Vignette moveNT

Guillaume Bastille-Rousseau

April 18, 2017

Simulating movement strategies - *sim_mov*

The function *sim_mov* generates movement trajectories including patches and movement between patches. Movement within patches can follow an Ornstein-Uhlenbeck process (based on *simm.mou* function from package *adehabitatLT*) or two-states movement model (based on *simmData* function from package *moveHMM*). Movement between patches is following a brownian bridge movement model (based on *simm.bb* function from package *adehabitatLT*). Generated outputs are of the class *ltraj* from package *adehabitatlt*.

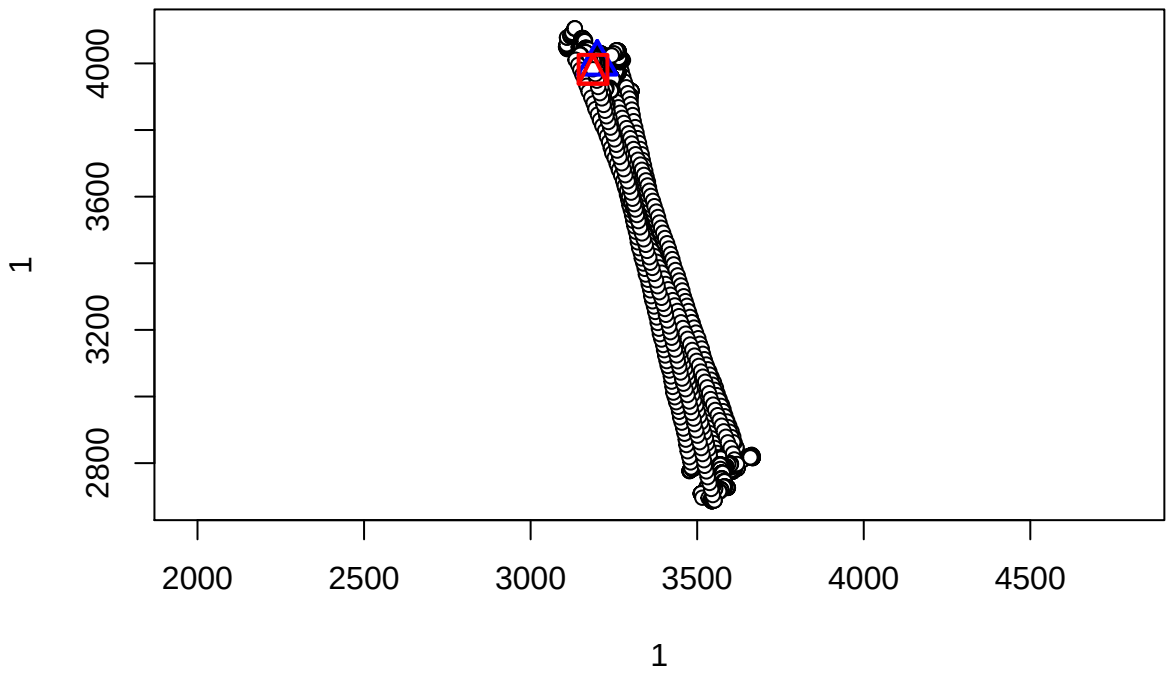
```
# Simulating migration with two-states model
mig<-sim_mov(type="2states", npatches=2, ratio=2, nswitch=25, ncore=150, grph=F)
mig
```

```
##
## ************ List of class ltraj ************
##
## Type of the traject: Type II (time recorded)
## * Time zone: GMT *
## Regular traject. Time lag between two locs: 1 seconds
##
## Characteristics of the bursts:
##   id burst nb.reloc NAs          date.begin            date.end
## 1 id    id     4200   0 1960-01-01 00:00:01 1960-01-01 01:10:00
##
##
##  infolocs provided. The following variables are available:
## [1] "out.Corri"
```

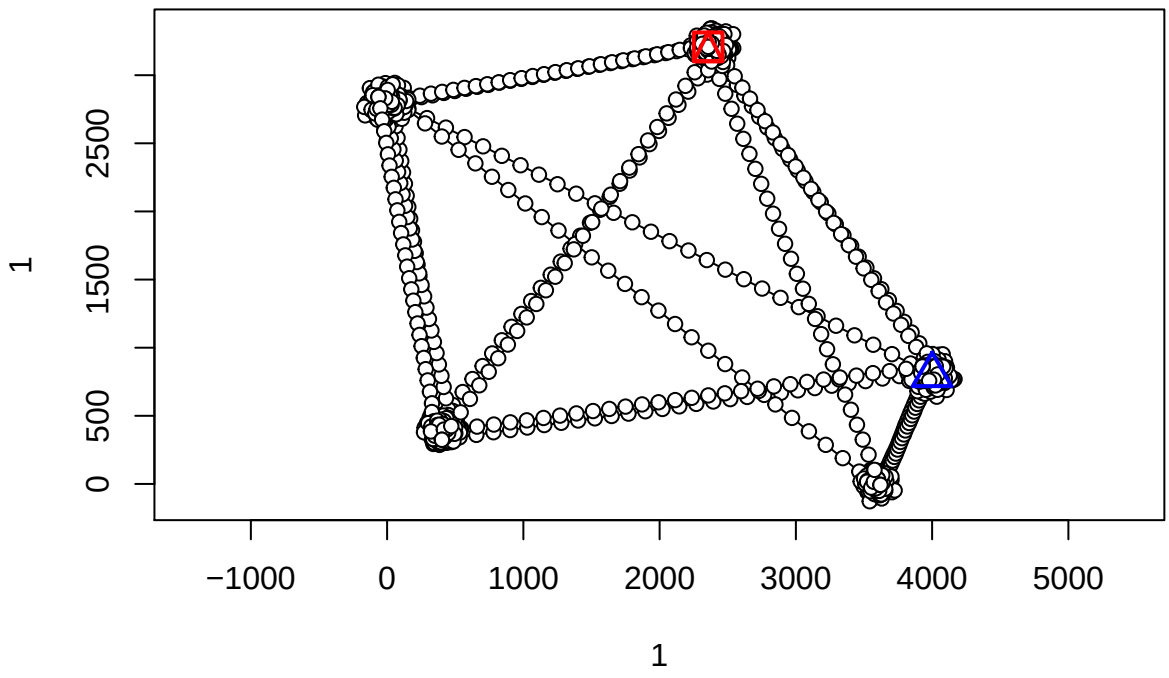
```
head(ld(mig))
```

```
##          x        y                date            dx            dy
## 1 3200.000 4000.000 1960-01-01 00:00:01  4.748067e-01  2.744838e-01
## 2 3200.475 4000.274 1960-01-01 00:00:02 -8.257575e-08 -2.074116e-07
## 3 3200.475 4000.274 1960-01-01 00:00:03 -1.188488e+00 -6.801594e-01
## 4 3199.286 3999.594 1960-01-01 00:00:04 -2.180047e-02  5.115861e-03
## 5 3199.265 3999.599 1960-01-01 00:00:05 -3.497203e-01  1.677394e-01
## 6 3198.915 3999.767 1960-01-01 00:00:06 -2.244024e+00  2.194760e+00
##           dist dt       R2n  abs.angle  rel.angle id burst out.Corri
## 1 5.484367e-01  1 0.0000000  0.5241578         NA id    id         2
## 2 2.232450e-07  1 0.3007828 -1.9496853 -2.4738431 id    id         2
## 3 1.369350e+00  1 0.3007826 -2.6217976 -0.6721124 id    id         2
## 4 2.239269e-02  1 0.6739139  2.9110958 -0.7502918 id    id         2
## 5 3.878670e-01  1 0.7013818  2.6943662 -0.2167296 id    id         2
## 6 3.138887e+00  1 1.2318691  2.3672925 -0.3270738 id    id         2
```

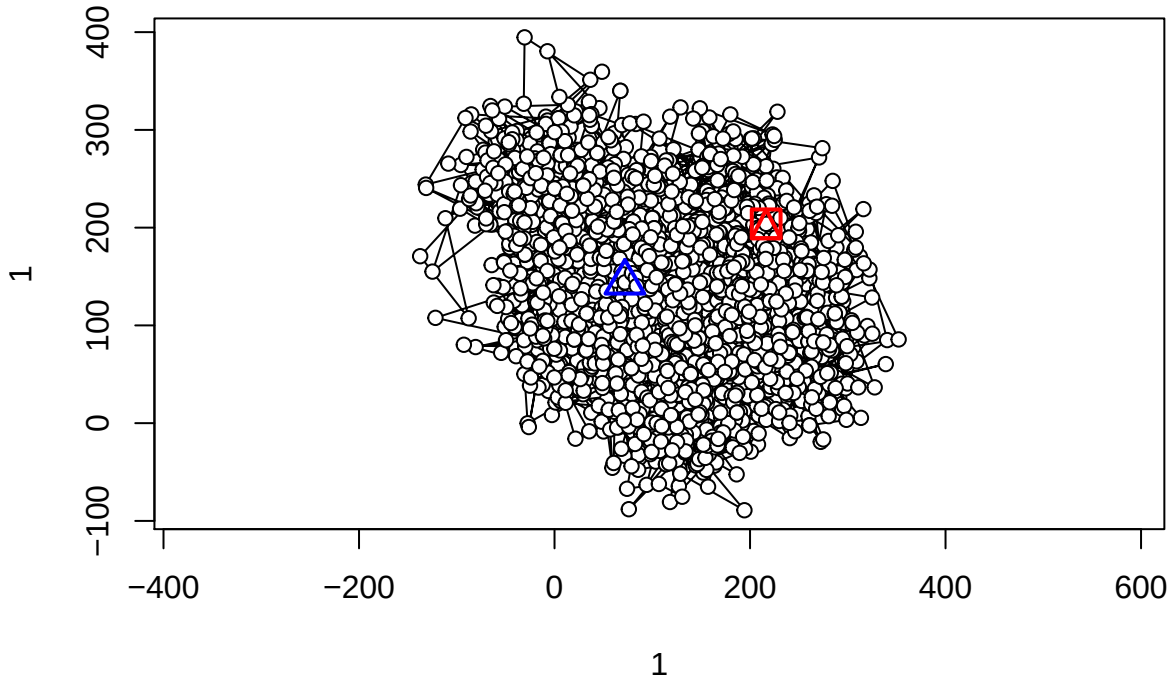
```
plot(mig)
```




```
# Simulating multi-patches movement with Ornstein-Uhlenbeck process
patches<-sim_mov(nswitch=25, ncore=150, ratio=5, type="OU", npatches=5, grph=T)
```




```
# Simulating sedentary movement
seden<-sim_mov(type="OU", npatches=10, spacecore=12, ratio=3, nswitch=150, ncore=20, grph=T)
```



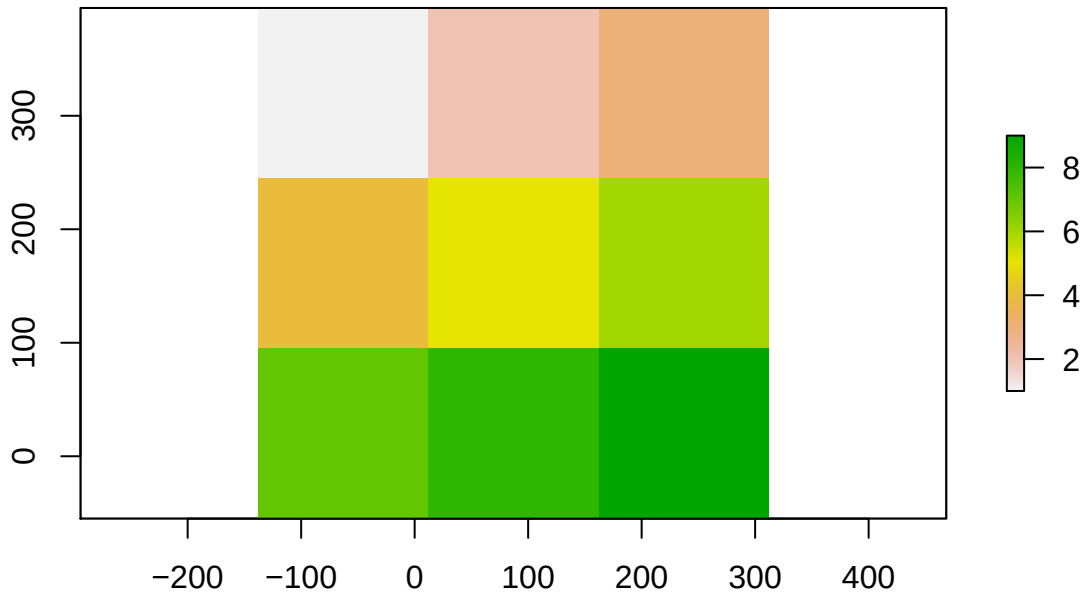
Converting movement to adjacency matrix - *traj2adj*

The function *traj2adj* converts a trajectory object of class *ltraj* to an adjacency matrix. This is done by overlapping a grid over the relocation data and tallying the number of transitions among each pixel. Users need to specify the grid size, which can be based on distance travelled. The function *quant* is a wrapper that allows to sample a quantile of step length distribution from a *ltraj* object. Output produced by *traj2adj* is a list containing the adjacency matrix, the grid used (raster format), and a raster indicating pixel numbers that are occupied. These rasters are used by other functions such as *adj2stack* and *clustnet*.

```
# Using sedentary movement and user specific grid-size
adj_seden<-traj2adj(seden, res=150) #Pixel size of 150m
adj_seden[[1]] # Adjency matrix
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
##  [1,]   56   16    0   31   19    0    0    0    0
##  [2,]   21   39    8   14   43    8    0    0    0
##  [3,]    0   12   16    0    8    9    0    0    0
##  [4,]   35   16    0  115   82    0   12   14    0
##  [5,]   10   46    5   87  780   93   11  106   19
##  [6,]    0    4   17    0   94  545    0   14   84
##  [7,]    0    0    0   16   11    0   15   12    0
##  [8,]    0    0    0   11   99   19   16  453   82
##  [9,]    0    0    0    0   20   84    0   81  249
```

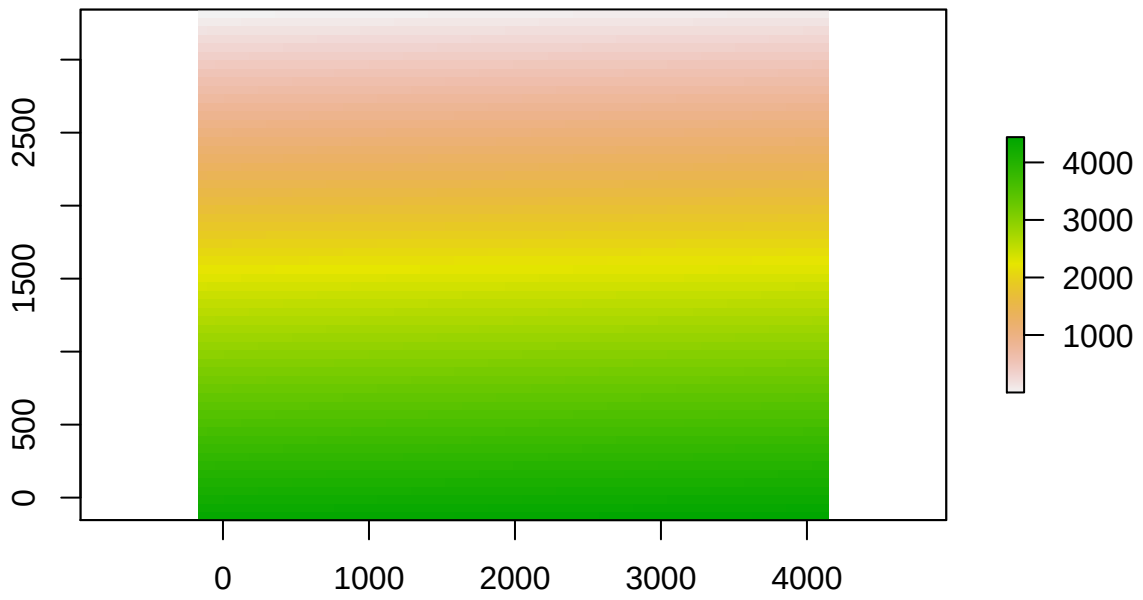
```r
plot(adj_seden[[2]]) #Plot grid used
```



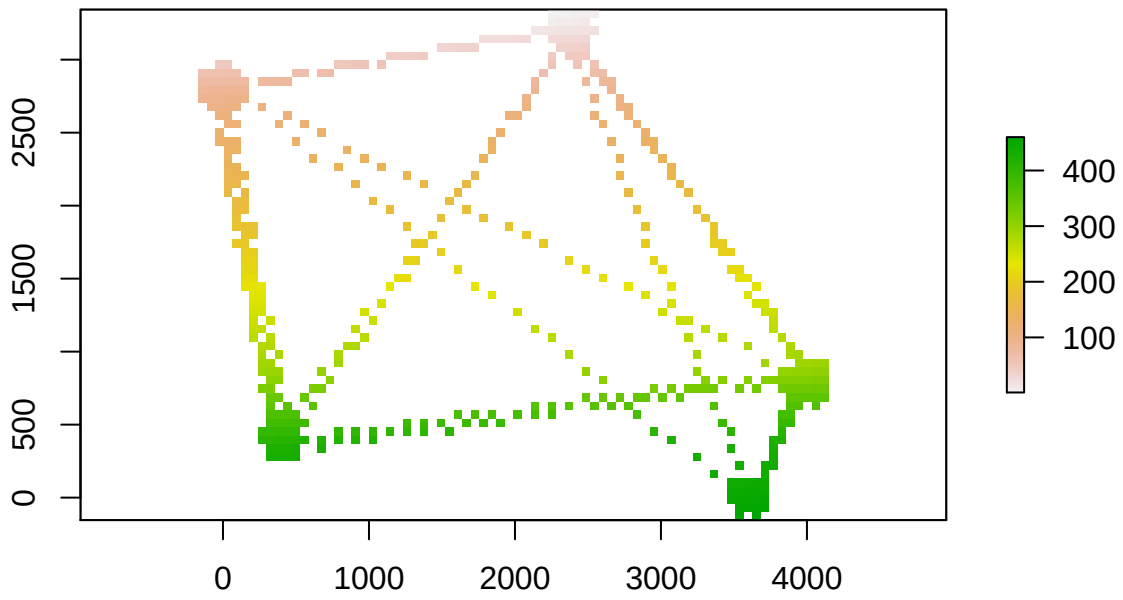

```r
# Using multi-patches movement and median distance travelled
adj_patches<-traj2adj(patches, res=quant(patches, p=0.5)) #Grid size based on median
dim(adj_patches[[1]]) # Size of the adjacency matrix
```

```
## [1] 460 460
```

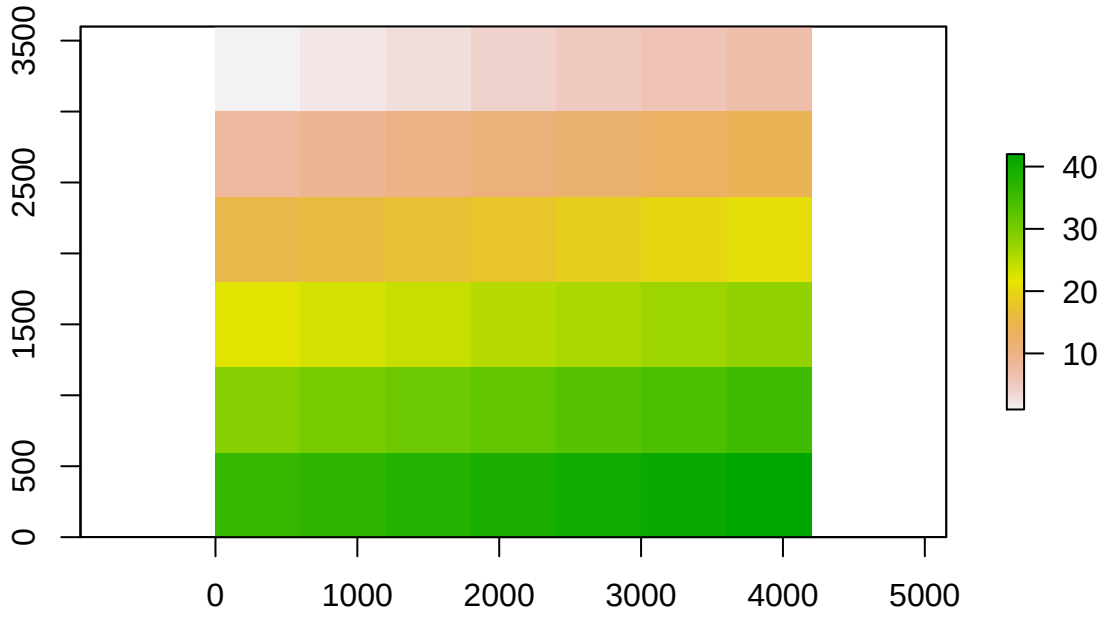
```r
plot(adj_patches[[2]]) #Plot grid used
```




```
plot(adj_patches[[3]]) #Plot occupied pixels
```



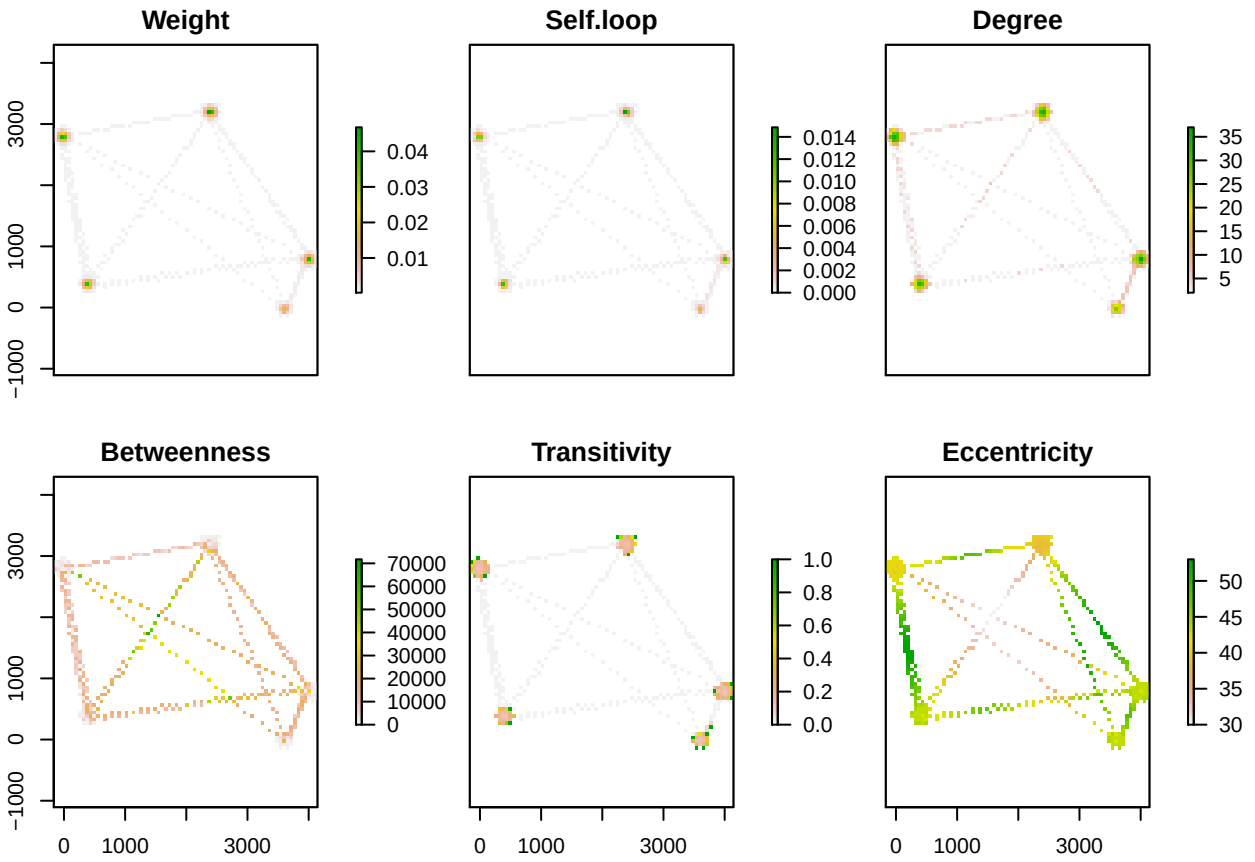

```
# Using user defined grid
ras<-raster(nrows=10, ncols=10, xmn=0, ymn=0, xmx=6000, ymx=6000)
adj_patches2<-traj2adj(patches, res=quant(patches, p=0.5), grid=ras) #Grid size based on median
plot(adj_patches2[[2]]) #Crop version of the grid created
```



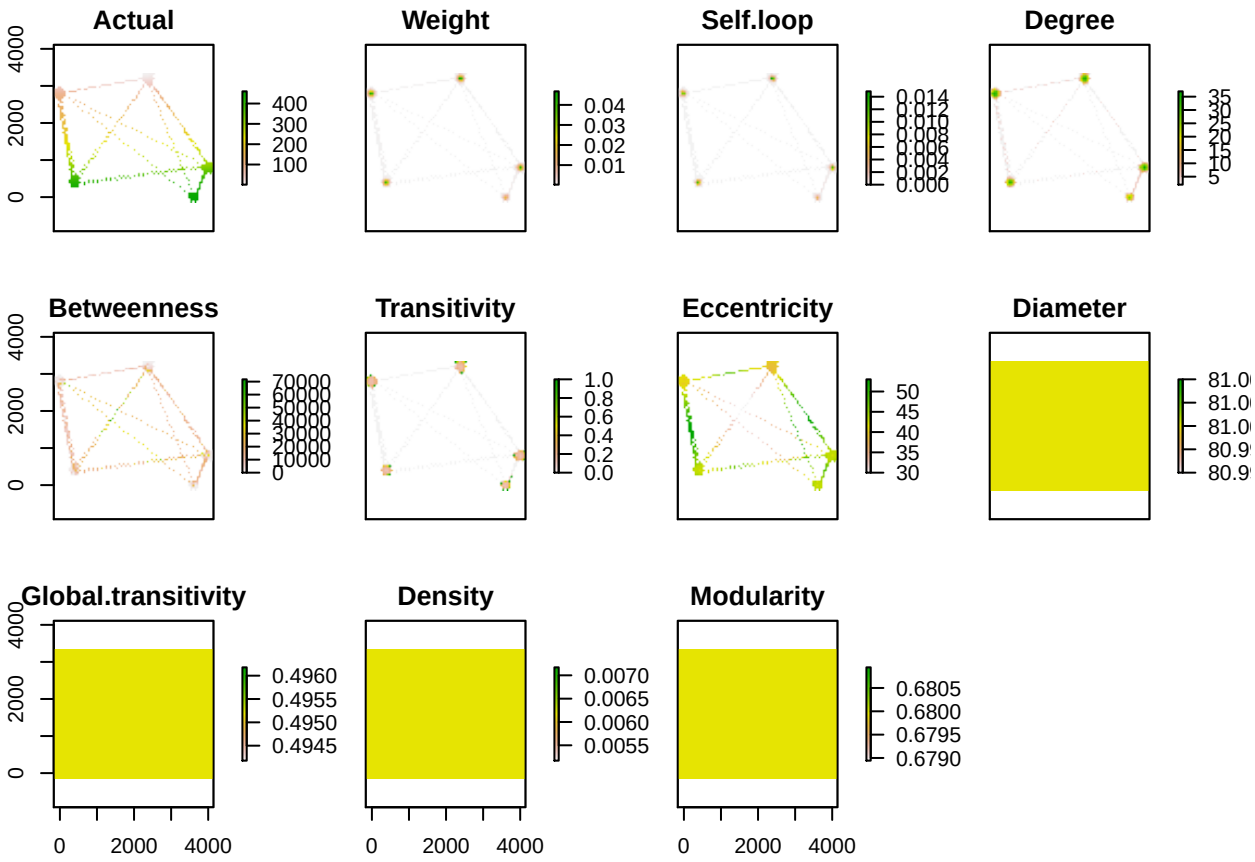
Calculation of network metrics - *adj2stack*

The function *adj2stack* takes the output of function *traj2adj* and calculates a series of node- and graph-level metrics. Each metric is stored as a individual raster and the output is a raster stack combining each metric. Graph-level metrics are also stored as a raster, each containing an unique value. The function *graphmet* extracts graph-level metrics. The function *val* extracts only the occupied cells (remove NA) in a raster and allows the calculation of statistics from node-level metrics.

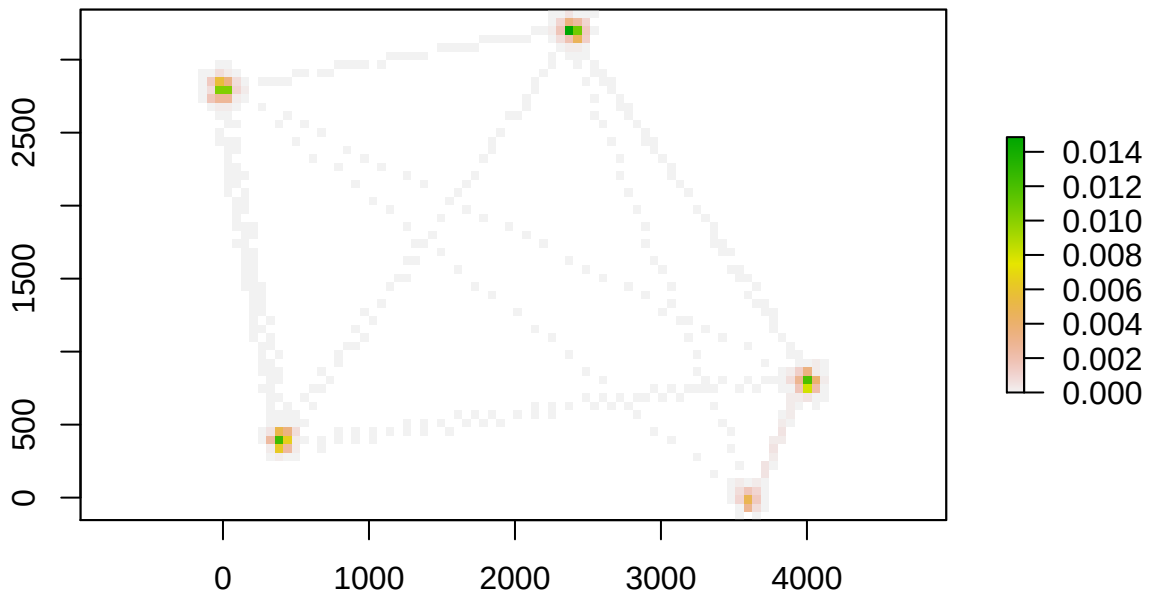
```
# Using multi-patches movement and median distance travelled
stck<-adj2stack(adj_patches,grph=T) #Plot the node-level metrics at the same time
```




```
plot(stck) #Plot also the graph-level metrics (not really useful)
```




```
plot(stck[[3]]) #Plot only one metric (degree)
```




```
graphmet(stck) # Extract graph-level metrics
```

```
##            Diameter Global.transitivity             Density
##        81.000000000         0.495178951         0.006171261
##          Modularity
##         0.679942429
```

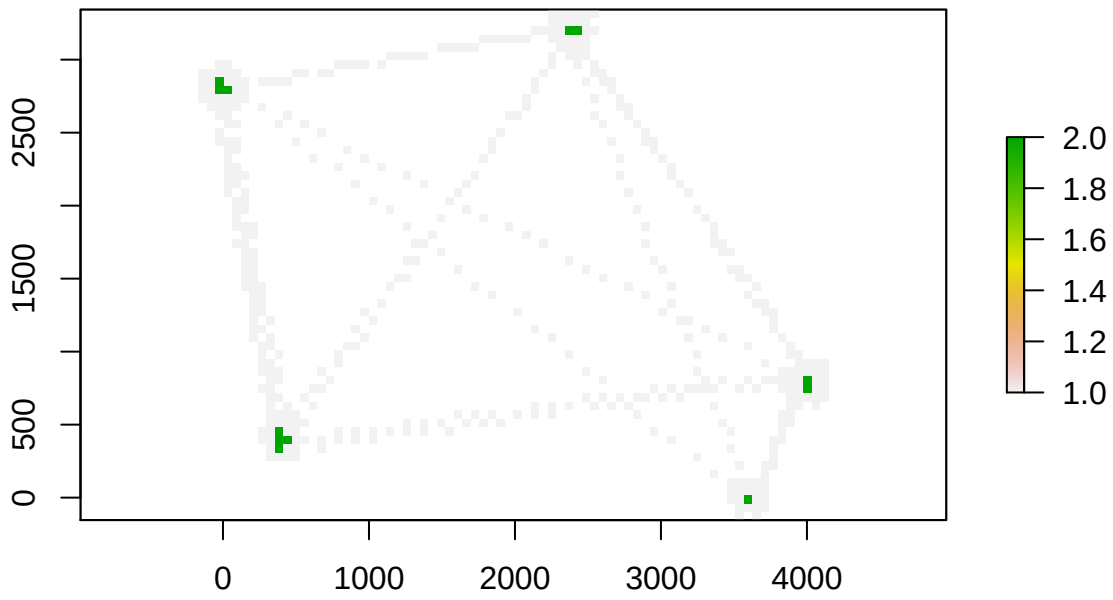
```
cv(val(stck, 4)) #Extract coefficient of variation of node-level betweenness.
```

```
## [1] 129.5666
```

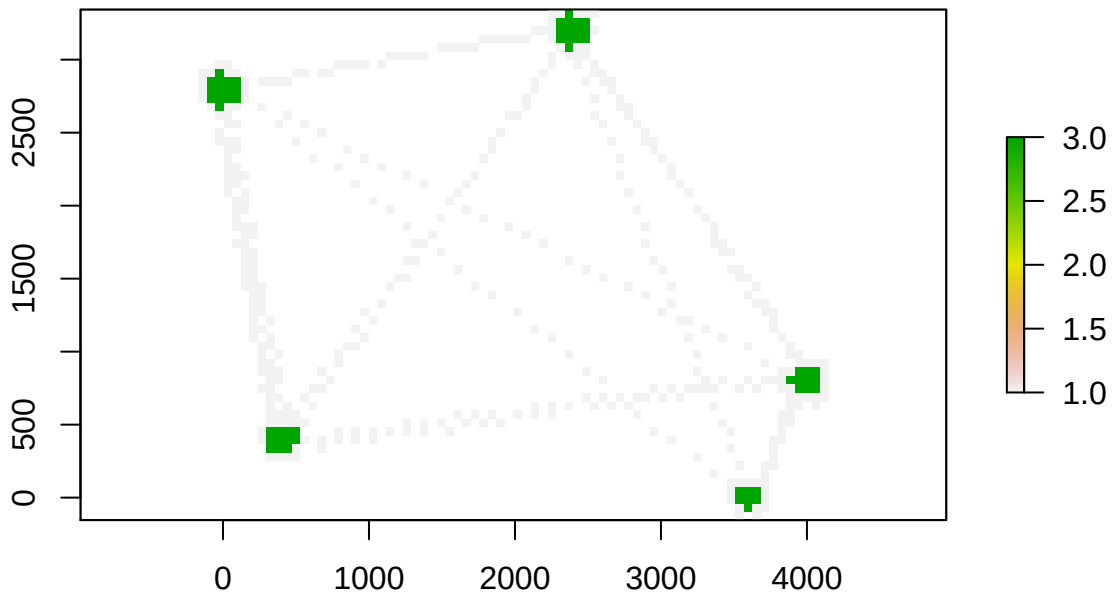
Clustering of node level metrics - *clustnet*

The function *clustnet* applies a normal mixture model to node-level metrics in order to cluster them into separate groups (default = 2). The function takes the output of function *adj2stack* with the user specifying the metric to cluster and the number of groups. Return a list containing output of function *Mclust* from package *mclust* and a raster displaying classification.

```
# Using multi-patches movement and median distance travelled
clust2<-clustnet(stck, id=3, nclust=2) # Clustering of degree in two groups
```



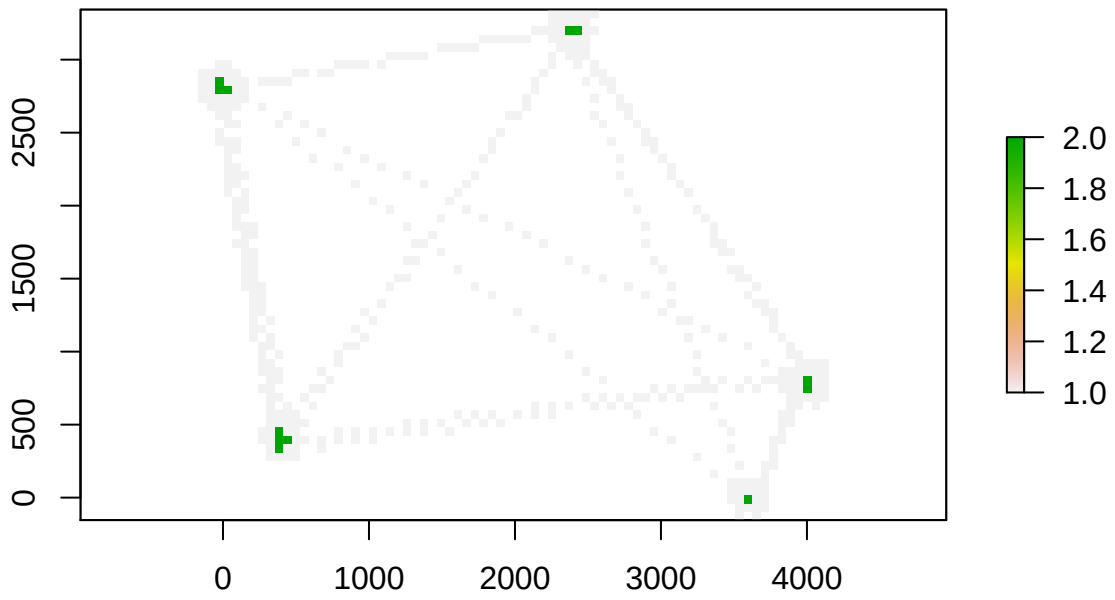

```
clust3<-clustnet(stck, id=4, nclust=3) #Clustering of betweenness in three groups
```




```
summary(clust2[[1]])
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust E (univariate, equal variance) model with 2 components:
##
##  log.likelihood   n df    BIC      ICL
##        2585.313 460  4 5146.1 5144.634
##
## Clustering table:
##   1   2
## 448  12
```

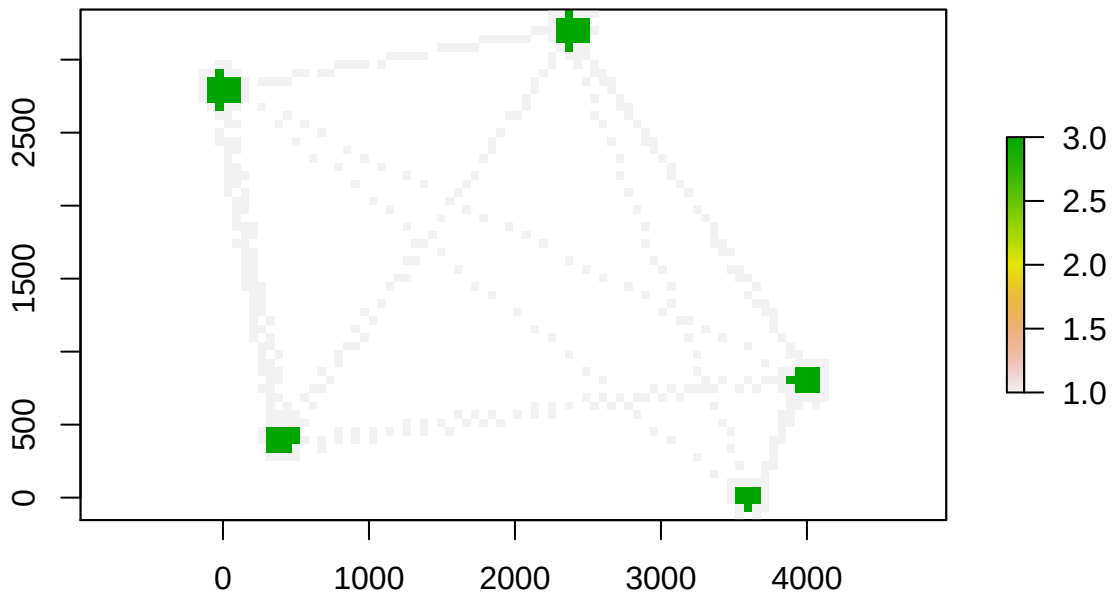
```
plot(clust2[[2]])
```




```
summary(clust3[[1]])
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust E (univariate, equal variance) model with 3 components:
##
##  log.likelihood   n df       BIC       ICL
##       -1305.045 460  6 -2646.877 -2859.403
##
## Clustering table:
##   1   2   3
## 404   0  56
```

```
plot(clust3[[2]])
```


0
500
1500
2500
0
1000
2000
3000
4000
3.0
2.5
2.0
1.5
1.0