# Package: EcoDiet (via r-universe)

October 22, 2024

**Type** Package

**Title** Estimating a Diet Matrix from Biotracer and Stomach Content Data

**Description** Biotracers and stomach content analyses are combined in a Bayesian hierarchical model to estimate a probabilistic topology matrix (all trophic link probabilities) and a diet matrix (all diet proportions). The package relies on the JAGS software and the 'jagsUI' package to run a Markov chain Monte Carlo approximation of the different variables.

**Version** 2.0.1

**Depends** R (>= 3.5)

**Imports** ggplot2 (>= 3.2), coda (>= 0.19), stats (>= 3.6), utils (>= 3.6), jagsUI (>= 1.5.2), ggmcmc (>= 1.1)

**Suggests** knitr, rmarkdown, devtools, testthat (>= 3.0.0)

**SystemRequirements** JAGS (>= 4.3)

**URL** https://github.com/pyhernvann/EcoDiet

**BugReports** https://github.com/pyhernvann/EcoDiet/issues

**License** GPL (>= 2)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.1

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Pierre-Yves Hernvann [cre, aut], Didier Gascuel [aut], Dorothee Kopp [aut], Marianne Robert [aut], Jerome Guitton [aut], Etienne Rivot [aut], Heloise Thero [aut]

**Maintainer** Pierre-Yves Hernvann <pierre.yves.hernvann@gmail.com>

**Repository** CRAN

**Date/Publication** 2024-03-25 20:40:11 UTC

# Contents



---

`diagnose_model` *Diagnose EcoDiet model*

---

## Description

This function operates a diagnostic of the fit EcoDiet model.

A message is printed to provide the number of variables for which the Gelman-Rubin diagnostic exceeds specific thresholds (> 1.01, > 1.05, >1.1). The list of the 10 worst variables in terms of convergence is also printed.

You need to have run the `run_model` function before using this function.

The design of this function is substantially inspired from a function with a similar objective in the MixSIAR package [(Stock et al. 2018)](https://doi.org/10.7717/peerj.5096), for which code is available online on the [MixSIAR GitHub repository](https://github.com/brianstock/MixSIAR). The diagnostic plots are generated using the ggmcmc package [(Fernández-i-Marín, 2016)](https://CRAN.R-project.org/package=ggmcmc).

## Usage

```
diagnose_model(jags_output, var.to.diag = "all", save = FALSE, save_path = ".")
```

## Arguments

| | |
|---|---|
| `jags_output` | The MCMC output summarized in the class jagsUI object output by `run_model` function |
| `var.to.diag` | The list of variables for which diagnostic plots should be produced and save. By default, this argument is "all" hence is run for all the variables. |
| `save` | Indicates whether diagnostic plots should be produced and saved. |
| `save_path` | The path indicating where to save the diagnostic plots. |

## Value

A matrix containing the Gelman diagnostic for all the variables monitored by the `run_model` function (`variables_to_save` argument).

## See Also

`run_model` to run the model

## Examples

```
realistic_biotracer_data <- read.csv(system.file("extdata", "realistic_biotracer_data.csv",
                                                 package = "EcoDiet"))
realistic_stomach_data <- read.csv(system.file("extdata", "realistic_stomach_data.csv",
                                               package = "EcoDiet"))

data <- preprocess_data(biotracer_data = realistic_biotracer_data,
                        trophic_discrimination_factor = c(0.8, 3.4),
                        literature_configuration = FALSE,
                        stomach_data = realistic_stomach_data)

write_model(literature_configuration = FALSE)

mcmc_output <- run_model("EcoDiet_model.txt", data, run_param="test")

Gelman_test <- diagnose_model(mcmc_output)
Gelman_test
```

---

# example_biotracer_data

*Example biotracer data*

---

## Description

This is an artificial and simple biotracer dataset, more specifically stable isotope analyses, made to illustrate the package on a simple case. All tables whose name start by "example" are describing different data from the same trophic groups.

## Format

A table with 15 rows and 3 columns. Each row is an isotopic sample from one individual. The columns are:

**group** the trophic group the individual belonged to

**d13C** the d13C measurement made on that individual

**d15N** the d15N measurement made on that individual

## example_literature_diets *Example literature diets*

### Description

This is an artificial and simple literature diets dataset, made to illustrate the package on a simple case. All tables whose name start by "example" are describing different data from the same trophic groups.

### Format

A table with 5 rows and 5 columns. The headers contain the predators' names, the first column contains the preys' names. Each cell contains the average diet proportions fond in the literature for the corresponding predator. The last row contains the average pedigree score for the literature on each predator.

## example_stomach_data *Example stomach data*

### Description

This is an artificial and simple stomachal dataset, made to illustrate the package on a simple case. All tables whose name start by "example" are describing different data from the same trophic groups.

### Format

A table with 5 rows and 5 columns. The headers contain the predators' names, the first column contains the preys' names. Each cell contains the number of the predator's stomachs in which this prey was found. The last row contains the total number of non-empty stomachs for each predator.

## mcmc_output_example *The MCMC output for running the example dataset*

### Description

This is the MCMC output for running the example dataset as illustrated in the introduction vignette (with 1e6 iterations, 1e3 adaptation steps) and with priors informed from the literature study. This data is here so that the plot_results fonction can be illustrated on results that have converged.

### Usage

```
mcmc_output_example
```

## Format

An object of class jagsUI of length 24.

## Examples

```
data("mcmc_output_example")
```

---

plot_data *Plot the input data*

---

## Description

This function is used to plot the input biotracer and/or the stomach content data. You can use the function with only one parameter to plot only one kind of data.

The figure(s) can be saved as PNG using: save = TRUE, and the directory path to which the figures are saved can be precised with: save_path = ".".

If only the stomach content data is entered, there will be a single raster plot containing the proportions of occurences in the stomachs.

For the biotracer data, there will be as many plots as the number of combinations of elements. For example if only two isotopes are entered, there will be a single biplot plotted. If three elements are entered (element A, B and C), three biplots will be shown : A vs. B, B vs. C and A vs. C.

## Usage

```
plot_data(
  biotracer_data = NULL,
  stomach_data = NULL,
  save = FALSE,
  save_path = "."
)
```

## Arguments

| | |
|---|---|
| biotracer_data | A dataframe containing the biotracer data in the specific format: the first column corresponds to the trophic group or latin species and the remaining columns contains the biotracer measures |
| stomach_data | A dataframe containing the stomach content data in a specific format: the first row contains the names of the prey trophic groups, the headers contains the names of the consumer / predator trophic groups, and the rest are the number of the predator's stomachs in which this prey was found. The last row contains the total number of non-empty stomach for the corresponding predator. |
| save | A boolean describing whether the figure should be saved as PNG. By default the figures are not saved. |
| save_path | A string describing the path to which the figures should be saved. By default the figures are saved in a temporary directory. |

## See Also

plot_prior to plot the prior means or probability distribution(s), plot_results to plot the posterior means or probability distribution(s)

## Examples

```
example_biotracer_data <- read.csv(system.file("extdata", "example_biotracer_data.csv",
                                               package = "EcoDiet"))
plot_data(biotracer_data = example_biotracer_data)

example_stomach_data <- read.csv(system.file("extdata", "example_stomach_data.csv",
                                             package = "EcoDiet"))

plot_data(biotracer_data = example_biotracer_data,
          stomach_data = example_stomach_data)
```

---

# plot_prior *Plot the prior means or probability distribution(s)*

---

## Description

This function plots the prior means or probability distribution(s) for one or the two variable(s) of interest : the trophic link probabilities ("eta") and/or the diet proportions ("PI").

The figure(s) can be saved as PNG using: `save = TRUE`, and the directory path to which the figures are saved can be precised with: `save_path = "."`.

If no "pred" nor "prey" parameter is entered, the plot will be a raster plot with the mean priors for all the trophic groups.

If one predator name is entered as "pred", the probability distribution(s) will be plotted for all its prey(s) by default. Some specific prey(s) name(s) can also be entered because if a predator has 22 preys, plotting them all will make the plot hard to read. So you can specify the one or many prey(s) of interest and only display their corresponding probability distribution(s).

The "variable" parameter can be specified if one wants to plot the priors for only one variable ("PI" or "eta").

## Usage

```
plot_prior(
  data,
  literature_configuration,
  pred = NULL,
  prey = NULL,
  variable = c("eta", "PI"),
  save = FALSE,
  save_path = "."
)
```

### Arguments

| | |
|---|---|
| data | the preprocessed data list output by the preprocess_data() function |
| literature_configuration | A boolean (TRUE or FALSE) indicating whether the model will have prior distributions informed by a literature study |
| pred | the predator name for which we want to plot the probability densities |
| prey | the prey(s) name(s) for which we want to plot the probability densities |
| variable | the variable(s) for which we want to plot the probability densities. By default we will plot the two variables of interest: eta and PI. |
| save | A boolean describing whether the figure should be saved as PNG. By default the figures are not saved. |
| save_path | A string describing the path to which the figures should be saved. By default the figures are saved in a temporary directory. |

### See Also

plot_results to plot the posterior means or probability distribution(s), plot_data to plot the input data

### Examples

```
realistic_biotracer_data <- read.csv(system.file("extdata", "realistic_biotracer_data.csv",
                                                 package = "EcoDiet"))
realistic_stomach_data <- read.csv(system.file("extdata", "realistic_stomach_data.csv",
                                               package = "EcoDiet"))

data <- preprocess_data(biotracer_data = realistic_biotracer_data,
                        trophic_discrimination_factor = c(0.8, 3.4),
                        literature_configuration = FALSE,
                        stomach_data = realistic_stomach_data)

plot_prior(data, literature_configuration = FALSE)
plot_prior(data, literature_configuration = FALSE, pred = "Cod")
plot_prior(data, literature_configuration = FALSE, pred = "Cod",
           prey = c("Crabs", "Shrimps"), variable = "eta")
```

---

## plot_results *Plot the posterior means or probability distribution(s)*

---

### Description

This function plots the posterior means or probability distribution(s) for one or the two variable(s) of interest : the trophic link probabilities ("eta") and/or the diet proportions ("PI").

The figure(s) can be saved as PNG using: save = TRUE, and the directory path to which the figures are saved can be precised with: save_path = ".".

If no "pred" nor "prey" parameter is entered, the plot will be a raster plot with the mean priors for all the trophic groups.

If one predator name is entered as "pred", the probability distribution(s) will be plotted for all its prey(s) by default. Some specific prey(s) name(s) can also be entered because if a predator has 22 preys, plotting them all will make the plot hard to read. So you can specify the one or many prey(s) of interest and only display their corresponding probability distribution(s).

The "variable" parameter can be specified if one wants to plot the priors for only one variable ("PI" or "eta").

## Usage

```
plot_results(
  jags_output,
  data,
  pred = NULL,
  prey = NULL,
  variable = c("eta", "PI"),
  save = FALSE,
  save_path = "."
)
```

## Arguments

| | |
|---|---|
| jags_output | the mcmc.list object output by the run_model() function |
| data | the preprocessed data list output by the preprocess_data() function |
| pred | the predator name for which we want to plot the probability densities |
| prey | the prey(s) name(s) for which we want to plot the probability densities |
| variable | the variable(s) for which we want to plot the probability densities. By default we will plot the two variables of interest: eta and PI. |
| save | A boolean describing whether the figure should be saved as PNG. By default the figures are not saved. |
| save_path | A string describing the path to which the figures should be saved. By default the figures are saved in a temporary directory. |

## See Also

plot_prior to plot the prior means or probability distribution(s), plot_data to plot the input data

## Examples

```
realistic_biotracer_data <- read.csv(system.file("extdata", "realistic_biotracer_data.csv",
                                                 package = "EcoDiet"))
realistic_stomach_data <- read.csv(system.file("extdata", "realistic_stomach_data.csv",
                                               package = "EcoDiet"))

data <- preprocess_data(biotracer_data = realistic_biotracer_data,
```

```
                        trophic_discrimination_factor = c(0.8, 3.4),
                        literature_configuration = FALSE,
                        stomach_data = realistic_stomach_data)

write_model(literature_configuration = FALSE)

mcmc_output <- run_model("EcoDiet_model.txt", data, run_param="test")

plot_results(mcmc_output, data)
plot_results(mcmc_output, data, pred = "Crabs")
plot_results(mcmc_output, data, pred = "Crabs",
             variable = "PI", prey = c("Bivalves", "Shrimps"))
```

---

`preprocess_data` *Check and preprocess the data*

---

## Description

This function preprocesses the data input by the user, checks that the different inputs have the right format, and creates the data list that will feed the JAGS model.

If an error appears with a clear message, it means that the input needs to be reformatted. Please follow the instructions in the error message. You can also look at the data examples to guide you.

## Usage

```
preprocess_data(
  biotracer_data,
  trophic_discrimination_factor,
  literature_configuration = FALSE,
  topology = NULL,
  element_concentration = 1,
  stomach_data = NULL,
  rescale_stomach = FALSE,
  literature_diets = NULL,
  nb_literature,
  literature_slope
)
```

## Arguments

`biotracer_data` A dataframe containing the biotracer data in the specific format: the first column corresponds to the trophic group or latin species and the remaining columns contains the biotracer measures

`trophic_discrimination_factor`
A vector containing the trophic discrimination factors corresponding to each column found in the biotracer data (except the group column of course)

`literature_configuration`
: A boolean (TRUE or FALSE) indicating whether the model will have prior distributions informed by a literature study

`topology`
: A matrix that the user may input if she wants the model to investigate some additionnal trophic links (by-default it is set on NULL and defined later from the stomach content data and the literature diets if present)

`element_concentration`
: A matrix containing the element concentration for each trophic group and each biotracer element (listed in the biotracer data). It is a matrix with as many columns as the number of trophic groups and as many rows as the number of elements. By default the matrix is filled with ones.

`stomach_data`
: A dataframe containing the stomach content data in a specific format: the first row contains the names of the prey trophic groups, the headers contains the names of the consumer / predator trophic groups, and the rest are the number of the predator's stomachs in which this prey was found. The last row contains the total number of non-empty stomach for the corresponding predator.

`rescale_stomach`
: A boolean (TRUE or FALSE) indicating whether the stomach content data will be rescaled. If TRUE, the stomach occurences are upscaled by dividing them by the maximum occurrences / the number of non-empty stomach.

`literature_diets`
: A dataframe containing the diet proportions found in the literature in a format similar to the stomach content data: the first row contains the names of the prey trophic groups, the headers contains the names of the consumer / predator trophic groups, and the rest are the average proportions of this prey in the predator's diet according to a literature study. The last row contains the pedigree score associated to the literature findings for each predators, a number between 0 and 1 indicating how much the literature findings are relevant estimates for the input data.

`nb_literature`
: A vector of one number containing the equivalent number of stomach for the literature priors on the eta variable

`literature_slope`
: A vector of one number containing the slope of the linear relationship between the pedigree scores and the PIs' coefficients of variation (CVs)

## Value

A list of preprocessed data, ready to be fed to the EcoDiet model through the 'rjags' package. This list contains a different number of elements, depending on which configuration is chosen by the user (literature_configuration = TRUE or FALSE).

## Examples

```
example_biotracer_data <- read.csv(system.file("extdata", "example_biotracer_data.csv",
                                               package = "EcoDiet"))
example_stomach_data <- read.csv(system.file("extdata", "example_stomach_data.csv",
                                             package = "EcoDiet"))
```

```
data <- preprocess_data(biotracer_data = example_biotracer_data,
                        trophic_discrimination_factor = c(0.8, 3.4),
                        literature_configuration = FALSE,
                        stomach_data = example_stomach_data)

example_literature_diets <- read.csv(system.file("extdata", "example_literature_diets.csv",
                                                 package = "EcoDiet"))

data2 <- preprocess_data(biotracer_data = example_biotracer_data,
                         trophic_discrimination_factor = c(0.8, 3.4),
                         literature_configuration = TRUE,
                         stomach_data = example_stomach_data,
                         literature_diets = example_literature_diets,
                         nb_literature = 10,
                         literature_slope = 0.5)
```

---

`realistic_biotracer_data` *Realistic biotracer data*

---

## Description

This is an artificial and realistic biotracer dataset, more specifically stable isotope analyses, made to illustrate the package on a complex dataset. All tables whose name start by "realistic" are describing different data from the same trophic groups.

## Format

A table with 300 rows and 3 columns. Each row is an isotopic sample from one individual, and there are 30 individuals sampled in each trophic group. The columns are:

**group** the trophic group the individual belonged to

**d13C** the d13C measurement made on that individual

**d15N** the d15N measurement made on that individual

---

`realistic_literature_diets` *Realistic literature diets*

---

## Description

This is an artificial and realistic literature diets dataset, made to illustrate the package on a complex dataset. All tables whose name start by "realistic" are describing different data from the same trophic groups.

### Format

A table with 11 rows and 11 columns. The headers contain the predators' names, the first column contains the preys' names. Each cell contains the average diet proportions fond in the literature for the corresponding predator. The last row contains the average pedigree score for the literature on each predator.

---

`realistic_stomach_data` *Realistic stomach data*

---

### Description

This is an artificial and realistic stomachal dataset, made to illustrate the package on a complex dataset. All tables whose name start by "realistic" are describing different data from the same trophic groups.

### Format

A table with 11 rows and 11 columns. The headers contain the predators' names, the first column contains the preys' names. Each cell contains the number of the predator's stomachs in which this prey was found. The last row contains the total number of non-empty stomachs for each predator.

---

`run_model` *Run the EcoDiet model*

---

### Description

This function runs the EcoDiet model using a Markov chain Monte Carlo approximation through the 'jagsUI' package to provide an approximated distribution for the variables of interest.

Depending on the `nb_iter` entered, this function may take hours, or even days to run. We advise you to first test whether your model is compiling properly with the by-default parameters, as this should take 1-2 min to run depending on your data size.

To save time, this function can solicit several cores (if available) to parallelize chains. Note that progress bars won't be displayed if chains are parallelized.

A warning message is printed if the model has not converged in the end (if the Gelman-Rubin diagnostic of at least one variable is > 1.1). For each run, the default 'jagsUI' package messages summarize the '.txt' file used for the definition of the BUGS model, the configuration of the model (iteration, adaptation, burnin, thin rate), the time required to run the model, and main statistics for the variables.

You need to have run the `preprocess_data` and the `write_model` functions before using this function, as their outputs are used as the inputs for `run_model`.

## Usage

```
run_model(
  model_file,
  data,
  inits = NULL,
  run_param = "test",
  variables_to_save = c("eta", "PI"),
  parallelize = FALSE,
  DIC.out = TRUE
)
```

## Arguments

| | |
|---|---|
| model_file | The file containing the BUGS definition of the EcoDiet model output by the write_model function |
| data | The preprocessed data list output by the preprocess_data() function |
| inits | A list containing the initial values of the variables. By default the initialisation values are NULL, which means that the chain initial values are drawn from the prior distributions. |
| run_param | A object that can be a list of the parameters to configure the JAGS model or a string acting as a shortcut characterizing the overall length of the run requested (e.g. "short" or "long"). If run_param is provided as a list, the user should provide at least nb_iter, i.e. the number of iterations to run (the more iterations, the better are the chances that the model will converge; very small by default to test if the model compiles properly), and nb_burnin, i.e. the number of burn-in steps to run (so that the variable approximations are not too influenced by the first initial random values). nb_thin, the thinning rate, is by default defined by the function. The number of adaptation steps nb_adapt can be specified but is not required (see jasgUI documentation for more details). If set manually, it should be at least set at 1000. |
| variables_to_save | A vector of variable names defining the variables to output. The number has a big number of variables but by default we only save the variables of interest that are the trophic link probabilities eta and the diet proportions PI. Only these saved variables are used to compute the Gelman-Rubin statistics that indicate whether the model has converged or not. |
| parallelize | Indicates whether chains should be parallelized using several cores. Recommended in case of complex models. |
| DIC.out | Indicates whether the DIC (Deviance Information Criterion) should be reported. |

## Value

A MCMC output formatted as a jagsUI object.

## See Also

preprocess_data to preprocess the data, and write_model to define the model.

## Examples

```
realistic_biotracer_data <- read.csv(system.file("extdata", "realistic_biotracer_data.csv",
                                                 package = "EcoDiet"))
realistic_stomach_data <- read.csv(system.file("extdata", "realistic_stomach_data.csv",
                                               package = "EcoDiet"))

data <- preprocess_data(biotracer_data = realistic_biotracer_data,
                        trophic_discrimination_factor = c(0.8, 3.4),
                        literature_configuration = FALSE,
                        stomach_data = realistic_stomach_data)

write_model(literature_configuration = FALSE)

mcmc_output <- run_model("EcoDiet_model.txt", data, run_param="test")
```

---

`write_model` *Write the EcoDiet model in BUGS*

---

## Description

This function writes the EcoDiet model in the BUGS syntax as a several line long string.

The model definition depends on whether or not literature data will be used to inform the priors, hence the parameter `literature_configuration`.

To know more about what is inside the model, please read the reference article.

## Usage

```
write_model(
  file.name = "EcoDiet_model.txt",
  literature_configuration = FALSE,
  print.model = FALSE
)
```

## Arguments

| | |
|---|---|
| `file.name` | The name and location under which the '.txt' BUGS definition of the model will be saved. If not provided, the file will be saved in the current repository under the "EcoDiet_model.txt" name. |
| `literature_configuration` | A boolean (TRUE or FALSE) indicating whether the model will have prior distributions informed by a literature study |
| `print.model` | Indicates whether the user wants to print the written model in the console. |

## Value

A string containing the model definition in BUGS

## See Also

run_model to run the model after it has been defined

## Examples

```
write_model(file.name="my_model_with_priors.txt", literature_configuration = TRUE)

write_model(literature_configuration = FALSE, print.model = TRUE)

unlink('my_model_with_priors.txt')
unlink('EcoDiet_model.txt')
```

# Index