INVERSE MODE SHAPE PROBLEM FOR BARS, BEAMS, AND PLATES

A Project Report

submitted in partial fulfillment of the

requirements for the degree of

Master of Technology

in

Computational Science

by

M.Meenakshi Sundaram

Supercomputer Education and Research Centre

Indian Institute of Science

Bangalore-560012

June 2009

ACKNOWLEDGEMENT

This project has been an interesting mix of numerical and analytical work and I definitely owe most of it to my Project Supervisor Dr. G. K. Ananthasuresh who had conceived this project long back and had given me the opportunity to extend the same. The proofs in the case of the bars had been inspired by a course on Analysis conducted by Dr. Kaushal Verma and a fellow mate in the course Mrs. Lalitha. V, a research scholar in the institute. I duly acknowledge their influence on this part of my project. Mr. A. Narayana Reddy and Mr. Sangamesh Deepak. R, research scholars in my lab were helpful especially when I was getting through the chunk of the reading in Gladwell's book. Some interactions with Dr. C. S. Jog, Dr. A. Mohanty and a brief discussion with Dr. Anindya Chatterjee proved really helpful and I thank them all. Beyond these people I would like to mention that my fellow lab mates both in SERC and in Mechanical Engineering and friends all around this campus have made this stay in IISc very memorable.

ABSTRACT

INVERSE MODE SHAPE PROBLEM FOR BARS, BEAMS, AND PLATES

By M. Meenakshi Sundaram

Project Advisor: Professor G. K. Ananthasuresh, Mechanical Engineering

The objective of this work is to determine the cross-sectional (c/s) profile of a regular structure such as a bar, beam, or a plate for a desired eigenmode shape and given boundary conditions and total mass. This inverse problem is formulated as a new eigenvalue problem in terms of the geometric parameters by using the original eigenvalue problem in a finite element setting. The numerical solution of the problem indicates that the eigenvalue is uniquely determined when it is linear in the geometric parameters and not so otherwise. We observe that the design variables are unique up to a scaling factor which is fixed by the total mass. We also observe that the numerical algorithm converges only when the mode shape is *valid* for the specified boundary conditions. The study of the validity of mode shapes is done in the case of bars which helped answer the questions of existence, uniqueness, and construction procedure for the same. The case of uniqueness in the cantilever beams when the c/s parameters are linear was also proven. An extension to the numerical algorithm is necessitated due a need for accuracy in the mode shape to be supplied. As a final conclusion a numerical test for the validity of a mode shape is proposed where a computation of a residual is indicated.

TABLE OF CONTENTS

1. INTRODUCTION

An instance where the *design for a mode shape* is necessary is the confinement of vibration amplitudes at desired locations on a structure [1]. Many resonance-based sensors working on capacitive principles depend largely upon modal deformations of the elastic body. Here too, it would be beneficial to develop a systematic method to design structures for the eigenmode. The intent of this work is to do so for the regular cases (bars, beams, and plates).

An example of a sensor where modal deflection plays a crucial role would be the case of the coupled microcantilever gravimetric sensors used in biochemical analyte detection [2]. Atomic Force Microscopes have probes which can be designed for large tip deflection during their traversal at resonant frequency in their tapping mode point to another application [3]. Though the 'inverse frequency' problem has been studied extensively [4, 5] there is little work done on the 'inverse mode shape' problem. These belong to a class of problems called *partially described inverse eigenvalue problems* [6].

The next section deals with the work done earlier in the direction of solving the inverse eigenvalue problems as applied to regular structures. The following section describes the *'inverse mode shape'* problem followed by the description of the numerical algorithm in a finite element framework and the discussion of the results so obtained in all the cases. The section after that, deals with the question of what exactly is a valid mode shape in the case of the bars. It is noted by Lai and Ananthasuresh [11] that any arbitrary function satisfying the boundary conditions need not be a valid mode shape and that they need to satisfy certain conditions. A study in the case of uniqueness for the cantilever case is done when the geometric parameters are linear in the equation. Some extensions to the numerical technique are further necessitated and developed owing to the accuracy desired. A numerical test for the numerical validity of a mode shape is also done. The report closes with a conclusion and the challenges that lie ahead.

2. RELATED WORK

The inverse eigenvalue problem has been addressed by many authors. Here the aim was to determine both the geometric and material data of the structure. Ram and Gladwell [7] had come up with a procedure to find all the elements of the finite element model matrices, i.e., the mass and stiffness matrices from two eigenvectors, one eigenvalue and total mass. This was further carried on, in the continuous case too by Ram [8].

For the case of the beams, Barcilon [9] had come up with a procedure for the beam clamped at one end and three spectra each with respect to three different

boundary conditions on the other end to obtain a unique solution. Gladwell [10] followed up with establishing necessary and sufficient conditions for the reconstruction of the Euler-Bernoulli beam from spectral data.

The inverse mode shape problem was introduced by Lai and Ananthasuresh [11] and the problem that was solved involved the knowledge of just one eigenmode and all parameters (mass, boundary conditions, Young's modulus, density) except the shape parameterized by a single unknown function. This work was further extended by Ananthasuresh [12] for the case of flexible supports.

It was noticed by them that the eigenfrequency was uniquely determined when the mode shape was specified and also that all possible deflections cannot be mode shapes. Our work attempts at justification of these statements and also extends the same to a more general framework by presenting a numerical algorithm.

3. THE INVERSE MODE SHAPE PROBLEM

Given the knowledge of material properties, boundary conditions, a constraint of total mass and a desired mode shape, the objective is to find a suitable function for the design variables that governs the shape of the structure.

The governing differential equation for the bars is

$$\frac{d}{dx}\left[EA\frac{du}{dx}\right]+\lambda\rho Au=0 \qquad x\in[0,L] \tag{1}$$

The symbols E,ρ and stand for Young's modulus, density and area of cross-section respectively, all positive everywhere in the domain. Symbol u refers to the mode shape and λ is a positive constant which is the square of the natural frequency. Area depends on the cross-sectional dimension and the equation in the unknown is a linear one.

The governing differential equation for the Euler-Bernoulli beams is given by

$$\frac{d^2}{dx^2}\left(EI\frac{d^2u}{dx^2}\right)-\lambda\rho Au=0 \qquad x\in[0,L] \tag{2}$$

where I stands for the moment of inertia of the beam which is also a positive function. The rest of the symbols also carry the same meaning as in the case of the bars. Both I and A are functions of the cross-sectional dimensions. So a common parameterization needs to be done depending upon the shape and the cross-sectional dimension of interest. If the beam is assumed to be of rectangular shape and the breadth of the beam as the varying dimension then the problem is linear in terms of the cross-sectional dimension or else it becomes nonlinear if we

assume the depth of the beam as the dimension of interest in the rectangular case (The dimension is raised to the power 3 in the case of the moment of inertia and raised to the power 1 in the case of area. So n, meaning the degree of the moment of inertia term in the equation, is 3). If the radius of the beam in the circular case is a variable (The dimension is raised to the power 4 in the case of the moment of inertia and raised to the power 2 in the case of the area. So n is 2) then too the problem becomes nonlinear. This is described in Eq. (3) with the variable $A(x)$

$$\frac{d^2}{dx^2}\left(EA^n\frac{d^2u}{dx^2}\right)-\lambda\rho Au=0 \qquad x\in[0,L] \qquad \begin{cases}\text{Circular c/s}\\ n=2\\ \text{Rectangular c/s}\\ n=3 \quad \text{Depth varying}\\ n=1 \quad \text{Width varying}\end{cases} \tag{3}$$

where A^n replaces the moment of inertia with n taking different values according to the design variable chosen.

As we extend the problem to plates, we have the thickness $t(x, y)$ of the plate to be found as shown in the governing differential equation for Kirchoff Plates in Eq. (4)

$$\nabla^2\left(D\nabla^2\left(u\right)\right)-\lambda\rho tu=0$$
$$D=\frac{Et^3}{12\left(1-\nu^2\right)} \tag{4}$$

where ν represents the poisons ratio. This is nonlinear as t is raised to the power of 3.

All these problems translate in the finite element setting into the generalized matrix eigenvalue problem in Eq. (5).

$$\boldsymbol{K}\left(\boldsymbol{A}\right)\boldsymbol{u}=\lambda\boldsymbol{M}\left(\boldsymbol{A}\right)\boldsymbol{u} \tag{5}$$

The structural information desired is the geometric parameter (A) inside the stiffness and mass matrices $\mathbf{K}$ and $\mathbf{M}$ respectively. The eigenvalue is also an unknown parameter here. Boundary conditions are varied in each of the problems and they do play an important role as observed by Lai and Ananthasuresh [11].

4. THE NUMERICAL ALGORITHM

From the numerical stand point the discretized problem poses a challenge as we are looking for the geometric parameter inside the matrix which occurs in a certain pattern. The proposed algorithm is depicted as a flowchart in Fig. 1. The central idea of this algorithm is 'swapping' – the rearrangement of variables an idea originally proposed in [7]. We extend the same to formulate an eigenvalue problem.

To understand this idea, let us take a look at a simple example in Eq.(6).This example illustrates that the underlying equations are same in both sets except that the variables $\{p,q,r\}^T$ are replaced by $\{x,y,z\}^T$. Let us now tackle the problem at hand. The assembly of finite element matrices is depicted in Fig. 2a. During the assembly of FE matrices corresponding to the discretized eigenvalue problem, the element matrices with the multiplication of element properties (assumed to be constant and of which one is of interest to us) are combined additively.

$$\begin{pmatrix} 4x & 5y & 6z \\ 1x & 3y & 3z \\ 4x & 2y & 5z \end{pmatrix}\begin{pmatrix} p \\ q \\ r \end{pmatrix} = \begin{pmatrix} l \\ m \\ n \end{pmatrix}$$
$$\begin{pmatrix} 4p & 5q & 6r \\ 1p & 3q & 3r \\ 4p & 2q & 5r \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} l \\ m \\ n \end{pmatrix} \qquad (6)$$

The geometric variables of interest are $\boldsymbol{g}_1, \boldsymbol{g}_2, \boldsymbol{g}_3$. The matrices are square of dimension equal to the free degrees of freedom (dof). On swapping, the result obtained is depicted in Fig. 2b. The element matrices with the unknown parameter pulled out is multiplied with the displacements corresponding to the element dof and assembled in the column corresponding to the element at the rows specified by the dof.

The resultant matrix will be rectangular if the number of elements and the number of free dof are different. To avoid the issue of rank deficiency one needs to have the system to be constrained (number of equations equal to the number of unknowns) or overconstrained (number of equations greater than the number of unknowns). So, a proper choice of order of interpolation for the geometric variable and the displacement is to be done. This necessitates a three-noded element in the FE model for the bar.

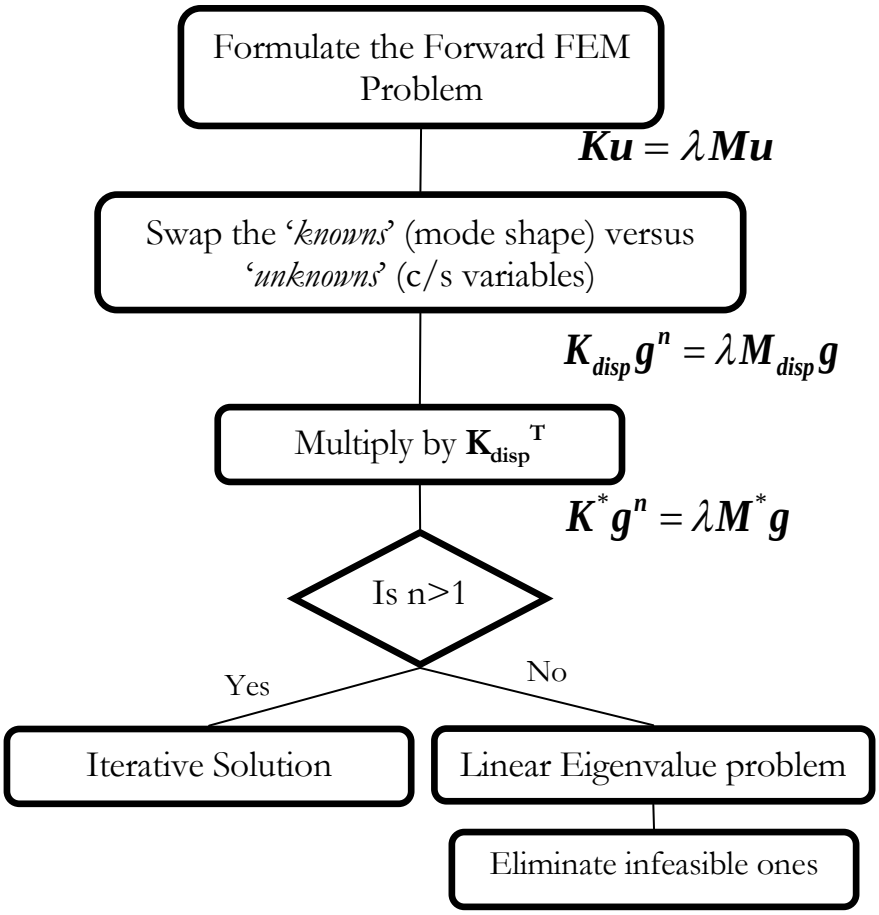

Fig 1: Flowchart of the numerical algorithm

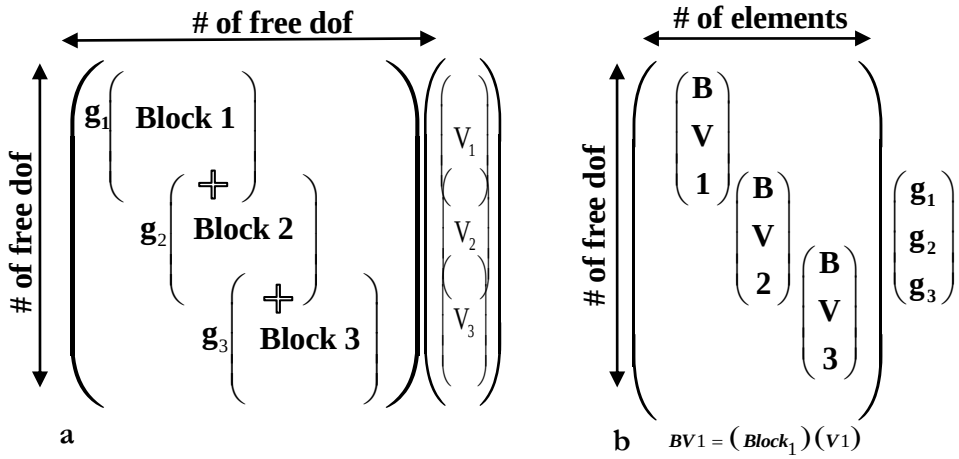

Fig 2: Pictorial description of the FE assembly and the result of swapping

If an overconstrained rectangular system is obtained, the system is converted to a square system by the multiplication of the transpose. The resulting problem might be nonlinear or linear depending on the dimension parametrizing the c/s profile. The linear problem is a generalized eigenvalue problem with the feature that $\boldsymbol{K}^* = \boldsymbol{K}_{disp}{}^T \boldsymbol{K}_{disp}$ is symmetric and real, while $\boldsymbol{M}^* = \boldsymbol{K}_{disp}{}^T \boldsymbol{M}_{disp}$ is real but not symmetric. After the enumeration of the eigenvectors and eigenvalues of $\boldsymbol{K}^* \boldsymbol{g}^n = \lambda \boldsymbol{M}^* \boldsymbol{g}^n$, the unreasonable values (negative, complex eigenvalues and negative eigenvectors after normalization) are ignored. This algorithm is seen to give a proper result only if, there is atleast one eigenvalue and eigenvector that is reasonable.

In the nonlinear case the notation $\boldsymbol{g}^n$ means each element is raised to the power n. This is solved iteratively in a manner very similar to power method [13]. The flowchart (Fig. 3) illustrates the procedure.

The mass constraint is used to determine the scaling value that must be taken by the vector $\boldsymbol{g}$. This does not affect the frequency in the linear case but does so in the nonlinear case and is given by Eq. (7).

$$\lambda_{new} = \lambda_{withnormalization} c^{n-1} \qquad (7)$$

c - scaling constant to match the mass

The results in all the cases are discussed next.

5. Results and Discussion

The numerical algorithm was tested using known data as follows. The geometric variables were assumed and the mass was calculated. The desired mode shape was found and then fed to the algorithm. The solution was compared against the assumed values. The Young's Modulus used for all these cases is 210 MPa and the density is 7800 kg/m^3 – the properties of steel.

5.1 BARS

The case of the bar is very illustrative and also faces some numerical issues. When the fixed-fixed end conditions are assumed for a two-noded element (with single dof per node) the number of free dof (total number of elements -1) is just one short of the number of elements involved. The problem therefore becomes underconstrained (meaning

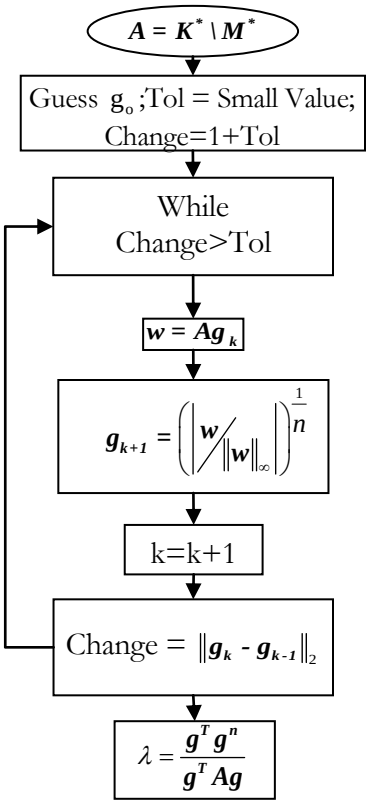

Fig 3: Algorithm for the problem $Kg^n = \lambda Mg$

lesser number of equations than unknowns). To avoid rank deficiency we took a three-noded element with constant area which provides more number of equations than unknowns. Since it results in a linear eigenvalue problem the solution was obtained using the 'eig' routine of Matlab (2007b, The MathWorks, Inc., Natick, MA, USA). The results are shown in Figs. 4 & 5. The relative error obtained is of very low magnitude for both the c/s profile (visually it is difficult to discern them) and the frequency. (Note: Relative Error in area is computed as $\left\| A_{obtained} - A_{given} \right\|_2 / \left\| A_{given} \right\|_2$).

As mentioned earlier the algorithm also gives profiles which have discontinuities in the c/s dimension, but which do follow the original c/s profile till a certain point. This is seen more frequently in the higher mode shapes in bars. This numerical artifact (will be justified later) also occurs and vanishes as one chooses different number of elements for the discretization. This can be removed by interpolating the areas as will be discussed in section 8. An example of this artifact is shown in Figs. 6 & 7.

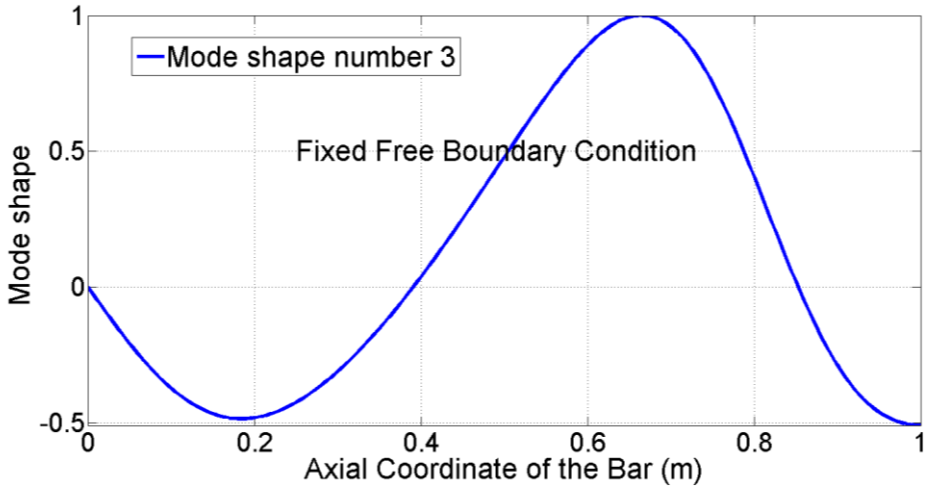

Fig 4: The third mode shape for the fixed-free boundary condition, which is sought and the area profile used to generate this case was $1e-6\times e^{(5+\sin(2\pi x/l))}m^2$. 150 three-noded elements were taken for a length of 1 m and a mass of 1.0633 kg.

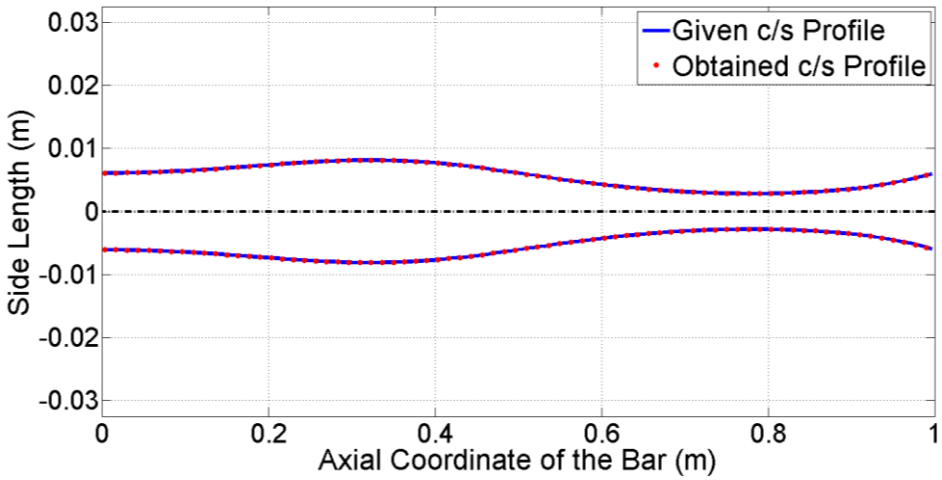

Fig 5: The relative error in area is of the order of 1e-9%. The angular frequency obtained by both the forward and the inverse problems is 4.1063e4 Hz and the relative error is of the order of 1e-11%. The cross section is assumed to be square for representation.

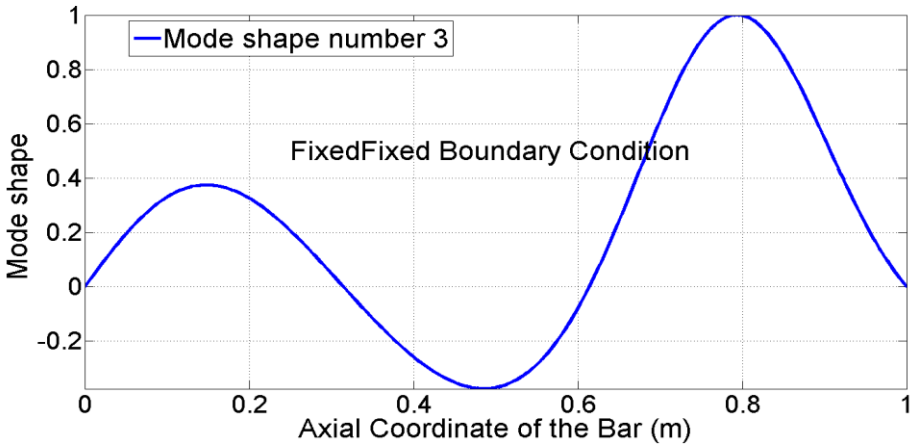

Fig 6: The third mode shape for the fixed-fixed boundary conditions with 150 three-noded elements generated from the same area profile and length as before.

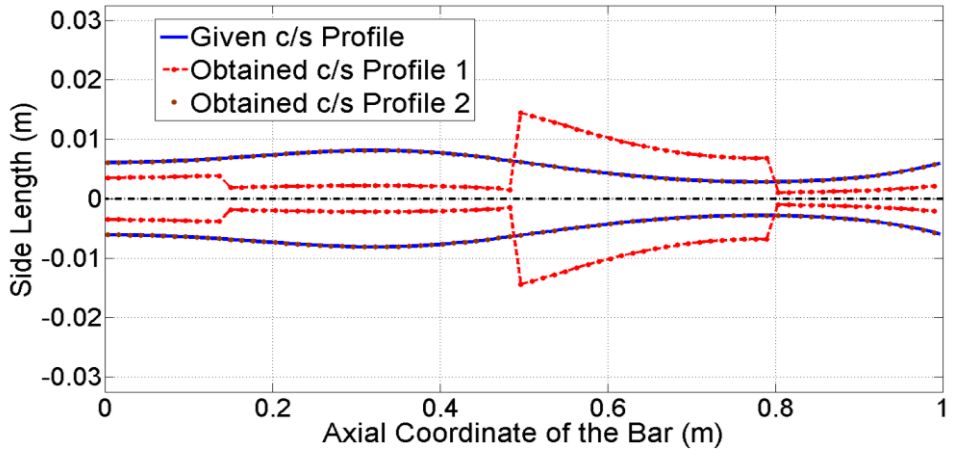

Fig 7: Two cross section profiles are obtained, one with discontinuities and one following the original profile. The relative error in area in the discontinuous case is about 153.86% and the relative error in frequency is about 0.79% while the other case has errors of the order of 1e-8% and 1e-11% respectively. The frequency obtained is 5.2467e4Hz. Again a square cross-section is assumed for representation.

5.2 EULER BERNOULLI BEAMS

We have observed that the beams do not suffer from numerical issues as the bars and we attribute it to the proper interpolation chosen. Here a two noded (two dof per node) element is chosen. In the case of the beams one can provide boundary conditions such as a fixed support, free support or a hinged support. The example for the linear case is shown in Figs. 8 and 9 followed by an example for the nonlinear case in Figs. 10 and 11. The linear one is solved using the 'eig'

function as in the case of the bars, but the nonlinear one is solved using the iterative algorithm depicted in Fig. 3. These cases show that the method indeed works and can identify the geometric properties with a good accuracy (low relative error).

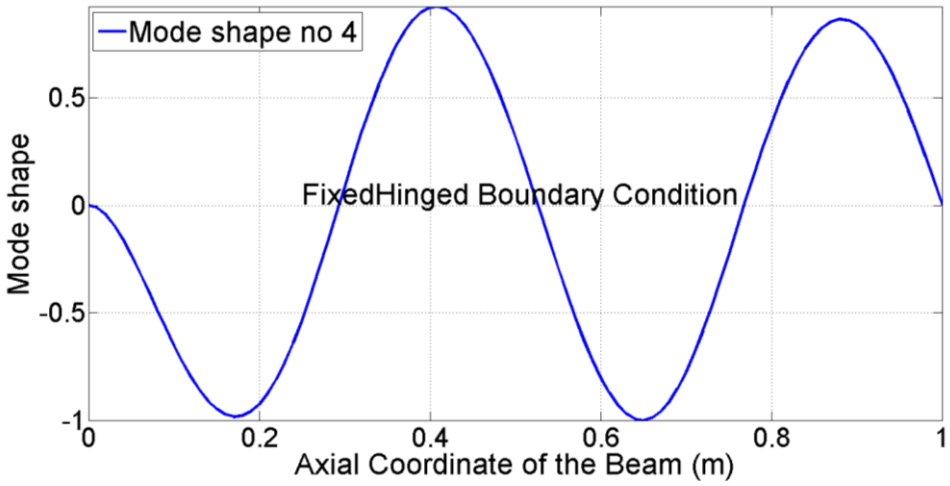

Fig 8: The fourth mode shape for the fixed hinged boundary condition and its derivative generated from the area profile $1e-6\times\left(\sin\left(5\pi x/l\right)+6\right)m^2$. The length of the beam is 1m and the moment of inertia is the same as the area profile (linear). 100 two-noded beam elements were used and the mass of the beam is 0.0478 kg.

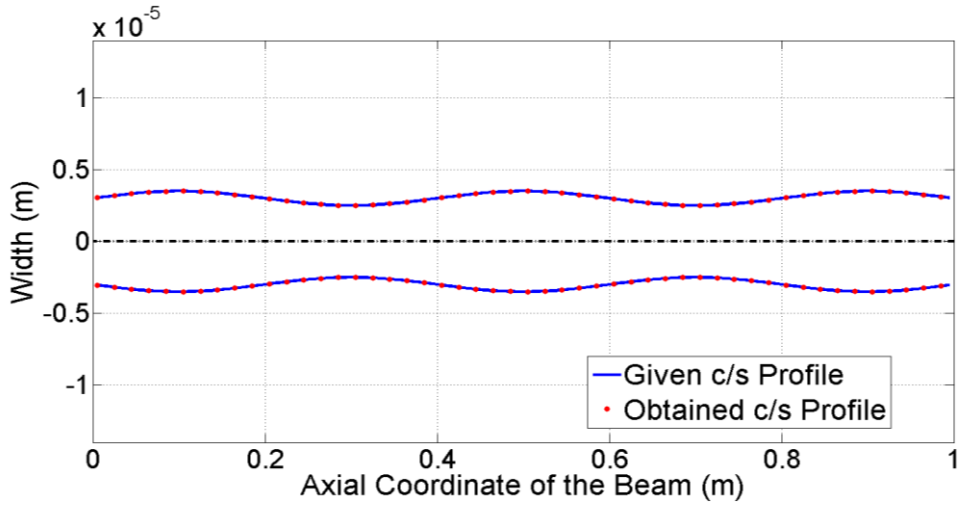

Fig 9: The width of the rectangular beam obtained follows the input. The error in the area is of the order of 1e-9% and the frequency obtained has an error of the 1e-10%. The angular frequency is 959.772 Hz.

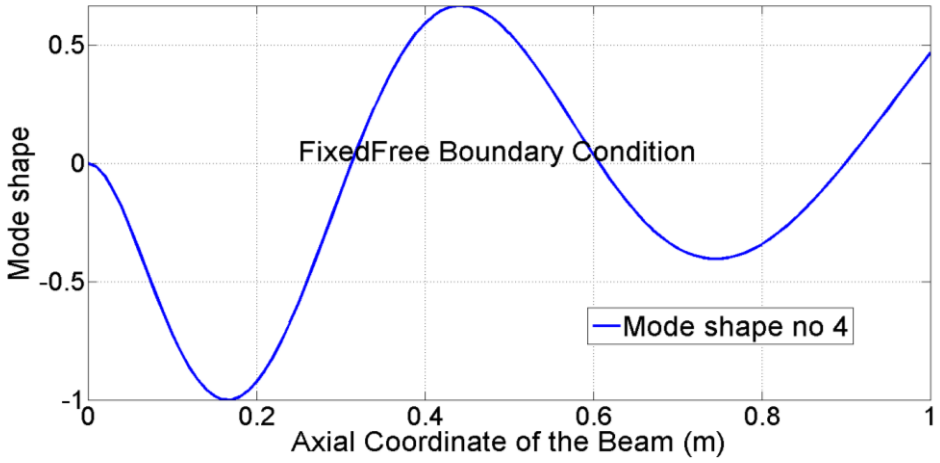

Fig 10: The fourth mode shape, generated from the circular area profile $e^{(2x)} \times 1e-6m^2$ for the fixed free boundary conditions. The length of the beam is 1m of mass 0.0245 kg. 100 two-noded beam elements were taken for simulation.

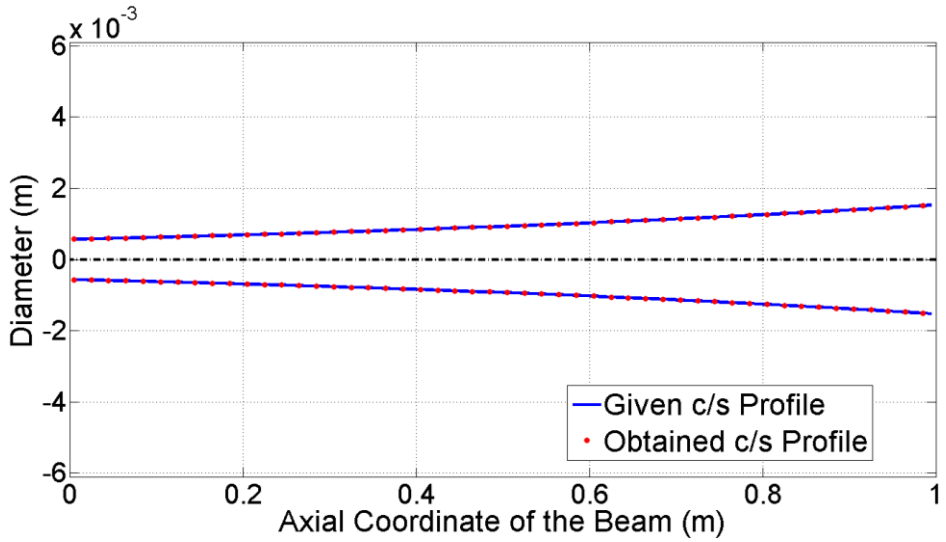

Fig 11: The radius profile and frequency match with an error of the order of 1e-10%. Frequency obtained is 31.236 Hz and the number of iterations taken for convergence is 45 under tolerance criterion on second norm of 1e-14.

5.3 PLATE

Poisson's ratio is taken to be 0.3 and the rest of the values are similar to the ones chosen in the case of bars and beams. Owing to the nonlinear nature, the iterative algorithm is used. The Adini-Clough-Melosh element with 4 nodes (three dof per node) is used for this Kirchoff Plate. The results are shown in Figs. 12 to 15 with

two different boundary conditions and mode shapes taken. The relative error is again very low indicating that the iterative method performs well.

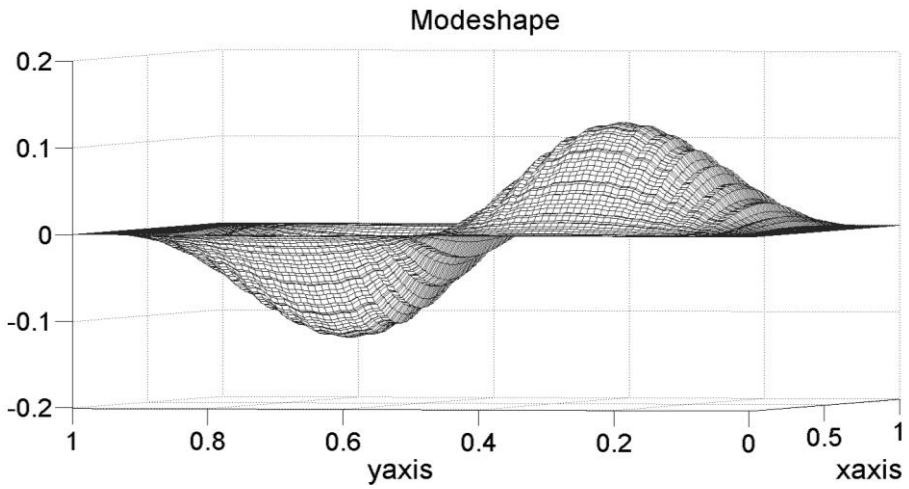

Fig 12: The second mode shape generated by the thickness profile given by $\left(\left(x-0.5\right)^2+\left(y-0.5\right)^2+1\right)\times 1e-5m^2$. This is a square plate of side 1 m discretized into a 30 × 30 mesh. The mass of the entire structure is 0.0919 kg. The plate is fixed on all edges and its Poisson's ratio is 0.3

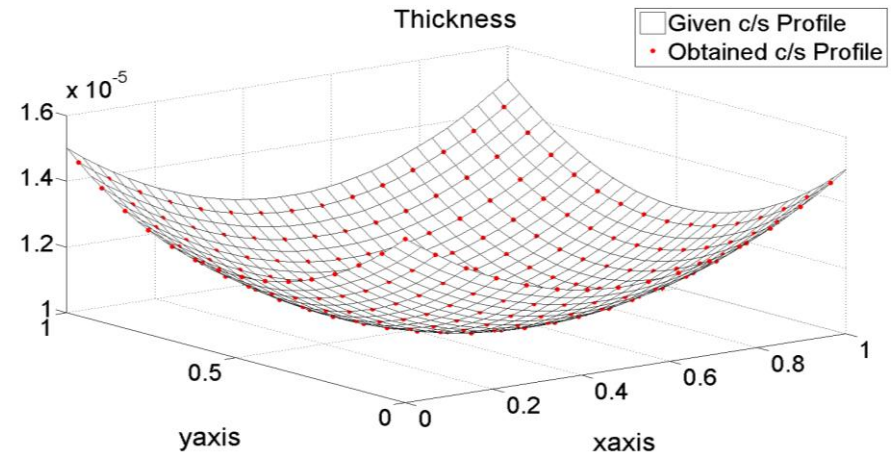

Fig 13: The element used is a 4 noded, 3 dof per node Adini-Clough-Melosh element.. The frequency obtained is 1.3465 Hz and the relative error in it is of the order of 1e-11% .The relative error in the thickness profile obtained is of the order of 1e-8%. The total number of iterations is taken to converge to an accuracy of 1e-14 is 25.

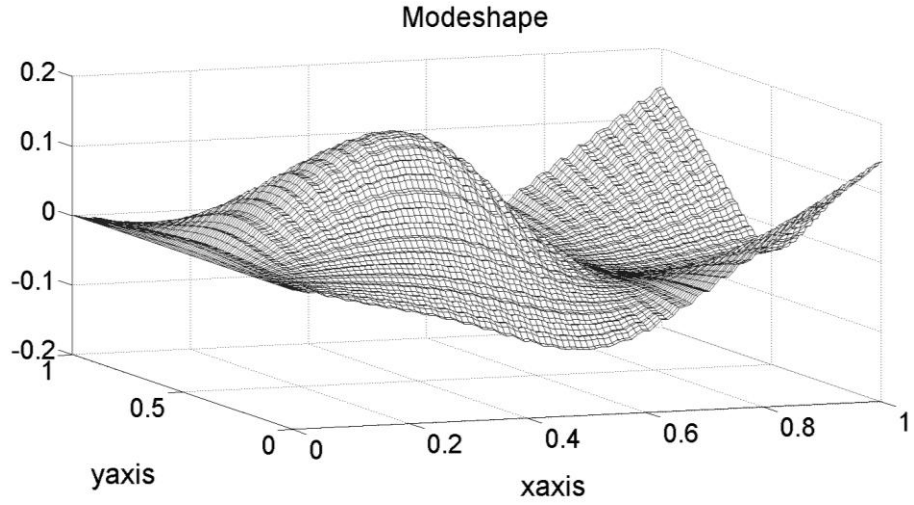

Fig 14: The sixth mode shape generated by the thickness profile given by $\left(\mathrm{Sin}\left(4\pi\left(x-0.5\right)^2\right)+\mathrm{Sin}\left(4\pi\left(y-0.5\right)^2\right)+1\right)\times 1e-5m^2$. This is a square plate of side 1 m discretized into a 30 × 30 mesh. The mass of the entire structure is 0.15376 kg. The plate is fixed on one edge and its Poisson's ratio is 0.3

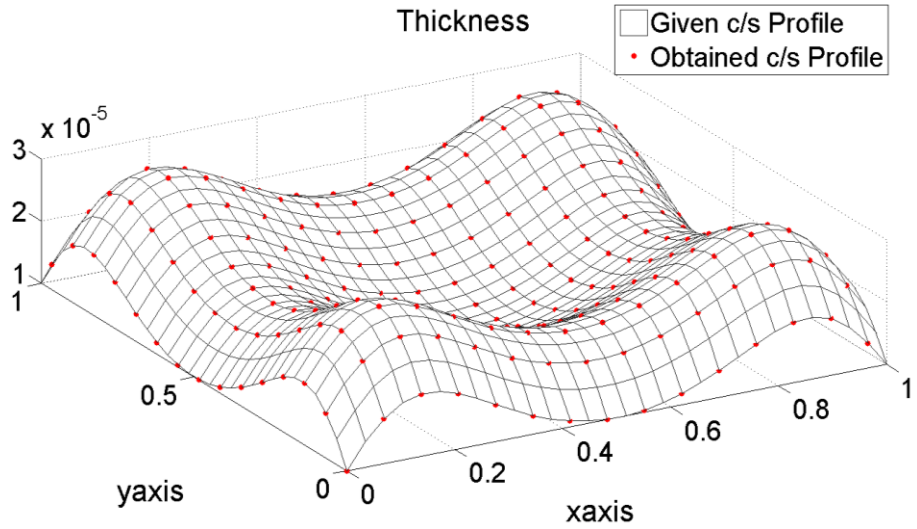

Fig 15: The frequency obtained is 1.7645 Hz and the relative error in it is of the order of 1e-10% .The error in the thickness profile obtained is of the order of 1e-8%. The total number of iterations required to converge to an accuracy of 1e-14 is 24.

5.4 Effect of Perturbation

To study the effect of arbitrary functions that satisfy the boundary condition as mode shapes we perturbed the actual mode shapes (the value at the boundary nodes were retained) by random perturbations of the order of 1e-6 relative to the normalized mode shape. We observed that the algorithm failed to yield any feasible solution in the linear case and it failed to converge in the nonlinear case. This leads us to the question: what exactly is a valid mode shape? We study this in the case of the bars to answer this question.

6. Valid Mode Shapes for a Bar

Let us begin by defining what we mean by a 'valid' mode shape

Valid Mode Shape:
A valid mode shape is one that arises as a solution of Eq. (1) which has the following conditions built in it:

1. Area is strictly positive and C^1 continuous.
2. Density is strictly positive and C^0 continuous.
3. Young's Modulus is strictly positive and C^1 continuous.
4. Boundary conditions such as one in Eq. (8)

$u\big\|_{x=0} = 0; u\big\|_{x=L} = 0$	Fixed support at both ends
$u\big\|_{x=0} = 0; u'\big\|_{x=L} = 0$	Fixed support and free end
$\alpha_0 u\big\|_{x=0} - \beta_0 u'\big\|_{x=0} = 0; u'\big\|_{x=L}$	Spring support and free end
$\alpha_0 u\big\|_{x=0} - \beta_0 u'\big\|_{x=0} = 0; u\big\|_{x=L} = 0$	Spring support and fixed end
$\alpha_0 u\big\|_{x=0} - \beta_0 u'\big\|_{x=0} = 0; \alpha_1 u\big\|_{x=L} + \beta_1 u'\big\|_{x=L} = 0$	Spring support at both ends

where,

$$\frac{\alpha_0}{\beta_0} = \frac{k_0}{EA}\bigg|_{x=0} > 0; \frac{\alpha_L}{\beta_L} = \frac{k_L}{EA}\bigg|_{x=L} > 0$$

k_0, k_L are spring constants

(8)

The objective of this section is to prove the following proposition:

Proposition 1: If the mode shape is valid, then the inverse problem of identifying the area and the eigenvalue from the mode shape, boundary conditions, mass constraint and material properties has a solution (existence) and is unique.

The governing differential equation Eq. (1) is recalled here for the ease of cross-referencing.

$$\frac{d}{dx}\left[EA\frac{du}{dx}\right]+\lambda\rho Au=0 \qquad x\in[0,L]$$

In order to prove this proposition, certain properties of the valid mode shapes for bars are needed and these will be discussed first.

Property 1.1: u and u' cannot become zero (vanish) simultaneously.

If u and u' are simultaneously zero for a homogeneous differential equation, by the theorem of uniqueness of ODE's [14] for initial value problems there can only be a single possible solution and that is of identically zero solution for u which is not of interest to us. So u and u' cannot be zero simultaneously.

Property 1.2: The location where u' vanishes fixes the eigenvalue, and the signs of u'' and u' are opposite at that point to ensure a positive eigenvalue.

Considering Eq. (1) and substituting the condition that the derivative becomes zero and also noting the fact that previous Property implies that when u is non-zero, one can get the following expression for the eigenvalue

$$\lambda=-Eu''/\rho u \tag{9}$$

As it is a valid mode shape, λ must be positive which indicates that u and u'' must have opposite signs at that location to maintain the sign.

Property 1.3: Wherever u' vanishes the eigenvalue prescribed by Eq. (9) has to be the same.

As the mode shape is a valid one, the eigenvalue is unique for a mode.

Property 1.4: There exists at least one point in the domain where u' vanishes.

This is seen by continuity of u and the boundary conditions. For the fixed-free and spring-free condition, the derivative vanishes at the free end. Rolle's Theorem guarantees the existence of at least one such point for the fixed-fixed condition.

The case where spring supports are present at both ends is handled this way. Suppose that u' is of the same sign throughout the interval (an assumption contrary to the intention). Then, owing to the end conditions, $u(0)/u'(0)>0; u(L)/u'(L)<0$, the above two statements imply that u must reverse sign and also that it is monotonic.

Without any loss of generality let $u(0)$ be positive and so $u'(0)$ has to be of the same sign too. Now $u'(L)$ is also positive. Monotonicity of u

implies $u(L) > u(0)$, which is a contradiction to the deduction that u must reverse sign. So u' has to vanish at least at one point. An argument along similar lines yields the same conclusion even for the spring-fixed case.

Property 1.5: Both u and u' possess isolated zeros (i.e. existence of at least a neighborhood around a zero where the function doesn't vanish except at that location of the zero) [15].

If u is zero at a location then u' is not zero (by Property 1.1) and by the continuity of u', one can find a neighborhood where u' is non-zero. This is the neighborhood we are looking for, to prove isolation of zeros of u. If u' is zero then u is non-zero (by Property 1.1). If u is non-zero then by Eq. (1) one can see that u'' is non-zero. Now, again a similar argument proves the isolation of zeros of u' too.

Property 1.6: The quantity $-\left(\left(Eu'\right)' + \lambda\rho u'\right)\Big/ Eu'$ has a limit at the point where $u' = 0$ and is continuous in the rest of the domain.

The quantity mentioned above is A'/A by Eq. (1) at locations where u' is non-zero. Now, this quantity is continuous everywhere in the domain. Taking the neighborhood around u' where it is non-zero (by Property 1.5) and the continuity of A'/A, we notice that the limit of the given quantity exists.

Now we are ready to prove the proposition which aims at constructing a non-zero solution to the inverse problem.

Proof of Proposition 1: λ can be found given a valid mode shape by Property 1.2 and 1.3. Now in Eq. (1), A is the unknown variable. Hence, we re-write the equation wherever u' is not zero in a straightforward manner. When there is a singularity, (i.e., u' is zero), we take the limit of the term which is guaranteed to exist by Property 1.6.

$$\frac{A'}{A} = \psi(x) = \begin{cases} -\left(\left(Eu'\right)' + \lambda\rho u'\right)\Big/ Eu' & u' \neq 0 \\ \lim\limits_{u'\to 0} -\left(\left(Eu'\right)' + \lambda\rho u'\right)\Big/ Eu' & u = 0 \end{cases} \tag{10}$$

The right hand side of Eq. (10) is a continuous function on the domain (by Property 1.6) and hence can be integrated over the domain to yield the following solution

$$A = Ce^{\int \psi(x)dx} \tag{11}$$

The mass constraint fixes the arbitrary constraint C. Hence this is a solution of the problem thereby guaranteeing the existence. Now for the uniqueness, suppose that there are two solutions to the above problem. Both will have to

obey Eq. (10) which means that both may differ only by a scaling constant which is fixed by the mass constraint and provides us with the desired uniqueness if the mode shape is valid. Property 1.2, 1.3, and 1.4 guarantee uniqueness of the eigenvalue. This proves the proposition.

7. The Case of the Cantilever (Linear)

Consider the Eq. (2) where the moment of inertia is linearly proportional to the area. We recall it here for convenience.

$$\frac{d^2}{dx^2}\left(EA\frac{d^2u}{dx^2}\right)-\lambda\rho Au=0 \qquad x\in[0,L]$$

In the case of the cantilever the boundary conditions are:

$$\begin{aligned} u(0)=0;u'(0)=0 \\ u''(L)=0;u'''(0)=0 \end{aligned} \tag{12}$$

Let us again begin by defining what a valid mode shape is in this case.

A valid mode shape is one that arises out as a solution of Eq. (2) where the following conditions are built in:

1. Young's Modulus is strictly positive and C^2continuous.
2. Area is strictly positive and C^2 continuous.
3. Density is strictly positive and C^0 continuous.
4. Boundary Conditions as specified by Eq. (12).

Our intention is to prove the following proposition

Proposition 2: If the mode shape is valid then the inverse problem of identifying the area profile given the mode shape and total mass is unique.

The proof is based on the following theorem (Theorem 13.3.2 in Pg 382) stated in a book by Gladwell [4].

Theorem: If $\phi_i(x)$ is an eigenfunction of a cantilever beam then $\phi_i(1)\phi_i'(1)>0$.

This theorem is based on the Beam equation formulated in the following manner

$$\left(r(x)u''(x)\right)''=\lambda a(x)u(x) \qquad x\in[0,1] \tag{13}$$

The reduction to this form is done in the following manner. Consider Eq. (2) and the following change of variables:

$$x = Ls; u(s) = u(x); r(s) = \frac{E(x)I(x)}{E(x_c)I(x_c)}; a(s) = \frac{A(x)\rho(x)}{A(x_c)\rho(x_c)}; \omega = \frac{\lambda A(x_c)\rho(x_c)L^4}{E(x_c)I(x_c)}$$

This results in the following equation which is indeed the one that we have in Eq. (13).

$$\left(r(s)u''(s)\right)'' = \omega a(s)u(s) \qquad s \in [0,1] \tag{14}$$

Property 2.1: This proves that at the free end of the cantilever, displacement or slope cannot be zero for a mode shape and hence, the eigenvalue gets fixed there as:

$$\lambda = \frac{\left(Eu''\right)''}{\rho u} \tag{15}$$

Property 2.2: The zeros of u'' are isolated.

Let us suppose that the zeros of u'' aren't isolated. That is, there is a neighborhood around the zero where u''' and u^{iv} are also zero. By Eq. (2) one can see that u also has to vanish in that region meaning u' also has to vanish. This implies that for the homogeneous differential equation by the theorem of uniqueness, the only possible solution is a mode shape that is zero everywhere which we are not interested in. This implies that the zeros of u'' have to be isolated.
The proof for the Proposition is described below.

Proof of Proposition 2: Let us suppose that there exist two different area profiles A_1 and A_2 both positive and C^2 continuous as a solution for the same mode shape u. The eigenvalue has to be same for both the area profiles as it gets fixed at the free end of the cantilever (Property 2.1). We rewrite the differential equations for both the profiles and expand them as follows.

$$\left(Eu''\right)A_1'' + 2\left(Eu''\right)'A_1' + \left(\left(Eu''\right)'' - \lambda\rho u\right)A_1 = 0 \tag{16}$$

$$\left(Eu''\right)A_2'' + 2\left(Eu''\right)'A_2' + \left(\left(Eu''\right)'' - \lambda\rho u\right)A_2 = 0 \tag{17}$$

Multiply Eq. (16) with A_2 and Eq. (17) with A_1 and subtracting the result (with an intention to cancel out the last term), one is left with the following equation.

$$\left(Eu''\right)\left(A_2A_1'' - A_1A_2''\right) + 2\left(Eu''\right)'\left(A_2A_1' - A_1A_2'\right) = 0 \tag{18}$$

This equation has terms possessing the Wronskian structure. We replace $A_2A_1' - A_1A_2'$ by $W(x)$ which is indeed C^1 continuous on the domain and on

differentiating it yields $W'(x) = A_2 A_1'' - A_1 A_2''$ (which is precisely what is there on the left end).

$$\left(Eu''\right)W'(x) + 2\left(Eu''\right)' W(x) = 0 \tag{19}$$

Now this is rewritten as

$$\left(\left(Eu''\right)^2 W\right)' = 0 \tag{20}$$

This implies that

$$\left(\left(Eu''\right)^2 W\right) = Const \tag{21}$$

We evaluate this constant at the free end of the cantilever where the value of u'' is zero. Hence, the constant is zero and it means that wherever u'' is non-zero (existence of such regions is offered by Property 2.2) W has to be zero. This implies that $A_2 A_1' - A_1 A_2' = 0$ which means that A_2 and A_1 are linearly dependent in the regions other than the zeros of u''. By continuity conditions on A it must be linearly dependent everywhere. Here the mass constraint plays a role in fixing the scaling and hence provides uniqueness.

8. Extensions to the Numerical Algorithm

8.1 Need for the Extension

Let us consider the case of the bars where we noticed that when a valid mode shape such as $1-\left(x-1\right)^2$ (fixed free boundary condition) is provided to the algorithm it does not provide the correct solution ($A = Ce^{x-0.5x^2}$ with eigenvalue $2E/\rho$) as shown in Figs. 16 and 17.

When the actual area is discretized and supplied to an FE routine which computes the first eigenmode, the error between the analytically obtained mode and the result is of the order of 1e-4% (Relative error $\|U_{disc} - U_{ana}\|_2 / \|U_{ana}\|_2$). The relative error in eigenvalue is 0.22%. But when the discretized version is supplied one gets the accurate results. The conclusion of this experiment is that a better interpolation for the area is required.

8.2 Swapping Under Higher Interpolation Schemes

During the choice of the higher interpolation, the rank of the resulting rectangular system (swapped system) needs to be equal to the number of geometric unknown parameters. The original forward square linear system always has a dimension equal to that of the no. of free dof for the displacement. The

result after swapping has a dimension of number of free dof for the displacement times the number of dof for the geometric term (i.e., the c/s profile).

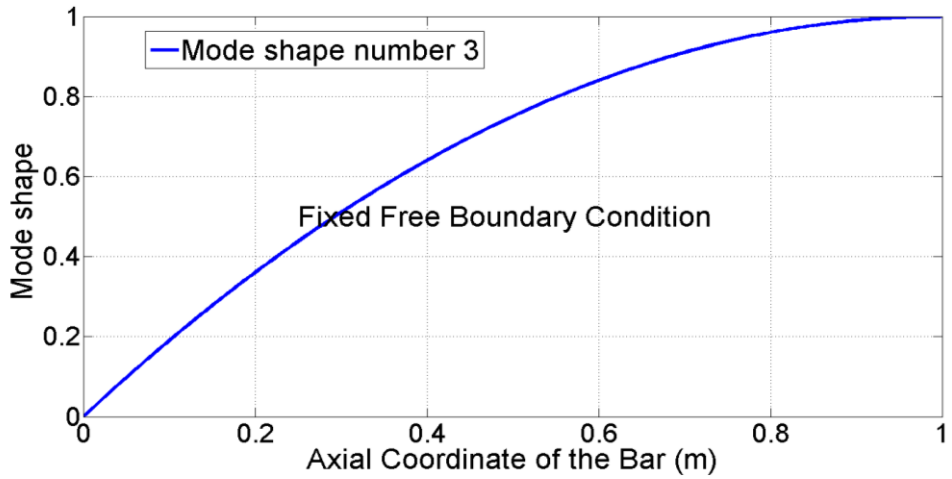

Fig 16: The first mode shape for the fixed-free boundary conditions with 150 three-noded elements given by $1-(x-1)^2$ m. The actual profile desired is $A = e^{x-0.5x^2} \times 1e-6m^2$ with an eigenvalue of 5.3846e7. The mass constraint is 0.0110 kg.

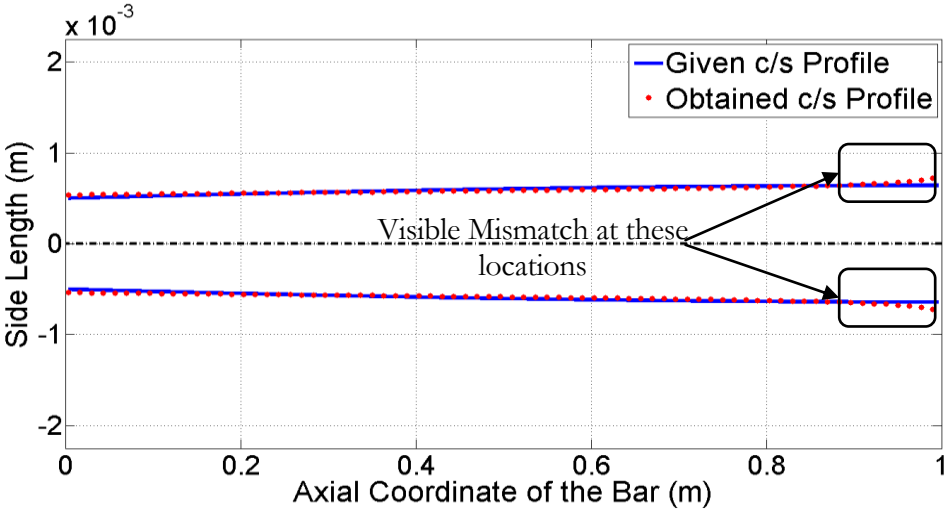

Fig 17: The obtained area profile has an error of 0.64 % with the actual one and the error in eigenvalue is 27.03%.

This system must be constrained or overconstrained to obtain a solution. So, this must be kept in mind when selecting the different interpolations schemes for the displacement variable and the geometric term.

To understand swapping in such a system, an useful example (and indeed the path of discovery) would be the formulation of the element stiffness matrix in the case of a three-noded bar element with a linear interpolation for the area (two dof). Let the shape function vector (assumed to be columnar) for the displacement profile be given by N_1 and the shape function column (assumed to be columnar) for the area profile be given by N_2. The element stiffness matrix here is given by

$$\int_{element} \left(\left\langle \frac{dN_1}{dx} \right\rangle \left\langle \frac{dN_1}{dx} \right\rangle^T \langle N_2 \rangle^T \left\langle \begin{matrix} A_1 \\ A_2 \end{matrix} \right\rangle \right) dx = \sum_{\text{gauss points}} w_{\text{gauss point}} \left\langle \frac{dN_1}{dx} \right\rangle \left\langle \frac{dN_1}{dx} \right\rangle^T \langle N_2 \rangle^T \left\langle \begin{matrix} A_1 \\ A_2 \end{matrix} \right\rangle \tag{22}$$

This can be further written in indicial notation and simplified as follows:

$$\begin{aligned} &\sum_{\text{gauss points}} w_{\text{gauss point}} \left\langle \frac{dN_1}{dx} \right\rangle \left\langle \frac{dN_1}{dx} \right\rangle^T \langle N_2 \rangle^T \left\langle \begin{matrix} A_1 \\ A_2 \end{matrix} \right\rangle \\ &= \sum_{\text{gauss points}} w_{\text{gauss point}} \left(\frac{dN_1}{dx} \right)_i \left(\frac{dN_1}{dx} \right)_j \left(N_{2_1} A_1 + N_{2_2} A_2 \right) \\ &= A_1 \sum_{\text{gauss points}} w_{\text{gauss point}} \left(\frac{dN_1}{dx} \right)_i \left(\frac{dN_1}{dx} \right)_j N_{2_1} + A_2 \sum_{\text{gauss points}} w_{\text{gauss point}} \left(\frac{dN_1}{dx} \right)_i \left(\frac{dN_1}{dx} \right)_j N_{2_2} \end{aligned} \tag{23}$$

The result of Eq. (23) can be pictorially summarized as follows in Fig. 18.

$$\mathbf{A_1} \begin{pmatrix} \\ \mathbf{Block_1} \\ \\ \end{pmatrix} + \mathbf{A_2} \begin{pmatrix} \\ \mathbf{Block_2} \\ \\ \end{pmatrix}$$

Fig 18: Pictorial Depiction of the element matrix formulation in FEM

We need to find Block_1 and Block_2 to perform a swap. To find Block_1 one can just supply (1, 0) as the input for the areas in the element matrix subroutine call and similarly (0, 1) as input for the areas in the element matrix subroutine call for Block_2. The assembly during the formation of the stiffness matrix proceeds in the manner illustrated in the Fig. 19a and the result of swapping is shown in Fig. 19b.

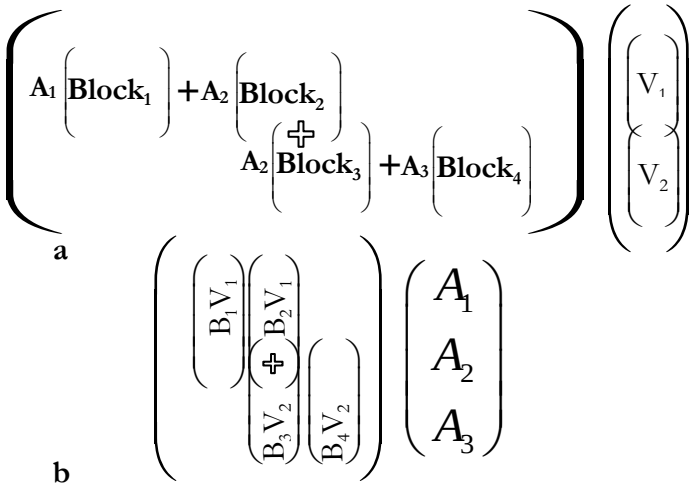

Fig 19:a. Depiction of FEM Assembly b. Result of Swapping displacements and Areas

During the assembly operations, each of the element stiffness matrices are placed at the element dof and assembled additively. During swapping, element blocks multiply the corresponding element displacement vectors and are assembled by adding.

Once the swapping is done the rest of the procedure is the same and the results are shown in Fig. 20 for the same mode shape as the previous one where, that interpolation had failed to provide a correct solution. This method can be extended easily to other systems.

This method also proves that the artifact is indeed due to the interpolation. The example where the artifact arose is rerun with the synthetic data generated using the interpolated area and the result is shown in Fig. 21.

8.3 Numerical Test for the Validity of a Mode Shape

If the algorithm doesn't converge at all then one can conclude that the mode shape is invalid. In case it produces a result one shouldn't stop at this stage and must carry on to estimate the residual which gives a numerical test for the validity of the mode shape. The procedure is described below.

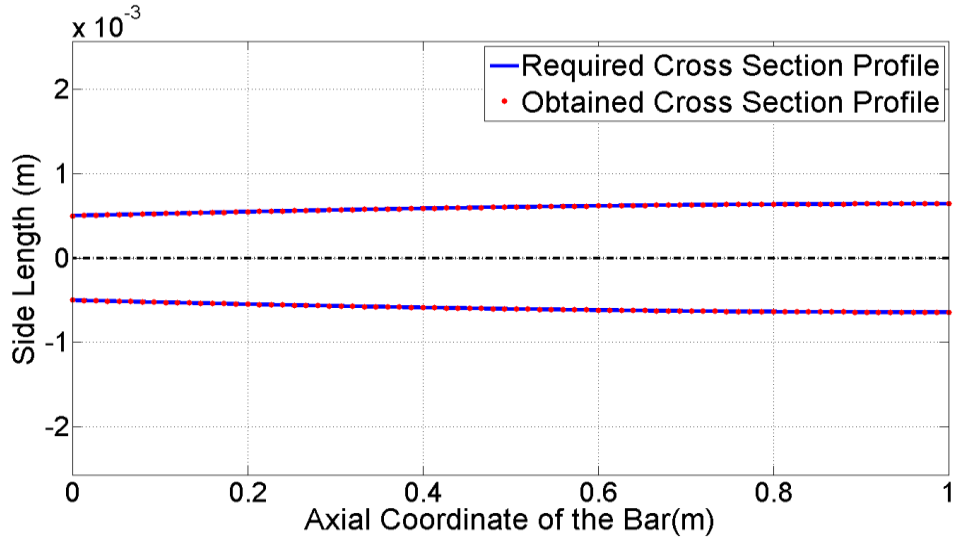

Fig 20: A linear interpolation for the c/s parameter is assumed in the three-noded element. The error in the c/s profile and the eigenvalue is of the order 1e-4%.

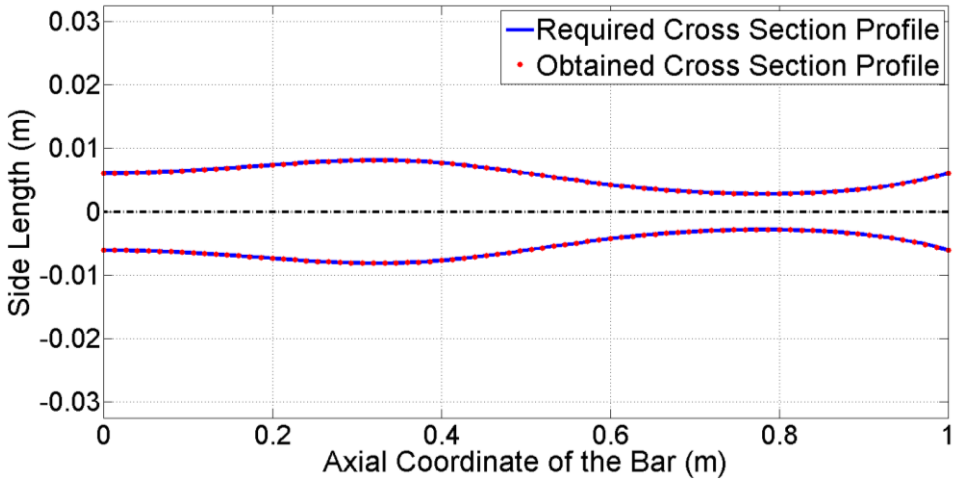

Fig 21: The original c/s profile for the FixedFixed boundary condition is exactly achieved when the third mode is supplied to it. The relative error in the profile is of the order of 1e-9 % and the relative error in frequency is of the order of 1e-11%.

1. One finds the solution of the rectangular system $\boldsymbol{K}_{disp}\boldsymbol{g}^{n} = \lambda \boldsymbol{M}_{disp}\boldsymbol{g}$ after converting it by multiplying it with its transpose to obtain a square system $\boldsymbol{K}^{*}\boldsymbol{g}^{n} = \lambda \boldsymbol{M}^{*}\boldsymbol{g}$.

2. The solution of this problem is then substituted back into the original rectangular system and the residual is computed as $\text{Residual} = \left\| \boldsymbol{K}_{disp}\boldsymbol{g}^{n} - \lambda \boldsymbol{M}_{disp}\boldsymbol{g} \right\|_2$.
3. If the residual for a given mode shape is below tolerance limits then we conclude that the mode shape is numerically valid corresponding to that interpolation scheme and the number of elements taken.

The Residual is not computed in the straightforward manner as described owing to round off error becoming large. So, it is computed using the following formula $\text{Residual} = \left\| \boldsymbol{K}_{disp}\boldsymbol{g}^{n} / \lambda - \boldsymbol{M}_{disp}\boldsymbol{g} \right\|_2$. This was computed in the case depicted in Figs. 14 and 15 and it was of the order of 1e2 which was higher than our tolerance value of 1e-4. So, it can be concluded that the mode shape is numerically invalid with respect to that interpolation scheme and the number of elements taken.

9. Conclusion and Future Work

The work builds a solution methodology in the finite element framework for the case of bars, beams, and plates to solve the inverse mode shape problem. It has also succeeded in identifying the need for a mode shape to be valid and analytical study has been done in the case of bars where the existence, uniqueness and construction procedure was developed. The uniqueness of the solution has also been proven for the case of the cantilever beam in the linear form of the problem. A numerical validation procedure has also been developed; if the algorithm provides a solution, it needn't mean that the mode shape is valid and a residual needs to be computed. Furthermore, the basic numerical technique is extended to provide for the accuracy needed to specify the mode shape.

The goal of achieving a design for a mode shape is still away and we need to take both the analyst's direction of providing the conditions for a valid mode shape and the engineer's line of providing a solution closest to the desired shape. A direction that could be taken to further the engineer's cause would be to use an optimization framework where minimization of the residual could be a possible objective.

10. CODES

Codes were written in Matlab 2007b and are arranged in the path of development.

10.1 Bars

The following routine is for the finite element solution of the free vibration of a bar with a one dof per node, three noded element, and a constant area per element for the eigenmode and the eigenvalue.

```
function [Modeshape,Lambda]=fp(L,E,Rho,A,Modeno,Fixeddof)
  % Formation of Element Matrices
  n=max(size(E));
  t_dof=2*n+1;
  Le=L/n;
  %Mass Matrix
  Me = [  4  2 -1
           2  16 2
          -1  2  4 ];

   Me = Me*Le/30;

   %Stifness Matrix
   Ke = [  7  -8  1
          -8  16 -8
            1  -8  7 ];

   Ke = Ke/(3*Le);

   % Assembly
   M = sparse(t_dof,t_dof);
   K = sparse(t_dof,t_dof);

   for ele = 1:n
      edof = [2*ele-1 2*ele 2*ele+1];
      K(edof,edof)=K(edof,edof)+E(ele)*Ke*A(ele);
      M(edof,edof)=M(edof,edof)+Rho(ele)*Me*A(ele);
   end

   %Applying Boundary Conditions
   Alldof=1:t_dof;
   Freedof=setdiff(Alldof,Fixeddof);

   %Finding the Eigen Values and the Eigen Vectors
   options.tol=1e-9;
   options.issym=1;
   options.isreal=1;
```

```
 options.disp=0;
 U=zeros(t_dof,Modeno);

 [U(Freedof,:),D]=eigs(K(Freedof,Freedof),M(Freedof,Freedof),Modeno…
 ,0,options);
Lambda=D(1,1);
%Normalisation
[val,loc]=max(abs(U(:,1)));
Modeshape=U(:,1)/U(loc,1);

end
```

The next routine is for the solution of the inverse mode shape problem of a bar with a 3-noded element and a constant area per element which provides the area profile and the eigenvalue.

```
function [A,Lambda ,Num]=ip(L,E,Rho,Modeshape,Fixeddof)
 n = max(size(E));
 t_nod=2*n+1;
 Le=L/n;
 %Element Matrices
 %Mass Matrix
Me = [  4  2 -1
          2  16 2
         -1  2  4 ];

Me = Me*Le/30;

%Stifness Matrix
Ke = [  7  -8  1
       -8  16 -8
         1  -8  7 ];

Ke = Ke/(3*Le);

%Assembly
K=sparse(t_nod,n);
M=sparse(t_nod,n);
for ele = 1 : n
   edof=[2*ele-1 2*ele 2*ele+1];
   K(edof,ele)=E(ele)*Ke*Modeshape(edof);
   M(edof,ele)=Rho(ele)*Me*Modeshape(edof);
end

Alldof=1:t_nod;
Freedof=setdiff(Alldof,Fixeddof);

%Multiplying by Transpose
Pseudo_K=K(Freedof,:)'*K(Freedof,:);
Pseudo_M=K(Freedof,:)'*M(Freedof,:);
```

```
  % The Generalised Eigen Value Problem
  [Eig_Vec,Eig_Val]=eig(full(Pseudo_K),full(Pseudo_M),'chol');
  %Post Processing the Area
  %Obtaining the EigenValues and EigenVectors that are meaningful
  %Check : Real,Everywhere positive sign after normalization
  Num=0;
  for i = 1:n
     if(isreal(Eig_Vec(:,i)) && isreal(Eig_Val(i,i)))
        [val,loc] = max(abs(Eig_Vec(:,i)));
        Eig_Vec(:,i) = Eig_Vec(:,i)/Eig_Vec(loc,i);
        flag=0;
        for j = 1:n
           if( Eig_Vec(j,i) <  1e-7)
                    flag=-1;
              break;
           end
        end
        if(flag==0)
           Num=Num+1;
           A(:,Num)=Eig_Vec(:,i);  %#ok<AGROW>
           Lambda(Num)=Eig_Val(i,i); %#ok<AGROW>
        end
     end
  end
  if(Num==0)
     A=0;
     Lambda=0;
  end
end
```

A sample code indicating how the two routines were used for testing the algorithm is shown below.

```
clc;
clear;
close all;

%Initializations
n=150;                %Number of Elements
L=1;                  %Length of the Rod
t_nod=2*n+1;

%FixedFree Case
Fixeddof=1;

inter=L/n;
y=linspace(inter/2,L-inter/2,n)';

E=210e9*ones(n,1);     %Young's Modulus
```

```
Rho=7800*ones(n,1);   %Density
Modeno=3;             %Modeshape found

A = exp(5 + 2*y.*sin(2*pi*y/L))*1e-6;
Mass=sum(Rho*inter.*A);
normA=A/max(A);

% Generating Synthetic Data
[Modeshape,Lambda]=fp(L,E,Rho,normA,Modeno,Fixeddof);
% Solving the Inverse Mode Shape Problem
[normI_A,I_Lambda,I_Num]=ip(L,E,Rho,Modeshape,Fixeddof);
%Fixing the Scaling of the c/s profile
I_A=(Mass/sum(Rho*inter.*normI_A))*normI_A;
```

10.2 Euler Bernoulli Beams

This routine was written for the FE solution of the forward problem to obtain the eigenmode and eigenvalue. The element used has 2 dof per node and it is a two node element with constant area.

```
function [Modeshape,Lambda]=fp(E,Rho,I,A,L,Modeno,Fixeddof)
    n = max(size(E));
    Le=L/n;
    %Element Matrices
    %Mass
    Me =  [13/35)*Le (11/210)*(Le^2) (9/70)*Le (-13/420)*(Le^2);
           (11/210)*(Le^2) (1/105)*(Le^3) (13/420)*(Le^2) (-1/140)*(Le^3) ;
           (9/70)*Le (13/420)*(Le^2) (13/35)*Le (-11/210)*(Le^2);
           (-13/420)*(Le^2) (-1/140)*(Le^3) (-11/210)*(Le^2) (1/105)*(Le^3) ];

    % Stiffness
    Ke   = [12/(Le^3) 6/(Le^2) -12/(Le^3) 6/(Le^2);
           6/(Le^2) 4/Le -6/(Le^2) 2/Le ;
           -12/(Le^3) -6/(Le^2) 12/(Le^3 ) -6/(Le^2);
           6/(Le^2) 2/Le -6/(Le^2) 4/Le ];

    %Assembly
    M = sparse(2*(n+1),2*(n+1));
    K = sparse(2*(n+1),2*(n+1));
    for ele =1:n
           edof=2*ele-1:2*(ele+1);
           K(edof,edof) = K(edof,edof) + E(ele)*I(ele)*Ke;
           M(edof,edof) = M(edof,edof) + Rho(ele)*Me*A(ele);
    end

    % Applying Boundary Conditions
    Alldof=1:2*(n+1);
    Freedof=setdiff (Alldof,Fixeddof);

    %Finding the Eig_val and Eig_vec
```

```
    options.tol=1e-9;
    options.issym=1;
    options.isreal=1;
    options.disp=0;
    U=zeros(2*(n+1),Modeno);
    [U(Freedof,:),D]=eigs(K(Freedof,Freedof),M(Freedof,Freedof),Modeno,…
    0,options);
    Lambda=D(1,1);

    %Normalisation
    [val,loc]=max(abs(U(:,1)));
    Modeshape = U(:,1)/U(loc,1);

end
```

The next routine is for the inverse routine when the c/s profile is linear in the equation.

```
function [A,Lambda,Num] = ip(E,Rho,Modeshape,L,Fixeddof)
    n = max(size(E));
    U=Modeshape;
    Le=L/n;
    %Element Matrices
    %Mass
    Me =  [13/35)*Le (11/210)*(Le^2) (9/70)*Le (-13/420)*(Le^2);
           (11/210)*(Le^2) (1/105)*(Le^3) (13/420)*(Le^2) (-1/140)*(Le^3) ;
           (9/70)*Le (13/420)*(Le^2) (13/35)*Le (-11/210)*(Le^2);
           (-13/420)*(Le^2) (-1/140)*(Le^3) (-11/210)*(Le^2) (1/105)*(Le^3) ];

    % Stiffness
    Ke   = [12/(Le^3) 6/(Le^2) -12/(Le^3) 6/(Le^2);
                  6/(Le^2) 4/Le -6/(Le^2) 2/Le ;
                  -12/(Le^3) -6/(Le^2) 12/(Le^3 ) -6/(Le^2);
                  6/(Le^2) 2/Le -6/(Le^2) 4/Le ];

    %Assembly
    K=sparse(2*(n+1),n);
    M=sparse(2*(n+1),n);
    for ele = 1:n
          edof=2*ele-1:2*ele+2;
          K(edof,ele)=Ke*E(ele)*U(edof);
          M(edof,ele)=Me*Rho(ele)*U(edof);
    end

    % Applying Boundary Conditions
    Alldof=1:2*(n+1);
    Freedof=setdiff (Alldof,Fixeddof);

    % Multiplying by the Transpose
    pseudoK=K(Freedof,:)'*K(Freedof,:);
```

```
    pseudoM=K(Freedof,:)'*M(Freedof,:);

    % The Generalised Eigen Value Problem
    [Eig_Vec,Eig_Val]=eig(full(pseudoK),full(pseudoM));

    %Post Processing the Area
    %Obtaining the EigenValues and EigenVectors that are meaningful
    %Check : Real,Homogeneous positive sign after normalization
    Num=0;
    for i = 1:n
        if( isreal(Eig_Vec(:,i)) && isreal(Eig_Val(i,i)) && (Eig_Val(i,i)>0) )
        [val,loc] = max(abs(Eig_Vec(:,i)));
            Eig_Vec(:,i) = Eig_Vec(:,i)/Eig_Vec(loc,i);
            flag=0;
            for j = 1:n
                if( Eig_Vec(j,i) < 1e-5 )
                flag=-1;
                break;
                end
            end
            if(flag==0)
                Num=Num+1;
                A(:,Num)=Eig_Vec(:,i);  %#ok<AGROW>
                Lambda(Num)=Eig_Val(i,i); %#ok<AGROW>
            end
        end
    end

    if(Num==0)
        A=0;
        Lambda=0;
    end
end
```

The routine described below is for the inverse routine when the c/s profile is non-linear in the equation. This routine is an implementation of the iterative algorithm

```
function [A,Lambda,iter]=ip(E,Rho,Modeshape,L,…
Fixeddof,Mass,Maxiter,power,tol)
    n = max(size(E));
    U=Modeshape;
    Le=L/n;
    %Element Matrices
    %Mass
    Me =  [13/35)*Le (11/210)*(Le^2) (9/70)*Le (-13/420)*(Le^2);
        (11/210)*(Le^2) (1/105)*(Le^3) (13/420)*(Le^2) (-1/140)*(Le^3) ;
        (9/70)*Le (13/420)*(Le^2) (13/35)*Le (-11/210)*(Le^2);
        (-13/420)*(Le^2) (-1/140)*(Le^3) (-11/210)*(Le^2) (1/105)*(Le^3) ];
    % Stiffness
```

```
    Ke   = [12/(Le^3) 6/(Le^2) -12/(Le^3) 6/(Le^2);
                    6/(Le^2) 4/Le -6/(Le^2) 2/Le ;
                    -12/(Le^3) -6/(Le^2) 12/(Le^3 ) -6/(Le^2);
                    6/(Le^2) 2/Le -6/(Le^2) 4/Le ];

    %Assembly
    K=sparse(2*(n+1),n);
    M=sparse(2*(n+1),n);
    for ele = 1:n
            edof=2*ele-1:2*ele+2;
            K(edof,ele)=Ke*E(ele)*U(edof);
            M(edof,ele)=Me*Rho(ele)*U(edof);
    end

    % Applying Boundary Conditions
    Alldof=1:2*(n+1);
    Freedof=setdiff (Alldof,Fixeddof);

    %Multiplying by Transpose
    psK=K(Freedof,:)'*K(Freedof,:);
    psM=K(Freedof,:)'*M(Freedof,:);
    psR=psK\psM;

    % The Iterations
    A=(ones(n,1));
    A=A/max(A);
    change=1.;
    iter=0;
    I=[];
    while(change>tol && iter<Maxiter)
            iter=iter+1;
            Aold=A;
            I=psR*A;
            I=abs(I/max(abs(I)));
            A=nthroot(I,power);
            change=norm(abs(A-Aold),2);
    end
    C=(Mass/sum(Rho*Le.*A));
    Lambda=(C^(power-1))*I'*I/(I'*psR*A);
    A=C*A;
end
```

These routines were utilized to verify the validity of the algorithm as shown below.

```
clc;
clear;
%Initialization
n=100;                  %Number of Elements
L=1;                    %Length of the Beam
```

```
tol=1e-14;%Tolerance

inter=L/n;
y=linspace(inter/2,L-inter/2,n)';

E=210e9*ones(n,1);            %Young's Modulus
Modeno=4;                     %Modeshape found
Rho=7800*ones(n,1);           %Density
Maxiter=1000;
power=2;

A = exp(2*y)*1e-6;
I = A.^power;

Mass=sum(Rho.*(L/n).*A);
normA=A/max(A);

%Boundary Conditions
%Cantilever
Fixeddof=[1,2];

%Generation of Synthetic Data
[Modeshape,Lambda]=fp(E,Rho,I,A,L,Modeno,Fixeddof);
%Solving the Inverse Problem
[I_A,I_Lambda,iter]=ip(E,Rho,Modeshape,L,Fixeddof,Mass,Maxiter,power,tol);
```

10.3 Plates

This FE routine was written to obtain the eigenvalue and eigenmode for the forward problem of the kirchoff plate. The element taken was a four noded, three degree of freedom per node one. The thickness is assumed constant per element.

```
function [Modeshape,Lambda,Error]=fp(E,lx,ly,nu,Rho,t,Modeno,fixeddofs)

    [nely,nelx]=size(t);
    t_dof=(nely+1)*(nelx+1)*3;
    le=lx/nelx;
    if(le~=(ly/nely))
        Error=0;
        Modeshape=0;
        Lambda=0;
    else
        Error=1;
        %Element Matrices
        %Stiffness
        [Ke Me]=elemat(nu,le);
        %Assembly
        M=sparse(t_dof,t_dof);
```

```
            K=sparse(t_dof,t_dof);
            for ely=1:nely
                for elx=1:nelx
                    n1=(elx-1)*(nely+1)+ely;
                    n2=elx*(nely+1)+ely;
                    el=(elx-1)*nely+ely;
                    edof=[3*n1-2 3*n1-1 3*n1 3*n2-2 3*n2-1 3*n2…
                    3*(n2+1)-2 3*(n2+1)-1 3*(n2+1) 3*(n1+1)-2 …
                    3*(n1+1)-1 3*(n1+1)];
                    K(edof,edof)=K(edof,edof)+E(ely,elx)*Ke*t(ely,elx)…
                    t(ely,elx)*t(ely,elx);
                    M(edof,edof)=M(edof,edof)+Rho(ely,elx)*Me*t(ely,elx);
                end
            end
            %Boundary Condition
            alldofs=1:t_dof;
            freedofs=setdiff(alldofs,fixeddofs);
            %Finding the Eigen Values and the Eigen Vectors
            options.tol=1e-9;
            options.issym=1;
            options.isreal=1;
            options.disp=0;
            U=zeros(t_dof,Modeno);
            [U(freedofs,:),D]=eigs(K(freedofs,freedofs),M(freedofs,freedofs),…
            Modeno,0,options);
            Lambda=D(1,1);
            %Normalisation
            [val,loc]=max(abs(U(:,1)));
            Modeshape=U(:,1)/U(loc,1);
        end
end
```

The inverse problem is solved using an iterative solution scheme as the variable (thickness) has a power of 3 in the stiffness matrix.

```
function [t,Lambda,Error,Iter]=ip(E,lx,ly,nu,Rho,fixeddofs,Modeshape,Mass)

    [nely,nelx]=size(E);
    t_dof=(nely+1)*(nelx+1)*3;
    t_nel=(nely*nelx);
    le=lx/nelx;
    if(le~=(ly/nely))
        Error=0;
        Modeshape=0;
        Lambda=0;
    else
        Error=1;
        U=Modeshape;
        %Element Matrices
        %Stiffness
```

```
            [Ke Me]=elemat(nu,le);
            %Assembly
            M=sparse(t_dof,t_nel);
            K=sparse(t_dof,t_nel);
            for ely=1:nely
                for elx=1:nelx
                    n1=(elx-1)*(nely+1)+ely;
                    n2=elx*(nely+1)+ely;
                    el=(elx-1)*nely+ely;
                    edof=[3*n1-2 3*n1-1 3*n1 3*n2-2 3*n2-1 3*n2…
                    3*(n2+1)-2 3*(n2+1)-1 3*(n2+1) 3*(n1+1)-2 …
                    3*(n1+1)-1 3*(n1+1)];
                    K(edof,el)=K(edof,edof)+E(ely,elx)*Ke*U(edof);
                    M(edof,el)=M(edof,edof)+Rho(ely,elx)*Me*U(edof);
                end
            end
            %Boundary Condition
            alldofs=1:t_dof;
            freedofs=setdiff(alldofs,fixeddofs);
            %Multiplying by Transpose
            pseudoK=K(freedofs,:)'*K(freedofs,:);
            pseudoM=K(freedofs,:)'*M(freedofs,:);
            %Iterations
            resMat=inv(pseudoK)*pseudoM;
            maxiter=10001;
            t1=abs(rand(t_nel,1));
            t1=t1/max(t1);
            change=1.;
            Iter=0;
            while(change>1e-14 && Iter <maxiter)
                Iter=Iter+1;
                t1old=t1;
                t3=resMat*t1;
                t3=t3/max(t3);
                t1=abs(nthroot(t3,3));
                change=norm(abs(t1-t1old),2);
            end
            Lambda=t3'*t3/(t3'*resMat*t1);
            t=zeros(nely,nelx);
            for elx=1:nelx
                t(:,elx)=t1((elx-1)*nely+1:elx*nely);
            end
            C=(Mass/sum(Rho(:).*t*le*le));
            Lambda=(C^2)*Lambda;
            t=t*C;
        end
    end
```

10.4 Bars – c/s Variables Interpolated

Here the element is still the three noded element but the c/s variables are also interpolated and the shape functions for them are taken too. The routine to obtain the element matrices is first shown below.

```
function [eK eM]=eleMat(E,Rho,A,ShFun2Pt,ShFun3Pt,le)

        Me1=(ShFun3Pt.N_1*ShFun3Pt.N_1')*(ShFun3Pt.NA_1'*A);
        Me2=(ShFun3Pt.N_2*ShFun3Pt.N_2')*(ShFun3Pt.NA_2'*A);
        Me3=(ShFun3Pt.N_3*ShFun3Pt.N_3')*(ShFun3Pt.NA_3'*A);

        eM=ShFun3Pt.Weight(1)*Me1+ShFun3Pt.Weight(2)*Me2+…
        ShFun3Pt.Weight(3)*Me3;
        eM=Rho*eM*le/2;

        Ke1=(ShFun2Pt.dN_1*ShFun2Pt.dN_1')*(ShFun2Pt.NA_1'*A);
        Ke2=(ShFun2Pt.dN_2*ShFun2Pt.dN_2')*(ShFun2Pt.NA_2'*A);

        eK=ShFun2Pt.Weight(1)*Ke1+ShFun2Pt.Weight(2)*Ke2;
        eK=E*eK*le/2;
end
```

The function to evaluate the shape function and its derivatives for both the displacements and the c/s variable at a location is described below.

```
function [N,dN,NA]=ShFun(le,loc)
        N=[0.5*loc*(loc-1);-loc*loc+1;0.5*loc*(loc+1)];
        dN=2/le*[loc-0.5;-2*loc;loc+0.5];
        NA=0.5*[-loc+1;loc+1];
end
```

The function to solve the forward problem finite element problem for the eigenmode and the eigenvalue is the next one.

```
function [Modeshape Lambda]=fp(ShFun2Pt,ShFun3Pt,E,Rho,A,L,n,…
Fixeddofs,Modeno)

%INPUT
%ShFun2pt,ShFun3pt       -   Structure Containing the evaluation of
%               Trans(N)*N Matrices (Mass) and the Trans(N')*N'
%             Matrices (Stiffness) evaluated at the gaussian points.
%               Also provided is the evaluation of the shape functions
%               for the area at the gaussian points.It contains the weights too.
%E                      -   Youngs Modulus that is a constant for the Element
%Rho                    -   Density that is constant for the Element
%A                      -       Area which is provided at the end nodes of the three
%                               noded element
```

```
%L              - Length of the Bar
%n              - Number of three noded elements
%Fixeddofs          - The fixed degrees of freedom
%Modeno             - The Mode desired

le=L/n;
totdofs=2*n+1;
%Assembly
gK=sparse(totdofs,totdofs);
gM=sparse(totdofs,totdofs);
for ele =1 :n
   eledofs=2*ele-1:2*ele+1;
   eleAdofs=ele:ele+1;
   [eK,eM]=eleMat(E(ele),Rho(ele),A(eleAdofs),ShFun2Pt,ShFun3Pt,le);
   gK(eledofs,eledofs)=gK(eledofs,eledofs)+eK;
   gM(eledofs,eledofs)=gM(eledofs,eledofs)+eM;
end

%Boundary Condition
Alldofs=1:totdofs;
Freedofs=setdiff(Alldofs,Fixeddofs);
%Generalized Eigenvalue Problem
options.tol=1e-9;
options.issym=1;
options.isreal=1;
options.disp=0;
U=zeros(totdofs,Modeno);
[U(Freedofs,:),D]=eigs(gK(Freedofs,Freedofs),gM(Freedofs,Freedofs),…
Modeno,0,options);
%Normalisation
[val,loc]=max(abs(U(:,1)));
U(:,1)=U(:,1)/U(loc,1);
Lambda=D(1,1);
Modeshape=U(:,1);
end
```

The routine for the solution of the inverse mode problem is presented next. Here the power iterations are used instead of the 'eig' routine and the computation of the residual is also described.

```
function [A Lambda Flag Residual]=ip(ShFun2Pt,ShFun3Pt,E,Rho,…
Modeshape,L,n,Fixeddofs)
le=L/n;
totdofs=2*n+1;
totAdofs=n+1;
%Assembly
gK=sparse(totdofs,totAdofs);
gM=sparse(totdofs,totAdofs);
for ele=1:n
        eledofs=2*ele-1:2*ele+1;
```

```
        eleAdofs=ele:ele+1;
        %First Area Component
        [eK,eM]=eleMat(E(ele),Rho(ele),[1;0],ShFun2Pt,ShFun3Pt,le);
        gK(eledofs,eleAdofs(1))=gK(eledofs,eleAdofs(1))+eK*…
        Modeshape(eledofs);
        gM(eledofs,eleAdofs(1))=gM(eledofs,eleAdofs(1))+eM*…
        Modeshape(eledofs);
        %Second Area Component
        [eK,eM]=eleMat(E(ele),Rho(ele),[0;1],ShFun2Pt,ShFun3Pt,le);
        gK(eledofs,eleAdofs(2))=gK(eledofs,eleAdofs(2))+eK*…
        Modeshape(eledofs);
        gM(eledofs,eleAdofs(2))=gM(eledofs,eleAdofs(2))+eM*…
        Modeshape(eledofs);
end
%Boundary Condition
Alldofs=1:totdofs;
Freedofs=setdiff(Alldofs,Fixeddofs);
%Multiplying by Transpose
Pseudo_K=gK(Freedofs,:)'*gK(Freedofs,:);
Pseudo_M=gK(Freedofs,:)'*gM(Freedofs,:);
tol=1e-16;Maxiter=10002;iter=0;change =1.+tol;
A=ones(totAdofs,1);
A=A/max(A);
Mat=Pseudo_K\Pseudo_M;
while change >tol && iter<Maxiter
        Aold=A;
        Anew=Mat*A;
        A=(Anew/max(Anew));
        change=norm(Aold-A,2)/n;
        iter=iter+1;
end
if(iter<Maxiter)
   Flag=1;
else
   Flag=0;
end
Lambda=(A'*Pseudo_K*A)/(A'*Pseudo_M*A);
Residual=gK(Freedofs,:)*A/Lambda-gM(Freedofs,:)*A;
```

For both these routines a pre-computation needs to be done of the structures before calling them. A sample piece of code to illustrate the way the routines are called for testing the algorithm is the one to follow.

```
clc;
clear;
close all;
L=1;
n=150;
le=L/n;
E=210e9*ones(n,1)';
```

```
Rho=7800*ones(n,1)';
totAdofs=n+1;totdofs=2*n+1;
y=linspace(0,L,totAdofs)';
x=linspace(0,L,totdofs)';
A = exp(5 + 2*y.*sin(2*pi*y/L))*1e-6;
Modeno=3;

%FixedFixed Case
Fixeddofs=[1,totdofs];
Mass=(Rho(1)*trapz(A)*L/n);
%Forming the Structure of Shape Functions at Gaussian Quadrature Points
Weight=[1;1];
N1of2 dN1of2 NA1of2]=ShFun(le,1/sqrt(3));
N2of2 dN2of2 NA2of2]=ShFun(le,-1/sqrt(3));
ShFun2Pt=struct('N_1',N1of2,'N_2',N2of2,'dN_1',dN1of2,'dN_2',dN2of2…
,'NA_1',NA1of2,'NA_2',NA2of2,'Weight',Weight);

Weight=[5/9;8/9;5/9];
N1of3 dN1of3 NA1of3]=ShFun(le,sqrt(3/5));
N2of3 dN2of3 NA2of3]=ShFun(le,0);
N3of3 dN3of3 NA3of3]=ShFun(le,-sqrt(3/5));
ShFun3Pt=struct('N_1',N1of3,'N_2',N2of3,'N_3',N3of3,'dN_1',dN1of3…
,'dN_2',dN2of3,'dN_3',dN3of3,'NA_1',NA1of3,'NA_2',NA2of3,'NA_3',…
NA3of3,'Weight',Weight);

Modeshape Lambda]=fp(ShFun2Pt,ShFun3Pt,E,Rho,A,L,n,…
Fixeddofs,Modeno);
[val,loc]=max(abs(Modeshape));
Modeshape=Modeshape/Modeshape(loc);
[I_A I_Lambda Flag Residual]=ip(ShFun2Pt,ShFun3Pt,E,Rho,…
Modeshape,L,n,Fixeddofs);
I_A=(Mass/(Rho(1)*trapz(I_A)*L/n))*I_A;
error_Area=norm(I_A-A,2)/(norm(A,2))*100;
errI_Lambda=(I_Lambda-Lambda)/Lambda*100;
Res=max(abs(Residual));
```

11. REFERENCES

1)Baccouch, M., Choura, S., El-Borgi S. and Nayfeh, A., "On the Selection of Physical and Geometrical Properties for the Confinement of Vibrations in Non-Homogeneous Beams," *ASCE Journal of Aerospace Engineering*, 19(3): 2006, 158-168.

2)Spletzer, M., Raman, A., Alexander Q. Wu, and Xianfan Xu, "Ultrasensitive mass sensing using mode localization in coupled microcantilevers," *Applied Physics Letters* 88, 2006.

3)Pedersen, N.L., "Design of cantilever probes for atomic force microscopy (AFM)," *Engineering Optimization* 32, 2000, pp. 373–392.

4)Graham, M.L., Gladwell, "Inverse Problems in Vibrations," *Kluwer Academic Publications, Second Edition, 2004*.
5)Chu, M., "Inverse eigenvalue problems," *SIAM Review* 40, 1992, 1-39.
*6)*Moody, Ten-Chao, Chu and Moody, T., Chu and Gene, H., Golub, "Inverse Eigenvalue Problems Theory, Algorithms, and Applications," *Oxford Science Publications,* 2005.
*7)*Ram, Y.M., and Graham, M.L., Gladwell, "Constructing a Finite element model of a vibratory rod from eigendata," *Journal of Sound and Vibration*, 1994, 169, 229-237.
*8)*Ram, Y.M., "An inverse mode problem for the continuous model of an axially vibrating rod," *American Society of Mechanical Engineers Transactions Journal of Applied Mechanics*, 1994, 61, 624-628.
*9)*Barcilon, V., "Inverse Problems for the Vibrating Beam in the free-clamped configuration," *Philosophical Transactions of the Royal Society of London,* 1982, A 304, 211-252.
*10)*Graham, M.L., Gladwell, "The Inverse Problem for the Euler-Bernoulli beam." *Proceedings of the Royal Society of London*, 1986, A 407, 199-218.
*11)*Lai, E., and Ananthasuresh, G.K., "On the Design of Bars and Beams for desired Mode Shapes", *Journal of Sound and Vibration,* 2002, 254(2), 393-406.
*12)*Ananthasuresh, G.K., "Inverse Mode Shape Problems for Bars and Beams with Flexible Supports", *Inverse Problems,Design and Optimization Symposium*, Rio de Janeiro, Brazil, 2004.
13) Gene, H.,Golub, Charles, F.,Van,Loan, "Matrix Computations",*JHU Press*,1996.
*14)*George,F.,Simmons, "Differential Equations with Applications and Historical Notes", Second Edition, *McGraw-Hill International Editions*, 1991.
*15)*Al-Gwaiz, M.A., "Sturm Liouville Theory and Its Applications", *Springer*, 2008.