Computational Mechanics Tools

GiD Homework

Problem-type

FRAME3DD

Submitted by
Karthik Neerala Suresh
MSc Computational Mechanics
Universitat Politécnica de Catalunya,
BarcelonaTECH

Submitted to
Mr. Enrique Escolano
Research Engineer, CIMNE
Universitat Politécnica de Catalunya,
BarcelonaTECH

13 January 2017

Contents

List of Figures

List of Tables

1 Introduction

GiD is a universal, adaptive and user-friendly pre and post processor for numerical simulations in science and engineering. It has been designed to cover all the common needs in the numerical simulations field from pre to post-processing such as geometrical modelling (CAD), mesh generation, definition of analysis data, data transfer to analysis software, postprocessing operations and visualisation of results. One of the features of GiD is customisation. In this project, the customisation feature of GiD is explored. Some of the features offered by GiD is creation of complete menu's that are customized to suit the specific needs of the user's simulation software and development of simple interfaces between the data definition and the simulation software. The customization in GiD is done by creating a Problem Type.

When GiD is to be used for a particular type of analysis, it is necessary to predefine all the information required from the user and to define the way the final information is given to the solver module. To do so, some files are used to describe conditions, materials, general data, units systems, symbols and the format of the input file for the solver. Since GiD has been designed to be a general-purpose pre- and postprocessor, the configurations for different analyses must be performed according to the particular specifications of each solver. It is therefore necessary to create specific data input files for every solver. However, GiD lets one perform this configuration process inside the program itself, without any change in the solver, and without having to program any independent utility. The name Problem Type is given to this collection of files used to configure GiD for a particular type of analysis. In this project, one is required to write a Problem Type to customize GiD to a simulation code named FRAME3DD, an open-source software for structural matrix calculation of frames.

The Problem Type should include files for problem data, conditions, materials, template and .bat that will help to generate an input file for the solver compiled for Windows (`frame3dd.exe`). The implementation must include definition of data for static analysis of 3D frames, with constraints, several load cases with loads on nodes and uniform and thermal loads on elements, self-weight and prescribed displacements on nodes, and the material properties. It should also be able to launch the calculation from the *Calculate*→*Calculate* menu and show its console output with *Calculate*→*View process info*.

This report is organised in the following manner. In Chapter 2, the method followed while writing the various files of the Problem Type is explained in detail. Subsequently in Chapter 3, three examples solved by making use of the written Problem Type files and the solver `frame3dd.exe` are described along with their results. The results obtained by inputting the GiD generated input fire to the solver are in agreement with already computed results obtained by inputting the default input file to the solver.

2 Problem Type files

The creation of a Problem Type involves the creation of a directory with the name of the Problem Type and the extension .gid. This directory can be located in the current working directory or the main GiD executable directory. The series of files must be inside the problem type directory. The name for most of them will follow the format `problem_type_name.xxx` where the extension refers to their particular function. There are three types of files that one needs to generate in this process of customisation, namely, configuration files which will help one assign materials and conditions, problem and interval data etc., template files which will result in the generation of the input file for the solver and command execution files which will call the solver from within GiD and run it with the input file generated to get the required output file. Finally Tcl extension files are extensions to GiD written in Tcl/Tk programming language that will read the output file and implement the automatic conversion of results to GiD post-process format. In this project, one is required to write configuration, template and command execution files that will create the input file for the solver `frame3dd.exe`, invoke the solver from within GiD and generate the output file. Tcl extension file to implement the conversion og results to GiD post-process format has already been provided. In this chapter, the files written will be explained.

First of all, the subdirectory `frame3dd.gid` is created. This subdirectory has a `.gid` extension and will contain all the configuration files, template files and calculating module files (`.prb`, `.mat`, `.cnd`, `.bas`, `.bat`, `.exe`). In sections 2.1, **??** and 2.3, each of these files are explained.

2.1 Configuration files

These files generate the conditions and material properties, as well as the general problem and intervals data to be transferred to the mesh, at the same time giving one the chance to define geometrical drawings or symbols to represent some conditions on the screen. Three different configuration files are required namely, `.prb` file to configure general parameters and interval data, `.mat` file for the configuration of materials and their properties and `.cnd` file for the configuration of the conditions imposed on the calculation.

2.1.1 Materials file

The materials file `frame3dd.mat` is created. This file stores the physical properties of the material under study for the problem type. In this case, a number of material properties need to be defined such as area of cross section, shear areas, torsional and bending moments of inertia, Young's modulus, rigidity modulus, roll and density.

The materials file is written as follows

```
MATERIAL: 2D_FRAME
QUESTION: Ax
VALUE: 10.0
QUESTION: Asy
VALUE: 1.0
QUESTION: Asz
VALUE: 1.0
QUESTION: Jxx
VALUE: 1.0
QUESTION: Iyy
VALUE: 1.0
QUESTION: Izz
VALUE: 0.01
QUESTION: E
VALUE: 29000
QUESTION: G
VALUE: 11500
QUESTION: ROLL
VALUE: 0
QUESTION: DENSITY
VALUE: 7.33E-7
END MATERIAL
```

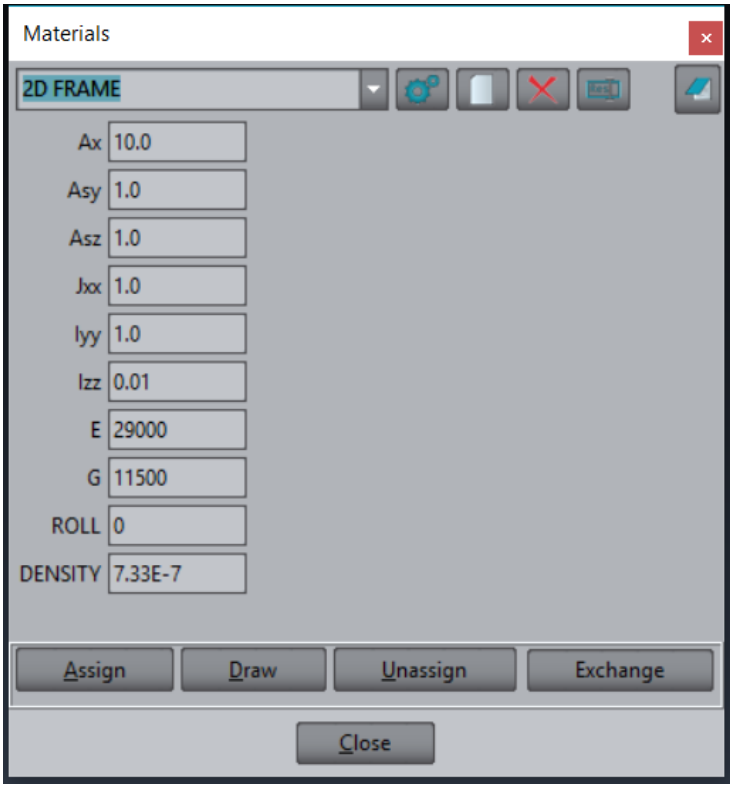


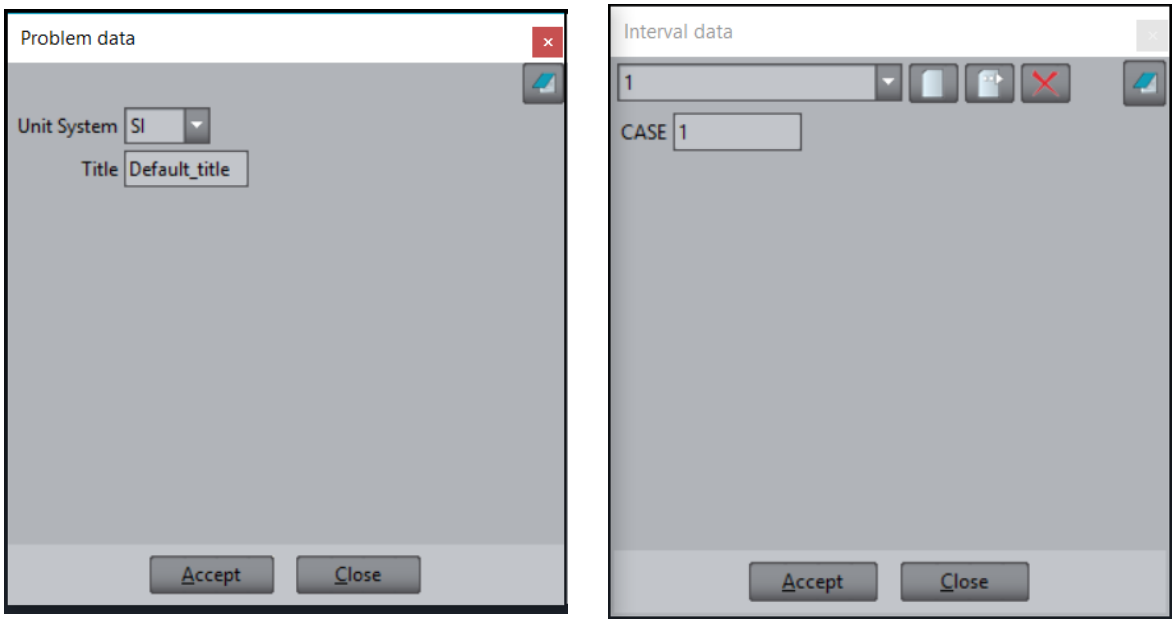
Figure 2.1: The GiD Materials window

2.1.2 General file

The general file `frame3dd.prb` is created. This file contains general information for the calculating module, such as the units system for the problem, or the type of resolution algorithm chosen. Also this file contains interval data. Within this data, one may consider the definition of specific problem data (for the whole process) and intervals data (variable values along the different solution intervals). An interval would be the subdivision of a general problem that contains its own particular data. In the problems considered in this project, interval data facilitates in the definition of a different load case for every interval.

The general file is written as

```
PROBLEM DATA
QUESTION:
Unit_System#CB#(SI,CGS,User)
VALUE: SI
QUESTION: Title
VALUE: Default_title
END PROBLEM DATA
INTERVAL DATA
QUESTION: CASE
VALUE: 1
END INTERVAL DATA
```



(a) (b)

Figure 2.2: The GiD problem (a) and interval (b) data window

2.1.3 Conditions file

The conditions file `frame3dd.cnd` is created. Files with extension `.cnd` contain all the information about the conditions that can be applied to different entities. The condition can adopt different field values for every entity. This type of information includes, for instance, all the displacement constraints and applied loads in a structural problem or all the prescribed and initial temperatures in a thermal analysis.

An important characteristic of the conditions is that they must define what kind of entity they are going to be applied over, i.e. over points, over lines, over surfaces, over volumes, over layers or over groups, and what kind of mesh entity they will be transferred over, i.e. over nodes, over face elements or over body elements.

The conditions file is written as follows. Here only two of the many conditions that can be applied are showed, one each applied over nodes and elements. In the problems considered, there are a number of conditions to be applied. These conditions are specified in the conditions file in the same format as shown below. Conditions of fixed nodes, point loads, points with extra inertia and prescribed displacement are applied over points, while those of uniform loads, trapezoidal loads, internal concentrated loads, temperature loads and elements with extra mass are assigned to elements. In addition to one of the nodes, gravitational acceleration is assigned to consider the effect of self-weight. If no self weight is to be considered, the acceleration is assigned as zero. Though this acceleration is assigned only to one node, the solver will use this information only to recognise whether self weight has to be considered or not during the analysis.

CONDITION: Fixed-Nodes
CONDTYPE: over points
CONDMESHTYPE: over nodes
QUESTION: X
VALUE: 1
QUESTION: Y
VALUE: 1
QUESTION: Z
VALUE: 1
QUESTION: XX
VALUE: 1
QUESTION: YY
VALUE: 1
QUESTION: ZZ
VALUE: 1
END CONDITION

CONDITION: Uniform-Load
CONDTYPE: over lines
CONDMESHTYPE: over body elements
QUESTION: Ux
VALUE: 0
QUESTION: Uy
VALUE: 0
QUESTION: Uz
VALUE: 0
END CONDITION

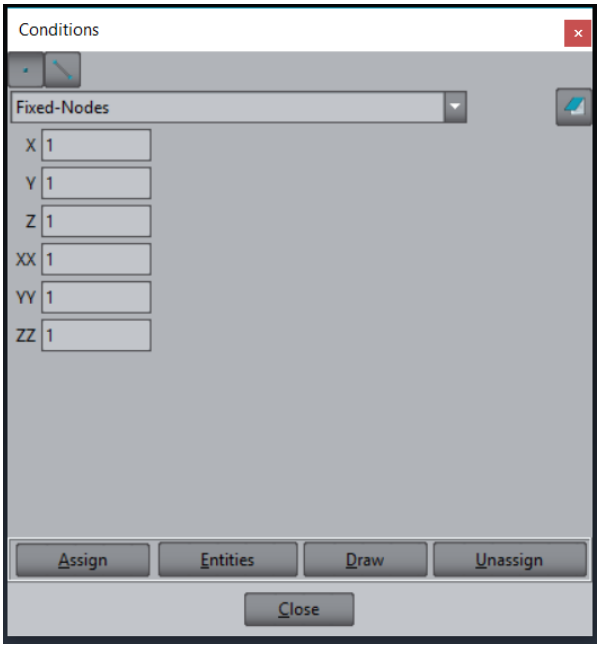


Figure 2.3: The GiD Conditions window

2.2 Template file

Once one has generated the mesh, and assigned the conditions and the materials properties, as well as the general problem and intervals data for the solver, the data input files to be processed by that program needs to be produced.

To manage this reading, GiD is able to interpret a file called `problem_type_name.bas`. In this project, a file called `frame3dd.bas` is created. This file works as an interface from GiD's standard results to the specific data input for any individual solver module. This means that the process of running the analysis simply forms another step that can be completed within the system. This file will define the format of the `.dat` text file created by GiD. It will store the geometric and physical data of the problem. The `.dat` file will be the input to the calculating module.

`frame3dd.bas` has to create a `.dat` file that will be able to read by the solver `frame3dd.exe`. See Appendix for the detailed file.

2.3 Command execution file

The file `frame3dd.bat` is created. This file connects the data file(s) (.dat) to the calculating module (the `frame3dd.exe` program). When the GiD Calculate option is selected, it executes the `.bat` file for the problem type selected.

The command execution file is written as follows.

```
rem OutputFile: %2\%1.log
rem ErrorFile: %2\%1.err

rename %1.dat %1.3dd

del %2\%1.post.res
del %2\%1.out

%3\frame3dd.exe -i %1.3dd -o %1.out -x > %1.log 2 > %1.err
```

Here the line `rem OutputFile : %2\%1.log` results in the creation of the file `project_name.log`, which has the data that appears on the *View process info* console. The line `rem ErrorFile : %2\%1.err` results in the creation of the file `project_name.err`, which records the errors, if any during a given run. Since the solver `frame3dd.exe` needs `.3dd` file as input, the line `rename %1.dat %1.3dd` renames `project_name.dat` to `project_name.3dd`. The next two lines will result in the deletion of the output file and the `.post.res` file from previous runs. The final line `%3\frame3dd.exe -i%1.3dd -o %1.out -x > %1.log2 > %1.err` will call the solver. The file `project_name.3dd` is given as input while the solver will return the file `project_name.out` as the output.

3 Problems solved

Three problems were solved using the built customisation to demonstrate their working. The examples include a 2-D truss frame, a frame for a pyramid and one for a triangular tower. In each of the cases, the model was built in GiD. Then, material, interval data and conditions corresponding to each interval was assigned and the problem is solved for the given model from within GiD by invoking the solver. The solution file in `.out` format is converted into `.post.res` format by `frame3dd.tcl` file and the post processed data was visualised.

3.1 2-D truss frame

3.1.1 Geometry

This frame comprises of 12 nodes and 21 elements as shown in Figure 3.1

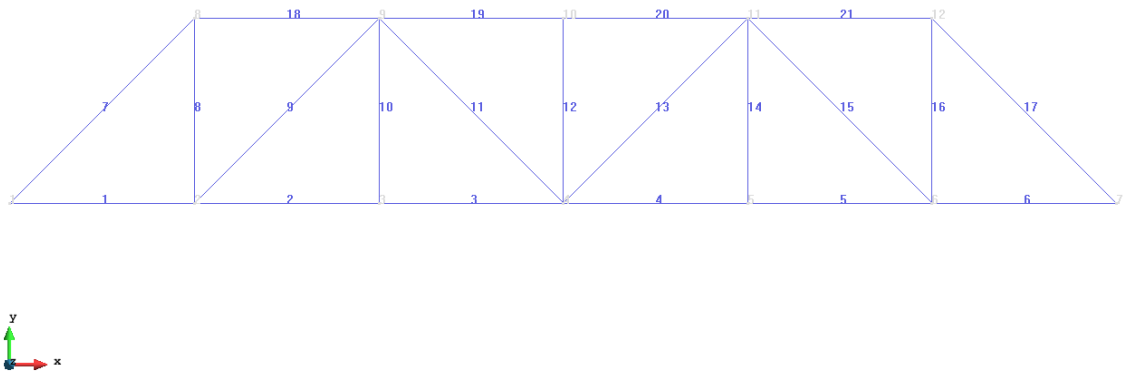


Figure 3.1: The geometry of the 2-D truss frame

The nodal coordinates and geometry are tabulated in Tables 3.1 and 3.2. The area of cross section of every element is equal to 10 mm^2, shear areas in y and z directions are both 1 mm^2, torsional moment of inertia is 1 mm^4 while moments of inertia for bending in local y and z directions, respectively, are 1 mm^4 and 0.1 mm^4. Every element is made of material with Young's modulus and rigidity modulus being respectively equal to 29000 and 11500 Mpa. The roll of every element is equal to 0 while the density is equal to 7.33×10^{-7} T/mm^3.

Table 3.1: Coordinated of nodes

Nodes	x (mm)	y (mm)	z (mm)
1	720	0	0
2	600	0	0
3	600	120	0
4	480	0	0
5	480	120	0
6	360	0	0
7	360	120	0
8	240	0	0
9	240	120	0
10	120	0	0
11	120	120	0
12	0	0	0

Table 3.2: Connectivities

Ele	N1	N2	Ele	N1	N2
1	12	10	12	6	7
2	10	8	13	6	5
3	8	6	14	4	5
4	6	4	15	2	5
5	4	2	16	2	3
6	2	1	17	1	3
7	12	11	18	11	9
8	10	11	19	9	7
9	10	9	20	7	5
10	8	9	21	5	3
11	6	9			

The truss is subjected to the following conditions.

3.1.2 Fixed Nodes

Table 3.3 gives the list of nodes and their displacements and rotations that are fixed. In the table columns x, y and z correspond to fixed displacements in the corresponding directions, while xx, yy and zz correspond to fixed rotations about the corresponding axes. A value of 1 corresponds to active fixed condition while 0 stands for free condition.

Table 3.3: Fixed nodes

Node	x	y	z	xx	yy	zz
1	0	1	1	1	1	0
2	0	0	1	1	1	0
3	0	0	1	1	1	0
4	0	0	1	1	1	0
5	0	0	1	1	1	0
6	0	0	1	1	1	0
7	0	0	1	1	1	0
8	0	0	1	1	1	0
9	0	0	1	1	1	0
10	0	0	1	1	1	0
11	1	0	1	1	1	0
12	1	1	1	1	1	0

3.1.3 Load Cases

There are two load cases. These are applied on the truss in two different intervals. In the first case point loads are applied on five nodes and a prescribed displacement is imposed

on one node. In the second case point loads are applied on three nodes, temperature loads on three elements and prescribed displacement on two nodes. These are tabulated in Tables 3.4, 3.5, 3.6, 3.7 and 3.8.

Load Case 1

Table 3.4: Point loads

Node	Fx (N)	Fy (N)	Fz (N)	Mxx (N mm)	Myy (N mm)	Mzz (N mm)
2	0	-20	0	0	0	0
4	0	-10	0	0	0	0
6	0	-20	0	0	0	0
8	0	-20	0	0	0	0
10	0	-10	0	0	0	0

Table 3.5: Prescribed displacement

Node	Dx (in)	Dy (in)	Dz (in)	Dxx (rad)	Dyy (rad)	Dzz (rad)
11	0.1	0	0	0	0	0

Load Case 2

Table 3.6: Point loads

Node	Fx (N)	Fy (N)	Fz (N)	Mxx (N mm)	Myy (N mm)	Mzz (N mm)
4	20	0	0	0	0	0
6	10	0	0	0	0	0
8	20	0	0	0	0	0

Table 3.7: Temperature loads

Element	α ($^oC^{-1}$)	hy (mm)	hz (mm)	Ty+ (^oC)	Ty- (^oC)	Tz+ (^oC)	Tz- (^oC)
10	6×10^{-12}	5	5	10	10	10	10
13	6×10^{-12}	5	5	15	15	15	15
15	6×10^{-12}	5	5	17	17	17	17

Table 3.8: Prescribed displacement

Node	Dx (in)	Dy (in)	Dz (in)	Dxx (rad)	Dyy (rad)	Dzz (rad)
11	0.1	0	0	0	0	0
12	0	-1.0	0	0	0	0

3.1.4 Results

Upon calculating from within GiD using the *Calculate→Calculate* menu, the console output is as shown below.

```
FRAME3DD version: 20100105
Analysis of 2D and 3D structural frames with elastic and geometric stiffness.
http://frame3dd.sf.net
GPL Copyright (C) 1992-2010, Henri P. Gavin
This is free software with absolutely no warranty.
For details, see the GPL license file, LICENSE.txt
** Example B: a pyramid-shaped frame — static and dynamic analysis (N mm ton)
**
number of joints ..... nJ = 12 ... complete
number of joints with reactions ..... nR = 12 ... complete
number of frame elements..... nE = 21 ... complete
number of load cases ..... nL = 2
load case 1 of 2:
number of loaded joints ..... nF = 5
number of uniformly distributed loads ..... nU = 0
number of trapezoidally distributed loads ..... nW = 0
number of concentrated frame element point loads .. nP = 0
number of temperature changes ..... nT = 0
number of prescribed displacements ..... nD = 1
load case 2 of 2:
number of loaded joints ..... nF = 3
number of uniformly distributed loads ..... nU = 0
number of trapezoidally distributed loads ..... nW = 0
number of concentrated frame element point loads .. nP = 0
number of temperature changes ..... nT = 3
number of prescribed displacements ..... nD = 2
load data ... complete
number of dynamic modes ..... nM = 0
mass data ... complete
Load Case 1 of 2 ...
Linear Elastic Analysis ... Mechanical Loads
RMS relative equilibrium precision: 1.102e-014 ... hoo-ray!
Load Case 2 of 2 ...
Linear Elastic Analysis ... Temperature Loads
Linear Elastic Analysis ... Mechanical Loads
RMS relative equilibrium precision: 3.509e-014 ... woo-hoo!
```

The results are stored in the file `Project2DFrameFullCase.out`. This file exactly resembles the file `exA.out`, which is obtained on running the solver using the default input file `exA.3dd`, thus confirming the accuracy of the written customization files. Upon postprocessing the results obtained for magnitude of displacement and reactions are shown in Figures 3.2 and 3.3.

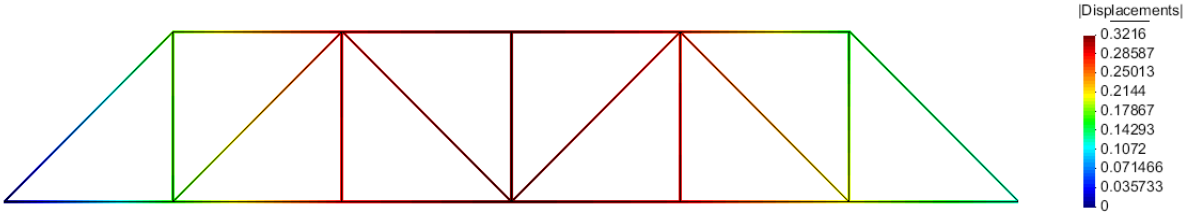


Figure 3.2: Magnitude of displacement

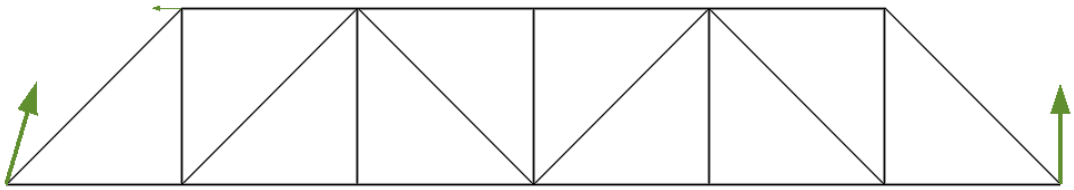


Figure 3.3: Reactions at nodes

3.2 Pyramid frame

3.2.1 Geometry

This frame comprises of 5 nodes and 4 elements as shown in Figure **??**

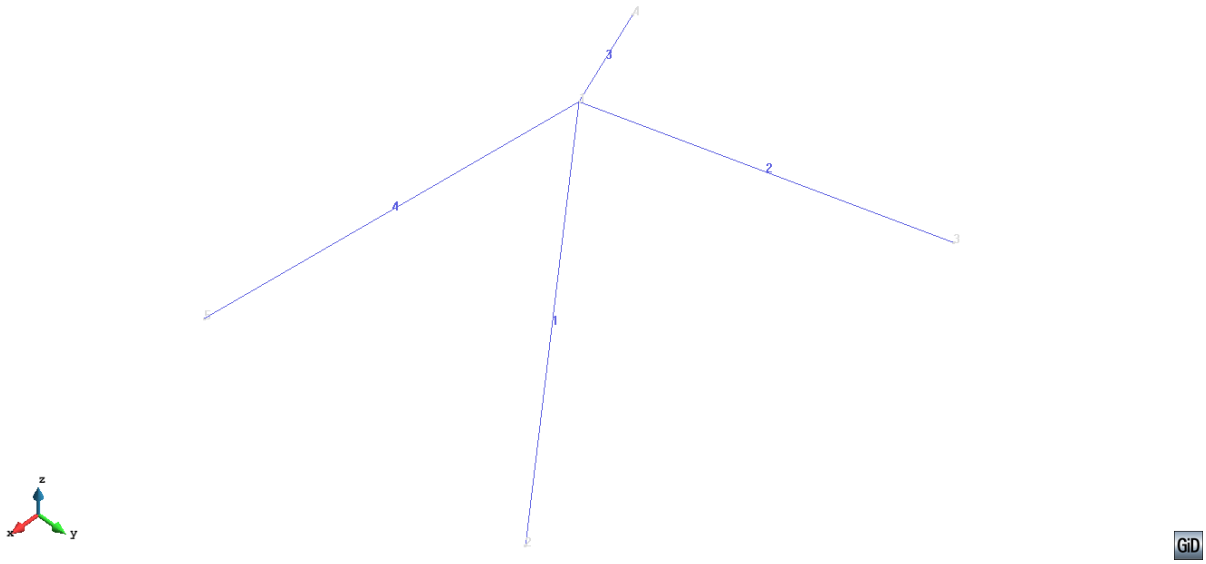


Figure 3.4: The geometry of the pyramid frame

The nodal coordinates and geometry are tabulated in Tables 3.9 and 3.10. The area of cross section of every element is equal to 36 mm^2, shear areas in y and z directions are both 20 mm^2, torsional moment of inertia is 1000 mm^4 while moments of inertia for bending in local y and z directions are both 492 mm^4. Every element is made of material with

Young's modulus and rigidity modulus being respectively equal to 200000 and 79300 Mpa. The roll of every element is equal to 0 while the density is equal to 7.85×10^{-9} T/mm^3.

Table 3.9: Coordinated of nodes

Nodes	x (mm)	y (mm)	z (mm)
1	-1200	900	0
2	-1200	-900	0
3	0	0	1000
4	1200	900	0
5	1200	-900	0

Table 3.10: Connectivities

Element	Node 1	Node 2
1	2	3
2	3	5
3	3	4
4	1	3

The truss is subjected to the following conditions.

3.2.2 Fixed Nodes

Table 3.11 gives the list of nodes and their displacements and rotations that are fixed. In the table columns x, y and z correspond to fixed displacements in the corresponding directions, while xx, yy and zz correspond to fixed rotations about the corresponding axes. A value of 1 corresponds to active fixed condition while 0 stands for free condition.

Table 3.11: Fixed nodes

Node	x	y	z	xx	yy	zz
1	1	1	1	1	1	1
2	1	1	1	1	1	1
4	1	1	1	1	1	1
5	1	1	1	1	1	1

3.2.3 Load Cases

There are three load cases. These are applied on the truss in three different intervals. In the first case point loads are applied on one nodes. In the second case uniform loads are applied on two elements, trapezoidal loads are applied on other two elements and temperature loads on one element. In the third load case only internally concentrated loads are applied on a particular location of two elements These are tabulated in Tables 3.12, 3.13, 3.14, 3.15 and 3.16.

Load Case 1

Table 3.12: Point loads

Node	Fx (N)	Fy (N)	Fz (N)	Mxx (N mm)	Myy (N mm)	Mzz (N mm)
3	100	-200	-100	0	0	0

Load Case 2

Table 3.13: Uniform loads

Element	Ux (N/mm)	Uy (N/mm)	Uz (N/mm)
1	0	0	0.1
2	0	0.1	0

Table 3.14: Trapezoidal loads

Ele	x1 mm	x2 mm	wx1 N/mm	wx2 N/mm	y1 mm	y2 mm	wy1 N/mm	wy2 N/mm	z1 mm	z2 mm	wz1 N/mm	wz2 N/mm
3	20	80	0.01	0.05	0	0	0	0	80	830	-0.05	0.07
4	0	0	0	0	68	330	0.05	0.00	80	830	-0.05	0.07

Table 3.15: Temperature loads

Element	α ($^oC^{-1}$)	hy (mm)	hz (mm)	Ty+ (^oC)	Ty- (^oC)	Tz+ (^oC)	Tz- (^oC)
1	12×10^{-6}	10	10	20	10	10	-10

Load Case 3

Table 3.16: Internal concentrated loads

Element	Px (mm)	Py (mm)	Pz (mm)	x (mm)
1	0	100	-900	600
2	0	-200	200	800

In addition to these, in all the three cases, self weight of the elements is considered. For self weight calculation, value of gravitational acceleration in the three directions is taken as (0,0,-9806.33) mm/s^2. Also node 3 is assumed to have extra inertia, with a mass of 0.1 ton causing this extra inertia.

3.2.4 Results

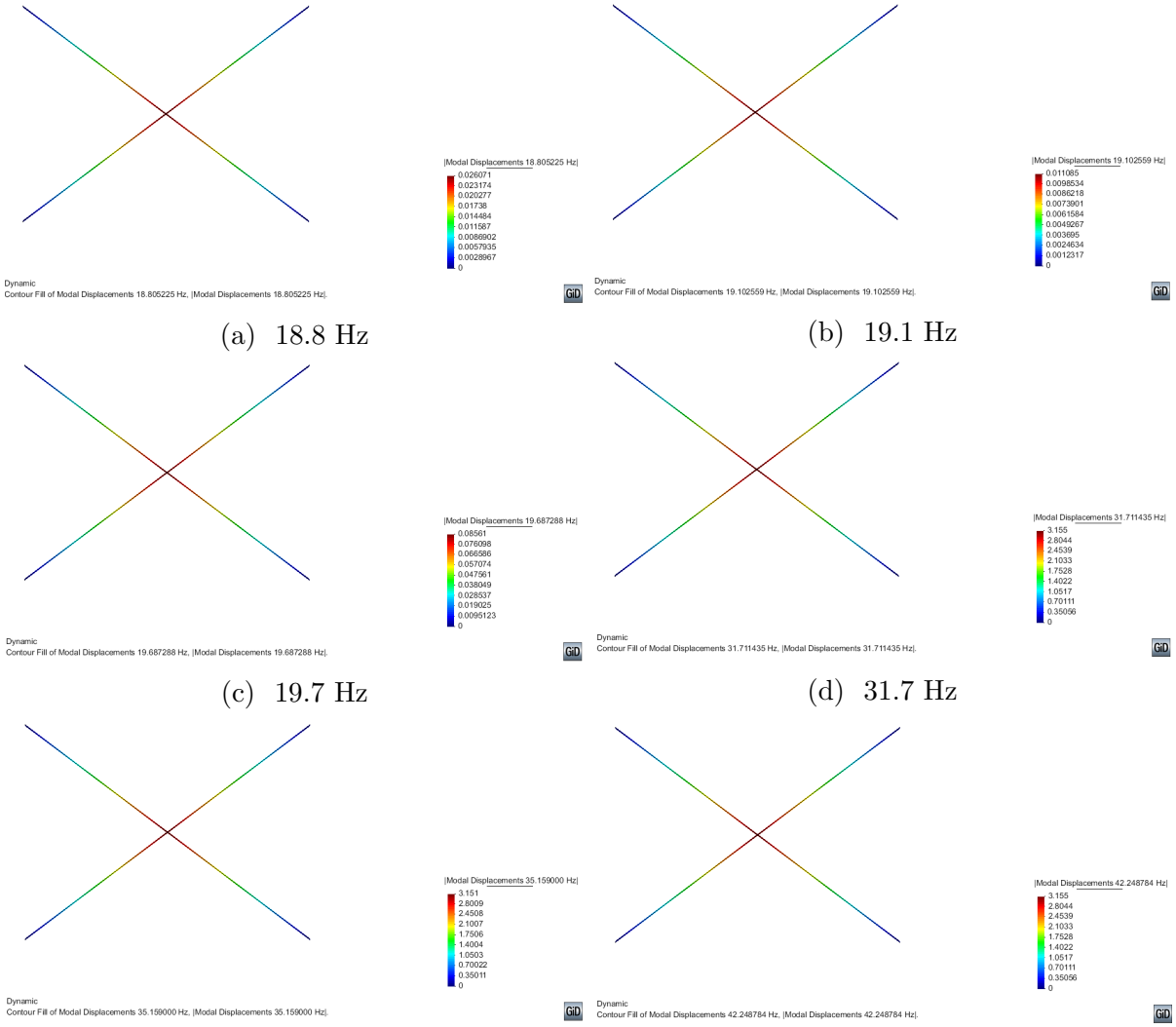
Upon calculating from within GiD using the *Calculate→Calculate* menu, the console output is as shown below.

```
FRAME3DD version: 20100105
Analysis of 2D and 3D structural frames with elastic and geometric stiffness.
http://frame3dd.sf.net
GPL Copyright (C) 1992-2010, Henri P. Gavin
```

```
This is free software with absolutely no warranty.
For details, see the GPL license file, LICENSE.txt
** Example B: a pyramid-shaped frame — static and dynamic analysis (N mm ton)
**
number of joints ..... nJ = 5 ... complete
number of joints with reactions ..... nR = 4 ... complete
number of frame elements..... nE = 4 ... complete
number of load cases ..... nL = 3
load case 1 of 3:
number of loaded joints ..... nF = 1
number of uniformly distributed loads ..... nU = 0
number of trapezoidally distributed loads ..... nW = 0
number of concentrated frame element point loads .. nP = 0
number of temperature changes ..... nT = 0
number of prescribed displacements ..... nD = 0
load case 2 of 3:
number of loaded joints ..... nF = 0
number of uniformly distributed loads ..... nU = 2
number of trapezoidally distributed loads ..... nW = 2
number of concentrated frame element point loads .. nP = 0
number of temperature changes ..... nT = 1
number of prescribed displacements ..... nD = 0
load case 3 of 3:
number of loaded joints ..... nF = 0
number of uniformly distributed loads ..... nU = 0
number of trapezoidally distributed loads ..... nW = 0
number of concentrated frame element point loads .. nP = 2
number of temperature changes ..... nT = 0
number of prescribed displacements ..... nD = 0
load data ... complete
number of dynamic modes ..... nM = 6
modal analysis method ..... 1 (Subspace-Jacobi)
number of joints with extra lumped inertia ..... nI = 1
number of frame elements with extra mass ..... nX = 0
structural mass ..... 2.0379e-003
total mass ..... 1.0204e-001
number of modes to be animated ..... nA = 6
mass data ... complete
Load Case 1 of 3 ...
Linear Elastic Analysis ... Mechanical Loads
Non-Linear Elastic Analysis ...
NR iteration 1 — RMS equilibrium precision: 3.70e-003
NR iteration 2 — RMS equilibrium precision: 1.09e-007
NR iteration 3 — RMS equilibrium precision: 2.94e-012
RMS relative equilibrium precision: 2.117e-005 ... up to snuff!
```

```
Load Case 2 of 3 ...
Linear Elastic Analysis ... Temperature Loads
Linear Elastic Analysis ... Mechanical Loads
Non-Linear Elastic Analysis ...
NR iteration 1 — RMS equilibrium precision: 3.72e-002
NR iteration 2 — RMS equilibrium precision: 8.90e-005
NR iteration 3 — RMS equilibrium precision: 1.89e-007
NR iteration 4 — RMS equilibrium precision: 2.53e-010
RMS relative equilibrium precision: 9.528e-006 ... outstanding!
Load Case 3 of 3 ...
Linear Elastic Analysis ... Mechanical Loads
Non-Linear Elastic Analysis ...
NR iteration 1 — RMS equilibrium precision: 6.24e-002
NR iteration 2 — RMS equilibrium precision: 1.41e-003
NR iteration 3 — RMS equilibrium precision: 4.28e-006
NR iteration 4 — RMS equilibrium precision: 3.67e-008
NR iteration 5 — RMS equilibrium precision: 1.13e-009
NR iteration 6 — RMS equilibrium precision: 1.01e-011
RMS relative equilibrium precision: 1.814e-006 ... yipee!
Modal Analysis ...
3 sub-space iterations, error: 3.8697e-012
mode: 1 DoF: 17 18.8052 Hz
mode: 2 DoF: 18 19.1026 Hz
mode: 3 DoF: 16 19.6873 Hz
mode: 4 DoF: 14 31.7114 Hz
mode: 5 DoF: 15 35.1590 Hz
mode: 6 DoF: 13 42.2488 Hz
mode: 7 DoF: 30 192.2659 Hz
mode: 8 DoF: 29 192.2659 Hz
mode: 9 DoF: 28 192.2659 Hz
mode: 10 DoF: 27 192.2659 Hz
mode: 11 DoF: 26 194.4523 Hz
mode: 12 DoF: 25 194.7445 Hz
There are 6 modes below 42.248784 Hz. All 6 modes were found.
```

The results are stored in the file `ProjectPyramidFrameFullCase2.out`. This file exactly resembles the file `exB.out`, which is obtained on running the solver using the default input file `exB.3dd`, thus confirming the accuracy of the written customization files. Upon postprocessing the results obtained for modal displacements and rotations are shown in Figures 3.5 and 3.6.

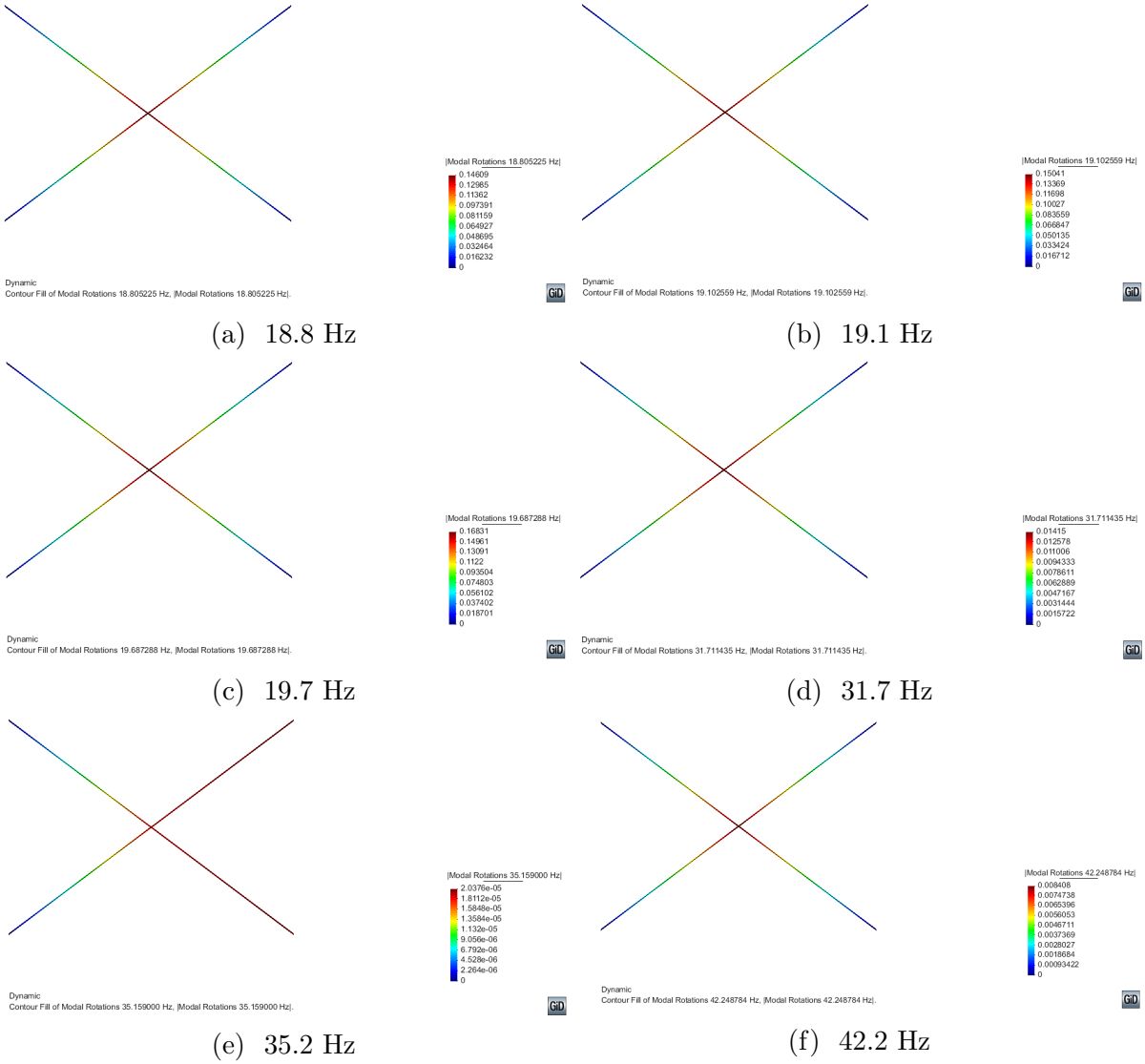


(a) 18.8 Hz (b) 19.1 Hz

(c) 19.7 Hz (d) 31.7 Hz

(e) 35.2 Hz (f) 42.2 Hz

Figure 3.5: Modal displacements for the first six modes



(a) 18.8 Hz (b) 19.1 Hz

(c) 19.7 Hz (d) 31.7 Hz

(e) 35.2 Hz (f) 42.2 Hz

Figure 3.6: Modal reactions for the first six modes

3.3 Triangular Tower frame

3.3.1 Geometry

This frame comprises of 15 nodes and 24 elements as shown in Figure 3.7

The nodal coordinates and geometry are tabulated in Tables 3.17 and 3.18. The area of cross section of every element is equal to 100 mm^2, shear areas in y and z directions are both 60 mm^2, torsional moment of inertia is 500 mm^4 while moments of inertia for bending in local y and z directions are both 1000 mm^4. Every element is made of material with Young's modulus and rigidity modulus being respectively equal to 9990 and 3800 Mpa. The roll of every element is equal to 0 while the density is equal to 2.4×10^{-10} T/mm^3.

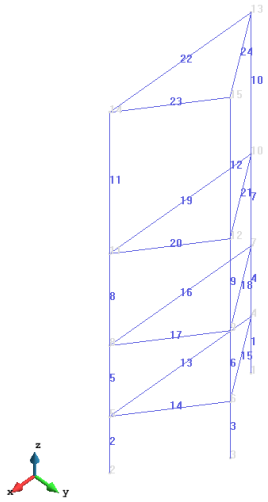


Figure 3.7: The geometry of the triangular tower frame

Table 3.17: Coordinated of nodes

Nodes	x (mm)	y (mm)	z (mm)
1	100	0	510
2	0	70	510
3	100	0	310
4	-100	0	510
5	0	70	310
6	-100	0	310
7	100	0	180
8	0	70	180
9	-100	0	180
10	100	0	80
11	0	70	80
12	-100	0	80
13	100	0	0
14	0	70	0
15	-100	0	0

Table 3.18: Connectivities

Ele	N1	N2	Ele	N1	N2
1	15	12	13	12	10
2	13	10	14	10	11
3	14	11	15	11	12
4	12	9	16	9	7
5	10	7	17	7	8
6	11	8	18	8	9
7	9	6	19	6	3
8	7	3	20	3	5
9	8	5	21	5	6
10	6	4	22	4	1
11	3	1	23	1	2
12	5	2	24	2	4

The truss is subjected to the following conditions.

3.3.2 Fixed Nodes

Table 3.19 gives the list of nodes and their displacements and rotations that are fixed. In the table columns x, y and z correspond to fixed displacements in the corresponding directions, while xx, yy and zz correspond to fixed rotations about the corresponding axes. A value of 1 corresponds to active fixed condition while 0 stands for free condition.

Table 3.19: Fixed nodes

Node	x	y	z	xx	yy	zz
13	1	1	1	1	1	1
14	1	1	1	1	1	1
15	1	1	1	1	1	1

3.3.3 Load Cases

There is one load case. In this case point load is applied on one node, uniform load is applied on twelve elements. These are tabulated in Tables 3.20 and 3.21

Table 3.20: Point loads

Node	Fx (N)	Fy (N)	Fz (N)	Mxx (N mm)	Myy (N mm)	Mzz (N mm)
2	0	200	0	0	0	0

Table 3.21: Uniform loads

Element	Ux (N/mm)	Uy (N/mm)	Uz (N/mm)
13	0	0	-2.361
14	0	0	-2.361
15	0	0	-2.361
16	0	0	-2.361
17	0	0	-2.361
18	0	0	-2.361
19	0	0	-2.361
20	0	0	-2.361
21	0	0	-2.361
22	0	0	-2.361
23	0	0	-2.361
24	0	0	-2.361

3.3.4 Results

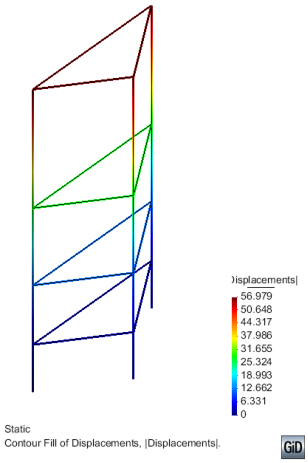
Upon calculating from within GiD using the *Calculate→Calculate* menu, the console output is as shown below.

```
FRAME3DD version: 20100105
Analysis of 2D and 3D structural frames with elastic and geometric stiffness.
http://frame3dd.sf.net
GPL Copyright (C) 1992-2010, Henri P. Gavin
This is free software with absolutely no warranty.
For details, see the GPL license file, LICENSE.txt
```

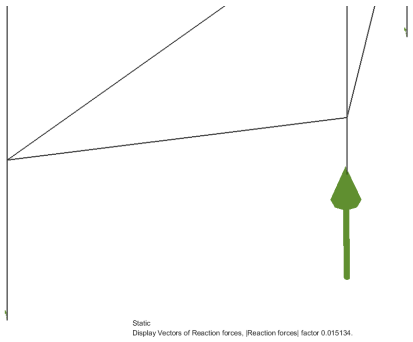
```
** Example B: a pyramid-shaped frame — static and dynamic analysis (N mm ton)
**
number of joints ..... nJ = 15 ... complete
number of joints with reactions ..... nR = 3 ... complete
number of frame elements..... nE = 24 ... complete
number of load cases ..... nL = 1
load case 1 of 1:
number of loaded joints ..... nF = 1
number of uniformly distributed loads ..... nU = 12
number of trapezoidally distributed loads ..... nW = 0
number of concentrated frame element point loads .. nP = 0
number of temperature changes ..... nT = 0
number of prescribed displacements ..... nD = 0
load data ... complete
number of dynamic modes ..... nM = 4
modal analysis method ..... 1 (Subspace-Jacobi)
number of joints with extra lumped inertia ..... nI = 0
number of frame elements with extra mass ..... nX = 0
structural mass ..... 7.9357e-005
total mass ..... 7.9357e-005
number of modes to be animated ..... nA = 4
mass data ... complete
Load Case 1 of 1 ...
Linear Elastic Analysis ... Mechanical Loads
Non-Linear Elastic Analysis ...
NR iteration 1 — RMS equilibrium precision: 2.88e-002
NR iteration 2 — RMS equilibrium precision: 2.41e-002
NR iteration 3 — RMS equilibrium precision: 2.64e-003
NR iteration 4 — RMS equilibrium precision: 4.50e-005
NR iteration 5 — RMS equilibrium precision: 3.34e-006
NR iteration 6 — RMS equilibrium precision: 8.57e-008
RMS relative equilibrium precision: 2.471e-004 ... good!
Modal Analysis ...
5 sub-space iterations, error: 1.0896e-007
mode: 1 DoF: 23 56.4351 Hz
mode: 2 DoF: 5 98.3373 Hz
mode: 3 DoF: 22 121.9289 Hz
mode: 4 DoF: 4 205.1359 Hz
mode: 5 DoF: 10 269.8723 Hz
mode: 6 DoF: 35 320.1958 Hz
mode: 7 DoF: 17 494.5193 Hz
mode: 8 DoF: 24 588.5382 Hz
There are 4 modes below 205.135860 Hz. All 4 modes were found.
```

The results are stored in the file `ProjectTriangularTower.out`. This file exactly resembles the file `exI.out`, which is obtained on running the solver using the default

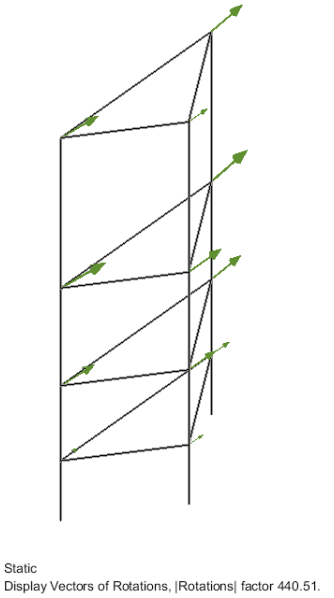
input file `exI.3dd`, thus confirming the accuracy of the written customization files. Upon postprocessing the results obtained for magnitude of displacement and reactions are shown in Figures 3.8a, 3.8b, 3.8c and 3.8d.



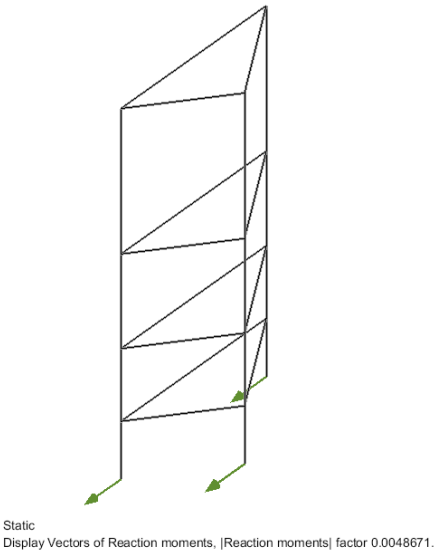
(a) Magnitude of displacement



(b) Reactions at fixed nodes



(c) Rotation at nodes



(d) Moments at nodes

Figure 3.8: Results

3.4 Conclusions

The results obtained for the three cases are in good agreement with the available results for these problems. This validates the correctness of the customization done in GiD for frame problems.

A frame3dd.bas

frame3dd.bas

```
Example B: a pyramid-shaped frame --- static and dynamic analysis N,mm,ton

*npoin
#.node  x       y       z       r
#        mm      mm      mm      mm

*set elemsall
*loop nodes
*NodesNum *NodesCoord1,real *NodesCoord2,real *NodesCoord3,real        0
*end nodes

*Set Cond Fixed-Nodes *nodes
*CondNumEntitiesint
#.n     x  y  z xx yy zz          1=fixed, 0=free

*loop nodes *OnlyInCond
 *NodesNum *CondX *CondY *CondZ *CondXX *CondYY *CondZZ
*end nodes

*nelem
#.e n1 n2 Ax    Asy     Asz     Jxx     Iyy     Izz     E       G   roll density
#   .  .  mm^2  mm^2    mm^2    mm^4    mm^4    mm^4    MPa     MPa  deg T/mm^3

*loop elems
*ElemsNum *ElemsConec *ElemsMatPropAx,real *ElemsMatPropAsy,real *ElemsMatPropAsz,real *ElemsMatPr
*end elems

*Set Cond Extra-Information *nodes
*loop nodes *OnlyInCond
*CondShear-deformation
*CondGeometric-stiffness
*CondStatic-mesh-deformations
*Condx-Axis-Increment-For-Internal-Forces                # x-axis increment for internal forces, n
                # if dx is -1 then internal force calculations are skipped.
*end nodes

*Nintervals
*loop intervals
# Begin Case
# gravitational acceleration for self-weight loading global
#.gX                gY              gZ
#.mm/s^2        mm/s^2                mm/s^2
*Set Cond Gravitational-Acceleration *nodes
*loop nodes *OnlyInCond
  *CondgX        *CondgY        *CondgZ
*end nodes
```

```
*Set Cond Loaded-Nodes *nodes
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e      Fx       Fy     Fz      Mxx     Myy     Mzz
#        N        N      N       N.mm    N.mm    N.mm
*loop nodes *OnlyInCond
 *NodesNum *CondFx *CondFy *CondFz *CondMxx *CondMyy *CondMzz
*end nodes
*endif
*Set Cond Uniform-Load *elems
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e    Ux   Uy   Uz
#     N/mm N/mm N/mm
*loop elems *OnlyInCond
 *ElemsNum *CondUx *CondUy *CondUz
*end elems
*endif
*Set Cond Trapezoidally-Distributed-Load *elems
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e     x1       x2        w1      w2
#       mm       mm       N/mm    N/mm
*loop elems *OnlyInCond
 *ElemsNum *Condx1 *Condx2 *CondXw1 *CondXw2
  *Condy1 *Condy2 *CondYw1 *CondYw2
  *Condz1 *Condz2 *CondZw1 *CondZw2
*end elems
*endif
*Set Cond Internal-Concentrated-Loads *elems
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e    Px   Py    Pz   x
#      N    N     N    mm
*loop elems *OnlyInCond
 *ElemsNum *CondPx *CondPy *CondPz *Condx
*end elems
*endif
*Set Cond Temperature-Loads *elems
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e  alpha   hy   hz   Ty+  Ty-  Tz+  Tz-
#    /degC   mm   mm   degC degC degC degC
*loop elems *OnlyInCond
*ElemsNum *Condalpha *Condhy *Condhz *CondTy+ *CondTy- *CondTz+ *CondTz-
*end elems
*endif
*Set Cond Prescribed-Displacement *nodes
*CondNumEntitiesint
*ifCondNumEntitiesint>0
*loop nodes *OnlyInCond
*NodesNum *CondDx *CondDy *CondDz *CondDxx *CondDyy *CondDzz
*end nodes
*endif
# End Case
```

```
*end intervals
*Set Cond Desired-Dynamic-Modes-Of-Vibration *nodes
*loop nodes *OnlyInCond
*CondNumber
*set var NUMBERint=OperationCondNumber,int
*end nodes
*ifNUMBER>0
*Set Cond Desired-Dynamic-Modes-Of-Vibration *nodes
*loop nodes *OnlyInCond
*Condsubspace
*CondType-Of-Mass-Matrix
*CondMode-Shape-Tolerance
*CondShift-Value
*CondExaggerate-Model-Mesh-Deformations
*end nodes
*endif

*ifNUMBER>0
# Concentrated mass and inertia
*Set Cond Extra-Inertia *nodes
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.n      Mass   Ixx      Iyy      Izz
#        ton    ton.mm^2 ton.mm^2 ton.mm^2
*loop nodes *OnlyInCond
*NodesNum *CondMass *CondIxx *CondIyy *CondIzz
*end nodes
*endif

*Set Cond Extra-Mass-Elements *elems
*CondNumEntitiesint
*ifCondNumEntitiesint>0
#.e      Mass
#        ton
*loop elems *OnlyInCond
*ElemsNum *CondMass
*end elems
*endif

*Set Cond Desired-Dynamic-Modes-Of-Vibration *nodes
*loop nodes *OnlyInCond
*CondNumber-Of-Modes-To-Animate
*set var ANIMATEint=OperationCondNumber-Of-Modes-To-Animate,int
*end nodes
*ifANIMATE==1
1
*elseifANIMATE==2
1 2
*elseifANIMATE==3
1 2 3
*elseifANIMATE==4
1 2 3 4
*elseifANIMATE==5
1 2 3 4 5
```

```
*elseifANIMATE==6
1 2 3 4 5 6                # list of modes to animate - omit if nA == 0
*elseifANIMATE==7
1 2 3 4 5 6 7
*elseifANIMATE==8
1 2 3 4 5 6 7 8
*elseifANIMATE==9
1 2 3 4 5 6 7 8 9
*elseifANIMATE==10
1 2 3 4 5 6 7 8 9 10
*elseifANIMATE==11
1 2 3 4 5 6 7 8 9 10 11
*elseifANIMATE==12
1 2 3 4 5 6 7 8 9 10 11 12
*endif
*loop nodes *OnlyInCond
*CondPan-During-Animation
*end nodes
*endif
```