

Package ‘CDatanet’

April 5, 2022

Type Package

Title Modeling Count Data with Peer Effects

Version 1.0.1

Date 2022-04-04

Description

Likelihood-based estimation and data generation from a class of models used to estimate peer effects on count data by controlling for the network endogeneity. This class includes count data models with social interactions (Houndetoungan 2022; <doi:10.2139/ssrn.3721250>), spatial tobit models (Xu and Lee 2015; <doi:10.1016/j.jeconom.2015.05.004>), and spatial linear-in-means models (Lee 2004; <doi:10.1111/j.1468-0262.2004.00558.x>).

License GPL-3

Language en-US

Encoding UTF-8

BugReports <https://github.com/ahoundetoungan/CDatanet/issues>

URL <https://github.com/ahoundetoungan/CDatanet>

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.0), Formula, formula.tools, ddpcr, Matrix

LinkingTo Rcpp, RcppArmadillo, RcppProgress, RcppDist

RoxygenNote 7.1.2

Suggests ggplot2, MASS, knitr, rmarkdown

NeedsCompilation yes

Author Elysée Aristide Houndetoungan [cre, aut]

Maintainer Elysée Aristide Houndetoungan <arie192and@gmail.com>

Repository CRAN

Date/Publication 2022-04-05 06:40:06 UTC

R topics documented:

CDatanet-package	2
cdnet	3
homophily	6
sar	9
sart	11
simcdnet	14
simsar	16
simsart	18
summary.cdnet	21
summary.sar	22
summary.sart	23
Index	25

CDatanet-package	<i>The CDatanet package</i>
------------------	-----------------------------

Description

The **CDatanet** package implements the count data model with social interactions and the dyadic linking model developed in Houndetoungan (2020). It also simulates data from the count data model and implements the Spatial Autoregressive Tobit model (LeSage, 2000; Xu and Lee, 2015) for left censored data and the Spatial Autoregressive Model (Lee, 2004). To make the computations faster **CDatanet** uses C++ through the **Rcpp** package (Eddelbuettel et al., 2011).

Author(s)

Maintainer: Elysée Aristide Houndetoungan <ariele92and@gmail.com>

References

- Atchade, Y. F., & Rosenthal, J. S. (2005). On adaptive markov chain monte carlo algorithms, *Bernoulli*, 11(5), 815-828, doi: [10.3150/bj/1130077595](https://doi.org/10.3150/bj/1130077595)
- Eddelbuettel, D., François, R., Allaire, J., Ushey, K., Kou, Q., Russel, N., ... & Bates, D., 2011, **Rcpp**: Seamless R and C++ integration, *Journal of Statistical Software*, 40(8), 1-18, doi: [10.18637/jss.v040.i08](https://doi.org/10.18637/jss.v040.i08)
- Houndetoungan, E. A., 2022, Count Data Models with Social Interactions under Rational Expectations. Available at SSRN 3721250, doi: [10.2139/ssrn.3721250](https://doi.org/10.2139/ssrn.3721250)
- Lee, L. F., 2004, Asymptotic distributions of quasi-maximum likelihood estimators for spatial autoregressive models. *Econometrica*, 72(6), 1899-1925, doi: [10.1111/j.14680262.2004.00558.x](https://doi.org/10.1111/j.14680262.2004.00558.x)
- Xu, X., & Lee, L. F., 2015, Maximum likelihood estimation of a spatial autoregressive Tobit model, *Journal of Econometrics*, 188(1), 264-280, doi: [10.1016/j.jeconom.2015.05.004](https://doi.org/10.1016/j.jeconom.2015.05.004)

See Also

Useful links:

- <https://github.com/ahoundetoungan/CDatanet>
- Report bugs at <https://github.com/ahoundetoungan/CDatanet/issues>

cdnet	<i>Estimate Count Data Model with Social Interactions using NPL Method</i>
-------	--

Description

Estimate Count Data Model with Social Interactions using NPL Method

Usage

```
cdnet(
  formula,
  contextual,
  Glist,
  Rbar = NULL,
  starting = list(theta = NULL, delta = NULL),
  yb0 = NULL,
  optimizer = "optim",
  npl.ctr = list(),
  opt.ctr = list(),
  cov = TRUE,
  data
)
```

Arguments

formula	an object of class formula : a symbolic description of the model. The formula should be as for example $y \sim x_1 + x_2 \mid x_1 + x_2$ where y is the endogenous vector, the listed variables before the pipe, x_1, x_2 are the individual exogenous variables and the listed variables after the pipe, x_1, x_2 are the contextual observable variables. Other formulas may be $y \sim x_1 + x_2$ for the model without contextual effects, $y \sim -1 + x_1 + x_2 \mid x_1 + x_2$ for the model without intercept or $y \sim x_1 + x_2 \mid x_2 + x_3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the the formula as $y \sim x_1 + x_2$ and contextual as TRUE is equivalent to set the formula as $y \sim x_1 + x_2 \mid x_1 + x_2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
Rbar	the value of Rbar. If not provided, it is automatically set at <code>quantile(y, 0.9)</code> .

starting	(optional) starting value of $\theta = (\lambda, \beta', \gamma)'$ and $\delta = (\delta_2, \dots, \delta_{\bar{R}})$. The parameter γ should be removed if the model does not contain contextual effects (see details).
yb0	(optional) expectation of y .
optimizer	is either <code>nlm</code> (referring to the <code>nlm</code> function) or <code>optim</code> (referring to the <code>optim</code> function). At every step of the NPL method, the estimation is performed using <code>nlm</code> or <code>optim</code> . Other arguments of these functions such as, <code>control</code> and <code>method</code> can be defined through the argument <code>opt.ctr</code> .
npl.ctr	list of controls for the NPL method (see details).
opt.ctr	list of arguments of <code>nlm</code> or <code>optim</code> (the one set in <code>optimizer</code>) such as <code>control</code> , <code>method</code> , ...
cov	a boolean indicating if the covariance should be computed.
data	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model. If not found in <code>data</code> , the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>cdnet</code> is called.

Details

Model:

Following Houndetoungan (2020), the count data y is generated from a latent variable y^* . The latent variable is given for all i as

$$y_i^* = \lambda \mathbf{g}_i \bar{y} + \mathbf{x}_i' \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, 1)$.

Then, $y_i = r$ iff $a_r \leq y_i^* \leq a_{r+1}$, where $a_0 = -\inf$, $a_1 = 0$, $a_r = \sum_{k=1}^r \delta_k$ if $1 \leq r \leq \bar{R}$, and $a_r = (r - \bar{R})\delta_{\bar{R}} + a_{\bar{R}}$ otherwise.

`npl.ctr`:

The model parameters is estimated using the Nested Partial Likelihood (NPL) method. This approach starts with a guess of θ and \bar{y} and constructs iteratively a sequence of θ and \bar{y} . The solution converges when the L_1 distance between two consecutive θ and \bar{y} is less than a tolerance. The argument `npl.ctr` is an optional list which contain

- tol the tolerance of the NPL algorithm (default 1e-4),
- maxit the maximal number of iterations allowed (default 500),
- print a boolean indicating if the estimate should be printed at each step.
- S the number of simulation performed use to compute integral in the covariance by important sampling.

Value

A list consisting of:

info	list of general information about the model.
estimate	NPL estimator.

yb ybar (see details), expectation of y.
 Gyb average of the expectation of y among friends.
 cov list of covariance matrices.
 details step-by-step output as returned by the optimizer.

See Also

[sart](#), [sar](#), [simcdnet](#).

Examples

```

# Groups' size
M      <- 5 # Number of sub-groups
nvec   <- round(runif(M, 100, 1000))
n      <- sum(nvec)

# Parameters
lambda <- 0.4
beta   <- c(1.5, 2.2, -0.9)
gamma  <- c(1.5, -1.2)
delta  <- c(1, 0.87, 0.75, 0.6, 0.4)
theta  <- c(lambda, beta, gamma)

# X
X      <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist <- list()

for (m in 1:M) {
  nm      <- nvec[m]
  Gm      <- matrix(0, nm, nm)
  max_d   <- 30
  for (i in 1:nm) {
    tmp    <- sample((1:nm)[-i], sample(0:max_d, 1))
    Gm[i, tmp] <- 1
  }
  rs      <- rowSums(Gm); rs[rs == 0] <- 1
  Gm      <- Gm/rs
  Glist[[m]] <- Gm
}

# data
data    <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta", "delta"))])

ytmp   <- simcdnet(formula = ~ x1 + x2 | x1 + x2, Glist = Glist, theta = theta,
                  delta = delta, data = data)

```

```

y      <- ytmp$y

# plot histogram
hist(y, breaks = max(y))

data  <- data.frame(yt = y, x1 = data$x1, x2 = data$x2)
rm(list = ls()[!(ls() %in% c("Glist", "data"))])

out <- cdnet(formula = yt ~ x1 + x2, contextual = TRUE, Glist = Glist, data = data, Rbar = 6)
summary(out)

```

homophily

Estimate Network Formation Model

Description

Estimate Network Formation Model

Usage

```

homophily(
  network,
  formula,
  data,
  fixed.effects = FALSE,
  init = list(),
  iteration = 1000,
  print = TRUE
)

```

Arguments

network	matrix or list of sub-matrix of social interactions containing 0 and 1, where links are represented by 1
formula	an object of class formula : a symbolic description of the model. The formula should be as for example $\sim x1 + x2$ where $x1$, $x2$ are explanatory variable of links formation
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>homophily</code> is called.
fixed.effects	boolean indicating if sub-network heterogeneity as fixed effects should be included.

<code>init</code>	(optional) list of starting values containing beta, an K-dimensional vector of the explanatory variables parameter, mu an n-dimensional vector, and nu an n-dimensional vector, smu2 the variance of mu, and snu2 the variance of nu, where K is the number of explanatory variables and n is the number of individuals.
<code>iteration</code>	the number of iterations to be performed.
<code>print</code>	boolean indicating if the estimation progression should be printed.

Value

A list consisting of:

<code>n</code>	number of individuals in each network.
<code>n.obs</code>	number of observations.
<code>n.links</code>	number of links.
<code>K</code>	number of explanatory variables.
<code>posterior</code>	list of simulations from the posterior distribution.
<code>iteration</code>	number of performed iterations.
<code>init</code>	returned list of starting values.

Examples

```

set.seed(1234)
library(MASS)
M      <- 4 # Number of sub-groups
nvec   <- round(runif(M, 100, 500))
beta   <- c(.1, -.1)
Glist  <- list()
dX     <- matrix(0, 0, 2)
mu     <- list()
nu     <- list()
cst    <- runif(M, -1.5, 0)
smu2   <- 0.2
snu2   <- 0.2
rho    <- 0.8
Smunu  <- matrix(c(smu2, rho*sqrt(smu2*snu2), rho*sqrt(smu2*snu2), snu2), 2)
for (m in 1:M) {
  n      <- nvec[m]
  tmp    <- mvrnorm(n, c(0, 0), Smunu)
  mum    <- tmp[,1] - mean(tmp[,1])
  num    <- tmp[,2] - mean(tmp[,2])
  X1     <- rnorm(n, 0, 1)
  X2     <- rbinom(n, 1, 0.2)
  Z1     <- matrix(0, n, n)
  Z2     <- matrix(0, n, n)

  for (i in 1:n) {
    for (j in 1:n) {
      Z1[i, j] <- abs(X1[i] - X1[j])
    }
  }
}

```

```

      Z2[i, j] <- 1*(X2[i] == X2[j])
    }
  }

  Gm      <- 1*((cst[m] + Z1*beta[1] + Z2*beta[2] +
                kronecker(mum, t(num), "+") + rnorm(n^2)) > 0)

  diag(Gm) <- 0
  diag(Z1) <- NA
  diag(Z2) <- NA
  Z1      <- Z1[!is.na(Z1)]
  Z2      <- Z2[!is.na(Z2)]

  dX      <- rbind(dX, cbind(Z1, Z2))
  Glist[[m]] <- Gm
  mu[[m]] <- mum
  nu[[m]] <- num
}

mu <- unlist(mu)
nu <- unlist(nu)

out <- homophily(network = Glist, formula = ~ dX, fixed.effects = TRUE,
                 iteration = 1e3)

# plot simulations
plot(out$posterior$beta[,1], type = "l")
abline(h = cst[1], col = "red")
plot(out$posterior$beta[,2], type = "l")
abline(h = cst[2], col = "red")
plot(out$posterior$beta[,3], type = "l")
abline(h = cst[3], col = "red")
plot(out$posterior$beta[,4], type = "l")
abline(h = cst[4], col = "red")

plot(out$posterior$beta[,5], type = "l")
abline(h = beta[1], col = "red")
plot(out$posterior$beta[,6], type = "l")
abline(h = beta[2], col = "red")

plot(out$posterior$sigma2_mu, type = "l")
abline(h = smu2, col = "red")
plot(out$posterior$sigma2_nu, type = "l")
abline(h = snu2, col = "red")
plot(out$posterior$rho, type = "l")
abline(h = rho, col = "red")

i <- 10
plot(out$posterior$mu[,i], type = "l")
abline(h = mu[i], col = "red")
plot(out$posterior$nu[,i], type = "l")
abline(h = nu[i], col = "red")

```

 sar

Estimate SAR model

Description

Estimate SAR model

Usage

```

sar(
  formula,
  contextual,
  Glist,
  lambda0 = NULL,
  optimizer = "optim",
  opt.ctr = list(),
  print = TRUE,
  cov = TRUE,
  data
)

```

Arguments

formula	an object of class formula : a symbolic description of the model. The formula should be as for example $y \sim x_1 + x_2 \mid x_1 + x_2$ where y is the endogenous vector, the listed variables before the pipe, x_1, x_2 are the individual exogenous variables and the listed variables after the pipe, x_1, x_2 are the contextual observable variables. Other formulas may be $y \sim x_1 + x_2$ for the model without contextual effects, $y \sim -1 + x_1 + x_2 \mid x_1 + x_2$ for the model without intercept or $y \sim x_1 + x_2 \mid x_2 + x_3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the formula as $y \sim x_1 + x_2$ and contextual as TRUE is equivalent to set the formula as $y \sim x_1 + x_2 \mid x_1 + x_2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
lambda0	(optional) starting value of λ . The parameter γ should be removed if the model does not contain contextual effects (see details).
optimizer	is either <code>nlm</code> (referring to the function nlm) or <code>optim</code> (referring to the function optim). Other arguments of these functions such as, the control values and the method can be defined through the argument <code>opt.ctr</code> .
opt.ctr	list of arguments of nlm or optim (the one set in optimizer) such as control, method, ...
print	a boolean indicating if the estimate should be printed at each step.
cov	a boolean indicating if the covariance should be computed.

`data` an optional data frame, list or environment (or object coercible by [as.data.frame](#) to a data frame) containing the variables in the model. If not found in `data`, the variables are taken from `environment(formula)`, typically the environment from which `mcmcARD` is called.

Details

Model:

The variable y is given for all i as

$$y_i = \lambda \mathbf{g}_i y + \mathbf{x}_i' \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$.

Value

A list consisting of:

<code>info</code>	list of general information on the model.
<code>estimate</code>	Maximum Likelihood (ML) estimator.
<code>cov</code>	covariance matrix of the estimate.
<code>details</code>	outputs as returned by the optimizer.

See Also

[sart](#), [cdnet](#), [simsar](#).

Examples

```
# Groups' size
M <- 5 # Number of sub-groups
nvec <- round(runif(M, 100, 1000))
n <- sum(nvec)

# Parameters
lambda <- 0.4
beta <- c(2, -1.9, 0.8)
gamma <- c(1.5, -1.2)
sigma <- 1.5
theta <- c(lambda, beta, gamma, sigma)

# X
X <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist <- list()

for (m in 1:M) {
  nm <- nvec[m]
  Gm <- matrix(0, nm, nm)
```

```

max_d      <- 30
for (i in 1:nm) {
  tmp      <- sample((1:nm)[-i], sample(0:max_d, 1))
  Gm[i, tmp] <- 1
}
rs        <- rowSums(Gm); rs[rs == 0] <- 1
Gm        <- Gm/rs
Glist[[m]] <- Gm
}

# data
data <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta"))])

ytmp <- simsar(formula = ~ x1 + x2 | x1 + x2, Glist = Glist,
               theta = theta, data = data)

y <- ytmp$y

# plot histogram
hist(y, breaks = max(y))

data <- data.frame(yt = y, x1 = data$x1, x2 = data$x2)
rm(list = ls()[!(ls() %in% c("Glist", "data"))])

out <- sar(formula = yt ~ x1 + x2, contextual = TRUE,
           Glist = Glist, optimizer = "optim", data = data)
summary(out)

```

sart

Estimate sart model

Description

Estimate sart model

Usage

```

sart(
  formula,
  contextual,
  Glist,
  theta0 = NULL,
  yb0 = NULL,
  optimizer = "optim",
  npl.ctr = list(),
  opt.ctr = list(),

```

```

    print = TRUE,
    cov = TRUE,
    RE = FALSE,
    data
  )

```

Arguments

formula	an object of class formula : a symbolic description of the model. The formula should be as for example $y \sim x1 + x2 \mid x1 + x2$ where y is the endogenous vector, the listed variables before the pipe, $x1, x2$ are the individual exogenous variables and the listed variables after the pipe, $x1, x2$ are the contextual observable variables. Other formulas may be $y \sim x1 + x2$ for the model without contextual effects, $y \sim -1 + x1 + x2 \mid x1 + x2$ for the model without intercept or $y \sim x1 + x2 \mid x2 + x3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the formula as $y \sim x1 + x2$ and contextual as TRUE is equivalent to set the formula as $y \sim x1 + x2 \mid x1 + x2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
theta0	(optional) starting value of $\theta = (\lambda, \beta, \gamma, \sigma)$. The parameter γ should be removed if the model does not contain contextual effects (see details).
yb0	(optional) expectation of y .
optimizer	is either <code>nlm</code> (referring to the function nlm) or <code>optim</code> (referring to the function optim). Other arguments of these functions such as, the control values and the method can be defined through the argument <code>opt.ctr</code> .
npl.ctr	list of controls for the NPL method (see cdnet).
opt.ctr	list of arguments of nlm or optim (the one set in optimizer) such as control, method, ...
print	a boolean indicating if the estimate should be printed at each step.
cov	a boolean indicating if the covariance should be computed.
RE	a boolean which indicates if the model is under rational expectation of not.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>sart</code> is called.

Details

Model:

The left-censored variable y is generated from a latent variable y^* . The latent variable is given for all i as

$$y_i^* = \lambda \mathbf{g}_i y + \mathbf{x}_i' \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$.

The count variable y_i is then defined that is $y_i = 0$ if $y_i^* \leq 0$ and $y_i = y_i^*$ otherwise.

Value

A list consisting of:

info	list of general information on the model.
estimate	Maximum Likelihood (ML) estimator.
yb	ybar (see details), expectation of y.
Gyb	average of the expectation of y among friends.
cov	List of covariances.
details	outputs as returned by the optimizer.

See Also

[sar](#), [cdnet](#), [simsart](#).

Examples

```
# Groups' size
M      <- 5 # Number of sub-groups
nvec   <- round(runif(M, 100, 1000))
n      <- sum(nvec)

# Parameters
lambda <- 0.4
beta   <- c(2, -1.9, 0.8)
gamma  <- c(1.5, -1.2)
sigma  <- 1.5
theta  <- c(lambda, beta, gamma, sigma)

# X
X      <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist <- list()

for (m in 1:M) {
  nm      <- nvec[m]
  Gm      <- matrix(0, nm, nm)
  max_d   <- 30
  for (i in 1:nm) {
    tmp    <- sample((1:nm)[-i], sample(0:max_d, 1))
    Gm[i, tmp] <- 1
  }
  rs      <- rowSums(Gm); rs[rs == 0] <- 1
  Gm      <- Gm/rs
  Glist[[m]] <- Gm
}

# data
```

```

data <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta"))])

ytmp <- simsart(formula = ~ x1 + x2 | x1 + x2, Glist = Glist,
                theta = theta, data = data)

y <- ytmp$y

# plot histogram
hist(y)

opt.ctr <- list(method = "Nelder-Mead",
                control = list(abstol = 1e-16, abstol = 1e-11, maxit = 5e3))
data <- data.frame(yt = y, x1 = data$x1, x2 = data$x2)
rm(list = ls()[!(ls() %in% c("Glist", "data"))])

out <- sart(formula = yt ~ x1 + x2, optimizer = "nlm",
            contextual = TRUE, Glist = Glist, data = data)
summary(out)

```

simcdnet

Simulate data from Count Data Model with Social Interactions

Description

Simulate data from Count Data Model with Social Interactions

Usage

```

simcdnet(
  formula,
  contextual,
  Glist,
  theta,
  delta,
  tol = 1e-15,
  maxit = 500,
  data
)

```

Arguments

formula an object of class [formula](#): a symbolic description of the model. The formula should be as for example $y \sim x1 + x2 \mid x1 + x2$ where y is the endogenous vector, the listed variables before the pipe, $x1$, $x2$ are the individual exogenous variables and the listed variables after the pipe, $x1$, $x2$ are the contextual observable variables. Other formulas may be $y \sim x1 + x2$ for the model without

	contextual effects, $y \sim -1 + x_1 + x_2 \mid x_1 + x_2$ for the model without intercept or $y \sim x_1 + x_2 \mid x_2 + x_3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the formula as $y \sim x_1 + x_2$ and contextual as TRUE is equivalent to set the formula as $y \sim x_1 + x_2 \mid x_1 + x_2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
theta	the true value of the vector $\theta = (\lambda, \beta', \gamma)'$. The parameter γ should be removed if the model does not contain contextual effects (see details).
delta	the true value of the vector $\delta = (\delta_2, \dots, \delta_{\bar{R}})$
tol	the tolerance value used in the Fixed Point Iteration Method to compute the expectancy of y . The process stops if the L_1 distance between two consecutive values of the expectancy of y is less than tol.
maxit	the maximal number of iterations in the Fixed Point Iteration Method.
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>mcmcARD</code> is called.

Details

Following Houndetoungan (2020), the count data \mathbf{y} is generated from a latent variable \mathbf{y}^* . The latent variable is given for all i as

$$y_i^* = \lambda \mathbf{g}_i \bar{\mathbf{y}} + \mathbf{x}_i' \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, 1)$.

Then, $y_i = r$ iff $a_r \leq y_i^* \leq a_{r+1}$, where $a_0 = -\inf$, $a_1 = 0$, $a_r = \sum_{k=1}^r \delta_k$ if $1 \leq r \leq \bar{R}$, and $a_r = (r - \bar{R})\delta_{\bar{R}} + a_{\bar{R}}$ otherwise.

Value

A list consisting of:

yst	ys (see details), the latent variable.
y	the observed count data.
yb	ybar (see details), the expectation of y .
Gyb	the average of the expectation of y among friends.
marg.effects	the marginal effects.
iteration	number of iterations performed by sub-network in the Fixed Point Iteration Method.

See Also

[cdnet](#), [simsart](#), [simsar](#).

Examples

```

# Groups' size
M      <- 5 # Number of sub-groups
nvec   <- round(runif(M, 100, 1000))
n      <- sum(nvec)

# Parameters
lambda <- 0.4
beta   <- c(1.5, 2.2, -0.9)
gamma  <- c(1.5, -1.2)
delta  <- c(1, 0.87, 0.75, 0.6, 0.4)
theta  <- c(lambda, beta, gamma)

# X
X      <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist  <- list()

for (m in 1:M) {
  nm    <- nvec[m]
  Gm    <- matrix(0, nm, nm)
  max_d <- 30
  for (i in 1:nm) {
    tmp  <- sample((1:nm)[-i], sample(0:max_d, 1))
    Gm[i, tmp] <- 1
  }
  rs    <- rowSums(Gm); rs[rs == 0] <- 1
  Gm    <- Gm/rs
  Glist[[m]] <- Gm
}

# data
data   <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta", "delta"))])

ytmp   <- simcdnet(formula = ~ x1 + x2 | x1 + x2, Glist = Glist, theta = theta,
                  delta = delta, data = data)

y      <- ytmp$y

# plot histogram
hist(y, breaks = max(y))

```


Description

Simulate data from the linear-in-mean Model with Social Interactions

Usage

```
simsar(formula, contextual, Glist, theta, data)
```

Arguments

formula	an object of class formula : a symbolic description of the model. The formula should be as for example $y \sim x1 + x2 \mid x1 + x2$ where y is the endogenous vector, the listed variables before the pipe, $x1$, $x2$ are the individual exogenous variables and the listed variables after the pipe, $x1$, $x2$ are the contextual observable variables. Other formulas may be $y \sim x1 + x2$ for the model without contextual effects, $y \sim -1 + x1 + x2 \mid x1 + x2$ for the model without intercept or $y \sim x1 + x2 \mid x2 + x3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the formula as $y \sim x1 + x2$ and contextual as TRUE is equivalent to set the formula as $y \sim x1 + x2 \mid x1 + x2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
theta	the parameter value as $\theta = (\lambda, \beta, \gamma, \sigma)$. The parameter γ should be removed if the model does not contain contextual effects (see details).
data	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>mcmcARD</code> is called.

Details

The variable y is given for all i as

$$y_i = \lambda \mathbf{g}_i y + \mathbf{x}'_i \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$.

Value

A list consisting of:

y	the observed count data.
Gy	the average of y among friends.

See Also

[sar](#), [simsart](#), [simcdnet](#).

Examples

```

# Groups' size
M      <- 5 # Number of sub-groups
nvec   <- round(runif(M, 100, 1000))
n      <- sum(nvec)

# Parameters
lambda <- 0.4
beta   <- c(2, -1.9, 0.8)
gamma  <- c(1.5, -1.2)
sigma  <- 1.5
theta  <- c(lambda, beta, gamma, sigma)

# X
X      <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist <- list()

for (m in 1:M) {
  nm      <- nvec[m]
  Gm      <- matrix(0, nm, nm)
  max_d   <- 30
  for (i in 1:nm) {
    tmp    <- sample((1:nm)[-i], sample(0:max_d, 1))
    Gm[i, tmp] <- 1
  }
  rs      <- rowSums(Gm); rs[rs == 0] <- 1
  Gm      <- Gm/rs
  Glist[[m]] <- Gm
}

# data
data    <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta"))])

ytmp    <- simsar(formula = ~ x1 + x2 | x1 + x2, Glist = Glist,
                  theta = theta, data = data)
y       <- ytmp$y

# plot histogram
hist(y)

```

Description

Simulate data from the Tobit Model with Social Interactions

Usage

```
simsart(
  formula,
  contextual,
  Glist,
  theta,
  tol = 1e-15,
  maxit = 500,
  RE = FALSE,
  data
)
```

Arguments

formula	an object of class <code>formula</code> : a symbolic description of the model. The formula should be as for example $y \sim x_1 + x_2 \mid x_1 + x_2$ where y is the endogenous vector, the listed variables before the pipe, x_1, x_2 are the individual exogenous variables and the listed variables after the pipe, x_1, x_2 are the contextual observable variables. Other formulas may be $y \sim x_1 + x_2$ for the model without contextual effects, $y \sim -1 + x_1 + x_2 \mid x_1 + x_2$ for the model without intercept or $y \sim x_1 + x_2 \mid x_2 + x_3$ to allow the contextual variable to be different from the individual variables.
contextual	(optional) logical; if true, this means that all individual variables will be set as contextual variables. Set the formula as $y \sim x_1 + x_2$ and contextual as TRUE is equivalent to set the formula as $y \sim x_1 + x_2 \mid x_1 + x_2$.
Glist	the adjacency matrix or list sub-adjacency matrix.
theta	the parameter value as $\theta = (\lambda, \beta, \gamma, \sigma)$. The parameter γ should be removed if the model does not contain contextual effects (see details).
tol	the tolerance value used in the Fixed Point Iteration Method to compute y . The process stops if the L_1 distance between two consecutive values of y is less than <code>tol</code> .
maxit	the maximal number of iterations in the Fixed Point Iteration Method.
RE	a boolean which indicates if the model is under rational expectation or not.
data	an optional data frame, list or environment (or object coercible by <code>as.data.frame</code> to a data frame) containing the variables in the model. If not found in data, the variables are taken from <code>environment(formula)</code> , typically the environment from which <code>mcmcARD</code> is called.

Details

The left-censored variable y is generated from a latent variable y^* . The latent variable is given for all i as

$$y_i^* = \lambda g_i y + \mathbf{x}_i' \beta + \mathbf{g}_i \mathbf{X} \gamma + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$.

The count variable y_i is then define that is $y_i = 0$ if $y_i^* \leq 0$ and $y_i = y_i^*$ otherwise.

Value

A list consisting of:

yst	ys (see details), the latent variable.
y	the observed count data.
yb	expectation of y under rational expectation.
Gy	the average of y among friends.
Gyb	Average of expectation of y among friends under rational expectation.
marg.effects	the marginal effects.
iteration	number of iterations performed by sub-network in the Fixed Point Iteration Method.

See Also

[sart](#), [simsar](#), [simcdnet](#).

Examples

```
# Groups' size
M      <- 5 # Number of sub-groups
nvec   <- round(runif(M, 100, 1000))
n      <- sum(nvec)

# Parameters
lambda <- 0.4
beta   <- c(2, -1.9, 0.8)
gamma  <- c(1.5, -1.2)
sigma  <- 1.5
theta  <- c(lambda, beta, gamma, sigma)

# X
X      <- cbind(rnorm(n, 1, 1), rexp(n, 0.4))

# Network
Glist  <- list()

for (m in 1:M) {
  nm    <- nvec[m]
  Gm    <- matrix(0, nm, nm)
  max_d <- 30
  for (i in 1:nm) {
    tmp  <- sample((1:nm)[-i], sample(0:max_d, 1))
    Gm[i, tmp] <- 1
  }
  rs    <- rowSums(Gm); rs[rs == 0] <- 1
  Gm    <- Gm/rs
}
```

```

  Glist[[m]] <- Gm
}

# data
data <- data.frame(x1 = X[,1], x2 = X[,2])

rm(list = ls()[!(ls() %in% c("Glist", "data", "theta"))])

ytmp <- simsart(formula = ~ x1 + x2 | x1 + x2, Glist = Glist,
                theta = theta, data = data)

y <- ytmp$y

# plot histogram
hist(y)

```

summary.cdnet

Summarize Count Data Model with Social Interactions

Description

Summary and print methods for the class `cdnet` as returned by the function `cdnet`.

Usage

```

## S3 method for class 'cdnet'
summary(object, Glist, data, S = 1000L, ...)

## S3 method for class 'summary.cdnet'
print(x, ...)

## S3 method for class 'cdnet'
print(x, ...)

## S3 method for class 'cdnets'
summary(object, ...)

## S3 method for class 'summary.cdnets'
print(x, ...)

## S3 method for class 'cdnets'
print(x, ...)

```

Arguments

`object` an object of class `cdnet`, output of the function `cdnet`.

Glist	adjacency matrix or list sub-adjacency matrix. This is not necessary if the covariance method was computed in cdnet .
data	a dataframe containing the explanatory variables. This is not necessary if the covariance method was computed in cdnet .
S	number of simulation to be used to compute integral in the covariance by important sampling.
...	further arguments passed to or from other methods.
x	an object of class <code>summary.cdnet</code> , output of the function summary.cdnet , class <code>summary.cdnets</code> , list of outputs of the function summary.cdnet (when the model is estimated many times to control for the endogeneity) or class <code>cdnet</code> of the function cdnet .

Value

A list of the same objects in `object`.

summary.sar	<i>Summarize SAR Model</i>
-------------	----------------------------

Description

Summary and print methods for the class `sar` as returned by the function [sar](#).

Usage

```
## S3 method for class 'sar'
summary(object, ...)

## S3 method for class 'summary.sar'
print(x, ...)

## S3 method for class 'sar'
print(x, ...)

## S3 method for class 'sars'
summary(object, ...)

## S3 method for class 'summary.sars'
print(x, ...)

## S3 method for class 'sars'
print(x, ...)
```

Arguments

object	an object of class sar, output of the function sar .
...	further arguments passed to or from other methods.
x	an object of class summary.sar, output of the function summary.sar or class sar, output of the function sar .

Value

A list of the same objects in object.

summary.sart	<i>Summarize sart Model</i>
--------------	-----------------------------

Description

Summary and print methods for the class sart as returned by the function [sar](#).

Usage

```
## S3 method for class 'sart'
summary(object, Glist, data, ...)

## S3 method for class 'summary.sart'
print(x, ...)

## S3 method for class 'sart'
print(x, ...)

## S3 method for class 'sarts'
summary(object, ...)

## S3 method for class 'summary.sarts'
print(x, ...)

## S3 method for class 'sarts'
print(x, ...)
```

Arguments

object	an object of class sart, output of the function sar .
Glist	adjacency matrix or list sub-adjacency matrix. This is not necessary if the covariance method was computed in cdnet .
data	dataframe containing the explanatory variables. This is not necessary if the covariance method was computed in cdnet .
...	further arguments passed to or from other methods.
x	an object of class summary.sart, output of the function summary.sart or class sart, output of the function sar .

Value

A list of the same objects in object.

Index

`as.data.frame`, [4](#), [6](#), [10](#), [12](#), [15](#), [17](#), [19](#)

`CDatanet` (`CDatanet`-package), [2](#)

`CDatanet`-package, [2](#)

`cdnet`, [3](#), [10](#), [12](#), [13](#), [15](#), [21–23](#)

`formula`, [3](#), [6](#), [9](#), [12](#), [14](#), [17](#), [19](#)

`homophily`, [6](#)

`nlm`, [4](#), [9](#), [12](#)

`optim`, [4](#), [9](#), [12](#)

`print.cdnet` (`summary.cdnet`), [21](#)

`print.cdnets` (`summary.cdnet`), [21](#)

`print.sar` (`summary.sar`), [22](#)

`print.sars` (`summary.sar`), [22](#)

`print.sart` (`summary.sart`), [23](#)

`print.sarts` (`summary.sart`), [23](#)

`print.summary.cdnet` (`summary.cdnet`), [21](#)

`print.summary.cdnets` (`summary.cdnet`), [21](#)

`print.summary.sar` (`summary.sar`), [22](#)

`print.summary.sars` (`summary.sar`), [22](#)

`print.summary.sart` (`summary.sart`), [23](#)

`print.summary.sarts` (`summary.sart`), [23](#)

`sar`, [5](#), [9](#), [13](#), [17](#), [22](#), [23](#)

`sart`, [5](#), [10](#), [11](#), [20](#), [23](#)

`simcdnet`, [5](#), [14](#), [17](#), [20](#)

`simsar`, [10](#), [15](#), [16](#), [20](#)

`simsart`, [13](#), [15](#), [17](#), [18](#)

`summary.cdnet`, [21](#), [22](#)

`summary.cdnets` (`summary.cdnet`), [21](#)

`summary.sar`, [22](#), [23](#)

`summary.sars` (`summary.sar`), [22](#)

`summary.sart`, [23](#), [23](#)

`summary.sarts` (`summary.sart`), [23](#)