Available online at www.sciencedirect.com

ELSEVIER

Physica A 338 (2004) 160 – 165

PHYSICA A

www.elsevier.com/locate/physa

Pricing financial derivatives with neural networks

Marco J. Morelli[a], Guido Montagna[b,c,*], Oreste Nicrosini[b,c], Michele Treccani[b], Marco Farina[d], Paolo Amato[d]

[a]*FOM Instituut voor Atoom- en Molecuulfysica (AMOLF), Kruislaan 407, 1098 SJ Amsterdam, The Netherlands*
[b]*Dipartimento di Fisica Nucleare e Teorica, Università di Pavia, Via A. Bassi 6, 27100 Pavia, Italy*
[c]*Istituto Nazionale di Fisica Nucleare, sezione di Pavia Via A. Bassi 6, 27100 Pavia, Italy*
[d]*STMicroelectronics, SST Corporate R& D Via C. Olivetti, 2, 20041 Agrate Brianza (Milano), Italy*

Abstract

Neural network algorithms are applied to the problem of option pricing and adopted to simulate the nonlinear behavior of such financial derivatives. Two different kinds of neural networks, i.e. multi-layer perceptrons and radial basis functions, are used and their performances compared in detail. The analysis is carried out both for standard European options and American ones, including evaluation of the Greek letters, necessary for hedging purposes. Detailed numerical investigation show that, after a careful phase of training, neural networks are able to predict the value of options and Greek letters with high accuracy and competitive computational time.
© 2004 Elsevier B.V. All rights reserved.

PACS: 87.18.Sn; 89.65.Gh

Keywords: Econophysics; Option pricing; Neural networks

1. Introduction

A classical problem in financial engineering and econophysics is the development of efficient computational algorithms to price financial derivatives, and, in particular, options. There is a well developed framework, the Black–Scholes–Merton (BSM) stochastic model [1], for option pricing. In this model, which is a standard for the activity of financial practitioners, the price of a financial derivative is given by a deterministic partial differential equation. In principle, by solving this equation with the

* Corresponding author. Dipartimento di Fisica Nucleare e Teorica, Università di Pavia, Via A. Bassi 6, Pavia 27100, Italy. Tel.: +39-0382-507742; fax: +39-0382-526938.
E-mail address: guido.montagna@pv.infn.it (G. Montagna)

0378-4371/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.physa.2004.02.038

appropriate boundary conditions, the value of any derivative of interest can be obtained. In practice, the BSM equation can be analytically solved only in the simple case of a so-called European option. Actually, if we consider other financial derivatives, which are commonly traded in the financial market and allow anticipated exercise or depend on the details of the stochastic dynamics of the underlying asset (such as American, Asian and Barrier options), analytical solutions do not exist and numerical techniques are necessary. As discussed in literature [2], the standard numerical procedures are binomial trees, the method of finite differences and the Monte Carlo simulation of random walks. Very recently, alternative ideas have been proposed, especially in the econophysics literature, in order to develop novel efficient and precise pricing algorithms [3,4]. In the present study, we make use of the fast path integral approach developed in Refs. [4,5], in order to generate large samples of precise option values.

With these samples at our disposal, we exploit the features of neural network algorithms to capture the nonlinear behavior of option prices and Greek letters (i.e. the derivatives of option values), with the goal of developing a neural network tool for option pricing and hedging with a degree of accuracy and computational costs to be of practical use. It turns out that neural networks are a suitable tool for this task, because of their flexibility, reliability and the capacity of handling the multi-parametric dependence of the solutions.

2. Neural networks: multi-layer perceptrons and radial basis functions

Neural networks (NN) are known as universal approximators, since they can interpolate a large variety of unknown mappings, once a training sample $(\vec{x}_t, \hat{y}_t)$ is given. Many types of NN, characterized by different topologies, are available in literature [6,7], but for our purposes layered networks perform better and we will therefore concentrate only on them.

Tackling the issue from a formal point of view, the interpolation problem can be put in the form of minimizing the following functional:

$$H(\hat{f}) \equiv \sum_{t=1}^{T} (\|\hat{y}_t - \hat{f}(\vec{x}_t)\|^2 + \lambda \|\nabla f(\vec{x}_t)\|^2) , \tag{1}$$

where ∇ is the gradient operator and λ is a parameter accounting for the smoothness of the required solution. This reduces to minimize the distance between a given set of points $\hat{y}_t$ and the answer of the unknown function $\hat{f}(\vec{x}_t)$. To this end, we have critically compared two different NN classes: the multi-layer perceptron (MLP) and the radial basis function (RBF).

The chosen MLP is a single layer back-propagation network which employs the scaled conjugate gradient (SCG) technique to update the parameters of the network. This network can be summarized as follows:

$$\hat{f}(\vec{x}_t) = h\left\{ \sum_{i=1}^{k} c_i h(\beta_{0i} + \vec{\beta}'_{1i}\vec{x}_t) + c_0 \right\} , \tag{2}$$

where $\vec{x}_t$ are the input of the sample, $h(x)$ is a smooth, monotonic, increasing function, called *activation function*, which is in our case the sigmoid function $1/(1+e^{-x})$, $\{c\}$ and β_i are the parameters tuned by the SCG technique to minimize the error and k is the number of hidden units or *neurons*.

On the other hand, the RBF can be cast in the following form:

$$\hat{f}(\vec{x}_t) = \sum_{i=1}^{k} c_i h_i(\|\vec{x}_t - \vec{z}_i\|) + p(\vec{x}_t) , \tag{3}$$

where $\vec{z}_i$ are the centers of the various activation functions, c_i are the parameters to be optimized, $p(\vec{x})$ is a polynomial, h is the activation function (a radially symmetric Gaussian with variance σ_{RBF}) and k is the number of *neurons*, which is in our case equal to the size of the sample t.

It is important to stress that in both cases the free parameters c and β are managed by the supervised learning technique which is peculiar to the architecture, while for other parameters, such as the number of neurons in MLP and the variance of the Gaussian in the RBF, there is not a universal and well established procedure to optimize the performance and the accuracy of the results.

3. Numerical results

When dealing with NN modeling, the numerical analysis follows two phases: the *training phase* and the *generalization phase*.

In the first phase we used a variable size sample of the option prices, as calculated by means of the path-integral method described in Ref. [4] and applied to European and American options. The method gives predictions for option prices and Greek letters. Although the investigation has been carried out for European and American options, here we show results for European options only, for brevity.

With this path-integral sample the two networks are trained, in order to search for their optimal configuration, which is achieved when the error of the NN prediction is lower than a desired level of accuracy. For RBF this amounts to tune the Gaussian variance σ_{RBF}, while for MLP the training procedure has to be repeated for a number of times, known as epochs. However, as discussed in literature [7], this strategy has a drawback: if we exceed in the number of epochs, once we have stopped the training phase, i.e. we have "frozen" all the parameters, the network typically performs very well on the training sample, but the error is large when the network predicts new points not present in the training sample. This is the so called *overfitting phenomenon*. In order to minimize it, we take a sample wider than the training one, on which to test the performance of the network and to start the generalization phase. Once the number of neurons in the MLP architecture has been fixed, a minimum in the generalization error is reached for a particular epoch.

3.1. Two-dimensional case

The price of options is completely specified in terms of five parameters: the spot price S, the strike price X, the risk-free interest rate r, the volatility σ and the maturity (or

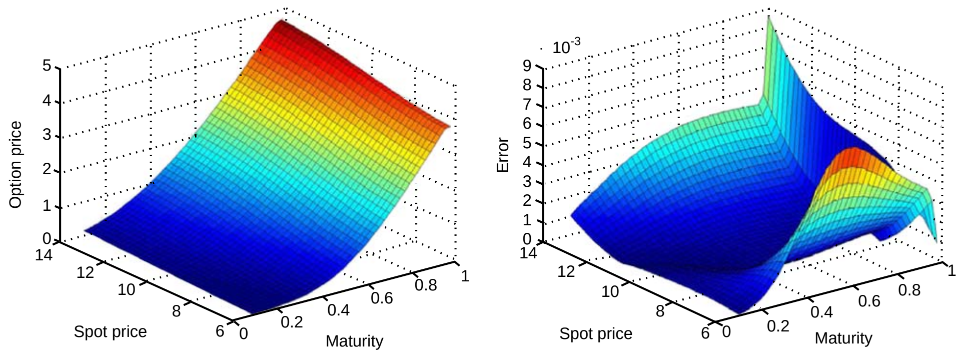

Fig. 1. Price of an European call option, as a function of the spot price and maturity and corresponding to $r = 0.1$, $X = 10$, $\sigma = 0.05$, simulated by a RBF Network (left) and the associated generalization error (right).

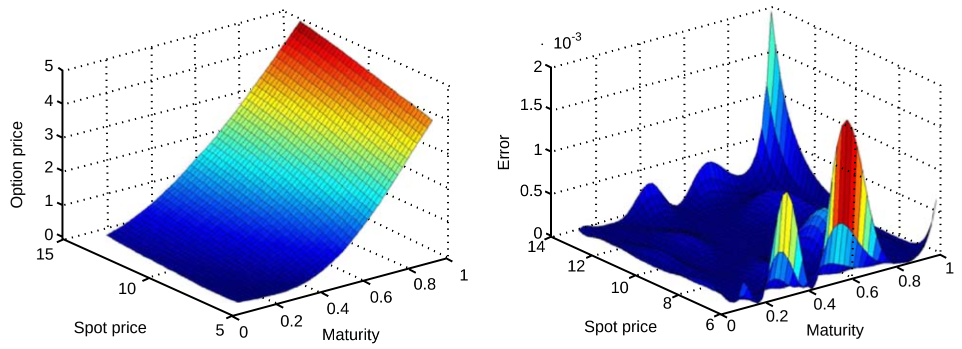

Fig. 2. Price of an European call option, as a function of the spot price and maturity and corresponding to $r = 0.1$, $X = 10$, $\sigma = 0.05$, simulated by a MLP network (left) and the associated generalization error (right).

expiration date) T. The preliminary studies about the networks' performances refer to a section of the price function $O(S, X, r, \sigma, T)$ and, specifically, to the two-dimensional domain defined by the parameters (S, T), keeping fixed the other three variables. This allows to provide a visual confirmation of the networks' behavior, as shown in Figs. 1 and 2 for the RBF and MLP, respectively.

In the two-dimensional case we used a training sample of 400 option values and the error obtained with the MLP is comparable with that of the RBF, being the average error at the level of a few 10^{-3}. The main difference between the two networks is in the CPU time required for training: on a standard PC, the MLP equipped with 12 neurons takes about 15 min to perform a training over 10^5 epochs, while the RBF, with a variance σ_{RBF} of 0.2, are trained very quickly in less than 1 min. Concerning RBF, we observe a steep rise of the generalization error close to the boundaries of the sample used in the training phase, as can be seen from Fig. 1. The reason for this effect, not present in the case of MLP, is the following: every RBF neuron has a

Table 1
Comparison between the NN multi-dimensional predictions for option price and Greek letters, as obtained by RBF with $\sigma_{\mathrm{RBF}} = 0.2$ and MLP with 50 neurons and 5×10^5 epochs

5D	Price error	Δ error	Θ error
RBF	9.7×10^{-3}	5.7×10^{-3}	7.4×10^{-3}
MLP	1.4×10^{-3}	3.1×10^{-4}	8.2×10^{-4}

Gaussian as activation function, which gives an accurate output only if the input lies within three standard deviations from the center. These "hats" overlap extensively, so that every part of the function is shared by many neurons. Close to the boundaries, a smaller number of Gaussians can cover the points, resulting in a poorer interpolation.

Besides the option price, other fundamental quantities of financial interest are the so-called Greek letters, which are the derivatives of the price with respect to one of the five parameters S, X, r, σ, T and are useful for hedging purposes. For the sake of simplicity, here we consider the Greek letters Δ and Θ, defined as $\Delta = \partial O/\partial S$ and $\Theta = \partial O/\partial t$. Since the path integral approach can give results also for the Greek letters, NN predictions for the Greek letters can be obtained according to two different procedures: we can set up a new MLP or RBF and perform the two phases of training and generalization directly on the values of the Greek letters, or we can keep the network trained on the option price and compute the derivatives numerically, treating the network itself as a function. We chose the second way, in order to evaluate the smoothness of the network interpolation and develop a unique NN tool for pricing and hedging. In this case the performance of the two classes differ dramatically: the RBF can catch the smoothness much better than the MLP architecture with 12 neurons. However, as hinted in Ref. [8], the MLP performances can be improved by increasing the number of neurons in order to let the SCG technique work in a wider space of parameters. Actually, it has been checked that almost doubling the number of neurons the MLP performance in the simulation of the Greek letters improved up to the RBF ones, provided the number of epochs is increased together with the computational effort.

3.2. Multidimensional case

To simulate the multidimensional dependence of option prices and Greek letters from the parameters S, X, r, σ, T, the size of the training sample has to be considerably increased in order to cover most of the regions in the whole set of five parameters and obtain, in turn, accurate predictions.

Starting from a sample of 4000 events, the performances of the two architecture significantly differ in the price prediction, being the price error at the 10^{-2} level for the RBF and at the 10^{-3} level for the MLP, as shown in Table 1. When considering the RBF, the only way to further reduce the error is to tune the σ_{RBF} parameter, even if this procedure turns out to be useless whenever the size of the sample is insufficient. This is not the case of the MLP, where it is enough to increase the number of neurons, and consequently the epochs, in order to reach the minimum in the generalization error. As it can be noticed from Table 1, the RBF shows its weak side in the predictions for the Greek letters, being its generalization error one order of magnitude larger than the one of the optimized MLP, obtained with 50 neurons and 5×10^5 epochs. To obtain the results discussed above, the computational effort required by the training phase is of the order of some minutes for the RBF and of more than 4 h for the MLP.

It is important to stress that, once the network is properly trained, the prediction phase takes only a few seconds both for MLP and RBF, since the networks execute only simple operations and the signal has only to be propagated forward once. All these observations led us to turn our attention to the MLP architecture for further optimization strategies (such as evolutionary algorithms), being the MLP the most versatile NN, even if not so fast in the training phase.

4. Conclusions

We applied NN to the problem of option pricing, in order to capture the nonlinear behavior of option prices and Greek letters with a high degree of accuracy and competitive computational time. It has been shown that NN are a suitable tool for such a task, being able to handle the complete multi-parametric dependence of the financial derivatives. More specifically, RBF networks proved to work with more limited performances, but they can be trained very quickly and can be used for preliminary checks, while MLP, which require a longer phase of training, turn out to be at the end a rather robust tool, especially in the modeling of the Greek letters.

Future developments concern the use of more sophisticated algorithms, in particular evolutionary strategies, for the optimization of the MLP architecture and search for optimal configuration. The application of NN algorithms to pricing exotic options, as well as to pricing options within non-Gaussian models of financial dynamics, is also under inspection.

References

[1] F. Black, M. Scholes, J. Pol. Econ. 72 (1973) 637;
R. Merton, Bell J. Econ. Manag. Sci. 4 (1973) 141.
[2] J.C. Hull, Options, Futures, and Other Derivatives, 4th Edition, Prentice-Hall, Englewood Cliffs, NJ, 2000.
[3] M. Rosa-Clot, S. Taddei, E. Bennati, Int. J. Theor. Appl. Finance 2 (1999) 381;
M. Rosa-Clot, S. Taddei, Int. J. Theor. Appl. Finance 5 (2002) 123.
[4] G. Montagna, O. Nicrosini, N. Moreni, Physica A 310 (2002) 450.
[5] G. Montagna, M. Morelli, O. Nicrosini, P. Amato, M. Farina, Physica A 324 (2003) 189.
[6] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice-Hall, Englewood Cliffs, NJ, 1999.
[7] C.M. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.
[8] J.M. Hutchinson, A. Lo, T. Poggio, A nonparametric approach to pricing and hedging derivative securities via learning networks, MIT A.I. Memo No. 1471 and C.B.C.L. Paper No. 92, April 1994.