Final Project Writeup

Shayaan Siddiqui, 400247500, siddis41

A beam with a cross-section that extends from y = 0 to Ly=0.5 m and x = 0 to Lx = 1.6 m is made of 3 different materials stacked on top of each other in the y-direction:

y range	E [GPa]	ν	ρ [kg/m^3]
$0..0.3Ly$	110	0.30	3800
$0.3Ly..0.5Ly$	100	0.21	4700
$0.5Ly..Ly$	220	0.38	1900

The beam extends from z = 0 (where it is fixed) to Lz =2 m (where it is free).
Gravity affects the beam in the -y direction.

a) Using these material parameters, use Maple (ignoring shear displacement) to plot the y-displacement from z = 0 to Lz and determine the y-displacement due to curvature of the z = Lz surface of the beam.
   i. First use Newtons laws to solve and plot Shear force and Moment. Attain flexural displacement
   ii. Then use Beam equation method to confirm the displacement value found in the previous step
   iii. Include shear displacement if possible.
b) Use Flex Pde to confirm flexural displacement found in part a)
c) Use Maple to find the first 3 resonance modes
   i. Using intuition, explain which resonance mode will most likely cause failure
   ii. Use the resonance mode frequency that is least likely to cause failure and output that for the next step
d) Plot the resonance modes displacement on flex pde found from the previous step
e) For the final step, assume that a cube is cut out of the beam. The beam portrays isotropic properties and is now fixed in the x dimension on its bottom surface (x = 0) and has a tensile stress of -6 MPa. Use average modulus of elasticity and Poisson’s ratio.
   i. First use maple to show strain values
   ii. Then use Flex Pde to show strain values

A) i)

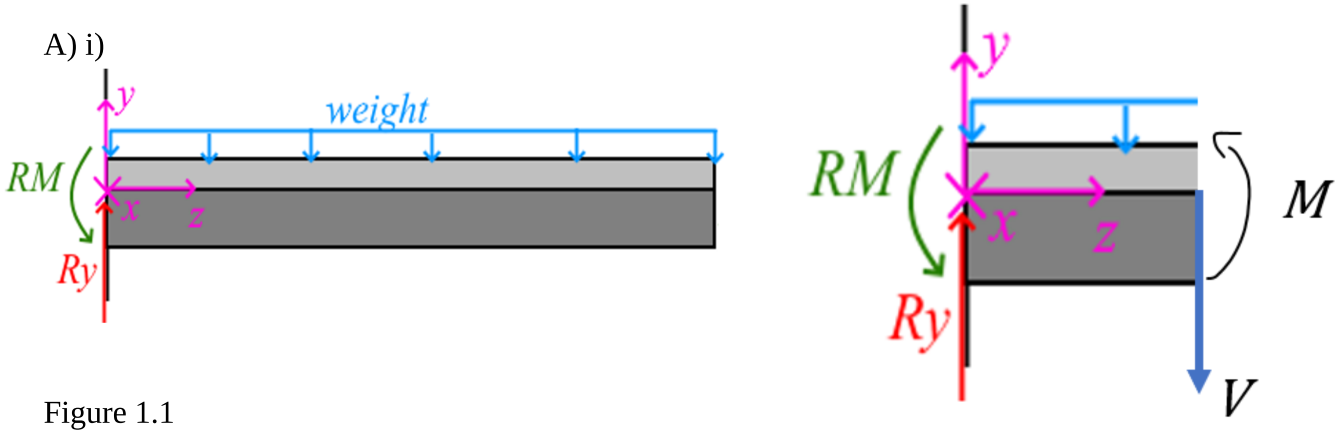


Figure 1.1

Using the free body diagram in Figure 1.1, we can set up Newton's law in Maple.

```
#statics Approach
solve([
Ry-int(p0,z=0..Lz),
RM-int(z*p0, z=0..Lz)]); assign(%):
V:=Ry-int(p0, z=0..z):
M:=-RM+int(V, z=0..z):
plot([V, M], z=0..Lz, legend=['V','M']);
```

$\{RM = 47558.88000, Ry = 47558.88000\}$

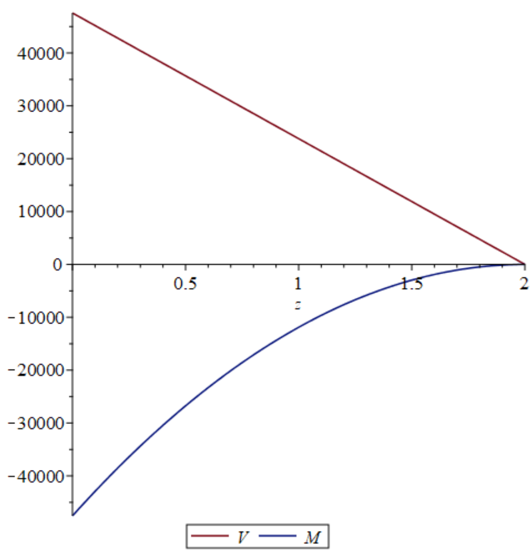


Figure1.2

We can see that the magnitude of shear force and moment decreases as is reaches the length of Lz.

To find p0, we use volume density and integrate to find linear density. Using linear density, we can multiply by gravitational constant to find Force per meter. This value is seen being used in the statics approach.

```
mu:=int(w*rho, y=0..Ly): #linear mass density
p0:=mu*9.81; #force of beam
```

Figure 1.3

To find the displacement due to curvature from the statics approach, we will need to find the flexural rigidity.

$$\bar{Y} \equiv \frac{\int_A EydA}{\int_A EdA}$$

We first use this formula to find Ybar which is the average Y distance.

$$I = \int_A (y - \bar{Y})^2 dA$$

Using Ybar, we can find the moment of inertia to find the flexural rigidity.

```
Ybar:=int(y*E*w, y=0..Ly)/int(E*w,y=0..Ly); #avg
distance
EI:=int( w*(y-Ybar)^2*E, y=0..Ly); #flexural rigidity
```

$Ybar := 0.2927914110$

$EI := 2.505890593\ 10^9$ Figure 1.4

$$M = EIc$$

$$c = \frac{\partial^2 v}{\partial z^2}$$

Then finding curvature from the flexural rigidity will allow me to integrate the curvature in respect to the axis the beam extends in. This allows me to find the flexural displacement in y.

```
c:=M/EI: #curvature
vbending:=int(int(c, z=0..z), z=0..z):  # double
integral of curvatur to find bending
vTipBending:=subs(z=Lz, vbending); #how much bending
due to own weight... bending displacement
```

$vTipBending := -0.00001897883337$

Figure 1.5

ii)
Now we can use the beam equation method to solve for the flexural displacement.

Type of boundary	$V = EI\dfrac{\partial^3 v}{\partial z^3}$	$M = EI\dfrac{\partial^2 v}{\partial z^2}$	$\theta = \dfrac{\partial v}{\partial z}$	v
Free	0	0	?	?
Clamped (Fixed)	?	?	0	0
Pivot joint (Pin)	?	0	?	0
Vertical Slider (Roller)	0	?	0	?

The initial conditions of the DE can be solved using this table. Since it is Fixed at z=0 then the function and first derivative will equation to 0. The beam is free at z=Lz therefore the second and third derivative at Lz will equate to 0.

$$EI\frac{\partial^4 v}{\partial z^4} = -p_0$$

Using this equation, we can solve for flexural displacement. In the previous steps we already solved for force per meter and the flexural rigidity.

```
#Beam Eqn Check
DE:= -(p0) = EI*(diff(v(z),z,z,z,z)):
ICS:= D(v)(0)=0, v(0)=0, D[1,1,1](v)(Lz)=0, D[1,1](v)
(Lz)=0:
dsolve([DE,ICS]):
assign(%):

displacement:= evalf(subs(z=Lz,v(z)));
```

$displacement := -0.00001897883337$ Figure 1.6

Figure 1.6 confirms that the statics and Beam equations method will both output the same answer.

iii)
To find the shear displacement we first find the area that the shear stress will affect. That would be the Z face. We would then need to solve for the avg shear modulus since the modulus of elasticity is a piecewise function. Then using stress and strain relationships we can solve for displacement due to shear force.

```
Az:=int(w, y=0..Ly); #cross sectional area
G:=E/(2*(1+nu)): Gavg:=int(w*G, y=0..Ly)/Az; #avg
shear modules
#shear stress/strain
tauzy:=-V/Az: gzy:=tauzy/Gavg:
#shear displacement
vshear:=int(gzy,z=0..z):
vTipshear:=subs(z=Lz, vshear);
```

$vTipshear := -9.775826074\ 10^{-7}$ Figure 1.7

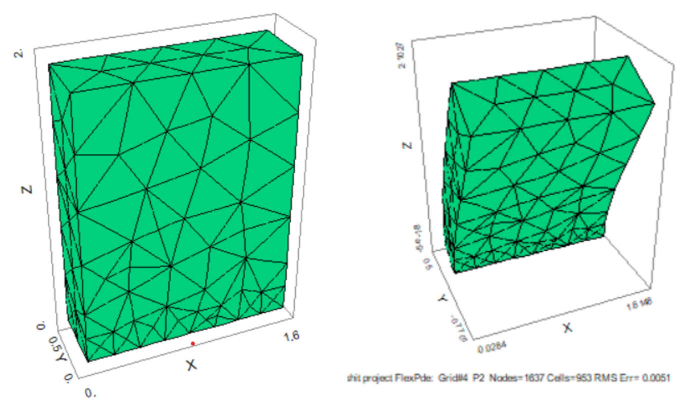
B)

```
Lx = 1.6
Ly = 0.5
Lz = 2
```

```
E = if(y<.3*Ly) then 120e9 else if(y<.5*Ly) then 100e9 else 220e9
nu = if(y<.3*Ly) then .30 else if(y<.5*Ly) then .21 else .38
rho = if(y<.3*Ly) then 3800 else if(y<.5*Ly) then 4700 else 1900
G = E/(2*(1+nu))
EQUATIONS      { PDE's, one for each variable }
u: dx(sx) + dy(sxy) + dz(sxz) = 0
v: dx(sxy) + dy(sy) + dz(syz) -9.81*rho= 0
w: dx(sxz) + dy(syz) + dz(sz) = 0
```

Figure 2.1

By initializing the stress and strain relationships into Flex Pde we can then input the material parameters. Then we can input force into the y direction equation since this is where gravity is affecting the beam.



val(v, 0,0,Lz)= -1.804670e-5

Figure 2.2

Flex Pde outputs the same answer as maple with a 5 % error which shows that Flex Pde and maple agree on the displacement caused by flexure.

C)

$$\hat{v}(x) = c_1 \cos\beta\, x + c_2 \sin\beta\, x + c_3 \cosh\beta\, x + c_4 \sinh\beta\, x$$

Using this general equation for displacement due to resonance we can then start solving for constant values. Using the boundaries chart in Part A ii) we can use the same initial conditions to solve for all constants.

```
v:=c1*cos(beta*z)+c2*sin(beta*z)+c3*cosh(beta*z)+c4*
sinh(beta*z);
vp:=diff(v, z): vpp:=diff(vp,z): vppp:=diff(vpp,z):
c2:
simplify(subs(z=0, v)); c1:=solve(%, c1);
simplify(subs(z=0,vp)); c4:=solve(%, c4);
simplify(subs(z=Lz, vpp)); c2:=solve(%, c2);
simplify(subs(z=Lz, vppp));
GenFn:=simplify(%/(c3));
```

$$GenFn := -\frac{\beta^3\left(2\cos(2\beta)\cosh(2\beta) + \cosh(2\beta)^2 - \sinh(2\beta)^2 + 1\right)}{\sin(2\beta) + \sinh(2\beta)}$$

Figure 3.1

The generated function plotted will give us a visual of where all the roots will be located.

```
plot(GenFn, beta=0..12,GF=-50..50);
```

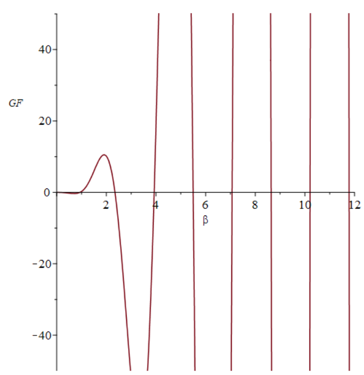
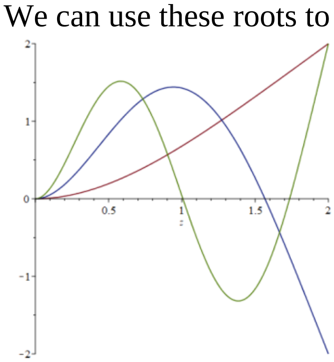


Figure 3.2

We can see that the first 3 roots will have the following ranges.

```
beta1:=fsolve(GenFn, beta=0..1.5);
beta2:=fsolve(GenFn=0, beta=2..3);
beta3:=fsolve(GenFn=0, beta=3.5..4);
```

We can use these roots to plot the resonance displacements.



```
v1:=subs(beta=beta1, v/c3):
v2:=subs(beta=beta2, v/c3):
v3:=subs(beta=beta3, v/c3):
plot([v1,v2,v3],z=0..Lz);
```

Figure 3.3

The resonance mode that is most likely to cause failure is the last resonance mode found due to the fact that

$$\omega = \sqrt{\frac{\beta^4 EI}{\mu}} = \beta^2 \sqrt{\frac{EI}{\mu}}$$

This means that the angular frequency is greater when the roots are bigger. This means that the structure will displace more radians per second. Although all resonance modes cause the same max and min displacement but the frequency being greater makes it more likely for failure to occur. That means the frequency least likely to cause failure is the smallest value.

ii)

```
omega:=beta^2*sqrt(EI/mu):
omega1:=subs(beta=beta1, omega);
```

Figure 3.4

$\omega1 := 893.7282923$

This value of frequency is quite high due to the fact that the flexural rigidity and linear density is large.

D)

Using the stages and refining the value for omega, we find that it is very close to the value found in maple.

```
omega= 893.222222+0.000001*stage
```

Figure 4.1

The error in omega values is minimal.

The first resonance mode displacement is shown to look like Figure 4.2

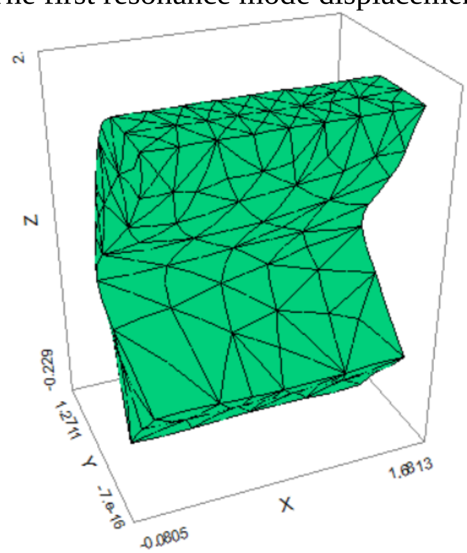


Figure 4.2

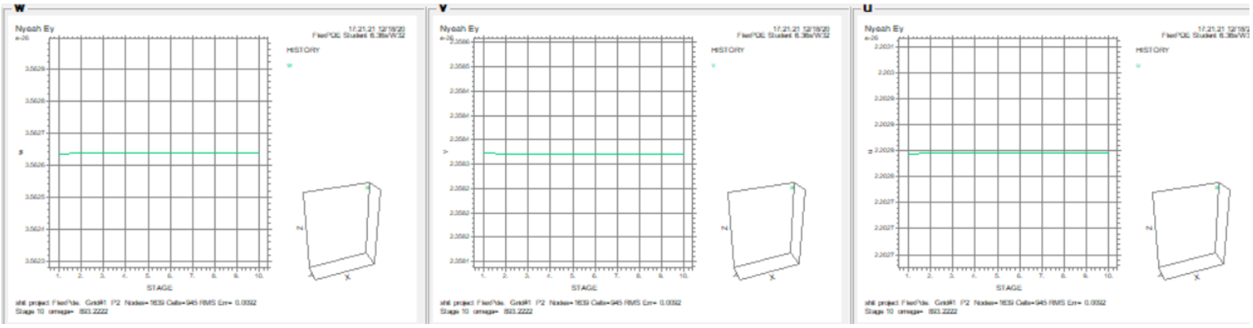


Figure 4.3

All displacement graphs match omega values therefore the beam is in resonance.

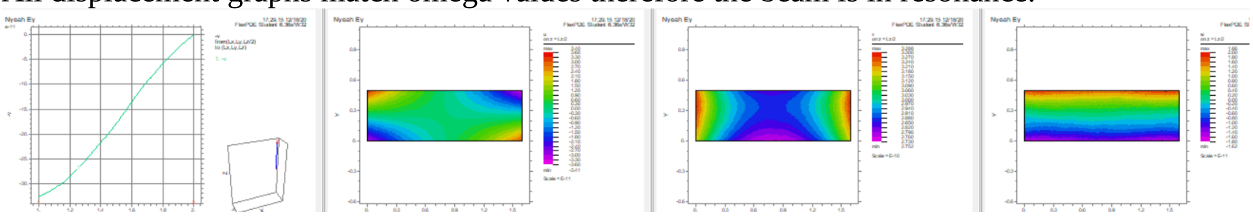


Figure 4.4

As we can see that Flex Pde agrees with Maple and displays the same resonance displacement for the first resonance mode. The contour plots show how the displacement varies across the beam. The beams displacement is oscillating as seen on the contour plots.

E)

i)

The compliance matrix of an isotropic matrix is given as such

$$\begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{bmatrix} = \frac{1}{E} \begin{bmatrix} 1 & -\nu & -\nu & 0 & 0 & 0 \\ -\nu & 1 & -\nu & 0 & 0 & 0 \\ -\nu & -\nu & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2(1+\nu) & 0 & 0 \\ 0 & 0 & 0 & 0 & 2(1+\nu) & 0 \\ 0 & 0 & 0 & 0 & 0 & 2(1+\nu) \end{bmatrix} \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{bmatrix}$$

The stiffness is given as the next matrix which is the inverse of the compliance matrix.

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{bmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}-\nu & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}-\nu & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}-\nu \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{bmatrix}$$

For this part, I have chosen to set up the compliance matrix as this is directly dot product with the stress values. To find the avg Poisson ratio and modulus we can integrate each value over the length y, and divide by this length.

```
avgnu:=(int(nu, y=0..Ly))/Ly;
avgE:=(int(E, y=0..Ly))/Ly;
```

$avgnu := 0.3220000000$

$avgE := 1.630000000\ 10^{11}$ Figure 5.1

Now we can set up the compliance matrix.

```
Smatrix:=(1/avgE)*Matrix([
[1,-avgnu,-avgnu,0,0,0],
[-avgnu,1,-avgnu,0,0,0],
[-avgnu,-avgnu,1,0,0,0],
[0,0,0,2*(1+avgnu),0,0],
[0,0,0,0,2*(1+avgnu),0],
[0,0,0,0,0,2*(1+avgnu)]]):
```

Figure 5.2

We can now initialize the stress and strain relationship with the compliance matrix

```
stress:=<-60e5,0,0,0,0,0>:
strain:=<epxx,epyy,epzz,epyz,epxz,epxy>:
HookesLawEqns:=strain=Smatrix.stress;
```

$$\{epxx = -0.00003680981595,\ epxy = 0.,\ epxz = 0.,\ epyy = 0.00001185276074,\ epyz = 0.,\ epzz = 0.00001185276074\}$$

Figure 5.3

This means that the displacement in y will be

$dy := 5.926380370\ 10^{-6}$ Figure 5.4

Which can be found using strain relationships.

ii) We can now confirm on Flex Pde. First we output all values of the stiffness matrix in maple so we can input into Flex Pde.

$$C := [[2.34820411707582\ 10^{11}, 1.11522378421595\ 10^{11}, 1.11522378421595\ 10^{11}, 0., 0., -0.], [1.11522378421595\ 10^{11}, 2.34820411707582\ 10^{11}, 1.11522378421595\ 10^{11}, 0., 0., -0.], [1.11522378421595\ 10^{11}, 1.11522378421595\ 10^{11}, 2.34820411707582\ 10^{11}, 0., 0., -0.], [0., 0., 0., 6.16490166429936\ 10^{10}, 0., -0.], [0., 0., 0., 0., 6.16490166429936\ 10^{10}, -0.], [0., 0., 0., 0., 0., 6.16490166429936\ 10^{10}]]$$

Figure 5.5

C11 =2.34820411707582*10^11
C22 = 2.34820411707582*10^11
C33 = 2.34820411707582*10^11

C12 = 1.11522378421595*10^11
C13 = 1.11522378421595*10^11
C23 = C13

C14=0
C15=0
C16=0
C24=0
C25=0
C26=0
C34=0
C35=0
C36=0

C44=6.16490166429936*10^10
C55=6.16490166429936*10^10
C66=6.16490166429936*10^10
C45=0
C46=0
C56=0

Figure 5.6

We can now check if the strain values match with maple.

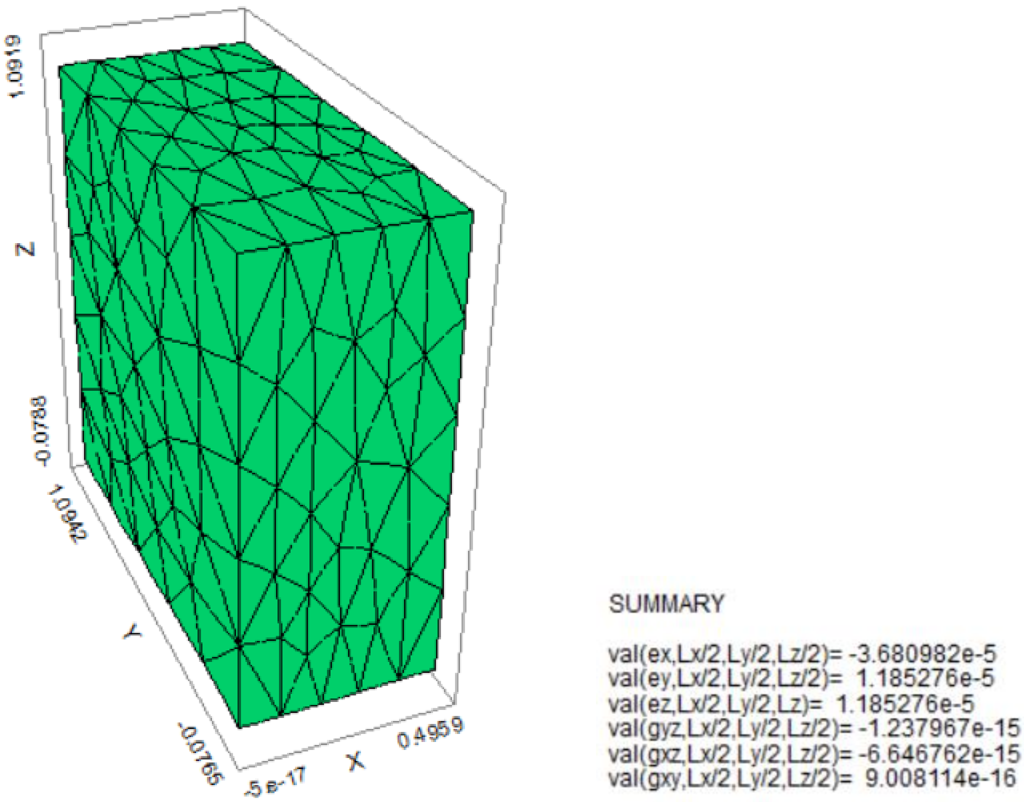


Figure 5.7

Figure 5.7 matches exactly with maples output therfore the we can confirm that with the applied load on the top face, these will be the strain values.

All maple code used will be in Appendix A.1 and all Flex Pde will be in Appendix A.2.

Appendix A.1

```
restart: Ly:=.5: Lx:=1.6: Lz:=2:
E:=piecewise(y<.3*Ly, 110e9, y<.5*Ly, 100e9, y>.5*Ly, 220e9):
nu:=piecewise(y<.3*Ly, .30, y<.5*Ly, .21, y>.5*Ly, .38):
rho:=piecewise(y<.3*Ly, 3800, y<.5*Ly, 4700, y>.5*Ly, 1900):

w:=Lx:
Ybar:=int(y*E*w, y=0..Ly)/int(E*w,y=0..Ly); #avg distance
EI:=int( w*(y-Ybar)^2*E, y=0..Ly); #flexural rigidity

mu:=int(w*rho, y=0..Ly): #linear mass density
p0:=mu*9.81; #force of beam

#Beam Eqn Check
DE:= -(p0) = EI*(diff(v(z),z,z,z,z)):
ICS:= D(v)(0)=0, v(0)=0, D[1,1,1](v)(Lz)=0, D[1,1](v)(Lz)=0:
dsolve([DE,ICS]):
assign(%):

displacement:= evalf(subs(z=Lz,v(z)));

#statics Approach
solve([
Ry-int(p0,z=0..Lz),
RM-int(z*p0, z=0..Lz)]); assign(%):
V:=Ry-int(p0, z=0..z):
M:=-RM+int(V, z=0..z):

c:=M/EI: #curvature
vbending:=int(int(c, z=0..z), z=0..z):  # double integral of curvatur to find bending
vTipBending:=subs(z=Lz, vbending); #how much bending due to own weight...
bending displacement

Az:=int(w, y=0..Ly); #cross sectional area
G:=E/(2*(1+nu)): Gavg:=int(w*G, y=0..Ly)/Az; #avg shear modules
#shear stress/strain
tauzy:=-V/Az: gzy:=tauzy/Gavg:
#shear displacement
```

```
vshear:=int(gzy,z=0..z):
vTipshear:=subs(z=Lz, vshear);
plot([V, M], z=0..Lz, legend=['V','M']);

v:=c1*cos(beta*z)+c2*sin(beta*z)+c3*cosh(beta*z)+c4*sinh(beta*z);
vp:=diff(v, z): vpp:=diff(vp,z): vppp:=diff(vpp,z):
c2:
simplify(subs(z=0, v)); c1:=solve(%, c1);
simplify(subs(z=0,vp)); c4:=solve(%, c4);
simplify(subs(z=Lz, vpp)); c2:=solve(%, c2);
simplify(subs(z=Lz, vppp));
GenFn:=simplify(%/(c3));
plot(GenFn, beta=0..12,GF=-50..50);
beta1:=fsolve(GenFn, beta=0..1.5);
beta2:=fsolve(GenFn=0, beta=2..3);
beta3:=fsolve(GenFn=0, beta=3.5..4);
v1:=subs(beta=beta1, v/c3):
v2:=subs(beta=beta2, v/c3):
v3:=subs(beta=beta3, v/c3):
plot([v1,v2,v3],z=0..Lz);
omega:=beta^2*sqrt(EI/mu):
omega1:=subs(beta=beta1, omega);
syy:= evalf(p0/(Lx*Lz));

avgnu:=(int(nu, y=0..Ly))/Ly;
avgE:=(int(E, y=0..Ly))/Ly;
with(LinearAlgebra):
Smatrix:=(1/avgE)*Matrix([
[1,-avgnu,-avgnu,0,0,0],
[-avgnu,1,-avgnu,0,0,0],
[-avgnu,-avgnu,1,0,0,0],
[0,0,0,2*(1+avgnu),0,0],
[0,0,0,0,2*(1+avgnu),0],
[0,0,0,0,0,2*(1+avgnu)]]):

stress:=<-60e5,0,0,0,0,0>:
strain:=<epxx,epyy,epzz,epyz,epxz,epxy>:
HookesLawEqns:=strain=Smatrix.stress;
solve([seq(lhs(HookesLawEqns)[n]=rhs(HookesLawEqns)[n],n=1..6)]);
assign(%):
```

```
dy:=evalf(epyy*Ly);
C:=MatrixInverse(Smatrix);
```

Appendix A.2

```
{TITLE 'Nyeah Ey'     { the problem identification }
COORDINATES cartesian3  { coordinate system, 1D,2D,3D, etc }
VARIABLES       { system variables }
u
v
w
SELECT        { method controls }
!spectral_colors
!ngrid=15
DEFINITIONS    { parameter definitions }
mag = .3*globalmax(magnitude(x,y,z))/globalmax(magnitude(u,v,w))
Lx = 1.6
Ly = 0.5
Lz = 2



E = if(y<.3*Ly) then 120e9 else if(y<.5*Ly) then 100e9 else 220e9
nu = if(y<.3*Ly) then .30 else if(y<.5*Ly) then .21 else .38
rho = if(y<.3*Ly) then 3800 else if(y<.5*Ly) then 4700 else 1900
G = E/(2*(1+nu))

ex = dx(u)
ey = dy(v)
ez = dz(w)

gyz = dz(v) + dy(w)
gxz = dz(u) + dx(w)
gxy = dy(u) + dx(v)

C11 = E/((1+nu)*(1-2*nu))*(1-nu)
C12 = E/((1+nu)*(1-2*nu))*nu
C13 =C12
C21 =C12
```

```
C22 = C11
C23 = C12
C31 = C12
C32 = C12
C33 = C11

sx = C11*ex + C12*ey+C13*ez
sy = C21*ex + C22*ey+C23*ez
sz = C31*ex + C32*ey+C33*ez

syz = G*gyz
sxz = G*gxz
sxy = G*gxy

! INITIAL VALUES
EQUATIONS        { PDE's, one for each variable }
u: dx(sx) + dy(sxy) + dz(sxz) = 0
v: dx(sxy) + dy(sy) + dz(syz) -9.81*rho= 0
w: dx(sxz) + dy(syz) + dz(sz) = 0

EXTRUSION
surface 'bottom' z = 0
surface 'top' z = Lz

BOUNDARIES       { The domain definition }
surface 'bottom'
load(u) = 0
value(v) = 0
value(w) = 0
surface 'top'



  REGION 1       { For each material region }
    START (0,0) line to (Lx,0) to (Lx,Ly) to (0,Ly) to close

! TIME 0 TO 1    { if time dependent }
MONITORS         { show progress }
PLOTS            { save result displays }
          grid(x+u*mag, y+v*mag,z+w*mag)
```

```
SUMMARY
report val(v, 0,0,Lz)

END}
{TITLE 'Nyeah Ey'
COORDINATES cartesian3  { coordinate system, 1D,2D,3D, etc }
VARIABLES        { system variables }
u
v
w
SELECT         { method controls }
!spectral_colors
!ngrid=15
stages = 10

DEFINITIONS    { parameter definitions }
mag = .3*globalmax(magnitude(x,y,z))/globalmax(magnitude(u,v,w))

Lx = 1.6
Ly = 0.5
Lz = 2



E = if(y<.3*Ly) then 120e9 else if(y<.5*Ly) then 100e9 else 220e9
nu = if(y<.3*Ly) then .30 else if(y<.5*Ly) then .21 else .38
rho = if(y<.3*Ly) then 3800 else if(y<.5*Ly) then 4700 else 1900

omega= 893.222222+0.000001*stage
G = E/(2*(1+nu))

ex = dx(u)
ey = dy(v)
ez = dz(w)

gyz = dz(v) + dy(w)
gxz = dz(u) + dx(w)
```

```
gxy = dy(u) + dx(v)

C11 = E/((1+nu)*(1-2*nu))*(1-nu)
C12 = E/((1+nu)*(1-2*nu))*nu
C13 =C12
C21 =C12
C22 = C11
C23 = C12
C31 = C12
C32 = C12
C33 = C11

sx = C11*ex + C12*ey+C13*ez
sy = C21*ex + C22*ey+C23*ez
sz = C31*ex + C32*ey+C33*ez

syz = G*gyz
sxz = G*gxz
sxy = G*gxy

phi = atan2(y,x)
xp = x+u
yp = y+v
thetatest = atan2(yp,xp)-phi
theta = if(thetatest<-pi) then thetatest+2*pi else if(thetatest > +pi) then thetatest-
2*pi else thetatest

! INITIAL VALUES
EQUATIONS       { PDE's, one for each variable }
u: dx(sx) + dy(sxy) + dz(sxz) = -rho*omega^2*u
v: dx(sxy) + dy(sy) + dz(syz) =  -rho*omega^2*v
w: dx(sxz) + dy(syz) + dz(sz) =  -rho*omega^2*w

EXTRUSION
surface 'bottom' z = 0
surface 'top' z = Lz

BOUNDARIES       { The domain definition }
surface 'bottom'
value(u) = 0
```

```
value(v) = 0
load(w) = 0
surface 'top'
value(u) = 0
value(v) = 0
value(w) = 0

!load(w) = 735294.1176*y+73529.41176


  REGION 1       { For each material region }
    START (0,0) line to (Lx,0)
line to (Lx,Ly)
load(v) = 4  line to (0,Ly)
load(v) = 0    line to close
          !start(Lx,0) arc(center=0,0) angle=360

! TIME 0 TO 1    { if time dependent }
MONITORS         { show progress }
PLOTS            { save result displays }
          grid(x+u*mag, y+v*mag,z+w*mag)
!
history(w) at (Lx,Ly, Lz) report(omega)
!
history(v) at (Lx,Ly, Lz) report(omega)
!
history(u) at (Lx,Ly, Lz) report(omega)
elevation(-v) from(Lx,Ly,Lz/2) to (Lx,Ly,Lz)
!
contour(u) on z = Lz/2 painted
!
contour(v) on z = Lz/2 painted
!
contour(w) on z = Lz/2 painted

END}

TITLE
'Nyeah Ey'
SELECT
```

```
errlim=1e-4
ngrid=5
spectral_colors
COORDINATES cartesian3 { coordinate system, 1D,2D,3D, etc }
VARIABLES { system variables }
u
v
w
DEFINITIONS { parameter definitions }
mag = .3*(globalmax(magnitude(x,y,z)))/globalmax(magnitude(u,v,w))
Lx=1
Ly=1
Lz =1
ex = dx(u)
ey = dy(v)
ez = dz(w)
gyz = dz(v)+dy(w)
gxz = dz(u)+dx(w)
gxy = dy(u)+dx(v)
C11 =2.34820411707582*10^11
C22 = 2.34820411707582*10^11
C33 = 2.34820411707582*10^11

C12 = 1.11522378421595*10^11
C13 = 1.11522378421595*10^11
C23 = C13

C14=0
C15=0
C16=0
C24=0
C25=0
C26=0
C34=0
C35=0
C36=0


C44=6.16490166429936*10^10
C55=6.16490166429936*10^10
```

```
C66=6.16490166429936*10^10
C45=0
C46=0
C56=0

C21 = C12
C31 = C13
C32 = C23
C41=C14
C42=C24
C43=C34
C51=C15
C52=C25
C53=C35
C61=C16
C62=C26
C63=C36
C54=C45
C64=C46
C65=C56
sx =C11*ex+C12*ey+C13*ez+C14*gyz+C15*gxz+C16*gxy
sy =C21*ex+C22*ey+C23*ez+C24*gyz+C25*gxz+C26*gxy
sz =C31*ex+C32*ey+C33*ez+C34*gyz+C35*gxz+C36*gxy
syz= C41*ex+C42*ey+C43*ez+C44*gyz+C45*gxz+C46*gxy
sxz= C51*ex+C52*ey+C53*ez+C54*gyz+C55*gxz+C56*gxy
sxy= C61*ex+C62*ey+C63*ez+C64*gyz+C65*gxz+C66*gxy
phi= atan2(y,x)
xp=x+u
yp=y+v
thetatest =atan2(yp, xp)-phi
theta=if(thetatest<-pi) then thetatest+2*pi else if(thetatest > pi) then thetatest-2*pi
else
thetatest
! INITIAL VALUES
EQUATIONS { PDE's, one for each variable }
u:dx(sx) +dy(sxy)+dz(sxz)= 0
v:dx(sxy)+dy(sy)+dz(syz)= 0
w:dx(sxz)+dy(syz)+dz(sz) = 0
! CONSTRAINTS { Integral constraints }
EXTRUSION
```

```
surface 'bottom' z=0
surface 'top' z = Lz
BOUNDARIES { The domain definition }
surface 'bottom'
surface 'top'
REGION 1 { For each material region }
START(0,0)
line to (Lx, 0)
load(u) = -60e5
line to (Lx, Ly)
load(u) = 0
line to (0, Ly)
value(u) = 0
!value(v) = 0
!value(w) = 0
line to close
! TIME 0 TO 1 { if time dependent }
MONITORS { show progress }
PLOTS { save result displays }
grid(x+u*mag, y+v*mag, z+w*mag)
SUMMARY
report val(ex,Lx/2,Ly/2,Lz/2)
report val(ey,Lx/2,Ly/2,Lz/2)
report val(ez,Lx/2,Ly/2,Lz)
report val(gyz,Lx/2,Ly/2,Lz/2)
report val(gxz,Lx/2,Ly/2,Lz/2)
report val(gxy,Lx/2,Ly/2,Lz/2)
END
```