



Sermpinis, G., Laws, J., and Dunis, C.L. (2014) Modelling commodity value at risk with Psi Sigma neural networks using open–high–low–close data. *European Journal of Finance*, 21(4), pp. 316-336.

There may be differences between this version and the published version. You are advised to consult the publisher's version if you wish to cite from it.

<http://eprints.gla.ac.uk/74500/>

Deposited on: 16 February 2016

Enlighten – Research publications by members of the University of Glasgow
<http://eprints.gla.ac.uk>

Modelling Commodity Value at Risk with Psi Sigma Neural Networks using Open-High-Low-Close data

September 2012

Abstract

The motivation for this paper is to investigate the use of a promising class of neural network models, Psi Sigma, when applied to the task of forecasting the one day ahead Value at Risk (VaR) of the oil Brent and gold bullion series using Open-High-Low-Close data. In order to benchmark our results we also consider VaR forecasts from two different neural network designs, the Multilayer Perceptron (MLP) and the Recurrent Neural Network (RNN), a genetic programming algorithm (GP), an Extreme Value Theory model (EVT) along with some traditional techniques such as an ARMA-GJR (1,1) model and the Riskmetrics volatility. The forecasting performance of all models for computing the VaR of the Brent oil and the gold bullion is examined over the period September 2001 to August 2010 using the last year and half of data for out-of-sample testing. The evaluation of our models is done by using a series of backtesting algorithms such as the Christoffersen tests, the violation ratio and our proposed loss function that considers not only the number of violations but also their magnitude.

Our results show that the Psi Sigma outperforms all other models in forecasting the VaR of gold and oil at both the 5% and 1% confidence levels, providing an accurate number of independent violations with small magnitude.

Keywords

Artificial Intelligence, Forecasting, Value at Risk, Extreme Value Theory, Loss Function.

JEL: C32, C53, C63

1. INTRODUCTION

Neural networks and genetic programming algorithms are emergent technologies with an increasing number of real-world applications including finance (Lisboa and Vellido (2000) and Chen (2002)). However their numerous limitations and contradictory empirical evidence around their forecasting power are often create scepticism about their use among practitioners. This scepticism is further fuelled by the fact that the selection of each algorithm inputs is based more on trial and error and the practitioner's market knowledge rather than on some formal statistical procedure.

The motivation for this paper is to investigate the VaR forecasting performance of a promising class of artificial intelligence model, the Psi Sigma (PSI) Neural Network, which tries to overcome some of these limitations. This is achieved by benchmarking their results with those of two different neural networks (NNs) designs, the Multilayer Perceptron (MLP) and the Recurrent Neural Network (RNN) and also a genetic programming algorithm (GP), an Extreme Value Theory model (EVT) , a traditional ARMA-GJR (1,1) model and Riskmetrics volatility. We consider two different confidence levels, 5% and 1% and we analyse both tails of the return distribution in order to provide VaR forecasts for short positions as well as long positions.

The forecasting performance of the one day ahead VaR for gold and oil is evaluated by using traditional measures such as the Christoffersen (1998) test and the violation ratio. We also use a newly introduced loss function that incorporates not only the number of violations but also their magnitude. This differs from previous approaches (see Lopez (1998), Blanco and Ihle (1999) and Sarma *et al.* (2003)) as it is not biased on the number of violations. Instead it considers equally the number of violations and their average magnitude. The Basel Committee on Banking Supervision (1996) indicates that the magnitudes as well as the number of exceptions are a matter of regulatory concern.

In our VaR estimations we use the daily Open-High-Low-Close prices of gold bullion and oil Brent in order to obtain more accurate estimation of the daily realised volatility and Value at Risk than could be achieved using only close to close prices.

We find that the Psi Sigma model outperforms all other models in forecasting the VaR of gold and oil at both the 5% and 1% confidence levels. The RNNs and GPs also provide satisfactory forecasts in the majority of cases. Unfortunately the VaR forecasts derived from the EVT model were accurate only for oil whilst the forecasts derived from MLPs, ARMA-GJR (1,1) and Riskmetrics were disappointing.

Our results confirm that the proposed VaR models and backtesting function add value to traditional risk management toolboxes and provide more accurate VaR estimations and evaluation.

The paper is organised as follows. In section 2, we present the literature relevant to Psi Sigma and the applications of Neural Networks to Value-at-

Risk. Section 3 describes the dataset used for this research and its characteristics. An overview of the VaR framework, the models used in this research, the backtesting algorithms and our empirical results is given in section 4 while section 5 provides some concluding remarks.

2. LITERATURE REVIEW

In our study we apply the Psi Sigma Neural Network, which has been developed recently, with the purpose of overcoming a number of the documented limitations of more classic neural and genetic programming architectures. Although the unique architecture and characteristics of Psi Sigma seem promising in pattern recognition, there is little empirical evidence regarding their financial forecasting power.

Psi Sigma networks were first introduced by Shin and Ghosh (1991) as a neural architecture capable of capturing higher order correlations within the data while avoiding some of the Neural Networks (NNs) limitations such as the combinatorial increase in weight numbers. Shin and Ghosh (1991) and Ghosh and Shin (1992) demonstrate these benefits and present empirical evidence on their forecasting superiority in function approximation when compared with a MLP network and a Higher Order Neural Network (HONN). Ghazali *et al.* (2006) compare them with a HONN and a MLP network on the task of forecasting and trading the IBM common stock closing price and the US 10-year government bond series. Psi Sigma outperformed both benchmarks in terms of statistical accuracy and annualized return. In a similar paper, Hussain *et al.* (2006) present satisfactory results of the Psi Sigma forecasting power on the EUR/USD, the EUR/GBP and the EUR/JPY exchange rates having as benchmarks a HONN model. On the other hand, Dunis *et al.* (2010) who also study the EUR/USD series with Psi Sigma having as benchmarks MLP, RNN and HONN architectures failed to outperform their benchmarks in a simple trading application.

In the field of Risk Management, Locarek-Junge and Prinzler (1998) estimated the VaR of a US dollar portfolio using a Mixture Density Network while Bartlmae and Rauscher (2000) using a Neural Network Volatility Mixture model forecasted successfully the one day ahead VaR of the German Stock index. Neely and Weller (2002) argued in favour of the use of genetic programming as an alternative to GARCH and RiskMetrics while Sadorsky (2005) highlighted the difficulties of constructing accurate parametric VaR forecasting models. Dunis and Chen (2005) demonstrated that Neural Networks Regression models are superior in forecasting the VaR of the EUR/USD exchange rate compared to GARCH and Stochastic Variance models. Furthermore, Liu (2005) by combining historical simulation and a GARCH (1,1) model with Neural Networks achieved accurate VaR estimates for the S&P 500 and the DJI indexes and the Ford and IBM stocks. Similarly, Ozun and Cifter (2007) combine various GARCH, historical simulation and Extreme Value Theory models with Neural Networks to provide accurate estimates of the VaR of the Istanbul Stock Exchange while Huang and Tseng (2009) used kernel estimators to propose an alternative approach of VaR estimation. On the other hand, Dunis *et al.* (2010) used Neural Networks to

forecast the Value-at-Risk of commodities with good results. One of their findings was that the addition of the RiskMetrics volatility of the series under study as input to their Neural Networks, improved substantially the forecasting ability of their models. Similarly, Chen and Hsieh (2010) who forecasted the Value-at-Risk of the Chinese and Hong Kong stock market achieved more accurate forecasts with a simple Neural Network architecture than with a historical simulation and a Monte Carlo approach.

Note that in all of these prior studies closing data was used to model the daily volatility and the VaR of the series under study.

3. THE BRENT OIL AND THE GOLD BULLION SERIES

We aim is to forecast the one day ahead VaR of Brent oil and gold bullion based on their opening-high-low-closing prices with PSI having as benchmarks several other linear and non linear models. This will allow us to test the pattern recognition capabilities of models and their utility in a Financial Risk Management environment. The series under study were obtained from a historical dataset provided by the Reuters X3000 platform.

For the purpose of our research we estimate the daily realised volatility using the Yang and Zhang (2000) extension to the Garman and Klass historical volatility estimator:

$$\sigma_i = \sqrt{\left[\left(\ln \frac{O_i}{C_i - 1} \right)^2 + \frac{1}{2} \left(\ln \frac{H_i}{L_i} \right)^2 - (2 \ln 2 - 1) \left(\ln \frac{C_i}{O_i} \right)^2 \right]} \quad [1]$$

Where $\tilde{\sigma}_i$ is the volatility of day i , O_i is the opening price of day i , L_i is the lowest price of day i , H_i is the highest price of day i and C_i is the closing price of day i . The above estimator which allows for an opening jump and assumes Brownian motion with zero drift is maximally 14 times more efficient than the close-to-close estimator (Yang and Zhang (2000)). Based on the characteristics of our data series¹ this volatility estimate is preferred to other OHLC measures such as the Garman and Klass (1980), Parkinson (1980), Rogers and Satchell (1991) and Rogers *et. al.* (1994).

In Appendix A.1 we present the inputs of our Neural Networks models and the methodology behind this choice.

We examine our series over the period 1 September 2001 to 31 August 2010. The data period is partitioned as follows (see Table 1), where we leave approximately 17% of our data set for out of sample evaluation.

Insert Table 1

¹ Both series exhibit opening jumps and have very low average return in the in-sample sub-period (0.03% and 0.06% for oil and gold respectively). See Yang and Zhang (2000), Chan and Lien (2003) and Floros (2009) for further justification in our use of equation [1].

In order to train our neural networks we further divided our dataset as illustrated in table 2:

Insert Table 2

4. THE VALUE AT RISK FORECASTING MODELS AND BACKTESTING

Under a probabilistic framework, at the time t , we are interested in the risk of a financial position for the next h periods. If we define " $V(h)$ " to be the asset value change from time t to $t+h$, then this quantity is measured in monetary terms and is a random variable at time t . If we denote the conditional density function (CDF) of " $V(h)$ " by $F_h(x)$ then we define the VaR of a long position over the time horizon h with probability p as:

$$p = \Pr[\Delta V(h) \leq VaR] = F_h(VaR) \quad [2]$$

For a long position, the loss occurs when $\Delta V(h) < 0$ and so the VaR defined in [2] is assumed to have a negative value. On the other hand, for the short position the loss occurs when $\Delta V(h) > 0$ and the VaR has a positive value. As an investor can take both long positions, where the left hand side of the tail is of interest from a risk perspective, and short positions, where the right hand side of the tail is of interest, we consider both tails of the distribution. Moreover, the asymmetries between the tails of the return distributions of our assets enables us to draw additional conclusions when large discrepancies are observed in our results for long and short positions.

Equation [2] can be also interpreted in such a way to represent the probability that the holder would encounter a loss greater than or equal to VaR. It follows then that in order to model VaR we are required to model the tail behavior of the CDF, $F_h(x)$. In practice the CDF is unknown and most VaR forecasting models therefore require an *a priori* definition of the CDF. In this paper, except for the EVT model, we assume that the unknown CDF is either the normal distribution or the Student t-distribution with 6 degrees of freedom². Note further that we are unable to estimate the CDF of the series with the frequency of data at our disposal, i.e. open, close, high and low.

4.1 Statistical Techniques

4.1.1 RiskMetrics Volatility

² Although we acknowledge that our series do not follow the normal or the t-distribution, we make this assumption based on the basis of the existing literature (see amongst others Lee and Saltoglu (2001), Dunis *et al.* (2005) and Rau-Bredow (2004) for the normal distribution and Jorion (1997) and Ho *et al.* (2000) for the Student t-distribution with 6 degrees of freedom).

The RiskMetrics volatility model is treated as a benchmark model owing to its popularity in risk measurement. RiskMetrics is one of the simplest tools for measuring financial market risk under the VaR framework. Derived from the GARCH(1,1) model, but with fixed coefficients, RiskMetrics volatility is calculated using the standard formula:

$RMVOL_t^2 = b\sigma_{t-1}^2 + (1-b)R_t^2$ [3] where σ_t^2 is the asset variance at time t, R_t^2 is the asset squared return at time t and $b=0.94$ for daily data. In this paper we use RiskMetrics volatility to forecast 1-day ahead volatility for the out-of-sample period. The RiskMetrics volatility is calculated from equation [3] and then we use equation [4] below to calculate the 1-step ahead volatility

forecast: $\hat{\sigma}_{t+1} = RMVOL_t$ [4]. **The computed VaR is then: $VaR_{t+1}^q = \hat{\sigma}_{t+1} c_q$ [5] where VaR_{t+1}^q is the VaR forecast for period t+1 at the q% confidence level, $\hat{\sigma}_{t+1}$ is the forecasted volatility for period t+1 and c_q is the critical value of the normal distribution at the q% confidence level.**

4.1.2 ARMA-GJR(1,1)

As an additional benchmark we also forecast the VaR of our assets with an ARMA-GJR(1,1) model. One of the primary restrictions of traditional GARCH models is their symmetric response to positive and negative shocks. However, it has been argued that a negative shock to financial time series is likely to cause volatility to rise by more than a positive shock of the same magnitude (Bekaert and Wu (2000) and Brooks (2002)). A popular asymmetric formulation of GARCH which has an additional term to account for possible asymmetries is the Glosten, Jagannathan and Runkle (hereafter GJR, 1993) model.

In mathematical terms an ARMA(p,q)-GJR(1,1) can be expressed as:

$$R_t = z + \sum_{x=1}^p \beta_x R_{t-x} + \sum_{x=1}^q \alpha_x \varepsilon_{t-x} + u_t \quad (\text{Mean equation}) \quad [6]$$

$$\sigma_t^2 = \chi + \gamma \varepsilon_{t-1}^2 + \delta \sigma_{t-1}^2 + \lambda \varepsilon_{t-1}^2 I_{t-1} \quad (\text{Variance equation}) \quad [7]$$

where β_x and α_x are the ARMA parameters, σ_t^2 and ε_t are the conditional variance and the error respectively at time t, z and χ are constants, R_t is the return of day t and $I_{t-1} = 1$ if $\varepsilon_{t-1} < 0$ or 0 otherwise.

The computer VaR using the ARMA-GJR(1,1) model is then:

$VaR_{t+1}^q = \hat{R}_{t+1} - c_q \hat{\sigma}_{t+1}$ [8] where VaR_{t+1}^q is the VaR forecast for t+1 period at the q% confidence level, \hat{R}_{t+1} and $\hat{\sigma}_{t+1}$ are obtained from equations [6] and [7] respectively while c_q is the critical value of the normal or the Student t-distribution at the q% confidence level.

4.1.3 Extreme Value Theory Model

Extreme Value Theory (EVT) is a powerful and yet fairly robust framework to study the tail behaviour of a distribution. Even though the theory has primarily been applied to climatology and hydrology, recently there has been an increasing number of extreme value studies in the VaR and Risk Management literature (see for example Bali (2003), Gencay and Selcuk (2004), Gilli and Kellezi (2006), Samuel (2008) and Dunis *et. al.* (2010)).

There are two main approaches in estimating VaR with EVT, namely the “method of block over maxima” and the method of “peaks over threshold” (POT). In our research, based on the empirical evidence provided by Gilli and Kellezi (2006) who compared the two methods, we will estimate VaR with the POT approach. Moreover, in our estimation we followed the unconditional approach. In real world environments, the unconditional approach is preferred as it can provide stable estimates through time while avoiding the time consuming computations required by the conditional approach (Gilli and Kellezi (2006)).

The POT method is based on a theorem stated by Picklands (1975) and Balkema and de Haan (1974). According to it, for a large class of underlying distribution functions F the conditional excess distribution $F_u(y)$, for a threshold u large enough, is well approximated by the Generalised Pareto Distribution (GPD). In our application we choose our threshold following the methodology employed by Bali (2003) who suggests choosing as a threshold the distance of 2 standard deviations from the in-sample mean. Then given the excesses over the threshold, we estimate the parameters of the GPD using the method of moments³. In the end, we compute the VaR using the

formula:
$$VaR^q = u + \frac{\hat{\phi}}{\hat{\xi}} \left[\left(\frac{n}{Nu} p \right)^{-\hat{\xi}} - 1 \right]$$
 [9] where VaR^q is the VaR estimate at

the $q\%$ confidence level, u is the threshold, $\hat{\phi}$ and $\hat{\xi}$ are the moments estimates of the shape and scaling parameters of the GPD respectively, n is the sample size and Nu is the number of observations above u . Further details about the POT method and our VaR estimation can be found in Gilli and Kellezi (2006) and Samuel (2008).

4.2 Neural Networks

The most popular neural network architecture is the Multi-Layer Perceptron (MLP). A standard MLP neural network has at least three layers. The first layer is called the input layer (the number of its nodes corresponds to the number of explanatory variables). The last layer is called the output layer (the number of its nodes corresponds to the number of response variables). An intermediary layer of nodes, the hidden layer, separates the input from the output layer. Its number of nodes defines the amount of complexity the model

³ Moments estimators for the GPD were derived by Hosking and Wallis (1987). According to the empirical evidence provided by Sing and Guo (1995) the method of moments seems more accurate than the maximum likelihood approach for estimating the parameters of a GPD distribution.

is capable of fitting. In addition, the input and hidden layer contain an extra node, called the bias node. This node has a fixed value of one and has the same function as the intercept in traditional regression models. Normally, each node of one layer has connections to all the other nodes of the next layer.

The network processes information as follows: the input nodes contain the value of the explanatory variables. Since each node connection represents a weight factor, the information reaches a single hidden layer node as the weighted sum of its inputs. Each node of the hidden layer passes the information through a nonlinear activation function and passes it on to the output layer if the calculated value is above a threshold.

The training of the network (which is the adjustment of its weights in the way that the network maps the input value of the training data to the corresponding output value) starts with randomly chosen weights and proceeds by applying a learning algorithm called backpropagation of errors⁴ (Shapiro (2000)). The learning algorithm simply tries to find those weights which optimise an error function (normally the sum of all squared differences between target and actual values). Since networks with sufficient hidden nodes are able to learn the training data (as well as their outliers and their noise) by heart, it is crucial to stop the training procedure at the right time to prevent overfitting (this is called 'early stopping'). This can be achieved by dividing the dataset into 3 subsets respectively called the training and test sets used for simulating the data currently available to fit and tune the model and the validation set used for simulating future values. The network parameters are then estimated by fitting the training data using the above mentioned iterative procedure (backpropagation of errors). The iteration length is optimised by maximising the forecasting accuracy for the test dataset. Finally, the predictive value of the model is evaluated applying it to the validation dataset (out-of-sample dataset).

The target value in our networks is the one day ahead estimated realised volatility of our series as defined by equation [1]. After we forecast the volatility with our Neural Network's and Genetic Programming models, we compute the VaR using equation [5].

Since the starting point for each network is a set of random weights, forecasts can differ between networks. In order to eliminate any variance between our forecasts, we used the average of a committee of 20 Neural Network's which presented the best in-sample statistical performance. The characteristics of the NNs used in this paper are presented in Appendix A.1.

4.2.1 The Multilayer Perceptron Model

The network architecture of a 'standard' Multi-Layer Perceptron is presented in figure 1⁵:

⁴ Backpropagation networks are the most common multilayer networks and are the most commonly used type in financial time series forecasting (Kaastra and Boyd (1996)).

⁵ The bias nodes are not shown here for the sake of simplicity.

Insert Figure 1


where:

$x_t^{[k]}$ ($k = 1, 2, \dots, N + 1$) are the model inputs (including the input bias node) at time t

$h_t^{[j]}$ ($j = 1, 2, \dots, M + 1$) are the hidden nodes outputs (including the hidden bias node)

\tilde{y}_t is the **network's output**

$u_t^{[jk]}$ and $w_t^{[jl]}$ are the network weights for the input layer and the hidden layer respectively

 is the transfer sigmoid function:
$$S(x) = \frac{1}{1 + e^{-x}}, \quad [10]$$

 is a linear function:
$$F(x) = \sum_i x_i \quad [11]$$

The error function to be minimised is:

$$E(u_t^{[jk]}, w_t^{[jl]}) = \frac{1}{T} \sum_{t=1}^T (y_t - \tilde{y}_t(u_t^{[jk]}, w_t^{[jl]}))^2, \quad [12]$$

with y_t being the target value and T the number of iterations.

4.2.2 The Recurrent Network

While a complete explanation of RNN models is beyond the scope of this paper, we present below a brief explanation of the RNN architecture. For an exact specification of recurrent networks, see Elman (1990). A simple recurrent network has an activation feedback which embodies short-term memory. In other words, the RNN architecture can potentially provide more accurate outputs because the inputs are taken from all previous values. The advantages of using recurrent networks over feedforward networks (such as the MLPs) for modelling non-linear time series, has been well documented in the past (see amongst others Elman (1990) and Tenti (1996)). However as mentioned by Tenti (1996), "the main disadvantage of RNNs is that they require substantially more connections, and more memory in simulation than standard backpropagation networks" (p.569), thus resulting in a substantial increase in computational time.

4.2.3 The Psi Sigma Network

Psi Sigma (PSI) networks can be considered as a class of feedforward fully connected higher order neural networks. First introduced by Shin and Ghosh (1991), the PSI network utilizes product cells as the output units to indirectly

incorporate the capabilities of higher-order networks while using a fewer number of weights and processing units. Their creation was motivated by the need to create a network combining the fast learning property of single layer networks with the powerful mapping capability of higher order neural networks whilst avoiding the combinatorial increase in the required number of weights. The order of the network in the context of Psi Sigma is represented by the number of hidden nodes.

In a PSI network the weights from the hidden to the output layer are fixed to 1 and only the weights from the input to the hidden layer are adjusted, something that greatly reduces the training time. Moreover, the activation function of the nodes in the hidden layer is the summing function while the activation function of the output layer is a sigmoid function. Figure 2 below shows a Psi Sigma with one output layer.

Insert Figure 2

where:

$$v_t^{[jk]} \quad \text{are the adjustable weights}$$

$$h(x) = \sum_i x_i \quad \text{is the hidden layer activation function} \quad [13]$$

$$\sigma(x) = \frac{1}{1 + e^{-xc}} \quad \text{is the output unit adaptive sigmoid} \quad [14]$$

activation function with c the adjustable term

Similar with MLPs, the error function to be minimised is:

$$E(c, v_t^{[jk]}) = \frac{1}{T} \sum_{t=1}^T (y_t - \tilde{y}_t(v_t^{[jk]}, c))^2 \quad [15]$$

with y_t being the target value and T the number of iterations.

Note that by using products in the hidden layer we directly incorporate the capabilities of higher order networks with a smaller number of weights and processing units. For example, a k-th degree higher order neural network with

- inputs needs $\sum_{i=0}^k \frac{(N+i-1)!}{i!(N+1)!}$ weights if all products of up to k components are

to be incorporated while a similar PSI network needs only $(\bullet + 1) \cdot k$ weights. Also note that the sigmoid function is neuron adaptive. As the network is trained not only the weights but also c in equation [14] is adjusted. This strategy provides better fitting properties and increases the approximation capability of a neural network by introducing an extra variable in the estimation, compared to classical architectures with sigmoidal neurons (see *Vecci et al. (1998)*).

4.2.4 The Genetic Programming Algorithm

For the purpose of our research, the GP application is coded and implemented to evolve tree based structures that present models (i.e. sub-trees) of input – output's. In the design phase of our GP application we focus primarily on execution time optimization as well as limiting the 'bloat effect'. The bloat effect is similar to the issue of overfitting experienced in neural networks however in our case we run the risk of continuously increasing and expanding the tree size. This algorithm is run in a 'steady state' in that a single member of the population is replaced at a time. Furthermore, our GP application reproduces newer models replacing the weaker ones in the population according to their fitness. The reasoning behind the decision to use a steady state algorithm is justified as they hold a greater selection strength and genetic drift over other algorithms such as typical generational GA. Additionally steady state algorithms also offer exceptional multiprocessing capabilities (see Ferreira (2006)).

See Koza (1998) for a summary description of the implementation of the of the GP algorithm.

Since the generation of the initial population is randomly constructed, forecasts can differ between GP algorithms. We overcome this issue in the same manner as we overcame a related issue in implementing the neural network models, i.e we used the average of a committee of 20 GP algorithms. The characteristics of the GPs used in this paper are presented in Appendix A.1. Similarly with our NNs models the target value in our GPs is the one day ahead estimated realised volatility of our series as defined by equation [1]. The one day ahead VaR of our series is then computed using equation [5].

4.3 Backtesting

4.3.1 Christoffersen Tests

Christoffersen (1998) introduced a three step VaR evaluation procedure. In a likelihood ratio (LR) testing environment, he introduced a test of correct unconditional coverage, a test of independence and a test for conditional coverage. As a first step in order to evaluate our models we follow this procedure.

4.3.1.1 LR test of correct unconditional coverage

Let us consider a dummy variable d_k^t for model k which takes the value of 1 when the return falls behind the VaR forecast estimated from model k and 0 in all other cases. Then the indicator sequence d_k^t should follow the binomial distribution with likelihood $L(a) = (1-a)^{n_0} a^{n_1}$ where $a = P(d_k^t = 1)$, n_0 is the number of 0 in the d_k^t sequence and n_1 is the number of 1. In an accurate VaR model with confidence level $q\%$, q should equal \pm . Consequently the probability of a violation is then $q\%$. Christoffersen (1998), under the null that

we have a correct violation ratio, formulated all this in the standard LR test presented below⁶:

$LRuc = -2 \log \left[\frac{q^{n_1} (1-q)^{n_0}}{\pi^{n_1} (1-\pi)^{n_0}} \right] \xrightarrow{d} \chi^2(1)$ [16] where n_1 is the number of violations, n_0 is the number of non-violations, q is the coverage rate of the (but we use q as confidence level) VaR model and $\pi = \frac{n_1}{n_0 + n_1}$ is the maximum likelihood estimate of q .

4.3.1.2 LR test of independence

The second test is to check whether our series of dummy variables d_k^t (is serially independent. Christoffersen (1998) was motivated to such a test by noting that a constant VaR given by the unconditional distribution from a GARCH model will have too many exceptions during periods of high volatility and too few during periods of low volatility. Because volatility tends to cluster, failing to adequately model volatility in the VaR will result in serial correlation in the violation sequence.

Under the null hypothesis that the violation sequence is serially independent and the alternative that it is a first order Markov process, the likelihood ratio of independence can be tested by:

$$LRind = -2 \log \left[\frac{(1-\pi_2)^{n_{00}+n_{11}} \pi_2^{n_{01}+n_{11}}}{(1-\pi_{01})^{n_{00}} \pi_{01}^{n_{01}} (1-\pi_{11})^{n_{10}} \pi_{11}^{n_{11}}} \right] \xrightarrow{d} \chi^2(1)$$
 [17] where n_{ij} is the

number that value i is followed by j in the violations sequence, $\pi_{01} = \frac{n_{01}}{n_{00} + n_{01}}$,

$$\pi_{11} = \frac{n_{11}}{n_{10} + n_{11}} \text{ and } \pi_2 = \frac{n_{01} + n_{11}}{n_{00} + n_{01} + n_{11} + n_{10}}.$$

4.3.1.3 LR test of conditional coverage

By combining the two tests a third test for conditional coverage can be constructed. This time the null hypothesis is that that we have an independent exception process with the correct violation ratio while the alternative is that we have a first order Markov process with a different transition probability matrix. **The likelihood ratio statistic of the test is the sum of the previous two statistics:**

$$LRcc = LRuc + LRind \xrightarrow{d} \chi^2(2)$$
 [18]

4.3.1.4 Results

The likelihood ratio statistics of the Christoffersen tests for our models are presented in Appendix A.2. Our results indicate that most our artificial intelligence models give a correct violation ratio with an independent

⁶ Kupiec (1995) and McNees (1995) apply similar tests of unconditional coverage.

exception process. In other words, that most our NNs and GP algorithms made accurate forecasts in a VaR framework. The only artificial intelligence models that failed to pass the Christoffersen tests were the MLPs for oil for long positions under the Student t-distribution assumption at the 5% confidence level and again for long oil positions under both the normal and Student t-distribution assumptions at the 1% confidence level. On the other hand, our statistical benchmarks, the ARMA-GJR(1,1) and the RiskMetrics models failed to produce accurate forecasts in most cases. Similarly, our more sophisticated EVT model provided unsatisfactory VaR forecasts for gold⁷.

The weakness of Christoffersen tests is that they are unable to distinguish between different, but close, alternative models (see Sarma *et al.* (2003), Cakici and Foster (2003) and Fantazzini (2009)). Therefore, in order to further distinguish our models we follow another approach based on loss functions.

4.3.2 Loss Functions

In order to further verify the reliability of our models and to distinguish their VaR forecasting performance we apply two loss functions to the models that passed the Christoffersen tests. Here we follow Lopez (1998) who defines the general form of those loss functions as:

$$G = \frac{1}{n} \sum_{i=1}^n C_{t+i} \quad [19]$$
 where R_{t+i} and VaR_{t+i} is the actual return and the forecasted VaR from our models for day $t+i$ and $C_{t+i} = f(R_{t+i}, VaR_{t+i})$ if $R_{t+i} < VaR_{t+i}$ while $C_{t+i} = g(R_{t+i}, VaR_{t+i})$ if $R_{t+i} \geq VaR_{t+i}$ such that $f(R_{t+i}, VaR_{t+i}) \geq g(R_{t+i}, VaR_{t+i})$.

4.3.2.1 Violation Ratio

The violation ratio (or the hit rate) is simply a descriptive statistics that measures the percentage occurrence of an actual loss greater than the predicted maximum loss in the VaR framework. In our application this **can be**

formulated for model k as:
$$G_k = \frac{1}{n} \sum_{i=1}^n d_k^t \quad [20]$$

where d_k^t is a dummy variable for model k as described in Section 4.3.1.1 and n is the number of trading days in the out-of-sample period. In our application, we want our models to have a violation ratio as close as it can be to our confidence level. In tables 3 and 4 below, we present the violations ratios of the models that passed the Christoffersen tests.

Insert Table 3

⁷ This can be attributed to the fact that only a few extreme events are present in our gold dataset, something that may have led to misspecifications of our GDP parameters for gold.

Insert Table 4

From the tables above, we note that PSIs provide the closest violation ratios to the benchmark in almost all cases. The GPs and the RNNs provide the second best performance with similar close violation ratios to the confidence levels. On the other hand, the MLPs and the EVT models seem unable to produce accurate VaR violation ratios in the few cases that they pass the Christoffersen tests. We also note that almost in all cases, the performance of our models worsens when we assume that our series follow the Student t-distribution.

4.3.2.2 Proposed Loss Function

A good VaR forecasting model should satisfy two conditions equally. Firstly it should provide an accurate number of hits (violations). Secondly, the magnitude of these violations should be as small as possible. **Our aim is to introduce a loss function that incorporates these two conditions equally.**

Different kinds of loss functions have already been proposed by Lopez (1998), Blanco and Ihle (1999) and Sarma *et al.* (2003). However, all functions proposed in these papers depend crucially on the number of exceptions. The fewer the exceptions, the smaller are the function results. This deficiency is crucial as the theoretical framework of these functions suggests accepting as best the model that gives us the smaller number of violations. So we may reject a correctly specified model with an accurate number of exceptions because it produces a higher loss function than a more conservative model. Therefore a conservative model has an advantage over models that may be more accurately specified (see Caporin (2003)). **To overcome this problem, we introduce a new descriptive statistic that considers the number of violations and their average magnitude in equal terms, avoiding thus the previous mentioned possible misspecifications. Equation [21] below presents our loss function.**

$$S = \left(\frac{V}{n} - p \right)^2 + \frac{1}{V} \sum_{i=1}^n T_i \quad [21]$$

Where V the number of violations of our model, p is the confidence level under study and $T_i = (R_{t+i} - VaR_{t+i})^2$ when $R_{t+i} > VaR_{t+i}$ and $T_i = 0$ when $R_{t+i} \leq VaR_{t+i}$. **A model which minimises [21] is preferred over alternative models. We use our loss function to further discriminate between models which passed the three Christoffersen tests. In table 5 below we present the loss function realizations of our models for gold.**

Insert Table 5

We note that PSIs networks produce the lowest realizations of our loss function in all cases. GPs present also a satisfactory performance with second lowest realizations of our loss function while the RNNs present the third

lowest realizations. On the other hand, the performance of the MLP and the ARMA-GJR models are disappointing with the highest realizations of our loss function at both the 5% and 1% confidence levels. Furthermore, the RiskMetrics and EVT models failed to pass the Christoffersen tests at the 5% confidence level while they were too strict at the 1% confidence level with 0 violations. Also we observe that the realizations of our loss function are higher under the Student t-distribution assumption.

The loss function realizations of our models for oil are included in table 6 below.

Insert Table 6

From table 6, we observe that the VaR forecasting superiority of PSIs is also confirmed for oil. For the majority of positions and confidence levels, PSIs VaR forecasts produce the lowest values of our loss function compared to their benchmarks. The GPs and RNNs present a similar performance with the second lowest realizations while our EVT model forecasts achieve the third lowest realizations. The MLPs, when they pass the Christoffersen tests, produce the fourth lowest realizations of our loss function. On the other hand, the results of our other statistical benchmarks are once more disappointing as they produce the highest values of our loss function in the few cases that they pass the Christoffersen tests.

Again, we note that in most cases the realizations of our loss function increase when we assume that our series follow the Student t-distribution. This supports our previous remarks that for the series and the period under consideration, the normal distribution assumption fits better in our VaR forecasting exercise.

It should be noted that the ranking of our models is close to the one we obtained using the violation ratios. However, the usage of our loss function enables us to distinguish between close models as we now consider not only the number but also the average magnitude of the violations.

With regard to estimation time, the processing time required to estimate our VaR forecasts with PSIs was at least half of the time required to determine the artificial algorithms benchmarks.

5. CONCLUDING REMARKS

In this article, we study the use of a class of neural network models, PSI, when applied to the task of forecasting the VaR of Brent oil and gold bullion using Open-High-Low-Close data. The results were benchmarked with those of two different neural network designs, the MLP and the RNN architectures, a GP algorithm, an EVT model in addition to a ARMA-GJR (1,1) model and the Riskmetrics volatility.

Our results show that the PSI networks demonstrate a better forecasting performance passing the Christoffersen tests and achieving lower realizations on our loss function at both long and short positions. The RNNs and the GPs

also provide a good performance when measured by realizations on our loss function. On the other hand, the MLPs present the worst performance compared to their NNs and GP benchmarks in our backtesting procedure. Moreover, our EVT estimation was successful only for oil, due to the few extreme events in our gold dataset. The results of the other two statistical models considered, the RiskMetrics and ARMA-GJR(1,1), were disappointing as in most cases they failed to pass the three Christoffersen tests.

Our results should go some way towards convincing a growing number of quantitative risk managers to experiment beyond the bounds of the more traditional risk models. Moreover, our proposed loss function adds another tool to risk managers in the difficult task of evaluating a VaR model. **Two features more necessary than ever before in today's volatile markets.**

APPENDIX

A.1 Neural Networks Training Procedure

The set of inputs of each NN and GP model is crucial for their pattern recognition and forecasting capabilities. Unfortunately there is no formal theory to assist this selection and practitioners need to conduct a sensitivity analysis to a pool of potential variables to help their decision (Zhang (2009)). In our application, we considered as potential inputs autoregressive lags up to the order of 20 of the daily volatility of the series under study, the daily volatility of other related commodities (such as Copper and Silver), the daily volatility of stock indices (such as the FTSE 100 and NYMEX), the daily volatility of exchange rates (such as the EUR/USD and the USD/JPY) and the RiskMetrics Volatility of the two series under study. This set of potential inputs was created based on the relevant literature problems (see amongst others Dunis and Chen (2005), Zhang (2009) and Chen and Hsieh (2010)).

The aim of our sensitivity analysis to this set of variables was to select the group of inputs for each asset and model that provided the most accurate one day ahead realised volatility forecasts in terms of RMSE, MAE and MAPE in the in-sample sub-period. As experimentation can be unlimited we followed the guidelines of Koza (1998), Dunis and Chen (2005) and Zhang (2009) in determining the number of potential inputs. The inputs of our artificial intelligence and genetic programming models for gold and oil are presented in table 7 below.

Insert Table 7

A.2 Networks specifications

In tables 8 and 9 below we present the characteristics of the networks used in this paper for gold. We selected the parameters that maximized the statistical performance of our models in the in-sample period.

Insert Table 8

Insert Table 9

The characteristics of the networks used in this paper for oil are presented in the following tables.

Insert Table 10

Insert Table 11

A.2 Christoffersen Test Results

Insert Table 12

Insert Table 13

Insert Table 14

Insert Table 15

REFERENCES

Bali, T. (2003), 'An Extreme Value Approach to Estimating Volatility and Value at Risk', *Journal of Business*, 76, 1, 83-108.

Balkema, A. and de Haan, L. (1974), 'Residual Life Time at Great Age', *Annals of Probability*, 2, 792-804.

Bartlmae, K. and Rauscher, F. (2000), 'Measuring DAX Market Risk: A Neural Network Volatility Mixture Approach', Presentation at the FFM2000 Conference, London, 31 May-2 June.

Basel (1996), 'Overview of the Amendment to the Capital Accord to Incorporate Market Risk', Basel Committee on Banking Supervision, Basel.

Bekaert, G. and Wu, G. (2000), 'Assymetry Volatility and Risk in Equity Markets', *The Review of Financial Studies*, 13, 1-42.

Blanco, C. and Ihle, G. (1999), 'How good is your VaR? Using Backtesting to Assess System Performance', *Financial Engineering News*, 11, August 1-2.

Brooks, C. (2002), 'Introductory Econometrics for Finance', Cambridge University Press, Cambridge.

Cakici, N. and Foster, K. (2003), 'Value at Risk for Interest Rate-Dependent Securities', *Journal of Fixed Income*, 12, 4, 81-96.

- Caporin, M. (2003), 'Evaluating Value-at-Risk Measures in Presence of Long Memory Conditional Volatility', *GRETA working paper n. 05.03*.
- Chan, L. and Lien, D. (2003), 'Using High, Low, Open, and Closing prices to Estimate the Effects of Cash Settlement on Futures Prices', *International Review of Financial Analysis*, 12, 35-47.
- Chen, C. and Hsieh, C. (2010), 'Measuring of Value at Risk (VAR) on Emerging Stock Markets by Neural Networks Method', *2010 International Symposium on Computer Communication Control and Automation (3CA)*, 137 – 140, Tainan, Taiwan.
- Chen, S. (2002), *Genetic Algorithms and Genetic Programming in Computational Finance*, Kluwer Academic Publishers, Amsterdam.
- Christoffersen, P. (1998), 'Evaluating Interval Forecasts', *International Economic Review*, 39, 4, 841-864.
- Dunis, C. and Chen, Y. (2005), 'Alternative Volatility Models for Risk Management and Trading: Application to the EUR/USD and USD/JPY Rates', *Derivatives Use, Trading & Regulation*, 11, 2, 126-156.
- Dunis, C., Laws, J. and Sermpinis, G. (2010), 'Modelling Commodity Value at Risk with Higher Order Neural Networks', *Applied Financial Economics*, 20, 7, 585-600.
- Elman, J. L. (1990), 'Finding Structure in Time', *Cognitive Science*, 14, 179-211.
- Fantazzini, D. (2009), 'The Effects of Misspecified Marginals and Copulas on Computing the Value at Risk: A Monte Carlo study', *Computational Statistics & Data Analysis*, Elsevier, vol. 53(6), pages 2168-2188.
- Ferreira C. (2006), *Gene Expression Programming: Mathematical Modelling by an Artificial Intelligence*, Springer, San Francisco.
- Floros, C. (2009), 'Modelling Volatility Using High, Low, Open and Closing Prices: Evidence from Four S&P Indices', *International Research Journal of Finance and Economics*, 28, 198-206.
- Garman, M. and Klass, M. (1980), 'On the Estimation of Security Price Volatilities from Historical Data', *Journal of Business*, 53, 67–78.
- Gencay, R. and Selcuk, F. (2004), 'Extreme Value Theory and Value-at-Risk: Relative Performance in Emerging Markets', *International Journal of Forecasting*, 20, 2, 287-303.
- Ghazali, R., Hussain, A. and Merabti, M. (2006), 'Higher Order Neural Networks for Financial Time Series Prediction', *The 10th IASTED*

International Conference on Artificial Intelligence and Soft Computing, 119-124, Palma de Mallorca, Spain.

Ghosh, J. and Shin, Y. (1992) 'Efficient Higher-Order Neural Networks for Classification and Function Approximation', *International Journal of Neural Systems*, 3, 4, 323-350.

Gilli, M. and Kellezi, E. (2006), 'An Application of Extreme Value Theory for Measuring Financial Risk', *Computational Economics*, 27, 1, 1-23.

Glosten, L., Jagannathan, R. and Runkle, D. (1993), 'On the Relation Between the Expected Value and the Volatility of the Nominal Excess Returns on Stocks', *The Journal of Finance*, 48, 1779-180.

Ho, L., Burridge, P., Cadle, J. and Theobald, M. (2000), 'Value-at-risk: Applying the Extreme Value Approach to Asian markets in the Recent Financial Turmoil', *Pacific-Basin Finance Journal*, 8, 249-275.

Hosking, J. and Wallis, J. (1987), 'Parameter and Quantile Estimation for the Generalized Pareto Distribution', *Technometrics*, 29, 3, 339-349.

Huang, A. and Tseng, T. (2009) 'Forecast of value at risk for equity indices: an analysis from developed and emerging markets', *The Journal of Risk Finance*, 10, 4, 393 – 409.

Hussain, A., Ghazali, R. and Al-Jumeily, D. (2006), 'Dynamic Ridge Polynomial Neural Network for Financial Time Series Prediction', *IEEE International conference on Innovation in Information Technology*, IIT06, Dubai.

Jorion, P. (1997), *Value at Risk*, Irwin Professional Publishing, Chicago.

Kaastra, I. and Boyd, M. (1996), 'Designing a Neural Network for Forecasting Financial and Economic Time Series', *Neurocomputing*, 10, 215-236.

Koza, J.R. (1998), 'Genetic Programming', 29-43, in Williams, J. G. and Kent, A., [eds.] *Encyclopedia of Computer Science and Technology*, NY: Marcel-Dekker, New York.

Kupiec, P. (1995), 'Technique for Verifying the Accuracy of Risk Measurement Models', *Journal of Derivatives*, 3, 2, 73-84.

Lee, T. and Saltoglu, B. (2001), 'Evaluating the Predictive Performance of Value-at-Risk Models in Emerging Markets: A Reality Check', *Mimeo, University of California, Riverside*.

Lisboa, P. J. G. and Vellido, A. (2000), 'Business Applications of Neural Networks', vii-xxii, in P. J. G. Lisboa, B. Edisbury and A. Vellido [eds.] *Business Applications of Neural Networks: The State-of-the-Art of Real-World Applications*, World Scientific, Singapore.

Liu, Y. (2005), 'Value-at-Risk Model Combination Using Artificial Neural Networks', *Emory University Working Papers*.

Locarek-Junge, H. and Prinzler, R. (1998), 'Estimating Value-at-Risk Using Neural Networks', *Application of Machine Learning and Data Mining in Finance*, ECML'98 Workshop Notes, Chemnitz.

Lopez, J. (1998), 'Methods for Evaluating Value-At-Risk Estimates', *Federal Reserve Bank of New York Research Paper n. 9802*.

McNees, S. (1995), 'Forecast Uncertainty: Can It Be Measured?', *Mimeo, Federal Reserve Bank of Boston*.

Neely, C. and Weller, P. (2002), 'Predicting Exchange Rate Volatility: Genetic Programming versus GARCH and RiskMetrics™', *Federal Reserve Bank of St. Louis*, 84, 3, 43–54.

Ozun, A. and Cifter, A. (2007), 'Nonlinear Combination of Financial Forecasts with Genetic Algorithm', *MPRA Paper 2488*, University Library of Munich, Germany.

Parkinson, M. (1980), 'The Extreme Value Method for Estimating the Variance of the Rate of Return', *Journal of Business*, 53, 61–65.

Picklands, J. (1975), 'Statistical Inference Using Extreme Value Order Statistics', *Annals of Statistics*, 3, 119-131.

Rau-Bredow, H. (2004), 'Value at Risk, Expected Shortfall, and Marginal Risk Contribution', *Risk Measures for the 21st Century*, Szegö, G.(ed.) , Wiley Finance, 61-68, Chichester.

Rogers, G. and Satchell, E. (1991), 'Estimating Variance from High, Low, and Closing prices', *Annals of Applied Probability*, 1, 504–512.

Rogers, G., Satchell, E. and Yoon, Y. (1994), 'Estimating the Volatility of Stock Prices: A Comparison of Methods that Use High and Low Prices', *Applied Financial Economics*, 4, 241–247.

Samuel, Z. (2008), 'Value at Risk and Conditional Extreme Value Theory via Markov Regime Switching Models', *The Journal of Futures Markets*, 28, 2, 155-181.

Sadorsky, P. (2005), 'Stochastic volatility forecasting and risk management', *Applied Financial Economics*, 15, 2, 121-135.

Sarma, M., Thomas, S. and Shah, A. (2003), 'Selection of Value-at-Risk Models', *Journal of Forecasting*, 22, 4, 337-358.

Shapiro, A. F. (2000), 'A Hitchhiker's Guide to the Techniques of Adaptive Nonlinear Models', *Insurance, Mathematics and Economics*, 26, 119-132.

Shin, Y. and Ghosh, J. (1991) 'The Psi-Sigma Network: An Efficient Higher-Order Neural Network for Pattern Classification and Function Approximation', *Proceedings IJCNN*, Seattle, July, 13-18.

Singh, V. and Guo, H. (1995), 'Parameter Estimation for 3-Parameter Generalized Pareto Distribution by the Principle of Maximum Entropy (POME)', *Journal des Sciences Hydrologiques*, 40, 2, 165-181.

Tenti, P. (1996), 'Forecasting Foreign Exchange Rates Using Recurrent Neural Networks', *Applied Artificial Intelligence*, 10, 567-581.

Vecci, L., Piazza, F. and Uncini, A. (1998), 'Learning and Approximation Capabilities of Adaptive Spline Activation Neural Networks', *Neural Networks*, 11, 259-270.

Yang, D. and Zhang, Q. (2000), 'Drift-Independent Volatility Estimation Based on High, Low, Open, and Close Prices', *Journal of Business*, 73, 3, 477-491.

Zhang, M. (2009), '*Artificial Higher Order Neural Networks for Economics and Business*', IGI Global, Hershey.