multifit: an R function

Multi-scale analysis for landscape ecology

Pablo Yair Huais

December 2017

Description

Function *multifit* attempts to ease multi-scale data analysis in landscape ecology. The user provides a data.frame containing a column with the response variable and several columns depicting a particular landscape attribute at different spatial scales. Also, the user must provide the type of model to be applied in the analysis, along with a formula and any other relevant arguments, as well as the criterion to be used for the selection of the 'best' model. The function's output includes the following elements: a plot depicting the strength of each model, an optional plot showing the estimates of the response variable for each model, and a list containing relevant information about the models (including the plot and the models themselves).

Dependencies

The function requires the correspondent package that contains the function that will be used to run the models of the multi-scale analysis.

Usage

```
multifit(mod, multief, formula = NULL, data = NULL, args = NULL, criterion = "AIC",
         signif = TRUE, alpha = 0.05, print_sum = FALSE, plot_est = FALSE,
         xlab = "Radio [m]", labels = NULL, type = "b", pch = c(1, 16))
```

Arguments

`mod`	string depicting the type of model to be applied (see details)
`multief`	character. A vector containing the column names of the data.frame holding the landscape attribute at different spatial scales
`formula`	formula to be applied to each model, labeling the landscape attribute to be analyzed as `multief` (see details)
`data`	data.frame containing the response variable and the landscape attribute at different spatial scales
`args`	character vector with any additional argument/s of the models (see details)
`criterion`	character. Criterion of selection of the 'best' model. So far, one of three options (`"AIC"`, `"BIC"` or `"R2"`) or a user-defined function specifying the name of the function in a first element and the model-selection criterion (`"max"` or `"min"`) in a second one (see details).
`signif`	logical. Differentiate non-significant and significant models in the plot?
`alpha`	numeric value, between 0 and 1. Statistical significance level (only relevant for the plotting of significance stars)
`print_sum`	logical. Print the summary of the best model?
`plot_est`	logical. Plot the estimates of the models?
`xlab`	character. A title for the x axis
`labels`	character vector with the labels of the models at the x axis
`type`	character. What type of plot should be drawn (see details)?
`pch`	numeric vector of two elements, containing the codes of the shapes of non-significant and significant models, respectively (see details)

Details

The aim of this function is to facilitate the analysis of ecological data in relation to any attribute of landscapes, via a multi-scale approach. In this way, *multifit* allows the user to run many statistical models at the same time (i.e. one model per spatial scale), and simplifies the analysis and selection of the appropriate spatial scale for the provided response variable.

First of all, the user must possess a data.frame with at least a column with the response variable to be analyzed, and several columns with information about the landscape attributes (e.g. habitat amount) at different spatial scales (i.e. one column per spatial scale). This data.frame must be specified in the argument data of the function.

The user must provide the statistical type of model to be applied in the analysis, specifying it as a character in the argument mod (e.g. "lm" for a classic linear model). Take into account that the user must have loaded the correspondent package containing the function before running the multi-scale analysis. In the argument `multief`, the user must provide a character vector depicting the names of the columns that contain the information of the landscape attribute in data (e.g. `multief = c("R_500", "R_1000", "R_1500")`, which would refer to the landscape attribute at three different spatial scales: radius of 500, 1000 and 1500 m). Is important to provide the elements of the vector in an order that makes sense (which probably would be an order representing an increase in the spatial scale), as this order will be used for the multi-scale analysis.

The argument `formula` must be fulfilled with the statistical formula to be applied to the models. This must include at least the main response variable and a predictor variable named '`multief`' (e.g. `formula = richness ~ multief`). The function will recognize this particular word (i.e. `multief`) in each model as the predictor variable containing the landscape attribute at a particular spatial scale. If the model definition of the function specified in `mod` does not include an argument called `formula`, then the response and predictor variables can be defined in the argument `args` (see below) in the way that the model definition requires.

The user may add as many arguments as needed to run the models at each spatial scale. These must be added in the argument `args` as a character vector, each element depicting a particular argument written as the user would in an individual analysis (e.g. assuming a classic linear model, `args = c("na.action = na.omit", "singular.ok = FALSE")`). As explained above, `args` can include the response and predictor variables for those functions that do not include an argument named `formula` (e.g. the function 'lme' of the package *nlme*) by specifying them in the correspondent arguments.

The `criterion` argument must include the criteria to be used for the selection of the 'best' model among the different spatial scales (i.e. the one with the strongest relationship with the response variable). So far, *multifit* allows choosing between three options: `"R2"` (for R-squared, i.e. coefficient of determination), `"AIC"` (for Akaike Information Criterion), and `"BIC"` (for Bayesian Information Criterion). The user must take into account if the type of model defined in `mod` allows the calculation of the specified criterion. If not, *multifit* would recommend the use of another one. The function allows as well the possibility of specifying a user-defined function for the calculation of a different criterion. This user-defined function should have the possibility of calculating a particular criterion value from an object containing the statistical models at each spatial scale. If this is the case, the user must specify the name of the function in a first element of the vector, and the model-selection criteria (`"max"` or `"min"` of the value of the criterion) in a second element (e.g. `criterion = c("my_function", "max")`).

The argument `signif` asks if the points of the plot should be drawn differentially according to the statistical significance of estimates, whereas the argument `alpha` defines the statistical level. The argument `print_sum` asks if the summary of the selected model (i.e. the 'best' model by the defined criterion) should be printed in the console at the end. The argument `plot_est` asks if a plot depicting the estimates of the response variable at each spatial scale should be drawn in a separated plot.

The user may change some aesthetic characteristics of the plot/s, such as `xlab` or `type`. The argument `pch` defines the shape of the points to be plotted, in a numeric vector of two elements, for non-significant and significant (if `signif = TRUE`) models respectively. By default, the function will plot the non-significant models as white-filled dots and the significant ones as black-filled dots. The argument `labels` allows the user to specify a character vector with names for each model at the x axis.

Value

multifit returns a list with the following components:

`lands_summary`	a data.frame containing a summary of the defined landscape attribute at each spatial scale: n, min, max, range, mean and median
`summary`	a data.frame containing relevant information of the models, including the value of the criterion, the estimates and the p.values
`plot`	a plot that shows the strength of the models at each spatial scale by the specified criterion, along with an optional plot of the estimates for each model
`models`	a list containing the models of all spatial scales as individual R objects. These will be useful for *a posteriori* tests of particular models
`warnings`	a list containing the warnings, if they occurred, during the analysis of the models for each spatial scale
`messages`	a list containing the messages, if they occurred, during the analysis of the models for each spatial scale

The function also draws a plot, which is the same that is included in the component plot of the returned list.

Note

So far, *multifit* has been tested in fitting models of the packages *stats* (e.g. "lm"), *lme4* (e.g. "glm", "lmer", "glmer"), *nlme* (e.g. "lme") and *pscl* (e.g. "zeroinfl", "hurdle"). The function should work with other models and packages, and it will be tested with more of them in the future.

Author

Pablo Yair Huais – phuais@gmail.com

Example

```
#               #
# First example #
#               #

# Read table: fake data simulating bird richness. You can find this data.frame in
#  the following link:
#  https://github.com/phuais/multifit/blob/master/fake_data/bird_richness.txt
data <- read.table("bird_richness.txt", header = T)

# Create an object with multifit
# In this case, we fit the response variable S (richness) against the proportion of
#  forest amount at ten different spatial scales, starting from 500 until 5000, by
#  500 m per step. We apply a GLMM with glmer function of the package lme4 and we use
#  AIC criterion for the model selection process (the default option)
# Notice that the effect at each spatial scale is specified as the correspondent names
#  of the columns of data
library(lme4)
fits <- multifit(mod = "glmer", multief = colnames(df)[3:12],
        formula =  S ~ multief + (1|site), args = c("family = poisson"),
        data = data, criterion = "AIC", plot_est = T)

# Once the fitting of the models finished, we can explore the object...
```

```r
# Prints a summary table of the landscape attribute at each spatial scale
fits$lands_summary

# Prints a summary table with relevant information of the multi-scale analysis
fits$summary

# Prints the plot (that was already plotted when running multifit)
fits$plot

# Prints a particular model object, and its summary
fits$models$R_2500
summary(fits$models$R_2500)

# Check for possible warnings or messages of a particular model
fits$warnings$R_2500
fits$messages$R_2500

#                #
# Second example #
#                #

# We can define our own criterion of model selection with a
#  user-defined function, for example, a pseudo-R-squared for GLMMs,
#  which can be calcultaed with the function r.squaredGLMM of the package MuMIn:

library(MuMIn)
pseudo_R <- function(x){
  as.numeric(suppressMessages(r.squaredGLMM(x)[1]))
}

# And then run multifit defininining the function and the criterion of selection
#  in the argument criterion. In this case, as we are defining a R-squared
#  criterion, the 'best' model would be the one that has de highest value of
#  this R-squared. To clarify this, we specify "max" as a second element of the argument:

fits <- multifit(mod = "glmer", multief = colnames(df)[3:12],
                 formula =  S ~ multief + (1|site), args = c("family = poisson"),
                 data = data, criterion = c("pseudo_R", "max"), plot_est = T)
```