Package: QuantilePeer (via r-universe)

June 5, 2026

Type Package

Title Quantile Peer Effect Models

Version 0.0.1

Date 2025-07-10

Description Simulating and estimating peer effect models including the quantile-based specification (Houndetoungan, 2025 <doi:10.48550/arXiv.2506.12920>), and the models with Constant Elasticity of Substitution (CES)-based social norm (Boucher et al., 2024 <doi:10.3982/ECTA21048>).

License GPL-3

Language en-US

Encoding UTF-8

BugReports `https://github.com/ahoundetoungan/QuantilePeer/issues`

URL `https://github.com/ahoundetoungan/QuantilePeer`

Depends R (>= 3.5.0)

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

LinkingTo Rcpp, RcppArmadillo, RcppEigen, RcppNumerical

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, PartialNetwork,

VignetteBuilder knitr

NeedsCompilation yes

Author Aristide Houndetoungan [cre, aut] (ORCID: <`https://orcid.org/0009-0003-2733-9362`>)

Maintainer Aristide Houndetoungan `<aristide.houndetoungan@ecn.ulaval.ca>`

Repository https://cran.r-universe.dev

Date/Publication 2025-06-19 15:10:02 UTC

RemoteUrl https://github.com/cran/QuantilePeer

RemoteRef HEAD

RemoteSha adec00efa1438a361ac90ce0e40a2604c2e63d89

Contents

`QuantilePeer-package` *The QuantilePeer package*

Description

The **QuantilePeer** package simulates and estimates peer effect models including the quantile-based specification (Houndetoungan, 2025 doi:10.3982/ECTA21048), and the models with Constant Elasticity of Substitution (CES)-based social norm (Boucher et al., 2024 doi:10.3982/ECTA21048).

Author(s)

Maintainer: Aristide Houndetoungan `<aristide.houndetoungan@ecn.ulaval.ca>` (ORCID)

References

Boucher, V., Rendall, M., Ushchev, P., & Zenou, Y. (2024). Toward a general theory of peer effects. Econometrica, 92(2), 543-565, doi:10.3982/ECTA21048.

Houndetoungan, A. (2025). Quantile peer effect models. arXiv preprint arXiv:2405.17290, doi:10.48550/arXiv.2506.12920.

Hyndman, R. J., & Fan, Y. (1996). Sample quantiles in statistical packages. The American Statistician, 50(4), 361-365, doi:10.1080/00031305.1996.10473566.

See Also

Useful links:

- `https://github.com/ahoundetoungan/QuantilePeer`
- Report bugs at `https://github.com/ahoundetoungan/QuantilePeer/issues`

`cespeer` *Estimation of CES-Based Peer Effects Models*

Description

`cespeer` estimates the CES-based peer effects model introduced by Boucher et al. (2024). See Details.

Usage

```
cespeer(
  formula,
  instrument,
  Glist,
  structural = FALSE,
  fixed.effects = FALSE,
  set.rho = NULL,
  grid.rho = seq(-400, 400, radius),
  radius = 5,
  tol = 1e-08,
  drop = NULL,
  compute.cov = TRUE,
  HAC = "iid",
  data
)
```

Arguments

`formula`	An object of class formula: a symbolic description of the model. `formula` should be specified as `y ~ x1 + x2`, where `y` is the outcome and `x1` and `x2` are control variables, which can include contextual variables such as averages or quantiles among peers.
`instrument`	An object of class formula indicating the excluded instrument. It should be specified as `~ z`, where `z` is the excluded instrument for the outcome. Following Boucher et al. (2024), it can be an OLS exogenous prediction of `y`. This prediction is used to compute instruments for the CES function of peer outcomes.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`structural`	A logical value indicating whether the reduced-form or structural specification should be estimated (see details).
`fixed.effects`	A logical value or string specifying whether the model includes subnet fixed effects. The fixed effects may differ between isolated and non-isolated nodes. Accepted values are `"no"` or `"FALSE"` (indicating no fixed effects), `"join"` or `TRUE` (indicating the same fixed effects for isolated and non-isolated nodes within each

	subnet), and `"separate"` (indicating different fixed effects for isolated and non-isolated nodes within each subnet). Note that `"join"` fixed effects are not applicable for structural models; `"join"` and `TRUE` are automatically converted to `"separate"`.
`set.rho`	A fixed value for the CES substitution parameter to estimate a constrained model. Given this value, the other parameters can be estimated.
`grid.rho`	A finite grid of values for the CES substitution parameter ρ (see Details). This grid is used to obtain the starting value and define the GMM weight. It is recommended to use a finely subdivided grid.
`radius`	The radius of the subset in which the estimate for ρ is determined. The subset is a segment centered at the optimal ρ found using `grid.rho`. For better numerical optimization performance, use a finely subdivided `grid.rho` and a small `radius`.
`tol`	A tolerance value used in the QR factorization to identify columns of explanatory variable and instrument matrices that ensure a full-rank matrix (see the qr function). The same tolerance is also used in the to minimize the concentrated GMM objective function (see optimise).
`drop`	A dummy vector of the same length as the sample, indicating whether an observation should be dropped. This can be used, for example, to remove false isolates or to estimate the model only on non-isolated agents. These observations cannot be directly removed from the network by the user because they may still be friends with other agents.
`compute.cov`	A logical value indicating whether the covariance matrix of the estimator should be computed.
`HAC`	A character string specifying the correlation structure among the idiosyncratic error terms for covariance computation. Options are `"iid"` for independent errors, `"hetero"` for heteroskedastic non-autocorrelated errors, and `"cluster"` for heteroskedastic errors with potential within-subnet correlation.
`data`	An optional data frame, list, or environment (or an object that can be coerced 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 `cespeer` is called.

Details

Let $\mathcal{N}$ be a set of n agents indexed by the integer $i \in [1, n]$. Agents are connected through a network characterized by an adjacency matrix $\mathbf{G} = [g_{ij}]$ of dimension $n \times n$, where $g_{ij} = 1$ if agent j is a friend of agent i, and $g_{ij} = 0$ otherwise. In weighted networks, g_{ij} can be a nonnegative variable (not necessarily binary) that measures the intensity of the outgoing link from i to j. The model can also accommodate such networks. Note that the network generally consists of multiple independent subnets (e.g., schools). The `Glist` argument is the list of subnets. In the case of a single subnet, `Glist` should be a list containing one matrix.

The reduced-form specification of the CES-based peer effects model is given by:

$$y_i = \lambda \left(\sum_{j=1}^{n} g_{ij} y_j^{\rho} \right)^{1/\rho} + \mathbf{x}_i' \beta + \varepsilon_i,$$

where ε_i is an idiosyncratic error term, λ captures the effect of the social norm $\left(\sum_{j=1}^{n} g_{ij} y_j^{\rho}\right)^{1/\rho}$, and β captures the effect of $\mathbf{x}_i$ on y_i. The parameter ρ determines the form of the social norm in the model.

- When $\rho > 1$, individuals are more sensitive to peers with high outcomes.
- When $\rho < 1$, individuals are more sensitive to peers with low outcomes.
- When $\rho = 1$, peer effects are uniform across peer outcome values.

The structural specification of the model differs for isolated and non-isolated individuals. For an **isolated** i, the specification is similar to a standard linear-in-means model without social interactions, given by:

$$y_i = \mathbf{x}_i'\beta + \varepsilon_i.$$

If node i is **non-isolated**, the specification is:

$$y_i = \lambda \left(\sum_{j=1}^{n} g_{ij} y_j^{\rho}\right)^{1/\rho} + (1 - \lambda_2)\mathbf{x}_i'\beta + \varepsilon_i,$$

where λ_2 determines whether preferences exhibit conformity or complementarity/substitution. Identification of β and λ_2 requires the network to include a sufficient number of isolated individuals.

Value

A list containing:

`model.info`	A list with information about the model, including the number of subnets, the number of observations, and other key details.
`gmm`	A list of GMM estimation results, including parameter estimates, the covariance matrix, and related statistics.
`first.search`	A list containing initial estimations on the grid of values for ρ.

References

Boucher, V., Rendall, M., Ushchev, P., & Zenou, Y. (2024). Toward a general theory of peer effects. Econometrica, 92(2), 543-565, doi:10.3982/ECTA21048.

See Also

`qpeer`, `linpeer`

Examples

```
set.seed(123)
ngr  <- 50  # Number of subnets
nvec <- rep(30, ngr)  # Size of subnets
n    <- sum(nvec)

### Simulating Data
```

```
## Network matrix
G <- lapply(1:ngr, function(z) {
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z], nvec[z])
  diag(Gz) <- 0
  # Adding isolated nodes (important for the structural model)
  niso <- sample(0:nvec[z], 1, prob = (nvec[z] + 1):1 / sum((nvec[z] + 1):1))
  if (niso > 0) {
    Gz[sample(1:nvec[z], niso), ] <- 0
  }
  # Row-normalization
  rs   <- rowSums(Gz); rs[rs == 0] <- 1
  Gz/rs
})

X   <- cbind(rnorm(n), rpois(n, 2))
l   <- 0.55
b   <- c(2, -0.5, 1)
rho <- -2
eps <- rnorm(n, 0, 0.4)

## Generating `y`
y <- cespeer.sim(formula = ~ X, Glist = G, rho = rho, lambda = l,
                 beta = b, epsilon = eps)$y

### Estimation
## Computing instruments
z <- fitted.values(lm(y ~ X))

## Reduced-form model
rest <- cespeer(formula = y ~ X, instrument = ~ z, Glist = G, fixed.effects = "yes",
                radius = 5, grid.rho = seq(-10, 10, 1))
summary(rest)

## Structural model
sest <- cespeer(formula = y ~ X, instrument = ~ z, Glist = G, fixed.effects = "yes",
                radius = 5, structural = TRUE, grid.rho = seq(-10, 10, 1))
summary(sest)

## Quantile model
z    <- qpeer.inst(formula = ~ X, Glist = G, tau = seq(0, 1, 0.1), max.distance = 2,
                   checkrank = TRUE)$instruments
qest <- qpeer(formula = y ~ X, excluded.instruments  = ~ z, Glist = G,
              fixed.effects = "yes", tau = seq(0, 1, 1/3), structural = TRUE)
summary(qest)
```

`cespeer.data` *Compute the CES Social Norm*

Description

`cespeer.data` computes the CES social norm, along with the first and second derivatives of the CES social norm with respect to the substitution parameter ρ.

Usage

```
cespeer.data(y, Glist, rho)
```

Arguments

`y`	A vector of outcomes used to compute the social norm.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`rho`	The CES substitution parameter.

Value

A four-column matrix with the following columns:

`y`	The outcome;
`ces(y, rho)`	The CES social norm;
`d[ces(y, rho)]`	The first derivative of the social norm;
`dd[ces(y, rho)]`	The second derivative of the social norm.

`cespeer.sim` *Simulating Peer Effect Models with a CES Social Norm*

Description

`cespeer.sim` simulates peer effect models with a Constant Elasticity of Substitution (CES) based social norm (Boucher et al., 2024).

Usage

```
cespeer.sim(
  formula,
  Glist,
  parms,
  rho,
  lambda,
  beta,
  epsilon,
  structural = FALSE,
  init,
  tol = 1e-10,
```

```
  maxit = 500,
  data
)
```

Arguments

`formula`	A formula object (formula): a symbolic description of the model. `formula` should be specified as, for example, `~ x1 + x2`, where `x1` and `x2` are control variables, which can include contextual variables such as averages or quantiles among peers.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`parms`	A vector defining the true values of $(\rho, \lambda', \beta')'$, where ρ is the substitution parameter of the CES function and λ is either the peer effect parameter for the reduced-form specification or a 2-vector with the first component being conformity peer effects and the second component representing total peer effects. The parameters ρ, λ, and β can also be specified separately using the arguments `rho`, `lambda`, and `beta` (see the Details section of `cespeer`).
`rho`	The true value of the substitution parameter of the CES function.
`lambda`	The true value of the peer effect parameter λ. It must include conformity and total peer effects for the structural model.
`beta`	The true value of the vector β.
`epsilon`	A vector of idiosyncratic error terms. If not specified, it will be simulated from a standard normal distribution (see the model specification in the Details section of `cespeer`).
`structural`	A logical value indicating whether simulations should be performed using the structural model. The default is the reduced-form model (see the Details section of `cespeer`).
`init`	An optional initial guess for the equilibrium.
`tol`	The tolerance value used in the Fixed Point Iteration Method to compute the outcome `y`. The process stops if the ℓ_1-distance between two consecutive values of `y` is less than `tol`.
`maxit`	The maximum number of iterations for the Fixed Point Iteration Method.
`data`	An optional data frame, list, or environment containing the model variables. If a variable is not found in `data`, it is retrieved from `environment(formula)`, typically the environment from which `cespeer.sim` is called.

Value

A list containing:

`y`	The simulated variable.
`epsilon`	The idiosyncratic error.
`init`	The initial guess.
`iteration`	The number of iterations before convergence.

References

Boucher, V., Rendall, M., Ushchev, P., & Zenou, Y. (2024). Toward a general theory of peer effects. *Econometrica, 92*(2), 543-565, doi:10.3982/ECTA21048.

See Also

`cespeer`, `qpeer.sim`

Examples

```
set.seed(123)
ngr  <- 50
nvec <- rep(30, ngr)
n    <- sum(nvec)
G    <- lapply(1:ngr, function(z){
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z])
  diag(Gz) <- 0
  Gz/rowSums(Gz) # Row-normalized network
})
tau  <- seq(0, 1, 0.25)
X    <- cbind(rnorm(n), rpois(n, 2))
l    <- 0.55
rho  <- 3
b    <- c(4, -0.5, 1)

out  <- cespeer.sim(formula = ~ X, Glist = G, rho = rho, lambda = l, beta = b)
summary(out$y)
out$iteration
```

`demean` *Demeaning Variables*

Description

demean demeans variables by subtracting the within-subnetwork average. In each subnetwork, this transformation can be performed separately for isolated and non-isolated nodes.

Usage

```
demean(X, Glist, separate = FALSE, drop = NULL)
```

Arguments

`X`	A matrix or vector to demean.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.

`separate`	A logical value specifying whether variables should be demeaned separately for isolated and non-isolated individuals. This is similar to setting `fixed.effects = "separate"` in `qpeer`.
`drop`	A logical vector of the same length as the sample, indicating whether an observation should be dropped. This can be used, for example, to remove false isolates or to estimate the model only on non-isolated agents. These observations cannot be directly removed from the network by the user, as they may still be connected to other agents.

Value

A matrix or vector with the same dimensions as `X`, containing the demeaned values.

`genpeer` *Estimating Peer Effects Models*

Description

`qpeer` estimates the quantile peer effect models introduced by Houndetoungan (2025). In the `linpeer` function, quantile peer variables are replaced with the average peer variable, and they can be replaced with other peer variables in the `genpeer` function.

Usage

```
genpeer(
  formula,
  excluded.instruments,
  endogenous.variables,
  Glist,
  data,
  estimator = "IV",
  structural = FALSE,
  drop = NULL,
  fixed.effects = FALSE,
  HAC = "iid",
  checkrank = FALSE,
  compute.cov = TRUE,
  tol = 1e-10
)

linpeer(
  formula,
  excluded.instruments,
  Glist,
  data,
  estimator = "IV",
  structural = FALSE,
```

```
  drop = NULL,
  fixed.effects = FALSE,
  HAC = "iid",
  checkrank = FALSE,
  compute.cov = TRUE,
  tol = 1e-10
)

qpeer(
  formula,
  excluded.instruments,
  Glist,
  tau,
  type = 7,
  data,
  estimator = "IV",
  structural = FALSE,
  fixed.effects = FALSE,
  HAC = "iid",
  checkrank = FALSE,
  drop = NULL,
  compute.cov = TRUE,
  tol = 1e-10
)
```

Arguments

`formula`	An object of class formula: a symbolic description of the model. `formula` should be specified as `y ~ x1 + x2`, where `y` is the outcome and `x1` and `x2` are control variables, which can include contextual variables such as averages or quantiles among peers.
`excluded.instruments`	An object of class formula to indicate excluded instruments. It should be specified as `~ z1 + z2`, where `z1` and `z2` are excluded instruments for the quantile peer outcomes.
`endogenous.variables`	An object of class formula that allows specifying endogenous variables. It is used to indicate the peer variables whose effects will be estimated. These can include average peer variables, quantile peer variables, or a combination of multiple variables. It should be specified as `~ y1 + y2`, where `y1` and `y2` are the endogenous peer variables.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`data`	An optional data frame, list, or environment (or an object that can be coerced 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 `qpeer` is called.

`estimator`	A character string specifying the estimator to be used. The available options are: `"IV"` for the standard instrumental variable estimator, `"gmm.identity"` for the GMM estimator with the identity matrix as the weight, `"gmm.optimal"` for the GMM estimator with the optimal weight matrix, `"JIVE"` for the Jackknife instrumental variable estimator, and `"JIVE2"` for the Type 2 Jackknife instrumental variable estimator.
`structural`	A logical value indicating whether the reduced-form or structural specification should be estimated (see Details).
`drop`	A dummy vector of the same length as the sample, indicating whether an observation should be dropped. This can be used, for example, to remove false isolates or to estimate the model only on non-isolated agents. These observations cannot be directly removed from the network by the user because they may still be friends with other agents.
`fixed.effects`	A logical value or string specifying whether the model includes subnet fixed effects. The fixed effects may differ between isolated and non-isolated nodes. Accepted values are `"no"` or `"FALSE"` (indicating no fixed effects), `"join"` or `TRUE` (indicating the same fixed effects for isolated and non-isolated nodes within each subnet), and `"separate"` (indicating different fixed effects for isolated and non-isolated nodes within each subnet). Note that `"join"` fixed effects are not applicable for structural models; `"join"` and `TRUE` are automatically converted to `"separate"`.
`HAC`	A character string specifying the correlation structure among the idiosyncratic error terms for covariance computation. Options are `"iid"` for independent errors, `"hetero"` for heteroskedastic non-autocorrelated errors, and `"cluster"` for heteroskedastic errors with potential within-subnet correlation.
`checkrank`	A logical value indicating whether the instrument matrix should be checked for full rank. If the matrix is not of full rank, unimportant columns will be removed to obtain a full-rank matrix.
`compute.cov`	A logical value indicating whether the covariance matrix of the estimator should be computed.
`tol`	A tolerance value used in the QR factorization to identify columns of explanatory variable and instrument matrices that ensure a full-rank matrix (see the qr function).
`tau`	A numeric vector specifying the quantile levels.
`type`	An integer between 1 and 9 selecting one of the nine quantile algorithms used to compute peer quantiles (see the quantile function).

Details

Let $\mathcal{N}$ be a set of n agents indexed by the integer $i \in [1, n]$. Agents are connected through a network that is characterized by an adjacency matrix $\mathbf{G} = [g_{ij}]$ of dimension $n \times n$, where $g_{ij} = 1$ if agent j is a friend of agent i, and $g_{ij} = 0$ otherwise. In weighted networks, g_{ij} can be a nonnegative variable (not necessarily binary) that measures the intensity of the outgoing link from i to j. The model can also accommodate such networks. Note that the network is generally constituted in many independent subnets (eg: schools). The `Glist` argument is the list of subnets. In the case of a single subnet, `Glist` will be a list containing one matrix.

Let $\mathcal{T}$ be a set of quantile levels. The reduced-form specification of quantile peer effect models is given by:

$$y_i = \sum_{\tau \in \mathcal{T}} \lambda_\tau q_{\tau,i}(\mathbf{y}_{-i}) + \mathbf{x}_i'\beta + \varepsilon_i,$$

where $\mathbf{y}_{-i} = (y_1, \dots, y_{i-1}, y_{i+1}, \dots, y_n)'$ is the vector of outcomes for other units, and $q_{\tau,i}(\mathbf{y}_{-i})$ is the sample τ-quantile of peer outcomes. The term ε_i is an idiosyncratic error term, λ_τ captures the effect of the τ-quantile of peer outcomes on y_i, and β captures the effect of $\mathbf{x}_i$ on y_i. For the definition of the sample τ-quantile, see Hyndman and Fan (1996). If the network matrix is weighted, the sample weighted quantile can be used, where the outcome for friend j of i is weighted by g_{ij}. It can be shown that the sample τ-quantile is a weighted average of two peer outcomes. For more details, see the quantile and `qpeer.instruments` functions.

The quantile $q_{\tau,i}(\mathbf{y}_{-i})$ can be replaced with the average peer variable in `linpeer` or with other measures in `genpeer` through the `endogenous.variables` argument. In `genpeer`, it is possible to specify multiple peer variables, such as male peer averages and female peer averages. Additionally, both quantiles and averages can be included (`genpeer` is general and encompasses `qpeer` and `linpeer`). See examples.

One issue in linear peer effect models is that individual preferences with conformity and complementarity/substitution lead to the same reduced form. However, it is possible to disentangle both types of preferences using isolated individuals (individuals without friends). The structural specification of the model differs between isolated and nonisolated individuals. For isolated i, the specification is similar to a standard linear-in-means model without social interactions, given by:

$$y_i = \mathbf{x}_i'\beta + \varepsilon_i.$$

If node i is non-isolated, the specification is given by:

$$y_i = \sum_{\tau \in \mathcal{T}} \lambda_\tau q_{\tau,i}(\mathbf{y}_{-i}) + (1 - \lambda_2)(\mathbf{x}_i'\beta + \varepsilon_i),$$

where λ_2 determines whether preferences exhibit conformity or complementarity/substitution. In general, $\lambda_2 > 0$ and this means that that preferences are conformist (anti-conformity may be possible in some models when $\lambda_2 < 0$). In contrast, when $\lambda_2 = 0$, there is complementarity/substitution between individuals depending on the signs of the λ_τ parameters. It is obvious that β and λ_2 can be identified only if the network includes enough isolated individuals.

Value

A list containing:

`model.info`	A list with information about the model, such as the number of subnets, number of observations, and other key details.
`gmm`	A list of GMM estimation results, including parameter estimates, the covariance matrix, and related statistics.
`data`	A list containing the outcome, outcome quantiles among peers, control variables, and excluded instruments used in the model.

References

Houndetoungan, A. (2025). Quantile peer effect models. arXiv preprint arXiv:2405.17290, doi:10.48550/arXiv.2506.12920.

Hyndman, R. J., & Fan, Y. (1996). Sample quantiles in statistical packages. The American Statistician, 50(4), 361-365, doi:10.1080/00031305.1996.10473566.

See Also

`qpeer.sim`, `qpeer.instruments`

Examples

```
set.seed(123)
ngr  <- 50  # Number of subnets
nvec <- rep(30, ngr)  # Size of subnets
n    <- sum(nvec)

### Simulating Data
## Network matrix
G <- lapply(1:ngr, function(z) {
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z], nvec[z])
  diag(Gz) <- 0
  # Adding isolated nodes (important for the structural model)
  niso <- sample(0:nvec[z], 1, prob = (nvec[z] + 1):1 / sum((nvec[z] + 1):1))
  if (niso > 0) {
    Gz[sample(1:nvec[z], niso), ] <- 0
  }
  Gz
})

tau <- seq(0, 1, 1/3)
X   <- cbind(rnorm(n), rpois(n, 2))
l   <- c(0.2, 0.15, 0.1, 0.2)
b   <- c(2, -0.5, 1)
eps <- rnorm(n, 0, 0.4)

## Generating `y`
y <- qpeer.sim(formula = ~ X, Glist = G, tau = tau, lambda = l,
               beta = b, epsilon = eps)$y

### Estimation
## Computing instruments
Z <- qpeer.inst(formula = ~ X, Glist = G, tau = seq(0, 1, 0.1),
                max.distance = 2, checkrank = TRUE)
Z <- Z$instruments

## Reduced-form model
rest <- qpeer(formula = y ~ X, excluded.instruments = ~ Z, Glist = G, tau = tau)
summary(rest)
summary(rest, diagnostic = TRUE)  # Summary with diagnostics
```

```
## Structural model
sest <- qpeer(formula = y ~ X, excluded.instruments = ~ Z, Glist = G, tau = tau,
              structural = TRUE)
summary(sest, diagnostic = TRUE)
# The lambda^* parameter is y_q (conformity) in the outputs.
# There is no conformity in the data, so the estimate will be approximately 0.

## Structural model with double fixed effects per subnet using optimal GMM
## and controlling for heteroskedasticity
sesto <- qpeer(formula = y ~ X, excluded.instruments = ~ Z, Glist = G, tau = tau,
               structural = TRUE, fixed.effects = "separate", HAC = "hetero",
               estimator = "gmm.optimal")
summary(sesto, diagnostic = TRUE)

## Average peer effect model
# Row-normalized network to compute instruments
Gnorm <- lapply(G, function(g) {
  d <- rowSums(g)
  d[d == 0] <- 1
  g / d
})

# GX and GGX
Gall <- Matrix::bdiag(Gnorm)
GX   <- as.matrix(Gall %*% X)
GGX  <- as.matrix(Gall %*% GX)

# Standard linear model
lpeer <- linpeer(formula = y ~ X + GX, excluded.instruments = ~ GGX, Glist = Gnorm)
summary(lpeer, diagnostic = TRUE)
# Note: The normalized network is used here by definition of the model.
# Contextual effects are also included (this is also possible for the quantile model).

# The standard model can also be structural
lpeers <- linpeer(formula = y ~ X + GX, excluded.instruments = ~ GGX, Glist = Gnorm,
                  structural = TRUE, fixed.effects = "separate")
summary(lpeers, diagnostic = TRUE)

## Estimation using `genpeer`
# Average peer variable computed manually and included as an endogenous variable
Gy     <- as.vector(Gall %*% y)
gpeer1 <- genpeer(formula = y ~ X + GX, excluded.instruments = ~ GGX,
                  endogenous.variables = ~ Gy, Glist = Gnorm, structural = TRUE,
                  fixed.effects = "separate")
summary(gpeer1, diagnostic = TRUE)

# Using both average peer variables and quantile peer variables as endogenous,
# or only the quantile peer variable
# Quantile peer `y`
qy <- qpeer.inst(formula = y ~ 1, Glist = G, tau = tau)
qy <- qy$qy

# Model estimation
```

```
gpeer2 <- genpeer(formula = y ~ X + GX, excluded.instruments = ~ GGX + Z,
                  endogenous.variables = ~ Gy + qy, Glist = Gnorm, structural = TRUE,
                  fixed.effects = "separate")
summary(gpeer2, diagnostic = TRUE)
```

`linpeer.sim` *Simulating Linear Peer Effect Models*

Description

`linpeer.sim` simulates linear peer effect models.

Usage

```
linpeer.sim(
  formula,
  Glist,
  parms,
  lambda,
  beta,
  epsilon,
  structural = FALSE,
  data
)
```

Arguments

`formula`	An object of class formula: a symbolic description of the model. `formula` should be specified as, for example, `~ x1 + x2`, where `x1` and `x2` are control variables, which can include contextual variables such as averages or quantiles among peers.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`parms`	A vector defining the true values of $(\lambda', \beta')'$, where λ is either the peer effect parameter for the reduced-form specification or a 2-vector with the first component being conformity peer effects and the second component representing total peer effects. The parameters λ and β can also be specified separately using the arguments `lambda`, and `beta`.
`lambda`	The true value of the vector λ.
`beta`	The true value of the vector β.
`epsilon`	A vector of idiosyncratic error terms. If not specified, it will be simulated from a standard normal distribution.
`structural`	A logical value indicating whether simulations should be performed using the structural model. The default is the reduced-form model (see the Details section of `qpeer`).

`data`	An optional data frame, list, or environment (or an object that can be coerced 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 `linpeer.sim` is called.

Value

A list containing:

`y`	The simulated variable.
`Gy`	the average of y among friends.

See Also

`qpeer`, `linpeer`

Examples

```
set.seed(123)
ngr  <- 50
nvec <- rep(30, ngr)
n    <- sum(nvec)
G    <- lapply(1:ngr, function(z){
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z])
  diag(Gz) <- 0
  Gz/rowSums(Gz) # Row-normalized network
})
X    <- cbind(rnorm(n), rpois(n, 2))
l    <- 0.5
b    <- c(2, -0.5, 1)

out  <- linpeer.sim(formula = ~ X, Glist = G, lambda = l, beta = b)
summary(out$y)
```

`print.qpeer.test` *Printing Specification Tests for Peer Effects Models*

Description

A print method for the class `qpeer.test`.

Usage

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

Arguments

`x`	an object of class `qpeer.test`
`...`	Further arguments passed to or from other methods.

Value

No return value, called for side effects

`qpeer.instruments` *Computing Instruments for Linear Models with Quantile Peer Effects*

Description

`qpeer.instruments` computes quantile peer variables.

Usage

```
qpeer.instruments(
  formula,
  Glist,
  tau,
  type = 7,
  data,
  max.distance = 1,
  checkrank = FALSE,
  tol = 1e-10
)

qpeer.instrument(
  formula,
  Glist,
  tau,
  type = 7,
  data,
  max.distance = 1,
  checkrank = FALSE
)

qpeer.inst(
  formula,
  Glist,
  tau,
  type = 7,
  data,
  max.distance = 1,
  checkrank = FALSE
)

qpeer.insts(
  formula,
  Glist,
```

```
  tau,
  type = 7,
  data,
  max.distance = 1,
  checkrank = FALSE
)
```

Arguments

`formula`	An object of class formula: a symbolic description of the model. The `formula` should be specified as, for example, `~ x1 + x2` or `y ~ x1 + x2`, where `x1` and `x2` are variables for which the quantiles will be computed and `y` is the dependent variable. If `y` is specified, then the quantiles of `x1` and `x2` are computed by ranking observations according to the values of `y` (see details).
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`tau`	The vector of quantile levels.
`type`	An integer between 1 and 9 selecting one of the nine quantile algorithms used to compute peer quantiles (see the quantile function).
`data`	An optional data frame, list, or environment (or an object that can be coerced 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 `qpeer.instruments` is called.
`max.distance`	The maximum network distance of friends to consider in computing instruments.
`checkrank`	A logical value indicating whether the instrument matrix should be checked for full rank. If the matrix is not of full rank, unimportant columns will be removed to obtain a full-rank matrix.
`tol`	A tolerance value used in the QR factorization to identify columns that ensure a full-rank matrix (see the qr function).

Details

The sample quantile is computed as a weighted average of two peer outcomes (see Hyndman and Fan, 1996). Specifically:

$$q_{\tau,i}(x_{-i}) = (1-\omega_i)x_{i,(\pi_i)} + \omega_i x_{i,(\pi_i+1)},$$

where $x_{i,(1)}, x_{i,(2)}, x_{i,(3)}, \ldots$ are the order statistics of the outcome within i's peers, and $q_{\tau,i}(x_{-i})$ represents the sample τ-quantile of the outcome within i's peer group. If `y` is specified, then the ranks π_i and the weights ω_i for the variables in `X` are determined based on `y`. The network matrices in `Glist` can be weighted or unweighted. If weighted, the sample weighted quantile is computed, where the outcome for friend j of i is weighted by g_{ij}, the (i, j) entry of the network matrix.

Value

A matrix including quantile peer variables

A list containing:

`qy`	Quantiles of peer variable y.
`instruments`	Matrix of instruments.
`index`	The indices of the two peers whose weighted average gives the quantile.
`weight`	The weights of the two peers whose weighted average gives the quantile.

References

Hyndman, R. J., & Fan, Y. (1996). Sample quantiles in statistical packages. The American Statistician, 50(4), 361-365, doi:10.1080/00031305.1996.10473566.

See Also

`qpeer`, `qpeer.sim`, `linpeer`

Examples

```
ngr  <- 50
nvec <- rep(30, ngr)
n    <- sum(nvec)
G    <- lapply(1:ngr, function(z){
  Gz <- matrix(rbinom(sum(nvec[z]*(nvec[z] - 1)), 1, 0.3), nvec[z])
  diag(Gz) <- 0
  Gz
})
tau  <- seq(0, 1, 0.25)
X    <- cbind(rnorm(n), rpois(n, 2))
l    <- c(0.2, 0.1, 0.05, 0.1, 0.2)
b    <- c(2, -0.5, 1)
y    <- qpeer.sim(formula = ~X, Glist = G, tau = tau, lambda = l, beta = b)$y
Inst <- qpeer.instruments(formula = ~ X, Glist = G, tau = tau, max.distance = 2)$instruments
summary(Inst)
```

`qpeer.sim` *Simulating Linear Models with Quantile Peer Effects*

Description

`qpeer.sim` simulates the quantile peer effect models developed by Houndetoungan (2025).

Usage

```
qpeer.sim(
  formula,
  Glist,
  tau,
  parms,
  lambda,
  beta,
  epsilon,
  structural = FALSE,
  init,
  type = 7,
  tol = 1e-10,
  maxit = 500,
  details = TRUE,
  data
)
```

Arguments

`formula`	An object of class formula: a symbolic description of the model. `formula` should be specified as, for example, `~ x1 + x2`, where `x1` and `x2` are control variables, which can include contextual variables such as averages or quantiles among peers.
`Glist`	The adjacency matrix. For networks consisting of multiple subnets (e.g., schools), `Glist` must be a list of subnets, with the `m`-th element being an $n_m \times n_m$ adjacency matrix, where n_m is the number of nodes in the `m`-th subnet.
`tau`	The vector of quantile levels.
`parms`	A vector defining the true values of $(\lambda', \beta')'$, where λ is a vector of λ_τ for each quantile level τ. The parameters λ and β can also be specified separately using the arguments `lambda` and `beta`. For the structural model, $\lambda = (\lambda_2, \lambda_{\tau_1}, \lambda_{\tau_2}, \dots)'$ (see the Details section of `qpeer`).
`lambda`	The true value of the vector λ.
`beta`	The true value of the vector β.
`epsilon`	A vector of idiosyncratic error terms. If not specified, it will be simulated from a standard normal distribution (see the model specification in the Details section of `qpeer`).
`structural`	A logical value indicating whether simulations should be performed using the structural model. The default is the reduced-form model (see the Details section of `qpeer`).
`init`	An optional initial guess for the equilibrium.
`type`	An integer between 1 and 9 selecting one of the nine quantile algorithms used to compute peer quantiles (see the quantile function).
`tol`	The tolerance value used in the Fixed Point Iteration Method to compute the outcome `y`. The process stops if the ℓ_1-distance between two consecutive values of `y` is less than `tol`.

`maxit`	The maximum number of iterations for the Fixed Point Iteration Method.
`details`	A logical value indicating whether to save the indices and weights of the two peers whose weighted average determines the quantile.
`data`	An optional data frame, list, or environment (or an object that can be coerced 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 `sim.qpeer` is called.

Value

A list containing:

`y`	The simulated variable.
`qy`	Quantiles of the simulated variable among peers.
`epsilon`	The idiosyncratic error.
`index`	The indices of the two peers whose weighted average gives the quantile.
`weight`	The weights of the two peers whose weighted average gives the quantile.
`iteration`	The number of iterations before convergence.

References

Houndetoungan, A. (2025). Quantile peer effect models. arXiv preprint arXiv:2405.17290, doi:10.48550/arXiv.2506.12920.

Hyndman, R. J., & Fan, Y. (1996). Sample quantiles in statistical packages. The American Statistician, 50(4), 361-365, doi:10.1080/00031305.1996.10473566.

See Also

`qpeer`, `qpeer.instruments`

Examples

```
set.seed(123)
ngr  <- 50
nvec <- rep(30, ngr)
n    <- sum(nvec)
G    <- lapply(1:ngr, function(z){
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z])
  diag(Gz) <- 0
  Gz
})
tau  <- seq(0, 1, 0.25)
X    <- cbind(rnorm(n), rpois(n, 2))
l    <- c(0.2, 0.1, 0.05, 0.1, 0.2)
b    <- c(2, -0.5, 1)

out  <- qpeer.sim(formula = ~ X, Glist = G, tau = tau, lambda = l, beta = b)
summary(out$y)
out$iteration
```

`qpeer.test` *Specification Tests for Peer Effects Models*

Description

`qpeer.test` performs specification tests on peer effects models. These include monotonicity tests on quantile peer effects, as well as tests for instrument validity when an alternative set of instruments is available.

Usage

```
qpeer.test(
  model1,
  model2 = NULL,
  which,
  full = FALSE,
  boot = 10000,
  maxit = 1e+06,
  eps_f = 1e-09,
  eps_g = 1e-09
)
```

Arguments

`model1, model2`	Objects of class `qpeer`, `linpeer`, or `genpeer`.
`which`	A character string indicating the type of test to be implemented. The value must be one of `"uniform"`, `"increasing"`, `"decreasing"`, `"wald"`, `"sargan"`, and `"encompassing"` (see Details).
`full`	A Boolean indicating whether the parameters associated with the exogenous variables should be compared. This is used for tests that compare two competing parameter sets.
`boot`	An integer indicating the number of bootstrap replications to use for computing `p-values` in the `"increasing"` and `"decreasing"` tests.
`maxit, eps_f, eps_g`	Control parameters for the `optim_lbfgs` solver used to optimize the objective function in the `"increasing"` and `"decreasing"` tests (see Kodde and Palm, 1986). The `optim_lbfgs` function is provided by the **RcppNumerical** package and is based on the `L-BFGS` method.

Details

The monotonicity tests evaluate whether the quantile peer effects λ_τ are constant, increasing, or decreasing. In this case, `model1` must be an object of class `qpeer`, and the `which` argument specifies the null hypothesis: `"uniform"`, `"increasing"`, or `"decreasing"`. For the `"uniform"` test, a standard Wald test is performed. For the `"increasing"` and `"decreasing"` tests, the procedure follows Kodde and Palm (1986).

The instrument validity tests assess whether a second set of instruments Z_2 is valid, assuming that a baseline set Z_1 is valid. In this case, both `model1` and `model2` must be objects of class `qpeer`, `linpeer`, or `genpeer`. The test compares the estimates obtained using each instrument set. If Z_2 nests Z_1, it is recommended to compare the overidentification statistics from both estimations (see Hayashi, 2000, Proposition 3.7). If Z_2 does not nest Z_1, the estimates themselves are compared. To compare the overidentification statistics, set the `which` argument to `"sargan"`. To compare the estimates directly, set the `which` argument to `"wald"`.

Given two competing models, it is possible to test whether one is significantly worse using an encompassing test by setting `which` to `"encompassing"`. The null hypothesis is that `model1` is not worse.

Value

A list containing:

`test`	A vector or matrix containing the test statistics, degrees of freedom, and p-values.
`lambda`	Peer effect estimates from tests based on a single model (monotonicity tests).
`diff.theta`	Differences in peer effect estimates from tests based on two models (endogeneity and encompassing tests).
`delta`	The estimate of δ for the encompassing test.
`which`	The value of `which` returned by the function.
`boot`	The value of `boot` returned by the function.

References

Hayashi, F. (2000). *Econometrics*. Princeton University Press.

Kodde, D. A., & Palm, F. C. (1986). Wald criteria for jointly testing equality and inequality restrictions. *Econometrica*, 54(5), 1243–1248.

Examples

```
set.seed(123)
ngr  <- 50  # Number of subnets
nvec <- rep(30, ngr)  # Size of subnets
n    <- sum(nvec)

### Simulating Data
## Network matrix
G <- lapply(1:ngr, function(z) {
  Gz <- matrix(rbinom(nvec[z]^2, 1, 0.3), nvec[z], nvec[z])
  diag(Gz) <- 0
  # Adding isolated nodes (important for the structural model)
  niso <- sample(0:nvec[z], 1, prob = (nvec[z] + 1):1 / sum((nvec[z] + 1):1))
  if (niso > 0) {
```

```
    Gz[sample(1:nvec[z], niso), ] <- 0
  }
  Gz
})

tau <- seq(0, 1, 1/3)
X   <- cbind(rnorm(n), rpois(n, 2))
l   <- c(0.2, 0.15, 0.1, 0.2)
b   <- c(2, -0.5, 1)
eps <- rnorm(n, 0, 0.4)

## Generating `y`
y <- qpeer.sim(formula = ~ X, Glist = G, tau = tau, lambda = l,
               beta = b, epsilon = eps)$y

### Estimation
## Computing instruments
Z1 <- qpeer.inst(formula = ~ X, Glist = G, tau = seq(0, 1, 0.1),
                 max.distance = 2, checkrank = TRUE)$instruments
Z2 <- qpeer.inst(formula = y ~ X, Glist = G, tau = seq(0, 1, 0.1),
                 max.distance = 2, checkrank = TRUE)$instruments

## Reduced-form model
rest1 <- qpeer(formula = y ~ X, excluded.instruments = ~ Z1, Glist = G, tau = tau)
summary(rest1, diagnostic = TRUE)
rest2 <- qpeer(formula = y ~ X, excluded.instruments = ~ Z1 + Z2, Glist = G, tau = tau)
summary(rest2, diagnostic = TRUE)

qpeer.test(model1 = rest1, which = "increasing")
qpeer.test(model1 = rest1, which = "decreasing")
qpeer.test(model1 = rest1, model2 = rest2, which = "sargan")

#' A model with a mispecified tau
rest3 <- qpeer(formula = y ~ X, excluded.instruments = ~ Z1 + Z2, Glist = G, tau = c(0, 1))
summary(rest3)
#' Test is rest3 is worse than rest1
qpeer.test(model1 = rest3, model2 = rest1, which = "encompassing")
```

`summary.cespeer` *Summary for the Estimation of CES-based Peer Effects Models*

Description

Summary and print methods for the class `cespeer`.

Usage

```
## S3 method for class 'cespeer'
summary(object, fullparameters = TRUE, ...)
```

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

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

Arguments

`object`	An object of class `cespeer`.
`fullparameters`	A logical value indicating whether all parameters should be summarized (may be useful for the structural model).
`...`	Further arguments passed to or from other methods.
`x`	An object of class `summary.cespeer` or `cespeer`.

Value

A list containing:

`model.info`	A list with information about the model, such as the number of subnets, number of observations, and other key details.
`coefficients`	A summary of the estimates, standard errors, and p-values.
`gmm`	A list of GMM estimation results, including parameter estimates, the covariance matrix, and related statistics.

`summary.genpeer` *Summary for the Estimation of Quantile Peer Effects Models*

Description

Summary and print methods for the class `qpeer`.

Usage

```
## S3 method for class 'genpeer'
summary(
  object,
  fullparameters = TRUE,
  diagnostic = FALSE,
  diagnostics = FALSE,
  ...
)

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

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

## S3 method for class 'linpeer'
summary(
  object,
  fullparameters = TRUE,
  diagnostic = FALSE,
  diagnostics = FALSE,
  ...
)

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

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

## S3 method for class 'qpeer'
summary(
  object,
  fullparameters = TRUE,
  diagnostic = FALSE,
  diagnostics = FALSE,
  ...
)

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

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

Arguments

`object`	An object of class `qpeer`.
`fullparameters`	A logical value indicating whether all parameters should be summarized (may be useful for the structural model).
`diagnostics, diagnostic`	Logical. Should diagnostic tests for the instrumental-variable regression be performed? These include an F-test of the first-stage regression for weak instruments, a Wu-Hausman test for endogeneity, and a Hansen's J-test for overidentifying restrictions (only if there are more instruments than regressors).
`...`	Further arguments passed to or from other methods.
`x`	An object of class `summary.qpeer` or `qpeer`.

Value

A list containing:

`model.info`	A list with information about the model, such as the number of subnets, number of observations, and other key details.
`coefficients`	A summary of the estimates, standard errors, and p-values.
`diagnostics`	A summary of the diagnostic tests for the instrumental-variable regression if requested.
`KP.cv`	Critical values for the Kleibergen–Paap Wald test (5% level).
`gmm`	A list of GMM estimation results, including parameter estimates, the covariance matrix, and related statistics.

Index