JOURNAL OF PROPULSION AND POWER
Vol. 12, No. 3, May–June 1996

Identification of Parameter Coupling in Turbine Design Using Neural Networks

Sanjay Goel*
General Electric Corporate R&D Center, Schenectady, New York 12301
and
Prabhat Hajela†
Rensselaer Polytechnic Institute, Troy, New York 12180

This article discusses a new technique for improving convergence in optimization problems by pruning the search space of weak variables. Such variables are identified by learning from a database of existing designs using neural networks. By using clustering techniques, different sets of weak variables are identified in different regions of the design space. Parameter sensitivity information obtained in the process of identifying weak variables provides accurate heuristics for formulating design rules. The impact of this methodology on obtaining converged designs has been investigated for turbine design problems. Results from a three-stage power turbine and an aircraft engine turbine design are presented in this article.

Introduction

THE size and complexity of engineering analysis codes have increased steadily over the past few years, motivated primarily by increased computational resources available on desktop machines. Engineering optimization problems in which analysis codes are run iteratively to evaluate the design at multiple points in the search space have become more computation intensive. In fact, an increase in the complexity of analysis codes has kept pace with the improvements to computational resources, necessitating new solution techniques that reduce the complexity of problems that need to be solved iteratively.

A number of engineering design problems have nonlinear, multimodal search spaces with a large number of design variables and constraints. When using gradient-based optimization methods, there is a propensity for the search process to converge to the nearest relative optimum. When using stochastic global search techniques, the computational expense tends to be quite significant. These problems may be ameliorated by using problem-specific information to understand the search space better. Human designers often use such problem-specific knowledge at an abstract plane; however, both the information and its use are often imprecise, incomplete, and hard to quantify. There is an important role, therefore, for machine learning paradigms in the context of engineering design. The fundamental problem in all such models is to extract useful information from a given problem domain most efficiently. Induction is a derivation of general laws by examining particular instances.[1] The process of induction can also be viewed as a search through an abstract plane of potential rules and descriptions.[2] Most inductive knowledge generators use domain-independent techniques for detecting patterns and are too inefficient to be practical for complex problems with large search spaces. A significant amount of work has also been done in the area of case-based reasoning, in which learning is achieved by storing examples for future retrieval. When solving a new problem, a stored case most similar to the current case is retrieved and generalization is done based on local transformations.[3] To obtain a meaningful generalization, an enormous amount of data needs to be stored for a large-dimension problem; therefore, storing, matching, and retrieving data have significant costs associated with case-based reasoning.

Learning techniques such as neural networks and regression analysis can be employed to extract salient characteristics of the search space from existing data. Parametric dependencies in different regions of the design space are highly relevant during design optimization. Such information, if available a priori, can be used to reduce the number of executions of expensive analysis codes required in the process. During optimization, the design is analyzed at multiple design points. These evaluations are used locally to determine search directions and the data are subsequently discarded. It is desirable to learn from this data in real time to improve the convergence rate in subsequent iterations of the optimization process. A simpler problem that is attempted in the present work focuses on the problem of learning knowledge from data generated prior to its use in optimization. This article describes an approach based on neural networks that allows learning important features of a design space from a database of existing designs. In particular, the focus of the present work is to use learning techniques to identify and prune the search space of weak design variables. Subsequent sections of this article describe this learning approach and demonstrate its effectiveness in an automated design environment. The design of engine and power turbines is used as a test bed for this demonstration.

Neural Network Based Learning

Figure 1 shows a schematic diagram of a feed-forward neural network with N layers of neurons, each containing n_i nodes, where i is the layer index. The first layer of neurons is referred to as the input layer, to which components of the input vector are presented. The output corresponding to the input vector is available at the last layer of neurons. The neurons are interconnected as shown in Fig. 1. The strength of these interconnections is referred to as network weights, and w_{ij}^K denotes the strength of connection between the ith and the jth neuron of the Kth layer.

The outputs of nodes from each layer are input to nodes in the subsequent layer; these outputs are weighted by the inter-

Received March 2, 1995; revision received Dec. 4, 1995; accepted for publication Dec. 4, 1995.

*Engineer, Fluid Mechanics Program. Member AIAA.

†Professor, Mechanical Engineering, Aeronautical Engineering, and Mechanics. Associate Fellow AIAA.

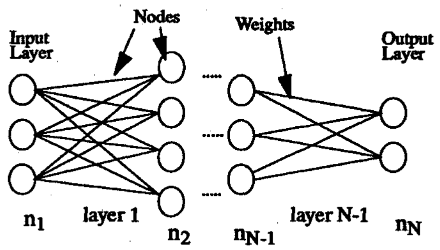

Fig. 1 Schematic diagram of a neural network.

connection weights that either amplify or attenuate the signal. Except for the input layer nodes, the net input to each node is the sum of the weighted output of the nodes of the prior layer. Each node processes this input in accordance with an activation function; a sigmoid function of the type $1/[1 + \exp(-x)]$, where x is the weighted sum of inputs, is widely used, and is adopted in the present work. In some cases, bias ϕ is associated with each neuron, and here the modified sigmoid function takes the form $1/[1 + \exp(-x + \phi)]$.

At the start of network training, the weights are randomly initialized. The network is then exposed to a set of training data that includes a number of input vectors and the corresponding outputs. As the training proceeds, the weights are incrementally adjusted until the error between the network-predicted output and the actual known output for all patterns is reduced below a threshold value. In each adjustment, weights leading to the output nodes are adjusted in proportion to the difference between the output node's predicted output and its desired or target output; the weights leading into hidden nodes are in turn adjusted in proportion to their contributions to the higher nodes until the training is complete. At the end of the training phase, the weights are frozen; then, inputs for which the outputs are not known can be fed into the stabilized network for characterization.[4]

A trained neural network can map the relationship between a set of input parameters and output parameters. Useful information to accelerate the design optimization process can be learned from this mapping, and the acquisition of this information and its use in the design process are discussed in the subsequent sections.

The search space for a problem is a mapping between some input and output parameters. The mapping is defined by a set of governing equations, which if invertible, could facilitate the solution to the optimization problem in a single step. Sobieski,[5] Novak,[6] and Giles[7] provide simple illustrations of the inverse approach. However, because of the sheer complexity of obtaining inverse solutions, this approach is feasible only for simple problems. In more complex problems, such solutions are obtained iteratively, using analysis models for evaluating each candidate solution; optimization algorithms are commonly used for directing such an iterative search process.

Optimization algorithms direct search in that region of the design space where problem-specific knowledge (e.g., gradients, heuristics) indicates the existence of a better design. The performance of these techniques can be improved by pruning the search space. One way of pruning the search space is to compute parameter sensitivities, which can be used for identifying and eliminating the weak parameters in the search space.

The mapping in a neural network is stored in a set of interconnection weight matrices and lends itself to mathematical manipulations that are required to obtain the inverse map. Since the mapping is generated over a region of the design space, both the direct and inverse mappings are an average of that entire region. This mapping can vary significantly over the entire search space of interest, and it is important to formulate a rational approach to define discrete regions within which such information may be meaningful. Clustering algorithms can be used to divide the search space into such finite regions where a particular causality relation can be identified. Clustering of data is particularly meaningful when using neural networks, because it may not be feasible to accurately train a network over a very diverse search space. To adequately capture the effect of nonlinearities and to reduce the training complexity, the data can be organized into clusters, and a neural network can be used to map the data in each cluster. Since only a single level of classification is required, partitional clustering is sufficient for the problem.[8]

Fukunaga and Short[4] have used clustering for problem localization whereby a simple decision rule can be implemented in local regions of the pattern space. The crucial problem in identifying clusters in data is to specify what proximity is, and how to measure it. The euclidean distance Minkowski metric[9] has been selected as the proximity index for the current problem. The membership of a new point is determined as the cluster with the minimum distance from cluster center to the point.

For an adequate representation of the search space, the data used to generate the input-output mappings should be distributed uniformly over the entire space. It would be beneficial if data generated in previous optimization studies could be used for this purpose. Numerical gradient-based optimization and heuristic search techniques generate data that is generally clustered around regions of local optima, whereas stochastic techniques such as genetic algorithms (GA) (Ref. 10) generate data that is more uniformly distributed over the search space. In genetic algorithms, the data develop a bias toward better designs after multiple generations of evolution. However, the bias is very small compared to the data from gradient-based optimization and heuristic search methods, and can be reduced further by either increasing the population size in the GA search or by considering data only from the first few generations of evolution. To avoid erroneous inferences from a strongly biased data, the present work was based on data generated during genetic algorithm-based optimization. Infeasible data points were included in the analysis because of a general requirement for tradeoff analysis at search boundaries. The data were stored as a set of input–output vector pairs and were normalized prior to their use in training a neural network. The following section describes the analysis of the weights of the trained network to extract useful information for use in the optimization process.

Weight Analysis of Neural Networks

Hajela and Szewczyk[11] have proposed a scheme for analyzing the weights of a single hidden layer neural network for predicting relative strengths of parameter interactions, and they have used this information for decomposing large-scale structural design problems.[12] In that analysis the signs of the weights were not taken into consideration, and only a figure of the relative strength of dependencies was available. The weight analysis scheme proposed in this scheme is simpler, more accurate, and provides additional directional information about parameter sensitivities. In addition, this scheme has the flexibility of accommodating weights from a multilayer feed-forward network.

Equation (1) describes this alternative process of calculating the relationships between input and output quantities based on the weights of a backpropagation neural network. This equation represents an averaged sensitivity matrix R (hereafter referred to as the sensitivity matrix), obtained by sequentially multiplying the weight matrices of the different layers of the trained backpropagation network,

$$[R]_{n_0\times n_N} = [W]_{n_0\times n_1} \times [W]_{n_1\times n_2} \ldots \times [W]_{n_{n-1}\times n_N} \tag{1}$$

where $[W]_{n_i\times n_{i+1}}$ are the matrix-containing interconnection weights between layers i and $i + 1$, and n_i is the number of neurons in the ith layer of the network.

In the sensitivity matrix the output and input quantities are represented by the columns and by the rows, respectively. The characterization of a design variable as being dominant in a region of the search space is relative to the other design variables. The sensitivity matrix is normalized with respect to the input variables. Each column of the matrix is independently normalized by the largest absolute value of sensitivity in the column. The most dominant parameter has a sensitivity of either +1 or −1, and the other parameters have sensitivities in the range [−1 to 1]:

$$T_{ij} = \left[\frac{R_{ij}}{\max\limits_{i=1}^{n_1} |R_{ij}|} \right] \quad \text{where} \quad i = 1 \ldots n_1 \qquad j = 1 \ldots n_N \tag{2}$$

The elements of the normalized sensitivity matrix T_{ij} contain information of the relative strength as well as the sign of the parametric dependency. Sensitivities to the performance parameter are used for improving the design. Similarity of sensitivities of the constraints are used for constraint satisfaction and to determine the feasibility of design changes during optimization.

Turbine Preliminary Design

The objective of the turbine preliminary design is to maximize the turbine performance while all constraints are satisfied. During the design process, the flow-path and the free vortex vector diagrams are modified to improve the design. There are three sets of variables in this problem that pertain to geometry, solidity, and work distribution.[13]

Since this preliminary design problem does not have a closed-form solution, it is solved iteratively by the use of optimization algorithms. The complexity of the search problem increases because of the presence of a large number of design variables. Different subsets of design variables are dominant in different regions of the search space, and the complexity of the problem can be reduced by identifying and retaining the dominant variables, while eliminating the weak variables. In the present approach, the set of active design variables changes dynamically as the search moves through different regions of the design space.

A rule-based search coupled with numerical optimization that has been used for turbine design optimization[14] was also used in the present work. In the past, information from a design engineer was used to generate design rules, which remained static throughout the search process. The current approach uses data generated in prior optimization runs to develop rules that dynamically change as the search progresses. To facilitate this process, a weight analysis was performed on all networks trained on clustered data to determine dominant parameters in each cluster. Rules were then generated using the dominant parameters and were assigned a priority based on their relative strengths.

Goal: Increase Efficiency
Weight: 1.0
Conditions: Current-state ε Cluster1
Actions: Increase Stage3-Blade_Zwiefel
Decrease Stage2-Blade_Zwiefel
Decrease Stage2-Reaction
Increase Stage1-Blade_Zwiefel

Goal: Increase Stage3-Swirl
Weight: 1.0
Conditions: Current-state ε Cluster1
Actions: Decrease Stage3-Blade_Zwiefel
Decrease Stage3-Vane_Zwiefel
Increase Stage2-Blade_Zwiefel

Fig. 2 Sample rules based on heuristics learned from prior data.

Two sample rules that were generated using the approach described previously are shown in Fig. 2; the first rule is required for improvement of the turbine efficiency, and the second rule is required for satisfying the swirl constraint. These dynamic rules can be used in conjunction with designer-defined heuristics with a higher weighting to the more efficient (dynamic) rules to ensure their preferential selection. Turbine aerodynamic parameters described in the rules such as efficiency, Zwiefel number, swirl, and reaction are defined in Horlock.[15]

Results

Data from the design of a three-stage power turbine and a three-stage aircraft engine turbine were used for investigating the effect of the proposed approach on design optimization. The objective of the design was to maximize the turbine performance. Nine design variables and 12 constraints were used in the problem formulation. The design variables were reaction, blade Zwiefel number, and vane Zwiefel number for each stage of the turbine. The constraints were upper and lower bounds on turning and swirl at the stage exit for each stage. However, for simplicity, the present work focused on the solution of the unconstrained problem.

Power Turbine

Fifteen hundred data points were randomly selected from data generated by a genetic algorithm during optimization. The data were divided into 12 clusters, and six of these clusters, which had enough data points (75–250) to generate meaningful parameter sensitivities, were used in the analysis. Numerous network configurations were evaluated on the basis of the residual error at the end of the network training. The process of network configuration selection was initiated by selecting a network architecture with a large number of hidden nodes (which trains easily but performs poorly on generalization), and then gradually reducing the number of hidden nodes until a satisfactory performance in both training and generalization was attained.

This network (9-24-7) contained 24 hidden nodes in addition to nine input and seven output nodes. The network was considered trained if the network error was reduced below 2%.

Weight analysis was performed on each cluster, and the sensitivities obtained for cluster 1 are shown in Table 1. The rows and the columns represent the input variables and the output response quantities, respectively. In addition to an excellent correlation with known heuristics, a clear separation of the weak and strong variables was obtained from this analysis. As an example, Table 1 shows that the turning in stage 1 is influenced only by the reaction in stage 1, and not by reactions in stages 2 and 3. It should be intuitively clear that the output response should be strongly influenced by upstream flow variables and weakly influenced by downstream variables, since

Table 1 Sensitivity matrix for cluster 1 using proposed approach

Design variables	Efficiency	Stage 1 turning	Stage 2 turning	Stage 3 turning	Stage 1 swirl	Stage 2 swirl	Stage 3 swirl
Stage 1 reaction	−0.217	−1.000	−0.452	−0.051	1.000	−0.492	−0.265
Stage 2 reaction	0	0	−1.000	−0.090	−0.057	0.476	−0.027
Stage 3 reaction	−0.507	0.002	−0.525	−1.000	−0.100	−1.000	1.000
Stage 1 Zwiefel vane	0.116	−0.032	−0.228	−0.081	−0.062	−0.071	0
Stage 2 Zwiefel vane	−0.031	−0.041	0	0	−0.035	−0.229	−0.077
Stage 3 Zwiefel vane	0.082	0.092	0.512	0.003	0.069	0.615	0.301
Stage 1 Zwiefel blade	0.330	0.005	−0.011	0.071	0	0	−0.014
Stage 2 Zwiefel blade	−0.648	−0.037	−0.401	−0.299	−0.022	−0.494	−0.237
Stage 3 Zwiefel blade	1.000	0.081	0.138	−0.039	0.016	0.387	0.463

Table 2 Sensitivity matrix for all clusters showing relationship with efficiency

	Cluster					
Design variables	1	2	3	4	5	6
Stage 1 reaction	−0.217	0.112	0.468	0.212	−0.435	0.533
Stage 2 reaction	0	0	0	0.226	0.724	−0.785
Stage 3 reaction	−0.507	0.921	0.640	0	0.051	−1.000
Stage 1 Zwiefel vane	0.116	−0.341	−0.404	−0.193	0	−0.395
Stage 2 Zwiefel vane	−0.031	−0.609	−0.352	0.109	1.000	0.389
Stage 3 Zwiefel vane	0.082	−0.124	0.562	−0.056	0.369	−0.783
Stage 1 Zwiefel blade	0.330	−0.089	0.529	−0.171	0.458	−0.362
Stage 2 Zwiefel blade	−0.648	−0.623	−1.000	0.118	−0.028	0
Stage 3 Zwiefel blade	1.000	−1.000	−0.596	−1.000	0.910	0.714

Table 3 Sensitivity matrix for all clusters showing relationship with efficiency

	Cluster					
Design variables	1	2	3	4	5	6
Stage 1 reaction	0.274	−0.153	0.537	0.208	0.219	0.0
Stage 2 reaction	−0.047	0.0	0.175	−0.121	0.01	−1.0
Stage 3 reaction	0.330	0.192	0.195	0.344	0.591	0.066
Stage 1 Zwiefel vane	−0.548	−0.283	−0.692	−0.099	−0.604	−0.003
Stage 2 Zwiefel vane	−0.007	0.080	0.0	0.0	0.127	0.175
Stage 3 Zwiefel vane	−0.051	0.063	0.084	0.007	0.0	−0.502
Stage 1 Zwiefel blade	0.0	0.027	−0.228	−0.020	0.231	−0.001
Stage 2 Zwiefel blade	0.082	0.028	0.455	0.303	0.221	−0.005
Stage 3 Zwiefel blade	−1.0	−1.0	−1.0	−1.0	−1.0	−0.036

Goal: To Increase Efficiency
Weight: 1.0
Conditions: Current-state ∈ Cluster1
Actions: Decrease Stg3-Zwib
Increase Stg2-Zwib
Increase Stg3-Rxh
Decrease Stg1-Zwib
Increase Stg1-Rxh

Goal: To Increase Efficiency
Weight: 1.0
Conditions: Current-state ∈ Cluster2
Actions: Increase Stg3−Zwib
Increase Stg2−Zwib
Decrease Stg3−Rxh
Increase Stg2−Zwiv
Increase Stg1−Zwiv

Fig. 3 Rules for power turbine based on heuristics learned from prior data.

the flow moves from upstream to downstream, transporting the changes in upstream conditions with it. Also, turning is reduced by increasing the reaction, which is a well-known heuristic.

A similar weight analysis was performed using the approach proposed by Szewczyk.[10,11] These results were not satisfactory and showed low correlation with known variable dependencies. A clear separation of the dominant variables from the rest of the variables was not obtained in that approach, partly because only absolute values of sensitivities are obtained in that approach, making rule formulation more imprecise.

Table 2 shows sensitivities of turbine efficiency to design variables in different clusters. Significant differences were observed in sensitivity values in different clusters. Moreover, the signs of these sensitivities also change from one cluster to another. This sensitivity information was used for generating design rules that were associated with a particular cluster.

The analysis also shows that stage 3 design variables have a stronger influence on the turbine efficiency compared to design variables of stages 1 and 2. In addition, the stage 2 reaction is near optimum in clusters 1, 2, and 3, and stage 3 reaction is a near optimum in cluster 4.

Sample rules generated for efficiency improvement using the data in Table 2 are shown in Fig. 3. Six rules were generated based on the parameter sensitivities. Each rule corresponds to one cluster. In each rule the variables are sequentially varied in a prioritized order based on their dominance in search. These rules were added to the database of rules used for optimization. To keep these rules simple, quantitative information of the constraint sensitivities was not used for estimating the step size during optimization.

An optimization of the turbine was performed using these rules. A simple one-dimensional search was used where parameters were varied sequentially. During the entire optimization, the design process was focused on the local hill in cluster 4, and hence, only the rule for cluster 4 was used. The optimization performance using rules with the reduced set of variables compared to the original set with all of the design variables is shown in Fig. 4. The solid line shows the optimization using the cluster-based rules generated by the weight analysis. The rule produces a better result in 25% fewer iterations.

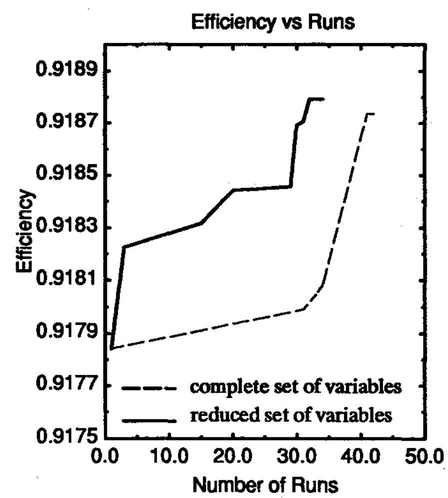

Fig. 4 Comparison of optimization search paths for power turbine.

Aircraft Engine Turbine

A three-stage aircraft engine turbine was also optimized by generating rules based on the new approach. Seven hundred data points were randomly selected from data generated by a genetic algorithm during design optimization. The data were divided into six clusters, each with 33–215 data points. After evaluating numerous configurations, a three-layer 9-10-7 network configuration was found suitable.

Weight analysis was performed on each cluster, and the sensitivity of the turbine efficiency to each design variable in dif-

Table 4 Design changes with new and original rules

Design variables	Starting values	New values: Reduced variable set	New values: Complete variable set
Stage 1 reaction	0.55	0.55	0.55
Stage 2 reaction	0.26	0.26	0.26
Stage 3 reaction	0.15	0.375	0.375
Stage 1 Zwiefel vane	0.6	0.54	0.54
Stage 2 Zwiefel vane	0.1	0.1	0.1
Stage 3 Zwiefel vane	0.9	0.9	0.9
Stage 1 Zwiefel blade	0.85	0.765	0.765
Stage 2 Zwiefel blade	0.65	0.65	0.725
Stage 3 Zwiefel blade	0.897	0.825	0.897
Efficiency	93.02	93.23	93.23

Goal: To Increase Efficiency
Weight: 1.0
Conditions: Current-state ∈ Cluster1
Actions: Decrease Stg3-zwib
Decrease Stg1-zwiv
Increase Stg3-rxh
Increase Stg1-rxh
Decrease Stg1-zwib

Goal: To Increase Efficiency
Weight: 1.0
Conditions: Current-state ∈ Cluster
Actions: Decrease Stg3−Zwib
Decrease Stg1−Zwiv
Increase Stg3−Rxh
Decrease Stg1−Rxh

Fig. 5 Rule for engine based on heuristics learned from prior data.

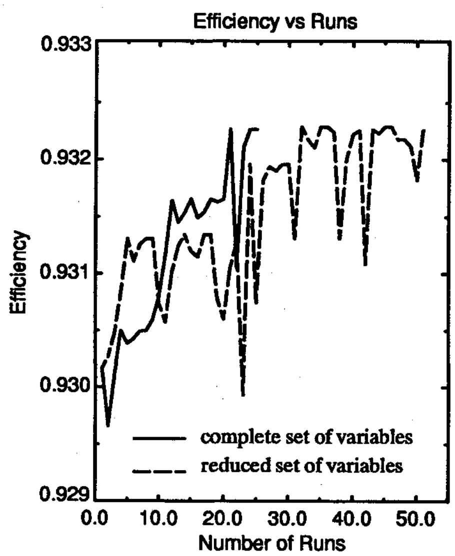

Fig. 6 Comparison of optimization search paths for engine turbine.

ferent clusters is shown in Table 3. The sensitivity matrix shows clear separation of strong and weak variables in each cluster. Sample rules generated using this sensitivity information are shown in Fig. 5.

The turbine was optimized using rules with a reduced set of variables and was compared with results obtained using the original set of rules and the complete set of design variables as shown in Fig. 6. The solid line shows the search path using rules with a reduced set of variables and the dashed line shows the search path with the original set of rules. The same efficiency gain was obtained with both approaches. However, the number of iterations was reduced by half when using rules based on a reduced set of variables. The search path using rules based on the reduced variable set contained fewer unproductive steps compared to optimization with the original set of rules.

Table 4 shows the changes in design variables in each of the two cases. The first column contains the starting values. The second column contains the new values of the design variables using rules based on the reduced variable set, and the third column contains the new values of the variables using the original rules. Although the two solutions are slightly different, similar performance improvements were obtained in each case. During optimization, the search was confined to cluster 1 for a majority of the iterations.

To investigate the generality of the rules using a reduced variable set, rules for the power turbine were used to optimize the aircraft engine turbine. Even though the configuration of the two turbines was identical, the rules from the power turbine were found to be completely ineffective in optimizing the aircraft engine turbine and, in fact, resulted in a poorer design. This exercise illustrates effectively the differences in the design spaces for the engine and the power turbines. The implication from this exercise suggests that new sensitivities need to be generated for each new case until criteria are established for identifying turbines that can be represented by the same search space and can be optimized using the same set of rules.

Conclusions

A new approach has been developed through which weak variables in a design space are identified and eliminated as a means of reducing design problem complexity. A neural network-based learning from existing data is a central feature of the approach and yields global or averaged sensitivities of the objective function and the constraint with respect to the design variables. These sensitivities can be used to eliminate weak variables from the search space and to formulate design rules that promote the use of fewer variables during optimization.

This methodology has been demonstrated for unconstrained preliminary design of a power turbine and an aircraft engine turbine. A 25–50% reduction in the number of iterations and a small improvement in the performance of one of the turbines were obtained from the optimization. The approach can be extended to a constrained design problem by including in the reduced variable set those design variables that are most effective in constraint satisfaction using those variables to create rules that satisfy constraints.

The reduction in the number of iterations becomes very critical in optimization problems that use complex and computationally intensive analysis codes. In the current problem a one-dimensional rule-based search was used because there are insufficient design heuristics available for creating n-dimensional rules. N-dimensional rules could be very useful for constrained search because they can be used to improve performance while keeping track of the constraint boundaries. This approach is currently under investigation and will be reported in forthcoming publications.

Acknowledgments

The first author would like to thank the General Electric Corporate R&D Center for their support in this work. Partial support for this work received under Grant NAG-3-1196 from NASA is gratefully acknowledged.

References

[1]Forsyth, R., *Machine Learning Principles and Techniques*, Chapman and Hall, New York, 1989.

[2]Mitchell, T., Keller, R., and Kedar-Cabelli, S., "Explanation-Based Generalization: Unifying View," *Machine Learning*, Vol. 1, No. 1, 1986, pp. 47–80.

[3]Minton, S., *Learning Search Control Knowledge—An Explanation-Based Approach*, Kluwer Academic, Norwell, MA, 1988.

[4]Fukunaga, K., and Short, R. D., "Generalized Clustering for Problem Localization," *IEEE Transactions on Computers*, Vol. C-27, No. 2, 1978, pp. 176–181.

[5]Jain, A. K., and Duber, R. C., *Algorithms for Clustering Data*, Prentice-Hall, Englewood Cliffs, NJ, 1988.

[6]Sobieczky, H., Yu, N. J., Fung, K-Y., and Seebass, A. R., "New Method for Designing Shock-Free Transonic Configuration," *AIAA Journal*, Vol. 17, No. 7, 1979, pp. 722–729.

[7]Novak, R. D., and Haymann-Haber, G., "A Mixed-Flow Cascade Passage Design Procedure Based on a Power Series Expansion," *Journal of Engineering for Power*, Vol. 105, No. 2, 1983, pp. 231–242.

[8]Giles, M., Drela, M., and Thompkins, W., "Newton Solution of Direct and Inverse Transonic Euler Equations," *Proceedings of the AIAA 7th Computational Fluid Dynamics Conference* (Cincinnati, OH), AIAA, New York, 1985, pp. 390–402.

[9]Chester, M., *Neural Networks—A Tutorial*, Prentice–Hall, Englewood Cliffs, NJ, 1993.

[10]Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison–Wesley, Reading, MA, 1989.

[11]Hajela, P., and Szewczyk, Z., "Neurocomputing Strategies in Structural Design—On Analyzing Weights of Feed Forward Neural Networks," *Structural Optimization*, Vol. 8, No. 4, 1994, pp. 236–241.

[12]Szewczyk, Z., "Neurocomputing Based Approximate Models in Structural Analysis and Design," Ph.D. Dissertation, Rensselaer Polytechnic Inst., Troy, NY, May 1993.

[13]Goel, S., Gregory, B., and Cherry, D., "Knowledge-Based System for the Preliminary Aerodynamic Design of Aircraft Engine Turbines," *Proceedings of the SPIE—The International Society for Optical Engineering*, Vol. 1963, 1993, pp. 160–170.

[14]Powell, D. J., Skolnick, M. M., and Tong, S. S., "Interdigitation: A Hybrid Technique for Engineering Design Optimization Employing Genetic Algorithms, Expert Systems, and Numerical Optimization," *Handbook of Genetic Algorithms*, edited by L. Davis, Van Nostrand Reinhold, New York, 1991, pp. 312–331.

[15]Horlock, J. H., *Axial Flow Turbines, Fluid Mechanics and Thermodynamics*, Krieger, Malabar, FL, 1966.