Approximate Methods for State-Space Models

Koyama et al. (2010)

STAT 548: supervised by Daniel J. Mcdonald,

Fanny Dupont.

2022

1 Introduction

State-Space Models (SSMs) are an important tool in ecology, commonly used to analyze time-series. They provide a versatile framework to perform on a wide range of data (e.g, capture/re-capture, location data, abundance data). Their flexible structure allows for many extensions, including taking into account structuring factors (environmental covariates, type of habitat), physiological processes (e.g,feedback mechanisms Lu et al. (2015)) and intra-specific interactions (Langrock et al. (2014)). Inferring such models can lead to computational issues. Among them, estimating the predictive distribution of the state since it usually consists of computing an intractable integral. This issue is addressed by Koyama et al. (2010) who derive an approximation procedure to estimate it. Recent contributions using particle filtering methods (Andrieu et al. (2010)) have also offered an alternative solution. In this report, we write the Laplace approximation method from Koyama et al. (2010) in R and test it with animal movement data. First, we summarize their work and then implement their method and test it. In a third part, we provide some possible extensions for the procedure.

2 Notation

Here are the notations we will use throughout this paper.

- X_t is the hidden state of the SSM at time t,
- $X_{1:T}$ is the sequence of hidden states from time = 1 to time = T,

- Y_t is the random observation of the SSM at time t,
- $Y_{1:T}$ is the sequence of random observations from time = 1 to time = T,
- $y_{1:T}$ is the data from time = 1 to time = T.
- θ is the vector of parameters of the SSM.
- $p(x_t|y_{1:t-1})$ is the predictive density function of the state.
- $\hat{p}(x_t|y_{1:t-1})$ is the approximation of the predictive density function of the state.

3 Summary

This section consists of a summary of the paper "Approximate Methods for State-Space Models" written by Koyama et al. (2010), which focuses on the use of Laplace Approximation to estimate the predictive density function of the state given the observation in a State-Space Model (SSM). This statistical problem is called *filtering*. In a first subsection, we will define and explain the SSM framework to be able to cover the Laplace Approximation and its application by Koyama et al. (2010) in a second subsection.

3.1 SSM

State-Space Models (SSMs) are flexible statistical models where the observation is an incomplete estimate of the true underlying (hidden) state. The hidden sequence of states $X_{1:T}$ evolves over time and emits a sequence of observations $Y_{1:T}$. Usually, the functions that define the relationships between states and observations have an unknown parameter $\theta \in \mathbb{R}^m$. An SSM is defined as follow:

$$\begin{cases} X_1 & \sim \quad q_\theta(X_1) \\ X_t & \sim \quad f(X_t|X_{t-1}, \theta), \quad t > 1, \\ Y_t & \sim \quad g(Y_t|X_t, \theta), \qquad t > 0. \end{cases} \tag{1}$$

q_θ is the initial density i.e the distribution of the first state, f the transition density that is the probability of the current state given the previous state. Finally, g corresponds to the density of observing the current observation given the hidden state. Usually, different parts of θ affect g and f such that we can have the following equivalent model:

$$\begin{cases} X_1 & \sim & q_{\theta_1}(X_1) \\ X_t & \sim & f(X_t|X_{t-1},\theta_1), \quad t>1, \\ Y_t & \sim & g(Y_t|X_t,\theta_2), \qquad t>0, \end{cases} \tag{2}$$

where $\theta = (\theta_1, \theta_2)$. Note that $\theta_1 \in \mathbb{R}^{n_1}$ and $\theta_2 \in \mathbb{R}^{n_2}$ with $n_1 + n_2 = m$.

3.2 Filtering

In this context, Koyama et al. (2010) address the problem of filtering, that is given a state process $\{x_t\}_{t=1,\dots,T}$ and a related observation process

$$\{y_t\}_{t=1,\dots,T},$$

estimating the state x_t given the sequence of observations $y_{1:t}$. This is equivalent to finding the posterior density $p(x_t|y_{1:t})$ of the state, given the sequence of observation. There exists a recursive solution to the filtering problem for state-space models. It is defined as follow:

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t}, \tag{3}$$

$$\text{where } p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}. \tag{4}$$

Equation (3) comes from the Bayes's rule. This set of equations is useful for relatively simple models (e.g linear). However, when the dynamics are nonlinear or high dimensional, these integrals can be intractable and computing them becomes challenging. The Laplace approximation is a convenient tool in these cases.

The general concept is that under certain conditions (reasonably well-behaved and unimodal), the marginal density function (here $p(x_t|y_{1:t-1})$) can be approximated by a normal distribution. The Laplace approximation mostly consists of a Taylor series approximation. Note that we usually define the behaviour of a density function by the concentration of its mass: it is well-behaved if it is concentrated in a small area of its domain. For example, the functions of the Lebesgue space of second order are adapted to this method.

First-order Approximation: This method is motivated by the computation that follows. We will use the first-order Laplace approximation to derive $p_\theta(x_t|y_{1:t})$ and use the same notation as Koyama et al. (2010) i.e

$$\bar{x}_{tt} = \arg\max_{x_t} L_\theta(x_t|y_{1:t}) := \arg\max_{x_t} \log(p_\theta(y_t|x_t)p_\theta(x_t|y_{1:t-1})).$$

We proceed as follow:

$$\begin{aligned}
\textstyle\int_{\mathbb{R}} x_t p_\theta(x_t|y_{1:t}) &= \frac{\int_{\mathbb{R}} x_t \overbrace{p_\theta(y_t|x_t)p_\theta(x_t|y_{1:t-1})}^{\exp[h_\theta(x_t)]} dx_t}{\int_{\mathbb{R}} p_\theta(y_t|x_t)p_\theta(x_t|y_{1:t-1})dx_t}, \\
&= \frac{\int_{\mathbb{R}} x_t \exp[h_\theta(x_t)]dx_t}{\int_{\mathbb{R}} \exp[h_\theta(x_t)]dx_t}, \\
&\approx \frac{\int_{\mathbb{R}} x_t \exp[h_\theta(\bar{x}_{tt}) + \frac{1}{2}h''_\theta(\bar{x}_{tt})(x_t - \bar{x}_{tt})^2]dx_t}{\int_{\mathbb{R}} \exp[h_\theta(\bar{x}_{tt}) + \frac{1}{2}h''_\theta(\bar{x}_{tt})(x_t - \bar{x}_{tt})^2]dx_t}, \\
&\approx \frac{\int_{\mathbb{R}} x_t \exp[\frac{1}{2}h''_\theta(\bar{x}_{tt})(x_t - \bar{x}_{tt})^2]dx_t}{\int_{\mathbb{R}} \exp[\frac{1}{2}h''_\theta(\bar{x}_{tt})(x_t - \bar{x}_{tt})^2]dx_t}, \\
&\approx \frac{\int_{\mathbb{R}} \cancel{\sqrt{\frac{2\pi}{-h''_\theta(\bar{x}_{tt})}}} x_t \psi(x_t|\bar{x}_{tt}, -[h''_\theta(\bar{x}_{tt})]^{-1})dx_t}{\int_{\mathbb{R}} \cancel{\sqrt{\frac{2\pi}{-h''_\theta(\bar{x}_{tt})}}} \psi(x_t|\bar{x}_{tt}, -[h''_\theta(\bar{x}_{tt})]^{-1})dx_t}, \\
&\approx \int_{\mathbb{R}} x_t \exp[\frac{1}{2}h''_\theta(\bar{x}_{tt})(x_t - \bar{x}_{tt})^2]dx_t.
\end{aligned}$$

Here, $\psi(\cdot|\mu, \sigma^2)$ corresponds to the gaussian density with mean μ and variance σ. The cancelled terms in the second to last line are because h''_θ is evaluated at $\bar{x}_{tt}$ which does not depend on x_t. We can thus take it out of the integral, both on the numerator and denominator and they cancel each other. The last line is due to the fact that ψ is a density function, integrated over its entire domain, which equals 1. Hence, as described by Koyama et al. (2010), $p_\theta(x_t|y_{1:t})$ can be approximated by a gaussian density with mean $\bar{x}_{tt}$ and variance $-[h''_\theta(\bar{x}_{tt})]^{-1}$. Observe that this approximation works only if the likelihood $L_\theta(x_t|y_{1:t})$ is unimodal. This computation explains the motivation behind the first-order Laplace approximation.

Second-order Approximation: We redefine $\bar{x}_{tt}$ as the maximum of

$$\log g(x_t)p(y_t|x_t)\hat{p}(x_t|y_{1:t1}),$$

for any positive function g, and keep the same variance. To estimate a multidimensional state $x_t \in \mathbb{R}^d$, we take g to be one coordinate of x_t at a time, i.e $g(x_t) = x_{t,i}$ for $i = 1 \ldots d$. Hence to obtain the estimate in d dimensions, the process has to be done d times, with g taken to be each coordinate in turn.

Now, assuming that θ is known, the Laplace Approximation of both order offers a relatively simple procedure to obtain the predictive distribution of the state that we summarize below for the one-dimensional case:

1. At time $t = 1$, the predictive distribution of the state is initialized to $q_\theta(x_1)$, chosen by the statistician.

2. Observe y_t (i.e the data at time t).

3. (Filtering) Obtain the approximate posterior mean $\bar{x}_{tt}$ and variance v_{tt} using the Laplace approximation (order chosen by the statistician), and set $\hat{p}(x_t|y_{1:t})$ to be a Gaussian distribution with mean $\bar{x}_{tt}$ and variance v_{tt}.

4. (Prediction) Calculate the predictive distribution, using Equation (4) i.e

$$\hat{p}(x_{t+1}|y_{1:t}) = \int p(x_{t+1}|x_t)\hat{p}(x_t|y_{1:t})dx_t,$$

5. Increment t and go to Step 2.

Koyama et al. (2010) call this procedure the Laplace Gaussian Filter (LGF). It can be of multiple order, depending on which order for the Laplace approximation is used.

Instead of directly approximating the predictive distribution, the authors introduce the function h, with $h(x_t) = \frac{1}{\gamma}p(y_t|x_t)p(x_t|y_{1:t-1})$ where γ is a scaling parameter which estimation depends on the model. γ is used to quantify both the error and stability of the procedure. Koyama et al. (2010) detailed some conditions (there are five of them) under which the procedure is valid. Among them, the unimodality of the function that we already mentioned, the state distribution $s \in \mathcal{C}^4$, and the existence of the integral defined in Equation (4). Note that a fairly good knowledge of the system is also required to ensure the validity of this procedure since it requires that θ is known and that $\hat{p}(x_t|y_{1:t})$ is a "well-behaved" function.
Under these conditions, the Laplace procedure 3.2 has some useful properties, among them the fact that both its accuracy and stability correspond to $O(\gamma^\beta)$, where $\beta = 1$ for $\alpha = 1$ and $\beta = 2$ for $\alpha \geq 2$.

Even though the LGF seems attractive, it is not the only procedure available. As we previously said, particle filtering is another method that has been commonly used (Auger-Méthé et al. (2021), Doucet et al. (2001), Andrieu et al. (2010)). Koyama et al. (2010) compare them in two different frameworks: simulated and real. In both cases, the Mean Integrated Squared Error (MISE) tables show that the Laplace Approximation of second order is the most accurate, followed by LGF-1. Both are performing better than the particle filtering. In the real data analysis, the authors compare both methods performance in estimating the hand motion from neural activity. In this context, they found the Laplace Approximation to be much more accurate than the particle filter with the same computational cost.

4 Project

4.1 Description of the project

The methodology introduced by Koyama et al. (2010) has only been tested in the context of neural decoding. Thus the goal of our project is to implement it in a more generalizable framework and software. We will evaluate its performance by estimating the latent states from a non-linear state-space model based on polar bear movement data.

4.2 Implementation

Here is the pseudo-code of our implementation of the Laplace Approximation function that takes 8 inputs. The implementation has been written in R.

```
1: Procedure Laplace Approximation
Inputs: Obs (observations), $l_1$ (initial log-likelihood), dim (state dimension), s (state probability density), M (state-space matrix with the extrema of each state), $likeY$ (density function of $Y$ knowing $X$), $A$ and $B$ (constraints for the maximization algorithm: matrices).
2: Initialize the distributions, $p \leftarrow$ list(),
#$\hat{p}$ is the approximation of the predictive probability density
3:     for $i = 1 : T$ do
4:         Maximize log likelihood and extract estimates
5:         Compute the hessian of the inverse of the log likelihood evaluated at these estimates.
6:         Approximate the predictive density probability $\hat{p}[[i]]$ according to eq (3)
7:         Update observation
8:         Update log likelihood as a product of $likeY(obs[t,], x_t) * \hat{p}[[i]](x_t|obs[1:t,])$
9:     end for
10: Return $\hat{p}$
11: end procedure
```

Line 6 of the procedure is done using the Get-Predictive-function which pseudo-code is described below.

```
1: **Procedure** Get-Predictive-function
   *Inputs: $\mu$ (approximated mode), $\Sigma$ (approximated covariance matrix), dim (state dimension), s (state probability density), M (state-space matrix with the extrema of each state)*
2:    Approximate the integrand from Equation (4), with $p(x_{t-1}|y_{1:t-1})$ approximated by a multivariate Gaussian with mean $\mu$ and covariance matrix $\Sigma$.
4:    Compute the integral with the *hcubature* function.
5: return(integral)
6: **end procedure**
```

The maximization function used in the procedure is "maxLik" and the derivatives are taken with the "numDeriv" package. Autodifferentiation methods have been considered but no package in R offers such a computation. You can see the complete code in the appendix A.

4.3 Data and SSM

The dataset we are using consists of the location data (i.e., latitude and longitude) of one adult female polar bear that was collared in the Beaufort Sea, Northwest Territories, Canada, in April of 2009. We have access to its GPS and ARGOS locations that are available every day at the same time (around 5pm), throughout the year. The GPS locations are extremely accurate ($\leq$ 30m) compared to Argos data, which can have errors as large as 36 km depending on the quality class.

We considered the ARGOS data to be an incomplete measure of the real movement of the polar bear represented here by the GPS location data. The true movement of the animal at time t is denoted x_t and is typically described using two data streams: step length $l_t \in (0, \infty)$ (distance between consecutive locations) and turning angle $\phi_t \in (-\pi, \pi]$ (change in direction between consecutive steps), where $t \in \{1, \dots, T\}$ represents the time of the step. We set the distribution of the step length to be a gamma distribution, with parameters that do not depend on time (in order to simplify their estimation). Note that modelling the step-length with parameters that vary through time and depend on environmental covariates (McClintock and Michelot (2018)) is closer to reality. We assumed that l_t is distributed as follows:

$$l_t \sim \text{gamma}(\alpha, \beta),$$

$\alpha > 0$ is the shape and $\beta > 0$ the scale of the gamma distribution. The probability of turning angle ϕ_t is often modelled with von Mises distribution (Duchesne et al. (2015)). We assumed that ϕ_t followed a von Mises distribution with the previous turning angle as mean:

$$\phi_t \sim \text{vMises}(\phi_{t-1}, \kappa_1),$$

where κ_1 is the concentration parameter of the von Mises distribution. We can summarize the true state densities as follows:

$$\begin{cases} f(l_t) & = & \dfrac{l_t^{\alpha-1} e^{-\beta l_t} \beta^\alpha}{\Gamma(\alpha)} \\ g_{VMc}(\phi_t) & = & \dfrac{1}{2\pi I_0(\kappa_1)} e^{\kappa_1 \cos(\phi_t - \phi_{t-1})}, \ \phi_t \in (-\pi, \pi], \end{cases}$$

for $t \in \{1, \ldots, T\}$. Note that ϕ_t is often modelled with a concentration parameter depending on the step-length at time t and a mean depending on the previous turning angle ϕ_{t-1} (Duchesne et al. (2015)). This dependence is called the directional persistence. It accounts for short-term effect. In more complex models, the turning angle consists of a compromise between the natural tendency to follow the same direction as the previous step (i.e *directional persistence*) and movement in response to preferred patches and to the target in direction ψ_t (Duchesne et al. (2015)). Finally, the ARGOS data i.e the observation y was modelled as follow:

$$y_t \sim \mathcal{N}(x_t, \Sigma),$$

where $x_t = (\phi_t, l_t)$ and $\Sigma = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$. We decided to take Σ to be the identity matrix times two since the ARGOS locations can have large error.

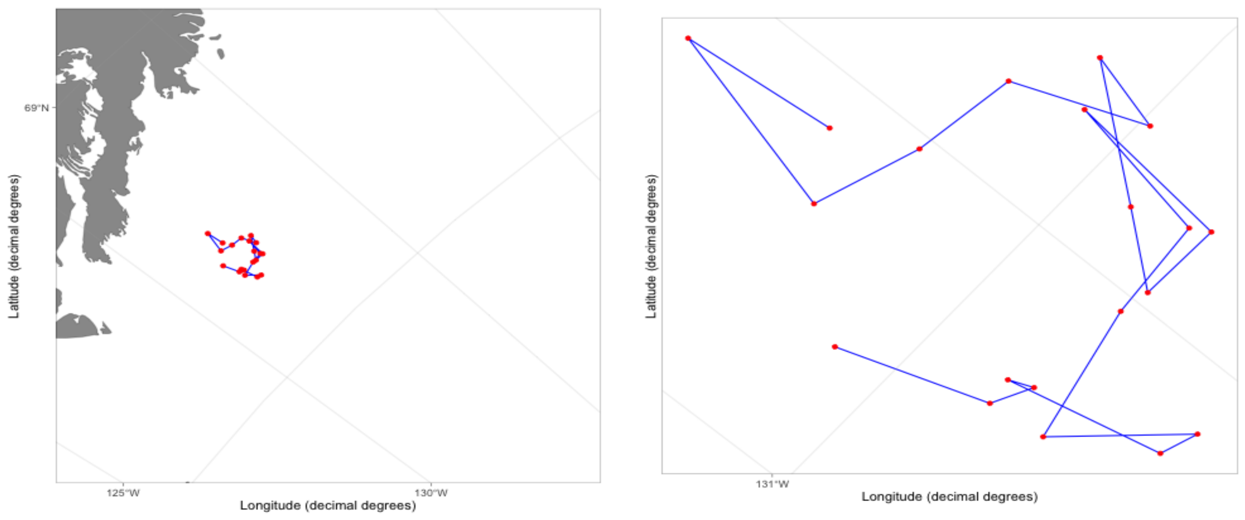
69°N
Latitude (decimal degrees)
125°W
130°W
Longitude (decimal degrees)
Latitude (decimal degrees)
131°W
Longitude (decimal degrees)

Figure 1: Map of the GPS location data (true states) of the polar bear for the first 20 steps of the dataset.

The locations are taken from the 20th of April 2009 until the 9th of May 2009. On both polar maps, the water (and ice) is in white and land is grey. On the map to the left, the land is a north-west point of Canada, which closest city is Tuktoyaktuk. The starting point is the non-connected point located the most south and the ending point is the non-connected one further north.

We estimated the parameters of the model using MLE and method of moments on the complete GPS dataset. To estimate the concentration parameter κ_1, we did maximum likelihood on the difference of the turning angles $\{\phi_t - \phi_{t-1}\}_{t \in \{2,\ldots,T}$, with $\phi_1 \sim \text{vMises}(0, \kappa_1)$. The joint density of the differences from time 2 to T and ϕ_1 is the product of the densities. Thus we could perform maximum likelihood on our track data (GPS data). To estimate the scale and shape parameters of the gamma distribution, we used the method of moments. Once the parameters were estimated, we ran the Laplace procedure on the observation taken to be the ARGOS data and estimated our predictive distribution $\hat{p}_{\hat{\theta}}(x_t|\phi_{1:t})$. Here $x_t = (\phi_t, l_t)$, the turning angle and the step length at time t. $\phi_{1:t}$ corresponds to the ARGOS locations from time one to time t and $\hat{\theta}$ is the vector of the estimated parameters$(\hat{\alpha}, \hat{\beta}, \hat{\kappa_1})$.

4.4 Results

Since the state $x_t = (\phi_t, l_t)$ has two dimensions, the computation of the integrals (using the function cubature in R) required some time which is why we limited our analysis to the first 20 times. Note that the Laplace procedure was not long to run ($\sim$ 5 min) but the computation of the expectation (integrals over 2 dimensions) required over a day to compute. Once the predictive density distribution was estimated, we indeed computed the expectation of the states to compare it with the real states.

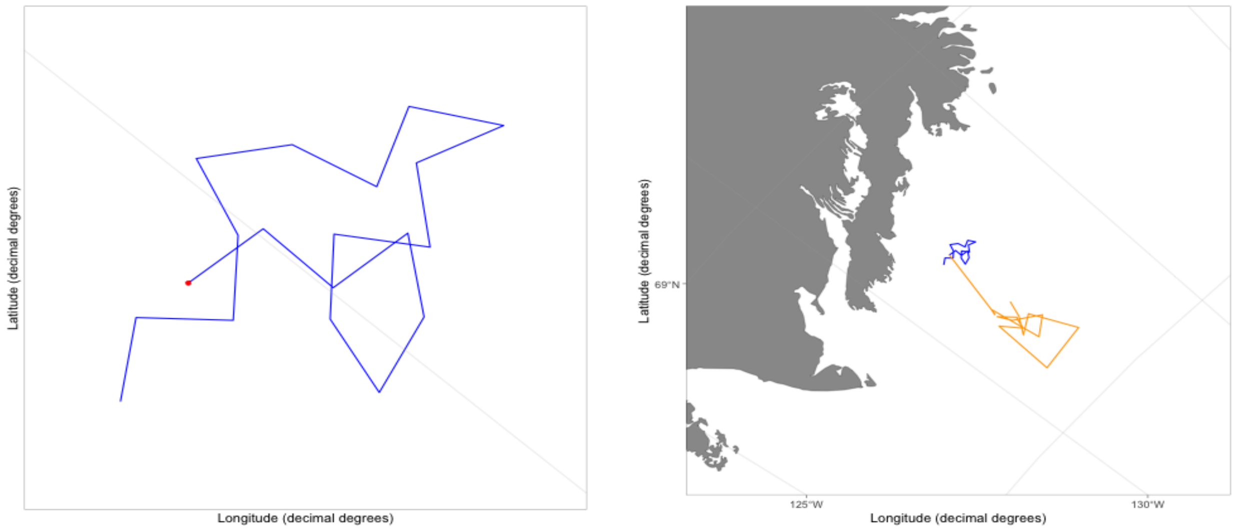
Latitude (decimal degrees)
Longitude (decimal degrees)
Latitude (decimal degrees)
69°N
125°W
130°W
Longitude (decimal degrees)

Figure 2: Polar map of the ARGOS location data (orange) of the polar bear for the first 20 steps of the dataset and with the results obtained from Laplace approximation (blue).

Both tracks have the same starting point. We can see on Fig. 2 that the predicted states are more concentrated than the observations. Note that since the step-length has parameters that do not depend on time, its expectation is the same at every time ($\sim$ 7 km), which agrees with the model we defined. We then compares our results to the real movement data, see Fig. 3 below.

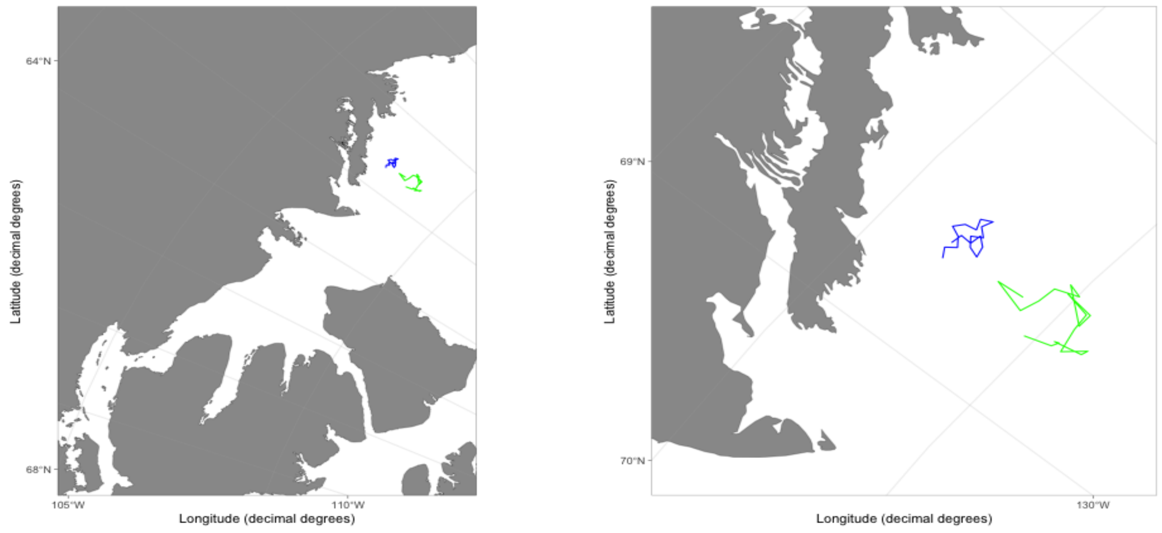
64°N
68°N
105°W
110°W
Latitude (decimal degrees)
Longitude (decimal degrees)
69°N
70°N
130°W
Latitude (decimal degrees)
Longitude (decimal degrees)

Figure 3: Polar map of the true GPS location data (green) of the polar bear for the first 20 steps of the dataset and with the results obtained from Laplace Approximation (blue).

The tracks are again pretty different and the results from our procedure are more concentrated than the true states.

We computed the mean squared error to assess the performance of our prediction. The MSE of the latitude is approximately 2.25 and for the longitude 1.15 which is pretty large if we are interested in the precise movement of the animal. We checked for any correlation in the MSE vector using the ACF and the autocorrelation was not significant so the MSE seems to be a good metric to investigate the performance of that model. Note that since the distribution of l_t and the observation associated with it does not depend on t (Σ and the gamma parameters are constant) thus we could have implemented a uni-dimensional Laplace approximation. We decided to implement the two-dimensional version to check for the algorithm precision and assess its computational complexity. The estimated expectation for the true step-length after the Laplace approximation was $\approx$ 7km whereas the mean of the gamma distribution with the estimated MLE parameters was 14.6 km. This loss of information can not be ignored. It can be due to the fact that we reduced the grid of the integrals in order to obtain a faster computation.

5 Discussion

The Laplace procedure is a good tool to estimate the predictive density of the state. It requires to properly define a model and to know its parameters. In our case, we chose to work with a simple model, where the step-length of the animal was taken to have parameters that do not depend on time. Thus the expectation of the states was constant which is not realistic for modeling animal movement. For example when studying polar bears, Togunov et al. (2017) divided their movement in multiple behavioural states, all of them associated with a different average step-length. In the olfactory search state, polar bears tend to make big steps while they let themselves drift on the sea-ice when they are resting. Thus it is not realistic to model the step-length with constant parameters. Moreover, we only had access to one GPS track data, limiting the performance of our estimation procedure.

6 Suggestions

A way to improve our prediction would be to use a Bayesian framework to estimate the parameters of our model since we have a small sample size.

1. A first idea would be to write an MCMC algorithm based on the format of pMCMC developped by Andrieu et al. (2010) with Laplace approximation introduced by Koyama et al. (2010) instead of particle filtering.
2. A second idea is to use pMCMC directly Andrieu et al. (2010)

The next subsection corresponds to a small development of the first idea since the second one is not an original idea.

6.1 Bayesian Framework

As we previously said, there are many ways to estimate the parameters of our model. The Bayesian framework is better suited for small sample size than maximum likelihood. Here, since we only have one polar bear, the performance of our estimation was limited. Our idea is based on the framework of pMCMC developped by Andrieu et al. (2010). According to Koyama et al. (2010), the Laplace approximation is more accurate than the particle filtering for the same time complexity. Thus the pMCMC might have a better performance if the particle filtering step is replaced by a Laplace approximation step. Here, we suggest another procedure that might reduce the time complexity of pMCMC.

We propose an algorithm following the idea of pMCMC, to compute $p(\theta, x_t|y_{1:t})$. Every step is the same, except for the filtering (estimation of the predictive distribution) step.

We describe the method below: At every iteration:

1. Propose a candidate $\theta^\star \sim q(\cdot|\theta)$, where θ is the previous sample (or on the prior for the first iteration). q is generally taken to be a gaussian distribution.

2. Use Laplace approximation (4.2) to obtain $p_{\theta^\star}(x_{1:T}|y_{1:T})$ (replace x_t by $x_{1:T}$ in the procedure described above) and to generate N samples from this distribution.

3. Select one of these samples: it becomes the proposed sequence of states from time 1 to T.

4. Now we obtain an estimate of the likelihood of the data given the parameters: $\hat{L}(y_{1:T}|x_{1:T}, \theta^\star)$.

5. Accepts the joint candidate sample with probability $\min(1, a)$ where the acceptance ratio a is:

$$\frac{\rho(\theta^\star)}{\rho(\theta)} \frac{\hat{L}(y_{1:T}|x_{1:T}, \theta^\star) q(\theta|\theta^\star)}{\hat{L}(y_{1:T}|x_{1:T}, \theta) q(\theta^\star|\theta)}.$$

The idea would be to code it by hand in R. As we previously said, we expect the time complexity of the pMCMC to be reduced since the particle filtering step is replaced by a Laplace approximation that has less time-complexity (Koyama et al. (2010)).

7 Conclusion

Koyama et al. (2010) defined an efficient way to compute the predictive density function of the states. However, the time complexity of the procedure increases with the dimension of the state. Thus it might not be a relevant method for a multidimensional state. The Laplace approximation is yet a good candidate to improve the pMCMC algorithm for a one-dimensional state. We suggested the implementation of a new procedure: incorporating the Laplace approximation in the pMCMC algorithm. This could reduce the time-complexity of the algorithm. However, this new method might be limited to a unidimensional state, since integrating over multidimensional is expensive.

References

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3): 269–342, 2010.

Marie Auger-Méthé, Ken Newman, Diana Cole, Fanny Empacher, Rowenna Gryba, Aaron A King, Vianey Leos-Barajas, Joanna Mills Flemming, Anders Nielsen, Giovanni Petris, et al. A guide to state–space modeling of ecological time series. *Ecological Monographs*, 91(4):e01470, 2021.

Arnaud Doucet, Nando De Freitas, Neil James Gordon, et al. *Sequential Monte Carlo methods in practice*, volume 1. Springer, 2001.

Thierry Duchesne, Daniel Fortin, and Louis-Paul Rivest. Equivalence between step selection functions and biased correlated random walks for statistical inference on animal movement. *PloS one*, 10(4):e0122947, 2015.

Shinsuke Koyama, Lucia Castellanos Pérez-Bolde, Cosma Rohilla Shalizi, and Robert E Kass. Approximate methods for state-space models. *Journal of the American Statistical Association*, 105(489):170–180, 2010.

Roland Langrock, J Grant C Hopcraft, Paul G Blackwell, Victoria Goodall, Ruth King, Mu Niu, Toby A Patterson, Martin W Pedersen, Anna Skarin, and Robert S Schick. Modelling group dynamic animal movement. *Methods in Ecology and Evolution*, 5(2):190–199, 2014.

Ji Lu, Junhao Pan, Qiang Zhang, Laurette Dube, and Edward H Ip. Reciprocal markov modeling of feedback mechanisms between emotion and dietary choice using experience-sampling data. *Multivariate behavioral research*, 50(6):584–599, 2015.

Brett T McClintock and Théo Michelot. momentuhmm: R package for generalized hidden markov models of animal movement. *Methods in Ecology and Evolution*, 9(6):1518–1530, 2018.

Ron R Togunov, Andrew E Derocher, and Nicholas J Lunn. Windscapes and olfactory foraging in a large carnivore. *Scientific Reports*, 7(1):1–10, 2017.

A CodeAppendix

Below is the code for the Laplace Approximation procedure that returns the predictive density at each time-step.

```
Laplace_Approximation <- function(obs,log_like1,dim,state_distribution,state_space,
LikeY,A,B){

#Initialize
log_like <- log_like1
T <- length(obs[,1])
grad_log_like<-function(x){
numDeriv::grad(log_like,x,method="Richardson")
}
x <- obs
x[1,]<-obs[1,]
hessian_log_like <- function(x){
numDeriv::hessian(log_like,x,method="Richardson")}
HatPPredict <- list()

#Compute
  for(t in 1:T){
    #Compute
    MaxParam <- maxLik(log_like,grad=grad_log_like,start=x[t,],
    constraints=list(ineqA=A, ineqB=B))
    result_max <-list(x_estimate =c(MaxParam$estimate),Log_Like_Value=c(MaxParam$maximum))
    cov <- solve(-hessian_log_like(result_max$x_estimate))
    if(det(cov)<0){
      cat("Iteration ", t, "\n", sep = "",
      red("The covariance matrix is not positive-definite"))
      break()
    }
    approx <- list(mean =c(result_max$x_estimate), Sigma = cov)
    f <- Get_Predictive_function(approx$mean,approx$Sigma,dim,state_distribution,state_space)
    HatPPredict<- append(HatPPredict,f)

    #Incrementation
    Predict<-HatPPredict[[t]]
    log_like <- function(xt) log(LikeY(obs[t,],xt)*Predict(xt))
    grad_log_like<-function(x){
      numDeriv::grad(log_like,x,method="Richardson")
    }
    hessian_log_like <- function(x){
      numDeriv::hessian(log_like,x,method="Richardson")}
    cat("Iteration ", t, "\n", sep = "")

  }

  return(HatPPredict)
}
```

obs is a dataframe of observations. It can have multiple columns is the states are multidimensional. log-like1 is the initial log-likelihood. LikeY is the density of y_t knowing x_t State-Space is a matrix with 2 columns, where each line is the boundary for the first element of the state. state-distribution is a function of x_{t+1} and x_t. First argument is x_{t+1} and second x_t If $\dim(x_t) = 2$ state-distribution has the following form: State-distribution $(x_1t + 1, x_2t + 1, x_1t, x_2t)$ A and B are for if the maximization requires constraints.

Below is the code for the Get-Predictive-function.

```
Get_Predictive_function<- function(mu,Sigma,dim,state_distribution,state_space){
  if (dim == 1){
    hat_p <- function(x){
      dnorm(x,mean=mu,sd=Sigma)
    }
    f <- function(x,y){
      state_distribution(y,x)*hat_p(x)
    }

    hat_p_predict <- function(y){
      g <- function(x){
        f(x,y)
      }
      return(integrate(g,state_space[1],state_space[2]))
    }
  }
  else{
    if(dim == 2){
      hat_p_predict <- function(y){

        gg<- function(x){
          if(abs(det(Sigma))>10000){
            scal <- 0
          }else{
            scal <- 1/(2*pi*det(Sigma)^{1/2})
          }
          d <- matrix(c(x[1],x[2])-mu,  ncol=2,nrow=1)
          term <-exp(-0.5*d%*%solve(Sigma)%*%t(d))
          scal*term
          #d <- matrix(c(x1,x2),  ncol=2,nrow=1)
          return(scal*term*state_distribution(y[1],y[2],x[1],x[2]))
        }
        return(hcubature(gg,lowerLimit = c(state_space[,1]),
        upperLimit=c(state_space[,2]))$integral)

      }
    }else{
      if(dim == 3){
        hat_p_predict <- function(y){
          gg<- function(x){
            d <- matrix(c(x[1],x[2],x[3])-mu,  ncol=2,nrow=1)
            term <-exp(-0.5*d%*%solve(Sigma)%*%t(d))
```

```
              scal*term
              #d <- matrix(c(x1,x2),  ncol=2,nrow=1)
              return(scal*term*state_distribution(y[1],y[2],y[3],x[1],x[2],x[3]))
            }
            return(hcubature(gg,lowerLimit = c(state_space[,1]),
            upperLimit=c(state_space[,2]))$integral)

          }
        }

      }
    }



    return(hat_p_predict)

  }
```