PARAMETER ESTIMATION USING MARKOV CHAIN MONTE CARLO METHODS

Sameen Islam
University of Southampton
England, United Kingdom
si1u19@soton.ac.uk

ABSTRACT

In the real-world, it is often not possible to make a direct measurement of an underlying phenomena. In such cases, we observe some effect and measure its change over time. However, this kind of measurement is prone to drift which can cause fatal errors. Parameter estimation can help us measure the underlying state of a system through which we can eliminate drift. In this paper we explore Monte Carlo methods of state estimation, also known as Particle filters, which does not make simplifying assumptions about the real world, like the Kalman filter does. We also consider the Extended Kalman filter, for estimating the parameters of a Logistic regression model for binary classification.

1 PRELIMINARIES

In state estimation, we have the process and the observation model where we use observations made to estimate the state which cannot be directly measured. The process model is defined as $\mathbf{x}_{n+1} = f_n(\mathbf{X}_n, \mathbf{w}_n)$ and the observation model is defined as $\mathbf{y}_n = h_n(\mathbf{x}_n, \mathbf{v}_n)$ where $\mathbf{w}_n$ and $\mathbf{v}_n$ is process and measurement noise.

More formally, our goal is to optimally estimate the state of the system $\mathbf{x}_n$ at some time n, given a sequence of observations $\{\mathbf{y}_n\}_0^{N-1}$. Our estimate of the state at time n will be based on some set of observations from the first moment $n = 0$ to an observation at time $n \geq 0$. Filters are able to recursively make this estimate without scaling computation time as more data arrives over time.

A Markov process is defined as a random process where $p(\mathbf{y}_n|\mathbf{x}_{0:n}) = p(\mathbf{y}_n|\mathbf{x}_n)$, meaning the state at time n carries all information up to that point and no additional information is gained by looking at states further back in time.

Under the Bayesian framework, the prediction step and filter step are as follows:

$$p\left(\mathbf{x}_n \mid \mathbf{y}_{0:n-1}\right) = \int p\left(\mathbf{x}_n \mid \mathbf{x}_{n-1}\right) p\left(\mathbf{x}_{n-1} \mid \mathbf{y}_{0:n-1}\right) d\mathbf{x}_{n-1} \tag{1}$$

$$p(\mathbf{x}_n|\mathbf{y}_{0:n}) = \frac{p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n)}{\int_{\mathbf{x}} p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n)} \tag{2}$$

however, these have no analytical solution other than the case of Kalman filter, where the PDF is a Gaussian, parameterised by mean and covariance. This integral can be approximated using Monte-Carlo methods, such as Sequential Importance Sampling (SIS) as we show below.

Monte Carlo methods numerically compute an integral by random sampling, as computing posterior $p(\mathbf{x}_n|\mathbf{y}_{0:n})$ is analytically intractable from Naesseth et al. (2019). We approximate (2) with:

$$p(\mathbf{x}_n|\mathbf{y}_{0:n}) \approx \sum_{\ell=0}^{L} \tilde{w}_n^{(\ell)} \cdot \delta(\mathbf{x}_n) \tag{3}$$

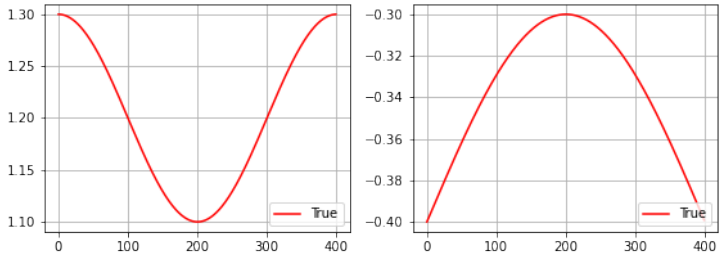


Figure 1: AR parameters α_0, α_1 slowly changing over time.

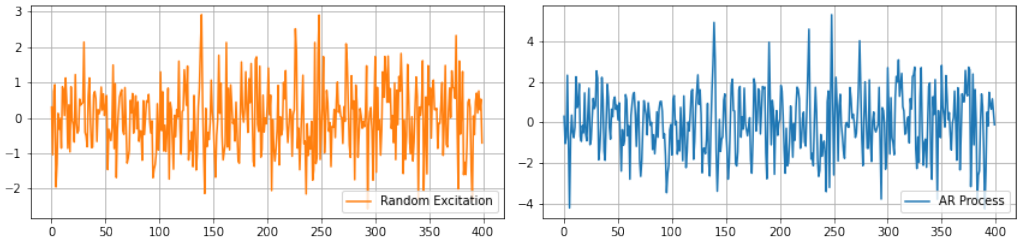


Figure 2: $AR(2)$ process noise and output.

Importance samples are obtained by picking a proposal distribution $q(\mathbf{x}_n|\mathbf{x}_{0:n-1}, \mathbf{y}_{0:n})$ from which we can sample, this is the Normal distribution in our case. In SIS, samples are weighted by:

$$w_n^\ell = \tilde{w}_{n-1}^\ell \frac{p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{x}_{n-1})}{q(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{y}_n)} \tag{4}$$

where $p(\mathbf{x}_n|\mathbf{x}_{n-1})$ is the prior PDF, which can be obtained from the process model. $p(\mathbf{y}_n|\mathbf{x}_n)$ is the likelihood obtained from the measurement model.

We will apply the Kalman filter and Sequential Importance Sampling (SIS) to estimate the parameters of the AR model shown in figure 2. Whereas with the Kalman filter (KF) we assume a linear state space model, a particle filter like SIS allows us to deal with cases where the state space model is non-linear.

To estimate the AR model parameters a_k where $k \in 1, 2$ we take $\boldsymbol{\theta}$ to be its estimate. As we assume the AR process is making a random walk, we add Gaussian noise $\mathbf{w}$ to the estimate. As we formulate the problem as a linear state space model, we assume the observations made from the AR signal y is the result of some process caused by an underlying linear function with some process noise v, noting that y is a scalar.

$$\begin{aligned} \boldsymbol{\theta}(n) &= \boldsymbol{\theta}(n-1) + \mathbf{w}(n) \\ y(n) &= \boldsymbol{\theta}^T \boldsymbol{x}_n + v(n) \end{aligned} \tag{5}$$

Under the state space model framework, we observe y the output of $AR(2)$ process and estimate $\boldsymbol{\theta}$ with KF, SIS and SIR. Since KF is the optimal estimator due to its assumption about Gaussian noise and linear process model, we anticipate KF to provide the best estimate of a_k.

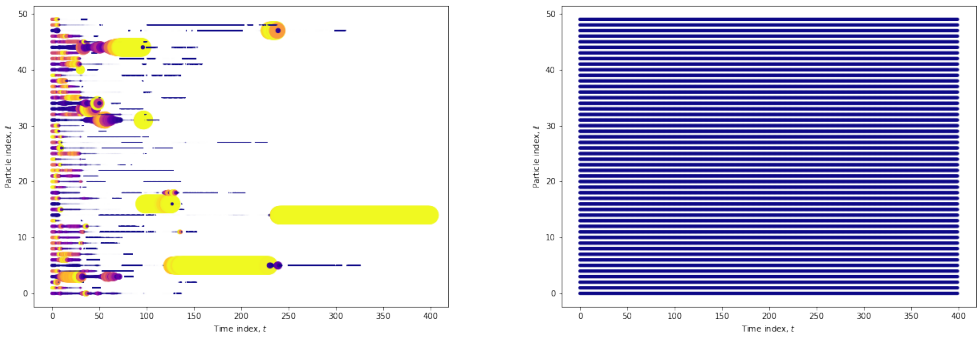
In SIS, when small weights are multiplied together, it results in even smaller weights. This continues over time until only a single particle has a very large weight. This condition is detected using the effective sample size at time n

$$ESS_n \triangleq \frac{1}{\sum_{\ell=1}^{L} \left(\tilde{w}_n^{(\ell)}\right)^2} \tag{6}$$

the value of ESS_n is in the range $[1, L]$. When $ESS_n = 1$ the degeneracy is at its strongest as only a single particle has a large weight and all other particles have a zero weight.

2 RESULTS

Applying SIS with no re-sampling causes weight degeneracy (figure 3a). This is the effect when after some time steps, only one particle has non-negligible weight. Since this implies that we approximate our target distribution with a single importance particle, it is not likely to be accurate. Furthermore, it is possible for the large weight to be in a region of target distribution with low probability, such that the importance weight can take a large number of time steps to reduce.



(a) Sequential Importance Sampling (SIS) (b) Sequential Importance re-sampling (SIR)

Figure 3: SIS suffering from weight degeneracy indicated by only a single particle with high weight remaining after some time passes (yellow). SIR immune to weight degeneracy indicated by uniform color and particle size.

We apply the ESS to quantitatively explore the strength of the weight degeneracy, which as figure 4 shows, is severe. This is because we find that after a brief number of time steps, $ESS = 1$ for most particles.

The effect of weight degeneracy can be countered with the addition of a re-sampling step (figure 3b). Particles are sampled with replacement from the set of all particles with a probability that depends on the importance weights, which propagates particles with large weights and cancels out smaller ones. Overall, the re-sampling step gives us a better approximation of target distribution $p(y)$ but at the cost of greater variance in estimates.

We now compare the performance of the SIR particle filter to the Kalman filter. Recall that unlike the Kalman filter, particle filters do not assume a Gaussian state space and a linear process model.

Figure 5 compares the tracking performance of Kalman filter against the two variants of particle filters: SIS and SIR. We find that in the case of tracking both AR coefficients, Kalman filter maintains an optimal estimate. SIS loses track of parameter values as they change over time due to path

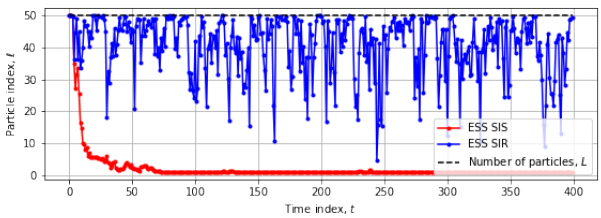


Figure 4: Effective Sample Size used as a measure of weight degeneracy. Within $t = 100$, $ESS_{SIS} = 1$ indicating all but one particle have weight 0. With re-sampling, $1 < ESS_{SIR} \leq L$ and weight degeneracy is avoided.

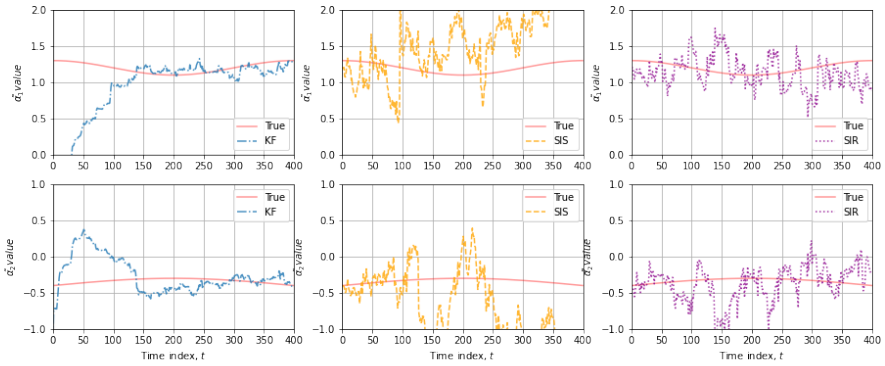


Figure 5: Estimating $AR(2)$ coefficients (red) α_1 (top row) and α_2 (bottom row) using Kalman filter (left column) and particle filters SIS and SIR (middle and right column). In this scenario, KF is the optimal estimator, SIS loses track of signal due to weight degeneracy and SIR provides sub-optimal estimates.

degeneracy. SIR maintains a good estimate, but upon closer inspection, we find that its estimates have greater variance as expected, compared to the Kalman filter.

We now move on to the problem of estimating the parameters of a logistic regression model performing binary classification using the extended Kalman filter (EKF).

The state space model for sequential estimation of $\boldsymbol{\theta}$ is given by the non-linear process model

$$\begin{aligned} \boldsymbol{\theta}(n) &= \boldsymbol{\theta}(n-1) + \boldsymbol{w}(n) \\ y(n) &= f\left(\boldsymbol{\theta}, \boldsymbol{x}_n\right) + v(n), \end{aligned} \tag{7}$$

where, $f\left(\boldsymbol{\theta}, \boldsymbol{x}_n\right)$ is the logistic regression model

$$\begin{aligned} f\left(\boldsymbol{\theta}, \boldsymbol{x}_n\right) &= g\left(\boldsymbol{\theta}(n)^T \boldsymbol{x}_n\right) \\ g(z) &= \frac{1}{1+e^{-x}} \end{aligned} \tag{8}$$

When using EKF, we linearise our non-linear process model $f\left(\boldsymbol{\theta}, \boldsymbol{x}_n\right)$ by taking the Jacobian $\hat{\boldsymbol{H}}$ which is a first-order Taylor series approximation from Arulampalam et al. (2007). We denote the sigmoid function of our logistic regression as $\sigma(x) = \dfrac{1}{1+e^{-x}}$ and differentiate w.r.t. $\sigma(x)$.

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{(1+e^{-x})^2} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right) \\
&= \sigma(x) \cdot (1-\sigma(x))
\end{aligned}
\tag{9}
$$

With this result, we may now write the Jacobian as

$$
\hat{\boldsymbol{H}} = g\left(\boldsymbol{\theta}(n \mid n-1)^T \boldsymbol{x}_n\right)\left(1 - g\left(\boldsymbol{\theta}(n \mid n-1)^T \boldsymbol{x}_n\right)\right)\boldsymbol{x}_n \tag{10}
$$

With the linearised approximation, we now apply the recursive Kalman predict-update equations from Niranjan (2008).

Algorithm 1: Extended Kalman Filter for Logistic Regression

Initialise: $\{\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\theta_0}, \boldsymbol{P_0}, \boldsymbol{Q}, \boldsymbol{R}\}$
Output: $\{\boldsymbol{\theta}(n|n)\}$
for $n = 1, \cdots, N$ **do**
- $\hat{\boldsymbol{H}} = g\left(\boldsymbol{\theta}(n \mid n-1)^T \boldsymbol{x}_n\right)\left(1 - g\left(\boldsymbol{\theta}(n \mid n-1)^T \boldsymbol{x}_n\right)\right)\boldsymbol{x}_n$
- $\boldsymbol{\theta}(n|n-1) = \boldsymbol{\theta}(n-1|n-1)$
- $\boldsymbol{P}(n|n-1) = \boldsymbol{Q} + \boldsymbol{P}(n-1|n-1)$
- $\hat{y} = \boldsymbol{\theta}(n|n-1)^T \boldsymbol{x}_n$
- $e(n) = y(n) - \hat{y}$
- $k(n) = \frac{\boldsymbol{P}(n|n-1)\cdot\hat{\boldsymbol{H}}^T}{R+\hat{\boldsymbol{H}}\cdot\boldsymbol{P}(n|n-1)\cdot\hat{\boldsymbol{H}}^T}$
- $\boldsymbol{\theta}(n|n) = \boldsymbol{\theta}(n|n-1) + k(n)\cdot e(n)$
- $\boldsymbol{P}(n|n) = (\boldsymbol{I} - k(n)\cdot\hat{\boldsymbol{H}})\cdot\boldsymbol{P}(n|n-1)$
- $\boldsymbol{\theta}(n-1|n-1) = \boldsymbol{\theta}(n|n)$
- $\boldsymbol{P}(n-1|n-1) = \boldsymbol{P}(n|n)$

end

Our data is generated from a two-class problem in two dimension with mean of class 0 set to $[-\alpha, \alpha]$ and mean of class 1 set to $[\alpha, -\alpha]$. The covariance is common to both classes and is set to $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$.

We illustrate two cases: one with small α where classes overlap each other completely and another where a higher α provides a clear separation between classes. Our objective then, is to sequentially estimate the posterior boundary using EKF. Figure 6 shows the converged class boundary.

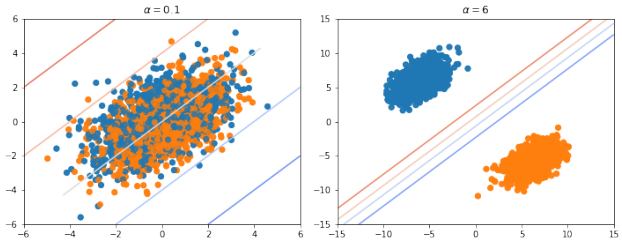


Figure 6: Converged estimate of Logistic regression posterior class boundary using Extended Kalman filter. When classes overlap, performance is poor. However, convergence is good with a clear separation.

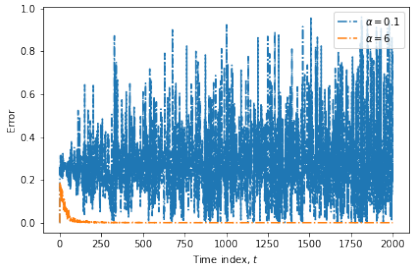


Figure 7: Extended Kalman filter prediction error for binary classification dataset when classes are separated at $\alpha = 0.1$ and $\alpha = 6$.

3 Conclusion

We have applied Sequential Importance Sampling (SIS) for a synthetic time varying AR problem and shown that it suffers from weight degeneracy. We have also extended SIS to include re-sampling (SIR) which we saw improved the tracking capability of SIR to be comparable to Kalman Filter. We have also presented an algorithm for estimating the parameters of a Logistic regression model using the Extended Kalman Filter and shown that class boundaries converge when datapoints from the two have clear separation.

References

M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/nongaussian bayesian tracking. *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking*, 50(2):723–737, 2007. doi: 10.1109/9780470544198.ch73.

Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. Elements of sequential Monte Carlo. *Foundations and Trends in Machine Learning*, 12(3):187–306, 2019. ISSN 19358245. doi: 10.1561/2200000074.

M. Niranjan. Sequential Bayesian computation of logistic regression models. 9:1065–1068 vol.2, 2008. doi: 10.1109/icassp.1999.759927.