Pergamon

Computers chem. Engng, Vol. 21, Suppl., pp. S1025-S1030, 1997

Printed in Great Britain

PII:S0098-1354(97)00184-1

0098-1354/97 $17.00+0.00

Inferential Estimation of Polymer Quality Using Stacked Neural Networks

J. Zhang, E. B. Martin*, A. J. Morris, and C. Kiparissides+

Centre for Process Analysis, Chemometrics and Control
Department of Chemical and Process Engineering; Department of Engineering Mathematics *
University of Newcastle, Newcastle upon Tyne NE1 7RU, United Kingdom
Department of Chemical Engineering, Aristotle University of Thessaloniki, P.O. Box 472, 54006 Greece+
E-mail: julian.morris@ncl.ac.uk; jie.zhang@ncl.ac.uk; e.b.martin@ncl.ac.uk

Abstract - The robust inferential estimation of polymer properties using stacked neural networks is presented. Data for building non-linear models is re-sampled using bootstrap techniques to form a number of sets of training and test data. For each data set, a neural network model is developed which are then aggregated through principal component regression. Model robustness is shown to be significantly improved as a direct consequence of using multiple neural network representations. Confidence bands for the neural network model predictions also result directly from the application of the bootstrap technique. The approach has been successfully applied to the building of software sensors for a batch polymerisation reactor.

INTRODUCTION

A major problem in the control of product quality in industrial polymerisation reactors is the lack of suitable on-line polymer quality measurements. Although instruments for measuring the number average molecular weight and the weight average molecular weight are available, these instruments possess substantial measurement delays. Some of these difficult-to-measure variables can however be related to certain easily measurable variables such as temperature, solution viscosity and density of the reaction mixture. Inferential estimators, or software sensors, of these difficult-to-measure 'quality' variables can then be derived from measurements of the more easily measured process variables. The key step in inferential estimation is to establish a relationship between the difficult-to-measure quantities and the more easily measured variables. One popular approach is through the use of a first principles mechanistic model of the process and state estimation techniques such as the extended Kalman filter (EKF), (Shuler and Zhang, 1985; Ellis *et al*, 1988; Dimitrators *et al*, 1989; Kozub and MacGregor, 1992). These approaches, however, require a deep understanding of the polymerisation process and consequently model development is usually very demanding in production processes; even for pilot plant models which involve large sets of differential, algebraic and kinetic equations, (Kiparissides, 1996).

To overcome this difficulty, especially in industrial polymerisation, neural network representations based upon monitored reactor data can be developed. Neural networks have been shown to be able to approximate any continuous non-linear functions, (Cybenko, 1989; Girosi and Poggio, 1990; Park and Sandberg, 1991) and have been widely applied to non-linear process modelling, (Bhat and McAvoy, 1990; Willis *et al*, 1991, Morris *et al*, 1994). Using the learning capability of a neural network, the relationship between polymer quality variables and the on-line measured variables in the reactor can be identified from the reactor operation data.

An important requirement of inferential estimators is that they should not only provide satisfactory estimation accuracy but also be robust to new plant data. The accuracy and robustness of a neural network model is strongly influenced by the availability of training data. When the amount of training data is limited, the network tends to over-fit and hence exhibit significant generalisation errors. To build an accurate and robust neural network model, ideally a large amount of training data should be made available. With today's on-line process data monitoring facilities, it is generally assumed that good process data is readily available, however, in practice in many industrial plants the collection of sufficient, appropriate 'good quality data' is still a real problem. In addition, product 'quality' information is usually limited due to analyser constraints, laboratory costs and long time delays before the measurement is made available. Limited process data is a serious problem in the development of accurate and robust network representations.

To address the problem of limited process data, stacked neural network models have been proposed, to improve both neural network model accuracy and robustness, (Wolpert, 1992; Breiman 1992). Stacked generalisation is a technique which combines different representations to improve the overall prediction properties. Process data is randomly re-sampled to form a number of different training and test data sets. Neural networks are then developed based upon each re-sampled data set. However, instead of selecting a perceived 'best' single neural network for prediction purposes, several networks are combined (aggregated) and the aggregated predictor is used as the final representation. This approach is applied to a pilot scale batch polymerisation reactor where the polymerisation of methyl methacrylate (MMA) takes place.

STACKED NEURAL NETWORKS

Neural network representations of system input-output data are typically approximations to the real system. The quality of a neural network model, or its-fitness-for-purpose, depends upon the amount and appropriateness of the available training data and the network training method. It is inappropriate to assume that a neural network trained on limited training data will be a good representation of the actual system for a variety of reasons. For example, the optimum network structure is not easy to pre-specify; the optimisation of the network weights can lead to local minima; different learning algorithms can result in contrasting generalisation characteristics; and alternative convergence criteria for network training can also result in different network solutions.

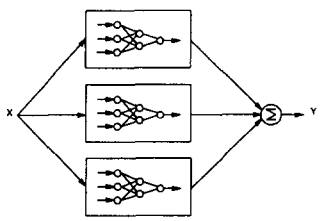

Figure 1. Stacking a series of Neural Networks

A promising approach to improve model accuracy is to combine several neural network models. Since each neural network representation can behave differently in different regions of the input space, representational accuracy over the entire input space can be improved by combining several neural network models. The combination, or aggregation, of several neural networks has been termed, stacking, Figure 1, (Wolpert, 1992; Sridhar *et al*, 1996).

In this paper, several neural network models are developed to represent the same underlying process, relationship, these are then combined in an appropriate manner. Instead of selecting a single 'best' network model, a stacked network which combines a number of network models to improve overall representational accuracy and robustness. The overall output of the stacked neural network is a weighted combination of the individual network outputs and is represented by:

$$f(X) = \sum_{i=1}^{n} w_i f_i(X) \qquad (1)$$

where $f(U)$ is the stacked network predictor, $f_i(U)$ is the ith network predictor, w_i is the stacking weight for combining the ith network, and U is a vector of network inputs.

Stacking weights can be determined in a number of ways. A simple, but inappropriate approach is to take equal weights for the individual networks. A second way is to obtain the weights through multiple linear regression. However, care needs to be taken because of the highly correlated nature of the individual network predictors since each network is trained to model the same underlying relationship. Stacking weights through multiple linear regression does not appear to give good results. This was also shown by Breiman (1992). He suggested constraining the stacking weights to be non-negative. A novel approach to calculating the appropriate stacking weights is through principal component regression (PCR).

Let y be a vector of the expected model outputs and $\hat{y}_i$ be a vector of the predictions from the ith neural network predictor. Predictions from a set of n predictors can be described as follows:

$$\hat{Y} = [\hat{y}_1 \hat{y}_2 ... \hat{y}_n] \qquad (2)$$

where each column corresponds to an individual predictor. The vector of predictions from the stacked neural network model, $\hat{y}_{stack}$, can then be represented as:

$$\hat{y}_{stack} = \hat{Y}w = w_1\hat{y}_1 + w_2\hat{y}_2 + ... + w_n\hat{y}_n \qquad (3)$$

The matrix $\hat{Y}$ can be decomposed into the sum of a series of rank one matrices through principal component decomposition.

$$\hat{Y} = t_1 p_1^T + t_2 p_2^T + ... + t_N p_N^T \qquad (4)$$

where t_i and p_i are the ith score vector and loading vector respectively. The score vectors are orthogonal, likewise the loading vectors. In addition they are of unit length. The loading vector p_1 defines the direction of greatest variability and the score vector t_1, also known as the first principal component, represents the projection of each column of $\hat{Y}$ onto p_1. Thus the first principal component is that linear combination of the columns in $\hat{Y}$ explaining the greatest amount of variability ($t_1 = \hat{Y} p_1$). The second principal component is that linear combination of the columns in $\hat{Y}$ explaining the next greatest amount of variability ($t_2 = \hat{Y} p_2$), subject to the condition that it is orthogonal to the first principal component. Since the columns in $\hat{Y}$ are highly correlated, the first few principal

components can explain the majority of variability in $\hat{Y}$.

Through PCR, the stacked neural network output is a linear combination of the first few principal components of $\hat{Y}$. Suppose that the first k principal components are used in PCR and they are denoted by T_k and $T_k = \hat{Y} P_k$, where $P_k = [p_1\ p_2\ \cdots\ p_k]$, then the stacked network model can be represented as:

$$\hat{y}_{stack} = T_k\theta = \hat{Y}P_k\theta \qquad (5)$$

The least squares estimation of θ is then:

$$\theta = (T_k^T T_k)^{-1} T_k^T y$$
$$= (P_k^T \hat{Y}^T \hat{Y} P_k)^{-1} P_k^T \hat{Y}^T y \qquad (6)$$

and the stacking weight vector w calculated through PCR becomes:

$$w = P_k\theta = P_k (P_k^T \hat{Y}^T \hat{Y} P_k)^{-1} P_k^T \hat{Y}^T y \qquad (7)$$

The weights determined through PCR give good performance.

ESTIMATION OF POLYMER QUALITY IN A BATCH POLYMERISATION REACTOR

The batch polymerisation reactor

The reactor studied here, and shown in Figure 2, is a pilot scale polymerisation reactor situated in the Department of Chemical Engineering, Aristotle University of Thessaloniki, Greece. The reaction is the free-radical solution polymerisation of methyl methacrylate (MMA). The solvent used is water and the initiator is benzoyl peroxide. The jacketed reactor is provided with a stirrer for thorough mixing of the reactants. Heating and cooling of the reaction mixture is achieved by circulating water at an appropriate temperature through the reactor jacket. The reactor temperature is controlled by a cascade control system consisting of a primary PID and two secondary PID controllers. The reactor temperature is fed back to the primary controller whose output is taken as the setpoint of the two secondary controllers. The manipulated variables for the two secondary controllers are the hot and cold water flow rates. The hot and cold water streams are mixed before entering the reactor jacket and provide heating or cooling for the reactor. The jacket outlet temperature is fed back to the two secondary controllers.

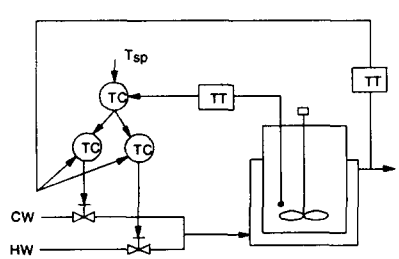

Figure 2. The batch MMA polymerisation reactor

Different grades of polymer product are produced by employing different batch recipes. The three key elements in a batch recipe are the reactor temperature setpoint, reactor wall fouling and the initial initiator weight. During the course of polymerisation, the reactor temperature is kept constant by the cascade temperature controller. A detailed mathematical model covering reaction kinetics and heat and mass balances has been developed. Based upon this model a rigorous simulation programme is used to generate polymerisation data under different batch operating conditions.

Inferential estimation of polymer quality

The on-line measured process variables are reactor temperature, jacket inlet temperature, jacket outlet temperature, monomer conversion (X), and coolant flow rate. Polymer property variables, number average molecular weight (M_n) and weight average molecular weight (M_w), are in practice not measured through the batch but require to be estimated from the on-line process measurements.

Batch No.	T_{sp} (°K)	I_0 (g)
1	343	2.5
2	348	3.0
3	338	2.0
4	343	2.8
5	346	2.0
6	350	1.8
7	332	3.5
8	340	2.6
9	345	2.6
10	342	2.2
11	335	2.4

Table 1. Batch Recipes

The evolution of the polymer properties during the course of polymerisation are mainly determined by the batch recipe, i.e. the reactor temperature setpoint, reactor wall fouling and the initial initiator weight. Different batch recipes will lead to different polymer growth profiles coupled with different heat generation profiles. Correlation analysis of the reactor operational data indicates that there is a strong relationship between the polymer properties and the reactor and jacket temperatures, the coolant flow rate, and monomer conversion. The reactor temperature setpoint, the initial initiator weight, the jacket inlet and outlet temperatures, the reaction time, the coolant flow rate through the reactor jacket, and monomer conversion are used to estimate the polymer quality variables. The nominal batch time for this reactor is 180 minutes. In this study, data from nine batches were used to develop neural network based inferential estimators. In designed experiments tests for the nine batches, off-line polymer property measurements are taken every 20

minutes to provide 9 sets of property analyses. Two additional batches, with different batch recipes from the nine batches, were used to validate the neural network based inferential estimator. Batch recipes for the eleven batches are shown in Table 1. Process measurements are corrupted with typical measurement noise of zero mean and 10% of the deviation of the corresponding variable.

Stacked neural network estimators for estimating the number average molecular weight M_n *(g/mol)* and weight average molecular weight M_w *(g/mol)* were developed. Each estimator comprises 30 neural network representations. The individual networks for estimating M_n and M_w take the following forms:

$$M_n(t)=f_1(T_{sp},I_0,t,T_{in}(t),T_{out}(t),F(t),X(t)) \quad (8)$$

$$M_w(t)=f_2(T_{sp},I_0,t,T_{in}(t),T_{out}(t),F(t),X(t)) \quad (9)$$

In these equations, T_{sp} (^oK) is the reactor temperature setpoint, I_0 (g), initial initiator weight, T_{in} (^oK), jacket inlet temperature, T_{out} (^oK), jacket outlet temperature, F (cm^3/s), coolant flow rate, X, monomer conversion, t $(min.)$, time from the beginning of a batch, and $f_1(.)$ and $f_2(.)$ are the underlying non-linear functions defined by the networks.

Training data for each neural network was selected from batches 1, 2, 3, 5, and 7 through the bootstrap technique, re-sampling with replacement, (Efron, 1982). Data from batches 4, 6, 8, and 9 were then used as the test data set. Batches 10 and 11 formed the unseen validation data set. The number of hidden neurons for each individual network in the stacked network was determined by considering several neural networks where the number of hidden neurons ranged from 5 to 25. The network giving the least error on the test data was selected. Each neural network was trained using the Levenberg-Marquardt optimisation algorithm together with a cross validation based "early stopping" mechanism to prevent over-fitting. During network training, the training algorithm continuously checks the network error on the test data set. Training was terminated at the point where the network error on the test data was at a minimum. Early stopping is an implicit way to implement regularisation which can improve model robustness, (Sjoberg *et al*, 1995). The training strategy implemented has the advantages of speed and not over-fitting the noise in the data. The weights for combining the individual neural networks were determined through PCR. Two principal components were retained in the PCR models, i.e. k=2 in Equations (5) to (7).

Estimation of M_n and M_w from the stacked neural network models on the validation data are plotted in Figure 3. Here the solid lines represent the measured values and the dotted lines represent the estimated values. The estimations from the stacked neural networks provide acceptable predictions of the actual reactor property profiles. The resulting predictions are more robust than those achievable from a single network predictor as seen from the histograms in Figure 4. These show the MSE (mean squared error) for the individual neural network models for the estimation of M_n on the training, testing, and validation data. It can be observed that these models give variable performances. Indeed, it can be seen that the eighth and the ninth neural network models give similar performance for the estimation of M_n on both the training and testing data. However, their performance on the validation data differs drastically. The 'best' individual network on training and testing data is the twelfth network, but its performance on the validation data is **not** the 'best' amongst all the networks. These studies together with many others, indicate the non-robust nature of single neural network models even when they might be selected as being theoretically 'the best'. Figure 5 shows the MSE of the stacked neural network models for estimating M_n on the training, testing, and validation data. The x-axis in each plot represents the number of neural networks in the stacked model.

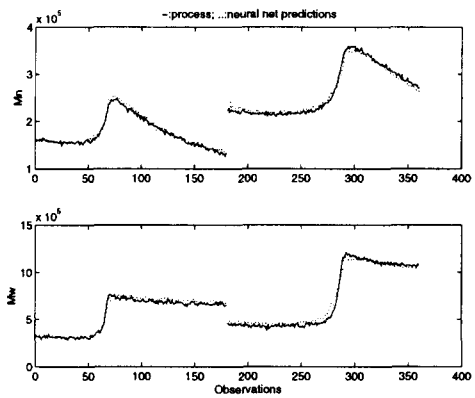

Figure 3. Stacked Estimation of M_n and M_w (Batch 10: 0 to 180; Batch 11: 181 to 361)

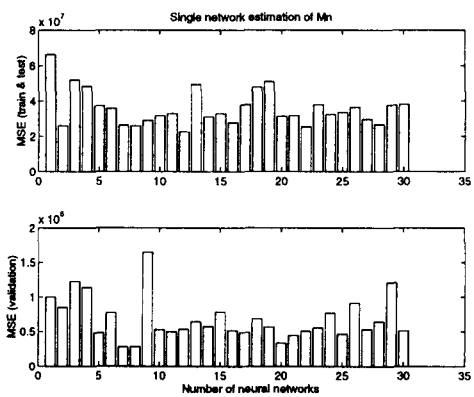

Figure 4. Single network MSE errors for the Train, Test and Validation data sets for the estimation M_n

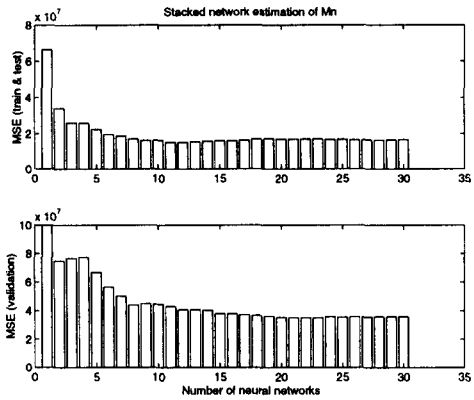

Figure 5. Stacked network MSE errors for the Train, Test and Validation data sets for the estimation M_n

It can be observed that the errors generally decrease with the number of individual neural network representations and gradually approach stable levels. The model errors of stacked neural network models on training, test and validation data become consistent. This is in sharp contrast to the single neural network models shown in Figure 4. This clearly demonstrates that stacked neural network models are more robust than single neural network models. By comparing Figures 4 and 5, it can be observed that the estimation accuracy of the stacked network on the unseen validation data is much higher than for most of the individual networks.

From Figure 5 it can be concluded that the model errors are stabilised after employing 20 networks. From experience on a wide range of case studies, it can be concluded that stacking 20 to 30 networks is usually sufficient to stabilise the model errors. In practice it is recommended that 30 networks are developed. Work is on-going to identify the number of networks to stack, using statistical procedures.

A major problem in the industrial application of neural network models is the current lack of confidence bounds. This is particularly the case for non-linear systems where non-parametric confidence bounds are required (Martin and Morris, 1996). Bootstrap techniques can be used to estimate the standard error of model predictions (Efron and Tibshirini, 1993; Tibshirani, 1996). Based on the estimated standard error, confidence bounds for neural network predictions can be calculated. The 95% confidence bounds for the stacked neural network estimation on the two validation batches (10 and 11) are shown in Figure 6. For the sake of clarity, the estimations and confidence bounds are only shown at 10 minute intervals. The narrower the confidence bounds, the higher is the confidence in the estimation. Although predictions for both validation batches are acceptable, Figure 6 clearly shows that the predictions for batch 10 have tighter confidence bounds than those for batch 11. By examining the batch recipes in Table 1, it can be seen that the recipe for batch 10 is closer to the recipes of the five training batches than for batch 11. The scaled distance between the recipe for batch 10 and those for the training batches is 0.2913, whilst for batch 11, it is 0.7315. This explains why the confidence bounds are tighter for batch 10 compared with batch 11.

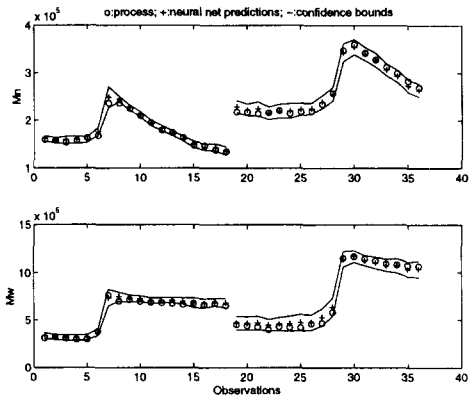

Figure 6. Stacked neural network estimations with confidence bounds
(Batch 10: 1 to 18; Batch 11: 19 to 36)

CONCLUSIONS AND DISCUSSION

Stacked neural networks are used to estimate polymer quality variables from easy-to-measure process variables. Training data for building neural networks is re-sampled using bootstrap techniques to form several sets of training data. For each set, a neural network model is developed and the individual networks stacked. Predictions from the stacked neural networks are then obtained as a combination of the predictions from individual neural networks with the stacking coefficients being calculated through principal component regression. In this way, the robustness and accuracy of the neural network models is significantly improved. A further benefit from using the bootstrap re-sampled data sets is that confidence bounds for the model predictions can automatically be calculated.

Rassmussen (1996) briefly studied the effect of Brieman's bootstrap aggregated regression (bagging) approach on Multivariate Adaptive Regression Splines (MARS). MARS models were developed for each of a number of bootstrap samples which comprised the training set and the results were then averaged. The results indicated that bagging can led to improved expected performance, with a substantial improvement for small training data sets. For larger data sets, the benefits of bagging are less clear cut. It is conceivable, that for large data sets bagging may degrade performance. Similar results have been achieved with bagging, ensembles of networks and with stacked networks. For small training data sets, potentially resulting in some poor models, the combined model

residuals are 'whitened' giving rise to a better overall performance.

The empirical evidence suggests that the benefits of aggregating models outweighs the decline in performance of the individually trained representations. When sufficient training data is available, a model performance close to the theoretical limit (the inherent variability of the data) can be achieved and aggregation of an ensemble of models does not provide any significant improvement in overall performance.

ACKNOWLEDGEMENTS

The authors acknowledge the support of the Centre for Process Analysis, Chemometrics and Control and the EU BRITE/EURAM Project INTELPOL BE-7009/93.

NOTATION

F: coolant flow rate (cm^3/s)
I_0: initial initiator weight (g)
M_n: number average molecular weight (g/mol)
M_{nw}: weight average molecular weight (g/mol)
T_{in}: jacket inlet temperature (^oK)
T_{out}: jacket outlet temperature (^oK)
T_{sp}: reactor temperature setpoint (^oK)
X: monomer conversion
t: reaction time ($min.$)

REFERENCES

Breiman, L., 1992, Stacked regressions, *Technical Report No. 367*, Department of Statistics, University of California at Berkeley, USA.

Bhat, N. V. and T., J. McAvoy, 1990, Use of neural nets for dynamical modelling and control of chemical process systems, *Computers and Chemical Engineering*, **14**, 573-583.

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

Dimitrators, J., C. Georgakis, M. S. El-Aasser, and A. Klein, 1989, Dynamic modelling and state estimation for an emulsion copolymerization reactor, *Computers and Chemical Engineering*, **13**, 21-33.

Efron, B., 1982, *The Jackknife, the Bootstrap and Other Resampling Plans*, Society for Industrial and Applied Mathematics, Philadelphia.

Efron, B. and R. Tibshirani, 1993, *An Introduction to the Bootstrap*, Chapman and Hall, London.

Ellis M. F., T. W. Taylor, V. Gonzalez, and K. F. Jensen, 1988, Estimation of the molecular weight distribution in batch polymerization, *AIChE Journal*, **34**, 1341-1353.

Girosi, F. and T. Poggio, 1990, Networks and the best approximation property, *Biological Cybernetics*, **63**, 169-179.

Haesloop, D. and B. R. Holt, 1990, A neural network structure for system identification, *Proc. ACC*, 2460-2465.

Kiparissides, C., 1996, Polymerisation reactor modeling: A review of recent developments and future directions, *Chemical Engineering Science*, **51**, 1637-1659.

Kozub, D. J. and J. F. MacGregor, 1992, State estimation for semi-batch polymerization reactors, *Chemical Engineering Science*, **47**, 1047-1062.

Martin, E. B. and A. J. Morris, 1996, Non-parametric confidence bounds for process performance monitoring charts, *Journal of Process Control*, **6**, 349-358.

Martin, J. R., R. W. Nunes, J. F. Johnson, 1982, Influence of molecular-weight and molecular-weight distribution on mechanical-properties of polymer, *Polymer Engineering Science*, **22**, 205-228.

Rasmussen, C.E., 1996, Evaluation of gaussian processes and other methods for non-linear regression, *PhD Thesis*, Department of Computer Science, University of Toronto.

Schulur, H. and S. Zhang, 1985, Real time estimation of chain length distribution in a polymerisation reactor, *Chemical Engineering Science*, **40**, 1891-1904.

Sjoberg, J., Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P. Y. Glorennec, H. Hjalmarsson, and A. Juditsky, 1995, Nonlinear black-box modelling in system identification: a unified overview, *Automatica*, **31**, 1691-1724.

Sridhar, D. V., R. C. Seagrave, and E. B. Bartlett, 1996, Process modelling using stacked neural networks, *AIChE Journal*, **42**, 2529-2539.

Tibshirani, R., 1996, A comparison of some error estimates for neural network models, *Neural Computation*, **8**, 152-163.

Willis, M. J., C. Di Massimo, G. A. Montague, M. T. Tham, and A. J. Morris, 1991, On artificial neural networks in process engineering, *Proceedings of IEE, Part D*, **138**, 256-266.

Wolpert, D. H., 1992, Stacked generalisation, *Neural Networks*, **5**, 241-259.