# AOE 5064 - Project Final Report

## Feedback Responses from Proposal

- Why do you want to achieve a certain mode shape?
  - Refer: **Need of this Optimization Study – Effect of Nodal Point Placement, Page 7**
- Definition of constraint needs some attention. Implementing Nelson's method to drive the optimization will be a good project.
  - Made some changes and addressed the subscript. **Identification of Constraints, Page 6**
- Beware of mode swapping, which may cause discontinuity in derivatives. You may need to implement mode tracking and/or require separation in frequency between second and first and third modes.
  - Mode swapping will be taken care and the normalization conditions will be checked once I will start working on optimization procedure.
- Please provide the equation. (Formulation for derivative of nodal point location)
  - Refer: **Need of this Optimization Study – Effect of Nodal Point Placement, Page 7**
- Why the second mode shape?
  - The only reason to take second mode shape is to deal only with one nodal location. Thus avoiding extensive calculation for eigenvalue sensitivity for every nodal location and thus making the optimization simpler.
  - The results from consideration of second mode can be considered as proof-of-concept study to reduce the generalized force over the rotor blade.
- Is this a continuous function of x? What is the index, i? Shouldn't it be the difference at numerous discrete locations? (Constraint function)
  - Constraint function is not a continuous function of x, it is a function of design variable. With different set of design variable values, a nodal location is obtained, and constraint is the distance between them.
- Feel free to use existing optimization method (*slp* or *sqp* or *Mathworks' fmincon*) to solve the problem.
  - Most probably will be using *fmincon* for the optimization problem.

- Programming Nelson's method (if not already available to you) will be sufficient effort for this project in conjunction with solving the optimization problem.
    - Not already available with me, using lecture notes and lecture videos, programming for Nelson's method subroutine has already been written to compute sensitivity w.r.t design variables.

# Introduction (5-step Problem Formulation)

## Project and Problem Description

In structural dynamics, modal analysis is one of the key aspects for studying mechanical vibrations. As we move ahead with the technology, the need to include vibration analysis in the design process increases. This project will be a study to change the nodal location of the mode shape for a helicopter rotor blade under a given air load criteria. This will be done by varying the lumped masses acting as the design variables to bring the node of a mode shape near to the desired location. This nodal replacement will effect directly on reducing the generalized force acting on the helicopter blade which would result in the minimization of vibration responses.

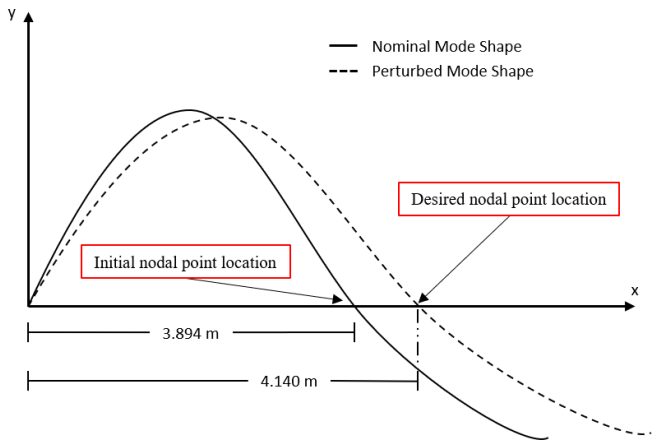


*Figure 1: Nominal vs Perturbed Mode Shape*

In order to perform optimization problem we have to design a convenient structure that can be simulated, considering the rotor blade fixed at one end and having displacement vertically (due to vibration) can be seen as a cantilever beam. Thus defining this structure to 1-D FEM model would be the best way to include all the necessary parameters and include the lumped mass that represent the beam box parameters.

This project considers the beam of a total length of 4.90 m and a total of 10 elements in the design. The dimensions of each beam box element is given in Table 1: Material Properties

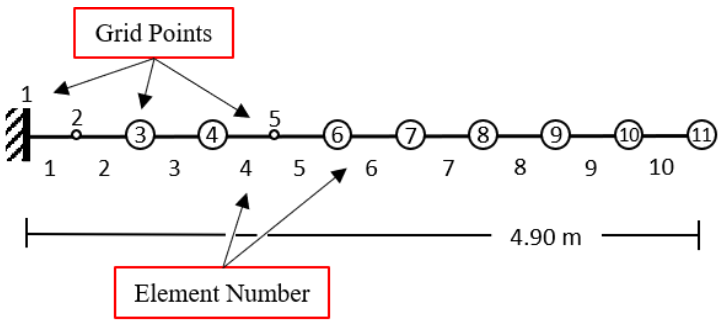


*Figure 2: FEM model for beam box*

## Background – Data and Information Collection

This project is based on nodal point placement by varying the mass distribution. δ is considered as the allowable distance from the desired location, this value is taken as 0.0254 m. This value will be used to set up the constraint function. This nodal location is chosen such a way that the mode shape will be orthogonal to the force distribution. In a study done in [1], results are very favorable as it the generalized force is found to be reduced.

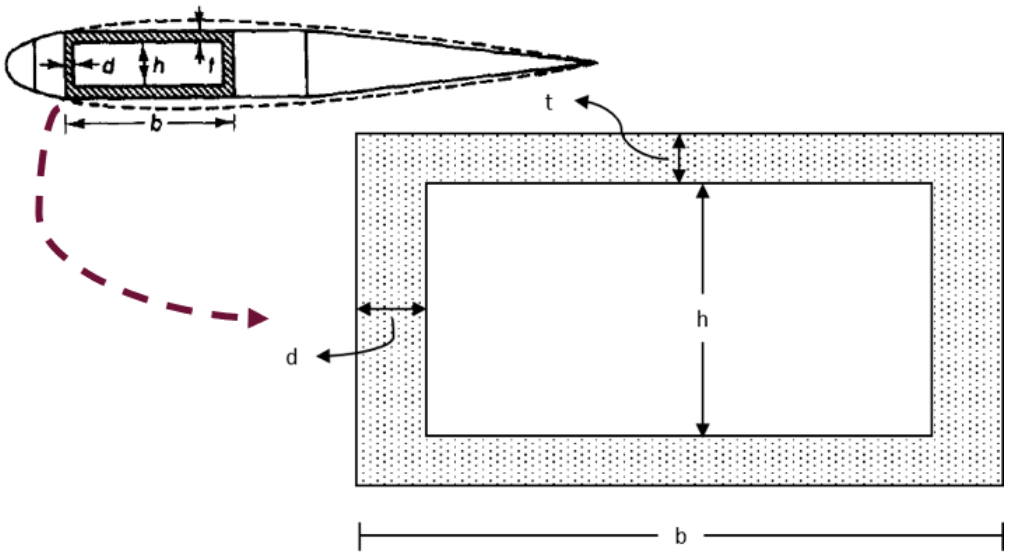


*Figure 3: Beam Box*

*Table 1: Material Properties*

| Element Number | E, psi | $\rho, lb/\text{in}^3$ | b, in | h, in | t, in | d, in |
|---|---|---|---|---|---|---|
| 1 | $0.490 \times 10^7$ | 0.07 | 3.75 | 2.5 | 0.8 | 0.1 |
| 2 - 10 | $0.585 \times 10^7$ | 0.07 | 3.75 | 2.5 | 0.8 | 0.1 |

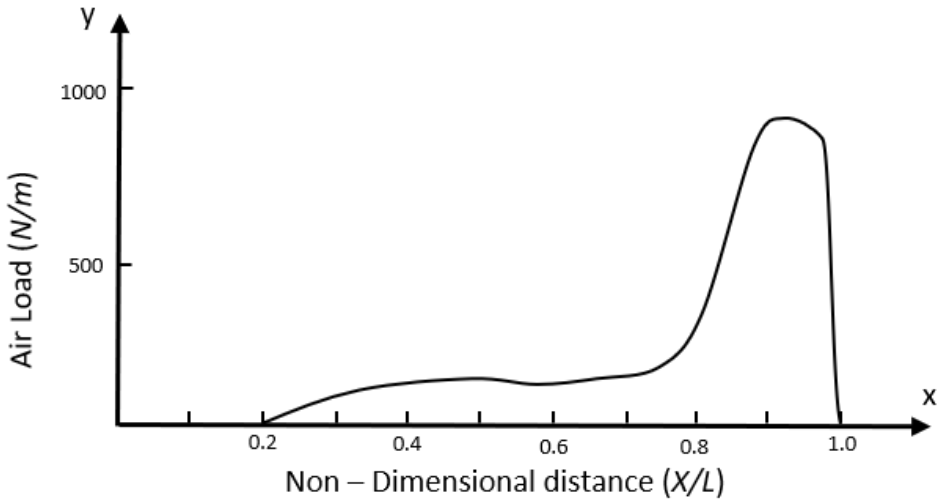


*Figure 4: Air Load Distribution*

$E$, Pa : *Young's Modulus*

$\rho$, kg/m$^3$ : *Weight density*

$b$, m : *Width of box cross-section*

$h$, m : *Height of box cross-section*

$t$, m : *upper/lower wall thickness of box cross-section*

$d$, m : *Sidewall thickness of box cross-section*

$x$, m : *coordinate along 1-D cantilever structure*

$x_{np}$, m : *Location for the nodal point from the origin*

$x_{op}$, m : *Optimum location for the nodal point from the origin*

$M_n$, kg : *Lumped mass at the $n^{th}$ grid point*

$M_i^*$, kg : *Initial Total Mass*

$\delta$, m : *Allowable distance from a desired nodal point location*

$v_i$ : *$i^{th}$ Design Variable*

$f$ : *Objective Function*

$g$ : *Constraint function*

$\omega$ : *Rotational velocity*

$\lambda$ : *Eigenvalue*

$\Phi$ : *Eigenvector*

$T$ : *Transpose of a Matrix (superscript)*

$M$ : *Mass Matrix*

$N$ : *Mode shape functions used to compute displacement in FEM analysis*

## Identification and Definition of Design Variables

The eight lumped masses placed on grid points 3, 4, 6-11 along the length of the beam. These masses will act as our design variables. With the initial mass values are given in Table 2, we can solve for the derivative of nodal point location with respect to the design variable (lumped masses). To approximate this derivative, modal displacement is expanded using Taylor Series and later two different derivatives are computed, first slope of the mode shape and second, modal displacement with respect to the design variable using Nelson's method.

With this sensitivity analysis, we can find which masses are most effective in moving the nodal point, and thus we can take three-four lumped masses to perform our optimization problem.

*Table 2: Initial Mass at grid points*

| Grid Point Number | 3 | 4 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|
| Initial Mass, *lbm* | 3.04 | 1.67 | 6.40 | 7.46 | 10.75 | 5.21 | 6.55 | 6.60 |

### Identification of Criteria to be Optimized

The objective function f is the sum of all the lumped masses, given by Eq 1. The problem is to bring the initial nodal point location near to the desired location by varying the magnitude of lumped masses while minimizing the total lumped mass.

$$f = \sum_{n=1}^{N} M_n \qquad \textit{Eq 1}$$

As discussed above the desired location is the point where the mode shape will be orthogonal to the air load distribution.

### Identification of Constraints

For this optimization problem constraint function is chosen such that the difference in the distance of the perturbed nodal location from the optimum location should be less than the allowable distance, given by Eq 2. For our analysis we have taken allowable distance $\delta = 0.0254\ m$ and $x_{np}$ is the nodal location computed on a set of design iterations (set of values of all the design variables.)

$$g = |x_{op} - x_{np}| \leq \delta \qquad \textit{Eq 2}$$

## Approach to Solving the Structural Design Problem

Approach to solve this optimization problem is divided into three different phases. Phase 1 corresponds of performing modal analysis and calculating the correct mode shape and computing the required nodal location of the second mode. This is done by solving the eigenvalue equation using Finite Element Method and writing the on code from scratch in MATLAB.

Phase 2 involves performing sensitivity analysis, to solve this problem it requires solving three sensitivity, first is nodal point sensitivity. As seen in the derivation given by Eq 5, this requires computing two different derivative, slope and eigenvector sensitivity. Computing slope is easy and a simple code to find this value can be written, but for eigenvector sensitivity, I used a simplified approach called Nelson's method as explained by Eq 6 - Eq 15. While solving we found that there is a need to solve another sensitivity that calls for a another sensitivity computation, eigenvalue sensitivity as given by the derivation of Nelson's Method.

Phase 3 involves solving optimization problem. I have written my own code for slp with trust regions and found the optimized value for lumped masses. Then I used Dr. Canfield's written code 'slp_trust' available on MATLAB File Exchange. This code has two features as it is faster than mine and is more credible than mine. This subroutine also includes commands to plot iteration history that shows convergence of constraint and objective function.

# Pertinent Theory

## Need of this Optimization Study – Effect of Nodal Point Placement

The concept of modal shaping has been proposed as a method to reduce structural vibration. [2] [3]. This project deals with the concept of nodal point placement to perform mode shaping by varying mass distribution of a structure, here helicopter wing represented as 1-D FEM model of cantilever beam with lumped masses representing mass of beam box placed at the grid points, see Figure 2.

In a study done in [1], nodal point placement has the potential for reducing overall response by placing the node at a strategic location of force distribution to reduce the generalized force. Typical candidates for nodal point placement are locations where low response for required, such as pilot or passenger sears, locations of sensitive electronic equipment etc. Effect of having higher response at these locations can be seen as very dangerous scenario, and these higher vibration responses can be minimized by minimizing the generalized force acting on the rotor blade.

It is also significant that once we remove the masses from the grid points the air distribution load along the helicopter blade also changes. A numerical study was performed to investigate the effect of generalized force of placing the nodal location of the second mode shape of a simply supported beam at the centroid of the force distribution. With the centroid at 52% of the length of the beam, the generalized force was computed. When the nodal location was varied between 25% to 75% of the beam length, it was found that the smallest generalized force occurs when the nodal location is between 50% and 55% of the beam length. [4]

Results shown in [4] are obtained for a helicopter blade having a rotational velocity of $\omega = 425\ rpm$, it was found that the magnitude of generalized force acting on the blade was reduced down to $50.67$ N from $110.8$ N which is lesser than half of what it was initially. This was made possible by modifying masses and optimizing such a way to bring the nodal location to $4.140$ m from $3.894$ m. see Figure 1

## Sensitivity Derivative to Nodal Location

The modal deflection normal to the length of a one-dimensional structure is denoted by $w(x, v)$. Nodal location is the location on the structure where the deflection is zero, and for this design problem, both nodal location $(x_{np})$ and the deflection are the function of design variable. When any of the design variable is perturbed the nodal location changes, as seen in Figure 1. To solve this optimization problem we need to find the rate of change of nodal location with respect to the design variable, this information can be obtained by computing the derivative, $\frac{\partial x_{np}}{\partial v}$. This information is called as sensitivity derivative and it is very crucial to optimize the nodal location.

### *Analytical Review*

The formulation of the derivative of the nodal location is based on expanding the perturbed mode in a Taylor series about the nominal nodal point.

$$w(x_{np} + dx_{np}, v + dv) = w(x_{np}, v) + \left.\frac{\partial w}{\partial x}\right|_{x_{np},v} dx_{np} + \left.\frac{\partial w}{\partial v}\right|_{x_{np},v} dv \qquad \textit{Eq 3}$$

As we know displacement at nodal location is zero, this means $w(x_{np}, v) = 0$ and $w(x_{np} + dx_{np}, v + dv) = 0$.

$$\left.\frac{\partial w}{\partial x}\right|_{x_{np},v} dx_{np} + \left.\frac{\partial w}{\partial v}\right|_{x_{np},v} dv = 0 \qquad \textit{Eq 4}$$

$$\frac{dx_{np}}{dv} = -\left[\frac{\frac{\partial w}{\partial v}}{\frac{\partial w}{\partial x}}\right]_{xnp,v} \qquad \textit{Eq 5}$$

Where, $\frac{\partial w}{\partial v}$ = derivative of the mode shape w.r.t design variable and $\frac{\partial w}{\partial x}$ = slope of the mode shape at the nodal location.

## Nelson Method

To solve this optimization problem derivatives for eigenvector with respect to design variable is required to compute the sensitivity of nodal location with respect to each design variable. These derivatives are very important to solve any design optimization, model updating, and many other applications [5].

Nelson gave an alternate and simplified procedure for calculating eigen vector derivatives of arbitrary nth order symmetric or non-symmetric eigensystem. It requires only the left and right eigenvectors and the associated eigenvalue under consideration. In this procedure, the nth order (n-1) rank system of equations is converted to a matrix of rank n requires only 'zeroing' out certain rows and columns entries of the matrix of the eigensystem and gives a matrix the same banded characteristics as the original system [5] . This is done to remove the singularity of the entire eigenvalue equation.

***Analytical Review***

Eigenvalue problem

$$(K - \lambda M)\Phi = 0 \qquad \textit{Eq 6}$$

Differentiating Eq 6

$$\left(K - \lambda_j M\right)\frac{\mathrm{d}\Phi_\mathrm{j}}{\mathrm{d}x_i} = \left(\frac{\mathrm{d}\lambda_j}{\mathrm{d}x_i}\right)M\Phi_\mathrm{j} - \left(\frac{\mathrm{d}K}{\mathrm{d}x_i}\right)\Phi_\mathrm{j} + \lambda_j\left(\frac{\mathrm{d}M}{\mathrm{d}x_i}\right)\Phi_\mathrm{j} \qquad \textit{Eq 7}$$

$$\left(K - \lambda_j M\right)\Phi_\mathrm{j}' = \lambda_j' M\Phi_\mathrm{j} - K'\Phi_\mathrm{j} + \lambda_j M'\Phi_\mathrm{j} \qquad \textit{Eq 8}$$

$$\left(K - \lambda_j M\right)\Phi_\mathrm{j}{}' = -\left[K' - \lambda_j{}'M - \lambda_{jM'}\right]\Phi_\mathrm{j} \qquad \textit{Eq 9}$$

Let, $K - \lambda_j M = F_j$ and $K' - \lambda_j{}'M - \lambda_\mathrm{j}M' = F_j{}'$

$$F_j\Phi_j{}' = -F_j{}'\Phi_\mathrm{j} \qquad \textit{Eq 10}$$

$$\Phi_\mathrm{j}' = -\left[F_\mathrm{j}\right]^{-1}F_j'\Phi_\mathrm{j} \qquad \textit{Eq 11}$$

Now, we know that here F is a singular matrix, and our goal is the find $\Phi'$, therefore the inverse cannot be created until unless we make F, a non-singular matrix, which can be done by removing any row and its corresponding column. Let us call it as $\tilde{F}$, now the Eq 11 becomes,

$$[\widetilde{F_J}]\widetilde{V_J} = -[F_j']\Phi_j \quad \textit{Eq 12}$$

$$\widetilde{V_J} = -[\widetilde{F_J}]^{-1}[F_j']\Phi_j \quad \textit{Eq 13}$$

Therefore, our final solution becomes,

$$\Phi_j' = \widetilde{V_J} + c_j\Phi_j \quad \textit{Eq 14}$$

$$\text{where, } c_j = -\Phi_j^T M\widetilde{V_j} + \frac{\Phi_j^T M'\Phi_j}{2} \quad \textit{Eq 15}$$

This $c_j$ is used to account the removed row column that we did initially.

## Eigenvalue Sensitivity

From Eq 6, we can see that in order to solve for eivenvector derivative we need to have $K', M', \lambda'$. From our understanding of this optimization problem we know that our design variables are the lumped masses and thus $K' = 0$ and $M' = 1$. The remaining part which is $\lambda'$ is remaining to compute and here is the analytical review to compute eigenvalue derivative.

***Analytical Review***

On multiplying Eq 5 with $\Phi_j^T$ we get,

$$\Phi_j^T\left(K - \lambda_j M\right)\Phi_j' = \lambda_j'\Phi_j^T M\Phi_j - \Phi_j^T K'\Phi_j + \lambda_j\Phi_j^T M'\Phi_j \quad \textit{Eq 16}$$

If eigenvectors are normalized with respect to M, that means $\Phi_j^T M\Phi_j = 1$ and we know that $\Phi_j^T\left(K - \lambda_j M\right)\Phi_j' = 0$

$$\lambda_j' = \Phi_j^T K'\Phi_j - \lambda_j\Phi_j^T M'\Phi_j \quad \textit{Eq 17}$$

# Program Design

## High-Level Implementation Review

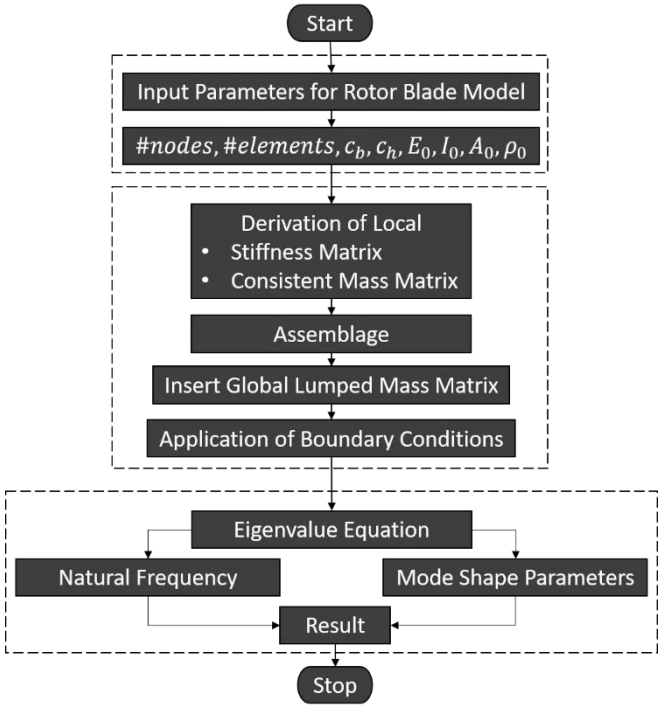


*Figure 5: FEM Overview*

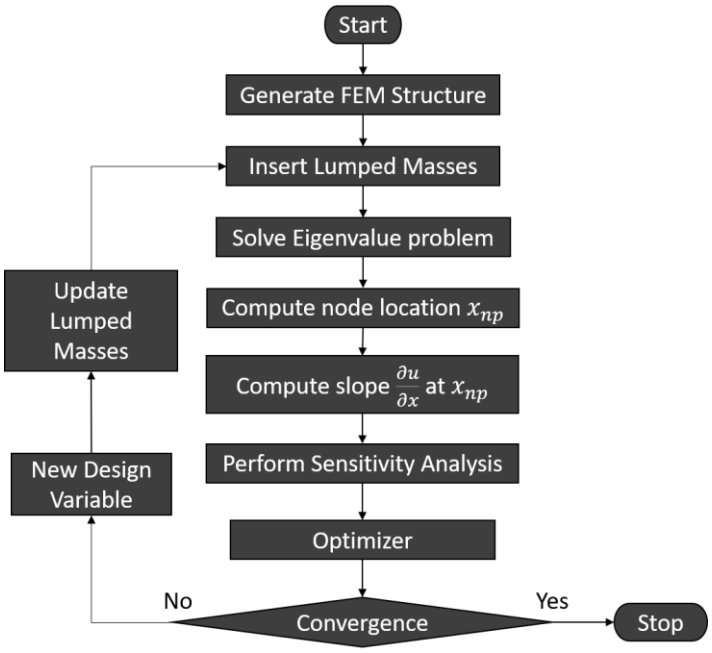


*Figure 6: Optimization Overview*

Starting with FEM model, we need to provide the required data to setup the fem model, details mentioned in the data requirement section. Next comes designing stiffness and mass matrix that will be used to solve eigenvalue problem. Since at every grid point there is there is a lumped mass we have included therefore our third step would be to include global lumped mass matrix and once everything is combined we apply boundary condition, which are deflection and slope equals zero for a cantilever beam. Boundary conditions are applied by curtailing the global stiffness and mass matrix. Once we are done with this we can solve eigen value equation.

The first computation is to generate the structural model and introducing lumped masses to the system. Next step is to perform vibrational analysis, and for that we need to obtain three things, nodal location ($\boldsymbol{x_{np}}$), slope of mode shape at nodal location ($\boldsymbol{\frac{\partial u}{\partial x}\Big|_{x_{np},v}}$), eigenvector derivative at nodal location ($\boldsymbol{\frac{\partial u}{\partial v}\Big|_{x_{np},v}}$). Once we have all these values we compute nodal point sensitivity given by equation Eq 5. Final step is to pass the desired values to the optimizer ( slp_trust – Dr. Robert Canfield ). This particular subroutine require 3 things, script that returns, objective and constraint, script that returns gradients of objective and constraints, initial design variables, and bounds for design variables. If converged stop, otherwise change design variables and update lumped mass matrix and reiterate.

## Data Requirements

Data such as geometrical and material properties are required in order to perform the main FEM and data analysis. Geometrical properties include thickness, length, width of beam box. Material properties such as Youngs Modulus, density of the material.

## Algorithm Pseudo-Codes

***Main Program***

```
define global variables to save values from different subroutines for plotting purpose
define lower and upper bound
define initial design variable
setup options for optimization slp_trust
run slp_trust
plot commands
```

***funcs.m***

```
define delta
define x0
run setup.m
run FEMsolve.m
run eigenAnalysis.m
run modalAnalysis
define objective function
define constraint functions, create row vector
```

***grads.m***

```
run setup.m
run FEMsolve.m
run eigenAnalysis.m
run modalAnalysis
run sensitivity
define gradient of objective function
define gradient of constraint functions, create row vector
```

***setup.m***

```
define shape functions
define number of elements
define material properties
define connectivity matrix
define lumped mass vector
assemble lumped mass vector to a matrix wrt to grid location
```

***FEMsolve.m***

```
define empty global stiffness matrix
define empty global mass matrix
for i = 1 : #elements
        define global stiffness and mass matrix for single element
        call connectivity matrix
        for j = 1 : #dof
                assemble global stiffness and mass matrix
        end
end
combine lumped and consistent mass matrix
```

***modalAnalysis.m***

```
for i = 1 : #elements
        compute deflection (using shape functions and elemental deflection calculated)
        find nodal location (where deflection = 0)
end
plot deflection plot
```

***eigenAnalysis.m***

```
apply boundary condition
```

save stiffness matrix – curtailed
save mass matrix – curtailed
compute eigenvectors and eigen values

***sensitivity.m***

define Mprime
define lambdaPrime
define Fprime
define F
compute max eigenvector location
remove that location row and column from all matrix and row vectors
define V, c, q
compute phiPrime
solve for slope
solve for derivative of deflection wrt to design variable
solve for derivative of nodal location wrt to design variable

# Test Approach

## Verification

Results are obtained using self-designed slp algorithm and results are verified using slp_trust and sqp codes provided with the course by the instructor.

## Validation

The optimized mass values are validated using the research paper [4] this entire project is based on.

# Results and Discussion

## Modal Analysis

Modal analysis is done on the cantilever beam to find nodal location of the second mode shape. It was found lying on the ninth element around in the middle. The exact location is at 154.883 inches.

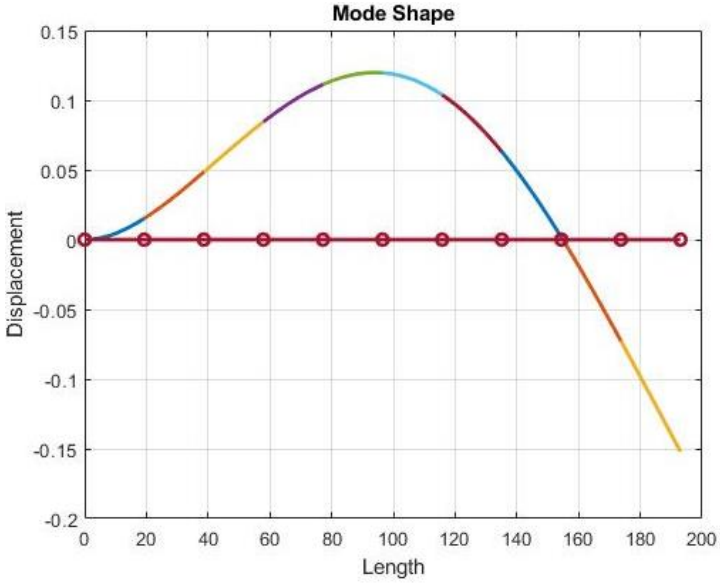


*Figure 7: Second Mode Shape for Cantiliver Beam (Rotor Blade)*

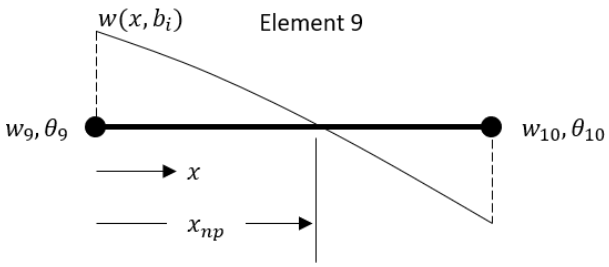


*Figure 8: Element 9*

## Objective and Constraint Function Plots

For both plot shown below, the white region is the feasible region that means, the mass values occurring in that white region will bring the nodal location within the feasible range of 1 in. The objective function is reducing in the direction of origin. Since these plots are plotted with respect two design variables, but still can refer the estimated result from graphical approach of optimization. The total mass should supposed to around 20 – 22 pounds.

| Obj/Cons Plot | Inference |
|---|---|
| Mass 10 - 11 | A good feasible region is found where mass is also found in the sensible values and total mass also makes sense. |
| Mass 9 - 11 | A good feasible region is found where mass is also found in the sensible values and total mass also makes sense. |

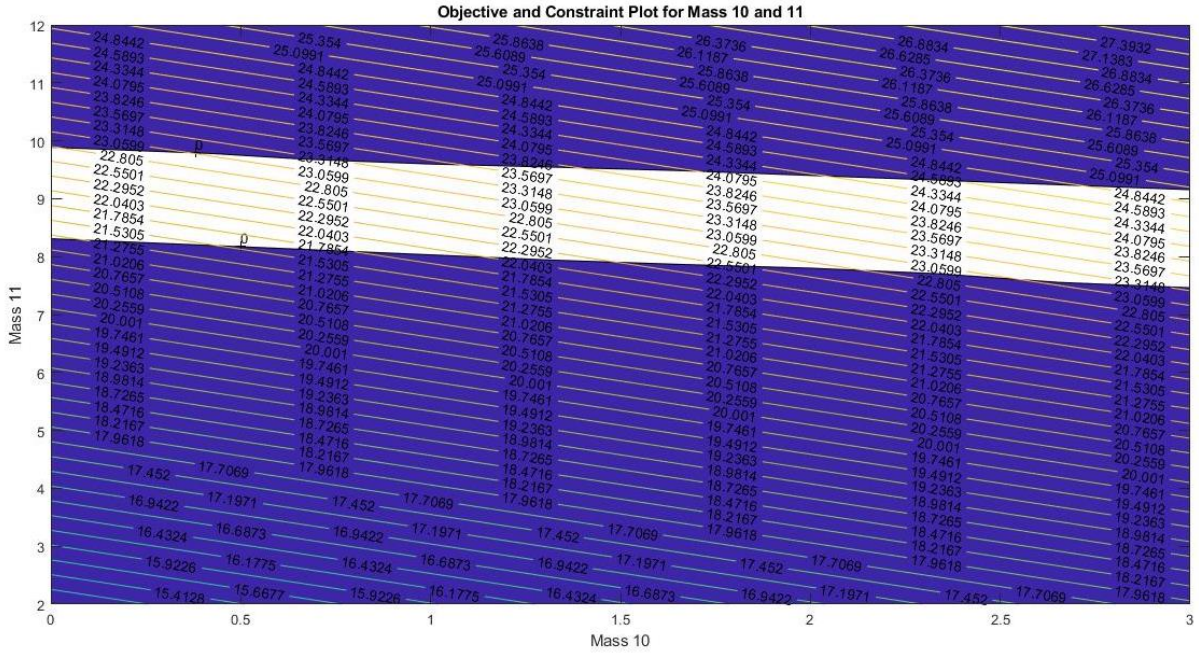


*Figure 9: Objective and Constraint - Mass 10 vs 11*

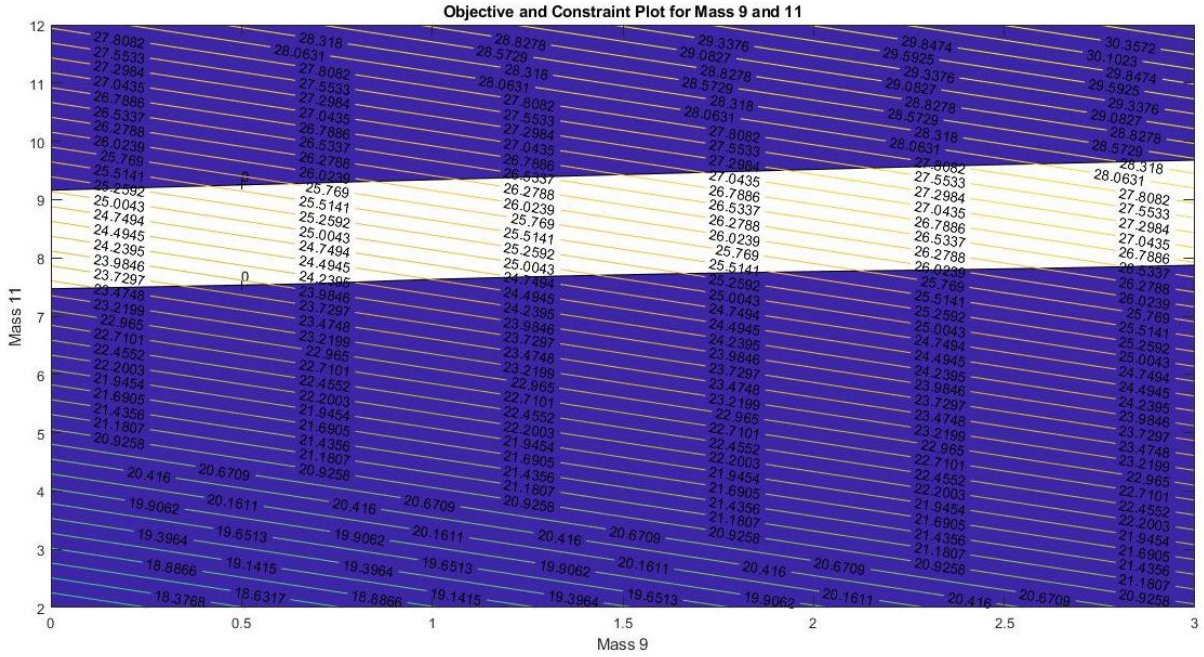


*Figure 10: Objective and Constraint - Mass 11 vs 9*

## Sensitivity Analysis

Few key results can be drawn from the shown sensitivity table. The most important we can observe is where $\frac{\partial x_{np}}{\partial v} > 0$, it means when the mass increases, it will result nodal point moves right, whereas where $\frac{\partial x_{np}}{\partial v} < 0$, it will take nodal point to left. Another thing is that Mass 10 and 11 are the most effective to move the node towards right. Similarly, decrease the masses at 10, 11 or increase the masses at 6, 7 will have the largest effect on moving nodal point towards left. Increasing masses at grid points 6 and 7 have highest impact on moving nodal point to the left.

*Table 3: Sensitivity Results*

| Mass at # Grid Point | Sensitivity Analysis | | | | | |
|---|---|---|---|---|---|---|
| | $\frac{\partial x_{np}}{\partial v}$ | | | $\frac{\partial u}{\partial x}$ | $\frac{\partial u}{\partial v}$ | $\frac{\partial \phi}{\partial v}$ |
| | My Solution | Verification from paper | | | | |
| | Nelson Method | Analytical Method | Finite Difference | Slope | Eigenvector Sensitivity | Eigenvalue Sensitivity |
| 3 | -0.0253 | -0.0278 | -0.0277 | -0.0036 | -0.0001 | -0.0995 |
| 4 | -0.0824 | -0.0881 | -0.088 | | -0.0003 | -0.3015 |
| 6 | -0.2211 | -0.231 | -0.23 | | -0.0008 | -0.6047 |
| 7 | -0.2294 | -0.237 | -0.236 | | -0.0008 | -0.4555 |
| 8 | -0.1636 | -0.166 | -0.165 | | -0.0006 | -0.1683 |
| 9 | -0.0056 | -0.0038 | -0.00361 | | 0 | -1.29E-04 |
| 10 | 0.3048 | 0.309 | 0.309 | | 0.0011 | -0.2226 |
| 11 | 0.8229 | 0.828 | 0.826 | | 0.003 | -0.9775 |

## Optimization Analysis

I initially run my own optimizer and it was found that mass 9 and mass 10 decrease down to zero and mass 11 shoots up to a high value as shown in the table. These values are verified using sqp and slp_trust code. These results are validated using the research paper, this project is based on. It was also found that that total

*Table 4: Optimizer Results*

| Parameter | Design | |
|---|---|---|
| | Initial | Final |
| Mass 9 | 5.21 | 5.44E-11 |
| Mass 10 | 6.55 | 5.28E-11 |
| Mass 11 | 6.60 | 20.198 |
| Total Mass | 18.36 | 20.198 |
| Nodal Location | 154.8873 | 163.00 |

Here we can see that how the distance between new calculated nodal location and the desired nodal location decreases. This shows the convergence of our constrain function and thus showing the performance of optimizer.

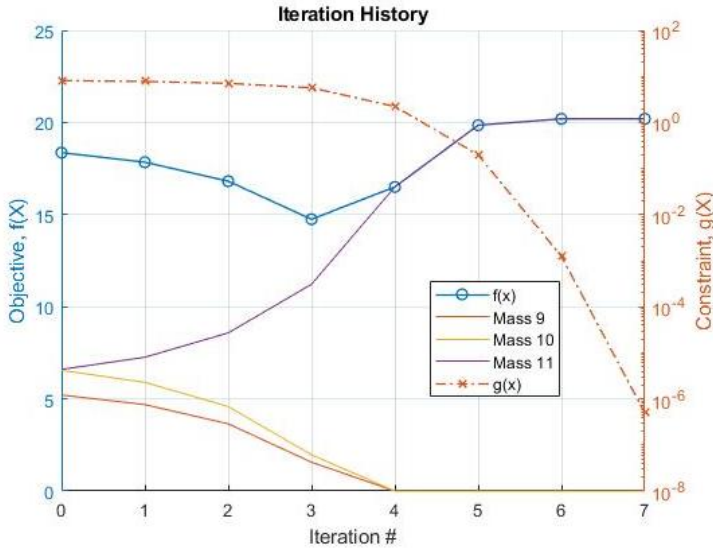


*Figure 11: Summarized Iteration History*

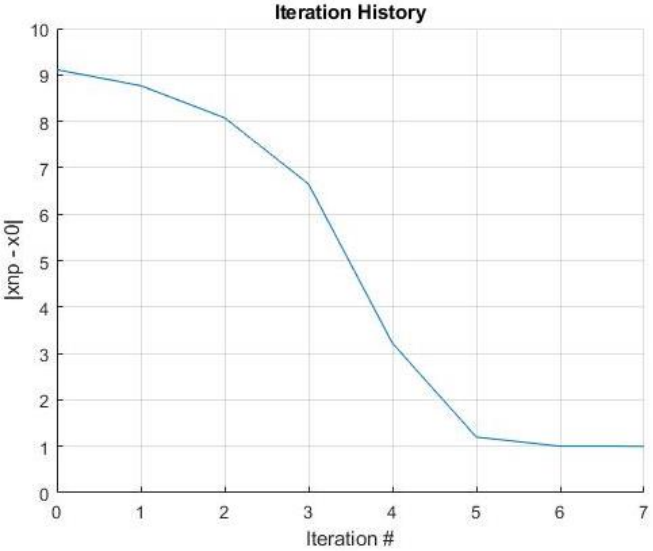


*Figure 12: Iteration History for distance between new nodal location from desired location*

From Figure 11 iteration we can see how well our optimizer is converged as the constrain value goes to the order of 10-6. Also we can see find that we have new masses and when we use the optimized mass values as our design variable F shows that our nodal location moves from 154.889 in to 163.00 in which is the desired location with the tolerance limit.

Results compared with the one given in paper. It can be seen that my results matches exactly with the one given in the paper and thus validation is done successfully

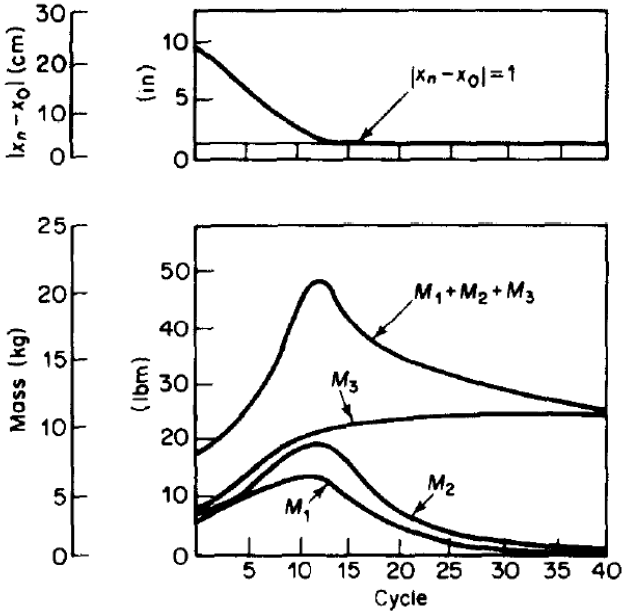


*Figure 13: Results given in research paper*

## Conclusions

Nodal point placement method for controlling vibration at desired location is demonstrated. Using slp with trust regions, the solutions get converged in less than 10 iterations. The optimizer computed the masses at grid point 9, 10, and 11 are 0, 0, 20.198 lbs respectively. This will result in nodal point location at 163 in.With the verification and validation it can be said confidently that these are computed correctly and precisely. Placing the nodal point at the centroid of load distribution minimizes the generalized load.

Future Work: Adding rotational velocity of 425 rpm to the system and reanalyzing the entire optimization. Creating current problem into multi-objective problem by minimizing the generalized force.

With the help of this project, I have a learned a lot many things specifically the basics of optimization and the state-of-the-art algorithms that are used in the industry. I have improved my programming skills with the help of this course, for instance, creating improved function handles and use of global variables and creating variables in structure that will be easy to pass while calling for the function. Implementation of Nelson's method was itself a quite challenging task, but with the help of textbook "Elements of Structural optimization" and course online lecture uploaded on canvas make it to do it successfully.

## References


[1] D. A. a. K. T. a. K. A. a. R. M. P. Peters, "Design of helicopter rotor blades for desired placement of natural frequencies," 1984.

[2] R. B. Taylor, Helicopter rotor blade design for minimum vibration, Vols. NASA CR-3825, Anaheim: National Aeronautics and Space Administration, Scientific and Technical~…, 1984.

[3] R. B. Taylor, "Helicopter vibration reduction by rotor blade modal shaping," in *Proceedings of the 38th Annual Forum of the American Helicopter Society*, 1982, pp. 90--101.

[4] H. A. R. H. J.I. Pritchard, "Sensitivity analysis and optimization of nodal point placement for vibration reduction," *Journal of Sound and Vibration,*, vol. 2, no. 119, pp. 277-289, 1987.

[5] R. B. Nelson, "Simplified Calculation of eigenvector derivatives," *American Institute of Aeronautics and Astronautics Journal*, no. 14, pp. 1201 - 1205, 1976.

# Appendix – Matlab Code

## Clearing and setting up environment

```
clear; clc; close all
```

## Global Variable Settings

```
global xiters1
global xiters2
global xiters3
global xnpiters
xiters1 =[];xiters2 =[];xiters3 =[];xnpiters=[];
```

```
X0 = [5.21, 6.55, 6.6];
dxlb = 0*ones(size(X0));
dxub = 50*ones(size(X0));
```

## Optimizer Settings

```
options.Display='iter';
options.MaxIter=100;
options.TolX=0.0001;
options.TolFun=0.0001;
options.TolCon=0.0001;
options.MoveLimit=0.1;
options.TrustRegion='on';
options.MaxFunEvals = 1000;
```

## Optimizer

```
[x,f,exitflag,output] = slp_trust(@funcs,X0,options,dxlb,dxub,@grads);

disp(' ')
disp('Final Design Variables, X')
disp(x)
```

```
          Sequential Linear Programming (SLP) Iteration History
Iteration      Objective MaxConstraint  Index   Step-size   Merit      MoveLimit  TrustRatio
        0          18.36         8.113      2           0       34.37
        1         17.844         7.767      2        0.66       33.17        0.1      0.9983  g * slack_LP
        2         16.812         7.076      2        1.32       30.78        0.2      0.9919  g * slack_LP
        3         14.748         5.649      2        2.64        25.9        0.4           1  f * slack_LP
        4           16.5         2.221      2        5.28       20.88        0.8      0.9355  g + slack_LP
        5          19.85        0.1989      2        3.35       20.24        1.6      0.9104  g +
        6         20.196      0.001236      2      0.3459        20.2        1.6      0.9938  g +
        7         20.198    -5.015e-07      2    0.002179        20.2        1.6           1  f + Unbound
               ----------  ------------         ----------
    Criteria      0.0001        0.0001             0.0001
SLP converged. Final objective function value = 20.1984
               Lagrangian gradient   2-norm = 0
               Lagrangian gradient inf-norm = 0
               Schittkowski KTO **          = 0.0021797
               Optimality Tolerance         = 1
Trust Region Strategy uses filter
* Dominates prior points
```

```
+ Nondominated
- Dominated by prior point(s)
f/g/m Objective/Constraint/Merit governs Trust Ratio

Final Design Variables, X
    0.0000
    0.0000
   20.1984
```

## Plotting

```
figure(2)
iter=(0:output.iterations);
[AX,H1,H2]=plotyy(iter,output.f,iter,abs(max(output.g,[],1)),'plot','semilogy');
set(get(AX(1),'Ylabel'),'String','Objective, f(X)');
set(get(AX(2),'Ylabel'),'String','Constraint, g(X)');
set(H1,'LineStyle','-','LineWidth',1,'Marker','o');
set(H2,'LineStyle','-.','LineWidth',1,'Marker','x');
xlabel('Iteration #');
title('Iteration History');
hold on
plot(iter,xiters1, iter,xiters2, iter,xiters3)
grid on

legend('f(x)','Mass 9','Mass 10','Mass 11','g(x)')

figure(3)
xlabel('Iteration #');
title('Iteration History');
hold on
plot(iter,abs(xnpiters-164.0))
grid on
ylabel('|xnp - x0|')
```

## grads.m

```
type grads
```

```
function [gradf, gradg] = grads(dv)
[FEM, mLumped, N, M, Mindex] = setup(dv);
FEM = FEMsolve(FEM, mLumped);
[eigVecG, eigVal, kCurtailed, mCurtailed] = eigenAnalysis(FEM);
[nodalLoc, w] = modalAnalysis(N, FEM, eigVecG);
dwdw = sensitivity(N, FEM, M, mLumped, nodalLoc);
gradf = ones(size(Mindex))';
gradg = [dwdw(Mindex)',-dwdw(Mindex)'];
```

## funcs.m

```
type funcs
```

```
function [f, g] = funcs(dv)
global xiters1
global xiters2
global xiters3
xiters1 = [xiters1,dv(1)];
xiters2 = [xiters2,dv(2)];
xiters3 = [xiters3,dv(3)];
```

```
delta = 1.0;
x0 = 164.0;
[FEM, mLumped, N, M, Mindex] = setup(dv);
FEM = FEMsolve(FEM, mLumped);
[eigVecG, eigVal, kCurtailed, mCurtailed] = eigenAnalysis(FEM);
[nodalLoc, w] = modalAnalysis(N, FEM, eigVecG);

global xnpiters
xnpiters = [xnpiters,nodalLoc(2,2)];

f = sum(dv);
g1 = nodalLoc(2,2) - x0 - delta;
g2 = -(nodalLoc(2,2) - x0) - delta;
g = [g1,g2];
```

## setup.m

```
type setup
```

```
function [FEM, mLumped, N, M, Mindex] = setup(dv)
    %% System Definition
    N.N1 = @(x,le) 1 - 3*x.^2./le^2 + 2*x.^3./le^3;
    N.N2 = @(x,le) x - 2*x.^2./le + x.^3./le^2;
    N.N3 = @(x,le) 3*x.^2./le^2 - 2*x.^3./le^3;
    N.N4 = @(x,le) -x.^2./le + x.^3./le^2;

    N.N1_prime = @(x,le) -6*x./le^2 + 6*x.^2./le^3;
    N.N2_prime = @(x,le) 1 - 4*x./le + 3*x.^2./le^2;
    N.N3_prime = @(x,le) 6*x./le^2 - 6*x.^2./le^3;
    N.N4_prime = @(x,le) -2*x./le + 3*x.^2./le^2;

    % number of element
    FEM.nelm = 10;

    % number of nodes
    FEM.nodes = FEM.nelm + 1;

    % number of degree of freedom per element
    FEM.ndof = 2;
    FEM.tdof = FEM.nodes*FEM.ndof;

    E = ones(1,FEM.nelm)*0.585e7;
    E(1) = 0.490e7;

    rho = ones(1,FEM.nelm)*0.07;
    FEM.elmL = ones(1,FEM.nelm)*(193/FEM.nelm);

    b = ones(1,FEM.nelm)*3.75;
    h = ones(1,FEM.nelm)*2.5;
    t = ones(1,FEM.nelm)*0.8;
    d = ones(1,FEM.nelm)*0.1;

    bh = b - 2*d;
    hh = h + 2*t;

    I = 1/12*(b.*hh.^3 - bh.*h.^3);
    A = b.*hh - bh.*h;
```

```
    FEM.kProp = (E.*I)./FEM.elmL.^3;
    FEM.mProp = (rho.*A.*FEM.elmL)/420;

    % Lumped Masses
    M = zeros(1,FEM.nodes);
    M(3) = 3.04;    M(4) = 1.67;    M(6) = 6.40;    M(7) = 7.46;
    M(8) = 10.75;    M(9) = 5.21;    M(10) = 6.55;    M(11) = 6.6;

    Mindex = [9, 10, 11];
    M(Mindex) = dv;

    for i=1:FEM.nelm
        FEM.connect(:,i) = [i;i+1];
    end

    mLumped = zeros(FEM.tdof, FEM.tdof);
    for i = 1:size(M,2)
        mLumped(2*i-1,2*i-1) = M(i);
    end
end
```

## sensitivity.m

```
type sensitivity
```

```
function dwdw = sensitivity(N, FEM, M, mLumped, nodalLoc)

[eigVecG, eigVal, kCurtailed, mCurtailed] = eigenAnalysis(FEM);
%% EigenVector Sensitivity

dv_check = M>0;

lambda = eigVal(2);
phi = eigVecG(3:end,2);
[m, loc] = max(abs(phi));

K_prime = zeros(size(kCurtailed));
for i = 1:size(M,2)
    M_primeG = zeros(size(mLumped));

    % check availibility of design variable at grid point
    if dv_check(i)
        M_primeG(2*i-1,2*i-1) = 1;
    end

    M_prime = M_primeG(3:end,3:end);
    lambda_prime = phi'*K_prime*phi - lambda*phi'*M_prime*phi;

    F = kCurtailed - lambda*mCurtailed;
    F_prime = K_prime - lambda_prime*mCurtailed - lambda*M_prime;

    F_temp = F;
    F_temp(loc,:)=[];
    F_temp(:,loc)=[];
    F_tilda = F_temp;

    Fprime_temp = F_prime;
    Fprime_temp(loc,:)=[];
    Fprime_temp(:,loc)=[];
```

```
    F_tilda_prime = Fprime_temp;

    phi_temp = phi;
    phi_temp(loc)=[];
    phi_tilda = phi_temp;

    V = - inv(F_tilda)*F_tilda_prime*phi_tilda;
    q = [V(1:loc-1);0;V(loc:end)];
    c = -phi'*mCurtailed*q - 0.5*phi'*M_prime*phi;

    phi_prime(:,i) = q + c*phi;
end

%% Derivatives

modeNum = 2;
np_elm = nodalLoc(2,1);
xnp = nodalLoc(2,2)-(np_elm-1)*FEM.elmL(1);

dNdx = [N.N1_prime(xnp, FEM.elmL(np_elm)), N.N2_prime(xnp, FEM.elmL(np_elm)),...
        N.N3_prime(xnp, FEM.elmL(np_elm)), N.N4_prime(xnp, FEM.elmL(np_elm))];
dwdx = dNdx * eigVecG(2*np_elm-1 : 2*np_elm + 2, modeNum);

for i=1:size(M,2)
    Nxnp = [N.N1(xnp,FEM.elmL(np_elm)), N.N2(xnp,FEM.elmL(np_elm)),...
            N.N3(xnp,FEM.elmL(np_elm)), N.N4(xnp,FEM.elmL(np_elm))];

    dwdv(i) = Nxnp * phi_prime((2*(np_elm-1) - 1) : 2*np_elm, i);
end
dwdx;
dwdv;
dwdw = -dwdv/dwdx;
```

## modalAnalysis.m

```
type modalAnalysis
```

```
function [nodalLoc, w] = modalAnalysis(N, FEM, eigVecG)

modeNum = 2;

N1=N.N1;
N2=N.N2;
N3=N.N3;
N4=N.N4;

elmL = FEM.elmL;

cnt = 0;
for i=1:FEM.nelm
    x1 = linspace(0,elmL(i),50);
    x2 = linspace(sum(elmL(1:i-1)),sum(elmL(1:i)),50);
    w(:,i) = N1(x1,elmL(i)).*eigVecG(2*i-1,modeNum) + N2(x1,elmL(i)).*eigVecG(2*i,modeNum)...
        + N3(x1,elmL(i)).*eigVecG(2*i+1,modeNum) + N4(x1,elmL(i)).*eigVecG(2*i+2,modeNum);

    xnp0_ind = find(diff(sign(w(:,i))));
    if find(diff(sign(w(:,i))))
        cnt = cnt + 1;
```

```
        nodalElm = i;
        xnp = @(x) N1(x,elmL(nodalElm))*eigVecG(2*nodalElm-1,modeNum) +
N2(x,elmL(nodalElm))*eigVecG(2*nodalElm,modeNum)...
            + N3(x,elmL(nodalElm))*eigVecG(2*nodalElm+1,modeNum) +
N4(x,elmL(nodalElm))*eigVecG(2*nodalElm+2,modeNum);
        options = optimset('Display','off');
        xnp = fsolve(xnp,w(xnp0_ind,nodalElm),options);
        nodalLoc(cnt,:) = [nodalElm, sum(elmL(1:i-1)) + xnp];
    end
    plot(x2,w(:,i),'LineWidth',2)
    hold on
end
grid on
title("Mode Shape")
xlabel('Length');
ylabel('Displacement');
hold on
```

## FEMsolve.m

```
type FEMsolve
```

```
function [FEM] = FEMsolve(FEM, mLumped)
FEM.kG = zeros(FEM.tdof, FEM.tdof);
FEM.mC = zeros(FEM.tdof, FEM.tdof);
elmL = FEM.elmL;
for i = 1:FEM.nelm
    kL = FEM.kProp(i)*[12         6*elmL(i)    -12          6*elmL(i)    ;
                       6*elmL(i)  4*elmL(i)^2  -6*elmL(i)   2*elmL(i)^2  ;
                      -12        -6*elmL(i)     12         -6*elmL(i)    ;
                       6*elmL(i)  2*elmL(i)^2  -6*elmL(i)   4*elmL(i)^2 ];

    mL = FEM.mProp(i)*[156         22*elmL(i)    54          -13*elmL(i)   ;
                       22*elmL(i)  4*elmL(i)^2   13*elmL(i)  -3*elmL(i)^2  ;
                       54          13*elmL(i)    156         -22*elmL(i)   ;
                      -13*elmL(i) -3*elmL(i)^2  -22*elmL(i)   4*elmL(i)^2 ];
    cnt = FEM.connect(:,i);
    conmat = [2*cnt(1)-1, 2*cnt(1), 2*cnt(2)-1, 2*cnt(2)];
    for j = 1:4
        for k=1:4
            FEM.kG(conmat(j),conmat(k)) = FEM.kG(conmat(j),conmat(k)) + kL(j,k);
            FEM.mC(conmat(j),conmat(k)) = FEM.mC(conmat(j),conmat(k)) + mL(j,k);
        end
    end
end
FEM.mG = FEM.mC + mLumped;
```

## eigenAnalysis.m

```
type eigenAnalysis
```

```
function [eigVecG, eigVal, kCurtailed, mCurtailed] = eigenAnalysis(FEM)
kCurtailed = FEM.kG(3:end,3:end);
mCurtailed = FEM.mG(3:end,3:end);
[eigVec, eigVal] = eig(kCurtailed,mCurtailed);
eigVal = diag(eigVal);
eigVecG(FEM.tdof,FEM.tdof-2) = 0;
eigVecG(3:end,:) = eigVec(1:end,:);
```