Package: bgms (via r-universe)

June 8, 2026

Type Package

Title Bayesian Analysis of Graphical Models

Version 0.2.0.0

Date 2026-03-26

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

Description Bayesian estimation and edge selection for graphical models of mixed binary, ordinal, and continuous variables. The variable types determine the model: an ordinal Markov random field for discrete data, a Gaussian graphical model for continuous data, or a mixed Markov random field combining both. Edge inclusion is determined through spike-and-slab priors, yielding posterior inclusion probabilities for each edge. Supports multi-group comparison via 'bgmCompare()', simulation, prediction, and missing data imputation.

License GPL (>= 2)

URL https://Bayesian-Graphical-Modelling-Lab.github.io/bgms/, https://github.com/Bayesian-Graphical-Modelling-Lab/bgms

BugReports https://github.com/Bayesian-Graphical-Modelling-Lab/bgms/issues

Imports Rcpp (>= 1.0.7), RcppParallel, Rdpack, S7, methods, lifecycle, stats

RdMacros Rdpack

LinkingTo Rcpp, RcppArmadillo, RcppParallel, dqrng, BH

Roxygen list(markdown = TRUE)

Depends R (>= 3.5)

LazyData true

Encoding UTF-8

Suggests coda, covr, knitr, MASS, parallel, qgraph, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website tidyverse/tidytemplate

Config/roxygen2/version 8.0.0

Config/pak/sysreqs make

Repository https://bayesian-graphical-modelling-lab.r-universe.dev

Date/Publication 2026-06-08 08:58:48 UTC

RemoteUrl https://github.com/bayesian-graphical-modelling-lab/bgms

RemoteRef HEAD

RemoteSha 9f85b8a8f23f96af740dc76e894459cf2303990b

Contents

ADHD *ADHD Symptom Checklist for Children Aged 6–8 Years*

Description

This dataset includes ADHD symptom ratings for 355 children aged 6 to 8 years from the Children's Attention Project (CAP) cohort (Silk et al. 2019). The sample consists of 146 children diagnosed with ADHD and 209 without a diagnosis. Symptoms were assessed through structured interviews with parents using the NIMH Diagnostic Interview Schedule for Children IV (DISC-IV) (Shaffer et al. 2000). The checklist includes 18 items: 9 Inattentive (I) and 9 Hyperactive/Impulsive (HI). Each item is binary (1 = present, 0 = absent).

Usage

```
data("ADHD")
```

Format

A data frame with 355 rows and 19 columns.

group ADHD diagnosis: 1 = diagnosed, 0 = not diagnosed

avoid Often avoids, dislikes, or is reluctant to engage in tasks that require sustained mental effort (I)

closeatt Often fails to give close attention to details or makes careless mistakes in schoolwork, work, or other activities (I)

distract Is often easily distracted by extraneous stimuli (I)

forget Is often forgetful in daily activities (I)

instruct Often does not follow through on instructions and fails to finish schoolwork, chores, or duties in the workplace (I)

listen Often does not seem to listen when spoken to directly (I)

loses Often loses things necessary for tasks or activities (I)

org Often has difficulty organizing tasks and activities (I)

susatt Often has difficulty sustaining attention in tasks or play activities (I)

blurts Often blurts out answers before questions have been completed (HI)

fidget Often fidgets with hands or feet or squirms in seat (HI)

interrupt Often interrupts or intrudes on others (HI)

motor Is often "on the go" or often acts as if "driven by a motor" (HI)

quiet Often has difficulty playing or engaging in leisure activities quietly (HI)

runs Often runs about or climbs excessively in situations in which it is inappropriate (HI)

seat Often leaves seat in classroom or in other situations in which remaining seated is expected (HI)

talks Often talks excessively (HI)

turn Often has difficulty awaiting turn (HI)

Source

Silk et al. (2019). Data retrieved from doi:10.1371/journal.pone.0211053.s004. Licensed under the CC-BY 4.0: https://creativecommons.org/licenses/by/4.0/

References

Shaffer D, Fisher P, Lucas CP, Dulcan MK, Schwab-Stone ME (2000). "NIMH Diagnostic Interview Schedule for Children Version IV (NIMH DISC-IV): description, differences from previous versions, and reliability of some common diagnoses." *Journal of the American Academy of Child & Adolescent Psychiatry*, **39**, 28–38. doi:10.1097/00004583200001000000014. PMID: 10638065.

Silk TJ, Malpas CB, Beare R, Efron D, Anderson V, Hazell P, Jongeling B, Nicholson JM, Sciberras E (2019). "A network analysis approach to ADHD symptoms: More than the sum of its parts." *PLOS ONE*, **14**(1), e0211053. doi:10.1371/journal.pone.0211053.

`bernoulli_prior` *Bernoulli Prior for Inclusion Indicators*

Description

Specifies a Bernoulli prior for inclusion indicators with a fixed inclusion probability. Used for edge selection in `bgm` and difference selection in `bgmCompare`.

Usage

```
bernoulli_prior(inclusion_probability = 0.5)
```

Arguments

`inclusion_probability`
: Numeric scalar or symmetric matrix. Prior probability of each edge being included. A scalar applies to all edges; a matrix allows edge-specific probabilities. Must be in (0, 1). Default: `0.5`.

Value

An object of class "bgms_indicator_prior" with family = "Bernoulli".

See Also

beta_bernoulli_prior, sbm_prior, bgm

Other prior-constructors: beta_bernoulli_prior(), beta_prime_prior(), cauchy_prior(), exponential_prior(), gamma_prior(), normal_prior(), sbm_prior()

Examples

```
bernoulli_prior()
bernoulli_prior(inclusion_probability = 0.25)
```

---

beta_bernoulli_prior *Beta-Bernoulli Prior for Inclusion Indicators*

Description

Specifies a Beta-Bernoulli prior for inclusion indicators. The inclusion probability is drawn from a Beta(α, β) distribution and shared across all edges.

Usage

```
beta_bernoulli_prior(alpha = 1, beta = 1)
```

Arguments

alpha	Positive numeric. First shape parameter of the Beta distribution. Default: 1.
beta	Positive numeric. Second shape parameter of the Beta distribution. Default: 1.

Value

An object of class "bgms_indicator_prior" with family = "Beta-Bernoulli".

See Also

bernoulli_prior, sbm_prior, bgm

Other prior-constructors: bernoulli_prior(), beta_prime_prior(), cauchy_prior(), exponential_prior(), gamma_prior(), normal_prior(), sbm_prior()

Examples

```
beta_bernoulli_prior()
beta_bernoulli_prior(alpha = 2, beta = 5)
```

beta_prime_prior *Beta-Prime Prior for Model Parameters*

Description

Specifies a beta-prime prior on model parameters. The parameterization follows the logistic transformation: $\sigma(\mu) \sim \text{Beta}(\alpha, \beta)$, so $\mu = \text{logit}(Y)$ where $Y \sim \text{Beta}(\alpha, \beta)$.

Usage

```
beta_prime_prior(alpha = 0.5, beta = 0.5)
```

Arguments

alpha	Positive numeric. First shape parameter. Default: 0.5.
beta	Positive numeric. Second shape parameter. Default: 0.5.

Value

An object of class "bgms_parameter_prior" with family = "beta-prime".

See Also

cauchy_prior, normal_prior, bgm

Other prior-constructors: bernoulli_prior(), beta_bernoulli_prior(), cauchy_prior(), exponential_prior(), gamma_prior(), normal_prior(), sbm_prior()

Examples

```
beta_prime_prior()
beta_prime_prior(alpha = 1, beta = 1)
```

bgm *Bayesian Estimation or Edge Selection for Markov Random Fields*

Description

The bgm function estimates the pseudoposterior distribution of the parameters of a Markov Random Field (MRF) for binary, ordinal, continuous, or mixed (discrete and continuous) variables. Optionally, it performs Bayesian edge selection using discrete spike-and-slab priors to infer the network structure.

Usage

```
bgm(
  x,
  variable_type = "ordinal",
  baseline_category,
  iter = 2000,
  warmup = 2000,
  interaction_prior = cauchy_prior(scale = 1),
  threshold_prior = beta_prime_prior(alpha = 0.5, beta = 0.5),
  means_prior = normal_prior(scale = 1),
  precision_scale_prior = gamma_prior(shape = 1, rate = 1),
  delta = NULL,
  edge_selection = TRUE,
  edge_prior = bernoulli_prior(0.5),
  na_action = c("listwise", "impute"),
  update_method = c("nuts", "adaptive-metropolis"),
  target_accept,
  nuts_max_depth = 10,
  learn_mass_matrix = TRUE,
  chains = 4,
  cores = parallel::detectCores(),
  display_progress = c("per-chain", "total", "none"),
  seed = NULL,
  standardize = FALSE,
  verbose = getOption("bgms.verbose", TRUE),
  progress_callback = NULL,
  pairwise_scale,
  main_alpha,
  main_beta,
  inclusion_probability,
  beta_bernoulli_alpha,
  beta_bernoulli_beta,
  beta_bernoulli_alpha_between,
  beta_bernoulli_beta_between,
  dirichlet_alpha,
  lambda,
  interaction_scale,
  burnin,
  save,
  threshold_alpha,
  threshold_beta
)
```

Arguments

x	A data frame or matrix with n rows and p columns. Columns may contain binary, ordinal, or continuous variables (see variable_type). Discrete variables are automatically recoded to non-negative integers (0, 1, ..., m); for regular

ordinal variables, unobserved categories are collapsed, while Blume–Capel variables retain all categories. Continuous variables are column-centered internally so that the GGM likelihood is formulated with a zero-mean assumption.

`variable_type` Character or character vector. Specifies the type of each variable in `x`. Allowed values: `"ordinal"`, `"blume-capel"`, or `"continuous"`. A single string applies to all variables. A per-variable vector that mixes discrete (`"ordinal"` / `"blume-capel"`) and `"continuous"` types fits a mixed MRF. Binary variables are automatically treated as `"ordinal"`. Default: `"ordinal"`.

`baseline_category`
Integer or vector. Baseline category used in Blume–Capel variables. Can be a single integer (applied to all) or a vector of length `p`. Required if at least one variable is of type `"blume-capel"`.

`iter` Integer. Number of post–burn-in iterations (per chain). Default: `2e3`.

`warmup` Integer. Number of warmup iterations before collecting samples. Short warmups trigger progressive warnings (NUTS only); see `validate_sampler()` for the thresholds. Default: `2e3`.

`interaction_prior`
A prior specification object for pairwise interaction parameters, created by one of the prior constructor functions:

- `cauchy_prior()`: Cauchy(0, scale) prior (default).
- `normal_prior()`: Normal(0, scale) prior.
- `beta_prime_prior()`: Beta-prime prior.

Default: `cauchy_prior(scale = 1)`.

`threshold_prior`
A prior specification object for threshold (main effect) parameters, created by one of the prior constructor functions:

- `beta_prime_prior()`: Beta-prime prior (default).
- `cauchy_prior()`: Cauchy(0, scale) prior.
- `normal_prior()`: Normal(0, scale) prior.

Default: `beta_prime_prior(alpha = 0.5, beta = 0.5)`.

`means_prior` A prior specification object for continuous variable means (mixed MRF models only), created by one of the prior constructor functions:

- `normal_prior()`: Normal(0, scale) prior (default).
- `cauchy_prior()`: Cauchy(0, scale) prior.
- `beta_prime_prior()`: Beta-prime prior.

Only used when the model includes continuous variables. Ignored for pure ordinal or pure continuous (GGM) models. Default: `normal_prior(scale = 1)`.

`precision_scale_prior`
A prior specification object for the diagonal elements of the precision matrix, created by one of:

- `gamma_prior()`: Gamma(shape, rate) prior (default).
- `exponential_prior()`: Exponential(rate) prior.

Only used for models with continuous variables (GGM and mixed MRF). Ignored for pure ordinal models. Default: `gamma_prior(shape = 1, rate = 1)`.

`delta` Non-negative numeric, or `NULL` for the dimension- adaptive default. Determinant-tilt exponent on the continuous-block precision matrix K (GGM) or K_{yy} (mixed MRF): multiplies the prior by $|K|^{\delta}$, softly repelling the chain from the positive-definite cone boundary. `delta = NULL` (default) auto-resolves to $0.5 \log(p)$ where p is the dimension of the continuous precision matrix (the number of variables for GGM, the number of continuous variables for mixed MRF). The rule is the simple form of the dimension-adaptive scaling $\delta(p) = c \log p$ with $c \in (0.3, 0.6)$ discussed in the companion paper on determinant-tilted spike-and-slab priors (Marsman et al., in preparation). Pass `delta = 0` for the untilted prior (the companion-paper baseline) or a non-negative numeric to override. Both NUTS and adaptive-Metropolis update paths apply the tilt. Not allowed for pure ordinal models (no precision matrix to tilt).

`edge_selection` Logical. Whether to perform Bayesian edge selection. If `FALSE`, the model estimates all edges. Default: `TRUE`.

`edge_prior` An edge prior specification object, or a character string (deprecated). Specifies the prior for edge inclusion. Preferred: pass an object from one of:

- `bernoulli_prior()`: Fixed inclusion probability (default).
- `beta_bernoulli_prior()`: Beta-distributed inclusion.
- `sbm_prior()`: Stochastic Block Model.

Legacy character strings `"Bernoulli"`, `"Beta-Bernoulli"`, `"Stochastic-Block"` are still accepted but deprecated. Default: `bernoulli_prior(0.5)`.

`na_action` Character. Specifies missing data handling. Either `"listwise"` (drop rows with missing values) or `"impute"` (perform single imputation during sampling). Default: `"listwise"`.

`update_method` Character. Specifies how the MCMC sampler updates the model parameters:

`"adaptive-metropolis"` Componentwise adaptive Metropolis–Hastings with Robbins–Monro proposal adaptation.

`"nuts"` The No-U-Turn Sampler, a gradient-based sampler available for all variable types, including under edge selection. The Gaussian graphical model uses a free-element Cholesky parameterization that keeps the precision matrix positive-definite; the mixed model uses RATTLE constrained integration when excluded edges impose constraints.

Default: `"nuts"`.

`target_accept` Numeric between 0 and 1. Target acceptance rate for the sampler. Defaults are set automatically if not supplied: `0.44` for adaptive Metropolis and `0.80` for NUTS.

`nuts_max_depth` Integer. Maximum tree depth in NUTS. Must be positive. Default: `10`.

`learn_mass_matrix`

Logical. If `TRUE`, adapt a diagonal mass matrix during warmup (NUTS only). If `FALSE`, use the identity matrix. Default: `TRUE`.

`chains` Integer. Number of parallel chains to run. Default: `4`.

`cores` Integer. Number of CPU cores for parallel execution. Default: `parallel::detectCores()`.

`display_progress`

Character. Controls progress reporting during sampling. Options: `"per-chain"` (separate bar per chain), `"total"` (single combined bar), or `"none"` (no progress). Default: `"per-chain"`.

seed — Optional integer. Random seed for reproducibility. Must be a single non-negative integer.

standardize — Logical. If TRUE, the prior scale for each pairwise interaction is adjusted based on the range of response scores. Variables with more response categories have larger score products $x_i \cdot x_j$, which typically correspond to smaller interaction effects σ_{ij}. Without standardization, a fixed prior scale is relatively wide for these smaller effects, resulting in less shrinkage for high-category pairs and more shrinkage for low-category pairs. Standardization scales the prior proportionally to the maximum score product, ensuring equivalent relative shrinkage across all pairs. After internal recoding, regular ordinal variables have scores $0, 1, \ldots, m$. The adjusted scale for the interaction between variables i and j is `pairwise_scale * m_i * m_j`, so that `pairwise_scale` itself applies to the unit interval case (binary variables where $m_i = m_j = 1$). For Blume-Capel variables with reference category b, scores are centered as $-b, \ldots, m - b$, and the adjustment uses the maximum absolute product of the score endpoints. For mixed pairs, ordinal variables use raw score endpoints $(0, m)$ and Blume-Capel variables use centered score endpoints $(-b, m - b)$. Default: FALSE.

verbose — Logical. If TRUE, prints informational messages during data processing (e.g., missing data handling, variable recoding). Defaults to `getOption("bgms.verbose", TRUE)`. Set `options(bgms.verbose = FALSE)` to suppress messages globally.

progress_callback — An optional R function with signature `function(completed, total)` that is called at regular intervals during sampling, where `completed` is the number of iterations completed across all chains and `total` is the total number of iterations. Useful for external front-ends (e.g., JASP) that supply their own progress reporting. When NULL (the default), no callback is invoked.

pairwise_scale — **[Deprecated]** Double. Scale of the Cauchy prior for pairwise interaction parameters. Use `interaction_prior` instead. Default: 1.

main_alpha, main_beta — **[Deprecated]** Double. Shape parameters of the beta-prime prior for threshold parameters. Use `threshold_prior` instead. Defaults: `main_alpha = 0.5` and `main_beta = 0.5`.

inclusion_probability — **[Deprecated]** Numeric scalar. Use `edge_prior = bernoulli_prior(inclusion_probability)` instead. Default: 0.5.

beta_bernoulli_alpha, beta_bernoulli_beta — **[Deprecated]** Double. Use `edge_prior = beta_bernoulli_prior(alpha, beta)` instead. Defaults: 1.

beta_bernoulli_alpha_between, beta_bernoulli_beta_between — **[Deprecated]** Double. Use `edge_prior = sbm_prior(alpha_between, beta_between)` instead. Defaults: 1.

dirichlet_alpha — **[Deprecated]** Double. Use `edge_prior = sbm_prior(dirichlet_alpha = ...)` instead. Default: 1.

lambda — **[Deprecated]** Double. Use `edge_prior = sbm_prior(lambda = ...)` instead. Default: 1.

`interaction_scale`, `burnin`, `save`, `threshold_alpha`, `threshold_beta`
: **[Deprecated]** Deprecated arguments as of **bgms 0.1.6.0**. Use `interaction_prior`, `warmup`, and `threshold_prior` instead.

Details

Depending on the variable types, the model is an ordinal MRF, a Gaussian graphical model (GGM), or a mixed MRF. Both regular ordinal variables and Blume–Capel ordinal variables (with a baseline category) are supported.

Edge selection uses spike-and-slab priors with Bernoulli, Beta-Bernoulli, or Stochastic-Block priors on the edge inclusion indicators. Parameters are sampled with NUTS (default) or adaptive Metropolis–Hastings, with a multi-stage warmup schedule. Missing data can be handled via listwise deletion or Gibbs imputation.

For full details on model specification, prior choices, warmup, and output interpretation, see the package website at https://bayesian-graphical-modelling-lab.github.io/bgms-docs/.

Value

An S7 object of class `bgms` with posterior summaries, posterior mean matrices, and access to raw MCMC draws. Its fields are accessible with `$` and `[[` for backward compatibility, and it can be passed to `print()`, `summary()`, and `coef()`.

Main components include:

- `posterior_summary_main`: Data frame with posterior summaries (mean, sd, MCSE, ESS, Rhat) for main-effect parameters. For OMRF models these are category thresholds; for mixed MRF models these are discrete thresholds and continuous means. `NULL` for GGM models (no main effects).
- `posterior_summary_quadratic`: Data frame with posterior summaries for the residual variance parameters (GGM and mixed MRF). `NULL` for OMRF models.
- `posterior_summary_pairwise`: Data frame with posterior summaries for partial association parameters.
- `posterior_summary_indicator`: Data frame with posterior summaries for edge inclusion indicators (if `edge_selection = TRUE`).
- `posterior_mean_main`: Posterior mean of main-effect parameters. `NULL` for GGM models. For OMRF: a matrix (p x max_categories) of category thresholds. For mixed MRF: a list with `$discrete` (threshold matrix) and `$continuous` (q x 1 matrix of means).
- `posterior_mean_pairwise`: Symmetric matrix of posterior mean partial associations (zero diagonal). For continuous variables these are unstandardized partial correlations; for discrete variables these are half the log adjacent-category odds ratio. Use `extract_precision()`, `extract_partial_correlations()`, or `extract_log_odds()` to convert to interpretable scales.
- `posterior_mean_residual_variance`: Named numeric vector of posterior mean residual variances $1/\Theta_{ii}$. Present for GGM and mixed MRF models; `NULL` for OMRF.
- `posterior_mean_indicator`: Symmetric matrix of posterior mean inclusion probabilities (if edge selection was enabled).
- Additional summaries returned when `edge_prior = "Stochastic-Block"`. For more details about this prior see Sekulovski et al. (2025).

  - `posterior_summary_pairwise_allocations`: Data frame with posterior summaries (mean, sd, MCSE, ESS, Rhat) for the pairwise cluster co-occurrence of the nodes. This serves to indicate whether the estimated posterior allocations,co-clustering matrix and posterior cluster probabilities (see blow) have converged.
  - `posterior_mean_coclustering_matrix`: a symmetric matrix of pairwise proportions of occurrence of every variable. This matrix can be plotted to visually inspect the estimated number of clusters and visually inspect nodes that tend to switch clusters.
  - `posterior_mean_allocations`: A vector with the posterior mean of the cluster allocations of the nodes. This is calculated using the method proposed in Dahl (2009).
  - `posterior_mode_allocations`: A vector with the posterior mode of the cluster allocations of the nodes.
  - `posterior_num_blocks`: A data frame with the estimated posterior inclusion probabilities for all the possible number of clusters.

- `raw_samples`: A list of raw MCMC draws per chain:

  `main` List of main effect samples.

  `pairwise` List of pairwise effect samples.

  `indicator` List of indicator samples (if edge selection enabled).

  `allocations` List of cluster allocations (if SBM prior used).

  `nchains` Number of chains.

  `niter` Number of post–warmup iterations per chain.

  `parameter_names` Named lists of parameter labels.

- `arguments`: A list of function call arguments and metadata (e.g., number of variables, warmup, sampler settings, package version).

The `summary()` method prints formatted posterior summaries, and `coef()` extracts posterior mean matrices.

NUTS diagnostics (tree depth, divergences, energy, E-BFMI) are included in `fit$nuts_diag` if `update_method = "nuts"`.

References

Dahl DB (2009). "Modal clustering in a class of product partition models." *Bayesian Analysis*, **4**(2), 243–264. doi:10.1214/09BA409.

Sekulovski N, Arena G, Haslbeck JMB, Huth KBS, Friel N, Marsman M (2025). "A Stochastic Block Prior for Clustering in Graphical Models." *Retrieved from* https://osf.io/preprints/psyarxiv/29p3m_v1. OSF preprint.

See Also

`vignette("intro", package = "bgms")` for a worked example.

Other model-fitting: `bgmCompare()`

Examples

```
# Run bgm on subset of the Wenchuan dataset
fit = bgm(x = Wenchuan[, 1:5], chains = 2)

# Posterior inclusion probabilities
summary(fit)$indicator

# Posterior pairwise effects
summary(fit)$pairwise
```

---

bgmCompare *Bayesian Estimation and Variable Selection for Group Differences in Markov Random Fields*

Description

The bgmCompare function estimates group differences in category threshold parameters (main effects) and pairwise interactions (pairwise effects) of a Markov Random Field (MRF) for binary and ordinal variables. Groups can be defined either by supplying two separate datasets (x and y) or by a group membership vector. Optionally, Bayesian variable selection can be applied to identify differences across groups.

Usage

```
bgmCompare(
  x,
  y,
  group_indicator,
  difference_selection = TRUE,
  main_difference_selection = FALSE,
  variable_type = "ordinal",
  baseline_category,
  difference_scale = 1,
  difference_prior = bernoulli_prior(0.5),
  difference_probability,
  interaction_prior = cauchy_prior(scale = 1),
  threshold_prior = beta_prime_prior(alpha = 0.5, beta = 0.5),
  iter = 2000,
  warmup = 2000,
  na_action = c("listwise", "impute"),
  update_method = c("nuts", "adaptive-metropolis"),
  target_accept,
  nuts_max_depth = 10,
  learn_mass_matrix = TRUE,
  chains = 4,
```

```
  cores = parallel::detectCores(),
  display_progress = c("per-chain", "total", "none"),
  seed = NULL,
  standardize = FALSE,
  verbose = getOption("bgms.verbose", TRUE),
  progress_callback = NULL,
  pairwise_scale,
  main_alpha,
  main_beta,
  beta_bernoulli_alpha,
  beta_bernoulli_beta,
  main_difference_model,
  reference_category,
  main_difference_scale,
  pairwise_difference_scale,
  pairwise_difference_prior,
  main_difference_prior,
  pairwise_difference_probability,
  main_difference_probability,
  pairwise_beta_bernoulli_alpha,
  pairwise_beta_bernoulli_beta,
  main_beta_bernoulli_alpha,
  main_beta_bernoulli_beta,
  interaction_scale,
  threshold_alpha,
  threshold_beta,
  burnin,
  save
)
```

Arguments

x	A data frame or matrix of binary and ordinal responses for Group 1. Variables should be coded as nonnegative integers starting at 0. For ordinal variables, unused categories are collapsed; for Blume–Capel variables, all categories are retained.
y	Optional data frame or matrix for Group 2 (two-group designs). Must have the same variables (columns) as x.
group_indicator	Optional integer vector of group memberships for rows of x (multi-group designs). Ignored if y is supplied.
difference_selection	Logical. If TRUE, spike-and-slab priors are applied to difference parameters. Default: TRUE.
main_difference_selection	Logical. If TRUE, apply spike-and-slab selection to main effect (threshold) differences. If FALSE, main effect differences are always included (no selection).

Since main effects are often nuisance parameters and their selection can interfere with pairwise selection under the Beta-Bernoulli prior, the default is `FALSE`. Only used when `difference_selection = TRUE`.

`variable_type` Character vector specifying type of each variable: `"ordinal"` (default) or `"blume-capel"`.

`baseline_category`
Integer or vector giving the baseline category for Blume–Capel variables.

`difference_scale`
Double. Scale of the Cauchy prior for difference parameters. Default: `1`.

`difference_prior`
An indicator prior specification object for difference selection, created by one of:

- `bernoulli_prior()`: Fixed inclusion probability (default).
- `beta_bernoulli_prior()`: Beta-distributed inclusion.
- `sbm_prior()`: Stochastic Block Model.

Legacy character strings `"Bernoulli"` and `"Beta-Bernoulli"` are still accepted but deprecated. Default: `bernoulli_prior(0.5)`.

`difference_probability`
[Deprecated] Numeric. Use `difference_prior = bernoulli_prior(probability)` instead. Default: `0.5`.

`interaction_prior`
A prior specification object for baseline pairwise interaction parameters, created by one of the prior constructor functions:

- `cauchy_prior()`: Cauchy(0, scale) prior (default).
- `normal_prior()`: Normal(0, scale) prior.

When supplied, overrides `pairwise_scale`. Default: `cauchy_prior(scale = 1)`.

`threshold_prior`
A prior specification object for threshold (main effect) parameters, created by one of the prior constructor functions:

- `beta_prime_prior()`: Beta-prime prior (default).
- `normal_prior()`: Normal(0, scale) prior.

When supplied, overrides `main_alpha` and `main_beta`. Default: `beta_prime_prior(alpha = 0.5, beta = 0.5)`.

`iter` Integer. Number of post–warmup iterations per chain. Default: `2e3`.

`warmup` Integer. Number of warmup iterations before sampling. Default: `2e3`.

`na_action` Character. How to handle missing data: `"listwise"` (drop rows) or `"impute"` (impute within Gibbs). Default: `"listwise"`.

`update_method` Character. Sampling algorithm: `"adaptive-metropolis"` or `"nuts"`. Default: `"nuts"`.

`target_accept` Numeric between 0 and 1. Target acceptance rate. Defaults: 0.44 (Metropolis), 0.80 (NUTS).

`nuts_max_depth` Integer. Maximum tree depth for NUTS. Default: `10`.

learn_mass_matrix
: Logical. If TRUE, adapts a diagonal mass matrix during warmup (NUTS only). Default: TRUE.

chains
: Integer. Number of parallel chains. Default: 4.

cores
: Integer. Number of CPU cores. Default: parallel::detectCores().

display_progress
: Character. Controls progress reporting: "per-chain", "total", or "none". Default: "per-chain".

seed
: Optional integer. Random seed for reproducibility.

standardize
: Logical. If TRUE, the Cauchy prior scale for each pairwise interaction (both baseline and difference) is adjusted based on the range of response scores. Without standardization, pairs with more response categories experience less shrinkage because their naturally smaller interaction effects make a fixed prior relatively wide. Standardization equalizes relative shrinkage across all pairs, with the interaction_prior (e.g. cauchy_prior(scale)) scale itself applying to the unit interval (binary) case. See bgm for details on the adjustment. Default: FALSE.

verbose
: Logical. If TRUE, prints informational messages during data processing (e.g., missing data handling, variable recoding). Defaults to getOption("bgms.verbose", TRUE). Set options(bgms.verbose = FALSE) to suppress messages globally.

progress_callback
: An optional R function with signature function(completed, total) that is called at regular intervals during sampling, where completed is the number of iterations completed across all chains and total is the total number of iterations. Useful for external front-ends (e.g., JASP) that supply their own progress reporting. When NULL (the default), no callback is invoked.

pairwise_scale
: **[Deprecated]** Double. Scale of the Cauchy prior for baseline pairwise interactions. Use interaction_prior = cauchy_prior(scale) instead.

main_alpha, main_beta
: **[Deprecated]** Doubles. Shape parameters of the beta-prime prior for baseline threshold parameters. Use threshold_prior = beta_prime_prior(alpha, beta) instead.

beta_bernoulli_alpha, beta_bernoulli_beta
: Doubles. Shape parameters of the Beta prior for inclusion probabilities in the Beta–Bernoulli model. Defaults: 1.

main_difference_model, reference_category, pairwise_difference_scale, main_difference_scale, pairwise_difference_prior, main_difference_prior, pairwise_difference_probability, main_difference_probability, pairwise_beta_bernoulli_alpha, pairwise_beta_bernoulli_beta, main_beta_bernoulli_alpha, main_beta_bernoulli_beta, interaction_scale, threshold_alpha, threshold_beta, burnin, save
: **[Deprecated]** Deprecated arguments as of **bgms 0.1.6.0**. Use difference_scale, difference_prior, difference_probability, beta_bernoulli_alpha, beta_bernoulli_beta, baseline_category, interaction_prior, threshold_prior, and warmup instead.

Details

Group-specific parameters are decomposed into a shared baseline plus group differences that sum to zero. Difference selection uses spike-and-slab priors (Bernoulli or Beta-Bernoulli). Parameters are sampled with NUTS (default) or adaptive Metropolis–Hastings, using the same multi-stage warmup schedule as bgm.

For full details on model specification, prior choices, and output interpretation, see the package website at https://bayesian-graphical-modelling-lab.github.io/bgms-docs/.

Value

An S7 object of class bgmCompare supporting list-style $ / [[access for backward compatibility, containing posterior summaries, posterior mean matrices, and raw MCMC samples. Some posterior_summary_* fields are computed lazily on first access:

- posterior_summary_main_baseline, posterior_summary_pairwise_baseline: summaries of baseline thresholds and pairwise interactions.
- posterior_summary_main_differences, posterior_summary_pairwise_differences: summaries of group differences in thresholds and pairwise interactions.
- posterior_summary_indicator: summaries of inclusion indicators (if difference_selection = TRUE).
- posterior_mean_main_baseline, posterior_mean_pairwise_baseline: posterior mean matrices (legacy style).
- raw_samples: list of raw draws per chain for main, pairwise, and indicator parameters.
- arguments: list of function call arguments and metadata.

The summary() method prints formatted summaries, and coef() extracts posterior means.

NUTS diagnostics (tree depth, divergences, energy, E-BFMI) are included in fit$nuts_diag if update_method = "nuts".

References

There are no references for Rd macro \insertAllCites on this help page.

See Also

vignette("comparison", package = "bgms") for a worked example.

Other model-fitting: bgm()

Examples

```
## Not run:
# Run bgmCompare on subset of the Boredom dataset
x = Boredom[Boredom$language == "fr", 2:6]
y = Boredom[Boredom$language != "fr", 2:6]

fit = bgmCompare(x, y, chains = 2)

# Posterior inclusion probabilities
```

```
summary(fit)$indicator

# Bayesian model averaged main effects for the groups
coef(fit)$main_effects_groups

# Bayesian model averaged pairwise effects for the groups
coef(fit)$pairwise_effects_groups

## End(Not run)
```

`Boredom` *Short Boredom Proneness Scale Responses*

Description

This dataset includes responses to the 8-item Short Boredom Proneness Scale (SBPS), a self-report measure of an individual's susceptibility to boredom (Martarelli et al. 2023). Items were rated on a 7-point Likert scale ranging from 1 ("strongly disagree") to 7 ("strongly agree"). The scale was administered in either English (Struk et al. 2015) or French (translated by (Martarelli et al. 2023)).

Usage

```
data("Boredom")
```

Format

A data frame with 986 rows and 9 columns. Each row corresponds to a respondent.

language Language in which the SBPS was administered: "en" = English, "fr" = French

loose_ends I often find myself at "loose ends," not knowing what to do.

entertain I find it hard to entertain myself.

repetitive Many things I have to do are repetitive and monotonous.

stimulation It takes more stimulation to get me going than most people.

motivated I don't feel motivated by most things that I do.

keep_interest In most situations, it is hard for me to find something to do or see to keep me interested.

sit_around Much of the time, I just sit around doing nothing.

half_dead_dull Unless I am doing something exciting, even dangerous, I feel half-dead and dull.

Source

Martarelli et al. (2023). Data retrieved from https://osf.io/qhux8. Licensed under the CC-BY 4.0: https://creativecommons.org/licenses/by/4.0/

References

Martarelli CS, Baillifard A, Audrin C (2023). "A Trait-Based Network Perspective on the Validation of the French Short Boredom Proneness Scale." *European Journal of Psychological Assessment*, **39**(6), 390–399. doi:10.1027/10155759/a000718.

Struk AA, Carriere JSA, Cheyne JA, Danckert J (2015). "A Short Boredom Proneness Scale: Development and Psychometric Properties." *Assessment*, **24**(3), 346–359. doi:10.1177/1073191115609996.

`cauchy_prior` *Cauchy Prior for Model Parameters*

Description

Specifies a Cauchy(0, scale) prior on model parameters. This is the default prior for pairwise interactions in `bgm` and produces heavy-tailed shrinkage toward zero.

Usage

```
cauchy_prior(scale = 1)
```

Arguments

`scale`	Positive numeric. Scale (half-width at half-maximum) of the Cauchy distribution. Default: `1`.

Value

An object of class `"bgms_parameter_prior"` with `family = "cauchy"`.

See Also

`normal_prior`, `beta_prime_prior`, `bgm`

Other prior-constructors: `bernoulli_prior()`, `beta_bernoulli_prior()`, `beta_prime_prior()`, `exponential_prior()`, `gamma_prior()`, `normal_prior()`, `sbm_prior()`

Examples

```
cauchy_prior()
cauchy_prior(scale = 2.5)
```

coef.bgmCompare *Extract Coefficients from a bgmCompare Object*

Description

Returns posterior means for raw parameters (baseline + differences) and group-specific effects from a bgmCompare fit, as well as inclusion indicators.

Usage

```
## S3 method for class 'bgmCompare'
coef(object, ...)
```

Arguments

object	An object of class bgmCompare.
...	Ignored.

Value

A list with components:

main_effects_raw Posterior means of the raw main-effect parameters (variables x (baseline + differences)).

pairwise_effects_raw Posterior means of the raw pairwise-effect parameters (pairs x (baseline + differences)).

main_effects_groups Posterior means of group-specific main effects (variables x groups), computed as baseline plus projected differences.

pairwise_effects_groups Posterior means of group-specific pairwise effects (pairs x groups), computed as baseline plus projected differences.

indicators Posterior mean inclusion probabilities as a symmetric matrix, with diagonals corresponding to main effects and off-diagonals to pairwise effects.

See Also

bgmCompare(), print.bgmCompare(), summary.bgmCompare()

Other posterior-methods: coef.bgms(), print.bgmCompare(), print.bgms(), summary.bgmCompare(), summary.bgms()

Examples

```
# See ?bgmCompare for a full example
```

`coef.bgms` *Extract Coefficients from a bgms Object*

Description

Returns the posterior mean main effects, pairwise effects, and edge inclusion indicators from a bgms model fit.

Usage

```
## S3 method for class 'bgms'
coef(object, ...)
```

Arguments

`object`	An object of class bgms.
`...`	Ignored.

Value

A list with the following components:

main Posterior mean of the main-effect parameters. `NULL` for GGM models (no main effects). For OMRF models this is a numeric matrix (p x max_categories) of category thresholds. For mixed MRF models this is a list with `$discrete` (p x max_categories matrix) and `$continuous` (q x 1 matrix of means).

pairwise Posterior mean of the partial association matrix (zero diagonal). Use `extract_precision()` for the full precision matrix.

indicator Posterior mean of the edge inclusion indicators (if available).

mean_allocations Posterior mean block allocations. Present only for Stochastic-Block edge-prior fits.

mode_allocations Posterior mode block allocations. Present only for Stochastic-Block edge-prior fits.

num_blocks Data frame of the posterior distribution over the number of blocks. Present only for Stochastic-Block edge-prior fits.

See Also

`bgm()`, `print.bgms()`, `summary.bgms()`

Other posterior-methods: `coef.bgmCompare()`, `print.bgmCompare()`, `print.bgms()`, `summary.bgmCompare()`, `summary.bgms()`

Examples

```
fit = bgm(x = Wenchuan[, 1:3])
coef(fit)
```

exponential_prior *Exponential Prior for Scale Parameters*

Description

Specifies an Exponential(rate) prior for positive scale parameters. This is a convenience function equivalent to `gamma_prior(shape = 1, rate = rate)`.

Usage

```
exponential_prior(rate = 1)
```

Arguments

`rate`	Positive numeric. Rate parameter of the Exponential distribution. Default: `1`.

Value

An object of class `"bgms_scale_prior"` with `family = "exponential"`.

See Also

`gamma_prior`, `bgm`

Other prior-constructors: `bernoulli_prior()`, `beta_bernoulli_prior()`, `beta_prime_prior()`, `cauchy_prior()`, `gamma_prior()`, `normal_prior()`, `sbm_prior()`

Examples

```
exponential_prior()
exponential_prior(rate = 2)
```

extract_arguments *Extract Model Arguments*

Description

Retrieves the arguments used when fitting a model with `bgm()` or `bgmCompare()`.

Usage

```
extract_arguments(bgms_object)
```

Arguments

`bgms_object`	A fitted model object of class `bgms` (from `bgm()`) or `bgmCompare` (from `bgmCompare()`).

Value

A named list containing all arguments passed to the fitting function, including data dimensions, prior settings, and MCMC configuration.

See Also

bgm(), bgmCompare(), summary.bgms(), summary.bgmCompare()

Other extractors: extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabili
extract_precision(), extract_rhat(), extract_sbm()

---

extract_category_thresholds

Extract Category Threshold Estimates

Description

[Deprecated]

extract_category_thresholds() was renamed to extract_main_effects() to reflect that main effects include continuous means and precision diagonal (mixed MRF), not only category thresholds.

Usage

```
extract_category_thresholds(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

See extract_main_effects() for details.

See Also

extract_main_effects()

Other extractors: extract_arguments(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabilities(), extract_precision(), extract_rhat(), extract_sbm()

extract_ess *Extract Effective Sample Size*

Description

Retrieves effective sample size estimates for all parameters from a model fitted with bgm() or bgmCompare().

Usage

```
extract_ess(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

A named list with ESS values for each parameter type present in the model (e.g., main, pairwise, indicator).

See Also

bgm(), bgmCompare(), extract_rhat()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabil
extract_precision(), extract_rhat(), extract_sbm()

extract_group_params *Extract Group-Specific Parameters*

Description

Computes group-specific parameter estimates by combining baseline parameters and group differences from a model fitted with bgmCompare().

Usage

```
extract_group_params(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgmCompare (from bgmCompare()).

Value

A list with elements `main_effects_groups` (main effects per group) and `pairwise_effects_groups` (pairwise effects per group).

See Also

`bgmCompare()`, `extract_pairwise_interactions()`, `extract_main_effects()`

Other extractors: `extract_arguments()`, `extract_category_thresholds()`, `extract_ess()`, `extract_indicator_priors()`, `extract_indicators()`, `extract_log_odds()`, `extract_main_effects()`, `extract_pairwise_interactions()`, `extract_partial_correlations()`, `extract_posterior_inclusion_probabil`
`extract_precision()`, `extract_rhat()`, `extract_sbm()`

---

extract_indicator_priors

Extract Indicator Prior Structure

Description

Retrieves the prior specification used for inclusion indicators in a model fitted with `bgm()` (edge indicators) or `bgmCompare()` (difference indicators).

Usage

```
extract_indicator_priors(bgms_object)
```

Arguments

`bgms_object`	A fitted model object of class `bgms` (from `bgm()`) or `bgmCompare` (from `bgmCompare()`).

Value

A named list describing the prior structure, including the prior type and any hyperparameters.

bgms Requires `edge_selection = TRUE`. Returns a list with the prior type (`"Bernoulli"`, `"Beta-Bernoulli"`, or `"Stochastic-Block"`) and associated hyperparameters.

bgmCompare Requires `difference_selection = TRUE`. Returns the difference prior specification.

See Also

`bgm()`, `bgmCompare()`, `extract_indicators()`

Other extractors: `extract_arguments()`, `extract_category_thresholds()`, `extract_ess()`, `extract_group_params()`, `extract_indicators()`, `extract_log_odds()`, `extract_main_effects()`, `extract_pairwise_interactions()`, `extract_partial_correlations()`, `extract_posterior_inclusion_probabil`
`extract_precision()`, `extract_rhat()`, `extract_sbm()`

extract_indicators *Extract Indicator Samples*

Description

Retrieves posterior samples of inclusion indicators from a model fitted with `bgm()` (edge inclusion indicators) or `bgmCompare()` (difference indicators).

Usage

```
extract_indicators(bgms_object)
```

Arguments

`bgms_object`	A fitted model object of class `bgms` (from `bgm()`) or `bgmCompare` (from `bgmCompare()`).

Value

A matrix with one row per post-warmup iteration and one column per indicator, containing binary (0/1) samples.

bgms One column per edge. Requires `edge_selection = TRUE`.

bgmCompare Columns for main-effect and pairwise difference indicators. Requires `difference_selection = TRUE`.

See Also

`bgm()`, `bgmCompare()`, `extract_posterior_inclusion_probabilities()`

Other extractors: `extract_arguments()`, `extract_category_thresholds()`, `extract_ess()`, `extract_group_params()`, `extract_indicator_priors()`, `extract_log_odds()`, `extract_main_effects()`, `extract_pairwise_interactions()`, `extract_partial_correlations()`, `extract_posterior_inclusion_probabili`
`extract_precision()`, `extract_rhat()`, `extract_sbm()`

extract_log_odds *Extract Posterior Mean Log-Odds (Pairwise Interactions)*

Description

Retrieves the posterior mean pairwise interaction matrix for discrete variables from a model fitted with `bgm()`. These are the log-odds parameters of the discrete (Markov random field) block. GGM models have no discrete variables and return `NULL`.

Usage

```
extract_log_odds(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()).

Value

A named numeric matrix of posterior mean log-odds interactions, or NULL for GGM models.

OMRF A symmetric matrix with zero diagonal and one row and column per variable.

Mixed MRF A symmetric matrix with zero diagonal and one row and column per discrete variable.

GGM NULL (invisibly).

See Also

bgm(), extract_pairwise_interactions(), extract_precision()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabil
extract_precision(), extract_rhat(), extract_sbm()

Examples

```
fit = bgm(x = Wenchuan[, 1:3])
extract_log_odds(fit)
```

---

extract_main_effects *Extract Main Effect Estimates*

Description

Retrieves main-effect parameters from a model fitted with bgm() (posterior means) or bgmCompare() (posterior samples of baseline main effects). For OMRF models these are category thresholds; for mixed MRF models these include discrete thresholds and continuous means. GGM models have no main effects and return NULL.

Usage

```
extract_main_effects(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

The structure depends on the model type:

GGM (bgms) NULL (invisibly). GGM models have no main effects; use extract_precision() to obtain the precision matrix.

OMRF (bgms) A numeric matrix with one row per variable and one column per category threshold, containing posterior means. Columns beyond the number of categories for a variable are NA.

Mixed MRF (bgms) A list with two elements:

- **discrete** A numeric matrix (p rows x max_categories columns) of posterior mean thresholds for discrete variables.
- **continuous** A numeric matrix (q rows x 1 column) of posterior mean continuous variable means.

bgmCompare A matrix with one row per post-warmup iteration, containing posterior samples of baseline main-effect parameters.

See Also

bgm(), bgmCompare(), extract_pairwise_interactions(), extract_category_thresholds()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabil
extract_precision(), extract_rhat(), extract_sbm()

Examples

```
fit = bgm(x = Wenchuan[, 1:3])
extract_main_effects(fit)
```

---

extract_pairwise_interactions

Extract Pairwise Interaction Samples

Description

Retrieves posterior samples of pairwise interaction parameters from a model fitted with bgm() or bgmCompare().

Usage

```
extract_pairwise_interactions(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

A matrix with one row per post-warmup iteration and one column per edge, containing posterior samples of interaction strengths.

bgms Columns correspond to all unique variable pairs.

bgmCompare Columns correspond to the baseline pairwise interaction parameters.

See Also

bgm(), bgmCompare(), extract_main_effects()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_partial_correlations(), extract_posterior_inclusion_probabilities(), extract_precision(), extract_rhat(), extract_sbm()

---

extract_partial_correlations

Extract Posterior Mean Partial Correlations

Description

Computes the posterior mean partial correlation matrix from a model fitted with bgm(). For GGM models this is the full matrix. For mixed MRF models this is the matrix for the continuous block. OMRF models have no partial correlations and return NULL.

Partial correlations are computed from the precision matrix as $\rho_{ij} = -\Theta_{ij}/\sqrt{\Theta_{ii}\Theta_{jj}}$.

Usage

```
extract_partial_correlations(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()).

Value

A named numeric matrix containing posterior mean partial correlations, or NULL for OMRF models.

GGM A symmetric matrix with ones on the diagonal and one row and column per variable.

Mixed MRF A symmetric matrix with ones on the diagonal and one row and column per continuous variable.

OMRF NULL (invisibly).

See Also

bgm(), extract_precision()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_posterior_inclusion_probabilities(), extract_precision(), extract_rhat(), extract_sbm()

Examples

```
fit = bgm(
  x = Wenchuan[, 1:3],
  variable_type = rep("continuous", 3)
)
extract_partial_correlations(fit)
```

---

extract_posterior_inclusion_probabilities

Extract Posterior Inclusion Probabilities

Description

Computes posterior inclusion probabilities from a model fitted with bgm() (edge inclusion) or bgmCompare() (difference inclusion).

Usage

```
extract_posterior_inclusion_probabilities(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

A symmetric p x p matrix of posterior inclusion probabilities, with variable names as row and column names.

bgms Off-diagonal entries are edge inclusion probabilities. Requires edge_selection = TRUE.

bgmCompare Diagonal entries are main-effect inclusion probabilities; off-diagonal entries are pairwise difference inclusion probabilities. Requires difference_selection = TRUE.

See Also

bgm(), bgmCompare(), extract_indicators()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_precision(), extract_rhat(), extract_sbm()

---

extract_precision *Extract Posterior Mean Precision Matrix*

Description

Retrieves the posterior mean precision matrix from a model fitted with bgm(). For GGM models this is the full precision matrix. For mixed MRF models this is the precision matrix of the continuous (Gaussian) block. OMRF models have no precision matrix and return NULL.

For mixed MRF models the precision matrix is reconstructed from the internal association-scale parameterization.

Usage

```
extract_precision(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()).

Value

A named numeric matrix containing the posterior mean precision matrix, or NULL for OMRF models.

GGM A symmetric matrix with one row and column per variable.

Mixed MRF A symmetric matrix with one row and column per continuous variable.

OMRF NULL (invisibly).

See Also

bgm(), coef.bgms(), extract_partial_correlations()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabilities(), extract_rhat(), extract_sbm()

Examples

```
fit = bgm(
  x = Wenchuan[, 1:3],
  variable_type = rep("continuous", 3)
)
extract_precision(fit)
```

`extract_rhat` *Extract R-hat Convergence Diagnostics*

Description

Retrieves R-hat convergence diagnostics for all parameters from a model fitted with `bgm()` or `bgmCompare()`.

Usage

```
extract_rhat(bgms_object)
```

Arguments

`bgms_object`	A fitted model object of class `bgms` (from `bgm()`) or `bgmCompare` (from `bgmCompare()`).

Value

A named list with R-hat values for each parameter type present in the model (e.g., `main`, `pairwise`, `indicator`).

See Also

`bgm()`, `bgmCompare()`, `extract_ess()`

Other extractors: `extract_arguments()`, `extract_category_thresholds()`, `extract_ess()`, `extract_group_params()`, `extract_indicator_priors()`, `extract_indicators()`, `extract_log_odds()`, `extract_main_effects()`, `extract_pairwise_interactions()`, `extract_partial_correlations()`, `extract_posterior_inclusion_probabilities()`, `extract_precision()`, `extract_sbm()`

extract_sbm *Extract Stochastic Block Model Summaries*

Description

Retrieves posterior summaries from a model fitted with the Stochastic Block prior. Works on both bgms fits (where SBM governs edge inclusion) and bgmCompare fits (where SBM governs the off-diagonal pairwise difference inclusions).

Usage

```
extract_sbm(bgms_object)
```

Arguments

bgms_object	A fitted model object of class bgms (from bgm()) or bgmCompare (from bgmCompare()).

Value

A list with elements posterior_num_blocks, posterior_mean_allocations, posterior_mode_allocations, and posterior_mean_coclustering_matrix. For bgms, requires edge_selection = TRUE and edge_prior = sbm_prior(...). For bgmCompare, requires difference_selection = TRUE and difference_prior = sbm_prior(...).

See Also

bgm(), bgmCompare(), extract_indicators(), extract_posterior_inclusion_probabilities()

Other extractors: extract_arguments(), extract_category_thresholds(), extract_ess(), extract_group_params(), extract_indicator_priors(), extract_indicators(), extract_log_odds(), extract_main_effects(), extract_pairwise_interactions(), extract_partial_correlations(), extract_posterior_inclusion_probabilities(), extract_precision(), extract_rhat()

gamma_prior *Gamma Prior for Scale Parameters*

Description

Specifies a Gamma(shape, rate) prior for positive scale parameters such as the diagonal elements of the precision matrix. The default gamma_prior(1, 1) corresponds to an Exponential(1) distribution.

Usage

```
gamma_prior(shape = 1, rate = 1)
```

Arguments

shape	Positive numeric. Shape parameter of the Gamma distribution. Default: 1.
rate	Positive numeric. Rate parameter of the Gamma distribution. Default: 1.

Value

An object of class "bgms_scale_prior" with family = "gamma".

See Also

exponential_prior, bgm

Other prior-constructors: bernoulli_prior(), beta_bernoulli_prior(), beta_prime_prior(), cauchy_prior(), exponential_prior(), normal_prior(), sbm_prior()

Examples

```
gamma_prior()
gamma_prior(shape = 2, rate = 0.5)
```

---

normal_prior *Normal Prior for Model Parameters*

Description

Specifies a Normal(0, scale) prior on model parameters. Produces lighter-tailed shrinkage than the Cauchy prior and is better suited for simulation-based calibration (SBC) studies. Can be used for interactions, thresholds, or continuous means.

Usage

```
normal_prior(scale = 1)
```

Arguments

scale	Positive numeric. Standard deviation of the normal distribution. Default: 1.

Value

An object of class "bgms_parameter_prior" with family = "normal".

See Also

cauchy_prior, beta_prime_prior, bgm

Other prior-constructors: bernoulli_prior(), beta_bernoulli_prior(), beta_prime_prior(), cauchy_prior(), exponential_prior(), gamma_prior(), sbm_prior()

Examples

```
normal_prior()
normal_prior(scale = 0.5)
```

`predict.bgmCompare` *Predict Conditional Probabilities from a Fitted bgmCompare Model*

Description

Computes conditional probability distributions for one or more variables given the observed values of other variables in the data, using group-specific parameters from a `bgmCompare` model.

Usage

```
## S3 method for class 'bgmCompare'
predict(
  object,
  newdata,
  group,
  variables = NULL,
  type = c("probabilities", "response"),
  method = c("posterior-mean"),
  ...
)
```

Arguments

`object`	An object of class `bgmCompare`.
`newdata`	A matrix or data frame with `n` rows and `p` columns containing the observed data. Must have the same variables (columns) as the original data used to fit the model.
`group`	Integer specifying which group's parameters to use for prediction (1 to number of groups). Required argument.
`variables`	Which variables to predict. Can be: • A character vector of variable names • An integer vector of column indices • `NULL` (default) to predict all variables
`type`	Character string specifying the type of prediction: `"probabilities"` Return the full conditional probability distribution for each variable and observation. `"response"` Return the predicted category (mode of the conditional distribution).
`method`	Character string specifying which parameter estimates to use: `"posterior-mean"` Use posterior mean parameters.
`...`	Additional arguments (currently ignored).

Details

Group-specific parameters are obtained by applying the projection matrix to convert baseline parameters and differences into group-level estimates. The function then computes the conditional distribution of target variables given the observed values of all other variables.

Value

For `type = "probabilities"`: A named list with one element per predicted variable. Each element is a matrix with `n` rows and `num_categories + 1` columns containing $P(X_j = c|X_{-j})$ for each observation and category.

For `type = "response"`: A matrix with `n` rows and `length(variables)` columns containing predicted categories.

See Also

`predict.bgms` for predicting from single-group models, `simulate.bgmCompare` for simulating from group-comparison models.

Other prediction: `predict.bgms()`, `simulate.bgmCompare()`, `simulate.bgms()`, `simulate_mrf()`

Examples

```
# Fit a comparison model
x = Boredom[Boredom$language == "fr", 2:6]
y = Boredom[Boredom$language != "fr", 2:6]
fit = bgmCompare(x, y, chains = 2)

# Predict conditional probabilities using group 1 parameters
probs_g1 = predict(fit, newdata = x[1:10, ], group = 1)

# Predict responses using group 2 parameters
pred_g2 = predict(fit, newdata = y[1:10, ], group = 2, type = "response")
```

`predict.bgms` *Predict Conditional Probabilities from a Fitted bgms Model*

Description

Computes conditional probability distributions for one or more variables given the observed values of other variables in the data. Supports ordinal, Blume-Capel, continuous (GGM), and mixed MRF models.

Usage

```
## S3 method for class 'bgms'
predict(
  object,
  newdata,
  variables = NULL,
  type = c("probabilities", "response"),
  method = c("posterior-mean", "posterior-sample"),
  ndraws = NULL,
  seed = NULL,
  ...
)
```

Arguments

`object`	An object of class `bgms`.
`newdata`	A matrix or data frame with `n` rows and `p` columns containing the observed data. Must have the same variables (columns) as the original data used to fit the model.
`variables`	Which variables to predict. Can be: • A character vector of variable names • An integer vector of column indices • `NULL` (default) to predict all variables
`type`	Character string specifying the type of prediction: `"probabilities"` Return the full conditional probability distribution for each variable and observation. `"response"` Return the predicted category (mode of the conditional distribution).
`method`	Character string specifying which parameter estimates to use: `"posterior-mean"` Use posterior mean parameters. `"posterior-sample"` Average predictions over posterior draws.
`ndraws`	Number of posterior draws to use when `method = "posterior-sample"`. If `NULL`, uses all available draws.
`seed`	Optional random seed for reproducibility when `method = "posterior-sample"`.
`...`	Additional arguments (currently ignored).

Details

For each observation, the function computes the conditional distribution of the target variable(s) given the observed values of all other variables. This is the same conditional distribution used internally by the Gibbs sampler.

For GGM (continuous) models, the conditional distribution of $X_j|X_{-j}$ is Gaussian with mean $-\omega_{jj}^{-1}\sum_{k\neq j}\omega_{jk}x_k$ and variance ω_{jj}^{-1}, where Ω is the precision matrix.

Value

Ordinal models:

For `type = "probabilities"`: A named list with one element per predicted variable. Each element is a matrix with `n` rows and `num_categories + 1` columns containing $P(X_j = c|X_{-j})$ for each observation and category.

For `type = "response"`: A matrix with `n` rows and `length(variables)` columns containing predicted categories.

When `method = "posterior-sample"`, probabilities are averaged over posterior draws, and an attribute `"sd"` is included containing the standard deviation across draws.

GGM (continuous) models:

For `type = "probabilities"`: A named list with one element per predicted variable. Each element is a matrix with `n` rows and 2 columns (`"mean"` and `"sd"`) containing the conditional Gaussian parameters $E(X_j|X_{-j})$ and $\text{SD}(X_j|X_{-j})$.

For `type = "response"`: A matrix with `n` rows and `length(variables)` columns containing conditional means.

When `method = "posterior-sample"`, conditional parameters are averaged over posterior draws, and an attribute `"sd"` is included.

Mixed MRF models:

For mixed models, the return list contains elements for both discrete and continuous predicted variables. Discrete variables return probability matrices (as in ordinal models); continuous variables return conditional mean and SD matrices (as in GGM models).

See Also

`simulate.bgms` for generating new data from the model.

Other prediction: `predict.bgmCompare()`, `simulate.bgmCompare()`, `simulate.bgms()`, `simulate_mrf()`

Examples

```
# Fit a model
fit = bgm(x = Wenchuan[, 1:5], chains = 2)

# Compute conditional probabilities for all variables
probs = predict(fit, newdata = Wenchuan[1:10, 1:5])

# Predict the first variable only
probs_v1 = predict(fit, newdata = Wenchuan[1:10, 1:5], variables = 1)

# Get predicted categories
pred_class = predict(fit, newdata = Wenchuan[1:10, 1:5], type = "response")
```

print.bgmCompare *Print method for* bgmCompare *objects*

Description

Minimal console output for bgmCompare fit objects.

Usage

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

Arguments

x	An object of class bgmCompare.
...	Ignored.

Value

Invisibly returns x.

See Also

bgmCompare(), summary.bgmCompare(), coef.bgmCompare()

Other posterior-methods: coef.bgmCompare(), coef.bgms(), print.bgms(), summary.bgmCompare(), summary.bgms()

Examples

```
# See ?bgmCompare for a full example
```

print.bgms *Print method for* bgms *objects*

Description

Minimal console output for bgms fit objects.

Usage

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

Arguments

x	An object of class bgms.
...	Ignored.

Value

Invisibly returns x.

See Also

bgm(), summary.bgms(), coef.bgms()

Other posterior-methods: coef.bgmCompare(), coef.bgms(), print.bgmCompare(), summary.bgmCompare(), summary.bgms()

Examples

```
fit = bgm(x = Wenchuan[, 1:3])
print(fit)
```

---

sample_ggm_prior *Sample from the GGM (Partial-Association) Prior*

Description

Draws from the prior of a Gaussian graphical model. The likelihood is omitted ($n = 0$, $S = 0$), so the chain targets the prior alone. Two specifications are supported via the spec argument:

- "conditional" (default): fix a graph Γ and sample $K \mid \Gamma$ via the same constrained NUTS sampler that drives bgm for continuous data. The chain targets $p(K \mid \Gamma) \propto \text{slab}(K) \cdot \text{diag}(K) \cdot |K|^{\delta} \cdot \mathbf{1}\{K \in \mathcal{M}^{+}(\Gamma)\}/Z(\Gamma)$.
- "joint": sample (K, Γ) jointly from the un-normalised joint prior $p(K, \Gamma) \propto \text{slab}(K) \cdot \text{diag}(K) \cdot |K|^{\delta} \cdot \mathbf{1}\{K \in \mathcal{M}^{+}(\Gamma)\} \cdot \pi(\Gamma)$. Uses the adaptive-Metropolis MH chain from bgm with edge selection on and the likelihood off, so the marginal on Γ is $\pi(\Gamma) \cdot Z(\Gamma)$ (joint specification, not hierarchical). Useful for simulation-based calibration of bgm's default sampler.

Usage

```
sample_ggm_prior(
  p,
  n_samples,
  n_warmup = 2000,
  interaction_prior = cauchy_prior(scale = 2.5),
  precision_scale_prior = gamma_prior(shape = 1, rate = 1),
  step_size = 0.1,
  max_depth = 10L,
```

```
  seed = 1L,
  verbose = TRUE,
  edge_indicators = NULL,
  delta = NULL,
  spec = c("conditional", "joint"),
  edge_inclusion_prob = 0.5
)
```

Arguments

p	Integer. Dimension of the precision matrix ($p \geq 2$).
n_samples	Integer. Number of post-warmup draws to keep.
n_warmup	Integer. NUTS warmup iterations. Default 2000.
interaction_prior	A bgms_parameter_prior for the partial-association off-diagonals $K_{yy,ij} = -K_{ij}/2$. Use cauchy_prior() or normal_prior(); beta_prime_prior() is not supported here. Default: cauchy_prior(scale = 2.5) (i.e. K_{ij} has an implied Cauchy$(0, 5)$ prior).
precision_scale_prior	A bgms_scale_prior for $K_{ii}/2$. Use gamma_prior() or exponential_prior(). Default: gamma_prior(1, 1), which implies $K_{ii}/2 \sim \text{Exp}(1)$ and therefore $K_{ii} \sim \text{Exp}(1/2)$ (mean 2).
step_size	Positive numeric. Initial NUTS step size used to seed dual-averaging adaptation. Default 0.1. Used only for spec = "conditional" (NUTS path); ignored for the "joint" MH path.
max_depth	Integer. Maximum NUTS tree depth. Default 10. Used only for spec = "conditional".
seed	Integer. RNG seed for the chain. Default 1L.
verbose	Logical. If TRUE (default), print a progress bar.
edge_indicators	Optional integer $p \times p$ matrix with 1 = edge included, 0 = excluded. Must be symmetric with 1s on the diagonal. Default: full graph (all edges included). Used only for spec = "conditional" (the chain samples $K \mid \Gamma$); ignored for spec = "joint".
delta	Non-negative numeric, or NULL for the dimension- adaptive default. Determinant-tilt exponent: multiplies the prior by $\lvert K\rvert^{\delta}$, softly repelling the chain from the positive-definite cone boundary. delta = NULL (default) auto-resolves to $0.5 \log(p)$, the simple form of the dimension-adaptive rule $\delta(p) = c \log p$ with $c \in (0.3, 0.6)$ discussed in the companion paper on determinant-tilted spike-and-slab priors (Marsman et al., in preparation). Pass delta = 0 for the untilted prior (the companion-paper baseline) or a non-negative numeric to override.
spec	One of "conditional" (default, sample $K \mid \Gamma$ at fixed Γ) or "joint" (sample (K, Γ) jointly from the un-normalised joint prior).
edge_inclusion_prob	Probability in $(0, 1)$ for the Bernoulli edge prior used when spec = "joint". Default 0.5. Ignored when spec = "conditional".

Details

The priors are specified on the partial-association scale $K_{yy} = -K/2$: `interaction_prior` acts on $K_{yy,ij} = -K_{ij}/2$, and `precision_scale_prior` acts on $-K_{yy,ii} = K_{ii}/2$. The same convention is used by `bgm` and by the continuous block of the mixed-MRF model, so a prior argument passed here means the same distribution it would mean there. Output samples are reported as entries of K; convert with $K_{yy} = -K/2$ if you want them on the partial-association scale.

When `spec = "conditional"` and `edge_indicators` is supplied, off-diagonals at excluded positions are constrained to zero throughout the chain. `edge_indicators` is ignored when `spec = "joint"` (the chain samples Γ).

Value

A list with components

`K_offdiag` Numeric matrix of size `n_samples` x `p * (p - 1) / 2` containing the upper-triangle off-diagonal entries of K for each draw, in row-major order (the upper triangle traversed by row) $(K_{12}, K_{13}, \ldots, K_{1p}, K_{23}, K_{24}, \ldots, K_{2p}, K_{34}, \ldots)$. Under `spec = "conditional"`, excluded edges are returned as `0`; under `spec = "joint"`, off-diagonals at excluded edges are sampled at `0` per the inclusion indicator.

`K_diag` Numeric matrix of size `n_samples` x `p` containing the diagonal entries $K_{11}, \ldots, K_{pp}$.

`offdiag_names` Character vector of length `p * (p - 1) / 2` naming the columns of `K_offdiag` (e.g. `"K_1_2"`).

`diag_names` Character vector of length `p` naming the columns of `K_diag`.

`edge_indicators` Under `spec = "conditional"`, the `p` x `p` integer matrix of fixed inclusion indicators used (full graph if not supplied). Under `spec = "joint"`, an `n_samples` x `p(p-1)/2` integer matrix of sampled Γ_{ij} indicators (column order matches `K_offdiag`).

See Also

`cauchy_prior`, `normal_prior`, `gamma_prior`, `exponential_prior`, `bgm`

Examples

```
# Default Cauchy(0, 2.5) off-diagonal, Gamma(1, 1) diagonal, p = 4.
draws = sample_ggm_prior(
  p = 4, n_samples = 200, n_warmup = 200,
  verbose = FALSE
)
dim(draws$K_offdiag) # 200 x 6
colnames(draws$K_offdiag) = draws$offdiag_names
head(draws$K_offdiag)

# Sparser graph: drop the (1, 4) edge.
E = matrix(1L, 4, 4)
E[1, 4] = E[4, 1] = 0L
draws = sample_ggm_prior(
  p = 4, n_samples = 200, n_warmup = 200,
  edge_indicators = E, verbose = FALSE
)
```

```
colnames(draws$K_offdiag) = draws$offdiag_names
all(draws$K_offdiag[, "K_1_4"] == 0) # TRUE
```

`sbm_prior` *Stochastic Block Model Prior for Inclusion Indicators*

Description

Specifies a Stochastic Block Model (SBM) prior for inclusion indicators. Variables are assigned to latent clusters, with separate Beta priors on within-cluster and between-cluster inclusion probabilities.

Usage

```
sbm_prior(
  alpha = 1,
  beta = 1,
  alpha_between = 1,
  beta_between = 1,
  dirichlet_alpha = 1,
  lambda = 1
)
```

Arguments

`alpha`	Positive numeric. First shape parameter of the Beta distribution for within-cluster edges. Default: `1`.
`beta`	Positive numeric. Second shape parameter of the Beta distribution for within-cluster edges. Default: `1`.
`alpha_between`	Positive numeric. First shape parameter of the Beta distribution for between-cluster edges. Default: `1`.
`beta_between`	Positive numeric. Second shape parameter of the Beta distribution for between-cluster edges. Default: `1`.
`dirichlet_alpha`	Positive numeric. Concentration parameter of the Dirichlet prior on cluster assignments. Default: `1`.
`lambda`	Positive numeric. Rate parameter of the zero-truncated Poisson prior on the number of clusters. Default: `1`.

Value

An object of class `"bgms_indicator_prior"` with `family = "Stochastic-Block"`.

See Also

bernoulli_prior, beta_bernoulli_prior, bgm

Other prior-constructors: bernoulli_prior(), beta_bernoulli_prior(), beta_prime_prior(), cauchy_prior(), exponential_prior(), gamma_prior(), normal_prior()

Examples

```
sbm_prior()
sbm_prior(alpha = 2, beta = 1, alpha_between = 1, beta_between = 5)
```

---

simulate.bgmCompare *Simulate Data from a Fitted bgmCompare Model*

Description

Generates new observations from the Markov Random Field model for a specified group using the estimated parameters from a fitted bgmCompare object.

Usage

```
## S3 method for class 'bgmCompare'
simulate(
  object,
  nsim = 500,
  seed = NULL,
  group,
  method = c("posterior-mean"),
  iter = 1000,
  ...
)
```

Arguments

object	An object of class bgmCompare.
nsim	Number of observations to simulate. Default: 500.
seed	Optional random seed for reproducibility.
group	Integer specifying which group to simulate from (1 to number of groups). Required argument.
method	Character string specifying which parameter estimates to use:
	"posterior-mean" Use posterior mean parameters (faster, single simulation).
iter	Number of Gibbs iterations for equilibration before collecting samples. Default: 1000.
...	Additional arguments (currently ignored).

Details

Group-specific parameters are obtained by applying the projection matrix to convert baseline parameters and differences into group-level estimates: `group_param = baseline + projection[group, ] %*% differences`.

The function then uses these group-specific interaction and threshold parameters to generate new data via Gibbs sampling.

Value

A matrix with `nsim` rows and `p` columns containing simulated observations for the specified group.

See Also

`simulate.bgms` for simulating from single-group models, `predict.bgmCompare` for computing conditional probabilities.

Other prediction: `predict.bgmCompare()`, `predict.bgms()`, `simulate.bgms()`, `simulate_mrf()`

Examples

```
# Fit a comparison model
x = Boredom[Boredom$language == "fr", 2:6]
y = Boredom[Boredom$language != "fr", 2:6]
fit = bgmCompare(x, y, chains = 2)

# Simulate 100 observations from group 1
new_data_g1 = simulate(fit, nsim = 100, group = 1)

# Simulate 100 observations from group 2
new_data_g2 = simulate(fit, nsim = 100, group = 2)
```

---

simulate.bgms *Simulate Data from a Fitted bgms Model*

Description

Generates new observations from the Markov Random Field model using the estimated parameters from a fitted `bgms` object. Supports ordinal, Blume-Capel, continuous (GGM), and mixed MRF models.

Usage

```
## S3 method for class 'bgms'
simulate(
  object,
  nsim = 500,
```

```
  seed = NULL,
  method = c("posterior-mean", "posterior-sample"),
  ndraws = NULL,
  iter = 1000,
  cores = parallel::detectCores(),
  display_progress = c("per-chain", "total", "none"),
  ...
)
```

Arguments

`object`	An object of class `bgms`.
`nsim`	Number of observations to simulate. Default: `500`.
`seed`	Optional random seed for reproducibility.
`method`	Character string specifying which parameter estimates to use: `"posterior-mean"` Use posterior mean parameters (faster, single simulation). `"posterior-sample"` Sample from posterior draws, producing one dataset per draw (accounts for parameter uncertainty). This method uses parallel processing when `cores` > 1.
`ndraws`	Number of posterior draws to use when `method = "posterior-sample"`. If `NULL`, uses all available draws.
`iter`	Number of Gibbs iterations for equilibration before collecting samples. Default: `1000`.
`cores`	Number of CPU cores for parallel execution when `method = "posterior-sample"`. Default: `parallel::detectCores()`.
`display_progress`	Character string specifying the type of progress bar. Options: `"per-chain"`, `"total"`, `"none"`. Default: `"per-chain"`.
`...`	Additional arguments (currently ignored).

Details

This function uses the estimated interaction and threshold parameters to generate new data via Gibbs sampling. When `method = "posterior-sample"`, parameter uncertainty is propagated to the simulated data by using different posterior draws. Parallel processing is available for this method via the `cores` argument.

Value

If `method = "posterior-mean"`: A matrix with `nsim` rows and `p` columns containing simulated observations.

If `method = "posterior-sample"`: A list of matrices, one per posterior draw, each with `nsim` rows and `p` columns.

For mixed MRF models, discrete columns contain non-negative integers and continuous columns contain real-valued observations, ordered as in the original data.

See Also

predict.bgms for computing conditional probabilities, simulate_mrf for simulation with user-specified parameters.

Other prediction: predict.bgmCompare(), predict.bgms(), simulate.bgmCompare(), simulate_mrf()

Examples

```
# Fit a model
fit = bgm(x = Wenchuan[, 1:5], chains = 2)

# Simulate 100 new observations using posterior means
new_data = simulate(fit, nsim = 100)

# Simulate with parameter uncertainty (10 datasets)
new_data_list = simulate(
  fit,
  nsim = 100,
  method = "posterior-sample", ndraws = 10
)

# Use parallel processing for faster simulation
new_data_list = simulate(fit,
  nsim = 100, method = "posterior-sample",
  ndraws = 100, cores = 2
)
```

---

simulate_mrf *Simulate Observations from a Markov Random Field*

Description

simulate_mrf() generates observations from a Markov Random Field using user-specified parameters. For ordinal and Blume-Capel variables, observations are generated via Gibbs sampling. For continuous variables (Gaussian graphical model), observations are drawn directly from the multivariate normal distribution implied by the precision matrix.

Usage

```
simulate_mrf(
  num_states,
  num_variables,
  num_categories,
  pairwise,
  main,
  variable_type = "ordinal",
  baseline_category,
```

```
  iter = 1000,
  seed = NULL
)
```

Arguments

num_states	The number of observations to be generated.
num_variables	The number of variables in the MRF.
num_categories	Either a positive integer or a vector of positive integers of length num_variables. The number of response categories on top of the base category: num_categories = 1 generates binary states. Only used for ordinal and Blume-Capel variables; ignored when variable_type = "continuous".
pairwise	A symmetric num_variables by num_variables matrix. For ordinal and Blume-Capel variables, this contains the pairwise interaction parameters; only the off-diagonal elements are used. For continuous variables, this is the precision matrix Ω (including diagonal) and must be positive definite.
main	For ordinal and Blume-Capel variables: a num_variables by max(num_categories) matrix of category thresholds. The elements in row i indicate the thresholds of variable i. If num_categories is a vector, only the first num_categories[i] elements are used in row i. If the Blume-Capel model is used for the category thresholds for variable i, then row i requires two values (details below); the first is α, the linear contribution of the Blume-Capel model and the second is β, the quadratic contribution. For continuous variables: a numeric vector of length num_variables containing the means μ for each variable. Defaults to zeros if not supplied (missing(main)).
variable_type	What kind of variables are simulated? Can be a single character string specifying the variable type of all p variables at once or a vector of character strings of length p specifying the type for each variable separately. Currently, bgm supports "ordinal", "blume-capel", and "continuous". Binary variables are automatically treated as "ordinal". Ordinal and Blume-Capel variables can be mixed freely, but continuous variables cannot be mixed with ordinal or Blume-Capel variables. When variable_type = "continuous", the function simulates from a Gaussian graphical model. Defaults to variable_type = "ordinal".
baseline_category	An integer vector of length num_variables specifying the baseline_category category that is used for the Blume-Capel model (details below). Can be any integer value between 0 and num_categories (or num_categories[i]).
iter	The number of iterations used by the Gibbs sampler (ordinal/Blume-Capel variables only). The function provides the last state of the Gibbs sampler as output. Ignored for continuous variables. By default set to 1e3.
seed	Optional integer seed for reproducibility. If NULL, a seed is generated from R's random number generator (so set.seed() can be used before calling this function).

Details

Ordinal / Blume-Capel variables: The Gibbs sampler is initiated with random values from the response options, after which it proceeds by simulating states for each variable from its full conditional distribution given the other variable states.

Continuous variables (GGM): Observations are drawn from $N(\mu, \Omega^{-1})$ where Ω is the precision matrix specified via `pairwise` and μ is the means vector specified via `main`. No Gibbs sampling is needed; `iter` is ignored.

There are two modeling options for the category thresholds. The default option assumes that the category thresholds are free, except that the first threshold is set to zero for identification. The user then only needs to specify the thresholds for the remaining response categories. This option is useful for any type of ordinal variable and gives the user the most freedom in specifying their model.

The Blume-Capel option is specifically designed for ordinal variables that have a special type of baseline_category category, such as the neutral category in a Likert scale. The Blume-Capel model specifies the following quadratic model for the threshold parameters:

$$\mu_{\mathrm{c}} = \alpha(\mathrm{c} - \mathrm{r}) + \beta(\mathrm{c} - \mathrm{r})^2$$

where μ_{c} is the threshold for category c (which now includes zero), α offers a linear trend across categories (increasing threshold values if $\alpha > 0$ and decreasing threshold values if $\alpha < 0$), if $\beta < 0$, it offers an increasing penalty for responding in a category further away from the baseline_category category r, while $\beta > 0$ suggests a preference for responding in the baseline_category category.

Value

A `num_states` by `num_variables` matrix of simulated observations. For ordinal/Blume-Capel variables, entries are non-negative integers. For continuous variables, entries are real-valued.

See Also

`simulate.bgms` for simulating from a fitted model.

Other prediction: `predict.bgmCompare()`, `predict.bgms()`, `simulate.bgmCompare()`, `simulate.bgms()`

Examples

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

Pairwise = matrix(0, nrow = num_variables, ncol = num_variables)
Pairwise[2, 1] = Pairwise[4, 1] = Pairwise[3, 2] =
  Pairwise[5, 2] = Pairwise[5, 4] = .25
Pairwise = Pairwise + t(Pairwise)
Main = matrix(0, nrow = num_variables, ncol = max(num_categories))

x = simulate_mrf(
  num_states = 1e3,
  num_variables = num_variables,
  num_categories = num_categories,
```

```
  pairwise = Pairwise,
  main = Main
)

# Generate responses from a network of 2 ordinal and 3 Blume-Capel variables.
num_variables = 5
num_categories = 4

Pairwise = matrix(0, nrow = num_variables, ncol = num_variables)
Pairwise[2, 1] = Pairwise[4, 1] = Pairwise[3, 2] =
  Pairwise[5, 2] = Pairwise[5, 4] = .25
Pairwise = Pairwise + t(Pairwise)

Main = matrix(NA, num_variables, num_categories)
Main[, 1] = -1
Main[, 2] = -1
Main[3, ] = sort(-abs(rnorm(4)), decreasing = TRUE)
Main[5, ] = sort(-abs(rnorm(4)), decreasing = TRUE)

x = simulate_mrf(
  num_states = 1e3,
  num_variables = num_variables,
  num_categories = num_categories,
  pairwise = Pairwise,
  main = Main,
  variable_type = c("b", "b", "o", "b", "o"),
  baseline_category = 2
)

# Generate responses from a Gaussian graphical model (GGM) with 4 variables.
num_variables = 4

# Precision matrix (symmetric, positive definite)
Omega = diag(c(1, 1.2, 0.8, 1.5))
Omega[2, 1] = Omega[1, 2] = 0.3
Omega[3, 1] = Omega[1, 3] = 0.3
Omega[4, 2] = Omega[2, 4] = -0.2

x = simulate_mrf(
  num_states = 500,
  num_variables = num_variables,
  pairwise = Omega,
  variable_type = "continuous"
)
```

`summary.bgmCompare` *Summary method for* `bgmCompare` *objects*

Description

Returns posterior summaries and diagnostics for a fitted `bgmCompare` model.

Usage

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

Arguments

object	An object of class bgmCompare.
...	Currently ignored.

Value

An object of class summary.bgmCompare with posterior summaries.

See Also

bgmCompare(), print.bgmCompare(), coef.bgmCompare()

Other posterior-methods: coef.bgmCompare(), coef.bgms(), print.bgmCompare(), print.bgms(), summary.bgms()

Examples

```
# See ?bgmCompare for a full example
```

---

summary.bgms *Summary method for* bgms *objects*

Description

Returns posterior summaries and diagnostics for a fitted bgms model.

Usage

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

Arguments

object	An object of class bgms.
...	Currently ignored.

Value

An object of class summary.bgms with posterior summaries.

See Also

bgm(), print.bgms(), coef.bgms()

Other posterior-methods: coef.bgmCompare(), coef.bgms(), print.bgmCompare(), print.bgms(), summary.bgmCompare()

Examples

```
fit = bgm(x = Wenchuan[, 1:3])
summary(fit)
```

Wenchuan *PTSD Symptoms in Wenchuan Earthquake Survivors Who Lost a Child*

Description

This dataset contains responses to 17 items assessing symptoms of post-traumatic stress disorder (PTSD) in Chinese adults who survived the 2008 Wenchuan earthquake and lost at least one child in the disaster (McNally et al. 2015). Participants completed the civilian version of the Posttraumatic Checklist, with each item corresponding to a DSM-IV PTSD symptom. Items were rated on a 5-point Likert scale from "not at all" to "extremely," indicating the degree to which the symptom bothered the respondent in the past month.

Usage

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

Format

A matrix with 362 rows and 17 columns. Each row represents a participant.

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

https://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*, **6**, 836–849. doi:10.1177/2167702614553230.

Index