Neural and Evolutionary Computing in Finite Element Analysis

Amir M. Sharif

Department of Information Systems and Computing, Brunel University, Uxbridge, Middlesex, Great Britain

This paper presents a discussion of current neural and evolutionary techniques, applied to the field of Finite Element Analysis and Finite Element Mesh Generation. This numerical method is widely used in many science, and engineering analyses to compute many forms of static and dynamic fields and potentials, such as heat, stress or velocity, on a mesh of interconnecting elements. The accuracy of the underlying finite element mesh determines the magnitude of the error of the solution to the differential equations. Meshes have to be adapted to limit this as far as possible, usually in an a-posteriori sense. These techniques have been widely automated and used with great success, but no means yet exist which allow the efficient a-priori evaluation of a prospective finite element mesh before the equations are to be solved. As such, the application of expert and heuristic knowledge is largely required to produce visible benefits from adaptive remeshing processes. This paper discusses how neural and evolutionary architectures have addressed this problem and presents a complementary evolutionary model, which may aid in the generation of finite element meshes, as a result of on-going research into the development of such 'intelligent' techniques.

Keywords: Neural and evolutionary computing, Finite Element Analysis, Mesh

1. Finite Element Analysis and the Role of Soft Computing

1.1. Finite Element Analysis and Mesh Generation

Finite Element Analysis (FEA) is a powerful analysis and simulation tool used to solve many forms of engineering and mathematics-related problems, primarily to find quantities such as heat or stress, which are described by differential equations. The method relies upon solving the discretised form of these equations on the vertices of a mesh of finite elements, i.e. interconnecting structured / unstructured polygons within a geometrical boundary (George 1991; Thompson et al. 1985; Zienkiwicz and Taylor 1971). Mesh adaption is required such that a high concentration of elements is generated where a large number of unknowns need to be solved for. By calculating the solution error of the discretised equations, the density of the mesh distribution can be refined in an iterative, a-posteriori sense (Zienkiwicz and Zhu 1990).

FEA has become a very powerful and popular way of modelling many engineering and physical phenomena in 2 and 3 dimensions. A mesh of interconnecting polygons, known as finite elements, are used as a basis for solving discretised PDEs (Partial Differential Equations) which describe many physical phenomena such as heat, stress, velocity vectors or electromagnetic fields. Modelling such systems of equations still remains to be both a science and a skill, with respect to how the equations are discretised and solved and to the choice of elements for a particular problem.

The essential format for applying FEA to a problem is quite straightforward and is shown in Figure 1. After the definition of the problem geometry, boundary conditions are stated and then a matrix of discretised algebraic equations is assembled which are subsequently solved with respect to an error tolerance. Because of this simplicity, FEA has understandbly become a very flexible tool for providing analyses of real-world phenomena.

Finite elements are used for generating a matrix of unknowns which relate directly to the differential equations described by a variational calculus formulation of the general form of the

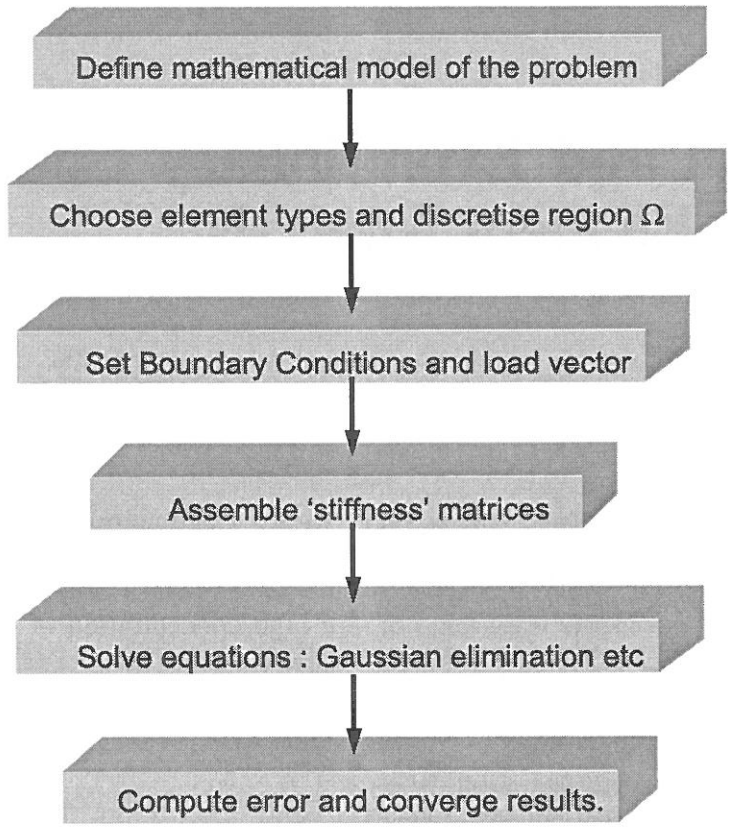

Fig. 1. The typical FEA procedure

equations,

$$\nabla^2 u = f \tag{1}$$

with the boundary conditions,

$$u = u_0 \text{ on the boundary } \Gamma \tag{2}$$

$$\frac{\partial u}{\partial n} = 0 \text{ within the region } \Omega \tag{3}$$

where u are the unknown quantities to be found, and n is the unit normal vector. As such equations (1-3) are widely known as the *Boundary Value Problem* (BVP), simply stated, it is necessary to find a function, u, which minimises a function, f.

Typically, finite elements are described using shape / basis functions, which are polynomial expressions of the form

$$u \approx \overline{u}(x) = \sum_{1}^{x} N_i \phi_i(x) \tag{4}$$

where ϕ_i are unknown variables and N_i are *shape or basis functions*. The solution of (4) involves finding the values of ϕ_i. A solution procedure must be formed whereby the approximation $\overline{u}$ via the shape functions N (or ϕ) will minimise u. Basis functions describe the subdivision of the domain into a series of finite regions and essentially are based on the topological relationship between element nodes.

Equation 1 is solved by a discretisation of the differential equations which subsequently decomposes the into the solution of a matrix problem,

$$Au = B \tag{5}$$

where A and B are vectors which relate to the function u. The vectors A and B are usually defined as the geometrical composition of the domain Ω, and the vector u respectively. In plane elasticity problems for example, this vector principally describes external forces and loads on the body being studied. The composition of u with respect to these vectors means that a series of algebraic equations needs to be solved in matrix form. This is usually carried out via schemes such as Gaussian Elimination and Gauss-Seidel iteration.

Of primary concern to the FEA process is the question of generating a mesh of interconnecting polygons for a particular problem. Meshes

can be generated fairly easily, through manual (CAD), automatic (algorithm-based) and even purely random means (through user-oriented arbitrary node insertion). The choice of element shape largely depends upon the nature of the problem being studied. Typically, quadrilateral elements (i.e. structured meshes) are used to mesh geometries or domains where irregular boundaries occur or where statically determinate solutions to PDEs are required (such as in plane stress and plane strain problems). These elements can be generated simply by interpolating between a uniform distribution of points in space (i.e. finite element nodes). Triangular elements are hence used for meshing well-defined geometries, such as rectangular or annular domains found in many fluid flow, diffusion and vibration problems Unstructured meshes such as these, are generated using either a Delaunay triangulation of the set of points in the plane (Risler 1992; Filipiak 1996) or by the Advancing Front technique, whereby triangles are added to the domain as required, so as to solve for many unknowns (George 1991).

Within classical algorithmic means of node and element generation the distribution characteristics are generally based on a polynomial law, the power of which defines the accuracy of any ensuing finite element (FE) calculation. Although elements can be generated and positioned within a geometrical domain quite easily, an optimal distribution of elements requires that the characteristics of a given problem be also taken into account (for example, increasing element distribution near a point of singularity, such as an acute corner where there may be a stress concentration). This is more commonly expressed through an error tolerance or 'energy norm', which describes convergence of the solution of the PDEs to the exact solution on the prescribed mesh (Zienkiwicz and Zhu 1990). This measure is then used as the basis for either increasing the density of the elements in the mesh (known as *h*-adaptivity) or for increasing the power of the polynomial power law (*p*-adaptivity). These levels of refinement can greatly improve the convergence properties of an FE simulation and are now widely employed throughout industry-standard FEA codes (Filipiak 1996).

1.2. AI-Driven Computation

Although the advent of FEA has greatly helped in the design of many engineered components from aircraft wings through to magnets, mesh adaptivity and refinement is still an area of great interest. Also, it has been argued that numerical methods can produce results which may appear to be accurate but can easily be misrepresented out of the context (Babuska 1996; Szabo and Actis 1995). Furthermore, the use of artificially intelligent means of control and manipulation of the FEA cycle has shown that automated assistance within this modelling technique is a developing science which has great potential (Sharif 1997; Sharif and Barrett 1998). There remains to be a plethora of knowledge-based systems which are based upon application-specific finite element case studies used to support FEA modelling decisions - for example as in the case of supplying mesh density distributions (Rank and Babuska 1987), or for collating diverse methods of solution to the same problem via a series of interconnected knowledge-bases (Turkiyyah and Fenves 1996). To aid in the description of elicited knowledge, fuzzy logic has also been used for describing verbose and imprecise information such as in the case of describing boundary and edge data for thermal analysis of a nuclear reactor wall (Yagawa 1995). For cases which require limited changes in the definition of a problem, as for pressure vessel design, formal relationships in terms of location, adjacency and size of mesh elements are more appropriate (Dolsak 1998). In such a way, the refinement of a mesh relies upon the refinement of the representation of each element (or subset of elements) within the mesh to achieve a low solution error. However, such approaches are not suitable for cases when both the problem geometry and mesh distribution requirements are likely to vary greatly and are therefore computationally brittle.

To overcome some of the limitations of these traditional AI approaches, biologically inspired means of computation, adaption, search and learning have been much sought after to augment human problem solving and analytical modelling tasks. Systems which mimic the processing capabilities of the human brain, Neural Networks (NN), and also the systems which are akin to processes of Darwinian principles of 'survival of the fittest', Genetic Algorithms

(GA), are two current techniques becoming widely used within engineering applications which rely on numerical computation (Michalewicz 1992). Techniques, such as these, have gained widespread popularity in engineering design and manufacture in recent years, where optimum criteria to be satisfied or searched for can be evaluated through evolving technique aims to develop successively better, 'fit', solutions to a problem via Darwinian principles of evolution i.e. 'survival of the fittest'.

This paper outlines applications of such tools, against the background of FEA which has been described in the preceeding paragraphs. A discussion of how feedforward, back-propagating and self-organising neural networks are utilised within FEA, is given in section 2. Section 3 subsequently discusses some similarities between neural and evolutionary architectures as a result of the author's on-going research in the application of genetic algorithms to this problem, wherein a combined neuro-genetic system is proposed.

2. Neural Networks within FEA

A Neural Network (NN) is a computational structure comprising a series of data processing units which serve to operate as a 'black box', analogous to the thinking and problem solving capabilities of the human brain. In its simplest form, a NN consists of a series of interconnected data processing units or 'neurons' which take input data (a_i) and produce output data (b), based upon the result of some neural weights (w_i). These weights define the influence that each neuron within the network has on its neighbour, based upon the 'strength' of information flowing into it. This allows data to flow in and out of the neural node via an activation function, described by an neuron threshold θ (see Figure 2). Neural networks are predominantly efficient at being trained to learn and find patterns within a given set of data and can be categorised into Supervised (teacher-pupil), or Unsupervised (experimentalist) learning paradigms (Aleksander and Morton 1990; Simpson 1990). The former method involves providing the network with information about characteristic features within a prescribed data set, wherein data is processed by the network at the pace of the expert supplying the information. This is typically defined as 'off-line' learning. Unsupervised, or 'on-line', learning involves the NN adjusting the input weights to new information as it is presented to the neurons, and as such is a dynamic process, more equivalent to how the human brain learns by trial and error, in real-time.

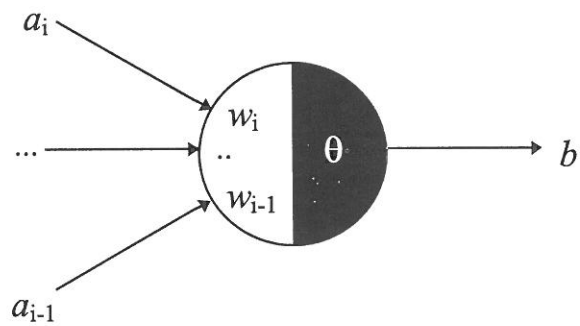

Fig. 2. A neuron, some inputs ($ai, \ldots, a_{i-1}$), its weights ($w_i, \ldots, w_{i-1}$), neuron threshold (θ), and an output (b)

Communication between the nodes can exist by passing information onto the proceeding nodes, which is therefore classed as a *feedforward* NN or by passing information between the output and input nodes during the learning phase, i.e. a *back-propagating* NN. Feedforward nets have been found to be useful for automating the entire FEA while back-propagation and self-organising maps are more suitable to error analysis and mesh generation tasks (i.e. low-level structural processes which are necessary for high level processes), as shown in Table 1. These details are discussed in more detail in the following sections.

2.1. Error Norms

Within adaptive mesh refinement (*a-posteriori* error analysis), a mesh is refined automatically by the mesh generator when the FE error exceeds an acceptable percentage. Mesh nodes are inserted or removed as requried within certain regions of the mesh and by iteratively carrying out this process the error of the FE solution is reduced and a convergent solution is obtained with this modified mesh. An energy norm that is generally used to describe the error is given as:

$$\|\nabla e_\Delta\| = \sqrt{\int_\Omega |\nabla e_\Delta|^2 d\Omega} \qquad (6)$$

Research Basis \ Neural Network Model	Feed-Forward	Backpropagation	Self-Organising Map (SOM)
FEA / FEMG APPLICATION	*Tape head design* Dyck *et al.* (1992)	*Mesh partitioning* Khan et al. (1993)	*Delaunay meshes* Ahn *et al.* (1991)
	Magnetic core design Lowther and Dyck (1993)	*Structural Mechanics* Jadid and Fairbairn (1994)	*Delaunay meshes* Alfonzetti *et al.* (1996)
	Conductor design Satsios *et al.* (1996)	*Mesh partitioning* Topping and Bahreininijad (1995)	*Quadrilateral meshes* Manevitz and Yousef (1997)
	Cavity Flow Yagawa and Okuda (1996)		
Aspect of the Modelling Task	*AUTOMATION OF ENTIRE FEA PROCESS*	*AUTOMATION OF ERROR ANALYSIS*	*MESH ADAPTION*
	High level task (process)	→	Low level task (structure)

Table 1. Taxonomy of Neural techniques in FEA/FEMG

where $\nabla e_\Delta = \sum_{n=1}^{\text{nodes}} N_n \nabla e_E$, Ω is a 2D domain, N are the elemental shape functions and E is an element. An associated refinement indicator relates how well the mesh is being refined:

$$\eta = \sqrt{\kappa \frac{\|\nabla e_\Delta\|^2}{\sum_{n=1}^{N} \|\nabla \phi\|^2} + (1-\kappa) \frac{\|e_\Delta\|}{\sum_{n=1}^{N} \|\phi\|^2}} \quad (7)$$

where ϕ is the computed solution, e is the error and κ is a weight value (Zienkiwicz and Zhu 1990). Only regions of the mesh which produce high error values are refined upon every successive iteration (i.e. a completely new mesh does not have to be produced, but only *locally* refined).

Similarly, within neural network models it is desirable to perform some error correction on the learning of any new input patterns, that should satisfy the training data. For example, a feed-forward net processes training data in the form of p ordered pairs (x_p, t_p), which are input to a node, x_i, and an output o_i is produced. The target output is t_i, i.e. the given pattern to be found, which means that for the learning to be convergent, $o_i \rightarrow t_i$. Hence we need to minimise the error between the input and output states, which can be written as:

$$E = \frac{1}{2} \sum_{i=1}^{p} \|o_i - t_i\|^2 \quad (8)$$

whence we require $E \approx 0$. Since the neural weights essentially control the convergence of the output data, we also require that the derivatives of the errors related to these weights also approach zero. The gradient of E can be subsequently written as:

$$\nabla E = \left(\frac{\partial e}{\partial w_1}, \frac{\partial e}{\partial w_2}, \ldots, \frac{\partial e}{\partial w_i} \right) \quad (9)$$

Each weight can then be updated by the gradient

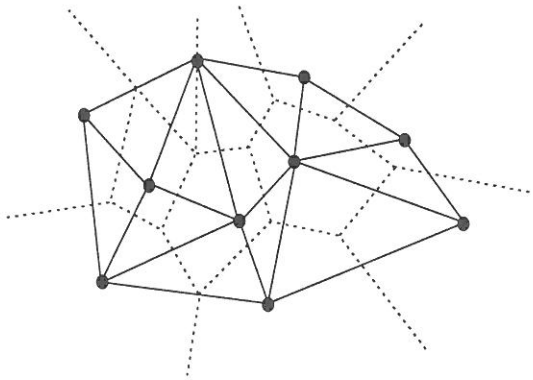

Fig. 3. An Unstructured mesh / Delaunay triangulation (-) of a set of points (.), based upon the Voronoi diagram (—) (George 1991; Risler 1992)

rule:

$$\Delta w_i = -\gamma \left(\frac{\partial E}{\partial w_i} \right) \qquad (10)$$

where γ is the learning rate (constant), for each weight w_i. Gradient learning seeks to minimise a functional describing the potential error in a training run, by finding the minimum value of its derivative. Hence minimisation of the error function in (8) relative to the input weights is required to increase the convergence of the output data towards the training set, t_p (Rojas 1996).

2.2. Feedforward Nets in FEA Processing

As stated in Section 1, the power of finite element analysis lies with applying variational calculus to the underlying equations which describe many engineering phenomena. Since all that is required is the discretisation and minimisation of the underlying differential equations, the formulation of an FE problem is relatively straightforward and can be represented through a series of algebraic equations.

As the underlying equations which the FEA process uses do not change, the only modification to a new problem to be solved is that of the model's geometry and material characteristics (i.e. width, height). In the case of tape head design (Dyck 1992; Lowther and Dyck 1993), a comparison is made between newly generated and previously learnt Delaunay-type meshes (as shown in Figure 3), for which an optimal solution for the magnetic field is reached through successive training runs. When a large amount of new data is to be learnt in this manner, scaling test data to new problems means that the potential for solution errors increases. In order to capture potentially 'lost' or unknown data as a result of scaling FE computations, fuzzy rule generation can be used in a neural net to learn new test problems (Satsios 1997). This is achieved by a fuzzy set of parameters which describe both network interconnections and the related parameters which are processed within them.

Research encountered so far has shown that the feedforward mechanism encompasses a 'black box' type processing ability, where patterns or solutions can be found between input and output states. Given a geometry and suitable boundary conditions for an FE problem, this can be imparted as training data into a network of neurons which contain activation functions based upon the variational equation (given as f in Figure 4).

In some cases it may be useful to adapt the feedforward processing capability to adhere to a state transition rule, i.e. the output from one neuron is dependent upon the output from a previous neuron. Since the convergence of FE computation is reliant upon previously computed values, the mesh used for a given problem can be made equivalent to a directed graph of neural nodes in the network (Yagawa and Okuda 1996). Communication between the nodes in the graph subsequently allows processing of the solution of the discretised equations to take place. In terms of solving the series of algebraic equations, the unknown values may be distributed across the neurons, the converged solutions for each of these being fed-forward in the network.

2.3. Back-propagation and Self-organisation for FEMG

For a mesh to be of any use to a finite element computation, it should be adaptable in order to limit any growth of errors in the solution of the differential equations as discussed in section 1. Back-propagating and Self-organising neural nets have the advantage over feedforward nets in this regard, such that learning errors are used to supervise and regulate the input weights. Back-propagation essentially deals with finding the minimum of a network error functional (as given by equation 8 in Section 2.2), such a network lends itself very well to variational problems (of which FEA is the best example).

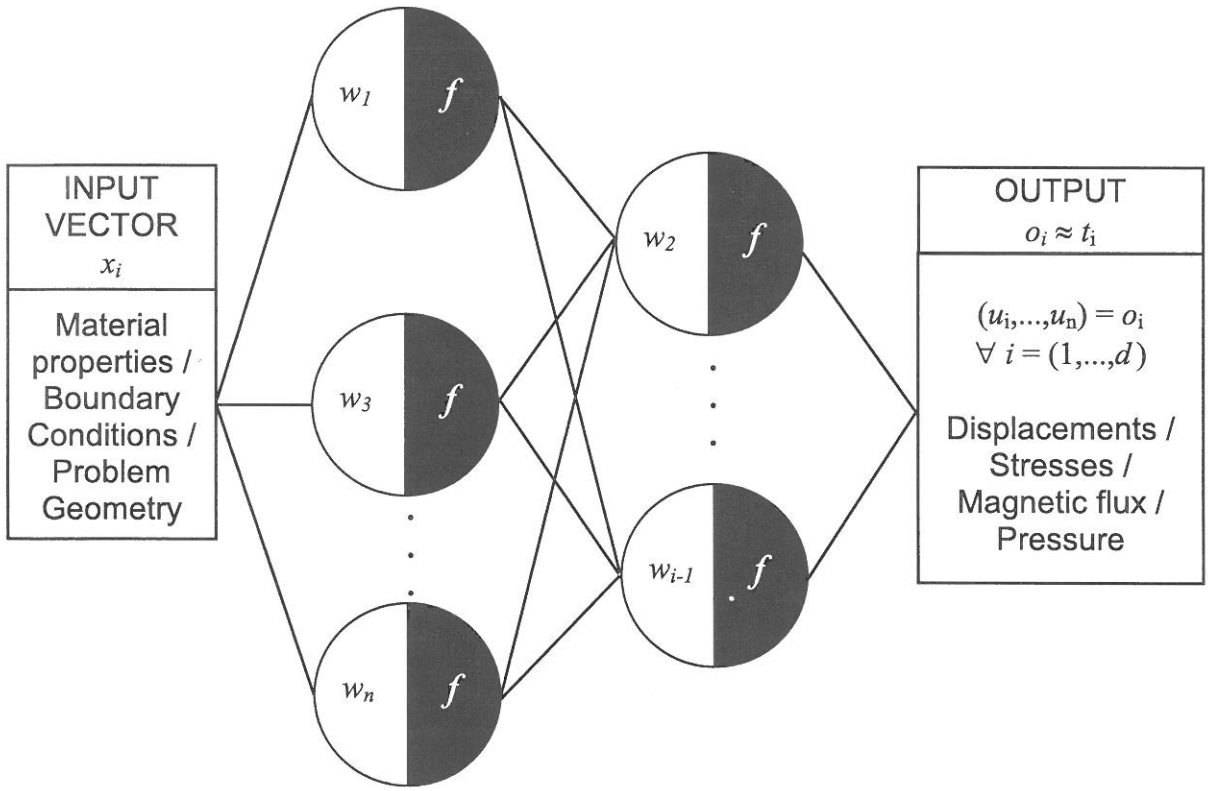

Fig. 4. A neuron within a Feedforward network applied to FEA (where f is an equivalent variational equation for the specific FEA case, u_i are the FE solutions to f and d are the respective degrees of freedom, w_i are weights based upon the discretisation of PDEs)

2.3.1. Back-propagation

Back-propagating algorithms are capable of storing both the initial and derivative data on each node and feedback the error in solution to preceeding nodes, as in Figure 5. The output vector (in o_i) is propagated back to the input weights and an evaluation is made between the original and current solutions. After this error analysis, E in Figure 5, the network is expected to interpolate between a new dataset, t_i, and the one that it has just learnt, o_i. The prime consideration in this type of network is to minimise the differences between learnt and unlearnt data, so as to recognise whether new input vectors presented to the network are similar to the previoulsy learnt ones. It therefore seems surprising that little application of back-propagation techniques to FEA and FEMG has been achieved in this light. This could possibly be due to the scaleability of some FEA problems, where the representation and learning of datasets may not be able to capture the required detail for new problems.

One method to overcome this is to decompose the problem into its constituent parts as in the sub-domain mesh generation problem. This requires that the amount of new elements to be included into an adapted mesh has to be found before the initial mesh is adapted, based upon a decomposition of the entire mesh over a series of processors (Khan 1993). Extensions to this work by Topping and Bahreininajad (1995) describe how an initial coarse mesh is fed into the NN, which predicts the number of new elements to be introduced into the mesh based on trained data (similar to Jadid and Fairbairn 1994). Similar to the state transition paradigm of Yagawa and Okuda, each neuron in the network fires only when it receives an output from a previous neuron. Optimisation of the partitioning of the mesh is then carried out using a simulated annealing procedure, whereby the error tolerance is allowed to converge to a locally acceptable value based upon the sum of the inputs and outputs (i.e. a *Hopfield energy functional*, Aleksander and Morton 1990). According to the authors, this is particularly useful since the application of the annealing algorithm helps to find a locally optimum adapted mesh el-

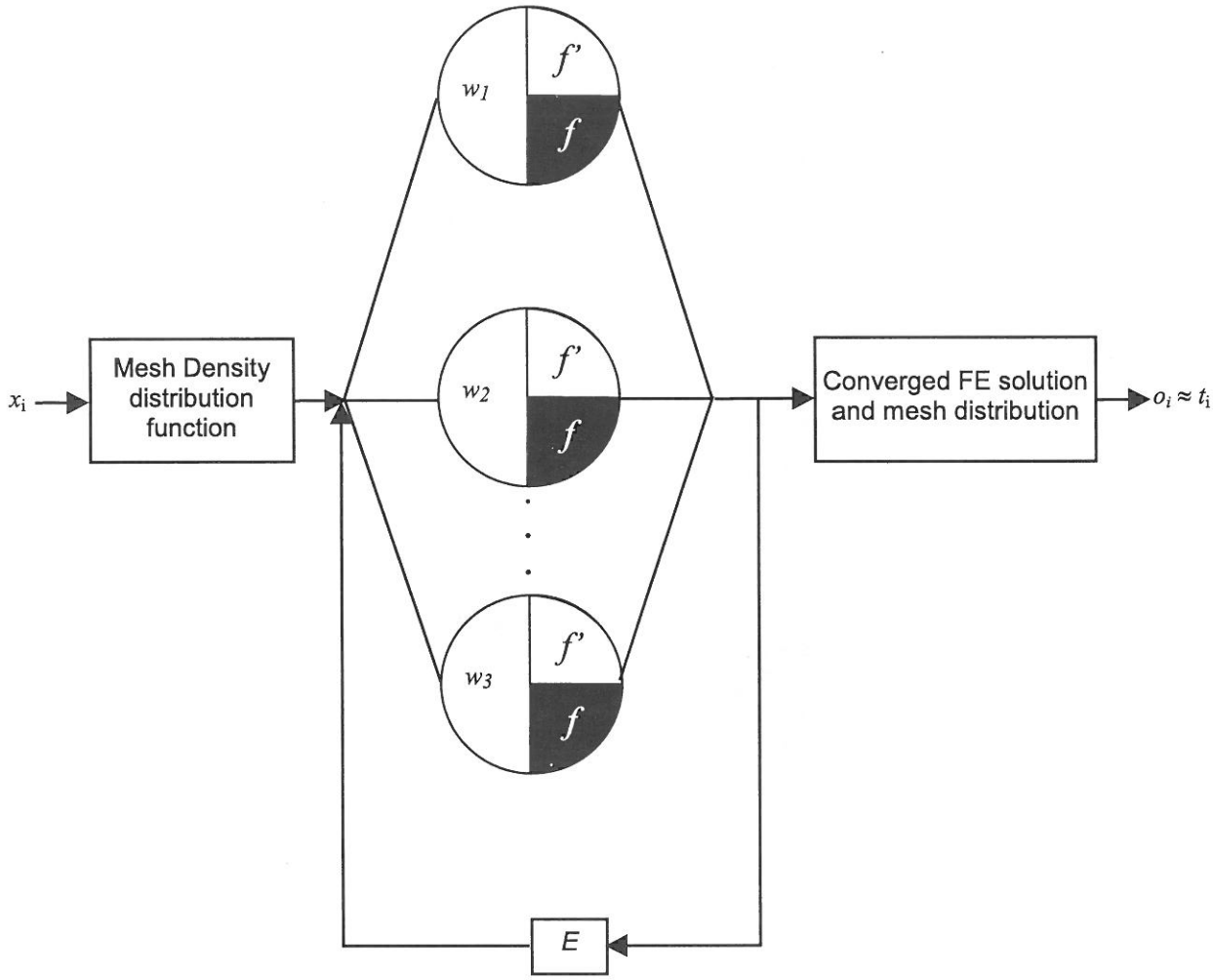

Fig. 5. A neuron within a Back-propagation network applied to FEA (where f is the equivalent FEA variational equation / elemental shape functions calculated on the backward pass, f' are the respective derivatives calculated on the forward pass and E is the network error function which is to be minimised)

ement. This means that parallel speedup of the partitioned mesh can be achieved. The learning rate and sensitivity of the annealing parameters mean that the optimum mesh decomposition is susceptible to increasing errors in computation.

2.3.2. Self-organisation

An interesting topological equivalence exists between unstructured meshes used in FEA and Kohonen's self-organising feature maps (SOM) (Simpson 1990). The purpose of an SOM is to provide a mapping of external stimuli to neural processing components. These mappings subsequently rely upon a spatial decomposition of a set of points from an input or sensory space to a computational or organised state, such as the Delaunay triangulation shown in Figure 3.

The behaviour of these mappings means that spatial topologies can be adapted to approach (meta)stable states of organised points in space, by 'sensing' and satisfying rules of growth and supervision. In the context of engineering analyses, the FE mesh and its solution error may be mapped in such a way by allowing it to respond to an external stimulant in the form of either a previously computed solution or randomly generated mesh coordinates (Ahn 1991), and Rao (1994) have used a similar approach in their work, based upon non-uniform mesh densities predefined by the user.

More recently, a 'Let-It-Grow' neural network has been described in which the nodal density distribution is increased gradually by the neural network (Alfonzetti 1996). Density function, based upon a homogenous probability distribution, is the given activation in this case, and the density distribution is allowed to increase to a pre-defined user value. Standard triangulation of the input space is performed on the learnt

a) *Appropriation* : Understand the problem which the user wishes to model and analyse, i.e. assist in the formulation of partial differential equations and their associated finite element formulation;

b) *Context* : Select appropriate element types for the resulting FE mesh that is required for FEA (high / low order triangles, quads, tetrahedra, bricks, etc);

c) *Evaluation* : Optimise the FE mesh in terms of geometry, topology or tesselative qualities (i.e. the overall design of the mesh);

d) *Characterisation* : Learn 'good' as well as 'bad' solutions, i.e. extract mesh characteristics via error analyses of proposed optimal meshes.

Table 2. Fundamental modelling criteria for Engineering Simulations

nodes and a mesh is produced. Since both mesh quality and speed of generation of the mesh are entirely reliant upon the relationship between mesh density and the amount of neural nodes carrying out the mapping, little user interaction is required to allow self-organisation of the elements to take place. It appears, however, that without inclusion of a functional which relates the outputs to the weights as are found in feed-forward and back-propagating networks, global minima cannot easily be evaluated.

In extreme cases, the SOM has been found to exhibit weak self-organisation, wherein the mapping may degenerate for 1D lines and 3D hypercube spaces (Fort and Pages 1996). This is partly due to the need for a target mapping to be given so that the stimuli space of the mapping can achieve equilibrium (equivalent to orthogonality and minimum angle criteria used in mesh adaption). This is particularly important when one considers providing a mapping to a defined boundary. One method to overcome this deficiency has been to "interweave" maps of different dimensions so that a 2D SOM approximates a user-specified density distribution while a 1D SOM is used to smooth the 2D SOM to the boundary edges (Manevitz and Yousef 1997).

3. Evolution and Adaption

It is apparent from the literature on FEA, that in order to produce consistent and definitive numerical simulation results, the context of the problem being analysed should be fully understood, at the same time realising that the underlying FE mesh should be as optimal as possible. These points are highlighted in Table 2. Automated adaption of the modelled problem should allow such errors to be eradicated, by means of learning and optimisation.

Biologically inspired adaptive systems, such as genetic algorithms, have shown the ability to learn and solve many scientific and engineering search and optimisation problems in an optimal sense (Goldberg 1989; Michalewicz 1992). Specifically, this technique encodes problem parameters into a population of bit string structures (known as chromosomes) and by using Darwinian principles of 'survival of the fittest', each individual is evaluated against a performance measure, called *fitness*. Those individuals which do not perform well are discarded whilst better performing individuals are evolved via genetic operators such as reproduction, crossover and mutation, from one generation to the next (see Figure 6). The algorithm proceeds until termination criteria set by the user are satisfied. A GA has the benefit of finding a global set of naturally fit or successful solutions as compared to other probabilistic techniques which have a tendency to converge to singluar points of global equilibrium (such as Hill climbing and Simulated Annealing approaches, Davis 1991).

Because of the inherently large number of populations that can be produced every successive

generation, each individual encoding a pattern or scheme that may represent potentially fit solutions, genetic algorithms can be quite computationally intensive. The scheme theory of genetic algorithms therefore provides the fundamental result, that this method of optimisation and search is inherently parallel and scaleable to multiprocessor computer architectures.

3.1. Evolving a Mesh to be Fit

If we concentrate on the optimisation of a mesh, we find that this task is essentially a *search* for a mesh which exhibits some 'nice' qualities defined by the FE analyst. This can be in the form of heuristics or previous solution attempts as may be coded in NN training data

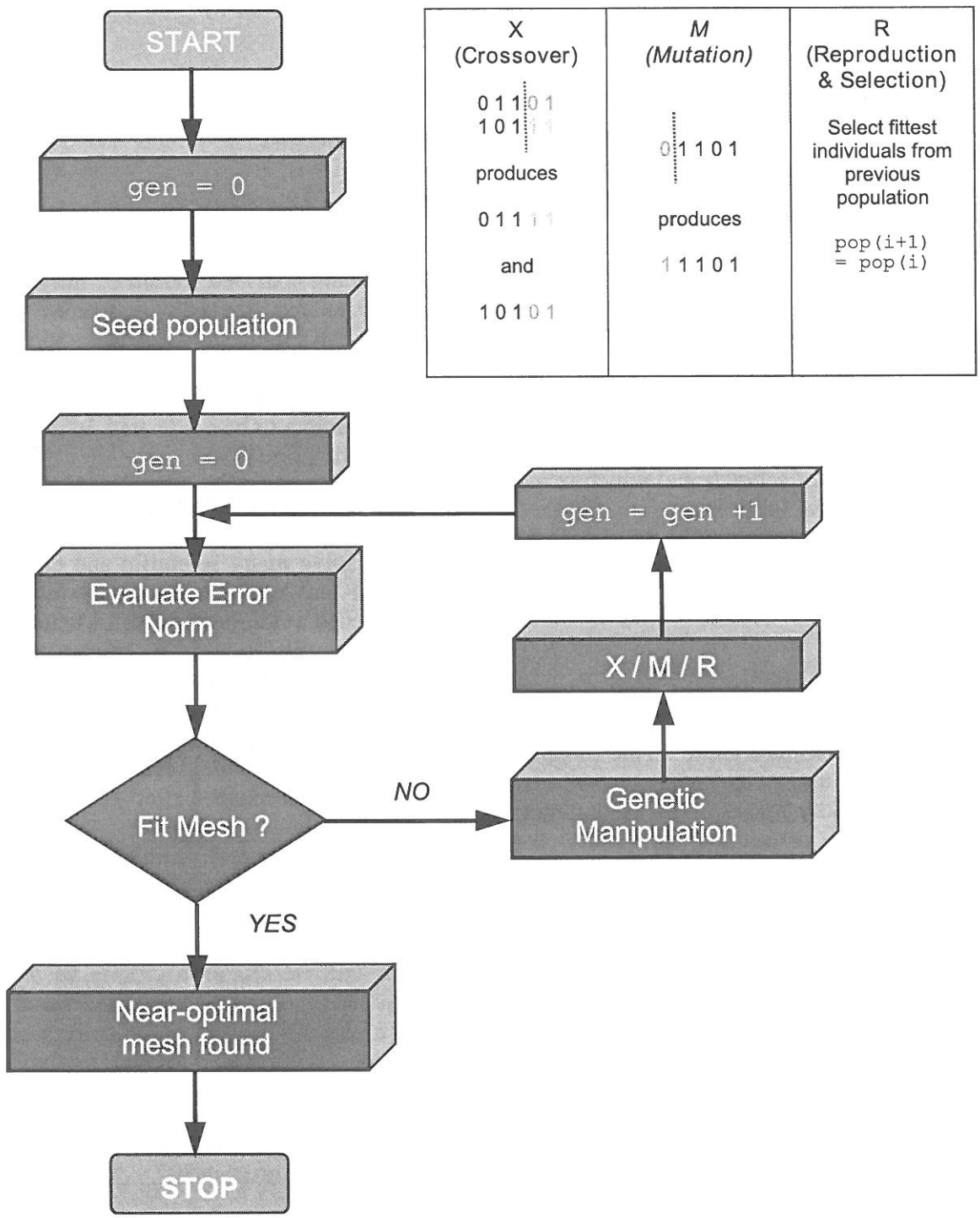

Fig. 6. A typical Genetic Algorithm

reviewed in section 2. Whilst it has been shown that neural processing structures are efficient at learning and recognising patterns in input data sets, the application of these techniques is at present specifically limited to the mapping of variational relationships as are found in the formulations within FEA.

The tedious and time-consuming "model-mesh-remesh" cycle can be overcome by the optimisation of the mesh generation task. The self-organising maps reviewed in section 2 appear to address this problem quite well. Since the basis of these mappings use classical Dirchlet tessellation techniques (Risler 1992), direct analogies can be made to similar tessellations used in some FEA calculations. However, the steady-state condition attained by these feature maps is heavily reliant upon some target mapping, which is required to be known beforehand. This is of course the basis of how a neural network operates. For generating optimal FE meshes, a target mesh may not be known in some cases and hence a suitable search and optimisation strategy might be better suited to the problem in hand.

It is proposed that the mesh generation process be optimised in such a way as to produce meshes which will ultimately limit the amount of time spent on excessive refinement strategies, by objectively evaluating the cost or 'fitness' of a candidate tessellation in an *a-priori* sense. The development of the concept of a *fit mesh* is now described.

A fit mesh is the one which needs to be adapted least for a range of different system parameters. In other words, changes in the underlying equations and boundary conditions of the finite element discretisation will mean that the mesh should be able to accomodate these changes with the least amount of adaption of the mesh. As described in earlier work (Sharif and Ettinger 1997), a fit mesh should be produced via a mesh generation scheme that allows the representation of the problem most effectively (MG) and provides the basis for producing convergent FE results (F_{conv}). At the same time consistent topological relationships throughout the mesh should be maintained (E_{conn}), whilst providing sufficient amounts of elements in those regions of high solution requirement (E_{dist}). This can summarily be represented as the functional relationship:

$$mesh\_fit = f[F_{conv}, E_{conn}, MG, E_{dist}]$$

Essentially, a fundamental fitness measure for the evolutionary design of a mesh should describe a large distribution of elements in regions of high solution requirement. More specifically, further measures for describing the quality of mesh elements can be given in terms of aspect ratio and geometrical construction. For example, each particular type of element should retain its topology even after refinement and adaptation. Taking extreme conditions into account, a quadrilateral element's width should not be equivalent to its height, i.e. all internal angles must sum to 360 degrees, and any single angle must equal 90 degrees. Similarly for triangular elements within a Delaunay triangulation, each element should satisfy the minimum angle condition, i.e. all internal angles must sum to 180 degrees, and any single angle should be equal to or less than 60 (Risler 1992). When evaluating each element for such dimensionality conditions, the mesh elements should relate to a structured and representative numbering scheme to aid with the refinement process. Canann et al. (1998) have recently described methods with which to resolve the topology of quadrilateral meshes in this regard, by analysing the connectivity characteristics of local elements on a node-by-node basis through suggesting the nodes which should either be removed or amalgamated.

However, it should be noted that the overriding concept of a mesh, which should be fit, may be misleading and difficult to define. A fit mesh for one class of problem may be totally incorrect for another class of problem. Such concepts are difficult to perceive in general terms. A generally fit mesh would be problem-unspecific and based on topological or geometrical characteristics, which for a range of different problems, would change. A range of different scenarios is highlighted in Table 3 which further categorises modelling phenomena into either internal or external boundary meshing.

For stress analysis problems such as Case (a) and (b), the mesh element density will largely be controlled by the physics of the constitutive parameters such as elastic *modulii*, tensile strength and buckling load characteristics. Hence, a check must be maintained on the adaptability of the mesh elements so that when solving the

<table>
<tr><th>CASE TYPE</th><th>Internal Geometry
(Meshing occurs inside domain)</th><th>External Geometry
(Meshing occurs outside domain)</th></tr>
<tr><td>Stress Mechanics</td><td>(a) Plane Strain
(constitutive parameters)

(b) Plane Bending
(in-plane elasticity)</td><td>-</td></tr>
<tr><td rowspan="2">Fluid-Structure Interaction</td><td colspan="2">(c) Turbulence Modelling
(velocity and shear profiles)</td></tr>
<tr><td>-</td><td>(d) Streamline Modelling
(Pressure and velocity profiles)</td></tr>
<tr><td rowspan="2">Electromagnetics</td><td>(e) Waveguiding Devices
(Polarisation and reflective power)</td><td>-</td></tr>
<tr><td colspan="2">(f) Magnetic Devices
(dispersion characteristics)</td></tr>
</table>

Table 3. Characteristic meshing problems

quantities such as plane strain or plane stress, mesh elements do not get distorted due to the effects of boundary conditions (i.e. so-called "hourglass" modes encountered with quadrilateral elements having few degrees of freedom). When concerned with modelling turbulent internal and / or external fluid flows (case (c)), a compromise must be made between producing a static or transient simulation of the flow. If this is the case then, *inter alia*, the mesh should be highly adaptable to changes in the rates of mixing between streamline vortices and body boundary layers. Transient analyses will almost certainly require elements to have a high degree of orthogonality so that the propagation of flow characteristics is not hampered by poorly sized quadrilateral or triangular elements.

Streamline modelling (case (d)) requires an opposite approach in that the choice of element will be dependent on the nature of the flow regime being looked at. For example, tetrahedral elements are better suited to three-dimensional supersonic flows as opposed to brick elements (Filipiak 1996). In such a case, the propagation of flow characteristics, such as a travelling shockwave, is most important to the solving the pressure and velocity profiles over a body moving at high speed. Finally, waveguiding and electromagnetic device simulation will require a reasonable level of element refinement in order to capture polarisation effects of the propagating light mechanism at different powers (case (e)) and the dispersive nature of electrical and magnetic waves (case (f)).

3.2. Neuro-genetic Architecture

Since the prime concern of many discretisation problems is in the effective manner with which to segment either a data set or given geometry, it appears that such a problem is principally of an optimisable type.

The minimisation problem described in finite element problems as shown in section 1, can be posed as a set of input vectors in the guise of mesh nodes, and output vectors in the guise of bounded solutions. Comparing equations (1) with (3) and (2) with (5) we can see that both error indicators rely upon their respective updates of weight values which relate directly to input data (in the case of the neural net) or local elemental node error (in the case of mesh refinement). Rewriting (3) in terms of required FEA components, we can say that the error to be minimised in training is equivalent to

$$E = \frac{1}{2} \sum_{n=1}^{nodes} \|\phi - \nabla e_{\Delta}\|^2 \qquad (11)$$

where *nodes* are the total number of nodes in the finite element mesh where the differential equations are to be solved.

In this respect, the allied usage of a back-propagating neural system to find and classify only those types of problem which are best suited to optimisation by a GA-based FEMG routine, may provide a solution to this problem (Table 4). The genetic algorithm would be able to guide the global minimisation of mesh relationships to satisfy error norms of the form of equation (1), after which the learning task would be handed over to either a feedforward or back-propagating neural network / SOM to locally refine FE solution attempts. Similar hybrid approaches have been used in the literature whereby local search algorithms such as Simulated Annealing are used to 'fine tune' optimal search results found by global evolutionary search methods (Hameyer and Belmans 1996).

4. Conclusions

The application of neural networks to finite element analysis has been reviewed and discussed in this paper. Principally, it has been found that at the present moment there is very little research interest in time, which addresses the utility of the adaptive learning processes within such artificially intelligent techniques. Generally, feedforward architectures have been found to be suitable for automating the complete FEA process, in terms of minimising an equivalent variational equation. Back-propagating networks have the ability to transmit and feedback data asynchronously, which, when applied to numerical algorithms, is a most useful asset as far as error minimisation is concerned. Self-organising feature maps appear to be of assistance in the generation of meshes where

TECHNIQUE	**Solution of PDEs** $Ax = B$	**Mesh Adaption & Refinement**
Neural Network	*Feedforward*	*Backpropagation / Self-Organising Maps (SOM)*
Genetic Algorithm	Minimisation of Eq. (8) (search for connection strength)	Minimisation of Eq. (6) (search for optimum location / size of elements)

Table 4. Neuro-genetic FEA

user-defined elemental and nodal density distributions are to be found, by using concepts of 'growing' tessellations.

Since such networks are suited to the learning of sets of training data which incorporate previously known solutions, the application to numerical analysis techniques allows reasonably efficient output to be computed for new input data. In cases where the input data does not adhere to learnt patterns, fuzzy rules or probabilistic methods of inference may need to be generated to allow realistic output data to be found (MacKay 1996). Efficient, low error FEA is reliant upon optimal meshes. Optimisation and convergence of tessellations by neural techniques may lead to solutions which satsify locally minimum conditions. For optimal meshes, the description of a fit mesh will allow the generation and evaluation of meshes on an *a-priori* basis, before fully fledged FEA computations are carried out.

The application of global search techniques, such as are found in Genetic Algorithms (GA) has the potential to not only encode known training data but also enable the generation of interesting and non-trivial globally optimum solutions. By application of a locally defined error correction algorithm in the shape of a neural architecture, recognition and convergence of potential FE solutions may provide a reduction of the recompute-and-analyse cycle, which is often found in FEA.

References

C.H. AHN, S.S. LEE, H.J. LEE, H.J., S.Y. LEE, (1991), A Self-organizing Neural Network Approach for Automatic Mesh Generation, *IEEE Transactions on Magnetics*, **5 (27)**, 4201–4203.

I. ALEKSANDER, H. MORTON, (1990), *An Introduction to Neural Computing*, Chapman and Hall, London.

S. ALFONZETTI, S. COCO, S. CAVALIERI, M. MALGERI, (1996), Automatic mesh generation by the Let-It-Grow Neural-Network, *IEEE Transactions on Magnetics*, **32(3)**, 1349–1352.

I. BABUSKA, (1996), New problems and trends in the Finite Element Method, *Proc. Conf. on the Mathematics of Finite Elements (MAFELAP '96)*, (Ed. J.R. Whiteman), Brunel University, UK.

S.A. CANANN, S.N. MUTHUKRISHNAN, R.K. PHILLIPS, (1998), Topological Improvement Procedures for Quadrilateral Finite Element Meshes, *Engineering with Computers*, **14 (2)**, 168–177.

L. DAVIS, (1991), *Handbook of Genetic Algorithms*, Van Nostrand Rheinhold, New York.

B. DOLSAK, I. BRATKO, A. JEZERNIK, (1998), Application of Machine Learning in Finite Element Computation, *In Machine Learning, Data Mining and Knowledge Discovery: Methods and Applications*, (R.R. Michalski, I. Bratko, M. Kubat Eds.), John Wiley.

D.N. DYCK, D.A. LOWTHER, S. MCFEE, (1992), Determining an approximate Finite Element Mesh density using Neural Network techniques, *IEEE Transactions on Magnetics*, **28 (2)**, 1767–1770.

M. FILIPIAK, *Mesh Generation* (1996) Available: http://www.epcc.ed.ac.uk/epcc-tec/documents.html.

J.–C. FORT, G. PAGES, (1996), About the Kohonen Algorithm: Strong or Weak self-organisation?, *Neural Networks*, **9 (5)**, 773–785.

P.L. GEORGE, (1991), *Automatic Mesh Generation*, John Wiley, New York.

D.E. GOLDBERG, (1989), *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison Wesley, Reading, MA.

K. HAMEYER, R. BELMANS, (1996), Design and Optimization of Electrotechnical Devices, *Journal of Engineering Design*, **7 (3)**, 235–251.

M.N. JADID, D.R. FAIRBAIRN, (1994), The application of Neural-Network techniques to Structural Analysis by implementing an Adaptive Finite Element Mesh Generation, *Artificial Intelligence in Engineering Design and Manufacture*, **8 (3)**, 177–191.

A.I. KHAN, B.H.V. TOPPING, (1993), Parallel training of Neural Networks for Finite Element Mesh Generation, In *Neural Networks and Combinatorial Optimization in Civil and Structural Engineering*, (B.H.V. Topping, A.I. Khan Eds.), 81–94. Civil-Comp Press, Edinburgh.

D.A. LOWTHER, D.N. DYCK, D.N., (1993), A density-driven mesh generator guided by a neural network, *IEEE Transactions on Magnetics*, **29(2)**, 1927–1930.

D.J.C. MACKAY, (1996), Bayesian Methods for Neural Networks: Theory and applications, Presented at the *Proceedings 6th Annual Cambridge Neural Networks Summer School*, Emmanuel College, Cambridge University, Cambridge, UK, 9th-12th September 1996.

L. MANEVITZ, M. YOUSEF, (1997), Finite-Element Mesh Generation using Self-Organizing Neural Networks, *Microcomputers in Civil Engineering*, **12**, 233–250.

Z. MICHALEWICZ, (1992) *Data Structures + Evolution Programs = Genetic Algorithms*, Springer–Verlag, New York.

E. RANK, I. BABUSKA, (1987), An Expert System for the optimal mesh design in the hp-versionof the finite element method, *International Journal of Numerical Methods in Engineering*, **24**, 2087–2106.

L. RAO, B. HE, W. YAN, (1994), Novel adaptive generator based on Kohonen's Neural Network model and vector quantization, In *Proceedings of the 2nd International Conference on Computation in Electromagnetics*, Nottingham, UK, Apr 12–14 1994, IEE Conference #384, 193–197.

J.–J. RISLER, (1992), *Mathematical Foundations of CAD*, Cambridge University Press, Cambridge.

R. ROJAS, (1996), *Neural Networks - A systematic introduction*, Springer–Verlag, Berlin.

K. SANADA, (1993), C.W. RICHARDS, D.K. LONGMORE, D.N. JOHNSTON, Finite element model of hydraulic pipelines using an optimized interlacing grid system, *J. Cont. Sys. Eng.*, **207 (4)**, 213–201.

K.J. SATSIOS, D.P. LABRIDIS, D.P., P.S. DOKOPOULOS, (1997), Fuzzy Logic for scalingFinite Element solutions of Electromagnetic fields, *y IEEE Transactions on Magnetics*, **33(3)**, 2299–2308.

A. SHARIF, The Management of Intelligence-Assisted Finite Element Analysis Technology, Presented at the *Proceedings of the Portland International Conference on Management of Engineering and Technology (PICMET '97)*, Portland, Oregon, July 27–31st (1997), Portland State University / IEEE, Portland, Oregon.

A. SHARIF, R.D. ETTINGER, (1997), Finite Element Mesh Generation using Genetic Algorithms, Presented at *Late Breaking Papers at the Genetic Programming 1997 Conference*, Stanford University, Stanford, California, USA, July 13–16th, Stanford University Bookstore, 9–224.

A.M. SHARIF, A.N. BARRETT, (1998), Utilising knowledge for optimum mesh design, *IEE Colloquiam on Knowledge Discovery and Data Mining*, IEE, London, 7–8 May 1998. Digest No. 98/310, London: IEE, 4/1–4/5.

P.K. SIMPSON, *Artificial Neural Systems: Foundations, Paradigms and Applications*, Pergamon Press, NY, New York, 1990.

B.A. SZABO, R.L. ACTIS, (1996), Finite Element Analysis in Professional practice, *Computer Methods in Applied Mechanics and Engineering*, **133 (3–4)**, 209–228.

H. TAKAHASHI, H. SHIMIZU, (1991), A General Purpose Automatic Mesh Generation using Shape Recognition Techniques, In *Computers in Engineering 1991*, Vol. 1, (*Proceedings of the ASME International Conference on Computers in Engineering*), 519–526 ASME, New York.

J.F. THOMPSON, Z.U.A. WARSI, C.W. MASTIN, (1985), *Numerical Grid Generation, Foundations and Applications*, North Holland / Elsevier: NY, New York.

B.H.V. TOPPING, A. BAHREININEJAD, (1995), Subdomain Generation using parallel Q-state Potts Neural Networks, In *Developments in Neural Networks and Evolutionary Computing for Civil and Structural Engineering*, (B.H.V. TOPPING, Ed.), 65–78. Civil-Comp Press, Edinburgh.

G.M. TURKIYYAH, S.J. FENVES, (1996), Knowledge-Based Assistance for Finite Element Modelling, *IEEE Expert*, **11 (3)**, 23–32.

G. YAGAWA, S. YOSHIMURA, H. KAWA, (1995), Automatic large-scale Mesh Generation based on Fuzzy knowledge processing and computational geometry (with a new function for Three-dimensional Adaptive remeshing), *Transactions of the Japanese Society of Mechanical Engineers (JSME) Part A*, **61** (583), 652–659.

G.I. YAGAWA, H. OKUDA, (1996), Finite Element Solutions with Feedback Network Mechanism through direct minimization of Energy functionals, *International Journal of Numerical Methods in Engineering*, **39**, 867–883.

O.C. ZIENKIWICZ, R.L.TAYLOR, *The Finite Element Method in Engineering Science*, McGraw–Hill, London, 1971.

O.C. ZIENKIWICZ, J.Z. ZHU, (1990), The three R's of engineering analysis and error estimation and adaptivity, *Computer Methods in Applied Mechanics and Engineering*, **82**, 95–11.

Received: January, 1998
Revised: February, 1999
Accepted: February, 1999

Contact address:
Amir M. Sharif
Department of Information Systems and Computing
Brunel University
Uxbridge
Middlesex, UB8 3PH
Great Britain
phone: (+44) 1895 274000
fax: (+44) 1895 251686
e-mail: Amir.Sharif@brunel.ac.uk

AMIR M. SHARIF obtained his BEng (Hons) in Aeronautical Engineering from City University, London in 1994 and is currently involved in PhD research in Computer Science in the Department of Information Systems and Computing at Brunel University, United Kingdom. Amir is also a Fellow within the department and his current research interests include Engineering applications of AI (Genetic Algorithms, Fuzzy Expert Systems), Internet Technologies, Evaluation of Strategic IT/IS investments, Information Management and Problem Solving Environments.