Package 'NeMO'

December 1, 2025

Title Nested eDNA Metabarcoding Occupancy

Version 1.0.0

Description A Bayesian framework for modelling multispecies site occupancy from eDNA metabarcoding data.

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

RdMacros Rdpack

VignetteBuilder knitr

LazyData true

Depends R (>= 3.5.0), HDInterval, R2jags, stats

Imports abind, magrittr, methods, Rdpack, utils

Suggests ggridges, gridExtra, knitr, RColorBrewer, rmarkdown, testthat (>= 3.0.0), tidyverse

Collate 'classes.R' 'Nemodel.R' 'WAIC.R' 'covarray.R' 'data.R' 'min_resources.R'

Config/testthat/edition 3

URL https://github.com/bastien-mace/NeMO

BugReports https://github.com/bastien-mace/NeMO/issues

R topics documented:

NeMO-package *Nested eDNA Metabarcoding Occupancy*

Description

This package provides a flexible Bayesian framework for modeling multispecies site occupancy from eDNA metabarcoding data. It supports diverse study designs, including individually indexed or pooled PCR replicates, enabling inference from either presence/absence or sequence read count data. It allows covariate implementation and subsequent model comparison using the Watanabe-Akaike Information Criterion. Additionally, it makes possible the estimate of the minimum resources required to confidently detect species.

Author(s)

Bastien Macé <bastien.mace.49@gmail.com>

covarray *Format Covariates for NeMO*

Description

This function facilitates the integration of covariates into the **NeMO** modelling framework by formatting them into arrays compatible with Nemodel.

Usage

```
covarray(
  protocol = c("PCR_rep", "seq_read", "PCR_rep_seq_read"),
  array,
  cov_list = list(list(cov_data = NULL, level = "psi", dimension = "species"))
)
```

Arguments

`protocol`	Character string. Specifies the modelling protocol to be used in Nemodel. Options are: • `'PCR_rep'`: Requires a 5D (N × I × J × K × C) presence/absence array • `'seq_read'`: Requires a 4D (N × I × J × C) sequence read count array • `'PCR_rep_seq_read'`: Requires a 5D (N × I × J × K × C) sequence read count array where: – N: Number of species – I: Number of sites – J: Number of samples – K: Number of PCR replicates – C: Number of campaigns
`array`	Input data array to be used in Nemodel. Its required dimensionality depends on the selected `protocol` argument.

`cov_list` A list of sublists. Each sublist represent a covariate. Sublists must be attributed a unique name and contain the following components:

- `cov_data`: Array of covariate values. These values should be either Boolean for categorical/semi-quantitative predictors or standardised values for quantitative predictors.
- `level`: Character string. Specifies the hierarchical level where the covariate applies (*e.g.*, `'psi'`, `'theta'`, `'p'`, `'phi'`).
- `dimension`: Character string. Specifies the dimensions associated to the covariate. Acceptable values include:
  - Single dimensions: `'species'`, `'site'`, `'sample'`, `'replicate'`, or `'campaign'`.
  - Combined dimensions: Combinations of the above, except `'species'` (always stands alone). In combinations, terms must be separated by an underscore `'_'`, following this order: 1) `'site'` 2) `'sample'` 3) `'replicate'` 4) `'campaign'` (*i.e.*, `'site_campaign'`).

Details

The `cov_data` array in each sublist must align with the specified `dimension`. For instance, if `dimension = 'site_sample_campaign'`, the covariate array should be structured such that dimension 1 corresponds to sites, dimension 2 corresponds to samples, and dimension 3 corresponds to campaigns.

Value

A structured class object with eight slots, organizing covariates across hierarchical levels for direct implementation in the NeMO modelling framework:

- `psi_cov`: Covariates for spatial/methodological/temporal components of ψ.
- `psi_cov_sp`: Species covariates of ψ.
- `theta_cov`: Covariates for spatial/methodological/temporal components of θ.
- `theta_cov_sp`: Species covariates of θ.
- `p_cov`: Covariates for spatial/methodological/temporal components of p.
- `p_cov_sp`: Species covariates of p.
- `phi_cov`: Covariates for spatial/methodological/temporal components of φ.
- `phi_cov_sp`: Species covariates of φ.

See Also

`Nemodel`

Examples

```
# Load input array
data(fish_PCR_rep)

# Load covariate data (Distance to sea)
data(distance_cov)

# Build the covariate array applied to sites on the psi level
covarray(protocol = 'PCR_rep',
```

```
            array = fish_PCR_rep,
            cov_list = list(Distance = list(cov_data = distance_cov$Distance,
                                            level = 'psi',
                                            dimension = 'site')))
```

`distance_cov` *Distance to Sea Covariate Data*

Description

A dataframe containing the standardised distance to the sea for each of the 10 sites in the fish datasets.

Usage

```
data(distance_cov)
```

Format

A dataframe with 10 rows and 1 column:

Distance Standardised distance to the sea for each site

The `$Distance` column can directly be used in covarray.

See Also

`covarray`

`fish_PCR_rep` *Presence/Absence Data with Individually Indexed PCR Replicates*

Description

A 5D array containing presence/absence data for fish species across sites, samples, PCR replicates, and campaigns.

Usage

```
data(fish_PCR_rep)
```

Format

A 5-dimensional array with:

Dimension 1 Species (10)

Dimension 2 Sites (10)

Dimension 3 Samples (2)

Dimension 4 PCR Replicates (5)

Dimension 5 Campaigns (1)

This dataset can directly be used in Nemodel with the `'PCR_rep'` protocol.

See Also

Nemodel

`fish_PCR_rep_seq_read` *Sequence Read Count Data with Individually Indexed PCR Replicates*

Description

A 5D array containing sequence read count data for fish species across sites, samples, PCR replicates, and campaigns.

Usage

```
data(fish_PCR_rep_seq_read)
```

Format

A 5-dimensional array with:

Dimension 1 Species (10)

Dimension 2 Sites (10)

Dimension 3 Samples (2)

Dimension 4 PCR Replicates (5)

Dimension 5 Campaigns (1)

This dataset can directly be used in Nemodel with the `'PCR_rep_seq_read'` protocol.

See Also

Nemodel

`fish_seq_read` *Sequence Read Count Data with Pooled PCR Replicates*

Description

A 4D array containing sequence read count data for fish species across sites, samples, and campaigns.

Usage

```
data(fish_seq_read)
```

Format

A 4-dimensional array with:

Dimension 1 Species (10)

Dimension 2 Sites (10)

Dimension 3 Samples (2)

Dimension 4 Campaigns (1)

This dataset can directly be used in Nemodel with the `'seq_read'` protocol.

See Also

`Nemodel`

`min_resources` *Calculate Minimum Resource Requirements for Reliable Species Detection*

Description

This function calculates the minimum resource requirements, *i.e.* samples (J_{min}), PCR replicates (K_{min}), and sequencing depth (M_{min}), needed to confidently detect species when present with a specified confidence level. This estimation relies on established probabilistic models, enabling rigorous resource planning for ecological studies.

Usage

```
min_resources(model, resources = c("J"), conf = 0.95)
```

Arguments

`model`	A fitted Nemodel object.
`resources`	A character vector. Specifies the resource types to compute. Acceptable values include `'J'` (minimum samples), `'K'` (minimum PCR replicates), and `'M'` (minimum sequencing depth). Multiple resource types can be specified simultaneously (*e.g.*, `resources = c('J', 'K', 'M')`).
`conf`	Numeric. Confidence level, or probability to achieve to detect species when present (default: 0.95).

Details

The function follows established equations from McArdle (1990) and probabilistic models to compute:

- **Minimum number of samples (J_{min})**: The function calculates the minimum number of samples required to ensure that the probability of missing species DNA in a sample is below 0.05 with 95% confidence (can be computed for any `protocol`):

$$J_{min} \geq \frac{\ln(0.05)}{\ln(1 - \theta_{nijc})}$$

where θ_{nijc} is the probability of DNA collection for species n at site i in sample j during campaign c.

- **Minimum number of PCR replicates (K_{min})**: The function calculates the minimum number of PCR replicates required to ensure that the probability of missing species DNA in a PCR replicate is below 0.05 with 95% confidence (requires a Nemodel object built with `'PCR_rep'` or `'PCR_rep_seq_read'` protocol):

$$K_{min} \geq \frac{\ln(0.05)}{\ln(1 - p_{nijkc})}$$

where p_{nijkc} is the probability of DNA amplification for species n at site i in sample j in replicate k during campaign c.

- **Minimum sequencing depth (M_{min})**: Following the multinomial theorem, the function calculates the minimum sequencing depth required to ensure that the probability of missing species DNA in a sample or in a PCR replicate is below 0.05 (requires a Nemodel object built with `'seq_read'` or `'PCR_rep_seq_read'` protocol):

$$M_{min} \geq \frac{\ln(0.05)}{\ln(1 - \pi_{nijkc})}$$

where π_{nijkc} is the relative sequence read count for species n at site i in sample j in replicate k (except for `'seq_read'` protocol) during campaign c.
The function automatically performs these computations for precise and efficient resource estimation.

Value

A structured class object with three slots:

- `J_min`: Minimum number of samples required to confidently detect species when present across sites.
- `K_min`: Minimum number of replicates required to confidently detect species when present across sites and samples.
- `M_min`: Minimum sequencing depth required to confidently detect species when present across sites, samples and replicates (except for `'seq_read'` protocol).

Each slot contains 3 arrays, corresponding to the median values and the lower and upper bounds of the 95% highest density interval (HDI) of MCMC samples.

See Also

`Nemodel()`, `covarray()`

Examples

```
# Load fitted model
data(model_PCR_rep)

# Calculate the minimum number of samples and PCR replicates required to
# confidently detect species when present with 95% confidence
```

```
min_resources(model = model_PCR_rep,
              resources = c('J', 'K'))
```

`model_PCR_rep` *Example model output using the* `PCR_rep` *protocol*

Description

This dataset contains the output from running Nemodel on fish_PCR_rep integrating distance_cov as covariate through covarray, using the `PCR_rep` protocol and the following parameters: `nb_iterations = 300`, `nb_burnin = 150`, `nb_thinning = 1`, `nb_chains = 2`.

Usage

```
data(model_PCR_rep)
```

Format

A structured class object (Nemodel object) with six slots:

model A fitted model in `rjags` format

protocol The modelling protocol used (`PCR_rep`)

array The model input data array (`fish_PCR_rep`)

covariates The covariate list implemented in the model (built from `distance_cov`)

names A list containing species, site, sample, replicate, campaign and estimates' names

loglik `TRUE`: log-likelihood was stored for model comparison

See Also

`Nemodel`, `covarray`, `fish_PCR_rep`, `distance_cov`

`Nemodel` *Bayesian Multispecies Occupancy Model for eDNA Data.*

Description

This function is the core of the **NeMO** package, designed for Bayesian modelling of multispecies occupancy in eDNA metabarcoding studies. It provides flexibility to fit different modelling protocols, accommodating various study designs and data structures. The function uses the **R2jags** package for MCMC sampling.

Usage

```
Nemodel(
  protocol = c("PCR_rep", "seq_read", "PCR_rep_seq_read"),
  array,
  covariates = NULL,
  name = "model",
  nb_iterations = 1000,
  nb_burnin = floor(nb_iterations/2),
  nb_thinning = max(1, floor((nb_iterations - nb_burnin)/1000)),
  nb_chains = 2,
  parallel = FALSE,
  loglik = FALSE,
  latent = NULL,
  posterior = NULL,
  tau = 1,
  rho = 1,
  lambda = 1,
  size = 1,
  prob = 0.001,
  ...
)
```

Arguments

`protocol`	Character string. Specifies the modeling protocol. Options are: • `'PCR_rep'`: Requires a 5D (N × I × J × K × C) presence/absence input array • `'seq_read'`: Requires a 4D (N × I × J × C) sequence read count input array • `'PCR_rep_seq_read'`: Requires a 5D (N × I × J × K × C) sequence read count input array where: – N: Number of species. – I: Number of sites. – J: Number of samples. – K: Number of PCR replicates. – C: Number of campaigns.
`array`	Input data array. Its required dimensionality depends on the selected `protocol` argument.
`covariates`	Optional. A preprocessed covariate list generated using covarray.
`name`	Character string specifying the filename and path for recording the BUGS language model file.
`nb_iterations`	Integer. Total number of MCMC iterations to run (default: 2000).
`nb_burnin`	Integer. Number of initial burn-in iterations (default: `floor(nb_iterations / 2)`).
`nb_thinning`	Integer. Thinning interval to reduce autocorrelation in MCMC samples (default: `max(1, floor((nb_iterations - nb_burnin) / 1000))`).
`nb_chains`	Integer. Number of independent MCMC chains to compute (default: 3).

parallel	Logical. If TRUE, enables parallel computation to speed up sampling for large datasets (default: FALSE).
loglik	Logical. If TRUE, computes and saves log-likelihoods for model comparison using WAIC (default: FALSE).
latent	Optional character vector. Specifies latent arrays (*e.g.*, 'Z', 'A', 'W', 'S', 'Y') to save.
posterior	Optional character vector. Specifies posterior distributions of key parameters (*e.g.*, 'psi', 'theta', 'p', 'pi', 'phi') to save.
tau	Numeric. Precision parameter for normal (hyper)priors (default: 1).
rho	Numeric. Shape parameter of the Gamma prior (only for 'seq_read' or 'PCR_rep_seq_read' protocols, default: 1).
lambda	Numeric. Rate parameter of the Gamma prior (only for 'seq_read' or 'PCR_rep_seq_read' protocols, default: 1).
size	Numeric. Initialization size parameter for the S array (only for 'seq_read' or 'PCR_rep_seq_read' protocols, default: 1).
prob	Numeric. Initialization probability parameter for the S array (only for 'seq_read' or 'PCR_rep_seq_read' protocols, default: 0.001).
...	Additional arguments (related to jags or jags.parallel functions from **R2jags**).

Details

This function provides flexibility for different study designs by allowing users to model occupancy using either presence/absence arrays (PCR_rep) or sequence read counts (seq_read, PCR_rep_seq_read). Covariates can be incorporated to account for external predictors. The MCMC sampling process can be customized using multiple control parameters (nb_iterations, nb_burnin, nb_thinning, etc.), and parallel computation can be enabled with parallel = TRUE. The function also allows users to store log-likelihoods, latent arrays, and posterior distributions for downstream analysis.

Value

A structured class object (Nemodel object) with five slots:

- model: The fitted model output in rjags format, with a BUGS language text file stored at the specified location (name argument).
- protocol: The modelling protocol used (*i.e.*, 'PCR_rep', 'seq_read', or 'PCR_rep_seq_read').
- array: The model input data array.
- covariates: The covariate list implemented in the model.
- names: A list containing species, site, sample, replicate, and campaign names. This slot also contains:
  - estimates_sp: Names associated with the species-level random effects.
  - estimates: Names associated with the intercepts and other random effects.
- loglik: Logical. If TRUE, log-likelihood values are stored for use in model comparison.

See Also

WAIC, covarray

Examples

```
# Load input array
data(fish_PCR_rep)

# Load covariate data (Distance to sea)
data(distance_cov)

# Build the covariate array applied to sites on the psi level
covariates <- covarray(protocol = 'PCR_rep',
                       array = fish_PCR_rep,
                       cov_list = list(Distance = list(cov_data = distance_cov$Distance,
                                                       level = 'psi',
                                                       dimension = 'site')))

# Run the 'PCR_rep' model on the input array with the distance covariate
Nemodel(protocol = 'PCR_rep',
        array = fish_PCR_rep,
        covariates = covariates)
unlink('model.txt')
```

`WAIC` *Calculate the Watanabe-Akaike Information Criterion (WAIC)*

Description

This function computes the Watanabe-Akaike Information Criterion (WAIC), also known as the Widely Applicable Information Criterion (Watanabe, 2010). This robust metric is used for model comparison and selection within Bayesian frameworks, offering a way to balance model fit and complexity. The WAIC estimates out-of-sample predictive accuracy, penalising overfitting.

Usage

```
WAIC(model)
```

Arguments

`model`	A fitted model object created with Nemodel with `loglik = TRUE` to record log-likelihoods.

Details

The WAIC is computed using the following equations, which incorporate both the mean log-likelihood and its variance:

- **Log Pointwise Predictive Density (LPPD)**:

$$LPPD = \sum_{nijkc=1}^{NIJKC} \ln\left(\frac{1}{Sim_{tot}} \sum_{sim=1}^{Sim_{tot}} e^{\log \ell_{nijkc,sim}}\right)$$

where:

- $\log \ell_{nijkc,sim}$ is the log-likelihood value calculated from simulation sim for species n at site i in sample j in replicate k (except for `'seq_read'` protocol) during campaign. c
- Sim_{tot} is the total number of simulations (MCMC samples) computed.

- **Effective Number of Parameters (p_WAIC)**:

$$p_{WAIC} = \sum_{nijkc=1}^{NIJKC} \frac{1}{Sim_{tot}} \sum_{sim=1}^{Sim_{tot}} \left(\log \ell_{nijkc,sim} - \overline{\log \ell_{nijkc}}\right)^2$$

where:

- $\log \ell_{nijkc,sim}$ is the log-likelihood value calculated from simulation sim for species n at site i in sample j in replicate k (except for `'seq_read'` protocol) during campaign. c
- $\overline{\log \ell_{nijkc}}$ is the mean log-likelihood value across all simulations for species n at site i in sample j in replicate k (except for `'seq_read'` protocol) during campaign. c
- Sim_{tot} is the total number of simulations (MCMC samples) computed.

- **WAIC Score**:

$$WAIC = -2 \times LPPD - p_{WAIC}$$

These computations are performed internally, yielding a scalar WAIC value that summarises the overall performance of the model.

Value

A structured class object containing three slots:

- `waic`: The WAIC score, quantifying the trade-off between model fit and complexity.
- `lppd`: The log-predictive density term, reflecting the model's fit.
- `p_waic`: The effective number of parameters, representing model complexity.

See Also

`Nemodel()`, `covarray()`

Examples

```
# Load fitted model
data(model_PCR_rep)

# Calculate the WAIC
WAIC(model_PCR_rep)
```

Index