Package 'jSDM'

<div align="center">March 21, 2022</div>

Type Package

Title Joint Species Distribution Models

Version 0.2.1

Date 2022-03-16

Imports Rcpp (>= 1.0.0), graphics, stats, coda, corrplot, stringi,
MASS, parallel, doParallel, foreach

LinkingTo Rcpp, RcppArmadillo, RcppGSL

NeedsCompilation yes

SystemRequirements GNU GSL

Suggests knitr, kableExtra, raster, sp, rgdal, dplyr, rmarkdown,
bookdown, testthat, boral, Hmsc, BayesComm, snow, snowfall,
ggplot2

Maintainer Jeanne Clément <jeanne.clement16@laposte.net>

Description Fits joint species distribution models ('jSDM')
in a hierarchical Bayesian framework (Warton et al. 2015
<doi:10.1016/j.tree.2015.09.007>). The Gibbs sampler is written
in C++. It uses 'Rcpp', 'Armadillo' and 'GSL' to maximize computation
efficiency.

Depends R (>= 3.5.0)

License GPL-3 | file LICENSE

URL https://ecology.ghislainv.fr/jSDM/,
https://github.com/ghislainv/jSDM

BugReports https://github.com/ghislainv/jSDM/issues

LazyLoad yes

Encoding UTF-8

RoxygenNote 7.1.2

VignetteBuilder knitr

Author Jeanne Clément [aut, cre] (<<https://orcid.org/0000-0002-5228-5015>>),
 Ghislain Vieilledent [aut] (<<https://orcid.org/0000-0002-1685-4997>>),
 Frédéric Gosselin [ctb] (<<https://orcid.org/0000-0003-3737-106X>>),
 CIRAD [cph, fnd]

R topics documented:

---

jSDM-package *joint species distribution models*

Description

jSDM is an R package for fitting joint species distribution models (JSDM) in a hierarchical Bayesian framework.

The Gibbs sampler is written in C++. It uses Rcpp, Armadillo and GSL to maximize computation efficiency.

Package:	jSDM
Type:	Package
Version:	0.1.0
Date:	2019-01-11

License:	GPL-3
LazyLoad:	yes

Details

The package includes the following functions to fit various species distribution models :

function	data-type
jSDM_binomial_logit	presence-absence
jSDM_binomial_probit	presence-absence
jSDM_binomial_probit_sp_constrained	presence-absence
jSDM_binomial_probit_long_format	presence-absence
jSDM_poisson_log	abundance

- jSDM_binomial_probit :

  Ecological process:
  $$y_{ij} \sim \mathcal{B}ernoulli(\theta_{ij})$$

  where

if n_latent=0 and site_effect="none"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="fixed"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="random"	$\mathrm{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

- jSDM_binomial_probit_sp_constrained :

  This function allows to fit the same models than the function jSDM_binomial_probit except for models not including latent variables, indeed n_latent must be greater than zero in this function. At first, the function fit a JSDM with the constrained species arbitrarily chosen as the first ones in the presence-absence data-set. Then, the function evaluates the convergence of MCMC λ chains using the Gelman-Rubin convergence diagnostic ($\hat{R}$). It identifies the species ($\hat{j}_l$) having the higher $\hat{R}$ for $\lambda_{\hat{j}_l}$. These species drive the structure of the latent axis l. The λ corresponding to this species are constrained to be positive and placed on the diagonal of the Λ matrix for fitting a second model. This usually improves the convergence of the latent variables and factor loadings. The function returns the parameter posterior distributions for this second model.

- jSDM_binomial_logit :

  Ecological process :
  $$y_{ij} \sim \mathcal{B}inomial(\theta_{ij}, t_i)$$

  where

if n_latent=0 and site_effect="none"	$\text{logit}(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\text{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\text{logit}(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="fixed"	$\text{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\text{logit}(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="random"	$\text{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

- jSDM_poisson_log :

  Ecological process :
  $$y_{ij} \sim \mathcal{P}oisson(\theta_{ij})$$
  where

if n_latent=0 and site_effect="none"	$\log(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\log(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="fixed"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\log(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="random"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

- jSDM_binomial_probit_long_format :

  Ecological process:
  $$y_n \sim \mathcal{B}ernoulli(\theta_n)$$
  such as $species_n = j$ and $site_n = i$, where

if n_latent=0 and site_effect="none"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j$
if n_latent>0 and site_effect="none"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="fixed"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + \alpha_i$
if n_latent>0 and site_effect="random"	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

Frédéric Gosselin <frederic.gosselin@inrae.fr>

References

Chib, S. and Greenberg, E. (1998) Analysis of multivariate probit models. *Biometrika*, 85, 347-361.

Warton, D. I.; Blanchet, F. G.; O'Hara, R. B.; O'Hara, R. B.; Ovaskainen, O.; Taskinen, S.; Walker,

S. C. and Hui, F. K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution*, 30, 766-779.

Ovaskainen, O., Tikhonov, G., Norberg, A., Blanchet, F. G., Duan, L., Dunson, D., Roslin, T. and Abrego, N. (2017) How to make more out of community data? A conceptual framework and its implementation as models and software. *Ecology Letters*, 20, 561-576.

---

aravo	*Distribution of Alpine plants in Aravo (Valloire, France)*

Description

This dataset describe the distribution of 82 species of Alpine plants in 75 sites. Species traits and environmental variables are also measured.

Usage

```
data("aravo")
```

Format

aravo is a list containing the following objects :

spe   is a data.frame with the abundance values of 82 species (columns) in 75 sites (rows).

env   is a data.frame with the measurements of 6 environmental variables for the sites.

traits   is data.frame with the measurements of 8 traits for the species.

spe.names   is a vector with full species names.

Details

The environmental variables are :

Aspect	Relative south aspect (opposite of the sine of aspect with flat coded 0)
Slope	Slope inclination (degrees)
Form	Microtopographic landform index: 1 (convexity); 2 (convex slope); 3 (right slope); 4 (concave slope); 5 (concavity)
Snow	Mean snowmelt date (Julian day) averaged over 1997-1999
PhysD	Physical disturbance, i.e., percentage of unvegetated soil due to physical processes
ZoogD	Zoogenic disturbance, i.e., quantity of unvegetated soil due to marmot activity: no; some; high

The species traits for the plants are:

Height	Vegetative height (cm)
Spread	Maximum lateral spread of clonal plants (cm)
Angle	Leaf elevation angle estimated at the middle of the lamina
Area	Area of a single leaf
Thick	Maximum thickness of a leaf cross section (avoiding the midrib)

SLA	Specific leaf area
Nmass	Mass-based leaf nitrogen content
Seed	Seed mass

Source

Choler, P. (2005) Consistent shifts in Alpine plant traits along a mesotopographical gradient. Arctic, Antarctic, and Alpine Research 37,444-453.

Dray S, Dufour A (2007). The ade4 Package: Implementing the Duality Diagram for Ecologists. *Journal of Statistical Software*, 22(4), 1-20. doi: 10.18637/jss.v022.i04.

Examples

```
data(aravo, package="jSDM")
summary(aravo)
```

---

birds	*birds dataset*

Description

The Swiss breeding bird survey ("Monitoring Häufige Brutvögel" MHB) has monitored the populations of 158 common species since 1999.

The MHB sample from data(MHB2014,package="AHMbook") consists of 267 1-km squares that are laid out as a grid across Switzerland. Fieldwork is conducted by about 200 skilled birdwatchers, most of them volunteers. Avian populations are monitored using a simplified territory mapping protocol, where each square is surveyed up to three times during the breeding season (only twice above the tree line).

Surveys are conducted along a transect that does not change over the years. The birds dataset has the data for 2014, except one quadrat not surveyed in 2014.

Usage

```
data("birds")
```

Format

A data frame with 266 observations on the following 166 variables.

158 bird species named in latin and whose occurrences are indicated as the number of visits to each site during which the species was observed, including 13 species not recorded in the year 2014 and **5 covariates** collected on the 266 1x1 km quadrat as well as their identifiers and coordinates :

siteID   an alphanumeric site identifier

coordx   a numeric vector indicating the x coordinate of the centre of the quadrat.
           The coordinate reference system is not specified intentionally.

coordy	a numeric vector indicating the y coordinate of the centre of the quadrat.
elev	a numeric vector indicating the mean elevation of the quadrat (m).
rlength	the length of the route walked in the quadrat (km).
nsurvey	a numeric vector indicating the number of replicate surveys planned in the quadrat; above the tree-line 2, otherwise 3.
forest	a numeric vector indicating the percentage of forest cover in the quadrat.
obs14	a categorical vector indicating the identifying number of the observer.

Details

Only the Latin names of bird species are given in this dataset but you can find the corresponding English names in the original dataset : `data(MHB2014,package="AHMbook")`.

Source

Swiss Ornithological Institute

References

Kéry and Royle (2016) Applied Hierarachical Modeling in Ecology Section 11.3

Examples

```
data(birds, package="jSDM")
head(birds)
# find species not recorded in 2014
which(colSums(birds[,1:158])==0)
```

---

eucalypts	*eucalypts dataset*

Description

The Eucalyptus data set includes 12 taxa recorded in 458 plots spanning elevation gradients in the Grampians National Park, Victoria, which is known for high species diversity and endemism. The park has three mountain ranges interspersed with alluvial valleys and sand sheet and has a semi-Mediterranean climate with warm, dry summers and cool, wet winters.

This dataset records presence or absence at 458 sites of 12 eucalypts species, 7 covariates collected at these sites as well as their longitude and latitude.

Usage

```
data("eucalypts")
```

Format

A data frame with 458 observations on the following 21 variables.

12 eucalypts species which presence on sites is indicated by a 1 and absence by a 0 :

ALA a binary vector indicating the occurrence of the species E. alaticaulis

ARE a binary vector indicating the occurrence of the species E. arenacea

BAX a binary vector indicating the occurrence of the species E. baxteri

CAM a binary vector indicating the occurrence of the species E. camaldulensis

GON a binary vector indicating the occurrence of the species E. goniocalyx

MEL a binary vector indicating the occurrence of the species E. melliodora

OBL a binary vector indicating the occurrence of the species E. oblique

OVA a binary vector indicating the occurrence of the species E. ovata

WIL a binary vector indicating the occurrence of the species E. willisii subsp. Falciformis

ALP a binary vector indicating the occurrence of the species E. serraensis, E. verrucata and E. victoriana

VIM a binary vector indicating the occurrence of the species E. viminalis subsp. Viminalis and Cygnetensis

ARO.SAB a binary vector indicating the occurrence of the species E. aromaphloia and E. sabulosa

7 covariates collected on the 458 sites and their coordinates :

Rockiness a numeric vector taking values from 0 to 95 corresponding to the rock cover of the site in percent estimated in 5 % increments in field plots

Sandiness a binary vector indicating if soil texture categorie is sandiness based on soil texture classes from field plots and according to relative amounts of sand, silt, and clay particles

VallyBotFlat a numeric vector taking values from 0 to 6 corresponding to the valley bottom flatness GIS-derived variable defining flat areas relative to surroundings likely to accumulate sediment (units correspond to the percentage of slope e.g. 0.5 = 16 %slope, 4.5 = 1 %slope, 5.5 = 0.5 %slope)

PPTann a numeric vector taking values from 555 to 1348 corresponding to annual precipitation in millimeters measured as the sum of monthly precipitation estimated using BIOCLIM based on 20m grid cell Digital Elevation Model

Loaminess a binary vector indicating if soil texture categorie is loaminess based on soil texture classes from field plots and according to relative amounts of sand, silt, and clay particles

cvTemp a numeric vector taking values from 136 to 158 corresponding to coefficient of variation of temperature seasonality in percent measured as the standard deviation of weekly mean temperatures as a percentage of the annual mean temperature from BIOCLIM

T0 a numeric vector corresponding to solar radiation in WH/m^2 measured as the amount of incident solar energy based on the visible sky and the sun's position. Derived from Digital Elevation Model in ArcGIS 9.2 Spatial Analyst for the summer solstice (December 22)

latitude a numeric vector indicating the latitude of the studied site

longitude a numeric vector indicating the longitude of the studied site

Source

Wilkinson, D. P.; Golding, N.; Guillera-Arroita, G.; Tingley, R. and McCarthy, M. A. (2018) A comparison of joint species distribution models for presence-absence data. *Methods in Ecology and Evolution*.

Examples

```
data(eucalypts, package="jSDM")
head(eucalypts)
```

---

frogs	*frogs dataset*

Description

Presence or absence of 9 species of frogs at 104 sites, 3 covariates collected at these sites and their coordinates.

Format

`frogs` is a data frame with 104 observations on the following 14 variables.

`Species_` 1 to 9 indicate by a 0 the absence of the species on one site and by a 1 its presence

`Covariates_` 1 and 3 continuous variables

`Covariates_` 2 discrete variables

`x` a numeric vector of first coordinates corresponding to each site

`y` a numeric vector of second coordinates corresponding to each site

Source

Wilkinson, D. P.; Golding, N.; Guillera-Arroita, G.; Tingley, R. and McCarthy, M. A. (2018) A comparison of joint species distribution models for presence-absence data. *Methods in Ecology and Evolution*.

Examples

```
data(frogs, package="jSDM")
head(frogs)
```

---

fungi	*fungi dataset*

Description

Presence or absence of 11 species of fungi on dead-wood objects at 800 sites and 12 covariates collected at these sites.

Usage

```
data("fungi")
```

Format

A data frame with 800 observations on the following 23 variables :

11 fungi species which presence on sites is indicated by a 1 and absence by a 0 :

antser a binary vector

antsin a binary vector

astfer a binary vector

fompin a binary vector

hetpar a binary vector

junlut a binary vector

phefer a binary vector

phenig a binary vector

phevit a binary vector

poscae a binary vector

triabi a binary vector

12 covariates collected on the 800 sites :

diam a numeric vector indicating the diameter of dead-wood object

dc1 a binary vector indicating if the decay class is 1 measured in the scale 1, 2, 3, 4, 5 (from freshly decayed to almost completely decayed)

dc2 a binary vector indicating if the decay class is 2

dc3 a binary vector indicating if the decay class is 3

dc4 a binary vector indicating if the decay class is 4

dc5 a binary vector indicating if the decay class is 5

quality3 a binary vector indicating if the quality is level 3

quality4 a binary vector indicating if the quality is level 4

ground3 a binary vector indicating if the ground contact is level 3 as 2 = no ground contact, 3 = less than half of the log in ground contact and 4 = more than half of the log in ground contact

ground4 a binary vector a binary vector indicating if the ground contact is level 4

epi a numeric vector indicating the epiphyte cover

bark a numeric vector indicating the bark cover

Source

Wilkinson, D. P.; Golding, N.; Guillera-Arroita, G.; Tingley, R. and McCarthy, M. A. (2018) A comparison of joint species distribution models for presence-absence data. *Methods in Ecology and Evolution*.

Examples

```
data(fungi, package="jSDM")
head(fungi)
```

---

get_enviro_cor	*Extract covariances and correlations due to shared environmental responses*

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(mod, type = "mean", prob = 0.95)
```

Arguments

mod	An object of class "jSDM"
type	A choice of either the posterior median (type = "median") or posterior mean (type = "mean"), which are then treated as estimates and the fitted values are calculated from. Default is posterior mean.
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 each of the columns of the response matrix Y are fitted to a set of explanatory variables given by X, the covariance between two columns j and j', due to similarities in their response to the model matrix, is thus calculated based on the linear predictors $X\beta_j$ and $X\beta'_j$, where β_j are species effects relating to the explanatory variables. Such correlation matrices are discussed and found in Ovaskainen et al., (2010), Pollock et al., (2014).

Value

results A list including :

cor, cor.lower, cor.upper

A set of $np \times np$ correlation matrices, containing either the posterior median
or mean estimate over the MCMC samples plus lower and upper limits of the
corresponding 95 % highest posterior interval.

sig.cor   A $np \times np$ correlation matrix containing only the "significant" correlations
whose 95 % highest posterior density (HPD) interval does not contain zero. All
non-significant correlations are set to zero.

cov   Average over the MCMC samples of the $np \times np$ covariance matrix.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Hui FKC (2016). "boral: Bayesian Ordination and Regression Analysis of Multivariate Abundance
Data in R." *Methods in Ecology and Evolution*, 7, 744–750.

Ovaskainen et al. (2010). Modeling species co-occurrence by multivariate logistic regression gen-
erates 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

cov2cor get_residual_cor jSDM-package jSDM_binomial_probit
jSDM_binomial_logit jSDM_poisson_log

Examples

```
library(jSDM)
# frogs data
 data(frogs, package="jSDM")
 # Arranging data
 PA_frogs <- frogs[,4:12]
 # Normalized continuous variables
 Env_frogs <- cbind(scale(frogs[,1]),frogs[,2],
                    scale(frogs[,3]))
 colnames(Env_frogs) <- colnames(frogs[,1:3])
 Env_frogs <- as.data.frame(Env_frogs)
 # Parameter inference
# Increase the number of iterations to reach MCMC convergence
mod <- jSDM_binomial_probit(# Response variable
                            presence_data=PA_frogs,
                            # Explanatory variables
                            site_formula = ~.,
```

```
                            site_data = Env_frogs,
                            n_latent=0,
                            site_effect="random",
                            # Chains
                            burnin=100,
                            mcmc=100,
                            thin=1,
                            # Starting values
                            alpha_start=0,
                            beta_start=0,
                            V_alpha=1,
                            # Priors
                            shape=0.5, rate=0.0005,
                            mu_beta=0, V_beta=10,
                            # Various
                            seed=1234, verbose=1)
 # Calcul of residual correlation between species
  enviro.cors <- get_enviro_cor(mod)
```

---

get_residual_cor   *Calculate the residual correlation matrix from a latent variable model*
                          (LVM)

Description

This function use coefficients $(\lambda_{jl} with j = 1, \ldots, n_{species} and l = 1, \ldots, n_{latent})$, corresponding to latent variables fitted using jSDM package, to calculate the variance-covariance matrix which controls correlation between species.

Usage

```
get_residual_cor(mod, prob = 0.95)
```

Arguments

mod   An object of class "jSDM"

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

Details

After fitting the jSDM with latent variables, the **fullspecies residual correlation matrix** : $R = (R_{ij})$ with $i = 1, \ldots, n_{species}$ and $j = 1, \ldots, n_{species}$ can be derived from the covariance in the latent variables such as :

$$\begin{aligned}\Sigma_{ij} &= \lambda_i.\lambda_j' + 1 \quad \text{if i=j}\\ &= \lambda_i.\lambda_j' \qquad \text{else,}\end{aligned}$$

then we compute correlations from covariances :

$$R_{ij} = \frac{\Sigma_{ij}}{\sqrt{\Sigma_i i \Sigma_j j}}$$

.

Value

results A list including :

cov.mean	Average over the MCMC samples of the variance-covariance matrix.
cov.median	Median over the MCMC samples of the variance-covariance matrix.
cov.lower	A $n_{species} \times n_{species}$ matrix containing the lower limits of the $(100 \times prob\%)$ HPD interval of variance-covariance matrices over the MCMC samples.
cov.upper	A $n_{species} \times n_{species}$ matrix containing the upper limits of the $(100 \times prob\%)$ HPD interval of variance-covariance matrices over the MCMC samples.
cov.sig	A $n_{species} \times n_{species}$ matrix containing the value 1 corresponding to the "significant" co-variances and the value 0 corresponding to "non-significant" co-variances, whose $(100 \times prob\%)$ HPD interval over the MCMC samples contain zero.
cor.mean	Average over the MCMC samples of the residual correlation matrix.
cor.median	Median over the MCMC samples of the residual correlation matrix.
cor.lower	A $n_{species} \times n_{species}$ matrix containing the lower limits of the $(100 \times prob\%)$ HPD interval of correlation matrices over the MCMC samples.
cor.upper	A $n_{species} \times n_{species}$ matrix containing the upper limits of the $(100 \times prob\%)$ HPD interval of correlation matrices over the MCMC samples.
cor.sig	A $n_{species} \times n_{species}$ matrix containing the value 1 corresponding to the "significant" correlations and the value 0 corresponding to "non-significant" correlations, whose $(100 \times prob\%)$ HPD interval over the MCMC samples contain zero.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Hui FKC (2016). "boral: Bayesian Ordination and Regression Analysis of Multivariate Abundance Data in R." *Methods in Ecology and Evolution*, 7, 744–750.

Ovaskainen et al. (2016). Using latent variable models to identify large networks of species-to-species associations at different spatial scales. *Methods in Ecology and Evolution*, 7, 549-555.

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 cov2cor jSDM-package jSDM_binomial_probit
jSDM_binomial_logit jSDM_poisson_log

Examples

```
library(jSDM)
# frogs data
 data(frogs, package="jSDM")
 # Arranging data
 PA_frogs <- frogs[,4:12]
 # Normalized continuous variables
 Env_frogs <- cbind(scale(frogs[,1]),frogs[,2],scale(frogs[,3]))
 colnames(Env_frogs) <- colnames(frogs[,1:3])
 Env_frogs <- as.data.frame(Env_frogs)
 # Parameter inference
# Increase the number of iterations to reach MCMC convergence
 mod <- jSDM_binomial_probit(# Response variable
                             presence_data=PA_frogs,
                             # Explanatory variables
                             site_formula = ~.,
                             site_data = Env_frogs,
                             n_latent=2,
                             site_effect="random",
                             # Chains
                             burnin=100,
                             mcmc=100,
                             thin=1,
                             # Starting values
                             alpha_start=0,
                             beta_start=0,
                             lambda_start=0,
                             W_start=0,
                             V_alpha=1,
                             # Priors
                             shape=0.5, rate=0.0005,
                             mu_beta=0, V_beta=10,
                             mu_lambda=0, V_lambda=10,
                             # Various
                             seed=1234, verbose=1)
# Calcul of residual correlation between species
result <- get_residual_cor(mod, prob=0.95)
# Residual variance-covariance matrix
result$cov.mean
## All non-significant co-variances are set to zero.
result$cov.mean * result$cov.sig
# Residual correlation matrix
result$cor.mean
## All non-significant correlations are set to zero.
result$cor.mean * result$cor.sig
```

`inv_logit`   *Generalized inverse logit function*

Description

Compute generalized inverse logit function.

Usage

```
inv_logit(x, min = 0, max = 1)
```

Arguments

x	value(s) to be transformed
min	Lower end of logit interval
max	Upper end of logit interval

Details

The generalized inverse logit function takes values on [-Inf,Inf] and transforms them to span [min, max] :

$$y = p'(max - min) + min$$

$$where$$

$$p = \frac{exp(x)}{(1 + exp(x))}$$

Value

y Transformed value(s).

Author(s)

Gregory R. Warnes <greg@warnes.net>

Examples

```
x <- seq(0,10, by=0.25)
xt <- jSDM::logit(x, min=0, max=10)
cbind(x,xt)
y <- jSDM::inv_logit(xt, min=0, max=10)
cbind(x,xt,y)
```

```
jSDM_binomial_logit      Binomial logistic regression
```

Description

The `jSDM_binomial_logit` function performs a Binomial logistic regression in a Bayesian framework. The function calls a Gibbs sampler written in C++ code which uses an adaptive Metropolis algorithm to estimate the conditional posterior distribution of model's parameters.

Usage

```
jSDM_binomial_logit(
  burnin = 5000,
  mcmc = 10000,
  thin = 10,
  presence_data,
  site_formula,
  trait_data = NULL,
  trait_formula = NULL,
  site_data,
  trials = NULL,
  n_latent = 0,
  site_effect = "none",
  beta_start = 0,
  gamma_start = 0,
  lambda_start = 0,
  W_start = 0,
  alpha_start = 0,
  V_alpha = 1,
  shape = 0.5,
  rate = 5e-04,
  mu_beta = 0,
  V_beta = 10,
  mu_gamma = 0,
  V_gamma = 10,
  mu_lambda = 0,
  V_lambda = 10,
  ropt = 0.44,
  seed = 1234,
  verbose = 1
)
```

Arguments

burnin   The number of burnin iterations for the sampler.

mcmc	The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to burnin+mcmc. burnin+mcmc must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed.
thin	The thinning interval used in the simulation. The number of mcmc iterations must be divisible by this value.
presence_data	A matrix $n_{site} \times n_{species}$ indicating the number of successes (or presences) and the absence by a zero for each species at the studied sites.
site_formula	A one-sided formula of the form '~x1+...+xp' specifying the explanatory variables for the suitability process of the model, used to form the design matrix X of size $n_{site} \times np$.
trait_data	A data frame containing the species traits which can be included as part of the model. Default to NULL to fit a model without species traits.
trait_formula	A one-sided formula of the form '~ t1 + ... + tk + x1:t1 + ... + xp:tk' specifying the interactions between the environmental variables and the species traits to be considered in the model, used to form the trait design matrix Tr of size $n_{species} \times nt$ and to set to 0 the γ parameters corresponding to interactions not taken into account according to the formula. Default to NULL to fit a model with all possible interactions between species traits found in trait_data and environmental variables defined by site_formula.
site_data	A data frame containing the model's explanatory variables by site.
trials	A vector indicating the number of trials for each site. n_i should be superior or equal to y_{ij}, the number of successes for observation n. If $n_i = 0$, then $y_{ij} = 0$. The default is one visit by site.
n_latent	An integer which specifies the number of latent variables to generate. Defaults to 0.
site_effect	A string indicating whether row effects are included as fixed effects ("fixed"), as random effects ("random"), or not included ("none") in the model. If fixed effects, then for parameter identifiability the first row effect is set to zero, which analogous to acting as a reference level when dummy variables are used. 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".
beta_start	Starting values for β parameters of the suitability process for each species must be either a scalar or a $np \times n_{species}$ matrix. If beta_start takes a scalar value, then that value will serve for all of the β parameters.
gamma_start	Starting values for γ parameters that represent the influence of species-specific traits on species' responses β, gamma_start must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If gamma_start takes a scalar value, then that value will serve for all of the γ parameters. If gamma_start is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row.
lambda_start	Starting values for λ parameters corresponding to the latent variables for each species must be either a scalar or a $n_{latent} \times n_{species}$ upper triangular matrix with strictly positive values on the diagonal, ignored if n_latent=0. If lambda_start takes a scalar value, then that value will serve for all of the λ parameters except those concerned by the constraints explained above.

W_start	Starting values for latent variables must be either a scalar or a $nsite \times n_l atent$ matrix, ignored if n_latent=0. If W_start takes a scalar value, then that value will serve for all of the W_{il} with $i = 1, \ldots, n_{site}$ and $l = 1, \ldots, n_{latent}$.
alpha_start	Starting values for random site effect parameters must be either a scalar or a n_{site}-length vector, ignored if site_effect="none". If alpha_start takes a scalar value, then that value will serve for all of the α parameters.
V_alpha	Starting value for variance of random site effect if site_effect="random" or constant variance of the Gaussian prior distribution for the fixed site effect if site_effect="fixed". Must be a strictly positive scalar, ignored if site_effect="none".
shape	Shape parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if site_effect="none" or site_effect="fixed". Must be a strictly positive scalar. Default to 0.5 for weak informative prior.
rate	Rate parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if site_effect="none" or site_effect="fixed" Must be a strictly positive scalar. Default to 0.0005 for weak informative prior.
mu_beta	Means of the Normal priors for the β parameters of the suitability process. mu_beta must be either a scalar or a np-length vector. If mu_beta takes a scalar value, then that value will serve as the prior mean for all of the β parameters. The default value is set to 0 for an uninformative prior, ignored if trait_data is specified.
V_beta	Variances of the Normal priors for the β parameters of the suitability process. V_beta must be either a scalar or a $np \times np$ symmetric positive semi-definite square matrix. If V_beta takes a scalar value, then that value will serve as the prior variance for all of the β parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
mu_gamma	Means of the Normal priors for the γ parameters. mu_gamma must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If mu_gamma takes a scalar value, then that value will serve as the prior mean for all of the γ parameters. If mu_gamma is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default value is set to 0 for an uninformative prior, ignored if trait_data=NULL.
V_gamma	Variances of the Normal priors for the γ parameters. V_gamma must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ positive matrix. If V_gamma takes a scalar value, then that value will serve as the prior variance for all of the γ parameters. If V_gamma is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default variance is large and set to 10 for an uninformative flat prior, ignored if trait_data=NULL.
mu_lambda	Means of the Normal priors for the λ parameters corresponding to the latent variables. mu_lambda must be either a scalar or a n_{latent}-length vector. If mu_lambda takes a scalar value, then that value will serve as the prior mean for all of the λ parameters. The default value is set to 0 for an uninformative prior.

V_lambda	Variances of the Normal priors for the λ parameters corresponding to the latent variables. V_lambda must be either a scalar or a $n_{latent} \times n_{latent}$ symmetric positive semi-definite square matrix. If V_lambda takes a scalar value, then that value will serve as the prior variance for all of λ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
ropt	Target acceptance rate for the adaptive Metropolis algorithm. Default to 0.44.
seed	The seed for the random number generator. Default to 1234.
verbose	A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler.

Details

We model an ecological process where the presence or absence of species j on site i is explained by habitat suitability.

Ecological process :

$$y_{ij} \sim \mathcal{B}inomial(\theta_{ij}, n_i)$$

where

if n_latent=0 and site_effect="none"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="fixed"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="random"	$\mathrm{logit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

In the absence of data on species traits (trait_data=NULL), the effect of species j: β_j; follows the same *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_\beta, V_\beta)$, for each species.

If species traits data are provided, the effect of species j: β_j; follows an *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_{\beta_j}, V_\beta)$, where $\mu_{\beta_j p} = \sum_{k=1}^{nt} t_{jk}.\gamma_{kp}$, takes different values for each species.

We assume that $\gamma_{kp} \sim \mathcal{N}(\mu_{\gamma_{kp}}, V_{\gamma_{kp}})$ as prior distribution.

We define the matrix $\gamma = (\gamma_{kp})_{k=1,...,nt}^{p=1,...,np}$ such as :

	x_0	x_1	...	x_p	...	x_{np}	
t_0	$\gamma_{0,0}$ **intercept**	$\gamma_{0,1}$	...	$\gamma_{0,p}$	...	$\gamma_{0,np}$	{ *effect of environmental variables*
t_1	$\gamma_{1,0}$	$\gamma_{1,1}$	...	$\gamma_{1,p}$	...	$\gamma_{1,np}$	
...	...	...	...	...	...	...	
t_k	$\gamma_{k,0}$	$\gamma_{k,1}$	...	$\gamma_{k,p}$	...	$\gamma_{k,np}$	
...	...	...	...	...	...	...	

t_{nt}		$\gamma_{nt,0}$	$\gamma_{nt,1}$	...	$\gamma_{nt,p}$	...	$\gamma_{nt,np}$
		average					
		trait effect		interaction	traits	environment	

Value

An object of class `"jSDM"` acting like a list including :

`mcmc.alpha`	An mcmc object that contains the posterior samples for site effects α_i, not returned if `site_effect="none"`.
`mcmc.V_alpha`	An mcmc object that contains the posterior samples for variance of random site effect, not returned if `site_effect="none"` or `site_effect="fixed"`.
`mcmc.latent`	A list by latent variable of mcmc objects that contains the posterior samples for latent variables W_l with $l = 1, \ldots, n_{latent}$, not returned if `n_latent=0`.
`mcmc.sp`	A list by species of mcmc objects that contains the posterior samples for species effects β_j and λ_j if `n_latent>0`.
`mcmc.gamma`	A list by covariates of mcmc objects that contains the posterior samples for γ_p parameters with $p = 1, \ldots, np$ if `trait_data` is specified.
`mcmc.Deviance`	The posterior sample of the deviance (D) is also provided, with D defined as : $D = -2\log(\prod_{ij} P(y_{ij}|\beta_j, \lambda_j, \alpha_i, W_i))$.
`logit_theta_latent`	Predictive posterior mean of the probability to each species to be present on each site, transformed by logit link function.
`theta_latent`	Predictive posterior mean of the probability associated to the suitability process for each observation.
`model_spec`	Various attributes of the model fitted, including the response and model matrix used, distributional assumptions as link function, family and number of latent variables, hyperparameters used in the Bayesian estimation and mcmc, burnin and thin.

The `mcmc.` objects can be summarized by functions provided by the `coda` package.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Gelfand, A. E.; Schmidt, A. M.; Wu, S.; Silander, J. A.; Latimer, A. and Rebelo, A. G. (2005) Modelling species diversity through species level hierarchical modelling. *Applied Statistics*, 54, 1-20.

Latimer, A. M.; Wu, S. S.; Gelfand, A. E. and Silander, J. A. (2006) Building statistical models to analyze species distributions. *Ecological Applications*, 16, 33-50.

Ovaskainen, O., Tikhonov, G., Norberg, A., Blanchet, F. G., Duan, L., Dunson, D., Roslin, T. and Abrego, N. (2017) How to make more out of community data? A conceptual framework and its implementation as models and software. *Ecology Letters*, 20, 561-576.

See Also

plot.mcmc, summary.mcmc jSDM_binomial_probit jSDM_poisson_log

Examples

```
#=======================================
# jSDM_binomial_logit()
# Example with simulated data
#=======================================

#=================
#== Load libraries
library(jSDM)

#==================
#== Data simulation

#= Number of sites
nsite <- 100
#= Number of species
nsp <- 20
#= Set seed for repeatability
seed <- 1234

#= Number of visits associated to each site
set.seed(seed)
visits <- rpois(nsite,3)
visits[visits==0] <- 1

#= Ecological process (suitability)
x1 <- rnorm(nsite,0,1)
set.seed(2*seed)
x2 <- rnorm(nsite,0,1)
X <- cbind(rep(1,nsite),x1,x2)
np <- ncol(X)
set.seed(3*seed)
W <- cbind(rnorm(nsite,0,1),rnorm(nsite,0,1))
n_latent <- ncol(W)
l.zero <- 0
l.diag <- runif(2,0,2)
l.other <- runif(nsp*2-3,-2,2)
lambda.target <- matrix(c(l.diag[1],l.zero,l.other[1],
                          l.diag[2],l.other[-1]),
                        byrow=TRUE, nrow=nsp)
beta.target <- matrix(runif(nsp*np,-2,2), byrow=TRUE, nrow=nsp)
V_alpha.target <- 0.5
alpha.target <- rnorm(nsite,0,sqrt(V_alpha.target))
logit.theta <- X %*% t(beta.target) + W %*% t(lambda.target) + alpha.target
theta <- inv_logit(logit.theta)
set.seed(seed)
Y <- apply(theta, 2, rbinom, n=nsite, size=visits)
```

```
#= Site-occupancy model
# Increase number of iterations (burnin and mcmc) to get convergence
mod <- jSDM_binomial_logit(# Chains
  burnin=150,
  mcmc=150,
  thin=1,
  # Response variable
  presence_data=Y,
  trials=visits,
  # Explanatory variables
  site_formula=~x1+x2,
  site_data=X,
  n_latent=n_latent,
  site_effect="random",
  # Starting values
  beta_start=0,
  lambda_start=0,
  W_start=0,
  alpha_start=0,
  V_alpha=1,
  # Priors
  shape=0.5,
  rate=0.0005,
  mu_beta=0,
  V_beta=10,
  mu_lambda=0,
  V_lambda=10,
  # Various
  seed=1234,
  ropt=0.44,
  verbose=1)
#==========
#== Outputs

#= Parameter estimates

## beta_j
# summary(mod$mcmc.sp$sp_1[,1:ncol(X)])
mean_beta <- matrix(0,nsp,np)
pdf(file=file.path(tempdir(), "Posteriors_beta_jSDM_logit.pdf"))
par(mfrow=c(ncol(X),2))
for (j in 1:nsp) {
  mean_beta[j,] <- apply(mod$mcmc.sp[[j]][,1:ncol(X)],
                           2, mean)
  for (p in 1:ncol(X)) {
    coda::traceplot(mod$mcmc.sp[[j]][,p])
    coda::densplot(mod$mcmc.sp[[j]][,p],
      main = paste(colnames(
        mod$mcmc.sp[[j]])[p],
        ", species : ",j))
    abline(v=beta.target[j,p],col='red')
  }
}
```

```
dev.off()

## lambda_j
# summary(mod$mcmc.sp$sp_1[,(ncol(X)+1):(ncol(X)+n_latent)])
# summary(mod$mcmc.sp$sp_2[,(ncol(X)+1):(ncol(X)+n_latent)])
mean_lambda <- matrix(0,nsp,n_latent)
pdf(file=file.path(tempdir(), "Posteriors_lambda_jSDM_logit.pdf"))
par(mfrow=c(n_latent*2,2))
for (j in 1:nsp) {
  mean_lambda[j,] <- apply(mod$mcmc.sp[[j]]
                                [,(ncol(X)+1):(ncol(X)+n_latent)], 2, mean)
  for (l in 1:n_latent) {
    coda::traceplot(mod$mcmc.sp[[j]][,ncol(X)+l])
    coda::densplot(mod$mcmc.sp[[j]][,ncol(X)+l],
                   main=paste(colnames(mod$mcmc.sp[[j]])
                                [ncol(X)+l],", species : ",j))
    abline(v=lambda.target[j,l],col='red')
  }
}
dev.off()

# Species effects beta and factor loadings lambda
par(mfrow=c(1,2))
plot(beta.target, mean_beta,
     main="species effect beta",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')
plot(lambda.target, mean_lambda,
     main="factor loadings lambda",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')

## W latent variables
par(mfrow=c(1,2))
for (l in 1:n_latent) {
  plot(W[,l],
       summary(mod$mcmc.latent[[paste0("lv_",l)]])[[1]][,"Mean"],
       main = paste0("Latent variable W_", l),
       xlab ="obs", ylab ="fitted")
  abline(a=0,b=1,col='red')
}

## alpha
# summary(mod$mcmc.alpha)
par(mfrow=c(1,3))
plot(alpha.target, summary(mod$mcmc.alpha)[[1]][,"Mean"],
     xlab ="obs", ylab ="fitted", main="site effect alpha")
abline(a=0,b=1,col='red')
## Valpha
# summary(mod$mcmc.V_alpha)
coda::traceplot(mod$mcmc.V_alpha)
coda::densplot(mod$mcmc.V_alpha)
abline(v=V_alpha.target,col='red')
```

```
## Deviance
# summary(mod$mcmc.Deviance)
plot(mod$mcmc.Deviance)

#= Predictions
# summary(mod$theta_latent)
par(mfrow=c(1,2))
plot(logit.theta, mod$logit_theta_latent,
     main="logit(theta)",
     xlab="obs", ylab="fitted")
abline(a=0 ,b=1, col="red")
plot(theta, mod$theta_latent,
     main="Probabilities of occurence theta",
     xlab="obs", ylab="fitted")
abline(a=0 ,b=1, col="red")
```

---

jSDM_binomial_probit   *Binomial probit regression*

Description

The jSDM_binomial_probit function performs a Binomial probit regression in a Bayesian framework. The function calls a Gibbs sampler written in C++ code which uses conjugate priors to estimate the conditional posterior distribution of model's parameters.

Usage

```
jSDM_binomial_probit(
  burnin = 5000,
  mcmc = 10000,
  thin = 10,
  presence_data,
  site_formula,
  trait_data = NULL,
  trait_formula = NULL,
  site_data,
  n_latent = 0,
  site_effect = "none",
  lambda_start = 0,
  W_start = 0,
  beta_start = 0,
  alpha_start = 0,
  gamma_start = 0,
  V_alpha = 1,
  mu_beta = 0,
  V_beta = 10,
  mu_lambda = 0,
```

```
      V_lambda = 10,
      mu_gamma = 0,
      V_gamma = 10,
      shape = 0.5,
      rate = 5e-04,
      seed = 1234,
      verbose = 1
)
```

Arguments

burnin	The number of burnin iterations for the sampler.
mcmc	The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to burnin+mcmc. burnin+mcmc must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed.
thin	The thinning interval used in the simulation. The number of mcmc iterations must be divisible by this value.
presence_data	A matrix $n_{site} \times n_{species}$ indicating the presence by a 1 (or the absence by a 0) of each species on each site.
site_formula	A one-sided formula of the form '~x1+...+xp' specifying the explanatory variables for the suitability process of the model, used to form the design matrix X of size $n_{site} \times np$.
trait_data	A data frame containing the species traits which can be included as part of the model. Default to NULL to fit a model without species traits.
trait_formula	A one-sided formula of the form '~ t1 + ... + tk + x1:t1 + ... + xp:tk' specifying the interactions between the environmental variables and the species traits to be considered in the model, used to form the trait design matrix Tr of size $n_{species} \times nt$ and to set to 0 the γ parameters corresponding to interactions not taken into account according to the formula. Default to NULL to fit a model with all possible interactions between species traits found in trait_data and environmental variables defined by site_formula.
site_data	A data frame containing the model's explanatory variables by site.
n_latent	An integer which specifies the number of latent variables to generate. Defaults to 0.
site_effect	A string indicating whether row effects are included as fixed effects ("fixed"), as random effects ("random"), or not included ("none") in the model. If fixed effects, then for parameter identifiability the first row effect is set to zero, which analogous to acting as a reference level when dummy variables are used. 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".
lambda_start	Starting values for λ parameters corresponding to the latent variables for each species must be either a scalar or a $n_{latent} \times n_{species}$ upper triangular matrix with strictly positive values on the diagonal, ignored if n_latent=0. If lambda_start takes a scalar value, then that value will serve for all of the λ parameters except those concerned by the constraints explained above.

W_start	Starting values for latent variables must be either a scalar or a $nsite \times n_l atent$ matrix, ignored if n_latent=0. If W_start takes a scalar value, then that value will serve for all of the W_{il} with $i = 1, \ldots, n_{site}$ and $l = 1, \ldots, n_{latent}$.
beta_start	Starting values for β parameters of the suitability process for each species must be either a scalar or a $np \times n_{species}$ matrix. If beta_start takes a scalar value, then that value will serve for all of the β parameters.
alpha_start	Starting values for random site effect parameters must be either a scalar or a n_{site}-length vector, ignored if site_effect="none". If alpha_start takes a scalar value, then that value will serve for all of the α parameters.
gamma_start	Starting values for γ parameters that represent the influence of species-specific traits on species' responses β, gamma_start must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If gamma_start takes a scalar value, then that value will serve for all of the γ parameters. If gamma_start is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row.
V_alpha	Starting value for variance of random site effect if site_effect="random" or constant variance of the Gaussian prior distribution for the fixed site effect if site_effect="fixed". Must be a strictly positive scalar, ignored if site_effect="none".
mu_beta	Means of the Normal priors for the β parameters of the suitability process. mu_beta must be either a scalar or a np-length vector. If mu_beta takes a scalar value, then that value will serve as the prior mean for all of the β parameters. The default value is set to 0 for an uninformative prior, ignored if trait_data is specified.
V_beta	Variances of the Normal priors for the β parameters of the suitability process. V_beta must be either a scalar or a $np \times np$ symmetric positive semi-definite square matrix. If V_beta takes a scalar value, then that value will serve as the prior variance for all of the β parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
mu_lambda	Means of the Normal priors for the λ parameters corresponding to the latent variables. mu_lambda must be either a scalar or a n_{latent}-length vector. If mu_lambda takes a scalar value, then that value will serve as the prior mean for all of the λ parameters. The default value is set to 0 for an uninformative prior.
V_lambda	Variances of the Normal priors for the λ parameters corresponding to the latent variables. V_lambda must be either a scalar or a $n_{latent} \times n_{latent}$ symmetric positive semi-definite square matrix. If V_lambda takes a scalar value, then that value will serve as the prior variance for all of λ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
mu_gamma	Means of the Normal priors for the γ parameters. mu_gamma must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If mu_gamma takes a scalar value, then that value will serve as the prior mean for all of the γ parameters. If mu_gamma is a vector of length nt or $nt.np$ the resulting $nt \times np$

matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default value is set to 0 for an uninformative prior, ignored if `trait_data=NULL`.

V_gamma
Variances of the Normal priors for the γ parameters. `V_gamma` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ positive matrix. If `V_gamma` takes a scalar value, then that value will serve as the prior variance for all of the γ parameters. If `V_gamma` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default variance is large and set to 10 for an uninformative flat prior, ignored if `trait_data=NULL`.

shape
Shape parameter of the Inverse-Gamma prior for the random site effect variance `V_alpha`, ignored if `site_effect="none"` or `site_effect="fixed"`. Must be a strictly positive scalar. Default to 0.5 for weak informative prior.

rate
Rate parameter of the Inverse-Gamma prior for the random site effect variance `V_alpha`, ignored if `site_effect="none"` or `site_effect="fixed"` Must be a strictly positive scalar. Default to 0.0005 for weak informative prior.

seed
The seed for the random number generator. Default to 1234.

verbose
A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler.

Details

We model an ecological process where the presence or absence of species j on site i is explained by habitat suitability.

Ecological process:
$$y_{ij} \sim \mathcal{B}ernoulli(\theta_{ij})$$

where

if `n_latent=0` and `site_effect="none"`	$\text{probit}(\theta_{ij}) = X_i\beta_j$
if `n_latent>0` and `site_effect="none"`	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if `n_latent=0` and `site_effect="random"`	$\text{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if `n_latent>0` and `site_effect="fixed"`	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if `n_latent=0` and `site_effect="fixed"`	$\text{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$
if `n_latent>0` and `site_effect="random"`	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

In the absence of data on species traits (`trait_data=NULL`), the effect of species j: β_j; follows the same *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_\beta, V_\beta)$, for each species.

If species traits data are provided, the effect of species j: β_j; follows an *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_{\beta_j}, V_\beta)$, where $\mu_{\beta_j p} = \sum_{k=1}^{nt} t_{jk}.\gamma_{kp}$, takes different values for each species.

We assume that $\gamma_{kp} \sim \mathcal{N}(\mu_{\gamma_{kp}}, V_{\gamma_{kp}})$ as prior distribution.

We define the matrix $\gamma = (\gamma_{kp})_{k=1,...,nt}^{p=1,...,np}$ such as :

	x_0	x_1	...	x_p	...	x_{np}	
t_0 |	$\gamma_{0,0}$	$\gamma_{0,1}$	...	$\gamma_{0,p}$	...	$\gamma_{0,np}$	{ *effect of*
|	**intercept**						*environmental*
|							*variables*
t_1 |	$\gamma_{1,0}$	$\gamma_{1,1}$	...	$\gamma_{1,p}$	...	$\gamma_{1,np}$	
... |	...	...	...	...	...	...	
t_k |	$\gamma_{k,0}$	$\gamma_{k,1}$	...	$\gamma_{k,p}$	...	$\gamma_{k,np}$	
... |	...	...	...	...	...	...	
t_{nt} |	$\gamma_{nt,0}$	$\gamma_{nt,1}$	...	$\gamma_{nt,p}$	...	$\gamma_{nt,np}$	
	average						
	trait effect		interaction	traits	environment		

Value

An object of class $"$jSDM$"$ acting like a list including :

mcmc.alpha	An mcmc object that contains the posterior samples for site effects α, not returned if site_effect="none".
mcmc.V_alpha	An mcmc object that contains the posterior samples for variance of random site effect, not returned if site_effect="none" or site_effect="fixed".
mcmc.latent	A list by latent variable of mcmc objects that contains the posterior samples for latent variables W_l with $l = 1, \ldots, n_{latent}$, not returned if n_latent=0.
mcmc.sp	A list by species of mcmc objects that contains the posterior samples for species effects β_j and λ_j if n_latent>0 with $j = 1, \ldots, n_{species}$.
mcmc.gamma	A list by covariates of mcmc objects that contains the posterior samples for γ_p parameters with $p = 1, \ldots, np$ if trait_data is specified.
mcmc.Deviance	The posterior sample of the deviance (D) is also provided, with D defined as : $D = -2 \log(\prod_{ij} P(y_{ij}|\beta_j, \lambda_j, \alpha_i, W_i))$.
Z_latent	Predictive posterior mean of the latent variable Z.
probit_theta_latent	
	Predictive posterior mean of the probability to each species to be present on each site, transformed by probit link function.
theta_latent	Predictive posterior mean of the probability to each species to be present on each site.
model_spec	Various attributes of the model fitted, including the response and model matrix used, distributional assumptions as link function, family and number of latent variables, hyperparameters used in the Bayesian estimation and mcmc, burnin and thin.

The mcmc. objects can be summarized by functions provided by the coda package.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Chib, S. and Greenberg, E. (1998) Analysis of multivariate probit models. *Biometrika*, 85, 347-361.

Warton, D. I.; Blanchet, F. G.; O'Hara, R. B.; O'Hara, R. B.; Ovaskainen, O.; Taskinen, S.; Walker, S. C. and Hui, F. K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution*, 30, 766-779.

Ovaskainen, O., Tikhonov, G., Norberg, A., Blanchet, F. G., Duan, L., Dunson, D., Roslin, T. and Abrego, N. (2017) How to make more out of community data? A conceptual framework and its implementation as models and software. *Ecology Letters*, 20, 561-576.

See Also

`plot.mcmc`, `summary.mcmc` `jSDM_binomial_logit` `jSDM_poisson_log` `jSDM_binomial_probit_sp_constrained`

Examples

```
#=====================================
# jSDM_binomial_probit()
# Example with simulated data
#=====================================

#=================
#== Load libraries
library(jSDM)

#==================
#' #== Data simulation

#= Number of sites
nsite <- 150

#= Set seed for repeatability
seed <- 1234
set.seed(seed)

#= Number of species
nsp<- 20

#= Number of latent variables
n_latent <- 2

#= Ecological process (suitability)
x1 <- rnorm(nsite,0,1)
x2 <- rnorm(nsite,0,1)
X <- cbind(rep(1,nsite),x1,x2)
np <- ncol(X)
#= Latent variables W
W <- matrix(rnorm(nsite*n_latent,0,1), nsite, n_latent)
#= Fixed species effect beta
beta.target <- t(matrix(runif(nsp*np,-2,2),
                        byrow=TRUE, nrow=nsp))
#= Factor loading lambda
```

```r
lambda.target <- matrix(0, n_latent, nsp)
mat <- t(matrix(runif(nsp*n_latent, -2, 2), byrow=TRUE, nrow=nsp))
lambda.target[upper.tri(mat, diag=TRUE)] <- mat[upper.tri(mat, diag=TRUE)]
diag(lambda.target) <- runif(n_latent, 0, 2)
#= Variance of random site effect
V_alpha.target <- 0.5
#= Random site effect alpha
alpha.target <- rnorm(nsite,0 , sqrt(V_alpha.target))
# Simulation of response data with probit link
probit_theta <- X%*%beta.target + W%*%lambda.target + alpha.target
theta <- pnorm(probit_theta)
e <- matrix(rnorm(nsp*nsite,0,1),nsite,nsp)
# Latent variable Z
Z_true <- probit_theta + e
# Presence-absence matrix Y
Y <- matrix (NA, nsite,nsp)
for (i in 1:nsite){
  for (j in 1:nsp){
    if ( Z_true[i,j] > 0) {Y[i,j] <- 1}
    else {Y[i,j] <- 0}
  }
}

#==================================
#== Site-occupancy model

# Increase number of iterations (burnin and mcmc) to get convergence
mod<-jSDM_binomial_probit(# Iteration
                          burnin=200,
                          mcmc=200,
                          thin=1,
                          # Response variable
                          presence_data=Y,
                          # Explanatory variables
                          site_formula=~x1+x2,
                          site_data = X,
                          n_latent=2,
                          site_effect="random",
                          # Starting values
                          alpha_start=0,
                          beta_start=0,
                          lambda_start=0,
                          W_start=0,
                          V_alpha=1,
                          # Priors
                          shape=0.5, rate=0.0005,
                          mu_beta=0, V_beta=1,
                          mu_lambda=0, V_lambda=1,
                          seed=1234, verbose=1)
# =================================================
# Result analysis
# =================================================
```

```r
#==========
#== Outputs

#= Parameter estimates

## beta_j
# summary(mod$mcmc.sp$sp_1[,1:ncol(X)])
mean_beta <- matrix(0,nsp,ncol(X))
pdf(file=file.path(tempdir(), "Posteriors_beta_jSDM_probit.pdf"))
par(mfrow=c(ncol(X),2))
for (j in 1:nsp) {
  mean_beta[j,] <- apply(mod$mcmc.sp[[j]]
                           [,1:ncol(X)], 2, mean)
  for (p in 1:ncol(X)){
    coda::traceplot(mod$mcmc.sp[[j]][,p])
    coda::densplot(mod$mcmc.sp[[j]][,p],
      main = paste(colnames(mod$mcmc.sp[[j]])[p],", species : ",j))
    abline(v=beta.target[p,j],col='red')
  }
}
dev.off()

## lambda_j
# summary(mod$mcmc.sp$sp_1[,(ncol(X)+1):(ncol(X)+n_latent)])
# summary(mod$mcmc.sp$sp_2[,(ncol(X)+1):(ncol(X)+n_latent)])
mean_lambda <- matrix(0,nsp,n_latent)
pdf(file=file.path(tempdir(), "Posteriors_lambda_jSDM_probit.pdf"))
par(mfrow=c(n_latent*2,2))
for (j in 1:nsp) {
  mean_lambda[j,] <- apply(mod$mcmc.sp[[j]]
                             [,(ncol(X)+1):(ncol(X)+n_latent)], 2, mean)
  for (l in 1:n_latent) {
    coda::traceplot(mod$mcmc.sp[[j]][,ncol(X)+l])
    coda::densplot(mod$mcmc.sp[[j]][,ncol(X)+l],
                   main=paste(colnames(mod$mcmc.sp[[j]])
                               [ncol(X)+l],", species : ",j))
    abline(v=lambda.target[l,j],col='red')
  }
}
dev.off()

# Species effects beta and factor loadings lambda
par(mfrow=c(1,2))
plot(t(beta.target), mean_beta,
     main="species effect beta",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')
plot(t(lambda.target), mean_lambda,
     main="factor loadings lambda",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')

## W latent variables
```

```
par(mfrow=c(1,2))
for (l in 1:n_latent) {
  plot(W[,l],
        summary(mod$mcmc.latent[[paste0("lv_",l)]])[[1]][,"Mean"],
        main = paste0("Latent variable W_", l),
        xlab ="obs", ylab ="fitted")
  abline(a=0,b=1,col='red')
}

## alpha
# summary(mod$mcmc.alpha)
par(mfrow=c(1,3))
plot(alpha.target, summary(mod$mcmc.alpha)[[1]][,"Mean"],
      xlab ="obs", ylab ="fitted", main="site effect alpha")
abline(a=0,b=1,col='red')
## Valpha
coda::traceplot(mod$mcmc.V_alpha)
coda::densplot(mod$mcmc.V_alpha)
abline(v=V_alpha.target,col='red')

## Deviance
summary(mod$mcmc.Deviance)
plot(mod$mcmc.Deviance)

#= Predictions

## Z
par(mfrow=c(1,2))
# summary(mod$Z_latent)
plot(Z_true,mod$Z_latent,
      main="Z_latent", xlab="obs", ylab="fitted")
abline(a=0,b=1,col='red')

## probit_theta
# summary(mod$probit_theta_latent)
plot(probit_theta,mod$probit_theta_latent,
      main="probit(theta)",xlab="obs",ylab="fitted")
abline(a=0,b=1,col='red')

## probabilities theta
# summary(mod$theta_latent)
par(mfrow=c(1,1))
plot(theta,mod$theta_latent,
      main="theta",xlab="obs",ylab="fitted")
abline(a=0,b=1,col='red')
```

---

jSDM_binomial_probit_long_format
            Binomial probit regression on long format data

Description

The jSDM_binomial_probit_long_format function performs a Binomial probit regression in a Bayesian framework. The function calls a Gibbs sampler written in C++ code which uses conjugate priors to estimate the conditional posterior distribution of model's parameters.

Usage

```
jSDM_binomial_probit_long_format(
  burnin = 5000,
  mcmc = 10000,
  thin = 10,
  data,
  site_formula,
  n_latent = 0,
  site_effect = "none",
  alpha_start = 0,
  gamma_start = 0,
  beta_start = 0,
  lambda_start = 0,
  W_start = 0,
  V_alpha = 1,
  shape = 0.5,
  rate = 5e-04,
  mu_gamma = 0,
  V_gamma = 10,
  mu_beta = 0,
  V_beta = 10,
  mu_lambda = 0,
  V_lambda = 10,
  seed = 1234,
  verbose = 1
)
```

Arguments

burnin	The number of burnin iterations for the sampler.
mcmc	The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to burnin+mcmc.burnin+mcmc must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed.
thin	The thinning interval used in the simulation. The number of mcmc iterations must be divisible by this value.
data	A data.frame with at least the following columns :
Y	n_{obs}-length vector indicating the presence by a 1 (or absence by a 0), of the species observed during each visit of the sites.
site	numeric or character n_{obs}-length vector indicating the visited site, (sites can be visited several times).
species	numeric or character eqnn_obsn_obs-length vector indicating the species observed,

(species may not have been recorded at all sites).

x1,...,xp   columns of explanatory variables for the suitability process of the model.

site_formula	A one-sided formula, as the formulas used by the lm function, of the form: '~ x1 + ... + xd + species:x1 + ... + species:xp' with p terms related to species effects β, specifying the explanatory variables for the suitability process of the model, including the intercept, different from the d terms related to γ parameters.
n_latent	An integer which specifies the number of latent variables to generate. Defaults to 0.
site_effect	A string indicating whether row effects are included as fixed effects ("fixed"), as random effects ("random"), or not included ("none") in the model. If fixed effects, then for parameter identifiability the first row effect is set to zero, which analogous to acting as a reference level when dummy variables are used. 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".
alpha_start	Starting values for random site effect parameters must be either a scalar or a n_{site}-length vector, ignored if site_effect="none". If alpha_start takes a scalar value, then that value will serve for all of the α parameters.
gamma_start	Starting values for gamma parameters of the suitability process must be either a scalar or a d-length vector. If gamma_start takes a scalar value, then that value will serve for all of the γ parameters.
beta_start	Starting values for beta parameters of the suitability process for each species must be either a scalar or a $p \times n_{species}$ matrix. If beta_start takes a scalar value, then that value will serve for all of the β parameters.
lambda_start	Starting values for lambda parameters corresponding to the latent variables for each species must be either a scalar or a $n_{latent} \times n_{species}$ upper triangular matrix with strictly positive values on the diagonal, ignored if n_latent=0. If lambda_start takes a scalar value, then that value will serve for all of the λ parameters except those concerned by the constraints explained above.
W_start	Starting values for latent variables must be either a scalar or a $nsite \times n_latent$ matrix, ignored if n_latent=0. If W_start takes a scalar value, then that value will serve for all of the W_{il} with $i = 1, \ldots, n_{site}$ and $l = 1, \ldots, n_{latent}$.
V_alpha	Starting value for variance of random site effect if site_effect="random" or constant variance of the Gaussian prior distribution for the fixed site effect if site_effect="fixed". Must be a strictly positive scalar, ignored if site_effect="none".
shape	Shape parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if site_effect="none" or site_effect="fixed". Must be a strictly positive scalar. Default to 0.5 for weak informative prior.
rate	Rate parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if site_effect="none" or site_effect="fixed" Must be a strictly positive scalar. Default to 0.0005 for weak informative prior.

mu_gamma	Means of the Normal priors for the γ parameters of the suitability process. mu_gamma must be either a scalar or a d-length vector. If mu_gamma takes a scalar value, then that value will serve as the prior mean for all of the γ parameters. The default value is set to 0 for an uninformative prior.
V_gamma	Variances of the Normal priors for the γ parameters of the suitability process. V_gamma must be either a scalar or a $d \times d$ symmetric positive semi-definite square matrix. If V_gamma takes a scalar value, then that value will serve as the prior variance for all of the γ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 1e+06 for an uninformative flat prior.
mu_beta	Means of the Normal priors for the β parameters of the suitability process. mu_beta must be either a scalar or a p-length vector. If mu_beta takes a scalar value, then that value will serve as the prior mean for all of the β parameters. The default value is set to 0 for an uninformative prior.
V_beta	Variances of the Normal priors for the β parameters of the suitability process. V_beta must be either a scalar or a $p \times p$ symmetric positive semi-definite square matrix. If V_beta takes a scalar value, then that value will serve as the prior variance for all of the β parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 1e+06 for an uninformative flat prior.
mu_lambda	Means of the Normal priors for the λ parameters corresponding to the latent variables. mu_lambda must be either a scalar or a n_{latent}-length vector. If mu_lambda takes a scalar value, then that value will serve as the prior mean for all of the λ parameters. The default value is set to 0 for an uninformative prior.
V_lambda	Variances of the Normal priors for the λ parameters corresponding to the latent variables. V_lambda must be either a scalar or a $n_{latent} \times n_{latent}$ symmetric positive semi-definite square matrix. If V_lambda takes a scalar value, then that value will serve as the prior variance for all of λ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
seed	The seed for the random number generator. Default to 1234.
verbose	A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler.

Details

We model an ecological process where the presence or absence of species j on site i is explained by habitat suitability.

Ecological process:

$$y_n \sim \mathcal{Bernoulli}(\theta_n)$$

such as $species_n = j$ and $site_n = i$, where :

if n_latent=0 and site_effect="none" $\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j$

if `n_latent>0` and `site_effect="none"`	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j$
if `n_latent=0` and `site_effect="fixed"`	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if `n_latent>0` and `site_effect="fixed"`	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j + \alpha_i$
if `n_latent=0` and `site_effect="random"`	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + \alpha_i$
if `n_latent>0` and `site_effect="random"`	$\text{probit}(\theta_n) = D_n\gamma + X_n\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

Value

An object of class `"jSDM"` acting like a list including :

`mcmc.alpha`	An mcmc object that contains the posterior samples for site effects α_i, not returned if `site_effect="none"`.
`mcmc.V_alpha`	An mcmc object that contains the posterior samples for variance of random site effect, not returned if `site_effect="none"` or `site_effect="fixed"`.
`mcmc.latent`	A list by latent variable of mcmc objects that contains the posterior samples for latent variables W_l with $l = 1, \ldots, n_{latent}$, not returned if `n_latent=0`.
`mcmc.sp`	A list by species of mcmc objects that contains the posterior samples for species effects β and the loading factors λ if `n_latent>0`.
`mcmc.gamma`	An mcmc objects that contains the posterior samples for parameters γ not returned if `d=0`.
`mcmc.Deviance`	The posterior sample of the deviance D is also provided, with D defined as: $D = -2\log(\prod_n P(y_n|\beta_j, \lambda_j, \alpha_i, W_i))$.
`Z_latent`	Predictive posterior mean of the latent variable Z.
`probit_theta_latent`	
	Predictive posterior mean of the probability to each species to be present on each site, transformed by probit link function.
`theta_latent`	Predictive posterior mean of the probability to each species to be present on each site.
`model_spec`	Various attributes of the model fitted, including the response and model matrix used, distributional assumptions as link function, family and number of latent variables, hyperparameters used in the Bayesian estimation and mcmc, burnin and thin.

The `mcmc.` objects can be summarized by functions provided by the `coda` package.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Chib, S. and Greenberg, E. (1998) Analysis of multivariate probit models. *Biometrika*, 85, 347-361.

Warton, D. I.; Blanchet, F. G.; O'Hara, R. B.; O'Hara, R. B.; Ovaskainen, O.; Taskinen, S.; Walker, S. C. and Hui, F. K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution*, 30, 766-779.

See Also

plot.mcmc, summary.mcmc jSDM_binomial_probit jSDM_binomial_logit jSDM_poisson_log

Examples

```
#=========================================
# jSDM_binomial_probit_long_format()
# Example with simulated data
#=========================================

#=================
#== Load libraries
library(jSDM)

#=================
#== Data simulation

#= Number of sites
nsite <- 50

#= Set seed for repeatability
seed <- 1234
set.seed(seed)

#' #= Number of species
nsp <- 25

#= Number of latent variables
n_latent <- 2
#'
# Ecological process (suitability)
## X
x1 <- rnorm(nsite,0,1)
x1.2 <- scale(x1^2)
X <- cbind(rep(1,nsite),x1,x1.2)
colnames(X) <- c("Int","x1","x1.2")
np <- ncol(X)
## W
W <- matrix(rnorm(nsite*n_latent,0,1),nrow=nsite,byrow=TRUE)
## D
SLA <- runif(nsp,-1,1)
D <- data.frame(x1.SLA= scale(c(x1 %*% t(SLA))))
nd <- ncol(D)
## parameters
beta.target <- t(matrix(runif(nsp*np,-2,2), byrow=TRUE, nrow=nsp))
mat <- t(matrix(runif(nsp*n_latent,-2,2), byrow=TRUE, nrow=nsp))
diag(mat) <- runif(n_latent,0,2)
lambda.target <- matrix(0,n_latent,nsp)
gamma.target <-runif(nd,-1,1)
lambda.target[upper.tri(mat,diag=TRUE)] <- mat[upper.tri(mat,
                                                    diag=TRUE)]
#= Variance of random site effect
```

```
V_alpha.target <- 0.5
#= Random site effect
alpha.target <- rnorm(nsite,0,sqrt(V_alpha.target))
## probit_theta
probit_theta <- c(X %*% beta.target) + c(W %*% lambda.target) +
                as.matrix(D) %*% gamma.target + rep(alpha.target, nsp)
# Supplementary observation (each site have been visited twice)
# Environmental variables at the time of the second visit
x1_supObs <- rnorm(nsite,0,1)
x1.2_supObs <- scale(x1^2)
X_supObs <- cbind(rep(1,nsite),x1_supObs,x1.2_supObs)
D_supObs <- data.frame(x1.SLA=scale(c(x1_supObs %*% t(SLA))))
probit_theta_supObs <- c(X_supObs%*%beta.target) + c(W%*%lambda.target) +
                       as.matrix(D_supObs) %*% gamma.target + alpha.target
probit_theta <- c(probit_theta, probit_theta_supObs)
nobs <- length(probit_theta)
e <- rnorm(nobs,0,1)
Z_true <- probit_theta + e
Y<-rep(0,nobs)
for (n in 1:nobs){
  if ( Z_true[n] > 0) {Y[n] <- 1}
}
Id_site <- rep(1:nsite,nsp)
Id_sp <- rep(1:nsp,each=nsite)
data <- data.frame(site=rep(Id_site,2), species=rep(Id_sp,2), Y=Y,
                   x1=c(rep(x1,nsp),rep(x1_supObs,nsp)),
                   x1.2=c(rep(x1.2,nsp),rep(x1.2_supObs,nsp)),
                   x1.SLA=c(D$x1.SLA,D_supObs$x1.SLA))
# missing observation
data <- data[-1,]

#=================================
#== Site-occupancy model

# Increase number of iterations (burnin and mcmc) to get convergence
mod<-jSDM_binomial_probit_long_format( # Iteration
  burnin=500,
  mcmc=500,
  thin=1,
  # Response variable
  data=data,
  # Explanatory variables
  site_formula=~ (x1 + x1.2):species + x1.SLA,
  n_latent=2,
  site_effect="random",
  # Starting values
  alpha_start=0,
  gamma_start=0,
  beta_start=0,
  lambda_start=0,
  W_start=0,
  V_alpha=1,
  # Priors
```

```
    shape=0.5, rate=0.0005,
    mu_gamma=0, V_gamma=10,
    mu_beta=0, V_beta=10,
    mu_lambda=0, V_lambda=10,
    seed=1234, verbose=1)

#= Parameter estimates

# gamma
par(mfrow=c(2,2))
for(d in 1:nd){
 coda::traceplot(mod$mcmc.gamma[,d])
 coda::densplot(mod$mcmc.gamma[,d],
                 main = colnames(mod$mcmc.gamma)[d])
 abline(v=gamma.target[d],col='red')
}
## beta_j
# summary(mod$mcmc.sp$sp_1[,1:ncol(X)])
mean_beta <- matrix(0,nsp,ncol(X))
pdf(file=file.path(tempdir(), "Posteriors_beta_jSDM_probit.pdf"))
par(mfrow=c(ncol(X),2))
for (j in 1:nsp) {
  mean_beta[j,] <- apply(mod$mcmc.sp[[j]]
                            [,1:ncol(X)], 2, mean)
  for (p in 1:ncol(X)){
    coda::traceplot(mod$mcmc.sp[[j]][,p])
    coda::densplot(mod$mcmc.sp[[j]][,p],
      main = paste0(colnames(mod$mcmc.sp[[j]])[p],"_sp",j))
    abline(v=beta.target[p,j],col='red')
  }
}
dev.off()

## lambda_j
# summary(mod$mcmc.sp$sp_1[,(ncol(X)+1):(ncol(X)+n_latent)])
# summary(mod$mcmc.sp$sp_2[,(ncol(X)+1):(ncol(X)+n_latent)])
mean_lambda <- matrix(0,nsp,n_latent)
pdf(file=file.path(tempdir(), "Posteriors_lambda_jSDM_probit.pdf"))
par(mfrow=c(n_latent*2,2))
for (j in 1:nsp) {
  mean_lambda[j,] <- apply(mod$mcmc.sp[[j]]
                            [,(ncol(X)+1):(ncol(X)+n_latent)], 2, mean)
  for (l in 1:n_latent) {
    coda::traceplot(mod$mcmc.sp[[j]][,ncol(X)+l])
    coda::densplot(mod$mcmc.sp[[j]][,ncol(X)+l],
                    main=paste0(colnames(mod$mcmc.sp[[j]])[ncol(X)+l], "_sp",j))
    abline(v=lambda.target[l,j],col='red')
  }
}
dev.off()

# Species effects beta and factor loadings lambda
par(mfrow=c(1,2))
```

```
plot(t(beta.target), mean_beta,
     main="species effect beta",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')
plot(t(lambda.target), mean_lambda,
     main="factor loadings lambda",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')

## W latent variables
par(mfrow=c(1,2))
for (l in 1:n_latent) {
  plot(W[,l],
       summary(mod$mcmc.latent[[paste0("lv_",l)]])[[1]][,"Mean"],
       main = paste0("Latent variable W_", l),
       xlab ="obs", ylab ="fitted")
  abline(a=0,b=1,col='red')
}

## alpha
# summary(mod$mcmc.alpha)
par(mfrow=c(1,3))
plot(alpha.target, summary(mod$mcmc.alpha)[[1]][,"Mean"],
     xlab ="obs", ylab ="fitted", main="site effect alpha")
abline(a=0,b=1,col='red')
## Valpha
coda::traceplot(mod$mcmc.V_alpha, main="Trace V_alpha")
coda::densplot(mod$mcmc.V_alpha,main="Density V_alpha")
abline(v=V_alpha.target,col='red')

## Deviance
#summary(mod$mcmc.Deviance)
plot(mod$mcmc.Deviance)

#= Predictions

## probit_theta
# summary(mod$probit_theta_latent)
par(mfrow=c(1,2))
plot(probit_theta[-1],mod$probit_theta_latent,
     main="probit(theta)",xlab="obs",ylab="fitted")
abline(a=0,b=1,col='red')

## Z
# summary(mod$Z_latent)
plot(Z_true[-1],mod$Z_latent,
     main="Z_latent", xlab="obs", ylab="fitted")
abline(a=0,b=1,col='red')
## theta
# summary(mod$theta_latent)
par(mfrow=c(1,1))
plot(pnorm(probit_theta[-1]),mod$theta_latent,
     main="theta",xlab="obs",ylab="fitted")
```

```
abline(a=0,b=1,col='red')
```

---

jSDM_binomial_probit_sp_constrained
Binomial probit regression with the chosen constrained species

---

Description

The `jSDM_binomial_probit_sp_constrained` function performs in parallel Binomial probit regressions in a Bayesian framework. The function calls a Gibbs sampler written in C++ code which uses conjugate priors to estimate the conditional posterior distribution of model's parameters. Then the function evaluates the convergence of MCMC λ chains using the Gelman-Rubin convergence diagnostic ($\hat{R}$). $\hat{R}$ is computed using the gelman.diag function. We identify the species ($\hat{j}_l$) having the higher $\hat{R}$ for $\lambda_{\hat{j}_l}$. These species drive the structure of the latent axis l. The λ corresponding to this species are constrained to be positive and placed on the diagonal of the Λ matrix for fitting a second model. This usually improves the convergence of the latent variables and factor loadings. The function returns the parameter posterior distributions for this second model.

Arguments

burnin	The number of burn-in iterations for the sampler.
mcmc	The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to burnin+mcmc for each chain. burnin+mcmc must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed.
thin	The thinning interval used in the simulation. The number of MCMC iterations must be divisible by this value.
nchains	The number of Monte Carlo Markov Chains (MCMCs) simulated in parallel. If not specified, the number of chains is set to 2.
ncores	The number of cores to use for parallel execution. If not specified, the number of cores is set to 2.
presence_data	A matrix $n_{site} \times n_{species}$ indicating the presence by a 1 (or the absence by a 0) of each species on each site.
site_formula	A one-sided formula of the form '~x1+...+xp' specifying the explanatory variables for the suitability process of the model, used to form the design matrix X of size $n_{site} \times np$.
site_data	A data frame containing the model's explanatory variables by site.
trait_data	A data frame containing the species traits which can be included as part of the model. Default to NULL to fit a model without species traits.
trait_formula	A one-sided formula of the form '~ t1 + ... + tk + x1:t1 + ... + xp:tk' specifying the interactions between the environmental variables and the species traits to be considered in the model, used to form the trait design matrix Tr of size $n_{species} \times nt$ and to set to 0 the γ parameters corresponding to interactions not taken into account according to the formula. Default to NULL to fit a model with all possible interactions between species traits found in trait_data and environmental variables defined by site_formula.

n_latent   An integer which specifies the number of latent variables to generate. Defaults to 0.

site_effect A string indicating whether row effects are included as fixed effects (`"fixed"`), as random effects (`"random"`), or not included (`"none"`) in the model. If fixed effects, then for parameter identifiability the first row effect is set to zero, which analogous to acting as a reference level when dummy variables are used. 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"`.

beta_start   Starting values for β parameters of the suitability process for each species must be either a scalar or a $np \times n_{species}$ matrix. If `beta_start` takes a scalar value, then that value will serve for all of the β parameters. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

gamma_start Starting values for γ parameters that represent the influence of species-specific traits on species' responses β, `gamma_start` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If `gamma_start` takes a scalar value, then that value will serve for all of the γ parameters. If `gamma_start` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

lambda_start Starting values for λ parameters corresponding to the latent variables for each species must be either a scalar or a $n_{latent} \times n_{species}$ upper triangular matrix with strictly positive values on the diagonal, ignored if `n_latent=0`. If `lambda_start` takes a scalar value, then that value will serve for all of the λ parameters except those concerned by the constraints explained above. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

W_start   Starting values for latent variables must be either a scalar or a $nsite \times n_latent$ matrix, ignored if `n_latent=0`. If `W_start` takes a scalar value, then that value will serve for all of the W_{il} with $i = 1, \ldots, n_{site}$ and $l = 1, \ldots, n_{latent}$. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

alpha_start Starting values for random site effect parameters must be either a scalar or a n_{site}-length vector, ignored if `site_effect="none"`. If `alpha_start` takes a scalar value, then that value will serve for all of the α parameters. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

V_alpha   Starting value for variance of random site effect if `site_effect="random"` or constant variance of the Gaussian prior distribution for the fixed site effect if `site_effect="fixed"`. Must be a strictly positive scalar, ignored if `site_effect="none"`. Different starting values for each chain can be specified by a list or a vector of length nchains, by default the same starting values are considered for all chains.

shape   Shape parameter of the Inverse-Gamma prior for the random site effect variance

	V_alpha, ignored if `site_effect="none"` or `site_effect="fixed"`. Must be a strictly positive scalar. Default to 0.5 for weak informative prior.
rate	Rate parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if `site_effect="none"` or `site_effect="fixed"` Must be a strictly positive scalar. Default to 0.0005 for weak informative prior.
mu_beta	Means of the Normal priors for the β parameters of the suitability process. `mu_beta` must be either a scalar or a np-length vector. If `mu_beta` takes a scalar value, then that value will serve as the prior mean for all of the β parameters. The default value is set to 0 for an uninformative prior, ignored if `trait_data` is specified.
V_beta	Variances of the Normal priors for the β parameters of the suitability process. `V_beta` must be either a scalar or a $np \times np$ symmetric positive semi-definite square matrix. If `V_beta` takes a scalar value, then that value will serve as the prior variance for all of the β parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
mu_gamma	Means of the Normal priors for the γ parameters. `mu_gamma` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If `mu_gamma` takes a scalar value, then that value will serve as the prior mean for all of the γ parameters. If `mu_gamma` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default value is set to 0 for an uninformative prior, ignored if `trait_data=NULL`.
V_gamma	Variances of the Normal priors for the γ parameters. `V_gamma` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ positive matrix. If `V_gamma` takes a scalar value, then that value will serve as the prior variance for all of the γ parameters. If `V_gamma` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default variance is large and set to 10 for an uninformative flat prior, ignored if `trait_data=NULL`.
mu_lambda	Means of the Normal priors for the λ parameters corresponding to the latent variables. `mu_lambda` must be either a scalar or a n_{latent}-length vector. If `mu_lambda` takes a scalar value, then that value will serve as the prior mean for all of the λ parameters. The default value is set to 0 for an uninformative prior.
V_lambda	Variances of the Normal priors for the λ parameters corresponding to the latent variables. `V_lambda` must be either a scalar or a $n_{latent} \times n_{latent}$ symmetric positive semi-definite square matrix. If `V_lambda` takes a scalar value, then that value will serve as the prior variance for all of λ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
seed	The seed for the random number generator. Default to `c(123,1234)` for two chains and for more chains different seeds are generated for each chain.
verbose	A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler.

Details

We model an ecological process where the presence or absence of species j on site i is explained by habitat suitability.

Ecological process:

$$y_{ij} \sim \mathcal{B}ernoulli(\theta_{ij})$$

where

if n_latent=0 and site_effect="none"	$\text{probit}(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="random"	$\text{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="fixed"	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="fixed"	$\text{probit}(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="random"	$\text{probit}(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

In the absence of data on species traits (trait_data=NULL), the effect of species j: β_j; follows the same *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_\beta, V_\beta)$, for each species.

If species traits data are provided, the effect of species j: β_j; follows an *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_{\beta_j}, V_\beta)$, where $\mu_{\beta_j p} = \sum_{k=1}^{nt} t_{jk}.\gamma_{kp}$, takes different values for each species.

We assume that $\gamma_{kp} \sim \mathcal{N}(\mu_{\gamma_{kp}}, V_{\gamma_{kp}})$ as prior distribution.

We define the matrix $\gamma = (\gamma_{kp})_{k=1,\dots,nt}^{p=1,\dots,np}$ such as :

	x_0	x_1	...	x_p	...	x_{np}	
t_0 |	$\gamma_{0,0}$ **intercept**	$\gamma_{0,1}$	...	$\gamma_{0,p}$	...	$\gamma_{0,np}$	{ *effect of environmental variables*
t_1 |	$\gamma_{1,0}$	$\gamma_{1,1}$	...	$\gamma_{1,p}$	...	$\gamma_{1,np}$	
... |	...	...	...	...	...	...	
t_k |	$\gamma_{k,0}$	$\gamma_{k,1}$	...	$\gamma_{k,p}$	...	$\gamma_{k,np}$	
... |	...	...	...	...	...	...	
t_{nt} |	$\gamma_{nt,0}$ *average trait effect*	$\gamma_{nt,1}$	...	$\gamma_{nt,p}$	...	$\gamma_{nt,np}$	
			interaction	traits	environment		

Value

A list of length nchains where each element is an object of class "jSDM" acting like a list including :

mcmc.alpha	An mcmc object that contains the posterior samples for site effects α, not returned if site_effect="none".
mcmc.V_alpha	An mcmc object that contains the posterior samples for variance of random site effect, not returned if site_effect="none" or site_effect="fixed".

mcmc.latent	A list by latent variable of mcmc objects that contains the posterior samples for latent variables W_l with $l = 1, \ldots, n_{latent}$, not returned if n_latent=0.
mcmc.sp	A list by species of mcmc objects that contains the posterior samples for species effects β_j and λ_j if n_latent>0 with $j = 1, \ldots, n_{species}$.
mcmc.gamma	A list by covariates of mcmc objects that contains the posterior samples for γ_p parameters with $p = 1, \ldots, np$ if trait_data is specified.
mcmc.Deviance	The posterior sample of the deviance (D) is also provided, with D defined as : $D = -2\log(\prod_{ij} P(y_{ij}|\beta_j, \lambda_j, \alpha_i, W_i))$.
Z_latent	Predictive posterior mean of the latent variable Z.
probit_theta_latent	
	Predictive posterior mean of the probability to each species to be present on each site, transformed by probit link function.
theta_latent	Predictive posterior mean of the probability to each species to be present on each site.
model_spec	Various attributes of the model fitted, including the response and model matrix used, distributional assumptions as link function, family and number of latent variables, hyperparameters used in the Bayesian estimation and mcmc, burnin and thin.
sp_constrained	Indicates the reference species ($\hat{j}_l$) considered that structures itself most clearly on each latent axis l, chosen such as $\lambda_{\hat{j}_l}$ maximize the $\hat{R}$ computed on all chains. The $lambda$ corresponding to this species are constrained to be positive by being placed on the diagonal of the Λ matrix.

The mcmc. objects can be summarized by functions provided by the coda package.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Chib, S. and Greenberg, E. (1998) Analysis of multivariate probit models. *Biometrika*, 85, 347-361.

Warton, D. I.; Blanchet, F. G.; O'Hara, R. B.; O'Hara, R. B.; Ovaskainen, O.; Taskinen, S.; Walker, S. C. and Hui, F. K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution*, 30, 766-779.

Ovaskainen, O., Tikhonov, G., Norberg, A., Blanchet, F. G., Duan, L., Dunson, D., Roslin, T. and Abrego, N. (2017) How to make more out of community data? A conceptual framework and its implementation as models and software. *Ecology Letters*, 20, 561-576.

See Also

plot.mcmc, summary.mcmc mcmc.list
mcmc gelman.diag jSDM_binomial_probit

Examples

```
#=================================
# jSDM_binomial_probit_sp_constrained()
# Example with simulated data
#=================================


#================
#== Load libraries
library(jSDM)

#=================
#== Data simulation

#= Number of sites
nsite <- 50

#= Set seed for repeatability
seed <- 1234
set.seed(seed)

#= Number of species
nsp <- 10

#= Number of latent variables
n_latent <- 2

#= Ecological process (suitability)
x1 <- rnorm(nsite,0,1)
x2 <- rnorm(nsite,0,1)
X <- cbind(rep(1,nsite),x1,x2)
np <- ncol(X)
#= Latent variables W
W <- matrix(rnorm(nsite*n_latent,0,1), nsite, n_latent)
#= Fixed species effect beta
beta.target <- t(matrix(runif(nsp*np,-2,2),
                        byrow=TRUE, nrow=nsp))
#= Factor loading lambda
lambda.target <- matrix(0, n_latent, nsp)
mat <- t(matrix(runif(nsp*n_latent, -2, 2), byrow=TRUE, nrow=nsp))
lambda.target[upper.tri(mat, diag=TRUE)] <- mat[upper.tri(mat, diag=TRUE)]
diag(lambda.target) <- runif(n_latent, 0, 2)
#= Variance of random site effect
V_alpha.target <- 0.5
#= Random site effect alpha
alpha.target <- rnorm(nsite,0 , sqrt(V_alpha.target))
# Simulation of response data with probit link
probit_theta <- X%*%beta.target + W%*%lambda.target + alpha.target
theta <- pnorm(probit_theta)
e <- matrix(rnorm(nsp*nsite,0,1),nsite,nsp)
# Latent variable Z
Z_true <- probit_theta + e
# Presence-absence matrix Y
```

```
Y <- matrix (NA, nsite,nsp)
for (i in 1:nsite){
  for (j in 1:nsp){
    if ( Z_true[i,j] > 0) {Y[i,j] <- 1}
    else {Y[i,j] <- 0}
  }
}

#=================================
#== Site-occupancy model

# Increase number of iterations (burnin and mcmc) to get convergence
mod <- jSDM_binomial_probit_sp_constrained(# Iteration
                                           burnin=100,
                                           mcmc=200,
                                           thin=1,
                                           # parallel MCMCs
                                           nchains=2, ncores=2,
                                           # Response variable
                                           presence_data=Y,
                                           # Explanatory variables
                                           site_formula=~x1+x2,
                                           site_data = X,
                                           n_latent=2,
                                           site_effect="random",
                                           # Starting values
                                           alpha_start=0,
                                           beta_start=0,
                                           lambda_start=0,
                                           W_start=0,
                                           V_alpha=1,
                                           # Priors
                                           shape=0.5, rate=0.0005,
                                           mu_beta=0, V_beta=1,
                                           mu_lambda=0, V_lambda=1,
                                           seed=c(123,1234), verbose=1)

# =================================================
# Result analysis
# =================================================

#==========
#== Outputs
burnin <- mod[[1]]$model_spec$burnin
ngibbs <- burnin + mod[[1]]$model_spec$mcmc
thin <-  mod[[1]]$model_spec$thin
require(coda)
arr2mcmc <- function(x) {
  return(mcmc(as.data.frame(x),
              start=burnin+1 , end=ngibbs, thin=thin))
}
mcmc_list_Deviance <- mcmc.list(lapply(lapply(mod,"[[","mcmc.Deviance"), arr2mcmc))
mcmc_list_alpha <- mcmc.list(lapply(lapply(mod,"[[","mcmc.alpha"), arr2mcmc))
```

```
mcmc_list_V_alpha <- mcmc.list(lapply(lapply(mod,"[[","mcmc.V_alpha"), arr2mcmc))
mcmc_list_param <- mcmc.list(lapply(lapply(mod,"[[","mcmc.sp"), arr2mcmc))
mcmc_list_lv <- mcmc.list(lapply(lapply(mod,"[[","mcmc.latent"), arr2mcmc))
mcmc_list_beta <- mcmc_list_param[,grep("beta",colnames(mcmc_list_param[[1]]))]
mcmc_list_beta0 <- mcmc_list_beta[,grep("Intercept", colnames(mcmc_list_beta[[1]]))]
mcmc_list_lambda <- mcmc.list(
  lapply(mcmc_list_param[, grep("lambda", colnames(mcmc_list_param[[1]]))],
         arr2mcmc))
# Compute Rhat
psrf_alpha <- mean(gelman.diag(mcmc_list_alpha, multivariate=FALSE)$psrf[,2])
psrf_V_alpha <- gelman.diag(mcmc_list_V_alpha)$psrf[,2]
psrf_beta0 <- mean(gelman.diag(mcmc_list_beta0)$psrf[,2])
psrf_beta <- mean(gelman.diag(mcmc_list_beta[,grep("Intercept",
                                            colnames(mcmc_list_beta[[1]]),
                                            invert=TRUE)])$psrf[,2])
psrf_lambda <- mean(gelman.diag(mcmc_list_lambda,
                              multivariate=FALSE)$psrf[,2],
                    na.rm=TRUE)
psrf_lv <- mean(gelman.diag(mcmc_list_lv, multivariate=FALSE)$psrf[,2])
Rhat <- data.frame(Rhat=c(psrf_alpha, psrf_V_alpha, psrf_beta0, psrf_beta,
                          psrf_lambda, psrf_lv),
                  Variable=c("alpha", "Valpha", "beta0", "beta",
                             "lambda", "W"))
# Barplot
library(ggplot2)
ggplot2::ggplot(Rhat, aes(x=Variable, y=Rhat)) +
  ggtitle("Averages of Rhat obtained with jSDM for each type of parameter") +
  theme(plot.title = element_text(hjust = 0.5, size=15)) +
  geom_bar(fill="skyblue", stat = "identity") +
  geom_text(aes(label=round(Rhat,2)), vjust=0, hjust=-0.1) +
  geom_hline(yintercept=1, color='red') +
  coord_flip()

#= Parameter estimates
nchains <- length(mod)
## beta_j
pdf(file=file.path(tempdir(), "Posteriors_species_effect_jSDM_probit.pdf"))
plot(mcmc_list_param)
dev.off()

## Latent variables
pdf(file=file.path(tempdir(), "Posteriors_latent_variables_jSDM_probit.pdf"))
plot(mcmc_list_lv)
dev.off()

## Random site effect and its variance
pdf(file=file.path(tempdir(), "Posteriors_site_effect_jSDM_probit.pdf"))
plot(mcmc_list_V_alpha)
plot(mcmc_list_alpha)
dev.off()

## Predictive posterior mean for each observation
# Species effects beta and factor loadings lambda
```

```
param <- list()
for (i in 1:nchains){
param[[i]] <- matrix(unlist(lapply(mod[[i]]$mcmc.sp,colMeans)),
                        nrow=nsp, byrow=TRUE)
}
par(mfrow=c(1,1))
for (i in 1:nchains){
  if(i==1){
    plot(t(beta.target), param[[i]][,1:np],
         main="species effect beta",
         xlab ="obs", ylab ="fitted")
    abline(a=0,b=1, col='red')
  }
  else{
    points(t(beta.target), param[[i]][,1:np], col=i)
  }
}
par(mfrow=c(1,2))
for(l in 1:n_latent){
  for (i in 1:nchains){
    if (i==1){
      plot(t(lambda.target)[,l],
           param[[i]][,np+l],
           main=paste0("factor loadings lambda_", l),
           xlab ="obs", ylab ="fitted")
      abline(a=0,b=1, col='red')
    } else {
      points(t(lambda.target)[,l],
             param[[i]][,np+l],
             col=i)
    }
  }
}
## W latent variables
mean_W <- list()
for (i in 1:nchains){
  mean_W[[i]] <- sapply(mod[[i]]$mcmc.latent,colMeans)
}
par(mfrow=c(1,2))
for (l in 1:n_latent) {
  for (i in 1:nchains){
    if (i==1){
      plot(W[,l], mean_W[[i]][,l],
           main = paste0("Latent variable W_", l),
           xlab ="obs", ylab ="fitted")
      abline(a=0,b=1, col='red')
    }
    else{
      points(W[,l], mean_W[[i]][,l], col=i)
    }
  }
}
```

```
#= W.lambda
par(mfrow=c(1,2))
for (i in 1:nchains){
  if (i==1){
    plot(W%*%lambda.target,
         mean_W[[i]]%*%t(param[[i]][,(np+1):(np+n_latent)]),
         main = "W.lambda",
         xlab ="obs", ylab ="fitted")
    abline(a=0,b=1, col='red')
  }
  else{
    points(W%*%lambda.target,
           mean_W[[i]]%*%t(param[[i]][,(np+1):(np+n_latent)]),
           col=i)
  }
}

# Site effect alpha et V_alpha
plot(alpha.target,colMeans(mod[[1]]$mcmc.alpha),
     xlab="obs", ylab="fitted",
     main="Random site effect alpha")
abline(a=0,b=1, col='red')
points(V_alpha.target, mean(mod[[1]]$mcmc.V_alpha),
       pch=18, cex=2)
legend("bottomright", legend=c("V_alpha"), pch =18, pt.cex=1.5)
for (i in 2:nchains){
  points(alpha.target, colMeans(mod[[i]]$mcmc.alpha), col=i)
  points(V_alpha.target, mean(mod[[i]]$mcmc.V_alpha),
         pch=18, col=i, cex=2)
}

#= Predictions
## probabilities theta
plot(pnorm(probit_theta), mod[[1]]$theta_latent,
     main="theta",xlab="obs",ylab="fitted")
for (i in 2:nchains){
  points(pnorm(probit_theta), mod[[i]]$theta_latent, col=i)
}
abline(a=0,b=1, col='red')
## probit(theta)
plot(probit_theta, mod[[1]]$probit_theta_latent,
     main="probit(theta)",xlab="obs",ylab="fitted")
for (i in 2:nchains){
  points(probit_theta, mod[[i]]$probit_theta_latent, col=i)
}
abline(a=0,b=1, col='red')
## Deviance
plot(mcmc_list_Deviance)
```

---

jSDM_poisson_log   *Poisson regression with log link function*

Description

The `jSDM_poisson_log` function performs a Poisson regression with log link function in a Bayesian framework. The function calls a Gibbs sampler written in C++ code which uses an adaptive Metropolis algorithm to estimate the conditional posterior distribution of model's parameters.

Usage

```
jSDM_poisson_log(
  burnin = 5000,
  mcmc = 10000,
  thin = 10,
  count_data,
  site_data,
  site_formula,
  trait_data = NULL,
  trait_formula = NULL,
  n_latent = 0,
  site_effect = "none",
  beta_start = 0,
  gamma_start = 0,
  lambda_start = 0,
  W_start = 0,
  alpha_start = 0,
  V_alpha = 1,
  shape = 0.5,
  rate = 5e-04,
  mu_beta = 0,
  V_beta = 10,
  mu_gamma = 0,
  V_gamma = 10,
  mu_lambda = 0,
  V_lambda = 10,
  ropt = 0.44,
  seed = 1234,
  verbose = 1
)
```

Arguments

burnin	The number of burnin iterations for the sampler.
mcmc	The number of Gibbs iterations for the sampler. Total number of Gibbs iterations is equal to `burnin+mcmc`. `burnin+mcmc` must be divisible by 10 and superior or equal to 100 so that the progress bar can be displayed.
thin	The thinning interval used in the simulation. The number of mcmc iterations must be divisible by this value.
count_data	A matrix $n_{site} \times n_{species}$ indicating the abundance of each species on each site.

site_data	A data frame containing the model's explanatory variables by site.
site_formula	A one-sided formula of the form '~x1+...+xp' specifying the explanatory variables for the suitability process of the model, used to form the design matrix X of size $n_{site} \times np$.
trait_data	A data frame containing the species traits which can be included as part of the model. Default to NULL to fit a model without species traits.
trait_formula	A one-sided formula of the form '~ t1 + ... + tk + x1:t1 + ... + xp:tk' specifying the interactions between the environmental variables and the species traits to be considered in the model, used to form the trait design matrix Tr of size $n_{species} \times nt$ and to set to 0 the γ parameters corresponding to interactions not taken into account according to the formula. Default to NULL to fit a model with all possible interactions between species traits found in trait_data and environmental variables defined by site_formula.
n_latent	An integer which specifies the number of latent variables to generate. Defaults to 0.
site_effect	A string indicating whether row effects are included as fixed effects ("fixed"), as random effects ("random"), or not included ("none") in the model. If fixed effects, then for parameter identifiability the first row effect is set to zero, which analogous to acting as a reference level when dummy variables are used. 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".
beta_start	Starting values for β parameters of the suitability process for each species must be either a scalar or a $np \times n_{species}$ matrix. If beta_start takes a scalar value, then that value will serve for all of the β parameters.
gamma_start	Starting values for γ parameters that represent the influence of species-specific traits on species' responses β, gamma_start must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If gamma_start takes a scalar value, then that value will serve for all of the γ parameters. If gamma_start is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row.
lambda_start	Starting values for λ parameters corresponding to the latent variables for each species must be either a scalar or a $n_{latent} \times n_{species}$ upper triangular matrix with strictly positive values on the diagonal, ignored if n_latent=0. If lambda_start takes a scalar value, then that value will serve for all of the λ parameters except those concerned by the constraints explained above.
W_start	Starting values for latent variables must be either a scalar or a $nsite \times n_latent$ matrix, ignored if n_latent=0. If W_start takes a scalar value, then that value will serve for all of the W_{il} with $i = 1, \ldots, n_{site}$ and $l = 1, \ldots, n_{latent}$.
alpha_start	Starting values for random site effect parameters must be either a scalar or a n_{site}-length vector, ignored if site_effect="none". If alpha_start takes a scalar value, then that value will serve for all of the α parameters.

V_alpha	Starting value for variance of random site effect if `site_effect="random"` or constant variance of the Gaussian prior distribution for the fixed site effect if `site_effect="fixed"`. Must be a strictly positive scalar, ignored if `site_effect="none"`.
shape	Shape parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if `site_effect="none"` or `site_effect="fixed"`. Must be a strictly positive scalar. Default to 0.5 for weak informative prior.
rate	Rate parameter of the Inverse-Gamma prior for the random site effect variance V_alpha, ignored if `site_effect="none"` or `site_effect="fixed"` Must be a strictly positive scalar. Default to 0.0005 for weak informative prior.
mu_beta	Means of the Normal priors for the β parameters of the suitability process. `mu_beta` must be either a scalar or a np-length vector. If `mu_beta` takes a scalar value, then that value will serve as the prior mean for all of the β parameters. The default value is set to 0 for an uninformative prior, ignored if `trait_data` is specified.
V_beta	Variances of the Normal priors for the β parameters of the suitability process. `V_beta` must be either a scalar or a $np \times np$ symmetric positive semi-definite square matrix. If `V_beta` takes a scalar value, then that value will serve as the prior variance for all of the β parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.
mu_gamma	Means of the Normal priors for the γ parameters. `mu_gamma` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ matrix. If `mu_gamma` takes a scalar value, then that value will serve as the prior mean for all of the γ parameters. If `mu_gamma` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default value is set to 0 for an uninformative prior, ignored if `trait_data=NULL`.
V_gamma	Variances of the Normal priors for the γ parameters. `V_gamma` must be either a scalar, a vector of length nt, np or $nt.np$ or a $nt \times np$ positive matrix. If `V_gamma` takes a scalar value, then that value will serve as the prior variance for all of the γ parameters. If `V_gamma` is a vector of length nt or $nt.np$ the resulting $nt \times np$ matrix is filled by column with specified values, if a np-length vector is given, the matrix is filled by row. The default variance is large and set to 10 for an uninformative flat prior, ignored if `trait_data=NULL`.
mu_lambda	Means of the Normal priors for the λ parameters corresponding to the latent variables. `mu_lambda` must be either a scalar or a n_{latent}-length vector. If `mu_lambda` takes a scalar value, then that value will serve as the prior mean for all of the λ parameters. The default value is set to 0 for an uninformative prior.
V_lambda	Variances of the Normal priors for the λ parameters corresponding to the latent variables. `V_lambda` must be either a scalar or a $n_{latent} \times n_{latent}$ symmetric positive semi-definite square matrix. If `V_lambda` takes a scalar value, then that value will serve as the prior variance for all of λ parameters, so the variance covariance matrix used in this case is diagonal with the specified value on the diagonal. The default variance is large and set to 10 for an uninformative flat prior.

ropt	Target acceptance rate for the adaptive Metropolis algorithm. Default to 0.44.
seed	The seed for the random number generator. Default to 1234.
verbose	A switch (0,1) which determines whether or not the progress of the sampler is printed to the screen. Default is 1: a progress bar is printed, indicating the step (in %) reached by the Gibbs sampler.

Details

We model an ecological process where the presence or absence of species j on site i is explained by habitat suitability.

Ecological process :

$$y_{ij} \sim \mathcal{P}oisson(\theta_{ij})$$

where

if n_latent=0 and site_effect="none"	$\log(\theta_{ij}) = X_i\beta_j$
if n_latent>0 and site_effect="none"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j$
if n_latent=0 and site_effect="fixed"	$\log(\theta_{ij}) = X_i\beta_j + \alpha_i$
if n_latent>0 and site_effect="fixed"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$
if n_latent=0 and site_effect="random"	$\log(\theta_{ij}) = X_i\beta_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$
if n_latent>0 and site_effect="random"	$\log(\theta_{ij}) = X_i\beta_j + W_i\lambda_j + \alpha_i$ and $\alpha_i \sim \mathcal{N}(0, V_\alpha)$

In the absence of data on species traits (trait_data=NULL), the effect of species j: β_j; follows the same *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_\beta, V_\beta)$, for each species.

If species traits data are provided, the effect of species j: β_j; follows an *a priori* Gaussian distribution such that $\beta_j \sim \mathcal{N}_{np}(\mu_{\beta_j}, V_\beta)$, where $\mu_{\beta_j p} = \sum_{k=1}^{nt} t_{jk}.\gamma_{kp}$, takes different values for each species.

We assume that $\gamma_{kp} \sim \mathcal{N}(\mu_{\gamma_{kp}}, V_{\gamma_{kp}})$ as prior distribution.

We define the matrix $\gamma = (\gamma_{kp})_{k=1,...,nt}^{p=1,...,np}$ such as :

	x_0	x_1	...	x_p	...	x_{np}	
t_0 |	$\gamma_{0,0}$ **intercept**	$\gamma_{0,1}$	...	$\gamma_{0,p}$	...	$\gamma_{0,np}$	{ *effect of environmental variables*
t_1 |	$\gamma_{1,0}$	$\gamma_{1,1}$	...	$\gamma_{1,p}$	...	$\gamma_{1,np}$	
... |	...	...	...	...	...	...	
t_k |	$\gamma_{k,0}$	$\gamma_{k,1}$	...	$\gamma_{k,p}$	...	$\gamma_{k,np}$	
... |	...	...	...	...	...	...	
t_{nt} |	$\gamma_{nt,0}$ *average trait effect*	$\gamma_{nt,1}$	...	$\gamma_{nt,p}$ interaction	... traits	$\gamma_{nt,np}$ environment	

Value

An object of class `"jSDM"` acting like a list including :

mcmc.alpha	An mcmc object that contains the posterior samples for site effects α_i, not returned if `site_effect="none"`.
mcmc.V_alpha	An mcmc object that contains the posterior samples for variance of random site effect, not returned if `site_effect="none"` or `site_effect="fixed"`.
mcmc.latent	A list by latent variable of mcmc objects that contains the posterior samples for latent variables W_l with $l = 1, \ldots, n_{latent}$, not returned if `n_latent=0`.
mcmc.sp	A list by species of mcmc objects that contains the posterior samples for species effects β_j and λ_j if `n_latent>0`.
mcmc.gamma	A list by covariates of mcmc objects that contains the posterior samples for γ_p parameters with $p = 1, \ldots, np$ if `trait_data` is specified.
mcmc.Deviance	The posterior sample of the deviance (D) is also provided, with D defined as : $D = -2 \log(\prod_{ij} P(y_{ij}|\beta_j, \lambda_j, \alpha_i, W_i))$.
log_theta_latent	
	Predictive posterior mean of the probability to each species to be present on each site, transformed by log link function.
theta_latent	Predictive posterior mean of the probability to each species to be present on each site.
model_spec	Various attributes of the model fitted, including the response and model matrix used, distributional assumptions as link function, family and number of latent variables, hyperparameters used in the Bayesian estimation and mcmc, burnin and thin.

The `mcmc.` objects can be summarized by functions provided by the coda package.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Gelfand, A. E.; Schmidt, A. M.; Wu, S.; Silander, J. A.; Latimer, A. and Rebelo, A. G. (2005) Modelling species diversity through species level hierarchical modelling. *Applied Statistics*, 54, 1-20.

Latimer, A. M.; Wu, S. S.; Gelfand, A. E. and Silander, J. A. (2006) Building statistical models to analyze species distributions. *Ecological Applications*, 16, 33-50.

Ovaskainen, O., Tikhonov, G., Norberg, A., Blanchet, F. G., Duan, L., Dunson, D., Roslin, T. and Abrego, N. (2017) How to make more out of community data? A conceptual framework and its implementation as models and software. *Ecology Letters*, 20, 561-576.

See Also

[plot.mcmc](), [summary.mcmc]() [jSDM_binomial_probit]() [jSDM_binomial_logit]()

Examples

```
#===================================================
# jSDM_poisson_log()
# Example with simulated data
#===================================================

#=================
#== Load libraries
library(jSDM)

#==================
#== Data simulation

#= Number of sites
nsite <- 60
#= Number of species
nsp <- 20
#= Set seed for repeatability
seed <- 1234

#= Ecological process (suitability)
set.seed(seed)
x1 <- rnorm(nsite,0,1)
set.seed(2*seed)
x2 <- rnorm(nsite,0,1)
X <- cbind(rep(1,nsite),x1,x2)
np <- ncol(X)
set.seed(3*seed)
W <- cbind(rnorm(nsite,0,1),rnorm(nsite,0,1))
n_latent <- ncol(W)
l.zero <- 0
l.diag <- runif(2,0,1)
l.other <- runif(nsp*2-3,-1,1)
lambda.target <- matrix(c(l.diag[1],l.zero,l.other[1],
                          l.diag[2],l.other[-1]),
                        byrow=TRUE, nrow=nsp)
beta.target <- matrix(runif(nsp*np,-1,1), byrow=TRUE, nrow=nsp)
V_alpha.target <- 0.5
alpha.target <- rnorm(nsite,0,sqrt(V_alpha.target))
log.theta <- X %*% t(beta.target) + W %*% t(lambda.target) + alpha.target
theta <- exp(log.theta)
Y <- apply(theta, 2, rpois, n=nsite)

#= Site-occupancy model
# Increase number of iterations (burnin and mcmc) to get convergence
mod <- jSDM_poisson_log(# Chains
                        burnin=200,
                        mcmc=200,
                        thin=1,
                        # Response variable
                        count_data=Y,
                        # Explanatory variables
```

```
                              site_formula=~x1+x2,
                              site_data=X,
                              n_latent=n_latent,
                              site_effect="random",
                              # Starting values
                              beta_start=0,
                              lambda_start=0,
                              W_start=0,
                              alpha_start=0,
                              V_alpha=1,
                              # Priors
                              shape=0.5,
                              rate=0.0005,
                              mu_beta=0,
                              V_beta=10,
                              mu_lambda=0,
                              V_lambda=10,
                              # Various
                              seed=1234,
                              ropt=0.44,
                              verbose=1)
#' #==========
#' #== Outputs

#= Parameter estimates

## beta_j
# summary(mod$mcmc.sp$sp_1[,1:ncol(X)])
mean_beta <- matrix(0,nsp,np)
pdf(file=file.path(tempdir(), "Posteriors_beta_jSDM_log.pdf"))
par(mfrow=c(ncol(X),2))
for (j in 1:nsp) {
  mean_beta[j,] <- apply(mod$mcmc.sp[[j]][,1:ncol(X)],
                         2, mean)
  for (p in 1:ncol(X)) {
    coda::traceplot(mod$mcmc.sp[[j]][,p])
    coda::densplot(mod$mcmc.sp[[j]][,p],
      main = paste(colnames(
        mod$mcmc.sp[[j]])[p],
        ", species : ",j))
    abline(v=beta.target[j,p],col='red')
  }
}
dev.off()

## lambda_j
# summary(mod$mcmc.sp$sp_1[,(ncol(X)+1):(ncol(X)+n_latent)])
# summary(mod$mcmc.sp$sp_2[,(ncol(X)+1):(ncol(X)+n_latent)])
mean_lambda <- matrix(0,nsp,n_latent)
pdf(file=file.path(tempdir(), "Posteriors_lambda_jSDM_log.pdf"))
par(mfrow=c(n_latent*2,2))
for (j in 1:nsp) {
  mean_lambda[j,] <- apply(mod$mcmc.sp[[j]]
```

```
                                  [,(ncol(X)+1):(ncol(X)+n_latent)], 2, mean)
    for (l in 1:n_latent) {
      coda::traceplot(mod$mcmc.sp[[j]][,ncol(X)+l])
      coda::densplot(mod$mcmc.sp[[j]][,ncol(X)+l],
                       main=paste(colnames(mod$mcmc.sp[[j]])
                                  [ncol(X)+l],", species : ",j))
      abline(v=lambda.target[j,l],col='red')
    }
  }
}
dev.off()

# Species effects beta and factor loadings lambda
par(mfrow=c(1,2))
plot(beta.target, mean_beta,
     main="species effect beta",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')
plot(lambda.target, mean_lambda,
     main="factor loadings lambda",
     xlab ="obs", ylab ="fitted")
abline(a=0,b=1,col='red')

## W latent variables
par(mfrow=c(1,2))
for (l in 1:n_latent) {
  plot(W[,l],
       summary(mod$mcmc.latent[[paste0("lv_",l)]])[[1]][,"Mean"],
       main = paste0("Latent variable W_", l),
       xlab ="obs", ylab ="fitted")
  abline(a=0,b=1,col='red')
}

## alpha
# summary(mod$mcmc.alpha)
par(mfrow=c(1,3))
plot(alpha.target, summary(mod$mcmc.alpha)[[1]][,"Mean"],
     xlab ="obs", ylab ="fitted", main="site effect alpha")
abline(a=0,b=1,col='red')
## Valpha
# summary(mod$mcmc.V_alpha)
coda::traceplot(mod$mcmc.V_alpha)
coda::densplot(mod$mcmc.V_alpha)
abline(v=V_alpha.target,col='red')

## Deviance
# summary(mod$mcmc.Deviance)
plot(mod$mcmc.Deviance)

#= Predictions
# summary(mod$theta_latent)
par(mfrow=c(1,2))
plot(log.theta, mod$log_theta_latent,
     main="log(theta)",
```

```
    xlab="obs", ylab="fitted")
abline(a=0 ,b=1, col="red")
plot(theta, mod$theta_latent,
     main="Expected abundance theta",
     xlab="obs", ylab="fitted")
abline(a=0 ,b=1, col="red")
```

---

logit   *Generalized logit function*

Description

Compute generalized logit function.

Usage

```
logit(x, min = 0, max = 1)
```

Arguments

x	value(s) to be transformed
min	Lower end of logit interval
max	Upper end of logit interval

Details

The generalized logit function takes values on $[min, max]$ and transforms them to span $[-\infty, +\infty]$ it is defined as:

$$y = log(\frac{p}{(1-p)})$$
$$where$$
$$p = \frac{(x - min)}{(max - min)}$$

Value

y Transformed value(s).

Author(s)

Gregory R. Warnes <greg@warnes.net>

Examples

```
x <- seq(0,10, by=0.25)
xt <- jSDM::logit(x, min=0, max=10)
cbind(x,xt)
y <- jSDM::inv_logit(xt, min=0, max=10)
cbind(x,xt,y)
```

---

mites	*mites dataset*

Description

This example data set is composed of 70 cores of mostly Sphagnum mosses collected on the territory of the Station de biologie des Laurentides of University of Montreal, Quebec, Canada in June 1989.

The whole sampling area was 2.5 m x 10 m in size and thirty-five taxa were recognized as species, though many were not given a species name, owing to the incomplete stage of systematic knowledge of the North American Oribatid fauna.

The data set comprises the abundances of 35 morphospecies, 5 substrate and micritopographic variables, and the x-y Cartesian coordinates of the 70 sampling sites.

See Borcard et al. (1992, 1994) for details.

Usage

```
data("mites")
```

Format

A data frame with 70 observations on the following 42 variables.

Abundance of **35 Oribatid mites morphospecies** named :

Brachy a vector of integers

PHTH a vector of integers

HPAV a vector of integers

RARD a vector of integers

SSTR a vector of integers

Protopl a vector of integers

MEGR a vector of integers

MPRO a vector of integers

TVIE a vector of integers

HMIN a vector of integers

HMIN2 a vector of integers

NPRA a vector of integers

TVEL a vector of integers

ONOV a vector of integers

SUCT a vector of integers

LCIL a vector of integers

Oribatul1 a vector of integers

Ceratoz1 a vector of integers

PWIL a vector of integers

Galumna1 a vector of integers

Steganacarus2 a vector of integers

HRUF a vector of integers

Trhypochth1 a vector of integers

PPEL a vector of integers

NCOR a vector of integers

SLAT a vector of integers

FSET a vector of integers

Lepidozetes a vector of integers

Eupelops a vector of integers

Minigalumna a vector of integers

LRUG a vector of integers

PLAG2 a vector of integers

Ceratoz3 a vector of integers

Oppia.minus a vector of integers

Trimalaco2 a vector of integers

5 covariates collected on the 70 sites and their coordinates :

substrate a categorical vector indicating substrate type using a 7-level unordered factor : sph1, sph2, sph3, sph4, litter, peat and inter for interface.

shrubs a categorical vector indicating shrub density using a 3-level ordered factor : None, Few and Many

topo a categorical vector indicating microtopography using a 2-level factor: blanket or hummock

density a numeric vector indicating the substrate density (g/L)

water a numeric vector indicating the water content of the substrate (g/L)

x a numeric vector indicating first coordinates of sampling sites

y a numeric vector indicating second coordinates of sampling sites

Details

Oribatid mites (Acari: Oribatida) are a very diversified group of small (0.2-1.2 mm) soil-dwelling, mostly microphytophagous and detritivorous arthropods. A well aerated soil or a complex substrate like Sphagnum mosses present in bogs and wet forests can harbour up to several hundred thousand individuals per square metre.

Local assemblages are sometimes composed of over a hundred species, including many rare ones. This diversity makes oribatid mites an interesting target group to study community-environment relationships at very local scales.

Source

Pierre Legendre

References

Borcard, D.; Legendre, P. and Drapeau, P. (1992) Partialling out the spatial component of ecological variation. *Ecology* 73: 1045-1055.

Borcard, D. and Legendre, P. (1994) Environmental control and spatial structure in ecological communities: an example using Oribatid mites (Acari, Oribatei). *Environmental and Ecological Statistics* 1: 37-61.

Borcard, D. and Legendre, P. (2002) All-scale spatial analysis of ecological data by means of principal coordinates of neighbour matrices. *Ecological Modelling* 153: 51-68.

Examples

```
data(mites, package="jSDM")
head(mites)
```

---

mosquitos	*mosquitos dataset*

Description

Presence or absence at 167 sites of 16 species that constitute the aquatic faunal community studied, 13 covariates collected at these sites and their coordinates.

Usage

```
data("mosquitos")
```

Format

A data frame with 167 observations on the following 31 variables :

16 aquatic species including larvae of four mosquito species (all potential vectors of human disease), which presence on sites is indicated by a 1 and absence by a 0 :

Culex_pipiens_sl a binary vector (mosquito species)

Culex_modestus a binary vector (mosquito species)

Culiseta_annulata a binary vector (mosquito species)

Anopheles_maculipennis_sl a binary vector (mosquito species)

waterboatmen__Corixidae a binary vector

diving_beetles__Dysticidae a binary vector

damselflies__Zygoptera a binary vector

swimming_beetles__Haliplidae a binary vector

opossum_shrimps__Mysidae a binary vector

ditch_shrimp__Gammarus a binary vector

beetle_larvae__Coleoptera a binary vector

`dragonflies__Anisoptera` a binary vector

`mayflies__Ephemeroptera` a binary vector

`newts__Pleurodelinae` a binary vector

`fish` a binary vector

`saucer_bugs__Ilyocoris` a binary vector

   13 covariates collected on the 167 sites and their coordinates :

`depth__cm` a numeric vector corresponding to the water depth in cm recorded as the mean of the depth at the edge and the centre of each dip site

`temperature__C` a numeric vector corresponding to the temperature in °C

`oxidation_reduction_potential__Mv` a numeric vector corresponding to the redox potential of the water in millivolts (mV)

`salinity__ppt` a numeric vector corresponding to the salinity of the water in parts per thousand (ppt)

   High-resolution digital photographs were taken of vegetation at the edge and centre dip points and the presence or absence of different vegetation types at each dipsite was determined from these photographs using field guides :

`water_crowfoot__Ranunculus` a binary vector indicating presence on sites by a 1 and absence by a 0 of the plant species Ranunculus aquatilis which common name is water-crowfoot

`rushes__Juncus_or_Scirpus` a binary vector indicating presence on sites by a 1 and absence by a 0 of rushes from the Juncus or Scirpus genus

`filamentous_algae` a binary vector indicating presence on sites by a 1 and absence by a 0 of filamentous algae

`emergent_grass` a binary vector indicating presence on sites by a 1 and absence by a 0 of emergent grass

`ivy_leafed_duckweed__Lemna_trisulca` a binary vector indicating presence on sites by a 1 and absence by a 0 of ivy leafed duckweed of species Lemna trisulca

`bulrushes__Typha` a binary vector indicating presence on sites by a 1 and absence by a 0 of bulrushes from the Typha genus

`reeds_Phragmites` a binary vector indicating presence on sites by a 1 and absence by a 0 of reeds from the Phragmites genus

`marestail__Hippuris` a binary vector indicating presence on sites by a 1 and absence by a 0 of plants from the Hippuris genus known as mare's-tail

`common_duckweed__Lemna_minor` a binary vector indicating presence on sites by a 1 and absence by a 0 of common duckweed of species Lemna minor

`x` a numeric vector of first coordinates corresponding to each site

`y` a numeric vector of second coordinates corresponding to each site

Source

Wilkinson, D. P.; Golding, N.; Guillera-Arroita, G.; Tingley, R. and McCarthy, M. A. (2018) A comparison of joint species distribution models for presence-absence data. *Methods in Ecology and Evolution*.

Examples

```
data(mosquitos, package="jSDM")
head(mosquitos)
```

---

plot_associations   *plot_associations plot species-species associations*

Description

plot_associations plot species-species associations

Usage

```
plot_associations(
  R,
  radius = 5,
  main = NULL,
  circleBreak = FALSE,
  top = 10L,
  occ = NULL,
  env_effect = NULL,
  cols_association = c("#FF0000", "#BF003F", "#7F007F", "#3F00BF", "#0000FF"),
  cols_occurrence = c("#BEBEBE", "#8E8E8E", "#5F5F5F", "#2F2F2F", "#000000"),
  cols_env_effect = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A", "#66A61E", "#E6AB02",
    "#A6761D", "#666666"),
  lwd_occurrence = 1,
  species_order = "abundance",
  species_indices = NULL
)
```

Arguments

R	matrix of correlation R
radius	circle's radius
main	title
circleBreak	circle break or not
top	number of top negative and positive associations to consider
occ	species occurence data
env_effect	environmental species effects β
cols_association	
	color gradient for association lines
cols_occurrence	
	color gradient for species
cols_env_effect	
	color gradient for environmental effect

`lwd_occurrence`   lwd for occurrence lines

`species_order`   order species according to :

`"abundance"`	their mean abundance at sites by default)
`"frequency"`	the number of sites where they occur
`"main environmental effect"`	their most important environmental coefficients

species_indices

      indices for sorting species

Details

After fitting the jSDM with latent variables, the **fullspecies residual correlation matrix** : $R = (R_{ij})$ with $i = 1, \ldots, n_{species}$ and $j = 1, \ldots, n_{species}$ can be derived from the covariance in the latent variables such as : can be derived from the covariance in the latent variables such as :

$$
\begin{aligned}
\Sigma_{ij} &= \lambda_i . \lambda'_j + 1 && \text{if i=j} \\
&= \lambda_i . \lambda'_j && \text{else,}
\end{aligned}
$$

this function represents the correlations computed from covariances :

$$
R_{ij} = \frac{\Sigma_{ij}}{\sqrt{\Sigma_i i \Sigma_j j}}
$$

.

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Pichler M. and Hartig F. (2021) A new method for faster and more accurate inference of species associations from big community data.
Methods in Ecology and Evolution, 12, 2159-2173 (doi:10.1111/2041-210X.13687).

See Also

jSDM-package get_residual_cor
jSDM_binomial_probit jSDM_binomial_probit_long_format
jSDM_binomial_probit_sp_constrained jSDM_binomial_logit jSDM_poisson_log

Examples

```
library(jSDM)
# frogs data
data(mites, package="jSDM")
# Arranging data
PA_mites <- mites[,1:35]
# Normalized continuous variables
```

```
Env_mites <- cbind(mites[,36:38], scale(mites[,39:40]))
colnames(Env_mites) <- colnames(mites[,36:40])
Env_mites <- as.data.frame(Env_mites)
# Parameter inference
# Increase the number of iterations to reach MCMC convergence
mod <- jSDM_poisson_log(# Response variable
                        count_data=PA_mites,
                        # Explanatory variables
                        site_formula = ~  water + topo + density,
                        site_data = Env_mites,
                        n_latent=2,
                        site_effect="random",
                        # Chains
                        burnin=100,
                        mcmc=100,
                        thin=1,
                        # Starting values
                        alpha_start=0,
                        beta_start=0,
                        lambda_start=0,
                        W_start=0,
                        V_alpha=1,
                        # Priors
                        shape=0.5, rate=0.0005,
                        mu_beta=0, V_beta=10,
                        mu_lambda=0, V_lambda=10,
                        # Various
                        seed=1234, verbose=1)
# Calcul of residual correlation between species
R <- get_residual_cor(mod)$cor.mean
plot_associations(R, circleBreak = TRUE, occ = PA_mites, species_order="abundance")
# Average of MCMC samples of species enrironmental effect beta except the intercept
env_effect <- t(sapply(mod$mcmc.sp,
                       colMeans)[grep("beta_", colnames(mod$mcmc.sp[[1]]))[-1],])
colnames(env_effect) <-  gsub("beta_", "", colnames(env_effect))
plot_associations(R, env_effect = env_effect, species_order="main env_effect")
```

---

plot_residual_cor	*Plot the residual correlation matrix from a latent variable model (LVM).*

Description

Plot the posterior mean estimator of residual correlation matrix reordered by first principal component using [corrplot](#) function from the package of the same name.

Usage

```
plot_residual_cor(
  mod,
```

```
    prob = NULL,
    title = "Residual Correlation Matrix from LVM",
    diag = F,
    type = "lower",
    method = "color",
    mar = c(1, 1, 3, 1),
    tl.srt = 45,
    tl.cex = 0.5,
    ...
)
```

Arguments

mod	An object of class `"jSDM"`.
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". If `prob=0.95` is specified only significant correlations, whose 95% HPD interval does not contain zero, are represented. Defaults to `prob=NULL` to represent all correlations significant or not.
title	Character, title of the graph.
diag	Logical, whether display the correlation coefficients on the principal diagonal.
type	Character, "full" (default), "upper" or "lower", display full matrix, lower triangular or upper triangular matrix.
method	Character, the visualization method of correlation matrix to be used. Currently, it supports seven methods, named "circle" (default), "square", "ellipse", "number", "pie", "shade" and "color".
mar	See [par](#)
tl.srt	Numeric, for text label string rotation in degrees, see [text](#).
tl.cex	Numeric, for the size of text label (variable names).
...	Further arguments passed to [corrplot](#) function

Value

Displays a reordered correlation matrix

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

References

Taiyun Wei and Viliam Simko (2017). R package "corrplot": Visualization of a Correlation Matrix (Version 0.84)

Warton, D. I.; Blanchet, F. G.; O'Hara, R. B.; O'Hara, R. B.; Ovaskainen, O.; Taskinen, S.; Walker, S. C. and Hui, F. K. C. (2015) So Many Variables: Joint Modeling in Community Ecology. *Trends in Ecology & Evolution*, 30, 766-779.

See Also

    corrplot jSDM-package jSDM_binomial_probit
    jSDM_binomial_logit jSDM_poisson_log

Examples

```
library(jSDM)
# frogs data
data(frogs, package="jSDM")
# Arranging data
PA_frogs <- frogs[,4:12]
# Normalized continuous variables
 Env_frogs <- cbind(scale(frogs[,1]),frogs[,2],scale(frogs[,3]))
 colnames(Env_frogs) <- colnames(frogs[,1:3])
# Parameter inference
# Increase the number of iterations to reach MCMC convergence
mod<-jSDM_binomial_probit(# Response variable
                          presence_data = PA_frogs,
                          # Explanatory variables
                          site_formula = ~.,
                          site_data = Env_frogs,
                          n_latent=2,
                          site_effect="random",
                          # Chains
                          burnin=100,
                          mcmc=100,
                          thin=1,
                          # Starting values
                          alpha_start=0,
                          beta_start=0,
                          lambda_start=0,
                          W_start=0,
                          V_alpha=1,
                          # Priors
                          shape=0.1, rate=0.1,
                          mu_beta=0, V_beta=1,
                          mu_lambda=0, V_lambda=1,
                          # Various
                          seed=1234, verbose=1)
# Representation of residual correlation between species
plot_residual_cor(mod)
plot_residual_cor(mod, prob=0.95)
```

---

    predict.jSDM   *Predict method for models fitted with jSDM*

Description

Prediction of species probabilities of occurrence from models fitted using the jSDM package

Usage

```
## S3 method for class 'jSDM'
predict(
  object,
  newdata = NULL,
  Id_species,
  Id_sites,
  type = "mean",
  probs = c(0.025, 0.975),
  ...
)
```

Arguments

object	An object of class "jSDM".
newdata	An optional data frame in which explanatory variables can be searched for prediction. If omitted, the adjusted values are used.
Id_species	An vector of character or integer indicating for which species the probabilities of presence on chosen sites will be predicted.
Id_sites	An vector of integer indicating for which sites the probabilities of presence of specified species will be predicted.
type	Type of prediction. Can be :

"mean"	for predictive posterior mean.
"quantile"	for producing sample quantiles from the predictive posterior, corresponding to the given probabilities (see probs argument).
"posterior"	for the full predictive posterior for each prediction.

	Using "quantile" or "posterior" might lead to memory problem depending on the number of predictions and the number of samples for the jSDM model's parameters.
probs	Numeric vector of probabilities with values in [0,1], used when type="quantile".
...	Further arguments passed to or from other methods.

Value

Return a vector for the predictive posterior mean when type="mean", a data-frame with the mean and quantiles when type="quantile" or an mcmc object (see coda package) with posterior distribution for each prediction when type="posterior".

Author(s)

Ghislain Vieilledent <ghislain.vieilledent@cirad.fr>

Jeanne Clément <jeanne.clement16@laposte.net>

See Also

jSDM-package jSDM_binomial_logit jSDM_binomial_probit jSDM_poisson_log

Examples

```
library(jSDM)
# frogs data
data(frogs, package="jSDM")
# Arranging data
PA_frogs <- frogs[,4:12]
# Normalized continuous variables
Env_frogs <- cbind(scale(frogs[,1]),frogs[,2],scale(frogs[,3]))
colnames(Env_frogs) <- colnames(frogs[,1:3])
# Parameter inference
# Increase the number of iterations to reach MCMC convergence
mod<-jSDM_binomial_probit(# Response variable
                          presence_data=PA_frogs,
                          # Explanatory variables
                          site_formula = ~.,
                          site_data = Env_frogs,
                          n_latent=2,
                          site_effect="random",
                          # Chains
                          burnin=100,
                          mcmc=100,
                          thin=1,
                          # Starting values
                          alpha_start=0,
                          beta_start=0,
                          lambda_start=0,
                          W_start=0,
                          V_alpha=1,
                          # Priors
                          shape=0.5, rate=0.0005,
                          mu_beta=0, V_beta=10,
                          mu_lambda=0, V_lambda=10,
                          # Various
                          seed=1234, verbose=1)

# Select site and species for predictions
## 30 sites
Id_sites <- sample.int(nrow(PA_frogs), 30)
## 5 species
Id_species <- sample(colnames(PA_frogs), 5)

# Predictions
theta_pred <- predict(mod,
                      Id_species=Id_species,
                      Id_sites=Id_sites,
                      type="mean")
hist(theta_pred, main="Predicted theta with simulated covariates")
```

Index