EE 5506 Advanced Statistical Signal Processing
Computer Assignment

Okyanus Oral 2305134

Spring 2021-2022

1 Introduction

This assignment briefly introduces us to stochastic differential equations and their discretization using Euler Maruyama method. From the perspective of stochastic differential equations and their discrete state space models, we consider the Heston model in finance. It is a non-linear model that relates the volatility of the market to the assets' value.

After the brief introduction, a discrete state space and observation model is provided to us for the Heston model. Accordingly, the tasks related to the Heston model are prediction of asset values (from S&P 500 index) with three filters, namely extended Kalman filter (EKF), unscented Kalman filter (UKF) and particle filter (PF). The Heston model parameters are pre-calibrated and given to us.

This report provides the necessary derivations for the application of respective filters to the given discrete time (DT) model. After the necessary derivations are made, conducted tests and their results are presented and discussed in detail.

2 DT Filtering Setup

For convenience I provide the discrete time system equations that were already given to us in our homework.

$$x_k = F_k x_{k-1} + A_k + q_k, \quad q_k \sim N(0, Q_k)$$
$$z_k = H_k x_k + B_k + R_k, \quad R_k \sim N(0, R_k)$$

$$F_k = 1 - \kappa\Delta t + \frac{1}{2}\rho\lambda\Delta t$$
$$A_k = \kappa\theta\Delta t - \lambda\rho\mu\Delta t + \lambda\rho(z_{k-1} - z_{k-2})$$
$$Q_k = \lambda^2(1-\rho^2)x_{k-1}\Delta t$$
$$H_k = -\frac{1}{2}\Delta t$$
$$B_k = z_{k-1} + \mu\Delta t$$
$$R_k = x_{k-1}\Delta t$$

In the following paragraphs, I utilize the following notation in order to derive necessary formulations in a more compact manner:

$$\phi(\Delta t) \triangleq 1 - \kappa\Delta t + \frac{1}{2}\rho\lambda\Delta t$$
$$\eta(\Delta t) \triangleq -\frac{1}{2}\Delta t$$
$$a_1(\Delta t) \triangleq \kappa\theta\Delta t - \lambda\rho\mu\Delta t$$
$$a_2 \triangleq \lambda\rho$$
$$b(\Delta t) \triangleq \mu\Delta t$$
$$r_{var}(\Delta t) = \Delta t$$
$$q_{var}(\Delta t) = \lambda^2(1-\rho^2)\Delta t$$

The notation behind dependency on Δt comes from the data given to us in our homework, where the sampling intervals sometimes differ and are not constant duration.

Realize that both noise covariance matrices involve state or measurement variables. In order to compute partial derivatives we need to factor out state and measurement variables from the pdfs. In order to do this we can define our noise random variables as follows,

$$q_k = q^{'}\sqrt{|x_{k-1}|} \quad \text{,where} \quad q^{'} \sim N(0, q_{var}(\Delta t))$$

$$r_k = r^{'}\sqrt{|x_{k-1}|} \quad \text{,where} \quad r^{'} \sim N(0, r_{var}(\Delta t))$$

In this new form the covariance of the noises are decoupled from state and measurements, however we obtain a nonlinear system of equations. Now let us express the state equations using the new auxiliary variables.

$$x_k = \phi(\Delta t)x_{k-1} + a_1(\Delta_t) + a_2 \cdot (z_{k-1} - z_{k-2}) + q^{'}\sqrt{|x_{k-1}|}$$
$$z_k = \eta(\Delta t)x_k + z_{k-1} + b(\Delta_t) + r^{'}\sqrt{|x_{k-1}|})$$

Realize that the current form of the state equations are not in the form $x_k = f(x_k, q_k)$ because of the dependence to the measurements. Also note that the measurement updates are 'added' to the previous. Therefore, the measurements are not in the form $z_k = h(x_k, r_k)$ either. Lucky for us, both the sate equation and the observation equation can be put to their standard forms by introducing a latent observation variable y_k as follows.

$$\begin{aligned} y_k &\triangleq z_k - z_{k-1} \\ &= \eta(\Delta t)x_k + b(\Delta_t) + r^{'}\sqrt{|x_{k-1}|} \end{aligned}$$

Then, by setting this latent variable as our observation model, we can also obtain our state equation in standard form,

$$\begin{aligned} x_k &= \phi(\Delta t)x_{k-1} + a_1(\Delta_t) + a_2 \cdot (y_{k-1}) + q^{'}\sqrt{|x_{k-1}|} \\ &= \phi(\Delta t)x_{k-1} + a_1(\Delta_t) + a_2 \cdot (\eta(\Delta t^{-})x_{k-1} + b(\Delta_t^{-}) + r^{'}\sqrt{|x_{k-2}|}) + q^{'}\sqrt{|x_{k-1}|} \\ &= \phi(\Delta t)x_{k-1} + a_2 \cdot \eta(\Delta t^{-})x_{k-1} + q^{'}\sqrt{|x_{k-1}|} + a_2 r^{'}\sqrt{|x_{k-2}|}) + a_1(\Delta_t) + a_2 \cdot b(\Delta_t^{-}) \end{aligned}$$

where Δt^{-} corresponds to the time difference between the samples $k-1$ and $k-2$. Accordingly state update and observation equations become;

State Equation:

$$\begin{bmatrix} x_k \\ x_{k-1} \end{bmatrix} = \begin{bmatrix} \phi(\Delta t)x_{k-1} + a_2 \cdot \eta(\Delta t^{-})x_{k-1} + q^{'}\sqrt{|x_{k-1}|} + a_2 r^{'}\sqrt{|x_{k-2}|}) + a_1(\Delta_t) + a_2 \cdot b(\Delta_t^{-}) \\ x_{k-1} \end{bmatrix} = \mathbf{f(x_{k-1}, q)}$$

Observation Equation:

$$y_k = \eta(\Delta t)x_k + b(\Delta_t) + r^{'}\sqrt{|x_{k-1}|} = h(\mathbf{x_k}, r')$$

Please again realize that our observations are $y_k = z_k - z_{k-1}$. Hence, just after we measure z_k, we form y_k by subtracting the z_{k-1}, which is stored in our system.

For this homework we are going to be implementing one step ahead predictor to estimate the next 'value', z_{k+1} of some assets. Accordingly, we predict z_{k+1} from the posterior expectation of $E\{z_{k+1}|z_{1:k}\}$ as follows;

$$\begin{aligned} E\{z_{k+1}|z_{1:k}\} = E\{z_{k+1}|z_k\} &= E\{\eta(\Delta t)x_k + b(\Delta t) + r'\sqrt{|x_{k-1}|} \quad | \quad z_k\} + z_k \\ &= \eta(\Delta t)m_k + b(\Delta t) + z_k \end{aligned}$$

3 Tasks

In the following sections I will not derive the respective equations but nevertheless try to provide you all the respective steps that is needed to be taken into account in order to implement the filters.

3.1 TASK 1: Extended Kalman Filter (EKF)

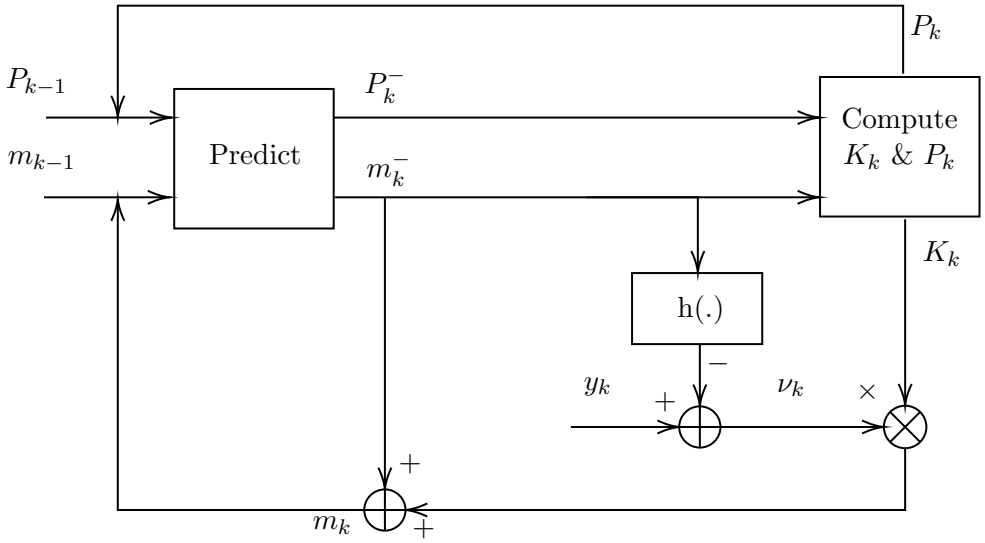


Figure 1: Kalman Filter Block Diagram

The block diagram of running Kalman filter on the observations is given in Fig.1. We can utilize this block diagram for our EKF. The necessary prediction and update steps are given below.

- *Prediction:*

$$\mathbf{m}_k^- = \mathbf{f}(\mathbf{m}_{k-1}, \mathbf{0}),$$
$$\mathbf{P}_k^- = \mathbf{F_x}(\mathbf{m}_{k-1})\,\mathbf{P}_{k-1}\,\mathbf{F_x^\intercal}(\mathbf{m}_{k-1}) + \mathbf{F_q}(\mathbf{m}_{k-1})\,\mathbf{Q}_{k-1}\,\mathbf{F_q^\intercal}(\mathbf{m}_{k-1}),$$

- *Update:*

$$\mathbf{v}_k = \mathbf{y}_k - \mathbf{h}(\mathbf{m}_k^-, \mathbf{0}),$$
$$\mathbf{S}_k = \mathbf{H_x}(\mathbf{m}_k^-)\,\mathbf{P}_k^-\,\mathbf{H_x^\intercal}(\mathbf{m}_k^-) + \mathbf{H_r}(\mathbf{m}_k^-)\,\mathbf{R}_k\,\mathbf{H_r^\intercal}(\mathbf{m}_k^-),$$
$$\mathbf{K}_k = \mathbf{P}_k^-\,\mathbf{H_x^\intercal}(\mathbf{m}_k^-)\,\mathbf{S}_k^{-1},$$
$$\mathbf{m}_k = \mathbf{m}_k^- + \mathbf{K}_k\,\mathbf{v}_k,$$
$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k\,\mathbf{S}_k\,\mathbf{K}_k^\intercal,$$

Where $\mathbf{F_x}, \mathbf{H_x}$ and $\mathbf{F_r}, \mathbf{H_r}$ are the Jacobian matrices of $\mathbf{f}(\mathbf{x_{k-1}}, \mathbf{q})$ and $h(\mathbf{x_k}, r')$ with respect to state variables $\mathbf{x}$ and noise variables $\mathbf{q}$ and r' respectively. Accordingly for our model, $\mathbf{F_x}, \mathbf{H_x}, \mathbf{F_r}$ and $\mathbf{H_r}$ become:

$$\mathbf{F_x} = \nabla_{\mathbf{x}}(\mathbf{f}(\mathbf{x},\mathbf{q}))|_{\mathbf{x=m},\mathbf{q=0}} = \begin{bmatrix} \phi(\Delta t) + a_2\eta\left(\Delta t^{-}\right) & 0 \\ 1 & 0 \end{bmatrix}$$

$$\mathbf{F_q} = \nabla_{\mathbf{q}}(\mathbf{f}(\mathbf{x},\mathbf{q}))|_{\mathbf{x=m},\mathbf{q=0}} = \begin{bmatrix} \sqrt{|m_{k-1}|} & a_2\sqrt{|m_{k-2}|} \\ 0 & 0 \end{bmatrix}$$

$$\mathbf{H_x} = \nabla_{\mathbf{x}}(\mathbf{f}(\mathbf{x},r))|_{\mathbf{x=m},r'=0} = \begin{bmatrix} \eta(\Delta t) & 0 \end{bmatrix}$$

$$\mathbf{H_r} = \nabla_{r}(\mathbf{f}(\mathbf{x},r))|_{\mathbf{x=m},r'=\mathbf{0}} = \begin{bmatrix} \sqrt{|m_{k-1}|} \end{bmatrix}$$

also since our noise states are now joint, the Q_k and R_k covariance matrices are formed as follows.

$$\mathbf{q} = \begin{bmatrix} q' \\ r' \end{bmatrix} \rightarrow \mathbf{Q_k} = \begin{bmatrix} q_{var}(\Delta t) & 0 \\ 0 & r_{var}(\Delta t) \end{bmatrix}$$

$$r = r' \rightarrow \mathbf{R_k} = \begin{bmatrix} r_{var}(\Delta t) \end{bmatrix}$$

As all of the matrices, vectors and estimation equations are well defined now, we can start EKF iterations.

I start both means, i.e. m_{-1}, m_{-2} from the initial value specified in this homework. Furthermore, I set $\mathbf{P_{-1}} = I \cdot P_0$ where P_0 is given in the homework as the initial variance.

3.2 TASK 2: Unscented Kalman Filter (UKF)

Unscented Kalman Filter does not require calculation of derivatives. Instead it utilizes a sampling strategy in order to predict mean and covariance. Since it takes into account of multiple points, it should be expected to work better than EKF.

For UKF we use unscented transforms in order to obtain our predictions. UKF starts by forming and propagating sigma points to state evolution. Doing so, we can compute the mean, m_k^-, of the transformed sigma points and form our predictions of the states. Similarly, these predictions' mean square error matrix is found by computing their covariance, P_k^-. We then use these estimates to form sigma points in order to propagate through the measurement model. Consequently, we are able to estimate observations' covariance matrix, S_k, mean of the observation, μ_k, and the cross correlation between predictions and observations C_k. Once we obtain these matrices, from joint Gaussian's we can form our estimate, m_k, and compute the conditional mean square error matrix, P_k as

$$m_k = m_k^- + K_k(y_k - \mu_k)$$

$$P_k = P_k^- - K_k S_k K_k^T \quad \text{where the the gain is defined as} \quad K_k = C_k S_k^{-1}$$

An important detail for UKF is the use of augmented variables and the formation of sigma points via augmented variables. Since the sigma points are propagated through $\mathbf{f}(\mathbf{x_{k-1}},\mathbf{q})$ and $h(\mathbf{x_k}, r')$ corresponding filter sizes for the generation of respective sigma points should be taken $L' = 4$ and $L'' = 3$.

For the computation of mean vectors and covariance matrices we utilize the formulas for unscented transform. As their formulation do not concern the content of this homework and since all the UKF steps are graphically depicted and verbally described, I ommit the explicit formulas for UKF from this report.

For the details of the Unscented transform and UKF the reader can refer to (Sarkka, Bayesian Filtering and Smoothing Ch.5.5)

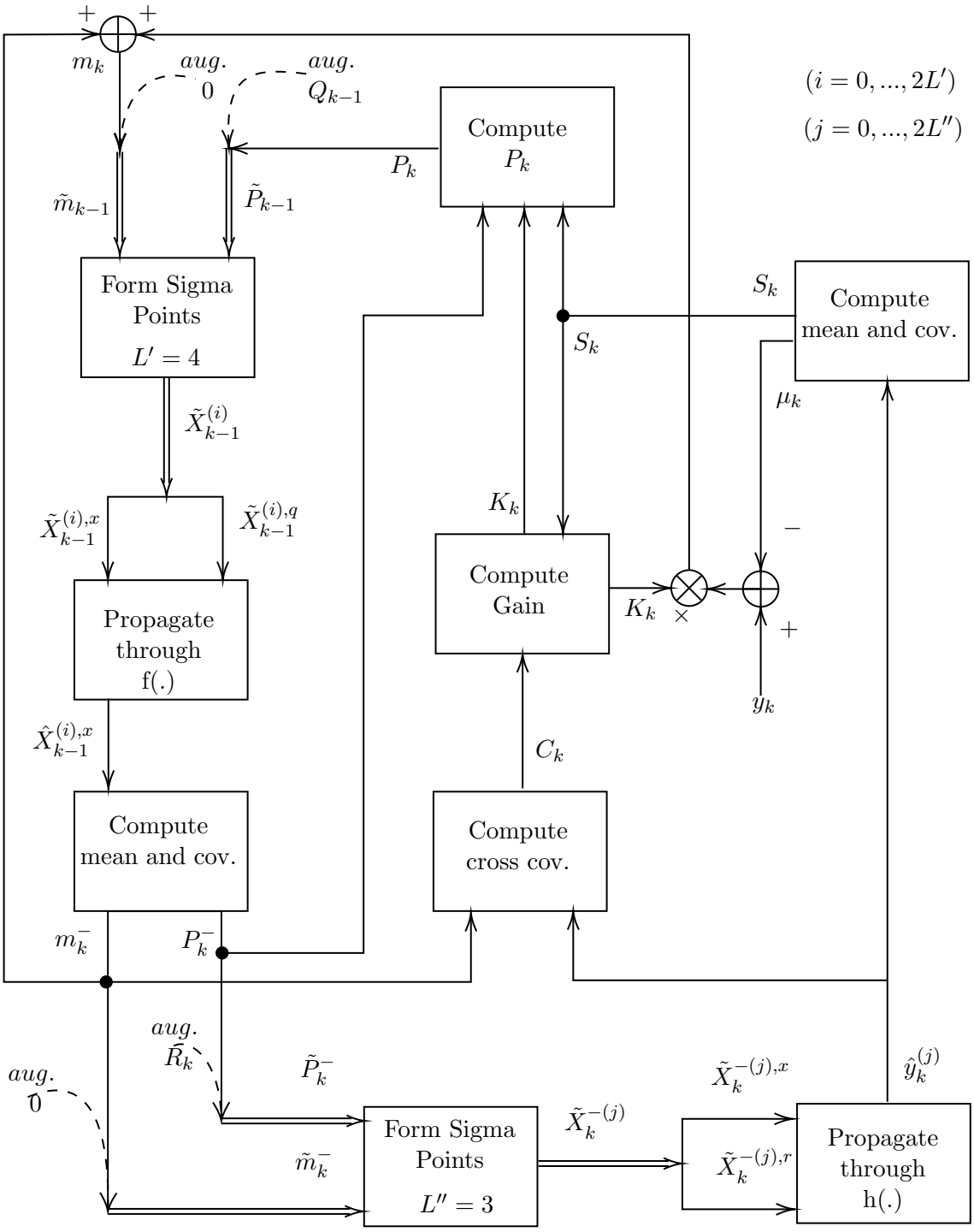


Figure 2: Unscented Kalman Filter Block Diagram

The augmented variables for our model becomes:

$$\text{L'=4:} \quad \tilde{m}_{k-1} = \begin{bmatrix} m_{k-1} \\ m_{k-2} \\ 0 \\ 0 \end{bmatrix} \quad , \quad \tilde{P}_{k-1} = \begin{bmatrix} P_{k-1} & [0\ 0]^T & [0\ 0]^T \\ 0\ 0 & q_{var}(\Delta t) & 0 \\ 0\ 0 & 0 & r_{var}(\Delta t) \end{bmatrix}$$

$$\text{L"=3:} \quad \tilde{m}_k^- = \begin{bmatrix} m_k^- \\ m_{k-1}^- \\ 0 \\ 0 \end{bmatrix} \quad , \quad \tilde{P}_k^- = \begin{bmatrix} P_{k-1} & [0\ 0]^T \\ 0\ 0 & r_{var}(\Delta t) \end{bmatrix}$$

where P_k is defined the same with EKF.As all of the necessary details are covered, we can start UKF iterations.

I start both means, i.e. m_{-1}, m_{-2} from the initial value specified in this homework. Furthermore, I set $\mathbf{P_{-1}} = I \cdot P_0$ where P_0 is given in the homework as the initial variance.

3.3 TASK 3: Particle Filter (PF)

As the task asks us to implement 'a particle filter' I have taken some initiatives. Block diagram of my particle filter can be seen in Fig.3.

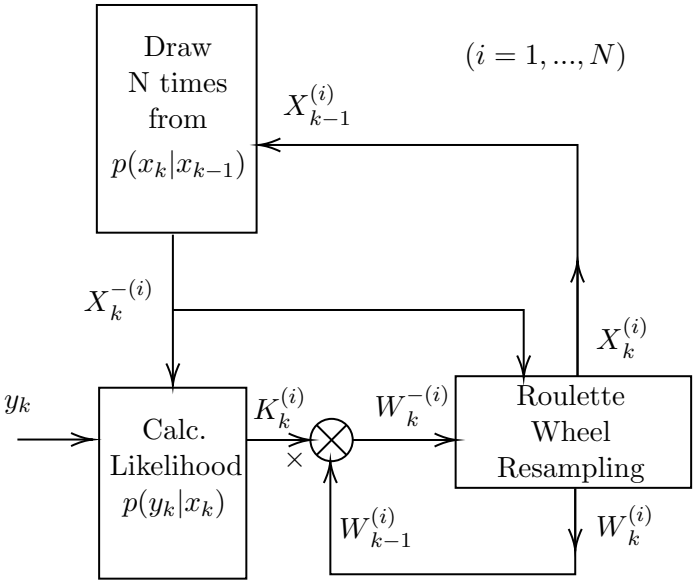


Figure 3: My Particle Filter Implementation

The following algorithm describes a general particle filter:

- Draw N samples, of $x_0^{(i)}$, from a prior distribution $p(x)$ to form the set $X_0^{(i)} = \bigcup_{i=1}^{N}\{x_0^{(i)}\}$. Set their weights as $W_0^{(i)} = \bigcup_{i=1}^{N}\{w_0^{(i)}\}$, where $w_0^{(i)} = \frac{1}{N}$.
- For each k=1...T repeat the following:
  1. Draw N samples from the importance distributions

$$x_k^{-(i)} \sim \pi(x_k|x_{k-1}^{(i)}, y_{1:k}) \quad \text{which,} \quad X_k^{-(i)} = \bigcup_{i=1}^{N}\{x_k^{-(i)}\}$$

  2. Calculate new gain according to

$$K_k^{(i)} \sim \frac{p(y_k|x_k^{(i)})p(x_k^{(i)}|x_{k-1}^{(i)})}{\pi(x_k|x_{k-1}^{(i)}, y_{1:k})}$$

  3. Update the Weights

$$w_k^{(i)} = K_k^{(i)} \cdot w_{k-1}^{(i)}$$

  normalize them.

  4. re-sample,

$$X_k^{(i)} = \text{resample}(X_k^{-(i)})$$
$$W_k^{(i)} = \text{resample}(W_k^{-(i)})$$

- Estimate the expectation of $g(x_k)$ as $\sum_i w_k^i \cdot g(x_k^{(i)})$

Accordingly my particle filter utilizes state evolution as importance distribution and since state evolution is Markovian $p(x_k^{(i)}|x_{k-1}^{(i)}) = \pi(x_k|x_{k-1}^{(i)}, y_{1:k})$. Hence the gain is just likelihood.

For our model,

$$y_k|\mathbf{x_k^{(i)}} \sim N(b(\Delta) + \eta(\Delta t)x_k^{(i)}, r_{var}(\Delta t) \cdot |x_{k-1}|)$$

and one step ahead prediction becomes,

$$\sum_{i=1}^{N} w_k^{(i)}(z_k + b(\Delta t) + \eta(\Delta t)x_k^{(i)})$$

Now everything is defined such that we can start the iterations. I set the mean and variance of the initial distribution to that given in the homework sheet.

4 Implementation & Results

This section describes the test setting and demonstrates/compares the performance of the algorithms.

For the initialization of the algorithms, please refer to the previous sections. All initialization of the algorithms are explained in the sections named with "TASK #".

A slight ambiguity that I have encountered while solving the homework was the choice of Δt. For the Δts, I have used the sequential differences of 'dtime' inside the 'SP500.mat' file. However, in order to test the magnitude of Δt on the performances, I scaled the time differences inside the algorithms with a scale factor. For all the results, this scaling factor is indicated with the corresponding 'simulated' Δt.

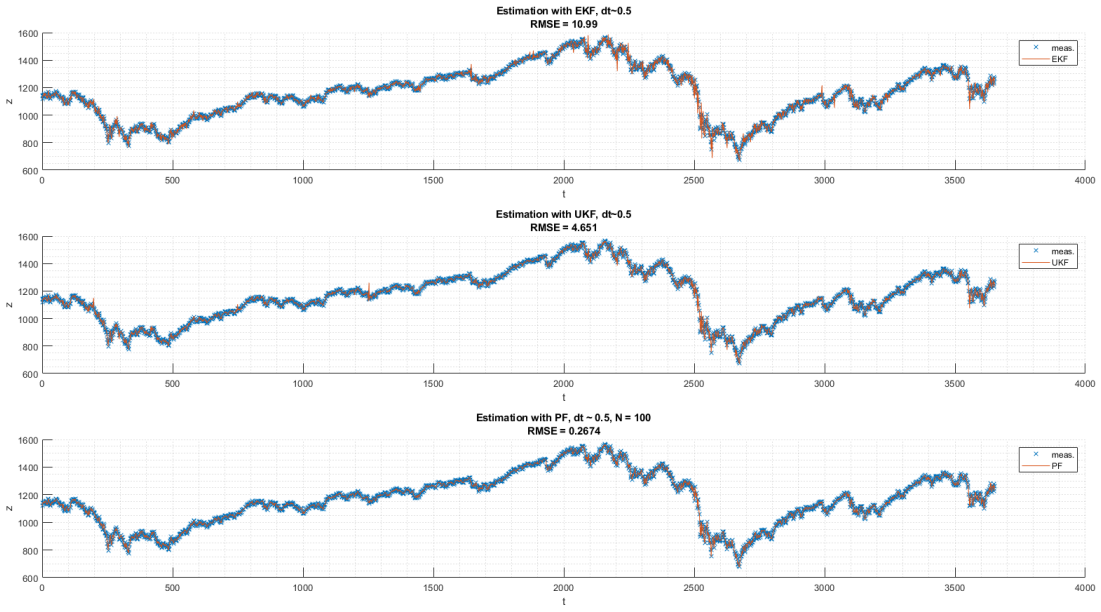


Figure 4: Asset Value Estimates (z estimate) with EKF, UKF, PF. The simulated time difference is $\Delta t = 0.5$. RMSE between z estimates and the actual observation are 10.99 (EKF), 4.651 (UKF), 0.2674 (PF).

In Fig. 4, although visually all filters seem to perform more or the less the same, when we consider RMSE of their z estimates, we can see that the best performing filter is PF while the worst is EKF. This was expected since EKF is relies on a single local linearization around the predicted states per iteration, whereas PF and UKF use mulitple points to approximate the nonlinearity . For convenience I provide the close-up visualizations of the estimations in $t \in (1850, 2300)$ in Fig. 5. From Fig.5, we can see that EKF

fails especially on the regions when the data is volatile. When the data is volatile, the observations are taken afar from the linearization point, hence EKF estimates become degraded.

The reason behind better performance of PF over UKF may be the amount of particles used N.

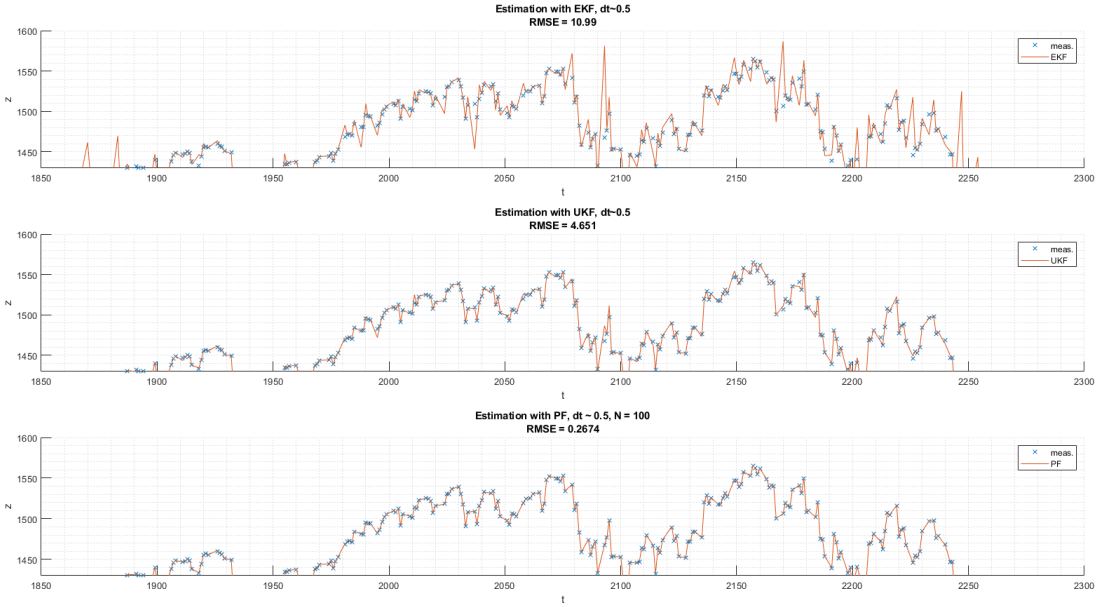
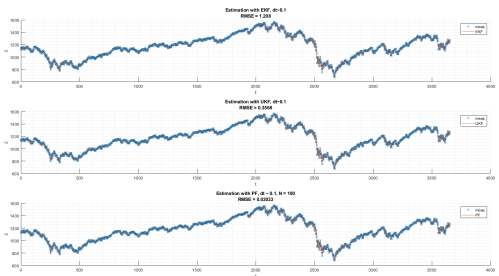
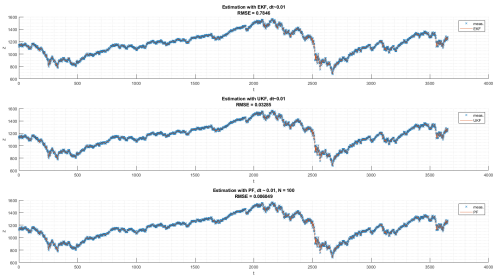


Figure 5: Close up on Asset Value Estimates (z estimate) with EKF, UKF, PF. The simulated time difference is $\delta t = 0.5$. RMSE between z estimates and the actual observation are 10.99 (EKF), 4.651 (UKF), 0.2674 (PF).

I have also tested for $\Delta t = 0.1$ and $\Delta t = 0.01$ settings and their results can be seen in Fig. 6a and 6b respectively. However, since the errors are too small, the visual comparison is not applicable. Therefore, as the close-up visualizations are uninformative , I am putting these results in smaller figures to not to crowed up the report.



(a) Asset Value Estimates (z estimate) with EKF, UKF, PF. The simulated time difference is $\Delta t = 0.1$. RMSE between z estimates and the actual observations are 1.208 (EKF), 0.3568 (UKF), 0.03033 (PF).



(b) Asset Value Estimates (z estimate) with EKF, UKF, PF. The simulated time difference is $\Delta t = 0.1$. RMSE between z estimates and the actual observations are 0.7846 (EKF), 0.03285 (UKF), 0.006049 (PF).

Figure 6: Z estimates and performances for $\Delta t = 0.1$ and $\Delta t = 0.01$

The results in Fig. 6 also show that as Δt decreases the estimates become more accurate. This is also as expected because, we are deriving discrete time equations from an approximation of a continuous time system. As Δt approaches zero, the approximations of derivatives as finite differences become more sound and the estimator performance increases. We are going to return the effect of Δt later, but now

let's inspect x_k estimates and the corresponding P_k variances. To clear up any possible confusions, on the upcoming results, I am providing the estimate of the first state variable, x_k, and its variance $[P_k]_{1,1}$,

In Fig. 7 we see the x estimates for different methods. Since we do not have ground truth, we cannot provide RMSE for x. But nevertheless, the visual inspection of the results are highly informative.

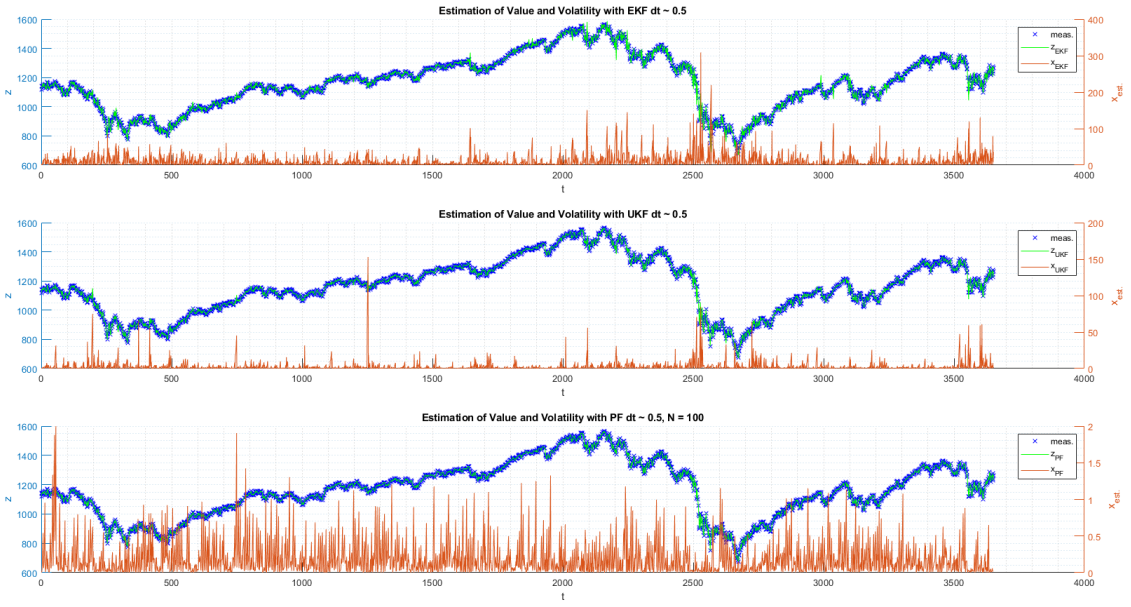


Figure 7: Asset (z_k) and Volatility (x_k) estimates, for $\Delta t = 0.5$.

Realize that x_k are volatility of the assets, as you can see from Fig.7 estimated volatility follows the variance of the observations (as it should do). Moreover, we can see a one fundamental difference between PF and other two Kalman filters. Since PF is Monte Carlo based, in order for us to properly predict/view the internal states, we would need thousands of particles. For my experiments we have only 100 (due to the low performance of my personal computer). That is why our estimates for x_k are quite noisy.

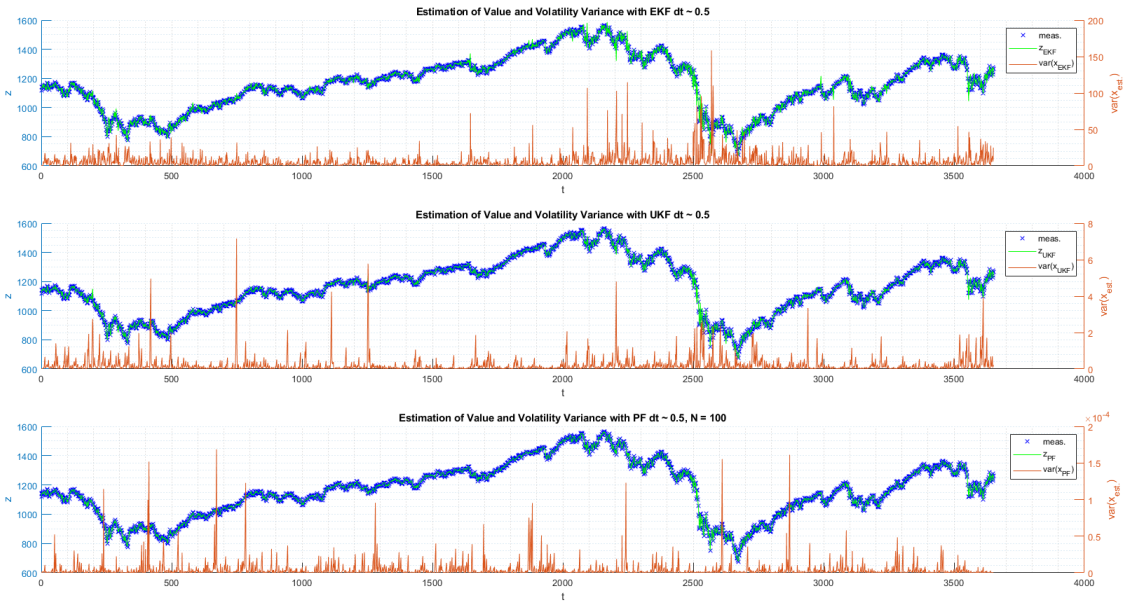


Figure 8: Evolution of $[P_k]_{1,1}$ for different filters, $\Delta t = 0.5$.

Also, when we inspect the volatility, we see that EKF outputs are more explosive. This is again due to poor linearization around rapidly fluctuating regions.

Now we are going to inspect the evolution of $[P_k]_{1,1}$. In Fig. 8 we see that $[P_k]_{1,1}$ are highly noisy, so it is hard to interpret. Nevertheless, we may suspect that all these graphs share a high concentration of peaks for $[P_k]_{1,1}$ estimates around $t = 500$ and $t = 2500$.

In order to decrease the noise on $[P_k]_{1,1}$, we can decrease the Δt.

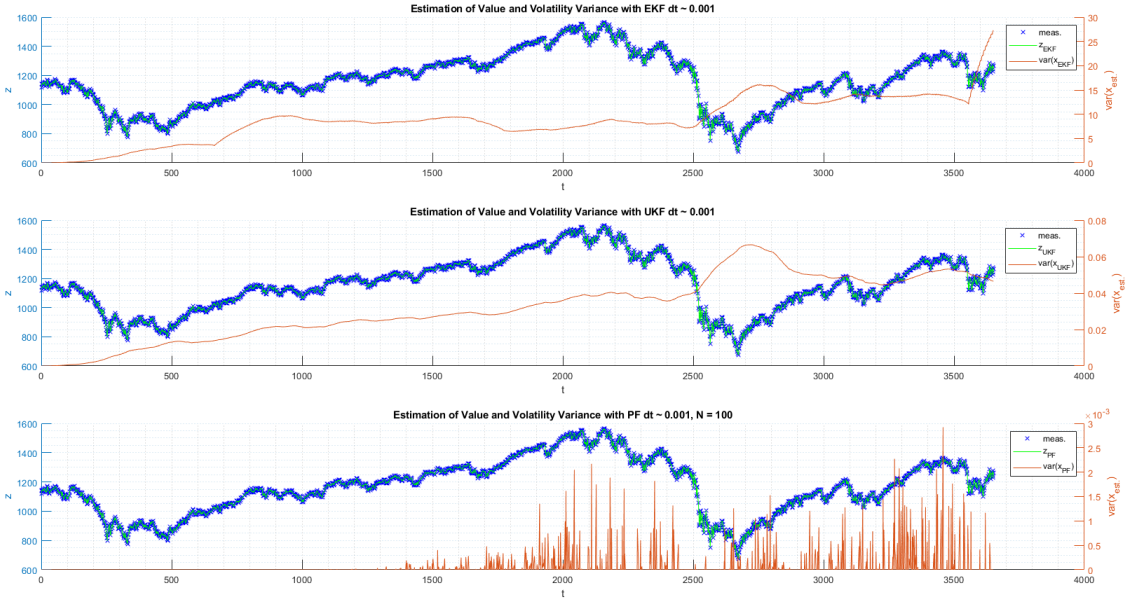


Figure 9: Evolution of $[P_k]_{1,1}$ for different filters, $\Delta t = 0.001$.

From the results in Fig.9 we can again observe the stochastic nature of PF. However, now, $[P_k]_{1,1}$ values for UKF and EKF are much smooth. If we were to run the estimation infinitely long, then $[P_k]_{1,1}$ would go to its steady state, i.e. steady state prediction error covariance. One might argue the reason why error covariance increases. If we were to start the estimations from a further away point than the current one, the prediction error covariance might have decreased (depending on the initialization point). For our case, since the prior exactly matches with the initial state, we obtain minimal variance at the start. Or in other words, we loose performance as we get further away from the only 'accurately know' data entry.

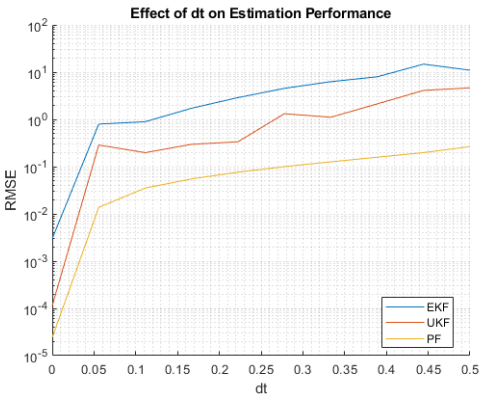


Figure 10: Effect of Δt on Filter Performances.

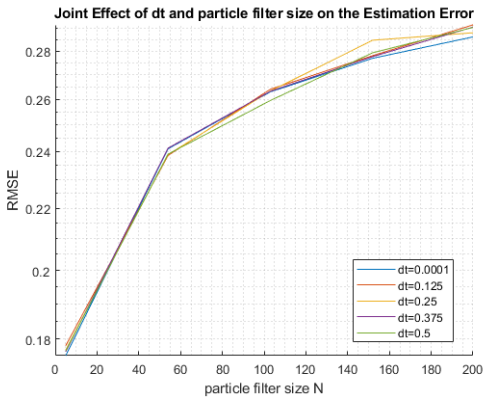


Figure 11: Joint Relation of Δt and N.

Now it is time to once again inspect the effect of Δt to the estimation performance. In order to do so, I computed the RMSE of z_k estimates for the all the filters, on different Δts, see Fig. 10. Δts are selected by taking 10 samples linearly from $\Delta t \in [0.0001, 0.5]$. Moreover, in order to account for the stochastic

nature of PF, I ran particle filter for 5 iterations for each Δt. The shown results for PF are therefore the averages of these trials. It is clear that when the time differences increase, error increases. However, this increase seems to saturate after a certain point for each filter. It is also evident that most severe effect on performance decrease is for EKF (due to bad linearization) while PF is not as effected as the other filters.

I also wanted to see if increasing the number of particles in the particle filter would allow me to compensate for low sampling rate. You can see the results in Fig. 11. It is obvious that the performance mostly depends on the number of particles and not so much on the sampling frequency.

5 Conclusion

In this homework EKF, UKF, and PF are applied for the Heston model. For EKF, the model is linearized and put into the standard form using latent and auxiliary variables. Although, EKF was more challenging to implement (due to calculating partial derivatives by hand), its performance was worse than the other two filters. Nevertheless, it requires the minimum amount of computational resources since there are not any particles. However, depending on the cost of the derivatives functions, other methods could potentially demand less resources (as an exception). After EKF, we have used augmented variables and unscented transforms for UKF. Due to utilization of multiple samples, UKF managed to account non-linearities better than EKF. This was an expected outcome. At the final task we have implemented PF. The particle filter outperformed all others on simulated tests. Throughout the simulations we have seen the differing behaviour of PF. Since PF is a Monte Carlo based method its internal states are not smooth but rather random samples from distributions. Here we talked about the effect of particle size on the 'observability' of the hidden states.

While analyzing the results, we have seen the explicit relation of volatility estimates and the fluctuations of the measurements. Moreover, we have inspected prediction error variance and discussed the effect of the prior information. This led to the discussion about convergence of Kalman filters to a steady state prediction error. At last we have tested the effect of time difference on estimation perfomance, and seen that with a smaller sampling period performances increase. For PF, we further investigated if we can compensate the performance drop due to sampling with particle filter size. We realized that the performance of the PF mostly depends on particle filter size and not the time difference.