A Hitchhiker’s guide to the techniques of adaptive nonlinear models

Arnold F. Shapiro*

Smeal College of Business, Penn State University, University Park, PA 16802, USA

Received 1 December 1998; received in revised form 1 November 1999; accepted 24 November 1999

Abstract

Adaptive nonlinear models (ANMs) are currently being proposed for use in actuarial and financial modeling. The techniques of these models included such things as neural networks and genetic algorithms. While there is a general awareness of the nature of these ANM techniques, there is often only vague familiarity with the details of how these techniques are implemented. This article is intended to help alleviate this situation. Its purpose is to present an overview of ANM techniques, which includes an explanation of what they are, how they work, and a description of their key features. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Adaptive; Nonlinear; Techniques; Heuristic

1. Introduction

Adaptive nonlinear models (ANMs) are models of problems where there are important nonlinearities between the observables (independent variables) and the dependent variable, and, because the underlying theory is not known, the situation dictates the use of an adaptive approach based on the observed data. ANM techniques are the techniques upon which these models are built.

Risk and insurance researchers generally have an awareness of the nature of ANM techniques. Most know, e.g., that genetic algorithms are based on genetics and evolution, neural networks are based on how the brain functions, chaos theory is related to the flapping of the wings of a butterfly, and some are even aware that simulated annealing is based on thermodynamics. However, there is often only vague familiarity with the details of how these techniques are implemented.

This dichotomy is unfortunate. Many researchers are confronted with problems where ANM techniques are appropriate. These include problems that require a heuristic solution because of the vagueness of the underlying theory, and situations involving nonlinearities, where there is an emphasis on not making unjustified assumptions about the nature of those nonlinearities. Since ANM techniques have the capacity to overcome these issues, one would expect to see them implemented more often.

A plausible explanation of why ANM techniques are not being implemented more often is that potential users are not sufficiently familiar with their characteristics and, consequently, forego opportunities for implementation. Assuming this to be the case, the purpose of this article is to help alleviate this situation by presenting an overview of ANM techniques, which includes an explanation of what they are, how they work, and a description of their key features.
2. A synopsis of ANM techniques

The ANM techniques discussed in this article are summarized in Fig. 1 and briefly described in the statements that follows:

- Statistical pattern recognition (SPR) (Therrien, 1989) identifies patterns in data and uses those patterns to reach conclusions.
- Neural networks represent an analogy of the human brain and the associated neural complex.
- Genetic algorithms are automated heuristics that optimize by emulating evolution in nature.
- Simulated annealing is a stochastic algorithm that minimizes numerical functions. Its distinguishing feature is that it involves a random process that eludes local minimums.
- Fractal analysis is based on probability density functions that are statistically self-similar in the sense that they retain their characteristics over different increments of time.
- Chaos theory promotes the notions that a deterministic nonlinear dynamic system can produce random-looking results and, conversely, there may be order in random-looking data.
- Data fusion is used to combine data from different sensors to facilitate target recognition.
- Data compression is used to maximize the amount of information contained in the data.
- Rule induction is a system that induces logical rules from historical data, and then applies the rules to make predictions on other given data.
- Fuzzy logic is a superset of conventional logic extended to handle the concept of partial truths.
- Case-based reasoning is an approach to problem solving based on the retrieval and adaptation of cases.

Of course, a given researcher may not be interested in all these ANM techniques. Moreover, some are better known than others, and some are certainly more interesting than others. However, rather than exclude techniques based on these kinds of considerations, they have all been included. The rest of the article is devoted to an overview of each of these techniques.

3. Statistical pattern recognition

Statistical pattern recognition (SPR) (Therrien, 1989) is used to identify patterns in data and to use those patterns to reach conclusions. It includes as sub-disciplines discriminant analysis, feature extraction, error estimation, and cluster analysis. The technique is designed to facilitate data interpretation when confronted with large data sets and many parameters, and, as such, is a preliminary step for many of the techniques that follow.

Generally speaking, the steps of the methodology underlying the SPR approach may be depicted by the flowchart shown in Fig. 2. The steps in this flowchart may be summarized as follows:

- Preliminary data evaluation involves the screening of variables, compatibility checks, and the analysis of the distribution characteristics of each variable.
- Stratification and grouping investigate the stratification of the variables and the natural grouping for each, as well the statistical differences in the groups obtained.
- Intervariable relationship analysis involves the study of sets and pairs of variables, including correlations and regressions.

---

1 Adapted from Gorman (1996), Slide 5.

2 Adapted from Wolff and Parsons (1983).
Unsupervised learning simultaneously analyzes all variables for natural groupings in the data base.

Supervised learning uses a knowledge base to check the validity of the “natural” groupings found in the previous step. Another technique is to train the computer to recognize known category definitions for the data and to study the variables that affect this grouping.

Data reduction analysis looks at the problems inherent in data reduction techniques and how different subgroups of variables in the study define the categories.

Data modification analysis addresses the problems of data modification procedures.

Of course, the foregoing is only one possible approach. A key step in the foregoing is the step involving supervised learning. If there is a priori knowledge of a preferred outcome, the problem becomes one of supervised SPR, the details of which are depicted in Fig. 3.

As indicated, under supervised learning the sample is subdivided into a training set and a testing set. The former is used to create the classification algorithm; the latter is used to test the classification algorithm. Once the optimal classification is attained, as measured in terms of a misclassification rate, the algorithm can be implemented.

Statistical pattern recognition has been used in numerous applications. A computer-based “nearest-neighbor” pattern recognition technique could be used to emulate underwriters who observe applications and compare them with all the ones they have in their knowledge base. The program can be dynamic, in the sense that the criteria for prediction could be altered continuously as the result of learning. This is similar to case-based reasoning and expert systems.

4. Neural networks

Neural networks (NNs) are software programs that emulate the biological structure of the human brain and its associated neural complex and are used for pattern classification, prediction and financial analysis, and control and optimization.

A sketch of the operation of an NN is shown in Fig. 4. The case depicted involves supervised learning, so that both the inputs and output of the system are known, and the objective is to find a relationship between them. The process begins by assigning random weights to the connection between each set of neurons in the network. These weights represent the intensity of the connection between any two neurons and will contain the memory of the network. Given the weights, the intermediate values (a hidden layer) and the output of the system are computed. If the output is optimal, the process is halted; if not, the weights are adjusted and the process is continued until an optimal solution is obtained or an alternate stopping rule is reached.

If the flow of information through the network is from the input to the output, it is known as a feed

3 As discussed in the section on neural networks, the training set itself is often subdivided to include a subset for a stopping rule test.

4 Example of known output include such things as firms that have become insolvent and claims which are fraudulent.
forward network. Since inadequacies in the output are fed back through the network so that the algorithm can be improved, the NN is said to involve back-propagation.

4.1. Neural processing unit

The core of an NN is the neural processing unit, an example of which is shown in Fig. 5. As indicated, the inputs to the neuron, \( x_j \), are multiplied by their respective weights, \( w_j \), and aggregated. The weight \( w_0 \) serves the same function as the intercept in a regression formula.\(^5\) The weighted sum is then passed through an activation function, \( F \), to produce the output of the unit. Typically, the activation function takes the form of the logistic function \( F(z) = \frac{1}{1+e^{-z}} \), as shown in the figure.

4.2. A three-layer neural network

An NN is composed of layers of neurons, an example of which is the three-layer NN depicted in Fig. 6. Extending the notation of the last section, the first layer, the input layer, has three neurons (labeled \( x_{0j} \), \( j=0, 1, 2 \)), the second layer, the hidden processing layer,\(^6\) has three neurons (labeled \( x_{1j} \), \( j=0, 1, 2 \)), and the third layer, the output layer, has one neuron (labeled \( x_{21} \)). There are two inputs \( I_1 \) and \( I_2 \).

The neurons are connected by the weights \( w_{ijk} \), where the subscripts \( i, j, \) and \( k \) refer to the \( i \)th layer, the \( j \)th node of the \( i \)th layer, and the \( k \)th node of the \( (i+1) \)th layer, respectively. Thus, e.g., \( w_{021} \) is the weight connecting node 2 of the input layer (layer 0) to node 1 of the hidden layer (layer 1). It follows that the aggregation in the neural processing associated with the hidden neuron \( x_{11} \) results in \( z = x_{00}w_{001} + x_{01}w_{011} + x_{02}w_{021} \), which is the input to the activation function, \( F \).

The network error in this simple example is \( (T-O)^2 \), where \( T \) is the targeted value and \( O \) is the output in a given iteration through the network.\(^7\)

4.3. Learning rules

The weights of the network serve as its memory. Thus, the network “learns” when its weights are updated, and the general form of this learning can be expressed as

\[
\Delta w_{ijk}(t) = \eta \delta_{ijk}(t) x_{ij},
\]

where \( w_{ijk}(t) \) is the weight during iteration \( t \) and \( \Delta w_{ijk}(t) \) is the adjustment to the weight after the \( t \)th iteration.

The adjustment is done using a learning rule, a common example of which is the Delta rule (Sheperd, 1997, p. 15),\(^8\) given by

\[
\Delta w_{ijk}(t) = \eta \delta_{ijk}(t) x_{ij},
\]


\(^6\) In essence, as observed by Brockett et al. (1994, p. 408), the “hidden” layers in the model are conceptually similar to non-orthogonal latent factors in a factor analysis, providing a mutually dependent summarization of the pertinent commonalities in the input data.

\(^7\) The general form of this version of the network error is the mean square error.

\(^8\) Other types of learning rules are discussed by Vonk et al. (1997, p. 12).
where $\eta$ is the learning rate, which controls the speed of convergence, $\delta_{ij}(t)$ the error signal, and $x_{ij}$ is the value associated with the $j$th node of the $i$th layer. Since the case at hand involves three layers, including a single output layer, the adjustments are $\Delta w_{0jk}(t)$ and $\Delta w_{1j1}(t)$, the values are $x_{0j}$ (the input values) and $x_{1j}$ (the hidden values), and the learning rates are $\delta_{0jk}(t)$ and $\delta_{1j1}(t)$.

### 4.4. Learning strategy of a neural network

The characteristic feature of NNs is their ability to learn. The strategy by which this takes place involves training, testing, and validation, and is exemplified in Fig. 7. As indicated, the clean and scrubbed data is randomly subdivided into three subsets: T1, 60%, is used for training the network; T2, 20%, is used for testing the stopping rule; and T3, 20%, is used for testing the resulting network. The stopping rule reduces the likelihood that the network will become overtrained, by stopping the training on T1 when the predictive ability of the network, as measured on T2, is no longer improved.

---

**5. Genetic algorithms**

Genetic algorithms (GAs) are automated heuristics that perform optimization by emulating biological evolution. They are particularly well suited for solving problems that involve loose constraints, such as discontinuity, noise, high dimensionality, and multimodal objective functions (Goldberg, 1989).

GAs can be thought of as an automated, intelligent approach to trial and error, based on principles of natural selection. In this sense, they are modern successors to Monte Carlo search methods. The flowchart in Fig. 8 gives a representation of the process.

As indicated, GAs are iterative procedures, where each iteration ($g$) represents a generation. The process starts with an initial population of solutions, $P(0)$, which are randomly generated. From this initial population, the best solutions are “bred” with each other and the worse are discarded. The process ends when the termination criterion is satisfied.

As a simple example, suppose that the problem is to find by trial and error, the value of $x$, $x=0, 1, \ldots, 31$, which maximizes $f(x)$, where $f(x)$ is the output of a black box. Using the methodology of Holland (1975), an initial population of potential solutions $\{y_j|j=1, \ldots, N\}$ would be randomly generated, where each solution would be represented in binary form. Thus, if 0 and 31 were in this initial population of solutions, they would be represented as 00000 and 11111, respectively. A simple measure of the fitness of $y_j$ is

---

9 If $n$ is too large, the error term may not converge at all, and if it is too small, the weight updating process may get stuck in a local minimum and/or be extremely time intensive.

10 Assuming the Delta rule, the error signals become $\delta_{1j1}(t)=F'(z_{1j1}(t))(T−O)$ and $\delta_{0jk}(t)=F'(z_{0k}(t))\sum_j \delta_{1jk}(t)x_{1j}(t)$, where $F'$ denotes the differential with respect to $z$ and $z_{1j1}(t)=\sum_j w_{1jk}(t)x_{ij}$. Given the logistic form of the activation function, $F=F(1−F)$.

11 This figure is based on a discussion of an application by Brockett et al. (1994, p. 415).

12 $31=1\times2^4+1\times2^3+1\times2^2+1\times2^1+1\times2^0$. 

---
Fig. 9. Reproduction, crossover and mutation.

\[ p_j = f(y_j) \sum_j f(y_j) \], and the solutions with the highest \( p_j \)'s would be bred with one another.

A flowchart of the process for generating new populations of solutions is depicted in Fig. 9.

As indicated, there are three ways to develop a new generation of solutions: reproduction, crossover, and mutation. Reproduction adds a copy of a fit individual to the next generation. In the previous example, reproduction would take place by randomly choosing a solution from the population, where the probability a given solution would be chosen would depend on its \( p_j \) value. Crossover emulates the process of creating children, and involves the creation of new individuals (children) from the two fit parents by a recombination of their genes (parameters). In the example, crossover would take place in two steps: first, the fit parents would be randomly chosen on the basis of their \( p_j \) values; second, there would be a recombination of their genes. If, e.g., the fit parents were 11000 and 01101, crossover might result in the two children 11001 and 01100. Under mutation, a small number of gene values in the population are replaced with randomly generated values. This has the potential effect of introducing good gene values that may not have occurred in the initial population or which were eliminated during the iterations. In this illustration, the process is repeated until the new generation has the same number of individuals as the current one.

6. Simulated annealing

Annealing is the physical process of heating a solid in a heat bath until it melts and then slowly cooling it down until it crystallizes into a state of perfect structure. The free energy (stress) of the solid is minimized during this process.

Simulated annealing is a heuristic that mimics the annealing process in as much as it allows solutions that increase the value of the objective function to be accepted with a certain probability. Thus, unlike decent methods, in which only sequences that decrease the value of the objective function are accepted for further consideration, simulated annealing is a randomized improvement method that sometimes accepts a new sequence even though its objective value exceeds that of the old sequence. This procedure, which is known as hill-climbing, is represented in Fig. 10. \(^{13}\)

The figure shows the interaction of four parameters: \( i \), the current solution configuration; \( j \), an alternate configuration, randomly chosen from the neighborhood of \( i \); \( \Delta C_{ij} \), the change in the cost function as a result of using configuration \( j \) rather than \( i \); and \( c \), a control parameter. \(^{14}\) As indicated, \( j \) is the next configuration in the series if \( \Delta C_{ij} \leq 0 \): otherwise, the probability that \( j \) is the next configuration in the series is \( \exp(-\Delta C_{ij}/c) \). For a given \( c \), this process continues until an equilibrium is reached.

---

\(^{13}\)This figure is based on the discussion in Aarts and Van Laarhoven (1987, Chapter 2).

\(^{14}\)In the annealing process, \( \Delta C_{ij} \) and \( c \) would represent the change in energy and temperature, respectively.
The control parameter \(c\) is high in the initial stages of the search so that many increases in the objective function are accepted. It is systematically lowered until it reaches a small value for which virtually no further deteriorations take place. The last configuration in the process is taken as the solution to the problem.

The Traveling Salesman Problem (Lawler et al., 1985) is a well-known problem in combinatorial optimization\(^{15}\) which can be solved using simulated annealing. The problem involves the pair \((i, C)\), where \(i\) represent the potential tours of the salesman and \(C\) is a cost function which assigns a length of time to each tour. The problem is to find the tour for which \(C\) is a minimum, i.e., the tour with the shortest length.

7. Fractal analysis

A simple explanation of a fractal is that it is an object in which the parts are in some way related to the whole. Trees are an example, since they branch according to a fractal scale.

An interesting context from an actuarial perspective involves fractal time series, i.e., time series that are statistically self-similar with respect to time. This characteristic is exemplified in Fig. 11, which is adapted from Mandelbrot (1963), and shows a comparison of cotton prices over time.

Two key characteristics of fractal analysis, as it may relate to capital markets, are exemplified in the figure: first, self-similarity, the distribution of the first difference of the log of the daily closing cotton prices (1944–1958) and the distribution of the first difference of the log of the monthly closing cotton prices (1880–1940) are clearly similar; and second, the distribution of prices seems to follow a symmetric stable Paretian hypothesis.\(^{16}\)

A number of other researchers (notably Peters, 1994) have come to comparable conclusions with respect to other markets and other timeframes.

8. Chaos theory

There are two important messages associated with chaos theory. First, the original message of Lorenz (1963), was that small variations in the initial conditions of a system can produce enormous changes in the subsequent distribution of that system. Second, normal equations can produce random-looking results and, conversely, there may be order in random-looking data. These features of chaos theory are captured in the trend of the logistic equation (Baumol and Quandt, 1985)

\[
x_{t+1} = ax_t(1 - x_t) \quad 0 < x < 1,
\]

as \(a\) varies between 2.0 and 3.6, where \(t\) is the number of iterations. As shown in Fig. 12, when \(a=2.0\) in this nonlinear difference equation, \(x_t\), quickly settles down to a stable value. However, when \(a=3.6\), the system loses all stability and the number of solutions is infinite. The result is chaos.

Chaos theory and fractal analysis (see previous section) form the basis of the fractal market hypothesis (Peters, 1991, 1994), which holds that market stability depends on investors with a large range of investment horizons and that instability occurs when investors are predominantly in a short-term mode. Day (1997) presents an interesting simulation of this using

\(^{15}\) Combinatorial optimization problems present difficulties because they cannot be computed in polynomial time. Instead, they require times that are exponential functions of the problem size.

\(^{16}\) The logarithm of the characteristic function of the stable Paretian family of distributions is of the form

\[
\ln \xi(t) = \delta t - \gamma |t|^{\alpha} [1 + \beta |t| |\tan(\alpha \pi / 2)|],
\]

where \(\delta\) is the location parameter, \(\alpha\) the index of peakedness, \(\beta\) the index of skewness, and \(\gamma\) is the scale parameter. The Gaussian hypothesis requires that \(\alpha = 2\) while the stable Paretian hypothesis requires that \(0 < \alpha < 2\). For the case at hand, Mandelbrot estimated that \(\alpha \sim 1.7\).
α-investors, who use a strategy based on an independent, sophisticated estimate of “long-run” investment value and β-investors, who use relatively simple rules based on an extrapolation of a current price and fundamental value. If β-investor demand is strong enough and if α-investor demand decreases sharply enough near the topping and bottoming prices, then for almost all initial conditions, chaos (erratic, speculative fluctuations) occurs.17

9. Rule induction

The purpose of rule induction is to extract implicit classification rules from a set of sample data. The essential feature of the technique is that it uses supervised clustering, based on observations of the data, to determine the boundaries and extent of membership functions. The method is comparable to discriminant analysis.

Rule induction is easily explained with the diagram shown in Fig. 13. Each of the nodes represents a classification of the data, \( C_{ij} \), and a response rate for that classification, \( r_{ij} \), where \( i \) indicates a classification level and \( j \) represents a node within that level. \( C_{ij} \) is a subset of \( C_{kj} \) if \( i > k \) and \( C_{ij} \) dominates \( C_{kj} \) if \( r_{ij} > r_{kj} \). Thus, e.g., if, in the figure, \( C_{01} \) was a preliminary acceptable classification of data, and \( r_{21} > r_{01} \), classification \( C_{21} \) would also be an acceptable classification.

Fig. 14 shows a simple rule induction example based on the experience of a direct marketing company.18

As suggested by the rule tree, an analysis of a past product offer found a 50% response rate. Given this experience, the goal is to determine who should be targeted in future product offers and the focus is on identifying subclassifications which improve the response rate. Using the separating characteristics of age, sex, and income, leads to the rule that the marketing effort could be improved by concentrating on the males

17 It is worth noting that while the fractal marked hypothesis provides a viable explanation of market behavior, critics are quick to point out that it does not improve our current ability to predict chaotic events.

18 Adapted from Yoder (1992). Other applications are discussed in John et al. (1996) and Langley and Simon (1995).
under age 65 and the females between the ages of 40 and 65.

10. Data fusion algorithms

Data fusion algorithms (DFAs) are used to combine data from different sensors to facilitate target recognition. A simple example is a summation algorithm which merely sums the incoming data.

A general representation of the implementation of the technique is shown in Fig. 15. Intuitively, DFAs reflect human cognition in so far as they take multiple data sources into account. For example, an insurance agent attempting to classify a potential client instinctively looks simultaneously at the many attributes of a client, such as income, age, and insurance needs.

11. Data compression

Data compression is one of the first steps of the modeling process. The concern here, as it is in information theory in general, is to maximize the amount of information contained in the data.

One way to address this issue is to implement a process that operates on the raw state of input variables and brings them together into a concise set of aggregates. A common example of this is the use of residential areas as a proxy for socioeconomic characteristics. This type of aggregation might be accomplished by the use of nonlinear compression, which is related to factor analysis or principal components.

Domain segmentation is another approach to this problem. Here, the goal is to focus on segments in the population that have relatively constant behavior. An example is shown in Fig. 16. As indicated, aggregating may misrepresent the underlying structure of the data and reduce the resolution in the model. In contrast, if the domains are isolated, and the model is allowed to focus on relatively stationary behavior, the technique is more likely to extract the underlying information.

12. Fuzzy logic

The essential structure of a fuzzy logic system is depicted in the flow chart shown in Fig. 17.19

As indicated in the figure, numerical variables are the input of the system. These variables are passed through a fuzzification stage, where they are transformed to linguistic variables and subjected to inference rules. The linguistic results are then transformed by a defuzzification stage into numerical values which become the output of the system.

The implementation of fuzzy logic is best illustrated with an example. To this end, consider the problem of categorizing the risk capacity of a pension plan participant who earns $35,000 per annum and has a networth of $300,000.20

19 Adapted from Von Altrock (1997, p. 37).
20 This problem is based on an example in Bojadziev and Bojadziev (1997, Chapter 5). A similar approach appears in Klir et al. (1997, Section 10.4). Representative applications in insurance are discussed in Derrig and Ostoszewski (1995) and Young (1996).
12.1. Fuzzification

The basic approach to fuzzification is shown in Fig. 18. As indicated, it is characterized by a membership function which assigns to each value a grade of membership ($\mu$) ranging between 0 and 1. In this case, which represents annual income, income of $20,000 or less is defined to be low, income of $50,000 is defined to be medium, and income of $85,000 or higher is defined to be high. Each of these is assigned a membership grade of 1 in their respective categories. In addition, income of $50,000 or more is defined not to be low, and assigned a membership grade of 0 in the “low” category. Similarly, income of $20,000 or less or $85,000 or more is defined not to be medium and assigned a grade of 0 with respect to that category, and income of $50,000 or less is defined not to be high and assigned a grade of 0 with respect to the high category. Between those incomes the grade of membership is fuzzy, and falls between 0 and 1. Finally, the specific income of $35,000 (35 K) intercepts both the low and medium categories at a membership level of 1/2.

The fuzzification of networth (Fig. 19) follows a similar pattern. For example, $200,000 (200 K), $500,000 (500 K), and $800,000 (800 K) are the breakpoints for membership levels of 1 for low, medium and high networth, respectively. The specific networth of $300,000 (300 K) has a membership level of 1/3 in the low category and a membership level of 2/3 in the medium category.

Finally, assuming that risk capacity can be represented by a scale whose values range between 0 and 100, the membership functions associated with risk capacity might be represented as shown in Fig. 20. The problem becomes one of determining where a plan participant with an annual income of $35,000 and a networth of $300,000 appears on this risk capacity scale.

12.2. Inference rules

The key to coordinating the annual income and networth with risk capacity is the inference rules. These are nothing more than if-then rules. For this example, assume that the inference rules for risk capacity, as a
function of low (L), medium (M), and high (H) annual income and networth, are as shown in Table 1. Thus, low income and medium networth implies a low (L) risk capacity.

Fig. 21 summarizes the effect of combining Figs. 18–20 and Table 1, and shows the inference rules that “fire” when annual income is $35,000 and networth is $300,000. The minimum of the fuzzy inputs in the first two columns gives the levels of the firing (shown by the dotted lines) and their impact on the estimated risk capacity (shown by the shaded area in the third column).

Taking the union of the shaded areas of column 3 of Fig. 21 results in the fuzzy set shown in Fig. 22, which represents the overall conclusion of the problem. All that remains is to interpret this result.

### 12.3. Defuzzification

Defuzzification converts the fuzzy overall conclusion (Fig. 22) into a numerical value which is a best estimate in some sense. A common tactic is to use the center of gravity (COG) approach, which defines the numerical value of the output to be the abscissa of the center of gravity of the union. As shown in the figure, this defuzzified value is 36.47.

### 13. Case-based reasoning

The distinguishing feature of case-based reasoning (CBR) is that it provides a solution to a problem by adapting a solution that was used to solve a similar problem in the past.

Generally speaking, CBR can be thought of as the five-phase process depicted in Fig. 23. A brief explanation of each phase is as follows:

- **Presentation** involves introducing the characteristics of the problem as input to the system.
- **Retrieval** selects the closest-matching cases from the case base (database of cases). This entails the use of an index library of some sort, which is essentially a search and retrieval facility, and may involve nearest-neighbor algorithms or decision trees.
- **Adaptation** generates a solution to the problem by adapting the closest-matching cases. The adaption

\[ \text{COG defuzzification is computed as } \sum_j w_j x_j \]

where the weight \( w_j \) is the relative value of the membership function at \( x_j \), i.e., \( w_j = \mu(x_j) / \sum_{j} \mu(x_j) \).
module can involve derivational adaption, which creates a solution by using a past solution or structural adaption, which creates a solution by modifying a past solution.

- Validation checks the validity of the solution.
- Update adds the solution to the case base if the validated solution is not represented in the case base.

As with other processes of this kind, the extent to which these steps are implemented and the techniques used can vary considerably.

The implementation of CBR is exemplified in Fig. 24, which depicts its use in a help desk situation. As indicated, when assistance is required, the help desk is contacted and a description of the problem is imputed. Given the input, the database is accessed for the closest-matching case and after clarifying questions have been resolved, a potential solution is proposed. If the solution is not optimal and the database is not exhausted, an alternate solution is proposed and evaluated. When the solution is not optimal and the database is exhausted, an expert is used to provide the solution. In this case, the database is updated to incorporate the new solution.

14. Comment

It seems inevitable that ANM techniques will become important tools for risk and insurance researchers. Whether driven by the need for an unbiased solution, or forced to use a heuristic approach because of the vagueness of the underlying theory, or because of the increasing importance of the interaction terms, or simply because of the proliferation of user-friendly ANM software and high speed personal computers, the use of ANM techniques is likely to gain momentum. In anticipation of this trend, this article was written to help readers become more familiar with the characteristics of ANM techniques.

Since this article is limited to an overview of ANM techniques, some issues were not addressed. These issues include such things as the underlying theory, the details of how the techniques are implemented in
practice, and practical risk and insurance examples of applications. Moreover, while each of the ANM techniques was presented individually, many of them could be employed within a single model. This is the essence of “soft computing”, which integrates neural networks, genetic algorithms and fuzzy logic, under the hypothesis that these methodologies are for the most part complementary and synergistic. 23

The foregoing omissions aside, if the overview presented in this article stimulates readers to add ANM techniques to their arsenal of research tools, the article will have served its purpose.

Acknowledgements

The help of J. Scott Pflumm on the genetic algorithms and fuzzy logic portions of this paper is gratefully acknowledged. This study was sponsored by the Robert G. Schwartz Faculty Fellowship at the Penn State University and a grant of the Committee on Knowledge Extension and Research (CKER) of the Society of Actuaries.

References


23 Soft computing is a concept that was introduced by Zadeh (1992), the discoverer of fuzzy logic. Applications are discussed in Deboeck (1994) and Von Altrock (1997).


Fig. 24. CBR flowchart.