storm_petrel_movement_code.R

akane

Thu Oct 20 17:30:12 2016

```
# Storm Petrel Movement Code
```

```
# clean everything first
rm(list=ls())

#load required libraries
library(adehabitatLT)
```

```
## Loading required package: sp
```

```
## Loading required package: ade4
```

```
## Loading required package: adehabitatMA
```

```
## Loading required package: CircStats
```

```
## Loading required package: MASS
```

```
## Loading required package: boot
```

```
library(geosphere)
library(moveHMM)
```

```
## Warning: package 'moveHMM' was built under R version 3.2.5
```

```
library(rworldmap)
```

```
## Warning: package 'rworldmap' was built under R version 3.2.5
```

```
## ### Welcome to rworldmap ###
```

```
## For a short introduction type :   vignette('rworldmap')
```

```
library(maps)       # Provides functions that let us plot the maps
```

```
##
##  # ATTENTION: maps v3.0 has an updated 'world' map.        #
##  # Many country borders and names have changed since 1990. #
##  # Type '?world' or 'news(package="maps")'. See README_v3. #
```

```
library(mapdata)    # Contains the hi-resolution points that mark out the countries
library(move)
```

```
## Loading required package: raster
```

```
##
## Attaching package: 'raster'
```

```
## The following objects are masked from 'package:MASS':
##
##     area, select
```

```
## The following object is masked from 'package:adehabitatMA':
##
##     buffer
```

```
## Loading required package: rgdal
```

```
## rgdal: version: 1.1-3, (SVN revision 594)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.0.1, released 2015/09/15
##  Path to GDAL shared files: C:/Users/akane/Documents/R/win-library/3.2/rgdal/gdal
##  GDAL does not use iconv for recoding strings.
##  Loaded PROJ.4 runtime: Rel. 4.9.1, 04 March 2015, [PJ_VERSION: 491]
##  Path to PROJ.4 shared files: C:/Users/akane/Documents/R/win-library/3.2/rgdal/proj
##  Linking to sp version: 1.2-1
```

```
##
## Attaching package: 'move'
```

```
## The following object is masked from 'package:adehabitatLT':
##
##     burst
```

```
library(RNCEP)

setwd("C:\\Users\\akane\\Desktop\\Science\\Manuscripts\\Storm Petrels\\Tracking Data")
data <- read.table("allStormies.csv", header=T,sep=",")
head(data)
```

```
##   day month year hour minute second        A B latitude longitude     C
## 1  23     8   16    5      0     53 18053.18 5 53.60396 -10.62676 42.75
## 2  23     8   16    5     30     51 19851.36 9 53.63333 -10.79087 56.75
## 3  23     8   16    6      0     55 21655.14 9 53.64509 -10.98963 45.25
## 4  23     8   16    6     30     55 23455.33 9 53.69830 -11.20415 53.75
## 5  23     8   16    7      5      8 25508.12 8 53.73100 -11.34850 57.50
## 6  23     8   16    7     35     12 27312.31 8 53.76850 -11.38927 59.50
##          D        E battery  ID  DateTimeFormula          DateTime
## 1 9999.999 7.52e-07    4.10 900 23/08/2016 05:00 23-08-16 05:00:00
## 2   17.270 2.91e-06    4.08 900 23/08/2016 05:30 23-08-16 05:30:00
```

```
## 3   17.185 4.68e-06    4.08 900 23/08/2016 06:00 23-08-16 06:00:00
## 4   17.190 7.81e-06    4.10 900 23/08/2016 06:30 23-08-16 06:30:00
## 5   17.225 5.91e-06    4.08 900 23/08/2016 07:05 23-08-16 07:05:00
## 6   17.230 3.34e-06    4.08 900 23/08/2016 07:35 23-08-16 07:35:00
##   bathymetry identity
## 1  -117.2506      900
## 2  -122.5151      900
## 3  -144.6082      900
## 4  -174.3880      900
## 5  -223.7407      900
## 6  -239.6527      900
```

```
data<-data[,c("latitude","longitude","DateTime", "ID","bathymetry")]
names(data)[names(data) == 'longitude'] <- 'lon'
names(data)[names(data) == 'latitude'] <- 'lat'
# the time stamp can be a pain - set the column in excel using dd-mm-yy hh:mm:ss
data$DateTime<-as.POSIXct(data$DateTime, format= "%d-%m-%y %H:%M", tz = "UTC")
head(data)
```

```
##        lat       lon            DateTime  ID bathymetry
## 1 53.60396 -10.62676 2016-08-23 05:00:00 900  -117.2506
## 2 53.63333 -10.79087 2016-08-23 05:30:00 900  -122.5151
## 3 53.64509 -10.98963 2016-08-23 06:00:00 900  -144.6082
## 4 53.69830 -11.20415 2016-08-23 06:30:00 900  -174.3880
## 5 53.73100 -11.34850 2016-08-23 07:05:00 900  -223.7407
## 6 53.76850 -11.38927 2016-08-23 07:35:00 900  -239.6527
```

```
length(data$lat)
```

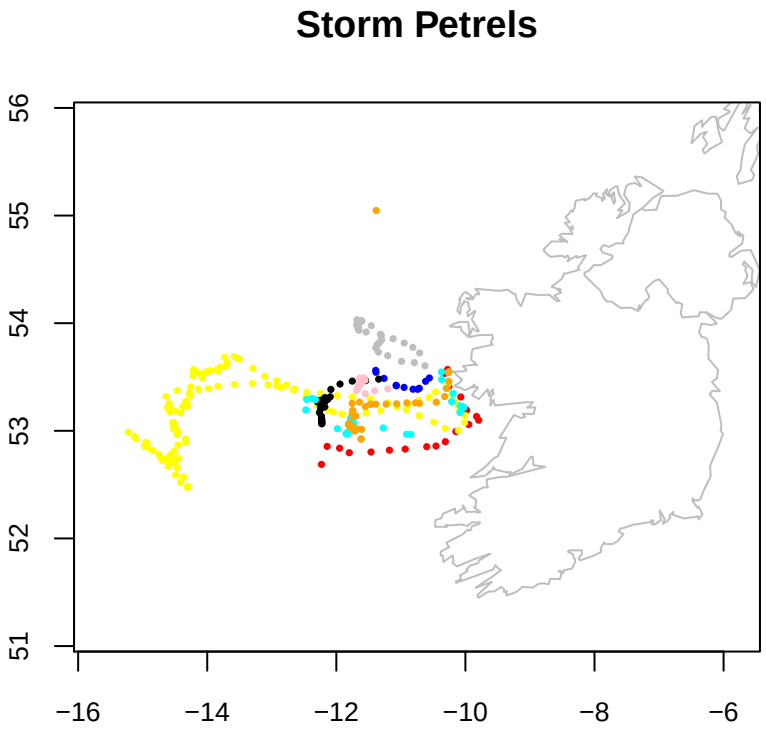
```
## [1] 402
```

```
# remove missing data
data<-data[ ! data$lat %in% 0, ]
length(data$lat)
```

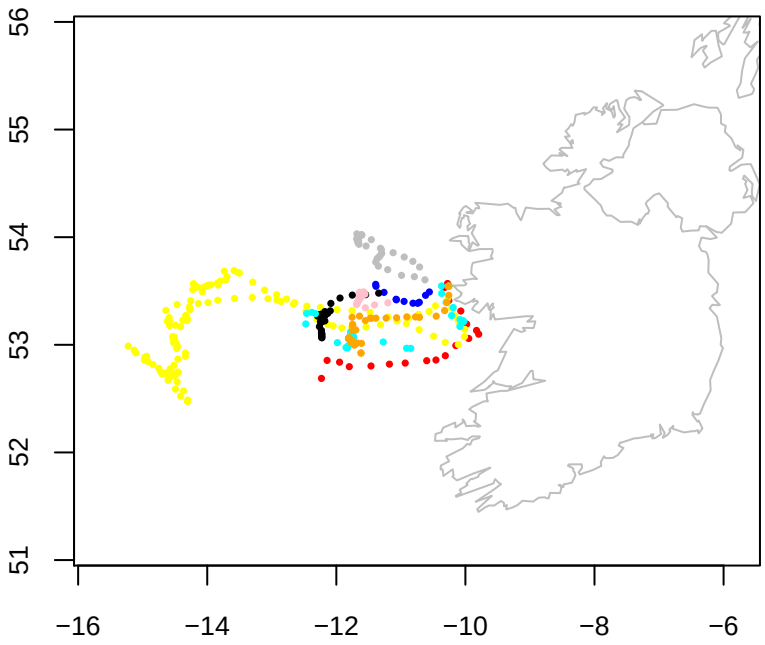
```
## [1] 311
```

```
# plot the data
```

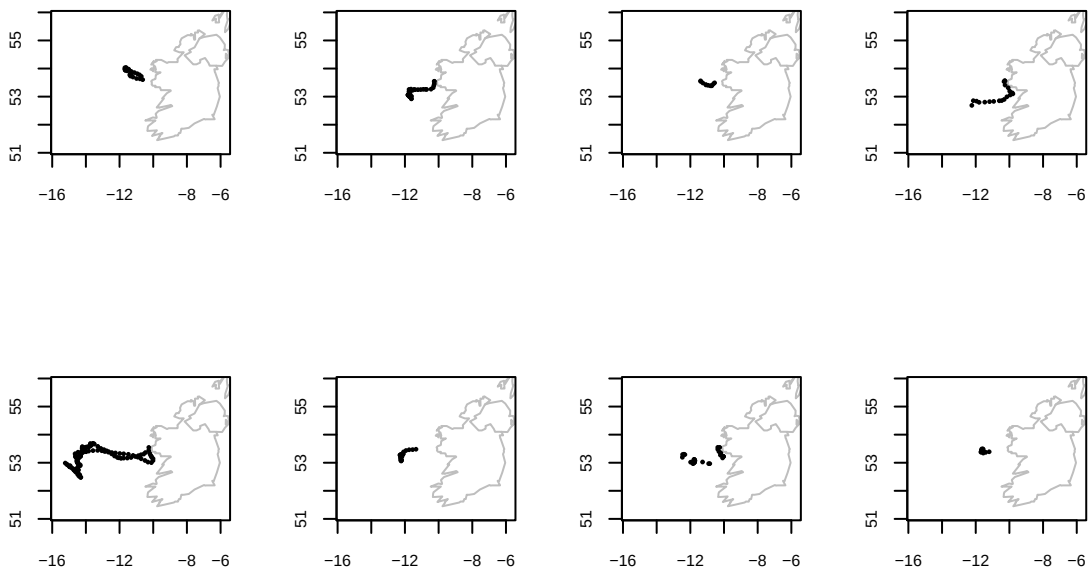
```
# specify the colours
palette(c("grey","orange","blue","red","yellow","black","cyan","pink"))

map('worldHires', c('Ireland', 'UK'),
    xlim=c(-16,-5.5),
    ylim=c(51,56))
points(data$lon,data$lat,col=data$ID,pch=16, cex=0.5, map.axes(cex.axis=0.8),title("Storm Petrels"),
       xlab="longitude",ylab="latitude")
```




```
# remove erroneous point
data<-data[data$lat < 54.5, ]

# replot the data
map('worldHires', c('Ireland', 'UK'),
    xlim=c(-16,-5.5),
    ylim=c(51,56))
points(data$lon,data$lat,col=data$ID,pch=16, cex=0.5, map.axes(cex.axis=0.8),title("Storm Petrels"),
       xlab="longitude",ylab="latitude")
```

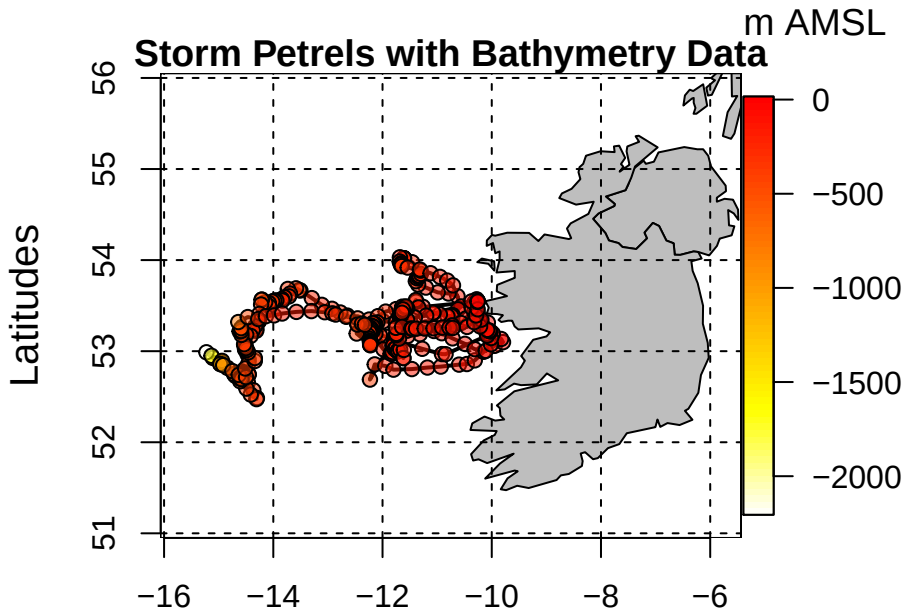



```
# alternatively, plot each of the bird tracks on a separate map
mapFunc <- function(dat) {
  map('worldHires', c('Ireland', 'UK'), xlim=c(-16,-5.5), ylim=c(51,56))
  points(dat$lon, dat$lat,pch=16, cex=0.5, map.axes(cex.axis=0.8),
         xlab="longitude",ylab="latitude")
}

op <- par(mfrow = c(2,4),
          oma = c(5,4,0,0) + 0.1,
          mar = c(0,0,1,1) + 0.1)
#birdID<-as.factor(data$ID)
sapply(split(data[1:2], data$ID), mapFunc)
```




```
## $`900`
## NULL
##
## $`902`
## NULL
##
## $`906`
## NULL
##
## $`906B`
## NULL
##
## $`908`
## NULL
##
## $`909`
## NULL
##
## $`909B`
## NULL
##
## $`910`
## NULL
```

```
# plot the tracking data with bathymetry data
par(mfrow = c(1,1))
NCEP.vis.points(wx=data$bathymetry, lats=data$lat, lons=data$lon,cols=rev(heat.colors(64)),
                title.args=list(main="Storm Petrels with Bathymetry Data"), points.args=list(cex=1),
                  image.plot.args=list(legend.args=list(text="m AMSL",adj=0, padj=-2, cex=1.15)),
                    map.args=list(xlim=c(-16,-5.5), ylim=c(51,56)))
```




```
# convert into a 'move' type file
```

```
movedata <- move(x=data$lon, y=data$lat,
             time=data$DateTime,
             data=data, proj=CRS("+proj=longlat +ellps=WGS84"), animal=data$ID)
movedata
```

```
## class       : MoveStack
## features    : 310
## extent      : -15.22459, -9.793738, 52.47343, 54.03079  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +ellps=WGS84
## variables   : 6
## names       :      lat,        lon,            DateTime,   bathymetry, individual.local.identifier,
## min values  : 52.47343, -10.011777, 2016-08-21 05:00:00, -1003.064883,                         X900,
## max values  : 54.03079,  -9.980319, 2016-08-29 04:18:00,    -1.269246,                         X910,
## timestamps  : 2016-08-21 05:00:00 ... 2016-08-29 04:18:00 Time difference of 8 days  (start ... end,
## sensors     : unknown
```

```
## indiv. data : ID
## min ID Data : 900
## max ID Data : 910
## individuals : X900, X906, X906B, X908, X909, X909B, X910, X902
## date created: 2016-02-04 02:26:00
```

```
summary(movedata)
```

```
## Loading required namespace: circular

## $X900
## $X900$X900
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 180065.4 15364.94 26.63112  84.11312 6430.906 5263.414 14.29843
##
## $X900$X900
##         Duration   AverDur     SDDur  dupl multseason
## 1 16.08333 hours 0.5744048 0.2170454 FALSE      FALSE
##
## $X900$X900
##   AverSpeed VarSpeed MaxSpeed
## 1  3.410438 8.251585 8.536075
##
## $X900$X900
##   AverAzimuth VarAzimuth SEAzimuth
## 1    46.05307   0.876566 -22.51073
##
##
## $X906
## $X906$X906
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1  66546.4 14350.52 632.2901  55.67717  6654.64 4436.365 55.67717
##
## $X906$X906
##         Duration   AverDur     SDDur  dupl multseason
## 1 9.933333 hours 0.9933333 0.6637064 FALSE      FALSE
##
## $X906$X906
##   AverSpeed VarSpeed MaxSpeed
## 1  2.183031 2.629678 5.217767
##
## $X906$X906
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -78.76355  0.2393251  -81.5763
##
##
## $X906B
## $X906B$X906B
##   TravDist  MaxDist MinDist FarthDist AverDist   SDDist   SEDist
## 1 258976.2 22709.89 2049.44  170.2408 13630.33 5827.164 158.8889
##
## $X906B$X906B
##         Duration  AverDur      SDDur  dupl multseason
```

```
## 1 9.933333 hours 0.522807 0.03478795 FALSE      FALSE
##
## $X906B$X906B
##   AverSpeed VarSpeed MaxSpeed
## 1  7.263217 9.192939  11.1323
##
## $X906B$X906B
##   AverAzimuth VarAzimuth SEAzimuth
## 1     54.9409  0.5076469  52.91957
##
##
## $X908
## $X908$X908
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1  1087563 20164.09 299.6321  320.8621 8841.972 4992.601 24.41883
##
## $X908$X908
##         Duration   AverDur     SDDur  dupl multseason
## 1 66.88333 hours 0.5437669 0.1799792 FALSE      FALSE
##
## $X908$X908
##   AverSpeed VarSpeed MaxSpeed
## 1  4.641128 7.087369 11.20227
##
## $X908$X908
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -95.36297  0.9000115  33.36271
##
##
## $X909
## $X909$X909
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 124486.1 13768.36 218.6027  74.67315 3772.305 3854.238 74.62662
##
## $X909$X909
##     Duration   AverDur     SDDur  dupl multseason
## 1 18.3 hours 0.5545455 0.1283761 FALSE      FALSE
##
## $X909$X909
##   AverSpeed VarSpeed MaxSpeed
## 1  1.876559 3.431987 7.487946
##
## $X909$X909
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -147.0014  0.5573626 -127.5086
##
##
## $X909B
## $X909B$X909B
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 282213.5 55772.29 121.1339  163.6599 8819.173 12338.25 152.8041
##
## $X909B$X909B
##      Duration  AverDur   SDDur  dupl multseason
```

```
## 1 37.15 hours 1.160937 1.44077 FALSE      FALSE
##
## $X909B$X909B
##   AverSpeed VarSpeed MaxSpeed
## 1  2.460938 3.652433 6.082342
##
## $X909B$X909B
##   AverAzimuth VarAzimuth SEAzimuth
## 1    38.19806  0.6082335  74.21069
##
##
## $X910
## $X910$X910
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 81854.76 13901.05 336.9774  32.39959 4308.145 3484.889 29.68398
##
## $X910$X910
##         Duration  AverDur      SDDur  dupl multseason
## 1 9.933333 hours 0.522807 0.03478795 FALSE      FALSE
##
## $X910$X910
##   AverSpeed VarSpeed MaxSpeed
## 1    2.3137 3.797404 7.722807
##
## $X910$X910
##   AverAzimuth VarAzimuth SEAzimuth
## 1    6.781629  0.8389743 -83.02665
##
##
## $X902
## $X902$X902
##   TravDist  MaxDist  MinDist FarthDist AverDist  SDDist   SEDist
## 1 219204.3 17394.59 113.9852  93.32326 5768.533 5123.17 93.32326
##
## $X902$X902
##     Duration   AverDur     SDDur  dupl multseason
## 1 23.8 hours 0.6263158 0.4436003 FALSE      FALSE
##
## $X902$X902
##   AverSpeed VarSpeed MaxSpeed
## 1  2.976849 6.891068 9.225599
##
## $X902$X902
##   AverAzimuth VarAzimuth SEAzimuth
## 1    62.62853   0.583357  66.92089
```

```
show(movedata)
```

```
## class       : MoveStack
## features    : 310
## extent      : -15.22459, -9.793738, 52.47343, 54.03079  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +ellps=WGS84
## variables   : 6
## names       :      lat,        lon,            DateTime,   bathymetry, individual.local.identifier,
```

```
## min values  : 52.47343, -10.011777, 2016-08-21 05:00:00, -1003.064883,                      X900, u
## max values  : 54.03079,  -9.980319, 2016-08-29 04:18:00,    -1.269246,                      X910, u
## timestamps  : 2016-08-21 05:00:00 ... 2016-08-29 04:18:00 Time difference of 8 days  (start ... end,
## sensors     : unknown
## indiv. data : ID
## min ID Data : 900
## max ID Data : 910
## individuals : X900, X906, X906B, X908, X909, X909B, X910, X902
## date created: 2016-02-04 02:26:00
```

```
# number of relocation for each bird
n.locs(movedata)
```

```
##  X900  X906 X906B  X908  X909 X909B  X910  X902
##    29    11    20   124    34    33    20    39
```

```
# summary of the speed statistics in metres per second
speedSummary(movedata)
```

```
## $X900
##   AverSpeed VarSpeed MaxSpeed
## 1  3.410438 8.251585 8.536075
##
## $X906
##   AverSpeed VarSpeed MaxSpeed
## 1  2.183031 2.629678 5.217767
##
## $X906B
##   AverSpeed VarSpeed MaxSpeed
## 1  7.263217 9.192939  11.1323
##
## $X908
##   AverSpeed VarSpeed MaxSpeed
## 1  4.641128 7.087369 11.20227
##
## $X909
##   AverSpeed VarSpeed MaxSpeed
## 1  1.876559 3.431987 7.487946
##
## $X909B
##   AverSpeed VarSpeed MaxSpeed
## 1  2.460938 3.652433 6.082342
##
## $X910
##   AverSpeed VarSpeed MaxSpeed
## 1    2.3137 3.797404 7.722807
##
## $X902
##   AverSpeed VarSpeed MaxSpeed
## 1  2.976849 6.891068 9.225599
```

```r
# summary of the time statistics in hours
timeSummary(movedata, units="hours")
```

```
## $X900
##         Duration   AverDur     SDDur  dupl multseason
## 1 16.08333 hours 0.5744048 0.2170454 FALSE      FALSE
##
## $X906
##         Duration   AverDur     SDDur  dupl multseason
## 1 9.933333 hours 0.9933333 0.6637064 FALSE      FALSE
##
## $X906B
##         Duration  AverDur      SDDur  dupl multseason
## 1 9.933333 hours 0.522807 0.03478795 FALSE      FALSE
##
## $X908
##         Duration   AverDur     SDDur  dupl multseason
## 1 66.88333 hours 0.5437669 0.1799792 FALSE      FALSE
##
## $X909
##     Duration   AverDur     SDDur  dupl multseason
## 1 18.3 hours 0.5545455 0.1283761 FALSE      FALSE
##
## $X909B
##      Duration  AverDur   SDDur  dupl multseason
## 1 37.15 hours 1.160937 1.44077 FALSE      FALSE
##
## $X910
##         Duration  AverDur      SDDur  dupl multseason
## 1 9.933333 hours 0.522807 0.03478795 FALSE      FALSE
##
## $X902
##     Duration   AverDur     SDDur  dupl multseason
## 1 23.8 hours 0.6263158 0.4436003 FALSE      FALSE
```

```r
# summary of distance measures in metres
distanceSummary(movedata)
```

```
## $X900
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 180065.4 15364.94 26.63112  84.11312 6430.906 5263.414 14.29843
##
## $X906
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1  66546.4 14350.52 632.2901  55.67717  6654.64 4436.365 55.67717
##
## $X906B
##   TravDist  MaxDist MinDist FarthDist AverDist   SDDist   SEDist
## 1 258976.2 22709.89 2049.44  170.2408 13630.33 5827.164 158.8889
##
## $X908
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1  1087563 20164.09 299.6321  320.8621 8841.972 4992.601 24.41883
```

```
##
## $X909
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 124486.1 13768.36 218.6027  74.67315 3772.305 3854.238 74.62662
##
## $X909B
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 282213.5 55772.29 121.1339  163.6599 8819.173 12338.25 152.8041
##
## $X910
##   TravDist  MaxDist  MinDist FarthDist AverDist   SDDist   SEDist
## 1 81854.76 13901.05 336.9774  32.39959 4308.145 3484.889 29.68398
##
## $X902
##   TravDist  MaxDist  MinDist FarthDist AverDist  SDDist   SEDist
## 1 219204.3 17394.59 113.9852  93.32326 5768.533 5123.17 93.32326
```

```
# summary of angle measures in degrees
angleSummary(movedata)
```

```
## $X900
##   AverAzimuth VarAzimuth SEAzimuth
## 1    46.05307   0.876566 -22.51073
##
## $X906
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -78.76355  0.2393251  -81.5763
##
## $X906B
##   AverAzimuth VarAzimuth SEAzimuth
## 1     54.9409  0.5076469  52.91957
##
## $X908
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -95.36297  0.9000115  33.36271
##
## $X909
##   AverAzimuth VarAzimuth SEAzimuth
## 1   -147.0014  0.5573626 -127.5086
##
## $X909B
##   AverAzimuth VarAzimuth SEAzimuth
## 1    38.19806  0.6082335  74.21069
##
## $X910
##   AverAzimuth VarAzimuth SEAzimuth
## 1    6.781629  0.8389743 -83.02665
##
## $X902
##   AverAzimuth VarAzimuth SEAzimuth
## 1    62.62853   0.583357  66.92089
```

```r
# The time.lag function calculates the time lags between locations
timeLag(movedata, units="mins")
```

```
## $X900
##  [1] 30 30 30 35 30 30 30 34 30 30 30 34 31 30 29 30 30 30 35 60 34 94 30
## [24] 30 35 30 30 34
##
## $X906
##  [1]  30  95  60  34  30  30  34 125 124  34
##
## $X906B
##  [1] 30 30 30 34 30 30 34 30 30 35 30 30 34 30 30 34 30 30 35
##
## $X908
##   [1]  30  30  30  30  60  30  30  34  30  31  30  30  30  30  30  30  30
##  [18]  30  30  30  30  30  30  30  30  30  30  30  64  30  30  30  30  30
##  [35]  30  34  30  30  34  31  30  30  30  30  30  30  30  60  30  64  30
##  [52]  30  30  30  30  30  30  30  30  30  30  30  30  30  34  30  30  60
##  [69]  30  35  30  30  34  30  30  34  30  30 129  30  30  30  30  34  30
##  [86]  30  35  30  30  30  30  30  30  34  30  30  34  30  31  30  34  30
## [103]  30  30  34  30  30  30  34  31  30  30  30  30  30  30  30  30  30
## [120]  30  30  30  34
##
## $X909
##  [1] 30 60 35 30 30 34 30 30 64 30 35 30 30 34 30 30 30 34 30 30 35 30 30
## [24] 34 30 30 34 30 30 35 30 30 34
##
## $X909B
##  [1]  30  30  30 184  30  30  30  34  30  31  34  30  30  34  30  30  35
## [18] 156 122  92 217 313  30  30  30  30  34  30  30  34  30 369
##
## $X910
##  [1] 30 30 30 35 30 30 34 30 30 34 30 30 35 30 30 34 30 30 34
##
## $X902
##  [1]  30  30  30  30  30  30  35  30  30  34  30  30  34  60  35  30  30
## [18]  34  30  30  60  30  34  30  30  31  29  30  30  35  30  30  64  30
## [35]  34  30  30 189
```

```r
#calculate time difference in minutes
df <- data[order(data$ID, data$DateTime),]
df$tdiff <- unlist(tapply(data$DateTime, INDEX = data$ID,
                          FUN = function(x) c(0, `units<-`(diff(x), "mins"))))
#df
which(df$tdiff > 1000)
```

```
## integer(0)
```

```r
# Apply Hidden Markov Model to the Data
```

```r
# Interpolate the tracks so that they are measured on the same interval

# Currently this does not work for all tracks being interpolated

idx = c("900","902","908","910","906","906B","909","909B")
dataSample2<-data[data$ID %in% idx,]
head(dataSample2)
```

```
##        lat       lon            DateTime  ID bathymetry
## 1 53.60396 -10.62676 2016-08-23 05:00:00 900  -117.2506
## 2 53.63333 -10.79087 2016-08-23 05:30:00 900  -122.5151
## 3 53.64509 -10.98963 2016-08-23 06:00:00 900  -144.6082
## 4 53.69830 -11.20415 2016-08-23 06:30:00 900  -174.3880
## 5 53.73100 -11.34850 2016-08-23 07:05:00 900  -223.7407
## 6 53.76850 -11.38927 2016-08-23 07:35:00 900  -239.6527
```

```r
tail(dataSample2)
```

```
##          lat       lon            DateTime  ID bathymetry
## 373 53.26624 -10.45220 2016-08-27 23:35:00 902  -94.29939
## 374 53.31818 -10.32203 2016-08-28 00:05:00 902 -108.79306
## 375 53.39190 -10.28610 2016-08-28 00:39:00 902  -87.10595
## 376 53.45968 -10.25767 2016-08-28 01:09:00 902  -35.74634
## 377 53.54189 -10.26372 2016-08-28 01:39:00 902  -24.18403
## 383 53.54685 -10.25435 2016-08-28 04:48:00 902  -15.62752
```

```r
# create a trajectory object using adehabitatLT
tr<-as.ltraj(data.frame(X=dataSample2$lon,Y=dataSample2$lat),date=dataSample2$DateTime,id=dataSample2$I
tstep<-1800 #time step we want for the interpolation, in seconds
newtr<-redisltraj(tr, u=tstep, type = "time")
head(newtr)
```

```
##
## ********** List of class ltraj **********
##
## Type of the traject: Type II (time recorded)
## Regular traject. Time lag between two locs: 1800 seconds
##
## Characteristics of the bursts:
##     id burst nb.reloc NAs          date.begin            date.end
## 1  900   900       33   0 2016-08-23 05:00:00 2016-08-23 21:00:00
## 2  902   902       48   0 2016-08-27 05:00:00 2016-08-28 04:30:00
## 3  906   906       20   0 2016-08-23 05:00:00 2016-08-23 14:30:00
## 4 906B  906B       20   0 2016-08-26 13:43:00 2016-08-26 23:13:00
## 5  908   908      134   0 2016-08-21 05:00:00 2016-08-23 23:30:00
## 6  909   909       37   0 2016-08-25 05:00:00 2016-08-25 23:00:00
##
##
##  infolocs provided. The following variables are available:
## [1] "pkey"
```

```
head(newtr[[1]])
```

```
##           x        y                date          dx         dy       dist
## 1 -10.62676 53.60396 2016-08-23 05:00:00 -0.16410700 0.02937700 0.16671567
## 2 -10.79087 53.63333 2016-08-23 05:30:00 -0.19875600 0.01175700 0.19910343
## 3 -10.98963 53.64509 2016-08-23 06:00:00 -0.21452200 0.05320700 0.22102188
## 4 -11.20415 53.69830 2016-08-23 06:30:00 -0.12373200 0.02803114 0.12686746
## 5 -11.32788 53.72633 2016-08-23 07:00:00 -0.05459450 0.03591936 0.06535105
## 6 -11.38247 53.76225 2016-08-23 07:30:00  0.01108717 0.03312283 0.03492918
##     dt        R2n abs.angle   rel.angle
## 1 1800 0.00000000  2.964458          NA
## 2 1800 0.02779412  3.082509  0.11805098
## 3 1800 0.13336156  2.898473 -0.18403569
## 4 1800 0.34227366  2.918806  0.02033314
## 5 1800 0.50653999  2.559663 -0.35914306
## 6 1800 0.59615607  1.247790 -1.31187259
```

```
# convert object of class ltraj to a dataframe
df<-ld(newtr)
names(df)[names(df) == 'x'] <- 'lon'
names(df)[names(df) == 'y'] <- 'lat'
head(df)
```

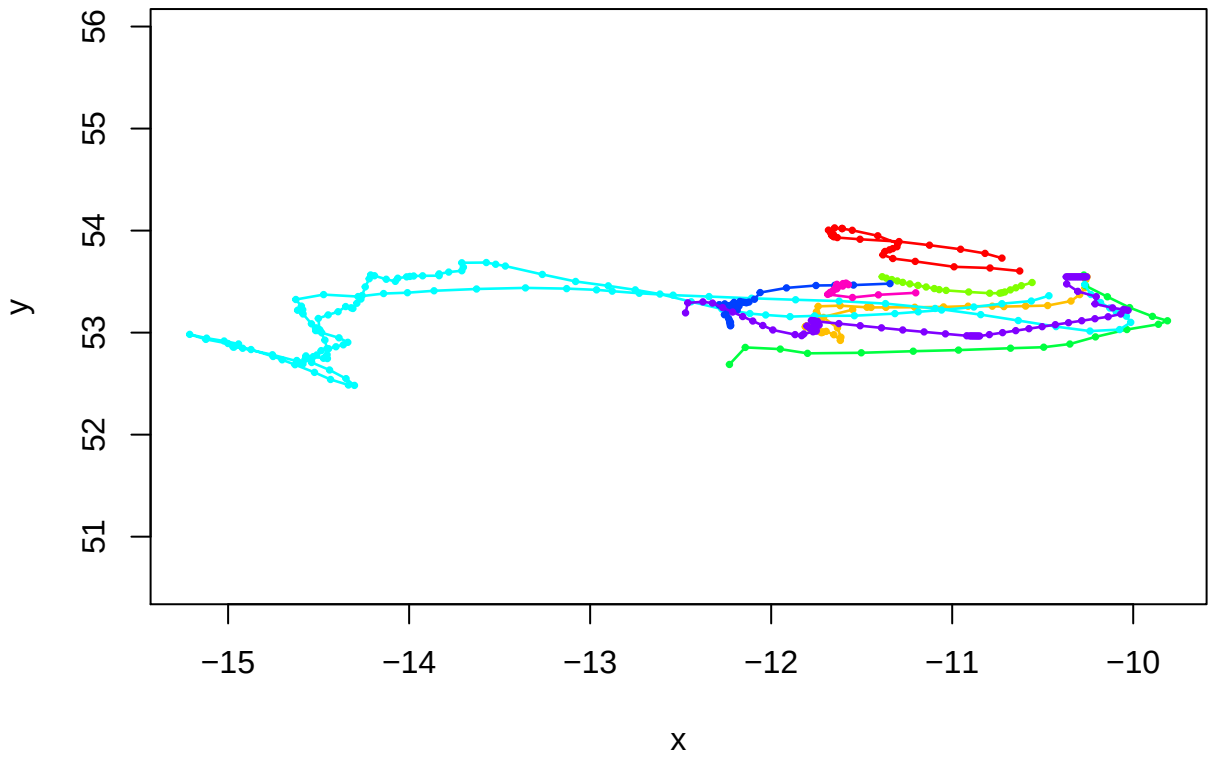
```
##         lon      lat                date          dx         dy       dist
## 1 -10.62676 53.60396 2016-08-23 05:00:00 -0.16410700 0.02937700 0.16671567
## 2 -10.79087 53.63333 2016-08-23 05:30:00 -0.19875600 0.01175700 0.19910343
## 3 -10.98963 53.64509 2016-08-23 06:00:00 -0.21452200 0.05320700 0.22102188
## 4 -11.20415 53.69830 2016-08-23 06:30:00 -0.12373200 0.02803114 0.12686746
## 5 -11.32788 53.72633 2016-08-23 07:00:00 -0.05459450 0.03591936 0.06535105
## 6 -11.38247 53.76225 2016-08-23 07:30:00  0.01108717 0.03312283 0.03492918
##     dt        R2n abs.angle   rel.angle  id burst                   pkey
## 1 1800 0.00000000  2.964458          NA 900   900 900.2016-08-23 05:00:00
## 2 1800 0.02779412  3.082509  0.11805098 900   900 900.2016-08-23 05:30:00
## 3 1800 0.13336156  2.898473 -0.18403569 900   900 900.2016-08-23 06:00:00
## 4 1800 0.34227366  2.918806  0.02033314 900   900 900.2016-08-23 06:30:00
## 5 1800 0.50653999  2.559663 -0.35914306 900   900 900.2016-08-23 07:00:00
## 6 1800 0.59615607  1.247790 -1.31187259 900   900 900.2016-08-23 07:30:00
```

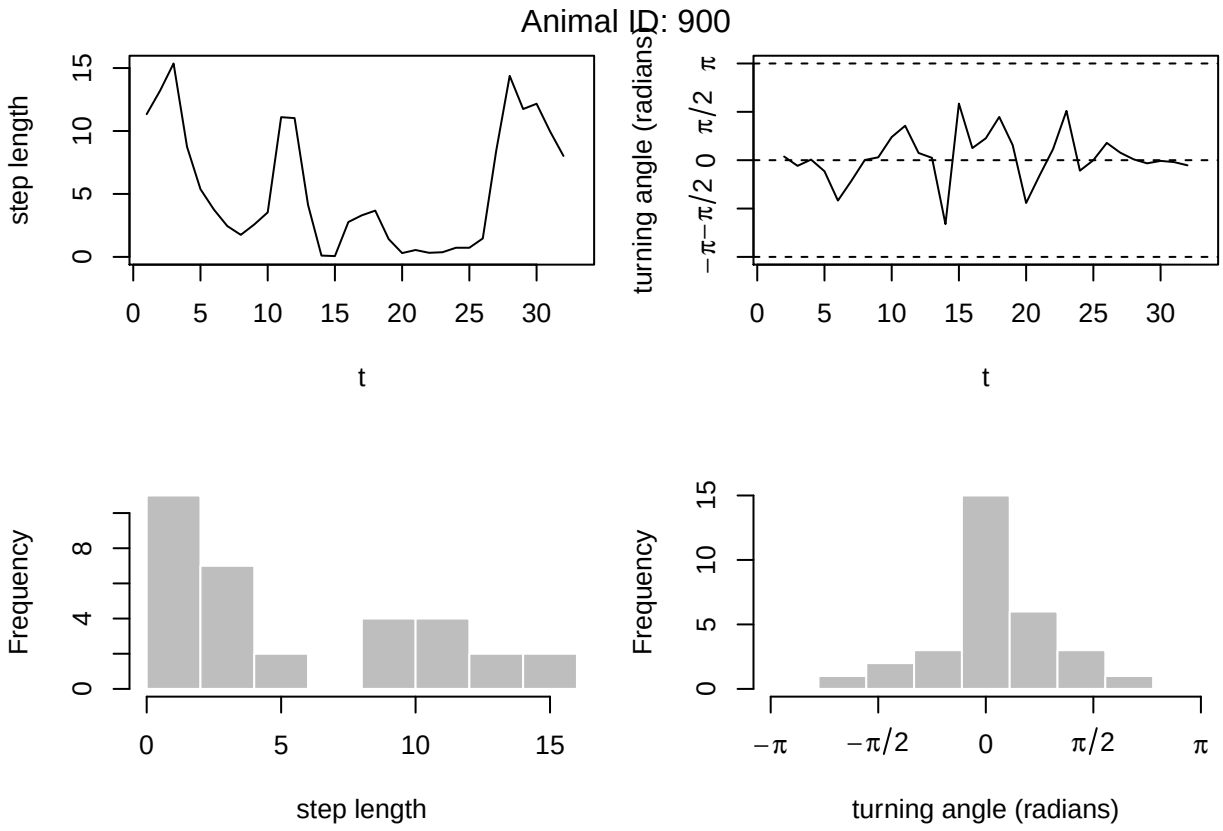
```
tail(df)
```

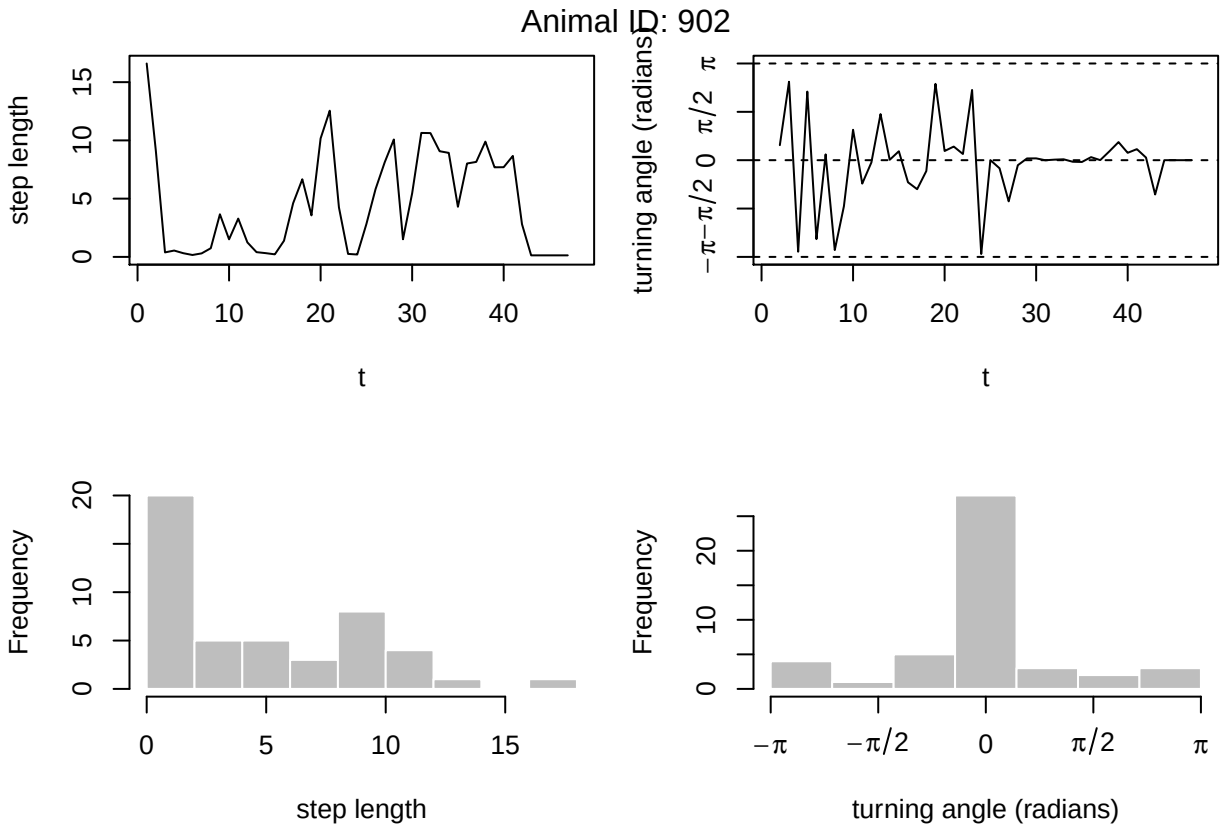
```
##           lon      lat                date           dx            dy
## 157 -11.65303 53.42141 2016-08-23 12:00:00  0.015643800  0.0075770000
## 167 -11.63739 53.42899 2016-08-23 12:30:00  0.032999400  0.0274746353
## 177 -11.60439 53.45646 2016-08-23 13:00:00  0.019526400  0.0274678980
## 187 -11.58486 53.48393 2016-08-23 13:30:00 -0.004645867 -0.0009967333
## 197 -11.58951 53.48294 2016-08-23 14:00:00 -0.040974710 -0.0331822235
## 207 -11.63048 53.44975 2016-08-23 14:30:00           NA            NA
##            dist   dt       R2n  abs.angle rel.angle  id burst
## 157 0.017382158 1800 0.2050408  0.4510455 2.4370980 910   910
## 167 0.042939678 1800 0.1916824  0.6942934 0.2432479 910   910
## 177 0.033701123 1800 0.1668769  0.9528051 0.2585117 910   910
## 187 0.004751584 1800 0.1559119 -2.9302543 2.4001259 910   910
```

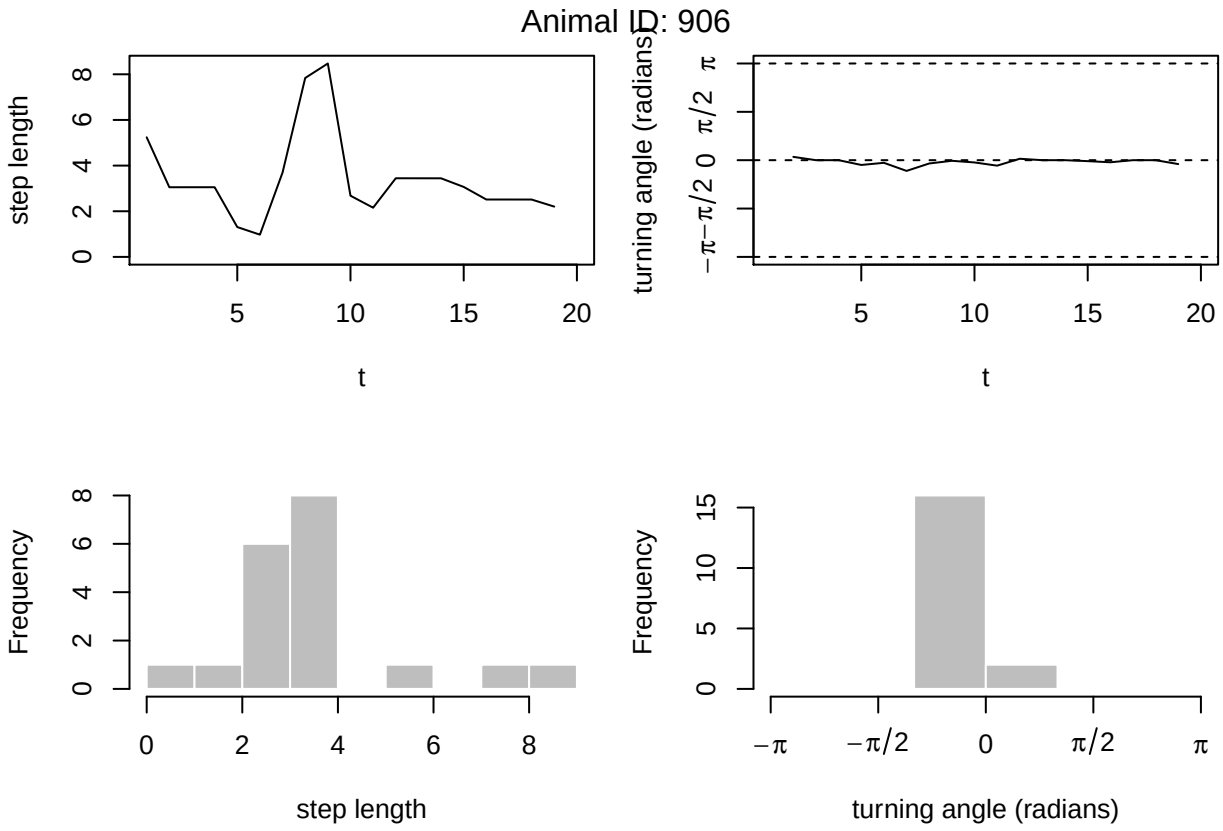
```
## 197 0.052725580 1800 0.1593116 -2.4608913 0.4693631 910   910
## 207          NA   NA 0.1877500         NA        NA 910   910
##                        pkey
## 157 910.2016-08-23 12:00:00
## 167 910.2016-08-23 12:30:00
## 177 910.2016-08-23 13:00:00
## 187 910.2016-08-23 13:30:00
## 197 910.2016-08-23 14:00:00
## 207 910.2016-08-23 14:30:00
```

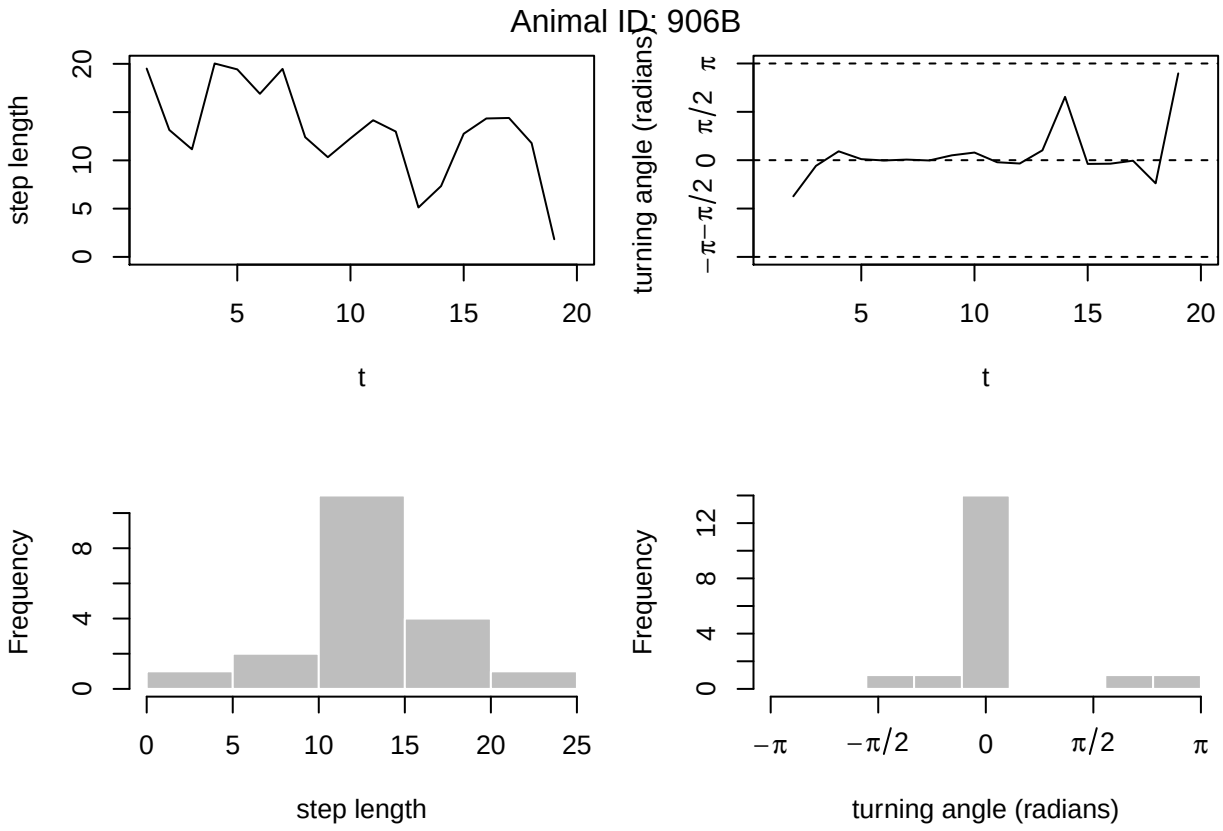
```r
# prepare data with moveHMM
trackData2 <- df[,c(1,2,11)]
colnames(trackData2)[3] <- c("ID")
data3 <- prepData(trackData2,type="LL",coordNames=c("lon","lat"))
plot(data3,compact=T)
```

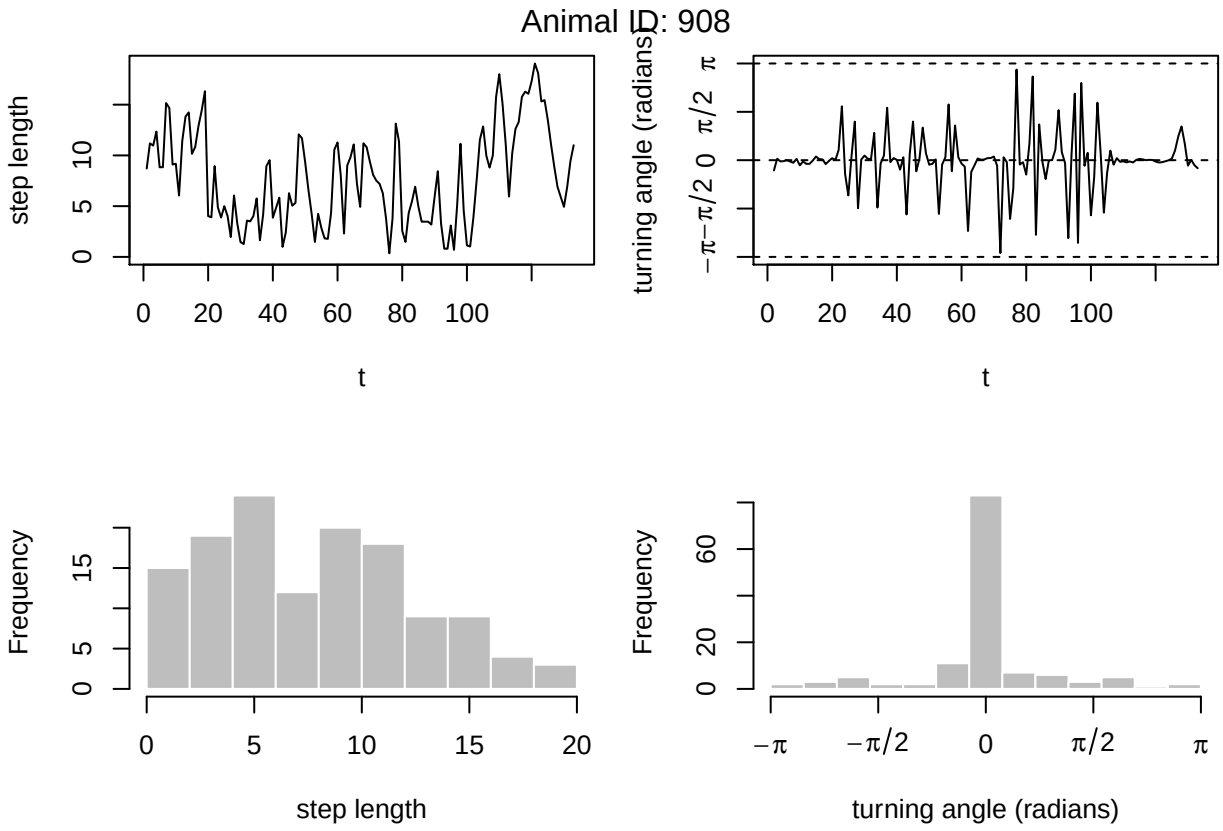


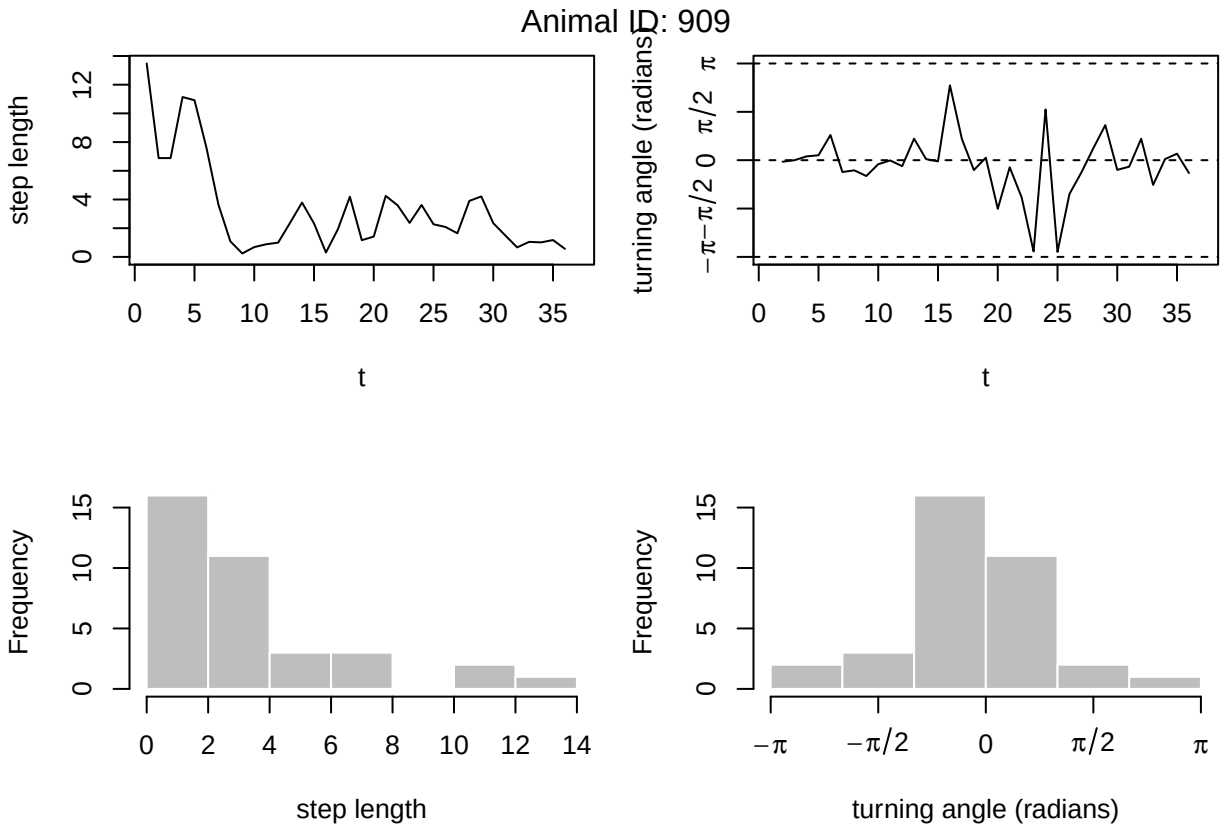

Animal ID: 900
step length
0
5
10
15
0
5
10
15
20
25
30
t
turning angle (radians)
$-\pi$
$-\pi/2$
0
$\pi/2$
π
0
5
10
15
20
25
30
t
Frequency
0
4
8
0
5
10
15
step length
Frequency
0
5
10
15
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)

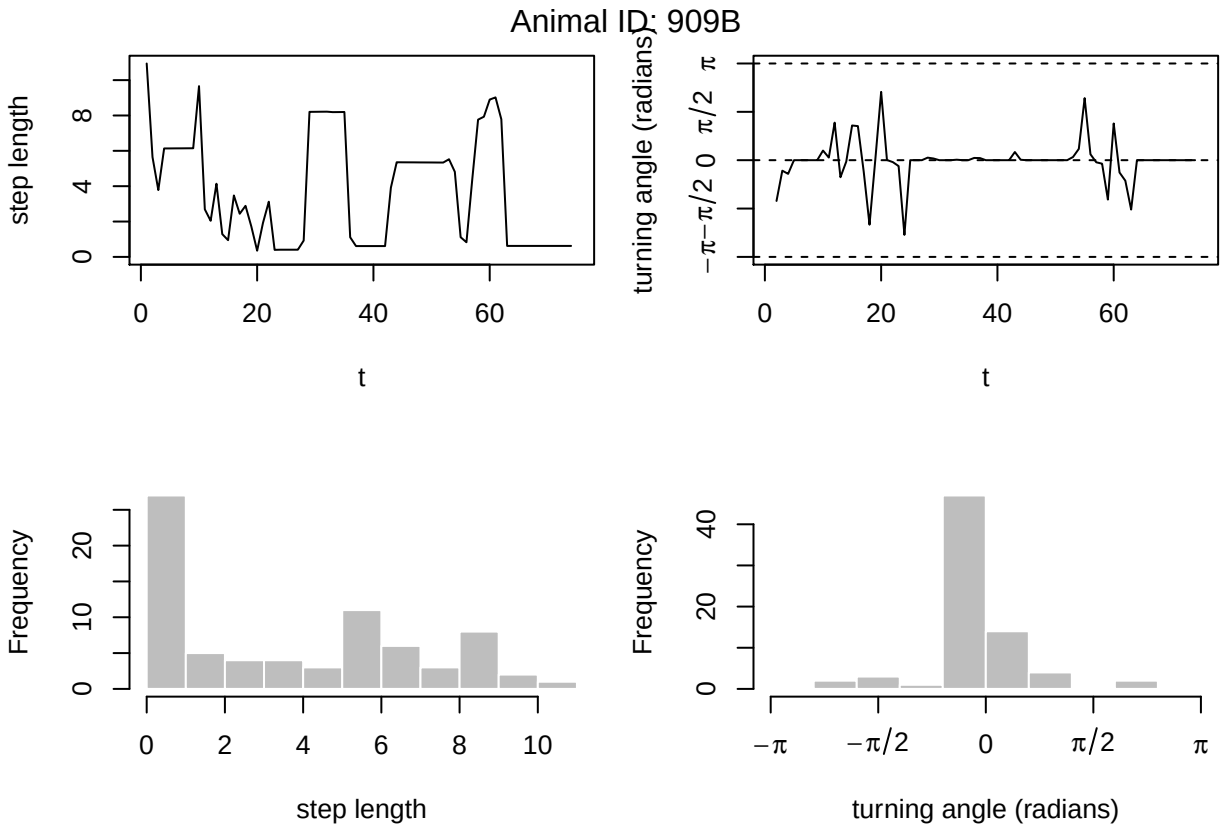

Animal ID: 902
step length
0
5
10
15
0
10
20
30
40
t
turning angle (radians)
$-\pi$
$-\pi/2$
0
$\pi/2$
π
0
10
20
30
40
t
Frequency
0
5
10
20
0
5
10
15
step length
Frequency
0
10
20
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)

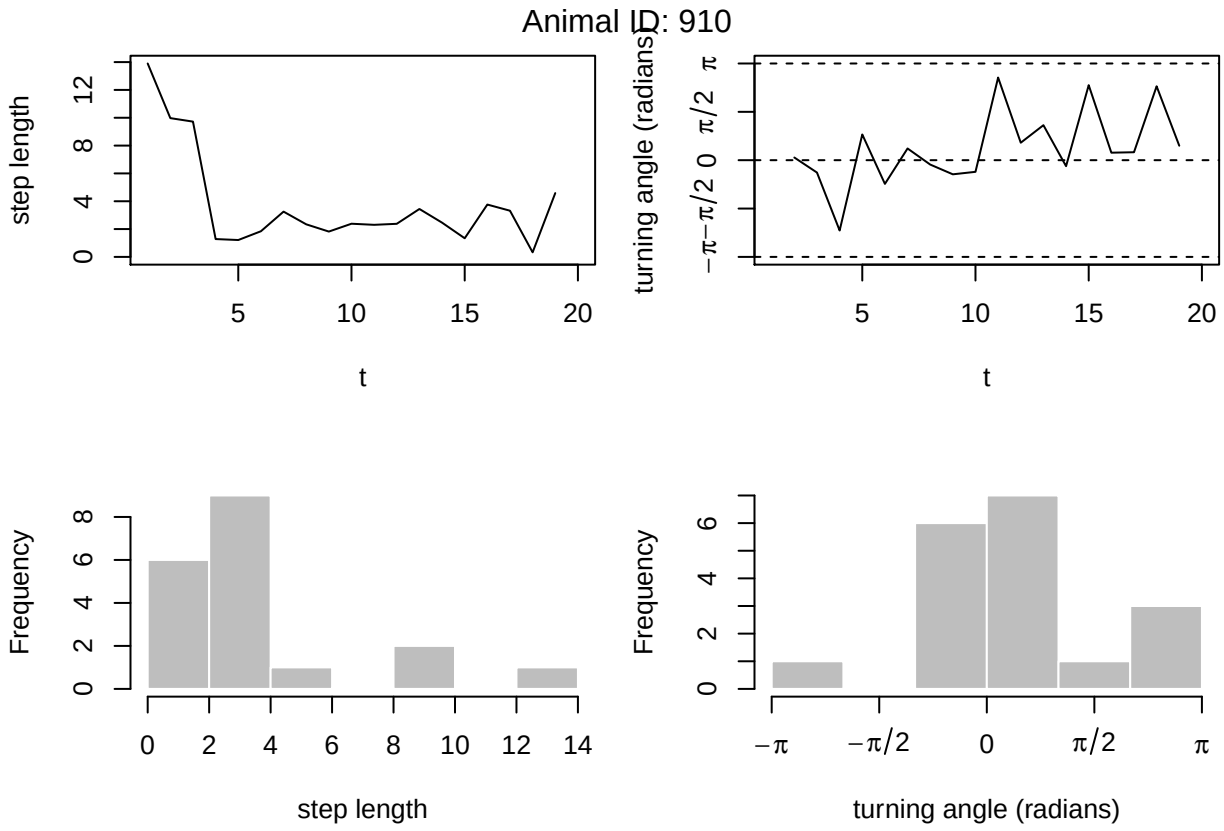

Animal ID: 906
step length
0 2 4 6 8
5 10 15 20
t
turning angle (radians)
$-\pi$ $-\pi/2$ 0 $\pi/2$ π
5 10 15 20
t
Frequency
0 2 4 6 8
0 2 4 6 8
step length
Frequency
0 5 10 15
$-\pi$ $-\pi/2$ 0 $\pi/2$ π
turning angle (radians)


Animal ID: 906B
step length
0
5
10
20
5
10
15
20
t
turning angle (radians)
$-\pi$
$-\pi/2$
0
$\pi/2$
π
5
10
15
20
t
Frequency
0
4
8
0
5
10
15
20
25
step length
Frequency
0
4
8
12
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)


Animal ID: 908
step length
0
5
10
0
20
40
60
80
100
t
turning angle (radians)
$-\pi$
$-\pi/2$
0
$\pi/2$
π
0
20
40
60
80
100
t
Frequency
0
5
15
0
5
10
15
20
step length
Frequency
0
20
60
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)


Animal ID: 909
step length
0
4
8
12
0
5
10
15
20
25
30
35
t
turning angle (radians)
$-\pi$
$-\pi/2$
0
$\pi/2$
π
0
5
10
15
20
25
30
35
t
Frequency
0
5
10
15
0
2
4
6
8
10
12
14
step length
Frequency
0
5
10
15
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)


Animal ID: 909B
step length
0
4
8
0
20
40
60
t
turning angle (radians)
$-\pi$ $-\pi/2$ 0 $\pi/2$ π
0
20
40
60
t
Frequency
0
10
20
0
2
4
6
8
10
step length
Frequency
0
20
40
$-\pi$
$-\pi/2$
0
$\pi/2$
π
turning angle (radians)



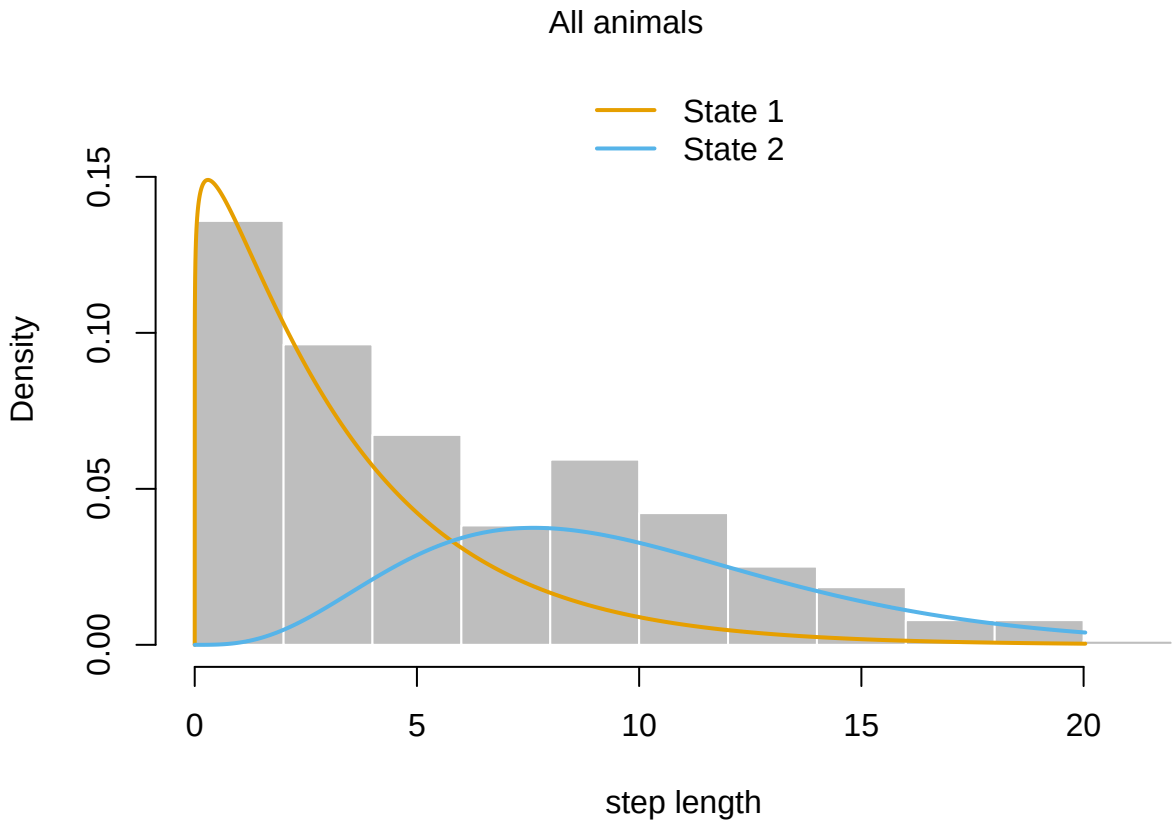

```
#apply two state HMM
# initial parameters for gamma and von Mises distributions
mu0 <- c(0.1,1) # step mean (two parameters: one for each state)
sigma0 <- c(0.1,1) # step SD
stepPar0 <- c(mu0,sigma0)
angleMean0 <- c(pi,0) # angle mean
kappa0 <- c(1,1) # angle concentration
anglePar0 <- c(angleMean0,kappa0)

m1 <- fitHMM(data=data3,nbStates=2,stepPar0=stepPar0,anglePar0=anglePar0,
             formula=~1) # no covariate

m1
```

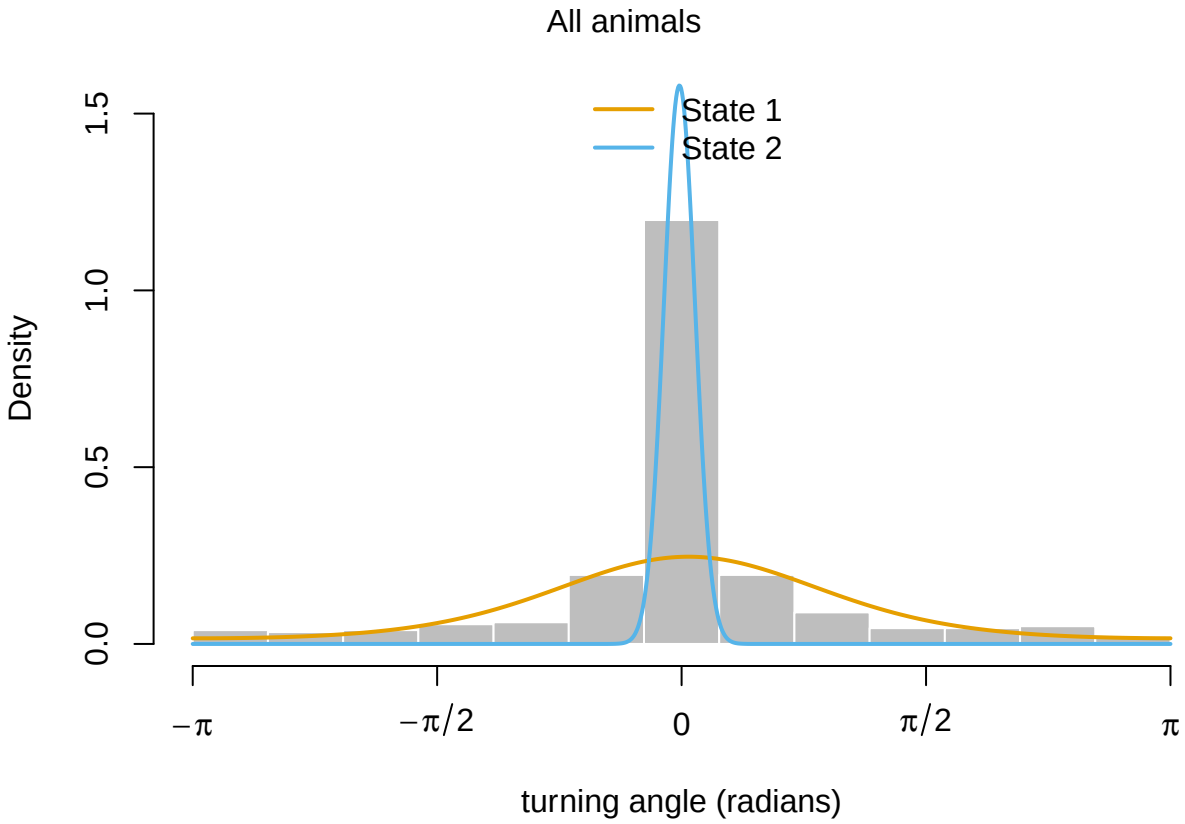
```
## Value of the maximum log-likelihood: -1284.268
##
## Step length parameters:
## ----------------------
##       state 1  state 2
## mean 3.367104 9.867656
## sd   3.214621 4.697946
##
## Turning angle parameters:
## ------------------------
##                  state 1     state 2
## mean          0.0442828 -0.01374861
```

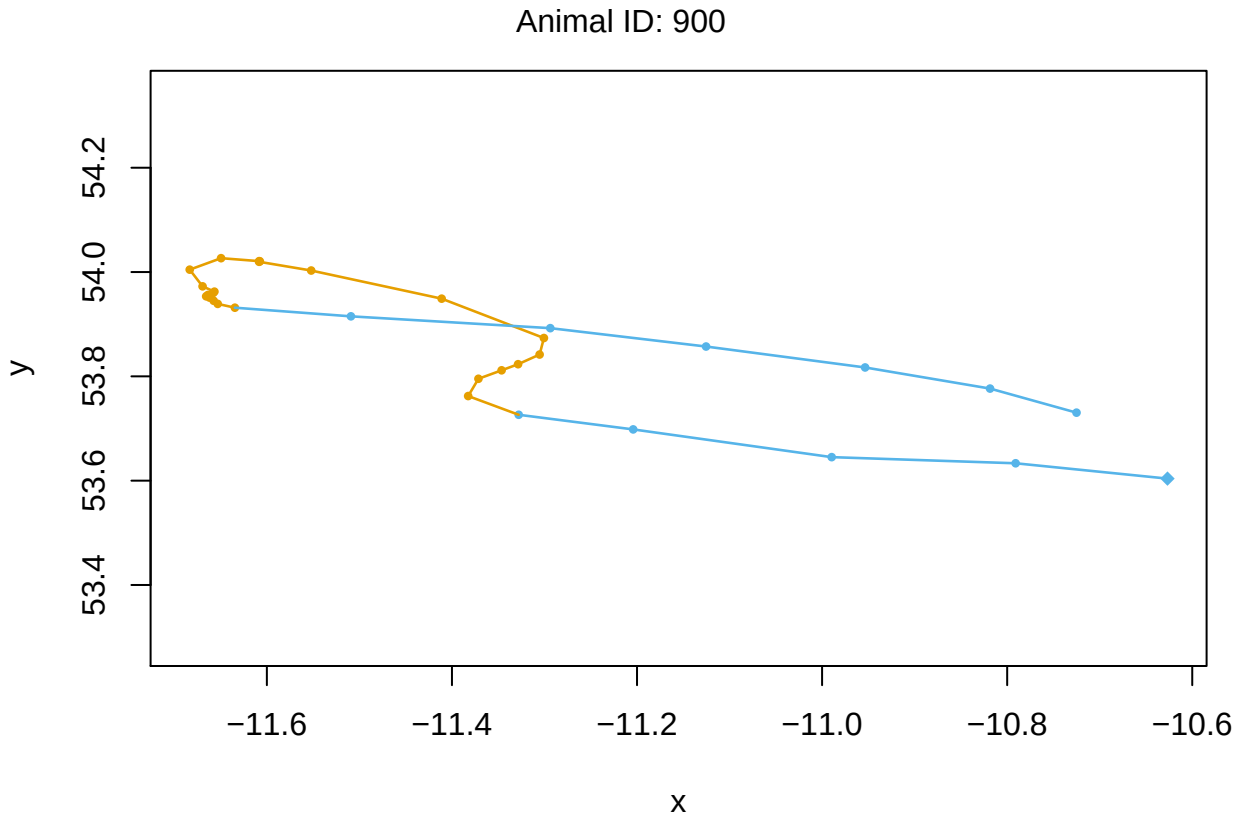
```
## concentration 1.3681332 99.27710434
##
## Regression coeffs for the transition probabilities:
## ---------------------------------------------------
##              1 -> 2    2 -> 1
## intercept -2.402516 -1.634608
##
## Transition probability matrix:
## ------------------------------
##           [,1]       [,2]
## [1,] 0.9170190 0.08298105
## [2,] 0.1632001 0.83679993
##
## Initial distribution:
## ---------------------
## [1] 2.664580e-07 9.999997e-01
```

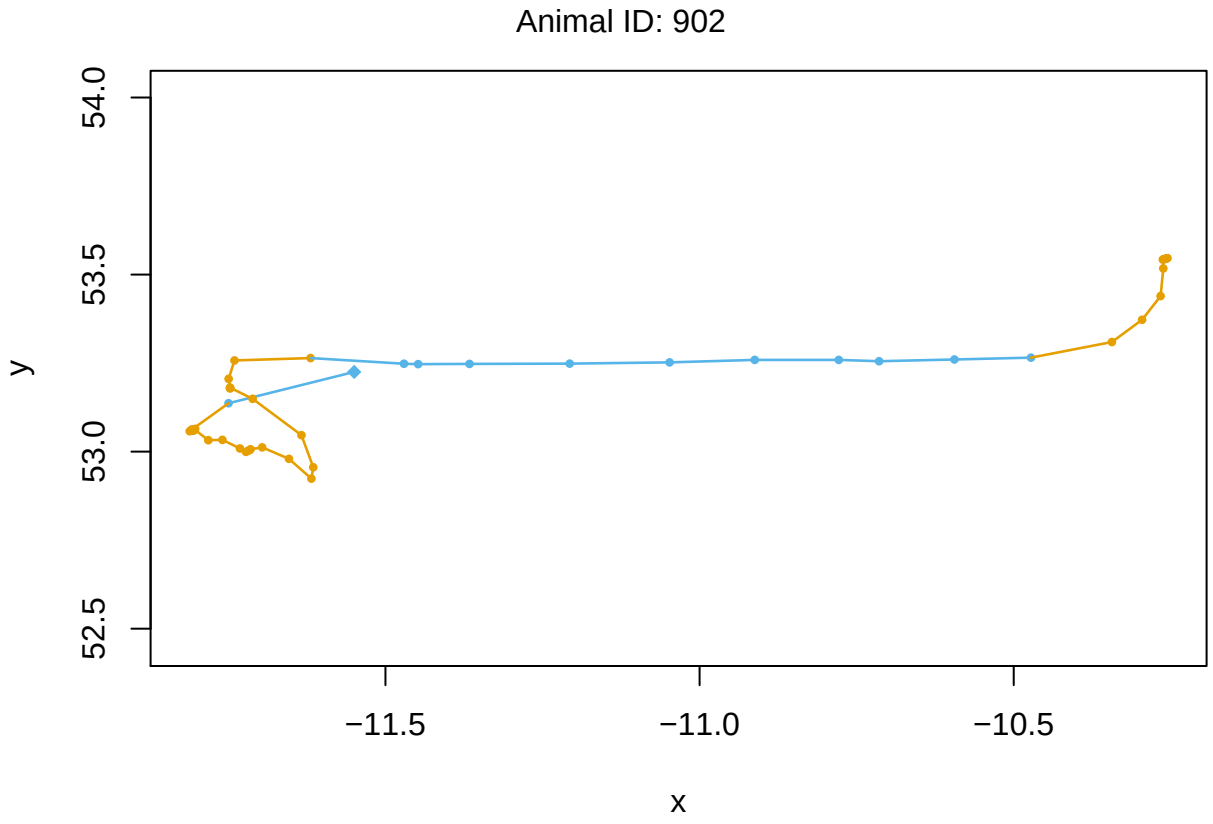
```
plot(m1)
```

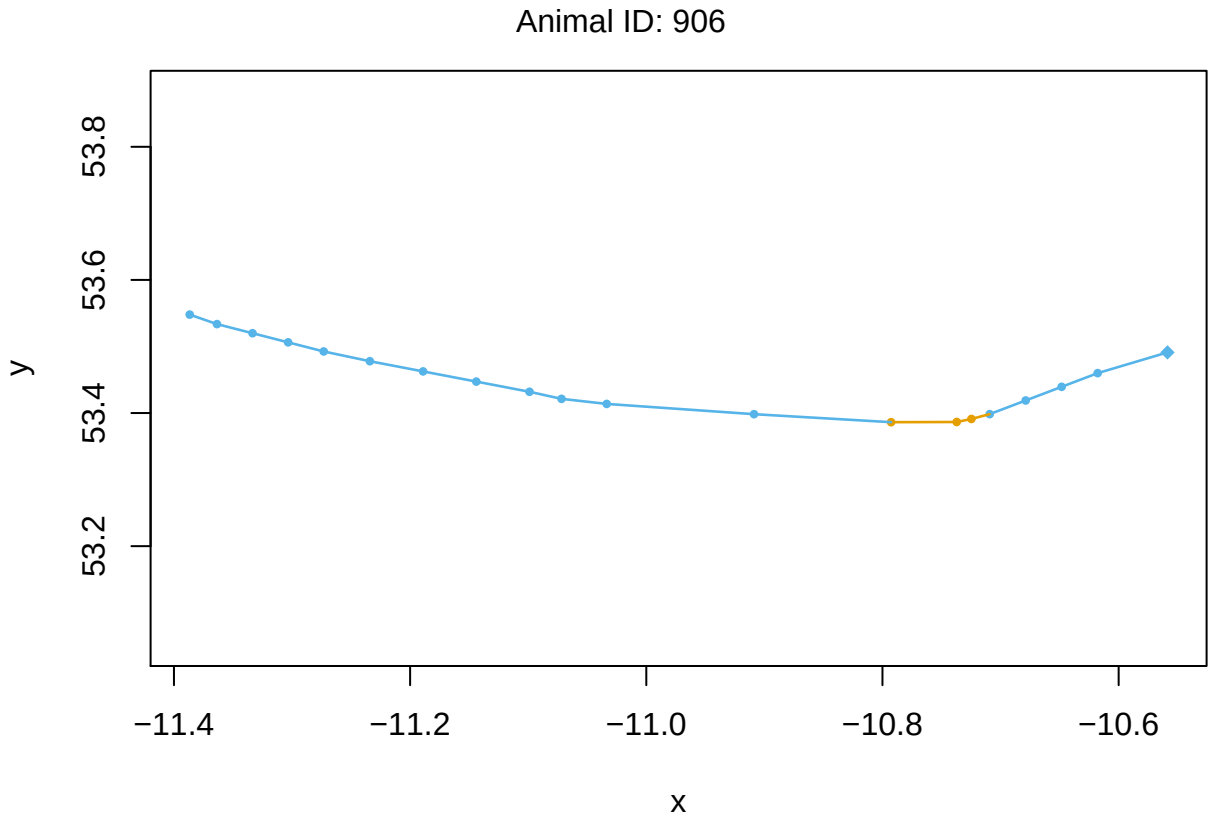
```
## Decoding states sequence... DONE
```

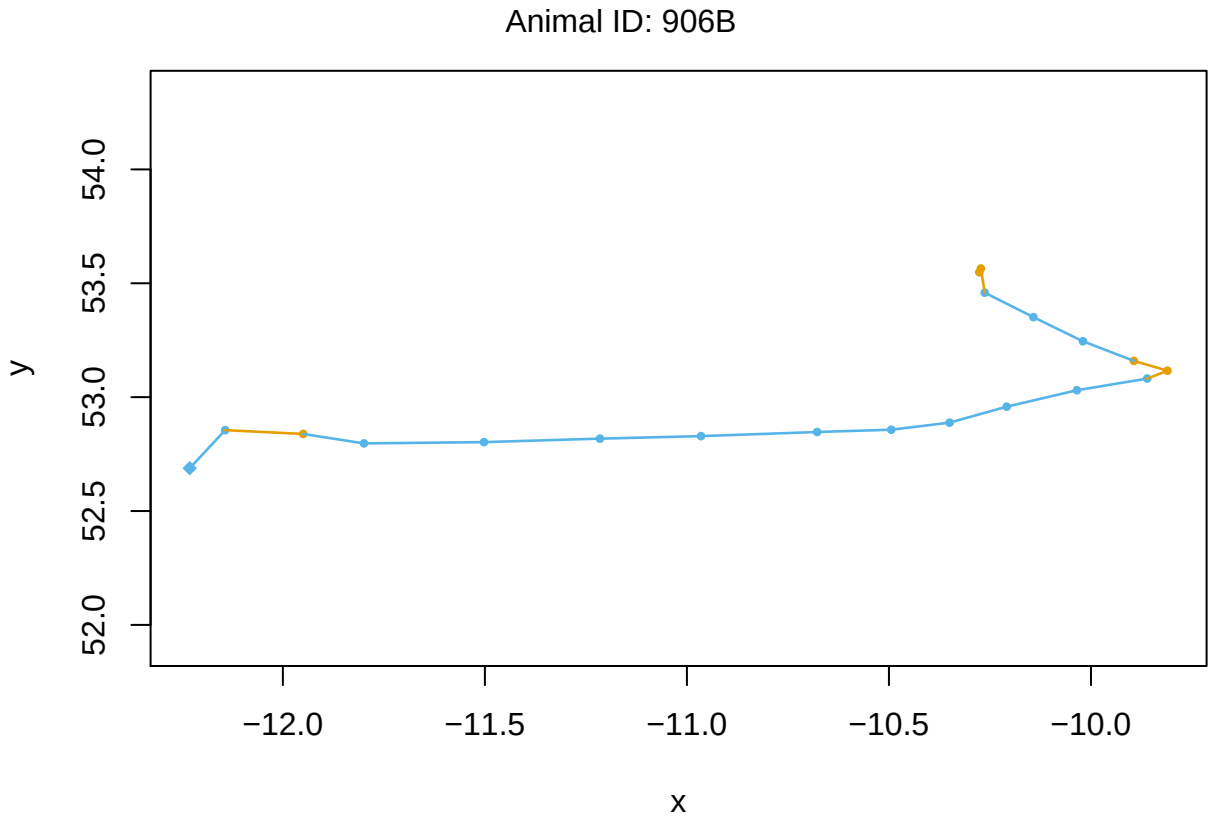


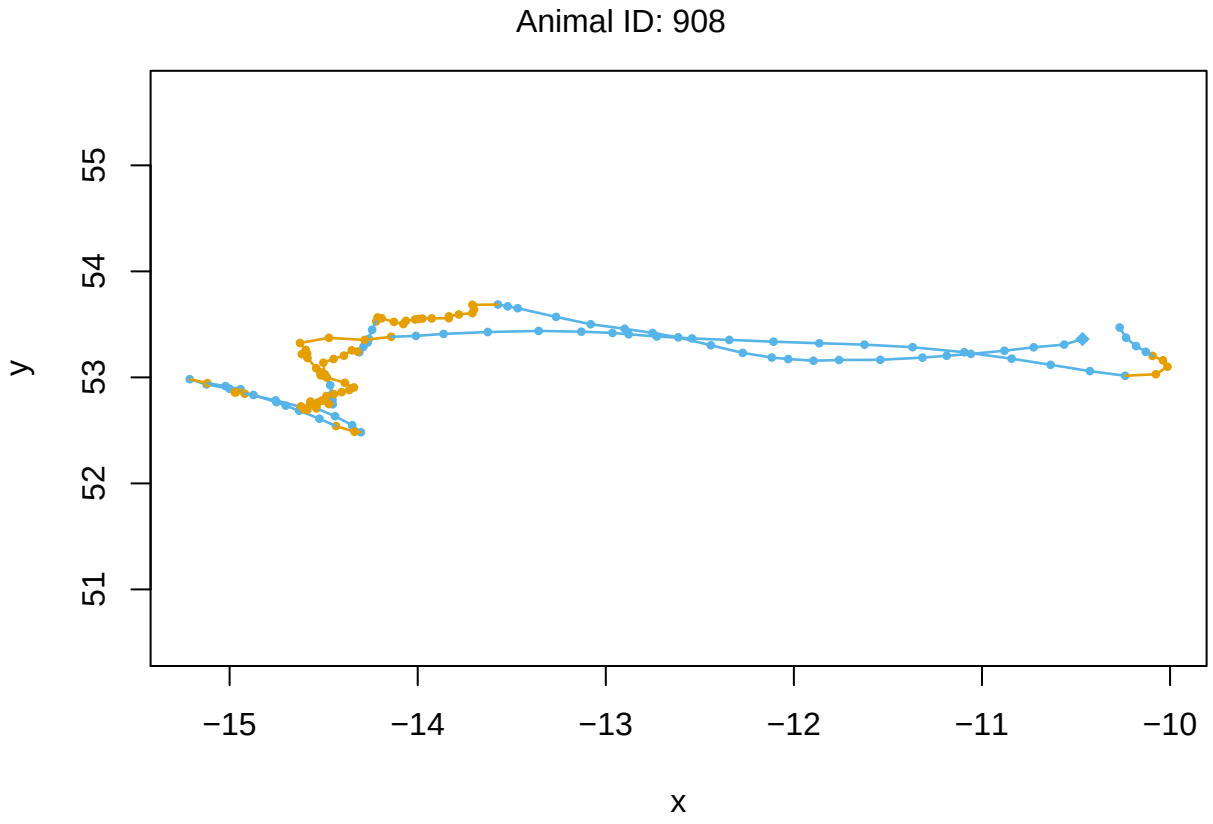

All animals
State 1
State 2
Density
0.0
0.5
1.0
1.5
−π
−π/2
0
π/2
π
turning angle (radians)

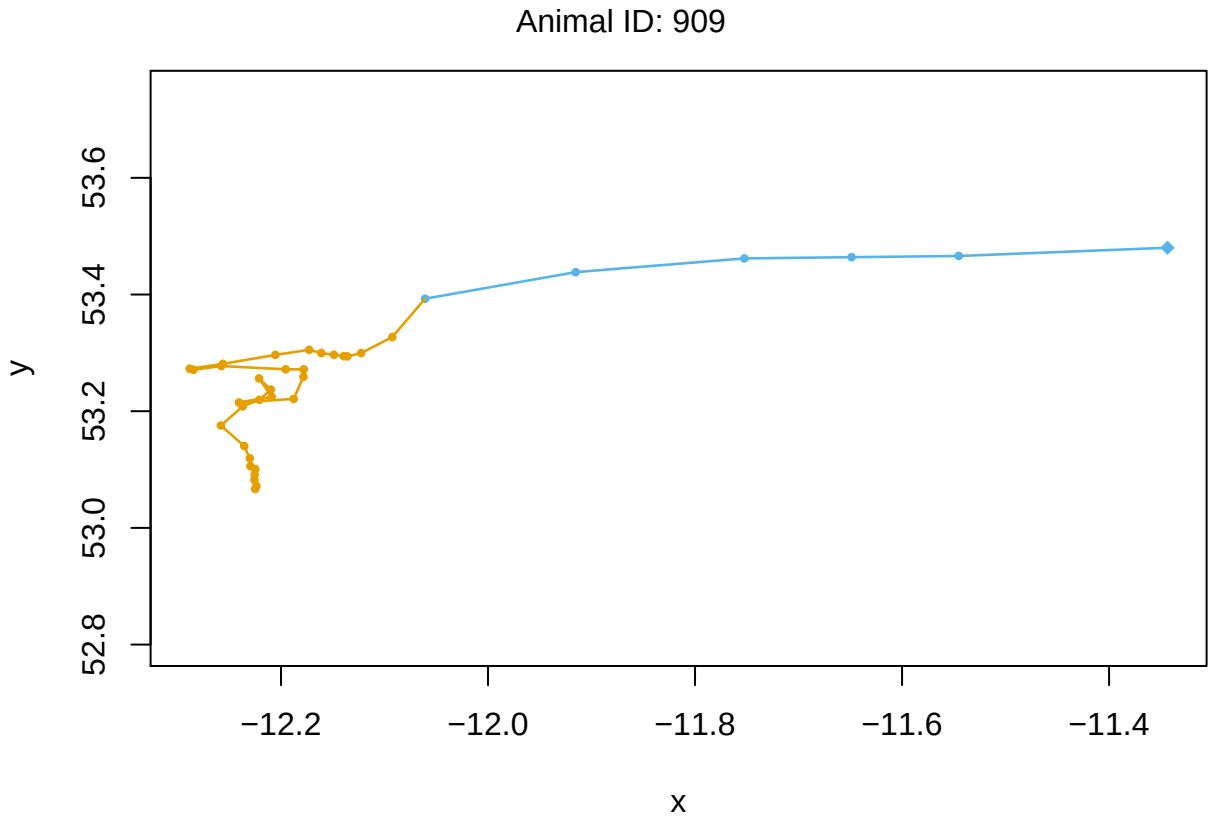

Animal ID: 900
54.2
54.0
53.8
53.6
53.4
y
−11.6
−11.4
−11.2
−11.0
−10.8
−10.6
x

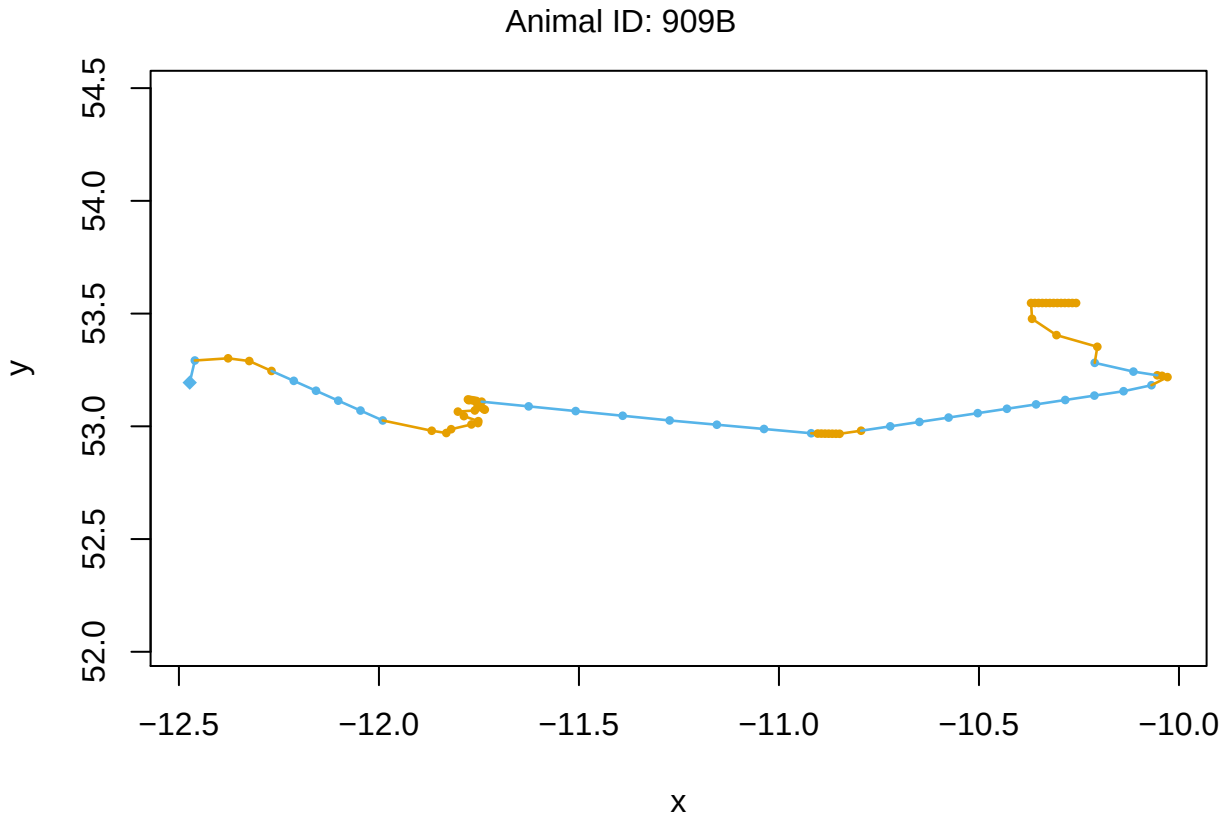

Animal ID: 902
54.0
53.5
53.0
52.5
y
−11.5
−11.0
−10.5
x

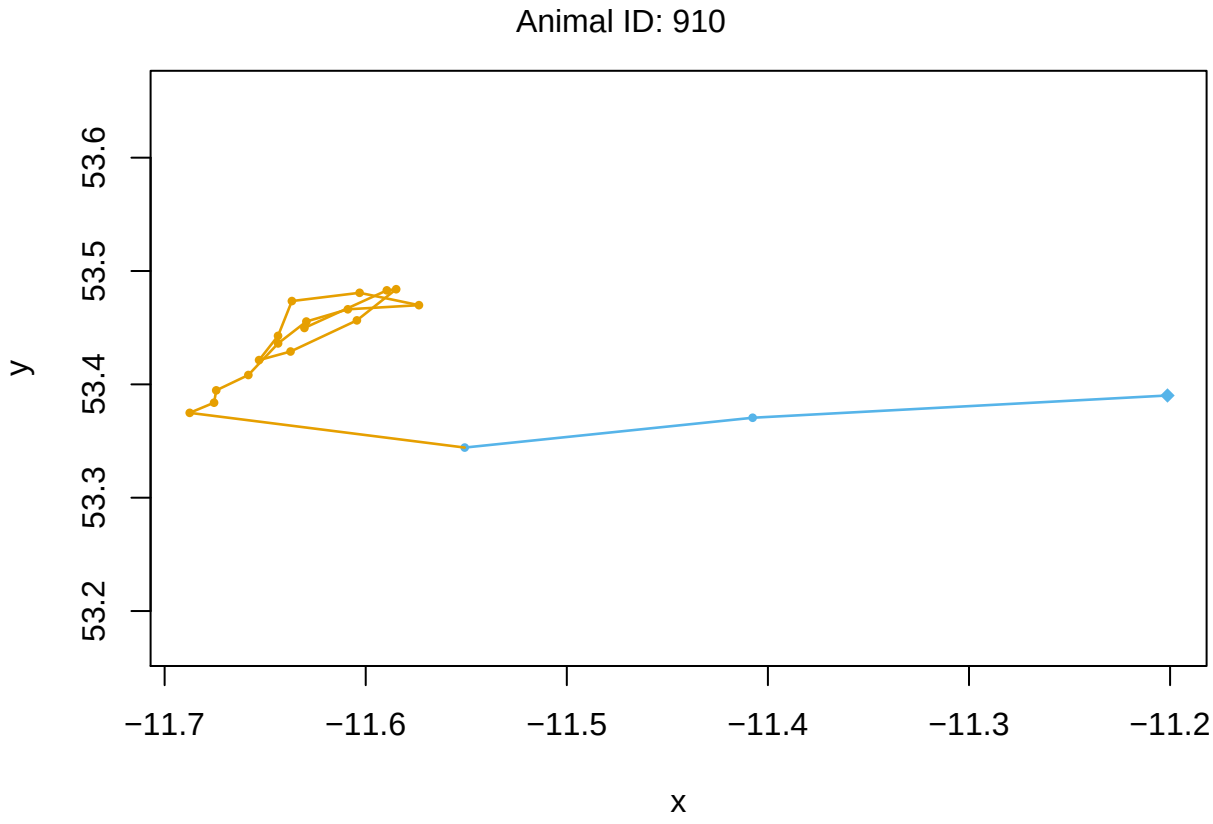

Animal ID: 906
y
53.8
53.6
53.4
53.2
−11.4
−11.2
−11.0
−10.8
−10.6
x


Animal ID: 906B
y
54.0
53.5
53.0
52.5
52.0
−12.0
−11.5
−11.0
−10.5
−10.0
x


Animal ID: 908
y
55
54
53
52
51
−15
−14
−13
−12
−11
−10
x


Animal ID: 909
y
53.6
53.4
53.2
53.0
52.8
−12.2
−12.0
−11.8
−11.6
−11.4
x


Animal ID: 909B
y
54.5
54.0
53.5
53.0
52.5
52.0
−12.5
−12.0
−11.5
−11.0
−10.5
−10.0
x




```
states <- viterbi(m1)
states[1:25]
```

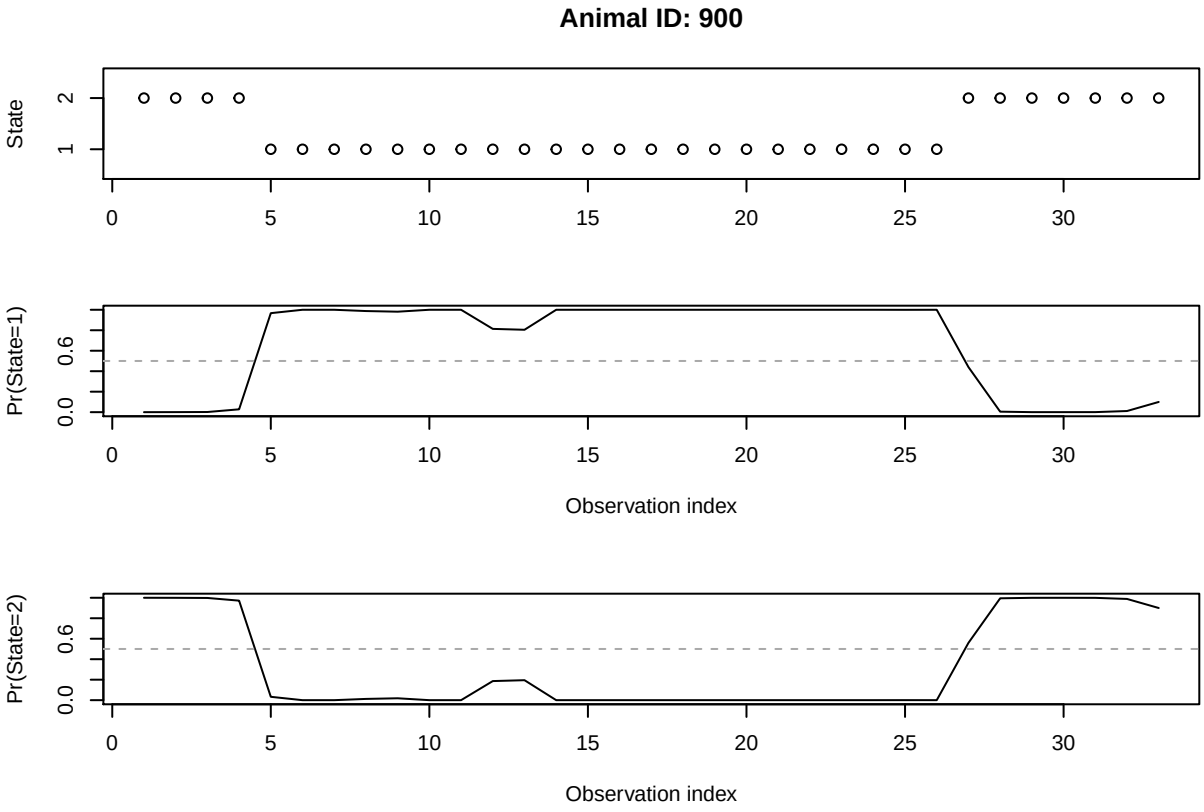
```
##  [1] 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

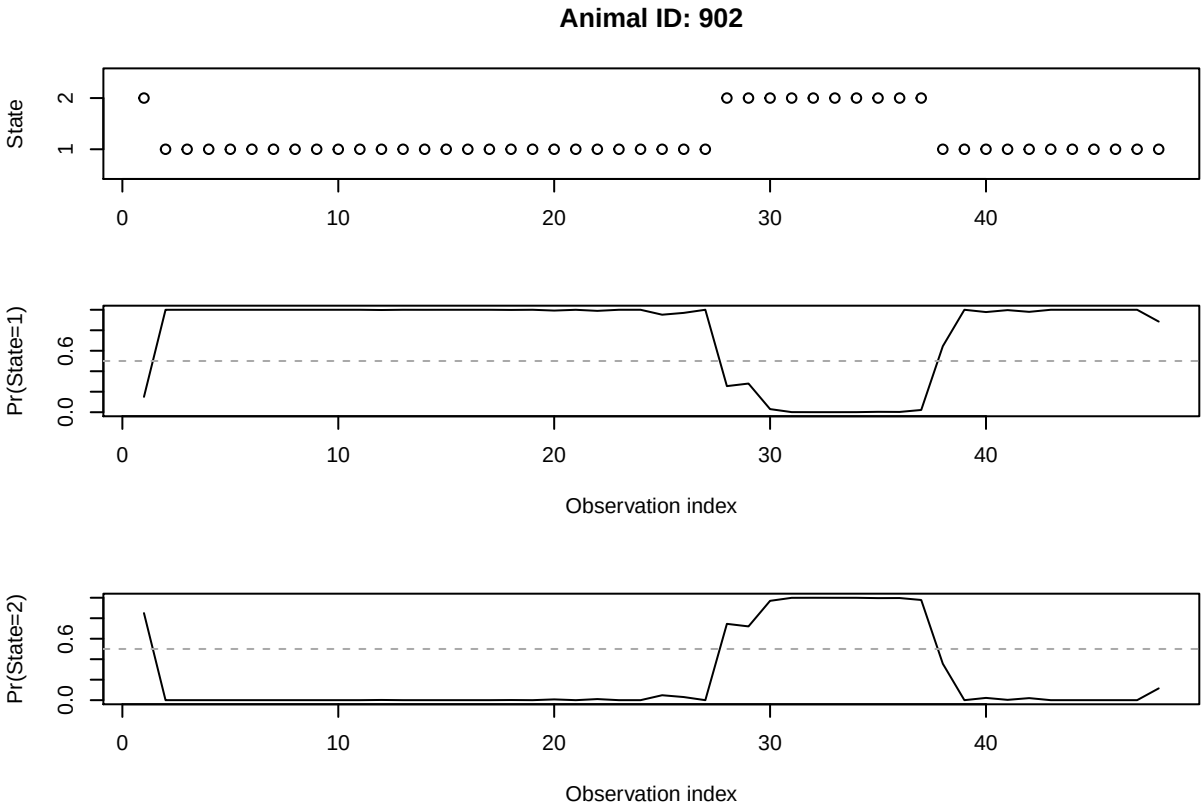
```
sp <- stateProbs(m1)
head(sp)
```

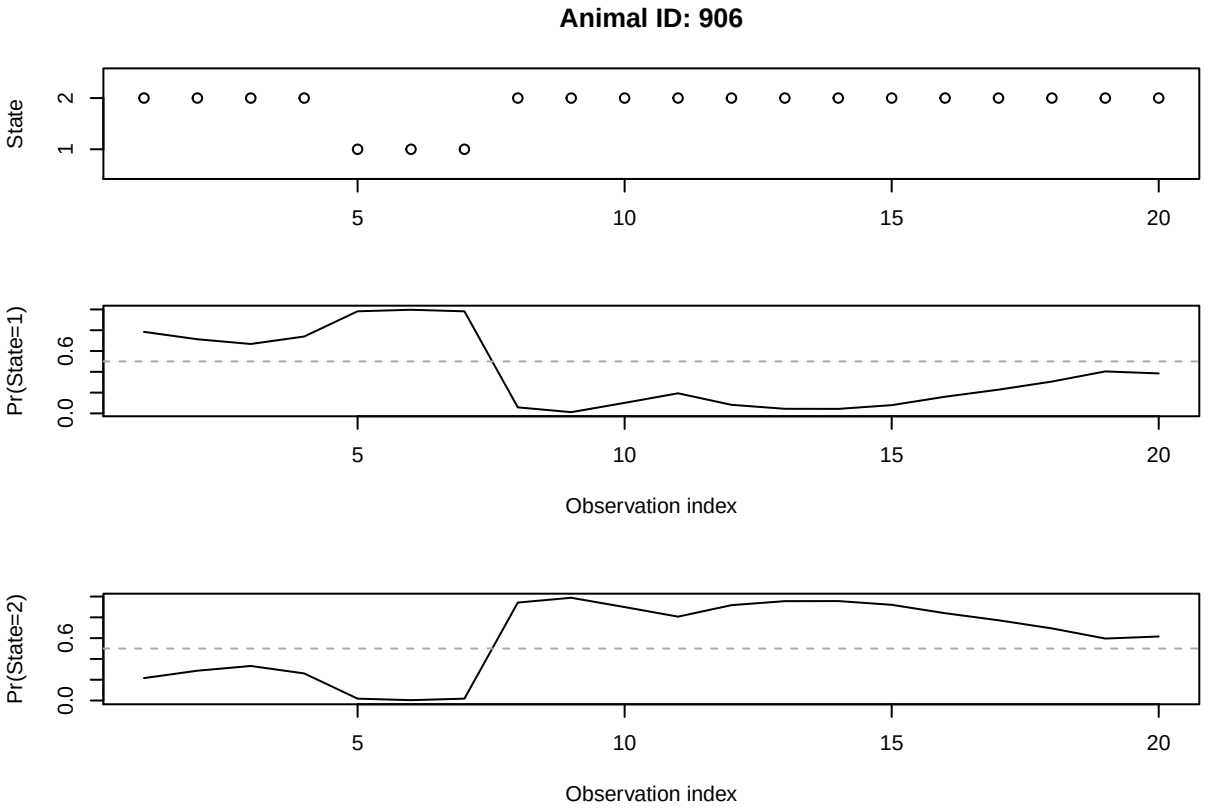
```
##              [,1]         [,2]
## [1,] 3.787847e-09 1.000000e+00
## [2,] 5.412343e-04 9.994588e-01
## [3,] 1.726155e-03 9.982738e-01
## [4,] 2.809875e-02 9.719013e-01
## [5,] 9.670721e-01 3.292788e-02
## [6,] 1.000000e+00 1.879564e-32
```

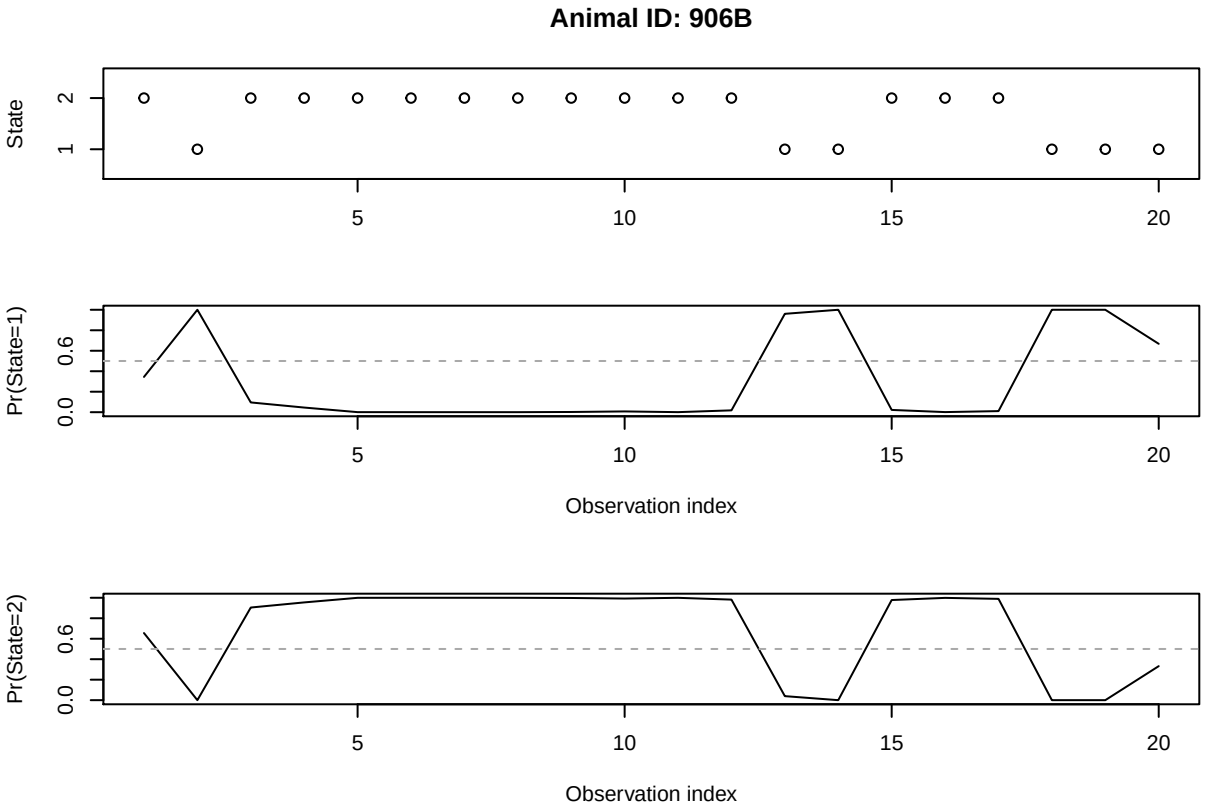
```
plotStates(m1)
```

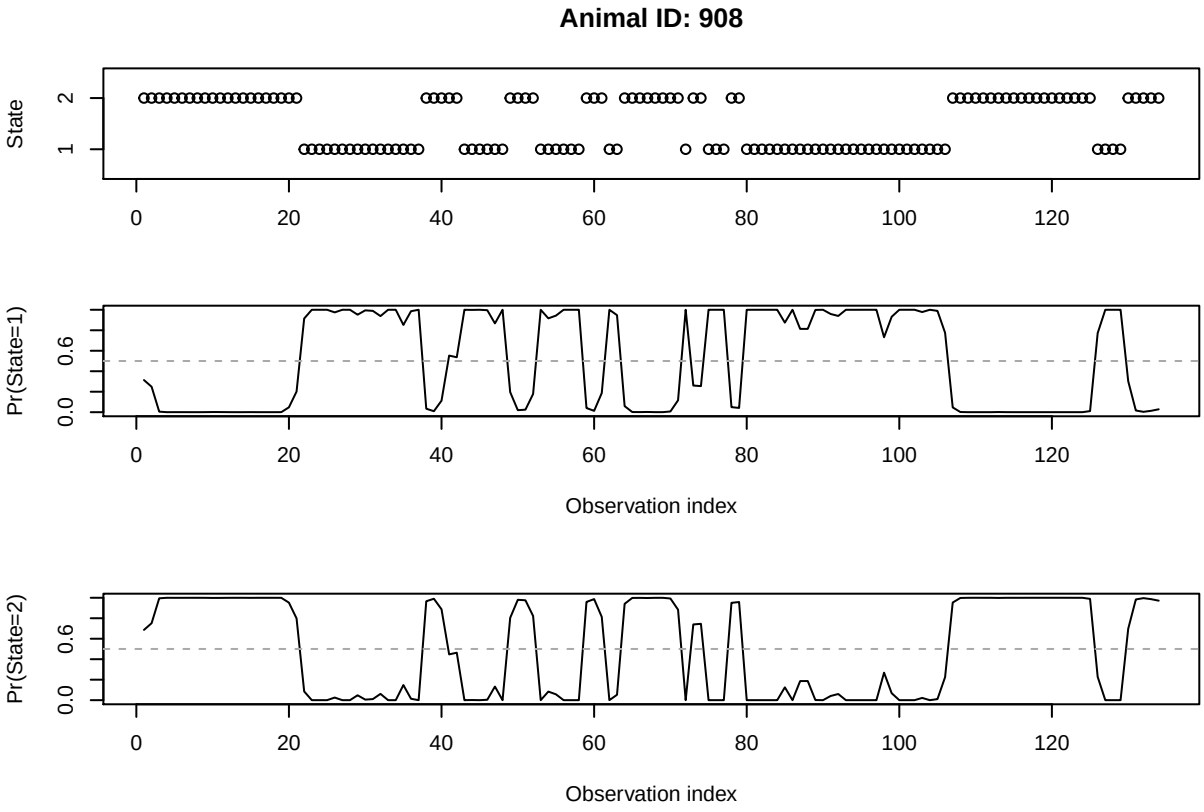
```
## Decoding states sequence... DONE
## Computing states probabilities... DONE
```

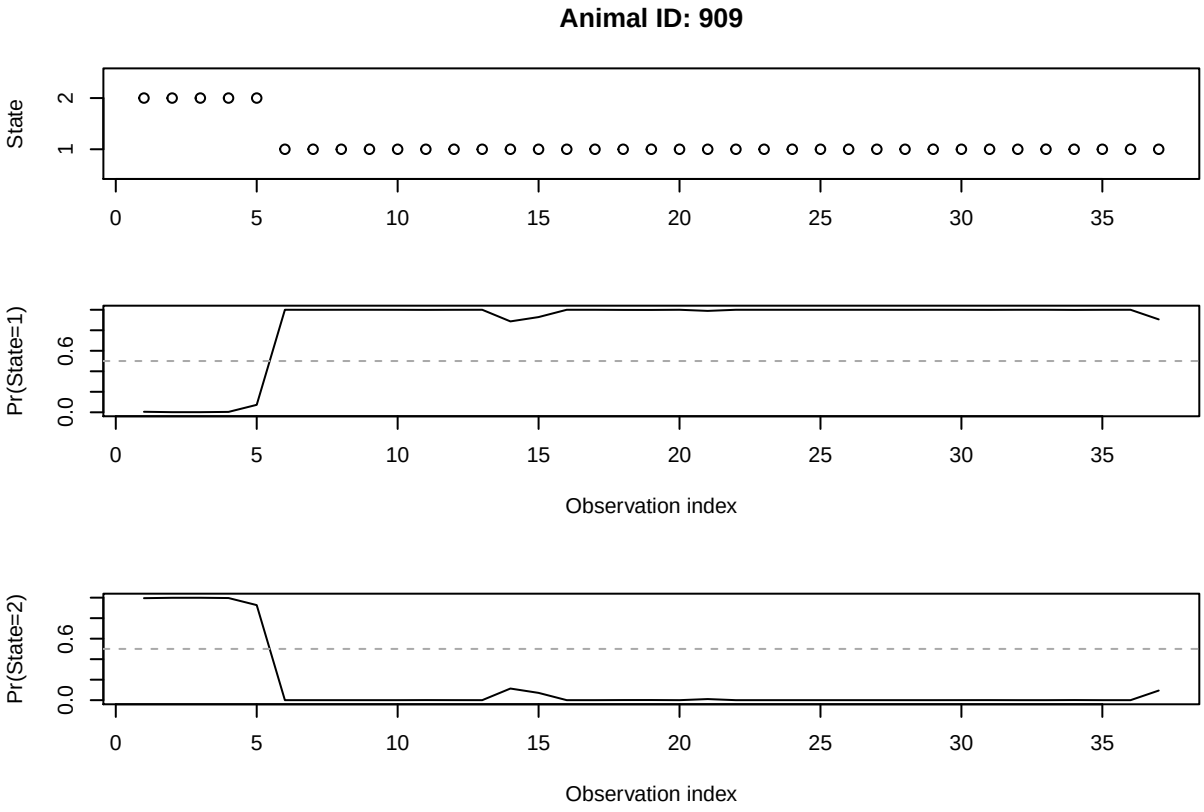

Animal ID: 900
State
1
2
0
5
10
15
20
25
30
Pr(State=1)
0.0
0.6
Observation index
Pr(State=2)
0.0
0.6
Observation index

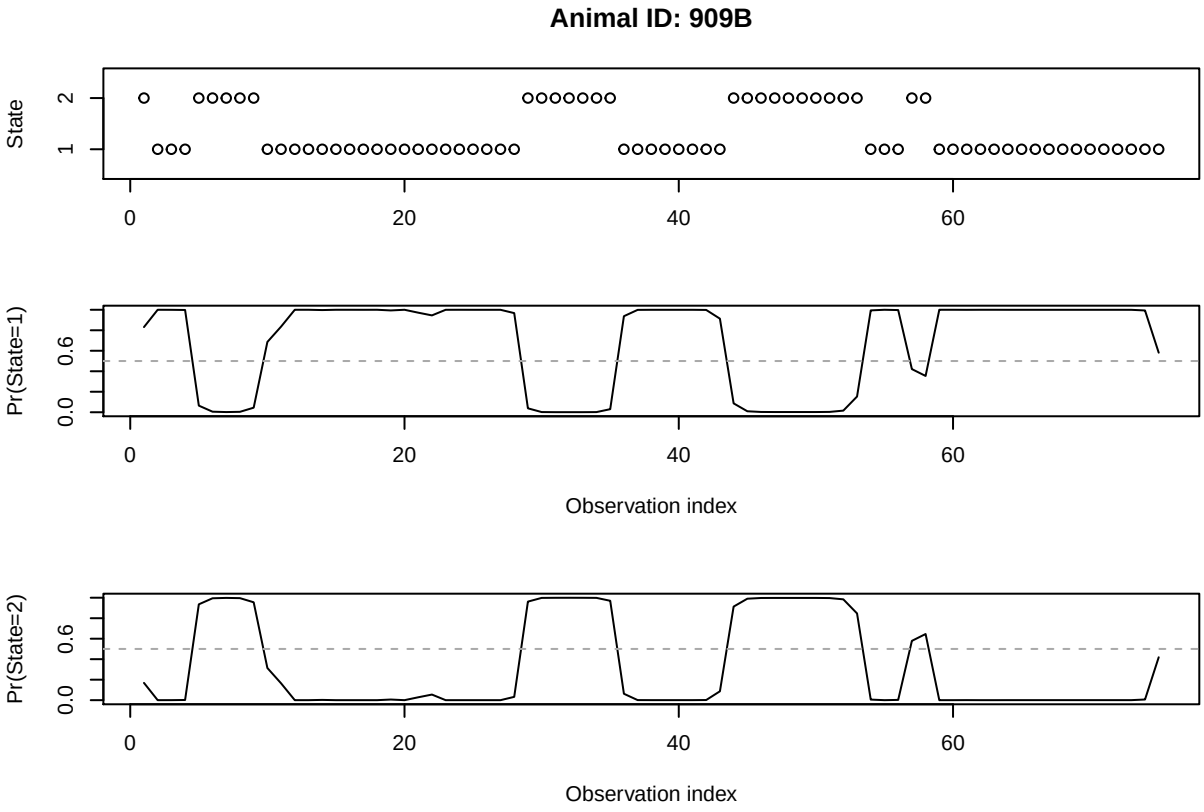

Animal ID: 902
State
1
2
0
10
20
30
40
Pr(State=1)
0.0
0.6
0
10
20
30
40
Observation index
Pr(State=2)
0.0
0.6
0
10
20
30
40
Observation index

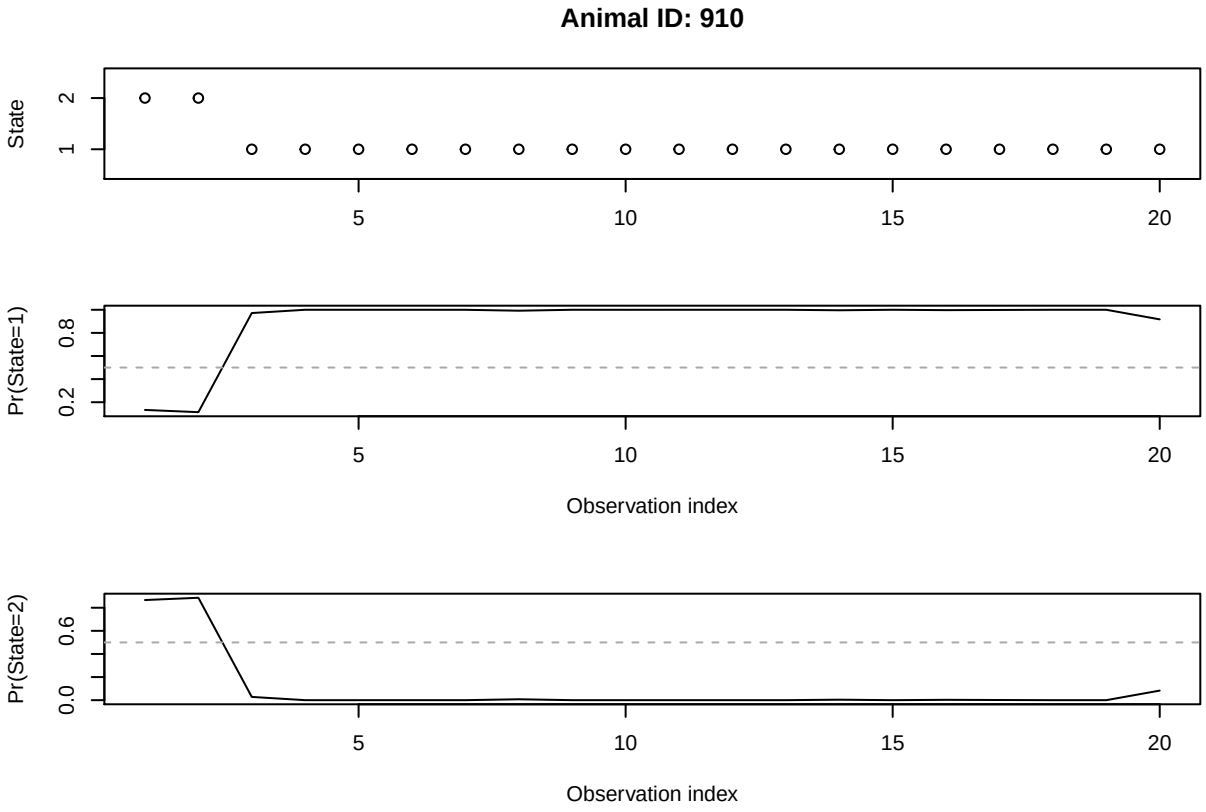

Animal ID: 906
State
1
2
5
10
15
20
Pr(State=1)
0.0
0.6
5
10
15
20
Observation index
Pr(State=2)
0.0
0.6
5
10
15
20
Observation index


Animal ID: 906B
State
1
2
5
10
15
20
Pr(State=1)
0.0
0.6
5
10
15
20
Observation index
Pr(State=2)
0.0
0.6
5
10
15
20
Observation index


Animal ID: 908
State
1
2
0
20
40
60
80
100
120
Pr(State=1)
0.0
0.6
0
20
40
60
80
100
120
Observation index
Pr(State=2)
0.0
0.6
0
20
40
60
80
100
120
Observation index

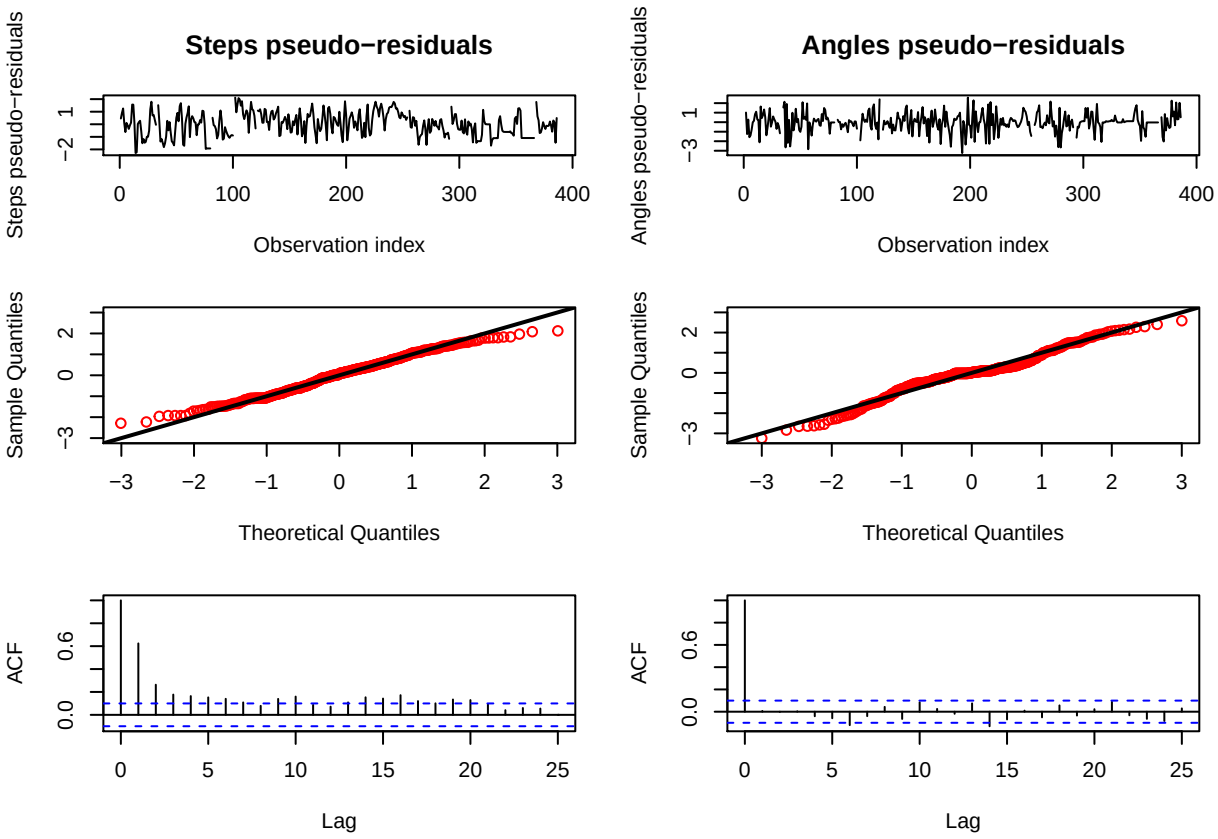

Animal ID: 909
State
1
2
0
5
10
15
20
25
30
35
Pr(State=1)
0.0
0.6
Observation index
Pr(State=2)
0.0
0.6
Observation index


Animal ID: 909B
State
1
2
0
20
40
60
Pr(State=1)
0.0
0.6
0
20
40
60
Observation index
Pr(State=2)
0.0
0.6
0
20
40
60
Observation index




```
# compute the pseudo-residuals
pr <- pseudoRes(m1)
# time series, qq-plots, and ACF of the pseudo-residuals
plotPR(m1)
```

```
## Computing pseudo-residuals... DONE
```




```
# Try interpolating for one well behaved track
```

```
#dataSample<-data[data$ID == 908, ]
#head(dataSample)

# create a trajectory object using adehabitatLT
#tr<-as.ltraj(data.frame(X=dataSample$lon,Y=dataSample$lat),date=dataSample$DateTime,id=dataSample$ID,t
#tstep<-1800 #time step we want for the interpolation, in seconds
#newtr<-redisltraj(tr, u=tstep, type = "time")
#head(newtr)
#head(newtr[[1]])

# convert object of class ltraj to a dataframe
#df<-ld(newtr)
#names(df)[names(df) == 'x'] <- 'lon'
#names(df)[names(df) == 'y'] <- 'lat'
#head(df)

#prepare data with moveHMM
#trackData2 <- df[,c(1,2,11)]
#colnames(trackData2)[3] <- c("ID")
#data3 <- prepData(trackData2,type="LL",coordNames=c("lon","lat"))
#plot(data3,compact=T)

#apply two state HMM
## initial parameters for gamma and von Mises distributions
```

```
#mu0 <- c(1,4) # step mean (two parameters: one for each state)
#sigma0 <- c(0.5,1) # step SD
#stepPar0 <- c(mu0,sigma0)
#angleMean0 <- c(pi,0) # angle mean
#kappa0 <- c(0.7,1.5) # angle concentration
#anglePar0 <- c(angleMean0,kappa0)

#m1 <- fitHMM(data=data3,nbStates=2,stepPar0=stepPar0,anglePar0=anglePar0,
#             formula=~1) # no covariate

#m1
#plot(m1)

#states <- viterbi(m1)
#states[1:25]

#sp <- stateProbs(m1)
#head(sp)
#plotStates(m1)
```

```
# Model with an environmental covariate
```

```
# dataSample<-data[data$ID == 908, ]
# head(dataSample)

# create a trajectory object using adehabitatLT
# tr<-as.ltraj(data.frame(X=dataSample$lon,Y=dataSample$lat),date=dataSample$DateTime,id=dataSample$ID,
# tstep<-1800 #time step we want for the interpolation, in seconds
# newtr<-redisltraj(tr, u=tstep, type = "time")
# head(newtr)
# head(newtr[[1]])

# convert object of class ltraj to a dataframe
# df<-ld(newtr)
# names(df)[names(df) == 'x'] <- 'lon'
# names(df)[names(df) == 'y'] <- 'lat'
# head(df)

# the environmental data will need to be applied to the interpolated data at this point
# for now we'll use non interpolated data for the best track

#prepare data with moveHMM
# trackData2 <- dataSample[,c(1,2,4,5)]
# colnames(trackData2)[3] <- c("ID")
# data3 <- prepData(trackData2,type="LL",coordNames=c("lon","lat"))
# plot(data3,compact=T)

#apply two state HMM
## initial parameters for gamma and von Mises distributions
# mu0 <- c(0.1,1) # step mean (two parameters: one for each state)
# sigma0 <- c(0.1,1) # step SD
# stepPar0 <- c(mu0,sigma0)
# angleMean0 <- c(pi,0) # angle mean
```

```
# kappa0 <- c(1,1) # angle concentration
# anglePar0 <- c(angleMean0,kappa0)

# m1 <- fitHMM(data=data3,nbStates=2,stepPar0=stepPar0,anglePar0=anglePar0,
#              formula=~1) # no covariate
# m2 <- fitHMM(data=data3,nbStates=2,stepPar0=stepPar0,anglePar0=anglePar0,
#              formula=~bathymetry) # covariate 'bathymetry'

## Model selection using the AIC
# print(AIC(m1,m2))

# m1
# plot(m1)
# m2
# plot(m2)
```