ON DEEP LEARNING FOR OPTION PRICING IN LOCAL VOLATILITY MODELS

S.G. Shorokhov

Peoples' Friendship University of Russia (RUDN University), 6 Miklukho-Maklaya St, Moscow, 117198, Russia

E-mail: shorokhov-sg@rudn.ru

We study neural network approximation of the solution to boundary value problem for Black-Scholes-Merton partial differential equation for a European call option price, when model volatility is a function of underlying asset price and time (local volatility model). Strike-price and expiry day of the option are assumed to be fixed. An approximation to option price in local volatility model is obtained via deep learning with deep Galerkin method (DGM), making use of the neural network of special architecture and stochastic gradient descent on a sequence of random time and underlying price points. Architecture of the neural network and the algorithm of its training for option pricing in local volatility models are described in detail. Computational experiment with DGM neural network is performed to evaluate the quality of neural network approximation for hyperbolic sine local volatility model with known exact closed form option price. The quality of the neural network approximation is estimated with mean absolute error, mean squared error and coefficient of determination. The computational experiment demonstrates that DGM neural network approximation converges to a European call option price of the local volatility model with acceptable accuracy.

Keywords: partial differential equation, local volatility model, option price, neural network

Sergey Shorokhov

1. Introduction

In a local volatility model (LVM) [1], in contrast to Black-Scholes constant volatility model [2, 3], the volatility depends on underlying asset price S and time t. Well-known LVM with exact closed form solutions include CEV model [4], shifted lognormal model [5] and normal model (a clone of Ornstein–Uhlenbeck model [6]).

When evaluating derivatives with LVM, the boundary (terminal) value problem for Black-Scholes-Merton (BSM) partial differential equation (PDE) is to be solved. Exact closed form solutions of terminal value problem for BSM PDE are known only in a few special cases, therefore, in general case application of numerical methods such as binomial trees, Monte Carlo simulations, Fourier or finite difference methods is required. Alternatively, derivative prices in LVM can be approximated with an artificial neural network (ANN).

The idea to use ANNs for option pricing is several decades old (see [7] and the references therein), however, the need to improve the quality of option price approximation stimulates further research. Deep Galerkin Method (DGM), introduced recently in [8] for the solution of PDEs, makes use of the neural network of special architecture and stochastic gradient descent (SGD) [9] on a sequence of random time and space points. The method has been successfully applied to various PDEs [8], including BSM PDE with constant volatility.

Our goal is to study application of DGM approach [8] to option pricing when volatility function is not constant and evaluate the quality of ANN approximation for an LVM with known exact analytical closed form solution.

2. Deep option pricing with local volatility models

For pricing of a European call option with strike-price K and expiry day T in LVM with a volatility function $\sigma(S,t)$ and a risk-free interest rate $r > 0$, the solution of BSM PDE [2, 3]

$$\frac{\partial \mathrm{u}}{\partial t} + r\,S\,\frac{\partial \mathrm{u}}{\partial S} + \frac{1}{2}\sigma^2(S,\,t)\,S^2\,\frac{\partial^2 \mathrm{u}}{\partial S^2} - r\,\mathrm{u} = 0 \#(1)$$

with terminal condition

$$\mathrm{u}(S,T,K,T) = \max(S-K,\,0)\ \#(2)$$

is to be determined. Strike-price K and expiry day T are assumed to be fixed.

To determine ANN approximation to PDE (1) with condition (2) using DGM approach the neural network of special architecture [8] is to be built and trained. The architecture of the ANN is similar to architectures of LSTM [10] and Highway [11] networks. It consists of the layers in Fig.1: an input layer, d hidden (LSTM) layers and an output layer.

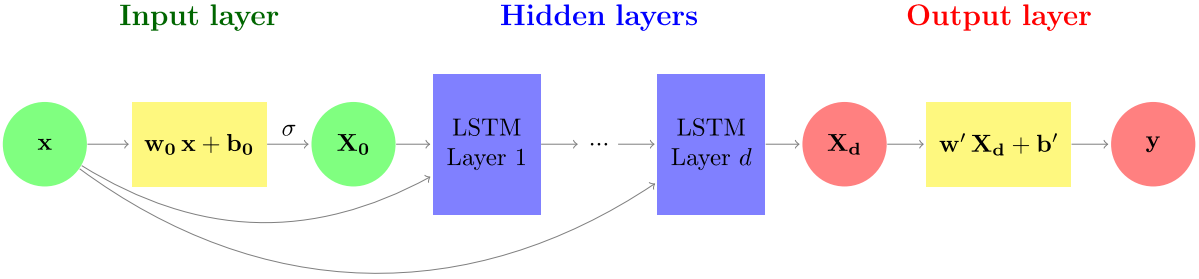

Figure 1. Architecture of DGM neural network

The input to DGM ANN is a set of randomly sampled price-time points $\mathrm{x} = (S,t)$. In the input layer, the price-time points x are transformed into the output X_0

$$X_0 = \sigma(w_0\, x + b_0)$$

with a nonlinear activation function σ and input layer parameters w_0 and b_0.

Each hidden (LSTM) layer receives as an input the original set of price-time points x and the output of the previous layer. In hidden layers, the price-time points x and the outputs of the previous layer X_{i-1} are processed with the following transformations:

$$Z_i = \sigma(u_i^z\, x + w_i^z\, X_{i-1} + b_i^z), \qquad R_i = \sigma(u_i^r\, x + w_i^r\, X_{i-1} + b_i^r),$$

$$G_i = \sigma\big(u_i^g\, x + w_i^g\, X_{i-1} + b_i^g\big), \qquad H_i = \sigma\big(u_i^h\, x + w_i^h\, (X_{i-1} \odot R_i) + b_i^h\big),$$

where $\odot$ denotes element-wise multiplication, and the outputs of the layer are

$$X_i = (1 - G_i) \odot H_i + Z_i \odot X_{i-1}.$$

In the output layer, the outputs of the last LSTM layer X_d are transformed into the neural network outputs y with a linear transform

$$y = f(x;\, \theta) = w'\, X_d + b',$$

where w' and b' are the output layer parameters. The output of the DGM neural network y is the approximation of option price u at the initial price-time points x.

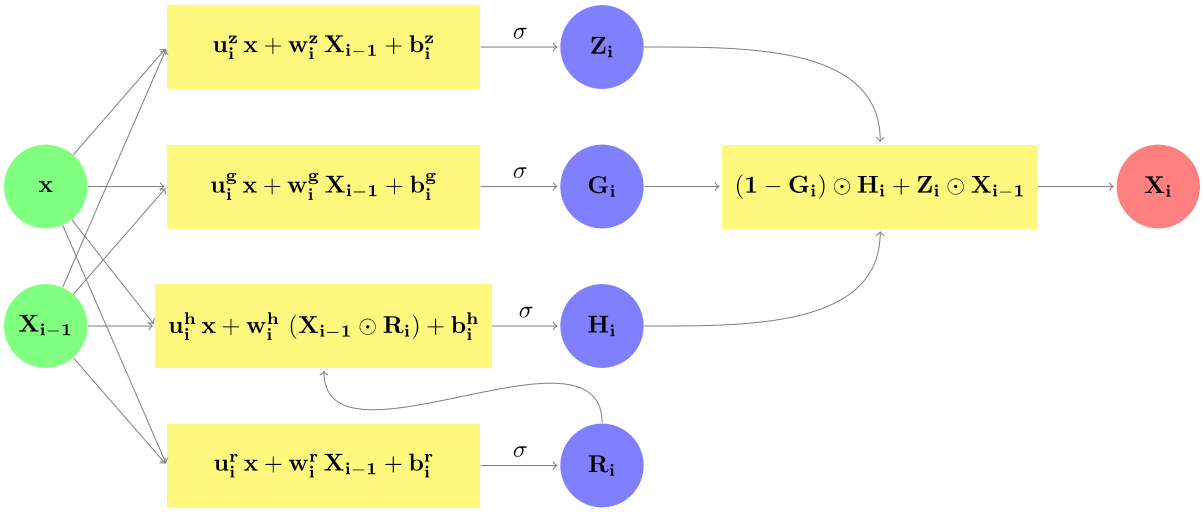

Figure 2. Architecture of LSTM layer in DGM network

The number of parameters (weights and biases) in DGM ANN can be calculated as follows. Let N be the number of neurons (nodes) in each hidden layer of DGM neural network.

In the input layer, the shape of the weight parameter w_0 is $2 \times N$ and the shape of the bias parameter b_0 is $1 \times N$.

In hidden LSTM layers, the shape of the weight parameters $u_i^z, u_i^g, u_i^r, u_i^h$ is $2 \times N$, the shape of the weight parameters $w_i^z, w_i^g, w_i^r, w_i^h$ is $N \times N$, the shape of the bias parameters $b_i^z, b_i^g, b_i^r, b_i^h$ is $1 \times N$.

In the output layer, the shape of the weight parameter w' is $N \times 1$ and b' is a scalar parameter. The neural network parameter set θ contains all weight and bias parameters mentioned above. Thus, the total number of parameters in DGM neural network is equal to

$$|\theta| = 4\, d\, (N + 1)^2 + 4\, N + 1.$$

DGM neural network is trained with adaptive algorithm Adam [12], which is an extension to classical SGD algorithm. General outline of DGM algorithm for the solution of BSM PDE (1)-(2) is shown below in Algorithm 1.

Algorithm 1: Approximation of option price in local volatility model with DGM neural network

Data: a volatility function $\sigma(S,t)$, strike-price K, time to maturity T, risk free interest rate r, distributions ν_1 and ν_2, absolute tolerance $\varepsilon > 0$;

Result: optimal parameter set θ^* for the approximation of option price in LVM;

– choose initial parameter set θ_0 and learning rate α_0;

repeat

– generate random time-price points (S_n, t_n) from $\Omega \times [0,T]$ with distribution ν_1 and random price points S_n' from Ω with distribution ν_2, $\Omega = [s_l, s_h] \subset \mathbb{R}$;

– calculate the loss function $L(\theta_n, \xi_n)$ at the randomly sampled points $\xi_n = \{(S_n, t_n), S_n'\}$, where

$$L(\theta_n, \xi_n) \leftarrow \left(\frac{\partial f(S_n,t_n;\theta_n)}{\partial t} + r\,S_n \frac{\partial f(S_n,t_n;\theta_n)}{\partial S} + \frac{1}{2}\sigma^2(S_n,t_n)\,S_n^2 \frac{\partial^2 f(S_n,t_n;\theta_n)}{\partial S^2} - r\,f(S_n,t_n;\theta_n)\right)^2 + \left(f(S_n',T;\theta_n) - \max(S_n' - K)\right)^2.$$

– update parameter set θ for a gradient descent step at the random points ξ_n for the learning rate α_n with adaptive algorithm Adam [12]:

$$\theta_{n+1} \leftarrow \theta_n - \alpha_n \nabla_\theta L(\theta_n, \xi_n).$$

until $\|\theta_{n+1} - \theta_n\| < \varepsilon$;

$\theta^* \leftarrow \theta_{n+1}$;

As a result of Algorithm 1, an approximation of the price of a European call option in LVM with volatility function $\sigma(S,t)$ is obtained in the form $u(t,S) = f(t,S;\theta^*)$.

3. Computational experiment for hyperbolic sine LVM

Consider hyperbolic sine LVM [13] with underlying asset price driven by SDE

$$dS = r\,S\,dt + \sqrt{2\,r\,S^2 + \lambda^2}\,dW,\ r > 0,\ \lambda > 0 \#(3)$$

with the following BSM PDE for a derivative price

$$\frac{\partial u}{\partial t} + r\,S\,\frac{\partial u}{\partial S} + \frac{1}{2}(2\,r\,S^2 + \lambda^2)\,\frac{\partial^2 u}{\partial S^2} - r\,u = 0. \#(4)$$

The goal is to find a DGM estimate of a European option price in hyperbolic sine LVM (3), compare it with known analytical option price [13] and evaluate the quality of approximation.

DGM ANN for PDE (4) was implemented with TensorFlow framework [14].

The computational experiment is performed with the following parameters: the number d of hidden layers is 3, the number N of nodes (neurons) per hidden layer is 50, the number of training stages is 100 with 10 SGD steps in each stage, and other parameters of the model are as follows $r = 0.05$, $\lambda = 0.25$, $K = 50$, $T = 1$, $S_0 = 0.5$. Quality of obtained ANN approximation is characterized by the following metrics:

- mean absolute error (MAE) is 0.2014;
- mean squared error (MSE) is 0.2483;
- coefficient of determination (R^2) is 99.93%.

The resulting error of approximation, i.e. difference between exact analytical option prices in hyperbolic sine LVM (3) [13] and option prices, predicted by DGM ANN, is visualized in Fig. 3.

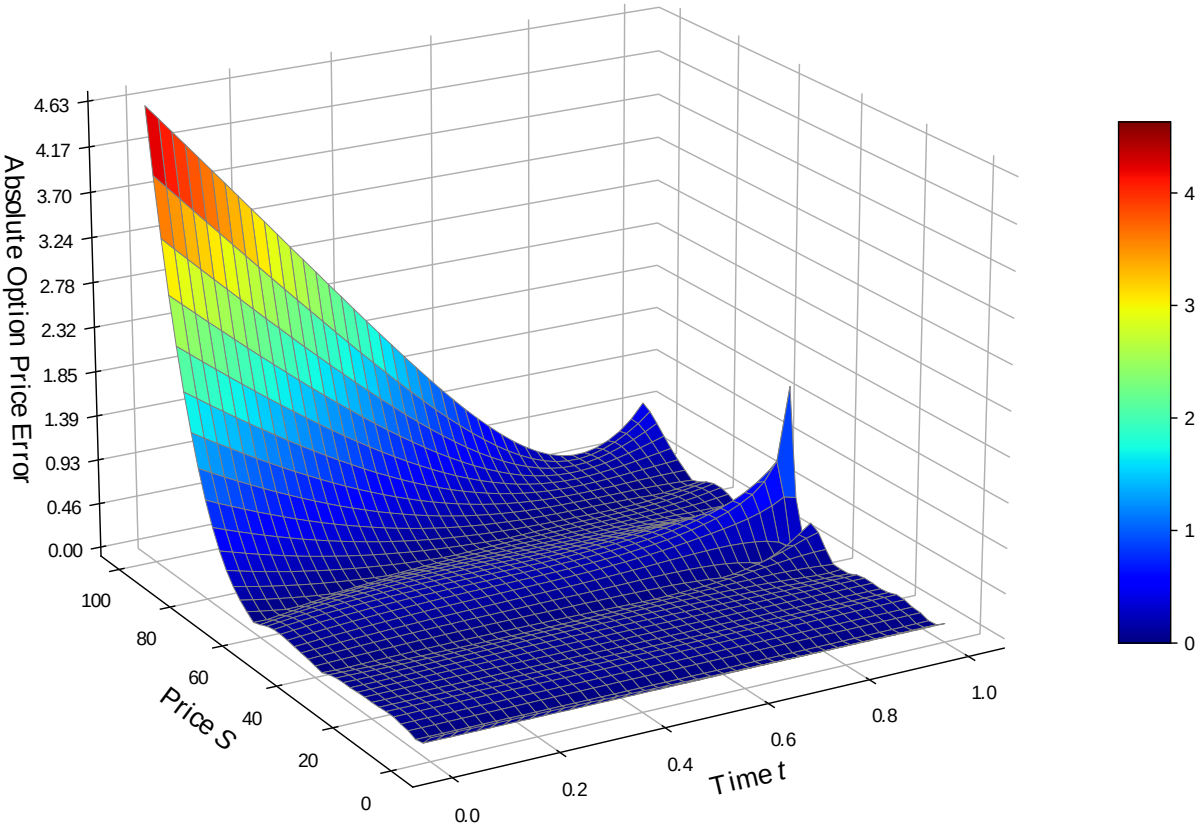

Figure 3. Absolute Error Surface of DGM Option Price Approximation

The computational experiment shows that the approximation, obtained with DGM ANN, predicts option prices in hyperbolic sine LVM (3) with acceptable accuracy, but the quality of approximation deteriorates for options ATM (At The Money) at expiry day and for options ITM (In The Money) with long time to maturity.

4. Conclusion and Future plans

Generally, option exchange trades various options on the same underlying with a range of exercise (strike) prices and expiry days, so to price all these options ANN shall receive as an input the set of price-time-strike price-expiry day points (S, t, K, T) instead of price-time points (S, t). This transition from input (S, t) to input (S, t, K, T) may require different architecture of ANN and another strategy of its training.

As noted in [15,16], the loss function in the form of energy functional (potential) is preferable for loss minimization, so construction of variational formulation for BSM PDE (1) can contribute to deep option pricing. Energy functional (potential) for BSM PDE (1) may be obtained using methods of the inverse problem of the calculus of variations [17].

The computational experiment with deep option pricing in hyperbolic sine LVM demonstrates that the algorithm converges to exact analytical European call option price of the LVM with acceptable accuracy, but oscillating behavior of the option price approximation makes it desirable to modify the neural network architecture for smoothing its output.

References

[1] B. Dupire. Pricing with a smile // Risk Magazine 7 (1) (1994) 18–20.

[2] F. Black, M. Scholes. The Pricing of Options and Corporate Liabilities // Journal of Political Economy 81 (3) (1973) 637–654. doi:10.1086/260062

[3] R. C. Merton. Theory of Rational Option Pricing // The Bell Journal of Economics and Management Science 4 (1) (1973) 141–183. doi:10.2307/3003143

[4] J. C. Cox, S. A. Ross. The valuation of options for alternative stochastic processes // Journal of Financial Economics 3 (1-2) (1976) 145–166. doi:10.1016/0304-405x(76)90023-4

[5] D. Brigo, F. Mercurio. Fitting volatility skews and smiles with analytical stock-price models, Seminar paper, Institute of Finance, University of Lugano (2000).

[6] G. E. Uhlenbeck, L. S. Ornstein. On the theory of the brownian motion // Phys. Rev. 36 (1930) 823–841. doi:10.1103/PhysRev.36.823

[7] S. Liu, C. W. Oosterlee, S. M. Bohte. Pricing Options and Computing Implied Volatilities using Neural Networks // Risks 7 (2019) 16. doi:10.3390/risks7010016

[8] J. Sirignano, K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations // Journal of Computational Physics 375 (2018) 1339–1364. doi:10.1016/j.jcp.2018.08.029

[9] L. Bottou, O. Bousquet. The tradeoffs of large scale learning, in: Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS'07, Curran Associates Inc., Red Hook, NY, USA (2007) 161–168.

[10] S. Hochreiter, J. Schmidhuber. Long short-term memory // Neural Computation 9 (8) (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735

[11] R. K. Srivastava, K. Greff, J. Schmidhuber. Training very deep networks, in: C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett (Eds.), Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada (2015) 2377–2385.

[12] D. P. Kingma, J. Ba. Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015). URL http://arxiv.org/abs/1412.6980

[13] S. Shorokhov, M. Fomin. Modeling of financial asset prices with hyperbolic-sine stochastic model, in: V. Sukhomlin, E. Zubareva (Eds.), Convergent Cognitive Information Technologies. Convergent 2018, Communications in Computer and Information Science, Springer, Cham 1140 (2020) Ch. 1, 3–10. doi:10.1007/978-3-030-37436-5_1

[14] M. Abadi, et al. TensorFlow: a system for large-scale machine learning, in: OSDI'16: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, 2015 (2016) 265–283.

[15] Y. Zhu, N. Zabaras, P.-S. Koutsourelakis, P. Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data // Journal of Computational Physics 394 (2019) 56–81. doi:10.1016/j.jcp.2019.05.024

[16] N. Geneva, N. Zabaras. Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks // Journal of Computational Physics 403 (2020) 109056. doi:10.1016/j.jcp.2019.109056

[17] V. M. Filippov, V. M. Savchin, S. G. Shorokhov. Variational principles for nonpotential operators // Journal of Mathematical Sciences 68 (3) (1994) 275–398. doi:10.1007/bf01252319