Package ‘NetMix’

March 1, 2021

Type Package
Title Dynamic Mixed-Membership Network Regression Model
Version 0.2.0
Date 2021-03-01
Encoding UTF-8
Author Santiago Olivella [aut, cre],
    Adeline Lo [aut, cre],
    Tyler Pratt [aut, cre],
    Kosuke Imai [aut, cre]
Maintainer Santiago Olivella <olivella@unc.edu>
Description Stochastic collapsed variational inference on mixed-
    membership stochastic blockmodel for networks,
    incorporating node-level predictors of mixed-membership vectors, as well as
    dyad-level predictors. For networks observed over time, the model defines a hidden
    Markov process that allows the effects of node-level predictors to evolve in discrete,
    historical periods. In addition, the package offers a variety of utilities for
    exploring results of estimation, including tools for conducting posterior
    predictive checks of goodness-of-fit and several plotting functions. The package
    implements methods described in Olivella, Pratt and Imai (2019) 'Dynamic Stochastic
    Blockmodel Regression for Social Networks: Application to International Conflicts',

BugReports https://github.com/solivella/NetMix/issues

NeedsCompilation yes
License GPL (>= 2)
Depends R (>= 3.5.0)
SystemRequirements C++11
Suggests ergm (>= 3.9.4), ggplot2 (>= 3.1.1), network (>= 1.13),
    scales (>= 1.0.0)
Imports clue (>= 0.3-58), graphics (>= 3.5.2), grDevices (>= 3.5.2),
    gtools (>= 3.8.1), igraph (>= 1.2.4.1), lda (>= 1.4.2), Matrix
    (>= 1.2-15), MASS (>= 7.3-51.4), methods (>= 3.5.2), poisbinom
    (>= 1.0.1), Rcpp (>= 1.0.2), stats (>= 3.5.2), utils (>= 3.5.2)
Description

These are various utilities and generic methods used by the main package function.

Usage

approxB(y, d_id, pi_mat, directed = TRUE)

getZ(pi_mat)

alphaLBound(par, tot_nodes, c_t, x_t, s_mat, t_id, var_beta, mu_beta)

alphaGrad(par, tot_nodes, c_t, x_t, s_mat, t_id, var_beta, mu_beta)

.cbind.fill(...)

.scaleVars(x, keep_const = TRUE)

.transf_muvar(orig, is_var, is_array, des.mat, nblock = NULL, nstate = NULL)
ApproxB

bar.legend(colPalette, range)

mpower(mat, p)

findPerm(block_list, target_mat = NULL, use_perms = TRUE)

transf(mat)

compute.alpha(X, beta)

vcovBeta(
  all_phi,
  beta_coef,
  n.sim,
  n.blk,
  n.hmm,
  n.nodes,
  n.periods,
  mu.beta,
  var.beta,
  est_kappa,
  t_id_n,
  X
)

e.pi(alpha_list, kappa, C_mat = NULL)

initPi(
  soc_mats,
  dyads,
  edges,
  nodes_pp,
  dyads_pp,
  n.blocks,
  periods,
  directed,
  ctrl
)

Arguments

y, d_id, pi_mat, directed
  Internal arguments for blockmodel approximation.

par
  Vector of parameter values.

tot_nodes
  Integer vector; total number of nodes each node interacts with.

c_t
  Integer matrix; samples from Poisson-Binomial counts of a node instantiating a group.
x_t  Numeric matrix; transposed monadic design matrices.
s_mat  Integer matrix; Samples of HMM states by time period.
t_id  Integer vector; for each node, what time-period is it observed in? zero-indexed.
mu_beta, var_beta  Numeric arrays; prior mean and variances of monadic coefficients.
...  Numeric vectors; vectors of potentially different length to be cbind-ed.
x, keep_const  Internal arguments for matrix scaling.
orig  Object to be transformed.
is_var  Boolean. Is the object to be transformed a variance term?
is_array  Boolean. Is the object to be transformed an array?
des.mat  Numeric matrix. Design matrix corresponding to transformed object.
nblock  Number of groups in model, defaults to NULL.
nstate  Number of hidden Markov states in model, defaults to NULL.
colPalette  A function produced by colorRamp.
range  The range of values to label the legend.
mat  Numeric matrix.
p  Numeric scalar; power to raise matrix to.
block_list  List of matrices; each element is a square, numeric matrix that defines a block-model,
target_mat  Numeric matrix; reference blockmodel that those in block_list should be aligned to. Optional, defaults to NULL.
use_perms  Boolean; should all row/column permutations be explored when realigning matrices? defaults to TRUE.
X  Numeric matrix; design matrix of monadic predictors.
beta  Numeric array; array of coefficients associated with monadic predictors. It of dimensions Nr. Predictors by Nr. of Blocks by Nr. of HMM states.
all_phi, beta_coef, n.sim, n.blk, n.hmm, n.nodes, n.periods, mu.beta, var.beta, est_kappa, t_id_n, Additional internal arguments for covariance estimation.
alpa_list  List of mixed-membership parameter matrices.
kappa  Numeric matrix; matrix of marginal HMM state probabilities.
C_mat  Numeric matrix; matrix of posterior counts of block instantiations per node.
soc_mats, dyads, edges, nodes_pp, dyads_pp, n.blocks, periods, ctrl  Internal arguments for MM computation.

Details

These functions are meant for internal use only.
**Value**

See individual return section for each function:

- `.cbind.fill` Matrix of `cbind`ed elements in ..., with missing values in each vector filled with NA.
- `.mpower` Matrix; the result of raising mat to the p power.
- `.findPerm` List of permuted blockmodel matrices.
- `.transf` Matrix with transformed mixed-membership vectors along its rows, s.t. no element is equal to 0.0 or 1.0.
- `.compute.alpha` List of predicted alpha matrices, one element per HMM state.
- `.e.pi` Matrix of predicted mixed-membership vectors along its rows, with expectation computed over marginal distribution over HMM states for each time period.
- `.missing` Transformed data.frame with missing values list-wise deleted, or expanded with missing indicator variables.
- `.createSocioB` List of sociomatrices.
- `.vertboot2` List of bootstrapped sociomatrices.

**Author(s)**

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

---

**coeff.mmsbm**

**Extract Regression Coefficients for a Fitted mmsbm Object**

**Description**

Extract Regression Coefficients for a Fitted mmsbm Object

**Usage**

```r
## S3 method for class 'mmsbm'
coef(object, param = "All", ...)
```

**Arguments**

- `object` An object of class mmsbm, a result of a call to mmsbm
- `param` Character string, which set of parameters should the vcov be extracted for? One of "MonadCoef", "DyadCoef" or "All" (the default).
- `...` Currently ignored

**Value**

For param="DyadCoef", a numeric vector. For param="MonadCoef", an array with HMM states along the third dimension. For param="All", named list of individual return components.
**Author(s)**
Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

**Examples**
```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                        ~ School,
                        senderID = "Lawyer1",
                        receiverID = "Lawyer2",
                        nodeID = "Lawyer",
                        data.dyad = lazega_dyadic,
                        data.monad = lazega_monadic,
                        n.blocks = 2,
                        mmsbm.control = list(seed = 123,
                                             conv_tol = 1e-2,
                                             hessian = FALSE))

coef(lazega_mmsbm, "MonadCoef")
```

---

**covFX**

*Generate estimated monadic covariate effects for estimated mmsbm model*

---

**Description**

The function estimates the effect of a shift in monadic covariate values on the probability of edge formation in the network.

**Usage**

`covFX(fm, cov, shift, max.val = FALSE)`

**Arguments**

- `fm`: An object of class mmsbm, a result of a call to mmsbm.
- `cov`: Character string identifying the monadic covariate to be shifted.
- `shift`: Numeric value specifying the desired increase or decrease in the monadic covariate. The monadic predictor will be shifted by this value for all nodes and time periods.
- `max.val`: An optional numeric value specifying the maximum possible value for the monadic covariate.
Value

List with named components:

**Overall Avg. Effect** Overall average effect of the covariate shift on the predicted probability of edge formation.

**Avg. Effect by Time** Vector of average effects of the covariate shift on the predicted probability of edge formation for each time period.

**Avg. Effect by Node** Vector of average effects of the covariate shift on the predicted probability of edge formation for each node.

**Avg. Effect by Dyad** Vector of average effects of the covariate shift on the predicted probability of edge formation for each node dyad.

**Avg. Effect Dyad-Time** Vector of estimated effects of the covariate shift on the predicted probability of edge formation for each node dyad-time unit.

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                        ~ Age,
                        senderID = "Lawyer1",
                        receiverID = "Lawyer2",
                        nodeID = "Lawyer",
                        data.dyad = lazega_dyadic,
                        data.monad = lazega_monadic,
                        n.blocks = 2,
                        mmsbm.control = list(seed = 123,
                                             conv_tol = 1e-2,
                                             hessian = FALSE))

## Compute effect of decreasing every lawyers' age by 10 years
fx_list <- covFX(lazega_mmsbm, cov = "Age", shift = -10)
fX_list[["Overall Avg. Effect of Age"]]
```
gof

Posterior predictive checks using structural network characteristics

Description
The function generates a variety of plots that serve as posterior predictive checks on the goodness of fit of a fitted mmsbm object.

Usage

gof(x, ...)

## S3 method for class 'mmsbm'
gof(
  x,
  gof_stat = c("Geodesics", "Degree"),
  level = 0.95,
  samples = 50,
  new.data.dyad = NULL,
  new.data.monad = NULL,
  seed = NULL,
  ...
)

Arguments

x
An object of class mmsbm, a result of a call to mmsbm.

... Currently ignored.

gof_stat

level
Double. Level of credible interval for posterior predictive distribution around structural quantities of interest.

samples
Integer. Number of sampled networks from model’s posterior predictive using simulate.mmsbm.

new.data.dyad
See simulate.mmsbm. Enables out-of-sample checking.

new.data.monad
See simulate.mmsbm. Enables out-of-sample checking.

seed
See simulate.mmsbm.

Details

Goodness of fit of network models has typically been established by evaluating how the structural characteristics of predicted networks compare to those of the observed network. When estimated in a Bayesian framework, this approach is equivalent to conducting posterior predictive checks on these structural quantities of interest. When new.data.dyad and/or new.data.monad are passed that are
different from those used in estimation, this is equivalent to conducting posterior predictive checks out-of-sample.

The set of structural features used to determine goodness of fit is somewhat arbitrary, and chosen mostly to incorporate various first order, second order, and (to the extent possible) third-order characteristics of the network. "Geodesics" focuses on the distribution over observed and predicted geodesic distances between nodes; "Indegree" and "Outdegree" focuses on the distribution over incoming and outgoing connections per node; "3-motifs" focus on a distribution over possible connectivity patterns between triads (i.e. the triadic census); "Dyad Shared Partners" focuses on the distribution over the number of shared partners between any two dyads; "Edge Shared Partners" is similarly defined, but w.r.t. edges, rather than dyads; and finally "Incoming K-stars" focuses on a frequency distribution over stars with k=1,... spokes.

Obtaining samples of the last three structural features can be very computationally expensive, and is discouraged on networks with more than 50 nodes.

Value

A ggplot object.

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")

## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                         senderID = "Lawyer1",
                         receiverID = "Lawyer2",
                         nodeID = "Lawyer",
                         data.dyad = lazega_dyadic,
                         data.monad = lazega_monadic,
                         n.blocks = 2,
                         mmsbm.control = list(seed = 123,
                                              conv_tol = 1e-2,
                                              hessian = FALSE))

## Plot observed (red) and simulated (gray) distributions over
## indegrees
## (typically a larger number of samples would be taken)
## (strictly requires ggplot2)

gof(lazega_mmsbm, gof_stat = "Indegree", samples = 2)
```
Identify nodes with most frequent membership in latent groups

**Description**

The function lists the nodes (optionally, node-time periods) that most frequently instantiate membership in each latent group.

**Usage**

```r
## S3 method for class 'mmsbm'
head(x, n = 6, t = NULL, node = TRUE, t.correct = FALSE, ...)
```

**Arguments**

- `x`: An object of class `mmsbm`, a result of a call to `mmsbm`.
- `n`: Numeric or integer; specifies how many units will be identified for each group.
- `t`: Optional vector of time periods to be used for assessing latent group membership.
- `node`: Logical; indicates whether latent group memberships should be averaged at the node level. If FALSE, the function returns the node-time period units with highest estimated membership in each latent group.
- `t.correct`: Logical; indicates whether latent group memberships should be corrected for temporal trends. If TRUE, the function returns the node-time period units with highest estimated membership in each latent group.
- `...`: Currently ignored

**Value**

List of length `n.blocks`. Each entry contains a sorted vector of average latent membership probabilities of length `n`.

**Author(s)**

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

**Examples**

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
set.seed(123)
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                      ~ School + Practice + Status,
                      ...)
lazega_dyadic

senderID = "Lawyer1",
receiverID = "Lawyer2",
nodeID = "Lawyer",
data.dyad = lazega_dyadic,
data.monad = lazega_monadic,
n.blocks = 2,
mmsbm.control = list(seed = 123,
                        conv_tol = 1e-2,
                        hessian = FALSE))

## Show top 6 lawyers in each estimated latent block
head(lazega_mmsbm)

---

**lazega_dyadic**

*Dyadic predictors in the Lazega friendship network (Lazega 2001)*.

**Description**


**Usage**

data(lazega_dyadic)

**Format**

A data frame with 5041 rows and 4 variables:

- **Lawyer1, Lawyer2** lawyer ID, corresponding to identifiers common to those in lazega_monadic; numeric
- **SocializeWith** value of edge in network; binary
- **Coworkers** are the corresponding lawyers in the same office? boolean

**Source**


**References**

lazega_monadic  Monadic predictors in the Lazega friendship network (Lazega 2001).

Description

Usage
data(lazega_monadic)

Format
A data frame with 71 rows and 7 variables:

- **Lawyer**: lawyer ID, corresponding to identifiers common to those in lazega_dyadic; numeric
- **Age**: age, in years; numeric
- **Gender**: 1=man; 2=woman; factor
- **School**: 1=harvard, yale; 2=ucon; 3= other; factor
- **Practice**: 1=litigation; 2=corporate; factor
- **Seniority**: time in the firm, in years; numeric
- **Status**: 1=partner; 2=associate; factor

Source

https://github.com/Z-co/networkdata/blob/master/networkdata/data/lazega.rda

mmsbm  Dynamic mixed-membership stochastic blockmodel with covariates

Description
The function estimates a dynamic mixed-membership stochastic blockmodel that incorporates covariates.
Usage

```r
mmsbm(
  formula.dyad,
  formula.monad = ~1,
  senderID,
  receiverID,
  nodeID = NULL,
  timeID = NULL,
  data.dyad,
  data.monad = NULL,
  n.blocks,
  n.hmmstates = 1,
  directed = TRUE,
  mmsbm.control = list()
)
```

Arguments

- `formula.dyad`: A formula object. The variable in `data.dyad` that contains binary edges should be used as a LHS, and any dyadic predictors can be included on the RHS (when no dyadic covariates are available, use `y ~ 1`). Same syntax as a `glm` formula.
- `formula.monad`: An optional formula object. LHS is ignored. RHS contains names of nodal attributes found in `data.monad`.
- `senderID`: Character string. Quoted name of the variable in `data.dyad` identifying the sender node. For undirected networks, the variable simply contains name of first node in dyad. Cannot contain special character "@".
- `receiverID`: Character string. Quoted name of the variable in `data.dyad` identifying the receiver node. For undirected networks, the variable simply contains name of second node in dyad. Cannot contain special character "@".
- `nodeID`: Character string. Quoted name of the variable in `data.monad` identifying a node in either `data.dyad[, senderID]` or `data.dyad[, senderID]`. If not `NULL`, every node `data.dyad[, senderID]` or `data.dyad[, senderID]` must be present in `data.monad[, nodeID]`. Cannot contain special character "@".
- `timeID`: Character string. Quoted name of the variable in both `data.dyad` and `data.monad` indicating the time in which network (and corresponding nodal attributes) were observed. The variable itself must be composed of integers. Cannot contain special character "@".
- `data.dyad`: Data frame. Sociomatrix in “long” (i.e. dyadic) format. Must contain at least three variables: the sender identifier (or identifier of the first node in an undirected networks dyad), the receiver identifier (or identifier of the second node in an undirected network dyad), and the value of the edge between them. Currently, only edges between zero and one (inclusive) are supported.
- `data.monad`: Data frame. Nodal attributes. Must contain a node identifier matching the names of nodes used in the `data.dyad` data frame.
- `n.blocks`: Integer value. How many latent groups should be used to estimate the model?
**n.hmmstates** Integer value. How many hidden Markov state should be used in the HMM? Defaults to 1 (i.e. no HMM).

**directed** Boolean. Is the network directed? Defaults to TRUE.

**mmsbm.control** A named list of optional algorithm control parameters.

- **seed** Integer. Seed the RNG. By default, a random seed is generated and returned for reproducibility purposes.
- **nstart** Integer. Number of random initialization trials. Defaults to 5.
- **spectral** Boolean. Type of initialization algorithm for mixed-membership vectors in static case. If TRUE (default), use spectral clustering with degree correction; otherwise, use kmeans algorithm.
- **init_gibbs** Boolean. Should a collapsed Gibbs sampler of non-regression mmsbm be used to initialize mixed-membership vectors, instead of a spectral or simple kmeans initialization? Setting to TRUE will result in slower initialization and faster model estimation. When TRUE, results are typically very sensitive to choice of alpha (see below).
- **alpha** Numeric positive value. Concentration parameter for collapsed Gibbs sampler to find initial mixed-membership values when init_gibbs=TRUE. Defaults to 1.0.
- **missing** Means of handling missing data. One of "indicator method" (default) or "listwise deletion".
- **svi** Boolean; should stochastic variational inference be used? Defaults to TRUE.
- **vi_iter** Number of maximum iterations in stochastic variational updates. Defaults to 5e2.
- **batch_size** When svi=TRUE, proportion of nodes sampled in each local. Defaults to 0.05 when svi=TRUE, and to 1.0 otherwise.
- **forget_rate** When svi=TRUE, value between (0.5,1], controlling speed of decay of weight of prior parameter values in global steps. Defaults to 0.75 when svi=TRUE, and to 0.0 otherwise.
- **delay** When svi=TRUE, non-negative value controlling weight of past iterations in global steps. Defaults to 1.0 when svi=TRUE, and ignored otherwise.
- **opt_iter** Number of maximum iterations of BFGS in global step. Defaults to 10e3.
- **hessian** Boolean indicating whether the Hessian matrix of regression coefficients should be returned. Defaults to TRUE.
- **assortative** Boolean indicating whether blockmodel should be assortative (i.e. stronger connections within groups) or disassortative (i.e. stronger connections between groups). Defaults to TRUE.
- **mu_block** Numeric vector with two elements: prior mean of blockmodel’s main diagonal elements, and and prior mean of blockmodel’s offdiagonal elements. Defaults to \(c(5.0, -5.0)\) if assortative=TRUE (default) and to \(c(-5.0, 5.0)\) otherwise.
- **var_block** Numeric vector with two positive elements: prior variance of blockmodel’s main diagonal elements, and and prior variance of blockmodel’s offdiagonal elements. Defaults to \(c(5.0, 5.0)\).
**mu_beta** Either single numeric value, in which case the same prior mean is applied to all monadic coefficients, or an array that is npredictors by n.blocks by n.hmmstates, where npredictors is the number of monadic predictors for which a prior mean is being set (prior means need not be set for all) predictors. The rows in the array should be named to identify which variables a prior mean is being set for. Defaults to a common prior mean of 0.0 for all monadic coefficients.

**var_beta** See mu_beta. Defaults to a single common prior variance of 5.0 for all (standardized) monadic coefficients.

**mu_gamma** Either a single numeric value, in which case the same prior mean is applied to all dyadic coefficients, or a named vector of numeric values (with names corresponding to the name of the variable for which a prior mean is being set). Defaults to a common prior mean of 0.0 for all dyadic coefficients.

**var_gamma** See mu_gamma. Defaults to a single common prior variance of 5.0 for all (standardized) dyadic coefficients.

**eta** Numeric positive value. Concentration hyper-parameter for HMM. Defaults to 1.0.

**se_sim** Number of samples from variational posterior of latent variables on which approximation to variance-covariance matrices are based. Defaults to 10.

**dyad_vcov_samp** Maximum number of dyads to sample in computation of variance-covariance of dyadic and blockmodel parameters, when compared to ten percent of the observed dyads. Defaults to 1000.

**fixed_mm** Optional character vector, with "nodeID@timeID" as elements, indicating which mixed-membership vectors should remain constant at their initial values throughout estimation. When only one year is observed, elements should be "nodeID@1". Typically used with mm_init_t.

**mm_init_t** Matrix, n.blocks by nodes across years. Optional initial values for mixed-membership vectors. Although initial values need not be provided for all nodes, column names must have a nodeID@timeID format to avoid ambiguity. When only one year is observed, names should be "nodeID@1".

**kappa_init_t** Matrix, n.hmmstates by number of years. Optional initial values for variational parameters for state probabilities. Columns must be named according to unique year values.

**b_init_t** Matrix, n.blocks by n.blocks. Optional initial values for blockmodel.

**beta_init** Array, predictors by n.blocks by n.hmmstates. Optional initial values for monadic coefficients. If

**gamma_init** Vector. Optional initial values for dyadic coefficients.

**permute** Boolean. Should all permutations be tested to realign initial block models in dynamic case? If FALSE, realignment is done via faster graph matching algorithm, but may not be exact. Defaults to TRUE.

**conv_tol** Numeric value. Absolute tolerance for VI convergence. Defaults to 1e-3.

**verbose** Boolean. Should extra information be printed as model iterates? Defaults to FALSE.
Value

Object of class mmsbm. List with named components:

- **MixedMembership** Matrix of variational posterior of mean of mixed-membership vectors. \(\text{n \_blocks} \times \text{n \_blocks}\).
- **BlockModel** \(\text{n \_blocks} \times \text{n \_blocks}\) matrix of estimated tie log-odds between members of corresponding latent groups. The blockmodel.
- **vcov \_blockmodel** If \(\text{hessian}=\text{TRUE}\), variance-covariance matrix of parameters in blockmodel, ordered in column-major order.
- **MonadCoef** Array of estimated coefficient values for monadic covariates. Has \(\text{n \_blocks}\) columns, and \(\text{n \_hmmstates}\) slices.
- **vcov \_monad** If \(\text{hessian}=\text{TRUE}\), variance-covariance matrix of monadic coefficients.
- **DyadCoef** Vector estimated coefficient values for dyadic covariates.
- **vcov \_dyad** If \(\text{hessian}=\text{TRUE}\), variance-covariance matrix of dyadic coefficients.
- **TransitionKernel** Matrix of estimated HMM transition probabilities.
- **Kappa** Matrix of marginal probabilities of being in an HMM state at any given point in time. \(\text{n \_hmmstates}\) by years (or whatever time interval networks are observed at).
- **LowerBound** Final LB value
- **lb** Vector of all LB across iterations, useful to check early convergence issues.
- **niter** Final number of VI iterations.
- **converged** Convergence indicator; zero indicates failure to converge.
- **NodeIndex** Order in which nodes are stored in all return objects.
- **monadic.data, dyadic.data** Model frames used during estimation (stripped of attributes).
- **forms** Values of selected formal arguments used by other methods.
- **seed** The value of RNG seed used during estimation.
- **call** Original (unevaluated) function call.

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
## Setting to `hessian=TRUE` increases computation time
## but is needed if standard errors are to be computed.
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                        ~ School + Practice + Status,
                        senderID = "Lawyer1")
```
receiverID = "Lawyer2",
nodeID = "Lawyer",
data.dyad = lazega_dyadic,
data.monad = lazega_monadic,
n.blocks = 2,
mmsbm.control = list(seed = 123,
conv_tol = 1e-2,
    hessian = FALSE))

mmsbm_fit  

**Fitter Function for dynamic MMSBM Model**

**Description**

This is the interface to the C++ fitter for the dynamic mixed-membership stochastic blockmodel for network regression.

**Arguments**

- **z_t** Numeric matrix; transpose of monadic design matrix. Should not include intercept row.
- **x_t** Numeric matrix; transpose of dyadic design matrix.
- **y** Numeric vector; vector of edge values. Must have same number of elements as ncol(x_t)
- **time_id_dyad** Integer vector; zero-based time-period identifier for each node.
- **nodes_per_period** Integer vector; total number of unique nodes observed in each time period.
- **node_id_dyad** Integer matrix; zero-based sender and receiver identifier per dyad.
- **mu_b** Numeric matrix; matrix of prior means for elements in blockmodel matrix.
- **var_b** Numeric matrix; matrix of prior variances for elements in blockmodel matrix.
- **pi_init** Numeric matrix; matrix of initial mixed-memberships. Nodes along columns.
- **kappa_init_t** Numeric matrix; matrix of initial marginal HMM state probabilities. Time-periods along columns.
- **b_init_t** Numeric matrix; square matrix of initial values of blockmodel.
- **beta_init** Numeric vector; flat array (column-major order) of initial values of monadic coefficients.
- **gamma_init** Numeric vector; vector of initial values of dyadic coefficients
- **control** List; see the mmsbm.control argument of mmsbm

**Value**

Unclassed list with named components; see Value of mmsbm
plot.mmsbm

Warning

This function is for internal use only. End-users should always resort to \texttt{mmsbm}. In particular, that interface post-processes the return value of this internal in important ways.

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (adelinel@princeton.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Description

The function provides a variety of plotting options for a fitted \texttt{mmsbm} object.

Usage

```r
## S3 method for class 'mmsbm'
plot(x, type = "groups", FX = NULL, node = NULL, ...)
```

Arguments

- \texttt{x}: An object of class \texttt{mmsbm}, a result of a call to \texttt{mmsbm}.
- \texttt{type}: character string denoting the type of plot. The default, "groups," plots the estimated matrix of group by group edge formation probabilities as a network plot, with nodes representing groups (sized proportional to relative membership) and edge colors encoding probability of between-group ties. "blockmodel" plots the same information, but using a tile plot instead of a network plot. "membership" plots average membership in each latent group by time period. "effect" provides a series of plots showing the estimated effect of a shift in monadic covariate values.
- \texttt{FX} with \texttt{type == "effect"}; a list resulting from a call to \texttt{covFX}.
- \texttt{node} with \texttt{type == "membership"}; a character string specifying the node for which group membership should be plotted.
- \texttt{...}: Currently ignored

Value

The requested plot object.

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aadine@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)
Examples

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                     ~ School + Practice + Status,
                     senderID = "Lawyer1",
                     receiverID = "Lawyer2",
                     nodeID = "Lawyer",
                     data.dyad = lazega_dyadic,
                     data.monad = lazega_monadic,
                     n.blocks = 2,
                     mmsbm.control = list(seed = 123,
                                         conv_tol = 1e-2,
                                         hessian = FALSE))

## Plot blockmodel as network
plot(lazega_mmsbm)
```

---

**predict.mmsbm**  
*Predict edges based on estimated mmsbm model*

**Description**

The function produces expected posterior edges based on estimated parameters and (optionally new) predictor data.

**Usage**

```r
## S3 method for class 'mmsbm'
predict(
  object,
  new.data.dyad = NULL,
  new.data.monad = NULL,
  forecast = FALSE,
  type = c("link", "response", "mm"),
  ...
)
```

**Arguments**

- **object**: Object of class mmsbm.
- **new.data.dyad**: An optional data.frame object.
- **new.data.monad**: An optional data.frame object.
simulate.mmsbm

Simulate a complete sociomatrix from an mmsbm object

Description

The function generates one sample network from the posterior predictive of the model represented by a fitted mmsbm object.

forecast  Boolean. Should prediction forecast one step into the future? Defaults to FALSE.

type  Character string. The default is to use the linear predictor of edges. The alternative "response" returns predicted probabilities. The alternative "mm" returns predicted mixed-membership vectors.

Value

If new.data.dyad = NULL, vector of length nrow(object$dyadic.data). Else, vector of length nrow(new.data.dyad).

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers, ~ School + Practice + Status, senderID = "Lawyer1", receiverID = "Lawyer2", nodeID = "Lawyer", data.dyad = lazega_dyadic, data.monad = lazega_monadic, n.blocks = 2, mmsbm.control = list(seed = 123, conv_tol = 1e-2, hessian = FALSE))

## Get in-sample predicted edge probabilities
lazega_preds <- predict(lazega_mmsbm, type = "response")
simulate.mmsbm

Usage

## S3 method for class 'mmsbm'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  new.data.dyad = NULL,
  new.data.monad = NULL,
  ...
)

Arguments

object An object of class mmsbm, a result of a call to mmsbm
nsim Number of networks to simulate
seed RNG seed.
new.data.dyad An optional data.frame object. If not NULL, use these dyadic predictor values instead of those used to fit the original model.
new.data.monad An optional data.frame object. See new.data.dyad.
... Currently ignored

Value

List of length nsim of simulated networks. If new.data.dyad = NULL, each element is a vector of length nrow(object$dyadic.data). Else, vector of length nrow(new.data.dyad). If seed is not NULL, return object includes its value as attribute "seed".

Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
  ~ School + Practice + Status,
  senderID = "Lawyer1",
  receiverID = "Lawyer2",
  nodeID = "Lawyer",
  data.dyad = lazega_dyadic,
  data.monad = lazega_monadic,
  n.blocks = 2,
  mmsbm.control = list(seed = 123,
    conv_tol = 1e-2,
## Simulate 5 new networks
lazega_sim <- simulate(lazega_mmsbm, nsim = 5, seed = 123)

---

**summary.mmsbm**

*Summarize 'mmsbm' object*

### Description

The function summarizes the output of a dynMMSBM model object.

### Usage

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

### Arguments

- **object**
  - An object of class `mmsbm`, a result of a call to `mmsbm`.
- **...**
  - Currently ignored

### Value

List with named components:

- **N** Total number of dyad-time period observations.
- **Number of Clusters** Number of latent groups included in the dynMMSBM model.
- **Percent of Observations in Each Cluster** Average membership in each latent group, across all node-time periods.
- **Edge Formation Probabilities** \( n \times groups \) by \( n \times groups \) matrix of estimated edge formation probabilities between latent groups.
- **Dyadic Coefficients** Vector of estimated coefficient values for dyadic covariates.
- **Monadic Coefficients** Array of estimated coefficient values for monadic covariates. Has \( n \times groups \) columns, and \( n \times hmmstates \) slices.
- **Markov State Probabilities** Average HMM state probabilities across all time periods.

### Author(s)

Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)
**Examples**

```r
library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
                        ~ School + Practice + Status,
                        senderID = "Lawyer1",
                        receiverID = "Lawyer2",
                        nodeID = "Lawyer",
                        data.dyad = lazega_dyadic,
                        data.monad = lazega_monadic,
                        n.blocks = 2,
                        mmsbm.control = list(seed = 123,
                                             conv_tol = 1e-2,
                                             hessian = TRUE))

## Summarize estimated model
summary(lazega_mmsbm)
```

---

**vcov.mmsbm**

*Extract Variance-Covariance Matrix for a Fitted mmsbm Object*

**Description**

Extract Variance-Covariance Matrix for a Fitted mmsbm Object

**Usage**

```r
## S3 method for class 'mmsbm'
vcov(object, param = "All", ...)
```

**Arguments**

- `object` An object of class mmsbm, a result of a call to mmsbm
- `param` Character string, which set of parameters should the vcov be extracted for? One of "MonadCoef", "DyadCoef", "BlockModel" or "All" (the default).
- `...` Currently ignored

**Value**

For `param="All"`, named list of individual return components. For all other values of `param`, a numeric covariance matrix.
Author(s)
Santiago Olivella (olivella@unc.edu), Adeline Lo (aylo@wisc.edu), Tyler Pratt (tyler.pratt@yale.edu), Kosuke Imai (imai@harvard.edu)

Examples

library(NetMix)
## Load datasets
data("lazega_dyadic")
data("lazega_monadic")
## Estimate model with 2 groups
lazega_mmsbm <- mmsbm(SocializeWith ~ Coworkers,
    ~ School + Practice + Status,
    senderID = "Lawyer1",
    receiverID = "Lawyer2",
    nodeID = "Lawyer",
    data.dyad = lazega_dyadic,
    data.monad = lazega_monadic,
    n.blocks = 2,
    mmsbm.control = list(seed = 123,
        conv_tol = 1e-2,
        se_sim = 2)) # Usually requires more samples.

vcov(lazega_mmsbm, "MonadCoef")
Index

* datasets
  lazega_dyadic, 11
  lazega_monadic, 12
  .bar.legend (approxB), 2
  .cbind.fill (approxB), 2
  .compute.alpha (approxB), 2
  .e.pi (approxB), 2
  .findPerm (approxB), 2
  .initPi (approxB), 2
  .mpower (approxB), 2
  .scaleVars (approxB), 2
  .transf (approxB), 2
  .transf_muvar (approxB), 2
  .vcovBeta (approxB), 2

  alphaGrad (approxB), 2
  alphaLBound (approxB), 2
  approxB, 2
  auxfuns (approxB), 2

  coef.mmsbm, 5
  covFX, 6

  getZ (approxB), 2
  gof, 8

  head.mmsbm, 10

  lazega_dyadic, 11
  lazega_monadic, 12

  mmsbm, 12, 17, 18
  mmsbm_fit, 17

  plot.mmsbm, 18
  predict.mmsbm, 19

  simulate.mmsbm, 8, 20
  summary.mmsbm, 22

  vcov.mmsbm, 23