Package ‘boral’

September 25, 2015

Title Bayesian Ordination and Regression AnaLysis

Version 0.9.1

Date 2015-10-01

Author Francis K.C. Hui

Maintainer Francis Hui <fhui28@gmail.com>

Description Bayesian approaches for analyzing multivariate data in ecology. Estimation is performed using Markov Chain Monte Carlo (MCMC) methods via JAGS. Three types of models may be fitted: 1) With explanatory variables only, boral fits independent column GLMs to each column of the response matrix; 2) With latent variables only, boral fits a purely latent variable model for model-based unconstrained ordination; 3) With explanatory and latent variables, boral fits correlated column GLMs with latent variables to account for any residual correlation between the columns of the response matrix.

License GPL-2

Depends coda

Imports R2jags, mvtnorm, fishMod, MASS, stats, graphics, grDevices

Suggests mvabund (>= 3.8.4), corrplot, testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2015-09-25 13:32:11

R topics documented:

`boral-package` *Bayesian Ordination and Regression AnaLysis (boral)*

Description

boral is a package offering Bayesian model-based approaches for analyzing multivariate data in ecology. Estimation is performed using Bayesian/Markov Chain Monte Carlo (MCMC) methods via JAGS (Plummer, 2003). Three “types" of models may be fitted: 1) With covariates and no latent variables, boral fits independent response GLMs such that the columns of `y` are assumed to be independent; 2) With no covariates, boral fits a pure latent variable model (Skrondal and Rabe-Hesketh, 2004) to perform model-based unconstrained ordination (Hui et al., 2014); 3) With covariates and latent variables, boral fits correlated response GLMs, with latent variables accounting for any residual correlation between the columns of `y`.

Details

Package:	boral
Type:	Package
Version:	0.6
Date:	2014-12-12
License:	GPL-2

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

References

- Hui et al., (2014). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6, 399-411.
- Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd International Workshop on Distributed Statistical

Computing. March (pp. 20-22).

- Skrondal, A., and Rabe-Hesketh, S. (2004). Generalized latent variable modeling: Multilevel, longitudinal, and structural equation models. CRC Press.
- Yi W. et al. (2013). mvabund: statistical methods for analysing multivariate abundance data. R package version 3.8.4.

Examples

```
## Please see examples in the help file for boral (?boral). Thanks!
```

boral *Fitting boral (Bayesian Ordination and Regression AnaLysis) models*

Description

Bayesian ordination and regression models for analyzing multivariate data in ecology. Three "types" of models may be fitted: 1) With covariates and no latent variables, boral fits independent response GLMs; 2) With no covariates, boral fits a pure latent variable model; 3) With covariates and latent variables, boral fits correlated response GLMs.

Usage

```
boral(y, ...)

## Default S3 method:
boral(y, X = NULL, traits = NULL, which.traits = NULL, family,
trial.size = 1, num.lv = 0, row.eff = "none", n.burnin = 10000,
n.iteration = 40000, n.thin = 30, save.model = FALSE, seed = 123,
calc.ics = TRUE, hypparams = c(100,20,100,50), ssvs.index = -1,
do.fit = TRUE, model.name = NULL, ...)

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

Arguments

`y` A response matrix of multivariate data e.g., counts, binomial or Bernoulli responses, continuous response, and so on. With multivariate abundance data ecology for instance, rows correspond to sites and columns correspond to species. Any categorical (multinomial) responses **must** be converted to integer values. For ordinal data, the minimum level of `y` must be 1 instead of 0.

`X` A model matrix of covariates, which can be included as part of the boral model. Defaults to `NULL`, in which case no model matrix was used. No intercept column should be included in `X`.

`x` An object for class "boral".

`traits` A model matrix of species covariates, which can be included as part of the boral model. Defaults to `NULL`, in which case no matrix was used. An intercept column should be included in `traits` if appropriate (usually is).

`which.traits` A list of length equal to (number of columns in `X` + 1), informing which columns of `traits` the column-specific intercepts and each of the column-specific regression coefficients should be regressed against. The first element in the list applies to the column-specific intercept, while the remaining elements apply to the regression coefficients. Each element of `which.traits` is a vector indicating which traits are to be used. For example, if `which.traits[[2]] = c(2,3)`, then the regression coefficients corresponding to the first column in `X` are regressed against the second and third columns of `traits`. If `which.traits[[2]] = 0`, then the regression coefficients are treated as independent. Please see help file below for more details.

Defaults to `NULL`, in conjunction with `traits = NULL`).

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`num.lv` Number of latent variables to fit. Can take any non-negative integer value. Defaults to 0.

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and unknown variance, analogous to a random intercept in mixed models. Defaults to "none".

`n.burnin` Length of burnin i.e., the number of iterations to discard at the beginning of the MCMC sampler.

`n.iteration` Number of iterations including burnin.

`n.thin` Thinning rate. Must be a positive integer. With the default values of `n.burnin` , `n.iteration` and `n.thin`, this leads to a final of 1000 MCMC samples.

`save.model` A logical value indicating whether to save the JAGS model file as a text file (with name based on `model.name`) in the current working directory, as well as the MCMC samples from the call to JAGS. If saved, various functions available in the coda package can be applied to the MCMC samples. Note MCMC samples can take up a lot of memory. Defaults to FALSE.

`seed` Seed for JAGS sampler. A `set.seed(seed)` command is run immediately before starting the MCMC sampler. Defaults to the value 123.

`calc.ics` A logical values indicating whether to return various information criteria values, which could be used to perform model selection (see `get.measures`). Defaults to TRUE.

`hypparams` Vector of four hyperparameters used in the set up of prior distributions. The first element is the variance for the normal priors of all column-specific intercepts, row effects, and cutoff points for ordinal data. It also controls the maximum of the uniform prior for the standard deviation of the random effects normal distribution, if `row.eff = "random"`. The second element is the variance for the normal priors of all latent variable coefficients (ignored if `num.lv = 0`). The third element is the variance for the normal priors of all column-specific coefficients relating to the model matrix `X` (ignored if `X = NULL`). When traits are included in the model, it also controls the maximum of the uniform prior for the standard deviation of the normally distributed random effects (please see section on *Including species traits* below). The fourth element controls the maximum of the uniform prior used for dispersion parameters, ϕ. Note the common power parameter in the tweedie distribution is assumed to have uniform prior from 1 to 2.

`ssvs.index` Indices to be used for Stochastic Search Variable Selection (SSVS, George and McCulloch, 1993). Either a single element or a vector with length equal to the number of columns in `X`. Each element can take values of -1 (no SSVS is performed on this covariate), 0 (SSVS is performed on individual coefficients for this covariate), or any integer exceeding 1 (SSVS is performed on collectively all coefficients on this covariate/s.) Defaults to -1, in which case no model selection is performed on the fitted model at all. Please see the `details` for more information regarding the implementation of SSVS.

`do.fit` A logical value indicating whether to actually fit the boral model. If set to FALSE, then only the JAGS model file is written to the current working directly (as text file with name based on `model.name`), no MCMC sampling is performed, and *nothing else* is returned. Defaults to TRUE.

`model.name` Name of the text file that the JAGS model is written to. Defaults to NULL, in which case the default of "jagsboralmodel.txt" is used.

`...` Not used.

Details

The boral package is designed to fit three types models which may be useful in ecology (and probably outside of ecology as well =D).

Independent response models: boral allows explanatory variables to be entered into the model via a model matrix `X`. This model matrix can contain anything the user wants, provided factors have been parameterized as dummy variables. It should NOT include an intercept column.

Without latent variables, i.e. `num.lv = 0`, boral fits separate GLMs to each column of the $n \times p$ matrix `y`, where the columns are assumed to be independent.

$$g(\mu_{ij}) = \alpha_i + \beta_{0j} + \boldsymbol{x}_i^T \boldsymbol{\beta}_j; \quad i = 1, \ldots, n; j = 1, \ldots, p,$$

where the mean response for element (i,j), denoted as mu_{ij}, is regressed against the covariates $\boldsymbol{x}_i$ via a link function $g(\cdot)$. The quantities $beta_{0j}$ and $\boldsymbol{beta}_j$ denote the column-specific intercepts and coefficients respectively, while `alpha_i` is an optional row effect that may be treated as a fixed or random effect. The latter assumes the row effects are drawn from a normal distribution with unknown variance ϕ^2.

Fitting the above type of model is sort of like a Bayesian analog of the `manyglm` function in the `mvabund` package (Wang et al., 2013). Unlike `manyglm` though, row effects can be added easily as a type of "row-standardization". Also, a wider range of assumed distributions (families) are possible, as discussed below.

A not-so-brief tangent on distributions: In the event different responses are collect for different columns, e.g., some columns of `y` are counts, while other columns are presence-absence, one can specify different distributions for each column. Aspects such as variable selection, residual analysis, and plotting of the latent variables are, in principle, not affected by having different distributions. Naturally though, one has to be more careful with interpretation of the row effects α_i and latent variables $\boldsymbol{z}_i$, as different link functions will be applied to each column of `y`. A situation where different distributions may prove useful is when `y` is a species-traits matrix, where each row is a species and each column a trait such as specific leaf area. In this case, traits could be of different response types, and the goal perhaps is to perform unconstrained ordination to look for patterns between species on an underlying trait surface e.g., a defense index for a species (Moles et al., 2013; see also the discussion below on how to perform model-based unconstrained ordination).

For multivariate abundance data in ecology (also known as community composition data, Legendre and Gallagher, 2001), species counts are often overdispersed. Using a negative binomial distribution (`family = "negative.binomial"`) to model the counts usually helps to account for this overdispersion. Please note the variance for the negative binomial distribution is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the dispersion parameter.

For non-negative continuous data such as biomass, the lognormal and tweedie distributions may be used (Foster and Bravington, 2013). Note however that a common power parameter is used for tweedie columns – there is almost always insufficient information to model column-specific power parameters. Normal responses are also implemented, just in case you encounter normal stuff in ecology (pun intended)!

The beta distribution can be used to model data between values between but *not* including 0 and 1. In principle, this would make it useful for percent cover data in ecology, if it not were for the fact that percent cover is commonly characterized by having lots of zeros (which are not permitted for beta regression). An *ad-hoc* fix to this would be to add a very small value to shift the data away from exact zeros and/or ones. This is however heuristic, and pulls the model towards producing conservative results (see Smithson and Verkuilen, 2006, for a detailed discussion on beta regression, and Korhonen et al., 2007, for an example of an application to forest canopy cover data). Note the parameterization of the beta distribution used here is directly in terms of the mean μ and the dispersion parameter ϕ (more commonly know as the "sample size"). In terms of the two shape parameters, this is equivalent to $shape1 = a = \mu\phi$ and $shape2 = b = (1-\mu)\phi$.

For ordinal response columns, cumulative probit regression is used (Agresti, 2010). boral assumes all ordinal columns are measured using the same scale i.e., all columns have the same number of theoretical levels. The number of levels is then assumed to be given by the maximum value from all the ordinal columns of `y`. Because of this, all ordinal columns then assumed to have the *same* cutoffs, $\boldsymbol{\tau}$, while the column-specific intercept effect, β_{0j}, allows for deviations away from these common cutoffs. That is,

$$probit(P(y_{ij} \leq k)) = \tau_k + \beta_{0j} + \ldots,$$

where $probit(\cdot)$ is the probit function, $P(y_{ij} \leq k)$ is the cumulative probability of element y_{ij} being less than or equal to level k, τ_k is the cutoff linking levels k and $k+1$ (and which are increasing in k), β_{0j} are the column effects, and $\ldots$ denotes what else is included in the model, e.g. latent variables and related coefficients. A sum-to-zero constraint is imposed on the β_{0j}'s of all ordinal columns, to ensure model identifiability.

The parameterization above is useful for modeling ordinal in ecology. When ordinal responses are recorded, usually the same scale is applied to all species e.g., level 1 = not there, level 2 = a bit there, level 3 = lots there, level 4 = everywhere! The quantity τ_k can thus be interpreted as this common scale, while β_{0j} allows for deviations away from these to account for differences in species prevalence. Admittedly, the current implementation of boral for ordinal data can be quite slow.

Pure latent variable models: If no explantory variables are included and `num.lv` > 0, boral fits a pure latent variable model to perform model-based unconstrained ordination (Hui et al., 2014),

$$g(\mu_{ij}) = \alpha_i + \beta_{0j} + \boldsymbol{z}_i^T \boldsymbol{\theta}_j,$$

where instead of measured covariates, we now have a vector of latent variables $\boldsymbol{z}_i$ with $\boldsymbol{\theta}_j$ being the column-specific coefficients relating to these latent variables. The column-specific intercept, beta_0j, accounts for differences between species prevalence, while the row effect, $alpha_i$, is included to account for differences in site total abundance (typically assuming a fixed effect, `row.eff = "fixed"`, although see Jamil and ter Braak, 2013, for a motivation for using random site effects), so that the ordination is then in terms of species composition. If α_i is omitted from the model i.e., `row.eff = FALSE`, then the ordination will be in terms of relative species abundance.

Unconstrained ordination is used for visualizing multivariate data in a low-dimensional space, without reference to covariates (Chapter 9, Legendre and Legendre, 2012). Typically, `num.lv` = 1 to 3 latent variables is used, allowing the latent variables to plotted (using `lvsplot`, for instance). The resulting plot can be interpreted in the same manner as plots from Nonmetric Multi-dimensional Scaling (NMDS, Kruskal, 1964) and Correspondence Analysis (CA, Hill, 1974), for example. A biplot can also be constructed by setting `biplot = TRUE` when using `lvsplot`, so that both the latent variables and their corresponding coefficients are plotted. For instance, with multivariate abundance data, biplots are used to visualize the relationships between sites in terms of species abundance or composition, as well as the indicator species for the sites.

Correlated response models: When one or more latent variables are included in conjunction with covariates i.e., `X` is given and `num.lv > 1`, boral fits separate GLMs to each column of `y` while allowing for residual correlation between columns via the latent variables. This is quite useful for multivariate abundance data in ecology, where a separate GLM is fitted to species (modeling its response against environmental covariates), while accounting for the fact species at a site are likely to be correlated for reason other than similiarites in environmental responses, e.g. biotic interaction, phylogeny, and so on. Correlated response model take the following form,

$$g(\mu_{ij}) = \beta_{0j} + \boldsymbol{x}_i^T\boldsymbol{\beta}_j, + \boldsymbol{z}_i^T\boldsymbol{\theta}_j.$$

This model is thus a mash of the first two types of models. The linear predictor $\boldsymbol{z}_i^T\boldsymbol{\theta}_j$ induces a residual covariance between the columns of `y` (which is of rank `num.lv`). For multivariate abundance data, this leads to a parsimonious method of accounting for correlation between species not due to the shared environmental responses. After fitting the model, the residual correlation matrix then can be obtained via the `get.residual.cor` function. Note `num.lv > 1` is necessarily in order to flexibly model the residual correlations; see Pollock et al. (2014) for residual correlation matrices in the context of Joint Species Distribution Models, and Warton et al. (2015) for an overview of latent variable models in multivariate ecology.

Including species traits: When covariates `X` are included (i.e. both the independent and correlated response models), one has the option of also including traits to help explain differences in species environmental responses to these covariates. Specifically, when `traits` and `which.traits` are supplied, then the β_{0j}'s and $\boldsymbol{\beta}_j$'s are then regarded as random effects drawn from a normal distribution. For the species-specific intercepts, we have

$$\beta_{0j} \sim N(\kappa_{01} + \boldsymbol{traits}_j^T\boldsymbol{\kappa}_1, \sigma_1^2),$$

where $(\kappa_{01}, \boldsymbol{\kappa}_1)$ are the regression coefficients relating to the traits to the intercepts and σ_1 is the error standard deviation. These are now the "parameters" in the model, in the sense that priors are assigned to them and MCMC sampling is used to estimate them (see the next section on estimation). Please note that in order of κ_{01} to be included in the model, an intercept column MUST be included in `traits`.

In an analogous manner, each of the elements in $\boldsymbol{\beta}_j = (\beta_{j1}, \ldots, \beta_{jd})$ are now drawn as random effects from a normal distribution. That is, for $k = 1, \ldots, d$ where `d = ncol(X)`, we have,

$$\beta_{jk} \sim N(\kappa_{0k} + \boldsymbol{traits}_j^T\boldsymbol{\kappa}_k, \sigma_k^2),$$

Which traits are to included (regressed) in the mean of the normal distributions is determined by the list `which.traits`. The first element in the list applies to $beta_{0j}$, while the remaining elements apply to the the $\boldsymbol{\beta}_j$. Each element of `which.traits` is a vector indicating which traits are to be used. For example, if `which.traits[[2]] = c(2,3)`, then the β_{j1}'s are drawn from a normal distribution with mean depending only on the second and third columns of `traits`. If `which.traits[[2]] = 0`, then the regression coefficients are treated as independent, i.e. the values of β_{j1} are given their own priors and estimated separately from each other.

Including species traits in the model can be regarded as a method of simplifying the model – rather than each to estimates p sets of species-specific coefficients, we instead say that these coefficients are linearly related to the corresponing values of their traits (Warton et al., 2015). That is, we are using trait data to help explain similarities/differences in species responses to the environment. This idea has close relations to the fourth corner problem in ecology (Brown et al., 2014). Unlike the models of Brown et al. (2014) however, which treat the β_{0j}'s and β_{jk}'s are fixed effects and fully explained by the traits, boral adopts a random effects approach similar to Jamil et al. (2013) to "soak up" any additional between species differences in environmental responses not explained by traits.

Estimation: For boral models, estimation is performed using Bayesian Markov Chain Monte Carlo (MCMC) methods via JAGS (Plummer, 2003). Please note that only *one* MCMC chain in run – this point is discussed further in this help file. Regarding prior distributions, the default settings are as follows:

- Normal priors with mean zero and variance given by `hypparams[1]` are assigned to all intercepts, cutoffs for ordinal responses, and row effects. If the row effects are assumed to random, then the standard deviation of the normal random effect is assigned a uniform prior with maximum `hypparams[1]`,
- Normal priors with mean zero and variance given by `hypparams[2]` are assigned coefficients relating to latent variables, $\boldsymbol{\theta}_j$,
- Normal priors with mean zero and variance given by `hypparams[3]` are assigned to all coefficients relating to covariates in $\boldsymbol{\beta}_j$. If traits are included, the same normal priors are assigned to the κ's, and the standard deviation σ_k are assigned uniform priors with maximum equal to `hypparams[4]`.
- For the negative binomial, normal, lognormal, and tweedie distributions, uniform priors with maximum equal to `hypparams[4]` are used on the dispersion parameters. Please note that for the normal and lognormal distributions, these uniform priors are assigned to the standard deviations ϕ (see Gelman, 2006).

With the default values of `hypparams`, all parameters are given uninformative prior distributions except for the priors of the latent variable coefficients $\boldsymbol{\theta}_j$. We recommend such a "semi-informative" prior for the latent variable coefficients, as this tends to be produce more stable MCMC sampling particularly if the response matrix is large and sparse.

Using information criteria at your own risk: Using information criterion from `calc.ics = TRUE` for model selection should be done with extreme caution, for two reasons: 1) The implementation of some of these criteria is heuristic and experimental, 2) Deciding what model to fit should also be driven by the science. For example, it may be the case that a criterion suggests a model with 3 or 4 latent variables is more appropriate. However, if we are interested in visualizing the data for ordination purposes, then models with 1 or 2 latent variables are more appropriate. As another example, whether or not we include row effects when ordinating multivariate abundance data depends on if we are interested in differences between sites in terms of relative species abundance (`row.eff = "none"`) or species composition (`row.eff = "fixed"`).

We also make the important point that if traits are included in the model, then the regression coefficients $\beta_{0j}, \boldsymbol{\beta}_j$ are now random effects. However, currently the calculation of all information criteria do not take this into account!

SSVS: As an alternative to using information criterion for model selection, Stochastic Search Variable Selection (SSVS, George and McCulloch, 1993) is also implemented for the column-specific coefficients $\boldsymbol{\beta}_j$. Basically, SSVS works by placing a spike-and-slab priors on these coefficients, such that the spike is a narrow normal distribution concentrated around zero and the spike is a normal distribution with a large variance.

$$\rho(\beta) = I_{\beta=1} \times \mathcal{N}(0, \sigma^2) + (1 - I_{\beta=1}) \times \mathcal{N}(0, 0.0001 * \sigma^2),$$

where σ^2 is determined by `hypparams[3]` (see section on estimation above) and $I_{\beta=1} = P(\beta = 1)$ is an indicator function representing whether coefficient is included in the model. It is given a Bernoulli prior with probability of inclusion 0.5. After fitting, the posterior probability of β being included in the model is returned based on posterior mean of the indicator function $I_{\beta=1}$. Note this is NOT the same as a *p*-value seen in maximum likelihood estimation – a *p*-value provides an indication of how much evidence there is against the null hypothesis of $\beta = 0$, while the posterior probability provides a measure of how likely it is for $\beta \neq 0$ given the data.

In boral, SSVS can be applied at a grouped or individual coefficient level, and this is governed by `ssvs.index`. For elements of `ssvs.index` equal to -1, SSVS is not applied on the corresponding covariate of the model matrix `X`. For elements equal to 0, SSVS is applied to each individual coefficient of the corresponding covariate in `X`. That is, the fitted model will return p posterior probabilities for this covariate, one for each column of `y`. For elements taking positive integers 1,2,..., SSVS is applied to each group of coefficients of the corresponding covariate in `X`. That is, the fitted model will return a single posterior probability for this covariate, indicating whether this covariate should be included for all columns of `y`; see O'Hara and Sillanpaa (2009) for an discussion of Bayesian variable selection methods.

Note the last application of SSVS allows multiple covariates to be tested *simultaneously*. For example, suppose `X` consists of five columns – the first two columns are environmental covariates, while the last three correspond to quadratic terms of the two covariates as well as their interaction. If we want to "test" whether any quadratic terms are required, then we can set
`ssvs.index = c(-1,-1,1,1,1)`, so a single posterior probability of inclusion is returned for the last three columns of `X`.

Finally, note using information criterion (and possibly residual analysis) should probably not be done at the same as when SSVS is used, and it is advised to separate out their applications e.g., choose the explanatory variables first using SSVS, and then use information criterion to select the number of latent variables???

Value

An object of class "boral" is returned, being a list containing the following components where applicable:

`call` The matched call.

`lv.coefs.mean/median/sd/iqr`
Matrices containing the mean/median/standard deviation/interquartile range of the posterior distributions of the latent variable coefficients. This also includes the column-specific intercepts, and dispersion parameters if appropriate.

`lv.mean/median/sd/iqr`
A matrix containing the mean/median/standard deviation/interquartile range of the posterior distributions of the latent variables.

`X.coefs.mean/median/sd/iqr`
Matrices containing the mean/median/standard deviation/interquartile range of the posterior distributions of the column-specific coefficients relating to the model matrix `X`.

`traits.coefs.mean/median/sd/iqr`
: Matrices containing the mean/median/standard deviation/interquartile range of the posterior distributions of the coefficients and standard deviation relating to the species traits (please see the section on including traits above).

`cutoffs.mean/median/sd/iqr`
: Vectors containing the mean/median/standard deviation/interquartile range of the posterior distributions of the common cutoffs for ordinal responses (please see the not-so-brief tangent on distributions above).

`powerparam.mean/median/sd/iqr`
: Scalars for the mean/median/standard deviation/interquartile range of the posterior distributions of the common power parameter for tweedie responses (please see the not-so-brief tangent on distributions above).

`row.coefs.mean/median/sd/iq`
: Vectors containing the mean/median/standard deviation/interquartile range of the posterior distributions of the row effects.

`row.sigma.mean/median/sd/iqr`
: Scalars containing the mean/median/standard deviation/interquartile range of the posterior distributions of the standard deviation for the row random effects normal distribution.

`ssvs.indcoefs.mean/ssvs.indcoefs.sd`
: Matrices containing the SSVS posterior probabilities and associated standard deviation of including individual coefficients in the model (please see the section on SSVS above).

`ssvs.gpcoefs.mean/ssvs.gpcoefs.sd`
: Matrices containing the SSVS posterior probabilities and associated standard deviation of including grouped coefficients in the model (please see the section on SSVS above).

`hpdintervals`
: A list containing components which correspond to the lower and upper bounds of highest posterior density (HPD) intervals for all the parameters indicated above. Please see `get.hpdintervals` for more details.

`ics`
: If `calc.ics = TRUE`, then a list of different information criteria values for the model calculated using `get.measures` is run. Please see help file for `get.measures` regarding details on the criteria. Also, please note the ics returned are based on `get.measures` with `more.measures = FALSE`.

`jags.model`
: If `save.model = TRUE`, the raw jags model fitted is returned. This can be quite large!

`n, p, family, trial.size, num.lv, ...`
: Various attributes of the model fitted, including the dimension of `y`, the response and model matrix used, distributional assumptions and trial sizes, number of latent variables, the number of covariates and traits, whether information criteria values were calculated, hyperparameters used in the Bayesian estimation, indices for SSVS, the number of levels for ordinal responses, and n.burin, n.iteration and n.thin.

Why is only one MCMC chain run?

Much like the `MCMCfactanal` function in the `MCMCpack` package (Martin et al., 2011) for conducting factor analysis, which is a special case of the pure latent variable model with Gaussian responses, boral deliberately runs only one MCMC chain. This runs contrary to the recommendation of most Bayesian analyses, where the advice is to run multiple MCMC chains and check convergence using (most commonly) the Gelman-Rubin statistic or “Rhat” (Gelman et al., 2013). The main reason for this is that, in the context of MCMC sampling, the latent variable model is invariant to a switch of the sign, i.e. $\boldsymbol{z}_i^T\boldsymbol{\theta}_j = (-\boldsymbol{z})_i^T(-\boldsymbol{\theta}_j)$, and so is actually unidentifiable. This is similar to well-known problem of label switching that occurs during the course of MCMC sampling for mixture models (see for instance, Section 4.9, McLachlan and Peel, 2004), and is due to the fact that the sign of the latent variables (ordination coordinates) is inherently arbitrary.

As a result of this sign-switching problem, it means that different MCMC chains can produce latent variables and corresponding coefficients values that, while having similar magnitudes, will be different in sign. Consequently, combining MCMC chains and checking Rhats, computing posterior means and medians etc...becomes inappropriate (in principle, one way to resolve this problem would be to post-process the MCMC chains and deal with sign switching, but this is really hard!). Therefore, to alleviate this issue together, boral chooses to only run one MCMC chain.

What does this mean for the user?

- For checking convergence, we recommend you look at trace plots of the MCMC chains. Using the `coda` package, which is automatically loaded when the `boral` package is loaded, try something like `traceplot(fit$jags.model, ask = T)`. You could also try `geweke.diag` for Geweke’s convergence diagnostic, although no promises this necessarily does what is meant it!
- If you have a lot of data, e.g. lots of sites compared to species, sign-switching tends to be less of problem and pops up less often.
- IMPORTANTLY, if the goal of your analysis is to inference while account for residual correlations between the columns of `y`, and not for model-based ordination, then the sign-switching problem is not a problem at all! This is because while the signs of the latent variables and associated coefficients may switch, the correlation and their signs are unaffected. In other words, looking the point estimates and credible intervals of regression coefficients $\boldsymbol{\beta}_j$, and functions like `get.residual.cor` are unaffected by sign-switching.

Warnings

- *No* intercept column is required in `X`. Column-specific intercepts are estimated automatically and given by the first column of `lv.coefs`.
- If num.lv > 5, a warning is printed asking whether you really want to fit an boral with more than five latent variables. A warning is also printed if num.lv == 1, as this is not going to be successful in modeling between the correlation between columns.
- For models including both explanatory covariates and latent variables, one requires `num.lv > 1` to allow flexible modeling of the residual correlation matrix.
- MCMC can take a long time to run, especially with if the response matrix `y` is large! The calculation of information criteria (`calc.ics = TRUE`) can also take a while. Apologies for this advance =(
- MCMC with lots of ordinal columns take an especially long time to run! Moreover, estimates for the cutoffs in cumulative probit regression may be poor for levels with little data. Major apologies for this advance =(

- As discussed in the details, the use of information criterion should be done so with caution. What model to select should be first and foremost driven by the question of interest. Also, the use of information criterion in the presence of model seelction using SSVS is questionable.
- If `save.model = TRUE`, the raw jags model is also returned. This can be quite very memory-consuming, since it indirectly saves all the MCMC samples.

Author(s)

Francis K.C. Hui `<fhui28@gmail.com>`

References

- Agresti, A. (2010). Analysis of Ordinal Categorical Data. Wiley.
- Brown, et al. (2014). The fourth-corner solution - using predictive models to understand how species traits interact with the environment. Methods in Ecology and Evolution 5, 344-352.
- Foster, S. D. and Bravington, M. V. (2013). A Poisson-Gamma model for analysis of ecological non-negative continuous data. Journal of Environmental and Ecological Statistics, 20, 533-552.
- Gelman et al. (2013) Bayesian Data Analysis. CRC Press.
- Gelman A. (2006) Prior distributions for variance parameters in hierarchical models. Bayesian Analysis 1, 515-533.
- George, E. I. and McCulloch, R. E. (1993). Variable selection via Gibbs sampling. Journal of the American Statistical Association, 85, 398-409.
- Hui et al. (2014). Model-based approaches to unconstrained ordination. Methods in Ecology and Evolution, 6, 399-411.
- Hill, M. O. (1974). Correspondence analysis: a neglected multivariate method. Applied statistics, 23, 340-354.
- Jamil, T., and ter Braak, C.J.F. (2013). Generalized linear mixed models can detect unimodal species-environment relationships. PeerJ 1: e95.
- Jamil, T. et al. (2013). Selecting traits that explain species-environment relationships: a generalized linear mixed model approach. Journal of Vegetation Science 24, 988-1000
- Korhonen, L., et al. (2007). Local models for forest canopy cover with beta regression. Silva Fennica, 41, 671-685.
- Kruskal, J. B. (1964). Nonmetric multidimensional scaling: a numerical method. Psychometrika, 29, 115-129.
- Legendre, P. and Gallagher, E. D. (2001). Ecologically meaningful transformations for ordination of species data. Oecologia, 129, 271-280. Numerical ecology, Volume 20. Elsevier.
- Legendre, P. and Legendre, L. (2012). Numerical ecology, Volume 20. Elsevier.
- Martin et al. (2011). MCMCpack: Markov Chain Monte Carlo in R. Journal of Statistical Software, 42, 1-21. URL: http://www.jstatsoft.org/v42/i09/.
- McLachlan, G., and Peel, D. (2004). Finite Mixture Models. Wiley.
- Moles et al. (2013). Correlations between physical and chemical defences in plants: Tradeoffs, syndromes, or just many different ways to skin a herbivorous cat? New Phytologist, 198, 252-263.

- O'Hara, B., and Sillianpaa, M.J. (2009). A Review of Bayesian Variable Selection Methods: What, How and Which. Bayesian Analysis, 4, 85-118.
- Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In Proceedings of the 3rd International Workshop on Distributed Statistical Computing. March (pp. 20-22).
- Pollock, L. J. et al. (2014). Understanding co-occurrence by modelling species simultaneously with a Joint Species Distribution Model (JSDM). Methods in Ecology and Evolution, 5, 397-406.
- Skrondal, A., and Rabe-Hesketh, S. (2004). Generalized latent variable modeling: Multilevel, longitudinal, and structural equation models. CRC Press.
- Smithson, M., and Verkuilen, J. (2006). A better lemon squeezer? Maximum-likelihood regression with beta-distributed dependent variables. Psychological methods, 11, 54-71.
- Warton et al. (2015). So Many Variables: Joint Modeling in Community Ecology. Trends in Ecology and Evolution, in review.
- Warton et al. (2012). Distance-based multivariate analyses confound location and dispersion effects. Methods in Ecology and Evolution, 3, 89-101.
- Wang et al. (2013). `mvabund`: statistical methods for analysing multivariate abundance data. R package version 3.8.4.

See Also

`lvsplot` for a scatter plot of the latent variables (and their coefficients if applicable) when `num.lv = 1` or 2, `summary.boral` for a summary of the fitted boral model, `get.measures` and `get.more.measures` for information criteria from the fitted boral model, `get.residual.cor` for calculating the residual correlation matrix.

Examples

```
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - model with two latent variables, site effects,
##  and no environmental covariates
spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed", n.burnin = 10, n.iteration = 100,
     n.thin = 1, calc.ics = FALSE)

summary(spider.fit.nb)

plot(spider.fit.nb, ask = FALSE, mfrow = c(2,2)) ## Plots used in residual analysis,
## Used to check if assumptions such an mean-variance relationship
## are adequately satisfied.

lvsplot(spider.fit.nb) ## Biplot of the latent variables,
## which can be interpreted in the same manner as an ordination plot.
## Example 2 - model with no latent variables, no site effects,
##  and environmental covariates
X <- scale(spider$x)
spider.fit.nb <- boral(y, X = X, family = "negative.binomial",
num.lv = 0, n.burnin = 10, n.iteration = 100, n.thin = 1)

summary(spider.fit.nb)
## The results can be compared with the default example from
## the manyglm() function in mvabund. Hopefully they are similar =D


## Example 3 - Extend example 2 to demonstrate grouped and individual
## covariate selection
spider.fit.nb2 <- boral(y, X = X, family = "negative.binomial",
num.lv = 0, n.burnin = 10, n.iteration = 100, n.thin = 1,
calc.ics = FALSE, ssvs.index = c(-1,-1,-1,0,1,2))

summary(spider.fit.nb2)


## Example 3 - model fitted to presence-absence data, no site effects, and
## two latent variables
data(tikus)
y <- tikus$abun
y[y > 0] <- 1
y <- y[1:20,] ## Consider only years 1981 and 1983
y <- y[,apply(y > 0,2,sum) > 2] ## Consider only spp with more than 2 presences

tikus.fit <- boral(y, family = "binomial", num.lv = 2,
n.burnin = 10, n.iteration = 100, n.thin = 1, calc.ics = FALSE)

lvsplot(tikus.fit, biplot = FALSE)
## A strong location between the two sampling years


## Example 4 - model fitted to count data, no site effects, and
## two latent variables, plus traits included to explain environmental responses
data(antTraits)
y <- antTraits$abun
X <- as.matrix(scale(antTraits$env))
## Include only traits 1, 2, and 5
traits <- as.matrix(cbind(1,antTraits$traits[,c(1,2,5)]))
which.traits <- vector("list",ncol(X)+1)
for(i in 1:length(which.traits)) which.traits[[i]] <- 1:ncol(traits)
## Just for fun, the regression coefficients for the second column of X
## will be estimated separately and not regressed against traits.
which.traits[[3]] <- 0

fit.traits <- boral(y, X = X, traits = traits, which.traits = which.traits,
family = "negative.binomial", num.lv = 2, n.burnin = 10, n.iteration = 100,
n.thin = 1, calc.ics = FALSE)

summary(fit.traits)
```

calc.condlogLik *Conditional log-likelihood for an boral model*

Description

Calculates the conditional log-likelihood for a set of parameter estimates from an boral model, whereby everything is treated as "fixed effects" (including latent variables, row effects, and so on).

Usage

```
calc.condlogLik(y, X = NULL, family, trial.size = 1, lv.coefs,
X.coefs = NULL, row.coefs = NULL, lv, cutoffs = NULL,
     powerparam = NULL)
```

Arguments

y The response matrix the boral model was fitted to.

X The model matrix used in the boral model. Defaults to NULL, in which case it is assumed no model matrix was used.

family Either a single element, or a vector of length equal to the number of columns in y. The former assumes all columns of y come from this distribution. The latter option allows for different distributions for each column of y. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

trial.size Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`lv.coefs` The column-specific intercept, coefficient estimates relating to the latent variables, and dispersion parameters from the boral model.

`X.coefs` The coefficients estimates relating to the model matrix `X` from the boral model. Defaults to `NULL`, in which it is assumed there are no covariates in the model.

`row.coefs` Row effect estimates for the boral model. Even if they were estimated as random effects, the conditional log-likelihood is defined conditional on these estimates i.e., they are (also) treated as “fixed effects". Defaults to NULL, in which case it is assumed there are no row effects in the model.

`lv` Latent variables "estimates" from the boral model, which the conditional log-likelihood is based on. For boral models with no latent variables, please use `calc.logLik.lv0` to calculate the conditional log-likelihood.

`cutoffs` Common cutoff estimates from the boral model when any of the columns of `y` are ordinal responses. Defaults to `NULL`.

`powerparam` Common power parameter from the boral model when any of the columns of `y` are tweedie responses. Defaults to `NULL`.

Details

For an nxp response matrix `y`, suppose we fit an boral model with one or more latent variables. If we denote the latent variables by $\boldsymbol{z}_i; i = 1,\ldots,n$, then the conditional log-likelihood is given by,

$$\log(f) = \sum_{i=1}^{n}\sum_{j=1}^{p}\log(f(y_{ij}|\boldsymbol{z}_i,\boldsymbol{\theta}_j,\beta_{0j},\ldots)),$$

where $f(y_{ij}|\cdot)$ is the assumed distribution for column j, $\boldsymbol{z}_i$ are the latent variables and $\boldsymbol{\theta}_j$ are the coefficients relating to them, β_{0j} are column-specific intercepts, and $\ldots$ denotes anything else included in the model, such as row effects, regression coefficients related `X` and `traits`, etc...

The key difference between this and the marginal likelihood (see `calc.marglogLik`) is that the conditional log-likelihood treats everything as "fixed effects", while the marginal log-likelihood treats the latent variables and row effects (if `row.eff = "random"` as random and integrates over them.

Value

A list with the following components:

`logLik` Value of the conditional log-likelihood.

`logLik.comp` A vector of the log-likelihood values for each row of `y`,
such that `sum(logLik.comp) = logLik`.

Note

The conditional DIC, WAIC, EAIC, and EBIC returned from `get.measures` are based on the conditional log-likelihood calculated from this function. Additionally, `get.measures` returns the conditional log-likelihood evaluated at all MCMC samples of a fitted boral model.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

get.measures for some information criteria based on the conditional log-likelihood; calc.marglogLik for calculation of the marginal log-likelihood; calc.logLik.lv0 to calculate the conditional/marginal log-likelihood for an boral model with no latent variables.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - model with 2 latent variables, site effects,
##  and no environmental covariates
spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
row.eff = "fixed", save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
site.coef.median <- apply(fit.mcmc[,grep("row.params", colnames(fit.mcmc))],
     2,median)
lvs.mat <- matrix(apply(fit.mcmc[,grep("lvs",colnames(fit.mcmc))],2,median),nrow=n)

## Caculate the conditional log-likelihood at the posterior median
calc.condlogLik(y, family = "negative.binomial",
     lv.coefs =  coef.mat, row.coefs = site.coef.median, lv = lvs.mat)


## Example 2 - model with two latent variables and environmental covariates
X <- scale(spider$x)
spider.fit.nb2 <- boral(y, X = X, family = "negative.binomial", num.lv = 2,
     save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb2$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
X.coef.mat <- matrix(apply(fit.mcmc[,grep("X.params",colnames(fit.mcmc))],
2,median),nrow=p)
lvs.mat <- matrix(apply(fit.mcmc[,grep("lvs",colnames(fit.mcmc))],2,median),nrow=n)
```

```
## Caculate the log-likelihood at the posterior median
calc.condlogLik(y, X = X, family = "negative.binomial",
lv.coefs =  coef.mat, X.coefs = X.coef.mat, lv = lvs.mat)

## End(Not run)
```

calc.logLik.lv0 *Log-likelihood for a boral model with no latent variables*

Description

Calculates the log-likelihood for a set of parameter estimates from an boral model with no latent variables. If the row effects are assumed to be random, then they are integrated over using Monte Carlo integration.

Usage

```
calc.logLik.lv0(y, X = NULL, family, trial.size = 1, lv.coefs,
X.coefs = NULL, row.eff = "none", row.params = NULL, cutoffs = NULL,
     powerparam = NULL)
```

Arguments

y	The response matrix the boral model was fitted to.
X	The model matrix used in the boral model. Defaults to NULL, in which case it is assumed no model matrix was used.
family	Either a single element, or a vector of length equal to the number of columns in y. The former assumes all columns of y come from this distribution. The latter option allows for different distributions for each column of y. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression). For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`lv.coefs` The column-specific intercept, coefficient estimates relating to the latent variables, and dispersion parameters from the boral model.

`X.coefs` The coefficients estimates relating to the model matrix `X` from the boral model. Defaults to `NULL`, in which it is assumed there are no covariates in the model.

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and standard deviation given by `row.params`. Defaults to "none".

`row.params` Parameters corresponding to the row effect from the boral model. If `row.eff = "fixed"`, then these are the fixed effects and should have length equal to the number of columns in `y`. If `row.eff = "random"`, then this is the standard deviation for the random effects normal distribution. If `row.eff = "none"`, then this argument is ignored.

`cutoffs` Common cutoff estimates from the boral model when any of the columns of `y` are ordinal responses. Defaults to `NULL`.

`powerparam` Common power parameter from the boral model when any of the columns of `y` are tweedie responses. Defaults to `NULL`.

Details

For an nxp response matrix `y`, the log-likelihood for a model with no latent variables included is given by,

$$\log(f) = \sum_{i=1}^{n} \sum_{j=1}^{p} \log(f(y_{ij} | \beta_{0j}, \alpha_i, \ldots)),$$

where $f(y_{ij}|\cdot)$ is the assumed distribution for column j, β_{0j} is the column-specific intercepts, α_i is the row effect, and ... generically denotes anything else included in the model, e.g. row effects, dispersion parameters etc...

If the row effects are assumed to be random (`row.eff = "random"`), then the log-likelihood is calculated by integrating over them,

$$\log(f) = \sum_{i=1}^{n} \log(\int \prod_{j=1}^{p} (f(y_{ij} | \beta_{0j}, \alpha_i, \ldots)) f(\alpha_i) d\alpha_i),$$

where $f(\alpha_i)$ is the random effects distribution with mean zero and standard deviation given by the `row.params`. The integration is performed using Monte Carlo methods.

Note that if traits are included in the model, then the regression coefficients $\beta_{0j}, \boldsymbol{\beta}_j$ are now random effects. However, currently the calculation of the log-likelihood does NOT take this into account, i.e. does not marginalize over them!

Value

A list with the following components:

`logLik` Value of the log-likelihood

`logLik.row.comp`
A vector of the log-likelihood values for each row of y, such that `sum(logLik.row.comp) = logLik`. This is only returned if `row.eff` was not set to "random".

`logLik.col.comp`
A vector of the log-likelihood values for each column of y, such that `sum(logLik.row.comp) = logLik`. This is only returned if `row.eff` was not set to "random".

`logLik.comp` A vector of the log-likelihood values for each row of y, such that `sum(logLik.comp) = logLik`. This is only returned if `row.eff = "random"`.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

See Also

`calc.marglogLik` for calculation of the log-likelihood marginalizing over one or more latent variables, and `calc.condlogLik` for calculation of the conditional log-likelihood for boral models with one or more latent variables (and random row effects if applicable).

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - NULL model with site effects and species specific intercepts
spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 0,
     row.eff = "fixed", save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
site.coef.median <- apply(fit.mcmc[,grep("row.params", colnames(fit.mcmc))],
     2,median)
```

```
## Calculate the log-likelihood at the posterior median
calc.logLik.lv0(y, family = "negative.binomial",
     lv.coefs =  coef.mat, row.eff = "fixed", row.params = site.coef.median)


## Example 2 - Model without site effects, latent variables,
##   but includes environmental covariates
X <- scale(spider$x)
spider.fit.nb2 <- boral(y, X = X, family = "negative.binomial", num.lv = 0,
     save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb2$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
X.coef.mat <- matrix(apply(fit.mcmc[,grep("X.params",colnames(fit.mcmc))],
2,median),nrow=p)

## Calculate the log-likelihood at the posterior median
calc.logLik.lv0(y, X = spider$x, family = "negative.binomial",
lv.coefs =  coef.mat, X.coefs = X.coef.mat)

## End(Not run)
```

calc.marglogLik *Marginal log-likelihood for an boral model*

Description

Calculates the marginal log-likelihood for a set of parameter estimates from an boral model, whereby the latent variables and random effects (if applicable) are integrated out. The integration is performed using Monte Carlo integration.

Usage

```
calc.marglogLik(y, X = NULL, family, trial.size = 1, lv.coefs,
     X.coefs = NULL, row.eff = "none", row.params = NULL, num.lv,
     X.mc = NULL, cutoffs = NULL, powerparam = NULL)
```

Arguments

`y` The response matrix that the boral model was fitted to.

`X` The model matrix used in the boral model. Defaults to `NULL`, in which case it is assumed no model matrix was used.

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`lv.coefs` The column-specific intercept, coefficient estimates relating to the latent variables, and dispersion parameters from the boral model.

`X.coefs` The coefficients estimates relating to the model matrix `X` from the boral model. Defaults to `NULL`, in which it is assumed there are no covariates in the model.

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and standard deviation given by `row.params`. Defaults to "none".

`row.params` Parameters corresponding to the row effect from the boral model. If `row.eff = "fixed"`, then these are the fixed effects and should have length equal to the number of columns in `y`. If `row.eff = "random"`, then this is standard deviation for the random effects normal distribution. If `row.eff = "none"`, then this argument is ignored.

`num.lv` The number of latent variables used in the boral model. For boral models with no latent variables, please use `calc.logLik.lv0` to calculate the marginal log-likelihood.

`X.mc` A matrix used for performing the Monte Carlo integration. Defaults to `NULL`, in which case a matrix is generated within the function.

`cutoffs` Common cutoff estimates from the boral model when any of the columns of `y` are ordinal responses. Defaults to `NULL`.

`powerparam` Common power parameter from the boral model when any of the columns of `y` are tweedie responses. Defaults to `NULL`.

Details

For an nxp response matrix `y`, suppose we fit an boral model with one or more latent variables. If we denote the latent variables by $\boldsymbol{z}_i; i = 1, \ldots, n$, then the marginal log-likelihood is given by

$$\log(f) = \sum_{i=1}^{n} \log(\int \prod_{j=1}^{p} f(y_{ij}|\boldsymbol{z}_i, \beta_{0j}, \boldsymbol{\theta}_j, \ldots) f(\boldsymbol{z}_i) d\boldsymbol{z}_i),$$

where $f(y_{ij}|\cdot)$ is the assumed distribution for column j, β_{0j} are the column-specific intercepts, $\boldsymbol{\theta}_j$ are the column-specific latent variable coefficients, and $\ldots$ generically denotes anything else included in the model, e.g. row effects, dispersion parameters etc... The quantity $f(\boldsymbol{z}_i)$ denotes the distribution of the latent variable, which is assumed to be standard multivariate Gaussian.

If the row effects are assumed to be random (`row.eff = "random"`), then the log-likelihood is calculated by integrating over these as well,

$$\log(f) = \sum_{i=1}^{n} \log(\int \prod_{j=1}^{p} (f(y_{ij}|\boldsymbol{z}_i, \beta_{0j}, \boldsymbol{\theta}_j, \alpha_i, \ldots)) f(\boldsymbol{z}_i) f(\alpha_i) d\boldsymbol{z}_i d\alpha_i),$$

where $f(\alpha_i)$ is the random effects distribution with standard deviation given by `row.params`.

The key difference between this and the conditional likelihood (see `calc.condlogLik`) is that the marginal log-likelihood treats the latent variables as "random effects" and integrates over them, whereas the conditional log-likelihood treats the latent variables as "fixed effects".

Monte Carlo integration is used for calculating the marginal log-likelihood. If `X.mc = NULL`, the function automatically generates a matrix as
`X.mc <- cbind(1, rmvnorm(2000, rep(0,num.lv)))`. If there is need to apply this function numerous times, we recommend a matrix be inserted into `X.mc` to speed up computation.

Note that if traits are included in the model, then the regression coefficients $\beta_{0j}, \boldsymbol{\beta}_j$ are now random effects. However, currently the calculation of the marginal log-likelihood does NOT take this into account, i.e. does not marginalize over them!

Value

A list with the following components:

`logLik` Value of the marginal log-likelihood.

`logLik.comp` A vector of the log-likelihood values for each row of `y`,
such that `sum(logLik.comp) = logLik`.

Note

The AIC and BIC at posterior median returned from `get.measures` are all based on the marginal log-likelihood calculated from this function. Additionally, `get.more.measures` returns even more information criteria based on the marginal log-likelihood. As mentioned in the details though, these information criteria do not take into acocunt that traits are included in the model!

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

get.measures and get.more.measures for information criteria based on the marginal log-likelihood; calc.condlogLik for calculation of the conditional log-likelihood; calc.logLik.lv0 to calculate the conditional/marginal log-likelihood for an boral model with no latent variables.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - model with two latent variables, site effects,
##  and no environmental covariates
spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed", save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
site.coef.median <- apply(fit.mcmc[,grep("row.params", colnames(fit.mcmc))],
     2,median)

## Caculate the marginal log-likelihood at the posterior median
calc.marglogLik(y, family = "negative.binomial",
lv.coefs = coef.mat, row.eff = "fixed", row.params = site.coef.median,
num.lv = 2)


## Example 2 - model with one latent variable, no site effects,
##  and environmental covariates
spider.fit.nb2 <- boral(y, X = spider$x, family = "negative.binomial",
     num.lv = 2, save.model = TRUE, calc.ics = FALSE)

## Extract all MCMC samples
fit.mcmc <- mcmc(spider.fit.nb2$jags.model$BUGSoutput$sims.matrix)

## Find the posterior medians
coef.mat <- matrix(apply(fit.mcmc[,grep("all.params",colnames(fit.mcmc))],
     2,median),nrow=p)
X.coef.mat <- matrix(apply(fit.mcmc[,grep("X.params",colnames(fit.mcmc))],
2,median),nrow=p)

## Caculate the log-likelihood at the posterior median
```

```
calc.marglogLik(y, X = spider$x, family = "negative.binomial",
lv.coefs = coef.mat, X.coefs = X.coef.mat, num.lv = 2)

## End(Not run)
```

`create.life` *Simulate a Multivariate Response Matrix*

Description

Simulate a multivariate response matrix, given parameters such as but not necessarily all of: family, number of latent variables and related coefficients, an matrix of explanatory variables and related coefficients, row effects, cutoffs for cumulative probit regression of ordinal responses.

Usage

```
create.life(true.lv = NULL, lv.coefs, X = NULL, X.coefs = NULL,
     traits = NULL, traits.coefs = NULL, family, row.eff = "none",
     row.params = NULL, trial.size = 1, cutoffs = NULL,
     powerparam = NULL, manual.dim = NULL)

## S3 method for class 'boral'
simulate(object, nsim = 1, seed = NULL, est = "median", ...)
```

Arguments

`object` An object of class "boral".

`nsim` Number of multivariate response matrices to simulate. Defaults to 1.

`seed` Seed for dataset simulation. Defaults to NULL, in which case no seed is set.

`est` A choice of either the posterior median (`est == "median"`) or posterior mean (`est == "mean"`), which are then treated as estimates and the fitted values are calculated from. Default is posterior median.

`true.lv` A matrix of true latent variables. With multivariate abundance data in ecology for instance, each row corresponds to the true site ordination coordinates. Defaults to `NULL`, in which case no latent variables are included.

`lv.coefs` A matrix containing column-specific intercepts, latent variable coefficients relating to `true.lv`, and dispersion parameters.

`X` An model matrix of covariates, which can be included as part of the data generation. Defaults to `NULL`, in which case no model matrix is used. No intercept column should be included in `X`.

`X.coefs` The coefficients relating to the model matrix `X`.

`traits` A model matrix of species covariates, which can be included as part of the data generation. Defaults to `NULL`, in which case no matrix is used. An intercept column should be included in `traits` if appropriate (usually is).

`traits.coefs` A matrix of coefficients that are used to generate "new" column-specific intercepts and `X.coefs`. The number of rows shoud equal to (`ncol(X)+1`) and the number of columns should equal to (`ncol(traits)+1`).

How this argument works is as follows: when both `traits` and `traits.coefs` are supplied, then new column-specific intercepts (i.e. the first column of `lv.coefs` is overwritten) are generated by simulating from a normal distribution with mean equal to `traits*    traits.coefs[1,-ncol(traits.coefs)]` and standard deviation `traits.coefs[1,ncol(traits.coefs)]`. In other words, the last column of `trait.coefs` provides the standard deviation of the normal distribution, with the other columns being the regression coefficients in the mean of the normal distribution. Analogously, new `X.coefs` are generated in the same manner using the remaining rows of `trait.coefs`. Please see the section on including species traits in the help file for `boral` for more information.

It is important that highlight then with in this data generation mechanism, the new column-specific intercepts and `X.coefs` are now random effects, being drawn from a normal distribution.

Defaults to `NULL`, in conjuction with `traits = NULL`.

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and standard deviation given by `row.params`. Defaults to "none".

`row.params` Parameters corresponding to the row effect from the boral model. If `row.eff = "fixed"`, then these are the fixed effects and should have length equal to the number of columns in `y`. If `row.eff = "random"`, then this is the standard deviation for the random effects normal distribution. If `row.eff = "none"`, then this argument is ignored.

- `trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.
- `cutoffs` A vector of common common cutoffs for proportional odds regression when any of `family` is ordinal. They should be increasing order. Defaults to `NULL`.
- `powerparam` A common power parameter for tweedie regression when any of `family` is tweedie. Defaults to `NULL`.
- `manual.dim` A vector of length 2, containing the number of rows (n) and columns (p) for the multivariate response matrix. This is a "backup" argument only required when `create.life` can not determine how many rows or columns the multivariate response matrix should be.
- `...` Not used.

Details

`create.life` gives the user full capacity to control the true parameters of the model from which the multivariate responses matrices are generated from.

`simulate` makes use of the generic function of the same name in `R`: it takes a fitted boral model, treats either the posterior medians and mean estimates from the model as the true parameters, and generates response matrices based off that.

Value

One of more multivariate response matrices of dimension n times p. If `simulate` is used, then an array is generated where the last dimension indexes the dataset number.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

See Also

`boral` for the default function for fitting a boral model.

Examples

```
## Example 1 - Simulate a response matrix of normally distributed data
library(mvtnorm)

## 30 rows (sites) with two latent variables
true.lv <- rbind(rmvnorm(n=15,mean=c(1,2)),rmvnorm(n=15,mean=c(-3,-1)))
## 30 columns (species)
lv.coefs <- cbind(matrix(runif(30*3),30,3),1)
```

```
X <- matrix(rnorm(30*4),30,4)
## 4 explanatory variables
X.coefs <- matrix(rnorm(30*4),30,4)

sim.y <- create.life(true.lv, lv.coefs, X, X.coefs, family = "normal")

## Not run:
fit.boral <- boral(sim.y, X = X, family = "normal", num.lv = 2)

summary(fit.boral)

## End(Not run)


## Example 2 - Simulate a response matrix of ordinal data

## 30 rows (sites) with two latent variables
true.lv <- rbind(rmvnorm(15,mean=c(-2,-2)),rmvnorm(15,mean=c(2,2)))
## 10 columns (species)
true.lv.coefs <- rmvnorm(10,mean = rep(0,3));
## Impose a sum-to-zero constraint on the column effects
true.lv.coefs[nrow(true.lv.coefs),1] <- -sum(true.lv.coefs[-nrow(true.lv.coefs),1])
## Cutoffs for proportional odds regression (must be in increasing order)
true.ordinal.cutoffs <- seq(-2,10,length=10-1)

sim.y <- create.life(true.lv = true.lv, lv.coefs = true.lv.coefs,
     family = "ordinal", cutoffs = true.ordinal.cutoffs)

## Not run:
fit.boral <- boral(y = sim.y, family = "ordinal", num.lv = 2)

## End(Not run)

## Not run:
## Example 3 - Simulate a response matrix of count data based off
## a fitted boral model involving traits (ants data from mvabund)
library(mvabund)
data(antTraits)

y <- antTraits$abun
X <- as.matrix(antTraits$env)
## Include only traits 1, 2, and 5, plus an intercept
traits <- as.matrix(cbind(1,antTraits$traits[,c(1,2,5)]))
## Please see help file for boral regarding the use of which.traits
which.traits <- vector("list",ncol(X)+1)
for(i in 1:length(which.traits)) which.traits[[i]] <- 1:ncol(traits)

fit.traits <- boral(y, X = X, traits = traits, which.traits = which.traits,
family = "poisson", num.lv = 2)

## The hard way
sim.y <- create.life(true.lv = NULL, lv.coefs = fit.traits$lv.coefs.median,
X = X, X.coefs = fit.traits$X.coefs.median,
traits = traits, traits.coefs = fit.traits$traits.coefs.median,
family = "poisson")

## The easy way
sim.y <- simulate(object = fit.traits)


## End(Not run)
```

ds.residuals *Dunn-Smyth Residuals for a boral model*

Description

Calculates the Dunn-Smyth residuals for a fitted boral model or, if some of the responses are ordinal, a table of agreement between predicted and true levels.

Usage

```
ds.residuals(object, est = "median")
```

Arguments

object	An object for class "boral".
est	A choice of either the posterior median (`est == "median"`) or posterior mean (`est == "mean"`), which are then treated as parameter estimates and the residuals are calculated from. Default is posterior median.

Details

Details regarding Dunn-Smyth residuals, based on the randomized quantile residuals of Dunn and Smyth (1996), can be found in `plot.manyglm` function in the `mvabund` package (Wang et al., 2012) where they are implemented in all their glory. Due their inherent stochasticity, Dunn-Smyth residuals will be slightly different each time this function is run. As with other types of residuals, Dunn-Smyth residuals can be used in the context of residual analysis.

For ordinal responses, a single table of agreement between the predicted levels (as based on the class with the highest probability) and true levels is returned. The table pools the results over all columns assumed to be ordinal.

The Dunn-Smyth residuals are calculated based on a point estimate of the parameters, as determined by the argument `est`. A fully Bayesian approach would calculate the residuals by averaging over the posterior distribution of the parameters i.e., ergodically average over the MCMC samples. In general however, the results (as in the trends seen in residual analysis) from either approach should be very similar.

Value

A list with potentially NULL elements, containing agree.ordinal which is a single table of agreement for ordinal columns, and residuals which contains Dunn-Smyth residuals.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

References

- Dunn, P. K., and Smyth, G. K. (1996). Randomized quantile residuals. Journal of Computational and Graphical Statistics, 5, 236-244.
- Wang, Y. et al. (2012). mvabund-an R package for model-based analysis of multivariate abundance data. Methods in Ecology and Evolution, 3, 471-474.

See Also

plot.boral for constructing residual analysis plots directly; fitted.boral which calculated fitted values from a boral model.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed")

ds.residuals(spider.fit.nb)

## End(Not run)
```

fitted.boral *Extract Model Fitted Values for an boral object*

Description

Calculated the predicted mean responses based on the fitted boral model, by using the posterior medians or means of the parameters.

Usage

```
## S3 method for class 'boral'
fitted(object, est = "median",...)
```

Arguments

object	An object of class "boral".
est	A choice of either the posterior median (est == "median") or posterior mean (est == "mean"), which are then treated as estimates and the fitted values are calculated from. Default is posterior median.
...	Not used.

Details

This fitted values here are calculated based on a point estimate of the parameters, as determined by the argument est. A fully Bayesian approach would calculate the fitted values by averaging over the posterior distribution of the parameters i.e., ergodically average over the MCMC samples. For simplicity and speed though (to avoid generation of a large number of predicted values), this is not implemented.

Value

A list with potential NULL elements in it, containing ordinal.probs which is an array with dimensions (no. of rows of y) x (no. of rows of y) x (no. of levels) containing the predicted probabilities for ordinal columns, and out which is a matrix of the same dimension as the original response matrix y containing the fitted values.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

plot.boral which uses the fitted values calculated from this function to construct plots for residual analysis; ds.residuals for calculating the Dunn-Smyth residuals for a fitted boral model.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed")

fitted(spider.fit.nb)

## End(Not run)
```

get.dic *Extract Deviance Information Criterion for boral model*

Description

Calculates the Deviance Information Criterion (DIC) for a boral model fitted using JAGS.

Usage

```
get.dic(jagsfit)
```

Arguments

jagsfit	The `jags.model` component of the output, from a model fitted using `boral` with `save.model = TRUE`.

Details

Details regarding the Deviance Information Criterion may be found in (Spiegelhalter et al., 2002; Ntzoufras, 2011; Gelman et al., 2013). The DIC here is based on the conditional log-likelihood i.e., the latent variables (and row effects if applicable) are treated as "fixed effects". A DIC based on the marginal likelihood is obtainable from `get.more.measures`, although this requires a much longer time to compute. For models with overdispered count data, conditional DIC may not perform as well as marginal DIC (Millar, 2009)

Value

DIC value for the jags model.

Note

This function and consequently the DIC value is automatically returned when a boral model is fitted using `boral` with `calc.ics = TRUE`.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

References

- Gelman et al. (2013). Bayesian data analysis. CRC press.
- Millar, R. B. (2009). Comparison of hierarchical Bayesian models for overdispersed count data using DIC and Bayes' factors. Biometrics, 65, 962-969.
- Ntzoufras, I. (2011). Bayesian modeling using WinBUGS (Vol. 698). John Wiley & Sons.
- Spiegelhalter, et al. (2002). Bayesian measures of model complexity and fit. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 64, 583-639.

See Also

get.measures and get.more.measures for other information criteria which could potentially be used for variable selection.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     save.model = TRUE, calc.ics = TRUE)

spider.fit.nb$ics ## DIC returned as one of several information criteria.

## End(Not run)
```

---

get.enviro.cor — *Extract covariances and correlations due to shared environmental responses from boral models*

Description

Calculates the correlation between columns of the response matrix, due to similarities in the response to explanatory variables (i.e., shared environmental response)

Usage

```
get.enviro.cor(object, est = "median", prob = 0.95)
```

Arguments

object	An object for class "boral".
est	A choice of either the posterior median (`est = "median"`) or posterior mean (`est = "mean"`), which are then treated as estimates and the fitted values are calculated from. Default is posterior median.
prob	A numeric scalar in the interval (0,1) giving the target probability coverage of the intervals, by which to determine whether the correlations are "significant". Defaults to 0.95.

Details

In both independent response and correlated response models, where the each of the columns of the response matrix `y` are fitted to a set of explanatory variables given by `X`, the covariance and thus between two columns j and j' due to similarities in their response to the model matrix is calculated based on the linear predictors $\boldsymbol{x}_i^T\boldsymbol{\beta}_j$ and $\boldsymbol{x}_i^T\boldsymbol{\beta}_{j'})$, where $\boldsymbol{\beta}_j$ are column-specific coefficients relating to the explanatory variables (see also the help file for `boral`).

For multivariate abundance data, the correlation calculated by this function can be interpreted as the correlation attributable to similarities in the environmental response between species. Such correlation matrices are discussed and found in Ovaskainen et al., (2010), Pollock et al., 2014.

Value

A list with the following components:

`cor` A $p \times p$ correlation matrix based on model matrix and the posterior or mean estimators of the associated regression coefficients.

`sig.cor` A $p \times p$ correlation matrix containing only the “significant" correlations whose 95% highest posterior interval does not contain zero. All non-significant correlations are zero to zero.

`cov` A $p \times p$ covariance matrix based on model matrix and the posterior or mean estimators of the associated regression coefficients.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

References

- Ovaskainen et al. (2010). Modeling species co-occurrence by multivariate logistic regression generates new hypotheses on fungal interactions. Ecology, 91, 2514-2521.
- Pollock et al. (2014). Understanding co-occurrence by modelling species simultaneously with a Joint Species Distribution Model (JSDM). Methods in Ecology and Evolution, 5, 397-406.

See Also

`get.residual.cor`, which calculates the residual correlation matrix for boral models involving latent variables.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
library(corrplot) ## For plotting correlations
data(spider)
y <- spider$abun
X <- scale(spider$x)
n <- nrow(y); p <- ncol(y);

spider.fit.nb <- boral(y, X = X, family = "negative.binomial",
     save.model = TRUE)

enviro.cors <- get.enviro.cor(spider.fit.nb)

corrplot(enviro.cors$sig.cor, title = "Shared response correlations",
type = "lower", diag = FALSE, mar = c(3,0.5,2,1), tl.srt = 45)

## End(Not run)
```

get.hpdintervals *Highest posterior density intervals for an boral model*

Description

Calculates the lower and upper bounds of the highest posterior density intervals for parameters and latent variables in a fitted boral model.

Usage

```
get.hpdintervals(y, X = NULL, traits = NULL, fit.mcmc, num.lv, prob = 0.95)
```

Arguments

y	The response matrix that the boral model was fitted to.
X	The model matrix used in the boral model. Defaults to NULL, in which case it is assumed no model matrix was used.
traits	The matrix of species traits used in the boral model. Defaults to NULL, in which case it is assumed no traits were included.
fit.mcmc	All MCMC samples for the fitted boral model, as obtained from JAGS. These can be extracted by fitting an boral model using boral with save.model = TRUE, and then accessing the jags.model component of the output.
num.lv	The number of latent variables used in the boral model. If zero, then HPD intervals will not be produced for latent variables.
prob	A numeric scalar in the interval (0,1) giving the target probability coverage of the intervals. Defaults to 0.95.

Details

The function uses the HPDinterval function from the coda package to obtain the HPD intervals. See HPDinterval for details regarding the definition of the HPD interval.

Value

`lv.coefs.hpd.lower/upper`
: Two matrices corresponding to the lower and upper bounds of the HPD intervals for the column-specific intercepts, latent variable coefficients, and dispersion parameters if appropriate.

`lv.hpd.lower/upper`
: Two matrices corresponding to the lower and upper bounds of the HPD intervals for the latent variables.

`row.coefs.lower/upper`
: Two vectors corresponding to the lower and upper bounds of the HPD intervals for row effects.

`row.sigma.lower/upper`
: Two scalars corresponding to the lower and upper bounds of the HPD interval for the standard deviation of the normal distribution for the row effects, if they were assumed to be random.

`X.coefs.hpd.lower/upper`
: Two matrices corresponding to the lower and upper bounds of the HPD intervals for coefficients relating to the model matrix `X`.

`traits.coefs.hpd.lower/upper`
: Two matrices corresponding to the lower and upper bounds of the HPD intervals for coefficients and standard deviation relating to the traits matrix `traits`.

`cutoffs.hpd.lower/upper`
: Two vectors corresponding to the lower and upper bounds of the HPD intervals for common cutoffs in proportional odds regression.

`powerparam.hpd.lower/upper`
: Two scalars corresponding to the lower and upper bounds of the HPD interval for common power parameter in tweedie regression.

Warnings

- HPD intervals tend to be quite wide, and inference is somewhat tricky with them. This is made more difficult by the multiple comparison problem due to the construction one interval for each parameter!
- Be very careful with interpretation of coefficients and HPD intervals if different columns of `y` have different distributions!
- HPD intervals for the cutoffs in proportional odds regression may be poorly estimated for levels with few data.

Note

`boral` fits the boral model and returns the HPD intervals by default.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - model with two latent variables, site effects,
##  and no environmental covariates
spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed", save.model = TRUE)

## Returns a list with components corresponding to values described above.
spider.fit.nb$hpdintervals

## Example 2 - model with two latent variable, site effects,
##  and environmental covariates
spider.fit.nb2 <- boral(y, X = spider$x, family = "negative.binomial",
num.lv = 2, row.eff = "fixed", save.model = TRUE, hypparams = c(100,20,100,50))

## Returns a list with components corresponding to values described above.
spider.fit.nb2$hpdintervals


## End(Not run)
```

get.measures *Information Criteria for boral models*

Description

Calculates some information criteria for an boral model, which could be used for model selection.

Usage

```
get.measures(y, X = NULL, family, trial.size = 1, row.eff = "none",
num.lv, fit.mcmc, more.measures = FALSE)
```

Arguments

`y` The response matrix that the boral model was fitted to.

`X` The model matrix used in the boral model. Defaults to `NULL`, in which case it is assumed no model matrix was used.

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and unknown standard deviation. Defaults to "none".

`num.lv` The number of latent variables used in the fitted boral model.

`fit.mcmc` All MCMC samples for the fitted boral model, as obtained from JAGS. These can be extracted by fitting an boral model using `boral` with `save.model = TRUE`, and then accessing the `jags.model` component of the output.

`more.measures` A logical value indicating whether to run `get.more.measures` to obtain additional information criteria.

Details

The following information criteria are currently implemented: 1) Widely Applicable Information Criterion (WAIC, Watanabe, 2010) based on the conditional log-likelihood; 2) expected AIC (EAIC, Carlin and Louis, 2011); 3) expected BIC (EBIC, Carlin and Louis, 2011); 4) AIC (using the marginal likelihood) evaluated at the posterior median; 5) BIC (using the marginal likelihood) evaluated at the posterior median.

1) WAIC has been argued to be more natural and extension of AIC to the Bayesian and hierarchical modeling context (Gelman et al., 2013), and is based on the conditional log-likelihood calculated at each of the MCMC samples.

2 & 3) EAIC and EBIC were suggested by (Carlin and Louis, 2011). Both criteria are of the form -2*mean(conditional log-likelihood) + penalty*(no. of parameters in the model), where the mean is averaged all the MCMC samples. EAIC applies a penalty of 2, while EBIC applies a penalty of $log(n)$.

4 & 5) AIC and BIC take the form -2*(marginal log-likelihood) + penalty*(no. of parameters in the model), where the log-likelihood is evaluated at the posterior median. If the parameter-wise posterior distributions are unimodal and approximately symmetric, these will produce similar results to an AIC and BIC where the log-likelihood is evaluated at the posterior mode. EAIC applies a penalty of 2, while EBIC applies a penalty of $log(n)$.

In our very limited experience, if information criteria are to be used for model selection between boral models, we found BIC at the posterior median tends to perform best. WAIC, AIC, and DIC (see `get.dic`) tend to over select the number of latent variables. For WAIC and DIC, part of this overfitting could be due to the fact both criteria are calculated from the conditional rather than the marginal log-likelihood (see Millar, 2009).

Intuitively, comparing boral models with and without latent variables (using information criteria such as those returned) amounts to testing whether the columns of the response matrix `y` are correlated. With multivariate abundance data for example, where `y` is a matrix of n sites and p species, comparing models with and without latent variables tests whether there is any evidence of correlation between species.

Note that if traits are included in the model, then the regression coefficients $\beta_{0j}, \boldsymbol{\beta}_j$ are now random effects. However, currently the calculation of all information criteria do not take this into account!

Value

A list with the following components:

`waic` WAIC based on the conditional log-likelihood.

`eaic` EAIC based on the mean of the conditional log-likelihood.

`ebic` EBIC based on the mean of the conditional log-likelihood.

`aic.median` AIC (using the marginal log-likelihood) evaluated at the posterior median.

`bic.median` BIC (using the marginal log-likelihood) evaluated at the posterior median.

`all.cond.logLik` The conditional log-likelihood evaluated at all MCMC samples. This is done via repeated application of `calc.condlogLik`.

`num.params` Number of estimated parameters used in the fitted model.

Warning

Using information criterion for variable selection should be done with extreme caution, for two reasons: 1) The implementation of these criteria are both *heuristic* and experimental. 2) Deciding what model to fit for ordination purposes should be driven by the science. For example, it may be the case that a criterion suggests a model with 3 or 4 latent variables. However, if we interested in visualizing the data for ordination purposes, then models with 1 or 2 latent variables are far more appropriate. As an another example, whether or not we include row effects when ordinating multivariate abundance data depends on if we are interested in differences between sites in terms of relative species abundance (`row.eff = FALSE`) or in terms of species composition (`row.eff = "fixed"`).

Also, the use of information criterion in the presence of variable selection using SSVS is questionable.

Note

When a boral model is fitted using boral with calc.ics = TRUE, then this function is applied with more.measures = FALSE, and the information criteria are returned as part of the model output.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

References

- Carlin, B. P., and Louis, T. A. (2011). Bayesian methods for data analysis. CRC Press.
- Gelman et al. (2013). Understanding predictive information criteria for Bayesian models. Statistics and Computing, 1-20.
- Millar, R. B. (2009). Comparison of hierarchical Bayesian models for overdispersed count data using DIC and Bayes' factors. Biometrics, 65, 962-969.
- Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. The Journal of Machine Learning Research, 11, 3571-3594.

See Also

get.dic for calculating the Deviance Information Criterion (DIC) based on the conditional log-likelihood; get.more.measures for even more information criteria.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

spider.fit.pois <- boral(y, family = "poisson",
num.lv = 2, row.eff = "random")

spider.fit.pois$ics ## Returns information criteria

spider.fit.nb <- boral(y, family = "negative.binomial",
num.lv = 2, row.eff = "random")

spider.fit.nb$ics ## Returns the information criteria

## End(Not run)
```

get.more.measures *Additional Information Criteria for boral models*

Description

Calculates some information criteria beyond those from `get.measures` for an boral model, although this set of criteria takes much longer to compute!!!

Usage

```
get.more.measures(y, X = NULL, family, trial.size = 1,
row.eff = "none", num.lv, fit.mcmc, verbose = TRUE)
```

Arguments

`y` The response matrix that the boral model was fitted to.

`X` The model matrix used in the boral model. Defaults to `NULL`, in which case it is assumed no model matrix was used.

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1-\mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and unknown standard deviation. Defaults to "none".

`num.lv` The number of latent variables used in the fitted boral model.

`fit.mcmc` All MCMC samples for the fitted boral model, as obtained from JAGS. These can be extracted by fitting an boral model using `boral` with `save.model = TRUE`, and then accessing the `jags.model` component of the output.

`verbose` If TRUE, a notice is printed every 100 samples indicating progress in calculation of the marginal log-likelihood. Defaults to TRUE.

Details

Currently, four information criteria has been implemented in this function: 1) AIC (using the marginal likelihood) evaluated at the posterior mode; 2) BIC (using the marginal likelihood) evaluated at the posterior mode; 3) Deviance information criterion (DIC) based on the marginal log-likelihood; 4) Widely Applicable Information Criterion (WAIC, Watanabe, 2010) based on the marginal log-likelihood. Since flat priors are used in fitting boral models, then the posterior mode should be approximately equal to the maximum likelihood estimates.

All four criteria require computing the marginal log-likelihood across all MCMC samples. This takes a very long time to run, since Monte Carlo integration needs to be performed for all MCMC samples. Consequently, this function is currently not implemented as an argument in main `boral` fitting function, unlike `get.measures` which is available via the `calc.ics = TRUE` argument.

The two main differences between the criteria and those returned from `get.measures` are:

- The AIC and BIC computed here are based on the log-likelihood evaluated at the posterior mode, whereas the AIC and BIC from `get.measures` are evaluated at the posterior median. The posterior mode and median will be quite close to one another if the component-wise posterior distributions are unimodal and symmetric. Furthermore, given uninformative priors are used, then both will be approximate maximum likelihood estimators.
- The DIC and WAIC computed here are based on the marginal log-likelihood, whereas the DIC and WAIC from `get.measures` are based on the conditional log-likelihood. Criteria based on the two types of log-likelihood are equally valid, and to a certain extent, which one to use depends on the question being answered i.e., whether to condition on the latent variables or treat them as "random effects" (see discussions in Spiegelhalter et al. 2002, and Vaida and Blanchard, 2005). Having said that, there is evidence to suggests, for models with overdispersed count data, conditional DIC/WAIC may not perform as well as than marginal DIC/WAIC for overdispered abundance data (Millar, 2009).

In our very limited experience, we found BIC evaluated at the posterior mode tends to be quite stable, whereas marginal DIC and WAIC tend to overfit the number of latent variables.

Note that if traits are included in the model, then the regression coefficients $\beta_{0j}, \boldsymbol{\beta}_j$ are now random effects. However, currently the calculation of all information criteria do not take this into account!

Value

A list with the following components:

`marg.aic` AIC (using on the marginal log-likelihood) evaluated at posterior mode.

`marg.bic` BIC (using on the marginal log-likelihood) evaluated at posterior mode.

`marg.dic` DIC based on the marginal log-likelihood.

`marg.waic` WAIC based on the marginal log-likelihood.

`all.marg.logLik`
The marginal log-likelihood evaluated at all MCMC samples. This is done via repeated application of `calc.marglogLik`.

`num.params` Number of estimated parameters used in the fitted model.

Warning

Using information criterion for variable selection should be done with extreme caution, for two reasons: 1) The implementation of these criteria are both *heuristic* and experimental. 2) Deciding what model to fit for ordination purposes should be driven by the science. For example, it may be the case that a criterion suggests a model with 3 or 4 latent variables. However, if we interested in visualizing the data for ordination purposes, then models with 1 or 2 latent variables are far more appropriate. As an another example, whether or not we include row effects when ordinating multivariate abundance data depends on if we are interested in differences between sites in terms of relative species abundance (`row.eff = FALSE`) or in terms of species composition (`row.eff = "fixed"`).

Also, the use of information criterion in the presence of variable selection using SSVS is questionable.

Note

This function can be run within `get.measures` by setting argument `more.measure = TRUE`.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

References

- Millar, R. B. (2009). Comparison of hierarchical Bayesian models for overdispersed count data using DIC and Bayes' factors. Biometrics, 65, 962-969.
- Spiegelhalter et al. (2002). Bayesian measures of model complexity and fit. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 64, 583-639.
- Vaida, F., and Blanchard, S. (2005). Conditional Akaike information for mixed-effects models. Biometrika, 92, 351-370.
- Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. The Journal of Machine Learning Research, 11, 3571-3594.

See Also

`get.measures` for several information criteria which take less time to compute, and are automatically implemented in `boral` with `calc.ics = TRUE`.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed", save.model = TRUE, calc.ics = TRUE)

## Extract MCMC samples
fit.mcmc <- mcmc(spider.fit.nb$jags.model$BUGSoutput$sims.matrix)

## WATCH OUT! The following takes a very long time to run!
get.more.measures(y, family = "negative.binomial",
     num.lv = 2, fit.mcmc = fit.mcmc, row.eff = "fixed")

## End(Not run)
```

`get.residual.cor` *Extract residual correlations from boral models*

Description

Calculates the residual correlation from models that include latent variables.

Usage

```
get.residual.cor(object, est = "median", prob = 0.95)
```

Arguments

`object` An object for class "boral".

`est` A choice of either the posterior median (`est = "median"`) or posterior mean (`est = "mean"`), which are then treated as estimates and the fitted values are calculated from. Default is posterior median.

`prob` A numeric scalar in the interval (0,1) giving the target probability coverage of the intervals, by which to determine whether the correlations are "significant". Defaults to 0.95.

Details

In models with latent variables, the residual covariance matrix is calculated based on the matrix of latent variables regression coefficients formed by stacking the rows of $\boldsymbol{\theta}_j$. That is, if we denote $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1 \ldots \boldsymbol{\theta}_p)'$, then the residual covariance and hence residual correlation matrix is calculated based on $\boldsymbol{\Theta}\boldsymbol{\Theta}'$ (see also the help file for `boral`).

For multivariate abundance data, the inclusion of latent variables provides a parsimonious method of accounting for correlation between species. Specifically, the linear predictor,

$$\beta_{0j} + \boldsymbol{x}_i^T \boldsymbol{\beta}_j + \boldsymbol{z}_i^T \boldsymbol{\theta}_j$$

is normally distributed with a residual covariance matrix given by $\boldsymbol{\Theta}\boldsymbol{\Theta}'$. A strong residual covariance/correlation matrix between two species can then be interpreted as evidence of species interaction (e.g., facilitation or competition), missing covariates, as well as any additional species correlation not accounted for by shared environmental responses (see also Pollock et al., 2014, for residual correlation matrices in the context of Joint Species Distribution Models).

In addition to the residual correlation matrix, the median or mean point estimator of trace of the residual covariance matrix is returned, $\sum\limits_{j=1}^{p} [\boldsymbol{\Theta}\boldsymbol{\Theta}']_{jj}$. Often used in other areas of multivariate statistics, the trace may be interpreted as the amount of covariation explained by the latent variables. One situation where the trace may be useful is when comparing a pure LVM versus a model with latent variables and some predictors (correlated response models) – the proportional difference in trace between these two models may be interpreted as the proportion of covariation between species explained by the predictors. Of course, the trace itself is random due to the MCMC sampling, and so it is not always guranteed to produce sensible answers =P

Value

A list with the following components:

`cor` A $p \times p$ residual correlation matrix based on posteriori median or mean estimators of the latent variables and coefficients.

`sig.cor` A $p \times p$ correlation matrix containing only the “significant" correlations whose 95% highest posterior interval does not contain zero. All non-significant correlations are zero to zero.

`cov` A $p \times p$ covariance correlation matrix based on posteriori median or mean estimators of the latent variables and coefficients.

`trace` The median/mean point estimator of the trace (sum of the diagonal elements) of the residual covariance matrix.

Note

Residual correlation matrices are reliably modeled only with two or more latent variables i.e., `num.lv > 1` when fitting the model using `boral`.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

References

- Pollock et al. (2014). Understanding co-occurrence by modelling species simultaneously with a Joint Species Distribution Model (JSDM). Methods in Ecology and Evolution, 5, 397-406.

See Also

get.enviro.cor, which calculates the correlation matrix due to similarities in the response to the explanatory variables (i.e., similarities due to a shared environmental response).

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
library(corrplot) ## For plotting correlations
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

spider.fit.nb <- boral(y, X = spider$x, family = "negative.binomial",
num.lv = 2, save.model = TRUE)

res.cors <- get.residual.cor(spider.fit.nb)

corrplot(res.cors$sig.cor, title = "Residual correlations",
type = "lower", diag = FALSE, mar = c(3,0.5,2,1), tl.srt = 45)

## End(Not run)
```

`lvsplot` *Plot the latent variables from an boral model*

Description

Construct a 1-D index plot or 2-D scatterplot of the latent variables, and their corresponding coefficients (i.e. a biplot), from a fitted boral model.

Usage

```
lvsplot(x, jitter = FALSE, a = 1, biplot = TRUE, ind.spp = NULL,
alpha = 0.5, main = NULL, est = "median",...)
```

Arguments

`x` An object for class "boral".

`jitter` If `jitter = TRUE`, then some jittering is applied so that points on the plots do not overlap exactly (which can often occur with discrete data, small sample sizes, and if some sites are identical in terms species co-occurence). Please see `jitter` for its implementation.

`a` Default parameter used in cex. Graphical options are adjusted as `par(cex = a, cex.axis = a, cex.lab = a+0.5, cex.main = a+0.5, ...)`. Defaults to 1.

biplot — If biplot = TRUE, then a biplot is construct such that both the latent variables *and* their corresponding coefficients are plotted. Otherwise, only the latent variable scores are plotted. Defaults to TRUE.

ind.spp — Controls the number of latent variable coefficients to plot if biplot = TRUE. If ind.spp is an integer, then only the first ind.spp "most important" latent variable coefficients are included in the biplot, where "most important" means the latent variable coefficients with the largests L2-norms. Defaults to NULL, in which case all latent variable coefficients are included in the biplot.

alpha — A numeric scalar between 0 and 1 that is used to control the relative scaling of the latent variables and their coefficients, when constructing a biplot. Defaults to 0.5, and we typically recommend between 0.45 to 0.55 so that the latent variables and their coefficients are on roughly the same scale.

main — Title for resulting ordination plot. Defaults to NULL, in which case a "standard" title is used.

est — A choice of either the posterior median (est = "median") or posterior mean (est = "mean"), which are then treated as estimates and the ordinations based off. Default is posterior median.

... — Additional graphical options to be included in par.

Details

This function allows an ordination plot to be constructed, based on either the posterior medians and posterior means of the latent variables respectively depending on the choice of est. The latent variables are labeled using the row index of the response matrix y.

If the fitted model did not contain any covariates, the ordination plot can be interpreted in the exactly same manner as unconstrained ordination plots constructed from methods such as Nonmetric Multidimensional Scaling (NMDS, Kruskal, 1964) and Correspondence Analysis (CA, Hill, 1974). With multivariate abundance data for instance, where the response matrix y consists of n sites and p species, the ordination plots can be studied to look for possible clustering of sites, location and/or dispersion effects, an arch pattern indicative of some sort species succession over an environmental gradient, and so on.

If the fitted model did include covariates, then a “residual ordination" plot is produced, which can be interpreted can offering a graphical representation of the (main patterns of) residual covarations, i.e. covariations after accounting for the covariates. With multivariate abundance data for instance, these residual ordination plots represent could represent residual species co-occurrence due to phylogency, species competition and facilitation, missing covariates, and so on (Warton et al., 2015)

If biplot = TRUE, then a biplot is constructed so that both the latent variables and their corresponding coefficients are included in their plot (Gabriel, 1971). The latent variable coefficients are shown in red, and are indexed by the column names of y. The number of latent variable coefficients to plot is controlled by ind.spp. In ecology for example, often we are only be interested in the "indicator" species, e.g. the species with most represent a particular set of sites or species with the strongest covariation (see Chapter 9, Legendre and Legendre, 2012, for additional discussion). In such case, we can then biplot only the ind.spp "most important" species, as indicated by the the L2-norm of their latent variable coefficients.

As with correspondence analysis, the relative scaling of the latent variables and the coefficients in a biplot is essentially arbitrary, and could be adjusted to focus on the sites, species, or put even weight on both (see Section 9.4, Legendre and Legendre, 2012). In `lvsplot`, this relative scaling is controlled by the `alpha` argument, which basically works by taking the latent variables to a power `alpha` and the latent variable coefficients to a power `1-alpha`.

For latent variable models, we are generally interested in "symmetric plots" that place the latent variables and their coefficients on the same scale. In principle, this is achieved by setting `alpha = 0.5`, the default value, although sometimes this needs to be tweaked slighlty to a value between 0.45 and 0.55 (see also the `corresp` function in the `MASS` package that also produces symmetric plots, as well as Section 5.4, Borcard et al., 2011 for more details on scaling).

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

References

- Borcard et al. (2011). Numerical Ecology with R. Springer.
- Gabriel, K. R. (1971). The biplot graphic display of matrices with application to principal component analysis. Biometrika, 58, 453-467.
- Hill, M. O. (1974). Correspondence analysis: a neglected multivariate method. Applied statistics, 23, 340-354.
- Kruskal, J. B. (1964). Nonmetric multidimensional scaling: a numerical method. Psychometrika, 29, 115-129.
- Legendre, P. and Legendre, L. (2012). Numerical ecology, Volume 20. Elsevier.
- Warton et al. (2015). So Many Variables: Joint Modeling in Community Ecology. Trends in Ecology and Evolution, in review.

Examples

```
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     n.burnin = 10, n.iteration = 100, n.thin = 1,
row.eff = "fixed", calc.ics = FALSE)

lvsplot(spider.fit.nb)
```

make.jagsboralmodel *Write a text file containing an boral model for use into JAGS*

Description

This function is designed to write boral models with one or more latent variables.

Usage

```
make.jagsboralmodel(family, num.X = 0, num.traits = 0,
   which.traits = NULL, row.eff = "none", trial.size = 1, n, p,
   hypparams = c(100,20,100,50), ssvs.index = -1, model.name = NULL)
```

Arguments

`family` Either a single element, or a vector of length equal to the number of columns in `y`. The former assumes all columns of `y` come from this distribution. The latter option allows for different distributions for each column of `y`. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`num.X` Number of columns in the model matrix `X`. Defaults to 0, in which case it is assumed that no covariates are included in the model.

`num.traits` Number of columns in the model matrix `traits`. Defaults to 0, in which case it is assumed no traits are included in model.

`which.traits` A list of length equal to (number of columns in `X` + 1), informing which columns of `traits` the column-specific intercepts and each of the column-specific regression coefficients should be regressed against. The first element in the list applies to the column-specific intercept, while the remaining elements apply to the regression coefficients. Each element of `which.traits` is a vector indicating which traits are to be used. For example, if `which.traits[[2]] = c(2,3)`, then the regression coefficients corresponding to the first column in `X` are regressed against the second and third columns of `traits`. If `which.traits[[2]] = 0`, then the regression coefficients are treated as independent.

Defaults to `NULL`, in conjunction with `num.traits = 0`).

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and unknown standard deviation. Defaults to "none".

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`n` The number of rows in the response matrix `y`.

`p` The number of columns in the response matrix `y`.

`hypparams` Vector of four hyperparameters used in the set up of prior distributions. The first element is the variance for the normal priors of all column-specific intercepts, row effects, and cutoff points for ordinal data. It also controls the maximum of the uniform prior for the standard deviation of the random effects normal distribution, if `row.eff = "random"`. The second element is the variance for the normal priors of all latent variable coefficients (ignored if `num.lv = 0`). The third element is the variance for the normal priors of all column-specific coefficients relating to the model matrix `X` (ignored if `X = NULL`). When traits are included in the model, it also controls the maximum of the uniform prior for the standard deviation of the normally distributed random effects (please see section on *Including species traits* below). The fourth element controls the maximum of the uniform prior used for dispersion parameters, ϕ. Note the common power parameter in the tweedie distribution is assumed to have uniform prior from 1 to 2. Note that if all columns of `y` are assumed to be ordinal responses, a sum-to-zero constraint is imposed on β_{0j} for model identifiability.

`ssvs.index` Indices to be used for Stochastic Search Variable Selection (SSVS, George and McCulloch, 1993). Either a single element or a vector with length equal to the number of columns in the implied model matrix `X`. Each element can take values of -1 (no SSVS is performed on this covariate), 0 (SSVS is performed on individual coefficients for this covariate), or any integer exceeding 1 (SSVS is performed on collectively all coefficients on this covariate/s.) Defaults to -1, in which case no model selection is performed on the fitted model at all.

This argument is only read if `X.eff = TRUE`, and is necessary to establish the prior distributions used for any explanatory variables. Please see the `boral` help file for more information regarding the implementation of SSVS.

`model.name` Name of the text file that the JAGS model is written to. Defaults to NULL, in which case the default of "jagsboralmodel.txt" is used.

Details

This function is automatically executed inside `boral`, and therefore does not need to be run separately before fitting the boral model. It can however be run independently if one is: 1) interested in what the actual JAGS file for a particular boral model looks like, 2) wanting to modify a basic JAGS model file to construct more complex model e.g., include environmental variables.

Please note that `boral` currently does not allow the user to manually enter a script to be run.

When running the main function `boral`, setting `save.model = TRUE` which automatically save the JAGS model file as a text file (with name based on the `model.name`) in the current working directory.

Value

A text file is created, containing the JAGS model to be called by the boral function for entering into jags. This file is automatically deleted once boral has finished running `save.model = TRUE`.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

`make.jagsboralnullmodel` for writing boral models JAGS scripts with no latent variables (so-called "null models").

Examples

```
## Not run:
library(mvtnorm)
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);

## Example 1 - Create a boral model JAGS script, where distributions alternative
## between Poisson and negative binomial distributions
##   across the rows of y...why not?
make.jagsboralmodel(family = rep(c("poisson","negative.binomial"),length=p),
row.eff = "fixed", num.X = 0, n = n, p = p)

## Example 2 - Create a boral model JAGS script, where distributions are all
## negative binomial distributions and covariates will be included.
make.jagsboralmodel(family = "negative.binomial", num.X = ncol(spider$x),
n = n, p = p)


## Example 3 - Simulate some ordinal data and create a JAGS model script
## 30 rows (sites) with two latent variables
true.lv <- rbind(rmvnorm(15,mean=c(-2,-2)),rmvnorm(15,mean=c(2,2)))
## 10 columns (species)
true.lv.coefs <- rmvnorm(10,mean = rep(0,3));
true.lv.coefs[nrow(true.lv.coefs),1] <- -sum(true.lv.coefs[-nrow(true.lv.coefs),1])
## Impose a sum-to-zero constraint on the column effects
true.ordinal.cutoffs <- seq(-2,10,length=10-1)

sim.y <- create.life(true.lv = true.lv, lv.coefs = true.lv.coefs,
family = "ordinal", cutoffs = true.ordinal.cutoffs)

make.jagsboralmodel(family = "ordinal", num.X = 0, row.eff = FALSE, n=30, p=10,
model.name = "myawesomeordmodel.txt")


## Have a look at the JAGS model file for a boral model involving traits,
## based on the ants data from mvabund.
```

```
library(mvabund)
data(antTraits)

y <- antTraits$abun
X <- as.matrix(antTraits$env)
## Include only traits 1, 2, and 5, plus an intercept
traits <- as.matrix(cbind(1,antTraits$traits[,c(1,2,5)]))
## Please see help file for boral regarding the use of which.traits
which.traits <- vector("list",ncol(X)+1)
for(i in 1:length(which.traits)) which.traits[[i]] <- 1:ncol(traits)

fit.traits <- boral(y, X = X, traits = traits, which.traits = which.traits,
family = "negative.binomial", num.lv = 2, do.fit = FALSE,
model.name = "anttraits.txt")


## End(Not run)
```

make.jagsboralnullmodel

Write a text file containing an boral model for use into JAGS

Description

This function is designed to write boral models with no latent variables (so-called "null" models).

Usage

```
make.jagsboralnullmodel(family, num.X = 0, num.traits = 0,
   which.traits = NULL, row.eff = "none", trial.size = 1, n, p,
   hypparams = c(100,20,100,50), ssvs.index = -1, model.name = NULL)
```

Arguments

`family` Either a single element, or a vector of length equal to the number of columns in y. The former assumes all columns of y come from this distribution. The latter option allows for different distributions for each column of y. Elements can be one of "binomial" (with probit link), "poisson" (with log link), "negative.binomial" (with log link), "normal" (with identity link), "lnormal" for lognormal (with log link), "tweedie" (with log link), "exponential" (with log link), "gamma" (with log link), "beta" (with logit link), "ordinal" (cumulative probit regression).

For the negative binomial distribution, the variance is parameterized as $Var(y) = \mu + \phi\mu^2$, where ϕ is the column-specific dispersion parameter. For the normal distribution, the variance is parameterized as $Var(y) = \phi^2$, where ϕ is the column-specific standard deviation. For the tweedie distribution, the variance is parameterized as $Var(y) = \phi\mu^p$ where ϕ is the column-specific dispersion parameter and p is a power parameter common to all columns assumed to be tweedie, with $1 < p < 2$. For the gamma distribution, the variance is parameterized as $Var(y) = \mu/\phi$ where ϕ is the column-specific rate (henceforth referred to also as dispersion parameter). For the beta distribution, the parameterization is in terms of the mean μ and sample size ϕ (henceforth referred to also as dispersion parameter), so that the two shape parameters are given by $a = \mu\phi$ and $b = (1 - \mu)\phi$.

All columns assumed to have ordinal responses are constrained to have the same cutoffs points, with a column-specific intercept to account for differences between the columns (please see *Details* for formulation).

`num.X` Number of columns in the model matrix `X`. Defaults to 0, in which case it is assumed that no covariates are included in the model.

`num.traits` Number of columns in the model matrix `traits`. Defaults to 0, in which case it is assumed no traits are included in model.

`which.traits` A list of length equal to (number of columns in `X` + 1), informing which columns of `traits` the column-specific intercepts and each of the column-specific regression coefficients should be regressed against. The first element in the list applies to the column-specific intercept, while the remaining elements apply to the regression coefficients. Each element of `which.traits` is a vector indicating which traits are to be used. For example, if `which.traits[[2]] = c(2,3)`, then the regression coefficients corresponding to the first column in `X` are regressed against the second and third columns of `traits`. If `which.traits[[2]] = 0`, then the regression coefficients are treated as independent.

Defaults to `NULL`, in conjunction with `num.traits = 0`).

`row.eff` Single element indicating whether row effects are included as fixed effects ("fixed"), random effects ("random") or not included ("none") in the boral model. If random effects, they are drawn from a normal distribution with mean zero and unknown standard deviation. Defaults to "none".

`trial.size` Either equal to a single element, or a vector of length equal to the number of columns in y. If a single element, then all columns assumed to be binomially distributed will have trial size set to this. If a vector, different trial sizes are allowed in each column of y. The argument is ignored for all columns not assumed to be binomially distributed. Defaults to 1, i.e. Bernoulli distribution.

`n` The number of rows in the response matrix `y`.

`p` The number of columns in the response matrix `y`.

`hypparams` Vector of four hyperparameters used in the set up of prior distributions. The first element is the variance for the normal priors of all column-specific intercepts, row effects, and cutoff points for ordinal data. It also controls the maximum of the uniform prior for the standard deviation of the random effects normal distribution, if `row.eff = "random"`. The second element is the variance for the normal priors of all latent variable coefficients (ignored if `num.lv = 0`). The third element is the variance for the normal priors of all column-specific coefficients relating to the model matrix `X` (ignored if `X = NULL`). When traits are included in the model, it also controls the maximum of the uniform prior for the standard deviation of the normally distributed random effects (please see section on *Including species traits* below). The fourth element controls the maximum of the uniform prior used for dispersion parameters, ϕ. Note the common power parameter in the tweedie distribution is assumed to have uniform prior from 1 to 2. Note that if all columns of y are assumed to be ordinal responses, a sum-to-zero constraint is imposed on β_{0j} for model identifiability.

ssvs.index Indices to be used for Stochastic Search Variable Selection (SSVS, George and McCulloch, 1993). Either a single element or a vector with length equal to the number of columns in the implied model matrix X. Each element can take values of -1 (no SSVS is performed on this covariate), 0 (SSVS is performed on individual coefficients for this covariate), or any integer exceeding 1 (SSVS is performed on collectively all coefficients on this covariate/s.) Defaults to -1, in which case no model selection is performed on the fitted model at all.

This argument is only read if X.eff = TRUE, and is necessary to establish the prior distributions used for any explanatory variables. Please see the boral help file for more information regarding the implementation of SSVS.

model.name Name of the text file that the JAGS model is written to. Defaults to NULL, in which case the default of "jagsboralmodel.txt" is used.

Details

This function is automatically executed inside boral, and therefore does not need to be run separately before fitting the boral model. It can however be run independently if one is: 1) interested in what the actual JAGS file for a particular boral model looks like, 2) wanting to modify a basic JAGS model file to construct more complex model e.g., include environmental variables.

Please note that boral currently does not allow the user to manually enter a script to be run.

When running the main function boral, setting save.model = TRUE which automatically save the JAGS model file as a text file (with name based on the model.name) in the current working directory.

Value

A text file is created, containing the JAGS model to be called by the boral function for entering into jags. This file is automatically deleted once boral has finished running unless save.model = TRUE.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

make.jagsboralmodel for writing boral model JAGS scripts with one or more latent variables.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun
n <- nrow(y); p <- ncol(y);
```

```
## Create a boral "null" model JAGS script, where distributions alternative
## between Poisson and negative distributions
##   across the rows of y...why not?
make.jagsboralnullmodel(family = rep(c("poisson","negative.binomial"),length=p),
     num.X = ncol(spider$x), row.eff = "fixed", n = n, p = p)


## Create a boral "null" model JAGS script, where distributions are all negative
##  binomial distributions and covariates will be included!
make.jagsboralnullmodel(family = rep("negative.binomial",length=p),
     num.X = ncol(spider$x), n = n, p = p, model.name = "myawesomeordnullmodel.txt")


## Have a look at the JAGS model file for a boral model involving traits,
## based on the ants data from mvabund.
library(mvabund)
data(antTraits)

y <- antTraits$abun
X <- as.matrix(antTraits$env)
## Include only traits 1, 2, and 5, plus an intercept
traits <- as.matrix(cbind(1,antTraits$traits[,c(1,2,5)]))
## Please see help file for boral regarding the use of which.traits
which.traits <- vector("list",ncol(X)+1)
for(i in 1:length(which.traits)) which.traits[[i]] <- 1:ncol(traits)

fit.traits <- boral(y, X = X, traits = traits, which.traits = which.traits,
family = "negative.binomial", num.lv = 0, do.fit = FALSE,
model.name = "anttraits.txt")

## End(Not run)
```

---

plot.boral *Plots of a fitted boral object*

Description

Produces four plots relating to the fitted boral object, which can be used for residual analysis.

Usage

```
## S3 method for class 'boral'
plot(x, est = "median", jitter = FALSE, a = 1,...)
```

Arguments

x	An object of class "boral".

`est`	A choice of either the posterior median (`est == "median"`) or posterior mean (`est == "mean"`) of the parameters, which are then treated as parameter estimates and the fitted values/residuals used in the plots are calculated from. Default is posterior median.
`jitter`	If `jitter = TRUE`, then some jittering is applied so that points on the plots do not overlap exactly (which can often occur with discrete data). Please see `jitter` for its implementation.
`a`	Default parameter used in cex. Graphical options are then adjusted as `par(ask = T, cex = a, cex.main = a, ...)`. Defaults to 1.
`...`	Additional graphical options to be included in `par`.

Details

Four types of plots are provided:

1. Plot of Dunn-Smyth residuals against the linear predictors. This can be useful to assess whether the assumed mean-variance relationship is adequately satisfied, as well as to look for particular outliers.
2. Plot of Dunn-Smyth residuals against the row index/row names.
3. Plot of Dunn-Smyth residuals against the column index/column names. Both this and the previous plot are useful for assessing how well each row/column of the response matrix is being modeled.
4. A normal quantile plot of the Dunn-Smyth residuals, which can be used to assess the normality assumption and overall goodness of fit.

Note

If all the columns of `y` were assumed to be ordinal, then this function is immediately stopped, as not residuals can be plotted in this case.

Due the inherent stochasticity, Dunn-Smyth residuals and consequently the plots will be slightly different time this function is run. Note also the fitted values and residuals are calculated from point estimates of the parameters, as opposed to a fully Bayesian approach (please see details in `fitted.boral` and `ds.residuals`). Consequently, it is recommended that this function is run several times to ensure that any trends observed in the plots are consistent throughout the runs.

Author(s)

Francis K.C. Hui <`fhui28@gmail.com`>

See Also

`fitted.boral` to obtain the fitted values, `ds.residuals` to obtain Dunn-Smyth residuals and details as to what they are.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun

spider.fit.p <- boral(y, family = "poisson", num.lv = 2,
row.eff = "fixed")

plot(spider.fit.p, ask = FALSE, mfrow = c(2,2))
## A distinct fan pattern is observed in the plot of residuals
## versus linear predictors plot.


spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
     row.eff = "fixed")

plot(spider.fit.nb, ask = FALSE, mfrow = c(2,2))
## The fan shape is not as clear now,
## and the normal quantile plot also suggests a better fit to the data

## End(Not run)
```

summary.boral *Summary of fitted boral object*

Description

A summary of the fitted boral objects including the type of model fitted e.g., error distribution, number of latent variables parameter estimates, values of the information criteria (if applicable), and so on.

Usage

```
## S3 method for class 'boral'
summary(object, est = "median", ...)

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

Arguments

`object`	An object of class "boral".
`x`	An object of class "boral".
`est`	A choice of either whether to print the posterior median (`est == "median"`) or posterior mean (`est == "mean"`) of the parameters.
`...`	Not used.

Value

Attributes of the model fitted, parameter estimates, and values of the information criteria if `calc.ics = TRUE` in the boral object, and posterior probabilities of including individual and/or grouped coefficients in the model based on SSVS if appropriate.

Author(s)

Francis K.C. Hui <fhui28@gmail.com>

See Also

`boral` for the fitting function on which `summary` is applied, `get.measures` for details regarding the information criteria returned.

Examples

```
## Not run:
library(mvabund) ## Load a dataset from the mvabund package
data(spider)
y <- spider$abun

spider.fit.nb <- boral(y, family = "negative.binomial", num.lv = 2,
row.eff = "fixed")

summary(spider.fit.nb)

## End(Not run)
```

Index