4

Model Selection: Selecting the Architecture of the Network

In this chapter we describe the model selection procedure. One of the most crucial steps is to identify the correct topology of the network. A desired wavelet network architecture should contain as few hidden units as necessary; at the same time it should explain as much variability of the training data as possible. A network with fewer hidden units than needed would not be able to learn the underlying function; selecting more hidden units than needed would result in an overfitted model. Therefore, it is essential to derive an algorithm to select the appropriate wavelet network model for a given problem.

The simplest way to select the optimal number of hidden units—in other words, the architecture of the wavelet network—is by trial and error, a method called *exhaustive search*. To do so, the training patterns must be split into a training sample and a validation sample. This method suggests that the optimum number of wavelons is given by the structure of the wavelet network that gives the minimum error on the validation set. This method is very simple but also very time consuming, and the information in the validation sample is not utilized for better training of the wavelet network.

In *early stopping* a fixed and large number of hidden units is used in construction of the network. The weights are allowed to change during the training phase. These free parameters are growing during the training phase. In early stopping the training is stopped to avoid overfitting of the wavelet network to the data.

Wavelet Neural Networks: With Applications in Financial Engineering, Chaos, and Classification, First Edition. Antonios K. Alexandridis and Achilleas D. Zapranis.

Another approach to avoid overfitting is *regularization*. In regularization, a parameter larger than zero is specified, and a regularized performance index is minimized instead of the original mean squared error. The idea is to keep the overall growth of weights to a minimum such that weights are pulled toward zero. In this process, only the important weights are allowed to grow; the others are forced to decay.

Both early stopping and regularization use all weights in training. As a result, the structural complexity of the network is not reduced. Alternatively, in the *pruning method* the complexity of the network is reduced so that only the essential weights and neurons remain in the model. However, the various criteria used in pruning the weights are not employed in a statistical way.

Finally, another method used to find the optimal architecture of a wavelet network is *minimum prediction risk* (MPR). The idea behind MPR is to estimate the out-of-sample performance of incrementally growing networks. The number of hidden units that minimizes the prediction risk is the appropriate number of hidden units that should be used for construction of a wavelet network. In other words, the prediction risk is a form of measurement of the generalization ability of the wavelet network.

Early methods used *information criteria* to estimate the minimum prediction risk. The most popular information criteria are Akaike's final prediction error (FPE), generalized cross-validation (GCV), and the Bayesian information criterion (BIC). Information criteria measure the error between the training data and the network output, but a penalty term is added for large networks. Information criteria were derived for linear models, and some of the assumptions that they employ are not necessarily true for nonlinear nonparametric wavelet networks.

Alternatively, *resampling schemes* such as *bootstrap* or *cross-validation* can be used. In resampling schemes, different versions of a single statistic that would ordinarily be calculated from one sample can be estimated. Bootstrap and cross-validation do not require prior identification of the data-generating process. The main disadvantage of the application of sampling techniques is the fact that they are computationally expensive.

THE USUAL PRACTICE

The usual approaches proposed in the literature are early stopping, regularization, and pruning. In early stopping a fixed and large number of hidden units is used in construction of the network. In regularization methods the weights of the network are trained to minimize the loss function plus a penalty term. The idea is to keep the overall growth of weights to a minimum such that weights are pulled toward zero. Therefore, only a subset of weights that become most sensitive to the output is used effectively. In this process, only the important weights are allowed to grow; others are forced to decay.

Early Stopping

In early stopping a fixed and large number of hidden units are used in construction of the network. Hence, a large number of weights must be initialized and optimized

during the training phase. The number of weights roughly defines the degrees of freedom of the network. If the training phase continues past the appropriate number of iterations and the weights grow very large in the training phase, the network will begin to learn the noise part of the data and will become overfitted. As a result, the generalization ability of the network will be lost. Hence, it is not appropriate to use the wavelet network in predicting new and unseen data. On the other hand, if the training is stopped at an appropriate point, it is possible to avoid overfitting the network.

A common practice to overcome the problems outlined above is the use of a validation sample. At each iteration, the network is trained using the training sample. Then the cost function between the training data and the network output is estimated and it is used to adjust the weights. Then the generalization ability of the network is measured using the validation sample. More precisely, the network is used to forecast the target values of the validation sample using the unseen input data of the validation sample. The error between the network output and the target data of the validation sample is calculated. Usually, the validation sample has 10 to 30% the size of the training sample.

At the beginning of the training phase the errors of both the training and the validation sample will start to decrease as the network weights are adjusted to the training data. After a particular iteration the network will begin to learn the noise part of the data. As a result, the error of the validation sample will begin to increase. This is an indication that the network is starting to lose its generalization ability and that the training phase must be stopped.

In the early stopping method a more complex model than needed is used. Hence, a large number of weights must be trained. As a result, large training times are expected. Moreover, the network incorporates a large number of connections, most of them with small weights. In addition, a validation sample should be used. However, in real applications, usually only a small amount of data is available, and splitting the data is not useful. Furthermore, growing validation errors indicate the reduction of a network's complexity (Anders and Korn, 1999). Finally, the solution $\hat{\mathbf{w}}_n$ of the network is highly dependent on dividing the data and the initial conditions (Dimopoulos et al., 1995).

Regularization

Another approach to avoiding overfitting is regularization. In regularization methods the weights of the network are trained to minimize the loss function plus a penalty term. Regularization is attempting to keep the overall growth of weights to a minimum by allowing only the important weights to grow. The remaining weights are pulled toward zero (Samarasinghe, 2006). This method is often called *weight decay* (Samarasinghe, 2006).

The regularization method tries to minimizes the sum:

$$W = L_n + \delta \sum_{j=1}^{J} w_j^2 \tag{4.1}$$

where the second term is the penalty term, w_j is a weight, J is the total number of weights in the network architecture, and δ is a regularization parameter. The

penalty term is not restricted to the choice above. However, the penalty terms are usually chosen arbitrarily without theoretical justification (Anders and Korn, 1999). Moreover, a bad regularization parameter, δ, can severely restrict the growth of weights, and as result, the network will be underfitted (Samarasinghe, 2006).

Pruning

Similar to other methods, the aim of pruning is to identify those parameters that contribute the least to network performance. Several approaches have been proposed to prune networks. However, the significance of each weight is usually not measured in a statistical way (Anders and Korn, 1999). Reed (1993) has provided an extensive survey of pruning methods. One of the disadvantages of pruning is that it often does not take correlated weights into account. Two weights that cancel each other out do not have any effect at the output of the network; however, each weight may have a large effect (Reed, 1993). Also, the time when the pruning should stop is usually arbitrary (Reed, 1993). Reed (1993) separated the pruning algorithms into two major groups: sensitivity calculation methods and penalty term methods. Here we present selected methods from each group.

Brute-Force Pruning In brute-force pruning, the simplest method, each weight is set to zero and the effect on the error is estimated. If the change in the error increases "too much," the weight is restored to its value. One way to do so is, first, to estimate the change in error for every weight and for every pattern and then delete the weight with the least effect. This procedure is repeated up to a fixed threshold. However, it is not very straightforward to define whether or not the increase in error is large.

Sensitivity Calculation: Saliency Reed (1993) has estimated the saliency of a weight using the second derivative of the error with respect to the weight:

$$\delta L_n = \sum_{i=1}^{p} g_i \, \delta w_i + \frac{1}{2} \sum_{i=1}^{p} a_{ii} \, \delta w_i^2 + \frac{1}{2} \sum_{i=1, i \neq j}^{p} a_{ij} \, \delta w_i \, \delta w_j + O\left(\|\delta W\|^2\right) \quad (4.2)$$

where the δw_i's are the components of δW, g_i are the components of the gradient of L_n with respect to W, and the a_{ij} are the elements of the Hessian matrix H:

$$g_i = \frac{\partial L_n}{\partial w_i} \quad (4.3)$$

$$a_{ij} = \frac{\partial^2 L_n}{\partial w_i \, \partial w_j} \quad (4.4)$$

Since pruning is done on a well-defined local minimum and for small perturbations, (4.2) can be simplified since the Hessian matrix is very large:

$$\delta L_n \approx \frac{1}{2} \sum_{i=1}^{p} a_{ii} \delta w_i^2 \tag{4.5}$$

Then the saliency of weight w_k is given by

$$S(w_i) = \frac{a_{ii} w_i^2}{2} \tag{4.6}$$

It can be considered that a_{ii} is an indication of the acceleration of the error with respect to a small perturbation to a weight w_i. Hence, through equation (4.6), an indication of the total effect of w_i on the error is obtained. The larger the saliency, the larger the influence of w_i on error. The other entries of the Hessian matrix are assumed to be zero; therefore, the second derivative with respect to weights other than itself is ignored (Samarasinghe, 2006). This implies that the weights of the network are independent. However, this may not be true for a network that has more than the optimum number of weights.

To apply this method, the following procedure is followed. First, a wavelet network should be trained in the normal way and the saliency computed for each weight. Then, weights with small values of saliency are removed. This may lead to pruning of weights as well as neurons. After a removal of a weight, the wavelet network is trained further. The simplified trained wavelet network should perform as well as the optimum network with a larger number of weights (Samarasinghe, 2006).

Irrelevant Connection Elimination Scheme An extension of sensitivity calculation is the irrelevant connection elimination scheme proposed by Zapranis and Haramis (2001). Once the parameters of the wavelet neural model $g_\lambda(\mathbf{x}; \mathbf{w})$ are estimated, we have to deal with the presence of flat minima (potentially, many combinations of the network parameters corresponding to the same level of the empirical loss), especially if the statistical properties of the model are of importance, as is the case in complex financial applications. To identify a locally unique solution, we have to remove all the irrelevant parameters: that is, the parameters that do not affect the level of the empirical loss.

For this purpose we use the irrelevant connection elimination (ICE) scheme, which is much less computationally demanding than other alternatives. The irrelevant connection elimination scheme, although it uses the full Hessian of L_n, does not require inverting the Hessian matrix, a common requirement of other algorithms. ICE is based on Taylor's approximation of the empirical loss:

$$\delta L_n = \sum_{i=1}^{p} g_i \, \delta w_i + \frac{1}{2} \sum_{i=1}^{p} a_{ii} \, \delta w_i^2 + \frac{1}{2} \sum_{i=1, i \neq j}^{p} a_{ij} \, \delta w_i \, \delta w_j + O\left((\delta w)^3\right) \tag{4.7}$$

where

$$a_{ij} = \frac{\partial^2 L_n}{\partial w_i \, \partial w_j} \tag{4.8}$$

From (4.7), ICE derives the "saliencies" $S(w_i)$ (i.e., the contribution of w_i to δL_n) when a small perturbation δw_k is added to all connections:

$$S(w_i) = g_i \, \delta w_i + \frac{1}{2} \sum_{j=1}^{p} a_{ij} \, \delta w_i \, \delta w_j \tag{4.9}$$

where $\delta L_n = \sum_{i=1}^{p} S(w_i)$.

At a well-defined local minimum, (4.9) can be simplified by setting $g_i = 0$, although this is not a requirement. The method can be summarized in the following steps:

Step 1: Train to convergence.

Step 2: Compute the saliencies $S(w_i)$.

Step 3: Deactivate the connection with the least associated saliency, unless it was reactivated in step 5. When a prespecified maximum number of steps has been reached, the algorithm STOPS.

Step 4: Train further for a small number of epochs, until the training error has stabilized.

Step 5: If the training error has increased, reactivate the connection; otherwise, remove it. Then go to step 3.

Because of possible dependencies in the connections, it is not advisable to remove more than one connection at a time (the removal of one connection can affect the standard errors and saliencies of others). This does not pose any computational problems to ICE, since computing the Hessian is of the same order of complexity as computing the derivatives $\partial L_n / \partial w_i$ during training.

MINIMUM PREDICTION RISK

The aim of model selection is to find the least complex model that can learn the underlying target function. Previous methods do not use the optimal architecture of a wavelet network. A very large wavelet network is used and then various methods are developed to avoid overfitting. Smaller networks usually are faster to train and need less computational power to build (Reed, 1993).

Alternatively, the minimum prediction risk principle can be applied (Efron and Tibshirani, 1993; Zapranis and Refenes, 1999). The idea behind minimum prediction risk is to estimate the out-of-sample performance of incrementally growing networks. Assuming that the explanatory variables $\mathbf{x}$ were selected correctly and remain fixed,

the model selection procedure is the following: The procedure starts with a fully connected network with no hidden units (in our proposed structure of wavelet networks, this is a linear model). The wavelet network is trained, and then the prediction risk is estimated. Then, iteratively, a new hidden unit is added to the network. The new wavelet networks are trained and the new prediction risk is estimated at each step. The number of hidden units that minimizes the prediction risk is the appropriate number of hidden units that should be used for construction of the wavelet network.

The prediction risk measures the generalization ability of the network. More precisely, the prediction risk of a network $g_\lambda(\mathbf{x}; \hat{\mathbf{w}}_n)$ is the expected performance of the network on new data that were not introduced during the training phase and is given by

$$P_\lambda = E\left[\frac{1}{n}\sum_{p=1}^{n}\left(y_p^* - \hat{y}_p^*\right)^2\right] \tag{4.10}$$

where $(\mathbf{x}_p^*, y_p^*)$ are the new observations that have not been used in the construction of the network $g_\lambda(\mathbf{x}; \hat{\mathbf{w}}_n)$, and $\hat{y}_p^*$ is the network output using the new observations, $g_\lambda(\mathbf{x}^*; \mathbf{w})$.

Finding a statistical measure that estimates the prediction risk is not a straightforward procedure, however. Since there is a linear relationship between the wavelons and the output of the wavelet network, Zhang (1993, 1994, 1997) and Zhang and Benveniste (1992) propose the use of information criteria widely applied previously in linear models. A different approach was presented by Zapranis and Refenes (1999). An analytical form of the prediction risk (4.10) was presented for sigmoid neural networks. However, the assumptions made by Zapranis and Refenes (1999) are not necessarily true in the framework of wavelet networks, and analytical forms are not available for estimating the prediction risk for wavelet networks. Alternatively, the use of sampling methods such as bootstrap and cross-validation can be employed since they do not depend on any assumptions regarding the model (Efron and Tibshirani, 1993). The only assumption made by sampling methods is that the data are a sequence of independent, identically distributed variables.

ESTIMATING THE PREDICTION RISK USING INFORMATION CRITERIA

In wavelet networks, the wavelons are connected linearly to the output of the wavelet network, Zhang (1993, 1994, 1997) and Zhang and Benveniste (1992) propose the use of information criteria to find the optimal architecture of a wavelet network. Information criteria are used widely and successfully in estimation of the number of parameters in linear models. More precisely, Zhang (1994) suggested that Akaike's final prediction error (FPE) can be used in various applications. More recently, Zhang (1997) suggested that generalized cross-validation (GCV) is an accurate tool for selecting the number of wavelets that constitutes the wavelet network's topology.

To estimate the prediction risk and to find the network with the best predicting ability, a series of information criteria was developed. As the model complexity increases and more parameters are added to the wavelet network, it is expected that the fit will improve, but not necessarily the forecasting ability of the wavelet network. The idea behind these criteria is to measure the error between the training data and the network output, but at the same time to penalize the complexity of the network.

To select the best architecture of the wavelet network, the following procedure is pursued. In the first step, a wavelet network with no hidden units is constructed. The wavelet network is trained and then the corresponding information criterion and the prediction risk are estimated. In the next step, 1 hidden unit is added to the architecture of the wavelet network and the procedure is repeated until the network contains a predefined maximum number of hidden units. The number of hidden units that produces the smallest prediction risk is the number of the appropriate wavelets for the construction of the wavelet network.

Akaike's Information Criterion Several criteria exist for model selection. Early studies make use of the generalized prediction error (GPE) proposed by Moody (1992) and the network information criterion (NIC) proposed by Murata et al. (1994). However, the results by Anders and Korn (1999) indicate that NIC significantly underperforms other criteria. Alternatively, Akaike's information criterion (AIC) (Akaike, 1973, 1974), which was proved to work well in various cases, was used. AIC is given by

$$J_{\mathrm{AIC}} = 2k + n \ln \left[\frac{1}{n} \sum_{p=1}^{n} \left(y_p - \hat{y}_p \right)^2 \right] \tag{4.11}$$

where k is the number of parameters of the network and n is the number of training patterns in the training sample. The target value is given by y_p, and $\hat{y}_p$is the approximation of the target value by the network.

Final Prediction Error Zhang (1994) suggested that Akaike's final prediction error (FPE) can be used in various applications. The FPE is given by

$$J_{\mathrm{FPE}} = \frac{1 + k/n}{2n - 2k} \sum_{p=1}^{n} \left(y_p - \hat{y}_p \right)^2 \tag{4.12}$$

Generalized Cross-Validation More recently, Zhang (1997) suggested that generalized cross-validation (GCV) should be used to select the number of wavelets that constitutes the wavelet network topology. GCV is given by

$$J_{\mathrm{GCV}} = \frac{1}{n} \sum_{p=1}^{n} \left(y_p - \hat{y}_p \right)^2 + \frac{2\mathrm{HU} \cdot \hat{\sigma}^2}{n} \tag{4.13}$$

In practice, the noise variance σ^2 is not known. In that case it has to be estimated. An estimate is given by the MSE between the network output and the target data (Zhang, 1997).

Bayesian Information Criterion Similar to GCV is the Bayesian information criterion (BIC), given by

$$J_{\text{BIC}} = \frac{1}{n}\sum_{p=1}^{n}\left(y_p - \hat{y}_p\right)^2 + \frac{k\hat{\sigma}^2 \ln(n)}{n} \tag{4.14}$$

To estimate the AIC, FPE, GCV, and BIC, the number of the hidden units is needed. Because we do not have a priori knowledge of the correct number of hidden units or parameters of a wavelet network for estimation of GCV and the BIC, we estimate the criteria above iteratively. The computational cost of these algorithms can be expressed as a function of wavelet networks to be trained. For example, to estimate the prediction risk using the FPE or GCV from 1 to five hidden units, five wavelet networks must be initialized and fully trained. The model selection algorithm using IC is illustrated in Figure 4.1. It is expected that the prediction risk will decrease (almost) monotonically until it reaches a minimum and then it will increase (almost) monotonically. The number of wavelons needed for the construction of the networks is the number of hidden units that minimize the prediction risk.

The criteria described above for estimation of the prediction risk are derived from linear models. Usually, these methods are based on assumptions that are not necessarily true in the framework of nonlinear nonparametric estimation. The hypothesis behind these information criteria is the asymptotic normality of the maximum likelihood estimators; hence, the information criteria are not theoretically justified for overparameterized networks (Anders and Korn, 1999).

Moreover, in fitting problems more complex than least squares, the number of parameters k is not known (Efron and Tibshirani, 1993) and it is unclear how to compute the degrees of freedom (Curry and Morgan, 2006) or the effective number of parameters described by Moody (1992).

Alternatively, the use of sampling methods such as bootstrapping and cross-validation was suggested (Efron and Tibshirani, 1993). The only assumption made by sampling methods is that the data are a sequence of independent, identically distributed variables. Bootstrapping and cross-validation do not require knowledge of the number of parameters k. Another advantage of bootstrapping and cross-validation is their robustness. In contrast to sampling methods, both GCV and BIC require a roughly correct model to obtain an estimate of the noise variance.

ESTIMATING THE PREDICTION RISK USING SAMPLING TECHNIQUES

Instead of using information criteria to obtain an estimate of the prediction risk, resampling schemes can be used. Two resampling schemes described in this section

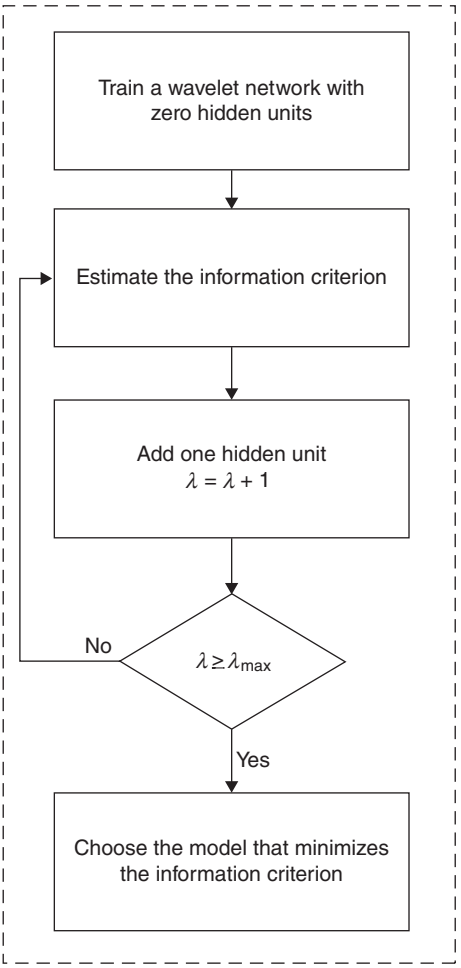

Figure 4.1 *Model selection algorithm using information criteria.*

are bootstrapping and cross-validation, which do not require prior identification of the data-generating process. Furthermore, they can approximate the distributional properties of a small sample $\hat{\mathbf{w}}_n$ accurately when the data are a sequence of independent, identically distributed variables. The estimation of these two approaches is asymptotically equal. The main disadvantage of the application of sampling techniques is the fact that they are computationally expensive.

Bootstrapping allows one to gather many alternative versions of a single statistic that would ordinarily be calculated from one sample. Where a set of observations can be assumed to originate from an independent, identically distributed population, bootstrapping can be implemented by constructing a number of new samples of the data set observed (and equal in size to the data set), each of which is obtained by random sampling with replacement from the original data set.

Similarly, in cross-validation, new samples are constructed from the original data set. However, there are two differences. First, random sampling is performed without replacement, and second, each sample is split into two parts, the training set and the validation set.

Bootstrapping

In this section the bootstrapping method is described. In summary, the simple bootstrapping approach generates new random instances from the original data. Then a new model is estimated for each sample, and finally, each fitted model is applied in order to generate an estimate of the prediction risk.

There are two methods of applying bootstrapping: bootstrapping pairs and bootstrapping residuals. In this book the bootstrapping pairs method is followed. The bootstrapping pairs method is less sensitive to assumptions than are bootstrapping residuals (Efron and Tibshirani, 1993). The only assumption behind bootstrapping pairs is that the original pairs were sampled randomly from some distribution. On the other hand, in bootstrapping residuals, the distribution of the residuals must be assumed beforehand. This is a very strong assumption which may lead to false conclusions.

Typically, a large number B of new samples $D_n^{*(b)} = \{x_p^{*(b)}, y_p^{*(b)}\}_{p=1}^n$ is created from the original sample, $D_n = \{x_p, y_p\}_{p=1}^n$, with size n, where $b = 1, \ldots, B$. Typically, the number of samples is $20 < B$. Each pattern $\{x_p, y_p\}$ has $1/n$ probability to be selected with replacement from the original sample.

Next, we illustrate how bootstrapping works by presenting a very simple example. Let us assume that the variable $x = \{1, 2, \ldots, 10\}$ and that the dependent variable y is given by the relationship $y_p = x_p^2$. Hence, $y = \{1, 4, \ldots, 100\}$. The original data set consists of 10 patterns $\{x_p, y_p\}$. By applying bootstrapping we create 10 new samples of size 10 each. An example of 10 bootstrapped samples is presented in Table 4.1.

For each new sample $D_n^{*(b)}$, the wavelet network is trained to find the weight vector $\hat{\mathbf{w}}_n^{*(b)}$, and the loss function $L_n(\hat{\mathbf{w}}_n^{*(b)})$ is estimated. Then an estimation of the prediction risk is given by

$$\hat{P}_\lambda = \frac{1}{nB} \sum_{b=1}^{B} \sum_{p=1}^{n} \left\{ y_p - g_\lambda \left(\mathbf{x}_p; \hat{\mathbf{w}}_n^{*(b)} \right) \right\}^2 \tag{4.15}$$

The previous formula for the prediction risk is the average performance of the wavelet networks, which were trained on bootstrapped samples, on the original data set. In other words, it is the average of all loss functions estimated of the bootstrapped wavelet networks to the original sample.

In the estimation of prediction risk above, the wavelet network from each bootstrapped sample was used to predict the target values of the original sample. The estimation above of the prediction risk is very simple in use; however, it is known that it is not very accurate (Efron and Tibshirani, 1993).

TABLE 4.1 Simple Bootstrapping Example

	Pattern									
Sample	1	2	3	4	5	6	7	8	9	10
$\mathbf{x}^{*(1)}$	5	7	10	10	7	2	1	7	6	10
$\mathbf{y}^{*(1)}$	25	49	100	100	49	4	1	49	36	100
$\mathbf{x}^{*(2)}$	8	7	6	3	10	6	1	7	6	1
$\mathbf{y}^{*(2)}$	64	49	36	9	100	36	1	49	36	1
$\mathbf{x}^{*(3)}$	9	4	3	2	4	3	7	1	3	3
$\mathbf{y}^{*(3)}$	81	16	9	4	16	9	49	1	9	9
$\mathbf{x}^{*(4)}$	9	5	8	7	8	2	10	9	1	5
$\mathbf{y}^{*(4)}$	81	25	64	49	64	4	100	81	1	25
$\mathbf{x}^{*(5)}$	4	7	3	6	7	6	5	1	6	5
$\mathbf{y}^{*(5)}$	16	49	9	36	49	36	25	1	36	25
$\mathbf{x}^{*(6)}$	2	5	5	6	9	8	9	1	3	5
$\mathbf{y}^{*(6)}$	4	25	25	36	81	64	81	1	9	25
$\mathbf{x}^{*(7)}$	10	8	5	4	1	8	6	2	5	2
$\mathbf{y}^{*(7)}$	100	64	25	16	1	64	36	4	25	4
$\mathbf{x}^{*(8)}$	8	4	10	1	9	7	6	7	8	1
$\mathbf{y}^{*(8)}$	64	16	100	1	81	49	36	49	64	1
$\mathbf{x}^{*(9)}$	9	1	3	2	10	1	6	7	7	9
$\mathbf{y}^{*(9)}$	81	1	9	4	100	1	36	49	49	81
$\mathbf{x}^{*(10)}$	1	9	6	7	3	6	8	10	5	1
$\mathbf{y}^{*(10)}$	1	81	36	49	9	36	64	100	25	1

A method proposed by Efron and Tibshirani (1993) to improve the estimated prediction risk given by (4.15) is the following. First, the apparent error is estimated:

$$\text{Aperr} = \frac{1}{nB} \sum_{b=1}^{B} \sum_{p=1}^{n} \left[y_p^{*(b)} - g_\lambda \left(\mathbf{x}_p^{*(b)}; \hat{\mathbf{w}}_n^{*(b)} \right) \right]^2 \tag{4.16}$$

Since each wavelet network is estimated using the bootstrapped samples $D_n^{*(b)}$ and is validated on the original sample D_n, the prediction risk $\hat{P}_\lambda$ given by (4.15) can be considered to be an out-of-sample validation. On the other hand, the apparent error can be considered to be an in-sample validation. The difference between these two measures, called the *optimism*, can be estimated by

$$\text{Opt} = \hat{P}_\lambda - \text{Aperr} \tag{4.17}$$

Finally, the optimism is added to the training error of the original training sample D_n:

$$\tilde{P}_\lambda = L_n(\hat{\mathbf{w}}_n) + \text{Opt} \tag{4.18}$$

The number of new samples B is usually over 30 (Aczel, 1993; Efron and Tibshirani, 1993). In our implementation 50 new samples were created. It is clear that as the number of new samples B increases, the bootstrapping method becomes more accurate but also more computationally expensive. The model selection algorithm using the bootstrapped method described above is illustrated in Figure 4.2.

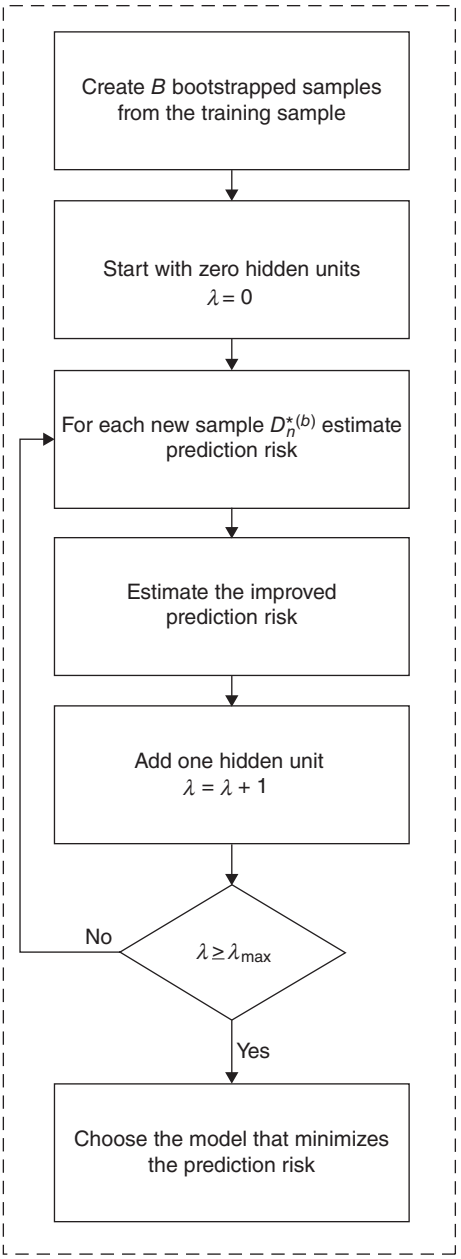

Figure 4.2 *Model selection algorithm using the bootstrap method.*

As in the case of information criteria, the prediction risk is expected to decrease monotonically until it reaches a minimum and then to increase monotonically. The number of hidden units that minimizes the prediction risk is selected for construction of the network.

Cross-Validation

Cross-validation is a standard tool for estimating the prediction error. The idea of simple validation is to split the training sample $D_n = \{x_p, y_p\}_{p=1}^{n}$ into two parts: the training sample $D_{\text{train}} = \{x_p, y_p\}_{p=1}^{m}$ and the validation sample $D_{\text{valid}} = \{x_p, y_p\}_{p=1}^{n-m}$ with $m < n$. Hence, we can train the network on the training sample and estimate the prediction risk from the new data of the validation sample. However, additional data are often not available. In simple validation not all the data available are used for network training. Hence, available information is lost and it is not utilized during the training phase of the network.

Cross-validation makes efficient use of the information available (Efron and Tibshirani, 1993). In the leave-one-out cross-validation proposed by Mosteller and Tukey (1968), the validation sample consists of only one training pattern. The procedure is the following. First, we assume that the data consist of n independently distributed random vectors. Starting with zero hidden units at step j, the jth training pair $\{x_j, y_j\}$ is removed from the training sample. Then a wavelet network is trained using the reduced sample D_{train}. The trained wavelet network, $g_\lambda(\mathbf{x}; \hat{\mathbf{w}}_{n-1}^{(j)})$, is validated on the validation sample D_{valid}, which consists of the jth training pair $\{x_j, y_j\}$.

This procedure is repeated n times and the *cross-validation criterion* is given by the equation

$$\text{CV} = \frac{1}{n} \sum_{p=1}^{n} \left[y_p - g_\lambda \left(\mathbf{x}_p; \hat{\mathbf{w}}_n^{(j)} \right) \right]^2 \tag{4.19}$$

and it is used as an estimator for the prediction risk $E[L(\hat{\mathbf{w}}_n)]$ for the wavelet network $g_\lambda(\mathbf{x}; \hat{\mathbf{w}}_n)$. Then 1 hidden unit is added to the network and the procedure is repeated up to a predefined maximum number of hidden units. The number of hidden units that generates the smallest prediction risk is the number of the appropriate wavelets for construction of the wavelet network. Again, it is expected that the prediction risk will decrease (almost) monotonically until it reaches a minimum and then will increase (almost) monotonically.

However, the leave-one-out cross-validation is very computationally expensive since HU $\cdot$ n networks must be trained. Hence, for large data sets where the training patterns are several hundreds or thousands, this method is very cumbersome and time consuming.

Alternatively, *v-fold* cross-validation can be used. In this procedure, in the first step, v new subsamples D_m^j of size $m < n$ are created with random sampling without replacement from the original training sample. Next, starting with zero hidden units, the subsamples D_m^j are removed one by one from the original sample D_n, and the

network is trained on the remaining data. The resulting weight parameters are defined by the vector $\hat{\mathbf{w}}_m^{(D_j)}$. Then the trained network is validated at the left-out subsample D_m^j by estimating the mean squared cross-validation error:

$$\mathrm{CV}_{D_j} = \frac{1}{n} \sum_{\{x_p, y_p\} \in D_j} \left[y_p - g_\lambda \left(\mathbf{x}_p; \hat{\mathbf{w}}_m^{(D_j)} \right) \right]^2 \tag{4.20}$$

The prediction risk is the average mean squared cross-validation error of all subsamples and is given by

$$\hat{P}_\lambda \equiv \mathrm{CV}_\lambda = \frac{1}{v} \sum_{j=1}^{v} \mathrm{CV}_{D_j} \tag{4.21}$$

After the estimation of the prediction risk, 1 hidden unit is added to the network and the procedure is repeated up to a predefined maximum number of hidden units. The number of hidden units that produce the smallest prediction risk is the number of appropriate wavelets for construction of the wavelet network. The model selection algorithm using the cross-validation is illustrated in Figure 4.3.

To illustrate how cross-validation works, a very simple example is presented. Let us assume that the variable $x = \{1, 2, \ldots, 10\}$ and that the dependent variable y is given by the relationship $y_p = x_p^2$. Hence, $y = \{1, 4, \ldots, 100\}$. Our original data set consists of 10 patterns $\{x_p, y_p\}$. By applying cross-validation we will create five new samples of size 10 each. Each sample is separated in training and in a test subsample. Moreover, the training sample consists of the 80% of the observations with the validation conducted on the remaining 20%. Note that in contrast to bootstrapping the pairs (x_p, y_p) are selected randomly from the original data set without replacement. An example of five samples is presented in Table 4.2. In the case of the leave-one-out cross-validation, 10 samples would be created and the test sample would contain only one value. It is clear in this simple example that by using *v-fold* cross-validation, the computational burden of the algorithm can be reduced significantly.

Since $v \ll n$, *v-fold* cross-validation is significantly less computationally expensive than is leave-one-out cross-validation. As v, increases, the computational burden increases but also the accuracy of the method increases. When $v = n$ the leave-one-out cross-validation is retrieved. In our implementation, the training data were split into 50 subsamples.

Model Selection Without Training

The preferred information criteria were estimated by Zhang (1997) after the initialization stage of the network was performed. More precisely, in the SSO and RBS the preferred information criteria were evaluated after the selection of each wavelet in the initialization stage. Similarly, when the BE algorithm was used, the preferred information criteria were evaluated after the elimination of each wavelet in the initialization stage.

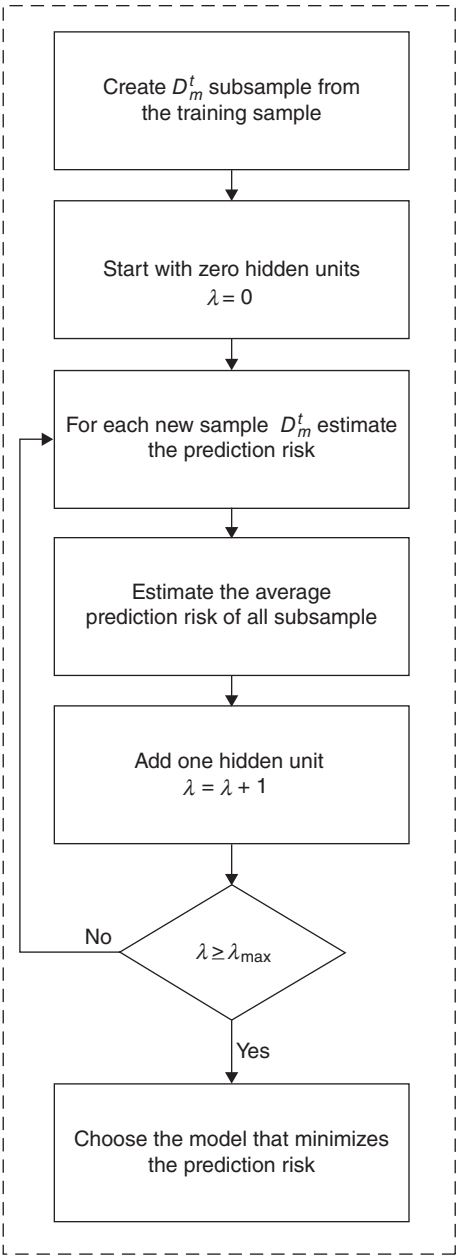

Figure 4.3 *Model selection algorithm using the cross-validation method.*

TABLE 4.2 Simple Cross-Validation Example

	Pattern								Test Training	
Sample	1	2	3	4	5	6	7	8	9	10
$\mathbf{x}^{*(1)}$	1	9	10	5	7	2	4	3	6	8
$\mathbf{y}^{*(1)}$	1	81	100	25	49	4	16	9	36	64
$\mathbf{x}^{*(2)}$	2	9	8	4	10	1	3	6	7	5
$\mathbf{y}^{*(2)}$	4	81	64	16	100	1	9	36	49	25
$\mathbf{x}^{*(3)}$	5	8	9	10	3	4	6	7	2	1
$\mathbf{y}^{*(3)}$	25	64	81	100	9	16	36	49	4	1
$\mathbf{x}^{*(4)}$	9	8	10	6	1	2	7	5	3	4
$\mathbf{y}^{*(4)}$	81	64	100	36	1	4	49	25	9	16
$\mathbf{x}^{*(5)}$	3	1	7	6	2	8	5	4	9	10
$\mathbf{y}^{*(5)}$	9	1	49	36	4	64	25	16	81	100

Since initialization of the wavelet network is very good, as discussed earlier, the initial approximation is expected to be very close to the target function. Hence, a good approximation of the prediction risk is expected to be obtained. The same idea can also be applied when bootstrapping or cross-validation is used.

As presented in earlier chapters, the computational burden and time needed for the initialization phase of a wavelet network are insignificant compared to the training phase. Hence, the procedure above is significantly less computationally expensive. However, the procedure is similar to early stopping techniques. Usually, early stopping techniques suggest a network with more hidden units than necessary, although the network is not fully trained to avoid overfitting (Samarasinghe, 2006). From our experience this method does not work satisfactorily in complex problems.

EVALUATING THE MODEL SELECTION ALGORITHM

To find an algorithm that will work well with wavelet networks and lead to a good estimation of prediction risk, in this section we compare the various criteria as well as the sampling techniques discussed earlier. More precisely, in this section we compare the sampling techniques that are used extensively in various studies with sigmoid neural networks and two information criteria proposed previously in the construction of a wavelet network. More precisely, the FPE proposed by Zhang (1994), the GCV proposed by Zhang (1997), the bootstrapping (BS) and *v-fold* cross-validation (CV) methods proposed by Efron and Tibshirani (1993) and Zapranis and Refenes (1999) are tested as well as the performance of the BIC criterion.

The following procedure is followed to evaluate each method. First, the prediction risk according to each method is estimated up to a predefined maximum of hidden units. Then the number of hidden units that minimizes the prediction risk is selected for construction of the wavelet network, which will be fully trained. Finally, the MSE between the wavelet network output and the target function is estimated. The best network topology will be considered the one that produces the smallest MSE and shows no signs of overfitting.

The four methods are evaluated using the functions $f(x)$ and $g(x)$ introduced in Chapter 3. Both training samples consist of 1.000 training patterns, as in the preceding section. The wavelet networks are trained using the backpropagation algorithm with 0.1 learning rate and zero momentum. To estimate the prediction risk using the bootstrapping approach, 50 new networks were created for each hidden unit ($B = 50$). Similarly, the prediction risk using the cross-validation method was estimated using 50 subsamples for each hidden unit. In other words, 50-fold cross-validation was used ($v = 50$). All wavelet networks were initialized using the BE algorithm since our results in previous sections indicate that the BE outperforms the alternative algorithms.

Case 1: Sinusoid and Noise with Decreasing Variance

In this section we focus on the function $f(x)$. As in Chapter 3, the function $f(x)$ is given by

$$f(x) = 0.5 + 0.4 \sin 2\pi x + \varepsilon_1(x) \qquad x \in [0, 1]$$

where x is equally spaced in $[0, 1]$ and the noise $\varepsilon_1(x)$ follows a normal distribution with mean zero and a decreasing variance

$$\sigma_\varepsilon^2(x) = 0.05^2 + 0.1(1 - x^2)$$

In the first case we estimate the prediction risk for a wavelet network with no hidden units, and iteratively, 1 hidden unit is added until a maximum number of 20 hidden units is reached. Table 4.3 presents the prediction risk and the suggested hidden units for each information criterion for the two functions described previously. Four of the five criteria—the FPE, BIC, BS, and CV—suggest that a wavelet network with only 2 hidden units is sufficient to model function $f(x)$. On the other hand, using the GCV, the prediction risk is minimized when a wavelet network with 3 hidden units is used.

First, we examine graphically the performance of each criterion. Figure 4.4 shows the approximation of the wavelet network to the training data using (a) 1 hidden unit, (b) 2 hidden units, and (c) 3 hidden units. Figure 4.4d shows the training data and

TABLE 4.3 Prediction Risk and Hidden Units for the Four Information Criteria[a]

	FPE	GCV	BIC	BS	CV
Case 1					
Prediction risk	0.01601	0.03149	0.03371	0.03144	0.03164
Hidden units	2	3	2	2	2
Case 2					
Prediction risk	0.00231	0.00442	0.00524	0.00490	0.03309
Hidden units	8	15	8	8	8

[a]Case 1 refers to function $f(x)$ and case 2 to function $g(x)$. FPE, final prediction error; GCV, generalized cross-validation; BS, bootstrapping; CV, 50-fold cross-validation.

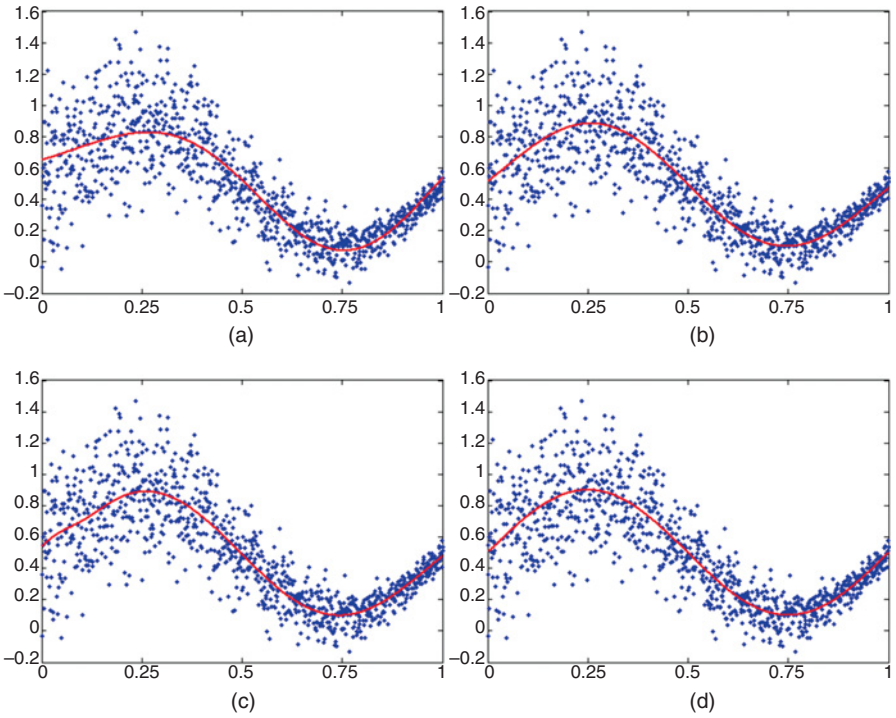

Figure 4.4 *Training a wavelet network with (a) 1, (b) 2, and (c) 3 hidden units. The target function is presented in part (d).*

the target function $f(x)$. It is clear that a wavelet network with only 1 hidden unit cannot learn the underlying function. On the other hand, wavelet networks with 2 and 3 hidden units approximate the underlying function very well. However, when 3 hidden units are used, the network approximation is affected by the large variation of the noise in the interval [0, 0.25]. To confirm the results above, the MSE between the output of the wavelet network and the underlying target function $f(x)$ is estimated. The MSE is 0.001825 when a wavelet network with only 1 hidden unit is used. Adding 1 more hidden unit, 2 in total, the MSE is reduced to only 0.000121. Finally, when 3 hidden units are used, the MSE increased to 0.000267. Hence, two wavelets should be used to construct a wavelet network to approximate function $f(x)$. The results above indicate that GCV suggested a more complex model than needed. Moreover, a wavelet network with 3 hidden units shows signs of overfitting.

From Table 4.3 it is shown that the FPE criterion suggests 2 hidden units; however, the prediction risk is only 0.01601, in contrast to GCV, BIC, BS, and CV, which is 0.03149, 0.03371, 0.03144, and 0.03164, respectively. To find the correct magnitude of the prediction risk, a validation sample is used to measure the performance of the wavelet network with 2 hidden units in out-of-sample data. The validation sample consists of 300 patterns randomly generated by $f(x)$. These patterns were not used

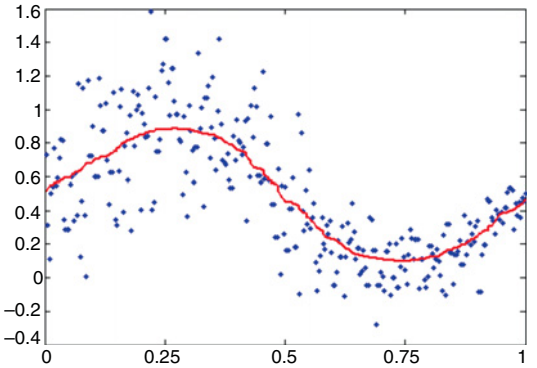

Figure 4.5 *Out-of-sample prediction for the first case.*

for training of the wavelet network. The MSE between the network forecasts and the out-of-sample targets is 0.048751, indicating that the FPE criterion is too optimistic as to estimation of the prediction risk. The approximations forecast for the wavelet network and the out-of-sample target values are shown in Figure 4.5.

Case 2: Sum of Sinusoids and Cauchy Noise

In the second part of Table 4.3, the results are presented for the model selection algorithm for the function $g(x)$. The function $g(x)$ is given by

$$g(x) = 0.5x \sin x + \cos^2 x + \varepsilon_2(x) \qquad x \in [-6, 6]$$

and $\varepsilon_2(x)$ follows a Cauchy distribution with location 0 and scale 0.05. As in the first case, the prediction risk for a wavelet network with no hidden units is estimated, and 1 hidden unit is added iteratively to the wavelet network until the predefined maximum number of 20 hidden units is reached. The FPE criterion suggests that 7 hidden units are appropriate for modeling the function $g(x)$. On the other hand, using GCV, the prediction risk is minimized when a wavelet network with 15 hidden units is used. Finally, using the BIC, bootstrapping, and the cross-validation criteria, the prediction risk is minimized when a wavelet network with 8 hidden units is used.

In Figure 4.6 the approximation of the wavelet network to the training data using (a) 7, (b) 8, and (c) 15 hidden units is presented. Part (d) of the figure shows the target function $g(x)$ and the training data. It is clear that all networks produce similar results, and it is difficult to compare them visually. To compare the results above, the MSE between the output of the wavelet network and the underlying target function $g(x)$ was estimated. The MSE is 0.001611 when a wavelet network with only 7 hidden units is used. Adding one more hidden unit, 8 in total, the MSE is reduced to only 0.000074, which is also the minimum MSE achieved. Adding additional hidden units results in an increase in the MSE between the underlying function $g(x)$ and the wavelet network. Finally, when 15 hidden units are used, the MSE increased to

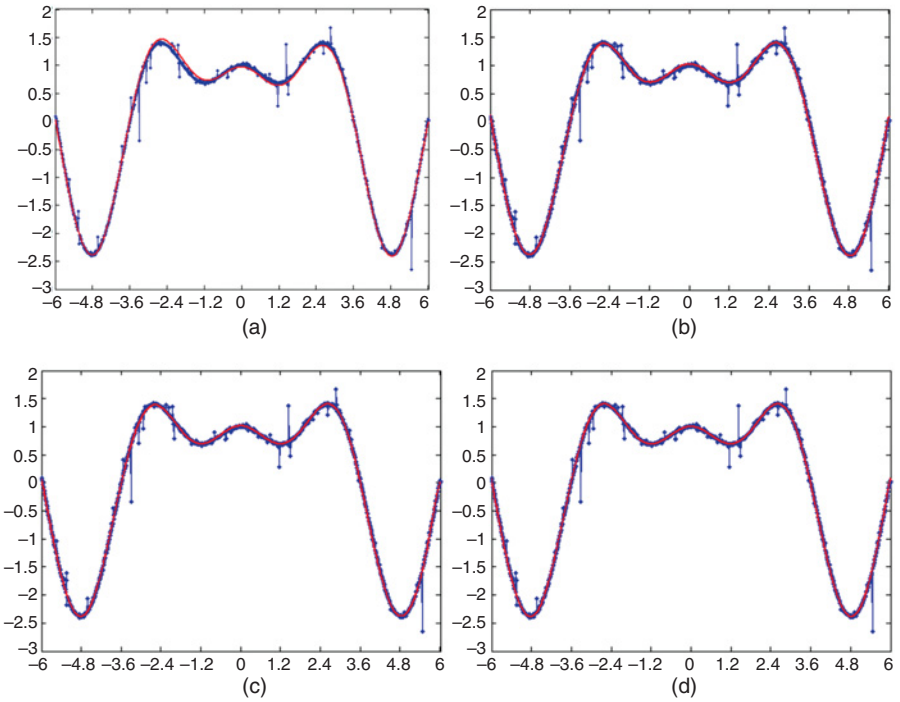

Figure 4.6 *Training a wavelet network with (a) 7, (b) 8, and (c) 15 hidden units. The target function is presented in part (d).*

0.000190. To find the best network, the MSE between the network approximation and the underlying function is estimated for a wavelet network with up to 20 hidden units. The MSE is minimized when a network with 8 hidden units is used. Hence, the optimum number of wavelets to approximate the function $g(x)$ is 8. The results above indicate that GCV suggests a more complex model, while the FPE suggests a simpler model than needed. On the other hand, our results indicate that the BIC and the sampling techniques again proposed the correct topology of the wavelet network.

As reported in Table 4.3, the estimated prediction risk proposed by the FPE criterion is 0.002312, in contrast to GCV, BIC, BS, and CV, which is 0.00442, 0.005238, 0.00490, and 0.00331, respectively. To find the correct magnitude of the prediction risk a validation sample is used to measure the performance of the wavelet network with 8 hidden units in out-of-sample data. The validation sample consists of 300 patterns randomly generated by the function $g(x)$. These patterns were not used for the training of the wavelet network. The MSE between the network forecasts and the out-of-sample targets is 0.0043. Our results again indicate that the FPE criterion is too optimistic on the estimation of the prediction risk. The out-of-sample data and the forecast produced by the wavelet network are shown in Figure 4.7.

A closer inspection of Figure 4.6 reveals that the wavelet network approximation was not affected by the presence of large outliers, in contrast to the findings of Li

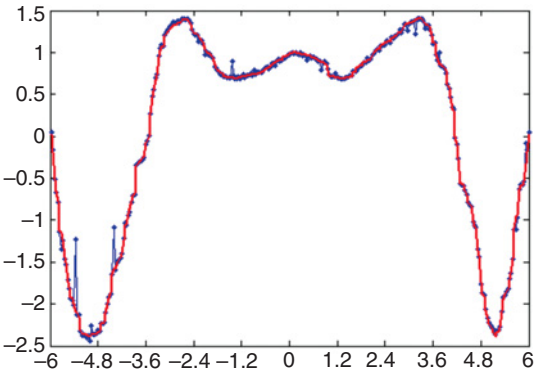

Figure 4.7 *Out-of-sample prediction for the second case.*

and Chen (2002). In this study 8 hidden units were used to construct the wavelet network, as proposed by v-fold cross-validation and bootstrapping, while in Li and Chen (2002) the architecture of the wavelet network had 10 hidden units, as proposed by the FPE criterion. Our results indicate that the FPE criterion does not perform as well as sampling techniques (bootstrapping or v-fold cross-validation) and should not be used.

Model Selection without Training Estimation of the preferred information criteria was performed by Zhang (1997) after the initialization stage of the network. More precisely, in the SSO and RBS the preferred information criterion is evaluated after the selection of each wavelet in the initialization stage. Similarly, when the BE algorithm is used, the preferred information criteria are evaluated after the elimination of each wavelet in the initialization stage. Since initialization of the wavelet network is very good, the initial approximation is expected to be very close to the target function. Hence, a good approximation of the prediction risk is expected to be obtained. The same idea can also be applied when bootstrap or cross-validation is used. The procedure above is significantly less computationally expensive since training additional wavelet networks is not required for the bootstrapped samples.

However, the procedure above is similar to early stopping techniques. Usually, early stopping techniques suggest a network with more hidden units than necessary, although the network is not fully trained to avoid overfitting (Samarasinghe, 2006), while they do not work satisfactorily in complex problems.

In the first case the results were similar to the case where the wavelet networks were fully trained. More precisely, the FPE, bootstrap, and cross-validation methods suggested that a wavelet network with 2 hidden units is sufficient to model $f(x)$, while GCV suggested a wavelet network with 3 hidden units. In the second case, both the information criteria and sampling techniques suggested that a wavelet network with more than 14 hidden units is needed to model function $g(x)$. On the other hand, the BIC criterion suggested correctly that 8 hidden units should be used for construction of the wavelet network. However, as mentioned, the BIC assumes the asymptotic

normality of the maximum likelihood estimators. Moreover, from our experience, in general, information criteria do not perform very well in complex applications. The results above indicate that when more complex problems are introduced, as in the second case, this method does not work satisfactorily.

Since sampling techniques are computationally expensive methods, the BIC criterion can be used initially to approximate the optimal number of wavelons. Then the bootstrap or cross-validation methods can be used (e.g., in ± 5 hidden units), around the number of hidden units proposed by the BIC to define the best network topology.

ADAPTIVE NETWORKS AND ONLINE SYNTHESIS

In contrast to previous constructive methods, online approaches do not require that the number of wavelets be determined before the start of the training (Wong and Leung, 1998). For some applications where real-time responses of the wavelet network are crucial, online approaches, can be useful. In off-line approaches, if there is a change in the system parameters, the trained network may not be able to adapt to the change and has to be retrained. On the other hand, online training and synthesis methods allow the parameters to be updated after the presentation of each training pattern. New wavelets are added to the network when it is needed, while wavelets that do not contribute to the performance of the network anymore are removed.

Cannon and Slotine (1995) and Wong and Leung (1998) used online synthesis in the construction of wavelet networks. Similarly, Xu and Ho (1999) proposed and introduced a wavelet network for adaptive nonlinear system identification. The basis functions were selected online according to the local spatial frequency content of the approximated function. The adaptive weight updating was based on Lyapunov stability theory.

In general, an adaptive algorithm can be summarized in the following steps. First, the appropriate time that a new wavelet should be added to the network must be determined. Second, if a new wavelet is entered in the network, the optimal position in which it should be placed must be found. Third, it must be determined if a wavelet entered previously no longer contributes to the wavelet network approximation and if it should be removed. Finally, the stopping condition of the algorithm should be determined.

For the first step, the following procedure is utilized. Assume that, so far, a wavelet network with j wavelets has been constructed. After each iteration of the training algorithm the MSE between the network approximation and the target values is measured. If the MSE is stabilized after a certain point, a new wavelet is entered in the wavelet network. More precisely, the structure of the wavelet network is increased by one wavelon (hidden unit). To determine if the MSE is stabilized, the difference of the MSE between time step t and $t-1$ can be compared to a fixed threshold:

$$\mathrm{MSE}_t - \mathrm{MSE}_{t-1} < \text{threshold} \tag{4.22}$$

or alternatively the ratio of the MSE between time step t and $t-1$:

$$\frac{\text{MSE}_t - \text{MSE}_{t-1}}{\text{MSE}_t} < \text{threshold} \tag{4.23}$$

In the second step, the optimal position of the new wavelet must be found. In other words, the newly entered wavelet must be initialized. At the previous stage, when j wavelets were used, the residuals between the wavelet network approximation and the target patterns could be found. Hence, the new wavelet, $j+1$, can be initialized on the residuals using the BE method.

Next, the wavelets that were entered in the model previously must be examined. The contribution of each wavelet to the total wavelet network output is examined. When the dynamics or the parameters of the underlying system were changed, there is a possibility that some of the wavelets no longer provide any useful information. To remove unnecessary wavelons, pruning methods can be employed.

The final problem that must be solved is to determine the stopping time of the algorithm. If a stopping criterion is not defined, it is possible to construct very large wavelet networks. Moreover, there is a possibility for the algorithm to be trapped in an infinite loop between steps 2 and 3. After a new wavelet is entered in the structure of the wavelet network, an information criterion such as the BIC can be estimated. If the new wavelet causes the BIC to increase, the algorithm stops.

Adaptive wavelet network can be useful in applications where dynamic systems are examined. More precisely, in cases where the parameters of the dynamic systems change, a response of the wavelet network is needed in real time. However, in problems of function approximation, the results from Wong and Leung (1998) indicate that this method is very prone to the initialization of the wavelet network. Their results indicate that the suggested topology of a particular function approximation problem varied from 4 to 10 hidden units.

CONCLUSIONS

When building a wavelet network a crucial decision that needs to be made is to choose the number of the wavelons or hidden units. The number of the wavelons also defines the architecture of the wavelet network. A network with fewer hidden units than needed would not be able to learn the underlying function; selecting more hidden units than needed will result in an overfitted model. In both cases the wavelet network cannot be used for forecasting. Moreover, the results of an analysis based on an overfitted or underfitted wavelet network are not credible or reliable.

In this chapter, various methods of selecting the optimal number of hidden units were presented and tested. More precisely, three information criteria and two resampling techniques were evaluated on two simulated cases. The five methods were used to estimate the prediction risk. In general, the resampling techniques outperformed the information criteria. In addition, both bootstrapping and cross-validation are not

based on restrictive assumptions as in the case of information criteria. However, resampling schemes are computationally expensive methods.

The information criteria performed satisfactorily on the simple example. However, the results were not good on the more complex problem, with the exception of the BIC. In general, the results obtained from BIC were very stable, and this can constitute a guideline to reduce the computational burden of the resampling techniques.

Alternatively, when dynamic systems are examined and a response from the wavelet network is needed in real time, an adaptive wavelet network can prove useful. However, the wavelet forecasts of a wavelet that was constructed adaptively may be unstable.

REFERENCES

Aczel, A. D. (1993). *Complete Business Statistics*. Irwin, Homewood, IL.

Akaike, H. (1973). "Information theory and an extension of the maximum likelihood principle." In *Second International Symposium of Information Theory*, B. N. Petrov and F. Csaki, eds., Akademiai Kiado, Budapest, 267–281.

Akaike, H. (1974). "New look at statistical-model identification." *IEEE Transactions on Automatic Control*, Ac19(6), 716–723.

Anders, U., and Korn, O. (1999). "Model selection in neural networks." *Neural Networks*, 12(2), 309–323.

Cannon, M., and Slotine, J.-J. E. (1995). "Space–frequency localized basis function networks for nonlinear system estimation and control." *Neurocomputing*, 9, 293–342.

Curry, B., and Morgan, P. H. (2006). "Model selection in neural networks: some difficulties." *European Journal of Operational Research*, 170(2), 567–577.

Dimopoulos, Y., Bourret, P., and Lek, S. (1995). "Use of some sensitivity criteria for choosing networks with good generalization ability." *Neural Processing Letters*, 2(6), 1–4.

Efron, B., and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall, New York.

Li, S. T., and Chen, S.-C. (2002). "Function approximation using robust wavelet neural networks." *Proceedings of ICTAI '02*, Washington, DC, 483–488.

Moody, J. E. (1992). "The effective number of parameters: an analysis of generalization and regularization in nonlinear learning systems." In *Advances in Neural Information Processing Systems*, J. E. Moody, S. J. Hanson, and R. P. Lippman, eds., Morgan Kaufmann, San Mateo, CA.

Mosteller, F., and Tukey, J. (1968). "Data analysis, including statistics". In *Handbook of Social Psychology*, G. Lindzey and E. Aronson, eds. Addison-Wesley, Reading, MA, Chap. 10.

Murata, N., Yoshizawa, S., and Amari, S. (1994). "Network information criterion-determining the number of hidden units for an artificial neural network model." *IEEE Transactions on Neural Networks*, 5(6), 865–872.

Reed, R. (1993). "Pruning algorithms: a survey." *IEEE Transactions on Neural Networks*, 4, 740–747.

Samarasinghe, S. (2006). *Neural Networks for Applied Sciences and Engineering*. Taylor & Francis, New York.

Wong, K.-W., and Leung, A. C.-S. (1998). "On-line successive synthesis of wavelet networks." *Neural Processing Letters*, 7, 91–100.

Xu, J., and Ho, D. W. C. (1999). "Adaptive wavelet networks for nonlinear system identification." *Proceedings of American Control Conference*, San Diego, CA.

Zapranis, A. D., and Haramis, G. (2001). "An algorithm for controlling the complexity of neural learning: the irrelevant connection elimination scheme." *Fifth Hellenic European Research on Computer Mathematics and Its Applications*, Athens, Greece.

Zapranis, A., and Refenes, A. P. (1999). *Principles of Neural Model Indentification, Selection and Adequacy: With Applications to Financial Econometrics*. Springer-Verlag, New York.

Zhang, Q. (1993). "Regressor selection and wavelet network construction." Technical Report, INRIA.

Zhang, Q. (1994). "Using Wavelet Network in Nonparametric Estimation." Technical Report, 2321, INRIA.

Zhang, Q. (1997). "Using wavelet network in nonparametric estimation." *IEEE Transactions on Neural Networks*, 8(2), 227–236.

Zhang, Q., and Benveniste, A. (1992). "Wavelet networks." *IEEE Transactions on Neural Networks*, 3(6), 889–898.