Model Parameter Estimation

Mestre, F.; Canovas, F.; Risk, B.; Pita, R.; Mira, A.; Beja, P.

September 14, 2024

1 Introduction

The correct SPOM parametrization is one of the most crucial steps of every simulation procedure allowed by the package. The parameters are the numeric translation of the relation of the species with the landscape. Although this is not the main focus of this package, MetaLandSim offers some basic tools that will help to estimate the parameters to run the simulations. However, the user can have the parameters values available from other sources (whether published papers or estimated with other software tools) in each case the simulations can be run using those parameters. For a good overview of the several methods to parametrize the SPOM see Etienne et al. (2004). Most of the methods described can be implemented with MetaLandSim. Also, it is recommended that the user reads the book by Hanski (1999) in order to acquire the basic knowledge about metapopulation ecology and the first estimation procedures. The function parameter.estimate is the package tool to parametrize the spom function. This vignette clarifies the options available and gives a step-by-step guide of the function usage. The current version of the package does not executes all computations. Rather, some of the methods ((‘MCsim’,‘rescue’and ‘norescue’) create the files needed as input to two applications provided by Moilanen (1999) (MCsim) and ter Braak and Etienne (2003) ((‘rescue’ and ‘norescue’). Future versions of the package should incorporate these procedures in the R code. The objective of this vignette is to describe the parametrization procedures, allowing the user to produce robust estimates of the parameters in order to proceed with the landscape simulation process or with the range expansion simulation. Third vignette is run with made up data based upon real information.

2 Which method to choose

Currently, the following methods are available in the function parameter.estimate:

- *Rsnap_1* - Regression of snapshot data, using one snapshot (based on Oksanen, 2004).

- *Rsnap_x* - Regression of snapshot data, using more than one snapshot (based on Oksanen, 2004).
- *MCsim* - Monte Carlo simulation (Moilanen, 1999).
- *norescue* - Bayesian MCMC, not considering Rescue effect (ter Braak and Etienne, 2003).
- *rescue* - Bayesian MCMC, considering Rescue effect (ter Braak and Etienne, 2003).

In addition, the following functions from Risk et al. (2011) are available, which estimate the parameters α (1 / dispersal distance), b (exponent of jth sites' area in the colonization model), y (term in the denominator of the colonization probabilities), e (numerator in extinction probability, i.e, probability of extinction for site of area equal to one), and x (exponent of area in denominator in extinction probability):

- *ifm.naive.MCMC* - For multi-year occupancy data assuming no missing data.
- *ifm.missing.MCMC* - For multi-year occupancy data assuming no missing data.
- *ifm.robust.MCMC* - For multi-year occupancy data with multiple visits per year, allowing imperfect detection.

When data are missing, the accuracy of the estimates of the parameters of the IFM is improved by using ifm.missing.MCMC, and when there is imperfect detection, the accuracy can be further improved by ifm.robust.MCMC.

Amongst the methods allowed by MetaLandSim, the choice is facilitated by a careful consideration of the characteristics of each method and the dataset (such as number of snapshots). An examination of the advantages and drawbacks of the methods used in parameter.estimate is available in Etienne et al. (2004). The methods 'rescue' and 'norescue' are computed using the application provided by ter Braak, and Etienne (2003) and the method 'MCsim' is computed using the application provided with Moilanen (1999). 'Rsnap_1' and 'Rsnap_x' are computed using the R code based on Oksanen (2004). Using the function parameter.estimate, the first three methods only create the needed files to run the applications. Next the user can use the function create.parameter.df to create a data frame with the estimated parameters. The application of Moilanen (1999) allows the estimation of the following parameters: x, y, e, A0, e' and alpha. The application by ter Braak and Etienne (2003) allow the estimation of the following parameters: e, x, y, z, alpha and b. Future versions of the package should include the virtual migration model (Hanski et al. 2000), allowing the estimation of b (which scales patch areas to population size and emigration rates) and c (here c1, which scales immigration with patch area).

3 Work-flow

3.1 Regression on Snapshot Data - method Rsnap_1 or Rsnap_x

This is the simplest approach; it runs faster but provides the least reliable estimates of the parameters. It does not use turnover, only spatial structure and occupancy status.

```
> library(MetaLandSim)
> data(occ.landscape)
> data(occ.landscape2)
> #Using data with only one snapshot of the occupancy status
> param1 <- parameter.estimate (occ.landscape, method='Rsnap_1')
> param1
```

```
       par_output
alpha 0.008333333
x     0.256707865
y     0.016724074
e     0.211572786
```

```
> #Using data with more than one snapshot of the occupancy status
> param2 <- parameter.estimate (occ.landscape2, method='Rsnap_x',
+ nsnap=10)
> param2
```

```
        par_output
alpha  0.008333333
x     -0.172333542
y      0.021030749
e      0.742599240
```

3.2 Monte Carlo Simulation - method MCsim

This approach is more time-consuming (depending on computing power). This option calls for the functions on the paper by Moilanen (1999). Here, create.parameter.df only creates the files to be used as input to the application. The user should read the paper thoughtfully, as well as the available help files. A file with the settings (inputMCsim.set) and a file with the data (inputMCsim.dat) will be created into the working directory. Editions to the settings file will be needed in order to run the application using the three step procedure described in the readme.txt file. First run the application using Nlr, then using Bnlr and finally using mc (between each step the setting file should be edited to change the method). After running Nlr and Bnlr replace, in the settings file, 'edit x', 'edit y' and 'edit e' with those values (these are the priors to the simulation). The application and help files can be downloaded from the Ecological Archives, available (here).

```
> library(MetaLandSim)
> data(occ.landscape2)
> #First, generate the files to be the input of the application
> parameter.estimate (occ.landscape2, method='MCsim')
> #run the application mcm.exe from Moilanen (1999).
> #Previously read the readme.txt file #available with the
> #application.
> #Consider particularly the three step procedure for estimation,
> #using nonlinear regression (Hanski, 1994) to produce priors
> #for the Monte Carlo simulation). In the command line (first put
> #the application and the files in a folder with no spaces
> #in the name.
> #e.g.: 'C:/moilanen/'):
>
> #mce.exe inputMCsim.dat inputMCsim
>
> #Or, from R:
>
> system('mce.exe inputMCsim.dat inputMCsim')
> #After, create a data frame, with create.parameter.df,
> #using the estimated parameters:
> param3 <- create.parameter.df(alpha, x, y, e)
>
```

3.3 Bayesian MCMC - methods 'rescue' and 'norescue'

This is the approach developed in the paper by ter Braak et al. (2003). The parameter.estimate function only produces the files needed to be used as input in this application. It produces a dataset file (input_rescue.dat or input_norescue.dat), a parameter file (input_rescue.par or input_norescue.par) and a distance file (input_rescue.dis or input_norescue.dis). Then, by using create.parameter.df, a data frame can be created with the parameters computed with the application. To understand what the created files contain, and to understand the method the user should read the paper by ter Braak et al. (2003) as well as the help files available with the application. Editions to the parameters file will be needed in order to run the application. It is recommended to run one of the simplest methods to provide priors to the Bayesian MCMC simulation. Then, in the parameter file, replace 'edit x', 'edit y' and 'edit e' with those values. Be attentive to the fact that the output is given log-transformed. Before using the parameters in the simulation procedure they need to be back-transformed, using an exponential. This application, the source code, help files and sample data can be downloaded from the Ecological Archives, available (here).

```
> library(MetaLandSim)
> data(occ.landscape2)
> #Method 'rescue'
> parameter.estimate (occ.landscape2, method='rescue')
> #run the application file fmetapop_rescue.exe from
> #the command line (first put the application and the
> #files in a folder with no spaces in the name.
> #e.g.: 'C:/terbraak/'):
>
> #fmetapop_rescue input_rescue
>
> #Or, from R:
>
> system('fmetapop_rescue input_rescue')
> #After, create a data frame, with create.parameter.df,
> #using the estimated parameters:
> #param4 <- create.parameter.df(alpha, x, y, e)
>
> #Method 'norescue'
> parameter.estimate (occ.landscape2, method='norescue')
> #run the application file fmetapop_norescue.exe from
> #the command line (first put the application and the
> #files in a folder with no spaces in the name.
> #e.g.: 'C:/terbraak/'):
>
> #fmetapop_norescue input_norescue
>
> #Or, from R:
>
> system('fmetapop_norescue input_norescue')
> #After, create a data frame, with create.parameter.df,
> #using the estimated parameters:
>
> param5 <- create.parameter.df(alpha, x, y, e)
>
```

3.4 Bayesian IFM Naive, IFM Missing, and IFM Robust

MetaLandSim includes the functions ifm.naive.MCMC, ifm.missing.MCMC, and ifm.robust.MCMC, which implement the models described in Risk et al. (2011). The function ifm.missing.MCMC can be used to improve the estimation accuracy of the dispersal ability ($1/\alpha$) when there are sites with missing data, where the occupancy states of the missing data are also estimated. The function ifm.robust.MCMC incorporates imperfect detection in addition to missing data.

To use these functions, one needs to scale the distances and areas to be similar to the scaling used in the simulated data, as described below. This is because the functions use uniform priors, specifically, $\alpha \sim Unif[2, 50]$, $b \sim$

$Unif[0,5]$, $y \sim Unif[0,20]$, $x \sim Unif[0,5]$, and $e \sim Unif[0,1]$.

The distances should be scaled to be less than one. The scaling of the distances affects the scaling of α. The idea is that small distances tend to result in larger values of α, which helps keep the posterior away from zero and improves convergence. Note that the parameter α is a measure of 1 / dispersal ability, so with the prior $Unif[2,50]$, this allows the dispersal ability to vary between $1/50$ and $1/2$. Thus the distances need to be scaled so that this is a reasonable restriction.

Secondly, the areas should also be scaled such that the average area is roughly equal to one. Note that when the average area is equal to one, the parameter e equals the extinction probability of the average-sized site. Additionally, this helps convergence of x.

Fitting the Bayesian formulation of the IFM can take patience. The parameters b and y can be correlated. We have found in our applications that the model requires fairly informative priors for the parameter y. Since there are already many arguments to the functions, the priors are not included as arguments and are fixed at the values described above. For applications with an abundance of data, one may want to create new copies of the functions with more diffuse priors.

As a general recommendation, we suggest using the maximum a posteriori (MAP) estimates in subsequent MetaLandSim functions rather than the posterior mean or median. This is because the posteriors are truncated at zero and non-Gaussian; the mean in particular is a poor estimate of the“most probable" parameter value. (Ideally, one would pursue using the entire posterior sample in subsequent MetaLandSim forecasting to better incorporate parameter uncertainty, although that would be computationally challenging.) To calculate the MAPs, we provide the function calcmode().

For detailed instructions on using the functions, please see the examples in the function help files. We have also included supporting functions that allow the use of the coda package (Plummer et al 2006) for assessing posterior convergence.

4 References

1. Etienne RS, ter Braak CJF and Vos CC (2004). Application of stochastic patch occupancy models to real metapopulations. In: Hanski I and Gaggiotti (Eds.) Ecology, Genetics, and Evolution of Metapopulations. Elsevier Academic Press. 696 pp.

2. Hanski, I. (1994). A practical model of metapopulation dynamics. Journal of Animal Ecology, 63: 151-162.

3. Hanski, I., Alho, J. and Moilanen, A. (2000). Estimating the parameters of survival and migration of individuals in metapopulations. Ecology, 81: 239-251.

4. Hanski, I. (1999). Metapopulation Ecology. Oxford University Press. 313 pp.

5. Moilanen, A. (1999). Patch occupancy models of metapopulation dynamics: efficient parameter estimation using implicit statistical inference. Ecology, 80(3): 1031-1043.

6. Oksanen, J. (2004). Incidence Function Model in R. url:`http://cc.oulu.fi/~jarioksa/opetus/openmeta/metafit.pdf`.

7. Plummer, M., N. Best, K. Cowles, K. Vines. (2006). CODA: Convergence diagnosis and output analysis for MCMC. R News, 6(1): 7-11.

8. Risk, B. B., P. de Valpine, S. R. Beissinger. (2011). A robust-design formulation of the incidence function model of metapopulation dynamics applied to two species of rails. Ecology, 92(2): 462-474.

9. Ter Braak, C. J., and Etienne, R. S. (2003). Improved Bayesian analysis of metapopulation data with an application to a tree frog metapopulation. Ecology, 84(1): 231-241.