Package 'CHEMIST'

July 21, 2025

Type Package

Title Causal Inference with High-Dimensional Error-Prone Covariates and Misclassified Treatments

Version 0.1.5

Maintainer Wei-Hsin Hsu <anson60214@gmail.com>

Depends R (>= 3.3.1),MASS

Imports stats,XICOR,LaplacesDemon

Description We aim to deal with the average treatment effect (ATE), where the data are subject to high-dimensionality and measurement error. This package primarily contains two functions, which are used to generate artificial data and estimate ATE with high-dimensional and error-prone data accommodated.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Author Wei-Hsin Hsu [aut, cre], Li-Pang Chen [aut]

Repository CRAN

Date/Publication 2023-05-01 14:15:38 UTC

Contents

CHEMIST_package *Causal Inference with High-Dimensional Error-Prone Covariates and Misclassified Treatments*

Description

The package CHEMIST, referred to Causal inference with High-dimsensional Error- prone Covariates and MISclassified Treatments, aims to deal with the average treatment effect (ATE), where the data are subject to high-dimensionality and measurement error. This package primarily contains two functions: one is Data_Gen that is applied to generate artificial data, including potential outcomes, error-prone treatments and covariates, and the other is FATE that is used to estimate ATE with measurement error correction.

Usage

```
CHEMIST_package()
```

Details

This package aims to estimate ATE in the presence of high-dimensional and error-prone data. The strategy is to do variable selection by feature screening and general outcome-adaptive lasso. After that, measurement error in covariates are corrected. Finally, with informative and error corrected data obtained, the propensity score can be estimated and can be used to estimate ATE by the inverse probability weight approach.

Value

CHEMIST_package

Data_Gen *Generation of Artificial Data*

Description

This function shows the demonstration of data generation based on some specific and commonly used settings, including exponential family distributed potential outcomes, error-prone treatments, and covariates. In this function, users can specify different magnitudes of measurement error and relationship between outcome, treatment, and covariates.

Usage

```
Data_Gen(
  X,
  alpha,
  beta,
  theta,
  a,
  sigma_e,
  e_distr = "normal",
  num_pi,
  delta,
  linearY,
  typeY
)
```

Arguments

X	The input of n x p dimensional matrix of true covariates, where n is sample size and p is number of covariates. Users can customize the data structure and distribution.
alpha	A vector of the parameters that reflects the relationship between treatment model and covariates. The dimension of alpha should be equal to the dimension of beta. If alpha and beta have the same nonzero components, then we call them Xc (covariates associated with both outcome and treatment). If components in alpha are zero but the same components in beta are nonzero, we call them Xp (covariates associated with outcome only), If components in alpha are nonzero but the same components in beta are zero, we call them Xi (covariates associated with treatment only). For example, if alpha = c(2,2,0,0,1,1) and beta = c(3,3,1,1,0,0), then the first two components are Xc, the middle two components are Xp, and the last two components are Xi.
beta	A vector of the parameters that reflects the relationship between outcome and covariates. The dimension of alpha should be equal to the dimension of beta. If alpha and beta have the same nonzero components, then we call them Xc (covariates associated with both outcome and treatment). If components in alpha are zero but the same components in beta are nonzero, we call them Xp (covariates associated with outcome only), If components in alpha are nonzero but the same components in beta are zero, we call them Xi (covariates associated with treatment only). For example, if alpha = c(2,2,0,0,1,1) and beta = c(3,3,1,1,0,0), then the first two components are Xc, the middle two components are Xp, and the last two components are Xi.
theta	The scalar of the parameter used to link outcome and treatment.
a	A weight of cov_e in the measurement error model W = cov_e*a + X + e, where W is observed covariates with measurement error, X is actual covariates, and e is noise term with covaraince matrix cov_e.
sigma_e	sigma_e is the common diagonal entries of covariance matrix in the measurement error model.

`e_distr` Distribution of the noise term in the classical measurement error model. The input "normal" refers to the normal distribution with mean zero and covariance matrix with diagonal entries `sigma_e`. The scalar input "v" represents t-distribution with degree of freedom v.

`num_pi` Settings of misclassification probability with option 1 or 2. `num_pi = 1` gives that pi_01 equals pi_10, and `num_pi = 2` refers to that pi_01 is not equal to pi_10.

`delta` The parameter that determines number of treatment with measurement error. `delta = 1` has equal number of treatment with and without measurement error. We set `default = 0.5` since it has smaller number of treatment who has measurement error.

`linearY` The boolean option that determines the relationship between outcome and covariates. `linearY = TRUE` gives linear relationship with a vector of parameters `alpha`, `linearY = FALSE` refers to non linear relationship between outcome and covariates, where the sin function is specified on Xc and the exponential function is specified on Xp.

`typeY` The outcome variable with exponential family distribution "binary", "pois" and "cont". `typeY = "binary"` refers to binary random variables, `typeY = "pois"` refers to Poisson random variables, and `typeY = "cont"` refers to normally distributed random variables.

Value

`Data` A n x (p+2) matrix of the original data without measurement error, where n is sample size and the first p columns are covariates with the order being Xc (the covariates associated with both treatment and outcome), Xp (the covariates associated with outcome only), Xi (the covariates associated with treatment only), Xs (the covariates independent of outcome and treatment), the last second column is treatment, and the last column is outcome.

`Error_Data` A n x (p+2) matrix of the data with measurement error in covariates and treatment, where n is sample size and the first p columns are covariates with the order being Xc (the covariates associated with both treatment and outcome), Xp (the covariates associated with outcome only), Xi (the covariates associated with treatment only), Xs (the covariates independent of outcome and treatment), the last second column is treatment, and the last column is outcome.

`Pi` A n x 2 matrix containing two misclassification probabilities pi_10 = P(Observed Treatment = 1 | Actual Treatment = 0) and pi_01 = P(Observed Treatment = 0 | Actual Treatment = 1) in columns.

`cov_e` A covariance matrix of the measurement error model.

Examples

```
##### Example 1: A multivariate normal continuous X with linear normal Y #####

## Generate a multivariate normal X matrix
mean_x = 0; sig_x = 1; rho = 0
Sigma_x = matrix( rho*sig_x^2,nrow=120 ,ncol=120 )
diag(Sigma_x) = sig_x^2
```

```
Mean_x = rep( mean_x, 120 )
X = as.matrix( mvrnorm(n = 60,mu = Mean_x,Sigma = Sigma_x,empirical = FALSE) )

## Data generation setting
## alpha: Xc's scale is 0.2 0.2 and Xi's scale is 0.3 0.3
## so this refers that there is 2 Xc and Xi
## beta: Xc's scale is 2 2 and Xp's scale is 2 2
## so this refers that there is 2 Xc and Xp
## rest with following setup
Data_fun <- Data_Gen(X, alpha = c(0.2,0.2,0,0,0.3,0.3), beta = c(2,2,2,2,0,0)
, theta = 2, a = 2, sigma_e = 0.75, e_distr = 10, num_pi = 1, delta = 0.8,
linearY = TRUE, typeY = "cont")

##### Example 2: A uniform X with non linear binary Y #####

## Generate a uniform X matrix
n = 50; p = 120
X = matrix(NA,n,p)
for( i in 1:p ){ X[,i] = sample(runif(n,-1,1),n,replace=TRUE ) }
X = scale(X)

## Data generation setting
## alpha: Xc's scale is 0.1 and Xi's scale is 0.3
## so this refers that there is 1 Xc and Xi
## beta: Xc's scale is 2 and Xp's scale is 3
## so this refers that there is 1 Xc and Xp
## rest with following setup
Data_fun <- Data_Gen(X, alpha = c(0.1,0,0.3), beta = c(2,3,0)
, theta = 1, a = 2, sigma_e = 0.5, e_distr = "normal", num_pi = 2, delta = 0.5,
linearY = FALSE, typeY = "binary")
```

FATE *Estimation of ATE under high-dimensional error-prone data*

Description

This function aims to estimate ATE by selecting informative covariates and correcting for measurement error in covariates and misclassification in treatments. The function FATE reflects the strategy of estimation method: Feature screening, Adaptive lasso, Treatment adjustment, and Error correction for covariates.

Usage

```
FATE(Data, cov_e, Consider_D, pi_10, pi_01)
```

Arguments

Data	A n x (p+2) matrix of the data, where n is sample size and the first p columns are covariates with the order being Xc (the covariates associated with both treatment

and outcome), Xp (the covariates associated with outcome only), Xi (the covariates associated with treatment only), Xs (the covariates independent of outcome and treatment), the last second column is treatment, and the last column is outcome.

`cov_e` Covariance matrix in the measurement error model.

`Consider_D` Feature screening with treatment effects accommodated. `Conidser_D = TRUE` refers to feature screening with A and (1-A) incorporated. `Consider_D = FALSE` will not multiply with A and (1-A).

`pi_10` Misclassifcation probability is P(Observed Treatment = 1 | Actual Treatment = 0).

`pi_01` Misclassifcation probability is P(Observed Treatment = 0 | Actual Treatment = 1).

Value

`ATE` A value of the average treatment effect.

`wAMD` A weighted absolute mean difference.

`Coef_prop_score` A table containing coefficients of propensity score.

`Kersye_table` The selected covariates by feature screening.

`Corr_trt_table` A summarized table containing corrected treatment.

Examples

```
##### Example 1: Input the data without measurement correction #####

## Generate a multivariate normal X matrix
mean_x = 0; sig_x = 1; rho = 0; n = 50; p = 120
Sigma_x = matrix( rho*sig_x^2 ,nrow=p ,ncol=p )
diag(Sigma_x) = sig_x^2
Mean_x = rep( mean_x, p )
X = as.matrix( mvrnorm(n ,mu = Mean_x,Sigma = Sigma_x,empirical = FALSE) )

## Data generation setting
## alpha: Xc's scale is 0.2 0.2 and Xi's scale is 0.3 0.3
## so this refers that there is 2 Xc and Xi
## beta: Xc's scale is 2 2 and Xp's scale is 2 2
## so this refers that there is 2 Xc and Xp
## rest with following setup
Data_fun <- Data_Gen(X, alpha = c(0.2,0.2,0,0,0.3,0.3), beta = c(2,2,2,2,0,0)
, theta = 2, a = 2, sigma_e = 0.75, e_distr = 10, num_pi = 1, delta = 0.8,
linearY = TRUE, typeY = "cont")

## Extract Ori_Data, Error_Data, Pi matrix, and cov_e matrix
Ori_Data=Data_fun$Data
Pi=Data_fun$Pi
cov_e=Data_fun$cov_e
Data=Data_fun$Error_Data
pi_01 = pi_10 = Pi[,1]
```

```
## Input data into model without error correction
Model_fix = FATE(Data, matrix(0,p,p), Consider_D = FALSE, 0, 0)

##### Example 2: Input the data with measurement correction #####

## Input data into model with error correction
Model_fix = FATE(Data, cov_e, Consider_D = FALSE, Pi[,1],Pi[,2])
```

Index