Package ‘KoulMde’

April 11, 2020

Title Koul's Minimum Distance Estimation in Linear Regression and Autoregression Model by Coordinate Descent Algorithm

Version 3.1.1

Author Jiwoong Kim <jwboys26 at gmail.com>

Maintainer Jiwoong Kim <jwboys26@gmail.com>

Description Consider linear regression model and autoregressive model of order q where errors in the linear regression model and innovations in the autoregression model are independent and symmetrically distributed. Hira L. Koul (1986) <DOI:10.1214/aos/1176350059> proposed a nonparametric minimum distance estimation method by minimizing L2-type distance between certain weighted residual empirical processes. He also proposed a simpler version of the loss function by using symmetry of the integrating measure in the distance. Kim (2018) <DOI:10.1080/00949655.2017.1392527> proposed a fast computational method which enables practitioners to compute the minimum distance estimator of the vector of general multiple regression parameters for several integrating measures. This package contains three functions: KoulLrMde(), KoulArMde(), and Koul2StageMde(). The former two provide minimum distance estimators for linear regression model and autoregression model, respectively, where both are based on Koul's method. These two functions take much less time for the computation than those based on parametric minimum distance estimation methods. Koul2StageMde() provides estimators for regression and autoregressive coefficients of linear regression model with autoregressive errors through minimum distant method of two stages. The new version is written in Rcpp and dramatically reduces computational time.

Depends R (>= 3.2.2)

License GPL-2

LazyData TRUE

Imports Rcpp (>= 0.12.7), expm

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.0.2

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-04-11 07:40:02 UTC
Description

Check whether or not an input matrix includes any non-numeric values (NA, NULL, "", character, etc) before being used for training. If any non-numeric values exist, then TrainBuddle() or FetchBuddle() will return non-numeric results.

Usage

CheckNonNumeric(X)

Arguments

X an n-by-p matrix.

Value

A list of (n+1) values where n is the number of non-numeric values. The first element of the list is n, and all other elements are entries of X where non-numeric values occur. For example, when the (1,1)th and the (2,3)th entries of a 5-by-5 matrix X are non-numeric, then the list returned by CheckNonNumeric() will contain 2, (1,1), and (2,3).

Examples

```r
n = 5;
p = 5;
X = matrix(0, n, p) # Generate a 5-by-5 matrix which includes two NA's.
X[1,1] = NA
X[2,3] = NA
lst = CheckNonNumeric(X)

lst
```
Koul2StageMde Two-stage minimum distance estimation in linear regression model with autoregressive error.

Description

Estimates both regression and autoregressive coefficients in the model \( Y = X\beta + \epsilon \) where \( \epsilon \) is autoregressive process of known order \( q \).

Usage

Koul2StageMde( Y, X, D, b0, RegIntMeasure, AR_Order, ArIntMeasure, TuningConst = 1.345 )

Arguments

- **Y**: Vector of response variables in linear regression model.
- **X**: Design matrix of explanatory variables in linear regression model.
- **D**: Weight Matrix. Dimension of D should match that of X. Default value is \( XA \) where \( A= (X'X)^{-1/2} \).
- **b0**: Initial value for beta.
- **RegIntMeasure**: Symmetric and \( \sigma \)-finite measure used for estimating \( \beta \): Lebesgue, Degenerate or Robust.
- **AR_Order**: Order of the autoregressive error.
- **ArIntMeasure**: Symmetric and \( \sigma \)-finite measure used for estimating autoregressive coefficients of the error: Lebesgue, Degenerate or Robust.
- **TuningConst**: Used only for Robust measure.

Value

MDE1stage - The list of the first stage minimum distance estimation result. It contains betahat1stage, residual1stage, and rho1stage.

- **betahat1stage**: The first stage minimum distance estimators of regression coefficients.
- **residual1stage**: Residuals after the first stage minimum distance estimation.
- **rho1stage**: The first stage minimum distance estimators of autoregressive coefficients of the error.
Koul2StageMde

MDE2stage - The list of the second stage minimum distance estimation result. It contains beta-hat2stage, residual2stage, and rho2stage.

- betahat2stage - The second stage minimum distance estimators of regression coefficients.
- residual2stage - Residuals after the second stage minimum distance estimation.
- rho2stage - The second stage minimum distance estimators of autoregressive coefficients of the error.

References


See Also

KoulArMde() and KoulLrMde()

Examples

```
#---------------------------
n <- 10
p <- 3
X <- matrix(runif(n*p, 0,50), nrow=n, ncol=p)  # Generate n-by-p design matrix X
beta <- c(-2, 0.3, 1.5)  # Generate true beta = (-2, 0.3, 1.5)
rho <- 0.4  # True rho = 0.4
eps <- vector(length=n)  # Generate innovation from N(0,1)
xi <- rnorm(n, 0,1)  # Generate autoregressive process of order 1
for(i in 1:n){
  if(i==1){eps[i] <- xi[i]}
  else{eps[i] <- rho*eps[i-1] + xi[i]}
}
Y <- X%*%beta + eps  # Use the default weight matrix
b0 <- solve(t(X)%*%X)%*%(t(X)%*%Y)  # Set initial value for beta

IntMeasure <- "Lebesgue"  # Define Lebesgue measure
MDEResult <- Koul2StageMde(Y,X, "default", b0, IntMeasure, IntMeasure, TuningConst = 1.345)
MDE1stageResult <- MDEResult[[1]]
MDE2stageResult <- MDEResult[[2]]

beta1 <- MDE1stageResult$betahat1stage
residual1 <- MDE1stageResult$residual1stage
```
**KoulArMde**

rho1 <- MDE1stageResult$rhohat1stage
beta2 <- MDE2stageResult$betahat2stage
residual2 <- MDE1stageResult$residual2stage
rho2 <- MDE2stageResult$rhohat2stage

---

**Description**

Estimates the autoressive coefficients in the \( X_t = \rho' Z_t + \xi_t \) where \( Z_t \) is the vector of \( q \) observations at times \( t - 1, \ldots, t - q \).

**Usage**

KoulArMde(X, AR_Order, IntMeasure, TuningConst = 1.345)

**Arguments**

- \( X \) - Vector of \( n \) observed values.
- \( AR\_Order \) - Order of the autoregression model.
- \( Int\_Measure \) - Symmetric and \( \sigma \)-finite measure: Lebesgue, Degenerate, and Robust
- \( Tuning\_Const \) - Used only for Robust measure.

**Value**

- \( \text{rhohat} \) - Minimum distance estimator of \( \rho \).
- \( \text{residual} \) - Residuals after minimum distance estimation.
- \( \text{ObjVal} \) - Value of the objective function at minimum distance estimator.

**References**


**See Also**

KoulLrMde() and Koul2StageMde()
Examples

```r
##### Generate stationary AR(2) process with 10 observations
n <- 10
q <- 2
rho <- c(-0.2, 0.8)  ##### Generate true parameters rho = (-0.2, 0.8)
eps <- rnorm(n, 0,1)  ##### Generate innovations from N(0,1)
X <- rep(0, times=n)
for (i in 1:n){
  tempCol <- rep(0, times=q)
  for (j in 1:q){
    if(i-j<=0){
      tempCol[j] <- 0
    }else{
      tempCol[j] <- X[i-j]
    }
  }
  X[i] <- t(tempCol)%*% rho + eps[i]
}

IntMeasure <- "Lebesgue"  ##### Define Lebesgue measure
MDEResult <- KoulArMde(X, q, IntMeasure, TuningConst=1.345)
rhohat <- MDEResult$rhohat  ##### Obtain minimum distance estimator
resid <- MDEResult$residual  ##### Obtain residual
objVal <- MDEResult$ObjVal  ##### Obtain the value of the objective function

IntMeasure <- "Degenerate"  ##### Define degenerate measure at 0
MDEResult <- KoulArMde(X, q, IntMeasure, TuningConst=1.345)
rhohat <- MDEResult$rhohat  ##### Obtain minimum distance estimator
resid <- MDEResult$residual  ##### Obtain residual
objVal <- MDEResult$ObjVal  ##### Obtain the value of the objective function

IntMeasure <- "Robust"  ##### Define "Robust" measure at 0
TuningConst <- 3  ##### Define the tuning constant
MDEResult <- KoulArMde(X, q, IntMeasure, TuningConst)
resid <- MDEResult$residual  ##### Obtain residual
objVal <- MDEResult$ObjVal  ##### Obtain the value of the objective function
```

Description

Estimates the regression coefficients in the model \( Y = X\beta + \epsilon \).
KoulLMde

Usage

KoulLMde(Y, X, D, b0, IntMeasure, TuningConst = 1.345)

Arguments

Y - Vector of response variables in linear regression model.
X - Design matrix of explanatory variables in linear regression model.
D - Weight Matrix. Dimension of D should match that of X. Default value is XA where A=(X'X)^(-1/2).
b0 - Initial value for beta.
IntMeasure - Symmetric and \( \sigma \)-finite measure: Lebesgue, Degenerate, and Robust
TuningConst - Used only for Robust measure.

Value

betahat - Minimum distance estimator of \( \beta \).
residual - Residuals after minimum distance estimation.
ObjVal - Value of the objective function at minimum distance estimator.

References


See Also

KoulArMde() and Koul2StageMde()

Examples

#######
n <- 10
p <- 3
X <- matrix(runif(n*p, 0,50), nrow=n, ncol=p) ### Generate n-by-p design matrix X
beta <- c(-2, 0.3, 1.5) ### Generate true beta = (-2, 0.3, 1.5)'
eps <- rnorm(n, 0.1) ### Generate errors from N(0,1)
Y <- X%*%beta + eps
D <- "default"  # Use the default weight matrix
b0 <- solve(t(X)%*%X)%*%(t(X)%*%Y)  # Set initial value for beta
IntMeasure <- "Lebesgue"  # Define Lebesgue measure

MDEResult <- KoulLrMde(Y,X,D, b0, IntMeasure, TuningConst=1.345)

betahat <- MDEResult$betahat  # Obtain minimum distance estimator
resid <- MDEResult$residual  # Obtain residual
objVal <- MDEResult$ObjVal  # Obtain the value of the objective function

IntMeasure <- "Degenerate"  # Define degenerate measure at 0

MDEResult <- KoulLrMde(Y,X,D, b0, IntMeasure, TuningConst=1.345)

betahat <- MDEResult$betahat  # Obtain minimum distance estimator
resid <- MDEResult$residual  # Obtain residual
objVal <- MDEResult$ObjVal  # Obtain the value of the objective function

IntMeasure <- "Robust"  # Define "Robust" measure
TuningConst <- 3  # Define the tuning constant

MDEResult <- KoulLrMde(Y,X,D, b0, IntMeasure, TuningConst)

betahat <- MDEResult$betahat  # Obtain minimum distance estimator
resid <- MDEResult$residual  # Obtain residual
objVal <- MDEResult$ObjVal  # Obtain the value of the objective function
Index

CheckNonNumeric, 2
Koul2StageMde, 3
KoulArMde, 5
KoulLrMde, 6