

# Constructing A Managed Portfolio Of High Frequency LIFFE Futures Positions

D L Toulson, S P Toulson, A Sinclair

Intelligent Financial Systems Limited  
66-68 Haymarket, London SW1Y 4RF Email: [ifs@if5.com](mailto:ifs@if5.com)  
Tel: (020) 7839 1863 <http://www.if5.com>

## 1. Introduction

In this Chapter, we explore the practical elements involved in the creation of a high frequency quantitative portfolio management model. The managed portfolio consists of a set of positions in 5 LIFFE futures contracts. The risk of the overall futures portfolio is managed such that the expected return volatility (risk) of the positions in the 5 future contracts matches the expected risk of a simple buy-and-hold strategy for the FTSE-100 index. The optimal portfolio weightings (futures positions) is assessed intra-daily and the portfolio re-balanced every 240 minutes during normal LIFFE trading hours, assumed to be 8:30AM – 4:05PM.

The first step in the construction of the portfolio management model involves the acquisition and pre-processing of the historical price data used to produce the forecasting models. This process is described in some detail in Section 2. In this study, we examine over 2 years of historical tick-by-tick data obtained from the five futures markets. This time period contains over 5,000,000 individual bid/ask quotes and trades. We begin by transforming the irregularly recorded tick-by-tick data into a minute-by-minute representation and describe the process by which this transformation is achieved. We conclude this Section with a simple analysis of the descriptive statistics and auto-correlation structure of the constructed 15-minutely prices in each of the five futures contracts.

We next describe the building of prediction models for each of the futures. In recent work (Toulson & Toulson, 199a, 1996b, 1997) we have demonstrated the combined use of the Discrete Wavelet Transform (DWT) (Daubechies, 1992; Chui, 1992; Szu & Telfer, 1992) and neural networks applied to the task of financial forecasting. This has resulted in the development of the Wavelet Encoding A Priori Orthogonal Network (WEAPON) architecture. This architecture is based on the familiar Multi Layer Perceptron (MLP) (Rummelhart et al, 1986) neural network architecture, but is enhanced through the use of a layer of *wavelet* processing nodes. The *complexity* of the network is controlled through the use of Bayesian regularisation techniques (MacKay, 1992; Williams, 1993). In Section 3, we give a brief description of the key features of the WEAPON architecture and describe its use in the prediction of the conditional mean of the expected returns of the five futures contracts.

In Section 4, we proceed to examine prediction models composed of committees of WEAPON networks for each of the five futures used to estimate both the expected future return and expected future volatility (risk) of each future. For comparison purposes, we also estimate the future volatility of the markets using a more standard GARCH technique (Bollerslev, 1986). The prediction models are built using historical data covering the period January 1995 to March 1996. The prediction models are then used to manage a portfolio consisting of positions in the five futures over the period April 1996 – April 1997. Section 5 describes how the portfolio management is performed using a standard Markowitz Mean-Variance approach (Markowitz, 1959). The WEAPON predictors provide the required estimates for the conditional mean of the five futures in the managed portfolio. The conditional variance is determined by a GARCH model for each of the five futures. The correlations of the returns between the five futures are estimated using a simple exponentially weighted moving average.

Finally, Section 6 examines in considerable detail, simulation results for managing the futures portfolio over 6 months of unseen test data. In particular we give details of the overall return performance, Sharpe Ratio, maximum draw-down and volatility of the managed portfolio.

## 2. The Historical Data

### 2.1 Partitioning the price data.

The first step in any quantitative analysis or trading model simulation involving financial time series is to obtain a suitable set of historical price data and to then define precisely how that data is to be used. In this study the historical price data is firstly used to develop and parameterise the forecasting models for the estimation of future returns and risks in our chosen markets. Secondly, the historical price data is used to obtain simulated results for the overall performance of the portfolio management model based on these forecasting models. To obtain genuine *ex ante* simulated trading performance, we define four clear partitions of the historical price data:

1. **Training data** – This portion of the historical price data is used to formulate forecasting strategies, and to parameterise or *train* the forecasting models.
2. **Validation data** – This data is used to validate (i.e. predict the out-of-sample generalisation ability) the forecasting models. It may be used more than once in an iterative forecasting model development cycle.
3. **Optimisation data** – This portion of the data is used to test different portfolio management strategies and optimise any parameters that may be associated with the portfolio management model. It usually does not contain training data (as this would produce biased forecasting accuracies) but may contain validation data.
4. **Test data** – This is the data that the final optimised portfolio management model is applied to, **only once**, to obtain the simulated real trading performance.

The historical price data used in this work consists of tick-by-tick records from five LIFFE futures contracts (shown in Figure 1 below), namely BTP (Italian Government Bond), Bund (German Government Bond), FTSE-100 index, JGB (Japanese Government Bond) and Long Gilt.

The historical data cover the period January 1<sup>st</sup> 1995 – September 1<sup>st</sup> 1997. The four data periods defined above, were partitioned as shown in Table 1.

*Table 1: The four data periods used in developing and testing a portfolio management model.*

<b>Period</b>	<b>Start</b>	<b>End</b>
Training	July 1 <sup>st</sup> , 1995	Dec 31 <sup>st</sup> , 1995
Validation	January 1 <sup>st</sup> , 1995	June 30 <sup>th</sup> , 1995
Optimisation	January 1 <sup>st</sup> , 1996	June 30 <sup>th</sup> , 1996
Test	July 1 <sup>st</sup> , 1996	September 1 <sup>st</sup> , 1997.

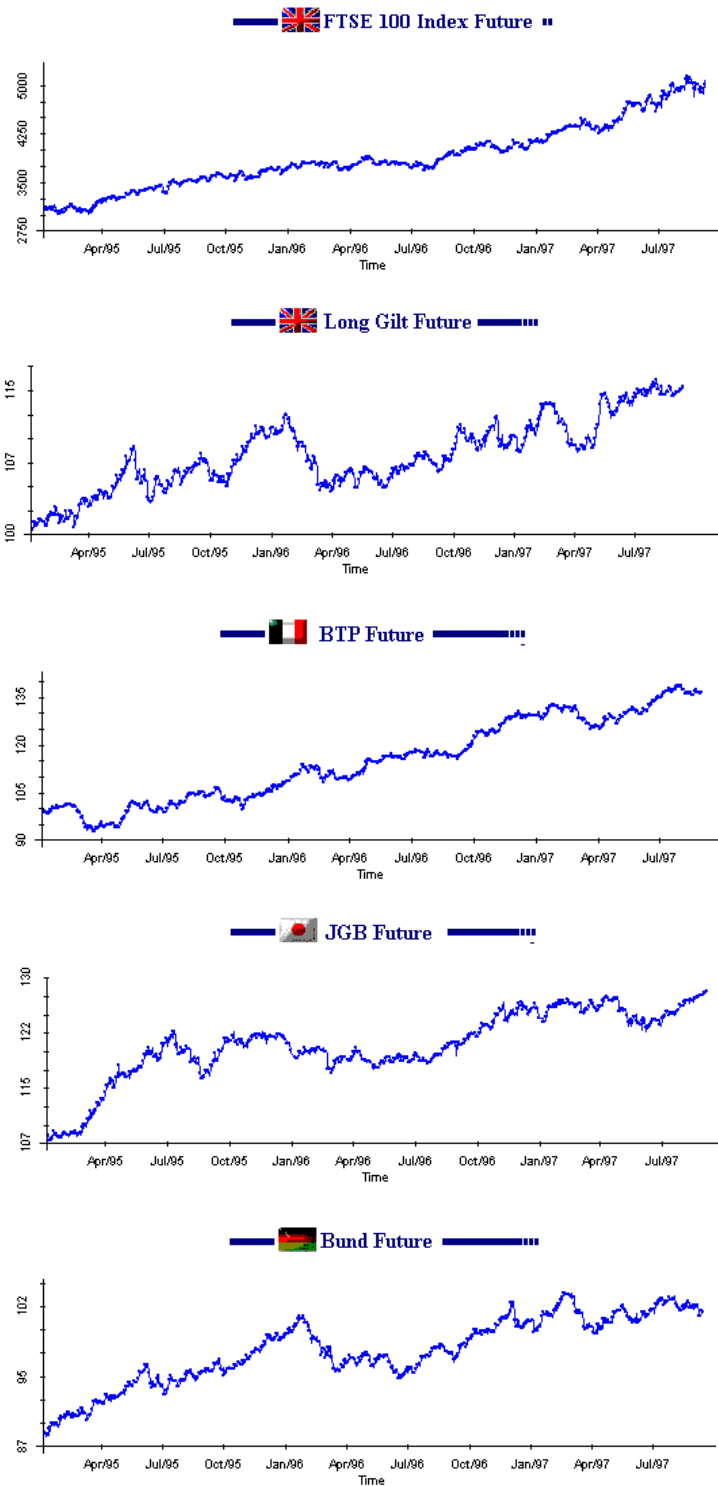


Figure 1: Price histories of the five futures contracts used in the portfolio.

## 2.2 Pre-processing the data

The historical price data used in this study consists of over 2 ½ years of tick-by-tick records from 5 LIFFE futures contracts. We refer to the tick data using the following notation. Each tick,  $i$ , on a particular market is denoted by four values, a time stamp  $T(i)$  expressed in seconds, a price  $P(i)$ , a volume  $V(i)$  and a type  $H(i)$  where  $H(i) \in Q$ ,  $Q = \{bid, ask, trade\}$ . The ticks do not have any particular spacing between their time-stamps, and may contain quite large discontinuities, for instance at the beginning and end of each trading day.

To build the forecasting and portfolio management models we convert the raw tick-by-tick data into a set of evenly spaced 15-minute price samples. We do this by forming volume weighted traded price averages within these 15-minute time bars. To illustrate this, we define the 15-minute volume weighted traded price average at time  $t$  to be

$$\tilde{p}_t = \frac{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=trade}} P(i)v(i)}{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=trade}} v(i)} \quad (1)$$

From the unevenly spaced tick data, we can obtain a set of evenly spaced fifteen-minute volume weighted price averages for the duration of each trading day. There still exist, however, obvious discontinuities (in time) at the beginning and end of each trading day.

Difficulties occur in intervals in which the overall traded volume is zero. We use the following strategies to cater for different possible situations in which no tradable price ticks were recorded in a particular 15-minute interval.

### Strategy 1 – Bids and Asks exist

In some intervals in which there is no actual traded volume, there may, however, exist a number of bid and ask ticks. In such cases we define the price for that interval to be the simple average of the mean bid,  $\tilde{b}_t$ , and mean ask prices,  $\tilde{a}_t$ , i.e.

$$\tilde{b}_t = \frac{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=bid}} P(i)}{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=bid}} 1}, \quad \tilde{a}_t = \frac{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=ask}} P(i)}{\sum_{\substack{i,t < T(i) < t+300, \\ H(i)=ask}} 1}, \quad \tilde{p}_t = \frac{\tilde{a}_t + \tilde{b}_t}{2} \quad (2)$$

### Strategy 2 – Either Bids or Asks exist

In other cases, it may be that either only bids or only asks exist in the 15-minute interval, i.e. perhaps three bids but no corresponding asks. In this case we take the values we have (for example the 3 bids) and either add or subtract half of the mean bid-ask spread measured from the previous 20 intervals. So in the case of bids but no asks,

$$\tilde{p}_t = \tilde{b}_t + \frac{\tilde{s}_t}{2}, \quad (3)$$

where  $\tilde{s}_t$  is the mean bid-ask spread at time  $t$ . In the case of asks only

$$\tilde{p}_t = \tilde{a}_t - \frac{\tilde{s}_t}{2} \quad (4)$$

### **Strategy 3 – No ticks at all.**

In the case in which there are no recorded ticks at all in the 15-minute period, we simply carry forward the price of the previous 15-minute interval, i.e.

$$\tilde{p}_t = \tilde{p}_{t-1} \quad (5)$$

#### **NB: Tradability of prices**

The 15-minute weighted average prices described above are used for *information and model building purposes only*. That is once, actual trading model simulations are performed, the system does **not** trade the averaged 15-minute prices. What occurs instead is that if, for instance, a long position is recommended by the system at a particular time then the ‘price’ actually traded is the price of *the next recorded ask*. Similarly, if a short position is required, then the price used is that for the next available bid. The time lag between the system recommending a position and the system actually taking that position may thus be anything between a couple of seconds in active trading and up to several minutes in sparse trading. Also, the system does not ‘know’ in advance the price it will receive for a particular desired trade.

## **2.3 Basic Statistics**

For each of the 5 contracts, the summary statistics for 15-minute returns were estimated. Table 2 shows the mean, standard deviation, skewness and kurtosis for each of the contracts. All 15-minute returns distributions are slightly positively skewed. The kurtosis values show that, as expected, all distributions exhibit fat tails.

*Table 2: The descriptive statistics for the 15-minutely returns for each of the five futures markets.*

<b>Future</b>	<b>Mean</b>	<b>SD</b>	<b>Skewness</b>	<b>Kurtosis</b>
BTP	0.0012	0.1016	0.3745	28.09
Bund	0.0015	0.0537	0.5996	39.24
FTSE 100 Index	0.0025	0.1198	0.7543	13.82
JGB	0.0013	0.0718	0.4533	93.82
Long Gilt	0.0013	0.1213	0.1298	19.33

Figure 2 depicts the autocorrelation structure of the returns for the first 15 lags for each of the contracts. Considering only the first ten lags, it can be seen that for all but one market only the first lag exhibits significant autocorrelation. The exception is the Long Gilt, for which the first four lags are significant. However, the first two lags are much more significant than the last two. The autocorrelation structure is considered in more detail in Section 5 where we develop combined AR(p)-GARCH(1,1) models to estimate the conditional variance of the 15-minute returns.

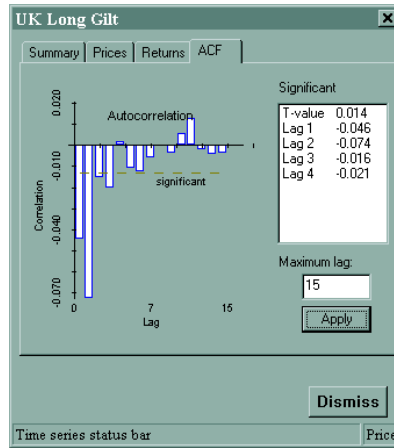
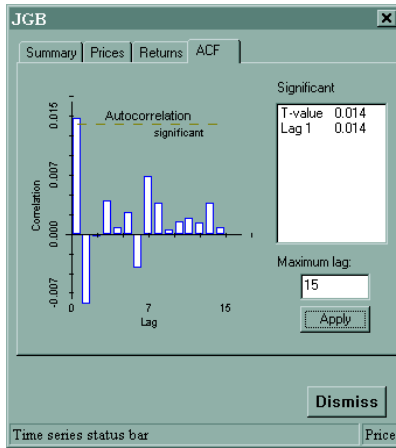
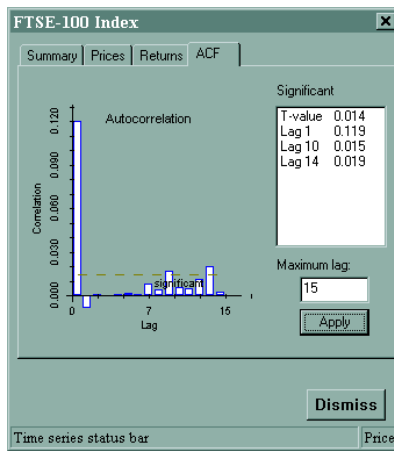
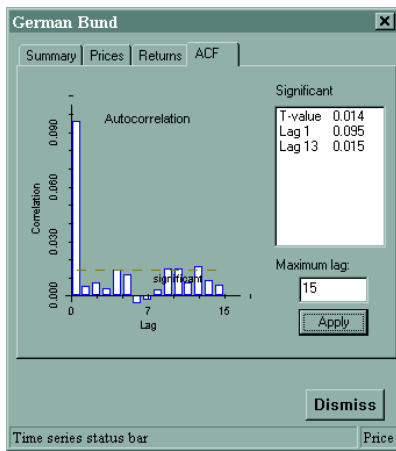
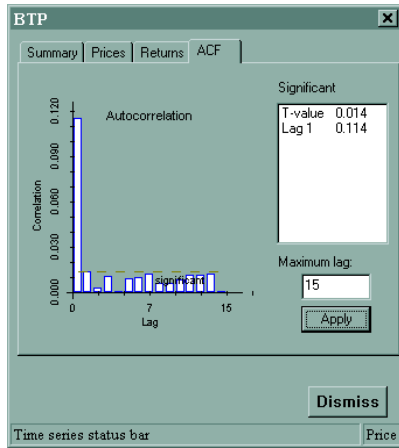


Figure 2: The autocorrelation structure of the 5 futures markets showing the significant lags (based on a 95% confidence band) in the right-hand box of each screenshot.

### 3. Forecasting Conditional Returns

#### 3.1 Background

Our aim is to develop a portfolio management model that attempts to maximise the return whilst managing the risk of a portfolio of positions taken in 5 futures contract. The risk management is performed using a fairly standard Mean-Variance (Markowitz) model adjusted for transaction charges (for more detail see Section 5). A key requirement for the success of the Mean-Variance model is the accurate and robust estimation of the future conditional returns, risks and correlations for each of the assets in the portfolio (in this case, the five futures contracts). In much of the financial literature, these estimates tend to be based on simple historical moving averages. Such estimates are suitable when managing a portfolio over long time periods in which case, short term market trends may be effectively ignored. However, in this paper we are looking at a portfolio management model that re-balances a portfolio of futures positions every 240-minutes and hence shorter term, possibly non-linear forecasting techniques may be more suitable.

There has recently been much interest in attempting to improve the accuracy of future conditional return and risk estimates using non-linear neural networks (primarily for conditional return estimation), and (G)ARCH and stochastic volatility methods for volatility (risk) estimates. In this section we describe the combined use of wavelets and neural networks to build 4-hour ahead prediction models for future *returns* for each of the 5 contracts.

#### 3.2 Forecasting Returns Using Neural Networks

In this section we examine the use of *committees* of neural network models to predict future 4-hourly returns. We use as input to each of the neural networks the previous 96 lagged 15-minute returns for each of the five markets (corresponding to three full 480-minute trading days). The required output of the neural network models is the predicted future conditional return for a particular market 4 hours into the future. This process is illustrated in Figure 3.

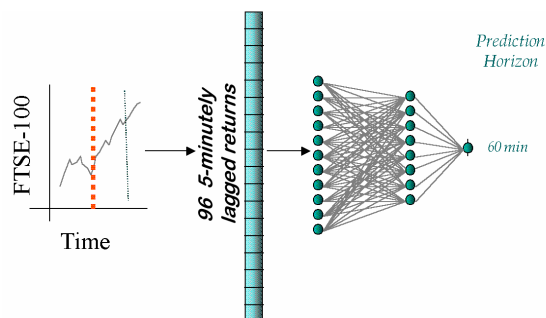


Figure 3: Predicting future returns for each of the five LIFFE future markets. For instance, 96 15-minutely-lagged returns are input to the neural network for the FTSE-100 Index future. The network then outputs a prediction for the future conditional 4-hour return.

A key consideration in this type of prediction strategy is the encoding of the 96 lagged 15-minute returns as a neural network input vector. One possibility is to simply use all 96 *raw* inputs. The problem with this approach is the relatively high dimensionality of the input vectors. This requires the use of an extremely large set of training examples to ensure that the parameters of the model (the weights of the neural network) may be properly determined. Due to computational complexities and the non-stationarity of

financial time series the use of extremely large training sets is seldom practical. A preferable strategy is to reduce the dimension of the input information to the neural network.

A popular approach to reduce the dimension of inputs for neural networks is Principal Components Analysis (PCA). PCA is used to reduce redundancy in the input vectors due to inter-component correlations. However, as we are working with lagged returns from a single high frequency financial time series we know, in advance, that there is little (auto) correlation in the lagged returns. In other work (Toulson & Toulson, 1996a, 1996b), the problem of dimension reduction was addressed by the use of Canonical Discriminant Analysis (CDA) techniques. These techniques were shown to lead to significantly improved performance in terms of prediction ability of the trained neural networks. However, such techniques do not, in general, take any advantage of our knowledge of the temporal structure of the input components, which are sequential lagged returns. Such techniques are also implicitly linear in their assumptions of separability, which may not be generally appropriate when considering inputs to (non-linear) neural network prediction models. We consider, as an alternative means of reducing the dimension of the input vectors, the use of the discrete wavelet transform (DWT).

### 3.3 The Discrete Wavelet Transform (DWT)

#### 3.3.1 Background

The Discrete Wavelet Transform (Telfer et al, 1995; Meyer, 1995) has recently received much attention as a powerful technique for the pre-processing of data in applications involving both the compact representation of the original data (i.e. data compression or factor analysis) or as a discriminatory basis for pattern recognition and regression problems (Casasent & Smokelin, 1994; Szu & Telfer, 1992). The transform operates by projecting the original signal onto a sub-space spanned by a set of *child wavelets* derived from a particular Mother wavelet (see Figure 4). The projection of the original signal into the lower dimensional sub-space can then be used, for instance, as a compressed input representation for a neural network.

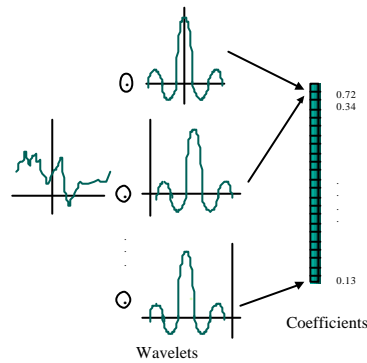


Figure 4: The discrete wavelet transform. The time series is convolved with a number of child wavelets characterised by different dilations and translations of a particular mother wavelet.

#### 3.3.2 Definition

Let us select the Mother wavelet to be the Mexican Hat function

$$y(t) = \frac{2}{\sqrt{3}} p^{\frac{1}{4}} (1-t^2) e^{-\frac{t^2}{2}}. \quad (6)$$

The wavelet *children* of the Mexican Hat Mother are the dilated and translated forms of (6), i.e.

$$f^{t,z}(t) = \frac{1}{\sqrt{z}} f\left(\frac{t-t}{z}\right) \quad (7)$$

Now, let us select a finite subset  $C$  from the infinite set of possible child wavelets. Let the members of the subset be identified by the discrete values of position  $t_i$  and scale  $z_i$ ,  $i = 1, \dots, K$ ,

$$C = \{t_i, z_i \quad i = 1, \dots, K\} \quad (8)$$

where  $K$  is the number of children.

Suppose we have an  $N$  dimensional discrete signal  $\mathbf{x}$ . The  $j^{\text{th}}$  component of the projection of the original signal  $\mathbf{x}$  onto the  $K$  dimensional space spanned by the child wavelets is then

$$y_j = \sum_{i=1}^N x_i f^{t_i, z_i}(i) \quad (9)$$

The reduced dimension vector  $\mathbf{y} = (y_1, y_2, \dots, y_K)$  is then the wavelet transform of the original signal,  $\mathbf{x}$ .

### 3.3.2 Choice of Child Wavelets

The significant questions to be answered with respect to the practical using of the DWT to reduce the dimension of input vectors to a neural network are:

1. How many child wavelets should be used?
2. What values of  $t_i$  and  $z_i$  should be chosen given the number of child wavelets?

For ‘representational’ problems, the child wavelets are generally chosen such that together they constitute a **wavelet frame**. There are a number of known Mother functions and choice of children that satisfy this condition (Debauchies, 1988). With such a choice of mother and children, the projected signal will retain all of its original information (in the Shannon sense, Shannon, 1948), and reconstruction of the original signal from the projection will be possible. There are a variety of conditions that must be fulfilled for a discrete set of child wavelets to constitute a frame, the most intuitive being that the number of child wavelets must be at least as great as the dimension of the original discrete signal. However, the choice of the optimal set of child wavelets becomes more complex in discrimination or regression problems. In such cases, reconstruction of the original signal is not relevant and the information we wish to preserve in the transformed space is the information that *distinguishes* different classes of input signal (i.e. positive and negative future returns).

In the following section, we present a method of choosing a suitable set of child wavelets such that the transformation of the original data (the 96 lagged 5-minute returns of the five future markets) enhances the non-linear separability of different classes of signal whilst significantly reducing the dimension of the data. We show how this may be achieved naturally by implementing the wavelet transform as a set of *wavelet neurons* contained in the first layer of a multi-layer perceptron (Rummelhart et al, 1986) (*henceforth R86*). The shifts and dilations of the wavelet nodes are found along with the other neural network parameters through the minimisation of a penalised least squares objective function. We extend this concept to include automatic determination of a suitable number of wavelet nodes by applying Bayesian priors on the child wavelet parameters during training of the neural network and enforcing orthogonality between the wavelet nodes using soft constraints.

### 3.4 The Wavelet Encoding A Priori Orthogonal Network (WEAPON)

We now derive a neural network architecture that includes wavelet neurons in its first hidden layer (WEAPON). We begin by defining the wavelet neuron and its use within the first layer of the WEAPON architecture. We then derive a learning rule whereby the parameters of each wavelet neuron (dilation and position) may be optimised with respect to the accuracy of the network's predictions. Finally, we consider issues such as wavelet node orthogonality and choice of the optimal number of wavelet nodes to use in the architecture (skeletonisation).

#### 3.4.1 The Wavelet Neuron

The most common activation function for neurons in the Multi-Layer Perceptron architecture is the sigmoidal activation function, see Equation (10).

$$\mathbf{j}(x) = \frac{1}{1 + e^{-\frac{x}{x_0}}} \quad (10)$$

The output of a neuron  $y_i$  is dependent on the activations of the nodes in the previous layer  $x_k$ , and on the weighted connections between the neuron and the previous layer  $\mathbf{W}_{k,i}$ , as shown in Equation (11).

$$y_i = \mathbf{j} \left( \sum_{j=1}^J x_j \mathbf{W}_{j,i} \right) \quad (11)$$

Due to the similarity between Equations (9) and (11), we can implement the Discrete Wavelet Transform as the first layer of hidden nodes of a multi layer perceptron (MLP). The weights connecting each wavelet node to the input layer  $\mathbf{W}_{j,i}$  must be constrained to be discrete samples of a particular wavelet child

$\mathbf{f}^{\mathbf{t}, \mathbf{z}_j}(i)$  and the activation function of the wavelet nodes should be the identity transformation  $\mathbf{j}(x) = x$ . In fact, we ignore the weights connecting the wavelet node to the previous layer and instead characterise the wavelet node purely in terms of values of translation and scale,  $\mathbf{t}$  and  $\mathbf{z}$ . The WEAPON architecture is shown below in Figure 5.

The WEAPON architecture is a standard four-layer MLP with a linear set of nodes in the first hidden layer and in which the weights connecting the input layer to the first hidden layer are constrained to be wavelets. This constraint on the first layer of weights acts to enforce our *a priori* knowledge that the input components are not presented in an arbitrary fashion, but in fact have a defined temporal ordering, i.e. they are, in fact, 96 sequential values of 15-minute lagged returns.

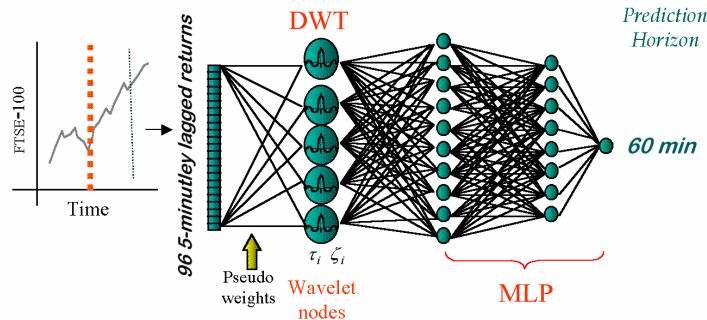


Figure 5: The WEAPON architecture

### 3.4.2 Training the Wavelet Neurons

The standard MLP is usually trained using error backpropagation (backprop) (R86) on a set of training examples. The most commonly used error function is simply the sum of squared error over all training samples  $E_D$

$$E_D = \sum_{i=1}^N |y_i - t_i|^2 \quad (12)$$

Backprop requires the calculation of the partial derivatives of the data error  $E_D$  with respect to each of the free parameters of the network (usually the weights and biases of the neurons). For the case of the wavelet neurons suggested above, the weights between the wavelet neurons and the input pattern are not free but are constrained to assume discrete values of a particular child wavelet. The free parameters for the wavelet nodes are therefore not the weights, but the values of translation and dilation  $\mathbf{t}$  and  $\mathbf{z}$ . To optimise these parameters during training, we must obtain expressions for the partial derivatives of the error function  $E_D$  with respect to these two wavelet parameters. The usual form of the backprop algorithm is shown in Equation (13).

$$\frac{\mathcal{J}E}{\mathcal{J}\mathbf{w}_{i,j}} = \frac{\mathcal{J}E}{\mathcal{J}y} \frac{\mathcal{J}y}{\mathcal{J}\mathbf{w}_{i,j}} \quad (13)$$

The term  $\frac{\mathcal{J}E}{\mathcal{J}y}$ , often referred to as  $\mathbf{d}_j$ , is the standard backpropagation of error term, which may be found

in the usual way for the case of the wavelet nodes. The partial derivative  $\frac{\mathcal{J}y}{\mathcal{J}\mathbf{w}_{i,j}}$  must be substituted with

the partial derivatives of the node output  $y$  with respect to the wavelet parameters. For a given Mother wavelet  $\mathbf{f}(x)$ , consider the output of the wavelet node, given in Equation (11). Taking partial derivatives with respect to the translation and dilation yields Equations (14).

$$\begin{aligned} \frac{\mathcal{J}y_j}{\mathcal{J}\mathbf{t}_j} &= \frac{\mathcal{J}}{\mathcal{J}\mathbf{t}_j} \left( \sum_{i=1}^N x_i \frac{1}{\sqrt{\mathbf{z}_j}} \mathbf{f} \left( \frac{i - \mathbf{t}_j}{\mathbf{z}_j} \right) \right) \\ &= - \sum_{i=1}^N x_i \frac{1}{\mathbf{z}_j^{\frac{3}{2}}} \mathbf{f}' \left( \frac{i - \mathbf{t}_j}{\mathbf{z}_j} \right) \\ \frac{\mathcal{J}y_j}{\mathcal{J}\mathbf{z}_j} &= \frac{\mathcal{J}}{\mathcal{J}\mathbf{z}_j} \left( \sum_{i=1}^N x_i \frac{1}{\sqrt{\mathbf{z}_j}} \mathbf{f} \left( \frac{i - \mathbf{t}_j}{\mathbf{z}_j} \right) \right) \\ &= - \frac{1}{2} \sum_{i=1}^N x_i \frac{1}{\mathbf{z}_j^{\frac{3}{2}}} \mathbf{f}' \left( \frac{i - \mathbf{t}_j}{\mathbf{z}_j} \right) - \sum_{i=1}^N x_i \frac{(i - \mathbf{t}_j)}{\mathbf{z}_j^{\frac{5}{2}}} \mathbf{f} \left( \frac{i - \mathbf{t}_j}{\mathbf{z}_j} \right) \end{aligned} \quad (14)$$

Using the above equations, it is possible to optimise the wavelet dilations and translations. For the case of the Mexican Hat wavelet we note that

$$\mathbf{f}(t) = -\frac{2}{\sqrt{3}}\mathbf{p} \frac{1}{2} 2t(2-t^2)e^{-\frac{t^2}{2}} \quad (15)$$

Once suitable expressions for the above have been derived, the wavelet parameters can be optimised in conjunction with the other parameters of the neural network by using any of the standard gradient based optimisation techniques.

### 3.4.3 Orthogonalisation of the Wavelet Nodes

A potential problem that might arise during the optimisation of parameters associated with the wavelet neurons is the duplication of parameters of some of the wavelet nodes. This leads to redundant correlations in the outputs of the wavelet nodes and hence to an overly complex final neural network model (in terms of number of parameters). One way of avoiding this type of parameter duplication is to apply a soft constraint of orthogonality on the wavelets of the hidden layer. This is done through the use of an additional error term in the standard data misfit function, as shown in Equations (16), (17) and (18).

$$E_W^f = \sum_{i=1, j \geq i}^N \langle \mathbf{f}^{t_i, z_i}, \mathbf{f}^{t_j, z_j} \rangle \quad (16)$$

where  $\langle \rangle$  denotes the projection

$$\langle f, g \rangle = \sum_{i=-\infty}^{\infty} f(i)g(i) \quad (17)$$

In the previous section, backprop error gradients were derived in terms of the unregularised sum of squares data error term,  $E_D$ . We now include an additional term for the orthogonality constraint to yield a combined error function  $M(W)$ , given by:

$$M(W) = \mathbf{a}E_D + \mathbf{g}E_W^f \quad (18)$$

To implement this new error term within the backprop training rule, we must derive the two partial derivatives of  $E_W^f$  with respect to the dilation and translation wavelet parameters  $\mathbf{z}_i$  and  $\mathbf{t}_i$ . Expressions for the partial derivatives above are obtained from (16) and are given by:

$$\begin{aligned} \frac{\partial E_W^f}{\partial \mathbf{t}_i} &= \sum_{j=1}^K \sum_{t=1}^N \mathbf{f}^{t_j, x_j}(t) \frac{\partial}{\partial \mathbf{t}_i} \mathbf{f}^{t_i, f_i}(t) \\ \frac{\partial E_W^f}{\partial \mathbf{x}_i} &= \sum_{j=1}^K \sum_{t=1}^N \mathbf{f}^{t_j, x_j}(t) \frac{\partial}{\partial \mathbf{x}_i} \mathbf{f}^{t_i, f_i}(t) \end{aligned} \quad (19)$$

These terms may then be included within the standard backprop learning algorithm. The ratio  $\frac{\mathbf{a}}{\mathbf{g}}$  determines the balance made between obtaining optimal training data errors against the penalty incurred by

the network containing overlapping or non-orthogonal nodes. The value of this ratio may either be estimated or optimised using the method of cross validation.

During training the orthogonalisation terms make the wavelet nodes *compete* with each other to occupy the most *relevant* areas of the input space with respect to the mapping being performed by the network. In the case of an excessive number of wavelet nodes in the hidden layer this process generally leads to the **marginalisation** of a number of wavelet nodes. The marginalised nodes are driven to areas of the input space in which little useful information with respect to the discriminatory task performed by the network is present (see Figure 6).

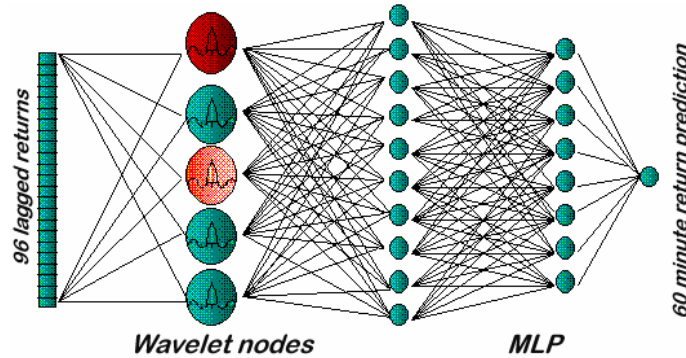


Figure 6: Marginalisation of wavelet nodes. In this case wavelet node 1 and 3 are marginalised.

### 3.4.4 Weight and Node Elimination

The *a priori* orthogonal constraints introduced in the previous section help to prevent significant overlap in the wavelets by encouraging orthogonality. However, *redundant* wavelet neurons remain in the hidden layer though they will have been marginalised to irrelevant (in terms of discrimination) areas of the time/frequency space. At best, these nodes play no significant role in modelling the data. At worst, they are used to model noise in the output targets and lead to poor generalisation performance. It is, therefore, preferable to eliminate these redundant nodes.

A number of techniques have been suggested in the literature for node and/or weight elimination in neural networks. We adopt the technique proposed by Williams (1993) and MacKay (1992a, 1992b) and use a Bayesian training technique, combined with a Laplacian prior on the network weights as a natural method of eliminating redundant nodes from the WEAPON architecture. The Laplacian Prior on the network weights implies an additional term in the previously defined error function (18), i.e.

$$M(W) = aE_D + gE_w^f + bE_w \quad (20)$$

where  $E_w$  is defined as

$$E_w = \sum_{i,j} |w_{i,j}| \quad (21)$$

A consequence of this prior is that during training, weights are forced to adopt one of two positions. A weight can either (1) adopt equal data error sensitivity as all the other weights, or (2) be forced to zero.

This leads to **skeletonisation** of the network. During this process, weights, hidden nodes or input components may be removed from the architecture. The combined effects of the soft orthogonality constraint on the wavelet nodes and the use of the Laplacian weight prior leads to what we term

**Marginalise and Murder (M&M)** training. At the beginning of training process, the orthogonality constraint forces certain wavelet nodes to *insignificant* areas of the input space with regards to the discrimination task being performed by the network. The weights emerging from these redundant wavelet nodes will then have little data error sensitivity and are forced to zero and deleted due to the effect of the Laplacian weight prior (see Figure 7).

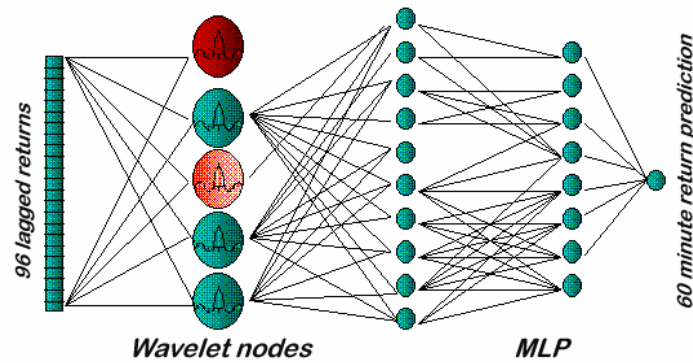


Figure 7: The WEAPON neural network after applying the Marginalise & Murder training algorithm. Wavelet node 1 is now completely disconnected and can be removed. wavelet node 3 has only one connection left.

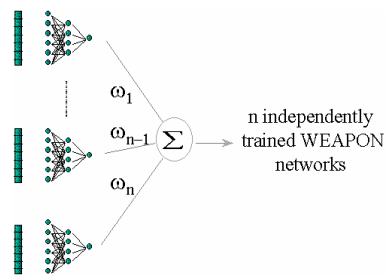
### 3.5 WEAPON Summary

In this section, we have developed a neural network prediction model in order to forecast future 4-hourly returns for the 5 LIFFE futures contracts on the basis of 96 lagged 15-minute returns. The neural network model used (WEAPON) operates by encoding the 96 lagged input returns using a Discrete Wavelet Transform. The model allows the shifts and dilations of the Wavelet Transform to be determined as part of the standard back-propagation of error training algorithm. A term is included in the learning rule to attempt to maintain a degree of orthogonality in the wavelet neurons during training. Also included in the training regime of the WEAPON model is a set of regularisation constraints on the weights of the network that prevent *over-training* which result in a model with better generalisation capability. These constraints are expressed as a Bayesian prior on the weights of the network. During training this prior (modelled as a Laplacian) tends to both limit the magnitude of the weights (associated with model complexity) and also to identify *redundant* weights that are eliminated from the model during training.

In the next section we report the practical results of training WEAPON networks to forecast 240-minute ahead returns for the 5 LIFFE future contracts.

## 4. Forecasting Results

To test the effectiveness of the WEAPON forecasting model, committee network prediction models are built for each of the 5 LIFFE futures. Each committee is composed of five WEAPON networks. To train each of the WEAPON networks, example input and output vectors are generated using 6 months of historical data. This yields approximately 5,000 training examples. Each training example consists of a 96-dimensional input vector of lagged 15-minute returns and a 1-dimensional target or output vector which contains the actual return 240 minutes ahead (see Figure 8). Each of the five WEAPON networks is initialised with a 96-20-20-1 architecture and then trained using the Marginalise & Murder training scheme described in the previous section. Training is continued until the combined data-misfit / orthogonality / weight magnitude error function is minimised. Optimal values for the hyper-parameters,  $\mathbf{a}$ ,  $\mathbf{b}$  and  $\mathbf{g}$  are obtained by cross validation.



*Figure 8: A committee of independently trained WEAPON networks. Each WEAPON network is trained to predict the four-hourly return for the associated futures market. The five resulting predictions are then combined using a simple arithmetic average to obtain a single one hour return prediction for the futures market.*

Once the 5 trained WEAPON architectures have been obtained for each futures market, the overall prediction model for that market is then defined to be a simple average of the predicted future conditional returns generated by each of the five independent WEAPON architectures. The forecasting abilities of the final trained committees are shown in Table 3 to 6. As a comparison, the accuracies of a simple, single MLP trained with cross validation and a weighted average forecasting model are also shown. The WEAPON networks quite clearly offer the best forecasting ability.

Table 3 : Results for FTSE 100 index future.

Architecture	% Accuracy	Large % Accuracy	RMSE
WEAPON committee	53.54%	54.24%	0.23432
Single MLP	51.26%	55.26%	0.24879
EWMA	52.13%	56.71%	0.24127

Figures are shown for predicting the one-hour return in terms of percentage turning point accuracy, large percentage turning point accuracy and root mean square prediction error.

Table 4 : Results for the Bund future.

Architecture	% Accuracy	Large % Accuracy	RMSE
WEAPON committee	53.37%	53.51%	0.0972
Single MLP	52.91%	51.73%	0.1062
EWMA	52.55%	53.29%	0.1045

Figures are shown for predicting the one-hour return in terms of percentage turning point accuracy, large percentage turning point accuracy and root mean square prediction error.

Table 5 : Results for BTP future.

Architecture	% Accuracy	Large % Accuracy	RMSE
WEAPON committee	53.85%	56.63%	0.1804
Single MLP	52.60%	53.08%	0.1922
EWMA	51.87%	51.98%	0.1901

Figures are shown for predicting the one-hour return in terms of percentage turning point accuracy, large percentage turning point accuracy and root mean square prediction error.

Table 6 : Results for JGB future.

Architecture	% Accuracy	Large % Accuracy	RMSE
WEAPON committee	50.34%	48.43%	0.0633
Single MLP	50.42%	49.65%	0.0659
EWMA	49.72%	49.30%	0.0642

Figures are shown for predicting the one-hour return in terms of percentage turning point accuracy, large percentage turning point accuracy and root mean square prediction error.

Table 7 : Results for Long Gilt future.

Architecture	% Accuracy	Large % Accuracy	RMSE
WEAPON committee	52.51%	56.04%	0.1843
Single MLP	49.98%	52.74%	0.1982
EWMA	51.59%	53.04%	0.1972

Figures are shown for predicting the one-hour return in terms of percentage turning point accuracy, large percentage turning point accuracy and root mean square prediction error.

## 5. Volatility Estimation

To manage a risk-controlled portfolio of positions we need estimates for the associated *risks* (predicted future volatilities or forecasting errors) of each contract traded. It is possible to extend the neural network model described in the previous sections to forecast future volatility as well as future conditional return. However, we have found the performance of non-linear neural networks in forecasting volatility to be rather unstable particularly in markets containing strong heteroskedasticity. Instead, in this section we examine the use of a combined AR(p)-GARCH(1,1) model for volatility forecasting.

### 5.1 Volatility Forecasting using AR(P)-GARCH(1,1)

Consider the basic linear AR(P) time series model

$$y_t = \mathbf{a}_0 + \mathbf{a}_1 y_{t-1} + \mathbf{a}_2 y_{t-2} + \dots + \mathbf{a}_p y_{t-p} + \mathbf{e}_t \quad (22)$$

The conditions on the noise process,  $\mathbf{e}_t$  for this model are that the noise should be independent identically distributed and have constant unconditional variance,  $\mathbf{s}^2$ , i.e.

$$E[\mathbf{e}_t] = 0$$
$$E[\mathbf{e}_t \mathbf{e}_{t-t}] = \begin{cases} \mathbf{s}^2 & \text{if } t = 0 \\ 0 & \text{otherwise} \end{cases} \quad (23)$$

Now, although the *unconditional* variance of  $\mathbf{e}_t$  is constant, it is possible to specify the *conditional* variance of  $\mathbf{e}_t$  to be non-constant. One possible way in which this non-constant conditional variance may be modeled is to use one of the families of Auto-Regressive Conditionally Heteroskedastic (ARCH) models. We consider the GARCH(1,1) model in which the noise process  $\mathbf{e}_t$  is modeled using

$$\mathbf{e}_t = \sqrt{h_t} w_t \quad (24)$$

where  $w_t$  is a zero mean unit variance white noise process and  $h_t$  is

$$h_t = \mathbf{b}_0 + \mathbf{b}_1 \mathbf{e}_t^2 + \mathbf{g}_1 h_{t-1} \quad (25)$$

It should be stressed that the GARCH(1,1) models magnitude of the conditional variances of the forecast residuals of the forecasting model (i.e. the AR(p) model in this case). The parameters of the GARCH(1,1) model must therefore be estimated in conjunction with the forecasting model (if any is used). However, in cases where the 'forecastability' of the time series is low (as is certainly the case with financial time series analysis) the prediction errors of a simple linear AR(P) model approximates the overall volatility of the time series. We thus can use the value of  $h_t$  as a proxy for the volatility of the time series.

### 5.2 Estimating the AR(P)-GARCH(1,1) Models

To obtain estimates for the parameters of AR(P)-GARCH(1,1) models we use the following 2stage procedure.

- First, given the partial autocorrelation functions for the 5 markets, suitable values for the *order* of the AR(p) models were selected.
- Second, parameters for the combined AR(p)+GARCH(1,1) models are estimated for each of the 5 contracts using Maximum Likelihood and the method of Berndt et al (1974).

Table 8 below shows the final parameter values with significance levels for each of the five markets obtained from examining price data for the combined training and validation periods.

Table 8 : AR(p)+GARCH(1,1) results for the five futures.

Future	Parameter	Value	SE
<b>BTP</b>	$a_0$	0.0053	0.00057
	$a_1$	0.1719	0.00945
	$b_0$	0.0041	0.00007
	$b_1$	0.6762	0.01347
	$g_1$	0.2290	0.01095
<b>Bund</b>	$a_0$	0.0017	0.00031
	$a_1$	0.1140	0.01296
	$b_0$	0.0005	0.00001
	$b_1$	0.4594	0.00980
	$g_1$	0.5543	0.00445
<b>FTSE 100 Index</b>	$a_0$	0.0014	0.00099
	$a_1$	0.1545	0.01272
	$b_0$	0.0034	0.00013
	$b_1$	0.2983	0.00843
	$g_1$	0.5407	0.01097
<b>JGB</b>	$a_0$	0.0014	0.00083
	$a_1$	0.0426	0.01246
	$b_0$	0.0004	0.00002
	$b_1$	-0.0026	0.00013
	$g_1$	0.9171	0.00441
<b>Long Gilt</b>	$a_0$	0.0012	0.00036
	$a_1$	0.0746	0.01198
	$a_2$	-0.0419	0.00843
	$a_3$	0.0221	0.00543
	$a_4$	0.0379	0.00396
	$b_0$	0.0048	0.00004
	$b_1$	1.994	0.04434
	$g_1$	0.187	0.00455

### 5.3 Forecasting Volatility

After parameterising the AR(P)-GARCH(1,1) models, a one-step ahead forecast of the value of conditional forecast error variance is simply

$$\hat{h}_t = \mathbf{b}_0 + \mathbf{b}_1 u_{t-1}^2 + \mathbf{g}_1 h_{t-1} \quad (26)$$

where  $u_{t-1}$  is the prediction error for the previous time step, i.e.

$$u_{t-1} = y_{t-1} - \mathbf{a}_0 - \mathbf{a}_1 y_{t-2} - \mathbf{a}_2 y_{t-3} - \dots - \mathbf{a}_p y_{t-p} \quad (27)$$

## 6. Portfolio Management Model

In previous sections we developed methods of estimating future conditional risks and returns for 5 LIFFE futures contracts. This section describes a portfolio management model that manages a set of varying positions on each of these markets so as to attempt to maximise the return for a pre-defined level of risk. We begin with a brief description of the mean-variance model used to manage the portfolio.

### 6.1 The Mean Variance Model

Assume that we hold a portfolio of  $N$  assets. Let the size of our holding in each asset be determined by a set of weightings,  $\mathbf{w}_i$ . In the context of a portfolio of futures positions these weightings represent the number (and sign) of contracts held in each futures market. It is convenient to scale the weightings such that a 1% positive change in the price of the futures market would lead to a 1 % return in the nominal portfolio value. This means that the value of  $\mathbf{w}_i$  depends on the number of contracts held, the monetary value of the contract, and the nominal total value of assets held in the portfolio.

The expected return of the portfolio is then given by

$$E(r) = \sum_{i=1}^N \mathbf{w}_i \tilde{r}_i \quad (28)$$

where  $\tilde{r}_i$  is the expected return of asset  $i$ . These return estimates are provided by the WEAPON committee networks described in Sections 3 and 4. The expected risk (future conditional variance) of the portfolio is given by Equation (29).

$$E(V) = \sum_{i=1}^N \sum_{j=1}^N \mathbf{w}_i \mathbf{w}_j \tilde{\mathbf{S}}_{i,j} \quad (29)$$

where  $\tilde{\mathbf{S}}_{i,j}$  is the covariance between the returns of assets  $i$  and  $j$  as defined in Equation (30)

$$\tilde{\mathbf{S}}_{i,j} = \tilde{\mathbf{S}}_i \tilde{\mathbf{S}}_j \tilde{\mathbf{r}}_{i,j} \quad (30)$$

In order estimate the portfolio risks we need estimates of the univariate risks of each asset,  $\tilde{\mathbf{S}}_i$  along with the inter-asset correlations,  $\tilde{\mathbf{r}}_{i,j}$ . We have shown in Section 5 how the individual market risks may be estimated using a GARCH model. We therefore only require estimates for the inter-asset correlations,  $\tilde{\mathbf{r}}_{i,j}$ . We estimate these using a simple Exponentially Weighted Moving Average Model (EWMA) given in Equation (31).

$$\tilde{\mathbf{r}}_{i,j}(t) = \sum_{t=1}^T \frac{(r_i(t-t) - \mathbf{m}_i)(r_j(t-t) - \mathbf{m}_j) e^{-\lambda t}}{e^{-\lambda t}} \quad (31)$$

where  $r_i(t)$  is the historical return of asset  $i$  at time  $t$  and  $\mathbf{m}_i$  is the mean return of asset  $i$  between times  $t - t$  and  $t$ . The parameter  $\lambda$  is set to be 0.99 and the time period  $T$  is set to be 5 days. Historical returns are calculated at 15-minute intervals. Because we are considering a portfolio of futures positions there are

none of the usual constraints placed on the weights of the portfolio (such as positivity of individual asset weightings or the asset weightings summing to unity).

## **6.2 Managing the Portfolio**

The equations for portfolio risk and return introduced in the previous section allow us to monitor the expected risk and return of a given portfolio. To manage the portfolio, however, we need to define the desired properties of the portfolio risk and return over time. These constraints are:

- The portfolio will at all times be balanced such that the expected return of the portfolio is maximised, **subject to the constraint** that the portfolio risk should not exceed the implicit estimated risk in adopting a buy and hold position of the FTSE 100 index spot market.
- The risk of the spot FTSE 100 index is calculated using an Exponentially Weighted Moving Average measure.

The portfolio is *re-balanced* at 240-minute intervals, that is every 240 minutes the optimal set of asset weightings,  $\mathbf{W}_i$ , is determined and trades performed such that these asset weightings are realised. The calculation of the optimal portfolio weightings is carried out using a quadratic programming technique. It is assumed that the total value of the managed portfolio is sufficiently large such that the discrete nature of contracts in the futures market can be ignored.

## 7. Portfolio Management Results

The portfolio of futures positions is simulated over a 6-month test period. During this period, every 240 minutes during each trading day, the portfolio weightings are re-balanced in the following manner,

- Estimates of future hourly returns for the 5 futures contracts were made using the committees of trained WEAPON networks described in Sections 3 and 4.
- Estimates of the future four-hourly variances (risks) of the 5 futures contracts were made using the GARCH model described in Section 5.
- Estimates of the inter-market correlations are made using an Exponentially Weighted Moving Average model.
- The risk of the spot FTSE 100 index is estimated using an Exponentially Weighted Moving Average model.
- The asset weightings that optimise the return of the portfolio subject to the risk being less than or equal to the risk of the FTSE 100 index are determined.
- Contracts are traded (if necessary) in the 5 futures markets to achieve these optimal weightings.

During simulation, the total asset value of the portfolio is monitored at all times net of any transaction costs. The trading conditions assumed during the simulation are as follows:

- We do not know the exact price we will obtain in any market. That is, for instance, if we wish to re-balance a position in one of the contracts at 9:00pm then the price at which the re-balancing takes place is the first relevant ask or bid tick **after** 9:00pm. This strategy means we do not need to take into account any bid-ask spread.
- We assume a round trip transaction cost of 4 ticks per trade for all markets.

The results of managing the portfolio are shown in Figure 9. The overall profitability, estimated annualised volatility and estimated annualised Sharpe Ratio are presented in Table 9.

*Table 9: Table of results comparing the FTSE 100 Index with a number of different managed portfolios.*

Series	Return (6 months)	Sharpe Ratio (annualised)	Volatility (annualised)
FTSE 100 Index	9.12%	1.19	10.04%
Managed (5%)	5.32%	1.47	07.21%
Managed (10%)	30.43%	3.43	15.84%
Managed (20%)	34.46%	1.99	32.34%
Managed (FTSE-100 Index)	18.66%	3.08	11.33%

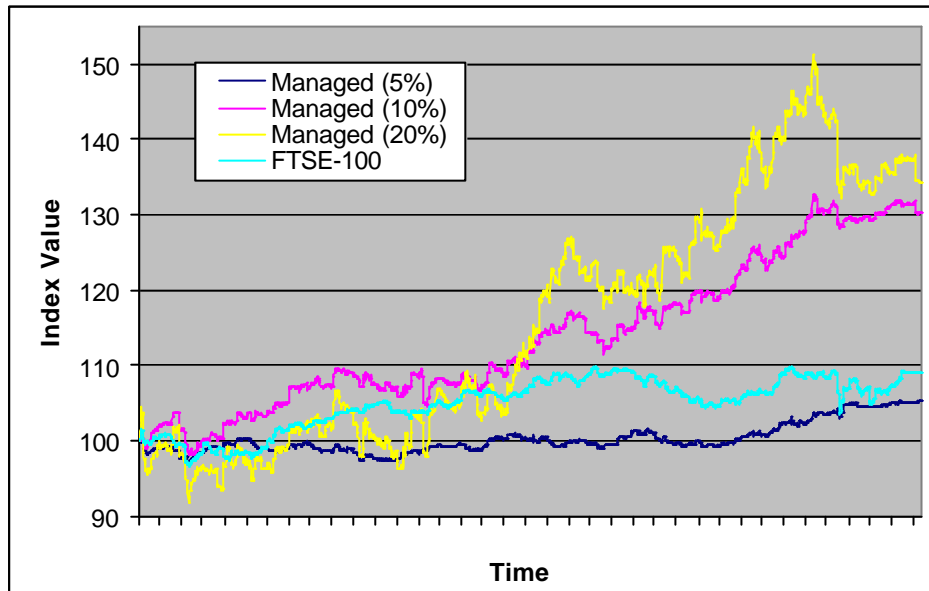
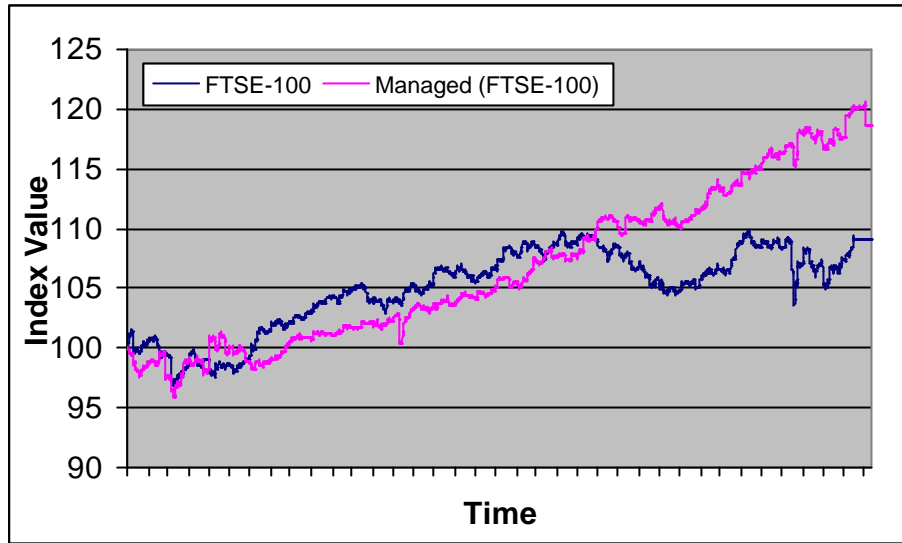


Figure 9: The total asset value of the managed portfolio over the previously unseen test period. As a comparison the FTSE 100 index value is also shown.

## 7. Conclusions

In this chapter, we presented a set of methodologies to quantitatively manage a high-frequency portfolio of positions on 5 LIFFE futures contracts. In order to do this we have

- Made forecasts of future returns for each market using the Wavelet Encoding A Priori Orthogonal Network.
- Made forecasts of risk for each market using a combined AR(p)-GARCH(1,1) method.
- Made forecasts of inter-market correlation using a simple Exponentially Weighted Moving Average (EWMA) technique.
- Combined these estimates to allocate positions in the futures markets using a risk management methodology that attempted to balance the portfolio of positions such that the risk of the futures positions did not exceed the corresponding risk of a buy and hold strategy in the FTSE-100 index.

The resulting portfolio management model appears to function correctly in terms of its realised risk (volatility), with the returns modestly in excess of buy and hold strategies on any of the underlying markets.

## 9. Bibliography

- Berndt, E K, B H Hall, R E Hall and J A Hausman, *Estimation and Inference in Nonlinear Structural Models*, Annals of Economic and Social Measurement, Vol. 3, 653-665, 1974.
- Bollerslev, T, *Generalized Autoregressive Conditional Heteroskedasticity*, Journal of Econometrics, Vol. 31, 307-327, 1986.
- Casasent, D P and J.S Smokelin, *Neural Net Design of Macro Gabor Wavelet Filters for Distortion-Invariant Object Detection In Clutter*, Optical Engineering, Vol. 33 (7), 2264-2270, July 1994.
- Chui, C, *An Introduction To Wavelets*, Academic Press, New York, 1992.
- Daubechies, I, *The Wavelet Transform, Time Frequency Localisation and Signal Analysis*, IEEE Transaction on Information Theory, 1992.
- Kalman, R E, *A New Approach to Linear filtering and Prediction Problems*, Trans ASME, J. Basic Engineering, Vol. 82, 34-45, 1960
- MacKay, D J, *A Practical Bayesian Framework For Backprop Nets*, Neural Computation, Vol. 4 (3) 448-472, 1992a.
- MacKay, D J, *Bayesian Interpolation*, Neural Computation, Vol. 4 (3), 415-447, 1992b.
- Markowitz, H M, *Portfolio Selection: Efficient diversification of Investments*, John Wiley, New York, 1959.
- Meyer, Y, *Wavelets and Operators*, Cambridge University Press, 1995.
- Rummelhart, D E, G E Hinton and R J Williams, *Learning Internal Representations by Error Propagation*, in *Parallel Distributed Processing*, Vol 1, Ch 8, MIT Press, 1986.
- Shannon, C E, *A Mathematical Theory of Communication*, The Bell System Technical Journal, Vol. 27 (3), 379-423 and 623 -656, 1948.
- Szu, H and B. Telfer, *Neural Network Adaptive Filters For Signal Representation*, Optical Engineering 31, 1907- 1916, 1992.
- Telfer, B A, H Szu and G J Dobeck *Time-Frequency, Multiple Aspect Acoustic Classification*, World Congress on Neural Networks, Vol. 2, II-134 – II-139, July 1995.
- Toulson, D L and S P Toulson, *Use of Neural Network Ensembles for Portfolio Selection and Risk Management*, Proc. Third Intl. Chemical Bank/Imperial College Conference: Forecasting Financial Markets, London, 1996a.
- Toulson, D L and S P Toulson, *Use of Neural Network Mixture Models for Forecasting and Application to Portfolio Management*, Proc. Sixth International Symposium on Forecasting, Istanbul, 1996b.
- Toulson, D L and S P Toulson, *Intra-Day Trading of the FTSE-100 Futures contract using Neural Networks with Wavelet Encodings*, Proc. Fourth Intl. Banque Nationale de Paris/Imperial College Conference: Forecasting Financial Markets, London, 1997.
- Williams, P M, *Bayesian Regularisation and Pruning Using A Laplace Prior*, Neural Computation 5 (3) 1993.