oracle variance of the unadjusted estimator.

References

Zhao, S., Wang, X., Liu, L., & Zhang, X. (2024). Covariate adjustment in randomized experiments motivated by higher-order influence functions. arXiv preprint. https://arxiv.org/abs/2411.08491

Examples

```
# Linear setting
set.seed(100)
n <- 500
p <- 50
beta <- rt(p,3)

X <- mvtnorm::rmvt(n, sigma = diag(1, p), df = 3)
Y1 <- as.numeric(X %*% beta)
pi1 <- 0.50
n1 <- ceiling(n*pi1)

result_adj_db <- get_oracle_bias_var_adj_db(X = X,Y1=Y1,n1=n1)
result_adj2c <- get_oracle_bias_var_adj2c(X = X,Y1=Y1,n1=n1)
result_adj2_3 <- get_oracle_bias_var_adj_2_3(X = X,Y1=Y1,n1=n1)
unlist(result_adj_db)
unlist(result_adj2c)
unlist(result_adj2_3)



# Nonlinear setting
n <- 500;
alpha <- 0.2;
set.seed(1000)
p <- ceiling(n*alpha)
Sigma_true <- matrix(0,nrow=p,ncol=p)
for(i in 1:p){
  for(j in 1:p){
    Sigma_true[i,j] <- 0.1**(abs(i-j))
  }
}

X <- mvtnorm::rmvt(n, sigma = Sigma_true, df = 3)
beta <- rt(p,3)
or_baseline <- sign(X %*% beta) * abs(X %*% beta)^(1/2) + sin(X %*% beta)
epsilon1 <- epsilon0 <- rt(n,3)
Y1 <- 1 + as.numeric(or_baseline) + epsilon1


pi1 <- 0.50
n1 <- ceiling(n*pi1)

result_adj_db <- get_oracle_bias_var_adj_db(X = X,Y1=Y1,n1=n1) # from LYW paper
result_adj2c <- get_oracle_bias_var_adj2c(X = X,Y1=Y1,n1=n1)
result_adj2_3 <- get_oracle_bias_var_adj_2_3(X = X,Y1=Y1,n1=n1)
unlist(result_adj_db)
unlist(result_adj2c)
unlist(result_adj2_3)
```

`get_oracle_bias_var_adj_db`

Estimate the oracle bias, the oracle variance of the unadjusted estimator, the adjusted estimator by Lei's (2020) and the debiased estimator tau_db by Lu et al.(2023).

Description

Estimate the oracle bias, the oracle variance of the unadjusted estimator, the adjusted estimator by Lei's (2020) and the debiased estimator tau_db by Lu et al.(2023).

Usage

```
get_oracle_bias_var_adj_db(X, Y1, n1 = NULL)
```

Arguments

`X`	The n by p covariates matrix.
`Y1`	Vector of n dimensional potential response Y(1).
`n1`	The number of subjects in the treatment group.

Value

A list of the oracle bias and variance of .

`bias_adj`	The oracle bias of the adjusted estimator tau_adj we proposed.
`variance_unadj`	The oracle variance of the unadjusted estimator.
`variance_adj_lin`	The oracle variance of Lei's (2020) debiased estimator with linear working model.
`variance_db`	The oracle variance of the debiased estimator tau_db by Lu et al.(2023).

References

Lihua Lei, Peng Ding. Regression adjustment in completely randomized experiments with a diverging number of covariates. Biometrika, 815–828, 2020.

Xin Lu, Fan Yang, and Yuhao Wang. Debiased regression adjustment in completely randomized experiments with moderately high-dimensional covariates. arXiv preprint arXiv:2309.02073, 2023.

Examples

```
NULL
```

Index