UKF-ESN Hybrid Method for Option Pricing

Author: Ziyang Bob Ding

July 3, 2020

Abstract

In this research, I'm aiming to utilize Echo State Neural Network (ESN) as the inner-dynamic of Unscented Kalman Filter (UKF) to model rough stochastic volatility process, therefore achieve the task of option pricing.

1 Introduction

2 ESN

2.1 Variable Explanation

- p_t Stock price on day t to $t-k+1$ (k dim vector).
- u_t Stock return on days t to $t-k+1$ (k dim vector). $u_t = \frac{p_t - p_{t-1}}{p_{t-1}}$ (element wise)
- θ_t state space dynamics on day t, N dimensional
- $v_t = \mathbb{1}^T \theta_t$ volatility on day t, scalar
- W_t dynamic noise, N dim vector
- $\mathbf{G}$ is a $N \times N$ square recurrent state matrix.
- $\mathbf{G}_{in}$ is a $N \times k$ input control matrix.

2.2 Sub-model Specification

$$\begin{aligned} \theta_t &= \sigma(\mathbf{G}\theta_{t-1} + \mathbf{G}_{in}u_t^2 + \mathbf{b}) + W_t \\ W_t &\sim \mathcal{N}(0, W) \\ \sigma(x) &= \frac{1}{1+e^{-x}} \quad \text{is the sigmoid (logistic) function} \end{aligned}$$

2.3 Discussion

For the ESN approach to work, the reservoir should satisfy the so-called echo state property: the state of the reservoir θ_t should be uniquely defined by the fading history of the input u_t. In other words, for a long enough input u_t, the reservoir state θ_t should not depend on the initial conditions that were before the input.

To satisfy Echo State Property, we need $\mathbf{G}$ to satisfy the following conditions:

1. $\mathbf{G}$ should have spectral radius $\rho(\mathbf{G}) < 1$ (similar as the stationarity statement as in the $AR(p)$
2. $\mathbf{G}$ should be sparse
3. $\mathbf{G}$, and therefore the dimension of θ_t, should be big.

All conditions above are not necessary conditions for echo state property, and they are also not sufficient. But in practice, under most circumstances, having the above conditions satisfied will lead to echo state property.

3 Black-Schole's Formula

In this project, I will be dealing with European Option solely, considering it is easier for computation. Black-Schole's Formula is used to measure an European vanilla option's price based on the related assets (can be a portfolio of assets) return and volatility. The formula assumes the following:

1. **The risk-free rate and volatility of the underlying are known and constant.**
2. **The returns on the underlying are normally distributed.**

BS formula takes the following form:

$$
\begin{aligned}
y_{ti} &= p_t \Phi\left(d_+\right) - K_{ti} e^{-r_t T_{ti}} \Phi\left(d_-\right) \\
d_+ &= \frac{\ln\left(\frac{p_t}{K_{ti}}\right) + \left(r_t + \frac{\theta_t}{2}\right) T_{ti}}{\sqrt{\theta_t T_{ti}}} \\
d_- &= \frac{\ln\left(\frac{p_t}{K_{ti}}\right) + \left(r_t - \frac{\theta_t}{2}\right) T_{ti}}{\sqrt{\theta_t T_{ti}}}
\end{aligned}
$$

Where p_t is asset price; r_t is risk-free interest rate; T_{ti} is exercise time; K_{ti} is strike price; θ_t is volatility; Φ is CDF of normal.

4 Unscented Kalman Filter (UKF)

4.1 Sub-model Specification

$$
\begin{aligned}
\theta_t &= g_t(\theta_{t-1}, u_t, w_t) \\
y_{it} &= f_{ti}(\theta_t, p_t, K_{ti}, T_{ti}) + \nu_t \\
\nu_t &\sim N(0, v)
\end{aligned}
$$

Where

$$g(\theta_{t-1}, u_t, w_t) = \sigma(\mathbf{G}\theta_t + \mathbf{G}_{in}u_t) + W_t$$

is the Echo State Neural Network dynamics. In other word, we're using Echo State Neural Network as

$$f_{ti}(\theta_t, p_t, K_{ti}, T_{ti}) = BS(\theta_t, p_t, K_{ti}, T_{ti})$$

is the Black Scholes formula. Here, because on each day, the stock market has several several different Option choices. In order to incorporate all of their information, we have multiple outputs on each timestep as our target variable y_{it}, the option price on day t.

4.2 Why UKF?

Different from Kalman Filter, the transformation is not linear. Therefore, at least we need EKF to approximate the transformed distribution using Taylor expansion. As I've reproduced the result using EKF, I find that for BS formula, under some circumstances, the function at priori $\mathbb{E}[\theta_t]$ is highly non-linear. Therefore, when using first-order Taylor expansion to approximate the function will lead to large approximation error. Higher order of Taylor approximation can theoretically reduce error. However, the non-linear transformation prohibits us from using Kalman Update equation. Therefore leading to worse predictions.

Thanks to Unscented Transformation, we can use UKF to achieve third-order Taylor expansion approximation. The resulting distribution is still Gaussian. Therefore, Kalman equation can still be utilized.

4.3 Forward Filtering

Forward Filtering Step is slightly different from EKF or KF in the way that a direct translation (or approximate transition) is not available. However, as the an approximated predictive distribution $p(\theta_t|\theta_{t-1}, \mathcal{D}_{t-1})$ is Gaussian, we don't really need the transition and just need to perform the normal-normal update.

$$\mathbb{P}(\theta_t|\mathcal{D}_t) \sim \mathcal{N}(m_t, C_t)$$
$$\mathbb{P}(\theta_t|\mathcal{D}_{t-1}) \sim \mathcal{N}(a_t, R_t)$$

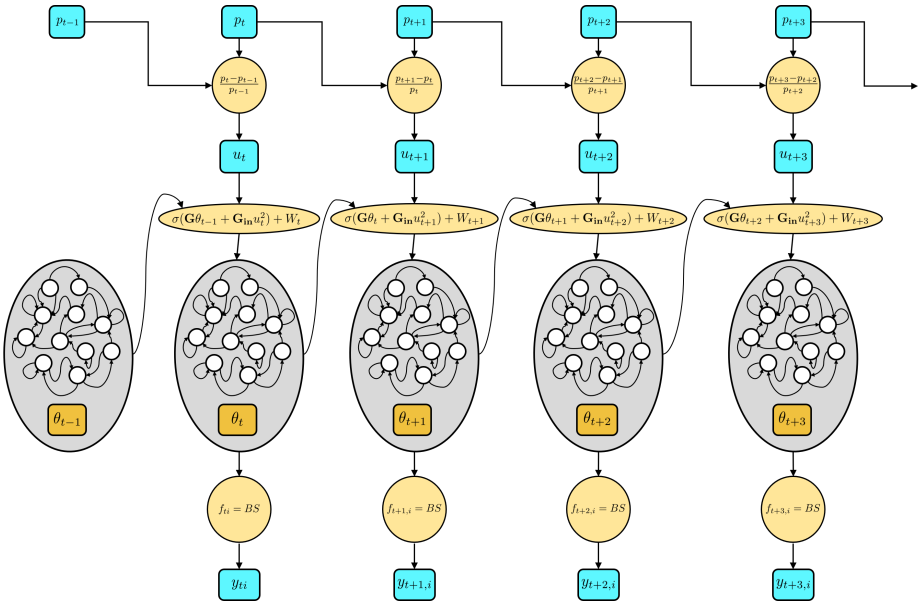
4.4 Backward Smoothing

Backward smoothing for Unscented Kalman Filter uses Rauch-Tung-Striebel smoother (RTS Smoother). The basic idea is the same as usual forward-backward algorithm that we calculate $p(\theta_t, \theta_{t+1}|\mathcal{D}_T)$ joint distribution, then by finding covariance and therefore understand $p(\theta_t|\theta_{t+1}\mathcal{D}_t)$, forming a dynamic programming problem. So we can know $p(\theta_t|\mathcal{D}_T)\forall t$ recursively.

$$\mathbb{P}(\theta_t|\mathcal{D}_T) \sim \mathcal{N}(m_t^*, C_t^*)$$

5 Model Illustration

Attached is a illustration for the model:



6 Parameter Inference

6.1 EM algorithm

In this work I'm proposing using EM algorithm for the work. Let X denote all the observable variables including $\{y_{1:T}, p_{1:T}, u_{1:T}\}$, let Y denote all the latent variables including $\{\theta_{1:T}\}$, and let our parameter set $\{\mathbf{G}, \mathbf{G}_{in}, W, v\}$ be denoted as Θ. Besides, for notational purpose, we denote the following unscented transformation of Gaussian variable:

$$\begin{aligned}
\theta_{t+1} &= ESN(\theta_t) \\
&\sim \mathcal{N}(\Phi_\mu(\theta_t), \Phi_\Sigma(\theta_t)) \\
y_{ti} &= BS_i(\theta_t) \\
&\sim \mathcal{N}(\Psi_\mu^{(i)}(\theta_t), \Psi_\Sigma^{(i)}(\theta_t))
\end{aligned}$$

We can then proceed and write out the E-step as

$$
\begin{aligned}
&\mathbb{E}_X[\log \mathcal{L}(X,Y|\Theta)] \\
=&\mathbb{E}_X\left[\sum_{t=1}^{T}\left\{\log \mathbb{P}(\theta_t|\theta_{t-1},\mathcal{D}_T)+\sum_{i=1}^{n_t}\log(\mathbb{P}(y_{ti}|\theta_t,\mathcal{D}_T)\right\}\right] \\
=&\mathbb{E}_X\left[\sum_{t=1}^{T}\left\{-\frac{1}{2}\log((2\pi)^k|W|)-\frac{1}{2}(\theta_t-\Phi_\mu(\theta_{t-1}))^T W^{-1}(\theta_t-\Phi_\mu(\theta_{t-1}))-\frac{\sum_t n_t}{2}\log(2\pi v)-\frac{1}{2v}\sum_{i=1}^{n_t}(y_{ti}-\Psi_\mu^{(i)}(\theta_t))^2\right\}\right] \\
=&\mathbb{E}_X\Big[-\frac{T}{2}(\log(|W|)-\frac{\sum_t n_t}{2}(\log(v))-\frac{1}{2}\sum_{t=1}^{T}[\theta_t^T W^{-1}\theta_t-2\theta_t^T W^{-1}\Phi_\mu(\theta_{t-1})+\Phi_\mu(\theta_{t-1})^T W^{-1}\Phi_\mu(\theta_{t-1})] \\
&-\frac{1}{2v}\sum_{t=1}^{T}[\sum_{i=1}^{n_t}(y_{ti})^2-2\sum_{i=1}^{n_t}y_{ti}\Psi_\mu^{(i)}(\theta_t)+\sum_{i=1}^{n_t}(\Psi_\mu^{(i)}(\theta_t))^2]\Big]+C \\
=&-\frac{T}{2}(\log(|W|)-\frac{\sum_t n_t}{2}(\log(v)) \\
&-\frac{1}{2}\left\{\mathbb{E}[\sum_{t=1}^{T}\theta_t^T W^{-1}\theta_t|X]-2\mathbb{E}[\sum_{t=1}^{T}\theta_t^T W^{-1}\Phi_\mu(\theta_{t-1})|X]+\mathbb{E}[\sum_{t=1}^{T}\Phi_\mu(\theta_{t-1})^T W^{-1}\Phi_\mu(\theta_{t-1})|X]\right\} \\
&-\frac{1}{2v}\left\{\sum_{t=1}^{T}\sum_{i=1}^{n_t}(y_{ti})^2-2\sum_{t=1}^{T}\sum_{i=1}^{n_t}y_{ti}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X]+\sum_{t=1}^{T}\sum_{i=1}^{n_t}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X]\right\}+C
\end{aligned}
$$

One step EM loss:

$$
\begin{aligned}
&-\frac{1}{2}(\log(|W|)-\frac{n_t}{2}(\log(v)) \\
&-\frac{1}{2}\left\{\mathbb{E}[\theta_t^T W^{-1}\theta_t|X]-2\mathbb{E}[\theta_t^T W^{-1}\Phi_\mu(\theta_{t-1})|X]+\mathbb{E}[\Phi_\mu(\theta_{t-1})^T W^{-1}\Phi_\mu(\theta_{t-1})|X]\right\} \\
&-\frac{1}{2v}\left\{\sum_{i=1}^{n_t}(y_{ti})^2-2\sum_{i=1}^{n_t}y_{ti}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X]+\sum_{i=1}^{n_t}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X]\right\} \\
&+(k+\frac{n_t}{2})\log(2\pi)
\end{aligned}
$$

We look at the non-close-form terms in the equation. We see that

1. $\mathbb{E}[\theta_t^T W^{-1}\theta_t|X] = \mathrm{tr}(W^{-1}C_t^*) + (m_t^*)^T(W)^{-1}m_t^*$
2. $\mathbb{E}[\theta_t^T W^{-1}\Phi_\mu(\theta_{t-1})|X] = \mathbb{E}[*]$
3. $\mathbb{E}[\Phi_\mu(\theta_{t-1})^T W^{-1}\Phi_\mu(\theta_{t-1})|X] = \mathrm{tr}(W^{-1}\Phi_\Sigma(\theta_{t-1}|X)) + \Phi_\mu(\theta_{t-1}|X)^T(W)^{-1}\Phi_\mu(\theta_{t-1}|X)$
4. $\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X] = \Psi_\mu^{(i)}(\mathbb{E}[\theta_t|X])$
5. $\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X] = [\Psi_\mu^{(i)}(\mathbb{E}[\theta_t|X])]^2 + \Psi_\Sigma^{(i)}(\mathbb{E}[\theta_t|X])$

1 is simply an expectation of quadratic form; 3 is unscented transformation with expectation of quadratic form; 4 is unscented transformation; 5 is law of total variance with unscented transform.

If W is diagonal $W = wI$, then we have

$$
\begin{aligned}
&\mathbb{E}_X[\log \mathcal{L}(X, Y|\Theta)] \\
=&-\frac{T}{2}(k\log(w) - \frac{\sum_t n_t}{2}(\log(v)) \\
&-\frac{1}{2}\left\{\mathbb{E}[w^{-1}\sum_{t=1}^{T}\theta_t^T\theta_t|X] - 2\mathbb{E}[w^{-1}\sum_{t=1}^{T}\theta_t^T\Phi_\mu(\theta_{t-1})|X] + \mathbb{E}[w^{-1s}\sum_{t=1}^{T}\Phi_\mu(\theta_{t-1})^T\Phi_\mu(\theta_{t-1})|X]\right\} \\
&-\frac{1}{2v}\left\{\sum_{t=1}^{T}\sum_{i=1}^{n_t}(y_{ti})^2 - 2\sum_{t=1}^{T}\sum_{i=1}^{n_t}y_{ti}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X] + \sum_{t=1}^{T}\sum_{i=1}^{n_t}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X]\right\} + C
\end{aligned}
$$

One step EM loss:

$$
\begin{aligned}
&-\frac{1}{2}(k\log(w)) - \frac{n_t}{2}(\log(v)) \\
&-\frac{1}{2}w^{-1}\left\{\mathbb{E}[\theta_t^T\theta_t|X] - 2w^{-1}\mathbb{E}[\theta_t^T\Phi_\mu(\theta_{t-1})|X] + w^{-1}\mathbb{E}[\Phi_\mu(\theta_{t-1})^T\Phi_\mu(\theta_{t-1})|X]\right\} \\
&-\frac{1}{2v}\left\{\sum_{i=1}^{n_t}(y_{ti})^2 - 2\sum_{i=1}^{n_t}y_{ti}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X] + \sum_{i=1}^{n_t}\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X]\right\} \\
&+(k+\frac{n_t}{2})\log(2\pi)
\end{aligned}
$$

We look at the non-close-form terms in the equation. We see that

1. $\mathbb{E}[\theta_t^T\theta_t|X] = \mathrm{tr}(C_t^*) + (m_t^*)^T m_t^*$
2. $\mathbb{E}[\theta_t^T\Phi_\mu(\theta_{t-1})|X] = \mathbb{E}[*]$
3. $\mathbb{E}[\Phi_\mu(\theta_{t-1})^T\Phi_\mu(\theta_{t-1})|X] = \mathrm{tr}(\Phi_\Sigma(\theta_{t-1}|X)) + \Phi_\mu(\theta_{t-1}|X)^T\Phi_\mu(\theta_{t-1}|X)$
4. $\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)|X] = \Psi_\mu^{(i)}(\mathbb{E}[\theta_t|X])$
5. $\mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2|X] = [\Psi_\mu^{(i)}(\mathbb{E}[\theta_t|X])]^2 + \Psi_\Sigma^{(i)}(\mathbb{E}[\theta_t|X])$

6.1.1 approx-covariate method 1: Unscented Transform

The first way to compute this is use unscented transform again. However, the problem is that even we know $\mathrm{Cov}(\theta_t, \theta_{t-1}|\mathcal{D}_t)$, we don't really know $\mathrm{Cov}(\theta_t, \Phi(\theta_{t-1})|\mathcal{D}_t)$. To get this, we need to compute joint distribution

$$
f\left(\begin{bmatrix}\theta_t \\ \theta_{t-1}\end{bmatrix}\right)\Bigg|\mathcal{D}_t = \begin{bmatrix}\theta_t \\ \Phi(\theta_{t-1})\end{bmatrix}\Bigg|\mathcal{D}_t
$$

Jointly as distribution. Therefore, the off diagonal covariance can be obtained and approximated.

$$
\begin{bmatrix}\theta_t \\ \theta_{t-1}\end{bmatrix}\Bigg|\mathcal{D}_t \sim \mathcal{N}\left(\begin{bmatrix}m_t^* \\ m_{t-1}^*\end{bmatrix}, \begin{bmatrix}C_t^* & P_{t,t-1}^* \\ P_{t-1,t}^* & C_{t-1}^*\end{bmatrix}\right)
$$

$$
\begin{bmatrix}\theta_t \\ \mathbf{G}\theta_{t-1} + \mathbf{G}_{in}u_t^2 + \mathbf{b}\end{bmatrix}\Bigg|\mathcal{D}_t \sim \mathcal{N}\left(\begin{bmatrix}m_t^* \\ \mathbf{G}m_{t-1}^* + \mathbf{G}_{in}u_t^2 + \mathbf{b}\end{bmatrix}, \begin{bmatrix}C_t^* & P_{t,t-1}^*\mathbf{G}^T \\ \mathbf{G}P_{t-1,t}^* & \mathbf{G}C_{t-1}^*\mathbf{G}^T\end{bmatrix}\right)
$$

From here, use unscented transform!

6.1.2 approx-covariate method 2: Taylor approximate

To get 2, we can compromise to use Delta method to approximate. By RTS smoother, we are able to get $\mathbb{P}(\theta_t, \theta_{t-1}|\mathcal{D}_T)$, the joint distribution of 2 Gaussian RV. Therefore, we know

$$\mathbb{P}(\theta_t, \mathbf{G}\theta_{t-1} + \mathbf{G}_{in}u_t^2|\mathcal{D}_T)$$

This is still Gaussian and exact distribution is known. We then want to get

$$\mathbb{E}(\theta_t^T W^{-1}\sigma(\mathbf{G}\theta_{t-1} + \mathbf{G}_{in}u_t^2)|\mathcal{D}_T)$$

Notice that $\sigma(\mathbf{G}\theta_{t-1} + \mathbf{G}_{in}u_t^2)$ is logit-normal distribution, whose shape looks similar to normal (but of course mathematically very different)

$$\begin{aligned}
\begin{bmatrix}\theta_t \\ \theta_{t-1}\end{bmatrix}\Bigg|\mathcal{D}_t &\sim \mathcal{N}\left(\begin{bmatrix}m_t^* \\ m_{t-1}^*\end{bmatrix}, \begin{bmatrix}C_t^* & P_{t,t-1}^* \\ P_{t-1,t}^* & C_{t-1}^*\end{bmatrix}\right) \\
\sigma(x) &= \frac{1}{1+e^{-x}} \\
\sigma(x) &= \frac{1}{1+e^{-x_0}} + \frac{e^{-x_0}}{(1+e^{-x_0})^2}(x-x_0) + O((x-x_0)^2) \\
\tau(x_0) &:= \frac{e^{-x_0}}{(1+e^{-x_0})^2} \\
M &= \mathbf{G}_{in}u_t^2 \\
\sigma(\mathbf{G}\theta_{t-1} + M) &= \sigma(\mathbf{G}m_{t-1}^* + M) + \tau(\mathbf{G}m_{t-1}^* + M) \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*) + O((\theta_{t-1} - m_{t-1}^*)^2) \\
\mathbb{E}(*) &\approx \mathbb{E}(\theta_t^T W^{-1}\sigma(\mathbf{G}m_{t-1}^* + M) + \theta_t^T W^{-1}\tau(\mathbf{G}m_{t-1}^* + M) \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*))) \\
&= (m_t^*)^T W^{-1}\sigma(\mathbf{G}m_{t-1}^* + M) + \mathbb{E}(\theta_t^T W^{-1}\tau(\mathbf{G}m_{t-1}^* + M) \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*)))
\end{aligned}$$

$$\begin{aligned}
M' &= \tau(\mathbf{G}m_{t-1}^* + M) \quad \text{n by 1 matrix} \\
M' &= \begin{bmatrix}q_1 \\ q_2 \\ \vdots \\ q_n\end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[*] &\approx (m_t^*)^T W^{-1}\sigma(\mathbf{G}m_{t-1}^* + M) + \mathbb{E}\left((\theta_t^T - (m_t^*)^T)W^{-1}\left[M' \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*)\right]\right) \\
&\quad + \mathbb{E}\left((m_t^*)^T W^{-1}[M' \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*)]\right) \\
&= (m_t^*)^T W^{-1}\sigma(\mathbf{G}m_{t-1}^* + M) + \mathbb{E}\left[(\theta_t^T - (m_t^*)^T)W^{-1}\left[M' \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*)\right]\right]
\end{aligned}$$

$$\begin{bmatrix} W^{-1}(\theta_t - m_t^*) \\ \mathbf{G}(\theta_{t-1} - m_{t-1}^*) \end{bmatrix} \Big| \mathcal{D}_t \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} W^{-1} C_t^* W^{-T} & W^{-1} P_{t,t-1}^* \mathbf{G}^T \\ \mathbf{G} P_{t-1,t}^* W^{-1} & \mathbf{G} C_{t-1}^* \mathbf{G}^T \end{bmatrix} \right)$$

$$\mathbb{E}\left\{ (\theta_t^T - (m_t^*)^T) W^{-1} \left[M' \odot \mathbf{G}(\theta_{t-1} - m_{t-1}^*) \right] \right\} = \sum_{i=0}^{n} q_i [W^{-1} P_{t,t-1}^* \mathbf{G}^T]_{i,i}$$

$$\begin{aligned} \mathbb{E}[*] &\approx (m_t^*)^T W^{-1} \sigma(\mathbf{G} m_{t-1}^* + M) + \sum_{i=0}^{n} q_i [W^{-1} P_{t,t-1}^* \mathbf{G}^T]_{i,i} \\ &= (m_t^*)^T W^{-1} \sigma(\mathbf{G} m_{t-1}^* + M) + M' \cdot \mathrm{diag}[W^{-1} P_{t,t-1}^* \mathbf{G}^T] \end{aligned}$$

6.1.3 approx-covariate method 3: Hölder

Like evidence lower bound optimization commonly used in variational inference, we can also maximize the upper bound to approximately maximize the expectation.

$$\mathbb{E}[\theta_t^T W^{-1} \Phi_\mu(\theta_{t-1}) | X] \le \|\theta_t^T W^{-1}\|_{L_\infty} \|\Phi_\mu(\theta_{t-1})\|_1$$

$$\begin{aligned} \mathbb{E}_X[\log \mathcal{L}(X, Y|\Theta)] &= -\frac{T}{2}(\log(|W|) - \frac{\sum_t n_t}{2}(\log(v)) \\ &\quad - \frac{1}{2}\left\{ \mathbb{E}[\theta_t^T W^{-1} \theta_t | X] - 2\mathbb{E}[\theta_t^T W^{-1} \Phi_\mu(\theta_{t-1}) | X] + \mathbb{E}[\Phi_\mu(\theta_{t-1})^T W^{-1} \Phi_\mu(\theta_{t-1}) | X] \right\} \\ &\quad - \frac{1}{2v}\left\{ \sum_{i=1}^{n_t} (y_{ti})^2 - 2\sum_{i=1}^{n_t} y_{ti} \mathbb{E}[\Psi_\mu^{(i)}(\theta_t) | X] + \sum_{i=1}^{n_t} \mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2 | X] \right\} + C \\ &\ge -\frac{T}{2}(\log(|W|) - \frac{\sum_t n_t}{2}(\log(v)) \\ &\quad - \frac{1}{2}\left\{ \mathbb{E}[\theta_t^T W^{-1} \theta_t | X] - 2\|\theta_t^T W^{-1}\|_{L_\infty} \|\Phi_\mu(\theta_{t-1})\|_1 + \mathbb{E}[\Phi_\mu(\theta_{t-1})^T W^{-1} \Phi_\mu(\theta_{t-1}) | X] \right\} \\ &\quad - \frac{1}{2v}\left\{ \sum_{i=1}^{n_t} (y_{ti})^2 - 2\sum_{i=1}^{n_t} y_{ti} \mathbb{E}[\Psi_\mu^{(i)}(\theta_t) | X] + \sum_{i=1}^{n_t} \mathbb{E}[\Psi_\mu^{(i)}(\theta_t)^2 | X] \right\} + C \end{aligned}$$

6.2 Regularization

As the reservoir size is usually huge and different from traditional ESN training, during which the input weight $\mathbf{G}_{in}$ and recurrent weight $\mathbf{G}$ are not trained, our model is training the both weights therefore it is highly possible to have overfitting.
Here, LASSO regularization is proposed for 2 reasons:

1. Lasso regularization is able to prevent overfitting
2. Lasso regularization can help achieve sparse $\mathbf{G}$, thus simplifying the dynamic system attractors from becoming chaotic. Also, ESN prefer sparser weights.

6.3 Algorithm

The Model requires statistical inference uppon the following parameters:

$$\{\mathbf{G}, \mathbf{G}_{in}, W, v\}$$

Here, we develop the EM algorithm with an additional LASSO cost for optimization.

1. Properly set initial values for $\mathbf{G}, \mathbf{G}_{in}$ by obeying the Echo State Neural Network requirements. Properly set W, v
2. Forward Filtering: Update all $m_t, C_t, \forall t \in 1:T$
3. Backward Smoothing: Calculate $\mathbb{P}(\theta_t|\mathcal{D}_T)$ including its expectation and covariances.
4. minimize loss function.
5. repeate 2,3,4 until converge.

References

[1] G. Rigatos *A Kalman filtering approach for detection of option mispricing in the Black-Scholes PDE model*. 2014 IEEE Conference on Computational Intelligence for Financial Engineering Economics (CIFEr), London, 2014, pp. 378-383.

[2] K. Liu and X. Wang *A Pragmatical Option Pricing Method Combining Black-Scholes Formula, Time Series Analysis and Artificial Neural Network*. 2013 Ninth International Conference on Computational Intelligence and Security, Leshan, 2013, pp. 149-153.

[3] Introduction to S&P 500, http://www.cboe.com/micro/spx/

[4] London Interbank Offer Rate 3-month rate for U.S. dollars, http://www.economagic.com/em-cgi/data.exe/libor/day-us3m

[5] LIBOR 3 month U.S. data, *FRED Economic Data*, https://fred.stlouisfed.org/series/USD3MTD156N

Experiments

lasso regularization

For Lasso regularization, I didn't place any regularization on $\mathbf{b}$ for the following reasons

1. the bias is different from the neural network bias in the way that in the transition, $\sigma(\mathbf{G}\theta + \mathbf{G}inu_t^2 + \mathbf{b})$ is merely a "mean shift" factor with entry value close to -2. Our volatility $1/n\theta_t \approx 0.1$. Due to the fact that it is slightly away from 0.5, the center of the range of sigmoid function, it is practically unfeasible to ensure "stability" of the volatility process. The only way to remedy this is to add $\mathbf{b}$. In this case, $\mathbf{b}$ doesn't really count or help to "control" the volatility. It is just a mean shift factor. So there is no need to regularizes it.

2. As mentioned above, $\mathbf{b}$ has average entry value close to -2. However, entry value of $\mathbf{G}, \mathbf{G}_{in}$ are relatively smaller, like 0.05 to 0.8. This discrepancy in absolute value will lead to the result that $\mathbf{b}$ will dominate $\mathbf{G}, \mathbf{G}_{in}$, degenerating the entire volatility trajectory. Therefore, no regularization (or at least 2 levels of magnitudes smaller than that of $\mathbf{G}, \mathbf{G}_{in}$) for $\mathbf{b}$

Therefore, the finalized Lasso loss is $\alpha(\sum_{i,j} |\mathbf{G}_{[i,j]}| + \sum_{k,l} |\mathbf{G}_{in[k,l]}|)$

θ dimension

θ dimension represents the size of the reservoir. Typically for traditional ESN, where only the output weights are trained ($\mathbf{G}_{[in]}, \mathbf{G}$ are not trained), the reservoir dimension is about 1.0×10^4. However, for our model, it is just too big for computation. Besides, as the inner weights are being trained, it doesn't require that big a reservoir to capture the dynamics. So I used θ dimension $= 8$.

Confidence Interval

Confidence Interval is very wide. Three things I've tried can help reducing the width of CI (of course also inducing some side-effects):

1. Reducing Lasso shrink coefficient α. Side effect is that it will significantly increase shorter term prediction error ($\alpha = 0.05 \to \alpha = 0.01$ will induce 1 step prediction error from $0.22 \to 1.3$)

2. Increase θ dimension. Side effect is similar as above but less harsh. However, computation speed increase of factor $O(n^3)$.

3. Adjust k, the number of options of greatest volumes at each time steps. This can reduce some noise however not guaranteed. In practice, $k = 5$ leads to smallest CI. But still very big... Side effect is that there isn't an optimal k. It really depends on luck whether the market perception volatility is bi-modeled or is uni-modeled.

4. Use year wise volatility rather than day wise volatility. I don't know why if we use year wise volatility, the CI is still the same as wide. However, as the year wise volatility has a wider scale in variation, that makes the CI to be comparatively narrower. No side effect at all. Therefore I used it.

Some Outputs:

Here are some of the output figures. Explanations of the captions are as follows:

1. THETA: θ dimensions
2. ALPHA: α lasso shrink factors
3. ITR: iteration of EM algorithm stopped to achieve this result
4. 1e-3bias: placing a 0.001 lasso additional shrink on bias term $\mathbf{b}$
5. top3: k, number of options I used for modeling at each time step
6. bias=-2: initialize $\mathbf{b} = -2$. A reasonable initialization of bias will significantly increase training efficiency. Without specification, bias is initialized to -3.
7. longerData: Indication that I used twice more length of training, validation, and testing data. Therefore, the prediction error inevitably gets higher.

Due to these comparisons, my final decisions of the parameters are as follows:

1. θ dimension is 8
2. α shrink factor is 0.05
3. k number of options used is 5
4. no bias $\mathbf{b}$ lasso loss
5. $\mathbf{b}$ initialization -2.3
6. run on shorter data (2018.1.1 - 2019.12.31)

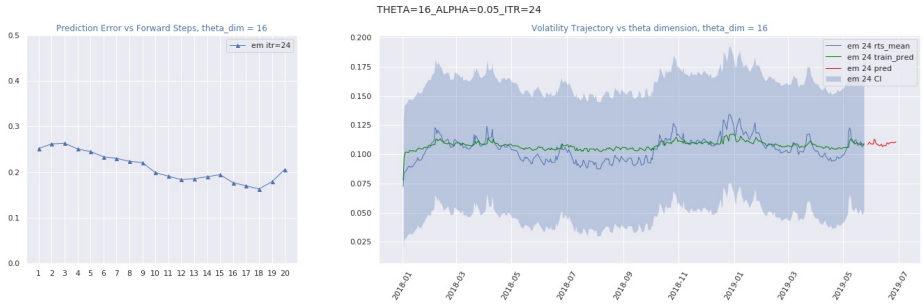


Figure 1: THETA=16 ALPHA=0.05 ITR=24 withCI

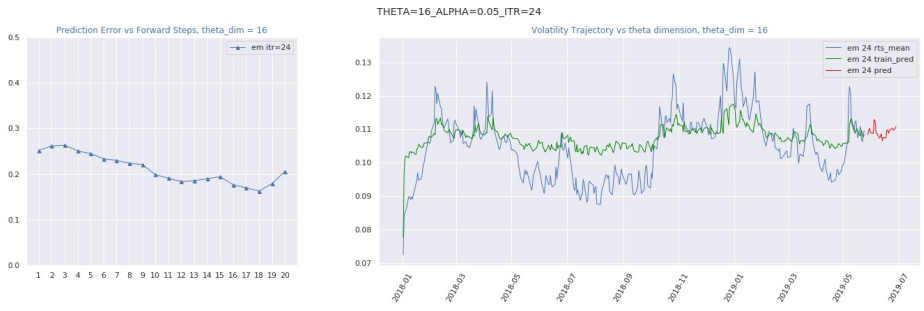


Figure 2: THETA=16 ALPHA=0.05 ITR=24

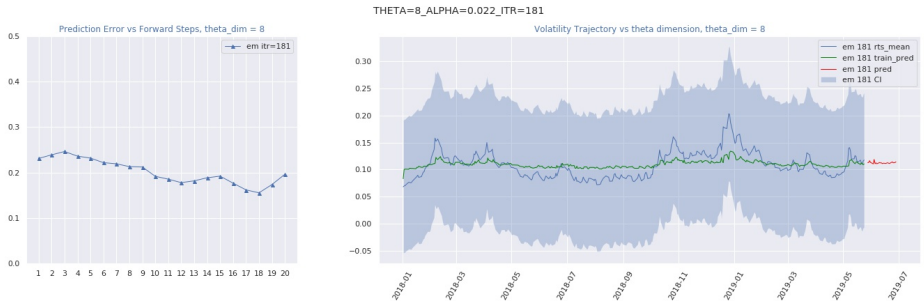


Figure 3: THETA=8 ALPHA=0.022 ITR=181 1e-3bias withCI

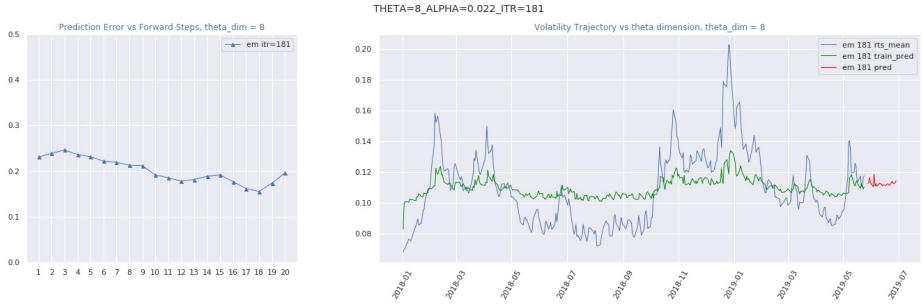


Figure 4: THETA=8 ALPHA=0.022 ITR=181 1e-3bias

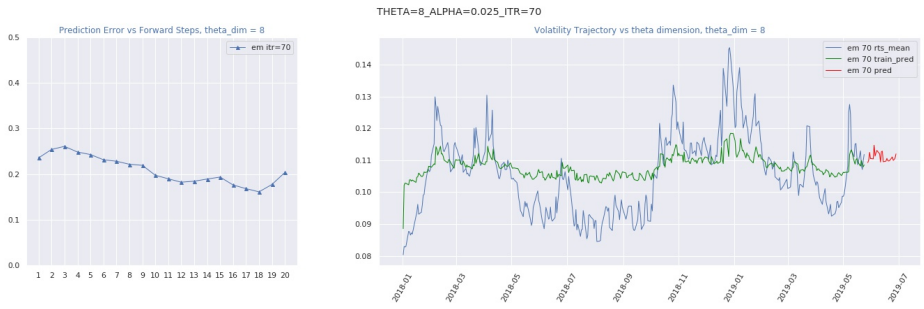


Figure 5: THETA=8 ALPHA=0.025 ITR=70

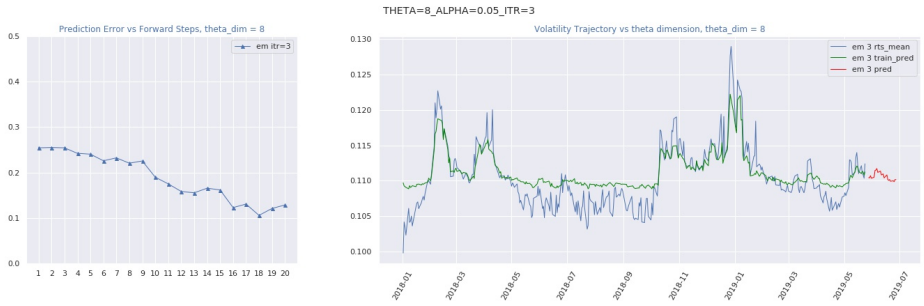


Figure 6: THETA=8 ALPHA=0.05 ITR=3 bias=-2 top3

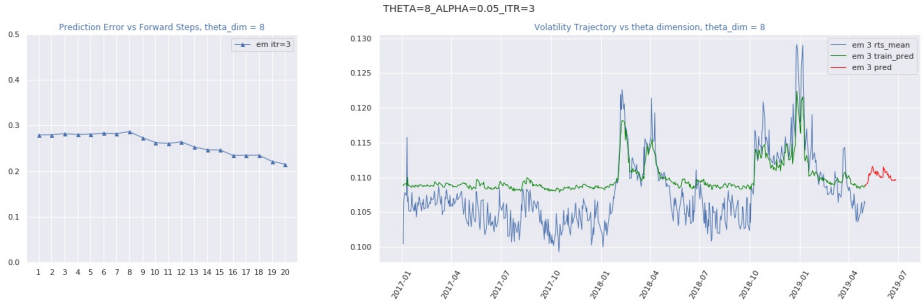


Figure 7: THETA=8 ALPHA=0.05 ITR=3 bias=-2 top5 longerData

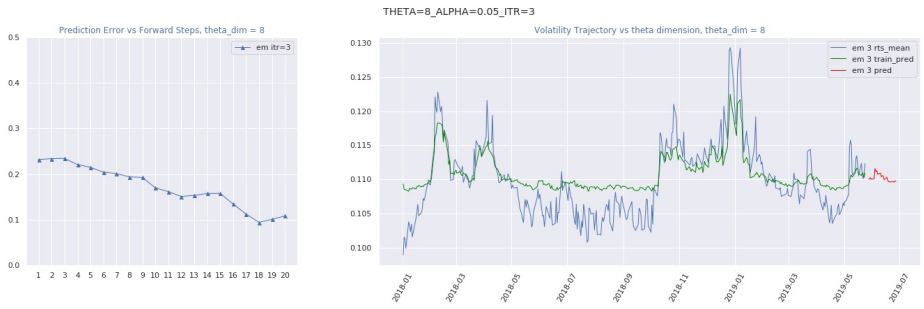


Figure 8: THETA=8 ALPHA=0.05 ITR=3 bias=-2

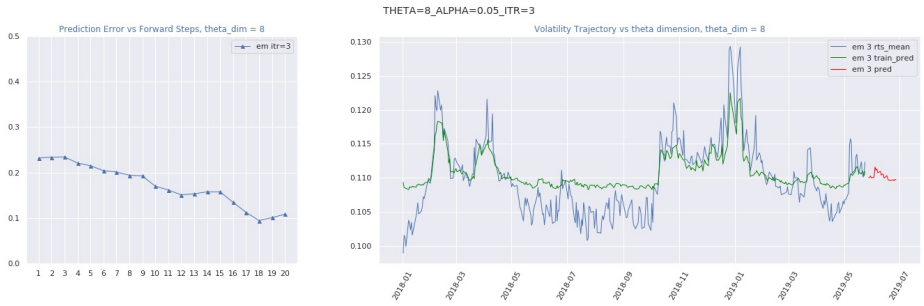


Figure 9: THETA=8 ALPHA=0.05 ITR=3 bias=-2

7 Results

	1-step	5-step	10-step	15-step	20-step
IV	0.9169	0.9108	0.9039	0.8858	0.9646
GARCH	0.7540	0.7364	0.7207	0.7078	0.7908
HARCH	0.7422	0.7234	0.7062	0.6942	0.7754
LSTM	≥ 1	≥ 1	≥ 1	≥ 1	≥ 1
UKFESN	0.2319	0.2149	0.1698	0.1574	0.1081

Table 1: Prediction Errors

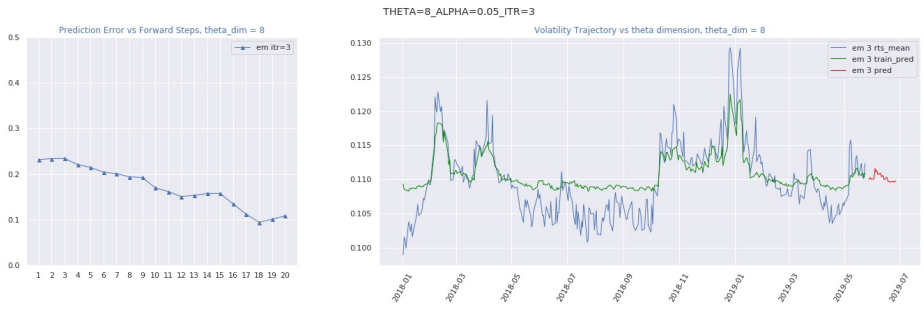


Figure 10: Final Output

Last Revised: July 3, 2020