Interactive Multiple Objective Programming Using Tchebycheff Programs and Artificial Neural Networks

Minghe Sun
Antonie Stam
Ralph E. Steuer

WP-95-058
June 1995

Interactive Multiple Objective Programming Using Tchebycheff Programs and Artificial Neural Networks

Minghe Sun
Antonie Stam
Ralph E. Steuer

WP-95-058
June 1995

IIASA International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 807 □ Fax: +43 2236 71313 □ E-Mail: info@iiasa.ac.at

Interactive Multiple Objective Programming Using Tchebycheff Programs and Artificial Neural Networks [1]

Minghe Sun
Division of Management and Marketing
College of Business
The University of Texas at San Antonio
San Antonio, TX 78249-0634, U.S.A.

Antonie Stam
International Institute for Applied Systems Analysis
A-2361 Laxenburg, Austria
and
Department of Management
Terry College of Business
The University of Georgia
Athens, GA 30602, U.S.A.

Ralph E. Steuer
Faculty of Management Science
Brooks Hall
University of Georgia
Athens, GA 30602, USA

June 14, 1995

[1]The first author was partially supported by a Faculty Summer Research Grant from the College of Business, the University of Texas at San Antonio. The second author gratefully acknowledges support from the International Institute for Applied Systems Analysis, Austria and a Terry Research Fellowship from the University of Georgia.

FOREWORD

Artificial neural networks have been shown to possess the ability to accurately learn and represent complex mappings, and have successfully been applied to pattern recognition problems. The authors of the current paper successfully argue that a decision maker's preference structure may be viewed as a pattern, so that it should be possible to use artificial neural networks to represent this structure accurately. In previous work, the authors developed a neural network based method for interactive multicriteria optimization problems that indeed yields more accurate solutions than traditional methods. The current paper extends their earlier work, combining elements of an effective interactive sampling method with neural networks to capture the decision maker's preference structure. In extensive simulation experiments, the resulting method proves more effective for most problems analyzed than both the sampling interactive method and their earlier neural network based method, thus providing an attractive alternative to existing interactive multicriteria optimization procedures, and offering an interesting methodological contribution to the field of decision support modeling as well as a useful application of artificial neural networks in this area.

Interactive Multiple Objective Programming Using Tchebycheff Programs and Artificial Neural Networks

ABSTRACT

A new interactive multiple objective programming procedure is developed that combines the strengths of the Interactive Weighted Tchebycheff Procedure (Steuer and Choo 1983) and the Interactive FFANN Procedure (Sun, Stam and Steuer 1993). In this new procedure, nondominated trial solutions are generated by solving Augmented Weighted Tchebycheff Programs (Steuer 1986), based on which the decision maker articulates his/her preference information by assigning "values" to these solutions or by making pairwise comparisons. The elicited preference information is used to train a feed-forward artificial neural network, which in turn is used to screen new trial solutions for presentation to decision maker in the next iteration. Computational results are reported, comparing the current procedure with the Interactive Weighted Tchebycheff Procedure and the Interactive FFANN Procedure. The results show that this new procedure yields good quality solutions.

Keywords: Artificial Neural Networks, Multiple Objective Programming, Interactive Procedure, Multiple Criteria Decision Making.

Interactive Multiple Objective Programming Using Tchebycheff Programs and Artificial Neural Networks

1. INTRODUCTION

Real life decision making problems often involve multiple criteria and can be modeled as multiple objective programming (MOP) problems. Due to the conflicting nature of the criteria, the search for the most preferred solution must incorporate feedback from the decision maker (DM) in the form of preference information. In practice, the DM's preference structure may be complex, requiring sophisticated methods and tools capable of both capturing these complex structures and aiding the solution procedure in identifying the most preferred solution. To date, the development of effective MOP solution procedures remains a most challenging research area. In practice, a wide range of MOP solution procedures is necessary, because not every solution procedure is suitable for all MOP problems and not all DMs like to use the same procedure.

One class of interactive MOP procedures is based on sampling nondominated solutions for presentation to the DM, who then provides preference information based on these solutions, after which additional solutions are sampled until a satisfactory solution is obtained. One of the most widely used sampling-based methods is the Interactive Weighted Tchebycheff Procedure (IWTP) (Steuer and Choo 1983; Steuer 1986). Features of the IWTP that contribute to its success include the way representative nondominated solutions are sampled, and the way solution space from which new solutions are sampled is modified based on the preference information provided by the DM. In the IWTP, the DM is presented with a small set of well-spaced, representative sample solutions, and is asked to select the most preferred one.

Recently, researchers have shown that artificial neural networks (ANNs) are effective in recognizing complex patterns, and pattern recognition has emerged as one of the most successful application areas of neural networks (Másson and Wang 1990). Since a DM's preference structure can be viewed as a pattern, the idea of using ANNs to capture this structure is natural, in particular if the structure is complex. Sun, Stam and Steuer (1993) have developed the Interactive FFANN Procedure which employs a feed-forward artificial neural network (FFANN) to represent the DM's preferences within an interactive MOP framework. In each iteration, a trained FFANN, approximating the DM's preference structure, is used as the objective function of a nonlinear programming model, and traditional nonlinear optimization techniques are used to search for an improved solution. In computational experiments involving several types of complex preference patterns, their procedure was shown to be more effective than traditional interactive MOP procedures.

Building on the strengths of the IWTP and the Interactive FFANN Procedure, the current paper develops a robust interactive MOP procedure that can be more effective and flexible than previous procedures. In the procedure, the DM initially reveals preference information by rating a sample of nondominated criterion vectors generated using Augmented Weighted Tchebycheff Programs (AWTPs). This information is then used to train a FFANN. At each subsequent iteration, the trained FFANN is used to select a subset of "most preferred" solutions from a large set of additional nondominated solutions for evaluation by the DM. The preference information elicited on this subset is used to further train the FFANN. This process is repeated until a satisfactory solution is reached.

The proposed procedure differs from the IWTP in the way preference information is elicited, represented and utilized, as well as in the way sample solutions are selected for presentation to the DM. Moreover, the proposed procedure is different from the Interactive FFANN Procedure in the way trial solutions are generated and preference information is utilized. As both the Interactive FFANN Procedure and the procedure developed in this paper employ a FFANN to represent the DM's preference structure, the former will be called the FFANN-1 Method, and the latter the FFANN-2 Method.

The remainder of this paper is organized as follows. MOP-related concepts and notation are introduced in Section 2. Section 3 briefly describes FFANNs, and discusses the training process and construction of the training sets. The FFANN-2 Method is presented in Section 4. Section 5 provides the design of a computational experiment to compare the FFANN-2 Method with the FFANN-1 Method and the IWTP, and the experimental results are reported in Section 6. Section 7 concludes with a discussion of directions for future research.

2. MULTIPLE OBJECTIVE PROGRAMMING PROBLEMS

An MOP problem may be stated as

$$\text{MOP:} \qquad \begin{array}{ll} max & \mathbf{z} = \boldsymbol{f}(\mathbf{x}) \qquad (1) \\ s.t. & \mathbf{x} \in X, \qquad (2) \end{array}$$

where $X \subset \Re^n$ is the feasible region in decision space and $\boldsymbol{f}(\mathbf{x})=(f_1(\mathbf{x}), ..., f_k(\mathbf{x}))$ is a vector-valued objective function. In continuous MOP problems, X is usually defined by a set of inequalities and equations. A vector $\mathbf{x}$ is a feasible solution in decision space if $\mathbf{x} \in X$. The set $Z = \{\mathbf{z} \mid \mathbf{z} = \boldsymbol{f}(\mathbf{x}), \mathbf{x} \in X\} \subset \Re^k$ is the feasible region in criterion space, and each criterion vector $\mathbf{z} \in Z$ is a feasible solution in criterion space.

A criterion vector $\bar{\mathbf{z}} \in Z$ is said to be nondominated if and only if there does not exist another $\mathbf{z} \in Z$ such that $z_i \geq \bar{z}_i$ for all i and $z_i > \bar{z}_i$ for at least one i (Yu 1985). The set of all nondominated criterion vectors is denoted by N. If $\bar{\mathbf{z}} \in Z$ and $\bar{\mathbf{z}} \notin N$, then $\bar{\mathbf{z}}$ is dominated. The vector $\mathbf{z}^{\mathbf{ideal}} \in \Re^k$ with components z_i^{ideal} defined by $z_i^{ideal} = max\ \{z_i \mid \mathbf{z} \in Z\}$ is called the ideal point. The components of a utopia vector $\mathbf{z^{utop}}$ are defined by $z_i^{utop} = z_i^{ideal} + \epsilon_i$, where $\epsilon_i > 0$ and small. Although $\mathbf{z^{ideal}}$ and $\mathbf{z^{utop}}$ are generally infeasible, they play an important role in many solution procedures and also in the FFANN-2 Method. The vector $\mathbf{z^{nadir}}$, defined by $z_i^{nadir} = min\{z_i \mid \mathbf{z} \in N\}$, is called the nadir point. In practice, $\mathbf{z^{nadir}}$ may have to be estimated from the payoff table (Isermann and Steuer 1988; Korhonen, Salo and Steuer 1994) if the MOP problem is complex.

The set $\Lambda = \left\{\boldsymbol{\lambda} \in \Re^k \mid \sum_{i=1}^{k} \lambda_i = 1 \text{ and } \lambda_i \geq 0,\ \forall i\right\} \subset \Re^k$ is the weighting vector space and any $\boldsymbol{\lambda} \in \Lambda$ is called a weighting vector. Given any $\boldsymbol{\lambda} \in \Lambda$, a nondominated solution can be obtained for the MOP problem defined in (1)–(2) by solving an AWTP:

$$\begin{array}{lll} \text{AWTP:} & min & \alpha + \rho \sum_{i=1}^{k} (z_i^{**} - z_i) \\ & s.t. & \mathbf{x} \in X \\ & & \mathbf{z} = \boldsymbol{f}(\mathbf{x}) \\ & & \alpha \geq \lambda_i (z_i^{**} - z_i)\ , \quad \forall i \end{array}$$

where ρ is a small scalar (Steuer 1986).

As the objectives in an MOP problem ($k \geq 2$) are usually conflicting, there may not exist any $\mathbf{x} \in X$ that maximizes all objective functions simultaneously. Hence, in the context of MOP, a solution $\mathbf{z^{opt}} \in Z$ most preferred by the DM is called an optimal solution. In practice, the final solution $\mathbf{z^{fin}} \in Z$ obtained using an MOP procedure may or may not be equal to $\mathbf{z^{opt}}$.

Good reviews of the theory of MOP can be found in Yu (1985), Steuer (1986) and Gardiner and Steuer (1994). Yu (1985) also covers a wide range of related topics, such as value function theory and an introduction to habitual domains. The most promising methods for solving MOP problems have been interactive MOP procedures, with typically two alternating phases -- the solution generation phase and the solution evaluation phase (Steuer 1986). Examples of interactive procedures include STEM (Benayoun, de Montgolfier, Tergny, and Laritchev 1971), the Geoffrion-Dyer-Feinberg Procedure (1972), the Visual Interactive Approach (Korhonen and Wallenius 1988), the IWTP (Steuer and Choo 1983; Steuer 1986; Steuer, Silverman and Whisman 1993), the Reference Point Method (Wierzbicki 1977, 1980), and the Zionts-Wallenius Method (1976, 1983). These interactive procedures differ in the way trial solutions are generated, preference information is elicited, preference information is represented, and preference information is used to search for improved solutions.

3. FEED-FORWARD ARTIFICIAL NEURAL NETWORKS

3.1. Feed-Forward Artificial Neural Networks

Artificial neural networks (ANNs) have been applied to many practical problems, especially to classification and pattern recognition problems (Másson and Wang 1990; Wasserman 1989). Recently, ANNs have been applied to traditional optimization problems, including combinatorial optimization (Hopfield and Tank 1985; Aarts and Korst 1989) and linear programming (Tank and Hopfield 1986; Barbosa and de Carvalho 1990; Maa and Shanblatt 1990; Wang and Chankong 1992). Wang and Malakooti (1992) and Malakooti and Zhou (1994) use FFANNs to solve discrete MOP problems. Sun (1992) and Sun, Stam and Steuer (1993) combine FFANNs with traditional optimization techniques to solve continuous MOP problems. Burke and Ignizio (1992) provide an overview of connections between ANNs and operations research.

If the DM's true preference structure can be represented by a value function V: $\Re^k \rightarrow \Re$, an optimal solution to the MOP problem in (1)–(2) can be obtained by maximizing V over Z. Unfortunately, in real-life applications, the exact nature of V is often unknown, and a functional form representing the DM's preference structure may be intractible and not even exist (Yu 1985). In addition, the preference pattern of the DM may not be stable over time. Even if a value function V over Z exists, it may be difficult, if not impossible, to assess. Yet, many traditional MOP procedures require either implicit or explicit assumptions about the functional form of V. Given their proven ability to learn complex mappings, we choose to use FFANNs to capture the potentially complicated preference structure of the DM.

An ANN is a set of processing units or computational elements (nodes), connected with links (arcs), with connectivity weights representing the strength of the connections. A FFANN is a hierarchically structured ANN in which nodes are organized in layers, and the directed arcs only link nodes in lower layers to nodes in higher layers. The direction of an arc represents the direction of the signal flow. Denote node r in layer i by u_r^i, and the connectivity weight from u_s^j to u_r^i by w_{rs}^{ij}. Let the number of nodes in layer i be given by n_i, and the number of layers with processing units by d. Nodes in the input layer (layer 0) simply distribute inputs from the outside world to nodes in other layers, do not perform any computational function and are not processing units. Nodes in the d-th (*i.e.*, the last) layer, called the output layer, generate output to the outside world. Nodes in layers between the input and output layer are called hidden nodes, and the corresponding layers hidden layers.

Each node u_r^i, for $i \geq 1$, has a bias or threshold θ_r^i. By adding an extra input unit $u_{n_0+1}^0$ with a constant input of 1, and connecting it to all nodes in all hidden and output layers, the node biases can be treated in the same way as all other connectivity weights, *i.e.*, $w_{r,n_0+1}^{i0} = \theta_r^i$. A fully connected FFANN with $d = 2$, $n_0 = 3$, $n_1 = 2$, and $n_2 = 1$ is shown schematically in Figure 1. In this figure, the connectivity weights are shown on the arcs, and the connectivity weights on the extra arcs from the extra input node are the node biases.

Figure 1 About Here

The connectivity weights and node biases are the parameters of the FFANN. For easy reference, $W \in \Re^{\omega}$ is used to denote the collection of all FFANN parameters, where $\omega = \sum_{i=1}^{d} \left(\sum_{j=0}^{i-1} n_i n_j + n_i \right)$. $\Re^{\omega}$ is called the parameter space of the FFANN. For the FFANN in Figure 1, $\omega = 14$.

A FFANN maps from the input space $\Re^{n_0}$ to the output space $\Re^{n_d}$, *i.e.*, for an input vector $\mathbf{z} \in \Re^{n_0}$, the FFANN computes an output vector $\mathbf{o} \in \Re^{n_d}$ (forward pass). In this mapping, node inputs and outputs are compared sequentially from the input layer to the output layer. Let z_r^i denote the input to u_r^i and let o_r^i denote the output from u_r^i. For $i > 0$, z_r^i is the weighted sum of the outputs of nodes directly connected to u_r^i in all other lower layers plus θ_r^i,

$$z_r^i = \sum_{j=0}^{i-1} \sum_{s=1}^{n_j} w_{rs}^{ij} o_s^j + \theta_r^i. \tag{3}$$

The node bias θ_r^i in (3) can be viewed as a constant term in the equation defining z_r^i, in a way similar to the constant term in a regression equation. Each node ($i > 0$) has an activation function which transforms its input into output. The most frequently used activation function, also used in this research, is the logistic (sigmoidal) function in (4), which transforms the input to an output between 0 and 1,

$$o_r^i = \left[1 + \exp(-\frac{z_r^i}{T}) \right]^{-1}, \tag{4}$$

where the temperature T determines the steepness of the activation function. The output of the output layer is the output of the FFANN, *i.e.*, $o_r = o_r^d$.

Figure 2 About Here

To demonstrate the mapping of the FFANN in Figure 2, suppose that an input vector is $\mathbf{z} = (0.99,\ 0.76,\ 0.42)$. From (3) it follows that $z_1^1 = -4.68(0.99) - 12.24(0.76) - 15.00(0.42) + 14.18 = -6.06$, and $z_2^1 = -4.38(0.99) - 13.69(0.76) - 20.30(0.42) + 14.19 = -9.08$. With $T = 10$, $o_1^1 = [1 + \exp(-\frac{-6.06}{10})]^{-1} = 0.35$ and $o_2^1 = [1 + \exp(-\frac{-9.08}{10})]^{-1} = 0.29$. Similarly, the input to the single output node is $z_1^2 = 27.73(0.99) + 33.02(0.76) + 57.74(0.42) - 37.12(0.35) - 40.43(0.29) - 55.27 = -3.20$, and the FFANN output is $o_1 = o_1^2 = [1 + \exp(-\frac{-3.20}{10})]^{-1} = 0.42$.

FFANNs can be treated as complex nonlinear functions. For instance, in the case of a FFANN with n_1 hidden nodes in one hidden layer and with a single node in the output layer, the network output can be written as

$$o_1 = o_1^2 = \left\{ 1 + \exp\left[-\tfrac{1}{T} \left(\sum_{s=1}^{k} w_{1s}^{20} z_s + \sum_{r=1}^{n_1} w_{1r}^{21} \Big(1 + \exp\big(-\tfrac{1}{T} (\sum_{s=1}^{k} w_{rs}^{10} z_s + \theta_r^1) \big) \Big)^{-1} + \theta_1^2 \right) \right] \right\}^{-1}. \qquad (5)$$

The coefficients of this nonlinear function are the parameters of the FFANN. This function is general and flexible in approximating any mapping. Cybenko (1989) shows that functions of this type can approximate any input-output relation of interest to any desired degree of accuracy, provided that a sufficient number of hidden nodes is used.

3.2. FFANN Training

The training of a FFANN is equivalent to the estimation of the unknown coefficients of a nonlinear equation such as (5), in a way similar to regression analysis. In the training process, the objective is to determine the unknown components of $W \in \Re^{\omega}$ such that the FFANN closely represents an unknown mapping. Supervised training procedures are used to train a FFANN, where the training set consists of Q example patterns from the unknown mapping, for which both inputs and desired outputs are known. Let $\mathbf{z}_q \in \Re^{n^0}$ be the *qth* input vector, and $\mathbf{v}_q \in \Re^{n^d}$ the corresponding desired output vector in the training set. Then the compound vector $(\mathbf{z}_q, \mathbf{v}_q) \in \Re^{n^0 + n^d}$ is the *qth* training pattern.

Denote the output vector computed by the FFANN from the input vector $\mathbf{z}_q$ by $\mathbf{o}_q \in \Re^{n^d}$. The error made by the FFANN for the *qth* training pattern is

$$E_q = \tfrac{1}{2} \sum_{i=1}^{n^d} (v_{qi} - o_{qi})^2,$$

and that for all training patterns combined is

$$E = \sum_{q=1}^{Q} E_q = \tfrac{1}{2} \sum_{q=1}^{Q} \sum_{i=1}^{n^d} (v_{qi} - o_{qi})^2. \qquad (6)$$

The quantity E in (6) is used to measure the quality of the approximation made by the FFANN of the unknown mapping. Alternative measures are possible as well (NeuralWare 1993). The measure in (6) corresponds directly with the sum of the squared errors criterion in regression analysis. For a given training set, E is a function of W and can be written as $E(W)$.

The FFANN-2 Method uses the training algorithm developed by Sun (1992), based on error backpropagation (Rumelhart, Hinton and Williams 1986). When the training process is started, the components of W are randomly initialized to small values. During the training process, the components of W are iteratively adjusted so as to minimize $E(W)$. Each iteration (epoch) consists of three major steps. In the first step, input vectors $\mathbf{z}_q$, $q = 1, \cdots, Q$, are presented to the FFANN, one at the time, the FFANN computes the output $\mathbf{o}_q$ corresponding to each input vector through the forward pass, and the error signal E_q is computed for each output node. In the second step, the error signals are propagated back (the backward path) through the network to compute an error signal for each of the hidden nodes. In the third step, the error signals are used to adjust the components of W in order to reduce $E(W)$.

Denote the gradient of $E(W)$ with respect to W at iteration h by $\nabla E(W_h)$. In the training algorithm, the adjustment in W is made along the direction $D_h = -\nabla E(W_h) + \alpha_h D_{h-1}$, where D_{h-1} is the direction of the previous iteration and α_h is a momentum factor. The term $\alpha_h D_{h-1}$ serves to speed the convergence of the algorithm by deflecting from the steepest descent direction. The connectivity weights are updated by $W_{h+1} = W_h + \eta D_h$, where η is the learning rate, and W_h represents the status of W at the beginning of iteration h. In the training algorithm, the value of α_h is based on conjugate directions (Luenberger 1984), and the learning rate η is determined by a line search method.

This training process is repeated until the FFANN responds to each input vector $\mathbf{z}_q$ with an output vector $\mathbf{o}_q$ that is sufficiently close to the desired output $\mathbf{v}_q$, *i.e.*, until $E(W)$ is reduced to a sufficiently small value. Throughout the training process, the knowledge of the unknown mapping from $\Re^{n_0}$ to $\Re^{n_d}$ contained in the training set is embodied by the components of W. Once the training has been completed, the connectivity weights and node biases are fixed (known) and the FFANN can be used to represent the unknown mapping.

In the training process, the structure of the FFANN is assumed to be known. In the application of FFANN to represent the DM's preference structure for MOP problems, the number of input and output nodes are the number of objective functions ($n_0 = k$) and 1 ($n_d = 1$), respectively. The number of hidden layers and the number of nodes in each hidden layer are determined by the complexity of the mapping that the FFANN is supposed to represent. Unfortunately, the number of hidden nodes and hidden layers needed is problem-dependent, and no general rules exist to guide their choice a priori. If in a given application too many hidden layers or hidden nodes are used, the FFANN will have too many free parameters and is therefore at risk of overtraining, resulting in a poor generalization ability. Any well-trained FFANN has the ability to generalize, *i.e.*, the ability to accurately reflect the mapping for patterns that were not in the training set. If too few hidden nodes or hidden layers are used, the FFANN may not be able to represent the actual mapping adequately.

Moreover, FFANN representations are not unique, and there may well be several different FFANN structures that can closely represent a given mapping. Recently, some techniques have been developed to determine a parsimonious (optimal) FFANN structure for a given application. A FFANN structure is called parsimonious if it minimizes the number of hidden layers and hidden nodes without compromising an adequate representation of the mapping. The techniques used to achieve network parsimony divide into two categories. One is network growing (see, for example, Fahlman and Lebiere 1990) in which one starts with a small FFANN, and hidden nodes and hidden layers are added gradually until the FFANN represents the mapping accurately. The other is network pruning (Weigend, Rumelhart and Huberman 1991), where one starts with an oversized FFANN, and redundant links and nodes are identified and deleted, until an FFANN without such redundancies is obtained.

The focus of this paper will not be on attempting to optimize the FFANN structure. Instead, the computational study below will explore the robustness of the network structure in the FFANN-2 Method, by solving the same set of test problems for a wide range of different FFANN structures. Fortunately, as the computational results will show, the performance of the FFANN-2 Method -- for the types of preference structures considered in this paper -- is not very sensitive to the particular FFANN structure used.

Prior to training a FFANN, it is necessary to construct a training set. In the FFANN-2 Method, the input of the training set consists of (rescaled) nondominated criterion vectors of the MOP problem under analysis, and the desired output consists of the corresponding preference information elicited from the DM. Following the FFANN-1 Method, in the FFANN-2 Method the DM either assigns "values" to criterion vectors directly, or makes pairwise comparisons, yielding ratings for each solution.

In the direct assignment approach, the nadir point $\mathbf{z}^{\mathbf{nadir}}$ is given a low "value", and the ideal point $\mathbf{z}^{\mathbf{ideal}}$ a high "value." The DM then provides a preference "value" between these limits for each nondominated criterion vector presented. Denote the "values" assigned to $\mathbf{z}^{\mathbf{nadir}}$, $\mathbf{z}^{\mathbf{ideal}}$ and an arbitrary criterion vector $\mathbf{z}$ by $V(\mathbf{z}^{\mathbf{nadir}})$, $V(\mathbf{z}^{\mathbf{ideal}})$ and $V(\mathbf{z})$, respectively. The components of $\mathbf{z}$ and $V(\mathbf{z})$ are rescaled as

$$z_i' = \frac{z_i - z_i^{nadir}}{z_i^{ideal} - z_i^{nadir}}\,, \text{ for } i = 1, \cdots, k, \tag{7}$$

and

$$v(\mathbf{z}) = \frac{V(\mathbf{z}) - V(\mathbf{z}^{\mathbf{nadir}})}{V(\mathbf{z}^{\mathbf{ideal}}) - V(\mathbf{z}^{\mathbf{nadir}})}\,, \tag{8}$$

so that $\mathbf{z}'$ and $v(\mathbf{z})$ constitute a pattern in the training set.

In the pairwise comparison approach, the DM is asked to answer questions about the relative attractiveness of the solutions in the training set that are similar to those posed in the Analytical Hierarchy Process (AHP) (Saaty 1988). The pairwise comparisons result in a reciprocal comparison matrix. The normalized principal eigenvector components of this reciprocal comparison matrix, which can be viewed as the priorities of the alternative solutions (Saaty 1988) has shown, are used as the desired outputs in the training set. For each trial solution $\mathbf{z}$, $\mathbf{z}'$ as determined by (7) and the normalized principal eigenvector component corresponding to $\mathbf{z}$ constitute a training pattern.

4. THE FFANN-2 METHOD

As mentioned in the introduction, the FFANN-2 Method is built on the strengths of several existing procedures. The framework of the FFANN-2 Method is based on the IWTP (Steuer and Choo 1983; Steuer 1986), and the method used to elicit and represent preference information is similar to that in the FFANN-1 method.

At each iteration of the IWTP, typically $2P$ solutions are sampled from N and then filtered to select the P most different (Steuer 1986) for evaluation by the DM. Even though this would be desirable, the IWTP does not ensure that the most attractive among the $2P$ solutions is included in the P solutions. Thus, while the P solutions that the DM evaluates may be representative of N, the filtering procedure implies that the DM may not necessarily have the chance to reach the most preferred solution. The discarded solutions will be "wasted" because they are not seen by the DM.

In the FFANN-1 Method, a trained FFANN is used as the objective function in a nonlinear programming problem which is solved with traditional nonlinear optimization techniques, in order to find an improved solution that maximizes the output of the FFANN. One drawback of this procedure is that there is no proof that an improved trial solution found in this way is nondominated, although it is usually better than the ones previously presented to the DM. This may be undesirable, because dominated solutions are not contenders for optimality (Steuer 1986). Moreover, solving nonlinear programming problems is inherently more complex than solving linear programs, and nonlinear programming software is not as readily available as linear programming packages.

Our motivation in developing the FFANN-2 Method is to overcome the shortcomings of the IWTP and FFANN-1, and to take advantage of their individual strengths. The FFANN-2 Method corresponds to the IWTP in the way the sample solutions are generated and the weighting vector space is reduced. The FFANN-2 Method elicits and represents the preference information in the same way as the FFANN-1 Method. For example, at each iteration, instead of filtering the $2P$ nondominated solutions to obtain the P most different ones, the FFANN-2 Method uses a trained FFANN to screen candidate nondominated solutions to obtain the P most preferred ones for presentation to the DM. Next, the FFANN-2 Method is outlined step-by-step, followed by comments and further explanations.

Step 1. Decide with the DM on the number of iterations t, and the number of nondominated solutions P to be evaluated at each iteration. Specify the weighting vector space reduction factor r. Determine the appropriate FFANN structure. Solve for $\mathbf{z}^{\mathbf{ideal}}$, and find or estimate $\mathbf{z}^{\mathbf{nadir}}$. Let $[\ell_i^{(1)}, u_i^{(1)}] = [0, 1]$ for all i, and let $h = 0$.

Step 2. Let $h := h + 1$, and randomly generate $20k$ weighting vectors from $\Lambda^{(h)} = \left\{\boldsymbol{\lambda} \in \Re^k \mid \lambda_i \in (l_i^{(h)}, u_i^{(h)}), \sum_{i=1}^{k} \lambda_i = 1\right\}$. Filter the $20k$ weighting vectors to obtain the $2P$ most widely dispersed ones.

Step 3. Solve one AWTP for each of the $2P$ weighting vectors to obtain $2P$ nondominated criterion vectors. If $h = 1$, execute only Step 3a, otherwise execute only Step 3b.

3a. Filter the $2P$ nondominated criterion vectors to obtain the P most different ones.

3b. Use the trained FFANN to select the P vectors with the highest output values from the $2P$ nondominated criterion vectors.

Step 4. Present the P nondominated criterion vectors, together with $\mathbf{z}^{\mathbf{ideal}}$ and $\mathbf{z}^{\mathbf{nadir}}$, as well as $\mathbf{z}^{(h-1)}$ if $h > 1$, to the DM. Let the DM articulate his preferences by assigning "values" to the P criterion vectors or by making pairwise comparisons among these criterion vectors. Rescale the P criterion vectors using (7). Normalize the "values" using (8) if "values" are assigned, or compute the normalized principal eigenvector of the reciprocal comparison matrix if pairwise comparisons are made. The criterion vector corresponding to the highest "value" or to the largest component of the eigenvector is the most preferred solution $\mathbf{z}^{(h)}$ of the current iteration. If the DM is satisfied with $\mathbf{z}^{(h)}$ as the final solution and wishes to terminate the solution process, go to Step 9. Otherwise go to Step 5.

Step 5. If $h < t$, go to Step 6, otherwise go to Step 8.

Step 6. Train the FFANN with the rescaled criterion vectors as inputs and the corresponding normalized "values" or the components of the normalized principal eigenvector as the desired outputs. The trained FFANN is an approximate representation of the DM's preference structure.

Step 7. Compute the weighting vector $\boldsymbol{\lambda}^{(h)}$ determined by $\mathbf{z}^{\mathbf{ideal}}$ and $\mathbf{z}^{(h)}$ by:

$$\lambda_i^{(h)} = \left(\frac{1}{z_i^{ideal} - z_i^{(h)}}\right)\left[\sum_{j=1}^{k} \frac{1}{z_j^{ideal} - z_j^{(h)}}\right]^{-1},$$

form

$$[l_i^{(h+1)}, u_i^{(h+1)}] = \begin{cases} [0, r^h], & \text{if } \lambda_i^{(h)} - \frac{r^h}{2} \leq 0 \\ [1 - r^h, 1], & \text{if } \lambda_i^{(h)} + \frac{r^h}{2} \geq 1 \\ [\lambda_i^{(h)} - \frac{r^h}{2}, \lambda_i^{(h)} + \frac{r^h}{2}], & \text{otherwise,} \end{cases}$$

and go to Step 2.

Step 8. If the DM wishes to extend the search, go to Step 6, otherwise, go to Step 9.

Step 9. Denote the inverse image of $\mathbf{z}^{(h)}$ by $\mathbf{x}^{(h)}$, terminate the solution process with $(\mathbf{x}^{\mathbf{fin}}, \mathbf{z}^{\mathbf{fin}}) = (\mathbf{x}^{(h)}, \mathbf{z}^{(h)})$ as the final solution.

Guidelines about the selection of appropriate values for P, t, and r in Step 1 can be found in Steuer (1986). The DM may terminate the solution procedure prior to the completion of t iterations, if a satisfactory solution has been identified, and can extend the search beyond t iterations in order to explore additional solutions. In Step 2, the number of randomly generated weighting vectors ($20k$) is merely a recommendation based on the authors' previous experience with MOP problems. If too many weighting vectors are generated, the $2P$ widely dispersed ones left after filtering may be very close to the boundary of $\Lambda^{(h)}$, thus not representing $\Lambda^{(h)}$ well. If too few are generated, the $2P$ remaining vectors may not be widely dispersed in $\Lambda^{(h)}$.

When solving the AWTPs in Step 3, it is recommended that the ranges of each of the objective functions over N, $z_i^{ideal} - z_i^{nadir}$, be equalized. Steuer (1986) has shown that the IWTP works better when these ranges are equalized. In Step 4, the DM is allowed to modify his judgments made in any previous iterations. During the solution process the DM is expected to learn about the types of solutions that can reasonably be reached, modify his/her preferences and aspirations, and correct errors in judgment made in previous iterations. These learning and feedback mechanisms are common to any behavioral decision process, and are easily accomodated within the flexible FFANN framework -- in contrast with most traditional interactive methods, which are often ineffective in dealing with learning effects. At the training process in each iteration, the FFANN seeks to learn from the DM's modified preference information and capture the changing subjective judgments within its structure.

The implementation of the FFANN-2 Method in this paper uses the LAMBDA and FILTER programs in the ADBASE package (Steuer 1993) to generate and select weighting vectors, while the nonlinear programming code GRG2 (Lasdon and Waren 1986) is used for solving the AWTPs. Even though the computational experiment below is limited to linear MOP problems, GRG2 is used, rather than a linear programming code, because the FFANN-2 Method is able in principle to handle both linear and nonlinear MOP problems.

While the FFANN-2 Method is expected to work well, there are several reasons why it may or may not generate better solutions than other currently available interactive procedures. First, the weighting vectors in Step 2 are randomly generated from $\Lambda^{(h)}$, so that the nondominated criterion vectors are randomly generated from a subset of N. As a result, selecting the best solution among all trial solutions at a given iteration does not guarantee that better solutions will be generated in subsequent iterations. Therefore, the search procedure is not guaranteed to always lead to a good final solution.

Second, the trained FFANN can only approximate the DM's preference structure to a certain degree. Since the training set is only a sample of preference patterns, the FFANN may in fact represent the training set well, but not do as well in generalizing to other patterns. Therefore, the trained FFANN is not guaranteed to select the best among all trial solutions generated.

Third, even after extensive training, the FFANN may not have learned much from the training patterns. After a certain number of epochs in the training process, the connectivity weights and node biases may not converge to a set of values that minimizes the differences between the desired and the computed outputs. In other words, the training process may end at a local minimum point (Wasserman 1989). If this occurs, the "trained" FFANN may not even represent the training set.

It should be stressed that the second and third reasons apply generally to any neural network model, are well-documented in the literature, and are by no means limited to their application to interactive MOP. Moreover, a caveat similar to the first remark above applies to any sampling-based interactive MOP procedure.

5. DESIGN OF COMPUTATIONAL EXPERIMENT

The performance of the FFANN-2 Method is compared with that of the FFANN-1 Method and the IWTP through computational experiments, using the same linear MOP test problems as in Sun, Stam and Steuer (1993). The FFANN-2 Method is measured against the IWTP, because the IWTP is one of the best known, most frequently used and effective interactive procedures (Buchanan and Daellenbach 1987), and because the development of the FFANN-2 Method was based in part on the IWTP. The reasons for comparing hte FFANN-2 Method with the FFANN-1 Method are that both procedures use FFANNs to represent the DM's preference structure, and Sun, Stam and Steuer (1993) report evidence that the FFANN-1 Method may yield better solutions than the IWTP.

Denote the dimensions of the test problems by $k \times m \times n$, where k is the number of objectives, n is the number of decision variables, and m is the number of linear constraints in the problem. Ten sample problems each of dimensions $3 \times 5 \times 6$, $5 \times 5 \times 10$, $5 \times 8 \times 15$, $5 \times 10 \times 20$, and $6 \times 50 \times 100$ were randomly generated with ADBASE (Steuer 1993).

In the controlled computational experiments, it is necessary to specify the "true" value function in order to evaluate the attractiveness of the trial solutions generated during the interactive process. These value functions were solely used as a proxy of the DM for the purpose of the experiments. The following four value functions (Yu 1985; Steuer 1986) were used in the experiment, to provide consistent preference information across all three procedures.

1. The linear value function:

$$V_l = \sum_{i=1}^{k} w_i z_i \,. \tag{9}$$

2. The quadratic value function:

$$V_q = K - \sqrt{\sum_{i=1}^{k} [w_i(z_i^{utop} - z_i)]^2} \,. \tag{10}$$

3. The L_4-metric value function:

$$V_f = K - \left[\sum_{i=1}^{k} [w_i(z_i^{utop} - z_i)]^4\right]^{\frac{1}{4}} . \tag{11}$$

4. The Tchebycheff metric (L_∞-metric) value function:

$$V_t = K - \max_{1 \le i \le k} \left\{ w_i (z_i^{utop} - z_i) \right\} . \tag{12}$$

In (10)–(12), K is a sufficiently large positive number to ensure that the corresponding values are positive, and $\mathbf{w} \in \Re^k$ is a weighting vector with $w_i > 0$ and $\sum_{i=1}^{k} w_i = 1$. One weighting vector of the form in (13) is used for all four value functions,

$$w_i = \frac{1}{z_i^{ideal} - z_i^{nadir}} \left[\sum_{j=1}^{k} \frac{1}{z_j^{ideal} - z_j^{nadir}} \right]^{-1} . \tag{13}$$

If each objective function is normalized by $f_i'(\mathbf{x}) = f_i(\mathbf{x})[z_i^{ideal} - z_i^{nadir}]^{-1}$, then using $\mathbf{w}$ as defined in (13) is equivalent to assigning equal weights of $\frac{1}{k}$ to each of the objective functions. The complexity of the above value functions increases from the linear value function (9) to the Tchebycheff metric value function (12). The true optimal solution for each test problem and each given value function, which is used only to measure the quality of the final solution, is found with GRG2.

To verify the effect of different FFANN structures on the performance of the FFANN-2 Method, FFANNs with different numbers of hidden nodes are used. For each test problem and for each given value function, the same problem is solved several times, using FFANNs with 0, 2, 4, and 6 hidden nodes in one hidden layer, respectively. Of course, FFANNs with 0 hidden nodes do not have any hidden layer.

For comparison purposes, all three procedures are run under identical conditions. At each iteration, $P = 7$ nondominated criterion vectors are presented to the DM. A total of $t = 5$ iterations are run for all problems, except for the $6 \times 50 \times 100$ problems, which are solved in 7 iterations. The weighting vector space reduction factor for the FFANN-2 Method and the IWTP is $r = 0.6$. In generating weighting vectors, the same random number seed is used for all three procedures.

In this paper, we use two benchmarks for measuring the quality of the final solution, the nadir point $\mathbf{z}^{\mathbf{nadir}}$ and the worst nondominated point $\mathbf{z}_w$. For linear MOP problems, the worst nondominated point is the nondominated extreme point that has the lowest preference value, found by evaluating all nondominated extreme points with the corresponding value function. For a given MOP problem, $\mathbf{z}^{\mathbf{nadir}}$ is unique and independent of the specific value function used, but $\mathbf{z}_w$ is dependent not only on the functional form, but also on the parameters in the value function, *i.e.*, the weighting vector $\mathbf{w}$. Because of the difficulty in finding all nondominated extreme solutions for the $6 \times 50 \times 100$ problems, the estimated nadir point from the payoff table is used for problems in this category.

Let $V(\mathbf{z}^{\mathbf{opt}})$, $V(\mathbf{z}^{\mathbf{nadir}})$, $V(\mathbf{z}_w)$ and $V(\mathbf{z}^{\mathbf{fin}})$ be the preference values of $\mathbf{z}^{\mathbf{opt}}$, $\mathbf{z}^{\mathbf{nadir}}$, $\mathbf{z}_w$ and $\mathbf{z}^{\mathbf{fin}}$ for a given value function, respectively. Then $v(\mathbf{z}^{\mathbf{fin}})$ and $v'(\mathbf{z}^{\mathbf{fin}})$ are relative measures of the quality of the final solution:

$$v(\mathbf{z}^{\mathbf{fin}}) = \frac{V(\mathbf{z}^{\mathbf{fin}}) - V(\mathbf{z}^{\mathbf{nadir}})}{V(\mathbf{z}^{\mathbf{opt}}) - V(\mathbf{z}^{\mathbf{nadir}})} \times 100 \ ,$$

and

$$v'(\mathbf{z}^{\mathbf{fin}}) = \frac{V(\mathbf{z}^{\mathbf{fin}}) - V(\mathbf{z}_w)}{V(\mathbf{z}^{\mathbf{opt}}) - V(\mathbf{z}_w)} \times 100 \ .$$

Of course, these measures are meaningful only when comparing the performance of two or more solution procedures (Sun and Gardiner 1995).

6. COMPUTATIONAL RESULTS

In the following tables, one for each problem size, we report the mean values and standard deviations (Std.) across 10 replications, of both $v(\mathbf{z}^{\mathbf{fin}})$ and $v'(\mathbf{z}^{\mathbf{fin}})$, for the three interactive procedures. In the case of the Tchebycheff metric value function, $v(\mathbf{z}^{\mathbf{fin}})$ and $v'(\mathbf{z}^{\mathbf{fin}})$ are the same. For problems in the $6 \times 50 \times 100$ category, only one quality measure is used.

Tables 1–5 About Here

Table 1 presents the results for the $3 \times 5 \times 6$ problems. In the case of a linear value function, the FFANN-1 Method found slightly better solutions, on average, than both the FFANN-2 Method and the IWTP. The reason is that an optimal solution of a MOP problem with a linear value function can always be found at one of the extreme points, and the FFANN-1 Method can locate such extreme points when traditional nonlinear optimization techniques are used to search for improved solutions. On the other hand, although these may be very close to an optimal extreme point, the trial solutions generated using the AWTPs in the FFANN-2 Method and the IWTP are not necessarily extreme points.

For the nonlinear value functions, the FFANN-2 Method identified better solutions, on average, than the FFANN-1 Method, which in turn performed better than the IWTP. Interestingly, in the case of nonlinear value functions, the variability of the solutions found by the FFANN-1 Method was higher than that of the FFANN-2 Method. The solutions identified by the FFANN-2 Method with different numbers of hidden nodes are very similar. Except for problems with a linear value function, FFANNs with hidden nodes found slightly better solutions than FFANNs without hidden nodes.

Similar trends were found for the $5 \times 5 \times 10$ problems in Table 2, the $5 \times 8 \times 15$ problems in Table 3, and the $5 \times 10 \times 20$ problems in Table 4, with both of the FFANN procedures locating better solutions than the IWTP, especially for problems with more complicated value functions and for larger size problems. The FFANN-2 Method yielded both a higher mean performance level and a lower variability than the FFANN-1 Method, indicating that the FFANN-2 method is more consistent than the FFANN-1 method in generating high quality final solutions.

Table 5 shows that for the $6 \times 50 \times 100$ problems, the FFANN-1 Method found better quality solutions than the FFANN-2 Method, on average. One explanation is that, as k increases, the nondominated set N becomes so large that the AWTPs used in the FFANN-2 Method are no longer able to locate trial solutions which are close enough to the optimal solution within a few iterations.

In real life problems, the complexity of the DM's preference structure is unknown, therefore the use of FFANNs with hidden nodes is recommended. If the preference structure is simple, *e.g.*, linear, the FFANN will represent the preference structure equally well, no matter whether or not hidden nodes are used. However, if the preference structure is more complex, FFANNs with hidden nodes will yield better results than ones without hidden nodes. From the computational results, it is seen that the quality of the final solutions in our experiments are not very sensitive to the number of hidden nodes, as long as hidden nodes are used.

For all problem sizes and across all three procedures, the quality of the final solutions "deteriorates" as one moves from a linear value function to a Tchebycheff metric value function. This phenomenon does not necessarily mean that the solution procedures are more effective for problems with linear value functions than for problems with more complex value functions. Rather, whereas on one hand problems with more complex value functions are more difficult to solve, on the other hand, due to the geometric properties of these value functions, the same nondominated criterion vector $\mathbf{z}$ may have higher $v(\mathbf{z})$ and $v'(\mathbf{z})$ values for linear value functions than for more complex value functions. Sun and Gardiner (1995) provide a detailed explanation for this phenomenon.

All computations were performed on an IBM ES 9000 Model 720 computer. Due to the effort of training the FFANNs, both the FFANN-1 Method and the FFANN-2 Method are computationally more intensive than the IWTP. Moreover, in the case of the linear MOP problems studied in this paper, the FFANN-1 Method requires a little more effort than the FFANN-2 Method, because it also requires solving a nonlinear program at each iteration. In terms of eliciting preference information from the DM, the IWTP only requires the identification of the most preferred solution among the P presented, whereas the FFANN-1 and FFANN-2 Methods require preference statements involving all P solutions. While picking the best solution may require less effort, it may not express adequately the DM's preference structure. In any case, the additional effort required from the DM in the FFANN-1 and FFANN-2 Methods appears worthwhile, in terms of better quality judgments.

None of the problems solved in the experiments of this paper required prohibitive amounts of computer time. The time needed to train the FFANN ranged from a fraction of one second for the small problems to several minutes for the large problems. In addition to the complexity of the preference structure, the FFANN training time depends among others on the number of criteria, which should not exceed 6–8 in most applications, and the number of hidden nodes. Hence, the FFANN approach to interactive MOP is indeed computationally feasible.

7. CONCLUDING REMARKS

The results of the computational experiments conducted in this paper confirm our conjecture that the use of FFANNs to represent a DM's preference structure in solving MOP problems can lead to improved solutions, especially if the DM's preference structure is complex. For the problems in the current experimental design, the FFANN-2 Method proposed in this paper generally identified better quality and more robust solutions than the other methods considered, for most problem sizes and types of value functions. An exception was the large problem size, for which the FFANN-1 Method performed the best. Both the FFANN-1 and FFANN-2 Methods performed better than the IWTP.

However, as the relative effectiveness of any interactive method in practice depends crucially on the particular nature of the problem at hand, and each of the three methods considered in this paper has specific advantages and disadvantages, each provides a useful contribution to the field of multiple criteria decision making. Our purpose was to explore the viability of the FFANN approach, and the outcomes of our extensive computational experiments are very promising. It appears that FFANN approaches to interactive MOP may have a better chance at learning in-depth about a DM's habitual domain of thinking than traditional methods, for instance through the FFANN's ability to meaningfully generalize the preference information elicited from the DM beyond the boundaries of the specific questions posed and solutions evaluated. If this is indeed the case, it would open the door to a more flexible treatment of the interface between human decision makers and computer models that seek to aid decision makers in achieving better quality decisions. Future research should carefully explore this issue further.

Some immediate extensions of the current research are as follows. One issue of interest is to explore how well the FFANN approach can deal with imprecise preference statements on the part of the DM. In the experiments of this paper, the DM was assumed to make exact preference judments according to the type of value function specified. Since research in pattern recognition has shown that FFANNs are capable of generalizing and dealing effectively with fuzzy information, we would expect FFANN-based methods to have an even greater advantage over traditional interactive MOP methods in the presence of imprecise preference information.

A second extension of the FFANN-2 Method is to search for improved solutions with the trained FFANN as the objective function in a nonlinear programming model, in a manner similar to the FFANN-1 Method. This would eliminate the apparent limitation of the FFANN-2 Method that it is less effective in solving large size problems. This extension is not the same as the FFANN-1 method, because the FFANN-1 Method does not use any weighting vector space reduction.

A third extension is to use the trained FFANN to evaluate all the nondominated solutions generated in previous iterations in step 3b, and present the P solutions with the highest output values to the DM. The rationale for proceeding in this manner is that the DM's preference structure may shift during the solution process, as the DM examines more nondominated solutions and learns about the problem itself. In this situation, more than $2P$ nondominated solutions may be generated in Step 3, since the FFANN is used to screen the promising solutions and hence those solutions not presented to the DM are not "wasted." Like in the Reference Point Method (Wierzbicki 1977, 1980) and in the hybrid method reported by Steuer, Silverman and Whisman (1993), the DM may specify a criterion vector representing his aspiration and use this reference point to reduce the weighting vector space in Step 7.

REFERENCES

Aarts, E. H. and J. H. Korst, *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley and Sons, New York, 1989a.

Benayoun, R., J. de Montgolfier, J. Tergny and O. Laritchev, "Linear Programming with Multiple Objective Functions: Step Method (STEM)," *Mathematical Programming*, **1** (3), 366–375, 1971.

Barbosa, R. and L. A. de Carvalho, "Feasible Direction Linear Programming by Neural Networks," *Proceeding of the International Joint Conference on Neural Networks*, **III**, 941–946, 1990.

Buchanan, J. T. and H. G. Daellenbach, "A Comparative Evaluation of Interactive Methods for Multiple Objective Decision Models," *European Journal of Operational Research*, **29** (3), 353–359, 1987.

Burden, R. L. and J. D. Faires, *Numerical Analysis*, Fourth Edition, PWS-Kent, Boston, MA, 1989.

Burke, L. I. and J. P. Ignizio, "Neural Networks and Operations Research: An Overview," *Computers and Operations Research*, **19** (2), 179–189, 1992.

Chakropani, J. and J. Skorin-Kapov, "A Connectionist Approach to the Quadratic Assignment Problem," *Computers and Operations Research*, **19** (2), 287–295, 1992.

Cybenko, G., "Approximations by Superpositions of a Sigmoidal Function," *Mathematics of Control, Signals, and Systems*, **2**, 303–314, 1989.

Fahlman, S. E., and C. Lebiere, "The Cascade-Correlation Learning Architecture," in *Advances in Neural Information Processing Systems*, **2** (D. S. Touretzky, ed.), Morgan Kaufmann, San Mateo, CA, 524–532, 1990.

Gardiner, L. R. and R. E. Steuer, "Unified Interactive Multiple Objective Programming," *European Journal of Operational Research*, **74** (3), 391–406, 1994.

Geoffrion, A. M., J. S. Dyer and A. Feinberg, "An Interactive Approach for Multicriterion Optimization, with an Application to the Operation of a Academic Department," *Management Science*, **19** (4), 357–368, 1972.

Hopfield, J. J. and D. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, **52**, 141–152, 1985.

Isermann, H. and R. E. Steuer, "Computational Experience Concerning Payoff Tables and Minimum Criterion Values over the Efficient Set," *European Journal of Operational Research*, **33** (1), 91–97, 1988.

Korhonen, P., S. Salo, and R. E. Steuer, "A Heuristic for Estimating Nadir Criterion Values in Multiple Objective Linear Programming," Working Paper, Helsinki School of Economics, Helsinki, Finland, 1994.

Korhonen, P. and J. Wallenius, "A Pareto Race," *Naval Research Logistics*, **35** (6), 615–623, 1988.

Lasdon, L. S. and A. D. Waren, "GRG2 User's Guide," University of Texas, Austin, Texas, 1986.

Looi, C. -K., "Neural Network Methods in Combinatorial Optimization," *Computers and Operations Research*, **19** (2), 191–208, 1992.

Luenberger, D. G., *Linear and Nonlinear Programming*, Second Edition, Addison-Wesley, Reading, Massachusetts, 1984.

Maa, C. -Y. and M. A. Shanblatt, "Stability of Linear Programming Neural Network for Problems with Hypercube Feasible Region," *Proceedings of the International Joint Conference on Neural Networks*, **III**, 759–764, 1990.

Malakooti, B. and Y. Zhou, "Feedforward Artificial Neural Networks for Solving Discrete Multiple Criteria Decision Making Problems," *Management Science*, **40** (11), 1542–1561, 1994.

Másson, E. and Y. J. Wang, "Introduction to Computation and Learning in Artificial Neural Networks," *European Journal of Operational Research*, **47** (1), 1–28, 1990.

NeuralWare, *Using NeuralWorks: A Tutorial for NeuralWorks Professional II/Plus and NeuralWorks Explorer*, NeuralWare, Inc., Pittsburgh, PA, 1993.

Rumelhart, D. E., G. E. Hinton and R. J. Williams, "Learning Internal Representations by Error Propagation," in D. E. Rumelhart, J. L. McClelland and the PDP Group (eds.), *Parallel Distributed Processing, Volume 1: Foundations*, MIT Press, Cambridge, MA, 318–362, 1986.

Saaty, T. L., *Multicriteria Decision Making: The Analytic Hierarchy Process*, Revised Edition, RWS Publications, Pittsburgh, 1988.

Steuer, R. E., *Multiple Criteria Optimization: Theory, Computation, and Application*, Wiley, New York, 1986.

Steuer, R. E., "User's Manual for the ADBASE Multiple Objective Linear Programming Package," Terry College of Business, University of Georgia, 1993.

Steuer, R. E. and E. -U. Choo, "An Interactive Weighted Tchebycheff Procedure for Multiple Objective Programming," *Mathematical Programming*, **26**, (1), 326–344, 1983.

Steuer R. E., J. Silverman and A. W. Whisman, "A Combined Tchebycheff/Aspiration Criterion Vector Interactive Multiobjective Programming Procedure," *Management Science*, **39** (10), 1255–1260, 1993.

Sun, M., "Interactive Multiple Objective Programming Procedures via Adaptive Random Search and Feed-Forward Artificial Neural Networks," Unpublished Ph.D. Dissertation, Terry College of Business, University of Georgia, 1992.

Sun, M. and L. Gardiner, "Some Issues in the Computational Testing of Interactive Multiple Objective Programming Procedures," presented at the 12th International Conference on Multiple Criteria Decision Making, Hagen, Germany, June 18-23, 1995.

Sun, M., A. Stam and R. E. Steuer, "Solving Interactive Multiple Objective Programming Problems Using Feed-Forward Artificial Neural Networks," Working Paper 93–364, Terry College of Business, University of Georgia, 1993.

Tank, D. and J. J. Hopfield, "Simple Neural Optimization Networks: an A/D Converter, Single Decision Circuit and a Linear Programming Circuit," *IEEE Transactions on Circuits and Systems*, **CAS-33**, 533–541, 1986.

Wang, J. and V. Chankong, "Recurrent Neural Networks for Linear Programming: Analysis and Decision Principles," *Computers and Operations Research*, **19** (2), 297–311, 1992.

Wang, J. and B. Malakooti, "A Feedforward Neural Network for Multiple Criteria Decision Making," *Computers and Operations Research*, **19** (2), 151–167, 1992.

Wasserman, P. D., *Neural Computing, Theory and Practice*, Van Nostrand Reinhold, New York, 1989.

Weigend, A. S., D. E. Rumelhart, and B. A. Huberman, "Generalization by Weight Elimination with Application to Forecasting," in *Advances in Neural Information Processing systems* **3** (R. P. Lippmann, J. E. Moody, and D. S. Touretzky, eds.), Morgan Kaufmann, San Mateo, CA, 875–882, 1991.

Wierzbicki, A. P., "Basic Properties of Scalarizing Functions for Multiobjective Optimization," *Mathematische Operationsforschung und Statistik - Series Optimization*, **8** (1), 55–60, 1977.

Wierzbicki, A. P. "The Use of Reference Objectives in Multiobjective Optimization," *Lecture Notes in Economics and Mathematical Systems*, **177**, Springer-Verlag, New York, NY, 468–486, 1980.

Yu, P. L., *Multiple-Criteria Decision Making: Concepts, Techniques, and Extensions*, Plenum Press, New York, NY, 1985.

Zionts, S. and J. Wallenius, "An Interactive Programming Method for Solving the Multiple Objective Programming Problem," *Management Science*, **22** (6), 652–663, 1976.

Table 1. Quality of Final Solutions for the 3 × 5 × 6 Test Problems

n_1	FFANN − 2				FFANN − 1				IWTP			
	$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
	Linear Value Function											
0	99.58	0.830	99.14	1.645	99.91	0.288	99.82	0.577	99.27	0.848	98.48	1.824
2	99.58	0.830	99.14	1.645								
4	99.58	0.830	99.14	1.645								
6	99.58	0.830	99.14	1.645								
	Quadratic Value Function											
0	99.63	0.459	99.40	0.775	96.06	6.175	94.14	11.737	98.44	1.753	97.52	2.876
2	99.84	0.225	99.77	0.306	99.69	0.429	99.48	0.743				
4	99.84	0.225	99.77	0.306								
6	99.84	0.224	99.77	0.305								
	L_4-Metric Value Function											
0	99.59	0.656	99.49	0.767					99.02	1.392	98.78	1.671
2	99.74	0.679	99.69	0.792	98.56	1.420	98.13	1.881				
4	99.75	0.681	99.70	0.796								
6	99.75	0.670	99.71	0.783	98.56	1.458	98.13	1.934				
	Tchebycheff (L_∞-) Metric Value Function											
0	97.00	2.181							95.94	2.418		
2	98.38	1.158			92.83	5.770						
4	98.62	1.097										
6	98.80	1.059			96.28	3.373						

Table 2. Quality of Final Solutions for the 5 x 5 x 10 Test Problems

n_1	FFANN – 2				FFANN – 1				IWTP			
	$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
	Linear Value Function											
0	99.89	0.099	99.76	0.231	100.00	0.000	100.00	0.000	99.05	1.586	97.81	3.761
2	99.90	0.093	99.78	0.204								
4	99.90	0.093	99.78	0.204								
6	99.90	0.093	99.78	0.204								
	Quadratic Value Function											
0	99.81	0.272	99.64	0.527	95.88	4.044	92.94	6.586	99.32	1.104	98.64	2.243
2	99.82	0.275	99.64	0.531	99.62	1.301	99.00	3.382				
4	99.82	0.275	99.64	0.531								
6	99.77	0.267	99.56	0.514								
	L_4-Metric Value Function											
0	98.86	1.715	98.46	2.123					96.47	4.743	94.15	8.692
2	98.94	1.760	98.52	2.211	98.65	2.215	97.70	4.172				
4	98.86	1.738	98.42	2.176								
6	98.93	1.751	98.55	2.175	98.33	2.146	97.21	4.895				
	Tchebycheff (L_∞-) Metric Value Function											
0	89.00	13.598							86.75	13.043		
2	93.83	7.220			92.55	4.521						
4	95.10	2.448										
6	95.25	2.558			92.13	4.265						

Table 3. Quality of Final Solutions for the $5 \times 8 \times 15$ Test Problems

n_1	FFANN − 2				FFANN − 1				IWTP			
	$v(\mathbf{z}^{\mathrm{fin}})$		$v'(\mathbf{z}^{\mathrm{fin}})$		$v(\mathbf{z}^{\mathrm{fin}})$		$v'(\mathbf{z}^{\mathrm{fin}})$		$v(\mathbf{z}^{\mathrm{fin}})$		$v'(\mathbf{z}^{\mathrm{fin}})$	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
	Linear Value Function											
0	99.55	0.464	98.94	1.147	99.85	0.483	99.71	0.920	99.01	1.411	97.19	4.624
2	99.55	0.464	98.94	1.147								
4	99.55	0.464	98.94	1.147								
6	99.55	0.464	98.94	1.147								
	Quadratic Value Function											
0	99.33	0.623	98.69	1.350	98.99	0.576	98.27	1.103	98.82	1.443	97.89	2.442
2	99.33	0.623	98.69	1.351	99.42	0.410	99.01	0.748				
4	99.33	0.623	98.69	1.350								
6	99.33	0.561	98.69	1.215								
	L_4-Metric Value Function											
0	98.95	0.876	98.48	1.343					96.78	4.677	95.15	7.617
2	98.92	0.888	98.44	1.354	98.34	1.159	97.61	1.553				
4	98.96	0.860	98.48	1.331								
6	98.96	0.860	98.48	1.331	97.95	1.421	97.15	1.891				
	Tchebycheff (L_∞-) Metric Value Function											
0	94.10	4.275							87.98	10.321		
2	94.52	3.940			92.28	6.642						
4	94.52	3.940										
6	94.35	3.864			91.63	8.161						

Table 4. Quality of Final Solutions for the $5 \times 10 \times 20$ Test Problems

n_1	FFANN − 2				FFANN − 1				IWTP			
	$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$		$v(\mathbf{z}^{\text{fin}})$		$v'(\mathbf{z}^{\text{fin}})$	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
	Linear Value Function											
0	99.65	0.349	99.24	0.834	99.68	0.683	99.26	1.564	98.97	1.039	97.69	2.567
2	99.65	0.349	98.98	0.807								
4	99.65	0.349	99.24	0.834								
6	99.65	0.349	99.24	0.834								
	Quadratic Value Function											
0	99.36	0.520	98.94	0.845	98.52	1.581	97.59	2.447	98.23	1.073	97.00	2.037
2	99.36	0.520	98.94	0.845	99.26	0.720	98.77	1.353				
4	99.37	0.519	98.95	0.840								
6	99.37	0.519	98.95	0.840								
	L_4-Metric Value Function											
0	99.18	0.795	98.86	1.151					96.61	3.985	95.36	5.316
2	99.38	0.470	99.16	0.658	98.62	1.119	98.35	1.588				
4	99.41	0.450	99.20	0.635								
6	99.43	0.472	99.23	0.662	98.69	1.512	98.11	2.258				
	Tchebycheff (L_∞-) Metric Value Function											
0	94.20	5.988							87.14	7.041		
2	95.18	4.126			91.94	3.739						
4	94.99	4.143										
6	95.26	3.943			94.84	2.982						

Table 5. Quality of Final Solutions for the $6 \times 50 \times 100$ Test Problems

n_1	FFANN − 2		FFANN − 1		IWTP	
	$v(\mathbf{z}^{\mathbf{fin}})$		$v(\mathbf{z}^{\mathbf{fin}})$		$v(\mathbf{z}^{\mathbf{fin}})$	
	Mean	Std.	Mean	Std.	Mean	Std.
	Linear Value Function					
0	98.39	1.051	99.81	0.116	96.98	2.044
2	98.41	1.043	99.80	0.139		
4	98.41	1.043				
6	98.41	1.043				
	Quadratic Value Function					
0	98.51	1.728	99.27	0.530	96.39	3.266
2	98.51	1.728	99.30	0.757		
4	98.51	1.728	99.59	0.268		
6	98.51	1.728	99.50	0.309		
	L_4-Metric Value Function					
0	97.48	1.969	95.77	2.194	96.24	1.930
2	97.38	1.919	97.13	2.154		
4	97.78	1.841	98.67	0.916		
6	97.78	1.841	99.02	0.894		
	Tchebycheff (L_∞-) Metric Value Function					
0	78.34	12.675	72.09	11.511	71.47	14.992
2	77.82	17.653	76.63	12.871		
4	80.72	14.684	76.10	9.572		
6	80.72	14.684	84.88	8.196		

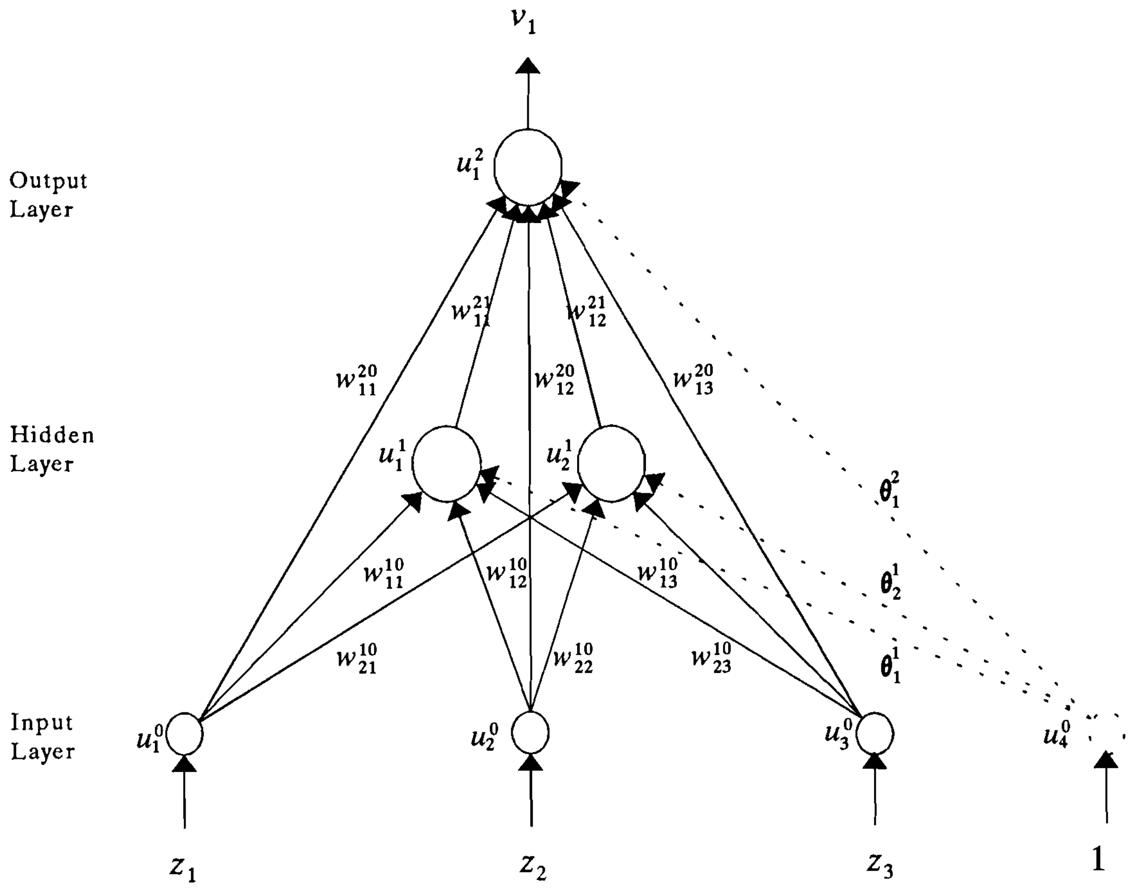

Figure 1: A Fully Connected FFANN.

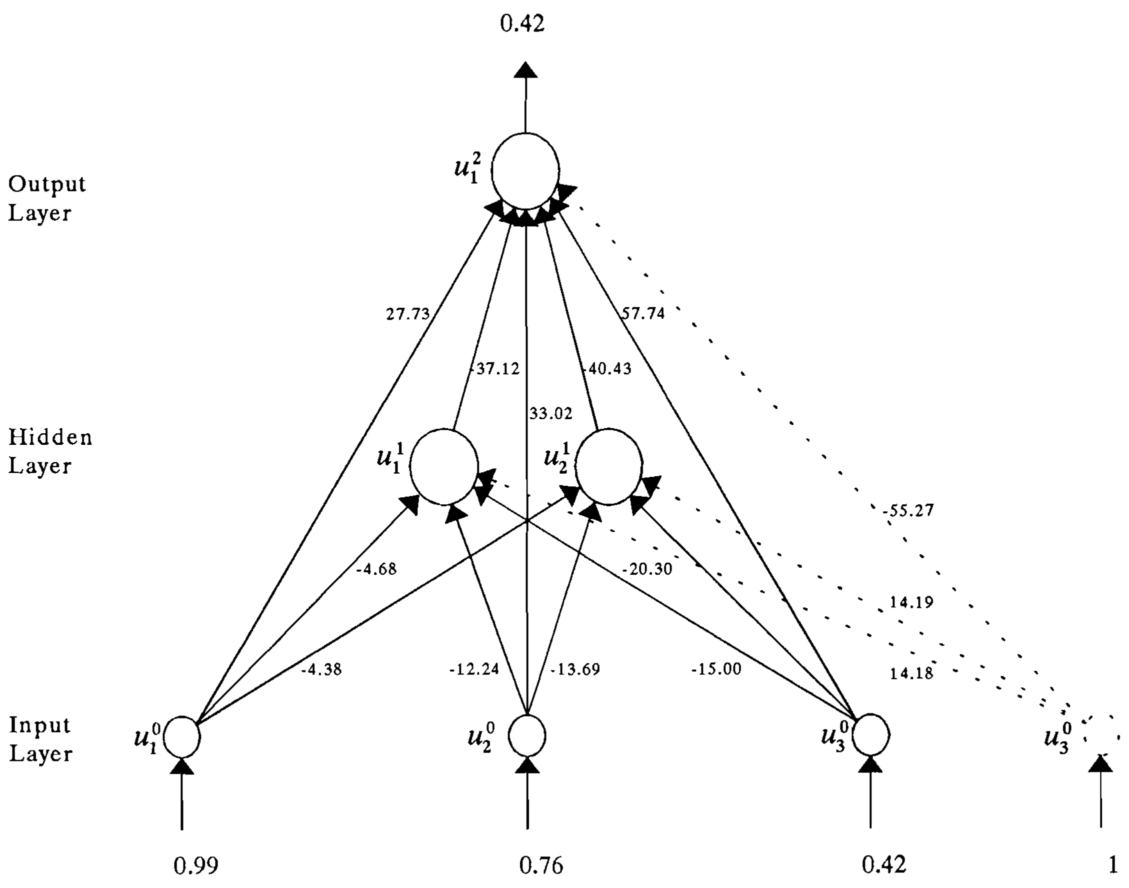

Figure 2: A Example FFANN.