Package 'econet'

September 2, 2020

Type Package

Title Estimation of Parameter-Dependent Network Centrality Measures

Version 0.1.92

Description
Provides methods for estimating parameter-dependent network centrality measures with linear-in-means models. Both non linear least squares and maximum likelihood estimators are implemented. The methods allow for both link and node heterogeneity in network effects, endogenous network formation and the presence of unconnected nodes. The routines also compare the explanatory power of parameter-dependent network centrality measures with those of standard measures of network centrality. Benefits and features of the 'econet' package are illustrated using data from Battaglini and Patacchini (2018) and Battaglini, Patacchini, and Leone Sciabolazza (2020). For additional details, see the vignette.

Depends R (>= 3.5.0)

Imports bbmle, igraph, intergraph, Matrix, MASS, minpack.lm, sna, spatstat.utils, stats, utils, plyr, dplyr, parallel, doParallel, progressr, foreach

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Suggests testthat, R.rsp

VignetteBuilder R.rsp

NeedsCompilation no

Author Marco Battaglini [aut] (<https://orcid.org/0000-0001-9690-0721>), Valerio Leone Sciabolazza [aut, cre] (<https://orcid.org/0000-0003-2537-3084>), Eleonora Patacchini [aut] (<https://orcid.org/0000-0002-3510-2969>), Sida Peng [aut] (<https://orcid.org/0000-0002-2151-0523>)

Maintainer Valerio Leone Sciabolazza <valerio.leonesciabolazza@uniparthenope.it>

Repository CRAN

Date/Publication 2020-09-02 11:20:02 UTC

R topics documented:

`a_db_alumni` *Dataset from Battaglini, Patacchini (2018)*

Description

Dataset from Battaglini, Patacchini (2018)

Usage

```
data("a_db_alumni")
```

Format

an object of class `data.frame` with 2176 rows and 48 columns

id unique congressman id.

PAC PAC contributions to a member of Congress (in US dollars).

isolate dummy variable taking value of 1 if the congressman did not graduate from the same institution within eight years with any other congressman.

S1-S65 list of school dummies. Each one refers to a different school. The dummy takes value one if the congressman attended the given school.

S_other dummy variable taking value of 1 if the congressman attended any other school not previously listed.

party dummy variable taking value 1 if the congressman is a Democrat, and 0 if the congressman is a Republican.

gender dummy variable taking value of 1 if the congressman is female.

nchair dummy variable taking value of 1 if the congressman is a chair of at least one committee.

sen number of consecutive years in the House of Representatives.

margin_b dummy variable taking value of 1 if the election margin of victory is less than 5%.

dw distance to the center in terms of ideology measured using the absolute value of the first dimension of the dw-nominate score created by McCarty et al. (1997).

majority dummy variable taking value of 1 if the member of Congress is a member of the party holding the majority of the seats in the House of Representatives.

relcom dummy variable taking value of 1 if the congressman is member of a powerful committee (Appropriations, Budget, Rules and Ways and Means).

time categorical variable. It takes 1 if the record refers to the 109th congress, 2 if the record refers to the 110th congress, 3 if the record refers to the 111th congress, 4 if the record refers to the 112th congress, 5 if the record refers to the 113th congress.

weights for each congressman *i*, the value is equal to the inverse of the variance of PAC contributions received by all congressmen in *i*'s State of election.

References

Battaglini M., E. Patacchini (2018), "Influencing Connected Legislators", Journal of Political Economy, forthcoming.
McCarty, N. M., K. T. Poole, and H. Rosenthal (1997), "Income redistribution and the realignment of American politics", AEIpress.

`a_G_alumni_111` *Column-normalized adjacency matrix representing the alumni network of the 111th U.S. Congress.*

Description

Column-normalized adjacency matrix representing the alumni network of the 111th U.S. Congress.

Usage

```
data("a_G_alumni_111")
```

Format

an object of class `Matrix` with 426 rows and 426 columns

References

Battaglini M., E. Patacchini (2018), "Influencing Connected Legislators", Journal of Political Economy, forthcoming.

boot *boot: Bootstrap residuals with cross-sectional dependence*

Description

boot: Bootstrap residuals with cross-sectional dependence

Usage

```
## S3 method for class 'econet'
boot(
  object,
  hypothesis = c("lim", "het", "het_l", "het_r", "par", "par_split_with",
    "par_split_btw", "par_split_with_btw"),
  group = NULL,
  niter,
  weights = FALSE,
  parallel = FALSE,
  cl,
  ...
)
```

Arguments

object	an object of class econet obtained using 'NLLS' estimation.
hypothesis	string. One of c("lim","het","het_l","het_r","par","par_split_with","par_split_btw","pa
group	NULL or vector of positive integers specifying the indices for resampling within groups.
niter	number of required iterations.
weights	logical. It is TRUE if the object object was estimated using weights, and FALSE otherwise. Default FALSE.
parallel	logical. It is TRUE if the user wants to make use of parallelization. Default is FALSE. Observe that this option is still in its beta version and it has been tested only for the Windows platform.
cl	numeric. Number of cores to be used for parallelization.
...	additional parameters

Details

For additional details, see the vignette. Warning: This function is available only when net_dep is run with estimation == "NLLS"

Value

a numeric vector containing bootstrapped standard errors (see Anselin, 1990). If the procedure is not feasible, it returns a vector of NAs.

References

Anselin, L., 1990, "Some robust approach to testing and estimation in spatial econometrics", Regional Science and Urban Economics, 20, 141-163.

See Also

net_dep

Examples

```
# Load data
data("db_cosponsor")
data("G_alumni_111")
db_model_B <- db_cosponsor
G_model_B <- G_cosponsor_111
G_exclusion_restriction <- G_alumni_111
are_factors <- c("party", "gender", "nchair")
db_model_B[are_factors] <- lapply(db_model_B[are_factors], factor)

# Specify formula
f_model_B <- formula("les ~gender + party + nchair")

# Specify starting values
starting <- c(alpha = 0.23952,
              beta_gender1 = -0.22024,
              beta_party1 = 0.42947,
              beta_nchair1 = 3.09615,
              phi = 0.40038,
              unobservables = 0.07714)

# object Linear-in-means model
lim_model_B <- net_dep(formula = f_model_B, data = db_model_B,
                       G = G_model_B, model = "model_B", estimation = "NLLS",
                       hypothesis = "lim", endogeneity = TRUE, correction = "heckman",
                       first_step = "standard",
                       exclusion_restriction = G_exclusion_restriction,
                       start.val = starting)
# Bootstrap
# Warning: this may take a very long time to run.
# Decrease the number of iterations to reduce runtime.
# If you run econet on a Windows platform, you can try to set the
# argument parallel = TRUE. However note that this option is still
# in its beta version.
boot_lim_estimate <- boot(object = lim_model_B, hypothesis = "lim",
                          group = NULL, niter = 10, weights = FALSE)
boot_lim_estimate
```

db_alumni_test *Dataset from Battaglini, Patacchini (2018)*

Description

Dataset from Battaglini, Patacchini (2018)

Usage

```
data("db_alumni_test")
```

Format

Data set for runtime executions. An object of class `data.frame` with 42 rows and 48 columns

id unique congressman id.

PAC PAC contributions to a member of Congress (in US dollars).

isolate dummy variable taking value of 1 if the congressman did not graduate from the same institution within eight years with any other congressman.

S1-S65 list of school dummies. Each one refers to a different school. The dummy takes value one if the congressman attended the given school.

S_other dummy variable taking value of 1 if the congressman attended any other school not previously listed.

party dummy variable taking value 1 if the congressman is a Democrat, and 0 if the congressman is a Republican.

gender dummy variable taking value of 1 if the congressman is female.

nchair dummy variable taking value of 1 if the congressman is a chair of at least one committee.

sen number of consecutive years in the House of Representatives.

margin_b dummy variable taking value of 1 if the election margin of victory is less than 5%.

dw distance to the center in terms of ideology measured using the absolute value of the first dimension of the dw-nominate score created by McCarty et al. (1997).

majority dummy variable taking value of 1 if the member of Congress is a member of the party holding the majority of the seats in the House of Representatives.

relcom dummy variable taking value of 1 if the congressman is member of a powerful committee (Appropriations, Budget, Rules and Ways and Means).

time categorical variable. It takes 1 if the record refers to the 109th congress, 2 if the record refers to the 110th congress, 3 if the record refers to the 111th congress, 4 if the record refers to the 112th congress, 5 if the record refers to the 113th congress.

weights for each congressman *i*, the value is equal to the inverse of the variance of PAC contributions received by all congressmen in *i*'s State of election.

db_cosponsor *Dataset from Battaglini, Leone Sciabolazza, Patacchini (2018)*

Description

Dataset from Battaglini, Leone Sciabolazza, Patacchini (2018)

Usage

```
data("db_cosponsor")
```

Format

an object of class `data.frame` with 2176 rows and 48 columns

id unique congressman id.

time categorical variable. It takes 1 if the record refers to the 109th congress, 2 if the record refers to the 110th congress, 3 if the record refers to the 111th congress, 4 if the record refers to the 112th congress, 5 if the record refers to the 113th congress.

party dummy variable taking value 1 if the congressman is a Democrat, and 0 if the congressman is a Republican.

gender dummy variable taking value of 1 if the congressman is female.

nchair dummy variable taking value of 1 if the congressman is a chair of at least one committee.

margin election Margin of Victory of the member of Congress.

dw distance to the center in terms of ideology measured using the absolute value of the first dimension of the dw-nominate score created by McCarty et al. (1997).

les weighted average of the following the number of a congressman's sponsored bill that were introduced, received any action in committee and beyond committee, passed the House, and became law. It differentially weights commemorative, substantive and significant legislation. Created by Volden C. and Wiseman A. E. (2014).

majority dummy variable taking value of 1 if the member of Congress is a member of the party holding the majority of the seats in the House of Representatives.

state_leg dummy variable taking value of 1 if the member of Congress served in state legislature.

state_leg_prof Squire's (1992) index of state professionalism relative to Congress.

speaker dummy variable taking value of 1 if the member of Congress is speaker of the House.

seniority number of consecutive years in the House of Representatives.

maj_leader dummy variable taking value of 1 if the member of Congress is member of the majority party leadership, as reported by the Almanac of American Politics.

min_leader dummy variable taking value of 1 if the member of Congress is member of the minority party leadership, as reported by the Almanac of American Politics.

nowhite dummy variable taking value of 1 if the member of Congress is Afro-American or Latino, and 0 otherwise.

H.102-251 list of committee dummies. Each one refers to a different committee. The dummy takes value of 1 if the congressman served in the given commitee.

weights for each congressman *i*, the value is equal to the inverse of the variance of LES of all congressmen in *i*'s State of election.

References

Battaglini M., V. Leone Sciabolazza, E. Patacchini (2018), "Effectiveness of Connected Legislators", Mimeo.
McCarty, N. M., K. T. Poole, and H. Rosenthal (1997), "Income redistribution and the realignment of American politics", AEIpress.
Squire, P. (1992), "Legislative professionalization and membership diversity in state legislatures", Legislative Studies Quarterly, 17(1) 69-79.
Volden, C., and A. E. Wiseman (2014), "Legislative Effectiveness in the United States Congress The Lawmakers", Cambridge University Press.

`G_alumni_111` *Column-normalized adjacency matrix representing the alumni network of the 111th U.S. Congress.*

Description

Column-normalized adjacency matrix representing the alumni network of the 111th U.S. Congress.

Usage

```
data("G_alumni_111")
```

Format

an object of class `Matrix` with 426 rows and 426 columns

References

Battaglini M., V. Leone Sciabolazza, E. Patacchini (2018), "Effectiveness of Connected Legislators", Mimeo.

G_cosponsor_111 *Column-normalized adjacency matrix representing the cosponsorship network of the 111th U.S. Congress.*

Description

Column-normalized adjacency matrix representing the cosponsorship network of the 111th U.S. Congress.

Usage

```
data("G_cosponsor_111")
```

Format

an object of class `Matrix` with 439 rows and 439 columns

References

Battaglini M., V. Leone Sciabolazza, E. Patacchini (2018), "Effectiveness of Connected Legislators", Mimeo.

G_model_A_test *Column-normalized adjacency matrix representing a subsample of the alumni network of the 111th U.S. Congress.*

Description

Column-normalized adjacency matrix representing a subsample of the alumni network of the 111th U.S. Congress.

Usage

```
data("G_model_A_test")
```

Format

Data set for runtime executions. An object of class `Matrix` with 42 rows and 42 columns

References

Battaglini M., E. Patacchini (2018), "Influencing Connected Legislators", Journal of Political Economy, forthcoming.

horse_race *Compare the explanatory power of parameter.dependent network centrality measures with those of standard measures of network centrality.*

Description

Compare the explanatory power of parameter.dependent network centrality measures with those of standard measures of network centrality.

Usage

```
horse_race(
  formula = formula(),
 centralities = c("indegree", "outdegree", "degree", "betweenness", "incloseness",
    "outcloseness", "closeness", "eigenvector"),
  directed = FALSE,
  weighted = FALSE,
  normalization = FALSE,
  data = list(),
  unobservables = list(),
  G = list(),
  model = c("model_A", "model_B"),
  estimation = c("NLLS", "MLE"),
  endogeneity = FALSE,
  first_step = NULL,
  data_first_step = NULL,
  exclusion_restriction = NULL,
  start.val = NULL,
  to_weight = NULL,
  time_fixed_effect = NULL,
  mle_controls = NULL
)
```

Arguments

formula	an object of class formula: a symbolic description of the model to be fitted. The constant (i.e. intercept) and the autogressive parameter needs not to be specified.
centralities	at least one of c("indegree","outdegree","degree","betweenness","incloseness", "outcloseness","closeness","eigenvector").
directed	logical. TRUE if the social network is directed, FALSE otherwise.
weighted	logical. TRUE if the social network is weighted, FALSE otherwise.
normalization	Default is NULL. Alternatively, it can be set to c("bygraph","bycomponent", "bymaxcomponent","bymaxgraph"). See details.
data	an object of class data.frame containing the variables in the model. If data are longitudinal, observations must be ordered by time period and then by individual.

unobservables — a numeric vector used to obtain an unbiased estimate of the parameter.dependent centrality when the network is endogenous. See details.

G — an object of class Matrix representing the social network. Row and column names must be specified and match the order of the observations in data.

model — string. One of c("model_A","model_B"). See details.

estimation — string. One of c("NLLS","MLE"). They are used to implement respectively a non-linear least square and a maximum likelihood estimator.

endogeneity — logical. Default is FALSE. If TRUE, net_dep implements a two-step correction procedure to control for the endogeneity of the network.

first_step — Default is NULL. If endogeneity = TRUE, it requires to specify one of c("standard", "fe","shortest","coauthors","degree"). See details.

data_first_step — an optional object of class data.frame. If provided, it is used to implement the first step of the estimation when endogeneity = TRUE.

exclusion_restriction — an object of class Matrix representing the exogenous matrix used to instrument the endogenous social network, if unobservables is non-NULL. Row and column names must be specified and match the order of the observations in data.

start.val — an optional list containing the starting values for the estimations. Object names must match the names provided in formula. It is also required to specify the value of both the constant and the decay parameter(s).

to_weight — an optional vector of weights to be used in the fitting process to indicate that different observations have different variances. Should be NULL or a numeric vector. If non-NULL, weighted non-linear least squares (if estimation = "NLLS") or weighted maximum likelihood (if estimation = "MLE") is estimated.

time_fixed_effect — an optional string. It indicates the name of the time index used in formula. It is used for models with longitudinal data.

mle_controls — a list allowing the user to set upper and lower bounds for control variables in MLE estimation and the variance for the ML estimator.

Details

A number of different normalization are available to the user:

- bygraph and bycomponent are used to divide *degree* and *closeness* centrality by $n-1$, and *betweenness* centrality by $(n-1)*(n-2)$ if directed = TRUE, or by $(n-1)*(n-2)/2$ if directed = FALSE. In the former case (i.e. bygraph), *n* is equal to the number of nodes in the network In the latter case (i.e. bycomponent), *n* is equal to the number of nodes of the component in which the node is embedded.
- bymaxgraph and bymaxcomponent are used to divide *degree*, *betweenness* and *closeness* centrality by the maximum value of the centrality of the network (bymaxgraph) or component (bymaxcomponent) in which the node is embedded.

If the network is endogenous, the user is required to run separately net_dep and extract from the resulting object the vector of unobservables necessary for obtaining an unbiased estimate of the

parameter.dependent centrality. This vector can be passed through the argument `unobservables`. If `endogeneity = TRUE`, a two-step estimation is implemented to control for network endogeneity. The argument `first_step` is used to control for the specification of the first-step model, e.g.:

- `first_step = "standard"` is used when agents' connection are predicted by the differences in their characteristics (i.e. those on the right hand side of `formula`), and an `exclusion_restriction`: i.e., their connections in a different network.
- `first_step = "fe"` adds individual fixed effects to the `standard` model, as in Graham (2017).
- `first_step = "shortest"` adds to the `standard` model, the shortest distance between *i* and *j*, excluding the link between *i* and *j* itself, as in Fafchamps et al (2010).
- `first_step = "coauthor"` adds to the `standard` model, the number of shared connections between *i* and *j*, as in Graham (2015).
- `first_step = "degree"` adds to the `standard` model, the difference in the degree centrality of *i* and *j*.

For additional details, see the vignette.

Value

A list of two objects:

- A list of estimates, each one setting the decay parameter to zero, and adding one of the `centralities` to the specification of `formula`. The last object adds to `formula` all the selected `centralities` and the decay parameter is set different from zero.
- An object of class `data.frame` containing the computed centrality measures.
- A list of first-step estimations used to correct the effect of centrality measures when the network is endogenous.

References

Battaglini M., V. Leone Sciabolazza, E. Patacchini, S. Peng (2020), "Econet: An R package for the Estimation of parameter-dependent centrality measures", Mimeo.

See Also

`net_dep`

Examples

```
# Load data
data("db_cosponsor")
data("G_alumni_111")
db_model_B <- db_cosponsor
G_model_B <- G_cosponsor_111
G_exclusion_restriction <- G_alumni_111
are_factors <- c("party", "gender", "nchair")
db_model_B[are_factors] <- lapply(db_model_B[are_factors], factor)
```

```
# Specify formula
f_model_B <- formula("les ~gender + party + nchair")

# Specify starting values
starting <- c(alpha = 0.214094,
              beta_gender1 = -0.212706,
              beta_party1 = 0.478518,
              beta_nchair1 = 3.09234,
              beta_betweenness = 7.06287e-05,
              phi = 0.344787)

# Fit model
horse_model_B <- horse_race(formula = f_model_B,
               centralities = "betweenness",
               directed = TRUE, weighted = TRUE,
               data = db_model_B, G = G_model_B,
               model = "model_B", estimation = "NLLS",
               start.val = starting)

# Store and print results
summary(horse_model_B)
summary(horse_model_B, centrality = "betweenness")
horse_model_B$centrality

# WARNING, This toy example is provided only for runtime execution.
# Please refer to previous examples for sensible calculations.
data("db_alumni_test")
data("G_model_A_test")
db_model <- db_alumni_test
G_model <- G_model_A_test
f_model <- formula("les ~ dw")
horse_model_test <- horse_race(formula = f_model, centralities = "betweenness",
                            directed = TRUE, weighted = FALSE, normalization = NULL,
                            data = db_model, unobservables = NULL, G = G_model,
                            model = "model_A", estimation = "NLLS",
                            start.val = c(alpha = -0.31055275,
                                          beta_dw = 1.50666982,
                                          beta_betweenness = 0.09666742,
                                          phi = 16.13035695))
summary(horse_model_test)
```

`net_dep` *Implement a number of modifications to the linear-in-means model to obtain different weighted versions of Katz-Bonacich centrality.*

Description

Implement a number of modifications to the linear-in-means model to obtain different weighted versions of Katz-Bonacich centrality.

Usage

```
net_dep(
  formula = formula(),
  data = list(),
  G = list(),
  model = c("model_A", "model_B"),
  estimation = c("NLLS", "MLE"),
  hypothesis = c("lim", "het", "het_l", "het_r", "par", "par_split_with",
    "par_split_btw", "par_split_with_btw"),
  endogeneity = FALSE,
  correction = NULL,
  first_step = NULL,
  z = NULL,
  data_first_step = NULL,
  exclusion_restriction = NULL,
  start.val = NULL,
  to_weight = NULL,
  time_fixed_effect = NULL,
  ind_fixed_effect = NULL,
  mle_controls = NULL,
  kappa = NULL
)
```

Arguments

formula	an object of class formula: a symbolic description of the model to be fitted. The constant (i.e. intercept) and the autogressive parameter needs not to be specified.
data	an object of class data.frame containing the variables in the model. If data are longitudinal, observations must be ordered by time period and then by individual.
G	an object of class Matrix representing the social network. Row and column names must be specified and match the order of the observations in data.
model	string. One of c("model_A","model_B"). See details.
estimation	string. One of c("NLLS","MLE"). They are used to implement respectively a non-linear least square and a Maximum Likelihood estimator.
hypothesis	string. One of c("lim","het","het_l","het_r","par","par_split_with","par_split_btw","par See details.
endogeneity	logical. Default is FALSE. If TRUE, net_dep implements a two-step correction procedure to control for the endogeneity of the network.
correction	Default is NULL. If endogeneity = TRUE, it is required to specify if the main regression should use an instrumental variable ("iv") or Heckman ("heckman") approach.
first_step	Default is NULL. If endogeneity = TRUE, it requires to specify one of c("standard" "fe","shortest","coauthors","degree"). See details.

z numeric vector. It specifies the source of heterogeneity for peer effects when hypothesis is equal to "het", "het_l", or "het_r". Alternatively, it specifies the groups in which the network should be partitioned when hypothesis is equal to "par", "par_split_with", "par_split_btw", or "par_split_with_btw"). See details.

data_first_step an optional object of class data.frame. If provided, it is used to implement the first step of the estimation when endogeneity = TRUE.

exclusion_restriction an object of class Matrix representing the exogenous matrix used to instrument the endogenous social network, if endogeneity = TRUE. Row and column names must be specified and match the order of the observations in data.

start.val an optional list containing the starting values for the estimations. Object names must match the names provided in formula. It is also required to specify the value of both the constant and the decay parameter(s). See details.

to_weight an optional vector of weights to be used in the fitting process to indicate that different observations have different variances. Should be NULL or a numeric vector. If non-NULL, it can be used to fit a weighted non-linear least squares (estimation = "NLLS").

time_fixed_effect an optional string. It indicates the name of the time index used in formula. It is used for models with longitudinal data.

ind_fixed_effect an optional vector. Default is NULL. It indicates the name of the individual index contained in the data. If provided, individual fixed effects are automatically added to the formula of the main equation. If endogeneity = TRUE, the field first_step is overridden, and automatically set equal to "fe". It is used for models with longitudinal data.

mle_controls a list allowing the user to set upper and lower bounds for control variables in MLE estimation and the variance for the ML estimator. See details.

kappa a normalization level with default equals 1.

Details

Agent's parameter-dependent centrality is obtained as a function of

- the agent's characteristics and the performance of its socially connected peers, as in Battaglini, Leone Sciabolazza, Patacchini (2020), if model = "model_B";
- the performance of its socially connected peers, as in Battaglini, Patacchini (2018), if model = "model_A".

Peer effects are assumed to be homogenous if hypothesis = "lim". They are assumed to be heterogenous by setting:

- hypothesis = "het", when peers' performance is susceptible to agent's characteristics and model = "model_A".
- hypothesis = "het_l", when peers' performance is susceptible to agent's characteristics and model = "model_B".

- hypothesis = "het_r", when agent's performance is susceptible to peers' characteristics and model = "model_B".
- hypothesis = "par", when model = "model_B", if the network is formed by interactions between and within two different groups.
- hypothesis = "par_split_with", when model = "model_B", if the network is formed by interactions between and within two different groups, and interactions within each group are different from the other.
- hypothesis = "par_split_btw", when model = "model_B", if the network is formed by interactions between and within two different groups, and interactions between groups are different according to their direction.
- hypothesis = "par_split_with_btw", when model = "model_B", if the network is formed by interactions between and within two different groups, interactions within each group are different from the other, and interactions between groups are different according to their direction.

When hypothesis is equal to "het", "het_l", or "het_r", the argument z is used to specify the source of heterogeneity: i.e. the attribute affecting the ability of the agent to influence or be influenced by peers. When hypothesis is equal to "par", "par_split_with", "par_split_btw", or "par_split_with_btw" the argument "z" is used to partition observations in two groups: e.g. the generic element *i* of vector z takes the value 1 if agent *i* is member of the first group, and it takes 2 otherwise.

If endogeneity = TRUE, a two-step estimation is implemented to control for network endogeneity. The argument first_step is used to control for the specification of the first-step model, e.g.:

- first_step = "standard" is used when agents' connection are predicted by the differences in their characteristics (i.e. those on the right hand side of formula), and an exclusion_restriction: i.e., their connections in a different network.
- first_step = "fe" adds to the standard model, individual fixed effects, as in Graham (2017).
- first_step = "shortest" adds to the standard model, the shortest distance between *i* and *j*, excluding the link between *i* and *j* itself, as in Fafchamps et al (2010).
- first_step = "coauthor" adds to the standard model, the number of shared connections between *i* and *j*, as in Graham (2015).
- first_step = "degree" adds to the standard model, the difference in the degree centrality of *i* and *j*.

The argument start.val is used to specify starting estimates. This can be done with a named list. If a factor is present, a value for each treatment contrast must be provided. Labels of treatment contrasts must be assigned following R model design standards: e.g., a number is appended to contrast names as in contrasts().
The starting value referring to the intercept (constant) must be labelled as "alpha". The label(s) for decay parameter(s) must be:

- "phi", if hypothesis="lim" or hypothesis="het"
- "theta_0","theta_1", if hypothesis="het_l"
- "eta_0","eta_1", if hypothesis="het_r"
- "phi_within","phi_between", if hypothesis="par"

- "phi_within_0","phi_within_1","phi_between", if hypothesis="par_split_with"
- "phi_within","phi_between_0","phi_between_1", if hypothesis="par_split_btw"
- "phi_within_0","phi_within_1","phi_between_0","phi_between_1", if hypothesis="par_split_with_btw"

The interaction term when hypothesis="het" must be labelled "gamma". The label to be used for unobservables when endogeneity = TRUE is "unobservables". When estimation = "MLE", it is required to set also the starting value for the variance of the ML estimator. This should be labelled as "sigma".
The argument mle_controls takes a list of two objects. The first is a named numeric vector used to set upper and lower bounds for control variables. The second object is a vector used to set upper (first value) and lower (second value) bounds for the variance of the Maximum Likelihood estimator.
Names in mle_controls must be equal to those used in start.val. For additional details, see the vignette.

Value

A list of three objects: i) Estimates of the main regression; ii) The vector of agents' parameter-dependent centrality; iii) Estimates of the first-step regression (if endogeneity = TRUE)

References

Battaglini M., E. Patacchini (2018), "Influencing Connected Legislators," Journal of Political Economy, 126(6): 2277-2322.
Battaglini M., V. Leone Sciabolazza, E. Patacchini (2020), "Effectiveness of Connected Legislators," American Journal of Political Science, forthcoming.
Battaglini M., V. Leone Sciabolazza, E. Patacchini, S. Peng (2020), "Econet: An R package for the Estimation of parameter-dependent centrality measures", Mimeo.
Fafchamps, M., M. J. Leij and S. Goyal (2010), "Matching and network effects," Journal of the European Economic Association, 8(1): 203-231.
Graham B. (2015), "Methods of identification in social networks," Annual Review of Economics, 7, 465 - 485.
Graham B. (2017), "An econometric model of network formation with degree heterogeneity," Econometrica 85 (4), 1033 - 1063.

Examples

```
# Model A

# Load data
data("a_db_alumni")
data("a_G_alumni_111")
db_model_A <- a_db_alumni
G_model_A <- a_G_alumni_111
are_factors <- c("party", "gender", "nchair", "isolate")
db_model_A[are_factors] <- lapply(db_model_A[are_factors] ,factor)
db_model_A$PAC <- db_model_A$PAC/1e+06
```

```
# Specify formula
f_model_A <- formula("PAC ~ gender + party + nchair + isolate")

# Specify starting values
starting <- c(alpha = 0.47325,
              beta_gender1 = -0.26991,
              beta_party1 = 0.55883,
              beta_nchair1 = -0.17409,
              beta_isolate1 = 0.18813,
              phi = 0.21440)

# Fit Linear-in-means model
lim_model_A <- net_dep(formula = f_model_A, data = db_model_A,
                       G = G_model_A, model = "model_A", estimation = "NLLS",
                       hypothesis = "lim", start.val = starting)

summary(lim_model_A)
lim_model_A$centrality

# Test Heterogeneity

# Heterogeneous factor
z <- as.numeric(as.character(db_model_A$gender))

# Specify formula
f_het_model_A <- formula("PAC ~ party + nchair + isolate")

# Specify starting values
starting <- c(alpha = 0.44835,
              beta_party1 = 0.56004,
              beta_nchair1 = -0.16349,
              beta_isolate1 = 0.21011,
              beta_z = -0.26015,
              phi = 0.34212,
              gamma = -0.49960)

# Fit model
het_model_A <- net_dep(formula = f_het_model_A, data = db_model_A,
                       G = G_model_A, model = "model_A", estimation = "NLLS",
                       hypothesis = "het", z = z, start.val = starting)

summary(het_model_A)
het_model_A$centrality

# Model B

# Load data
data("db_cosponsor")
data("G_alumni_111")
db_model_B <- db_cosponsor
G_model_B <- G_cosponsor_111
G_exclusion_restriction <- G_alumni_111
are_factors <- c("party", "gender", "nchair")
```

```
db_model_B[are_factors] <- lapply(db_model_B[are_factors], factor)

# Specify formula
f_model_B <- formula("les ~ gender + party + nchair")

# Specify starting values
starting <- c(alpha = 0.23952,
              beta_gender1 = -0.22024,
              beta_party1 = 0.42947,
              beta_nchair1 = 3.09615,
              phi = 0.40038,
              unobservables = 0.07714)

# Fit Linear-in-means model
lim_model_B <- net_dep(formula = f_model_B, data = db_model_B,
                       G = G_model_B, model = "model_B", estimation = "NLLS",
                       hypothesis = "lim", endogeneity = TRUE,
                       correction = "heckman", first_step = "standard",
                       exclusion_restriction = G_exclusion_restriction,
                       start.val = starting)

summary(lim_model_B)
lim_model_B$centrality
summary(lim_model_B, print = "first.step")

# Test Heterogeneity

# Heterogeneous factor (node -level)
z <- as.numeric(as.character(db_model_B$gender))

# Specify formula
f_het_model_B <- formula("les ~ party + nchair")

# Specify starting values
starting <- c(alpha = 0.23952,
              beta_party1 = 0.42947,
              beta_nchair1 = 3.09615,
              beta_z = -0.12749,
              theta_0 = 0.42588,
              theta_1 = 0.08007)

# Fit model
het_model_B_l <- net_dep(formula = f_het_model_B,
                         data = db_model_B,
                         G = G_model_B, model = "model_B", estimation = "NLLS",
                         hypothesis = "het_l", z = z, start.val = starting)

# Store and print results
summary(het_model_B_l)
het_model_B_l$centrality

# Specify starting values
starting <- c(alpha = 0.04717,
```

```
              beta_party1 = 0.51713,
              beta_nchair1 = 3.12683,
              beta_z = 0.01975,
              eta_0 = 1.02789,
              eta_1 = 2.71825)

# Fit model
het_model_B_r <- net_dep(formula = f_het_model_B,
                         data = db_model_B,
                         G = G_model_B, model = "model_B", estimation = "NLLS",
                         hypothesis = "het_r", z = z, start.val = starting)

# Store and print results
summary(het_model_B_r)
het_model_B_r$centrality

# Heterogeneous factor (edge -level)
z <- as.numeric(as.character(db_model_B$party))

# Specify starting values
starting <- c(alpha = 0.242486,
              beta_gender1 = -0.229895,
              beta_party1 = 0.42848,
              beta_nchair1 = 3.0959,
              phi_within  = 0.396371,
              phi_between = 0.414135)

# Fit model
party_model_B <- net_dep(formula = f_model_B, data = db_model_B,
                         G = G_model_B, model = "model_B",
                         estimation = "NLLS", hypothesis = "par",
                         z = z, start.val = starting)

# Store and print results
summary(party_model_B)
party_model_B$centrality

# WARNING, This toy example is provided only for runtime execution.
# Please refer to previous examples for sensible calculations.
data("db_alumni_test")
data("G_model_A_test")
db_model_A <- db_alumni_test
G_model_A <- G_model_A_test
f_model_A <- formula("les ~ dw")
lim_model_A_test <- net_dep(formula = f_model_A, data = db_model_A,
                        G = G_model_A, model = "model_A", estimation = "NLLS",
                        hypothesis = "lim", start.val = c(alpha = 0.09030594,
                                                          beta_dw = 1.21401940,
                                                          phi = 1.47140647))
summary(lim_model_A_test)
```

quantify *quantify: quantification of marginal effects in linear-in-means models.*

Description

quantify: quantification of marginal effects in linear-in-means models.

Usage

```
## S3 method for class 'econet'
quantify(object, ...)
```

Arguments

object	first object in the list of outcomes returned by net_dep (available only if the argument model is set to "model_B").
...	other arguments

Details

quantify returns marginal effects for net_dep objects when model = "model_B" and hypothesis = "lim". For additional details, see the vignette

Value

an object of class data.frame listing direct and indirect variable effects (mean, standard deviation, max, min).

See Also

net_dep

Examples

```
# Load data
data("db_cosponsor")
data("G_alumni_111")
db_model_B <- db_cosponsor
G_model_B <- G_cosponsor_111
G_exclusion_restriction <- G_alumni_111
are_factors <- c("party", "gender", "nchair")
db_model_B[are_factors] <- lapply(db_model_B[are_factors], factor)

# Specify formula
f_model_B <- formula("les ~gender + party + nchair")

# Specify starting values
starting <- c(alpha = 0.23952,
```

```
               beta_gender1 = -0.22024,
               beta_party1 = 0.42947,
               beta_nchair1 = 3.09615,
               phi = 0.40038,
               unobservables = 0.07714)

# Fit Linear-in-means model
lim_model_B <- net_dep(formula = f_model_B, data = db_model_B,
                       G = G_model_B, model = "model_B", estimation = "NLLS",
                       hypothesis = "lim", endogeneity = TRUE, correction = "heckman",
                       first_step = "standard",
                       exclusion_restriction = G_exclusion_restriction,
                       start.val = starting)
quantify(lim_model_B)

# WARNING, This toy example is provided only for runtime execution.
# Please refer to previous examples for sensible calculations.
data("db_alumni_test")
data("G_model_A_test")
db_model <- db_alumni_test
G_model <- G_model_A_test
f_model <- formula("les ~ dw")
lim_model_test <- net_dep(formula = f_model, data = db_model,
                       G = G_model, model = "model_B", estimation = "NLLS",
                       hypothesis = "lim", start.val = c(alpha = 0.4553039,
                                                         beta_dw = -0.7514903,
                                                         phi = 1.6170539))
quantify(lim_model_test)
```

Index