Package: dmrfit (via r-universe)

October 21, 2026

Date 2026-10-21

Type Package

Title Scalable Bayesian Inference for Discrete Markov Random Fields

Version 0.1.1

Description Estimation of discrete Markov random fields through the pseudo-likelihood, for both Ising models (Ising (1925) <doi:10.1007/BF02980577>) and ordinal Markov random fields (Marsman et al. (2025) <doi:10.1017/psy.2024.4>). Point estimates, either maximum pseudo-likelihood or maximum a posteriori, are obtained through a trust region optimization algorithm, with robust standard errors from the Godambe-Huber-White (GHW, or sandwich) estimator, for both full and constrained network structures. Bayesian inference corrects the pseudo-posterior through coordinate rescaling (Arena and Marsman (2026) <doi:10.48550/arXiv.2601.17205>), which matches its covariance to the GHW covariance, and samples from it with the Fisher adaptive Metropolis-adjusted Langevin algorithm (Titsias (2024) <doi:10.48550/arXiv.2305.14442>). For comparison, the posterior based on the full likelihood can also be sampled, exactly for small networks or with the double Metropolis-Hastings algorithm (Liang (2010) <doi:10.1080/00949650902882162>). Savage-Dickey Bayes factors test each pairwise interaction, computed either by sampling importance resampling (Skare et al. (2003) <doi:10.1111/1467-9469.00360>) or from the posterior draws.

License MIT + file LICENSE

Copyright The dataset 'rads2' is licensed under CC BY 4.0; see file COPYRIGHTS.

URL https://jupepis.github.io/dmrfit/, https://github.com/jupepis/dmrfit

BugReports https://github.com/jupepis/dmrfit/issues

Depends R (>= 4.0.0)

Imports ggplot2, grDevices, graphics, parallel, Rcpp (>= 1.1.0), stats

Suggests igraph, knitr, rmarkdown, tinytest

VignetteBuilder knitr

LinkingTo Rcpp, RcppArmadillo, RcppProgress

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

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

Date/Publication 2026-10-21 12:42:07 UTC

RemoteUrl https://github.com/jupepis/dmrfit

RemoteRef HEAD

RemoteSha e898bf9bd798a9558ea6d8d57d215752bec5c15c

Contents

confint.dmrfit 2
dmrfit 3
dmrfit_bayes 7
plot.dmrfit 9
print.dmrfit 13
print.dmrfit_bayes 13
print.summary.dmrfit 14
print.summary.dmrfit_bayes 14
rads2 15
summary.dmrfit 16
summary.dmrfit_bayes 17

Index **18**

`confint.dmrfit` *Confidence and credible intervals of a fitted discrete Markov random field*

Description

Intervals for the parameters of a `dmrfit` or `dmrfit_bayes` fit. For a `dmrfit` fit, the adjusted likelihood-ratio intervals computed with `dmrfit(..., lrt_intervals = TRUE)` (`method = "lrt"`, see `dmrfit`), or Wald intervals from the Godambe-Huber-White (GHW) standard errors (`method = "wald"`). For a `dmrfit_bayes` fit, highest posterior density intervals of the posterior draws.

Usage

```
## S3 method for class 'dmrfit'
confint(object, parm = NULL, level = NULL, method = c("lrt", "wald"), ...)
```

Arguments

`object`	a `dmrfit` or `dmrfit_bayes` object.
`parm`	the parameters, by name (as in `names(object$argument)`) or position. By default, all free parameters (for a constrained fit, the thresholds and the interactions present in the structure).
`level`	confidence or credible level. By default, the level of a `dmrfit` fit (its `level` argument), and 0.95 for a `dmrfit_bayes` fit. For `method = "lrt"`, it must be the level of the fit.
`method`	for a `dmrfit` fit: `"lrt"` (default) or `"wald"`. Ignored for a `dmrfit_bayes` fit.
`...`	further arguments (currently unused).

Value

a matrix with one row per parameter and the lower and upper bounds as columns.

See Also

`dmrfit`, `dmrfit_bayes`

Examples

```
data(rads2)
dysphoria <- names(which(attr(rads2, "clusters") == "Dysphoria"))
fit <- dmrfit(rads2[, dysphoria], lrt_intervals = c("mu[1,3]", "theta[2,1]"))
confint(fit, parm = c("mu[1,3]", "theta[2,1]"))
confint(fit, parm = c("mu[1,3]", "theta[2,1]"), method = "wald")
```

`dmrfit` *Point estimation and Bayes factors for discrete Markov random fields*

Description

Fits a discrete Markov random field (Ising or ordinal) through the pseudo-likelihood. The estimates maximize the pseudo-likelihood, or the pseudo-posterior with `with_prior = TRUE`, and are obtained with a trust region algorithm; the standard errors come from the Godambe-Huber-White (GHW, or sandwich) estimator. Optionally, Savage-Dickey Bayes factors are computed for each pairwise interaction from the coordinate-rescaled pseudo-posterior (Arena and Marsman, 2026).

Usage

```
dmrfit(
  data,
  parinit = NULL,
  structure = NULL,
  with_prior = FALSE,
  savage_dickey = FALSE,
  M = 1000,
  oversampling = 10,
  ncores = 1,
  thresholds_alpha = 0.5,
  thresholds_beta = 0.5,
  interactions_location = 0,
  interactions_scale = 2.5,
  proposal_df = 5,
  lrt_intervals = FALSE,
  level = 0.95,
  seed = 123
)
```

Arguments

`data` data matrix, with rows as samples and columns as variables. Each variable should be rescaled to the range of 0 to m-1, where m is the number of categories for that variable. The baseline category is always the minimum value in the variable. The internal processing will check if the variables are rescaled and will rescale them if necessary. If there are any NAs in the data, they will be removed before optimization (listwise deletion).

`parinit` initial parameter values for the optimization, a vector of length equal to the number of parameters in the model, `sum(n_categories - 1) + P * (P - 1) / 2`, where `P` is the number of variables. If NULL (default), a vector of zeros.

`structure` network structure, a P x P symmetric matrix with 1 for an edge and 0 for no edge. If NULL (default), the network is fully connected.

`with_prior` logical, whether to include the prior in the optimization: a Beta-Prime prior on the thresholds and a Cauchy prior on the pairwise interactions. Default is FALSE.

`savage_dickey` logical, whether to compute the Savage-Dickey density ratio Bayes factor for each pairwise interaction via Bayesian Sampling Importance Resampling (BSIR) from the coordinate-rescaled pseudo-posterior: the pseudo-posterior rescaled around its mode to the Godambe-Huber-White (GHW) covariance, which corrects the underestimated posterior variability of the pseudo-likelihood. The proposal is a multivariate t with the GHW covariance. The posterior density of an interaction at zero is a Gaussian kernel estimate from the resampled draws, or, when few draws lie near zero, the density of a normal distribution with the mean and standard deviation of the draws, as recorded in `savage_dickey$density_at_zero`. Only available when `with_prior = TRUE`. Default is FALSE.

`M` number of draws resampled in the BSIR step (default is 1000). Only used when `savage_dickey = TRUE`.

`oversampling` multiplier for the number of proposal draws in the BSIR step (default is 10). The total number of proposal draws is `M * oversampling`. Only used when `savage_dickey = TRUE`.

`ncores` number of cores used to compute the gradient, Hessian and pseudo-likelihood in parallel. It is capped at the number of available cores minus one, and has no effect when the package was built without OpenMP support. Default is 1.

`thresholds_alpha` alpha parameter for the Beta-Prime prior on thresholds (default is 0.5). Only used when `with_prior = TRUE`.

`thresholds_beta` beta parameter for the Beta-Prime prior on thresholds (default is 0.5). Only used when `with_prior = TRUE`.

`interactions_location` location parameter for the Cauchy prior on pairwise interactions (default is 0.0). Only used when `with_prior = TRUE`.

`interactions_scale` scale parameter for the Cauchy prior on pairwise interactions (default is 2.5). Only used when `with_prior = TRUE`.

`proposal_df` degrees of freedom for the multivariate t proposal distribution in the BSIR step (default is 5). Only used when `savage_dickey = TRUE`.

`lrt_intervals` whether to compute likelihood-ratio intervals (default is FALSE; see Details): `TRUE` for all free parameters, or a vector of parameter names (as in `names(fit$argument)`) for some of them. Each interval needs a series of constrained optimizations, which take a while for large networks. They are returned by `confint(fit, method = "lrt")`. When they are not computed and there are fewer than 10 observations per free parameter, a warning suggests them.

`level` confidence level of the intervals (default is 0.95): the Wald intervals printed by `summary()` and the likelihood-ratio intervals, so that both are at the same level. It is also the default level of `confint()`.

`seed` random seed for reproducibility of the BSIR step (default is 123). The caller's random number stream is restored on exit. Only used when `savage_dickey = TRUE`.

Details

The likelihood-ratio interval of a parameter θ_k is based on its profile pseudo-likelihood: for each value of θ_k, the other parameters are set to the values that maximize the pseudo-likelihood (or the pseudo-posterior with `with_prior = TRUE`). The interval collects the values where

$$C_k \cdot 2[\log PL(\hat{\theta}) - \max_{\theta_{-k}} \log PL(\theta_k, \theta_{-k})] \leq \chi^2_{1,level}.$$

For a full likelihood the statistic would be χ^2_1-distributed ($C_k = 1$); for the pseudo-likelihood it is $\lambda_k \chi^2_1$-distributed, with $\lambda_k = \Sigma_{kk}/(H^{-1})_{kk}$, because the curvature H of the pseudo-likelihood overstates the information in the data (Σ is the GHW covariance). The scaling $C_k = 1/\lambda_k$ calibrates

the statistic (Geys, Molenberghs and Ryan, 1999; Pace, Salvan and Sartori, 2011; Varin, Reid and Firth, 2011). Unlike the Wald intervals, the likelihood-ratio intervals can be asymmetric around the estimate. The bounds are found by `uniroot` on each side of the estimate. The asymmetry matters when the sample size is small relative to the number of parameters (Arena and Marsman, 2026), so with fewer than 10 observations per free parameter `dmrfit()` warns that the Wald intervals may be inaccurate.

Value

an object of class `dmrfit`, a list including the estimates (`argument`), the value, gradient, Hessian and GHW covariance of the objective at the estimates (`utils`), the matched call (`call`), the data dimensions (`P`, `N`, `n_categories`), and, with `savage_dickey = TRUE`, the Savage-Dickey Bayes factors with the importance-sampling effective sample size (`savage_dickey`).

References

Arena, G. and Marsman, M. (2026). Bayesian inference for discrete Markov random fields through coordinate rescaling. arXiv preprint. doi:10.48550/arXiv.2601.17205

Geys, H., Molenberghs, G., and Ryan, L. M. (1999). Pseudolikelihood modeling of multivariate outcomes in developmental toxicology. *Journal of the American Statistical Association*, 94(447), 734-745. doi:10.1080/01621459.1999.10474176

Pace, L., Salvan, A., and Sartori, N. (2011). Adjusting composite likelihood ratio statistics. *Statistica Sinica*, 21(1), 129-148.

Skare, Ø., Bølviken, E., and Holden, L. (2003). Improved sampling-importance resampling and reduced bias importance sampling. *Scandinavian Journal of Statistics*, 30(4), 719-737.

Varin, C., Reid, N., and Firth, D. (2011). An overview of composite likelihood methods. *Statistica Sinica*, 21(1), 5-42.

See Also

`dmrfit_bayes` for posterior sampling.

Examples

```
# binary responses of 1000 observations on 3 nodes (independent, for illustration)
set.seed(123)
n <- 1000
P <- 3
prob <- c(0.2, 0.5, 0.3)[rep(1:P, each = n)]
data <- matrix(rbinom(n * P, size = 1, prob = prob), nrow = n, ncol = P)
fit <- dmrfit(data, with_prior = TRUE, savage_dickey = TRUE)
fit
summary(fit)

# constrained network: no edge between nodes 1 and 3
structure <- matrix(c(0, 1, 0, 1, 0, 1, 0, 1, 0), nrow = P)
fit_constrained <- dmrfit(data, structure = structure)
summary(fit_constrained)
```

`dmrfit_bayes` *Posterior sampling for discrete Markov random fields*

Description

Bayesian inference for a discrete Markov random field (Ising or ordinal) by Markov chain Monte Carlo. By default, the posterior is approximated by the coordinate-rescaled (CoRe) pseudo-posterior of Arena and Marsman (2026): the pseudo-posterior rescaled around its mode so that its covariance matches the Godambe-Huber-White (GHW) covariance, which corrects the underestimated posterior variability of the pseudo-likelihood. All samplers use the Fisher adaptive Metropolis-adjusted Langevin algorithm (FisherMALA; Titsias, 2024).

Usage

```
dmrfit_bayes(
  data,
  parinit = NULL,
  method = c("core", "adacore", "exact", "dmh"),
  scale = c("ghw", "mch", "rm"),
  nsim = 1000,
  burnin = 1000,
  ncores = 1,
  thresholds_alpha = 0.5,
  thresholds_beta = 0.5,
  interactions_location = 0,
  interactions_scale = 2.5,
  sigma2 = 1,
  seed = 123,
  progress = TRUE,
  control = list()
)
```

Arguments

`data` data matrix, with rows as samples and columns as variables. Each variable should be rescaled to the range of 0 to m-1, where m is the number of categories for that variable. The baseline category is always the minimum value in the variable. The internal processing will check if the variables are rescaled and will rescale them if necessary. If there are any NAs in the data, they will be removed before optimization (listwise deletion).

`parinit` initial parameter values for the optimization that finds the pseudo-posterior mode, a vector of length equal to the number of parameters in the model, `sum(n_categories - 1) + P * (P - 1) / 2`, where `P` is the number of variables. If NULL (default), a vector of zeros.

`method` the sampler: `"core"` (default) samples the CoRe pseudo-posterior with a fixed rescaling matrix; `"adacore"` adapts the rescaling matrix during burn-in at the

running mean of the draws; "exact" samples the posterior based on the full likelihood, whose normalizing constant is computed by enumerating all response patterns (only for small networks, see control); "dmh" samples the posterior based on the full likelihood with the double Metropolis-Hastings algorithm (Liang, 2010). DMH simulates auxiliary data at every iteration and is much slower than the other methods (minutes to hours instead of seconds, depending on the network size and control$dmh_aux); "core" runs at the cost of the pseudo-posterior sampler.

scale
: the covariance to which the pseudo-posterior is rescaled, only used when method = "core": "ghw" (default) the Godambe-Huber-White (GHW) covariance; "mch" the inverse of the negative Hessian of the full log-posterior at the pseudo-posterior mode, estimated by Monte Carlo simulation; "rm" the same Hessian at the full-posterior mode, estimated by a Newton-type Robbins-Monro algorithm (the sampler remains centered at the pseudo-posterior mode). "mch" and "rm" simulate data from the model and are slower.

nsim
: number of posterior draws kept after burn-in. Default is 1000.

burnin
: number of burn-in iterations, after an initial adaptive stage of control$adaptive_stage iterations. Default is 1000.

ncores
: number of cores used to compute the gradient, Hessian and pseudo-likelihood in parallel when finding the pseudo-posterior mode. It is capped at the number of available cores minus one, and has no effect when the package was built without OpenMP support. Default is 1.

thresholds_alpha
: alpha parameter for the Beta-Prime prior on thresholds (default is 0.5).

thresholds_beta
: beta parameter for the Beta-Prime prior on thresholds (default is 0.5).

interactions_location
: location parameter for the Cauchy prior on pairwise interactions (default is 0.0).

interactions_scale
: scale parameter for the Cauchy prior on pairwise interactions (default is 2.5).

sigma2
: initial value for the adaptive variance parameter in the FisherMALA sampler (default is 1.0).

seed
: random seed for reproducibility of the MCMC sampler (default is 123). The caller's random number stream is restored on exit.

progress
: logical, whether to show a progress bar while sampling (default is TRUE).

control
: a list of tuning settings, each with a default: adaptive_stage (500) iterations of the initial adaptive stage; max_states (1e6) largest number of response patterns for method = "exact"; dmh_aux (25000) and dmh_gibbs_iter (5) auxiliary draws and Gibbs sweeps per iteration of method = "dmh"; mc_draws (1e5) and mc_gibbs_iter (5) simulated observations and Gibbs sweeps for scale = "mch" and "rm"; rm_iter (50) Robbins-Monro iterations for scale = "rm".

Value

an object of class dmrfit_bayes (also of class dmrfit), including the pseudo-posterior mode (argument), the posterior draws (draws, a matrix with parameters by rows and iterations by columns),

the acceptance rate (`acceptance`), the sampling method and scale (`method`, `scale`), the Savage-Dickey Bayes factors with the effective sample size of each parameter (`savage_dickey`; the posterior density of an interaction at zero is a Gaussian kernel estimate, or, when few draws lie near zero, the density of a normal distribution with the mean and standard deviation of the draws, as recorded in `savage_dickey$density_at_zero`), and the multivariate effective sample size of the draws (`mess`; Vats, Flegal and Jones, 2019), which is `NA` when there are fewer than `P + 1` batches of `floor(sqrt(nsim))` draws per parameter.

References

Arena, G. and Marsman, M. (2026). Bayesian inference for discrete Markov random fields through coordinate rescaling. arXiv preprint. doi:10.48550/arXiv.2601.17205

Liang, F. (2010). A double Metropolis-Hastings sampler for spatial models with intractable normalizing constants. *Journal of Statistical Computation and Simulation*, 80(9), 1007-1022.

Titsias, M. K. (2024). Optimal preconditioning and Fisher adaptive Langevin sampling. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (NIPS '23). Red Hook, NY, USA: Curran Associates.

Vats, D., Flegal, J. M., and Jones, G. L. (2019). Multivariate output analysis for Markov chain Monte Carlo. *Biometrika*, 106(2), 321-337.

See Also

`dmrfit` for point estimation and Bayes factors by sampling importance resampling.

Examples

```
# binary responses of 1000 observations on 3 nodes (independent, for illustration)
set.seed(123)
n <- 1000
P <- 3
prob <- c(0.2, 0.5, 0.3)[rep(1:P, each = n)]
data <- matrix(rbinom(n * P, size = 1, prob = prob), nrow = n, ncol = P)
fit <- dmrfit_bayes(data = data, nsim = 500, burnin = 500, progress = FALSE)
fit
summary(fit)

# small network: the exact posterior is also available
fit_exact <- dmrfit_bayes(data = data, method = "exact", nsim = 500, burnin = 500, progress = FALSE)
summary(fit_exact)
```

`plot.dmrfit` *Plot a fitted discrete Markov random field*

Description

Plots of a `dmrfit` or `dmrfit_bayes` fit, made with **ggplot2**. The interactions are the estimates of `dmrfit()`, or a posterior summary of the draws of `dmrfit_bayes()` (see `estimate`).

`type = "network"` The nodes are the variables and the edges the pairwise interactions: the edge width is the absolute value of the interaction, the color and line type its sign. When the fit has Savage-Dickey Bayes factors, only the included edges are drawn (see Details).

`type = "bf"` One circle per pair of variables (lower triangle): its color is the evidence for the edge given by its Savage-Dickey Bayes factor (see Details) and its area the absolute value of the interaction. Requires a fit with Bayes factors.

`type = "trace"` The posterior draws (after burn-in) of the parameters in `pars` against the iteration, one panel per parameter. Only for a `dmrfit_bayes` fit.

`type = "density"` The kernel density of the posterior draws of the parameters in `pars`, one panel per parameter, with a solid line at the posterior summary chosen by `estimate` and dashed lines at the highest posterior density interval of mass `prob`. Only for a `dmrfit_bayes` fit.

`type = "intervals"` The estimate and the interval of every parameter in `pars`, the thresholds and the interactions in separate panels, each sorted by estimate. The intervals are the likelihood-ratio intervals of a `dmrfit` fit computed with `lrt_intervals = TRUE`, otherwise its Wald intervals, both at the level of the fit, or the highest posterior density intervals of mass `prob` of a `dmrfit_bayes` fit (see `confint.dmrfit`). The caption states which.

`type = "centrality"` The expected influence of every variable, $EI_i = \sum_j \theta_{ij}$, the sum of its interactions, sorted by its estimate. For a `dmrfit_bayes` fit, the posterior summary chosen by `estimate` and the highest posterior density interval of mass `prob` of the expected influence computed on every draw, and next to it the posterior probability that the variable is the most central (the share of draws in which its expected influence is the largest). For a `dmrfit` fit, the estimate and its Wald interval at the level of the fit; the standard error of a sum of interactions follows from their Godambe-Huber-White (GHW) covariance.

Usage

```
## S3 method for class 'dmrfit'
plot(
  x,
  type = c("network", "bf", "trace", "density", "intervals", "centrality"),
  estimate = c("mode", "mean", "median"),
  pars = NULL,
  prob = 0.95,
  all_edges = FALSE,
  groups = NULL,
  layout = c("fr", "circle"),
  seed = 30,
  node_size = 10,
  show_comments = TRUE,
  ...
)
```

Arguments

`x`	a `dmrfit` or `dmrfit_bayes` object.
`type`	the plot: `"network"` (default), `"bf"`, `"trace"`, `"density"`, `"intervals"` or `"centrality"`.
`estimate`	the posterior summary of the interactions of a `dmrfit_bayes` fit: `"mode"` (default) the marginal posterior mode, the maximum of the kernel density estimate of the draws (`stats::density`), `"mean"` or `"median"`. The mode is the default because the posterior can be asymmetric in small samples, which moves the mean and the median away from the mode; Arena and Marsman (2026) found this for the thresholds of rarely chosen response categories. Ignored for a `dmrfit` fit, which has point estimates.
`pars`	(trace, density, intervals) the parameters to show (at most 9 for trace and density): their names, as in `names(x$argument)` (for instance `"theta[2,1]"` for the interaction between variables 2 and 1, `"mu[1,2]"` for the second threshold of variable 1), or their positions in that vector. By default, the four interactions with the largest absolute posterior summary (see `estimate`) for trace and density, and all the free parameters for intervals. The parameters are labeled with the variable names.
`prob`	(density, intervals, centrality) mass of the highest posterior density interval of a `dmrfit_bayes` fit (default 0.95). The intervals of a `dmrfit` fit are at its `level`.
`all_edges`	(network) logical, whether to draw every estimated interaction instead of only the included ones (default FALSE). Every estimated interaction is also drawn for a fit without Bayes factors (only the free ones for a constrained fit).
`groups`	(network) optional grouping of the nodes (for instance `attr(rads2, "clusters")`), a vector with one entry per variable, in the column order of the data or named by variable. At most four groups, shown by the color and shape of the nodes.
`layout`	(network) the node positions: `"fr"` (default) the Fruchterman-Reingold layout of **igraph**, which must be installed; `"circle"` the nodes on a circle, in the column order of the data; or a matrix of coordinates with two columns and one row per variable (in the column order of the data, or with the variable names as row names), rescaled by the same factor on both axes. The `"fr"` layout is computed from all estimated interactions, weighted by their absolute value, so that the nodes keep their positions when `all_edges` changes.
`seed`	(network) random seed of the `"fr"` layout (default 30); other seeds give other arrangements. The caller's random number stream is restored on exit.
`node_size`	(network) size of the nodes (default 10); the labels scale with it.
`show_comments`	logical, whether to show the explanation below the plot (default TRUE): the rule for the drawn edges, or what the points, lines and probabilities show. With FALSE, only the plot and its legends are drawn.
`...`	further arguments (currently unused).

Details

The evidence for an edge is classified by its Savage-Dickey Bayes factor BF_{01}, the evidence for the absence of the edge: included ($BF_{01} < 1/10$), weak included ($1/10 \leq BF_{01} < 1/3$), inconclusive

($1/3 \leq BF_{01} < 3$), weak excluded ($3 \leq BF_{01} < 10$) and excluded ($BF_{01} \geq 10$), shown from orange (included) through gray to violet (excluded). The network plot draws the included edges.

The edge widths of the network plot are scaled to the range of the absolute interactions drawn in each plot. To compare widths between plots, fix the range, for instance with `+ ggplot2::scale_linewidth_continuous(limits = c(0, 1), range = c(0.3, 2.4))`.

Value

a `ggplot` object, which can be modified further with **ggplot2**.

References

Arena, G. and Marsman, M. (2026). Bayesian inference for discrete Markov random fields through coordinate rescaling. arXiv preprint. doi:10.48550/arXiv.2601.17205

See Also

`dmrfit`, `dmrfit_bayes`

Examples

```
# four items of the RADS-2 data (a small network keeps the examples fast)
data(rads2)
items <- c("D3", "D6", "D7", "D8")
fit <- dmrfit(rads2[, items], with_prior = TRUE, savage_dickey = TRUE)
# network of the included edges (the default layout uses the igraph package)
if (requireNamespace("igraph", quietly = TRUE)) plot(fit)

# all estimated interactions, with the nodes grouped and placed on a circle
groups <- c("first", "first", "second", "second")
plot(fit, all_edges = TRUE, groups = groups, layout = "circle")

# evidence for every edge from its Savage-Dickey Bayes factor
plot(fit, type = "bf")

# trace and density of the posterior draws
fit_bayes <- dmrfit_bayes(rads2[, items], nsim = 500, burnin = 300, progress = FALSE)
plot(fit_bayes, type = "trace")
plot(fit_bayes, type = "density", pars = c("theta[2,1]", "mu[1,1]"))

# estimates and intervals: Wald, likelihood-ratio and highest posterior density
plot(fit, type = "intervals")
fit_lrt <- dmrfit(rads2[, items], lrt_intervals = c("mu[1,3]", "theta[2,1]"))
plot(fit_lrt, type = "intervals", pars = c("mu[1,3]", "theta[2,1]"))
plot(fit_bayes, type = "intervals")

# expected influence of every variable, without the explanation below the plot
plot(fit, type = "centrality", show_comments = FALSE)
plot(fit_bayes, type = "centrality")
```

print.dmrfit *Print a* dmrfit *object*

Description

Prints a brief overview of a fitted discrete MRF model, analogous to `print.lm`.

Usage

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

Arguments

`x`	a `dmrfit` object.
`...`	further arguments passed to `print`.

Value

the `dmrfit` object `x`, invisibly.

print.dmrfit_bayes *Print a* dmrfit_bayes *object*

Description

Prints a brief overview of a discrete MRF model fitted via Bayesian MCMC sampling.

Usage

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

Arguments

`x`	a `dmrfit_bayes` object.
`...`	further arguments passed to `print`.

Value

the `dmrfit_bayes` object `x`, invisibly.

print.summary.dmrfit *Print a* summary.dmrfit *object*

Description

Prints the detailed summary of a fitted discrete MRF model, analogous to print.summary.lm.

Usage

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

Arguments

x	a summary.dmrfit object.
...	further arguments passed to printCoefmat.

Value

the summary.dmrfit object x, invisibly.

print.summary.dmrfit_bayes *Print a* summary.dmrfit_bayes *object*

Description

Prints the detailed summary of a discrete MRF model fitted via Bayesian MCMC sampling.

Usage

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

Arguments

x	a summary.dmrfit_bayes object.
...	further arguments passed to print.

Value

the summary.dmrfit_bayes object x, invisibly.

`rads2` *Reynolds Adolescent Depression Scale (RADS-2) responses of Peruvian adolescents*

Description

Responses of 917 adolescents from Lima (Peru) to the 25 items retained in the four-factor version of the Reynolds Adolescent Depression Scale, second edition (RADS-2), each answered on a four-point Likert scale. The data come from Ramos-Vera et al. (2023), who also grouped the items into four clusters (stored with the data).

Usage

```
rads2
```

Format

A data frame with 917 rows (adolescents) and 25 integer columns (items), with values 1 to 4. Higher values indicate more frequent depressive symptoms for every item. The columns keep the item numbering of the RADS-2 (`D1`, `D3`, ..., `D30`). The attribute `"clusters"` is a named character vector giving the cluster of each item (see Details).

Details

The items are grouped into four clusters (Ramos-Vera et al., 2023), stored in `attr(rads2, "clusters")`:

- Dysphoria: D3, D6, D7, D8, D16, D21, D26
- Anhedonia/Negative Affect: D1, D5, D12
- Negative Self-Assessment: D4, D9, D13, D14, D15, D19, D20, D30
- Somatic Complaint: D11, D17, D18, D22, D24, D27, D28

Subsetting the data frame (e.g. `rads2[, 1:10]`) drops the attribute; retrieve it first with `attr(rads2, "clusters")`.

The data are derived from "S1 Data" of Ramos-Vera et al. (2023), with the following changes: the 25 items of the four-factor version were kept (items 2, 10, 23, 25 and 29, and the age and sex of the respondents, were removed), the items were renamed from `A<k>` to `D<k>`, and the positively worded items `D1`, `D5` and `D12` were reverse-coded (`5 - x`), so that higher values indicate more symptoms for every item. The rows are in the original order. The item wording of the RADS-2 is not included.

Source

Ramos-Vera, C., Quispe Callo, G., Basauri Delgado, M., Vallejos Saldarriaga, J., and Saintila, J. (2023). "S1 Data. Reynolds Adolescent Depression Scale", supporting information of the article below, doi:10.1371/journal.pone.0286081.s001. Licensed under the Creative Commons Attribution 4.0 International license (CC BY 4.0, https://creativecommons.org/licenses/by/4.0/); the changes are listed under Details. The rest of the package is licensed under the MIT license.

References

Ramos-Vera, C., Quispe Callo, G., Basauri Delgado, M., Vallejos Saldarriaga, J., and Saintila, J. (2023). Factorial and network structure of the Reynolds Adolescent Depression Scale (RADS-2) in Peruvian adolescents. *PLOS ONE*, 18(5), e0286081. doi:10.1371/journal.pone.0286081

Reynolds, W. M. (2002). *Reynolds Adolescent Depression Scale, 2nd Edition: Professional Manual*. Lutz, FL: Psychological Assessment Resources.

Examples

```
data(rads2)
dim(rads2)
table(attr(rads2, "clusters"))

# point estimates and Godambe-Huber-White standard errors for the seven dysphoria items
dysphoria <- names(which(attr(rads2, "clusters") == "Dysphoria"))
fit <- dmrfit(rads2[, dysphoria], with_prior = TRUE)
summary(fit)
```

---

summary.dmrfit *Summary of a* dmrfit *object*

Description

Produces a detailed summary of a fitted discrete MRF model, analogous to `summary.lm`. Standard errors are obtained from the Godambe-Huber-White (GHW) estimator.

Usage

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

Arguments

`object`	a `dmrfit` object.
`...`	further arguments (currently unused).

Value

an object of class `summary.dmrfit` containing:

`call`	the matched call.
`coefficients`	a matrix with columns for the estimate, standard error, z-value and p-value.
`thresholds`	coefficient matrix for threshold parameters.
`interactions`	coefficient matrix for free pairwise interaction parameters.
`neg_pseudo_loglik`	the negative log pseudo-likelihood at convergence.

`P`	number of nodes.
`N`	number of observations.
`n_categories`	vector of category counts per node.
`with_prior`	logical, whether a prior was used.
`structured`	logical, whether the network structure was constrained.
`intervals`	a matrix with the Wald intervals of the free parameters, and their likelihood-ratio intervals when the fit has them (`lrt_intervals = TRUE`), at the level of the fit.
`level`	the level of the intervals.

`summary.dmrfit_bayes` *Summary of a* `dmrfit_bayes` *object*

Description

Produces a detailed summary of a discrete MRF model fitted via Bayesian MCMC sampling.

Usage

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

Arguments

`object`	a `dmrfit_bayes` object.
`...`	further arguments (currently unused).

Value

an object of class `summary.dmrfit_bayes`.

Index

∗ **datasets**
    rads2, 15

confint.dmrfit, 2, *10*

dmrfit, *2*, 3, *3*, *9*, *12*
dmrfit_bayes, *3*, *6*, 7, *12*

plot.dmrfit, 9
print.dmrfit, 13
print.dmrfit_bayes, 13
print.summary.dmrfit, 14
print.summary.dmrfit_bayes, 14

rads2, 15

summary.dmrfit, 16
summary.dmrfit_bayes, 17