Neural modelling of polypropylene fibre processing: Predicting the structure and properties and identifying the control parameters for specified fibres

G. ALLAN*, R. YANG, A. FOTHERINGHAM, R. MATHER
School of Textiles, Heriot-Watt University, Scottish Borders Campus, Galashiels, Scotland, TD1 3HF UK

This paper describes the application of artificial intelligence to data derived from polypropylene drawing carried out at Galashiels using designed experiments. The topology of the data is visualised in two dimensions with respect to specific properties to be modelled, as a quality check on the process data. A series of neural network models are used successfully to predict the tenacity, elongation, modulus and heat shrinkage and also the crystallographic order and polymer chains orientation of the output fibres from the draw parameters values. A software harness is constructed for using the neural predictors to find the draw parameters which come closest to achieving any specified combination of fibre properties. © *2001 Kluwer Academic Publishers*

1. Introduction
The building of models or simulations of industrial processes offers many benefits. These include: forecasting the outputs from untried combinations of parameters values, unlimited experimentation and searching at no cost, process improvements through studying the model, minimised downtime for trials, training for operatives, and investigation of hazardous events.

In recent years the availability of fast personal computers has provided software tools which can enable researchers to build computer simulations of the relationships observed in sets of observations, by adaptation to the data. These types of computer simulations are called Neural Networks (from their copying of learning and recall processes in the brain cortex), and they learn from experience, like children. The internal functions which result from "training" neural networks may well be too complex to capture by mathematical models or express as rules; nevertheless, if neural networks are trained on examples which are a good statistical representation of the parameter space and are also optimised in their own design, the resulting models of processes are likely to be able to provide all the model features which are presented above.

Two types of neural network were used in this study: the Kohonen Self-organising Map (or SOM) and the Multilayer Perceptron (or MLP). Both these neural network paradigms are to an extent biological metaphors of the brain cortex, since in the software medium they consist of multiple nodes or "neurons" interconnected by "axons" or links of adjustable strengths or weights. The SOM is trained to build an internal representation of the data presented to it through a process of learning which is unsupervised, in that no evaluations (or targets) are supplied by the user, and the SOM develops its own interpretation of the observed data space. The SOM is a practical tool, because it can express the data in a two-dimensional (2D) map regardless of the original number of parameters. The arrangement of the data examples (observations) in the map will indicate any natural clusters and/or outliers which may help or hinder a successful simulation of the process represented by the data. As regards the MLP, this is a feed-forward network originally-derived from studies of the human visual system. It learns through repeated iterations to map observed patterns or vectors to their associated targets, which can be evaluations of the patterns, or arbitrary classifications or labels for them. Both of these neural network paradigms are explained in detail in [1–5], but the basic architectures and algorithms are presented below in Section 2.

Yang, Mather and Fotheringham [6] carried out a multivariate analysis of the parameters for drawing of melt-spun polypropylene and the nature of the resultant fibres. The experimental design used to set up the trials in [6] by its nature set out to study the effects of all the parameters of the data space, providing information about the covariance but using the minimum number of trials. The data set was therefore well-suited to constructing effective adaptive models, that is, training a Kohonen map and MLP networks, and the modelling as well as a methodology for optimisation of the draw process will be described in this paper.

*Author to whom all correspondence should be addressed.

2. Kohonen networks and multilayer perceptron networks

The Kohonen network consists of a single layer of neurons, each connected to all of the parameter inputs from the vectors or patterns input as training data. Before training, weight values are allocated to all of these connections, and set initially to random values between 0 and 1. As each vector or pattern of observation values is presented to the layer of neurons, the weights vectors of all the neurons are assessed for competitive activation a_j, as measured by the Euclidean distance from the input vector. The winning neuron is the one closest to the input vector, that is, whose a_j is smallest:

$$a_j = \sqrt{\sum_{i=0}^{n}(w_i - i_i)^2} \tag{1}$$

Where n is the number of weights per neuron, w_i is each of the weights, and i_i is the corresponding value of the input vector.

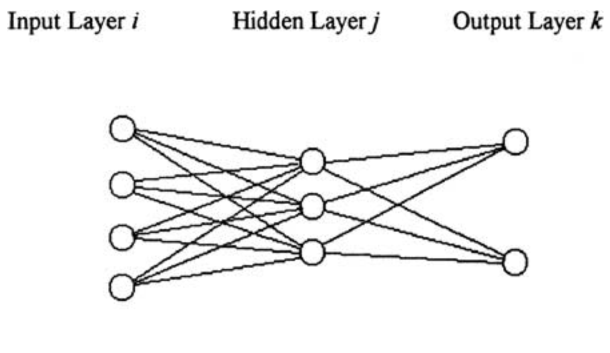
Once the winning neuron is found for the current input vector, the weights of the neuron are adjusted to bring the the neuron's weights vector closer to the inputs vector using (2) below:

$$\delta w_i = -\eta(w_i - i_i), \tag{2}$$

where δw_i is the change in the weight i, η is a learning rate control factor such that $0 < \eta < 1$, w_i is the current weight i, and i_i is the normalised parameter i of the input vector.

Around the winning neuron is a neighbourhood of connected neurons whose weights vectors are all adjusted by δw. As the network in training mode is presented with each of the input vectors in turn, the learning rate and neighbourhood size are slowly decreased. The effect of these controls is hopefully to recognise fine distinctions in the data and damp down oscillations between close clusters. Iterating this process many times through the training vectors set eventually results in convergence, and no further changes will be seen in the topographical visualisation of the neurons layer. In the neural software development environment [7] used to implement the models the layer of Kohonen neurons can be displayed graphically during the learning, and shows the referenced vectors of the training data allocated to layer neurons according to their natural clusters, or similarities. The trained state of the Kohonen neurons can thus be graphically represented as a 2D topographical map or SOM (self-organising map) of the natural clusters of patterns found in the original data. Although the Kohonen SOM learns without supervision (that is, without target evaluations of the input vectors), the map of positioned references to the inputs can be labelled according to a variety of ways of evaluating the inputs. In this way a single topographical map of the dataspace can spawn relational maps of a range of properties attributable to the data vectors. This can indicate the *quality* of the dataspace as a basis for adaptive modelling, since the natural clusters and boundaries for each property may suggest linearity (where a line can separate vectors with high values for the property from vectors with low ones), non-linearity (where more than one boundary would be needed), or chaos (with no clusters for values of this property).

The multilayer perceptron or MLP is also modelled on the brain biology with its neurons and multiple connections, but learns by means of directional processes involving several layers of neurons, summations, and thresholding of signals using an activation function, an observed feature of brain activity. Consider an MLP with 3 layers of neurons. In this nomenclature let $w_{pq,k}$ be the weight of the connection from neuron p of layer j to neuron q of layer k.



Multilayer Perceptron

Using the above convention, $w_{pq,k}$ would be the weight of the connection from neuron p in layer j into node q of layer k. The network learns by calculating an output at the nodes of layer k in response to the input vector at layer i (the Forward Pass), then calculating from the difference between the target output and the actual output an adjustment to the weights of the connections into the hidden layer j and the output layer k so that an error or cost function will be reduced. The process of "back-propagating" the error values through the weights into the i and j layers is called the Reverse Pass. The two mechanisms will now be explained in more detail.

2.1. The forward pass in MLP learning

Suppose an input vector ξ is applied to layer i. The signals to each of the layer j neurons will be the products of the connection weights into these neurons and the values of the ξ signals into these connections. These products are now summed, then "squashed" to the range 0 to 1 by activation function F. This is normally the sigmoid or logistic function as shown in (3) below.

$$F(\alpha_{p,j}) = \frac{1}{1 + e^{-\alpha_{p,j}}} \tag{3}$$

where $\alpha_{p,j}$ is the activation at neuron p of layer j. This function of the output has two roles: it keeps the summed signals within reasonable bounds, and its derivative is a function of itself, which is useful in the Reverse Pass described below. Since the outputs of layer j are the inputs to layer k, the Forward Pass is completed by repeating the above multiplications, summations and squashings to find the activations at layer k, the network outputs for the input ξ.

2.2. The reverse pass in MLP learning

To illustrate this we will look at the process of training the weight from neuron p of hidden layer j into neuron q_k of the output layer. The error δ is based on the difference between the target value and the network output (OUT) at q_k, but multiplied by the derivative of the sigmoid squashing function (3, above) which gives gradient descent of the error:

The derivative

$$\begin{aligned} F'(\alpha_{p,j}) &= F(\alpha_{p,j})(1 - F(\alpha_{p,j})) \\ &= OUT(1 - OUT) \end{aligned}$$

Then

$$\delta = OUT(1 - OUT)(Target - OUT) \tag{4}$$

The adjustment to the weight is found by multiplying δ by η, a learning rate control factor where ($0 < \eta < 1$), then by the input signal, which would be the OUT activation of the neuron in the preceding layer j. This process is followed for all the weights from the hidden layer into the output layer. An illustration would be:

$$\Delta w_{pq,k} = \eta \delta_{q,k} OUT_{p,j} \tag{5}$$

where $\Delta w_{pq,k}$ is the weight adjustment for the connection from neuron p into neuron q of layer k, $\delta_{q,k}$ is the result from (4) above relating to the error at neuron q of output layer k, and $OUT_{p,j}$ is the output activation from neuron p of the hidden layer j. Then:

$$w_{pq,k}(n+1) = w_{pq,k}(n) + \Delta w_{pq,k} \tag{6}$$

where $w_{pq,k}(n+1)$ is the adjusted weight from neuron p to neuron q of layer k at step $(n+1)$, and $w_{pq,k}(n)$ is the corresponding weight at step n, before adjustment.

As regards the weights from the input layer i to the hidden layer j, the error δ cannot be calculated from a comparison of net output with targets at layer j, since these are not available. To solve this problem and allow the errors to be back-propagated through the network to the ij connections, a δ value is calculated for each hidden layer neuron by summing all the products of the output layer δ values and adjusted weights into this neuron then multiplying by the derivative of the squashing function, as illustrated below in (7):

$$\delta_{p,j} = OUT_{p,j}(1 - OUT_{p,j}) \left(\sum_q \delta_{q,k} w_{pq,k} \right) \tag{7}$$

where $\delta_{p,j}$ is the back-propagated error at neuron p of hidden layer j, $OUT_{p,j}$ is the output activation at neuron p of layer j, $\delta_{q,k}$ is the error at neuron q of output layer k, and $w_{pq,k}$ is the weight from neuron p into neuron q of layer k, already adjusted by $\delta_{q,k}$ as in (5) and (6) above. The forward/reverse passes of this learning by back-propagation of error would be continued until the required overall error (normally root mean square or RMS) is reached, or in the case of training failure, a higher RMS error. The nature of the back-propagation algorithm enables an MLP to solve any linear or non-linear problem (that is, of mapping inputs to correct outputs) as long as there are at least two stages of sigmoid neurons and the learning rate is kept low.

3. The experimental method

The parameters values for the 32 trials are shown below in Table I. The 9 process parameters of the drawing trials consisted of rollers' and plates' temperatures and rollers' speeds, and were varied between 2 levels according to a factorial design [8, 9]. The 32 output fibres were measured for mechanical properties, and also for shrinkage under heat treatment, crystallographic order and chains orientation, and the results are shown in Table II. The series of 32 polypropylene fibres were all melt-spun using a fixed, commercially-typical set of parameter values:

Spinneret hole size	0.4 mm
Melt flow index	35 g/10 min
Spinning temperature	230°C
Metering pump speed	12 rpm
Air quench velocity	30%
Spin finish speed	0.5 rpm
Godet 1 speed	100 m/min

The Kohonen network used input from all 9 experimental variables, and learned the natural topology of the data , placing the 32 fibres (represented as 32 vectors each of 9 values) onto a 2D map. The Figs 1–3 show the topographical map shade-coded for modulus, tenacity, and elongation respectively. The analysis indicates the locations in the data space (that is, clusters of similar patterns of draw parameters) where particular property values or their combinations exist. For example, in Fig. 1 (tenacity) fibre 29 had the highest tenacity but was shown to have lower values in the maps of Fig. 2 (modulus) and Fig. 3 (elongation). Fibres 31 and 32, however, were shown to be more consistent in mechanical properties, with very high values of both tenacity and modulus. The data analysis provided by training the SOM revealed whether particular property combinations were available from the value

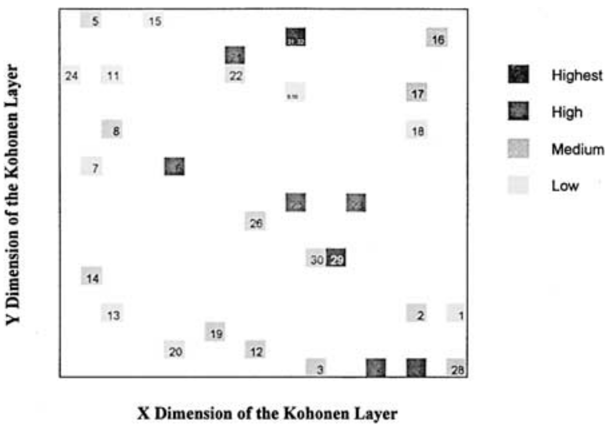


Figure 1 The tenacity SOM.

TABLE I The draw trials data

Ref	Temp Roller1 Deg C	Temp Roller2 Deg C	Temp Roller3 Deg C	Temp Plate1 Deg C	Temp Plate2 Deg C	Temp Plate3 Deg C	Speed Roller2 RPM	Speed Roller3 RPM	Speed Roller4 RPM
1	40	120	125	40	120	120	300	500	580
2	40	120	125	40	120	120	300	500	650
3	40	120	125	40	140	140	400	600	580
4	40	120	125	40	140	140	400	600	650
5	40	120	140	80	120	120	400	600	580
6	40	120	140	80	120	120	400	600	650
7	40	120	140	80	140	140	300	500	580
8	40	120	140	80	140	140	300	500	650
9	10	140	125	80	120	140	300	600	580
10	40	140	125	80	120	140	300	600	650
11	40	140	125	80	140	120	400	500	580
12	40	140	125	80	140	120	400	500	650
13	40	140	140	40	120	140	400	500	580
14	40	140	140	40	120	140	400	500	650
15	40	140	140	40	140	120	300	600	580
16	40	140	140	40	140	120	300	600	650
17	80	120	125	80	120	140	400	500	650
18	80	120	125	80	120	140	400	500	580
19	80	120	125	80	140	120	300	600	650
20	80	120	125	80	140	120	300	600	580
21	80	120	140	40	120	140	300	600	650
22	80	120	140	40	120	140	300	600	580
23	80	120	140	40	140	120	400	500	650
24	80	120	140	40	140	120	400	500	580
25	80	140	125	40	120	120	400	600	650
26	80	140	125	40	120	120	400	600	580
27	80	140	125	40	140	140	300	500	650
28	80	140	125	40	140	140	300	500	580
29	80	140	140	80	120	120	300	500	650
30	80	140	140	80	120	120	300	500	580
31	80	140	140	80	140	140	400	600	650
32	80	140	140	80	140	140	400	600	580

TABLE II The output fibres properties

Ref	Tenacity cN/tex	Elongation %	Modulus cN/tex	Cryst fp (W 1/2) − 1	crys birf delta $n \times 1000$	Shrinkage at 130 deg C
1	40.5	92	200	0.92	31.6	19.1
2	48.3	55	238	0.90	32.2	18.4
3	47.8	79	212	0.80	31.4	16.6
4	50	57	242	0.82	33.0	19.4
5	45.7	77	187	0.85	31.5	15.2
6	50.4	63	243	0.82	33.1	19.1
7	40.4	68	205	0.90	31.8	16.0
8	47.8	63	234	0.93	32.6	17.9
9	43	70	209	0.99	31.3	14.2
10	46.6	59	217	0.96	31.8	16.3
11	41.6	86	199	0.85	31.5	15.6
12	47.2	63	199	0.85	31.9	18.1
13	40.4	81	181	0.90	31.5	15.0
14	48.2	66	228	0.87	32.1	16.3
15	44.5	49	180	1.10	31.4	11.1
16	47.7	54	228	0.98	32.0	11.5
17	48.2	59	232	0.87	32.2	19.1
18	41.1	83	205	0.90	31.9	16.6
19	46.7	66	221	0.97	32.6	16.7
20	44.8	60	188	1.00	31.6	14.7
21	50.4	41	218	0.97	32.8	17.7
22	45.2	80	181	0.97	31.7	13.4
23	50.2	49	225	0.88	32.5	12.5
24	43	71	199	0.92	31.9	12.5
25	52.6	40	228	0.95	33.4	10.3
26	46.9	64	210	0.98	32.0	9.3
27	53.1	40	234	0.96	33.1	9.3
28	45.1	70	231	0.99	32.2	8.5
29	53.3	43	226	1.00	33.0	8.6
30	45	64	208	1.00	32.3	8.4
31	53.3	42	259	0.96	33.9	9.4
32	53	47	232	0.95	32.1	8.0

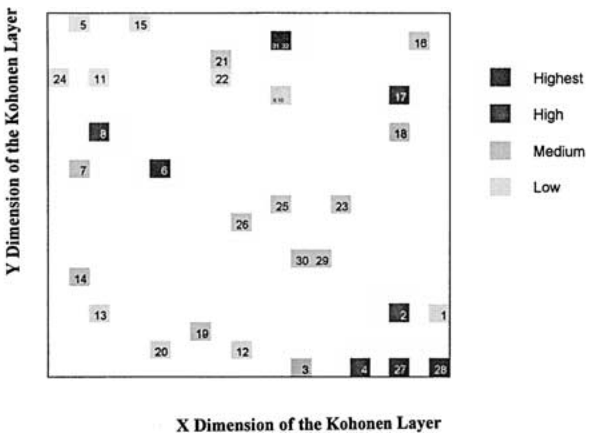


Figure 2 The modulus SOM.

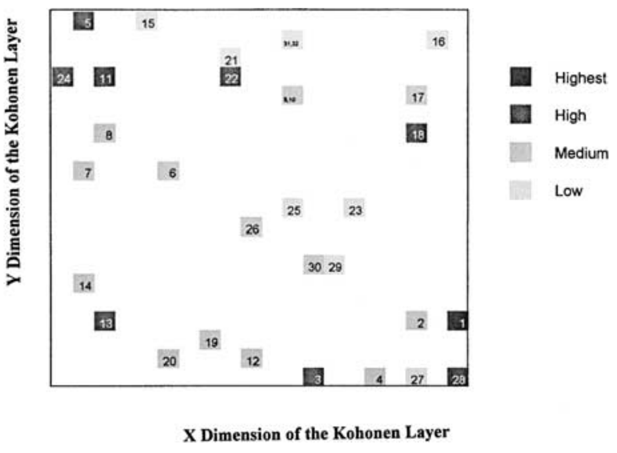


Figure 3 The elongation SOM.

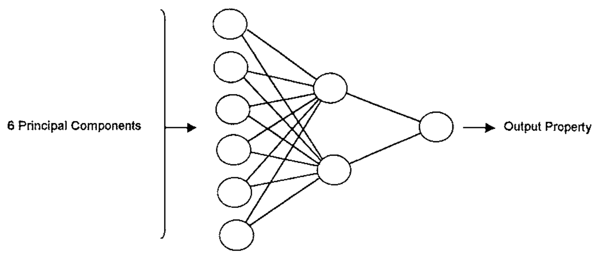


Figure 4 Multilayer perceptron.

ranges of the process parameters. The analysis showed that a user requiring, say, high values for all three mechanical properties would have to consider conducting 32 more of the designed trials, this time with larger ranges of values for the draw parameters, to expand the data space while maintaining its multivariate quality. The Kohohen analysis also served to confirm that the draw data fell naturally into topological areas corresponding to values of fibre properties: the Figs 1–3 show clear clusters of low/medium results, and defined locations for high values. The non-linearity of the prediction problem is also evident, for example high modulus appears in several clusters separated by areas of low modulus, and similar but different clustering is evident for elongation. This quality check suggested that the the trials data would provide enough information to enable successful modelling of the draw process, but that there were non-linearities, justifying the use of an MLP (as explained in Section 2.2, above).

In modelling, care must be taken to ensure that the degrees of freedom in the model are fully taken up by the size and complexity of the data space sampled. Spare capacity in a model reduces its ability to generalise beyond its training examples. The problem of overfitting data by large neural networks is fully explained elsewhere, such as [3, 4]. Principal components analysis [3] is a statistical tool used to reduce data dimensions in problems where there is a degree of linear correlation between the input variables. As regards the polypropylene draw data, it was found that 70% of the parameters variance could be represented by 6 principal components. Replacing the 9 parameters by their principal components reduced the input layer of the neural nets from 9 to 6, reducing the size (number of connections) of the networks and encouraging the development of draw models which would curve-fit rather than point-fit.

A series of MLP's were successfully trained to output the range of fibre properties, using as inputs the 6 principal components of the draw data. A commercial neural software tool [7] was used to implement our designs for the networks, and for all the target properties the models were found to be capable of learning the trials data quite easily, as illustrated by the performance (see Fig. 5) of the neural predictor for shrinkage at 130 °C. The ease of learning was expected, in view of the boundaries and clusterings evident in the SOM analysis for shrinkage and the capability of MLP networks to solve non-linear problems. The configuration of most of the neural networks is shown in Fig. 4. The high correlation (R^2 of 0.963 and slope 0.965) between the forecast and observed shrinkage in Fig. 5 shows the ability of the trained MLP to relate the shrinkage property to the patterns of draw parameters for the 32 fibres.

Each net was optimised for ability to generalise from the sample (that is, the 32 trials) to the global population (that is, not overfit) by cross validation based on holding out random pairs of examples from the training: in each pair, one example would be used for validation and the other for a blind test. The accuracy of the models in the blind tests and the optimised designs and weights states were as shown below. In the table the network configuration is shown as layer sizes for the input nodes, hidden layer nodes, and output nodes. The root mean squared

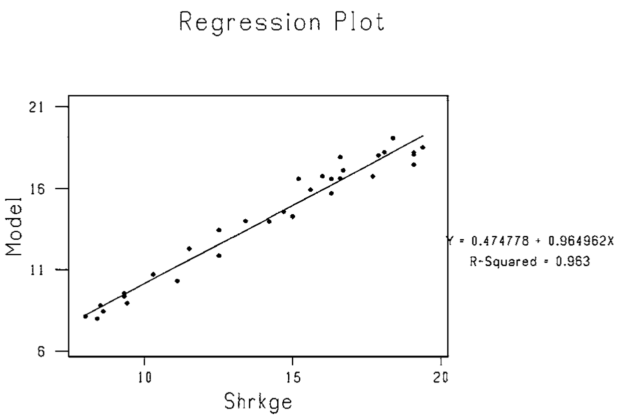


Figure 5 An example of the modelling achieved.

errors show the optimal extent of training (mean distance between the model output and target output) for each model, on the neural software scale of 0.1 to 0.9:

Prediction	Configuration	%Accuracy	RMS Error
Tenacity	{6, 2, 1}	98.2	0.06
Elongation	{6, 2, 1}	89.4	0.09
Modulus	{6, 3, 1}	91.1	0.06
Shrinkage 130	{6, 2, 1}	91.9	0.06
Cryst. Order	{6, 2, 1}	96.9	0.03
Orientation	{6, 2, 1}	98.6	0.05

Although all the models performed quite accurately, some properties were found to be easier to model than others. A possible reason for this may be that the target values for some properties such as elongation showed more variation than those for other properties such as tenacity. Extra variation would make it more difficult for the neural model to build a trend function for all the training points. Another important factor would be the natural clustering, and the SOM for tenacity (Fig. 1) displayed better separation than the SOM for elongation (Fig. 3).

4. A software harness

Once trained, a neural model is defined mainly as the set of weights for its inter-node (synaptic) connections. These and other data items specifying the 7 neural networks were embedded in software written in-house at the School of Textiles. This program was created separately from the neural nets development system used to build and test the models, and does not require its users to have experience of artificial neural systems. The software behaves as a search engine, simulating thousands of polypropylene fibre draw experiments throughout the data space for which the contributing neural networks have been trained.

A fitness function F was defined, so that the search engine could optimise for whatever fibre properties the user has specified:

$$F = \left(\sum_{i=1}^{P} \left[1 - \left(\left| p_i^T - p_i^M \right| / p_i^T \right) \right] \right) \Big/ P$$

where P is the number of property values specified in the search, p_i^T is the target value of property i, and p_I^M is the model output for property I.

Since the error distance used for F is relative, the fitness is independent of the property units, and the division by P expresses the fitness as a mean success over all the property values specified. The system was found to be capable of delivering optimal solutions for producers: for example, the set of 9 draw parameter values would be found which would best produce a fibre with a low modulus but high tenacity, or a fibre with low shrinkage when heated. Since the crystallinity models were also embedded, the production specification report included information on the order and orientation of the polymer which would be produced.

5. Conclusions

This paper shows that combining experimental design and neural networks modelling can help polypropylene fibre spinners predict the nature of the outputs from the drawing of spun fibres. It was found that although network training relied on a small set of examples and parameters were varied using only 2 levels, the balance of the experiments and interpolative ability of the neural models allowed models to be developed which were able to generalise quite well to examples outwith their training.

By the use of search software, the parameters can be found to provide the products specified by customers, so the fibre producer is put in a position to optimise production without risking excessive downtime for trials.

There are many other current and potential applications for this approach in textiles processing, and few reasons why it should not be taken up by those interested in competitiveness and flexible production through optimisation.

References

1. D. RUMELHART, G. HINTON and R. WILLIAMS, Learning internal representations by error propagation, in "Parallel Distributed Processing: Explorations in the Microstructure of Cognition" (Cambridge, MA, MIT, 1986).
2. J. ZUPAN and J. GASTEIGER, Neural Networks for Chemists (VCH Publications) (1993).
3. C. BISHOP, Neural Networks for Pattern Recognition (Oxford University Press, 1995).
4. J. HERTZ, A. KROGH and R. G. PALMER, Introduction to the Theory of Neural Computation (Addison Wesley, NY, 1991).
5. T. KOHONEN, Self-organised formation of topographically correct feature maps, *Biological Cybernetics* **43** (1998) 56. Reprinted in Anderson and Rosenfield.
6. R. D. YANG, R. R. MATHER and A. F. FOTHERINGHAM, 2000, unpublished data.
7. NeuFrame (Neusciences, unit 2, Lulworth Business Centre, Nutwood Way, Totton, Southampton SO40 3WW, UK).
8. N. LOGOTHETIS and H. P. WYNN, "Quality Through Design" (Clarendon Press, Oxford, 1989).
9. R. A. FISHER, "Design of Experiments" (Oliver and Boyd, Edinburgh, 1966).
10. E. BAUM and D. HAUSSLER, What size net gives valid generalisation? *Neural Computation* **1** MIT (1989).

Received 10 August
and accepted 19 December 2000