Package 'bgms'

October 13, 2023

Type Package

Title Bayesian Variable Selection for Networks of Binary and/or Ordinal Variables

Version 0.1.2

Date 2023-10-12

Maintainer Maarten Marsman <m.marsman@uva.nl>

Description Bayesian variable selection methods for analyzing the structure of a Markov Random Field model for a network of binary and/or ordinal variables. Details of the implemented methods can be found in: Marsman and Haslbeck (2023) <doi:10.31234/osf.io/ukwrf>.

License GPL (>= 2)

URL https://maartenmarsman.github.io/bgms/

BugReports https://github.com/MaartenMarsman/bgms/issues

Imports Rcpp (>= 1.0.7), Rdpack, methods

RdMacros Rdpack

LinkingTo Rcpp, RcppProgress

RoxygenNote 7.2.3

Depends R (>= 2.10)

LazyData true

Encoding UTF-8

Suggests knitr, qgraph, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website tidyverse/tidytemplate

NeedsCompilation yes

Author Maarten Marsman [aut, cre] (<https://orcid.org/0000-0001-5309-7502>),
Karoline Huth [ctb] (<https://orcid.org/0000-0002-0662-1591>),
Nikola Sekulovski [ctb] (<https://orcid.org/0000-0001-7032-1684>),
Don van den Bergh [ctb] (<https://orcid.org/0000-0002-9838-7308>)

Repository CRAN

Date/Publication 2023-10-13 19:20:02 UTC

R topics documented:

bgm 2
bgm.em 6
mple 9
mppe 10
mrfSampler 11
Wenchuan 12

Index **14**

`bgm` *Bayesian structure learning in Markov Random Fields of mixed binary and ordinal variables using MCMC.*

Description

The function `bgm` explores the joint pseudoposterior distribution of structures and parameters in a Markov Random Field for mixed binary and ordinal variables.

Usage

```
bgm(
  x,
  iter = 10000,
  burnin = 1000,
  interaction_prior = c("Cauchy", "UnitInfo"),
  cauchy_scale = 2.5,
  edge_prior = c("Bernoulli", "Beta-Bernoulli"),
  inclusion_probability = 0.5,
  beta_bernoulli_alpha = 1,
  beta_bernoulli_beta = 1,
  threshold_alpha = 0.5,
  threshold_beta = 0.5,
  adaptive = FALSE,
  na.action = c("listwise", "impute"),
  save = FALSE,
  display_progress = TRUE
)
```

Arguments

`x`	A data frame or matrix with `n` rows and `p` columns containing binary and ordinal variables for `n` independent observations and `p` variables in the network. Variables are recoded as non-negative integers (`0, 1, ..., m`) if not already done. Unobserved categories are collapsed into other categories after recoding (i.e., if category 1 is unobserved, the data will be recoded from (0, 2) to (0, 1)).

`iter` The number of iterations of the Gibbs sampler. The default of `1e4` is for illustrative purposes. For stable estimates, it is recommended to run the Gibbs sampler for at least `1e5` iterations.

`burnin` The number of iterations of the Gibbs sampler before its output is saved. Since it may take some time for the Gibbs sampler to converge to the posterior distribution, it is recommended not to set this number too low.

`interaction_prior` The type of prior distribution that is used for the interaction effects. Currently, two prior densities are implemented: The Cauchy prior (`interaction_prior = "Cauchy"`) and the Unit Information prior (`interaction_prior = "UnitInfo"`).

`cauchy_scale` The scale of the Cauchy prior for interactions. Defaults to `2.5`.

`edge_prior` The prior distribution for the edges or structure of the network. Two prior distributions are currently implemented: The Bernoulli model `edge_prior = "Bernoulli"` assumes that the probability that an edge between two variables is included is equal to `inclusion_probability` and independent of other edges or variables. When `inclusion_probability = 0.5`, this implies that each network structure receives the same prior weight. The Beta-Bernoulli model `edge_prior = "Beta-Bernoulli"` assumes a beta prior for the unknown inclusion probability with shape parameters `beta_bernoulli_alpha` and `beta_bernoulli_beta`. If `beta_bernoulli_alpha = 1` and `beta_bernoulli_beta = 1`, this means that networks with the same complexity (number of edges) receive the same prior weight. Defaults to `edge_prior = "Bernoulli"`.

`inclusion_probability` The prior edge inclusion probability for the Bernoulli model. Can be a single probability, or a matrix of `p` rows and `p` columns specifying an inclusion probability for each edge pair. Defaults to `inclusion_probability = 0.5`.

`beta_bernoulli_alpha, beta_bernoulli_beta` The two shape parameters of the Beta prior density for the Bernoulli inclusion probability. Must be positive numbers. Defaults to `beta_bernoulli_alpha = 1` and `beta_bernoulli_beta = 1`.

`threshold_alpha, threshold_beta` The shape parameters of the beta-prime prior density for the threshold parameters. Must be positive values. If the two values are equal, the prior density is symmetric about zero. If `threshold_beta` is greater than `threshold_alpha`, the distribution is skewed to the left, and if `threshold_beta` is less than `threshold_alpha`, it is skewed to the right. Smaller values tend to lead to more diffuse prior distributions.

`adaptive` Should the function use an adaptive Metropolis algorithm to update interaction parameters within the model? If `adaptive = TRUE`, bgm adjusts the proposal variance to match the acceptance probability of the random walk Metropolis algorithm to be close to the optimum of `.234` using a Robbins-Monro type algorithm. If `adaptive = FALSE`, it sets the proposal variance to the inverse of the observed Fisher information matrix (the second derivative at the posterior mode). Defaults to `FALSE`.

`na.action` How do you want the function to handle missing data? If `na.action = "listwise"`, listwise deletion is used. If `na.action = "impute"`, missing data are imputed

iteratively during the MCMC procedure. Since imputation of missing data can have a negative impact on the convergence speed of the MCMC procedure, it is recommended to run the MCMC for more iterations. Also, since the numerical routines that search for the mode of the posterior do not have an imputation option, the bgm function will automatically switch to `interaction_prior = "Cauchy"` and `adaptive = TRUE`.

`save` Should the function collect and return all samples from the Gibbs sampler (`save = TRUE`)? Or should it only return the (model-averaged) posterior means (`save = FALSE`)? Defaults to `FALSE`.

`display_progress`
Should the function show a progress bar (`display_progress = TRUE`)? Or not (`display_progress = FALSE`)? Defaults to `TRUE`.

Details

A discrete spike and slab prior distribution is stipulated on the pairwise interactions. By formulating it as a mixture of mutually singular distributions, the function can use a combination of Metropolis-Hastings and Gibbs sampling to create a Markov chain that has the joint posterior distribution as invariant. Current options for the slab distribution are the unit-information prior or a Cauchy with an optional scaling parameter. A Beta-prime distribution is used for the exponent of the category parameters. A uniform prior is used for edge inclusion variables (i.e., the prior probability that the edge is included is 0.5).

Value

If `save = FALSE` (the default), the result is a list of class "bgms" containing the following matrices:

- `gamma`: A matrix with `p` rows and `p` columns, containing posterior inclusion probabilities of individual edges.
- `interactions`: A matrix with `p` rows and `p` columns, containing model-averaged posterior means of the pairwise associations.
- `thresholds`: A matrix with `p` rows and `max(m)` columns, containing model-averaged category thresholds.

If `save = TRUE`, the result is a list of class "bgms" containing:

- `gamma`: A matrix with `iter` rows and `p * (p - 1) / 2` columns, containing the edge inclusion indicators from every iteration of the Gibbs sampler.
- `interactions`: A matrix with `iter` rows and `p * (p - 1) / 2` columns, containing parameter states from every iteration of the Gibbs sampler for the pairwise associations.
- `thresholds`: A matrix with `iter` rows and `sum(m)` columns, containing parameter states from every iteration of the Gibbs sampler for the category thresholds.

Column averages of these matrices provide the model-averaged posterior means.

In addition to the analysis results, the bgm output lists some of the arguments of its call. This is useful for post-processing the results.

Examples

```
#Store user par() settings
op <- par(no.readonly = TRUE)

##Analyse the Wenchuan dataset

# Here, we use 1e4 iterations, for an actual analysis please use at least
# 1e5 iterations.
fit = bgm(x = Wenchuan)


#---------------------------------------------------------------------------|
# INCLUSION - EDGE WEIGHT PLOT
#---------------------------------------------------------------------------|

par(mar = c(6, 5, 1, 1))
plot(x = fit$interactions[lower.tri(fit$interactions)],
     y = fit$gamma[lower.tri(fit$gamma)], ylim = c(0, 1),
     xlab = "", ylab = "", axes = FALSE, pch = 21, bg = "gray", cex = 1.3)
abline(h = 0, lty = 2, col = "gray")
abline(h = 1, lty = 2, col = "gray")
abline(h = .5, lty = 2, col = "gray")
mtext("Posterior Mode Edge Weight", side = 1, line = 3, cex = 1.7)
mtext("Posterior Inclusion Probability", side = 2, line = 3, cex = 1.7)
axis(1)
axis(2, las = 1)


#---------------------------------------------------------------------------|
# EVIDENCE - EDGE WEIGHT PLOT
#---------------------------------------------------------------------------|

#The bgms package currently assumes that the prior odds are 1:
prior.odds = 1
posterior.inclusion = fit$gamma[lower.tri(fit$gamma)]
posterior.odds = posterior.inclusion / (1 - posterior.inclusion)
log.bayesfactor = log(posterior.odds / prior.odds)
log.bayesfactor[log.bayesfactor > 5] = 5

par(mar = c(5, 5, 1, 1) + 0.1)
plot(fit$interactions[lower.tri(fit$interactions)], log.bayesfactor, pch = 21, bg = "#bfbfbf",
     cex = 1.3, axes = FALSE, xlab = "", ylab = "", ylim = c(-5, 5.5),
     xlim = c(-0.5, 1.5))
axis(1)
axis(2, las = 1)
abline(h = log(1/10), lwd = 2, col = "#bfbfbf")
abline(h = log(10), lwd = 2, col = "#bfbfbf")

text(x = 1, y = log(1 / 10), labels = "Evidence for Exclusion", pos = 1,
     cex = 1.7)
text(x = 1, y = log(10), labels = "Evidence for Inclusion", pos = 3, cex = 1.7)
```

```
text(x = 1, y = 0, labels = "Absence of Evidence", cex = 1.7)
mtext("Log-Inclusion Bayes Factor", side = 2, line = 3, cex = 1.5, las = 0)
mtext("Posterior Mean Interactions ", side = 1, line = 3.7, cex = 1.5, las = 0)


#-----------------------------------------------------------------------------|
# THE LOCAL MEDIAN PROBABILITY NETWORK
#-----------------------------------------------------------------------------|

tmp = fit$interactions[lower.tri(fit$interactions)]
tmp[posterior.inclusion < 0.5] = 0

median.prob.model = matrix(0, nrow = ncol(Wenchuan), ncol = ncol(Wenchuan))
median.prob.model[lower.tri(median.prob.model)] = tmp
median.prob.model = median.prob.model + t(median.prob.model)

rownames(median.prob.model) = colnames(Wenchuan)
colnames(median.prob.model) = colnames(Wenchuan)

library(qgraph)
qgraph(median.prob.model,
       theme = "TeamFortress",
       maximum = .5,
       fade = FALSE,
       color = c("#f0ae0e"), vsize = 10, repulsion = .9,
       label.cex = 1.1, label.scale = "FALSE",
       labels = colnames(Wenchuan))

#Restore user par() settings
par(op)
```

---

bgm.em *EM variable selection for a Markov Random Field model for ordinal variables.*

Description

The function bgm.em selects promising edges for the ordinal MRF using the joint pseudolikelihood and a continuous spike and slab prior distribution stipulated on the MRF's interaction or association parameters.

Usage

```
bgm.em(
  x,
  precision = 0.975,
  convergence_criterion = sqrt(.Machine$double.eps),
  theta = 0.5,
  hierarchical = FALSE,
```

```
  indicator_alpha = 1,
  indicator_beta = 1,
  maximum_iterations = 1000,
  threshold_alpha = 0.5,
  threshold_beta = 0.5
)
```

Arguments

x	A dataframe or matrix with n rows and p columns, containing binary and ordinal variables for n independent observations and p variables in the network. Variables are recoded as non-negative integers (0, 1, ..., m) if not done already. Unobserved categories are collapsed into other categories after recoding. See reformat_data for details.
precision	A value between 0 and 1 representing the desired precision for edge selection, equal to one minus the desired type-1 error rate. Default is 0.975.
convergence_criterion	The criterion for the pseudoposterior values' convergence in the EM algorithm. Default is sqrt(.Machine$double.eps).
theta	The prior inclusion probability. A value of 0.5, combined with hierarchical = FALSE, specifies a uniform prior on the network structures' space.
hierarchical	If TRUE, a beta prior distribution with hyperparameters indicator_alpha, indicator_beta is imposed on the prior inclusion probability theta. A uniform prior on inclusion probability, using a beta with indicator_alpha = indicator_beta = 1, specifies a uniform prior on network structure complexity.
indicator_alpha, indicator_beta	Hyperparameters for the beta prior distribution on the prior inclusion probability theta when hierarchical = TRUE. Default is 1.
maximum_iterations	Maximum number of EM iterations used. Default is 1e3. A warning appears if the procedure hasn't converged within the maximum number of iterations.
threshold_alpha, threshold_beta	Shape parameters for the Beta-prime prior on thresholds. Default is 1.

Value

A list containing:

- interactions: A matrix with p rows and p columns, containing the pairwise association estimates in the off-diagonal elements.
- gamma: A matrix with p rows and p columns, containing the expected values of edge inclusion variables (local posterior probabilities of edge inclusion).
- thresholds: A matrix with p rows and max(m) columns, containing the category thresholds for each node.
- theta (if hierarchical == TRUE): A numeric value representing the modal estimate of the prior inclusion probability.

Examples

```
#Store user par() settings
op <- par(no.readonly = TRUE)

##Analyse the Wenchuan dataset
fit = bgm.em(x = Wenchuan)


#--------------------------------------------------------------------------------|
# INCLUSION - EDGE WEIGHT PLOT
#--------------------------------------------------------------------------------|

par(mar = c(6, 5, 1, 1))
plot(x = fit$interactions[lower.tri(fit$interactions)],
     y = fit$gamma[lower.tri(fit$gamma)], ylim = c(0, 1),
     xlab = "", ylab = "", axes = FALSE, pch = 21, bg = "#bfbfbf", cex = 1.3)
abline(h = 0, lty = 2, col = "#bfbfbf")
abline(h = 1, lty = 2, col = "#bfbfbf")
abline(h = .5, lty = 2, col = "#bfbfbf")
mtext("Posterior Inclusion Probability", side = 1, line = 3, cex = 1.7)
mtext("Posterior Mode Edge Weight", side = 2, line = 3, cex = 1.7)
axis(1)
axis(2, las = 1)


#--------------------------------------------------------------------------------|
# THE LOCAL MEDIAN PROBABILITY NETWORK
#--------------------------------------------------------------------------------|

library(qgraph) #For plotting the estimated network

posterior.inclusion = fit$gamma[lower.tri(fit$gamma)]
tmp = fit$interactions[lower.tri(fit$interactions)]
tmp[posterior.inclusion < 0.5] = 0

median.prob.model = matrix(0, nrow = ncol(Wenchuan), ncol = ncol(Wenchuan))
median.prob.model[lower.tri(median.prob.model)] = tmp
median.prob.model = median.prob.model + t(median.prob.model)

rownames(median.prob.model) = colnames(Wenchuan)
colnames(median.prob.model) = colnames(Wenchuan)

qgraph(median.prob.model,
       theme = "TeamFortress",
       maximum = .5,
       fade = FALSE,
       color = c("#f0ae0e"), vsize = 10, repulsion = .9,
       label.cex = 1.1, label.scale = "FALSE",
       labels = colnames(Wenchuan))

#Restore user par() settings
```

```
par(op)
```

`mple` *Maximum Pseudolikelihood Estimation for an Ordinal Markov Random Field Model*

Description

The function `mple` estimates the parameters for the ordinal MRF by optimizing the joint pseudolikelihood with the Newton-Raphson method.

Usage

```
mple(
  x,
  convergence_criterion = sqrt(.Machine$double.eps),
  maximum_iterations = 1000,
  thresholds,
  interactions
)
```

Arguments

`x`	A dataframe or matrix with `n` rows and `p` columns, containing binary and ordinal variables for `n` independent observations and `p` variables in the network. Variables are recoded as non-negative integers (`0, 1, ..., m`) if not done already. Unobserved categories are collapsed into other categories after recoding. See `reformat_data` for details.
`convergence_criterion`	The convergence criterion for the pseudoposterior values in the EM algorithm. Defaults to `sqrt(.Machine$double.eps)`.
`maximum_iterations`	The maximum number of EM iterations used. Defaults to `1e3`. A warning is issued if the procedure has not converged in `maximum_iterations` iterations.
`thresholds`	A matrix with `p` rows and `max(m)` columns, containing the category thresholds for each node. Used as starting values in the Newton-Raphson procedure. Optional.
`interactions`	A matrix with `p` rows and `p` columns, containing the pairwise association estimates in the off-diagonal elements. Used as starting values in the Newton-Raphson procedure. Optional.

Value

A list containing:

- `interactions`: A matrix with `p` rows and `p` columns, containing the maximum pseudolikelihood estimates of the pairwise associations in the off-diagonal elements.
- `thresholds`: A matrix with `p` rows and `max(m)` columns, containing the maximum pseudolikelihood estimates of the category thresholds for each node.

`mppe` *Optimize Pseudoposterior for an Ordinal Markov Random Field Model*

Description

The function `mppe` estimates the parameters for the ordinal MRF by optimizing the pseudoposterior with the Newton-Raphson method.

Usage

```
mppe(
  x,
  interaction_prior = c("Cauchy", "UnitInfo"),
  cauchy_scale = 2.5,
  threshold_alpha = 0.5,
  threshold_beta = 0.5,
  convergence_criterion = sqrt(.Machine$double.eps),
  maximum_iterations = 1000,
  thresholds,
  interactions
)
```

Arguments

`x`	A dataframe or matrix with `n` rows and `p` columns, containing binary and ordinal variables for `n` independent observations and `p` variables in the network. Variables are recoded as non-negative integers (`0`, `1`, `...`, `m`) if not done already. Unobserved categories are collapsed into other categories after recoding. See `reformat_data` for details.
`interaction_prior`	The prior distribution for the interaction effects. Currently, two prior densities are implemented: The Unit Information prior (`interaction_prior = "UnitInfo"`) and the Cauchy prior (`interaction_prior = "Cauchy"`). Defaults to `"Cauchy"`.
`cauchy_scale`	The scale of the Cauchy prior for interactions. Defaults to `2.5`.
`threshold_alpha, threshold_beta`	The shape parameters of the Beta-prime prior for the thresholds. Default to `0.5`.

`convergence_criterion`
: The convergence criterion for the pseudoposterior values in the EM algorithm. Defaults to `sqrt(.Machine$double.eps)`.

`maximum_iterations`
: The maximum number of EM iterations used. Defaults to `1e3`. A warning is issued if the procedure has not converged in `maximum_iterations` iterations.

`thresholds`
: A matrix with `p` rows and `max(m)` columns, containing the category thresholds for each node. Used as starting values in the Newton-Raphson procedure. Optional.

`interactions`
: A matrix with `p` rows and `p` columns, containing the pairwise association estimates in the off-diagonal elements. Used as starting values in the Newton-Raphson procedure. Optional.

Value

A list containing:

- `interactions`: A matrix with `p` rows and `p` columns, containing the maximum pseudoposterior estimates of the pairwise associations in the off-diagonal elements.
- `thresholds`: A matrix with `p` rows and `max(m)` columns, containing the maximum pseudoposterior estimates of the category thresholds for each node.
- `hessian`: A square matrix with `sum(m) + p(p-1)/2` rows and columns, evaluated at the maximum pseudoposterior estimates. The top-left square contains the thresholds, the bottom-right square the associations (of the form `(1,2), (1, 3), ..., (2, 1), ...`).

In the case that `interaction_prior = "UnitInfo"`, the list also contains the `p` by `p` matrix `unit_info`, which contains the asymptotic variances that are used to set the unit information prior for the association effects in the `bgms` function.

`mrfSampler` *Sample states of the ordinal MRF*

Description

This function samples states from the ordinal MRF using a Gibbs sampler. The Gibbs sampler is initiated with random values from the response options, after which it proceeds by simulating states for each node from a logistic model using the other node states as predictor variables.

Usage

```
mrfSampler(
  no_states,
  no_nodes,
  no_categories,
  interactions,
  thresholds,
  iter = 1000
)
```

Arguments

no_states	The number of states of the ordinal MRF to be generated.
no_nodes	The number of nodes in the ordinal MRF.
no_categories	Either a positive integer or a vector of positive integers of length no_nodes. The number of response categories on top of the base category: no_categories = 1 generates binary states.
interactions	A symmetric no_nodes by no_nodes matrix of pairwise interactions. Only its off-diagonal elements are used.
thresholds	A no_nodes by max(no_categories) matrix of category thresholds. The elements in row r indicate the thresholds of node r. If no_categories is a vector, only the first no_categories[r] elements are used in row r.
iter	The number of iterations used by the Gibbs sampler. The function provides the last state of the Gibbs sampler as output. By default set to 1e3.

Value

A no_states by no_nodes matrix of simulated states of the ordinal MRF.

Examples

```
# Generate responses from a network of five binary and ordinal variables.
no_nodes = 5
no_categories = sample(1:5, size = no_nodes, replace = TRUE)

Interactions = matrix(0, nrow = no_nodes, ncol = no_nodes)
Interactions[2, 1] = Interactions[4, 1] = Interactions[3, 2] =
  Interactions[5, 2] = Interactions[5, 4] = .25
Interactions = Interactions + t(Interactions)
Thresholds = matrix(0, nrow = no_nodes, ncol = max(no_categories))
x = mrfSampler(no_states = 1e3,
               no_nodes = no_nodes,
               no_categories = no_categories,
               interactions = Interactions,
               thresholds = Thresholds)
```

Wenchuan — *Post-traumatic stress disorder symptoms of Wenchuan earthquake survivors*

Description

A data set containing items measuring symptoms of posttraumatic stress disorder (PTSD) (McNally et al. 2015). Participants were 362 Chinese adults who survived the Wenchuan earthquake and lost at least one child in the disaster. PTSD symptoms were reported using the civilian version of the Posttraumatic Checklist, which consists of 17 items, each assessing one of the DSM-IV symptoms of PTSD. Participants rated each item on a five-point scale ranging from “not at all” to “extremely” to indicate how much the symptom bothered them in the past month.

Usage

```
data("Wenchuan")
```

Format

A matrix with 362 rows and 17 columns:

intrusion Repeated, disturbing memories, thoughts, or images of a stressful experience from the past?

dreams Repeated, disturbing dreams of a stressful experience from the past?

flash Suddenly acting or feeling as if a stressful experience were happening again (as if you were reliving it)?

upset Feeling very upset when something reminded you of a stressful experience from the past?

physior Having physical reactions (e.g., heart pounding, trouble breathing, sweating) when something reminded you of a stressful experience from the past?

avoidth Avoiding thinking about or talking about a stressful experience from the past or avoiding having feelings related to it?

avoidact Avoiding activities or situations because they reminded you of a stressful experience from the past?

amnesia Trouble remembering important parts of a stressful experience from the past?

lossint Loss of interest in activities that you used to enjoy?

distant Feeling distant or cut off from other people?

numb Feeling emotionally numb or being unable to have loving feelings for those close to you?

future Feeling as if your future will somehow be cut short?

sleep Trouble falling or staying asleep?

anger Feeling irritable or having angry outbursts?

concen Having difficulty concentrating?

hyper Being "super-alert" or watchful or on guard?

startle Feeling jumpy or easily startled?

Source

http://psychosystems.org/wp-content/uploads/2014/10/Wenchuan.csv

References

McNally RJ, Robinaugh DJ, Wu GWY, Wang L, Deserno MK, Borsboom D (2015). "Mental disorders as causal systems: A network approach to posttraumatic stress disorder." *Clinical Psychological Science*, **5**(6), 836–849. doi:10.1177/2167702614553230.

Index

∗ **datasets**
Wenchuan, 12

bgm, 2
bgm.em, 6

mple, 9
mppe, 10
mrfSampler, 11

Wenchuan, 12