Analysing animal movement data with hidden Markov models

🐧 🐬 🐘

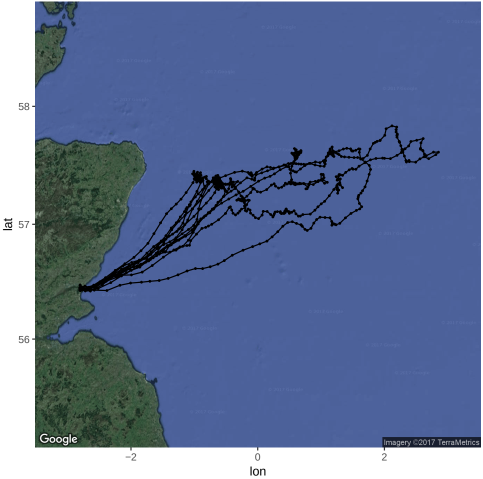
CANSSI/OTN workshop

15 November 2023

Slides adapted from: Théo Michelot

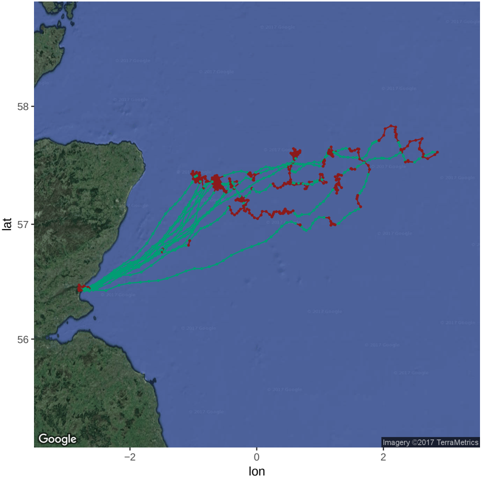
Background on hidden Markov models

HMMs for animal movement



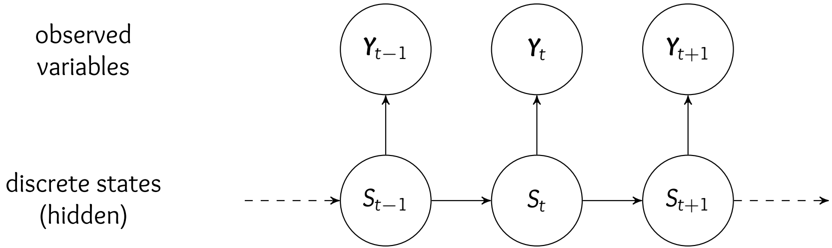
Data from: Russell et al. (2015), "Intrinsic and extrinsic drivers of activity budgets in sympatric grey and harbour seals", Oikos, 124(11).

HMMs for animal movement



Data from: Russell et al. (2015), "Intrinsic and extrinsic drivers of activity budgets in sympatric grey and harbour seals", Oikos, 124(11).

HMM structure

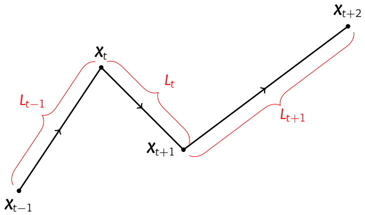


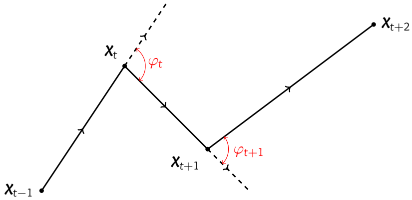
We need sensible ways to describe how the states give rise to the data (i.e., **observation model**) and the state-switching dynamics (i.e., **state process**).

Movement metrics

From GPS data, we can derive $\boldsymbol{Y}_t$ as:

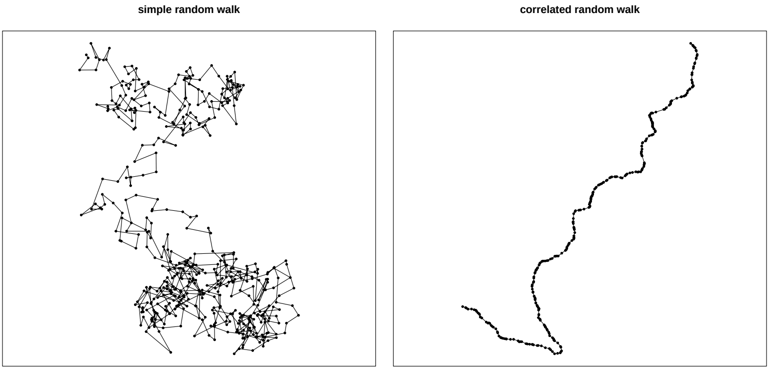
- step lengths (L_t);
- turning angles (φ_t).



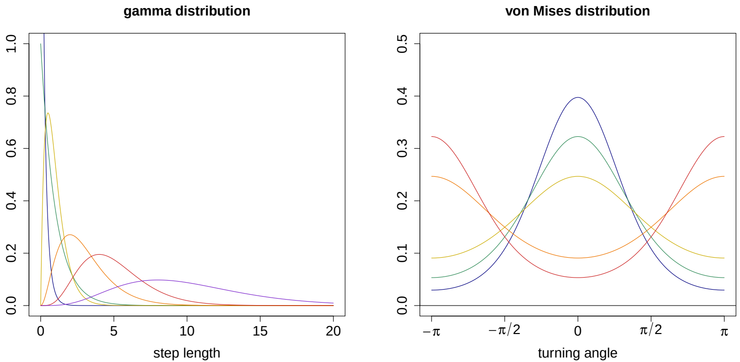


Correlated random walk

A correlated random walk includes **persistence in direction**.
$\longrightarrow$ Correlation between successive directions.



Modelling the steps and angles



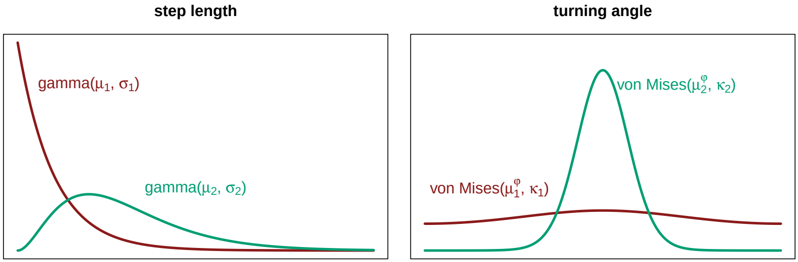
Multistate random walk

Idea: the animal switches between several movement processes, corresponding to several **behaviours**.

Multistate random walk

Idea: the animal switches between several movement processes, corresponding to several **behaviours**.
$\longrightarrow$ **Observation process** = state-dependent distributions.

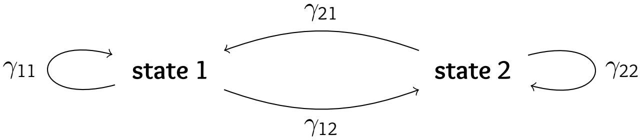
Example:



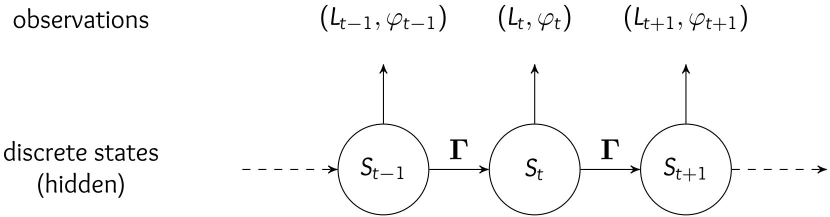
Multistate random walk

Idea: the animal switches between several movement processes, corresponding to several **behaviours**.
$\longrightarrow$ Observation process = state-dependent distributions.
$\longrightarrow$ **State process** = unobserved Markov chain (S_t).

Example:



HMM for animal movement



Example:

$$\begin{cases} L_t \mid S_t = j \sim \text{gamma}(\mu_j, \sigma_j) \\ \varphi_t \mid S_t = j \sim \text{von Mises}(\mu_j^{\varphi}, \kappa_j) \end{cases} \quad \text{and} \quad \boldsymbol{\Gamma} = \begin{pmatrix} \gamma_{11} & \gamma_{12} & \cdots \\ \gamma_{21} & \gamma_{22} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

Covariates (transition probabilities)

Does [*insert covariate*] have an effect on the probability that the animal is [*insert behaviour*]?

$\longrightarrow$ Time-varying transition probabilities $\mathbf{\Gamma}_t$.

In a 2-state model:

$$\begin{cases} \Pr(S_{t+1} = 2 | S_t = 1) = \text{logit}^{-1}(\beta_0^{(12)} + \sum_{i=1}^m \beta_i^{(12)} w_{i,t}) \\ \Pr(S_{t+1} = 1 | S_t = 2) = \text{logit}^{-1}(\beta_0^{(21)} + \sum_{i=1}^m \beta_i^{(21)} w_{i,t}) \end{cases}$$

Covariates (observation model)

Does [*insert covariate*] have an effect on the movement in [*insert behaviour*]?

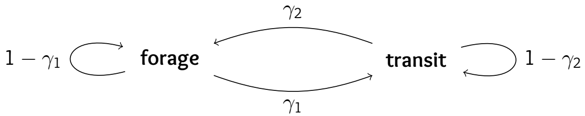
$\longrightarrow$ Covariate-dependent observation parameters.

Example: age-dependent mean step length

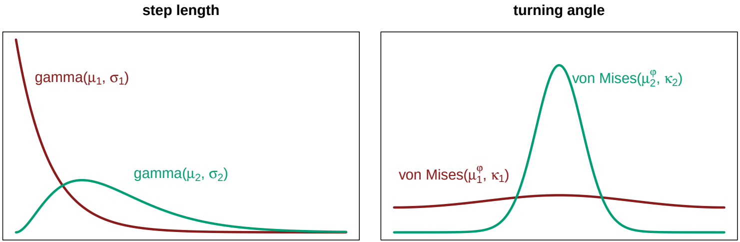
$$\log(\mu_t^{(j)}) = \beta_0^{(j)} + \beta_1^{(j)} age_t$$

HMM assumption: regular sampling frequency

- Transition probabilities assume regular time intervals.



- State-dependent distributions assume regular time intervals.



$\longrightarrow$ We need **regular time intervals** between data rows.

Fitting HMMs in R

Introduction to R packages

Two main packages for animal movement:

- moveHMM (Michelot, Langrock, & Patterson 2016, *MEE*)
- momentuHMM (McClintock & Michelot 2017, *MEE*)

Both available on CRAN:

```
install.packages("moveHMM")
install.packages("momentuHMM")
```

The **package vignettes** are a good place to look for information and get started!

Do I need moveHMM or momentuHMM?

moveHMM is easier to use, momentuHMM is more flexible.

Examples of additional functionalities:

- unlimited number of data streams;
- larger choice of distributions for data streams;
- covariates on the parameters of observation distributions;
- multiple imputation (irregular sampling, measurement error)

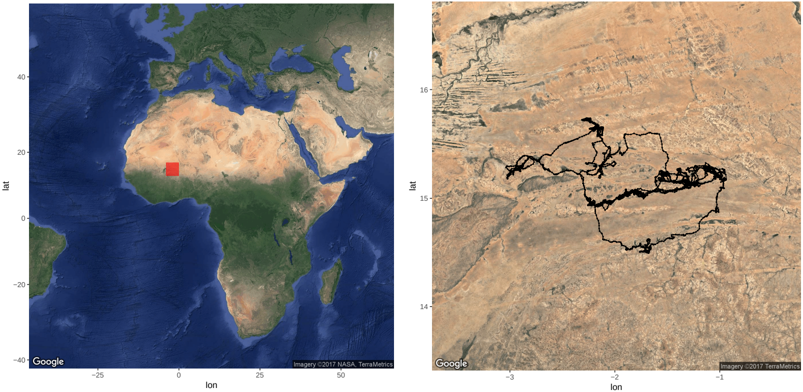
We will be using **momentuHMM** in the practical session today.

Workflow in R

1. Visualise and prepare the data.
2. Choose model formulation:
   $\longrightarrow$ how many states?
   $\longrightarrow$ which distributions for the steps/angles?
   $\longrightarrow$ any covariates?
3. Fit model(s).
4. Visualise model:
   $\longrightarrow$ map of "decoded" tracks;
   $\longrightarrow$ covariate effects.
5. Visualise diagnostics (pseudo-residuals).

Elephant case study

Hourly locations over one year + temperature recordings.



Wall et al. (2014), "Elliptical time-density model to estimate wildlife utilization distributions" Methods in Ecology and Evolution, 5 (780–790).

(From the Movebank data repository.)

1. Prepare the data
2. Fit the model
3. Visualise the results
4. Include covariates

Formatting the data

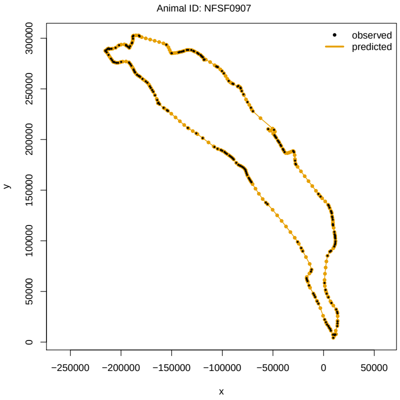
Several steps in data processing:

- ensure data is at a regular time resolution;
- decide how to deal with missing data (locations/covariates);
- calculate movement metrics (i.e., data streams) and obtain covariates;
- visualise data prior to model fitting.

Ensure a regular resolution: crawl

A common option for regularisation is to interpolate locations with a continuous-time correlated random walk (crawl) via momentuHMM. Caution:

- Not good over large gaps.
- Be mindful that these locations have uncertainty.



Formatting the data

Data are already at a 1-hour resolution, without missing data.

```
track <- read.csv("elephant.csv")

head(track)

##   ID         x        y temp tod
## 1  1 -2.160167 15.65350   38  17
## 2  1 -2.160075 15.65452   35  18
## 3  1 -2.159902 15.65451   32  19
## 4  1 -2.159435 15.65489   30  20
## 5  1 -2.158113 15.65512   29  21
## 6  1 -2.157848 15.65461   28  22
```

prepData

Function for basic **data processing**

```r
# longitude-latitude
data <- prepData(track, type="LL")
```

⟶ Default in moveHMM.
⟶ Step lengths computed with `spDistsN1` (package sp).
⟶ Turning angles computed with `bearing` (package geosphere).

```r
# Easting-Northing
data <- prepData(track, type="UTM")
```

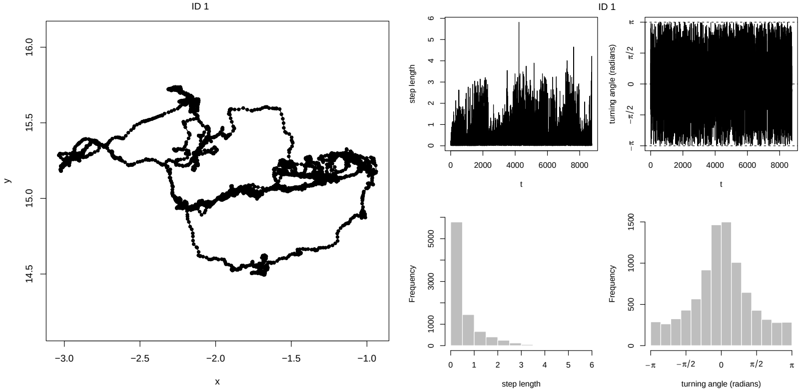
⟶ Default in momentuHMM.

prepData

```r
library(momentuHMM)

data <- momentuHMM::prepData(track, type = "LL", covNames = c("temp", "tod"))

## Warning in prepData.default(track, type = "LL", covNames = c("temp",
"tod")):  There are 15 missing covariate values.  Each will be replaced
by the closest available value.

head(data)

##   ID       step      angle         x        y temp tod
## 1  1 0.11329503         NA -2.160167 15.65350   38  17
## 2  1 0.01862565 -1.5534731 -2.160075 15.65452   35  18
## 3  1 0.06477228  0.7575722 -2.159902 15.65451   32  19
## 4  1 0.14400818 -0.5090123 -2.159435 15.65489   30  20
## 5  1 0.06253091 -1.2781269 -2.158113 15.65512   29  21
## 6  1 0.20910448  2.8261263 -2.157848 15.65461   28  22
```

Visualise the data

```r
plot(data, ask=FALSE)
```



1. Prepare the data
2. Fit the model
3. Visualise the results
4. Include covariates

fitHMM

Fitting function in momentuHMM. Need to choose:

- number of states;
- distributions for the data streams;
- initial parameters;
- covariates (covered later).

```
m <- fitHMM(data,
            nbStates = 2,
            dist = list(step = "gamma", angle = "vm"),
            Par0 = list(step = c(stepMean0, stepSD0),
                        angle = c(angleMean0, angleCon0)))
```

fitHMM: number of states

There is no general method to select the “optimal” number of states.

1. Fit 2-state model, 3-state model, etc., and compare them:
   - Model checking using pseudo-residuals.
   - Comparison with AIC tends to favour models with more states.

   ```
   AIC(mod2, mod3, mod4)
   ```

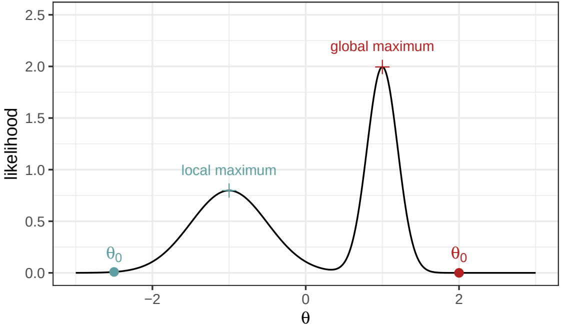
2. **Biological interpretation!**

Pohle et al. (2017). Selecting the number of states in hidden Markov models: pragmatic solutions illustrated using animal movement, *JABES*.

fitHMM: initial parameters

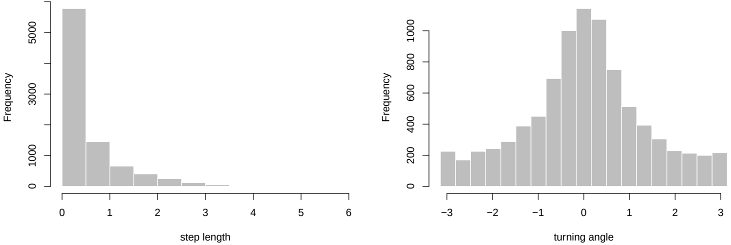
fitHMM uses a numerical optimiser to maximise the likelihood

- we want to find the “global” maximum (i.e., best model)
- the optimisation will sometimes identify “local” maxima
- we need to choose good starting values!



fitHMM: initial parameters

1. Plot histograms of step lengths and turning angles.



2. "What are some plausible values for the parameters?"

```
stepMean0 <- c(0.1, 1) # mean of step length distribution
stepSD0 <- c(0.2, 1.2) # SD of step length distribution
angleMean0 <- c(0, 0)  # mean of angle distribution
angleCon0 <- c(0.5, 5) # concentration of angle distribution
```

3. Try many different initial parameters, maybe chosen at random.

fitHMM

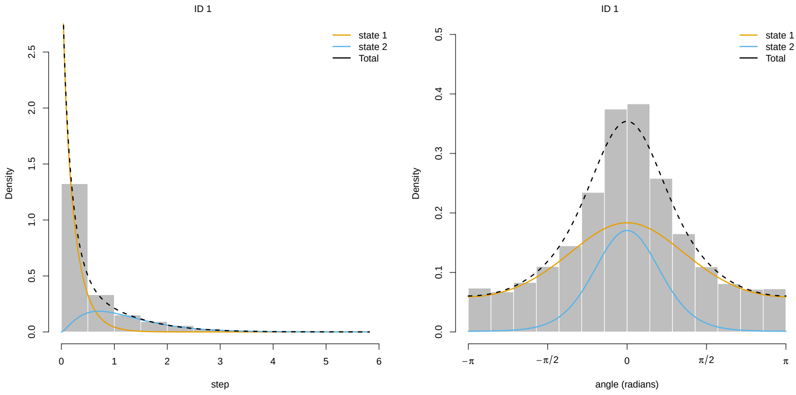
```r
# list of data stream distributions
dist <- list(step = "gamma", angle = "vm")

# define initial parameters (one value for each state)
stepMean0 <- c(0.1, 1) # mean of step length distribution
stepSD0 <- c(0.2, 1.2) # SD of step length distribution
angleMean0 <- c(0, 0)  # mean of angle distribution
angleCon0 <- c(0.5, 5) # concentration of angle distribution

# create list of initial parameters
# by default, the angular mean is NOT estimated
Par0 <- list(step = c(stepMean0, stepSD0), angle = c(angleCon0))

# fit 2-state model
m <- momentuHMM::fitHMM(data,
                        nbStates = 2,
                        dist = dist,
                        Par0 = Par0)
```

Fitted model

```
## Value of the maximum log-likelihood: -15414.76
##
##
## step parameters:
## ----------------
##        state 1   state 2
## mean 0.2248240 1.2082112
## sd   0.2401748 0.7743635
##
## angle parameters:
## -----------------
##                 state 1  state 2
## mean          0.0000000 0.000000
## concentration 0.5651883 2.467007
##
## Regression coeffs for the transition probabilities:
## --------------------------------------------------
##                1 -> 2    2 -> 1
## (Intercept) -2.245608 -1.299824
##
## Transition probability matrix:
## ------------------------------
##           state 1   state 2
## state 1 0.9042710 0.0957290
## state 2 0.2141946 0.7858054
##
## Initial distribution:
## ---------------------
##      state 1      state 2
## 9.999893e-01 1.070558e-05
```

Plot state-dependent distributions

```r
plot(m)
```

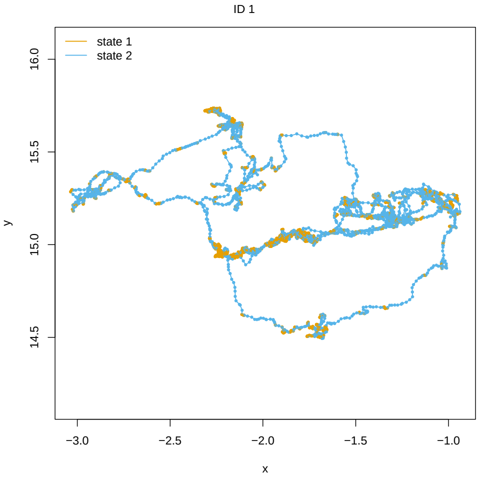


Decode the state process

```r
# global decoding of the state sequence
states <- viterbi(m)
head(states)

## [1] 1 1 1 1 1 1

# local decoding to get state probabilities
sp <- stateProbs(m)
head(sp)

##        state 1      state 2
## [1,] 0.9985081 0.0014919230
## [2,] 0.9999718 0.0000282271
## [3,] 0.9991291 0.0008709232
## [4,] 0.9957905 0.0042094540
## [5,] 0.9996394 0.0003605643
## [6,] 0.9997340 0.0002660305
```

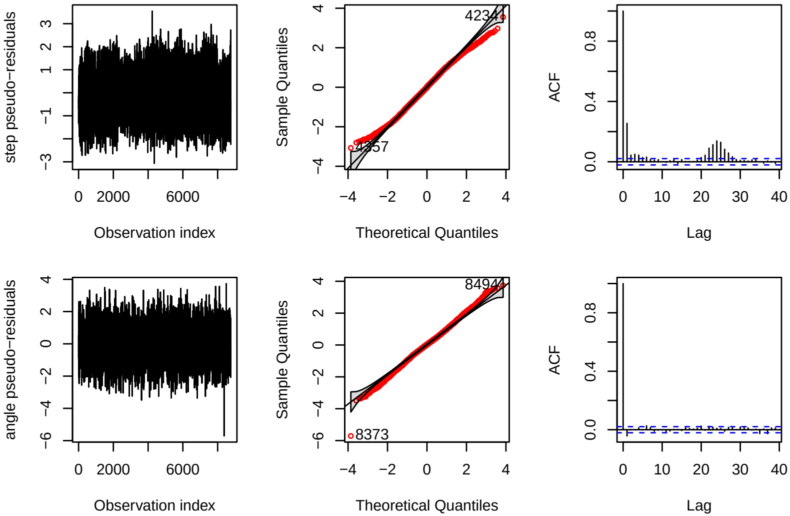
Plot the Viterbi sequence

```
plot(m)
```



Pseudo-residuals

```r
plot(m)
```



Building more complex models

1. Prepare the data
2. Fit the model
3. Visualise the results
4. Include covariates

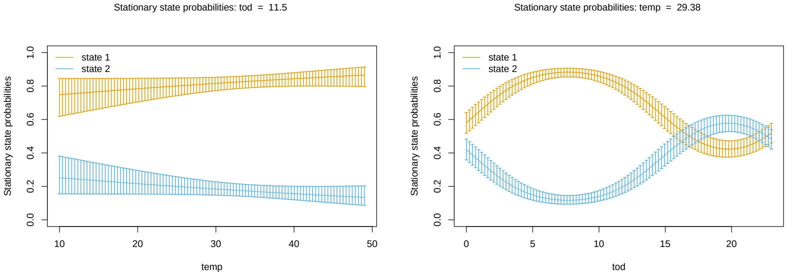
Covariates (transition probabilities)

How do temperature and time of day affect the probability of switching between states?

```
# formula for transition probabilities
formula <- ~temp + cosinor(tod, period = 24)

# generate initial parameters for new model
Par0_m2 <- getPar0(model = m, formula = formula)

m2 <- momentuHMM::fitHMM(data,
                         nbStates = 2,
                         dist = dist,
                         Par0 = Par0_m2$Par,
                         beta0 = Par0_m2$beta,
                         formula = formula)
```

Results

```r
plotStationary(m2, plotCI = TRUE)
```



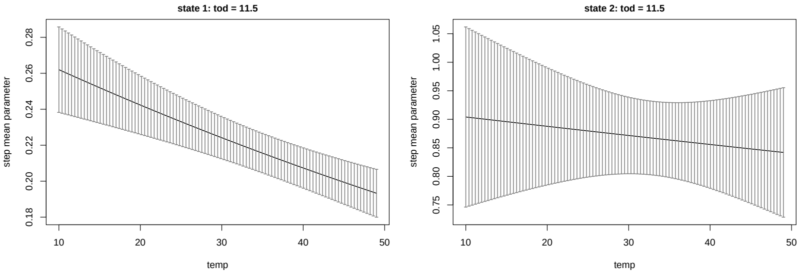
Covariates (step length distribution)

How do temperature and time of day affect the mean step length (within each state)?

```
# formulas for observation parameters
DM <- list(step=list(mean = ~temp + cosinor(tod,period=24),
                     sd = ~1))

# generate initial parameters for new model
Par0_m3 <- getPar0(model = m, DM = DM)

m3 <- momentuHMM::fitHMM(data,
                         nbStates = 2,
                         dist = dist,
                         Par0 = Par0_m3$Par,
                         beta0 = Par0_m3$beta,
                         DM = DM)
```

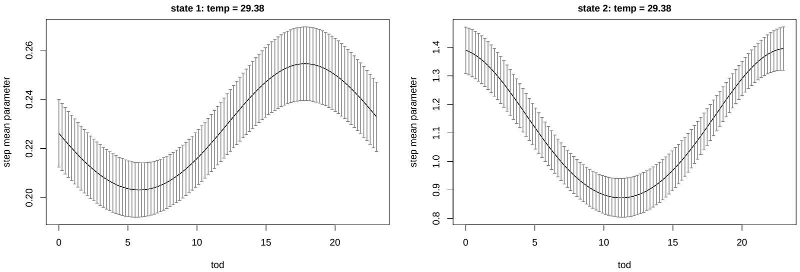
Results: effect of temperature

```
plot(m3, plotCI=TRUE, ask=FALSE)
```



Results: effect of time of day

```r
plot(m3, plotCI=TRUE, ask=FALSE)
```

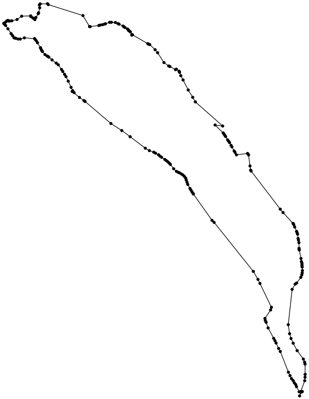


Other options and extensions

Multiple imputation

Solution to irregular sampling and measurement error:

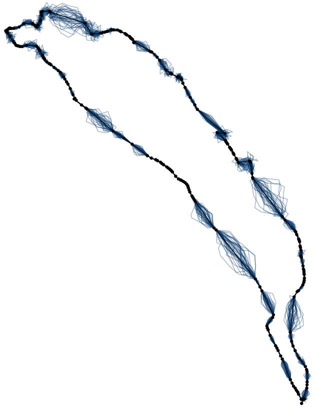
- Fit continuous-time movement model (crawl);
- Draw many regularly-sampled realisations from the fitted model;
- Fit a HMM to each realisation.



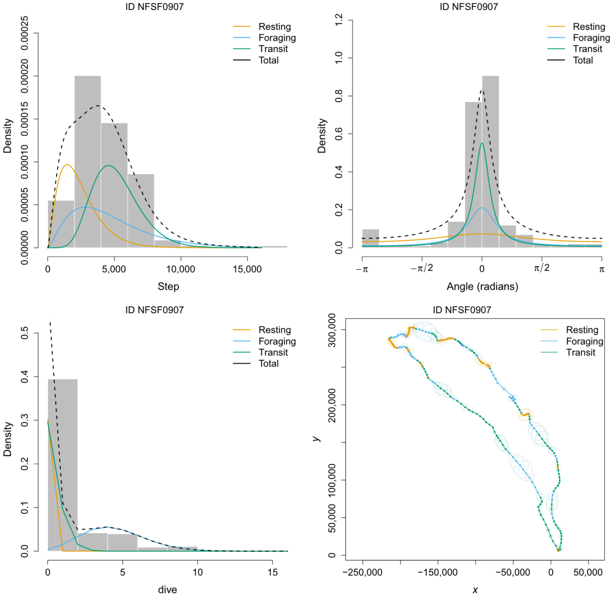
Multiple imputation

Solution to irregular sampling and measurement error:

- Fit continuous-time movement model (crawl);
- Draw many regularly-sampled realisations from the fitted model;
- Fit a HMM to each realisation.

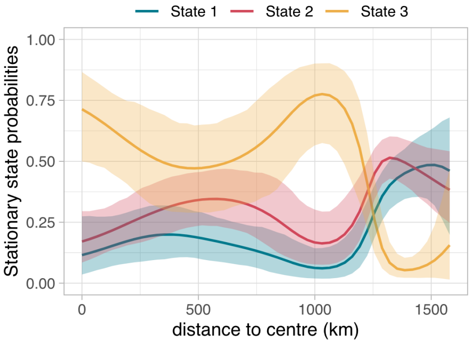


Additional data streams (e.g., dive data)



Non-linear and random effects

New package **hmmTMB** for more complex HMMs



Michelot T. 2023. In press at Journal of Statistical Software.

Thanks!