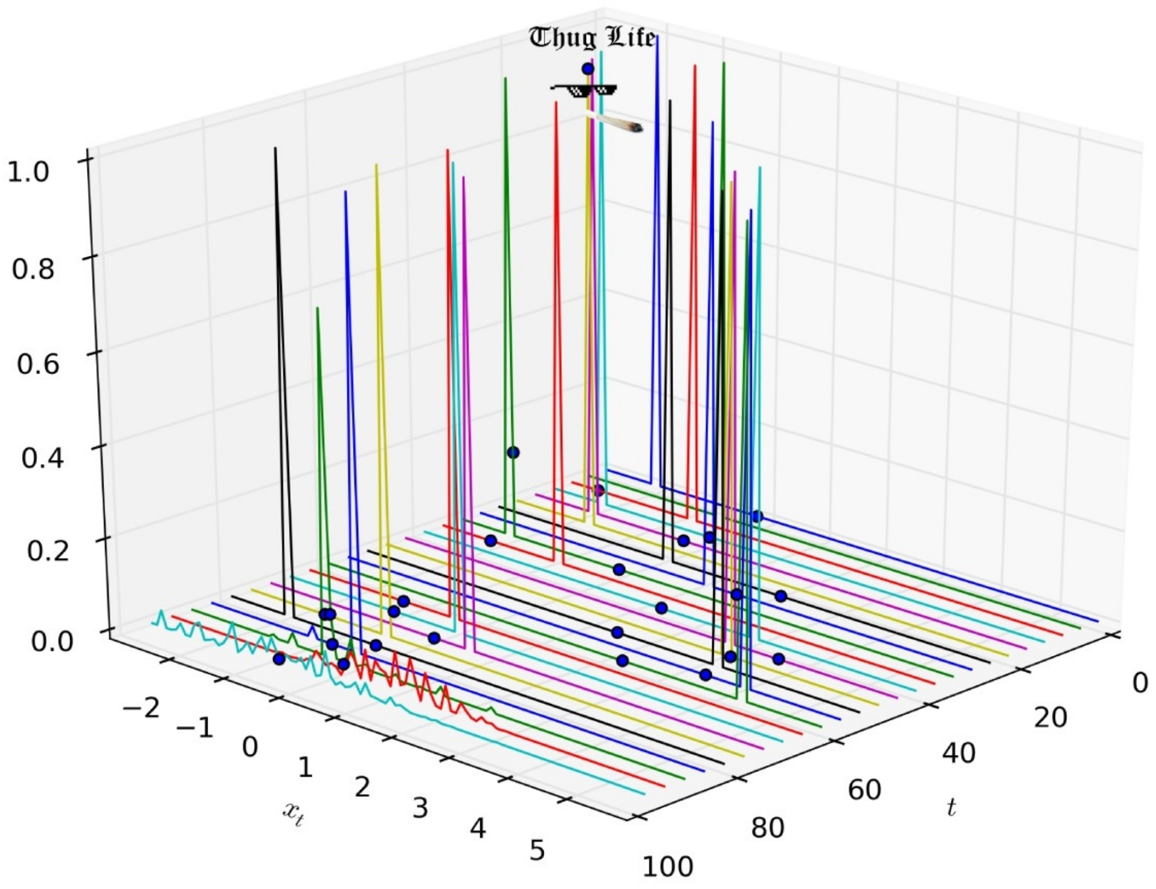
Reproduction of "A Tutorial on Particle Filtering and Smoothing: Fifteen years later"

DD2434 Machine Learning, Advanced Course

Leo Carlsson, Salma el Alaoui Talibi, Ludvig Ericson, Paulina Hensman, Hlynur Davíð Hlynsson

December 2015



1 Introduction and Overview

Particle filtering, or Sequential Monte Carlo (SMC) methods, are a popular means of estimating non-Gaussian non-linear state-space models. They have enjoyed widespread use in fields as diverse as chemical engineering, computer vision and financial econometrics. The particle filtering methods are an especially good fit for real-time systems where constant-time approximation is preferred to more exact algorithms.

In this report, we aim to reproduce the results of the paper "A Tutorial on Particle Filtering and Smoothing: Fifteen years later" by Doucet and Johansen [1]. Their paper is a tutorial of the particle filtering methods commonly used as of 2008, introducing a generic particle filtering algorithm that we will implement and discuss in section 2.2, and continue to show that all other particle filtering methods can be interpreted as instances and variations of that basic algorithm. We will therefore implement these more advanced methods as special cases of the particle filtering algorithms and discuss them in the rest of section 2.

1.1 Particle Filtering

Particle filtering is, as the name implies, a method of solving the so-called *filtering problem* for hidden Markov models, namely estimating $\mathrm{p}(x_{1:T}|y_{1:T})$.

$$\begin{aligned}\mathrm{p}(x_{1:T}|y_{1:T}) &= \frac{\mathrm{p}(x_{1:T}, y_{1:T})}{\mathrm{p}(y_{1:T})} \\ &= \frac{\mathrm{p}(x_{1:T})\,\mathrm{p}(y_{1:T}|x_{1:T})}{\int \mathrm{p}(x_{1:T})\,\mathrm{p}(y_{1:T}|x_{1:T})\,\mathrm{d}x_{1:T}}\end{aligned} \tag{1}$$

For non-linear, non-Gaussian models, it is intractable to compute these distributions in closed-form. Particle filtering methods approximate the distributions $\{\mathrm{p}(x_{1:n}|y_{1:n})\}_{n\geq 1}$, the marginal likelihoods $\{\mathrm{p}(y_{1:n})\}_{n\geq 1}$ and the smoothing distributions $p(x_n|y_{1:T})$ for $n = 1 \ldots T$.

1.2 Sequential Monte Carlo methods

Particle filtering is an example of a sequential Monte Carlo method, in which the target distribution $\pi_n(x_{1:n})$ is estimated by drawing a sequence of samples from the distribution. In non-trivial cases, it's impossible to draw samples directly from the target distribution, and by eq. (1) we instead sample from the distribution $\gamma_n(x_{1:n})$,

$$\pi(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n}$$

where

$$Z_n = \int \gamma_n(x_{1:n}) dx_{1:n}$$

However, it is important to remember that the above expression is just one choice of distributions. There are many different choices of target distributions due to the often essential need to approximate the intractability of the expressions. This will be more prevalent when subsequent methods are shown.

Straightforwardly, $\pi(x_{1:n})$, can be approximated using basic Monte Carlo. However, two problems may still exist. The first problem is if $\pi(x_{1:n})$ is complex and high-dimensional, which makes it impossible to sample from. The second problem is that the computational complexity of $\pi(x_{1:n})$ increases linearly with n which may pose problems for benchmarking systems. These problems can be solved with importance sampling (IS) and sequential importance sampling (SIS), respectively[1]. These methods will be presented in section 2.1.

1.3 Stochastic Volatility

Stochastic volatility models are a class of models where the variance, or *volatility*, is itself regarded as a latent stochastic process.[2] We model this by letting the next state X_n in a process be given by the previous state X_{n-1} and some volatility variable V_n, that is to say

$$X_n = \alpha X_{n-1} + \sigma V_n,$$

and the emission Y_n is an exponential function of the state with some Wiener process W_n,

$$Y_n = \beta \exp(\frac{X_n}{2}) W_n.$$

In accordance with the original paper, we have $\mathcal{X} = \mathcal{Y} = \mathbb{R}$, $X_1 \sim \mathcal{N}\left(0, \frac{\sigma^2}{1-\alpha^2}\right)$, $V_n \overset{i.i.d}{\sim} \mathcal{N}(0,1)$ and $W_n \overset{i.i.d}{\sim} \mathcal{N}(0,1)$. This gives us

$$f(x'|x) = \mathcal{N}(x'; \alpha x, \sigma^2) \qquad \text{and} \qquad g(y|x) = \mathcal{N}(y; 0, \beta^2 \exp(x)). \tag{2}$$

Lastly μ is chosen to be equal to the marginal distribution of X_n,

$$\mu(x) = \mathcal{N}\left(0, \frac{\sigma^2}{1-\alpha^2}\right). \tag{3}$$

We use the same parameter values as in the original paper, i.e. $\alpha = 0.91$, $\sigma = 1.0$ and $\beta = 0.5$[1]. A sample simulation is shown in fig. 1, and shows that as the latent measure of volatility increases, the spread of the observations increases exponentially.

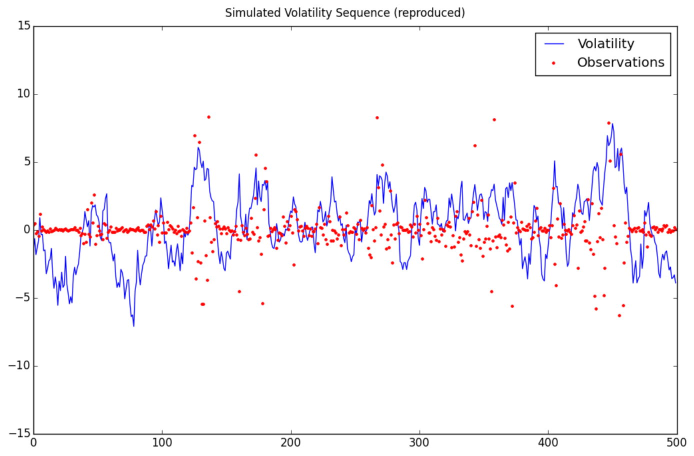


Figure 1: Simulation of the Stochastic Volatility model.

2 Methods

2.1 Sequential Importance Sampling and Resampling

IS introduces an importance density $q_n(x_{1:n})$ along with an unnormalised weight function $w_n(x_{1:n})$ such that

$$\gamma_n(x_{1:n}) = q_n(x_{1:n})w_n(x_{1:n}).$$

One then selects an importance function $q_n(x_{1:n})$ from which it is easy to sample, e.g. a multivariate Gaussian. The approximations to $\pi(x_{1:n})$ and Z_n are then calculated using a Monte Carlo approximation.

SIS builds upon IS but has a *sequential* importance structure

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1})$$

which makes it possible to recursively calculate the unnormalised weights through the decomposition

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})} = \frac{\gamma_{n-1}(x_{1:n-1})}{q_{n-1}(x_{1:n-1})}\frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1})}$$

which can be written, using the incremental importance weight, denoted α, as

$$w_{n-1}(x_{1:n-1})\alpha_n(x_{1:n}) = w_1(x_1)\prod_{k=2}^{n}\alpha_k(x_{1:k})$$

.

However, the problem with SIS is that the variance increases with n, often exponentially. This issue can partially be solved by resampling after the calculation of the importance weights. Various resampling techniques can be used, and we will first use multinomial resampling. It is the combination of SIS and resampling that creates the framework of SMC methods.

2.2 Sequential Importance Resampling

The simplest method is known as Sequential Importance Resampling. The SIR algorithm for the Stochastic Volatility (SV) model is presented below. We choose the conditional prior as a proposal distribution, i.e.:

$$q(x_n|x_{n-1}) = f(x_n|x_{n-1}) \tag{4}$$

This will be the case for all methods presented in this report. Using the state transition as a proposal distribution means that the weight does not depend on the past trajectory of the particles, but only on the likelihood of the observation $g(y_n|x_n)$. The use of this proposal distribution is popular because sampling is simple and computing the incremental weights amounts to estimating the conditional likelihood of the new observation given the updated particle position [3].

At time $n = 1$:

- For $i = 1 \dots N$, sample $X_1^i \sim q(x_1|y_1) = \mu(x_1)$
- For $i = 1 \dots N$, compute the unnormalized weight function:

$$w_1(X_1^i) = \frac{\mu(x_1)g(y_1|X_1^i)}{q(X_1^i|y_1)} = \frac{\mu(x_1)g(y_1|X_1^i)}{\mu(x_1)} = g(y_1|X_1^i) \tag{5}$$

  And then compute the normalized weights $W_1^i \propto w_1(X_1^i)$.

- Resample $\{W_1^i, X_1^i\}$ to obtain N equally weighted particles, i.e. $\{\frac{1}{N}, \bar{X}_1^i\}$.

At time $n \geq 2$:

- For $i = 1 \dots N$, sample $X_n^i \sim q(x_n | y_n, X_{n-1}^i) = f(x_n | X_{n-1}^i)$
- For $i = 1 \dots N$, compute the unnormalized weight function:

$$\alpha_n(X_{n-1:n}^i) = \frac{g(y_n|X_n^i) f(X_n^i|X_{n-1}^i)}{q(X_n^i|y_n, X_{n-1}^i)} = \frac{g(y_n|X_n^i) f(X_n^i|X_{n-1}^i)}{f(X_n^i|X_{n-1}^i)} = g(y_n|X_n^i) \qquad (6)$$

  And then compute the normalized weights $W_n^i \propto \alpha_n(X_{n-1:n}^i)$.

- Resample $\{W_n^i, X_{1:n}^i\}$ to obtain N equally weighted particles, i.e. $\{\frac{1}{N}, \bar{X}_{1:n}^i\}$.

To compare the performance of SMC algorithms with and without the resampling steps, we use the stochastic volatility model described in section 1.3, and run the SIS algorithm (corresponding to the above SIR algorithm without the resampling step) with $N = 1000$ particles, in accordance with the tutorial[1]. At each iteration n of the algorithm, the mean and standard deviation of $x_n | y_{1:n}$ is shown in fig. 2. It can be seen that it performs reasonably well initially, but in the later time steps, the estimated mean becomes inaccurate, and the estimated standard deviation becomes inaccurately low. The failure

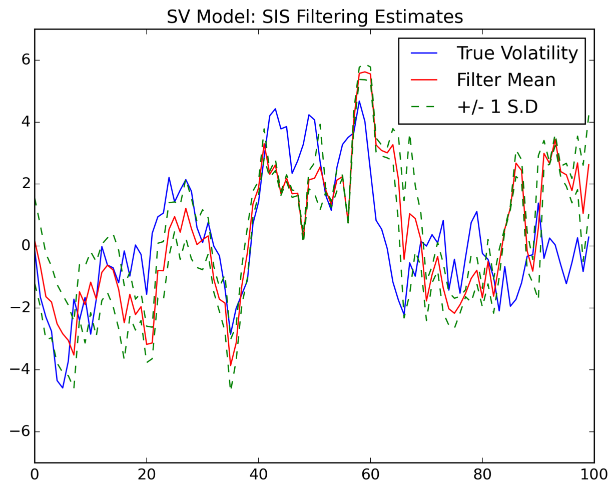


Figure 2: Filtering estimates obtained for the stochastic volatility model using SIS

of this algorithm is due to weight degeneracy[1], which is supported by fig. 3 where it can be seen that while the initialisation of the algorithm is reasonable, by iteration 10 only a few particles have large weights, and by iteration 50, only one particle does.

The SIR algorithm (with multinomial resampling) described above was applied to the same problem, using the same settings as for the SIS algorithm, and the filtering estimates are shown in fig. 4. Unlike for the SIS algorithms, the estimates of the mean and the standard deviation remain reasonable even at the last time steps. That is due to the fact that resampling has allowed to carry forward the particles with large weights, as can be seen in fig. 5: there are many particles with significant weights in all 3 iterations.

However, frequent resampling results in a poorer path-space representation, and has been referred to as the sample degeneracy and impoverishment trade-off in [4]. We have evidence of this path degeneracy when we consider the smoothing distributions $p(x_n | y_{1:500})$ shown in fig. 6.

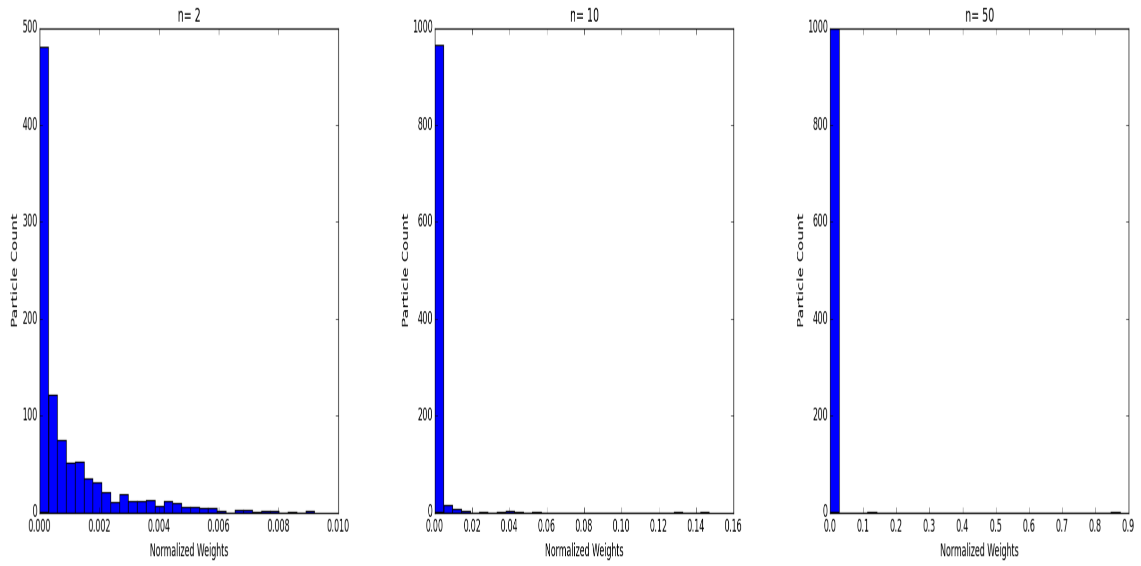


Figure 3: Distributions of the particle weights obtained with SIS for the stochastic volatility model at iterations 2, 5, 50

2.3 Auxiliary Particle Filtering

Auxiliary particle filtering (APF) is a method that extends the SIR algorithm by trying to predict which samples will exist in high probability regions at time $n+1$ by using information from the current state n. In essence, APF uses information from time step $n+1$ to calculate the weights at time step n, i.e. α_n[1].

Using the SMC framework, the target distribution, $\gamma(x_{1:n})$, for APF is expressed as

$$\gamma(x_{1:n}) = p(x_{1:n}, y_{1:n})\tilde{p}(y_{n+1}|x_n)$$

$\tilde{p}(y_{n+1}|x_n)$ is chosen as an approximation to $p(y_{n+1}|x_n)$ if the latter cannot be calculated analytically. Indeed, due to the form of the SV model, the approximation has to be conducted because $p(y_{n+1}|x_n) = \int g(y_{t+1}|x_{t+1})f(x_{t+1}|x_t)dx_{t+1}$ and thus requires the calculation of an integrand of the form $e^{e^{-x}-x}$. It also follows that

$$\pi(x_{1:n}) = \tilde{p}(x_{1:n}|y_{1:n+1}) \propto p(x_{1:n}|y_{1:n})\tilde{p}(y_{n+1}|x_n)$$

The incremental weights, $\alpha_n(x_{n-1:n})$, are now instead calculated as

$$\alpha_n(x_{n-1:n}) = \frac{g(y_n|x_n)f(x_n|x_{n-1})\tilde{p}(y_{n+1}|x_n)}{\tilde{p}(y_n|x_{n-1})q(x_n|y_n, x_{n-1})} = \frac{g(y_n|x_n)\tilde{p}(y_{n+1}|x_n)}{\tilde{p}(y_n|x_{n-1})}$$

However, as mentioned before, an approximation to $\tilde{p}(y_n|x_{n-1})$ has to be expressed. There are various ways of doing this but the most straightforward mentioned in the literature is

$$\tilde{p}(y_n|x_{n-1}) = g(y_n|\mu(x_{n-1}))$$

where $\mu(x_{n-1})$ is the mode, mean, or median of $f(x_n|x_{n-1})$[5]. As the distribution of f is Gaussian, the mode, mean, and median are equal. However, this approximation can lead to a very large or infinite filter variance.

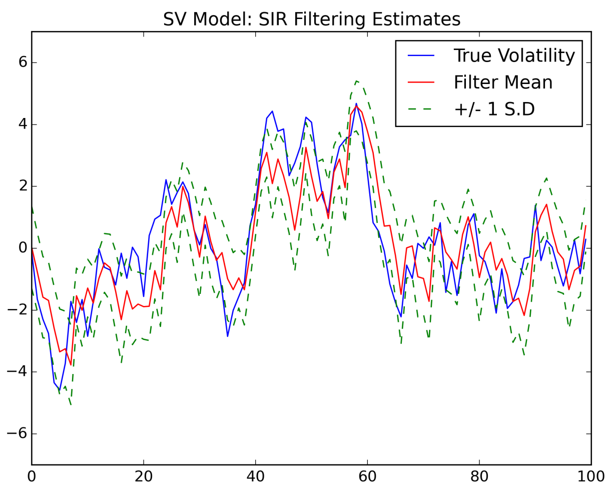


Figure 4: Filtering estimates obtained for the stochastic volatility model using SIR

2.4 Drawbacks of Particle Filters

The algorithms that have been used up until now impose several limitations. The denominator function, $p(y_n|x_{n-1})$, approximated by the importance distribution, q, may have very large variance. In order to combat this issue, resampling has to be conducted to a much larger extent which leads to the unreliability of the approximation $\tilde{p}(x_{1:n}|y_{1:n})$. This issue is very prevalent for the first particles, i.e. for $k << n$, as the resulting approximations will be based on very few unique particles.

Furthermore, the previous algorithms rely only on fixed paths X^i_{n-1}, as they only sample X_n at time n. The improvements presented in the following subsections are two solutions attempting to reduce this degeneracy problem.

2.5 Adaptive Resampling

One simple approach to lessen the severity of the path-degeneracy caused by repeated resampling is to use adaptative resampling: if the variance of the particles is small enough, resampling can be skipped. This can be assessed using criteria such as the Effective Sample Size (ESS), given by

$$ESS = \left(\sum_{i=1}^{N}(W^i_n)^2\right)^{-1}$$

at time n, or the entropy of the weights. Resampling is only done at the time steps where the criterion goes below a threshold.

2.6 Resample-Move using MCMC

The Resample-Move algorithm samples from a Markov kernel with invariant distribution after the resampling step in order to introduce diversity among the particles and thus reduce the degeneracy commonly present in the previous algorithms. We choose a Markov transition kernel such that

$$\int p(x_{1:n}|y_{1:n})K_n(x'_{1:n}|x_{1:n})dx_{1:n} = p(x'_{1:n}|y_{1:n})$$

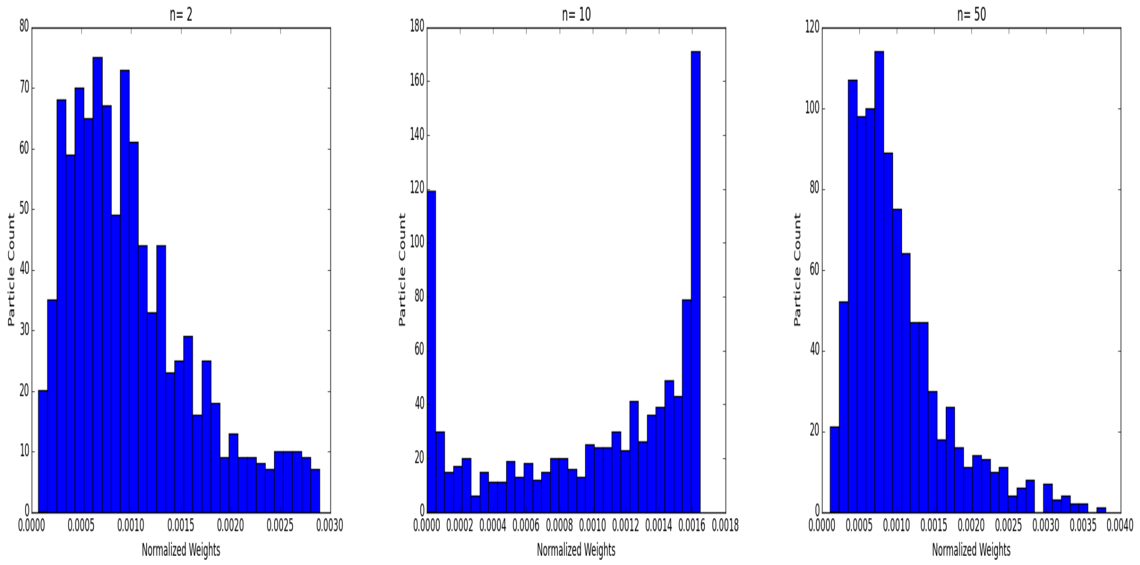


Figure 5: Distributions of the particle weights obtained with SIR for the stochastic volatility model at iterations 2, 5, 50

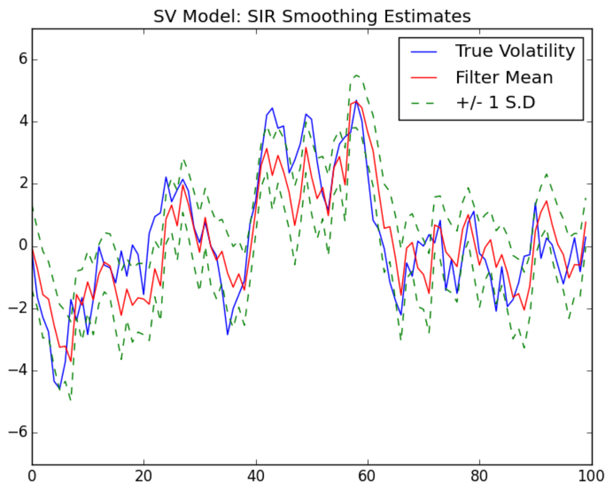


Figure 6: Smoothing estimates obtained for the stochastic volatility model using SIR

We then sample from the kernel like so: set $x'_{1:n-L} = x_{1:n-L}$, then sample x'_{n-L+1} from $p(x'_{n-L+1}|y_{n-L+1}, x'_{n-L}, x_{n-L+2})$, x'_{n-L+2} from $p(x'_{n-L+2}|y_{n-L+2}, x'_{n-L+1}, x_{n-L+3})$ and so on.

To do this we did a Metropolis-Hastings step where we drew a candidate sample from the proposal distribution

$$q(x'_k|y_k, x'_{k-1}, x_{k+1}) = p(x'_k|x'_{k-1})p(x'_k|x'_{k+1} = x_{k+1}) =$$

$$\mathcal{N}(x'_k; \alpha x'_{k-1}, \sigma^2)\mathcal{N}(x'_k; \frac{1}{\alpha}x_{k+1}, \sigma^2) = \mathcal{N}\left(x'_k; \frac{\alpha x'_{k-1} + \frac{1}{\alpha}x_{k+1}}{2}, \frac{\sigma^2}{2}\right)$$

and replaced x'_i with the candidate with probability:

$$\min\left(1, \frac{g(y_k|x'_k)f(x_{k+1}|x'_k)f(x'_k|x'_{k-1})q(x_k|x'_{k-1}, x_{k+1})}{g(y_k|x_k)f(x_{k+1}|x_k)f(x_k|x'_{k-1})q(x'_k|x'_{k-1}, x_{k+1})}\right)$$

otherwise we set $x'_i = x_i$. By only "jittering" the last L particles this way we ensure that the running time w.r.t. n is bounded.

2.7 Block Sampling

Block Sampling constitutes an alternative to the Resample-Move method in order to decrease the number of resampling steps between times k and n, thus avoiding that the marginal distribution $\hat{p}(x_{1:n}|y_{1:n})$ will only be approximated by a few particles for $k << n$. The block sampling has the following approach to limiting the degeneracy problem: At time $n-1$, we have a set of N weighted particles $\{W^{(i)}_{n-1}, X^{(i)}_{1:n-1}\}_{1\leq i\leq N}$ approximating $p(x_{1:n}|y_{1:n})$. We not only extend the currents paths but also sample a section of the previous paths over a fixed lag L [6], instead of just using MCMC moves to rejuvenate these paths as done in the Resample-Move algorithm.

2.8 Method performances

The method performances were measured as average absolute error and average standard deviation on a set of 20 sequences generated from the Stochastic Volatility model. The error is the difference between the filter mean and the true volatility in each time step, and the standard deviation is the standard deviation of the particle set. These measurements are presented in table 1. We consider both the performance on the entire time sequence but also the performance on the final five time steps, as in a practical situation it is usually more important to have a good approximation in the final stages for prediction purposes.

2.9 Financial data

To evaluate the stochastic volatility model, and the methods presented in this paper, we have run the Sequential Importance Resampling algorithm on a series of closing prices of the Google stock in fig. 7. The results are about what one would expect – when the price shoots up or crashes down, the volatility quickly increases. In fact, it is rather easy to find stable points by looking at when the volatility goes down, for example around day 225.

Model	Average absolute error, full series	Average absolute error, final 5	Average standard deviation
SIS	1.43	1.90	0.64
SIR	0.90	0.87	1.10
SIR adaptive ESS	1.17	1.02	1.89
SIR adaptive entropy	1.20	1.07	1.73
APF	1.29	1.25	0.79
MCMC	0.90	0.88	1.10
Block	1.12	0.83	0.16

Table 1: Average performance of all models on 20 data sets generated by the Stochastic Volatility model.

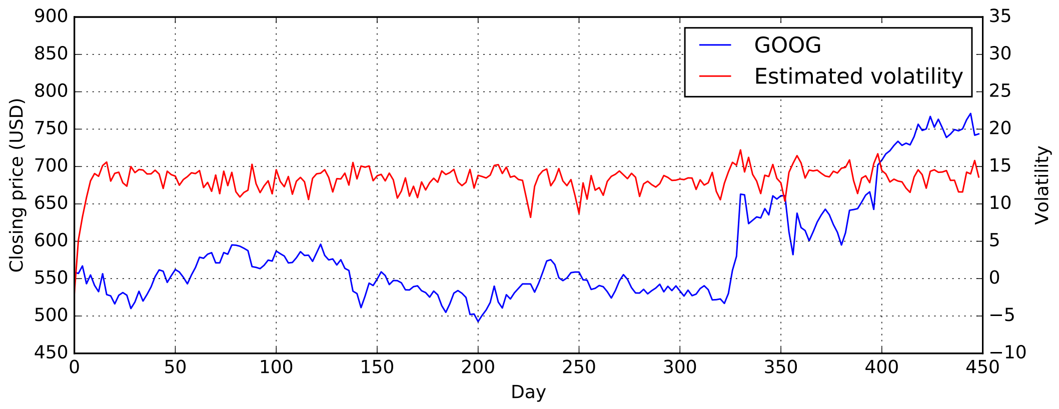


Figure 7: Plot of Google's closing price day by day, and the estimated volatility for the parameters $\alpha = 0.86$, $\beta = 0.005$, and $\sigma = 1$.

3 Discussion

It was previously stated and shown that the variance of SIS increases with n, and it is clear to see in table 1, as the average error is much worse towards the end than over the whole series. As expected, SIS is the worst performing method of all. Introducing the resampling steps with SIR improves the accuracy by fixing the weight degeneracy. Both versions of SIR with adaptive resampling perform better than SIS, but slightly worse than the standard SIR, which would indicate that the improvement of path-space representation did not compensate for the slight additional weight degeneracy compared to SIR.

The APF method performed worse than the basic SIR method when considering their error values. However, the standard deviation of APF is significantly less than for SIR. As mentioned, the approximation, $g(y_n|\mu(x_{n-1}))$, could lead to a very large variance. As seen, this is not an issue for the SV model for the first 100 time-steps. However, one could theorize to why APF does not perform better than SIR with regards to the error values. One possible explanation could be that the approximation is not a suitable one for case of the SV model. In fact, the literature points out very frequently that the choice of approximations should be well suited to the model one wants to estimate.

The MCMC method did not perform better than the SIR method. This could mean that the degeneracy, which is what MCMC aims to reduce, does not seem to be prevalent for SIR on the SV data. It could also mean that our choice of the MCMC kernel has a very little effect on the chains of particles.

The Block Sampling method had a large discrepancy in performance between the full series and the last five time steps. This is an effect of resampling, as we in this method copy the entire time sequence of a resampled particle, the early values will eventually converge to that of a single particle. This also explains the low average standard deviation. In the final steps however, we see a better performance than the other methods, and the variance also increases in the last steps as we have more unique particles. This effect would normally be present in the other resampling methods as well, but as the early values are not used in any calculations in those methods, we chose to only resample the current time step of the particles. One other reason that could explain why the Block sampling method performed worse than SIR is our choice of proposal distribution. As pointed out in [6], the block sampling strategies will only outperform one at a time strategies if the proposal distributions to sample blocks are designed carefully.

4 Conclusion

- The more advanced methods, for example MCMC moves and APF, don't necessarily outperform the basic SIR algorithm for the stochastic volatility model presented here. The reason for this is most likely that the SIR algorithms don't exhibit particularly strong degeneracy.
- For block sampling, the gains are conditional on the use of a sensible approximation of the importance distribution, and of a high variance between successive target distributions. One could argue that the conditional prior we used as an importance distribution (4) mostly for its simplicity, doesn't present these requirements.

References

[1] Arnaud Doucet and Adam M. Johansen. "A tutorial on Particle Filtering and Smoothing: Fifteen years later". In: *Handbook of Nonlinear Filtering* 12 (2008), pp. 656–704.

[2] J Gatheral. *The volatility surface: a practitioner's guide.* Wiley, 2006.

[3] Simon J. Godsill Olivier Cappe and Eric Moulines. "An overview of existing methods and recent advances in sequential Monte Carlo". In: *Proceedings of the IEEE* 95 (2007), pp. 899 –924.

[4] Tariq Pervez Sattar Juan Manuel Corchado Tiancheng Li Shudong Sun. "Review: Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches". In: *Expert Systems with Applications: An International Journal* 41.8 (2014), pp. 3944–3954.

[5] Adam M. Johansen and Arnaud Doucet. "A note on auxiliary particle filters". In: *Statistics and Probability Letters* 78 (2008), pp. 1498–1504.

[6] Arnaud Doucet, Mark Briers, and Stéphane Sénécal. "Efficient block sampling strategies for sequential Monte Carlo methods". In: *Journal of Computational and Graphical Statistics* 15.3 (2006).