# multifit: an R function

Multi-scale analysis for landscape ecology

*Pablo Yair Huais*

*May 2018*

### Description

Function *multifit* attempts to automate multi-scale analysis in landscape ecology. The user provides a data.frame containing a column with the studied response variable and several columns depicting a particular landscape attribute at various 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 (e.g. blocking factors, co-variables, random effects, etc.), 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 estimated model coefficients of the response variable for each model, and a list containing relevant information about the models (including the plot/s and the models themselves).

### Dependencies

The function requires the correspondent package with the function that will be used to run the models.

### Usage

```
multifit(mod, multief, formula = NULL, data, args = NULL, criterion = "AIC", site_id = NULL,
         signif = TRUE, alpha = 0.05, print_sum = FALSE, plot_est = FALSE,
         xlab = "Radius [m]", ylab = NULL, 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 various 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 various 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 element (see details) |
| `site_id` | character. A string depicting the column name of the data.frame holding the identity of the different sites. Only relevant for the summary of the landscape attributes (see details) |
| `signif` | logical. Differentiate non-significant from significant models in the plot with different point shapes? |
| `alpha` | numeric value, between 0 and 1. Statistical significance level (only relevant if `signif = TRUE`, see details) |
| `print_sum` | logical. Print the summary of the 'best' model? |
| `plot_est` | logical. Plot the estimated model coefficients of the models? |
| `xlab` | character. A title for the x axis |
| `ylab` | character. A title for the y axis of the plot that shows the strength of the models |
| `labels` | character vector with the labels of the models on 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 automate the analysis of ecological data in relation to a particular landscape attribute, 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 one column with the response variable to be analyzed, and several columns holding the values of a landscape attribute (e.g. habitat amount or number of patchs) at various 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). 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 values of the landscape attribute (e.g. `multief = c("R_500", "R_1000", "R_1500")`, which would refer, for example, to the landscape attribute at three different spatial scales: radii of 500, 1000 and 1500 m). It is important to provide the elements of the vector in an order that makes sense (which logically would be an order representing an increase in the spatial scale), as this order will be used for the multi-scale analysis and the plots.

The argument `formula` must be fulfilled with the statistical formula to be applied in each model. This must include at least the main response variable and a predictor variable named '`multief`' (e.g. `formula = response_variable ~ multief`). The function will recognize this particular string (i.e. `multief`) in each model as the predictor variable containing the landscape attribute at each 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 do it 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 (i,e, `multief`) for those functions that do not include an argument named `formula` (e.g. the function 'lme' of the package *nlme*, Pinheiro et al. 2018) by specifying them in the correspondent arguments.

The `criterion` argument must include the criterion to be used for the selection of the 'best' model among the various 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* will 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 capacity of calculating a particular criterion value from an object containing the statistical models at each spatial scale (i.e. the output of a model being run individually). 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 criterion (`"max"` or `"min"` of the value of the criterion) in a second element (e.g. `criterion = c("my_function", "max")`).

The `site_id` argument should include the name of the column in the data.frame that holds the identity of the sites from which the response variable was gathered. Clarifying this is only necessary to properly calculate the n, mean and median of the landscape attribute at each spatial scale (which are included as a summary table in the output of the function). If `NULL`, the n, mean and median of the landscape attribute will not be calculated, due to possible ambiguities in the values of the landscape attribute at different sites.

The argument `signif` asks if the points of the plot should be drawn differentially according to the statistical significance of the estimated model coefficients, whereas the argument `alpha` defines the statistical level. The argument `plot_est` asks if a plot depicting the estimated model coefficents at each spatial scale should be drawn in a plot, aside the default plot. 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.

The user may change some aesthetic characteristics of the plot/s, such as `xlab` (the labels of the x axis) or `type` (the type of plot, see ?type). The argument `ylab` changes the title of the y axis of the plot that shows the strength of the models at each spatial scale. The argument `pch` defines the shape of the points to be plotted, in a numeric vector of two elements, for non-significant and significant models respectively (only relevant if `signif = TRUE`). 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 on the x axis, which must have equal number of elements as number of analyzed spatial scales.

### 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 estimated model coefficients 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 estimated model coefficients |
| `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 draws a plot, which is the same that is included in the component 'plot' of the returned list (available as an R object for the user to draw it in the R graphic device anytime in the future).

### Note

So far, *multifit* can only be applied for univariate multi-effects only (i.e. with a unique landscape attribute). Additionally, *multifit* has been tested in fitting models of the packages *stats* ("lm" and "glm"; R Core Team 2017), *lme4* ("lmer" and "glmer"; Bates et al. 2015), *glmmadmb* ("glmmADMB"; Fournier et al. 2012), *glmmTMB* ("glmmTMB"; Magnusson et al. 2017), *nlme* ("lme"; Pinheiro et al. 2018) and *pscl* ("zeroinfl" and "hurdle"; Zeileis et al. 2008). Nevertheless, the function should work with other models and packages, and will be updated in the future.

### Author

Pablo Yair Huais – phuais@gmail.com

### Example

```r
#               #
# First example #
#               #

# Read table: fake data simulating bird richness along a gradient of forest amount.
#  This contains a column with the response variable, in this case the richness of
#  birds, and 10 colums holding the values of forest amount at 10 different spatial
#  scales (from 500 until 5000, by 500 m step). Also, another column called 'site'
#  holds the identity of the sites from wich the response variable would be gathered.
#  There are 50 sites, 10 measures of bird richness per site, which make a total of
#  500 measures (i.e. rows).
# You can find and download this data from the following link:
#  https://github.com/phuais/multifit/blob/master/fake_data/bird_richness.csv

data <- read.csv("fake_data/bird_richness.csv")

# Create an object with multifit
#  In this case, I fit the response variable 'S' (richness) against the proportion of
#  forest amount at this set of spatial scales. I apply a GLMM with the function
#  'glmer' from package 'lme4' (Bates et al. 2015) and use the 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. Also
#  notice that I define a random effect 'site' for the GLMM.

library(lme4)
fits <- multifit(mod = "glmer", multief = colnames(data)[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, a plot will be drawn and explore the object
# 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 again in the graphic device
fits$plot

# Prints a particular model object
fits$models$R_2500

# And its summary
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 #
#                #

# I can define my 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 (Bartoń 2018):

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

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

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

### References

Bates D., Maechler M., Bolker B., Walker S. (2015). Fitting Linear Mixed-Effects Models Using lme4. Journal of Statistical Software, 67(1), 1-48.

Bartoń K. (2018). MuMIn: Multi-Model Inference. R package version 1.40.4.

Fournier D.A., Skaug H.J., Ancheta J., Ianelli J., Magnusson A., Maunder M., Nielsen A., Sibert J. (2012). AD Model Builder: using automatic differentiation for statistical inference of highly parameterized complex nonlinear models. Optimization Methods and Software, 27(2), 233-249.

Magnusson A., Skaug H.J., Nielsen A., Berg C.W., Kristensen K., Maechler M., van Bentham K. J., Bolker B.M., Brooks M.E. (2017). glmmTMB: Generalized Linear Mixed Models using Template Model Builder. R package version 0.1.3.

Pinheiro J., Bates D., DebRoy S., Sarkar D., R Core Team (2018). nlme: Linear and Nonlinear Mixed Effects Models. R package version 3.1-131.1.

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.

Zeileis A., Kleiber C., Jackman S. (2008). Regression Models for Count Data in R. Journal of Statistical Software 27(8).