Flexible animal movement modelling using hmmTMB

Théo Michelot

Contents

1 Introduction

In this vignette, we showcase how hmmTMB can be used to analyse time series of animal locations. This is a popular application of hidden Markov models (HMMs), to identify movement modes often interpreted as behavioural states (Patterson et al. (2009); Langrock et al. (2012)). There are two specialised R packages focused on HMMs for the analysis of animal movement: moveHMM and momentuHMM (Michelot, Langrock, and Patterson (2016); McClintock and Michelot (2018)). Those might be a good alternative; in particular, momentuHMM has additional functionalities for processing animal telemetry data, and to specify some complex movement models (e.g., biased random walks). The main advantage of hmmTMB over those packages is the possibility to include random effects on the model parameters, as well as non-parametric effects of covariates (with automatic smoothness selection).

The most common approach is to assume that, within each state, the animal's movement follows a correlated random walk, i.e., a model that captures the speed of movement and the degree of persistence in the movement. This is convenient because animals often alternative between periods of fast, directed movement (e.g., "transiting", "exploring"), and periods of slow, undirected movement (e.g., "resing", "foraging"). There are two main ways to model correlated random walks: based on step lengths and turning angles (Morales et al. (2004)), or based on locations directly (Jonsen, Mills Flemming, and Myers (2005)). Both approaches can be implemented in hmmTMB, and we describe them in turn below.

Data requirements

We need to make two key assumptions about the data, which may be limiting in some animal movement applications: (a) the locations are observed at regular time intervals, and (b) the locations are observed with negligible measurement error.

In cases where these assumptions are not satisfied, another modelling approach will be needed. Various state-space models have been developed to accommodate noisy, irregular animal trajectories (Jonsen, Mills Flemming, and Myers (2005); Johnson et al. (2008)). Relevant R packages include crawl (Johnson et al. (2008)) and foieGras (Jonsen et al. (2020)), and momentuHMM has dedicated functions to prepare the data using crawl before an HMM analysis.

We start by loading a few packages, including moveHMM, which contains a data set of wild haggis movement data that we will use here, as well as functions for data preparation.

```
library(moveHMM)
library(ggplot2)
```

```
theme_set(theme_bw(13))
library(hmmTMB)
```

2 Step-and-turn model

The most common correlated random walk formulation is based on two derived variables: the step length L_t (distance between successive locations) and the turning angle φ_t (angle between successive steps). We consider the followind widely-used HMM formulation:

$$\begin{cases} L_t \mid \{S_t = j\} \sim \text{gamma}(\mu_j, \sigma_j) \\ \varphi_t \mid \{S_t = j\} \sim \text{von Mises}(\theta_j, \kappa_j^2) \end{cases}$$

where, in state j, $\mu_j > 0$ is the mean step length, σ_j is the standard deviation of step length, $-\pi < \theta_j \leq \pi$ is the mean turning angle, and $\kappa_j > 0$ is the concentration of turning angle.

2.1 Data preparation

In many movement studies, HMMs are used to analyse time series of step lengths and turning angles, to identify phases with different levels of speed and/or directional persistence. Those variables can directly be computed from the locations; for convenience, we use the function `prepData()` from moveHMM to do it here.

```
head(haggis_data)
```

```
  ID         x         y     slope      temp
1  1  0.000000  0.000000 25.957002 10.344959
2  1 -1.068761 -0.194650 18.606632  8.352531
3  1 -6.152549  2.051343 16.524004 13.529650
4  1 -6.703983  3.338480  9.154917 10.951095
5  1 -6.541667  3.553843  5.547686 11.243328
6  1 -7.160298  1.960377  8.129402 13.187280
```

```
# Get step lengths and turning angles
hmm_data <- prepData(haggis_data, type = "UTM")
head(hmm_data)
```

```
  ID      step      angle         x         y     slope      temp
1  1 1.0863417         NA  0.000000  0.000000 25.957002 10.344959
2  1 5.5578218 -0.5961622 -1.068761 -0.194650 18.606632  8.352531
3  1 1.4002860 -0.7500230 -6.152549  2.051343 16.524004 13.529650
```

```
4  1 0.2696813 -1.0506197 -6.703983  3.338480  9.154917 10.951095
5  1 1.7093394 -2.8660552 -6.541667  3.553843  5.547686 11.243328
6  1 1.1529149  2.3676683 -7.160298  1.960377  8.129402 13.187280
```

The columns of the data frame are:

- `ID` is an identifier the track or individual. This data set has three different tracks, to which we will fit a common model.
- `step` and `angle` will be the response variables for the HMM.
- `x` and `y` are the coordinates of the observed locations. They will not be used in the HMM analysis directly.
- `slope` and `temp` are environmental covariates, and we will demonstrate how their effects on the transition probabilities can be estimated.

2.2 Model specification

An HMM has two components: a hidden state process, and an observation model. Those two parts can be specified separately in hmmTMB. We first create an object of class `MarkovChain` for the hidden process model. Here, we just indicate that it is a 2-state process, i.e., we assume that the movement patterns are driven by two underlying states.

```
hid1 <- MarkovChain$new(data = hmm_data, n_states = 2)
```

Defining the observation process takes only a little bit more work. We need to specify the family of distribution used for each response variable; here, a gamma distribution for step length, and von Mises distribution for turning angle. Specifically, we use the `gamma2` distribution, which is formulated in terms of a mean and standard deviation, rather than shape and scale.

We also need to choose initial parameter values, which will be used as a starting point for model fitting. The moveHMM package has a vignette discussing some suggested approaches to choosing those initial values: https://cran.r-project.org/package=moveHMM/vignettes/moveHMM-starting-values.pdf.

The distributions need to be stored in a list (with one element for each observed variable), and the initial parameters are in a nested list, as shown below. Finally, we can use those to create an object of class `Observation`.

```
# Observation distributions
dists <- list(step = "gamma2", angle = "vm")
```

```r
# Initial observation parameters
par0 <- list(step = list(mean = c(1, 5), sd = c(1, 5)),
             angle = list(mu = c(0, 0), kappa = c(1, 5)))

obs1 <- Observation$new(data = hmm_data, dists = dists,
                        par = par0, n_states = 2)
```

2.3 Model fitting

We can now create an `HMM` object, which combines the hidden process and the observation model. Printing it shows the model formulation, as well as the initial parameter values that it stores. Note that, because we did not specify initial parameter values for the `MarkovChain` object, it defaulted to a matrix with 0.9 on the diagonal, but we could have entered it manually using the argument `tpm`.

```r
hmm1 <- HMM$new(obs = obs1, hid = hid1)

hmm1
```

```
########################
## Observation model ##
########################
+ step ~ gamma2(mean, sd)
  * mean.state1 ~ 1
  * mean.state2 ~ 1
  * sd.state1 ~ 1
  * sd.state2 ~ 1

+ angle ~ vm(mu, kappa)
  * mu.state1 ~ 1
  * mu.state2 ~ 1
  * kappa.state1 ~ 1
  * kappa.state2 ~ 1

> Initial observation parameters (t = 1):
            state 1 state 2
step.mean         1       5
step.sd           1       5
```

```
angle.mu          0       0
angle.kappa       1       5

#########################
## State process model ##
#########################
        state 1 state 2
state 1       .      ~1
state 2      ~1       .

> Initial transition probabilities (t = 1):
        state 1 state 2
state 1     0.9     0.1
state 2     0.1     0.9
```

We fit the model using the `fit` method, which takes a few seconds. After the model is fitted, printing it shows the estimated (rather than initial) parameter values. Note that, in models where the parameters depend on covariates, only the parameters corresponding to the first row of covariates are shown, which is why the output says `(t = 1)`. Here, the parameters are not time-varying, so this does not matter.

```
hmm1$fit(silent = TRUE)

hmm1
```

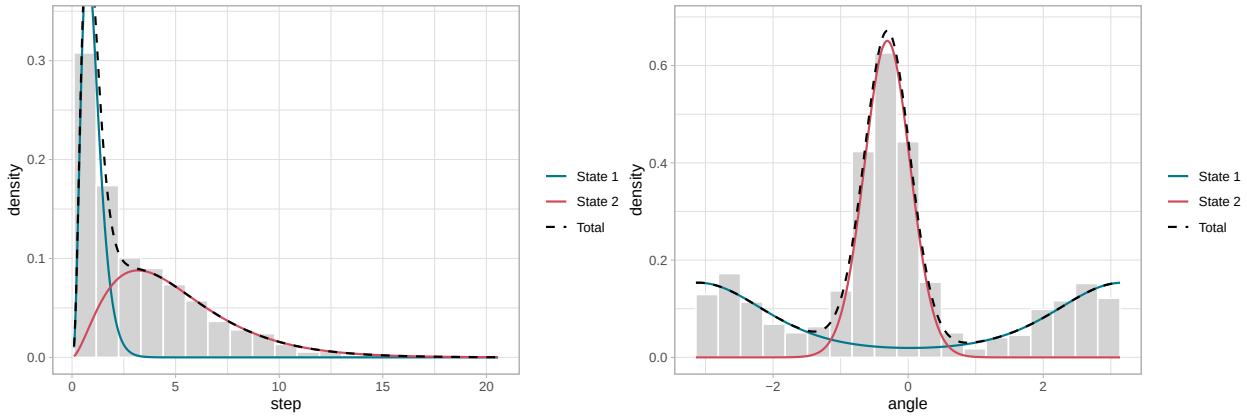
```
########################
## Observation model ##
########################
+ step ~ gamma2(mean, sd)
  * mean.state1 ~ 1
  * mean.state2 ~ 1
  * sd.state1 ~ 1
  * sd.state2 ~ 1

+ angle ~ vm(mu, kappa)
  * mu.state1 ~ 1
  * mu.state2 ~ 1
  * kappa.state1 ~ 1
  * kappa.state2 ~ 1
```

```
> Estimated observation parameters (t = 1):
            state 1 state 2
step.mean     0.998   5.022
step.sd       0.494   3.039
angle.mu     -3.110  -0.308
angle.kappa   1.034   8.773

#########################
## State process model ##
#########################
        state 1 state 2
state 1       .      ~1
state 2      ~1       .

> Estimated transition probabilities (t = 1):
        state 1 state 2
state 1   0.847   0.153
state 2   0.120   0.880
```

2.4 Model visualisation

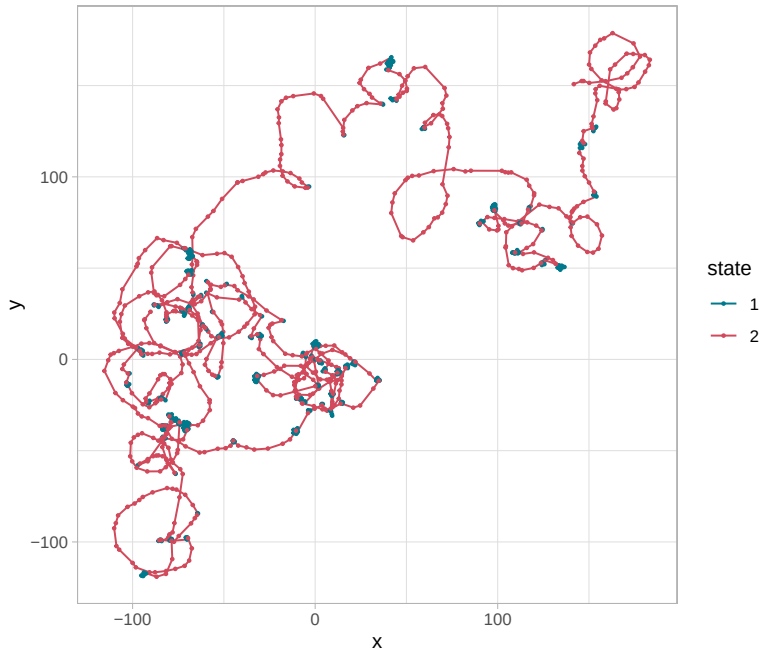
The fitted model can be plotted in a few different ways. The method `plot_dist` creates a plot of the state-dependent distributions for the response variable specified as input. We can use such plots to help with interpretation of the states, to see how much overlap there is between distributions in different states, and to check whether the distributions seem to fit the data well.

```
# Plot state-dependent distributions of steps and angles
hmm1$plot_dist("step")
hmm1$plot_dist("angle")
```



We can also use `plot_ts` to show a time series plot of one or two observed variables, coloured by the Viterbi-estimated state sequence. The variables don't have to be those to which the model was fitted, so we can for example use `x` and `y` here, to create a map of the tracks. This code also highlights that the plotting functions all return ggplot objects, which can be edited as needed.

```
# Plot track coloured by estimated states
hmm1$plot_ts("x", "y") +
  geom_point(size = 0.5) +
  coord_equal()
```



2.5 Covariates

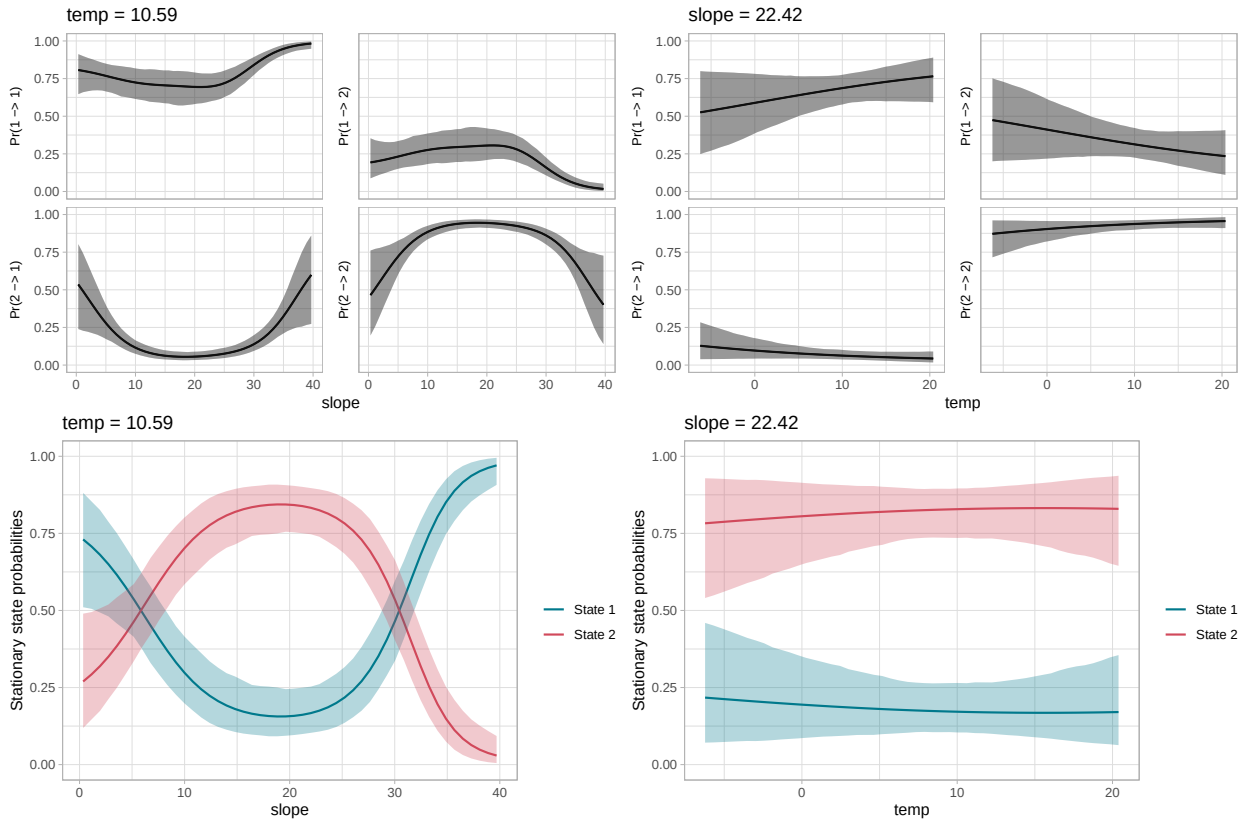
A popular extension of HMMs in ecology consists in including the effects of time-varying (or individual-specific) covariates on parameters of the model. This can for example be used to estimate effects of covariates on the transition probabilities of the hidden state process, and we illustrate this case here.

Consider that we assume that temperature has a linear effect on transition probabilities, whereas slope has an unknown non-linear effect. The latter can be specified using the syntax from the package mgcv (which implements generalised additive models in R), with the function `s` for a smooth term. We create a new `MarkovChain`, with the new formula, and the `Observation` object is the same as before because the observation model hasn't changed. This time, fitting the model takes a little longer because of the need to estimate the non-parametric relationship between transition probabilities and slope.

```
# Formula for transition probabilities (using splines)
formula <- ~ temp + s(slope, k = 8, bs = "ts")
hid2 <- MarkovChain$new(data = hmm_data, n_states = 2, formula = formula)

# Same observation model as before
obs2 <- Observation$new(data = hmm_data, dists = dists,
                        par = par0, n_states = 2)

# Create and fit model
hmm2 <- HMM$new(obs = obs2, hid = hid2)
hmm2$fit(silent = TRUE)
```

Covariate effects can be visualised with the `plot` method. The first argument indicates what part of the model should be plotted, and can be `tpm` (transition probabilities), `delta` (stationary state probabilities), or `obspar` (observation parameters). The covariates are included in the hidden process here, so we might want to visualise the first two.

```
hmm2$plot("tpm", var = "slope")
hmm2$plot("tpm", var = "temp")

hmm2$plot("delta", var = "slope")
hmm2$plot("delta", var = "temp")
```



The plots suggest there is no evidence that temperature affects transition probabilities. There seems to be a strong effect of slope, however: it is more likely to be in state 2 for intermediate values (between 10 and 30 degrees), and more likely to be in state 1 for extreme values (either smaller than 10 or larger than 30).

3 Position-based model

As an alternative to the model based on step lengths and turning angles, we turn to the location-based correlated random walk of Jonsen, Mills Flemming, and Myers (2005). If we denote as $\boldsymbol{Z}_t$ the location of the animal at time t (i.e., a vector of its x and y coordinate), we consider the model

$$\boldsymbol{Z}_t \mid \{S_t = j\} \sim N\left[\boldsymbol{Z}_{t-1} + \alpha_j(\boldsymbol{Z}_{t-1} - \boldsymbol{Z}_{t-2}),\ \sigma_j \boldsymbol{I}_2\right],$$

where $\boldsymbol{I}_2$ is the 2-by-2 identity matrix and, in state j, the parameter $\alpha_j \in \mathbb{R}$ measures the degree of autocorrelation in the location process, and $\sigma_j > 0$ measures variability. This model gives rise to correlated movement steps because, for $\alpha_j > 0$, the expected direction of change from $\boldsymbol{Z}_{t-1}$ to $\boldsymbol{Z}_t$ is aligned with the previous step from $\boldsymbol{Z}_{t-2}$ to $\boldsymbol{Z}_{t-1}$. If $\sigma_j = 0$, this reduces to a simple Gaussian random walk where each location is normally distributed around the previous location.

3.1 Data preparation

For each coordinate, we will need to include two covariates in the observation model: the previous location ($\boldsymbol{Z}_{t-1}$), and the previous step ($\boldsymbol{Z}_{t-1} - \boldsymbol{Z}_{t-2}$). We derive these two variables from the data, and we remove data rows where either is missing.

```
for(id in unique(hmm_data$ID)) {
  ind <- which(hmm_data$ID == id)
  hmm_data$x_lag[ind] <- c(NA, hmm_data$x[ind[-length(ind)]])
  hmm_data$y_lag[ind] <- c(NA, hmm_data$y[ind[-length(ind)]])
  hmm_data$x_diff[ind] <- c(NA, NA, diff(hmm_data$x[ind])[1:(length(ind) - 2)])
  hmm_data$y_diff[ind] <- c(NA, NA, diff(hmm_data$y[ind])[1:(length(ind) - 2)])
}

hmm_data <- na.omit(hmm_data)
```

3.2 Model specification

Similarly to the step-and-turn analysis, we use a 2-state model. We create the hidden state model with no special arguments, but we could include covariate effects in the transition probabilities like we did above.

```
hid3 <- MarkovChain$new(data = hmm_data, n_states = 2)
```

We define the observation model as follows:

- Each observed variable (i.e., each coordinate) follows a state-specific normal distribution.
- The mean of the normal distribution depends on the previous location and on the previous step. We remove the intercept from the R formula because there is no additive constant in the mean of the normal distribution. (If there was, this would create a systematic drift towards a preferred direction.)
- We define initial values for σ_1 and σ_2, shared between the two coordinates because we assume that the movement characteristics are the same along the two dimensions. The initial mean values are not used because there is no intercept.

```
dists <- list(x = "norm", y = "norm")
f <- list(x = list(mean = ~ x_lag + x_diff - 1),
          y = list(mean = ~ y_lag + y_diff - 1))
par0 <- list(x = list(mean = c(0, 0), sd = c(0.3, 3)),
             y = list(mean = c(0, 0), sd = c(0.3, 3)))
```

```
obs3 <- Observation$new(data = hmm_data, dists = dists, formulas = f,
                        n_states = 2, par = par0)
```

We need to add parameter constraints to ensure that the coefficient in from of the lagged covariate $\boldsymbol{Z}_{t-1}$ is not estimated (fixed to 1), and that α_j and σ_j have the same value for the two observed variables (i.e., same movement parameters along x and y dimensions). We update the parameters stored in `obs` to set the slope parameters for $\boldsymbol{Z}_{t-1}$ to 1. The function `obs3$coeff_fe()` can be used to remember what the order of the coefficients is. Then, we define a vector of `NA`s and integers that indicates which parameters are fixed (`NA`s), and which share a common value (integers).

```
obs3$update_coeff_fe(c(1, 0, 1, 0.6, log(0.3), log(3),
                       1, 0, 1, 0.6, log(0.3), log(3)))

fixpar_obs <- c(NA, 1, NA, 2, 3, 4,
                NA, 1, NA, 2, 3, 4)
names(fixpar_obs) <- rownames(obs3$coeff_fe())
fixpar <- list(obs3 = fixpar_obs)
```

3.3 Model fitting and results

We combine the hidden state and observation models, and pass `fixpar` to specify the parameter constraints, to create the `HMM` object.

```
hmm3 <- HMM$new(obs = obs3, hid = hid3, fixpar = fixpar)
hmm3$fit(silent = TRUE)
```

Although we can't plot the state-dependent distributions over the data like we did for step lengths and turning angles (because the distributions are covariate-dependent), we can extract the parameter estimates to interpret the states. The standard deviation parameters were $\sigma_1 = 0.74$ and $\sigma_2 = 3.18$, suggesting that there is more variability in state 2, perhaps associated with higher movement speed. The estimated persistence parameters were $\alpha_1 = -0.09$ and $\alpha_2 = 0.68$, indicating that there was virtually no persistence in state 1, and fairly strong persistence in state 2. We can also plot the trajectory, coloured by the Viterbi state sequence; from this, it is clear that state 1 captures slow clustered movement, and state 2 is fast and directed.

```
# Transition probability matrix
hid3$tpm()[,,1]
```

```
          state 1   state 2
state 1 0.8380761 0.1619239
state 2 0.1286120 0.8713880
```

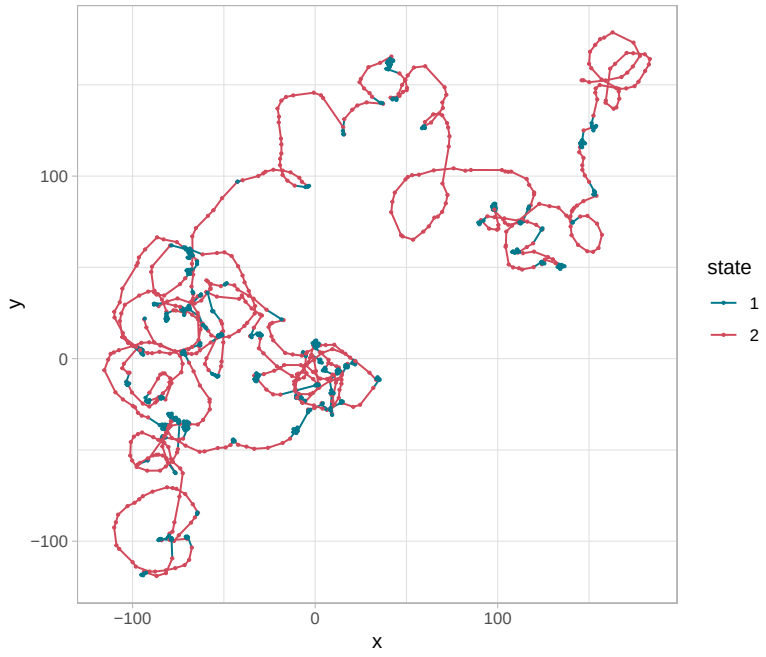
```
# Standard deviations
obs3$par()[2,,1]
```

```
  state 1   state 2
0.7436789 3.1846082
```

```
# Persistence parameters
obs3$coeff_fe()[c(2,4),]
```

```
x.mean.state1.x_diff x.mean.state2.x_diff
         -0.08890926           0.67853769
```

```
# Tracks coloured by most likely state sequence
hmm3$plot_ts("x", "y") +
  coord_equal() +
  geom_point(size = 0.4)
```



References

Johnson, Devin S, Joshua M London, Mary-Anne Lea, and John W Durban. 2008. "Continuous-Time Correlated Random Walk Model for Animal Telemetry Data." *Ecology*

89 (5): 1208–15.

Jonsen, Ian D, Joanna Mills Flemming, and Ransom A Myers. 2005. "Robust State–Space Modeling of Animal Movement Data." *Ecology* 86 (11): 2874–80.

Jonsen, Ian D, Patterson Toby A, Daniel P Costa, Philip D. Doherty, Brendan J. Godley, W. James Grecian, Christophe Guinet, et al. 2020. "A Continuous-Time State-Space Model for Rapid Quality-Control of Argos Locations from Animal-Borne Tags." *Movement Ecology* 8: 31. https://doi.org/10.1186/s40462-020-00217-7.

Langrock, Roland, Ruth King, Jason Matthiopoulos, Len Thomas, Daniel Fortin, and Juan M Morales. 2012. "Flexible and Practical Modeling of Animal Telemetry Data: Hidden Markov Models and Extensions." *Ecology* 93 (11): 2336–42.

McClintock, Brett T, and Théo Michelot. 2018. "momentuHMM: R Package for Generalized Hidden Markov Models of Animal Movement." *Methods in Ecology and Evolution* 9 (6): 1518–30.

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." *Methods in Ecology and Evolution* 7 (11): 1308–15.

Morales, Juan Manuel, Daniel T Haydon, Jacqui Frair, Kent E Holsinger, and John M Fryxell. 2004. "Extracting More Out of Relocation Data: Building Movement Models as Mixtures of Random Walks." *Ecology* 85 (9): 2436–45.

Patterson, Toby A, Marinelle Basson, Mark V Bravington, and John S Gunn. 2009. "Classifying Movement Behaviour in Relation to Environmental Conditions Using Hidden Markov Models." *Journal of Animal Ecology* 78 (6): 1113–23.