UNIVERSITY OF Southampton

EVOLUTIONARY AND GENETIC STRATEGIES FOR TOPOLOGY OPTIMIZATION OF FRAMEWORKS: PARETO-COMPARISONS AND HYBRID METHODS

Charalambos Tsatsaris

Thesis for the degree of Doctor of Philosophy

Faculty of Engineering, Science and Mathematics
School of Engineering Sciences

March 2009

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ENGINEERING SCIENCES

DOCTOR OF PHILOSOPHY

EVOLUTIONARY AND GENETIC STRATEGIES FOR TOPOLOGY OPTIMIZATION OF FRAMEWORKS: PARETO-COMPARISONS AND HYBRID METHODS

By Charalambos Tsatsaris

The main objective of this research is to develop new efficient and cost-effective topology optimization methods for framework structures. The first part of the thesis concentrates on the assessment of the effectiveness of the Evolutionary Structural Optimization (ESO) method of designing frameworks. This method is critically examined by studying the trajectory in which designs evolve during the ESO process on the weight-maximum stress plane and later overlaying it with the Pareto Front (PF) for the two-objective problem of simultaneously minimising weight and the maximum stress within the structure. To study the Pareto-efficiency of the ESO method, the designs obtained by ESO are compared with the designs obtained using exhaustive search for combined performance on two counts: the maximum stress within the structure and the overall weight. Whilst for complex problems an exhaustive search is not practical, the approach adopted here is to encode the problem formulation using a genetic algorithm (GA) and to allow the formulation to evolve in the direction of improving Pareto optimal designs. Since GA is a stochastic method, the robustness of the conclusions has been assessed by running GA with multiple seeds. Numerical experiments show that ESO produces reasonable designs at little expense; however, the procedure misses out on several efficient designs if one could afford the computational expense. As far as topology and size optimization is concerned, it is observed that ESO produces Pareto sub-optimal designs, but is superior to GA if one could not afford a computationally demanding search. ESO is computationally efficient but it fails to produce some designs with very good structural performance.

In the second part of the thesis two new strategies for topology optimization of frameworks are developed by combining the Evolutionary Structural Optimization (ESO) method and Genetic Algorithms (GA). This approach combines the quality of a stochastic global search such as GA and the computational efficiency of ESO. The first method proposed here is the ESO assisted GA method (ESOaGA) in which ESO obtained designs are inserted in the GA population, helping the GA search to operate in more promising directions. The second method is the GA assisted ESO method (GAaESO), in which GA produced designs are used as starting points for a family of ESO runs. The designs obtained by the proposed methods and the "unassisted" GA are compared visually and quantitatively using three quality indicators: the hypervolume, epsilon and R indicators. The statistical significance of the quality indicators is also assessed. Again, two goals are used in this comparison: the maximum stress within the structure and the overall weight. Both hybrid methods can obtain better optimized designs in less computational time than the respective "unassisted" methods. Finally, an iterative application of GA and ESO is explored.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Charalambos Tsatsaris, declare that the thesis entitled

"EVOLUTIONARY AND GENETIC STRATEGIES FOR TOPOLOGY OPTIMIZATION OF FRAMEWORKS: PARETO-COMPARISONS AND HYBRID METHODS"

and the work presented in it are my own. I confirm that:

- this work was done wholly while in candidature for a research degree at the University;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- Parts of this work will be published as:

  - C. Tsatsaris, A. Bhaskar, A. J. Keane, "Comparison between Evolutionary Structural Optimization and genetic algorithm methods for Framework Design", In preparation for submission.

  - C. Tsatsaris, A. Bhaskar, A. J. Keane, "Evolutionary Structural Optimization assisted GA methods for framework topology", In preparation for submission.

Acknowledgements

First of all, I would like to express my gratitude and appreciation to my supervisor Dr. Atul Bhaskar, for his inspiring guidance, support and encouragement through my academic years.

I wish to thank Professor Andy Keane for his very useful suggestions and comments during my research.

I thank Dr. Ivan Voutchkov for his assistance with OptionsMatlab.

I would also like to thank my friends Maria Nestoridi, Emmanuel Fleris, Andreas Prongidis, Praveen Thokala and Dr. Thanasis Makrodimopoulos for their support and giving me a place to stay when I needed throughout this year.

I am forever indebted to my parents for their encouragement and support through all these years.

Nomenclature

σ^{vm}	Von Mises stress
σ_e^{vm}	Von Mises stress of the element
σ_x	Normal stresses in x direction
σ_y	Normal stresses in y direction
$\sigma_{\max}^{vm}$	Maximum von Mises of the whole structure
τ_{xy}	Shear stress
RR_i	Current rejection ratio
RR_o	Initial rejection ratio
ER	Evolutionary rate
MRR	Member removal ratio
N	Number of nodes
q_i	Local displacement of a plane frame element
$\overline{q}_i$	Global displacement of a plane frame element
T	Transformation matrix
E	Modulus of elasticity
I	Moment of inertia
A	Cross-sectional area
L	Length of plane frame element
θ	Element inclination angle
c, s, α	Constants
$[K]$	Stiffness matrix
$[K_i]$	Stiffness matrix for the i^{th} element in local coordinates
$[\overline{K}_i]$	Stiffness matrix for the i^{th} element in global coordinates
$\{\overline{f}\}$	Force vector for each element in global coordinates

$\{\bar{q}\}$	Nodal vector for each element in global coordinates
$\{f\}$	Force vector for each element in local coordinates
$\{q\}$	Nodal vector for each element in local coordinates
F	Force matrix
Q_{global}	Matrix that includes all the global nodal displacements
n, M	Total number of members in a fully connected design
r	Number of members removed from a fully connected design
n_{C_r}	No of designs with r members removed
Ω	Set of all approximation sets
B, C	Approximation sets
I_{ind}	Quality indicator
R	Reference set
I_H	Hypervolume indicator
I_H^-	Hypervolume indicator according to reference set R
I_ε	Epsilon indicator
$I_{\varepsilon+}$	Additive Epsilon indicator
ε	Minimum factor
Λ	Set of parameters
u	Utility function
u_{lambda}	Utility function according to set Λ of parameters
λ	Weight vector
u_λ	Weighted sum of the objective vectors
u^*	Maximum value of u_λ
Z	Set of objective vectors
z^1, z^2	Objective vectors
I_{R2}, I_{R3}	R indicators
H_0	Null hypothesis
H_a	Alternative hypothesis
P	Probability of test
α	Statistical significance level

Chapter 1

Introduction

1.1. The importance of structural optimization

While developing new products, it is important to search for optimal designs given the objectives and the constraints. Consideration to topology must be given at an early stage of the design process. Over the last decade a lot of research has been carried out on the development of efficient procedures for the solution of such problems. This thesis builds on such work.

Structural mechanics is significant at all stages of design while developing a mechanical component. The loading and support conditions of a particular design problem are usually known in advance, but the designer cannot be sure of what the actual structure is going to look like. Low weight is one of the main objectives for load carrying structures. This is particularly true for airworthy structures. Therefore weight reduction is frequently a main objective of the design task.

Research into the fields of material laws, advanced materials, contact mechanics, damage mechanics, etc., have proven to be of particular importance for solving various problems. This includes fast development in computer science and technology, the programming and the availability of sophisticated systems for analysing large scale, highly non-linear systems.

The development and construction of products, especially in industrial practice frequently raises the question as to what measures must be taken to improve quality without unnecessarily increasing the cost. Thus, a new area within the scope of Computer Aided Engineering commonly known as Structural Optimization has evolved.

Topology optimization has become an important and well recognized sub-area of structural optimization. Topology optimization aims at finding the topology of a structure that optimizes certain objectives. The aim is often expressed as maximizing the stiffness or strength of a body using a fixed amount of material. Topology optimization schemes are categorized into two classes - those for discrete structures and those for continuous structures. Extensive research and development has been focused on structural optimization in the past three decades. Most of this work has been related to the optimization of member cross-sections, while less effort has been devoted to topology optimization.

1.2. Categories of structural optimization

Minimizing the weight of a structure while satisfying various requirements on structural response, cost and manufacturing is a complicated task. Experienced engineers may be able to come up with solutions that fulfil some of these requirements, but they will seldom be able to obtain the optimal structure. In order to both optimize the structure and meet the given requirements, the engineers must make use of the speed and efficiency afforded by computer programs. The development of efficient computer algorithms for the optimization of structures is a very active area of research.

Nowadays structural optimization involves all problems in which the geometry is the subject of the optimization. Bearing in mind that a mechanical design has basic elements: topology, shape and size, we distinguish the following three types of structural optimization:

- *Sizing optimization*: a typical size of a structure is optimized (a thickness distribution of a beam or a plate, orientation of fibres in composite material). A simple sizing optimization problem is shown in Figure 1.1a.
- *Shape optimization*, when the shape of a structure is optimized, without changing its topology (Figure 1.1b).
- *Topology optimization*: besides the shape also the topology of a structure is optimized by creating holes, e.g. Figure 1.1c.

The boundary between sizing and shape optimization is not very sharp. Alterations in size optimization inevitably alter the shape of an object. In other words sizing optimization and shape optimization go hand in hand, and they are closely related. Both sizing optimization and shape optimization methods consider the optimization of structures with fixed topologies. However, topology optimization methods are based on the material connectivity of the structure. These methods find the optimal number of holes or members in a structure as far as a continuum or discrete structure is concerned (Figure 1.1c).

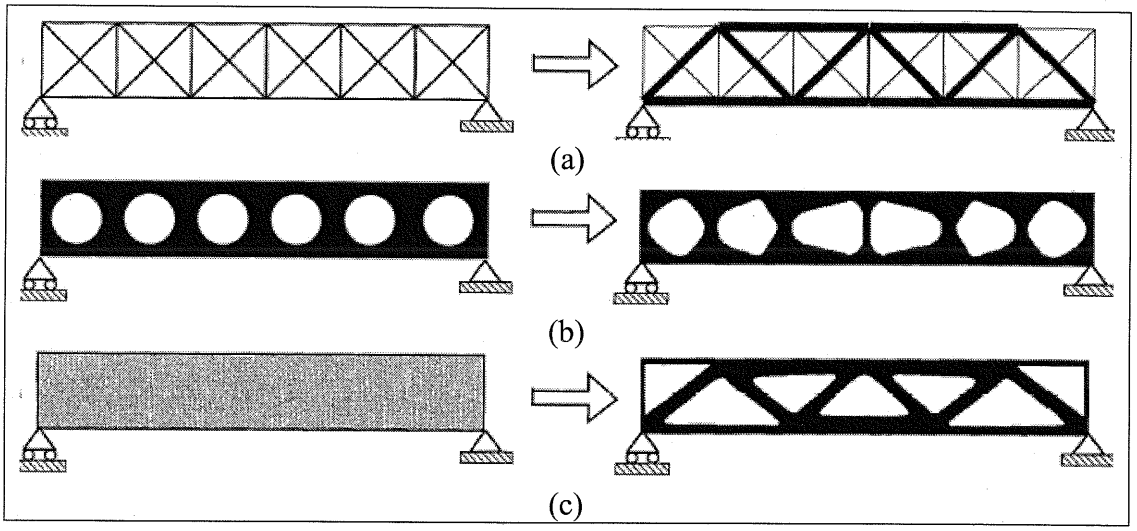


Figure 1.1: Three categories of structural optimization: (a) sizing optimization, (b) shape optimization, (c) topology optimization. The initial problems are shown at the left-hand side and the optimal solutions are shown at the right [1].

One of the important features of shape optimization is its *interdisciplinary character*. On the one hand, the problem has to be well posed from the mechanical point of view, requiring a good understanding of the physics. On the other hand, the problem has to be mathematically formulated and after that numerically solved [2]. At this stage no less than three mathematical disciplines interact: theory of partial differential equations (PDE's), approximation of PDE's usually by finite element methods and the theory of nonlinear mathematical programming.

The stiffness of a structure is one of the major requirements a designer has to take into account to design structures such as buildings and bridges. Despite the significant effort in the area of structural optimization over the past three decades,

most techniques developed so far are restricted to sizing optimization or shape optimization with fixed topology. Work on the effect of changes in topology and shape is limited. A basic reason for this is due to the inability to describe shape and topological changes via increments. The calculus of continuous variables is not easily applicable now in order to obtain sensitivity information, which is useful for design search and optimization.

1.3. Definition and terms of topology

The term topology is next discussed and defined mathematically. Topology as a sub-field of geometry should be explained. Etymologically, the word is derived from the Greek noun topos, which means location, place, space or domain. Mathematically speaking, topology is concerned with objects that are deformable in a so-called "rubber-like" manner. Topology optimization involves changing the connectivity of material in structural geometries. The earliest results were, perhaps Michell-type structures [3] regarding the optimal layout design of trusses. This work was later developed into the optimal layout theory [4].

In the 1950s and 1960s important papers on topology were contributed. The theories developed in that period were concerned with the topological domain. All subsets including straight lines and sets of points are called topological domains. From a mathematical point of view, all distortions are transformations or reversibly unique mappings. Topological transformations or topological mappings can be defined as those transformations of one topological domain into another that neither destroy existing nor generate new neighbourhood relations. Two topological domains are termed topologically equivalent if there exists a topological mapping of one of the domains into the other one (Figure 1.2). Hence, a topological property of a domain is a characteristic maintained at all topological mappings, i.e., it is invariant.

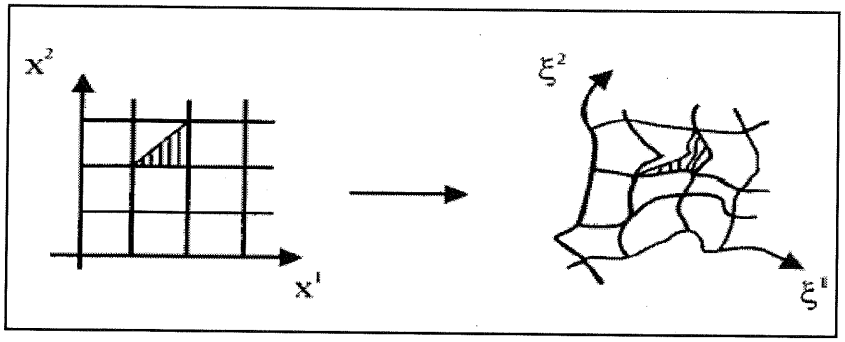


Figure 1.2: Topological mapping/transformation [5].

"Generally, topological transformations can be formulated as continuous transformations whose reverse transformation is also continuous. The latter case is also called homomorphism, i.e., the transformations are reversibly unique and continuous" [5].

Based on the above definitions of topology, a connection should be established between topology and optimization. The term "topology class" describes certain objects that are topologically equivalent (Figure 1.3a). A topology class is defined by the degree of connection of domains. A second topology class is defined by the degree to which the domains are connected (Figure 1.3b). A topology class is termed n-fold connected, if n-1 cuts from one boundary to another are required to transform a given, multiply connected domain into a simply connected domain (Figure 1.3c).

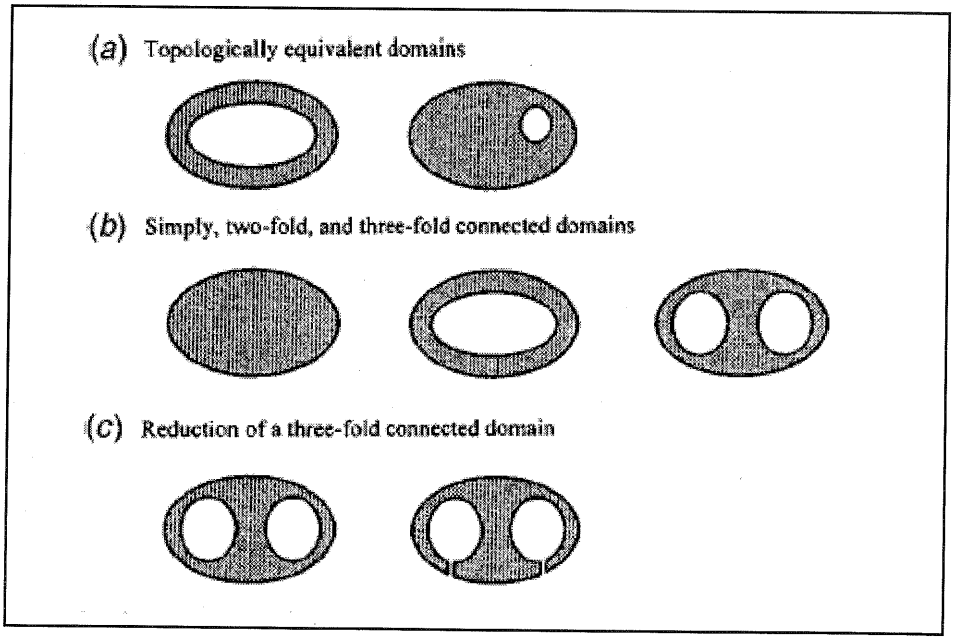


Figure 1.3: Topological properties of two-dimensional domains [5].

1.4. Applications of Topology Optimization

Reducing cost and improving performance are two key objectives in structural design. In the aerospace and automotive industries, this is particularly useful with respect to design criteria such as strength, stiffness, mass, fatigue life, manufacturing cost and maintenance cost [6]. Topology optimization is one method of reducing costs and improving performance in structural systems.

Research in topology optimization is currently of great interest. Potentially a large spectrum of industrial problems (see Figure 1.4) could benefit from such studies. The main contributions of topology optimization could be weight reduction and performance optimization of automobiles, aircraft, space vehicles and many other structures (Figure 1.4). It can be used to obtain the best layout of vehicle structural components to achieve prescribed performance goals. Recent optimization projects have shown that topology optimization helps to develop unique concepts and drives innovation. Good examples are several projects published by Airbus Industries and EADS where the use of topology optimization led to radically new design concepts that are superior to those developed using a classical design approach [7, 8]. Other applications include the design of transducers for underwater sound detection, car parts for crash-worthiness, medical implants and Micro-Electromechanical Systems for use in hearing aids and micro robots [9].

One of the goals of the vehicles industry is to decrease vehicle weight in order to reduce fuel consumption and increase transport efficiency. One way to achieve a lighter vehicle is to reduce the structural weight by use of structural optimization tools. Structural optimization includes several types of methods, e.g. size, shape, and topology optimization, whereby size and shape optimization are mostly used for improvements of existing structures while topology optimization is a more general method used to find an optimal structural layout [10-12]. A vehicle body can be seen as a frame structure with beams and joints, i.e. a space frame (see Figure 1.5). In the conceptual design phase, topology optimization of the frame structure is of interest to determine how beams should be arranged and how dimensions of beams and joints should be chosen.

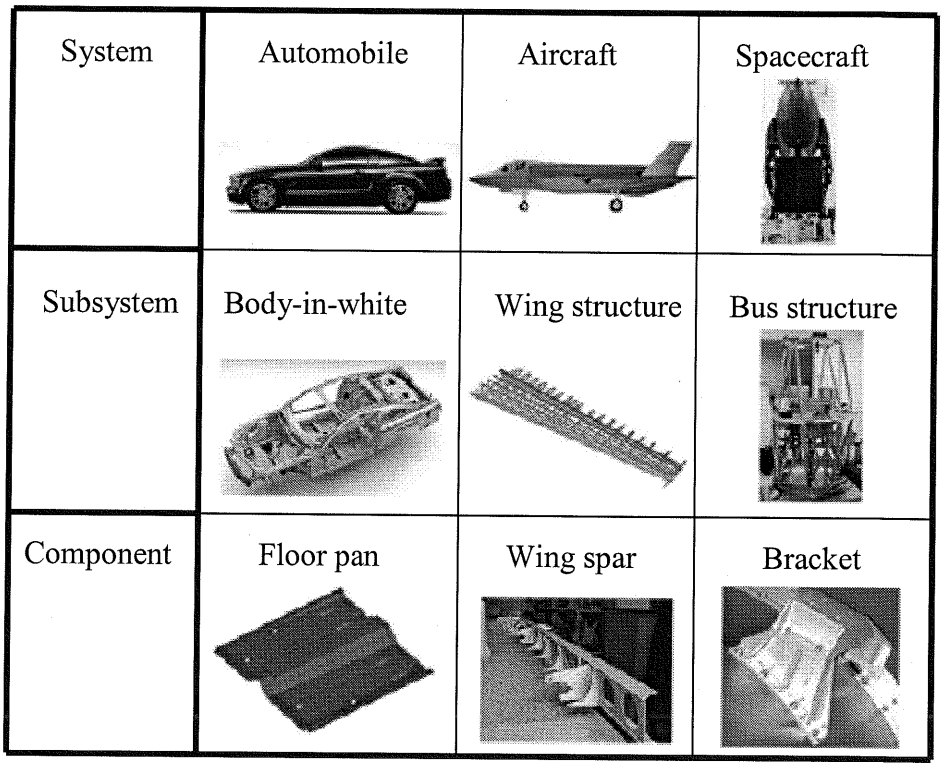


Figure 1.4: Examples of structural systems, subsystems, and components for the automotive, aircraft and spacecraft industries [13].

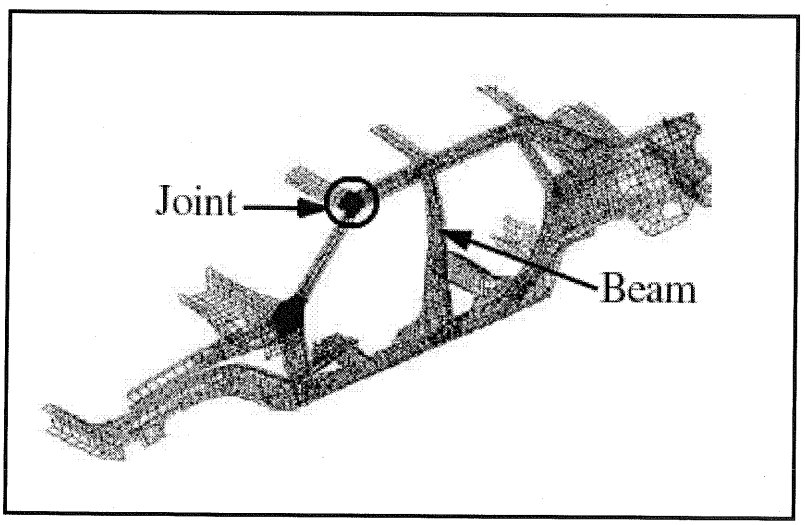


Figure 1.5: A car body as a frame structure consisting of beams and joints [11].

A basic dilemma that an automotive engineer faces is how to combine the desire for low fuel consumption and increased driving comfort at the same time. In order to decrease the fuel consumption they must decrease the weight of the car. But,

increased driving comfort requires a bigger (heavier) car. The same dilemma arises for the engineers of the aeroplane. The weight of the aeroplane should be minimized in order to save fuel and carry more passengers, but at the same time, the aeroplane should be strong enough to withstand storms, turbulence and hard landings. Minimizing the weight of a structure while at the same time satisfying various requirements on structural response such as cost, aesthetics and manufacturing, is a complicated task.

Topology optimization has been applied successfully in the automotive industry [12] for a considerable time and is increasingly becoming a mainstream technology for the design of aircraft components [8]. A reason for this is partly the larger problem sizes and often quite complicated support and loading conditions for aircraft components. However, compliance based topology optimization methods are unable to cope with other design constraints such as buckling – often an important design consideration in the aircraft industry. Typical components that have been constructed for optimization are wing leading edge ribs, main wing box ribs, different types of wing trailing edge brackets as well as fuselage doorstops, fuselage door intercostals and aeroplane floor supports.

Framework structures are widely employed in the mechanical, civil and aeronautical domains. They appear as bridges, towers, pylons, roof supports, building exoskeletons or high technology light space structures (e.g. small satellites). Specifically, the development of topology optimization of framework structures on aeronautical and space related projects such as satellites and spacecrafts would be crucial. Topology optimization methods can help aeronautical and space industry by providing them optimal designs, combining less cost, less weight and more strength at the same time.

1.5. Objectives

A commonly used method of shape and topology optimization is the so-called Evolutionary Structural Optimization (ESO) [14-28]. There is the intuitive appeal that the ESO process attempts to render the stress distribution over the entire structure uniform so that when the structure is loaded all the material points exceed the

allowable stress simultaneously. Therefore, it is only reasonable to hypothesize that removal of relatively under-utilised members will lead to a reduction in weight without much penalty on the increase in maximum stress. In this thesis, we propose to test this hypothesis by observing the trajectory of designs during the ESO process on the weight-maximum stress plane and later overlaying it with the Pareto Front (PF) for the two-objective problem of simultaneously minimizing weight and the maximum stress within the structure. As a plane frame affords an ideal setting to study topology optimization [29-31] —we have therefore kept the scope of the present work limited to planar frame topology optimization.

The effectiveness of the Evolutionary Structural Optimization (ESO) method of designing frameworks is studied further. The designs obtained by ESO are compared with the designs obtained using exhaustive search for combined performance on two counts: the maximum stress within the structure and the overall weight. Numerical experiments were conducted to examine the quality of framework designs produced by ESO. To study the Pareto-efficiency of ESO for various problems, exhaustive search and genetic algorithms (GA) are used as metrics. Further, sizing optimization using the ESO method for framework structures is studied. Cross-sectional areas of members are considered as design variables and the coordinates of the nodes and connectivity among various members are considered to be fixed.

New topology optimization approaches have been developed, so that improvement in both efficiency and computational time can be obtained. In topology optimization, usually there are many local optimal solutions but only one global optimum exists to a given problem. Most of the methods such as ESO cannot perform a global search and thus do not necessarily converge to the global optimal solution for the given objective functions and constraints. Instead of searching for a local optimum, one may want to find the globally best solution in the feasible region. It is known that GAs, which are based on the Darwinian survival-of-the-fittest principle to mimic natural biological evolution, are a stochastic global optimization method. However, GAs are usually computationally expensive, as a very large number of function evaluations is normally required to attain an optimal solution. An improvement in the computational efficiency of the GAs is achieved by finding ways to combine GA and ESO methods.

1.6. Outline of Thesis

The first part of this work, presented in Chapter 2, looks into the literature related to the structural optimization methods. The most important approaches in structural optimization are reviewed. Various families of structural topology optimization methods that have been extensively developed in the last three decades are presented.

The work presented in the next chapters is focused on the topology optimization of frameworks. A plane frame member is considered which combines the characteristics of a truss and a beam because it has axial as well as transverse degrees of freedom. In Chapter 3, the optimal designs of frameworks are obtained by a FEA code written in MATLAB using the idea of the ESO. Various examples are given, leading to reasonable optimal structural designs. Moreover, the effect of scaling the cross-sectional size in framework topology optimization is examined. Simultaneous sizing and topology optimization based on the ESO idea is achieved. In this chapter, general trends that ESO possesses for discrete frameworks are studied in the weight-maximum stress space.

In Chapter 4, the trajectory of designs as obtained by the Evolutionary Structural Optimization are compared with the Pareto-optimal designs for various test problems. The approach adopted is to encode the problem formulation in a genetic algorithm and to allow the formulation to evolve in the direction of improving Pareto optimal designs. A variety of examples are given in order to demonstrate this approach and its usefulness in improving Pareto efficient designs. The designs produced by the Evolutionary Structural Optimization method are compared with the Pareto-optimal designs. The difficulty of dealing with a large number of candidate designs while using exhaustive search is resolved, using a Genetic Algorithm and OptionsMatlab, as GA detects good solutions and eliminates bad solutions in a population.

Chapter 5 presents two strategies for topology optimization of frameworks by combining the Evolutionary Structural Optimization (ESO) method and the Genetic Algorithms (GA). Numerical experiments carried out as a part of Chapter 4 suggest that ESO, when applied to frameworks, is computationally cheap but it misses out the structurally most efficient designs. On the other hand, GA obtains structurally more

efficient designs but is expensive. Two new strategies are proposed and implemented in this work in order to combine the quality of structural design obtained by GA and the computational efficiency of ESO. In the first method called the ESO assisted GA method (ESOaGA), ESO obtained designs are inserted in the GA population, helping GA in the process to converge faster. The second method presented here is a GA assisted ESO method (GAaESO), in which GA produced designs are used as starting points for a family of ESO runs. The designs obtained by the two proposed methods, are compared with the designs obtained using the "unassisted" GA for combined performance on two counts: the maximum stress within the structure and the overall weight. Finally, a comparison between the ESOaGA method and the GAaESO method is made and the multiple use of ESO and GA for further performance improvement is explored. Comparisons of the proposed methods and the "unassisted" GA are carried out visually and quantitatively using quality indicators. The quality indicator results of the methods under comparison are then examined for statistical significance by applying the Kruskal-Wallis test.

Concluding remarks and suggestions for future areas of investigation are presented in Chapter 6.

Chapter 2

Structural optimization methods

2.1. Introduction

Structural topology optimization methods have been discussed in a large number of publications [5, 32, 33] and a review of the relevant literature is presented in this chapter. They can be categorized into discrete element approaches and continuum approaches. First, a general review of the continuum optimization will be given. The continuum approach is classified into two methods. The first approach, the assumed microstructure approach, attempts to find the microstructure parameters (e.g., size and orientation of holes) of each designed element in a finite element model. The second approach assumes no microstructure, but rather heuristically designs the material properties (e.g., Young's modulus and density) of each finite element directly to find optimal material distributions. A more analytic view of these two main optimization approaches will be presented in this chapter. Following this, discrete structural topology optimization will be introduced which is the main subject of this thesis.

2.2. Continuum optimization of the structural system

In continuum methods of analysis, the structure is modelled as a homogeneous piece of matter. Continuum structural optimization differs significantly from discrete structural optimization. The structure is modelled as a solid continuum rather than a finite system of frame members. Discrete techniques deal with a finite number of framework elements each having well-defined length and cross-sectional

characteristics, but on the other hand, continuum techniques deal with spatial arrangements of material that might be difficult to interpret as a system of discrete structural elements. For some large-scale structural problems, the structure must actually be considered as a system of discrete structural elements that are joined together.

While optimizing the spatial thickness distributions of plate structures, regions of "zero thickness" were essentially recognised as "holes" in the plate structure [34]. "It was realised that optimization techniques using spatial distributions of design variables are able to change and even optimize the topology of material distributions in a structure" [35]. Structural topology optimization via distributed parameter optimization techniques was first proposed in [36] and first demonstrated in [37] using a homogenization method. Continuous design variable methods such as the homogenization method, involve complex calculus operations and mathematical programming. The two main approaches of continuum methods that are described next are the microstructure (left hand column of Figure 2.1) and macrostructure approaches (right hand column of Figure 2.1).

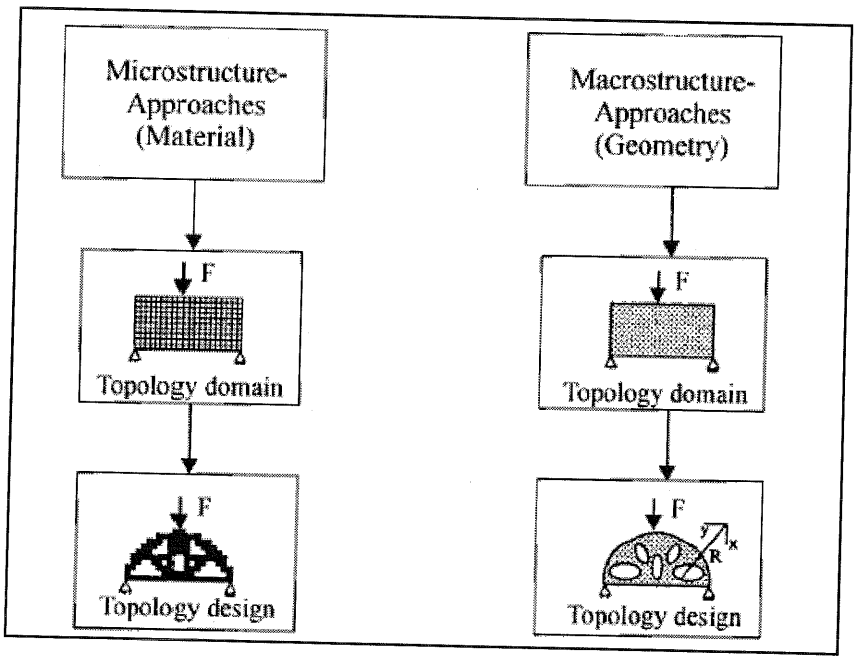


Figure 2.1: Conceptual processes of topology optimization of continuous structures [5].

2.2.1. Microstructure-approaches

The objective of this class of approaches is to find the structural topology, which provides a given design objective an optimum value subject to a given amount of structural material. It is assumed that in solid form the amount of structural material is less than the amount that would be needed to cover the entire admissible domain for the continuum. Hence, for the initial design it is normally chosen to distribute the material evenly in some porous, micro-structural form over the admissible design domain (see left-hand column of Figure 2.1). In the microstructure-approach to topology optimization, one uses a fixed finite element mesh to describe the geometry and the mechanical response fields within the entire allowable design domain. The mesh is considered to be a uniform, rectangular partition of space. The design variables are assumed to achieve constant values within each finite element. For the analysis, the finite element method is applied with constitutive properties that reflect relationships between stiffness components and material density. The optimization consists of determining whether each volume element in the continuum should contain material or not. To this end, the density of material within each finite element is used as a design variable defined between limits 1 (solid material, shown in black in the left-hand column of Figure 2.1) and 0 (void or very weak material indicated by white). Two typical microstructure approaches are the homogenization and material distribution methods discussed next.

i) Homogenization method: The most common microstructure-approach is the homogenization method. The term homogenization means that an inhomogeneous structural element, containing discontinuities in material or geometrical properties, is replaced by a homogeneous but generally anisotropic element, whose stiffness depends on direction but not location within an element [4].

A general method capable of performing simultaneous shape and topology optimization was proposed in [37-40]. The method known as the homogenization method uses representation of a structure with micro-voids and the objective is to seek the optimal porosity of the medium using an optimality criterion. This approach has been applied to optimize structures subject to a single static load [39] and multiple loads [35].

Models for topology design have been viewed as material distribution problems in a fixed domain [41, 42]. It is central to this concept that computations

work with a fixed FEM mesh. This implies that low-density areas are also included in the analysis for each feasible design. For stress constraints this leads to the difficulty of the so-called *stress singularity phenomenon*, where low density regions may have high stress but are structurally insignificant for the final design, rendering computations difficult.

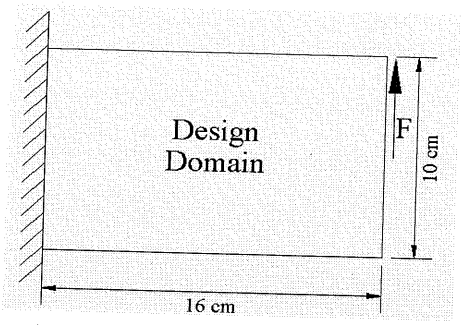
Additionally, there are other methods using the continuous approaches. These include the *cellular automation generation* [43] and *the soft kill option* (SKO) [44]. A "birth-and-death rule" is used to remove members of very low elasticity modulus so that a final design only displays black and white areas.

In the topology optimization of continuum structures, the shape of the external as well as the internal boundaries and the number of inner holes are optimized simultaneously with respect to a predefined design objective. It is assumed that the loading is prescribed and that a given amount of structural material is specified within a given 2D or 3D design domain with given boundary conditions.

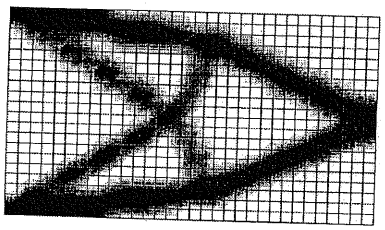
The topology of a design is in most cases chosen either intuitively or inspired by already existing designs. However, there is an interest in improving the quality of the products by finding their best possible topology at a very early stage of the design process. One can distinguish between two classes of approaches, the so-called Material or Micro-approaches and the geometrical or Macro-approaches.

ii) Material distribution method: Another approach is the material distribution method, in which the material density of each member is selected as the design variable. During the optimization process, intermediate density is penalised to force the design variables to approach 0 or 1. If the material density of a member is close to 0 at the end of topology optimization, the member does not exist in the optimal topology.

Figure 2.2 shows the optimal topology using the material distribution method obtained in [45]. As shown in the figure, the result from topology optimization is often a non-smooth, skeleton type of structure, which may not be practical. Further post-processing, either by human interpretation using certain smoothing algorithms [46], or integrating with shape optimization algorithms is required [47].



(a) The design domain and boundary conditions [45].



(b) The result from topology optimization [45].

Figure 2.2

In both homogenization and the material distribution method, the objective functions are usually chosen to minimize compliance of the structure, or to maximize stiffness of the structure. Maximizing the lowest natural frequency of the structure is also used as the objective function in some practical situations. In almost all the examples, the only constraint is that the amount of material that can be used in the design domain is limited. Stress constraints are usually not included in topology optimization, though stress constraints were considered using global stress functions to approximate local stresses in [48].

2.2.2. Macrostructure-approaches

In this category of approaches, solid isotropic material is considered as opposed to the porous, micro-structured one, and the topology optimization is performed in conjunction with a shape optimization (right hand column of Figure 2.1). The finite element mesh cannot be fixed, but must change with the changes of the boundaries of the design. Within the macrostructure-approach, the topology of a solid body can be changed by growing or degenerating material or by inserting holes. The first method recognises that an optimal design is simply a subset of the

admissible design domain and that it can be obtained by adding or removing material from the admissible design domain.

The second method consists of an iterative positioning of new holes ("bubbles") at specific points in the topology domain. In each iteration, the holes and the existing variable boundaries of the continuous body are simultaneously subjected to a shape optimization procedure [5].

2.3. Optimization of discrete structural systems

For discrete structures, the optimum topology involves determining the optimum number and connectivity of the structural members. The optimization of structural systems for maximum stiffness and least weight has been studied extensively, see [30, 39, 41, 49-55]. The first optimum topology (layout) solutions for loads of design dependent locations were the so-called "Prager structures". Prager structures are stress-constrained least-weight trusses where the sign of the member stresses must be the same in all elements and the loads are allowed to move along their lines of action [4]. By saying that the external loads are allowed to move along their lines of action, it is meant that we have a set of loads to be supported by the structure but we do not specify the exact location of the points of application of the loads. In fact what we are seeking is not only the optimal topology for a set of loads and supports but also the optimal point of application of the loads.

The historical development of shape and topology optimization of discrete structural systems such as frames and trusses can be classified into three periods:

(1) In the initial period, pioneering studies have been made in the field [3, 56]. Although the study by Michell is important in view of theoretical background [3], the techniques apply only to limited types of discrete structures and constraints.

(2) During the 1960s and the 1970s, interest in structural optimization grew due to the development of high-speed computers. During this period many important theoretical results for general optimization methods and their numerical implementations were first presented. Difficulties in structural topology optimization were given relatively

more attention. Methods for discrete shape/topology optimization were implemented on very small test problems due to computing limitations.

Early work on topology optimization of trusses was carried out in [57]. All nodal points represented a possible connection for a member truss. A ground structure was defined by connecting all nodal points with truss members. In this work the member forces were assumed to be variables, which led to a linear programming problem. The number of variables in this problem is equal to twice the number of members. The constraints are the equations of equilibrium in all joints of the structure. The solution to this linear programming problem gives the member forces. Cross-sectional areas are then obtained by dividing the absolute value of the member force by the maximum stress. This means that the optimal topology is statically determinate and fully stressed. The members corresponding to non-basic variables are zero, and therefore deleted from the structure. Members corresponding to basic variables are also deleted.

This method was extended to solve statically indeterminate structures subject to multiple loading conditions [58]. Member cross-sectional areas were considered as design variables. The objective function is linear, but the constraints become non-linear functions of the design variables. Members with cross-sectional areas which approach zero were removed from the structure. No proof that these members should not re-enter the optimization process was given [59].

(3) The 1980s and the 1990s are characterised by extremely large growth in computing technologies. Numerical techniques were further developed, and applied to larger-scale, more realistic structures [60]. During this period, while advances in discrete topology optimization continued, continuum structural topology optimization methods were also introduced.

The initial design of discrete structures such as trusses can be broken down into a set of nodal locations, and design of the connectivity of structural elements. The former process is called geometry optimization or configuration optimization and the latter is called topology optimization. Simultaneous optimization of topology and geometry may be referred to as layout optimization or simply configuration optimization [61]. Theoretical works such as explicit optimality criteria approaches and the method based on so-called grillages are summarised in [30] and monographs such as [4, 52, 62-66]. Moreover, unnecessary members are removed from a highly

connected ground structure while the nodal locations are fixed [30, 67, 68]. Many methods have been presented based on this ground structure approach [69].

In paper [14], a simple evolutionary procedure capable of performing simultaneous shape and topology optimization is presented. It is based on finite element analysis to minimize the weight of structures while satisfying stiffness requirements. At the end of each finite element analysis, a sensitivity number, indicating the change in the stiffness due to removal of each member is calculated; members which make the least change in stiffness of a structure are then removed from the structure. Also, a wide range of problems including multiple displacement constraints, multiple load cases and moving loads are considered in this paper [14]. Evolutionary Structural Optimization (ESO) takes advantage of powerful computing technology and thereby has a much simpler formulation. In most implementations of ESO, unnecessary material is gradually eliminated from a structure under one or several specified evolutionary criteria. Clearly, this criterion for material elimination or addition plays an important role in this method.

Another discrete design variable method, which uses binary decision-making algorithms to remove the unnecessary material, is the Simulated Annealing (SA) method [70]. Simulated Annealing has been developed from statistical thermodynamics to simulate the behaviour of the atomic arrangements in liquid or solid material during the annealing process. Lowering the temperature of the melted material, the material reaches to the lowest energy level (global stable condition). Using this concept, simulated annealing has been successfully applied to large scale combinatorial optimization problems in various fields.

The shape annealing method, which uses shape grammar rules with the simulated annealing algorithm to perform shape optimization of trusses, is presented in [71, 72]. Starting with a random initial structure, topology exploration occurs by applying topology modification rules that transform configurations in the current design; metrics for design performance determine the search direction in the simulated annealing algorithm. The shape annealing method is a technique that combines a generative grammar with directed stochastic search using simulated annealing to produce optimally directed designs. A shape grammar is a way of representing the relation between form and function in structural design through the specification of design transformations that define a language of structures. Since the number of structures in this language is quite large, directed stochastic search is used to drive the

generation of efficient designs that satisfy the desirable set of objectives and constraints. The shape annealing is applied in order to obtain simultaneously various design goals such as efficiency, economy, utility and elegance. Shape annealing as a structural design tool has been applied to provide new possibilities for structural forms that may enhance both creativity and insight [72]. Designs that satisfy both the architect's preference for visual impact and the engineer's preference for functional efficiency were obtained. The primary scope of engineers is to focus on the functionality efficiency and construction cost of a structure. On the other hand, architects' approach maybe more aesthetic. A computational method that supports these varying preferences was achieved using shape grammar and shape annealing methods [72].

Having analysed the discrete and continuum optimization methods, a comparison of these two kinds of optimization methods is presented in Table 2.1 below. Generally the discrete methods seem more naturally suited to civil structures using beam/truss type structural members. However, there are difficulties while solving large 3D structural design problems with both methods. The analysis cost produced by the continuum method and the excessive design possibilities by the discrete method for optimization of 3D structures are problems that can not be avoided.

Discrete Optimization	*Continuum Optimization*
1. Single analysis cost trivial, but many analyses required	1. Computational expense: single analysis cost significant, but few designs iterations required
2. Design space (structural possibilities): discrete does not allow many arrangements of members	2. Design space (structural possibilities): continuum clearly allows many more arrangements of members
3. Allows modelling of cross-sections	3. Continuum designs tend to be "unrealistically heavy" due to continuum modelling

Table 2.1: Comparison between 2D Discrete and Continuum Optimization.

2.3.1. Topology Optimization of Frameworks

The aim of framework optimization is to best utilize the material often requiring the lightest structure particularly for aeronautical applications while satisfying all the design and manufacturing constraints. The objective of framework topology optimization is to find, for a given weight, the stiffest or the strongest structure, defined as a subset of an initially chosen set of bars called the ground structure. Alternatively, given the strength or the stiffness the topology that produces the lightest structure may be sought. Topology optimization consists of determining the nature and connectivity of the constitutive members of a structure for which only the boundary conditions and the spatial domain are specified.

Attempts to apply optimization procedures to the design of mechanical structures like frameworks, which are widely used in civil engineering and in space engineering, have been made for a long time. Previous studies on optimal topologies are concerned mostly with frame structures. This may be attributed to the fact that the frame by its nature is most suitable for optimization of the topology. It possesses several nodes and members that can be deleted or retained without affecting the functional requirements. The plane frame is a relatively simple structure. It is therefore an ideal system for the investigation of topology and the search for the optimal [30].

Different possibilities to optimize frames exist. For topology optimization, starting from an arbitrarily initial structure, an improved structure will evolve by changing the member joint connectivities of the structure. Continuous changes in the design parameters are possible in shape optimization, while this is not true in topology optimization. This makes the problem of topology optimization particularly different. Each change of the topology of a framework is a severe operation. Some authors identify topology optimization as the most challenging of the structural optimization tasks [30, 68].

Another approach to topology design is the homogenization method. Here the optimal material distribution in a continuous design domain is interpreted as a discrete plane frame. The mathematical formulation of these methods is based on the minimization of the compliance (maximization of the stiffness) or minimization of the mass. Common to all the methods is that they are restricted to particular objective functions and cannot be used for more general optimization problems. Stochastic

algorithms such as the evolutionary algorithms or the random cost method are more universal optimization methods [73]. They offer the possibility to deal with arbitrary objective functions.

2.3.2. Evolutionary Structural Optimization (ESO)

In recent years the finite element method has become a widely used analysis tool for engineers in many disciplines. Since the early 1990s, scientists have carried out studies on simple approaches to optimal structural design. Structural optimization, however, has not achieved a similar level of popularity in industrial practice despite the progress of optimization theory over the past 40 years. This situation is caused mainly by the mathematical complexities of the existing optimization methods.

ESO is based on the concept of gradually removing redundant members to achieve an optimal design [21]. The Evolutionary Structural Optimization method has been studied for the last ten years and it was found to be efficient for the full-range of structural situations such as topology and size optimization with stress, stiffness, frequency, stability constraints in 2D and 3D with single or multiple loads conditions. Although the idea of member removal has been tried by other researchers [74], these studies have not resulted in a generalized method. The original idea of ESO is that the optimal shape and layout of a structure can be obtained by systematically removing members having low stress values from the structure [75]. Some examples of ESO applied to problems with stress consideration can be found in [21, 28]. This idea has been extended to frequency optimization problems [18], where a sensitivity number for member removal has been introduced and calculated for each member based on information available from the solution of the eigenproblem. By removing members with special values of sensitivity numbers, the specified frequency of a structure can be shifted toward a desired value. Compared with other existing methods, the ESO method [76] is much more straightforward. In fact, it can be easily implemented into any general purpose finite element analysis program. ESO involves no mathematical programming techniques in the optimization process.

A new type of sensitivity number, which indicates the change in the overall stiffness due to removal of a member, is formulated in [18] using results from a finite element analysis. Then a number of members with the lowest sensitivity numbers will be eliminated from the structure. The optimal design of the structure is then obtained

by repeating the cycle of finite element analysis, calculation of sensitivity numbers and member elimination until the overall stiffness reaches its prescribed limit.

The optimal design of structures with frequency constraints [77] is extremely important in the aeronautical industry. For most structures, it is desirable to avoid excessive vibration due to resonance. This can be achieved by manipulating the natural frequencies of the structure so that they are out of the frequency band of the dynamic excitation.

A comprehensive review on the development and applications of structural optimization with frequency constraints was provided in [78]. It is noted that studies of frequency optimization have been restricted in the past to changing the size or thickness of frames, plates, etc. Extensions of these works to the simultaneous shape and topology design of structures with frequency constraints prove to be a tremendous challenge. Little progress has been made, although it is worth noting the contribution of applying the homogenization method to frequency constraints as presented in [39].

An important feature of the ESO method is that it is easy to understand and learn while at the same time producing reasonable results. In addition, with ESO, shape optimization and topology optimization are achieved simultaneously. The optimality constraints can be stress based, stiffness/displacement based, frequency based, bucking load based, with single or multiple environments and the ESO method can be applied to all of them [23].

Size optimization is the earliest form of structural optimization whereby aspects of the cross-sectional size of discrete structural elements are adjusted to give better structural performance. A combination of size and topology optimization is proposed in [16]. If the size of the members is allowed to go to zero then they are removed.

A common cause of structural failure of engineering components is excessive stress or strain. Therefore low stress or strain indicates inefficient use of material. Ideally the stress in every part of a structure is at the same level of factor of safety. This fully stressed design concept leads to rejection criterion based on local stress level, where lowly stressed material is assumed to be under-utilised and will be removed. By gradually removing material with lower stress, the stress level in the new design will become more and more uniform.

First a piece of material, which is large enough to cover the area of the final design, is divided into a fine mesh of finite elements. Loads and boundary conditions

are applied and stress analysis is performed using a finite element program. More often than not, it is found that a part of the material is under-stressed compared to the rest of the structure. Using a prescribed criterion, called a rejection criterion, such inefficiently used material may be eliminated.

Since the structure has been divided into many small elements, the removal of material from structure can be represented by deleting elements from the finite element model.

The stress level at each point can be measured by a representative stress value. For this purpose the von Mises stress has been one of the most frequently used criteria for isotropic materials. For plane stress problems the von Mises stress σ^{vm} is defined as:

$$\sigma^{vm} = \sqrt{\sigma^2{}_x + \sigma^2{}_y - \sigma_x\sigma_y + 3\tau^2{}_{xy}} \quad , \tag{2.1}$$

where σ_x and σ_y are normal stresses in the x and y directions, respectively and τ_{xy} is the shear stress. A justification for the use of the von Mises stress (or any of the similar failure criterion) for the representative value of stress is that it is invariant of the choice of co-ordinate axes.

The stress level of each element is determined by comparing the von Mises stress of the element σ_e^{vm} to the maximum von Mises of the whole structure $\sigma_{\max}^{vm}$.

At the end of each finite element analysis, all the elements that satisfy the following condition are deleted from the model:

$$\frac{\sigma_e^{vm}}{\sigma_{\max}^{vm}} \le RR_i, \tag{2.2}$$

where RR_i is the current rejection ratio.

The cycle of finite element analysis and element removal is repeated using the same value of RR_i until a steady state is reached, which means that there are no more elements to be deleted at the current iteration. At this stage an evolutionary rate (ER) is introduced and added to the rejection ratio,

$$RR_{i+1} = RR_i + ER, \; i=0, 1, 2, 3... \tag{2.3}$$

With this increased rejection ratio the cycle of finite element analysis and element removal takes place until a new steady state is reached. Such an evolutionary process continues until a desired optimum is reached. Ideally the final structure becomes a fully stressed design where the material at each point of the structure is stressed to its full strength. However, only in a few special cases can a fully stressed structure be possible.

The evolutionary procedure requires two parameters to be described. The first is the initial rejection ratio RR_o and the second is the evolutionary rate ER. Typical values of RR_o=1% and ER=1% have been used for many test examples. But for some problems much lower values need to be used. For example, if too much material has been removed from the structure within one iteration or one steady state, it indicates that smaller values should be used for RR_o or ER.

Whilst ESO uses only element rejections, its modified version BESO (Bidirectional ESO, [79]) can also admit new efficient elements. ESO is an iterative method and several runs of finite element analysis and elements removal may be required before the optimum is reached. For this reason, the size of the finite element model becomes an important factor which can affect the solution time. To ensure that there are adequate elements left after repetitive iterations, an oversized initial FE model is needed in ESO and it is divided by a finite element mesh. For large structures, the computational cost of ESO can be very high. Also, considering the fact that elements removed in previous iterations cannot be recovered later in ESO, elements can be deleted prematurely and the evolution may be misled. An attempt to make ESO algorithm more reliable led to a new technique called BESO [79, 80] in which material is allowed to be added during iterations. The final optimum is evolved by removing the low-stressed elements and at the same time adding elements around those of high stress. BESO has been applied to continua problems of stress, stiffness displacement and frequency constraints [79, 80].

Furthermore, another approach to member removal is proposed [15], since member and node removal is awkward, as there are many possible alternatives to remove a member or node. The use of an imaginary bar to replace the removed member is one such example. An imaginary bar will have a cross-sectional area, which reaches zero. A tiny value is assigned to this bar to keep the mesh dimension of the FE model. As the cross-sectional area has a very small value, the difference in the

structural stiffness and mass matrices before and after removing the imaginary bar is very small and can be ignored.

2.3.3. Influence of the member removal ratio (*MRR*), mesh size and member type

Apart from the rejection ratio (*RR*) and evolutionary rate (*ER*), another parameter of the ESO process is the member removal ratio (*MRR*). The ratio of the number of members removed in each iteration during the ESO procedure to the total number of members is called the *member removal ratio*. In [81], a new member removal ratio was developed for ESO. The member removal ratio of ESO is fixed throughout topology optimization at 1 or 2%. A new *MRR* for ESO was developed in order to improve the convergence rate.

The smaller the value of the member removal ratio, the better is the quality of the final design [81]. The use of a larger member removal ratio will reduce the number of members of the resulting design more rapidly, so the time for each subsequent iteration will decrease. When the member removal ratio varies from 1% to 4%, it has little effect on the weight and the outer shape of the optimal design. The member removal ratio does affect the details of the inner parts; however, the main pattern and orientation of these inner parts are similar. It is suggested in [81] that one could use a member removal ratio as high as 4% to obtain optimal shape and topology with sufficient accuracy and significant computational time saving.

The mesh size has little effect on the weight of the evolved designs, even though it affects the details of the final design. Even a coarse mesh can provide a rough idea of the shape and topology of the optimal design.

The type of members with similar sizes has almost no effect on the weight; however, it has minor influence on the shape and topology of the optimal design.

2.4. Metamorphic Development: A new topology optimization method

Another recently proposed method in the field of topology optimization is Metamorphic Development (MD) [82]. This optimization method starts from the simplest possible geometry rather than from a complex ground mesh. The structure is

then developed using rectangular and triangular members that can be of any specified size.

This method can overcome some of the disadvantages that ESO and the homogenization method have. ESO has usually no mechanism for re-introducing a member once removed. Although some researchers have started to introduce a member-adding capability by reinstating the property values of some members in a ground grid, the method is still based on a large dense FE mesh [82]. On the other hand, the homogenization method introduces composites with perforated microstructures of continuously varying density and orientation as admissible materials for the structural design. These density function methods can only produce a blurred, grey-scaled structure. The most noteworthy disadvantage of both the evolutionary method and the homogenization method is that the resulting design can only be formed by degenerating a dense ground mesh. These methods can also produce a structural layout with non-smooth boundaries, which may lead to notch stresses and incorrect optimization results. Moreover, these methods can be computationally very expensive, particularly if most of the original ground structure has to be removed during optimization.

The MD method [82] can be used for both trusses and continuum structures and also for combined truss/continuum structures. In this method, the metamorphic development of a continuum structure starts from a very basic description of the structure: just the specification of a minimal number of nodes and members connecting the applied loads and support points. A dense FE mesh is not required. The method produces not only the lightest structure but also a clear and simple structure with maximum integral stiffness or minimum elastic compliance at equilibrium. During the optimization procedure, members that do not effectively contribute strength and stiffness to the structure are removed, and new members are added to the structure in the positions they are most needed. Since both rectangular and triangular members are used to build a structure, the interior and exterior boundaries of the optimized structure are quite smooth. In addition to the usual requirements of structural strength and stiffness, the development of the structure satisfies kinematic stability requirements (Figure 2.3).

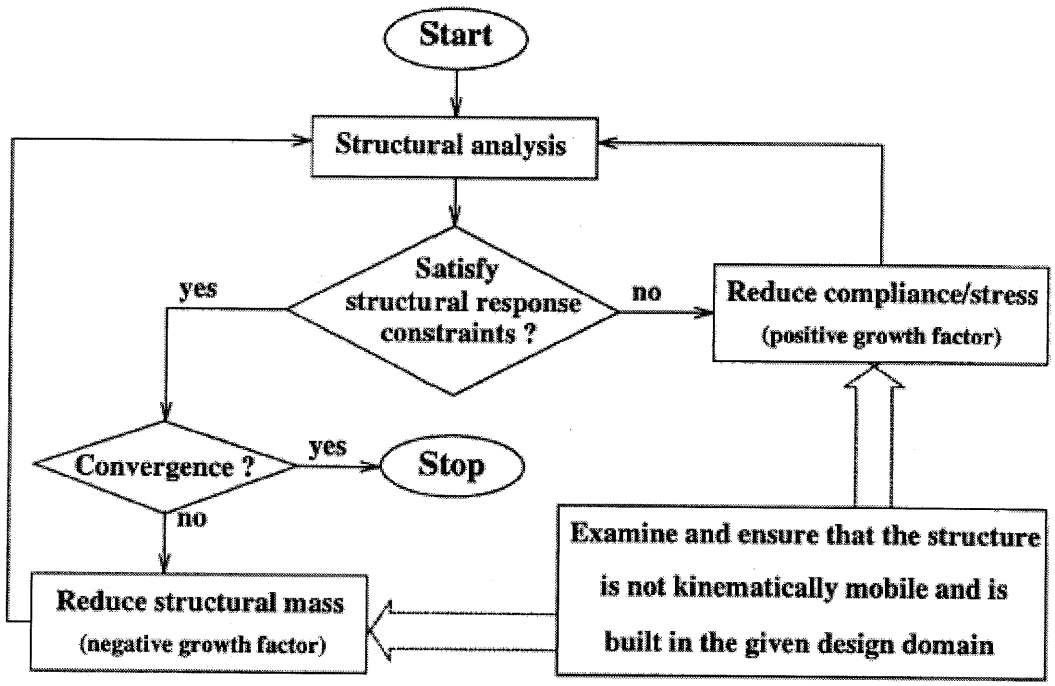


Figure 2.3: Flow chart of the MD method developed in [82].

2.5. Multi-objective Optimization

Most real engineering problems have several objectives to be simultaneously optimized. For example, the design of a structure may include objectives related to material cost, manufacturing cost, failure cost, etc. A multi-objective optimization problem is a problem of finding a set of design variables, which satisfy the specified constraints and optimize multiple objective functions [83-87]. These functions form a mathematical description of performance criteria that are usually in conflict with each other. Hence, the term "optimize" means finding a solution which would give values of all the objective functions acceptable to the decision maker.

The notion of optimum in multi-objective optimization was clarified in the original context of economics [88]. Such designs are now called "Pareto-optimal". Pareto optimality can be seen as the basic multicriteria optimization concept in virtually all of the previous literature. A general multi-objective optimization problem is to find the vector of design variables $X=(x_1, x_2, \ldots, x_N)^T$ that minimizes a vector objective function $F(X)$ over the feasible design space X. Pareto optimality can be

stated as follows: "A design is Pareto-optimal if there exists no feasible design which would improve one of the objectives without causing a simultaneous worsening in at least one other objective" [5]. A trade off between competing objectives is then required. This concept almost never provides a single solution, but rather a set of solutions called the Pareto-optimal set.

2.6. Genetic Algorithms

Multi-objective optimization problems can be solved by a heuristic technique inspired by the mechanics of natural selection known as genetic algorithms. Current research is focused on developing new and effective/robust evolutionary algorithms to solve discrete optimization and combinatorial optimization problems because many complex optimization problems (such as structural topology design optimization) are discrete in nature. Evolutionary techniques such as *genetic algorithms* are favourable because they can handle discrete and combinatorial problems, and at the same time provide multiple solutions with a good chance of achieving the global optimum [89]. As the handling of constraints is still a difficult issue in discrete problems [90], new techniques are being developed in conjunction with the treatment of multi-objective problems. In addition, stochastic optimization methods such as simulated annealing are also applied as these methods can treat discrete problems, sometimes with lesser computational effort than evolutionary techniques (but obtaining only single solutions). Genetic algorithms have been applied to various areas, such as:

(i) Design synthesis of compliant mechanisms, micro-sensors/actuators and micro-machines by structural topology/shape design optimization.

(ii) Structural design optimization (especially topology, shape and sizing optimization) of aerospace, civil and mechanical structures for minimum weight, stress, compliance and/or optimum frequency, stiffness, heat transfer, etc.

(iii) Aerodynamic shape design optimization.

(iv) Design optimization of manufacturing processes such as injection moulding, metal forming, etc.

Genetic algorithms are a part of evolutionary computing, which is a rapidly growing area of research. They are inspired by Darwin's theory of evolution. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved. Evolutionary computing [91] was introduced in the 1960s. The idea was then developed by other researchers. Genetic Algorithms (GA) were invented in 1975 and developed in [92]. In 1992 a new method called "genetic programming" (GP) was developed in which genetic algorithms were used to evolve programs to perform certain tasks [93]. The genetic programming paradigm provides a way to genetically breed a computer program to solve a wide variety of problems. Genetic programming starts with a population of randomly created computer programs and iteratively applies the Darwinian reproduction operation and the genetic crossover (sexual recombination) operation in order to breed better individual programs. LISP programs were used, because programs in this language can be expressed in the form of a "parse tree", which is the object the GA works on. With GA the best solution among a number of possible solutions is obtained.

2.7. Conclusions

Structural topology optimization is a powerful tool which can help the designer select suitable initial structural topologies and, more importantly, it is identified as economically the most rewarding task in structural design. Structural topology optimization as a generalized shape optimization problem has received considerable attention recently.

Summing up, various families of structural topology optimization methods have been extensively developed. One of the most established families of methods is the one based on the homogenization approach. As an important approach within this family, the power-law approach, which is also called the SIMP (Solid Isotropic Micro-structure with Penalization) method, has gained a fairly general acceptance in recent years. It adopts the member relative density as the design variable and assumes that the material properties within each member are uniform, which are modeled as

the relative material density raised to some power times the material properties of solid material.

This chapter has also described the ESO and its modified version of BESO. A well-developed family of structural optimization methods is the one based on the Evolutionary Structural Optimization (ESO) approach, in which the material in a design domain which is not structurally active is considered as inefficiently used and can thus be removed by using some member rejection criteria. This method was further developed by allowing material to be added as well as removed leading to a new approach called Bidirectional ESO (BESO).

Both the homogenization method and the ESO have been further developed by a number of researchers, leading to the extensive exposition and exploration of these two families of methods. Although computationally effective, neither can perform a global search and thus do not necessarily converge to the global optimal solution for the given objective function and constraints.

Another emerging family of structural topology optimization methods is the one using Genetic Algorithms (GAs), which are based on the Darwinian survival-of-the-fittest principle to mimic natural biological evolution. GAs have been gradually recognized as a kind of powerful and robust stochastic global search method. More recently, GAs have been increasingly employed in the structural topology optimization field in order to perform a global search in the design domain. This dissertation provides a systematic study of ESO and GA, a quantitative critique of ESO and further presents two new methods of combining the strengths of ESO and GA.

Chapter 3

Evolution of maximum stress and weight during ESO: general trends

3.1. Introduction

Compared with other existing methods of structural optimization, the implementation of the ESO method is fairly straightforward around any general purpose finite element analysis program. It is reasonable to expect that removal of relatively under-utilised members will lead to a reduction in weight without much penalty on the increase in maximum stress.

In this chapter, we propose to test this hypothesis by observing the trajectory of designs during the ESO process on the weight-maximum stress plane. Initially, a program developed in MATLAB is developed in order to simulate a 2-D framework fully connected and its deformed shape under any boundary conditions and applied loads. Further ESO is implemented using this finite element code. Various examples are given, leading to reasonable optimal structural designs. The graph of the current structure's weight against the current maximum stress of the structure for the whole optimization procedure is plotted for each example so that a general trend that ESO possesses can be obtained. The purpose of this is to examine the trajectory in which designs evolve during the ESO process so that later these can be overlaid with the Pareto Front for the two-objective problem of simultaneously minimizing weight and the maximum stress within the structure (see Chapter 4) and hence the efficiency of ESO can be assessed. Furthermore, a simultaneous sizing and topology optimization is obtained, while scaling the cross-sectional size of the structure during the ESO process.

3.2. Finite Element Modelling and Simulation of Frameworks

In order to study the general trends of the quality of designs produced by the ESO process, weight-wise and maximum stress-wise, an ESO algorithm was implemented in the MATLAB environment. But before we proceeded to the implementation of the ESO algorithm, a Finite Element Analysis code was initially developed to simulate a 2-D framework that can have any number of nodes, boundary and loading conditions and obtain all the necessary stress, strain and nodal displacements calculations. An example of a framework structure considered during the finite element modelling and the ESO method can be seen in Figure 3.1.

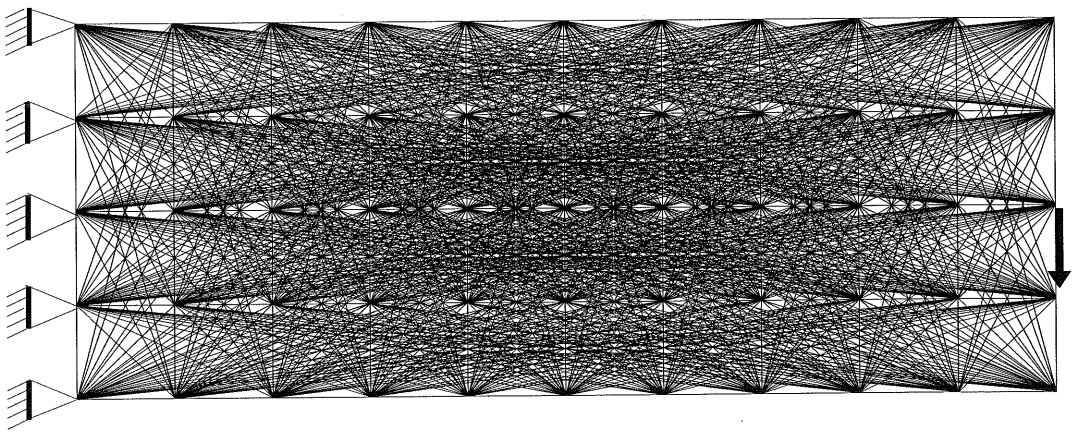


Figure 3.1: Fully connected framework with 55 nodes.

Application of the stiffness method of structural analysis requires subdividing the structure into a set of finite members, where the endpoints are called nodes. For the case of plane frames, we will consider each member as a finite element, and each joint becomes a node. We will determine strain energy and the external work done for each member separately, and then combine each individual contribution to the whole structure which results in a global structural stiffness matrix after applying the principle of minimum total potential energy. This procedure of summing up energy is known as assembly. The stiffness of each member is transformed from the local level into the global coordinate system. We now have the contribution of a single member. For a structure with multiple members, we assemble the stiffness matrix $[K]$ for each

member, and then place the entries appropriately into a global structural stiffness matrix that covers every degree of freedom in the entire framework. The global stiffness matrix will be a square matrix with as many rows as columns as there are total degrees of freedom.

3.2.1. Finite Element Modelling

The finite element modelling of the framework is described next. First, we look at a single member in a local coordinate system, and define its stiffness. A plane frame member will be considered in this work. This member combines the characteristics of a truss and a beam. It is clear that a beam member has four degrees of freedom, a transverse displacement and a rotation at each node. On the other hand, a plane truss member has four degrees of freedom as well, one horizontal and one vertical displacement at each node. The plane frame member has six degrees of freedom, three at each node (two displacements and a rotation). Consequently for a structure with N nodes, the global stiffness matrix will be of size $3N \times 3N$. Considering the member in Figure 3.2, the local displacements at the first node are labelled q_1, q_2 and q_3, and at the second node q_4, q_5, q_6 correspondingly. There is no need to further discretize the structure when applying FEM to a frame problem if the prescribed loads are applied at the nodes. This is because the exact elastic response is obtained for such problems with two noded pin-joined truss members. Therefore, each member is a finite element and the joints are the nodes. Forces are applied only at the nodes. Framework members are subjected to either axial tension or compression and at the same time carry bending moments. The plane frame member is a two-dimensional finite element with both local and global coordinates. Each member has modulus of elasticity E, second moment of cross-sectional area I, cross-sectional area A, and length L. In addition, each member has two nodes and is inclined with an angle θ as can be seen in Figure 3.2.

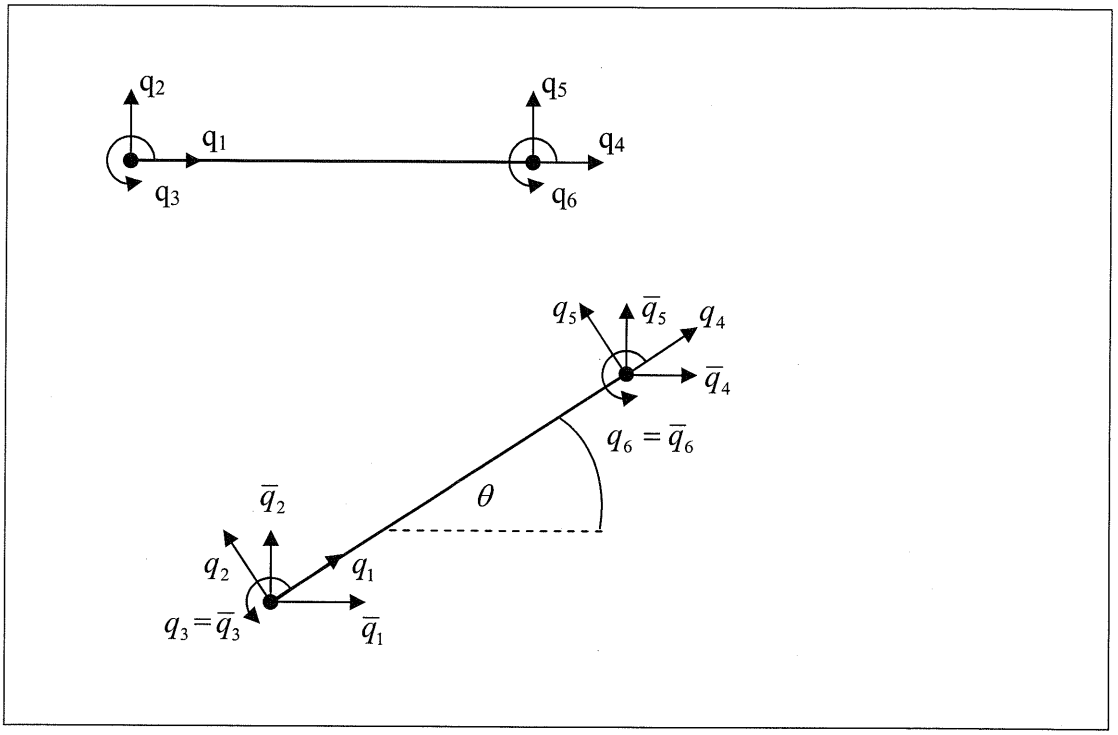


Figure 3.2: Nodal displacements of a plane frame member.

Since a structure consists of many members in many orientations, we will transform all local displacements into a single uniform global coordinate system. If the local displacements of a plane frame member are q_1, q_2, q_3, q_4, q_5 and q_6 then the corresponding global coordinates $\overline{q}_1$, $\overline{q}_2$, $\overline{q}_3$, $\overline{q}_4$, $\overline{q}_5$ and $\overline{q}_6$ can be found by the following relationships:

$$\begin{aligned}
\overline{q}_1 &= q_1 \cos\theta + q_2 \sin\theta \\
\overline{q}_2 &= -q_1 \sin\theta + q_2 \cos\theta \\
\overline{q}_3 &= q_3 \\
\overline{q}_4 &= q_4 \cos\theta + q_5 \sin\theta \\
\overline{q}_5 &= -q_4 \sin\theta + q_5 \cos\theta \\
\overline{q}_6 &= q_6
\end{aligned}$$

The global displacements are defined by the relationship $\{\overline{q}\} = [T]\{q\}$, where T is the transformation matrix. This can consequently be organised in a matrix form as follows:

$$\begin{Bmatrix} \bar{q}_1 \\ \bar{q}_2 \\ \bar{q}_3 \\ \bar{q}_4 \\ \bar{q}_5 \\ \bar{q}_6 \end{Bmatrix} = \begin{bmatrix} c & s & 0 & 0 & 0 & 0 \\ -s & c & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c & s & 0 \\ 0 & 0 & 0 & -s & c & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \end{Bmatrix}, \tag{3.1}$$

where $c = \cos\theta$ and $s = \sin\theta$.

If the nodal displacement for the i^{th} member is: $\{\bar{q}\} = [T(\theta_i)]\{q\}$ then the stiffness matrix for the i^{th} member is:

$$[\bar{K}_i] = [T(\theta_i)]^T [K_i][T(\theta_i)] , \tag{3.2}$$

where $[\bar{K}_i]$ and $[K_i]$ are the stiffness matrices for the i^{th} member in global and local coordinates correspondingly.

The local stiffness matrices for axial and bending degrees of freedom are given by

$$[K]_{axial} = \frac{EA}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \text{ and } [K]_{Bending} = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}.$$

Combining both matrices, the local stiffness matrix for a member with 3 degrees of freedom per node is given by

$$[K]_{Local} = \frac{EI}{L^3}\begin{bmatrix} \alpha & 0 & 0 & -\alpha & 0 & 0 \\ 0 & 12 & 6L & 0 & -12 & 6L \\ 0 & 6L & 4L^2 & 0 & 6L & 2L^2 \\ -\alpha & 0 & 0 & \alpha & 0 & 0 \\ 0 & -12 & -6L & 0 & 12 & -6L \\ 0 & 6L & 2L^2 & 0 & -6L & 4L^2 \end{bmatrix} \tag{3.3}$$

where $\alpha = \frac{L^2 A}{I}$.

The equilibrium equation in the global coordinates is given by: $\{\bar{f}\} = [\bar{K}]\{\bar{q}\}$, where $[\bar{K}]$ is the stiffness matrix for each member in global coordinates, $\{\bar{f}\}$ is the corresponding force vector and $\{\bar{q}\}$ is the corresponding nodal vector.

The same equation in the local coordinates is given by: $\{f\}=[K]\{q\}$, where $[K]$ is the stiffness matrix for each member in local coordinates, $\{f\}$ is the corresponding force vector and $\{q\}$ is the corresponding nodal vector. If we use the transformation matrix to relate local vector quantities to the corresponding global quantities we have:

$$\{\bar{f}\}=[T]\{f\}$$

$$\{\bar{q}\}=[T]\{q\},$$

where $[T]$ is the transformation matrix.

Then substituting, $\{\bar{f}\}=[T]\{f\}=[\bar{K}]\{\bar{q}\}=[\bar{K}][T]\{q\}$

$\{f\}=[T]^{-1}[\bar{K}][T]\{q\}=[T]^{T}[\bar{K}][T]\{q\}=[K]\{q\}$ leading to the global stiffness matrix:

$$[K]=[T]^{T}[\bar{K}][T]. \tag{3.4}$$

3.2.2. Computer code validation

An initial configuration is generated by connecting all the N nodal points to each other to construct the ground structure. In the ground structure, we start from a dense mesh of N nodal points in 2-D and each of these nodes can be connected by a bar. Thus we work with N nodes with up to $N\times(N-1)/2$ potential bars. After completing the structural connectivity the computer code written in MATLAB performs the following steps:

Step 1: Creation of local stiffness matrix, transformation to global co-ordinates using equation (3.4), followed by assembly of members. This requires taking care of the local node numbering and the corresponding global node numbering.

Step 2: Apply boundary conditions to the global stiffness matrix, leading to the reduced global stiffness matrix K.

Step 3: All the global nodal displacements are then calculated by solving the equation $Q_{global} = K^{-1}{}_{global} F$, where F is the force matrix of the structure and Q_{global} is the matrix that includes all the global nodal displacements. In practice, the stiffness matrix is not explicitly inverted but a simultaneous solution is sought.

Step 4: The global nodal displacement is added to the corresponding coordinate of each node, obtaining the new coordinate of each node, thus obtaining the deformed shape of the structure.

An example is provided next in which the number of nodes being used is N=55. This means that each node is connected to the remaining 54 nodes, having in total 1485 frame members. This fully connected structure is presented in Figure 3.1. Even the frames crossing the internal nodes and those connecting the clamped nodes are generated. The particular framework is constructed of steel tubular members with diameter of 0.02 m and it supports a load of 10 kN at the middle of the right edge. Applying the corresponding FEA code a plot of the deformed structure of Figure 3.1 is obtained and illustrated in Figure 3.3. The new nodal coordinates were attained by summing the initial nodal coordinates with the new nodal displacements scaled by 10,000.

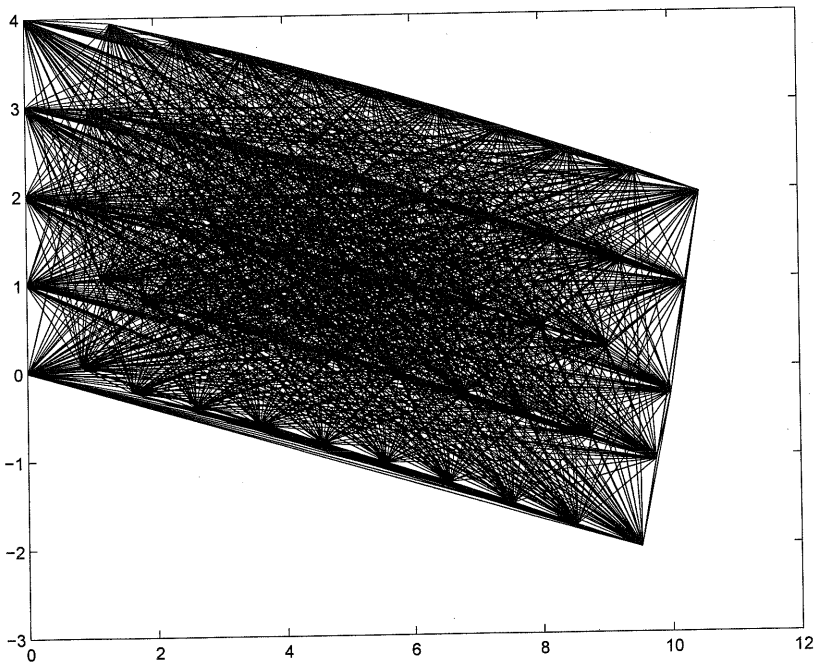
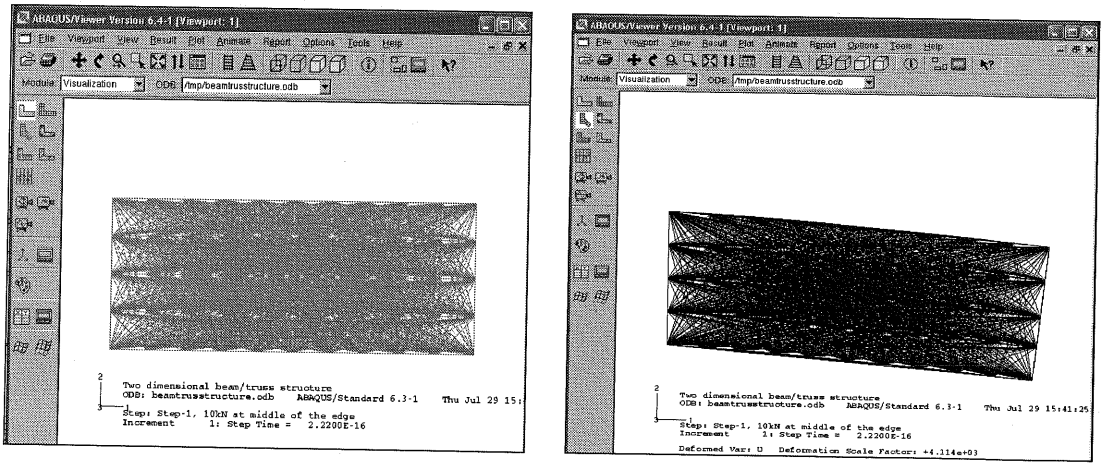


Figure 3.3: Deformed framework obtained in MATLAB.

The results from the finite element program written in MATLAB are compared with the results taken from the commercial FEA software ABAQUS for code validation. The figures of the framework and its deformed shape, obtained by MATLAB (Figures 3.1 & 3.3) match with the corresponding figures obtained by ABAQUS (Figure 3.4). Apart from that, the ABAQUS results also match numerically with the MATLAB output. A smaller framework is considered in order to view the similarity in the results produced by ABAQUS and MATLAB. The particular structure has 9 nodes and each node connected to each other (36 members in total). The values of the nodal displacements for each node of this deformed framework found from MATLAB are compared with the corresponding values taken from ABAQUS and found to be the same (see Tables 3.1 & 3.2). The values are close to within 2%. Some small non-zero values in Table 3.2 are of the order of machine-epsilon. The validation proved that the MATLAB based FEA implementation is working successfully. A reason for developing our own finite element code and not relying on the commercial software such as ABAQUS is a greater access to the variables in the FE calculations which commercial codes often do not afford or allow only in a complicated way. Further, combining features of ESO with GA proved to be much easier than it would have been with a commercial code. Having illustrated that a newly developed procedure works in principle, commercial implementation can be taken up at a later stage.



(a) (b)

Figure 3.4: a) Framework presented in ABAQUS, b) Deformed framework presented in ABAQUS.

Number of Nodes	Nodal Displacements-ABAQUS results		
	q_1	q_2	q_3
1	0	0	0
2	0	0	0
3	0	0	0
4	-4.8471E-05	-6.6585E-05	-6.9386E-05
5	0.	-5.6925E-05	-6.8601E-05
6	4.8471E-05	-6.6585E-05	-6.9386E-05
7	-5.5958E-05	-1.4247E-04	-6.9369E-05
8	0.	-1.7944E-04	-9.4703E-05
9	5.5958E-05	-1.4247E-04	-6.9369E-05

Table 3.1: Nodal displacements of a fully connected 9-node structure obtained by ABAQUS.

Number of Nodes	Nodal Displacements-MATLAB results		
	q_1	q_2	q_3
1	0	0	0
2	0	0	0
3	0	0	0
4	-4.8471E-05	-6.6590E-05	-6.8193E-05
5	-4.6228E-21	-5.6928E-05	-6.7657E-05
6	4.8471E-05	-6.6590E-05	-6.8193E-05
7	-5.5957E-05	-1.4247E-04	-6.8993E-05
8	-5.0138E-21	-1.7944E-04	-9.2156E-05
9	5.5957E-05	-1.4247E-04	-6.8993E-05

Table 3.2: Nodal displacements of a fully connected 9-node structure obtained by MATLAB.

3.3. Implementation of the ESO algorithm to framework topology design

We start from a plane frame in which each key point is connected to every other key point by an elastic member. Members are discarded from this over-specified frame in the spirit of ESO. This procedure can be thought of as the 'discrete' version of ESO. Instead of discarding members from the continuum, the suggestion is to discard complete structural members during the process. The nodes that are not connected to any members during ESO cycles are totally removed from the structure.

The objective of our optimization is to select from this network the most profitable set of nodes and bars able to carry given load under any boundary conditions.

The computer code that was developed to implement the ESO procedure is summarized next:

Step 1 to Step 4: The same procedure is followed as in Section 3.2.2.

Step 5: The axial and bending stress for each of the two faces (top and bottom) on either side of the neutral axis of the member is calculated. Because the stresses in a framework are due to axial tension-compression as well as bending effects, the strains due to these two are superposed and the worst case will always appear on one side of the neutral axis at fibres farthest away. The maximum absolute value of the combined stress (due to bending and tension/compression) is recorded for each member. This process is carried out for all the members of the structure. A prescribed fraction of the current number of members that have minimum stress is removed from the structure.

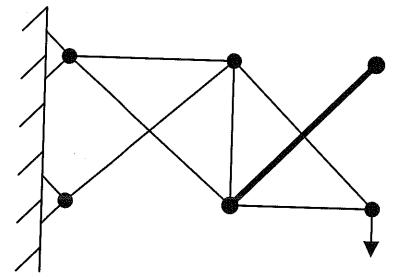
Step 6: The nodes that have no members connected to them are removed from the structure. Following this, the nodes and members of the remaining structure have to be *renumbered* for a new FE analysis. As the ground structure for topology optimization is very redundant, generally speaking, there are many possible nodal layouts and for a specific nodal layout there are many candidate frameworks. To ensure that the structures being generated during the ESO process possess the support and loading conditions imposed by the original design problem, care must be taken while removing members. In the ground structure, some nodes can be removed, but others cannot be, such as those associated with supports, external loads, etc. (Figure 3.5). Consequently, the boundary condition nodes are maintained during the optimization process.

To trace the changes in structural topology and to avoid singularities in the stiffness matrix due to removal of members, updating of the FE model is needed. This is a serious problem when truss models are to be used because it amounts to detecting mechanisms—fortunately we escape this by using frame members that are not pin-jointed at the ends. At every iteration step the FE model is automatically altered. In this part of the process, an examination must take place to check if there is a node, which is only connected to one member. A node with no external loads and boundary

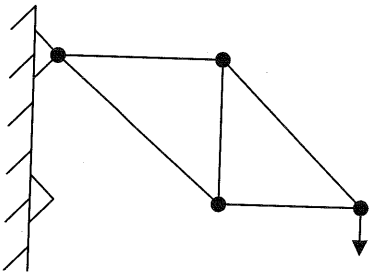
conditions, which is only connected to one member, is considered useless in the topology and it is removed (Figure 3.5).

The appearance of hanging members could cause problems in the framework optimization examples that have been already produced. These hanging members are stress-free, therefore, the ESO algorithm detects and eliminates them.

Step 7: Step 1 to Step 6 are repeated until the structure has 30% of the initial number of members. This fraction is somewhat arbitrary and is based on trial and error and is chosen on the basis of similar work in the area.



Needless Members



Unstable Structure

Figure 3.5: Undesirable structures.

Next we study this ESO algorithm by considering several examples based on the MATLAB code that is developed upon the ESO idea. The flow chart of the ESO approach used here is shown in Figure 3.6.

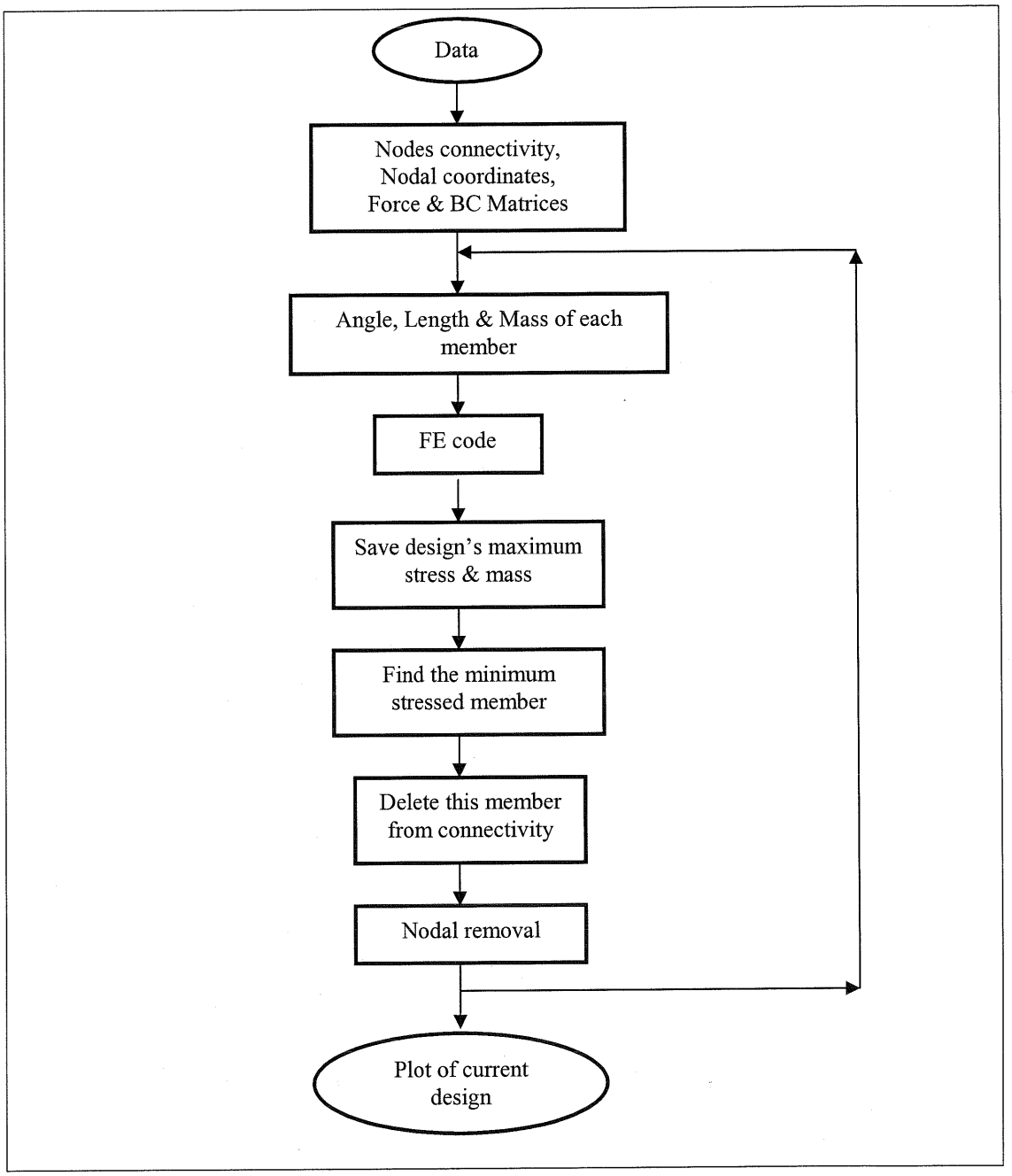


Figure 3.6: Flow Chart of discrete ESO method.

Numerical Examples

(a) Fully connected 6-node cantilever framework

Figure 3.7(i) shows the initial ground structure of the fully connected 6-node cantilever framework. The three vertical nodes at the left edge are considered fixed.

An external load is applied at the middle node of the right edge. By removing at each step a number of members that have the lowest stress values and using the previously mentioned strategy, an optimal design for the current structure is obtained (Figure 3.7(vii)). As for all demonstrated structural examples the BC nodes are kept unchanged during optimization.

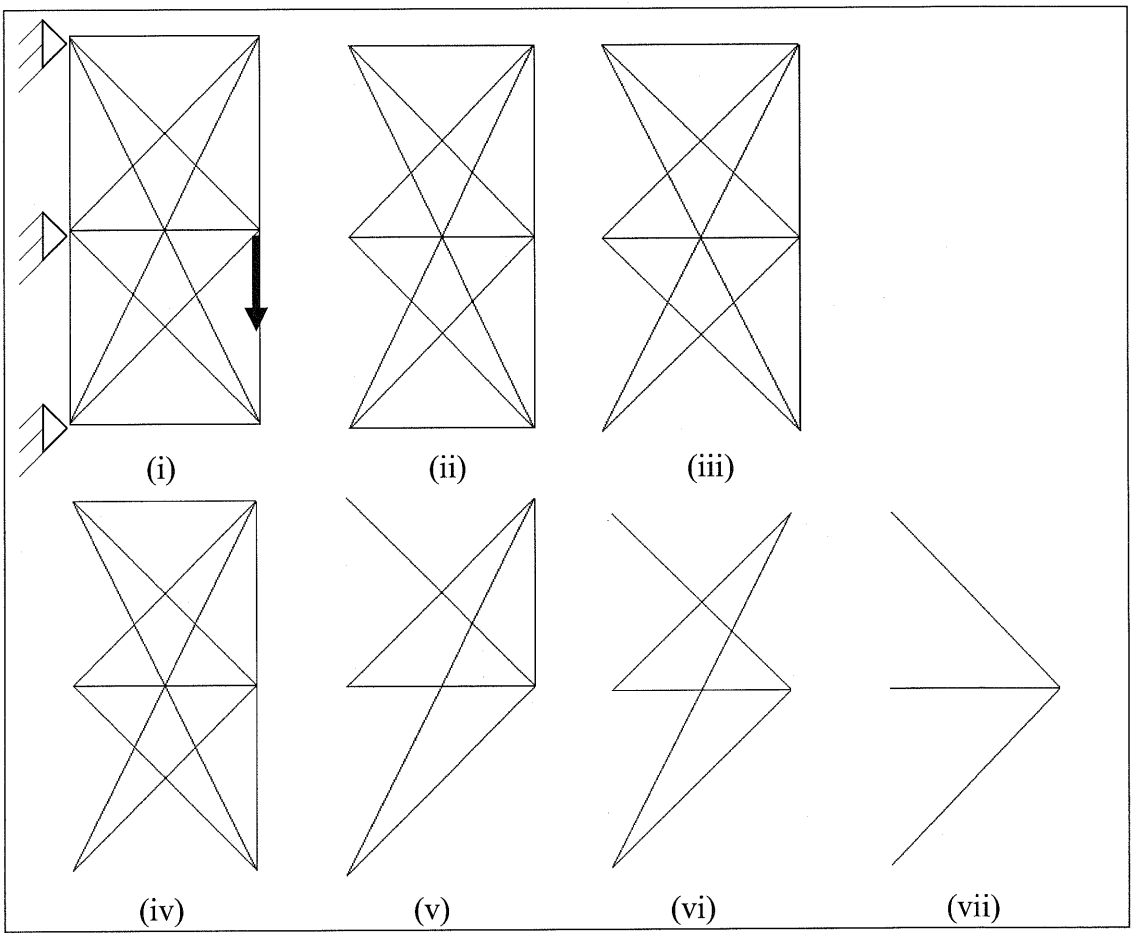


Figure 3.7: (i) Ground structure and first topology for the fully connected 6-node structure, (ii)-(vi) Intermediate results, (vii) Optimal topology for the fully connected 6-node framework obtained by the ESO process. Note that (iii) and (iv) are not the same structures because the members removed from structure (iii) are collinear.

(b) Squared 9-node cantilever framework with adjacent connectivity and 2 fixed supports

Figure 3.8(i) shows the ground structure of the 9-node cantilever framework with adjacent connectivity. The two vertical nodes at the top and bottom of the left edge are considered fixed. An external load is applied at the middle node of the right

edge as shown. By removing at each step a number of members that have the lowest stress values and using the strategy as described before, an optimal design for the current structure is obtained (Figure 3.8(iv)).

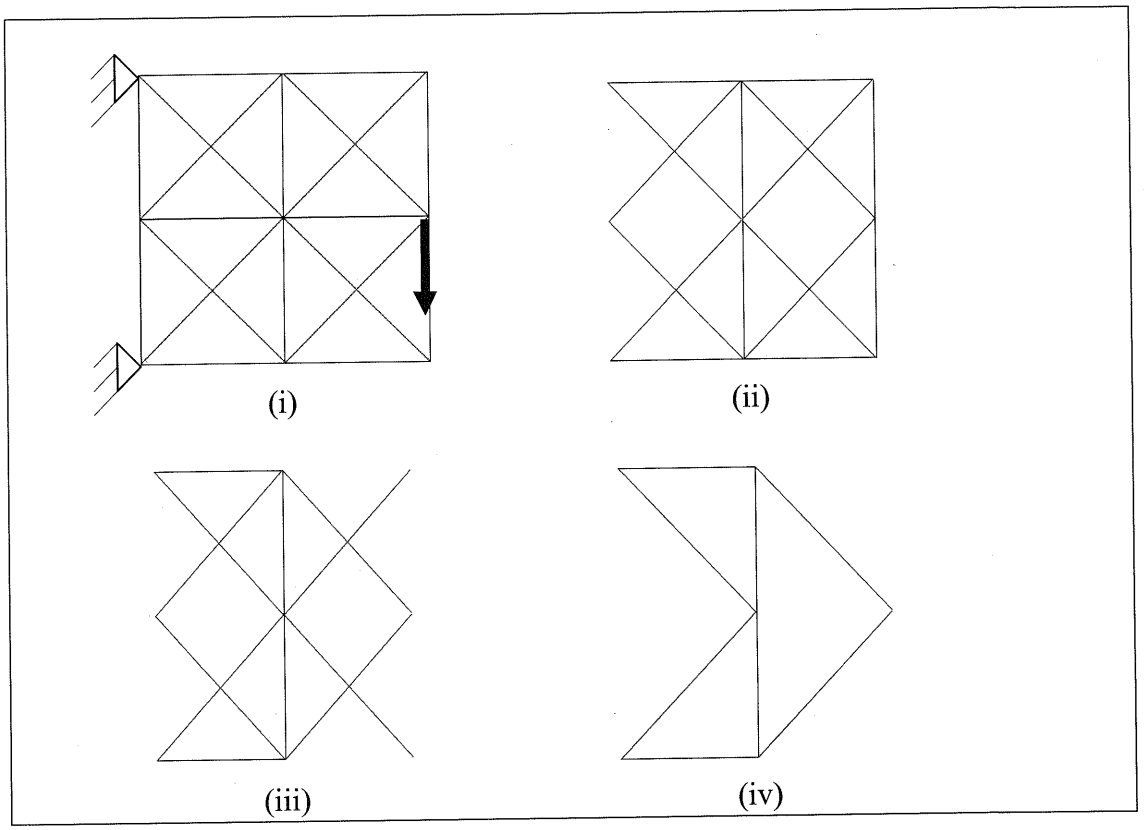


Figure 3.8: (i) Ground structure and first topology for the 9-node structure with adjacent connectivity (2 fixed supports), (ii)-(iii) Intermediate results, (iv) Optimal topology for the 9-node structure with adjacent connectivity (2 fixed supports) as obtained by the ESO process.

(c) Squared 9-node cantilever framework with adjacent connectivity and 3 fixed supports

Figure 3.9(i) shows the initial ground structure of the 9-node cantilever framework with adjacent connectivity. The three vertical nodes at the left edge are considered fixed. An external load is applied at the middle node of the right edge. By removing at each step a number of members that have the lowest stress values and using the strategy as described before, ESO generated designs are produced and are displayed in Figures 3.9(ii) through to 3.9(v).

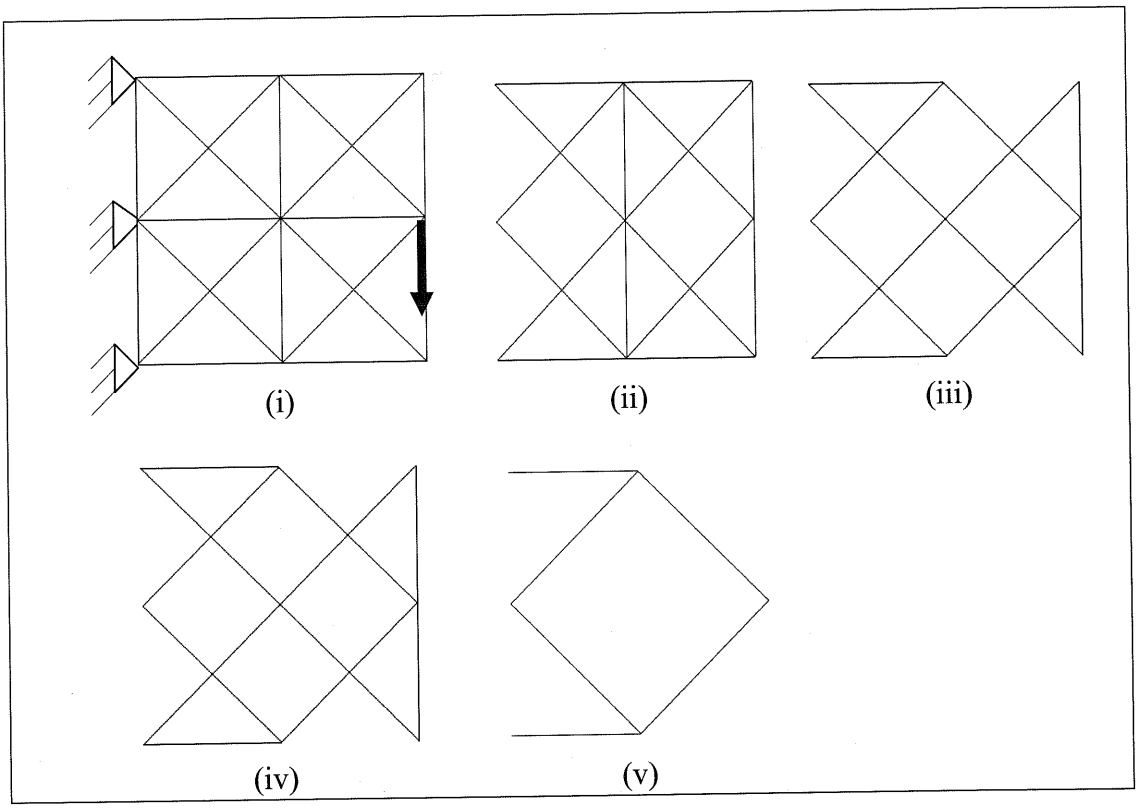


Figure 3.9: (i) Ground structure and first topology for the 9-node structure with adjacent connectivity (3 fixed supports), (ii)-(iv) Intermediate results, (v) Optimal topology for the 9-node structure with adjacent connectivity (3 fixed supports) as obtained by the application of ESO. Note that (iii) and (iv) are not the same structures because the members removed from structure (iii) are collinear.

Since ESO works by eliminating members from an existing structure, we study what effect the initial structure may have on the final design that is obtained. The optimal topology obtained by an adjacently connected structure can be compared with the optimal topology that can be gained by a corresponding fully connected structure.

(d) Squared 9-node cantilever framework with full connectivity and 3 fixed supports

Figure 3.10(i) shows the initial ground structure of the 9-node cantilever framework with full connectivity. The BC's and the force are applied at the same nodes as in the previous example, but all the nodes are connected with each other. The same procedure is applied.

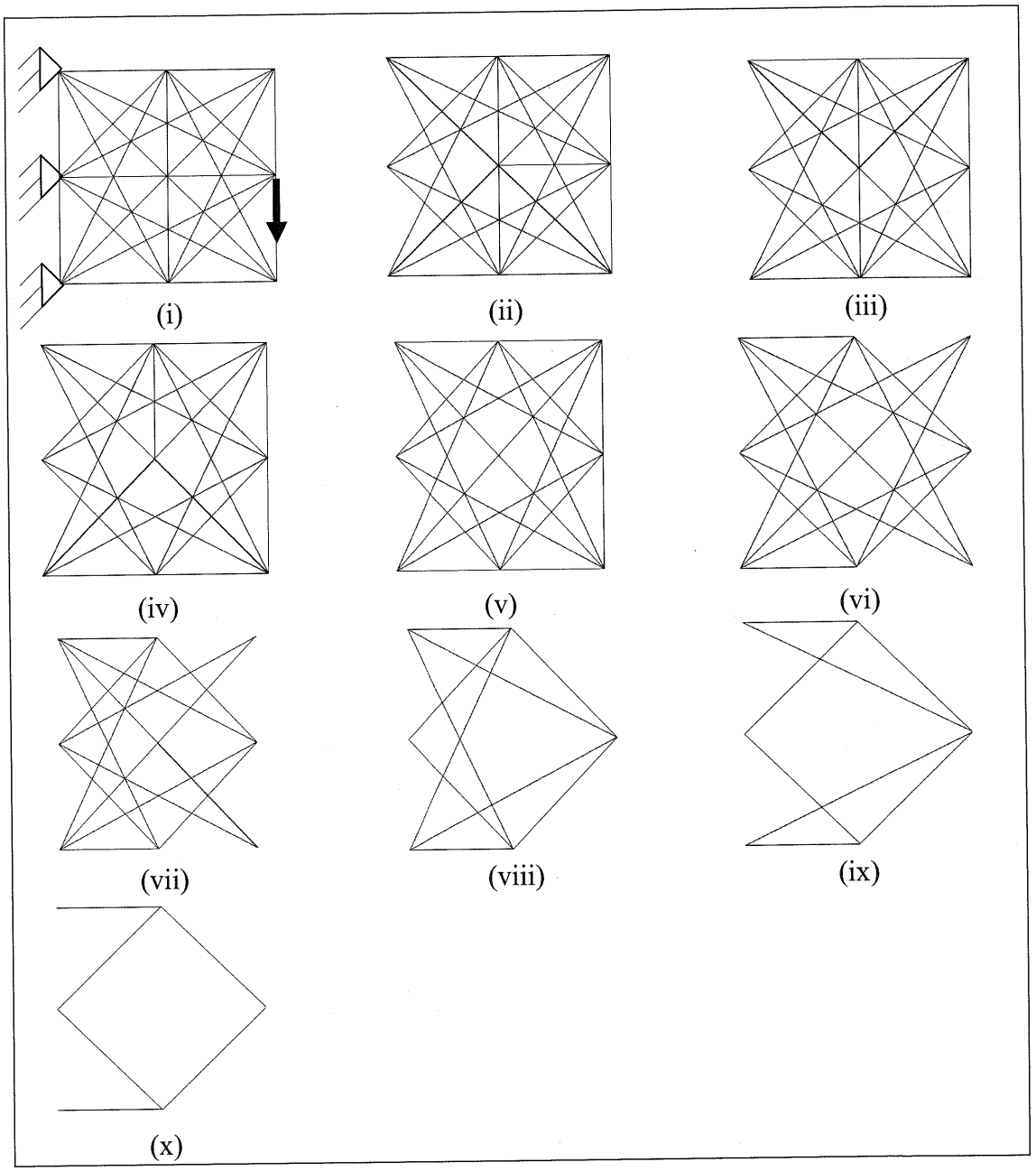


Figure 3.10: (i) Ground structure and first topology for the 9-node structure with full connectivity, (ii)-(ix) Intermediate results, (x) Optimal topology for the 9-node structure with full connectivity obtained by the ESO process.

It can be observed that the final design of the last two examples (c) and (d) is the same. These two examples have the same number of nodes but different numbers of members. Although the initial designs were different, both structures ended up to the same optimal design. There are some intermediate designs in Figure 3.10 (e.g. Figure 3.10(viii) and Figure 3.10(ix)) that may be efficient compared to the intermediate designs in Figure 3.9.

(e) Squared 9-node cantilever framework with full connectivity, 3 fixed supports and different loading

Figure 3.11(i) shows the initial ground structure of the 9-node cantilever framework with fully connectivity. The three nodes located at the left edge are considered fixed. An external load is applied at the node of the bottom right edge. By removing at each step a number of members that have the lowest stress values and using the previously mentioned strategy, an optimal design for the current structure is obtained. Note the lack of symmetry in the final design when a different loading condition from the previous example is applied (Figure 3.11(ix)).

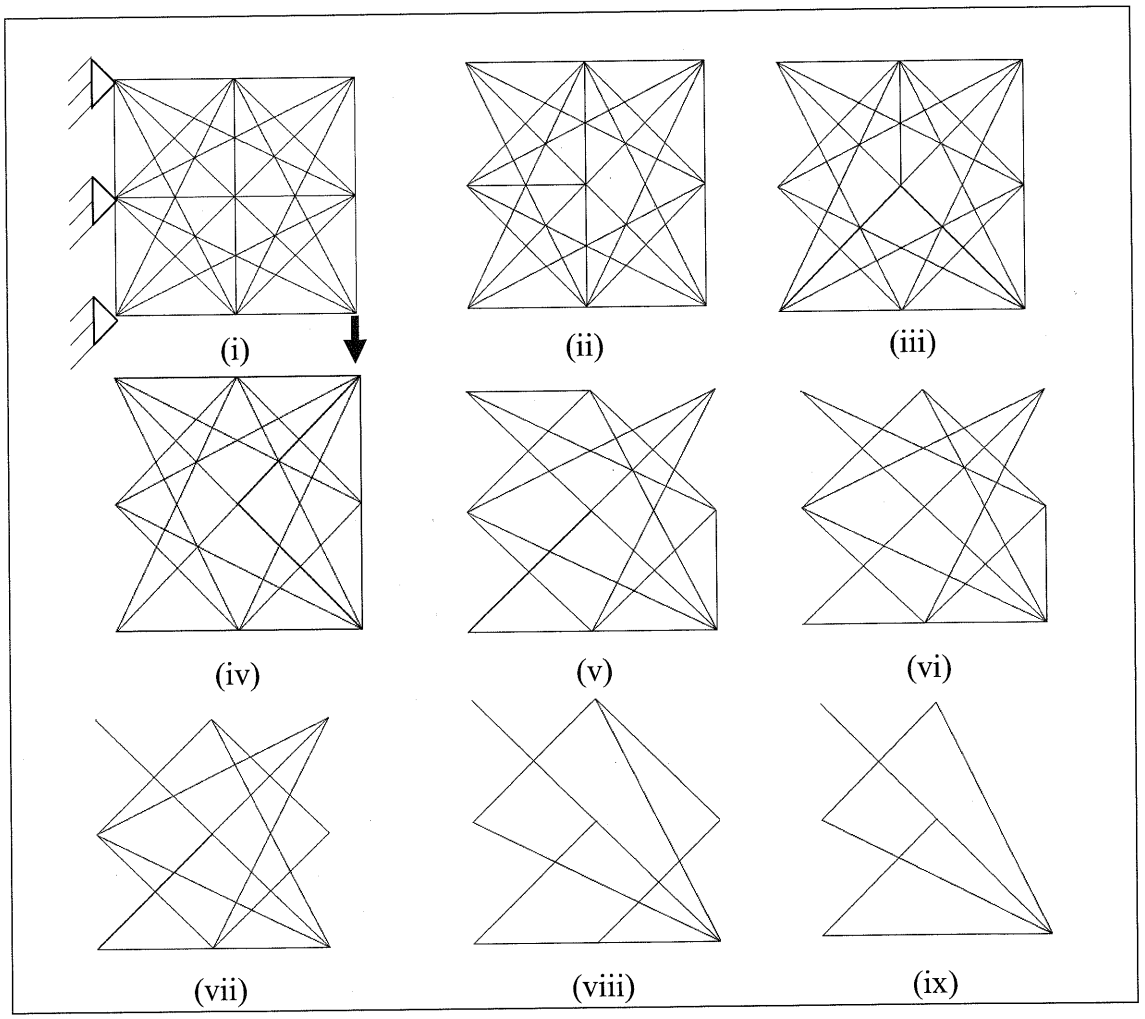


Figure 3.11: (i) Ground structure and first topology for the 9-node structure with full connectivity, (ii)-(viii) Intermediate results, (ix) Optimal topology for the 9-node structure with fully connectivity obtained by the ESO method.

The corresponding ABAQUS optimal designs for some of the previous examples are shown next (Figures 3.12 & 3.13). The ESO concept of the lowly

stressed member removal was adopted using ABAQUS. A fully connected structure is initially modelled in ABAQUS and the most lowly stressed member of the structure is detected. After eliminating this member, the new structure is modelled again and the same procedure is repeated manually. The results of ABAQUS software and MATLAB code are the same, proving that the implementation of the ESO based computer code is done correctly.

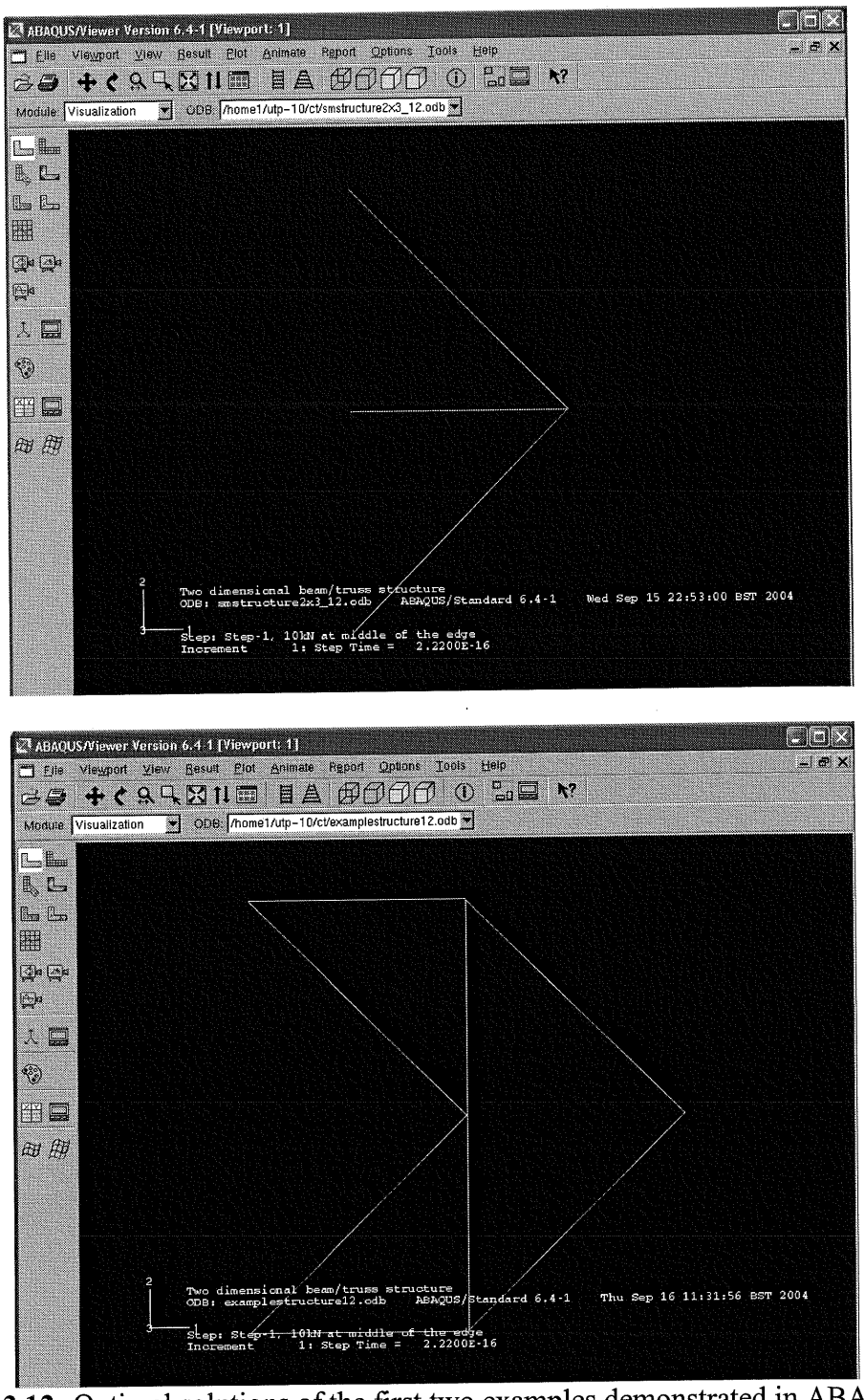


Figure 3.12: Optimal solutions of the first two examples demonstrated in ABAQUS.

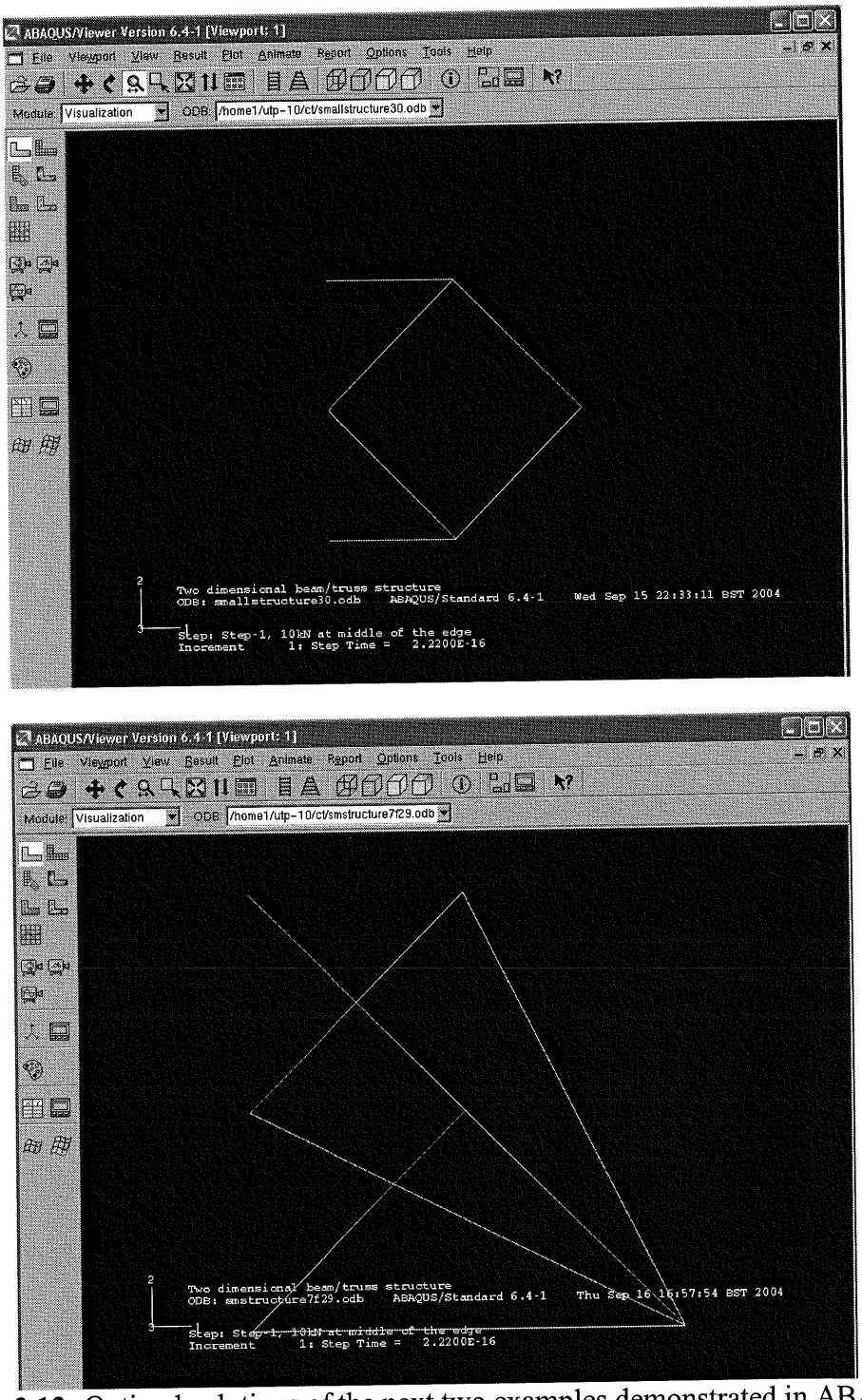


Figure 3.13: Optimal solutions of the next two examples demonstrated in ABAQUS.

3.4. General trends of the trajectory of designs during the ESO process on the maximum stress-weight plane

ESO reduces the weight of a design at every step of the iteration because it works by eliminating the least efficient members successively. However, this reduction in weight may be accompanied by an increase in the maximum stress within the structure. Often, in practice, the maximum allowable stress is specified and one looks for designs that reduce the overall weight. But, when comparisons of different designs or design concepts need to be made, a rational approach will be to account for the two measures of structural performance together and view the comparison problem as a multi-objective optimization problem with the weight and the maximum stress to be minimized simultaneously. This is convenient for design comparisons; however, for practical problem solving the conventional approach of treating the overall weight as an objective and the maximum allowable stress as a constraint may still be used. Furthermore, design objectives may additionally include maximizing the overall stiffness—hence the approach of design comparisons in a multi-objective optimization framework is presented here. From now on, we restrict ourselves to the two objectives mentioned earlier—the overall weight and the maximum stress over all the members.

The optimization of simple structural designs has been obtained so far using the MATLAB code developed and described previously. The trajectory of the current structure's weight against the current structure's maximum stress for the whole optimization procedure is then plotted for the purpose of design comparison during the ESO process. Evolution of designs on the weight-maximum stress plane with different starting points takes place. Different starting points are considered for the structures and the corresponding graphs of weight vs. maximum stress are compared. The trajectory of the current structure's weight against the corresponding maximum stress for the whole optimization procedure is then an indicator of the quality of designs being generated during the course of the ESO process.

Framework design examples in the shape of cantilever beams and MBB beams (bridge-like structures with the two ends supported) under fixed supports and load conditions have been considered (assuming no self weight of the structures). An assessment of the evolution of ESO generated designs as well as design comparisons are presented next via examples.

Example 1: MBB Framework

(a) 15-Node framework

(i) Fully connected initial design

An MBB framework with 15 nodes is shown in Figure 3.14. All the nodes are fully connected and the total number of members is 105. Boundary conditions are applied at the bottom corners of the structure as shown. An external load is applied at the middle of the top of the structure. Also, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set to 1.

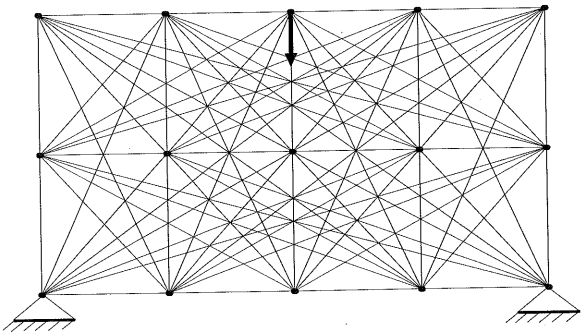


Figure 3.14: Fully connected MBB framework with 15 nodes.

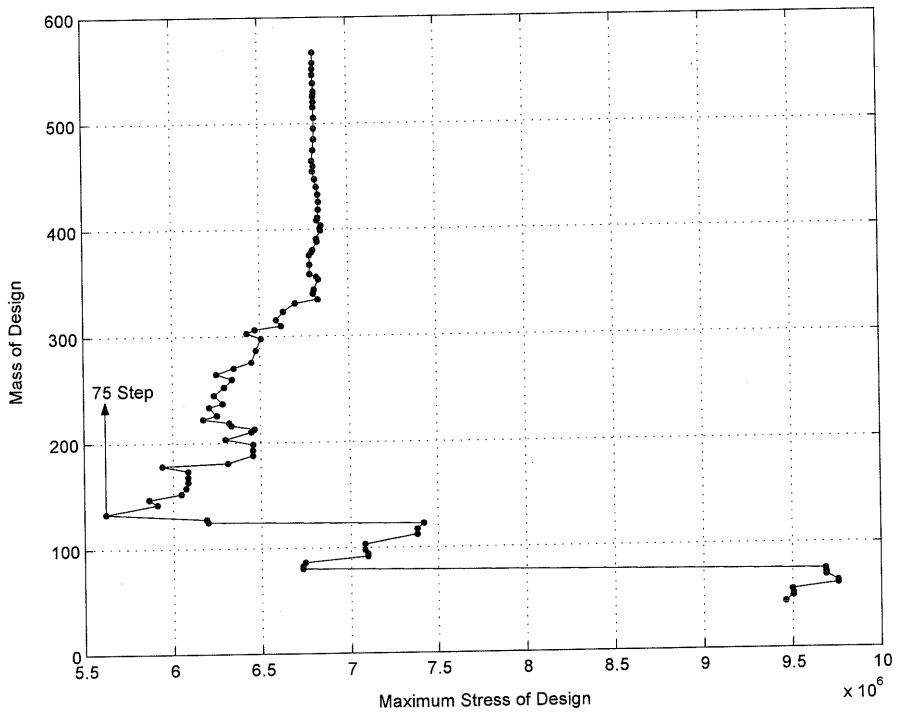


Figure 3.15: Discrete ESO trajectory for 15-node fully connected MBB framework.

The trajectory in the weight-maximum stress plane shows a complex behaviour (Figure 3.15). The units are arbitrary since a linear analysis has been carried out for an arbitrary load: stress values can be proportionately scaled for a different value of external loading. The highest weight point represents *the over connected initial design*. It is evident from the figure that the steps taken by the ESO strategy are not always Pareto efficient. Initially, an improvement takes place in both maximum stress as well as weight until the critical point at step 75 (the corresponding design is shown in Figure 3.16). Beyond this step, a reduction in weight is generally accompanied by an increase in maximum stress except for few steps where improvement on both counts of stress and weight is observed. This trend is quite revealing as to how ESO works. We have observed this general character in several examples. A detailed discussion of this behaviour is postponed until section 3.7.

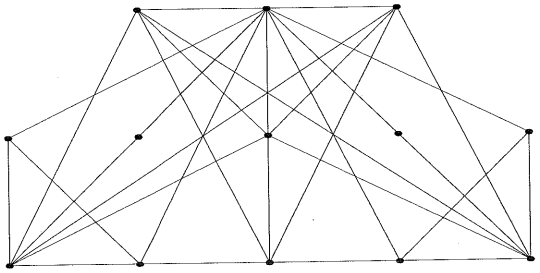


Figure 3.16: Drawing of the particular structure at step 75 critical point.

(ii) ESO starting point with adjacent connectivity only

We next study what difference it makes to the evolution of designs on the weight-maximum stress plane when different starting points are considered. Different initial designs are considered for a 15-node structure and the corresponding trajectories of weight vs. maximum stress are compared. An example of a MBB framework with 15 nodes but with a smaller number of members than in the previous example is shown in Figure 3.17. In this case, the total number of members is 38. The two bottom corners of the structure are fixed as shown. An external load is applied at the middle of the top of the structure. The number of members that have the lowest maximum stress and that are removed from the structure for each iteration is again set

to 1. The trajectory on the stress-weight plane is shown in Figure 3.18. The structure that corresponds to the critical point (labelled step 19) is shown in Figure 3.19.

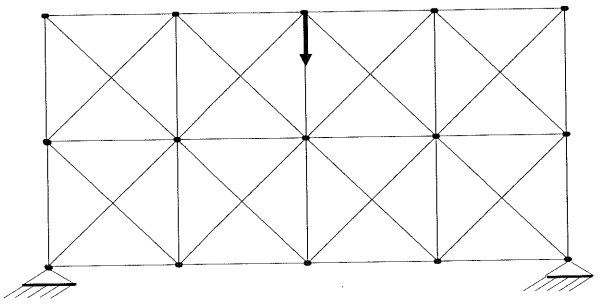


Figure 3.17: MBB framework with 15 nodes.

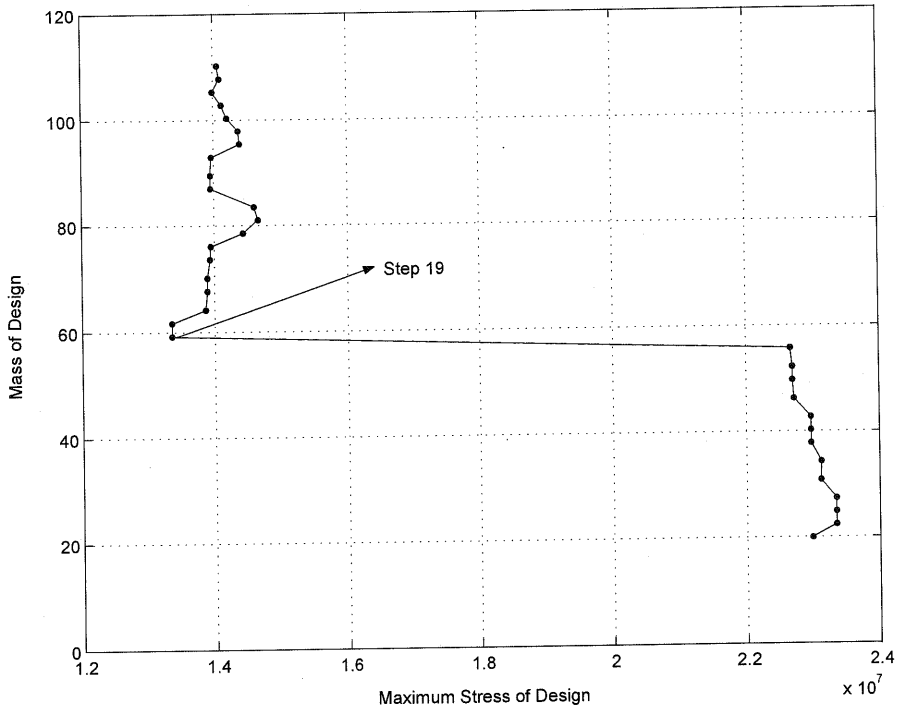


Figure 3.18: Discrete ESO trajectory for 15-node MBB framework.

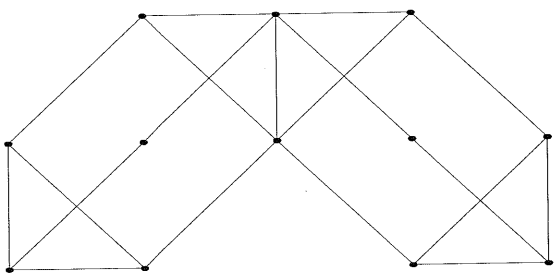


Figure 3.19: Drawing of the particular structure at step 19 critical point.

The ESO trajectories of the fully connected 15-node MBB structure (Figure 3.15) and the corresponding 38-member structure (Figure 3.18) are plotted on the same graph in Figure 3.20. It can be seen that the 105-member structure is initially very poor weight-wise but very good stress-wise. As ESO runs, the weight reduces and the quality overtakes those produced by the 38-member initial design based ESO run on both counts of stress and weight. The explanation is that the initial fully connected structure offers a much larger choice of designs to work from and the ESO process has an opportunity to exploit this extra choice. Clearly, ESO is very sensitive as far as the selection of an initial structure is concerned. Choosing an initial structure that is not fully connected may tend to cause ESO to give sub-optimal results.

The initial structure with 38 members is completely excluded out of the ESO trajectory based on the fully connected initial design. This indicates that this particular 38-member structure is not a good starting design, and that optimization based on this initial design is likely to give designs that are not satisfactory. The design options afforded by the 38-member structure can be viewed as a subset of the design options afforded by the design with 105 members as the starting point. Note that the member cross-sections for all the members in the two starting designs have the same geometrical properties.

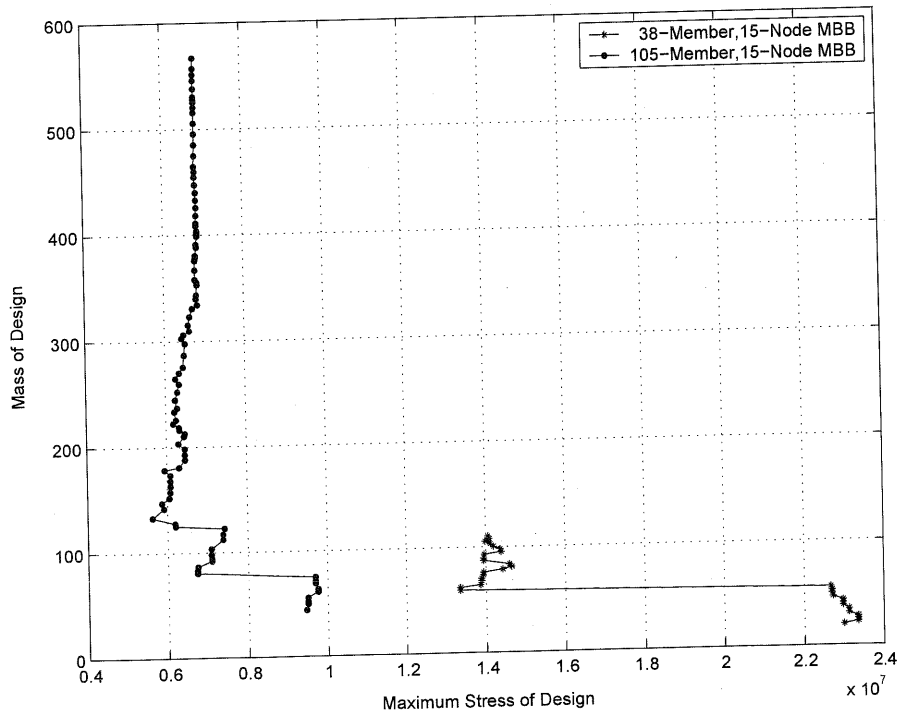


Figure 3.20: Comparison of the ESO trajectories of the fully connected 15-node MBB and the 38-member, 15-node MBB.

(b) 25-Node MBB Framework

An example of a MBB framework with 25 nodes is shown in Figure 3.21. All the nodes are fully connected and the total number of members is 300. Boundary conditions are applied at the bottom corners of the structure while an external load is applied at the middle of the top of the structure. The number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set to 1 again.

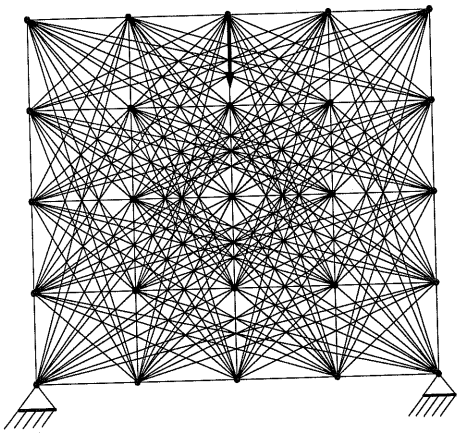


Figure 3.21: Fully connected MBB framework with 25 nodes and 2 BC nodes.

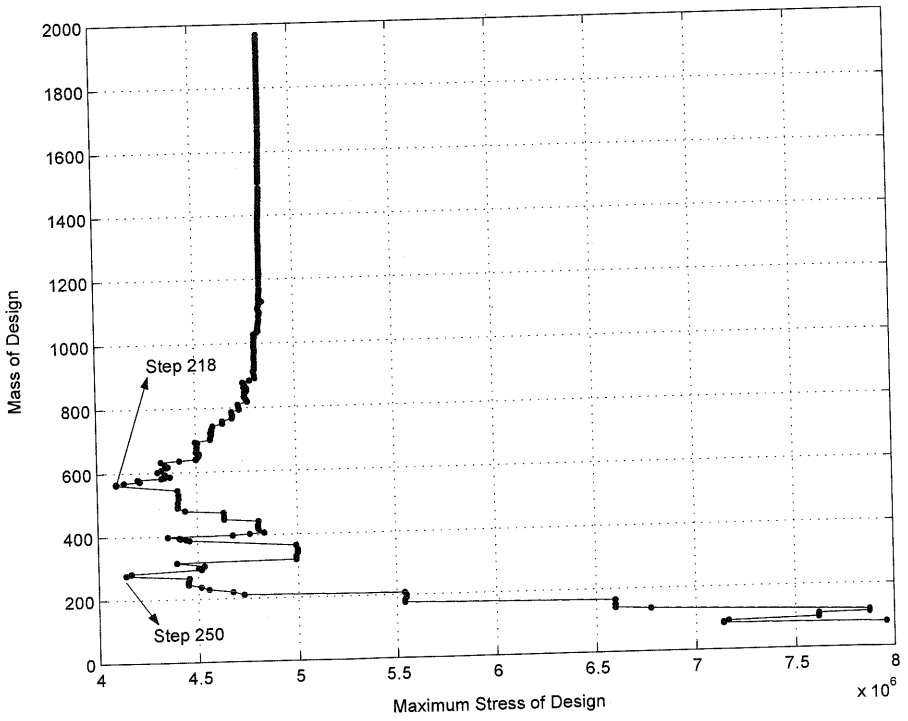


Figure 3.22: Discrete ESO trajectory for 25-node fully connected framework.

The trajectory on the weight–maximum stress plane (see Figure 3.22) is more complex now. There is again an initial phase of improvement on both counts of weight and maximum stress. Beyond step 218 (Figure 3.22), this trend changes and a reduction in weight is nearly always accompanied by an increase in the maximum stress. After a few steps, improvement on both counts of stress and weight is observed occasionally.

The trajectories presented so far have some common features. It seems that the moves made by the ESO strategy are Pareto-efficient in some parts (particularly initially) whereas, at a later stage, reduction in weight has a stress penalty. This general observation may have an interesting implication. The point on the weight-stress plane where the designs stop producing Pareto-efficient designs with each ESO step may suggest a point to stop the ESO process. So far, this is often carried out in an ad hoc manner. The point at step 218 dominates all the points above it, as it has the lowest maximum stress value and the lowest mass value at the same time. The structure that comes out at this point (Figure 3.23) is the optimum structure among the structures created from the initial step 1 to the step 218. From the step 218 and on, the ESO trajectory follows a similar Pareto shaped curve to that observed in the previous 25-node framework example.

Beyond step 218 the ESO trajectory is not monotonic. First, the weight of the structure is decreased and the maximum stress increases and no decision can be made about the optimality of the structure. However, some good designs can be chosen along this Pareto path according to the desirable values for the maximum stress and weight of the structure. Indeed, a very good design is found at step 250, as can be seen from Figure 3.22. The value of the maximum stress for this design is very close to the corresponding value of the optimum structure at step 218. Furthermore, the value of its weight is less (almost half) than the corresponding weight of the previous optimum one. The design of the structure that belongs at step 250 is shown in Figure 3.24. This shows the usefulness of the trajectories on the stress-weight plane in design selection in conjunction with the ESO strategy.

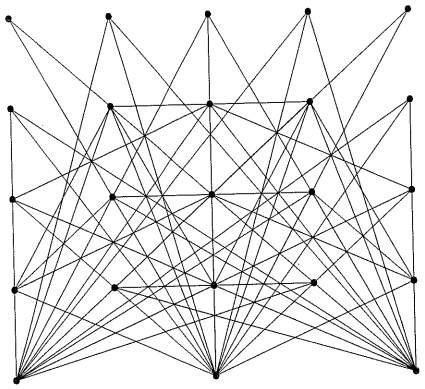


Figure 3.23: Drawing of the particular structure at step 218 of the ESO optimization (critical point).

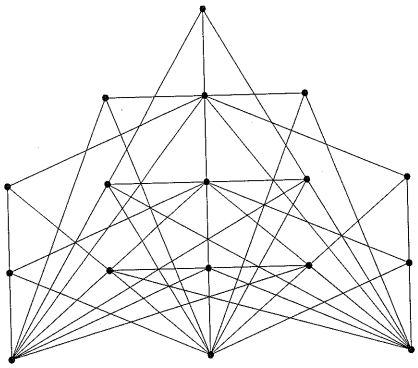


Figure 3.24: Drawing of the particular structure at step 250.

Example 2: 25-Node Cantilever Framework

An example of a framework with 25 nodes is illustrated in Figure 3.25. All the nodes are fully connected and the total number of members is 300. Boundary conditions are applied at all nodes, which are located in the left hand side of the structure on the vertical line. An external load is applied at the middle of the right free edge of the structure. Also, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is again set to 1.

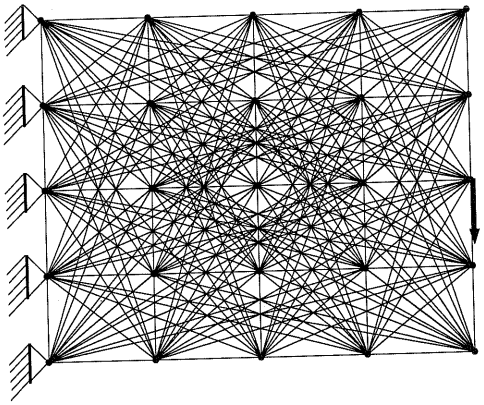


Figure 3.25: Fully connected framework with 25 nodes.

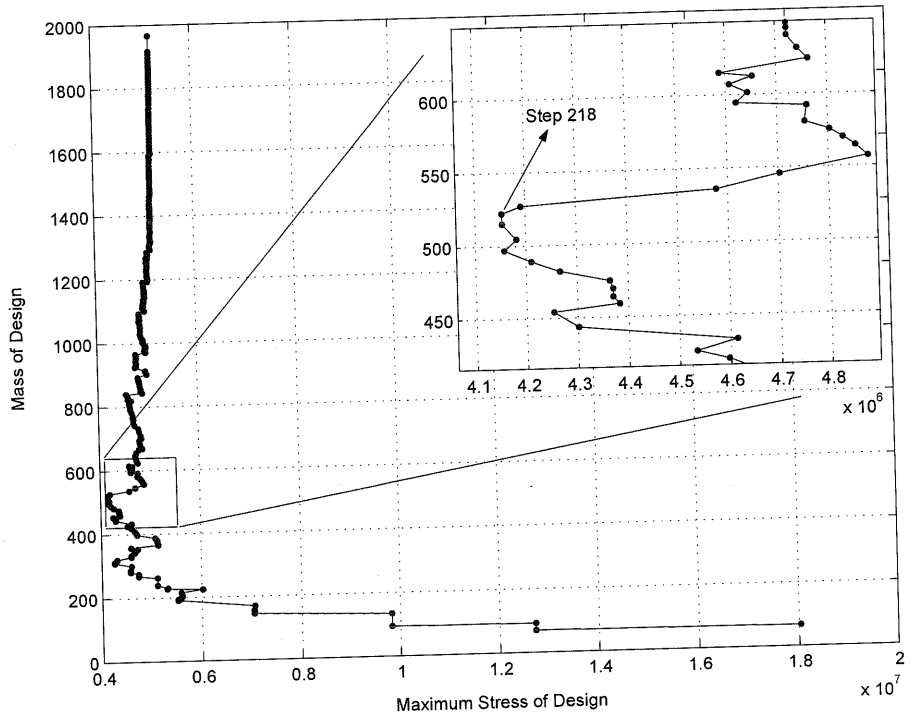


Figure 3.26: Discrete ESO trajectory for 25-node fully connected framework.

The trajectory obtained during the ESO process is presented in Figure 3.26. Notice an initial improvement in both maximum stress as well as weight. Beyond a certain point, this trend changes and a reduction in weight is nearly always accompanied by an increase in the maximum stress. It can be noticed that the structure at step 218 dominates all the structures above it, as it has the lowest value for maximum stress. From that point, a 'Pareto Front shaped curve' is created. This may not be the actual Pareto Front for the design options available; however in general

trend it suggests a decrease in weight at the expense of an increase in the maximum stress. The structure of step 218 is considered as the optimum design among the designs between steps 1 and 218. From that point and on, an optimum design can be decided according to the preferred objectives.

Example 3: Large Scale Problems

(a) 55-Node Michell type Framework

An example of a Michell type structure with 55 nodes is illustrated in Figure 3.27. All the nodes are fully connected and the total number of members is 1485. Boundary conditions are applied at nodes, which are located at the bottom corners of the structure. An external load is applied in the middle of the bottom of the structure. Also, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set this time to 10. In Figures 3.28 & 3.29, we present the ESO trajectory for the particular structure and the design found at the critical point correspondingly. It is worth mentioning that the asymmetry appearing in the structure belonging to the critical point (Figure 3.29) is due to the fact that the algorithm tends to remove a fixed percentage of members each time and not only members of equal stress.

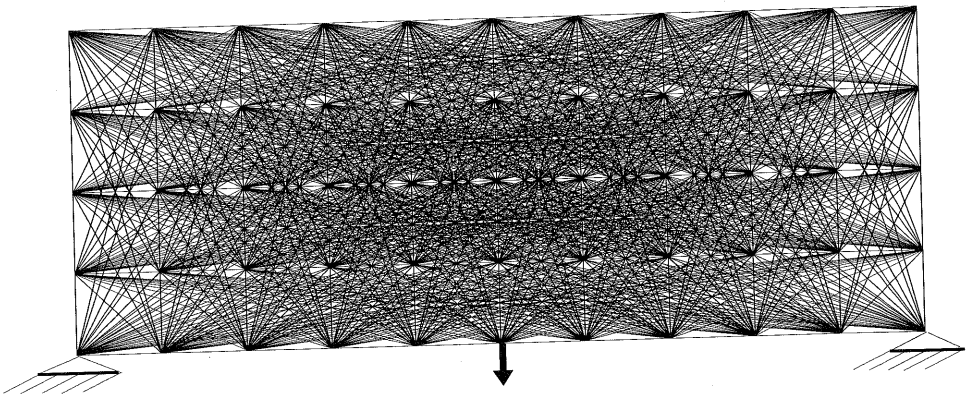


Figure 3.27: Michell type framework with 55 nodes.

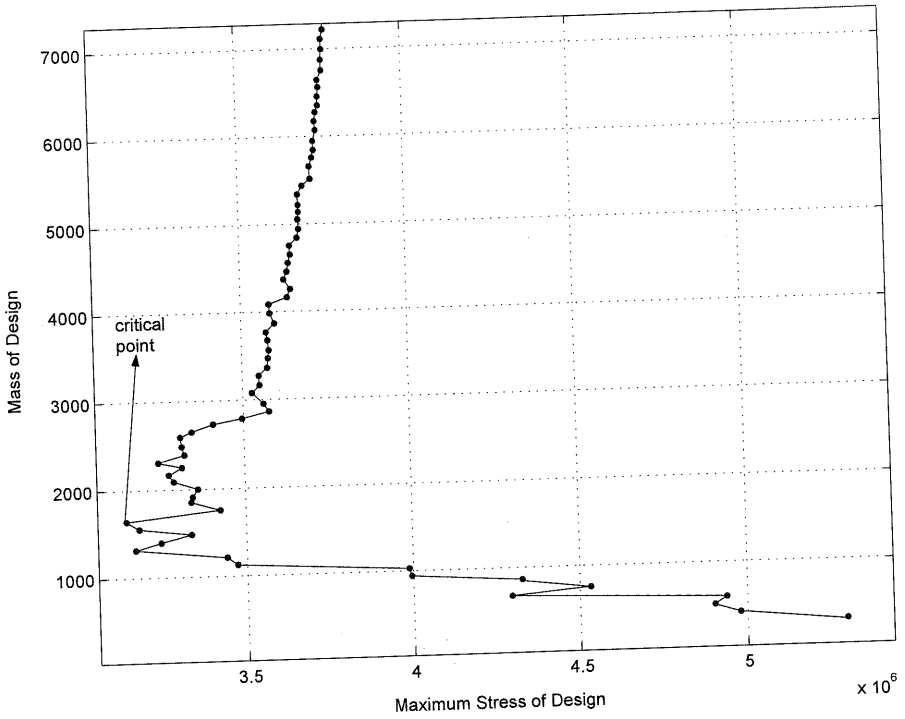


Figure 3.28: Discrete ESO trajectory for 55-node Michell type framework.

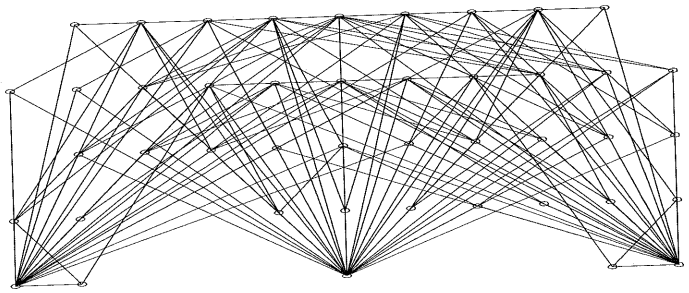


Figure 3.29: Design of the Michell type structure that corresponds to the critical point of the previous graph.

(b) 105-Node MBB Framework

An example of a MBB framework with 105 nodes is also considered (Figure 3.30). Each node is linked to all the others and the number of members is 5460. Boundary conditions are applied at nodes, which are located at the bottom corners of the structure. An external load is applied at the middle of the top of the structure. Also, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set to 1. Step 489 is found to be the critical point in the following graph (Figure 3.31).

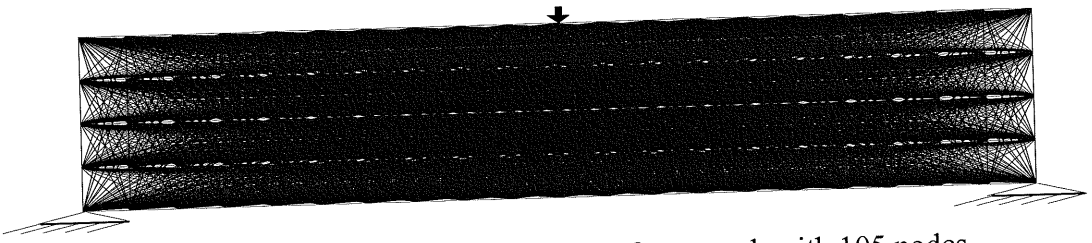


Figure 3.30: Fully connected MBB framework with 105 nodes.

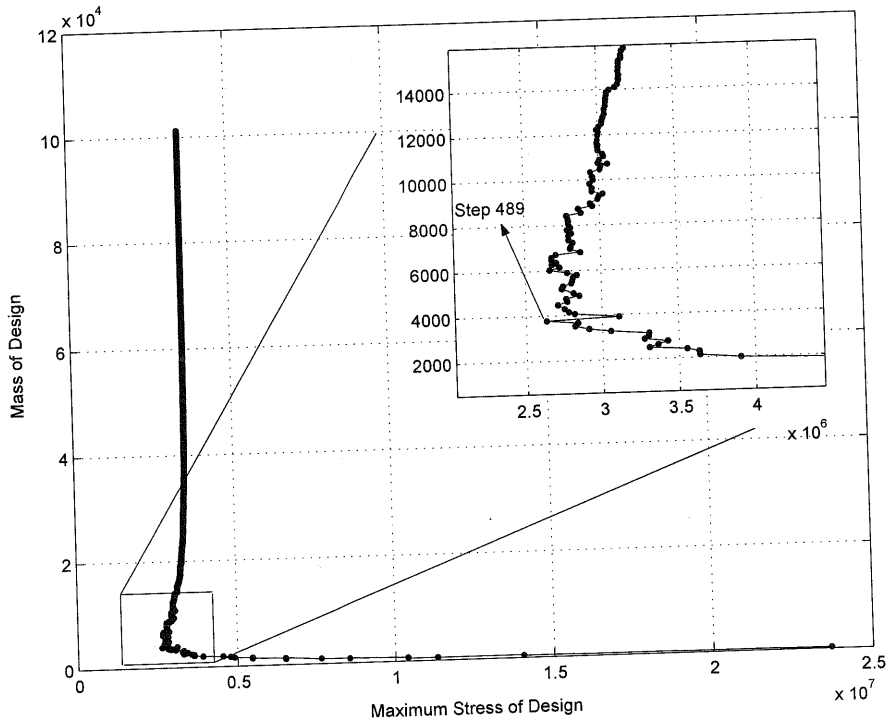


Figure 3.31: Discrete ESO trajectory for 105-node MBB framework.

The trajectory on the weight-maximum stress plane for the 105-node MBB framework can be seen in Figure 3.31. There is again an initial phase of improvement on both counts of weight and maximum stress. Beyond step 489 (Figure 3.31), this trend changes and a reduction in weight is nearly always accompanied by an increase in the maximum stress. After the critical point at step 489, improvement on both counts of stress and weight is observed occasionally.

(c) 121-Node Cantilever Framework

An example of a framework with 121 nodes is next studied (Figure 3.32). Each node is connected to all the others and the number of members is 7260.

Boundary conditions are applied at nodes, which are located in the left hand side of the structure. An external load is applied at the middle of the right free edge of the structure. Also, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set to 1. The ESO trajectory for this case is presented in Figure 3.33.

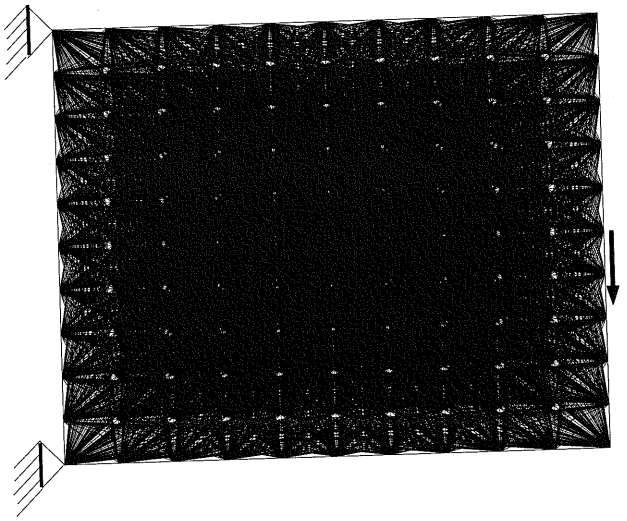


Figure 3.32: Fully connected framework with 121 nodes.

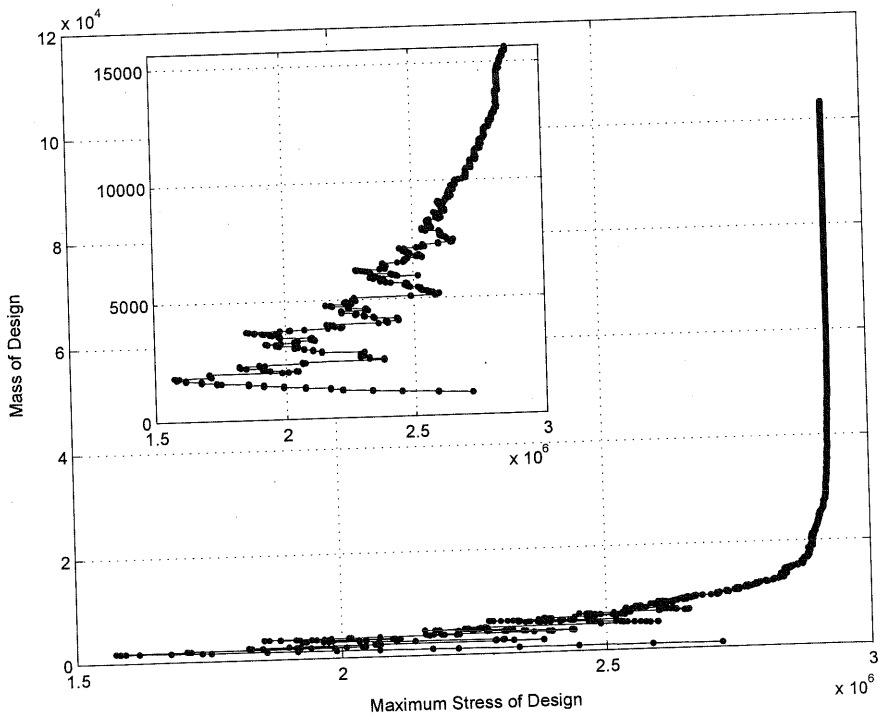


Figure 3.33: Discrete ESO trajectory for 121-node framework.

The trajectory on the weight-maximum stress plane as can be seen clearly in Figure 3.33 becomes quite complex. There is an initial phase of improvement on both counts of weight and maximum stress until the mass of the design reaches roughly the value of 10000. Between mass values of approximately 2000 and 10000, improvement on both counts of stress and weight is observed continuously. Beyond the critical point, this trend changes and a reduction in weight is nearly always accompanied by an increase in the maximum stress. None of the designs during the optimization have higher stress than the initial because of the chosen early termination condition for this problem.

3.5. The effect of member removal ratio in large scale problems

The effect of the 'step size' on design evolution will be studied next for two examples. Changing the number of steps during the ESO process, we will be able to notice the change in the Pareto efficiency (a state in which there is no more Pareto improvement).

Example: 55-Node Michell type Framework

For the same Michell type structure illustrated in Figure 3.27, the number of members that have the lowest maximum stress and that are removed from the structure for each iteration is set now to 1. In Figure 3.35, we compare Figure 3.34 (step size 1) with Figure 3.28 (step size 10). We can conclude that a larger number of calculation steps (small step size) in the optimization increases the possibility of finding the optimum design. In other words, keeping the number of members removal during ESO as small as possible, improves the optimization procedure. Obviously, having a large number of minimum stressed members being removed from structure, some important members may be removed and, as a result of that, the design tends to commit to certain features very early. Figures 3.35 & 3.36 indicate that when we remove only one member during the process, it actually finds better designs after the supposed optimum design that we get when the number of removed members is increased to 10 and 40 respectively. In particular, the maximum stress value and

weight of the critical point found during ESO for step size 1 (Figure 3.34) is $2.3968 \cdot 10^6$ and 859 respectively. Instead, the corresponding values that we get for the critical point during ESO for step size 10 (Figure 3.28) are $3.1412 \cdot 10^6$ and 1645. Comparing the maximum stress and weight values for these two critical points, it is obvious that the critical point found for step size 1 dominates the critical point for step size 10. This means that the design of the critical point obtained for step size 1 is better than the one for step size 10. This does however require ten times as many FEA solutions.

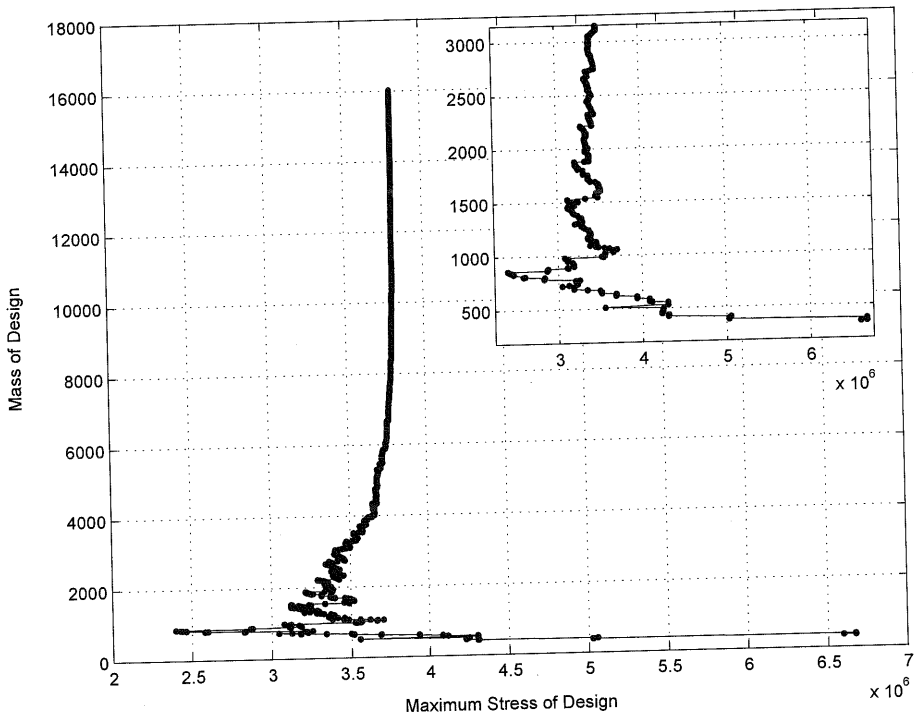


Figure 3.34: Discrete ESO trajectory (step size 1) for 55-node Michell type framework.

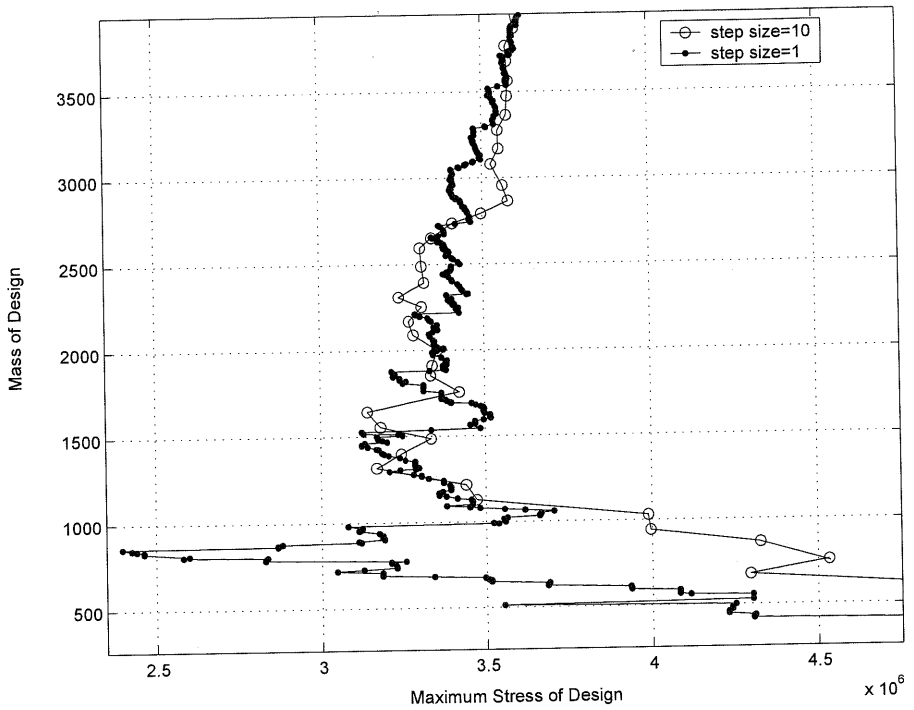


Figure 3.35: Comparison between the ESO trajectories with step size 1 and step size 10 for 55-node Michell type framework.

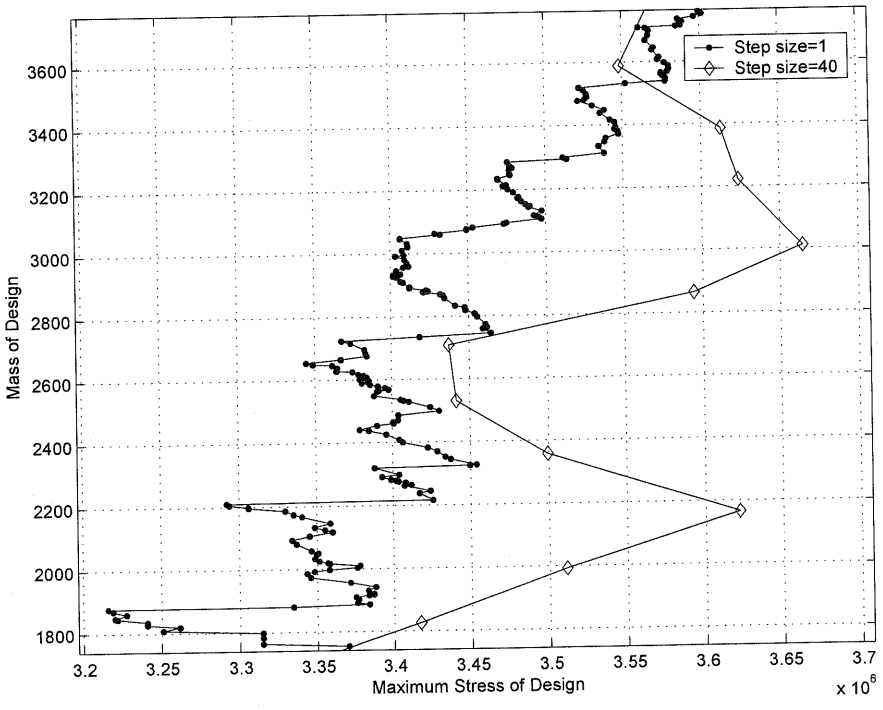


Figure 3.36: Comparison between the ESO trajectories with step size 1 and step size 40 for 55-node Michell type framework.

3.6. The effect of scaling the cross-sectional size in framework topology optimization

The observation that the designs produced by the application of ESO to a highly connected initial design are superior to the designs produced by a lesser connected starting point that offers only a subset of the original design options does not necessarily lead to the conclusion that a highly connected initial design will always produce the most efficient designs. The obvious extra design variable is the cross-section of the members. Suppose we keep the cross-sectional property in all the members fixed within a design but make this a variable between two different ESO runs, then simpler initial designs may lead to better overall designs after ESO runs.

To illustrate this, we fix the initial cross-sections in the 38-member initial design (Figure 3.17) and the 105-member initial design (Figure 3.14) in such a way that the total mass of the initial design for both two cases is the same (hence treating the thickness as a variable across two different ESO runs, but constant over all the members for a given run).

The ESO trajectory of the same MBB structure that has 38 members but the same weight as the fully connected one appears in Figure 3.37. Increasing the thickness of the members and hence the member area, the weight of the structure with fewer members will be equal to the weight of the fully connected structure. The plots of the maximum absolute member stress of the structure against the weight of the structure for the fully connected MBB framework (Figure 3.14) and the MBB structure with 38 members but same total weight (Figure 3.17) are presented in the same graph (Figure 3.37), so that comparisons can be made.

The plot at the left of the same figure is the ESO trajectory of the same MBB structure that has 38 members. The plot at the right of Figure 3.37 is the ESO trajectory of the fully connected MBB structure consisted of 15 nodes and in total 105 members. It can be observed that the plot line of the structure with fewer members seems to have lower maximum member stress during the whole optimization procedure than one obtained from the fully connected structure as the initial design. Notice that the 38-member structure gives better design solutions than those obtained from the fully connected starting design.

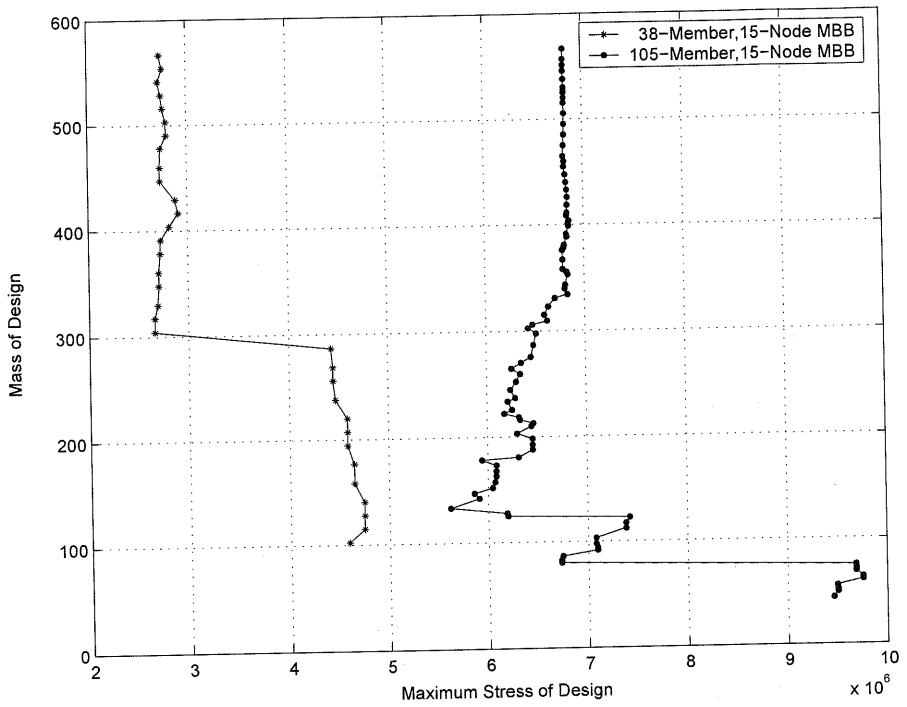


Figure 3.37: Discrete ESO trajectory comparison between a fully connected MBB structure and a similar MBB structure with fewer members.

The initial design of the next example is the structure that has been evolved in 26th step of the fully connected 15-node MBB structure (Figure 3.38). It has 79 members but the *same weight* as the initial fully connected structure of Figure 3.14. Boundary conditions are applied at nodes, which are located at the bottom corners of the structure. An external load is applied at the middle of the top of the structure. The number of members that appear to have the lower maximum stress and that are removed from the structure in each iteration is set to 1 as before.

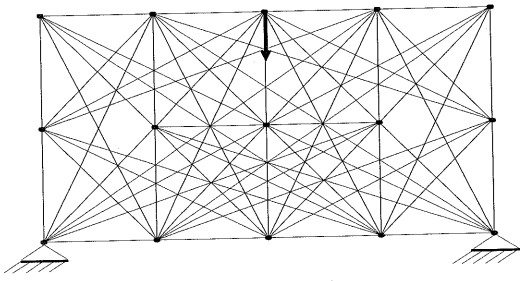


Figure 3.38: 79-member MBB framework with 15 nodes.

The ESO trajectory for the 79-member, 15-node MBB framework is presented in Figure 3.39. It can be seen that the trajectory of designs with fewer initial members seems to have lower maximum member stress during the whole optimization

procedure than the fully connected structure (Figure 3.40). This means that if the cross-section is allowed as a variable of the problem, then a simpler starting point with thicker cross-section may be a better starting point. This also suggests that simultaneous alterations in the cross-section and topology may be the most desirable ESO strategy.

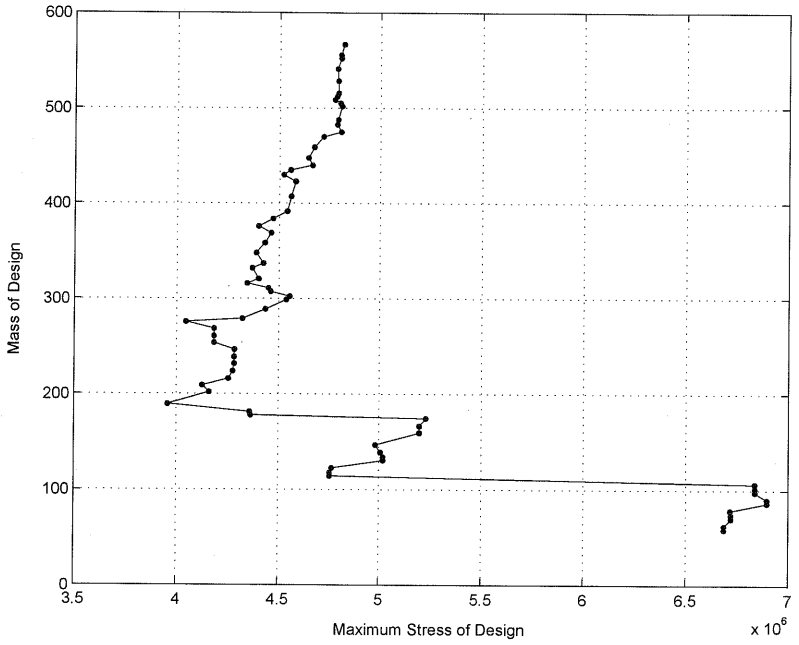


Figure 3.39: Discrete ESO trajectory for 79-member, 15-node MBB framework.

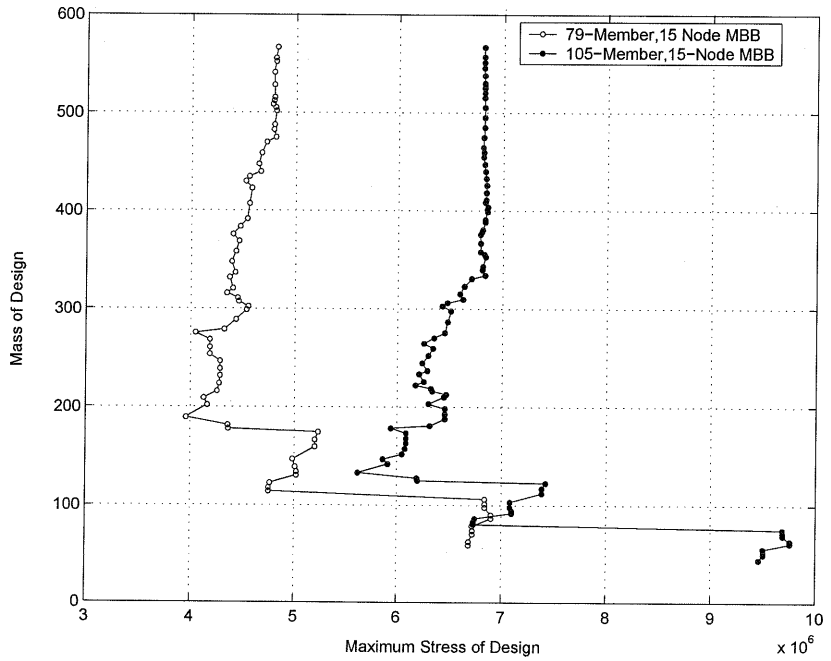


Figure 3.40: Discrete ESO trajectory comparison between a fully connected MBB framework and a similar MBB structure with 79 members.

Since here we are carrying out linear static analysis, it will be clear that any design can be moved along a hyperbola in the mass/stress plane simply by factoring up or down the thickness of the frame members in this way. This has no effect when comparing two alternative topologies from a Pareto, since if a design dominates another for a given thickness of structure, it will do so however the structures are scaled. It does mean, however, that designs in the Pareto set can be subtly moved along the Pareto Front by suitable scaling.

Size and topology optimization is simultaneously obtained by scaling the thickness (cross-sectional area) of every design that comes up during the ESO procedure. While member removal occurs during ESO, changing the topology of the structure, a hyperbola containing all possible designs with the same topology and maximum stress/mass ratio but with different thickness is provided for each ESO design. This is because stress σ is inversely proportional to the cross-sectional area A: $\sigma \approx Force \times A^{-1}$ and weight W is proportional to the cross-sectional area A: $W \approx \rho \times l \times A$ (ρ = density, l = length of element), so that their product $\sigma \times W = const$; a rectangular hyperbola on the σ-W plane (see Figure 3.41).

Bending stress of a truss-frame member has a very small effect compared to the axial stress hence it is neglected through the stress analysis. When an ESO design is obtained, the member with the maximum stress value is detected. After finding the force applied to the particular member, the equation that relates the maximum stress and the mass of the design is obtained and plotted. We can observe from Figure 3.41 that if an ESO design dominates another design for fixed thickness, it will keep dominating it even when the thickness of both designs is scaled. For fixed cross-sectional design search, the designs at the tail end of the ESO trajectory are not necessarily superior to the previously obtained designs. However, Figure 3.41 shows that given the freedom to alter cross-sectional area, the design at the end of the ESO run is superior to the previous designs. We will not pursue this further here because the main objective of this work is not to explore shape/size optimization.

In Figure 3.42, the plots of the analytical and non-analytical approach of thickness scaling on one of the ESO designs of the fully connected 15-node MBB structure are compared. The very small difference that can be noticed between these two approaches, confirms that bending stress is negligible in stressed frameworks. The non-analytical approach was obtained by simply finding the topology of the ESO

design and then making various runs with different thicknesses including both axial and bending stress in order to find the corresponding maximum stress and mass design values. While in the analytical approach, the hyperbola represented by the thickness scaling occurs on the ESO design with no bending stress consideration.

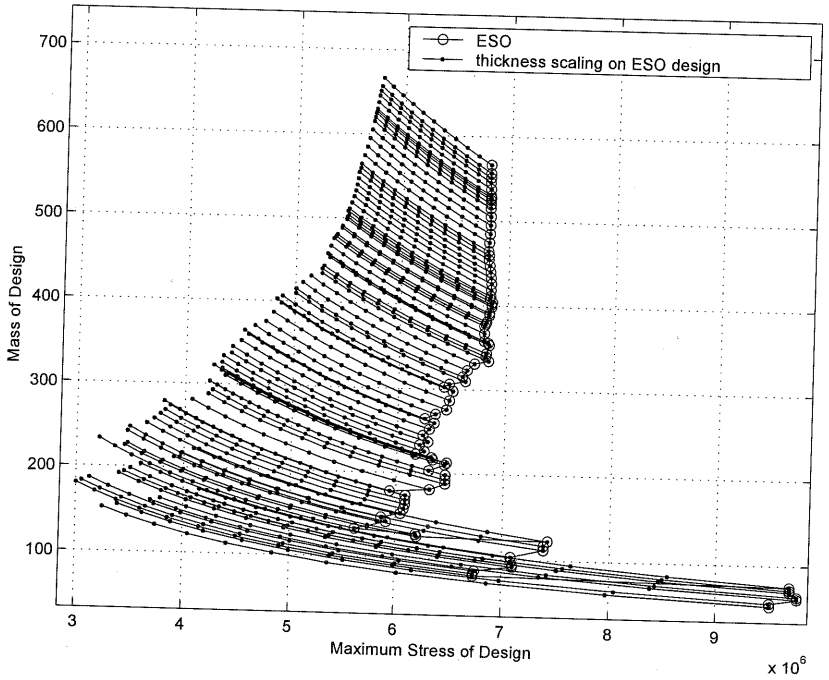


Figure 3.41: Size and topology optimization of the fully connected 15-node MBB structure (see Figure 3.14).

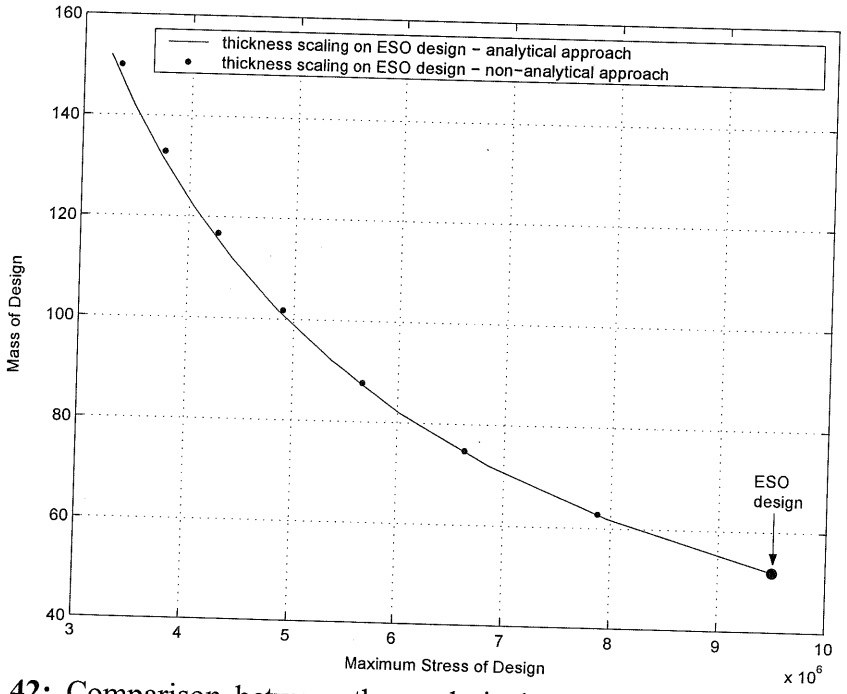


Figure 3.42: Comparison between the analytical and non-analytical approaches of thickness scaling on one of the ESO designs of the fully connected 15-node MBB structure.

3.7. General shape of ESO trajectory for frameworks

The trajectory of weight against maximum stress of the structure during ESO has been plotted for various example cases all of which have the following generic features. The initial portion shows a reduction in weight as well as the overall maximum stress. Clearly these ESO steps are Pareto-efficient in that they improve designs on both counts. The trajectory then reaches a turning point beyond which weight is reduced only at the expense of increased maximum stress. The very shallow slope of the trajectory in this portion indicates that the gain in weight reduction is only marginal whereas the increase in the maximum stress of the structure is very substantial. This suggests the possible use of the turning point as an indicator for stopping the ESO iterations in a natural way without recourse to introducing an arbitrary stopping criterion.

Figure 3.43 shows a schematic sketch of the general shape of the trajectories observed in our numerical experiments. The initial steps seem to produce designs that are lighter as well as less stressed. Beyond a critical point we cannot claim that ESO improves designs because for (small) gains in weight, there is a (usually large) penalty on the maximum stress.

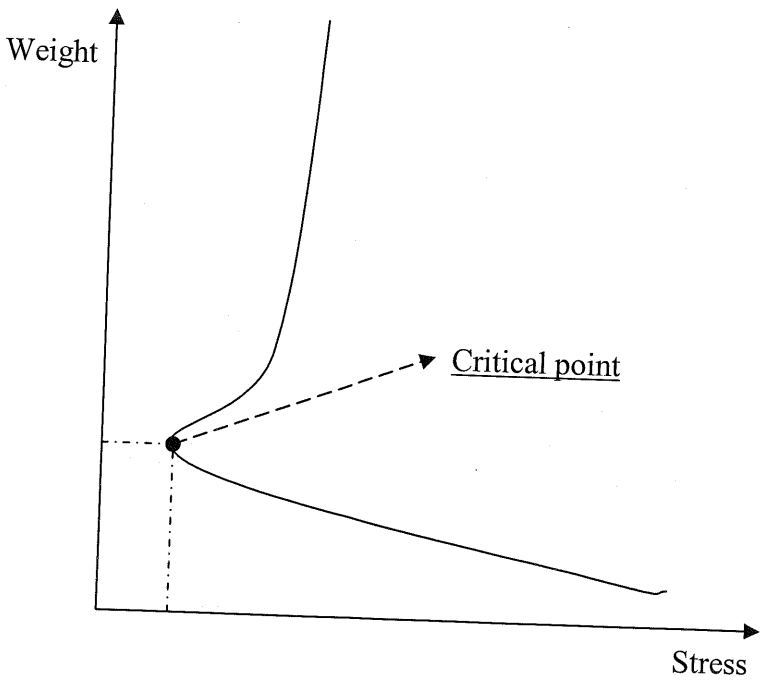


Figure 3.43: General ESO trajectory for frameworks.

The critical point is the last design in the trajectory that dominates all previous designs. Beyond this, the trajectory sometimes shows a complex behaviour with "multiple turning points". The general trend, however, is a reduction in weight with an increase in maximum stress during the latter stages of the ESO process.

Since beyond the critical point, the weight of the structure decreases but the maximum stress simultaneously increases, no decision can be made about the optimality of the structure. However, some good designs can be chosen along this portion of the trajectory according to a trade off between values for the maximum stress and weight of the structure—should other design specifications provide such information.

3.8. Conclusions

We have found by now that the Evolutionary Structural Optimization strategy reduces the weight of a design at every step of the iteration, after a critical point it increases the maximum stress within the structure. Evolution of designs on the weight-stress plane with different starting points have been considered. Different starting points were considered for some of the structures and the corresponding graphs of weight vs. maximum stress were compared. According to the plots obtained, it seems that all the designs with different starting points move towards the same Pareto Front. However, all the structural design examples that have been considered verify the general shape of the ESO trajectory. Observations on the ESO trajectory can provide us with a significant outcome on selecting the stopping criterion for the ESO iterations and the best optimized design according to the preferred constraints. Besides that, scaling of the cross-sectional size of the structure was implemented at each iteration of the ESO process. Simultaneous alterations in the cross-section and topology of a structure can be proposed as the most desirable ESO strategy.

Chapter 4

Pareto-comparison between the Evolutionary Structural Optimization, Exhaustive Search and Genetic Algorithm applied to frameworks

4.1. Introduction

In this chapter, the effectiveness of the Evolutionary Structural Optimization (ESO) method of designing frameworks is studied. The designs obtained by ESO are just compared with those obtained using exhaustive search for combined performance on two counts: the maximum stress within the structure and the overall weight. Numerical experiments suggest that ESO produces reasonable designs at little expense; however, the procedure misses out on many efficient designs if one could afford the computational expense of an exhaustive search. To study the Pareto-efficiency of the ESO method for complex problems, an exhaustive search is not practical. Therefore, for computationally demanding problems Pareto Fronts (PF) are constructed by the use of a Genetic Algorithm (GA) and comparisons are made with multiple runs of GA generated Pareto Fronts. The approach adopted is to encode the problem formulation using a genetic algorithm and to allow the formulation to evolve in the direction of improving Pareto optimal designs [83, 94, 95]. Again, a general observation is that ESO produces Pareto sub-optimal designs, but is superior to GA if one could not afford a computationally demanding search. Finally, the cross-sectional area of each member is considered as a design variable and the coordinates of the nodes and connectivity among various members are considered to be fixed. The

trajectory of designs on the weight-maximum stress plane obtained by the ESO method with member thickness as a variable is found to be similar to that of the ESO trajectory when structurally inefficient members were eliminated from the design during evolution. The results of these numerical experiments demonstrate that while ESO is computationally efficient it fails to produce some designs with very good structural performance.

4.2. Pareto Optimal Front

To begin with the terms Pareto dominance and Pareto optimality are further clarified. Mathematically, the concept of Pareto dominance can be defined as follows: *A vector* $\boldsymbol{u} = (u_1,...,u_p)$ *is said to dominate* $\boldsymbol{v} = (v_1,...,v_p)$ *if, and only if,* $\boldsymbol{u}$ *is partially less than* $\boldsymbol{v}$, *i.e.,* $\forall i \in \{1,...,p\}, u_i \leq v_i \wedge \exists j \in \{1,...,p\}: u_j < v_j$. In the context of optimization, p is the number of objectives.

Similarly, the concept of Pareto optimality can be defined as follows: *A solution* $x_u \in U$ *is said to be Pareto-optimal if, and only if, there is no* $x_v \in V$ *for which* $v = f(x_v) = (v_1,...,v_p)$ *dominates* $u = f(x_u) = (u_1,...,u_p)$. This set of solutions that are non-dominated regarding the entire search space is the so-called Pareto-optimal set.

The Pareto Front (PF) can be defined as the function space representation of all the solutions in the Pareto-optimal set. Generally, when there are two objectives as in our case, the Pareto Front is a curve. The Pareto-optimal designs are the best designs that can be produced for a given problem formulation for a given set of criteria [96]. If the goal is to improve the performance in those criteria then it is possible to manipulate the problem formulation to achieve an improvement.

The ESO strategy reduces the weight of a design at each step of the iteration. However, this does not mean that every ESO step is a sensible step because the reduction in weight may be accompanied by an increase in the maximum stress within the structure. Often, in practice, the maximum allowable stress is specified and one looks for designs that reduce the overall weight. When comparisons of different designs or design concepts need to be made, a rational approach will be to account for

the two factors together and view the comparison problem as a multi-objective optimization problem with the weight and maximum stress to be minimized simultaneously. This is convenient for design comparisons; however, for practical problem solving, the conventional approach of treating the overall weight as an objective and the maximum allowable stress as a constraint may still be used.

4.3. Pareto Fronts by exhaustive design search

The ESO algorithm evolves by shedding portions of the current design during design search in favour of highly stressed regions in the design. A difficulty with this approach is that while the evolution occurs, there is very little scope for reconsideration of design features that have been removed by a previous stage. In other words, the design tends to commit to certain features too early, approaching a gradient descent approach. A study of the evolution trajectory during the ESO process should be able to examine these issues.

In order to assess the quality of designs produced by the ESO process, all the possible designs of a structure will be considered for a prescribed given number of members eliminated from the original ground structure. Then the PF of all the possible designs for a given number of removed members is obtained and it is compared with the corresponding design that ESO obtains for the same number of members eliminated from the original design. A computer code (written in MATLAB) was developed as part of this study to find all the possible designs with only one member removed. Then the code plots the corresponding maximum stress and current weight of each of the possible designs in the same graph and finally finds out the PF. A set of good designs with one member removed are produced and in this way non-competitive (dominated) designs are eliminated from consideration. A design is dominated if another design has been found that is better in both objectives. After that the PF of all the possible designs when one member is eliminated can be compared with the design that ESO finds when one member is removed. Similarly, the procedure is continued for the possible designs that have 2 members removed from the ground structure, then those that have 3 members removed from the ground structure, and so on.

Suppose the total number of members in the fully connected design is n. We can produce new designs by systematically eliminating r members at a time. The total number of design options can be calculated as follows:

If n_{C_0} : *No of designs with no members removed,*

n_{C_1} : *No of designs with 1 member removed,*

n_{C_2} : *No of designs with 2 members removed,*

$\vdots$

n_{C_r} : *No of designs with r members removed,*

then $\sum_{i=0}^{r} n_{C_i} \equiv$ *total number of designs*, with r or less number of members removed from the original completely connected design.

This method of calculating all the possible designs that can exist, for a structure having N number of nodes and 1 to r number of members removed, can be applied only in discrete structures. Therefore, discrete structural problems such as trusses and frameworks provide an ideal setting to study design options exhaustively. The number of the total possible different connectivities in a discrete structure can be large but finite in comparison with the topology in continuum structures where it is considered as infinite and dependent on the parameterization.

As already noted, in a framework design, certain nodes are important and must exist in any feasible design. Moreover certain nodes are added for load sharing and are optional. The important nodes are usually the ones that carry a load or which support the framework. This information can be specified by the designer and must be kept in mind in the design process of a structure. On the other hand, the optional nodes are sometimes used in a frame to help distribute the stresses better to individual members. This constraint is checked first in the program. If any one of these important nodes is absent in the design, a large penalty is assigned to the solution and no further calculation of objective function or constraint is done. This approach is introduced to enhance the creation of satisfactory structures and also to save computations by not performing expensive FEM analysis for unsatisfactory designs.

In our case, we assume a ground structure, which is a complete framework with all possible member connections among all nodes in the structure. Thus, in a framework having N nodes, there are a total of $M = N \times (N-1)/2$ different members

and 2^M total number of possible designs (this includes the trivial cases of fully connected design and infeasible designs such as all members removed).

In a feasible framework, all members must have stresses within the allowable strength of the material. Since usually a framework is subjected to a number of different loading conditions applied separately, these constraints must be used for each loading condition. Since several design options need to be evaluated, a computer model to calculate stress and weight is required. Thus, we have used a finite element method (FEM) to calculate the stresses in a framework. Furthermore, a suitable automatic node re-numbering scheme is developed. Since the design options have different topology, the members and nodes of the framework need to be automatically numbered before calling the FE code. A flow chart of the scheme implemented to obtain the PF of designs with a specified number of members missing from the ground structure is shown in Figure 4.1.

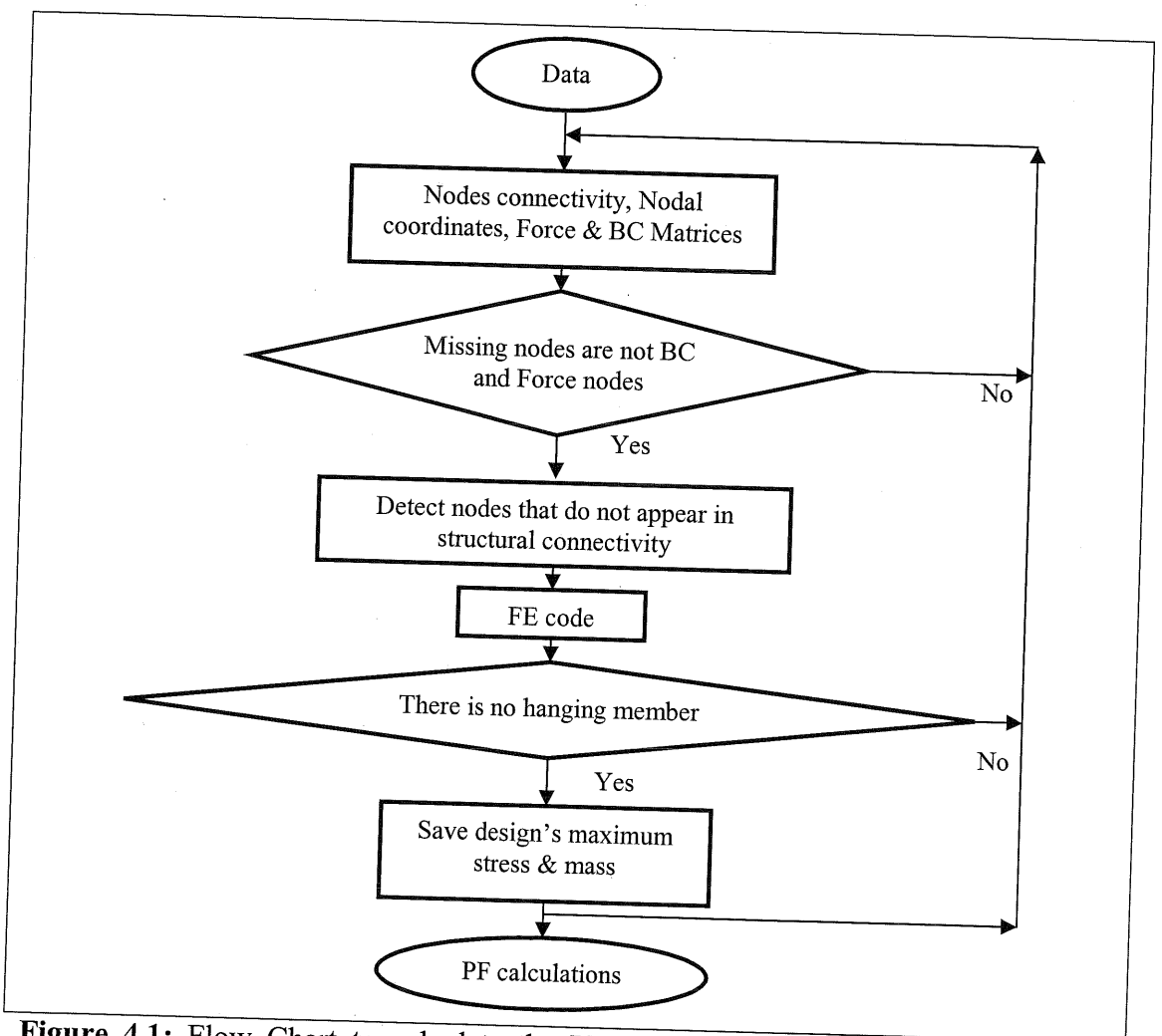


Figure 4.1: Flow Chart to calculate the PF for a prescribed number of members missing from the fully connected design.

4.4. Pareto-comparison of ESO with exhaustive search: results and discussions

Although the complete PF of the 6-node structure has been obtained, the PF for 1 to 7 members removal cases are presented here in Figure 4.2a. The lines with numbers marked on them correspond to the PF of designs obtained by exhaustive search for those many number of members removed from the ground structure. The complete PF of the 6-node structure for up to 7 members is then obtained and presented in Figure 4.2b. This PF will be used for comparisons of the methods developed for the 6-node problem in the next chapter.

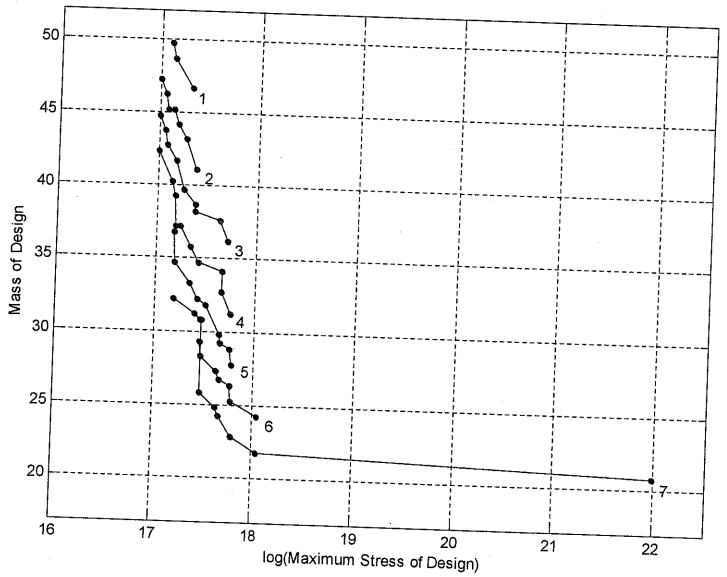


Figure 4.2a: Pareto Fronts for 1 to7 members removal cases of 6-node structure.

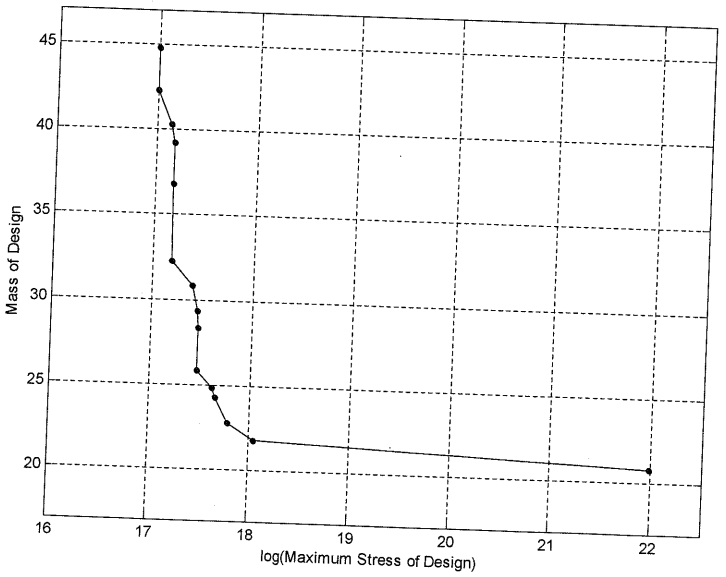


Figure 4.2b: Pareto Front of the 6-node structure as obtained by the exhaustive search when up to 7 members are removed.

The PF calculations performed on a 6-node framework having 15 members when one member is removed are presented in Figure 4.3. The three designs on the PF are presented in Figure 4.4a, Figure 4.4b and Figure 4.4c. Then, running ESO for the same structure with 15 members, the design that we get when one member is removed is dominated by the designs that are on the PF as can be seen in Figure 4.3. As in all design problems considered in the current work, the load applied to the structure is equal to 10 kN. This ESO design is shown in Figure 4.5. Design D1 on the PF has the same weight as the ESO design, but slightly lower maximum stress value and for this reason D1 dominates the design obtained by ESO. Design D1 obtained by the exhaustive search has member 4-6 removed from the structure (this cannot be seen clearly in Figure 4.4a as collinear members exist in the structure). As we can see from Figure 4.4b and Figure 4.4c, in D2 design the member 4-5 is removed and in D3 design the member 1-6 is removed.

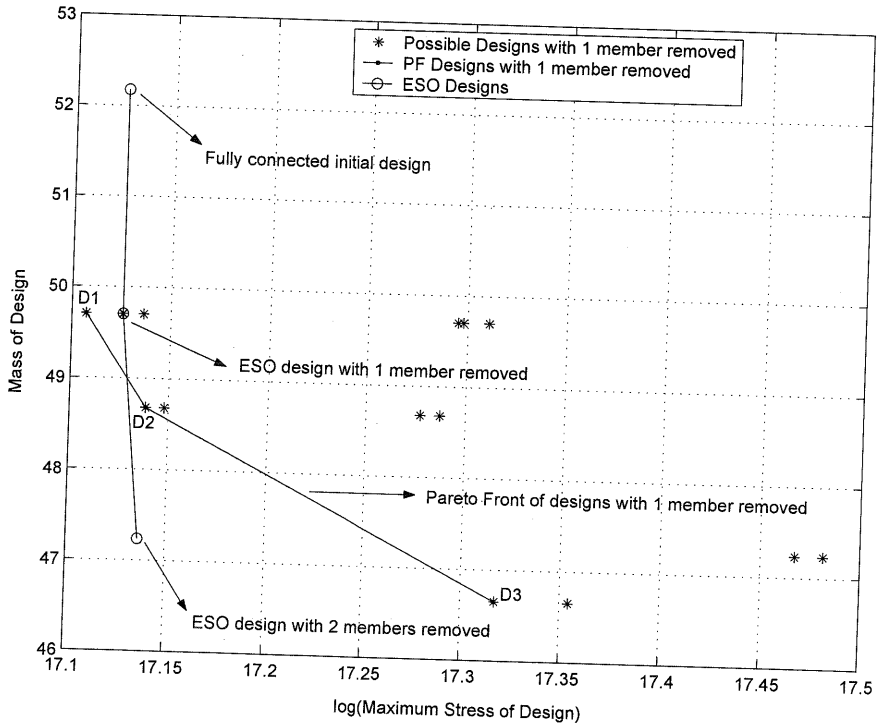


Figure 4.3: Comparison of the ESO method and PF of the exhaustive search for the case of one member removal.

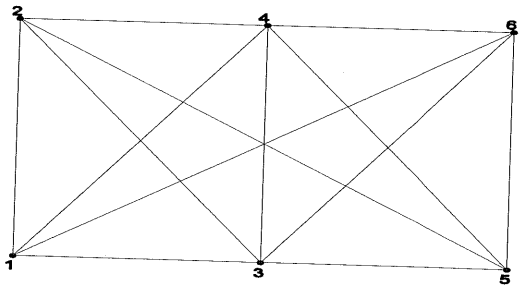


Figure 4.4a: Design D1 of the PF.

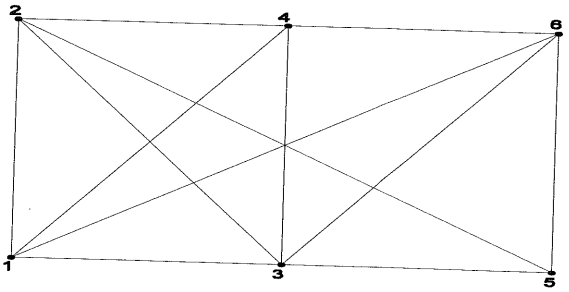


Figure 4.4b: Design D2 of the PF.

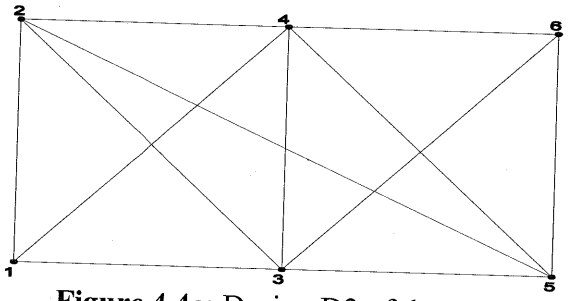


Figure 4.4c: Design D3 of the PF.

Figure 4.5 shows that the first member that the ESO removes is 1-2. As this member connects the two fixed nodes that support the structure, it has zero stress and therefore ESO decides to remove it first. On the other hand, D1 on the PF has the same weight as the ESO design with 1 member removed (because one member of the same weight is removed in both cases), but the structure's maximum stress for D1 is less than that of the corresponding ESO design. Removing a member that is stressed instead of a member that has zero stress, the stress in the structure is distributed to the rest of the members of the structure, decreasing that way the maximum stress of the structure. Removing the member 1-2 leads only to the decrease of weight, as the maximum stress of the structure over all members is kept the same.

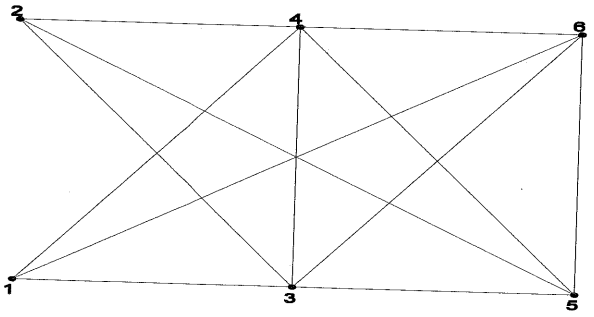


Figure 4.5: ESO design with one member removed.

In Figure 4.6, the ESO trajectory applied to this structure and the PF for each case of member removal are compared for the case of 2 members removed. It can be seen again that the design found by the ESO for 2 members removed is dominated by the corresponding PF designs.

The ESO trajectory and the PF for a specified number of members removed (1 through to 6) are presented in Figure 4.7. The lines with numbers marked on them correspond to the PF of designs obtained by exhaustive search for those numbers of members removed from the ground structure.

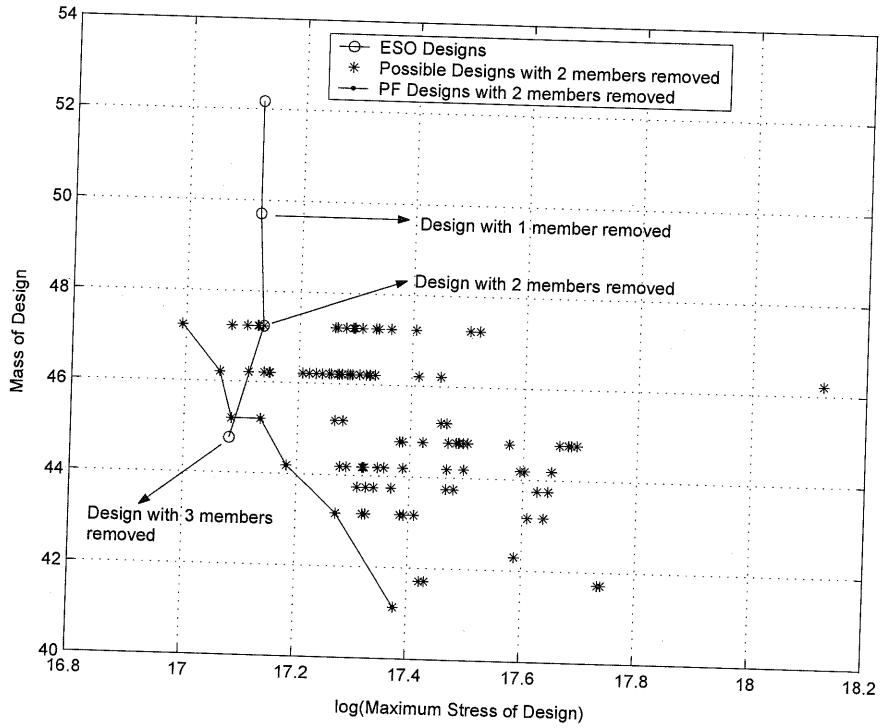


Figure 4.6: Comparison of the ESO method and PF of the exhaustive search for the case of 2 members removal.

It can be observed in Figure 4.7 and Figure 4.8 that the ESO design for 1, 2, 3 and 6 members removed is dominated by the PF of all the possible designs with the same number of members removed from the initial structure. Interestingly, the designs when 4 and 5 members are to be eliminated fall on the corresponding PFs showing that the ESO trajectory does hit the front for these two instances; following this it goes in the region of sub-optimal designs in the next cycles of member removal.

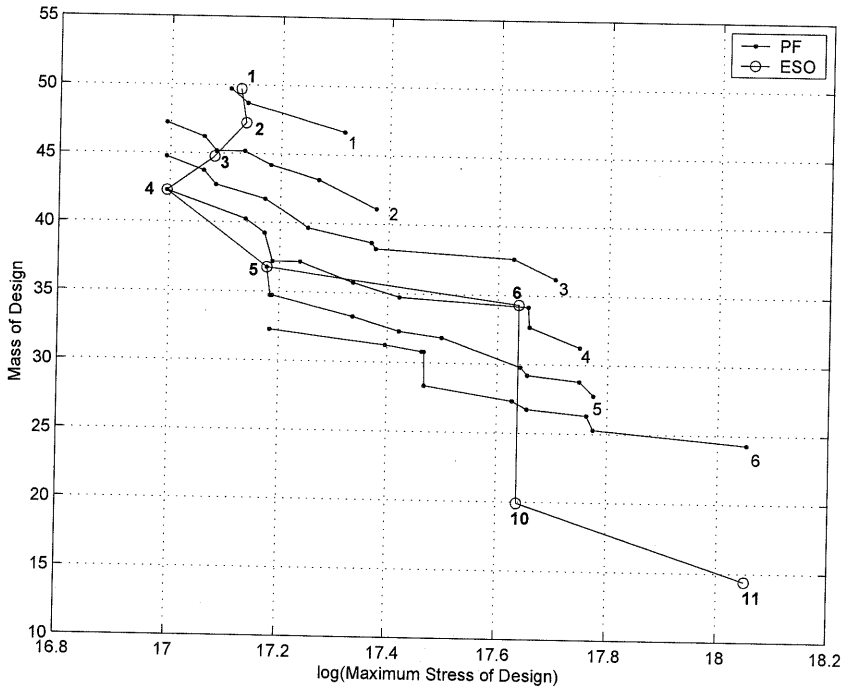


Figure 4.7: Comparison of the ESO method and PF of the exhaustive search for the cases of 1 to 6 members removal.

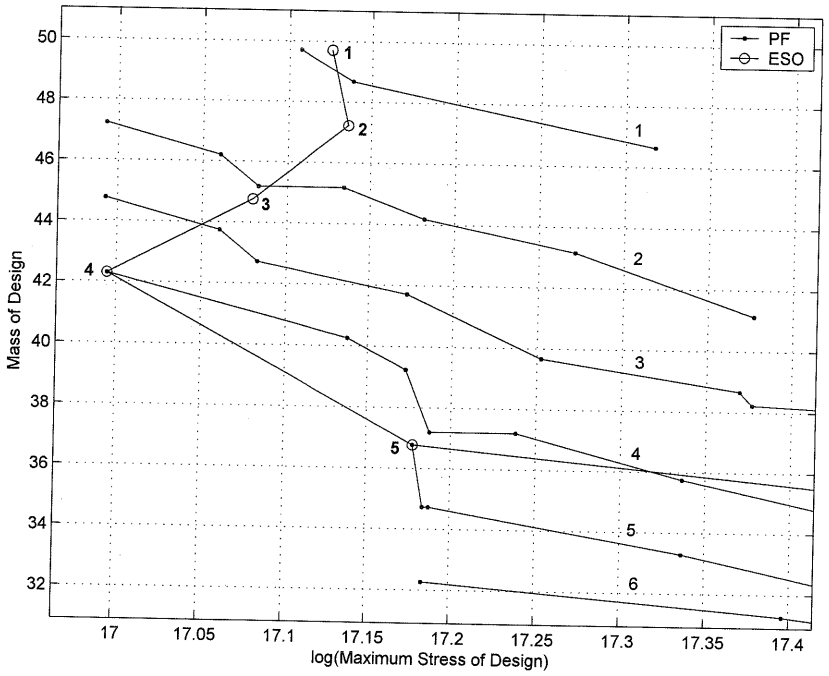


Figure 4.8: Closer view of the comparison of the ESO method and PF of the exhaustive search for the cases of 1 to 6 members removal.

A general conclusion that arises out of this numerical experiment is that ESO is sub-optimal and it does not always find the best design that could exist when a specified number of members are removed from the original fully connected structure. In other words, removing the least stressed members from a design may not always be the best design search strategy.

All the possible designs of a 6-node structure could be computed exhaustively and compared with the corresponding ESO designs for any member removal case. It is not possible to test if the same general conclusions hold for more complex design problems by using exhaustive search because the number of design options increases dramatically with increased number of members in the ground structure.

For various numbers of designs N and numbers of members removed, the computational effort is estimated. For a simple case of a 6-node framework, we can see from the following graph (Figure 4.9) that we can get a large number of possible designs. In particular the number of possible designs for this structure is equal to $2^{\frac{6\times(6-1)}{2}} = 2^{15} = 32768$ designs.

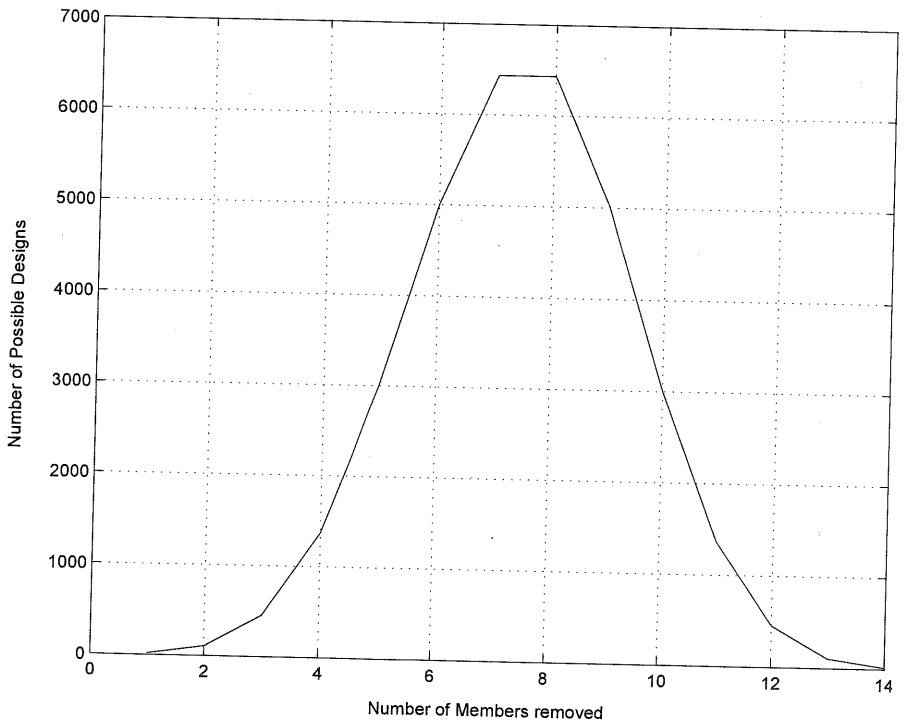


Figure 4.9: Graph of Number of Members removed against the corresponding Number of Possible Designs, for a 6-node framework.

Even greater is the number of the possible designs, when larger structures are considered, making the solution time enormous. For example, in the case of a 12-node framework with 66 members in total, the number of all the candidate designs is equal to 2^{66}, which is approximately 7.38×10^{19} designs. The variation of the possible designs according to the number of the removed members is plotted in Figure 4.10. As we can notice from this figure, the direct evaluation of a finite element model such as a simple 12-node framework can take a lot of time as it is almost impossible to try several billions of designs.

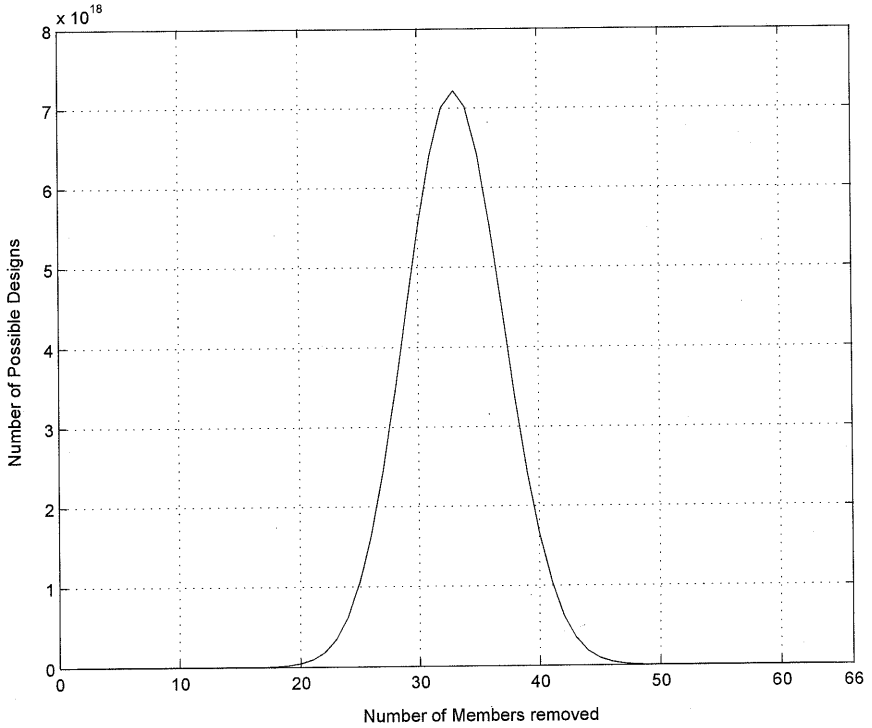


Figure 4.10: Graph of Number of Members removed against the corresponding Number of Possible Designs, for a 12-node framework.

To overcome the problem of large number of candidate designs in a structure, Genetic Algorithms will be used to search for Pareto-efficient designs next. If the number of possible design options is large, exhaustive search may be computationally prohibitive. A genetic algorithm may then be a natural choice for searching for a Pareto set. Genetic algorithms begin with a starting generation of designs and proceed from generation to generation. Here the goal is that the final generation approaches the Pareto set for the universe of possible designs for the case of two competing objectives—maximum stress and weight.

4.5. Structural Topology Optimization using Genetic Algorithms

A GA is a search and optimization procedure that is motivated by the principles of natural genetics and natural selection. Some fundamental ideas of genetics are borrowed and used to construct search algorithms that are robust. The main difference between the GA and most of the traditional optimization methods is that the GA works with a population of solutions instead of a single solution [97-100]. There is more than one string (representing a design) that is processed at the same time and used to update any one string in the population. For this reason, the expected GA solutions may be a global solution. Moreover, since a population is updated at every generation, a set of solutions (such as in Multi-objective Pareto Optimization) can be obtained simultaneously.

A simple GA proceeds by first randomly generating a population of solution strings. A pseudo random number generator is used to generate the initial population. Based on the statistics of this population, the next generation is reproduced according to probabilities assigned to the members. This means that poor designs will be assigned low probabilities and good designs will be assigned high probabilities of surviving in the next generation.

The population is a set of configurations called chromosomes. Chromosomes are encoded data and the chromosome values (genotypes) are uniquely mapped onto the decision variable (phenotypic) domain. The basic GA operators are listed as following (Figure 4.11):

(i) Selection: is the process of determining the number of times that a particular individual is chosen for reproduction and the number of offspring that an individual will produce. The selection operator is intended to improve the average quality of the population by giving individuals of higher fitness a higher probability to be copied to the next generation.

(ii) Crossover (recombination): is the basic operator for producing new chromosomes in the GA. It produces new individuals that have some parts of both parents' genetic material. The crossover operator is intended to combine the genetic data of the existing population in generating offspring. Pairs of chromosomes are recombined on a random basis to form two new individuals.

(iii) Mutation: is randomly applied with low probability to modify values in the chromosomes. The mutation operator plays a secondary role, as it allows new genetic patterns to be formed, thus improving the search method. Occasionally, it protects some useful genetic material against loss.

(iv) Elitist strategy: in a standard GA the best possible solution is not preserved, thereby increasing the chance of losing the obtainable best possible solution. The elitist strategy overcomes this problem by copying the best member of each generation into the next one.

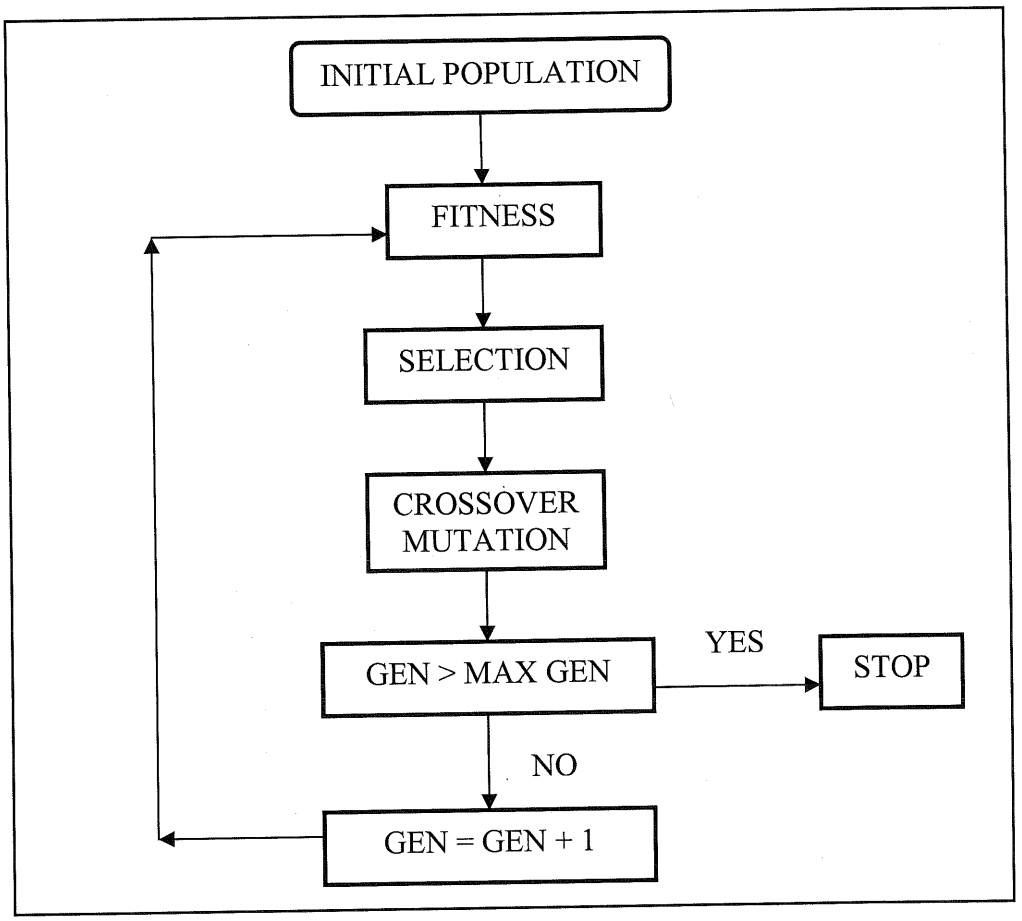


Figure 4.11: Outline of a genetic algorithm.

At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using crossover and mutation operators – ideas borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from.

Another important characteristic of GA is that it uses probabilistic rules to guide the search. A common problem with most of the traditional search methods is that there are fixed rules to move from one solution to another [98]. Because of that, these methods can only be applied to a special class of problems. They are not considered robust and cannot be applied to a wide variety of problems. In trying to solve any other problem, if a wrong move is made early on, it is very hard to recover from this. On the other hand, GA uses probabilistic rules and an initial random population [101-105]. Thus, early on, the search may proceed in any direction and no major decision is made in the beginning. Later on, when the population has converged in some locations the search direction narrows and a near-optimal solution is found. The nature of narrowing the search space as generations progress is adaptive and is a unique feature of the GA. This characteristic gives to GA robustness, which is very useful in optimization problems.

The first contribution to optimal design of structures using GA is presented in [106]. They discussed the optimal design of a 10-bar truss structure using continuous variables. Also, with regard to the optimization of truss structures cross-section sizing optimizations of discrete-member trusses were investigated [104, 107-109]. Moreover, modified versions of bit-string coded GA have been proposed [110-112], considering discrete variables. The shape optimization of structural members has been studied in [102, 103].

GA-based topology optimization of discrete truss structures was investigated mainly in [107, 109, 113]. A two-stage optimization process was used in [107, 109]: (1) in the first stage, kinematic stability requirements were used to identify stable topological configurations and (2) in the second stage member adding/removal and re-sizing were considered. The sizing, shape and topology of frame structures were addressed in [113], which was one of the first papers to examine the design of framed structures using GAs.

The bit-array representation method [89] is similar to the binary-string method [114] in which they described a binary material/void design representation that is encoded in GA chromosome data structures. This representation was intended to approximate a material continuum as opposed to discrete truss structures.

Size/topology optimization of trusses has been performed using some concepts of the force method, graph theory and genetic algorithm [105]. The design variables

consist of cross-sections of members and topological Boolean bits corresponding to presence or absence of members.

Furthermore, a bit-array representation for structural topology optimization of continuous structures using GA was implemented in [115]. A bit array is mapped into the two-dimensional design domain discretized by a fixed regular mesh, where each of the small, square elements contains either material or void and is thus treated as a binary design variable.

4.6. Quality indicators

Since interest in applying randomized search algorithms in order to obtain Pareto-optimal solutions is constantly growing, the issue for comparing the performance of different algorithms has become more and more important. In multiobjective evolutionary algorithms, the outcome is an approximation of the Pareto-optimal set, which will be called as the approximation set [116]. The main issue now is to evaluate the *performance of approximation sets*. However, in this thesis we will use the term "design sets" instead of "approximation sets". Two basic approaches of evaluating the performance assessment of multiobjective algorithms are the attainment function approach, which models the outcome of a multiobjective optimization algorithm as a probability function in the objective space [117] and the indicator approach, which summarizes the outcome of a run on the basis of quantitative performance measures [118]. In this work the indicator approach is followed in order to compare the output of the new evolutionary methods that are developed.

When visual comparisons of multiobjective optimization methods are not feasible, quantitative methods of comparing and evaluating the performance of different optimization methods are needed. In order to assess the performance of two multiobjective optimization methods and find which outperforms the other, two measures of quality are considered here. The performance measures used in this work are termed as quality indicators since are focused on the quality aspect while comparing several design sets [119].

Quality indicators represent means to express and measure quality differences between design sets (see Figure 4.12). The most popular quality measures are the so called *unary quality indicators*, which are functions that map each design set to a real number as a single figure of merit and the algorithms are assigned samples of corresponding indicator values. On the basis of statistical testing procedures, it is then possible to check whether an algorithm provides significantly better solutions than another optimization method with respect to the preferences represented by the considered indicator.

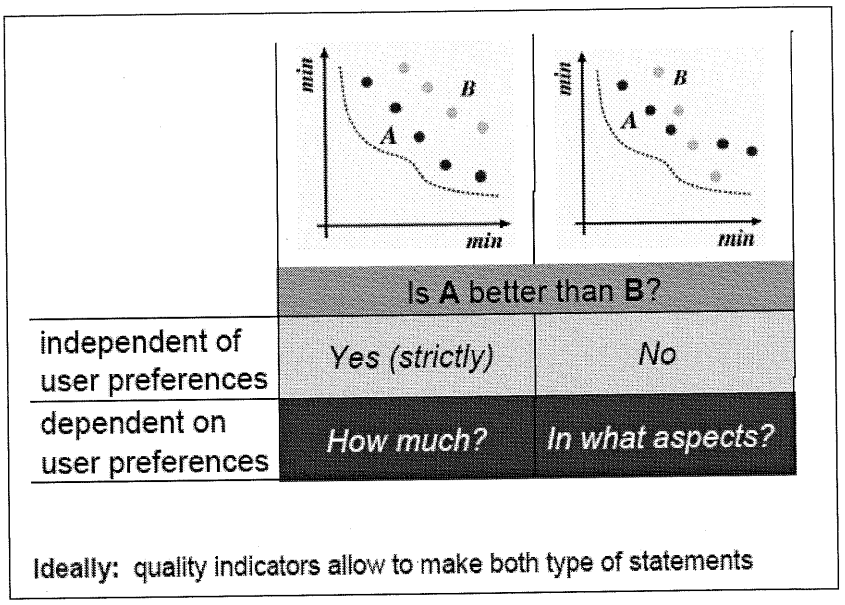


Figure 4.12: The need for quality indicators from [120].

According to [119], unary indicator I_{ind} is defined as a mapping from the set of all approximation sets Ω to the set of real numbers:

$$I_{ind} : \Omega \rightarrow \mathrm{R}.$$

Given a pair of approximation sets, B and C, the difference between their corresponding indicator values $I_{ind}(B)$ and $I_{ind}(C)$ should reveal a difference in the quality of the two sets even when the two sets are considered as incomparable. For example, if the indicator values are assumed to be maximized, B is preferable to C if $I_{ind}(B) > I_{ind}(C)$. So, the outcomes of two multiobjective optimizers can be compared by simply comparing their corresponding indicator values.

In the following work, three common unary quality indicators are used for the purpose of comparing the quality performance of pair of sets when visual evaluation becomes difficult. Each of these indicators is considered to measure the performance of each design set in a slightly different way. These three unary quality indicators are presented next.

4.6.1. The Hypervolume Indicator

The hypervolume indicator, proposed in [83], measures the hypervolume of the portion of the objective space that is dominated by a design set B, and is to be maximized. In order to measure this quality, the objective space must be bounded, otherwise a bounding reference point that is dominated by all points should be used (Figure 4.13).

Also, the hypervolume of the objective space can be calculated according to a reference set R, giving the indicator referred as I_H^-. "Given an approximation set A, the indicator value is defined as

$$I_H^-(A) = I_H(R) - I_H(A) \tag{4.1}$$

where smaller values correspond to higher quality, in contrast to the original hypervolume I_H" [119]. Given a pair of design sets, B and C, the difference between their corresponding indicator values $I_{ind}(B)$ and $I_{ind}(C)$ should reveal a difference in the quality of the two sets. In particular, for a minimization problem, if the reference set is considered to dominate the sets B and C, then B is better than C if the hypervolume value of B $I_{ind}(B)$ is smaller than the hypervolume of C $I_{ind}(C)$. Accordingly, if a reference set is chosen to be dominated by both design sets, then B is better than C, only if the hypervolume of B is larger than the hypervolume of C, considering the objective space is to be minimized.

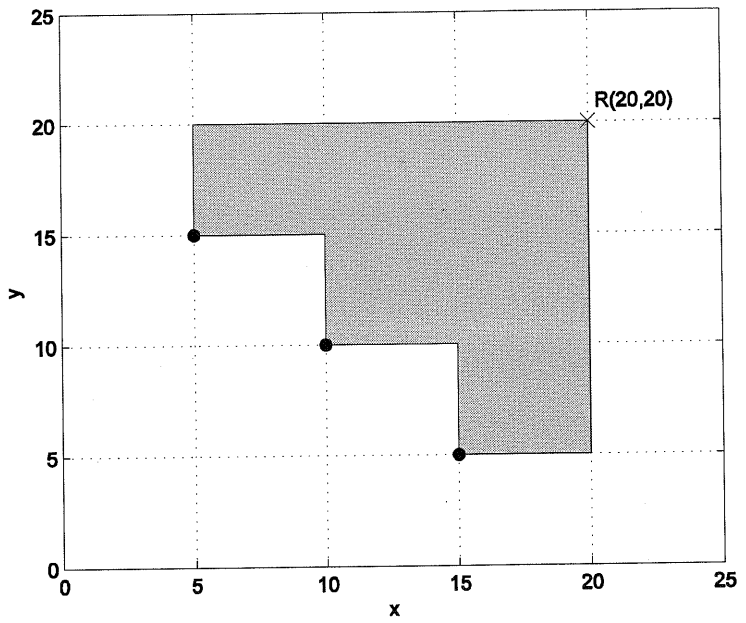


Figure 4.13: Illustration of the hypervolume indicator. In this example, design set B is assigned the indicator value $I_H(B) = 150$. The objective vector (20, 20) is taken as the reference point.

4.6.2. The Unary Epsilon Indicator

The epsilon indicator family has been proposed in [116]. Multiplicative and additive versions exist in both unary and binary forms. The binary multiplicative epsilon indicator $I_\varepsilon(B,C)$, gives the minimum factor ε, by which each point in C can be multiplied such that the resulting transformed set is dominated by B. The epsilon indicator I_ε can be defined as:

$$I_\varepsilon(B,C) = \inf_{\varepsilon \in R}\left\{\forall z^2 \in C \exists z^1 \in B : \forall i \in 1..n,\ z_i^1 \leq \varepsilon \cdot z_i^2\right\}, \qquad (4.2)$$

for a minimization problem of any two design sets B, $C \in \Omega$, corresponding objective vectors z^1, z^2 and assuming that all points are positive in all objectives [116]. Similarly, the unary multiplicative epsilon indicator $I_\varepsilon^1(B)$ for a set B is

$$I_\varepsilon^1(B) = I_\varepsilon(B,R), \qquad (4.3)$$

where R is any reference set of points. An equivalent unary additive epsilon indicator

$I^1_{\varepsilon+}(B)$ is defined in a similar way, but it is based on additive ε-dominance

$$I_\varepsilon(B,R) = \inf_{\varepsilon \in R}\left\{\forall z^2 \in R \exists z^1 \in B : \forall i \in 1..n,\ z_i^1 \le \varepsilon + z_i^2\right\}. \tag{4.4}$$

For the unary epsilon indicators, whenever B is better than C, then $I^1_\varepsilon(B) \le I^1_\varepsilon(C)$ respectively $I^1_{\varepsilon+}(B) \le I^1_{\varepsilon+}(C)$. On the other hand, if $I^1_\varepsilon(B) > I^1_\varepsilon(C)$ respectively $I^1_{\varepsilon+}(B) > I^1_{\varepsilon+}(C)$, we can deduce that B is not better than C [116].

4.6.3. The Unary *R* Indicator

The R indicators proposed in [121] can be used to assess and compare design sets on the basis of a set of utility functions. "A utility function u is defined as a mapping from the set Z of n-dimensional objective vectors to the set of real numbers:

$$u : Z \mapsto R.$$

In this sense, it represents the counterpart to a quality indicator with the difference that the domain is Z and not Ω" [119].

Supposing that the decision maker's preferences are given in terms of a parameterized utility function u_{lambda} and a corresponding set Λ of parameters, u_λ could represent a weighted sum of the objective values, where $\lambda = (\lambda_1,...\lambda_n) \in \Lambda$ is a particular weight vector. Several ways to transform such a family of utility functions into a quality indicator were proposed in [121]. The binary I_{R2} and I_{R3} indicators are defined as:

$$I_{R2}(B,C) = \frac{\sum_{\lambda \in \Lambda} u^*(\lambda,B) - u^*(\lambda,C)}{|\Lambda|}, \tag{4.5}$$

$$I_{R3}(B,C) = \frac{\sum_{\lambda \in \Lambda} [u^*(\lambda,C) - u^*(\lambda,B)]/u^*(\lambda,C)}{|\Lambda|}. \tag{4.6}$$

where u^* is the maximum value that the utility function u_λ can reach with weight vector λ on a design set B, i.e., $u^*(\lambda,B) = \max_{z \in B} u_\lambda(z)$ [119].

As in the case of the epsilon indicators, the unary R indicators are defined on the basis of the binary versions by replacing C by an arbitrary, but fixed, reference set: $I^1_{R2}(B) = I_{R2}(R,B)$ and $I^1_{R3}(B) = I_{R3}(B,R)$. The R indicators can guarantee that, in the case of minimization, the indicator value for a design set B is less than or equal to the indicator value associated with C, whenever B is better than C.

4.7. Implementation of GA for 2-objective optimization problems using NSGA algorithm

A basic part of the GA procedure is to enhance good solutions and eliminate poor solutions in a population, while keeping the population size constant. Therefore, although the complete design space is not explored, the chances of sampling the most promising designs are greatly enhanced given the computational constraints.

The fact that a simple framework can have an extremely large number of possible designs when a number of members are removed makes it very difficult to investigate all the designs and detect which of these appear on the Pareto Font. In addition, it takes a lot of time to run all the cases, as for a normal case of a relatively small number of members in a structure; the possible designs can easily reach a number of the order of billions. The use of Genetic Algorithms will reduce the overall computation time substantially. In this way, the designs that are not promising will be ignored during the GA runs.

The aim is to produce a well spread out set of optimal designs, with as few function evaluations as possible. The former is to ensure a good and even resolution of the Pareto Front. There are number of methods published in order to satisfy this. Generally the most preferred in the literature seems to be NSGA2 (Non-dominated Sorting in Genetic Algorithms, see [122]).

The algorithm presented in [122] has been implemented to generate the PF in this work. In order to find a number of Pareto-optimal solutions in a multi-objective optimization problem using GA, the concept of non-dominated sorting of population members is used [122, 123]. In a population, the non-dominated solutions are defined as those solutions which are better in at least one objective than any other solution in the population. The procedure that is adopted in order to implement this non-

dominated sorting concept follows. The population is sorted to find the non-dominated set of solutions. Then all the individuals in this sub-population are assigned a large artificial fitness value. The fitness of each solution string is a measure of performance of the design variables defined by the objective function.

Since the objective is to find a number of Pareto-optimal solutions, a sharing procedure is performed among these non-dominated solutions and a new shared fitness is calculated for each of these solutions. These solutions are temporarily counted out of the population and the next non-dominated set is found. These solutions are assigned an artificial fitness value smaller than the least shared fitness value in the previous non-dominated set. This is done to impose a higher preference for solutions in the previous set than for the current set. Sharing is performed again among the new non-dominated set and this process continues till all population members are ranked in descending order of the non-dominated sets. Then the so-called reproduction operation takes place, in which the good solutions are kept and the poor solutions are eliminated.

Typically, a GA works with coding of variables instead of the variables themselves as in the traditional optimization methods [98, 109]. Here coding the decision variables in a binary string is used to achieve a pseudo-chromosomal representation of a solution. In our case, an M-bit-string is a representation of the designs generated from an M-member fully connected structure.

Considering the case of a framework that has M number of members, the connectivity of the structure can be coded in a binary string of length $1 \times M$, consisting of ones and zeros. When a member is removed, we associate it as a binary number "0" and if it remains, we associate it as a binary number "1". So, initially the search will begin from a design (initial seed) that has all the nodes connected to each other, which means that the string will include only 1's. If we call this string bitstring, then,

$$\text{bitstring} = [1 \quad 1 \quad 1 \quad 1 \ldots\ldots\ldots 1 \quad 1 \quad 1].$$

Now that a string representation of a design solution has been achieved, some genetic operations previously described are applied to such strings to hopefully find better populations of solutions. However, since GA has multiple offspring, it can explore the solution space in multiple directions at once. If one path turns out to be a dead end, it can easily eliminate it and continue work on more promising areas, giving

it a greater chance of finding the optimal solution. In order to avoid the appearance of infeasible designs, penalty features are also introduced in the algorithm.

The bitstring encoding of the connectivity is used as an input to the software OptionsMatlab [124]. In order to produce a well spread out set of optimal designs, with as few function evaluations as possible, the NSGA2 method [122] is adopted within OptionsMatlab (see Appendix). Using an improved NSGA2 algorithm and the OptionsMatlab interface, the number of function evaluations is reduced, and a high quality Pareto optimal front is achieved. This idea is implemented on a 6-node and 12-node ground structures. The ESO trajectory for each of these examples will be compared with the corresponding GA designs that will be obtained.

4.8. Pareto-comparison of ESO with GA: results and discussions

A small framework that has 6 nodes is considered first. This is a well-known test case in structural topology optimization of frameworks. It consists of 6 joints, two fixed supports and a load acting simultaneously. The ground structure is shown in Figure 4.14. We take this example because exhaustive search is feasible and it would give us confidence in the efficiency of GA while comparing with ESO.

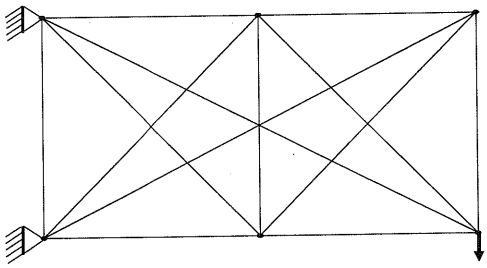


Figure 4.14: Fully connected framework with 6 nodes.

In this work, GA is run for 30 different randomly generated seeds in order to test the robustness of the conclusions derived. The final output of the multiple runs of GA contains the PF designs obtained from each GA run. For the problem of the 6-node structure, the designs obtained from GA for each different initial seed are presented on the same plot (Figure 4.15). The points denoted as circles and diamonds

represent the most efficient designs and the least efficient designs of the multiple GA search respectively.

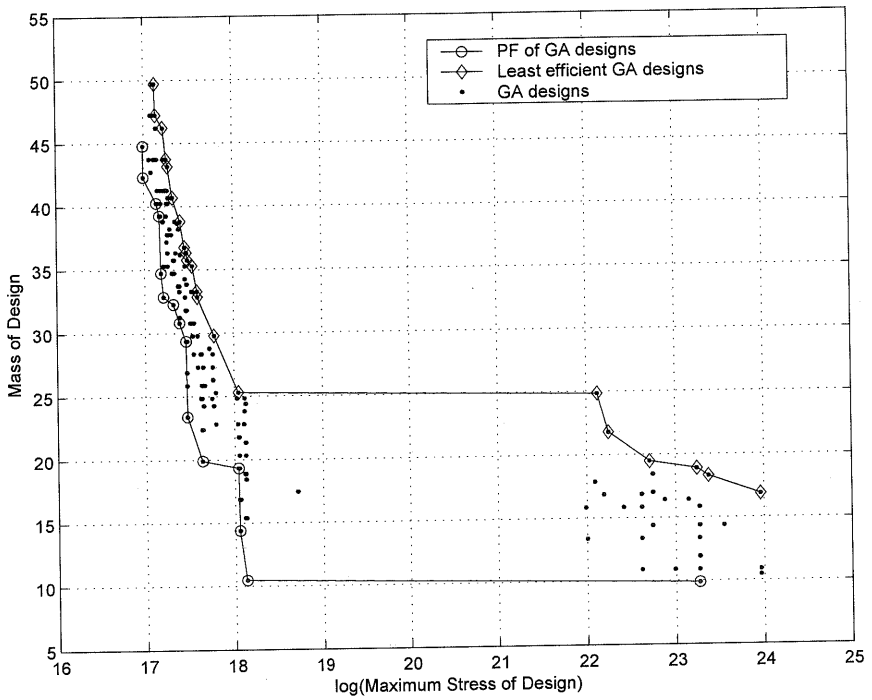


Figure 4.15: GA designs (20 generations and 20 designs in each population).

The results of the exhaustive search for 1 through to 5 member removal cases are compared with the designs obtained by GA for 20 generations & 20 designs per population in Figure 4.16. We observe that the GA finds designs obtained from the exhaustive search for 1-5 member removal cases. In addition, GA obtains superior results than the exhaustive search of the 5 member removal case in less computational time. GA obtains better designs than the exhaustive PF of 5 member removal case consuming time between the computational time of the 4 and 5 member removal cases as expected (Table 4.1).

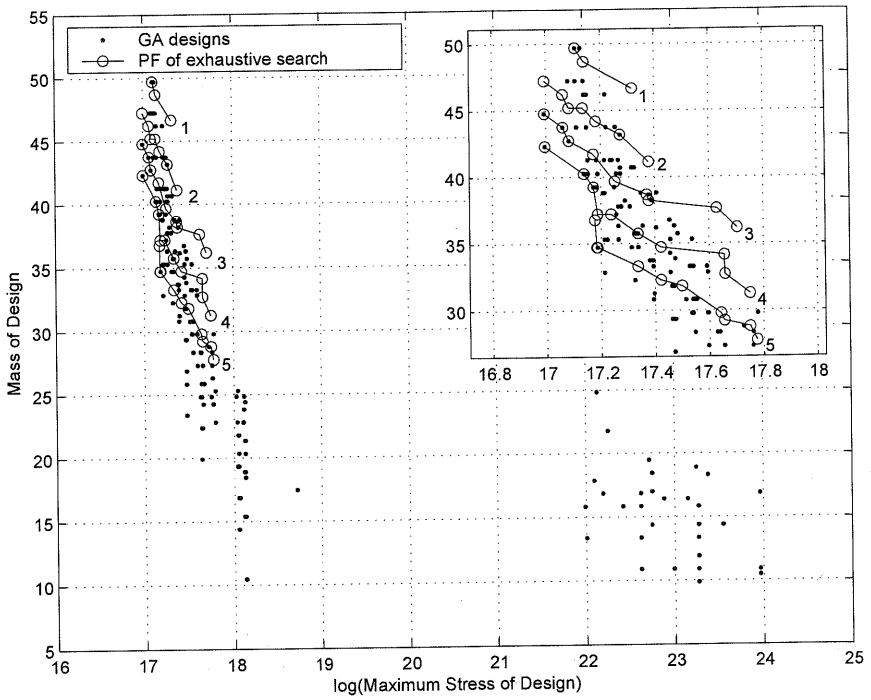


Figure 4.16: Comparison of the GA designs and the PF designs of the exhaustive search for 1-5 members removal cases (20 generations and 20 designs in each population).

	CPU Time (s)
Exhaustive search - PF of 1 member removal	3
Exhaustive search - PF of 2 members removal	5
Exhaustive search - PF of 3 members removal	30
Exhaustive search - PF of 4 members removal	257
Exhaustive search - PF of 5 members removal	1374
GA (20 generations & 20 designs in each population)	547

Table 4.1: Comparison of CPU time between exhaustive and GA search.

As the generation size increases from 20 to 80 and with the same applied set of 30 randomly generated seeds (but different from those for Figures 4.15 & 4.16), GA finds even better designs indicating convergence (Figure 4.17). Consequently, only GA will be used for computationally demanding problems where exhaustive search is impossible to carry out in future.

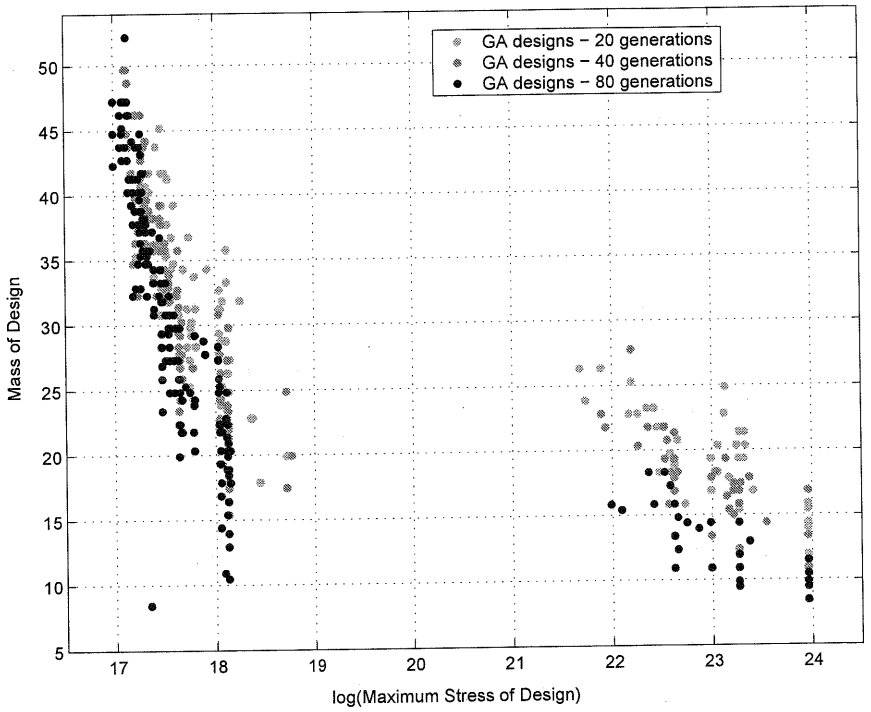


Figure 4.17: GA results for different numbers of generations.

The ESO trajectory of design evolution is presented overlaid on the designs explored using GA for 20 designs per population and 20 generations in Figure 4.18 and 40 designs per population evolved for 40 generations in Figure 4.19. As the number of populations and generations increases, GA can find more designs that dominate the designs in the ESO trajectory. We observe again that ESO captures some good designs (and in fact obtains the same three designs when 1, 2, 4, 10, and 11 members are eliminated from the ground structure) but for all other cases, the designs obtained by the use of GA outperform designs obtained by the use of ESO.

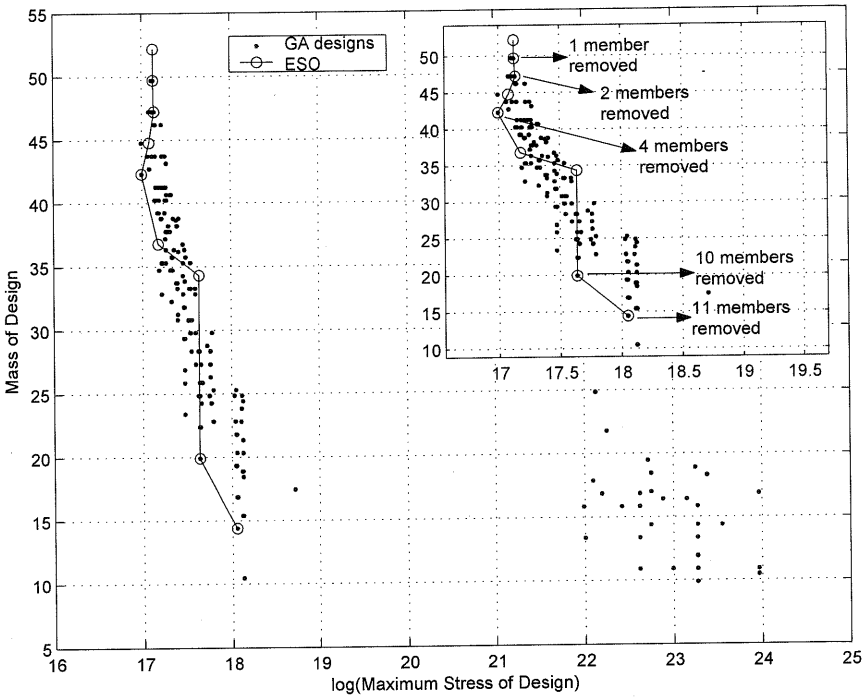


Figure 4.18: Comparison of the GA results and the ESO trajectory of the 6-node structure (20 generations and 20 designs in each population).

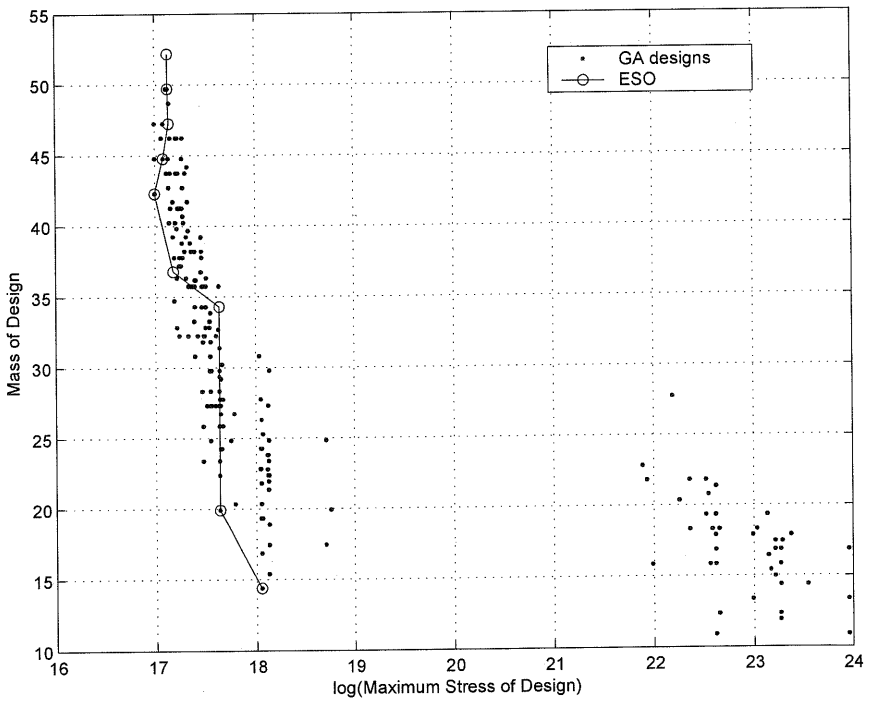


Figure 4.19: Comparison of the GA results and the ESO trajectory of the 6-node structure (40 generations and 40 designs in each population).

A larger structure that consists of 12 nodes and 66 members is considered next (see the ground structure in Figure 4.20). Exhaustive search is computationally prohibitive for this size of problem. The PF of this framework is found using the

NSGA2 algorithm and OptionsMatlab for 200 designs in each population and 200 generations (for 30 different randomly generated seeds) as can be seen in Figure 4.21. The figure indicates that GA finds some designs that are better than the designs obtained by ESO. This further supports the observation that the discrete ESO misses out good designs during its procedure, and hence it can be considered as a Pareto sub-optimum method. However, in this case the GA results do not dominate entirely the results of ESO. As the problem size is increased a larger number of generations and populations are required for GA to improve. Note that the differences on the x-axis are much greater than they appear because of the logarithmic scale chosen for the maximum stress within the structure.

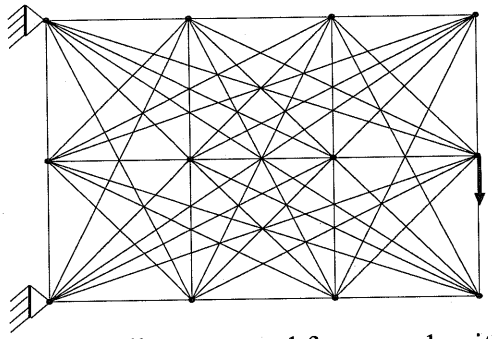


Figure 4.20: Fully connected framework with 12 nodes.

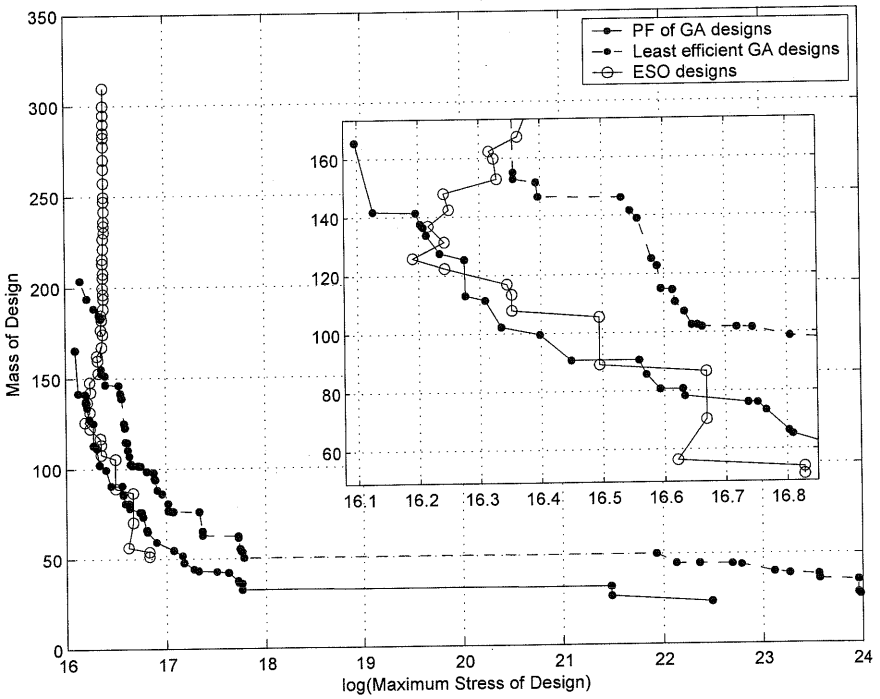


Figure 4.21: Comparison of the GA results and the ESO trajectory of the 12-node structure (200 generations and 200 designs in each population).

4.9. Sizing optimization using ESO in framework structures

In the problem of framework sizing optimization, cross-sectional areas of members are considered as design variables and the coordinates of the nodes and connectivity among various members are considered to be fixed. Here the discrete ESO method has been modified so that instead of completely removing low stressed members from the structure during the process, the thickness of the low stressed members of the structure is made smaller each time, reaching a smallest threshold value. Therefore the thickness of each member in the structure can vary in a range. The members that are less significant in the structure are assigned the smallest thickness value. This method simplifies the original ESO method with member removal, as no member is removed—thus reducing the computational time and effort. The two structures considered for numerical experiments are shown in Figures 4.14 & 4.20. The trajectory obtained by this ESO size optimization method with member thickness as a variable shows a similar trajectory to the original ESO method as can be seen in Figures 4.22 & 4.23.

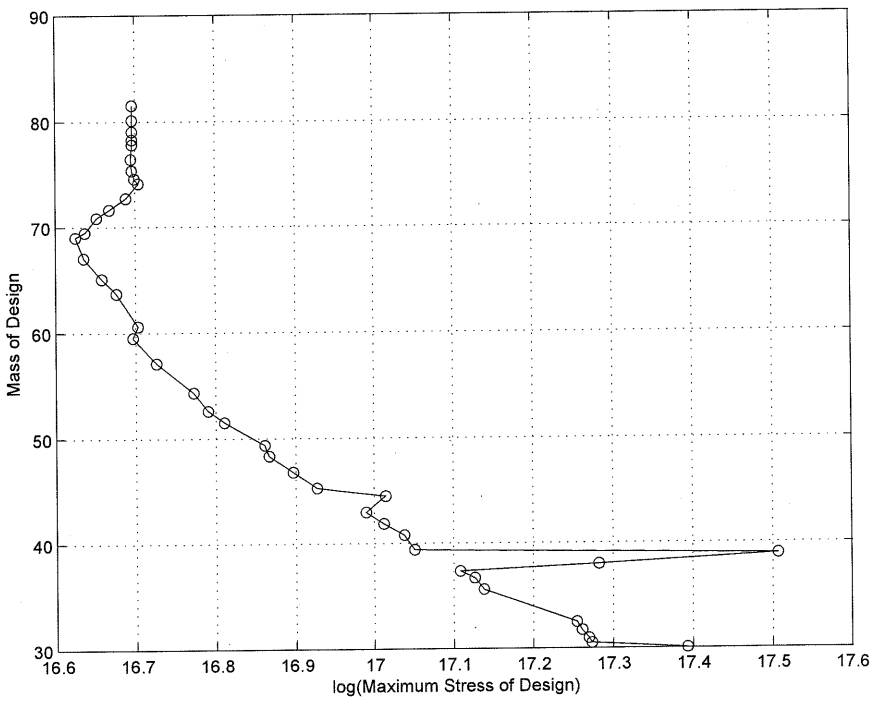


Figure 4.22: ESO trajectory for 6-node structure *with varying member thickness* (ranging from 0.005 to 0.025 m in steps of 0.005).

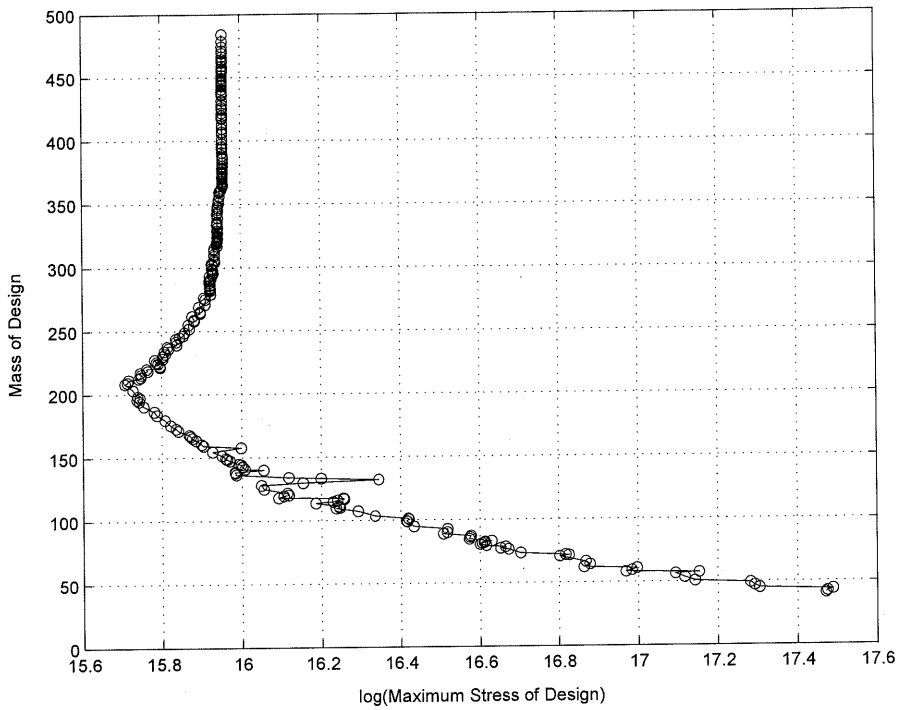


Figure 4.23: ESO trajectory for 12-node structure *with varying member thickness* (ranging from 0.005 to 0.025 m in steps of 0.005).

4.10. A bit-string representation for structural size optimization in discrete framework structures using Genetic Algorithms

In order to compare the results obtained by the ESO method with thickness as a variable, a bit-string representation method in GA is used again. This time, the bit-string represents the thickness of the each member of the structure, which varies within a defined range. Thus, OptionsMatlab works out structures with different values for the thickness of each structural member, obtained after a number of generations to produce the PF designs. The designs produced by the GA are compared with those obtained using ESO in Figures 4.24, 4.25 & 4.26. The ESO trajectory of the 6-node structure (see Figure 4.22) is overlaid with the GA results for 50 and 100 generations with 20 and 50 designs in each population correspondingly in Figures 4.24 & 4.25. Similarly, the ESO trajectory of the 12-node structure (see Figure 4.23) is overlaid with the GA results for 100 generations and 100 designs in each population in Figure 4.26.

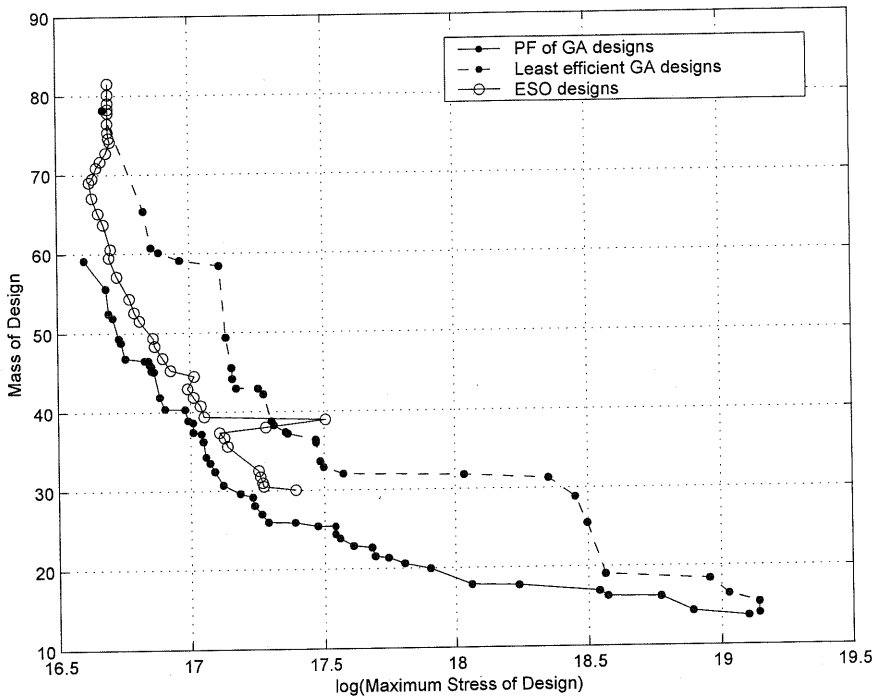


Figure 4.24: ESO & GA (50 generations & 20 designs in each population) comparison for 6 node structure with thickness as a variable ranging from 0.005 to 0.025 m in steps of 0.005.

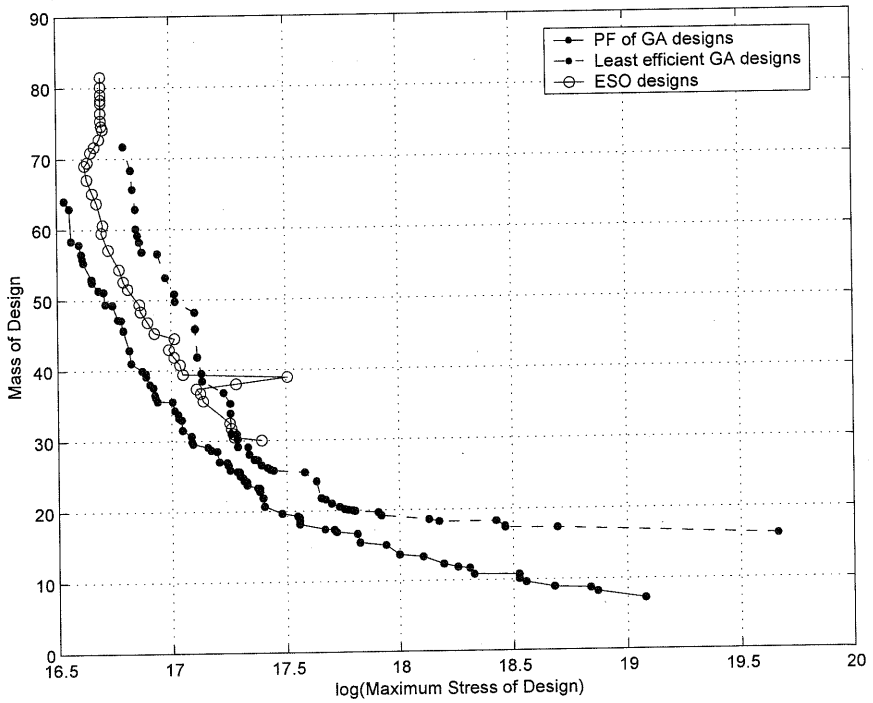


Figure 4.25: ESO & GA (100 generations & 50 designs in each population) comparison for 6 node structure with thickness as a variable ranging from 0.005 to 0.025 m in steps of 0.005.

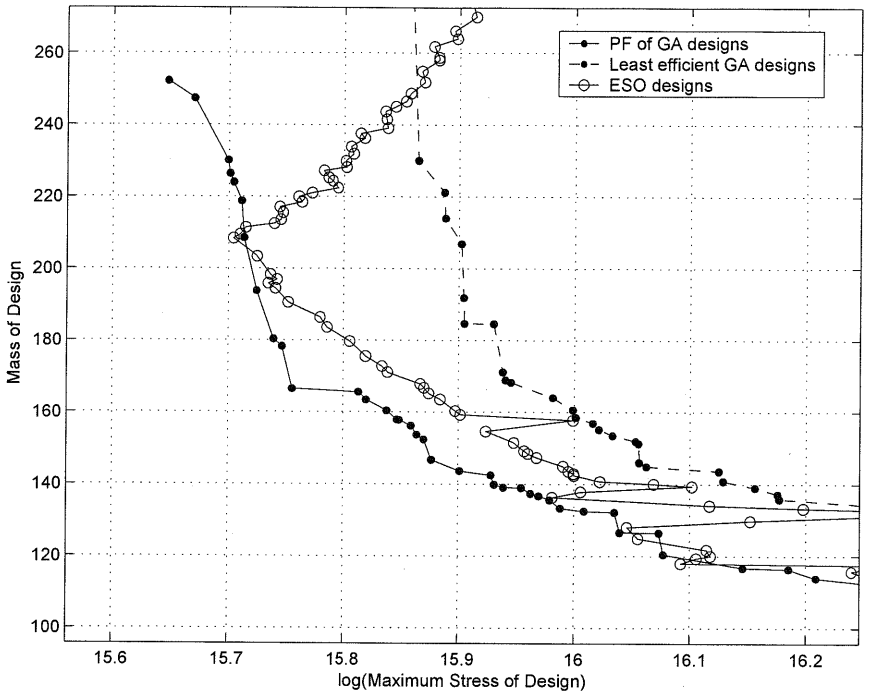


Figure 4.26: ESO & GA (100 generations & 100 designs in each population) comparison for 12 node structure with thickness as a variable ranging from 0.005 to 0.025 m in steps of 0.005.

The results illustrate again that the GA produces more efficient designs than ESO, although at a higher cost. On the other hand, ESO produces some remarkable designs if computational resources are limited.

The comparison of CPU time required for various examples presented in this thesis for ESO calculations and for GA calculations are given in Table 4.2 (the second column refers to the corresponding figure number). Comparisons with exhaustive search are meaningless – hence not made here – because it is always the most expensive search. However, when GA computation time is considered, it is clear that ESO designs require fewer calculations by several orders of magnitude. The gap in computational time increases with the increased complexity of the original fully-connected structure.

Example No.	Figure No.	CPU Time for ESO (s)	CPU Time for GA (s)
1	(4.18)	0.7	645
2	(4.19)	0.7	1975
3	(4.21)	1.34	95422
4	(4.24)	0.55	2864
5	(4.25)	0.55	16071
6	(4.26)	6.15	24118

Table 4.2: Comparison of CPU time for ESO and GA calculations.

4.11. Conclusions

The quality of framework designs obtained by the use of the Evolutionary Structural Optimization (ESO) was critically examined. It is often claimed that the method produces optimal structures because at each stage of evolution one discards the structurally most inefficient portions in a design. The consideration of weight alone does not provide an ideal metric for comparisons unless we fix the maximum stress in all the designs that are being compared to a prescribed value. We have, therefore, explored the two-objective problem of minimizing the weight and the maximum stress in a structure where comparisons can be readily made between the PF of various sets of designs.

The ESO designs were found, in many cases, to be dominated by those computed by an exhaustive search or a Genetic Algorithm (GA). Comparisons with the Pareto-optimal sets obtained from exhaustive search show that ESO does not always provide optimal solutions. However, it does produce some very good designs at a small computational expense. Discrete structures such as frames and trusses afford the opportunity to explore all possible designs—unlike a continuum model where such comparisons are not possible because of infinity of possible design options and because the conclusions are dependent on a specific parameterisation.

For complex topologies having a large number of joints, the number of design options cannot be exhaustively searched. Therefore, for such cases, multiple runs of GA has been used to produce Pareto-optimal design sets and compared with the designs produced by ESO. The conclusions from our numerical experiments remain

the same—ESO does not often produce Pareto-optimal designs; however, if one can afford only limited computational resources, then it produces some very good designs at relatively small cost.

Finally, the topology optimization problem was reformulated as one of gradually reducing the thickness in a range—so that a structural member is not altogether removed in one step, but is reduced only when the thickness approaches zero (or a prescribed lower threshold). The general observations for this case are consistent with the other numerical experiments presented in this chapter—ESO does not often produce Pareto-optimal solutions, however, it affords some very good designs inexpensively.

Chapter 5

Combining the Genetic Algorithm and Evolutionary Structural Optimization for the topology design of frameworks

5.1. Introduction

This chapter presents two new strategies for topology optimization of frameworks by combining the Evolutionary Structural Optimization (ESO) method and Genetic Algorithms (GA). Numerical experiments in the previous chapter suggested that ESO, when applied to frameworks, is computationally cheap but it misses out some of the structurally most efficient designs. On the other hand, the GA obtains structurally high quality designs but is expensive to run. Guided by these observations, two new strategies are proposed and implemented in this chapter in order to combine the quality of GA and the computational efficiency of ESO. In the first method called the ESO assisted GA method (ESOaGA), ESO obtained designs are inserted in the GA population, helping the GA search. The second method presented here is the GA assisted ESO method (GAaESO), in which GA produced designs are used as starting points for a family of ESO runs. The designs obtained by the two proposed methods, are compared with the designs obtained using the "unassisted" GA. Suggestions for multiple uses of ESO and GA are considered in order to accomplish more efficient results. Accordingly, the ESO-assisted-GA-assisted-ESO (ESOaGAaESO) method is further explored that combines ESOaGA and GAaESO concepts. Apart from the visual comparisons among the proposed methods, quantitative comparisons of the performances of the different methods using quality indicators are presented. Finally, a Kruskal-Wallis test is applied for each

comparison. Two design goals are used for this comparison: the maximum stress within the structure and the overall weight.

5.2. Observations on Evolutionary Structural Optimization of frameworks and Genetic Algorithms applied to framework design

It is often claimed that ESO tends to produce optimal structures because at each stage of evolution one discards the structurally most inefficient portions in a design. However, the consideration of weight alone does not provide an ideal metric for comparisons unless we fix the maximum stress in all the designs that are being compared to a prescribed value. Here, therefore, we have explored the two-objective problem of minimizing the weight and the maximum stress in a structure where comparisons can be readily made between the Pareto Fronts of various sets of designs. The ESO designs were found, in many cases, to be dominated by those computed by a Genetic Algorithm (GA). Comparisons with the Pareto-optimal sets obtained from the GA show that ESO does not always provide optimal solutions. However, it does produce some very good designs at a relatively small computational expense.

The possibility of combining the best features of the two algorithms is next explored to enhance the combined computational performance (or alternatively enhance the structural performance of designs given fixed computational resources). We propose two methods that combine the ability of the GA to search widely over the design space with the computational economy inherent in ESO.

The essential ingredient of the first method is to replace a part of the population of a GA by some promising designs obtained from ESO. In this way, the GA will look at more promising areas close to the ESO trajectory and below it, where the most optimum designs can be found. Overdoing this may not be a good idea because it may kill the production of novel designs from regions unexplored by ESO. By replacing some of the inefficient designs that the GA finds in the initial population or each of the populations one would expect to assist the GA in finding efficient designs economically.

The second method developed here is a GA assisted ESO method. In this approach, the GA produced designs for a given number of generations are selected and taken as starting points for a family of ESO runs. The GA is run as a two-

objective optimization problem. The set of designs on the PF as obtained by such a GA run are taken as starting points for a family of further ESO runs. The final solutions depend on the given initial design and thus different solutions to the same problem with the same discretization and optimization method can be obtained by using different starting designs. Applying ESO to each of the final GA designs obtained for a number of generations and populations expands the search for the globally best design. Note that the additional computational cost of running ESO several times is very marginal.

Both approaches are examined and compared with the "unassisted" GA method. The results of these comparisons show that these two new methods are able to produce efficient designs in a relatively smaller computer run time than the "unassisted" GA.

As the selection of the initial seed of GA can have an effect on the GA performance, every GA based method is run multiple times with different initial seeds so that a more representative output is obtained. All "assisted" and "unassisted" GA cases presented in this chapter are computed for 30 different randomly generated initial seeds. In order to compare 2 or more GA based methods, the same randomly generated set of initial seeds are considered for the corresponding GA based methods. The final output of a GA based method consists of the PF designs obtained from each of the 30 different initial seed cases. Instead of presenting only the PF and the least efficient GA designs for a GA search result, the area covered by the PF designs and least efficient designs produced by GA is presented and filled with colour (see shaded bands in Figures 5.1(ii) & 5.1(iv)). In this way, the solution space of GA based methods can be easily visualized especially when comparisons between various methods need to be done.

The designs obtained from all GA based optimization methods for the 6-node structural problem presented in this chapter are structures with 7 or less members removed from the ground structure. Designs with more than 7 members removed are discarded from the final outcome of the optimization method. We treat the maximum number of members that can be removed from the ground structure as a design constraint because otherwise the process is dominated by one or two trivial designs with only a few members in them (e.g. a simple beam), or their close variants.

The best designs, given two objectives, appear on the Pareto Front. Similarly, the worst designs produce another curve. The band between these two extremes

indicates the coverage of the designs explored. If we joined the left-most and right-most ends of the PF and the front of the worst of the explored designs, some of the existing designs may fall out of this band. To ensure that all the designs explored fall within the band, a top left corner and a bottom right corner are created. The top left corner is given by (min(x), max(y)) over the data and the bottom right corner is given by (max(x), min(y)) of the data (Figures 5.1(i) & 5.1(ii)). However, this definition of the band is not satisfactory when max(y) on PF is greater that max(y) on the worst design front, or max(x) on PF is greater that max(x) on the worst design front because the use of the strategy in Figures 5.1(iii) & 5.1(iv) will *not* produce corner points. We detect these situations and join the ends of the fronts directly (as shown in Figures 5.1(iii) & 5.1(iv)) to create the band.

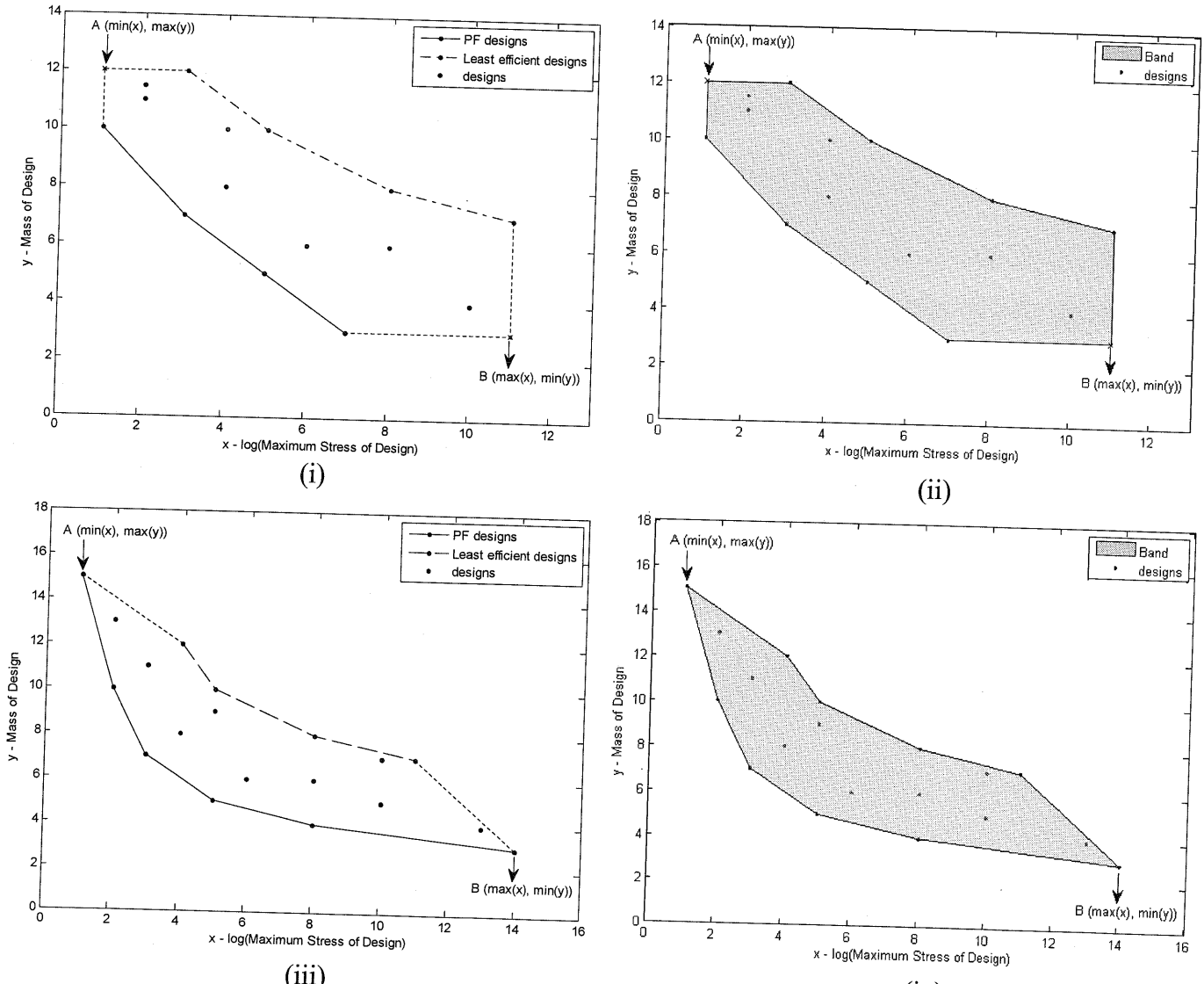


Figure 5.1: Definition of the design bands—(i) and (ii): when corner points need to be created to include all the designs within the band, (iii) and (iv): when joining the ends of the best and the worst fronts is satisfactory.

5.3. Performance assessment of multiobjective optimization methods using quality indicators

Visual comparisons of more than one two-objective optimization methods presented are often not reliable. An alternative approach is to compare and evaluate the performance of different optimization methods by the use of quality indicators. In order to compare the results among various cases of the same or different optimization methods, the unary quality indicators are used as discussed in the previous chapter. The design sets associated with each developed optimization method are first transformed into the representation format of the indicator values before a statistical testing method is applied. Three unary quality indicators: hypervolume, epsilon and R indicator are commonly used for the purpose of comparing the performance of methods. Each of these indicators measures the performance quantitatively with a slightly different approach.

Generally, many studies in the literature evaluate algorithms with respect to several different quality indicators. A combination of Pareto compliant indicators can yield interpretations that are more powerful than can be made by a single indicator. For example, if two Pareto compliant indicators contradict one another on the preference ordering of two design sets, then this implies that the two sets are incomparable. It is possible to make quantitative statements about the differences in quality even for incomparable design sets as unary quality indicators represent specific preference information. However, if several quality indicators are used, slightly different preferences are assessed by each of the indicators and this helps to build up a better picture of overall set quality of different sets of designs.

In the current work, for the comparison of the new developed hybrid optimization methods the quality indicators provided from the *Platform and programming language independent Interface for Search Algorithms (PISA)* [125] are used. PISA is a text-based interface for search algorithms used to test and compare new multi-objective optimization algorithms. It contains various packages of optimization algorithms and test and benchmark problems that can be used to assess the performance of different optimizers. However, in the current work, the performance assessment package of PISA was only used for the comparison and analysis of the developed optimization methods.

PISA initially runs all the combinations of optimization methods and test problems that are considered in the tool environment. After all the runs have been finished, a set of files is created, each one containing the design sets for all runs that have been generated by a particular optimizer-test problem pair after the same number of generations. Then, the hypervolume, epsilon and *R* indicators are applied to the resulting outcomes, obtaining the indicator values for the corresponding design sets of each optimizer-test problem pair. In our case, where we want to test the performance of new developed methods, only the comparing feature (quality indicators) of this software is used.

The design sets obtained from the developed methods under comparison are included in the same file of the directory in which the output of all the optimizer-test problem pairs is normally stored. During the execution of the program to compute the indicator values, PISA calculates the bounds that are needed for the tools to obtain the indicator values. The lower and upper bounds of the objective vectors are calculated. Initially, PISA calculates the worst value for each objective called "nadir point" (or upper bound) and correspondingly the best value of each objective called "ideal point" (or lower bound). Then, the nadir and utopia bounds for all the design sets that need comparison are calculated by the following relationships:

$$Nadir\ bound = nadir\ point + 0.1 \times (nadir\ point - ideal\ point), \tag{5.1}$$

$$Utopia\ bound = ideal\ point - 0.1 \times (nadir\ point - ideal\ point). \tag{5.2}$$

Based on the bounds determined above, PISA normalizes all data of the design sets transforming all objective values to the interval between 1 and 2, replacing the minimum value of the first objective for all sets as 1 and the maximum value of the second objective for all sets as 2. When using quality indicators, normalization can be necessary in order to allow the different objectives to contribute approximately equally to indicator values.

Moreover, PISA obtains the non-dominated set from all the normalized data, and uses this as a reference set according to which the indicator values are calculated. For example, if the non-dominated set of the normalized data is identical to one of the normalized design sets, then all indicator values for the corresponding design set are equal to zero.

In order to compare the performances of the GA-based optimization methods that are presented next, visual comparisons are accompanied with the corresponding tables of quantitative quality indicators. The data of a GA based method used to determine the corresponding values of the quality indicators, consists of *all the PF designs obtained from each of the 30 individual runs* (each with different random initial seed).

The complete PF of the exhaustive search, when up to 7 members are removed (see Figure 4.2b), is considered as a *fixed reference set* for the estimation of the quality indicators of all GA based methods for the 6-node example. Usually, the PF for a multiobjective problem is unknown and cannot be computed in a reasonable time. However, in our case the PF of the 6-node framework for up to 7 members removal has been exhaustively computed. The data sets of each method are compared separately with the exhaustive PF (reference set) providing fixed indicator values that are used when further comparisons are carried out among different cases or methods.

The exhaustive search PF for the 12-node example is not available. Therefore, when obtaining quality indicators for the 12-node example, we have used the overall non-dominated data as the reference. Because of the unavailability of a reference set (e.g. the exhaustive search PF), the values of the quality indicators are not fixed when the same data are compared with two (or more) different data sets for the 12-node case.

5.4. A Genetic Algorithm assisted by ESO (ESOaGA)

A method that combines the efficiency of GA and the small computational time of ESO by including ESO produced designs in the GA population is presented first. Enrichment of the population by cheaply obtained designs using ESO gives the GA a chance to incorporate good features in future generations. However, since we know that ESO does not produce optimum designs globally, we need to keep the genetic diversity in the population alive and, therefore, not be tempted to dominate the population with ESO designs. If we use too many ESO designs, the result would be the GA returning designs that perform very similarly to those obtained from ESO. Using a judicious balance, it is hoped to combine the good features of the two

algorithms — (i) ESO being computationally cheap, and (ii) GA being able to find novel designs by means of genetic operations that ESO cannot 'reach'. The results from the "unassisted" GA and the ESO assisted GA will be compared in order to examine the performance of this new method. The flow chart for this approach is shown in Figure 5.2.

First, a sequence of designs is generated by the ESO process. The connectivities of these designs are then transformed to the bit-string representation of the topology. The members that have been removed in the connectivities obtained by ESO are coded as "0" and the members that have remained are coded as '1'. When the first generation of GA runs, a Pareto sorting program is used to find the current PF created by the designs obtained by the first population of the GA. The designs that are worse than the current PF are detected and replaced by ESO designs. The new modified population is saved and the next generation is produced following the normal procedures. The population of the following generation will thus be based on the modified population.

Different variants of this method will be presented next. Examples are restricted to framework structures having 6 or 12 key points. In the first instance, the addition of ESO designs takes place only in the initial population. Then the ESO designs replace a prescribed number of poor designs generated by the GA in each generation. In addition, the ESO assisted GA method is tested for a variety of population and generation numbers and compared with the corresponding "unassisted" GA method.

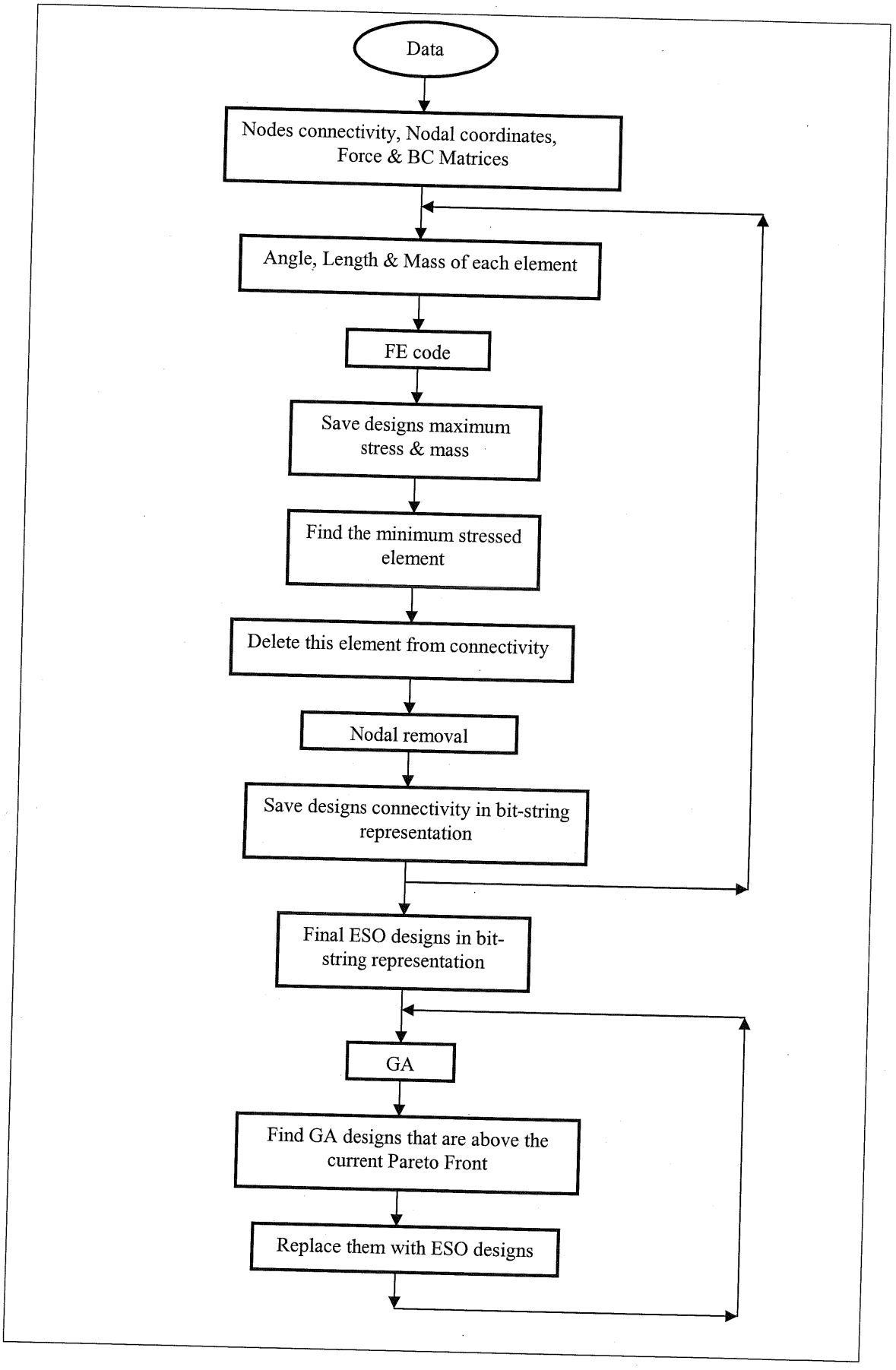


Figure 5.2: Flow chart of ESO assisted GA (ESOaGA) method.

5.4.1. ESO assisted GA, including ESO designs *in initial population* only

The 6-node structure

In this section we present an example of framework design that has six key points as shown in Figure 5.3. Each key point is connected to every other key point. When members are eliminated from this fully-connected configuration by the use of an ESO algorithm, there are 9 possible designs in the search trajectory (Figure 5.4). All nine ESO designs are then included in the initial population of the GA. The ESO assisted GA is run for 20 generations with 20 members in each population. Nearly half of the designs in the initial population are replaced by ESO designs. In Figure 5.5, the ESO assisted GA designs are compared with the designs produced by the "unassisted" GA. As discussed before, the member removal process stops at a maximum of 7 members removed—this is to exclude some trivial designs that otherwise dominate the quality.

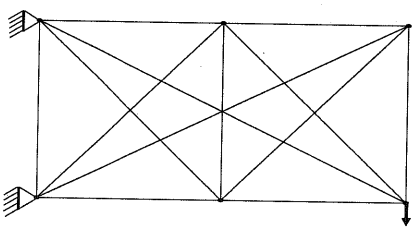


Figure 5.3: Fully connected framework with 6 nodes.

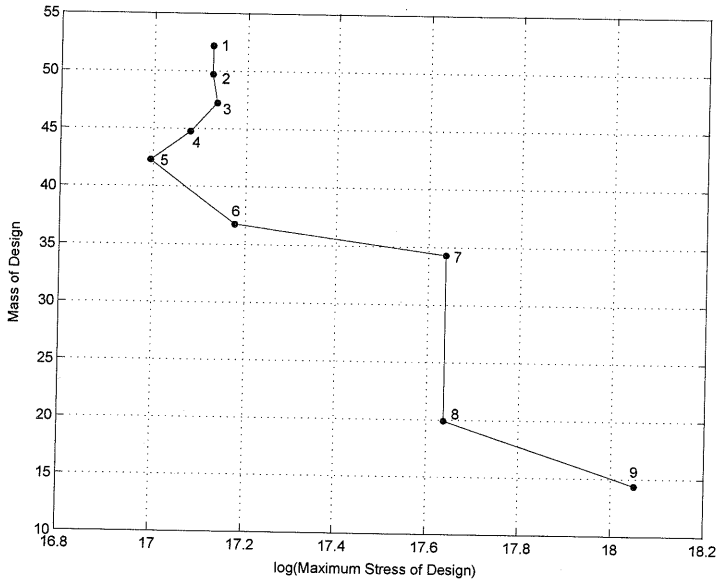


Figure 5.4: ESO trajectory of a 6-node structure.

The "unassisted" GA, ESOaGA and the complete PF of the exhaustive search are compared in Figure 5.5. There is a general overlap. It is not visually obvious if one of the two methods clearly produces designs closer to the PF. However, ESOaGA does find some efficient designs in the low maximum stress end of the solution space. It also appears to avoid a significant number of poor designs that the "unassisted" GA obtains. On the other hand, the "unassisted" GA also has some superior designs missed out by ESOaGA—hence the need for a quantitative comparison. Note that *all* the ESO designs were included in assisting ESOaGA in this example. This has a negative effect in the proper functioning of the GA because it reduces the diversity of the population. It will be shown later that including only a few ESO designs in assisting GA is most profitable.

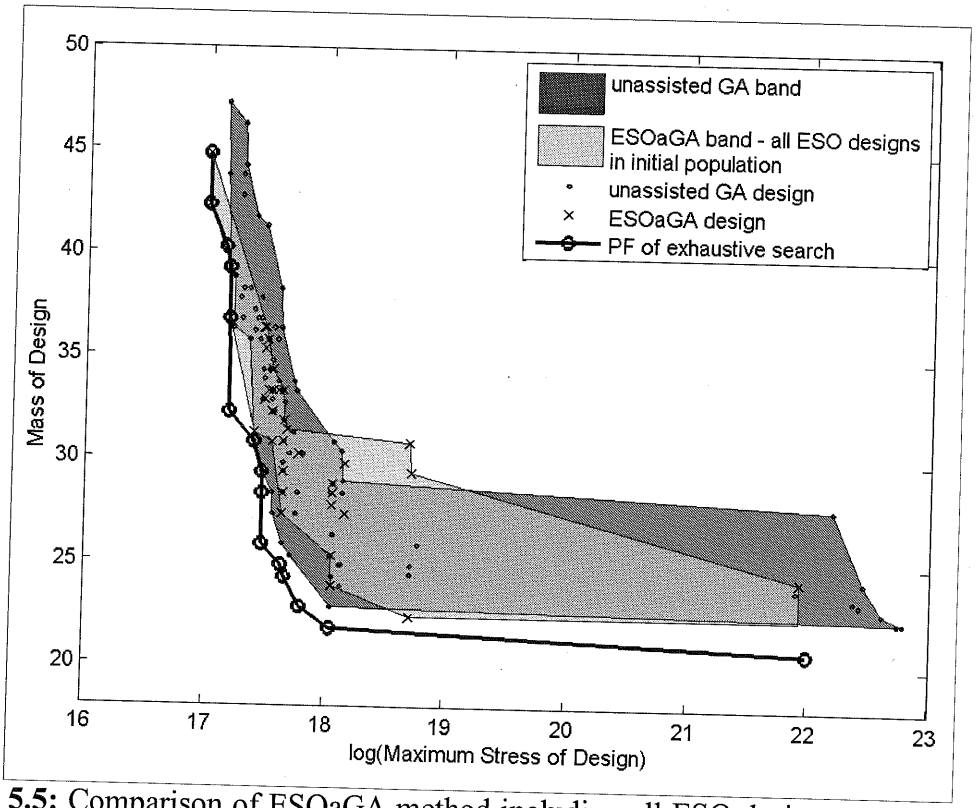


Figure 5.5: Comparison of ESOaGA method including all ESO designs *in the initial population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

No robust conclusions can be made about which method outperforms the other visually. Therefore, further comparison is achieved using the unary quality indicators of PISA's performance assessment tools, so that more clear decisions can be made on the performance assessment of the two methods. The pair of the design sets coming from the two methods is compared using the hypervolume, epsilon and R indicators.

For each of these three indicators, each design set is assigned a single indicator value and the one with the smaller value corresponds to a better design set.

For a better representation of the results of the quality indicators, the indicator values of the corresponding design sets are assigned ranking numbers from 1 to 5, with rank 1 to represent the best design set (lowest indicator value) and rank 5 the worst design set (highest indicator value) respectively. Each ranking number is assigned a different colour, with the blue colour becoming darker as the rank number reduces from 5 to 1 (Table 5.1). This colour scheme is used in all tables presenting the results of quality indicators (unless stated otherwise).

Accordingly, from Table 5.2, we can conclude that ESOaGA is better than "unassisted" GA in two of the three quality indicators. When ESO designs are included in the initial population, the performance of the designs on the two objectives is better in more aspects than those produced by the "unassisted" GA.

Rank 1
Rank 2
Rank 3
Rank 4
Rank 5

Table 5.1: Ranking colour scheme used in the following tables unless stated otherwise. Rank 1 refers to the best design set.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
Unassisted GA	6.54E-02	7.73E-02	3.26E-02
ESOaGA - all ESO designs in initial population	6.38E-02	8.53E-02	2.43E-02

Table 5.2: Comparison between "unassisted" GA and ESOaGA including all ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

In order to improve the efficiency of ESOaGA, fewer ESO designs inserted to GA initial population is considered next. Consider inserting only one ESO design in the initial population first. A question that arises next is which of the nine ESO designs should be included in the population. All cases of including each time a different ESO design are considered. The design sets of all possible cases of including

ESO assisted GA method best. For example, when two ESO designs replace two poor GA designs, we have to consider which ones to choose from the total of nine ESO designs available. When ESO designs are included in the initial GA population, the results vary according to the choice of ESO designs that has been made. In Figure 5.7, we notice that different pairs of ESO designs lead to slightly different results. As can be seen from the figure, the best pair of ESO designs to use is the fifth and sixth, which are the ones that in the middle of the ESO trajectory (see Figure 5.4). As we can see from Table 5.3, the pair of the fifth and sixth ESO designs obtains the highest rankings in two of the three quality indicators.

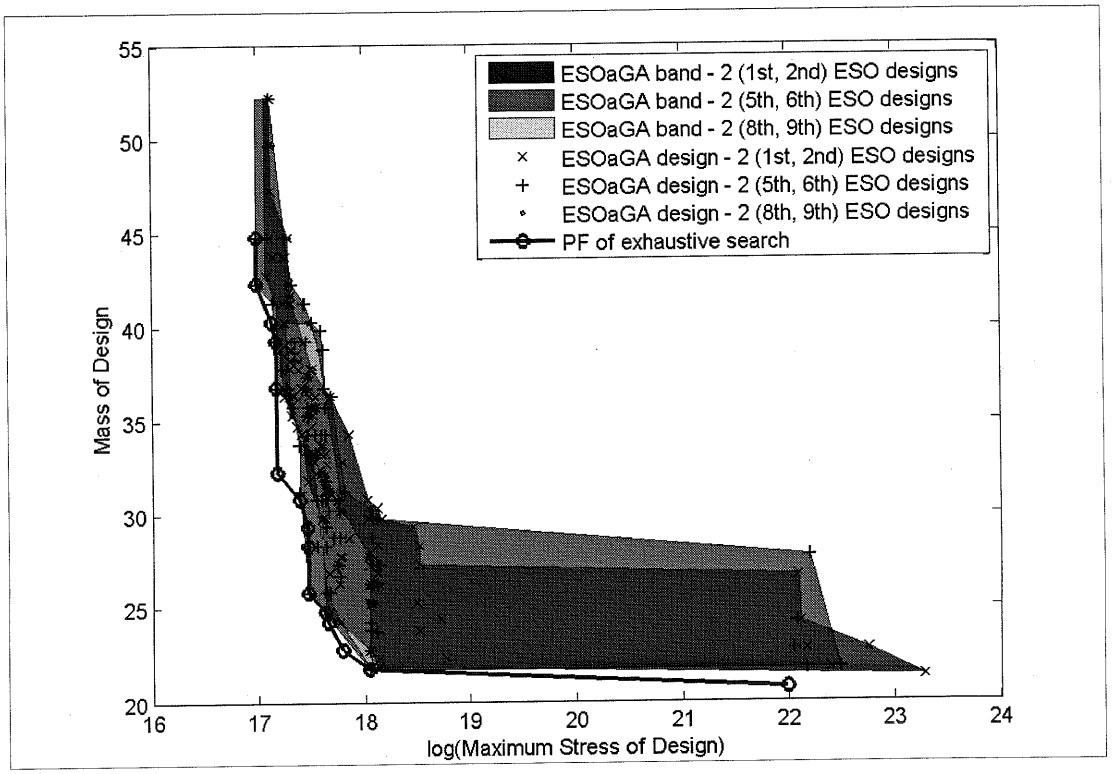


Figure 5.7: Comparison among cases of ESOaGA including different pairs of ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (1^{st}, 2^{nd}) ESO designs in initial population	3.18E-02	4.53E-02	1.28E-02
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in initial population	1.52E-02	5.06E-02	6.15E-03
ESOaGA - 2 (8^{th}, 9^{th}) ESO designs in initial population	4.86E-02	6.61E-02	3.61E-02

Table 5.3: Comparison among various cases of ESOaGA including 2 ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

Another example that supports the fact that different sets of ESO designs can give rather different results is presented. In Figure 5.8, it can be observed that including the fifth, sixth and seventh ESO designs in the initial population of the GA, gives the best results, compared with two other sets of designs. This particular set of ESO designs obtains the highest rankings in two of the three quality indicators as shown in Table 5.4.

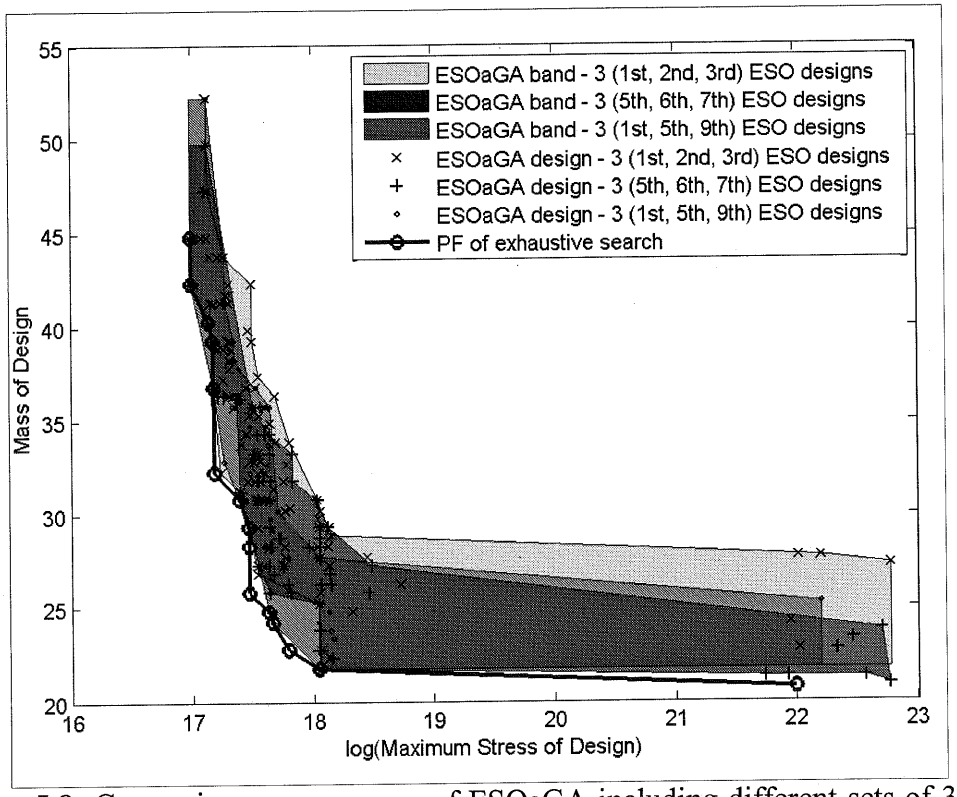


Figure 5.8: Comparison among cases of ESOaGA including different sets of 3 ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 3 (1^{st}, 2^{nd}, 3^{rd}) ESO designs in initial population	1.95E-02	6.63E-02	1.15E-02
ESOaGA - 3 (1^{st}, 5^{th}, 9^{th}) ESO designs in initial population	1.86E-02	5.55E-02	2.25E-03
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02

Table 5.4: Comparison among various cases of ESOaGA including 3 ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

As a general rule, the very first few designs in the ESO run are likely to be poor quality. On the other hand, the use of ESO designs which are at the middle of the

ESO trajectory, tend to produce better results. Therefore, it is recommended that designs *excluding* the first initial ones of the ESO run should be kept as part of the initial GA population.

The performance of the ESOaGA does not only depend on the quality of designs selected and the stochastic nature of the GA but on the number of ESO generated designs added to the population. It is not simply the case that including the highest quality of ESO generated designs will always improve the performance. It depends also on the number of the ESO designs that need to be inserted.

To study this, just one design was included in the population and then all the nine ESO designs were included in a GA population. The resulting designs obtained by GA are shown in Figure 5.9. It can be observed that when only one ESO design is included in the initial population, the resultant designs dominate most of the designs produced by GA when all ESO designs are included in the initial population. The quality indicator values presented in Table 5.5 validates that including one ESO design in the initial population of GA yields better results than including all ESO designs in the initial population.

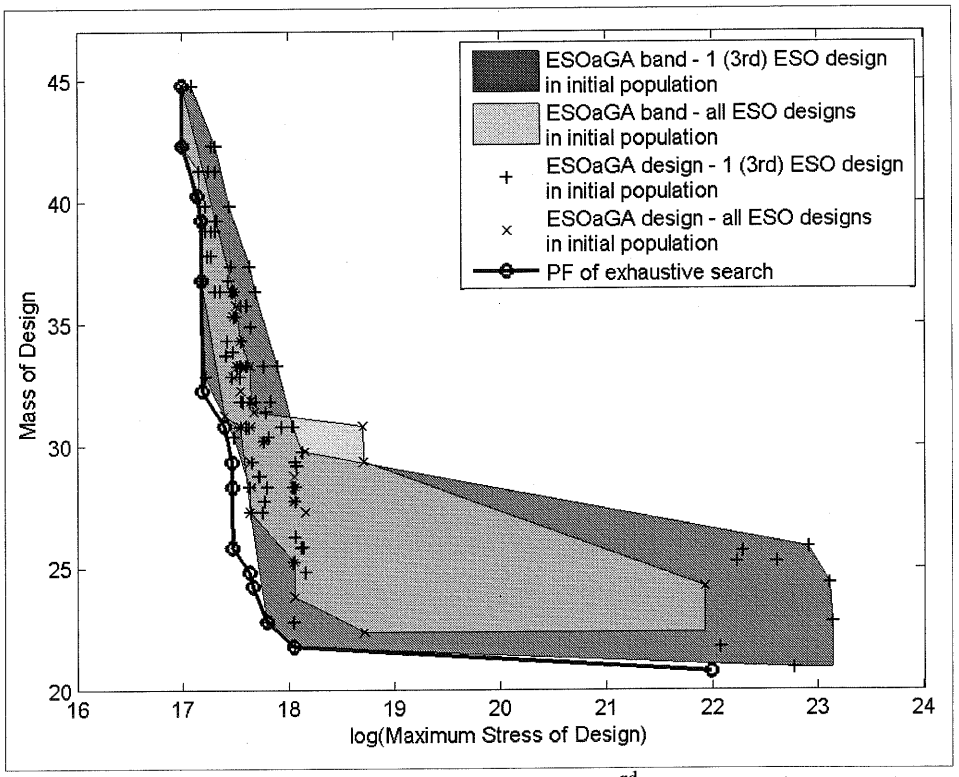


Figure 5.9: Comparison of ESOaGA including 1 (3rd) ESO design *in the initial population* and ESOaGA including all 9 ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3rd) ESO design in initial population	1.81E-02	5.25E-02	2.71E-03
ESOaGA - all ESO designs in initial population	6.38E-02	8.53E-02	2.43E-02

Table 5.5: Comparison of ESOaGA including 1 (3rd) and all ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

Adding as many ESO designs in the initial population of GA as possible, does not mean that this will help the topology optimization process. As we can see from Figures 5.10 & 5.11, increasing the number of ESO designs in the initial population does not improve the optimization process beyond a point, as it depends on the choice of the sets of ESO designs included in the population. The number of inserted ESO designs is not solely responsible for the enhancement of the quality of the results. The quality of the selected designs matters as well. This is proved by comparing ESOaGA cases with different sets of ESO designs (each set consists of a different ESO design selection) inserted into the initial population. In Figure 5.10, it can be clearly seen that the case of inserting two ESO designs is a better option than inserting one or three ESO designs. The designs obtained by ESOaGA with two ESO designs in the initial population covering the red area in Figure 5.10 dominate most of the designs obtained by the other two cases.

In Table 5.6, comparing the indicator values for the best case of including one ESO design in the initial population and the best case of including two ESO designs in the initial population, we observe that the case of including two ESO designs in the initial population is obtaining higher rankings than the case of including only one ESO design. This means that the amount of ESO designs that can be included in the population can affect the results of ESOaGA method. However, the case of including three ESO designs in the initial population, does not obtain better results than the case of inserting two ESO designs. Note that in this comparison we used the case of including three (1st, 2nd, 3rd) ESO designs in the initial population (see Table 5.4), to prove that increasing the number of inserted ESO designs does not necessarily lead to better results.

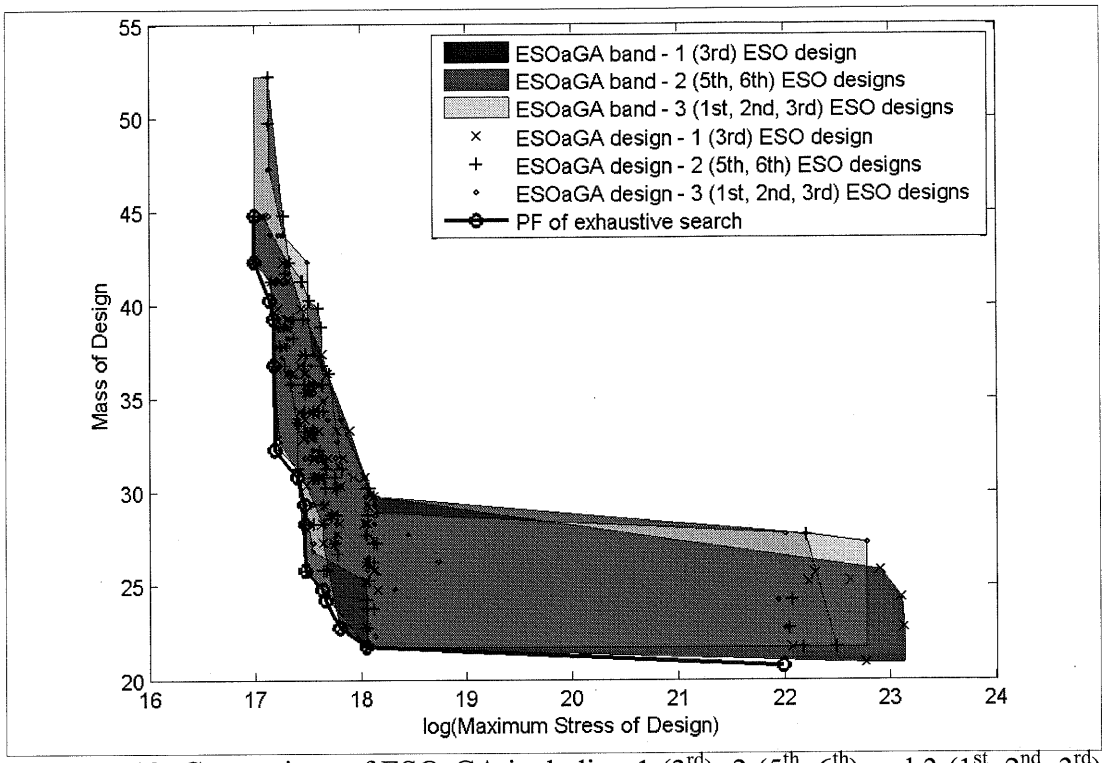


Figure 5.10: Comparison of ESOaGA including 1 (3rd), 2 (5th, 6th) and 3 (1st, 2nd, 3rd) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3rd) ESO design in initial population	1.81E-02	5.25E-02	2.71E-03
ESOaGA - 2 (5th, 6th) ESO designs in initial population	1.52E-02	5.06E-02	6.15E-03
ESOaGA - 3 (1st, 2nd, 3rd) ESO designs in initial population	1.95E-02	6.63E-02	1.15E-02

Table 5.6: Comparison of ESOaGA including 1 (3rd), 2 (5th, 6th) and 3 (1st, 2nd, 3rd) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

When a particular set of 3 ESO designs is included in the initial population, it may produce better results than using 5 ESO designs in the population, although the same three ESO designs are included in the set of five ESO designs (Figure 5.11). From both visual comparison and quantitative comparison (Table 5.7), we conclude that the case of including three particular ESO designs is better than the case of including one ESO design and five particular ESO designs.

In Table 5.8, the cases of including one, two, three, five and all (nine) ESO designs are compared for 20 generations and 20 designs in each population. Inserting just three ESO designs leads to higher ranking than the rest of the cases after the

interpretation of the quality values. Including the 5th, 6th and 7th designs from the ESO trajectory of Figure 5.4 turns out to be the most efficient strategy (including two extra ESO generated designs in the GA run actually worsens the performance, Figure 5.11).

When we compare the same cases but for 40 generations and 20 designs in each population (Table 5.9), we observe that including two ESO designs obtains the highest rankings as far as the quality indicators are concerned. When running the particular problem with 20 generations, it is more effective to include a few ESO designs (three) instead of two, because the small number of generations does not allow GA to search efficiently. When the number of generations is increased to 40, fewer ESO designs need to be inserted as GA runs more efficiently and it has more time to explore the solution space. Providing less assistance to GA is a better option than providing more assistance (beyond a certain point) as the computational time increases. Including many ESO designs reduces the diversity of the GA, as the number of generations is increased.

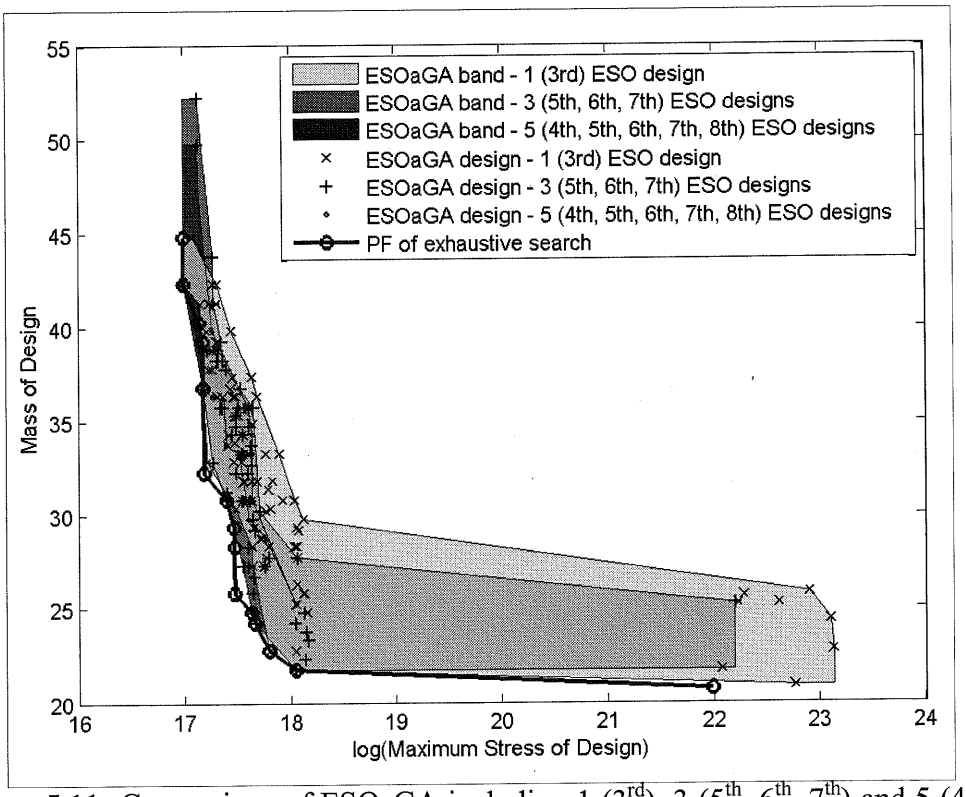


Figure 5.11: Comparison of ESOaGA including 1 (3rd), 3 (5th, 6th, 7th) and 5 (4th, 5th, 6th, 7th, 8th) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3rd) ESO design in initial population	1.81E-02	5.25E-02	2.71E-03
ESOaGA - 3 (5th, 6th, 7th) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02
ESOaGA - 5 (4th, 5th, 6th, 7th, 8th) ESO designs in initial population	1.43E-01	1.56E-01	5.67E-02

Table 5.7: Comparison of ESOaGA including 1 (3rd), 3 (5th, 6th, 7th) and 5 (4th, 5th, 6th, 7th, 8th) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3rd) ESO design in initial population	1.81E-02	5.25E-02	2.71E-03
ESOaGA - 2 (5th, 6th) ESO designs in initial population	1.52E-02	5.06E-02	6.15E-03
ESOaGA - 3 (5th, 6th, 7th) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02
ESOaGA - 5 (4th, 5th, 6th, 7th, 8th) ESO designs in initial population	1.43E-01	1.56E-01	5.67E-02
ESOaGA - all ESO designs in initial population	6.38E-02	8.53E-02	2.43E-02

Table 5.8: Comparison of ESOaGA including 1 (3rd), 2 (5th, 6th), 3 (5th, 6th, 7th), 5 (4th, 5th, 6th, 7th, 8th) and all ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3rd) ESO design in initial population	1.92E-02	4.36E-02	1.52E-02
ESOaGA - 2 (5th, 6th) ESO designs in initial population	1.17E-02	4.45E-02	6.16E-03
ESOaGA - 3 (5th, 6th, 7th) ESO designs in initial population	1.15E-02	4.57E-02	6.21E-03
ESOaGA - 5 (4th, 5th, 6th, 7th, 8th) ESO designs in initial population	1.74E-01	2.06E-01	7.43E-02
ESOaGA - all ESO designs in initial population	5.98E-02	7.73E-02	1.60E-02

Table 5.9: Comparison of ESOaGA including 1 (3rd), 2 (5th, 6th), 3 (5th, 6th, 7th), 5 (4th, 5th, 6th, 7th, 8th) and all ESO designs *in the initial population* for a 6-node structure (40 generations and 20 designs in each population).

A visual comparison between ESOaGA and "unassisted" GA is easier in Figure 5.12 where the ESOaGA case of including 3 (5th, 6th, 7th) ESO designs (best set selection of three ESO designs) is compared with the "unassisted" GA for 20 generations and 20 designs in each population. In contradiction with the comparison

of the "unassisted" GA and ESOaGA with all ESO designs included in the population (Figure 5.5), this time ESOaGA dominates entirely the objective space of the "unassisted" GA. We can also observe that this ESOaGA case finds a lot more complete PF designs than the ESOaGA case of including all ESO designs in the initial population (Figure 5.5). A further comparison is provided from the quality indicators results in Table 5.10. The ESOaGA has higher rankings in all three quality indicators as expected. It is clear in this particular example that ESOaGA shows larger improvement on the results when 3 ESO designs are inserted to GA rather than all ESO designs (Table 5.2). The performance of ESOaGA is still efficient when a larger number of generations is considered. ESOaGA still obtains better designs than "unassisted" GA when the generation size of GA is doubled (Figure 5.13).

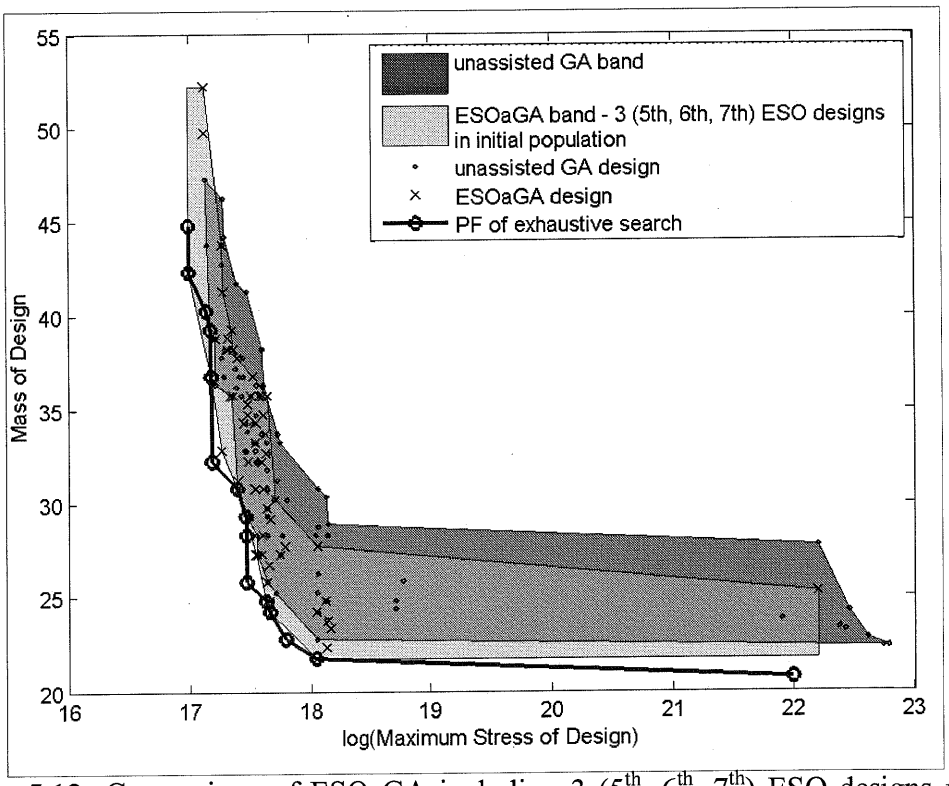


Figure 5.12: Comparison of ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in the initial population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
Unassisted GA	6.54E-02	7.73E-02	3.26E-02
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02

Table 5.10: Comparison between ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in the initial population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population).

In Figure 5.13, the best ESOaGA case obtained from Table 5.9 is compared with the results of the "unassisted" GA for 40 generations and 20 designs in each population. ESOaGA obtains a majority of the complete PF designs and it is slightly superior in most parts of the "unassisted" GA's solution space. The output provided by the quality indicators agrees with the above observation. ESOaGA obtains higher rankings than "unassisted" GA in hypervolume and *R* indicators. As noticed in the previous case, the ESOaGA finds less poor designs than the "unassisted" GA as most of the least efficient designs of ESOaGA dominate the corresponding designs of plain GA.

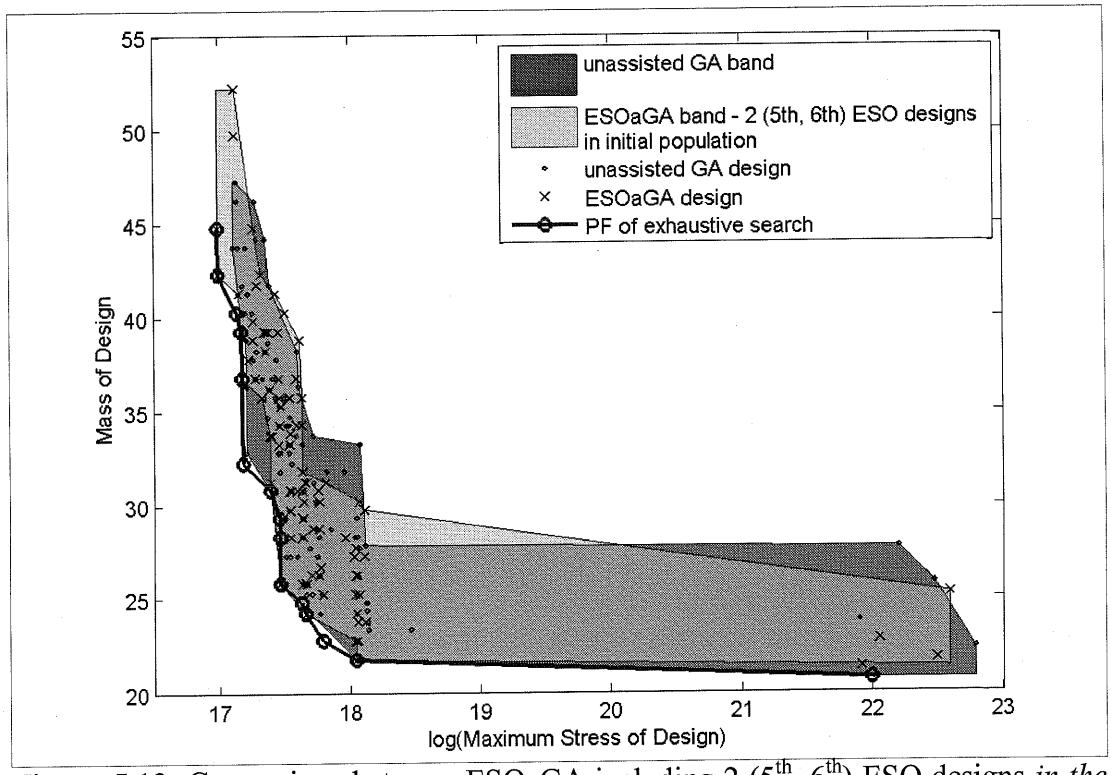


Figure 5.13: Comparison between ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in initial population	1.17E-02	4.45E-02	6.16E-03
Unassisted GA	4.36E-02	1.23E-02	7.69E-03

Table 5.11: Comparison between ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population).

The 12-node structure

A computationally more demanding problem is considered next (Figure 5.14). The ground structure has 12 key-points now and the total number of designs on the ESO trajectory is 52. Various ESO assisted GA runs for 20 generations with a population of 20 produce clearly better results than the "unassisted" GA for the same size of population and number of generations as shown in Table 5.12. In particular, the indicator values of the ESOaGA cases of including 5, 10, and 20 ESO designs correspondingly in the initial population have higher rankings than the "unassisted" GA ones. This fact shows once more the positive effect ESO generated designs have in enriching the population for GA runs. We also observe that when 10 ESO designs out of the total 52 are included in the initial population, the results are better than when five or twenty ESO designs are included every time (Table 5.12). The number of ESO designs needed to be inserted to the GA population increases when the structural problem increases. In the 6-node example, an effective approach was to include 3 particular ESO designs in the initial population. However, in the 12-node case, a larger number of ESO designs need to be inserted in order to obtain efficient results. As the structural size is increased, the solution space for GA is also increased and more assistance coming from the ESO designs is profitable.

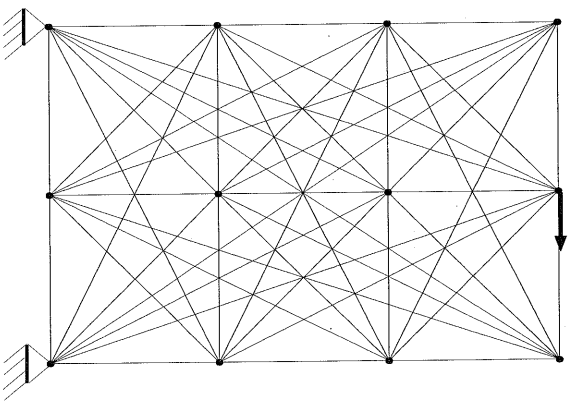


Figure 5.14: Fully connected framework with 12 nodes.

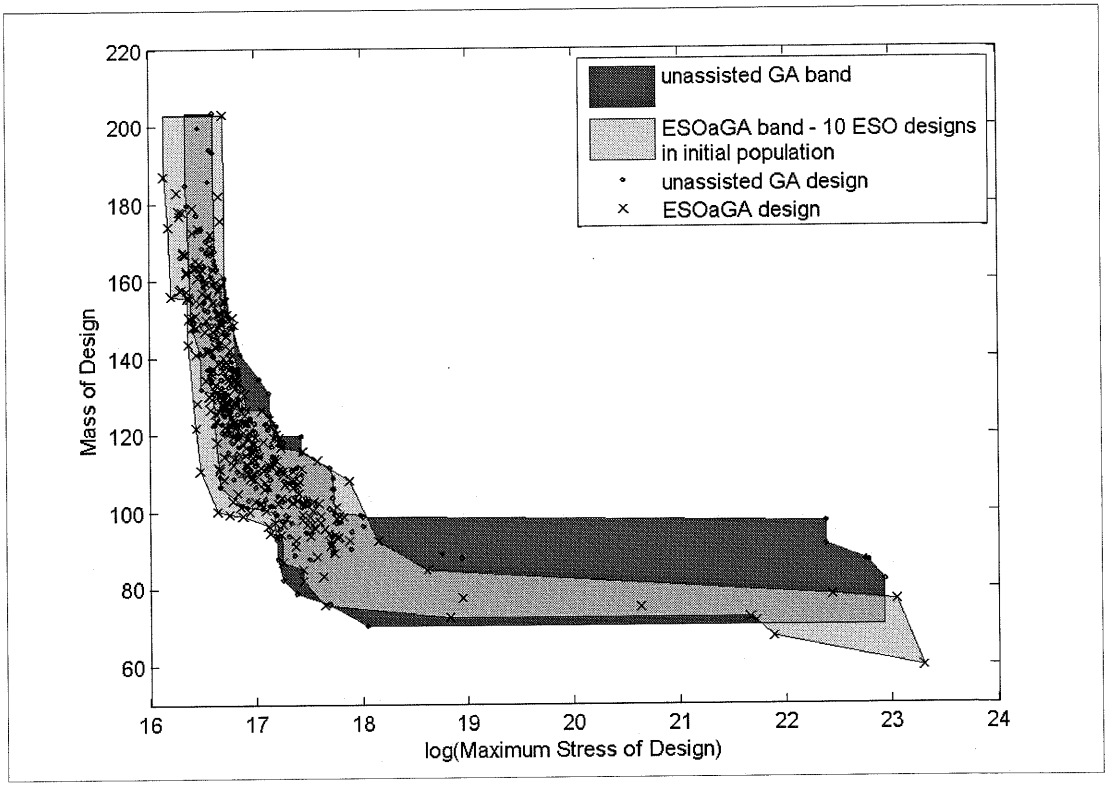


Figure 5.15: Comparison between ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* and "unassisted" GA for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 5 ESO designs in initial population	3.65E-02	4.27E-02	3.12E-02
ESOaGA - 10 ESO designs in initial population	2.48E-02	5.12E-02	2.47E-03
ESOaGA - 20 ESO designs in initial population	5.51E-02	1.12E-01	4.07E-02
Unassisted GA	4.18E-02	8.44E-02	4.38E-02

Table 5.12: Comparison of ESOaGA including 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}), 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}), 20 (1^{st}, 5^{th}, 7^{th}, 10^{th}, 12^{th}, 15^{th}, 17^{th}, 20^{th}, 22^{nd}, 25^{th}, 27^{th}, 30^{th}, 32^{nd}, 35^{th}, 37^{th}, 40^{th}, 42^{nd}, 45^{th}, 47^{th}, 50^{th}) ESO designs *in the initial population* and "unassisted" GA for a 12-node structure (20 generations and 20 designs in each population).

ESOaGA cases with less than 10 ESO designs included in the initial population are considered next in order to examine the effect of the number of the inserted ESO designs in the initial population for the 12-node structure. The cases of including 1, 2, 5 and 10 ESO designs in the initial population are presented in Figure 5.16. These four ESOaGA cases cannot easily be compared by visual inspection and, therefore, the comparison is assessed using the quality indicators. Since visual

comparisons become difficult when overlaying more than three bands in one figure, only quantitative comparisons will be presented.

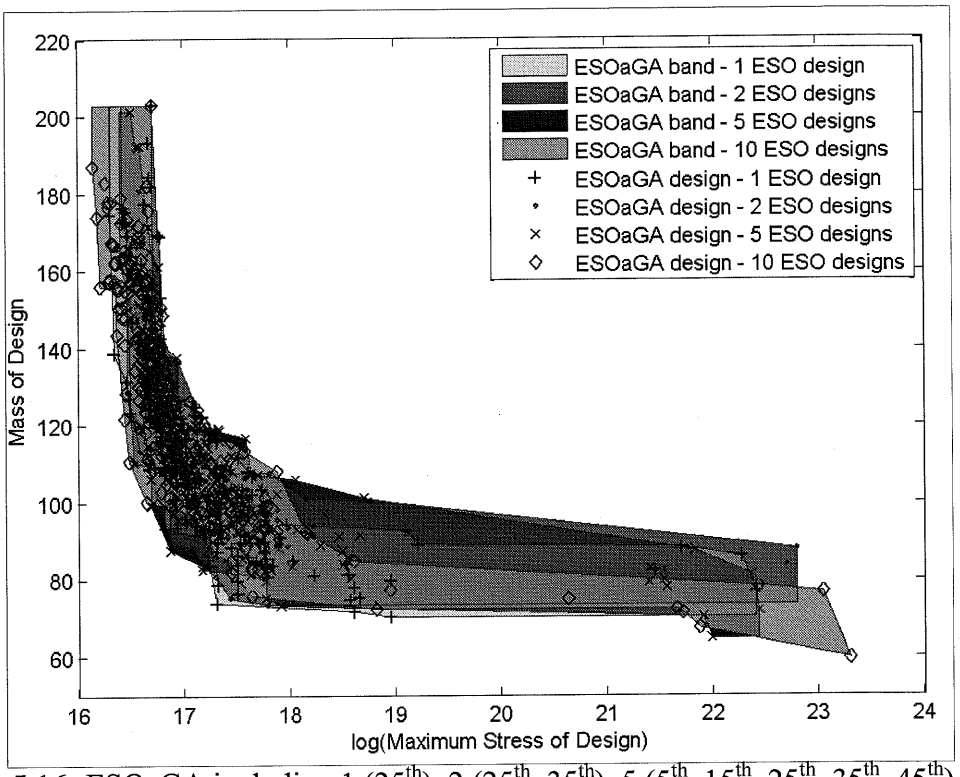


Figure 5.16: ESOaGA including 1 (25^{th}), 2 (25^{th}, 35^{th}), 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}) and 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (20 generations and 20 designs in each population).

All the cases with less than 10 inserted ESO designs obtain lower indicator values than the case of the 10 inserted ESO designs (Table 5.13). This continues to be so even when the number of generations is increased from 20 to 40 (keeping the population size to 20, see Table 5.14). In order to get the best results from the ESOaGA method including ESO designs in the initial population, the right balance between the inserted ESO designs and GA designs in the initial population needs to be found. In this particular case, inserting 10 ESO designs in the initial population provides very good results compared to other ESOaGA cases and most importantly to the "unassisted" GA.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 ESO design in initial population	3.18E-02	8.50E-02	3.96E-02
ESOaGA - 2 ESO designs in initial population	5.67E-02	1.08E-01	5.45E-02
ESOaGA - 5 ESO designs in initial population	3.52E-02	5.85E-02	2.88E-02
ESOaGA - 10 ESO designs in initial population	2.34E-02	5.15E-02	0.00E+00

Table 5.13: Comparison of ESOaGA including 1 (25^{th}), 2 (25^{th}, 35^{th}), 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}) and 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 ESO design in initial population	5.38E-02	8.79E-02	3.93E-02
ESOaGA - 2 ESO designs in initial population	8.66E-02	9.24E-02	4.93E-02
ESOaGA - 5 ESO designs in initial population	4.93E-02	7.95E-02	4.29E-02
ESOaGA - 10 ESO designs in initial population	6.41E-03	4.61E-02	0.00E+00

Table 5.14: Comparison of ESOaGA including 1 (25^{th}), 2 (25^{th}, 35^{th}), 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}) and 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (40 generations and 20 designs in each population).

Tables 5.15-5.17 illustrate the quantitative comparisons between the ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs in the initial population and "unassisted" GA (40 generations and 20 designs in each population), for each case of 30 different initial seeds. In the following tables, the darker blue colour corresponds to the method with the higher ranking (Rank 1) and the lighter blue colour to the method of lower ranking (Rank2). We can observe that ESOaGA obtains higher rankings than "unassisted" GA for the most of the different seed cases. Particularly, ESOaGA achieves higher rankings in almost 2/3 of the total seed cases. Occasionally, we find that GA is superior because of its stochastic nature.

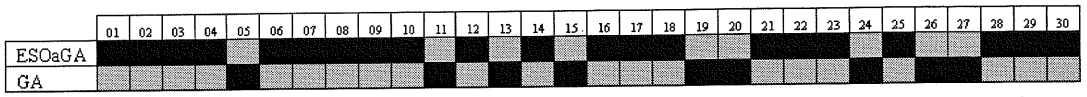


Table 5.15: Comparison of ESOaGA including ESO designs *in the initial population* and "unassisted" GA for each initial seed case with respect to hypervolume indicator.

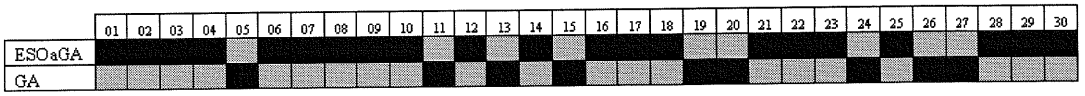


Table 5.16: Comparison of ESOaGA including ESO designs *in the initial population* and "unassisted" GA for each initial seed case with respect to epsilon indicator.

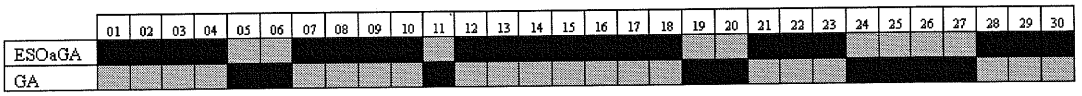


Table 5.17: Comparison of ESOaGA including ESO designs *in the initial population* and "unassisted" GA for each initial seed case with respect to *R* indicator.

5.4.2. ESO assisted GA, including ESO designs *in each population*

The 6-node structure

ESO designs are included in each population of the GA next. In all the following examples, the same ESO designs are selected and inserted into each GA population. Initially, the ESOaGA method is applied to the 6-node structural example. The "unassisted" GA has a very good performance for 40 generations and 20 designs in each population as it reaches most of the exhaustive PF's designs. However, inserting all ESO designs in each population improves the performance of GA even more but in a restricted region of GA's solution space (Figure 5.17). Inserting ESO designs in each population influences negatively the diversity of GA search. In this case ESOaGA keeps GA focused in a particular region missing in this way a lot of designs that the "unassisted" GA obtains. The results seen in Figure 5.17 show the improvement in the efficiency of GA when ESO designs replace inefficient designs in the population of GA only in a limited area. Despite the limited spread due to a lack of diversity, ESOaGA does successfully exclude many poor designs.

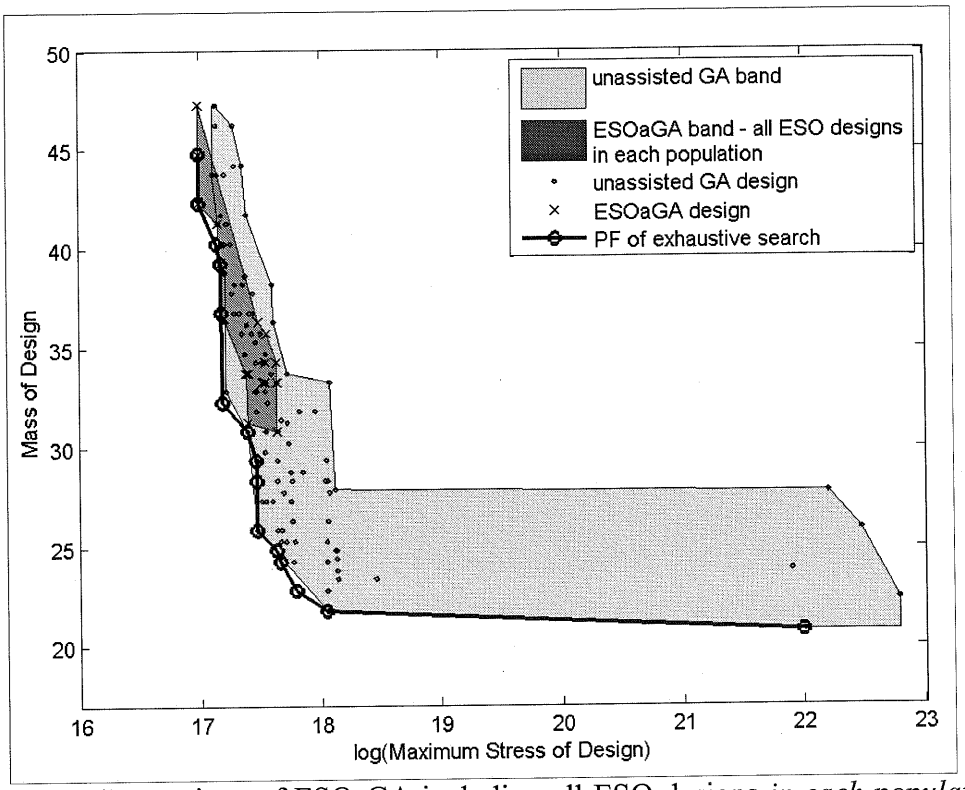


Figure 5.17: Comparison of ESOaGA including all ESO designs *in each population* and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

Table 5.18 shows that putting all the ESO designs in each population does not provide overall better results than the "unassisted" GA. The hypervolume, epsilon and *R* indicator values presented in Table 5.18, indicate that the "unassisted" GA outperforms the ESOaGA.

As the 6-node framework is a relatively small structural problem, a large number of generations increases the performance of "unassisted" GA, leaving no space for much further improvement. The output designs of plain GA match with most of the PF designs obtained from the exhaustive search. Inserting all ESO designs in each GA population obtains some even better results in the regions only that "unassisted" GA failed to obtain.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - all ESO designs in each population	6.61E-02	3.95E-02	2.04E-02
Unassisted GA	4.36E-02	1.23E-02	7.69E-03

Table 5.18: Comparison of ESOaGA including all ESO designs *in each population* and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population).

The case of including all ESO designs in each GA population is compared with the case of including fewer ESO designs (5^{th} & 6^{th}) in Figure 5.18. From visual comparison as well as quality indicators (Table 5.19), we observe an improvement in ESOaGA's performance when the number of ESO designs inserted in GA runs is reduced. The measurements of all three quality indicators show that the case of including these two particular ESO designs in each population is a better option than including all.

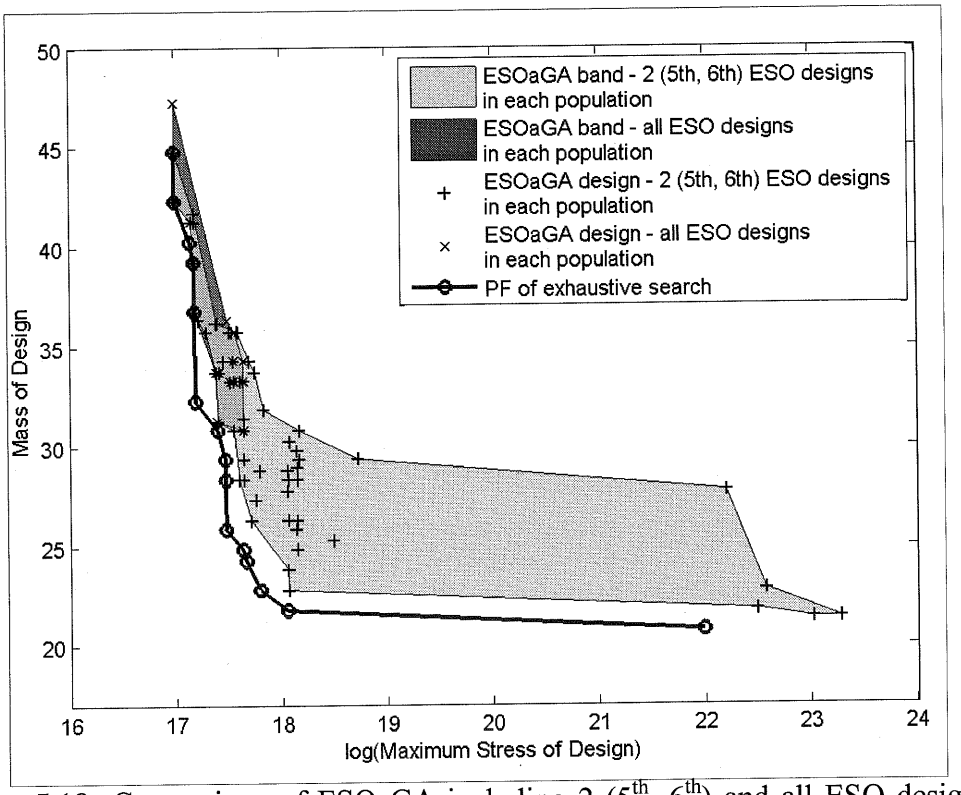


Figure 5.18: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) and all ESO designs *in each population* for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in each population	6.07E-02	7.92E-02	9.63E-03
ESOaGA - all ESO designs in each population	6.61E-02	3.95E-02	2.04E-02

Table 5.19: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) and all ESO designs *in each population* for a 6-node structure (40 generations and 20 designs in each population).

The ESOaGA results are superior to the "unassisted" GA only in a limited area on the left side of Figure 5.19 when just two ESO designs (5^{th} & 6^{th}) are included in

GA runs. However, ESOaGA still does not dominate entirely the solution space of "unassisted" GA. The results taken from PISA's quality indicators agree with this observation (Table 5.20). All the quality indicators give higher ranking to "unassisted" GA. This is in contrast to the situations (a) when GA designs are inserted in the initial population only, (b) when the structure is more complex (to be taken up later).

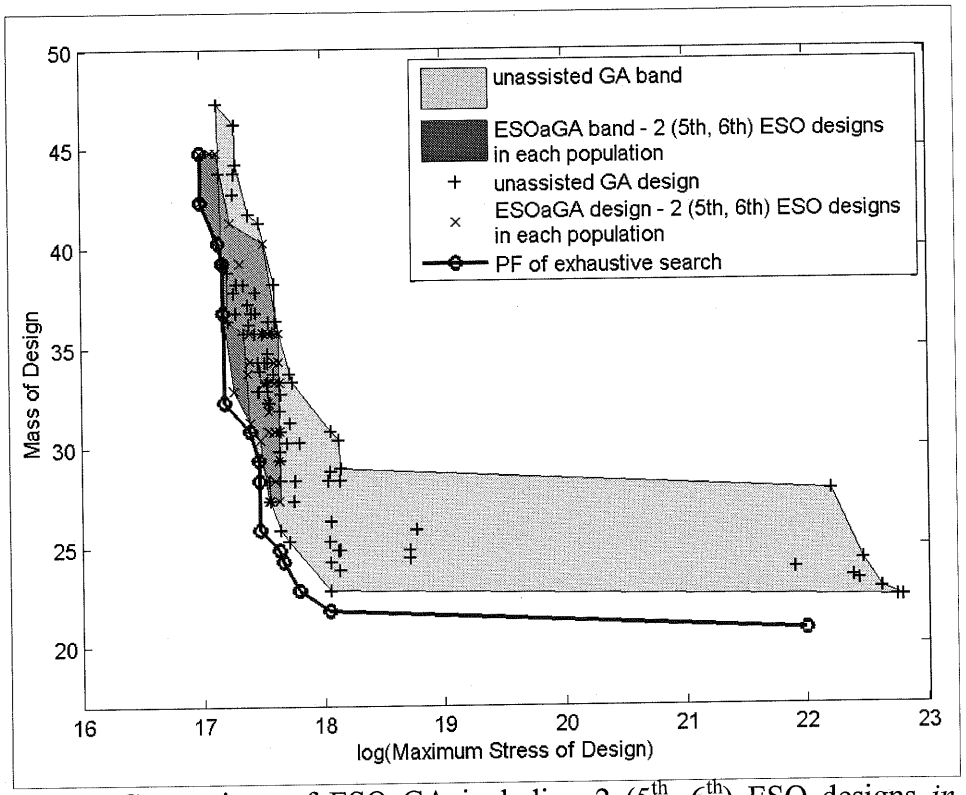


Figure 5.19: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in each population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in each population	2.31E-01	2.73E-01	9.71E-02
Unassisted GA	6.54E-02	7.73E-02	3.26E-02

Table 5.20: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in each population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population).

Various cases of ESOaGA including ESO designs in each population are considered next so that the effect of the number of ESO inserted designs can be examined. The effect of the number of generations is examined by changing this

number from 40 to 20. All the examined cases for 20 generations and 40 generations are presented correspondingly in Tables 5.21 and 5.22. Because of the fact that visual comparison among four (or more) cases becomes difficult (see Figure 5.16), in these cases only the quantitative comparison using the quality indicators is assessed. The previous case of two ESO designs in each population is compared with cases of including more than two designs. As can be seen from Table 5.21, inserting three ESO designs in each population is the best option compared to the cases of including two, five and all ESO designs for 20 generations and 20 designs in each population but still is not more efficient overall than "unassisted" GA. Comparing the same cases but with an extra one of including one ESO design in each population and for 40 generations, the case of including two ESO designs in each population receives the highest rankings in two out of the three quality indicators (Table 5.22a). The best case of including two ESO designs is still though not as good as the "unassisted" GA as shown from the results of the quality indicators in Table 5.22b.

From the results so far, we observe that the ESOaGA method for the 6-node framework achieves superior performance when a small fraction of the GA designs are replaced by ESO designs in the initial population. However, when ESO designs are included in each GA population the performance of GA improves in a restricted region of its solution space. In contrast to inserting designs in the initial GA population, no entire domination is observed while inserting designs in each population. The existence of ESO designs in each GA population assists partially the GA search but at the same time influences negatively the diversity of GA when a small structural problem such as the 6-node case is considered.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th})ESO designs in each population	2.31E-01	2.73E-01	9.71E-02
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in each population	1.06E-01	1.30E-01	4.71E-02
ESOaGA - 5 (4^{th}, 5^{th}, 6^{th}, 7^{th}, 8^{th}) ESO designs in each population	2.80E-01	3.15E-01	1.11E-01
ESOaGA - all ESO designs in each population	3.80E-01	4.18E-01	1.44E-01
Unassisted GA	6.54E-02	7.73E-02	3.26E-02

Table 5.21: Comparison of ESOaGA including 2 (5^{th}, 6^{th}), 3 (5^{th}, 6^{th}, 7^{th}), 5 (4^{th}, 5^{th}, 6^{th}, 7^{th}, 8^{th}), all ESO designs *in each population* and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (3^{rd}) ESO design in each population	1.28E-01	1.27E-01	9.66E-03
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in each population	6.07E-02	7.92E-02	9.63E-03
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in each population	1.11E-01	1.41E-01	5.11E-02
ESOaGA - 5 (4^{th}, 5^{th}, 6^{th} , 7^{th}, 8^{th}) ESO designs in each population	3.15E-01	2.80E-01	1.11E-01
ESOaGA - all ESO designs in each population	6.61E-02	3.95E-02	2.04E-02

Table 5.22a: Comparison of ESOaGA including 1 (3^{rd}), 2 (5^{th}, 6^{th}), 3 (5^{th}, 6^{th}, 7^{th}), 5 (4^{th}, 5^{th}, 6^{th}, 7^{th}, 8^{th}) and all ESO designs *in each population* for a 6-node structure (40 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in each population	6.07E-02	7.92E-02	9.63E-03
Unassisted GA	4.36E-02	1.23E-02	7.69E-03

Table 5.22b: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in each population* and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population).

The 12-node structure

A computationally more demanding problem of the 12-node structure (Figure 5.14) is considered next. The cases of including one, two, five and ten designs produced by ESO (out of 52) in each population are compared. From Table 5.23, we can observe that the corresponding indicator values are reduced as the number of ESO inserted designs is decreased, with the case of including one ESO design giving the highest rankings from all quality indicators. Using only one ESO design in each population of GA is a very efficient approach. As mentioned previously, the balance between the ESO assistance and GA diversity plays an important role for the performance of ESOaGA method. The case of one ESO inserted design is then compared with the "unassisted" GA for the same generation and population size.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25^{th}) ESO design in each population	3.56E-02	5.77E-02	1.25E-02
ESOaGA - 2 (25^{th}, 35^{th}) ESO designs in each population	7.51E-02	6.14E-02	2.41E-02
ESOaGA - 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}) ESO designs in each population	7.94E-02	8.18E-02	2.17E-02
ESOaGA - 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs in each population	1.26E-01	1.53E-01	5.57E-02

Table 5.23: Comparison of ESOaGA including 1 (25^{th}), 2 (25^{th}, 35^{th}), 5 (5^{th}, 15^{th}, 25^{th}, 35^{th}, 45^{th}) and 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in each population* for a 12-node structure (20 generations and 20 designs in each population).

The ESOaGA case of including one ESO design in each population for 20 generations with a population of 20, produces clearly better results than the "unassisted" GA for the same size of population and number of generations as Table 5.24 indicates. While the visual comparison of the ESOaGA method and "unassisted" GA is not easy using the Figure 5.20, the output coming from the quality indicators measurements proves to be useful. In Figure 5.20, we see that replacing a GA design in each population with one ESO design improves the efficiency of GA without any computational cost in many directions of the design space. This shows the positive effect ESO generated designs have in enriching the population for GA runs.

The same ESOaGA cases presented in Table 5.23 are also compared in Table 5.25, for a larger number of generations, to confirm the performance of ESOaGA when the number of the inserted ESO designs is reduced. Again, the same results are obtained from ESOaGA even with a higher number of generations. In contrast to the 6-node structural problem, inserting ESO designs in each GA population assists the GA search. As the objective domain is much larger in the 12-node problem than in the 6-node, the inclusion of ESO designs in each GA population improves the performance of GA without effecting negatively the required diversity of GA. As we can observe the method of inserting ESO designs in each population is much more useful when larger structural problems need to be considered.

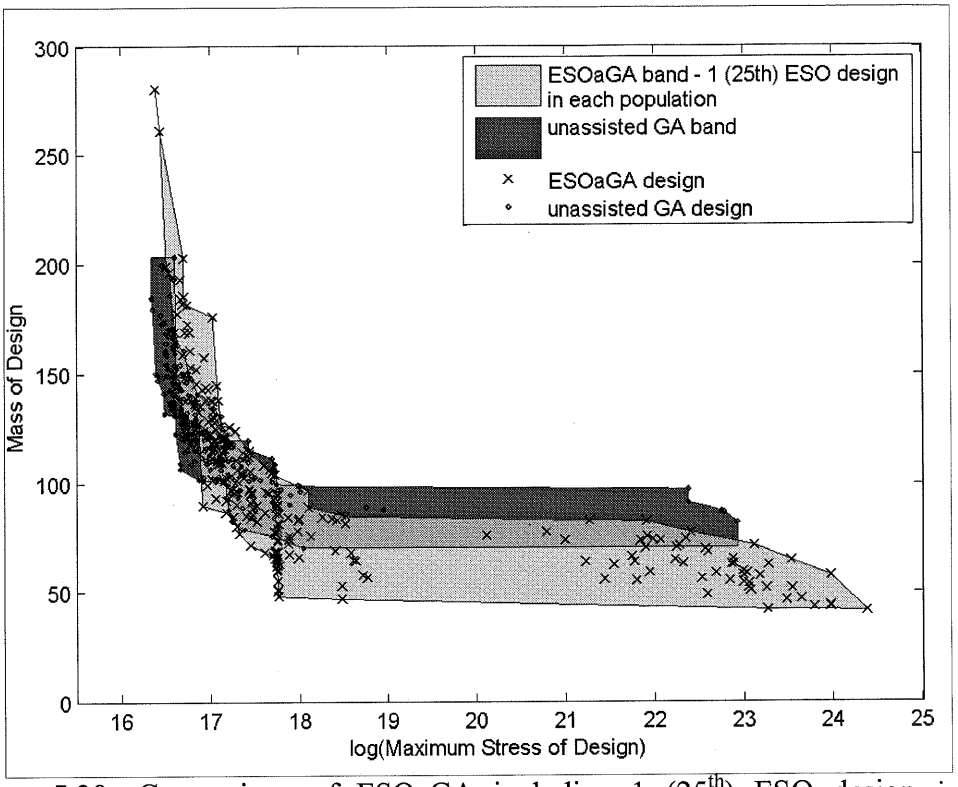


Figure 5.20: Comparison of ESOaGA including 1 (25th) ESO design *in each population* and "unassisted" GA for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25th) ESO design in each population	1.87E-02	3.27E-02	2.72E-03
Unassisted GA	9.94E-02	1.21E-01	4.38E-02

Table 5.24: Comparison of ESOaGA including 1 (25th) ESO design *in each population* and "unassisted" GA for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25th) ESO design in each population	2.52E-02	5.04E-02	6.57E-03
ESOaGA - 2 (25th, 35th) ESO designs in each population	5.76E-02	9.01E-02	3.35E-02
ESOaGA - 5 (5th, 15th, 25th, 35th, 45th) ESO designs in each population	1.01E-01	1.19E-01	4.69E-02
ESOaGA - 10 (5th, 10th, 15th, 20th, 25th, 30th, 35th, 40th, 45th, 50th) ESO designs in each population	1.77E-01	2.23E-01	8.23E-02

Table 5.25: Comparison of ESOaGA including 1 (25th), 2 (25th, 35th), 5 (5th, 15th, 25th, 35th, 45th) and 10 (5th, 10th, 15th, 20th, 25th, 30th, 35th, 40th, 45th, 50th) ESO designs *in each population* for a 12-node structure (40 generations and 20 designs in each population).

The ESOaGA case of one ESO inserted design is compared again with the "unassisted" GA for validation reasons. Similar to Figure 5.20, ESOaGA explores the

design space in regions that "unassisted" GA cannot reach under the same generation and population size (Figure 5.21). The fact that ESOaGA improves in certain regions but not in all, missing some good designs that "unassisted" GA obtains, means further comparison is required using the assistance of quality indicators (Table 5.26). All three quality indicators give higher rankings to ESOaGA method, verifying in this way the superiority of ESOaGA against the "unassisted" GA approach for a larger number of generations.

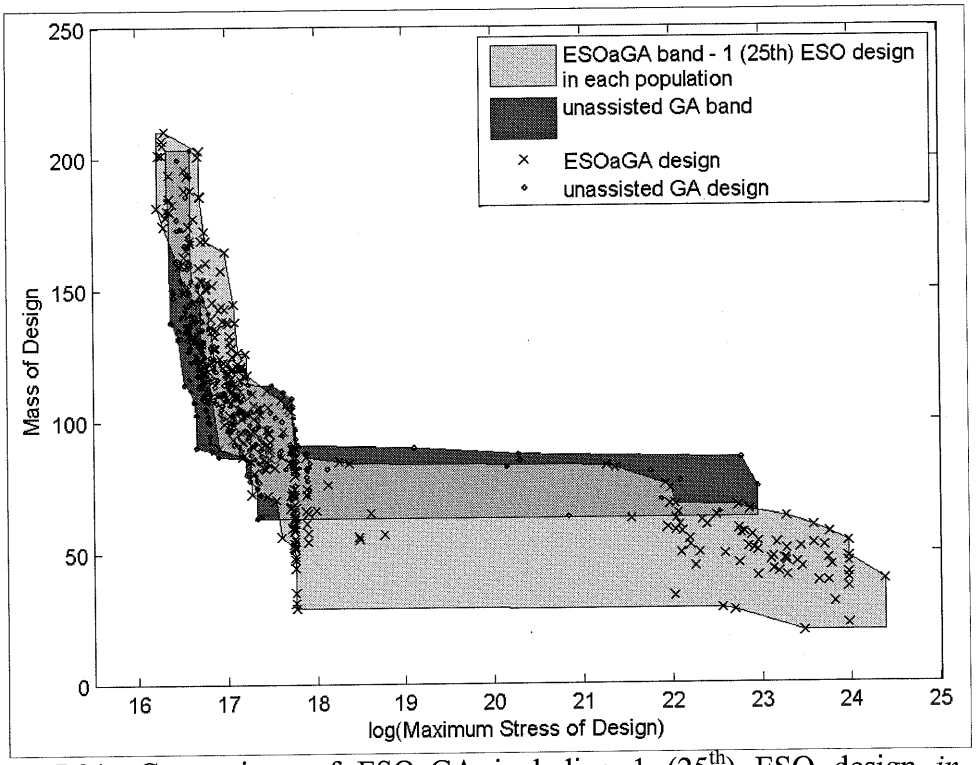


Figure 5.21: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and "unassisted" GA for a 12-node structure (40 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25^{th}) ESO design in each population	1.99E-02	4.63E-02	0.00E+00
Unassisted GA	1.93E-01	2.62E-01	1.00E-01

Table 5.26: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and "unassisted" GA for a 12-node structure (40 generations and 20 designs in each population).

Tables 5.27-5.29, illustrate the quantitative comparisons between the ESOaGA including one (25^{th}) ESO design in each population and "unassisted" GA (40

generations and 20 designs in each population), for each of the 30 different initial seed cases. As in Tables 5.15-5.17, here also the darker blue colour corresponds to the method with the higher ranking (Rank 1) and the lighter blue colour to the method of lower ranking (Rank2). We can observe that ESOGA obtains higher rankings than "unassisted" GA for the most of the different seed cases. In particular, ESOaGA achieves higher rankings in most of the total seed cases.

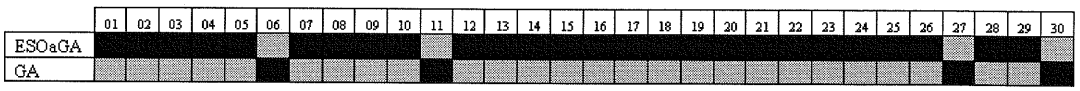


Table 5.27: Comparison of ESOaGA including ESO designs *in each population* and "unassisted" GA for each initial seed case with respect to hypervolume indicator.

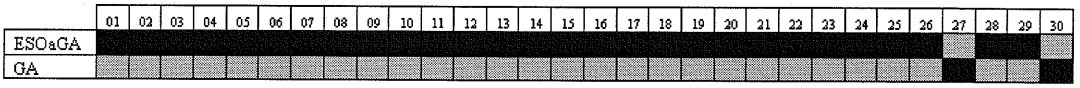


Table 5.28: Comparison of ESOaGA including ESO designs *in each population* and "unassisted" GA for each initial seed case with respect to epsilon indicator.

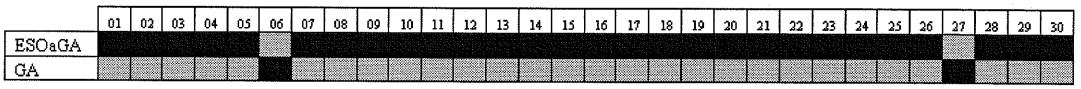


Table 5.29: Comparison of ESOaGA including ESO designs *in each population* and "unassisted" GA for each initial seed case with respect to *R* indicator.

Comparisons between the method of replacing GA designs in the initial population and replacing GA designs in each population are presented next. The comparison between these two methods which share the same basic concept takes place in order to find the most efficient approach.

To show that it is more profitable to include ESO designs in just the initial population and not in each population, the results are presented in Figure 5.22. The results from ESO designs being inserted just in the initial population are superior visually as well as quantitatively. All the three quality indicators (see Table 5.30) confirm this.

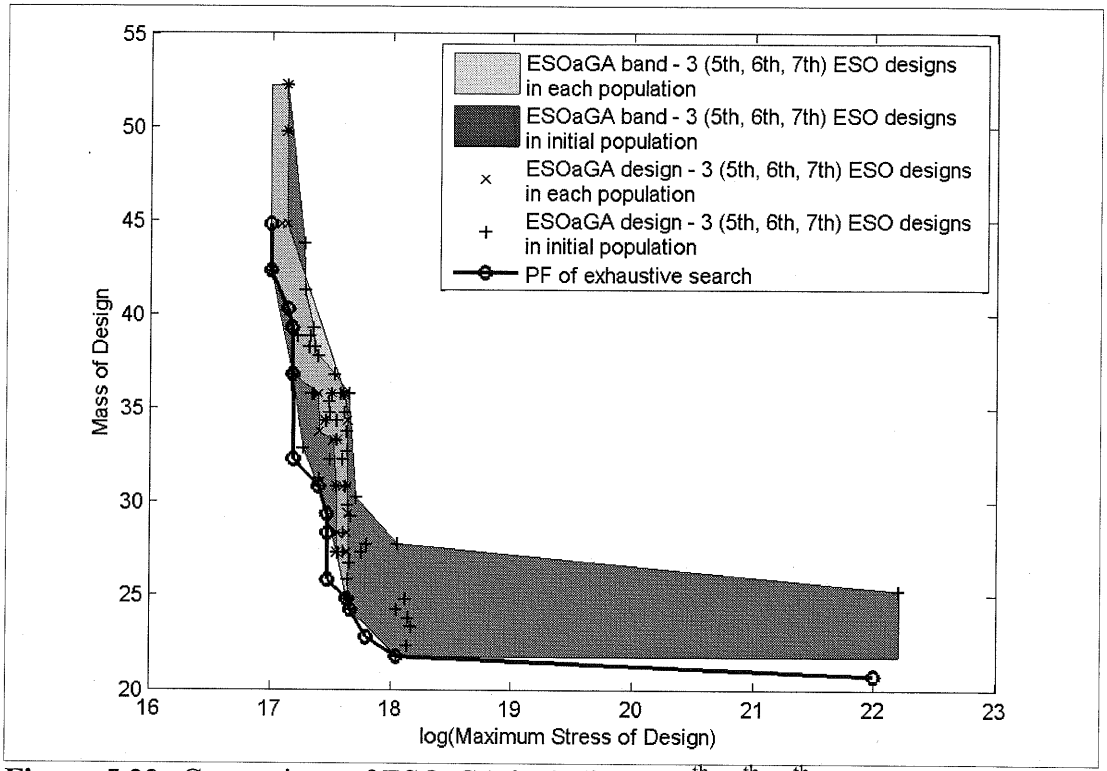


Figure 5.22: Comparison of ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in each population* and ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in each population	1.06E-01	1.30E-01	4.71E-02
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02

Table 5.30: Comparison of ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in each population* and ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in the initial population* for a 6-node structure (20 generations and 20 designs in each population).

The conclusions drawn from Figure 5.22 and Table 5.30 remain the same when the number of generations is increased from 20 to 40. These results are presented in Figure 5.23 and Table 5.31.

For small structural problems such as the 6-node framework, inserting ESO designs only in the initial population provides a more significant improvement in the

performance of GA than doing the same in each population. For larger problems, it turns out that inserting ESO designs *in each population* is more advantageous.

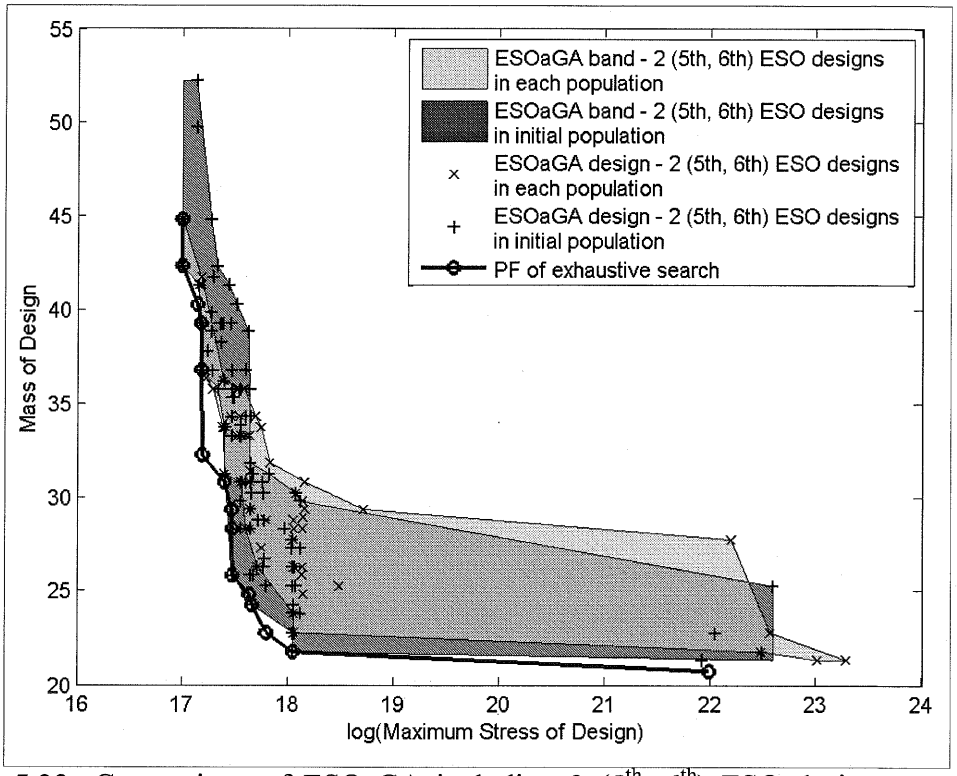


Figure 5.23: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in each population* and ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in initial population	1.17E-02	4.45E-02	6.16E-03
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in each population	6.07E-02	7.92E-02	9.63E-03

Table 5.31: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in each population* and ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* for a 6-node structure (40 generations and 20 designs in each population).

The best cases found so far for the ESOaGA method of including ESO designs in the initial and in each population for the 12-node structure are also compared. Figure 5.24 demonstrates that the ESOaGA approach of including one ESO design in each population finds more promising designs and in more directions than the

ESOaGA approach of including ten ESO designs in the initial population only. As we can see from Figure 5.24, a majority of the designs obtained by the approach of inserting ESO designs in the initial population are concentrated in the same region of the design space. Choosing to insert more designs in the initial population only instead of inserting fewer designs but in each population, might not be a better approach as the GA search is directed in a more limited region.

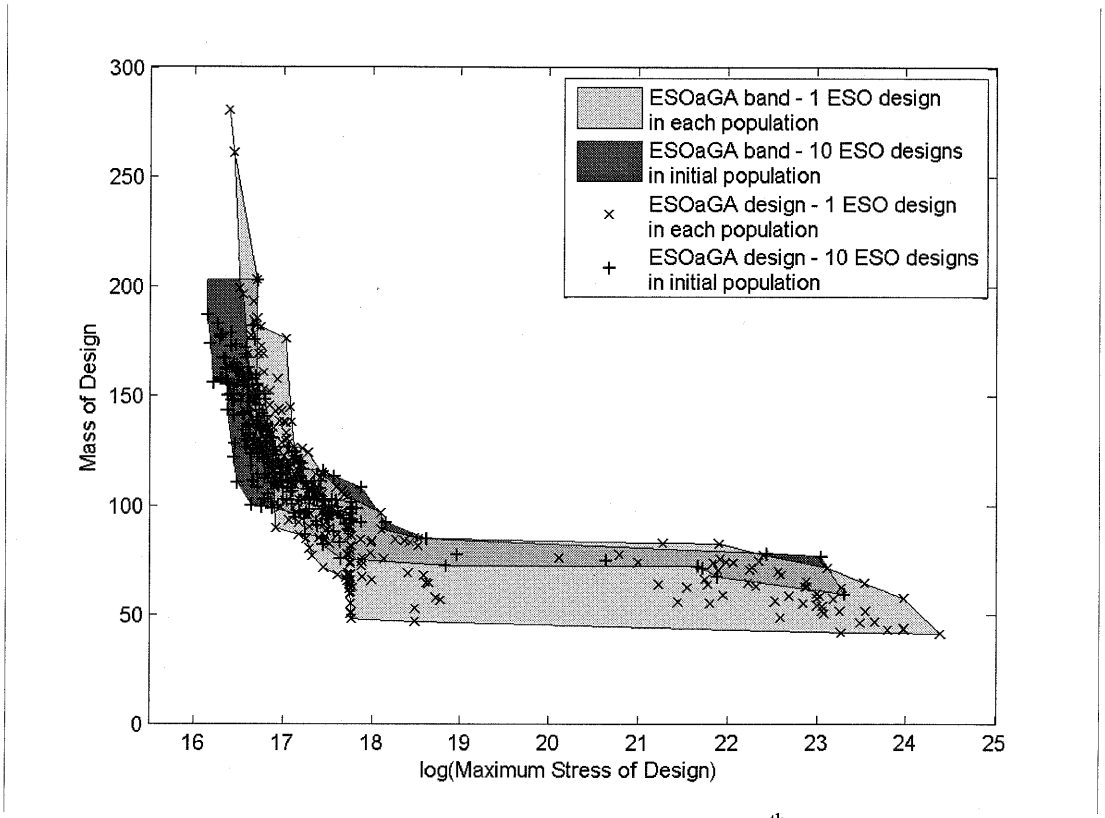


Figure 5.24: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (20 generations and 20 designs in each population).

The results provided by the quality indicators (Table 5.32) agree with the above observations. The indicator values for the case of one ESO inserted design in each population have higher rankings (lower indicator values) than the case of inserting ten in the initial population.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25^{th}) ESO design in each population	3.64E-02	5.24E-02	1.17E-02
ESOaGA - 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs in initial population	9.05E-02	1.16E-01	2.81E-02

Table 5.32: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (20 generations and 20 designs in each population).

The same comparison but for more generations is studied, so that the effect of the generation size on the performance of the two ESOaGA approaches is assessed. Figure 5.25 shows similar behaviours for the two ESOaGA approaches with the ones obtained with less generations. For a computationally more demanding problem such as the 12-node structure the method of including ESO designs in each population is clearly more efficient than the method of including designs in the initial population (see Table 5.33). However, the performance of the ESOaGA when including ESO generated designs in each population depends on the fraction of the ESO designs and GA designs in each population.

Additionally, the quality of the ESO designs inserted to the GA search is a significant factor on the ESOaGA performance. The selection of the ESO designs that need to be inserted in GA population can influence the output of the optimization process. Some ESO designs can be proved to be more helpful to the GA search than others. From the studied examples, the quality of ESO designs has a significant impact on the ESOaGA approach of including ESO designs in the initial population. From the moment that ESO designs can be inserted only in the initial population, the best selections should be made, in order for the GA to have the best possible start without many generations to be considered. In spite of that, if poor quality ESO designs are inserted to GA, this does not necessarily mean that ESOaGA will perform poorly. The selection process of the ESO designs can be skipped if GA is planned to run for a large number of generations. Further improvement on the performance of ESOaGA method could be achieved by inserting different sets of ESO designs in each population instead of inserting the same every time, especially when larger and more complicated structural examples are considered. In this way, the selection process of ESO designs could be skipped and GA search can be also enhanced in multiple directions.

Summing up, for small structural problems such as the 6-node framework the ESOaGA method of inserting ESO designs in the initial GA population has proven to be more efficient than inserting ESO designs in each population. The smaller the structural problem, the smaller its solution space will be, and consequently including designs in each population will affect the GA diversity negatively—thus preventing GA to search sufficiently for a global solution. On the other hand, for larger structures such as the 12-node framework, the most efficient approach of ESOaGA is to insert ESO designs in each population provided that a balance between the ESO and GA designs is maintained during the search process. As mentioned previously, including a very large number of ESO designs in each population will influence the diversity of the global search negatively, leaving no space for more new promising designs to appear. Keeping in mind that the ESO and GA designs should be in a right balance at every generation, enriching each GA population with ESO designs leads to more efficient results for larger structural problems with larger solution spaces.

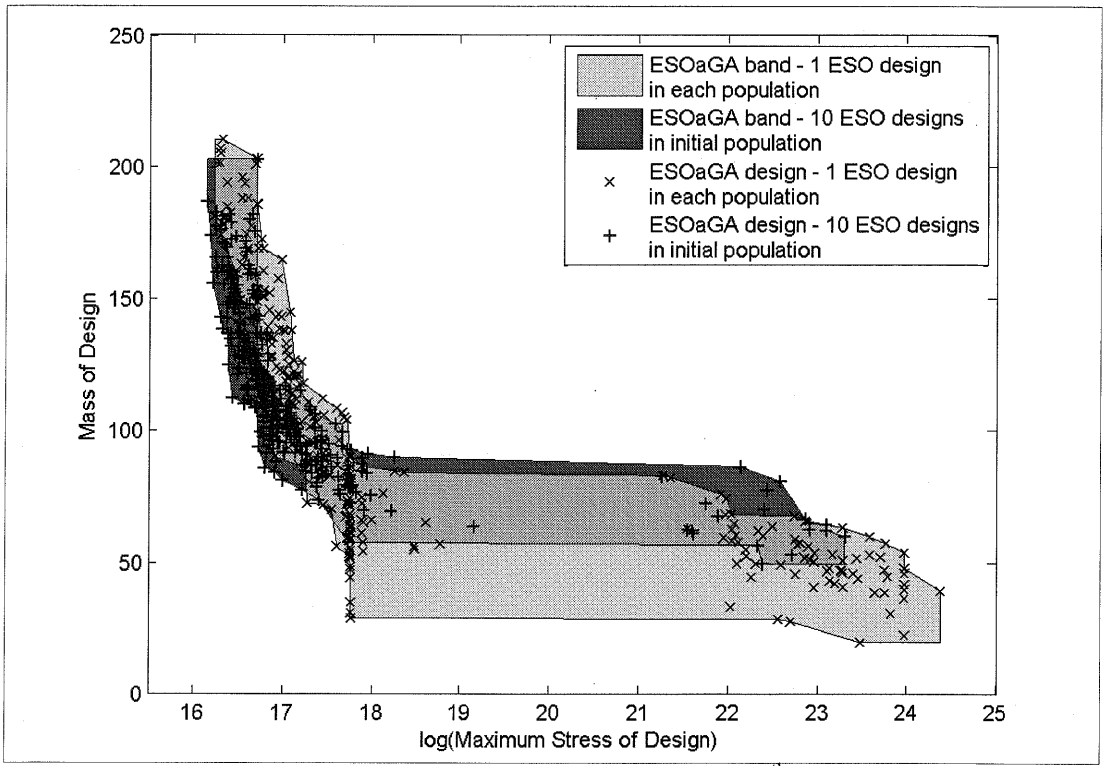


Figure 5.25: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in the initial population* for a 12-node structure (40 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25th) ESO design in each population	2.56E-02	5.26E-02	4.20E-03
ESOaGA - 10 (5th, 10th, 15th, 20th, 25th, 30th, 35th, 40th, 45th, 50th) ESO designs in initial population	1.48E-01	1.78E-01	6.53E-02

Table 5.33: Comparison of ESOaGA including 1 (25th) ESO design *in each population* and ESOaGA including 10 (5th, 10th, 15th, 20th, 25th, 30th, 35th, 40th, 45th, 50th) ESO designs *in the initial population* for a 12-node structure (40 generations and 20 designs in each population).

5.5. The use of GA designs as starting points for ESO runs (GAaESO)

An alternative approach of blending the strengths of a GA with those of ESO is presented next. Here, the designs that are obtained by the GA for 30 different random initial seed cases are used as starting points for a family of ESO runs. The efficiency is now achieved by running the GA only for a *relatively small number* of generations and using the best designs thus obtained as the starting designs for a number of subsequent ESO runs. Since each ESO run is computationally cheap, the overall cost of the search is hoped to be substantially less than that of running the GA for a large number of generations in order to achieve the same quality of designs. Alternatively, given fixed computational resources the hybrid method is expected to produce designs that are superior to "unassisted" GA or ESO. It is important to economise on the number of generations used at each stage of running GA and ESO successively—otherwise the benefits of combination will be minimal and the algorithm will increasingly behave like "unassisted" GA.

A flow chart of the proposed GA assisted ESO method (GAaESO) is shown in Figure 5.26. The GA generated designs are represented in a bit-string format and are transformed into the appropriate ESO connectivities for the ESO method to start. The procedure of running ESO for each design that was produced by the initial GA run is followed one by one. Finally, all the designs generated at the end of each ESO run are ranked for Pareto optimality.

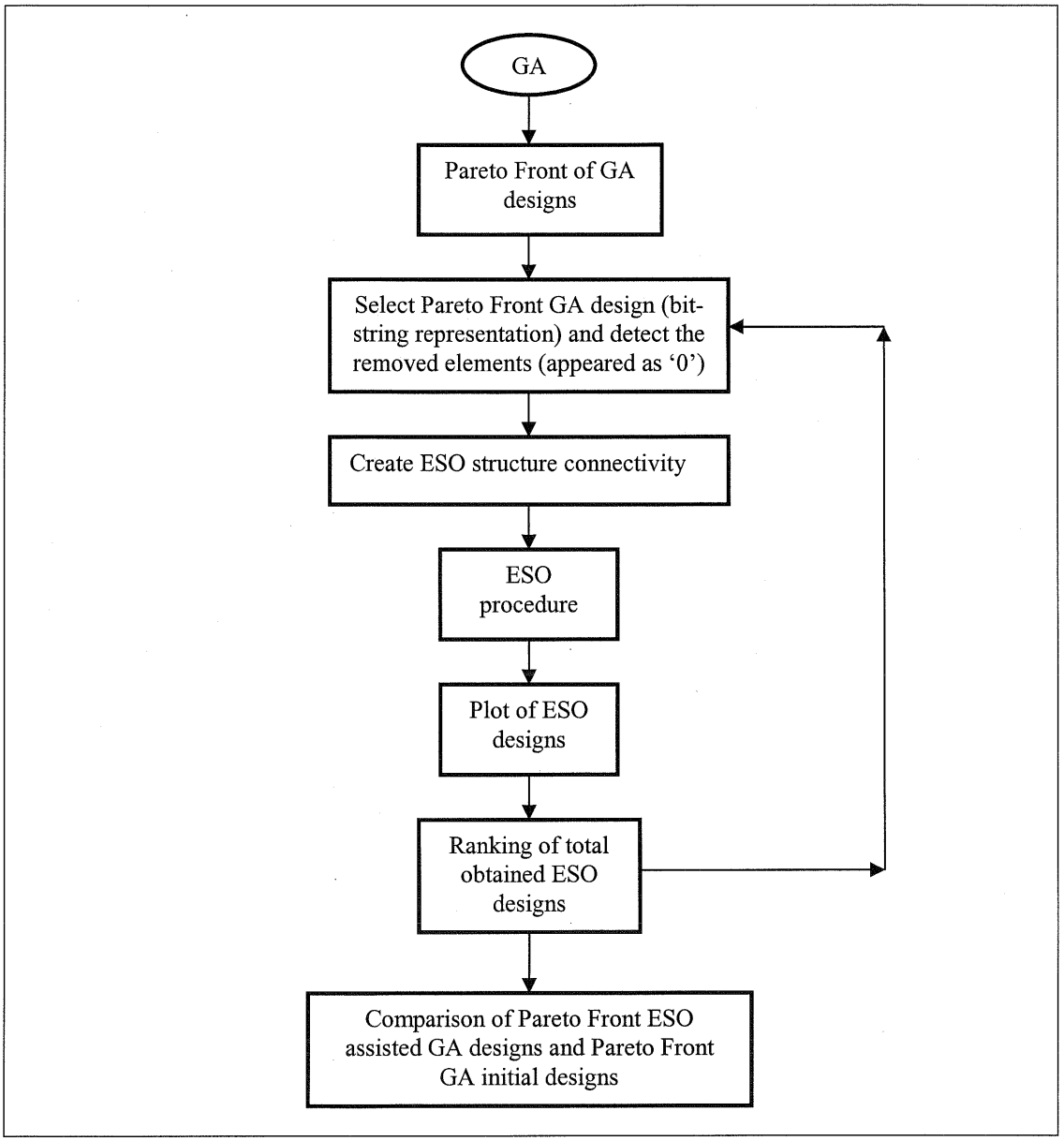


Figure 5.26: Flow chart of GA assisted ESO (GAaESO) method.

Example 1: 6-Node framework structure

The method proposed in this section is applied to the problem of topology search for the 6-noded structure. The GA is run 30 times (with different initial seed each time) for 20 generations and 20 designs in each population; the resulting designs are shown using circles (Figure 5.27). Similar to the ESOaGA case, here the final output designs of the GAaESO method must have at the most 7 members removed from the ground structure. Therefore, the designs obtained from GA should be

structures with 7 or less members removed. The best GA designs are taken as starting points for further ESO runs. Finally, each ESO run stops when a total of 7 members have been removed from the ground structure. If a GA design has already 7 members removed from the ground structure, it is obviously not further considered for an initial structure for ESO run. Each ESO trajectory is shown using a solid line with stars. Since the ESO runs are computationally efficient, the method provides a cheap way of improving a GA based topology search with the help of ESO. As shown in Figure 5.27, this method obtains better designs than the "unassisted" GA without significant additional computational cost. The results of the GAaESO method are presented in the same style as the ESOaGA method. The band coloured as yellow includes the design space explored by GAaESO and represents the area created between the least efficient designs and non-dominated designs of the proposed hybrid method.

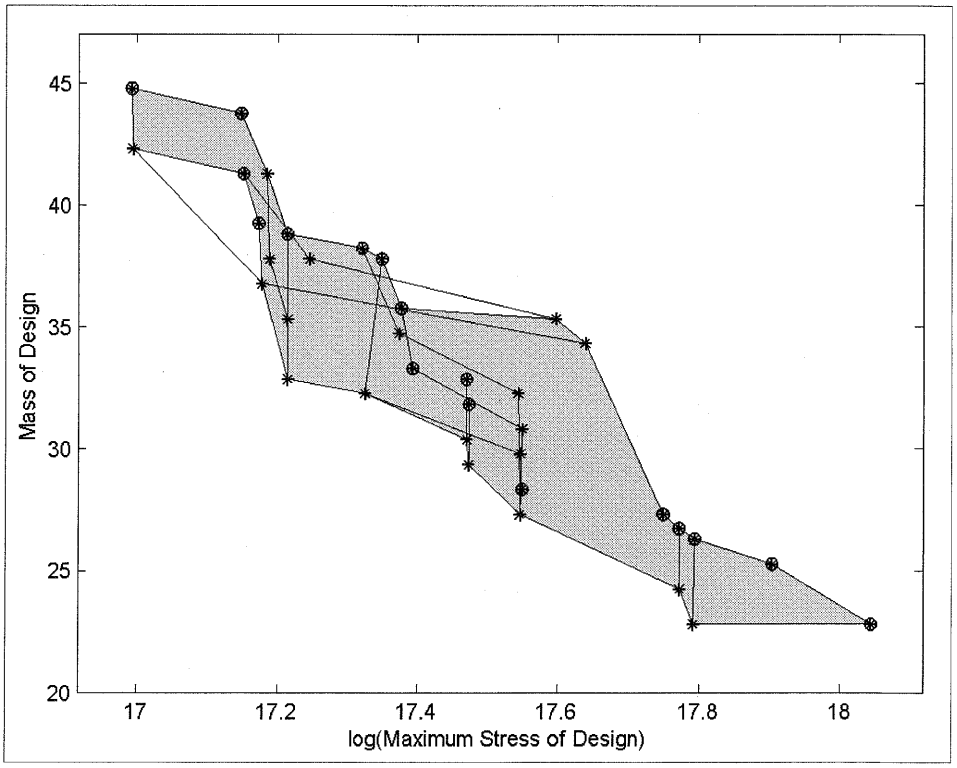


Figure 5.27: "Unassisted" GA designs (circled) functioned as starting points for ESO runs (solid lines with stars) for the 6-node structure.

The designs obtained by the use of this hybrid method are superior to the designs produced by plain GA for 20 generations and 20 designs in each population (Figure 5.28). In Figure 5.28, GA designs are taken as starting points for ESO runs, having in that case one ESO trajectory for each GA design. As we can see from Figure 5.28, all the GAaESO designs dominate the designs obtained by "unassisted"

GA. Moreover, GAaESO does not provide as many as poor designs as "unassisted" GA under the same computational time. This observation agrees with the results gained from quantitative measurements of quality indicators presented in Table 5.34.

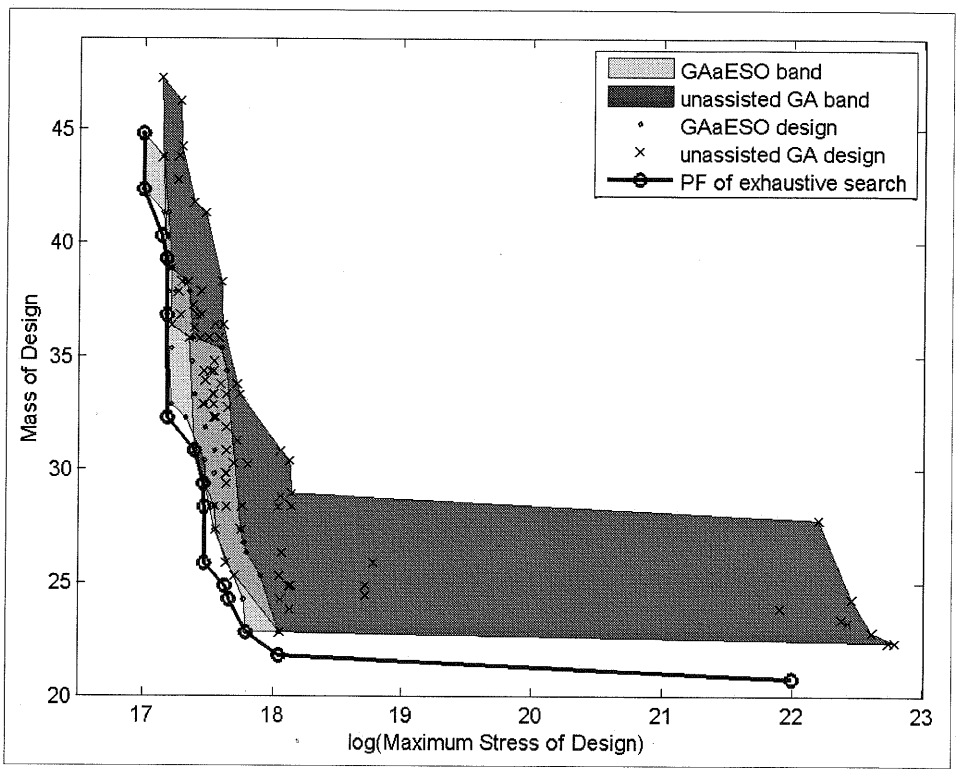


Figure 5.28: Comparison of GAaESO designs and "unassisted" GA designs for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
GAaESO	5.22E-02	8.53E-02	3.09E-02
Unassisted GA	6.54E-02	7.73E-02	3.26E-02

Table 5.34: Comparison of GAaESO and "unassisted" GA for a 6-node structure (20 generations and 20 designs in each population).

Some results obtained by GAaESO for 20 generations and 20 designs in each population are found to be even better than the results of the "unassisted" GA for 40 generations and 20 designs in each population (Table 5.35). This is particularly encouraging as GAaESO proves to be more efficient in some regions than "unassisted" GA with even less computational cost. Figure 5.29, shows several GAaESO designs to be in the non-dominated set of the entire regions explored by both methods.

Although GAaESO is more effective than the "unassisted" GA, further improvement in its efficiency can be observed when applied to larger structures. We

can notice that taking the "unassisted" GA designs as starting points for a family of ESO runs, does not evolve as much as expected in the regions of smaller stress values. Applying ESO to the GA produced designs; in most of the cases the weight is reduced at the expense of increased maximum stress. The reason for this behaviour of the ESO trajectories is that the GA designs used as starting points for ESO runs are generally structures with few remaining members. As the considered 6-node example is a relatively small structure, it is logical to think that the designs produced by GA have few members left connected to each other. Consequently, these small produced designs do not allow ESO to evolve in more promising paths. The difference of the results between the GAaESO and "unassisted" GA expands when a larger structural example is considered.

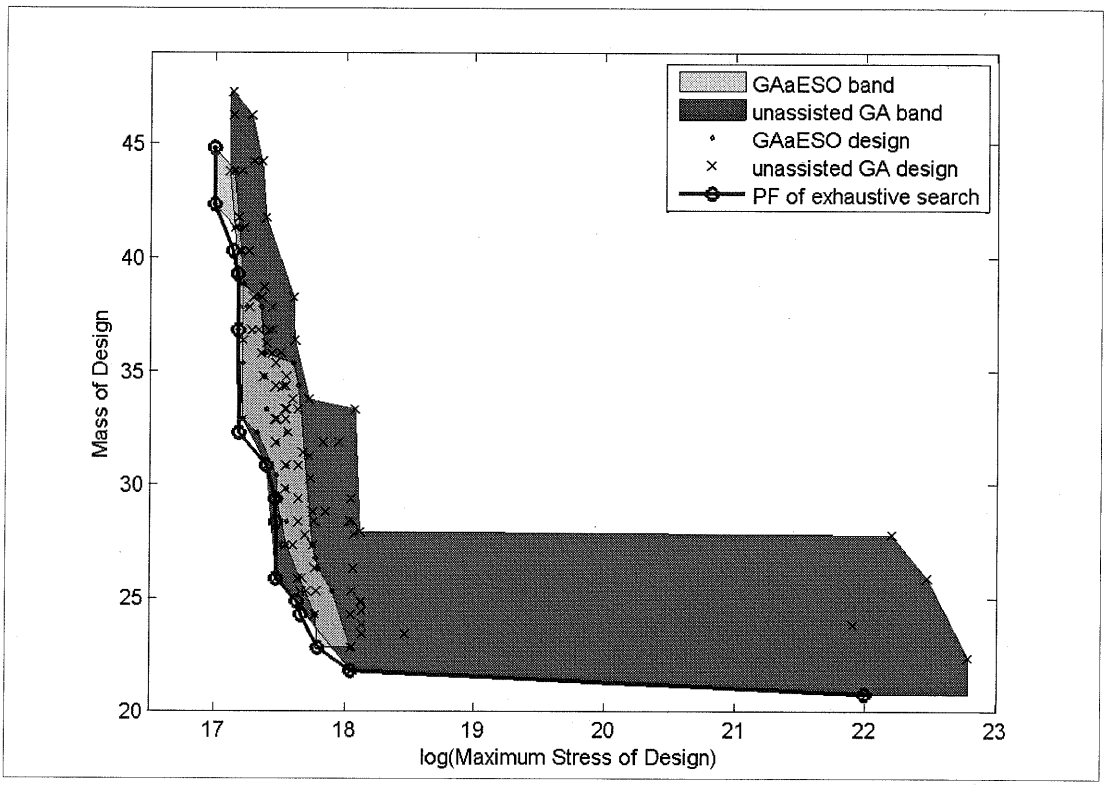


Figure 5.29: Comparison of GAaESO method (20 generations and 20 designs in each population) and "unassisted" GA for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
GAaESO	5.22E-02	8.53E-02	3.09E-02
Unassisted GA	4.36E-02	1.23E-02	7.69E-03

Table 5.35: Comparison of GAaESO (20 generations and 20 designs in each population) and "unassisted" GA (40 generations and 20 designs in each population) for a 6-node structure.

Example 2: 12-Node framework structure

To explore the efficiency of GAaESO for more complex design tasks, the 12-node structure (Figure 5.14) was considered next for numerical experiments. Using the GA designs for 20 generations and 20 designs per population as starting points for ESO runs, significantly improved designs are obtained (Figure 5.30). As in the case of the previous example, the band constructed of the least efficient and PF designs of the GAaESO is used to represent the corresponding solution space. Here, the designs obtained initially by GA can be considered as better starting points for ESO runs with respect to the GA designs of the 6-node structure, because of the more dense connectivities in their structures. Applying ESO to a highly connected structure enhances the scope for improvement, as the possibilities of ESO to follow a more successful trajectory rise.

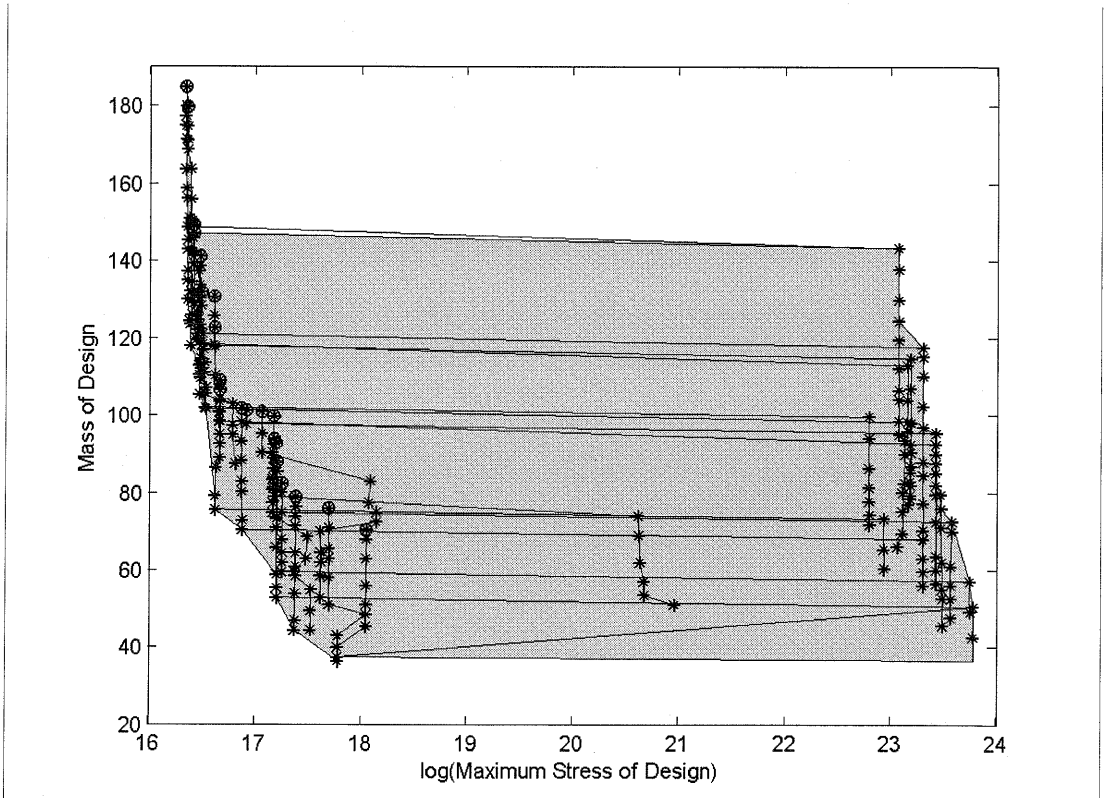


Figure 5.30: "Unassisted" GA designs (circled) functioned as starting points for ESO runs (solid lines with stars) for the 12-node structure.

Next, the band produced by GAaESO for 20 generations and 20 designs in each population is compared with the results of GA for the same and larger sizes of generations and populations. It is shown in Figure 5.31 that the PF designs generated by GAaESO dominate all the designs of "unassisted" GA with larger number of

generations and populations. Considering that ESO takes relatively small computational time, we can apply it repeatedly with already obtained GA designs as starting points. We notice that applying ESO at GA designs obtained in 20 generations and 20 designs per population produces more efficient designs than running "unassisted" GA for 40 generations and 20 designs per population (Figure 5.31). Instead of running GA for a large number of generations, we can save a lot of computational time by simply running GA for a small number of generations followed by ESO with the GA provided designs as the starting points. For clarity, the intermediate ESO steps (shown as stars in Figure 5.30) have been omitted from Figure 5.30 and only the starting points provided by the "unassisted" GA and the designs obtained from the ESO runs are included in the coloured blue band of GAaESO in Figure 5.31. The superiority of GAaESO can be confirmed by the indicator values in Tables 5.36 & 5.37.

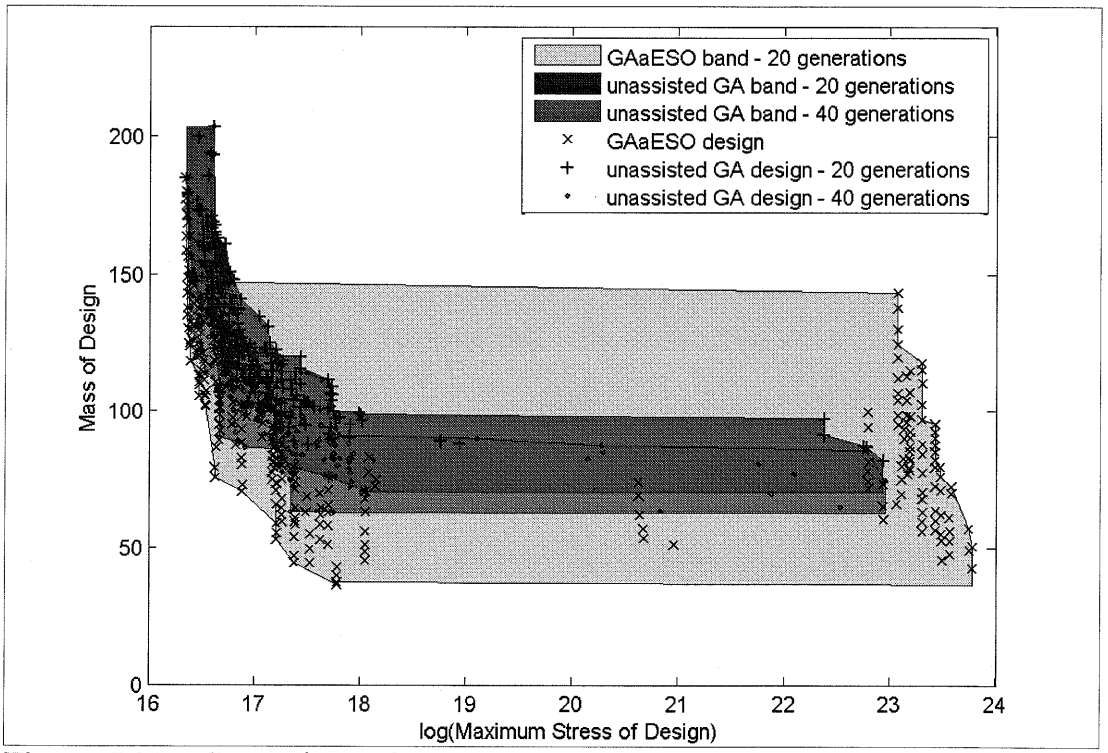


Figure 5.31: Comparison of GAaESO method (20 generations and 20 designs in each population) and "unassisted" GA (20 & 40 generations and 20 designs in each population) for a 12-node structure.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
GAaESO	0.00E+00	0.00E+00	0.00E+00
Unassisted GA	2.50E-01	2.29E-01	8.36E-02

Table 5.36: Comparison of GAaESO and "unassisted" GA for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
GAaESO	0.00E+00	0.00E+00	0.00E+00
Unassisted GA	1.43E-01	1.78E-01	6.56E-02

Table 5.37: Comparison of GAaESO (20 generations and 20 designs in each population) and "unassisted" GA (40 generations and 20 designs in each population) for a 12-node structure.

5.6. Comparisons between ESOaGA and GAaESO methods

5.6.1. Comparison of the two proposed approaches for the 6-node structure

Initially, the ESO assisted GA method is applied to the 6-node structure (see Figure 5.3). Three of the nine designs obtained by ESO using as the starting point the ground structure of the 6-node structure, shown in Figure 5.3, replace the three most inefficient designs (lowest ranking) in the initial GA population. The results of this process are compared with the results of the GA assisted ESO method in which designs of the "unassisted" GA are taken as starting points for ESO runs. Comparison of both approaches is completed for 20 generations and the size of population kept as 20 (Figures 5.32).

Figure 5.32 shows that the designs obtained by the ESOaGA method dominate most of the designs of the GAaESO method. Although, the GAaESO method obtains better results than the "unassisted" GA, it cannot surpass the performance of ESOaGA method. Most of the non-dominated designs of ESOaGA match with the non-dominated designs of GAaESO. The best designs obtained from both methods are quite close. GAaESO, however, does not produce as many poor designs as ESOaGA. In contrast to ESOaGA, the majority of the GAaESO designs are concentrated close to its non-dominated set. If GA needs to be used for the optimization of a framework structure, the quality of the results can be easily enhanced by applying ESO separately to the produced GA designs with no significant extra computational cost. However, for this particular framework example the ESOaGA method achieves slightly superior results (Table 5.38). This does not mean necessarily that ESOaGA is universally a better method than GAaESO. In the 6-node framework, the superiority of ESOaGA is due to the poor performance of GAaESO on small sized frameworks.

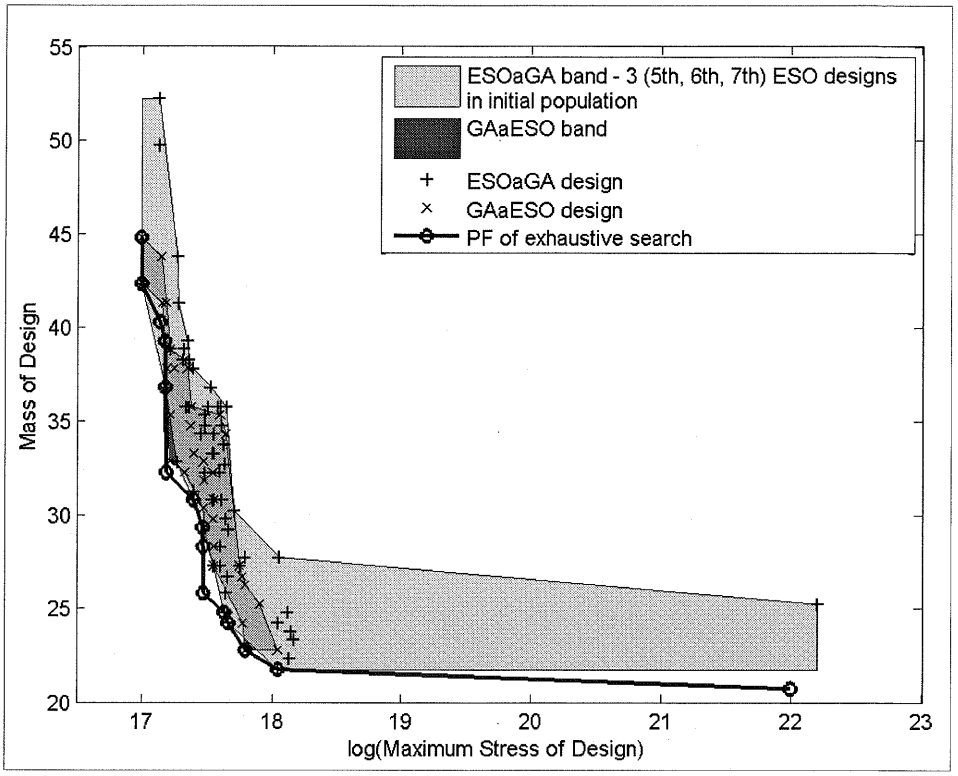


Figure 5.32: Comparison of ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in each population* and GAaESO for a 6-node structure (20 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 3 (5^{th}, 6^{th}, 7^{th}) ESO designs in initial population	1.33E-02	4.61E-02	1.14E-02
GAaESO	5.22E-02	8.53E-02	3.09E-02

Table 5.38: Comparison of ESOaGA including 3 (5^{th}, 6^{th}, 7^{th}) ESO designs *in the initial* population and GAaESO for a 6-node structure (20 generations and 20 designs in each population).

Moreover, the ESOaGA case of including two ESO designs in the initial population is compared with the GAaESO for 40 generations and 20 designs in each population. In Figure 5.33, we notice again that the ESOaGA method is superior to the GAaESO method for a larger number of generations. The increase of generations does not influence the evolution significantly for the GAaESO. The range of possible designs that can exist for a simple structure is limited compared to a complex one. Therefore, whilst GA reaches a level where good designs are found for a small computational time, running more generations will only obtain marginally better designs. Considering this, applying ESO to the GA designs will only provide a slight

improvement in the output. This comparison highlights the fact that GAaESO provides poorer designs than the ESOaGA when applied to relatively small framework structures. The corresponding indicator values support this conclusion and are shown in Table 5.39.

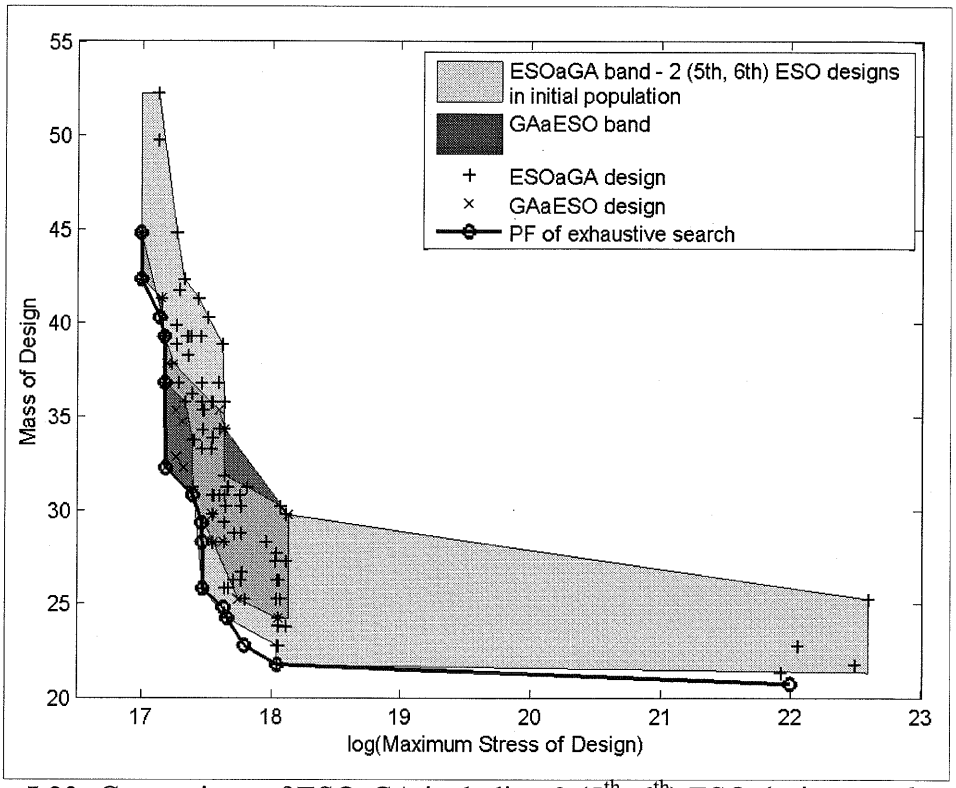


Figure 5.33: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* and GAaESO for a 6-node structure (40 generations and 20 designs in each population). The PF of the exhaustive search is also shown.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 2 (5^{th}, 6^{th}) ESO designs in initial population	1.17E-02	4.45E-02	6.16E-03
GAaESO	1.10E-01	1.46E-01	5.29E-02

Table 5.39: Comparison of ESOaGA including 2 (5^{th}, 6^{th}) ESO designs *in the initial population* and GAaESO for a 6-node structure (40 generations and 20 designs in each population).

5.6.2. Comparison of the two proposed approaches for the 12-node structure

A comparison between the two methods now is made for the 12-node structure (Figure 5.14). The results achieved by ESOaGA method in which one ESO design is entered in each population of GA are compared with the results of GAaESO method

for the same number of generations and same population size per generation. In Figure 5.34, we see clearly that the GAaESO method obtains superior designs to the ESOaGA method for 20 generations and 20 designs included in each population. GAaESO designs dominate almost all the designs of ESOaGA for the same number of generations and the same population size. The dominance of GAaESO against the ESOaGA method is also supported by the quality indicators results in Table 5.40. In contrast to the 6-node case, the GAaESO applied to larger structures such as the 12-node framework, obtains better results than the ESOaGA method. The GA designs obtained for the 12-node structure, as opposed to the 6-node structure, are good starting points for ESO runs as they have mostly a large number of remaining members. ESO generally becomes more efficient when it is applied to more highly connected structures.

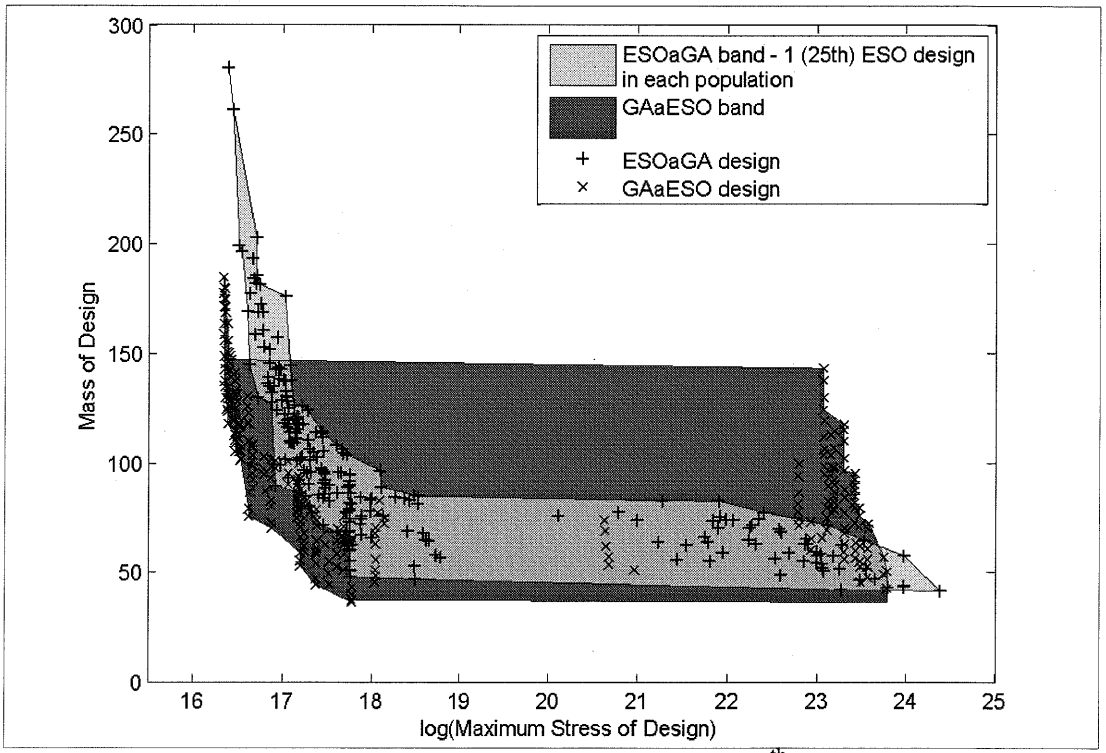


Figure 5.34: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and GAaESO for a 12-node structure (20 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25^{th}) ESO design in each population	7.38E-02	6.24E-02	1.18E-02
GAaESO	0.00E+00	0.00E+00	0.00E+00

Table 5.40: Comparison of ESOaGA including 1 (25^{th}) ESO design *in each population* and GAaESO for a 12-node structure (20 generations and 20 designs in each population).

The results produced by the application of the methods are presented for a larger number of generations in order to verify this trend (Figure 5.35). This time, the ESOaGA case of one ESO design inserted in each population is not superior to the GAaESO method for 40 generations and 20 designs in each population (Table 5.41). ESOaGA method seems to have been improved with respect to the corresponding case of 20 generations. Although, GAaESO dominates a lot of non-dominated designs of ESOaGA, as found from the results of the quality indicators GAaESO is not as good as it was for smaller number of generations. Clearly, when the generation size is increased ESOaGA method improves in a higher grade than GAaESO. The behaviour of GAaESO concerning the generation size can be also seen in Figure 5.37.

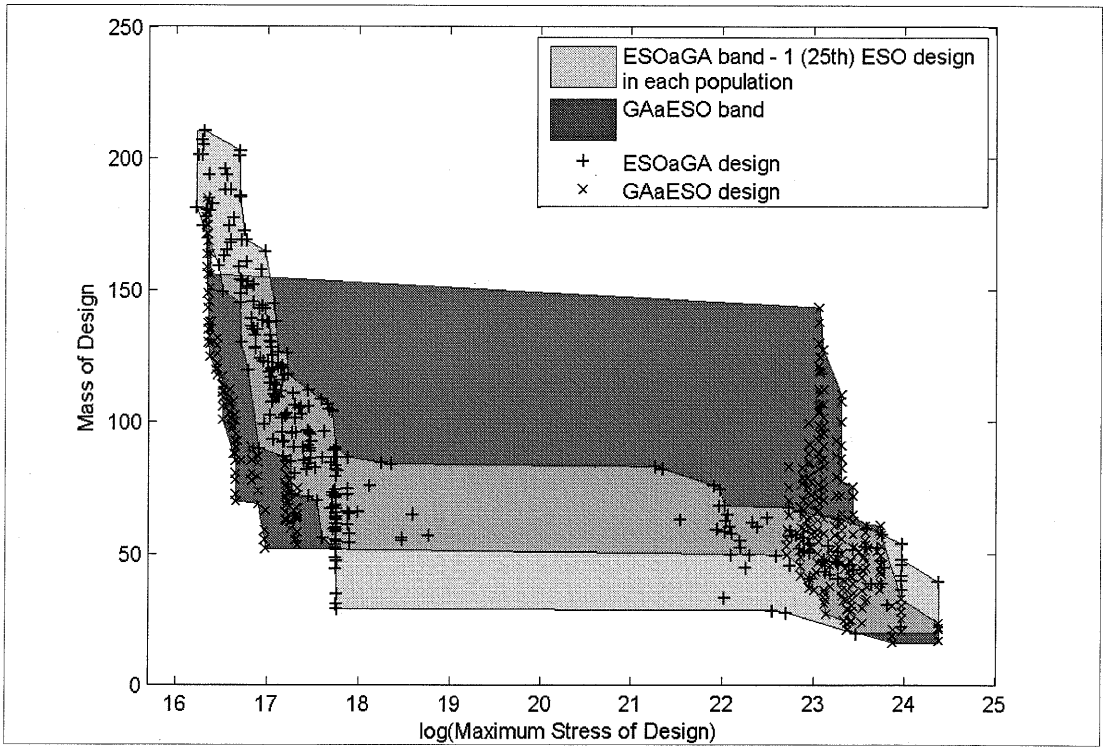


Figure 5.35: Comparison of ESOaGA including 1 (25th) ESO design *in each population* and GAaESO for a 12-node structure (40 generations and 20 designs in each population).

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 1 (25th) ESO design in each population	1.99E-02	4.63E-02	0.00E+00
GAaESO	1.93E-01	2.62E-01	1.00E-01

Table 5.41: Comparison of ESOaGA including 1 (25th) ESO design *in each population* and GAaESO for a 12-node structure (40 generations and 20 designs in each population).

The behaviour of each of the two new methods according to the increase of the number of generations (increase of computational time) is considered next. The designs attained by the ESO assisted GA method for 20, 40 and 80 generations correspondingly are presented in Figure 5.36 to compare computational time and the benefit to structural quality. It can be observed that improvements occur as the number of generations is increased.

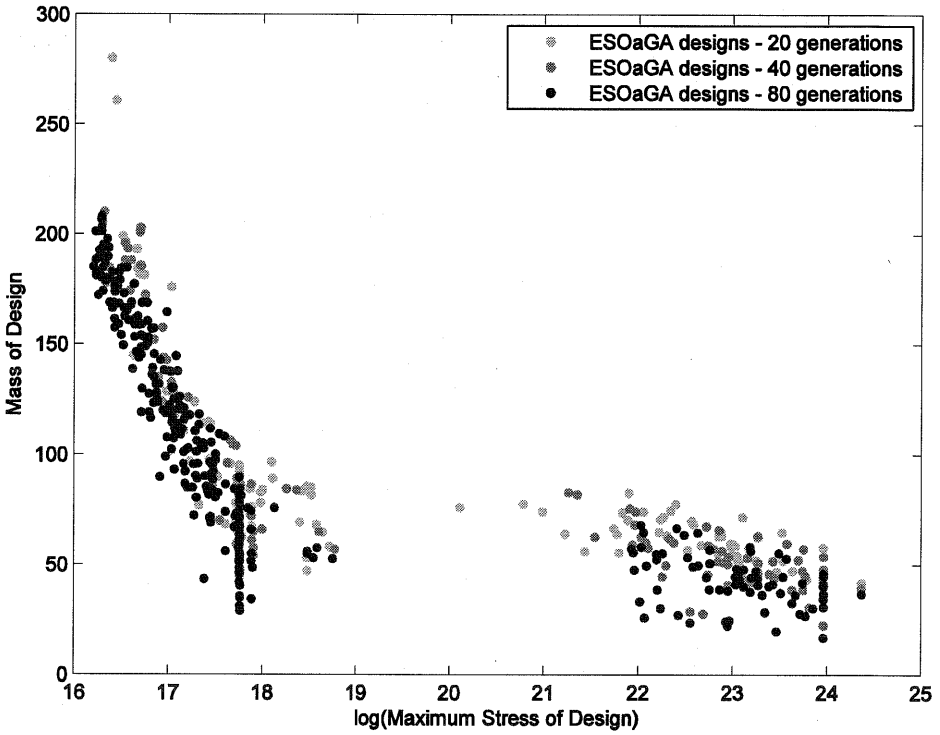


Figure 5.36: Comparison of ESOaGA for a 12-node structure (20, 40 & 80 generations and 20 designs in each population).

The performance of the GA assisted ESO method is examined next when the number of generations is increased. The designs attained by the GA assisted ESO method for 20, 40 and 80 generations correspondingly are presented in Figure 5.37. We notice that GAaESO does not converge as smoothly as ESOaGA when the number of generations is increased. An improvement is observed as the number of generations is increased. However, GAaESO designs obtained for more generations do not dominate entirely the sets of designs of less generations. The final results of GAaESO mainly depend on the performance of ESO applied to each of the generated "unassisted" GA designs. While the number of generations is increased, "unassisted" GA obtains even better designs which in most cases are designs with less dense

connectivities in their structure. Additionally, the denser the connectivity of a structure considered as ESO starting point is, the higher the efficiency of ESO will be.

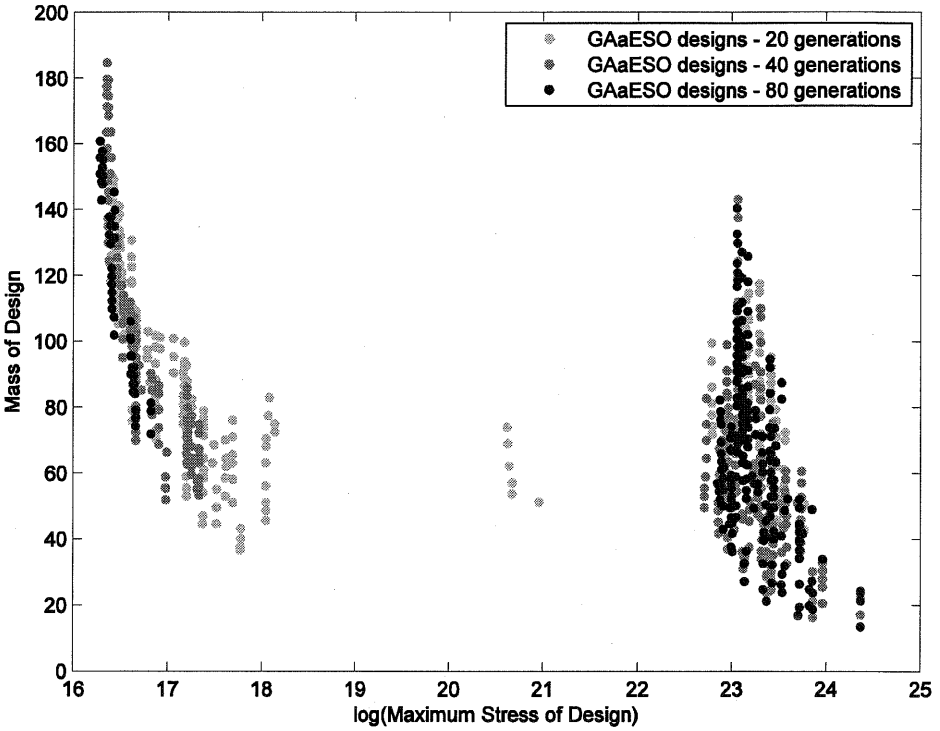


Figure 5.37: Comparison of GAaESO for a 12-node structure (20, 40 & 80 generations and 20 designs in each population).

Finally, by combining ESO and GA iteratively, it is hoped that the method will improve further. One of the schemes presented here is the ESO-assisted-GA-assisted-ESO (ESOaGAaESO), which combines the ideas of the two pre-mentioned methods of ESOaGA and GAaESO. Here, we apply ESO following an ESOaGA run aiming to improve the results over ESOaGA with minimal extra computational cost. After running ESOaGA (for 30 different randomly generated initial seeds) in which ten ESO designs are added to each generation of the GA, the results obtained are taken as starting points for ESO runs. The results of the ESOaGAESO approach are presented in Figure 5.38 and they are compared with the results that have been obtained by ESOaGA and GAaESO for the same population and generation size.

As expected, the ESOaGAaESO designs dominate all the ESOaGA designs at the same computational time; the cost of running an extra ESO is marginal. A large improvement in the results can be obtained by taking ESOaGA designs as starting points for ESO runs. Combining ESO and GA iteratively in this sequence can provide an increase in the efficiency with almost no extra computational cost. Despite that, GAaESO still remains the most efficient optimization method as it dominates the

results of the other two methods. Although, the results of ESOaGA method are better than the results of the "unassisted" GA, applying ESO to both results does not mean that the dominance relationship will be maintained. The efficiency of ESO method does not depend on the quality of the design considered for a starting point but on the connectivity of the structure. Generally, denser designs are more promising starting points for ESO. For this reason, GAaESO is superior to ESOaGAaESO method for 20 generations and 20 designs in each population as seen in both Figure 5.38 and Table 5.42.

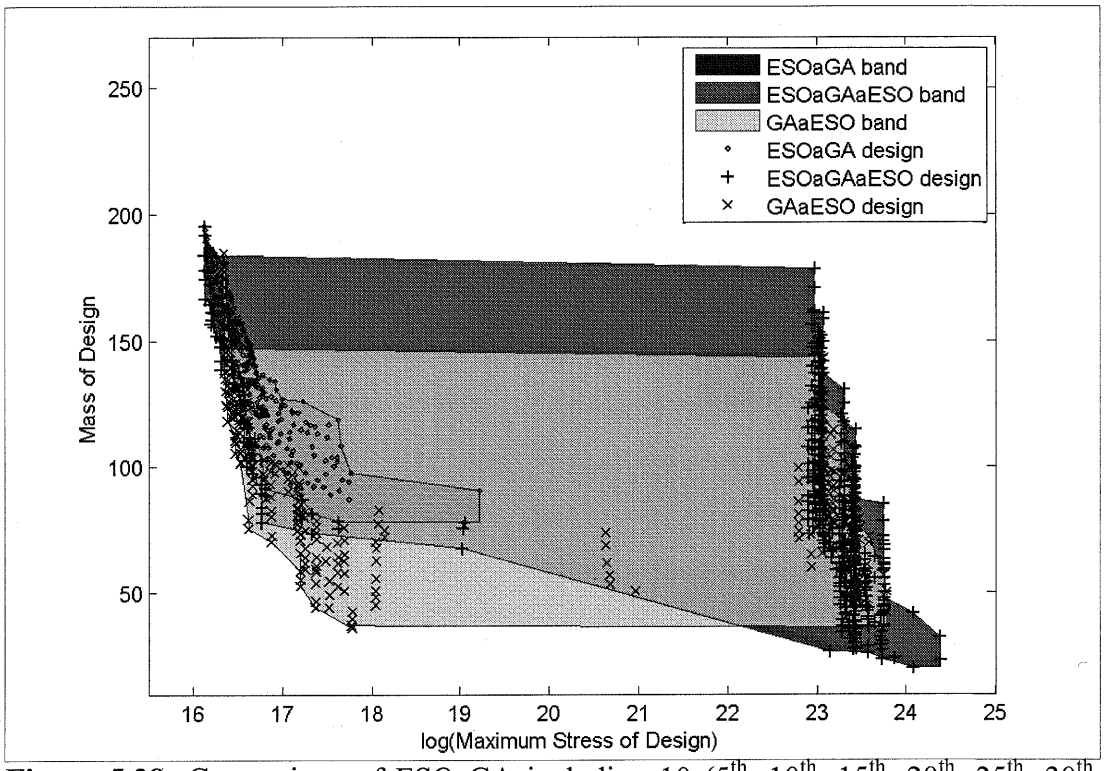


Figure 5.38: Comparison of ESOaGA including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in each population*, ESOaGAaESO and GAaESO for a 12-node structure (20 generations and 20 designs in each population). The brown band represents the ESOaGA as the blue, red and yellow bands are overlaid.

	Hypervolume Indicator	Epsilon Indicator	*R* Indicator
ESOaGA - 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs in each population	9.58E-02	8.08E-02	4.30E-02
GAaESO	2.82E-02	6.11E-02	3.19E-02
ESOaGAaESO	9.46E-02	1.44E-01	0.00E+00

Table 5.42: Comparison of ESO assisted GA (ESOaGA) including 10 (5^{th}, 10^{th}, 15^{th}, 20^{th}, 25^{th}, 30^{th}, 35^{th}, 40^{th}, 45^{th}, 50^{th}) ESO designs *in each population*, ESO-assisted-GA-assisted-ESO (ESOaGAaESO) and GA assisted ESO (GAaESO) for a 12-node structure (20 generations and 20 designs in each population).

With respect to the results so far, GAaESO and ESOaGAaESO have the same behaviour while the generations increase, as they share the same concept of applying ESO to the final "unassisted" GA and ESOaGA designs correspondingly. It is worth mentioning that ESOaGA method is a more reliable method with much more potential compared to the others, especially when larger generations can be applied.

5.7. Kruskal-Wallis Test

The significance of the differences among the compared methods is now examined by the rigorous Kruskal-Wallis statistical test [126]. On the basis of statistical testing procedure, it is possible to check whether a method or case provides significantly better solutions than another.

The Kruskal-Wallis test is a nonparametric statistical test used to compare multiple samples. It is used to test the null hypothesis H_0 that all populations have identical distribution functions against the alternative hypothesis H_a that at least two of the samples differ only with respect to location.

Statistical significance test was carried out for all the comparisons of both 6-node and 12-node frameworks. The statement being tested is called the null hypothesis H_0. The test of significance is designed to assess the strength of the evidence against the null hypothesis. Usually the null hypothesis is a statement of "no difference". The probability, computed assuming that H_0 is true, that the test statistic would take a value as extreme or more extreme than that actually observed is called the P-value of the test. The smaller the P-value, the stronger is the evidence against the null hypothesis H_0 provided by the data. The decisive value of P is called the significance level. It is denoted by α and it is usually chosen to be α=0.05 (5%). If α=0.05 we are requiring that the data give evidence against H_0 so strong that it would happen no more than 5% of the time when H_0 is true. If the P-value is as small or smaller than α, we say that the data are statistically significant at level α. Here, the significance level chosen is α=0.05. In order for the null hypothesis H_0 to be rejected the P-value must be less than 0.05. We state that a pair of data has statistical significance if P<0.05 and a non statistical significance when P>0.05.

The quality indicator values from each method or case are compared using the Kruskal-Wallis test featured in Matlab in order to examine if there is statistical

significance among the studied optimization methods. The *P*-values obtained for each of the tables that compare the quality indicator results among different methods or cases are presented in Tables 5.43 & 5.44 for the 6-node and 12-node problem respectively.

When more than two algorithms are compared (as in Table 5.21 where there are 5), a matrix-valued input matrix is used to calculated the P-values.[1] So, for example, when the data in Table 5.21 are compared, a matrix-valued input of size 3x5 is created. When only two algorithms are compared, the size of this matrix is 3x2.

From the results presented in Table 5.43, we observe that the majority of *P*-values for the comparisons associated with the 6-node problem are higher than the significance level of 0.05. This means that, in almost all cases, the difference in the results of the methods that are compared in the previous ranking tables do not have statistical significance. The indicator values of the methods under comparison are very close for the 6-node problem. This does not mean that there is no difference among the output design sets of the developed methods applied to the 6-node structure, but that there is not sufficient evidence to reject the null hypothesis which is that the data sets are not different. However, there are few cases where the indicator results of the methods under comparison have statistical significance. As can be observed, the Tables 5.20, 5.21, 5.30 & 5.39 have *P*-values less than 0.05 (highlighted with bold). Consequently, these statistical test results enhance the conclusions that the method of ESOaGA of including ESO designs in each population performs poorly in comparison with the "unassisted" GA for small structures such as the 6-node framework. In addition, the Kruskal-Wallis test results suggest the superiority of the ESOaGA inserting ESO designs in the initial population against the ESOaGA of inserting ESO designs in each population and against the GAaESO method for the 6-node problem.

The superiority of the developed methods of ESOaGA and GAaESO is highly supported by the statistical test results for the larger example of the 12-node structure. Unlike the 6-node problem, the GA based methods applied to the 12-node problem are statistically significant. The *P*-values for most of the comparisons made for this

[1] The Matlab function 'p=krushkalwallis(X)' performs a Krushkal-Wallis test to compare samples from two or more groups. Each column of the *m*-by-*n* matrix *X* represents an independent sample containing *m* mutually independent observations.

case emphasize the difference in the results among the methods. In particular, the *P*-values (highlighted with bold) obtained using Kruskal-Wallis test (see Tables 5.24, 5.26, 5.33, 5.40 & 5.41) indicate that the ESOaGA method of including ESO designs in each population is indeed more efficient than the "unassisted" GA, the ESOaGA of inserting ESO designs in the initial population and the GAaESO method. Moreover, the corresponding *P*-values (highlighted with bold) of Tables 5.36 & 5.37 show that the GAaESO method is superior to the "unassisted" GA. The current statistical test results support the fact that the developed hybrid methods perform more efficiently for larger structural problems.

Table No.	*P*-value
5.2	0.8273
5.3	0.3012
5.4	0.7326
5.5	0.1266
5.6	0.7326
5.7	0.0665
5.8	0.0729
5.9	0.0691
5.10	0.1266
5.11	0.8273
5.18	0.2752
5.19	0.8273
5.20	**0.0495**
5.21	**0.0452**
5.22a	0.1495
5.22b	0.2752
5.30	**0.0495**
5.31	0.2752
5.34	0.8273
5.35	0.1266
5.38	0.1266
5.39	**0.0495**

Table 5.43: *P*-values for each of the comparisons among the quality indicators for the *6-node problem*. The table numbers refer to those previously discussed in this chapter.

Table No.	*P*-value
5.12	0.2181
5.13	0.1916
5.14	0.1301
5.23	0.2815
5.24	**0.0495**

5.25	0.0824
5.26	**0.0495**
5.32	0.2752
5.33	**0.0495**
5.36	**0.0369**
5.37	**0.0369**
5.40	**0.0369**
5.41	**0.0495**
5.42	0.4298

Table 5.44: *P*-values for each of the comparisons among the quality indicators for the *12-node problem*. The table numbers refer to those previously discussed in this chapter.

5.8. Conclusions

Two new methods have been developed and implemented in order to combine the quality of designs afforded by a GA and the computational economy of ESO. Visual comparisons and quantitative measures based on quality indicators have been used for performance assessment of the proposed methods.

The first method is termed ESO assisted GA (ESOaGA) and decreases the computational time of GA substantially. Including ESO generated designs in the GA population leads the GA to look at more promising areas close to the ESO trajectory and to access regions with superior designs.

Initially, poor GA generated designs were replaced by ESO designs in the initial population only of GA. The ESO assisted GA designs were found to dominate the designs of the "unassisted" GA. However, putting as many as possible ESO designs in the population, we would normally expect the best results, as GA would focus on the area of ESO trajectory much faster. Adding as many ESO designs in the initial population of GA as possible, does not mean that it will help the topology optimization process. The number of ESO designs needed to be inserted to the GA population should increase when the structural problem increases. As the structural size is increased, the solution space for GA is also increased and more assistance coming from the ESO designs is required.

Another ESOaGA approach of replacing poor GA designs with ESO designs in each GA population is studied. Various cases of inserting ESO designs in each population were considered and found to be superior to the "unassisted" GA only for the large problem of 12-node framework. This type of approach can be useful only if

large structures are considered as inserting ESO designs in each GA population has a negative effect on the GA diversity for small structures such as the 6-node case. Additionally, not all of the ESO designs are equally good to be added as the seed designs for GA. The quality of ESO designs has a significant impact on the ESOaGA approach of including ESO designs in the initial population. When ESO designs are inserted only in the initial population, the best selections should be made, in order for the GA to have the best possible start.

Including too many ESO designs influences the diversity of GA negatively. From the results so far, we observe that the ESOaGA method achieves high performance when a small fraction of the GA designs are replaced by ESO designs in initial or in each population. In this way, the existence of ESO designs in the GA populations, assists the GA search and at the same time does not influence the diversity of GA.

For small structures such as the 6-node framework the most efficient approach of ESOaGA is to insert ESO designs only in the initial GA population. On the other hand, for large structures such as the 12-node framework the most efficient approach of ESOaGA is to insert ESO designs in each population provided that a balance between the ESO and GA designs is maintained during the optimization process.

The second method termed as GA assisted ESO (GAaESO) works using GA for a limited number of generations followed by ESO runs for each GA design as the starting point. Instead of running the GA for a large number of generations, we can save substantial computational time by simply running an "unassisted" GA for a small number of generations and then applying the ESO method to the resulting designs.

Finally, comparison between the two new methods was made. ESOaGA method is superior to GAaESO for small structural problems. However, the GA assisted ESO method provides better results than the ESO assisted GA method under the same computational time when a large structural problem is considered. It loses its efficiency though when the size of generations is increased. However, both methods can obtain better optimized designs in less computational time than the "unassisted" GA method. The efficiency of ESOaGA method can be further improved by applying ESO to the results of ESOaGA. This approach (called ESOaGAaESO) can provide even better results than ESOaGA at roughly the same computational time, as ESO cost is minimal.

Chapter 6

Conclusions and future work

A summary of the conclusions given at the end of each chapter is presented here. After reporting the main contributions of this thesis, a section referring to several aspects of the framework topology optimization than would need further study is provided.

6.1. Concluding remarks

The quality of framework designs obtained by the use of the Evolutionary Structural Optimization (ESO) has been critically examined. It is often claimed that the method produces optimal structures because at each stage of evolution one discards the structurally most inefficient portions in a design. The consideration of weight alone does not provide an ideal metric for comparisons unless we fix the maximum stress in all the designs that are being compared to a prescribed value. We have, therefore, explored the two-objective problem of minimizing the weight and the maximum stress in a structure where comparisons can be readily made between the Pareto Fronts of various sets of designs.

A general trend of the trajectory of designs during the ESO process on the maximum stress-weight plane was obtained. In the initial phase of the optimization procedure, we observed that a substantial decrease in weight of the design is accompanied with a small decrease in the maximum stress of the design, reaching a critical point. After this point, the maximum stress of design increases dramatically while the weight of the design is marginally decreased. In order to reach this critical point, we obtain a significant decrease in weight (large material removal) at the same time as a reduction in the maximum stress value of the design. The continuation of the

optimization from this point will lead to a large increase in maximum stress with very small weight loss profit. Consequently, the design corresponding to this critical point can be considered the best design that we can obtain from the optimization when working with fixed cross sectional structures. Observations on the ESO trajectory can provide us with vital suggestions on selecting the stopping criterion for the ESO iterations and the most appropriate design according to the preferred constraints. Apart from that, scaling of the cross-sectional size of the structure was implemented at each iteration of the ESO process. Simultaneous sizing and topology framework optimization based on ESO can offer much more desirable results compared to the corresponding topology optimization.

Furthermore, the effectiveness of ESO on framework topology design was studied. The ESO designs were found, in many cases, to be dominated by those computed by an exhaustive search or a Genetic Algorithm (GA). Comparisons with the Pareto-optimal sets obtained from exhaustive search show that ESO does not always provide optimal solutions. However, it does produce some very good designs at a small computational expense. Discrete structures such as frames and trusses afford the opportunity to explore all possible designs—unlike a continuum model where such comparisons are not possible because of infinity of possible design options and because the conclusions depend on a specific parameterisation.

For complex topologies having a large number of joints, the number of design options cannot be exhaustively searched. Therefore, for such cases, a multi-objective GA has been used to produce Pareto-optimal sets and compared with the designs produced by ESO. The conclusions remain the same—ESO does not often produce Pareto-optimal designs; however, if one can afford only limited computational resources, then it produces some very good designs at relatively small cost. Finally, the topology optimization problem was reformulated as one of gradually reducing the thickness in a range—so that a structural member is not altogether removed in one step, but is reduced only when the thickness approaches zero (or a prescribed lower threshold). The general observations for this case are consistent with the other numerical experiments presented in this thesis—ESO does not often produce Pareto-optimal solutions, however, it affords some very good designs inexpensively.

Two new methods were also implemented in order to combine the efficiency of GA and the small computational cost of ESO. The first method (ESO assisted GA) decreases significantly the large computational cost of GA in the way it concentrates

on the designs surrounding the ESO trajectory. Including ESO designs in the GA population enriches the population with promising designs thus increasing the effectiveness of GA.

A second method of combining the ESO and GA was proposed (GA assisted ESO) in which the designs that are obtained by GA for a number of populations and generations were used as starting points for a family of ESO runs. Instead of running the GA for a large number of generations, we can save a lot of computational time by simply running the GA for a small number of generations followed by multiple runs of ESO each of which is computationally cheap.

After visual and quantitative comparisons among the new optimization methods for the two structural problems of 6-node and 12-node frameworks, we conclude that the performance of the hybrid optimization methods becomes even more efficient when large structures such the 12-node example are considered. This conclusion is supplied by the Kruskal-Wallis test of statistical significance. We also observed that for the same computational time, the GA assisted ESO (GAaESO) is relatively more efficient than the ESO assisted GA (ESOaGA) for larger structural problems. Both methods though can obtain better optimized designs in less computational time than the "unassisted" GA method. The efficiency of ESOaGA method can be improved by applying ESO to the results of ESOaGA. This new promising approach called ESOaGAaESO can provide even better results than ESOaGA in the same computational time, as ESO cost is considered marginal. However, its results are still dominated by GAaESO. In spite of that, further research could be focused on the effect of the iterative application of GA and ESO on the performance of ESOaGA and GAaESO.

To summarize, the major contributions of this thesis are listed below:

- Achievement of general shape of the trajectory in which designs evolve during the ESO process on the maximum stress-weight plane.
- Pareto-comparison between the ESO and Genetic Algorithm (GA) applied to frameworks.
- Sizing optimization using ESO in framework structures.
- Development of two new algorithms that blend the strengths of ESO and GA.

6.2. Future Work

Based on the results obtained from the current study, future work could concentrate on the following aspects:

- Modification of the objective function when a specified number of structural members are removed from the original fully connected design is necessary. A third objective -deflection- could be introduced, which is a measure of stiffness.
- The current work could be extended to problems including other constraints such as buckling. The applications of the approaches developed in this thesis could be investigated further if local buckling constraints are included in the topology optimization.
- The use of long members is not economical, and their removal results in structures of lower weight. On the other hand, the use of too many members in the ground structure increases the computational time and in the process of optimization leads to non-practical configurations. One of the most suitable forms for topology optimization is when every node is connected to the neighbouring nodes only. Avoid co-linear members. The use of fully connected structures as initial structures for genetic algorithms increases the length of the chromosomes, decreasing the speed of convergence.
- Movement of internal nodes may play an important role during the optimization. This approach could be used so that variation in performance can be checked.
- The Pareto Front concept presented in Chapter 4 and the two new methods proposed in Chapter 5 could be applied in 3-D frameworks. The method developed for 2-D structures can easily be extended immediately to structures of three dimensions. This only needs the generation of a three-dimensional mesh and the definition for each node of six degrees of freedom instead of three. However, the ground structures would contain a lot of nodes. Moreover, the simple fact of working in three dimensions will increase considerably the number of possible designs.

- A further study could be focused on the analytical approach and proof of the general results such as the obtained trends of ESO for the framework topology design.
- Further research is needed in order to investigate the effect of applying ESO and GA several times iteratively and the advantage of this over the ESOaGA and GAaESO methods.

Appendix

The input structure to NSGA2 used within the OptionsMatlab package is presented here. It describes the problem, and configures the design search and optimization. A number of optional fields may be used additionally to adjust the control parameters.

In this example, some important parameters required for running the "unassisted" GA are – the number of variables (15), the number of generations (20) and the number of designs in each population (20).

```
%--------------------------Run NSGA2-------------------------------

disp('*** Initiating NSGA2 ...');

% Create the input structure

s.OBJCON_FUNCTION='trussconfun';
s.NVRS=15;
s.LVARS=zeros(1,s.NVRS);
s.UVARS=ones(1,s.NVRS);
s.OBJECTIVE_NAMES={'MASS','MAXS'};
s.VARS=s.UVARS;
s.NOBJECTIVES=2;
s.NCONSTRAINTS=0;
s.GA_GEN=20;
s.GA_NPOP=20;


s=createNSGA2OptionsStruct(s);
truss_struct=createTrussStruct;

s.DVARS=truss_struct.DVARS;
s.NDVRS=truss_struct.NDVRS;

% Run NSGA2 with OptionsMatlab

disp('*** Results after each generation will be saved to
NSGA2OUTPUT.mat');
```

```
r=OptionsNSGA2(s);
disp('*** OptionsNSGA2 completed successfully');

title([num2str(size(r.PF,1)),' points on the pareto front for
function ',s.OPTCON]);
ylabel(s.CNAM(1));xlabel(s.CNAM(2));

disp('*** END OF trussnsga2.m ***');
```

The Matlab function `trussconfun` provides the values of the two objective functions. The penalties used in order to exclude the infeasible designs from the optimization process are also included in this function. The file `createTrussStruct` includes the definitions of all the parameters needed by OptionsMatlab. A brief description of the parameters that appear in the input stack is followed:

NVRS:	The number of design variables
VARS:	A vector of NVRS design variables corresponding to the initial design variables to be evaluated
LVARS:	A vector of length NVRS representing the lower limits to the design variable values
UVARS:	A vector of length NVRS representing the upper limits to the design variable values
NDVRS:	The maximum number of discrete design variable values
DVARS:	A matrix of size NVRS by NDVRS of the discrete design variable values
OBJECTIVE_NAMES:	Names of the objectives
NOBJECTIVES:	Number of objectives
NCONSTRAINTS:	Number of constraints
OBJCON_FUNCTION:	Objective function
GA_GEN:	Number of generations
GA_NPOP:	Number of designs included in each population

The meaning and default values of the OptionsMatlab [127] optional control parameters with respect to the genetic algorithm search used in the current thesis are presented in the table below.

Control Parameter	Meaning	Value
GA_NBIN	The number of bits used per variable in binary discretization	2
GA_NPOP	Population size each generation	20
GA_PENAL	Set the penalty function control parameter, r, with values less than one invoking the modified Fiacco and McCormick function (OPTIM2) otherwise the one pass method is used (OPTIM1)	1.00E+20
GA_PBEST	The proportion of the solutions that are used to form the parents of the next generation	0.8
GA_PCROSS	The proportion of the solutions in the population that are crossed to form new solutions	0.8
GA_PINVRT	The proportion of the solutions in the population that have their ordering codes inverted to form new solutions	0.2
GA_PMUTNT	Mutation is allowed at a level set by this parameter, i.e., this fraction of the total number of binary digits are reversed at each pass (n.b. greater than 0.5 results in randomisation)	0.005
GA_PRPTNL	If .TRUE. the make-up of the following generation is then biased in favour of the most successful according to their objective function values, otherwise survival is proportional to ranking but scaled to prevent dominance and stagnation	1 (.TRUE.)

GA_ALPHA	The cluster penalising function. Small values giving less severe penalties than those nearer one, and a value less than zero turning the mechanism off	0.2
GA_DMIN	The minimum distance between cluster centroids	0.05
GA_DMAX	The furthest distance a new solution can be from an existing cluster centroid without a new cluster being formed	0.2
GA_NCLUST	The initial number of clusters, either in absolute terms or, if it is < 1.0, as a fraction of the population size	0.1
GA_NBREED	Breeding is restricted to be between members of the same cluster if there are at least this many members in the cluster, otherwise random members of the population are used, if this parameter is set to zero breeding is not restricted, producing a greater variety of solutions but more "lethals"; if the number is less than unity it is taken to be a fraction of of the population, default 0.1, min.=2/GANPOP, max.=GA_NPOP.	0.1
GA_PSEED	Seeding of the initial, randomly generated members of the population is allowed at a level set by this parameter (0 = random, 1.0 clones of initial point)	0
GA_NRANDM	The number of random numbers drawn and discarded before starting the optimiser	0

References

[1] O. Sigmund, Topology optimization: a tool for the tailoring of structures and materials, A special issue of the Philosophical Transactions of the Royal Society: Science into the next Millennium (Issue III, Mathematics, Physics and Engineering), Vol. 358, Issue 1765, pp. 211-228, 2000.

[2] J. Sokolowski and A. Zochowski, On the topological derivative in shape optimisation, SIAM Journal on Control and Optimization, Vol. 37, No. 4, pp. 1251-1272, 1999.

[3] A. G. M Michell, The limits of economy of material in frame-structures, Philosophical Magazine, Vol. 8, pp. 305-316, 1904.

[4] G. I. N. Rozvany and W. Prager, A new class of structural optimization problems: Optimal archgrids, Computer Methods in Applied Mechanics and Engineering, Vol. 19, Issue 1, pp. 127-150, 1979.

[5] H. A. Eschenauer and N. Olhoff, Topology optimisation of continuum structures: A review, Applied Mechanics Reviews, Vol. 54, No. 4, pp. 331-390, 2001.

[6] K. Saitou, K. Izui, S. Nishiwaki and P. Papalambros, A Survey of Structural Optimization in Mechanical Product Development, Journal of Computing and Information Science in Engineering, Vol. 5, Issue 3, pp. 214-226, 2005.

[7] U. Schramm, Optimization Technology in Design – Trends, Direction, Gaps, 6th World Congress of Structural and Multidisciplinary Optimization, Rio de Janeiro, Brazil, 2005.

[8] L. Krog, A. Tucker and G. Rollema, Application of Topology, Sizing and Shape Optimization Methods to Optimal Design of Aircraft Components, Airbus, 2003.

[9] M. Bendsøe, E. Lund and O. Sigmund, Topology Optimization – broadening the areas of application, Control and Cybernetics, Vol. 34, Issue 1, pp. 7-35, 2005.

[10] H. Fredricson, Structural topology optimisation: an application review, International Journal of Vehicle Design, Vol. 37, No. 1, pp. 67-80, 2005.

[11] H. Fredricson, T. Johansen, A. Klarbring and J. Petersson, Topology optimization of frame structures with flexible joints, Structural and Multidisciplinary Optimization, Vol. 25, pp. 199-214, 2003.

[12] C. Reed, Applications of Optistruct optimization to body in white design, Jaguar, Altair Engineering, 2002.

[13] W. D. Nadir, Multidisciplinary Structural Design and Optimization for Performance, Cost and Flexibility, Master of Science Dissertation, M.I.T., 2005.

[14] D. N. Chu, Y. M. Xie, A. Hira and G. P. Steven, Evolutionary structural optimization for problems with stiffness constraints, Finite Elements in Analysis and Design, Vol. 21, pp. 239-251, 1996.

[15] K. A. Proos, G. P. Steven, O. M. Querin and Y. M. Xie, Multicriterion evolutionary structural optimisation using weighting and the global criterion methods, American Institute of Aeronautics and Astronautics Journal, Vol. 39, Issue 10, pp. 2006-2012, 2001.

[16] G. P. Steven, O. M. Querin and Y. M. Xie, Evolutionary structural optimisation (ESO) for combined topology and size optimisation of discrete structures, Computer Methods in Applied Mechanics and Engineering, Vol. 188, pp. 743-754, 2000.

[17] D. N. Chu, Y. M. Xie, G. P. Steven, An evolutionary structural optimisation method for sizing problems with discrete design variables, Computers and Structures, Vol. 68, pp. 419-431, 1998.

[18] Y. M. Xie and G. P. Steven, Evolutionary structural optimisation for dynamic problems, Computers and Structures, Vol. 58, No. 6, pp. 1067-1073, 1996.

[19] H. Kim, O. M. Querin, G. P. Steven and Y. M. Xie, Improving efficiency of evolutionary structural optimisation by implementing fixed grid mesh, Structural and Multidisciplinary Optimization, Vol. 24, pp. 441-448, 2003.

[20] D. N. Chu, Y. M. Xie, A. Hira and G. P. Steven, On various aspects of evolutionary structural optimisation for problems with stiffness constraints, Finite Elements in Analysis and Design, Vol. 24, pp. 197-212, 1997.

[21] Y. M. Xie and G. P. Steven, A simple evolutionary procedure for structural optimisation, Computers and Structures, Vol. 49, pp. 885-896, 1993.

[22] D. N. Chu, Y. M. Xie and G. P. Steven, An evolutionary method for optimal design of plates with discrete variable thicknesses subject to constant weight, Structural Optimisation, Vol. 17, pp. 55-64, 1999.

[23] Y. M. Xie and G. P. Steven, Evolutionary Structural Optimisation, Springer, 1997.

[24] Q. Li, O. M. Querin, G. P. Steven and Y. M. Xie, A simple checkerboard suppression algorithm for evolutionary structural optimization, Structural and Multidisciplinary Optimization, Vol. 22, pp. 230-239, 2001.

[25] X. Y. Yang, Y. M. Xie, G. P. Steven and O. M. Querin, Topology optimisation for frequencies using an evolutionary method, Journal of Structural Engineering, Vol. 125, Issue 12, pp. 1432-1438, 1999.

[26] Y. M. Xie and G. P. Steven, A simple approach to structural frequency optimisation, Computers and Structures, Vol. 53, Issue 6, pp. 1487-1491, 1994.

[27] Y. M. Xie, G. P. Steven and Q. Li, On equivalence between stress criterion and stiffness criterion in evolutionary structural optimisation, Structural and Multidisciplinary Optimization, Vol. 18, No. 1, pp. 67-73, 1999.

[28] Y. M. Xie and G. P. Steven, Optimal design of multiple load case structures using an evolutionary procedure, Engineering Computations, Vol. 11, pp. 295-302, 1994.

[29] G. Cheng, Some aspects of truss topology optimisation, Structural Optimization, Vol. 10, pp. 173-179, 1995.

[30] U. Kirsch, Optimal topologies of truss structures, Applied Mechanics Reviews, Vol. 42, pp. 223-239, 1989.

[31] U. Kirsch and B. H. V. Topping, Minimum weight design of structural topologies, Journal of Structural Engineering – ASCE, Vol. 118, Issue 7, pp. 1770-1785, 1992.

[32] J. P. Leiva, B. C. Watson and I. Kosaka, Modern Structural Optimization Concepts Applied to Topology Optimization, Proceedings of the 40th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Material Conference, St. Louis, MO, pp 1589-1596, 1999.

[33] G. Vanderplaats, Thirty years of modern structural optimization, Advances in Engineering Software, Vol. 16, pp. 81-88, 1993.

[34] G. Cheng and N. Olhoff, An investigation concerning optimal design of solid elastic plates, International Journal of Solids and Structures, Vol. 17, pp. 305-323, 1981.

[35] A. Diaz and M. P. Bendsøe, Shape optimisation of structures or multiple loading situations using a homogenization method, Structural Optimization, Vol. 4, pp. 17-22, 1992.

[36] R. V. Kohn and G. Strang, Optimal design in elasticity and plasticity, International Journal for Numerical Methods in Engineering, Vol. 22, pp. 183-188, 1986.

[37] M. P. Bendsøe and N. Kikuchi, Generating optimal topologies in structural design using a homogenisation method, Computer Methods in Applied Mechanics and Engineering, Vol. 71, pp. 197-224, 1988.

[38] S. Nishiwaki, M. I. Frecker, S. Min and N. Kikuchi, Topology optimisation of compliant mechanisms using the homogenisation method, International Journal for Numerical Methods in Engineering, Vol. 42, pp. 535-559, 1998.

[39] A. R. Diaz and N. Kikuchi, Solutions to shape and topology eigenvalue optimisation problems using a homogenisation method, International Journal for Numerical Methods in Engineering, Vol. 35, pp. 1487-1502, 1992.

[40] K. Suzuki and N. Kikuchi, A homogenisation method for shape and topology optimisation, Computer Methods in Applied Mechanics and Engineering, Vol. 93, pp. 291-318, 1991.

[41] M. P. Bendsøe, Recent developments in topology design of materials and mechanisms, ESAIM: Proceedings, Vol. 11, pp. 41-60, 2002.

[42] M. P. Bendsøe, Optimal shape design as a material distribution problem, Structural Optimization, Vol. 1, pp. 193-202, 1989.

[43] N. Inou, N. Shimotai and T. A. Uesugi, Cellular automaton generation topological structures, In: Proc of the 2nd European Conference on Smart Structures and Materials, Glasgow, UK, pp. 47-50, 1994.

[44] A. Baumgartner, L. Harzheim and C. Mattheck, SKO: Soft Kill Option. The biological way to find optimum structure topology, International Journal of Fatigue, Vol. 14, pp. 387-393, 1992.

[45] Y. L. Hsu, M. S. Hsu and C. T. Chen, Interpreting results from topology optimization using density contours, Computers and Structures, Vol. 79, pp. 1049-1058, 2001.

[46] K. Maute, and E. Ramm, Adaptive topology optimisation, Structural Optimization, Vol. 10, pp. 100-112, 1995.

[47] N. Olhoff, M. P. Bendsøe and J. Rasmussen, On CAD-integrated structural topology and design optimisation, Computer Methods in Applied Mechanics and Engineering, Vol. 89, pp. 259-279, 1991.

[48] R. J. Yang and C. J. Chen, Stress-based topology optimisation, Structural and Multidisciplinary Optimization, Vol. 12, pp. 98-105, 1996.

[49] W. Achtziger, M. P. Bendsøe, A. Ben-Tal and J. Zowe, Equivalent displacement-based formulations for maximum strength truss topology design, Impact of Computing in Science and Engineering, Vol. 4, pp. 315-345, 1992.

[50] M. Bendsøe, A. Ben-Tal and J. Zowe, Optimisation methods for truss geometry and topology design, Structural Optimization, Vol. 7, pp. 141-159, 1994.

[51] A. Ben-Tal and M. P. Bendsøe, A new method for optimal truss topology design, SIAM Journal on Optimization, Vol. 3, pp. 322-358, 1993.

[52] G. Rozvany, M. P. Bendsøe and U. Kirsch, Layout optimization of structures, Applied Mechanics and Reviews, Vol. 48, pp. 41-119, 1995.

[53] M. P. Bendsøe, Optimisation of Structural Topology, Shape and Material, Springer, Berlin, 1995.

[54] M. P. Bendsøe and O. Sigmund, Topology Optimization: Theory, Methods and Applications, Springer, New York, 2003.

[55] N. Olhoff and J. E. Taylor, On structural optimisation, Journal of Applied Mechanics, Vol. 50, pp. 1134-1151, 1983.

[56] J. C. Maxwell, in Niven, W. D. (Ed), The Scientific Papers, Cambridge University Press, Cambridge, 1890.

[57] W. S. Dorn, R. E. Gomory and H. J. Greenberg, Automatic design of optimal structures, J. de Mechanique, Vol. 3, pp. 25-52, 1964.

[58] M. W. Dobbs and L. P. Felton, Optimization of truss geometry, ASCE Journal of Structural Division, Vol. 95, pp. 2105-2118, 1969.

[59] U. T. Ringertz, On topology optimisation of trusses, Engineering Optimization, Vol. 9, pp 209-218, 1985.

[60] A. J. Keane and P. B. Nair, Computational Approaches for Aerospace Design, John Wiley and Sons, 2005.

[61] M. Ohsaki and C. C. Swan, Topology and geometry optimization of trusses and frames, ASCE/SEI State-of-Art-Rep. on Structural Optimization, S. A. Burns, ed., New York, 2002.

[62] G. I. N. Rozvany, A note on truss design for stress and displacement constraints by optimality criteria methods, Structural Optimization, Vol. 3, No. 1, pp. 45-50, 1991.

[63] G. I. N. Rozvany, Stress ratio and compliance based methods in topology optimisation – a critical review, Structural and Multidisciplinary Optimization, Vol. 21, pp. 109-119, 2001.

[64] G. I. N. Rozvany, Topological optimization of grillages – past controversies and new directions, International Journal of Mechanical Sciences, Vol. 36, pp. 495-512, 1994.

[65] G. I. N. Rozvany, Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics, Structural and Multidisciplinary Optimization, Vol. 21, Issue 2, pp. 90-108, 2001.

[66] G. I. N. Rozvany, M. Zhou and T. Birker, Generalized shape optimization without homogenization, Structural Optimization, Vol. 4, pp. 250-254, 1992.

[67] U. Kirsch, On the relationship between optimum structural topologies and geometries, Structural Optimization, Vol. 2, pp. 39-45, 1990.

[68] B. H. V. Topping, Shape Optimisation of Skeletal Structures: A review, Journal of Structural Engineering, ASCE, Vol. 109, Issue 8, pp. 1933-1951, 1983.

[69] S. Sankaranarayanan, R. T. Haftka and R. K. Kapania, Truss topology optimization with simultaneous analysis and design, AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 33rd, Dallas, Texas; United States, pp. 2576-2585, 1992.

[70] P. Y. Shim and S. Manoochehri, Generating optimal configurations in structural design using simulated annealing, International Journal for Numerical Methods in Engineering, Vol. 40, pp. 1053-1069, 1997.

[71] K. Shea and J. Cagan, Languages and Semantics of Grammatical Discrete Structures, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, Vol. 13, pp. 241-251, 1999.

[72] K. Shea and J. Cagan, The design of novel roof trusses with shape annealing: assessing the ability of a computational method in aiding structural designers with varying design intent, Design Studies, Vol. 20, pp. 3-23, 1999.

[73] B. Baumann and B. Kost, Topology Optimization – Random Cost Method versus Evolutionary Algorithms, Computation Optimization and Applications, Vol. 14, No. 2, pp. 203-218, 1999.

[74] J. Rodriguez-Velazquez and A. A. Seireg, Optimizing the shapes of structures via a rule-based computer program, Computers in Mechanical Engineering, Vol. 4, Issue 1, pp. 20-28, 1985.

[75] E. Hinton, and J. Sienz, Fully stressed topological design of structures using an evolutionary procedure, Engineering Computations, Vol. 12, pp. 229-244, 1993.

[76] P. Tanskanen, The evolutionary structural optimisation method: theoretical aspects, Computer Methods in Applied Mechanics and Engineering, Vol. 191, pp. 5485-5498, 2002.

[77] B. Xu, J. Jiang, W. Tong and K. Wu, Topology group concept for truss topology optimisation with frequency constraints, Journal of Sound and Vibration, Vol. 261, pp. 911-925, 2003.

[78] R. V. Grandhi, Structural optimisation with frequency constraints – a review, American Institute of Aeronautics and Astronautics Journal, Vol. 31, pp. 2296-2303, 1993.

[79] O. M. Querin, G. P. Steven and Y. M. Xie, Evolutionary structural optimization (ESO) using bidirectional algorithm, Engineering Computation, Vol. 15, pp. 1031-1048, 1998.

[80] V. Young, O. M. Querin and G. P. Steven, 3D and multiple load case bi-directional evolutionary structural optimization (BESO), Structural and Multidisciplinary Optimization, Vol. 18, pp. 183-192, 1999.

[81] S. Y. Han, T. H. Lee and J. K. Lim, A study on efficiency improvement of evolutionary structural optimisation, Key Engineering Materials, Vols. 183-187, pp. 379-384, 2000.

[82] J. S. Liu, G. T. Parks and P. J. Clarkson, Metamorphic development: A new topology optimisation method for continuum structures, Structural and Multidisciplinary Optimization, Vol. 20, pp. 288-300, 2000.

[83] E. Zitzler and L. Thiele, Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, IEEE Transactions on Evolutionary Computation, Vol. 3, No. 4, pp. 257-271, 1999.

[84] D. A. Van Veldhuizen and G. B. Lamont, Evolutionary Computation and Convergence to a Pareto Front, Late Breaking Papers at the Genetic Programming

1998 Conference, edited by J. R. Koza. Stanford, CA: Stanford University Bookstore, pp. 221-228, 1998.

[85] C. A. Coello Coello, A Comprehensive Survey of Evolutionary-Based Multiobjective Optimisation Techniques, Knowledge and Information Systems, Vol. 1, Issue 3, pp. 269-308, 1999.

[86] C. M. Fonseca and P. J. Fleming, An overview of Evolutionary Algorithms in Multiobjective Optimisation, Evolutionary Computation, Vol. 3, pp. 1-16, 1995.

[87] C. A. Coello Coello and A. D. Christiansen, MOSES: A Multiobjective Optimisation Tool for Engineering Design, Engineering Optimization, Vol. 31, Issue 3, pp. 337-368, 1999.

[88] V. Pareto, Cours d' Economie Politique, F. Rouge & Cie., Lausanne, Switzerland, Vol. 1, 1896.

[89] C. Kane and M. Schoenauer, Topological optimum design using genetic algorithms, Control and Cybernetics, Vol. 25, Issue 5, pp. 1059-1088, 1996.

[90] H. Kawamura, H. Ohmori and N. Kito, Truss topology optimisation by a modified genetic algorithm, Structural and Multidisciplinary Optimization, Vol. 23, pp. 467-472, 2002.

[91] I. Rechenberg, Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog Verlag, Stuttgart, 1973.

[92] J. H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan (1975) and MIT Press: Cambridge, MA, 1992.

[93] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, Cambridge, MA: The MIT Press, 1992.

[94] J. S. Gero and S. J. Louis, Improving Pareto Optimal Designs Using Genetic Algorithms, Microcomputers in Civil Engineering, Vol. 10, Issue 4, pp. 241-249, 1995.

[95] K. W. Park and D. E. Grierson, Pareto-Optimal Conceptual Design of the Structural Layout of Building Using a Multicriteria Genetic Algorithm, Computer-Aided Civil and Infrastructure Engineering, Vol. 14, pp. 163-170, 1999.

[96] F. Xue, A. C. Sanderson and R. J. Graves, Pareto-based multi-objective differential evolution, In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003), Canberra, Australia, IEEE Press, Vol. 2, pp. 862-869, 2003.

[97] K. Deb and S. Gulati, Design of truss-structures for minimum weight using genetic algorithms, KanGAL Report No. 99001, 1999.

[98] K. Deb, An Introduction to Genetic Algorithms, In: Sadhana, Vol. 24, Issue 4, pp. 205-230, 1999.

[99] K. Deb, S. Gulati and S. Chakrabarti, Optimal Truss-Structure Design using Real-Coded Genetic Algorithms, Genetic Programming: Proceedings of the Third Annual Conference, 1998.

[100] S. J. Louis and F. Zhao, Domain Knowledge for Genetic Algorithms, International Journal of Expert Systems Research and Applications, Vol. 8, Issue 3, pp. 195-212, 1995.

[101] S. D. Rajan, Sizing, Shape, and Topology Design Optimization of Trusses Using Genetic Algorithm, Journal of Structural Engineering, Vol. 121, Issue 10, pp. 1480-1487, 1995.

[102] W. M. Jenkins, Plane Frame Optimum Design Environment Based on Genetic Algorithm, Journal of Structural Engineering, Vol. 118, No. 11, pp. 3103-3112, 1992.

[103] W. M. Jenkins, Towards structural optimization via the genetic algorithm, Computers and Structures, Vol. 40, No. 5, pp. 1321-1327, 1991.

[104] S. Rajeev and C. S. Krishnamoorthy, Genetic Algorithms-Based Methodologies for Design Optimization of Trusses, Journal of Structural Engineering, Vol. 123, Issue 3, pp. 350-358, 1997.

[105] A. Kaveh and V. Kalatjari, Topology Optimization of trusses using genetic algorithm, force method and graph theory, International Journal for Numerical Methods in Engineering, Vol. 58, pp. 771-791, 2003.

[106] D. E. Goldberg and M. P. Samtani, Engineering Optimization via Genetic Algorithm, Proc. 9th Conf. on Electronic Computation, ASCE, New York, NY, pp. 471-484, 1986.

[107] P. Hajela and E. Lee, Genetic Algorithms in Truss Topological Optimization, International Journal of Solids and Structures, Vol. 32, No. 22, pp. 3341-3357, 1995.

[108] S. Rajeev and C. Krishnamoorthy, Discrete optimization of structures using genetic algorithms, Journal of Structural Engineering, Vol. 118, No. 5, pp. 1233-1250, 1992.

[109] C. Y. Lin and P. Hajela, Genetic algorithms in optimization problems with discrete and integer design variables, Engineering Optimization, Vol. 19, pp. 309-327, 1992.

[110] C. A. Coello Coello, Discrete Optimization of Trusses Using Genetic Algorithms, EXPERSYS-94, I.I.T.T., pp. 331-336, 1994.

[111] M. Galante, Genetic Algorithms as an Approach to Optimize Real-World Trusses, International Journal for Numerical Methods in Engineering, Vol. 39, pp. 361-382, 1996.

[112] M. R. Ghasemi, E. Hinton and R. D. Wood, Optimization of Trusses using Genetic Algorithms for Discrete and Continuous variables, Engineering Computations, Vol. 3, pp. 272-301, 1999.

[113] D. E. Grierson and W. H. Pak, Optimal Sizing, Geometrical and Topological Design Using Genetic Algorithms, Structural Optimization, Vol. 6, pp. 151-159, 1993.

[114] M. J. Jakiela, C. Chapman, J. Duda, A. Adewuya and K. Saitou, Continuum structural topology design with genetic algorithms, Computer Methods in Applied Mechanics and Engineering, Vol. 186, pp. 339-356, 2000.

[115] S. Y. Wang and K. Tai, Structural topology design optimization using Genetic Algorithms with a bit-array representation, Computer Methods in Applied Mechanics and Engineering, Vol. 194, pp. 3749-3770, 2005.

[116] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation, Vol. 7, pp. 117-132, 2003.

[117] V. Grunert da Fonseca, C. M. Fonseca, and A. O. Hall, Inferential performance assessment of stochastic optimizers and the attainment function, in Evolutionary Multi-Criterion Optimization (EMO2001), E. Zitzler *et al.*, Eds. Berlin, Germany: Springer, pp. 213-225, 2001.

[118] D. A. Van Veldhuizen and G. B. Lamont, On measuring multiobjective evolutionary algorithm performance, in Congress on Evolutionary Computation (CEC2000), A. Zalzala and R. Eberhart, Eds. Piscataway, NJ:IEEE Press, Vol. 1, pp. 204-211, 2000.

[119] J. Knowles, L. Thiele, and E. Zitzler, A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers, TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.

[120] C. M. Fonseca, J. D. Knowles, L. Thiele, and E. Zitzler, A tutorial on the performance assessment of stochastic multiobjective optimizers, Presented at the

Third International Conference on Evolutionary Multi-Criterion Optimization (EMO), 2005.

[121] M. P. Hansen and A. Jaszkiewicz, Evaluating the quality of approximations to the non-dominated set, Technical Report IMM-REP-1998-7, Technical University of Denmark, 1998.

[122] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 2, pp. 181-197, 2002.

[123] N. Srinivas and K. Deb, Multiobjective Optimisation Using Nondominated Sorting in Genetic Algorithms, Journal of Evolutionary Computation, Vol. 2, No. 3, pp. 221-248, 1995.

[124] I. I. Voutchkov and A. J. Keane, Multiobjective optimization using surrogates, in: Proc. 7th Int. Conf. Adaptive Computing in Design and Manufacture, Bristol, UK, pp. 167-175, 2006.

[125] S. Bleuler, M. Laumanns, L. Thiele and E. Zitzler, PISA—A Platform and Programming Language Independent Interface for Search Algorithms, TIK Report 154, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2003

[126] J. Conover, Practical Nonparametric statistics (Third ed.), New York, NY: John Wiley and Sons, 1999.

[127] G. Pound and A. Price, Geodise OptionsMatlab Tutorial – A User's Guide, The Geodise Project, University of Southampton, pp. 1-130, 2007.