m2b tutorial

Laurent Dubroca, Andréa Thiebault

2017-05-16

Introduction

Animal behaviours affect animal distributions and trophic relationships and are therefore fundamental in determining ecological patterns. Whereas the direct observation of animal behaviour is often limited due to logistical constraints, collection of movement data have been greatly facilitated through the development of bio-logging. Animal movement data obtained through tracking instrumentation may potentially constitute a relevant proxy to infer animal behaviour. This is, however, based on the premise that a range of movement patterns can be linked to specific behaviours.

Statistical learning constitutes a number of methods that can be used to assess the link between given variables from a fully informed training dataset and then predict the values on a non-informed variable. We chose the random forest algorithm for its capacity to deal with imbalanced data (particularly relevant for behavioural data), its high prediction accuracy and its ease of implementation (Breiman (2001), Chen, Liaw, and Breiman (2004)). The strength of random forest partly relies in its ability to handle a very large number of variables. Hence, our methodology is based on the derivation of multiple predictor variables from the movement data over various temporal scales, in order to capture as much information as possible on the changes and variations of movement.

In this package we developed a method to link the movement patterns of animals with their behavioural states, using the random forest algorithm. The behavioural state of an animal can refer here to any categorization of the observed behaviour. The specificity of this method relies on the derivation of multiple predictor variables from the movement data over a range of temporal windows. This procedure allows to capture as much information as possible on the changes and variations of movement and ensures the use of the random forest algorithm to its best capacity. The method is very generic, applicable to any dataset providing movement data together with observation of behaviour.

This tutorial presents a new class named `xytb`, the functions that were implemented for the use of this package, and an example of application. The package can be installed using the CRAN system (`install.package("m2b")`), but the development version can be found on github (https://github.com/ldbk/m2b).

A new class

`xytb` is a S4 class built to provide in a single object all the information associated to a track. This includes the tracking data (two dimension space coordinates, time and behavioural states), the predictor variables derived from the movement data, the resulting model (to be used on other datasets) and the prediction (on the given dataset). `xyt` relates to information about the movement data, and `b` to the behavioural state. These information are containted into 8 slots, each of them deriving from different methods and functions (see Figure 1 for details). This object was created for the user to keep everything (data, model and prediction) in a single container, but also, by extension, for the user to (1) keep track of the precision of the model predictions and (2) exchange the analyses and results with different users easily.

Methods and function overview

The analytical procedure is summarized in 4 main functions and methods. For a full description of the use of the functions, please refer to the help provided for each function.

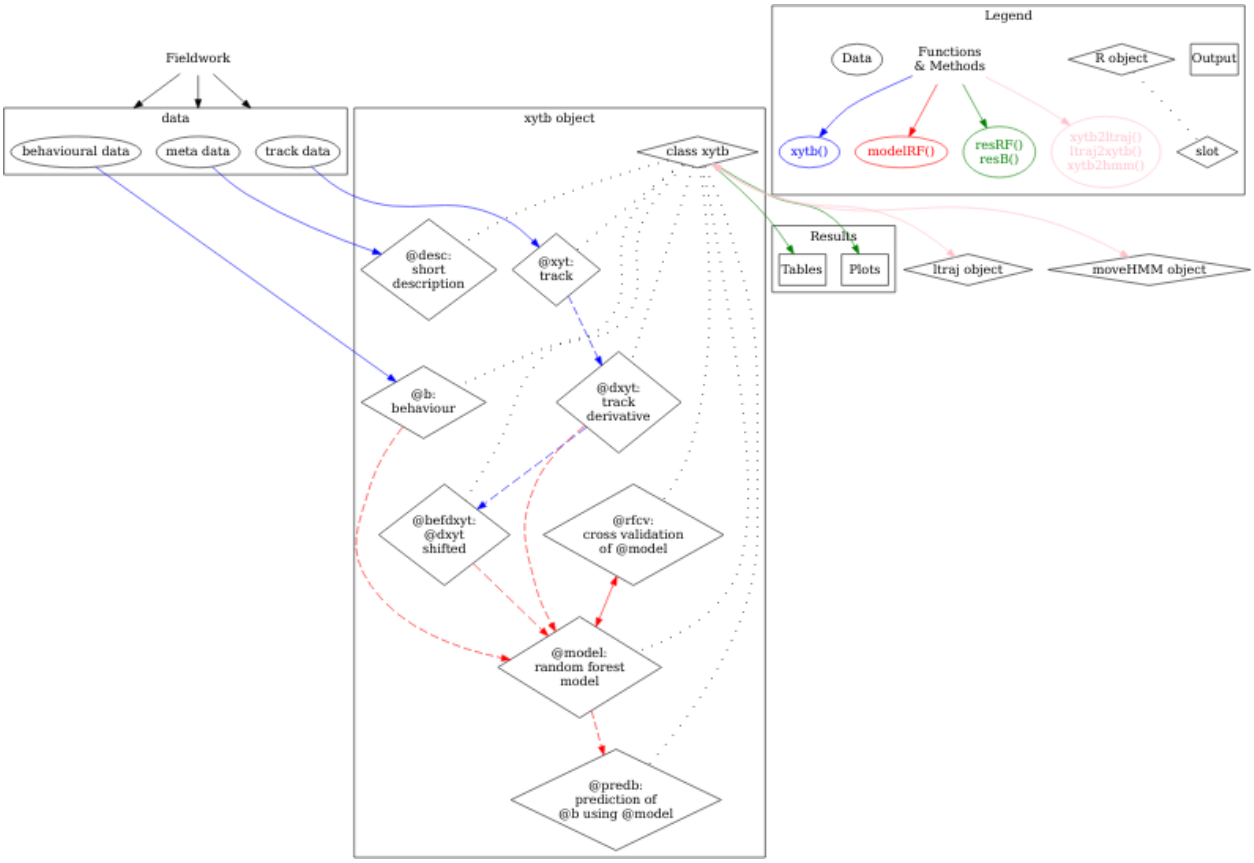


Figure 1: Schematic of the work flow from the raw data to the results. The `legend` box provides the symbolic representation (shape and color) of the different objects. The arrows between boxes represent the use of the package functions and methods: in blue the computation of the data, in red the modelling, in green the outputs and in pink the export to the ltraj format (adehabitat package). The diamond boxes inside the `xytb object` box represent the slots of the classi (dotted lines link the xytb class and the slots). Diagram generated using Graphviz.

xytb

xytb is a class, but also a method. Used as a function xytb will calculte the predictor variables and store all the data in a newly created xytb object. 4 distinct signatures help the user to load trackings and behavioural data in the object.

The data must be presented in a dataframe with locations in lines, and 5 variables in column: x the longitude, y the latitude, t the time in POSIXct format, b the behavioural state and id the individual identification for the track. The function xytb can then be used with this dataframe as an input, so the predictor variables are calculated and stored in the slots dxyt and befdxyt of the xytb created object. For the calculation of predictor variables, three parameters can be set:

- winsize specifies the sizes of sliding windows on which to compute the statistical operators,
- idquant specifies the quantiles to be computed,
- move (optional) specifies the number of points for the calculated variables to be shifted backward (variables to be added to the ones calculated at the time).

If included, the latter parameter will account for a delay between the reaction of the animal captured in the movement data, and its behaviour as recorded by the observer. In all cases, the original data are not modified, but derivated data are saved in the corresponding slots.

modelRF

This function computes a random forest model to predict the behavioural state (response) from the movement data (predictors). This is a simple wrapper calling the randomForest function from the randomForest package (https://CRAN.R-project.org/package=randomForest). This function will update the xytb object and store the outputs in the slots rfcv (cross-validation for the choice of the parameter mtry), model (model itself) and predb (predictions). Cross-validation has to be done independently to set up the mtry parameters for the model.

resRF and resB

These two functions compute and plot the diagnostics and results of the model. The function resRF provides the error rate, the convergence of the model, a confusion matrix and the importance of variables. The function resB plots the predictions vs observations, over time or space.

xytb2ltraj and ltraj2xytb

These functions import or export a xytb object to an object of class ltraj. The latter is used in the adehabitatLT package where numerous function are dedicated to the analysis of trajectories (see Calenge, Dray, and Royer-Carenzi (2008)).

xytb2hmm

This function import a xytb object to an object of class moveHMM. The latter is used in the moveHMM package which provides functions dedicated to the analysis of trajectories using hidden Markov models (see Michelot, Langrock, and Patterson (2016)).

An example

Data

The data frame `track_CAGA_005` contains the tracking and behavioural data collected from a Cape gannet (*Morus capensis*, Lichtenstein 1823). Tracking data include latitude, longitude and time (class POSIXct). Behavioural data include three states coded as '1' (bird diving), '2' (bird sitting on the water), '3' (bird flying), and a state `-1` for data points where the behaviour could not be observed. A state with no observation can be declared in some functions (`resB` for example) using the parameter `nob`, equal to '-1' in our case (see functions' help).

```
library(m2b)
str(track_CAGA_005)
```

```
## 'data.frame':    3597 obs. of  5 variables:
##  $ x : num  26.3 26.3 26.3 26.3 26.3 ...
##  $ y : num  -33.8 -33.8 -33.8 -33.8 -33.8 ...
##  $ t : POSIXct, format: "2010-12-11 08:08:00" "2010-12-11 08:08:13" ...
##  $ b : chr  "3" "3" "3" "3" ...
##  $ id: chr  "CAGA_005" "CAGA_005" "CAGA_005" "CAGA_005" ...
```

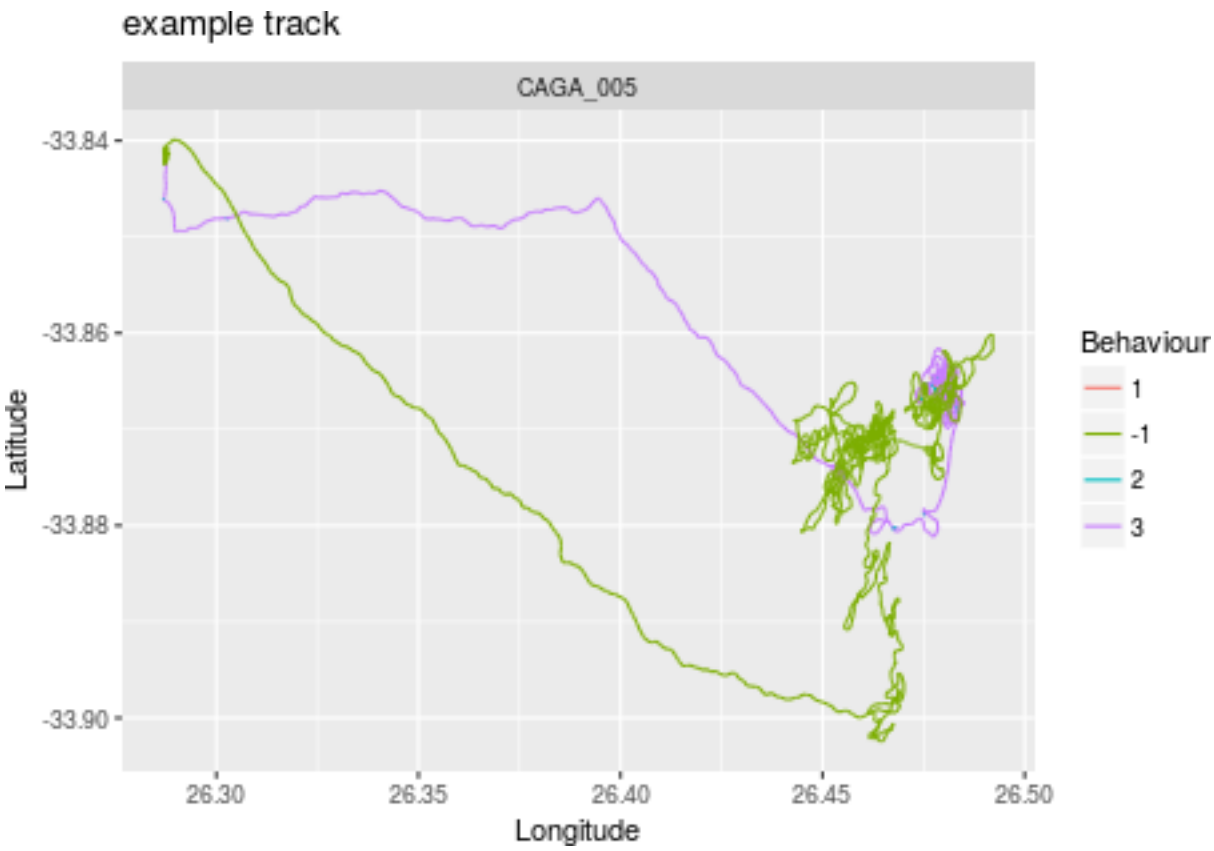
Different methods are available to build a `xytb` objec. Here, the tracking and behavioural data are directly taken from the dataframe, and the predictor variables deriving from the tracking data are computed at the same time using the function `xytb`. The variables are computed over sliding windows of sizes 3, 5, 7, 9, 11, 13 and 15 locations (the `winsize` parameter). In addition to the standard statistical operators (mean, standard deviation and median absolute deviation), the quantiles at 0, 25, 50, 75 and 100% are computed (the `idquant` parameter). All those values calculated can be then shifted in time to 5, 10 and 15 points backwards (the `move` parameter), if the user is interested to investigate the effect of the delay between the reaction of the animal captured in the movement data, and its behaviour as recorded by the observer. The rationale behind this operation is based on the fact that some changes in movement can be related to a change in behavioural state observed only later by the scientist. For example, an animal may react to something detected from a distance (like a possible feeding area), a change immediately captured in the movement data, but the reason for its movement (starting to feed) will only appear later in the observation data.

```
library(m2b)
#convert to xybt object with computation of windows operators and some quantiles
xytb<-xytb(track_CAGA_005,desc="example track",
     winsize=seq(3,15,2),idquant=seq(0,1,.25),move=c(5,10,15))
```

```
## [1] "Compute 169 indicators on 7 moving windows"
## [1] "Compute indicators on 3 points"
## [1] "Done"
## [1] "Compute indicators on 5 points"
## [1] "Done"
## [1] "Compute indicators on 7 points"
## [1] "Done"
## [1] "Compute indicators on 9 points"
## [1] "Done"
## [1] "Compute indicators on 11 points"
## [1] "Done"
## [1] "Compute indicators on 13 points"
## [1] "Done"
```

```
## [1] "Compute indicators on 15 points"
## [1] "Done"
## [1] "shift value backward"
## [1] "shift backward 5"
## [1] "shift backward 10"
## [1] "shift backward 15"
## [1] "Done"
```

```
#a simple plot method
plot(xytb)
```



Modelling

Model

To build a random forest predicting the behavioural states based on the movement information, the function `modelRF` is used. It's a simple wrapper calling the `randomForest` function of the `randomForest` package, using the behavioural observation as response, and movement information as predictors.

```
#a model (the function modelRF updates the model inside the xytb object)
xytb<-modelRF(xytb,type="actual",ntree=501,mtry=15)
```

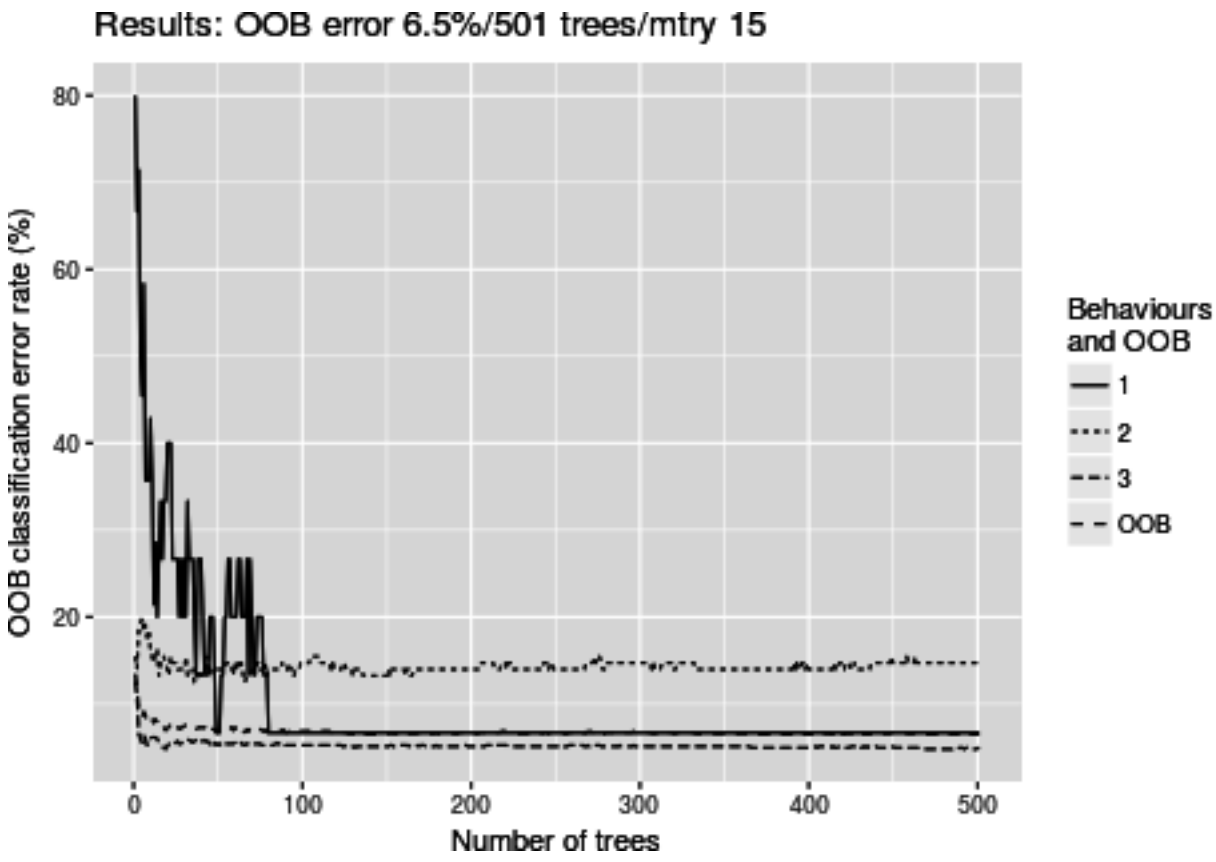
```
## [1] "removing lines with NA values"
## [1] "removing colinearity among predictors"
```

```
## [1] "v,dist,thetarel keeped"
## [1] "vmean_w3,distmean_w3,vmean_w5,distmean_w5,vmean_w7,thetarelmean_w7,distmean_w7,vsd_w7,distsd_w7
## [1] "removing near zero variance predictors"
```

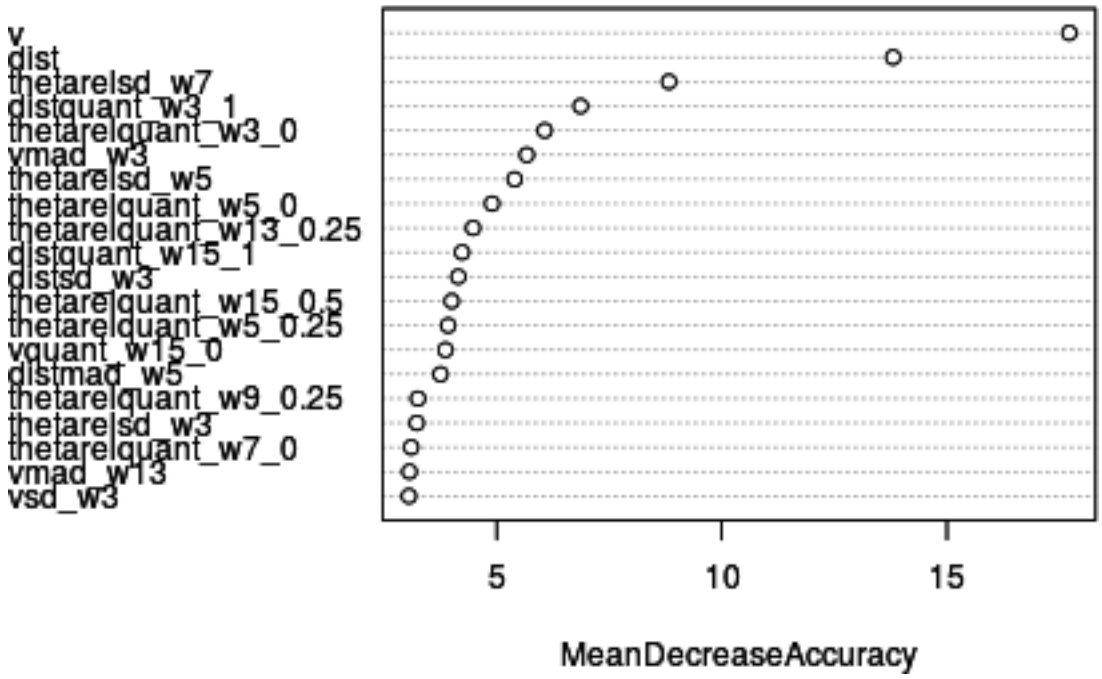
Results

Some diagnostic plots are available using the `resRF` function to check the fit of the model. In addition, the function `extractRF` can be used to export the resulting model to the `randomForest` format, so that other function from the `randomForest` package can be used to perform a deep analysis of the model.

```
resRF(xytb)
```




```
resRF(xytb,"importance")
```



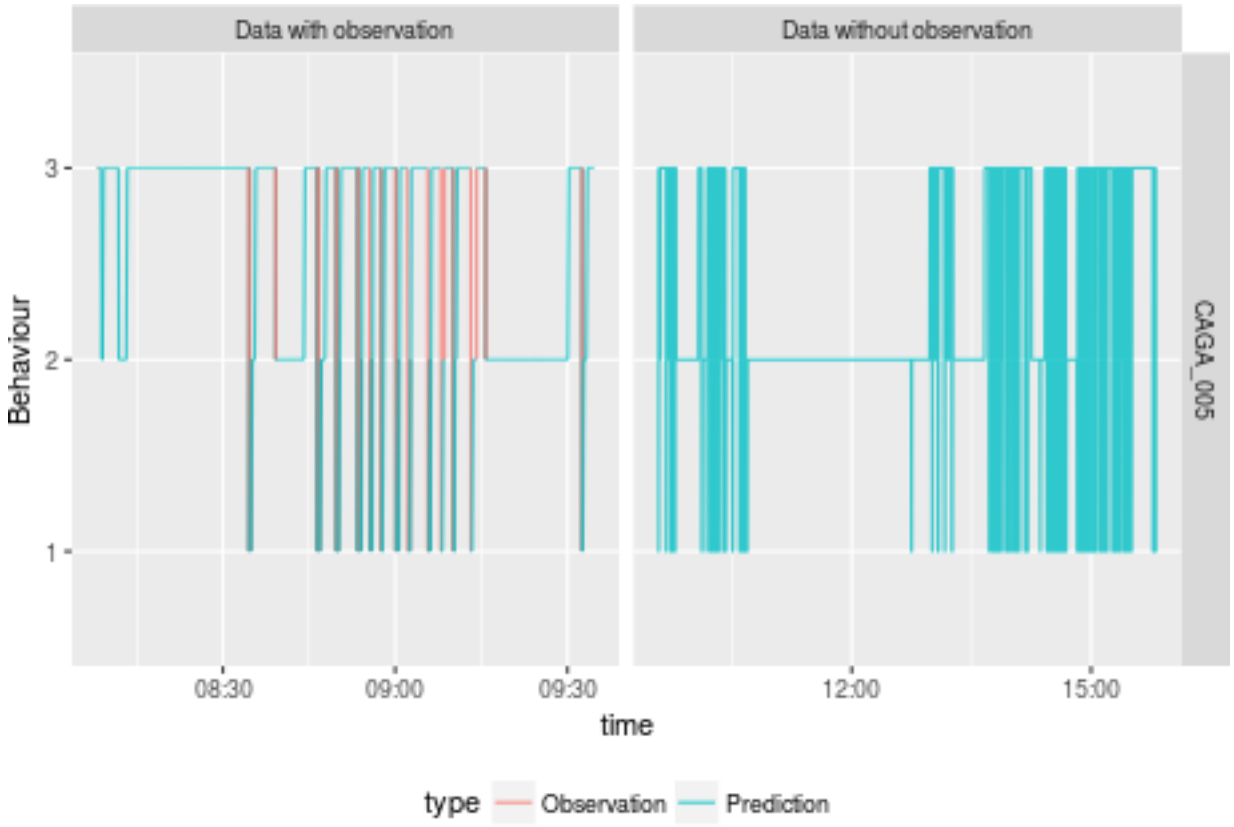

```
resRF(xytb,"confusion")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3
##          1  14  17  28
##          2   0 116   4
##          3   1   3 635
##
## Overall Statistics
##
##                Accuracy : 0.9352
##                  95% CI : (0.9161, 0.9511)
##     No Information Rate : 0.8154
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8079
##  Mcnemar's Test P-Value : 3.498e-09
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity           0.93333   0.8529   0.9520
## Specificity           0.94396   0.9941   0.9735
## Pos Pred Value        0.23729   0.9667   0.9937
## Neg Pred Value        0.99868   0.9713   0.8212
## Prevalence            0.01834   0.1663   0.8154
## Detection Rate        0.01711   0.1418   0.7763
```

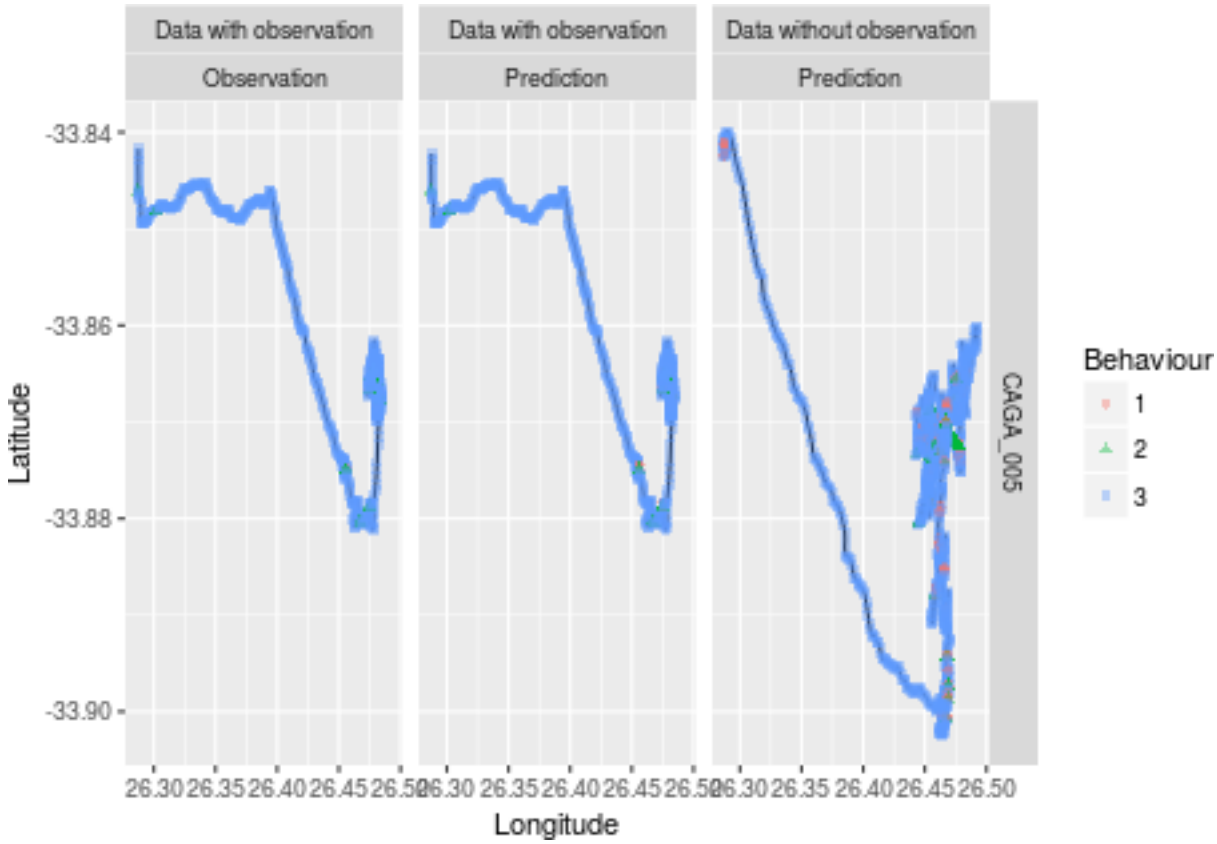
```
## Detection Prevalence  0.07213   0.1467   0.7812
## Balanced Accuracy     0.93865   0.9235   0.9628
```

The results regarding the behavioural states predicted vs the one observed are illustrated thanks to the `resB` functions.

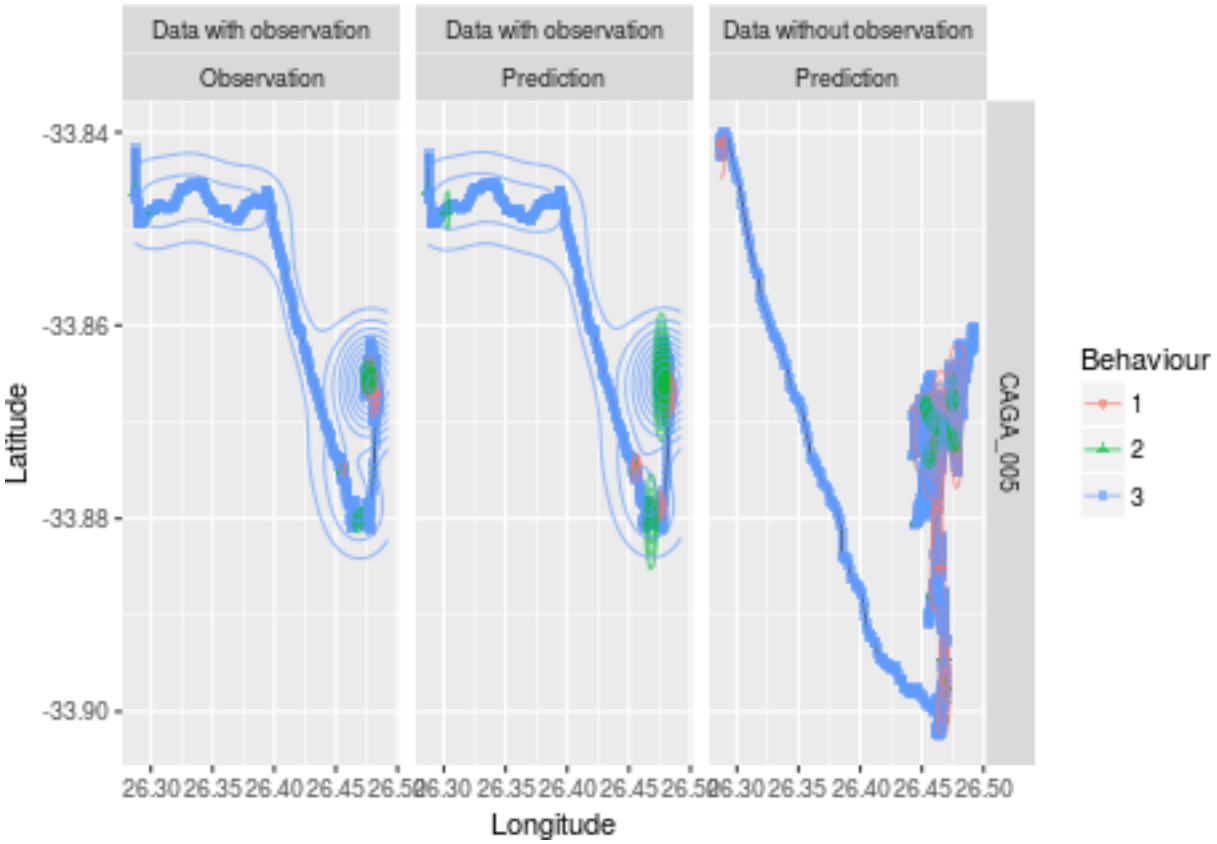
```
resB(xytb,"time",nob="-1")
```




```
resB(xytb,"space",nob="-1")
```




```
resB(xytb,"density",nob="-1")
```



Bibliography

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45: 5–32.

Calenge, Clément, Stéphane Dray, and Manuela Royer-Carenzi. 2008. "The Concept of Animals' Trajectories from a Data Analysis Perspective." *Ecological Informatics* In Press, Corrected Proof: –. http://www.sciencedirect.com/science/article/B7W63-4V28T4D-1/2/f122562e04e57b400dfd8b3858f30cf5.

Chen, Chao, Andy Liaw, and Leo Breiman. 2004. "Using Random Forest to Learn Imbalanced Data." Department of Statistics,UC Berkeley.

Michelot, Théo, Roland Langrock, and Toby A. Patterson. 2016. "moveHMM: An R Package for the Statistical Modelling of Animal Movement Data Using Hidden Markov Models." Edited by Greg McInerny. *Methods in Ecology and Evolution* 7 (11). Wiley-Blackwell: 1308–15. doi:10.1111/2041-210x.12578.