Vignette movescape

Guillaume Bastille-Rousseau

February 26, 2020

This vignette presents a simple workflow to extract the movescape of GPS tracked individuals. We recommend interest readers read the documentation associated to the *moveNT* package.

A- Data preparation

For simplification, we initiate the analysis with a simple trajectory object of class *ltraj*. This trajectory object is freely available in the *adehabitatLT* package and contains the GPS locations of 6 albatross. We will first use the *loop* function in *moveNT*, which is a wrapper function extracting the movement metrics for each individual. We then use the function *table_grid* to convert the *loop* object into a *data.frame*.

```r
library(moveNT)
```

```
##
## Attaching package: 'moveNT'
```

```
## The following objects are masked _by_ '.GlobalEnv':
##
##     adj2stack, clustnet, graphmet, interpolation, loop,
##     mosaic_network, quant, sim_mov, traj2adj, val
```

```r
data(albatross)
grid<-loop(albatross, 35000)
```

```
##  balise.11378
##  balise.11380
##  balise.16256
##  balise.25070
##  balise.8196
##  balise.8337
```

```r
#Stack showing all metrics calculated for the first albatros
plot(grid[[1]])
```
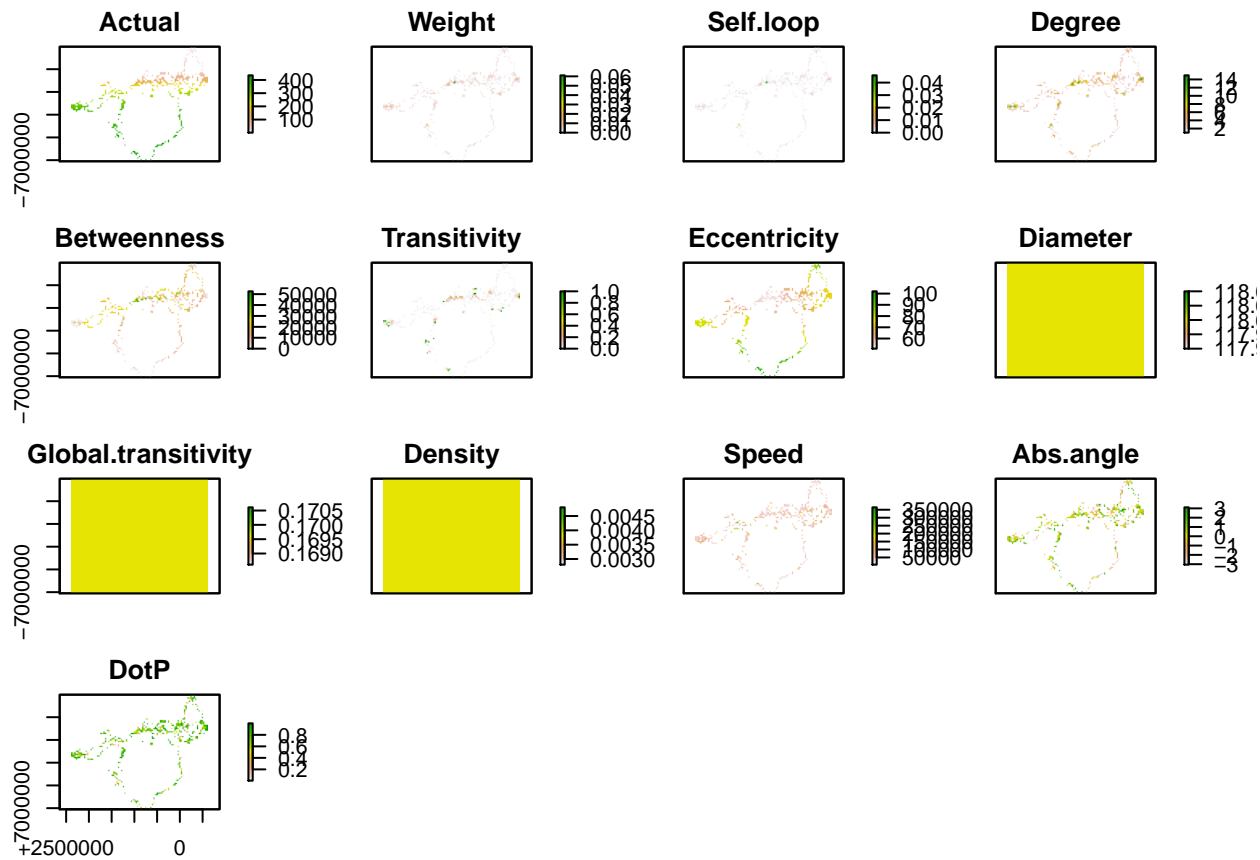
```r
table_grid<-table_cluster(albatross, grid)
#Showing the first few rows of the table created.
head(table_grid)
```

```
##     Actual      Weight     Self.loop Degree Betweenness Transitivity
## 75       1 0.001090513 0.000000000      2        2531            0
## 76       2 0.004362050 0.001090513      6        3692            0
## 77       3 0.009814613 0.008724100      5         382            0
## 78       4 0.001090513 0.000000000      2           0            0
## 160      5 0.001090513 0.000000000      2        2516            0
## 162      6 0.002181025 0.001090513      4        3707            0
##     Eccentricity Diameter Global.transitivity      Density    Speed
## 75            87      118           0.1696113 0.003803168 50784.60
## 76            88      118           0.1696113 0.003803168 32739.75
## 77            89      118           0.1696113 0.003803168 15622.52
## 78            90      118           0.1696113 0.003803168 26281.46
## 160           86      118           0.1696113 0.003803168 40014.98
## 162           89      118           0.1696113 0.003803168 24601.32
##      Abs angle      DotP            ID
## 75  -2.0312774 0.9703201  balise.11378
## 76  -0.4432067 0.6445507  balise.11378
## 77   0.1772387 0.6329915  balise.11378
## 78   3.0051036 0.7281993  balise.11378
## 160 -1.5583292 0.8902292  balise.11378
## 162  0.2959155 0.9960083  balise.11378
```

B- Individual-level clustering

The first step of the analysis is to apply the clustering to each individual. *ind_clust* apply a mixture model to each individual. It is possible to specify the maximum number of clusters (here 8) and also the covariates to use for the clustering, but the function automatically selects the optimal number of clusters (based on BIC). In our case, 2 individuals had 6 clusters, 2 had seven, and 2 had eight clusters. *ls_ind* simply return a list object with each element representing a single individual.

```
ls_ind<-ind_clust(table_grid, max.n.clust=8)
```

```
## [1] "mclust is loaded correctly"
## [1] " balise.11378"
## [1] " balise.11380"
## [1] " balise.16256"
## [1] " balise.25070"
## [1] " balise.8196"
## [1] " balise.8337"
#Showing the number of individuals with 6, 7, and 8 clusters
#   (i.e. no individual had less than 6 clusters)
table(unlist(lapply(ls_ind, function(x) x$G)))
```

```
##
## 6 7 8
## 2 2 2
```

C- Population-level clustering

After performing the individual clustering, a second clustering is applied via *pop_clust*. This second clustering takes the ouptut of *ind_clust* and will identify which individual clusters could be considered as one population-level clusters. The function automatically selects the optimal number of clusters (based on BIC). It is possible for two clusters from the same individual to be in the same population-level cluster. Likewise, it is possible that a population level cluster does not have all individuals. Here, 3 different population clusters were calculated. The second line extract the center (mean) of each cluster which is helpful in interpreting their meaning. The first cluster was heavily used (weight), well connected (degree), and important for connectivity (betweenness), but albatross were moving slowly and not linearly in them. The second cluster was a cluster with intermediate use, not important for connectivity and still with meandering movement. The third cluster was important for connectivty and albatross were moving fast and linearly in it. We also extract the proportion of each cluster.

```
pop<-pop_clust(albatross, ls_ind)
#Paremeters associated to each cluster (i.e. center)
pop[[1]]$parameters$mean
```

```
##                   [,1]        [,2]        [,3]
## Weight       2.0414281  0.07588085 -0.2203643
## Degree       2.1634587  0.70690604 -0.5993716
## Betweenness  0.5743463 -0.13990333  0.3104720
## Speed       -0.3411859 -0.39086569  0.7518956
## DotP        -0.1277000 -0.16974009  0.6275062
```

```
#Proportion of each cluster (how frequent it is spatially)
pop[[1]]$parameters$pro
```

```
## [1] 0.2868044 0.5703384 0.1428571
```

D- Mapping and results export

After performing the population level cluster, the function *clust_stack* recombines the individual and population level clustering and produce a *stack* object for each individual albatross showing the most likely cluster, and also the probability of observing each cluster (uncertainty) in any given pixel. This individual level data (but which contains the population level clustering) can be used in a regression based analysis as presented in the manuscript or simply mapped. We developed two functions to produce these maps. *pop_stack* generate for each population cluster, rasters showing if at least one individual is using this pixel for this specific cluster. *pop_overl* display for each pixel all potential use observed, for example a pixel having the value *123* will have at least one individual using this pixel as cluster 1, another individual using it as 2, and another individual as 3. We show how frequent each combination are using the *table* function. These object can be exported to be used in other software using the *writeRaster* function.

```
clust_stack<-clust_stack(grid, pop, ls_ind, table_grid)
```
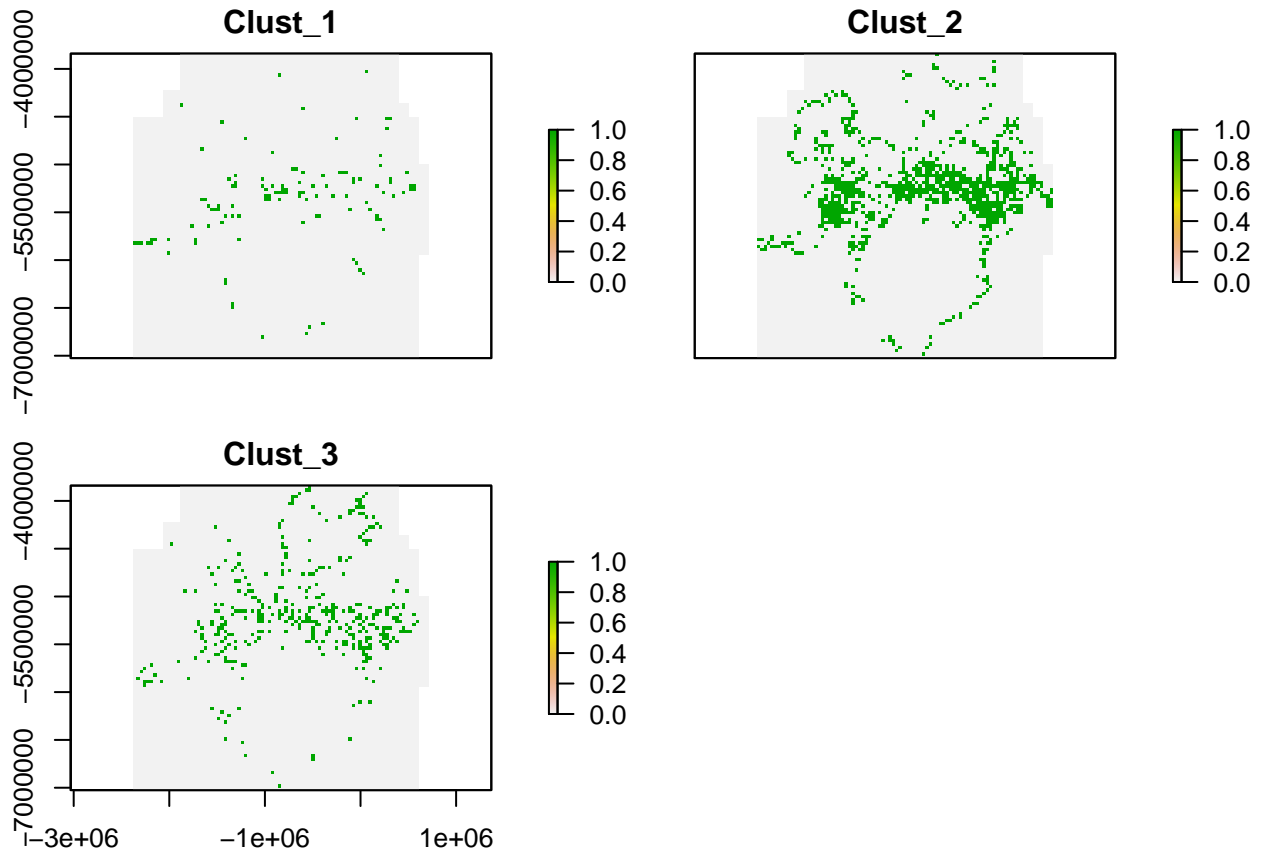
```
## [1] " balise.11378"
## [1] " balise.11380"
## [1] " balise.16256"
## [1] " balise.25070"
## [1] " balise.8196"
## [1] " balise.8337"
```

```
#Plotting the stack for the first individual
plot(clust_stack[[1]])
```
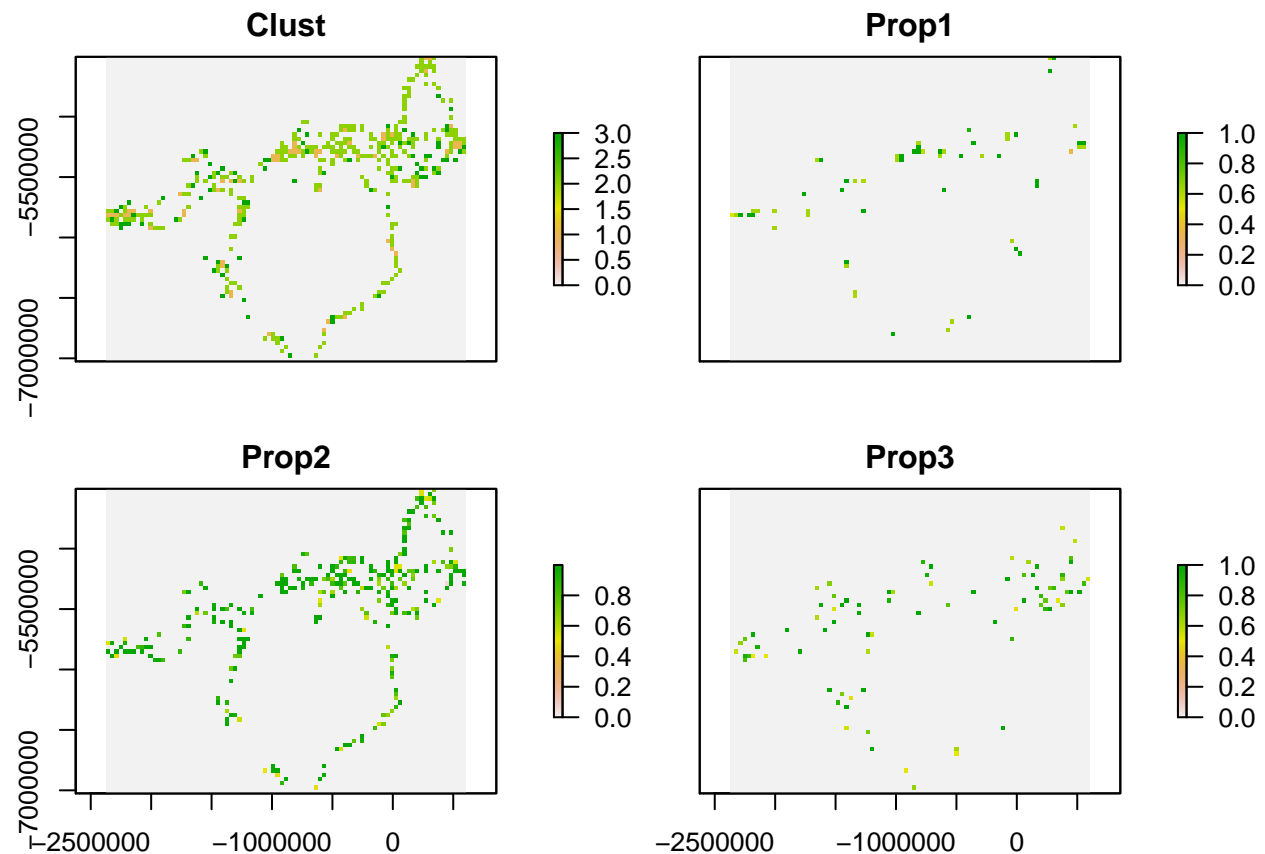


```
pop_stack<-pop_stack(clust_stack)
#Plotting the stack object. Each raster shows if at least one
```

4

```
# individual is using the pixel as a specific cluster.
plot(pop_stack)
```



```
#writeRaster(pop_stack, "Network.tif", format="GTiff",  bylayer=T, suffix="names", overwrite=T)
```

```
pop_overl<-pop_overl(clust_stack)
#Table showing how frequent each pixels is. A value of 123 indicates
# that at least one individual is using the pixel as 1, one individual
# is using it as 2, and one as 3
table(values(pop_overl))
```

```
##
##    0    1    2    3   12   13   23  123
## 6287   73  719  253   33    5  114    4
```

```
#Plotting of the overlap raster. The legend makes it hard to see,
# but values range from 0 (no animal present) to 123
# (each type of used is observed in the pixel).
plot(pop_overl)
```