In[48]:= **Start[]**

Out[48]=

Mathematica Beam Analysis Package(MBAP)

Version:Alpha

Written by: Ji Yiling

Test Use Only,Do Not Share to Others

----Data Input Module---

Step1 | Step2 | Step3

----General Analysis Module----

Linear Static | Linear Buckling | Real Eigenvalue

Complex Eigenvalue | Direct Frequency Response | Linear Transient

----Rotor Dynamic Module----

Undamped Rotor Plot | Rotor DFR Plot | Rotor Disp. Plot3D

Rotor Stability

Recommended Units	
Length	mm
Mass	t
Time	Sec
Force	N->t.mm/s^2
Stress&E	MPa->N/mm^2
Stiffness	N/mm
Damping	N.Sec/mm

Element Data Input:

aa->Area,dd->Material Density,ee->Ela.M

Iyy/Izz->Area Moment of Inertia,Ksy,Ksz->Shear Stiffness Factor

ν->Poisson's ratio

600	0	0
700	0	0
5000	$\frac{7}{1\,000\,000\,000}$	210 000
3 000 000	3 000 000	
1	1	
0.3		

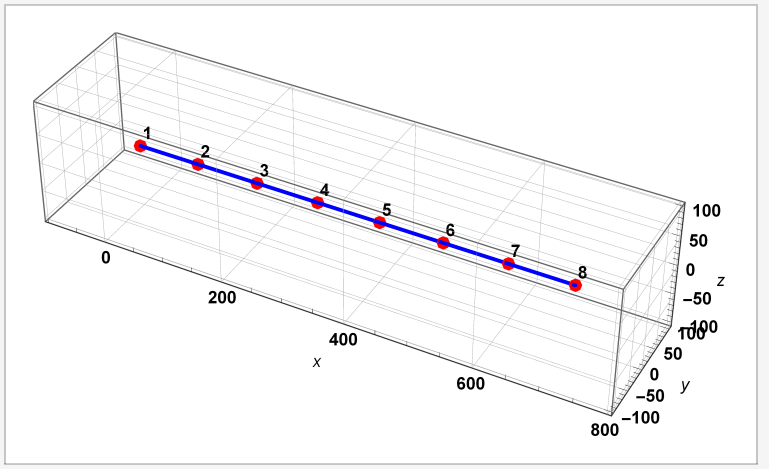
Add New Point

Element Data:

▼ Element Data

No.	x1	y1	z1	x2	y2	z2	Area	Density	Elastic.M	Iyy	Izz	Ksy	Ksz	ν
1	0	0	0	100	0	0	5000	7 / 1 000 000 0-00	210 000	3 000 000	3 000 000	1	1	0.3
2	100	0	0	200	0	0	5000	7 / 1 000 000 0-00	210 000	3 000 000	3 000 000	1	1	0.3
3	200	0	0	300	0	0	5000	7 / 1 000 000 0-00	210 000	3 000 000	3 000 000	1	1	0.3
4	300	0	0	400	0	0	12 000	7 / 1 000 000 0-00	210 000	8 000 000	8 000 000	1	1	0.3
5	400	0	0	500	0	0	12 000	7 / 1 000 000 0-00	210 000	8 000 000	8 000 000	1	1	0.3
6	500	0	0	600	0	0	6500	7 / 1 000 000 0-00	210 000	3 000 000	3 000 000	1	1	0.3
7	600	0	0	700	0	0	5000	7 / 1 000 000 0-00	210 000	3 000 000	3 000 000	1	1	0.3

Plot



Element Data Modified:

Input the Modified Element Number First:

1

0	0	0
100	0	0
5000	$\frac{7}{1\,000\,000\,000}$	210 000
3 000 000	3 000 000	
1	1	
0.3		

Element Data Delete:

Input the Deleted Element Number First:

ClickToDelete

----------Next Step----------

Apply DOF To All(0->Free,1->Constrained,Default=0):

1	0	0
1	0	0

Click to Apply

Input Node Number:

8

Select Node

Selected Node Coordinate is:

{700, 0, 0}

Apply DOF To Selected Node(0->Free,1->Constrained,Default=0):

1	0	0
1	0	0

Apply Force To Selected Node(Default=0):

0	0	0
0	0	0

Apply Unbalance Mass to Selected Node(RotorDynamic Only):

Mass(m)	0.002
Distance(r)	0.2
Phase(rad)	0.5

Stress Recovery Data:

Natural Coordinate(from -1 to 1)

```
{-1, 0, 1}
```

Stress Recovery Local Coordinate(Y&Z)

```
{{-5, -5}, {5, 5}, {5, -5}, {-5, 5}}
```

Add Elastic Spring To Selected Node:

Input the Spring Coordinate Related to the Selected Node:

0	1	0

Input Spring Constant K:

```
10 000
```

Input Spring Damping Cofficient B:

```
100
```

Add Spring

Spring Modified:

```
2
```

Spring Position:

{{700, 0, 0}, {700, 0, 1}}

K:

```
10 000
```

B:

```
100
```

Add Dynamic Elastic Spring To Selected Node:

Input the Spring Coordinate Related to the Selected Node:

0	1	0

Input Dynamic Spring Constant K(eg.{{cpm1,k1},{cpm2,k2}}):

```
{{1, 10 000}, {10 000, 10 000}}
```

Input Dynamic Spring Damping Cofficient B(eg.{{cpm1,b1},{cpm2,b2}}):

```
{{1, 100}, {10, 300}}
```

Add Spring

Dynamic Spring Modified:

4

Spring Position:

{{700, 0, 0}, {700, 1, 0}}

K:

```
{{1, 10 000}, {10 000, 10 000}}
```

B:

```
{{1, 100}, {10, 100}}
```

Add Concentrated Mass:

2

0.00001	1	1	1
0	0	0	

Add Con.Mass

{}

Con.Mass Delete:

1

Delete

DFR Input

Start Frequency	114
End Frequency	116
Increasement	1

Rotor DFR Plot Node Number

{4, 8}

Rpm for 3D Rotor Disp.Plot(Input After RDFR Analysis)

{6900, 8200}

Rotor Operation Speed Range(cpm)

{6500, 8000}

Bearing Stiffness Range(Rotor Dynamic Only)

{8000, 12 000}

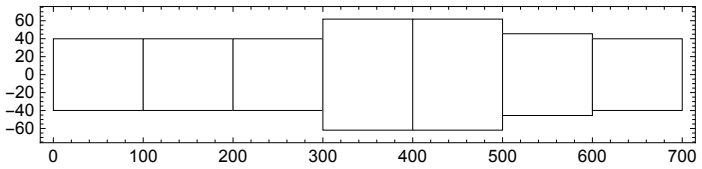
Transient Analysis Parameter:

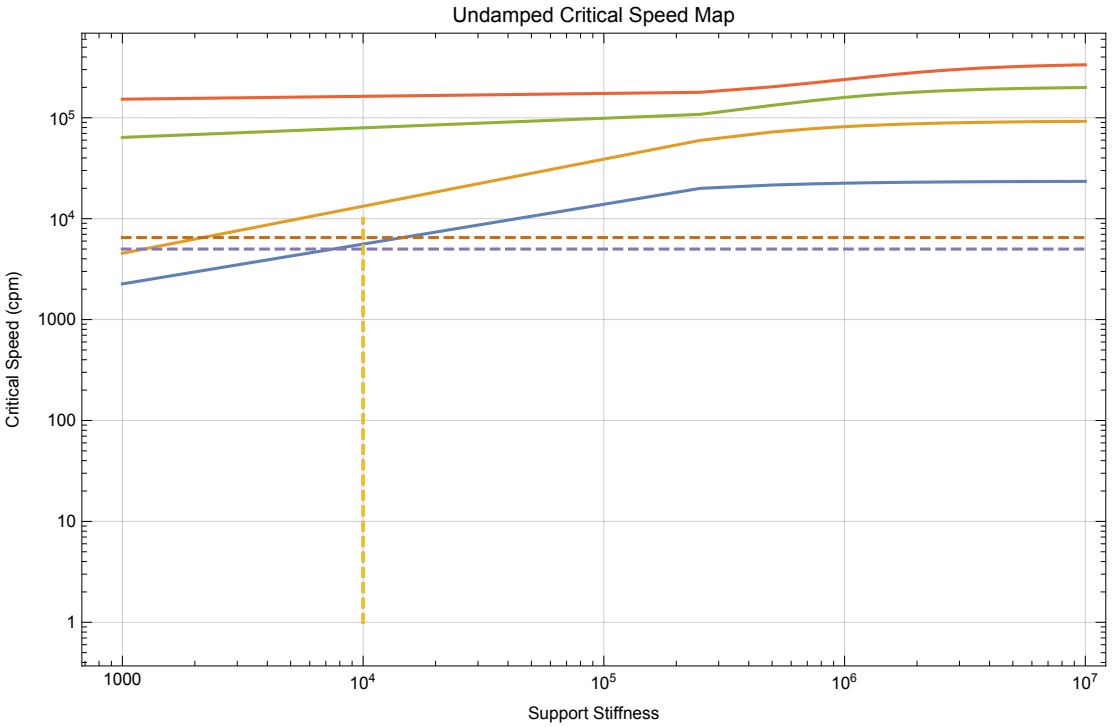
Time increment	0
Number of time steps	0

Final Result is:

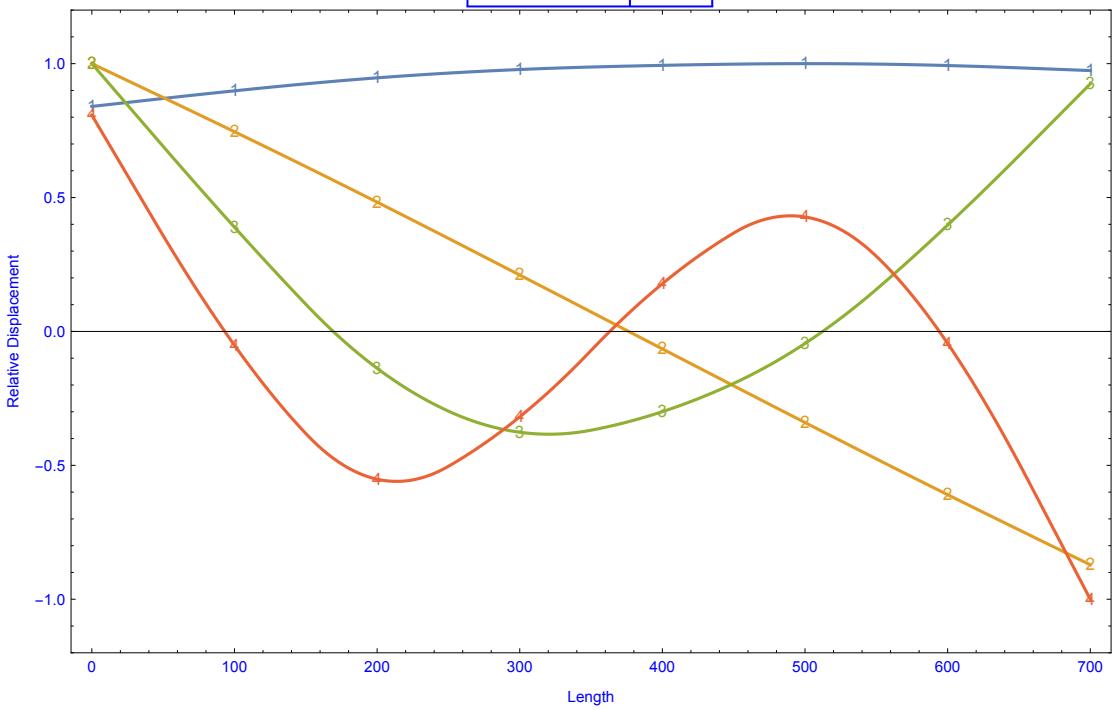
Rotor Undamped Speed Map&Mode Shape

Rotor Geometry:

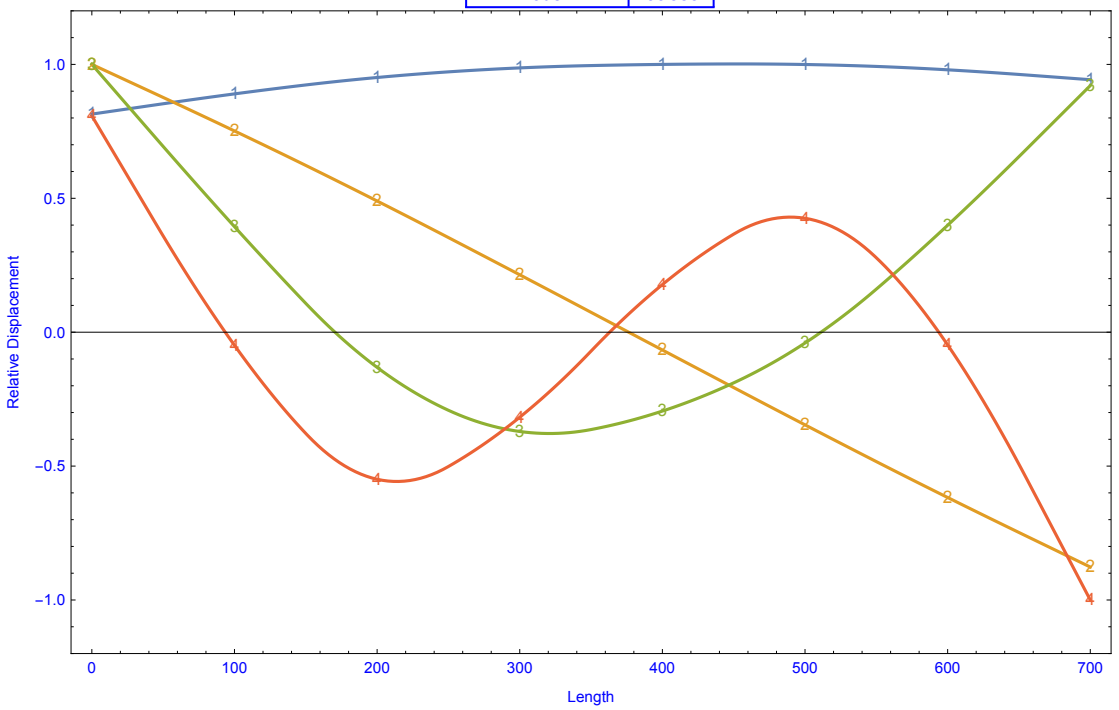




Mode Shape Plot	
Bearing Stiffness:	$8. \times 10^3$
Critical Speed(cpm)	
Mode1:	6211.73
Mode2:	12771.5
Mode3:	65577.3
Mode4:	153475.

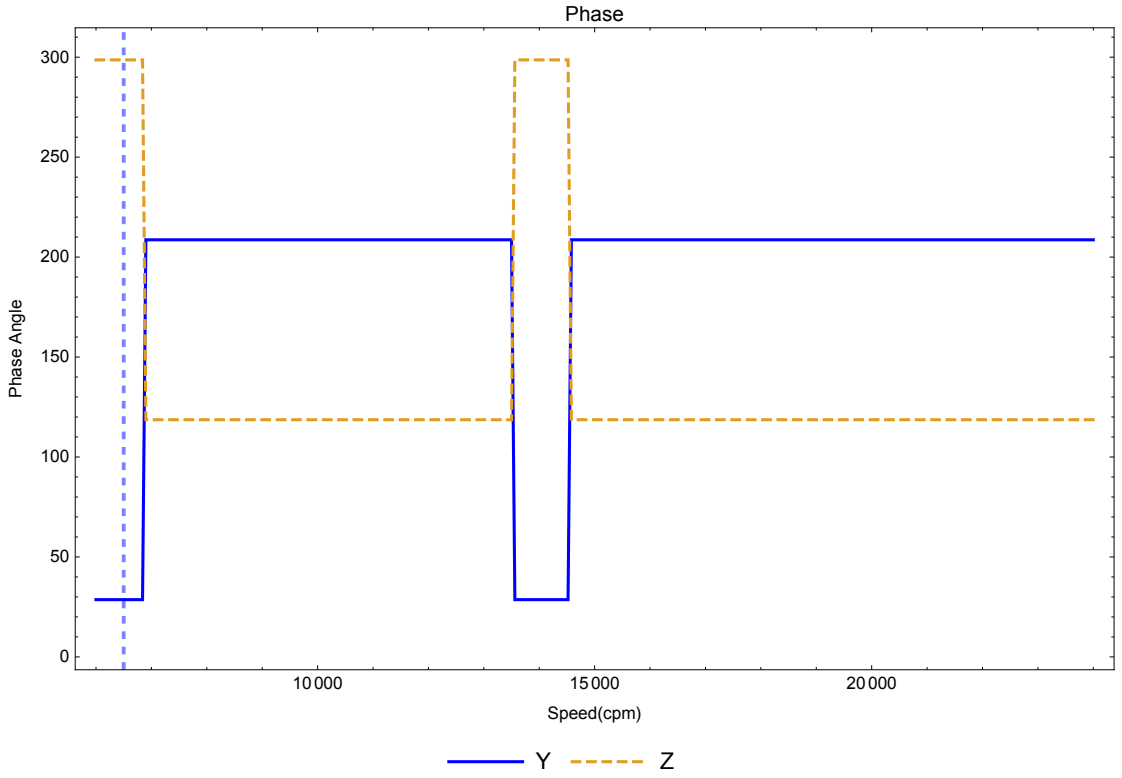


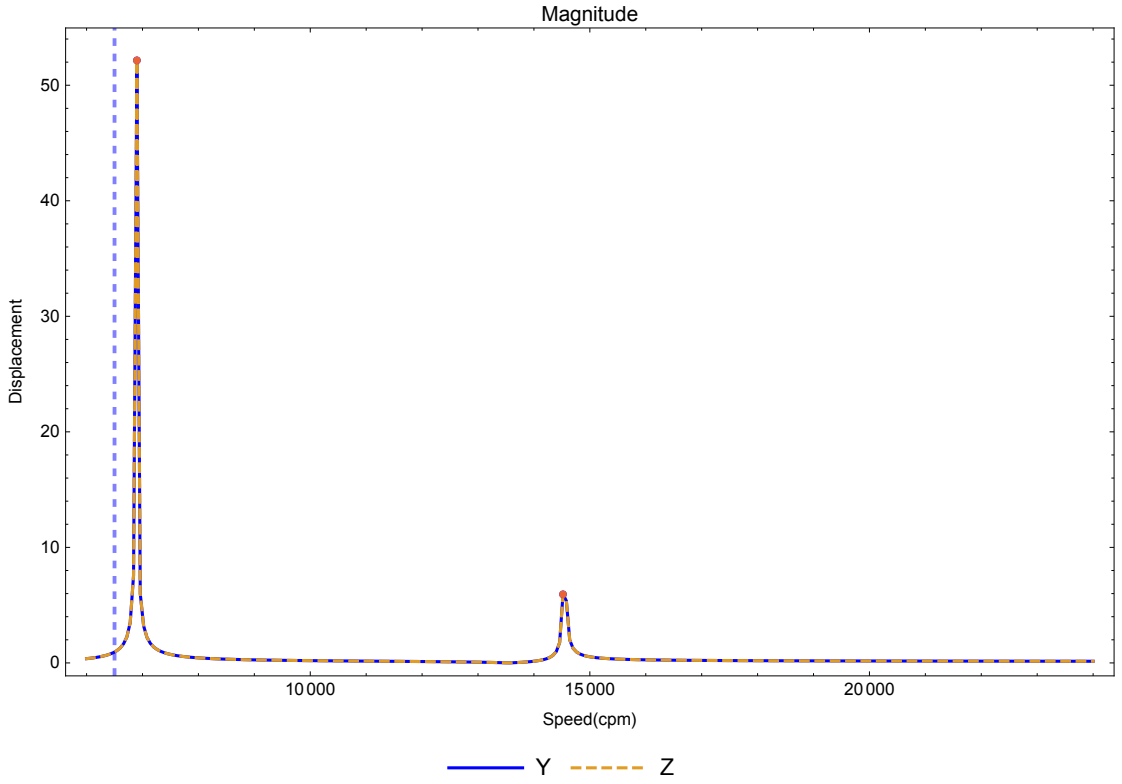
Mode Shape Plot	
Bearing Stiffness:	1.2×10^4
Critical Speed(cpm)	
Mode1:	7492.71
Mode2:	15592.6
Mode3:	66528.
Mode4:	153883.



Rotor Direct Frequency Response

Node Number is:	4



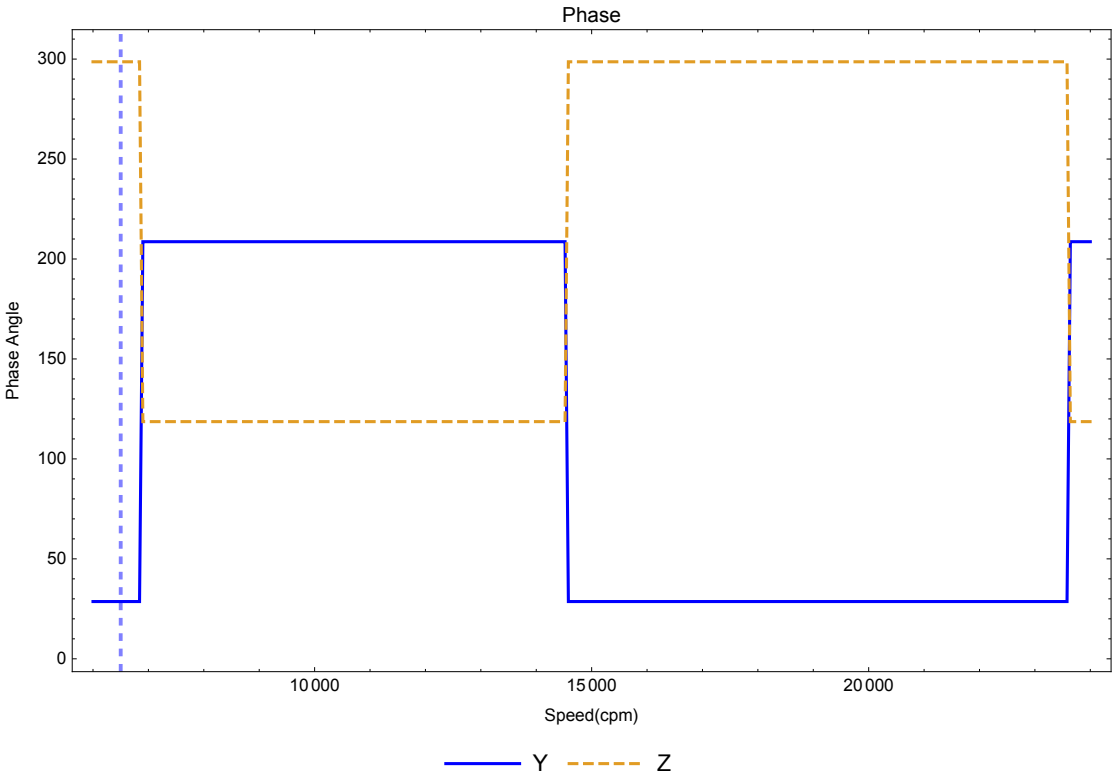


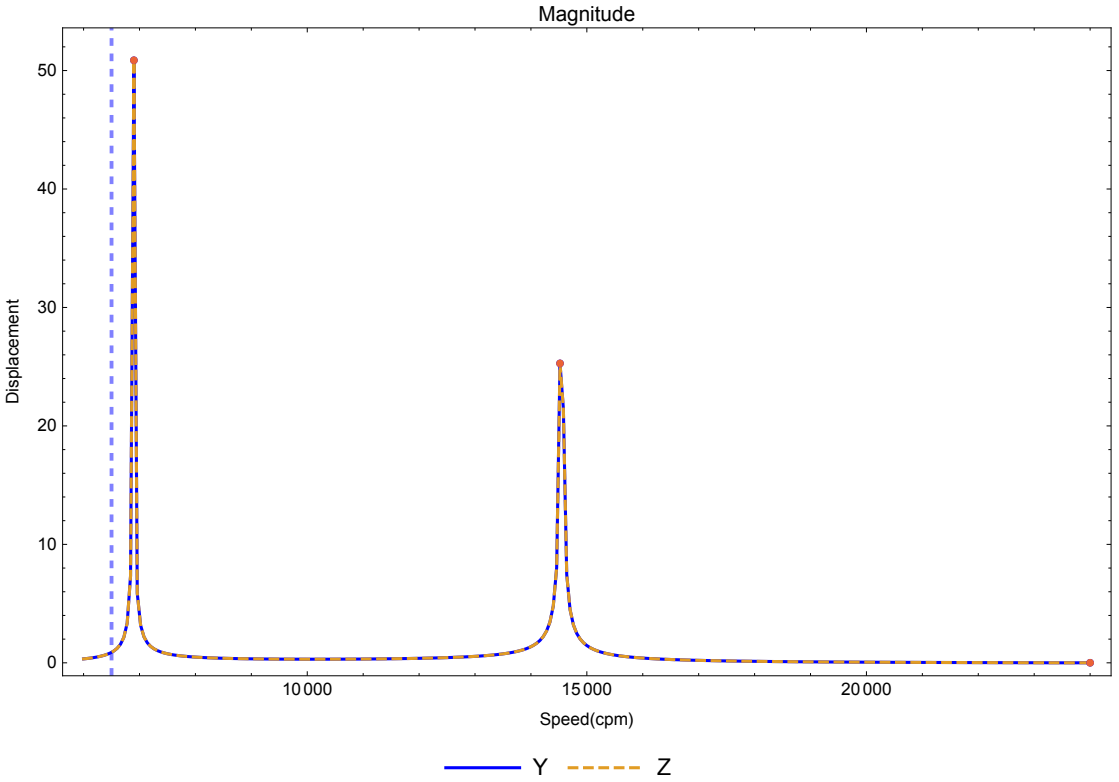
Y Maximize			Z Maximize		
cpm	Max	Amp.F	cpm	Max	Amp.F
6900	52.1536	ComplexInfinity	6900	52.1536	ComplexInfinity
14520	5.93765	1.83333	14520	5.93765	1.83333

cpm	Y Disp.	Z Disp.
5000	-0.169297	-0.169297
6500	0.945521	0.945521

```
Null
```

Node Number is:	8



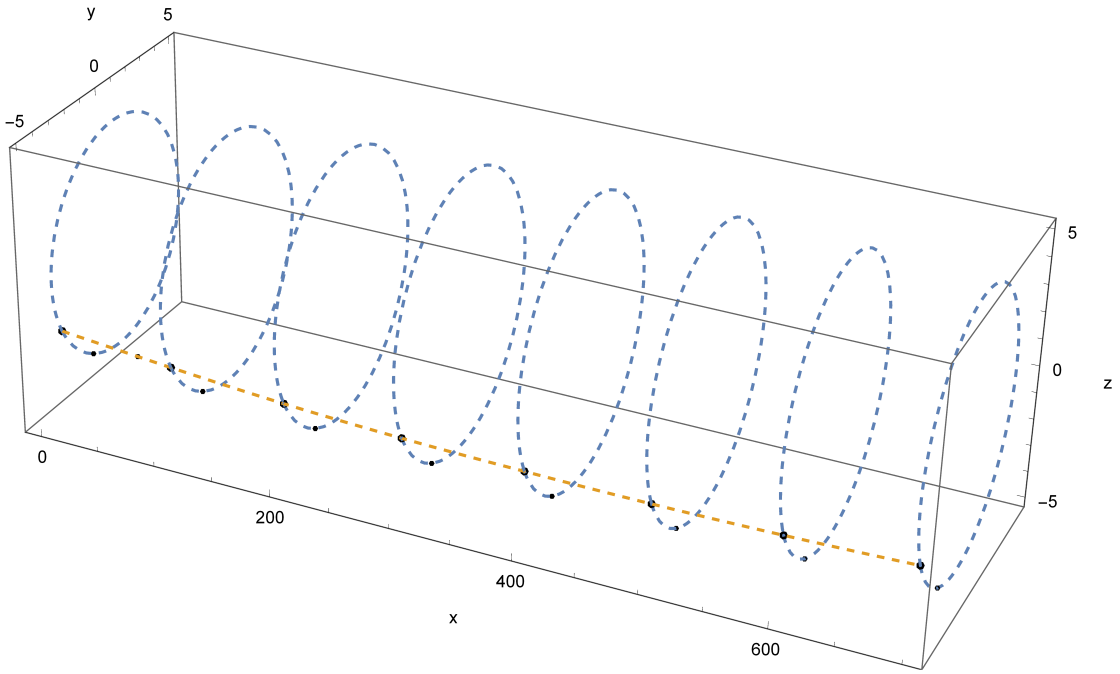


Y Maximize			Z Maximize		
cpm	Max	Amp.F	cpm	Max	Amp.F
6900	50.869	0.898438	6900	50.869	0.898438
14 520	25.2793	80.6667	14 520	25.2793	80.6667
24 000	0.00367049	25.	24 000	0.00367049	25.

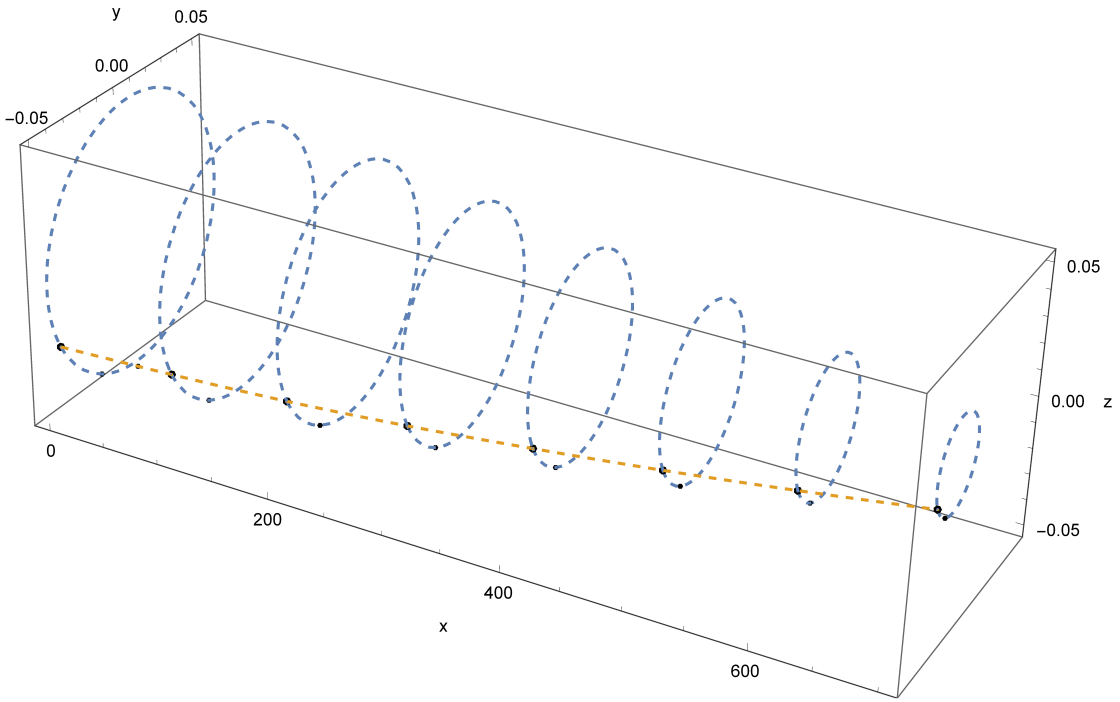
cpm	Y Disp.	Z Disp.
5000	-0.18129	-0.18129
6500	0.888715	0.888715

Rotor 3D Displacement Plot

Speed:6900.



Speed:8200.



Rotor Critical Speed&Log Decreasement

Log Decrement for the First Four Critical Speed:

Critical Speed(cpm)	Log Decrement
6892.3232	$0.\times10^{-8}$
14 547.654	$0.\times10^{-8}$
23 282.102	0.5891572
69 572.428	$0.\times10^{-8}$

In[2]:=

(*Main Program*)

In[3]:=
```
Start[] := Module[{},
  Print[Grid[{{"Mathematica Beam Analysis Package(MBAP)"}, {"Version:Alpha"},
     {"Written by: Ji Yiling"}, {"Test Use Only,Do Not Share to Others"}},
    Frame → True, FrameStyle → Gray, ItemStyle → Gray, Spacings → {10, 1}]];
  Print[];
  Print[];
  Grid[{{"----Data Input Module---", SpanFromLeft, SpanFromLeft},
    {Button["Step1", DataInput[], Method → "Queued"],
     Button["Step2", {nodcoord2, elenod2, aa2, dd2, ee2, ibeam, ksbeam} =
       PreProcessor[pp], Method → "Queued"],
     Button["Step3", AddConstrain[nodcoord2], Method → "Queued"]},
    {"----General Analysis Module----", SpanFromLeft, SpanFromLeft},
    {Button["Linear Static", SpaceTrussSolution[nodcoord2, elenod2, ee2, aa2,
       nodtag, nodval, nodspp, ibeam, ebeam, bsp, ksbeam], Method → "Queued"],
     Button["Linear Buckling", SpaceTrussBuckling[nodcoor2, aa2, elenod2,
       dd2, ee2, ibeam, nodtag, nodspp, conmall, ksbeam], Method → "Queued"],
     Button["Real Eigenvalue", SpaceTrussMode[nodcoord2, aa2, elenod2,
       dd2, ee2, ibeam, nodtag, nodspp, conmall, ksbeam], Method → "Queued"]},
    {Button["Complex Eigenvalue", SpaceTrussModeC[nodcoord2, aa2, elenod2,
       dd2, ee2, ibeam, nodtag, nodspp, conmall, ksbeam], Method → "Queued"],
     Button["Direct Frequency Response", {std1, std2} =
       SpaceTrussDFR[nodcoord2, elenod2, ee2, aa2, nodtag, nodval, nodsppd, ibeam,
        ww, conmall, dd2, ksbeam], Method → "Queued"], Button["Linear Transient",
      SpaceLinearTransient[nodcoord2, elenod2, ee2, aa2, nodtag, nodspp,
       ibeam, conmall, dd2, ksbeam, dt, in, nodvald], Method → "Queued"]},
    {"----Rotor Dynamic Module----", SpanFromLeft, SpanFromLeft},
    {Button["Undamped Rotor Plot", RotorMode[nodcoord2, aa2, elenod2, dd2, ee2,
       ibeam, nodtag, nodsppd, conmall, ros, bps, ksbeam], Method → "Queued"],
     Button["Rotor DFR Plot", RotorDFR[nodcoord2, elenod2, ee2,
       aa2, nodtag, nodval, nodsppd, ibeam, ww, conmall,
       dd2, nodum, rdfrnode, ksbeam], Method → "Queued"],
     Button["Rotor Disp. Plot3D", RotorDFR2[nodcoord2, elenod2,
       ee2, aa2, nodtag, nodval, nodsppd, ibeam, ww2,
       conmall, dd2, nodum, ksbeam], Method → "Queued"]},
    {Button["Rotor Stability", RotorModeC[nodcoord2, aa2, elenod2, dd2,
       ee2, ibeam, nodtag, nodspp, conmall, ksbeam], Method → "Queued"]}}]]
```

In[4]:=
```
DataInput[] := DynamicModule[{},
  pp = {{x1, y1, z1, x2, y2, z2, aa, dd, ee, Iyy, Izz, Ksy, Ksz, ν}};
 mn = 1;
  k = 1;
  Print[
   Grid[{{"Recommended Units", SpanFromLeft}, {"Length", "mm"}, {"Mass", "t"},
     {"Time", "Sec"}, {"Force", "N->t.mm/s^2"}, {"Stress&E", "MPa->N/mm^2"},
     {"Stiffness", "N/mm"}, {"Damping", "N.Sec/mm"}}, Frame → All]];
```

```
    ss = {{Dynamic[pp[[k, 1]]], Dynamic[pp[[k, 2]]], Dynamic[pp[[k, 3]]],
       Dynamic[pp[[k, 4]]], Dynamic[pp[[k, 5]]], Dynamic[pp[[k, 6]]],
       Dynamic[pp[[k, 7]]], Dynamic[pp[[k, 8]]], Dynamic[pp[[k, 9]]],
       Dynamic[pp[[k, 10]]], Dynamic[pp[[k, 11]]], Dynamic[pp[[k, 12]]],
       Dynamic[pp[[k, 13]]], Dynamic[pp[[k, 14]]]}};
    Print["Element Data Input:"];
    Print["aa->Area,dd->Material Density,ee->Ela.M"];
    Print["Iyy/Izz->Area Moment of Inertia,Ksy,Ksz->Shear Stiffness Factor"];
    Print["ν->Poisson's ratio"];
    Print[Grid[{{InputField[ss[[1, 1]], FieldSize → Tiny], InputField[ss[[1, 2]],
         FieldSize → Tiny], InputField[ss[[1, 3]], FieldSize → Tiny]},
       {InputField[ss[[1, 4]], FieldSize → Tiny], InputField[ss[[1, 5]],
         FieldSize → Tiny], InputField[ss[[1, 6]], FieldSize → Tiny]},
       {InputField[ss[[1, 7]], FieldSize → Tiny], InputField[ss[[1, 8]],
         FieldSize → Tiny], InputField[ss[[1, 9]], FieldSize → Tiny]},
       {InputField[ss[[1, 10]], FieldSize → Tiny], InputField[ss[[1, 11]],
         FieldSize → Tiny]}, {InputField[ss[[1, 12]], FieldSize → Tiny],
        InputField[ss[[1, 13]], FieldSize → Tiny]},
       {InputField[ss[[1, 14]], FieldSize → Tiny]}}, Frame → All]];
Print[Button["Add New Point", k = k + 1;
     pp = Insert[pp,
       {pp[[k - 1, 1]], pp[[k - 1, 2]], pp[[k - 1, 3]], pp[[k - 1, 4]], pp[[k - 1, 5]],
        pp[[k - 1, 6]], pp[[k - 1, 7]], pp[[k - 1, 8]], pp[[k - 1, 9]], pp[[k - 1, 10]],
        pp[[k - 1, 11]], pp[[k - 1, 12]], pp[[k - 1, 13]], pp[[k - 1, 14]]}, k];
     ss = Insert[ss, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, k];
     ss[[k]] = {Dynamic[pp[[k, 1]]], Dynamic[pp[[k, 2]]],
       Dynamic[pp[[k, 3]]], Dynamic[pp[[k, 4]]], Dynamic[pp[[k, 5]]],
       Dynamic[pp[[k, 6]]], Dynamic[pp[[k, 7]]], Dynamic[pp[[k, 8]]],
       Dynamic[pp[[k, 9]]], Dynamic[pp[[k, 10]]], Dynamic[pp[[k, 11]]],
       Dynamic[pp[[k, 12]]], Dynamic[pp[[k, 13]]], Dynamic[pp[[k, 14]]]};]];
    Print["Element Data:"];
    tt1 = {"No.", "x1", "y1", "z1", "x2", "y2", "z2", "Area",
      "Density", "Elastic.M", "Iyy", "Izz", "Ksy", "Ksz", "ν"};
    Print[sss = OpenerView[{Style["Element Data", Gray],
        Dynamic[Grid[Prepend[Join[Table[i, {i, Length[pp]}, {j, 1}], pp, 2], tt1],
          Dividers → {False, All}]]}, True]];
    elec = elecp = elecl = {{0, 0, 0}};
    ntt = {Text[0, {0, 0, 0}, {-1.5, -1.5}]};
    mpp = Manipulate[Graphics3D[{ntt, {PointSize[Large], Red, Point[elecp]},
        {Thick, Blue, Line[elecl]}}, Axes → True, AxesLabel → {x, y, z},
       FaceGrids → All, TicksStyle → Directive[Bold], PlotRangePadding →
        If[NumberQ[pp[[1, 7]]], 1.5 * Sqrt[pp[[1, 7]]], 10]], Button["Plot",
       elec = pp[[All, {1, 2, 3, 4, 5, 6}]];
       elec = Flatten[elec];
       elecp = Partition[elec, 3];
       elecl = Partition[elecp, 2];
```

```
    ]];
  Print[mpp];
Print["Element Data Modified:"];
  Print["Input the Modified Element Number First:"];
  Print[InputField[Dynamic[mn], Number, FieldSize → Tiny]];
  Print[Grid[{{InputField[Dynamic[pp[[mn, 1]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 2]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 3]]], FieldSize → Tiny]},
     {InputField[Dynamic[pp[[mn, 4]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 5]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 6]]], FieldSize → Tiny]},
     {InputField[Dynamic[pp[[mn, 7]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 8]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 9]]], FieldSize → Tiny]},
     {InputField[Dynamic[pp[[mn, 10]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 11]]], FieldSize → Tiny]},
     {InputField[Dynamic[pp[[mn, 12]]], FieldSize → Tiny],
      InputField[Dynamic[pp[[mn, 13]]], FieldSize → Tiny]},
     {InputField[Dynamic[pp[[mn, 14]]], FieldSize → Tiny]}}, Frame → All]];
  Print["Element Data Delete:"];
  Print["Input the Deleted Element Number First:"];
  Print[InputField[Dynamic[dln], Number, FieldSize → Tiny]];
  Print[Button["ClickToDelete", pp = Drop[pp, {dln}]; k = k - 1]];
  Print[Style["--------------------Next Step----------------------",
    FontSize → 16, FontWeight → Bold]];
 ]
```

In[5]:=
```
PreProcessor[raw_] :=
 Module[{rawcoord, nodcoord, elenod, rawaa, rawdd, rawee, rawibeam,
   aa, dd, ee, ibeam, i, nood, nft, rawksbeam, ksbeam, rawnu},
  rawcoord = raw[[All, {1, 2, 3, 4, 5, 6}]];
  rawcoord = Flatten[rawcoord];
  rawcoord = Partition[rawcoord, 3];
  nodcoord = Union[rawcoord];
  For[i = 1, i ≤ Length[rawcoord], i++,
   rawcoord[[i]] = Position[nodcoord, rawcoord[[i]]][[1]]];
  rawcoord = Flatten[rawcoord];
  elenod = Partition[rawcoord, 2];
  rawaa = raw[[All, 7]];
  aa = Flatten[rawaa];
  rawdd = raw[[All, 8]];
  dd = Flatten[rawdd];
  rawee = raw[[All, 9]];
  ee = Flatten[rawee];
  rawibeam = raw[[All, {10, 11}]];
  ibeam = rawibeam;
  rawksbeam = raw[[All, {12, 13}]];
  ksbeam = rawksbeam;
  rawnu = raw[[All, 14]];
  nua = Flatten[rawnu];
  nood = Length[nodcoord];
  ntt = {};
  For[i = 1, i ≤ nood, i++,
   ntt = AppendTo[ntt, Text[Style[i, FontColor → Black, FontWeight → Bold],
       nodcoord[[i]], {-1.5, -1.5}]]];
  CreatePalette[Button[Style["Click To Plot Node Number ",
     FontWeight → Bold, FontSize → 25], Print["Finish"];
    NotebookClose[ ButtonNotebook[]], FrameMargins → Automatic],
   WindowMargins → Automatic, WindowTitle → "Step2 Check"];
  Return[{nodcoord, elenod, aa, dd, ee, ibeam, ksbeam}]
 ]
```

In[6]:=
```
AddConstrain[nodcoord_] := Module[{},
  nodnum = Length[nodcoord];
  nodtag = Table[{0, 0, 0, 0, 0, 0}, {nodnum}];
  nodval = Table[{0, 0, 0, 0, 0, 0}, {nodnum}];
  nodum = Table[{0, 0, 0}, {nodnum}];
  nodsp = {0, 0, 0};
  nodspp = {};
  nodsppd = {};
  alldof = {0, 0, 0, 0, 0, 0};
  xyz = {0, 0, 0};
  nnn = 1;
```

```
ksp = 0;
dsp = 0;
kspd = {{1, 100}, {10, 100}};
dspd = {{1, 10}, {10, 10}};
ebeam = {-1, 0, 1};
bsp = {{-5, -5}, {5, 5}, {5, -5}, {-5, 5}};
Print["Apply DOF To All(0->Free,1->Constrained,Default=0):"];
Print[Grid[{{InputField[Dynamic[alldof[[1]]], FieldSize → Tiny],
    InputField[Dynamic[alldof[[2]]], FieldSize → Tiny],
    InputField[Dynamic[alldof[[3]]], FieldSize → Tiny]},
   {InputField[Dynamic[alldof[[4]]], FieldSize → Tiny],
    InputField[Dynamic[alldof[[5]]], FieldSize → Tiny],
    InputField[Dynamic[alldof[[6]]], FieldSize → Tiny]}}, Frame → All]];
Print[Button["Click to Apply", nodtag = Table[alldof, {nodnum}]]];
Print["Input Node Number:"];
Print[
 Grid[{{InputField[Dynamic[nnn], FieldSize → Tiny]}}, Frame → All]];
Print[Button["Select Node", xyz = nodcoord[[nnn]]]];
Print["Selected Node Coordinate is:"];
Print[Style[Dynamic[xyz], FontWeight → Bold, FontSize → 20, FontColor → Red]];
Print["Apply DOF To Selected Node(0->Free,1->Constrained,Default=0):"];
Print[Grid[{{InputField[Dynamic[nodtag[[nnn, 1]]], FieldSize → Tiny],
    InputField[Dynamic[nodtag[[nnn, 2]]], FieldSize → Tiny],
    InputField[Dynamic[nodtag[[nnn, 3]]], FieldSize → Tiny]},
   {InputField[Dynamic[nodtag[[nnn, 4]]], FieldSize → Tiny],
    InputField[Dynamic[nodtag[[nnn, 5]]], FieldSize → Tiny],
    InputField[Dynamic[nodtag[[nnn, 6]]], FieldSize → Tiny]}}, Frame → All]];
Print["Apply Force To Selected Node(Default=0):"];
Print[Grid[{{InputField[Dynamic[nodval[[nnn, 1]]], FieldSize → Tiny],
    InputField[Dynamic[nodval[[nnn, 2]]], FieldSize → Tiny],
    InputField[Dynamic[nodval[[nnn, 3]]], FieldSize → Tiny]},
   {InputField[Dynamic[nodval[[nnn, 4]]], FieldSize → Tiny],
    InputField[Dynamic[nodval[[nnn, 5]]], FieldSize → Tiny],
    InputField[Dynamic[nodval[[nnn, 6]]], FieldSize → Tiny]}}, Frame → All]];
Print["Apply Unbalance Mass to Selected Node(RotorDynamic Only):"];
Print[
 Grid[{{"Mass(m)", InputField[Dynamic[nodum[[nnn, 1]]], FieldSize → Tiny]},
   {"Distance(r)", InputField[Dynamic[nodum[[nnn, 2]]], FieldSize → Tiny]},
   {"Phase(rad)", InputField[Dynamic[nodum[[nnn, 3]]],
     FieldSize → Tiny]}}, Frame → All]];
Print["Stress Recovery Data:"]
 Print["Natural Coordinate(from -1 to 1)"];
Print[
 Grid[{{InputField[Dynamic[ebeam], FieldSize → Small]}}, Frame → All]];
Print["Stress Recovery Local Coordinate(Y&Z)"];
Print[
```

```
 Grid[{{InputField[Dynamic[bsp], FieldSize → Large]}}, Frame → All]];
Print["Add Elastic Spring To Selected Node:"];
Print["Input the Spring Coordinate Related to the Selected Node:"];
Print[Grid[{{InputField[Dynamic[nodsp[[1]]], FieldSize → Tiny],
    InputField[Dynamic[nodsp[[2]]], FieldSize → Tiny],
    InputField[Dynamic[nodsp[[3]]], FieldSize → Tiny]}}, Frame → All]];
Print["Input Spring Constant K:"];
Print[Grid[{{InputField[Dynamic[ksp], FieldSize → Tiny]}}, Frame → All]];
Print["Input Spring Damping Cofficient B:"];
Print[Grid[{{InputField[Dynamic[dsp], FieldSize → Tiny]}}, Frame → All]];
Print[Button["Add Spring",
  nodspp = AppendTo[nodspp, {xyz, {xyz[[1]] + nodsp[[1]],
      xyz[[2]] + nodsp[[2]], xyz[[3]] + nodsp[[3]]}, ksp, dsp, nnn}]]];
Print["Spring Modified:"];
nn2 = 1;
Print[Grid[{{InputField[Dynamic[nn2], FieldSize → Tiny]}}, Frame → All]];
Print["Spring Position:"];
Print[{Dynamic[nodspp[[nn2, 1]]], Dynamic[nodspp[[nn2, 2]]]}];
Print["K:"];
Print[Grid[
  {{InputField[Dynamic[nodspp[[nn2, 3]]], FieldSize → Tiny]}}, Frame → All]];
Print["B:"];
Print[Grid[
  {{InputField[Dynamic[nodspp[[nn2, 4]]], FieldSize → Tiny]}}, Frame → All]];
Print["Add Dynamic Elastic Spring To Selected Node:"];
Print["Input the Spring Coordinate Related to the Selected Node:"];
Print[Grid[{{InputField[Dynamic[nodsp[[1]]], FieldSize → Tiny],
    InputField[Dynamic[nodsp[[2]]], FieldSize → Tiny],
    InputField[Dynamic[nodsp[[3]]], FieldSize → Tiny]}}, Frame → All]];
Print["Input Dynamic Spring Constant K(eg.{{cpm1,k1},{cpm2,k2}}):"];
Print[Grid[{{InputField[Dynamic[kspd], FieldSize → Large]}}, Frame → All]];
Print[
 "Input Dynamic Spring Damping Cofficient B(eg.{{cpm1,b1},{cpm2,b2}}):"];
Print[Grid[{{InputField[Dynamic[dspd], FieldSize → Large]}}, Frame → All]];
Print[Button["Add Spring",
  nodsppd = AppendTo[nodsppd, {xyz, {xyz[[1]] + nodsp[[1]],
      xyz[[2]] + nodsp[[2]], xyz[[3]] + nodsp[[3]]}, kspd, dspd, nnn}]]];
Print["Dynamic Spring Modified:"];
nn3 = 1;
Print[Grid[{{InputField[Dynamic[nn3], FieldSize → Tiny]}}, Frame → All]];
Print["Spring Position:"];
Print[{Dynamic[nodsppd[[nn3, 1]]], Dynamic[nodsppd[[nn3, 2]]]}];
Print["K:"];
Print[Grid[{{InputField[Dynamic[nodsppd[[nn3, 3]]], FieldSize → Large]}},
  Frame → All]];
Print["B:"];
```

```
Print[Grid[{{InputField[Dynamic[nodsppd[[nn3, 4]]], FieldSize → Large]}},
  Frame → All]];
Print["Add Concentrated Mass:"];
conm = {m, i11, i22, i33, i21, i31, i32};
nodcon = {0};
conmall = {};
Print[
 Grid[{{InputField[Dynamic[nodcon[[1]]], FieldSize → Tiny]}}, Frame → All]];
Print[Grid[{{InputField[Dynamic[conm[[1]]], FieldSize → Tiny],
    InputField[Dynamic[conm[[2]]], FieldSize → Tiny],
    InputField[Dynamic[conm[[3]]], FieldSize → Tiny],
    InputField[Dynamic[conm[[4]]], FieldSize → Tiny]},
   {InputField[Dynamic[conm[[5]]], FieldSize → Tiny],
    InputField[Dynamic[conm[[6]]], FieldSize → Tiny],
    InputField[Dynamic[conm[[7]]], FieldSize → Tiny]}}, Frame → All]];
Print[Button["Add Con.Mass", conmall =
   AppendTo[conmall, {nodcon[[1]], conm}]]];
Print[Dynamic[conmall // MatrixForm]];
conmdel = 1;
Print["Con.Mass Delete:"];
Print[Grid[{{InputField[Dynamic[conmdel], FieldSize → Tiny]}}, Frame → All]];
If[ArrayQ[Dynamic[conmall]], Null,
 Print[Button["Delete", conmall = Drop[conmall, {conmdel}]]]];
ww = {0, 0, 0};
Print["DFR Input"];
Print[
 Grid[{{"Start Frequency", InputField[Dynamic[ww[[1]]], FieldSize → Tiny]},
   {"End Frequency", InputField[Dynamic[ww[[2]]], FieldSize → Tiny]},
   {"Increasement", InputField[Dynamic[ww[[3]]], FieldSize → Tiny]}}, Frame →
   All]];
Print["Rotor DFR Plot Node Number"];
rdfrnode = {1, 2};
Print[
 Grid[{{InputField[Dynamic[rdfrnode], FieldSize → Small]}}, Frame → All]];
Print["Rpm for 3D Rotor Disp.Plot(Input After RDFR Analysis)"];
ww2 = {1};
Print[Grid[{{InputField[Dynamic[ww2], FieldSize → Small]}}, Frame → All]];
Print["Rotor Operation Speed Range(cpm)"];
ros = {0, 0};
Print[Grid[{{InputField[Dynamic[ros], FieldSize → Small]}}, Frame → All]];
Print["Bearing Stiffness Range(Rotor Dynamic Only)"];
bps = {0, 0};
Print[Grid[{{InputField[Dynamic[bps], FieldSize → Small]}}, Frame → All]];
Print["Transient Analysis Parameter:"];
dt = 0; in = 0;
Print[Grid[{{"Time increment", InputField[Dynamic[dt], FieldSize → Tiny]},
```

```
      {"Number of time steps",
       InputField[Dynamic[in], FieldSize → Tiny]}}, Frame → All]];
    Print[Style["-----------------------------------------------------------",
      FontSize → 16, FontWeight → Bold]];
    Print[Style["Final Result is:", FontSize → 18,
      FontWeight → Bold, FontSlant → Italic]]
   ]

  RDFRInput[nodspp_] := Module[{nnn, sppfa},
    nnn = Length[nodspp];
    spn = 1;
    sppf = {"Input"};
    sppfa = {};
    Print[InputField[Dynamic[spn]]];
    Print[Dynamic[nodspp[[spn, 1]]]];
    Print[InputField[Dynamic[sppf]]];
    Print[Button["Add Table to Spring",
      sppfa = Append[sppfa, {spn, Interpolation[sppf]}]]];
    Return[sppfa]
   ]

In[7]:= SpaceBeamStiffness[ncoor_, Em_, A_, Iyy_, Izz_, ksy_, ksz_, ν_] :=
   Module[{Kebar, l, x0, y0, z0, xm, ym, zm, x1, y1, z1, x2, y2, z2,
     tzx, tzy, tzz, tyx, tyy, tyz, txx, txy, txz, Te, Ke, ll, lll, zL,
     dx, dy, dz, EA, x21, y21, z21, Gm, Ip, dd, phiy, phiz}, EA = A Em;
    {{x1, y1, z1}, {x2, y2, z2}} = ncoor;
    Gm = Em / (2 (1 + ν));
    dd = 2 √(A / π);
    Ip = Iyy + Izz;
    ll = (x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2;
    l = √ll;
    lll = l ll;
    {x21, y21, z21} = {x2 - x1, y2 - y1, z2 - z1};
    phiy = 12 Em Iyy / (Gm A ksy ll);
    phiz = 12 Em Izz / (Gm A ksz ll);
    Kebar = {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
        {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, Gm Ip / l, 0, 0, 0, 0, 0, -Gm Ip / l, 0, 0},
        {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
        {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
        {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, -Gm Ip / l, 0, 0, 0, 0, 0, Gm Ip / l, 0, 0},
        {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}} +
```

```
  1
-------- Em Iyy {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0,
 3
l  (phiy + 1)
     0, 0, 0, 0}, {0, 0, 12, 0, -6 l, 0, 0, 0, -12, 0, -6 l, 0}, {0, 0, 0, 0, 0, 0, 0,
     0, 0, 0, 0, 0}, {0, 0, -6 l, 0, (phiy + 4) l^2, 0, 0, 0, 6 l, 0, (2 - phiy) l^2, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, -12, 0, 6 l, 0, 0, 0, 12, 0, 6 l, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, -6 l, 0, (2 - phiy) l^2, 0, 0, 0, 6 l,
     0, (phiy + 4) l^2, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}} + 1 / l^3 (phiz + 1) Em Izz
   {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 12, 0, 0, 0, 6 l, 0, -12, 0, 0, 0, 6 l},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 6 l, 0, 0, 0, (phiz + 4) l^2, 0, -6 l,
     0, 0, 0, (2 - phiz) l^2}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, -12, 0,
     0, 0, -6 l, 0, 12, 0, 0, 0, -6 l}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 6 l, 0, 0, 0, (2 - phiz) l^2, 0, -6 l, 0, 0, 0, (phiz + 4) l^2}} +
  1 / l EA {{1, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {-1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}};
{x0, y0, z0} = {xm, ym, zm} = 1/2 {x1 + x2, y1 + y2, z1 + z2};
If[x1 == x2, {x0, y0, z0} += {1, 0, 0}, {x0, y0, z0} += {0, 0, 1}];
{dx, dy, dz} = {x0 - xm, y0 - ym, z0 - zm};
tzx = dz y21 - dy z21;
tzy = dx z21 - dz x21;
tzz = dy x21 - dx y21;
zL = Sqrt[tzx^2 + tzy^2 + tzz^2];
{tzx, tzy, tzz} = {tzx, tzy, tzz} / zL;
{txx, txy, txz} = {x21, y21, z21} / l;
tyx = txz tzy - txy tzz;
tyy = txx tzz - txz tzx;
tyz = txy tzx - txx tzy;
Te = {{txx, txy, txz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {tyx, tyy, tyz, 0, 0, 0, 0, 0, 0, 0,
    0, 0}, {tzx, tzy, tzz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, txx, txy, txz,
    0, 0, 0, 0, 0, 0}, {0, 0, 0, tyx, tyy, tyz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, tzx,
    tzy, tzz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, txx, txy, txz, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, tyx, tyy, tyz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, tzx, tzy,
    tzz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, txx, txy, txz}, {0, 0, 0, 0, 0,
    0, 0, 0, 0, tyx, tyy, tyz}, {0, 0, 0, 0, 0, 0, 0, 0, 0, tzx, tzy, tzz}};
Ke = Transpose[Te].Kebar.Te;
Return[Ke];]
```

In[8]:=
```
BeamDifferentialStiffness[ncoor_, eleaf_, Em_, A_, Iyy_, ksy_, ν_] :=
 Module[{x1, x2, y1, y2, z1, z2, ll, l, kdbar, x0, y0, z0, xm, ym, zm,
   tzx, tzy, tzz, tyx, tyy, tyz, txx, txy, txz, dx, dy, dz, x21, y21,
   z21, zL, Te, Kd, L, phi, Gm}, {{x1, y1, z1}, {x2, y2, z2}} = ncoor;
  ll = (x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2;
  {x21, y21, z21} = {x2 - x1, y2 - y1, z2 - z1};
  l = L = Sqrt[ll];
  Gm = Em/(2 (ν + 1));
  phi = (12 Em Iyy)/(Gm A ksy ll);
  kdbar = eleaf {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
      0, 0}, {0, 0, (5 + 1/(phi+1)^2)/(5 L), 0, -1/(10 (phi + 1)^2), 0, 0, 0, -(5 + 1/(phi+1)^2)/(5 L), 0,
     -1/(10 (phi + 1)^2), 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, -1/(10 (phi + 1)^2),
     0, 1/60 L (5 + 3/(phi + 1)^2), 0, 0, 0, 1/(10 (phi + 1)^2), 0, 1/60 L (3/(phi + 1)^2 - 5), 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, -(5 + 1/(phi+1)^2)/(5 L), 0, 1/(10 (phi + 1)^2), 0,
     0, 0, (5 + 1/(phi+1)^2)/(5 L), 0, 1/(10 (phi + 1)^2), 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, -1/(10 (phi + 1)^2), 0, 1/60 L (3/(phi + 1)^2 - 5), 0, 0, 0, 1/(10 (phi + 1)^2),
     0, 1/60 L (5 + 3/(phi + 1)^2), 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}} +
   eleaf {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, (5 + 1/(phi+1)^2)/(5 L), 0, 0, 0, 1/(10 (phi + 1)^2),
     0, -(5 + 1/(phi+1)^2)/(5 L), 0, 0, 0, 1/(10 (phi + 1)^2)}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {0, 1/(10 (phi + 1)^2), 0, 0, 0, 1/60 L (5 + 3/(phi + 1)^2), 0, -1/(10 (phi + 1)^2), 0,
     0, 0, 1/60 L (3/(phi + 1)^2 - 5)}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0,
     -(5 + 1/(phi+1)^2)/(5 L), 0, 0, 0, -1/(10 (phi + 1)^2), 0, (5 + 1/(phi+1)^2)/(5 L), 0, 0, 0, -1/(10 (phi + 1)^2)},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
```

```
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 1/(10 (phi + 1)^2), 0, 0, 0,
     1/60 L (3/(phi + 1)^2 - 5), 0, -1/(10 (phi + 1)^2), 0, 0, 0, 1/60 L (5 + 3/(phi + 1)^2)}};
  {x0, y0, z0} = {xm, ym, zm} = 1/2 {x1 + x2, y1 + y2, z1 + z2};
  If[x1 == x2, {x0, y0, z0} += {1, 0, 0}, {x0, y0, z0} += {0, 1, 0}];
  {dx, dy, dz} = {x0 - xm, y0 - ym, z0 - zm};
  tzx = dz y21 - dy z21;
  tzy = dx z21 - dz x21;
  tzz = dy x21 - dx y21;
  zL = Sqrt[tzx^2 + tzy^2 + tzz^2];
  {tzx, tzy, tzz} = {tzx, tzy, tzz}/zL;
  {txx, txy, txz} = {x21, y21, z21}/l;
  tyx = txz tzy - txy tzz;
  tyy = txx tzz - txz tzx;
  tyz = txy tzx - txx tzy;
  Te = {{txx, txy, txz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {tyx, tyy, tyz, 0, 0, 0, 0, 0, 0, 0,
      0, 0}, {tzx, tzy, tzz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, txx, txy, txz,
      0, 0, 0, 0, 0, 0}, {0, 0, 0, tyx, tyy, tyz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, tzx,
      tzy, tzz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, txx, txy, txz, 0, 0, 0},
    {0, 0, 0, 0, 0, 0, tyx, tyy, tyz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, tzx, tzy,
     tzz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, txx, txy, txz}, {0, 0, 0, 0, 0,
     0, 0, 0, 0, tyx, tyy, tyz}, {0, 0, 0, 0, 0, 0, 0, 0, 0, tzx, tzy, tzz}};
  Kd = Transpose[Te].kdbar.Te;
  Return[Kd];]

(*Test Data*)
ClearAll[L, Em, A, Izz, Iyy];
ncoor = {{0, 0, 0}, {3, 4, 0}}; Em = 100;
A = 125; Izz = 250; Iyy = 250;
Ke = SpaceBeamStiffness[ncoor, Em, A, Iyy, Izz, 1, 1];
Print["Numerical Elem Stiff Matrix: "];
Print[SetPrecision[Ke, 4] // MatrixForm];
Print["Eigenvalues of Ke=", Chop[Eigenvalues[Ke]]];
Print["Eigenvectors of Ke=", Chop[Eigenvectors[Ke] // MatrixForm]];
```

In[9]:=

```
SpaceBeamMass[ncoor_, A_, des_, Iyy_, Izz_, Em_, ksy_, ksz_, ν_] :=
 Module[{x1, x2, y1, y2, z1, z2, x0, y0, z0, xm, ym, zm, D, l, ll, dx, dy, dz, zL,
   txx, txy, txz, tyx, tyy, tyz, tzx, tzy, tzz, x21, y21, z21, Te, Me, Mebar,
   me, ip, dd, L, iiR, phiy, phiz, Gm}, {{x1, y1, z1}, {x2, y2, z2}} = ncoor;
  D = des;
  {x21, y21, z21} = {x2 - x1, y2 - y1, z2 - z1};
  ll = (x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2;
```

```
l = L = √ll ;
me = A des l;
Gm = Em / (2 (1 + ν));
dd = 2 √(A/π) ;
phiy = 12 Em Iyy / (Gm A ksy ll);
phiz = 12 Em Izz / (Gm A ksz ll);
ip = Iyy + Izz;
iiR = (Iyy + Izz) / 2;
(*Mebar={{0,0,0,0,0,0,0,0,0,0,0,0},
    {0,(252 D iiR+L me (78+7 phiz (21+10 phiz)))/(210 L (1+phiz)^2),0,0,0,
     (84 D iiR (1-5 phiz)+L me (44+7 phiz (11+5 phiz)))/(840 (1+phiz)^2),
     0,(-252 D iiR+L me (27+7 phiz (9+5 phiz)))/(210 L (1+phiz)^2),0,0,0,
     (84 D iiR (1-5 phiz)-L me (26+7 phiz (9+5 phiz)))/(840 (1+phiz)^2)},
    {0,0,(252 D iiR+L me (78+7 phiy (21+10 phiy)))/(210 L (1+phiy)^2),
     0,(84 D iiR (-1+5 phiy)-L me (44+7 phiy (11+5 phiy)))/(840 (1+phiy)^2),
     0,0,0,(-252 D iiR+L me (27+7 phiy (9+5 phiy)))/(210 L (1+phiy)^2),0,
     (84 D iiR (-1+5 phiy)+L me (26+7 phiy (9+5 phiy)))/(840 (1+phiy)^2),
     0},{0,0,0,0,0,0,0,0,0,0,0,0},{0,0,
     (84 D iiR (-1+5 phiy)-L me (44+7 phiy (11+5 phiy)))/(840 (1+phiy)^2),
     0,(L (L me (8+7 phiy (2+phiy))+28 D iiR (4+5 phiy (1+2 phiy))))/(840
        (1+phiy)^2),0,0,0,(84 D iiR (1-5 phiy)-L me (26+7 phiy (9+5 phiy)))/
      (840 (1+phiy)^2),0,-((L (28 D iiR (1-5 (-1+phiy) phiy)+
            L me (6+7 phiy (2+phiy))))/(840 (1+phiy)^2)),0},
    {0,(84 D iiR (1-5 phiz)+L me (44+7 phiz (11+5 phiz)))/(840 (1+phiz)^2),
     0,0,0,(L (L me (8+7 phiz (2+phiz))+28 D iiR (4+5 phiz (1+2 phiz))))/
      (840 (1+phiz)^2),0,(84 D iiR (-1+5 phiz)+L me (26+7 phiz (9+5 phiz)))/
      (840 (1+phiz)^2),0,0,0,-((L (28 D iiR (1-5 (-1+phiz) phiz)+
            L me (6+7 phiz (2+phiz))))/(840 (1+phiz)^2))},{0,0,0,0,0,0,0,0,0,
     0,0,0},{0,(-252 D iiR+L me (27+7 phiz (9+5 phiz)))/(210 L (1+phiz)^2),0,
     0,0,(84 D iiR (-1+5 phiz)+L me (26+7 phiz (9+5 phiz)))/(840 (1+phiz)^2),
     0,(252 D iiR+L me (78+7 phiz (21+10 phiz)))/(210 L (1+phiz)^2),0,0,0,
     (84 D iiR (-1+5 phiz)-L me (44+7 phiz (11+5 phiz)))/(840 (1+phiz)^2)},
    {0,0,(-252 D iiR+L me (27+7 phiy (9+5 phiy)))/(210 L (1+phiy)^2),
     0,(84 D iiR (1-5 phiy)-L me (26+7 phiy (9+5 phiy)))/(840 (1+phiy)^2),
     0,0,0,(252 D iiR+L me (78+7 phiy (21+10 phiy)))/(210 L (1+phiy)^2),0,
     (84 D iiR (1-5 phiy)+L me (44+7 phiy (11+5 phiy)))/(840 (1+phiy)^2),
```

```
    0},{0,0,0,0,0,0,0,0,0,0,0,0},
   {0,0,(84 D iiR (-1+5 phiy)+L me (26+7 phiy (9+5 phiy)))/(840 (1+phiy)^2),
    0,-((L (28 D iiR (1-5 (-1+phiy) phiy)+L me (6+7 phiy (2+phiy))))/
       (840 (1+phiy)^2)),0,0,0,
    (84 D iiR (1-5 phiy)+L me (44+7 phiy (11+5 phiy)))/(840 (1+phiy)^2),
    0,(L (L me (8+7 phiy (2+phiy))+28 D iiR (4+5 phiy (1+2 phiy))))/(840
       (1+phiy)^2),0},{0,(84 D iiR (1-5 phiz)-L me (26+7 phiz (9+5 phiz)))/
    (840 (1+phiz)^2),0,0,0,-((L (28 D iiR (1-5 (-1+phiz) phiz)+
          L me (6+7 phiz (2+phiz))))/(840 (1+phiz)^2)),0,
    (84 D iiR (-1+5 phiz)-L me (44+7 phiz (11+5 phiz)))/(840 (1+phiz)^2),
    0,0,0,(L (L me (8+7 phiz (2+phiz))+28 D iiR (4+5 phiz (1+2 phiz))))/
    (840 (1+phiz)^2)}}+1/6 A D l {{2,0,0,0,0,0,1,0,0,0,0,0},
   {0,0,0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0,0,0},
   {0,0,0,2 ip/A,0,0,0,0,ip/A,0,0},{0,0,0,0,0,0,0,0,0,0,0,0},
   {0,0,0,0,0,0,0,0,0,0,0,0},{1,0,0,0,0,0,2,0,0,0,0,0},{0,0,0,0,0,0,0,
    0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0,0,0},{0,0,0,ip/A,0,0,0,0,0,2 ip/A,0,0},
   {0,0,0,0,0,0,0,0,0,0,0,0},{0,0,0,0,0,0,0,0,0,0,0,0}};*)
Mebar = A D l {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 13/35, 0, -1/210 (11 l), 0, 0, 0, 9/70, 0, 13 l/420, 0},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, -1/210 (11 l), 0, l^2/105, 0,
    0, 0, -1/420 (13 l), 0, -l^2/140, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 9/70, 0, -1/420 (13 l), 0, 0, 0, 13/35, 0, 11 l/210, 0},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 13 l/420, 0, -l^2/140, 0,
    0, 0, 11 l/210, 0, l^2/105, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}} +
  A D l {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 13/35, 0, 0, 0, 11 l/210, 0,
    9/70, 0, 0, 0, -1/420 (13 l)}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 11 l/210, 0, 0, 0, l^2/105, 0, 13 l/420, 0, 0, 0, -l^2/140},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 9/70, 0, 0, 0, 13 l/420, 0,
    13/35, 0, 0, 0, -1/210 (11 l)}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
   {0, -1/420 (13 l), 0, 0, 0, -l^2/140, 0, -1/210 (11 l), 0, 0, 0, l^2/105}} +
```

$$\frac{1}{6}\,A\,D\,l\,\{\{2,0,0,0,0,0,1,0,0,0,0,0\},\{0,0,0,0,0,0,0,0,0,0,0,0\},$$
$$\{0,0,0,0,0,0,0,0,0,0,0,0\},\left\{0,0,0,\frac{2\,ip}{A},0,0,0,0,0,\frac{ip}{A},0,0\right\},$$
$$\{0,0,0,0,0,0,0,0,0,0,0,0\},\{0,0,0,0,0,0,0,0,0,0,0,0\},$$
$$\{1,0,0,0,0,0,2,0,0,0,0,0\},\{0,0,0,0,0,0,0,0,0,0,0,0\},$$
$$\{0,0,0,0,0,0,0,0,0,0,0,0\},\left\{0,0,0,\frac{ip}{A},0,0,0,0,0,\frac{2\,ip}{A},0,0\right\},$$
$$\{0,0,0,0,0,0,0,0,0,0,0,0\},\{0,0,0,0,0,0,0,0,0,0,0,0\}\};$$

```
{x0, y0, z0} = {xm, ym, zm} = 1/2 {x1 + x2, y1 + y2, z1 + z2};
If[x1 == x2, {x0, y0, z0} += {1, 0, 0}, {x0, y0, z0} += {0, 1, 0}];
{dx, dy, dz} = {x0 - xm, y0 - ym, z0 - zm};
tzx = dz y21 - dy z21;
tzy = dx z21 - dz x21;
tzz = dy x21 - dx y21;
zL = Sqrt[tzx^2 + tzy^2 + tzz^2];
{tzx, tzy, tzz} = {tzx, tzy, tzz}/zL;
{txx, txy, txz} = {x21, y21, z21}/l;
tyx = txz tzy - txy tzz;
tyy = txx tzz - txz tzx;
tyz = txy tzx - txx tzy;
Te = {{txx, txy, txz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {tyx, tyy, tyz, 0, 0, 0, 0, 0, 0, 0,
    0, 0}, {tzx, tzy, tzz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, txx, txy, txz,
    0, 0, 0, 0, 0, 0}, {0, 0, 0, tyx, tyy, tyz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, tzx,
    tzy, tzz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, txx, txy, txz, 0, 0, 0},
   {0, 0, 0, 0, 0, 0, tyx, tyy, tyz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, tzx, tzy,
    tzz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, txx, txy, txz}, {0, 0, 0, 0, 0,
    0, 0, 0, 0, tyx, tyy, tyz}, {0, 0, 0, 0, 0, 0, 0, 0, 0, tzx, tzy, tzz}};
Me = Transpose[Te].Mebar.Te;
Return[Me];]

ClearAll[L, Em, A, Izz, Iyy];
ncoor = {{100, 0, 0}, {200, 0, 0}}; Em = 10 000;
A = 125; Izz = 250; Iyy = 250; des = 500;
Me = SpaceBeamMass[ncoor, A, des, Iyy, Izz, Em, 1, 1];
Print["Numerical Elem Stiff Matrix: "];
Print[N[Me, 10] // MatrixForm];
Print["Eigenvalue of Ke=", N[Chop[Eigenvalues[Me]] // MatrixForm]]
```

In[10]:=
```
SpaceTrassMasterStiffness[nodcoor_, elenod_, Aa_, elemat_, elemi_, ksbeam_] :=
 Module[{numele = Length[elenod], numnod = Length[nodcoor],
   K, e, ni, nj, eftab, Ke, ncoor, Em, A, Iyy, Izz, ksy, ksz, ν},
  K = Table[0, {6 * numnod}, {6 * numnod}];
  For[e = 1, e ≤ numele, e++, {ni, nj} = elenod[[e]];
   eftab = {6 * ni - 5, 6 * ni - 4, 6 * ni - 3, 6 * ni - 2, 6 * ni - 1,
     6 * ni, 6 * nj - 5, 6 * nj - 4, 6 * nj - 3, 6 * nj - 2, 6 * nj - 1, 6 * nj};
   ncoor = nodcoor[[elenod[[e]]]];
   Em = elemat[[e]];
   A = Aa[[e]];
   {Iyy, Izz} = elemi[[e]];
   {ksy, ksz} = ksbeam[[e]];
   ν = nua[[e]];
   Ke = SpaceBeamStiffness[ncoor, Em, A, Iyy, Izz, ksy, ksz, ν];
   K[[eftab, eftab]] += Ke;];
  Return[K];]
```

In[11]:=
```
SpaceTrassMasterDiffStiffness[nodcoor_, elenod_, Aa_, elemat_, elemi_, ksbeam_] :=
 Module[{numele = Length[elenod], numnod = Length[nodcoor], Kd,
   e, ni, nj, eftab, eaf, ncoor, Em, A, Kde, Iyy, Izz, ksy, ksz, ν},
  Kd = Table[0, {6 * numnod}, {6 * numnod}];
  For[e = 1, e ≤ numele, e++, {ni, nj} = elenod[[e]];
   eftab = {6 * ni - 5, 6 * ni - 4, 6 * ni - 3, 6 * ni - 2, 6 * ni - 1,
     6 * ni, 6 * nj - 5, 6 * nj - 4, 6 * nj - 3, 6 * nj - 2, 6 * nj - 1, 6 * nj};
   ncoor = nodcoor[[elenod[[e]]]];
   Em = elemat[[e]]; A = Aa[[e]];
   {Iyy, Izz} = elemi[[e]];
   {ksy, ksz} = ksbeam[[e]];
   eaf = pafa[[e]];
   ν = nua[[e]];
   Kde = BeamDifferentialStiffness[ncoor, eaf, Em, A, Iyy, ksy, ν];
   Kd[[eftab, eftab]] += Kde;];
  Return[Kd];]
```

In[12]:=
```
SpaceTrassMasterMass[nodcoor_,
  elenod_, Aa_, eledes_, elemi_, elemat_, ksbeam_] :=
 Module[{numele = Length[elenod],
   numnod = Length[nodcoor], M, e, ni, nj, eftab,
  Ke, ncoor, Em, A, Iyy, Izz, q, ksy, ksz, des, Me, ν},
  M = Table[0, {6 * numnod}, {6 * numnod}];
  For[e = 1, e <= numele, e++, {ni, nj} = elenod[[e]];
   eftab =
    {6 * ni - 5, 6 * ni - 4, 6 * ni - 3, 6 * ni - 2, 6 * ni - 1, 6 * ni, 6 * nj - 5,
   6 * nj - 4, 6 * nj - 3, 6 * nj - 2, 6 * nj - 1, 6 * nj};
   ncoor = nodcoor[[elenod[[e]]]]; A = Aa[[e]]; des = eledes[[e]];
   {Iyy, Izz} = elemi[[e]];
   Em = elemat[[e]];
   {ksy, ksz} = ksbeam[[e]];
   ν = nua[[e]];
   Me = SpaceBeamMass[ncoor, A, des, Iyy, Izz, Em, ksy, ksz, ν];
   M[[eftab, eftab]] += Me; ];
  For[q = 1, q <= 6 * numnod, q++, If[M[[q, q]] == 0, M[[q, q]] = 1]];
  Return[M]; ]

ClearAll[nodcoor, elemat, elemi];
nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
elemat = Table[100, {3}]; elemi = {{100, 100}, {200, 200}, {300, 300}} * 100;
Aa = {100, 100, 100};
eledes = {100, 100, 100};
ksbeam = {{1, 1}, {1, 1}, {1, 1}};
K = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
Print["Master Stiffness of Example Truss in 3D:\n",
 MatrixForm[N[K]]];
Print["eigs of K:", Chop[Eigenvalues[N[K]]]];

(*Test Data*)
ClearAll[nodcoor, elemat, elemi];
nodcoor = {{0, 0, 0}, {10, 0, 0}, {50, 0, 0}};
elenod = {{1, 2}, {2, 3}};
elemat = Table[100., {3}]; elemi = {{10., 15.}, {10., 15.}, {10., 15.}};
Aa = {20., 20., 20.};
K = SpaceTrassMasterStiffness[
   nodcoor, elenod, Aa, elemat, elemi, {{1, 1}, {1, 1}}];
K = SetPrecision[K, 80];
K[[1, 1]]
Print[K];
Print["Master Stiffness of Example Truss in 3D:\n",
 MatrixForm[N[K]]];
Print["eigs of K:", Chop[Eigenvalues[N[K]]]];
```

In[13]:=
```
PrescDisplacementDOFTags[nodtag_] :=
  Module[{j, n, numnod = Length[nodtag], pdof = {}, k = 0, m},
   For[n = 1, n ≤ numnod, n++, m = Length[nodtag[[n]]];
    For[j = 1, j ≤ m, j++, If[nodtag[[n, j]] > 0, AppendTo[pdof, k + j]];];
    k += m;];
   Return[pdof]];
PrescDisplacementDOFValues[nodtag_, nodval_] :=
  Module[{j, n, numnod = Length[nodtag], pval = {}, k = 0, m},
   For[n = 1, n ≤ numnod, n++, m = Length[nodtag[[n]]];
    For[j = 1, j ≤ m, j++, If[nodtag[[n, j]] > 0, AppendTo[pval, nodval[[n, j]]]];];
    k += m;];
   Return[pval]];

(*Test Data*)
PrescDisplacementDOFTags[
 {{1, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 1}}]
PrescDisplacementDOFValues[{{1, 0, 0, 0, 1, 0}, {0, 0, 0, 1, 0}, {0, 0, 0, 0, 1}},
 {{0, 0, 0, 0, 20, 20}, {20, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}}]
```

In[15]:=
```
ApplyDofToMaster[nodtag_, K_] := Module[{i, j, k, n = Length[K], pdof, np, Kmod = K},
   pdof = PrescDisplacementDOFTags[nodtag];
   np = Length[pdof];
   For[k = 1, k ≤ np, k++, i = pdof[[k]];
    For[j = 1, j ≤ n, j++, Kmod[[i, j]] = Kmod[[j, i]] = 0];
    Kmod[[i, i]] = 1];
   Return[Kmod]];
ApplyLoad[nodtag_, nodval_, K_, f_] :=
  Module[{i, j, k, n = Length[K], pdof, pval, np, d, c, fmod = f},
   pdof = PrescDisplacementDOFTags[nodtag];
   np = Length[pdof];
   pval = PrescDisplacementDOFValues[nodtag, nodval];
   c = Table[1, {n}];
   For[k = 1, k ≤ np, k++, i = pdof[[k]]; c[[i]] = 0];
   For[k = 1, k ≤ np, k++, i = pdof[[k]]; d = pval[[k]];
    fmod[[i]] = d; If[d == 0, Continue[]];
    For[j = 1, j ≤ n, j++, fmod[[j]] -= K[[i, j]] * c[[j]] * d];];
   Return[fmod]];
```

In[17]:=
```
ApplySpringToMaster[nodspp_, K_] :=
 Module[{nodsp1, ppd, spn, i, ppdn, j, kss, n, nsp, ksp, kf, bsp, bf},
  kf = K;
  nsp = Length[nodspp];
  For[i = 1, i ≤ nsp, i++,
   nodsp1 = nodspp[[i]];
   n = nodspp[[i, 5]];
   {ksp, bsp} = SpaceSpring[nodsp1];
   kf[[{6 * n - 5, 6 * n - 4, 6 * n - 3}, {6 * n - 5, 6 * n - 4, 6 * n - 3}]] += ksp];
  Return[kf]]
```

In[18]:=
```
ApplySpringToMasterB[nodspp_, B_] :=
 Module[{nodsp1, ppd, spn, i, ppdn, j, kss, n, nsp, ksp, kf, bsp, bf},
  bf = B;
  nsp = Length[nodspp];
  For[i = 1, i ≤ nsp, i++,
   nodsp1 = nodspp[[i]];
   n = nodspp[[i, 5]];
   {ksp, bsp} = SpaceSpring[nodsp1];
   bf[[{6 * n - 5, 6 * n - 4, 6 * n - 3}, {6 * n - 5, 6 * n - 4, 6 * n - 3}]] += bsp];
  Return[bf]]
```

In[19]:=
```
ApplyConMasstoMasterM[nodmm_, M_] :=
 Module[{mm, ll, n, i, mt, i11, i22, i33, i21, i31, i32}, mm = M;
  ll = Length[nodmm];
  For[i = 1, i ≤ ll, i++, n = nodmm⟦i, 1⟧;
   mt = nodmm⟦i, 2⟧;
   mm⟦{6 n - 5, 6 n - 4, 6 n - 3, 6 n - 2, 6 n - 1, 6 n},
      {6 n - 5, 6 n - 4, 6 n - 3, 6 n - 2, 6 n - 1, 6 n}⟧ += {{mt⟦1⟧, 0, 0, 0, 0, 0},
      {0, mt⟦1⟧, 0, 0, 0, 0}, {0, 0, mt⟦1⟧, 0, 0, 0}, {0, 0, 0, mt⟦2⟧, -mt⟦5⟧, -mt⟦6⟧},
      {0, 0, 0, -mt⟦5⟧, mt⟦3⟧, -mt⟦7⟧}, {0, 0, 0, -mt⟦6⟧, -mt⟦7⟧, mt⟦4⟧}};];
  Return[mm]]

(*Test Data*)
ClearAll[K, f, v1, v2, v4]; Km = Array[K, {18, 18}];
Print["Master Stiffness: ", Km // MatrixForm];
nodtag = {{1, 1, 1, 0, 0, 0}, {0, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
nodval = {{v1, v2, 0, 0, 0, 0}, {0, 0, 0, 0, 0, v4}, {0, 0, 0, 0, 0, 0}};
Kmod = ApplyDofToMaster[nodtag, Km];
Print["Modified Master Stiffness:", Kmod // MatrixForm];
fm = Array[f, {18}]; Print["Master Force Vector:", fm];
fmod = ApplyLoad[nodtag, nodval, Km, fm];
Print["Modified Force Vector:", fmod // MatrixForm];
```

In[20]:=
```
u = LinearSolve[Kmod, fmod];
```

In[21]:=
```
FlatNodePartVector[nv_] := Flatten[nv];
NodePartFlatVector[nfc_, v_] :=
  Module[{i, k, m, n, nv = {}, numnod}, If[Length[nfc] == 0, nv = Partition[v, nfc]];
   If[Length[nfc] > 0, numnod = Length[nfc]; m = 0;
    nv = Table[0, {numnod}];
    For[n = 1, n ≤ numnod, n++, k = nfc[[n]];
     nv[[n]] = Table[v[[m + i]], {i, 1, k}];
     m += k]];
   Return[nv]];
```

In[23]:=
```
(*Test Data*)
pp = {1, 2, 3, 4, 5, 6, 7, 8, 9}
NodePartFlatVector[3, pp]
```

In[24]:=
```
SpaceBeam2IntForce[ncoor_, Em_,
   A_, ue_, ebeam_, bsp_, Iyy_, Izz_, ksy_, ksz_, ν_] :=
  Module[{x1, x2, y1, y2, z1, z2, x21, y21, z21, EA, numer, LL, pe, x0, y0, z0, xm,
    ym, zm, dx, dy, dz, txx, txy, txz, tyx, tyy, tyz, tzx, tzy, tzz,
    zL, L, Te, B, ubar, By, Bz, bsy, bsz, yb, zb, i, j, beamsn, sccd,
    pep, Gm, phiy, phiz}, {{x1, y1, z1}, {x2, y2, z2}} = ncoor;
   {x21, y21, z21} = {x2 - x1, y2 - y1, z2 - z1};
   pe = {};
   EA = A Em;
   LL = x21^2 + y21^2 + z21^2;
   L = √LL;
   {x0, y0, z0} = {xm, ym, zm} = 1/2 {x1 + x2, y1 + y2, z1 + z2};
   If[x1 == x2, {x0, y0, z0} += {1, 0, 0}, {x0, y0, z0} += {0, 0, 1}];
   {dx, dy, dz} = {x0 - xm, y0 - ym, z0 - zm};
   tzx = dz y21 - dy z21;
   tzy = dx z21 - dz x21;
   tzz = dy x21 - dx y21;
   zL = √(tzx^2 + tzy^2 + tzz^2);
   {tzx, tzy, tzz} = {tzx, tzy, tzz}/zL;
   {txx, txy, txz} = {x21, y21, z21}/L;
   tyx = txz tzy - txy tzz;
   tyy = txx tzz - txz tzx;
   tyz = txy tzx - txx tzy;
   Te = {{txx, txy, txz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {tyx, tyy, tyz, 0, 0, 0, 0, 0, 0, 0,
      0, 0}, {tzx, tzy, tzz, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 0, 0, txx, txy, txz,
      0, 0, 0, 0, 0, 0}, {0, 0, 0, tyx, tyy, tyz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, tzx,
      tzy, tzz, 0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, txx, txy, txz, 0, 0, 0},
     {0, 0, 0, 0, 0, 0, tyx, tyy, tyz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, tzx, tzy,
      tzz, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, txx, txy, txz}, {0, 0, 0, 0, 0,
      0, 0, 0, 0, tyx, tyy, tyz}, {0, 0, 0, 0, 0, 0, 0, 0, 0, tzx, tzy, tzz}};
```

```
ubar = Te.ue;
beamsn = Length[ebeam];
sccd = Length[bsp];
Gm = Em / (2 (1 + ν));
phiy = 12 Em Iyy / (Gm A ksy LL);
phiz = 12 Em Izz / (Gm A ksz LL);
For[i = 1, i ≤ beamsn, i++,
 By = 1/LL 4 {{0, 6 ebeam[[i]] / (4 + 4 phiz), 0, 0, 0, -((L (1 + phiz - ebeam[[i]])) / (4 (1 + phiz))) +
      L ebeam[[i]] / (2 (1 + phiz)), 0, (1 - 2 ebeam[[i]]) / (2 (1 + phiz)) - (1 + ebeam[[i]]) / (2 (1 + phiz)), 0, 0,
    0, L ebeam[[i]] / (2 (1 + phiz)) + (L (1 + phiz + ebeam[[i]])) / (4 (1 + phiz))}};
 Bz = 1/LL 4 {{0, 0, 6 ebeam[[i]] / (4 + 4 phiy), 0, -(-((L (1 + phiy - ebeam[[i]])) / (4 (1 + phiy))) +
      L ebeam[[i]] / (2 (1 + phiy))), 0, 0, 0, (1 - 2 ebeam[[i]]) / (2 (1 + phiy)) - (1 + ebeam[[i]]) / (2 (1 + phiy)), 0,
    -(L ebeam[[i]] / (2 (1 + phiy)) + (L (1 + phiy + ebeam[[i]])) / (4 (1 + phiy))), 0}};
 pep = {};
 paf = 1/LL EA (x21 (ue[[7]] - ue[[1]]) + y21 (ue[[8]] - ue[[2]]) + z21 (ue[[9]] - ue[[3]]));
 For[j = 1, j ≤ sccd, j++, {yb, zb} = bsp[[j]];
  bsy = Em zb Bz.ubar;
  bsz = Em yb By.ubar;
  pep = Append[pep,
    N[1/LL EA (x21 (ue[[7]] - ue[[1]]) + y21 (ue[[8]] - ue[[2]]) + z21 (ue[[9]] - ue[[3]])) +
      -A (bsy[[1]] + bsz[[1]])]]];
 pe = Append[pe, pep]];
Return[pe / A]];

(*By=1/L{{0,1/L6 ebeam[[i]],0,0,0,
    3 ebeam[[i]]-1,0,-1/L6 ebeam[[i]],0,0,0,3 ebeam[[i]]+1}};
 Bz=1/L{{0,0,1/L6 ebeam[[i]],0,-3 ebeam[[i]]+1,0,0,0,
    -1/L6 ebeam[[i]],0,-3 ebeam[[i]]-1,0}};*);

(*Test Data*)
Test = {{0, 0, 0}, {10, 5, 0}, {10, 0, 0}, {20, 8, 0}, {20, 0, 0}, {30, 9, 0},
   {30, 0, 0}, {40, 8, 0}, {40, 0, 0}, {50, 5, 0}, {50, 0, 0}, {60, 0, 0}};
SpaceBeam2IntForce[Test[[{2, 8}]], 100, 20, {1, 2, 5, 3, 0, 0, -5, 4, 2, 0, 0, 0},
 {-1, -0.8, -0.2, 0, 0.5, 1}, {{-2, -2}, {2, 2}, {2, -2}, {-2, 2}}]
```

In[25]:=
```
SpaceTrussIntForces[nodcoor_, elenod_,
    elemat_, Aa_, noddis_, ebeam_, bsp_, ksbeam_, elemi_] :=
  Module[{numnod = Length[nodcoor], numele = Length[elenod], e, ni,
    nj, ncoor, Em, A, options, ue, p, Iyy, Izz, ksy, ksz, ν}, p = {};
   pafa = {};
   For[e = 1, e ≤ numele, e++, {ni, nj} = elenod[[e]];
    ncoor = {nodcoor[[ni]], nodcoor[[nj]]};
    ue = Flatten[{noddis[[ni]], noddis[[nj]]}];
    Em = elemat[[e]]; A = Aa[[e]];
    ν = nua[[e]];
    {Iyy, Izz} = elemi[[e]]; {ksy, ksz} = ksbeam[[e]];
    p = Append[p,
      SpaceBeam2IntForce[ncoor, Em, A, ue, ebeam, bsp, Iyy, Izz, ksy, ksz, ν]];
    pafa = Append[pafa, paf]];
   Return[p]];

(*Test Data*)
sqq = SpaceTrussIntForces[{{0, 0, 0}, {5, 4, 0}, {10, 20, 0}},
   {{1, 2}, {2, 3}, {1, 3}}, Table[100, 3], {5, 5, 5},
   {{0, 0, 0, 0, 0, 0}, {1, 2, 0, 0, 0, 0}, {2, 1, 0, 0, 0, 0}},
   {-1, -0.5, 0, 0.5, 1}, {{-2, -2}, {2, 2}, {2, -2}, {-2, 2}}] // MatrixForm
```

In[26]:=
```
SpaceTrussStresses[Aa_, elefor_, elenod_] :=
  Module[{numele = Length[elenod], e, elesig}, elesig = Table[0, {numele}, 2];
   For[e = 1, e ≤ numele, e++, elesig[[e]] = elefor[[e]] / Aa[[e]]];
   Return[elesig]];

(*Test Data*)
SpaceTrussStresses[{5, 5, 5},
 {{171, 146}, {-19, -19}, {40, 35}}, {{1, 2}, {2, 3}, {1, 3}}]
```

In[27]:=
```
SpaceTrussSolution[nodcoor_, elenod_, elemat_, Aa_,
  nodtag_, nodval_, nodsp_, elemi_, ebeam_, bsp_, ksbeam_] :=
 Module[{K, Kmod, f, fmod, u, noddis, nodfor, elefor, elesig,
   He, nodcoorc, noddisc, comb, Title, en, elesigl, bsp2},
  K = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  K = SetPrecision[K, 60];
  Kmod = K;
  If[VectorQ[nodsp], Null, Kmod = ApplySpringToMaster[nodsp, K]];
  Kmod = ApplyDofToMaster[nodtag, Kmod];
  Kmod = SetPrecision[Kmod, 60];
  f = FlatNodePartVector[nodval];
  fmod = ApplyLoad[nodtag, nodval, K, f];
  fmod = SetPrecision[fmod, 60];
  u = LinearSolve[Kmod, fmod];
  u = Chop[u];
  He = 0.5;
  f = Chop[K.u, 1/10.^8];
  nodfor = NodePartFlatVector[6, f];
  noddis = NodePartFlatVector[6, u];
  elesig = Chop[SpaceTrussIntForces[nodcoor,
     elenod, elemat, Aa, noddis, ebeam, bsp, ksbeam, elemi]];
  (*elesig2=SpaceTrussStresses[Aa,elefor,elenod];*)
  comb = Join[nodcoor, noddis, 2];
  Title = {Text["Node Coordinate"], SpanFromLeft,
    SpanFromLeft, Text["Node Displacement"], SpanFromLeft,
    SpanFromLeft, SpanFromLeft, SpanFromLeft, SpanFromLeft};
  comb = Prepend[comb, Title];
  Print["Node Displacement Result:"];
  Print[ScientificForm[Grid[comb, Frame → All, ItemSize → All,
      ItemStyle → Directive[FontSize → 8]], 5] // MatrixForm];
  elesig = Partition[Flatten[elesig], Length[bsp]];
  en = Length[elenod];
  elesigl = Partition[Flatten[Array[ebeam &, en]], 1];
  ala = {};
  ff[x_] := If[x == 1, xx, SpanFromAbove];
  For[xx = 1, xx ≤ en, xx++, ala = Append[ala, Array[ff, Length[ebeam]]]];
  ala = Partition[Flatten[ala], 1];
  elesig = Join[elesigl, elesig, 2];
  elesig = Join[ala, elesig, 2];
  bsp2 = Prepend[Prepend[bsp, "Natural Coord"], "Element No. "];
  elesig = Prepend[elesig, bsp2];
  Print["Element Stress Result:"];
  Print[Grid[elesig, Frame → All, ItemSize → All, Alignment → Center]]
 ];
```

```
(*Test Data*)
ClearAll[nodcoor, elemat, elemi];
nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
numele = Length[elenod];
elemat = Table[210 000, {numele}];
numnod = Length[nodcoor];
Aa = {6280.3, 6280.3, 6280.3};
nodtag = Table[{0, 0, 0, 0, 0, 0}, {numnod}];
nodval = Table[{0, 0, 0, 0, 0, 0}, {numnod}];
elemi = Table[{7 854 000, 7 854 000}, numele];
nodval[[4]] = {100, 0, -100, 0, 0, 0};
nodtag[[1]] = {1, 1, 1, 1, 1, 1};
ebeam = {-1, -0.5, 0, 0.5, 1};
bsp = {{2, 2}, {-2, -2}, {2, -2}, {-2, 2}};
nodsp = {{{0, 0, 0}, {0, 0, 10}, 100, 0, 1}, {{300, 0, 0}, {300, 0, 10}, 100, 0, 4}};
SpaceTrussSolution[nodcoor, elenod,
 elemat, Aa, nodtag, nodval, nodsp, elemi, ebeam, bsp]


(*K=SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi];
Kmod=ApplyDofToMaster[nodtag,K];
f=FlatNodePartVector[nodval];
fmod=ApplyLoad[nodtag,nodval,K,f];
u=LinearSolve[Kmod,fmod];
u=Chop[u];
f=Chop[K.u, 1/10.^8];
nodfor=NodePartFlatVector[6,f];
noddis=NodePartFlatVector[6,u];
elefor=Chop[SpaceTrussIntForces[nodcoor,elenod,elemat,Aa,noddis,He]];
elesig=SpaceTrussStresses[Aa,elefor,elenod];
Print["Node Displacement Result:",noddis//MatrixForm];*)

In[28]:= (*Dynamic Test*)
```

```
SpaceTrussMode[nodcoor_, Aa_, elenod_, eledes_, elemat_, elemi_, nodtag_,
  nodsp_, nodconm_, ksbeam_] := Module[{mm, kk, al, kmod, ww, ei1, ei2, λ},
  mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
  mm = ApplyConMasstoMasterM[nodconm, mm];
  kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  al = Inverse[mm].kk;
  (*For[i=1,i≤6,i++,For[j=1,j≤24,j++,al[[i,j]]=0]];
  For[i=1,i≤6,i++,For[j=1,j≤24,j++,al[[j,i]]=0]];*)
  al // MatrixForm;
  (*ww=Re[Chop[Eigenvalues[al]]];
  ww=Sort[ww];
  Print["Free Eigenvalue:",SlideView[ww,
     AppearanceElements→{ "FirstSlide", "PreviousSlide", "NextSlide",
       "LastSlide", "SlideNumber", "SlideTotal"}]//ScientificForm];*)
  If[VectorQ[nodsp], Null, kk = ApplySpringToMaster[nodsp, kk]];
  kk = ApplyDofToMaster[nodtag, kk];
  mm = ApplyDofToMaster[nodtag, mm];
  al = Inverse[mm].kk;
  ei2 = Abs[Chop[Eigenvalues[al]]];
  rpmode = Abs[Chop[Eigensystem[al]]];
  ei2 = Re[Chop[Eigenvalues[al]]];
  ei2 = Sort[ei2];
  saa = Eigenvalues[al];
  Print["Constrained Eigenvalue:", SlideView[ei2,
     AppearanceElements → { "FirstSlide", "PreviousSlide", "NextSlide",
       "LastSlide", "SlideNumber", "SlideTotal"}] // ScientificForm];
  Print[Select[ei2, Abs[Re[#]] > 1 &, 20] // MatrixForm]]

In[30]:= SpaceTrussBuckling[nodcoor_, Aa_, elenod_, eledes_, elemat_,
  elemi_, nodtag_, nodsp_, nodconm_, ksbeam_] := Module[{kk, kd, eib},
  kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  kd = SpaceTrassMasterDiffStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  If[VectorQ[nodsp], Null, kk = ApplySpringToMaster[nodsp, kk]];
  kk = ApplyDofToMaster[nodtag, kk];
  kd = ApplyDofToMaster[nodtag, kd];
  eib = Eigenvalues[{kk, -kd}];
  eib = Select[Sort[Abs[Chop[eib]]], #1 > 1 &, 1];
  Print["1st Buckling Load Cofficient:"];
  Print[eib[[1]]];
 ]

In[31]:= RotorMode[nodcoor_, Aa_, elenod_, eledes_, elemat_,
  elemi_, nodtag_, nodsppd_, nodconm_, ops_, kkp_, ksbeam_] :=
 Module[{mm, kk, al, kmod, ww, ei1, ei2, λ, kkd, nodsp1, i, kkda,
   cs1, cs2, cs3, cs4, nnn, spt, spt2, spt3, spty, msp1, msp2,
   rpmode, dn, mmm, mma1, mma2, cm1, cm2, kks, kke, rr, rpp},
```

```
mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
mm = ApplyConMasstoMasterM[nodconm, mm];
kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
Print[Grid[{{Style["Rotor Undamped Speed Map&Mode Shape",
      FontSize → 15, FontWeight → Bold]}}, Frame → All]];
Print["Rotor Geometry:"];
rr = Sqrt[(Aa) / Pi];
rpp = {};
Do[rpp = Append[rpp, Rectangle[{nodcoor[[i, 1]], -rr[[i]]},
     {nodcoor[[i + 1, 1]], rr[[i]]}]], {i, Length[nodcoor] - 1}];
rpp = Prepend[rpp, White];
rpp = Prepend[rpp, EdgeForm[Thin]];
Print[Graphics[rpp, Frame → True]];
Print[];
Print[];
Print[];
kks = 1000;
kke = 10 000 000;
mm = ApplyDofToMaster[nodtag, mm]; ei2 = {};
kkda = {};
rpmode = {};
Do[nodsp1 = nodsppd;
 If[VectorQ[nodsp1], Null,
  For[i = 1, i ≤ Length[nodsp1], i++, Quiet[nodsp1[[i, 3]] = kkp[[j]]];
   Quiet[nodsp1[[i, 4]] = 10]]];
 If[VectorQ[nodsp1], Null, kkd = ApplySpringToMaster[nodsp1, kk]];
 kkd = ApplyDofToMaster[nodtag, kkd];
 al = Inverse[mm].kkd;
 rpmode = Append[rpmode, Eigensystem[al]], {j, 2}];
Do[nodsp1 = nodsppd;
 kkda = Append[kkda, kki];
 If[VectorQ[nodsp1], Null,
  For[i = 1, i ≤ Length[nodsp1], i++, Quiet[nodsp1[[i, 3]] = kki];
   Quiet[nodsp1[[i, 4]] = 10]]];
 If[VectorQ[nodsp1], Null, kkd = ApplySpringToMaster[nodsp1, kk]];
 kkd = ApplyDofToMaster[nodtag, kkd];
 al = Inverse[mm].kkd;
 ei2 = Append[ei2, Select[Abs[Chop[Eigenvalues[al]]], #1 != 1 &]],
 {kki, kks, kke, (kke - kks)/40}];
ei2 = Sqrt[ei2]/(2 π) * 60;
nnn = Length[ei2[[1]]];
cs1 = Join[Partition[kkda, 1], Partition[ei2[[All, nnn]], 1], 2];
cs2 = Join[Partition[kkda, 1], Partition[ei2[[All, nnn - 1]], 1], 2];
cs3 = Join[Partition[kkda, 1], Partition[ei2[[All, nnn - 2]], 1], 2];
```

```
cs4 = Join[Partition[kkda, 1], Partition[ei2[[All, nnn - 3]], 1], 2];
spt = nodsppd[[1, 3]];
spt[[All, 1]] = spt[[All, 1]];
spty = nodsppd[[2, 3]];
spty[[All, 1]] = spty[[All, 1]];
spt2 = {};
spt3 = {};
Do[spt2 = Append[spt2, Reverse[spt[[i]]]], {i, Length[spt]}];
Do[spt3 = Append[spt3, Reverse[spty[[i]]]], {i, Length[spty]}];
Print[ListLogLogPlot[{cs1, cs2, cs3, cs4, {{kks, ops[[1]]}, {kke, ops[[1]]}},
    {{kks, ops[[2]]}, {kke, ops[[2]]}}, spt2, spt3}, Joined → True,
  PlotRange → All, PlotStyle → {Thickness[0.003], Thickness[0.003],
    Thickness[0.003], Thickness[0.003], Dashed, Dashed, Dashed, Dashed},
  Frame → True, GridLines → Automatic, FrameLabel →
   {"Support Stiffness", "Critical Speed (cpm)"},
  ImageSize → Large, PlotLabel → "Undamped Critical Speed Map"]];
msp1 = rpmode[[1]];
msp2 = rpmode[[2]];
dn = Position[msp1[[1]], 1.];
msp1[[1]] = Delete[msp1[[1]], dn];
msp1[[2]] = Delete[msp1[[2]], dn];
msp2[[1]] = Delete[msp2[[1]], dn];
msp2[[2]] = Delete[msp2[[2]], dn];
Do[msp1[[2, i]] = msp1[[2, i]] / Max[Abs[msp1[[2, i]]]],
 {i, Length[msp1[[2]]]}];
Do[msp2[[2, i]] = msp2[[2, i]] / Max[Abs[msp2[[2, i]]]],
 {i, Length[msp2[[2]]]}];
mma1 = {};
Do[mmm = {};
 Do[mmm = Append[mmm, msp1[[2, Length[msp1[[2]]] - j]][[6 * i - 3]]],
  {i, Length[nodcoor]}];
 mma1 = Append[mma1, Transpose[Join[{nodcoor[[All, 1]]}, {mmm}]]], {j, 0, 3}];
mma2 = {};
Do[mmm = {};
 Do[mmm = Append[mmm, msp2[[2, Length[msp2[[2]]] - j]][[6 * i - 3]]],
  {i, Length[nodcoor]}];
 mma2 = Append[mma2, Transpose[Join[{nodcoor[[All, 1]]}, {mmm}]]], {j, 0, 3}];
cm1 = Sqrt[Reverse[Take[msp1[[1]], -4]]] * 60 / (2 * Pi);
cm2 = Sqrt[Reverse[Take[msp2[[1]], -4]]] * 60 / (2 * Pi);
Print[];
Print[];
Print[ListLinePlot[
  {mma1[[1]], mma1[[2]], mma1[[3]], mma1[[4]]}, InterpolationOrder → 2,
  Mesh → Full, PlotMarkers → {"1", "2", "3", "4"}, PlotStyle → Thickness[0.003],
  PlotLabel → Grid[{{"Mode Shape Plot", SpanFromLeft},
     {"Bearing Stiffness:", ScientificForm[N[kkp[[1]]]]},
```

```
        {"Critical Speed(cpm)", SpanFromLeft}, {"Mode1:", cm1[[1]]}, {"Mode2:",
         cm1[[2]]}, {"Mode3:", cm1[[3]]}, {"Mode4:", cm1[[4]]}}, Frame → All],
     Frame → True, FrameLabel → {"Length", "Relative Displacement"},
     LabelStyle → Directive[Blue, FontSize → 8],
     ImageSize → Large, PlotRange → {-1.2, 1.2}]];
  Print[];
  Print[];
  Print[ListLinePlot[
     {mma2[[1]], mma2[[2]], mma2[[3]], mma2[[4]]}, InterpolationOrder → 2,
     Mesh → Full, PlotMarkers → {"1", "2", "3", "4"}, PlotStyle → Thickness[0.003],
     PlotLabel → Grid[{{"Mode Shape Plot", SpanFromLeft},
        {"Bearing Stiffness:", ScientificForm[N[kkp[[2]]]]},
        {"Critical Speed(cpm)", SpanFromLeft}, {"Mode1:", cm2[[1]]}, {"Mode2:",
         cm2[[2]]}, {"Mode3:", cm2[[3]]}, {"Mode4:", cm2[[4]]}}, Frame → All],
     Frame → True, FrameLabel → {"Length", "Relative Displacement"},
     LabelStyle → Directive[Blue, FontSize → 8],
     ImageSize → Large, PlotRange → {-1.2, 1.2}]];
 ]
nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
Aa = {628.3, 1200, 628.3};
elenod = {{1, 2}, {2, 3}, {3, 4}};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemat = {210 000, 210 000, 210 000};
elemi = {{7854, 7854}, {12 000, 12 000}, {7854, 7854}} * 100;
nodconm = {{3, {0, 0, 0, 0, 0, 0, 0}}};
nodtag =
  {{1, 1, 0, 1, 0, 1}, {1, 1, 0, 1, 0, 1}, {1, 1, 0, 1, 0, 1}, {1, 1, 0, 1, 0, 1}};
kkp = {100, 200 000};
ops = {8500, 12 000};
nodsp1 = {{{0, 0, 0}, {0, 0, 10}, {{1, 0}, {100, 0}}, 0, 1},
   {{300, 0, 0}, {300, 0, 10}, {{1, 0}, {100, 0}}, 0, 4}};
nodsppd = {{{0, 0, 0}, {0, 0, 10}, {{20, 100 000}, {6000, 150 000}, {9000, 190 000}},
    {{1, 10}, {100, 10}}, 1},
   {{0, 0, 0}, {0, 10, 0}, {{20, 200 000}, {6000, 250 000}, {9000, 290 000}}, 10, 1},
   {{300, 0, 0}, {300, 0, 10}, {{20, 100 000}, {6000, 150 000}, {9000, 190 000}},
    10, 4}, {{300, 0, 0}, {300, 10, 0},
    {{20, 200 000}, {6000, 250 000}, {9000, 290 000}}, 10, 4}};
(*SpaceTrussMode[nodcoor,Aa,elenod,eledes,elemat,elemi,
 nodtag,nodsp,nodconm]*)
RotorMode[nodcoor, Aa, elenod, eledes, elemat,
 elemi, nodtag, nodsp1, nodconm, ops, kkp]
```

In[32]:=
```
SpaceTrussModeC[nodcoor_, Aa_, elenod_, eledes_,
  elemat_, elemi_, nodtag_, nodsp_, nodconm_, ksbeam_] :=
 Module[{mm, kk, al, kmod, ww, ei1, ei2, λ, cla, nnn, bbb, bb},
  mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
  mm = ApplyConMasstoMasterM[nodconm, mm];
  kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  nnn = 6 * Length[nodcoor];
  bb = Table[0, nnn, nnn];
  If[VectorQ[nodsp], Null, kk = ApplySpringToMaster[nodsp, kk]];
  kk = ApplyDofToMaster[nodtag, kk];
  mm = ApplyDofToMaster[nodtag, mm];
  If[VectorQ[nodsp], Null, bbb = ApplySpringToMasterB[nodsp, bb]];
  cla = Join[{mm}, {bbb}];
  cla = Join[cla, {kk}];
  ei2 = PolynomialEigenvalues[cla];
  ei2 = Sort[ei2, Abs[Im[#1]] < Abs[Im[#2]] &];
  Print["Constrained Complex Eigenvalue:",
   SlideView[ei2, AppearanceElements → { "FirstSlide", "PreviousSlide",
      "NextSlide", "LastSlide", "SlideNumber", "SlideTotal"}]];
  Print[Select[ei2, Abs[Im[#]] > 1 &, 20] // MatrixForm];]

(*QuadEigenSolver*)

PolynomialEigenvalues[matCof : {__?MatrixQ}] :=
 Module[{p = Length[matCof] - 1, n = Length[First[matCof]]},
   Eigenvalues[{ArrayFlatten[Prepend[NestList[RotateRight,
        PadRight[{IdentityMatrix[n]}, p], p - 2], -Rest[matCof]]],
     SparseArray[{Band[{1, 1}] → First[matCof], {k_, k_} → 1}, {n p, n p}]}]] /;
  Precision[matCof] < Infinity && SameQ @@ (Dimensions /@ matCof)

PolynomialEigenvectors[matCof : {__?MatrixQ}] :=
 Module[{p = Length[matCof] - 1, n = Length[First[matCof]]},
   Map[Take[#, n] &, Eigenvectors[{ArrayFlatten[Prepend[NestList[RotateRight,
        PadRight[{IdentityMatrix[n]}, p], p - 2], -Rest[matCof]]],
     SparseArray[{Band[{1, 1}] → First[matCof], {k_, k_} → 1}, {n p, n p}]}]]] /;
  Precision[matCof] < Infinity && SameQ @@ (Dimensions /@ matCof)

PolynomialEigensystem[matCof : {__?MatrixQ}] :=
 Module[{p = Length[matCof] - 1, n = Length[First[matCof]]},
   MapAt[Map[Take[#, n] &, #] &, Eigensystem[{ArrayFlatten[Prepend[NestList[
        RotateRight, PadRight[{IdentityMatrix[n]}, p], p - 2], -Rest[matCof]]],
     SparseArray[{Band[{1, 1}] → First[matCof], {k_, k_} → 1}, {n p, n p}]}],
    2]] /; Precision[matCof] < Infinity && SameQ @@ (Dimensions /@ matCof)

(*QuadEigenSolver*)
```

In[36]:=
```
SpaceSpring[ncoor_] := Module[
  {x1, x2, y1, y2, z1, z2, x21, y21, z21, EA, numer, L, LL, LLL, Ke, kkk, bbb, Be},
  {{x1, y1, z1}, {x2, y2, z2}} = {ncoor[[1]], ncoor[[2]]};
  kkk = ncoor[[3]];
  bbb = ncoor[[4]];
  {x21, y21, z21} = {x2 - x1, y2 - y1, z2 - z1};
  LL = x21^2 + y21^2 + z21^2; L = Sqrt[LL];
  {x21, y21, z21, EA, LL, L} = N[{x21, y21, z21, EA, LL, L}];
  Ke = (kkk) / LL * {{x21 * x21, x21 * y21, x21 * z21},
     {y21 * x21, y21 * y21, y21 * z21}, {z21 * x21, z21 * y21, z21 * z21}};
  Be = (bbb) / LL * {{x21 * x21, x21 * y21, x21 * z21},
     {y21 * x21, y21 * y21, y21 * z21}, {z21 * x21, z21 * y21, z21 * z21}};
  Return[{Ke, Be}];]
```

In[37]:=
```
RotorGyroscopic[nodcoor_, elenod_, Aa_, eledes_, elemi_] :=
 Module[{numele = Length[elenod], numnod = Length[nodcoor], M, e, ni,
   nj, eftab, Ke, ncoor, Em, A, Iyy, Izz, q, des, x1, x2, y1, y2, z1, z2,
   ll, l, dd, Ip, jp, eftab2, rgm}, rgm = Table[0, {6 numnod}, {6 numnod}];
  For[e = 1, e ≤ numele, e++, {ni, nj} = elenod[[e]];
   eftab = {6 ni - 5, 6 ni - 4, 6 ni - 3, 6 ni - 2, 6 ni - 1, 6 ni};
   eftab2 = {6 nj - 5, 6 nj - 4, 6 nj - 3, 6 nj - 2, 6 nj - 1, 6 nj};
   ncoor = nodcoor[[elenod[[e]]]];
   des = eledes[[e]];
   {{x1, y1, z1}, {x2, y2, z2}} = ncoor;
   ll = (x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2;
   l = √ll ;
   {Iyy, Izz} = elemi[[e]];
   Ip = Iyy + Izz;
   jp = des Ip l;
   gmn = Table[0, 6, 6];
   gmn[[5, 6]] = jp/2;
   gmn[[6, 5]] = -jp/2;
   rgm[[eftab, eftab]] += gmn;
   rgm[[eftab2, eftab2]] += gmn;];
  Return[rgm];]
```

In[38]:=
```
ApplyConMasstoRGM[nodmm_, rgm_] :=
 Module[{mm, ll, n, i, mt, i11, i22, i33, i21, i31, i32}, mm = rgm;
  ll = Length[nodmm];
  For[i = 1, i ≤ ll, i++, n = nodmm[[i, 1]];
   mt = nodmm[[i, 2]];
   mm[[{6 n - 5, 6 n - 4, 6 n - 3, 6 n - 2, 6 n - 1, 6 n}, {6 n - 5, 6 n - 4, 6 n - 3, 6 n - 2,
      6 n - 1, 6 n}]] += {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
     {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, mt[[4]]}, {0, 0, 0, 0, -mt[[4]], 0}};];
  Return[mm]]
```

```
(*Test*)


nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
aa = {628.3, 628.3, 628.3};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemi = {{7854, 7854}, {7854, 7854}, {7854, 7854}};
elemat = {210 000, 210 000, 210 000};
nodtag =
  {{1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
nodspp = {{{300, 0, 0}, {300, 0, 10}, 100, 10, 4}};
conmall = {{3, {100, 100, 100, 100, 100, 100, 100}}};
RotorModeC[nodcoor, aa, elenod,
  eledes, elemat, elemi, nodtag, nodspp, conmall, ksbeam];


In[39]:= krotor[damp_, kr_] := Module[{ti, kcv, kcgr, n, i, ta, eft},
  ti = {{0, 0, 0, 0, 0, 0}, {0, 0, 0.5`, 0, 0, 0}, {0, -0.5`, 0, 0, 0, 0},
    {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0.5`}, {0, 0, 0, 0, -0.5`, 0}};
  n = Length[damp];
  ta = Table[0, n, n];
  For[i = 1, i ≤ n/6, i++, eft = {6 i - 5, 6 i - 4, 6 i - 3, 6 i - 2, 6 i - 1, 6 i};
   ta[[eft, eft]] += ti;];
  kcv = damp.ta + ta.damp;
  kcgr = kr.ta + ta.kr;
  Return[{kcv, kcgr}]]
```

In[49]:=
```
RotorModeC[nodcoor_, Aa_, elenod_, eledes_, elemat_, elemi_,
  nodtag_, nodsp_, nodconm_, ksbeam_] := Module[{mm, kk, al, kmod, ww,
   ei1, ei2, λ, cla, nnn, bbb, bb, rgm, kcv, kcgr, ei2r, ei2i, ei2d, np},
  mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
  np = Length[mm];
  mm = ApplyConMasstoMasterM[nodconm, mm];
  kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  mm = SetPrecision[mm, IntegerPart[1.0 np]];
  kk = SetPrecision[kk, IntegerPart[1.0 np]];
  nnn = 6 Length[nodcoor];
  bb = Table[0, nnn, nnn];
  Print[Grid[{{Style["Rotor Critical Speed&Log Decreasement",
        FontSize → 15, FontWeight → Bold]}}, Frame → All]];
  If[VectorQ[nodsp], Null, kk = ApplySpringToMaster[nodsp, kk]];
  If[VectorQ[nodsp], Null, bbb = ApplySpringToMasterB[nodsp, bb]];
  rgm = RotorGyroscopic[nodcoor, elenod, Aa, eledes, elemi];
  rgm = ApplyConMasstoRGM[nodconm, rgm];
  {kcv, kcgr} = krotor[bbb, kk];
  kk = ApplyDofToMaster[nodtag, kk];
  mm = ApplyDofToMaster[nodtag, mm];
  mm = SetPrecision[mm, IntegerPart[1.0 np]];
  kk = SetPrecision[kk, IntegerPart[1.0 np]];
  rgm = SetPrecision[rgm, IntegerPart[1.0 np]];
  bbb = SetPrecision[bbb, IntegerPart[1.0 np]];
  kcv = SetPrecision[kcv, IntegerPart[1.0 np]];
  cla = Join[{mm - rgm i}, {bbb - kcv i}];
  cla = Join[cla, {kk}];
  ei2 = PolynomialEigenvalues[cla];
  ei2 = saa = Chop[N[ei2, 8]];
  ei2 = Sort[ei2, Abs[Im[#1]] < Abs[Im[#2]] &];
  ei2 = Select[ei2, Im[#1] > 1 &, 4];
  ei2r = Re[ei2];
  ei2i = Im[ei2];
  ei2d = -2 π ei2r / Abs[ei2i];
  Transpose[{ei2i, ei2d}];
  Print["Log Decrement for the First Four Critical Speed:"];
  Print[Grid[Prepend[Transpose[{60 ei2i / (2 π), ei2d}],
     {"Critical Speed(cpm)", "Log Decrement"}], Frame → All, Alignment → Left]]]


SpaceTrussDFR[nodcoor_, elenod_, elemat_, Aa_, nodtag_,
  nodval_, nodsp_, elemi_, ww_, nodconm_, eledes_, ksbeam_] :=
 Module[{K, Kmod, f, fmod, u, noddis, nodfor, elefor, elesig, He,
   nodcoorc, noddisc, comb, Title, mm, nnn, kk, bbb, bb, sft, comb1, comb2,
```

```
 ww0, wwe, wwp, noddisall, Kmodl, wwn, wwall, i, fmodrd, nodsp1},
K = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
noddisall = {};
wwall = {};
{ww0, wwe, wwp} = ww * 2 * Pi;
mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
mm = ApplyConMasstoMasterM[nodconm, mm];
nnn = 6 * Length[nodcoor];
mm = ApplyDofToMaster[nodtag, mm];
f = FlatNodePartVector[nodval];
fmod = ApplyLoad[nodtag, nodval, K, f];
For[wwn = ww0, wwn ≤ wwe, wwn += wwp,
 nodsp1 = nodsp;
 Kmod = K;
 bb = Table[0, nnn, nnn];
 If[VectorQ[nodsp1], Null,
  For[i = 1, i ≤ Length[nodsp1], i++, Quiet[nodsp1[[i, 3]] =
     Interpolation[nodsp1[[i, 3]], InterpolationOrder → 1][wwn * 60 / (2 * Pi)]];
   Quiet[nodsp1[[i, 4]] = Interpolation[nodsp1[[i, 4]],
       InterpolationOrder → 1][wwn * 60 / (2 * Pi)]]]];
 If[VectorQ[nodsp1], Null, bb = ApplySpringToMasterB[nodsp1, bb]];
 If[VectorQ[nodsp1], Null, Kmod = ApplySpringToMaster[nodsp1, K]];
 Kmod = ApplyDofToMaster[nodtag, Kmod];
 Kmodl = (-wwn^2) * mm + wwn * bb * I + Kmod;
 fmodrd = fmod;
 (*Print[wwn/(2*Pi)];*)
 (*For[i=1,i≤Length[nodum],i++,If[nodum[[i,1]]≠0,fmodrd[[6*i-4]]+=
     1*(-1*nodum[[i,1]]*nodum[[i,2]]*wwn^2*Sin[nodum[[i,3]]]+
        I*nodum[[i,1]]*nodum[[i,2]]*wwn^2*Cos[nodum[[i,3]]]);
    fmodrd[[6*i-3]]+=1*(-1*nodum[[i,1]]*nodum[[i,2]]*wwn^2*
         Cos[nodum[[i,3]]]-I*nodum[[i,1]]*nodum[[i,2]]*
         wwn^2*Sin[nodum[[i,3]]]),Null]];*)
 u = LinearSolve[Kmodl, fmodrd]; (*u=Chop[u];*)
 He = 0.5;
 f = Chop[K.u, 1/10.^8];
 nodfor = NodePartFlatVector[6, f];
 noddis = NodePartFlatVector[6, u];
 noddisall = Append[noddisall, noddis];
 sft = Table[SpanFromAbove, Length[nodcoor], 3];
 comb1 = Join[nodcoor, Re[noddis], 2];
 comb2 = Join[sft, Im[noddis], 2];
 comb = Riffle[comb1, comb2];
 Title = {Text["Node Coordinate"], SpanFromLeft,
   SpanFromLeft, Text["Node Displacement(Real/Imag)"], SpanFromLeft,
   SpanFromLeft, SpanFromLeft, SpanFromLeft, SpanFromLeft};
```

```
    comb = Prepend[comb, Title];
    (*Print["Node Displacement Result:",ScientificForm[Grid[comb,Frame→All,
          ItemSize→All,ItemStyle→Directive[FontSize→8]],5]//MatrixForm];*)
    wwall = Append[wwall, wwn / (2 * Pi)]];
   (*elefor=Chop[SpaceTrussIntForces[nodcoor,elenod,elemat,Aa,noddis,He]];
   elesig=SpaceTrussStresses[Aa,elefor,elenod];*)
   noddisall = Flatten[noddisall];
   noddisall = Partition[noddisall, nnn];
   Print["Input Plot Freedom(eg.Node6,Ux->6*6-5=31):"];
   Print[InputField[Dynamic[DFRNode], FieldSize → Tiny]];
   DFRNode = 1;
   Print[Manipulate[STDFRPlot[wwall, noddisall, DFRNode], Text["Bode Plot"]]];
  ]
```

In[42]:=

```
nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
aa = {628.3, 628.3, 628.3};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemi = {{7854, 7854}, {7854, 7854}, {7854, 7854}};
elemat = {210 000, 210 000, 210 000};
nodtag =
  {{1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {1, 1, 0, 0, 0, 0}};
nodspp = {{{300, 0, 0}, {300, 0, 10}, {{0, 100}, {100, 100}},
    {{0, 10}, {100, 10}}, 4}};
conmall = {{3, {100, 100, 100, 100, 100, 100, 100}}};
nodval =
  {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 111, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
nodum = {{0, 0, 0}, {0, 0, 0}, {0, 0, 0}, {0, 0, 0}};
ww = {0, 200, 20};
{std1, std2} = SpaceTrussDFR[nodcoor, elenod,
   elemat, aa, nodtag, nodval, nodspp, elemi, ww, conmall, eledes];


RotorDFR[nodcoor_, elenod_, elemat_, Aa_, nodtag_, nodval_,
  nodsp_, elemi_, ww_, nodconm_, eledes_, nodum_, RDFRNode_, ksbeam_] :=
 Module[{K, Kmod, f, fmod, u, noddis, nodfor, elefor, elesig, He, nodcoorc,
   noddisc, comb, Title, mm, nnn, kk, bbb, bb, sft, comb1, comb2, ww0, wwe,
   wwp, noddisall, Kmodl, wwn, wwall, i, fmodrd, rgm, nodsp1, kcv, kcgr},
  K = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  noddisall = {};
  wwall = {};
  {ww0, wwe, wwp} = ww * 2 * Pi;
  Print[Grid[{{Style["Rotor Direct Frequency Response",
        FontSize → 15, FontWeight → Bold]}}, Frame → All]];
```

```
mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
mm = ApplyConMasstoMasterM[nodconm, mm];
nnn = 6 * Length[nodcoor];
rgm = RotorGyroscopic[nodcoor, elenod, Aa, eledes, elemi];
rgm = ApplyConMasstoRGM[nodconm, rgm];
mm = mm - I * rgm;
mm = ApplyDofToMaster[nodtag, mm];
f = FlatNodePartVector[nodval];
fmod = ApplyLoad[nodtag, nodval, K, f];
For[wwn = ww0, wwn ≤ wwe, wwn += wwp,
 nodsp1 = nodsp;
 Kmod = K;
 bb = Table[0, nnn, nnn];
 If[VectorQ[nodsp1], Null,
  For[i = 1, i ≤ Length[nodsp1], i++, Quiet[nodsp1[[i, 3]] =
     Interpolation[nodsp1[[i, 3]], InterpolationOrder → 1][wwn * 60 / (2 * Pi)]];
   Quiet[nodsp1[[i, 4]] = Interpolation[nodsp1[[i, 4]],
       InterpolationOrder → 1][wwn * 60 / (2 * Pi)]]]];
 If[VectorQ[nodsp1], Null, bb = ApplySpringToMasterB[nodsp1, bb]];
 If[VectorQ[nodsp1], Null, Kmod = ApplySpringToMaster[nodsp1, K]];
 {kcv, kcgr} = krotor[bb, Kmod];
 bb = bb - kcv * I;
 Kmod = ApplyDofToMaster[nodtag, Kmod];
 Kmodl = (-wwn^2) * mm + wwn * bb * I + Kmod;
 fmodrd = fmod;
 For[i = 1, i ≤ Length[nodum], i++, If[nodum[[i, 1]] ≠ 0, fmodrd[[6 * i - 4]] +=
    -1 * (-1 * nodum[[i, 1]] * nodum[[i, 2]] * wwn^2 * Cos[nodum[[i, 3]]] -
       I * nodum[[i, 1]] * nodum[[i, 2]] * wwn^2 * Sin[nodum[[i, 3]]]);
   fmodrd[[6 * i - 3]] += -1 * (-1 * nodum[[i, 1]] * nodum[[i, 2]] *
        wwn^2 * Sin[nodum[[i, 3]]] + I * nodum[[i, 1]] *
        nodum[[i, 2]] * wwn^2 * Cos[nodum[[i, 3]]]), Null]];
 u = Quiet[LinearSolve[Kmodl, fmodrd]]; (*u=Chop[u];*)
 He = 0.5;
 f = Chop[K.u, 1/10.^8];
 nodfor = NodePartFlatVector[6, f];
 noddis = NodePartFlatVector[6, u];
 noddisall = Append[noddisall, noddis];
 sft = Table[SpanFromAbove, Length[nodcoor], 3];
 comb1 = Join[nodcoor, Re[noddis], 2];
 comb2 = Join[sft, Im[noddis], 2];
 comb = Riffle[comb1, comb2];
 Title = {Text["Node Coordinate"], SpanFromLeft,
   SpanFromLeft, Text["Node Displacement(Real/Imag)"], SpanFromLeft,
   SpanFromLeft, SpanFromLeft, SpanFromLeft, SpanFromLeft};
 comb = Prepend[comb, Title];
```

```
    (*Print["Node Displacement Result:",ScientificForm[Grid[comb,Frame→All,
          ItemSize→All,ItemStyle→Directive[FontSize→8]],5]//MatrixForm];*)
     wwall = Append[wwall, wwn / (2 * Pi)]];
    (*elefor=Chop[SpaceTrussIntForces[nodcoor,elenod,elemat,Aa,noddis,He]];
    elesig=SpaceTrussStresses[Aa,elefor,elenod];*)
    noddisall = Flatten[noddisall];
    noddisall = Partition[noddisall, nnn];
    Do[Print[RSTDFRPlot[wwall, noddisall, RDFRNode[[i]]]],
     {i, Length[RDFRNode]}]]
```

In[44]:=
```
STDFRPlot[wwall_, noddisall_, nodnum_] := Module[{ndd, nddm, nddp, nodl, ndph, i},
  ndd = noddisall[[All, nodnum]];
  nddm = Riffle[wwall, Abs[ndd]];
  ndph = Arg[ndd];
  For[i = 1, i ≤ Length[ndph], i++, If[ndph[[i]] < 0, ndph[[i]] += 2 * Pi]];
  nddp = Riffle[wwall, ndph * 360 / (2 * Pi)];
  nddm = Partition[nddm, 2];
  nddp = Partition[nddp, 2];
  GraphicsGrid[{{ListLinePlot[nddp, PlotTheme → "Detailed",
      PlotLabel → Phase, PlotRange → All]}, {ListLinePlot[nddm,
      PlotTheme → "Detailed", PlotLabel → Magnitude, PlotRange → All]}},
   Alignment → Right, Frame → True, ImageSize → Large]
 ]
```

In[45]:=
```
RSTDFRPlot[wwall_, noddisall_, nodnum_] := Module[
  {nddy, nddz, nddmy, nddmz, nddpy, nddpz, nodl, ndphy, ndphz, i, rsmy, rsmz, rsmyp,
   rsmzp, ampy, ampz, sy, symax, symin, sz, szmax, szmin, dist, rsmypn, rsmzpn},
  nddy = noddisall[[All, nodnum * 6 - 4]];
  nddz = noddisall[[All, nodnum * 6 - 3]];
  nddmy = Riffle[wwall * 60, Abs[nddy]];
  ndphy = Arg[nddy];
  nddmz = Riffle[wwall * 60, Abs[nddz]];
  ndphz = Arg[nddz];
  For[i = 1, i ≤ Length[ndphy], i++, If[ndphy[[i]] < 0, ndphy[[i]] += 2 * Pi]];
  For[i = 1, i ≤ Length[ndphz], i++, If[ndphz[[i]] < 0, ndphz[[i]] += 2 * Pi]];
  nddpy = Riffle[wwall * 60, ndphy * 360 / (2 * Pi)];
  nddmy = Partition[nddmy, 2];
  nddpy = Partition[nddpy, 2];
  nddpz = Riffle[wwall * 60, ndphz * 360 / (2 * Pi)];
  nddmz = Partition[nddmz, 2];
  nddpz = Partition[nddpz, 2];
  rsmy = TimeSeriesResample[TimeSeries[nddmy]];
  rsmyp = FindPeaks[rsmy];
  rsmz = TimeSeriesResample[TimeSeries[nddmz]];
  rsmzp = FindPeaks[rsmz];
  ampy = {};
  ampz = {};
```

```
dist[{u_, v_}, {x_, y_}] := Abs[v - y];
rsmypn = Normal[rsmyp];
rsmzpn = Normal[rsmzp];
Do[sy = Nearest[Normal[rsmy], rsmypn[[i]],
    {Infinity, 0.707 * rsmypn[[i, 2]]}, DistanceFunction → dist];
 symax = MaximalBy[sy, First];
 symin = MinimalBy[sy, First];
 ampy = Append[ampy, Quiet[rsmypn[[i, 1]] / (symax[[1, 1]] - symin[[1, 1]])]];,
 {i, Length[rsmypn]}];
Do[sz = Nearest[Normal[rsmz], rsmzpn[[i]],
    {Infinity, 0.707 * rsmzpn[[i, 2]]}, DistanceFunction → dist];
 szmax = MaximalBy[sz, First];
 szmin = MinimalBy[sz, First];
 ampz = Append[ampz, Quiet[rsmzpn[[i, 1]] / (szmax[[1, 1]] - szmin[[1, 1]])]];,
 {i, Length[rsmzpn]}];
ampy = Partition[N[ampy], 1];
ampz = Partition[N[ampz], 1];
Print[Grid[{{"Node Number is:", nodnum}}, Frame → All]];
Print[Grid[
  {{ListLinePlot[{nddpy, nddpz}, PlotTheme → "Detailed", PlotLabel → "Phase",
     PlotRange → All, PlotLegends → Placed[{"Y", "Z"}, Below],
     PlotStyle → {Blue, Dashed}, ImageSize → Large,
     GridLines → {{{ros[[1]], Directive[Dashed, Thick, Blue]},
        {ros[[2]], Directive[Dashed, Thick, Blue]}}, {}},
     Frame → True, FrameLabel → {"Speed(cpm)", "Phase Angle"}]},
   {ListPlot[{nddmy, nddmz, rsmyp, rsmzp}, PlotTheme → "Detailed",
     PlotLabel → "Magnitude", PlotRange → All, PlotLegends →
      Placed[{"Y", "Z"}, Below], PlotStyle → {Blue, Dashed}, ImageSize → Large,
     GridLines → {{{ros[[1]], Directive[Dashed, Thick, Blue]},
        {ros[[2]], Directive[Dashed, Thick, Blue]}}},
     Joined → {True, True, False, False}, Frame → True,
     FrameLabel → {"Speed(cpm)", "Displacement"}]}}, Alignment → Axis]];
Print[Grid[{{"Y Maximize", "Z Maximize"},
   {Grid[Prepend[Join[Normal[rsmyp], ampy, 2], {"cpm", "Max", "Amp.F"}]],
    Grid[Prepend[Join[Normal[rsmzp], ampz, 2],
      {"cpm", "Max", "Amp.F"}]]}, Frame → All]];
Print[Grid[{{"cpm", "Y Disp.", "Z Disp."}, {ros[[1]], rsmy[ros[[1]]], rsmz[
     ros[[1]]]}, {ros[[2]], rsmy[ros[[2]]], rsmz[ros[[2]]]}}, Frame → All]];
]
```

```
nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
aa = {628.3, 628.3, 628.3};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemi = {{7854, 7854}, {7854, 7854}, {7854, 7854}};
elemat = {210 000, 210 000, 210 000};
nodtag =
  {{1, 1, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {1, 1, 0, 0, 0, 0}};
nodspp = {{{300, 0, 0}, {300, 0, 10}, {{0, 100}, {100, 100}},
    {{0, 10}, {100, 10}}, 4}};
conmall = {{3, {100, 100, 100, 100, 100, 100, 100}}};
nodval =
  {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
nodum = {{0, 0, 0}, {0, 0, 0}, {0.1, 0.2, Pi / 6}, {0, 0, 0}};
ww = {10, 12, 1};
ros = {650, 670};
RDFRNode = {2, 4};
ksbeam = {{1, 1}, {1, 1}, {1, 1}};
RotorDFR[nodcoor, elenod, elemat, aa, nodtag, nodval,
  nodspp, elemi, ww, conmall, eledes, nodum, RDFRNode, ksbeam];
```

```
RotorDFR2[nodcoor_, elenod_, elemat_, Aa_, nodtag_, nodval_,
  nodsp_, elemi_, ww_, nodconm_, eledes_, nodum_, ksbeam_] := Module[
  {K, Kmod, f, fmod, u, noddis, nodfor, elefor, elesig, He, nodcoorc, noddisc, comb,
   Title, mm, nnn, kk, k, bbb, bb, sft, comb1, comb2, wwn, noddisall, Kmodl, wwc,
   wwall, i, j, t, fmodrd, rgm, nodsp1, ss, sss, ssf, faa, fy, fz, coo, kcv, kcgr},
  K = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  noddisall = {};
  wwall = {};
  Print[Grid[
    {{Style["Rotor 3D Displacement Plot", FontSize → 15, FontWeight → Bold]}},
    Frame → All]];
  wwc = (ww / 60) * 2 * Pi;
  mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
  mm = ApplyConMasstoMasterM[nodconm, mm];
  nnn = 6 * Length[nodcoor];
  rgm = RotorGyroscopic[nodcoor, elenod, Aa, eledes, elemi];
  rgm = ApplyConMasstoRGM[nodconm, rgm];
  mm = mm - I * rgm;
  mm = ApplyDofToMaster[nodtag, mm];
  f = FlatNodePartVector[nodval];
  fmod = ApplyLoad[nodtag, nodval, K, f];
  For[j = 1, j ≤ Length[wwc], j++,
   wwn = wwc[[j]];
```

```
nodsp1 = nodsp;
Kmod = K;
bb = Table[0, nnn, nnn];
If[VectorQ[nodsp1], Null,
 For[i = 1, i ≤ Length[nodsp1], i++, Quiet[nodsp1[[i, 3]] =
    Interpolation[nodsp1[[i, 3]], InterpolationOrder → 1][wwn * 60 / (2 * Pi)]];
  Quiet[nodsp1[[i, 4]] = Interpolation[nodsp1[[i, 4]],
      InterpolationOrder → 1][wwn * 60 / (2 * Pi)]]]];
If[VectorQ[nodsp1], Null, bb = ApplySpringToMasterB[nodsp1, bb]];
If[VectorQ[nodsp1], Null, Kmod = ApplySpringToMaster[nodsp1, K]];
{kcv, kcgr} = krotor[bb, Kmod];
bb = bb - kcv * I;
Kmod = ApplyDofToMaster[nodtag, Kmod];
Kmodl = (-wwn^2) * mm + wwn * bb * I + Kmod;
fmodrd = fmod;
For[i = 1, i ≤ Length[nodum], i++, If[nodum[[i, 1]] ≠ 0, fmodrd[[6 * i - 4]] +=
   -1 * (-1 * nodum[[i, 1]] * nodum[[i, 2]] * wwn^2 * Cos[nodum[[i, 3]]] -
      I * nodum[[i, 1]] * nodum[[i, 2]] * wwn^2 * Sin[nodum[[i, 3]]]);
  fmodrd[[6 * i - 3]] += -1 * (-1 * nodum[[i, 1]] * nodum[[i, 2]] *
       wwn^2 * Sin[nodum[[i, 3]]] + I * nodum[[i, 1]] *
       nodum[[i, 2]] * wwn^2 * Cos[nodum[[i, 3]]]), Null]];
u = Quiet[LinearSolve[Kmodl, fmodrd]]; (*u=Chop[u];*)
He = 0.5;
f = Chop[K.u, 1/10.^8];
nodfor = NodePartFlatVector[6, f];
noddis = NodePartFlatVector[6, u];
noddisall = Append[noddisall, noddis];
wwall = Append[wwall, wwn / (2 * Pi)]];
(*elefor=Chop[SpaceTrussIntForces[nodcoor,elenod,elemat,Aa,noddis,He]];
elesig=SpaceTrussStresses[Aa,elefor,elenod];*)
noddisall = Flatten[noddisall];
noddisall = Partition[noddisall, nnn];
Do[sss = noddisall[[k]];
 ss = Partition[sss, 6];
 ssf = ss[[All, {2, 3}]];
 faa = {};
 Do[fy = Abs[ssf[[i, 1]]] * Cos[t + Arg[ssf[[i, 1]]]];
  fz = Abs[ssf[[i, 2]]] * Cos[t + Arg[ssf[[i, 2]]]];
  faa = Append[faa, {fy, fz}], {i, Length[ssf]}];
 coo = Partition[nodcoor[[All, 1]], 1];
 faa = Join[coo, faa, 2];
 npp = faa /. t → 0.0001 Pi;
 Print["Speed:", N[wwall[[k]] * 60]];
 Print[ParametricPlot3D[{faa, {Last[nodcoor][[1]] t / (2 Pi),
     Interpolation[npp[[All, {1, 2}]]][Last[nodcoor][[1]] t / (2 Pi)],
```

```
      Interpolation[npp[[All, {1, 3}]]][Last[nodcoor][[1]] t / (2 Pi)]}},
    {t, 0, 2 Pi}, BoxRatios → {3, 1, 1}, ImageSize → Large,
    AxesLabel → {"x", "y", "z"}, PlotStyle → Directive[Dashed, Thickness[0.003]],
    Mesh → {{{0.0001 Pi, PointSize[Medium]}, {0.2 Pi}}},
    PlotRange → All], {k, Length[wwall]}];
]
```

In[47]:=
```
SpaceLinearTransient[nodcoor_, elenod_, elemat_, Aa_, nodtag_,
  nodsp_, elemi_, nodconm_, eledes_, ksbeam_, dt_, in_, pta_] := Module[
  {u, u0, v0, nnn, mm, kk, dd, bb, p, a0, A1, A2, A3, A4, un1, pp, pt, ii, ia, tt},
  nnn = 6 * Length[nodcoor];
  u = {};
  kk = SpaceTrassMasterStiffness[nodcoor, elenod, Aa, elemat, elemi, ksbeam];
  mm = SpaceTrassMasterMass[nodcoor, elenod, Aa, eledes, elemi, elemat, ksbeam];
  mm = ApplyConMasstoMasterM[nodconm, mm];
  mm = ApplyDofToMaster[nodtag, mm];
  bb = Table[0, nnn, nnn];
  If[VectorQ[nodsp], Null, bb = ApplySpringToMasterB[nodsp, bb]];
  If[VectorQ[nodsp], Null, kk = ApplySpringToMaster[nodsp, kk]];
  kk = ApplyDofToMaster[nodtag, kk];
  u0 = Table[0, nnn];
  v0 = Table[0, nnn];
  pt = Flatten[pta, 1];
  Do[If[MatrixQ[pt[[j]]],
    Quiet[pt[[j]] = Interpolation[pt[[j]], InterpolationOrder → 1][ia * dt]]], {j,
    Length[pt]}];
  pp[ii_] := Module[{}, ia = ii; pt; Return[pt]];
  a0 = Inverse[mm].(pp[0] - kk.u0 - bb.v0);
  un1 = u0 - dt * v0 + (dt^2 / 2) * a0; A1 = mm / dt^2 + bb / (2 * dt) + kk / 3;
  A2[ii_] = 1 / 3 * (pp[ii + 1] + pp[ii] + pp[ii - 1]);
  A3 = 2 * mm / dt^2 - kk / 3;
  A4 = -mm / dt^2 + bb / (2 * dt) - kk / 3;
  u = Quiet[Append[u, LinearSolve[A1, A2[0] + A3.u0 + A4.un1]]];
  u = Append[u, LinearSolve[A1, A2[1] + A3.u[[1]] + A4.u0]];
  Do[
   u = Append[u, LinearSolve[A1, A2[i] + A3.u[[i]] + A4.u[[i - 1]]]], {i, 2, in - 1}];
  tt = Table[{dt * (nn - 1)}, {nn, in + 1}];
  u = Prepend[u, Table[0, nnn]];
  u = Join[tt, u, 2];
  Print[N[u] // MatrixForm];
]
```

```
(*Test Data-All Pass*)

nodcoor = {{0, 0, 0}, {100, 0, 0}, {200, 0, 0}, {300, 0, 0}};
elenod = {{1, 2}, {2, 3}, {3, 4}};
aa = {628.3, 628.3, 628.3};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemi = {{7854, 7854}, {7854, 7854}, {7854, 7854}};
elemat = {210 000, 210 000, 210 000};
nodtag =
  {{1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
nodspp = {{{300, 0, 0}, {300, 0, 10}, 100, 10, 4}};
nodconm = {{3, {0, 0, 0, 0, 0, 0, 0}}};
pta = {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}, {0, 0, {{-100, 0}, {0, 0}, {100, 10 000}}, 0, 0, 0}};
ksbeam = {{1, 1}, {1, 1}, {1, 1}};
dt = 0.1;
in = 20;
nua = {0.3, 0.3, 0.3};
(*SpaceLinearTransient[nodcoor,elenod,elemat,aa,
   nodtag,nodspp,elemi,nodconm,eledes,ksbeam,dt,in,pta]--Pass*)


nodcoor = {{0, 0, 0}, {50, 0, 0}, {100, 0, 0}, {150, 0, 0},
   {200, 0, 0}, {300, 0, 0}, {350, 0, 0}, {350, 0, 100}, {380, 120, 180}};
Aa = {5026, 5026, 20 110, 20 110, 5026, 5026, 5026, 5026};
elenod = {{1, 2}, {2, 3}, {3, 4}, {4, 5}, {5, 6}, {6, 7}, {7, 8}, {8, 9}};
eledes = {7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9,
   7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9, 7.85 * 10^-9};
elemat = {210 000, 210 000, 210 000, 210 000, 210 000, 210 000, 210 000, 210 000};
elemi = {{2 000 000, 2 000 000}, {2 000 000, 2 000 000},
   {33 000 000, 33 000 000}, {33 000 000, 33 000 000}, {2 000 000, 2 000 000},
   {2 000 000, 2 000 000}, {2 000 000, 2 000 000}, {2 000 000, 2 000 000}};
nodconm = {};
nodtag = {{1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}};
kkp = {10 000, 200 000};
ops = {85 000, 120 000};
ebeam = {-1, 0, 1};
bsp = {{-5, -5}, {5, 5}, {5, -5}, {-5, 5}};
RDFRNode = {2, 6};
nodum = {{0, 0, 0}, {0, 0, 0}, {0.01, 0.2, Pi / 6},
   {0, 0, 0}, {0, 0, 0}, {0, 0, 0}, {0, 0, 0}, {0, 0, 0}, {0, 0, 0}};
ww = {100, 350, 1};
ww2 = {100, 8400};
ros = {200 * 60, 220 * 60};
```

```
nodval = {{0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0},
   {0, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0}, {1000, -1000, 2000, 0, 0, 0}};
ksbeam = {{1, 1}, {1, 1}, {1, 1}, {1, 1}, {1, 1}, {1, 1}, {1, 1}, {1, 1}};
nodsp = {{{0, 0, 0}, {0, 0, 10}, 10 000, 0, 1},
   {{0, 0, 0}, {0, 10, 0}, 10 000, 0, 1}, {{350, 0, 0}, {350, 10, 0}, 10 000, 0, 7},
   {{350, 0, 0}, {350, 0, 10}, 10 000, 0, 7}};
nodsp1 = {{{50, 0, 0}, {50, 0, 10}, 10 000, 100, 2},
   {{50, 0, 0}, {50, 10, 0}, 10 000, 100, 2}, {{300, 0, 0}, {300, 0, 10},
    10 000, 100, 6}, {{300, 0, 0}, {300, 10, 0}, 10 000, 100, 6}};
nodsppd = {{{50, 0, 0}, {50, 0, 10}, {{0, 10 000}, {1, 10 000}}, {{0, 100}, {1, 100}},
    2}, {{50, 0, 0}, {50, 10, 0}, {{0, 10 000}, {1, 10 000}}, {{0, 100}, {1, 100}}, 2},
   {{300, 0, 0}, {300, 0, 10}, {{0, 10 000}, {1, 10 000}}, {{0, 100}, {1, 100}}, 6},
   {{300, 0, 0}, {300, 10, 0}, {{0, 10 000}, {1, 10 000}}, {{0, 100}, {1, 100}}, 6}};
nua = {0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3, 0.3};
(*SpaceTrussSolution[nodcoor,elenod,elemat,
   Aa,nodtag,nodval,nodsp,elemi,ebeam,bsp,ksbeam]--Pass*)
(*SpaceTrussMode[nodcoor,Aa,elenod,eledes,elemat,
   elemi,nodtag,nodsp,nodconm,ksbeam]--Pass*)
(*RotorMode[nodcoor,Aa,elenod,eledes,elemat,elemi,nodtag,
 nodsppd,nodconm,ops,kkp,ksbeam];
--Pass*)
(*SpaceTrussDFR[nodcoor,elenod,elemat,Aa,nodtag,
   nodval,nodsppd,elemi,ww,nodconm,eledes,ksbeam]--Pass*)
(*RotorDFR[nodcoor,elenod,elemat,Aa,nodtag,nodval,nodsppd,
 elemi,ww,nodconm,eledes,nodum,RDFRNode,ksbeam];
--Pass*)
(*RotorDFR2[nodcoor,elenod,elemat,Aa,nodtag,
 nodval,nodsppd,elemi,ww2,nodconm,eledes,nodum,ksbeam];
--Pass*)
(*RotorModeC[nodcoor,Aa,elenod,eledes,
 elemat,elemi,nodtag,nodsp1,nodconm,ksbeam];
--Pass*)
(*SpaceTrussModeC[nodcoor,Aa,elenod,eledes,
   elemat,elemi,nodtag,nodsp1,nodconm,ksbeam]--Pass*)
```