Computational Finance: Option Pricing with Neural Network Model

Watthanasak Jeamwatthanachai
School of Electronics and Computer Science
University of Southampton
United Kingdom
wj1g14@soton.ac.uk

ABSTRACT

In this report, we study the neural network that can be used as a replacement of Black-Scholes model. We also show the result of neural network model introduced by Hutchinson et al. For the most of this paper, we mostly focus on a comparison between Black-Scholes model and neural network model.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Application and Expert Systems; F.2.0 [**Analysis of Algorithm and Problem Complexity**]: General; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

analysis, performance

Keywords

option pricing, Black-Scholes, neural network

1. LEARNING THE BLACK-SCHOLES FORMULA

According to the Black-Scholes model in the previous assignment, we learnt that Black-Scholes model is a model of price variation during the period of financial instruments determining the price of the call option. Looking back for a couple of decades there is a literature of learning network questioning that how possible to having Black-Scholes formula learnt and estimated in the neural network.

In [1], Hutchinson et al. present their results that brought by the neural network that giving a remarkable accuracy. They purpose the algorithm that consists of two main components in Monte Carlo simulation experiment: training and testing. In training phase, they use history data as a part of training set data in training network and repeatedly for many times to get *average success* network pricing formula with independent option and stock price sample paths. After that we compare the results from learning networks with the results that we do in Black-Scholes formula.

In this papar, the main problem is that how we find the surface that fit the *Gaussian mixture model*, *Shared Covariance*, Σ and four different men, μ. After we get all of these parameters, we now design the matrix in order find *weights*, λ_i which it is shown in the equation 1

$$\begin{aligned}\widehat{C/X} = {} & \lambda 1\sqrt{\begin{bmatrix} S/X-\mu_{11} \\ T-t-\mu_{12}\end{bmatrix}^T \times \begin{bmatrix}\sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22}\end{bmatrix} \times \begin{bmatrix} S/X-\mu_{11} \\ T-t-\mu_{12}\end{bmatrix}} \\ & + \lambda 2\sqrt{\begin{bmatrix} S/X-\mu_{21} \\ T-t-\mu_{22}\end{bmatrix}^T \times \begin{bmatrix}\sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22}\end{bmatrix} \times \begin{bmatrix} S/X-\mu_{21} \\ T-t-\mu_{22}\end{bmatrix}} \\ & + \lambda 3\sqrt{\begin{bmatrix} S/X-\mu_{31} \\ T-t-\mu_{32}\end{bmatrix}^T \times \begin{bmatrix}\sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22}\end{bmatrix} \times \begin{bmatrix} S/X-\mu_{31} \\ T-t-\mu_{32}\end{bmatrix}} \\ & + \lambda 4\sqrt{\begin{bmatrix} S/X-\mu_{41} \\ T-t-\mu_{42}\end{bmatrix}^T \times \begin{bmatrix}\sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22}\end{bmatrix} \times \begin{bmatrix} S/X-\mu_{41} \\ T-t-\mu_{42}\end{bmatrix}} \\ & + W_1 \times S/X \\ & + W_2 \times T-t \\ & + W_0 \end{aligned} \tag{1}$$

2. THE DATASET

According to the dataset we produced in assignment 2 "Option Pricing", given by Black-Scholes Model, we estimate the history volatility given by the length of the majority time, used $T/4+1$ to T. Hence, produce *Call Option price*, $[S/X(T-t)]^T$ and *Delta* afterward by considering that the first 90 data[1] as a training data and the last are for prediction.

3. TRAINING PHASE

In training phase in RBF model, it is split into two part since they cannot be calculated in the same time. It consists of two main as follows:

1. Non-linear function
2. Linear function

[1]This parameter directly affects the accuracy in prediction. More number more accuracy it is.

3.1 Fitting Gaussian Mixture Model

In this part, we can use MATLAB fucntion, called **fitgmdist** in order to satisfy the equation **??** that introduced by Mahalanobis. This equation called Mahalanobis distance that is used to find the distance between a point P and a distribution D. In another word, it is the idea of measuring how many standard deviations away, P, from the mean, D

$$\phi_j(x) = [x - m_j^T \Sigma_j (x - m_j)]^{\frac{1}{2}} \tag{2}$$

While using **fitgmdist**, we must set $TRUE$ one parameter called **Shared Covariance** in order to fit the 90 rows of training data. Consequently, we will get four different pairs of mean and a share covariance. We are now done in this step.

3.2 Design Matrix and Estimate Weights

In this section, we focus on how we can design the matrix and estimate the weight regarded the results from non-linear function, in the previous section, to make the matrix as a linear function. Note that the number of row is depend on how many history data you pick to be as a part of your training set.

$$\begin{bmatrix} \sqrt{J=1} & \sqrt{J=2} & \sqrt{J=3} & \sqrt{J=4} & S/X & T-t & 1 \\ \sqrt{J=1} & \sqrt{J=2} & \sqrt{J=3} & \sqrt{J=4} & S/X & T-t & 1 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ \sqrt{J=1} & \sqrt{J=2} & \sqrt{J=3} & \sqrt{J=4} & S/X & T-t & 1 \end{bmatrix}_{90\times 7} \tag{3}$$

After we product such the matrix above, we need to find the call option price matrix, with the same number of row, by using Black-Schole model. as following:

$$[TheMatrix]_{90\times 7} \times \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_3 \\ W_1 \\ W_2 \\ W_0 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ . \\ . \\ . \\ C_{90} \end{bmatrix}_{90\times 1} \tag{4}$$

After that we now can find *Weights* by calculate the fraction of the matrix divided by *Call Option Price* we get from above.

$$[Weights] = [TheMatrix] \,/\, [CallOptionPrice] \tag{5}$$

4. RESULTS AND PERFORMACE

In this section, we are talking about the results and performance when we plug all of parameters into the learning network equation. The filled equation is shown is Figure **??**. Regarding to the results that we have, we use stock data and strike price from file **c3225.prn** and *Strike Price* is 3225. After we run the training phase, we reproduct the figure 4 and 5 that presented in [1].

In addtion, we compare the performace between Black-Scholes model and RBF model as it is shonw in figure **??**. The results is disclose that it is going quite well even though there are some point modulate different. This is because the accuracy is depend on the given number of data that we used in the training phase.

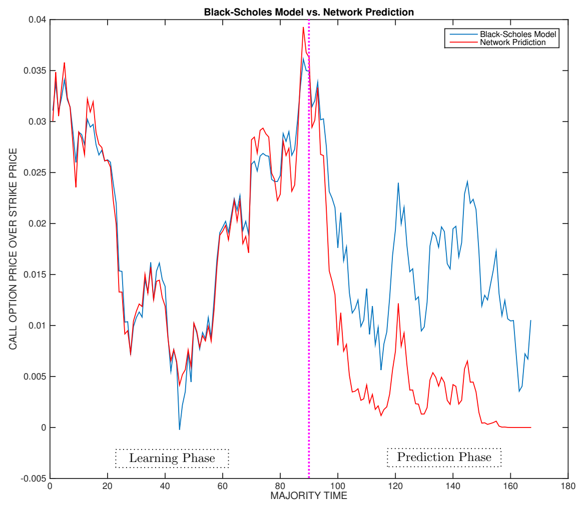

Figure 2: Performace: BS model vs. RBF model

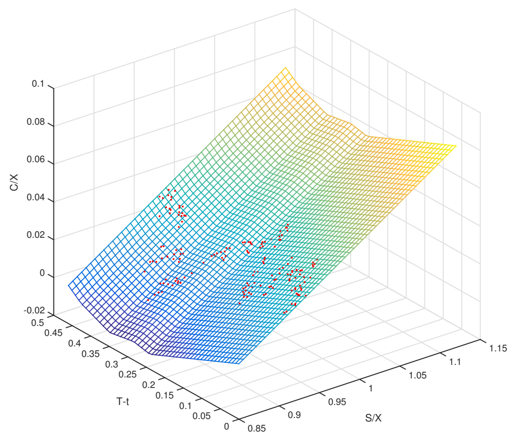

Figure 3: Estimated $\widehat{C/X}$ in RBF model

In figure **??**, we show the result that we get from simulation. It shows the normalization between call option prices and strike price againts stock prices. In figure 4 - 7, we reproduce the figure 5 in [1] with our data regarding **c3225.prn** by using **Curve tool** in MATLAB.

According to Network error showing in the In figure **??**,

$$
\begin{aligned}
\widehat{C/X} = 6.6133 & \sqrt{\begin{bmatrix} S/X - 0.9330 \\ T-t-0.3723 \end{bmatrix}^T \times \begin{bmatrix} 1.4912\times 10^{-4} & -1.7217\times 10^{-4} \\ -1.7217\times 10^{-4} & 3.9681\times 10^{-4} \end{bmatrix} \times \begin{bmatrix} S/X - 0.9330 \\ T-t-0.3723 \end{bmatrix}} \\
& + 0.7875 \sqrt{\begin{bmatrix} S/X - 0.9766 \\ T-t-0.2569 \end{bmatrix}^T \times \begin{bmatrix} 1.4912\times 10^{-4} & -1.7217\times 10^{-4} \\ -1.7217\times 10^{-4} & 3.9681\times 10^{-4} \end{bmatrix} \times \begin{bmatrix} S/X - 0.9766 \\ T-t-0.2569 \end{bmatrix}} \\
& - 2.8503 \sqrt{\begin{bmatrix} S/X - 0.9150 \\ T-t-0.3251 \end{bmatrix}^T \times \begin{bmatrix} 1.4912\times 10^{-4} & -1.7217\times 10^{-4} \\ -1.7217\times 10^{-4} & 3.9681\times 10^{-4} \end{bmatrix} \times \begin{bmatrix} S/X - 0.9150 \\ T-t-0.3251 \end{bmatrix}} \\
& - 2.0731 \sqrt{\begin{bmatrix} S/X - 0.9663 \\ T-t-0.4274 \end{bmatrix}^T \times \begin{bmatrix} 1.4912\times 10^{-4} & -1.7217\times 10^{-4} \\ -1.7217\times 10^{-4} & 3.9681\times 10^{-4} \end{bmatrix} \times \begin{bmatrix} S/X - 0.9663 \\ T-t-0.4274 \end{bmatrix}} \\
& + 0.3003 \times S/X \\
& + 0.0524 \times T - t \\
& - 0.2869
\end{aligned}
$$

Figure 1: Filled Equation

we develop the partial derivative as follows:

$$
\begin{aligned}
\frac{\partial C}{\partial S} = & \frac{1}{2}\lambda_1\{[\sigma_{11}(S/X-\mu_{11})^2 \\
& + (\sigma_{12}+\sigma_{21})(S/X-\mu_{11})(T-t-\mu_{12}) \\
& + \sigma_{22}(T-t-\mu_{12})^2]^{-\frac{1}{2}} \\
& \times [2\sigma_{11}(S/X-\mu 11) + (\sigma_{12}-\sigma_{21})(T-t-\mu_{12})]\} \\
& + \frac{1}{2}\lambda_2\{[\sigma_{11}(S/X-\mu_{21})^2 \\
& + (\sigma_{12}+\sigma_{21})(S/X-\mu_{21})(\mu_{22}) \\
& + \sigma_{22}(T-t-\mu_{22})^2]^{-\frac{1}{2}} \\
& \times [2\sigma_{11}(S/X-\mu_{21}) + (\sigma_{12}+\sigma_{21})(T-t-\mu_{22})]\} \\
& + \frac{1}{2}\lambda_3\{[\sigma_{11}(S/X-\mu_{31})^2 \\
& + (\sigma_{12}+\sigma_{21})(S/X-\mu_{31})(T-t-\mu_{32}) \\
& + \sigma_{22}(T-t-\mu_{32})^2]^{-\frac{1}{2}} \\
& \times [2\sigma_{11}(S/X-\mu_{31}) + (\sigma_{12}+\sigma_{21})(T-t-\mu_{32})]\} \\
& + \frac{1}{2}\lambda_4\{[\sigma_{11}(S/X-\mu_{41})^2 \\
& + (\sigma_{12}+\sigma_{21})(S/X-\mu_{41})(T-t-\mu_{42}) \\
& + \sigma_{22}(T-t-\mu_{42})^2]^{-\frac{1}{2}} \\
& \times [2\sigma_{11}(S/X-\mu_{41}) + (\sigma_{12}+\sigma_{21})(T-t-\mu_{42})]\} \\
& + W_1;
\end{aligned} \tag{6}
$$

Note that we use MATLAB function called **blsdelta** to calculate *Delta*.

5. CONCLUSION

Having proved that by the paper, we learn that power of flexibilty can be used to learn Black-Scholes formula. The result is definitely surprised that working so well. On top of that, it depends on the number of data that put into training set.

6. REFERENCES

[1] J. M. Hutchinson, A. W. Lo, and T. Poggio. A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, 49(3):851–889, 1994.

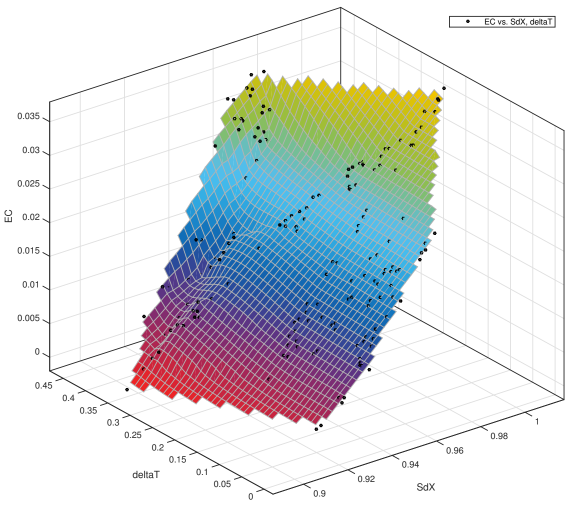

Figure 4: Network Call Price $\widehat{C/X}$

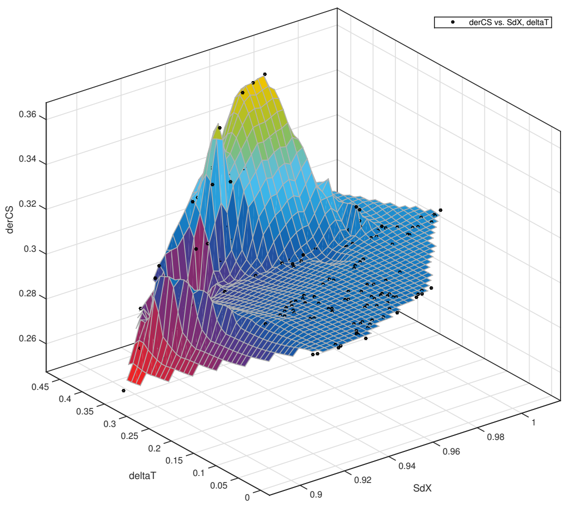

Figure 5: Network delta $\widehat{\frac{\partial c}{\partial S}}$

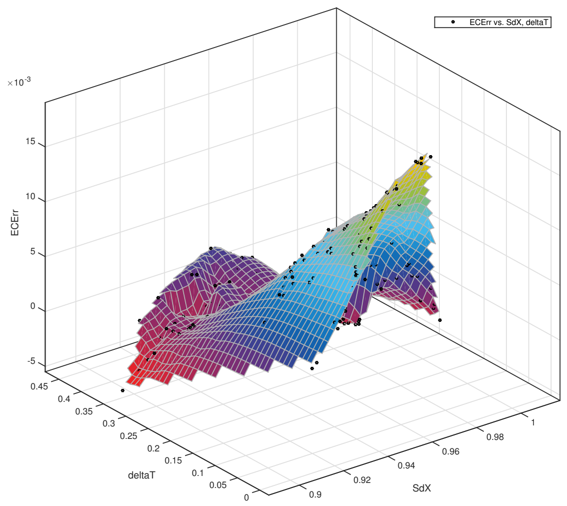

Figure 6: Call Price Error $\widehat{C/X}$ - C/X

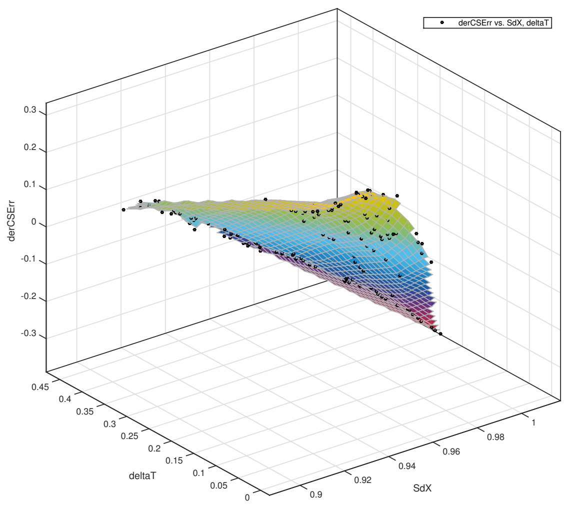

Figure 7: Delta error $\widehat{\frac{\partial c}{\partial S}}$ - $\frac{\partial c}{\partial S}$