Scale Mixture Of Normals (SMiN) Representation

In foreign exchange markets such as the EUR/USD, market moves are widely regarded to be symmetric in nature [**tankov2003financial**]. This motivates the selection of the parameter $\beta = 0$. Instead of considering the general α-stable process, we can instead restrict ourselves to a more analytically tractable sub-class, the a Symmetric α-stable (SαS) process instead.

For $\beta = 0$, we have a convenient Scale Mixture of Normals (SMiN) representation based on the product property of α-stable distributions: If X and Y are independent random variables with $\lambda_k \sim S_{\alpha/2}(1,1,0)$ and $\eta_k \sim S_2(1,0,0) = \mathcal{N}(0,1)$, then $\lambda_k\eta_k \sim S_\alpha(1,0,0)$.

We can then convert the discretised state space model specified in **??** into SMiN form as specified in Equation 1 below.

$$\underbrace{\begin{bmatrix} x_{1,k} \\ x_{2,k} \end{bmatrix}}_{\mathbf{x_k}} = \underbrace{\begin{bmatrix} 1 & \delta t \\ 0 & e^{\theta \delta t} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} x_{1,k-1} \\ x_{2,k-1} \end{bmatrix}}_{\mathbf{x_{k-1}}} + \underbrace{\begin{bmatrix} 0 \\ \sigma_{\delta t} \end{bmatrix}}_{\mathbf{b}} \sqrt{\lambda_k}\eta_k; \qquad \eta_k \sim \mathcal{N}(0,1),\ \lambda_k \sim S_{\alpha/2}(1,1,0) \tag{1}$$

This means that conditional upon us observing λ_k, $\boldsymbol{x}_k$ is gaussian, making inference much more tractable. We exploit this fact by designing a Rao-Blackwellised Particle Filter to sample λ_k by using a simple particle filter, and propogating the states using a Kalman Filter conditioned upon the sampled λ_k of the particle.

We take a short detour to formulate the inference problem using the SMiN representation here. In the inference problem, we are seeking to infer the state variables $\mathbf{x}_k$ given the observations y_k. we assume model dynamics as follows:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{A}\mathbf{x}_{k-1} + \mathbf{b}\sqrt{\lambda_k}\eta_k \\ y_k &= \mathbf{C}\mathbf{x}_k + d\epsilon_k \\ \lambda_k &\sim S_{\alpha/2}(1,1,0) \\ \eta_k, \epsilon_k &\sim \mathcal{N}(0,1) \end{aligned} \tag{2}$$

It will also be useful to use the following expression for y_k:

$$
\begin{aligned}
y_k &= \mathbf{C}\mathbf{A}\mathbf{x}_{k-1} + \underbrace{\begin{bmatrix}\mathbf{C}\mathbf{b}\sqrt{\lambda_k} & d\end{bmatrix}}_{\mathbf{e}_k} \underbrace{\begin{bmatrix}\eta_k \\ \epsilon_k\end{bmatrix}}_{\mathbf{n_k}} \qquad (3) \\
\mathbf{n}_k &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
\lambda_k &\sim S_{\alpha/2}(1,1,0)
\end{aligned}
$$

This transforms our state equation form into a (nearly?) α-stable sub-gaussian form. (This is NOT in α-stable sub-gaussian form, see: $<$ α-stable sub-gaussian definition $>$ for reference)

Generic Rao-Blackwellised Particle Filter (RBPF)

For Rao-Blackwellised Particle Filtering, we partition the state vector into gaussian and non-gaussian components. We can then use standard Kalman Filtering to obtain optimal estimates for the gaussian state components, after obtaining estimates for the non-gaussian state components.

At each time step k, the Rao-Blackwellised Particle Filter produces for each time step k a set of weighted samples $\left\{w_k^{(i)}, \lambda_k^{(i)}, \mu_k^{(i)}, \Sigma_k^{(i)} : i = 1, ..., N\right\}$ according to:

1. Draw new latent variables $\lambda_t^{(i)}$ for each particle in $i = 1, ..., N$ from the corresponding importance distribution:

$$\lambda_k^{(i)} \sim \pi(\lambda_k | \lambda_{0:k-1}^{(i)}, y_{1:k}) \qquad (4)$$

For the generic RBPF, we choose the importance distribution:

$$\pi(\lambda_k | \lambda_{0:k-1}^{(i)}, y_{1:k}) = p(\lambda_k | \lambda_{0:k-1}^{(i)}) = S_{\alpha/2}(\lambda_k | 1, 1, 0) \qquad (5)$$

2. Calculate new weights as follows:

$$w_k^{(i)} \propto w_{k-1}^{(i)} \frac{p(y_k | \lambda_{0:k}^{(i)}, y_{1:k-1}) p(\lambda_k^{(i)} | \lambda_{k-1}^{(i)})}{\pi(\lambda_k^{(i)} | s_{0:k-1}^{(i)}, y_{1:k})} \qquad (6)$$

Here, the likelihood term $p(y_k | u_{0:k}^{(i)}, y_{1:k-1})$ is obtained using the predictive error decomposition from the Kalman Filter:

Kalman Filtering Prediction Step:

$$
\begin{aligned}
p(\boldsymbol{x}_{0:k}|\lambda_{0:k}^{(i)}, y_{1:k-1}) &= \mathcal{N}(\boldsymbol{x}_k|\mu_k^{-(i)}, \Sigma_k^{-(i)}) \qquad (7)\\
\text{where:} \qquad \mu_k^{-(i)} &= \boldsymbol{A}\mu_{k-1}^{(i)}\\
\Sigma_k^{-(i)} &= \boldsymbol{A}\Sigma_{k-1}^{-(i)}\boldsymbol{A}^T + \boldsymbol{b}^T\boldsymbol{b}\lambda_k^{(i)}
\end{aligned}
$$

Predictive Error Decomposition:

$$
\begin{aligned}
p(y_k|\lambda_{0:k}^{(i)}, y_{1:k-1}) &= \int p(y_k|\lambda_{0:k}^{(i)}, \boldsymbol{x}_{0:k})p(\boldsymbol{x}_{0:k}|\lambda_{0:k}^{(i)}, y_{1:k-1})d\boldsymbol{x}_{0:k}\\
&= \mathcal{N}(y_k|\boldsymbol{C}\mu_k^{-(i)}, \boldsymbol{C}\Sigma_k^{-(i)}\boldsymbol{C}^T + d^2) \qquad (8)
\end{aligned}
$$

3. Perform Kalman Filter updates for each of the particles conditional on the drawn latent variables $\lambda_k^{(i)}$.

$$
\begin{aligned}
p(\boldsymbol{x}_{0:k}|\lambda_{0:k}^{(i)}, y_{1:k}) &= \mathcal{N}(\boldsymbol{x}_k|\mu_k, \Sigma_k) \qquad (9)\\
\text{where:} \qquad \boldsymbol{v}_k^{(i)} &= y_k - \boldsymbol{C}\mu_k^{(i)}\\
\boldsymbol{S}_k^{(i)} &= \boldsymbol{C}\Sigma_k^{-(i)}\boldsymbol{C}^T + d^2\\
\boldsymbol{K}_k^{(i)} &= \Sigma_k^{-(i)}\boldsymbol{C}^T\boldsymbol{S}_k^{-1}\\
\\
\mu_k^{(i)} &= \mu_k^{-(i)} + \boldsymbol{K}_k^{(i)}\boldsymbol{v}_k^{(i)}\\
\Sigma_k^{(i)} &= \Sigma_k^{-(i)} - \boldsymbol{K}_k^{(i)}\boldsymbol{S}_k^{(i)}[\boldsymbol{K}_k^{(i)}]^T
\end{aligned}
$$

4. Perform multinomial resampling to increase the number of effective particles.

Potential Problems with the RBPF

When $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$ is large, the RBPF is often unable to get a good importance sampling estimate for λ_k.

When $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$ is large, this implies that λ_k is likely to be large also. (See Figure **??**, noting that $p(\lambda_k|y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}) \propto S_{\alpha/2}(\lambda_k)\mathcal{N}(y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}|\lambda_k))$. As a large λ_k lies in the low probability right tail of the particle proposal distribution given by Equation 5, very few particles $\lambda_k^{(i)}$ are generated from the particle proposal distribution which are close to the actual λ_k.

This problem is exacerbated by the fact that the α parameter of the proposal distribution is half of the original α. This causes the tails of the proposal distribution is to decay very slowly, increasing the number of particles needed to give a good importance sampling estimate.

This results in sample impoverishment, whereby there are only a few effective particles with non-negligible weights, which causes the performance of the RBPF to be slightly worse for very low numbers of particles.

One method of quantifying sample impoverishment in a particle filter (whilst adjusting for number of particles) is by measuring the entropy of the particle filter weights given in Equation 10.

$$H(w) = \sum_{i=1}^{N} w_i \log(w_i) \qquad \text{where: } \sum_{i=1}^{N} w_i = 1 \tag{10}$$

In order to compare the entropy of the particle weights across different number of particles, we instead use a normalised entropy measure given in Equation 11.This normalised entropy measures the change in entropy between a set of weights with uniform distribution (an ideal "optimized" set of weights) and a set of weights with a non-uniform distribution, and is scaled to be independent of the number of particles, as well as the base used in calculating the entropy.

$$H_n(w) = \frac{1}{\log_b(N)} \sum_{i=1}^{N} w_i \log_b(w_i) \qquad \text{where: } \sum_{i=1}^{N} w_i = 1 \tag{11}$$

We demonstrate the problem of sample impoverishment by simulating a single time step of the particle filter, for varying values of $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$. Fixing $\mu_{k-1}^{(i)}$, $\Sigma_{k-1}^{(i)}$ whilst varying y_k and N, we simulate one time step of the RBPF update step described above and present the normalised entropy of the particle filter weights obtained in Figure 1. We see that the normalised entropy of the particle filter weights drops rapidly for large $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$, and that the as N increases, the normalised entropy of the particle filter weights is more resistant to sample impoverishment at large values of $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$ as expected.

List of Figures

List of Tables

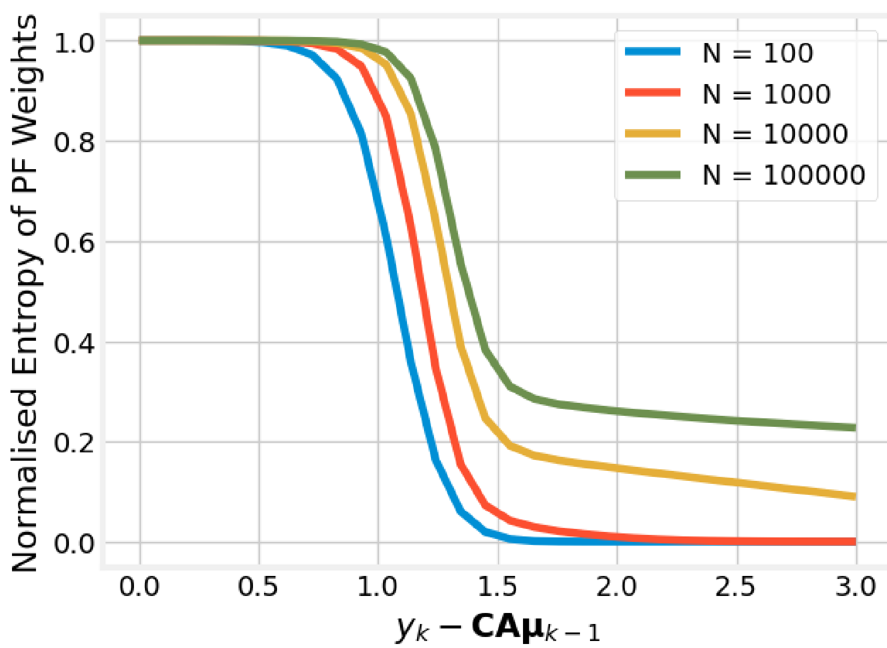

Figure 1: Change in normalised entropy of PF weights as $y_k - \boldsymbol{C}\boldsymbol{A}\boldsymbol{\mu}_{k-1}^{(i)}$ and N are varied