Table of Contents

2. Problem solving with cane Multiphysics

cane Multiphysics is a framework for numerical simulations in engineering. It's input is made externally, using GiD pre-processing software [1] to describe the parameters of the problem, whereas the output files of cane can be postpocessed in Paraview [2], an open source post-processing software. cane is capable of performing many different analyses that need to be performed in order to accurately model physical phenomena that occur in engineering problems. Such analyses are:

- Computational Fluid Dynamics analyses (CFD)
- Computational Fluid-Structure Interaction analyses (FSI)
- Contact Mechanics analyses
- Thermal Conduction analyses
- Plate in membrane action analyses (Plane Stress, Plane Strain)
- Isogeometric analyses (IGA)

This chapter is devoted to giving a brief introduction on how to use cane to solve Solid and Fluid Mechanics problems both separately and when interacting with each other as a coupled system. The problems solved in this chapter are carefully selected from bibliography in order for the reader to be able to compare results between cane and the results from the solution of the problem in the selected references.

2.1. Steady state plane stress problem

In this problem, a uniformly distributed load is subjected to a thin plate structure as shown in the figure below. The plate is discretized using two linear triangular elements for illustration purposes. More elements must be used in order to obtain reliable results. This problem can be found in [3].

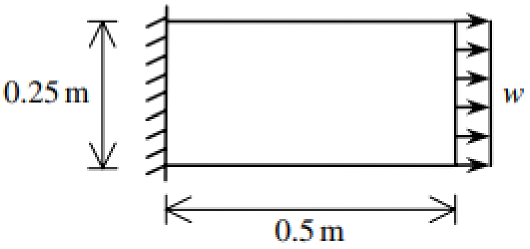


Figure 1

The abovementioned problem will be solved using the following parameters:

- Modulus of Elasticity: E = 210 GPa
- Poisson's ratio: v = 0.3
- Plate thickness: t = 0.25m
- Distributed load: w = 3000 kN/m2

Since the thickness is relatively small compared to the other dimensions of the plate, we can assume plane stress state for the static analysis.

2.1.1 Preprocessing with GID

Firstly, the user must specify the MATLAB GiD problem type. Select Data -> Problem Type -> matlab.

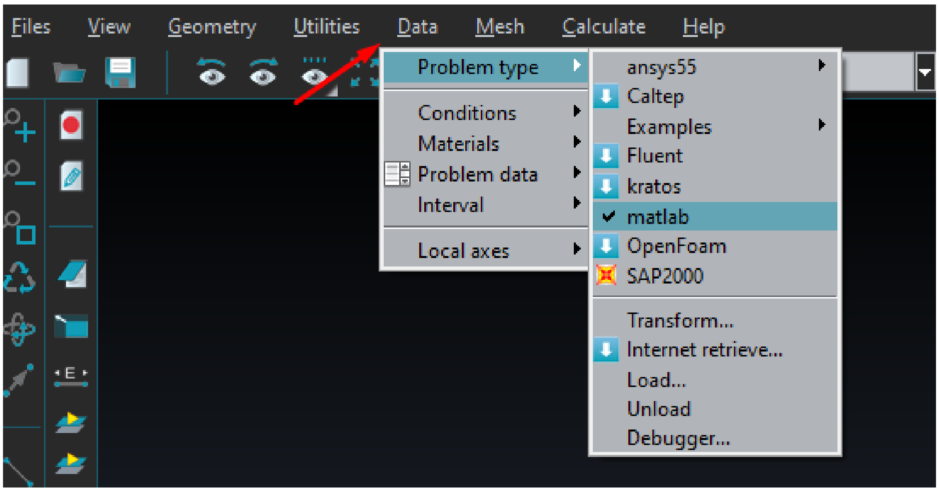
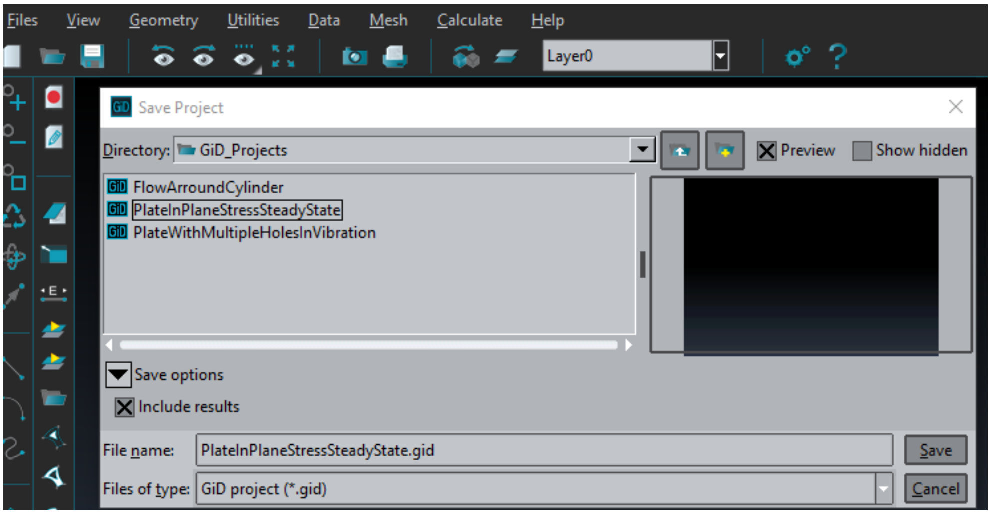


Figure 2

At this point the user should save the project with a name of his choice. Select Files->Save and type the file name.



Directory: GiD_Projects

FlowArroundCylinder
PlateInPlaneStressSteadyState
PlateWithMultipleHolesInVibration

☒ Preview □ Show hidden

Save options

☒ Include results

File name: PlateInPlaneStressSteadyState.gid

Files of type: GiD project (*.gid)

Save

Cancel

Figure 3

2.1.1.1. Geometry Setup

To create a geometry, select create object → rectangle. The rectangle can be drawn by clicking on the drawing plane or by specifying the coordinates of its corner edges (Fig. 1). The coordinates must be typed in the format x y z. They must contain white space between each coordinate, whereas omitting coordinates are assumed by default to be zero. Enter the first point (0, 0) in the command line and confirm with esc. Now enter the second point (0.5, 0.25).

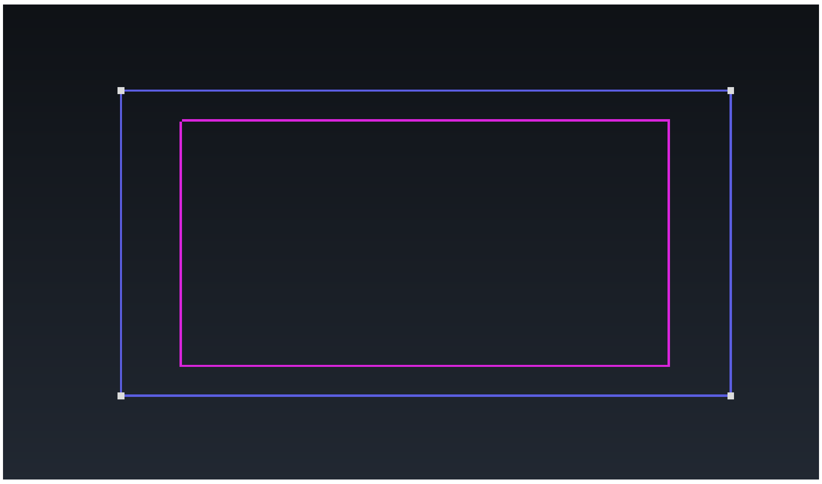
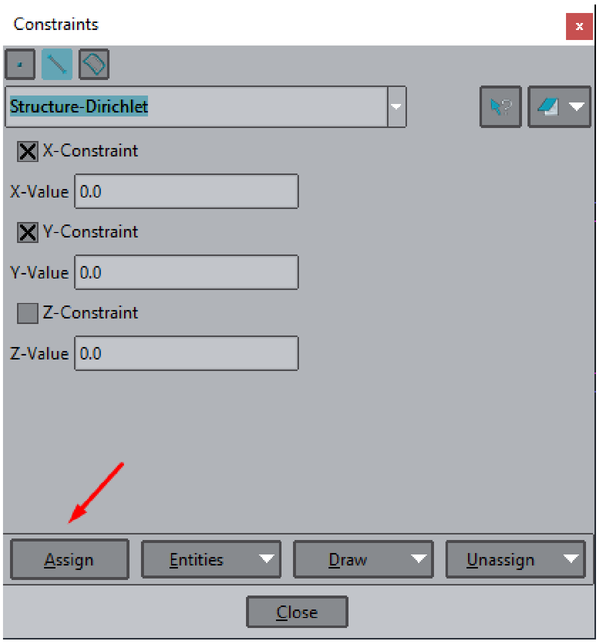


Figure 4

2.1.1.2 Boundary Conditions

To specify the Dirichlet boundary conditions select Data → Conditions → Constraints. Select lines (line icon) as selection type and select Structure-Dirchlet.


Constraints

Structure-Dirichlet

☒ X-Constraint

X-Value 0.0

☒ Y-Constraint

Y-Value 0.0

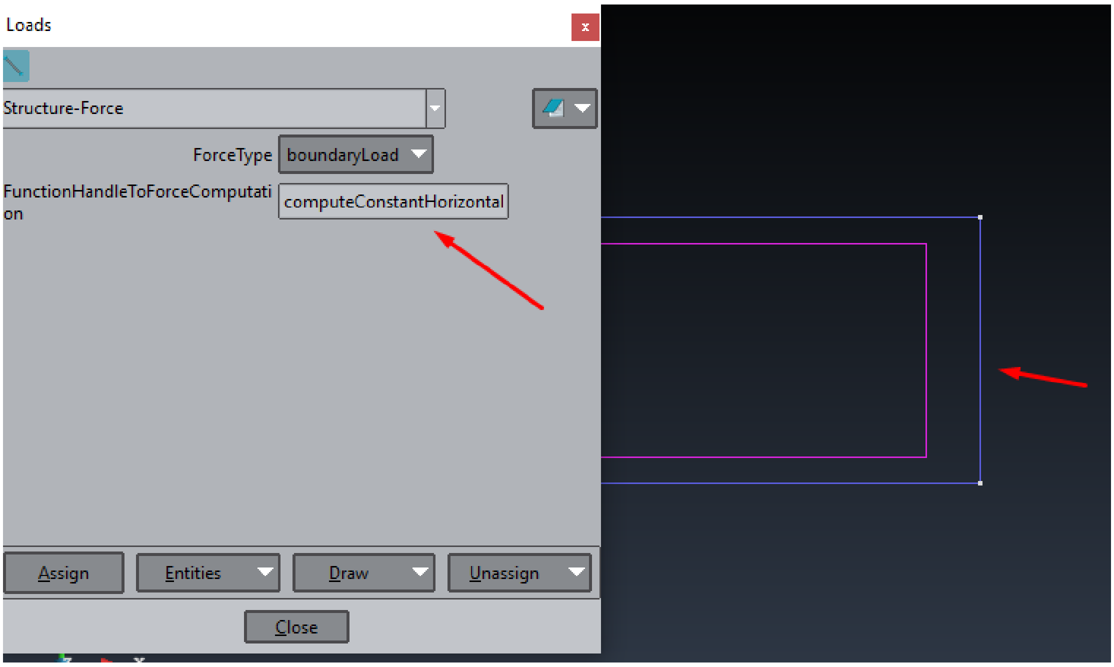
☐ Z-Constraint

Z-Value 0.0

Figure 5

Then click on the Assign button and select the appropriate side to apply the boundary condition. Here, both translations on x and y are so we choose the value 0.

To assign the constant horizontal load select Data -> Conditions -> Loads


Loads

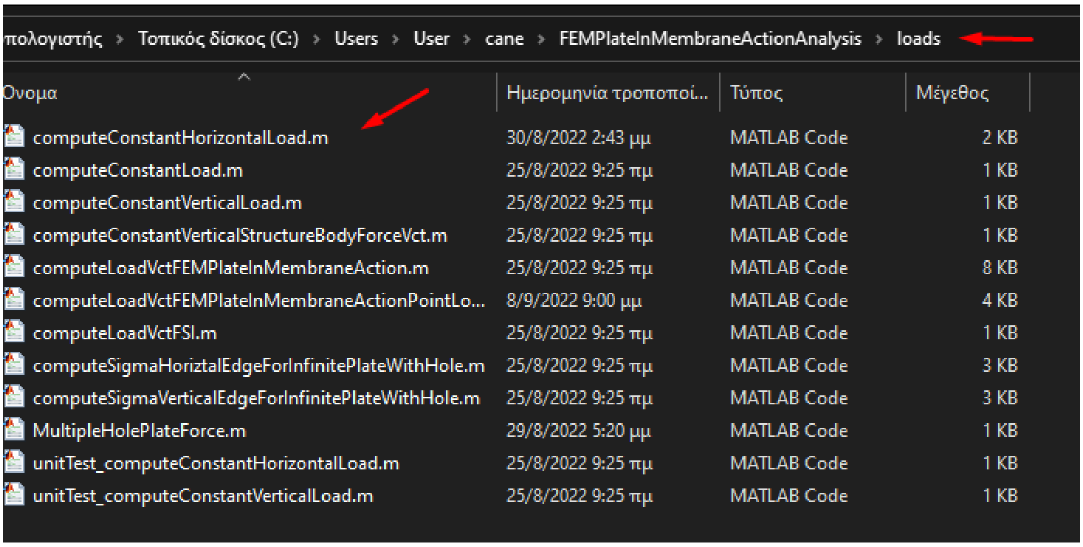
Structure-Force

ForceType boundaryLoad

FunctionHandleToForceComputation computeConstantHorizontal

Figure 6

In this case, we use the function ‘computeConstantHorizontalLoad’ which is located in the loads directory of cane repository:

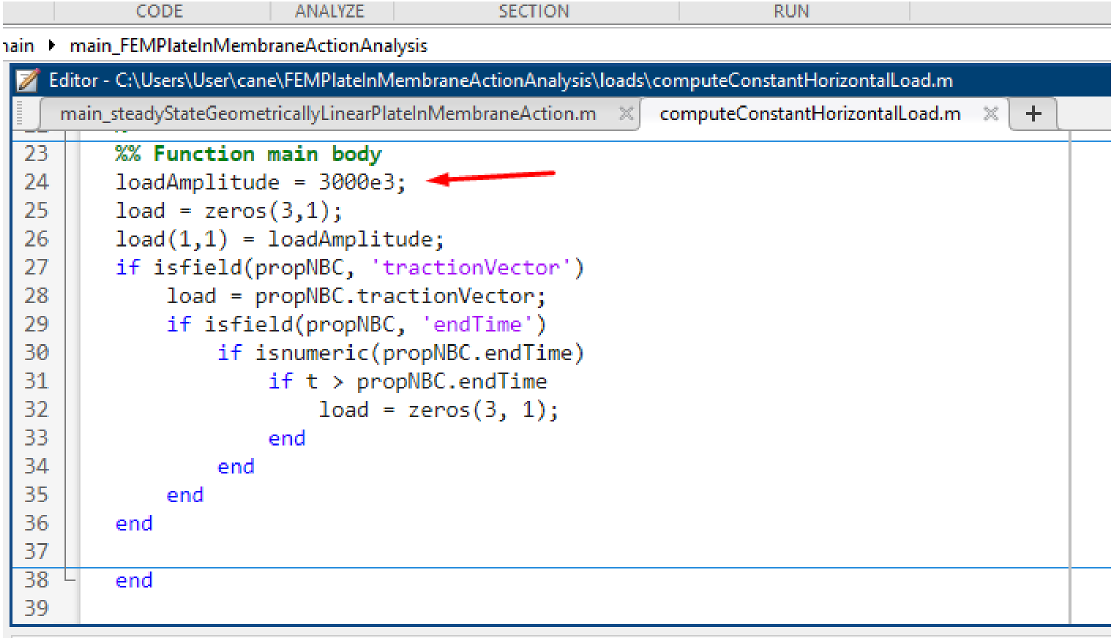


πολογιστής › Τοπικός δίσκος (C:) › Users › User › cane › FEMPlateInMembraneActionAnalysis › loads

Όνομα	Ημερομηνία τροποποί...	Τύπος	Μέγεθος
computeConstantHorizontalLoad.m	30/8/2022 2:43 μμ	MATLAB Code	2 KB
computeConstantLoad.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB
computeConstantVerticalLoad.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB
computeConstantVerticalStructureBodyForceVct.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB
computeLoadVctFEMPlateInMembraneAction.m	25/8/2022 9:25 πμ	MATLAB Code	8 KB
computeLoadVctFEMPlateInMembraneActionPointLo...	8/9/2022 9:00 μμ	MATLAB Code	4 KB
computeLoadVctFSI.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB
computeSigmaHoriztalEdgeForInfinitePlateWithHole.m	25/8/2022 9:25 πμ	MATLAB Code	3 KB
computeSigmaVerticalEdgeForInfinitePlateWithHole.m	25/8/2022 9:25 πμ	MATLAB Code	3 KB
MultipleHolePlateForce.m	29/8/2022 5:20 μμ	MATLAB Code	1 KB
unitTest_computeConstantHorizontalLoad.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB
unitTest_computeConstantVerticalLoad.m	25/8/2022 9:25 πμ	MATLAB Code	1 KB

Figure 7

Inside the script, we have to set the value for the amplitude of the load (3000kN/m2):


CODE
ANALYZE
SECTION
RUN
ıain ▸ main_FEMPlateInMembraneActionAnalysis
Editor - C:\Users\User\cane\FEMPlateInMembraneActionAnalysis\loads\computeConstantHorizontalLoad.m
main_steadyStateGeometricallyLinearPlateInMembraneAction.m
computeConstantHorizontalLoad.m
+

```
%% Function main body
loadAmplitude = 3000e3;
load = zeros(3,1);
load(1,1) = loadAmplitude;
if isfield(propNBC, 'tractionVector')
    load = propNBC.tractionVector;
    if isfield(propNBC, 'endTime')
        if isnumeric(propNBC.endTime)
            if t > propNBC.endTime
                load = zeros(3, 1);
            end
        end
    end
end

end
```

Figure 8

2.1.1.3. Definition of the computational domain

The computational mesh needs then to be assigned to a domain, so that the nodes and the elements for the chosen domain are written out to the desirable input file. Select Data → Conditions → Domains, choose Structure-Nodes from the drop-down menu and select the whole surface. Confirm with esc. Then, select Structure-Elements and repeat the previous step to assign the elements to the computational domain.

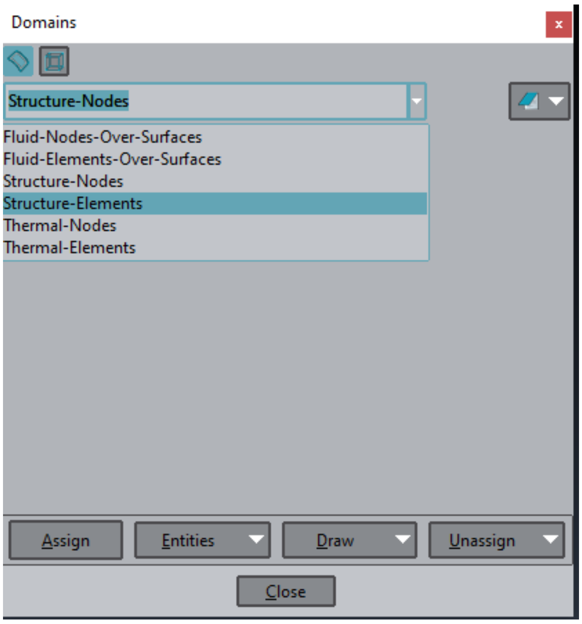
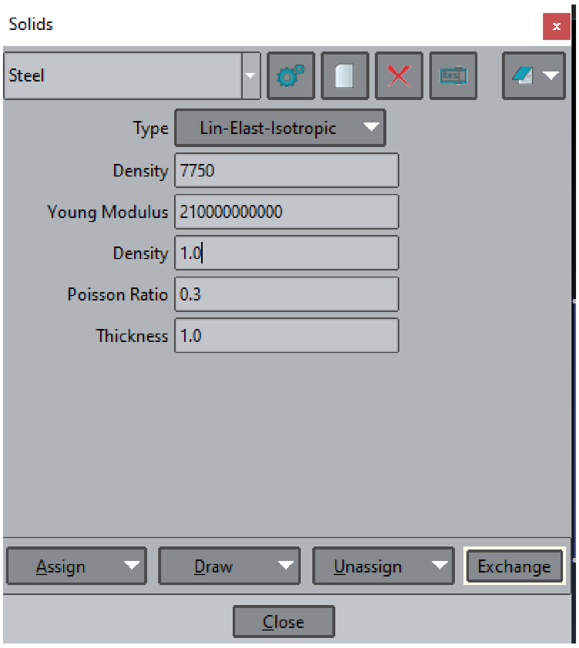


Figure 9

2.1.1.4 Selection of material properties

In order to select the material properties, select Data → Materials → Solids. There one can select the default material Steel from the drop-down menu or just change the given parameters to adjust material properties. Apply the material to the geometry of the problem by selecting Assign -> Surfaces and choose the surface of defining the problem's domain. Save the changes if asked so. The user may also expand the materials selection by editing the corresponding files under the folder matlab.gid.



Solids

Steel

Type: Lin-Elast-Isotropic

Density: 7750

Young Modulus: 210000000000

Density: 1.0

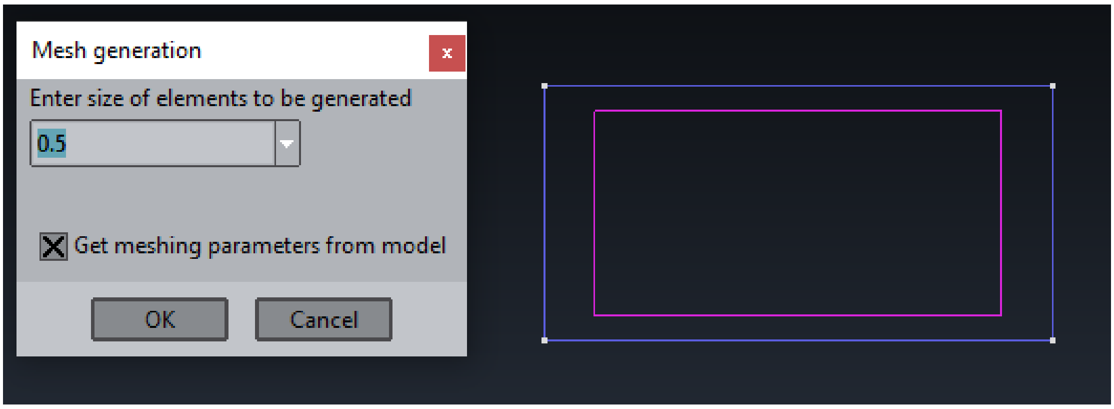
Poisson Ratio: 0.3

Thickness: 1.0

Figure 10

2.1.1.5 Generation of the computational mesh

Lastly, the finite element mesh needs to be generated. The simplest way is to go to Mesh → Generate Mesh and specify the element size.



Mesh generation

Enter size of elements to be generated: 0.5

☒ Get meshing parameters from model

Figure 11

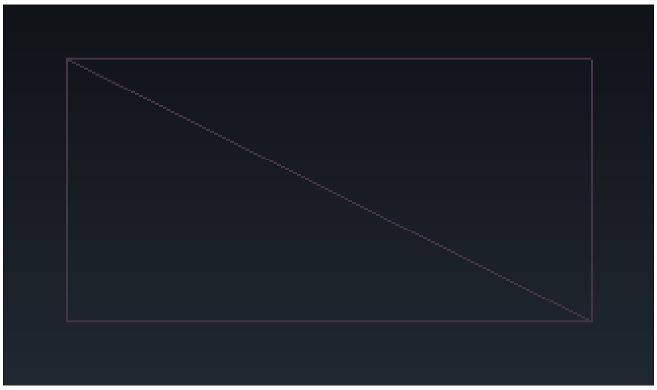
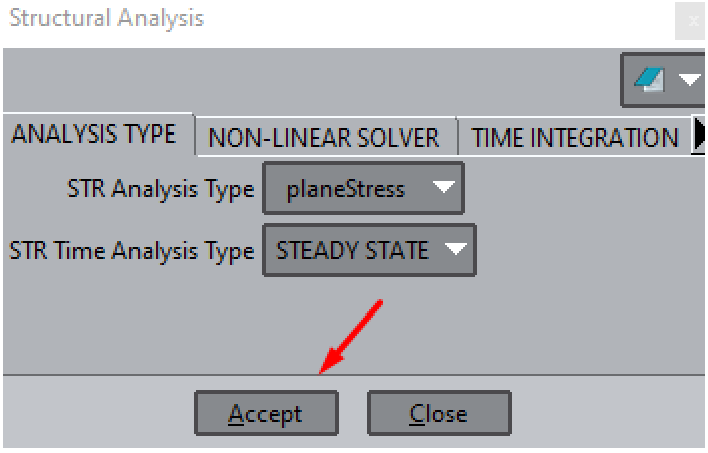


Figure 12

2.1.1.6 Selection of the analysis type and solver setup

To select the analysis type and setup the solver, select Data → Problem Data → Structural Analysis. In this window the user can select plane stress or plain strain analysis type and choose between a steady state or a transient analysis. Many more settings are possible, specifically on the time integration schemes and Gauss integration, but they are not needed for this simple case in which the default options are sufficients.



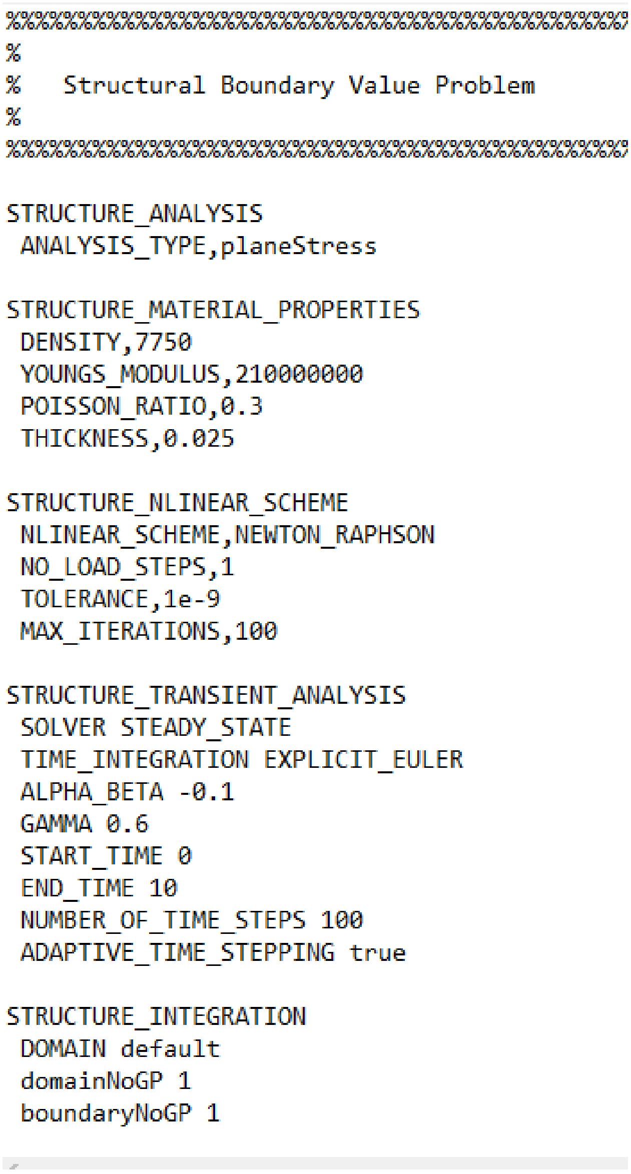
STR Analysis Type: planeStress

STR Time Analysis Type: STEADY STATE

Figure 13

2.1.1.7 Generation of input file for analysis in MATLAB

After the setup is complete, select Calculation → Calculation (F5) or just select F5 to write out the input file which will be later on parsed within cane MATLAB framework. This file has the same name as our project whereas its extension is .dat. The user can also open it with any text editor to check or adjust the data. This file needs then to be placed under folder ./cane/inputGiD/FEMPlateInMembraneActionAnalysis and then a new caseName with the same name needs to be defined in the MATLAB main driver script.

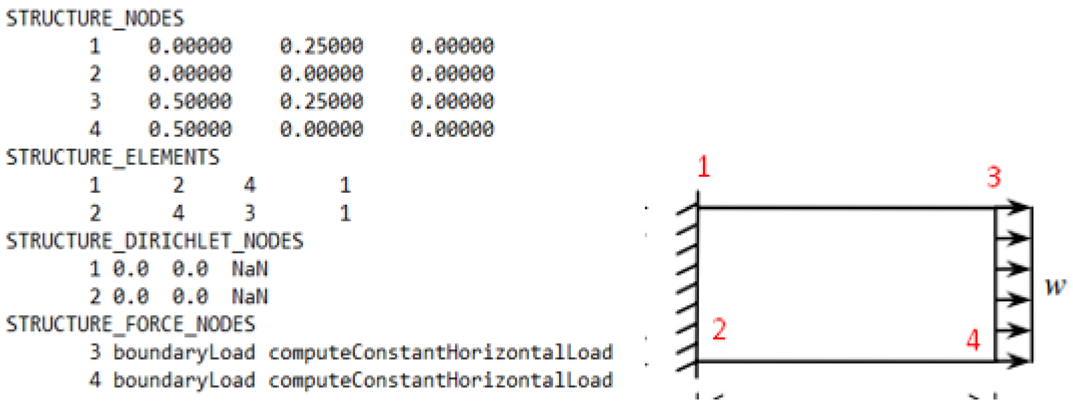


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Structural Boundary Value Problem
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

STRUCTURE_ANALYSIS
 ANALYSIS_TYPE,planeStress

STRUCTURE_MATERIAL_PROPERTIES
 DENSITY,7750
 YOUNGS_MODULUS,210000000
 POISSON_RATIO,0.3
 THICKNESS,0.025

STRUCTURE_NLINEAR_SCHEME
 NLINEAR_SCHEME,NEWTON_RAPHSON
 NO_LOAD_STEPS,1
 TOLERANCE,1e-9
 MAX_ITERATIONS,100

STRUCTURE_TRANSIENT_ANALYSIS
 SOLVER STEADY_STATE
 TIME_INTEGRATION EXPLICIT_EULER
 ALPHA_BETA -0.1
 GAMMA 0.6
 START_TIME 0
 END_TIME 10
 NUMBER_OF_TIME_STEPS 100
 ADAPTIVE_TIME_STEPPING true

STRUCTURE_INTEGRATION
 DOMAIN default
 domainNoGP 1
 boundaryNoGP 1
```

Figure 14

2.1.1.8 Enumeration of nodes and degrees of freedom

The generated data file contains the structure nodes and elements enumerated. The enumeration starts from the top and ends in the bottom of each column of nodes. An illustrative example is defined below:



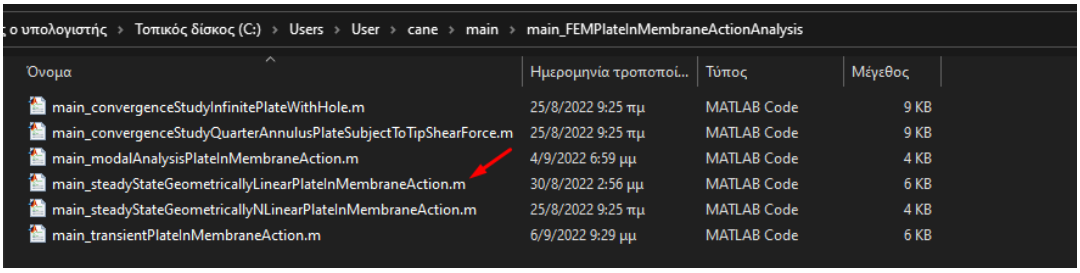
```
STRUCTURE_NODES
        1    0.00000    0.25000    0.00000
        2    0.00000    0.00000    0.00000
        3    0.50000    0.25000    0.00000
        4    0.50000    0.00000    0.00000
STRUCTURE_ELEMENTS
        1      2     4       1
        2      4     3       1
STRUCTURE_DIRICHLET_NODES
        1 0.0   0.0   NaN
        2 0.0   0.0   NaN
STRUCTURE_FORCE_NODES
        3 boundaryLoad computeConstantHorizontalLoad
        4 boundaryLoad computeConstantHorizontalLoad
```

Figure 15

2.1.2. Solving with cane Multiphysics

Now, the geometry of the problem is properly defined, along with the boundary conditions and the mesh. The next step is to open the appropriate matlab script from the cane repository and specify how to handle our case problem.

From the cane main repository select main->mainFEM_PlateInMembraneActionAnalysis-> main_steadyStateGeometricallyLinearPlateInMembraneAction.m. This is the matlab script which will parse the data file created from GiD.

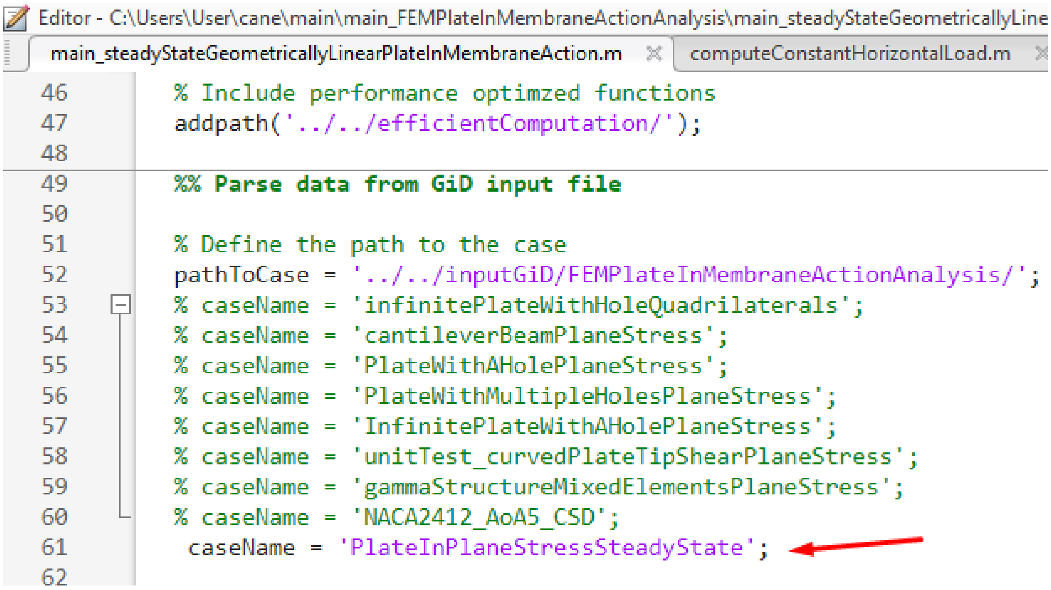


: ο υπολογιστής › Τοπικός δίσκος (C:) › Users › User › cane › main › main_FEMPlateInMembraneActionAnalysis

Όνομα	Ημερομηνία τροποποί...	Τύπος	Μέγεθος
main_convergenceStudyInfinitePlateWithHole.m	25/8/2022 9:25 πμ	MATLAB Code	9 KB
main_convergenceStudyQuarterAnnulusPlateSubjectToTipShearForce.m	25/8/2022 9:25 πμ	MATLAB Code	9 KB
main_modalAnalysisPlateInMembraneAction.m	4/9/2022 6:59 μμ	MATLAB Code	4 KB
main_steadyStateGeometricallyLinearPlateInMembraneAction.m	30/8/2022 2:56 μμ	MATLAB Code	6 KB
main_steadyStateGeometricallyNLinearPlateInMembraneAction.m	25/8/2022 9:25 πμ	MATLAB Code	4 KB
main_transientPlateInMembraneAction.m	6/9/2022 9:29 μμ	MATLAB Code	6 KB

Figure 16

The first thing we need to do when we open the script is to define a path for matlab in order to find the generated file from GiD. We create a new case with the name of the generated file, in our case ‘PlateInPlaneStressSteadyState’.


Editor - C:\Users\User\cane\main\main_FEMPlateInMembraneActionAnalysis\main_steadyStateGeometricallyLine
main_steadyStateGeometricallyLinearPlateInMembraneAction.m
computeConstantHorizontalLoad.m

```
% Include performance optimzed functions
addpath('../../efficientComputation/');

%% Parse data from GiD input file

% Define the path to the case
pathToCase = '../../inputGiD/FEMPlateInMembraneActionAnalysis/';
% caseName = 'infinitePlateWithHoleQuadrilaterals';
% caseName = 'cantileverBeamPlaneStress';
% caseName = 'PlateWithAHolePlaneStress';
% caseName = 'PlateWithMultipleHolesPlaneStress';
% caseName = 'InfinitePlateWithAHolePlaneStress';
% caseName = 'unitTest_curvedPlateTipShearPlaneStress';
% caseName = 'gammaStructureMixedElementsPlaneStress';
% caseName = 'NACA2412_AoA5_CSD';
 caseName = 'PlateInPlaneStressSteadyState';
```

Figure 17

Now, everything is set and the problem is ready to be solved. Hit F5 or run for matlab to start the calculations. Matlab will automatically generate two figures. The first figure shows the initial configuration of the problem along with the forces applied and the second one shows deformation and stresses. For more post-processing options we open the output .vtk file in Paraview.

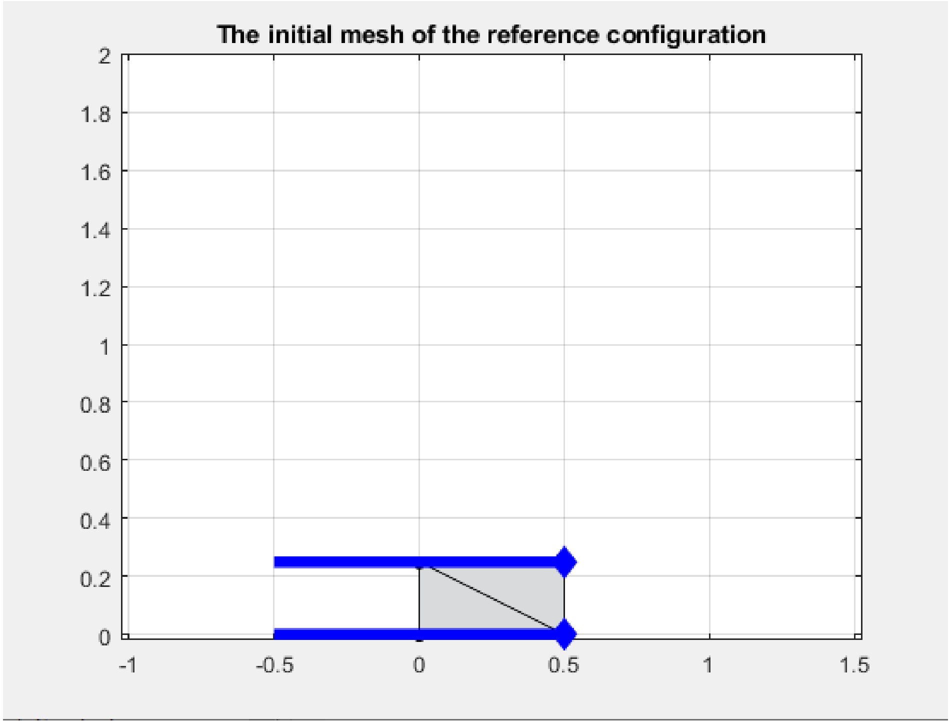


Figure 18

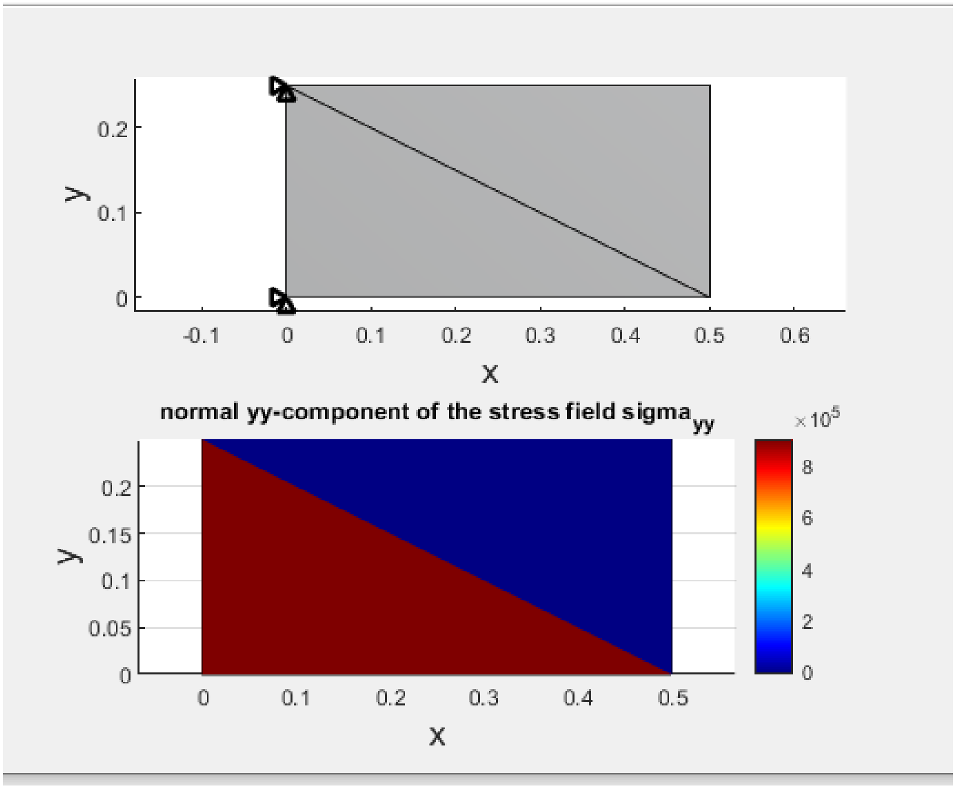
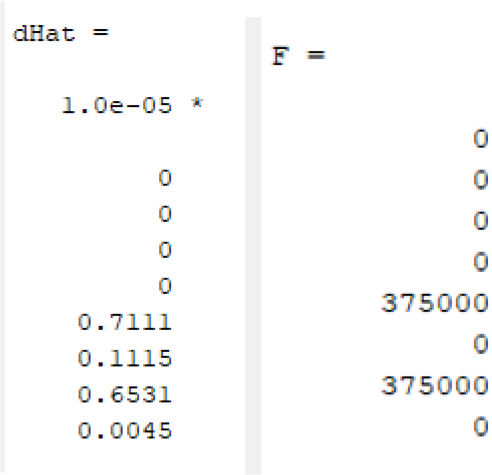


Figure 19

We can print the results inside matlab through the command window. By typing ‘dHat’ we get the displacement vector and by typing F the user can get the force vector.



```
dHat =

   1.0e-05 *

         0
         0
         0
         0
    0.7111
    0.1115
    0.6531
    0.0045
```

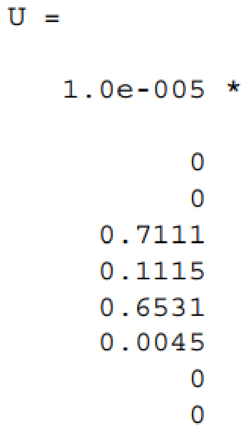
```
F =

           0
           0
           0
           0
      375000
           0
      375000
           0
```

Figure 20

From a quick view we can easily validate that the Dirchlett boundary condition on the left end of the plate has been imposed, since the corresponding displacements are set to zero. Also, since the thickness of the plate is 0.25m we can equally distribute the uniform force in the right end to the two edges of the plate. The calculation is simple:

- Fdistributed = (3000000 N/m2) * (0.5 m) * (0.25 m) = 375000 N

As the figure above shows, the force vector is calculated as expected. We can further validate our results by comparing the displacement vector calculated from cane and the one found in [3]. We can easily see that the two displacement vectors are the same.



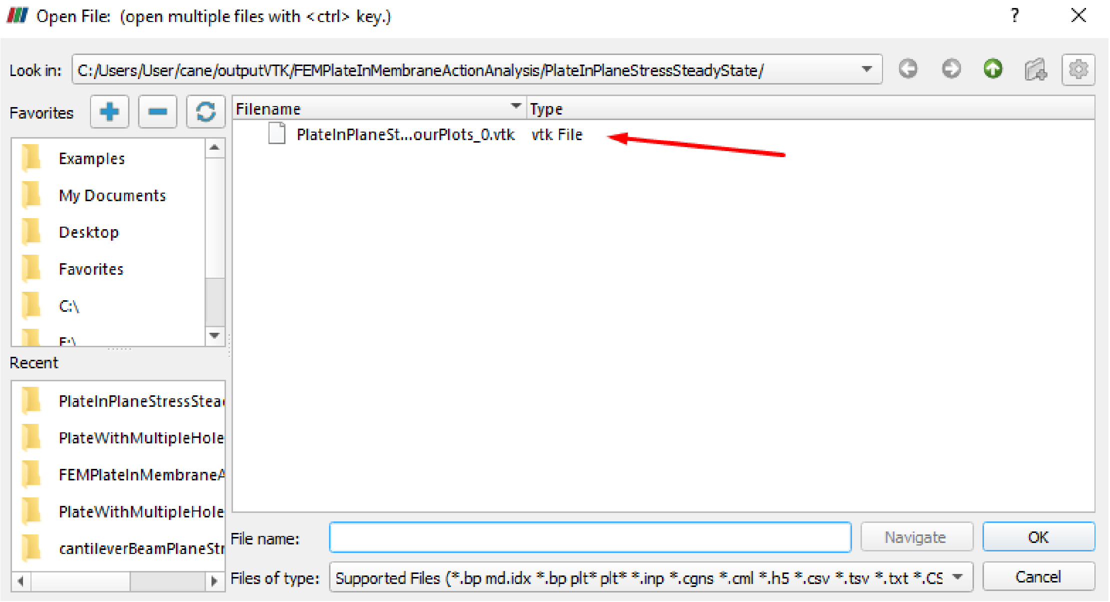
```
U =

   1.0e-005 *

         0
         0
    0.7111
    0.1115
    0.6531
    0.0045
         0
         0
```

Figure 21

2.1.3. Post-Processing with Paraview

At this point, the problem has been solved and the output files have been prepared. We can open the output files with Paraview, an open source post-processing software for better visualization of the results.

The first step is to open Paraview and then choose the appropriate VTK file. The file is located in cane repository in the path /cane/outpoutVTK/FEMPlateInMembraneActionAnalysis/PlateInPlaneStressSteadyState.



Open File: (open multiple files with <ctrl> key.)

Look in: C:/Users/User/cane/outputVTK/FEMPlateInMembraneActionAnalysis/PlateInPlaneStressSteadyState/

Filename	Type
PlateInPlaneSt...ourPlots_0.vtk	vtk File

File name:

Files of type: Supported Files (*.bp md.idx *.bp plt* plt* *.inp *.cgns *.cml *.h5 *.csv *.tsv *.txt *.CS

Figure 22

Then we select Apply from the Properties tab:

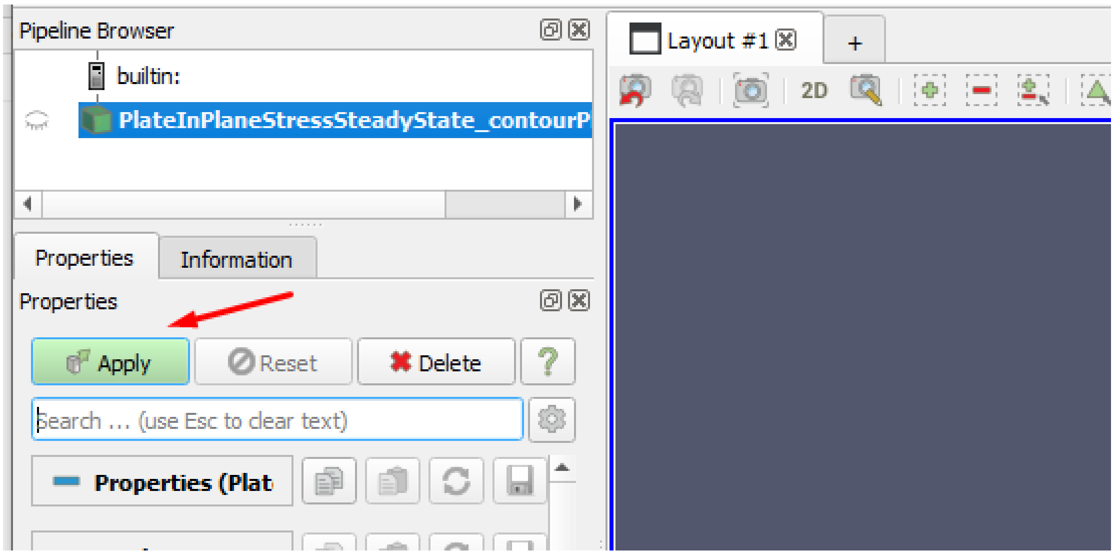


Figure 23

In order to visualize the resultings displacements of structure we select the following options from Paraview's interface, as shown in the figure below:

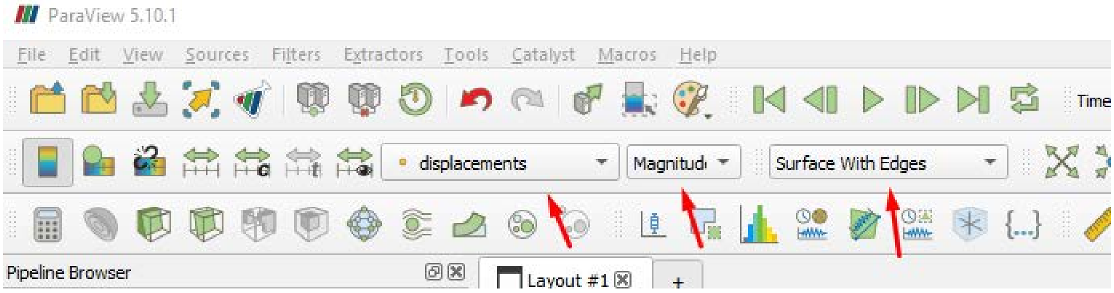


Figure 24

The corresponding displacement field is shown in the figure below:

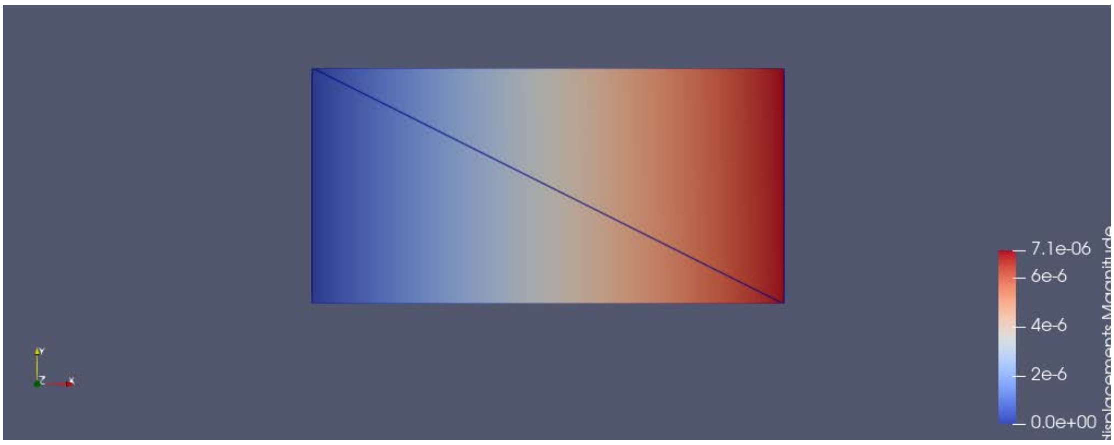


Figure 25

2.2. Modal Analysis of a multihole panel

A panel with four circular holes is considered to demonstrate cane's modal analysis capabilities. The complete description of the problem can be found in [4].The problem has the following parameters:

- Height: H = 1 m
- Length: L = 5 m
- Radius of holes: r = 0.3 m
- Modulus of elasticity: E = 206 GPa
- Mass density: ρ = 7800 kg/m3
- Poisson's ratio v = 0.3

The geometry of the problem is presented in the figure below:

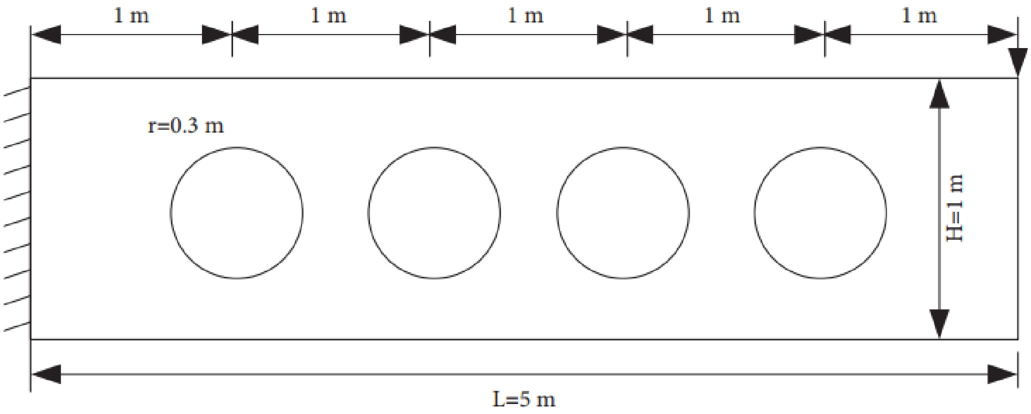
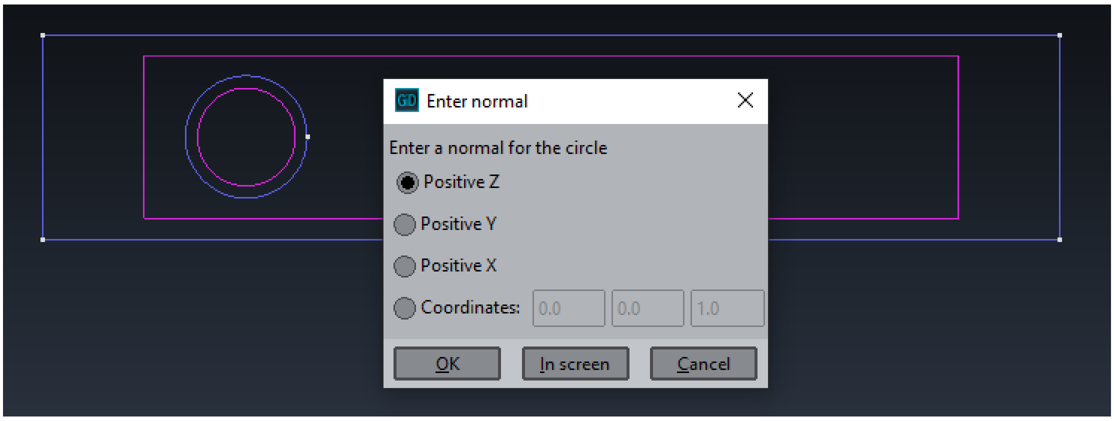


Figure 26

2.2.1 Preprocessing with GID

The pre-processing stage of the problem doesn't differ so much from the previous steady state problem, so now we will focus only on the different features of GiD that we will need to use in order to properly describe the problem.

Firstly we have to create a rectangle object with two points, namely (0,0) and (5,1). After that, we need to create four circles. These circles will have a radius of 0.3m and the distance between them will be set at 1m. The normal vector of all the circles have to point in the positive Z direction.



Enter normal

Enter a normal for the circle

● Positive Z

○ Positive Y

○ Positive X

○ Coordinates: 0.0 0.0 1.0

Figure 27

Now that the four circles have been designed, we have to pass the information that these circles represent holes. This can be done with the Boolean surface operations of GiD. The main idea is that we subtract from the rectangle surface, the four circular surfaces.

In order to do this, we select Geometry -> Edit -> Surface Boolean op. ->Subtraction

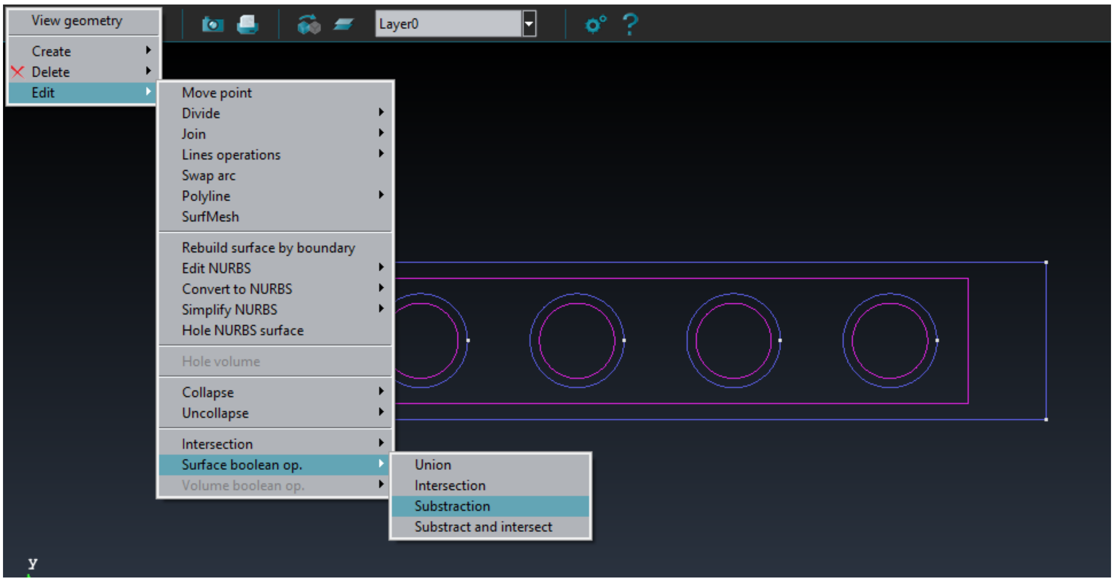


Figure 28

At this point we have to select the surface from which we will subtract the other surfaces, namely the rectangle surface. After selection hit the ESC button. Then select the four circular surfaces and hit the ESC button again. If the operation was successful GiD will return the message 'Surfaces subtraction finished'. The resulting surface is shown below.

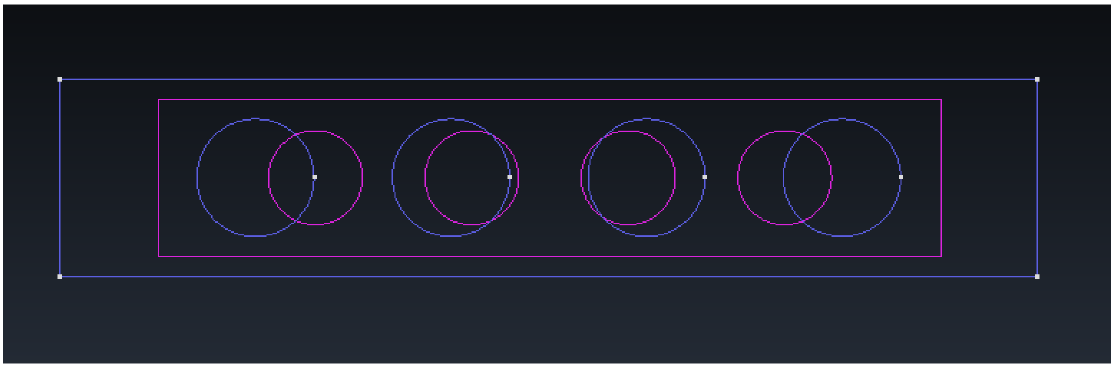
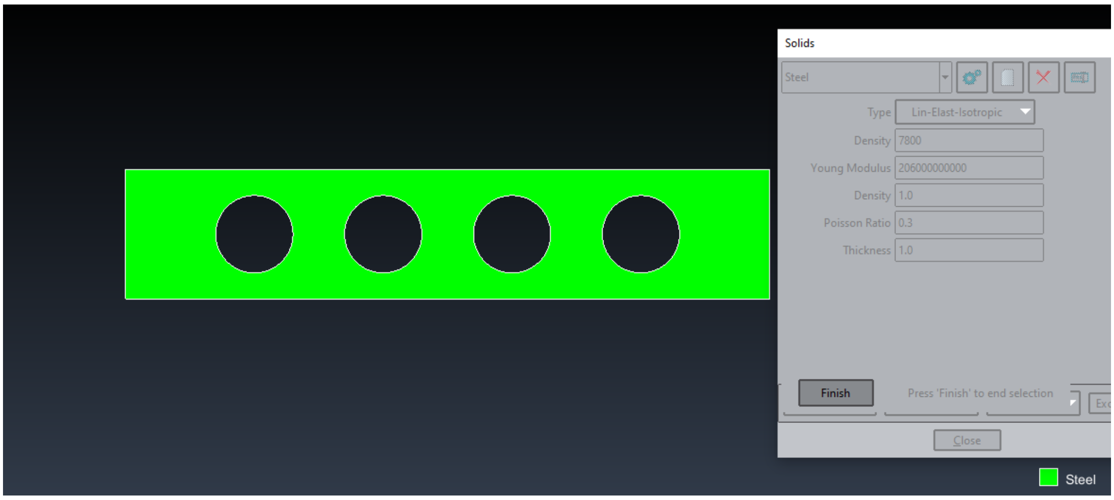


Figure 29

A good way to validate that our operation was successful is to see whether you can apply material property to the circular surfaces. The figure below clearly shows that the circular holes are void, and consequently the program prevents you from assigning material properties to them.



Solids

Steel

Type: Lin-Elast-Isotropic

Density: 7800

Young Modulus: 206000000000

Density: 1.0

Poisson Ratio: 0.3

Thickness: 1.0

Press 'Finish' to end selection

Figure 30

For the discretization of the structure we choose to use quadrilateral plane stress finite elements. To do that, we select Mesh -> Element type -> Quadrilateral

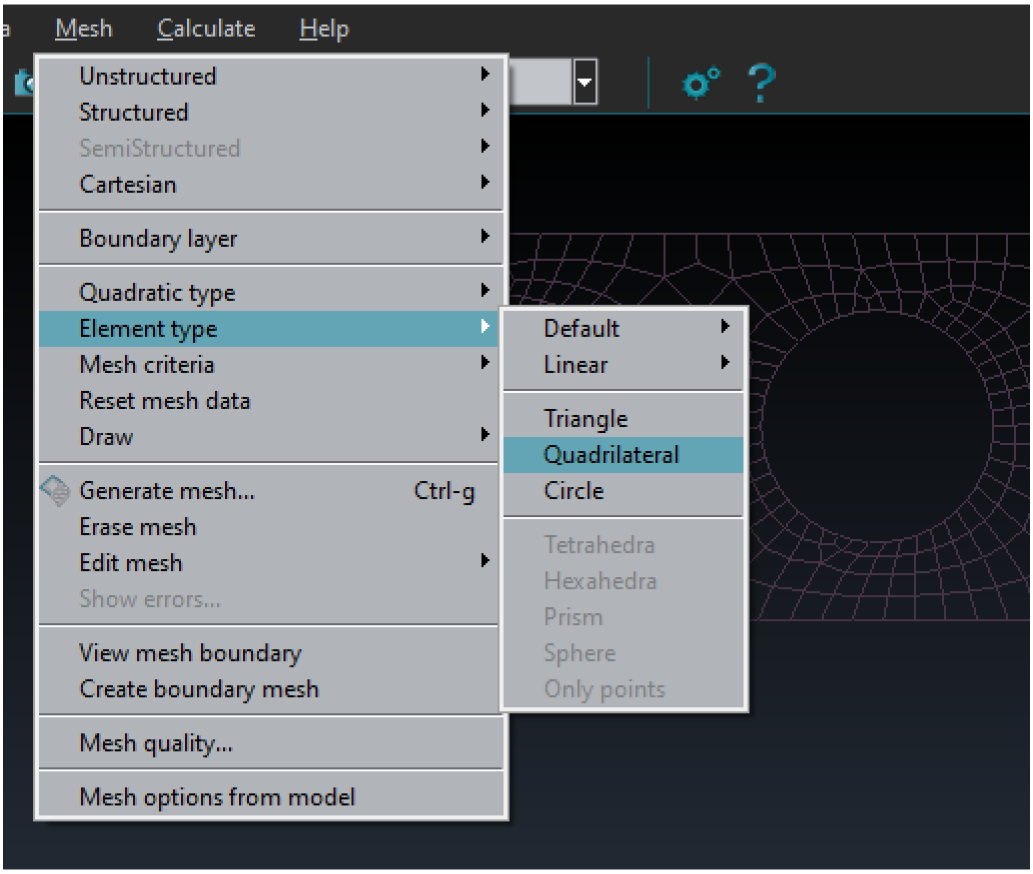


Figure 31

Finally, we generate the mesh and press F5 to produce the .dat file that will be processed by the cane matalb script for modal analysis

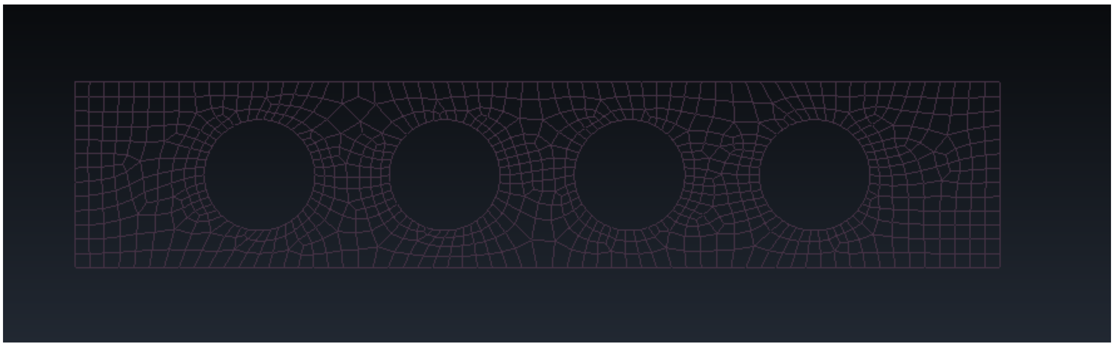
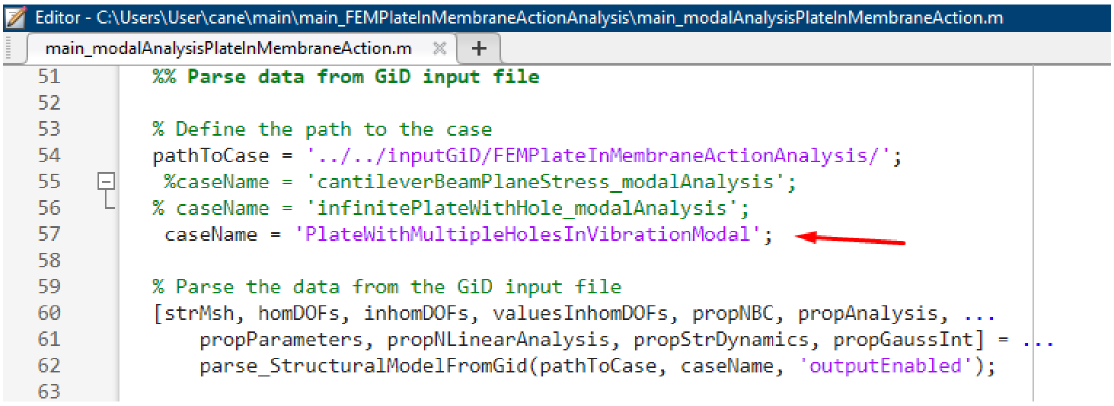


Figure 32

2.2.2. Solving with cane Multiphysics

At this point, we open the matlab script in cane repository which will perform the modal analysis. From the cane main repository we select main -> main_FEMPlateInMembraneActionAnalysis ->

main_modalAnalysisPlateInMembraneAction.m. As we did in the previous example, we now have to define a new case with the name of the generated data file from GiD, as shown in the figure below:



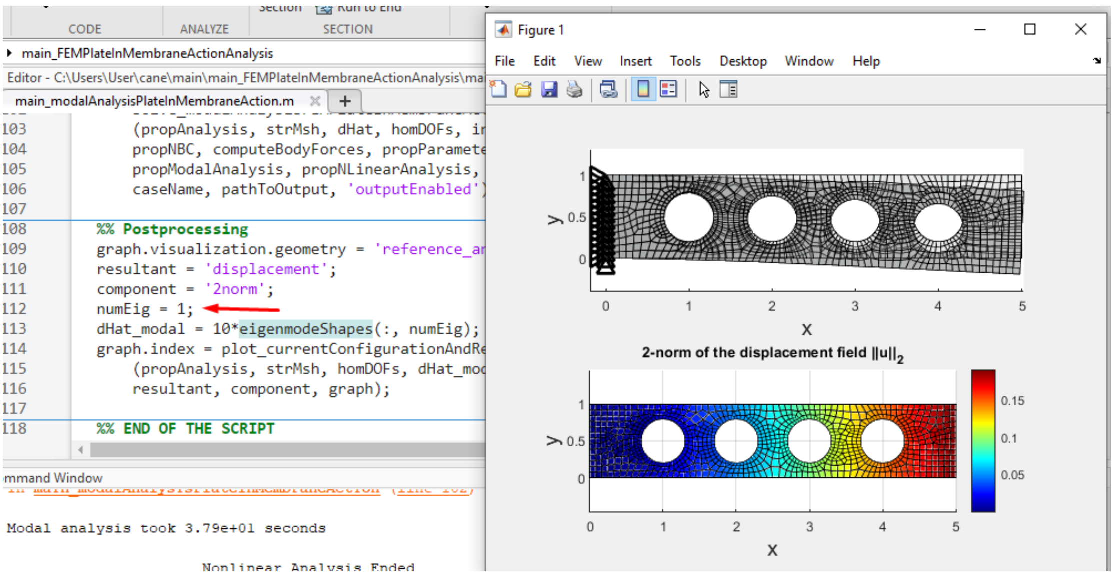
Editor - C:\Users\User\cane\main\main_FEMPlateInMembraneActionAnalysis\main_modalAnalysisPlateInMembraneAction.m
main_modalAnalysisPlateInMembraneAction.m

```
    %% Parse data from GiD input file

    % Define the path to the case
    pathToCase = '../../inputGiD/FEMPlateInMembraneActionAnalysis/';
     %caseName = 'cantileverBeamPlaneStress_modalAnalysis';
    % caseName = 'infinitePlateWithHole_modalAnalysis';
     caseName = 'PlateWithMultipleHolesInVibrationModal';

    % Parse the data from the GiD input file
    [strMsh, homDOFs, inhomDOFs, valuesInhomDOFs, propNBC, propAnalysis, ...
        propParameters, propNLinearAnalysis, propStrDynamics, propGaussInt] = ...
        parse_StructuralModelFromGid(pathToCase, caseName, 'outputEnabled');
```

Figure 33

The modal analysis is independent of loads, so we don't have to specify any loading condition. We can now run the analysis by pressing F5.

Matlab will automatically produce a figure that shows the first eigenmode that corresponds to the first natural eigenfrequency.



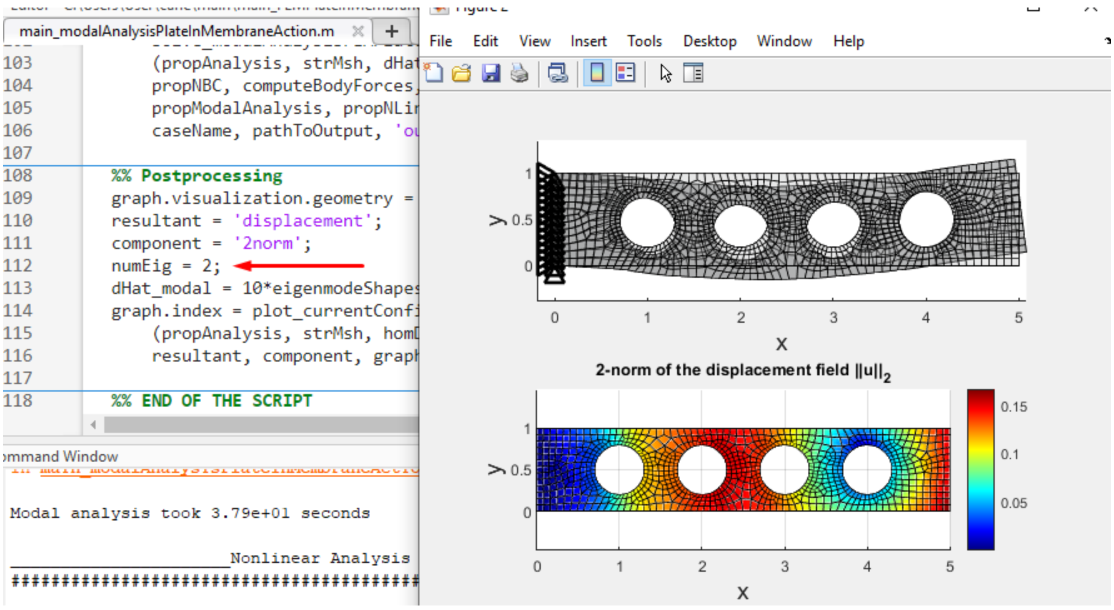
Section
Run to End
CODE
ANALYZE
SECTION
main_FEMPlateInMembraneActionAnalysis
Editor - C:\Users\User\cane\main\main_FEMPlateInMembraneActionAnalysis\ma
main_modalAnalysisPlateInMembraneAction.m
Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
1
0.5
0
y
0
1
2
3
4
5
x
2-norm of the displacement field ||u||2
1
0.5
0
y
0
1
2
3
4
5
x
0.15
0.1
0.05
mmand Window

```
        (propAnalysis, strMsh, dHat, homDOFs, i
        propNBC, computeBodyForces, propParamete
        propModalAnalysis, propNLinearAnalysis,
        caseName, pathToOutput, 'outputEnabled')

    %% Postprocessing
    graph.visualization.geometry = 'reference_ar
    resultant = 'displacement';
    component = '2norm';
    numEig = 1;
    dHat_modal = 10*eigenmodeShapes(:, numEig);
    graph.index = plot_currentConfigurationAndRe
        (propAnalysis, strMsh, homDOFs, dHat_moc
        resultant, component, graph);

    %% END OF THE SCRIPT
```

```
Modal analysis took 3.79e+01 seconds

                    Nonlinear Analysis Ended
```

Figure 34

We can visualize any other eigenfrequency by changing the value of numEig variable. We now set it to 2 for illustration purposes and press Run Section while inside the Postprocessing section.

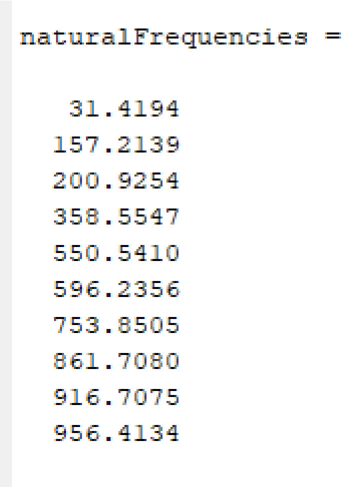


```
        (propAnalysis, strMsh, dHat
        propNBC, computeBodyForces,
        propModalAnalysis, propNLir
        caseName, pathToOutput, 'ou

    %% Postprocessing
    graph.visualization.geometry =
    resultant = 'displacement';
    component = '2norm';
    numEig = 2;
    dHat_modal = 10*eigenmodeShapes
    graph.index = plot_currentConfi
        (propAnalysis, strMsh, homD
        resultant, component, graph

    %% END OF THE SCRIPT
```

```
Modal analysis took 3.79e+01 seconds

_____________________________Nonlinear Analysis
##########################################
```

Figure 35

The user can print the natural frequencies calculated through the command window by typing 'naturalFrequencies'. The figure below shows the first 10 natural frequencies. To validate our results, we get the values of natural frequencies as computed in [4].



```
naturalFrequencies =

   31.4194
  157.2139
  200.9254
  358.5547
  550.5410
  596.2356
  753.8505
  861.7080
  916.7075
  956.4134
```

Figure 36

Solution Type	Mode 1	Mode 2	Mode 3	Mode 4
Reference Solutions	31.811	150.19	209.31	330.64
Cane Solutions	31.42	157.21	200.92	358.55

The small differences in the values is due to the discretization of the structure. The reference solution uses more quadrilateral elements and thus, a more accurate solution is obtained.

2.2.3. Post-Processing with Paraview

We can visualize the resulting eigenmodes in paraview by opening the .vtk output file. The first four eigenmodes corresponding to the first four natural eigenfrequencies are shown below.

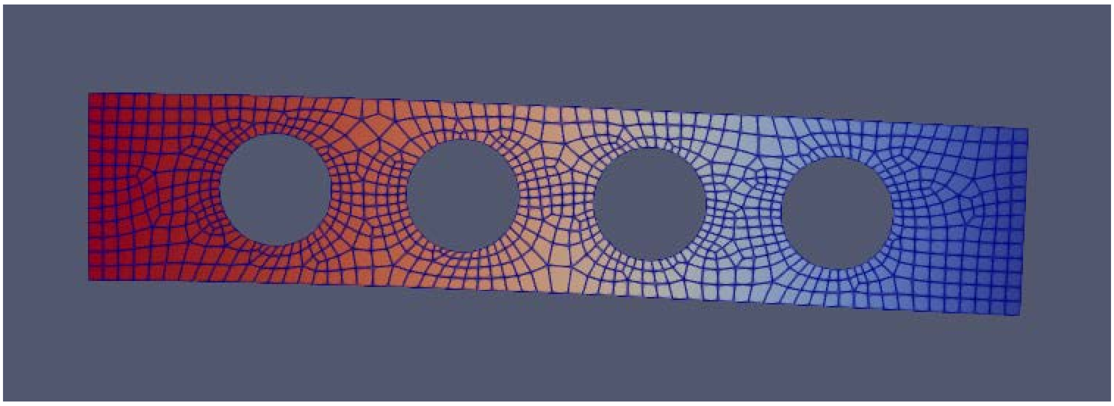


Figure 37

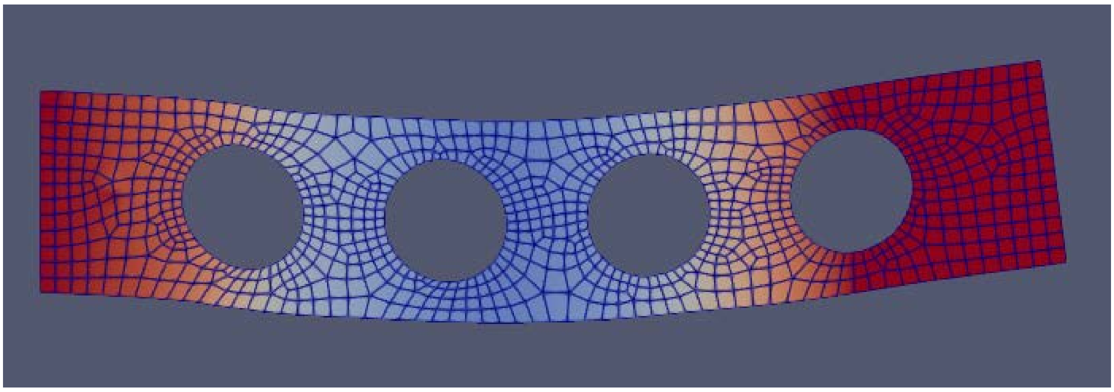


Figure 38

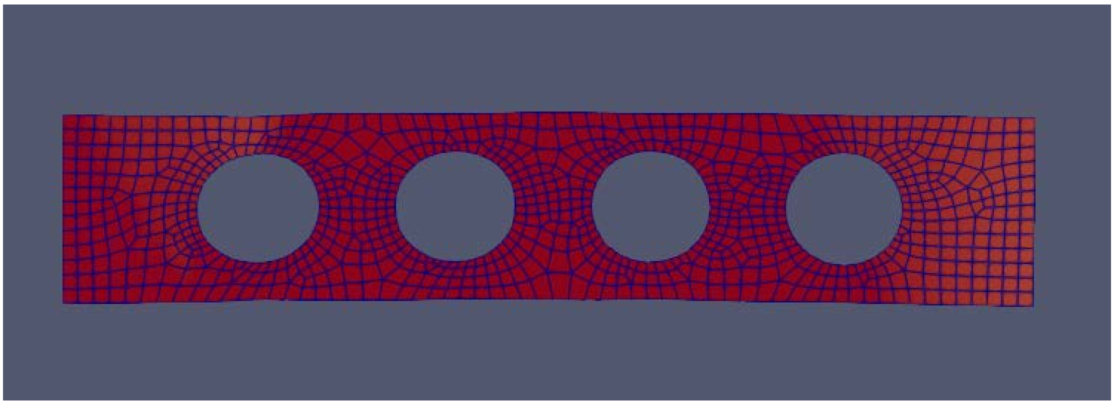


Figure 39

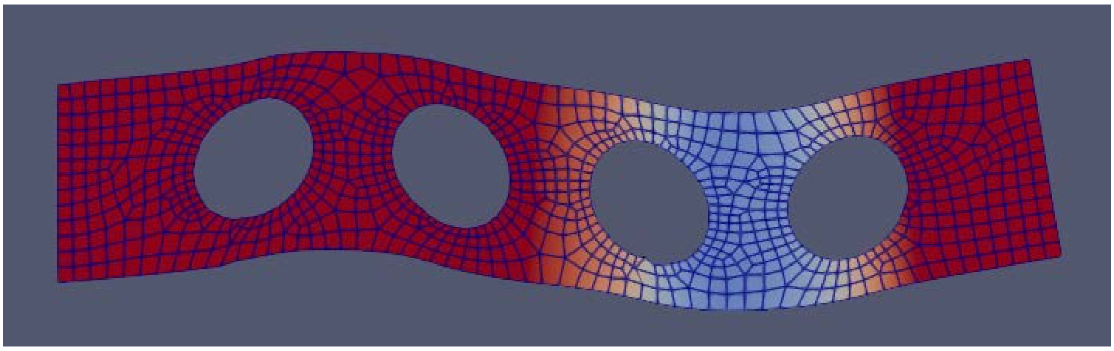


Figure 40

The user can navigate through the eigenmodes using the ‘Next Frame’ and ‘Previous Frane’ buttons

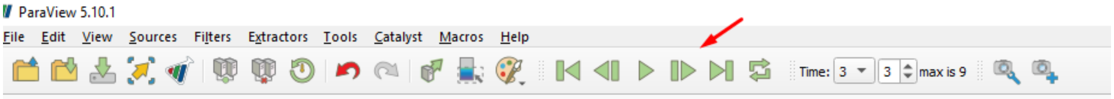


Figure 41

2.3. Transient Analysis of a multihole panel

The third problem [4] addressed in this chapter is about the forced vibration of the same multihole panel presented in the previous example. A time-dependent force p(t) = 10000 * sin(199.77 * π * t) acts on the right end of the panel, as presented below. The same mesh and material properties are applied in this problem as well. We are going to monitor the transient response of the vertical displacement of the bottom right node and compare it with the reference response.

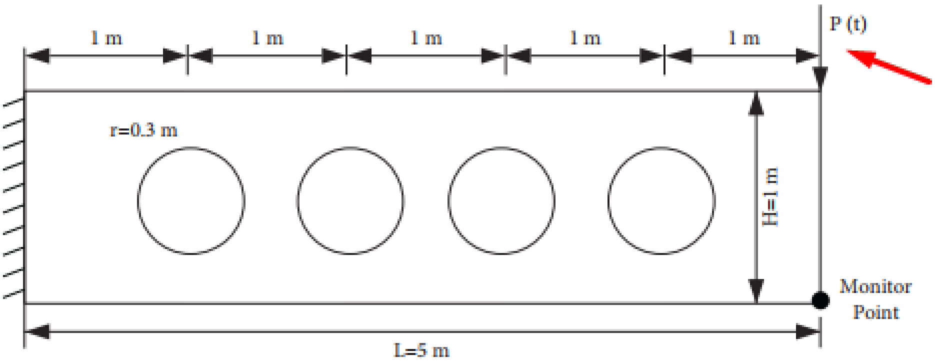
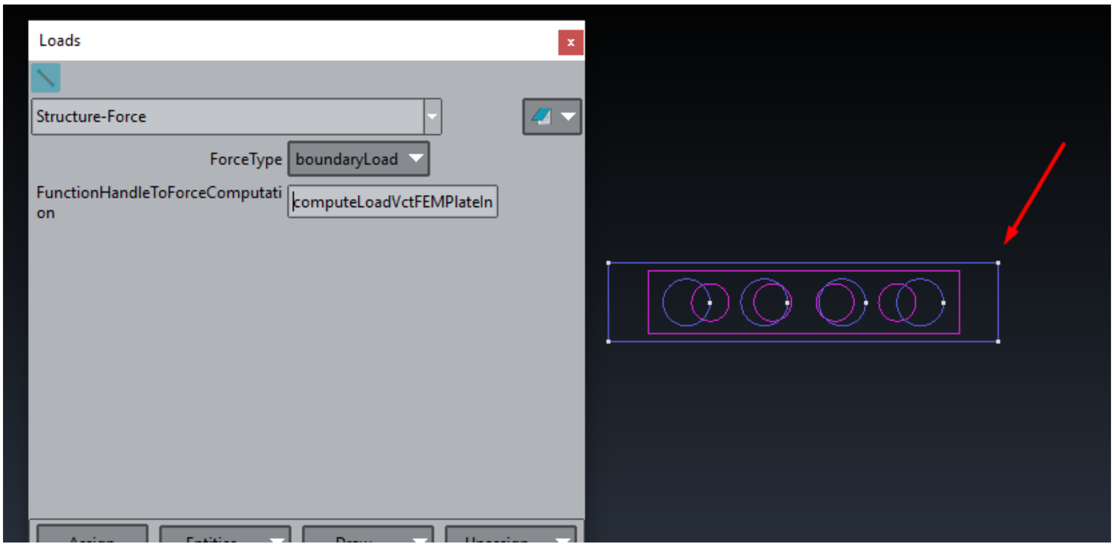


Figure 42

2.3.1 Preprocessing with GID

In the previous example, we didn’t have to create a loading boundary condition because the modal analysis is independent of loads. In this occasion, we have to create a new condition inside GiD.



Loads

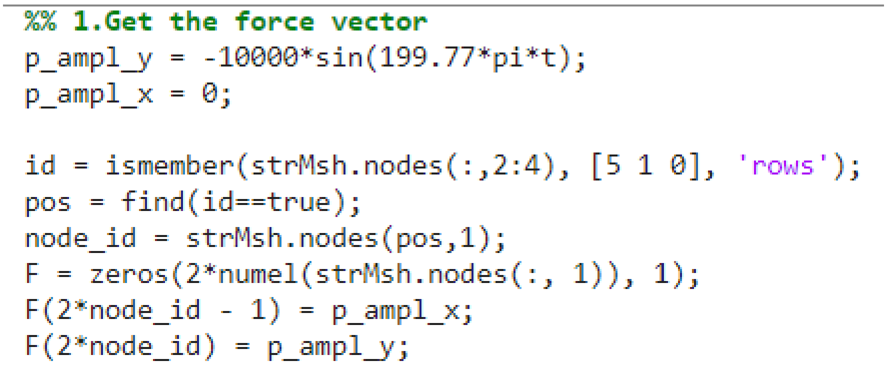
Structure-Force

ForceType boundaryLoad

FunctionHandleToForceComputation computeLoadVctFEMPlateIn

Figure 43

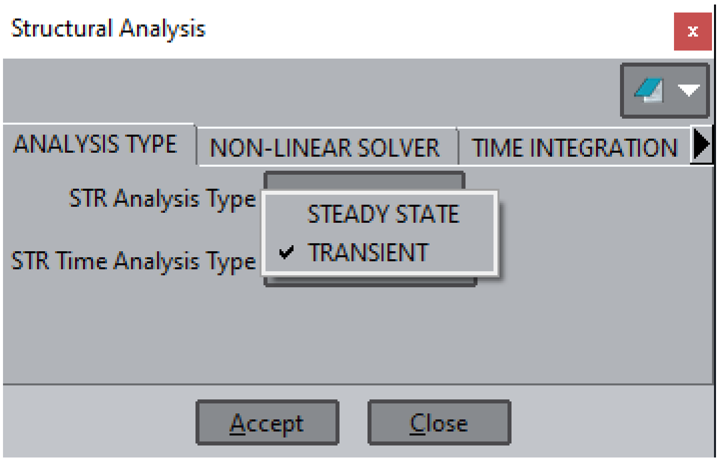
The cane framework doesn’t have a function for calculating nodal forces on specific nodes, so there is a need to create a new matlab script. In this script, we will have to search our mesh and find the node we want by checking if there exists a node in the mesh with the exact coordinates given. In this occasion, the node we has coordinates (5,1,0) which is the top right node. The following code finds the node and applies the force amplitude in the corresponding components of the force vector. The name of the matlab script is computeLoadVctFEMPlateInMembraneActionPointLoad and through GiD we created a function handle that points to that script, so that when the load vector is calculated during the analysis, this function will be used.



```
%% 1.Get the force vector
p_ampl_y = -10000*sin(199.77*pi*t);
p_ampl_x = 0;

id = ismember(strMsh.nodes(:,2:4), [5 1 0], 'rows');
pos = find(id==true);
node_id = strMsh.nodes(pos,1);
F = zeros(2*numel(strMsh.nodes(:, 1)), 1);
F(2*node_id - 1) = p_ampl_x;
F(2*node_id) = p_ampl_y;
```

Figure 44

Furthermore, we have to tell GiD that the analysis is going to be transient. Select Data-> Structural Analysis. In the window that pops up choose transient analysis.



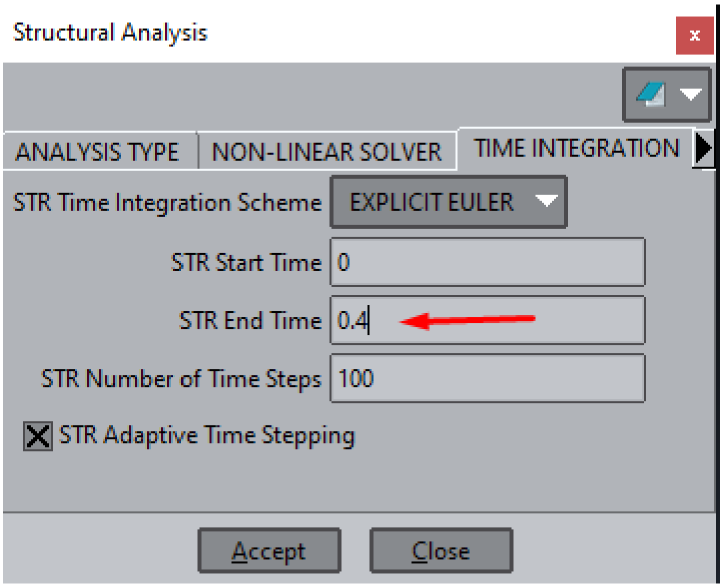
STR Analysis Type

STR Time Analysis Type

STEADY STATE
✓ TRANSIENT

Figure 45

Moving to the time integration tab, we set the end time of the analysis to 0.4. This step is done in order to produce relevant results to those in the solution of the problem in [4].



STR Time Integration Scheme: EXPLICIT EULER

STR Start Time: 0

STR End Time: 0.4

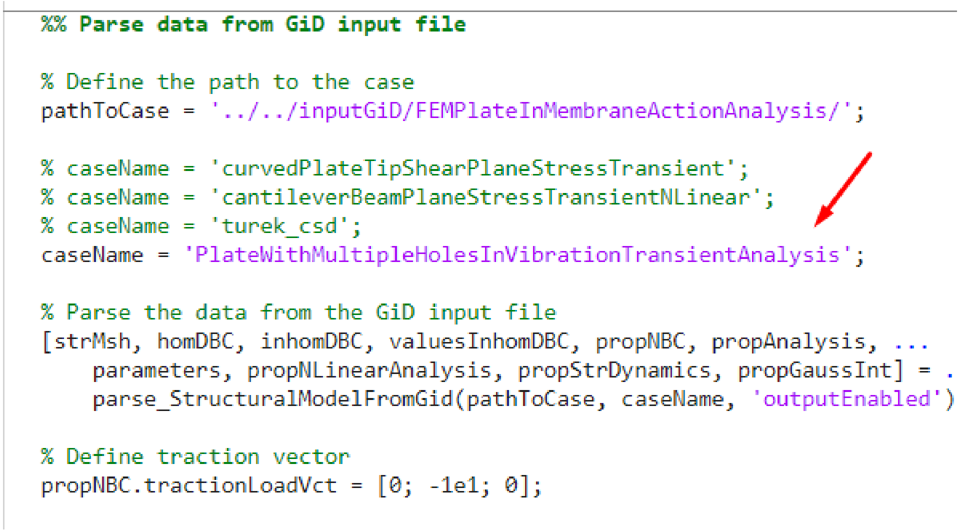
STR Number of Time Steps: 100

☒ STR Adaptive Time Stepping

Figure 46

2.3.2. Solving with cane Multiphysics

Once again, we create a new case named after the data file produced by GiD.



```
%% Parse data from GiD input file

% Define the path to the case
pathToCase = '../../inputGiD/FEMPlateInMembraneActionAnalysis/';

% caseName = 'curvedPlateTipShearPlaneStressTransient';
% caseName = 'cantileverBeamPlaneStressTransientNLinear';
% caseName = 'turek_csd';
caseName = 'PlateWithMultipleHolesInVibrationTransientAnalysis';

% Parse the data from the GiD input file
[strMsh, homDBC, inhomDBC, valuesInhomDBC, propNBC, propAnalysis, ...
    parameters, propNLinearAnalysis, propStrDynamics, propGaussInt] = .
    parse_StructuralModelFromGid(pathToCase, caseName, 'outputEnabled')

% Define traction vector
propNBC.tractionLoadVct = [0; -1e1; 0];
```

Figure 47

Everything is set, so we press F5 to run the analysis.

2.3.3. Post-Processing with Paraview

References

[1] GiD Simulation, GiD Simulation, [Ηλεκτρονικό]. Available: https://www.gidsimulation.com/.

[2] ParaView, [Ηλεκτρονικό]. Available: https://www.paraview.org/.

[3] P. I.Kattan, MATLAB Guide to Finite Elements_An interactive Approach, 2008.

[4] C. S. Y. Y. Nan Ye, «Free and Forced Vibration Analysis in Abaqus Based on the Polygonal Scaled Boundary Finite Element Method,» *Hindawi - Advances in Civil Engineering,* τόμ. 2021, p. 17, 2021.