Optimization of Large-Scale 3-D Trusses Using Evolution Strategies and Neural Network

ARTICLE *in* INTERNATIONAL JOURNAL OF SPACE STRUCTURES · SEPTEMBER 1999

DOI: 10.1260/0266351991494830

3 AUTHORS:

Papadrakakis Manolis
National Technical University of Athens
249 PUBLICATIONS **2,792** CITATIONS
SEE PROFILE

Nikos Lagaros
National Technical University of Athens
129 PUBLICATIONS **1,298** CITATIONS
SEE PROFILE

Yiannis Tsompanakis
Technical University of Crete
65 PUBLICATIONS **357** CITATIONS
SEE PROFILE

OPTIMIZATION OF LARGE-SCALE 3-D TRUSSES USING EVOLUTION STRATEGIES AND NEURAL NETWORKS

Manolis Papadrakakis*, Nikos D. Lagaros, Yiannis Tsompanakis

Institute of Structural Analysis & Seismic Research,
National Technical University Athens,
Zografou Campus, Athens 15773,
Greece

*To whom correspondence should be addressed

ABSTRACT

The objective of this paper is to investigate the efficiency of optimization algorithms, based on evolution strategies, for the solution of large-scale structural optimization problems. Furthermore, the structural analysis phase is replaced by a neural network prediction for the computation of the necessary data for the ES optimization procedure. The use of NN was motivated by the time-consuming repeated analyses required by ES during the optimization process. A back propagation algorithm is implemented for training the NN using data derived from selected analyses. The trained NN is then used to predict, within an acceptable accuracy, the values of the objective and constraint functions. The proposed methodology has been applied in sizing structural optimization problems of large-scale three dimensional roof trusses. The numerical tests presented demonstrate the computational advantages of the proposed approach which become more pronounced for large-scale optimization problems.

Keywords: structural optimization, evolution strategies, neural networks.

1. INTRODUCTION

The sizing optimization of large-scale three dimensional trusses which are often used to cover wide span column-free areas is a computationally intensive task. In sizing optimization the aim is to minimize the weight of the structure under certain restrictions imposed by design codes. When a gradient-based optimizer is used the most time-consuming part of the optimization process is devoted to the sensitivity analysis phase which is an important ingredient of all mathematical programming optimization methods [1,2]. On the other hand the application of probabilistic search methods, such as evolution strategies (ES), do not need gradient information and therefore avoid to perform the computationally expensive sensitivity analysis step [3,4]. Furthermore, it is widely recognized that probabilistic search optimization techniques are in general more robust and present a better global behaviour than mathematical programming methods. They may suffer, however, from a slow rate of convergence towards the global optimum.

During the last fifteen years there has been a growing interest in problem solving systems based on algorithms which rely on analogies to natural processes. The best known algorithms in this class include evolutionary programming (EP) [5], genetic algorithms (GA) [6,7] and evolution strategies (ES) [8,9]. Another important technique that follows natural processes, and in particular human brain functions, is artificial neural networks which simulate the structure of the biological neural network of the human brain. The use of artificial intelligence techniques, such as neural networks, to predict analysis outputs has been studied previously in the context of optimal design of structural systems [10,11], and also in some other areas of structural engineering applications. In the review papers of Berrais [12] and Waszczyszyn [13] a number of references can be found on the application of neural networks (NN) in computational mechanics. The principal advantage of a properly trained NN is that it requires a trivial computational effort to produce an acceptable approximate solution. Such approximations appear to be valuable in situations where the actual response computations are intensive in terms of computing time and a quick estimation is required.

In this work the efficiency of ES combined with NN in sizing structural optimization problems is investigated in an effort to increase the robustness as well as the computational efficiency of the optimization procedure. The use of NN was motivated by the time-consuming repeated analyses required for ES during the optimization process. The suitability of NN predictions is investigated in a number of structural problems optimized using ES and the computational advantages of the proposed approach are demonstrated. In addition a thorough investigation is performed on the selection of the training set used for the NN learning procedure in order to ensure the generality and robustness of the proposed methodology. In a recent study [4] a similar methodology was proved very efficient in sizing optimization of multi-storey 3-D frames and shape optimization of 2-D mechanical parts.

For each optimization problem a NN is trained utilizing information generated from a number of properly selected analyses. The data from these analyses are processed in order to obtain the necessary input and output pairs which are subsequently used to produce a trained NN. The trained NN is then used to predict the response of the structure in terms of objective and constraints function values due to different sets of design variables. The predicted values of the optimization functions should resemble closely to the corresponding values of the conventional analyses, which are considered exact. The NN type considered here is based on the feed-forward architecture trained by the algorithm known as Back-propagation [14].

2. SIZING OPTIMIZATION

In sizing optimization problems the aim is usually to minimize the weight of the structure under certain behavioural constraints on stresses and displacements. The design variables are most frequently chosen to be dimensions of the cross-sectional areas of the members of the structure. Due to engineering practice demands the members are divided into groups having the same design variables. This linking of elements results in a trade-off between the use of more material and the need of symmetry and uniformity of structures due to practical considerations. Furthermore, it has to be to taken into account that due to fabrication limitations the design variables are not continuous but discrete since cross-sections belong to a certain set.

A discrete structural optimization problem can be formulated in the following form:

$$\begin{aligned} &\text{min} && F(s) \\ &\text{subject to} && g_j(s) \le 0 \quad j = 1,\ldots,m \\ & && s_i \in R^d, \quad i = 1,\ldots,n \end{aligned} \tag{1}$$

where F(s) and g(s) denote the objective and constraints functions respectively. R^d is a given set of discrete values and design variables s_i (i=1,...,n) can take values only from this set. In the present study the sizing optimization of large-scale 3-D trusses is investigated. These type of structures is very common in engineering practice in order to cover long and/or wide span and column-free spaces such as stadiums, exhibition halls, airplane hangars, etc. The performance of these type of structures has been investigated in terms of economy, structural safety, aesthetic quality and optimum design in a number of papers [15-19].

Space truss structures usually have the topology of single or multi-layered flat or curved grids that can be easily constructed in practice. Most frequently the objective function is the weight or the volume of the structure and the constraints are the member stresses, nodal displacements, or frequencies. The stress constraints can be written as $|\text{ó}| \le |\text{ó}_a|$, where $\text{ó} = \frac{\text{Ñ}}{\text{Á}}$ is the maximum axial stress in each element group for all loading cases, σ_a=0.60*F_y is the allowable axial stress and F_y is the yield stress and A is the cross-sectional area. Similarly, the displacement constraints can be written as $|d| \le d_a$, where d_a is the limiting value of the displacement at a certain node, or the maximum nodal displacement.

Euler buckling occurs in truss structures when the magnitude of a member's compressive stress is greater than a critical stress which, for the first buckling mode of a pin-connected member, is equal to

$$\sigma_b = \frac{P_b}{A} = -\frac{1}{A}\left(\frac{\pi^2 EI}{L^2}\right) \tag{2}$$

where P_b is the computed compressive axial force, I is the moment of inertia, L is the member length. For thin-walled tubular members with a diameter-to-thickness ratio ñ=D/t=10-20 the cross-sectional area is approximately equal to $A \cong \pi Dt$, and the moment of inertia is approximated by $I \cong \frac{\pi Dt(D^2 + t^2)}{8}$. The expression for the buckling stress can, therefore, be written as a function of the cross-sectional areas, which are the design variables of the optimization problem, as follows

$$ó_b = -\frac{AE}{L^2} \cdot \frac{\pi(ñ^2 + 1)}{8ñ} \tag{3}$$

Thus, the compressive stress should be less (in absolute values) than the critical Euler buckling stress $|ó| \le |ó_b|$. The values of the constraint functions are normalized in order to improve the performance of the optimization procedure as

$$\sigma/\sigma_a \le 1 \text{ for tension member } \sigma_a = 0.6 \cdot F_Y$$
$$\sigma/\sigma_b \le 1 \text{ for compression member } \sigma_b = E(\pi/(l/r))^2$$
$$d/d_a \le 1$$

The sizing optimization methodology proceeds with the following steps: (i) At the outset of the optimization the geometry, the boundaries and the loads of the structure under investigation have to be defined. (ii) The design variables, which may or may not be independent to each other, are also properly selected. Furthermore, the constraints are also defined in this stage in order to formulate the optimization problem as in eq. (1). (iii) A finite element analysis, is then carried out and the displacements and stresses are evaluated. (iv) If a gradient-based optimizer, like the sequential quadratic programming (SQP) algorithm, is used then the sensitivities of the constraints and the objective function to small changes of the design variables are computed either with the finite difference, or with the semi-analytical method. (v) The design variables are being optimized. If the convergence criteria for the optimization algorithm are satisfied, then the optimum solution has been found and the process is terminated, else the optimizer updates the design variable values and the whole process is repeated from step (iii).

3. EVOLUTION STRATEGIES (ES)

Evolution strategies were proposed for parameter optimization problems in the seventies by Rechenberg [8] and Schwefel [9]. ES imitate biological evolution in nature and have three characteristics that make them differ from other conventional optimization algorithms: (i) in place of the usual deterministic operators, they use randomized operators: mutation, selection as well as recombination; (ii) instead of a single design point, they work simultaneously with a population of design points in the space of variables; (iii) they can handle continuous, discrete or mixed optimization problems. The second characteristic allows for a natural implementation of ES on parallel computing environments. The ES, however, achieve a high rate of convergence than GA due to their self-adaptation search mechanism and are considered more efficient for solving real world problems [20]. In the case of ES no difference exists between genotype (encoding) and phenotype (appearance). The ES were initially applied for continuous optimization problems, but recently they have also been implemented in discrete and mixed optimization problems. The ES algorithms used in the present study are based on the work of Thierauf and Cai who applied the ES methodologies in sizing structural optimization problems having discrete and/or continuous design variables [21,22].

3.1 ES for discrete optimization problems

In engineering practice the design variables are not continuous because usually the structural parts are constructed with certain variation of their dimensions. Thus design variables can only take values from a predefined discrete set. For the solution of discrete optimization problems a modified ES algorithm has been proposed by Thierauf and Cai [21]. The basic differences between discrete and continuous ES are focused on the mutation and the recombination operators. The mutation operator ensures that each parent $s_p^{(g)}$ of the current generation g produces an offspring $s_o^{(g)}$, whose genotype is slightly different from that of the parent:

$$s_o^{(g)} = s_p^{(g)} + z^{(g)} \tag{4}$$

where $z^{(g)} = \left[z_1^{(g)}, z_2^{(g)}, \ldots, z_n^{(g)}\right]^T$ is a random vector. The mutation operator in the continuous version of ES produces a normally distributed random change vector $z^{(g)}$. Each component of this vector has small standard deviation value $ó_i$ and zero mean value. As a result of this there is a possibility that all components of a parent vector may be changed, but usually the changes are small. In the discrete version of ES the random vector $z^{(g)}$is properly generated in order to force the offspring vector to move to another set of discrete values.

The fact that the difference between any two adjacent values can be relatively large is against the requirement that the variance $ó_i^2$ should be small. For this reason it is suggested that not all the components of a parent vector, but only a few of them (eg. ℓ) should be randomly changed in every generation. This means that n-ℓ components of the randomly changed vector $z^{(g)}$ will have zero value. In other words, the terms of vector $z^{(g)}$ are derived from

$$z_i^{(g)} = \begin{cases} (\hat{e} + 1)\ddot{a}s_i & \text{for } \ell \text{ randomly chosen components} \\ 0 & \text{for n - } \ell \text{ other components} \end{cases} \qquad (5)$$

where $\ddot{a}s_i$ is the difference between two adjacent values in the discrete set and ê is a random integer number which follows the Poisson distribution

$$p(\hat{e}) = \frac{(\tilde{a})^{\hat{e}}}{\hat{e}\,!} e^{-\tilde{a}} \qquad (6)$$

ã is the standard deviation as well as the mean value of the random number ê. The choice of ℓ depends on the size of the problem and it is usually taken as the 1/5 of the total number of design variables. The ℓ components are selected using uniform random distribution in every generation according to eq. (5).

In both versions, continuous and discrete, of multi-membered ES there are two different types of selection:

(ì+ë)-ES: The best *ì* individuals are selected from a temporary population of (*ì*+*ë*) individuals to form the parents of the next generation.

(ì,ë)-ES: The *ì* individuals produces *ë* offsprings (*ì*≤*ë*) and the selection process defines a new population of *ì* individuals from the set of *ë* offsprings only.

In the second type, the life of each individual is limited to one generation. This allows the (ì,ë)-ES selection to perform better on dynamic problems where the optimum is not fixed, or on problems where the objective function is noisy [23].

For discrete optimization the procedure terminates when one of the following heuristic criteria is satisfied [21,22]: (i) when the best value of the objective function in the last 4∗n∗ì/ë generations remains unchanged, (ii) when the mean value of the objective values from all parent vectors in the last 2∗n∗ì/ë generations has not been improved by less than a given value å$_b$ (=0.0001), (iii) when the relative difference between the best objective function value and the mean value of the objective function values from all parent vectors in the current generation is less than a given value å$_c$(=0.0001), (iv) when the ratio ì$_b$/ì has reached a given value å$_d$ (=0.5 to 0.8) where ì$_b$ is the number of the parent vectors in the current generation with the best objective function value.

3.2 ES in structural optimization problems

So far comparatively little effort has been spent in applying probabilistic search methods to structural optimization problems [3,4,24]. Usually this type of problems are solved with a mathematical programming algorithm such as the sequential quadratic programming method SQP [25], the generalized reduced gradient method (GrG) [26], the method of moving asymptotes (MMA) [27], which need gradient information. In structural optimization problems, where the objective function and the constraints are particularly highly non-linear functions of the design variables, the computational effort spent in gradient calculations is usually large.

In two recent studies by Papadrakakis et. al. [3,4] it was found that probabilistic search algorithms are computationally efficient even if greater number of analyses are needed to reach the optimum. These analyses are computationally less expensive than in the case of mathematical programming algorithms since they do not need gradient information. Furthermore, probabilistic methodologies were found, due to their random search, to be more robust in finding the global optimum, whereas mathematical programming algorithms may be trapped in local optima. Finally, the natural parallelism inherent in probabilistic search algorithms makes them very attractive for application in parallel computer architectures.

The ES optimization procedure starts with a set of parent vectors and if any of these parent vectors gives an infeasible design then this parent vector is modified until it becomes feasible. Subsequently, the offsprings are generated and checked if they are in the feasible region. According to (ì+ë) selection scheme in every generation the values of the objective function of the parent and the offspring vectors are compared and the worst vectors are rejected, while the remaining ones are considered to be the parent vectors of the new generation. On the other hand, according to (ì,ë) selection scheme only the offspring vectors of each generation are used to produce the new generation. This procedure is repeated until the chosen termination criterion is satisfied.

The computational efficiency of the multi-membered ES discussed in this work is affected by the number of parents and offsprings involved. It has been observed that values of ì and ë equal to the number of the design variables produce best results [3]. The ES algorithm for structural optimization applications can be stated as follows :

1. *Selection step :*
   selection of s_i (i = 1,2,...,ì) parent vectors of the design variables
2. *Analysis step :* solve $K(s_i)u_i = f$ (i=1,2,...,ì)
3. *Constraints check :* all parent vectors become feasible
4. *Offspring generation :*
   generate s_j, (j=1,2,...,ë) offspring vectors of the design variables
5. *Analysis step :* solve $K(s_j)u_j = f$ (j=1,2,...,ë)
6. *Constraints check :*
   if satisfied continue, else change s_j and go to *step 4*

7. *Selection step :*
   selection of the next generation parents according to (ì+ë) or (ì,ë) selection schemes
8. *Convergence check :* If satisfied stop, else go to *step 3*

4. ELEMENTS OF ARTIFICIAL NEURAL NETWORKS THEORY

The aim of the present study is to train a neural network to provide computationally inexpensive estimates of analysis outputs required during the optimization process. A trained network presents some distinct advantages over the numerical computing paradigm. It provides a rapid mapping of a given input into the desired output quantities, thereby enhancing the efficiency of the redesign process. This major advantage of a trained NN over the conventional procedure, under the provision that the predicted results fall within acceptable tolerances, leads to results that can be produced in a few clock cycles, representing orders of magnitude less computational effort than the conventional computational process. The learning algorithm which was employed for the training is the well known Back Propagation (BP) algorithm [14,28].

4.1 The NN training

In the present implementation the objective is to investigate the ability of the NN to predict accurate structural analysis outputs that are necessary for the ES optimizer. This is achieved with a proper training of the NN. The NN training comprises the following tasks: (i) select the proper training set, (ii) find a suitable network architecture and (iii) determine the appropriate values of characteristic parameters such as the learning rate and momentum term.

The main limitation of a NN training algorithm is the fact that its efficiency depends on the learning rate, momentum term and network architecture. Unfortunately, there are not any general rules on the selection of these parameters. They usually depend on the type of the problem and the experience of the designer and sometimes they can be chosen with a trial and error approach. The appropriate selection of I/O training data is one of the most important features of NN training. The number and the distribution of training patterns is of great importance especially for the BP algorithm which provides good results only if the training set includes data over the entire range of the design space.

The output of the sigmoid function used in the conventional BP algorithm lies between 0 and 1, to produce meaningful results the output values of the training patterns should be scaled within the same range. As the network is trained, the weights can become adjusted to very large values. This can force all or most of the neurons to operate at large output values in a region where the derivative of the activation function is very small. Since the correction of the weights depends on the derivative of the sigmoid function the network may come to virtual standstill. Initializing the weights to small random values would help to avoid this situation, however it is more appropriate to normalize the input patterns to be also between 0 and 1.

The learning rate coefficient and the momentum term are two user defined BP parameters that affect the training of NN. The learning rate coefficient, employed during the adjustment of weights, can speed-up or slow-down the learning process. A bigger learning coefficient increases the weight changes, hence large steps are taken towards the minimum error level but may lead to oscillation, while smaller learning coefficients increase the number of steps taken to reach the desired error level and may trap the process at a local optimum. If an error curve shows a downward trend but with poor convergence at the latest stages a decrease of the learning rate coefficient is likely to accelerate the convergence.

The momentum term α when used with batch training affects the learning procedure in such a way that if the gradient points lie at the same direction for several consecutive iterations, it increases the step size by a factor of approximately 1/(1-α). Therefore, a momentum term close to one is useful when a small learning rate is used. However, a large momentum term can exacerbate divergence problems when a large learning rate is used. In this study the initial values for the momentum term and the learning rate are taken equal to 0.6 and 0.05, respectively. Then, when the error curve starts to oscillate the value of the learning rate coefficient is divided by two. If a flat error curve is obtained the value of the momentum term is increased to the value 0.99 while the value of the learning rate coefficient remains unchanged.

The basic NN configuration employed is fully connected with two hidden layer. Tests performed for more than one hidden layer showed no significant improvement in the obtained results. Based on this configuration various NN architectures are tested in order to find the most suitable one in terms of the smallest prediction error. This is done either with a direct comparison of predicted with "exact" results, or by means of the Root Mean Square (RMS) error which is given by

$$e_{RMS} = \sqrt{\frac{1}{N_P N_{out}} \sum_{i=1}^{N_P} \sum_{j=1}^{N_{out}} (tar_{i,j} - out_{i,j})^2} \qquad (7)$$

where N_P is the total number of I/O pairs in the training set and N_{out} is the number of output units. “Exact” results are those computed by a conventional structural analysis.

4.2 Selection of the training set

An important factor governing the success of the learning procedure of a NN architecture is the selection of the training set. A sufficient number of input data properly distributed in the design space together with the output data resulting from complete structural analyses are needed for the BP algorithm in order to provide satisfactory results. Overloading the network with unnecessary similar information results to overtraining without increasing the accuracy of the predictions. A few tens of structural analyses have been found sufficient for the examples considered to produce a satisfactory training of the NN. Ninety percent of those runs is used for training and the rest is used to test the results of the NN.

Most researchers split the design space into subregions and try to combine randomly the values within each subregion in order to obtain a training set which is representative of the whole design space. This procedure leads frequently to a huge number of training patterns in order to ensure that the whole design space is properly represented. In an effort to increase the robustness as well as the computational efficiency of the NN procedure various types of training set selection were investigated in a previous study [4]. In this study two types of training set selection are used: (i) the training set is chosen automatically based on a Gaussian distribution of the design variables around the midpoints of the design space, (ii) the training set is chosen using data from the structural analyses carried out in a number of ES optimization steps until the design reaches stationarity near the optimum. This happens when the value of objective function remains unchanged for a number of ES generations.

The first type of the training set selection was motivated from the fact that usually the searching for the optimum and its location lies in the region near the midpoints of the design space. A Gaussian distribution was therefore used for the random selection of input data in order to cover the whole design space and enforce the selection of most input patterns around the midpoints of the design space. This approach proved to be more efficient than choosing randomly combinations of input data from the whole range of the design space using a uniform distribution of the design variables [4]. The second type of the training set selection is based on the fact that in most cases the ES optimizer very fast tracks the path to the optimum and then it may oscillate around it until convergence in a slower rate. Therefore it is more efficient to produce the training sets in the vicinity of the design point where the optimizer has reached a stationary point. This way a smaller number of training sets is required and the NN training is performed much faster and accurately.

5. STRUCTURAL OPTIMIZATION BASED ON ES AND NN

After the selection of the suitable NN architecture the training procedure is performed using a number (M) of data sets, selected as described previously, in order to obtain the I/O pairs needed for the NN training. Since the NN based structural analysis can only provide approximate results it is recommended that a correction on the output values should be performed in order to alleviate any inaccuracies entailed, especially when the constraint value is near the limit which separates the feasible and the infeasible region. This is achieved with a relaxation of this limit during the NN testing phase before entering the optimization procedure. A "correction" of the allowable constraint values was therefore performed proportional to the maximum testing error of the NN configuration. The maximum testing error is the largest average error of the output values among testing patterns. Whenever the predicted values were found smaller than those derived from a conventional structural analysis the allowable values of the constraints were decreased according to the maximum testing error of the NN configuration and vice versa.

The proposed ES-NN methodology can be described with the following algorithms according to the two types of training set selection schemes that were previously described:

5.1 Algorithm 1

The combined ES-NN optimization procedure is performed in two phases. The first phase includes the training set selection, the structural analyses required to obtain the necessary I/O data for the NN training, and finally the selection, training and testing of a suitable NN configuration. The second phase is the ES optimization stage where the trained NN is used to predict the response of the structure in terms of constraints function values, due to different sets of design variables, instead of the standard structural analysis computations. The proposed methodology ES-NN can be described with the following algorithm:

- NN training phase :

1. *Training set selection step :* select s_i (i = 1,2,...,M) input patterns.
2. *Structural analysis step :* solve $K(s_i)u_i = f$ (i=1,...,M).
3. *Training step :* selection and training of a suitable NN architecture.
4. *Testing step :* test NN and “correct” allowable constraint values.

- ES-NN optimization phase :

1. *Selection step :*
   selection of s_i (i = 1,2,...,ì) parent vectors of the design variables.
2. *Prediction step :* using NN to compute optimization function values for the μ parent vectors.
3. *Constraints check :* all parent vectors become feasible.
4. *Offspring generation :*
   generate s_j, (j=1,2,...,ë) offspring vectors of the design variables.
5. *Prediction step :* using NN to compute optimization function values for the λ offspring vectors.
6. *Constraints check :*
   if satisfied continue, else change s_j and go to *step 4.*
7. *Selection step :*

selection of the next generation parents according to (ì+ë) or (ì,ë) selection schemes.

8. *Convergence check :* If satisfied stop, else go to *step 3.*

5.2 Algorithm 2

According to the second type of training set selection the proposed ES-NN methodology can be described with the following algorithm: The combined ES-NN optimization procedure is performed in three phases. The first phase is the ES optimization stage until a stationary point is obtained. This is the case when the mean value of the objective values from all parent vectors in the last n∗ì/ë generations has not been improved by less than a given value $å_d$ (=0.05). The second phase includes the training set selection in the vicinity of the stationary point from the previous structural analyses during previous ES steps. This way the necessary I/O data required for the NN training are obtained, and finally the selection, training and testing of a suitable NN configuration. The third phase is identical to the second phase of algorithm 1. The second algorithm is described as follows:

- ES optimization phase :

1. *Selection step :*
   selection of s_i (i = 1,2,...,ì) parent vectors of the design variables
2. *Analysis step :* solve $K(s_i)u_i = f$ (i=1,2,...,ì)
3. *Constraints check :* all parent vectors become feasible
4. *Offspring generation :*
   generate s_j, (j=1,2,...,ë) offspring vectors of the design variables
5. *Analysis step :* solve $K(s_j)u_j = f$ (j=1,2,...,ë)
6. *Constraints check :*
   if satisfied continue, else change s_j and go to *step 4*

7. *Selection step :*
   selection of the next generation parents according to (ì+ë) or (ì,ë) selection schemes
8. *Stationarity check :* If satisfied continue, else go to *step 3*

- NN training phase :

1. *Training set selection step :* choose s_i (i = 1,2,...,M) I/O data.
2. *Training step :* selection and training of a suitable NN architecture.
3. *Testing step :* test NN and "correct" allowable constraint values.

- ES-NN optimization phase : as in algorithm 1

6. NUMERICAL EXAMPLES

Test example 1

The performance of the optimization methodology discussed in previous sections is investigated for a characteristic test example in sizing structural optimization of three dimensional roof trusses. The NN method used in this study is the back propagation algorithm META-NETS [29]. In the tables containing the results of the test examples the following abbreviations are used: *ES* refers to the standard evolution strategies optimization procedure, in which structural analyses are performed in the conventional way. *ES-NN* refers to the combined NN and ES optimization procedure, where the structural analysis response is predicted by a trained NN. For the two different types of training set selection that have been compared in this study the following abbreviations are used: (i) *GT* stands for the random selection of training set based on a Gaussian distribution of the design variables in the design space according to Algorithm 1, (ii) *AT* stands for the "automatic" training set selection using the results obtained at the initial stages of the ES optimization procedure according to Algorithm 2. The symbol "*(c)* " is used when the allowable limits of the constraints have been adjusted, as discussed previously, in order to “correct” the NN predictions near the feasible region limits, while symbol “*(v)* ” indicates that the final design is violating the constraints and thus it is infeasible. All examples were run on a Silicon Graphics Power Challenge computer.

The test example of a typical double-layered space truss structure has been considered to illustrate the efficiency of the proposed methodology in sizing optimization problems with discrete design variables. This configuration has been tested for a number of different curvatures as shown in Figure 1, in order to investigate the influence of the curvature in the optimum design. Four different topologies were tested corresponding to $0^{ï}$, $5^{ï}$, $10^{ï}$, $15^{ï}$ inclination of the curved surface at the supports. The modulus of elasticity is 200 GPa (29,000 ksi) and the yield stress is F_y=250 MPa (36 ksi). Each member is assumed to have a thin-walled tubular cross section. The cross-sectional area is considered to be the design variable of each member. Members are divided to forty eight groups according to their position. For all test cases the finite element model consists of 8,000 members, 2,071 nodes and 6,183 degrees of freedom. The loads are taken as uniform vertical forces applied at joints equivalent to uniform load of 0.10 kN and a concentrated vertical load 50 kN at the center of the structure which corresponds to the maximum load of a crane and it is equally distributed to the central nodes of the roof. The objective function in all test cases is the weight of the structure. The constraints are imposed on the maximum nodal displacement and the maximum axial and buckling stresses in each element group. The values of allowable axial stress is $ó_a$=150 MPa, whereas the maximum allowable displacement is limited to 3 cm. The space truss with $5^{ï}$ inclination is susceptible to bifurcation buckling at lower loads as compared to $10^{ï}$, $15^{ï}$ inclination when there is imperfection. In this study only the buckling of the individual members is checked, not the overall buckling.

For all test cases the (ì+ë)-ES approach is used with ì=ë=20. The number of NN input units is equal to the number of design variables, whereas the output units are ninety eight corresponding to the two values of axial and buckling stresses for the forty eight element groups and the value of the maximum nodal displacement. The NN configuration has two hidden layers each one consisting of 35 nodes, which results in a 48-35-35-97 NN architecture used for all runs. The performance of the Gaussian and the “automatic” NN training set selection with 480 and 200 training sets, respectively, is shown in Tables 1-4 for the four configurations of the roof. The 200 training patterns that were used for the “automatic” NN training set selection were selected from the initial structural analyses of the ES procedure as described in section 4.2. The iteration history of the value of the objective function for the first test case is shown in Figure 2. It is obvious from the results that the performance of the proposed ES-NN methodology is superior to the performance of the conventional ES optimization procedure, since a dramatic improvement in total computing time required by ES-NN over ES is observed in all test cases examined. A significant improvement is also observed in the performance, both in terms of computing time and optimum values of the objective function, of the proposed ES-NN methodology when the “automatic” type of NN training is used over the Gaussian type of NN training. As it can be observed from the results obtained the curved type of structure is more economical from the flat roof type eventhough the surface of the structure is longer. For greater slopes, however, the overall weight grows since the surface of the structure increases significantly.

<u>Test example 2</u>

The performance of the optimization methodology is also investigated for the typical double-layered space truss roof of Figure 3. The optimum design of this space structure is based on a limit state analysis. Each member is assumed to have a thin-walled tubular cross section. The modulus of elasticity is 210 Gpa, while the permissible tension and compression forces at the ultimate limit state for each element type are given in Tables 5-7. The design variable of each member is considered to be its cross-sectional area. Members are divided: (i) into three groups, top-bottom-diagonal elements (test case 1); (ii) into eleven groups (4 subgroups for the top elements, 4 subgroups for the diagonal elements and 3 subgroups for the bottom elements) according to their stress values of a preliminary linear analysis (test case 2). For both test cases the finite element model consists of 584 members and 189 nodes with 527 degrees of freedom. The objective function in this test example is also the weight of the structure. For this test example the (ì+ë)-ES approach is used with ì=ë=10. The results for the two test cases are shown in Table 8.

7. CONCLUSIONS

The implementation of a hybrid optimization procedure, based on the combination of Evolution Strategies and Neural Networks for large-scale sizing optimization problems of 3-D truss structures was found to be very effective. The time-consuming requirements of repeated analyses associated with the optimization procedure using Evolution Strategies motivated the use of Neural Networks. The computational effort involved in the optimization procedure using Evolution Strategies becomes excessive in large-scale problems. The capability of Neural Networks to “predict”, within acceptable tolerance, the necessary data for Evolution Strategies can practically eliminate any limitation on the size of the problem. The methodology presented is an efficient, robust and generally applicable optimization procedure capable of finding the global optimum design of complicated structural optimization problems. It was also found that the proposed hybrid optimization methodology can reach the optimum for large and computationally intensive problems at a substantially reduced computing time compared to the standard Evolution Strategies optimization algorithm and conventional methods based on mathematical programming techniques [4].

The study performed on the NN training scheme showed that the technique of using the conventional ES optimization procedure until a stationary point of the optimization problem is located followed by NN training, using previously computed data, is superior to the rule of choosing randomly combinations of input data with a Gaussian distribution around the midpoints of the design space. This approach was motivated by the fact that the searching for the optimum by the ES optimizer proceeds very fast in the initial stages and eventually starts to slow down. Thus, when ES reach a stationary point then it can be replaced by the hybrid ES-NN methodology in order to speed-up the optimization procedure. The proposed NN training scheme makes the ES-NN methodology more robust and efficient since it restricts the space of interest and permits the use of smaller number of training sets from a narrower region. This results in a cost-effective and accurate NN training as it was shown in the test examples presented.

8. ACKNOWLEDGMENTS

This work has been supported by HC&M/9203390 project of the EU. The authors wish to thank G. Thierauf, J. Cai of University of Essen for their cooperation.

9. REFERENCES

1. Sienz, J., Integrated structural modelling, adaptive analysis and shape optimization, PhD Thesis, Dept. of Civil Eng., Univ. of Wales, Swansea, UK, 1994.
2. Papadrakakis M., Tsompanakis Y., Hinton E., and Sienz J,. Advanced Solution Methods in Topology Optimization and Shape Sensitivity

Analysis, Journal of Engineering Computations, Vol. 13, No. 5, 57-90, 1996.

3. Papadrakakis M., Tsompanakis Y., and Lagaros N.D., Structural shape optimization using Evolution Strategies, Report 96-5, Institute of Structural Analysis and Seismic Research, NTUA, Athens, Greece, 1996.
4. Papadrakakis M., Lagaros N.D., and Tsompanakis Y., Structural optimization using Evolution Strategies and Neural Networks, Comp. Meth. Appl. Mechanics & Engrg, (to appear), 1997.
5. Fogel L.J., Owens A.J., and Walsh M.J., Artificial intelligence through simulated evolution, Wiley, New York, 1966.
6. Goldberg D.E., Genetic algorithms in search, optimization and machine learning, Addison-Wesley Publishing, Reading, Massachusetts, 1989.
7. Holland J., Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor, 1975.
8. Rechenberg I., Evolution strategy: optimization of technical systems according to the principles of biological evolution, Frommann-Holzboog, Stuttgart, 1973.
9. Schwefel H.P., Numerical optimization for computer models, Wiley & Sons, Chichester, UK, 1981.
10. Hajela, P. and Berke, L., Neurobiological computational models in structural analysis and design, Computers & Structures, 41, 657-667, 1991.
11. Berke L., Patnaik S.N., and Murthy P.L.N., Optimum Design of Aerospace Structural Components using Neural Networks, Computers & Structures, 48, 1001-1010, 1993.
12. Berrais A., Neural networks in Structural Engineering: State of the art, in B.H.V. Topping (ed) Advances in Computational Structures Technology, CIVIL-COMP Press, Edinburgh, 93-101, 1996.
13. Waszczyszyn Z., Some recent and current problems of neurocomputing in civil and structural engineering, in B.H.V. Topping (ed.) Advances in Computational Structures Technology, CIVIL-COMP Press, Edinburgh, 43-58, 1996.
14. Rummelhart D.E., and McClelland J.L., Parallel Distributed Processing, Volume 1: Foundations, The MIT Press, Cambridge, 1986.
15. Prete G., Space truss structures: Typological characteristics and principal construction systems in Italy, International Journal of Space Structures, Vol. 9, No. 4, 191-200, 1994.

16.La Tegola A., Ombres L., and Pecora C., Minimum weight design of reticular space structures, International Journal of Space Structures, Vol. 9, No. 4, 179-190, 1994.
17.Estevez J., Valcarcel P., and Pablos J., Non-linear analysis to optimize the dimensioning of space structures of bars, in Hangai Y., Mashita K., and Mitsui K. (eds), Nonlinear analysis and design for shell and spatial structures, Proceedings of the SEIKEN-IASS symposium, 19-22 Oct. 1993, Tokyo, Japan, 273-280, 1993.
18.Mukai H., and Wada A., Seismic analysis and design of ductile struss structures, in Hangai Y., Mashita K., and Mitsui K. (eds.), Nonlinear analysis and design for shell and spatial structures, Proceedings of the SEIKEN-IASS symposium, 19-22 Oct. 1993, Tokyo, Japan, 409-416, 1993.
19.Ishikawa K., and Kato S., Earthquake resistant capacity and collapse mechanism of dynamic buckling on double layer latticed domes under vertical motions, in Hangai Y., Mashita K., and Mitsui K. (eds), Nonlinear analysis and design for shell and spatial structures, Proceedings of the SEIKEN-IASS symposium, 19-22 Oct. 1993, Tokyo, Japan, 569-576, 1993.
20.Hoffmeister F., and Back T., Genetic Algorithms and Evolution Strategies-Similarities and Differences, in Schwefel, H.P. and Manner, R. (eds.), Parallel Problems Solving from Nature, Springel-Verlag, Berlin, Germany, 455-469, 1991.
21.Thierauf G., and Cai J., A two level parallel evolution strategy for solving mixed-discrete structural optimization problems. The 21th ASME Design Automation Conference, Boston MA, September 17-221, 1995.
22.Thierauf G., and Cai J., Structural optimization based on Evolution Strategy, in Papadrakakis M. and Bugeda G. (eds), Advanced computational methods in structural mechanics, CIMNE, Barcelona, 266-280, 1996.
23.Michaliewicz, Z., ‘Genetic Algorithms + Data Structures = Evolution Programs’, Springer-Verlag, 1992.
24.Papadrakakis, M. and Lagaros N.D., Thierauf, G. and Cai, J.; Advanced Solution Methods in Structural Optimization Based on Evolution Strategies, Engineering Computations, (to appear), 1997.
25.Thanedar P.B., Arora J.S., Tseng C.H., Lim O.K., and Park G.J., Performance of some SQP methods on structural optimization problems, Inter. Journal of Num. Meth. Engng, 23, 2187-2203, 1986.

26.Lasdon L.S., Warren A.D., Jain A., and Ratner R., Design and testing of a generalized reduced gradient code for nonlinear programming, ACM Trans. Math. Softw., 4 (1), 34-50, 1978.
27.Svanberg K., The method of moving asymptotes, a new method for structural optimization, Int. J. Num. Meth. Eng., 23, 359-373, 1987.
28.Theocharis P.S., and Panagiotopoulos P.D., Neural networks for computing in fracture mechanics. Methods and prospects of applications, Comp. Meth. Appl. Mechanics & Engrg, 106, 1993.
29.Embrechts M.J., Metaneural-Version 4.1, Rensselear Polytechnique Institute, 1994.

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum Weight (tn)
			Analysis	Training	NN-ES	Total	
ES	4,150	-	-	-	-	93,125	460.8
NN-ES-GT	480	3,827	10,771	1,546	131	12,448	416.8(v)
NN-ES-GTc	480	4,094	10,771	1,546	140	12,457	466.3
NN-ES-A	472	3174	10,591	345	109	11,045	453.7(v)
NN-ES-Ac	472	3515	10,591	345	120	11,056	466.3

Table 1 - Test case 1 (0^o inclination): Performance of the optimization methods

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum Weight (tn)
			Analysis	Training	NN-ES	Total	
ES	3,940	-	-	-	-	88,220	349.3
NN-ES-GT	480	4,017	10,771	1,493	138	12,402	368.2
NN-ES-GTc	480	3,862	10,771	1,493	133	12,397	350.9
NN-ES-A	512	3116	11,489	361	106	11,956	328.9(v)
NN-ES-Ac	512	3406	11,489	361	117	11,967	351.7

Table 2 - Test case 2 (5^o inclination): Performance of the optimization methods

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum Weight (tn)
			Analysis	Training	NN-ES	Total	
ES	4,210	-	-	-	-	95,640	402.7
NN-ES-GT	480	4,132	10,771	1,575	140	12,486	435.7
NN-ES-GTc	480	4,256	10,771	1,575	147	12,493	405.8
NN-ES-A	423	3213	9,492	337	109	9,938	420.1
NN-ES-Ac	423	3147	9,492	337	107	9,936	405.1

Table 3 - Test case 3 (10^o inclination): Performance of the optimization methods

Analysis type	Number of FE analyses	Number of NN analyses	Computing time (s)				Optimum Weight (tn)
			Analysis	Training	NN-ES	Total	
ES	4,280	-	-	-	-	96,035	426.9
NN-ES-GT	480	4,021	10,771	1,631	138	12,540	404.3(v)
NN-ES-GTc	480	4,267	10,771	1,631	147	12,549	428.6
NN-ES-A	437	3613	9,806	356	124	10,286	433.3
NN-ES-Ac	437	3221	9,806	356	110	10,272	428.6

Table 4 - Test case 4 (15^o inclination): Performance of the optimization methods

Diameter (mm)	Thickness (mm)	Allowable compression (kN)	Allowable tension (kN)
42.4	3.25	44.12	123.97
48.3	3.25	62.26	142.65
60.3	3.65	116.44	201.46
76.1	4.50	224.49	313.92
88.9	4.85	315.14	397.16
114.3	3.65	350.23	393.49

Table 5 - Test example 2: Permission tension and compression in top chords diagonally laid out and bracing diagonal members

Diameter (mm)	Thickness (mm)	Allowable compression (kN)	Allowable tension (kN)
42.4	3.25	24.70	123.97
48.3	3.25	36.33	142.65
60.3	3.65	74.23	201.46
76.1	4.50	159.47	313.92
88.9	4.85	241.28	397.16
114.3	3.65	299.91	393.49

Table 6 - Test example 2: Permission tension and compression in top edge chords and bottom chords

Diameter (mm)	Thickness (mm)	Allowable compression (kN)	Allowable tension (kN)
42.4	3.25	12.95	123.97
48.3	3.25	19.45	142.65
60.3	3.65	41.81	201.46
76.1	4.50	97.06	313.92
88.9	4.85	157.11	397.16
114.3	3.65	221.66	393.49

Table 7 - Test example 2: Permission tension and compression in bottom corner chords

Test case	Optimum Weight (tn)	Number of Generations	Number of FE analysis
1	4.25	11	52
2	4.09	42	146

Table 8 - Test example 2: the two test cases