HMM Summary

Camrin Braun and Ben Galuardi

October 11, 2016

The Approach

We present a process-based approach to estimate residency and behavior from uncertain and temporally correlated movement data collected with electronic tags. To do this, we use a state-space model (SSM) for which the animal tracking problem is governed by two parts. The system process describes the animal movement and behavior, and the observation model links the process (i.e. movements) to the data. Inference regarding the unobservable system process can then be established via this link using statistical methodology (filtering) which updates location and behavior estimates with observed data. The system process estimation problem is formulated as a hidden Markov model (HMM) on a spatial grid in continuous time. Using the grid to explicitly resolve space, location estimation can be supplemented by or based entirely on environmental data (e.g. temperature).

Model formulation

Note that much of this content is taken directly from Pedersen et al 2011:

Since animals change their movement patterns through time, it is necessary to regard the system as a hierarchy of two sub-processes: An underlying behavior process that controls the switching between a number of different movement states; and a derived process that describes the movement dynamics conditional on the behavioral state. Formally we model the behavior process as a continuous-time Markov chain, I_t, with a finite state-space, $I_t \epsilon \{1, 2, ..., n\}$, where t denotes time. State switching of the behavior process is described by the generator matrix, $\mathrm{G^b}$ (superscript b for behavior), which contains the switching rates, λ_{ij}, of jumping from behavior state i to behavior state j.

The movement of the animal in continuous time is a (biased) brownian motion in the longitudinal (x) and latitudinal (y) direction. Given the current behavior state I_t of the animal, the Brownian motion is decribed by a drift $u_{I_t} = (u_x, u_y)_{I_t}$ with unit km day^{-1} and a diffusivity matrix D_{I_t} with unit km^2 day^{-1}.

To proceed with the analysis of the joint process of movement and behavioral shifts, we introduce the probability density $\phi_\mathrm{i}(x,y,t)$ which describes the probability that the animal at time t is located at (x, y) and is in behavioral state i. This can be formulated as a partial differential equation (PDE) describing the time evolution of ϕ_i as an advection-diffusion equation that includes behavior switching:

$$\frac{d\phi_i}{dt} = -\nabla \times (u_i \phi_i - D_i \nabla_{\phi_i}) + \sum_j \lambda_{ji}\ \phi_j$$

where ∇ is the two-dimensional spatial gradient operator. The advection and diffusion terms (in parentheses) describe the flow of probability between regions in space. The behavior switching (summation term) is a weighted sum over the switching rates that represents probability of flow between behavioral states. Together, the n coupled PDEs in Eq. 1 describe the underlying dynamics (movement and behavior) of the system in continuous time and continuous space.

To solve Eq. 1 some form of numerical approximation is required. Our approach discretizes the continuous spatial state–space into a finite, albeit large, number of uniformly sized squares (states) (Thygesen et al. 2009). On a discrete state-space, ϕ is no longer a probability density, but is instead represented by a vector j containing the state probabilities ϕ_α, where the state index $\alpha = (x, y, i)$ is composed of location in x and y and the behavior state i. The discretized state–space allows us to derive the generator matrices, G_i^m (superscript m for movement), related to the movement processes $i \epsilon \{1, 2, ..., n\}$.

Observations are assumed to be generated through a function h of the true animal location x_t and a random perturbation or error w_t which is related to the uncertainty of the measurement process. Formally the observation equation is written

$$z_t = h(t, x_t, w_t)$$

where z_t is a vector containing the observations available at t. Note that the behavior state i is not part of the observation equation and is therefore fully hidden. This formulation does not require h to be linear and there are no restrictions on the form of the distribution of w_t. The non-linear form of h allows for more subtle relations between observations and state, e.g. linking observations of daylight intensity to location (Nielsen et al. 2006). For marine animals, the lack of constraints on h is particularly useful as observations are rarely of location itself but rather of light intensity, depth, temperature etc. and so h becomes strongly nonlinear (Pedersen 2008).

With tracking data we have observations at N points in time, i.e. t_k is the time point of the kth observation and the set of observations available by this time is $Z_k = z_{t1}, ..., z_{tk}$. For a given time interval length we can compute the probability transition matrices P_k of the combined behavior and movement process using the generator matrices G_i^m and G^b.

Parameter Estimation

Maximum likelihood

A hidden Markov model (HMM) filter (Zucchini and MacDonald 2009) provides the probability distribution of the states forward in time conditional on data, $\phi(t_k|Z_k)$ (hereafter termed 'state estimates'). State estimates are calculated successively by alternating between so-called time and data updates of the current state. Time updates predict the state at the next time given the current state, while data updates use the next observation available to correct the time updates. Similar recursions are well known from other algorithms such as the Kalman filter or particle filter (Andersen et al. 2007) and are generally referred to as filtering recursions. In addition to the state estimates, the filter returns a likelihood measure which indicates how well the model fits the data. Thus, the likelihood function, L, of the unknown parameters θ (drift, diffusion, switching rates) can be evaluated at, say, θ_0 by running the filter using the parameter values in θ_0 (Thygesen et al. 2009).

Maximum likelihood (ML) estimation of model parameters is then straight forward:

$$\hat{\theta} = arg\,max\, L\left(\theta|Z_N\right)$$

This optimisation problem can be solved by most standard numerical optimizers. In a Bayesian context it is common to introduce a priori information about the parameters through the prior density $\pi(\theta)$. The maximum a posteriori (MAP) estimate of the parameters is therefore the value of θ which maximizes the posterior density $L(\theta|Z_N)\pi(\theta)$. In practice, however, substantial prior information is rarely available (Jonsen et al. 2005) in which case the MAP and the ML estimates are close to identical. Many animal tracking studies only consider ML parameter estimation using gradient-based maximization. However, recent work by Woillez et al (2016) further exploited the properties of the discrete setting and addressed a joint ML estimation of all model parameters using an iterative Expectation-Maximization (EM) framework.

Expectation maximization

The great interest of the EM algorithm is that it delivers a two-step iterative algorithm: the E-step computes the posterior distribution $P((X_t)_{t=0:N-1}|(Y_t)_{t=0:N-1})$ given current model parameter estimates; and the M-step updates model parameters according to a ML criterion reweighted by the posterior distribution. The EM algorithm guarantees to increase the likelihood after each EM iteration and as such it can be regarded as a gradient-based procedure. Woillez et al (2016) took the EM one step further because of sensitivity to initial

values. Thus, they considered a stochastic version of the EM where it replaces the numerical evaluation of the posterior by a sampling step. Overall, the SEM procedure involves two steps: [1] the E-step comes to sampling N_{SEM} trajectories $(X^{(i)})_{i=1:N_{SEM}}$ from posterior $P(X|Y, \Theta^{(v)})$ using the standard forward-backward HMM procedure, where $\Theta^{(v)}$ refers to model parameter(s) and could include diffusion, advection, observation error, etc - the M-step comes to updating estimate $\Theta^{(v+1)}$

This two-step procedure is iterated until some convergence threshold (change over some number of iterations).

Smoothing

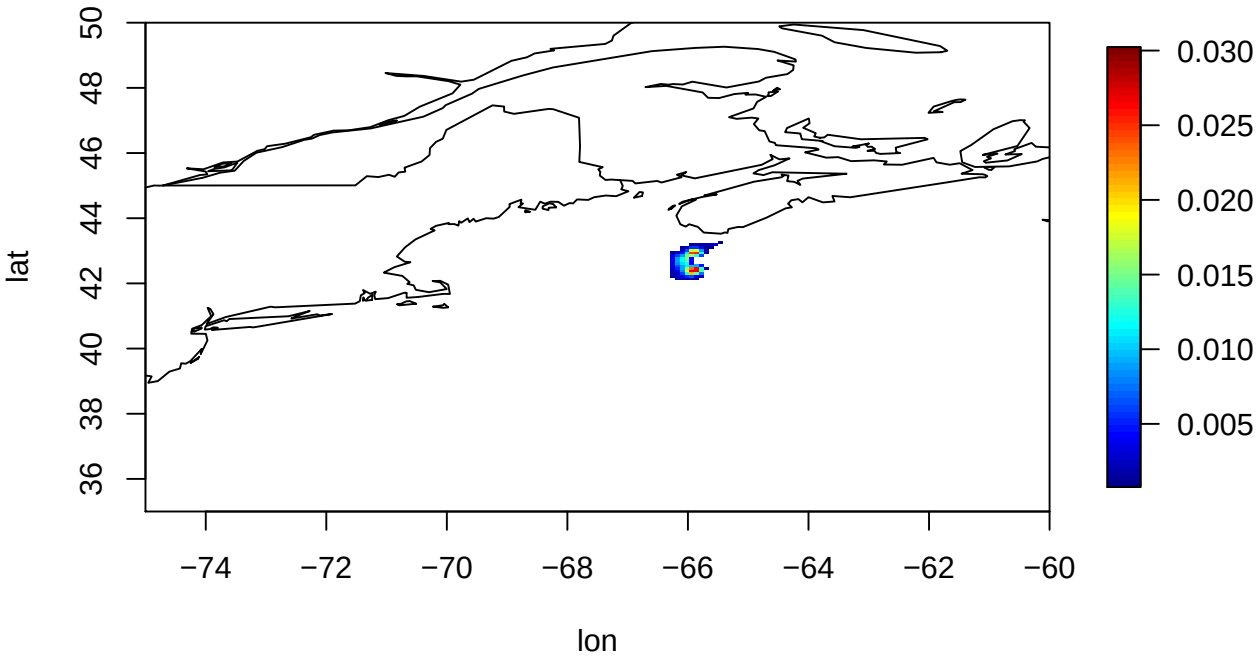
When parameters have been estimated only one step remains which is the so-called HMM smoothing step (Thygesen et al. 2009). The recursions of the smoothing step work backwards in time using the filtered state estimates and all available data to determine the smoothed state estimates, $\phi(t_k|Z_N)$. The smoothed state estimates are more accurate and generally appear 'smoother' than the filtering estimates because they exploit the full data set (Z_N). When fitting an SSM in a Bayesian context the smoothing step provides the posterior distribution of the state. By posterior distribution we mean the probability distribution of all states at all times given all data. Obviously, this distribution has a high dimension and is quite complex. For post-processing purposes it is therefore common to use time marginals of the posterior distribution (i.e. probability distributions of all states at specific times) which, in fact, are the state estimates returned from the HMM smoothing algorithm.

Implementing the EM

From Pedersen et al 2011: *When fitting an SSM in a Bayesian context the smoothing step provides the posterior distribution of the state... it is therefore common to use time marginals of the posterior distribution (i.e. probability distributions of all states at specific times) which, in fact, are the state estimates returned from the HMM smoothing algorithm.*

For us, this is the output of `hmm.smooth` that we call *s* of dim(states, time, x, y). In a classic EM approach (non-stochastic) the E-step is simply the computation of the posterior distribution given current model parameter estimates which we've already done in *s*.

```
##  num [1:2, 1:135, 1:539, 1:564] 0 0 0 0 0 0 0 0 0 0 ...
```



So the E-step in a traditional EM sense should be complete. If we want to go the stochastic route as in Woillez then we need to finish the E-step with sampling of N trajectories (which I honestly don't really understand but seems like it would be straightforward).

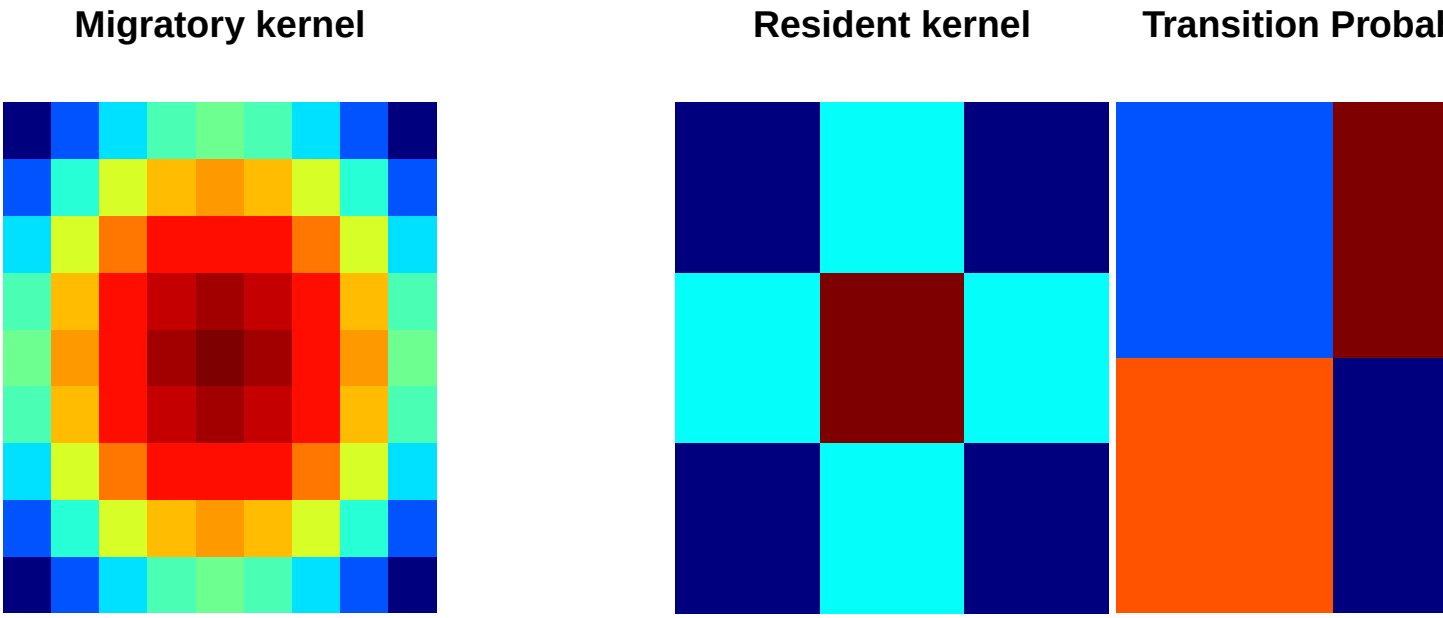
The M-step "updates model parameters according to a ML criterion reweighted by the posterior distribution."(from Woillez) I don't understand how this works either and, if my assumptions are correct, this is where we need the help from Andy? We need to update D and P(switch) before re-running kernels, filter and smoother. Something like:

```
# GENERATE MOVEMENT KERNELS. D VALUES ARE MEAN AND SD PIXELS
K1 = as.cimg(gausskern(D1[1], D1[2], muadv = 0))
K2 = as.cimg(gausskern(D2[1], D2[2], muadv = 0))
P <- matrix(c(p[1],1-p[1],1-p[2],p[2]),2,2,byrow=TRUE)

# RUN THE FILTER STEP
f = hmm.filter(g,L,K1,K2,P)

# RUN THE SMOOTHING STEP
s = hmm.smoother(f, K1, K2, P, plot = F)

# UPDATE D


# UPDATE P
# get states from s
sv <- apply(s[1,,,], 1, sum) > apply(s[2,,,], 1, sum)
sv <- -sv + 2

# difference it to find transitions
svd = rbind(sv, c(0,diff(sv)))

# get a new transition probability matrix
r1 = rev(table(svd[2, svd[1,]==1])/sum(svd[1,]==1))
r2 = rev(table(svd[2, svd[1,]==2])/sum(svd[1,]==2))
P = matrix(rbind(r1, r2), 2,2)

# CALCULATE CHANGE IN PARAMETER VALUES OVER LAST SEVERAL ITERATIONS
# Woillez used "ratio betwen the average over the last 20 values of D and the new value of D below 1%"


# RE-ITERATE
# now have a new set of diffusion parameters (D1, D2) and switching probability (P) and we can iterate
```

I anticipate the code above would eventually end up within a function like `get.nll.fun`, right?

The Questions

The process portion of the HMM is represented by diffusive kernels *K1* and *K2* based on diffusion parameters in *D*. These parameters are of differing bandwidth and extent estimates for 'migratory' and 'resident' states (Fig. 1a,b).



To run an expectation Maximization process, we need to be able to calculate input parameters based on our posterior output in an iterative fashion. Our inputs are K_1, K_2 and P_1, representing our resident, migratory kernels and transition probability matrix (Fig. 1).

Calculating a new P_n is straightforward as outlined above. What we need help with is determining kernel parameters based on our output. In other words, given a posterior probability grid, how do we calculate kernel bandwidth and extent? All literature so far has pointed to the standard process of determing a kernel based on bandwidth etc., not the other way around.

In our functions, the parameters are siz and $Sigma$ representing the extent in each dimension and bandwidth, respectively.

We use kernels to represent the process portion of the HMM that we parameterize with our best estimates of movement parameters like diffusion D. Remember we're also considering differential movement parameters based on an animal's behavior state so kernels are quite different between 'migratory' and 'resident' states (Fig. 1) and we need a robust way to estimate probability of switching between states.