

Analysis of Space Frame by Employing the Integrated Force Method

Jaber Almasi

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the Degree of

Master of Science
in
Civil Engineering

Eastern Mediterranean University
January 2013
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Civil Engineering.

Asst. Prof. Dr. Mürüde Çelikağ
Chair, Department of Civil Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Civil Engineering.

Asst. Prof. Dr. Erdinç Soyer
Supervisor

Examining Committee

1. Asst. Prof. Dr. Alireza Rezaei

2. Asst. Prof. Dr. Erdinç Soyer

3. Asst. Prof. Dr. Giray Ozay

ABSTRACT

As newly developed approach, the Integrated Force Method (IFM) is applied as the principal approach for analysis of indeterminate space frames. For automatic generation of equilibrium equations, computer codes are developed and then nodal displacement, internal forces, support reactions and deformations are calculated using the aforementioned equations and compatibility conditions

In the first two programs the member forces of space frames are calculated as primary unknowns. Since obtaining the compatibility condition is one of initial steps in IFM, two codes were developed that address this issue through different approaches which are null space and singular value decomposition.

In third code the displacement method known as Dual Integrated Force Method is applied. Generation of global stiffness matrix is the initial step in Dual Integrated Force Method. In contrast with the previous two methods, in Dual Integrated Force Method displacements are the main unknowns.

Keywords: Integrated force method, Space Frame, Equilibrium Equation

ÖZ

Yeni geliştirilen bir yaklaşım olarak, Entegre Kuvvet Yöntemi statik belirsiz uzay çerçeve analizi için temel yaklaşım olarak uygulanır. Denge denklemleri otomatik elde edebilmesi için, bilgisayar kodları geliştirildi ve daha sonra noktasal yer değiştirme, iç kuvvetleri, ve deformasyonlar söz konusu denklemler ve uygunluk şartları kullanılarak hesaplanır.

İlk iki programlarda uzay çerçeve eleman kuvvetleri hesaplanır. Uyumluluk koşulu elde edebilmek için ilk adımlardan biri bu ana bilinmeyenler olarak konuda singular value decomposition ve null space ayrışımı olan farklı yaklaşımlar geliştirilmiştir.

Üçüncü kod Çift Tümlşik Kuvvet Yöntemi olarak bilinen deplasman yöntemi uygulanır. Küresel rijitlik matrisinin Nesil Çift Entegre Kuvvet Yöntemi ilk adımdır. Daha önceki yöntem ile tezat olarak, yer değiştirme ana bilinmeyenlerdir.

Anahtar kelimeler: Entegre Kuvvet Yöntemi, Uzay çerçeve, Denge denklemleri

TO MY FAMILY

ACKNOWLEDGMENT

I would like to acknowledge and extend my heartfelt gratitude to the following persons who have made the completion of this master thesis possible:

Asst. Prof. Dr. Erdinc Soyer, my dear supervisor, Department of Civil Engineering for the constant reminders and much needed motivation,

All my dear friends specially Changiz Ahbab and Hamed Farajzade for the help and inspiration they extended,

All Eastern Mediterranean University's Civil Engineering faculty members and Staff,

I would like to express my deep gratitude to my family for motivation, guidance and prayer that enabled me to undertake my higher studies and supports to realize my own potential. All the support they have provided me over my study years was the greatest gift anyone has ever given me. I could not have done it without them.

TABLE OF CONTENTS

ABSTRACT	iii
ÖZ.....	iv
DEDICATION	v
ACKNOWLEDGMENT.....	vi
LIST OF TABLES.....	x
LIST OF FIGURES	xi
LIST OF SYMBOLS	xv
LIST OF ABBREVIATIONS	xvii
1 INTRODUCTION	1
1.1 Introduction	1
1.2 The Research Question.....	3
1.3 Research Objectives	4
1.4 Research Limitations.....	4
1.5 Organization of Thesis	5
2 FEATURES OF INTEGRATED FORCE METHOD AND LITERATURE REVIEW	6
2.1 Introduction	6
2.2 Literature Review.....	6
2.2.1 Force Method	6
2.2.2 Integrated Force Method.....	6
2.2.3 Displacement Method.....	7
2.3 Recent IFM Applications in Structural Analysis.....	7

2.4 Features of Space Frame	8
2.4.1 Stability and Determinacy of Space Frame	8
2.4.2 Generation of Transformation Matrix for Space Frame	9
2.4.2.1 Transformation from global to local coordinate system.....	11
2.4.2.1.1 Member Rotation Matrix in terms of Reference Point	14
3 PROBLEM STATEMENT	17
3.1 Introduction	17
3.2 The Problem	17
3.3 Preferred Features of the Soft-wares for IFM and IFMD	17
3.3.1 Features of software package for IFM.....	17
3.3.2 Features of Software package for IFMD	18
3.3.3 Other aspects of analysis packages	18
3.4 An Overview of NS and SVD	19
3.5 Outline of IFMD Method	21
3.6 Mathematica Software as an Instrument	22
4 METHODOLOGY	23
4.1 Introduction	23
4.2 Integrated Force Method (IFM).....	24
4.2.1 Assembling of CC in IFM	25
4.2.2 Null specification of Equations and how to assemble.....	26
4.2.3 Assemble CC in Singular Value Decomposition Method	27
4.3 Dual Integrated Force Method (IFMD).....	28
4.4 Solution Method Outline	29
4.4.1 Equilibrium Equation Application	30

4.4.2 Step by step Computer Programming for IFM and IFMD	31
4.5 Programming	34
4.5.1 Characteristics and advantages of the written program.....	34
5 EQUILIBRIUM EQUATIONS AND MATHEMATICA PROGRAMMING FOR IFM AND IFMD	36
5.1 Introduction	36
5.2 Generation of EE for Space Frame	37
5.2.1 Equilibrium equation assembly:.....	48
5.3 Solution Algorithms	58
5.3.1 IFM via NS	59
5.3.2 IFM via SVD.....	74
5.3.3 IFMD Method	78
6 EXPLANATORY EXAMPLES.....	83
6.1 Introduction	83
6.2 Example 1	84
6.2.1 Results of Mastan program for example 1:.....	90
6.3 Example 2.....	91
6.3.1 Mastan's result of example 2:	100
7 CONCLUSION	103
7.1 Introduction	103
7.2 Contributions	104
7.3 Recommendation for Further Researches	104
REFERENCES	106

LIST OF TABLES

Table 1: Elemental and nodal properties of example 1	85
Table 2: Nodal condition of elements and uniform loads on each member	85
Table 3: Elemental and nodal properties of example 2	93
Table 4: Nodal condition of elements / uniform loads on each member - Example 2	94

LIST OF FIGURES

Figure 1: An arbitrary member m of a space frame.....	9
Figure 2: Member end forces and end displacements in the local coordinate system.....	10
Figure 3: Member end forces and end displacements in the global coordinate system ..	11
Figure 4: Orientation of member local x axis	12
Figure 5: Location of P point in XY plane.....	15
Figure 6: Different Steps of IFM.....	20
Figure 7: Outline of DIFM.....	21
Figure 8: Application of EE	31
Figure 9: Overview of IFM.....	32
Figure 10: Overview of Dual Integrated Force Method Programming	33
Figure 11: Procedure for determinate frame structures	34
Figure 12: Input Data part of Program.....	43
Figure 13: Geometry Data Input Section	43
Figure 14: Space frame with three members.....	44
Figure 15: Calculation of coordination of P-Point	45
Figure 16: Input freedoms of Joints.....	46
Figure 17: Input data part - properties and material	46
Figure 18: Loads data input section.....	47
Figure 19: Computer codes to assemble Equilibrium Equation.....	48
Figure 20: Computer codes to assemble Equilibrium Equation (2)	49
Figure 21: Computer codes to assemble Equilibrium Equation (3)	50

Figure 22: Computer codes to assemble Equilibrium Equation (4)	51
Figure 23: Computer codes to assemble Equilibrium Equation (5)	52
Figure 24: Computer codes to assemble Equilibrium Equation (6)	53
Figure 25: Computer codes to assemble Equilibrium Equation (7)	54
Figure 26: Computer codes to assemble Equilibrium Equation (8)	55
Figure 27: Computer codes to assemble Equilibrium Equation (9)	56
Figure 28: Computer codes to assemble Equilibrium Equation (10)	57
Figure 29: Computer codes to assemble Equilibrium Equation (11)	58
Figure 30: Matrix form of Equilibrium Equations	58
Figure 31: Flexibility and EE matrix plot	59
Figure 32: Coupled EE and CC via NS	59
Figure 33: Algorithm of IFM via NS.....	60
Figure 34: Unconnected flexibility matrix.....	61
Figure 35: Unconnected flexibility matrix (2)	62
Figure 36: Flexibility Matrix.....	63
Figure 37: Matrix plot of flexibility matrix	63
Figure 38: Null Property of Equilibrium Equations Matrix.....	64
Figure 39: Compatibility Condition – Combination of Fig 37 & 38.....	64
Figure 40: Matrix Plot of CC Matrix	64
Figure 41: Coupling of EE with CC and its Matrix Plot	65
Figure 42: Computer Codes for Assembling of Joints Loads.....	66
Figure 43: Loads Matrix of Joints	67
Figure 44: Calculation of Degree of Indeterminacy	67
Figure 45: Computer Codes for Formation of Fixed End Forces.....	68

Figure 46: Computer Codes for Formation of Fixed End Forces (2)	69
Figure 47: Matrix of Fixed End Forces	70
Figure 48: Final Applied Forces.....	70
Figure 49: Finding Matrix of Members independent Forces	71
Figure 50: Calculation of Nodal Displacements Matrix	72
Figure 51: Calculation of Member End Forces	73
Figure 52: Calculation of Member End Forces (continued)	74
Figure 53: Coupled EE and Compatibility Condition via SVD	75
Figure 54: Algorithm of IFM via SVD	76
Figure 55: Codes definition to find SVD.....	77
Figure 56: Assembling of CC by Using SVD and Matrix Plot.....	77
Figure 57: (a) Flexibility matrix plot - (b) Plot of stiffness Matrix.....	78
Figure 58: Algorithm for IFMD	79
Figure 59: Matrix Plot of S Matrix	80
Figure 60: Calculation of Stiffness Matrix for IFMD	80
Figure 61: Matrix Plot of Stiffness Matrix.....	81
Figure 62: Calculation of Nodal Displacements by IFMD	81
Figure 63: Calculation of Member Forces By IFMD	82
Figure 64: Structure of input data.....	83
Figure 65: Scheme of 8 member structure.....	84
Figure 66: Results (1)	86
Figure 67: Results - Member Forces (2).....	87
Figure 68: Results - Member Forces (3).....	88
Figure 69: Results Reporting - Member Forces (4).....	89

Figure 70 : Scheme of a 16 member structure 92

Figure 71: Member Forces – Ex.2 95

Figure 72: Member Forces - Ex.2 - (2)..... 96

Figure 73: Member Forces - Ex.2 - (3)..... 97

Figure 74: Member Forces – Ex.2 - (4) 98

Figure 75: Member Forces – Ex.2 - (5) 99

Figure 76: Member Forces – Ex.2 - (6) 100

LIST OF SYMBOLS

MATRIX QUANTITIES

[A]	Equilibrium Equation
$([A]^T)^{pinv}$	Moore-Penrose Pseudo Inverse of $[A]^T$
[C]	Compatibility Condition
[D]	Direction Cosine
[G]	Unconnected Flexibility Matrix
[I]	Identity Matrix
[J]	Transpose Matrix of $[S]^{-1}$
$[K]_{ifmd}$	Pseudo Stiffness Matrix
[M]	Singular Value Decomposition Matrix
$[M_u]$	Orthogonal Matrix
$[M_v]$	Orthogonal Matrix
$[M_\sigma]$	Diagonal Matrix
[NS]	Null Space Matrix of [A]
[S]	Matrix of Coupling [A] with [C]

NON MATRIX QUANTITIES

A	Cross-Sectional Area
E	Modulus of Elasticity
F	Internal Force

f_s	Flexibility of Element
{F}	Internal Forces Vector
L	Length of Element
l_{ij}	Direction Cosine Along X Axis
m	Number of Elements
m_{ij}	Direction Cosine Along Y Axis
n	Number of Nodes
n_{ij}	Direction Cosine Along Z Axis
{P}	Applied Load Vector
R	Support Reactions
RP	Reference Point
{X}	Displacement Vector
{ β }	Deformations Vector
{ ΔR }	Initial Deformation Vector

LIST OF ABBREVIATIONS

CC	Compatibility Condition
CSM	Classical Stiffness Method
DOF	Degree of Freedom
DDR	Deformation Displacement Relations
EE	Equilibrium Equations
EM	Equation Matrix
FM	Force Method
GCC	Global Compatibility Conditions
IE	Internal Energy
IFM	Integrated Force Method
IFMD	Dual Integrated Force Method
MRM	Member Rotation Matrix
NS	Null Space
RP	Reference Point
SVD	Singular Value Decomposition
UFM	Unconnected Flexibility Matrix
WD	Work Done

Chapter 1

INTRODUCTION

1.1 Introduction

Structure refers to most of observable objects in the environment like plants, trees, skeleton of animals and human, spider nest etc. Most of these structures are developed by either the genetic imprints of the body itself or by instinct. The only creature that conceptualizes, imagines and design structures prior to its construction is human.

Any built object and its constituting parts must be in equilibrium state and stable in order to be accepted as a structure. A structure in order to attain and remain in equilibrium state, all of external and internal forces applied to it should be in balance.

One of the fundamental principles of structural design and analysis is equilibrium equation concept that sets the total internal forces at all nodes and joints equal to external loads. It links these set of forces at global degrees of freedom to reveal the stability state of structure. The manual calculation of these equations is an easy task when dealing with the small frames but in the case of large scale and complex structures, it is a very time consuming and calculation intensive task (Fillipou 2001).

One of the most prevalent types of built structures is the space frame form. The elements of such frames might be oriented in any direction in three-dimensional space and also connected with rigid and/or flexible connections. Moreover external loads oriented towards any random direction could be applied directly on the joints, as well as space frame members. Bending moments around principal axis, shear forces in principal directions, torsion and axial forces are generated as a reaction to external. Analyzing space frame structures is much more difficult than analyzing trusses and beams independently. Introduction of numerous algebraic equations adds to the complexity of this task (Kassimali, 2010).

Two main methods of analyzing the structures are:

- Displacement method
- Force method

Displacement method which is founded by Navire (1785-1836) that uses displacement as primary unknown. Force method developed by Maxwell (1831-1879) uses forces as primary unknown. An alternative and advanced form of force method was also developed by Patnaik et al., termed as Integrated Force Method (IFM). (S.N. Patnaik, L. Berke and R.H. Gallanghar, 1991)

In IFM, through simultaneously coupling of equilibrium equation and compatibility condition (CC), in addition to redundants all of independent forces are calculated (S.N. Patnaik, L. Berke and R.H. Gallanghar, 1991). Along with elimination of need for selecting redundants, the determination of independent internal forces has been reduced to one solution process. Because of aforementioned reasons for designing large scale and complex structures the IFM is a proper option. IFM has many advantages over other

methods. It converges to the correct answer much faster and has a well-conditional system in order to carry out a finite element discrete analysis (FEDA). It also provides much more precise results. The scope of problems that could be addressed via IFM have also been broadened to cover analysis of nonlinear structures (N. R. B. Krishnam Raju, and J. Nagabhushanam, 2000) and optimization (R. Sedaghati, 2005)

Use of Singular Value Decomposition (SVD) and Null Space (NS) has rendered the formation of CC for IFM much easier (Sensoy, 1995) (Maovaghar, 2005). The CC could be directly calculated, when NS and SVD are applied to equilibrium matrix. The CC of IFM is calculated through multiplication of unconnected flexibility matrix with NS or SVD. The NS and SVD are acquired via simple programming Mathematica ver.8. This simplicity is the chief justification for recommendation of NS and SVD when calculating the compatibility matrix and analyzing structures through IFM.

There is an advanced form of IFM called Dual Integrated Force Method (IFMD). In this method only unconnected stiffness matrix and equilibrium matrix of the structure are used in the generation of global stiffness matrix. Hence the development of long and complex codes for generation of global stiffness matrix is avoided. This matrix is also obtained through programming in Mathematica ver.8. This advantage is the reason behind the recommendation of using IFM.

1.2 The Research Question

This research aims to develop a computer code that analyzes space frames through three different methods that are IFM via NS, IFM via SVD and IFMD. These codes are very

user-friendly and also very beneficial for instructors, students, researchers and designers when analyzing the structures. Different problems were analyzed through the mentioned three methods and the outcomes are compared with outcomes of Mastan ver.3. The comparison shows conformity between the results of research and Mastan ver.3 results. In this thesis the theories for these 3 methods is discussed, step by step. The discussion covers the following topics:

- Equilibrium equation;
- Compatibility conditions;
- Unconnected flexibility matrix;
- Nodal displacement;
- Main IFM matrix;
- End force of members.

1.3 Research Objectives

The available soft-wares released for analysis of frames mostly apply stiffness method and classical force method but there is no software for structural analysis through frame with integrated force method. Beside this, many of these soft-wares just generate frame analysis like nodal displacement, forces of member end, support reactions and also diagrams of shear force, axial force, bending moment diagrams. These soft-wares also do not expose the complete steps of these methods.

1.4 Research Limitations

Thermal, triangular, trapezoidal loading and support settlement are not included within this research.

1.5 Organization of Thesis

The basic explanations regarding EE, IFM via NS, IFM via SVD and IFMD are discussed in chapter 2. This chapter also covers the theory of these three methods.

In chapter 3 the significance of the problem and the problem itself is stated. The solution approach and the desired characteristics of the program are discussed as well.

Chapter 4 is about the methodologies used in this research.

Chapter 5 describes how to assemble the equilibrium equation automatically and the Mathematica programming for equilibrium equation.

Chapter 6 contains illustrative examples and their solution thorough three different methods.

Chapter 7 includes a summary and conclusion of the research and recommendation for further researches.

Chapter 2

FEATURES OF INTEGRATED FORCE METHOD AND LITERATURE REVIEW

2.1 Introduction

In this chapter, Basics of space frame structures and their common characteristics and behavior are explained. Furthermore transformation matrix is discussed.

2.2 Literature Review

2.2.1 Force Method

In Civil Engineering structural analysis, the force method (FM) is a popular alternate technique for the classical stiffness method (CSM). It is favored by structural analyzers due to its more accurate and perfect estimates for forces. Patnaik formulated and suggested an innovate formulation in the FM and named it “Integrated Force Method” for continuous and discrete systems analysis.

2.2.2 Integrated Force Method

As a force technique, IFM integrates both EE system and the global compatibility conditions (GCC) all together. With having well- defined parameters, this method could be applied to different types of structures.

2.2.3 Displacement Method

The main idea in displacement method of structural analysis is that the displacements of each joints are taken as unknown, and after computing the displacement, the internal forces are calculated. There are various methods for the displacement method. Among these methods; the most important one is Dual Integrated Method.

2.3 Recent IFM Applications in Structural Analysis

IFM has been developed by Patnaik for analyzing structures. Before him Navier wrote the equilibrium equation for four-leg table, but his method was unsuccessful in solving equations due to indeterminate nature of structure. In Patnaik method, internal forces are supposed to be as independent variables. The next step is integration of system equilibrium equations with the GCC in order to make the governing set of conditions. One of the advantages of this method to the other methods like standard force method is that there is no need for redundant load systems. This property of IFM provides us with more accurate results than the ones generated with CSM. (Patnaik S. N., 1986)

Application of IFM to other structural topics continued by Patnaik to consider the initial deformation behavior. Then Patnaik and friends established structural analyzing of finite elements by applying IFM on structures with two dimensions. In this analysis, space framed structures have not been argued. Nonlinear analyzing of structures by IFM was also investigated by other researchers like Krishnam at 2000 (Krishnam Raju N. R. B. , and Nagabhushanam J. , 2000).

Some other investigations on IFM were done by Civil Engineering Students of Eastern Mediterranean University in recent years. Analysis of two dimensional truss structures

by Saied Khosravi in 2005 (Khosravi S., 2005), two dimensional analysis of frame structures by Seyed Saeed Kamkar in 2010 (Kamkar S., 2010) and three dimensional truss analysis by Hamed Farajzadeh in 2012 (Farajzadeh, 2012) were done.

2.4 Features of Space Frame

The most general style of framed structures is space frames. Members in this type of frames may be connecting to each other in any direction in three-dimensional space. Also members are connected by rigid and/or flexible connections. Additionally, external loads in any random direction can be applied to the joints, and also members of space frames. The members of a space frame are usually exposed to bending moments under the act of the external loads (Kassimali, 2010).

Space frames are regularly used as a multi directional span. They are used to build long spans with few supports (Company, 2012).

2.4.1 Stability and Determinacy of Space Frame

Each of the nodes in a space frame contains intersecting forces and three moment equations. Therefore, six independent force equilibrium equations and moment equations should be written for each node. The formula to identify the circumstance and number of determinacy should be written and obtained from:

$$d = 6 \times m + \text{rest} - 6 \times n \quad \text{Eq.1}$$

Where

d: Number of determinacy

m: Number of members or unknown forces number

rest: number of support's reactions

n: Number of nodes

Conditions:

If $6 \times m + \text{rest} < 6n$, frame is unstable

If $6 \times m + \text{rest} = 6n$, frame is determinate

If $6 \times m + \text{rest} > 6n$, frame is indeterminate

2.4.2 Generation of Transformation Matrix for Space Frame

Generation of transformation matrix for space frame structure is quite different with the transformation matrices for trusses, plane frames, and grids. Transformation matrix for space frames is written according to direction cosines of x , y and z axes of the members in local coordinate system with respect to its global coordinate system of the structure (XYZ). Unlike it, transformation matrix for trusses, plane frames and grids are written with cosines of only the member's x direction or longitudinal axis. Position of a member in a space frame is written based on the angles concerning its local and global axes. Figure 1 is showing an arbitrary member m of a space frame.

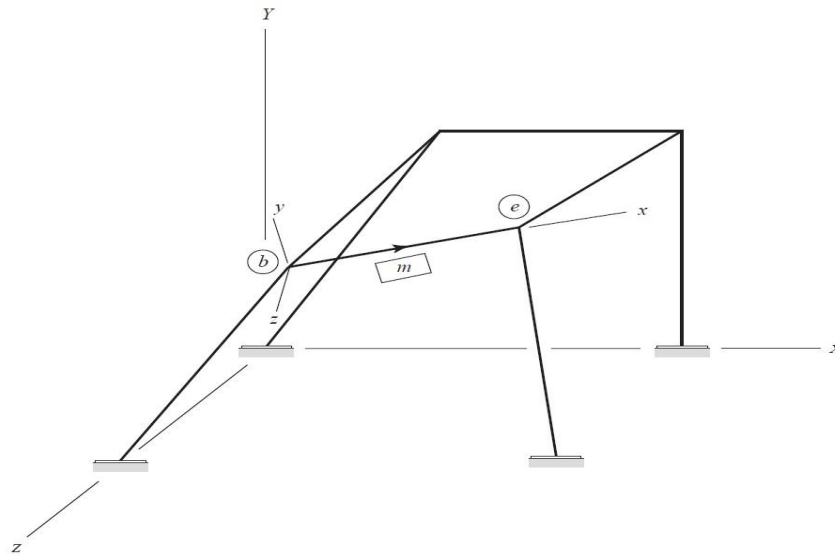


Figure 1: An arbitrary member m of a space frame

In figure 2 member m is considered in a space frame. The member end forces and end displacements in the local coordinate system are shown as \mathbf{Q} and \mathbf{u} . Figure 3 shows the

corresponding system of member end forces \mathbf{F} and end displacements \mathbf{v} , in the global coordinate system. As it is shown in Figure 3, the global member end forces and displacements are named with numbers. It is in a way similar to the local forces and displacements, except that they act in the directions of the global X , Y , and Z axes (Kassimali, 2010).

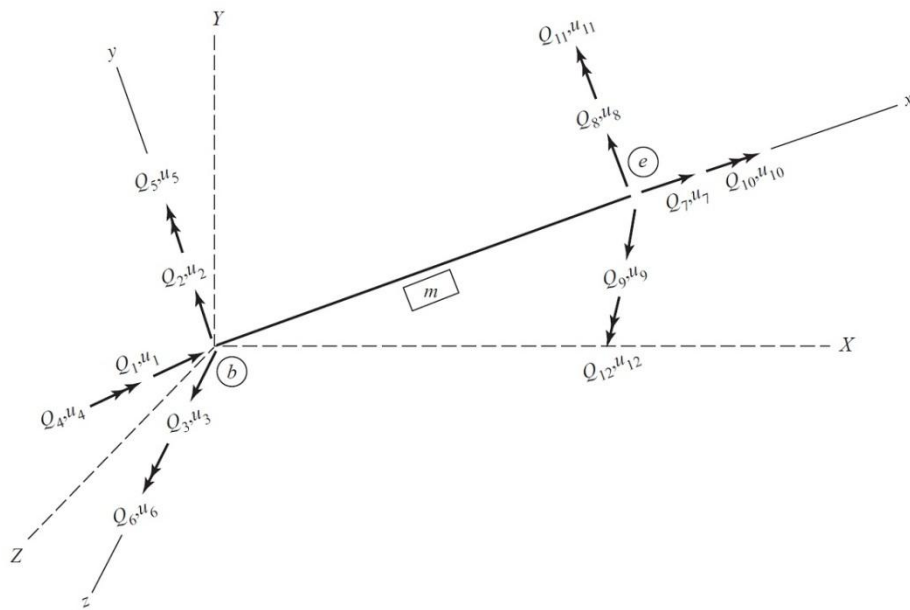


Figure 2: Member end forces and end displacements in the local coordinate system

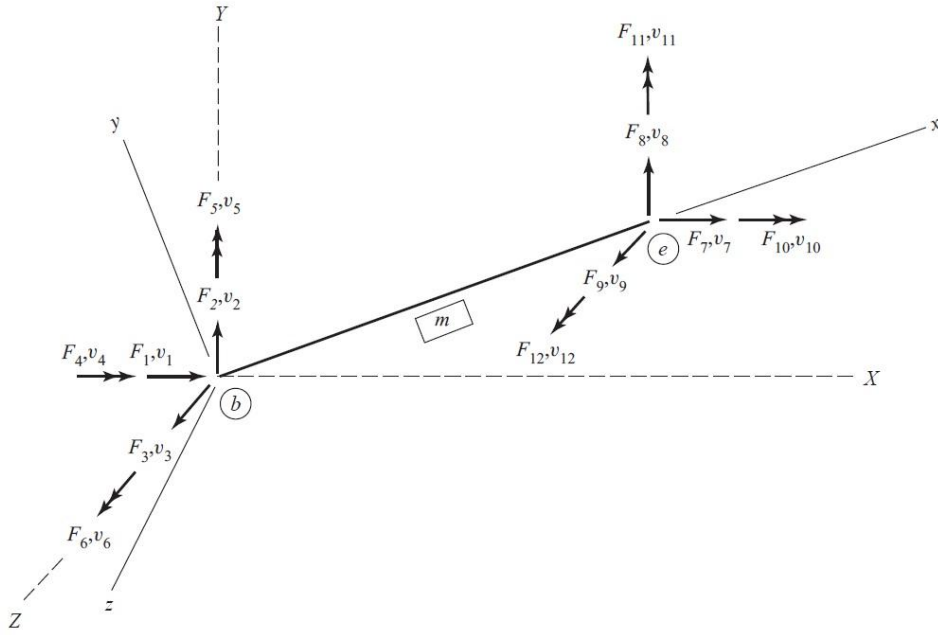


Figure 3: Member end forces and end displacements in the global coordinate system

2.4.2.1 Transformation from global to local coordinate system

It is realized by comparing Figures 2 and 3 that at member end b, the local forces Q_1 , Q_2 , and Q_3 should be as same as the algebraic total quantities of the components of the global forces F_1 , F_2 , and F_3 in the directions of the local x , y , and z axes respectively. Figure 4 is also showing the angle between local and global axes. The angles between the global axes are denoted by $\theta_x X$. It means the angle between local x and global X axes. $\theta_x Y$, and $\theta_x Z$, are written respectively. In the same way, the angles between the

local and global y and z axis are symbolized by θ_yX , θ_yY , θ_yZ , θ_zX , θ_zY , and θ_zZ ,

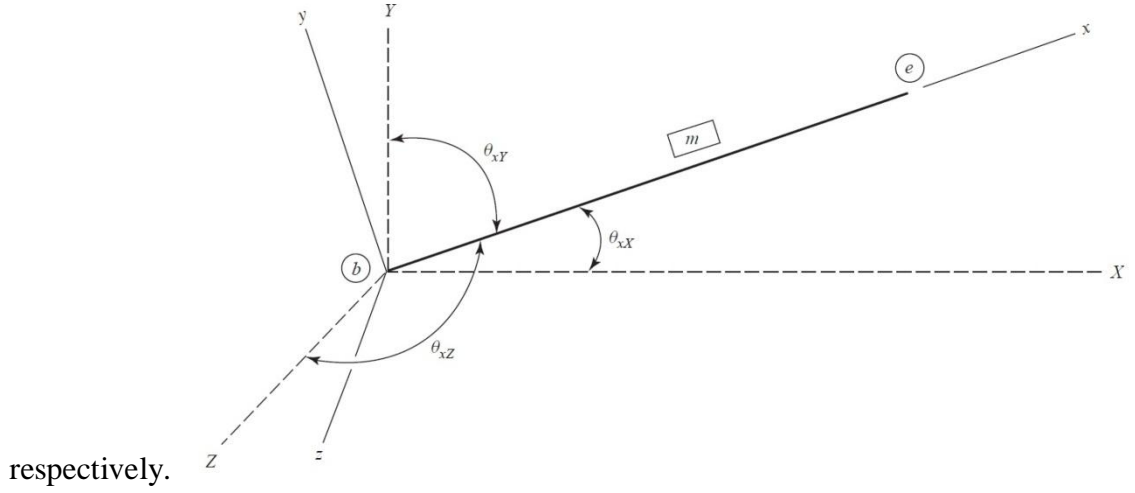


Figure 4: Orientation of member local x axis

$$Q_1 = F_1 \cos \theta_xX + F_2 \cos \theta_xY + F_3 \cos \theta_xZ \quad \text{Eq.2}$$

$$Q_2 = F_1 \cos \theta_yX + F_2 \cos \theta_yY + F_3 \cos \theta_yZ \quad \text{Eq.3}$$

$$Q_3 = F_1 \cos \theta_zX + F_2 \cos \theta_zY + F_3 \cos \theta_zZ \quad \text{Eq.4}$$

Equations (1) can be converted to matrix form as:

$$\begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \end{bmatrix} = \begin{bmatrix} r_{xX} & r_{xY} & r_{xZ} \\ r_{yX} & r_{yY} & r_{yZ} \\ r_{zX} & r_{zY} & r_{zZ} \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} \quad \text{Eq.5}$$

In which

$$r_{ij} = \cos \theta_{ij} \quad i = x, y, \text{ or } z \quad \text{and} \quad j = X, Y, \text{ or } Z$$

The local actions Q_4 , to Q_{12} , at member end b to e, can be written in the same way in terms of their global matching part F_4 to F_{12} .

By merging these equations, the transformation relationship between the 12×1 member local end force vector \mathbf{Q} and the 12×1 member global end force vector \mathbf{F} , can be expressed in the regular formula of:

$$\mathbf{Q} = \mathbf{TF} \quad \text{Eq.6}$$

T: 12×12 transformation matrix on behalf of the members of space frames

The Eq.7 stands for T.

$$[T] = \begin{bmatrix} r_{xX} & r_{xY} & r_{xZ} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ r_{yX} & r_{yY} & r_{yZ} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ r_{zX} & r_{zY} & r_{zZ} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_{xX} & r_{xY} & r_{xZ} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_{yX} & r_{yY} & r_{yZ} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_{zX} & r_{zY} & r_{zZ} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_{xX} & r_{xY} & r_{xZ} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_{yX} & r_{yY} & r_{yZ} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_{zX} & r_{zY} & r_{zZ} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_{xX} & r_{xY} & r_{xZ} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_{yX} & r_{yY} & r_{yZ} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & r_{zX} & r_{zY} & r_{zZ} \end{bmatrix} \quad \text{Eq.7}$$

The compacted form of matrix T is written in terms of its sub-matrices as:

$$[T] = \begin{bmatrix} r & 0 & 0 & 0 \\ 0 & r & 0 & 0 \\ 0 & 0 & r & 0 \\ 0 & 0 & 0 & r \end{bmatrix} \quad \text{Eq.8}$$

In which:

O: is a 3x3 null matrix;

r: is a 3x3 member rotation matrix (Eq.9)

$$[r] = \begin{bmatrix} r_{xX} & r_{xY} & r_{xZ} \\ r_{yX} & r_{yY} & r_{yZ} \\ r_{zX} & r_{zY} & r_{zZ} \end{bmatrix} \quad \text{Eq.9}$$

In analysis of space frames, the rotation matrix r has an essential role. There are some methods to create this matrix. The most advantageous and common method for this

purpose is Member Rotation Matrix (MRM) in Terms of a Reference Point (RP).
(Kassimali, 2010)

2.4.2.1.1 Member Rotation Matrix in terms of Reference Point

Members of space frame structures are typically sloped. If the angle of member is known, it is easy to determine the rotation matrix by Eq.10.

$$r = \begin{bmatrix} r_{xX} & r_{xY} & r_{xZ} \\ \frac{-r_{xX}r_{xY}\cos\Psi - r_{xZ}\sin\Psi}{\sqrt{r_{xX}^2 + r_{xZ}^2}} & \sqrt{r_{xX}^2 + r_{xZ}^2}\cos\Psi & \frac{-r_{xY}r_{xZ}\cos\Psi + r_{xX}\sin\Psi}{\sqrt{r_{xX}^2 + r_{xZ}^2}} \\ \frac{r_{xX}r_{xY}\sin\Psi - r_{xZ}\cos\Psi}{\sqrt{r_{xX}^2 + r_{xZ}^2}} & -\sqrt{r_{xX}^2 + r_{xZ}^2}\sin\Psi & \frac{r_{xY}r_{xZ}\sin\Psi + r_{xX}\cos\Psi}{\sqrt{r_{xX}^2 + r_{xZ}^2}} \end{bmatrix} \quad \text{Eq.10}$$

If the angles of roll are unknown, it can be calculated by inspection. In some structure members, determination for angles of roll cannot be found easily due to their orientations. There are different ways to find angle of roll. One of these methods is P point Method. Considering Figure 5 the member rotation matrix r with the help of a reference point like P, the P equation can be written as:

$$P = (X_p - X_b)I_x + (Y_p - Y_b)I_y + (Z_p - Z_b)I_z \quad \text{Eq.11}$$

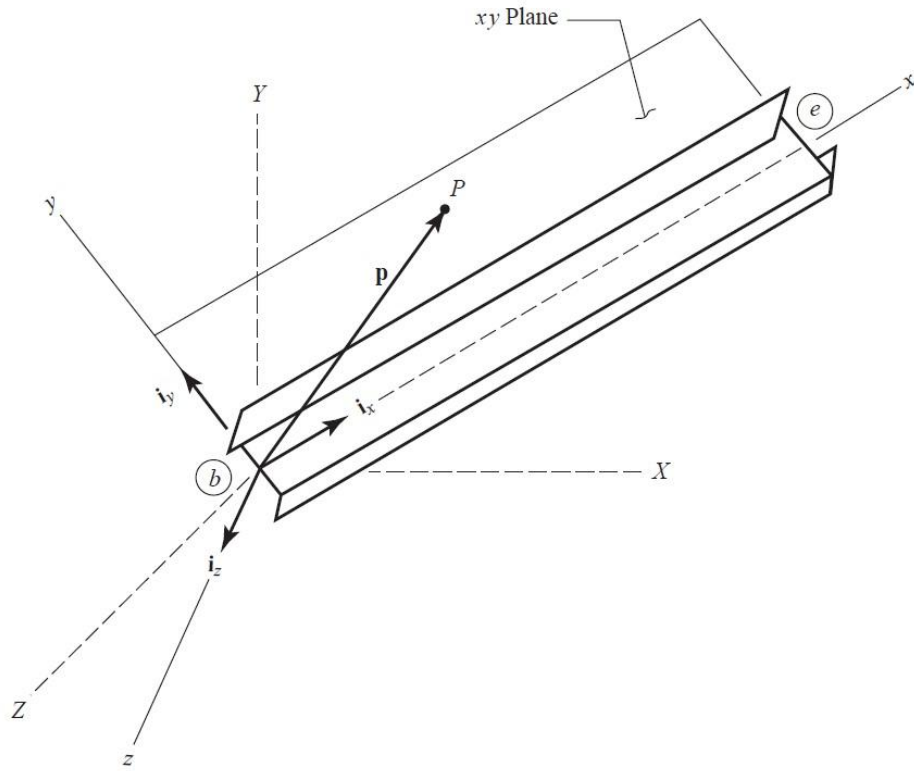


Figure 5: Location of P point in XY plane

I_z and I_y can be determined respectively according to following equations:

$$I_z = \frac{I_x \cdot p}{|I_x \cdot p|} \quad \text{Eq.12}$$

$$I_y = I_z \cdot I_x \quad \text{Eq.13}$$

These formulas are obtained without involving angle of roll (ψ). To determine the relationship between angles of roll and reference point P, first of all it should define the components of the position vector P in the secondary $\overline{x\overline{y}\overline{z}}$ coordinate system. P point should be lying in the XY or XZ plane. Then the relationship equation can be written as:

$$\text{Sin}\psi = \frac{P_z}{\sqrt{p_y^2 + p_z^2}} \quad \text{Eq.14}$$

$$\text{Cos}\psi = \frac{P_y}{\sqrt{p_y^2 + p_z^2}} \quad \text{Eq.15}$$

These equations are used for space frame members which are present in any random directions as well as vertical members. With respect to these formulas,

$$P_x = r_{xy}(Y_P - Y_b) \quad \text{Eq.16}$$

$$P_y = -r_{xy}(X_P - X_b) \quad \text{Eq.17}$$

$$P_z = Z_P - Z_b \quad \text{Eq.18}$$

Position of P will be:

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} 0 & r_{xy} & 0 \\ -r_{xy} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (X_P - X_b) \\ (Y_P - Y_b) \\ (Z_P - Z_b) \end{bmatrix} \quad \text{Eq.19}$$

(Kassimali, 2010)

Chapter 3

PROBLEM STATEMENT

3.1 Introduction

In this chapter the research problems will be discussed. Main goals of this study will also be mentioned and an overview of the solutions to our problems will be discussed.

3.2 The Problem

In this thesis algorithms were developed which will use IFM and the DIFM to allow the analysis of space frames. These algorithms will generate the following results:

1. Independent member forces;
2. End forces of members;
3. Nodal displacements.

The programming process in creating this algorithm is carried out using computer algebra system Mathematica 8.

3.3 Preferred Features of the Soft-wares for IFM and IFMD

In this part, different characteristics of the soft-wares for IFM and IFMD will be discussed.

3.3.1 Features of software package for IFM

In the literature review, in existing documents written by Patnaik the following process is deployed to gain compatibility conditions:

1. Generating relations of the deformation displacement.

2. Eliminating the displacements after the first step.
3. Gain the compatibility conditions. (Patnaik S. N., 1986) (Patnaik S. , 1999)
(Patnaik S. N. and Joseph K. T. , 1986) (Patnaik S. N. , Hopkins D. A. , and Halford G. R. , 2004)

In this research a different numerical method has been followed. After generation of EE, the following two numerical methods have been used:

1. Combining the unconnected flexibility matrix with NS of the equilibrium matrix.
2. Combining the SVD of the equilibrium matrix with unconnected flexibility matrix.

3.3.2 Features of Software package for IFMD

Although in this computer code the emphasis has been put on generating the global stiffness matrix, however in the IFMD the overall stiffness matrix is calculated through using the dimply generated equilibrium matrix and drawing upon matrix management capabilities of Mathematica 8. Utilization of such programming leads to cutting the process of global stiffness matrix calculation to a single programming line. Hence the use of Mathematica 8 is far less time consuming.

3.3.3 Other aspects of analysis packages

1) Effortless

In comparison with current commercial analysis packages, the developed codes are much easier to use because much less options and parameters have to be specified.

2) Easy procedure:

The programs are developed in a way that users in different levels can operate and learn it without any need to read certain documentation or manual.

3) Debugging Variables:

Being suspicious of any results or eager to know how the calculations are performed, users can track all the variables during the calculation procedure to find the source of possible mistakes using the debugging mode.

4) Flexible:

In each level of the calculations the code that executes the process is shown. It is a huge advantage for beginners to learn more about programming techniques. Professional users can edit the code to change its utility.

5) Apparent Theory:

The theory which is used in the methods is elaborated on throughout the programs making it easier to understand and follow the procedures.

6) Educational:

Similar to tutorials, the IFM via NS, IFM via SVD, IFMD and theories are introduced to the user while using the program. At each step it has been tried to provide sufficient tutorials and hints.

7) Accessible:

The packages are available for instructors, students and engineers without any limits, so there is no need for them to search the literature for many hours to find such packages that contain these characteristics.

3.4 An Overview of NS and SVD

The approach for IFM is depicted in Figure 6:

1. Equilibrium equations generation [S].
2. Unconnected flexibility matrix assembly [G].

3. Finding CC, by using the NS and SVD.
4. Solving process for independent member forces.
5. Compute nodal displacements.
6. Finding twelve member end forces.

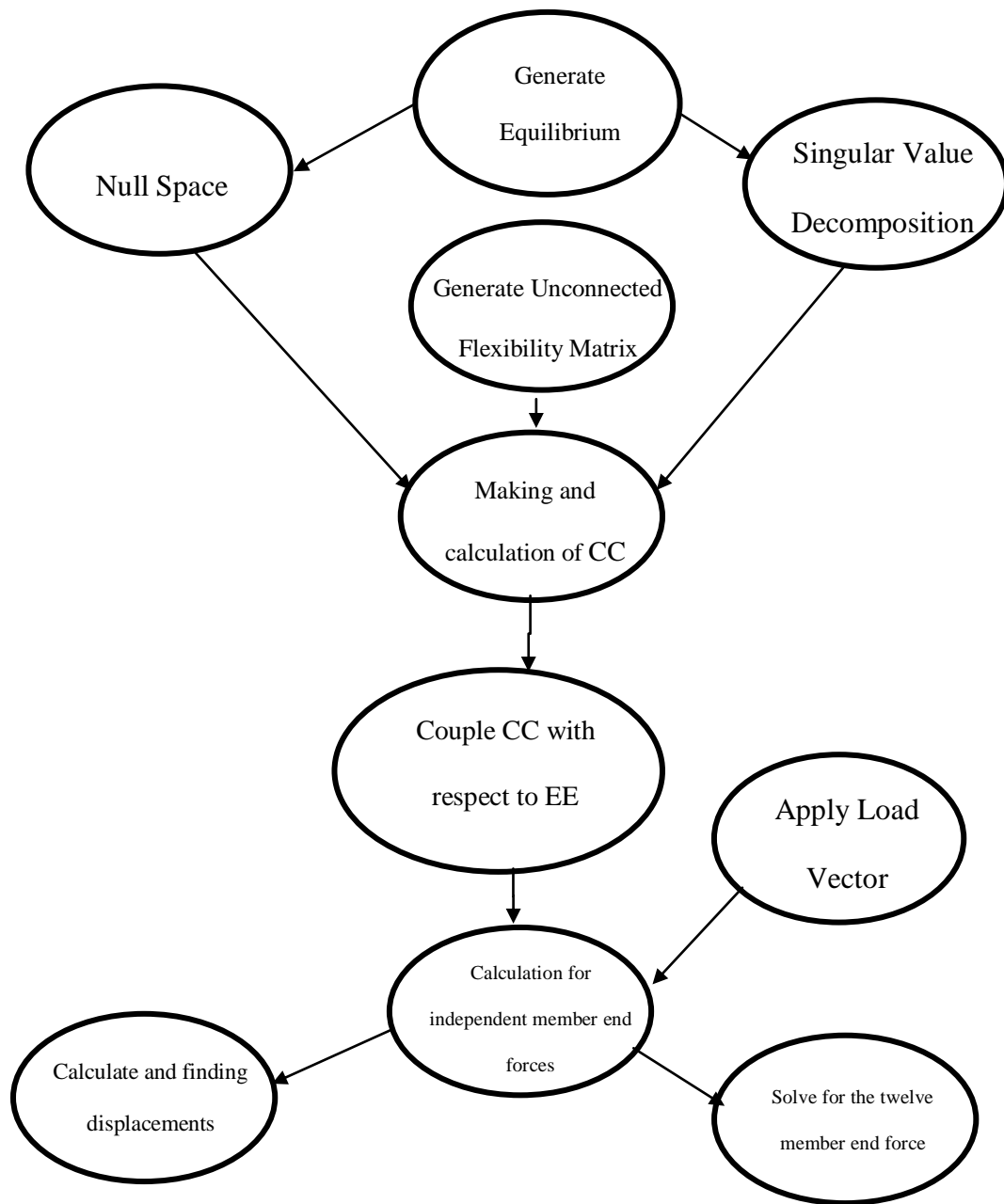


Figure 6: Different Steps of IFM

3.5 Outline of IFMD Method

The approach for IFM is depicted in Figure 7:

1. Equilibrium equations generation [S].
2. Unconnected flexibility matrix assembly [G].
3. Calculation of inverse unconnected flexibility matrix [G]
4. Global Stiffness Matrix generation [K].
5. Solve for nodal displacements [X].
6. Finding forces of Independent member forces.
7. Computation of twelve member end force.

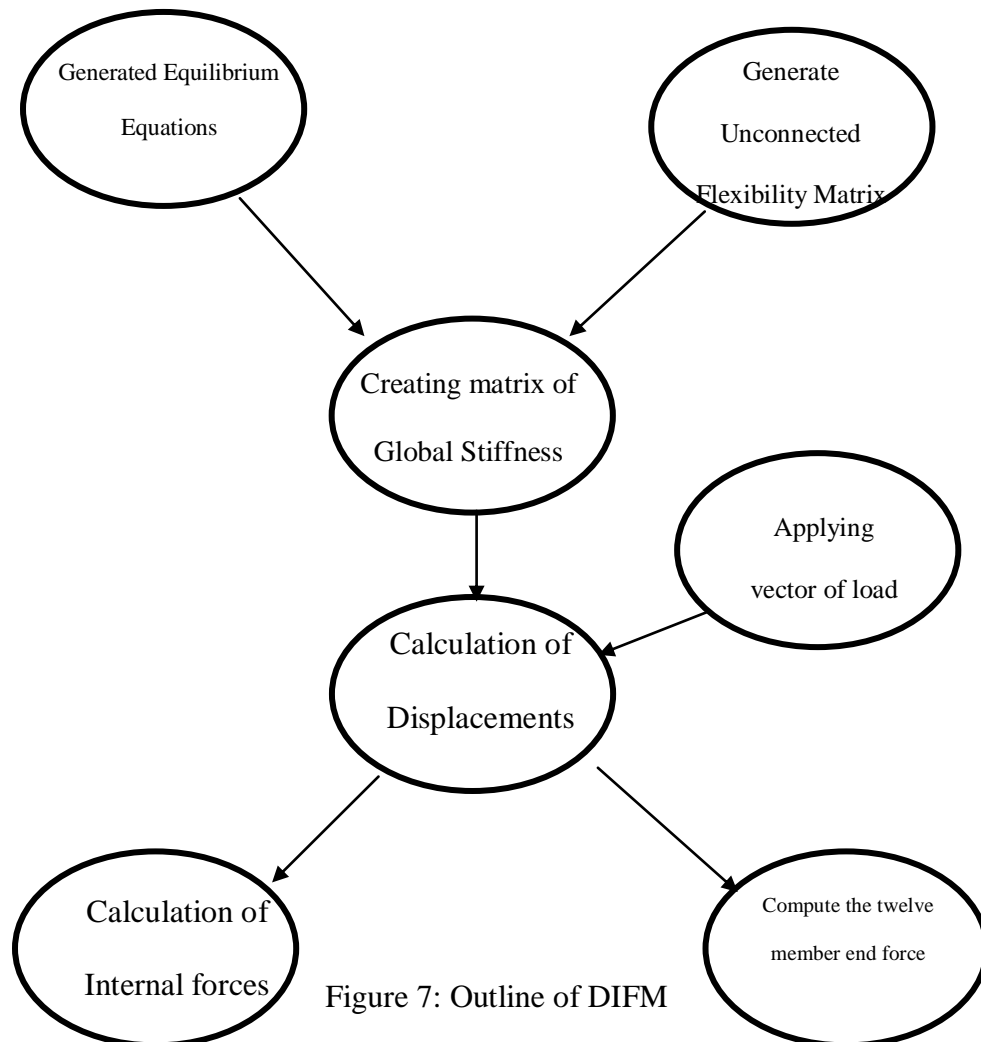


Figure 7: Outline of DIFM

3.6 Mathematica Software as an Instrument

The software used in this research is Mathematica 8 because of capabilities and options that it provides. The chief reasons for choosing Mathematica 8 are as follows:

1. Interactive calculation via notebooks.
2. Easy as much as a calculator.
3. Availability of over a thousand built-in functions.
4. Numerical calculation to any level of accuracy.
5. Possibility of symbolic calculation.
6. Solving equations numerically or symbolically.
7. Vectors and matrices functions.
8. Capability of user-defined functions.

Chapter 4

METHODOLOGY

4.1 Introduction

To analyze any structure, the relation between S, F and P matrices is used. It is done according to Equation 20

$$[S]\{F\} = \{P\} \quad \text{Eq.20}$$

S matrix in determinate and indeterminate structures is different. S matrix for determinate structure is square (m x m) and equation 20 can be solved easily.

The created EE for indeterminate structures is in rectangular shape. It is (m x n) where m is total quantity EEs and n is the number of forces which are unknown. In order to create a square matrix, it is needed to write and generate additional equation named compatibility condition.

In this study two methods are undertaken in order to solve the indeterminate space frame structures through using EE. IFM and Displacement method are these two methods. IFM includes Null Space and Singular Value Decomposition and Displacement method through using Dual IFM.

4.2 Integrated Force Method (IFM)

This method is developed based on the below equation (Eq.21). In this method EE and Compatibility Condition (CC) are merged.

$$\left[\begin{array}{c} \text{Equilibrium Equation} \\ \text{Compatibility condition} \end{array} \right] \{ \text{Forces} \} = \left\{ \begin{array}{c} \text{Mechanical Load} \\ \text{Initial Deformation} \end{array} \right\} \quad \text{Eq.21}$$

This method is being used for analysis of indeterminate space frames based on EE and CC. It has potential to develop for large scale structures even if it has confusing topology. (Patnaik S. N. , and Hopkins D. A. , 1998), (Patnaik S. N. , Hopkins D. A. , and Halford G. R. , 2004)

EE is made based on forces whereas CC is generated depend on deformation and displacement. In order to merge these two equations together, CC should be written in terms of forces. Consequently the equation for IFM will change to:

$$\left[\begin{array}{cc} [A] & \\ [C] & [G] \end{array} \right] \{ F \} = \left\{ \begin{array}{c} P \\ \Delta R \end{array} \right\} \quad \text{Eq.22}$$

In this new equation, A, C, G, F, P, ΔR stand for: EE matrix, CC matrix, unconnected flexibility matrix, internal forces vector, external loads vector and initial deformations vector respectively.

Short form of this equation is

$$[S]\{F\} = \{P^*\} \quad \text{Eq.23}$$

In this equation matrix [S] is created by merging EE, CC and flexibility matrix in square shape. The number of rows in P vector is equal to external loads vector. In special cases where there is no initial deformation, Zeros should be placed in order to make the

equation dimensionally balanced. By using this method, independent member forces can be determined. In some cases it is needed to find displacements of members. In such cases displacements can be calculated based on the Equation 24 (Patnaik S. N. , Hopkins D. A. , and Halford G. R. , 2004)

$$\{X\} = [J] [G] [F] \quad \text{Eq.24}$$

Where: X, G and F are nodal displacements vector, unconnected flexibility matrix and calculated member forces, respectively. Also J is transpose matrix of inversed S in this equation.

$$J = \left[[S]^{-1} \right]^T \quad \text{Eq.25}$$

4.2.1 Assembling of CC in IFM

According to Patnaik (Patnaik S. , 1999) in order to calculate CC, equation should be written under the energy theory in structures. To write deformation displacement relation and according the energy theory:

$$IE = \frac{1}{2} \{F\}^T \{\beta\} \quad \text{Eq.26}$$

With respect to the work-energy rule (IE=W), together with knowing that, in space frame structures deformations $(\beta_1, \beta_2, \dots, \beta_m)$ are corresponding to internal forces (F_1, F_2, \dots, F_m) and also external loads lead to be done work in structure the equation 27 is obtained:

$$W = \frac{1}{2} \{P\}^T \{X\} \quad \text{Eq.27}$$

Therefore:

$$\frac{1}{2}\{F\}^T \{\beta\} = \frac{1}{2}\{P\}^T \{X\} \quad \text{Eq.28}$$

By replacing EE (Eq.27) into Eq.28 it is changed to:

$$\{F\}^T \left(\left\{ \beta - [A]^T \{X\} \right\} \right) = 0 \quad \text{Eq.29}$$

And also this equation 29 can be written as:

$$\{\beta\} = [A]^T \{X\} \quad \text{Eq.30}$$

In this equation deformations should be written in terms of displacements. It is also needed to eliminate displacements from deformation displacement relation. By doing this it obtained as

$$[C]\{\beta\} = \{0\} \quad \text{Eq.31}$$

Which in this equation P is equal to (m-n). it means that the CC has (m-n) columns.

4.2.2 Null specification of Equations and how to assemble

According to Patnaik, to get CC, the null space of EE is used. In the other hand for making CC, the null space of the EE should be merged with unconnected flexibility matrix (UFM). Null property of EE and after it CC can be obtained through using equation Eq.20 and Eq.30. (Patnaik S. , 1999) (Patnaik S. N. , Hopkins D. A. , and Halford G. R. , 2004)

CC can be written in the format of Eq.32, if deformations between Eq.30 and Eq.31 will be removed.

$$[C][A]^T \{X\} = \{0\} \quad \text{Eq.32}$$

Since displacements are subjective and they have not null vector properties in this equation, coefficient can be removed from the equation. Therefore the equation can be written as:

$$[C][A]^T = \{0\} \quad \text{Eq.33}$$

Or

$$[A][C]^T = \{0\} \quad \text{Eq.34}$$

CC is found by null space of EE. After it CC and EE should be merged.

In this study, Mathematica software is being used. By using this software and its defined commands, it is possible to find null space matrix.

4.2.3 Assemble CC in Singular Value Decomposition Method

Employing SVD is another technique to calculate and obtain the CC. According to Patnaik in this method matrix M is generated.

(Patnaik S. N. and Joseph K. T. , 1986), (Patnaik S. , 1999):

$$[M] = \left[[I] - [A]^T \left([A]^T \right)^{pinv} \right] \quad \text{Eq.35}$$

In this equation [I] stands for the identity matrix. In this matrix number of members and number of columns and rows are equal. $[A]^T$ Stands for transpose form of EE and

$\left([A]^T \right)^{pinv}$ is obtained by Eq.36.

$$\left([A]^T \right)^{pinv} = \left([A][A]^T \right)^{-1} [A] \quad \text{Eq. 36}$$

To obtain matrix M, SVM is applied.

$$[M] = [M_u][M_\delta][M_v]^T \quad \text{Eq.37}$$

$[M_u]$: Orthogonal matrix

$[M_v]$: Orthogonal matrix

In these matrices the number of elements is equal to the number of rows and columns.

$[M_\delta]$: Is a square matrix and calculated according to Eq.38.

$$[M_\delta] = \begin{pmatrix} \Lambda & 0 \\ 0 & 0 \end{pmatrix} \quad \text{Eq.38}$$

Also:

$$\Lambda = \text{diag} (\Lambda_1, \Lambda_2, \dots, \Lambda_\rho) \quad \text{Eq.39}$$

Λ_ρ : Degree of indeterminacy and:

$$\Lambda_1 \geq \Lambda_2 \geq \dots \geq \Lambda_\rho \geq 0 \quad \text{Eq.40}$$

At last M matrix and CC can be changed to:

$$[M] = [M_u] \begin{bmatrix} [NS] \\ [0] \end{bmatrix} \quad \text{Eq.41}$$

$$[C] = [NS][G] \quad \text{Eq.42}$$

Where:

[NS]: Null space matrix of EE. (Patnaik S. , 1999), (Patnaik S. N. and Joseph K. T. , 1986)

4.3 Dual Integrated Force Method (IFMD)

Patnaik developed IFM to Dual IFM. In this method basic equation is:

$$[K]_{ifmd} \{X\} = \{P\}_{ifmd} \quad \text{Eq.43}$$

In this equation [K] is Stiffness matrix. The equation for this matrix is:

$$[K]_{ifmd} = [A][G]^{-1} [A]^T \quad \text{Eq.44}$$

In this equation, [A], $[G]^{-1}$, [X] and [P], are EE matrix, inverse of flexibility matrix, vector of displacements and external loads respectively.

In space frame structures flexibility matrix will be obtained from Equations 44 and 45.

$$f_s = \frac{L_i}{E_i A_i} \quad \text{Eq.45}$$

$$[G] = \begin{bmatrix} f_1 & & & 0 \\ & f_2 & & \\ & & \ddots & \\ 0 & & & f_s \end{bmatrix} \quad \text{Eq.46}$$

The main equation for load vector is Eq.47:

$$\{P\}_{ifmd} = \left\{ \{P\} + \left([A][G]^{-1} \{\beta^0\} \right) \right\} \quad \text{Eq.47}$$

Where:

$\{\beta^0\}$ is initial deformation vector

Number of total degree of freedoms is equal to number of rows and columns. In this study initial deformation of supports are not considered and $\{\beta^0\}$ vector is zero consequently. Therefore equation 47 can be written in Equation 48 format:

$$\{P\}_{ifmd} = \{P\} \quad \text{Eq.48}$$

Next step is assembling of $\{K\}_{ifmd}$ matrix and calculating displacements using Equation

48. After it the internal forces can be obtained with:

$$\{F\} = [G]^{-1} [A]^T \{X\} \quad \text{Eq.49}$$

Using EE matrix [A] is common point between IFM and DIFM. Uncommon point between these two methods is that, in IFM method primary unknowns are internal forces whereas in DIFM method primary unknowns are displacements.

4.4 Solution Method Outline

The main equation to analyze any structure is:

$$[A] \{F\} = \{P\} \quad \text{Eq.50}$$

In this part utilizing EE to find the internal forces is discussed. An outline of computer programming process and the algorithms of it are also explained.

4.4.1 Equilibrium Equation Application

EE is used for both determinate and indeterminate structures. Since in determinate frame structures member forces are unknown, writing EE is sufficient to solve. It is because that the number of EE and unknowns are equal. Finding internal forces will help to calculate deformations and displacements.

Unlike determinate frame structures, in indeterminate structures number of EE is not equal and unknowns are not equal to each other due to unequal known and unknowns. To solve this problem, it is needed to add some new relations.

Figure 8 shows the procedure of EE application in detail in this study including determinate and indeterminate frames.

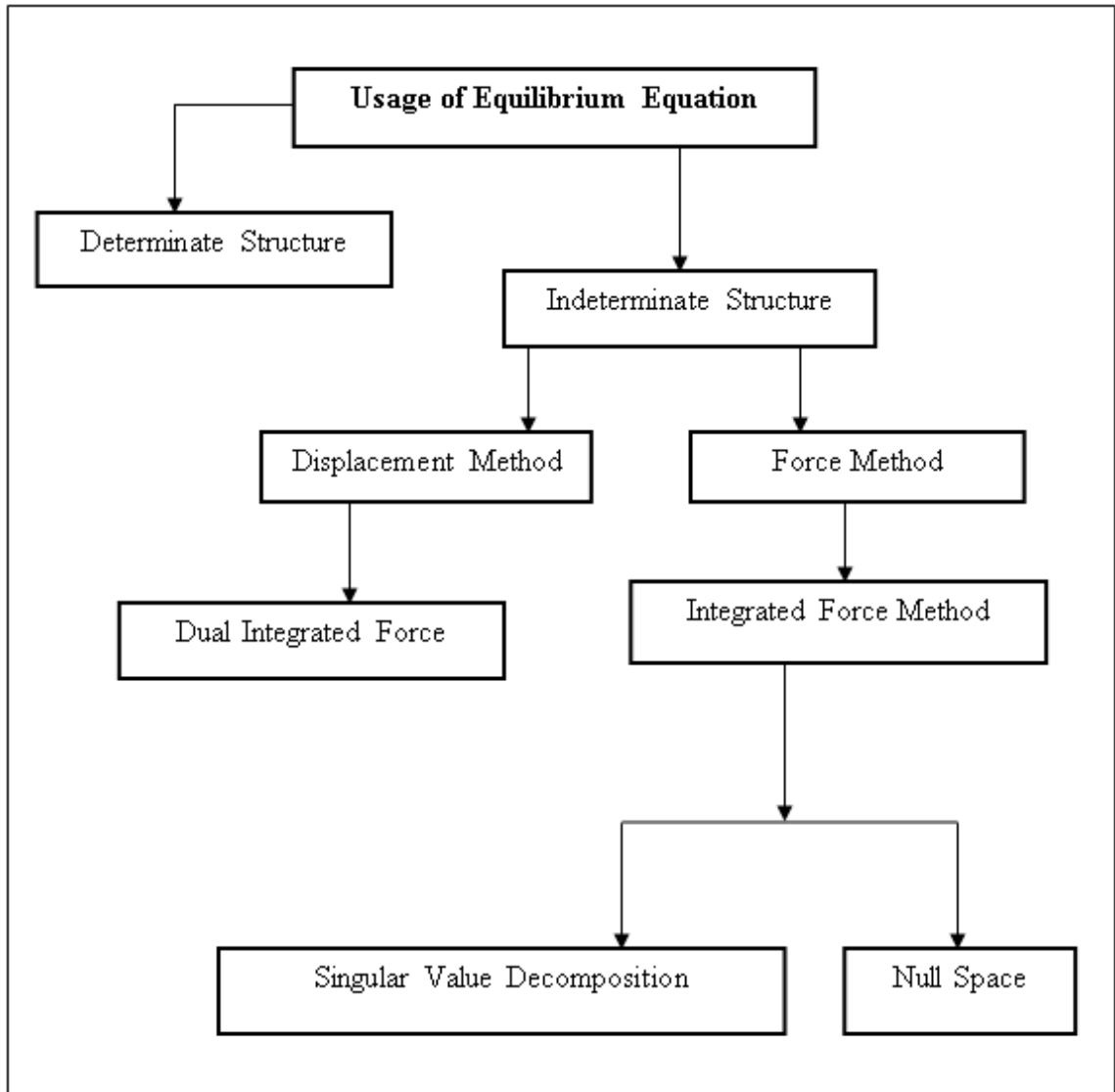


Figure 8: Application of EE

4.4.2 Step by step Computer Programming for IFM and IFMD

Since hand calculation is time consuming and might be inaccurate for large scale space frame structures, it is needed to write codes and computer programs in order to solve this problem. The step by step procedure for IFM and IFMD is explained in this part.

4.4.2.1 IFM step by step

Figure 9 shows the step by step procedure and important stages in IFM method.

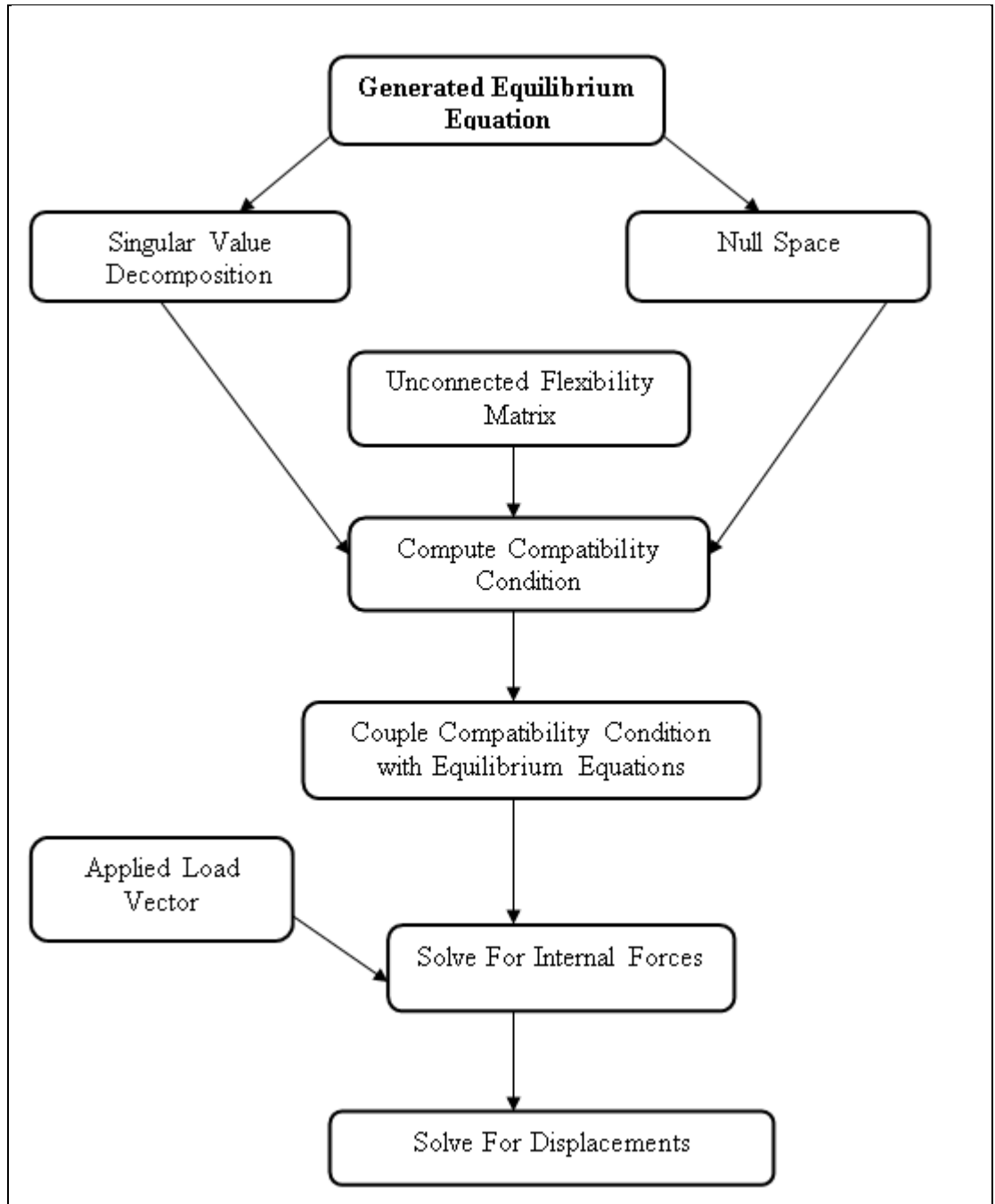


Figure 9: Overview of IFM

4.4.2.2 Procedure in IFMD Method

Figure 10 shows the main steps in IFMD method.

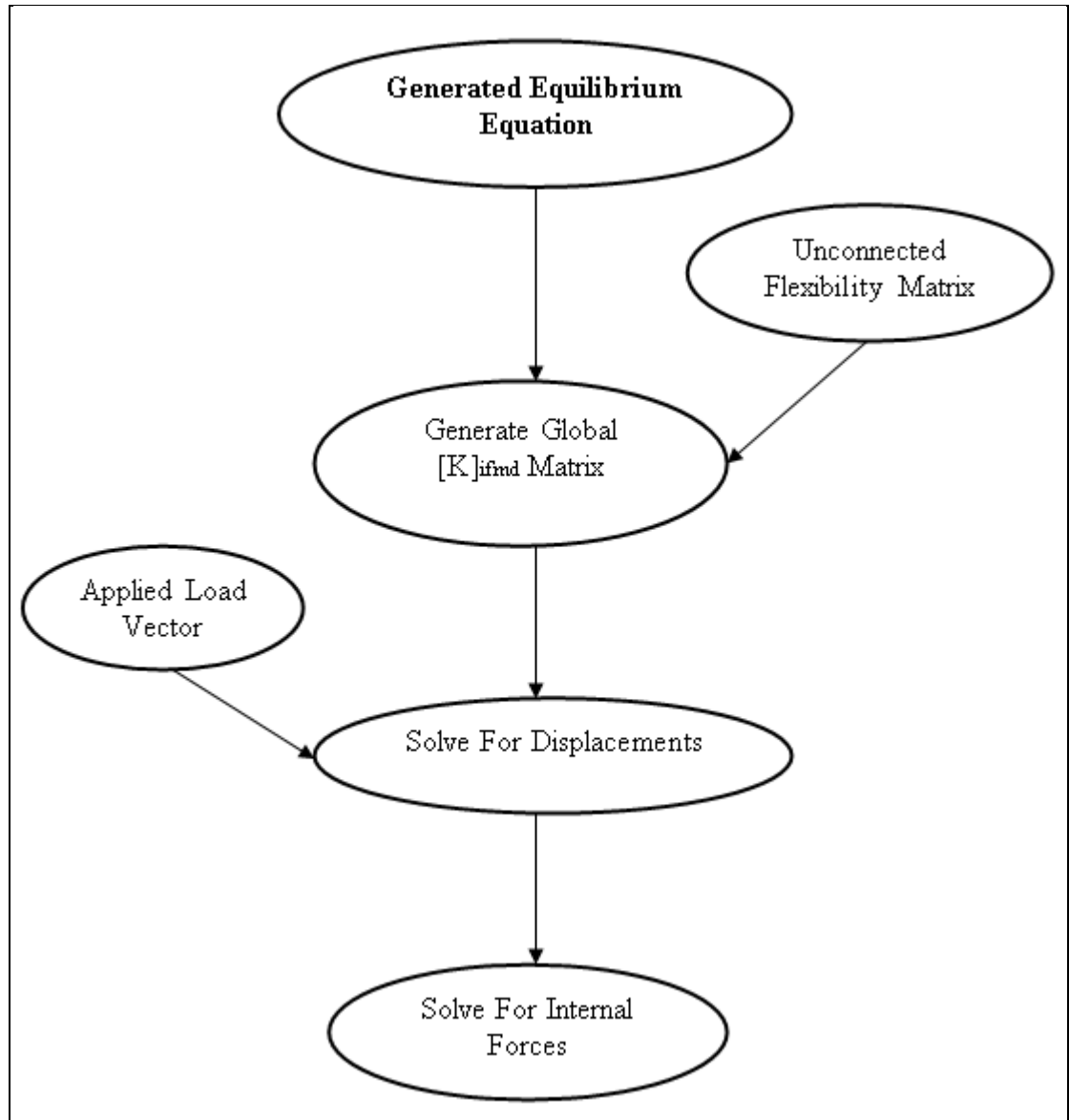


Figure 10: Overview of Dual Integrated Force Method Programming

Figure 9 and 10 are valid for indeterminate structures. In case of indeterminate structures

Figure 11 shows the procedure.

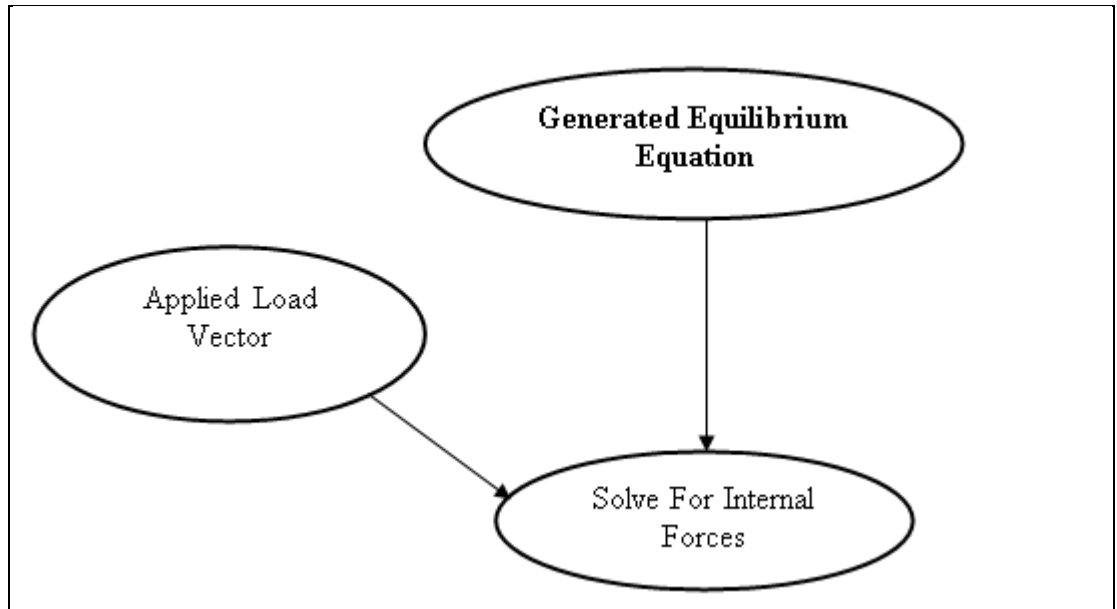


Figure 11: Procedure for determinate frame structures

4.5 Programming

In this study a program application has been written in order to analyze and solve space frame structures. It is written based on the above methods and using matrix decomposition, matrix operations and linear system of equation. Also import and export of data, symbolic and numerical mathematics are drawn upon. The matrix scatter plots are used as well. Also Mathematica software has been used which is a symbolic and numeric computational engine and programming system with the strong ability to other related applications.

4.5.1 Characteristics and advantages of the written program

This program is easy to use, simple and user friendly. It is not necessary to read or learn any instructions for the first time users in order to operate the program. Also it is easy to understand and run analyzing procedure in this program in comparison to other similar applications. Equations, relations and the theory of the methods are used in each level of analyzing giving the programs transparent theory. Furthermore, capability to have a

quick control on the procedure is easy, in order to find any possible and probable mistakes. In each part, results are shown individually. The most advantageous ability of written program is its capability and skill in step by step teaching of the theories and formulation procedure for IFM via null space and IFM via singular value decomposition.

Chapter 5

EQUILIBRIUM EQUATIONS AND MATHEMATICA PROGRAMMING FOR IFM AND IFMD

5.1 Introduction

Since, in the survey carried out for this research no computer code intended for analysis of space frames were based on IFM and also because most of the available codes applied the stiffness method as their principal approach of analysis, the main aim of this research has been decided to be the development of a computer code to analyze the space frames based on IFM.

In this research 3 different packages of computer programs were developed for analysis of space frames. Each package draws on different approaches of integrated force method which were explained in chapter 3. The packages are as follows:

- 1st package: Null Space approach.
- 2nd package: Singular Value Decomposition approach.
- 3rd package: Dual Integrated Force approach.

In this chapter, first process of generating the equilibrium equation is discussed then these equations are used for analysis of space frames by three different methods. In order to introduce and elaborate on the codes, a three member frame was used as an example. Also the method of data entry and use of program were explained.

5.2 Generation of EE for Space Frame

What is going to be explained in this part is the way of developing equilibrium equations matrix of the space frame structure. As we know for each member of the three dimensional frames, there are six actions at each end and the matrix of equilibrium equations of whole structure is a combination of equilibrium equation of each member. Therefore, in this program, the equilibrium equation of each member is made during twelve steps at first and then with a suitable combination of these matrices, we will have the equilibrium equation of the structure at the end. (Fillppou, 2001)

Step 1: At the first step, we should have the location of start point and end point of each member in our structure. For this purpose, the number of each start and end point for all members is read by the program:

$$\text{mincb} = \text{inc}[[i, 2]];$$

$$\text{mince} = \text{inc}[[i, 3]];$$

$$\text{dtabl} =$$

$$\left\{ \begin{array}{l} 6 * \text{mincb} - 5,6 * \text{mincb} - 4,6 * \text{mincb} - 3,6 * \text{mincb} - 2,6 * \text{mincb} - 1,6 * \text{mincb}, \\ \quad \quad \quad 6 * \text{mince} - 5,6 * \text{mince} - 4,6 * \text{mince} - 3, \\ \quad \quad \quad 6 * \text{mince} - 2,6 * \text{mince} - 1,6 * \text{mince} \end{array} \right\};$$

Step 2: The coordinates of these start and end points are read as follows:

$$\text{xd} = \text{cord}[[\text{mince}, 2]] - \text{cord}[[\text{mincb}, 2]];$$

$$\text{yd} = \text{cord}[[\text{mince}, 3]] - \text{cord}[[\text{mincb}, 3]];$$

$$\text{zd} = \text{cord}[[\text{mince}, 4]] - \text{cord}[[\text{mincb}, 4]];$$

Step 3: The program reads the coordinates of P-points which are related to each member of the structure.

$$\text{xPd} = \text{cordP}[[i, 2]] - \text{cord}[[\text{mincb}, 2]];$$

$$yPd = \text{cordP}[[i, 3]] - \text{cord}[[\text{mincb}, 3]];$$

$$zPd = \text{cord}[[i, 4]] - \text{cord}[[\text{mincb}, 4]];$$

Step 4: In the next part of the program length of each member of the structure is calculate as shown below:

$$\text{If} [\text{freet}[[\text{mincb}, 2]] == 1, \text{ReplacePart}[\text{dtabl}, 0, 1]];$$

For complete code refer to Figure 19

$$\text{If} [\text{freet}[[\text{mince}, 7]] == 1, \text{ReplacePart}[\text{dtabl}, 0, 12]];$$

$$Lm = \text{Sqrt}[xd^2 + yd^2 + zd^2];$$

Step 5: After that the direction cosines of each member can be calculated in accordance to the previous amounts like what has shown under this paragraph:

$$rxX = \frac{xd}{Lm}; \quad rxY = \frac{yd}{Lm}; \quad rxZ = \frac{zd}{Lm};$$

Step 6: After all these calculations the fixed unit vector of each member can be found in according to the above values and calculations and also the coordinate of P-point as:

$$uvx = \{rxX, rxY, rxZ\};$$

$$zL = \text{Sqrt} \left[vz[[1]]^2 + vz[[2]]^2 + vz[[3]]^2 \right];$$

$$uvz = \frac{vz}{zL};$$

For further codes refer to Figure 20

$$uvy = \text{Cross}[uvz, uvx];$$

Step 7: In this step of the program we have to generate the global member equilibrium matrix but before that, generation of member equilibrium matrix and transformation

matrix is needed to be obtained. To achieve this purpose we generate the member equilibrium matrix as:

$$b = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & Lm & 0 & -1 & 0 \\ 0 & -Lm & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$

And the transformation matrix as:

$$t = \begin{pmatrix} tv1 & tv2 & tv3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ tv4 & tv5 & tv6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ tv7 & tv8 & tv9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv1 & tv2 & tv3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv4 & tv5 & tv6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv7 & tv8 & tv9 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv1 & tv2 & tv3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv4 & tv5 & tv6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv7 & tv8 & tv9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv1 & tv2 & tv3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv4 & tv5 & tv6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv7 & tv8 & tv9 \end{pmatrix};$$

In which there are some fixed values that can be found as:

$$tv1 = uvx[[1]]; \quad tv2 = uvx[[2]]; \quad tv3 = uvx[[3]]; \quad tv4 = uvy[[1]]; \quad tv5 = uvy[[2]]; \quad tv6 = uvy[[3]]; \quad tv7 = uvz[[1]]; \quad tv8 = uvz[[2]]; \quad tv9 = uvz[[3]];$$

Step 8: By using the member equilibrium matrix, b, and transformation matrix, t, global member equilibrium matrix is generated as:

$$bg = \text{Transpose}[t].b;$$

After all these the generated matrices have to be stored in the memory that this will be done by the following part of the program:

AppendTo[bgmem, bg];

AppendTo[bmem, b];

AppendTo[Lmem, Lm];

AppendTo[dtablmem, dtabl];

AppendTo[tmem, t];

Step 9: By using the values calculated in the previous part of the program, fixed-end actions of each member of the structure can be found as:

$$Q_f = \begin{pmatrix} \frac{\omega[[i]] * Lm}{2.} \\ z \\ z \\ \frac{\omega[[i]] * Lm^2}{12.} \\ z \\ \frac{\omega[[i]] * Lm}{2.} \\ z \\ z \\ z \\ -\frac{\omega[[i]] * Lm^2}{12.} \end{pmatrix};$$

Step 10: By using this vector as a generating vector, the global fixed end actions matrix can be generated:

Ff = Transpose[t].Qf;

Then, both the global fixed-end action and member fixed-end action matrices should be stored in the memory with the following commands:

AppendTo[Ffmem, Ff];

AppendTo[Qfmem, Qf];

Step 11: After generating these mentioned matrices and vectors, this program determines the unrestrained degrees of freedom (D.O.F.) by the commands that are written bellow:

kk = 0;

Do[

cj1 = 6 * i - 5; cj2 = 6 * i - 4; cj3 = 6 * i - 3;

cj4 = 6 * i - 2; cj5 = 6 * i - 1; cj6 = 6 * i;

Print[i, " ", {cj1, cj2, cj3, cj4, cj5, cj6}];

If[freet[[i, 2]] == 1, dof[[6 * i - 5]] = 0];

If[freet[[i, 2]] == 0, kk = kk + 1];

If[freet[[i, 2]] == 0, dof[[6 * i - 5]] = kk];

Further codes are available in Figure 23 and Figure 24

If[freet[[i, 7]] == 1, dof[[6 * i]] = 0];

If[freet[[i, 7]] == 0, kk = kk + 1];

If[freet[[i, 7]] == 0, dof[[6 * i]] = kk]

, {i, 1, noden}]

Step 12: Continuing, in the program we start with empty equilibrium matrix as given below:

S = Table[0., {sr, 1, 6 * noden - rest}, {sc, 1, 6 * m}];

Then, in a loop of the number of members, some variables of EE are filled. The process of this filling is related to the situation of members and their equilibrium equation

matrices. The beneath variables will be filled if there are any restrained degrees of freedom:

```
node1 = inc[[i, 2]];
```

```
node2 = inc[[i, 3]];
```

```
k1 = 6 * node1 - 5;
```

Further codes are shown in Figure 24

```
kc12 = dof[[k12]];
```

Combination step: After these variable fillings, the program begins to store the global equilibrium equation matrix values of each member in the structure equilibrium equation matrix as shown in the following part:

```
If [kc1 ≠ 0, S[[kc1, c1]] = S[[kc1, c1]] + bgmem[[i]][[1,1]]];
```

Further codes are shown in Figures 25,26,27,28 and 29

```
If [kc12 ≠ 0, S[[kc12, c6]] = S[[kc12, c6]] + bgmem[[i]][[12,6]]];
```

After this section is completed, generation of equilibrium equation matrix of the structure will be completed.

To show the complete process of developing equilibrium equations in this program and all other steps of analyzing the space frame with 3 different methods are explained via an example of a three member structure.

The first part of program is the data input phase.

GENERAL DATA INPUT

GIVE NUMBER OF ELEMENTS

`m = 3;`

GIVE NUMBER OF NODES

`noden = 4;`

Figure 12: Input Data part of Program

In Figure 12, number of structure members and nodes are indicated by m and n respectively. In this part number of members and nodes are manually entered.

Geometry data input part:

GEOMETRY DATA INPUT

GIVE MEMBERS INCIDENCE

$$\mathbf{inc} = \begin{pmatrix} 1 & 1 & 3 \\ 2 & 2 & 3 \\ 3 & 3 & 4 \end{pmatrix};$$

GIVE COORDINATE OF JOINTS

$$\mathbf{cord} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & -5. & 4. & 0 \\ 3 & 0 & 4. & 0 \\ 4 & 0 & 4. & -5. \end{pmatrix};$$

Figure 13: Geometry Data Input Section

In next step information regarding the number of elements and their connectivity indicating geometry of structures are manually entered by the user. The number of row and columns of the matrix varies, depending on the number of members of the structure.

1st column indicates the number of elements, 2nd column indicates number of starting-nodes and 3rd columns stand for the number of end-nodes.

The coordinates of nodes must be written in the “cord” matrix as shown in Figure 13. The 1st column shows number of each node. Coordinates of X, Y and Z are shown in remaining columns respectively.

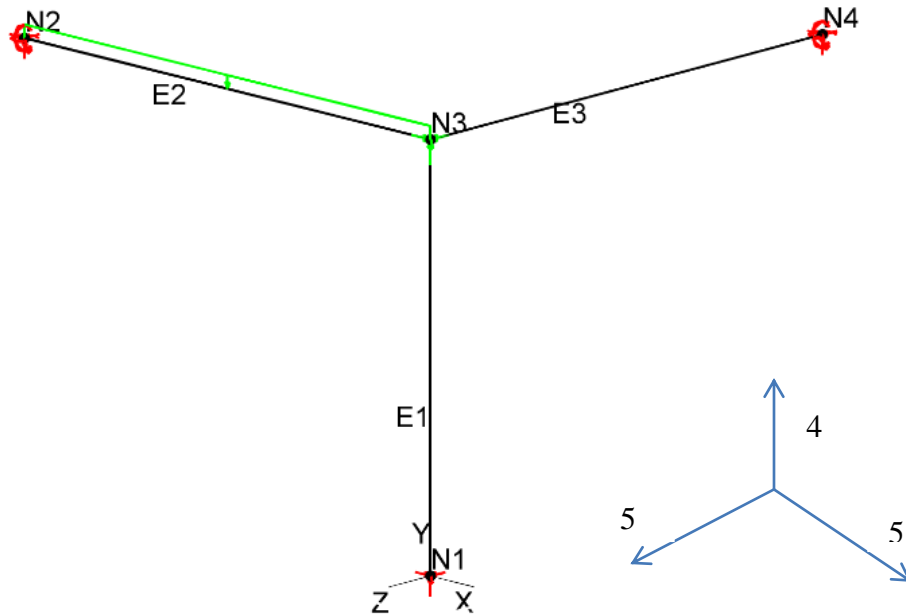


Figure 14: Space frame with three members

Calculation of P-point for each member of the structure:

GIVE COORDINATE OF POINT P

```

const = 1;
cordP = Table[
{
  counter
,
  If[
    (cord[[inc[[counter, 2], 2]] == cord[[inc[[counter, 3], 2]])
    ,
    cord[[inc[[counter, 2], 2]]
    ,
     $\frac{1}{2} \left( \text{cord}[[\text{inc}[[\text{counter}, 2], 2]] + \text{cord}[[\text{inc}[[\text{counter}, 3], 2]] \right)$ 
  ]
,
   $\frac{1}{2} \left( \text{cord}[[\text{inc}[[\text{counter}, 2], 3]] + \text{cord}[[\text{inc}[[\text{counter}, 3], 3]] \right) -$ 
  (-1)Length[inc] const
,
  If[
    (cord[[inc[[counter, 2], 2]] == cord[[inc[[counter, 3], 2]]) &
    (cord[[inc[[counter, 2], 4]] == cord[[inc[[counter, 3], 4]])
    ,
     $\frac{1}{2} \left( \text{cord}[[\text{inc}[[\text{counter}, 2], 4]] + \text{cord}[[\text{inc}[[\text{counter}, 3], 4]] \right) -$ 
    (-1)Length[inc] const
    ,
     $\frac{1}{2} \left( \text{cord}[[\text{inc}[[\text{counter}, 2], 4]] + \text{cord}[[\text{inc}[[\text{counter}, 3], 4]] \right)$ 
  ]
  }
,
  {counter, 1, m}
];
Grid[cordP, Dividers -> Center]

```

1	0	3.	1
2	-2.5	5.	0
3	0	5.	-2.5

Figure 15: Calculation of coordination of P-Point

Released degrees of freedom at each end of the members:



Figure 16: Input freedoms of Joints

Freedom degrees of the frame are represented in an nx7 matrix. In this matrix the joint numbers should be entered in first column and freedom degrees in X, Y and Z directions should be entered in the 2nd, 3rd and 4th columns. The remaining columns stand for moment restraint of nodes in X, Y and Z direction. As shown in the Figure 16, if there is a restraint 1 should be entered and in the case of lack of restraint 0 must be entered.

Properties of sections and materials:

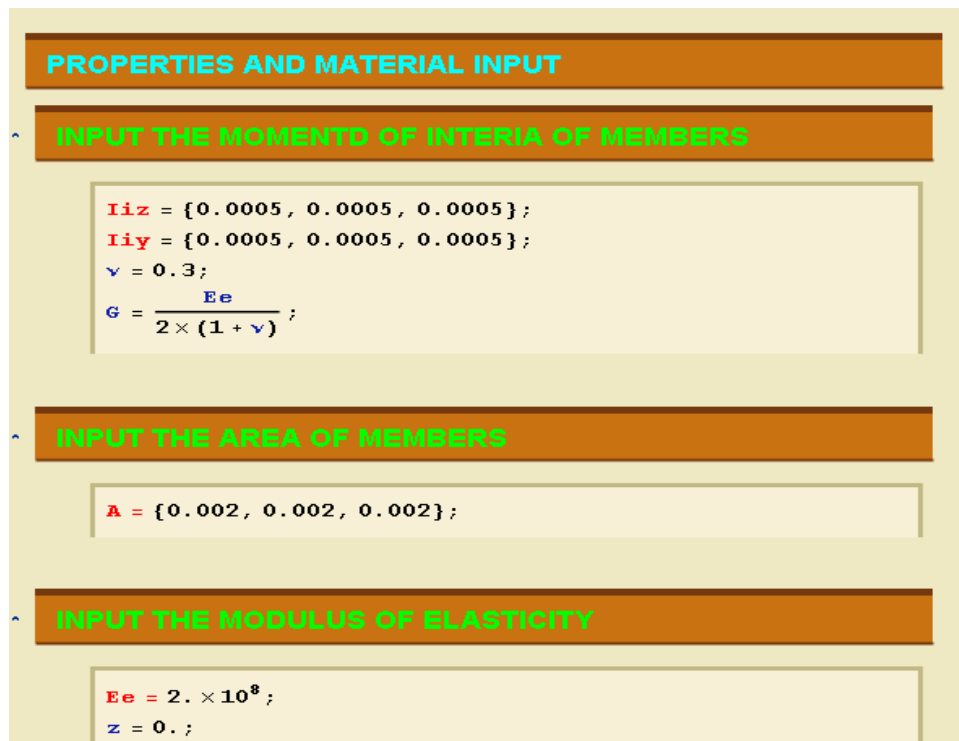


Figure 17: Input data part - properties and material

Loads data input part:

The screenshot shows a software interface with a title bar 'LOADS DATA INPUT'. Below it, a green header reads 'GIVE FORCE APPLIED AT THE JOINTS'. A table is displayed with the following data:

1	0.	0.	0.	0.	0.	0.
2	0.	0.	0.	0.	0.	0.
3	50.	60.	70.	0.	0.	0.
4	0.	0.	0.	0.	0.	0.

The table is labeled 'applfrcs ='. Below the table, another green header reads 'GIVE THE FIXED END FORCE'. A vector is shown as $\phi = \{0., 12., 0.\};$.

Figure 18: Loads data input section

As depicted in Figure 18, the applied loads at each node should be inserted in applied forces matrix. The amounts of loads have to be inserted in the relevant cell and if there were no forces, zero must be inserted.

5.2.1 Equilibrium equation assembly:

```
EQUILIBRIUM EQUATIONS

COMPUTER CODES

kmem = {};
Lmem = {};
bgmem = {};
bmem = {};
tmem = {};
Qfmem = {};
Ffmem = {};
dtablmem = {};
Off[General::spell]
Do[

    mincb = inc[[i, 2]];
    mince = inc[[i, 3]];
    dtabl = {6*mincb - 5, 6*mincb - 4, 6*mincb - 3,
            6*mincb - 2, 6*mincb - 1, 6*mincb,
            6*mince - 5, 6*mince - 4, 6*mince - 3,
            6*mince - 2, 6*mince - 1, 6*mince};
    xd = cord[[mince, 2]] - cord[[mincb, 2]];
    yd = cord[[mince, 3]] - cord[[mincb, 3]];
    zd = cord[[mince, 4]] - cord[[mincb, 4]];
    xPd = cordP[[i, 2]] - cord[[mincb, 2]];
    yPd = cordP[[i, 3]] - cord[[mincb, 3]];
    zPd = cordP[[i, 4]] - cord[[mincb, 4]];
    If[freet[[mincb, 2]] == 1,
        ReplacePart[dtabl, 0, 1]];
    If[freet[[mincb, 2]] == 1,
        Print[i, "  YES", dtabl[[1]]]];
    If[freet[[mincb, 3]] == 1, ReplacePart[dtabl, 0, 2]];
    If[freet[[mincb, 4]] == 1, ReplacePart[dtabl, 0, 3]];
    If[freet[[mincb, 5]] == 1, ReplacePart[dtabl, 0, 4]];
    If[freet[[mincb, 6]] == 1, ReplacePart[dtabl, 0, 5]];
    If[freet[[mincb, 7]] == 1, ReplacePart[dtabl, 0, 6]];
    If[freet[[mince, 2]] == 1, ReplacePart[dtabl, 0, 7]];


```

Figure 19: Computer codes to assemble Equilibrium Equation

```

If[freet[[mince, 3]] == 1, ReplacePart[dtabl, 0, 8]];
If[freet[[mince, 4]] == 1, ReplacePart[dtabl, 0, 9]];
If[freet[[mince, 5]] == 1,
  ReplacePart[dtabl, 0, 10]];
If[freet[[mince, 6]] == 1,
  ReplacePart[dtabl, 0, 11]];
If[freet[[mince, 7]] == 1,
  ReplacePart[dtabl, 0, 12]];
Lm = Sqrt[xd2 + yd2 + zd2];
rxX =  $\frac{xd}{Lm}$ ;
rxY =  $\frac{yd}{Lm}$ ;
rxZ =  $\frac{zd}{Lm}$ ;
uvx = {rxX, rxY, rxZ};
pvec = {xPd, yPd, zPd};
vz = Cross[uvx, pvec];
zL = Sqrt[vz[[1]]2 + vz[[2]]2 + vz[[3]]2];
uvz =  $\frac{vz}{zL}$ ;
uvy = Cross[uvz, uvx];
Print["uvx  =", i, " ", uvx];
Print["pvec  =", i, " ", pvec];
Print["uvz  =", i, " ", uvz];
Print["uvy  =", i, " ", uvy];
tv1 = uvx[[1]];
tv2 = uvx[[2]];
tv3 = uvx[[3]];
tv4 = uvy[[1]];
tv5 = uvy[[2]];
tv6 = uvy[[3]];
tv7 = uvz[[1]];
tv8 = uvz[[2]];
tv9 = uvz[[3]];

```

Figure 20: Computer codes to assemble Equilibrium Equation (2)

$$b = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & Lm & 0 & -1 & 0 \\ 0 & -Lm & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix};$$

t =

$$\begin{pmatrix} tv1 & tv2 & tv3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ tv4 & tv5 & tv6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ tv7 & tv8 & tv9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv1 & tv2 & tv3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv4 & tv5 & tv6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & tv7 & tv8 & tv9 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv1 & tv2 & tv3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv4 & tv5 & tv6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & tv7 & tv8 & tv9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv1 & tv2 & tv3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv4 & tv5 & tv6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & tv7 & tv8 & tv9 \end{pmatrix};$$

```

bg = Transpose[t].b;
AppendTo[bgmem, bg];
AppendTo[bmem, b];
AppendTo[Lmem, Lm];
AppendTo[dtablmem, dtabl];
AppendTo[tmem, t];

```

Figure 21: Computer codes to assemble Equilibrium Equation (3)

$$Qf = \begin{pmatrix} Z \\ \frac{\omega[[i]]+Lm}{2.} \\ Z \\ Z \\ Z \\ \frac{\omega[[i]]+Lm^2}{12.} \\ Z \\ \frac{\omega[[i]]+Lm}{2.} \\ Z \\ Z \\ Z \\ -\frac{\omega[[i]]+Lm^2}{12.} \end{pmatrix};$$

```
Ff = Transpose[t].Qf;
```

```
AppendTo[Ffmem, Ff];
```

```
AppendTo[Qfmem, Qf];
```

```
Print["transformation mat = ", i, " ", MatrixForm[t]];
```

```
Print["cosines = ", i, " ", {CCx, SSy, CCz}];
```

```
Print["bg = ", i, " ", MatrixForm[bg]];
```

```
Print["dtabl = ", i, " ", MatrixForm[dtabl]];
```

```
Print[mincb, " --- ", mince],
```

```
{i, 1, m}];
```

Figure 22: Computer codes to assemble Equilibrium Equation (4)

```

rest = 0;
Do[If[freet[[i, 2]] == 1, rest = rest + 1];
  If[freet[[i, 3]] == 1, rest = rest + 1];
  If[freet[[i, 4]] == 1, rest = rest + 1];
  If[freet[[i, 5]] == 1, rest = rest + 1];
  If[freet[[i, 6]] == 1, rest = rest + 1];
  If[freet[[i, 7]] == 1, rest = rest + 1],

  {i, 1, noden}]
rest
dof = Table[jj, {jj, 1, 6 * noden}]
Clear[kk]
kk = 0;
Do[
  cj1 = 6 * i - 5;
  cj2 = 6 * i - 4;
  cj3 = 6 * i - 3;
  cj4 = 6 * i - 2;
  cj5 = 6 * i - 1;
  cj6 = 6 * i;
  Print[i, " ", {cj1, cj2, cj3, cj4, cj5, cj6}];
  If[freet[[i, 2]] == 1, dof[[6 * i - 5]] = 0];
  If[freet[[i, 2]] == 0, kk = kk + 1];
  If[freet[[i, 2]] == 0, dof[[6 * i - 5]] = kk];
  If[freet[[i, 3]] == 1, dof[[6 * i - 4]] = 0];
  If[freet[[i, 3]] == 0, kk = kk + 1];
  If[freet[[i, 3]] == 0, dof[[6 * i - 4]] = kk];
  If[freet[[i, 4]] == 1, dof[[6 * i - 3]] = 0];
  If[freet[[i, 4]] == 0, kk = kk + 1];
  If[freet[[i, 4]] == 0, dof[[6 * i - 3]] = kk];
  If[freet[[i, 5]] == 1, dof[[6 * i - 2]] = 0];
  If[freet[[i, 5]] == 0, kk = kk + 1];
  If[freet[[i, 5]] == 0, dof[[6 * i - 2]] = kk];
  If[freet[[i, 6]] == 1, dof[[6 * i - 1]] = 0];
  If[freet[[i, 6]] == 0, kk = kk + 1];
  If[freet[[i, 6]] == 0, dof[[6 * i - 1]] = kk];

```

Figure 23: Computer codes to assemble Equilibrium Equation (5)

```

If[freet[[i, 7]] == 1, dof[[6 * i]] = 0];
If[freet[[i, 7]] == 0, kk = kk + 1];
If[freet[[i, 7]] == 0, dof[[6 * i]] = kk],
{i, 1, noden}]
dof
S = Table[0., {sr, 1, 6 * noden - rest}, {sc, 1, 6 * m}];
Dimensions[S]
MatrixQ[S]
MatrixForm[S];
mi = 0;
Do[
  node1 = inc[[i, 2]];
  node2 = inc[[i, 3]];
  k1 = 6 * node1 - 5;
  k2 = 6 * node1 - 4;
  k3 = 6 * node1 - 3;
  k4 = 6 * node1 - 2;
  k5 = 6 * node1 - 1;
  k6 = 6 * node1;
  mi = mi + 1;
  c1 = 6 * mi - 5;
  c2 = 6 * mi - 4;
  c3 = 6 * mi - 3;
  c4 = 6 * mi - 2;
  c5 = 6 * mi - 1;
  c6 = 6 * mi;
  Print[i, "k1---k6", {k1, k2, k3, k4, k5, k6}];
  k7 = 6 * node2 - 5;
  k8 = 6 * node2 - 4;
  k9 = 6 * node2 - 3;
  k10 = 6 * node2 - 2;
  k11 = 6 * node2 - 1;
  k12 = 6 * node2;
  Print[i, "k7---k12", {k7, k8, k9, k10, k11, k12}];
  kc1 = dof[[k1]];
  kc2 = dof[[k2]];
  kc3 = dof[[k3]];
  kc4 = dof[[k4]];
  kc5 = dof[[k5]];

```

Figure 24: Computer codes to assemble Equilibrium Equation (6)

```

kc6 = dof[[k6]];
Print[i, ".kc.", {kc1, kc2, kc3, kc4, kc5, kc6}];
kc7 = dof[[k7]];
kc8 = dof[[k8]];
kc9 = dof[[k9]];
kc10 = dof[[k10]];
kc11 = dof[[k11]];
kc12 = dof[[k12]];
Print[i, ".kc.", {kc7, kc8, kc9, kc10, kc11, kc12}];
Print[i, "^", bgmem[[i]][[1, 1]], ":",
      bgmem[[i]][[1, 2]], ":", bgmem[[i]][[1, 3]]];
If[kc1 ≠ 0,
  S[[kc1, c1]] = S[[kc1, c1]] + bgmem[[i]][[1, 1]];
If[kc1 ≠ 0,
  S[[kc1, c2]] = S[[kc1, c2]] + bgmem[[i]][[1, 2]];
If[kc1 ≠ 0,
  S[[kc1, c3]] = S[[kc1, c3]] + bgmem[[i]][[1, 3]];
If[kc1 ≠ 0,
  S[[kc1, c4]] = S[[kc1, c4]] + bgmem[[i]][[1, 4]];
If[kc1 ≠ 0,
  S[[kc1, c5]] = S[[kc1, c5]] + bgmem[[i]][[1, 5]];
If[kc1 ≠ 0,
  S[[kc1, c6]] = S[[kc1, c6]] + bgmem[[i]][[1, 6]];
If[kc2 ≠ 0,
  S[[kc2, c1]] = S[[kc2, c1]] + bgmem[[i]][[2, 1]];
If[kc2 ≠ 0,
  S[[kc2, c2]] = S[[kc2, c2]] + bgmem[[i]][[2, 2]];
If[kc2 ≠ 0,
  S[[kc2, c3]] = S[[kc2, c3]] + bgmem[[i]][[2, 3]];
If[kc2 ≠ 0,
  S[[kc2, c4]] = S[[kc2, c4]] + bgmem[[i]][[2, 4]];
If[kc2 ≠ 0,
  S[[kc2, c5]] = S[[kc2, c5]] + bgmem[[i]][[2, 5]];
If[kc2 ≠ 0,
  S[[kc2, c6]] = S[[kc2, c6]] + bgmem[[i]][[2, 6]];

```

Figure 25: Computer codes to assemble Equilibrium Equation (7)

```

If[kc3 ≠ 0,
  S[[kc3, c1]] = S[[kc3, c1]] + bgmem[[i]][[3, 1]];
If[kc3 ≠ 0,
  S[[kc3, c2]] = S[[kc3, c2]] + bgmem[[i]][[3, 2]];
If[kc3 ≠ 0,
  S[[kc3, c3]] = S[[kc3, c3]] + bgmem[[i]][[3, 3]];
If[kc3 ≠ 0,
  S[[kc3, c4]] = S[[kc3, c4]] + bgmem[[i]][[3, 4]];
If[kc3 ≠ 0,
  S[[kc3, c5]] = S[[kc3, c5]] + bgmem[[i]][[3, 5]];
If[kc3 ≠ 0,
  S[[kc3, c6]] = S[[kc3, c6]] + bgmem[[i]][[3, 6]];
If[kc4 ≠ 0,
  S[[kc4, c1]] = S[[kc4, c1]] + bgmem[[i]][[4, 1]];
If[kc4 ≠ 0,
  S[[kc4, c2]] = S[[kc4, c2]] + bgmem[[i]][[4, 2]];
If[kc4 ≠ 0,
  S[[kc4, c3]] = S[[kc4, c3]] + bgmem[[i]][[4, 3]];
If[kc4 ≠ 0,
  S[[kc4, c4]] = S[[kc4, c4]] + bgmem[[i]][[4, 4]];
If[kc4 ≠ 0,
  S[[kc4, c5]] = S[[kc4, c5]] + bgmem[[i]][[4, 5]];
If[kc4 ≠ 0,
  S[[kc4, c6]] = S[[kc4, c6]] + bgmem[[i]][[4, 6]];
If[kc5 ≠ 0,
  S[[kc5, c1]] = S[[kc5, c1]] + bgmem[[i]][[5, 1]];
If[kc5 ≠ 0,
  S[[kc5, c2]] = S[[kc5, c2]] + bgmem[[i]][[5, 2]];
If[kc5 ≠ 0,
  S[[kc5, c3]] = S[[kc5, c3]] + bgmem[[i]][[5, 3]];
If[kc5 ≠ 0,
  S[[kc5, c4]] = S[[kc5, c4]] + bgmem[[i]][[5, 4]];
If[kc5 ≠ 0,
  S[[kc5, c5]] = S[[kc5, c5]] + bgmem[[i]][[5, 5]];
If[kc5 ≠ 0,
  S[[kc5, c6]] = S[[kc5, c6]] + bgmem[[i]][[5, 6]];

```

Figure 26: Computer codes to assemble Equilibrium Equation (8)


```

If[kc6 ≠ 0,
  S[[kc6, c1]] = S[[kc6, c1]] + bgmem[[i]][[6, 1]]];
If[kc6 ≠ 0,
  S[[kc6, c2]] = S[[kc6, c2]] + bgmem[[i]][[6, 2]]];
If[kc6 ≠ 0,
  S[[kc6, c3]] = S[[kc6, c3]] + bgmem[[i]][[6, 3]]];
If[kc6 ≠ 0,
  S[[kc6, c4]] = S[[kc6, c4]] + bgmem[[i]][[6, 4]]];
If[kc6 ≠ 0,
  S[[kc6, c5]] = S[[kc6, c5]] + bgmem[[i]][[6, 5]]];
If[kc6 ≠ 0,
  S[[kc6, c6]] = S[[kc6, c6]] + bgmem[[i]][[6, 6]]];
If[kc7 ≠ 0,
  S[[kc7, c1]] = S[[kc7, c1]] + bgmem[[i]][[7, 1]]];
If[kc7 ≠ 0,
  S[[kc7, c2]] = S[[kc7, c2]] + bgmem[[i]][[7, 2]]];
If[kc7 ≠ 0,
  S[[kc7, c3]] = S[[kc7, c3]] + bgmem[[i]][[7, 3]]];
If[kc7 ≠ 0,
  S[[kc7, c4]] = S[[kc7, c4]] + bgmem[[i]][[7, 4]]];
If[kc7 ≠ 0,
  S[[kc7, c5]] = S[[kc7, c5]] + bgmem[[i]][[7, 5]]];
If[kc7 ≠ 0,
  S[[kc7, c6]] = S[[kc7, c6]] + bgmem[[i]][[7, 6]]];
If[kc8 ≠ 0,
  S[[kc8, c1]] = S[[kc8, c1]] + bgmem[[i]][[8, 1]]];
If[kc8 ≠ 0,
  S[[kc8, c2]] = S[[kc8, c2]] + bgmem[[i]][[8, 2]]];
If[kc8 ≠ 0,
  S[[kc8, c3]] = S[[kc8, c3]] + bgmem[[i]][[8, 3]]];
If[kc8 ≠ 0,
  S[[kc8, c4]] = S[[kc8, c4]] + bgmem[[i]][[8, 4]]];
If[kc8 ≠ 0,
  S[[kc8, c5]] = S[[kc8, c5]] + bgmem[[i]][[8, 5]]];
If[kc8 ≠ 0,
  S[[kc8, c6]] = S[[kc8, c6]] + bgmem[[i]][[8, 6]]];

```

Figure 27: Computer codes to assemble Equilibrium Equation (9)

```

If[kc9 ≠ 0,
  S[[kc9, c1]] = S[[kc9, c1]] + bgmem[[i]][[9, 1]]];
If[kc9 ≠ 0,
  S[[kc9, c2]] = S[[kc9, c2]] + bgmem[[i]][[9, 2]]];
If[kc9 ≠ 0,
  S[[kc9, c3]] = S[[kc9, c3]] + bgmem[[i]][[9, 3]]];
If[kc9 ≠ 0,
  S[[kc9, c4]] = S[[kc9, c4]] + bgmem[[i]][[9, 4]]];
If[kc9 ≠ 0,
  S[[kc9, c5]] = S[[kc9, c5]] + bgmem[[i]][[9, 5]]];
If[kc9 ≠ 0,
  S[[kc9, c6]] = S[[kc9, c6]] + bgmem[[i]][[9, 6]]];
If[kc10 ≠ 0,
  S[[kc10, c1]] = S[[kc10, c1]] + bgmem[[i]][[10, 1]]];
If[kc10 ≠ 0,
  S[[kc10, c2]] = S[[kc10, c2]] + bgmem[[i]][[10, 2]]];
If[kc10 ≠ 0,
  S[[kc10, c3]] = S[[kc10, c3]] + bgmem[[i]][[10, 3]]];
If[kc10 ≠ 0,
  S[[kc10, c4]] = S[[kc10, c4]] + bgmem[[i]][[10, 4]]];
If[kc10 ≠ 0,
  S[[kc10, c5]] = S[[kc10, c5]] + bgmem[[i]][[10, 5]]];
If[kc10 ≠ 0,
  S[[kc10, c6]] = S[[kc10, c6]] + bgmem[[i]][[10, 6]]];
If[kc11 ≠ 0,
  S[[kc11, c1]] = S[[kc11, c1]] + bgmem[[i]][[11, 1]]];
If[kc11 ≠ 0,
  S[[kc11, c2]] = S[[kc11, c2]] + bgmem[[i]][[11, 2]]];
If[kc11 ≠ 0,
  S[[kc11, c3]] = S[[kc11, c3]] + bgmem[[i]][[11, 3]]];
If[kc11 ≠ 0,
  S[[kc11, c4]] = S[[kc11, c4]] + bgmem[[i]][[11, 4]]];
If[kc11 ≠ 0,
  S[[kc11, c5]] = S[[kc11, c5]] + bgmem[[i]][[11, 5]]];
If[kc11 ≠ 0,
  S[[kc11, c6]] = S[[kc11, c6]] + bgmem[[i]][[11, 6]]];

```

Figure 28: Computer codes to assemble Equilibrium Equation (10)

```

S[[kc11, c6]] = S[[kc11, c6]] + bgmem[[i]][[11, 6]];
If[kc12 ≠ 0,
  S[[kc12, c1]] = S[[kc12, c1]] + bgmem[[i]][[12, 1]];
If[kc12 ≠ 0,
  S[[kc12, c2]] = S[[kc12, c2]] + bgmem[[i]][[12, 2]];
If[kc12 ≠ 0,
  S[[kc12, c3]] = S[[kc12, c3]] + bgmem[[i]][[12, 3]];
If[kc12 ≠ 0,
  S[[kc12, c4]] = S[[kc12, c4]] + bgmem[[i]][[12, 4]];
If[kc12 ≠ 0,
  S[[kc12, c5]] = S[[kc12, c5]] + bgmem[[i]][[12, 5]];
If[kc12 ≠ 0,
  S[[kc12, c6]] = S[[kc12, c6]] + bgmem[[i]][[12, 6]],

{i, 1, m}]

```

Figure 29: Computer codes to assemble Equilibrium Equation (11)

And here is equilibrium equation matrix of the structure:

```

^ EQUILIBRIUM EQUATIONS
Print["S = ", MatrixForm[S]]
S =
(
0. -4. 0. 0. 0. -1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 4. 0. -1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. -1. 0. 0. 0.
1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. -1. 0. 0. 0. 0.
0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 0. -5. 0. 0. 0. -1.
0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 5. 0. -1. 0.
0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0.
)
Dimensions[S]
{9, 18}

```

Figure 30: Matrix form of Equilibrium Equations

5.3 Solution Algorithms

As stated in chapters 2 and 3, the IFMD and IFM are two methods employed in this study. In both methods the first step in analysis process is equilibrium equation generation. The only shared step in these two methods is generation of equilibrium equation matrix. In this section the algorithm upon which the computer programs for both methods are based on, is explained.

5.3.1 IFM via NS

In calculation of CC, the NS property of EE is utilized. Then the compatibility conditions are generated and equilibrium equations are coupled to get square matrix [S]. As a final point the square matrix is used to compute the unknown variables using the Equation21. The null space form of integrated force method is presented in Figure 33. The unconnected flexibility matrix of space frames and equilibrium equation are obtained via application of null space method and it is illustrated in Figure 31 in scatter plot form.

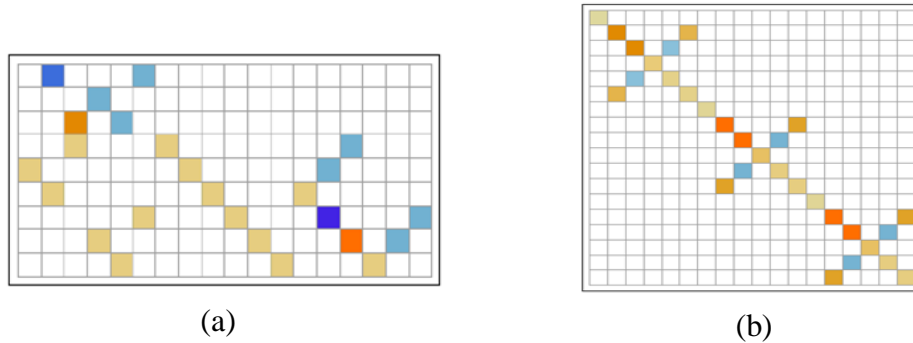


Figure 31: Flexibility and EE matrix plot

Figure 32 shows the scatter plot of coupled compatibility condition and equilibrium equation.

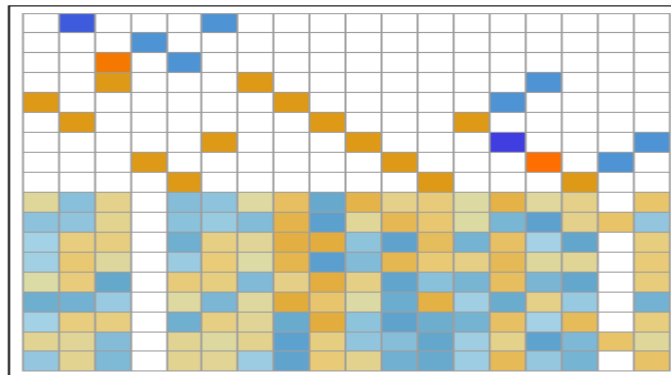


Figure 32: Coupled EE and CC via NS

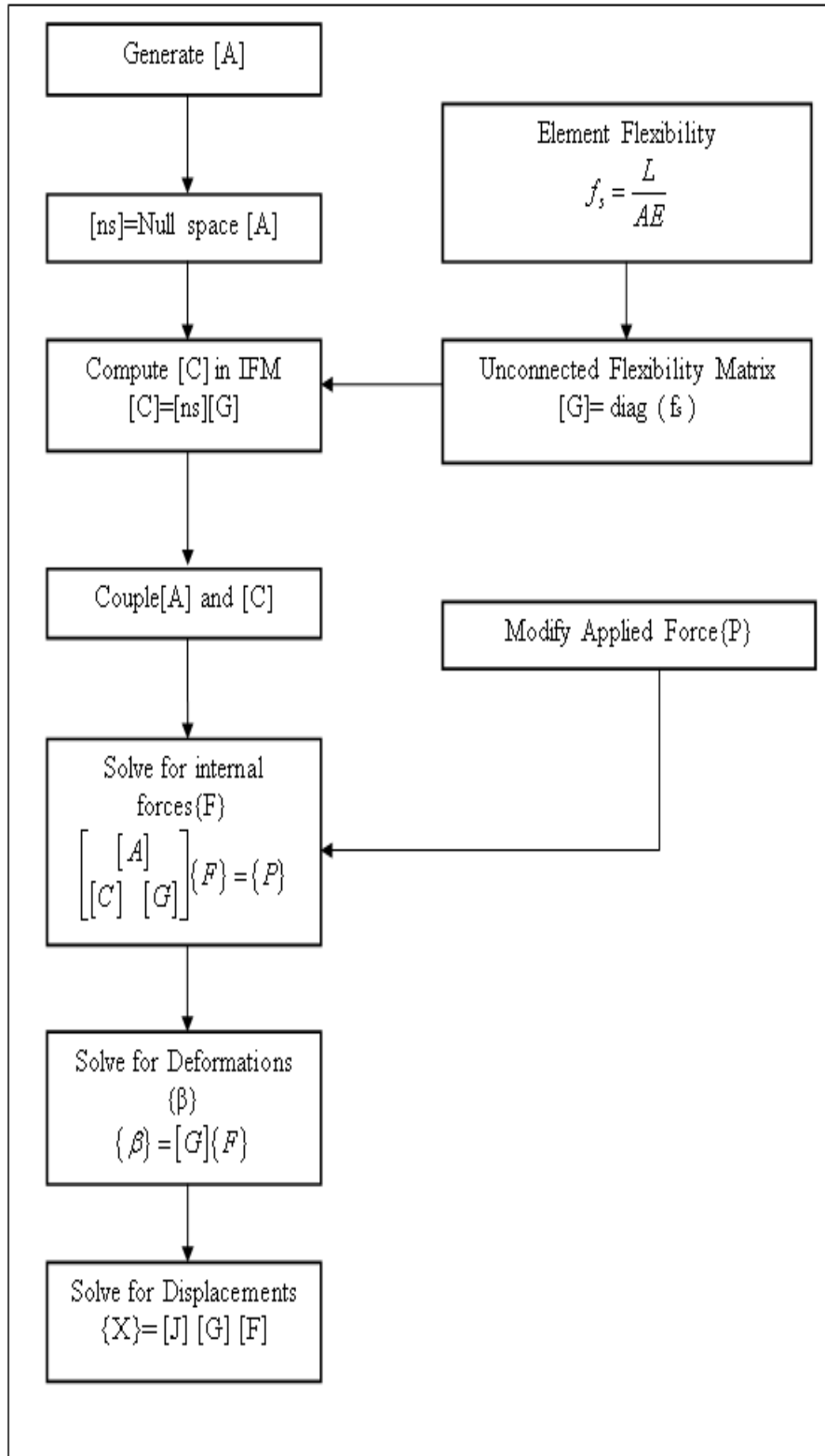


Figure 33: Algorithm of IFM via NS

5.3.3.1 Solution via Null Space

UNCONNECTED FLEXIBILITY MATRIX

COMPUTER CODES

```
Lmem;  
F = Table[0., {sr, 1, 6*m}, {sc, 1, 6*m}];  
Dimensions[F];  
mi = 0;  
Do[  
  node1 = inc[[i, 2]];  
  node2 = inc[[i, 3]];  
  k1 = 6*node1 - 5;  
  k2 = 6*node1 - 4;  
  k3 = 6*node1 - 3;  
  k4 = 6*node1 - 2;  
  k5 = 6*node1 - 1;  
  k6 = 6*node1;  
  mi = mi + 1;  
  c1 = 6*mi - 5;  
  c2 = 6*mi - 4;  
  c3 = 6*mi - 3;  
  c4 = 6*mi - 2;  
  c5 = 6*mi - 1;  
  c6 = 6*mi;  
  k7 = 6*node2 - 5;  
  k8 = 6*node2 - 4;  
  k9 = 6*node2 - 3;  
  k10 = 6*node2 - 2;  
  k11 = 6*node2 - 1;  
  k12 = 6*node2;  
  kc1 = dof[[k1]];  
  kc2 = dof[[k2]];  
  kc3 = dof[[k3]];  
  kc4 = dof[[k4]];  
  kc5 = dof[[k5]];  
  kc6 = dof[[k6]];  
  kc7 = dof[[k7]];
```

Figure 34: Unconnected flexibility matrix

```

kc8 = dof[[k8]];
kc9 = dof[[k9]];
kc10 = dof[[k10]];
kc11 = dof[[k11]];
kc12 = dof[[k12]];
flex1 =  $\frac{Lmem[[i]]}{A[[i]] \times Ee}$ ;
flex2 =  $\frac{(Lmem[[i]])^3}{3. \times Ee \times Iiz[[i]]}$ ;
flex2y =  $\frac{(Lmem[[i]])^3}{3. \times Ee \times Iiy[[i]]}$ ;
flex3 =  $\frac{Lmem[[i]]}{Ee \times Iiz[[i]]}$ ;
flex3y =  $\frac{Lmem[[i]]}{Ee \times Iiy[[i]]}$ ;
flex4 =  $\frac{(Lmem[[i]])^2}{2. \times Ee \times Iiz[[i]]}$ ;
flex4y =  $\frac{(Lmem[[i]])^2}{2. \times Ee \times Iiy[[i]]}$ ;
jpb = Iiz[[i]] + Iiy[[i]];
fm =  $\begin{pmatrix} flex1 & 0 & 0 & 0 & 0 & 0 \\ 0 & flex2 & 0 & 0 & 0 & flex4 \\ 0 & 0 & flex2y & 0 & -flex4y & 0 \\ 0 & 0 & 0 & \frac{Lmem[[i]]}{G \times jpb} & 0 & 0 \\ 0 & 0 & -flex4y & 0 & flex3y & 0 \\ 0 & flex4 & 0 & 0 & 0 & flex3 \end{pmatrix}$ ;
Print[i, " ", " ", " ", " ", MatrixForm[fm]];
Print[i, " ", " ", {c1, c2, c3, c4, c5, c6}];
F[[c1, c1]] = F[[c1, c1]] + fm[[1, 1]];
F[[c2, c2]] = F[[c2, c2]] + fm[[2, 2]];
F[[c3, c3]] = F[[c3, c3]] + fm[[3, 3]];
F[[c4, c4]] = F[[c4, c4]] + fm[[4, 4]];
F[[c5, c5]] = F[[c5, c5]] + fm[[5, 5]];
F[[c6, c6]] = F[[c6, c6]] + fm[[6, 6]];
F[[c2, c2 + 4]] = F[[c2, c2 + 4]] + fm[[2, 6]];
F[[c3, c3 + 2]] = F[[c3, c3 + 2]] + fm[[3, 5]];
F[[c5, c5 - 2]] = F[[c5, c5 - 2]] + fm[[5, 3]];
F[[c6, c6 - 4]] = F[[c6, c6 - 4]] + fm[[6, 2]],
{i, 1, m}

```

Figure 35: Unconnected flexibility matrix (2)

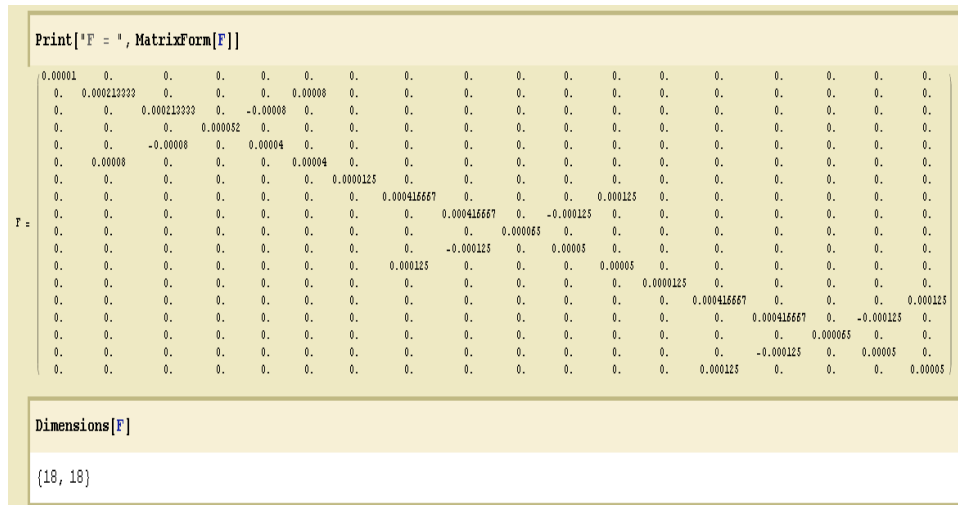


Figure 36: Flexibility Matrix

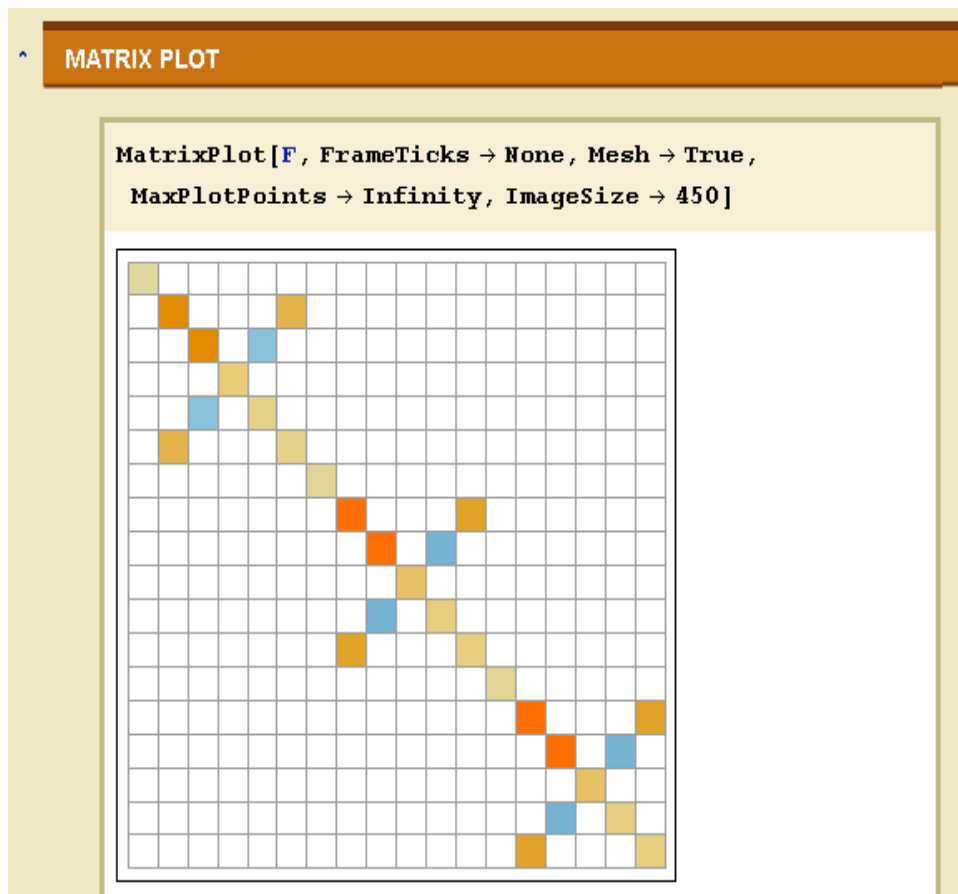


Figure 37: Matrix plot of flexibility matrix

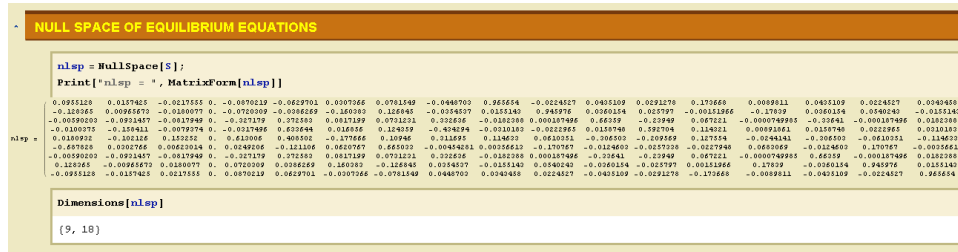


Figure 38: Null Property of Equilibrium Equations Matrix

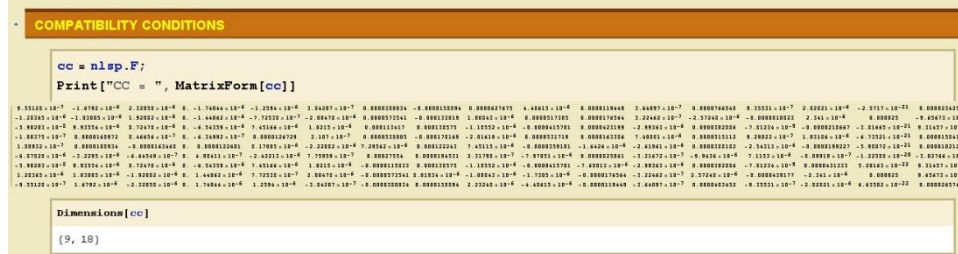


Figure 39: Compatibility Condition – Combination of Fig 37 & 38

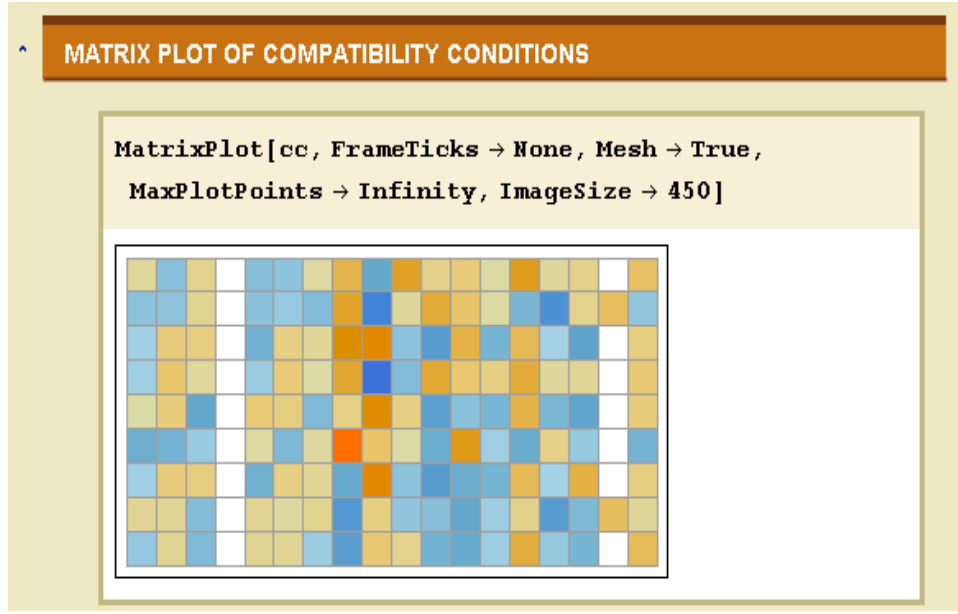


Figure 40: Matrix Plot of CC Matrix

COUPLE THE EQUILIBRIUM EQUATIONS WITH COMPATIBILITY CONDITIONS

COUPLE THE EQUILIBRIUM EQUATIONS WITH COMPATIBILITY CONDITIONS

```
ifm = Join[S, cc];  
Print["ifm = ", MatrixForm[ifm]]
```

```
Dimensions[ifm]
```

```
{18, 18}
```

MATRIX PLOT

```
MatrixPlot[ifm, FrameTicks → None, Mesh → True,  
MaxPlotPoints → Infinity, ImageSize → 450]
```

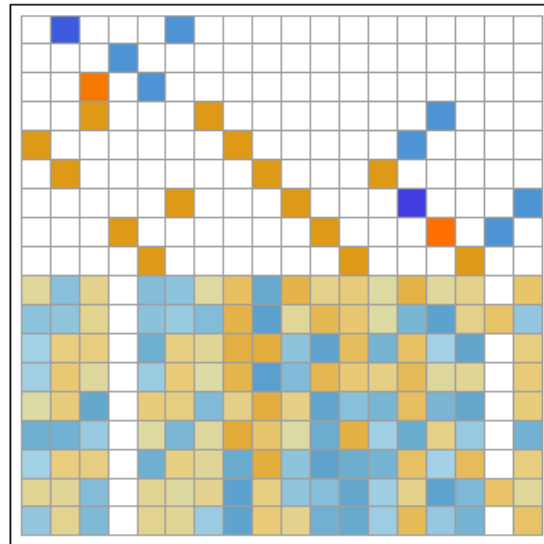


Figure 41: Coupling of EE with CC and its Matrix Plot

FORM THE JOINT LOAD VECTOR

COMPUTER CODES

```
P = Table[0., {sr, 1, 6 * noden - rest}, {sc, 1, 1}];
MatrixForm[P];
Do[k1 = 6 * i - 5;
  k2 = 6 * i - 4;
  k3 = 6 * i - 3;
  k4 = 6 * i - 2;
  k5 = 6 * i - 1;
  k6 = 6 * i;
  kc1 = dof[[k1]];
  kc2 = dof[[k2]];
  kc3 = dof[[k3]];
  kc4 = dof[[k4]];
  kc5 = dof[[k5]];
  kc6 = dof[[k6]];
  If[kc1 ≠ 0,
    P[[kc1, 1]] = P[[kc1, 1]] + applfrcs[[i, 2]];
  If[kc2 ≠ 0,
    P[[kc2, 1]] = P[[kc2, 1]] + applfrcs[[i, 3]];
  If[kc3 ≠ 0,
    P[[kc3, 1]] = P[[kc3, 1]] + applfrcs[[i, 4]];
  If[kc4 ≠ 0,
    P[[kc4, 1]] = P[[kc4, 1]] + applfrcs[[i, 5]];
  If[kc5 ≠ 0,
    P[[kc5, 1]] = P[[kc5, 1]] + applfrcs[[i, 6]];
  If[kc6 ≠ 0,
    P[[kc6, 1]] = P[[kc6, 1]] + applfrcs[[i, 7]];
  Print[i, " ", k1, " ", k2, " ", k3, " ", kc1,
    " ", kc2, " ", kc3],

  {i, 1, noden}]
MatrixForm[P]
```

Figure 42: Computer Codes for Assembling of Joints Loads

JOINT LOAD VECTOR

```
Print["Joint Loads = ", MatrixForm[P]]
```

Joint Loads = $\begin{pmatrix} 0. \\ 0. \\ 0. \\ 50. \\ 60. \\ 70. \\ 0. \\ 0. \\ 0. \end{pmatrix}$

```
Dimensions[P]
```

{9, 1}

Figure 43: Loads Matrix of Joints

CALCULATE THE DEGREE OF INDETERMINACY

```
di = 6 × m + rest - 6 × noden
```

9

Figure 44: Calculation of Degree of Indeterminacy

FORM THE FIXED FORCE

COMPUTER CODES

```
initial = Table[0., {sr, 1, di}, {sc, 1, 1}]
Pact = Join[P, initial]
Ffixed = Table[0., {sr, 1, 6 * noden - rest}, {sc, 1, 1}]
Dimensions[Ffixed]
initialW = Table[0., {sr, 1, di}, {sc, 1, 1}]
mi = 0;
Do[
  node1 = inc[[i, 2]];
  node2 = inc[[i, 3]];
  k1 = 6 * node1 - 5;
  k2 = 6 * node1 - 4;
  k3 = 6 * node1 - 3;
  k4 = 6 * node1 - 2;
  k5 = 6 * node1 - 1;
  k6 = 6 * node1;
  mi = mi + 1;
  c1 = 6 * mi - 5;
  c2 = 6 * mi - 4;
  c3 = 6 * mi - 3;
  c4 = 6 * mi - 2;
  c5 = 6 * mi - 1;
  c6 = 6 * mi;
  k7 = 6 * node2 - 5;
  k8 = 6 * node2 - 4;
  k9 = 6 * node2 - 3;
  k10 = 6 * node2 - 2;
  k11 = 6 * node2 - 1;
  k12 = 6 * node2;
  kc1 = dof[[k1]];
  kc2 = dof[[k2]];
  kc3 = dof[[k3]];
  kc4 = dof[[k4]];
  kc5 = dof[[k5]];
  kc6 = dof[[k6]];
```

Figure 45: Computer Codes for Formation of Fixed End Forces

```

kc7 = dof[[k7]];
kc8 = dof[[k8]];
kc9 = dof[[k9]];
kc10 = dof[[k10]];
kc11 = dof[[k11]];
kc12 = dof[[k12]];
If[kc1 ≠ 0, Ffixed[[kc1, 1]] =
  Ffixed[[kc1, 1]] + Ffmem[[i]][[1, 1]]];
If[kc2 ≠ 0, Ffixed[[kc2, 1]] =
  Ffixed[[kc2, 1]] + Ffmem[[i]][[2, 1]]];
If[kc3 ≠ 0, Ffixed[[kc3, 1]] =
  Ffixed[[kc3, 1]] + Ffmem[[i]][[3, 1]]];
If[kc4 ≠ 0, Ffixed[[kc4, 1]] =
  Ffixed[[kc4, 1]] + Ffmem[[i]][[4, 1]]];
If[kc5 ≠ 0, Ffixed[[kc5, 1]] =
  Ffixed[[kc5, 1]] + Ffmem[[i]][[5, 1]]];
If[kc6 ≠ 0, Ffixed[[kc6, 1]] =
  Ffixed[[kc6, 1]] + Ffmem[[i]][[6, 1]]];
If[kc7 ≠ 0, Ffixed[[kc7, 1]] =
  Ffixed[[kc7, 1]] + Ffmem[[i]][[7, 1]]];
If[kc8 ≠ 0, Ffixed[[kc8, 1]] =
  Ffixed[[kc8, 1]] + Ffmem[[i]][[8, 1]]];
If[kc9 ≠ 0, Ffixed[[kc9, 1]] =
  Ffixed[[kc9, 1]] + Ffmem[[i]][[9, 1]]];
If[kc10 ≠ 0, Ffixed[[kc10, 1]] =
  Ffixed[[kc10, 1]] + Ffmem[[i]][[10, 1]]];
If[kc11 ≠ 0, Ffixed[[kc11, 1]] =
  Ffixed[[kc11, 1]] + Ffmem[[i]][[11, 1]]];
If[kc12 ≠ 0, Ffixed[[kc12, 1]] =
  Ffixed[[kc12, 1]] + Ffmem[[i]][[12, 1]]];
Print[i, " ---", kc1, " ", kc1, " ", kc3,
  " ", kc4, " ", kc5, " ", kc6],
{i, 1, m}]

```

Figure 46: Computer Codes for Formation of Fixed End Forces (2)

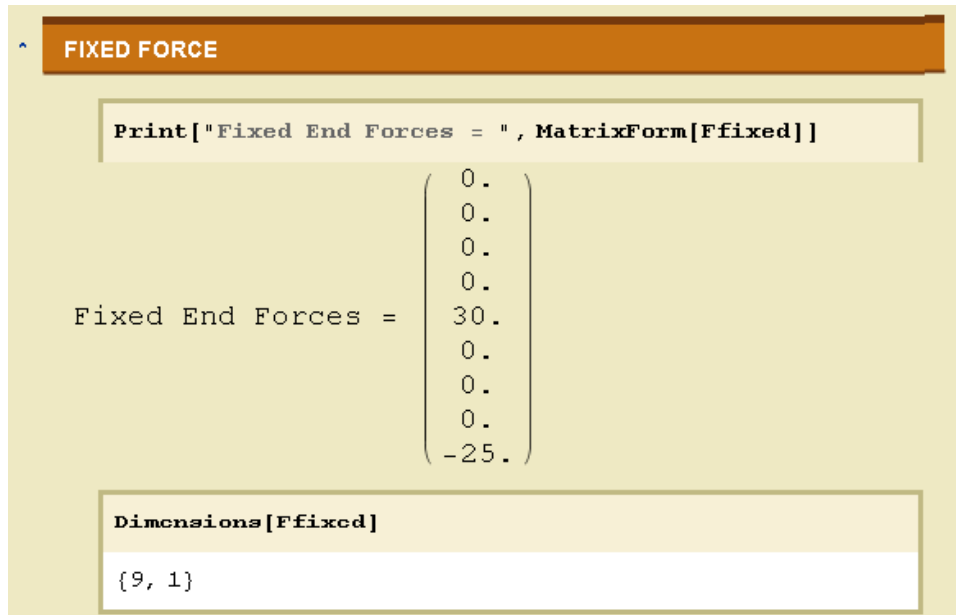


Figure 47: Matrix of Fixed End Forces

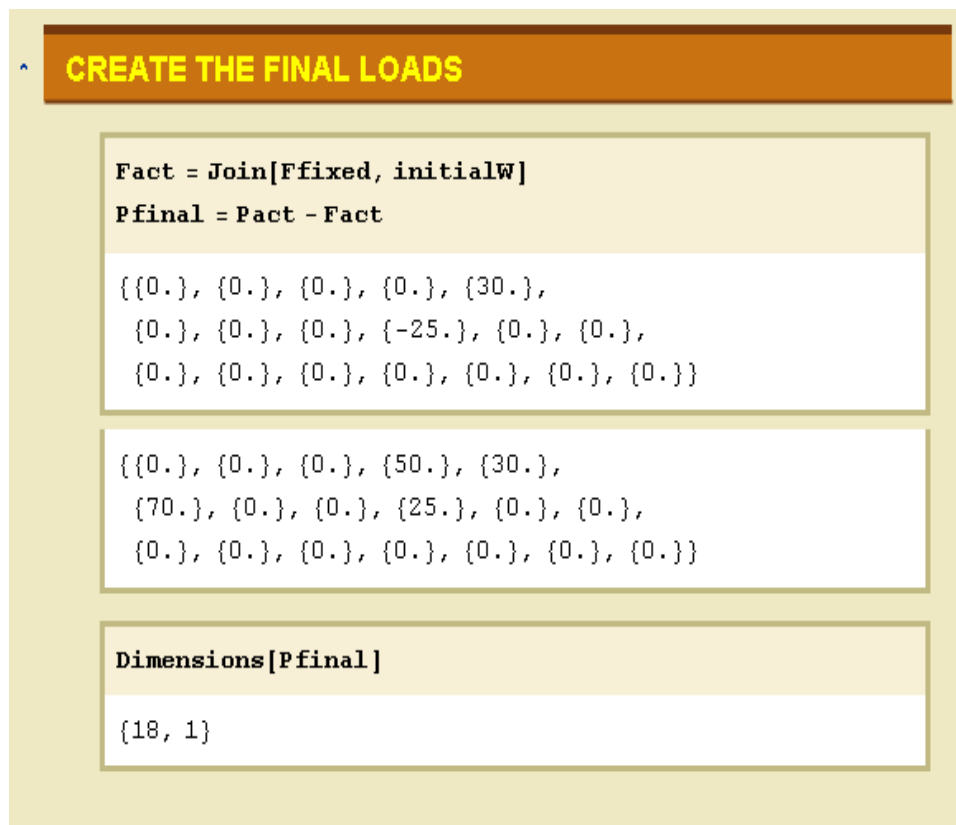


Figure 48: Final Applied Forces

FIND THE INDEPENDENT FORCES

```
indFrcs = LinearSolve[ifm, Pfinal];  
Print["          indFrcs = ", MatrixForm[indFrcs]]
```

```
indFrcs = 
$$\begin{pmatrix} 26.8629 \\ 2.70848 \\ 4.75743 \\ 0. \\ 19.0297 \\ -10.8339 \\ 39.5351 \\ -0.545536 \\ 6.34971 \\ 0.707552 \\ 15.0715 \\ 3.96749 \\ 60.9418 \\ -3.68262 \\ -5.70751 \\ 2.00281 \\ -13.466 \\ 8.28673 \end{pmatrix}$$

```

```
Dimensions[indFrcs]
```

```
{18, 1}
```

Figure 49: Finding Matrix of Members independent Forces

CALCULATE DISPLACEMENTS

COMPUTER CODES

```
invIFM = Inverse[ifm];  
tinvIFM = Transpose[invIFM];  
jd = Take[tinvIFM, 6*noden - rest];  
Dimensions[jd]  
Disp = jd.F.indFrcs
```

```
{9, 18}
```

```
{{0.000262669}, {-0.0000401376}, {-0.000250412},  
 {0.000494188}, {0.000268629}, {0.000761773},  
 {0.0000459909}, {-0.0000401376}, {0.000130182}}
```

NODAL DISPLACEMENTS

```
MatrixForm[Disp]
```

```

$$\begin{pmatrix} 0.000262669 \\ -0.0000401376 \\ -0.000250412 \\ 0.000494188 \\ 0.000268629 \\ 0.000761773 \\ 0.0000459909 \\ -0.0000401376 \\ 0.000130182 \end{pmatrix}$$

```

Figure 50: Calculation of Nodal Displacements Matrix

FIND THE MEMBER END FORCES

```

mend = 0;
endfrcs = {};
Do[
  mend = mend + 1;
  cm1 = 6 * mend - 5;
  cm2 = 6 * mend - 4;
  cm3 = 6 * mend - 3;
  cm4 = 6 * mend - 2;
  cm5 = 6 * mend - 1;
  cm6 = 6 * mend;

  endfrc = bmem[[i]].
    
$$\begin{pmatrix} \text{indFrcs}[[\text{cm1}, 1]] \\ \text{indFrcs}[[\text{cm2}, 1]] \\ \text{indFrcs}[[\text{cm3}, 1]] \\ \text{indFrcs}[[\text{cm4}, 1]] \\ \text{indFrcs}[[\text{cm5}, 1]] \\ \text{indFrcs}[[\text{cm6}, 1]] \end{pmatrix} +$$

  Qfmem[[i]];
  Print["member ", i, " ", MatrixForm[endfrc]];
  AppendTo[endfrcs, endfrc],
  {i, 1, m}
]

```

```

member 1

$$\begin{pmatrix} -26.8629 \\ -2.70848 \\ -4.75743 \\ 0. \\ 0. \\ 0. \\ 26.8629 \\ 2.70848 \\ 4.75743 \\ 0. \\ 19.0297 \\ -10.8339 \end{pmatrix}$$


```

Figure 51: Calculation of Member End Forces

$$\begin{matrix} & \begin{pmatrix} -39.5351 \\ 30.5455 \\ -6.34971 \\ -0.707552 \\ 16.677 \\ 23.7602 \\ 39.5351 \\ 29.4545 \\ 6.34971 \\ 0.707552 \\ 15.0715 \\ -21.0325 \end{pmatrix} \\ \text{member } 2 & \\ & \\ & \\ & \\ & \\ \begin{pmatrix} -60.9418 \\ 3.68262 \\ 5.70751 \\ -2.00281 \\ -15.0715 \\ 10.1264 \\ 60.9418 \\ -3.68262 \\ -5.70751 \\ 2.00281 \\ -13.466 \\ 8.28673 \end{pmatrix} \\ \text{member } 3 & \end{matrix}$$

Figure 52: Calculation of Member End Forces (continued)

5.3.2 IFM via SVD

Singular value decomposition approach outlined in section 4.2.3 is another approach for finding the compatibility condition. After having equilibrium equation generated, the $([A]^T)^{pinv}$ is acquired via Eq.36 and Then, using Eq.37 the $[M]$ matrix is calculated. In

the next step the SVD of the $[M]$ is undertaken in order to calculate the $[M_u]$, $[M_v]$ and $[M_s]$. Finally, CC is calculated via equation 41 & 42. This procedure is illustrated in figure number 54.

It worth noting that, flexibility matrix, and equilibrium equation and scatter plots remains the same as null space after the application of this method on the frame depicted in Figure14. In Figure 54 depicts the scatter plot of coupled compatibility condition and equilibrium equation.

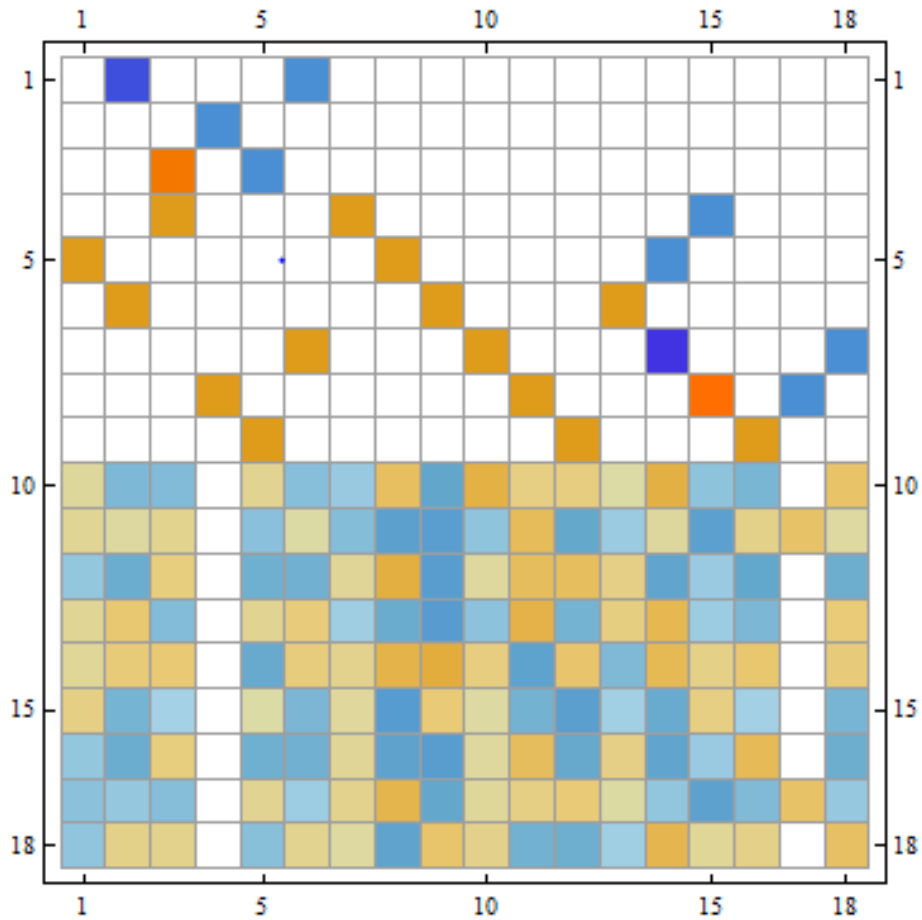


Figure 53: Coupled EE and Compatibility Condition via SVD

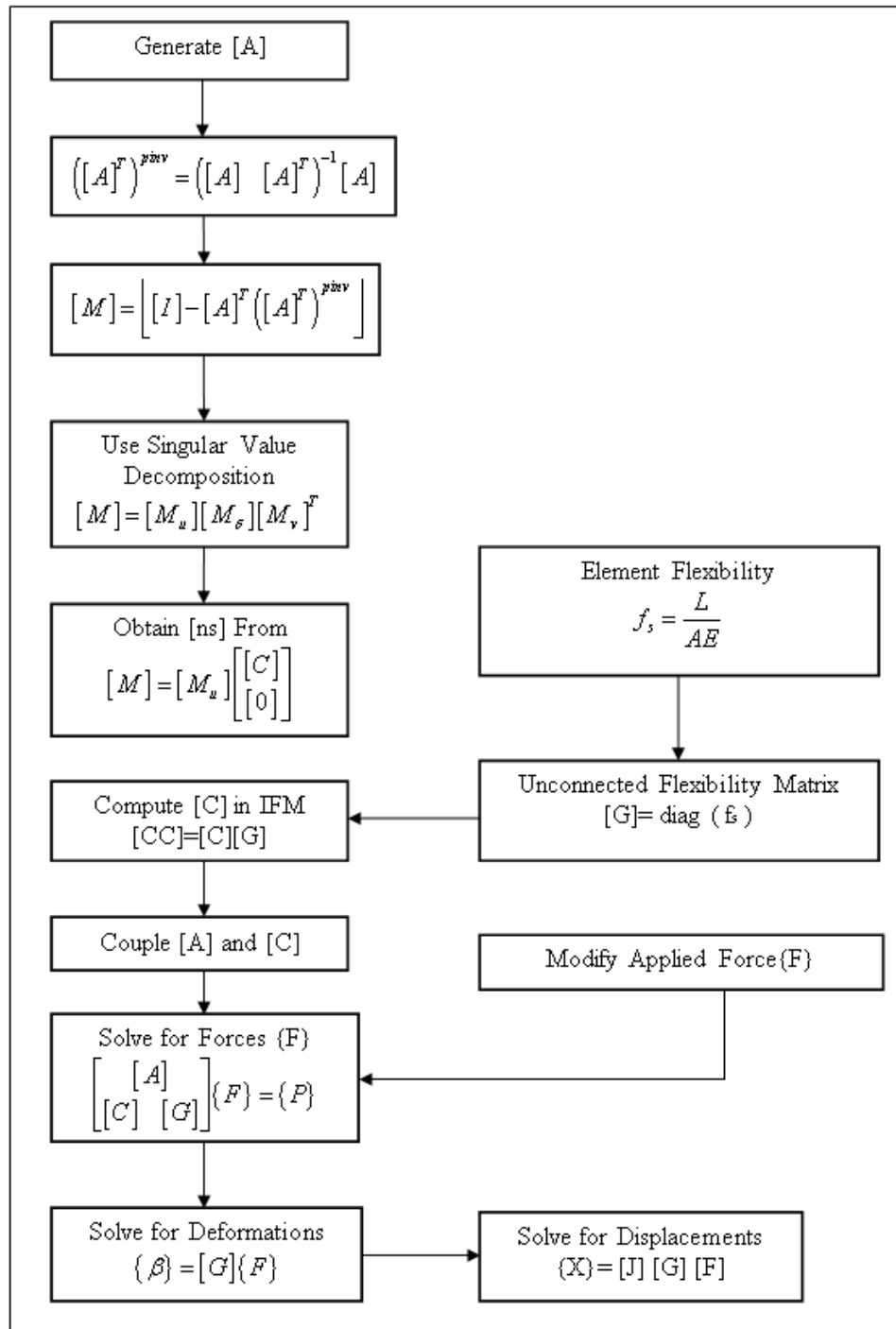


Figure 54: Algorithm of IFM via SVD

5.3.2.1 Solution of example via SVD

SINGULAR VALUE DECOMPOSITION EQUILIBRIUM EQUATIONS

```
apinv = (Inverse[{S.Transpose[S]}]).S;  
sa = IdentityMatrix[6 × m] - (Transpose[S].apinv);  
{u, w, v} = SingularValueDecomposition[sa];  
Print[" Mu = " MatrixForm[u]]  
Print[" Mw = " MatrixForm[w]]  
Print[" MvT = " MatrixForm[v]]
```

Figure 55: Codes definition to find SVD

COMPATIBILITY CONDITIONS

```
c2 = Chop[Inverse[u].sa];  
{row, col} = Dimensions[S];  
c1 = Take[c2, col - row, col];  
cc = c1.F;  
Print["CC = ", MatrixForm[cc]]
```

Dimensions[cc]

```
{9, 18}
```

MATRIX PLOT

```
MatrixPlot[cc, FrameTicks → None, Mesh → True,  
MaxPlotPoints → Infinity, ImageSize → 450]
```

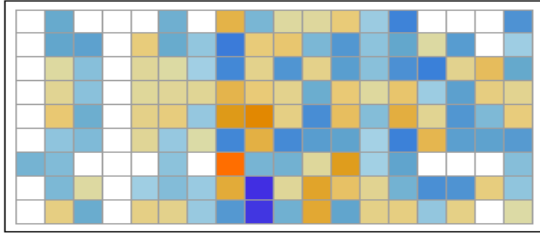


Figure 56: Assembling of CC by Using SVD and Matrix Plot

5.3.3 IFMD Method

The step by step process and equations for the IFMD is shown in Figure 58. After creation of EE in IFMD by using of Eq. 44 the matrix $[K]_{ifmd}$ is assembled. The inverse of flexibility matrix is used to get $[K]_{ifmd}$. After it Eq.43 gives the nodal displacements. In the last step Eq.49 gives the internal forces.

In this technique the scatter plot of EE remains same since the element and nodal system of numbering are same. Figure 57 shows the plots of flexibility matrix and K matrix.

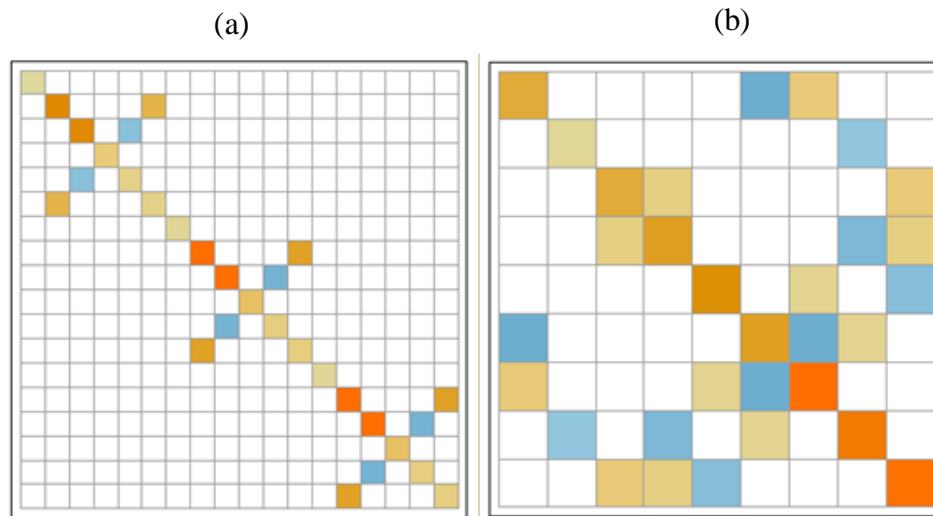


Figure 57: (a) Flexibility matrix plot - (b) Plot of stiffness Matrix

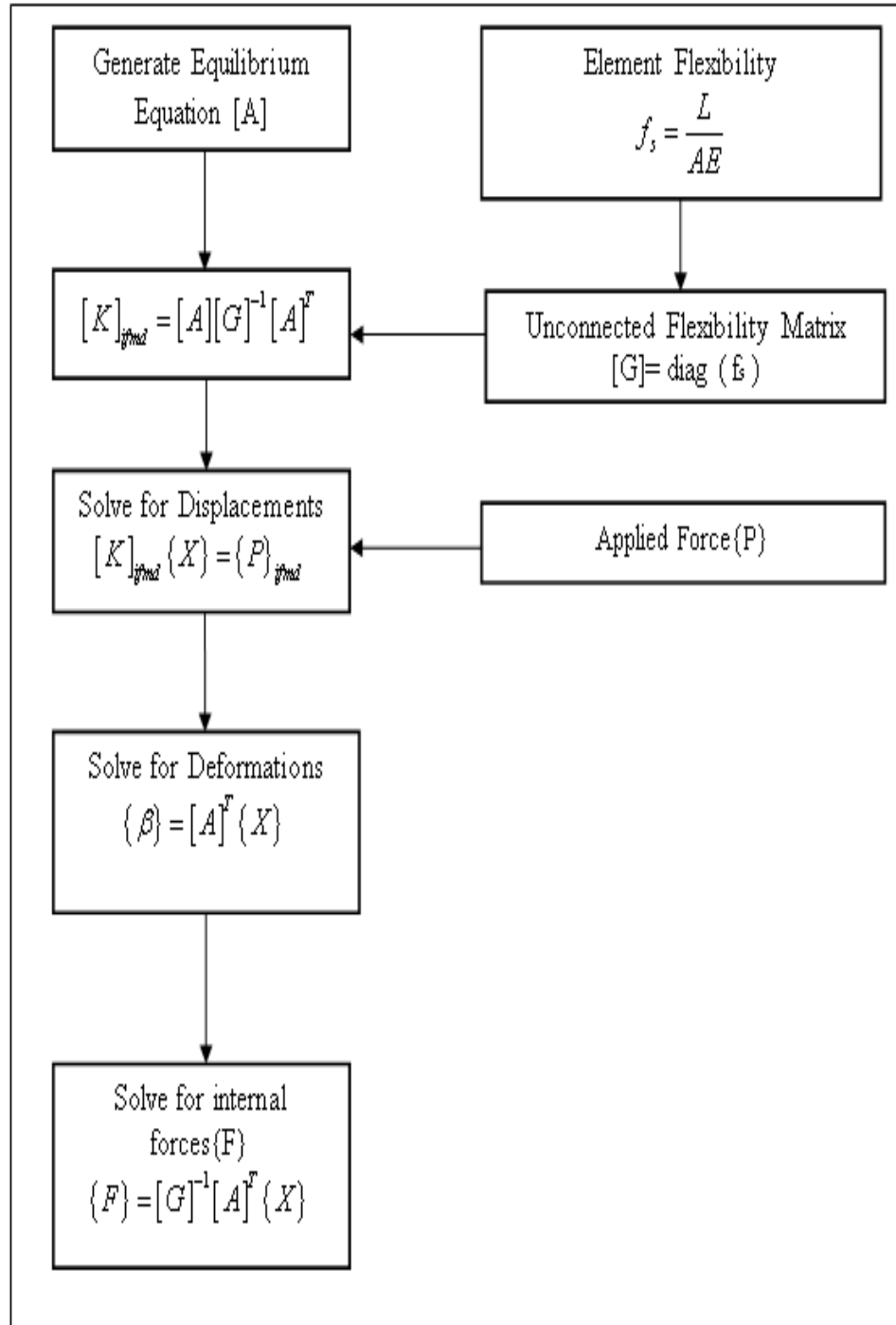


Figure 58: Algorithm for IFMD

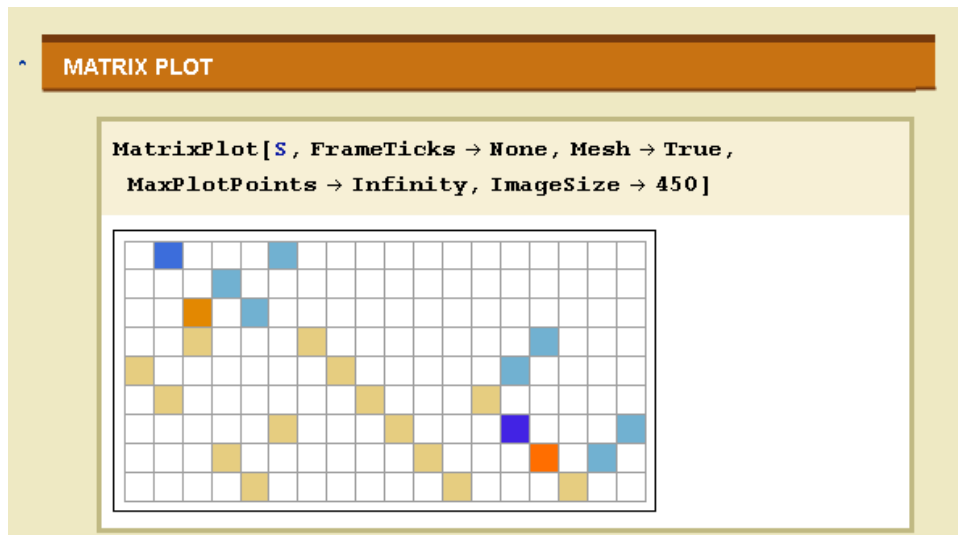


Figure 59: Matrix Plot of S Matrix

5.3.3.1 Solution of the example via Dual Integrated Force Method

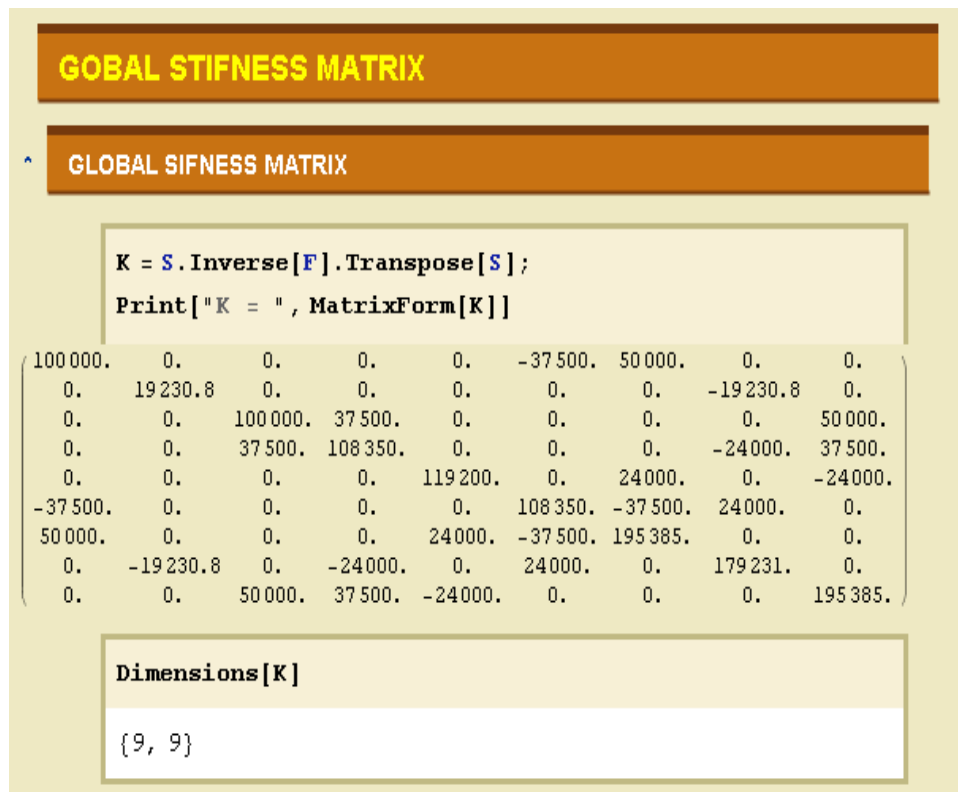


Figure 60: Calculation of Stiffness Matrix for IFMD

MATRIX PLOT

```
MatrixPlot[K, FrameTicks -> None, Mesh -> True,  
MaxPlotPoints -> Infinity, ImageSize -> 450]
```

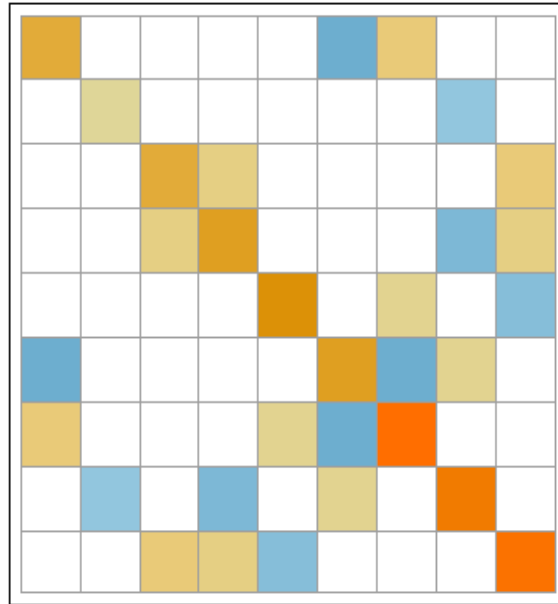


Figure 61: Matrix Plot of Stiffness Matrix

FIND THE DISPLACEMENTS

```
dsplacements = LinearSolve[K, Pfinal];  
Print["      x = ",  
MatrixForm[dsplacements]]
```

```
x = 
$$\begin{pmatrix} 0.000262669 \\ -0.0000401376 \\ -0.000250412 \\ 0.000494188 \\ 0.000268629 \\ 0.000761773 \\ 0.0000459909 \\ -0.0000401376 \\ 0.000130182 \end{pmatrix}$$

```

```
Dimensions[dsplacements]
```

```
{9, 1}
```

Figure 62: Calculation of Nodal Displacements by IFMD

FIND THE INDEPENDENT FORCES

```
indFrcs = Inverse[F].Transpose[S].dsplacements;  
Print["      indFrcs = ", MatrixForm[indFrcs]]
```

```
indFrcs = 
$$\begin{pmatrix} 26.8629 \\ 2.70848 \\ 4.75743 \\ 0. \\ 19.0297 \\ -10.8339 \\ 39.5351 \\ -0.545536 \\ 6.34971 \\ 0.707552 \\ 15.0715 \\ 3.96749 \\ 60.9418 \\ -3.68262 \\ -5.70751 \\ 2.00281 \\ -13.466 \\ 8.28673 \end{pmatrix}$$

```

```
Dimensions[indFrcs]
```

```
{18, 1}
```

Figure 63: Calculation of Member Forces By IFMD

Chapter 6

EXPLANATORY EXAMPLES

6.1 Introduction

Two examples are presented in this chapter, one of which is an 8 member frame and the other a 16 member frame. These two examples are solved via Null Space, SVD and IFMD approaches and the pertaining results are compared to Mastan version 3 to find the accuracy level of our results.

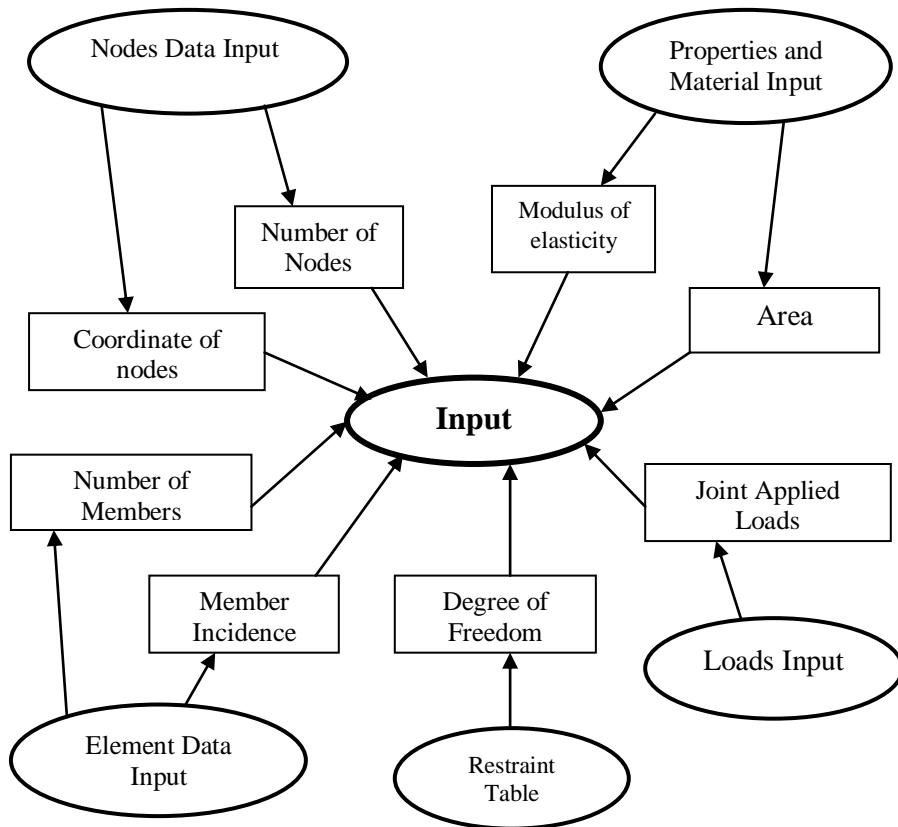


Figure 64: Structure of input data

Figure 64 shows the structure of input data in the program which has been discussed before and should be taken into account during solving process.

6.2 Example 1

In this example a space frame with 8 members and 8 nodes were analyzed. The information regarding elemental and nodal properties is presented in Table 1. The area of members, the moment of inertia and the modulus of elasticity are assumed to be 0.002 m^2 , 0.0005 m^4 and $2 \times 10^8 \text{ N/m}^2$ respectively.

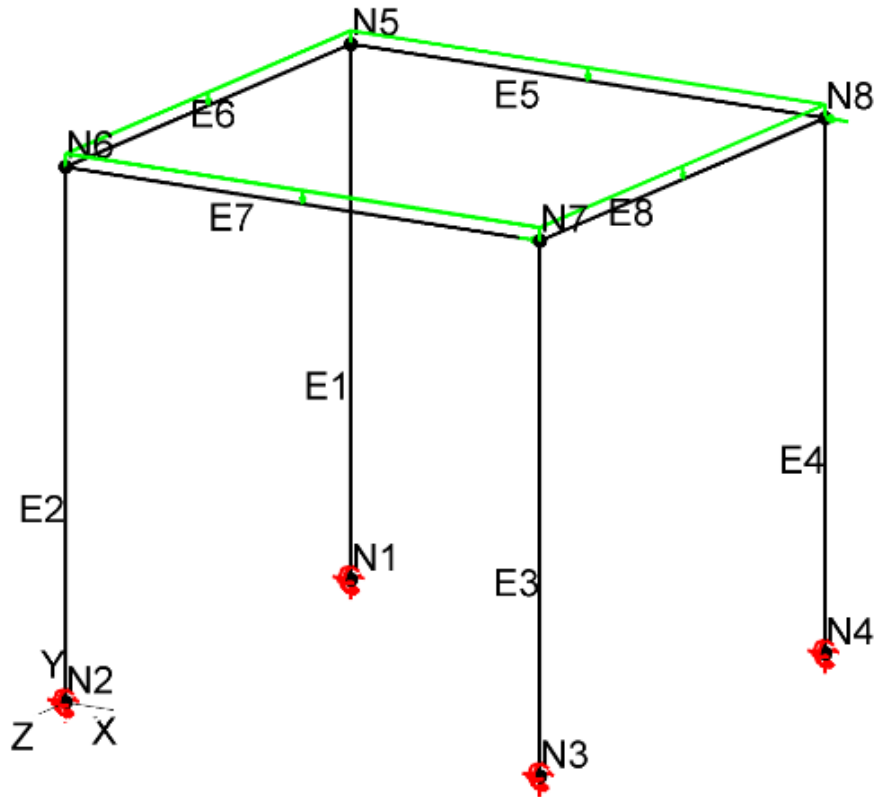


Figure 65: Scheme of 8 member structure

Table 1: Elemental and nodal properties of example 1

Node number	Coordinates			Applied Loads (kN)			Applied Moment (kNm)			Restrains			Restrains		
	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
1	0	0	-9.144	0	0	0	0	0	0	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
3	9.144	0	0	0	0	0	0	0	0	1	1	1	1	1	1
4	9.144	0	-9.144	0	0	0	0	0	0	1	1	1	1	1	1
5	0	9.144	-9.144	0	0	0	0	0	0	0	0	0	0	0	0
6	0	9.144	0	0	0	0	0	0	0	0	0	0	0	0	0
7	9.144	9.144	0	40	0	0	0	0	0	0	0	0	0	0	0
8	9.144	9.144	-9.144	-40	0	0	0	0	0	0	0	0	0	0	0

Table 2: Nodal condition of elements and uniform loads on each member

Element Number	Connectivity		Applied Uniform Load XY Plane
	Start Node	End Node	
1	1	5	0
2	2	6	0
3	3	7	0
4	4	8	0
5	5	8	-2
6	5	6	-2
7	6	7	-2
8	7	8	-2

FIND THE DISPLACEMENTS

```
dsplacements = LinearSolve[K, Pfinal];  
Print[" X = ", MatrixForm[dsplacements]]
```

X =

$$\begin{pmatrix} -0.00950421 \\ -0.000499781 \\ 0.0050808 \\ 0.000492923 \\ 0.00148846 \\ 0.000329747 \\ 0.00953874 \\ -0.000336347 \\ 0.00504627 \\ 0.0000643938 \\ 0.00148846 \\ -0.000758276 \\ 0.00995151 \\ -0.000499781 \\ -0.0050808 \\ -0.000492923 \\ 0.0015554 \\ -0.000368682 \\ -0.00998604 \\ -0.000336347 \\ -0.00504627 \\ -0.0000643938 \\ 0.0015554 \\ 0.00079721 \end{pmatrix}$$

Figure 66: Results (1)

FIND THE MEMBER END FORCES

member 1	$\begin{pmatrix} 21.8627 \\ 4.43733 \\ -12.551 \\ -12.5216 \\ 60.9893 \\ 25.6782 \\ -21.8627 \\ -4.43733 \\ 12.551 \\ 12.5216 \\ 53.777 \\ 14.8968 \end{pmatrix}$
member 2	$\begin{pmatrix} 14.7133 \\ 7.45824 \\ 9.5301 \\ -12.5216 \\ -51.8642 \\ 34.8033 \\ -14.7133 \\ -7.45824 \\ -9.5301 \\ 12.5216 \\ -35.279 \\ 33.3948 \end{pmatrix}$
member 3	$\begin{pmatrix} 21.8627 \\ -4.43733 \\ 12.9737 \\ -13.0847 \\ -63.3475 \\ -25.6782 \\ -21.8627 \\ 4.43733 \\ -12.9737 \\ 13.0847 \\ -55.2836 \\ -14.8968 \end{pmatrix}$

Figure 67: Results - Member Forces (2)

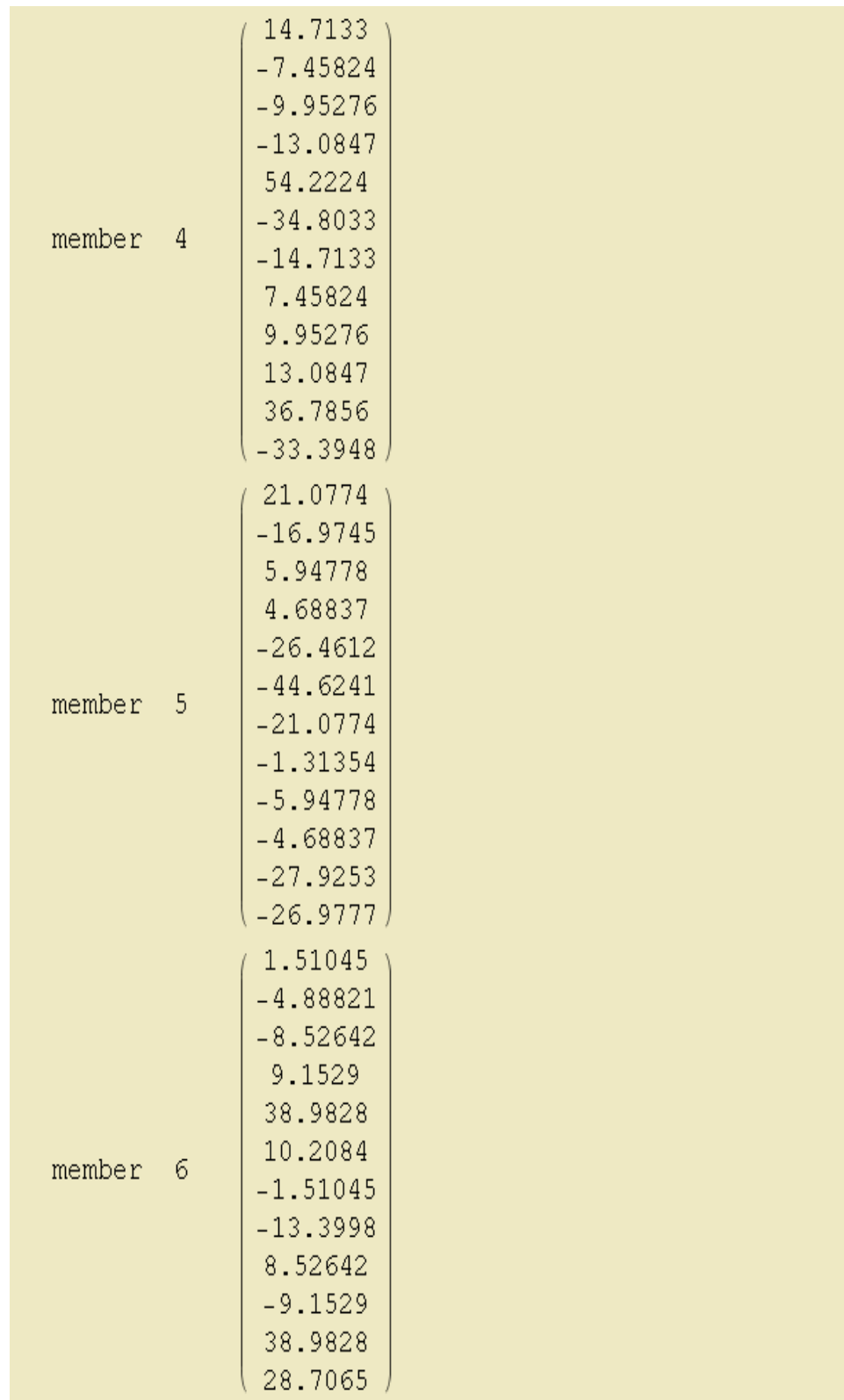


Figure 68: Results - Member Forces (3)

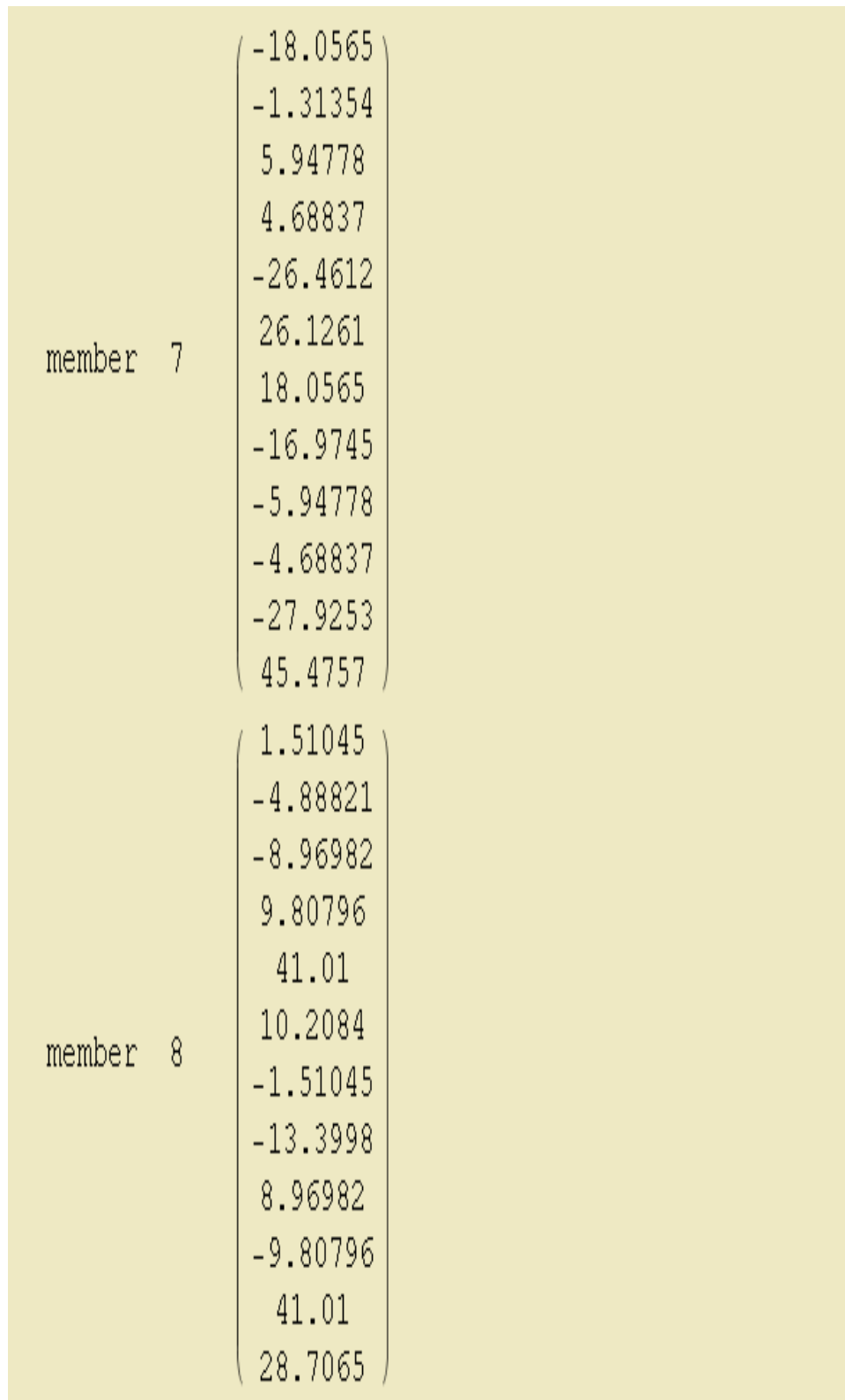


Figure 69: Results Reporting - Member Forces (4)

6.2.1 Results of Mastan program for example 1:

***** MASTAN2 v3.3.1 *****

Time: 02:13:11 Date: 11/18/2012

Problem Title: 8 Member

Results of Structural Analysis
#####

General Information:

Structure Analyzed as: Space Frame
Analysis Type: First-Order Elastic

Analytical Results:

(i) Displacements at Step # 1, Applied Load Ratio = 1.0000

Deflections

Node	X-disp	Y-disp	Z-disp
1	0.0000e+000	0.0000e+000	0.0000e+000
2	0.0000e+000	0.0000e+000	0.0000e+000
3	0.0000e+000	0.0000e+000	0.0000e+000
4	0.0000e+000	0.0000e+000	0.0000e+000
5	-9.5042e-003	-4.9978e-004	5.0808e-003
6	9.5387e-003	-3.3635e-004	5.0463e-003
7	9.9515e-003	-4.9978e-004	-5.0808e-003
8	-9.9860e-003	-3.3635e-004	-5.0463e-003

Rotations (radians)

Node	X-rot	Y-rot	Z-rot
1	0.0000e+000	0.0000e+000	0.0000e+000
2	0.0000e+000	0.0000e+000	0.0000e+000
3	0.0000e+000	0.0000e+000	0.0000e+000
4	0.0000e+000	0.0000e+000	0.0000e+000
5	4.9292e-004	1.4885e-003	3.2975e-004
6	6.4394e-005	1.4885e-003	-7.5828e-004
7	-4.9292e-004	1.5554e-003	-3.6868e-004
8	-6.4394e-005	1.5554e-003	7.9721e-004

(ii) Element Results at Step # 1, Applied Load Ratio = 1.0000

Internal End Forces (Note: Refers to local coordinates)

Element	Node	Fx	Fy	Fz
1	1	2.1863e+001	-1.2551e+001	-4.4373e+000
	5	-2.1863e+001	1.2551e+001	4.4373e+000

2	2	1.4713e+001	9.5301e+000	-7.4582e+000
	6	-1.4713e+001	-9.5301e+000	7.4582e+000
3	3	2.1863e+001	1.2974e+001	4.4373e+000
	7	-2.1863e+001	-1.2974e+001	-4.4373e+000
4	4	1.4713e+001	-9.9528e+000	7.4582e+000
	8	-1.4713e+001	9.9528e+000	-7.4582e+000
5	5	2.1077e+001	1.6974e+001	-5.9478e+000
	8	-2.1077e+001	1.3135e+000	5.9478e+000
6	5	1.5105e+000	4.8882e+000	8.5264e+000
	6	-1.5105e+000	1.3400e+001	-8.5264e+000
7	6	-1.8057e+001	1.3135e+000	-5.9478e+000
	7	1.8057e+001	1.6974e+001	5.9478e+000
8	7	1.5105e+000	4.8882e+000	8.9698e+000
	8	-1.5105e+000	1.3400e+001	-8.9698e+000

Internal End Moments (Note: Refers to local coordinates)

Element	Node	Mx	My	Mz
1	1	-1.2522e+001	2.5678e+001	-6.0989e+001
	5	1.2522e+001	1.4897e+001	-5.3777e+001
2	2	-1.2522e+001	3.4803e+001	5.1864e+001
	6	1.2522e+001	3.3395e+001	3.5279e+001
3	3	-1.3085e+001	-2.5678e+001	6.3348e+001
	7	1.3085e+001	-1.4897e+001	5.5284e+001
4	4	-1.3085e+001	-3.4803e+001	-5.4222e+001
	8	1.3085e+001	-3.3395e+001	-3.6786e+001
5	5	4.6884e+000	2.6461e+001	4.4624e+001
	8	-4.6884e+000	2.7925e+001	2.6978e+001
6	5	9.1529e+000	-3.8983e+001	-1.0208e+001
	6	-9.1529e+000	-3.8983e+001	-2.8706e+001
7	6	4.6884e+000	2.6461e+001	-2.6126e+001
	7	-4.6884e+000	2.7925e+001	-4.5476e+001
8	7	9.8080e+000	-4.1010e+001	-1.0208e+001
	8	-9.8080e+000	-4.1010e+001	-2.8706e+001

```
#####
End of Results of Structural Analysis
#####
```

6.3 Example 2

In this example a space frame with 16 members and 12 nodes were analyzed. The information regarding elemental and nodal properties is presented in Table 3. The area of members, the moment of inertia and the modulus of elasticity are assumed to be 0.002 m², 0.0005 m⁴ and 2* 10⁸ N/m² respectively.

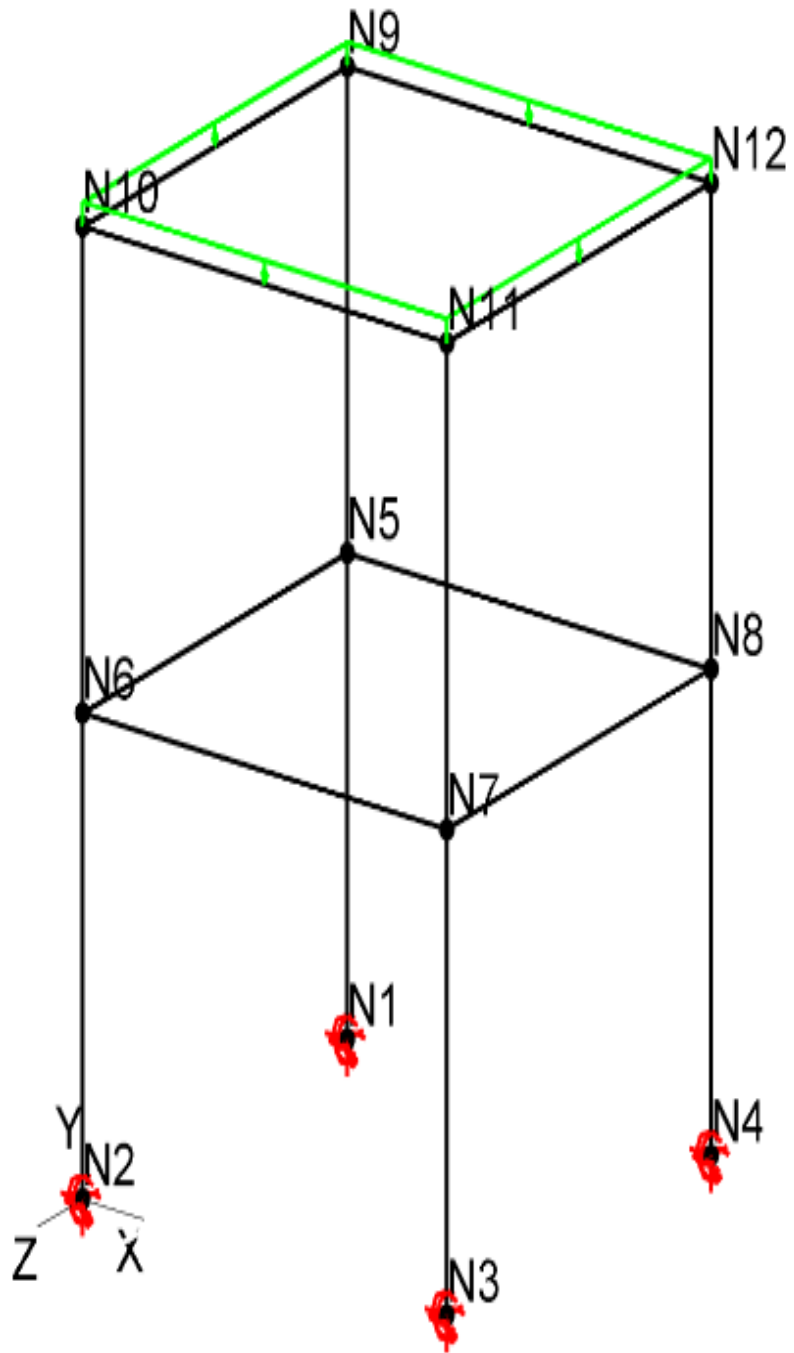


Figure 70 : Scheme of a 16 member structure

Table 3: Elemental and nodal properties of example 2

Node Number	Coordinates			Applied Loads (kN)			Applied Moment (kNm)			Restrains			Restrains		
	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
1	0	0	-5	0	0	0	0	0	0	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
3	5	0	0	0	0	0	0	0	0	1	1	1	1	1	1
4	5	0	-5	0	0	0	0	0	0	1	1	1	1	1	1
5	0	4	-5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0
7	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0
8	5	4	-5	0	0	0	0	0	0	0	0	0	0	0	0
9	0	8	-5	0	0	0	0	0	0	0	0	0	0	0	0
10	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0
11	5	8	0	0	0	0	0	0	0	0	0	0	0	0	0
12	5	8	-5	0	0	0	0	0	0	0	0	0	0	0	0

Table 4: Nodal condition of elements / uniform loads on each member - Example 2

Element Number	Connectivity		Applied Uniform Load XY Plane
	Start Node	End Node	
1	1	5	0
2	2	6	0
3	3	7	0
4	4	8	0
5	5	8	0
6	5	6	0
7	6	7	0
8	7	8	0
9	5	9	0
10	6	10	0
11	7	11	0
12	8	12	0
13	9	12	-15
14	9	10	-25
15	10	11	-15
16	11	12	-25

FIND THE INDEPENDENT FORCES

```
indFrcs = LinearSolve[ifm, Pfinal];  
Print["          indFrcs = ", MatrixForm[indFrcs]]
```

```
          -100.  
          -1.5956  
          -0.957358  
          1.57864 × 10-14  
          -3.1312  
          5.21866  
          -100.  
          1.5956  
          -0.957358  
          1.51807 × 10-15  
          -3.1312  
          -5.21866  
          -100.  
          1.5956  
          0.957358  
          1.64771 × 10-14  
          3.1312  
          -5.21866  
          -100.  
          -1.5956  
          0.957358  
          -7.8605 × 10-15  
          3.1312  
          5.21866  
          7.40153  
          4.43062 × 10-15  
          1.20802 × 10-14  
          1.1272 × 10-14  
          3.01483 × 10-14  
          1.94637  
          12.3359
```

Figure 71: Member Forces – Ex.2

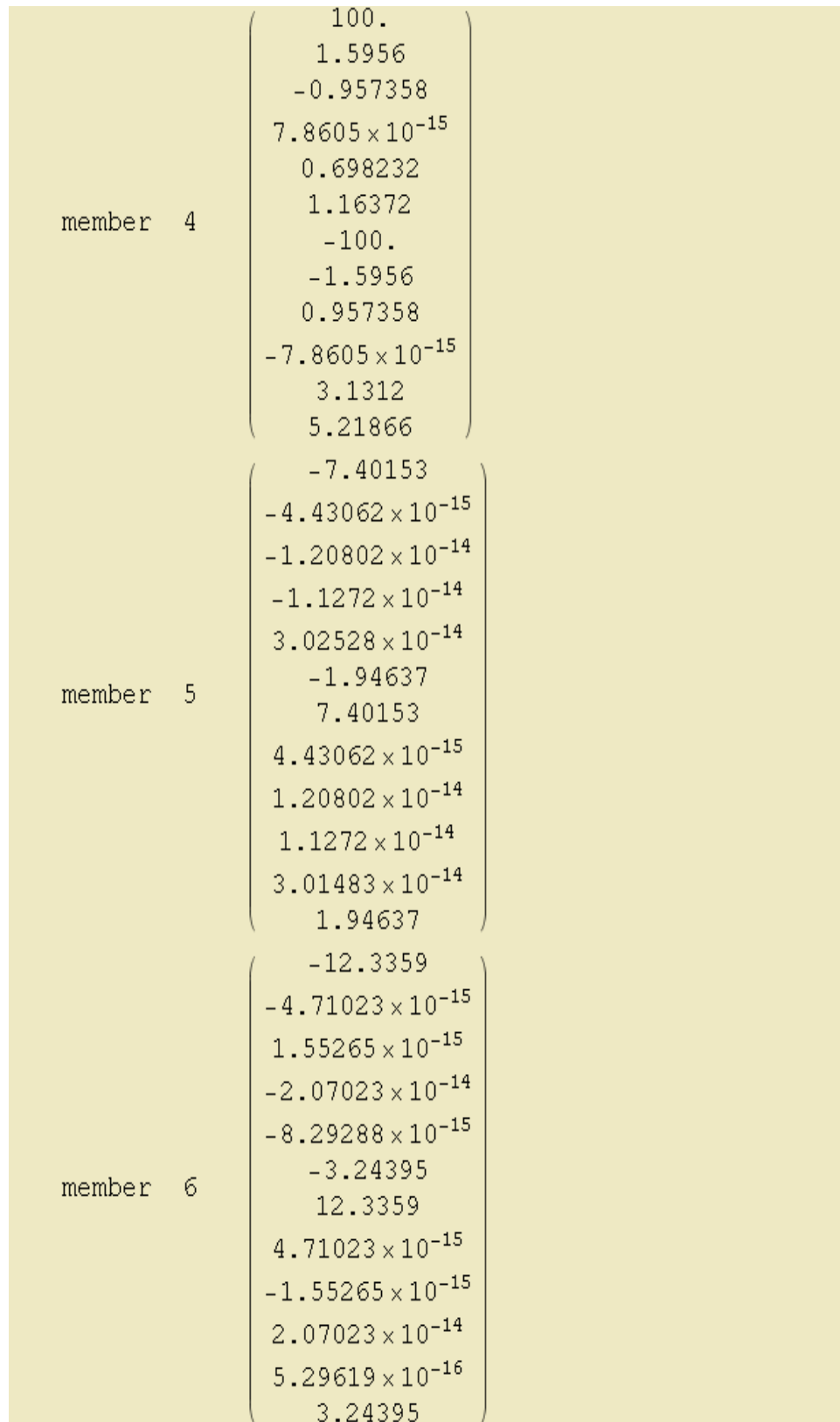


Figure 72: Member Forces - Ex.2 - (2)

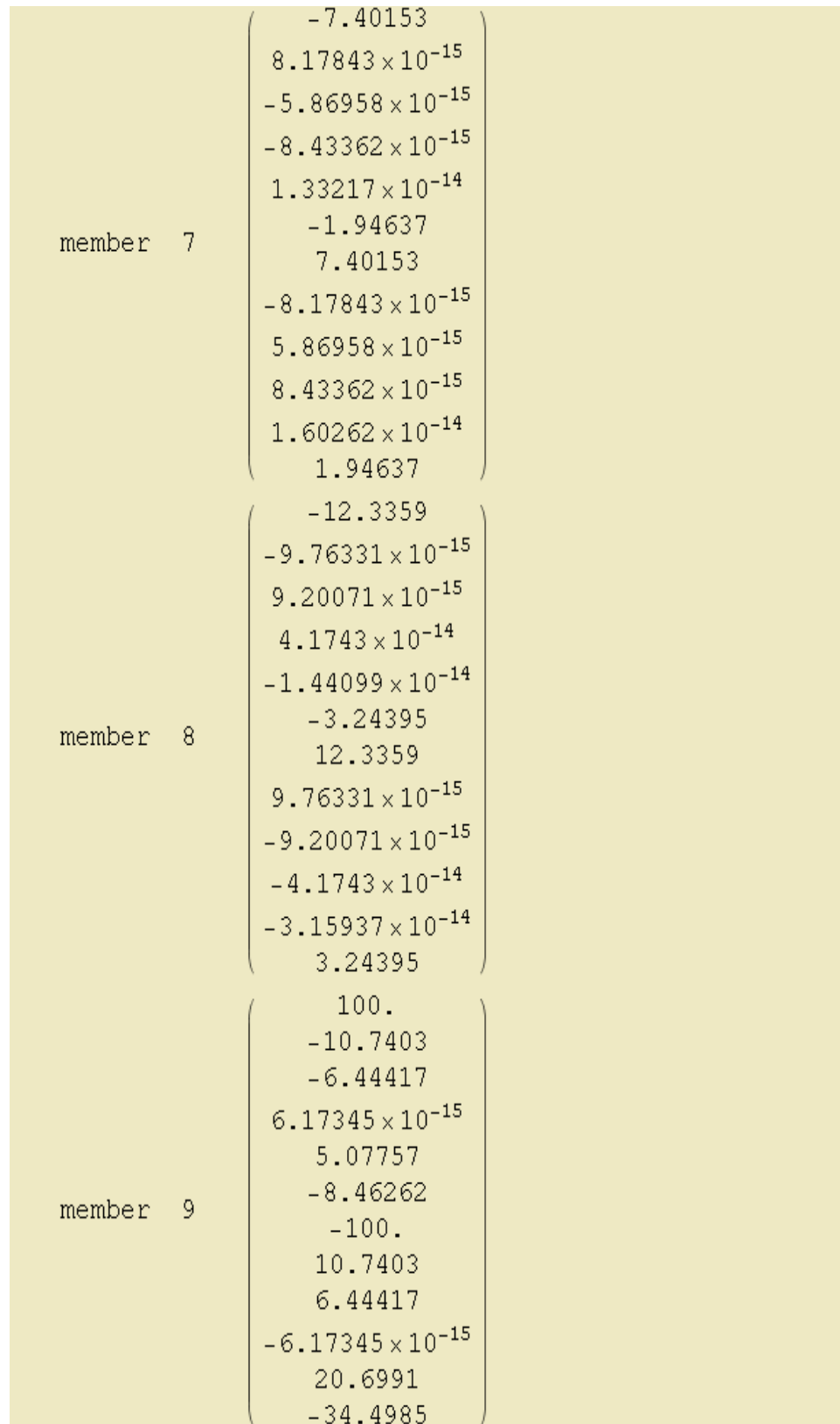


Figure 73: Member Forces - Ex.2 - (3)

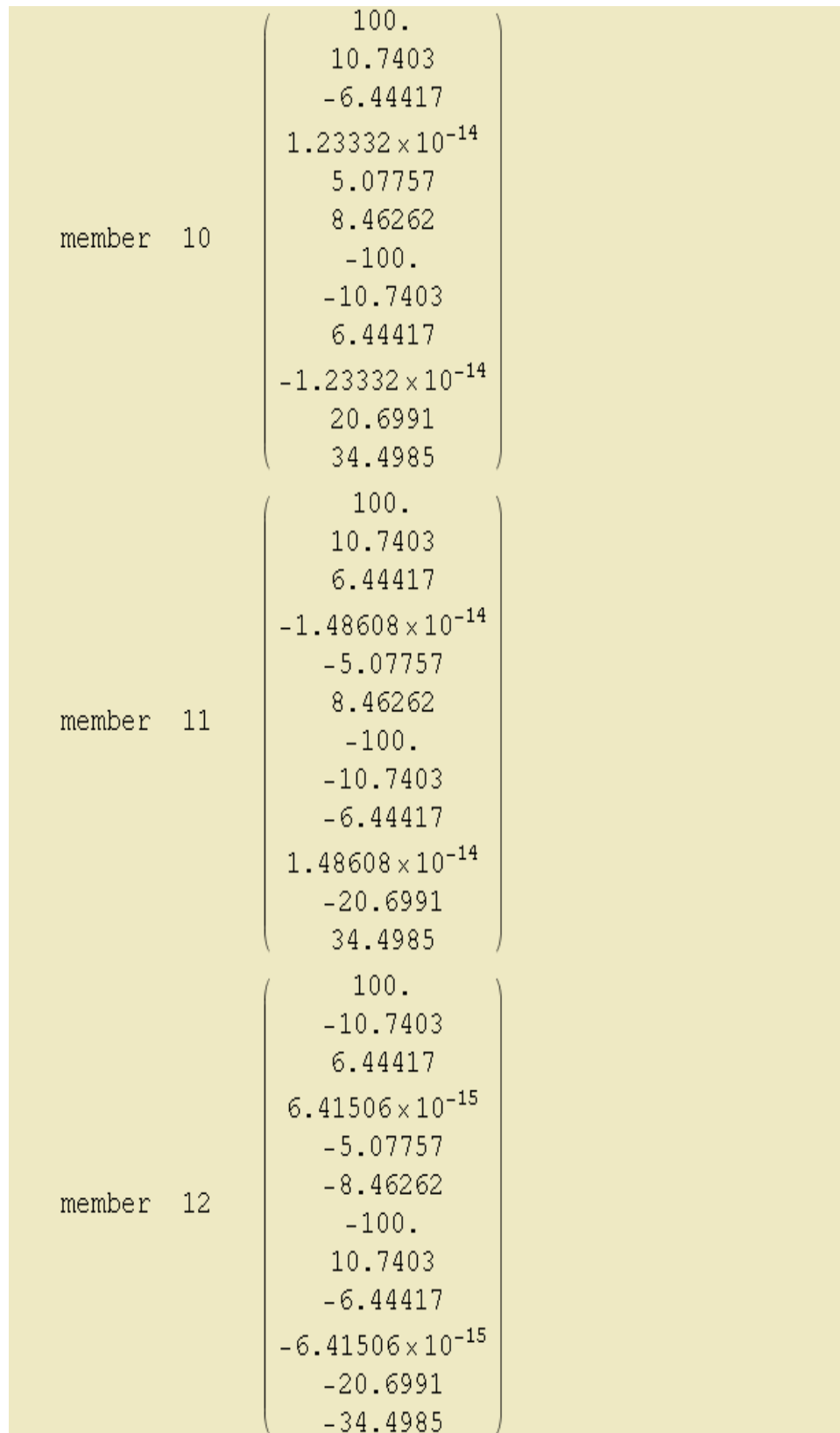


Figure 74: Member Forces – Ex.2 - (4)

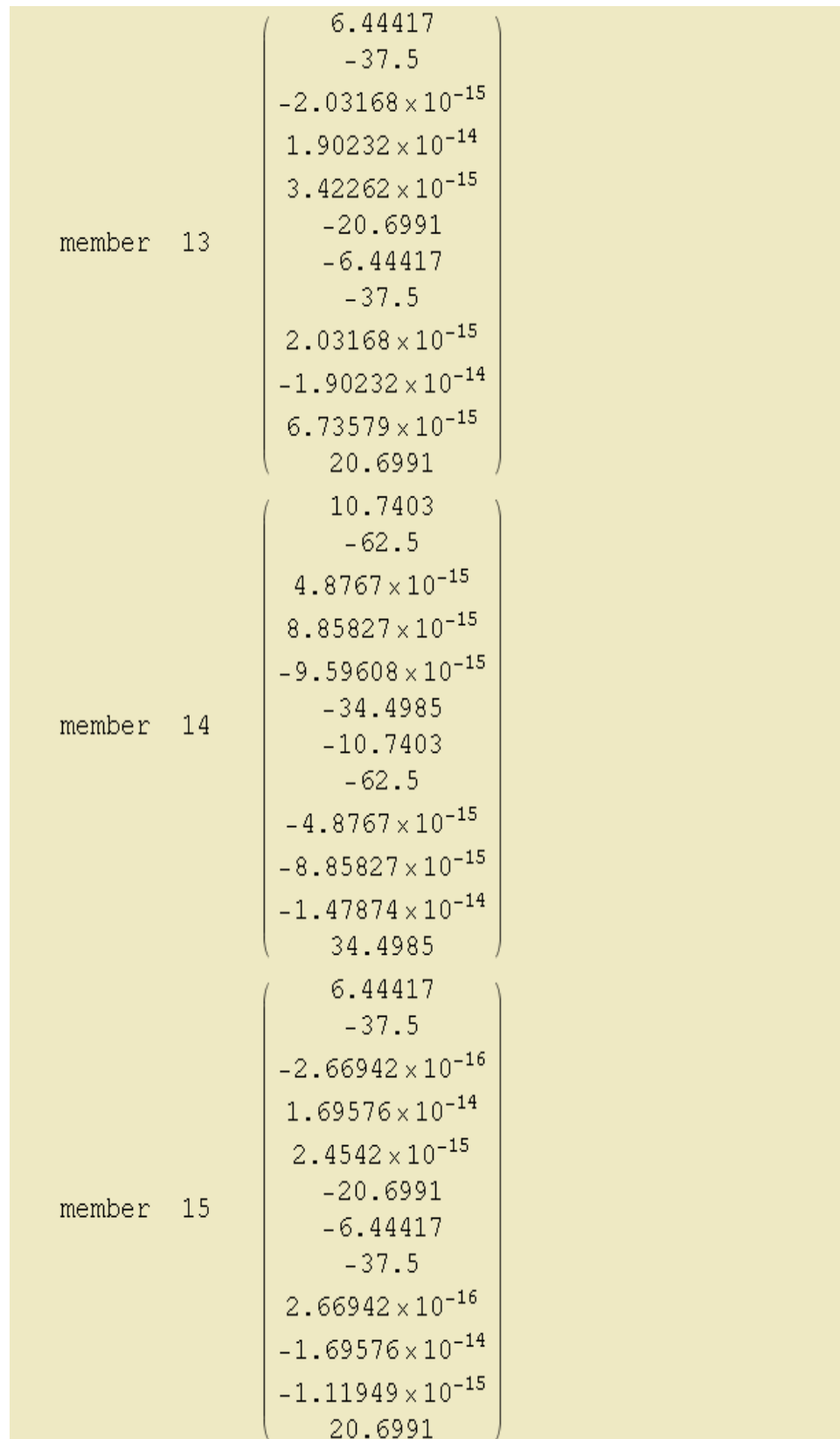


Figure 75: Member Forces – Ex.2 - (5)

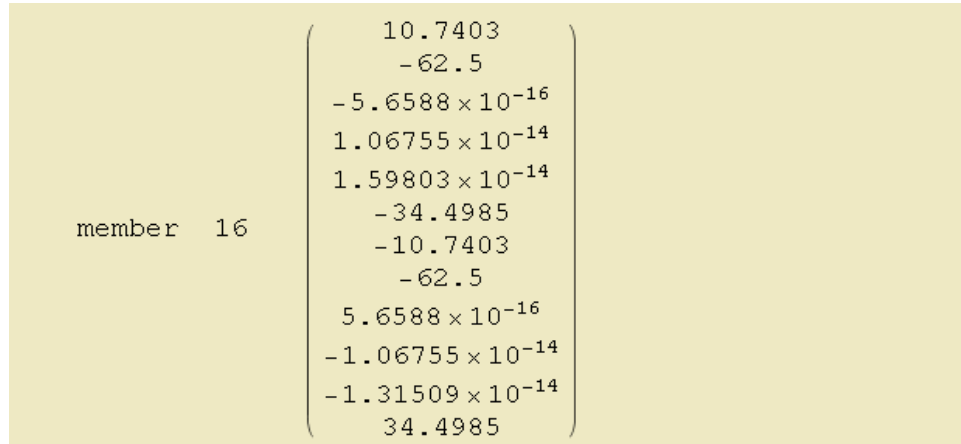


Figure 76: Member Forces – Ex.2 - (6)

6.3.1 Mastan's result of example 2:

```

*****      MASTAN2 v3.3.1      *****

Time:  02:49:23      Date:  11/18/2012

Problem Title:  not provided
*****

#####
Results of Structural Analysis
#####

General Information:
      Structure Analyzed as:  Space Frame
      Analysis Type:  First-Order Elastic

Analytical Results:

(i)  Displacements at Step # 1, Applied Load Ratio = 1.0000

Deflections
Node      X-disp      Y-disp      Z-disp
1         0.0000e+000    0.0000e+000    0.0000e+000
2         0.0000e+000    0.0000e+000    0.0000e+000
3         0.0000e+000    0.0000e+000    0.0000e+000
4         0.0000e+000    0.0000e+000    0.0000e+000
5        -4.6260e-005    -1.0000e-003    -7.7099e-005
6        -4.6260e-005    -1.0000e-003     7.7099e-005
7         4.6260e-005    -1.0000e-003     7.7099e-005
8         4.6260e-005    -1.0000e-003    -7.7099e-005
9         4.0276e-005    -2.0000e-003     6.7127e-005
10        4.0276e-005    -2.0000e-003    -6.7127e-005
11       -4.0276e-005    -2.0000e-003    -6.7127e-005
12       -4.0276e-005    -2.0000e-003     6.7127e-005

```

Rotations (radians)

Node	X-rot	Y-rot	Z-rot
1	0.0000e+000	0.0000e+000	0.0000e+000
2	0.0000e+000	0.0000e+000	0.0000e+000
3	0.0000e+000	0.0000e+000	0.0000e+000
4	0.0000e+000	0.0000e+000	0.0000e+000
5	-8.1099e-005	-1.7584e-020	4.8659e-005
6	8.1099e-005	-8.0353e-021	4.8659e-005
7	8.1099e-005	-2.0246e-020	-4.8659e-005
8	-8.1099e-005	-6.8300e-021	-4.8659e-005
9	4.3962e-004	-2.5424e-020	-2.6377e-004
10	-4.3962e-004	-1.2829e-020	-2.6377e-004
11	-4.3962e-004	-4.0246e-020	2.6377e-004
12	4.3962e-004	-1.8945e-020	2.6377e-004

(ii) Element Results at Step # 1, Applied Load Ratio = 1.0000

Internal End Forces (Note: Refers to local coordinates)

Element	Node	Fx	Fy	Fz
1	1	1.0000e+002	9.5736e-001	-1.5956e+000
	5	-1.0000e+002	-9.5736e-001	1.5956e+000
2	2	1.0000e+002	9.5736e-001	1.5956e+000
	6	-1.0000e+002	-9.5736e-001	-1.5956e+000
3	3	1.0000e+002	-9.5736e-001	1.5956e+000
	7	-1.0000e+002	9.5736e-001	-1.5956e+000
4	4	1.0000e+002	-9.5736e-001	-1.5956e+000
	8	-1.0000e+002	9.5736e-001	1.5956e+000
5	5	-7.4015e+000	1.3323e-015	-6.1324e-016
	8	7.4015e+000	-1.3323e-015	6.1324e-016
6	5	-1.2336e+001	3.1086e-015	6.3694e-016
	6	1.2336e+001	-3.1086e-015	-6.3694e-016
7	6	-7.4015e+000	-1.1102e-015	-9.5739e-016
	7	7.4015e+000	1.1102e-015	9.5739e-016
8	7	-1.2336e+001	8.8818e-016	4.4148e-016
	8	1.2336e+001	-8.8818e-016	-4.4148e-016
9	5	1.0000e+002	-6.4442e+000	1.0740e+001
	9	-1.0000e+002	6.4442e+000	-1.0740e+001
10	6	1.0000e+002	-6.4442e+000	-1.0740e+001
	10	-1.0000e+002	6.4442e+000	1.0740e+001
11	7	1.0000e+002	6.4442e+000	-1.0740e+001
	11	-1.0000e+002	-6.4442e+000	1.0740e+001
12	8	1.0000e+002	6.4442e+000	1.0740e+001
	12	-1.0000e+002	-6.4442e+000	-1.0740e+001
13	9	6.4442e+000	3.7500e+001	1.7869e-015
	12	-6.4442e+000	3.7500e+001	-1.7869e-015
14	9	1.0740e+001	6.2500e+001	-2.6342e-015
	10	-1.0740e+001	6.2500e+001	2.6342e-015
15	10	6.4442e+000	3.7500e+001	2.4092e-015
	11	-6.4442e+000	3.7500e+001	-2.4092e-015
16	11	1.0740e+001	6.2500e+001	-2.0988e-015
	12	-1.0740e+001	6.2500e+001	2.0988e-015

Internal End Moments (Note: Refers to local coordinates)

Element	Node	Mx	My	Mz
1	1	2.9994e-016	1.1637e+000	6.9823e-001
	5	-2.9994e-016	5.2187e+000	3.1312e+000
2	2	2.3486e-016	-1.1637e+000	6.9823e-001
	6	-2.3486e-016	-5.2187e+000	3.1312e+000
3	3	4.1881e-016	-1.1637e+000	-6.9823e-001
	7	-4.1881e-016	-5.2187e+000	-3.1312e+000
4	4	1.8385e-016	1.1637e+000	-6.9823e-001
	8	-1.8385e-016	5.2187e+000	-3.1312e+000
5	5	-5.4701e-016	1.2811e-015	1.9464e+000
	8	5.4701e-016	1.6740e-015	-1.9464e+000
6	5	-4.9805e-016	-1.6537e-015	3.2440e+000
	6	4.9805e-016	-1.3090e-015	-3.2440e+000
7	6	2.2331e-016	2.9649e-015	1.9464e+000
	7	-2.2331e-016	2.5992e-015	-1.9464e+000
8	7	-5.9760e-016	-1.1614e-015	3.2440e+000
	8	5.9760e-016	-7.6844e-016	-3.2440e+000
9	5	1.7205e-016	-8.4626e+000	-5.0776e+000
	9	-1.7205e-016	-3.4499e+001	-2.0699e+001
10	6	3.2332e-017	8.4626e+000	-5.0776e+000
	10	-3.2332e-017	3.4499e+001	-2.0699e+001
11	7	3.4400e-016	8.4626e+000	5.0776e+000
	11	-3.4400e-016	3.4499e+001	2.0699e+001
12	8	1.7398e-016	-8.4626e+000	5.0776e+000
	12	-1.7398e-016	-3.4499e+001	2.0699e+001
13	9	3.1115e-015	-4.5326e-015	2.0699e+001
	12	-3.1115e-015	-4.4022e-015	-2.0699e+001
14	9	-4.4858e-015	6.2592e-015	3.4499e+001
	10	4.4858e-015	6.7452e-015	-3.4499e+001
15	10	5.0881e-015	-5.3846e-015	2.0699e+001
	11	-5.0881e-015	-6.3283e-015	-2.0699e+001
16	11	-2.2956e-015	4.7933e-015	3.4499e+001
	12	2.2956e-015	5.5898e-015	-3.4499e+001

```
#####
End of Results of Structural Analysis
#####
```

Chapter 7

CONCLUSION

7.1 Introduction

In this research the method used for analyzing space frame is Integrated Force Method, which is independent of redundant selection process and also only one solution process is demanded for calculating the independent internal forces. These two advantages of the IFM renders it as a better option in designing the complex and large scale structures. Much precise stress results, an enhanced system intended for discrete finite element analysis, much quicker convergence to answers and capability of being applied in nonlinear analysis of structures and optimization problems are other advantages of IFM. (Patnaik S. N. , Hopkins D. A. , and Halford G. R. , 2004)

In analyzing space frames other generally applied method is IFMD. In generating global stiffness matrix only the equilibrium matrix and unconnected stiffness matrix are used. Therefore, the need for developing long and complex programming is avoided and global stiffness matrix is obtained through a much simpler programming in Mathematica version8.

In this research the equilibrium equations generated via computer code developed in Mathematica are used in both of displacement and force methods for analyzing the space frames.

In displacement method, the only approach employed is the Dual integrated force method. Regarding force method two approaches are employed which are null space and singular value decomposition methods.

7.2 Contributions

Aforementioned programs were developed:

1. For automation of equilibrium equation generation.
2. For analysis of space frames with 3 different methods.
3. For generation of $[K]$ matrix from equilibrium equation.

The characteristics of developed programs are as follows:

1. Flexibility in data input and also output style.
2. Elimination of any need for program manual.
3. Development of simple and easy to run programs.
4. Use of step by step calculation to make the value of variables traceable.

Advantages of programs are:

1. No restriction on the number of members and joints
2. Capability of utility change

7.3 Recommendation for Further Researches

The results of this study may be developed further or integrated with the following cases:

1. Addition of other codes that is capable of considering the thermal and settlement forces on the frame.
2. Development of a computer code that makes it possible to analyze the trusses and frames as single structure.

3. Since this program assumes uniform forces as perpendicular to XY plane, a more enhanced version of this code could have the capacity of applying other uniform forces with different orientations.
4. Computer codes that are especially developed for concrete or steel structures.
5. Dynamic Analysis

REFERENCES

- Company, N. I. (2012, 3 11). Space Frame, Lightweight Rigid Structure, Interlocking Struts, Double Layered Grids, Mumbai, India. 11 3, 2012 tarihinde New life steel company: <http://www.newlifesteel.net/space-frame.html> adresinden alındı
- Farajzadeh, H. (2012). Investigation of Space Truss Using the Integrated Force Method. Master Thesis . Famagusta, Northern Cyprus: Eastern Mediterranean University.
- Fillppou, F. C. (2001). Chapter 2. Berkley: University of California.
- Hopkins, D., & Patnaik, S. (2004). Strength Of Materials : A Unified Theory For The 21st Century. Butterworth-Heinemann.
- Kamkar S. (2010). Investigation of Rigid Frame by Integrated Force Method. Gazimagusa: Eastern Mediterranean University.
- Kassimali, A. (2010). Matrix Analysis of structures. USA: Cengage Learning.
- Khosravi S. (2005). Usage of equilibrium Equation in Truss Analysis. Gazimagusa: Eastern Mediterranean University.
- Krishnam Raju N. R. B. , and Nagabhushanam J. . (2000). Nonlinear structural analysis using integrated force method. Sadhana, , Vol. 25, Part 4, pp. 353-365.
- McGuire W. and Gallagher H.R. . (1979). Matrix Structural Analysis. New York: Wiley

- Movaghar, S. k. (2005). Usage of Equilibrium Equations in Truss Analysis. Master Thesis . Famagusta, Northern Cyprus: Eastern Mediterranean University.
- Patnaik S. N., Hopkins D. A. and Coroneos R. . (1996). Structural optimization with approximate sensitivities. *Computer and Structures*, vol. 58, no.2 , pp 407-418.
- Patnaik S. N. and Joseph K. T. . (1986). Generation of compatibility matrix in the integrated force method. *Comp. Meth. Appl. Mech. Eng.* , vol.55, no. 3, pp. 239-257.
- Patnaik S. N. , and Hopkins D. A. . (1998). Recent advance in the method of forces: Integrated force method of structural analysis. *Advances Engrg. Software* , vol.29, no. 3-6, pp. 463-474.
- Patnaik S. N. , Hopkins D. A. , and Halford G. R. . (2004). Integrated Force Method solution to indeterminate structural mechanics problem. NASA/TP-207430.
- Patnaik S. N. , Berke L. and Gallagher R. H. . (1991). Integrated force method versus displacement method for finite element analysis. *Computer and Structure* , vol.38, no.4, pp 377-407.
- Patnaik, S. (1999). Compatibility Condition in Structural Mechanics. NASA/TM-209175.
- Patnaik, S. N. (1986). Integrated force method versus the standard force method. *Computer and Structure* , vol. 22, no. 2, pp.151-163.
- Pellegrino S. (1993). Structural Computations with Singular Value Decomposition of the Equilibrium Matrix. *Int. J. Solid Structures* , Vol. 30, No.21, pp.3025-3035.

- Przemieniecki, J. S. (1968). Theory of Matrix Structural Analysis. New York: McGraw-Hill.
- Saouma, V. E. (1999). Matrix Structural Analysis with an Introduction to Finite Elements. University of Colorado, Boulder.
- Sedaghati R. . (2005). Benchmark Case Studies in Structural Design Optimization Using Integrated Force Method. International Journal of Solids and Structures 42 , pp 5848–5871.
- Sensoy, S. (1995). Two dimensional structural analysis by felexibility method. Master Thesis . Eastern Mediterranean University.
- Soyer E. (2001). A New Numerical Technique for the Generation of Sparse and banded Self-Stress Matrix by Using Groping of Redundants. Gazimagusa: Eastern Mediterranean University.
- Soyer E., and Topcu A. . (2001). Sparse Self-Stress Matrices for the Finite Element Force Method. Int. J. Numer. Meth. Engng, 50 , pp. 2175-2194.
- Wang, C. K. (1983). Indeterminate Structural Analysis. Civil Engineering Series. McGraw-Hill.
- West, H. H. (1993). Fundamental of structural Analysis. John Wiley & Sons.
- Ziemian R. D. , and McGuire W. . (2000). Matrix Structural Analysis 2 (MASTAN 2). John Willey & Sons.