

# FEM Package Handbook

Developed

By

SHAHROKH SHAHI

[www.sshahi.com](http://www.sshahi.com)

**FEM Package** Georgia Institute of Technology | 2018

Problem

Input File: \\prism.nas.gatech.edu\ssha [Browse] [View/Edit] [Reload]

Solver: General 3D [View/Edit]

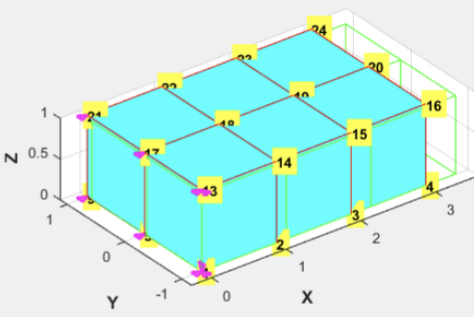
Options

- Save Model/Solution Structs
- Display Stiffness Matrices
- Save Results in Output File

**RUN ANALYSIS**

Graphical View

XY XZ YZ 3D



Graphic Options

- Node/Edge Color [Red]
- Node Labels [Yellow] Font Size = 10
- Boundary Conditions
- Element Color [Cyan]
- Element Labels [Magenta] Font Size = 10
- Applied Nodal Loads

Developed by Shahrokh Shahi | [shahi@gatech.edu](mailto:shahi@gatech.edu) [www.sshahi.com](http://www.sshahi.com)



Final Release of FEM Package version 1.0.2 by Shahrokh Shahi  
This package is a part of the FEM Package, a collection of finite element analysis tools developed by Shahrokh Shahi at Georgia Institute of Technology.

2018-2019

## 1. Overview

In general, there are two ways of using this FEM package which are presented in two different directories in the main folder:

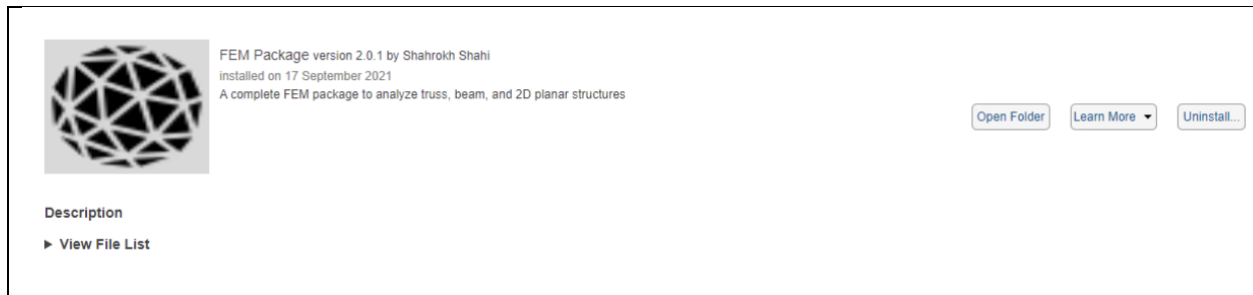
**Folder (1): Installation File**, which includes the main installation MATLAB Toolbox file (\*.mltbx).

**Folder(2): Source Code**, including all the codes developed for this package and can be used as a stand-alone package

In the following, the toolbox and its graphical interface will at first be introduced. Then, the format of input files and the most important variables will be discussed. Finally, the various parts of source code will be explained.

## 2. FEM MATLAB Toolbox

By clicking on the installation file in the first folder, you can “install” this package as a MATLAB toolbox on your personal computer. In order to install, after clicking on the file, simply press the “install” button to add the whole package to your MATLAB toolboxes.



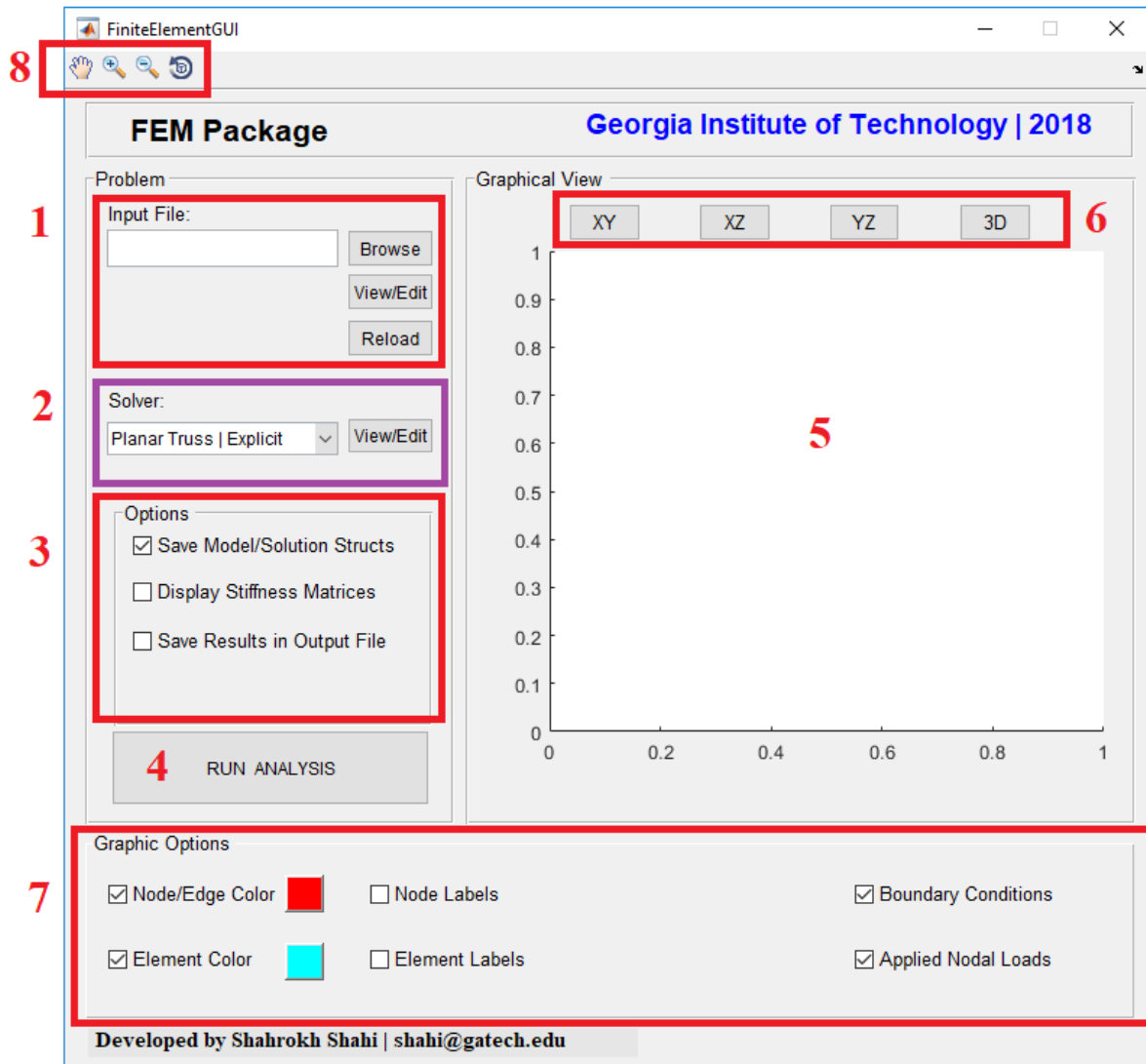
### Remarks:

- This package is cross-platform, so it can be used on any operating system (Windows, Linux, Mac OS), and it is compatible on MATLAB releases from R2014b to the latest.

Once the installation become complete, you can run the toolbox GUI by executing the following command (There is no need to change the “current directory”; it can be anything):

```
>> fem
```

Then, the graphical user interface will pop up:



According to the picture, the different sections of this interface are as follows:

- (1) By clicking on the Browse button, you can choose an INPUT file that contains the problem data. (In general to use this package, at first you need to create an input file in a specific format. This format will be introduced in the next section)

As soon as you pick a valid input file, a function called “inpFileReader.m” interprets the file and stores all the data in a MATLAB “struct” variable named “Model”.

Along with the interpreter and the computational features, a visualizer module has also been developed to help getting a better vision of the problem. The main visualizer function is “PlotMeshX.m” which gets a “Model” struct variable as an input and generates a featured graphical view of the problem. This figure will be displayed in the “Graphical View” section (specified with number (5)).

Therefore, as soon as the user picks a valid input file, the package interprets the data, stores the data in the Model struct, and generate a visualization in Graphical View section.

By clicking on the View/Edit button, you can see (and possibly edit) the picked input file in the MATLAB editor. Once you modify the input file, you can re-load it in the GUI, by pressing the Reload button. (The Graphical View will also be updated).

- (2) In this section, you can choose the Solver (computational function) to solve the problem. You can also see (and possibly edit) the chosen function by clicking on View/Edit button. Currently, six different solvers are defined in this package:

Solver	Function Name	Description
Planar Truss   Explicit	trussAnalysisExplicit.m	Planar Truss Analysis in which stiffness matrices are formed <b>explicitly</b>
Planar Truss   Numerical	trussAnalysisNumerical.m	Planar Truss Analysis in which stiffness and load matrices are formed using <b>Numerical Integration</b>
Planar Frame   Explicit	planeFrameExplicit.m	Planar Beam/Frame Analysis in which stiffness matrices are formed <b>explicitly</b>
Planar Frame   Numerical	planeFrameNumerical.m	Planar Beam/Frame Analysis in which stiffness and load matrices are formed using <b>Numerical Integration</b>
Plane Stress/Plane Strain	femGeneral.m	Plane stress and Plane strain Analysis (Calculation with <b>Numerical Integrator</b> )
General 3D	femGeneral.m	General 3D Analysis (Calculation with <b>Numerical Integrator</b> )

- (3) This section provides user with more analysis options to
- o Saving the Model and Solution structs in workspace (This can be useful for further investigation of the inputs and outputs. In general, all the variables will be vanished after executing an analysis. By using this item, we can still access the problem and its solution data even after closing the GUI)

By using this item, "Model" and "Solution" structs will be stored in "Model.mat" and "Solution.mat" data variables directly and can be fetched (anytime) by using the following command

```
>> load Model.mat
>> load Solution.mat
```

- Displaying Stiffness Matrices: which can be very useful for verification purposes
  - Saving Results in an Output File: By using this item you can specify the name of the output file to store all the inputs and outputs printed on the Command Window.
- (4) RUN ANALYSIS button: By pressing this button, the chosen solver (in section (2)) will solve the problem with the data stored in the Model struct from section (1), and will store the results of analysis in another struct variable named "Solution".

Moreover, a summary of output will be displayed on the Command Window.

(5) Graphical View

(6) View buttons: To change the view of visualization

(7) Graphic Options: provides user with some tools to tweak the visualization to improve the clarity

(8) Some handy tools to zoom in/out and rotate the visualization

### 3. Input File

In this package, problem data are defined in an input file, then it will be interpreted to a MATLAB struct variable, by means of **inpFileReader.m** function:



- General information about input files
  - In general, if a line starts with one of these three characters (**%**, **#**, **-**), it will be considered as a COMMENT line and the interpreter (inpFileReader.m) will ignore that.

This can be very useful to add more information in the input file for future using

- If a line starts with **"\*"**, it means it is a COMMAND line and it follows by a space and a command. A command is a pre-defined keyword which lets the interpreter know about the data comes immediately after this line.

#### IMPORTANT REMARKS:

- (1) Between the starting \* and the command MUST be AT LEAST one space
- (2) Between a command line and the data related to that command there must be NO empty line. In other words, the next line after a command line cannot be an empty line and it must include the data related to that command.
- (3) Commands are NOT case sensitive, so for instance, **"\* COORDINATES"**, **"\* coordinates"**, **"\* Coordinates"**, **"\*CoORdiNatEs"** are all equal.

The pre-defined command keywords are presented in the following table

	Command Keyword	Description
1	INFO	General information about the problem
2	ANALYSIS	Problem type
3	COORDINATES	Nodes coordinates
4	ELEMENTS	Elements connectivity
5	SECTIONS	Sections material properties
6	NODAL	Nodal loads information
7	TRACTION	Traction forces information

8	BODY	Body forces information
9	BOUNDARY	Boundary conditions information
10	HINGE	Specifying hinge nodes (only for beams and frames)

(1) \* INFO

The line(s) comes after this line provides information about the problem. This information will be stored in the Model struct for future usages.

**REMARK:**

The information comes after this line is not restricted to only one line and can be multiple lines (Use enter to continue the information in a new line)

Example:

```
* INFO
This is an example of employing numerical integration to calculate
the traction
forces on a plane stress/strain element edge.
[REF] Logan, D.L., 2011. A first course in the finite element
method. Cengage Learning. (Problem 10.14)
```

(2) \* ANALYSIS

The line comes after this line specifies the (problem) analysis type and can be one of the following keywords:

- Truss (for truss structures)
- Beam (for both beam and frame structures)
- Plane Stress (for plane stress problems)
- Plane Strain (for plane strain problems)
- General 3D (for three dimensional problems)

**REMARK:**

Again, these keywords can be used in both upper case and lower case (or their combinations) and using space(s) between two words does not make any difference, e.g. Plane Stress = planestress = planeStress

Example:

```
* ANALYSIS TYPE
Plane Stress
```

(3) \* COORDINATES

The lines come after this line include the nodal coordinates of the geometry, and it should be in the following format:

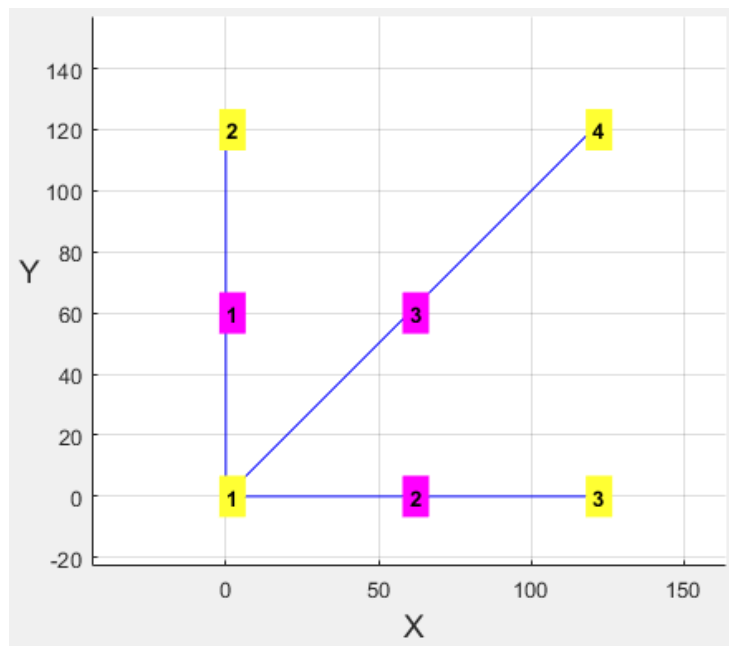
- For 2D geometries: Node ID  $X_i$   $Y_i$
- For 3D geometries: Node ID  $X_i$   $Y_i$   $Z_i$

**REMARK:**

The nodes can be inputted in any order. They will automatically ordered by their Node IDs.

Example:

For the following geometry,



The nodal coordinates should be inputted as follows:

```
* COORDINATES
1 0.0 0.0
2 0.0 120.0
3 120.0 0.0
4 120.0 120.0
```



(4) \* ELEMENTS

The lines come after this line include elements connectivity and also section data, and it should be in the following format:

Element ID    Section ID    Node ID 1    Node ID 2    ....

where the section ID specifies the section (and the material properties of the element)

Example:

For the same geometry the element data should be inputted as the following:

```
* ELEMENTS
1  1  1  2
2  1  1  3
3  2  1  4
```

In the above example element 1 and 2 have the same material properties (Section ID 1) and element 3 has different material properties (Section ID 2). These material properties will be defined under the SECTION command.

(5) \* SECTIONS

The line comes after this line include the material properties of each section and have the following format:

Section ID    Material Properties 1    Material Properties 2    ...

Depending on the problem analysis type (Truss/Beam/Plane/...), the number and definition of each material property can be different. The following table describes these properties for each case:

Analysis Type	Material Properties Order
Truss	Section ID, $E_i$ , $A_i$
Beam	Section ID, $E_i$ , $I_{zi}$ , $A_i$
Plane Stress	Section ID, $E_i$ , $\nu_i$ , $t_i$
Plane Strain	Section ID, $E_i$ , $\nu_i$ , $t_i$
General 3D	Section ID, $E_i$ , $\nu_i$

Example:

```
* SECTIONS
1  30e6  2
2  30e6  10
```

(6) `* NODAL` or `* NODAL FORCES`

The line(s) comes after this line include the nodal force(s) information

`Node ID` `Fxi` `Fyi` `[Mzi]`

Example:

```
* NODAL FORCES
1    0 -1e4
4    1e3 2e3
```

(7) `* TRACTION` or `* TRACTION FORCES`

The line(s) comes after this line describes the traction force(s) applied on the elements edge(s) (in 2D cases) or face(s) (in 3D cases).

Moreover, this command can be used to define distributed loads in beam and frame analysis.

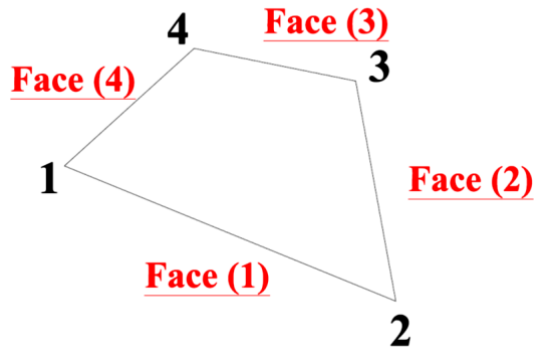
In general, the input should have the following format:

`Element ID` `Face No.` `Txi` `Tyi`

where "Face No." specifies the face (or the edge, in case of 2D elements) to apply Tx<sub>i</sub> and Ty<sub>i</sub>. The convention for the face nodes numbering is presented in the following table.

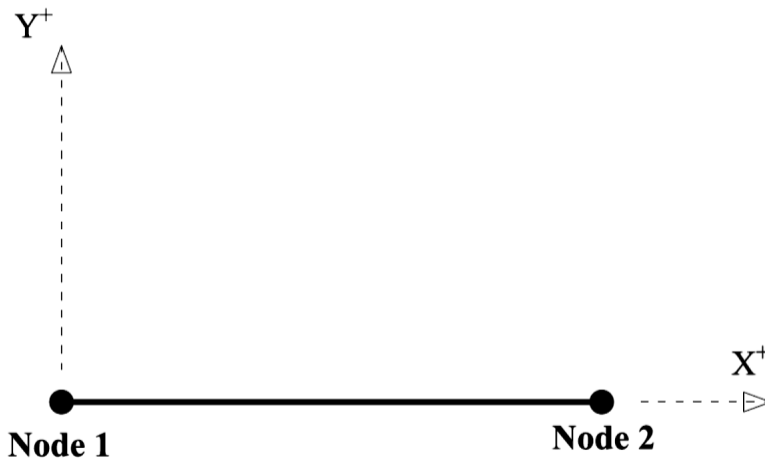
Element	Dim.	No of Nodes per Each Element	Face (Edge) Node Index
Bar	2D	2	Face 1 = [1, 2]
Triangle	2D	3	Face 1 = [1, 2] Face 2 = [2, 3] Face 3 = [3, 1]
Q4	2D	4	Face 1 = [1, 2] Face 2 = [2, 3] Face 3 = [3, 4] Face 4 = [4, 1]
Wedge	3D	4	Face 1 = [1, 2, 3] Face 2 = [1, 4, 2] Face 3 = [2, 4, 3] Face 4 = [3, 4, 1]
Brick	3D	8	Face 1 = [1, 2, 3, 4] Face 2 = [5, 8, 7, 6] Face 3 = [1, 5, 6, 2] Face 4 = [2, 3, 7, 6] Face 5 = [3, 7, 8, 4] Face 6 = [4, 8, 5, 1]

For instance, for a Q4 element:



**REMARKS:**

- Forces are defined in element local coordinates
- On the edge of bar elements (Truss, Beam/Frame) the positive direction of x is from Node 1 to Node 2, and the positive direction of y axis is upward.



- For the bar elements, face number is always 1.
- $T_{x_i}$  and  $T_{y_i}$  can be defined as a function of x (in element local coordinate). This function can be a combination of MATLAB internal functions or it can be the name of a user-defined function which is defined in a MATLAB m-file.

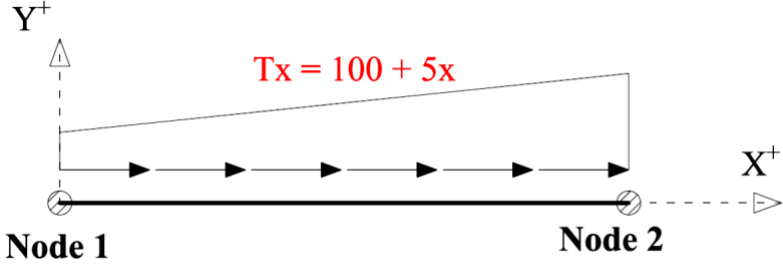
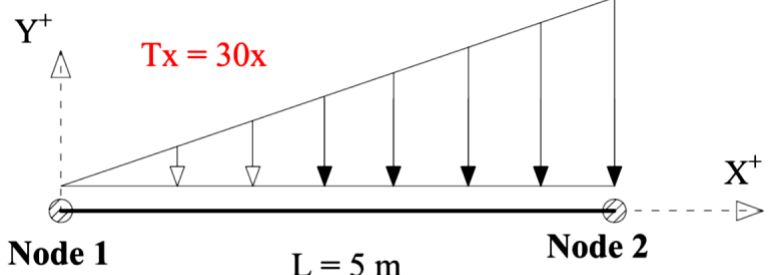
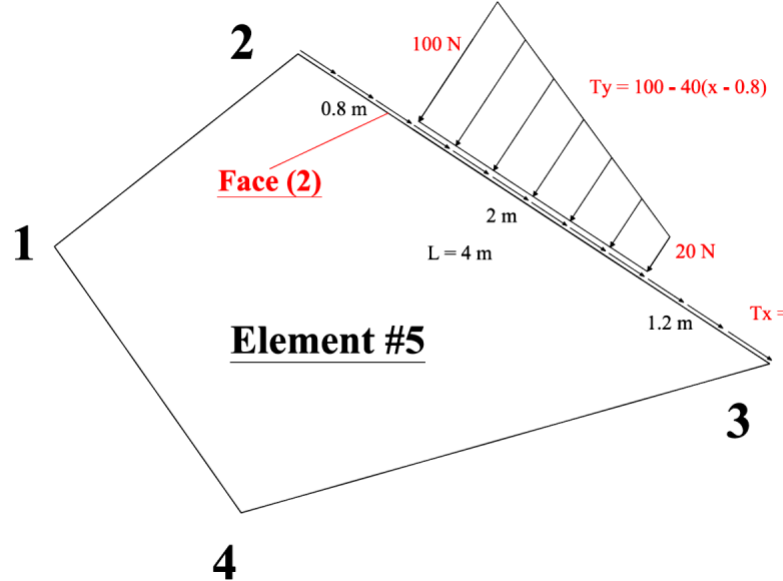
IMPORTANT NOTE: Space is not allowed in the definition of a function in the input file. Using spaces will break the interpreter. For instance, the following line is a valid traction definition:

```
3 1 3*x+sin(x) 0
```

, but the following is invalid:

```
3 1 3 * x + sin(x) 0
```

Example: The following examples clarify the traction definition.

Problem	Traction Definition Line in Input File
 <p style="text-align: center;"><b>Element #4</b></p>	<pre>* TRACTION FORCES ... 4 1 100+5*x 0 ... ...</pre>
 <p style="text-align: center;"><b>Element #2</b></p>	<pre>* TRACTION FORCES ... 2 1 0 -30*x ... ...</pre>
 <p style="text-align: center;"><b>Element #5</b></p>	<pre>* TRACTION FORCES ... 5 2 10 MyLoad(x) ... ...</pre> <p><b>MyLoad.m :</b></p> <pre>function f = MyLoad(x) if (x &lt; 0.8) f = 0; elseif (x &gt;= 0.8) &amp;&amp; (x &lt;= 4.8) f = -(100 - 40 * (x - 0.8)); else f = 0; end end</pre>

(8) 

* BODY	or	* BODY FORCES
--------	----	---------------

The line(s) comes after this line include the nodal force(s) information

Element ID     $F_{x_i}$      $F_{y_i}$

where  $F_{x_i}$  and  $F_{y_i}$  are the density of the element .

REMARKS:

For the forces in y direction, the upward direction is considered positive. Therefore, the force caused by an element force will essentially be inputted with negative sign.

**Example:**

```
* BODY FORCES
1    0  -7800
```

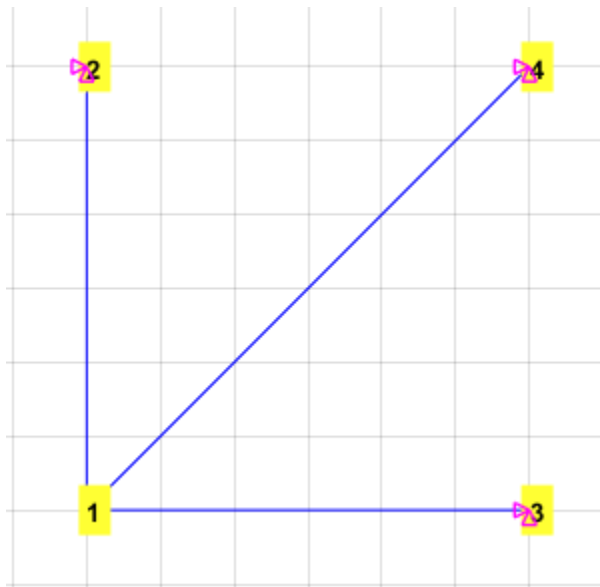
(9) 

* BOUNDARY	or	* BOUNDARY CONDITIONS
------------	----	-----------------------

The boundary condition information will be defined in the lines after BOUNDARY command in the following format:

Node ID, DOF number, Prescribed Value

For instance, for the following structure



the boundary condition data is the following:

```
* BOUNDARY
2 1 0.0
2 2 0.0
3 1 0.0
3 2 0.0
4 1 0.0
4 2 0.0
```

(10) \* HINGE

The line comes after this line include the hinges node IDs. This item is only applicable for beam/plane frame analysis and it will be used only if there is at least one hinge in the structure. For instance, if nodes 3 and 7 are hinges in a plane frame structure, in the input file:

```
* HINGE
3 7
```

## GENERAL REMARKS and DIAGNOSTICS:

Due to the robustness of the interpreter, the input file definition is very flexible:

- The order of the sections is not restricted and they can be defined in any order. For instance, you can start the input file definition with boundary condition definition (Of course, it is always better to keep the rational order of defining sections)
- In each section (e.g. COORDINATES, ELEMENTS, SECTIONS, etc) the data line can be in any order (the data will automatically be ordered by the interpreter)
- Commands can be in both lower and upper case or a combination of them. This is also true for the Analysis Type definition
- You can add as many as comments or separator lines or even empty lines that you need to improve the clarity and readability of an input file. (Note: do NOT forget to use #, %, or – at the beginning of each comment line)

Limitations (Diagnostics):

There are some limitations in the input file definition that the interpreter may fail in successfully reading of the input file:

- In a command line, there MUST be a space between the starting \* and the command.
- Load equation can only be defined for TRACTION FORCES.
- NO SPACE is allowed in load equation definition
- NO SPACE at the end of each data line is allowed. For instance, the following input file is invalid because at the third line of boundary condition definition, there is an additional space at the end of line:

```
33 8 1 9 10 15 14
34
35 * SECTIONS
36 1 26.0 0.3 1.0
37
38
39 * TRACTION FORCES
40 4 2 1.0 0.0
41 8 2 1.0 0.0
42
43 * BOUNDARY
44 1 . . 1 . 0.0
45 1 . . 2 . 0.0
46 6 . . 1 . 0.0
47 11 . 1 . 0.0
48
49
```

## A Brief Review on Variable Names and the “Model” Struct Fields

Once the function `inpFilereader.m` interprets an input file, the content of the input file will be stored in a MATLAB struct variable named “Model”. The following is a hierarchical view of the most important fields in a Model struct:

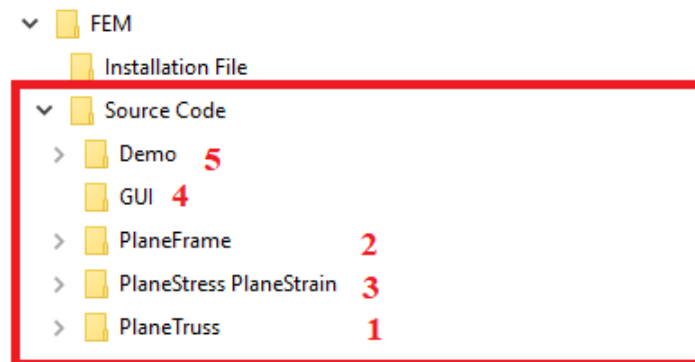
### Model:

```
|
+--- info: information about the model
+--- analysisType: TRUSS/BEAM/PLANE STRESS/PLANE STRAIN/GENERAL 3D
+--- nDim: 1=1D / 2=2D / 3=3D
+--- nNode: total number of nodes
+--- nElem: total number of elements
+--- nElemNode: number of nodes per each element
+--- nDof: number of degrees-of-freedom per each node
+--- nSection: number of section
+--- nMaterialProp: number of material properties to define
|                       section(s)
|
+--- geometry: an inner structure including geometry data
| |
| +--- coordinates: node coordinates matrix (nNode x nDim)
| +--- elements: elements connectivity (nElem x nElemNode)
|
+--- elemSectId: the section id for each element (nElem x 1)
+--- sections: section data/material (nSect x nMaterialProp)
|
+--- nNodalForce: number of nodal forces
+--- nBodyForce: number of element body forces
+--- nTractionForce: number of element traction forces
|
+--- loading: an inner structure including loading data
| |
| +--- nodalForces: nodal forces data (nNodalForce x nDof)
| +--- bodyForces: body forces data (nBodyForce x nDim)
| +--- tractionForces: traction forces data (nTra. x nDim)
|
+--- nBoundary: number of boundary condition entries
+--- boundary: boundary condition data (nBoundary x 3)
|                       3: nodeID, dof, value
```



## 4. Source Codes

The second folder in the main directory is **Source Code**. This folder includes all the codes developed for this package. You can run each solver separately to test their accuracy and performance in detail. Moreover, there are a few snippets (demos) to demonstrate some particular calculations e.g. Numerical Integration, Distributed Traction Forces, etc. In the following all the available folders and functions is briefly explained:

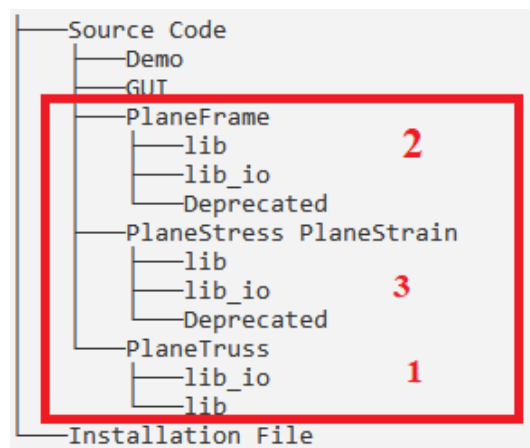


According to the above figure:

- 1- PlaneTruss: All the functions required for analysis of planar trusses
- 2- PlanarFrame: All the functions required for analysis of planar beams and frames
- 3- PlaneStress PlaneStrain: All the functions required for plane stress/strain analysis
- 4- GUI: Including all the files related to the graphical user interface
- 5- Demo: This folder is including a few specific demonstrations for educational purposes

Amongst the above-mentioned folders, the first three folders form the computational core of the package. In other words, the contents of these folders are used by the GUI to conduct the analysis.

Moreover, each folder can be used as a stand-alone program. For this purpose, each folder has the following structure:



In each of these folders, there are two LIBRARY folder:

**lib\_io**: including the functions that handle the **pre- and post-processing** e.g. reading (processing) the input file, displaying the outputs on the command window, or storing the results in an output file

**lib**: this folder contains all the **computational functions** e.g. the main solvers (analyzer functions), shape functions and their derivatives, numerical integration, etc.

Moreover, there is a **main.m** file in each of these folders which provides a step-by-step manual for analyzing a given problem.

You can also find a few example **input** file in each folder. The purpose and the source of each problem is briefly described in each input file under the **\* INFO** command.