Robust Artificial Neural Networks for Pricing of European Options

Panayiotis C. Andreou, Chris Charalambous[1], and Spiros H. Martzoukos

<u>This version</u>: October 2005

Abstract

The option pricing ability of Robust Artificial Neural Networks optimized with the Huber function is compared against those optimized with Least Squares. Comparison is in respect to pricing European call options on the S&P 500 using daily data for the period April 1998 to August 2001. The analysis is augmented with the use of several historical and implied volatility measures. Implied volatilities are the overall average, and the average per maturity. Beyond the standard neural networks, hybrid networks that directly incorporate information from the parametric model are included in the analysis. It is shown that the artificial neural network models with the use of the Huber function outperform the ones optimized with least squares.

Keywords: Option pricing & trading, implied parameters, artificial neural networks, Huber function, robust estimation.

JEL classification: G13, G14.

[1] University of Cyprus, Department of Public and Business Administration, P.O. Box 20537, CY 1678 Nicosia, Cyprus, Fax: +357-22892460, tel.: +357-22892461/2, Email: bachris@ucy.ac.cy.

1. Introduction

The scope of this work is to compare alternative feed-forward Artificial Neural Network (ANN) configurations in respect to pricing the S&P 500 European call options. Robust ANNs that use the Huber function are developed, and configurations that are optimized based solely on the least squares norm are compared with robust[1] configurations that are closer to the least absolute norm. The data for this research come from the New York Stock Exchange (NYSE) for the S&P 500 equity index and the Chicago Board of Options Exchange (CBOE) for call option contracts, spanning a period from April 1998 to August 2001.

Black and Scholes introduced in 1973 their milestone (BS) formula which is still a most prominent conventional Options Pricing Model (OPM). The options we price are on the S&P 500 index, which is extremely liquid and is the closest to the theoretical setting of the Black and Scholes model (Garcia and Gencay, 2000). Empirical research in the last three decades has shown that the formula suffers from *systematic biases* for various reasons (for details see Black and Scholes, 1975, Rubinstein, 1985, Bates, 1991 and 2003, Bakshi et al., 1997, Andersen et al., 2002, and Cont and Fonseca, 2002). Despite this, BS is frequently used to price European options[2] mainly because alternative parametric models (e.g. stochastic volatility, jump-diffusion, stochastic interest rates, etc.) have drastically failed to provide results truly consistent with the observed market data. Additionally, these models are often too complex to implement and be used for real trading (see Bakshi et al., 1997). On the other hand, artificial neural networks are promising alternatives to the parametric OPMs; they do not necessarily rely on any financial theory and are trained inductively using historical or implied input variables and option transactions data. Their popularity is constantly increasing, and contemporary financial econometric textbooks (e.g. Tsay, 2002) dedicate special sections or even whole chapters to this topic.

It is well known that market participants change their option pricing attitudes from time to time (i.e. Rubinstein, 1985), so a parametric model might fail to adjust to such rapidly changing market behavior. ANNs can potentially correct the aforementioned BS bias (Hutchison et al., 1994, Lajbcygier et al., 1996, Garcia and Gencay, 2000, Yao and Tan, 2000). Neural networks trained on the least squares error criterion are highly influenced by outliers, especially in the presence of non-Gaussian noise (Bishop, 1995). Options data are known to be heavily influenced *at least* by noise due either to thin trading or to abnormal volume trading (Long and

Officer, 1997, and Ederington and Guan, 2005) and exhibit a strong time-varying element (Dumas et al., 1995, and Cont and Fonseca, 2002). Consecutively, robust estimation is expected to improve out-of-sample pricing of options.

In previous empirical research on option pricing, ANNs have been optimized based on the l_2 (the least squares) norm. The l_2 norm is a convenient way to train ANNs, since ready to use statistical packages are widely available for this purpose. Of course, the least squares optimization is highly susceptible to the influence of large errors since some abnormal datapoints (or few outlier observations) can deliver non-reliable networks. On the contrary, robust optimization methods that exploit the l_1 (the least absolute) norm are unaffected by large (or catastrophic) errors but are doomed to fail when dealing with small variation errors (e.g. Bandler et al., 1993, and Devabhaktuni et al., 2001, for applications in the electrical engineering field).

Here the Huber function (Huber, 1981) is used as the error penalty criterion during the ANNs optimization process to immunize the adaptable weights in the presence of data-point peculiarities. The Huber function utilizes the robustness of l_1 and the unbiasedness of l_2 and has proved to be an efficient tool for robust optimization problems for various tasks (Bandler et al., 1993, Jabr, 2004, Chang, 2005), albeit it does not constitute the mainstream. The training of ANNs with the Huber technique has recently gained attention in electrical engineering (i.e. Devabhaktuni et al., 2001, Xi et al., 1999), but to our knowledge has not gained attention in options pricing, where it is possible to observe both small and large errors for a variety of reasons (e.g. Bakshi et al., 1997). Our choice of the Huber function is because it is widely referenced on robust estimation (Bishop, 1995), it provides a simple generalization of the least squares approach, it avoids the need for any probabilistic assumptions, and it is not difficult to implement with neural networks. Comparison with other estimators, like the *MM* estimators (Yohai, 1987), the *S* estimators (Rousseeuw and Yohai, 1984), and the redescenting estimators (Morgenthaler, 1990), is beyond the scope of this work, but can be part of future research.

The standard ANN target functions are employed that are comprised by the market value of the call option standardized with the strike price. Furthermore, the hybrid ANN target function suggested by Watson and Gupta (1996) and used for pricing options with ANNs in Lajbcygier et al. (1997) and Andreou et al. (2005) are examined. In the hybrid models the target function is the residual between the actual call market price and the parametric option price estimate standardized with the strike price. It can capture the potential misspecification of the BS assumptions of geometric Brownian motion (see for example, Lim et al., 1997). Unlike Hutchison et al. (1994), in the parametric as well as in the nonparametric models both historical and implied volatility measures are used. To train the ANNs, the modified *Levenberg-Marquardt* (LM) algorithm which is efficient in terms of time capacity and accuracy (Hagan and Menhaj, 1994) is utilized. In contrast to many previous studies, thorough cross-validation allows the use of a different network configuration in different testing periods.

In the following, first the parametric BS model, and the standard and hybrid ANN model configuration with the Huber function and with least squares (mean square error to be precise) are reviewed. Then, the dataset, and the historical and implied parameter estimates are discussed, and the parametric and ANN models are defined according to the parameters used. Subsequently, the numerical results are reviewed with respect to the in- and out-of-sample pricing errors; the economic significance of dynamic trading strategies both in the absence and in the presence of transaction costs is also discussed. The final section concludes. It is demonstrated that with the use of the Huber function, ANNs outperform their counterparts optimized with least squares. The best (hybrid and standard) ANN models with the Huber function are identified, and evidence is provided that, even in the presence of transaction costs, profitable trading opportunities still exist.

2. Option Pricing Models: The parametric BS and ANNs

2.1. The Black and Scholes option pricing model

The Black Scholes formula for European call options modified for dividend-paying underlying asset is:

$$c^{BS} = Se^{-\delta T} N(d_1) - Xe^{-rT} N(d_2), \tag{1}$$

with

$$d_1 = \frac{\ln(S/X) + (r - \delta)T + (\sigma\sqrt{T})^2/2}{\sigma\sqrt{\mathrm{T}}}, \tag{1.a}$$

and

$$d_2 = d_1 - \sigma\sqrt{T}. \tag{1.b}$$

In the above, $c^{BS} \equiv$ *estimated premium for the European call option*; $S \equiv$ *spot price of the underlying asset*; $X \equiv$ *exercise price of the option*; $r \equiv$ *continuously compounded riskless interest rate*; $\delta \equiv$ *continuous dividend yield paid by the underlying asset*; $T \equiv$ *time left until the option expiration*; $\sigma^2 \equiv$ *yearly variance rate of return for the underlying asset*; $N(.) \equiv$ *the standard normal cumulative distribution.*

The standard deviation of continuous returns (σ) is not observed and an appropriate forecast should be used. The literature has used both *historical* and *implied volatility* forecasts. Contrary to the historical estimates, the implied volatility forecasts have desirable properties that make them attractive to practitioners: they are forward looking and avoid the assumption that past volatility will be repeated. In this study, similarly with Hutchison et al. (1994) and in addition to the other volatility measures, the 60 days historical volatility which is a widely used historical volatility benchmark is also employed.

If BS is a well-specified model, then all implied volatilities of the same underlying asset should be the same or at least some deterministic functions of time. Unfortunately, this is far from being empirically true. For example, Rubinstein (1985) has shown that the implied volatilities derived via BS as a function of the moneyness ratio (*S/X*) and time to expiration (*T*) often exhibit a **U** shape, known as the *volatility smile*. This is why BS is usually referred to as being a misspecified model with an inherent source of bias (see also Latane and Jr., 1976, Bates, 1991, Canica and Figlewski, 1993, Bakshi et al., 2000, and Andersen et al., 2002). Under the existence of this anomaly, any historical volatility measure is doomed to fail, while measures (like the implied ones) that mitigate this bias could perform better.

2.2. Neural networks

A neural network is a collection of interconnected simple processing elements structured in successive layers and can be depicted as a network of arcs/connections and nodes/neurons. The network has the input layer, one or more hidden layers and an output layer. Each interconnection corresponds to a numerical value named weight, which is modified according to the faced problem via an optimization algorithm. The particularity of ANNs relies on the fact that the neurons on each layer operate collectively and in a parallel manner on all input data and that each neuron behaves as a summing vessel that works, under certain conditions, as a non-linear mapping junction for the forward layer.

Figure 1 depicts an ANN architecture similar to the one applied for the purposes of this study. This network has three layers: an *input* layer with *N* input variables, a *hidden* layer with *H* neurons, and a *single* neuron *output* layer. Each neuron is connected with all neurons in the previous and the forward layer. Each connection is associated with a *weight*, w_{ki}, and a *bias*, w_{k0}, in the hidden layer and a *weight*, v_k, and a *bias*, v_0, in the output layer (k=1,2,...,H, i=1,2,...N). In addition, the outputs of the hidden layer ($y_1^{(1)}, y_2^{(1)} .. y_H^{(1)}$) are the inputs for the output layer.

Inputs are set up in feature vectors, $\tilde{x}_q = [x_{1q}, x_{2q} \ldots, x_{Nq}]$ for which there is an associated and known target, t_q, $q \equiv 1,2,\ldots,P$, where P is the number of the available sample feature vectors for a particular training sample. According to Figure 1, the operation carried out for computing the final estimated output, y, is the following:

$$y = f_0[v_0 + \sum_{k=1}^{H} v_k f_H(w_{k0} + \sum_{i=1}^{N} w_{ki} x_i)], \tag{2}$$

where f_0 and f_H are the transfer functions associated with the output and hidden layers respectively.

[**Figure 1, here]**

For the purpose of this study, the hidden layer always uses the hyperbolic tangent sigmoid transfer function, while the output layer uses a linear transfer function. In addition, ANN architectures with only one hidden layer are considered since research has shown that this is adequate in order to approximate most functions arbitrarily well. This is based on the universal approximation theorem provided by Cybenko (1989) (for further details see also Haykin, 1999):

Let $f_H(.)$ be a non-constant, bounded and monotone-increasing continuous function. Let l_N denote the N-dimensional unit hypercube $[0,1]^N$. The space of continuous functions on l_N is denoted by $C(l_N)$. Then, given any function $g \in C(l_N)$ and $\varepsilon > 0$, there exist an integer number H and sets of real constants,

$$w_{k0}, w_{ki}, v_k,\ k = 1,2,\ldots,H,\ i = 1,2,\ldots,N$$

such that we may define, $y(x) = \sum_{k=1}^{H} v_k f_H(w_{k0} + \sum_{i=1}^{N} w_{ki} x_i)$ as an approximate realization of the function $g(.)$; that is, $| y(x) - g(x) | < \varepsilon$ for all vectors x that lie in the input space.

Training ANNs is a non-linear optimization process in which the network's weights are modified according to an error function. For the case that the ANN model has only one output neuron, the error function between the estimated response y_q and the actual response t_q is defined as:

$$e_q(w) = y_q(w) - t_q\,, \tag{3}$$

where, w is an n-dimensional column vector containing the weights and biases given by: $w = [w_{10} \ldots, w_{H0}, w_{11}, \ldots, w_{HN}, v_0, \ldots, v_H]^T$. The Huber function that is used to optimize the trainable parameters w is defined as (i.e. Huber, 1981, Bandler et al., 1993):

$$E(w) = \sum_{q=1}^{P} \rho_k(e_q(w))\,, \tag{4}$$

where ρ_k is the Huber function defined as:

$$\rho_k(e) = \begin{cases} 0.5e^2 & \text{if } |e| \le k \\ k|e| - 0.5k^2 & \text{if } |e| > k \end{cases}, \tag{5}$$

where k is a positive constant. It is obvious that when $|e| > k$ the Huber function treats the error in the l_1 sense and in the l_2 sense only if $|e| \le k$ depending on the value of threshold parameter k. Figure 2 depicts the Huber function along with the least absolute (l_1) and least squares (l_2) error functions. The Huber function has a smooth transition between the two norms at $|e| = k$, so that the first derivative of ρ_k is continuous everywhere.

The choice of k defines the threshold between *large* and *small* errors. Different values of k determine the proportion of the errors to be treated in the l_1 or the l_2 norm. As seen, when k is sufficiently large the Huber function encompasses the widely used and conventional least squares (l_2) training of the ANNs. As the k parameter approaches zero, the Huber function approaches the l_1 function and the errors are penalized in the least absolute sense. Figure 2 makes obvious that the Huber function should be more robust to abnormal data since it penalizes them less compared to the l_2 norm.

[Figure 2, here]

The nice properties of the Huber function compared to the l_2 norm are more distinct when they are compared according to their gradient vector. The gradient vector of the least squares error function is:

$$\nabla E_{l_2}(w) = \sum_{q=1}^{P} e_q \nabla e_q(w), \tag{6}$$

whilst the gradient for the Huber function is:

$$\nabla E(w) = \sum_{q=1}^{P} \zeta_q \nabla e_q(w), \tag{7}$$

where,

$$\zeta_q = \frac{\partial \rho_k(e_q)}{\partial e_q} = \begin{cases} e_q & if\ |e_q| \le k \\ -k & if\ e_q < k \\ +k & if\ e_q > k \end{cases}. \tag{8}$$

The $P \times n$ Jacobian matrix, $J(w)$, of the *P-dimensional* output error column vector is given by:

$$J(w) = \begin{bmatrix} \nabla e_1^T(w) \\ \vdots \\ \nabla e_P^T(w) \end{bmatrix}. \tag{9}$$

Using this notation, (7) can be written in the form:

$$\nabla E(w) = J(w)^T \zeta(w), \tag{10}$$

where ζ is a *P-dimensional* column vector with elements the ζ_q values.

The quantity $\nabla e_q(w)$ is the gradient vector of $e_q(w)$ with respect to the trainable parameter vector w. The difference between (6) and (7) depends on the weighting factor of the $\nabla e_q(w)$. The weighting factor of $\nabla e_q(w)$ for the Huber gradient is the same with the least squares gradient only when $|e_q| \le k$. In all other cases the weighing factor for the Huber gradient is held constant at the value of the threshold k unlike in the l_2 case that gives more weight to large errors. This is how the Huber function immunizes against the influence of large errors.

Moreover, the Hessian matrix in the case of the Huber function is given by:

$$\nabla^2 E(w) = \sum_{q=1}^{P} d_q \nabla e_q(w) \nabla e_q(w)^T + \sum_{q=1}^{P} \zeta_q \nabla^2 e_q(w) , \tag{11}$$

where

$$d_q = \frac{\partial^2 \rho_k(e_q)}{\partial e_q{}^2} = \begin{cases} 1 & if\ |e_q| \le k \\ 0 & if\ |e_q| > k \end{cases} . \tag{12}$$

The quantity $\nabla e_q(w)$ is computed based on the back-propagation algorithm that is commonly used in the context of feed-forward ANNs. Based on the neural network depicted in Figure 1, the partial derivative of the error function (3) with respect to the weight v_k at the hidden layer is:

$$\frac{\partial e_q}{\partial v_k} = y_k^{(1)} f_0'(\psi) , \tag{13}$$

where $f_0'(\psi)$ is the differential of the output neuron transfer function at point ψ. Since a linear transfer function is used at the output neuron, the $f_0'(\psi)$ is equal to unity. Furthermore, the partial derivative of the error function (3) with respect to the weight w_{ki} at the input layer is:

$$\frac{\partial e_q}{\partial w_{ki}} = x_i f_H'(\psi_\kappa^{(1)}) v_k f_0'(\psi) , \tag{14}$$

where $f_H'(\psi_\kappa^{(1)})$ is the differential of the transfer function associated with the k^{th} hidden neuron at point $\psi_\kappa^{(1)}$. For our case, we always use the hyperbolic tangent as a transfer function:

$$f_H(a) = \frac{2}{1 + e^{-2a}} - 1 \equiv tanh(a). \tag{15}$$

The differential of this function with respect to a can be expressed in a particularly simple form:

$$f'_H(a) = 1 - (f_H(a))^2 \,. \tag{16}$$

To optimize the weights, the modified *Levenberg-Marquardt* (LM) algorithm is employed. According to LM, the weights and the biases of the network are updated in order to minimize $E(w)$. At each iteration τ of the LM, the weights vector w is updated as follows:

$$w_{\tau+1} = w_\tau - [G_\tau + \mu_\tau I]^{-1} J(w_\tau)^T \zeta(w_\tau)\,, \tag{17}$$

where G is an *approximation* of the $n \times n$ Hessian matrix defined as:

$$G = \sum_{q=1}^{P} d_q \nabla e_q(w) \nabla e_q(w)^T \,, \tag{18}$$

and d_q is defined in (12). The matrix G is obtained from the Hessian matrix by deleting its second term which is usually considered small. Moreover, I is an $n \times n$ identity matrix, $J(w_\tau)$ is the Jacobian matrix at the τ^{th} iteration, and μ_τ is like a learning parameter that is automatically adjusted in each iteration in order to secure convergence. Large values of μ_τ lead to directions that approach the steepest descent, while small values lead to directions that approach the Gauss-Newton algorithm. Further technical details about the implementation of LM can be found in Hagan and Menhaj (1994) and Hagan et al. (1996). Based on (17), weights and biases update takes place in a batch mode manner where update occurs only when all input vectors have been presented to the network.

In addition to the standard ANNs with $t \equiv c^{mrk} / X$ (call market values standardized with their strike price), hybrid ANNs according to which the target function is the residual between the actual call market price and the BS call option estimation $t \equiv c^{mrk} / X - \hat{c}^{\Theta} / X$ (again standardized with the strike price) are also investigated, where $\hat{c}^{\Theta}$ should define a pricing estimate taken by the BS under a certain volatility forecast (this is explained further in the following section). For effective training, the input/output variables are scaled using the z-score transformation $\tilde{z} = (\tilde{x} - \mu)/s$, where $\tilde{x}$ is the vector of an input/output variable, μ is the mean and s the standard deviation of this vector. Moreover, the network initialization technique proposed by Nguyen and Windrow (see Hagan et al., 1996) is utilized that generates initial weights and bias values for a nonlinear transfer function so that the active regions of the layer's neurons are distributed roughly evenly over the input space.

For a given set of training data and for a given value of the Huber *k* value, the optimal number of hidden neurons is chosen via a cross-validation procedure. ANN structures with 2 to 10 hidden neurons are trained, and the one that performs the best in the validation period is selected. Since the initial network weights affect the final network performance, for a specific number of hidden neurons and Huber *k* value, the network is initialized, trained and validated five separate times. Huber (1981) gives a formula for deriving the optimal *k* value, but this formula was not derived with applications of neural networks in mind. Most importantly, restrictive probabilistic assumptions (of symmetrically contaminated Gaussian distributions) are made. In addition, (as pointed out also in Koenker, 1982, p. 232), we need to know the degree of contamination (i.e., the percent of abnormal observations). With neural networks we neither make any probabilistic assumptions, nor we know a priori the degree of contamination. Thus, we follow an empirical approach. The optimal *k* value is shown from the data after investigating a wide range of potential values. Different networks are developed for the following Huber *k*-values: 0.1, 0.2, 0.30, 0.40, 0.5, 0.60, 0.70, 0.80, 0.90, 1, 1.5, 2 and *Inf* (that corresponds to the optimization of the ANNs based on the l_2 norm). After defining the optimal ANN structure, its weights are frozen and its pricing capability is tested (out of sample) in a third separate *testing dataset* in order to verify the ANNs ability to generalize to unseen data.

3. Data, parameter estimates (historical and implied), and model implementation

The dataset covers the period from April 1998 to August 2001. The S&P 500 Index call options are considered because the CBOE option market is extremely liquid and these index options among the most popular. This market is the closest to the theoretical setting of the parametric models (Garcia and Gencay, 2000). Our prices are closing quotes. The majority (around 75%) of our call options lies in the +/-15% moneyness area. As suggested by Day and Lewis (1988), because trading volume tends to concentrate in the options that are around at-the-money and just in-the-money, any lack of synchronization between closing index level and the closing option price will be minimized for these options (pg. 107). Of course, it is not the first time that non intra-day option and index prices are used in analysis (see for example, Day and Lewis, 1988, Hutchison et al., 1994, Ackert and Tian, 2001, and Ederington and Guan, 2005). Specifically, Ackert and Tian (2001) argue that closing prices, which are non-synchronous, constitute the best alternative solution to examine the options arbitrage violations for the S&P index. Kamara and Miller (1995) compare intraday and closing option pricing results for market efficiency tests and argue that closing option prices are appropriate for analysis because they are representative of the transaction prices that prevailed during the day. This suggests that it is not unreasonable to use closing data in empirical options research. In our case, the Huber function is helpful in treating the options data according to the noise level.

Along with the index, a daily dividend yield, δ , is collected (provided online by Datastream). After applying various filtering rules, the dataset consists of 64,627 data points, with an approximate average of 35,000 data points per sub-period. Hutchison et al. (1994) have an average of 6,246 data points per sub-period. Lajbcygier et al. (1996) include in total 3,308 data points, Yao et al. (2000) include in total 17,790 data points, and Schittenkopf and Dorffner (2001) include 33,633 data points.

3.1. Observed and historically estimated parameters

The moneyness ratio, *S/X*, is the basic input to be used with ANNs since it is highly related with the pricing bias associated with the BS. The moneyness ratio *S/X* is calculated and used with ANNs like in Hutchison et al. (1994) (see also Garcia and Gencay, 2000). The dividend adjusted moneyness ratio $(Se^{-\delta T})/X$ is preferred here since dividends are relevant. In addition, the time to maturity (T) is computed assuming 252 days in a year.

Previous studies have used 90-day T-bill rates as approximation of the interest rate. In this study we use nonlinear cubic spline interpolation for matching each option contract with a continuous interest rate, r , that corresponds to the option's maturity. For this purpose, the 3-month, 6-month and one-year T-bill rates collected from the U.S. Federal Reserve Bank Statistical Releases are used.

Moreover, the 60-days volatility is a widely used historical estimate (see Hutchison et al., 1994, and Lajbcygier et al., 1997). This estimate is calculated using all the past 60 log-relative index returns and is symbolized as σ_{60} . In addition, the VIX Volatility Index is an estimate that can be directly observed from the CBOE. It was developed by CBOE in 1993 and is a measure of the volatility of the S&P 100 Index and is frequently used to approximate the volatility of the S&P 500 as well. In our dataset the 30-day returns of the two indexes were strongly correlated (with Pearson correlation coefficient between 0.94 and 0.99). VIX is calculated as a weighted average of S&P 100 option with an average time to maturity of 30 days and emphasis on at-the-money options. This volatility measure is symbolized as σ_{vix}.

3.2. Implied parameters

The Whaley's (1982) simultaneous equation procedure is adopted to minimize a price deviation function with respect to the unobserved parameters. For a given day the optimal implied parameter values correspond to the solution of an unconstrained optimization problem that minimizes the sum of squares residuals between the actual call option market values and the BS estimates. The optimization is done via a non-linear least squares optimization based on the Levenberg-Marquardt algorithm. His approach is an alternative to using the methodology proposed by Chiras and Manaster (1978) (CM), or Latane and Rendleman (1976) (LR). His reasoning is that: "rather than explicitly weighting the implied standard deviations of a particular stock where the weights are assigned in an ad-hoc fashion, the call prices are allowed to provide an implicit weighting scheme that yields an estimate of standard deviation which has little prediction error as is possible" (pg. 39). Bates (1996b) remarks that the Whaley's (1982) least squares weighting scheme effectively assigns heavier weights on the near the money options than CM and LR. His approach is widely applied even in more recent research; for instance Bakshi et al. (1997). Nevertheless, we tried these two weighting schemes (the CM and the modified LR as recommended by CM), and at least in our dataset the results are inferior to those of the overall average approach (or its per-maturity variant). The per-maturity versions worked even better since they can capture time-varying volatility effects (Bakshi, 1997, and Bates, 2003).

Similarly to Bakshi et al. (1997), two different implied volatility measures are taken from the above procedure. The first optimization is performed by including all available options transaction data in a day to obtain *daily average* implied parameters (σ_{av}). Second, *daily average per maturity* parameters (σ_{avT}) can be obtained by fitting the BS to all options that share the same maturity date as long as four different available call options exist.

For pricing and trading reasons at time instant *t*, the implied structural parameters derived at day *t*-1 are used together with all other needed information (*S*, *T*, *δ*, *X* and *r*). The same reasoning holds for the historical (σ_{60}) and the weighted implied average (σ_{vix}) estimates.

It is known that ANN input variables should be presented in a way that maximizes their information content (Garcia and Gencay, 2000). When options are priced, the parametric OPM formulas adjust those values to represent the appropriate value that corresponds to an option's expiration period. According to this rationale, for use with the ANNs, the volatility measures are transformed by multiplying each of the yearly volatility forecast with the square root of each option's maturity ($\tilde{\sigma}_j = \sigma_j \sqrt{T_{252}}$, where *j*={*60, vix, av, avT* }). They are named *maturity (or expiration) adjusted volatilities*. Also, and following the advice by a referee, we have constructed tables (not included for brevity) for all nine sub-periods (in several moneyness and maturity ranges) in order to compare between the volatilities of the training and the volatilities of the testing sub-periods. If these estimates differ considerably, this may imply saturation of the neural network with poor performance as a consequence. On average, we have observed no volatility jumps. Furthermore, the superior out-of-sample performance of the neural networks (see section 4) is additional evidence that the saturation problem mentioned by the referee does not seem to be present.

3.3. Output variables, filtering and processing

For training ANNs, the call standardized by the striking price, c_q^{mrk} / X_q, is used as one target function to be approximated. In addition, the hybrid structure is implemented, where the target function represents the pricing error between the option's market price and the parametric models estimate, $c^{mrk} / X - \hat{c}^{\Theta} / X$, where Θ is one of BS_{60}, BS_{vix}, BS_{av}, and BS_{avT}. In both cases, target residuals are standardized using the mean-variance scaling; hence the output neuron transfer function is linear.

Before filtering, more than 90,000 observations were included for the period April 1998 – August 2001. The final dataset consists of 64,627 datapoints. The filtering rules adopted are: *i) eliminate an observation if the call contract price is greater than the underlying asset value*; *ii) exclude an observation if the call* *moneyness ratio is larger than unity and the call price is less than its dividend adjusted lower bound; iii) eliminate all the options observations with time to maturity less than 6 trading days* (adopted to avoid extreme option prices that are observed due to potential illiquidity problems); *iv) price quotes lower than 0.5 index points are not included*; *v) maturities with less than four call option observations are also eliminated; vi)* in addition, to remove impact from thin trading observations are eliminated according to the rule: *eliminate an observation if the call option price at day t is the same as with day t*-1 *and if the open interest for these days stays unchanged and if the underlying asset has changed.* We filter data when we believe that they are "bad data" (filtering rules *i*, *ii*, *iv*, *vi*), or that they come from a different "data generating process" (filtering rule *iii*, following Bakshi et al., 1997). Filtering rule *v* was perceived as necessary in order to get an average volatility per maturity (Bakshi et al., 1997, recommend no less than two observations).

3.4. Validation and testing, and pricing performance measures

The available data are partitioned into training, validation and testing datasets using a chronological splitting, and via a rolling-forward procedure. Our dataset is divided into nine overlapping sub-periods in chronological order. Each sub-period is divided into a training (*Tr*), a validation (*Vd*) and a testing (*Ts*) set, again in chronological sequence. In each sub-period the training, validation and testing sets are non-overlapping. The nine testing sets are non-overlapping and they cover completely the last 20 months of the dataset.

There are *M* available call option contracts, for each of which there exist Ξ_m observations taken in consecutive time instances *t*, resulting in a total of *P* ($P = \sum_{m=1}^{M} \Xi_m$) available call option datapoints (*P* is the total number of call option transactions that cover the whole period and is equal to 64,627). To determine the pricing accuracy of each model's estimates, $\hat{c}$, the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) are examined:

$$RMSE = \sqrt{1/p\sum_{q=1}^{p}(c_q^{mrk} - \hat{c}_q)^2}\ , \tag{19}$$

$$\text{MAE} = 1/p\sum_{q=1}^{p}|\ c_q^{mrk} - \hat{c}_q\ |\ , \tag{20}$$

where $p \leq P$ indicates the number of observations used per case. These error measures are computed for an aggregate testing period (*AggTs*) with 35,734 (so *p* is equal to 35,734) datapoints by simply pooling together the pricing estimates of all nine testing periods. For *AggTs*, the Median Absolute Error (MdAE) as well as the 5th (5th APE) and 95th (95th APE) percentile Absolute Pricing Error values derived from the whole pricing error distribution are also computed and tabulated. Since ANNs are not optimized solely based on the mean square error and there are cases that the ANNs are optimized with the Huber function, it is wise to take into consideration various error measures.

3.5. The parametric and nonparametric models used

With the BS models input includes *S, X, T, δ, r*, and any of the four different volatility measures: σ_{60} , σ_{vix} σ_{av}, and σ_{avT}; the four different models are symbolized as: BS_{60} , BS_{vix} , BS_{av} , and BS_{avT} .

To train ANNs inputs of the parametric BS model are also used. These include the three standard input variables/parameters: $(Se^{-\delta T})/X$, T and r . The various versions of the ANNs also depend on the BS volatility estimate considered, the kind of the target function, and the *k* value of the Huber function.

As mentioned before, ANNs are trained based on twelve different values of the Huber function ($k \in$ [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.5, 2]). Additionally, ANN structures trained with the use of the mean square error (l_2 norm) which is equivalent to the case where the Huber *k* value is set to a very large value that approaches infinity (*k* = Inf) are included.

Specifically, for each of the four different BS volatility measures, there are thirteen ANN models that are trained to map the standard target function c^{mrk}/X (fifty-two models). Furthermore, each of the previous ANN structures is rebuild based on the hybrid target function, $c^{mrk} / X - \hat{c}^{\Theta} / \mathrm{X}$ where Θ is one of BS_{60} , BS_{vix} , BS_{av} , and BS_{avT} . In total, there are 104 different ANN versions.

The standard ANNs are denoted by *Ns*, and the hybrid versions by *Nh*. To distinguish between various Huber function versions, the corresponding value of the *k* parameter is used in the superscript and the BS volatility reference is used in the subscript. For instance, Ns_{avT}^{Inf} (Nh_{avT}^{Inf}) is the ANN model that uses as fourth input the (maturity adjusted) volatility, $\tilde{\sigma}_{avT}$, maps the standard (hybrid) target function and is trained based on the mean square error.

4. Pricing results and discussion

Table 1 exhibits the out-of sample pricing performance of BS and ANN models with alternative volatility measures. As mentioned before, the various models are compared in terms of RMSE, MAE, MdAE and the 5th and 95th Absolute Pricing Errors. All statistics are reported for the *AggTs* (aggregate) period; for the neural networks the aggregate results are created by selecting the optimal Huber *k*-value in the RMSE measure for each sub-period, aggregating, and then comparing with least squares (*inf*) estimation.

[Tables 1 and 2, here]

It is obvious that the implied volatility measures lead to lower pricing errors in the case of BS. Looking at the parametric models and similarly to Bakshi et al. (1997), the overall best BS model is the one that utilizes the implied average per maturity volatility, BS_{avT}, followed by BS_{av} that utilizes the overall average. The BS_{avT} model outperforms significantly all others in all error measures. Specifically, BS_{avT} has RMSE equal to 7.952, MAE equal to 4.646 and MdAE equal to 3.274. In addition, this model has a higher chance for small pricing errors and considerably smaller chance for large pricing errors compared to the other models (see the 5th and 95th APE statistics).

In comparing the parametric models with the standard (non-hybrid) ANNs that were trained based on the mean square error criterion, it is true that in general, the standard ANN models underperform the equivalent parametric ones (see also Lajbcygier et al, 1996). But Huber standard ANN models perform better than the equivalent least squares ones. The significance of the improvement provided by the Huber approach is obvious from the APE error measures. In some cases (Ns_{vix}) the improvement provides a model better than the equivalent parametric one.

Before considering the impact of the Huber approach, it is evident that the hybrid least squares ANNs outperform significantly both the respective parametric ones, and the standard ANNs, in all measures considered in practically all cases. Similarly to the parametric OPMs, the out of sample pricing accuracy of ANNs seems to be highly dependent on the implied parameters used; that is, as we move from Nh_{60}^{Inf} to Nh_{avT}^{Inf} the pricing accuracy improves significantly. The hybrid least squares ANNs even with historical or weighted average input parameters are considerably better than the equivalent parametric alternatives. Furthermore, it can be observed that Nh_{avT}^{Inf} outperforms all other parametric and least squares ANN models.

The Huber optimized hybrid ANN models outperform significantly all equivalent standard ANNs (Huber and least squares) in all error measures considered. The Huber optimized hybrid ANN models outperform significantly all equivalent least squares hybrid ANNs, in all measures considered in practically all cases. The only exception is when *vix* volatility is used and in a small difference among the RMSE measures; in all other measures, this model with the Huber approach proved to be superior to the least squares one. Again, the Huber optimized hybrid ANN model with *avT* volatility is the overall best, with RMSE equal to 6.83, MAE equal to 3.56, MdAE equal to 2.38, and 5th APE equal to 0.20. We should feel confident in selecting this model, since its 95th APE is equal to 9.13, compared to 11.37 of the equivalent least squares ANN.

Since in each testing sub-period we used the optimal Huber *k*-value determined from the validation set, Table 2 demonstrates a clustering summary for standard and hybrid ANNs, in the RMSE and the MAE error measure. It shows the range that includes the majority of observed optimal *k* values (six out of the nine). For the standard ANNs we have a strong clustering around 0.1 and 0.2, and for the hybrid ANNs values around 0.3 and 0.6 are the most likely ones.

[Tables 3 and 4, here]

Tables 3 (for the standard ANN) and 4 (for the hybrid ANN) present information about the percent of observations treated as outliers by the use of the Huber function (using the RMSE as the error measure). Each cell is for a maturity and degree of moneyness classification, and one line gives the number of observations in that cell and the following line the percent of those observations treated as outliers. For the standard neural networks we observe outliers heavily concentrated in the in-the-money observations of short and medium maturity options. There is also evidence of outliers present in at-the-money long maturity options. Drawing on Long and Officer (1997) the long-maturity at-the-money outliers instead, may be attributed to microstructure effects. As Long and Officer show, excessive demand for certain options may also induce the presence of outliers. For the hybrid neural networks we observe that the Huber technique is even more important since outliers are heavily concentrated not only in in-the-money but also in out-of-the-money observations; furthermore, other cells also often show significant evidence of outliers. The wide range of outliers in the hybrid model is a hint that the misspecification of the BS model is in general rather significant in all ranges of moneyness and maturity. Heavily out-of-money outliers may also be due to thin (non-synchronous) trading effects (Day and Lewis, 1988). For the hybrid model, the choice of volatility used with BS seems to be more important than for the standard neural network.

In the spirit of Black and Scholes (1972), Galai (1977), and Whaley (1982), the economic significance of the OPMs has also been investigated by implementing trading strategies. Trading strategies are implemented based on single instrument hedging as for example in Bakshi et al. (1997), and in addition, transaction costs and cost-effective heuristics are incorporated (see Andreou et al., 2005). Portfolios are created by buying (selling) options undervalued (overvalued) relative to a model's prediction and taking a delta hedging position in the underlying asset. This (single-instrument) delta hedging follows the no-arbitrage strategy of Black and Scholes (1973), where a portfolio including a short (long) position in a call is hedged via a long (short) position in the underlying asset, and the hedged portfolio rebalancing takes place in discrete time intervals. Rebalancing is done in an optimal manner, not necessarily daily; the position is held *as* long *as* the call is undervalued (overvalued) without necessarily daily rebalancing. Proportional transaction costs of 0.2% are also paid for both positions (in the call option and in the "index shares"). Strategies with enhanced cost-effectiveness are also implemented by ignoring trades that involve call options whose absolute percentage mispricing error is less than a mispricing margin of 15%. Even with transaction costs, there still exist opportunities for profitable trading. Again, the hybrid neural networks outperform all other models, and when estimated via the Huber approach they outperform the ones estimated via least squares.

5. Conclusions

The option pricing ability of Robust ANNs optimized with the Huber function is compared with that of ANNs optimized with Least Squares. Comparison is in respect to pricing European call options on the S&P 500 Index from April 1998 to August 2001. In the analysis, a historical parameter, a VIX volatility proxy derived by a weighted implied, and implied parameters (an overall average, and an average per maturity) are used. Simple ANNs (with input supplemented by historical or implied parameters), and hybrid ANNs that in addition use pricing information directly derived by the parametric model are considered. The economic significance of the models is investigated through trading strategies with transaction costs. Instead of *naïve* trading strategies, improved (dynamic and cost-effective) ones are implemented. The use of the robust Huber technique has delivered better ANN structures. The results can be synopsized as follows:

Regarding out-of-sample pricing, the hybrid models outperform both the standard ANNs and the parametric ones. Huber optimization improves significantly the performance of both the standard and the hybrid ANNs. The non-hybrid ANNs are affected more by large errors, and thus require smaller Huber *k*-value. The overall best models were the Huber based hybrid ANNs. In general, within each class, the best performing Huber model has considerably smaller probability of large mispricing compared to the least squares counterpart. Lye and Martin (1993) identify the importance of the generalized exponential distributions for the error function, in the presence of skewed fat-tailed error distribution. Future work could consider option pricing with robust ANNs that explicitly account for such error distributions. Regarding the economic significance of the models, the Huber models are the overall best models. We have also found that profitable opportunities still exist with single-instrument cost-effective trading strategies and 0.2% proportional transaction costs.

Acknowledgements

This work has been partly funded by the Center for Banking and Financial Research at the University of Cyprus, and a University of Cyprus grant for research in ANNs and Derivatives.

Notes

[1] Huber (1981) and Hampel et al. (1986) offer an overview for the tools and concepts of the theory of robust statistics. As pointed out for example by Franses et al. (1999), parametric estimators that are derived under the assumption of normally distributed errors are very sensitive to outliers and other departures from the normality assumption (see also Krishnakumar and Ronchetti, 1997, and Ortelli and Trojani, 2005). They show that the results obtained under a robust analysis can differ significantly from the ones obtained under similar techniques that are based on the Gaussian analysis. Chang (2005) has found that the use of the Huber estimation can significantly reduce the influence of outliers for the estimation of block-angular linear regression model.

[2] According to Andersen et al., (2002), "*the option pricing formula associated with the Black and Scholes diffusion is routinely used to price European options*".

References

Ackert, L.F. and Tian, Y.S. (2001). Efficiency in index option markets and trading in stock baskets. *Journal of Banking and Finance*, **25**, 1607-1634.

Andersen, T.G., Benzoni, L. and Lund, J. (2002). An empirical investigation of continuous-time equity return models. *Journal of Finance*, **57**(3), 1239-1276.

Andreou, P.C., Charalambous, C. and Martzoukos, S.H. (2005). Pricing and trading European options by combining artificial neural networks and parametric models with implied parameters. To be published in the *European Journal of Operational Research*.

Bakshi, G., Cao, C. and Chen, Z. (1997). Empirical performance of alternative options pricing models. *Journal of Finance*, **52**(5), 2003-2049.

Bakshi, G., Cao, C. and Chen, Z. (2000). Pricing and hedging long-term options. *Journal of Econometrics*, **94**, 277-318.

Bandler, W.J., Chen, H.S., Biernacki, M.R., Gao, L. and Madsen, K. (1993). Huber optimization of circuits: a robust approach. *IEEE Transactions on Microwave Theory and Techniques*, **41**(12), 2279-2287.

Bates, D.S. (1991). The Crash of '87: Was it expected? The evidence from options markets. *Journal of Finance*, **46**(3), 1009-1044.

Bates, D.S. (1996). Jumps and stochastic volatility: Exchange rate processes implicit in Deutsche mark options. *The Review of Financial Studies*, **9**(1), 69-107.

Bates, D.S. (1996b). Testing Option Pricing Models. In Maddala, G S. and Rao, C.R., eds., *Statistical Methods in Finance* (Handbook of Statistics, v. 14). Amsterdam: Elsevier, 567-611.

Bates, D.S. (2003). Empirical option pricing: A retrospection. *Journal of Econometrics*, **116**, 387-404.

Bishop, M.C., 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Black, F. and Scholes, M. (1972). The valuation of option contracts and a test of market efficiency. *The Journal of Finance*, **27**, 399-417.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, **81**, 637-654.

Black, F. and Scholes, M. (1975). Fact and fantasy in the use of options. *The Financial Analysts Journal*, **31**, 36-41 and 61-72.

Canica, L. and Figlewski, S. (1993). The informational content of implied volatility. *The Review of Financial Studies*, **6**(3), 659-681.

Chang, W.X. (2005). Computation of Huber's M-estimated for a block-angular regression problem. Forthcoming in the *Computational Statistics & Data Analysis*.

Chiras, D.P. and Manaster, S. (1978). The informational content of option prices and a test of market efficiency. *Journal of Financial Economics*, **6**, 213-234.

Cont, R. and Fonseca, J. (2002). Dynamics of implied volatility surfaces. *Quantitative Finance*, **2**, 45-60.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signal and Systems*, **2**, 303-314.

Day, T.E. and Lewis, C.M. (1988). The behavior of the volatility implicit in the prices of stock index options. *Journal of Financial Economics*, **22**, 103-122.

Devabhaktuni, V., Yagoub, M.C.E., Fang, Y., Xu, J. and Zhang, Q.J. (2001). Neural networks for microwave modeling: model development issues and nonlinear modeling techniques. *International Journal of RF and Microwave CAE*, **11**, 4-21.

Dumas, B., Fleming, J. and Whaley, R. (1995). Implied volatility smiles: Empirical tests. *Journal of Finance*, **LIII**(6), 2059-2106.

Ederington, L. and Guan, W. (2005). The information frown in option prices. *Journal of Banking and Finance*, **29**(6), 1429-1457.

Franses, H.P., Kloek, T. and Lucas, A. (1999). Outlier robust analysis of long-run marketing effects for weekly scanning data. *Journal of Econometrics*, **89**, 293-315.

Galai, D. (1977). Tests of market efficiency of the Chicago Board Options Exchange. *The Journal of Business*, **50**, 167-197.

Garcia, R. and Gencay, R. (2000). Pricing and hedging derivative securities with neural networks and a homogeneity hint. *Journal of Econometrics*, **94**, 93-115.

Hagan, M.T., Demuth, H. and Beale, M. (1996). *Neural Network Design*. PWS Publishing Company.

Hagan, M.T. and Menhaj, M. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, **5**(6) 989-993.

Hampel, F.R., Ronchetti, E.M., Rousseeuw, P.J. and Stahel, P.J. (1986). *Robust Statistics: The Approach Based on Influenced Functions*. Wiley, New York.

Haykin, S. (1999). *Neural Netwroks – A Comprehensive Foundation*, 2nd Ed., Prentice Hall.

Huber, P. (1981). *Robust Statistics*. Wiley, New York.

Hutchison, J.M., Lo, A.W. and Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, **49**(3), 851-889.

Jabr, R.A. (2004). Power system Huber M-estimation with equality and inequality constraints. Forthcoming in *Electric Power Systems Research*.

Kamara, A. and Miller, T.W. (1995). Daily and intradaily tests of put-call parity. *Journal of Financial and Quantitative Analysis*, **30**, 519-539.

Koenker, R. (1982). Robust Methods in Econometrics. *Econometric Reviews*, **1**(2), 213-255.

Krishnakumar, J. and Ronchetti, E. (1997). Robust estimators for simultaneous equations models. *Journal of Econometrics*, **78**, 295-314.

Lajbcygier P., Boek C., Palaniswami M. and Flitman A. (1996). Comparing conventional and artificial neural network models for the pricing of options on futures. *Neurovest Journali,* **4**(5), 16-24.

Lajbcygier, P., Flitman, A., Swan, A. and Hyndman, R. (1997). The pricing and trading of options using a hybrid neural network model with historical volatility. *Neurovest Journal,* **5**(1) 27-41.

Latane, H.A. and Rendleman, R.J.Jr. (1976). Standard deviations of stock price ratios implied in option prices. *The Journal of Financei,* **31**(2), 369-381.

Lim, G.C., Lye J.N., Martin G.M. and Martin V.L. (1997). Jump Models and Higher Moments. In Creedy J., Martin V. L. (eds.), *Nonlinear Economic Models. Cross-sectional, Time Series and Neural Networks Applications*. Edward Elgar Publishing, Inc., Lyme NH, US.

Long, D.M. and Officer, D.T. (1997). The Relation Between Option Mispricing and Volume in the Black-Scholes Option Model. *Journal of Financial Research*, **XX** (1), 1-12.

Lye, J.N. and Martin, V.L. (1993). Robust Estimation, Non-normalities and Generalized Exponential Distributions. *Journal of the American Statistical Association*, **88** (421), 253-259.

Morgenthaler, S. (1990). Fitting Redescenting M-estimators in Regression. In Lawrence, K.D. and Arthur J.L. (eds.) *Robust Regression*, 105-128. Dekker, NY.

Ortelli, C. and Trojani, F. (2005). Robust efficient method of moments. *The Journal of Econometrics*, **128**(1), 69-97.

Rousseeuw, P. and Yohai, V.J. (1984). Robust Regression by Means of S-estimators. Robust and Nonlinear Time Series Analysis. *Lecture Notes in Statistics*, **26**, 256-272. Springer, NY.

Rubinstein, M. (1985). Nonparametric tests of alternative option pricing models using all reported trades and quotes on the 30 most active CBOE option classes from August 23, 1976 through August 31, 1978. *The Journal of Finance*, **XL**, 455-480.

Schittenkopf C. and Dorffner, G. (2001). Risk-neutral density extraction from option prices: Improved pricing with mixture density networks. *IEEE Transactions on Neural Networks*, **12**(4), 716-725.

Tsay, S.R. (2002) *Analysis of Financial Time Series*, Wiley Series in Probability and Statistics.

Watson, P. and Gupta, K.C. (1996). EM-ANN models for microstript vias and interconnected in multilayer circuits. *IEEE Trans., Microwave Theory and Techniques*, **44**, 2495-2503.

Whaley, R.E. (1982). Valuation of American call Options on dividend-paying stocks. *The Journal of Financial Economics*, **10**, 29-58.

Xi, C., Wang, F., Devabhaktuni, V.K. and Zhang, J.Q. (1999). Huber optimization of neural networks: a robust training method. *International Joint Conference on Neural Networks*, 1639-1642.

Yohai, V.J. (1987). High Breakdown-Point and High Efficiency Robust Estimates for Regression. *Annals of Statistics*, **15** (2), 642-656.

Yao, J., Li, Y. and Tan, C.L. (2000). Option price forecasting using neural networks. *The International Journal of Management Science*, **28**, 455-466.

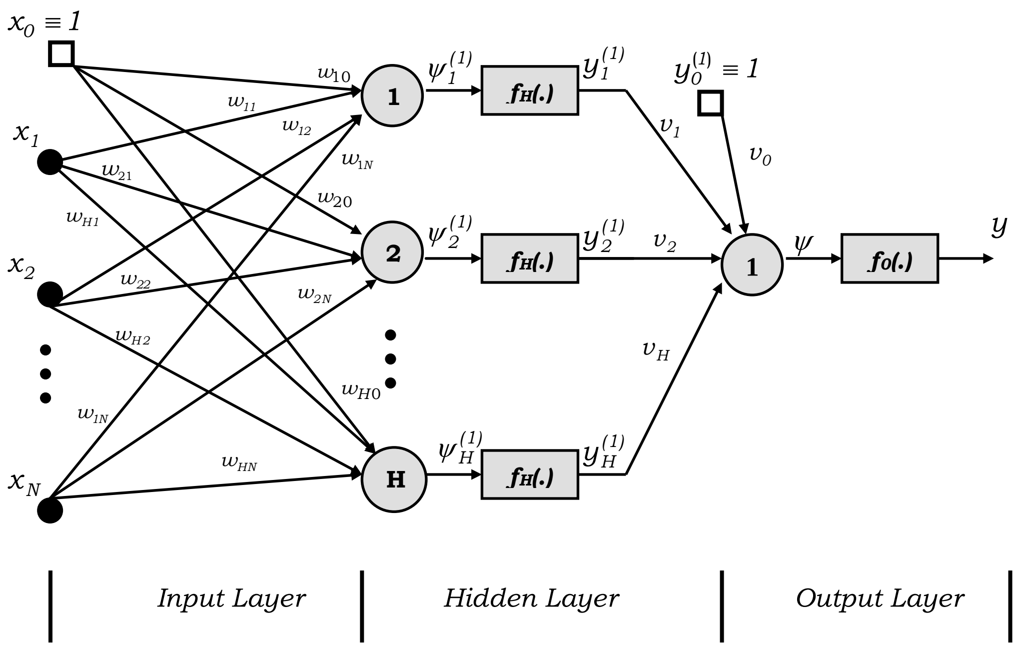

Figure 1: ***A single hidden layer feedforward neural network***

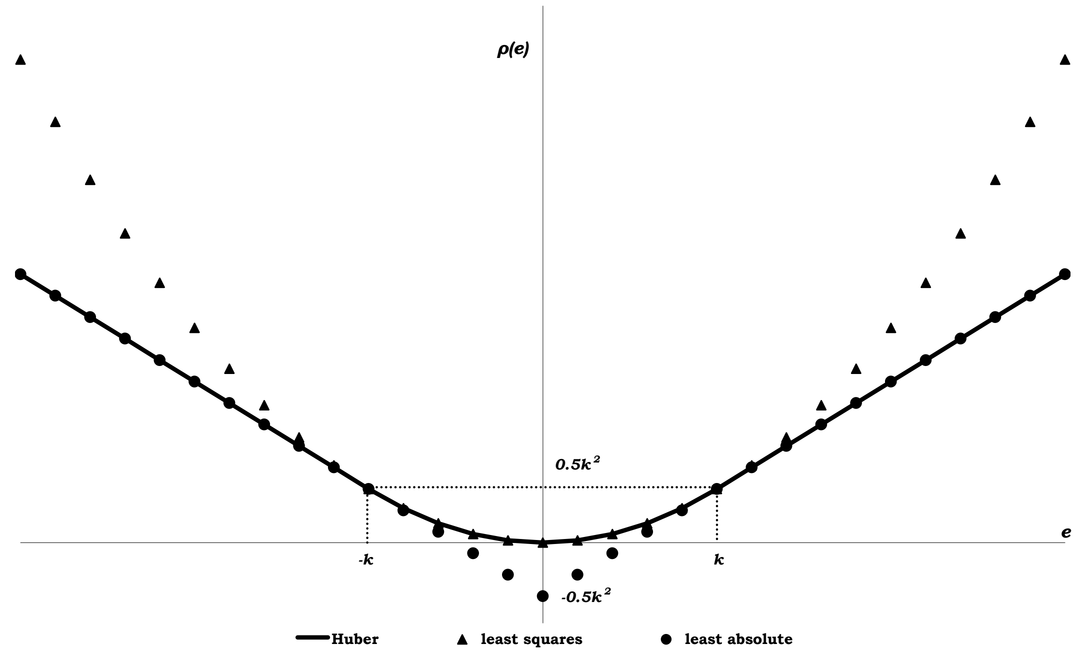

Figure 2: ***The Huber, the least absolute (l_1) and the least squares (l_2) error functions***

	Parametric Models			
	BS_{60}	BS_{vix}	BS_{av}	BS_{avT}
RMSE	10.360	12.302	8.266	7.952
MAE	6.620	8.631	4.989	4.646
MdAE	4.458	6.386	3.630	3.274
5th APE	0.302	0.482	0.323	0.256
95th APE	19.448	23.732	12.399	11.672
	Standard Neural Networks (Optimal *k*, *Inf*)			
	Ns_{60}	Ns_{vix}	Ns_{av}	Ns_{avT}
RMSE	10.52, 15.38	10.08, 12.70	11.25, 11.92	10.76, 12.07
MAE	5.73, 9.51	4.67, 6.44	5.18, 6.62	5.42, 5.90
MdAE	4.06, 6.58	2.99, 3.98	3.35, 4.28	3.40, 3.53
5th APE	0.44, 0.50	0.30, 0.41	0.34, 0.44	0.33, 0.36
95th APE	14.90, 26.54	12.71, 18.92	13.29, 20.20	15.10, 17.39
	Hybrid Neural Networks (Optimal *k*, *Inf*)			
	Nh_{60}	Nh_{vix}	Nh_{av}	Nh_{avT}
RMSE	8.16, 8.58	7.88, 7.79	7.21, 7.73	6.83, 7.15
MAE	5.05, 5.59	3.95, 4.60	4.13, 4.52	3.56, 4.02
MdAE	3.54, 4.02	2.50, 3.07	2.87, 3.03	2.38, 2.58
5th APE	0.29, 0.30	0.22, 0.29	0.24, 0.26	0.20, 0.20
95th APE	13.84, 15.21	10.32, 13.40	10.91, 12.65	9.13, 11.37

Table 1: RMSE is the Root Mean Square Error, MAE the Mean Absolute Error, MdAE the Median Absolute Error, 5th APE is the fifth percentile Absolute Pricing Error and 95th APE the 95th percentile Absolute Pricing Error. The right hand side subscripts refer to the kind of historical/implied parameters used. For the neural networks, the information provided is first under optimal *k*-value in each sub-period, and then under least squares estimation.

	σ_{60}	σ_{vix}	σ_{av}	σ_{avT}
ANN RMSE MAE	0.1-0.3, 0.1-0.3	0.1-0.1, 0.1-0.1	0.1-0.2, 0.1-0.2	0.1-0.2, 0.1-0.2
Hybrid RMSE MAE	0.2-0.4, 0.2-0.5	0.1-0.5, 0.1-0.3	0.1-0.8, 0.1-0.7	0.5-0.9, 0.2-0.6

Table 2: Range of observed optimal *k* values (it includes at least the 66.66% of observed optimal *k* values for the 9 testing sub-periods, after the 3 out of the 9 were removed). The first range is for the RMSE and the second for the MAE error measures.

S/X	<0.85	0.85-0.95	0.95-0.99	0.99-1.01	1.01-1.05	1.05-1.15	1.15-1.35	≥1.35
Ns_{60}								
<60 Days	1.13	0.76	0.27	0.09	0.13	0.18	0.32	9.30
60-180 Days	0.48	0.20	0.15	0.17	0.34	0.18	0.51	11.48
≥ 180 Days	0.00	0.06	2.14	4.17	1.56	0.08	0.89	1.62
Ns_{vix}								
<60 Days	0.56	0.22	0.27	0.09	0.15	0.17	0.24	7.57
60-180 Days	0.14	0.10	0.17	0.21	0.40	0.29	0.55	13.59
≥ 180 Days	1.25	0.06	2.31	4.33	1.47	0.20	1.14	1.48
Ns_{av}								
<60 Days	0.71	0.00	0.00	0.00	0.01	0.07	0.06	8.03
60-180 Days	0.07	0.05	0.06	0.13	0.24	0.06	0.37	14.73
≥ 180 Days	0.16	0.13	2.82	4.98	1.91	0.20	1.46	1.96
Ns_{avT}								
<60 Days	1.69	0.73	0.28	0.11	0.17	0.18	0.26	9.10
60-180 Days	0.95	0.22	0.15	0.17	0.32	0.22	0.41	10.79
≥ 180 Days	0.00	0.06	2.14	3.69	1.04	0.16	0.76	1.35

Table 3: The table shows the % of observations that are outliers in each cell (per maturity and degree of moneyness) when the RMSE is used as error measure in the *standard* neural network. The information is grouped vertically for the four volatility measures, starting with the 60 days maturity, then the *VIX*, the overall average (*av*), and finally the average per maturity (*avT*).

S/X	<0.85	0.85-0.95	0.95-0.99	0.99-1.01	1.01-1.05	1.05-1.15	1.15-1.35	≥1.35
Nh_{60}								
<60 Days	53.7	13.2	16.8	21.0	19.8	21.0	19.5	28.7
60-180 Days	40.8	22.0	27.3	30.1	31.6	32.9	22.1	40.4
≥ 180 Days	47.3	43.7	55.4	57.6	56.0	55.1	48.7	61.5
Nh_{vix}								
<60 Days	21.0	7.8	13.8	16.3	16.7	16.6	14.3	27.9
60-180 Days	16.2	10.9	22.1	22.3	22.6	24.2	21.9	37.1
≥ 180 Days	8.6	22.0	31.0	33.5	31.9	30.5	28.9	34.5
Nh_{av}								
<60 Days	44.8	14.1	17.2	18.3	17.8	16.2	17.4	30.7
60-180 Days	29.2	13.7	21.6	20.7	21.3	24.0	25.4	39.0
≥ 180 Days	33.2	25.9	30.3	29.7	28.2	26.0	33.4	41.3
Nh_{avT}								
<60 Days	25.3	2.0	1.7	3.4	3.3	3.1	2.5	18.8
60-180 Days	13.0	1.5	4.3	5.4	6.2	6.4	6.5	23.3
≥ 180 Days	16.1	3.4	9.8	11.4	8.8	7.0	9.8	10.4

Table 4: The table shows the % of observations that are outliers in each cell (per maturity and degree of moneyness) when the RMSE is used as error measure in the *hybrid* neural network. The information is grouped vertically for the four volatility measures, starting with the 60 days maturity, then the *VIX*, the overall average (*av*), and finally the average per maturity (*avT*).