Package: SparseLPM (via r-universe)

September 12, 2024

Type Package

Title The Sparse Latent Position Model for Nonnegative Interaction Data

Version 1.0

Date 2018-08-29

Description Models the nonnegative entries of a rectangular adjacency matrix using a sparse latent position model, as illustrated in Rastelli, R. (2018) ``The Sparse Latent Position Model for nonnegative weighted networks'' <arXiv:1808.09262>.

License GPL-3

Imports Rcpp (>= 0.12.10), gtools, vegan

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Author Riccardo Rastelli [aut, cre]

Maintainer Riccardo Rastelli <riccardoras@gmail.com>

Date/Publication 2018-08-31 23:10:11 UTC

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

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

RemoteRef HEAD

RemoteSha e316fe44f956399e3642b5310306fa5c7018e7bd

Contents

SparseLPM-package 2
slpm_elbo 2
slpm_gen 3
slpm_gof 4
slpm_init 5
slpm_nga 6

Index **9**

SparseLPM-package *The Sparse Latent Position Model for Nonnegative Interaction Data*

Description

Models the nonnegative entries of a rectangular adjacency matrix using a sparse latent position model, as illustrated in Rastelli, R. (2018) "The Sparse Latent Position Model for nonnegative weighted networks" <arXiv:1808.09262>.

Author(s)

Riccardo Rastelli

Mantainer: Riccardo Rastelli <riccardoras@gmail.com>

References

Rastelli, R. (2018) "The Sparse Latent Position Model for nonnegative weighted networks", https://arxiv.org/abs/1808.09262

---

slpm_elbo *slpm_elbo*

Description

Evaluates the evidence lower bound for a given configuration of variational parameters.

Usage

```
slpm_elbo(X, var_pars, hyper_pars, verbose = F)
```

Arguments

`X`	Rectangular adjacency matrix with non-negative entries.
`var_pars`	A list defining the variational parameters of the model. See *Details* for more specific indications.
`hyper_pars`	A list defining the hyperparameters of the model. The list should contain three vectors of length `K` denoted `delta`, `a_gamma` and `b_gamma`, where `K` is the number of latent dimensions.
`verbose`	`TRUE` or `FALSE` indicating whether a lengthy output should be printed out.

Details

The list `var_pars` must contain:

alpha_u_tilde `M*K` matrix denoting the Gaussian means for senders.

alpha_v_tilde `N*K` matrix denoting the Gaussian means for receivers.

beta_u_tilde `M*K` matrix denoting the Gaussian variances for senders.

beta_v_tilde `N*K` matrix denoting the Gaussian variances for receivers.

lambda_tilde `M*N*K` array representing the soft clustering for the edges. This may be interpreted as the posterior probability that edge `ij` is determined by the `k`-th latent dimension.

delta_tilde `K` dimensional vector containing the variational parameters for the mixing proportions. This may be interpreted as the importance of each latent dimension.

a_tilde `K` dimensional vector containing the shapes of the variational Gamma distributions associated to the precisions.

b_tilde `K` dimensional vector containing the rates of the variational Gamma distributions associated to the precisions.

Value

`computing_time`	Number of seconds required for the evaluation.
`elbo`	Value of the ELBO for the given variational parameters.

Examples

```
set.seed(12345)
M <- N <- 10
K <- 2
network <- slpm_gen(M = M, N = N, K = K)
var_pars <- slpm_init(X = network$adj, K = K)
hyper_pars <- list(delta = rep(1,K), a_gamma = rep(1,K), b_gamma = rep(1,K))
slpm_elbo(X = network$adj, var_pars = var_pars, hyper_pars = hyper_pars, verbose = FALSE)
```

`slpm_gen` *slpm_gen*

Description

Generates the adjacency matrix `adj` of a SparseLPM by sampling both the data and model parameters from the posterior distribution.

Usage

```
slpm_gen(M, N, K, hyper_pars = NULL)
```

Arguments

`M`	Number of rows of `adj`.
`N`	Number of cols of `adj`.
`K`	Number of latent dimensions of the SparseLPM.
`hyper_pars`	A list defining the hyperparameters of the model. If left as `NULL` all the hyperparameters are initialised to `1`. Otherwise, the list should contain three vectors of `K` positive values denoted `delta`, `a_gamma` and `b_gamma`, respectively.

Value

A list with components:

`adj`	Generated adjacency matrix.
`U`	Generated latent positions for senders.
`V`	Generated latent positions for receivers.
`lambda`	Latent variables attached to each of the edges, selecting which dimension determines the edge probability.
`gamma`	Vector of the Gaussian precisions associated to the latent dimensions.

Examples

```
set.seed(12345)
network <- slpm_gen(M = 10, N = 8, K = 2)
```

`slpm_gof` *slpm_gof*

Description

Evaluates the expected adjacency matrix for a fitted SparseLPM.

Usage

```
slpm_gof(var_pars)
```

Arguments

`var_pars`	A list defining the variational parameters of the model. See *Details* for more specific indications.

Details

The list `var_pars` must contain:

alpha_u_tilde `M*K` matrix denoting the Gaussian means for senders.

alpha_v_tilde `N*K` matrix denoting the Gaussian means for receivers.

beta_u_tilde `M*K` matrix denoting the Gaussian variances for senders.

beta_v_tilde `N*K` matrix denoting the Gaussian variances for receivers.

lambda_tilde `M*N*K` array representing the soft clustering for the edges. This may be interpreted as the posterior probability that edge `ij` is determined by the `k`-th latent dimension.

delta_tilde `K` dimensional vector containing the variational parameters for the mixing proportions. This may be interpreted as the importance of each of the latent dimensions.

a_tilde `K` dimensional vector containing the shapes of the variational Gamma distributions associated to the precisions.

b_tilde `K` dimensional vector containing the rates of the variational Gamma distributions associated to the precisions. Note that this function only uses the alphas and the lambdas. Also, to avoid numerical instability, the lambdas are automatically pre-transformed into a hard partitioning using a Maximum A Posterior method.

Value

An adjacency matrix with non-negative entries.

Examples

```
set.seed(12345)
M <- N <- 10
K <- 2
fitted_var_pars <- list()
fitted_var_pars$alpha_u_tilde = matrix(rnorm(M*K),M,K)
fitted_var_pars$alpha_v_tilde = matrix(rnorm(N*K),N,K)
fitted_var_pars$lambda_tilde = array(NA,c(M,N,K))
fitted_var_pars$lambda_tilde[,,1] = matrix(runif(M*N),M,N)
fitted_var_pars$lambda_tilde[,,2] = 1-fitted_var_pars$lambda_tilde[,,1]
expected_adj <- slpm_gof(fitted_var_pars)
```

`slpm_init` *slpm_init*

Description

Initialises the variational parameters of a SparseLPM.

Usage

```
slpm_init(X, K, method = "random", threshold = 0.1, stdev = NULL)
```

Arguments

X	Rectangular adjacency matrix with non-negative entries.
K	Number of latent dimensions of the SparseLPM.
method	The variational parameters are initialised at random. However, if method="distance", a distance-based method is used as described in Rastelli ... (2018).
threshold	A small number added to each of the entries of X to avoid numerical instability.
stdev	Standard deviation of the Gaussian used to generate the random latent positions.

Value

Returns a list of variational parameters that can be used as input for slpm_nga or slpm_elbo.

Examples

```
set.seed(12345)
M <- N <- 10
K <- 2
network <- slpm_gen(M = M, N = N, K = K)
var_pars_init <- slpm_init(X = network$adj, K = K)
```

---

slpm_nga *slpm_nga*

Description

Runs a Natural Gradient Ascent algorithm to maximise the variational objective for a Sparse LPM.

Usage

```
slpm_nga(X, K, var_pars_init, hyper_pars = NULL, tol = 0.01, n_iter_max = 100000,
      natural_gradient = T, learning_rate_factor_up = 2, learning_rate_factor_down = 2,
        verbose = F)
```

Arguments

X	Rectangular adjacency matrix with non-negative entries.
K	The number of latent dimension of the model.
var_pars_init	List of variational parameters to be used as starting point for the optimisation. See *Details* for more specific indications.
hyper_pars	List defining the hyperparameters of the model. The list should contain three vectors of K positive values denoted delta, a_gamma and b_gamma, respectively, where K is the number of latent dimensions. If left as null, all delta parameters are set to 0.001, whereas the rest is set to 1.
tol	Positive number setting the stop condition: the algorithm stops if one entire iteration yields an increase in the objective function smaller than this value.

`n_iter_max` Maximum number of iterations the algorithm should be run for.

`natural_gradient`
: `TRUE` or `FALSE` indicating whether the natural gradient instead of the standard gradient should be used.

`learning_rate_factor_up`
: Before any natural gradient ascent update, the current step size is multiplied by this number to ensure that the algorithms tries new solutions which are relatively far from the current one.

`learning_rate_factor_down`
: During any natural gradient ascent update, if a certain step size leads to a decrease in the objective function, then the step is divided by this number repeatedly until an increase is ensured.

`verbose` `TRUE` or `FALSE` indicating whether a lengthy output should be printed out.

Details

`var_pars` and `var_pars_init` are lists with components:

alpha_u_tilde `M*K` matrix representing the Gaussian means for the latent positions of senders.

alpha_v_tilde `N*K` matrix representing the Gaussian means for the latent positions of receivers.

beta_u_tilde `M*K` matrix representing the Gaussian variances for the latent positions of senders.

beta_v_tilde `N*K` matrix representing the Gaussian variances for the latent positions of receivers.

lambda_tilde `M*N*K` array with entries corresponding to the posterior probabilities of assigning each edge to each latent dimension.

delta_tilde Vector of `K` positive values representing the Dirichlet parameters generating the mixing proportions.

a_tilde Vector of `K` positive values corresponding to the shapes of the variational Gamma distribution on the precisions.

b_tilde Vector of `K` positive values corresponding to the rates of the variational Gamma distribution on the precisions.

Value

A list with components:

`computing_time` Number of seconds required for the optimisation process.

`var_pars` List containing the optimal values for the variational parameters.

`learning_rates_u`
: Current step-size for the update of the variational parameters of each Gaussian distribution on the latent positions of senders.

`learning_rates_v`
: Current step-size for the update of the variational parameters of each Gaussian distribution on the latent positions of receivers.

`elbo_values` Values of the variational objective at the end of each of the iterations.

`elbo_init` Value of the variational objective for the initial configuration.

`elbo_final` Value of the variational objective for the optimal solution found.

References

Rastelli, R. (2018) "The Sparse Latent Position Model for nonnegative weighted networks", https://arxiv.org/abs/1808.09262

Examples

```
set.seed(12345)
network <- slpm_gen(M = 15, N = 10, K = 2)
K <- 6
var_pars_init <- slpm_init(X = network$adj, K = K)
res <- slpm_nga(X = network$adj, K = K, var_pars_init = var_pars_init)
```

Index

∗ **package**
    SparseLPM-package, 2

slpm_elbo, 2, *6*
slpm_gen, 3
slpm_gof, 4
slpm_init, 5
slpm_nga, *6*, 6
SparseLPM *(*SparseLPM-package*)*, 2
SparseLPM-package, 2