Package 'Disequilibrium'

October 12, 2022

Type Package

Title Disequilibrium Models

Version 1.1

Maintainer Nate Latshaw <dcms2015@gmail.com>

Depends R (>= 3.5.0)

Description Estimate, summarize, and perform predictions with the market in disequilibrium model, as found in Gourieroux, C. (2000) <doi:10.1017/CBO9780511805608> and Maddala, G. (1983) <doi:10.1017/CBO9780511810176>. The parameters are estimated with maximum likelihood.

License GPL-3

Encoding UTF-8

Imports optimr, Formula, numDeriv

Suggests MASS, knitr, rmarkdown, sandwich

LazyData true

RoxygenNote 7.0.2

VignetteBuilder knitr

NeedsCompilation no

Author Nate Latshaw [aut, cre],
Michael Guggisberg [aut],
Institute for Defense Analyses [cph]

Repository CRAN

Date/Publication 2020-05-31 04:10:02 UTC

R topics documented:

DE *Market in Disequilibrium Model*

Description

DE estimates a market in disequilibrium model.

The market in disequilibrium model is defined as follows. Let i denote the ith observation which takes values from 1 to N, X_1 be a covariate matrix of dimension $N \times k_1$, X_2 be a covariate matrix of dimension $N \times k_2$, X_{1i} be the ith row of X_1, X_{2i} be the ith row of X_2, β_1 be a coefficient vector of length k_1 and β_2 be a coefficient vector of length k_2. Define the latent response for stage one to be

$$y_{1i}^{\star} = X_{1i}\beta_1 + \epsilon_{1i}$$

and stage two to be

$$y_{2i}^{\star} = X_{2i}\beta_2 + \epsilon_{2i}.$$

Define the observed outcome to be $y_i = min(y_{1i}^{\star}, y_{2i}^{\star})$. The pair $(\epsilon_{1i}, \epsilon_{2i})$ is distributed independently and identically multivariate normal with means $E[\epsilon_{1i}] = E[\epsilon_{2i}] = 0$, variances $Var[\epsilon_{1i}] = \sigma_{11}$, $Var[\epsilon_{2i}] = \sigma_{22}$, and covariance $Cov(\epsilon_{1i}, \epsilon_{2i}) = \sigma_{12}$.

The model is estimated by (frequentist) maximum likelihood. The default maximum likelihood algorithm is based off the Limited-memory Broyden Fletcher Goldfarb Shanno (L-BFGS) algorithm. See optimr for details.

Usage

```
DE(formula, data, subset = NULL, par = NULL, control = list())
```

Arguments

formula	An object of class Formula: a symbolic description of the model to be fitted. The details of model specification are given under 'Details'.
data	A required data frame containing the variables provided in formula.
subset	An optional numeric vector specifying a subset of observations to be used in the fitting process.
par	A vector of initial values for the parameters for which optimal values are to be found. The order of the parameters is the coefficients of equation 1, the coefficients of equation 2, the variance for equation 1, the covariance of equations 1 and 2, and the variance of equation 2. The default is 0 for all parameters, except the variance parameters, which are set to 1.
control	A list of control parameters. See 'Details'.

Details

Models for DE are specified symbolically. A typical model has the form response ~ terms1 | terms2 where response is the name of the numeric response variable and terms1 and terms2 are each a series of variables that specifies the linear predictor(s) of the respective model. For example, if the first equation has two independent variables, X1 and X2, then terms1 = X1 + X2.

A formula has an implied intercept term for both equations. To remove the intercept from equation 1 use either response ~ terms1 - 1 | terms2 or response ~ 0 + terms1 | terms2. The intercept may be removed from equation 2 analgously.

The control argument is a list that can supply any of the following components:

method A method to be used in the function optimr. The default is "L-BFGS-B". See optimr for further details.

lower, upper Bounds on the variables for methods such as "L-BFGS-B" that can handle box (or bounds) constraints. The default is -Inf and Inf, respectively.

hessian A logical control that if TRUE forces the computation of an approximation to the Hessian at the final set of parameters. See optimr for further details. The default is FALSE.

na.action A function indicating what happens when data contains NAs. The default is na.omit. The only other possible value is na.fail.

transformR3toPD A logical. If TRUE, the covariance matrix will be manually converted to a positive definite matrix for optimization. The default is TRUE.

Equation1Name A string name for the first equation. The default is "_1".

Equation2Name A string name for the second equation. The default is "_2".

MaskRho A logical or numeric to determine if the correlation is masked. A value of FALSE means the correlation is not fixed. A value between -1 and 1 will fix the correlation to that value. The default is FALSE. A free correlation parameter can be numerically unstable, use with caution.

Value

An object of class 'DE' is returned as a list with the following components:

par The set of parameter maximum likelihood estimates.

value The negative of the log likelihood evaluated at `par`.

counts A two-element integer vector giving the number of calls to `fn` and `gr` respectively. This excludes those calls needed to compute the Hessian, if requested, and any calls to `fn` to compute a finite-difference approximation to the gradient.

convergence An integer code. '0' indicates successful completion.

message A character string giving any additional information returned by the optimizer, or `NULL`.

Sigma The estimated covariance matrix. This is a subset of `par`.

Xbeta1 The predicted outcome for each observation in equation 1 using the parameters estimated in `par`.

Xbeta2 The predicted outcome for each observation in equation 2 using the parameters estimated in `par`.

Y A vector of the response (quantity) values.

X A list of the design matricies for the two equations.

hessian A numerical approximation to the Hessian matrix of the likelihood function at the estimated parameter values. The Hessian is returned even if it is not requested.

vcov The variance covariance matrix of the estimates in `par`.

MaskRho The value of `MaskRho` used.

The object of class 'DE' has the following attributes:

originalNamesX1 The original names of the covariates for equation 1 specified in `formula`.

originalNamesX2 The original names of the covariates for equation 2 specified in `formula`.

namesX1 The names of the covariates for equation 1 to be used in the `summary.DE` function. This is equivalent to `paste0(originalNamesX1, Equation1Name)`.

namesX2 The names of the covariates for equation 2 to be used in the `summary.DE` function. This is equivalent to `paste0(originalNamesX2, Equation2Name)`.

namesSigma The names of the variance-covariance matrix parameters to be used in the `summary.DE` function.

Equation1Name The user-specified name of equation 1.

Equation2Name The user-specified name of equation 2.

betaN1 The number of coefficient and slope parameters to be estimated in equation 1.

betaN2 The number of coefficient and slope parameters to be estimated in equation 2.

References

Gourieroux, C. (2000). Econometrics of Qualitative Dependent Variables (Themes in Modern Econometrics) (P. Klassen, Trans.). Cambridge: Cambridge University Press. http://doi.org/10.1017/CBO9780511805608

Maddala, G. (1983). Limited-Dependent and Qualitative Variables in Econometrics (Econometric Society Monographs). Cambridge: Cambridge University Press. http://doi.org/10.1017/CBO9780511810176

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

str(results)
```

DlhoodDatanhrho *Derivative of likelihood with respect to the inverse hyperbolic tangent of correlation*

Description

Derivative of likelihood with respect to the inverse hyperbolic tangent of correlation

Usage

```
DlhoodDatanhrho(Y, mu, logsigma11, logsigma22, atanhrho)
```

Arguments

Y	A vector of observed responses.
mu	A $N \times 2$ matrix of means for equations 1 and 2.
logsigma11	A scalar log of the variance of the equation 1.
logsigma22	A scalar log of the variance of the equation 2.
atanhrho	A scalar log of inverse hyperbolic tangent of the correlation of equations 1 and 2.

Value

A vector of derivatives for each observation.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])

d = DlhoodDatanhrho(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2])
head(d)
```

DlhoodDbeta1 *Derivative of likelihood with respect to the coefficients of equation 1*

Description

Derivative of likelihood with respect to the coefficients of equation 1

Usage

```
DlhoodDbeta1(Y, mu, logsigma11, logsigma22, atanhrho, X1)
```

Arguments

Y	A vector of observed responses.
mu	A $N \times 2$ matrix of means for equations 1 and 2.
logsigma11	A scalar log of the variance of the equation 1.
logsigma22	A scalar log of the variance of the equation 2.
atanhrho	A scalar of the inverse hyperbolic tangent of the correlation of equations 1 and 2.
X1	A $N \times k_1$ design matrix for equation 1.

Value

A matrix of derivatives for each observation and parameter.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])

d = DlhoodDbeta1(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2], X1 = X1)
head(d)
```

`DlhoodDlogsigma11` *Derivative of likelihood with respect to the log of variance for equation 1*

Description

Derivative of likelihood with respect to the log of variance for equation 1

Usage

```
DlhoodDlogsigma11(Y, mu, logsigma11, logsigma22, atanhrho)
```

Arguments

`Y`	A vector of observed responses.
`mu`	A $N \times 2$ matrix of means for equations 1 and 2.
`logsigma11`	A scalar log of the variance of the equation 1.
`logsigma22`	A scalar log of the variance of the equation 2.
`atanhrho`	A scalar of the inverse hyperbolic tangent of the correlation of equations 1 and 2.

Value

A vector of derivatives for each observation.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
```

```
p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])

d = DlhoodDlogsigma11(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2])
head(d)
```

`DllhoodDatanhrho` *Derivative of log likelihood with respect to the inverse hyperbolic tangent of correlation*

Description

Derivative of log likelihood with respect to the inverse hyperbolic tangent of correlation

Usage

```
DllhoodDatanhrho(Y, mu, logsigma11, logsigma22, atanhrho, lhood)
```

Arguments

`Y`	A vector of observed responses.
`mu`	A $N \times 2$ matrix of means for equations 1 and 2.
`logsigma11`	A scalar log of the variance of the equation 1.
`logsigma22`	A scalar log of the variance of the equation 2.
`atanhrho`	A scalar of the inverse hyperbolic tangent of the correlation of equations 1 and 2.
`lhood`	A vector of length N of likelihood values.

Value

A vector of derivatives for each observation.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])
```

```
Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])
lhood = exp(-nLLikelihoodDE(theta, Y, X, transformR3toPD = TRUE, summed = FALSE))

d <- DllhoodDatanhrho(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2], lhood = lhood)
head(d)
```

`DllhoodDbeta1` *Gradient of log likelihood with respect to the coefficients of equation 1*

Description

Gradient of log likelihood with respect to the coefficients of equation 1

Usage

```
DllhoodDbeta1(Y, mu, logsigma11, logsigma22, atanhrho, lhood, X1)
```

Arguments

`Y`	A vector of observed responses.
`mu`	A $N \times 2$ matrix of means for equations 1 and 2.
`logsigma11`	A scalar log of the variance of the equation 1.
`logsigma22`	A scalar log of the variance of the equation 2.
`atanhrho`	A scalar of the inverse hyperbolic tangent of the correlation of equations 1 and 2.
`lhood`	A vector of length N of likelihood values.
`X1`	A $N \times k_1$ design matrix for equation 1.

Value

A matrix of derivatives for each observation and parameter.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])
lhood = exp(-nLLikelihoodDE(theta, Y, X, transformR3toPD = TRUE, summed = FALSE))

d = DllhoodDbeta1(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2], lhood = lhood, X1 = X1)
head(d)
```

`DllhoodDlogsigma11` *Derivative of log likelihood with respect to the log of variance for equation 1*

Description

Derivative of log likelihood with respect to the log of variance for equation 1

Usage

```
DllhoodDlogsigma11(Y, mu, logsigma11, logsigma22, atanhrho, lhood)
```

Arguments

`Y`	A vector of observed responses.
`mu`	A $N \times 2$ matrix of means for equations 1 and 2.

logsigma11	A scalar log of the variance of the equation 1.
logsigma22	A scalar log of the variance of the equation 2.
atanhrho	A scalar of the inverse hyperbolic tangent of the correlation of equations 1 and 2.
lhood	A vector of length N of likelihood values.

Value

A vector of derivatives for each observation.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))
mu = cbind(X[[1]] %*% theta[1:p1], X[[2]] %*% theta[(p1 + 1):(p1 + p2)])
lhood = exp(-nLLikelihoodDE(theta, Y, X, transformR3toPD = TRUE, summed = FALSE))

d <- DllhoodDlogsigma11(Y = Y, mu = mu, logsigma11 = theta[p1 + p2 + 1],
   logsigma22 = theta[p1 + p2 + 3], atanhrho = theta[p1 + p2 + 2], lhood = lhood)
head(d)
```

---

estfun.DE *estfun method for class 'DE'*

Description

estfun method for class 'DE'

Usage

```
estfun.DE(x, ...)
```

Arguments

x	An object of class DE.
...	Unused

Value

A $N \times (k_1 + k_2 + 3)$ if rho is not masked (otherwise $N \times (k_1 + k_2 + 2)$) matrix of the gradient is returned.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

head(sandwich::estfun(results))
```

---

fjdata *Data from Fair and Jaffee (1972).*

Description

A data set of 126 monthly observations of the housing market from June 1959 to November 1969. The variables are

T: Time. A continuous values time variable $\sum_{i=1}^{t-1} HS_i$: Stock of houses. A sum of housing starts over all previous periods. Assuming initial stock is 0. RM_{t-2}: Mortgage rate lagged by two months. $DF6_{t-1}$: A six month moving average of DF_t. DF_t is the flow of private deposits into savings and loan associations (SLA) and mutual savings banks during period t. $DHF3_{t-2}$: The flow of borrowings by the SLAs from the federal home-loan bank during month t. RM_{t-1}: Mortgage rate lagged by one month.

Usage

```
fjdata
```

Format

An object of class `data.frame` with 127 rows and 8 columns.

References

Fair, Ray C., and Dwight M. Jaffee. "Methods of estimation for markets in disequilibrium." Econometrica: Journal of the Econometric Society (1972): 497-514.

`GetDeltaMethodParameters`

GetDeltaMethodParameters

Description

Transforms the mean and variance covariance matrix of the estimators in an unrestricted space to a positive definite space for the delta method.

Usage

```
GetDeltaMethodParameters(mu, covmat = NULL, MaskRho = FALSE)
```

Arguments

`mu`	A numeric vector output from DE()$par
`covmat`	A numeric matrix output from DE()$vcov
`MaskRho`	The output from DE()$MaskRho

Value

A list containing the parameters of the normal distribution after transforming the coavariance variables to a positive definite space

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

GetDeltaMethodParameters(results$par,results$vcov,results$MaskRho)
```

GradientDE *Gradient of log likelihood with respect to all parameters*

Description

Gradient of log likelihood with respect to all parameters

Usage

```
GradientDE(theta, Y, X, summed = TRUE, MaskRho = FALSE)
```

Arguments

theta	A vector of parameter values to obtain the gradient at. The order of parameters is coefficients of equation 1, coefficients of equation 2, log variance of equation 1, inverse hyperbolic tangent of the correlation of equations 1 and 2, and log variance of equation 2.
Y	A vector of observed responses.
X	A list of two elements. The first element is a $N \times k_1$ design matrix for equation 1 and the second element is a $N \times k_2$ design matrix for equation 2.

summed	A logical to determine if gradient values are summed over observations.
MaskRho	A logical or numeric to determine if the correlation is masked. A value of FALSE means the correlation is not fixed. A value between -1 and 1 will fix the correlation to that value.

Value

A $(k_1 + k_2 + 3)$ dimension vector of derivatives if summed = TRUE, else a $N \times (k_1 + k_2 + 3)$ matrix of derivatives.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))

Gradient = GradientDE(theta, Y, X, summed = TRUE)
head(Gradient)
```

LLikelihoodDE *Log likelihood of market in disequilibrium model*

Description

Log likelihood of market in disequilibrium model

Usage

```
LLikelihoodDE(theta, Y, X, transformR3toPD = TRUE, summed = TRUE,
  MaskRho = FALSE)
```

Arguments

theta	A vector of parameter values to obtain the gradient at. The order of parameters is coefficients of equation 1, coefficients of equation 2, variance of equation 1, correlation of equations 1 and 2, and variance of equation 2.
Y	A vector of observed responses.
X	A list of two elements. The first element is a $N \times k_1$ design matrix for equation 1 and the second element is a $N \times k_2$ design matrix for equation 2.
transformR3toPD	A logical to determine if the covariance matrix is transformed to an unrestricted 3 dimension real space (transformR3toPD = TRUE) or not (transformR3toPD = FALSE).
summed	A logical to determine if the negative log likelihood values are summed over observations.
MaskRho	A logical or numeric to determine if the correlation is masked. A value of FALSE means the correlation is not fixed. A value between -1 and 1 will fix the correlation to that value.

Value

A scalar value of the negative log likelihood if summed = TRUE, else a N length vector of negative log likelihood observations.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
```

```
p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))

lhood = LLikelihoodDE(theta, Y, X, summed = TRUE)
head(LLikelihoodDE)
```

---

nGradientDE *Negative gradient of log likelihood with respect to all parameters*

Description

Negative gradient of log likelihood with respect to all parameters

Usage

```
nGradientDE(theta, Y, X, summed = TRUE, MaskRho = FALSE)
```

Arguments

theta	A vector of parameter values to obtain the gradient at. The order of parameters is coefficients of equation 1, coefficients of equation 2, log variance of equation 1, inverse hyperbolic tangent of the correlation of equations 1 and 2, and log variance of equation 2.
Y	A vector of observed responses.
X	A list of two elements. The first element is a $N \times k_1$ design matrix for equation 1 and the second element is a $N \times k_2$ design matrix for equation 2.
summed	A logical to determine if gradient values are summed over observations.
MaskRho	A logical or numeric to determine if the correlation is masked. A value of FALSE means the correlation is not fixed. A value between -1 and 1 will fix the correlation to that value.

Value

A $(k_1 + k_2 + 3)$ dimension vector of derivatives if summed = TRUE, else a $N \times (k_1 + k_2 + 3)$ matrix of derivatives.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
```

```
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))

Gradient = nGradientDE(theta, Y, X, summed = TRUE)
head(Gradient)
```

`nLLikelihoodDE` *Negative log likelihood of market in disequilibrium model*

Description

A wrapper function that makes the output of LLikelihoodDE negative. See ?LLikelihoodDE for details.

Usage

```
nLLikelihoodDE(...)
```

Arguments

`...`	arguements to be passed to LLikelhoodDE

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])
```

```
Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)

p1 = 2
p2 = 2
theta = c(beta01, beta02, log(SigmaX[1, 1]), atanh(SigmaX[1, 2]), log(SigmaX[2, 2]))

lhood = nLLikelihoodDE(theta, Y, X, summed = TRUE)
head(nLLikelihoodDE)
```

---

nobs.DE *nobs method for class 'DE'*

Description

nobs method for class 'DE'

Usage

```
## S3 method for class 'DE'
nobs(object, ...)
```

Arguments

object	An object of class DE.
...	Unused

Value

A scalar representing the number of observations N.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
```

```
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

nobs(results)
```

`predict.DE` *Predict method for class 'DE'*

Description

Predict method for class 'DE'

Usage

```
## S3 method for class 'DE'
predict(object, newdata = NULL, ...)
```

Arguments

`object`	An object of class `DE`.
`newdata`	An optional data frame with column names matching the dependent variables specified in the `formula` of the `DE` function. If not provided, the `data` from the `DE` function will be used.
`...`	Unused

Value

A data frame is returned. The columns are:

Y_1 Linear prediction of the outcome variable in equation 1.

Y_2 Linear prediction of the outcome variable in equation 2.

Min(Y_1,Y_2) The minimum of `Y_1` and `Y_2`.

Prob(Y_2>Y_1) The probability that the outcome variable in equation 2 is greater than the outcome variable in equation 1. This is the probability that `Y_1` is the observed quantity. This probability does not account for estimation uncertainty. Also note that all predictions are unconditional on the observed quantity.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

head(predict(results))
```

`residuals.DE` *residuals method for class 'DE'*

Description

residuals method for class 'DE'

Usage

```
## S3 method for class 'DE'
residuals(object, ...)
```

Arguments

`object`	An object of class `DE`.
`...`	Unused

Value

A 2 dimension matrix of raw residuals for equations 1 and 2. Allows use with Sandwich package.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

head(residuals(results))
```

---

summary.DE *Summary method for class 'DE'*

Description

Summary method for class 'DE'

Usage

```
## S3 method for class 'DE'
summary(object, robust = FALSE, ...)
```

Arguments

object	An object of class DE for which a summary is desired.
robust	A logical determining if robust standard errors should be used. If true standard errors are from sandwich::sandwich, else hessian.
...	Unused

Value

A matrix summary of estimates is returned. The columns are:

Estimate Maximum likelihood point estimate.

Std. Error Asymptotic standard error estimate of maximum likelihood point estimators using numerical hessian.

z value z value for zero value null hypothesis using asymptotic standard error estimate.

Pr(>|z|) P value for a two sided null hyptothesis test using the z value.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

summary(results)
```

TransformSigma_PDtoR2 *TransformSigma_PDtoR2*

Description

TransformSigma_PDtoR2

Usage

```
TransformSigma_PDtoR2(vec)
```

Arguments

vec	A length 2 vector of the variance of equation 1 followed by the variance of equation 2.

Value

A length 2 vector spanning unrestricted R2.

Examples

```
PD_vec <- c(1, 1)
TransformSigma_PDtoR2(PD_vec)
```

TransformSigma_PDtoR3 *TransformSigma_PDtoR3*

Description

TransformSigma_PDtoR3

Usage

```
TransformSigma_PDtoR3(vec)
```

Arguments

vec	A length 3 vector of the variance of equation 1, followed by the covariance of equations 1 and 2, followed by the variance of equation 2.

Value

A length 3 vector spanning unrestricted R3.

Examples

```
PD_vec <- c(1, 0, 1)
TransformSigma_PDtoR3(PD_vec)
```

TransformSigma_R2toPD *TransformSigma_R2toPD*

Description

TransformSigma_R2toPD

Usage

```
TransformSigma_R2toPD(vec)
```

Arguments

vec	A length 2 vector output from TransformSigma_PDtoR2.

Value

A length 2 vector of variances spanning a positive definite space.

Examples

```
PD_vec <- c(1, 1)
R2_vec <- TransformSigma_PDtoR2(PD_vec)
TransformSigma_R2toPD(R2_vec)
```

TransformSigma_R3toPD *TransformSigma_R3toPD*

Description

TransformSigma_R3toPD

Usage

```
TransformSigma_R3toPD(vec)
```

Arguments

vec	A length 3 vector output from TransformSigma_PDtoR3.

Value

A length 3 vector spanning a positive definite space.

Examples

```
PD_vec <- c(1, 0, 1)
R3_vec <- TransformSigma_PDtoR3(PD_vec)
TransformSigma_R3toPD(R3_vec)
```

---

vcov.DE *vcov method for class 'DE'*

Description

vcov method for class 'DE'

Usage

```
## S3 method for class 'DE'
vcov(object, ...)
```

Arguments

object	An object of class DE.
...	Unused

Value

A $N \times (k_1 + k_2 + 3)$ if matrix of the gradient is returned.

Examples

```
set.seed(1775)
library(MASS)
beta01 = c(1,1)
beta02 = c(-1,-1)
N = 10000
SigmaEps = diag(2)
SigmaX = diag(2)
MuX = c(0,0)
par0 = c(beta01, beta02, SigmaX[1, 1], SigmaX[1, 2], SigmaX[2, 2])

Xgen = mvrnorm(N,MuX,SigmaX)
X1 = cbind(1,Xgen[,1])
X2 = cbind(1,Xgen[,2])
X = list(X1 = X1,X2 = X2)
eps = mvrnorm(N,c(0,0),SigmaEps)
eps1 = eps[,1]
eps2 = eps[,2]
Y1 = X1 %*% beta01 + eps1
Y2 = X2 %*% beta02 + eps2
Y = pmin(Y1,Y2)
```

```
df = data.frame(Y = Y, X1 = Xgen[,1], X2 = Xgen[,2])

results = DE(formula = Y ~ X1 | X2, data = df)

vcov(results)
```

Index