Genetic algorithm based discrete optimum design of geometrically non-linear plane and space trusses

Khalil Sabaghalvani
University of Wollongong

Recommended Citation

Unless otherwise indicated, the views expressed in this thesis are those of the author and do not necessarily represent the views of the University of Wollongong.
GENETIC ALGORITHM BASED DISCRETE OPTIMUM DESIGN OF GEOMETRICALLY NON-LINEAR PLANE AND SPACE TRUSSES

by

KHALIL SABAGHALVANI

A thesis submitted in partial fulfilment of the requirements for the award of the degree of

MASTER OF ENGINEERING (HONOURS)

from

UNIVERSITY OF WOLLONGONG

FACULTY OF ENGINEERING

2002
Declaration

The work described in this dissertation was carried out in the Faculty of Engineering University of Wollongong from July 2000 to March 2002, under the Supervision of Dr. Muhammad N.S Hadi.

The author wishes to declare that, except for commonly understood ideas, or where specified reference is made to work of others, the work reported in this dissertation is his own and includes nothing is the outcome of work done in collaboration. It has not been submitted, in part or in whole, to any other university for any degree diploma or other qualification.

Khalil Sabaghialvani

February 2002
Acknowledgments

I wish to express my sincere gratitude to my advisor, Dr. M. N. S. Hadi, who has always given me guidance, assistance and support since I first came to Wollongong University. I admire him as a teacher, as a leader, and, above all, as a friend. He provided an excellent research atmosphere that was very helpful in the successful completion of this work. I also want to express my gratitude to Prof. Najdat Aziz for his social and spiritual support.

I would like to thank the staff members of the Faculty of Engineering, University of Wollongong, for their assistance during the undertaking of this research program.

I want to thank my friends especially Mahmed Aktash for their support and friendship. Finally, I wish to express my special thanks to my family for their support and encouragement during the course of this work and to my sons, Araz and Anar, for their understanding and patience during my study at the University of Wollongong.
Abstract

The field of optimal design of engineering structures has been revolutionised by the rapid development of computer technology in the past few decades, which has enabled the design and analysis to be achieved with much greater speed and accuracy than ever before.

A number of design methodologies have been developed and are in use for the optimum design of structural systems. In the last decade, the use of evolutionary algorithms, especially genetic algorithms (GAs) to optimise the design of structures has received much research attention mainly because of their simplicity, global perspective, and inherent parallel processing. Furthermore, in GA the gradient of the objective function and the constraint functions are not needed to find optimal solutions. Therefore GAs have capability to handle any design problems that may involve non-differentiable objective function and/or a combination of continuous, discrete, and integer design parameters.

Many research studies have reported the solution of truss structure optimisation problems through GAs in recent years. In all the recent research studies only linear analysis was considered to determine the response of the structure to the applied load in the optimisation process using GA. The linear analysis for some structures such as long-span and slender structures may not be applicable because of the geometric non-linear behaviour of the structure, which may be due to the presence of large deflection. These structures require non-linear analysis to obtain their behaviour and response prediction under the external loading, thus it has become mandatory to carry out geometric non-linear analysis of long-span and slender structures such as suspension bridges.

In the present study a GA-based methodology is presented for the optimum design
of structures with geometric non-linear behaviour. Attention is focused on plane and space truss structures under a range of different loading conditions. The cross sectional areas of the truss members are considered as continuous or discrete design variables. The total weight or volume of the structure is considered as the objective function with the constraints as limitation on the member stresses, buckling and nodal displacements.

In order to undertake a comparison with the optimum design obtained for plane and space truss structures based on a non-linear analysis, an optimum design algorithm is also developed for plane and space truss structures based on linear analysis.

In the analytical part of the study, the stiffness method was considered for the analysis of plane and space truss structures. Four programs based on linear and geometrically non-linear analyses were developed, verified and used. The non-linear response of trusses is obtained by utilising a Newton-Raphson type iteration technique.

Numerical design problems are presented to illustrate the efficiency of the proposed algorithms and to study the effect of various parameters such as the type of analysis (linear or geometrically non-linear) and member buckling. For the cases studied, the results show that the proposed algorithms are effective and reliable to solve discrete optimisation truss structure problems and it is possible to obtain optimum solutions by including geometric non-linear analysis in the design of these structures.

It was found that in geometric non-linear analysis used in the proposed methodology a population size as small as 30 produces optimum results as compared to a large population size required for a linear analysis. A very accurate result can be obtained using geometric non-linear analysis to optimise the truss structure problem, but at the expense of time, which was found to be more than 3 times the time required by the linear analysis. The algorithm developed in the current study can be applied as an optimisation module making it free from gradient information contrary to classic
optimisation techniques. It can take discrete as well as continuous design variables into account and can be applied to any discrete set of section produced according to different standards.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declaration</td>
<td>i</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>ii</td>
</tr>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>Contents</td>
<td>vi</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xii</td>
</tr>
<tr>
<td>Notations</td>
<td>xiv</td>
</tr>
</tbody>
</table>

### CHAPTER 1: INTRODUCTION

1.1 General introduction .............................................. 1
1.2 Aim of present study ............................................. 4
1.3 Outline of thesis .................................................. 4

### CHAPTER 2: REVIEW OF THE OPTIMISATION TECHNIQUES

2.1 Introduction ......................................................... 6
2.2 General Formulation of Optimal Design ............................ 7
   2.2.1 General problem formulation ................................. 7
   2.2.2 Design variables .............................................. 8
     2.2.2.1 Sizing design variables ......................... 9
     2.2.2.2 Shape design variables ......................... 9
     2.2.2.3 Topological design variables ................. 10
   2.2.3 Constraints ................................................. 10
   2.2.4 Objective function ..................................... 11
2.3 Optimisation Techniques ........................................................................ H

2.3.1 Background ........................................................................... 12

2.3.2 Gradient-based techniques ................................................... 14
  2.3.2.1 Mathematical programming ........................................ 15
  2.3.2.2 Optimality criteria ................................................ 19

2.3.3 Evolutionary algorithms ...................................................... 21
  2.3.3.1 Genetic algorithm ................................................ 22

2.4 Concluding Remarks ........................................................................ 23

CHAPTER 3. OVERVIEW OF THE GENETIC ALGORITHMS

3.1 Introduction ...................................................................................... 26

3.2 Genetic Algorithms .......................................................................... 27

3.3 How Genetic Algorithms Work ....................................................... 28
  3.3.1 Encoding the design variables ............................................. 30
  3.3.2 Selection procedure .............................................................. 33
  3.3.3 Mating/Crossover operator .................................................. 36
  3.3.4 Mutation operator ................................................................ 38

3.4 Concluding Remarks ........................................................................ 40

CHAPTER 4. OPTIMAL DESIGN OF TRUSS STRUCTURES

4.1 Introduction ...................................................................................... 42

4.2 Design of Truss Structures ............................................................. 43

4.3 Types of Structural Optimisation Problems ................................... 43
  4.3.1 Size optimisation of trusses .................................................. 44
  4.3.2 Shape optimisation of trusses ................................................ 45
4.3.3 Topology optimisation of trusses ........................................ 47
4.4 Optimal Design of Trusses using Genetic Algorithms ............... 49
4.5 Concluding Remarks ............................................................... 54

CHAPTER 5. DEVELOPED OPTIMISATION TECHNIQUE

5.1 Introduction ............................................................................. 56
5.2 Truss Structure Analysis ........................................................... 57
5.3 Linear analysis of trusses ......................................................... 58
  5.3.1 Plane truss ................................................................. 59
  5.3.2 Space truss ................................................................. 63
  5.3.3 Calculation steps ........................................................... 66
5.4 Non-linear analysis of trusses .................................................. 66
  5.4.1 Geometrically non-linear analysis ..................................... 67
  5.4.2 Plane truss ................................................................. 68
  5.4.3 Space truss ................................................................. 72
  5.4.4 An iterative solution (the Newton-Raphson method) ........... 73
  5.4.5 Calculation steps ........................................................... 76
5.5 Buckling analysis ................................................................. 76
5.6 Design Problem Formulation .................................................. 78
5.7 Penalty Function .................................................................... 80
5.8 Fitness Evaluation ................................................................... 83
5.9 Profile List of Discrete Design Variables ................................... 84
5.10 Member Grouping ................................................................. 85
5.11 Structure of Proposed Design Procedure ................................. 85
5.12 MATLAB Program ............................................................... 89
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>Source Code for the Linear Analysis of Space Truss</td>
<td>154</td>
</tr>
<tr>
<td>D</td>
<td>Source Code for the Geometrical Non-linear Analysis of Plane Truss</td>
<td>159</td>
</tr>
<tr>
<td>E</td>
<td>Source Code for the Geometrical Non-linear Analysis of Space Truss</td>
<td>164</td>
</tr>
<tr>
<td>F</td>
<td>Profile List of Discrete Design Variables</td>
<td>169</td>
</tr>
<tr>
<td>G</td>
<td>Stresses and displacements obtained for the final solutions of all solved examples</td>
<td>172</td>
</tr>
</tbody>
</table>
List of Figures

Fig. 2.1. The linearised problem 16
Fig. 2.2. Under constrained problem and move limits 17
Fig. 3.1. The process of a genetic algorithm 29
Fig. 3.2. Example of binary string representation 31
Fig. 3.3. Roulette wheel selection 35
Fig. 3.4. One-point crossover 38
Fig. 3.5. Mutation procedure 39
Fig. 4.1. Typical ground structure 48
Fig. 5.1. A plane truss structural element 60
Fig. 5.2. A space truss structural element 64
Fig. 5.3. Large displacements on a truss structural element (a-b) 69
Fig. 5.4. Newton-Raphson iteration for non-linear analysis 74
Fig. 5.5. Schematic flow chart of the proposed algorithm 86
Fig. 6.1. Geometry and loading of a 10-bar cantilever truss 93
Fig. 6.2. Deformed shape of 10-bar cantilever truss 93
Fig. 6.3. Generation history for 10-bar cantilever truss 95
Fig. 6.4. Geometry and loading of 20-bar bridge-type truss 97
Fig. 6.5. Generation history for 20-bar bridge-type truss 99
Fig. 6.6. Geometry and loading of 51-bar roof truss 101
Fig. 6.7. 25-bar transmission tower space truss 105
Fig. 6.8. 52-bar dome space truss 108
Fig. 6.9. 56-bar transmission tower space truss 113
Fig. 6.10. 120-bar dome space truss 117
List of Tables

Table 3.1. Details of computations- first generation 34
Table 3.2. Results of selection 36
Table 3.3. Results after crossover 38
Table 3.4. Final results after the evolution of the initial population 40
Table 6.1. Definition of the 10-bar cantilever truss 94
Table 6.2. Comparison of the results for the 10-bar cantilever truss 96
Table 6.3. Comparison of the results for the 10-bar cantilever truss 96
Table 6.4. Definition of the 20-bar bridge-type truss 98
Table 6.5. Member grouping detail and result of optimisation for 20-bar truss 100
Table 6.6. Definition of the 51-bar roof truss 101
Table 6.7. Comparison of the results for the 51-bar roof truss 103
Table 6.8. Discrete solution for the 51-bar roof truss 103
Table 6.9. Loading details for the 25-bar space truss 105
Table 6.10. Group membership for the 25-bar space truss 105
Table 6.11. Coordinates of the joints of the 25-bar space truss 106
Table 6.12. Definition of the 25-bar space truss 106
Table 6.13. Comparison of the results for the 25-bar space truss 107
Table 6.14. Definition of the 52-bar dome space truss 109
Table 6.15. Comparison of the results for the 52-bar dome space truss 110
Table 6.16. Discrete solution for the 52-bar dome space truss 111
Table 6.17. Loading details for the 56-bar transmission tower space truss 114
Table 6.18. Definition of the 56-bar transmission tower space truss 114

xiii
Table 6.19. Comparison of the results for the 56-bar transmission tower space truss

Table 6.20. Discrete solution for the 56-bar transmission tower space truss

Table 6.21. Definition of the 120-bar dome space truss

Table 6.22. Comparison of the results for the 120-bar dome space truss

Table 6.23. Discrete solution for the 120-bar dome space truss

Table F.1 Dimension and properties of circular hollow sections (AS 1163)

Table G.1 Stresses and displacements obtained for the discrete optimised solution for example 6.1 (10-bar plane truss) based on non-linear analysis.

Table G.2 Stresses obtained for the optimised solution for example 6.2 (20-bar plane truss) based on linear and non-linear analysis.

Table G.3 Displacements obtained for the optimised solution for example 6.2 (20-bar plane truss) based on linear and non-linear analysis.

Table G.4 Stresses obtained for the optimised solution for example 6.3 (51-bar roof truss) based on non-linear analysis.

Table G.5 Displacements obtained for the optimised solution for example 6.3 (51-bar roof truss) based on non-linear analysis.

Table G.6 Stresses obtained for the optimised solution for example 6.4 (25-bar space truss) based on linear and non-linear analysis.

Table G.7 Displacements obtained for the optimised solution for example 6.4 (25-bar space truss) based on linear and non-linear analysis.

Table G.8 Stresses obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis.

Table G.9 Displacements obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis.
Table G.10 Stresses obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis.

Table G.11 Displacements obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis.

Table G.12 Stresses obtained for the optimised solution for example 6.6 (56-bar transmission tower space truss) based on linear and non-linear analysis.

Table G.13 Displacements obtained for the optimised solution for example 6.6 (56-bar transmission tower space truss) based on linear and non-linear analysis.

Table G.14 Stresses obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

Table G.15 Displacements obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.
\begin{itemize}
\item \( A \) = cross-sectional area of the truss element
\item \( A^l \) = lower bound value of size variables
\item \( A^u \) = upper bound value of size variable
\item \( c \) = limiting value of the constraint
\item \( C_j \) = binary string number
\item \( C \) = column slenderness ratio
\item \( Cs \) = stress violation coefficient
\item \( Cd \) = displacement violation coefficient
\item \( d \) = nodal displacement
\item \( d^a \) = allowable displacement
\item \( E \) = modulus of elasticity
\item \( f_i \) = fitness value of an individual
\item \( F_y \) = yield stress of material
\item \( \Delta F \) = increment load, of the total external load
\item \( FS \) = factor of safety
\item \( \{F\} \) = vector of applied joint loads
\item \( g \) = inequality constraint
\item \( h \) = equality constraint
\item \( I_i \) = integer mapping of a binary string
\item \( k \) = decimal precision
\item \( K \) = effective length factor
\end{itemize}
\[ [K] \] = structure’s global stiffness matrix

\[ [K^e] \] = element stiffness matrix

\[ [K_T] \] = structure’s overall tangent stiffness matrix

\[ [K_s] \] = system tangent stiffness matrix

\[ [K_E] \] = structure’s linear elastic overall tangent stiffness matrix

\[ [K_G] \] = structure’s geometric stiffness matrix

\( l \) = truss element’s length

\( L \) = length of binary string of design variables

\( n \) = population size

\( N \) = total number of degrees of freedom

\( n_c \) = number of coordinates

\( n_e \) = total number of elements in the structure

\( n_g \) = maximum number of generation

\( P \) = reproduction probability

\( P \) = intermediate axial force of truss element

\( p_c \) = probability of crossover, (crossover rate)

\( P_e \) = penalty value

\( P_m \) = probability of mutation (mutation rate)

\( \Delta P \) = unbalanced nodal force

\( q_n \) = cumulative probability of selection

\( r \) = appropriate radius of gyration

\( s \) = slenderness ratio

\( S \) = list of design candidate (discrete values)

\( t \) = time
\( T \) = integer variable that assume values of 1 or 0, respectively, for presence or absence of an element

\( u \) = truss element’s end deformation in the local \( x \)-direction

\( U \) = local elastic strain energy increment

\( \{U\} \) = vector of nodal displacements

\( \Delta u \) = incremental value of joint displacements

\( v \) = truss element’s end deformation in the local \( y \)-direction

\( V \) = volume of the truss element

\( v_p \) = violation parameter

\( W \) = objective function of the problem

\( x \) = vector of design variables = \([x_1, x_2, \ldots, x_n]^T\)

\( x^o \) = initial design point

\( x^l \) = lower bound value of design variable

\( x^u \) = upper bound value of design variable

\( e^o \) = strain increment

\( \phi \) = angle of rotation of truss element

\( \Phi \) = fitness value of an individual

\( \lambda_k \) = Lagrange multiplier

\( \nu \) = total number of node points

\( \rho \) = weight density of material

\( \sigma \) = member stress

\( \sigma^a \) = allowable stress in both tension and compression.

\( \sigma^- \) = compressive stress
\( \sigma^+ \) = tensile stress

\( \sigma^{-b} \) = Euler buckling stress

\( \sigma^{+b} \) = allowable stress in tension
1.1 General Introduction

During the past several decades, the optimal design of engineering structures has been the subject of considerable research activity. To find the "best" or "optimum" design is always a very challenging work. In structural design, safety is considered to be one of the most important design criteria. Cost is typically the second most important factor in structural design. Safety is usually improved by analytical checks of the problem and on the other hand, cost is improved by optimising design. The central purpose of structural design optimisation problem is to determine a set of design variables, which are usually member properties or dimensions that minimise the total weight or cost of the structure, while satisfying safety and performance constraints.

Traditionally, the optimisation problem has been solved by trial-and-error dictated by design specifications and guided by the practice and intuition of the designer, a method that has worked well as evident from the existence of many fine buildings and other structures. The advancement of the application of high-speed electronic computers has made the analysis and optimal design of problems much more accurate than ever
before, which in turn has led an increased use of structural optimisation research to achieve more efficient and economical design. Research is still vigorously pursued for a range of reasons, including the need to handle a wider class of problems, to include realistic definitions of design variables, to find techniques to locate the global optimum and to reduce the design cycle time, and to improve the efficiency of the numerical procedure Ref. [1].

Much of the past research in optimisation of structures has dealt with steel truss structures. In the optimisation of steel truss structures one of the most important practical considerations is that the design variables such as cross sectional areas of truss members have to be chosen from a list of discrete values due to the reason that components are only available in discrete standard sizes because of the manufacturing practices. This leads to a discrete optimisation problem, in which continuous optimisation techniques cannot be directly used, making the problem much more complicated to solve.

In recent years, genetic algorithms (GAs), which are applications of biological principles into computational algorithms, has become most successful and powerful for solving a wide variety of engineering problems, because of their simplicity, global perspective, and inherent parallel processing [2]. Furthermore unlike classical optimisation methods, GAs do not require gradients of the objective or constraint functions but only require representation of the structures being optimised and an evaluation function to determine the quality of the structures. Many research studies have recently reported the solution of truss structure optimisation problems through GAs [1], [3-19] and all these studies have shown that the GA is an effective tool for the optimal design of truss structures. These studies are mainly focused on the design of trusses with linear behaviour.
In structural optimisation there are cases where non-linear analysis is needed to determine the behaviour of the structures such as long-span, cable structures and slender structures (suspension bridges) under external loading. None of the previous work is based on non-linear analysis with GA as the discrete optimisation tool.

The study reported here presents an optimum design method based on GA for geometrically non-linear plane and space truss using only member sections, which are available in discrete size. In addition, in order to make comparison with the optimum design obtained for plane and space truss structures based on non-linear analysis, an optimisation method is also developed for plane and space truss structures based on linear analysis. The proposed algorithms are based on a roulette-wheel reproduction scheme, a one-point crossover and a standard mutation scheme. An elitist strategy is also used that passes the best designs of a generation to the next generation. GAs are well suited for unconstrained optimisation problems, therefore a penalty function method based on violation of normalised constraints is used to transfer the constrained optimisation problem into an unconstrained one. In optimisation process the non-linear analysis of trusses is solved by an incremental strategy and non-linear equations are linearised and iteratively solved by the Newton-Raphson method.

The efficiency of the algorithms has been investigated by applying them to the minimum weight design of a number of plane and space truss structures under a range of different loading conditions. From these problems, it has been proven that the proposed optimum design methods are effective and reliable to solve both continuous and discrete optimisation problems, and it is possible to obtain optimum solutions by including the geometrically non-linear analysis in the design of truss structures.
1.2 Aim of Present Study

The research study reported herein is aimed at devising an efficient methodology based on binary coded GA for optimum design of steel plane and space truss structures subject to member stresses, nodal displacements and/or member buckling constraints. The objective function of optimisation is to minimise total weight and/or volume of the structure in concern. The cross sectional areas of truss members are considered as design variables and are assumed to be either continuous or discrete. The basic objectives that were considered are as follows:

1. To develop computer programs to solve both linear and geometrical non-linear truss structures using finite element (FE) method.
2. To develop an algorithm that is based on GA and FE method for the optimum design of plane and space truss structures subject to discrete design variables with geometrical non-linear as well as linear analysis.
3. To compare the results obtained from the optimisation methodology based on linear behaviour with the established results available from previous research.
4. To validate and compare the results obtained from the above said methodologies based on linear as well as non-linear behaviour.

1.3 Outline of Thesis

The general introduction of this chapter is followed by the discussion of the basic formulation of the structural design optimisation problem and the most relevant techniques of structural design optimisation in Chapter 2. Chapter 3 gives an overview of genetic algorithm by demonstrating how different operators work. Different types of truss structure optimisation problems are described along with application of GA in
different types of truss structure optimisation problems are discussed in Chapter 4. Chapter 5 describes the proposed optimisation technique for plane and space truss structure. The application of the proposed algorithm in different truss structure problems with a range of different load conditions is illustrated in Chapter 6. Finally, in Chapter 7, the discussion and conclusion are presented.
2.1 Introduction

The goal in optimum design is to obtain the best solution for a given engineering design problem, which must satisfy all the limitations and constraints. In various practical applications of engineering designs, it is of common occurrence that the design variables of an optimisation problem are not continuous and some or all of the design variables must be selected from a list of integers or discrete values. Furthermore, the optimisation problems are characterised by various objective and constraint functions, which are generally non-linear functions of the design variables. Various gradient-based mathematical optimisation algorithms have been developed and are used for structural optimisation. In recent years, the use of evolutionary algorithms, especially genetic algorithms (GAs), which are based on the mechanism of natural selection and evolution of natural genetics, are getting increasingly more attention from researchers, mainly because of the inability of gradient-based techniques to handle the discrete nature or mixed discrete continuous of design variables efficiently.

The basic formulation of the structural design optimisation problem and the
2.2 General Formulation of Optimal Design

Broadly structural optimisation can be defined as the process of choosing the values of a set of design variables that optimise (i.e. minimise or maximise) a specific quantity termed the objective (or cost) function, while satisfying performance constraints that are related to the design problem and the behaviour of the structure.

2.2.1 General Problem Formulation

The purpose of the optimisation design problem formulation is to create a mathematical model, which can be solved using an optimisation algorithm and can be expressed mathematically in the following general form:

Minimise or maximise: 

\[ W(x) \]  

Subject to: 

\[ g_k(x) \leq c_i \quad k = 1, 2, \ldots, m \]  

\[ h_j(x) = 0 \quad j = 1, 2, \ldots, l \]  

and 

\[ x_i^l \leq x_i \leq x_i^u \quad i = 1, 2, \ldots, n \]  

\[ x_i \in S \]  

where:

- \( x = \{x_1, x_2, \ldots, x_n\}^T \) is an \( n \)-dimensional vector of design variables
- \( W(x) \) is the objective function of the problem
- \( g(x) \) and \( h(x) \) are functional expressions of the \( m \)th inequality constraint and \( l \)th equality constraint respectively
• $c_i$ is the limiting value of the constraint

• $S$ represents a list of design candidates (discrete values) and the values $x_i^l$ and $x_i^u$ are the lower and upper bounds of the design variables, respectively.

Equations (2.1)-(2.5) represent the discrete formulation of the optimisation problem. However, if equation (2.5) is ignored, the optimisation problem becomes a continuous one.

The vector $x$ is an $n$-dimensional space also known as design space, where $n$ is the number of design variables. A set of design variables defines a design point in the design space. Such a point is called a solution. A design, which satisfies all the constraints in the optimisation problem, is a feasible design, whereas if all the constraints of the problem are not satisfied, it is termed an infeasible design.

2.2.2 Design Variables

A structural design problem usually involves many design parameters of which are of significant importance and can be varied by the design modification procedure. These parameters are called design variables. In other words, they are the parameters that control the geometry of the optimised structure during the optimisation process. From a physical point of view the design variables that are varied by the optimisation procedure may represent the following properties of the structure [20]:

• the structure’s member sizes or cross-sectional areas

• the shape or geometric layout of the structure

• the topology of the structure, i.e., the pattern of connection of members or the number of elements in a structure.

• the material distribution.
The design variables in structural optimisation problems can be of *continuous* or *discrete* variables or can be a combination of both discrete and continuous. In the case of continuous design variables, the search space is usually infinite and the design variable is the one that takes any value in the range of the variation in its region. In the case of discrete design variable, the search space is finite and the design variable is the one that takes only values from a list of available values or a catalogue.

2.2.2.1 Sizing Design Variables

The cross-sectional properties of a truss member, the moment of inertia of a flexural member and the thickness of a plate structure are some examples of this type of design variables. The variables can take various values, such as discrete, continuous, or a combination of both discrete and continuous. In many practical problems in engineering design, sizing design variables may be restricted to discrete values. This is due to the availability of components in standard sizes and the constraints due to construction and manufacturing practices. In such cases the design variables can only take the given discrete set of available values.

2.2.2.2 Shape Design Variables

The shape or geometric design variables may represent the change of member length, hence, the change of joint location of a truss structure or a frame structure. Another example of this class of design variables is the location of supports in a bridge, the length of spans in a continuous beam and the height of a shell structure. In general, the geometry of the structure is represented by continuous variables therefore shape design variables can be treated by most optimisation techniques.
2.2.2.3 Topological Design Variable

The topological design variables can be structural parameters such as the presence or absence of members, and the presence or absence of fixity conditions at supports. A topology optimisation problem is a discrete and continuous combination problem with many local optimum solutions. The design space is strongly non-convex therefore classical optimisation methods have not been used adequately in these types of problems [12]. In certain cases, optimisation allows certain members to reach zero size, thereby eliminating the need of uneconomical members.

2.2.3 Constraints

In structural design optimisation problems, a constraint is a restriction that must be satisfied in order for the design to be acceptable or feasible. There are two main types of constraints [20]: equality constraints and inequality constraints. Equality constraints are those that a legal solution must be satisfied, such as equilibrium, compatibility, and constitutive relation. Constraints in the form of inequality determine an allowable domain of solution. Limitations on the member stresses, nodal deflections, vibration frequencies and buckling strengths are examples of this type of constraints. Moreover, some constraints may determine a limit on the range of variation of a design variable. Such constraints are called the side constraints.

Constraints are usually handled either in direct methods or penalty methods. In direct methods, the active constraints are used directly as limiting surfaces and the gradient of the objective function are used to determine a direction that improves the objective function while not violating any constraints. In penalty methods a penalty value is added to the value of the objective function for the violation of constraints and then unconstrained optimisation is performed using the augmented objective function.
Minimisation of the function also minimises the constraint. This method can be applied to both equality and inequality constraints.

### 2.2.4 Objective Function

An objective function (also known as the merit or cost function) is itself a function of the design variables. It constitutes a basis for choice between alternative acceptable designs. The most common for the formulation of structural optimisation problem is the total weight of the structure, which is employed as the objective function. In structural optimisation problem the objective function is the criterion that will be minimised. Therefore optimum design problem are dominantly minimum weight design problems, although other quantities such as reliability, deflection, stiffness and actual cost have been taken as an objective function as well.

In the present study, the objective function is taken as the weight or volume of the overall truss structure. The constraints are limitation on the design variables, member stresses, nodal displacements and member buckling. The design variables (cross sectional areas of truss members) are considered either continuous or discrete values. In the discrete optimisation the cross sectional areas of truss members and the type of the sections are assumed to be the standard circular hollow sections suggested by the AS 1163 [21].

### 2.3 Optimisation Techniques

Numerical optimisation techniques have been developed and are in use for design optimisation of structural systems. In general, these techniques can be classified as:

- **gradient-based techniques**
- **evolutionary algorithms**
Numerous literature exist that go into detail of these techniques, among them Goldberg [2], Atrek *et al.* [22], Vanderplaats [23], Arora [24], Spears [25], Back [26] and Michalewicz [27]. Prior to a detailed description of the optimisation techniques, the following section provides a brief background leading onto the use of optimisation techniques.

2.3.1 Background

Engineering optimisation problem has traditionally been solved by trial-and-error dictated by design specifications and guided by the experience, intuition and knowledge of the designer. The quality of the final design as well as the design process depends heavily on the designer’s experience and intuition. This method has worked well in the past as evidenced by the existence of many fine buildings and other structures.

According to Vanderplaats [23], the modern developments of numerical structural optimisation began in 1960 when Schmit [28] applied non-linear mathematical programming methods and numerical techniques to the optimal design of a simple three-bar truss structure. Schmit [28] showed that a significant class of structural design optimisation problems could be cast as non-linear mathematical programming problems. A great deal of effort has gone into improving the computational efficiency of the various mathematical programming approaches [29,30]. In 1964, Moses [31] introduced the technique of *sequential linear programming*. Most of these improvements reduced the number of finite element analyses required to determine the optimal solution. In 1968 another approach called *optimality criterion* methods emerged [32]. These methods are based on minimising the weight subject to criteria specified equality constraints. Prager and Marcal [33] and Taylor [34] have been instrumental in the development of much of this work. Ovadia [20] provides a comprehensive list of
research work covering development of numerical structural optimisation methods in the 1970s.

The development in automating the engineering design process can provide the benefits such as the ability to have a systematized logical design procedure dealing with a wide range of design variables and constraints with minimal human intervention.

Several disadvantages, however, do exist. Early form of numerical optimisation techniques had difficulties dealing with discrete design variables and non-linear problems, usually resulting in slow or no solution at all. In addition these techniques are not guaranteed to provide a global optimum solution [23].

With continuing research in both optimisation methods and their implementations to develop better techniques, and advances in modern computing methods, relatively new techniques such as genetic algorithms and simulated annealing have been suggested in the engineering optimisation field. According to Goldberg [2], the interest in genetic algorithm method began as early as the 1970s, when Holland’s [35] book “Adaptation in Natural and Artificial Systems” presented the genetic algorithm as an abstraction of biological evolution and gave a theoretical framework for adaptation under the genetic algorithm. This interest was followed by Kirkpatrick’s [36] simulated annealing technique in 1983. This technique is based on the analogy to the natural energy minimisation process as found in melt metal during a controlled temperature-dropping schedule. Prior to this, in the late 1960s, Rechenberg [37] introduced evolution strategies, a method first designed to optimise real-value parameters. This idea was further developed by Schwefel [38]. Also in 1966, Fogel et al. [39] developed evolutionary programming in which candidate solutions to given tasks are represented as finite-state machines, and the evolutionary operators primarily consisted of selection and mutation.
Gradient-based techniques require the evaluation of the objective function and constraint equations to guide the algorithm to the optimum solution and require continuous design variables. Therefore these techniques have been shown to be able to locate the continuous optimal design for large practical structures and can obtain the local optimum solution around the initial design points more easily. However, some engineering design problems involve only a limited set of discrete variables. For example reinforcing bars and rolled steel beams are generally available in standard sizes. Consequently, in optimum structural design a necessary practical requirement is that the optimum design should employ only the standard discrete sections. This usually leads to a discrete optimisation problem. In discrete optimisation problems, searching for the global or local optimal solution becomes a difficult task. Furthermore the discrete design space is disjointed and non-convex. Thus, the continuous optimisation methods cannot be directly used to solve the discrete optimisation problems and global optimality of a local solution cannot be guaranteed [23].

The simple and practical method to obtain a discrete solution using gradient-based techniques is to consider the optimisation problem with continuous design variables and then a set of discrete values is selected based on the continuous solution [40]. However, this method can easily result in overweight or violated discrete design, furthermore, it cannot guarantee the globally optimum design [41].

Another obstacle encountered by gradient-based techniques is the need to continuously represent secondary properties (e.g. radius of gyration) often required by buckling constraint equations. For example, when considering buckling constraints in structural optimisation, the radius of gyration appears as a design variable, which consequently may produce difficulties [42]. One solution, as presented by Adeli and
Balasubramanyan [43] and John and Ramakrishnan [44] involves the approximation of radius of gyration to cross-section area.

The most widely used approaches in structural optimisation include the mathematical programming and optimality criteria methods. The following two sections offer a brief discussion of these techniques.

2.3.2.1 Mathematical Programming

A number of different mathematical programming techniques have been developed. Depending on the form of the objective function and constraints, mathematical programming can be subdivided into two main methods:

- *linear programming* (LP)
- *non-linear programming* (NLP)

The main characteristic of the LP is that the objective function and constraints in the design problem are expressed as a linear combination of the design variables. If some of the objective or constraint functions are non-linear, the problem is then classified as NLP. LP is particularly important because a wide variety of problems can be modelled as linear problems, and because there are faster and successful methods for solving linear problems even with a large number of variables and constraints. However, structural optimisation problems are always non-linear. In many cases the objective function can be formulated as a linear function of the design variable, but the constraint functions are usually non-linear.

In applying LP techniques to the structural optimisation problems, the relationship between the objective function and constraints to the design variables need to be linearised. Vanderplaats [23] confirms this linearising as being the most fundamental concept of automated design, that is, "always use the simplest, most direct design
Figure 2.1 shows a typical linearisation process. The axes represent two design variables, \( x_1 \) and \( x_2 \).

**Fig. 2.1** The linearised problem (Vanderplaats [23]).

The objective function \( W(x) \) is initially represented as non-linear (concave curves) and the problem's constraints are shown as non-linear curves \( g_1(x) \) and \( g_2(x) \). At the initial design point \( x^0 \), the objective function and constraints are linearised using the first-order Taylor series and are shown as the dotted straight lines. Generally a feasible solution is located in the area bounded by the constraints curves. However, after linearisation the optimum solution may sometimes be located outside the solution area.
This solution is therefore considered not feasible, and only after a number of linearising iterations at that point, can a feasible solution be obtained.

Other problems arise, when fewer active constrains are considered in the design problem. This is illustrated by Figure 2.2, showing only one non-linear constraint $g_1(x)$. Consequently the linear approximation to the problem is unbounded.

![Fig. 2.2 Under constrained problem and move limits (Vanderplaats [23]).](image)

The problem is overcome by using a 'Move Limits' technique (shown as dotted box), which during each iteration searches for a solution within the feasible domain. Consequently, when a linear relationship is used to model a non-linear problem, it can therefore be understood, that errors are inevitable during the analysing process.
Non-linear mathematical programming was developed for non-linear unconstrained optimisation problems. The constrained NLP problem can be transformed into unconstrained problems. A common means to accomplish this transformation is to define a Lagrangian function of the form \[45\]:

\[
L(x, \lambda) = W(x) + \sum_{j=1}^{l} \lambda_j h_j(x)
\]  

(2.6)

where

- \(W(x)\) is the objective function
- the constraints, \(h_j(x)\), can represent both equality and inequality constraints
- \(\lambda_j\) are referred to Lagrangian multipliers.

The minimum of \(L(x, \lambda)\) proves the true minimum of the objective function. In order to guarantee a solution, the number of constraints \(l\) must be less than the number of design variables. In addition, the objective function and constraints function need to be differentiable with respect to the continuous design variables. A stationary point or relative minimum, \(x^0\) exists if,

\[
\frac{\partial L(x, \lambda)}{\partial x_i} = 0 \quad i = 1, 2, \ldots, n
\]  

(2.7)

\[
\frac{\partial L(x, \lambda)}{\partial \lambda_j} = 0 \quad i = 1, 2, \ldots, l
\]  

(2.8)

The above \((n+l)\) equations are known as the Kuhn-Tucker necessary conditions.

A wide spectrum of structural optimisation problems has been solved using the above-mentioned mathematical programming techniques. Vanderplaats [23] goes on to describe several application of LP in structural optimisation. These include the design of
truss structures by Lapay and Goble [46], the limit design of truss structures by Pearson [47] and limit design of frames by Livesley [48].

Using NLP, Moses [31], Vanderplaats and Moses [49], Sheu and Schmit [50] Farshi and Schmit [51] and Lipson and Gwin [52] found a minimum weight of truss structures. Levy at el. [53] report that, Rosen and Schmit [54]-[56] used NLP approach for optimisation in conjunction with an approximate analysis in dealing with local and system imperfection for truss structures. Solution to non-linear structural optimisation problems have been attempted by sequential quadratic programming (SQP) methods combined with a series of iterations [57,58]. These involve linearising all the original non-linear functions with respect to the design variables. John at el. [59], Lamberti and Pappalettere, [60] and Chen [61] used the sequential linear programming (SLP) method to find the minimum weight design of plane and space truss structures. They formulated the design of trusses as a problem of NLP in the space of design variables and solved it by the optimisation method of (SLP) with Move Limits technique.

Mathematical programming methods present a satisfactory local rate of convergence, but they cannot ensure that the global optimum can be found. These methods require restarting the optimisation process from several different points, which may prove to be an extensive and time-consuming procedure [23]. As a result more computationally efficient methods incorporating optimality criteria have been developed, a description of which follows.

### 2.3.2.2 Optimality Criteria

The optimality criteria (OC) methods reduce the computational effort using an iterative procedure with criteria based on the behaviour of the structure under consideration. The concept of statically determinate or indeterminate structures and certain variational
principles of structural mechanics are used in OC methods. Kuhn-Tucker conditions of non-linear mathematical programming combined with Lagrangian multipliers were indirectly used in the development of the OC methods. In OC methods each iterative cycle consists of two steps. In the first step the structure is analysed under the applied load to find their behaviour. In the second step, the design variables are modified to minimise the objective function and satisfy the constraints [22].

Atrek et al. [22] reported that, the OC approach has been used for the optimisation of structures with stability and dynamic stiffness constraints. Khot et al. [62], Venkayya and Khot [63], Kiusalaas and Shaw [64], and Segenreich and McIntosh [65].

Khot and Kamat [66], Saka [67] and Levy and Perng [53] developed an optimisation method based on an OC to the minimum weight design of truss structures under system non-linear stability. Pezeshhk [68] applied an OC approach to determine the minimum weight design of truss structures with constraints on the non-linear strain energy density distribution. Saka [69] proposes a method based on an OC to optimum shape of roof truss structures with displacement, stress, and buckling and minimum size constraints. The roof slope of structure has been treated as a design variable in addition to cross sectional areas. Saka and Ulker [70] and Saka [67] have shown that the OC approach is well suited for the optimal design of large space truss structures based on geometrically non-linear analysis. The algorithm was based on coupling the OC approach with a large deformation analysis technique. Adeli and Soegiarso [42] present a method based on an OC to optimum design of various large steel structures involving many sizing variables. They studied difference of optimum designs obtained from two versions of the design procedure proposed by AISC. These two-design procedures are the Load and Resistance Facture Design (LRFD) and the Allowable Stress Design (ASD). Although it is noted that the OC method can be remarkably efficient if solution
convergence is attained, there is no guarantee that the OC approach can always converge to a solution point [71,72].

The above discussion suggests that the gradient-based methods are not very efficient for discrete optimisation of structural design problems. In the following section, the evolutionary algorithms are described, which are direct probabilistic search approaches and works according to the principles of natural genetics and evolution.

2.3.3 Evolutionary Algorithms

Evolutionary algorithms (EAs) have been used in various forms and received much attention from scientists working in many different disciplines. A variety of evolutionary algorithms have been proposed. The major ones are:

- Evolutionary Programming (EP)
- Evolution Strategies (ES)
- Genetic Algorithms (GAs).

These algorithms can be considered as an analogy of the mechanism of the natural evolution, where a biological population evolves over generations to adapt to an environment by selection, recombination and mutation. EAs work on function evaluations alone and do not require derivatives or gradients of the objective and constraint functions. This condition makes EAs suitable to be used for hard and complex optimisation problems.

When EAs are applied to optimisation problems, fitness, individual and genes usually correspond to an objective function value, a design candidate, and design variables, respectively. The starting point of the evolutionary process is a population of randomly created individuals. The individuals of this initial population will represent possible solutions to the problem. Each of these individuals has certain characteristics
that make them more or less fit as members of the population. The most fit members will have a higher probability to be selected and placed into the mating pool. The less fit members will die and be replaced by the most fit one. This corresponds to the principle "Survival of the fittest" in nature. New individuals for the next generation are created through crossover and mutation process. The aim of the EA is to find an individual with the maximum fitness.

Spears [25] and Back [26] discuss the three types of EAs and explain their difference in evaluation methodologies. It is understood, that the most obvious difference is given by the interpretation of the role of genetic operators. In EP, mutation is regarded as the main search operator, while GAs emphasize on recombination. ESs are known to use both operators.

Genetic algorithms are part of the larger class of evolutionary algorithms that also includes evolution strategies and genetic programming. GAs are attracting much interest from researchers all over the world [26].

2.3.3.1 Genetic Algorithms

As already noted, the genetic algorithms (GAs) are direct probabilistic search approached based on the Darwinian’s principle of reproduction and survival of the fittest. GAs are most appropriate for complex non-linear models where locating the global optimum is a difficult task. It may be possible to use GA techniques to consider problems, which may not be modelled as accurately using other approaches. Therefore, GA appears to be a potentially useful approach. A more detailed description of the GAs is provided in Chapter 3.

Successful applications of GAs technique to various problems have been reported in recent years. GAs have been applied to commerce, engineering, mathematics,
medicine and pattern recognition with promising results [2]. The GA was introduced to
the civil engineering community by Goldberg and Samtani in 1986 [16] and was
immediately embraced by structural optimisation researchers. Since, several research
efforts have applied GAs to engineering design optimisation problems in a variety of
domains. Hadi and Arfidi [73] applied GAs in active control of civil engineering
structures. Hadi and Schmidt [74] and Hadi [75] used GAs to find the optimum design
of reinforced concrete beams. Much of the GA research applied to structures has
focused on trusses. Rajeev and Krishnamoorthy [3,8], Shyue and Pei-Tse [7] and
Erbature et al. [11] used GAs to find the minimum weight design of truss structures
with discrete design variables. Adeli and Cheng [4,5] used an integrated GA to optimal
design of large structures and space trusses. Coello and Christiansen [10] studied the
basic concept and part of the most relevant work in the multiobjective structural
optimisation using GAs. GAs have also been used to find the best structural topology as
illustrated by Sakamoto and Oda [12], Ohsaki [13], Hajela and Lee [14] and Grieson
and Pak [15]. Rajan [1] used GA to design of truss structures by combining sizing,
shape and topology.

All these studies have shown that the GAs are an effective and ideal optimisation
algorithm for engineering optimal design problems and are simple to implement
when compared to other optimisation techniques.

2.4 Concluding Remarks

This chapter introduces the basic formulation of a structural design optimisation
problem and reviews some relevant optimisation techniques. These techniques including
gradient-based and evolutionary algorithms.

Some optimisation problems in structural engineering are very complex in nature
and cannot be solved by the gradient-based techniques efficiently. These techniques demonstrate a number of difficulties when faced with complex problems. The common difficulties are summarised below:

- Gradient-based techniques require continuous functions and their derivatives.
- Gradient-based techniques do not have the breadth to solve different type of problems. This is because each gradient-based techniques is designed to solve only a particular class of problem efficiently.
- Most gradient-based techniques converge to the local optimal solution, but they cannot ensure that a global optimum is found, unless the constraints are convex.
- Gradient-based techniques are not efficient in handling the discrete nature of design variables.
- Gradient-based techniques cannot be efficiently used in parallel computing environment.

Evolutionary algorithms, especially genetic algorithm, have been attracting much interest from researchers to solve optimisation problems. The genetic algorithms offer a new approach to structural optimisation, which overcomes most of the problems associated with gradient-based techniques. Unlike many gradient-based techniques genetic algorithms do not require continuous representation of the design variables, and the derivatives of the objective and constraint functions. This is a great advantage of genetic algorithms allowing them to solve a wide range of optimisation problems. Furthermore genetic algorithms offer an effective and reliable method to solve discrete optimisation problem and can be applied to any discrete set of sections produced according to the different standards. Successful applications of this technique to various truss problems are reported in this thesis.
The next chapter of this thesis offers a detailed description of the genetic algorithm.
3.1 Introduction

Genetic algorithms (GAs) were first described by John Holland in the early 1960s and further developed by John Holland and his colleagues and students at the University of Michigan in the early seventies. Prior to this, in the early 1950's, biologists had used digital computers to perform simulations of genetic system. In 1962, Fraser [76] showed the possibility of applying genetic search for function optimisation. This technique became popular after the publication of Holland's book "Adaptation in Natural and Artificial Systems" in 1975. The main goals of the research of Holland and his students were [2]:

- to abstract and rigorously explain the adaptive processes of natural systems.
- to design artificial systems software that retained the important mechanisms of natural systems.

The GA technique has been notably developed by Goldberg [2] and Michalewicz [27] who also give an excellent introduction to the subject.
In GA, a candidate of a design variable can be represented using binary strings, graphs (neural networks), Lisp expressions, ordered lists, and real-valued vectors [25]. The most common type of GAs used are binary string genetic algorithms and have proved to be useful in a variety of optimisation problems [2]. The current research is mainly focused on the binary coded genetic algorithms.

The goal of this chapter is to provide the necessary details to understand the proposed methodology used in this thesis.

3.2 Genetic Algorithms

Genetic algorithms (GAs) are the stochastic search methods that explore the design space by generating random numbers to guide the algorithm to an optimal design. The basic idea behind the mechanics of GAs is to simulate the Darwinian principle of reproduction and survival of the fittest, with information exchange among the survivors. Like biological systems, there is some randomness to this process, but instead of causing detrimental effects, this randomness gives GA robustness and the ability to generate better solutions. GAs work with a population of candidate solutions, which is sampling of several points by considering several equations at the same time. This makes GAs less susceptible to difficulties encountered in problems with noisy design space. In this approach, the design variables are represented by a chromosome (binary bit string). The use of such representation instead of actual design variable values allows for an easy inclusion of discrete and integer design variables in the problem. GAs also can treat continuous design variables by specifying required precision. GA's use only a fitness or objective function value to guide the search strategy and do not rely on derivative information. This condition makes GAs to have capability of finding a global optimum. Furthermore GAs are easy to understand and simple to realize.
3.3 How Genetic Algorithms Work

Genetic algorithm (GA) is an iterative optimisation procedure that consists of a constant-size population of candidate designs. Each iterative step is called a generation. An initial set of possible designs, called an initial population, is generated at random. All of the candidate designs are encoded as artificial individuals (chromosomes). Each chromosome is assigned a fitness, which is directly related to the objective function of the search and optimisation problem. Based on the fitness, chromosomes undergo selection process. Chromosomes with higher fitness values have a higher probability to be selected and placed into the mating pool. The chromosomes in the mating pool are altered using crossover (exchanging of portions of binary strings) and mutation (random changing of binary bits) operations resulting in a new population of chromosomes (offspring), which combines the desirable characteristics of the old population, and then the new population replaces the old one. The GAs procedure is repeated over many generations until a termination criterion is achieved.

The adaptive process of genetic algorithm is roughly represented in Figure 3.1, where $P(t)$ is the population of strings and $t$ is the time or generation number [27].

There are six stages in the genetic search process. These stages are summarized below:

1. **Initialise**: Generate random initial population of chromosomes.
2. **Evaluate**: Evaluate each of the chromosomes in the initial population and calculate the fitness of each chromosome according to the objective function.
3. **Selection**: Based on the fitness, select two parent chromosomes and place them in the mating pool (the better the fitness, higher the probability to be selected).
**Fig. 3.1** The Process of a Genetic Algorithm [27]

4. **Crossover:** Crossover the parents to produce new chromosomes (offspring). If no crossover was performed, the offspring is an exact copy of the parents.

5. **Mutation:** Mutate the genes of the new offspring. If no mutation occurred, the offspring is the direct result of crossover.

6. **Evaluate:** Evaluate fitness of each chromosome.

The new population of chromosomes replace the old one and step (3) to step (6) are repeated over many generations. After several generations, the result is often one or more highly fit chromosomes in the population. It is important to note that the GA searches for the fittest chromosome within the population.

The reproduction mechanism in a GA is composed of selection, crossover and mutation. These three main operations do much of the genetic algorithm’s work. The
application of these operators depends strongly on the design variables representation used. The details of the coding of the design variable and the GA operators are described in the following sections.

3.3.1 Encoding the Design Variables

The first step in GA formulation is encoding the components of candidate solutions. The aim of encoding is to create a representation of the solutions to a GA representation. The representation scheme determines how the problem is structured in the GA. Depending on the type of design variable in the problem at hand appropriate decoding schemes have to be adopted. There are two basic methods used by GAs to encode the parameters:

- *binary encoding*
- *direct value encoding*

In the direct value encoding the design variables are represented by real number and GA operators are directly applied on real numbers [27]. However in binary coding the design variables are coded in binary strings. Binary coding was the original formulation of GAs and is most commonly used method in design variables encoding. This method of encoding is simple to create and manipulate and can represent maximum information with minimum number of bits. The binary representation is applicable to any design variables and can be operated with common genetic operators. This method has proved to be useful in a variety of optimisation problems [2]. The main emphasis is based on binary coding in the current research.

In binary encoding, the design variables are represented as *genes*, with each gene being an instance of a particular *allele* (1's and 0's) on a chromosome (binary bit string). Each bit in the binary string represents some characteristics of the solution. As
an example, the binary representation of four chromosomes that represents two design
variables $x_1$ and $x_2$ is shown in Figure 3.2.

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Binarystring</th>
<th>Genetic terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0110110111</td>
<td>allele = (1 or 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>genes = $x_1$ or $x_2$ = 01101 or 10111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chromosome = $x_1x_2$ = 0110110111</td>
</tr>
<tr>
<td>2</td>
<td>1110111010</td>
<td>allele = (1 or 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>genes = $x_1$ or $x_2$ = 01101 or 10111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chromosome = $x_1x_2$ = 0110110111</td>
</tr>
<tr>
<td>3</td>
<td>0000100100</td>
<td>allele = (1 or 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>genes = $x_1$ or $x_2$ = 01101 or 10111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chromosome = $x_1x_2$ = 0110110111</td>
</tr>
<tr>
<td>4</td>
<td>1000001001</td>
<td>allele = (1 or 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>genes = $x_1$ or $x_2$ = 01101 or 10111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chromosome = $x_1x_2$ = 0110110111</td>
</tr>
</tbody>
</table>

Fig. 3.2. Example of Binary String Representation

The total length of a chromosome is equal to the sum of the sub-chromosome
length of each design variable. The length of the bit string for each gene depends on the
size of the search domain namely lower and upper boundaries and desired precision of
the design variable. The lower bound solution is represented by the solution (00000) and
the upper bound solution is represented by the solution (11111). In the example shown
above, the string integer values lie between 4 and 7, and required precision is one place
after the decimal point. These 5 bits show the length of each design variable, when
performed by using:

$$2^{(i-1)} < (x_i' - x_i'') \times 10^k \leq 2^L$$

(3.1)

where:

- $L$ is the length of binary string for each gene, Subscript $i$ denotes the $i^{th}$ design
  variables
• \( x'_i \) and \( x''_i \) are the lower and upper bound values of the search domain, respectively.

• \( k \) is the required decimal precision.

Decoding of each sub-binary string will produce the corresponding decimal digits that then represent real values of the variables. The conversion of the binary string into real number is performed in two steps:

1. Translate the binary sub-string from the base 2 to the base 10, which is performed using:

\[
I_i = \sum_{j=1}^{r} C_j \times 2^{j-1}
\]  
(3.2)

where:

• \( I_i \) equal an integer mapping of a binary string

• \( C_j (j = 1, 2, \ldots, r) \) is the binary string as \( \langle C_r, C_{r-1} \ldots C_2, C_1 \rangle \).

2. Find the corresponding real number, which can be obtained by using:

\[
x_i = x'_i + \frac{(x''_i - x'_i)}{(2^r - 1)} \times I_i
\]  
(3.3)

where:

• \( x_i \) is the real number of \( i^{th} \) design variables.

For example, by using the above decoding process, the real values of the variables \( x_1 \) and \( x_2 \) for individual number 1 in Figure 3.2 can be obtained as follows:

Individual no. 1: 0110110111

First convert each sub-string from the base 2 to the base 10 according to eq.(3.2).  

\[
I_1 = 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13
\]
\[ I_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 23 \]

So both \( x_1 \) and \( x_2 \) are between 4 and 7. Then the corresponding real number \( x_1 \) and \( x_2 \) can be calculated according to eq. (3.3).

\[
x_1 = 4 + \frac{(7 - 4)}{(2^5 - 1)} \times 13 = 5.3 \quad \text{and} \quad x_2 = 4 + \frac{(7 - 4)}{(2^5 - 1)} \times 23 = 6.2
\]

After the corresponding real number of each binary string in the population is calculated, the fitness of each individual can be evaluated according to the objective function.

For maximisation problems, the fitness of the string can be equal to the objective value of the string. However, for minimisation problems, the goal is to find a solution having the minimum objective function value. Thus minimisation of an objective function is treated exactly like maximization problems except that the object function is first multiplied by (-1). This is because maximizing the negative of an objective function is the same as minimizing the objective function.

### 3.3.2 Selection Procedure

The selection of chromosomes to produce offspring plays an extremely important role in a GA. This operation decides which chromosome will survive in the next generation and how many copies of it should be produced in the mating pool according to their corresponding fitness. The fitter the chromosomes are, the more chances they will have to be selected. For example Table 3.1 shows the population of four chromosomes (as seen in the previous example in Figure 3.2).
The results of decoding of each chromosome and the corresponding real number of \( x_1 \) and \( x_2 \) are shown in columns 3 and 4 of Table 3.1, respectively. The fitness function \( f(x) = (x_1^2 + 2x_2) \) is used to assign a fitness value to each individual and results are shown in column 5. Column 6 shows the reproduction probability \( P_i \) (between 0% and 100%), which is calculated by:

\[
P_i = \frac{f_i}{\sum_{j=1}^{n} f_j}
\]

where:

- \( f_i \) equals the fitness value of the \( i^{th} \) individual
- \( n \) is the population size, i.e. the number of individual in the population.

In order to reproduce offspring, parents need to be selected. There are a number of selection algorithms commonly used such as roulette wheel selection, scaling techniques, tournament, elitist model and ranking methods [2,27]. The selection procedure used in the current research is a roulette-wheel selection.
The roulette-wheel is the classic and more popular fitness-proportionate selection [2]. As an example, consider the roulette wheel in Fig. 3.3, which is based on the previous example.

Each individual receives a slice of the wheel proportional to their relative percentage fitness values. In order for the individuals to be selected and placed into the mating pool, the wheel is spun and the individual on which it stops gets selected. Statistically speaking, individuals with a higher fitness value will have a greater chance of being selected. For example, from Fig. 3.3, it is obvious that the individual number 2 is the fittest and should be selected with higher probability for reproduction. To implement this selection procedure the following steps are taken [27]: First the cumulative probability $q_n$ for each individual is calculated by:

$$q_n = \sum_{j=1}^{n} p_j$$  \hspace{1cm} (3.5)

where:

- $n$ is the population size

then a random number $r$, between 0 and 1 is generated and compared with the
cumulative probability of selection \( q_n \). The appropriate individual \( n \) is selected and copied into the mating pool if the random number \( r \leq q_n \). It is possible for an individual to be selected more than once. As shown in Figure 3.3, assume that the individuals number 1 and 4 get one reproduction each, individual number 2 reproduces twice and number 3 the least fit individual fails to reproduce. This process of reproduction confirms the evolutionary principle know as "survival of the fittest". This selection process yields a new population as shown Table 3.2.

**Table 3.2. Results of Selection**

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Chromosome</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0110110111</td>
</tr>
<tr>
<td>2</td>
<td>1110111010</td>
</tr>
<tr>
<td>3</td>
<td>1110111010</td>
</tr>
<tr>
<td>4</td>
<td>1000001001</td>
</tr>
</tbody>
</table>

Once the mating pool has been formed, the crossover and mutation operators are applied to recombine the individuals in the population and form a new population of individuals that is passed onto the next generation.

### 3.3.3 Mating / Crossover Operator

After reproduction, the newly reproduced individuals in the mating pool will be selected randomly and mated in pairs. If the parents are allowed to mate, crossover operator is employed to exchange genes between the two parents to generate two offspring, which will take the place of their parents within the population. If they are not allowed to mate, the parents are placed into the next generation unchanged. There are three types of crossover operations associated with binary representation [27]:
One-point crossover is the simplest form of this operator and most commonly used in binary coded GA [2,13,14,18]. Throughout the current research, only one-point crossover is used.

The one-point crossover operator picks a random point within the chromosomes then the genes up to that point are swapped between the two chromosomes. The chromosomes undergo crossover process with probability, $p_c$ (crossover rate). For each chromosome in the population a random number $r$ between 0 and 1 is generated, and the individuals undergo crossover if the random number is smaller than the crossover rate [27]. The crossover probability controls the search effect of crossover. The higher the crossover rate, the higher the search effect. Crossover is not necessarily applied to all pairs of chromosomes selected for mating, it occurs with probability that is specified as one of the GA’s parameters. A typical value for the probability of crossover is between 0.7 and 1 [2].

Figure 3.4 uses the chromosomes from the previous example (Table 3.2) to illustrate one-point crossover. Assume that the individuals number 1 and 2 are randomly selected to mate (to be crossed).

Figure 3.4 shows how one-point crossover creates two new individuals through swapping all the bits beyond the randomly chosen crossover point. After performing crossover the new chromosomes then replace the parents in the next population. Table 3.3 shows the population after crossover process.
3.3.4 Mutation Operator

Mutation is the final operation in GA that alters one or more gene values in a chromosome. This operation safeguards the process from a complete premature loss of valuable genetic material during selection and crossover. When a chromosome is selected for mutation, a random choice is made of some of the genes of the chromosome, and these genes are modified. In terms of binary string, it sweeps down the list of bits, replacing each bit by a randomly selected bit according to a mutation probability ($p_m$).

The mutation operator plays a secondary role in GAs, therefore, the probability of
chromosomes having mutation is taken as a low value [2]. The total number of bits undergoing mutation is equal \((p_m \times \text{total number of bits} \times \text{populationsize})\). For each bit in the bit string, the operator generates a random number \(r\) between 0 and 1. If the random number is smaller than the mutation rate, then the operator replaces the bit by a randomly generated bit (either 0 or 1).

Consider the previous example (Table 3.3), assume that the mutation probability \(p_m = 0.05\). Since there are four individuals, each with eight bits, then following mutation is accepted \(0.05 \times 8 \times 4 = 1.6\). Every bit in the population has the same probability to be mutated, as a result: \(8 \times 4 = 32\) random numbers between 0 and 1 are generated and compared with the mutation rate \((p_m = 0.05)\). If the random number \(r < 0.05\) then that bit will be mutated. Assume that the random number obtained points to the fourth bit and the seventh bit of individuals number 2, and 3 respectively. Since the fourth and the seventh bits of individuals number 2, and 3 was originally a 0 and 1, the complement of those bits are 1 and 0 respectively. Thus the resulting yields two new chromosomes as shown in Figure 3.5.

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Selected for mutation</th>
<th>After mutation</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1110110111</td>
<td>1111110111</td>
</tr>
<tr>
<td>3</td>
<td>1110111010</td>
<td>1110110010</td>
</tr>
</tbody>
</table>

Fig. 3.5 Mutation Procedure

A bit change in the least significant bit of a gene would not cause much change in the design variable value. A bit change in the most significant bit of the same gene
would cause significant change in the design variable value. Table 3.4 shown the results after the initial population has undergone all the steps of the GA process in first generation.

Table 3.4. Final Results After the Evolution of the Initial Population

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Chromosome</th>
<th>Value</th>
<th>Corresponding Real number</th>
<th>Fitness \[ f(x) = (x_1^2 + 2x_2) ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0110111010</td>
<td>13 26</td>
<td>5.3 6.5</td>
<td>41.1</td>
</tr>
<tr>
<td>2</td>
<td>1110111111</td>
<td>29 31</td>
<td>6.8 7.0</td>
<td>60.3</td>
</tr>
<tr>
<td>3</td>
<td>1110110010</td>
<td>29 19</td>
<td>6.8 5.8</td>
<td>57.9</td>
</tr>
<tr>
<td>4</td>
<td>1000001001</td>
<td>16 9</td>
<td>5.5 4.9</td>
<td>40.0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td></td>
<td>199.3</td>
</tr>
</tbody>
</table>

From Table 3.4 it is clear that the chromosomes within the final population are much better as a whole than the chromosomes within the initial population. Also, the best chromosome in the final population will generally be the optimal solution if the GA was run for enough generations.

3.4 Concluding Remarks

This chapter provided an overview of the genetic algorithms. The focus of this chapter is to offer the reader an understanding of the underlying concepts of the proposed algorithm.

Genetic algorithms are inspired by the biological principle of natural selection or survival of the fittest. The majority of the chapter describes the primary components of the evolutionary process: initialisation, evaluation, selection, crossover and mutation. Initialisation is the process of randomly generating a population of candidate solutions.
All of the candidate solutions are encoded as artificial individuals. Each individual in the population are evaluated according to the objective function to establish fitness. Individuals are selected into a mating pool based on fitness. The individuals in the mating pool are exchanged and altered using crossover and mutation operations resulting in a new population carried over to the next generation. The evolution continues for a predetermined number of generations. Also, the best individual in the final population will generally be optimal solution if the genetic algorithm was run for enough generations.
4.1 Introduction

The optimal design of steel truss structures has been an active area of research in the field of structural optimisation mainly due to the fact that the truss systems have been used extensively and reliably for structures that cover large open areas, bridges, transmission towers, ship masts, offshore platforms and roof supports. In addition to their practical importance as useful structures, finite-element code for truss systems can easily be written for analysis and testing. Moreover, truss structures varying from simple to highly complex non-linear and indeterminate forms can be created. Thus, these structures provide excellent test cases for the study of optimisation techniques [23].

In this chapter, the different types of truss structural optimisation problems and the application of genetic algorithms for truss optimisation are briefly discussed, based on published studies.
4.2 Design of Truss Structures

The most important considerations in the structural design of trusses are safety \((i.e.\) the structure must carry loads safely), economy \((i.e.\) the structure should be economical in material, construction and overall cost) and feasibility \((i.e.\) the structure should be designed to allow simple fabrication and construction). The improvement of safety is covered by analytical checks on the structure's behaviour under the external loadings. Economy, on the other hand, is improved by optimum design. The analysis of truss structures is discussed in the next chapter.

The optimum design of truss structures typically involves finding a set of design variables to minimise the cost while satisfying performance constraints that are related to the design and the behaviour of the structure. Generally, cost is directly related to the total weight of the truss structure. Therefore, the weight of the truss structure has been considered as an objective function in several optimal designs of truss structures. The member properties of truss structures are usually assumed as design variables. The selection of the design variables is subject to the type of constraints, such as geometric constraints \((i.e.\) minimum and/or maximum areas or thicknesses), stress constraints \((i.e.\) maximum allowable stresses either tensile or compressive) and displacement constraints \((i.e.\) minimum and/or maximum values). From this point of view, many researchers, among them Rajan [1], Sabaghalvani and Hadi [6], Krishnamoorthy [8], Erbatur \textit{et al.} [11], Saka [67] and Rajeev \textit{et al.} [77] have investigated methods seeking minimum weight design and considering different constraints.

4.3 Types of Truss Structural Optimisation Problems

Truss structures can be optimised in different ways. The type of design variables, loadings and constraints, may classify the truss structural optimisation problem. In
In general, there are three distinct classes of truss optimisation problems. These are:

- **Size optimisation**
- **Shape optimisation**
- **Topology optimisation**.

In practice, each class of the optimisation problem requires a different method or strategy of solution. During size optimisation, the domain is fixed and does not change. In shape optimisation, the domain shape is variable and the topology is fixed. Sub-optimal solutions may result from size and shape optimisation process since they do not necessarily have an optimum starting topology. To overcome this drawback topology optimisation must also be considered. The ideal solution involves the simultaneous optimisation of size, shape and topology. This is sometimes called layout optimisation [78].

Numerous literature exists that goes into the details of the structural optimisation problems, among them Vanderplaats [23], Hassani and Hinton [78] and Bendsoe [79].

In the following sections the details and basic formulations of these optimisation problems are briefly described.

### 4.3.1 Size Optimisation of Trusses

In the size optimisation of truss structural problems, the cross-sectional areas of all member elements are modified to meet the design requirements. The coordinates of the nodes and connectivity among various members are considered to be fixed. The elements of trusses are usually chosen from a finite set of rolling profiles and parameters of bars determined by commercially available discrete values. This fact has to be taken into account in the optimisation procedure.

The common formulation of size optimisation problems for truss structures with
discrete cross-section areas where the minimum weight is taken as the objective function, can be expressed as follows [11]:

\[
\text{Minimise:} \quad W(A) = \sum_{i=1}^{ne} A_i l_i \rho \tag{4.1}
\]

Subject to:
\[
g_k(A) \leq 0 \quad , \quad k=1,2,\ldots,m \tag{4.2}
\]
\[
\left\{ \begin{array}{l}
A_i^l \leq A_i \leq A_i^u \\
A_i \in S \quad , \quad i=1,2,\ldots,ne
\end{array} \right. \tag{4.3}
\]

where:

- Eq. (4.1) defines the weight of the truss structure. \( ne \) is the total number of elements in the structure, \( A_i \) and \( l_i \) are, respectively, the cross-section area and length of the \( i^{th} \) member and \( \rho \) is the weight density of material.

- Eq (4.2) represents both equality and inequality constraints, which the design must satisfy. The constraints may be limits upon parameters such as section stresses, nodal displacements, natural frequencies, and stability.

- In the Eq (4.3), \( A = \{ A_1, A_2, \ldots, A_{ne} \}^T \) is the sizing variables (cross-sectional areas) vector. The values \( A_i^l \) and \( A_i^u \) are, respectively, the lower and upper bounds of the size variables, and \( S \) represents a list of discrete values to be assigned to size variables.

### 4.3.2 Shape Optimisation of Trusses

The shape optimisation of truss structures involves change of member length, hence, change of joint location. Therefore, the change in nodal coordinates, are taken as design variables, along with member areas [7]. The lengths of members are represented in the
form of coordinates. Shape optimisation of truss structures is a typical example of optimisation problem in which the design variable is a combination of continuous and discrete values, since the node coordinates vary continuously while the cross-sectional areas of the elements can only take the given values of standard profiles.

The general formulation of the shape optimal design problem for a truss structure is given as follows [7]: find the cross-sectional areas \( A = \{A_1, A_2, \ldots, A_{ne}\}^T \) and the joint coordinates \( x = \{x_1, x_2, \ldots, x_{nc}\}^T \) that minimise the total weight of the structure

\[
\text{Minimise: } \quad W(A, x) = \sum_{i=1}^{ne} A_i l_i \rho, \quad l_i = \left[ \sum_{r=1}^{3} (x_{ir}^a - x_{ir}^b)^2 \right]^{1/2} \tag{4.4}
\]

Subject to:

\[
g_k(A, x) \leq 0, \quad k = 1, 2, \ldots, m \tag{4.5}
\]

\[
x_s^l \leq x_s \leq x_s^u, \quad s = 1, 2, \ldots, nc \tag{4.6}
\]

\[
\begin{cases}
A_i^l \leq A_i \leq A_i^u \\
A_i \in S, \quad i = 1, 2, \ldots, ne
\end{cases} \tag{4.7}
\]

where:

- Eq. (4.4) defines the weight of the truss structure. \( ne \) is the total number of elements in the structure, \( \rho \) is the weight density of material. \( l \) is the vector of member's lengths, which are function of joint coordinates and \( x_{ir}^t \) is the \( r \)th coordinate of joint \( t \) for the \( i \)th member.

- Eq (4.5) represents both equality and inequality constraints, which the design must satisfy.

- In the Eq. (4.6), \( x \) is the joint coordinate variable vector and \( nc \) is the number of coordinates, which are allowed to change. \( x_s^l \) and \( x_s^u \) are lower and upper
bounds values respectively.

- In the Eq (4.7), \( A \) is the sizing variables vector. The values \( A_i^l \) and \( A_i^u \) are, respectively, the lower and upper bounds of the size variables, and \( S \) represents a list of discrete values to be assigned to size variables.

### 4.3.3 Topology Optimisation of Trusses

In the topology optimisation of truss structural problems, the absence or presence of the structural elements can be represented by integer variables such as 0 and 1, where 0 represents absence and 1 represents presence of that element. For discrete truss structures the topology is not only concerned with how nodes are connected to each other. Topology must also consider how many nodes are to be placed and how they are to be supported. [1]. Topology design is the most difficult task in comparison with its size and shape design. The reason is that the structural model is itself allowed to vary during the design process. In addition, the problem can have singular global optima that cannot be reached by assuming a continuous set of variables [14].

Within the area of truss topology optimisation, most of the work published in the past is based on the ground structure approach (Fig. 4.1), containing all possible connections between all the nodes in the domain [14]. The total member connectivity in the structure can be calculated by: \( \frac{\nu \times (\nu - 1)}{2} \), where \( \nu \) is the total number of node points.

The optimisation process aims to eliminate the unnecessary members and joints from the initial highly connected ground structure that would result in a least weight structure, and also satisfy the prescribed design constraints.
A primary topology optimisation problem for a truss structure with discrete cross-section areas where the minimum weight is taken as the objective, can be stated as follows [14]:

Minimise:  

\[ W(A, T) = \sum_{i=1}^{ne} A_i l_i T_i \]  

Subject to:  

\[ g_k(A, T) \leq 0 , \quad k=1,2,\ldots,m \]  

\[
\begin{cases} 
A_i^l \leq A_i \leq A_i^u \\
A_i \in S, \quad i=1,2,\ldots,ne
\end{cases}
\]  

where:

- Eq. (4.8) defines the weight of the truss structure. \( ne \) is the total number of elements in the structure, \( A_i \) and \( l_i \) are, respectively, the cross-section area and length of \( i^{th} \) member and \( T_i \) denotes an integer variable that assume values of 1 or 0, respectively, for presence or absence of an element.
• Eq (4.9) represents structural response constraints.

• Eq (4.10), as already noted represent the sizing variables vector. The values $A^l_i$ and $A^u_i$ are, respectively, the lower and upper bounds of the size variables, and $S$ represents a list of discrete values to be assigned to size variables.

Although the above three optimisation problems are discussed separately, there is a great degree of coupling between these three design problems, and a good deal of work has been performed with the intent to integrate two or more of these design problems. Various techniques have been developed to solve the above three optimisation problems. Genetic algorithms-based methodologies have proved to be one of the most effective and robust optimisation techniques, for considering the above three problems separately or simultaneously because in GA, the gradient of the objective function and the constraint functions are not needed to find optimal solutions. Therefore, GAs are generally suitable for problems with discrete design variables as well as continuous design variables. The applications of GAs for optimal design of truss structures are presented in the following section.

### 4.4 Optimal Design of Trusses Using Genetic Algorithms

A variety of structural design problems have been solved by the use of genetic algorithms (GAs). Application of a simple GA to the optimal design of a 10-bar plane truss structure was first put forward by Goldberg and Samtani [16] in 1986.

The use of GAs for search and optimisation problems, and its feasibility are evident in Goldberg and Samtani's [16] example. They used binary strings for representing the design variables fitness based reproduction process, a simple crossover and mutation to guide the search process. Transforming a mathematical programming
problem into an unconstrained problem by use of on exterior penalty function is formulated in this example. The objective function was to minimise the total weight of the structure of using minimum and maximum stress constraints on each member. Cross-sectional areas of members were considered as the design variables and the coordinates of the nodes and the connectivity among various members was considered to be fixed.

Goldberg and Samtani [16] found that mutation plays a secondary role in the operation of GAs and it is needed because, even though selection and crossover effectively search and recombine notions, occasionally they may become overzealous and they could lose some potentially useful genetic material. Due to the secondary importance of the mutation, the probability of individuals having mutation is taken as a low value.

In 1992 Rajeev and Krishnamoorthy [3] applied simple GA to discrete optimum design of plane and space truss structures based on linear analysis. The weight of the structure was considered as the objective function. Member stresses and nodal displacements were considered as constraints in the formulation of the design problem. They presented a decoding scheme for discrete design variables for the use in GAs based upon an assumption that design variables are required from a set of available values, specified by the lower and upper bounds. Furthermore, since GAs were developed to tackle unconstrained optimisation problems, a penalty-based transformation method, which depends on the degree of constraint violation have been presented to transform the constrained optimisation problem into an unconstrained problem. Constraint violation has been found to be quite suitable for a parallel search using GAs. This paper also described the basic formulation of GA in greater detail using a three bar truss structure. Rajeev and Krishnamoorthy [3] concluded that the
consideration of design variables as discrete quantities is essential in the optimisation of most structural system.

In 1993 Adeli and Cheng [4] applied GA to obtain minimum weight design of a 72-bare space truss structure. The structural members were divided into 16 groups, and the same cross section was assumed for each group. In this paper, the design variables (cross sectional areas) were considered as continuous values. Limitation on member stresses, nodal displacements were considered as constraints. A quadratic penalty function was used to transform this constrained optimisation problem into an unconstrained one. Adeli and Cheng [4] found that the solution usually converges to become infeasible if a small value is used for the penalty-function coefficient. However a large value for the penalty-function coefficient will cause the solution to oscillate. Moreover there is a problematic characteristic of the penalty-function method. When integrating a GA with the penalty function method, the solution normally goes into infeasible region directly after the initial, or first few iterations. They found that the performance of GA depends on the high survival rate of good string patterns or schemata during the crossover phase.

In 1994 Hajela and Lee [14] used GA in the topological design of truss structures. Their objective was to generate the minimal weight of structures with constraints on member stresses, nodal displacements and element buckling. They used a structural universe of all possible connections between all the nodes in the domain and assigned 0 bit to represent the absence and 1 bit to represent the presence of an element. The topological optimisation was performed in two-stages. To identify stable topological configurations in the first stage, kinematics stability requirements are used. Member resizing and addition/removal of members were considered in the second stage.

Hajela and Lee [14] presented two examples in their paper; a 14-bar truss and a
bridge-type truss. They found that the population size is fundamental to success of the genetic search process because small population sizes converge very rapidly. Therefore only a few design alternatives can be explored. On the other hand, excessively large populations imply long waiting times for convergence and significant increases in computational costs. They concluded that the genetic search procedure is a valuable exploratory tool for analysing topologies in a discontinuous design space for several reasons. The member-sizing problem is related to a locally convergent optimisation algorithm. Computational resources are considerably reduced. There is less function evaluation required to generate optimal topologies because more traditional search techniques are used, as well.

In 1995, Ohsaki [13] studied the use of GAs to determine the optimal topologies of trusses with member stresses and displacements under static loading conditions. The examples used in this paper include a 3-bar truss and a 20-bar plane truss. Ohsaki [13] emphasized the importance of considering nodal cost (number of nodes). It was also shown that an optimal topology with smaller numbers of nodes and members could be found by adding nodal cost in the objective function. For each member an indication of its existence is introduced a topological bit. With small number of members the bit allows for a rapid convergence of the solution to an optimal topology. This technique starts from a ground structure with sufficient number of members and nodes. In this ground structure unnecessary members and nodes are removed to achieve an optimal topology. However if this is applied to large trusses, then the proposed algorithm may require substantial computational effort.

In 1995 Rajan [1] applied GA to the design of truss structures by combining their size, shape and topology. Rajan applied all three optimisation methods to a 6-node truss and a 14-node truss. The cross sectional areas of each of the members were defined by
the use of discrete and continuous values. In order to handle skeletal structures the hybrid shape optimisation methodology, also used in continuum structures, was made adaptive and the nodal locations were treated as design variables. In the context of topology design the member connectivity and the support conditions of the elements were treated as Boolean design variables, (1 for presence and 0 for absence). To deal with zero force members, unfeasible and unstable structures Rajan [1] used a penalty function as the fitness, and exception handling was seen to deal with the aforementioned characteristics. In addition to this, in order not to compute the fitness function of the chromosome, the history of each chromosome was recorded twice.

Rajan [1] found that the effect of population size, the lack of a convergence criterion, and the effect of probability values for crossover and mutation are considerable. This finding was not only associated with the arrival of the best possible solution but also utilising an efficient method. In order to maintain sufficient diversity so that the best possible topology can be obtained, a dynamically changing penalty approach was necessary. He also found that the possibility exists for the truss to become unstable during the optimisation process. This is due to the generated string, even when the base structure remains stable. Instability does not occur in the sizing optimal design. This is because in the sizing process involving linear response, the member cross-sectional areas are the only variable able to change. Finally, in shape optimal design, the problem of instability can be controlled somewhat by using good judgment to select the upper and lower bounds.

In 1997 Rajeev and Krishnamoorthy [8] proposed a GA for designing the optimum topology, configuration, and sizes of cross-sectional parameters of truss structures. They presented two methods. The first method is a two-phase method. The reduction of the size of the search space by way of arriving at lower bound values for
design variables is the main objective of the two-phase method. The second method was based on a variable string length genetic algorithm (VGA). They argued that using all three types of truss variables; topology, configuration, and sizes of cross-sectional parameters, made the problem too complex. The wide range of variations in the nature of the design variables includes both discrete and continuous variables. Therefore in many situations it may be necessary to solve problems with fixed topology and/or fixed configuration, which demand an efficient method to handle such different situations.

Rajeev and Krishnamoorthy [8] used a 10-bar truss, an 18-bar truss and a Microwave antenna tower, to illustrate the capability of their method and to consider many practical aspects in design and construction as well as the robustness of their proposed methodology and its usefulness in solving large problems. Rajeev and Krishnamoorthy [8] solved the size and configuration optimisation problems by using their two-phase method, which considered discrete size variables and continuous configuration variables. They further improved their two-phase method by using a VGA, so that variation in topology apart from size and configuration could also be considered. By using appropriate genetic coding schemes, the design space, which is represented in parametric form of variables, are transformed into genetic space.

4.5 Concluding Remarks

In this chapter, the different types of truss structural optimisation problems and the application of genetic algorithms (GAs) for optimal design of various truss structure problems are discussed.

For optimal design of truss structures GAs are classified into the three main categories, namely: size, shape, and topology optimisation. In order to handle the sizing
optimal design problem, the design variables are to be considered as discrete values. In the shape optimisation of trusses, member sizes and changes in nodal coordinates are the design variables, which are a mix of discrete and continuous values. In the topology optimisation of truss structures, a structural universe of all possible connections between all the nodes in the domain is used. Unnecessary members and nodes are removed to find an optimal topology.

The studies discussed above have shown that GAs are an effective tool for the optimal design of truss-structures. Virtually all of the optimum design algorithms developed in the area of optimal design of truss structures has dealt with linear behaviour. However, in structural optimisation there are cases, such as long span and slender structures (e.g. suspension bridges), where the behaviour of these types of structures under external loading requires non-linear analysis. None of the previous work is based on non-linear analysis using GA as the discrete optimisation tool.

In this research, a optimum design methodology is developed based on GA for geometrically non-linear truss structures with discrete design variables. The proposed algorithm is discussed in the next chapter.
5.1 Introduction

In the previous chapter, the application of genetic algorithms (GAs) for optimal design of various truss structure problems was discussed. None of the previous work is based on non-linear analysis using GAs as the discrete optimisation tool.

This chapter describes a new optimum design algorithm developed for geometrically non-linear plane and space truss structures composed of elements that are chosen from a given set of cross-sections. This is achieved by coupling the GA with a geometrically non-linear analysis procedure. In order to make comparison with the optimum design obtained for plane and space trusses based on non-linear analysis, an optimum design algorithm was also developed for plane and space trusses based on linear analysis.

The author has already presented the proposed algorithm for discrete size optimisation of plane truss structures [6], where the effect of geometrical non-linearity is considered. It was shown by [6] it is possible to obtain realistic solutions by including the geometric non-linearity in the formulation of design problem.
The proposed GA is based on a roulette-wheel reproduction scheme, a one-point crossover, and a standard mutation scheme. The weight and/or volume of the overall truss structure is considered as the objective function. Limitation on member stresses, nodal displacements and member buckling are included as constraints in the design. Since GAs are directly applicable only to unconstrained optimisation problems, a penalty function method based on the violation of normalised constraints is used to transform the constrained optimisation problem into an unconstrained one. In order to solve the non-linear response of trusses it is typically necessary to use an iterative method. In this thesis an incremental load approach with a Newton-Raphson type of iterations was used for the geometrically non-linear analysis and for developing the optimisation algorithm.

In the following sections, the formulations of both linear and geometrically non-linear analysis procedures for two-dimensional and three-dimensional truss structures are briefly discussed, followed by a description of the optimisation method to minimise the weight of the truss structures with discrete design variables, while satisfying member stresses, nodal displacements and member buckling constraints. This chapter also provides a detailed description of all the necessary steps of the proposed design procedure that were developed for solving the optimisation problems given in chapter 6. The proposed algorithm was written in MATLAB Version 5.3 (2001).

5.2 Truss Structure Analysis

A truss consists of straight members connected at their ends and arranged in such a way as to form a rigid framework capable of supporting loads. Each member of a truss is considered as a two-force member subjected to axial forces only, with no applied moments allowing the members to rotate freely at the joints so as to prevent restraint
moments [80].

In this research, two classes of truss structure optimisation problems are studied. The first class of optimisation problems is formulated based on linear analysis and the second class of optimisation problems is based on geometrically non-linear analysis whereas the latter type of optimisation problem has not been investigated before using GAs as the optimisation tool. In the optimal design process, the linear or non-linear analysis procedures are used for predicting the stress and deformation response of the truss structures under the external loads so as to evaluate the fitness function.

Numerous literature exists that go into detail of the truss structural analysis, among them Kwon and Bang [80], Levy and Spillers [81], Crisfield [82] and Przemieniecki [83].

In the following sections, the linear and non-linear analyses of plane and space truss structures are briefly described. The source code for each analysis procedure is presented in Appendices B, C, D and E.

5.3 Linear Analysis of Trusses

In linear structural analysis, it is assumed that the joint displacements of the structure under the applied loads are small with respect to the original joint coordinates. Thus, the geometric changes in the structure can be ignored and the overall stiffness of the structure in the deformed configuration can be assumed to be equal to the stiffness of the undeformed structure. For a linear truss system, the static displacements can easily computed by solving the set of linear simultaneous stiffness equations [80], presented below as equation (5.1):

\[
\{F\} = [K]\{U\} \tag{5.1}
\]
where:

- \( \{ F \} \) is the vector of applied joint load (\( N \times 1 \))
- \( \{ U \} \) is the vector of nodal displacements (\( N \times 1 \))
- \( [K] \) is the global stiffness matrix of the structure (\( N \times N \))
- \( N \) is the total number of degrees of freedom

The terms in the global stiffness matrix \([K]\) depend on the degree of freedom of each node. These matrix terms are constants, which do not change as the linear structure deforms.

5.3.1 Plane Truss

To analyse plane truss structures, the relationship between the forces and displacements at each end of a single truss element is needed. Consider a plane truss structural element connecting joints 1 and 2 inclined at an angle \( \phi \) with the horizontal axis \( \bar{x} \) as shown in Fig. 5.1. This element has two end nodes with two degrees of freedom per node \((\bar{u}, \bar{v})\). \((x, y)\) refer to the system axes and \((x, y)\) as the member axes.

The transformation matrix between the \(xy\) and \(\bar{xy}\)-coordinate systems is given below [80]:

\[
\begin{bmatrix}
  \bar{u}_1 \\
  \bar{v}_1 \\
  \bar{u}_2 \\
  \bar{v}_2
\end{bmatrix} =
\begin{bmatrix}
  \lambda & \mu & 0 & 0 \\
  -\mu & \lambda & 0 & 0 \\
  0 & 0 & \lambda & \mu \\
  0 & 0 & -\mu & \lambda
\end{bmatrix}
\times
\begin{bmatrix}
  u_1 \\
  v_1 \\
  u_2 \\
  v_2
\end{bmatrix}
\]

(5.2)

where:

- \( \lambda = \cos \phi \) and \( \mu = \sin \phi \).
The Eq. (5.2) can be written in compact form as:

$$\{u\} = [T][\tilde{u}]$$  \hspace{1cm} (5.3)

where:

- $[T]$ is the transformation matrix.

The discussion above deals with displacements components of the member at positions 1 and 2. However, similar results also apply to the respective $x$ and $y$ force components. Thus the transformation equation between the force components and system axis can be expressed in matrix form as follows:

$$\{f\} = [T][\tilde{f}]$$  \hspace{1cm} (5.4)

where the force matrices are given by:
In the above matrices \( \overline{f}_{1x}, \overline{f}_{1y} \) and \( \overline{f}_{2x}, \overline{f}_{2y} \) represent \( \bar{x} \) and \( \bar{y} \) force components at positions 1 and 2 of the member, respectively, and \( f_{1x}, f_{1y} \) and \( f_{2x}, f_{2y} \) represent corresponding \( x \) and \( y \) force components at these positions. Now assuming member 1-2 to be subjected to a uniaxial loading, thus \( f_{1y} = f_{2y} = 0 \). The equilibrium equation in this case will give:

\[
\{f\} = [K^e]\{u\} \tag{5.5}
\]

in which \( [K^e] \) denotes the element stiffness matrix of the member and can be expressed as:

\[
[K^e] = \frac{AE}{l} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.6}
\]

where:

- \( A, E \) and \( l \) are the area of cross-section, elastic modulus and the length of the member, respectively.

The concept of strain energy can be applied to transform the element stiffness matrix from the \( xy \)-coordinate system to the \( \bar{x}\bar{y} \)-coordinate system where the strain energy can be expressed as:
in terms of the $xy$-coordinate system.

Substituting equation (5.5) into equation (5.7) yields:

\[ S = \frac{1}{2} \{ u \}^T [K^e] \{ u \} \tag{5.8} \]

The strain energy can now be expressed in terms of the $\bar{x}\bar{y}$-coordinate system as follows:

\[ S = \frac{1}{2} \{ \bar{u} \}^T \{ \bar{\bar{K}}^e \} \{ \bar{u} \} \tag{5.9} \]

where:

- $[\bar{\bar{K}}^e]$ is the transformed element stiffness matrix in terms of the $\bar{x}\bar{y}$-coordinate system.

The strain energy in equation (5.9) should be the same as that in equation (5.8) because strain energy is independent of the coordinate system. Equating equation (5.8) to equation (5.9) shows that:

\[ [\bar{\bar{K}}^e] = [T]^T [K^e] [T] \tag{5.10} \]

Substitution of equations (5.2) and (5.6) into equation (5.10) results in the transformed stiffness matrix:
\[
[K^e] = \frac{EA}{l} \begin{bmatrix} 
\lambda^2 & \lambda \mu & -\lambda^2 & -\lambda \mu \\
\lambda \mu & \mu^2 & -\lambda \mu & -\mu^2 \\
-\lambda^2 & -\lambda \mu & \lambda^2 & \lambda \mu \\
-\lambda \mu & -\mu^2 & \lambda \mu & \mu^2 
\end{bmatrix}
\]  
(5.11)

where:

- \( A, E \) and \( l \) are the area of cross-section, elastic modulus and the length of the member, respectively.
- \( \lambda = \cos \phi \) and \( \mu = \sin \phi \).

Equation (5.11) provides a general transformation relation for the stiffness matrix between the member and system axes for the nodal degrees of freedom \( \{u_1, v_1, u_2, v_2\} \).

A relationship between externally applied loads and node point displacements for the entire structure can be written in matrix form:

\[
\begin{bmatrix} 
\bar{f}_{1x} \\
\bar{f}_{1y} \\
\bar{f}_{2x} \\
\bar{f}_{2y} 
\end{bmatrix} = \frac{EA}{l} \begin{bmatrix} 
\lambda^2 & \lambda \mu & -\lambda^2 & -\lambda \mu \\
\lambda \mu & \mu^2 & -\lambda \mu & -\mu^2 \\
-\lambda^2 & -\lambda \mu & \lambda^2 & \lambda \mu \\
-\lambda \mu & -\mu^2 & \lambda \mu & \mu^2 
\end{bmatrix} \begin{bmatrix} 
u_1 \\
v_1 \\
u_2 \\
v_2 
\end{bmatrix}
\]  
(5.12)

### 5.3.2 Space Truss

The detailed workings for the plan truss element of Fig. 5.1, is readily applied to a space truss element. The element stiffness matrix in terms of the global coordinate system is obtained in the same way as given in equation (5.10). Consider the space truss member 1-2 shown in Fig. 5.2.
Fig. 5.2 A space truss structural element [80]

The local or member axes are presented by \((x, y, z)\) and the global or system axes by \((\bar{x}, \bar{y}, \bar{z})\) respectively. In these circumstances, the stiffness matrix \([K^e]\) similar to equation (5.6) becomes:

\[
[K^e] = \frac{AE}{l} \begin{bmatrix}
1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix} \tag{5.13}
\]

for the nodal degrees of freedom of \(\{u_1, v_1, w_1, u_2, v_2, w_2\}\). The transformation matrix between the two coordinate system \([T]\) becomes:
\[
[T] =\begin{bmatrix}
\lambda_x & \mu_x & \nu_x & 0 & 0 & 0 \\
\lambda_y & \mu_y & \nu_y & 0 & 0 & 0 \\
\lambda_z & \mu_z & \nu_z & 0 & 0 & 0 \\
0 & 0 & 0 & \lambda_x & \mu_x & \nu_x \\
0 & 0 & 0 & \lambda_y & \mu_y & \nu_y \\
0 & 0 & 0 & \lambda_z & \mu_z & \nu_z \\
\end{bmatrix}
\] (5.14)

where:

\[
\lambda_x = \cos(\phi_{xx}), \quad \phi_{xx} \text{ is angle between } x, x
\]

\[
\mu_x = \cos(\phi_{yx}), \quad \phi_{yx} \text{ is angle between } y, x
\]

\[
\nu_x = \cos(\phi_{zx}), \quad \phi_{zx} \text{ is angle between } z, x
\]

\[...
\]

\[
\nu_z = \cos(\phi_{zz}), \quad \phi_{zz} \text{ is angle between } z, z
\]

Using equation (5.10) and (5.14), the stiffness matrix in global coordinate may be written as:

\[
[K^e] = \frac{EA}{l} \begin{bmatrix}
\lambda^2 & \lambda \mu & \mu^2 & \nu^2 & \text{symmetric} \\
\lambda \mu & \mu^2 & \nu^2 & \text{symmetric} \\
\lambda \nu & \mu \nu & \nu^2 & \text{symmetric} \\
-\lambda^2 & -\lambda \mu & -\lambda \nu & -\mu^2 \\
-\lambda \mu & -\mu^2 & -\nu^2 & -\lambda \mu & \mu^2 \\
-\lambda \nu & -\mu \nu & -\nu^2 & -\lambda \nu & \mu \nu & \nu^2 \\
\end{bmatrix}
\] (5.15)

where:

- \(\lambda, \mu, \) and \(\nu\) are written for \(\lambda_x, \mu_x\) and \(\nu_x\) respectively.

For more detailed explanation of the linear analysis of plane and space truss structures, Ref [80] is recommended.
5.3.3 Calculation Steps

Based on the linear analysis described in the preceding sections two computer programs were developed. The first program performs linear analysis for plane truss structure and the second one performs the linear analysis for space truss structure. These analysis procedures have been coded in the MATLAB program. The programs developed are based on the work of Kwon and Bang [80] and listed in Appendices B and C.

The main steps involved in these programs are:

1) Input geometry and material properties data.
2) Calculate element matrices and load vectors for each element.
3) Develop global stiffness matrix and vector using the element matrices and load vectors.
4) Apply constraints to the global stiffness matrix.
5) Determine primary nodal variables by solving the matrix equation.
6) Compute secondary nodal variables.
7) Output required results.

The source code for linear analysis procedure for plane and space truss structures are presented in Appendices B and C, respectively.

The matrix analysis methods developed for linear truss structures can be extended to non-linear analysis. The next sections describe the analysis of trusses with non-linear behaviour.

5.4 Non-linear Analysis of Trusses

The causes of structural non-linearities may be broadly classified into two groups [81]:

- *material non-linearity*
Material non-linearity is due to non-linear stress-strain relationships of the materials that make up the structure. Both the displacements and the strains are assumed to be small in this case. Geometric non-linearity is ascribed to large deflection problems in which the equations of equilibrium of the structure must be established in the current configuration.

In the present work, only the geometrical non-linear effects are considered and material non-linearity is not taken into account in the truss analysis and design procedure.

5.4.1 Geometrical Non-linear Analysis

Geometrical non-linear analysis is required, as already noted, in long-span, cable structures and slender structures such as suspension bridges, where deflections are large enough to cause significant changes in the geometry of the structures. Although the materials that constitute a truss behave in a linearly elastic manner, the overall external load-deflection relationship of these members may become non-linear. Consequently, the stiffness of the structure should be calculated based on the deformed configuration, which complicates the analysis of such structures to some extent. For non-linear truss structures, the linear relationship \( \{F\} = [K]\{U\} \) of equation (5.1) can no longer be used, because the stiffness matrix \([K]\) is a function, in this case, of the joint displacements \( \{U\} \), which are as yet unknown. To account for the change in geometry a solution for the displacements \( \{U\} \) can be obtained by treating this non-linear problem in a sequence of linear steps, each step representing a local increment. However, because of large deflection, the strain-displacement equations contain non-linear terms, which must

- geometric non-linearity.
be included in calculating the stiffness matrix [83]. The non-linear effects in the strain-displacement equations modify the overall stiffness matrix:

\[ [K_T] = [K_E] + [K_G] \]  \hspace{1cm} (5.16)

where:

- \([K_T]\) is known as total tangent stiffness matrix of structure \((N \times N)\)
- \([K_E]\) is linear elastic global stiffness matrix \((N \times N)\)
- \([K_G]\) is the geometric stiffness matrix \((N \times N)\)
- \(N\) is the total number of degrees of freedom

It is quite inefficient to solve the set of non-linear stiffness equations, since a direct solution is not possible and an iterative method is required. In the current work the non-linear problems are linearised and iteratively solved by the Newton-Raphson method, which is discussed in section 5.4.4.

In the following sections geometrical non-linear analysis for both plane and space truss elements are described.

### 5.4.2 Plane Truss

Consider a truss element connecting joints \(a\) and \(b\) as shown in Fig. 5.3. Under the action of applied loading the element is displaced from its original location \(a\)-\(b\) to \(a'\)-\(b'\).

For the incremental displacements \(u\) and \(v\) the strain-displacement equation for a large deflection is given as:

\[ \varepsilon^a = \frac{du}{dx} + \frac{1}{2} \left( \frac{dv}{dx} \right)^2 \]  \hspace{1cm} (5.17)

where:

- \(\varepsilon^a\) is the strain increment [83].
The work done by an external force in causing deformation is stored within the member as strain energy. In an ideal elastic process, no dissipation of energy takes place and all the stored energy is recoverable upon unloading. The local elastic strain energy increment $U$ is given by:

$$U = \int_V \left( \int_0^a \sigma d\epsilon \right) dV$$

(5.18)

where:

- $V$ is the volume of the element a-b. Using Hook’s law $\sigma = E\epsilon$ and integrating over the length of the truss element $l$ yields:

$$U = \int_V \left( \int_0^a E \epsilon d\epsilon \right) dV = \frac{AE}{2} \int_0^l (\epsilon_a^g)^2 dx$$

(5.19)

where:
• $A$ and $E$ are the cross-section area and the elastic modulus of the bar material respectively.

Use the linear shape functions for the truss element

\[
u = \left(1 - \frac{x}{l}\right)u_a + \left(\frac{x}{l}\right)u_b
\]

(5.20)

\[
v = \left(1 - \frac{x}{l}\right)v_a + \left(\frac{x}{l}\right)v_b
\]

(5.21)

where:

• $u_a, u_b, v_a$ and $v_b$ are the incremental nodal displacement components as shown in Fig. 5.3.

After differentiating and substituting equations (5.20) and (5.21) into the strain energy expression 5.19 and neglecting the higher order term $\frac{d^4 v}{dx^4}$:

\[
U = \frac{AE}{2l} \left(u_a^2 - 2u_a u_b + u_b^2\right) + \frac{AE}{2l^2} \left(u_b - u_a\right) \times \left(v_b^2 - 2v_a v_b + v_a^2\right)
\]

(5.21)

Note that even for relatively large deflections the quantity $\frac{AE}{2} (u_b - u_a)$ may be treated as a constant equal to the axial tensile force in the element denoted by $P$. The strain energy $U$ is given by:

\[
U = \frac{AE}{2l} \left(u_a^2 - 2u_a u_b + u_b^2\right) + \frac{P}{2l} \left(v_b^2 - 2v_a v_b + v_a^2\right)
\]

(5.22)

Now differentiating with respect to $u_a, u_b, v_a, v_b$ and combining into matrix notation the element force-displacement relations are obtained:
\[ f_{ax} = \frac{du}{du_a} = \frac{AE}{l} (u_a - u_b) \quad , \quad f_{ay} = \frac{du}{dy_a} = \frac{P}{l} (v_a - v_b) \]

\[ f_{bx} = \frac{du}{du_b} = \frac{AE}{l} (u_a - u_b) \quad , \quad f_{by} = \frac{du}{dy_b} = \frac{P}{l} (-v_a + v_b) \]  \hspace{1cm} (5.23)

Collecting equations (5.23) into a single matrix equation, we obtain:

\[
\begin{pmatrix}
  f_{ax} \\
  f_{ay} \\
  f_{bx} \\
  f_{by}
\end{pmatrix}
= \begin{bmatrix}
  1 & 0 & -1 & 0 \\
  0 & 0 & 0 & 0 \\
  -1 & 0 & 1 & 0 \\
  0 & 0 & 0 & 0
\end{bmatrix}
\begin{pmatrix}
  u_a \\
  v_a \\
  u_b \\
  v_b
\end{pmatrix}
+ \begin{pmatrix}
  0 & 0 & 0 & 0 \\
  0 & 1 & 0 & -1 \\
  0 & 0 & 0 & 0 \\
  0 & -1 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
  u_a \\
  v_a \\
  u_b \\
  v_b
\end{pmatrix}
\]

\hspace{1cm} (5.24)

where:

- \( P \) is the intermediate axial force of the truss element at the current stage of loading.

Equation 5.24 can be written in compact form as:

\[
\{F\} = ([K_E] + [K_G]) \times \{U\}
\]

\[ = [K_T] \times \{U\} \]  \hspace{1cm} (5.25)

Thus, it can be seen from equation (5.25) that the total stiffness of the truss element consists of two parts, the elastic stiffness matrix \([K_E]\), which is the same as that used in linear analysis and \([K_G]\) which describes the changes in geometry of the element. While loading, initially no change in geometry is registered, \(K_G = 0\), rendering this stage similar to a linear analysis. As the loading is increased, the member experiences a change in geometry, which then requires non-linear analysis.

Transforming the stiffness matrices in equation (5.24) into the system axis by the appropriate transformation matrix yields:
\[ [K_e] = \frac{EA}{l} \begin{bmatrix} \lambda^2 & \lambda\mu & -\lambda^2 & -\lambda\mu \\ \lambda\mu & \mu^2 & -\lambda\mu & -\mu^2 \\ -\lambda^2 & -\lambda\mu & \lambda^2 & \lambda\mu \\ -\lambda\mu & -\mu^2 & \lambda\mu & \mu^2 \end{bmatrix} \] (5.26)

and

\[ [K_G] = \frac{P}{l} \begin{bmatrix} \mu^2 & -\lambda\mu & -\mu^2 & \lambda\mu \\ -\lambda\mu & \lambda^2 & \lambda\mu & -\lambda^2 \\ -\mu^2 & \lambda\mu & \mu^2 & -\lambda\mu \\ \lambda\mu & -\lambda^2 & -\lambda\mu & \lambda^2 \end{bmatrix} \] (5.27)

where:

- \( \lambda = \cos \phi \) and \( \mu = \sin \phi \), \((\phi)\) is the angle of rotation of the member).

### 5.4.3 Space Truss

Development of the geometric stiffness matrix \([K_G]\) for a space truss member is similar to that for a plane truss member. Consider the previous example of Fig. 5.2 in Section 5.3.2. The local or member axes are presented by \((x,y,z)\) and the global or system axes by \((\bar{x}, \bar{y}, \bar{z})\) respectively. The geometric stiffness matrix of this element can be represented as [81]:

\[
(K_G)_i = \frac{P_i}{l_i} \begin{bmatrix} (K_G)_i^B & - (K_G)_i^B \\ - (K_G)_i^B & (K_G)_i^B \end{bmatrix}
\] (5.28)

where:

- \( P_i \) is the force in member \( i \), \( l_i \) is its length and \((K_G)_i^B\) is the transformation matrix given by:
(K_G)^{\beta} = \begin{bmatrix}
1 - \lambda^2 & -\lambda\mu & -\lambda\nu \\
-\lambda\mu & 1 - \mu^2 & -\mu\nu \\
-\lambda\nu & -\mu\nu & 1 - \nu^2
\end{bmatrix} \tag{5.29}

where:

\lambda = \cos(\phi_{xx}), \phi_{xx} is angle between _x, x
\mu = \cos(\phi_{yx}), \phi_{yx} is angle between _y, x
\nu = \cos(\phi_{zx}), \phi_{zx} is angle between _z, x
\vdots
\nu = \cos(\phi_{zz}), \phi_{zz} is angle between _z, z

The global stiffness matrix [K_i] depends on P and \phi, which are both functions of nodal displacements. Thus the equilibrium equation \{F\} = [K_T] \times \{U\} in (5.25) represents a system of non-linear equations and an iterative solution scheme is required to solve these non-linear equations.

5.4.4 An Iterative Solution (the Newton-Raphson Method)

In the incremental load methods, the Newton-Raphson method have been employed on the basis of iterative procedures and applied to the above geometrically non-linear analysis problems. This method starts with a known equilibrium point and then uses iterative procedures to find the next equilibrium point at an increment of load. This procedure is repeated again until the limit point is reached. Figure 5.4 depicts the linearised nature of the Newton-Raphson solution and serves as a schematic representation of the pertinent algorithm variables defined in the following formulation [84].

The Newton-Raphson method begins at a known point i on the equilibrium point
as shown in Fig. 5.4 [84].

An increment load $\Delta F_i$, of the total external load is applied to the structure followed by the calculation of incremental displacements $\{\Delta u\}$ by making use of the system tangent stiffness matrix $[K_t]$, at point $i$ and solving for the global system of equation:

$$[K_t]_i \{\Delta u\} = \{\Delta F\}_i$$  \hspace{1cm} (5.30)

where:

- $\{\Delta F\}$ and $\{\Delta u\}$ are incremental values of external loads and joint displacements, respectively.
The increment of displacements is added to the previous displacement to form a new total displacement \( u_i + \Delta u \). The new displacement is not necessarily the correct value of the displacement of the structure at a load level \( F_{i+1} \). Therefore the joint equilibrium equations are not satisfied, and indicate that the internal nodal forces are not in equilibrium with the nodal external forces. This difference of displacements can be improved by applying a Newton-Raphson type of iteration.

The Newton-Raphson procedure is as follows:

1) The unbalanced nodal force is computed from:

\[
\Delta P_j = F_{i+1} - f(u)_j
\]  

(5.31)

where

- \( f(u)_j \) represent the internal joint forces corresponding to the \( j \)th iteration, and \( F_{i+1} \) is the load level at step \( i+1 \), which is kept constant during iterations.

2) The unbalanced joint forces are treated as the incremental values of external loads \( \Delta F \), and the adjustment vector, \( \Delta u_j \), is obtained from:

\[
[K_i]_j \Delta u_j = \Delta P_j
\]  

(5.32)

3) The new displacements are updated by:

\[
u_{i+1} = u_j + \Delta u_j
\]  

(5.33)

The process is repeated until convergence is achieved, that is until the load approaches the limit point \( i+1 \), so the deformed configuration of the structure is obtained corresponding to the load level \( F_{i+1} \).
5.4.5 Calculation Steps

Two computer programs are developed for geometrical non-linear analysis of plane and space truss structures. The analysis procedures are coded in the MATLAB program and the source code for each analysis procedure is presented in Appendices C and D, respectively. The basics of the computerised method, detailed theory and formal proofs can be found in [81]. These two non-linear analyses programs are iterative. In each iteration a new non-linear global stiffness matrix is updated and subsequently solved. The main steps involved in these programs are:

1) Read input data.

2) Allocate parameters / array sizes.

3) Apply the Newton-Raphson iteration method and compute the stiffness matrix.

4) Compute the unbalanced force.

5) Calculate the incremental displacements resulted from the unbalanced force by using equation (5.32).

6) Solve for change in displacement.

7) Repeat the above process until the unbalanced force approaches zero.

8) Compute the stresses and member displacements in the structural elements.

9) Output of results.

5.5 Buckling Constraint

A truss member can be subjected to either tensile or compressive loads. For a slender truss member, which is subject to compressive loads, stress limitations are often not enough to ensure the safety against the loss of stability. As in the optimisation process
of truss structures the cross-sectional area of members decrease, so does their stability. Thus constraints on member buckling are needed to avoid structural failure. In the case of dominant buckling constraints for the optimisation problems in this project, it was decided to use the American Institute of Steel Construction (AISC) formulas as explained in Ref. [42].

For compression members, one of the principal indicators of stability is the slenderness ratio, which can be shown as:

\[ s_i = \frac{Kl_i}{r_i}, \quad i = 1, 2, \ldots, n \]  (5.34)

where:

- \( K \) is the effective length factor (for truss structures is equal 1.0)
- \( l_i \) is the length of the \( i \)th member
- \( r_i \) is the appropriate radius of gyration for member \( i \).

Another non-dimensional ratio that is used is known as the column slenderness ratio:

\[ C = \sqrt{\frac{2\pi^2 E}{F_y}} \]  (5.35)

where:

- \( E \) is modulus of elasticity
- \( F_y \) is the yield stress of the materials

Once the slenderness ratio \( s_i \) and \( C \) have been computed, the allowable axial compression stress \( \sigma_{i-b} \) can be determined. Depending upon the compressive failure
mode, a different formula is used to define the allowable axial compression stress $\sigma_{i}^{b}$:

If $s_{i} < C$ then:

$$\sigma_{i}^{b} = \frac{F_{y}}{FS} \times \left[ 1 - \frac{s_{i}^{2}}{2C^{2}} \right]$$  \hspace{1cm} (5.36)

where:

- $FS$ is the factor of safety:

$$FS = \frac{5}{3} + \frac{3}{8} \left( \frac{s_{i}}{C} \right) - \frac{1}{8} \left( \frac{s_{i}^{3}}{C^{3}} \right)$$  \hspace{1cm} (5.37)

else, if $s_{i} \geq C$ then:

$$\sigma_{i}^{b} = \frac{12\pi^{2}E}{23s_{i}^{2}}$$  \hspace{1cm} (5.38)

The following sections present a detailed description of the GAs design formulation and procedure developed to solve the plane and space truss structures based on linear and non-linear analysis presented in the next chapter.

5.6 Design Problem Formulation

The discrete truss structural optimisation problem is to select optimal values of the design variables from a set of available values, such that the specified objective function is minimised and the necessary constraints are satisfied.

In the context of GAs, the discrete truss structural optimisation problem with $ne$ elements and discrete member size can be expressed as follows:

find the cross-sectional areas $A = \{A_{1}, A_{2}, ..., A_{ne}\}^{T}$ that minimise the total weight of the structure:
Minimise

\[ W(A) = \sum_{i=1}^{ne} A_i l_i \rho \]  \hspace{1cm} (5.39)

subject to:

\[ \frac{|\sigma_i|}{\sigma_i^a} - 1 \leq 0 \quad , \quad i = 1, 2, \ldots, ne \]  \hspace{1cm} (5.40)

\[ \frac{|d_j|}{d_j^a} - 1 \leq 0 \quad , \quad j = 1, 2, \ldots, r \]  \hspace{1cm} (5.41)

\[ \left\{ \begin{array}{c}
A_i^l \leq A_i \leq A_i^u \\
A_i \in S \quad , \quad i = 1, 2, \ldots, ne
\end{array} \right. \]  \hspace{1cm} (5.42)

where:

- Eq. (5.39) defines the weight of the truss structure. \( A_i \) and \( l_i \) are, respectively, the cross-section area and length of \( i \)th member and \( \rho \) is the weight density of material.

- Eq (5.40) defines the member stress constraints. \( \sigma_i \) and \( \sigma_i^a \) are respectively, the member stress and allowable stress in both tension and compression.

- Eq. (5.41) defines the displacement of the \( j \)th degree of freedom. \( d_j \) and \( d_j^a \) are respectively, the nodal displacement and the allowable displacement. \( r \) is the number of restricted displacements.

- In the Eq. (5.42), the values \( A_i^l \) and \( A_i^u \) are, respectively, the lower and upper bounds of the size variables, and \( S \) represents a list of discrete values to be assigned to size variables.

In the case of dominant buckling constraints in the design problem, the stress constraint is defined as:
where:

- \( \sigma_i^- \) and \( \sigma_i^+ \) are respectively, stresses in compression and tension in the \( i \)th member.
- \( \sigma_i^{\text{c}} \) and \( \sigma_i^{\text{t}} \) are the allowable stresses in compression and tension respectively.

According to the AISC specifications as explained in Ref. [42], the allowable tensile stresses \( \sigma_i^{\text{t}} \) is given as \( 0.6 \times F_y \), where \( F_y \) is the yield stress of steel and the allowable compression stress \( \sigma_i^{\text{c}} \) is defined in equation 5.36.

The above problem formulations represent a constrained optimisation problem. Since GAs require that the constrained optimisation problem be formulated as unconstrained one, therefore, some mechanisms of handling constraints are needed to transfer the constrained optimisation problem into an unconstrained problem, by using a penalty function method that is described in the following section.

### 5.7 Penalty Function Method

There are several methods of handling constraints in GAs. A review of the different ways GAs handle constraints can be found in [27]. The simple and most widely used method of handling constraints is a penalty function, where the infeasible solutions are penalised in proportion to the degree of violation of constraints [2]. Minimisation of the penalty function also minimises the constraint. The technique is very general and can be applied to both equality and inequality constraints. Although in this study, a penalty
function as suggested in [3] is used to penalise the weight computed using equation (5.39) to reflect violations of the problem constraints. The unconstrained optimisation problem is then written as:

\[
\text{minimise } U(A) = W(A) \times Pe
\]  

(5.44)

where:

- \( U(A) \) is the function to be minimised.
- \( W(A) \) is the truss weight as defined in Eq (5.39) and \( Pe \) is a penalty value, if no violation is found, then the penalty value is 1 that is no penalty is imposed on the objective function. If a constraint is violated, then the penalty value is defined as:

\[
Pe = (1 + vp \times (Cs + Cd))
\]  

(5.45)

where:

- \( vp \) is termed as a violation parameter, and is selected depending on the required influence of a violated individual in the next generation for which a value of 30 was found suitable for all the design examples presented in this study.
- \( Cs \) and \( Cd \) are stress and displacement violation coefficients respectively, which are defined as:

\[
Cs = \sum_{s=1}^{n} g_s , \quad Cd = \sum_{d=1}^{m} g_d
\]  

(5.46)

where:

- \((n+m)\) is the number of constraints in the problem. \( g_s \) and \( d_d \) represent
constraints that are expressed in normalised form and can be defined as:

\[
\begin{align*}
    g_s &= \left( \frac{\sigma_i}{\sigma_i^a} \right) - 1 \\
    g_d &= \left( \frac{d_j}{d_j^a} \right) - 1
\end{align*}
\]

if

\[
\begin{align*}
    \frac{\sigma_i}{\sigma_i^a} - 1 > 0 \\
    \frac{d_j}{d_j^a} - 1 > 0
\end{align*}
\]

where:

- \( \sigma_i \) and \( \sigma_i^a \) are respectively, the member stress and the allowable stress
- \( d_j \) and \( d_j^a \) are respectively, the nodal displacement and the allowable displacement.

In the case of dominant buckling constraints in the design problem, the constraint violation value of \( g_s \) is defined as:

\[
g_s = g^{-b} + g^{+b},
\]

\[
\begin{align*}
    g^{-b} &= \left( \frac{\sigma_i^{-}}{\sigma_i^{-b}} \right) - 1 \\
    g^{+b} &= \left( \frac{\sigma_i^{+}}{\sigma_i^{+b}} \right) - 1
\end{align*}
\]

if

\[
\begin{align*}
    \frac{\sigma_i^{-}}{\sigma_i^{-b}} - 1 > 0 \\
    \frac{\sigma_i^{+}}{\sigma_i^{+b}} - 1 > 0
\end{align*}
\]

where:

- \( g_s \) is the summation of \( g^{-b} \) and \( g^{+b} \) under buckling constraints where some members may have compressive force while others may have tensile force,
- \( \sigma_i^{-} \) and \( \sigma_i^{+} \) are respectively, stresses in compression and tension in the \( i \)th member,
- \( \sigma_i^{-b} \) and \( \sigma_i^{+b} \) are the allowable stresses in compression and tension respectively.

It can be easily seen from equation (5.44) if a constraint is violated, the design
solution is regarded as infeasible and a penalty parameter is multiplied to the weight of the structure $W(A)$, making it less desirable due to the increased weight.

5.8 Fitness Evaluation

As discussed in section (5.6), the truss weight computed using equation (5.39) is penalised to reflect violations of the problem constraints. This penalised weight is defined as unconstrained objective function or individual's fitness $U(A)$ and is expressed as the product of the weight and penalties as shown in equation (5.44). These fitness values are the usual starting point for implementation of a GA in an optimisation problem.

In the first operator in GAs, the reproduction operator, a mating pool is created by letting individuals with higher fitness values have more chance to be selected and allowing them to recombine, i.e., crossover and mutation. The reproduction operator may be implemented in algorithmic form in a number of ways. Perhaps the easiest way is the fitness-proportionate roulette-wheel selection, as discussed in Section (3.3.2). In order to obtain the minimal unconstrained objective or fitness $U(A)$, the algorithm requires a condition to perform the selection between the individuals. This is done in such a way, that the best individual has the highest fitness. An expression for the fitness becomes:

$$
\Phi_i = [U(A)_{\text{max}} + U(A)_{\text{min}}] - U_i(A)
$$

(5.49)

where:

- $\Phi_i$ is the fitness of the $i$th individual
- $U(A)_{\text{max}}$ and $U(A)_{\text{min}}$ are respectively the maximum and minimum values
of unconstrained objective function of equation (5.44).

After the evaluation of fitness value, the fitness factor (probability of selection) for the $i$th individual is calculated by:

$$P(I_i) = \frac{\Phi(I_i)}{\sum_{j=1}^{n} \Phi(I_j)}$$

(5.50)

where:

- $I_i$ represents the $i$th individual in the population and $n$ is the population size.

According to the fitness factor individuals get a number of copies in the mating pool. Highly fit individuals get more copies in the mating pool, whereas the less fit ones get fewer copies.

### 5.9 Profile List of Discrete Design Variables

Since the design variables i.e. cross-section properties of truss structural members are generally in a discrete standard form, the selection of standard cross-section areas of members becomes an important practice, during the optimisation process. Consequently, it is necessary to supply a list of values that the design variables can take. GA can be applied to any discrete set of sections produced according to the different standards.

In the case of discrete optimisation problems, the cross-sectional property is selected from existing members fabricated by Australia's BHP [21]. Altogether there are 70 standard circular hollow sections that are used in the optimum design of trusses. The details of dimensions and properties are listed in Appendix F.
5.10 Member Grouping

It is always desirable to specify groups of truss members, which are required to be of the same size. Such a design practice is usually implemented to standardization problem. The truss structures often comprise symmetric members of similar size, which can be easily grouped together in the same size group. For example, if \( ne \) is allocated the total number of groups, say 20, rather than total number of members, say 200, in equations (5.39)-(5.42), a considerable amount of computing time can be reduced as a result of reduced number of design variables.

The proposed system has the capability to intelligently decide on the grouping of structural members so as to achieve its goal in optimising the design process more effectively. In order to simplify the supply of members and to make the fabrication of the construction easier, it was decided to categorize the truss members into groups. Furthermore grouping of members has a large effect upon the optimum design problem in that it greatly reduces the number of design variables in the problem.

5.11 Structure of Proposed Design Procedure

The schematic diagram shown in Fig. 5.5 represents the operations of the GA-based procedure used for determining the discrete optimum design of truss structures based on linear and non-linear analysis. The proposed design algorithm consists of two main parts, where the first one is the analysis procedure of truss structure and the second one is the GA approach, which makes use of the results from analysis to determine the new cross section areas of truss members of trusses.
Start

Input structural data and GA parameters, such as the number of generations (ng)

Randomly generate chromosomal binary string of individuals

Decode binary values to integer values and select the cross-sectional areas from the proper catalogue for each design variable

Perform structural analysis using FEM and compute the weight of the structure

Check problem constraints and evaluate fitness function for each design set

Encode the integer values for each individual into binary strings.

Store the best individual and place into the next generation

If Generation = ng?

No

Yes

Print best values of variables and weight

End

Generation = Generation + 1

Mutation

Crossover

Selection

Fig. 5.5. Schematic Flow Chart of the Proposed Algorithm
In each optimisation cycle the analysis of truss is repeated and the new cross section areas are obtained. This process continues until minimum weight of truss is obtained, and the necessary limitations and constraints are satisfied.

The algorithm steps are outlined below. Detailed descriptions of the GA operations such as decoding, encoding, selection, crossover and mutation have been discussed in Chapter 3. These steps are summarised below:

**Step 1. Input:** The user-provided input includes:
A list of parameters for the linear or geometrically non-linear analysis including the total number of nodes and elements in the system, coordinate values of every node in terms of the global coordinate system, types of every elements and information of boundary conditions.
- A list of discrete standard cross-sectional areas.
- The number of lower and upper bounds.
- GA control parameters, such as crossover and mutation rate, number of individuals in the population (pop-size) and the maximum number of generation ($n_g$). Establishing these parameters is very crucial in an optimisation problem because there are no guidelines. One has to fix the GA parameters for a particular problem based on the convergence of the problem as well as solution time.

**Step 2. Initialisation:** GA generates an initial population (binary strings). The initial population is usually created randomly and presents a possible solution within the domain of the solution space.

**Step 3. Decoding:** The initial population is passed to the decoding process where the binary strings for each individual are decoded into decimal values and the sequence number in the available section list are obtained.
Step 4. **Analysis:** In the analysis step the linear and/or non-linear analysis procedures are performed and the responses of each individual are obtained.

Step 5. **Fitness evaluation:** The convergence criteria for each individual are checked. For each individual in the initial population the value of unconstrained function $U(x)$ is calculated using penalty function method from equation (5.44). The maximum and minimum values of this function are obtained and the fitness value for each individual is calculated using equation (5.49).

Step 6. **Encoding:** The initial population is passed to the encoding process where the decimal values for each individual are encoded into binary strings.

Step 7. **Elitist strategy:** The best individual is mutated before it is copied to the next generation using elitist strategy \([2] \) followed by the comparison of the fitness value of the mutated string with the original one. If the mutated string is better, it replaces the original one (and is copied to the next generation). Otherwise, the original string is copied to the next generation.

Step 8. **Selection:** The selection process determines which of the individuals will survive and continues on to the next generation. Using equation (5.50), the fitness factor for each individual in the population is calculated, and then through the selection module, individuals are selected based on their corresponding fitness to form a mating pool.

Step 9. **Crossover and mutation:** The individuals in the mating pool are altered through crossover (i.e. exchanging of portions of binary strings) and mutation operations (i.e. random changing of binary bits) resulting in a
new population of individuals.

**Step 10.** The new population replaces the initial population and stages 3–10 are repeated for the prescribed number of generations \((ng)\).

After the evolution of the initial population through many generations, the individuals within the final population will generally be much better as a whole than the individuals within the initial population. Also, the best individual in the final population will generally be optimal solution when the specified maximum number of generation \((ng)\) is reached.

### 5.12 MATLAB Program

The interactive software Matrix Laboratory (MATLAB) is a technical computing environment, which is used, in various high performance numeric computations \([80]\). MATLAB integrates numerical analysis with matrix computation and graphics. The advantages of MATLAB are that it provides many built-in auxiliary functions useful for function optimisation, it is easy to learn, written on an intuitive basis, and does not require in-depth computer programming knowledge.

The proposed algorithm (shown in Fig. 5.5 above) is written in the MATLAB programming environment. The program consists of one main program and several subprograms, which are stored in *m-files*. The main program uses other function subprograms to perform all the necessary GA operations (i.e. selection, crossover and mutation). The source listing for the main program and the sub-programs including initialisation, decoding, fitness evaluation, encoding, selection, crossover and mutation are presented in Appendix A and the analysis procedures for plane and space truss structures for both linear and non-linear behaviours are provides in Appendices B, C, D and E.
5.13 Concluding Remarks

As mentioned above, GAs have been extensively used in various truss structural optimisation problems with basic applications restricted to the linear behaviour of the structures only. However in the case of slender structures where the respective displacements significantly alter the original geometry, the accurate response of the structure under the applied loads can only be obtained by considering geometrical non-linear analysis.

This chapter provides a detailed depiction of the development of GA-based methodology for the sizing optimisation of plane and space truss structures with discrete design variables, where the effect of geometrical non-linearity is considered. This type of solution for truss optimisation problems has not been investigated before.

The application of the proposed methodology to solve plane and space truss structure problems as explained in this chapter is illustrated in the next chapter.
6.1 Introduction

This chapter presents the application of the genetic algorithm (GA) based methodology developed in this thesis for determining the optimum design of plane and space truss structures under a range of different loading conditions. The primary objectives are to demonstrate the effectiveness and ability of the proposed algorithm to obtain the optimal design of truss structures with discrete design variables based on geometrical non-linear analysis. To introduce the new GA-based optimisation approach, several truss structural problems are optimised including:

1) 10-bar cantilever truss
2) 20-bar bridge-type truss
3) 51-bar roof truss
4) 25-bar transmission tower space truss
5) 52-bar dome space truss
6) 56-bar transmission tower space truss
7) 120-bar dome space truss
Most of the problems are taken from research publications. For each of the problems, two or three different cases have been investigated and the optimal design problems for each of them is formulated assuming that:

1) the design variables are continuous and linear analysis is considered to obtain the response of the structure,
2) the design variables are continuous and the effect of geometrical non-linearity is considered,
3) the design variables are considered as discrete standard sections and linear analysis is considered to obtain the response of the structure,
4) the design variables are considered as discrete standard sections and the effect of geometrical non-linearity is considered.

In all the cases mentioned above trusses are optimised to achieve the minimum weight structure, while being subjected to member stress, nodal displacement and member buckling constraints. Design variables are cross sectional areas of each member of the truss structure. In the discrete optimisation of the truss structures where the design variables are selected on the basis of AS 1163 [21], the yield stress is taken as 350 MPa in accordance with AS 4100 cited in Ref. [21].

It should be noted that the proposed algorithm can be applied to many different classes of structures having a large number of members. These examples are specially selected to demonstrate the applicability of the method to different situations without necessarily emphasizing the size of the structure. In the following sections, the application of the proposed algorithm to optimise the design of each of the above problems is described and the results of optimisation are presented. In order to show that the solution obtained using proposed method is a likely optimal solution and to evaluate the effect of geometric non-linearity in the optimum design, the responses of
the final trusses such as nodal displacements and member stresses under the external loading were investigated and are presented in Appendix G.

6.2 Design of 10-Bar Cantilever Truss

The first problem to be considered is the 10-bar truss taken from Ref. [85] shown in Fig. 6.1 where it was treated as a continuous and discrete optimisation problem. The details of the dimensions and loading considered are presented in the Fig. 6.1. The initial, infeasible structure shown in Fig. 6.2 is obtained by assigning the lowest cross-sectional area from the catalogue of cross-section area to all members.

Fig. 6.1. Geometry and Loading of a 10-Bar Cantilever Truss

Fig. 6.2. Deformed Shape of 10-Bar Cantilever Truss
The total weight of the truss structure is considered as the objective function, which is solved using geometrical non-linear analysis under the constraints of member stresses and nodal displacements. The buckling constraint is not dominant in this design problem. The problem has 10 design variables. In the case of the continuous optimisation the lower and upper bound of cross-sectional areas are given as $65\,\text{mm}^2$ and $29032\,\text{mm}^2$ respectively. In the discrete optimisation process the design variables are selected from the following set of discrete values of two catalogues the first catalogue is: $(65, 323, 645, 1290, 2581, 4516, 7742, 12258, 17419, 23226)\,\text{(mm}^2\text{)}$, and the second catalogue is: $(65, 645, 1290, 3226, 5161, 7742, 9677, 11613, 12903, 16129, 19355, 22581, 29032)\,\text{(mm}^2\text{)}$. The design properties used are summarized in Table 6.1.

**Table 6.1. Definition of the 10-Bar Cantilever Truss**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>$E = 6.89 \times 10^4 \text{MPa}$</td>
</tr>
<tr>
<td>Material density</td>
<td>$\rho = 2770 \text{Kg/m}^3$</td>
</tr>
<tr>
<td>Allowable stress</td>
<td>$\sigma = 172.25 \text{MPa}$</td>
</tr>
<tr>
<td>Allowable displacement</td>
<td>$</td>
</tr>
</tbody>
</table>

The corresponding optimised truss was obtained after a run of 400 generations using a population size of 30. The convergence history of the minimum value of the objective function for both cases is shown in Fig. 6.3. The trajectories rationally report the fitness (i.e. the modified weight and constraint violations) for the two cases. As expected, the trajectories start with heavier designs and ultimately converge to lighter designs. However, the stagnation occurring at relatively early stages in the evolutionary process (within 0 to 50 generations) is noteworthy. This illustrates the ability of the
proposed algorithm to recognize and exploit the better “genetic” material contained within the population of truss.

*Case 1 is based on the solution from catalogue 1
**Case 2 is based on the solution from catalogue 2.

Fig. 6.3. Generation History for 10-Bar Cantilever Truss

The final results of the optimal design problem are listed in Table 6.2 and 6.3 with the comparison with results obtained by Gutkowski and Zawidzka [85]. The results of optimisation using the first catalogue are presented as case 1 whereas the results of optimisation using second catalogue is presented as case 2. Here comparing with the results of both continuous and discrete optimisation solutions shows that the present algorithm gives a better minimum weight.
### Table 6.2. Comparison of the Results for the 10-Bar Cantilever Truss (Case 1)*

<table>
<thead>
<tr>
<th>Design variables</th>
<th>Cross-section areas (mm²)</th>
<th>Gutkowski-Zawidzka [85]</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Continuous solution</td>
<td>Sequential algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Enumeration algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Continuous solution</td>
<td>Discrete solution</td>
</tr>
<tr>
<td>A1</td>
<td>19375</td>
<td>23226</td>
<td>23226</td>
</tr>
<tr>
<td>A2</td>
<td>65</td>
<td>326</td>
<td>323</td>
</tr>
<tr>
<td>A3</td>
<td>15015</td>
<td>17419</td>
<td>17419</td>
</tr>
<tr>
<td>A4</td>
<td>9862</td>
<td>12258</td>
<td>12258</td>
</tr>
<tr>
<td>A5</td>
<td>65</td>
<td>323</td>
<td>323</td>
</tr>
<tr>
<td>A6</td>
<td>365</td>
<td>323</td>
<td>1290</td>
</tr>
<tr>
<td>A7</td>
<td>4818</td>
<td>4516</td>
<td>4516</td>
</tr>
<tr>
<td>A8</td>
<td>13676</td>
<td>17419</td>
<td>12258</td>
</tr>
<tr>
<td>A9</td>
<td>13947</td>
<td>12258</td>
<td>12258</td>
</tr>
<tr>
<td>A10</td>
<td>66</td>
<td>323</td>
<td>65</td>
</tr>
<tr>
<td>Weight (kg)</td>
<td>2295.9</td>
<td>2484.7</td>
<td>2429.5</td>
</tr>
</tbody>
</table>

*Case 1 is based on the solution from catalogue 1.

### Table 6.3. Comparison of the Results for the 10-Bar Cantilever Truss (Case 2)**

<table>
<thead>
<tr>
<th>Design variables</th>
<th>Cross-section areas (mm²)</th>
<th>Gutkowski-Zawidzka [85]</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Continuous solution</td>
<td>Sequential algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Enumeration algorithm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Continuous solution</td>
<td>Discrete solution</td>
</tr>
<tr>
<td>A1</td>
<td>19375</td>
<td>19355</td>
<td>19355</td>
</tr>
<tr>
<td>A2</td>
<td>65</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>A3</td>
<td>15016</td>
<td>19355</td>
<td>16129</td>
</tr>
<tr>
<td>A4</td>
<td>9862</td>
<td>9677</td>
<td>7742</td>
</tr>
<tr>
<td>A5</td>
<td>65</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>A6</td>
<td>365</td>
<td>65</td>
<td>645</td>
</tr>
<tr>
<td>A7</td>
<td>4818</td>
<td>5161</td>
<td>5161</td>
</tr>
<tr>
<td>A8</td>
<td>13676</td>
<td>12903</td>
<td>12903</td>
</tr>
<tr>
<td>A9</td>
<td>13947</td>
<td>12903</td>
<td>16129</td>
</tr>
<tr>
<td>A10</td>
<td>65</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>Weight (kg)</td>
<td>2295.9</td>
<td>2340.4</td>
<td>2339.9</td>
</tr>
</tbody>
</table>

**Case 2 is based on the solution from catalogue 2.
The response of the final truss with discrete solutions in Case 1 and Case 2 are given in Table G.1, which indicate that the stresses developed in all the members are within the allowable strength of (172.25 MPa) and all the nodal displacements lie within the maximum allowable displacement (50 mm) in any direction.

6.3 Design of 20-Bar Bridge-Type Truss

The geometry and nodal coordinates of this example are shown in Fig. 6.4. The structure is subjected to two load conditions of 1000 kN each at nodes 4 and 6.

![Fig. 6.4 Geometry and Loading of 20-Bar Bridge-Type Truss](image)

The design properties are summarised in Table 6.4. The objective is to minimise the total weight of the structure by considering discrete design variables. This example is designed based on linear and geometric non-linear analysis.

The solution of the problem is obtained for three different cases as follows:

1) Based on member stresses and nodal displacements constraints considering non-linear behaviour.

2) Based on member stresses, nodal displacements and member buckling
3) Based on member stresses, nodal displacements and member buckling constraints considering linear behaviour.

<table>
<thead>
<tr>
<th>Table 6.4. Definition of the 20-Bar Bridge-Type Truss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
</tr>
<tr>
<td>Material density</td>
</tr>
<tr>
<td>Allowable stress</td>
</tr>
<tr>
<td>Yield stress</td>
</tr>
<tr>
<td>Allowable displacement</td>
</tr>
</tbody>
</table>

In the case of dominant buckling constraints, all member stresses are constrained to be below the buckling stress as indicated in equation (5.43). In the optimisation process the design variables are selected from the set of available sections in Table F.1. In total there are 76 different available sections and it is assumed that each design variable can take any one of the 76 values from the list of discrete available sections. In order to simplify the supply of members and the fabrication of the structures, it was decided to categorize the members into groups as discussed in Chapter 5. Thus the 20 members of the structure are divided into ten groups, which are required to have the same cross-section area.

In Case 1 three different runs using three different population sizes of 30, 40 and 60 are tried, and the generation history in each of these runs is shown in Fig. 6.5. In this case, the optimisation was carried out including member stress and nodal displacement constraints.
The grouping of members and values of cross section area for each member and the results of the optimal design problem are given in Table 6.5. It can be seen from Table 6.5 that different results are obtained for each run with different population size. However, the effect of population size is considerable in terms of obtaining the solution efficiently. It is also observed from the various runs that in the design optimisation based on geometrical non-linear analysis using small population size a better solution can be obtained as compared to large population size.

The final solution of a weight of 5435.1 kg has been obtained for a population size of 30. In the cases of dominant buckling constraints, the final design was obtained with a minimum weight of 6253 kg considering linear analysis and 6222 kg considering geometrical non-linear analysis.

Fig. 6.5 Generation History for 20-Bar Bridge-Type Truss
Table 6.5 Member Grouping Detail and Result of Optimisation for 20-Bar Truss

<table>
<thead>
<tr>
<th>Member grouping</th>
<th>Cross-section area (mm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Case 1)*</td>
</tr>
<tr>
<td>Group Member</td>
<td></td>
</tr>
<tr>
<td></td>
<td>30</td>
</tr>
<tr>
<td>A1 1, 4, 8, 13, 18</td>
<td>9380</td>
</tr>
<tr>
<td>A2 3, 9, 14, 19</td>
<td>989</td>
</tr>
<tr>
<td>A3 5, 10, 15</td>
<td>10300</td>
</tr>
<tr>
<td>A4 2, 20</td>
<td>8040</td>
</tr>
<tr>
<td>A5 6</td>
<td>6380</td>
</tr>
<tr>
<td>A6 7</td>
<td>4050</td>
</tr>
<tr>
<td>A7 11</td>
<td>2280</td>
</tr>
<tr>
<td>A8 12</td>
<td>2760</td>
</tr>
<tr>
<td>A9 16</td>
<td>325</td>
</tr>
<tr>
<td>A10 17</td>
<td>8230</td>
</tr>
<tr>
<td>Total weight (Kg)</td>
<td>5435.1</td>
</tr>
</tbody>
</table>

*Case 1: Solution with discrete design variables using three different population sizes of 30, 40 and 60, considering member stresses and nodal displacements constraints (non-linear behaviour).

**Case 2: Solution with discrete design variables considering member stresses, nodal displacements and member buckling constraints (non-linear behaviour).

***Case 3: Solution with discrete design variables considering member stresses, nodal displacements and member buckling constraints (linear behaviour).

It can be seen from Table 6.5 that the difference in the results of Case 2 and Case 3 is not very significant which is due the fact that this problem is inherently a linear one. The response of the final designs of linear and non-linear behaviour under external loading is given in the Tables G.2 and G.3.

6.4 Design of 51-Bar Roof Truss

Saka [69] demonstrated the solution of this problem by adopting double angle section for chords and single angle section for diagonals. It was treated as continuous optimisation problem based on linear analysis and it was noted that the buckling
constraints were dominant in the design problem.

The 51-members of the roof truss are divided into four design variables. The dimensions, details of the members grouping and two load cases considered in the design problem are shown in Fig. 6.6. The slope of the upper chord is taken as 5°. The design properties are summarised in Table 6.6.

![Fig. 6.6 Geometry and Loading of 51-Bar Roof Truss](image)

**Table 6.6. Definition of The 51-Bar Roof Truss**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>$E = 210 \times 10^3$ MPa</td>
</tr>
<tr>
<td>Yield stress in the case 1</td>
<td>$F_y = 275$ MPa</td>
</tr>
<tr>
<td>Yield stress in the cases 2 and 3</td>
<td>$F_y = 350$ MPa</td>
</tr>
<tr>
<td>Allowable displacement</td>
<td>$</td>
</tr>
</tbody>
</table>
The objective of this problem is to minimise the total volume of the structure by considering the following cases:

1) Continuous optimisation of the cross sectional areas of the members by considering stress, displacement and member buckling constraints.

2) Discrete optimisation of the cross sectional areas of the members considering member stress and displacement constraints.

3) Discrete optimisation of the cross sectional areas of the members considering stress, displacement and member buckling constraints.

In Case 1, the design variables are considered as continuous. In this case the yield stress is taken as 275 MPa, and the material is assumed to be double angle section for chords and single angle section for diagonals with the minimum size of cross-section area of 200 mm$^2$, and the radius of gyration ($R$) in terms of the design variable areas $R = 0.584 \times A^{0.524}$, where $A$ is the cross-section area [69]. In Case 2 and Case 3 the yield stress is taken as 350 MPa and the values of design variables are selected from the set of available circular hollow sections as illustrated in Table F.1. Member stress and nodal displacement are considered as constraints in Case 2 whereas in Case 3 buckling constraint is also considered in addition to the above mentioned constraints and the stress constraint in this case is defined using equation (5.43).

Results obtained in the case of continuous design variables are summarized in Table 6.7 and are compared with existing literature in Ref. [69], whereas results obtained by using proposed methodology in case of discrete design variables are presented in Table 6.8.
Table 6.7 Comparison of the Results for the 51-Bar Roof Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
<th>Saka [69]</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Continuous solution (Case 1)*</td>
<td>Continuous solution (Case 1)*</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>1673</td>
<td>1473</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>1149</td>
<td>1095</td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td>378</td>
<td>1550</td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td>721</td>
<td>665</td>
<td></td>
</tr>
<tr>
<td>Volume$\times 10^3$ (mm$^3$)</td>
<td>101537</td>
<td>95172</td>
<td></td>
</tr>
</tbody>
</table>

*Case 1: Solution with continuous design variables considering stress, displacement and buckling constraints (non-linear behaviour).

Table 6.8. Discrete Solution for the 51-Bar Roof Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Case 2)** (Case 3)***</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>1500</td>
<td>1230</td>
</tr>
<tr>
<td>A2</td>
<td>1240</td>
<td>1500</td>
</tr>
<tr>
<td>A3</td>
<td>523</td>
<td>1120</td>
</tr>
<tr>
<td>A4</td>
<td>332</td>
<td>533</td>
</tr>
<tr>
<td>Volume$\times 10^3$ (mm$^3$)</td>
<td>82098</td>
<td>92169</td>
</tr>
</tbody>
</table>

**Case 2: Solution with discrete design variables considering stress and displacement constraints (non-linear behaviour).

***Case 3: Solution with discrete design variables considering stress, displacement and buckling constraints (non-linear behaviour).

It can be seen from the results presented in Table 6.7 that the optimal design in Case 1 is obtained with a minimum volume of $95172 \times 10^3$ (mm$^3$), which is 6.3% lesser than $101737 \times 10^3$ (mm$^3$) obtained by Saka [69]. The difference in the results indicate
that the proposed methodology based on geometric non-linear analysis in this case yields a better result as compared to the optimality criteria methodology based on linear analysis used by Saka [69]. Table 6.8 indicates that the total volume obtained in Cases 2 and 3 using discrete design variables are lesser than the volume obtained in Case 1, which is due to the use of yield stress of 350 MPa instead of 275 MPa in Cases 2 and 3. It was found that there was an increase in volume from $82098 \times 10^3$ (mm$^3$) to $92169 \times 10^3$ (mm$^3$) by 10.9%. This relatively high change in minimum volume can be mainly attributed to the buckling constraint used in the Case 3.

The stresses in all members and displacements in all nodes of the final design in Case 3 are presented in Tables G.4 and G.5.

6.5 Design of 25-Bar Transmission Tower Space Truss

Consider the 25-bar transmission tower space truss taken from Rajeev and Krishnamoorthy [3] shown in Fig. 6.7. This problem was designed using genetic algorithm considering the linear behaviour only. The weight of the structure is taken as the objective function with constraints imposed on the member stresses and nodal displacements, without taking buckling constraints into account. The solution of this problem involves two cases. In the first case the design of the truss is based on linear analysis and in the second case the effect of geometrical non-linearity is considered.

The details of the loading are given in Table 6.9, member groupings are given in Table 6.10, node coordinates are given in Table 6.11 and the design parameters are summarised in Table 6.12.
Fig. 6.7 25-Bar Transmission Tower Space Truss

Table 6.9 Loading Details for the 25-Bar Space Truss

<table>
<thead>
<tr>
<th>Node</th>
<th>Fx (N)</th>
<th>Fy (N)</th>
<th>Fz (N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4,453.74</td>
<td>-4,453.74</td>
<td>-4,453.74</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>-4,453.74</td>
<td>-4,453.74</td>
</tr>
<tr>
<td>3</td>
<td>2,226.87</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>4</td>
<td>2,672.24</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 6.10 Group Membership for the 25-Bar Space Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Member</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>1-2</td>
</tr>
<tr>
<td>A2</td>
<td>1-4, 2-3, 1-5, 2-6</td>
</tr>
<tr>
<td>A3</td>
<td>2-5, 2-4, 1-3, 1-6</td>
</tr>
<tr>
<td>A4</td>
<td>3-6, 4-5</td>
</tr>
<tr>
<td>A5</td>
<td>3-4, 5-6</td>
</tr>
<tr>
<td>A6</td>
<td>3-10, 6-7, 4-9, 5-8</td>
</tr>
<tr>
<td>A7</td>
<td>3-8, 4-7, 6-9, 5-10</td>
</tr>
<tr>
<td>A8</td>
<td>3-7, 4-8, 5-9, 6-10</td>
</tr>
</tbody>
</table>
Table 6.11 Coordinates of the Joints of the 25-Bar Space Truss

<table>
<thead>
<tr>
<th>Node</th>
<th>X (mm)</th>
<th>Y (mm)</th>
<th>Z (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-952.5</td>
<td>0.0</td>
<td>5080.0</td>
</tr>
<tr>
<td>2</td>
<td>952.5</td>
<td>0.0</td>
<td>5080.0</td>
</tr>
<tr>
<td>3</td>
<td>-952.5</td>
<td>952.5</td>
<td>2540.0</td>
</tr>
<tr>
<td>4</td>
<td>952.5</td>
<td>952.5</td>
<td>2540.0</td>
</tr>
<tr>
<td>5</td>
<td>952.5</td>
<td>-952.5</td>
<td>2540.0</td>
</tr>
<tr>
<td>6</td>
<td>-952.5</td>
<td>-952.5</td>
<td>2540.0</td>
</tr>
<tr>
<td>7</td>
<td>-2540.0</td>
<td>2540.0</td>
<td>0.0</td>
</tr>
<tr>
<td>8</td>
<td>2540.0</td>
<td>2540.0</td>
<td>0.0</td>
</tr>
<tr>
<td>9</td>
<td>2540.0</td>
<td>-2540.0</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>-2540.0</td>
<td>-2540.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 6.12 Definition of The 25-Bar Space Truss

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>$E = 68.9 \times 10^3$ MPa</td>
</tr>
<tr>
<td>Material density</td>
<td>$\rho = 2770$ Kg/m$^3$</td>
</tr>
<tr>
<td>Allowable stress in the central member</td>
<td>$\sigma = 275.6$ MPa</td>
</tr>
<tr>
<td>Allowable displacement</td>
<td>$</td>
</tr>
</tbody>
</table>

It can be seen from Table 6.10 that the 25 members of the structure are divided into 8 groups, and the same cross section is assumed for each group. Hence, there are 8 design variables in this example. The cross-sectional area of each member as listed in Ref. [3] is taken from the following 30 discrete values: 64.5, 129.0, 193.5, 258.0, 322.5, 387.0, 451.5, 516.0, 580.5, 645.0, 709.5, 774.0, 838.5, 903.0, 967.5, 1032.0, 1096.5, 1161.5, 1225.5, 1290.4, 1354.5, 1419.4, 1483.5, 1548.3, 1612.5, 1677.4, 1806.4, 1935.4, 2064.5 and 2193.5 mm$^2$. A comparison of the final results obtained using the proposed method and the results obtained by Rajeev and Krishnamoorthy [3] are listed in Table 6.13.
Table 6.13 Comparison of the Results for the 25-Bar Space Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Discrete solution (Case 1)*</td>
</tr>
<tr>
<td>A1</td>
<td>64.5</td>
</tr>
<tr>
<td>A2</td>
<td>1161.5</td>
</tr>
<tr>
<td>A3</td>
<td>1483.5</td>
</tr>
<tr>
<td>A4</td>
<td>129.0</td>
</tr>
<tr>
<td>A5</td>
<td>64.5</td>
</tr>
<tr>
<td>A6</td>
<td>516.0</td>
</tr>
<tr>
<td>A7</td>
<td>1161.5</td>
</tr>
<tr>
<td>A8</td>
<td>1935.4</td>
</tr>
<tr>
<td>Weight (kg)</td>
<td>247.6</td>
</tr>
</tbody>
</table>

*Case 1: Discrete solution based on linear analysis.
**Case 2: Discrete solution based on geometrical non-linear analysis.

Table (6.13) indicates that better results are obtained by the proposed method. Once again as explained in example (6.3) the variation of the weight in Case 1 and Case 2 is not significant due to the linear nature of the truss structure. The responses of the final design of linear and non-linear behaviour under external loading are presented in Tables G.6 and G.7.

### 6.6 Design of 52-Bar Dome Space Truss

This example, taken from Ref. [70] was solved using the optimality criteria approach with continuous design variables and the effect of geometrical non-linearity considered. The truss structure was designed for the minimum weight with the cross-sectional areas of members being the design variables. The configuration, dimensions and the grouping of members of the 52-bar-space truss are shown in Fig. 6.8.
The space truss was designed under two load cases. For each of these load cases the solution to the problem was obtained by different formulations.

1) Vertical loads of 150kN are subjected in the negative direction of the Z-axis
at joints 6-13 causing compression in all truss members. In this load case the design problem was formulated as listed below.

a. Based on geometrical non-linear analysis with continuous design variables.
b. Based on linear analysis with discrete design variables.
c. Based on geometrical non-linear analysis with discrete design variables.
d. Based on geometrical non-linear analysis with discrete design variables. In addition to the limitation on stress and displacement, the member buckling constraints are also included in the design formulation of the problem in Case (d).

2) The second load case consists of the same loads applied at the same joints but in reverse direction causing tension in all the truss members. In this load case the problem was formulated in the same way as Case 1 except that the buckling constraint is not considered.

The structure members are divided into 8 different groups with the same cross section as shown in Fig. 6.8. The truss element and design properties are summarised in Table 6.14.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>$E = 210 \times 10^3$ MPa</td>
</tr>
<tr>
<td>Material density</td>
<td>$\rho = 7850$ Kg/m³</td>
</tr>
<tr>
<td>Yield stress in the case (a)</td>
<td>$F_y = 240$ MPa</td>
</tr>
<tr>
<td>Yield stress in the cases (b, c and d)</td>
<td>$F_y = 350$ MPa</td>
</tr>
<tr>
<td>Allowable displacement</td>
<td>$</td>
</tr>
</tbody>
</table>
In the case of continuous optimisation the yield stress used for the material was 240 MPa and the minimum size of cross-sectional area was taken as 200 mm$^2$ and in the discrete optimisation the yield stress was taken as 350 MPa and the design variables are selected from the set of available sections in Table F.1. The final results obtained for the continuous optimisation are given in Table 6.15 with the comparison with the results in the literature [70]. The results obtained by the proposed method using discrete design variables in both load cases are summarised in Table 6.16.

Table 6.15 Comparison of the Results for the 52-Bar Dome Space Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Saka [70]</td>
</tr>
<tr>
<td></td>
<td>Continuous solution</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>A1</td>
<td>8182</td>
</tr>
<tr>
<td>A2</td>
<td>2241</td>
</tr>
<tr>
<td>A3</td>
<td>3358</td>
</tr>
<tr>
<td>A4</td>
<td>1445</td>
</tr>
<tr>
<td>A5</td>
<td>1064</td>
</tr>
<tr>
<td>A6</td>
<td>2516</td>
</tr>
<tr>
<td>A7</td>
<td>200</td>
</tr>
<tr>
<td>A8</td>
<td>200</td>
</tr>
<tr>
<td>Weigh (kg)</td>
<td>5161</td>
</tr>
</tbody>
</table>

*Case 1 vertical load case in the negative direction of Z-axis:
   a) Solution with continuous design variables considering stress and displacement constraints (non-linear behaviour).

**Case 2 vertical load case in the positive direction of Z-axis:
   a) Solution with continuous design variables considering stress and displacement constraints (non-linear behaviour).
<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm²)</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Discrete solutions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Case1*</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(b) (c) (d) (b) (c)</td>
</tr>
<tr>
<td>A1</td>
<td>96.6</td>
<td>156</td>
</tr>
<tr>
<td>A2</td>
<td>733</td>
<td>332</td>
</tr>
<tr>
<td>A3</td>
<td>1440</td>
<td>1910</td>
</tr>
<tr>
<td>A4</td>
<td>1440</td>
<td>1780</td>
</tr>
<tr>
<td>A5</td>
<td>3600</td>
<td>3600</td>
</tr>
<tr>
<td>A6</td>
<td>4280</td>
<td>3540</td>
</tr>
<tr>
<td>A7</td>
<td>130</td>
<td>80.9</td>
</tr>
<tr>
<td>A8</td>
<td>130</td>
<td>80.9</td>
</tr>
<tr>
<td>Weigh (kg)</td>
<td>5121.8</td>
<td>4902.0</td>
</tr>
</tbody>
</table>

*Case 1: Vertical load case in the negative direction of Z-axis:

b. Solution with discrete design variables considering stress and displacement constraints (linear behaviour).

c. Solution with discrete design variables considering stress and displacement constraints (non-linear behaviour).

d. Solution with discrete design variables by considering stress, displacement and buckling constraints (linear behaviour).

**Case 2: Vertical load case in the positive direction of Z-axis:

b. Solution with discrete design variables considering stress and displacement constraints (linear behaviour).

c. Solution with discrete design variables considering stress and displacement constraints (non-linear behaviour).

It can be seen from Table 6.15, the final solution of the current method is better being 4767 kg in Case 1 (a) and 5074.6 kg in Case 2 (a) respectively when compared with the solution obtained by the improved optimality criteria approach of 5161 kg as in Ref. [70]. Also comparison of the results of discrete and continuous optimisation solution shows that the current algorithm gives a better minimum weight. The optimum design of the truss considering the linear behaviour with discrete design variables was obtained with the minimum weight of 5121.8 kg for the first load case. For the second
load case, the final design was also obtained with minimum weight of 4981.8 kg as shown in Table 6.16. A minimum weight of 4902 kg was obtained while considering geometrical non-linearity with discrete design variables in the first load case. This is only 4.3% lesser than the truss with linear behaviour. However, in the second load case, the optimum weight of non-linear behaviour was obtained with the minimum weight of 5037 kg, which is 1.1% heavier than the truss with linear behaviour and is not very significant. However, it is apparent that consideration of geometrical non-linearity makes optimum design to be based on realistic behaviour of the structure.

An interesting observation was made in Case 1(d) where the weight of the truss increased considerably, which can be largely attributed to the fact that the buckling constraints become active in the case of non-linear analysis. To evaluate the effect of geometric non-linearity in the optimum design, the response of the final designs with discrete design variables of linear and non-linear behaviour under external loading in both load cases are given in the Tables G.8, G.9, G.10 and G.11.

6.7 Design of 56-Bar Transmission Tower Space Truss

Consider the 56-bar transmission tower space truss taken from Saka [70] shown in Fig. 6.9. It was treated as a continuous optimisation problem and designed based on both linear and geometrical non-linear analysis. It was observed by Saka [70] that there can be a significant variation in the solution of linear and non-linear analysis to the extent that sometimes the direction of the member force is reversed when geometric non-linearity is taken into account indicating its importance in obtaining realistic solutions.

The 56-members of the space truss structure are divided into four groups, and the same cross section area is assumed for each group. Thus, there are four design variables
in this example. The details of the dimension and grouping of the members are presented in the Fig. 6.9. The loading conditions are given in Table 6.17 and the design properties are given in Table 6.18.

Fig. 6.9 56-Bar Transmission Tower Space Truss [70]
Table 6.17 Loading Details for the 56-Bar Transmission Tower Space Truss

<table>
<thead>
<tr>
<th>Node</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>13</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fx (kN)</td>
<td>45.5</td>
<td>45.5</td>
<td>0.0</td>
<td>0.0</td>
<td>45.5</td>
<td>45.5</td>
<td>45.5</td>
<td>45.5</td>
<td>45.5</td>
<td>45.5</td>
</tr>
<tr>
<td>Fy (kN)</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Fz (kN)</td>
<td>-91.0</td>
<td>-91.0</td>
<td>-91.0</td>
<td>-91.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 6.18. Definition of The 56-Bar Transmission Tower Space Truss

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>( E = 210 \times 10^3 \text{ MPa} )</td>
</tr>
<tr>
<td>Material density</td>
<td>( \rho = 7850 \text{ Kg/m}^3 )</td>
</tr>
<tr>
<td>Yield stress in case 1</td>
<td>( F_y = 240 \text{ MPa} )</td>
</tr>
<tr>
<td>Yield stress in cases 2 and 3</td>
<td>( F_y = 350 \text{ MPa} )</td>
</tr>
<tr>
<td>Allowable displacement in the x-direction</td>
<td>(</td>
</tr>
</tbody>
</table>

The space truss was designed by considering the following cases:

1) Continuous optimisation of the cross-sectional areas of the members based on non-linear analysis.

2) Discrete optimisation of the cross-sectional areas of the members based on linear analysis.

3) Discrete optimisation of the cross-sectional areas of the members based on non-linear analysis.

In the case of continuous optimisation the yield stress used for the material was 240 MPa and the minimum size of cross-sectional area was taken as 500 mm² and in the case of discrete optimisation the yield stress was taken as 350 MPa and the design variables are selected from the set of available sections in Table F.1. In this example buckling constraints are not considered.

The final results of the optimal design problem considering continuous design
variables and the results obtained by Saka [70] are compared and listed in Table 6.19. These results show that the minimum weight obtained by the GA-based approach is less than the minimum weight obtained by Saka [70] using the optimality criteria approach considering continuous design variables. The final results obtained by the proposed method using discrete design variables are summarised in Table 6.20.

Table 6.19. Comparison of the Results for the 56-Bar Transmission Tower Space Truss (Case 1)*

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Saka [70]</td>
<td>Proposed method</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Continuous solution</td>
<td>Continuous solution</td>
<td></td>
</tr>
<tr>
<td></td>
<td>non-linear behaviour</td>
<td>non-linear behaviour</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>1136</td>
<td>744</td>
<td>791.6</td>
</tr>
<tr>
<td>A2</td>
<td>10878</td>
<td>11102</td>
<td>10760.8</td>
</tr>
<tr>
<td>A3</td>
<td>927</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>A4</td>
<td>4562</td>
<td>4646</td>
<td>4958.4</td>
</tr>
<tr>
<td>Weight (kg)</td>
<td>13723.37</td>
<td>13577.16</td>
<td>13642</td>
</tr>
</tbody>
</table>

*Case 1: Solution with continuous design variables considering stress and displacement constraints (yield stress $F_y = 240$ MPa).

Table 6.20 Discrete Solution for the 56-Bar Transmission Tower Space Truss

<table>
<thead>
<tr>
<th>Group</th>
<th>Cross-section area (mm$^2$)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Discrete solution</td>
<td>(Case 2)**</td>
<td>(Case 3)***</td>
</tr>
<tr>
<td>A1</td>
<td>809</td>
<td>705</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>10400</td>
<td>11800</td>
<td></td>
</tr>
<tr>
<td>A3</td>
<td>80.9</td>
<td>80.9</td>
<td></td>
</tr>
<tr>
<td>A4</td>
<td>5360</td>
<td>4280</td>
<td></td>
</tr>
<tr>
<td>Weight (kg)</td>
<td>13590</td>
<td>13569</td>
<td></td>
</tr>
</tbody>
</table>

**Case 2: Solution with discrete design variables based on linear analysis considering stress and displacement constraints (yield stress $F_y = 350$ MPa).

***Case 3: Solution with discrete design variables based on non-linear analysis considering stress and displacement constraints (yield stress $F_y = 350$ MPa).
It is also observed that by including the geometric non-linear analysis in the design problem a better solution is obtained. The responses of the final designs with discrete design variables of linear and non-linear behaviour under external loading are presented in the Tables G.12 and G.13.

6.8 Design of 120-Bar Dome Space Truss

The last problem considered is the 120-bar dome space truss as shown in Fig. 6.10. This example was designed as a continuous optimisation problem considering both linear and non-linear behaviour [70]. The 120 members of the space truss structure are divided into seven different groups, and the same cross section is assumed for each group. The detailed dimensions and grouping of members are presented in the Fig. 6.10 and the design properties are given in Table 6.21. The space truss is subjected to a vertical load of 60 kN at joint 1, 30 kN at joints 2-14 and 10 kN at joints 15-37, acting in the negative direction of Z-axis.

The space truss was designed by considering the following cases:

1) Continuous optimisation of the cross-sectional areas of the members based on non-linear analysis considering stress and displacement constraints.

2) Discrete optimisation of the cross-sectional areas of the members based on linear analysis considering stress and displacement constraints.

3) Discrete optimisation of the cross-sectional areas of the members based on non-linear analysis considering stress and displacement constraints.

4) Discrete optimisation of the cross-sectional areas of the members based on non-linear analysis. In addition to the limitation on stress and displacement, the member buckling constraints are also included in the design formulation of the problem in the case 4.
In the case of continuous optimisation the yield stress used for the material was 240 MPa and the minimum size of cross-sectional area was taken as 200 mm². In the case of discrete optimisation the yield stress was taken as 350 MPa and the design variables are selected from the set of available sections in Table F.1.
Table 6.21. Definition of The 120-Bar Dome Space Truss

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of elasticity</td>
<td>$E = 210 \times 10^3$ MPa</td>
</tr>
<tr>
<td>Material density</td>
<td>$\rho = 7850$ Kg/m$^3$</td>
</tr>
<tr>
<td>Yield stress in the case 1</td>
<td>$F_y = 240$ MPa</td>
</tr>
<tr>
<td>Yield stress in the cases 2,3 and 4</td>
<td>$F_y = 350$ MPa</td>
</tr>
<tr>
<td>Allowable displacement in x-direction</td>
<td>$</td>
</tr>
</tbody>
</table>

Results for the continuous optimisation are summarised in Table 6.22, which are compared with the findings of [70], and the final results obtained for the discrete optimisation using the proposed method are presented in Table 6.23. It can be seen from the results presented in Table 6.22 that the continuous solution of the current algorithm is 7158.6 kg and the solution obtained by the optimality criteria approach [70] is 7587 kg. Comparison of the above results clearly indicates that the GAs-based approach gives a better weight.

Table 6.22. Comparison of the Results for the 120-Bar Dome Space Truss

<table>
<thead>
<tr>
<th>Proposed method Continuous solution (Case 1)*</th>
<th>Cross section area (mm$^2$)</th>
<th>Wight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A1$</td>
<td>$A2$</td>
</tr>
<tr>
<td>Saka [70] Continuous solution</td>
<td>1750</td>
<td>4484</td>
</tr>
<tr>
<td>(non-linear behaviour)</td>
<td>1666</td>
<td>4556</td>
</tr>
<tr>
<td>(linear behaviour)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Case 1: Solution with continuous design variables based on non-linear analysis considering stress and displacement constraints (yield stress $F_y = 240$ MPa).
Table 6.23. Discrete Solution for the 120-Bar Dome Space Truss

<table>
<thead>
<tr>
<th>Proposed method</th>
<th>Cross section area (mm$^2$)</th>
<th>Wight (kg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete solution</td>
<td>A1</td>
<td>A2</td>
</tr>
<tr>
<td>(Case 2)****</td>
<td>1500</td>
<td>4670</td>
</tr>
<tr>
<td>(Case 3)****</td>
<td>1230</td>
<td>5360</td>
</tr>
<tr>
<td>(Case 4)*****</td>
<td>1650</td>
<td>4420</td>
</tr>
</tbody>
</table>

**Case 2: Solution with discrete design variables based on linear analysis considering stress and displacement constraints (yield stress $F_y = 350$ MPa ).

***Case 3: Solution with discrete design variables based on non-linear analysis considering stress and displacement constraints (yield stress $F_y = 350$ MPa ).

****Case 4: Solution with discrete design variables based on non-linear analysis considering stress, displacement and member buckling constraints (yield stress $F_y = 350$ MPa ).

The total weight of the structure rises to 8750.7 kg from 7229 kg in Case 3. This relatively high change in minimum weight can be mainly attributed to the buckling constraint used in this case. The response of the structure under external loading in Case 2 and 3 is presented in Table G.1, G.2 and G.3.

6.9 Concluding Remarks

This chapter presented the design of seven different types of truss structures using the proposed GAs-based methodology. Optimisation problems were formulated as continuous and discrete design variables respectively and both linearities and geometrical non-linearities were considered in the design process. The results were compared with some design problems investigated previously and it was found that the proposed methodology delivered better results than the results obtained in all these cases.

The study shows that the GA is effective and reliable to solve both continuous and
discrete optimisation problems and it can be applied to any discrete set of sections produced according to the different standards. Furthermore, it is essential to consider geometrical non-linearities in the optimisation of truss structures to obtain a realistic design.

It was noticed from the numerical design problems solved that the non-linear analysis routine algorithm used most of the computation time and that computing time increases depending on the population size and the number of design variables. A large population size implies longer waiting times for convergence due to the fact that in each cycle of optimisation process the non-linear analysis requires 3-15 or more iterations.

In the next chapter the concluding remarks, general summary and the discussion is presented along with the future research direction remarks.
CHAPTER
SEVEN

DISCUSSION AND CONCLUSIONS

7.1 Introduction

In this chapter, the work presented in this thesis is discussed, and the main conclusions drawn from it are presented followed by suggestions on possible areas for future research.

7.2 Discussion

Designer's intuition and experience has played a major role in structural design process in the past instead of an intensive application of optimisation theory. This has recently changed because of advances in high speeds computer technology and new methods of incorporating highly complex and computational methods and the increased importance of optimal design of structures to reduce cost in manufacturing.

Much of the past research in optimisation of structures has dealt with steel truss structures. In the optimal design of truss structures there are important characteristics that must be considered. The first important characteristic is that, in structural design optimisation, the solution sought is the global optimal solution. Moreover, in practical
optimal design of truss structures, the design variables are discrete variables. This leads to a discrete optimisation problem, which is somewhat tedious to solve. Although there already exists a large number of design methodologies, GAs are one of the few optimisation tools available that are well suited to such discrete problem solving environments with an emphasis on search for the near global optimum point.

Virtually all of the optimisation methods developed in the area of optimal design of plane and space truss structures through GAs have dealt with linear behaviour [3-17]. The linear behaviour for some structures may not be valid because of the non-linear behaviour of the structure, which may be due to the geometry of the structure or the presence of geometric imperfections. The behaviour of these structures under external loading requires non-linear analysis and none of the previous work is based on non-linear analysis using GAs as the discrete optimisation tool.

Two computer codes based on the GA approach for the design of plane and space truss structures were developed. The first code was developed for the optimal design of the plane and space trusses based on linear analysis whereas the second code was devised for optimal design of the plane and space trusses based on geometrical non-linear analysis. The GA design methodology was presented in Chapter 3 and the formulation of both linear and geometrically non-linear analysis procedures were presented in Chapter 5. The source code for each analysis and GA procedures are presented in Appendices A, B, C, D and E respectively. The design problem formulation is presented in Section (5.6). In both design codes the constraints were applied on member stress, nodal displacement and/or member buckling. In order to solve the non-linear response of the truss structures an incremental load approach with a Newton-Raphson type of iteration was used for the geometrically non-linear analysis and for developing the optimisation algorithm, Section (5.4.4). In the design problems a
penalty function technique was introduced and implemented to handle constraints and evaluate the objective function, Section (5.7). The structure of proposed design algorithm is presented in Section (5.11). Typically, an elitist strategy is implemented where the best design from the population is copied into the next population to ensure that “good” designs found previously are not lost.

Several design problems have been used to measure the performance of the proposed algorithm. In all problems considered, trusses are designed for the minimum weight or volume with the cross-sectional area of truss members being the design variables. Constraints are imposed on member stresses, nodal displacements and member buckling.

The results reported in the previous chapter illustrate that the new design algorithm developed can deal effectively with the truss structural optimisation problems including continuous and discrete design variables. Furthermore, it has been shown that a realistic solution can be obtained by including geometrical non-linear analysis in the design of trusses. Most of the design results are compared with the best solution reported in the literature available. In all design problems, the results obtained by the proposed algorithm are better illustrating the advantages of the proposed GA-based methodology for the discrete and continuous structural optimisation problems.

The efficiency of a structural design optimisation is often measured by the GAs parameters such as crossover and mutation probability, population size and number of generations. However, because of the random nature of the search, the guidelines are extremely difficult to establish the GA parameters. Generally, it is recommended to have a high crossover and a low mutation probability [2]. Values of 0.8 and 0.002 were used for the crossover and mutation probability, respectively. The population size and the maximum number of generations were fixed after trying various values and studying
the convergence history of the algorithm. Generally, the bigger the population size, the more design features are included. However it is found from the numerical solutions that a population size as small as 30 produces an adequate results for truss problems based on geometrical non-linear analysis.

Figure 6.5 shows the convergence history of the minimum value of the objective function for the 20-bar plane truss using three independent runs with different starting populations. It is interesting that in all three cases very near optimal solutions are obtained after only 60 generations. This illustrates the ability and effectiveness of the proposed algorithm.

Most of the problems were solved for both continuous and discrete design variables. Comparing results shows that continuous design variable produces most effective solutions but it is important to emphasise that although results for both cases are mathematically feasible, continuous design variables are not preferred. This is due to the fact that commercial availability of member sections and the discrete nature of standards of cross sectional area properties for truss members. Discrete design variable although harder to implement, is less prone to errors, and is the preferred method in practice. In other words, practical feasibility of solution is best achieved by the proposed method of discrete structural optimisation as opposed to continuous structural optimisation.

As mentioned above, realistic solution can be obtained using non-linear analysis in the optimal design of truss structures. Unfortunately, because of the use of a range of different computing stations in implementing the optimisations throughout the current study, an accurate comparison of computing times for the different methods is not readily available. However, it was noticed from the numerical design problems solved that in the case of non-linear behaviour, computing time increase considerably. This is
due to the fact that each load increment requires three or more iterations to satisfy the equilibrium equations.

The results of the member stresses and nodal displacements under external loading as listed in Appendix G, indicate that the stresses developed in all the members are within the allowable strength and all the nodal displacements lie within the maximum allowable displacement in any direction for each truss example. These results are encouraging and suggest that the proposed algorithm can be used effectively and efficiently in other complex and realistic designs often encountered in engineering applications.

7.3 Conclusions

Based on the present work, the main conclusions can be summarised as follows:

The proposed genetic algorithms-based methodologies provide an ideal technique for optimal design of plane and space truss structures considering discrete as well as continuous design variables. The application of the proposed algorithms as the optimisation module makes this method free from gradient information, which is typical for the classic optimisation methods. It was found through several design problems, that the proposed algorithms are effective and reliable to solve both continuous and discrete optimisation problems and can be applied to any discrete set of sections produced according to the different standards. Furthermore, the use of objective or fitness function instead of using any gradient or other supplementary problem information makes the proposed algorithm to handle any design problems that may involve non-differentiable objective function and/or combination of continuous, discrete, and integer design parameters.

The use of geometrical non-linear analysis in the optimal design of plane and
space structures with slender members is of major importance because in slender structures deflections are large enough to cause significant change in the geometry of the structure. Consequently, the stiffness of the structure should be calculated based on deformed configuration.

In the optimisation of truss structures the cross-sectional area of members decrease, so does their stability. The role of buckling in the design of truss structures is of major importance to ensure the safety against the loss of stability. Thus constraints on member buckling are needed to avoid structural failure.

It has been proven that the proposed GAs-based methodologies are a possible powerful alternative to gradient-based techniques for optimisation problems based on both linear and geometrical non-linear analysis considering discrete design variables. The optimal solutions obtained using the proposed GAs-based methodologies are encouraging when compared with those achieved by the optimality criteria techniques. It was also observed that, a realistic solution can be achieved by including geometrical non-linear analysis in the design of truss structures.

Since GAs are initially developed to solve unconstrained optimisation problems therefore a penalty function method based on violation of normalized constraints was used to transfer the constrained optimisation problem into an unconstrained problem. GA parameters such as probabilities of crossover and mutation, population size and number of generations plays an important role in the value of optimum designs. The population size and the maximum number of generations were fixed after trying various values and studying the convergence history of the algorithm. Generally, the bigger the population size, the more design features are included. However, it was found from the numerical solutions that a population size as small as 30 produces favourable results for the design problems based on geometrical non-linear analysis.
It was also noticed from the numerical design problems solved that the most of the computation time was used by the non-linear analysis routine of algorithm. Computing time increase depending on the population size and the number of design variables used. Large population size implies longer waiting period for convergence because in each cycle of optimisation process, the non-linear analysis requires 3-15 or more iterations.

The implementation of design variable linking was particularly useful to reduce the number of independent design variables and consequently reduction in the computing time.

7.4 Suggestions for Future Work

The work carried out in this thesis has revealed many promising areas of further research in design optimisation field. A few of these areas worthy of further investigations are summarised as follows:

1. Structural optimisation using geometrically non-linear analysis is a time-consuming process. Further investigations are required to reduce time length of the process of the optimisation.

2. The concept of the proposed methodology should be extended to solve the truss structural optimisation problems formulated by shape and topology optimisation methods.

3. Although, in this research the design of both linear and geometrically non-linear plane and space truss structures are considered, the design methodology is general and further study is needed so that the design method can be easily extended to the design of geometrically non-linear structures other than truss structures such as frames.
4. In the present work the GA operators used were a roulette-wheel reproduction, a one-point crossover and a standard mutation only. Better results could be achieved if different types of GA operators are investigated revealing promising areas for further work.
References


Vol.11, No. 2, pp. 151-170.


Structural Optimisation, 2, pp.203-312.


material.” Springer, Berlin.


Appendix A

Source Code for the Proposed GA-Based Methodology

(Chapter 5)

% This script shows how to use the GA using a binary string representation.
global bounds st ds dn
%-------------------------------------
% GA parameters
%-------------------------------------
% Setting the seed back to the beginning for comparison sake
% Setting the seed back to the beginning for comparison sake
rand('seed',0);
xFns = 'Xover';
xOpt = []; % Crossover operators
mFns = 'Mutate';
mOpt = [] ; % Mutation operators
termFns = 'maxGenTerm';
check = 'Checking';
Pop_size=[ ];
termOps = [] ; % Maximum number of population
selectFn = 'roulette'
selectOps = [] ; % Selection function
evalFn = 'FEM_Eval';
evalOps = [] ; % Evaluation Function
bounds = [] ;

gaOpt=[1e-6 0 1];

% Generate an initialize population
%-------------------------------------
startPop = ini_pop(Pop_size,bounds,'FEM_Eval1,[],1 0);
disp('Initialize population and evaluated fitness ');
startPop

% Start Optimisation
%---------------------
[x,endPop,bestPop,trace,ca,S,D]=ga(bounds,evalFn,evalOps,startPop,gaOpt,
termFns,termOps,selectFns,selectOps,xFns,xOpt,mFns,mOpt);

% Result of optimisation
%-----------------------
disp('Element No., Area, Stresses, Displacements');
n=size(S,2);
nnum=1:1:n;
Result= ['numm', ' ca', ' S', ' D']
% Plot the best individual over time

% Plot the best over time
subplot(2,1,1); % Plot the best over time
plot(trace(:,1),abs(trace(:,2)),'b-*'); % Hit a return to continue
xlabel('Generation Number');
ylabel('Fitness (lbs)');
ylabel('Fitness (lbs)');
title('Best of penalized function','Color','r');
grid
subplot(2,1,2); % Add the average to the graph
plot(trace(:,1),abs(trace(:,3)),'g+-'); % Hit a return to continue
xlabel('Generation Number');
ylabel('Weight in(lbs)');
title('Best of objective function','Color','r');
grid
%end GA

% Subprogram: Initialisation

% Purpose: Generating Initial Population: ini_pop.m

function [pop] = ini_pop(num, bounds, evalFN, evalOps, options)

% pop - the initial, evaluated, random population
% populationSize - the size of the population, i.e. the number to create
% variableBounds - a matrix which contains the bounds of each variable, i.e. [var1_high var1_low; var2_high var2_low; ....]
% evalFN - the evaluation fn, usually the name of the .m file for evaluation
% evalOps - any options to be passed to the eval function
defaults []
% options - options to the initialize function, i.e. [type prec] where eps is the epsilon value and the second option is 1 for float and 0 for binary,
% prec is the precision of the variables
defaults [le-6 1]

if nargin<5
    options=[le-6 1];
end
if nargin<4
    evalOps=[];
end
if any(evalFN<48) %Not a .m file
    if options(2)==1 %Float GA
        estr=['x=pop(i,1); pop(i,xZomeLength)=' evalFN ' ;'];
    else %Binary GA

% A .m file
if options(2)==1 % Float GA
estr=['x=b2f(pop(i,:),bound,bits); pop(i,xZomeLength)=', evalFN ';
end
else % Binary GA
estr=['x=b2f(pop(i,:),bound,bits); [x v]= evalFN ...
'(x,[0 evalOps]); pop(i,:)=f2b(x,bound,bits) v];'
end
end
numVars = size(bound,1); % Number of variables
rng = (bound(:,2)-bound(:,1))'; % The variable ranges'
if options(2)==1 % Float GA
xZomeLength = numVars+1; % Length of string is numVar + fit
pop = zeros(num,xZomeLength); % Allocate the new population
pop(:,1:numVars) = (ones(num,1)*rng) .* (rand(num,numVars)) + ...
(ones(num,1)*bound(:,1))';
else % Binary GA
bits=calcbits(bound,options(1));
xZomeLength = sum(bits)+1; % Length of string is numVar + fit
pop = round(rand(num,sum(bits)+1));
end
for i=1:num
eval(estr);
end

% Binary Coded GA

function [x,endPop,bPop,traceInfo]=ga(bounds,evalFN,evalOps,startPop,opts,
termFN,termOps,selectFN,selectOps,
xOverFNs,xOverOps,mutFNs,mutOps)

Output Arguments:
x - the best solution found during the course of the run
endPop - the final population
bPop - a trace of the best population
traceInfo - a matrix of best of the ga for each generation
Input Arguments:
bounds - a matrix of upper and lower bounds on the variables
FEM Eval - the name of the evaluation .m function (Stress analysis)
evalOps - options to pass to the evaluation function ([NULL])
startPop - a matrix of solutions that can be initialized
from initialize.m
opts - [epsilon prob_ops display] change required to consider two solutions different, prob_ops 0 if you want to apply the genetic operators probabilistically to each solution, 1 if you are supplying a deterministic number of operator
for %quiet. ([1e-6 1 0])
% termFN - name of the .m termination function (['maxGenTerm'])
% termOps - options string to be passed to the termination function([100]).
% selectFN - name of the .m selection function (['normGeomSelect'])
% selectOpts - options string to be passed to select after
% select(pop,#,opts) ([0.08])
% xOverFNS - a string containing blank seperated names of Xover.m files (['Xover'])
% xOverOps - A matrix of options to pass to Xover.m files with the first column being the number of that xOver to perform
% mutFNS - a string containing blank seperated names of mutation.m files (['Mutation '])
% mutOps - A matrix of options to pass to Xover.m files with the first column being the number of that xOver to perform
0])

---

n=nargin;
if n<2 | n==6 | n==10 | n==12
    disp('Insufficient arguements')
end
if n<3 %Default evalation opts.
    evalOps=[];
end
if n<5
    opts = [1e-6 1 0];
end
if isempty(opts)
    opts = [1e-6 1 0];
end
if any(FEM_Eval<48) %Not using a .m file
    if opts(2)==1 %Float ga
        elstr=['x=c1; c1(xZomeLength)=' FEM_Eval ';'];
        e2str=['x=c2; c2(xZomeLength)=' FEM_Eval ';'];
    else %Binary ga
        elstr=['x=b2f(endPop(j,:),bounds,bits);
    endPop(j,:)=',...
    endPop(j,xZomeLength)=',...
    FEM_Eval ';'];
    end
else %Are using a .m file
    if opts(2)==1 %Float ga
        elstr=['[c1 c1(xZomeLength)]= FEM_Eval '(c1,[gen evalOps]);'];
        e2str=['[c2 c2(xZomeLength)]= FEM_Eval '(c2,[gen evalOps]);'];
    else %Binary ga
        elstr=['x=b2f(endPop(j,:),bounds,bits);[x v]= FEM_Eval ...
    'x,[gen evalOps]); endPop(j,:)={f2b(x,bounds,bits) v};'];
    end
end
if n<6 %Default termination information
termOps=[5];
termFN='maxGenTerm';
end
if n<12 %Default mutation information
if opts(2)==1 %Float GA
mutFNs=['Mutate'];
mutOps=[4 0 0;6 termOps(1) 3;4 termOps(1) 3;4 0 0];
else %Binary GA
mutFNs=['Mutate'];
mutOps=[0.05];
end
end
if n<10 %Default crossover information
if opts(2)==1 %Float GA
xOverFNs=['Xover';
xOverOps=[2 0;2 3;2 0];
else %Binary GA
xOverFNs=['Xover';
xOverOps=[0.6];
end
end
if n<9 %Default select opts only i.e. roulette wheel.
selectOps=[];
end
if n<8 %Default select info
selectFN=['normGeomSelect'];
selectOps=[0.08];
end
if n<6 %Default termination information
termOps=[5];
termFN='maxGenTerm';
end
if n<4 %No starting population passed given
startPop=[];
end
if isempty(startPop) %Generate a population at random
startPop=zeros(80,size(bounds,1)+1);
startPop=initializega(20,bounds,FEM_Eval,evalOps,opts(1:2));
end
if opts(2)==0 %binary
bits=calcbits(bounds,opts(1));
end
xOverFNs=parse(xOverFNs);
mutFNs=parse(mutFNs);
xZomeLength = size(startPop,2); %Length of the
xzome=numVars+fitness
numVar = xZomeLength-1; %Number of variables
popSize = size(startPop,1); %Number of individuals in the pop
endPop = zeros(popSize,xZomeLength); %A secondary population
matrix
c1 = zeros(1,xZomeLength); %An individual
c2 = zeros(1,xZomeLength); %An individual
numXovers = size(xOverFNs,1); %Number of Crossover operators
numMuts = size(mutFNs,1); %Number of Mutation operators
epsilonion = opts(1); % Threshold for two fitness to
differ
oval = max(startPop(:,xZomeLength)); %Best value in start pop
bFoundIn = 1; %Number of times best has changed
done = 0; %Done with simulated evolution
gen = 1; %Current Generation Number
collectTrace = (nargout>3); %Should we collect info every gen
floatGA = opts(2)==1; %Probabilistic application of ops
display = opts(3); %Display progress
% Increment generational
  [bval,bindx] = min(startPop(:,xZomeLength)); %Best of current pop
den=0;
% Update display and record current best individual
  gest(gen+1-1) = startPop(bindx,xZomeLength);
  plot((gest), 'b-*'); xlabel('(Generation)'); ylabel('(Fitness)');
  text(0.5,0.95,['Best = ', num2str(gest(gen+1-1))], 'Units', 'normalized');
  drawnow;

while(-done)
  % %Elitist Model
  %
  % [bval,bindx] = min(startPop(:,xZomeLength)); %Best of current pop
  best = startPop(bindx,:);
  x=b2f(best,bounds,bits);
  % % Analysis of Structure
  %
  [A,Q,W] = FEM_Eval(x,bounds);
  %
  if collectTrace
    traceInfo(gen,1)=gen; %current generatio
    traceInfo(gen,2)=startPop(bindx,xZomeLength); %Best fittness
    traceInfo(gen,3)=W; %total weight
    traceInfo(gen,4)=std(startPop(:,xZomeLength));
  end

  if ( (abs(bval - oval)>epsilon) | (gen==1) ) %If we have a new best sol
    if display
      fprintf(1,'\n%d \%f\n',gen,bval); %Update the display
    end
    if floatGA
      bPop(bFoundIn,:)=gen startPop(bindx,:)); %Update bPop Matrix
      else
      bPop(bFoundIn,:)=gen
      b2f(startPop(bindx,1:numVar),bounds,bits)
      startPop(bindx,xZomeLength)];
    end
    bFoundIn=bFoundIn+1; %Update number of changes
    oval=bval;
  else
    if display
      fprintf(1,'%d ',gen); %Otherwise just update
    num gen
  end
  end
% Penalty function
% Get the parameters of the population
numVars = size(startPop,2);
umSols = size(startPop,1);
Qx=startPop(:,numVars);
% calculate the fitness values
for r=1:numSols
  F=[max(Qx)+min(Qx)]-Qx(r);
  Fit(r)=F;
end
% Selection
endPop = feval(selectFN,startPop,[gen selectOps],Fit); %Select
if floatGA %Running with the model where the parameters are numbers of ops
  for i=1:numXOvers,
    for j=1:xOverOps(i,1),
      a = round(rand*(popSize-1)+1); %Pick a parent
      b = round(rand*(popSize-1)+1); %Pick another parent
      xN=deblank(xOverFNs(i,:)); %Get the name of crossover function
      [c1 c2] = feval(xN,endPop(a,:),endPop(b,:),bounds,[gen
      xOverOps(i,:)]);
      if c1(1:numVar)==endPop(a,(1:numVar)) %Make sure we created a new
        c1(xZomeLength)=endPop(a,xZomeLength); %solution before evaluating
      elseif c1(1:numVar)==endPop(b,(1:numVar))
        c1(xZomeLength)=endPop(b,xZomeLength);
      else
        [c1(xZomeLength) c1] = feval(FEM_Eval,c1,[gen evalOps]);
        eval(1str);
      end
      if c2(1:numVar)==endPop(a,(1:numVar))
        c2(xZomeLength)=endPop(a,xZomeLength);
      elseif c2(1:numVar)==endPop(b,(1:numVar))
        c2(xZomeLength)=endPop(b,xZomeLength);
      else
        [c2(xZomeLength) c2] = feval(FEM_Eval,c2,[gen evalOps]);
        eval(e2str);
      end
      endPop(a,:)=c1;
    endPop(b,:)=c2;
    end
  end
for i=1:numMuts,
  for j=1:mutOps(i,1),
    a = round(rand*(popSize-1)+1);
    c1 = feval(deblank(mutFNs(i,:)),endPop(a,:),bounds,[gen
    mutOps(i,:)]);
    if c1(1:numVar)==endPop(a,(1:numVar))
      c1(xZomeLength)=endPop(a,xZomeLength);
    else
      [c1(xZomeLength) c1] = feval(FEM_Eval,c1,[gen evalOps]);
      eval(1str);
    end
    endPop(a,:)=c1;
  end
else % We are running a probabilistic model of genetic operators
    % Crossover
    for i = 1:numXOvers,
        xN = deblank(xOverFNs(i,:)); % Get the name of crossover
        function
            cp = find(rand(popSize,1) < xOverOps(i,1) == 1);
            if rem(size(cp,1),2) cp = cp(1:(size(cp,1)-1));
        end
            cp = reshape(cp,size(cp,1)/2,2);
            for j = 1:size(cp,1)
                a = cp(j,1); b = cp(j,2);
                [endPop(a,:) endPop(b,:)] = feval(xN,endPop(a,:),endPop(b,:),...
                    bounds,[gen xOverOps(i,:)]);
            end
        end
        % Mutation
        for i = 1:numMuts
            mN = deblank(mutFns(i,:));
            for j = 1:popSize
                endPop(j,:) = feval(mN,endPop(j,:),bounds,[gen mutOps(i,:)]);
                eval(elstr);
            end
        end
    end
    gen = gen + 1;
    done = feval(termFN,[gen termOps],bPop,endPop); % See if the ga is done
    startPop = endPop; % Swap the populations

    [bval,bindx] = max(startPop(:,xZomeLength)); % Keep the best solution
    startPop(bindx,:) = best; % Replace it with the worst
    end % for While

    [bval,bindx] = min(startPop(:,xZomeLength));
    if display
        fprintf(1,'\n%d %f
',gen,bval);
    end

    x = startPop(bindx,:);
    if opts(2) == 0 % binary
        x = b2f(x,bounds,bits);
        bPop(bFoundIn,:) = [gen b2f(startPop(bindx,1:genVar),bounds,bits)...
            startPop(bindx,xZomeLength)];
    else
        bPop(bFoundIn,:) = [gen startPop(bindx,:)];
    end

    if collectTrace
        traceInfo(gen,1) = gen; % Current generation
        traceInfo(gen,2) = startPop(bindx,xZomeLength); % Best fitness
        traceInfo(gen,3) = mean(startPop(:,xZomeLength)); % Avg fitness
    end

    % End of GA
% Subprogram: Decode
% Purpose: Translate the binary representation of design variable
% into corresponding float number

function [fval] = b2f(bval,bounds,bits)

% fval - the float representation of the number
% bval - the binary representation of the number
% bounds - the bounds on the variables
% bits - the number of bits to represent each variable

% The range of the variables
scale=(bounds(:,2)-bounds(:,1))'./(2.^bits-1);
numV=size(bounds,1);
cs=[0 cumsum(bits)];
for i=1:numV
    a=bval((cs(i)+1):cs(i+1));
    X(i)=sum(2.^(size(a,2)-1:-1:0).*a)*scale(i)+bounds(i,1);
end

% Subprogram: Evaluation
% Purpose: Check Problem Constraints and evaluate penalty function

function [Q]=Check(W,S,D,bounds)

nums = size(S',1); % Get the parameters of the stress and displacement
numd = size(D,1);
Dep=D(:,2);
for t=1:nums % Violation coefficient Cj (stresses)
    if abs((S(t))/st)-1<=0;
        vj=0;
    else
        vj= abs((S(t))/st)-1;
    end
    gj(t)=vj;
end
Cj=sum(gj);
for i=1:numd % Violation coefficient Ck (displacement)
    if abs((Dep(i))/ds)-1<=0;
        vk=0;
    else
        vk= abs((Dep(i))/ds)-1;
    end
    gk(i)=vk;
end
Cd=sum(gk);
% Unconstrained Objective function
U(x)=W*(1+(K*(Cj+Cd)));
% Subprogram: Encode
% Purpose: Return the Binary Representation of the Float Number

function [bval] = function(fval,bounds,bits)

% fval - the float representation of the number
% bval - the binary representation of the number
% bounds - the bounds on the variables
% bits - the number of bits to represent each variable

scale=(2.^bits-1)./(bounds(:,2)-bounds(:,1))'; %The range of the variables
numV=size(bounds,1);
cs=[0 cumsum(bits)];
bval=[];
for i=1:numV
    X(i)=(X(i)-bounds(i,1)) * scale(i);
bval=[bval rem(floor(X(i)*pow2(1-bits(i):0)),2)];
end

% Subprogram: Selection
% Purpose: Select Individual into Mating Pool Based on Their Fitness Values

function [newPop] = selection(oldPop,options,Fit)

% roulette-wheel selection
%newPop - the new population selected from the oldPop
%oldPop - the current population
%options - options [gen]
%f - Fitness of rth individual

numVars = size(oldPop,2); %Get the parameters of the population
numSols = size(oldPop,1);
Qx=oldPop(:,numVars);
totalFit = sum(Fit); %Generate the relative probabilities of selection
prob=Fit / totalFit;
prob=cumsum(prob);

rNums=sort(rand(numSols,1)); %Generate random numbers
fitIn=1;newIn=1; %Select individuals from the oldPop to the new
while newIn<=numSols
    if(rNums(newIn)<prob(fitIn))
        newPop(newIn,:) = oldPop(fitIn,:);
        newIn = newIn+1;
    else
        fitIn = fitIn + 1;
    end
end
function [c1,c2] = Xover(p1,p2,bounds,Ops)

---

numVar = size(p1,2)-1;  % Get the number of variables
% Pick a cut point randomly from 1-number of vars

cPoint = round(rand * (numVar-2)) + 1;

cl = [p1(1:cPoint) p2(cPoint+1:numVar+1)];  % Create the children

c2 = [p2(1:cPoint) p1(cPoint+1:numVar+1)];

function [parent] = Mutate(parent,bounds,Ops)

% Binary mutation
% function [newSol] = binaryMutate(parent,bounds,Ops)
% parent - the first parent ( [solution string function value] )
% bounds - the bounds matrix for the solution space
% Ops - Options for binaryMutation [gen prob_of_mutation]

pm=Ops(2);

% Get the number of variables
numVar = size(parent,2)-1;

% Pick a variable to mutate randomly from 1-number of vars
rN=random(1,numVar)<pm;

parent=[abs(parent(1:numVar) - rN) parent(numVar+1)];
Appendix B

Source Code for the Linear Analysis of Plane Truss Structure
(Chapter 5)

% Linear Analysis
% Purpose: Linear Analysis of Plane Truss Structure
%

function [ST,le,DS]=FEM(A,bounds)

% Variable descriptions
% k = element stiffness matrix
% kk = system stiffness matrix
% ff = system force vector
% index = a vector containing system dofs associated with each
% element
% gcoord = global coordinate matrix
% disp = nodal displacement vector
% elforce = element force vector
% eldisp = element nodal displacement
% stress = stress vector for every element
% elprop = element property matrix
% nodes = nodal connectivity matrix for each element
% bc dof = a vector containing dofs associated with boundary
% conditions
% bcval = a vector containing boundary condition values
% associated with the dofs in 'bc dof'
%
% Control Input Data

nel=..; % number of elements
nnel=..; % number of nodes per element
ndof=..; % number of dofs per node
nnode=..; % total number of nodes in system
sdof=nnode*ndof; % total system dofs
%
% Nodal coordinates
%
% x, y-coordinate of node 1
gcoord(1,1)=..; gcoord(1,2)=..;
gcoord(2,1)=..; gcoord(2,2)=..;
...
gcoord(nnode,1)=..; gcoord(nnode,2)=..
%
% Material and geometric properties
%
prop(1)=..; % elastic modulus
prop(2)=..; % cross-sectional area
%
% Nodal connectivity
%
nodes(1,1)=..; nodes(1,2)=..
nodes(2,1)=..; nodes(2,2)=..
...
nodes(nel,1)=..; nodes(nel,2)=..;

% Applied constraints
%
bcdof( 1 )=..; % dof (horizontal displ) is constrained
bcval( 1 )=..; % whose described value is 0
bcdof( 2 )=..; % dof (vertical displ) is constrained
bcval( 2 )=..; % whose described value is 0
...
bcval(sdof)=..; % dof (horizontal displ) is constrained
bcval(sdof)=..; % whose described value is 0

% Initialization to zero
%
ff=zeros(sdof,1); % system force vector
kk=zeros(sdof,sdof); % system stiffness matrix
index=zeros(nnel*ndof,1); % index vector
elforce=zeros(nnel*ndof,1); % element force vector
eldisp=zeros(nnel*ndof,1); % element nodal displacement vector
k=zeros(nnel*ndof,nnel*ndof); % element stiffness matrix
stress=zeros(nel,1); % stress vector for every element

% Applied nodal force
%
ff( 1 )=..;
ff( 2 )=..;
ff(nnode)=..;
%
% Loop for elements
%
for iel=1:nel % loop for the total number of elements
nd(l)=nodes(iel,1); % 1st connected node for the (iel)-th element
nd(2)=nodes(iel,2); % 2nd connected node for the (iel)-th element
x1=gcoord(nd(1),1); y1=gcoord(nd(1),2); % coordinate of 1st node
x2=gcoord(nd(2),1); y2=gcoord(nd(2),2); % coordinate of 2nd node
leng=sqrt((x2-x1)^2+(y2-y1)^2); % element length
if (x2-x1)==0;
beta=2*atan(1); % angle between local and global axes
else
beta=atan((y2-y1)/(x2-x1));
end
el=prop(l); % extract elastic modulus
area=prop(2); % extract cross-sectional area
index=feeldof(nd,nnel,ndof); % extract system dofs for the element
k=fetruss2(el,leng,area,0,beta,1); % compute element matrix
kk=feasmbll(kk,k,index); % assemble into system matrix
end
%
% Apply constraints and solve the matrix
%
[kk,ff]=feaplyc2(kk,ff,bcdof,bcval); % apply the boundary conditions
solve the matrix equation to find nodal displacements

% Post computation for stress calculation

for iel=1:nel
    % loop for the total number of elements
    nd(1)=nodes(iel,1); % 1st connected node for the (iel)-th element
    nd(2)=nodes(iel,2); % 2nd connected node for the (iel)-th element
    x1=gcoord(nd(1),1); y1=gcoord(nd(1),2); % coordinate of 1st node
    x2=gcoord(nd(2),1); y2=gcoord(nd(2),2); % coordinate of 2nd node
    leng=sqrt((x2-x1)^2+(y2-y1)^2); % element length
    if (x2-x1)==0;
        beta=2*atan(1);
    else
        beta=atan((y2-y1)/(x2-x1));
    end
    el=prop(iel); % extract elastic modulus
    area=prop(2); % extract cross-sectional area
    index=feeldof(nd,nnel,ndof); % extract system dofs for the element
    k=fetruss2(el,leng,area,0,beta,1); % compute element matrix
    for i=1:(nnel*ndof) % extract displacements associated with
        eldisp(i)=disp{index(i)}; % (iel)-th element
    end
    elforce=k*eldisp; % element force vector
    % Stress calculation
    stress(iel)=sqrt(elforce(1)^2+elforce(2)^2)/area;
    if ((x2-x1)*elforce(3)) < 0;
        stress(iel)=-stress(iel);
    end
end
% Print fem solutions
num=1:1:sdof;
disp=[num' disp] % print displacements
numm=1:1:nel;
stresses=[numm' stress] % print stresses

function [index]=feeldof(nd,nnel,ndof)

% Synopsis:
% [index]=feeldof(nd,nnel,ndof)
% Variable Description:
index - system dof vector associated with element
"iel"
iel - element number whose system dofs are to be
determined
nnel - number of nodes per element
ndof - number of dofs per node

edof = nnel*ndof;
k=0;
for i=1:nnel
    start = (nd(i)-1)*ndof;
    for j =1:ndof
        k=k+1;
        index(k)=start+j;
    end
end

% Subprogram: fetruss2
% Purpose: Compute Stiffness and Mass Matrices for the Plane Truss Element. Nodal dof \{u_1 v_1 u_2 v_2\}

function [k,m]=fetruss2(el,leng,area,rho,beta,ipt)

% Synopsis:
% [k,m]=fetruss2(el,leng,area,rho,beta,ipt)
% Variable Description:
% k - element stiffness matrix (size of 4x4)
% m - element mass matrix (size of 4x4)
% el - elastic modulus
% leng - element length
% area - area of truss cross-section
% rho - mass density (mass per unit volume)
% beta - angle between the local and global axes
% ipt = 1: consistent mass matrix
% positive if the local axis is in the ccw direction
% from
% the global axis
% ipt = 1 - consistent mass matrix
% = 2 - lumped mass matrix

%c=cos(beta); s=sin(beta);
k= (area*el/leng)*[ c*c c*s -c*c -c*s;...
c*s s*s -c*s -s*s;...
-c*c -c*s c*c c*s;...
-c*s -s*s c*s s*s];

% consistent mass matrix
if ipt==1
    m=(rho*area*leng/6)*[ 2*c*c+2*s*s 0 c*c+s*s 0;...
                          0 2*c*c+2*s*s 0 c*c+s*s;...
                          c*c+s*s 0 2*c*c+2*s*s 0;...
                          0 c*c+s*s 0 2*c*c+2*s*s];
else
    m=(rho*area*leng/2)*[ c*c+s*s 0 0 0;...
function [kk]=feasmb1l(kk,k,index)

% Synopsis:
% [kk]=feasmb1l(kk,k,index)
% Variable Description:
% kk - system matrix
% k - element matrix
% index - d.o.f. vector associated with an element

edof = length(index);
for i=1:edof
  ii=index(i);
  for j=1:edof
    jj=index(j);
    kk(ii,jj)=kk(ii,jj)+k(i,j);
  end
end

% Subprogram: feaplyc2
% Purpose: Apply Costraints to Matrix Equation.

function [kk,ff]=feaplyc2(kk,ff,bcdof,bcval)

% Synopsis:
% [kk,ff]=feaplybc(kk,ff,bcdof,bcval)
% Variable Description:
% kk - system matrix before applying constraints
% ff - system vector before applying constraints
% bcdof - a vector containing constrained d.o.f
% bcval - a vector containing contained value

n=length(bcdof);
sdof=size(kk);
for i=1:n
  c=bcdof(i);
  for j=1:sdof
    kk(c,j)=0;
  end
  kk(c,c)=1;
  ff(c)=bcval(i);
end
Appendix C

Source Code for the Linear Analysis of Space Truss Structure
(Chapter 5)

% Linear Analysis
% Purpose: Linear Analysis of Space Truss Structure
% function [ST, le, DS] = FEM(A, bounds)
%
% Control Input Data
% nel = ..; % number of elements
% nnel = ..; % number of nodes per element
% ndof = ..; % number of dofs per node
% nnode = ..; % total number of nodes in system
% sdof = nnode * ndof; % total system dofs
%
% Nodal coordinates
% % x, y, z-coordinate of node 1
% gcoord(1,1) = ..; gcoord(1,2) = ..; gcoord(1,3) = ..;
% gcoord(2,1) = ..; gcoord(2,2) = ..; gcoord(2,3) = ..;
% ... .. ...
% gcoord(nnode,1) = ..; gcoord(nnode,2) = ..; gcoord(nnode,3) = ..;
%
% Material and geometric properties
% prop(1) = ..; % elastic modulus
% prop(2) = ..; % cross-sectional area
%
% Nodal connectivity
% nodes(1,1) = ..; nodes(1,2) = ..;
% nodes(2,1) = ..; nodes(2,2) = ..;
% ... ...
% nodes(nel,1) = ..; nodes(nel,2) = ..;
%
% Applied constraints
% bcdof(1) = ..; % dof (horizontal displ) is constrained
% bcval(1) = ..; % whose described value is 0
% bcdof(2) = ..; % dof (vertical displ) is constrained
% bcval(2) = ..; % whose described value is 0
% ...
% bcdof(sdof) = ..; % dof (horizontal displ) is constrained
% bcval(sdof) = ..; % whose described value is 0
%
% Initialization to zero
% ff = zeros(sdof,1); % system force vector
% kk = zeros(sdof, sdof); % system stiffness matrix
% index = zeros(nnel * ndof, 1); % index vector
elforce=zeros(nnel*ndof,1);  % element force vector
eldisp=zeros(nnel*ndof,1);  % element nodal displacement vector
k=zeros(nnel*ndof,nnel*ndof); % element stiffness matrix
stress=zeros(nel,1);       % stress vector for every element

% Applied nodal force
% ff(  1  )=..;
ff(  2  )=..;
ff(nnode)=..;
%
% Loop for elements
% for iel=1:nel  % loop for the total number of elements
nd(1)=nodes(iel,1); % 1st connected node for the (iel)-th element
nd(2)=nodes(iel,2); % 2nd connected node for the (iel)-th element
%
% coordinate of 1st node
x1=gcoord(nd(1),1); y1=gcoord(nd(1),2); z1=gcoord(nd(1),3);%
% coordinate of 2nd node
x2=gcoord(nd(2),1); y2=gcoord(nd(2),2); z2=gcoord(nd(2),3);
% element length
leng=sqrt((x2-xl)^2+(y2-yl)^2+(z2-zl)^2); % element length
% node coordinates
betx=(x2-xl)/leng;
bety=(y2-yl)/leng;
betz=(z2-zl)/leng;
ell=elprop(iel,1); % extract elastic modulus
area=elprop(iel,2); % extract cross-sectional area
index=feeldof(nd,nnel,ndof); % extract system dofs for the element

k=fetruss2(el,leng,area,0,betx,1,bety,betz); % compute element matrix
kk=feasmbll(kk,k,index); % assemble into system matrix
%
% Apply constraints and solve the matrix
% [kk,ff]=feaplyc2(kk,ff,bcdof,bcval); % apply the boundary conditions
disp=kkf; % solve the matrix equation to find nodal displacements
%
% Post computation for stress calculation
% for iel=1:nel  % loop for the total number of elements
nd(1)=nodes(iel,1); % 1st connected node for the (iel)-th element
nd(2)=nodes(iel,2); % 2nd connected node for the (iel)-th element
%
% coordinate of 1st node
x1=gcoord(nd(1),1); y1=gcoord(nd(1),2); z1=gcoord(nd(1),3);%
% coordinate of 2nd node
x2=gcoord(nd(2),1); y2=gcoord(nd(2),2); z2=gcoord(nd(2),3);
% element length
leng=sqrt((x2-xl)^2+(y2-yl)^2+(z2-zl)^2); % element length
% node coordinates
betx=(x2-xl)/leng;
bety=(y2-yl)/leng;
betz=(z2-zl)/leng;
el=elprop(iel,1); % extract elastic modulus
area=elprop(iel,2); % extract cross-sectional area
index=feeldof(nd,nnel,ndof); % extract system dofs for the element
k=fetruss2(el,leng,area,0,betx,1,bety,betz); % compute element matrix

for i=1:(nnel*ndof) % extract displacements associated with
disp(i)=disp(index(i)); % (iel)-th element
end
elforce=k*eldisp; % element force vector

% Stress calculation
stress(iel)=sqrt(elforce(1)^2+elforce(2)^2)/area;
if ((x2-xl)*elforce(3)) < 0;
stress(iel)= -stress(iel);
end
end

% Print fem solutions
num=1:1:sdof;
disp=[num' disp] % print displacements
numm=1:1:nel;
stresses=[numm' stress] % print stresses

% Subprogram: Feeldof
% Purpose: Compute System dofs Associated With each Element

function [index]=feeldof(nd,nnel,ndof)

% Synopsis:
% [index]=feeldof(nd,nnel,ndof)
% Variable Description:
% index - system dof vector associated with element
% "iel"
% iel - element number whose system dofs are to be
determined
% nnel - number of nodes per element
% ndof - number of dofs per node

edof = nnel*ndof;
k=0;
for i=1:nnel
    start = (nd(i)-1)*ndof;
    for j=1:ndof
        k=k+1;
        index(k)=start+j;
    end
end
Subprogram: fetruss3
Purpose: Compute Stiffness and Mass Matrices for the Space Truss Element. Nodal dof \{u_1 v_1 W_1 u_2 v_2 W_2\}

\[ \text{Synopsis:} \]
\[ [k,m]=\text{fetruss3}(el,leng,area,rho,beta,ipt) \]

Variable Description:
\( k \) - element stiffness matrix (size of 4x4)
\( m \) - element mass matrix (size of 4x4)
\( el \) - elastic modulus
\( leng \) - element length
\( area \) - area of truss cross-section
\( rho \) - mass density (mass per unit volume)
\( beta \) - angle between the local and global axes
\( ipt = 1 \): consistent mass matrix
\( ipt = 1 \) - consistent mass matrix
\( = 2 \) - lumped mass matrix

\[ \text{stiffness matrix} \]
\[ c=(\text{betx}); s=(\text{bety}); d=(\text{betz}); \]
\[ k=(\text{area}*el/leng)*[ \begin{bmatrix} c*c & c*s & c*d & -c*c & -c*s & -c*d; & \ldots \\ c*s & s*s & s*d & -c*s & -s*s & -s*d; & \ldots \\ c*d & s*d & d*d & -c*d & -s*d & -d*d; & \ldots \\ -c*c & -c*s & -c*d & c*c & c*s & c*d; & \ldots \\ -c*s & -s*s & -s*d & c*s & s*s & s*d; & \ldots \\ -c*d & -s*d & -d*d & c*d & s*d & d*d; & \ldots \end{bmatrix} ; \]

\[ \text{consistent mass matrix} \]
\[ \text{if } ipt==1 \]
\[ m=(\text{rho} \ast \text{area} \ast \text{leng} / 6)*[ \begin{bmatrix} 2*c*c & 2*c*s & 2*c*d & c*c & c*s & c*d; & \ldots \\ 2*c*s & 2*s*s & 2*s*d & c*s & s*s & s*d; & \ldots \\ 2*c*d & 2*s*d & 2*d*d & c*d & s*d & d*d; & \ldots \\ c*c & c*s & c*d & 2*c*c & 2*c*s & 2*c*d; & \ldots \\ c*s & s*s & s*d & 2*c*s & 2*s*s & 2*s*d; & \ldots \\ c*d & s*d & d*d & 2*c*d & 2*s*d & 2*d*d; & \ldots \end{bmatrix} ; \]

\[ \text{lumped mass matrix} \]
\[ \text{else} \]
\[ m=(\text{rho} \ast \text{area} \ast \text{leng} / 2)*[ \begin{bmatrix} c*c & c*s & c*d & 0 & 0 & 0; & \ldots \\ c*s & s*s & s*d & 0 & 0 & 0; & \ldots \\ c*d & s*d & d*d & 0 & 0 & 0; & \ldots \\ 0 & 0 & 0 & c*c & c*s & c*d; & \ldots \\ 0 & 0 & 0 & c*s & s*s & s*d; & \ldots \\ 0 & 0 & 0 & c*d & s*d & d*d; & \ldots \end{bmatrix} ; \]

Subprogram: feasmbl1
Purpose: Assembly of Element Matrices into the System Matrix
% Synopsis: 
% [kk]=feasmbl1(kk,k,index) 
% Variable Description: 
% kk - system matrix 
% k - element matrix 
% index - d.o.f. vector associated with an element

edof = length(index);
for i=1:edof
    ii=index(i);
    for j=1:edof
        jj=index(j);
        kk(ii,jj)=kk(ii,jj)+k(i,j);
    end
end

% Subprogram: feaplyc3
% Purpose: Apply Constraints to Matrix Equation.

function [kk,ff]=feaplyc3(kk,ff,bcdof,bcval)

n=length(bcdof);
sdof=size(kk);
for i=1:n
    c=bcdof(i);
    for j=1:sdof
        kk(c,j)=0;
    end
    kk(c,c)=1;
    ff(c)=bcval(i);
end
Appendix D

Source Code for the Geometrically Non-Linear Analysis of Plane Truss Structure (Chapter 5)

```matlab
% Geometrically Non-Linear Analysis
% Purpose: Geometrically Non-Linear Analysis of Plane Truss Structure

function [ST, le, DS, Bc] = FEM-p(A, bounds)

% Control Input Data
E = ..; % elastic modulus of elements
NB = ..; % number of elements
NN = ..; % number of nodes
NS = ..; % number of supports
NSTEP = 1.000; % number of load steps
NIT = 10.000; % number of iterations
FAC = 1;

% Initialize parameters/Arrays
if NSTEP == 0.000
    NSTEP = 1.000;
end
if NIT == 0.000;
    NIT = 1.000;
end
NNS = NN - NS;
N = 2*NNS;
H = 2*NN;

% Zero matrix for displacements and stresses
FSAVE = zeros(NB, 1);
P = zeros(H, 1);
DS = zeros(H, 1);
ST = zeros(NB, 1);

% Nodal coordinates
R(1) = ..; R(2) = ..; % x, y-coordinate of node 1
R(3) = ..; R(4) = ..; % x, y-coordinate of node 2
... % ...
R(..) = ..; R(..) = ..; % x, y-coordinate of node NN

% Applied nodal force
PSAVE(1) = ..; PSAVE(2) = ..; % Px, Py-LOAD node 1
PSAVE(3) = ..; PSAVE(4) = ..; % Px, Py-LOAD node 2
... % ...
PSAVE(.) = ..; PSAVE(.) = ..; % Px, Py-LOAD node NN

% Material and geometric properties
```

159
\% \text{cross-sectional area of element number 1}
\text{S(1)\ldots} \\
\% \text{cross-sectional area of element number 2}
\text{S(2)\ldots} \\
\% \text{cross-sectional area of element number NB}
\text{S(NB)\ldots}
\% \text{Nodal connectivity}
\% \text{nodes associated with element 1}
\text{NP(1)\ldots} \\
\text{NM(1)\ldots} \\
\% \text{nodes associated with element 2}
\text{NP(2)\ldots} \\
\text{NM(2)\ldots} \\
\% \text{nodes associated with element NB}
\text{NP(NB)\ldots} \\
\text{NM(NB)\ldots}
\% \text{Start load steps and iterations}
\% \text{for LDSTP=1:NSTEP}
\text{STEP=(LDSTP+1)/NSTEP;}
\text{if LDSTP==NSTEP;}
\text{STEP=1;}
\end
\text{for ITER=1:NIT}
\text{disp(' Iteration number: ')}; \text{ITER,}
\text{disp(' Load step:')} ; \text{LDSTP,}
\% \text{Set up system matrix}
\% \text{for I=1:N}
\text{P(I) = PSAVE(I)*STEP;}
\text{for J=1:N}
\text{C(I,J)=0;}
\end
\text{end}
\text{for L=1:NB}
\text{K=2*NP(L);} \\
\text{M=2*NM(L);} \\
\text{[UVEC,C1]=UNITV(K,M,R);} \\
\text{if K<=N}
\text{P(K-1)=P(K-1)-FSAVE(L)*UVEC(1);} \\
\text{P(K )=P(K )-FSAVE(L)*UVEC(2);} \\
\end
\text{if M<=N}
\text{P(M-1)=P(M-1)+FSAVE(L)*UVEC(1);} \\
\text{P(M )=P(M )+FSAVE(L)*UVEC(2);} \\
\end
\text{[C]=INSERT-P(UVEC,FSAVE(L),K,N,E,S(L),M,C1,C);} \\
\text{end}
\% \% \text{Error at start of iteration}
\% \text{C1=0;}
\text{for i=1:N}
\text{C1=C1+P(i)^2;}
\text{end}
\text{C1=sqrt(C1);} \\
\text{disp(' ERROR: ')}; \text{C1,}
\% \% \text{Solve for displacements}
\% \text{M=N-1;}
\text{for l=1:M}
L=I+1;
for J=L:N
    if C(J,I)<0 | C(J,I)>0
        for K=L:N
            C(J,K)=C(J,K)-C(I,K)*C(J,I)/C(I,I);
        end
        P(J)=P(J)-P(I)*C(J,I)/C(I,I); 
    end
end
end
P(N)=P(N)/C(N,N);
if C(N,N)<0
    disp('NEG TERM ON THE DIAGONAL AT ROW');K
end
for I=1:M
    K=N-I;
    L=K+1;
    for J=L:N
        P(K)=P(K)-P(J)*C(K,J);
    end
    if C(K,K)<0
        disp('NEG TERM ON THE DIAGONAL AT ROW');K
    end
    P(K)=P(K)/C(K,K);
end

% Compute member forces and displacements
%
for I=1:NB
    K=2*NP(I);
    M=2*NW(I);
    [UVEC,Cl]=UNITV(K,M,R);
    K1=K;
    D1=0;
    FAC=1;
    for J=1:2
        if K1<=N
            D1=D1+FAC*(P(K1-1)*UVEC(1) + P(K1 )*UVEC(2) ) ;
        end
        FAC=-1;
        K1=M;
    end
    F1=D1*E*S(I)/Cl;
    F2=F1/S(I);
    UVEC(1)=R(K-1)-R(M-1);
    UVEC(2)=R(K )-R(M );
    if K<=N
        UVEC(1)=UVEC(1)+P(K-1);
        UVEC(2)=UVEC(2)+P(K );
    end
    if M<=N
        UVEC(1)=UVEC(1)-P(M-1);
        UVEC(2)=UVEC(2);-
    end
    C2= sqrt(UVEC(1)^2+UVEC(2)^2+UVEC(3)^2);
    C2=C2-Cl;
    FSAVE(I)=FSAVE(I)+C2*S(I)*E/C1;
    D(I)=D1;
    f(I)=F1;
    ff(I)=F2;
end
% Undated coordinates
for b=1:N
    R(b)=R(b)+P(b);
end
% stress
DS=DS+P;
ST=ST+ff';
end
end

% Print member forces, stress and displacements

disp(' ***-member forces and displacements-***
');
disp(' Member......Force......Stress......Displacement');
numm=1:1:NB;
    [ numm', D', f', ST ' DS]

% Subprogram: UNITV
% Purpose: Unit vector components

function [UVEC,C1]=UNITV(K,M,R)

C1=0;
for I=1:2
    UVEC(I)=R(K+I-2) - R(M+I-2);
    C1=C1+UVEC(I)^2;
end
C1=sqrt(C1);
for I=1:2
    UVEC(I)=UVEC(I)/C1;
end

% Subprogram: INSERT-P
% Purpose: Compute System Matrix for the Plane Truss Element

function [C]= INSERT-P(UVEC,FSAVE,K,N,E,S,M,C1,C)

K1=K;
for I=1:2
    if K1<=N
        M1=K;
        for J=1:2
            if M1<=N
                FAC=1;
                if I<J | I>J
                    FAC=-1;
                end
                for L=1:2
                    I1=K1-2+L;
                    for L1=1:2
                        J1=M1-2+L1;
                        C(I1, J1)=C(I1, J1)+UVEC(I)*UVEC(J)*FAC;  
        end
    end
end
\[ C(I_1, J_1) = C(I_1, J_1) + UVEC(L) \cdot UVEC(L_1) \cdot (S \cdot E - FSAVE) \cdot FAC/C1; \]
if \( L = L_1 \)
\[ C(I_1, J_1) = C(I_1, J_1) + FAC \cdot FSAVE/C1; \]
end
end
M1 = M;
end
end
K1 = M;
end
end
Appendix E

Source Code for the Geometrically Non-Linear Analysis of Space Truss Structure (Chapter 5)

```
function [ST,le,DS,Bc]=FEM-s(A,bounds)

   % Control Input Data
   %
   E = ..;       % elastic modulus of elements
   NB=..;        % number of elements
   NN=..;        % number of nodes
   NS=..;        % number of supports
   NSTEP=1.000;  % number of load steps
   NIT=10.000;   % number of iterations
   FAC=1;
   %
   % Initialize parameters/Arrays
   %
   if NSTEP==0.000
      NSTEP = 1.000;
   end
   if NIT==0.000;
      NIT = 1.000;
   end
   NNS=NN-NS;
   N=3*NNS;
   H=3*NN;
   %
   % Zero matrix for displacements and stresses
   %
   PSAVE=zeros(NB,1);
   P=zeros(H,1);
   DS=zeros(H,1);
   ST=zeros(NB,1);
   %
   % Nodal coordinates
   %
   R(1)=..; R(2)=..; R(3)=..;   % x, y, z-coordinate of node 1
   R(4)=..; R(5)=..; R(6)=..;   % x, y, z-coordinate of node 2
   ...
   R(..)=..; R(..)=..; R(..)=..;   % x, y, z-coordinate of node NN
   %
   % Applied nodal force
   %
   PSAVE(1)=..; PSAVE(2)=..; PSAVE(3)=..; % Px, Py, Pz-LOAD node 1
   PSAVE(4)=..; PSAVE(5)=..; PSAVE(6)=..; % Px, Py, Pz-LOAD node 2
   ...
   PSAVE(.)=..; PSAVE(.)=..; PSAVE(.)=..; % Px, Py, Pz-LOAD node NN
   %
   % Material and geometric properties
   %
```

164
S(1)=..; % cross-sectional area of element number 1
S(2)=..; % cross-sectional area of element number 2
...
S(NB)=..; % cross-sectional area of element number NB

% Nodal connectivity
% NP(1)=..; NM(1)=..; % nodes associated with element 1
NP(2)=..; NM(2)=..; % nodes associated with element 2
...
NP(NB)=..; NM(NB)=..; % nodes associated with element NB

% Start load steps and iterations
% for LDSTP=1:NSTEP
  STEP=(LDSTP+1)/NSTEP;
  if LDSTP==NSTEP;
    STEP=1;
  end
  for ITER=1:NIT
    disp('Iteration number:');,ITER,
    disp('Load step:');,LDSTP,
    % % Set up system matrix
    % for I=1:N
      P(I) = PSAVE(I)*STEP;
      for J=1:N
        C(I,J)=0;
      end
    end
    for L=1:NB
      K=3*NP(L);
      M=3*NM(L);
      [UVEC, C1] = UNITV(K, M, R);
      if K<=N
        P(K-2)=P(K-2)-FSAVE(L)*UVEC(1);
        P(K-1)=P(K-1)-FSAVE(L)*UVEC(2);
        P(K )=P(K )-FSAVE(L)*UVEC(3);
      end
      if M<=N
        P(M-2)=P(M-2)+FSAVE(L)*UVEC(1);
        P(M-1)=P(M-1)+FSAVE(L)*UVEC(2);
        P(M )=P(M )+FSAVE(L)*UVEC(3);
      end
      [C] = INSERT-S(UVEC, FSAVE(L), K, N, E, S(L), M, C1, C);
    end
    % % Error at start of iteration
    % C1=0;
    for i=1:N
      C1=C1+P(i)^2;
    end
    C1=sqrt(C1);
    disp('ERROR:');,C1
    % % Solve for displacements
    %
\[ M = N - 1; \]
\[ \text{for } I = 1 : M \]
\[ \text{L} = I + 1; \]
\[ \text{for } J = L : N \]
\[ \text{if } C(J, I) < 0 \mid C(J, I) > 0 \]
\[ \text{for } K = L : N \]
\[ C(J, K) = C(J, K) - C(I, K) * C(J, I) / C(I, I); \]
\[ \text{end} \]
\[ P(J) = P(J) - P(I) * C(J, I) / C(I, I); \]
\[ \text{end} \]
\[ \text{end} \]
\[ P(N) = P(N) / C(N, N); \]
\[ \text{if } C(N, N) < 0 \]
\[ \text{disp('NEG TERM ON THE DIAGONAL AT ROW'); K} \]
\[ \text{end} \]
\[ \text{for } I = 1 : M \]
\[ K = N - I; \]
\[ \text{L} = K + 1; \]
\[ \text{for } J = L : N \]
\[ P(K) = P(K) - P(J) * C(K, J); \]
\[ \text{end} \]
\[ \text{if } C(K, K) < 0 \]
\[ \text{disp('NEG TERM ON THE DIAGONAL AT ROW'); K} \]
\[ \text{end} \]
\[ P(K) = P(K) / C(K, K); \]
\[ \text{end} \]
\% Compute member forces and displacements
\%
\[ \text{for } I = 1 : NB \]
\[ K = 3 * NP(I); \]
\[ M = 3 * NM(I); \]
\[ [\text{UVEC, C1}] = \text{UNITV}(K, M, R); \]
\[ K1 = K; \]
\[ D1 = 0; \]
\[ FAC = 1; \]
\[ \text{for } J = 1 : 2 \]
\[ \text{if } K1 < N \]
\[ D1 = D1 + FAC * (P(K1 - 2) * UVEC(1) + P(K1 - 1) * UVEC(2) + P(K1) * UVEC(3)); \]
\[ \text{end} \]
\[ FAC = -1; \]
\[ K1 = M; \]
\[ \text{end} \]
\[ F1 = D1 * E * S(I) / C1; \]
\[ F2 = F1 / S(I); \]
\[ \text{UVEC(1)} = R(K - 2) - R(M - 2); \]
\[ \text{UVEC(2)} = R(K - 1) - R(M - 1); \]
\[ \text{UVEC(3)} = R(K) - R(M); \]
\[ \text{if } K < N \]
\[ \text{UVEC(1)} = \text{UVEC(1)} + P(K - 2); \]
\[ \text{UVEC(2)} = \text{UVEC(2)} + P(K - 1); \]
\[ \text{UVEC(3)} = \text{UVEC(3)} + P(K); \]
\[ \text{end} \]
\[ \text{if } M < N \]
\[ \text{UVEC(1)} = \text{UVEC(1)} - P(M - 2); \]
\[ \text{UVEC(2)} = \text{UVEC(2)} - P(M - 1); \]
\[ \text{UVEC(3)} = \text{UVEC(3)} - P(M); \]
\[ \text{end} \]
\[ C2 = \sqrt{(UVEC(1)^2 + UVEC(2)^2 + UVEC(3)^2)}; \]
\[ C2 = C2 - C1; \]
FSAVE(I)=FSAVE(I)+C2*S(I)*E/C1;
D(I)=D1;
f(I)=F1;
ff(I)=F2;
end
% Undated coordinates
for b=1:N
    R(b)=R(b)+P(b);
end
% stress
    DS=DS+P;
    ST=ST+ff';
end
end
%
% Print member forces, stress and displacements
%
disp(' ***-member forces and displacements-*** ');
disp(' Member......Force......Stress......Displacement');
numm=1:1:NB;
    [ numm', D', f', ST ', DS]

% Subprogram: UNITV
% Purpose: Unit vector components
%
function [UVEC,C1]=UNITV(K,M,R)

%-----------------------------------------------
C1=0;
for I=1:3
    UVEC(I)=R(K+I-3)-R(M+I-3);
    C1=C1+UVEC(I)^2;
end
C1=sqrt(C1);
for I=1:3
    UVEC(I)=UVEC(I)/C1;
end

% Subprogram: INSERT-S
% Purpose: Compute System Matrix for the Space Truss Element
%
function [C]=INSERT-S(UVEC,FSAVE,K,N,E,S,M,C1,C)

K1=K;
for I=1:2
    if K1<=N
        M1=K;
        for J=1:2
            if M1<=N
                FAC=1;
            end
        end
    end
end
if I<J | I>J
  FAC=-1;
end
for L=1:3
  I1=KI-3+L;
  for L1=1:3
    J1=MI-3+L1;
    C(I1,J1)=C(I1,J1)+UVEC(L)*UVEC(L1)*(S*E-FSAVE)*FAC/C1;
    if L==L1
      C(I1,J1)=C(I1,J1)+FAC*FSAVE/C1;
    end
  end
end
M1=M;
end
K1=M;
end
end
<table>
<thead>
<tr>
<th>Designation</th>
<th>Mass per m</th>
<th>External Surface Area</th>
<th>Torsion Factor</th>
<th>Form Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>d₄</td>
<td>t</td>
<td>per m</td>
<td>per m²</td>
<td>10⁶</td>
</tr>
<tr>
<td>mm</td>
<td>mm</td>
<td>kg/m²</td>
<td>m²/m</td>
<td>m²/m²</td>
</tr>
<tr>
<td>405.4 x 12.7 CHS</td>
<td>9.5 CHS</td>
<td>123</td>
<td>1.28</td>
<td>10.4</td>
</tr>
<tr>
<td>405.4 x 12.7 CHS</td>
<td>6.4 CHS</td>
<td>63.1</td>
<td>1.28</td>
<td>20.2</td>
</tr>
<tr>
<td>405.4 x 12.7 CHS</td>
<td>4.0 CHS</td>
<td>34.7</td>
<td>1.12</td>
<td>32.2</td>
</tr>
<tr>
<td>355.6 x 12.7 CHS</td>
<td>9.5 CHS</td>
<td>81.1</td>
<td>1.12</td>
<td>13.8</td>
</tr>
<tr>
<td>355.6 x 12.7 CHS</td>
<td>6.4 CHS</td>
<td>55.1</td>
<td>1.12</td>
<td>20.3</td>
</tr>
<tr>
<td>355.6 x 12.7 CHS</td>
<td>4.0 CHS</td>
<td>34.8</td>
<td>1.12</td>
<td>32.2</td>
</tr>
<tr>
<td>323.9 x 12.7 CHS</td>
<td>9.5 CHS</td>
<td>73.7</td>
<td>1.02</td>
<td>13.6</td>
</tr>
<tr>
<td>323.9 x 12.7 CHS</td>
<td>6.4 CHS</td>
<td>50.1</td>
<td>1.02</td>
<td>20.3</td>
</tr>
<tr>
<td>323.9 x 12.7 CHS</td>
<td>4.0 CHS</td>
<td>34.8</td>
<td>1.02</td>
<td>32.2</td>
</tr>
<tr>
<td>273.1 x 12.7 CHS</td>
<td>9.3 CHS</td>
<td>60.5</td>
<td>0.858</td>
<td>14.2</td>
</tr>
<tr>
<td>273.1 x 12.7 CHS</td>
<td>6.4 CHS</td>
<td>42.1</td>
<td>0.858</td>
<td>20.4</td>
</tr>
<tr>
<td>273.1 x 12.7 CHS</td>
<td>4.8 CHS</td>
<td>31.8</td>
<td>0.858</td>
<td>14.2</td>
</tr>
</tbody>
</table>

Notes:
1. In this table, the properties of these products are calculated in accordance with AS 4100 using design yield stress \(\sigma_y = 350\) MPa and design tensile strength \(\sigma_t = 430\) MPa as per AS 4100.
2. In this table, the properties of these products are calculated in accordance with AS 4100 using design yield stress \(\sigma_y = 350\) MPa and design tensile strength \(\sigma_t = 430\) MPa as per AS 4100.
3. Type 2 and 3 products are not made strictly in accordance with AS 1163. Care should be used when designing structures using these products.
4. Grade C350L0 is cold formed and therefore is allocated the CF residual stresses classification in AS 4100.
5. C = Compact Section; N = Non-compact Section; S = Slender Section; as defined in AS 4100.
<table>
<thead>
<tr>
<th>DESIGNATION</th>
<th>MASS</th>
<th>DIA</th>
<th>AREA</th>
<th>SECTION</th>
<th>TORSION</th>
<th>MODULUS</th>
<th>AREA</th>
<th>TORSION</th>
<th>MODULUS</th>
<th>COMPACTNESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>219.1 x 12.7 CHS</td>
<td>64.6</td>
<td>0.688</td>
<td>10.6</td>
<td>17.3</td>
<td>8230</td>
<td>44.0</td>
<td>402</td>
<td>642</td>
<td>73.1</td>
<td>88.0</td>
</tr>
<tr>
<td>8.2 CHS</td>
<td>42.6</td>
<td>0.688</td>
<td>16.1</td>
<td>26.7</td>
<td>8430</td>
<td>30.9</td>
<td>276</td>
<td>365</td>
<td>74.8</td>
<td>60.5</td>
</tr>
<tr>
<td>6.4 CHS</td>
<td>33.6</td>
<td>0.688</td>
<td>20.8</td>
<td>34.2</td>
<td>4980</td>
<td>24.2</td>
<td>221</td>
<td>290</td>
<td>78.2</td>
<td>48.4</td>
</tr>
<tr>
<td>4.8 CHS</td>
<td>25.4</td>
<td>0.688</td>
<td>27.1</td>
<td>45.6</td>
<td>3230</td>
<td>18.6</td>
<td>169</td>
<td>220</td>
<td>75.8</td>
<td>37.1</td>
</tr>
<tr>
<td>4.0 CHS</td>
<td>21.2</td>
<td>0.688</td>
<td>32.4</td>
<td>54.8</td>
<td>2700</td>
<td>15.6</td>
<td>143</td>
<td>185</td>
<td>75.1</td>
<td>31.3</td>
</tr>
</tbody>
</table>

Table F.1 (Cont.) Dimensions and properties of circular hollow sections (AS 1163)
Table F.1 (Cont.) Dimensions and properties of circular hollow sections (AS 1163)

<table>
<thead>
<tr>
<th>Designation</th>
<th>Mass per m</th>
<th>External Surface Area</th>
<th>Gross Section Area</th>
<th>Torsion Constant</th>
<th>Torsion Modulus</th>
<th>Form Factor</th>
<th>Compaction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>kg/m</td>
<td>m²/m</td>
<td>m²/m²</td>
<td>J</td>
<td>C</td>
<td>A₀</td>
<td>Z</td>
</tr>
<tr>
<td>mm x CHS **</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(C,N,S)</td>
</tr>
<tr>
<td>88.9 x 5.8 CHS</td>
<td>11.3</td>
<td>0.279</td>
<td>24.7</td>
<td>16.2</td>
<td>1440</td>
<td>42.6</td>
<td>28.3</td>
</tr>
<tr>
<td>4.8 CHS</td>
<td>9.86</td>
<td>0.279</td>
<td>28.1</td>
<td>18.6</td>
<td>1670</td>
<td>1.72</td>
<td>25.3</td>
</tr>
<tr>
<td>3.2 CHS</td>
<td>6.76</td>
<td>0.279</td>
<td>41.3</td>
<td>27.8</td>
<td>1051</td>
<td>0.792</td>
<td>17.8</td>
</tr>
<tr>
<td>2.6 CHS</td>
<td>5.53</td>
<td>0.279</td>
<td>50.5</td>
<td>34.2</td>
<td>705</td>
<td>0.657</td>
<td>14.9</td>
</tr>
<tr>
<td>76.1 x 3.2 CHS</td>
<td>5.75</td>
<td>0.239</td>
<td>41.8</td>
<td>23.8</td>
<td>833</td>
<td>0.486</td>
<td>12.8</td>
</tr>
<tr>
<td>2.3 CHS</td>
<td>4.18</td>
<td>0.239</td>
<td>37.1</td>
<td>33.1</td>
<td>533</td>
<td>0.383</td>
<td>9.55</td>
</tr>
<tr>
<td>60.3 x 2.9 CHS **</td>
<td>4.11</td>
<td>0.189</td>
<td>45.1</td>
<td>20.8</td>
<td>523</td>
<td>0.216</td>
<td>7.16</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>3.29</td>
<td>0.189</td>
<td>57.6</td>
<td>29.2</td>
<td>419</td>
<td>0.177</td>
<td>5.65</td>
</tr>
<tr>
<td>49.3 x 3.2 CHS</td>
<td>3.56</td>
<td>0.152</td>
<td>42.6</td>
<td>15.1</td>
<td>453</td>
<td>0.110</td>
<td>4.80</td>
</tr>
<tr>
<td>2.9 CHS</td>
<td>3.25</td>
<td>0.152</td>
<td>46.7</td>
<td>16.7</td>
<td>414</td>
<td>0.107</td>
<td>4.43</td>
</tr>
<tr>
<td>2.3 CHS**</td>
<td>2.61</td>
<td>0.152</td>
<td>58.2</td>
<td>21.0</td>
<td>332</td>
<td>0.0881</td>
<td>3.65</td>
</tr>
<tr>
<td>42.4 x 2.6 CHS</td>
<td>2.55</td>
<td>0.133</td>
<td>52.2</td>
<td>16.3</td>
<td>325</td>
<td>0.0646</td>
<td>3.05</td>
</tr>
<tr>
<td>2.6 CHS**</td>
<td>1.89</td>
<td>0.133</td>
<td>68.8</td>
<td>21.2</td>
<td>254</td>
<td>0.0519</td>
<td>2.45</td>
</tr>
<tr>
<td>33.7 x 2.9 CHS</td>
<td>1.99</td>
<td>0.106</td>
<td>53.1</td>
<td>13.0</td>
<td>234</td>
<td>0.0309</td>
<td>1.84</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>1.56</td>
<td>0.106</td>
<td>67.7</td>
<td>16.8</td>
<td>189</td>
<td>0.0251</td>
<td>1.49</td>
</tr>
<tr>
<td>26.9 x 2.3 CHS</td>
<td>1.40</td>
<td>0.0845</td>
<td>60.6</td>
<td>11.7</td>
<td>178</td>
<td>0.0130</td>
<td>1.10</td>
</tr>
<tr>
<td>2.3 CHS**</td>
<td>1.23</td>
<td>0.0845</td>
<td>68.8</td>
<td>13.5</td>
<td>156</td>
<td>0.0122</td>
<td>0.97</td>
</tr>
<tr>
<td>21.3 x 3.2 CHS</td>
<td>1.43</td>
<td>0.0669</td>
<td>46.8</td>
<td>6.66</td>
<td>135</td>
<td>0.00768</td>
<td>0.722</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>1.20</td>
<td>0.0660</td>
<td>55.8</td>
<td>8.19</td>
<td>153</td>
<td>0.00884</td>
<td>0.539</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>0.852</td>
<td>0.0669</td>
<td>70.3</td>
<td>10.7</td>
<td>121</td>
<td>0.00571</td>
<td>0.538</td>
</tr>
<tr>
<td>17.2 x 2.9 CHS**</td>
<td>1.02</td>
<td>0.0540</td>
<td>52.8</td>
<td>5.93</td>
<td>130</td>
<td>0.00347</td>
<td>0.463</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>0.845</td>
<td>0.0540</td>
<td>68.9</td>
<td>7.48</td>
<td>108</td>
<td>0.00306</td>
<td>0.356</td>
</tr>
<tr>
<td>13.5 x 2.9 CHS</td>
<td>0.768</td>
<td>0.0424</td>
<td>55.9</td>
<td>4.66</td>
<td>96.9</td>
<td>0.00148</td>
<td>0.218</td>
</tr>
<tr>
<td>2.9 CHS**</td>
<td>0.535</td>
<td>0.0424</td>
<td>65.8</td>
<td>5.87</td>
<td>80.9</td>
<td>0.00132</td>
<td>0.186</td>
</tr>
</tbody>
</table>

Note: CHS = Circular Hollow Section
Appendix G

Stresses and displacements obtained for the final solutions of all solved examples (Chapter 6)

Table G.1 Stresses and displacements obtained for the discrete optimised solution for example 6.1 (10-bar plane truss) based on non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>(Case 1)*</th>
<th>(Case 2)**</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5 3</td>
<td>46.5</td>
<td>51.7</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3 1</td>
<td>-2.6</td>
<td>-12.2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6 4</td>
<td>-54.5</td>
<td>-50.5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4 2</td>
<td>-57.3</td>
<td>-36.2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4 3</td>
<td>167.5</td>
<td>167.2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1 2</td>
<td>-2.6</td>
<td>-12.2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>5 4</td>
<td>119.6</td>
<td>136.7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>6 3</td>
<td>-39.9</td>
<td>-52.7</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3 2</td>
<td>48.8</td>
<td>36.2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>4 1</td>
<td>3.8</td>
<td>17.2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
<th>(Case 1)*</th>
<th>(Case 2)**</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(direction)</td>
<td>(direction)</td>
</tr>
<tr>
<td>1</td>
<td>5.7</td>
<td>-49.9</td>
<td>5.2</td>
</tr>
<tr>
<td>2</td>
<td>-14.9</td>
<td>-49.7</td>
<td>-11.6</td>
</tr>
<tr>
<td>3</td>
<td>6.2</td>
<td>-16.8</td>
<td>6.8</td>
</tr>
<tr>
<td>4</td>
<td>-7.3</td>
<td>-39.1</td>
<td>-6.8</td>
</tr>
<tr>
<td>5</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

*Case 1: Results of non-linear analysis based on the solution from catalogue 1.
**Case 2: Results of non-linear analysis based on the solution from catalogue 2.
Table G.2 Stresses obtained for the optimised solution for example 6.2 (20-bar plane truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>(Case 2)**</th>
<th>(Case 3)***</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 2</td>
<td>98</td>
<td>98</td>
<td>96</td>
</tr>
<tr>
<td>2</td>
<td>1 3</td>
<td>-135</td>
<td>-135</td>
<td>-137</td>
</tr>
<tr>
<td>3</td>
<td>2 3</td>
<td>89</td>
<td>77</td>
<td>89</td>
</tr>
<tr>
<td>4</td>
<td>2 4</td>
<td>113</td>
<td>113</td>
<td>113</td>
</tr>
<tr>
<td>5</td>
<td>3 5</td>
<td>-147</td>
<td>-155</td>
<td>-155</td>
</tr>
<tr>
<td>6</td>
<td>3 4</td>
<td>148</td>
<td>146</td>
<td>146</td>
</tr>
<tr>
<td>7</td>
<td>2 5</td>
<td>-86</td>
<td>-99</td>
<td>-99</td>
</tr>
<tr>
<td>8</td>
<td>4 6</td>
<td>189</td>
<td>189</td>
<td>189</td>
</tr>
<tr>
<td>9</td>
<td>4 5</td>
<td>70</td>
<td>59</td>
<td>59</td>
</tr>
<tr>
<td>10</td>
<td>5 7</td>
<td>-164</td>
<td>-173</td>
<td>-173</td>
</tr>
<tr>
<td>11</td>
<td>5 6</td>
<td>43</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>12</td>
<td>4 7</td>
<td>27</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>13</td>
<td>6 8</td>
<td>108</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>14</td>
<td>6 7</td>
<td>44</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>15</td>
<td>7 9</td>
<td>-151</td>
<td>-142</td>
<td>-142</td>
</tr>
<tr>
<td>16</td>
<td>7 8</td>
<td>-125</td>
<td>-73</td>
<td>-73</td>
</tr>
<tr>
<td>17</td>
<td>6 9</td>
<td>156</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>18</td>
<td>8 10</td>
<td>98</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td>19</td>
<td>9 8</td>
<td>62</td>
<td>147</td>
<td>147</td>
</tr>
<tr>
<td>20</td>
<td>9 10</td>
<td>-135</td>
<td>-137</td>
<td>-137</td>
</tr>
</tbody>
</table>

**Case2: Results of non-linear analysis for 20-bar truss.
***Case3: Results of linear analysis for 20-bar truss.
Table G.3 Displacements obtained for the optimised solution for example 6.2 (20-bar plane truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Case 2)**</td>
<td>x-direction</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2.3</td>
<td>-21.3</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>12.8</td>
<td>-19.3</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>4.9</td>
<td>-34.1</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>9.3</td>
<td>-32.4</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>9.5</td>
<td>-34.3</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>5.3</td>
<td>-33.2</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>12.0</td>
<td>-20.5</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>1.7</td>
<td>-19.1</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>14.3</td>
<td>-0.0</td>
</tr>
</tbody>
</table>

**Case 2: Results of non-linear analysis for 20-bar truss.

***Case 3: Results of linear analysis for 20-bar truss.
Table G.4 Stresses obtained for the optimised solution for example 6.3 (51-bar roof truss) based on non-linear analysis (Case 3)***.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 4</td>
<td>-4.1</td>
<td>27</td>
<td>26 27</td>
<td>-37.4</td>
</tr>
<tr>
<td>2</td>
<td>4 6</td>
<td>55.1</td>
<td>28</td>
<td>2 3</td>
<td>128.5</td>
</tr>
<tr>
<td>3</td>
<td>6 8</td>
<td>88.6</td>
<td>29</td>
<td>3 4</td>
<td>-118.2</td>
</tr>
<tr>
<td>4</td>
<td>8 10</td>
<td>106.1</td>
<td>30</td>
<td>4 5</td>
<td>79.2</td>
</tr>
<tr>
<td>5</td>
<td>10 12</td>
<td>112.9</td>
<td>31</td>
<td>5 6</td>
<td>-75.1</td>
</tr>
<tr>
<td>6</td>
<td>12 14</td>
<td>112.2</td>
<td>32</td>
<td>6 7</td>
<td>44.9</td>
</tr>
<tr>
<td>7</td>
<td>14 16</td>
<td>106.2</td>
<td>33</td>
<td>7 8</td>
<td>-43.6</td>
</tr>
<tr>
<td>8</td>
<td>16 18</td>
<td>112.9</td>
<td>34</td>
<td>8 9</td>
<td>18.9</td>
</tr>
<tr>
<td>9</td>
<td>18 20</td>
<td>114.2</td>
<td>35</td>
<td>9 10</td>
<td>-18.9</td>
</tr>
<tr>
<td>10</td>
<td>20 22</td>
<td>108.1</td>
<td>36</td>
<td>10 11</td>
<td>-2.3</td>
</tr>
<tr>
<td>11</td>
<td>22 24</td>
<td>91.4</td>
<td>37</td>
<td>11 12</td>
<td>1.7</td>
</tr>
<tr>
<td>12</td>
<td>24 26</td>
<td>58.6</td>
<td>38</td>
<td>12 13</td>
<td>-20.2</td>
</tr>
<tr>
<td>13</td>
<td>1 3</td>
<td>0.3</td>
<td>39</td>
<td>13 14</td>
<td>19.5</td>
</tr>
<tr>
<td>14</td>
<td>3 5</td>
<td>-51.6</td>
<td>40</td>
<td>14 15</td>
<td>21.7</td>
</tr>
<tr>
<td>15</td>
<td>5 7</td>
<td>-51.1</td>
<td>41</td>
<td>15 16</td>
<td>-22.4</td>
</tr>
<tr>
<td>16</td>
<td>7 9</td>
<td>-37.6</td>
<td>42</td>
<td>16 17</td>
<td>3.7</td>
</tr>
<tr>
<td>17</td>
<td>9 11</td>
<td>-89.3</td>
<td>43</td>
<td>17 18</td>
<td>-4.2</td>
</tr>
<tr>
<td>18</td>
<td>11 13</td>
<td>-116.3</td>
<td>44</td>
<td>18 19</td>
<td>-16.9</td>
</tr>
<tr>
<td>19</td>
<td>13 15</td>
<td>-127.3</td>
<td>45</td>
<td>19 20</td>
<td>16.9</td>
</tr>
<tr>
<td>20</td>
<td>15 17</td>
<td>-127.5</td>
<td>46</td>
<td>20 21</td>
<td>-41.9</td>
</tr>
<tr>
<td>21</td>
<td>17 19</td>
<td>-119.8</td>
<td>47</td>
<td>21 22</td>
<td>43.1</td>
</tr>
<tr>
<td>22</td>
<td>19 21</td>
<td>-119.4</td>
<td>48</td>
<td>22 23</td>
<td>-73.4</td>
</tr>
<tr>
<td>23</td>
<td>21 23</td>
<td>-127.1</td>
<td>49</td>
<td>23 24</td>
<td>77.4</td>
</tr>
<tr>
<td>24</td>
<td>23 25</td>
<td>-126.9</td>
<td>50</td>
<td>24 25</td>
<td>-116.6</td>
</tr>
<tr>
<td>25</td>
<td>25 27</td>
<td>-115.9</td>
<td>51</td>
<td>25 26</td>
<td>126.7</td>
</tr>
<tr>
<td>26</td>
<td>1 2</td>
<td>-88.9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

***Case3: Results of non-linear analysis for 51-bar roof truss.
Table G.5 Displacements obtained for the optimised solution for example 6.3 (51-bar roof truss) based on non-linear analysis (Case 3)***. 

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x-direction</td>
<td>y-direction</td>
</tr>
<tr>
<td>1</td>
<td>-0.0</td>
<td>-0.0</td>
</tr>
<tr>
<td>2</td>
<td>6.3</td>
<td>-0.3</td>
</tr>
<tr>
<td>3</td>
<td>-0.1</td>
<td>-7.8</td>
</tr>
<tr>
<td>4</td>
<td>7.4</td>
<td>-15.1</td>
</tr>
<tr>
<td>5</td>
<td>0.4</td>
<td>-21.6</td>
</tr>
<tr>
<td>6</td>
<td>7.7</td>
<td>-27.6</td>
</tr>
<tr>
<td>7</td>
<td>1.3</td>
<td>-32.6</td>
</tr>
<tr>
<td>8</td>
<td>7.5</td>
<td>-36.9</td>
</tr>
<tr>
<td>9</td>
<td>2.3</td>
<td>-40.5</td>
</tr>
<tr>
<td>10</td>
<td>6.9</td>
<td>-43.3</td>
</tr>
<tr>
<td>11</td>
<td>3.3</td>
<td>-45.3</td>
</tr>
<tr>
<td>12</td>
<td>5.9</td>
<td>-46.5</td>
</tr>
<tr>
<td>13</td>
<td>4.4</td>
<td>-47.1</td>
</tr>
<tr>
<td>14</td>
<td>4.9</td>
<td>-47.1</td>
</tr>
<tr>
<td>15</td>
<td>5.4</td>
<td>-47.1</td>
</tr>
<tr>
<td>16</td>
<td>3.8</td>
<td>-46.6</td>
</tr>
<tr>
<td>17</td>
<td>6.5</td>
<td>-45.4</td>
</tr>
<tr>
<td>18</td>
<td>2.9</td>
<td>-43.4</td>
</tr>
<tr>
<td>19</td>
<td>7.6</td>
<td>-40.6</td>
</tr>
<tr>
<td>20</td>
<td>2.3</td>
<td>-37.2</td>
</tr>
<tr>
<td>21</td>
<td>8.6</td>
<td>-32.7</td>
</tr>
<tr>
<td>22</td>
<td>2.1</td>
<td>-27.7</td>
</tr>
<tr>
<td>23</td>
<td>9.4</td>
<td>-21.7</td>
</tr>
<tr>
<td>24</td>
<td>2.5</td>
<td>-15.2</td>
</tr>
<tr>
<td>25</td>
<td>9.9</td>
<td>-7.9</td>
</tr>
<tr>
<td>26</td>
<td>3.5</td>
<td>-0.3</td>
</tr>
<tr>
<td>27</td>
<td>9.9</td>
<td>-0.0</td>
</tr>
</tbody>
</table>

***Case3: Results of non-linear analysis for 51-bar roof truss.
Table G.6 Stresses obtained for the optimised solution for example 6.4 (25-bar space truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(Case 1)*</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-50.4</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>-5.5</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>26.9</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>-5.9</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>-48.9</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>8.9</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>-46.9</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>28.9</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>-26.2</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>4.2</td>
</tr>
<tr>
<td>11</td>
<td>4</td>
<td>-33.1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>19.3</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>-19.3</td>
</tr>
<tr>
<td>14</td>
<td>3</td>
<td>33.1</td>
</tr>
<tr>
<td>15</td>
<td>6</td>
<td>-26.2</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>8.3</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>-51.7</td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>8.9</td>
</tr>
<tr>
<td>19</td>
<td>3</td>
<td>17.2</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>-31.0</td>
</tr>
<tr>
<td>21</td>
<td>6</td>
<td>-18.6</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>-22.7</td>
</tr>
<tr>
<td>23</td>
<td>3</td>
<td>31.7</td>
</tr>
<tr>
<td>24</td>
<td>4</td>
<td>0.7</td>
</tr>
<tr>
<td>25</td>
<td>5</td>
<td>-53.1</td>
</tr>
</tbody>
</table>

*Case 1: Results of linear analysis for 25-bar space truss.
**Case 2: Results of linear non-analysis for 25-bar space truss.
Table G.7 Displacements obtained for the optimised solution for example 6.4 (25-bar space truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(Case 1)* direction</td>
</tr>
<tr>
<td></td>
<td>x</td>
</tr>
<tr>
<td>1</td>
<td>5.0978</td>
</tr>
<tr>
<td>3</td>
<td>-0.0229</td>
</tr>
<tr>
<td>4</td>
<td>0.5156</td>
</tr>
<tr>
<td>5</td>
<td>-0.1702</td>
</tr>
<tr>
<td>6</td>
<td>0.3531</td>
</tr>
<tr>
<td>7</td>
<td>0.0000</td>
</tr>
<tr>
<td>8</td>
<td>0.0000</td>
</tr>
<tr>
<td>9</td>
<td>0.0000</td>
</tr>
<tr>
<td>10</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

*Case 1: Results of linear analysis for 25-bar space truss.
**Case 2: Results of linear non-analysis for 25-bar space truss.
Table G.8 Stresses obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis Case 1*.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 2</td>
<td>-0.0 -0.4</td>
<td>27</td>
<td>13 6</td>
<td>-53.9 -52.9</td>
</tr>
<tr>
<td>2</td>
<td>1 3</td>
<td>0.0 -0.4</td>
<td>28</td>
<td>13 6</td>
<td>-54.9 -52.7</td>
</tr>
<tr>
<td>3</td>
<td>1 4</td>
<td>-0.0 -0.4</td>
<td>29</td>
<td>6 14</td>
<td>-58.4 -68.6</td>
</tr>
<tr>
<td>4</td>
<td>1 5</td>
<td>0.0 -0.4</td>
<td>30</td>
<td>7 15</td>
<td>-37.3 -42.5</td>
</tr>
<tr>
<td>5</td>
<td>2 3</td>
<td>-24.6 -53.6</td>
<td>31</td>
<td>8 16</td>
<td>-58.4 -68.6</td>
</tr>
<tr>
<td>6</td>
<td>3 4</td>
<td>-24.6 -53.6</td>
<td>32</td>
<td>9 17</td>
<td>-37.3 -42.5</td>
</tr>
<tr>
<td>7</td>
<td>4 5</td>
<td>-24.6 -53.6</td>
<td>33</td>
<td>10 18</td>
<td>-58.4 -68.6</td>
</tr>
<tr>
<td>8</td>
<td>5 2</td>
<td>-24.6 -53.6</td>
<td>34</td>
<td>11 19</td>
<td>-37.3 -42.5</td>
</tr>
<tr>
<td>9</td>
<td>2 6</td>
<td>-41.4 -31.6</td>
<td>35</td>
<td>12 20</td>
<td>-58.4 -68.6</td>
</tr>
<tr>
<td>10</td>
<td>3 8</td>
<td>-41.4 -31.6</td>
<td>36</td>
<td>13 21</td>
<td>-37.3 -42.5</td>
</tr>
<tr>
<td>11</td>
<td>4 10</td>
<td>-41.4 -31.6</td>
<td>37</td>
<td>6 15</td>
<td>-38.4 -136.6</td>
</tr>
<tr>
<td>12</td>
<td>5 12</td>
<td>-41.4 -31.6</td>
<td>38</td>
<td>15 8</td>
<td>-114.3 -18.7</td>
</tr>
<tr>
<td>13</td>
<td>2 7</td>
<td>29.9 24.1</td>
<td>39</td>
<td>8 17</td>
<td>-38.4 -136.6</td>
</tr>
<tr>
<td>14</td>
<td>3 7</td>
<td>29.9 24.1</td>
<td>40</td>
<td>17 10</td>
<td>-114.3 -18.7</td>
</tr>
<tr>
<td>15</td>
<td>3 9</td>
<td>29.9 24.1</td>
<td>41</td>
<td>10 19</td>
<td>-38.4 -136.6</td>
</tr>
<tr>
<td>16</td>
<td>4 9</td>
<td>29.9 24.1</td>
<td>42</td>
<td>19 12</td>
<td>-114.3 -18.7</td>
</tr>
<tr>
<td>17</td>
<td>4 11</td>
<td>29.9 24.1</td>
<td>43</td>
<td>12 21</td>
<td>-38.4 -136.6</td>
</tr>
<tr>
<td>18</td>
<td>5 11</td>
<td>29.9 24.1</td>
<td>44</td>
<td>21 6</td>
<td>-114.3 -18.7</td>
</tr>
<tr>
<td>19</td>
<td>5 13</td>
<td>29.9 24.1</td>
<td>45</td>
<td>14 7</td>
<td>-43.7 56.7</td>
</tr>
<tr>
<td>20</td>
<td>2 13</td>
<td>29.9 24.1</td>
<td>46</td>
<td>7 16</td>
<td>36.2 -0.42</td>
</tr>
<tr>
<td>21</td>
<td>6 7</td>
<td>-53.9 -52.9</td>
<td>47</td>
<td>16 9</td>
<td>-43.7 68.5</td>
</tr>
<tr>
<td>22</td>
<td>7 8</td>
<td>-54.9 -52.9</td>
<td>48</td>
<td>9 18</td>
<td>36.2 -68.5</td>
</tr>
<tr>
<td>23</td>
<td>8 9</td>
<td>-53.9 -52.9</td>
<td>49</td>
<td>18 11</td>
<td>-43.7 56.7</td>
</tr>
<tr>
<td>24</td>
<td>9 10</td>
<td>-54.9 -52.9</td>
<td>50</td>
<td>11 20</td>
<td>36.2 -68.5</td>
</tr>
<tr>
<td>25</td>
<td>0 11</td>
<td>-53.9 -52.9</td>
<td>51</td>
<td>20 13</td>
<td>-43.7 56.7</td>
</tr>
<tr>
<td>26</td>
<td>1 12</td>
<td>-54.9 -42.7</td>
<td>52</td>
<td>13 14</td>
<td>36.2 -68.5</td>
</tr>
</tbody>
</table>

*Case 1:

b) Results of linear analysis for 52-bar dome space truss.

c) Results of non-linear analysis for 52-bar dome space truss.
Table G.9 Displacements obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis Case 1*.

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
<th>Case 1*(b)</th>
<th>Case 1*(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x</td>
<td>y</td>
<td>z</td>
</tr>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>2.3</td>
</tr>
<tr>
<td>2</td>
<td>-0.6</td>
<td>-1.2</td>
<td>-0.6</td>
</tr>
<tr>
<td>3</td>
<td>1.2</td>
<td>-0.6</td>
<td>-0.6</td>
</tr>
<tr>
<td>4</td>
<td>0.6</td>
<td>1.2</td>
<td>-0.6</td>
</tr>
<tr>
<td>5</td>
<td>-1.2</td>
<td>0.6</td>
<td>-0.6</td>
</tr>
<tr>
<td>6</td>
<td>-8.2</td>
<td>-2.4</td>
<td>-9.9</td>
</tr>
<tr>
<td>7</td>
<td>2.7</td>
<td>-0.9</td>
<td>-0.5</td>
</tr>
<tr>
<td>8</td>
<td>2.4</td>
<td>-8.3</td>
<td>-9.9</td>
</tr>
<tr>
<td>9</td>
<td>0.9</td>
<td>2.7</td>
<td>-0.5</td>
</tr>
<tr>
<td>10</td>
<td>8.3</td>
<td>2.4</td>
<td>-9.9</td>
</tr>
<tr>
<td>11</td>
<td>-2.7</td>
<td>0.9</td>
<td>-0.5</td>
</tr>
<tr>
<td>12</td>
<td>-2.4</td>
<td>8.3</td>
<td>-9.9</td>
</tr>
<tr>
<td>13</td>
<td>-0.9</td>
<td>-2.7</td>
<td>-0.5</td>
</tr>
<tr>
<td>14</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>15</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>16</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>17</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>18</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>19</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>20</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>21</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

*Case 1:

b) Results of linear analysis for 52-bar dome space truss.

c) Results of non-linear analysis for 52-bar dome space truss.
Table G.10 Stresses obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis Case 2**.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(b)</td>
<td></td>
<td></td>
<td>(c)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>-0.0</td>
<td>27</td>
<td>2</td>
<td>70.3</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-0.2</td>
<td></td>
<td>3</td>
<td>71.6</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>-0.0</td>
<td></td>
<td>5</td>
<td>62.1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-0.2</td>
<td></td>
<td>7</td>
<td>62.1</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>0.0</td>
<td></td>
<td>10</td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0.0</td>
<td></td>
<td>12</td>
<td>55.6</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>25.7</td>
<td></td>
<td>15</td>
<td>39.0</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>25.7</td>
<td></td>
<td>16</td>
<td>55.6</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>25.7</td>
<td></td>
<td>17</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>25.7</td>
<td></td>
<td>18</td>
<td>55.6</td>
</tr>
<tr>
<td>6</td>
<td>21</td>
<td>25.7</td>
<td></td>
<td>19</td>
<td>55.6</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
<td>25.7</td>
<td></td>
<td>20</td>
<td>55.6</td>
</tr>
<tr>
<td>8</td>
<td>23</td>
<td>25.7</td>
<td></td>
<td>21</td>
<td>55.6</td>
</tr>
<tr>
<td>9</td>
<td>24</td>
<td>25.7</td>
<td></td>
<td>22</td>
<td>55.6</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>25.7</td>
<td></td>
<td>23</td>
<td>55.6</td>
</tr>
<tr>
<td>11</td>
<td>26</td>
<td>25.7</td>
<td></td>
<td>24</td>
<td>55.6</td>
</tr>
<tr>
<td>12</td>
<td>27</td>
<td>25.7</td>
<td></td>
<td>25</td>
<td>55.6</td>
</tr>
<tr>
<td>13</td>
<td>28</td>
<td>25.7</td>
<td></td>
<td>26</td>
<td>55.6</td>
</tr>
<tr>
<td>14</td>
<td>29</td>
<td>25.7</td>
<td></td>
<td>27</td>
<td>55.6</td>
</tr>
<tr>
<td>15</td>
<td>30</td>
<td>25.7</td>
<td></td>
<td>28</td>
<td>55.6</td>
</tr>
<tr>
<td>16</td>
<td>31</td>
<td>25.7</td>
<td></td>
<td>29</td>
<td>55.6</td>
</tr>
<tr>
<td>17</td>
<td>32</td>
<td>25.7</td>
<td></td>
<td>30</td>
<td>55.6</td>
</tr>
<tr>
<td>18</td>
<td>33</td>
<td>25.7</td>
<td></td>
<td>31</td>
<td>55.6</td>
</tr>
<tr>
<td>19</td>
<td>34</td>
<td>25.7</td>
<td></td>
<td>32</td>
<td>55.6</td>
</tr>
<tr>
<td>20</td>
<td>35</td>
<td>25.7</td>
<td></td>
<td>33</td>
<td>55.6</td>
</tr>
<tr>
<td>21</td>
<td>36</td>
<td>25.7</td>
<td></td>
<td>34</td>
<td>55.6</td>
</tr>
<tr>
<td>22</td>
<td>37</td>
<td>25.7</td>
<td></td>
<td>35</td>
<td>55.6</td>
</tr>
<tr>
<td>23</td>
<td>38</td>
<td>25.7</td>
<td></td>
<td>36</td>
<td>55.6</td>
</tr>
<tr>
<td>24</td>
<td>39</td>
<td>25.7</td>
<td></td>
<td>37</td>
<td>55.6</td>
</tr>
<tr>
<td>25</td>
<td>40</td>
<td>25.7</td>
<td></td>
<td>38</td>
<td>55.6</td>
</tr>
<tr>
<td>26</td>
<td>41</td>
<td>25.7</td>
<td></td>
<td>39</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>42</td>
<td>25.7</td>
<td></td>
<td>40</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>43</td>
<td>25.7</td>
<td></td>
<td>41</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>44</td>
<td>25.7</td>
<td></td>
<td>42</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>45</td>
<td>25.7</td>
<td></td>
<td>43</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>46</td>
<td>25.7</td>
<td></td>
<td>44</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>47</td>
<td>25.7</td>
<td></td>
<td>45</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>48</td>
<td>25.7</td>
<td></td>
<td>46</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>49</td>
<td>25.7</td>
<td></td>
<td>47</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>25.7</td>
<td></td>
<td>48</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>25.7</td>
<td></td>
<td>49</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>25.7</td>
<td></td>
<td>50</td>
<td>55.6</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>25.7</td>
<td></td>
<td>52</td>
<td>55.6</td>
</tr>
</tbody>
</table>

**Case 2:

b) Results of linear analysis for 52-bar dome space truss.

c) Results of non-linear analysis for 52-bar dome space truss.
Table G.11 Displacements obtained for the optimised solution for example 6.5 (52-bar dome space truss) based on linear and non-linear analysis Case 2**.

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm) Case 2**(b)</th>
<th>Nodal displacement (mm) Case 2**(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
<td>y</td>
</tr>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>0.6</td>
<td>1.2</td>
</tr>
<tr>
<td>3</td>
<td>-1.2</td>
<td>0.6</td>
</tr>
<tr>
<td>4</td>
<td>-0.6</td>
<td>-1.3</td>
</tr>
<tr>
<td>5</td>
<td>1.3</td>
<td>-0.6</td>
</tr>
<tr>
<td>6</td>
<td>9.8</td>
<td>2.6</td>
</tr>
<tr>
<td>7</td>
<td>-2.6</td>
<td>1.2</td>
</tr>
<tr>
<td>8</td>
<td>-2.6</td>
<td>9.8</td>
</tr>
<tr>
<td>9</td>
<td>-1.3</td>
<td>-2.6</td>
</tr>
<tr>
<td>10</td>
<td>-9.8</td>
<td>-2.6</td>
</tr>
<tr>
<td>11</td>
<td>2.6</td>
<td>-1.2</td>
</tr>
<tr>
<td>12</td>
<td>2.6</td>
<td>-9.8</td>
</tr>
<tr>
<td>13</td>
<td>1.2</td>
<td>2.6</td>
</tr>
<tr>
<td>14</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>15</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>16</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>17</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>18</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>19</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>20</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>21</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Case 2:

b) Results of linear analysis for 52-bar dome space truss.

c) Results of non-linear analysis for 52-bar dome space truss.
Table G.12 Stresses obtained for the optimised solution for example 6.6 (56-bar transmission tower space truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress (MPa)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Case 2</strong></td>
<td><strong>Case 3</strong>*</td>
<td></td>
<td><strong>Case 2</strong></td>
</tr>
<tr>
<td>1</td>
<td>1 2</td>
<td>1.6</td>
<td>2.2</td>
<td>29</td>
<td>1 15</td>
</tr>
<tr>
<td>2</td>
<td>2 3</td>
<td>-54.4</td>
<td>-62.6</td>
<td>30</td>
<td>12 16</td>
</tr>
<tr>
<td>3</td>
<td>3 4</td>
<td>1.9</td>
<td>2.2</td>
<td>31</td>
<td>13 14</td>
</tr>
<tr>
<td>4</td>
<td>4 1</td>
<td>1.9</td>
<td>2.3</td>
<td>32</td>
<td>14 15</td>
</tr>
<tr>
<td>5</td>
<td>1 3</td>
<td>1.8</td>
<td>-0.6</td>
<td>33</td>
<td>15 16</td>
</tr>
<tr>
<td>6</td>
<td>2 4</td>
<td>-26.2</td>
<td>-27.3</td>
<td>34</td>
<td>16 13</td>
</tr>
<tr>
<td>7</td>
<td>1 5</td>
<td>-2.5</td>
<td>-2.2</td>
<td>35</td>
<td>13 17</td>
</tr>
<tr>
<td>8</td>
<td>2 6</td>
<td>-8.7</td>
<td>-7.9</td>
<td>36</td>
<td>14 18</td>
</tr>
<tr>
<td>9</td>
<td>3 7</td>
<td>-14.2</td>
<td>-12.6</td>
<td>37</td>
<td>15 19</td>
</tr>
<tr>
<td>10</td>
<td>4 8</td>
<td>-8.7</td>
<td>-7.8</td>
<td>38</td>
<td>16 20</td>
</tr>
<tr>
<td>11</td>
<td>5 6</td>
<td>1.4</td>
<td>1.4</td>
<td>39</td>
<td>13 15</td>
</tr>
<tr>
<td>12</td>
<td>6 7</td>
<td>57.6</td>
<td>66.0</td>
<td>40</td>
<td>14 16</td>
</tr>
<tr>
<td>13</td>
<td>7 8</td>
<td>1.4</td>
<td>1.4</td>
<td>41</td>
<td>1 6</td>
</tr>
<tr>
<td>14</td>
<td>8 5</td>
<td>-54.8</td>
<td>-63.1</td>
<td>42</td>
<td>6 9</td>
</tr>
<tr>
<td>15</td>
<td>5 7</td>
<td>-20.1</td>
<td>-18.1</td>
<td>43</td>
<td>9 14</td>
</tr>
<tr>
<td>16</td>
<td>6 8</td>
<td>22.9</td>
<td>21.0</td>
<td>44</td>
<td>14 17</td>
</tr>
<tr>
<td>17</td>
<td>5 9</td>
<td>-2.4</td>
<td>-2.4</td>
<td>45</td>
<td>3 6</td>
</tr>
<tr>
<td>18</td>
<td>6 10</td>
<td>8.5</td>
<td>7.4</td>
<td>46</td>
<td>6 11</td>
</tr>
<tr>
<td>19</td>
<td>7 11</td>
<td>-14.7</td>
<td>-12.7</td>
<td>47</td>
<td>11 14</td>
</tr>
<tr>
<td>20</td>
<td>8 12</td>
<td>-26.4</td>
<td>-23.5</td>
<td>48</td>
<td>14 19</td>
</tr>
<tr>
<td>21</td>
<td>9 10</td>
<td>0.4</td>
<td>0.4</td>
<td>49</td>
<td>3 8</td>
</tr>
<tr>
<td>22</td>
<td>10 11</td>
<td>0.4</td>
<td>0.3</td>
<td>50</td>
<td>8 11</td>
</tr>
<tr>
<td>23</td>
<td>11 12</td>
<td>0.4</td>
<td>0.3</td>
<td>51</td>
<td>11 16</td>
</tr>
<tr>
<td>24</td>
<td>12 9</td>
<td>0.4</td>
<td>0.3</td>
<td>52</td>
<td>16 19</td>
</tr>
<tr>
<td>25</td>
<td>9 11</td>
<td>7.3</td>
<td>6.1</td>
<td>53</td>
<td>1 8</td>
</tr>
<tr>
<td>26</td>
<td>0 12</td>
<td>-6.4</td>
<td>-5.3</td>
<td>54</td>
<td>8 9</td>
</tr>
<tr>
<td>27</td>
<td>9 13</td>
<td>26.4</td>
<td>23.2</td>
<td>55</td>
<td>9 16</td>
</tr>
<tr>
<td>28</td>
<td>10 4</td>
<td>8.5</td>
<td>7.4</td>
<td>56</td>
<td>16 17</td>
</tr>
</tbody>
</table>

**Case 2**: Results of linear analysis for 56-bar transmission tower space truss.

***Case 3**: Results of non-linear analysis for 56-bar transmission tower space truss.
Table G.13 Displacements obtained for the optimised solution for example 6.6 (56-bar transmission tower space truss) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Joints number</th>
<th>Nodal displacement (mm)</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Direction</strong></td>
<td>Case 2**</td>
<td>Case 3***</td>
<td></td>
</tr>
<tr>
<td></td>
<td>x</td>
<td>y</td>
<td>z</td>
<td>x</td>
</tr>
<tr>
<td>1</td>
<td>28.8</td>
<td>-3.1</td>
<td>1.3</td>
<td>28.6</td>
</tr>
<tr>
<td>2</td>
<td>30.0</td>
<td>-3.1</td>
<td>1.6</td>
<td>30.0</td>
</tr>
<tr>
<td>3</td>
<td>28.8</td>
<td>-3.1</td>
<td>-3.3</td>
<td>28.6</td>
</tr>
<tr>
<td>4</td>
<td>28.8</td>
<td>-3.1</td>
<td>-3.7</td>
<td>28.7</td>
</tr>
<tr>
<td>5</td>
<td>22.6</td>
<td>-2.4</td>
<td>1.4</td>
<td>23.2</td>
</tr>
<tr>
<td>6</td>
<td>20.9</td>
<td>-2.4</td>
<td>1.9</td>
<td>21.3</td>
</tr>
<tr>
<td>7</td>
<td>22.2</td>
<td>-2.9</td>
<td>-2.9</td>
<td>22.8</td>
</tr>
<tr>
<td>8</td>
<td>21.4</td>
<td>-2.9</td>
<td>-3.6</td>
<td>21.8</td>
</tr>
<tr>
<td>9</td>
<td>13.1</td>
<td>-1.8</td>
<td>1.5</td>
<td>13.8</td>
</tr>
<tr>
<td>10</td>
<td>13.2</td>
<td>-1.8</td>
<td>1.7</td>
<td>13.9</td>
</tr>
<tr>
<td>11</td>
<td>13.2</td>
<td>-1.7</td>
<td>-2.5</td>
<td>13.9</td>
</tr>
<tr>
<td>12</td>
<td>13.1</td>
<td>-1.7</td>
<td>-2.6</td>
<td>13.8</td>
</tr>
<tr>
<td>13</td>
<td>7.1</td>
<td>-2.0</td>
<td>0.8</td>
<td>7.8</td>
</tr>
<tr>
<td>14</td>
<td>5.4</td>
<td>-1.9</td>
<td>1.4</td>
<td>6.0</td>
</tr>
<tr>
<td>15</td>
<td>6.6</td>
<td>-2.4</td>
<td>-1.2</td>
<td>7.4</td>
</tr>
<tr>
<td>16</td>
<td>5.9</td>
<td>-2.5</td>
<td>-1.9</td>
<td>6.5</td>
</tr>
<tr>
<td>17</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>18</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>19</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>20</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Case 2**: Results of linear analysis for 56-bar transmission tower space truss.

***Case 3**: Results of non-linear analysis for 56-bar transmission tower space truss.
Table G.14 Stresses obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress MPa</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress MPa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Case2**</td>
<td></td>
<td></td>
<td>Case3***</td>
</tr>
<tr>
<td>1</td>
<td>1 2</td>
<td>-22.3</td>
<td>28</td>
<td>5 20</td>
<td>-30.0</td>
</tr>
<tr>
<td>2</td>
<td>1 3</td>
<td>-9.8</td>
<td>29</td>
<td>6 22</td>
<td>-30.8</td>
</tr>
<tr>
<td>3</td>
<td>1 4</td>
<td>-8.9</td>
<td>30</td>
<td>7 24</td>
<td>-31.4</td>
</tr>
<tr>
<td>4</td>
<td>1 5</td>
<td>-10.3</td>
<td>31</td>
<td>8 26</td>
<td>-31.6</td>
</tr>
<tr>
<td>5</td>
<td>1 6</td>
<td>-11.9</td>
<td>32</td>
<td>9 28</td>
<td>-31.4</td>
</tr>
<tr>
<td>6</td>
<td>1 7</td>
<td>-13.1</td>
<td>33</td>
<td>10 30</td>
<td>-30.8</td>
</tr>
<tr>
<td>7</td>
<td>1 8</td>
<td>-13.5</td>
<td>34</td>
<td>11 32</td>
<td>-30.0</td>
</tr>
<tr>
<td>8</td>
<td>1 9</td>
<td>-13.2</td>
<td>35</td>
<td>12 34</td>
<td>-28.9</td>
</tr>
<tr>
<td>9</td>
<td>1 10</td>
<td>-11.9</td>
<td>36</td>
<td>13 36</td>
<td>-27.6</td>
</tr>
<tr>
<td>10</td>
<td>1 11</td>
<td>-10.4</td>
<td>37</td>
<td>2 15</td>
<td>-5.5</td>
</tr>
<tr>
<td>11</td>
<td>1 12</td>
<td>-8.9</td>
<td>38</td>
<td>15 3</td>
<td>-23.7</td>
</tr>
<tr>
<td>12</td>
<td>1 13</td>
<td>-9.8</td>
<td>39</td>
<td>3 17</td>
<td>-13.9</td>
</tr>
<tr>
<td>13</td>
<td>2 3</td>
<td>-31.3</td>
<td>40</td>
<td>17 4</td>
<td>-15.2</td>
</tr>
<tr>
<td>14</td>
<td>3 4</td>
<td>-32.2</td>
<td>41</td>
<td>4 19</td>
<td>-15.6</td>
</tr>
<tr>
<td>15</td>
<td>4 5</td>
<td>-32.2</td>
<td>42</td>
<td>19 5</td>
<td>-13.5</td>
</tr>
<tr>
<td>16</td>
<td>5 6</td>
<td>-31.9</td>
<td>43</td>
<td>5 21</td>
<td>-15.7</td>
</tr>
<tr>
<td>17</td>
<td>6 7</td>
<td>-31.8</td>
<td>44</td>
<td>21 6</td>
<td>-13.4</td>
</tr>
<tr>
<td>18</td>
<td>7 8</td>
<td>-31.7</td>
<td>45</td>
<td>6 23</td>
<td>-15.4</td>
</tr>
<tr>
<td>19</td>
<td>8 9</td>
<td>-31.7</td>
<td>46</td>
<td>23 7</td>
<td>-13.7</td>
</tr>
<tr>
<td>20</td>
<td>9 10</td>
<td>-31.8</td>
<td>47</td>
<td>7 25</td>
<td>-14.9</td>
</tr>
<tr>
<td>21</td>
<td>10 11</td>
<td>-31.9</td>
<td>48</td>
<td>25 8</td>
<td>-14.2</td>
</tr>
<tr>
<td>22</td>
<td>11 12</td>
<td>-32.2</td>
<td>49</td>
<td>8 27</td>
<td>-14.2</td>
</tr>
<tr>
<td>23</td>
<td>12 3</td>
<td>-32.2</td>
<td>50</td>
<td>27 9</td>
<td>-14.8</td>
</tr>
<tr>
<td>24</td>
<td>13 2</td>
<td>-31.4</td>
<td>51</td>
<td>9 29</td>
<td>-13.7</td>
</tr>
<tr>
<td>25</td>
<td>2 14</td>
<td>-40.4</td>
<td>52</td>
<td>29 10</td>
<td>-15.3</td>
</tr>
<tr>
<td>26</td>
<td>3 16</td>
<td>-27.7</td>
<td>53</td>
<td>10 31</td>
<td>-13.4</td>
</tr>
<tr>
<td>27</td>
<td>4 18</td>
<td>-28.9</td>
<td>54</td>
<td>31 11</td>
<td>-15.7</td>
</tr>
</tbody>
</table>

**Case 2: Results of linear analysis for 120-bar dome space truss.
***Case 3: Results of non-linear analysis for 120-bar dome space truss.
Table G.14 (Cont.) Stresses obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress MPa</th>
<th>Member number</th>
<th>Joints</th>
<th>Member stress MPa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Case2**</td>
<td></td>
<td></td>
<td>Case3***</td>
</tr>
<tr>
<td>55</td>
<td>11 33</td>
<td>-13.5</td>
<td>82</td>
<td>35 36</td>
<td>21.2</td>
</tr>
<tr>
<td>56</td>
<td>33 12</td>
<td>-15.6</td>
<td>83</td>
<td>36 37</td>
<td>21.2</td>
</tr>
<tr>
<td>57</td>
<td>12 35</td>
<td>-15.2</td>
<td>84</td>
<td>37 14</td>
<td>18.6</td>
</tr>
<tr>
<td>58</td>
<td>35 13</td>
<td>-13.9</td>
<td>85</td>
<td>14 38</td>
<td>-90.6</td>
</tr>
<tr>
<td>59</td>
<td>13 37</td>
<td>-23.7</td>
<td>86</td>
<td>16 39</td>
<td>-47.9</td>
</tr>
<tr>
<td>60</td>
<td>37 2</td>
<td>-5.4</td>
<td>87</td>
<td>18 40</td>
<td>-49.5</td>
</tr>
<tr>
<td>61</td>
<td>14 15</td>
<td>18.6</td>
<td>88</td>
<td>20 41</td>
<td>-50.8</td>
</tr>
<tr>
<td>62</td>
<td>15 16</td>
<td>21.4</td>
<td>89</td>
<td>22 42</td>
<td>-51.9</td>
</tr>
<tr>
<td>63</td>
<td>16 17</td>
<td>21.2</td>
<td>90</td>
<td>24 43</td>
<td>-52.6</td>
</tr>
<tr>
<td>64</td>
<td>17 18</td>
<td>22.6</td>
<td>91</td>
<td>26 44</td>
<td>-52.9</td>
</tr>
<tr>
<td>65</td>
<td>18 19</td>
<td>22.6</td>
<td>92</td>
<td>28 45</td>
<td>-52.6</td>
</tr>
<tr>
<td>66</td>
<td>19 20</td>
<td>23.7</td>
<td>93</td>
<td>30 46</td>
<td>-51.9</td>
</tr>
<tr>
<td>67</td>
<td>20 21</td>
<td>23.7</td>
<td>94</td>
<td>32 47</td>
<td>-50.8</td>
</tr>
<tr>
<td>68</td>
<td>21 22</td>
<td>24.6</td>
<td>95</td>
<td>34 48</td>
<td>-49.5</td>
</tr>
<tr>
<td>69</td>
<td>22 23</td>
<td>24.6</td>
<td>96</td>
<td>36 49</td>
<td>-47.9</td>
</tr>
<tr>
<td>70</td>
<td>23 24</td>
<td>25.2</td>
<td>97</td>
<td>15 38</td>
<td>-79.8</td>
</tr>
<tr>
<td>71</td>
<td>24 25</td>
<td>25.2</td>
<td>98</td>
<td>15 39</td>
<td>-106.9</td>
</tr>
<tr>
<td>72</td>
<td>25 26</td>
<td>25.4</td>
<td>99</td>
<td>17 39</td>
<td>-78.8</td>
</tr>
<tr>
<td>73</td>
<td>26 27</td>
<td>25.4</td>
<td>100</td>
<td>17 40</td>
<td>-107.9</td>
</tr>
<tr>
<td>74</td>
<td>27 28</td>
<td>25.4</td>
<td>101</td>
<td>19 40</td>
<td>-79.2</td>
</tr>
<tr>
<td>75</td>
<td>28 29</td>
<td>25.3</td>
<td>102</td>
<td>19 41</td>
<td>-107.5</td>
</tr>
<tr>
<td>76</td>
<td>29 30</td>
<td>24.6</td>
<td>103</td>
<td>21 41</td>
<td>-81.3</td>
</tr>
<tr>
<td>77</td>
<td>30 31</td>
<td>24.6</td>
<td>104</td>
<td>21 42</td>
<td>-105.4</td>
</tr>
<tr>
<td>78</td>
<td>31 32</td>
<td>23.7</td>
<td>105</td>
<td>22 42</td>
<td>-85.1</td>
</tr>
<tr>
<td>79</td>
<td>32 33</td>
<td>23.7</td>
<td>106</td>
<td>23 43</td>
<td>-101.6</td>
</tr>
<tr>
<td>80</td>
<td>33 34</td>
<td>22.6</td>
<td>107</td>
<td>25 43</td>
<td>-90.4</td>
</tr>
<tr>
<td>81</td>
<td>34 35</td>
<td>22.6</td>
<td>108</td>
<td>25 44</td>
<td>-96.3</td>
</tr>
</tbody>
</table>

**Case 2: Results of linear analysis for 120-bar dome space truss.

***Case 3: Results of non-linear analysis for 120-bar dome space truss.
Table G.14 (Cont.) Stresses obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Member number</th>
<th>Joints</th>
<th>Member stress MPa</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Case2**</td>
</tr>
<tr>
<td>109</td>
<td>27</td>
<td>44</td>
</tr>
<tr>
<td>110</td>
<td>27</td>
<td>45</td>
</tr>
<tr>
<td>111</td>
<td>29</td>
<td>45</td>
</tr>
<tr>
<td>112</td>
<td>29</td>
<td>46</td>
</tr>
<tr>
<td>113</td>
<td>31</td>
<td>46</td>
</tr>
<tr>
<td>114</td>
<td>31</td>
<td>47</td>
</tr>
<tr>
<td>115</td>
<td>33</td>
<td>47</td>
</tr>
<tr>
<td>116</td>
<td>33</td>
<td>48</td>
</tr>
<tr>
<td>117</td>
<td>35</td>
<td>48</td>
</tr>
<tr>
<td>118</td>
<td>35</td>
<td>49</td>
</tr>
<tr>
<td>119</td>
<td>37</td>
<td>49</td>
</tr>
<tr>
<td>120</td>
<td>37</td>
<td>38</td>
</tr>
</tbody>
</table>

**Case 2: Results of linear analysis for 120-bar dome space truss.

***Case 3: Results of non-linear analysis for 120-bar dome space truss.
Table G.15 Displacements obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Joints</th>
<th>Nodal displacements (mm)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case2** directions</td>
<td>Case3*** directions</td>
</tr>
<tr>
<td></td>
<td>$x$</td>
<td>$y$</td>
</tr>
<tr>
<td>1</td>
<td>0.0</td>
<td>-0.5</td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>-2.1</td>
</tr>
<tr>
<td>3</td>
<td>-0.4</td>
<td>-1.5</td>
</tr>
<tr>
<td>4</td>
<td>-0.8</td>
<td>-1.0</td>
</tr>
<tr>
<td>5</td>
<td>-1.1</td>
<td>-0.5</td>
</tr>
<tr>
<td>6</td>
<td>-0.9</td>
<td>0.0</td>
</tr>
<tr>
<td>7</td>
<td>-0.5</td>
<td>0.4</td>
</tr>
<tr>
<td>8</td>
<td>0.0</td>
<td>0.6</td>
</tr>
<tr>
<td>9</td>
<td>0.5</td>
<td>0.4</td>
</tr>
<tr>
<td>10</td>
<td>0.9</td>
<td>0.0</td>
</tr>
<tr>
<td>11</td>
<td>1.1</td>
<td>-0.5</td>
</tr>
<tr>
<td>12</td>
<td>0.8</td>
<td>-1.0</td>
</tr>
<tr>
<td>13</td>
<td>0.4</td>
<td>-1.4</td>
</tr>
<tr>
<td>14</td>
<td>0.0</td>
<td>-1.1</td>
</tr>
<tr>
<td>15</td>
<td>0.4</td>
<td>-0.8</td>
</tr>
<tr>
<td>16</td>
<td>1.1</td>
<td>0.7</td>
</tr>
<tr>
<td>17</td>
<td>0.6</td>
<td>-0.5</td>
</tr>
<tr>
<td>18</td>
<td>1.5</td>
<td>0.2</td>
</tr>
<tr>
<td>19</td>
<td>0.3</td>
<td>-0.5</td>
</tr>
<tr>
<td>20</td>
<td>1.6</td>
<td>-0.6</td>
</tr>
<tr>
<td>21</td>
<td>0.3</td>
<td>-0.6</td>
</tr>
<tr>
<td>22</td>
<td>1.3</td>
<td>-1.3</td>
</tr>
<tr>
<td>23</td>
<td>0.3</td>
<td>-0.7</td>
</tr>
<tr>
<td>24</td>
<td>0.8</td>
<td>-1.8</td>
</tr>
<tr>
<td>25</td>
<td>0.1</td>
<td>-0.8</td>
</tr>
</tbody>
</table>

**Case 2: Results of linear analysis for 120-bar dome space truss.**

***Case 3: Results of non-linear analysis for 120-bar dome space truss.
Table G.15 (Cont.) Displacements obtained for the optimised solution for example 6.7 (120-bar dome space truss structure) based on linear and non-linear analysis.

<table>
<thead>
<tr>
<th>Joints</th>
<th>Nodal displacements (mm)</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Case2** directions</td>
<td></td>
<td>Case3*** directions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>x</td>
<td>y</td>
<td>z</td>
<td>x</td>
<td>y</td>
<td>z</td>
</tr>
<tr>
<td>26</td>
<td></td>
<td>0.0</td>
<td>-1.2</td>
<td>0.5</td>
<td>0.0</td>
<td>-2.3</td>
<td>1.5</td>
</tr>
<tr>
<td>27</td>
<td></td>
<td>-0.1</td>
<td>-0.8</td>
<td>-4.5</td>
<td>0.2</td>
<td>-0.1</td>
<td>-6.3</td>
</tr>
<tr>
<td>28</td>
<td></td>
<td>-0.6</td>
<td>-1.8</td>
<td>0.5</td>
<td>-0.9</td>
<td>-2.1</td>
<td>1.4</td>
</tr>
<tr>
<td>29</td>
<td></td>
<td>-0.3</td>
<td>-0.7</td>
<td>-4.6</td>
<td>0.4</td>
<td>-0.2</td>
<td>-6.5</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>-1.3</td>
<td>-1.4</td>
<td>0.3</td>
<td>-1.5</td>
<td>-1.5</td>
<td>1.2</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>-0.3</td>
<td>-0.6</td>
<td>-4.9</td>
<td>0.6</td>
<td>-0.5</td>
<td>-6.8</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>-1.6</td>
<td>-0.6</td>
<td>0.2</td>
<td>-1.8</td>
<td>-0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td>-0.3</td>
<td>-0.5</td>
<td>-5.3</td>
<td>0.6</td>
<td>-0.9</td>
<td>-7.2</td>
</tr>
<tr>
<td>34</td>
<td></td>
<td>-1.5</td>
<td>0.2</td>
<td>-0.1</td>
<td>-1.2</td>
<td>0.1</td>
<td>0.7</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td>-0.2</td>
<td>-0.5</td>
<td>-5.6</td>
<td>0.4</td>
<td>-1.4</td>
<td>-7.7</td>
</tr>
<tr>
<td>36</td>
<td></td>
<td>-1.1</td>
<td>0.7</td>
<td>-0.3</td>
<td>-1.2</td>
<td>0.7</td>
<td>0.4</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>-0.4</td>
<td>-0.8</td>
<td>-6.2</td>
<td>-0.1</td>
<td>-2.2</td>
<td>-8.6</td>
</tr>
<tr>
<td>38</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>39</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>42</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>43</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>44</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>46</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>48</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>49</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Case 2: Results of linear analysis for 120-bar dome space truss.**

***Case 3: Results of non-linear analysis for 120-bar dome space truss.***