Artificial Intelligence Techniques in Simulation of Viscoplasticity of Polymeric Composites

M.S. Al-Haik,1 M.Y. Hussaini,2 C.S. Rogan3

1Department of Mechanical Engineering, University of New Mexico, Albuquerque, New Mexico 87131
2School of Computational Science and Information Technology, Florida State University, Tallahassee, Florida 32306
3Department of Physics, Princeton University, Princeton, New Jersey 08544

The viscoplastic behavior of a carbon fiber/polymer matrix composite is investigated via different modeling schemes. The first model is phenomenological in nature based on the overstress-viscoplasticity. The second model utilizes neural networks paradigms. Genetic algorithm-based strategies are used to prune the proposed neural network. Several optimization algorithms are implemented for training the network. In comparison, the neurocomputational model is found to outperform the phenomenological model. POLYM. COMPOS., 30:1701–1708, 2009. © 2008 Society of Plastics Engineers

INTRODUCTION

Fiber-reinforced polymer-matrix composites (PMCs) exhibit both time- and rate-dependent nonlinear behavior under uniaxial loading. This is mainly due to the dependence of the mechanical properties of polymer matrices on both time and rate over a wide range of temperatures [1–3]. Structural components made of PMCs in aerospace, automotive, and civil engineering applications require an assessment of the durability of these engineering materials. PMCs durability must be evaluated by analyses [4]. Therefore, it is a prerequisite to develop a constitutive model for describing the time-, rate-, and temperature-dependent behavior of PMCs.

PMCs can become brittle, viscoelastic, or viscoplastic, depending on the loading conditions they are subjected to. Viscoelastic or viscoplastic behavior manifests itself in various ways, including creep under constant load, stress relaxation under constant deformation, time-dependent recovery of deformation following load removal, and time-dependent creep rupture [5]. In cases of relatively small stress (or strain), the time-dependent behavior of the composite is described by linear and nonlinear viscoelastic modeling of polymer matrices. When the composite is loaded to a high stress (or strain), the time-dependent behavior of polymers tends to deviate from the predictions of viscoelastic models. This is partly because loading of solid polymers to high stress or high strain levels results in yielding [6] and time-dependent inelastic deformation [7]. To describe the time-dependent inelastic behavior of PMCs, phenomenological viscoplasticity models were introduced [2].

The deformation of polymer-matrix materials strongly depends on the duration and the rate of loading, and this dependence becomes even more critical as the temperature approaches the glass transition temperature. For appropriate use of these materials, there is a need for the development of comprehensive material property characterization techniques and analytical modeling methods to predict their long-term mechanical response at elevated temperatures, and there has been an extensive research effort to that end [8].

Viscoplastic constitutive equations are written explicitly and they involve many parameters, which significantly influence the behavior of the constitutive equations. Several viscoplastic models were presented [2, 7–9]. Throughout these models, appropriate parameters must be determined accordingly, so the accurate behaviors of the material can be expressed. The parameter identification problem for the viscoplastic model was investigated by Mahnken and Stein [10]. Once the model is set, the behavior of the material can only be expressed approximately by adjusting the parameters in the model. Under conditions of high stress, strain rate, and temperature, the model has to become more complicated in its mathematical formulation to secure more accurate results. However, as shown by Gates and Sun [11], the problem of parameter identification of the viscoplastic model is not straightforward due to the nature of the constitutive equations.
identification of the material parameters will introduce more numerical errors and instabilities to the model.

As an alternative to traditional explicit constitutive modeling, several investigators recently suggested what might be called implicit constitutive modeling based on artificial neural networks (ANN). AN can directly map the behavior of a viscoplastic material [12–18]. An artificial neural network is capable of making decisions based on incomplete, noisy, and disordered information, and it can also generalize rules from those cases on which it was trained and apply these rules to new stimuli.

In an earlier investigation, Al-Haik et al. [14, 15] constructed a three-layer feed forward ANN to capture the creep behavior of polymeric composite, carbon fiber reinforced epoxy. The inputs were the loading level, temperature and time, and the output was the creep strain. This model was trained by data from experiments using steepest descent, conjugate gradient, scaled conjugate gradient, and truncated Newton minimization algorithms. The difficulty in constructing these ANN implicit viscoplastic models stems from the uncertainty in constructing the neural network itself; i.e., the number of layers and the number of neurons in each layer.

The objective of this work is to construct an ANN model to predict the stress relaxation behavior of a polymer-matrix composite, and to compare and contrast the results. The current work implements a state-of-the-art genetic algorithm to construct the neural networks and utilizes several training algorithms to train the neural network to predict the stress relaxation behavior of the composite based on learning different load relaxation tests performed at different temperature/strain levels.

Comparison of the ANN model results to those of the phenomenological approach is provided to test the ANN predictive capability for inelastic phenomena. To perform these tasks and to develop the framework for the modeling effort, simplistic unidirectional loading, rather than the more sophisticated anisotropic treatment, is used for the experimental result. The extension to anisotropic behavior may be easily incorporated into the present formulation, but it is not the focus of the present work.

**VISCOPLASTIC MODEL**

Gates and Sun developed a model to predict rate dependent behaviors for several types of PMCs [11]. This model requires five material parameters, obtained by performing load relaxation tests, to describe the viscoplastic behavior. The goal of the model is to use a potential function that accounts for material anisotropy to characterize the yield surface. Trials of the model have shown favorable results and adequate correlation between short-term creep and the load relaxation tests. Gates’s model is presented below. The total strain for the time dependent constitutive relation may be written as a combination of the elastic and plastic strain.

\[
e^t = e^e + e^p
\]

Hooke’s law holds for the material’s elastic strain \(e^e\) and stress. The plastic strain must be represented with a power law,

\[
e^p = A(\sigma)^n
\]

where \(A\) and \(n\) are material parameters at a given temperature drawn from tensile test experimental data. Details of the tensile test can be found in an earlier work by the authors [15]. The rate dependent constitutive relation is displayed in a similar manner, separating the elastic and viscoplastic components:

\[
e^t = e^e + e^{vp}
\]

Again the elastic strain rate is derived from Hooke’s law

\[
e^e = \frac{\dot{\epsilon}}{E}
\]

But the viscoplastic strain rate must be divided into the plastic and overstressed viscoplastic portions of the term,

\[
e^{vp} = e^p + e^{vp'}
\]

A simple differentiation of the plastic strain given above displays the first term for the viscoplastic strain rate.

\[
e^p = \begin{cases} A n (\sigma)^{n-1} & \text{for } \sigma > 0 \\ 0 & \text{for } \sigma \leq 0 \end{cases}
\]

\[
e^{vp'} = \begin{cases} \left(\frac{\sigma - \sigma^*}{K}\right)^{1/n} & \text{for } \sigma > \sigma^* \\ 0 & \text{for } \sigma \leq \sigma^* \end{cases}
\]

where \((\sigma - \sigma^*)\) is the overstress, \(<\sigma > \) are Macaulay brackets, and \(K\) and \(m\) are material constants found from stress relaxation experimental data. The overstress is considered as a scalar quantity that relates the quasi-static stress, \(\sigma^*\), to the dynamic or instantaneous stress, \(\sigma\), at the same strain level [19]. The quasi-static stress, \(\sigma^*\), is found through a common elastoplastic relation given by,

\[
e = \frac{\sigma^*}{E} + A(\sigma^*)^n
\]

while the dynamic stress is the stress resulting from the time-dependent material behavior.

**NEURAL NETWORKS VISCOPLASTIC MODEL**

A multilayer feed-forward neural network model is proposed for predicting the stress relaxation of a polymer-matrix carbon fiber composite material. It consists of a number of simple neuron-like processing elements (PEs), also called neurons or nodes, and is organized in layers that are classified as an input layer, a hidden layer, and
an output layer. Every unit within a layer is connected with all of the units in the previous layer. These connections are not all equal; each connection has a different strength or weight. The weights of these connections encode the knowledge of the network. Data enters at the input layer and passes through the network, layer by layer, until it arrives at the output layer. During normal operation, there is no feedback between layers, and hence it is called a feed-forward neural network.

**Determination of Network Topology/Architecture**

The first task is to determine the neural network topology, which includes the number of hidden layers, the number of PEs in each layer, and the connections between them. The number of neurons in the input and output layers is governed by the dimensionality of the problem. In this case, there are three input quantities (temperature $T$, normalized stress $\sigma_0$, and time $t$) and one output quantity (relaxation stress $\sigma(t)$). As such, three neurons suffice for the input layer and one neuron for the output layer. Regarding the number of hidden layers, it has been proven that a neural network requires at most two hidden layers to approximate any function to an arbitrary order of accuracy \[20, 21\]. We choose two hidden layers. It is not known how to determine the number of hidden units (number of PEs in the hidden layers) \textit{a priori}, although arbitrarily high accuracy can be achieved by increasing the number at the cost of “over fitting.” After a critical training iteration, further training of a neural network by backpropagation will continue to improve the results for the training set, but the test set performance will begin to deteriorate. This phenomenon is called overfitting. Basically, the overtrained neural network will attempt to fit a curve to the test set data points, producing a curve with high curvature which fits the test data points well, but does not model the underlying function well, its shape being distorted by the noise inherent in the data. The topology (PEs and connections) of the proposed neural network model is given in Fig. 1.

In this work, genetic algorithms (GAs) are used to optimize the neural network’s size. A synthetic method is used; it is a combination of a constructive algorithm and pruning based on a genetic algorithm. This method was shown to possess high efficiency in dealing with complex dynamic environments \[22\]. The method includes the following steps: a dynamic constructive method is adopted to train the initial networks, the genetic algorithm is used to prune the trained network, and then the global optimal solution can be obtained rapidly due to the good initial solution.

**Preparation of Data**

The data is obtained from stress relaxation experiments conducted at temperatures, $T$, equal to 25, 35, 45, 50, 55, 60, 65, and 75°C with six initial stress levels, $\sigma_0$, 30, 40, 50, 60, 70, and 80% of the composite strength at the corresponding temperature, and stress relaxation, $\sigma(t)$, measurements taken at 250 instants with two-second intervals. Details of the experiments can be found in our previous work \[14, 15\]. Thus, the total data set consists of $8 \times 6 \times 250 = 12,000$ quadruple vectors $(T, \sigma_0, t, \sigma(t))$, the first three elements of which are input quantities and the fourth is the output/target quantity. As neural network computations do not favor the “raw” data for training, the input-output data $(T, \sigma_0, t, \sigma(t))$ is transformed by the mapping

$$x_q = 2 \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} - 1,$$

so that their values lie in the interval $(-1, 1)$. Here, $x_q$ is the transformed value of the vector $x = [T, \sigma_0, t, \sigma(t)]$, and $x_{\text{min}}, x_{\text{max}}$ are the minimum and maximum values in the database for the vector $x$. The 12,000 transformed input–output vectors are split into three subsets—the first subset is used for training, the second is used for validation, and the third is used for testing the network performance. The transformed data subsets are randomized to eliminate any bias that might exist in the training data set and to ensure that the training process of the network does not become a table look-up problem.

The transformed data subsets are represented as a pair of input and output matrices whose indices are randomized using uniform distribution on the interval $(0, 1)$ keeping the relationship between the elements of input and output matrices fixed.

The training and validation sets consist of 9,000 vectors (3/4 of the entire data set) that cover the temperatures 25, 45, 50, 55, 60, and 75°C. Of these 9,000 vectors, 6,000 are chosen for training and 3,000 for validation.
The remaining 3,000 vectors that cover the stress relaxation tests at temperatures of 35 and 65°C are used for testing the ANN prediction of stress relaxation once the network has been trained and validated.

The objective of the training procedure is to find a set of possible weights, \( w_{ij} \) that will enable the network to produce a prediction \( (b_i) \) as similar as possible to the known output \( (t_i) \). This is achieved by minimizing the cost function \( E \), which is the mean square error:

\[
E = \frac{1}{2q} \sum_{i} (t_i - b_i)^2
\]

where \( q \) is the number of the training patterns.

**Pruning Neural Networks Using Genetics Algorithm**

Genetic algorithms (GAs) are a class of evolutionary algorithms that use population-based stochastic search processes that originated from principles of natural evolution.

The important feature of GAs is their population-based strategy. The individuals in a population compute and exchange information with each other to fulfill certain tasks. The general functionality of GAs can be described as follows:

1. Generate the initial population \( X(0) = (x_1(0), x_2(0), \ldots, x_N(0)) \) randomly and set \( t = 0 \); Determine some parameters such as \( N, L, P_c, P_m \), where \( N \) is the population size, \( L \) is the length of a chromosome, \( P_c \) and \( P_m \) are the probability of crossover and mutation operators;
2. Repeat
   - Compute the fitness of each individual \( f(x_t(t)) \) in the population;
     - Select parents from \( X(t) \) randomly, \( X'_j(t), j = 1, 2, \ldots, 2m \);
     - Apply crossover and mutation operators to parents and produce offspring which form \( X(t + 1) \);
     - \( t = t + 1 \);
3. Until “termination criterion” is satisfied.

In GAs, some search operators are adopted. The mutation operator can introduce new genes to regain the diversity of individuals. The crossover operator can recombine existing information to find better individuals. The selection operator acts as a guide, which can lead the search towards the best possible district in searching space. The operators can help to obtain the global best solution.

Pruning the network means removing unimportant neurons and/or connections. In this work, GAs are adopted to prune the trained network with satisfied training error. There is no need to specify the threshold or adjust the remaining weights with this kind of network.

Assume \( m \) and \( n \) are the dimensions of the input and output vectors and \( N \) is the number of the hidden neurons. The neurons and links are encoded together. The neurons are encoded at the head and the links are encoded at the tail. Only neurons in the hidden layers need to be encoded. Neurons in the first layer are encoded first, then the nodes in the second layer are encoded. The connecting weights are arranged in sequence. The links between the input layer and the first hidden layer, the second hidden layer, and the output layer are encoded first. The encoding of links between the first hidden layer and the second hidden layer, and the second hidden layer and the output layer are encoded last. The binary strings to encode are used, while these correspond to each neuron or link. If a neuron or link exists, the value is 1; otherwise, the value is 0 If a neurons does not exist, all connections corresponding to it will be removed (weight = 0).

The pruning algorithm process is as follows:

1. Generate the initial population \( X(0) = (x_1(0), x_2(0), \ldots, x_N(0)) \) and set \( t = 0 \); Each \( x_1(0) \) corresponds to an encoded network.
2. Decode each individual in the current generation into a set of neurons and connection weights; construct a corresponding ANN according to the above encoding method.
3. Apply the training data to the network and compute the fitness of each individual; preserve the network with the largest fitness value. In our study the fitness function is defined as,

\[
\text{Fitness} = \frac{zA}{(m + n + N)L}
\]

where \( z \) is a constant and \( L \) is the number of all links. \( A \), defined as \( 1/E \), denotes the average computing precision for the training set.
4. Select parents for reproduction based on their fitness values.
5. Apply uniform crossover and mutation operators to the parents to generate offspring, which form the next generation, and set \( t = t + 1 \).
6. Decide whether the termination criterion is satisfied. When the individuals with the largest fitness in the last five generations are the same, the pruning process will stop, and the network with the largest fitness in the last population will be the optimal solution. Otherwise, return to step 2.

**RESULTS**

**Explicit Viscoplastic Model Results**

The data from the load relaxation tests was used to determine the temperature-dependent material constants \( K \) and \( m \). For each temperature, the load relaxation test was conducted at six different stress levels, as shown in Fig. 2.

First, the data from each strain level at each temperature was isolated. The stress relaxation data was then fit into polynomial functions. From the exponential fits, the constants \( K \) and \( m \) were calculated by plotting...
Log\((-d\sigma/dt)/E\) vs. Log \((\sigma - \sigma^*)\), and calculating the linear fit. The viscoplastic model parameters at each temperature are tabulated together with measured material properties in Table 1. The total strain rate is zero during the load relaxation test, leading to the following differential equation

\[
\frac{1}{E} \left( -\frac{d\sigma}{dt} \right) = \left( \frac{\sigma - \sigma^*}{K} \right)^{1/n}
\]  

(12)

The explicit model solution was generated by solving this differential equation using the fourth order Runge-Kutta method. Different step sizes were experimented with, and an example solution is shown below in Fig. 3.

**Implicit Viscoplastic Model**

**Training ANN.** Training a feed-forward backpropagating neural network consists of giving the network a vectorized training data set each epoch. Each individual vector’s inputs (temperature, strain level, time) are propagated through the network, and the output is incorporated with the vector’s experimental output in the error equation above. Training the network consists of minimizing this error in the space of the weight function, and adjusting the network’s weights using unconstrained local optimization methods. An example of a training session graph is shown in Fig. 4, in this case using a gradient descent method with variable learning rate and momentum terms to minimize the error function.

**Pruning ANN Structure Using Genetic Algorithm.** A genetic algorithm was used to determine optimal network architecture. On the basis of the results of earlier exhaustive methods [15], a domain from 1 to 15 and 1 to 35 was used for the number of neurons in the first and second hidden layers, respectively. A population of random networks in this domain was generated, each network encoded as a binary chromosome. The probability of a particular network’s survival is a linear function of its rank in the population. Stochastic remainder selection without replacement was used for population selection. For crossovers, a two-point crossover of chromosomes’ reduced surrogates was used, as shown in Fig. 5. This method allows pruning not only of neurons but links, as each layer of neurons is not necessarily completely connected to the next, and connections between nonadjacent layers are permitted. The genetic algorithm was run with varying parameter values and two different objective functions: one seeking to minimize only the training performance error of the networks and another minimizing both the performance error and the number of neurons and links. Figure 6a displays an optimal network when only the performance error is considered. Figure 6b shows an optimal network when the number of neurons and links was taken into account. Figure 7 shows the results of an exhaustive architecture search in a smaller domain, the first arrow pointing to a minimum that coincides with the network architecture displayed in Fig. 6a.

**Choosing Training Algorithm.** A network architecture of \((6 \times 25 \times 1)\) was used for training and testing the

<table>
<thead>
<tr>
<th>Temperature (°C)</th>
<th>E (GPa)</th>
<th>(\sigma_u) (MPa)</th>
<th>(n)</th>
<th>(K) (MPa)</th>
<th>(A) (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>81.01</td>
<td>400</td>
<td>0.73</td>
<td>4.10 \times 10^6</td>
<td>3.619</td>
</tr>
<tr>
<td>35</td>
<td>72.04</td>
<td>369</td>
<td>0.69</td>
<td>1.04 \times 10^5</td>
<td>1.166</td>
</tr>
<tr>
<td>45</td>
<td>64.46</td>
<td>333</td>
<td>0.70</td>
<td>9.41 \times 10^4</td>
<td>0.630</td>
</tr>
<tr>
<td>50</td>
<td>62.01</td>
<td>320</td>
<td>0.62</td>
<td>1.16 \times 10^3</td>
<td>1.372</td>
</tr>
<tr>
<td>55</td>
<td>59.25</td>
<td>315</td>
<td>0.67</td>
<td>2.10 \times 10^2</td>
<td>2.705</td>
</tr>
<tr>
<td>60</td>
<td>56.84</td>
<td>296</td>
<td>0.69</td>
<td>1.57 \times 10^1</td>
<td>1.518</td>
</tr>
<tr>
<td>65</td>
<td>49.94</td>
<td>285</td>
<td>0.70</td>
<td>5.80 \times 10^1</td>
<td>1.266</td>
</tr>
<tr>
<td>75</td>
<td>41.45</td>
<td>244</td>
<td>0.66</td>
<td>1.39 \times 10^0</td>
<td>1.793</td>
</tr>
</tbody>
</table>

FIG. 2. Experimental stress relaxation at 45°C. The stress levels correspond to 30, 40, 50, 60, 70, and 80% of the strength at each temperature.

FIG. 3. Comparison between experimental stress relaxation and prediction using the explicit viscoplastic model. The data represents the stress relaxation test at temperature \(T = 45°C\) and 70% of the composite strength at this temperature.

FIG. 4. Example training session graph using a gradient descent method with variable learning rate and momentum terms to minimize the error function.

FIG. 5. Example of a training session graph using a gradient descent method with variable learning rate and momentum terms to minimize the error function.

FIG. 6a. Optimal network when only the performance error is considered.

FIG. 6b. Optimal network when the number of neurons and links was taken into account.

FIG. 7. Results of an exhaustive architecture search in a smaller domain, the first arrow pointing to a minimum that coincides with the network architecture displayed in Fig. 6a.
neural networks. Several different minimization algorithms were tested and compared for the training of the network and are listed in Figs. 8a and b. These two figures display the training performance error and gradient over 1,000 epochs. It is clear that for the pruned ANN with final structure of $(6 \times 25 \times 1)$, the Quasi-Newton (BFGS) algorithm attains the best performance and the fastest convergence rate.

Utilizing the testing and validation data sets, the $(6 \times 25 \times 1)$ ANN will predict the load relaxation for these two sets. Figures 9a-d displays the final performance of both the ANN and the explicit viscoplastic models compared to the experimental data. The Quasi-Newton (BFGS) algorithm was used for the implicit model, as it performed the best.

**DISCUSSION AND CONCLUSION**

The explicit viscoplastic model analysis suggests that there is no uniform trend regarding how the material parameters ($A$, $n$, $m$, and $K$) behave with temperature. This ambiguity suggests that there are other factors affecting those constants, such as the stress levels and, essentially, the reduction in the glass transition range at elevated temperature conditions. The results of the explicit viscoplastic model, Fig. 9, suggest that there is a discrepancy between the experimental and the predicted stress levels at the beginning of the relaxation response. The model overestimates the stress relaxation rate at the beginning. To control the magnitude of the initial rate of stress relaxation, we should refine the evolution equation for the quasi static stress ($\sigma^*$) and the material parameters.

Simulating the stress relaxation behavior of the composite at temperatures $(75^\circ C)$ closer to the glass transition temperature of the composite $(86^\circ C)$, revealed the discrepancy of the explicit viscoplastic model to capture the actual behavior of the composite in the vicinity of the glass transition region. Unlike the explicit viscoplastic model, the implicit model based on ANN-pruned with the genetic algorithm predicted more accurate results under the condition of relatively high temperature.

The neural network model was built directly from the experimental results obtained via stress relaxation tests performed at various stress-temperature conditions. The optimal structure of the neural network was achieved through the universal approximation theory (2 hidden layers ANN can approximate any function) and the
dimensionality of the load relaxation problem (stress level, temperature, and time).

The structure of the ANN was optimized further using genetic algorithm pruning. The dynamic reconstruction of

FIG. 7. The pruned ANN performance. The arrows point to minimums that coincide with the network architecture displayed in Figure 6(a).

FIG. 8. (a) MSE error for training ($6 \times 25 \times 1$) ANN based on different training (minimization) algorithms and (b) the convergence rate of the different training algorithms for the ANN network.

FIG. 9. Validation of the viscoplastic and neural network models (truncated Newton) for (a) the stress relaxation at $T = 75^\circ C$ and stress level of 40%, and (b) for stress relaxation at $T = 75^\circ C$ and stress level of 50% of the composite strength at this temperature.
the ANN structure using GA reduces the network complexity leading to less error and faster convergence. Moreover, the algorithm is simple and easy to accomplish. The performance of the pruned ANN viscoplastic model is represented by the mean square error between the neural network prediction and the experimental stress relaxation results. To minimize this error, several optimization techniques were examined. The minimization of the error with structure \((6 \times 25 \times 1)\) is determined by the Quasi-Newton (BFGS) method. This method outperforms several training algorithms such as steepest descent and conjugate gradient methods in terms of convergence rate and accuracy.

Unlike the explicit viscoplastic model, the neural network model utilizing the Quasi-Newton algorithm predicted more accurate results at different stress-temperature conditions. Moreover, in building the neural network stress relaxation model, only one type of data is required, that is stress relaxation data at different thermomechanical histories, while the viscoplastic model requires both tensile test data together with load relaxation data. The optimization technique utilized for training the neural networks (BFGS) is one that minimizes the function toward finding “local minimums” rather than global ones. Therefore, at successive iterations within the predesigned errors, the training algorithm might move away from direction of a local minimum, error increases, till it hits other local minima, error decreases. The oscillating nature of the prediction in Fig. 9 reflects this fact.

REFERENCES