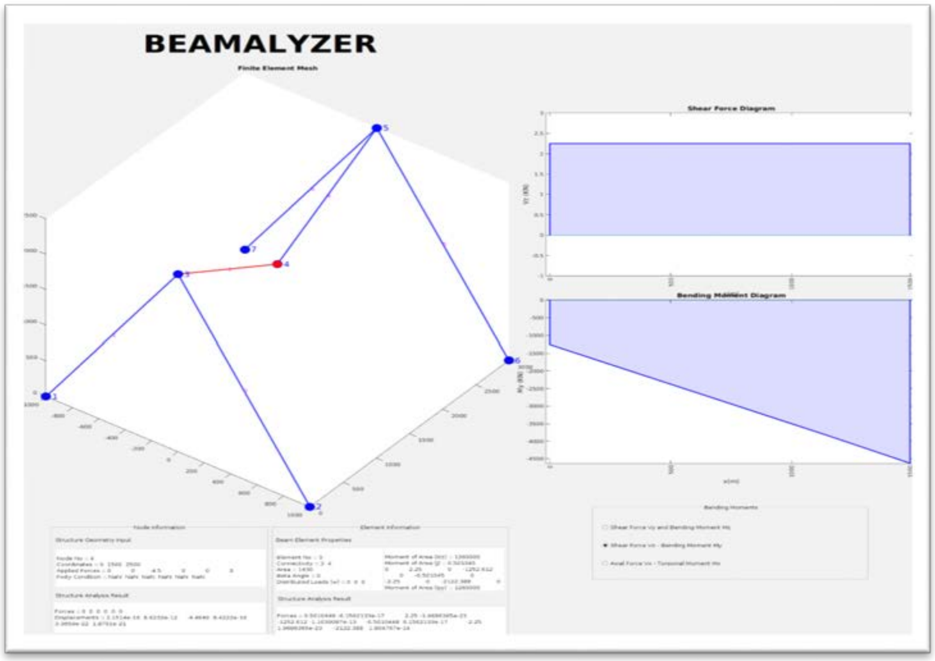
BEAMALYZER



12/3/2015

ECI 211 Final Project Report

Beamalyzer is a 3D finite element program to solve space frame problems of 1st Order. It's a GUI based program developed in Matlab with abilities of visualizing structure, deformed mesh and individual members deflection diagram and shear force & bending moment diagram

BEAMALYZER

ECI 211 FINAL PROJECT REPORT

INTRODUCTION

Beamalyzer is a 3-D finite element program for solving space frame problems. It's a GUI based program developed in Matlab. Some of the main features of this program are

- Very flexible input and output
- Visualization of geometry(mesh) and deformed structure
- Ability to take ***distributed loads and settlement at nodes***
- Shear Force and Bending moment diagram
- Deflection Diagram of beams
- Ability to save files and reopen it later
- Easily extendable to accommodate temperature loads

INSIDE THE CODE

Beamalyzer is developed in Matlab using various subroutines. The basic flow of the program is as shown below

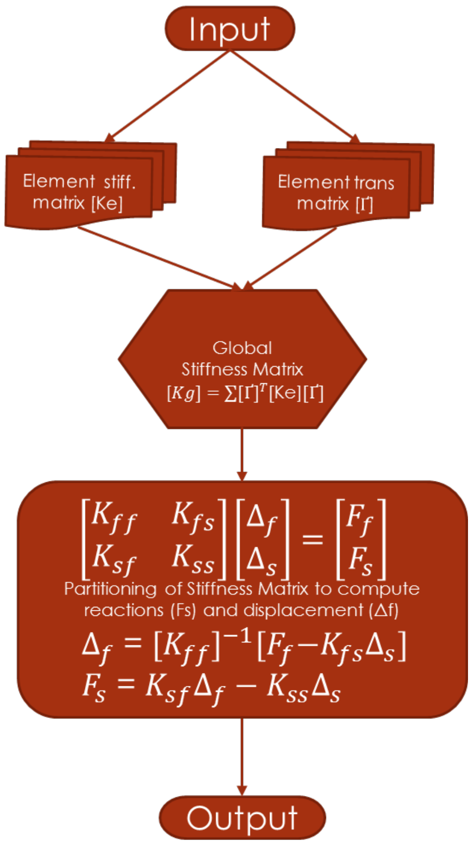

Input
Element stiff. matrix [Ke]
Element trans matrix [Γ]
Global Stiffness Matrix $[Kg] = \sum[\Gamma]^T[Ke][\Gamma]$
$\begin{bmatrix} K_{ff} & K_{fs} \\ K_{sf} & K_{ss} \end{bmatrix}\begin{bmatrix} \Delta_f \\ \Delta_s \end{bmatrix} = \begin{bmatrix} F_f \\ F_s \end{bmatrix}$
Partitioning of Stiffness Matrix to compute reactions (Fs) and displacement (Δf)
$\Delta_f = [K_{ff}]^{-1}[F_f - K_{fs}\Delta_s]$
$F_s = K_{sf}\Delta_f - K_{ss}\Delta_s$
Output

FIGURE 1 BASIC PROGRAM FLOW IN MATLAB

INPUT

The input basically comprises of three parts as shown below. Along with each part the matrix-variable name in Matlab are also given in brackets.

- Defining nodal geometry and forces
  - Number of nodes (**nnodes**)
  - Coordinates (***coord***)
  - Nodal Forces (***concen***)
  - Fixities (*fixity*)
- Defining element and its properties
  - Number of Elements (**nele**)
  - Connectivity (***ends***)
  - Beta Angle (***beta_ang***)
  - Distributed Load (***w***)
- Defining material and section properties
  - Area of the crossection (***A***)
  - Moment of Area about z-axis (***Izz***)
  - Moment of Area about y-axis (***Iyy***)
  - Moment of Area about x-axis (***J***)
  - Young's Modulus (***E***)
  - Poisons ratio (***v***)

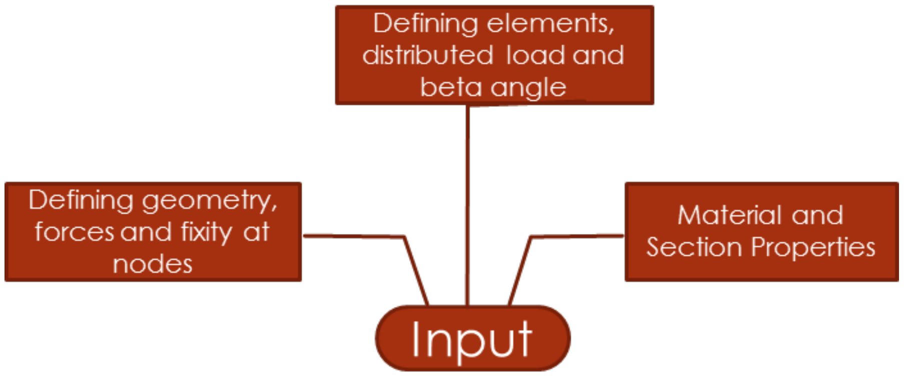

Defining elements, distributed load and beta angle
Defining geometry, forces and fixity at nodes
Material and Section Properties
Input

FIGURE 2 INPUT STRUCTURE FOR THE PROGRAM

ASSEMBLY OF GLOBAL STIFFNESS MATRIX AND FORCE VECTOR

Once the program has all the inputs with it, the next step is to create local stiffness matrix and transformation matrix for each individual elements and finally assemble it to generate the global stiffness matrix.

In order to take in account of the **distributed loads**, equivalent nodal loads are computed inside the program for each beam and are then added to the force vector.

And finally, the global stiffness matrix is portioned to calculate the reactions as supports and deflections at free joints.

Matlab subroutine ***Finiteleyzer.m*** *takes* all the input and performs the task. The code snippet with comments is shown below.

Finitelyzer.m – main program which carries out the finite element analysis

```
function [DEFL, REACT, ELE_FOR, AFLAG]=Finiteleyzer( nnodes, nele, coord, concen, fixity, ends, A, Izz, Iyy, J, E, v, beta_ang, w)

   t = waitbar(0,'Please wait...');waitbar(0.1);

   DEFL=zeros(nnodes,6);
   REACT=zeros(nnodes,6);
   ELE_FOR=zeros(nele,12);

   % creating dof_id for each node
   node_id = 1:nnodes*6; node_id = reshape(node_id,6,nnodes)';waitbar(0.15);

   % creating mem_id for each element
   mem_id = zeros(nele,12); mem_id(:,1:6) = node_id(ends(:,1),:); mem_id(:,7:12) = node_id(ends(:,2),:);

   % creating free_dof and fixed_dof vectors
   D_vector = fixity'; D_vector = D_vector(:); free_dof = find(isnan(D_vector)); fixed_dof = find(D_vector==0);waitbar(0.2);

   % finding the length of element
   Length = sqrt(sum((coord(ends(:,1),:)'-coord(ends(:,2),:)').^2))';waitbar(0.3);

   % finding x axis of the beam
   x_axis = (coord(ends(:,2),:)'-coord(ends(:,1),:)')'./([Length(:,1),Length(:,1),Length(:,1)]);

   % Building local_stiffness_stack
   k_local_stack = zeros(nele,12,12);waitbar(0.35);
   for i=1:nele
```

```
        k_local_stack(i,:,:) = estiff(A(i,1),Izz(i,1),Iyy(i,1),J(i,1),E(i,1),v(i,1),Length(i,1));% reshape(k_local_stack(i,:,:),12,12);
    end

    % Building local_global_transformation_stack
    k_trans_stack = zeros(nele,12,12);waitbar(0.5);
    for i=1:nele
        k_trans_stack(i,:,:) = etran(beta_ang(i,1),x_axis(i,:)'); %reshape(k_trans_stack(i,:,:),12,12)
    end

    % Building global_stiffness_stack
    k_global_stack = zeros(nele,12,12);waitbar(0.6);
    for i=1:nele
        k_global_stack(i,:,:) = (reshape(k_trans_stack(i,:,:),12,12))'*(reshape(k_local_stack(i,:,:),12,12))*(reshape(k_trans_stack(i,:,:),12,12));
    end

    % Building total stiffness matrix
    k_total = zeros(nnodes*6,nnodes*6);waitbar(0.7);
    for i=1:nele
        k_total(mem_id(i,1:6),mem_id(i,1:6))   = k_total(mem_id(i,1:6),mem_id(i,1:6))   + reshape(k_global_stack(i,1:6,1:6),6,6)  ;
        k_total(mem_id(i,7:12),mem_id(i,1:6))  = k_total(mem_id(i,7:12),mem_id(i,1:6))  + reshape(k_global_stack(i,7:12,1:6),6,6) ;
        k_total(mem_id(i,1:6),mem_id(i,7:12))  = k_total(mem_id(i,1:6),mem_id(i,7:12))  + reshape(k_global_stack(i,1:6,7:12),6,6) ;
        k_total(mem_id(i,7:12),mem_id(i,7:12)) = k_total(mem_id(i,7:12),mem_id(i,7:12)) + reshape(k_global_stack(i,7:12,7:12),6,6);
    end

   %Addding Distributed Loads, Temperature Loads .. etc to Nodes
    concen_dl = zeros(nnodes,6);waitbar(0.8);
    for i=1:nele
        l = Length(i);
        F_d = [-w(i,1)*l/2;-w(i,2)*l/2;-w(i,3)*l/2;0;-w(i,3)*l^2/12;-w(i,2)*l^2/12;-w(i,1)*l/2;-w(i,2)*l/2;-
w(i,3)*l/2;0;w(i,3)*l^2/12;w(i,2)*l^2/12];
        F_d = (reshape(k_trans_stack(i,:,:),12,12))'*(-F_d);
        concen_dl(ends(i,1),:) = concen_dl(ends(i,1),:) + F_d(1:6)' ; concen_dl(ends(i,2),:) = concen_dl(ends(i,2),:) + F_d(7:12)';
    end

    % Total Load vector
    concen_tl = concen + concen_dl; waitbar(0.85);

    % Partitioning the matrix
    kff = k_total(free_dof,free_dof);
    kfs = k_total(free_dof,fixed_dof);
    ksf = k_total(fixed_dof,free_dof);
    kss = k_total(fixed_dof,fixed_dof);

     if (det(kff)==0)
        errordlg('The structure is under-constrained','Error');
        return
    end

    % Building force vector
    F_vector = concen_tl'; F_vector = F_vector(:);

    D_vector(free_dof) =  kff\(F_vector(free_dof)-kfs*D_vector(fixed_dof));
    F_vector(fixed_dof) = ksf*D_vector(free_dof) + kss*D_vector(fixed_dof);

    % Output matrix
    DEFL = reshape(D_vector,6,nnodes)';
    REACT = reshape(F_vector,6,nnodes)' - concen_tl;waitbar(0.95);

    for i=1:nele
        ELE_FOR(i,:) = (reshape(k_local_stack(i,:,:),12,12))*(reshape(k_trans_stack(i,:,:),12,12))*(D_vector(mem_id(i,1:12)));
        l = Length(i); F_d = [-w(i,1)*l/2;-w(i,2)*l/2;-w(i,3)*l/2;0;-w(i,3)*l^2/12;-w(i,2)*l^2/12;-w(i,1)*l/2;-w(i,2)*l/2;-
w(i,3)*l/2;0;w(i,3)*l^2/12;w(i,2)*l^2/12];
        ELE_FOR(i,:) = ELE_FOR(i,:) + F_d(:)';
    end

    ELE_FOR; AFLAG=1;
end
```

Finitelyzer.m uses two additional sub-routines that help in generating element stiffness matrix (**estiff.m**) and element transformation matrix (**etran.m**). Both subroutines are shown below.

Estiff.m – Program to generate element stiffness matrix

```
function[elk] = estiff(A,Iz,Iy,J,E,v,L)

   % Genearal Stiffness Matrix of Beam.
   %   A = Crossectional-Area of The Beam.
   %   Iz = Moment of Area of Beam about Izz-Axis.
   %   Iy = Moment of Area of Beam about Iyy-Axis.
   %   J  = Torsoinal Moment of Area.
   %   E  = Young's Modulus of Material
   %   v  = Poison's ratio
   %   L  = Length of the Beam


   elk=E* [A/L    ,0          ,0          ,0              ,0          ,0          ,-A/L       ,0          ,0          ,0              ,0          ,0          ;
          0       ,12*Iz/L^3  ,0          ,0              ,0          ,6*Iz/L^2   ,0          ,-12*Iz/L^3 ,0          ,0              ,0          ,6*Iz/L^2   ;
          0       ,0          ,12*Iy/L^3  ,0              ,-6*Iy/L^2  ,0          ,0          ,0          ,-12*Iy/L^3 ,0              ,-6*Iy/L^2  ,0          ;
          0       ,0          ,0          ,J/(2*(1+v)*L)  ,0          ,0          ,0          ,0          ,0          ,-J/(2*(1+v)*L) ,0          ,0          ;
          0       ,0          ,-6*Iy/L^2  ,0              ,4*Iy/L     ,0          ,0          ,0          ,6*Iy/L^2   ,0              ,2*Iy/L     ,0          ;
          0       ,6*Iz/L^2   ,0          ,0              ,0          ,4*Iz/L     ,0          ,-6*Iz/L^2  ,0          ,0              ,0          ,2*Iz/L     ;
          -A/L    ,0          ,0          ,0              ,0          ,0          ,A/L        ,0          ,0          ,0              ,0          ,0          ;
          0       ,-12*Iz/L^3 ,0          ,0              ,0          ,-6*Iz/L^2  ,0          ,12*Iz/L^3  ,0          ,0              ,0          ,-6*Iz/L^2  ;
          0       ,0          ,-12*Iy/L^3 ,0              ,6*Iy/L^2   ,0          ,0          ,0          ,12*Iy/L^3  ,0              ,6*Iy/L^2   ,0          ;
          0       ,0          ,0          ,-J/(2*(1+v)*L) ,0          ,0          ,0          ,0          ,0          ,J/(2*(1+v)*L)  ,0          ,0          ;
          0       ,0          ,-6*Iy/L^2  ,0              ,2*Iy/L     ,0          ,0          ,0          ,6*Iy/L^2   ,0              ,4*Iy/L     ,0          ;
          0       ,6*Iz/L^2   ,0          ,0              ,0          ,2*Iz/L     ,0          ,-6*Iz/L^2  ,0          ,0              ,0          ,4*Iz/L     ];

end
```

Etran.m –Program to generate element transformation matrix

```
function[gamma] = etran(beta_angle,xaxis)

   % Transformation Matrix for 3D frame member.
   %   beta_angle = Crossectional-Area of The Beam.
   %   xaxis      = Moment of Area of Beam about Izz-Axis.

   R_zero = zeros(3,3); e_y = [0;1;0];

   alpha_x = xaxis./sqrt(sum(xaxis.^2));

   if(alpha_x(1)==0 && (alpha_x(2)==1 || alpha_x(2)==-1) && alpha_x(3)==0);
      e_y = [-1;0;0];
   end

   alpha_z = cross(alpha_x,e_y)./sqrt(sum(cross(alpha_x,e_y).^2));
   alpha_y = cross(alpha_z,alpha_x)./sqrt(sum(cross(alpha_z,alpha_x).^2));

   R_axis = [  alpha_x' ; alpha_y' ; alpha_z'];

   R_beta = [  1   ,0                ,0                ;
               0   ,cos(beta_angle)  ,sin(beta_angle)  ;
               0   ,-sin(beta_angle) ,cos(beta_angle)  ];

   R = R_beta * R_axis;

   gamma = [  R        ,R_zero   ,R_zero   ,R_zero  ;
              R_zero   ,R        ,R_zero   ,R_zero  ;
              R_zero   ,R_zero   ,R        ,R_zero  ;
              R_zero   ,R_zero   ,R_zero   ,R       ;];
end
```

The **Finitelyzer.m** subroutine then spits out the output mainly deflection (***DEFL***), reactions (***REACT***) and element forces (**ELE_FOR**). Also it checks for the stability of structure and sends error message if the structure is not stable.

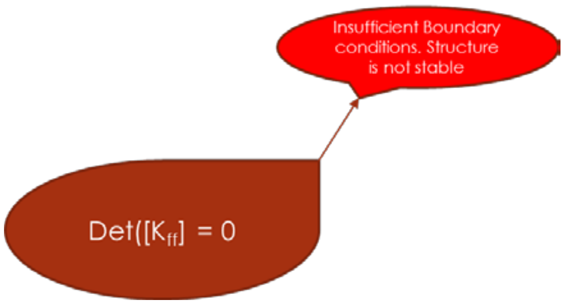


FIGURE 3 CHECKING THE STABILITY OF STRUCTURE

OUTPUT – Reaction at supports, Deflection and element forces

After the partitioning of matrix, the equations are solved to get the reaction at supports, deflection and element forces as commented in the code below.

```
% Partitioning the matrix
kff = k_total(free_dof,free_dof);
kfs = k_total(free_dof,fixed_dof);
ksf = k_total(fixed_dof,free_dof);
kss = k_total(fixed_dof,fixed_dof);

% Building force vector
F_vector = concen_tl'; F_vector = F_vector(:);

D_vector(free_dof) =  kff\(F_vector(free_dof)-kfs*D_vector(fixed_dof));
F_vector(fixed_dof) = ksf*D_vector(free_dof) + kss*D_vector(fixed_dof);

% Output matrix
DEFL = reshape(D_vector,6,nnodes)';
REACT = reshape(F_vector,6,nnodes)' - concen_tl;

for i=1:nele
    ELE_FOR(i,:) = (reshape(k_local_stack(i,:,:),12,12))*(reshape(k_trans_stack(i,:,:),12,12))*(D_vector(mem_id(i,1:12)));
    l = Length(i); F_d = [-w(i,1)*l/2;-w(i,2)*l/2;-w(i,3)*l/2;0;-w(i,3)*l^2/12;-w(i,2)*l^2/12;-w(i,1)*l/2;-w(i,2)*l/2;-
w(i,3)*l/2;0;w(i,3)*l^2/12;w(i,2)*l^2/12];
    ELE_FOR(i,:) = ELE_FOR(i,:) + F_d(:)';
end
```

From the outputs obtained the user can then draw the bending moment and shear force diagram for the beams, Also the deflection diagram for each beam can be drawn solving the differential equation of beam. The program Beamalyzer does all that for the user and presents a graphical results as would be illustrated later in the report.

The outputs are stored in the matrix variable ***DEFL, REACT*** and ***ELE-FOR*** in the program. The ***AFLAG*** variable *return* **1** if the structure is stable else its returns **0.**

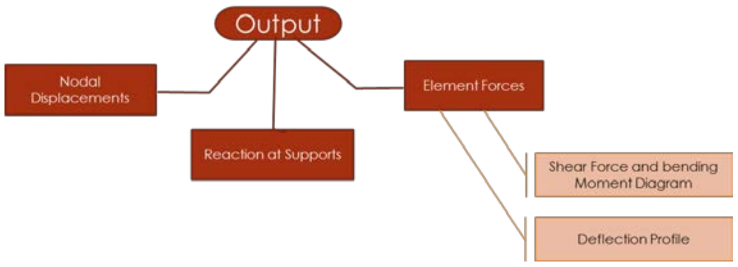


FIGURE 4 OUTPUT GENERATED BY THE PROGRAM

The subroutine **SFBMD.m** creates bending moment and shear force diagram for individual beams along all the axis of the beam. The Matlab subroutine is shown below.

SFBMD.m – Program to generate shear force & bending moment diagram about any axis

```
function SFBMD(option,forces,distloads,len)

   Xmin = -0.01*len;
   Xmax =  1.01*len;

   if(option==1)
      sf1 = forces(2); sf2 = forces(8);
      bm1 = forces(6); bm2 = forces(12);
      m = 1; w   = distloads(2);
      sf_label='Vy (KN)'; bm_label='Mz (KN)';
   end
   if(option==2)
      sf1 = forces(3); sf2 = forces(9);
      bm1 = forces(5); bm2 = forces(11);
      m = 1; w   = distloads(3);
      sf_label='Vz (KN)'; bm_label='My (KN)';
   end
   if(option==3)
      sf1 = forces(1); sf2 = forces(7);
      bm1 = forces(4); bm2 = forces(10);
      m = 0; w   = distloads(1);
      sf_label='Vx (KN)'; bm_label='Mx (KN)';
   end

   if(option ~=0)

      h = waitbar(0,'Please wait...');
      waitbar(0.2);
      syms sf_(x) bm_(x);
      % sf(x)=dsolve(diff(sf_,1)==w,sf_(0)==sf1,sf_(len)==-sf2);
      %bm(x)=dsolve(diff(bm_,1)==1*m*sf(x),bm_(0)==-bm1,bm_(len)==bm2);
      sf(x)=dsolve(diff(sf_,1)==w,sf_(0)==sf1); waitbar(0.5);
      bm(x)=dsolve(diff(bm_,1)==1*m*sf(x),bm_(0)==-bm1);waitbar(0.7);
      diff(bm_,1)==1*m*sf(x);
      bm_(0)==-bm1;
      bm_(len)==bm2;

      x = linspace(0,len);

      %% Shear Force Diagram

      Structural_Diagrams_1 = findobj('Tag','Structural_Diagrams_1');
      cla(Structural_Diagrams_1); axes(Structural_Diagrams_1);

      X = [0,x,len];  Y = [0,sf(x),0];
      fill(X,Y,'b','facealpha',0.15,'edgecolor','none'); hold on;
      plot([0,x,len],[0,sf(x),0],'b','Linewidth',2);hold off;
```

```
        refline(0,0); xlabel('x(m)'); ylabel(sf_label);
        title('Shear Force Diagram','FontSize',12);
        Ymin =  int64(min(Y)-1);    Ymax = int64(max(Y)+1);

        Structural_Diagrams_1.XLim =[Xmin Xmax];
        Structural_Diagrams_1.YLim =[Ymin Ymax];
        rotateXLabels( Structural_Diagrams_1, 90 );
        set(Structural_Diagrams_1,'FontSize',10);

        Structural_Diagrams_1.Tag = 'Structural_Diagrams_1';

        %% Bending Moment Diagram

        Structural_Diagrams_2 = findobj('Tag','Structural_Diagrams_2');
        cla(Structural_Diagrams_2); axes(Structural_Diagrams_2);

        X = [0,x,len];  Y = [0,-bm(x),0];
        fill(X,Y,'b','facealpha',0.15,'edgecolor','none'); hold on;
        plot([0,x,len],[0,-bm(x),0],'b','Linewidth',2);hold off;
        refline(0,0); xlabel('x(m)'); ylabel(bm_label);
        title('Bending Moment Diagram','FontSize',12);
        Ymin =  int64(min(Y)-1);    Ymax = int64(max(Y)+1);

        Structural_Diagrams_2.XLim = [Xmin  Xmax];
        Structural_Diagrams_2.YLim = [Ymin Ymax];
        rotateXLabels( Structural_Diagrams_2, 90 );
        set(Structural_Diagrams_2,'FontSize',10);

        Structural_Diagrams_2.Tag = 'Structural_Diagrams_2';

        waitbar(1);
        close(h)

    end

end

% Example Cantelever load with concentrated force at the end; SFBMD(1,[0,1,0,0,0,10,0,-1,0,0,0,0],[0,0,0],10)
% Example Cantilever example with distributed load; SFBMD(1,[0,10,0,0,0,50,0,0,0,0,0,0],[0,-1,0],10)
```

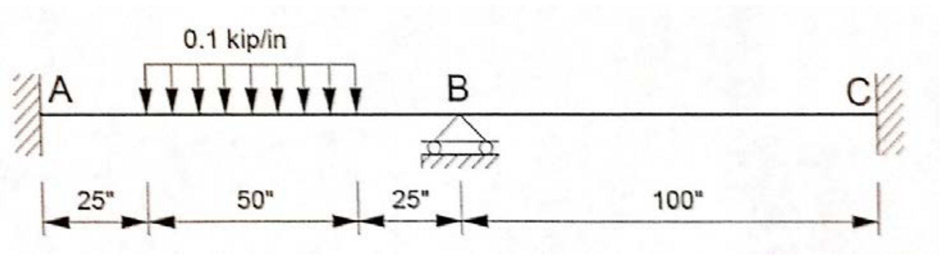
Similarly, to draw deflection diagram Matlab subroutine **DD.m** is used. The code for deflection diagram is shown below.

DD.m– Deflection Diagram for beams

```
function DD(forces,dsp,distloads,len,E,Izz,Iyy,J,A)

    Xmin = -0.01*len;
    Xmax =  1.01*len;

    Structural_Diagrams_2 = findobj('Tag','Structural_Diagrams_2');
    cla(Structural_Diagrams_2);

    Structural_Diagrams_1 = findobj('Tag','Structural_Diagrams_1');
    cla(Structural_Diagrams_1); axes (Structural_Diagrams_1);

    t = waitbar(0,'Please wait...');
    waitbar(0.2);

    %%%%% Vy Mz --> gives y coordinates
        sf1 = forces(2); sf2 = forces(8);
        bm1 = forces(6); bm2 = forces(12);
        m = 1; w   = distloads(2);
        syms sf_y(x) bm_z(x) y_(x); Dy_ = diff(y_, x);
        sfy(x)=dsolve(diff(sf_y,1)==w,sf_y(0)==sf1);waitbar(0.3);
        bmz(x)=dsolve(diff(bm_z,1)==1*m*sfy(x),bm_z(0)==-bm1);waitbar(0.4);
        Y(x) =dsolve(diff(y_,2)==bmz(x)/Izz/E,  y_(0)==dsp(2),y_(len)==dsp(8)); waitbar(0.5);

    % x = linspace(0,len);
    % plot(x,Y(x)); hold on ;

    %%%%% Vz My --> gives z coordinates
        sf1 = forces(3); sf2 = forces(9);
        bm1 = forces(5); bm2 = forces(11);
        m = 1; w   = distloads(3);

        syms sf_z(x) bm_y(x) z_(x); Dz_ = diff(z_, x);
        sfz(x)=dsolve(diff(sf_z,1)==w,sf_z(0)==sf1);waitbar(0.6);
        bmy(x)=dsolve(diff(bm_y,1)==1*m*sfz(x),bm_y(0)==-bm1);waitbar(0.7);
        Z(x) =dsolve(diff(z_,2)==bmy(x)/E/Iyy, z_(0)==dsp(3),z_(len)==dsp(9));waitbar(0.8);

    x = linspace(0,len);
    plot3(Structural_Diagrams_1,x,Y(x),Z(x),'b','Linewidth',2);waitbar(0.95);
    xlabel(Structural_Diagrams_1,'x(m)');
    ylabel(Structural_Diagrams_1,'Disp_y(m)');
    zlabel(Structural_Diagrams_1,'Disp_z(m)');
    title(Structural_Diagrams_1,'Deflection Diagram','FontSize',12)
    Ymin =  int64(min(Y(x))-1); Ymax = int64(max(Y(x))+1);
    % Structural_Diagrams_1.CameraPosition=[0,0,1];
    view(Structural_Diagrams_1,[0,0,1]);

    waitbar(1);
    close(t);

end

% Example Cantelever load with concentrated force at the end; DD([0,1,0,0,0,10,0,-1,0,0,0,0],[0,0,0,0,0,0,0,-333.33,0,0,0,-
50],[0,0,0],10,1,1,1,1,1)
% Example Cantilever example with distributed load; DD([0,10,0,0,0,50,0,0,0,0,0,0],[0,0,0,0,0,0,0,-1250,0,0,0,-166.66],[0,-1,0],10,1,1,1,1,1)
```

DEMONSTRATION OF BEAMALYZER

In this section we are going to see how Beamalyzer works i.e. how to give inputs, plot mesh and generate outputs and visualize the structure and draw bending moment, shear force and deflection diagrams. We would be using the following structure as shown below to see show how Beamalyzer works. To *run the program* **just type Beamalyzer in Matlab.**



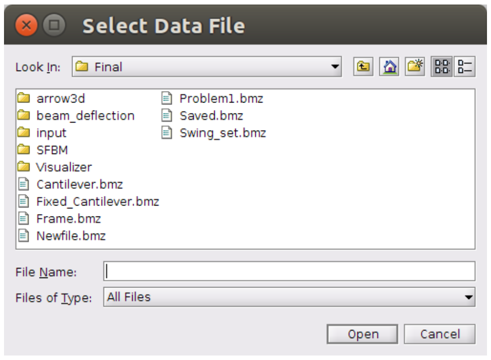
1. Input

In Beamalyzer, Input can be given through a number of ways as described below

1. ***Through file*** – Beamalyzer has an option to load input or analysis file directly into the program. A typical file for this problem would look like as shown below

```
--- Nodal geometry force and boundary conditions

nnodes
5

coord
0.000000 0.000000 0.000000
25.000000 0.000000 0.000000
75.000000 0.000000 0.000000
100.000000 0.000000 0.000000
200.000000 0.000000 0.000000

concen
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

fixity
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
NaN NaN 0.000000 0.000000 0.000000 NaN
NaN NaN 0.000000 0.000000 0.000000 NaN
NaN 0.000000 0.000000 0.000000 0.000000 NaN
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

--- Element connectivity and distributed loads

nele
4

ends
1 2
2 3
3 4
4 5

beta_ang
0.000000
0.000000
0.000000
0.000000

w
0.000000 0.000000 0.000000
0.000000 -0.100000 0.000000
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000


--- Material properties of beams

A
16.000000
16.000000
16.000000
16.000000

E
29000.000000
29000.000000
29000.000000
29000.000000

v
0.300000
0.300000
0.300000
0.300000

Izz
21.333330
21.333330
21.333330
21.333330

Iyy
21.333330
21.333330
21.333330
21.333330

J
128.000000
128.000000
128.000000
128.000000
```

The file is given to the program using the file menu option on the top.

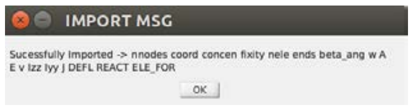


Look In: Final

arrow3d
beam_deflection
input
SFBM
Visualizer
Cantilever.bmz
Fixed_Cantilever.bmz
Frame.bmz
Newfile.bmz
Problem1.bmz
Saved.bmz
Swing_set.bmz

File Name:
Files of Type: All Files

A message is prompted about the success of import as shown below.


IMPORT MSG

Sucessfully imported -> nnodes coord concen fixity nele ends beta_ang w A E v Izz Iyy J DEFL REACT ELE_FOR

2. ***Manual Input*** – Beamalyzer has also a flexible way of defining input variable ***(Nodal geometry, Element Properties and Material & Section properties)*** directly in the program through its GUI interface. Different tabs corresponding to three different inputs has been made to directly edit the input for the structure. The user can navigate to individual tabs

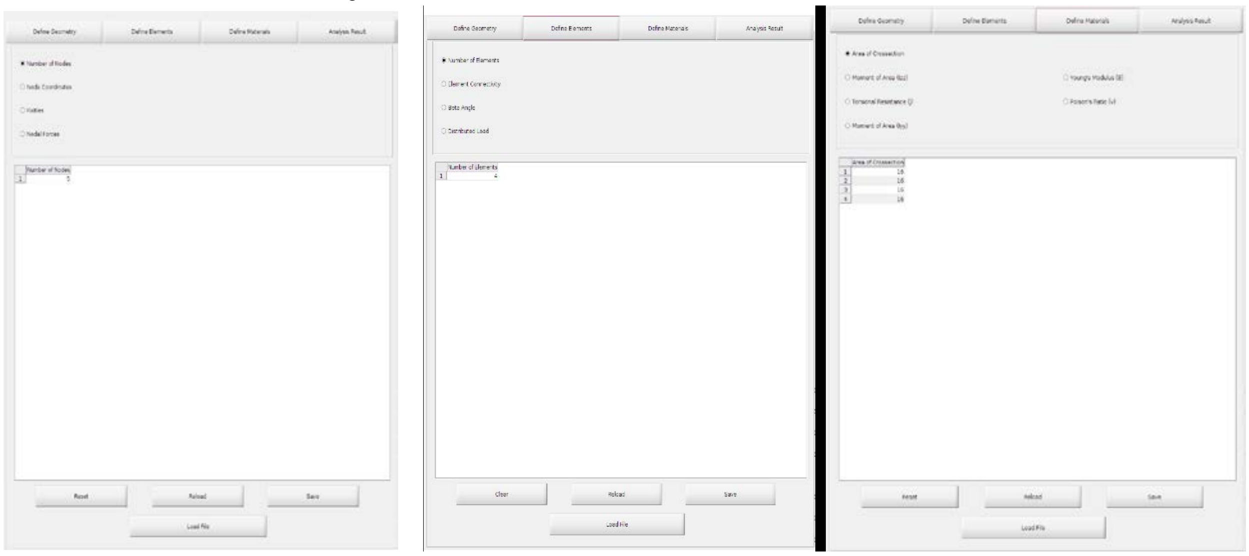


FIGURE 5 MANUAL EDITING OF INPUT THROUGH GUI INTERFACE

The clear, reload, save and load file command provides more flexibility in editing the input.

2. Visualizing the Geometric Mesh

Through Beamalyzer one can visualize the mesh and the nodal forces on the structure. To do this, the user has to just press the '**Check Geometry Mesh'** option.

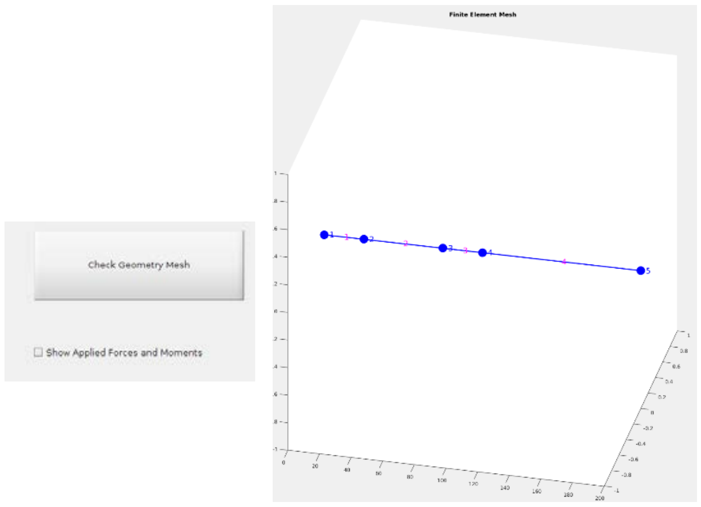
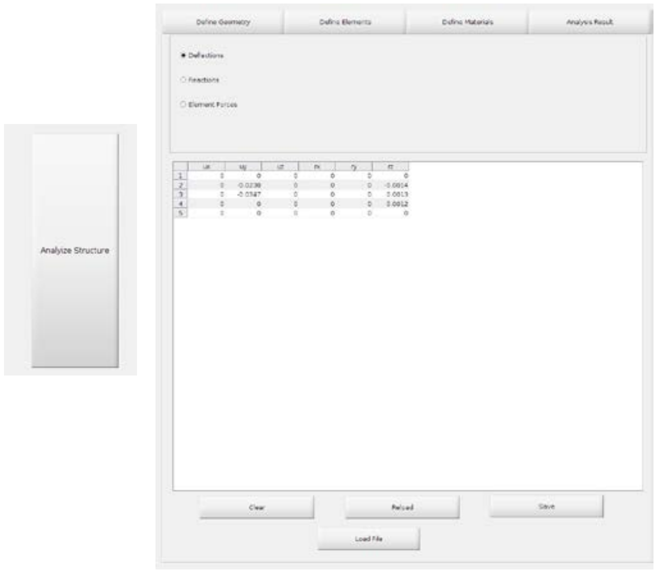


FIGURE 6 VISULIZING THE GEOMETRIC MESH OF THE STRUCTURE

3. ANALYSIS of the structure

To analyze the structure the user has to just press the analyze button located on the bottom-left of beamalyzer application window. The program takes some time to solve the equations and thus shows a wait bar to tell the user the percentage of work completed and remaining so that the user made aware about the time taken by the program.

If there are any errors i.e. the structure is unstable an error message indicating an unstable structure is thrown. The results of the analysis can be seen the last '**Analysis Result**' tab as shown below.



	Ux	Uy	Uz	Rx	Ry	Rz
1	0	0	0	0	0	0
2	0	-0.0238	0	0	0	-0.0014
3	0	-0.0347	0	0	0	0.0013
4	0	0	0	0	0	0.0012
5	0	0	0	0	0	0

FIGURE 7 PICTURE SHOWING THE ANALYSIS BUTTON AND RESULT PRESENTED IN 'ANALYSIS' TAB

4. Visualization of Deformed Structure

Beamalyzer has an option to visualize the deformed structure by pressing the '**Deformed Structure'** button. There is also an option to control the magnification of deformation if in case the deformations are too small to reflect in the deformed structure. The deformed structure for the sample structure is shown below.

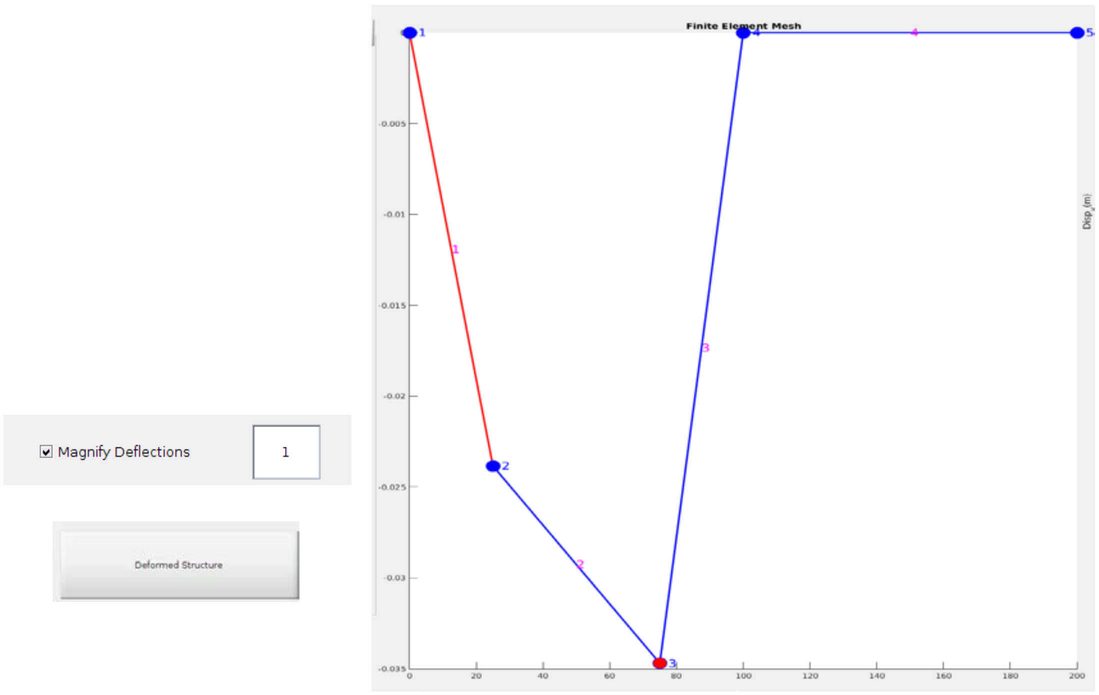
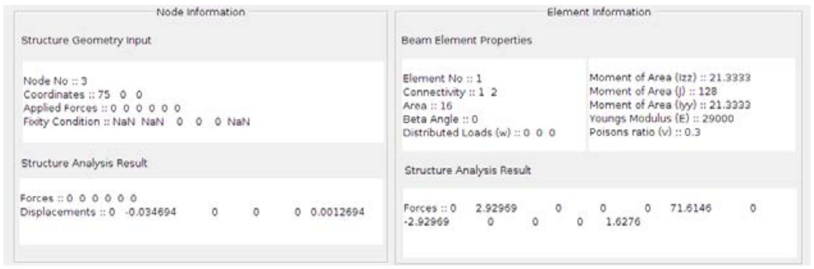


FIGURE 8 VISUALIZING DEFORMED STRUCTURE

5. Shear Force, Deflection and Bending Moment Diagram

Beamalyzer has an option to draw the shear force and bending moment and deflection diagram for any beam in 3-D space along local coordinates of that beam. The user can select any beam and node and can see the information of that beam or node. The information panel shows the information about the selected nodes and elements. An example for the node and element selected in figure 8 has been shown below.



Node Information

Structure Geometry Input

```
Node No :: 3
Coordinates :: 75   0   0
Applied Forces :: 0  0  0  0  0  0
Fixity Condition :: NaN  NaN   0   0   0  NaN
```

Structure Analysis Result

```
Forces :: 0  0  0  0  0  0
Displacements :: 0  -0.034694        0         0         0  0.0012694
```

Element Information

Beam Element Properties

```
Element No :: 1
Connectivity :: 1  2
Area :: 16
Beta Angle :: 0
Distributed Loads (w) :: 0  0  0
```

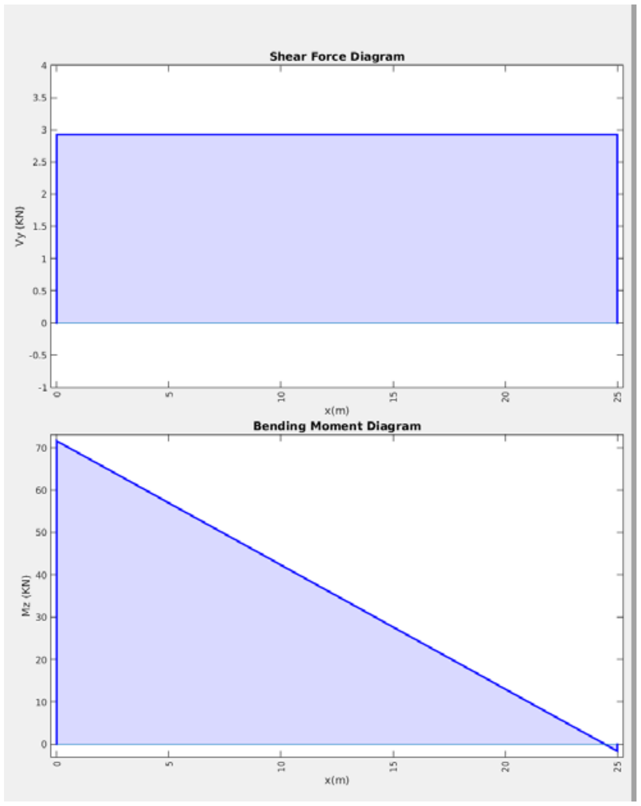
```
Moment of Area (Izz) :: 21.3333
Moment of Area (J) :: 128
Moment of Area (Iyy) :: 21.3333
Youngs Modulus (E) :: 29000
Poisons ratio (v) :: 0.3
```

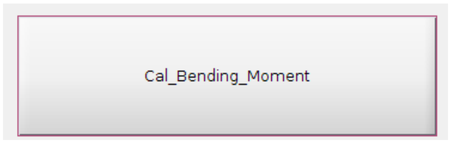
Structure Analysis Result

```
Forces :: 0     2.92969         0         0         0    71.6146         0
-2.92969        0          0         0     1.6276
```

FIGURE 9 INFORMATION PANEL FOR SELECTED NODES AND ELEMENTS

The user can then draw the shear force and bending moment diagram for the selected beam. An example of bending moment diagram for the selected beam for the given sample problem is shown below.





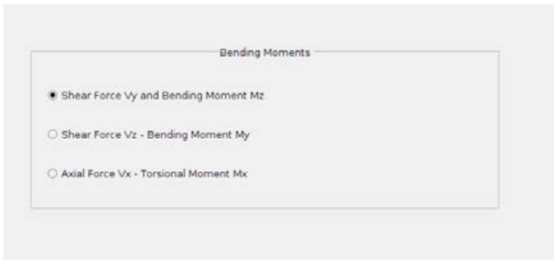
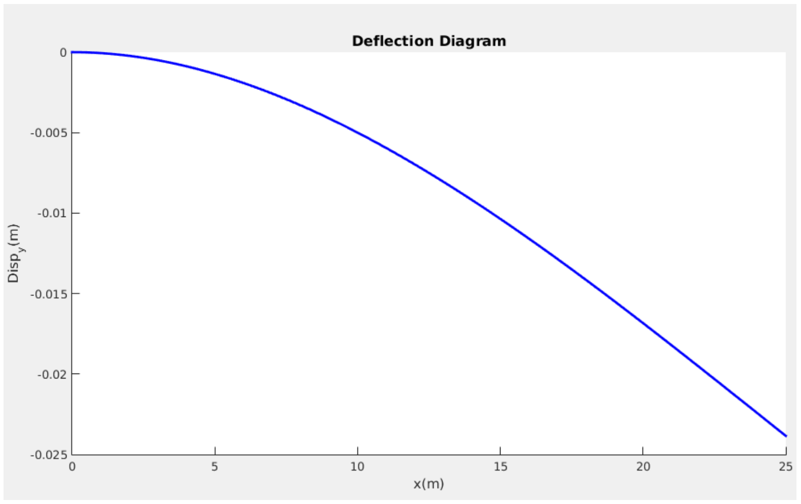

Bending Moments
Shear Force Vy and Bending Moment Mz
Shear Force Vz - Bending Moment My
Axial Force Vx - Torsional Moment Mx

FIGURE 10 DRAWING SHEAR FORCE AND BENDING MOMENT DIAGRAM FOR THE SELECTED BEAM

Also, the user has the option to draw deflection diagrams for the selected beam as shown below by just pressing the '**Deflection Diagram'** button.



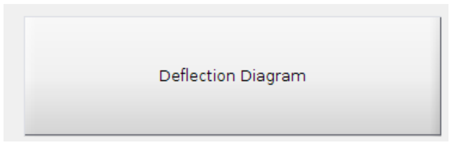


FIGURE 11 DEFLECTION DIAGRAM OF THE SELECTED BEAM

6. Save the Analysis/Result File

Beamalyzer has option to save the analysis result along with all the inputs for the structure to any folder. The analysis results just gets appended to the input file at the end. The analysis results appended to the file would look like as shown below.

```
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------- Analysis Result ----------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------------------------

DEFL
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 -0.023842 0.000000 0.000000 0.000000 -0.001414
0.000000 -0.034694 0.000000 0.000000 0.000000 0.001269
0.000000 0.000000 0.000000 0.000000 0.000000 0.001158
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

REACT
0.000000 2.929687 0.000000 0.000000 0.000000 71.614583
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 2.500000 0.000000 0.000000 0.000000 0.000000
0.000000 -0.429688 0.000000 0.000000 0.000000 14.322917

ELE_FOR
0.000000 2.929687 0.000000 0.000000 0.000000 71.614583 0.000000 -2.929687 0.000000 0.000000 0.000000 1.627604
0.000000 2.929688 0.000000 0.000000 0.000000 -1.627604 0.000000 2.070312 0.000000 0.000000 0.000000 23.111979
0.000000 -2.070313 0.000000 0.000000 0.000000 -23.111979 0.000000 2.070313 0.000000 0.000000 0.000000 -28.645833
0.000000 0.429688 0.000000 0.000000 0.000000 28.645833 0.000000 -0.429688 0.000000 0.000000 0.000000 14.322917

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

The user can load or edit this file and can be used again in beamalyzer application. This feature makes the reusability of input files again and again with minimum changes.

SWING SET VERIFICATION PROBLEM

The 3-D swing set problem was verified using Beamalyzer program. The input file for this verification problem is saved in folder **Examples** with name ***Swing_set.bmz.*** The input file for Beamalyzer for this structure is shown below.

Input File to Beamalyzer

```
-------------------------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------- Structural Input -------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------------


--- Nodal geometry force and boundary conditions


nnodes
7

coord
-1000.000000 0.000000 0.000000
1000.000000 0.000000 0.000000
0.000000 0.000000 2500.000000
0.000000 1500.000000 2500.000000
0.000000 3000.000000 2500.000000
1000.000000 3000.000000 0.000000
-1000.000000 3000.000000 0.000000

concen
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 -4.500000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000

fixity
0.000000 0.000000 0.000000 NaN NaN NaN
0.000000 0.000000 0.000000 NaN NaN NaN
NaN NaN NaN NaN NaN NaN
NaN NaN NaN NaN NaN NaN
NaN NaN NaN NaN NaN NaN
0.000000 0.000000 0.000000 NaN NaN NaN
0.000000 0.000000 0.000000 NaN NaN NaN



--- Element connectivity and distributed loads



nele
6

ends
1 3
2 3
3 4
4 5
5 6
5 7

beta_ang
0.000000
0.000000
0.000000
0.000000
```

```
0.000000
0.000000

w
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000
0.000000 0.000000 0.000000


--- Material properties of beams


A
1430.000000
1430.000000
1430.000000
1430.000000
1430.000000
1430.000000

E
200.000000
200.000000
200.000000
200.000000
200.000000
200.000000

v
0.300000
0.300000
0.300000
0.300000
0.300000
0.300000

Izz
1260000.000000
1260000.000000
1260000.000000
1260000.000000
1260000.000000
1260000.000000

Iyy
1260000.000000
1260000.000000
1260000.000000
1260000.000000
1260000.000000
1260000.000000

J
2520000.000000
2520000.000000
2520000.000000
2520000.000000
2520000.000000
2520000.000000
```

Visualization of Structural Mesh in Beamalyzer

The mesh for the given swing set problem was visualized in Beamalyzer to see that the program takes all the correct values and the structure looks the same that's intended to be. The mesh is shown in the figure below. The mesh shows the node numbering, element numbering and the nodal forces applied.

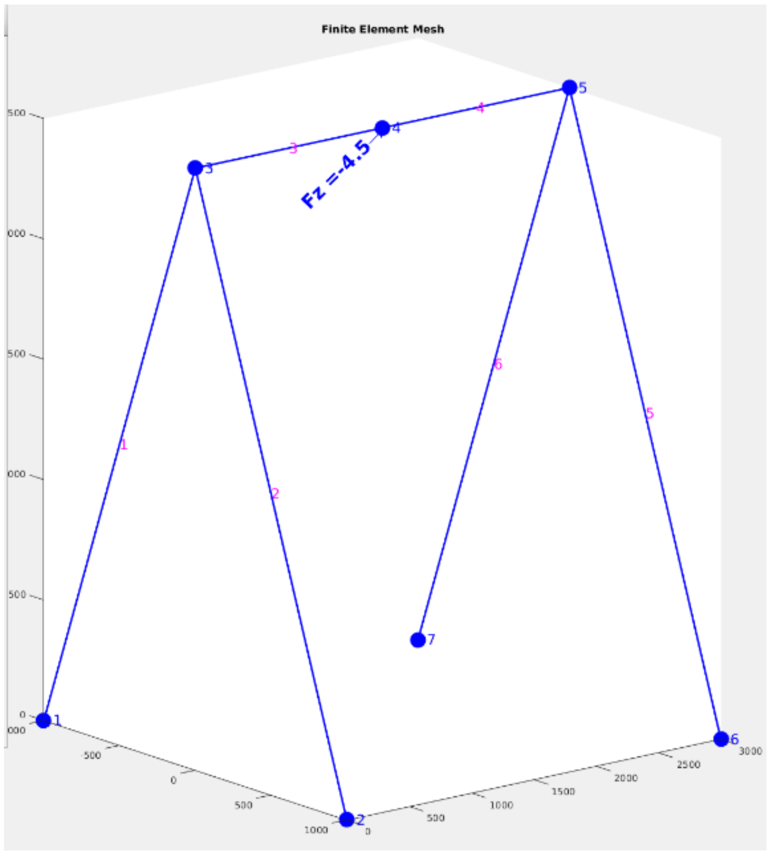
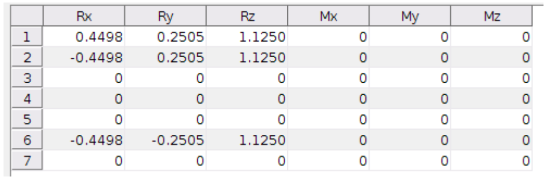


FIGURE 12 VISUALIZATION OF THE STRUCTURE MESH OF VERIFICATION PROBLEM

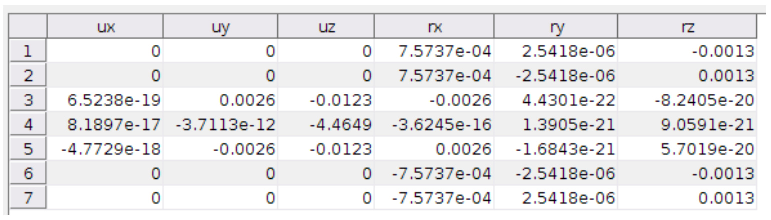
Results from the Analysis

The structure was then analyzed to get the final reactions at support, displacements at free ends and individual beam-element forces.



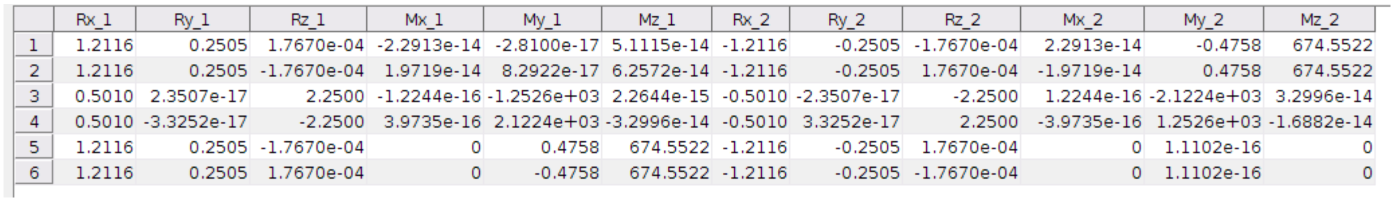
	Rx	Ry	Rz	Mx	My	Mz
1	0.4498	0.2505	1.1250	0	0	0
2	-0.4498	0.2505	1.1250	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	-0.4498	-0.2505	1.1250	0	0	0
7	0	0	0	0	0	0

FIGURE 13 REACTION AT NODES



	ux	uy	uz	rx	ry	rz
1	0	0	0	7.5737e-04	2.5418e-06	-0.0013
2	0	0	0	7.5737e-04	-2.5418e-06	0.0013
3	6.5238e-19	0.0026	-0.0123	-0.0026	4.4301e-22	-8.2405e-20
4	8.1897e-17	-3.7113e-12	-4.4649	-3.6245e-16	1.3905e-21	9.0591e-21
5	-4.7729e-18	-0.0026	-0.0123	0.0026	-1.6843e-21	5.7019e-20
6	0	0	0	-7.5737e-04	-2.5418e-06	-0.0013
7	0	0	0	-7.5737e-04	2.5418e-06	0.0013

FIGURE 14 DEFLECTION AT NODES



	Rx_1	Ry_1	Rz_1	Mx_1	My_1	Mz_1	Rx_2	Ry_2	Rz_2	Mx_2	My_2	Mz_2
1	1.2116	0.2505	1.7670e-04	-2.2913e-14	-2.8100e-17	5.1115e-14	-1.2116	-0.2505	-1.7670e-04	2.2913e-14	-0.4758	674.5522
2	1.2116	0.2505	-1.7670e-04	1.9719e-14	8.2922e-17	6.2572e-14	-1.2116	-0.2505	1.7670e-04	-1.9719e-14	0.4758	674.5522
3	0.5010	2.3507e-17	2.2500	-1.2244e-16	-1.2526e+03	2.2644e-15	-0.5010	-2.3507e-17	-2.2500	1.2244e-16	-2.1224e+03	3.2996e-14
4	0.5010	-3.3252e-17	-2.2500	3.9735e-16	2.1224e+03	-3.2996e-14	-0.5010	3.3252e-17	2.2500	-3.9735e-16	1.2526e+03	-1.6882e-14
5	1.2116	0.2505	-1.7670e-04	0	0.4758	674.5522	-1.2116	-0.2505	1.7670e-04	0	1.1102e-16	0
6	1.2116	0.2505	1.7670e-04	0	-0.4758	674.5522	-1.2116	-0.2505	-1.7670e-04	0	1.1102e-16	0

FIGURE 15 BEAM ELEMENT FORCES

Visualization of Deformed Structure

The deformed mesh is shown in the figure below

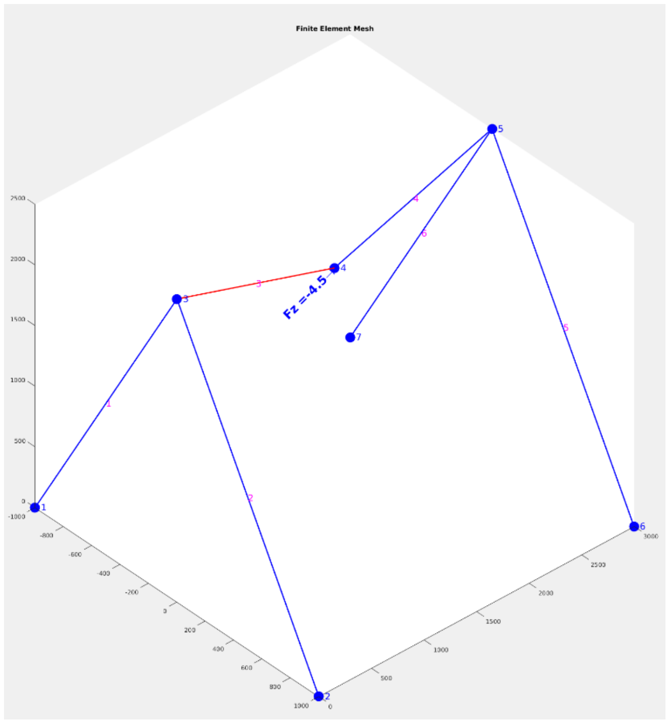


FIGURE 16 DEFORMED STRUCTURE AFTER LOAD APPLICATION

Visualizing Bending Moment, Shear force diagrams and deflection diagrams

An example of bending moment, shear force and deflection diagram of the selected beam in **Figure 16** is shown below.

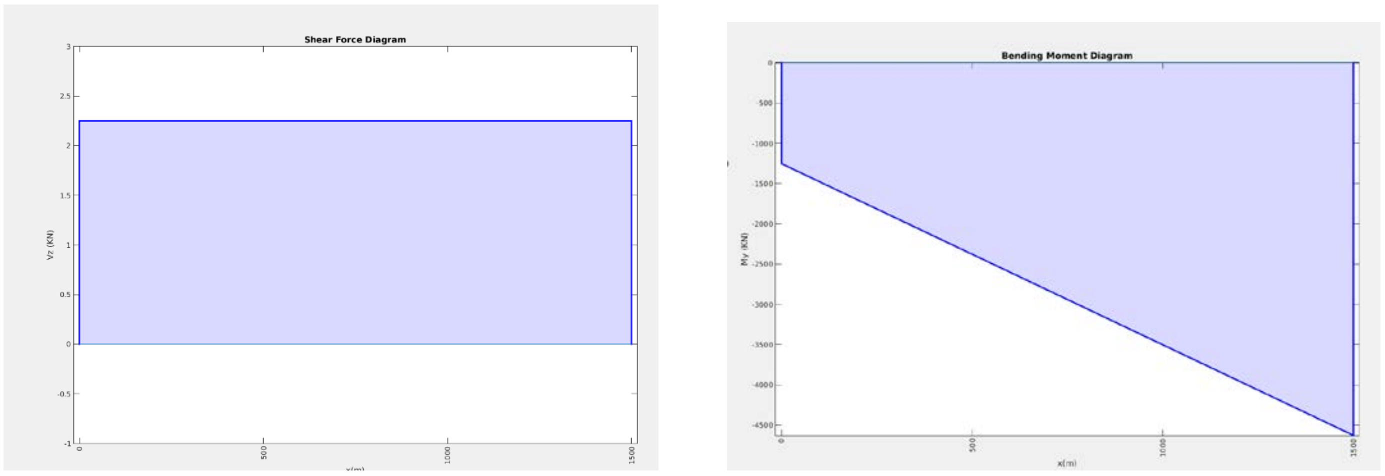


FIGURE 17 SHEAR FORCE (V_Z) AND BENDING MOMENT (M_X) DIAGRAM OF BEAM 3

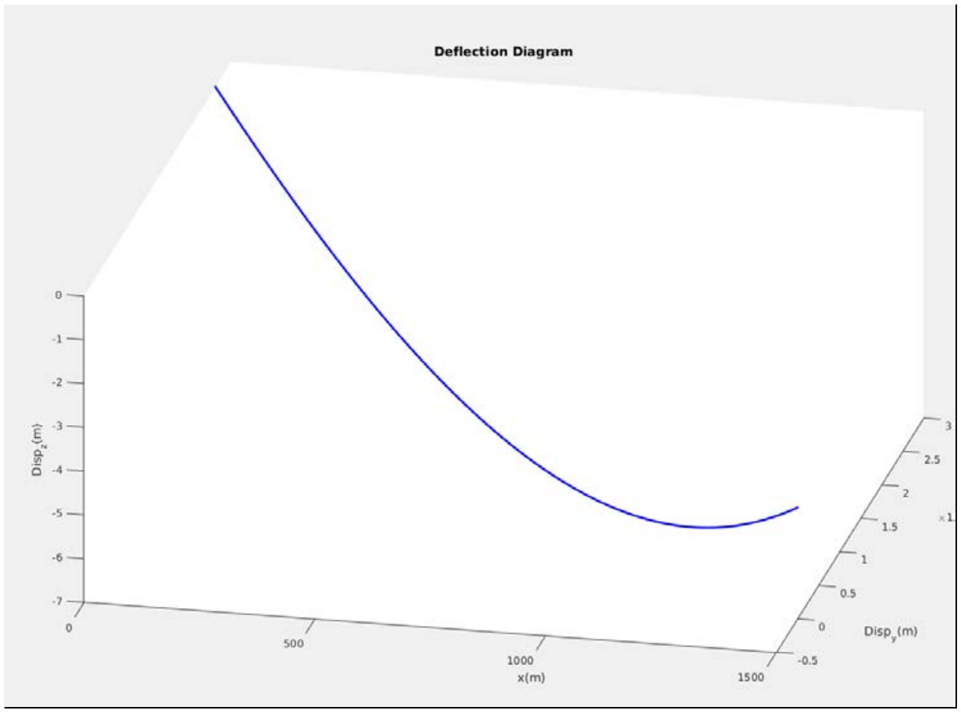


FIGURE 18 3-D DEFLECTION DIAGRAM FOR BEAM 3

Reporting Desired Results asked the Verification problem

In the model analyzed in Beamalyzer

- ***Node a*** corresponds to ***node no. 1***
- ***Node b*** corresponds to ***node no. 2***
- ***Node c*** corresponds to ***node no. 3***
- ***Node d*** corresponds to ***node no. 4***
- ***Element a-c*** corresponds to ***element no. 1***
- ***Element c-b*** corresponds to ***element no. 2 [from node b (2) to node c (3)]***
- ***Element c-d*** corresponds to ***element no. 3***

Thus,

1. *Deflection* at **node c** and **d** are

Deflection at nodes c and d						
Node No	**u**	**v**	**w**	**Θx**	**Θy**	**Θz**
c	0	0.002628	-0.01229	-0.00259	0	0
d	0	0	-4.46491	0	0	0

2. *Reactions* at **node a** and **b** are

Reactions at nodes a and b						
Node No	Fx (kN)	Fy (kN)	Fz (kN)	Mx (kN-mm)	My (kN-mm)	Mz (kN-mm)
a	0.44981	0.250522	1.125	0	0	0
b	-0.44981	0.250522	1.125	0	0	0

3. *Element Forces*

Element Forces in Local reactions for element a-c, c-b and c-d												
Element	Fx1 (kN)	Fy1 (kN)	Fz1 (kN)	Mx1 (kN-mm)	My1 (kN-mm)	Mz1 (kN-mm)	Fx2 (kN)	Fy2 (kN)	Fz2 (kN)	Mx2 (kN-mm)	My2 (kN-mm)	Mz2 (kN-mm)
a-c	1.211591	0.250522	0.000177	0	0	0	-1.211591	-0.25052	-0.00018	0	-0.475776	674.5522
c-b	1.211591	0.250522	-0.00018	0	0	0	-1.211591	-0.25052	0.000177	0	0.475776	674.5522
c-d	0.501045	0	2.25	0	-1252.61199	0	-0.501045	0	-2.25	0	-2122.38801	0

4. *Axial Forces of beams a-c, c-b and c-d*

Axial Force	
Element	Axial Force (kN)
a-c	1.211591
c-b	1.211591
c-d	0.501045