Package 'BayesComm'

January 20, 2025

Type Package

Title Bayesian Community Ecology Analysis

Version 0.1-2

Date 2015-07-20

Author Nick Golding and David J. Harris

Maintainer Nick Golding

<nick.golding.research@gmail.com>

Description Bayesian multivariate binary (probit) regression models for analysis of ecological communities.

License GPL (>= 2)

Imports Rcpp (>= 0.11.6), abind, coda, mvtnorm

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-07-23 16:32:57

Contents

BayesComm-package *Bayesian community ecology analysis*

Description

BayesComm fits Bayesian multivariate binary (probit) regression models for analysis of ecological communities. These models can be used to make inference about underlying inter-species interactions in communities and to separate the effects of environmental covariates and inter-species interactions on community assembly. This package accompanies the paper (in preparation) by Golding et al. (2013) and is based on a model described by Edwards and Allenby (2003).

Details

Package:	BayesComm
Type:	Package
Version:	0.1-1
Date:	2014-03-07
License:	GPL (>=2)

BayesComm models take as input a matrix of species presence/absence records and optionally a matrix of environmental covariates. `BC` is the main function for setting up models. It is a wrapper function to `BCfit` and returns a `bayescomm` object. `bayescomm` objects have associated `summary`, `plot`, `print`, `window` and `residuals` functions. Functions are also provided to calculate Deviance Information Criteria (`DIC`) and run a deviance partitioning procedure on model outputs (`devpart`).

Full details of formulation of the model are given in Golding et al. (2013).

Author(s)

Nick Golding <nick.golding@zoo.ox.ac.uk> \& Dave Harris

References

Golding (2013) Mapping and understanding the distributions of potential vector mosquitoes in the UK: New methods and applications. (Chapter 3) http://dx.doi.org/10.6084/m9.figshare.767289

Edwards, Y., Allenby, G. (2003) Multivariate analysis of multiple response data. *Journal of Marketing Research*, 40 (3) 321-34.

See Also

`BC`, `BCfit`, `window.bayescomm`, `plot.bayescomm`, `print.bayescomm`, `summary.bayescomm`, `residuals.bayescomm`, `DIC`, `devpart`,

BC *Run a BayesComm model*

Description

BC is the main function for running BayesComm models. It is a wrapper function for BCfit; it checks inputs, sets up the model types and specifies a number of default BCfit settings.

Usage

```
BC(Y, X = NULL, model = "null", covlist = NULL, condition = NULL, its = 100, ...)
```

Arguments

Y	matrix of species presence/absence data
X	matrix of environmental covariates
model	type of model to run
covlist	optional list of which covariates to assign to each species
condition	matrix of conditioning variables
its	number of iterations for sampling phase
...	further arguments to pass to BCfit

Details

Y must be a matrix with records as rows and species as columns and X a matrix with records as rows and covariates as columns. model must be one of: "null" (intercept only), "environment" (intercept & covariates), "community" (intercept & community matrix) or "full" (intercept, covariates & community matrix). covlist must have the same length as the number of species with, each element a vector of column indices for X. covlist defaults to NULL, which includes all covariates for all species. For more details of arguments for model fitting see BCfit. condition is an optional matrix of conditioning variables. These are fitted in the same way as X but are not removed in null and community models.

Value

An object of class bayescomm containing the model call and parameter chains which can be viewed and manipulated using window, plot, print and summary.

See Also

BCfit

Examples

```
# create fake data
n <- 100
nsp <- 4
k <- 3

X <- matrix(c(rep(1, n), rnorm(n * k)), n)  # covariate matrix
W <- matrix(rnorm(nsp * nsp), nsp)
W <- W %*% t(W) / 2  # true covariance matrix
B <- matrix(rnorm(nsp * (k + 1), 0, 3), nsp)  # true covariates
mu <- apply(B, 1, function(b, x) x %*% b, X)  # true mean
e <- matrix(rnorm(n * nsp), n) %*% chol(W)  # true e
z <- mu + e  # true z
Y <- ifelse(z > 0, 1, 0)  # true presence/absence

# run BC (after removing intercept column from design matrix)
m1 <- BC(Y, X[, -1], model = "full", its = 100)
```

BCfit *Fit a BayesComm model*

Description

`BCfit` is the workhorse function for the BayesComm model. It is highly recommended to use the wrapper function `BC` which checks inputs and sets up different model types and initial values. `BCfit` arguments can be accessed through `BC` using the `...` argument.

Usage

```
BCfit(y, X, covlist, R, z, mu, updateR, iters, thin = 1, burn = 0,
priW = c(nrow(z) + 2 * ncol(z), 2 * ncol(z)), verbose = 0)
```

Arguments

`y`	matrix of species presence/absence data
`X`	matrix of environmental covariates
`covlist`	optional list of which covariates to assign to each species
`R`	initial values for correlation matrix
`z`	initial values for z
`mu`	initial values for mu
`updateR`	logical; if true the correlation matrix is updated, if false it is fixed at `R`
`iters`	total number of iterations
`thin`	amount to thin the posterior chains. Defaults to 1 (no thinning)
`burn`	number of iterations to discard at the beginning of the chain
`priW`	prior specification for correlation matrix W
`verbose`	how often to print updates to the console. `0` for no updates. `1` for updates every `thin` iterations after burnin. `2` for updates every iteration.

Details

priW specifies the inverse Wishart prior on the unknown and unidentifiable covariance matrix W from which the correlation matrix R is derived. priW is a vector of length two, the first element specifies the degrees of freedom, the second element is multiplied by an identity matrix to form the scale matrix. The default for priW is c(n + 2p, 2p), where n is the number of records and p is the number of species in the community; this therefore forms the prior: iW(n + 2p, 2pI). This prior was determined to exert minimal influence on the posterior of R whilst limiting dependence of R on the unidentifiable variance parameters of W.

For further details on how to specify Y, X and covlist see BC.

Value

A list containing elements:

R	samples from posteriors of the correlation matrix
B	samples from posteriors of regression coefficients (a list of matrices)
z	samples from posteriors of latent variables z

See Also

BC

---

devpart *Deviance partitioning*

Description

Runs a deviance partitioning procedure on a set of four bayescomm objects.

Usage

```
devpart(null, environment, community, full)
```

Arguments

null	a bayescomm object containing a 'null' model
environment	a bayescomm object containing an 'environment' model
community	a bayescomm object containing a 'community' model
full	a bayescomm object containing a 'full' model

Details

The deviance partitioning procedure determines the proportion of the null deviance explained by each of the other three model types. The four model types are those created by BC.

Value

A list containing elements

devpart	matrix containing the proportion of the null deviance explained by each model for each species
null	a matrix containing the mean and 95% credible intervals for the deviance for each species in the null model
environment	a matrix containing the mean and 95% credible intervals for the deviance for each species in the evironment model
community	a matrix containing the mean and 95% credible intervals for the deviance for each species in the community model
full	a matrix containing the mean and 95% credible intervals for the deviance for each species in the full model

See Also

BC

Examples

```
# create fake data
n <- 100
nsp <- 4
k <- 3

X <- matrix(c(rep(1, n), rnorm(n * k)), n)  # covariate matrix
W <- matrix(rnorm(nsp * nsp), nsp)
W <- W %*% t(W) / 2  # true covariance matrix
B <- matrix(rnorm(nsp * (k + 1), 0, 3), nsp)  # true covariates
mu <- apply(B, 1, function(b, x) x %*% b, X)  # true mean
e <- matrix(rnorm(n * nsp), n) %*% chol(W)  # true e
z <- mu + e  # true z
Y <- ifelse(z > 0, 1, 0)  # true presence/absence

# run BC (after removing intercept column from design matrix)
null <- BC(Y, X[, -1], model = "null", its = 100)
comm <- BC(Y, X[, -1], model = "community",its = 100)
envi <- BC(Y, X[, -1], model = "environment", its = 100)
full <- BC(Y, X[, -1], model = "full", its = 100)

devpart(null, envi, comm, full)
```

DIC *Deviance Information Criterion*

Description

Calculates Deviance Information Criteria for bayescomm objects.

Usage

```
DIC(BC)
```

Arguments

BC	a bayescomm object

References

Spiegelhalter, D.J.. Best, N.G., Carlin, B.P., van der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society, Series B*, 64 (4): 583-639.

See Also

BC

Examples

```
m1 <- example(BC)[[1]]
DIC(m1)
```

---

plot.bayescomm *Plot bayescomm parameter chains*

Description

plot.bayescomm creates summary plots of a subset of the parameter chains in a bayescomm object using the coda package.

Usage

```
## S3 method for class 'bayescomm'
plot(x, chain, ...)
```

Arguments

x	a bayescomm object
chain	a character string of the parameter chains to plot
...	further arguments to pass to plot.mcmc

Details

chain should be one of 'R' (for correlation coefficients) or 'B$sp' where sp is the species of interest (for regression coefficients).

See Also

BC, plot.mcmc

Examples

```
m1 <- example(BC)[[1]]
plot(m1, 'R')
plot(m1, 'B$sp1')
```

`predict.bayescomm` *Function to make predictions at new locations*

Description

For each set of parameter values sampled by the model (including values of Z), simulate the occurrence probabilities for each species at each new location.

Usage

```
## S3 method for class 'bayescomm'
predict(object, newdata, ...)
```

Arguments

`object`	A bayescomm object
`newdata`	A data.frame with the same columns as X from the original BC model
`...`	Further arguments passed to or from other methods.

Value

An array of occurrence probabilities. Rows index locations. Columns index species. Slices index MCMC samples.

Author(s)

David J. Harris (http://davharris.github.io)

Examples

```
# load model from first example
m1 <- example(BC)[[1]]

# use the first five sites of the training data as newdata
newdata <- X[1:5, -1]

# get predictions
prob <- predict(m1, newdata)
```

print.bayescomm *Print a bayescomm object*

Description

print.bayescomm prints a brief summary of a bayescomm object.

Usage

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

Arguments

x	a bayescomm object
...	further arguments to pass to print

See Also

BC

Examples

```
m1 <- example(BC)[[1]]
print(m1)
m1
```

residuals.bayescomm *Extract bayescomm model residuals*

Description

residuals.bayescomm extracts model residuals from a bayescomm object. Residuals are calculated based on the mean of the posterior probability of presence.

Usage

```
## S3 method for class 'bayescomm'
residuals(object, ...)
```

Arguments

object	a bayescomm object
...	other arguments

See Also

BC

Examples

```
m1 <- example(BC)[[1]]
m1.res <- residuals(m1)
```

---

summary.bayescomm *Summarise bayescomm parameter chains*

Description

summary.bayescomm creates summaries of a subset of the parameter chains in a bayescomm object using the coda package.

Usage

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

Arguments

object	a bayescomm object
chain	a character string of the parameter chains to plot
...	further arguments to pass to summary.mcmc

Details

chain should be one of 'R' (for correlation coefficients) or 'B$sp' where sp is the species of interest (for regression coefficients).

See Also

BC, summary.mcmc

Examples

```
m1 <- example(BC)[[1]]
summary(m1, 'R')
summary(m1, 'B$sp1')
```

window.bayescomm *Window bayescomm objects*

Description

window.bayescomm is window function for bayescomm objects, it calls window.mcmc from the coda package. Parameter chains are subsetted by start and end and thinned by thin.

Usage

```
## S3 method for class 'bayescomm'
window(x, start = NULL, end = NULL, thin = 1, ...)
```

Arguments

x	a bayescomm object
start	start iteration
end	end iteration
thin	thinning interval
...	further arguments to pass to window.mcmc

Details

If start = NULL (default) the start is taken as the first iteration. If end = NULL (default) the end is taken as the final iteration. If thin = 1 (default) all iterations within the window are retained.

Value

A bayescomm object with windowed parameter chains.

See Also

BC, window.mcmc

Examples

```
m1 <- example(BC)[[1]]
m2 <- window(m1, 51, 150, 10)
```

Index