Finite element analysis of multi-dimensional and simplified models for beams and plates

by

Rudi du Toit

15000258

supervisor: Prof NFJ Van Rensburg

Submitted in fulfilment of the requirements for the degree of

Magister Scientiae

in the Faculty of Natural and Agricultural Sciences

University of Pretoria

April 26, 2024

Declaration

I declare that the dissertation, which I hereby submit for the degree MSc Applied Mathematics at the University of Pretoria, is my own work and has not previously been submitted by me for a degree at this or any other tertiary institution

Name: Rudi du Toit

Date: December 6, 2023

Faculty of Natural and
Agricultural Sciences

TURNITIN DECLARATION

Full names of student	Rudi du Toit
Student number	15000258
Degree	MSc Applied Mathematics
Department	Mathematics and Applied Mathematics

Declaration by student:

I declare that I have used Turnitin according to University's policy in this regard.

SIGNATURE OF CANDIDATE: Rudi du Toit

I declare that I have seen and am satisfied with the Turnitin reports.

NAME OF SUPERVISOR: Prof. NFJ van Rensburg

SIGNATURE OF SUPERVISOR: NFJ. Rensburg

Date: 05/12/2023

Title Finite element analysis of multi-dimensional and simplified models for beams and plates

Name Rudi du Toit

Supervisor Prof NFJ van Rensburg

Department Mathematics and Applied Mathematics

Degree MSc Applied Mathematics

Summary

This dissertation is a literature study that investigates the validity of different linear models in application. Validity in this context refers to how well simplified lower-dimensional models, such as the Timoshenko beam and Reissner-Mindlin plate models, compare to more realistic higher-dimensional models, including a two-dimensional beam, and three-dimensional beam and plate models. The models in this dissertation are all special cases of a general vibration problem.

First, the dissertation examines the existence and uniqueness of the general vibration problem. An example is used to explain the theory, which is then subsequently applied to by proving that the assumptions are satisfied.

Following this, the concept of modal analysis is introduced using an example, before delving into the general case. These results on modal analysis are crucial to the dissertation, as they explain that the solutions of the models will compare well if the eigenvalues and eigenfunctions of the models compare well.

The dissertation then explores two theoretical results for the Finite Element Method (FEM). The initial result involves an analysis of an article on the convergence of the Galerkin Approximation. The findings of the article are reformulated as theorems with simplified notation for clearer presentation

Subsequently, the dissertation reviews results from a textbook regarding the convergence of eigenvalues and eigenfunctions in a general vibration problem when utilizing FEM. These results are adapted with updated notation and expanded upon for a more comprehensive explanation of the theory.

Concerning the Timoshenko beam model, the dissertation investigates an article that presents a method to calculate the exact solutions of the eigenvalue problem. Two practical examples are provided to illustrate the application of this method. Additionally, the dissertation looks at an article that compares

the theoretical results of the eigenvalue problem with an empirical study done by the authors.

For the remaining models, the dissertation employs FEM to solve the eigenvalue problems. The boundary value problems of each model are rewritten as systems of ordinary differential equations in matrix form using FEM. The eigenvalue problems are then derived from this matrix representation. Piecewise Hermite cubic basis functions are used, and the solutions of the eigenvalue problems are approximated using MATLAB scripts.

In investigating the validity of simplified models, the dissertation first considers an article comparing the Timoshenko beam model to a two-dimensional beam model. The authors method of comparison is discussed, and their results are replicated with a higher degree of accuracy. The dissertation then extends this approach to assess the validity of a two-dimensional beam model and a Reissner-Mindlin plate model, using the three-dimensional model as reference. The method to compare the models is the same as in the article. First the eigenvalues are calculated, sorted and matched by analyzing the corresponding mode shapes. The mode shapes are also used to identify eigenvalues specific to beam- and plate-type problems. The error can then be calculated. Different shapes of beams and plate models are considered that are realistic in application.

The derivation and comparison of the two and three-dimensional models is the main contribution of this dissertation.

Contents

1 Models for comparison

In this chapter, the different models are presented.

1.1 Model problem for a three-dimensional elastic solid

In this section we introduce the linear theory for a three-dimensional elastic solid undergoing vibration. The textbook of Fung [Fun65] is used with some changes in the notation.

The equation of motion and constitutive equation are given in Subsection 1.1.1. The dimensionless form of the equation of motion and constitutive equation are derived in Subsection 1.1.2. Model problems are presented in Subsection 1.1.3. The variational form for the vibration problem is given in Section 1.1.4. Plane stress is discussed in Section 1.1.5

1.1.1 Equations of motion and constitutive equations

Consider a vector valued function u defined on domain $\Omega \subset R^3$ where u describes the displacement of Ω in R^3.

Equation of motion

$$\rho \partial_t^2 u = \mathrm{div} T + Q. \tag{1.1.1}$$

In (1.1.1), ρ is the density of the elastic body, T is the stress tensor with components σ_{ij}, Q is an external body force acting on Ω and $\mathrm{div}T$ is the divergence of the tensor T and represented by

$$\text{div}T = \begin{bmatrix} \partial_1\sigma_{11} + \partial_2\sigma_{12} + \partial_3\sigma_{13} \\ \partial_1\sigma_{21} + \partial_2\sigma_{22} + \partial_3\sigma_{23} \\ \partial_1\sigma_{31} + \partial_2\sigma_{32} + \partial_3\sigma_{33} \end{bmatrix}. \tag{1.1.2}$$

Only small vibrations are considered. This means that the local displacements and rotations are small. Hence the stress tensor T is symmetric. Let $\text{Tr}(T)$ denote the trace of the stress tensor T, that is

$$\text{Tr}(T) = \sigma_{11} + \sigma_{22} + \sigma_{33}. \tag{1.1.3}$$

Remark Some books prefer to use ρQ for the external body force, but Q is also correct. Q is then a force per unit volume.

Strain

The infinitesimal strain tensor is defined on Ω as $\mathcal{E}$, with components

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \tag{1.1.4}$$

In this dissertation, we only consider isotropic materials. As a consequence, the constitutive equation (Hooke's law) takes the following form.

Hooke's law in terms of E and ν

$$\mathcal{E} = \left(\frac{1+\nu}{E}\right)T - \frac{\nu}{E}\text{Tr}(T)I, \tag{1.1.5}$$

where E is Young's Modulus and ν is Poisson's ratio.

Hooke's law in the alternative form

If the principal stresses σ_i are all non-zero, then Hooke's law can be written in the following form

$$T = \left(\frac{E}{1+\nu}\right)\mathcal{E} + \frac{\nu E}{(1+\nu)(1-2\nu)}\text{Tr}(\mathcal{E})I. \tag{1.1.6}$$

$\mathrm{Tr}(\mathcal{E}) = \varepsilon_{11} + \varepsilon_{12} + \varepsilon_{13}$ is the trace of the strain operator $\mathcal{E}$.

Hooke's Law in the alternative form (1.1.6) is the constitutive equation for the three-dimensional elastic model used in problems. Conditions for the problems to be well posed, are discussed in Chapter 2.

1.1.2 Dimensionless form

The dimensionless form of the equation of motion and constitutive equation are derived in this subsection. Suppose ℓ represents some notable dimension (e.g. the length) of the elastic body and G the shear modulus of elasticity.

Set

$$\tau = \frac{t}{t_0}, \quad \xi_i = \frac{x_i}{\ell}, \quad u^*(\xi,\tau) = \frac{1}{\ell t_0} u(x,t), \quad Q^* = \ell G \kappa^2 Q,$$
$$\text{and} \quad \sigma^*_{ij}(\xi) = \frac{1}{G\kappa^2}\sigma_{ij}(x),$$

where κ^2 is a dimensionless constant and t_0 must be specified. A convenient choice for t_0 (see Section 1.3) is

$$t_0 = \ell\sqrt{\frac{\rho}{G\kappa^2}}.$$

Substitution into (1.1.1) yields

$$\partial_\tau^2 u^* = \mathrm{div}T^* + Q^*,$$

where $T^*_{ij} = \sigma^*_{i,j}(\xi)$. The strain $\mathcal{E}$ is already dimensionless.

The dimensionless form of (1.1.5) and (1.1.6) are

$$\mathcal{E} = \frac{G\kappa^2}{E}\left[(1+\nu)T^* - \nu \mathrm{Tr}(T^*)I\right] \tag{1.1.7}$$

and

$$T^* = \frac{E}{G\kappa^2}\left[\left(\frac{1}{1+\nu}\right)\mathcal{E} + \frac{\nu}{(1+\nu)(1-2\nu)}\mathrm{Tr}(\mathcal{E})I\right]. \tag{1.1.8}$$

Introduce a dimensionless constant

$$\gamma = \frac{G\kappa^2}{E}.$$

Using this dimensionless constant, (1.1.7) and (1.1.8) becomes

$$\mathcal{E} = \gamma(1+\nu)T^* - \gamma\nu \mathrm{Tr}(T^*)I \tag{1.1.9}$$

and

$$T^* = \frac{1}{\gamma(1+\nu)}\mathcal{E} + \frac{\nu}{\gamma(1+\nu)(1-2\nu)}\mathrm{Tr}(\mathcal{E})I. \tag{1.1.10}$$

Remark The constant $G\kappa^2$ is introduced to allow for comparisons of the models in later chapters. The constant comes from the Timoshenko beam theory and is explained in Section 1.3

In the rest of this section's the original notation is retained for convenience.

Equations of motion in dimensionless form

$$\partial_t^2 u \quad = \quad \mathrm{div} T + Q \tag{1.1.11}$$

with

$$\mathrm{div} T \quad = \quad \begin{bmatrix} \partial_1\sigma_{11} + \partial_2\sigma_{12} + \partial_3\sigma_{13} \\ \partial_1\sigma_{21} + \partial_2\sigma_{22} + \partial_3\sigma_{23} \\ \partial_1\sigma_{31} + \partial_2\sigma_{32} + \partial_3\sigma_{33} \end{bmatrix}. \tag{1.1.12}$$

Constitutive equation in dimensionless form

$$T = \frac{1}{\gamma(1+\nu)}\mathcal{E} + \frac{\nu}{\gamma(1+\nu)(1-2\nu)}\mathrm{Tr}(\mathcal{E})I \tag{1.1.13}$$

1.1.3 Model problems

Suppose $\Omega \subset R^3$ is the reference configuration for a solid executing small vibrations. The boundary of Ω can be divided into two distinct parts, referred to as Σ and Γ. The following will be considered a model problem for a three-dimensional elastic body executing small vibrations.

The equations of motion (1.1.11) and (1.1.12) are satisfied in Ω;

Hooke's law (1.1.13) is satisfied in Ω;

The displacement $u = u_\Sigma$ is specified on Σ;

The traction $Tn = t_\Gamma$ is specified on Γ.

The two model problems considered in this dissertation are described below.

Problem 3D-1

Find a vector valued function u, satisfying equations (1.1.11) to (1.1.13) and the following boundary conditions:

$$\begin{aligned} u &= 0 \quad \text{on } \Sigma \\ Tn &= 0 \quad \text{on } \Gamma \end{aligned}$$

with n the outward normal vector to Ω.

Problem 3D-2

Find a vector valued function u, satisfying equations (1.1.11) to (1.1.13) and the following boundary conditions:

$$Tn = 0 \quad \text{on } \partial\Omega$$

with n the outward normal vector to Ω.

1.1.4 Variational form

Let $\phi \in C(\Omega)$ be an arbitrary vector valued function. Multiplying ϕ with equation (1.1.11) and integrating over the domain Ω results in,

$$\int_\Omega (\partial_t^2 u) \cdot \phi \, dV = \int_\Omega (\mathrm{div} T) \cdot \phi \, dV + \int_\Omega Q \cdot \phi \, dV.$$

Since the stress tensor T is symmetric,

$$\operatorname{div}(T\phi) \quad = \quad (\operatorname{div}T)\cdot\phi + \operatorname{Tr}(T\Phi),$$

where

$$\Phi \quad = \quad \begin{bmatrix} \partial_1\phi_1 & \partial_2\phi_1 & \partial_3\phi_1 \\ \partial_1\phi_2 & \partial_2\phi_2 & \partial_3\phi_2 \\ \partial_1\phi_3 & \partial_2\phi_3 & \partial_3\phi_3 \end{bmatrix},$$

and

$$\begin{aligned} \operatorname{Tr}(T\Phi) \quad &= \quad \sigma_{11}\partial_1\phi_1 + \sigma_{12}\partial_1\phi_2 + \sigma_{13}\partial_1\phi_3 + \sigma_{21}\partial_2\phi_1 + \sigma_{22}\partial_2\phi_2 + \sigma_{23}\partial_2\phi_3 \\ &\quad +\sigma_{31}\partial_3\phi_1 + \sigma_{32}\partial_3\phi_2 + \sigma_{33}\partial_3\phi_3. \end{aligned}$$

Using the Divergence Theorem and the symmetry of T,

$$\begin{aligned} \int_\Omega \operatorname{div}(T\phi)\ dV \quad &= \quad \int_{\partial\Omega} T\phi\cdot n\ dS, \\ &= \quad \int_{\partial\Omega} Tn\cdot\phi\ dS. \end{aligned}$$

The divergence formula gives

$$\int_\Omega \operatorname{div}(T)\cdot\phi\ dV \quad = \quad -\int_\Omega \operatorname{Tr}(T\Phi)\ dV + \int_{\partial\Omega} Tn\cdot\phi dS.$$

Therefore,

$$\int_\Omega (\partial_t^2 u)\cdot\phi\ dV = -\int_\Omega \operatorname{Tr}(T\Phi)\ dV + \int_\Omega Q\cdot\phi\ dV + \int_{\partial\Omega} Tn\cdot\phi\ dS.$$

The general variational form of the three-dimensional model is given as

$$\int_\Omega (\partial_t^2 u)\cdot\phi\ dV \quad = \quad \int_\Omega c_1\operatorname{Tr}(\mathcal{E}\Phi) + c_2\operatorname{Tr}(\mathcal{E})\operatorname{Tr}(\Phi)\ dV + \int_\Omega Q\cdot\phi\ dV + \int_{\partial\Omega} Tn\cdot\phi\ dS,$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1+\nu)(1-2\nu)}$.

Bilinear forms and integral

Define the bilinear forms

$$b(u,\phi) = \int_\Omega c_1 \text{Tr}(\mathcal{E}\Phi) + c_2 \text{Tr}(\mathcal{E})\text{Tr}(\Phi)\ dV \tag{1.1.14}$$

and

$$c(u,\phi) = \int_\Omega (\partial_t^2 u) \cdot \phi\ dV \tag{1.1.15}$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1+\nu)(1-2\nu)}$.

Also define the integral

$$(f,g) \ = \ \int_\Omega f \cdot g\ dV \tag{1.1.16}$$

Test functions

Define the following set of test functions $T(\Omega)$ for Problem 3D-1 and Problem 3D-2:

Problem 3D-1

$$T(\Omega) \ = \ \left\{\phi \in C^1(\bar{\Omega}) \mid \phi = 0 \ \text{ on } \ \Gamma\right\}$$

Problem 3D-2

$$T(\Omega) \ = \ C^1(\bar{\Omega})$$

The variational problem of Problem 3D-1 is given as Problem 3D-1V.

Problem 3D-1V

Find a function u such that for all $t > 0$, $u \in T(\Omega)$ and

$$c(u,\phi) = -b(u,\phi) + (Q,\phi), \tag{1.1.17}$$

for all $\phi \in T(\Omega)$.

The well-posedness of the model is treated in Chapter 2, where Korn's inequality is presented. Applications are continued in Chapters 4, 5 and 6. In Chapter 4, a free-free three-dimensional beam is discussed as part of an empirical study. In Chapter 5, the finite element analysis is applied to a cantilever beam to solve the eigenvalue problem. In Chapter 6, a cantilever beam is used in the comparison of the linear models.

1.1.5 Plane stress

When the stresses in an elastic body act parallel to a single plane, it is referred to as plane stress.

Consider the right-hand orthonormal set $\{e_1, e_2, e_3\}$. Without loss of generality, assume the stresses act parallel to the e_1-e_2 plane i.e. $\sigma_{3i} = 0$ for all i. From Hooke's Law (1.1.9) the following equations are obtained:

$$\begin{aligned} \varepsilon_{11} &= \gamma(\sigma_{11} - \nu\sigma_{22}) \qquad & \varepsilon_{33} &= -\gamma\nu(\sigma_{11} + \sigma_{22}) \\ \varepsilon_{22} &= \gamma(\sigma_{22} - \nu\sigma_{11}) \qquad & \varepsilon_{12} &= \gamma(1+\nu)\sigma_{12} \end{aligned} \tag{1.1.18}$$

and a sufficient condition

$$\varepsilon_{13} = \varepsilon_{23} = 0. \tag{1.1.19}$$

After some manipulation, we verify the following equations for the stress components given in [Fun65].

$$\begin{aligned} \sigma_{11} &= \frac{1}{\gamma(1-\nu^2)}(\varepsilon_{11} + \nu\varepsilon_{22}), \\ \sigma_{22} &= \frac{1}{\gamma(1-\nu^2)}(\nu\varepsilon_{11} + \varepsilon_{22}). \end{aligned} \qquad \sigma_{12} = \frac{1}{\gamma(1+\nu)}\varepsilon_{12}, \tag{1.1.20}$$

Another necessary strain condition for plane stress is found by substituting σ_{11} and σ_{22} from (1.1.20) into ε_{33} in (1.1.18) to obtain

$$\varepsilon_{33} = -\frac{\nu}{1-\nu}(\varepsilon_{11} + \varepsilon_{22}). \tag{1.1.21}$$

This is known as general plane stress. In this dissertation only a special case of general plane stress is considered in Section 1.2.

1.2 Two-dimensional model problem for an elastic solid

1.2.1 Introduction

The following derivation of the two-dimensional model is based on my own work, using the textbook [Sad05] as a guide.

Assume that $\sigma_{3i} = 0$ for $i = 1, 2, 3$; $\partial_3 u_1 = 0$, $\partial_3 u_2 = 0$ (u_1 and u_2 are functions of x_1 and x_2), and the strain component $\varepsilon_{33} = 0$. Then using the definition of strain

$$\partial_i u_3 = 0 \quad \text{for } i = 1, 2, 3 \text{ on } \Omega. \tag{1.2.1}$$

It follows that u_3 is a constant on Ω and u is a vector valued function in R^2 with two-dimensional stress and strain. The out of plane conditions for strain (1.1.21) falls away and Hooke's law can be written in a two-dimensional form using the stress components (1.2.16) as $T = \dfrac{1}{\gamma(1+\nu)}\mathcal{E} + \dfrac{\nu}{\gamma(1-\nu^2)}\text{tr}(\mathcal{E})I$.

1.2.2 Equations of motion and constitutive equations

The following equations of motion and constitutive equations follow from subsections 1.1.2 and 1.3.2.

Equation of motion

$$\partial_t^2 u = \text{div}T + Q, \tag{1.2.2}$$

where

$$\text{div}T = \begin{bmatrix} \partial_1\sigma_{11} + \partial_2\sigma_{12} \\ \partial_1\sigma_{21} + \partial_2\sigma_{22} \end{bmatrix}. \tag{1.2.3}$$

Constitutive equation

$$T = \frac{1}{\gamma(1+\nu)}\mathcal{E} + \frac{\nu}{\gamma(1-\nu^2)}\mathrm{Tr}(\mathcal{E})I. \tag{1.2.4}$$

1.2.3 Model problem

Suppose $\Omega \subset R^2$ is the reference configuration for a solid executing small vibrations. The boundary of Ω consists of two parts Σ and Γ. The model problem is described as:

> The equation of motion (1.2.2) and constitutive equation (1.2.4) are satisfied in Ω.
>
> The displacement u is specified on Σ;
>
> Traction Tn is specified on Γ.

Problem 2D-1

Find a vector valued function u, satisfying equations (1.2.2) to (1.2.4) and the boundary conditions:

$$\begin{aligned} u &= 0 \quad \text{on } \Sigma, \\ Tn &= 0 \quad \text{on } \Gamma. \end{aligned}$$

with n the outward unit normal to $\partial\Omega$.

1.2.4 Variational form

The steps to derive the variational form is almost identical to the three-dimensional case. Therefore it is not shown again and only the differences are discussed.

Instead of volume integrals (dV) and surface integrals (dS), the two-dimensional model has area (dA) and line integrals (ds). Rather than the divergence formula, the divergence form of Green's formula is used to obtain the following result:

$$\int_\Omega \mathrm{div}(T)\cdot\phi \; dA \;\; = \;\; -\int_\Omega \mathrm{Tr}(T\Phi)\; dA + \int_\Gamma Tn\cdot\phi ds.$$

The general variational form of the two-dimensional model is given as

$$\int_\Omega (\partial_t^2 u) \cdot \phi \; dA \;\; = \;\; \int_\Omega c_1 \mathrm{Tr}(\mathcal{E}\Phi) + c_2 \mathrm{Tr}(\mathcal{E})\mathrm{Tr}(\Phi) \; dA + \int_\Omega Q \cdot \phi \; dA + \int_\Gamma Tn \cdot \phi \; ds,$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1-\nu^2)}$.

Bilinear forms and integral

Define the bilinear forms

$$b(u, \phi) = \int_\Omega c_1 \mathrm{Tr}(\mathcal{E}\Phi) + c_2 \mathrm{Tr}(\mathcal{E})\mathrm{Tr}(\Phi) \; dA \tag{1.2.5}$$

and

$$c(u, \phi) = \int_\Omega (\partial_t^2 u) \cdot \phi \; dA \tag{1.2.6}$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1-\nu^2)}$.
Also define the integral

$$(f, g) \;\; = \;\; \int_\Omega f \cdot g \; dA \tag{1.2.7}$$

Test functions for problem 2D-1

The test function space has the same definition of the test function space for Problem 3D-1. But since $\Omega \in R^2$, we have that $T(\Omega) \subset R^2$.

$$T(\Omega) \;\; = \;\; \left\{ \phi \in C^1(\bar{\Omega}) \mid \phi = 0 \;\text{ on }\; \Gamma \right\}.$$

Problem 2D-1V

Find a function u such that for all $t > 0$, $u \in T(\Omega)$ and

$$c(u, \phi) = -b(u, \phi) + (Q, \phi) \tag{1.2.8}$$

for all $\phi \in T(\Omega)$.

Applications are continued in Chapter 5 and 6. In Chapter 5, the finite element method is applied to a cantilever beam to solve the eigenvalue problem. In Chapter 6, a cantilever beam in used in the comparison of our linear models.

1.3 Timoshenko beam models

Consider the classical Timoshenko model for the vibration of a beam with no damping.

1.3.1 Equations of motion and constitutive equations

In this section we introduce the Timoshenko beam theory for the transverse vibration of a uniform beam. For a reference of the model [Tim21], and [Fun65] were used.

Consider a beam defined on the interval $[0, \ell]$. Let w represent the transverse displacement and ϕ a rotation of the cross-sections of the beam.

Equations of motion

$$\rho A \partial_t^2 w = \partial_x V + Q, \tag{1.3.1}$$

$$\rho I \partial_t^2 \phi = V + \partial_x M. \tag{1.3.2}$$

In (1.3.1) and (1.3.2) ρ denotes the density, A is the area of a cross section, I is the area moment of inertia, M is the moment, V is the shear force and Q is an external force acting on the beam.

Constitutive equations

$$M = EI\partial_x \phi, \tag{1.3.3}$$

$$V = AG\kappa^2(\partial_x w - \phi), \tag{1.3.4}$$

E is Young's modulus, G the shear modulus and κ^2 the shear correction factor.

Dimensionless form

As mentioned in Subsection 1.1.2, the same dimensionless scaling is used for all the models in this dissertation.

Set

$$\tau = \frac{t}{t_0},\ \xi = \frac{x}{\ell},\ w^*(\xi, \tau) = \frac{w(x,t)}{\ell} \text{ and } \phi^*(\xi, \tau) = \phi(x, t).$$

The dimensionless forms of the shear force, moment and external force density are

$$V^*(\xi,\tau) = \frac{V(x,t)}{AG\kappa^2}, \quad M^*(\xi,\tau) = \frac{M(x,t)}{AG\kappa^2\ell} \quad \text{and} \quad Q^*(\xi,\tau) = \frac{Q(x,t)\ell}{AG\kappa^2}.$$

Choose t_0 the same as in Subsection 1.1.2, i.e.

$$t_0 = \ell\sqrt{\frac{\rho}{G\kappa^2}}.$$

As in [VV06] and [LLV09] we use the dimensionless constants

$$\alpha = \frac{A\ell^2}{I} \text{ and } \beta = \frac{AG\kappa^2\ell^2}{EI}.$$

Interestingly, it turns out that

$$\frac{\beta}{\alpha} = \gamma$$

where γ is the dimensionless parameter defined in Subsection 1.1.2.

Remark Recall that in Subsection 1.1.2 the constant $G\kappa^2$ was used for the scaling of the stresses.

Dimensionless equations of motion

$$\partial_t^2 w = \partial_x V + Q, \tag{1.3.5}$$
$$\frac{1}{\alpha}\partial_t^2 \phi = V + \partial_x M. \tag{1.3.6}$$

Dimensionless constitutive equations

$$M = \frac{1}{\beta}\partial_x \phi, \tag{1.3.7}$$
$$V = \partial_x w - \phi. \tag{1.3.8}$$

The original notation is retained for convenience.

1.3.2 Boundary conditions

The following boundary conditions are considered for the Timoshenko beam models in this dissertation.

Clamped or built-in endpoint - At the clamped end the boundary conditions are $w = 0$ and $\phi = 0$.

Free endpoint - At the free end the boundary conditions are $M = 0$ and $V = 0$.

Pinned or hinged endpoint - At the pinned endpoint the boundary conditions are $w = 0$ and $M = 0$.

Suspended endpoint - At the pinned endpoint the boundary conditions are $V = kw$ and $M = 0$. The parameter k is the elastic constant of the linear spring that suspends the beam.

1.3.3 Boundary value problems

The following model problems are used in this dissertation.

Problem T-1

The beam is pinned at both endpoints.

Boundary Conditions

$$\begin{aligned} w(0,\cdot) &= 0, \qquad M(0,\cdot) = 0, \\ w(1,\cdot) &= 0, \qquad M(1,\cdot) = 0. \end{aligned}$$

Problem T-2

The beam is clamped at the left endpoint where $x = 0$, and free-hanging where $x = 1$. (In this configuration, the beam is called a cantilever beam.)

Boundary Conditions

$$
\begin{aligned}
w(0,\cdot) = 0, &\quad \phi(0,\cdot) = 0,\\
M(1,\cdot) = 0, &\quad V(1,\cdot) = 0.
\end{aligned}
$$

Problem T-3

The beam is suspended at both endpoints.

Boundary Conditions

$$
\begin{aligned}
V(0,\cdot) = kw(0,\cdot), &\quad M(0,\cdot) = 0,\\
V(1,\cdot) = -kw(1,\cdot), &\quad M(1,\cdot) = 0.
\end{aligned}
$$

Remark: These boundary conditions are only valid for w "small enough" for the motion to remain linear.

Problem T-4

The beam is free at both endpoints. Boundary Conditions

$$
\begin{aligned}
V(0,\cdot) = 0, &\quad M(0,\cdot) = 0,\\
V(1,\cdot) = 0, &\quad M(1,\cdot) = 0.
\end{aligned}
$$

An example of a free-free beam is given in Chapter 4.

1.3.4 Variational form

Let $v, \psi \in C^1[0,1]$ be arbitrary functions. Multiply by these functions in equations (1.3.5) and (1.3.6) and integrate over the interval $[0,1]$ to obtain:

$$
\begin{aligned}
\int_0^1 \partial_t^2 w v &= \int_0^1 \partial_x V v + \int_0^1 Q v,\\
\int_0^1 \frac{1}{\alpha}\partial_t^2 \phi \psi &= \int_0^1 V\psi + \int_0^1 \partial_x M \psi.
\end{aligned}
$$

Integration by parts yields

$$
\begin{aligned}
\int_0^1 \partial_t^2 w v &= -\int_0^1 V v' + \int_0^1 Q v + V(\cdot,t)v(\cdot)|_0^1,\\
\int_0^1 \frac{1}{\alpha}\partial_t^2 \phi \psi &= \int_0^1 V\psi - \int_0^1 M\psi' + M(\cdot,t)\psi(\cdot)|_0^1.
\end{aligned}
$$

Substitute the constitutive equations (1.3.7) and (1.3.8) to obtain

$$\begin{aligned}
\int_0^1 \partial_t^2 w v &= -\int_0^1 (\partial_x w - \phi)v' + \int_0^1 Qv \\
&\quad + \partial_x w(\cdot,t)v(\cdot)|_0^1 - \phi(\cdot,t)v(\cdot)|_0^1, \qquad (1.3.9) \\
\int_0^1 \frac{1}{\alpha}\partial_t^2 \phi\psi &= \int_0^1 (\partial_x w - \phi)\psi - \frac{1}{\beta}\int_0^1 \partial_x\phi\psi' \\
&\quad + \frac{1}{\beta}\partial_x\phi(\cdot,t)\psi(\cdot)|_0^1. \qquad (1.3.10)
\end{aligned}$$

Function spaces

It is convenient to define the following function spaces:

$$F_0[0,1] = \{f \in C^1[0,1] \mid f(0) = f(1) = 0\} \qquad (1.3.11)$$
$$F_1[0,1] = \{g \in C^1[0,1] \mid g(0) = 0\} \qquad (1.3.12)$$

Test function spaces for different problems

Problem T-1

$$T[0,1] := F_0[0,1] \times C^1[0,1]$$

Problem T-2

$$T[0,1] := F_1[0,1] \times F_1[0,1]$$

Problem T-3

$$T[0,1] := C^1[0,1] \times C^1[0,1]$$

Problem T-4

$$T[0,1] := C^1[0,1] \times C^1[0,1]$$

The variational problem of the pinned-pinned beam

Using the test function space for Problem T-1, the equations (1.3.9) and (1.3.10) reduce to

$$\int_0^1 \partial_t^2 wv = -\int_0^1 (\partial_x w - \phi)v' + \int_0^1 Qv, \tag{1.3.13}$$

$$\int_0^1 \frac{1}{\alpha}\partial_t^2 \phi\psi = \int_0^1 (\partial_x w - \phi)\psi - \frac{1}{\beta}\int_0^1 \partial_x\phi\psi'. \tag{1.3.14}$$

for all $v, \psi \in T[0,1]$.

Bilinear forms

For $f, g \in T[0,1]$, define the bilinear forms

$$c(f,g) = \int_0^1 \partial_t^2 f_1 g_1 + \frac{1}{\alpha}\int_0^1 \partial_t^2 f_2 g_2, \tag{1.3.15}$$

$$b(f,g) = \int_0^1 (f_1' - f_2)(g_1' - g_2) + \frac{1}{\beta}\int_0^1 f_2' g_2', \tag{1.3.16}$$

Define the integral

$$(f,g) = \int_0^1 fg \tag{1.3.17}$$

for all $f, g \in L^2(0,1)$.

Remark
$L^2(a,b)$ is the space of all square integrable functions on the interval (a,b). The inner product is defined by $(f,g) = \int_a^b fg$, and the induced norm $||f|| = \int_a^b f^2$. See the appendix for more information.

Problem T-1V

Find a function $u = \langle w, \phi\rangle$ such that for all $t > 0$, $u \in T[0,1]$ satisfying

$$c(\partial_t^2 u, \phi) = -b(u,\phi) + (Q,\phi) \tag{1.3.18}$$

for each $\phi = \langle v, \psi\rangle \in T[0,1]$.

The variational problem of the cantilever beam.

Using the test function space for Problem T-2, the equations (1.3.9) and (1.3.10) reduce to

$$\int_0^1 \partial_t^2 w v = -\int_0^1 (\partial_x w - \phi)v' + \int_0^1 Qv, \tag{1.3.19}$$

$$\int_0^1 \frac{1}{\alpha}\partial_t^2 \phi\psi = \int_0^1 (\partial_x w - \phi)\psi - \frac{1}{\beta}\int_0^1 \partial_x \phi\psi', \tag{1.3.20}$$

for all $v, \psi \in T[0,1]$.

This variational form is the same as for the case of the pinned-pinned beam, Problem T-1 as discussed above. Therefore the bilinear forms (1.3.15) and (1.3.16) can be used.

Problem T-2V

Find a function $u = \langle w, \phi\rangle$ such that for all $t > 0$, $u \in T[0,1]$ satisfying

$$c(u,\phi) = -b(u,\phi) + (Q,\phi) \tag{1.3.21}$$

for each $\phi = \langle v, \psi\rangle \in T[0,1]$.

Remark The formulation of Problem T-3V and Problem T-4V are the same. But there are some complications for Problem T-3V, which are discussed in Chapter 4.

The application for the Timoshenko beam theory are continued in Chapters 2,4 and 6. In Chapter 2, the cantilever beam is used an example to the existence theory. In Chapter 4, modal analysis is applied to the free-free and cantilever beam. The free-free and suspended beams are also used to discuss an empirical study. In Chapter 6, the cantilever beam is used in the comparison of our linear models.

1.4 Reissner-Mindlin Plate Model

Consider small vibrations of a plate. The model is from the article [LVV09b]. This motion can be described by using spherical coordinates. Assume that the plate has a two dimensional domain $\Omega \subset R^2$. For $r \geq 0$, the coordinates x_1 and x_2 can be rewritten as

$$x_1 = r \sin\phi \cos\theta,$$

$$x_2 = r \sin\phi \sin\theta.$$

Let

$$r = \sqrt{q^2 + x_3^2}, \quad \sin\phi = \frac{q}{r} \quad \text{and} \quad \cos\theta = \frac{x_3}{r}.$$

and define the unit vectors

$$\begin{aligned} \bar{e}_r &= \cos\theta\ \bar{e}_1 + \sin\theta\ \bar{e}_2, \\ \bar{e}_n &= \sin\phi\ \bar{e}_r + \cos\psi\ \bar{e}_3, \\ \bar{e}_\phi &= \cos\phi\ \bar{e}_r - \sin\psi\ \bar{e}_3. \end{aligned}$$

Any point on the plate can be described by

$$\bar{x} = x_1\bar{e}_1 + x_2\bar{e}_2 + x_3\bar{e}_3 = q\bar{e}_r + x_3\bar{e}_3.$$

Let w represent the displacement of the plate body and ψ the angle between the material line and a line perpendicular to the plate. In the spherical coordinate form, the angle can easily be calculated for the directions of $\bar{e}_1$ and $\bar{e}_2$, i.e.

$$\begin{aligned} \psi_1 &= \frac{q}{r}\bar{e}_r \cdot \bar{e}_1, \\ \psi_2 &= \frac{q}{r}\bar{e}_r \cdot \bar{e}_2. \end{aligned}$$

Define

$$\psi = \langle \psi_1,\ \psi_2 \rangle = \langle \sin\phi\cos\theta,\ \sin\phi\sin\theta \rangle.$$

The plate model in consideration is restricted to a linear model. Therefore $\sin\phi$ can be approximated by ϕ such that

$$\psi = \langle \psi_1,\ \psi_2 \rangle = \langle \phi\cos\theta,\ \phi\sin\theta \rangle.$$

The equations for the Reissner-Mindlin plate model is given in the next section. The model is restricted to the linear theory.

1.4.1 Equations of Motion and Constitutive Equations

Consider a Reissner-Mindlin plate with reference configuration $\Omega \in R^2$. Let $u \in \Omega$ define the transverse motion of the plate model and $\psi = \langle \psi_1,\ \psi_2 \rangle$ the angle between the material line and a line perpendicular to the plate.

Equations of Motion

$$\rho h \partial_t^2 w = \operatorname{div}\mathbf{Q} + q \tag{1.4.1}$$

$$\rho I \partial_t^2 \psi = \operatorname{div} M - Q \tag{1.4.2}$$

In these equations ρ denotes the density of the plate, $I = \dfrac{h^3}{12}$ the length moment of inertia, M the moment density and Q the shear force density. The parameter q is an external force acting on the plate. $\mathbf{Q}$ is defined as part of the constitutive equations below. The moment density is defined as

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}.$$

Constitutive Equations

For the linear model, the constitutive equations are defined as:

$$\mathbf{Q} = \kappa^2 G h (\nabla w + \psi) \tag{1.4.3}$$

$$M_{11} = \frac{1}{2} D \left[2(\partial_1 \psi_1 + \nu \partial_2 \psi_2) \right] \tag{1.4.4}$$

$$M_{12} = M_{21} = \frac{1}{2} D \left[(1-\nu)(\partial_1 \psi_2 + \nu \partial_2 \psi_2) \right] \tag{1.4.5}$$

$$M_{22} = \frac{1}{2} D \left[2(\partial_2 \psi_2 + \nu \partial_1 \psi_1) \right] \tag{1.4.6}$$

where G is the shear modulus, κ^2 a shear correction factor and D is a measure of stiffness for the plate and is defined by

$$D = \frac{EI}{1-\nu^2}$$

where E is Young's modulus and ν Poisson's ratio.

In classical plate theory, $\psi = -\nabla w$ and the constitutive equation for $\mathbf{Q}$ is excluded.

1.4.2 Dimensionless Form

Set

$$\tau = \frac{t}{t_0}, \qquad \xi_1 = \frac{x_1}{\ell}, \qquad \xi_2 = \frac{x_2}{\ell},$$

$$w^*(\xi,\tau) = \frac{w(x,t)}{\ell} \text{ and } \psi^*(\xi,\tau) = \psi(x,t).$$

The dimensionless forms of the force density, moment density and load can be constructed as

$$Q^*(\xi,\tau) = \frac{Q(x,t)}{\ell G\kappa^2}, \quad M^*(\xi,\tau) = \frac{M(x,t)}{\ell^2 G\kappa^2} \text{ and } q^*(\xi,\tau) = \frac{q(x,t)}{G\kappa^2}.$$

Choose t_0 the same as in Section 1.1.2, i.e.

$$t_0 = \ell\sqrt{\frac{\rho}{G\kappa^2}}.$$

The dimensionless constants are

$$h = \frac{h}{\ell}, \qquad I^* = \frac{h^3}{12} \text{ and } \beta = \frac{\ell^3 G\kappa^2}{EI}.$$

Remark

Similar to the model in Section 1.1.2, the constant G is introduced to allow for the comparison of models in later chapters. It will also be required that κ is the same for the beam and plate models.

For convienience, the original notation is used for the dimensionless form.

Dimensionless Equations of Motion

$$h\partial_t^2 w = \mathrm{div}\mathbf{Q} + q \tag{1.4.7}$$

$$I\partial_t^2 \psi = \mathrm{div}M - Q \tag{1.4.8}$$

Dimensionless Constitutive Equations

$$
\begin{aligned}
\mathbf{Q} &= h(\nabla w + \psi) && (1.4.9)\\
M_{11} &= \frac{1}{2\beta(1-\nu^2)}\left[2(\partial_1\psi_1 + \nu\partial_2\psi_2\right] && (1.4.10)\\
M_{12} = M_{21} &= \frac{1}{2\beta(1-\nu^2)}\left[(1-\nu)(\partial_1\psi_2 + \partial_2\psi_1)\right] && (1.4.11)\\
M_{11} &= \frac{1}{2\beta(1-\nu^2)}\left[2(\partial_2\psi_2 + \nu\partial_1\psi_1)\right] && (1.4.12)
\end{aligned}
$$

1.4.3 Boundary Conditions

The boundary conditions are applied along an edge of a plate. Let n be the outward normal of the edge, and τ be a unit vector tangent to that edge.

Some commonly used boundary (or edge) conditions are:

Free Edge:

$$Mn \cdot n = 0, \quad Mn \cdot \tau = 0 \quad \text{and} \quad \mathbf{Q} \cdot n = 0$$

Soft Supported Edge:

$$Mn \cdot n = 0, \quad w = 0 \quad \text{and} \quad Mn \cdot \tau = 0$$

Rigidly Supported Edge:

$$Mn \cdot n = 0, \quad w = 0 \quad \text{and} \quad \psi \cdot \tau = 0$$

Soft Clamped Edge:

$$w = 0, \quad Mn \cdot \tau = 0 \quad \text{and} \quad \psi \cdot n = 0$$

Rigidly Clamped Edge:

$$w = 0, \quad \psi_1 = 0 \quad \text{and} \quad \psi_2 = 0$$

In this dissertation, only the free edge and rigidly clamped edge boundary conditions are considered.

1.4.4 Model Problems

Let $\Omega \subset R^2$ denote reference configuration of the plate model. Following [Wu06], Ω is a rectangle.

The boundary $\partial\Omega$ can be divided into 4 distinct parts. Denote any two opposing sides by Σ_0 and Σ_1 and the remaining two opposing sides by Γ_0 and Γ_1.

Problem P-1

Consider a cantilever plate model. In this configuration, the plate is clamped at one edge and free hanging at the rest of the boundary. Without loss of generality, assume that the plate is clamped on the edge Σ_0.

Find functions w and ψ satisfying equations (1.4.7) to (1.4.12) and the boundary conditions:

$$w = 0, \quad \text{and} \quad \psi = \bar{0} \quad \text{on } \Sigma_0.$$
$$Mn \cdot n = 0, \quad Mn \cdot \tau = 0, \quad \text{and} \quad \mathbf{Q} \cdot n = 0 \quad \text{on } \partial\Omega \setminus \Sigma_0.$$

with τ a unit vector perpendicular to n.

Test Functions for Problem P-1

Define the following spaces $T_1(\bar{\Omega})$ and $T_2(\bar{\Omega})$ determined by the boundary conditions for w and ϕ

$$\begin{aligned} T_1(\bar{\Omega}) &= \left\{ v \in C^1(\bar{\Omega}) \mid v = 0 \text{ on } \Sigma_0 \right\}, \\ T_2(\bar{\Omega}) &= \left\{ \phi = [\phi_1 \ \phi_2]^T \mid \phi_1, \phi_2 \in C^1(\bar{\Omega}), \ \phi_1 = \phi_2 = 0 \text{ on } \Sigma_0 \right\}. \end{aligned}$$

Problem P-2

Consider a plate model that is rigidly clamped on $\partial\Omega$. Find functions w and ψ satisfying equations (1.4.7) to (1.4.12) and the boundary conditions:

$$w = 0, \quad \text{and} \quad \psi = \bar{0} \quad \text{on } \partial\Omega.$$

1.4.5 Variational Form

Let $v \in C_2^1[0,1]$ and $\phi \in C^1[0,1] \times C^1[0,1]$ such that ϕ is a vector valued function. Multiplication these artitrary functions to (1.4.7) and (1.4.8) yields:

$$\begin{aligned}
\int_\Omega h\partial_t^2 wv \; dA &= \int_\Omega \operatorname{div}(\mathbf{Q})v \; dA + \int_\Omega qv \; dA \\
\int_\Omega I\partial_t^2 \psi \cdot \phi \; dA &= \int_\Omega \operatorname{div}(M) \cdot \phi \; dA - \int_\Omega Q \cdot \phi \; dA
\end{aligned}$$

Following from Green's Formulas, similar to Section 1.1.4,

$$\begin{aligned}
\int_\Omega \operatorname{div}(\mathbf{Q})v \; dA &= -\int_\Omega \mathbf{Q} \cdot \nabla v \; dA + \int_{\partial\Omega} (Q \cdot n)v ds, \\
\int_\Omega \operatorname{div}(M) \cdot \phi \; dA &= -\int_\Omega \operatorname{Tr}(M\Phi) \; dA + \int_{\partial\Omega} Mn \cdot \phi ds.
\end{aligned}$$

$\operatorname{Tr}(M\Phi)$ is the trace of the matrix $M\Phi$, n is the normal vector to Ω, and

$$\Phi = \begin{bmatrix} \partial_1\phi_1 & \partial_2\phi_1 \\ \partial_1\phi_2 & \partial_2\phi_2 \end{bmatrix}.$$

The variational form is given as

$$\int_\Omega h\partial_t^2 wv \; dA = -\int_\Omega Q \cdot \nabla v \; dA + \int_\Omega qv \; dA + \int_{\partial\Omega} (Q \cdot n)v ds, \quad (1.4.13)$$

$$\int_\Omega I\partial_t^2 \psi \cdot \phi \; dA = -\int_\Omega \operatorname{Tr}(M\Phi) \; dA - \int_\Omega Q \cdot \phi \; dA + \int_{\partial\Omega} Mn \cdot \phi ds. \quad (1.4.14)$$

Using the test function space for Problem P-1, the equations (1.4.13) and (1.4.14) reduce to

$$\int_\Omega h\partial_t^2 wv \; dA = -\int_\Omega \mathbf{Q} \cdot \nabla v \; dA + \int_\Omega qv \; dA, \quad (1.4.15)$$

$$\int_\Omega I\partial_t^2 \psi \cdot \phi \; dA = -b(\phi, v) - \int_\Omega Q \cdot \phi \; dA, \quad (1.4.16)$$

for all $v \in T_1(\bar{\Omega})$ and $\phi \in T_2(\bar{\Omega})$

Bilinear Forms and integral

Let $u = \langle w, \psi \rangle$ and $\phi = \langle v, \phi \rangle$. Define the bilinear forms

$$b(u, \phi) \quad = \quad \int_\Omega \mathbf{Q} \cdot \nabla v \; dA + \int_\Omega \mathrm{Tr}(M\Phi) \; dA,$$

and

$$c(u, \phi) = \int_\Omega h(\partial_t^2 w) v \; dA + \int_\Omega I(\partial_t^2 \psi) \cdot \phi \; dA \tag{1.4.17}$$

Also define the integral

$$(f, g) \quad = \quad -\int_\Omega f \cdot g \; dA \tag{1.4.18}$$

Problem P-1V

Find a function $u = \langle w, \psi \rangle$, such that for all $t > 0$, $u \in T_1(\bar{\Omega}) \times T_2(\bar{\Omega})$ and the following equations are satisfied

$$c(u, \phi) \quad = \quad -b(u, \phi) + (Q, \phi), \tag{1.4.19}$$

with $\phi = \langle v, \phi \rangle \in T_1(\bar{\Omega}) \times T_2(\bar{\Omega})$ an arbitrary function.

Applications are continued in Chapter 5 and 6. In Chapter 5, the finite element method is applied to a cantilever plate to solve the eigenvalue problem. In Chapter 6, a cantilever plate in used in the comparison of the linear models.

2 Mathematical analysis of vibration problems

2.1 Introduction

The models in this dissertation concern vibration of elastic bodies. These type of models have a similar variational form as the wave equation and are called second order hyperbolic type problems. This chapter discusses theory for the existence and uniqueness of solutions for these type of problems. The theoretical basis for modal analysis is also presented.

In this chapter we consider the work done by the authors of [VV02] and [VS18]. In these articles the authors prove the existence and uniqueness of solutions for general second order hyperbolic type problems. The problems in Chapter 1 are examples. The article [VV02] concerns the case of a symmetric bilinear form b. The article [VS18] extends this work where b need not be symmetric. The article [VV02] is sufficient for the models of this dissertation, while [VS18] is only used where the notation is more convenient and to gain more insight.

Before the general theory is discussed, a model is presented to be used as an example to show the application of the theory. The model is a cantilever Timoshenko beam, denoted by Problem T-2 in Section 1.3.3.

In Section 1.3.4, the variational problem for the pinned-pinned beam is derived. The derivation of the cantilever beam model is similar, with a different test function space. Recall the variational problem for the cantilever beam Problem T-2V, with the test function space $T[0,1] = F_1[0,1] \times F_1[0,1]$. To determine the solvability of the problem, the weak variational problem is considered first.

To obtain the weak variational form of Problem T-2V some preparation is required. A natural setting for the problem is the product space $L^2(0,1) \times L^2(0,1)$, denoted by X. The inner product for $L^2(0,1)$ yields the inner product

$(x_1, y_1) + (x_2, y_2)$ for X. The idea is to replace the pair $\langle w, \phi \rangle$ by a function u "of time only" and values $u(t)$ in the space X.

Define a function u with domain J, an interval of real numbers. The range of u is contained in X:

$$u(t)(x) = \langle w(x,t), \phi(x,t) \rangle. \tag{2.1.1}$$

A derivative u' for u may be defined by

$$|| \left(h^{-1} u(t+h) - u(t) \right) - u'(t) ||_X \to 0.$$

Then u'' is defined by $u'' = (u')'$. Sometimes the derivatives are denoted by $\dot{u}$ and $\ddot{u}$.

Using the bilinear forms, b and c from Section 1.3.4, Problem T-2V can be rewritten in the following form

$$c(\ddot{u}(t), v) + b(u(t), v) \quad = \quad (Q(t), v), \tag{2.1.2}$$

for each $v \in T[0,1]$.

To apply the theory from [VV02], complete function spaces are required. It is known that $L^2(0,1)$ is a complete function space and hence X is complete.

The Sobolev space $H^1(0,1)$ is all the functions in $L^2(0,1)$ with at least a first-order weak derivative. $H^1(0,1)$ is complete (see Appendix A).

The test functions $T[0,1]$ do not form a complete space. Let $V(0,1)$ be the closure of $F_1(0,1)$ in $H^1(0,1)$ (see Section 1.3.4). It follows that $V(0,1)$ is complete and the product space $V(0,1) \times V(0,1)$, denoted by V, is complete. It is called the energy space.

Using these complete product spaces and (2.1.2), the weak variational problem for the cantilever Timoshenko beam is defined as Problem T-2W.

Problem T-2W

Find a function u such that $\forall\ t \in J$, $u(t) \in V$, $\ddot{u}(t) \in W$ and

$$c(\ddot{u}(t), v) + b(u(t), v) \quad = \quad (Q(t), v)$$

for all $v \in V$. The initial conditions $u(0) = u_0$ and $\dot{u}(0) = u_1$ must be specified.

The space X equipped with the inner-product c is denoted by W. W is called the inertia space and in Section 2.3 it is shown that c is an inner-product for W.

In the next section, the theory of the article [VV02] is presented. In Section 2.3 the theory is applied to Problem T-2W.

2.2 Existence and uniqueness of solutions

The general weak variational problem is studied in this section. The following theory is from the article [VV02]. To start, some notation is given, as well as the relations between various Hilbert spaces. The necessary assumptions are also stated.

As mentioned before, ideas and notation from [VS18] are also used.

2.2.1 The variational approach

Let V, W and X be real Hilbert spaces such that W is a linear subspace of X, and V is a linear subspace of W, i.e. $V \subset W \subset X$.

> X is the global space with inner product $(\cdot,\cdot)_X$ and the induced norm $||\cdot||_X$.
>
> W is the inertia space with inner product $(\cdot,\cdot)_W$ and the induced norm $||\cdot||_W$.
>
> V is the energy space with inner product $(\cdot,\cdot)_V$ and the induced norm $||\cdot||_V$.

The following notation is important for the theory that follows.

Let J be a interval of real numbers containing zero. It can have one of the following forms $[0,\ T)$, $[0,\ \infty)$ or an arbitrary open interval. For any function u on the interval J and range in a Hilbert space Z, derivatives can be defined. A derivative u' for u may be defined by

$$||(h^{-1}u(t+h) - u(t)) - u'(t)||_Z \to 0.$$

Then u'' is defined by $u'' = (u')'$.

Notation

$u \in C(J, Z)$ if u is continuous on J with respect to the norm of Z;

$u'(t) \in Z$ if u is differentiable with respect to the norm of Z;

$u \in C^k(J, Z)$ if $u^{(k)} \in C(J, Z)$.

Remark For Problem T-2, Z can be X, W or V.

Let a, b and c be bilinear forms where a and b are defined on V and c is defined on W. For the models in this dissertation, the bilinear forms b and c are symmetric. Furthermore, $b(\cdot,\cdot) = (\cdot,\cdot)_V$ and $c(\cdot,\cdot) = (\cdot,\cdot)_W$.

Problem GVar

Given a function $f : J \to X$, find a function $u \in C(J,\ X)$ such that u' is continuous at 0 with respect to $\|\cdot\|_W$ and for each $t \in J$, $u(t) \in V$, $u'(t) \in V$, $u''(t) \in W$ and

$$c(u''(t), v) + a(u'(t), v) + b(u(t), v) = (f(t), v)_X \quad \text{for each } v \in V, \quad (2.2.1)$$

while $u(0) = u_0, \quad u'(0) = u_1$.

This general variational form is applicable to all the models in this dissertation. In [VV02], a more general form is given that includes the possibility for damping terms in the models. Damping is not considered in this dissertation.

Assumptions

The following assumptions are made in [VV02] for the existence results.

A1 - V is dense in W and W is dense in X.

A2 - There exists a positive constant C_W such that $\|w\|_X \leq C_W\|w\|_W$ for each $w \in W$.

A3 - There exists a positive constant C_V such that $\|v\|_W \leq C_V\|v\|_V$ for each $v \in V$.

A4 - The bilinear form a is non-negative, symmetric and bounded on V, i.e. there exists a positive constant K_a such that for $u, v \in V$,

$$|a(u,v)| \leq K_a\|u\|_V\|v\|_V.$$

In general, the bilinear form a in **A4** is defined on V. However it is possible for a to be defined on the space W and bounded by the norm $||\cdot||_W$, i.e. there exists a $k > 0$ so that

$$|a(u,v)| \leq k||u||_W||v||_W \qquad \text{for all} \;\; u, \; v \in W. \tag{2.2.2}$$

This is called **weak damping**. Note that (2.2.2) is trivially satisfied if $a = 0$.

2.2.2 Main results for existence and uniqueness

To start, the main results of [VV02] are presented. There are three existence theorems but Theorem 2 is important for this dissertation.

Theorem 1 (Main Result). Suppose assumptions **A1**-**A4** hold. If, for $u_0 \in V$ and $u_1 \in V$, there exists some $y \in W$ such that

$$b(u_0, v) + a(u_1, v) = c(y, v) \qquad \text{for all} \;\; v \in V, \tag{2.2.3}$$

then for each $f \in C^1(J, X)$, there exists a unique solution $u \in C^1(J, V) \cap C^2(J, W)$ for Problem GVar.

This theorem allows for a solution of the abstract variational problem Problem GVar, if the assumptions **A1**-**A4** is satisfied and the initial values u_0 and u_1 are admissible. It is not always easy to verify that (2.2.3) is satisfied.

In [VV02] the authors consider a special case of weak damping. Define a space $E_b \subset V$ where

$$E_b = \{x \in V \mid \text{there exists a } y \in W \text{ such that } c(y, v) = b(x, v) \text{ for all } v \in V\}.$$

It is proved in [VV02] that condition (2.2.3) is satisfied if $u_1 \in V$ and $u_0 \in E_b$ (the pair u_0, u_1 is admissible).

Theorem 2 (Weak Damping). If a is bounded with respect to the norm in W, then there exists a unique solution $u \in C^1(J, V) \cap C^2(J, W)$ for Problem GVar for each $u_0 \in E_b$. each $u_1 \in V$ and each $f \in C^1(J, X)$.

Remark In general, the bilinear form a is non-negative. However it is possible for a to be positive definite on the space V with respect to the norm $|| \cdot ||_V$, i.e. there exists a $c_a > 0$ so that

$$a(v, v) \geq c_a||v||_V^2 \qquad \text{for all} \;\; v \in V.$$

This is called strong damping. It is not considered in this dissertation.

Theorem 3 (Strong Damping). If a is positive definite on V, there exists a unique solution $u \in C^1([0, \infty), V) \cap C^2((0, \infty), W)$ for Problem GVar for any $u_0 \in V$, $u_1 \in W$ and any f which is Lipschitz on V. If $f = 0$, then $u \in C^1([0, \infty), V) \cap C^2([0, \infty), W) \cap C^\infty((0, \infty), V)$.

As mentioned, in the models of this dissertation, the bilinear form a is identically zero. This automatically satisfies the weak damping condition and hence Theorem 2 may be applied.

2.2.3 First order system

To prove the main results in [VV02], the authors introduce an equivalent first order system.

This variational form is rewritten as a first order system of differential equations. Let $y(t) = u'(t)$, then

$$c(y'(t), v) + a(y(t), v) + b(u(t), v) = (f(t), v)_X.$$

To make this precise, a Hilbert space H is defined by $H := V \times W$. For $x \in H$, $x = \langle x_1, x_2 \rangle$ with $x_1 \in V, x_2 \in W$. An inner product on H is defined by

$$(x, y)_H := b(x_1, y_1) + c(x_1, y_1) \text{ for all } x, y \in H.$$

Then the authors define an operator Λ as a mapping on H by $\Lambda y = -x$ when $-x_2 = y_1$ and $x_1 \in V$ such that

$$b(x_1, v) + a(x_2, v) = c(y_2, v) \quad \text{for each} \quad v \in V. \tag{2.2.4}$$

The operator A is defined in [VV02] as $A = \Lambda^{-1}$ with $D(A) = \mathcal{R}(\Lambda)$.

From a result in [VV02], $x \in D(A)$ if and only if $x_1 \in V$, $x_2 \in V$ and there exists a $z \in W$ such that $b(x_1, v)+a(x_2, v) = c(z, v)$ for all $v \in V$. Furthermore $y = Ax$ if $y_1 = -x_2$ and

$$b(x_1, v) + a(x_2, v) = c(y_2, v) \quad \text{for all } v \in V.$$

This operator is used to rewrite the equation (2.2.1) of Problem GVar as a first order differential equation in the form

$$x' \;=\; Ax + f. \tag{2.2.5}$$

To be more precise, the following problem is introduced.

Problem IVP

Given a function $F : J \to H$, find a function $U \in C(J,\ H)$ such that for each $t \in J$, $U(t) \in D(A)$, $U(t) \in H$ and

$$\begin{aligned} U(t)' &= AU(t) + F(t), \\ U(0) &= U_0. \end{aligned}$$

To link Problem IVP and Problem GVar, consider the following lemma from [VV02].

Lemma 1. Suppose $F(t) = \langle 0, f(t) \rangle$ for each $t \in J$.

a) If u is a solution of Problem GVar, then $U = \langle u, u' \rangle$ is a solution of Problem IVP, with $U_0 = \langle u_0, u_1 \rangle$.

b) If U is a solution of Problem IVP with $U_0 = \langle u_0, u_1 \rangle$, then the first component $u = U_1$ of U is a solution for Problem GVar.

Semi-group theory is used in [VV02] to investigate the solvability of Problem IVP and obtain a solution for Problem IVP. It is also important to mention that in [VV02], the authors provide the necessary result that shows the function F is uniquely defined by f.

As mentioned before, if $a = 0$ then the inequality in (2.2.2) hold trivially. However, the operator A is defined by the operator Λ and in the definition of Λ, the form a is used (see (2.2.4)). It is proved in [VV02] that equation (2.2.4) is uniquely solvable and hence Λ is well defined. The proof remains unchanged as presented for a identically zero.

To apply Theorem 2, the admissible initial conditions are $u_0 \in E_b$ and $u_1 \in V$. As mentioned above, this implies $\langle u_0, u_1 \rangle \in D(A)$. From semigroup theory, it follows that $U(t) \in D(A)$ for each t. Therefore $u(t) \in E_b$ and $u'(t) \in V$ for each t.

Using the assumptions **A1**-**A4**, it is proved in [VV02] that the linear operator A is an infinitesimal generator of a C_0-semigroup of contractions and the domain of A is dense in H (see Section 2.4).

2.3 Application: Timoshenko beam model

In this section the theory of [VV02] (presented in Section 2.2) is applied to Problem T-2W. This is an continuation of the example from Section 2.1.

Recall the spaces defined in Section 2.1:

$X = L^2(0,1) \times L^2(0,1)$ with inner product $(\cdot,\cdot)_X$ and induced norm $||\cdot||_X$.

$W = X$ with inner product c and induced norm $||\cdot||_W$.

$V = V(0,1) \times V(0,1)$ with inner product b and induced norm $||\cdot||_V$.

To apply the theory, Problem T-2W must satisfy assumptions **A1**-**A4** and the initial values must be admissible. We must show that the assumptions are satisfied.

To prove **Assumption A1**, observe that W is dense in X. (The set $W = X$).

Let $u \in C_0^\infty(0,1)$. Then $u(0) = u(1) = 0$ and therefore $u \in T(0,1)$ and $C_0^\infty(0,1) \subset T(0,1)$. Recall that $V(0,1)$ is the closure of $T(0,1)$ in $H^1(0,1)$. So it follows that $C_0^\infty(0,1) \subset V(0,1) \subset H^1(0,1)$. And since $C_0^\infty(0,1)$ is dense in $L^2(0,1)$, both $V(0,1)$ and $H^1(0,1)$ are dense in $L^2(0,1)$. Therefore $V(0,1) \times V(0,1)$ is dense in $X = L^2(0,1) \times L^2(0,1)$.

Consider **Assumption A2**. From the definition of the bilinear form c,

$$c(f,f) \;=\; \int_0^1 (f_1)^2 + \frac{1}{\alpha}\int_0^1 (f_2)^2 = ||f_1||^2 + \frac{1}{\alpha}||f_2||^2.$$

From this and the definition of the X norm, the following inequalities can be obtained:

$$\min\left\{1, \frac{1}{\alpha}\right\} ||f||_X^2 \le c(f,f) \le \max\left\{1, \frac{1}{\alpha}\right\} ||f||_X^2.$$

Since c is a bilinear form and by the inequality above, c is an inner-product for W. The norm of W is defined as $||\cdot||_W = \sqrt{c(\cdot,\cdot)}$. Let $C_1 = \min\left\{1, \frac{1}{\alpha}\right\}$ and $C_2 = \max\left\{1, \frac{1}{\alpha}\right\}$. Then

$$C_1||x||_X \le ||x||_W \le C_2||x||_X \tag{2.3.1}$$

for all $x \in X$.

To prove **Assumption A3**, some preparation is required. Consider the following proposition for a Poincaré-Type inequality for the one-dimensional case.

Proposition 1. Suppose that $f \in C^1[a,b]$, and $f(a) = 0$. Then $||f|| \leq (b-a)||f'||$.

Proof. Let $f \in C^1(a,b)$ such that $f(a) = 0$. By the Fundamental Theorem of Calculus,

$$|f(x)| = \left| \int_a^x f'(s)\ ds \right| \leq \int_a^x |f'(s)|\ ds.$$

Since x is arbitrary,

$$||f||_{\sup} \leq \int_a^b |f'(s)| ds.$$

Also for $f, g \in C^1(a,b)$,

$$\left| \int_a^b fg \right| \quad \leq \quad ||f||\ ||g|| \qquad \text{by Cauchy-Swartz Inequality.} \tag{2.3.2}$$

Choose $g = 1$ then since $||f|| \leq ||f||_{\sup}\sqrt{b-a}$, the result follows. □

Corollary. Suppose that $f \in H^1(a,b)$, and $f(a) = 0$. Then $||f|| \leq (b-a)||f'||$.

Proof. There exists a sequence $(g_n) \subset C^1[a,b]$ such that $||g_n - f|| \to 0$ and $||g_n' - f'|| \to 0$.

Therefore $\dfrac{||f||}{||f'||} = \lim\limits_{n \to \infty} \dfrac{||g_n||}{||g_n'||} \leq (b-a)$. □

Theorem 1. There exists a positive constant C_V such that $\|v\|_W \leq C_V \|v\|_V$ for each $v \in V$.

Proof. Let $f \in V$, then $f_1(0) = f_2(0) = 0$. Following from the corollary, $||f_1|| \leq ||f_1'||$ and $||f_2|| \leq ||f_2'||$. Therefore,

$$\begin{aligned} ||f_1'|| &= ||f_1' - f_2 + f_2||, \\ &\leq ||f_1' - f_2|| + ||f_2||, \\ &\leq ||f_1' - f_2|| + ||f_2'||. \end{aligned}$$

It follows that $||f_1'||^2 \leq 2||f_1' - f_2||^2 + 2||f_2'||^2$, since $(a+b)^2 \leq 2a^2 + 2b^2$. Hence, from the definition of the norm $||\cdot||_X$, and again from the corollary,

$$\begin{aligned} ||f||_X^2 &= ||f_1||^2 + ||f_2||^2, \\ &\leq ||f_1'||^2 + ||f_2'||^2, \\ &\leq 2||f_1' - f_2||^2 + 3||f_2'||^2. \end{aligned}$$

It follows from the inequality (2.3.1) that

$$\|f\|_W^2 \leq 2C_2||f_1' - f_2||^2 + 3C_2||f_2'||^2. \tag{2.3.3}$$

But we also have

$$\begin{aligned} b(f,f) &= \int_0^1 (f_1' - f_2)^2 + \frac{1}{\beta}\int_0^1 (f_2')^2, \\ &= ||f_1' - f_2||^2 + \frac{1}{\beta}||f_2'||^2, \\ &\geq \min\left\{1,\ \frac{1}{\beta}\right\}\left(||f_1' - f_2||^2 + ||f_2'||^2\right). \end{aligned} \tag{2.3.4}$$

Now combine the inequalities (2.3.3) and (2.3.4). There exists a positive constant C_V such that

$$||f||_W^2 \leq C_V b(f,f),$$

for all $f \in V$. And since b is a bilinear form, and also positive definite in V, b is an inner product for V. $\square$

Consider **Assumption A4**. In Problem T-2W, it is clear that the bilinear form a is identically zero. As mentioned before, if $a = 0$, the weak damping as well as assumption **A4** is satisfied. Also, as explained in Section 2.2 the constuction of the operator A is not affected.

2.4 Modal analysis

2.4.1 Timoshenko beam

To understand what is meant by modal analysis, it is in the first place necessary to consider modes of vibration. And to understand how modal analysis is applied, a Timoshenko beam is used as an example. Specifically, Problem T-1 as the boundary conditions makes it easy to illustrate the idea of modal analysis. A detailed approach to modal analysis for the Timoshenko beam theory is discussed in Chapter 4.

Consider the partial differential equations of a pinned-pinned Timoshenko beam, obtained by substituting the constitutive equations (1.3.7) and (1.3.8) into the equations of motion (1.3.5) and (1.3.6):

$$\begin{aligned} \partial_x^2 w - \partial_x \phi &= \partial_t^2 w, && (2.4.1) \\ \alpha \partial_x w - \alpha \phi + \frac{1}{\gamma} \partial_x^2 \phi &= \partial_t^2 \phi, && (2.4.2) \end{aligned}$$

with boundary conditions

$$\begin{aligned} w(0,t) &= 0, \qquad & w(1,t) &= 0, \\ M(0,t) &= 0, \qquad & M(1,t) &= 0. \end{aligned}$$

The boundary conditions of the moments can be written in terms of ϕ using the constitutive equation (1.3.7):

$$\partial_x \phi(0,t) = 0, \qquad \partial_x \phi(1,t) = 0.$$

Consider a trial solution $w(x,t) = T(t)\tilde{w}(x)$ and $\phi(x,t) = T(t)\tilde{\phi}(x)$. Substituting these trial solutions in (2.4.1) and (2.4.2) yields

$$\begin{aligned} T(t)\tilde{w}''(x) - T(t)\tilde{\phi}'(x) &= T''(t)\tilde{w}(x), \\ \alpha T(t)\tilde{w}'(x) - \alpha T(t)\tilde{\phi}(x) + \frac{1}{\gamma} T(t)\tilde{\phi}''(x) &= T''(t)\tilde{\phi}(x). \end{aligned}$$

Dividing by $T(t)$,

$$\begin{aligned} \tilde{w}''(x) - \tilde{\phi}'(x) &= \frac{T''(t)}{T(t)}\tilde{w}(x), \\ \alpha \tilde{w}'(x) - \alpha \tilde{\phi}(x) + \frac{1}{\gamma}\tilde{\phi}''(x) &= \frac{T''(t)}{T(t)}\tilde{\phi}(x). \end{aligned}$$

Therefore $\dfrac{T''(t)}{T(t)}$ is constant since the left hand side does not depend on t. Suppose $\dfrac{T''(t)}{T(t)} = -\lambda$. This can also be written as the following differential equation,

$$\ddot{T} + \lambda T = 0. \qquad (2.4.3)$$

To determine if such a number λ exists, consider the following eigenvalue problem.

Problem T-1E

Find $\lambda \in R$ and functions $\tilde{w}$ and $\tilde{\phi}$ such that

$$\begin{aligned} -\tilde{w}'' + \tilde{\phi}' &= \lambda \tilde{w}, \\ -\alpha \tilde{w}' + \alpha \tilde{\phi} - \frac{1}{\gamma} \tilde{\phi}'' &= \lambda \tilde{\phi}, \end{aligned}$$

with boundary conditions $\tilde{w}(0) = \tilde{w}(1) = 0$ and $\tilde{\phi}'(0) = \tilde{\phi}'(1) = 0$.

Solutions to Problem T-1E is a pair of functions $\langle \tilde{w}, \tilde{\phi} \rangle$ called the eigenfunction with corresponding eigenvalue λ. Substitution of $\tilde{w} = \sin(k\pi x)$ and $\tilde{\phi} = \cos(k\pi x)$ show that they are solutions of the differential equations and satisfy the boundary conditions.

Clearly the eigenvalue problem has infinitely many solutions $\langle \tilde{w}_n, \tilde{\phi}_n \rangle$ and corresponding eigenvalues λ_n. Another solution is the vector $\langle \tilde{w}, \tilde{\phi} \rangle = \langle 0, 1 \rangle$ which also satisfies the eigenvalue problem with boundary conditions.

A systematic approach to obtain the eigenvalues and eigenfunctions is given in [VV06]. This is discussed in more detail in Section 4.2. To verify the trial solutions for Problem T-1E, the following theorem can be derived from [VV06].
Theorem 1. If $\langle u, \phi \rangle$ is a non-constant eigenfunction of Problem T-1E, then $\langle u, \phi \rangle = \langle \sin k\pi x, A_k \cos k\pi x \rangle$ which satisfies the boundary conditions. A_k is a constant depending on the integer k and the eigenvalue λ_k. If $\langle u, \phi \rangle$ is a constant eigenfunction of Problem T1-E, then $\langle u, \phi \rangle = \langle 0, 1 \rangle$. Also all eigenvalues less than α are simple eigenvalues.

Proof. See Section 4.2. □

Corresponding to this sequence of eigenfunctions, we have a sequence of solutions $(T_n(t))$:

$$T_n(t) = A_n \cos(\sqrt{\lambda_n} t) + B_n \sin(\sqrt{\lambda_n} t), \tag{2.4.4}$$

(with A_n and B_n arbitrary constants) for the ordinary differential equation (2.4.3).

Combining the solutions of the eigenvalue problem with the solution (2.4.4) yields:

$$\begin{aligned} w_n(x,t) &= T_n(t) \tilde{w}_n(t), \\ \phi_n(x,t) &= T_n(t) \tilde{\phi}_n(x). \end{aligned}$$

Substitution show that these solutions do indeed satisfy the differential equations (2.4.1) and (2.4.2) with the boundary conditions. These solutions are the modal solutions and they are clearly periodic with natural angular frequencies $\sqrt{\lambda_n}$ (and consequently the natural frequencies are $\frac{\sqrt{\lambda_n}}{2\pi}$).

Therefore the formal series solution for Problem T-1 is

$$\langle u, \phi \rangle = \sum_{n=1}^{\infty} T_n(t) \langle u_n(x), \phi_n(x) \rangle.$$

The variational eigenvalue problem for Problem T-1E can be obtained using similar steps to how Problem T-1V is obtained. This variational eigenvalue problem is referred to as Problem T-1EV

Problem T-1EV

Find a pair of functions $\tilde{w}$ and $\tilde{\phi}$ such that for all $t > 0$, $\langle \tilde{w}\ \tilde{\phi} \rangle \in T[0,1]$ and satisfying the equations

$$\begin{aligned} -\int_0^1 \tilde{w}'v' + \int_0^1 \tilde{\phi}v' &= \int_0^1 \lambda \tilde{w} v, \\ -\int_0^1 \alpha \tilde{w}'\psi + \int_0^1 \alpha \tilde{\phi}\psi - \int_0^1 \frac{1}{\gamma}\tilde{\phi}'\psi' &= \int_0^1 \lambda \tilde{\phi}\psi. \end{aligned}$$

for each $\langle v,\ \psi \rangle \in T[0,1]$.

2.4.2 General vibration problem

This section is a discussion of the general vibration problem GVar. The discussion will refer to the article [CVV18]. In this article, the authors take damping into consideration, which is not covered in this dissertation.

Consider Problem GVar defined in Section 2.2.1. Assume there is no damping and no forcing.

Problem GVar

Find a function $u \in C(J,\ X)$ such that u' is continuous at 0 with respect to $\|\cdot\|_W$, and for each $t \in J$, $u(t) \in V$, $u'(t) \in V$, $u''(t) \in W$, satisfying

$$c(u''(t), v) + b(u(t), v) = 0 \quad \text{for each } v \in V, \tag{2.4.5}$$

with initial conditions $u(0) = u_0$ and $u'(0) = u_1$.

To try and find a solution to this problem, consider a trial solution $u(t) = T(t)x$ with $x \in V$ and $x \neq 0$. Substituting this trial solution into (2.4.5) results in

$$b(T(t)x, v) = -c(T''(t)x, v).$$

Due to the linearity of the bilinear form, this equation can be rewritten as

$$T(t)b(x, v) = -T''(t)c(x, v).$$

Dividing both sides by $T(t)$ gives

$$b(x, v) = -\frac{T''(t)}{T(t)}c(x, v).$$

Therefore $-\frac{T''(t)}{T(t)}$ must be constant. Suppose that $\frac{T''(t)}{T(t)} = -\lambda$. The existence of such a λ is uncertain at this point. So we consider the following eigenvalue problem.

Find a real number λ and a $x \in V$ with $x \neq 0$ such that

$$b(x, y) = \lambda c(x, y) \qquad \text{for each} \quad y \in V.$$

A solution to the eigenvalue problem consists of an eigenvalue λ with corresponding eigenvector x.

The article [CVV18] is a convenient reference to show that the eigenvalue problem has a solution if the following assumption, aditional to **A1**, **A2**, **A3** and **A4**, is satisfied:

A5 - The embedding of V into W is compact.

(This assumption expands on the assumptions **A1**-**A4** already made in Section 2.2.)

Using these assumptions, the authors of [CVV18] prove that there exists a complete orthonormal sequence of eigenvectors for the eigenvalue problem with a corresponding sequence of eigenvalues. These eigenvalues are positive and the orthogonality is with respect to the bilinear form c. In fact it is also orthogonal with respect to the bilinear form b,

$$b(x_i, x_j) = \lambda_i c(x_i, x_j) = 0, \qquad \text{for each} \quad i \neq j.$$

Also the sequence of normalized eigenvectors x_i forms an orthonormal basis in W and sequence of eigenvalues λ_i is an infinite sequence with $\lambda_n \to \infty$ as $n \to \infty$.

Following these results from [CVV18], for any $u \in V$, $u = \sum_{i=1}^{\infty} a_i x_i$. These coefficients a_i are generalized Fourier coefficients of u with respect to the eigenvectors x_i. Therefore, for any $u \in V$,

$$u = \sum_{i=1}^{\infty} a_i x_i = \sum_{i=1}^{\infty} c(u, x_i) x_i.$$

Now that the eigenvalue problem has many solutions, the following ordinary differentiable equation can be considered,

$$T_n'' + \lambda T_n = 0.$$

Since this differential equation is a simple second order differential equation, $T_n(t)$ has the following possible solutions:

$$T_n(t) = A_n \cos(\sqrt{\lambda_n} t) + B_n \sin(\sqrt{\lambda_n} t) \quad \text{if } \lambda_n > 0, \tag{2.4.6}$$

Then combining the solutions of the eigenvalue problem and the differential equation, the formal series solution for the boundary value problem is

$$u(t) = \sum_{n=1}^{\infty} T_n(t) x_n. \tag{2.4.7}$$

2.4.3 Validity of series solution

Consider the following question: When is the formal series solution valid for the vibration problem with initial values $u(0) = u_0$ and $u'(0) = u_1$? To answer this question, we again refer to the article [CVV18].

In the article, the validity of the series solution is proved using energy norms, where the energy $\mathcal{E}$ of the function u given by

$$\mathcal{E}(t) = \frac{1}{2} b(u(t), u(t)) + \frac{1}{2} c(u'(t), u'(t)). \tag{2.4.8}$$

Then,

$$\mathcal{E}'(t) \;=\; b(u(t), u'(t)) + c(u'(t), u''(t)) = 0 \text{ following from (2.4.5).}$$

Therefore $\mathcal{E}(t) = \mathcal{E}(0)$ for all $t > 0$. For the case of weak damping, it can be proved that $\mathcal{E}(t) \leq \mathcal{E}(0)$ for all $t > 0$ which is the case of [CVV18].

Denote the partial sum $u^N(t) = \sum_{n=1}^{N} T_n(t)w_n$ where w_n are is the n'th eigenfunction. Ideally

$$u_0^N = \sum_{n=1}^{N} T_n(0)w_n \quad \text{and} \quad u_1^N = \sum_{n=1}^{N} T_n'(0)w_n,$$

but T_n is not uniquely defined by the differential equation.

Substitution shows that these partial sums with initial conditions $u^N(0) = u_0^N$ and $(u^N)'(0) = u_1^N$ are solutions for (2.4.5) in Problem GVar.

The authors then define an error function $u_N^E = u - u^N$. Since both u and u^N satisfies (2.4.5), so does this error function with the following the initial conditions, $u_N^E(0) = u_0 - u_0^N$ and $(u_N^E)'(0) = u_1 - u_1^N$.

Let $\mathcal{E}$ denote the energy of the error function u_N^E. Since $\mathcal{E}(t) = \mathcal{E}(0)$ for all $t > 0$ it follows that,

$$||u(t) - u^N(t)||_V^2 + ||u'(t) - (u^N)'(t)||_W^2 = ||u_0 - u_0^N||_V^2 + ||u_1 - u_1^N||_W^2 \tag{2.4.9}$$

Now, u_0 and u_1 are given in Problem GVar. Therefore the generalized Fourier coefficients for u_0 and u_1 must be used to compute u_0^N and u_1^N. These Fourier coefficients are $u_0^N = \displaystyle\sum_{n=1}^{N} b(u_0, w_n)w_n$ and $u_1^N = \displaystyle\sum_{n=1}^{N} c(u_1, w_n)w_n$.

Then $||u_0 - u_0^N||_V^2 \to 0$ and $||u_1 - u_1^N||_W^2 \to 0$ as $N \to \infty$. Therefore $\mathcal{E}(t) \to 0$ as $N \to \infty$ by (2.4.9).

It follows that the partial sums of the series solution converges to the solution u as the initial conditions u_0^N and u_1^N converges to u_0 and u_1 respectively.

2.4.4 Comparison of models

In this dissertation, the objective is to compare different linear beam and plate models for use in applications. The first of the comparisons are in Section 4.6. This section is a discussion of the article [SP06] comparing a Timoshenko beam model and a three dimensional beam model to empirical results. The authors of this article use the natural frequencies (equivalently the eigenvalues) to compare the different models.

Sections 4.5 and 6.2 are a discussion and extension of the article [LVV09a]. This article compares a Timoshenko beam model to a two-dimensional beam model. This is then extended to a comparison of a two-dimensional beam model to a three-dimensional beam model as well as a comparison of a two-dimensional plate model to a three-dimensional plate model.

Being able to express the solutions as valid series solutions, enable us to compare the different models by only considering the eigenvalues and eigenfunctions of the different models. This is explained in detail in [CVV18].

In [CVV18], the authors consider a beam model and wave equation model for a vibrating string. Suppose that for the same initial conditions, u_b and u_w are the exact solutions. A comparison of the exact solutions is not possible, but the partial sums can be compared if the bounds for the errors $u_b - u_b^N$ and $u_w - u_w^N$ can be guaranteed.

3 Finite element theory

In this chapter, two important convergence results of the Finite Element Method (FEM) are discussed. The first result looks at the convergence of the Galerkin approximation for second order hyperbolic type problems (or general vibation problems). The papers considered for the first result are [BV13] and [BSV17]. The second result is from the textbook [SF73] and examines the convergence of eigenvalues and eigenfunctions for a vibration problem, using the Finite Element Method.

3.1 Galerkin approximation for second order hyperbolic type problems

In the article [BV13], the authors investigate the convergence of the Galerkin approximation for second order hyperbolic type problems. The article [BSV17] extends the work of [BV13] by including general damping and damping at the endpoints. For the models in Chapter 1, [BV13] is sufficient while [BSV17] provides more insight and improved notation.

In Section 2.2 of this dissertation, Problem GVar is presented. It is identical to the problem in [BV13]. For convenience, the problem is repeated here.

3.1.1 Formulation of the Galerkin approximation

Recall the spaces V, W and X from Section 2.2 where $V \subset W \subset X$.

Problem GVar

Given a function $f : J \to X$, find a function $u \in C(J,\ X)$ such that u' is continuous at 0 with respect to $\|\cdot\|_W$ and for each $t \in J$, $u(t) \in V$, $u'(t) \in$

$V,\ u''(t) \in W$ and

$$c(u''(t), v) + a(u'(t), v) + b(u(t), v) = (f(t), v)_X \quad \text{for each } v \in V, \quad (3.1.1)$$

while $u(0) = u_0$, $u'(0) = u_1$.

Assume that the assumptions **A1-A4** from Section 2.2 are satisfied ensuring that Problem GVar has a unique solution.

Before the theory of [BV13] can be discussed, some preliminary work is necessary. The structure of this section is as follows. First the Galerkin approximation for Problem GVar is derived. Then an equivalent system of ordinary differential equations is derived using the Finite Element Method. Finally, the convergence of the Galerkin approximation is discussed using the work of the article [BV13].

Consider the example of the cantilever Timoshenko beam model from Section 2.1. In this section the variational problem Problem T-2V is given in terms of bilinear forms.

Problem T-2V

Find a function $u \in T[0,1]$ such that for all $t \geq 0$,

$$c(u''(t), v) + b(u(t), v) \quad = \quad (Q(t), v),$$

for each $v \in T[0,1]$.

The interval $[0,1]$ is divided into n equal subintervals $[x_i, x_{i+1}]$, each of length $h = \dfrac{1}{n}$, such that $x_i = ih$ for $i = 0, 1, ..., n$.

Consider a set of $n+1$ linear independent, piecewise linear basis functions δ_i. The subset of these functions that satisfies the boundary conditions are called admissible basis functions. For Problem 2-T, the admissible basis functions are δ_i for $i = 1, 2, ..., n$. Define the space S^h as the space spanned by the admissible basis functions, i.e.

$$S^h = \text{span}\{\delta_1, \delta_2, ..., \delta_n\}.$$

This space $S^h \times S^h$ is a finite dimensional subspace of $T[0,1]$. Define the following functions $w^h \in S^h$ and $\phi^h \in S^h$ as

$$\begin{aligned} w^h(t) &= \sum_{i=1}^{n} w(x_i^*, t)\delta_i(t), \\ \phi^h(t) &= \sum_{i=1}^{n} w(x_i^*, t)\delta_i(t), \end{aligned}$$

where $x_i^* \in [x_i, x_{i+1}]$. Then let $u_h = (w^h, \phi^h)$.

Using these functions, the Galerkin approximation for Problem T-2, referred to as Problem T-2V^h, can be derived.

Problem T-2V^h

Find a function $u_h \in S^h \times S^h$ such that for all $t \geq 0$,

$$c(u_h''(t), v) + b(u_h(t), v) = (Q^I(t), v),$$

for each $v \in S^h \times S^h$. For each t, $Q^I(t)$ is the interpolant of $Q(t)$ in S^h.

This example serves as an illustration of the derivation of the Galerkin approximation and the convention of symbols before the general case is presented. Piecewise linear basis functions are used for this example, but other basis functions can be used. In Chapter 5, the basis functions used are piecewise cubic Hermite polynomials.

For the general case presented below, S^h is a finite dimensional subspace of V.

Problem GVar^h

Given a function $f : J \to X$, find a function $u_h \in C^2(J, S^h)$ such that for each $t \in J$

$$c(u_h''(t), v) + a(u_h'(t), v) + b(u_h(t), v) = (f(t), v)_X \quad \text{for each } v \in S^h, \tag{3.1.2}$$

with the initial values $u_h(0) = u_0^h$ and $u_h'(0) = u_1^h$. The initial conditions u_0^h and u_1^h are projections of u_0 and u_1 in the finite dimensional space S^h.

3.1.2 System of ordinary differential equations

Problem GVar^h is equivalent to a system of second order differential equations. Consider the standard FEM matrices defined by

$$\begin{aligned} K_{ij} &= b(\phi_j, \phi_i), \\ C_{ij} &= a(\phi_j, \phi_i), \\ M_{ij} &= c(\phi_j, \phi_i), \\ F_i(t) &= c(f(t), \phi_i), \end{aligned}$$

where ϕ_i and ϕ_j are admissible basis functions.

Using these matrices, Problem GVar^h is rewritten as a system of ordinary differential equations denoted by Problem ODE.

Recall that $u^h(t) = \sum_k u_k(t)\phi_k$ where $\bar{u}_k = (u_1(t), u_2(t), ..., u_n(t))$ where each ϕ_k corresponds to a node number k. More complex cases are treated in Chapter 5.

Problem ODE

Find a function $\bar{u} \in S^h$ such that

$$M\bar{u}'' + C\bar{u}' + K\bar{u} = F(t) \quad \text{with } \bar{u}(0) = \bar{u}_0 \text{ and } \bar{u}(1) = \bar{u}_1 \tag{3.1.3}$$

The following propositions related to Problem ODE are given in [BV13].

Proposition 1. If $F \in C(J)$, then Problem ODE has a unique solution for each pair of vectors $\bar{u}_0$ and $\bar{u}_1$

Proposition 2. Suppose $M, K, C, F, \bar{u}_0$ and $\bar{u}_1$ are defined as above. Then, the function u_h is a solution of Problem GVar^h if and only if the function $\bar{u}$ is a solution of Problem ODE.

Proposition 2 provides a link between the solution of Problem ODE and the solution of Problem GVar^h. Theorem 1 below follows.

Theorem 1. If $f \in C(J, X)$, then there exists a unique solution $u_h \in C^2(J, S^h)$ for Problem GVar^h for each u_0^h and u_1^h in S^h. If $f = 0$ then $u_h \in C^2((-\infty, \infty))$.

It is now required to find an approximation for the solution of $\bar{u}$ of Problem ODE.

Consider the time interval $J = [0, T]$. Divide J into N steps with length $\tau = \frac{T}{N}$. Each interval can be expressed as $[t_{k-1},\ t_k]$ for $k = 1, ..., N$. Denote the approximation of u_h on the interval $[t_{k-1},\ t_k]$ by u_k^h, i.e. $u_h(t_k)$ corresponds to u_k^h for each k. A finite difference method is used to compute u_k^h for each k.

3.1.3 Error estimates

In this subsection we consider estimates for the error $u(t_k) - u_k^h$. To simplify the process, the error is divided into errors for the semi-discrete problem and the fully discrete problem.

Under the assumptions **A1**-**A4** of Chapter 2 and continuity of f, there exists a unique solution for Problem GVar^h. The next step is to show that the solution of Problem GVar^h converges to the solution of Problem GVar.

Let u be the solution of Problem GVar and u^h be the solution of Problem GVar^h. The authors of [BV13] define the following error,

$$e^h(t) = u(t) - u^h(t). \tag{3.1.4}$$

In [BV13] it is assumed that there exists a subspace H of V and a positive integer α such that

$$\inf_{v \in S^h} \|w - v\|_V \leq Ch^\alpha |||w|||_H,$$

for each $w \in H$ where $|||w|||_H$ is a norm or semi-norm for H.
Theorem 2. If $u(t) \in H$ and $u'(t) \in H$ for each t, then

$$||e^h(t)||_W \quad \leq \quad Ch^\alpha \left(|||u(t)||_H + |||u'(t)||_H \right),$$

for each t.

From Theorem 2 for the semi-discrete problem an error estimate for $e(t) = u(t) - u_h(t)$ with respect to the norm of W was obtained. The authors of [BV13] then proceed to obtain an error estimate for $e_k = u_h(t_k) - u_k^h$.

The error can then be expressed as

$$e(t_k) = u(t_k) - u_k^h \quad = \quad [u(t_k) - u_h(t_k)] + [u_h(t_k) - u_k^h]. \tag{3.1.5}$$

In (3.1.5), the the error for the semi-discrete problem is the term $u(t_k) - u_h(t_k)$ and the term $u_h(t_k) - u_k^h$ is the error for the fully disrete appreximation of the semi-discrete approximation.

Since the dimension of S^h is not fixed, the equivalence of norms cannot be used, and therefore this error estimate for $e_k = u_h(t_k) - u_k^h$ should also be with respect to the norm of W. The local error e_1 can be estimated using Taylor polynomials, but then e_k 'grows' as k increases.

A stability result is derived in [BV13]. Recall that $\tau = \dfrac{T}{N}$.

Lemma.

$$\max \|e_n\|_W^2 \leq KT\tau$$

where K depends on u, u_h and their derivatives.

Using this lemma, [BV13] prove the error estimate. The error estimate for the term $u_h(t_k) - u_k^h$ with respect to the norm of W as proven by the authors of [BV13] is presented here as Theorem 3.
Theorem 3. If $f \in C^2([0,T], X)$, then

$$\|u_h(t_k) - u_k^h\|_W \quad \leq \quad K\tau^2$$

for each $t \in (0,\ T)$.

3.1.4 Main result

Finally, Theorem 2 of the semi-discrete problem and Theorem 3 of the fully discrete problem gives an error estimate for the error $e(t)$. Consequently, the error estimate $e^h(t) = u(t) - u^h(t)$ is obtained.

The main result proving the convergence of the solution of the Galerkin Approximation is given in [BV13] as follows.
Theorem 4. Main Result
If $f \in C^2([0,T], X)$, then

$$\|u(t_k) - u_k^h\|_W \quad \leq \quad K\tau^2$$

for each $t \in (0,\ T)$.

The constant K depend on u, u_h and their derivatives.

3.2 FEM computation of eigenvalues and eigenfunctions

Let W be a Hilbert space with the inner product $c(\cdot,\cdot)$ and induced norm $||\cdot||_W$. In [SF73], the authors use a Hilbert space H, with inner product $(\cdot,\cdot)$ and induced norm $||\cdot||$. We remain with the notation that is consistent with this dissertation. Let V be a linear subspace of W, with inner product defined by the bilinear form $b(\cdot,\cdot)$. It is assumed that the bilinear form b is symmetric and that the assumptions in Section 2.4 holds.

The following eigenvalue problem is considered in [SF73]. The same problem was treated in Section 2.4, although the notation differs slightly.

Problem E

Find a vector $u \in V$ and number $\lambda \in R$ such that $u \neq 0$ and

$$b(u, v) = \lambda c(u, v) \tag{3.2.1}$$

for each $v \in V$.

Recall that the eigenvectors can be ordered in such a way that

$$\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq ...$$

where λ_i are the corresponding eigenvalues.

Properties of eigenvalues

Since any multiple of a eigenfunction is still an eigenfunction, the eigenfunctions can be normalized so that $||u_i||_W = 1$ for all i.

Let $\{\phi_k \in V \mid k = 1, 2, ..., N_e\}$ be a set of linear independent admissible basis functions. Define $S^h := \text{span}\,\{\phi_k \in V \mid k = 1, 2, ..., N_e\}$ so that S^h is a finite dimensional subspace of V.

Consider the Galerkin approximation for (3.2.1):

Problem E^h

Find $u^h \in S^h$ such that $u^h \neq 0$ (with corresponding eigenvalue λ^h) and

$$b(u^h, v) = \lambda^h c(u^h, v) \text{ for all } v \in S^h.$$

For examples, see Chapter 5.

Problem E^h can be written as a matrix eigenvalue problem,

$$\lambda^h M\bar{u}_n = K\bar{u}_n. \tag{3.2.2}$$

Since N_e is never small and usually large to very large, a compute algorithm is required to calculate the eigenvalues (and eigenfunctions) of (3.2.2).

The pair $(\lambda^h, \bar{u}_n)$ correspond to the pair (λ_k^h, u_k^h) which is the solution of Problem E^h. It is necessary to understand some of the theory to make the connection.

In S^h, the ordering of vectors is the same as in the original space. Denote the normalized eigenvectors in the space S^h as u_k^h with corresponding eigenvalues λ_k^h for $k = 1, 2, ..., N_e$.

In Section 3.3 it is proved that the error $|\lambda_k^h - \lambda_k|$ is large when k is large. It is for example possible that $|\lambda_1^h - \lambda_1|$ is sufficiently small while λ_k^h cannot even be considered as an approximation for λ_k when $k > \dfrac{1}{2}N_e$.

Finally, any subspace of S^h will also be a subspace of V. So the minmax principle applies and a lower bound for the approximate eigenvalues hold [SF73]:

$$\lambda_i \leq \lambda_i^h. \tag{3.2.3}$$

3.3 Estimating the eigenvalues.

In this section, the work in the textbook [SF73] is discussed. The results are the same as given in the textbook, however the proofs are expanded for greater clarity.

3.3.1 Projection of the eigenfunctions

Some theory is required before the main results can be proven. The theory is from [SF73].

Rayleigh quotient

$$R(v) = \frac{b(v,v)}{c(v,v)} \quad \text{for } v \in V. \tag{3.3.1}$$

Projection

If $u \in V$, then Pu is its projection in the subspace S^h.

$$b(u - Pu, v^h) = 0 \quad \text{for all } v \in S^h.$$

Let $E_j \in V$ denote the eigenspace spanned by the exact eigenvectors $\{u_1, u_2, ..., u_j\}$ for $j = 1, 2, ..., m$. Clearly $m \leq N_e$.

Consider the subspace S_j of S^h where

$$S_j = PE_j \quad \text{for } j = 1, 2, ..., m.$$

The elements Pu_j are the projections of the eigenfunctions u_j into the space S^h. These projections Pu_j are not necessarily equal to $u^h \in S^h$. In fact, u_j^h can be vastly different from u_j. The situation is not simple. It is possible that $Pu_k = 0$ for large k Assume that the dimension of S^h is large enough, substantially larger than m.

Let $B_m = \{u \in E_m \mid ||u||_W = 1\}$ and define $\mu_m = \inf \{(Pu, Pu \mid u \in B_m)\}$.

The first step to obtain estimates for the eigenvalues, it to show that the elements of B_m are linearly independent. In the first part we follow the approach in [Zie00]. The author introduced the quantity μ_m above.
Proposition 1. $\mu_m > 0$ if and only if $\dim S_m = m$.

Proof. To show that the dimension of $S_m = m$, suppose that the elements of B_m are linearly dependent. Then there exists a $u \in B_m$ such that $Pu = 0$ and consequently $\mu_m = 0$. The result follows from the contra-positive. □

3.3.2 Upper bounds for approximate eigenvalues

Recall the definition of the Rayleigh quotient R in (3.3.1).
Proposition 2. $\lambda_m^h \leq \max R(Pu) \;\text{ for } u \in B_m$

Proof. Since $\dim S_m = m$, following from the minmax principle that

$$\lambda_m^h \leq \max R(v) \;\text{ for } v \in S_m. \tag{3.3.2}$$

Take an arbitrary nonzero $v \in S_m$. Then there exists a $Py \in E_m$ such that $v = Py$.

Now we take an arbitrary $v \in S_m$, $v \neq 0$. Then there exists a $u \in E_m$ such that $Pu = v$. This Pu is the projection into S_m of some $u \in E_m$ (which is also $\dfrac{1}{||u||_W} u \in B_m$).

Next we show that $R(||u||_W^{-1} u) = R(u)$:

$$\frac{b\left(\frac{1}{||u||_W}u, \frac{1}{||u||_W}u\right)}{c\left(\frac{1}{||u||_W}u, \frac{1}{||u||_W}u\right)} = \frac{b(u,u)}{c(u,u)} = R(u).$$

Finally, from (3.3.2)

$$\begin{aligned}\lambda_m^h &\leq \max R(v) \quad \text{for} \quad v \in S_m, \\ &= \max R(Pu) \quad \text{for} \quad u \in E_m, \\ &= \max R(Pu) \quad \text{for} \quad u \in B_m.\end{aligned}$$

$\square$

In the textbook, the authors show that the eigenfunctions are orthogonal. But they do so using matrix representations of the eigenvalue problem. A different method can be used to show this.

Pick any $i, j \in \mathbb{N}$ such that $i, j \leq m$. Then

$$\begin{aligned} b(u_i, \nu) &= \lambda_i c(u_i, \nu), \\ \text{and } b(u_j, \nu) &= \lambda_j c(u_j, \nu). \end{aligned}$$

for each $\nu \in V$. Clearly

$$\begin{aligned} b(u_i, u_j) &= \lambda_i c(u_i, u_j), \\ \text{and } b(u_j, u_i) &= \lambda_j c(u_j, u_i). \end{aligned}$$

Then using the symmetry of $b(\cdot, \cdot)$ and $c(\cdot, \cdot)$,

$$0 = (\lambda_i - \lambda_j) c(u_i, u_j).$$

So if $i \neq j$ then $\lambda_i \neq \lambda_j$. Therefore u_i and u_j are orthogonal when $\lambda_i \neq \lambda_j$.

The next steps in the textbook [SF73] contains proofs with multiple results. In an attempt to better understand the results, the proofs are broken up into smaller proofs.

Lemma 1. $\lambda_m^h \leq \dfrac{\lambda_m}{\mu_m^h}$

Proof. Consider the linearity of the bilinear form b, and fact that any $u \in E_m$ can be expressed as a linear combination $u = \sum_{i=1}^m c_i u_i$. Then

$$\begin{aligned} b(u, u) &= b\left(\sum_{i=1}^m c_i u_i, \sum_{j=1}^m c_j u_j\right), \\ &= \sum_{i=1}^m c_i \sum_{j=1}^m c_j b(u_i, u_j). \end{aligned}$$

The summation parameters can be merged into a single parameter. Then

$$\begin{aligned} b(u,u) &= \sum_{i=1}^{m} c_i^2 \lambda_i u_i, \\ &\leq \lambda_m \sum_{i=1}^{m} c_i^2 u_i, \\ &= \lambda_m ||u||_W^2. \end{aligned}$$

for all $u \in B_m$.

And since $B_m \subset E_m$, $b(Pu, Pu) \leq \lambda_m$ for all $u \in B_m$

Using the Rayleigh quotient, and the definition of μ_m^h,

$$\begin{aligned} R(Pu) &= \frac{b(Pu,Pu)}{c(Pu,Pu)}, \\ &= \frac{b(Pu,Pu)}{||Pu||_W^2}, \\ &\leq \frac{\lambda_m}{\mu_m^h}. \end{aligned}$$

Together with Proposition 2 it follows that

$$\lambda_m^h \leq \frac{\lambda_m}{\mu_m^h}.$$

□

3.3.3 The error bound

Following Lemma 1, and since $\lambda_i^h \geq \lambda_i$ it follows that $0 < \mu_m^h \leq 1$. It is now possible to define the 'error bound' in [SF73]:

$$\sigma_m^h = 1 - \mu_m^h. \tag{3.3.3}$$

Proposition 3. $0 \leq \sigma_m^h < 1$ and $\lambda_m^h - \lambda_m \leq \lambda_m^h \sigma_m^h$

Proof. Starting with the result of Lemma 1, $\lambda_m^h \mu_m^h \leq \lambda_m$.

Since $-\lambda_m \leq -\lambda_m^h \mu_m^h$, it follows that

$$\lambda_m^h - \lambda_m \leq \lambda_m^h - \lambda_m^h \mu_m^h = \lambda_m^h (1 - \mu_m^h).$$

$\square$

This result gives an error estimate for the eigenvalues. To prove the convergence of the eigenvalues, it is necessary to prove that the error estimate σ_m^h converges to zero as $h \to 0$.

Proposition 4. $\sigma_m^h = \max\{2c(u, u - Pu) - ||u - Pu||_W^2 \mid u \in B_m\}$

Proof. Let $u \in B_m$. Then

$$\begin{aligned} ||u - Pu||_W^2 &= c(u - Pu, u - Pu), \\ &= c(u, u) - 2c(u, Pu) + c(Pu, Pu), \\ &= 2c(u, u) - 2c(u, Pu) + c(Pu, Pu) - c(u, u), \\ &= 2c(u, u - Pu) + c(Pu, Pu) - c(u, u). \end{aligned}$$

Consequently,

$$c(u, u) - c(Pu, Pu) = 2c(u, u - Pu) - ||u - Pu||_W^2.$$

Since $u \in B_m$, $||u||_W^2 = 1$ and hence

$$1 - ||Pu||_W^2 = 2c(u, u - Pu) - ||u - Pu||_W^2.$$

On the right hand side, $1 - ||Pu||_W^2 \leq 1 - \mu_m^h = \sigma_m^h$ for all $u \in B_m$. Therefore

$$\sigma_m^h = \max\left\{2c(u, u - Pu) - ||u - Pu||_W^2 \mid u \in B_m\right\}.$$

$\square$

Proposition 4 is a result given in [SF73] without explaining how it is derived.

Introduce some new notation for convenience. For any $u \in E_m$, let $u^* = \sum_{i=1}^m c_i \lambda_i^{-1} u_i$ where $u = \sum_{i=1}^m c_i u_i$.

Proposition 5. For any $u \in E_m$

$$c(u, u - Pu) = b(u^* - Pu^*, u - Pu)$$

Proof. For any $i = 1, 2, ..., m$,

$$\begin{aligned}
\lambda_i c(u_i, u - Pu) &= b(u_i, u - Pu), \\
&= b(u_i, u - Pu) - b(u - Pu, Pu_i) \quad \text{(Rayleigh-Ritz Projection)}, \\
&= b(u_i, u - Pu) - b(Pu_i, u - Pu), \\
&= b(u_i - Pu_i, u - Pu).
\end{aligned}$$

Multiplying by $c_i\lambda_i^{-1}$ and summation over i gives:

$$\begin{aligned}
\sum_{i=1}^{m} c_i\lambda_i^{-1}\lambda_i c(u_i, u - Pu) &= \sum_{i=1}^{m} c_i\lambda_i^{-1} b(u_i - Pu_i, u - Pu), \\
&= b(\sum_{i=1}^{m} c_i\lambda_i^{-1}u_i - \sum_{i=1}^{m} c_i\lambda_i^{-1}Pu_i, u - Pu), \\
&= b(u^* - Pu^*, u - Pu).
\end{aligned}$$

Therefore $c(u, u - Pu) = b(u^* - Pu^*, u - Pu)$. □

Lemma 2. $\sigma_m^h \leq \max\{2||u^* - Pu^*||_W ||u - Pu||_W \mid u \in B_m\}$.

Proof. From Proposition 4,

$$\begin{aligned}
\sigma_m^h &= \max\left\{2c(u, u - Pu) - ||u - Pu||_W^2 \mid u \in B_m\right\}, \\
&\leq \max\{2c(u, u - Pu) \mid u \in B_m\}.
\end{aligned}$$

From Proposition 5,

$$\sigma_m^h = \max\{2b(u^* - Pu^*, u - Pu) \mid u \in B_m\}.$$

Using the Schwartz inequality,

$$b(u^* - Pu^*, u - Pu) \leq ||u^* - Pu^*||_W ||u - Pu||_W.$$

Finally,

$$\sigma_m^h \leq \max\{2||u^* - Pu^*||_W ||u - Pu||_W \mid u \in B_m\}.$$

□

3.3.4 Convergence of the eigenvalues

Assumption

For any $\epsilon > 0$ there exists a $\delta > 0$ such that if $h < \delta$, then

$$||u - Pu||_W < \epsilon \text{ for each } u \in B_m.$$

Remark: Pu is the closest element in S^h to u. In particular, $||u - Pu||_W \leq ||u - \Pi u||_W$, where Πu is the interpolant of u in S^h. The operator Π is treated in Section 3.5.

Lemma 3. For any $\epsilon > 0$ there exists a $\delta > 0$ such that

$$\sigma_m^h < \epsilon \text{ if } h < \delta. \tag{3.3.4}$$

Substitute the assumption into the estimate for σ_m^h in Lemma 2.

Lemma 4. There exists a $\delta > 0$ such that for $h < \delta$

$$\lambda_m^h - \lambda_m \leq 2\lambda_m \sigma_m^h. \tag{3.3.5}$$

Proof. Using Lemma 3, choose δ such that $\sigma_m^h < \frac{1}{2}$. Then (by Lemma 1) $\lambda_m^h < 2\lambda_m$ and therefore $\lambda_m^h - \lambda_m \leq 2\lambda_m \sigma_m^h$. □

The convergence of the eigenvalues follows from (3.3.4) and (3.3.5). An estimate of the error depends on an estimate for $u - \Pi u$, see Section 3.5

3.4 Convergence of the eigenfunctions

The next step is to show the convergence of the eigenfunctions. The problem can be formulated using the following result.

Lemma 5.

$$b(u_m - u_m^h, u_m - u_m^h) \;=\; \lambda_m c(u_m - u_m^h, u_m - u_m^h) + \lambda_m^h - \lambda_m.$$

Proof.

$$\begin{aligned} b(u_m - u_m^h, u_m - u_m^h) &= b(u_m, u_m) - 2b(u_m, u_m^h) + b(u_m^h, u_m^h), \\ &= \lambda_m c(u_m, u_m) - 2\lambda_m c(u_m, u_m^h) + \lambda_m^h c(u_m^h, u_m^h), \\ &= \lambda_m - 2\lambda_m c(u_m, u_m^h) + \lambda_m^h, \\ &= 2\lambda_m - 2\lambda_m c(u_m, u_m^h) + \lambda_m^h - \lambda_m, \\ &= \lambda_m c(u_m - u_m^h, u_m - u_m^h) + \lambda_m^h - \lambda_m. \end{aligned}$$

□

It has been shown that the eigenvalues converge to the exact eigenvalues as $h \to 0$. So from this result, it only remains to show that $c(u_m - u_m^h, u_m - u_m^h) \to 0$ as $h \to 0$.

At this point, another assumption must be made. Assume that there are not eigenvalues with multiplicity more than 1. In other words, all the eigenvalues correspond only to one eigenfunction. In [SF73], the authors mention that for repeated eigenvalues, then the eigenfunctions can be chosen so that the main convergence results hold. This case is ommited in this dissertation.

Lemma 6. For all m and j

$$(\lambda_j^h - \lambda_m) c(Pu_m, u_j^h) = \lambda_m c(u_m - Pu_m, u_j^h).$$

Proof. Since the term $\lambda_m c(Pu, u_j^h)$ appears on both sides of the equation, it is only required to show that

$$\lambda_j^h c(Pu, u_j^h) \quad = \quad \lambda_m c(u, u_j^h).$$

Since both u and u_j^h are eigenfunctions, then

$$\begin{aligned} \lambda_j^h c(Pu, u_j^h) &= b(Pu, u_j^h), \\ \lambda_m c(u, u_j^h) &= b(u, u_j^h). \end{aligned}$$

Then equality follows from the definitions of the projection P. □

The set $\left\{u_1^h, u_2^h, ..., u_N^h\right\}$ forms an orthonormal basis for S^h. The projection Pu_m can be written as:

$$Pu_m \quad = \quad \sum_{j=1}^{N} c(Pu_m, u_j^h) u_j^h. \tag{3.4.1}$$

From Lemma 6, it follows that $c(P_m, u_j^h)$ is small if λ_m^h is not close to λ_j. Therefore (3.4.1) tells us that Pu_m is close to u_m^h. The estimate for $Pu_m - u_m^h$ will follow from this result.

Following the convergence of the eigenvalues, $\exists \rho > 0$ and $\exists \delta > 0$ such that if $h < \delta$,

$$|\lambda_m - \lambda_j^h| \quad > \quad \rho \quad \text{for all} \quad j = 1, 2, ..., N. \tag{3.4.2}$$

Therefore

$$\frac{\lambda_m}{|\lambda_m - \lambda_j^h|} \quad \leq \quad \rho \quad \text{for all} \quad j = 1, 2, ..., N. \tag{3.4.3}$$

To simplify the notation, let $\beta = c(Pu_m, Pu_m^h)$.
Lemma 7.

$$||Pu - \beta Pu_m^h||_W^2 \quad \leq \quad \rho^2 ||u_m - Pu_m||_W^2.$$

Lemma 8.

$$||u_m - \beta u_m^h||_W \quad \leq \quad (1 + \rho)\, ||u_m - Pu_m||_W.$$

The proofs for lemma's 7 and 8 are given in [SF73].

So again using the Approximation Theorem, it follows that $||u_m - \beta u_m^h||_W \leq Ch^k ||u^k||_W$.
Lemma 9.

$$||u_m - u_m^h||_W \quad \leq \quad 2||u_m - \beta u_m^h||_W.$$

Proof.

$$\begin{aligned} ||u_m - u_m^h||_W &= ||u_m - \beta u_m^h + \beta u_m^h - u_m^h||_W, \\ &\leq ||u_m - \beta u_m^h||_W + ||\beta u_m^h - u_m^h||_W, \\ &= 2||u_m - \beta u_m^h||_W. \end{aligned}$$

□

Therefore $||u_m - u_m^h||_W \leq Ch^k ||u^k||_W$. So for any $\epsilon > 0$, a $\delta > 0$ can be found such that if $h < \delta$, $||u_m - u_m^h||_W < \epsilon$.

3.5 The approximation theorem

Consider a interpolation operator Π. This projection is linear, i.e.

$$\begin{aligned} \Pi(f + g) &= \Pi f + \Pi g, \\ \Pi(\alpha f) &= \alpha \Pi f \quad \text{for a constant } \alpha. \end{aligned}$$

Define the interval $I_e = [a, a+h]$. A necessary condition is for the operator Π is that when Πu is restricted to the interval I_e, this must equal the projection of u restricted to the interval I_e. This can be written as

$$[\Pi u]_{I_e} = \Pi_e [u]_{I_e}.$$

The following notation is introduced.

> $\mathcal{P}_j(I_e)$: Is the set of all polynomials on the interval I_e of degree at most j.
>
> $r(\Pi_e)$: If the range of Π_e is contained in $\mathcal{P}_j(I_e)$ and $k < j$ is the largest integer such that $\Pi_e f = f$ for each $f \in \mathcal{P}_j(I_e)$, then $r(\Pi_e) = k$.
>
> $s(\Pi_e)$: Is a integer and the largest order derivative used in the definition of Π_e.

From the textbook [OR76], the following approximation theorem for finite elements is given verbatim:
Theorem (The Interpolation Theorem for Finite Elements). Let Ω be an open bounded domain in $\mathcal{R}^n$ satisfying the cone condition. Let k be a fixed integer and m an integer such that $0 \leq m \leq k+1$. Let $\Pi \in L(H^{k+1}(\Omega), H^m(\Omega))$ be such that

$$\Pi u = u \qquad \text{for all } u \in \mathcal{P}_k(\Omega) \tag{3.5.1}$$

Then for any $u \in H^{k+1}(\Omega)$ and for sufficiently small h, there exists positive a constant C, independent of u and h, such that

$$||u - \Pi u||_{H^m(\Omega)} \leq C \frac{h^{k+1}}{p^m} |u|_H^{k+1}(\Omega) \tag{3.5.2}$$

where $|u|_H^{k+1}(\Omega)$ is the seminorm.

For the requirements of this dissertation, this can be simplified with an assumption. The assumption is that the basis of S^h consists of polynomials. With these assumptions, the semi-norm $|u|_H$ is equal to the norm $||u||_W$. This approximation theorem can be rewritten as
Theorem 1. Suppose there exists an integer k such that for each element

$$s(\Pi_e) + 1 \leq k \leq r(\Pi_e) + 1.$$

Then there exists a constant C such that for any $u \in C_+^k(I)$,

$$||(\Pi u)^{(m)} - u^{(m)}||_W \leq C h^{k-m} ||u^{(k)}||_W \quad \text{for } m = 0, 1, ..., k.$$

4 Timoshenko beam model

4.1 Introduction

An example of modal analysis of a Timoshenko beam is given in Section 2.4. In this chapter, more general theory for modal analysis of the Timoshenko beam theory is discussed, as well as more examples of the application of the theory.

4.2 Eigenvalue problem

This section is a discussion of a systematic method to solve the eigenvalue problem for the Timoshenko beam theory. The article discussed is [VV06].

Consider a general eigenvalue problem for a Timoshenko beam Problem.

General eigenvalue problem

Find functions u and ϕ and a real number λ satisfying the following equations

$$-u'' + \phi' = \lambda u, \tag{4.2.1}$$

$$-\alpha u' + \alpha\phi - \frac{1}{\gamma}\phi'' = \lambda\phi. \tag{4.2.2}$$

Recall from Section 1.3, u represents the transverse motion of the beam, ϕ is the angle of rotation of the cross-section of the beam, α and γ are dimensionless constants defined in Section 5.4 and λ represents the eigenvalue.

The authors of [VV06] first derive a general solution for the system of ordinary differential equations (4.2.1) and (4.2.2). In [VV06] the authors show that for the Timoshenko models in this dissertation that λ is non-negative. Assume that the solution is of the form $\langle u, \phi \rangle = e^{mx}\bar{w}$ where $m \in R$ or $m \in C$. Substitution into (4.2.1) and (4.2.2) yields:

$$\begin{aligned} -m^2 e^{mx} w_1 + m e^{mx} w_2 &= \lambda e^{mx} w_1, \\ -\frac{1}{\gamma} m^2 e^{mx} w_2 - \alpha m e^{mx} w_1 + \alpha e^{mx} w_2 &= \lambda e^{mx} w_2. \end{aligned}$$

From these equations it can be concluded that $e^{mx}\bar{w}$ is a solution of the system if and only if the pair $\langle m, \bar{w} \rangle$ is a solution of the linear system:

$$\begin{bmatrix} -m^2 - \lambda & m \\ -\alpha m & -\frac{1}{\gamma} m^2 + (\alpha - \lambda) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{4.2.3}$$

Let A denote the coefficient matrix of (4.2.3). For a nontrivial solution $\bar{w}$, it is required that the determinant of matrix A is 0.

$$\det(A) = m^4 + \lambda(1+\gamma)m^2 + \gamma\lambda(\lambda - \alpha) = 0.$$

This equation is called the characteristic equation. The characteristic equation is quadratic with respect to m^2. The roots of the characteristic equation can be expressed as

$$m^2 = -\frac{1}{2}\lambda(1+\gamma)(1 \pm \sqrt{\Delta}), \tag{4.2.4}$$

where

$$\Delta = 1 - \frac{4\gamma}{(1+\gamma)^2}\left(1 - \frac{\alpha}{\lambda}\right) = \frac{4\gamma}{(1+\gamma)^2}\frac{\alpha}{\lambda} + \frac{(1-\gamma)^2}{(1+\gamma)^2}. \tag{4.2.5}$$

It is clear that $\Delta > 0$, since λ, α and γ are all positive. Therefore m^2 will always be real, and have distinct roots.

Consider the case where $m^2 = 0$, which occurs when $\lambda = \alpha$. Then the matrix A simplifies to

$$\begin{bmatrix} \alpha & 0 \\ 0 & 0 \end{bmatrix}.$$

In this case it is easy to find two linearly independent solutions:

$$[u(x)\ \phi(x)]^T = [0\ 1]^T \quad \text{and} \quad [u(x)\ \phi(x)]^T = [1\ \alpha x]^T.$$

Now if $m^2 \neq 0$, we consider the system $A\bar{w} = 0$ with $w_1 = m$ and $w_2 = m^2 + \lambda$. From [VV06], the parameters ω, μ and θ are uniquely determined by λ.

$$\omega^2 = \frac{1}{2}\lambda(1+\gamma)(\Delta^{\frac{1}{2}}+1) \quad \text{for } \lambda > 0 \tag{4.2.6}$$
$$\mu^2 = \frac{1}{2}\lambda(1+\gamma)(\Delta^{\frac{1}{2}}-1) \quad \text{for } \lambda < \alpha \tag{4.2.7}$$
$$\theta^2 = \frac{1}{2}\lambda(1+\gamma)(1-\Delta^{\frac{1}{2}}) \quad \text{for } \lambda > \alpha \tag{4.2.8}$$

The three cases $\lambda < \alpha$, $\lambda = \alpha$ and $\lambda > \alpha$ are considered separately by the authors of [VV06]. The general solution from [VV06] for each case is presented below.

Case $\boldsymbol{\lambda < \alpha}$

Denote the roots of (4.2.4) by $\pm\mu$ and $\pm\omega i$. Thus the general solution is given by

$$\begin{aligned}\begin{bmatrix} u(x) \\ \phi(x) \end{bmatrix} &= A\begin{bmatrix} \sinh(\mu x) \\ \frac{\lambda+\mu^2}{\mu}\cosh(\mu x) \end{bmatrix} + B\begin{bmatrix} \cosh(\mu x) \\ \frac{\lambda+\mu^2}{\mu}\sinh(\mu x) \end{bmatrix} + C\begin{bmatrix} \sin(\omega x) \\ -\frac{\lambda-\omega^2}{\omega}\cos(\omega x) \end{bmatrix} \\ &\quad + D\begin{bmatrix} \cos(\omega x) \\ \frac{\lambda-\omega^2}{\omega}\sin(\omega x) \end{bmatrix}.\end{aligned}$$

Case $\boldsymbol{\lambda = \alpha}$

In this case the roots of (4.2.4) are 0 with multiplicity 2, and $\pm\omega i$. The general solution is

$$\begin{bmatrix} u(x) \\ \phi(x) \end{bmatrix} = A\begin{bmatrix} 0 \\ 1 \end{bmatrix} + B\begin{bmatrix} 1 \\ \alpha x \end{bmatrix} + C\begin{bmatrix} \sin(\omega x) \\ -\frac{\lambda-\omega^2}{\omega}\cos(\omega x) \end{bmatrix} + D\begin{bmatrix} \cos(\omega x) \\ \frac{\lambda-\omega^2}{\omega}\sin(\omega x) \end{bmatrix}.$$

Case $\boldsymbol{\lambda > \alpha}$

All the roots of (4.2.4) are complex. Denote them by $\pm\theta i$ and $\pm\omega i$. The general solution is

$$\begin{aligned}\begin{bmatrix} u(x) \\ \phi(x) \end{bmatrix} &= A\begin{bmatrix} \sin(\theta x) \\ -\frac{\lambda-\theta^2}{\theta}\cos(\theta x) \end{bmatrix} + B\begin{bmatrix} \cos(\theta x) \\ \frac{\lambda-\theta^2}{\theta}\sin(\theta x) \end{bmatrix} + C\begin{bmatrix} \sin(\omega x) \\ -\frac{\lambda-\omega^2}{\omega}\cos(\omega x) \end{bmatrix} \\ &\quad + D\begin{bmatrix} \cos(\omega x) \\ \frac{\lambda-\omega^2}{\omega}\sin(\omega x) \end{bmatrix}.\end{aligned}$$

Strategy for determining the eigenvalues and eigenvectors

The authors of [VV06] provide a detailed example of the application of the above strategy for determining the eigenvalues and eigenvectors. The example used is a pinned-pinned beam, the same model as in Section 2.4.1.

The strategy can be summarized as follows:

From the general solutions for u and ϕ, the eigenvalues and eigenfunctions can be determined by imposing the boundary conditions at $x = 0$ to reduce the solution space of four dimensions to a solution space of dimension at least two.

The boundary condition at $x = 1$ is then substituted which results in a homogeneous system of linear equations of the form

$$A\bar{b} = \bar{0}.$$

This system either has the zero solution or infinitely many solutions. To ensure the zero solution, the determinant of the matrix A is set to zero. This equation $\det(A) = 0$ is called the frequency equation.

The frequency equation has infinitely many solutions. Each of the solutions corresponds to a eigenvalue with a unique vector $\bar{b}$. The eigenfunctions can then be obtained by substituting the vector $\bar{b}$ into the general solution for u and ϕ.

Pinned-pinned beam

Returning back to the example of the pinned-pinned beam, we present some of the results from [VV06]:

For $\lambda < \alpha$ the general solution reduces to $\langle u(x), \phi(x)\rangle = \langle \sin(k\pi x), A_k \cos(k\pi x)\rangle$.

For $\lambda = \alpha$, the general solution reduces to $\langle u(x), \phi(x)\rangle = \langle 0, 1\rangle$.

For $\lambda > \alpha$, the general solution also reduces to $\langle u(x), \phi(x)\rangle = \langle \sin(k\pi x), A_k \cos(k\pi x)\rangle$.

A rigourous proof of the above results can be found in [VV06]. These results also prove the Theorem 1 presented in Section 2.4.1.

4.3 Cantilever beam

Consider a cantilever Timoshenko beam model (Problem T-3 as defined in Section 1.3.3). The eigenvalue problem of Problem T-3 is denoted by Problem

T-3E. This section serves as an example of the application of the theory from [VV06].

Consider the general solutions of the eigenvalue problem for the three cases $\lambda < \alpha$, $\lambda = \alpha$ and $\lambda > \alpha$. Imposing the boundary conditions $x = 0$ results in the following constants:

$$C = \frac{\mu(\lambda - \omega^2)}{\omega(\lambda + \mu^2)} A \qquad \text{and} \quad D = -B \quad \text{if } \lambda < \alpha \tag{4.3.1}$$

$$C = \frac{\omega}{\lambda - \omega^2} A \qquad \text{and} \quad D = -B \quad \text{if } \lambda = \alpha \tag{4.3.2}$$

$$C = -\frac{\omega(\lambda - \theta^2)}{\theta(\lambda - \omega^2)} A \qquad \text{and} \quad D = -B \quad \text{if } \lambda > \alpha \tag{4.3.3}$$

The boundary conditions at $x = 1$ reduces the general solution to the following two-dimensional homogeneous system for each of the three cases.

$$\begin{bmatrix} M_{11}(\lambda) & M_{12}(\lambda) \\ M_{21}(\lambda) & M_{22}(\lambda) \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.3.4}$$

To obtain non-zero solutions, the determinant of the coefficient matrix M is set to zero, ie. $\det(M) = 0$, and is called the frequency equation after simplification. The frequency equations for all three cases is given in [VV06] and is presented below.

Case $\lambda < \alpha$:

$$\left(\frac{\lambda + \mu^2}{\lambda - \omega^2} + \frac{\lambda - \omega^2}{\lambda + \mu^2} \right) \cosh(\mu)\cos(\omega) + \left(\frac{\omega}{\mu} - \frac{\mu}{\omega} \right) \sinh(\mu)\sin(\omega) - 2 = 0 \tag{4.3.5}$$

Case $\lambda = \alpha$:

$$\left(\frac{\alpha}{\alpha - \omega^2} + \frac{\alpha - \omega^2}{\alpha} \right) \cos(\omega) + \omega \sin(\omega) - 2 = 0 \tag{4.3.6}$$

Case $\lambda > \alpha$:

$$\left(\frac{\lambda+\theta^2}{\lambda-\omega^2}+\frac{\lambda-\omega^2}{\lambda+\theta^2}\right)\cos(\theta)\cos(\omega)+\left(\frac{\omega}{\theta}-\frac{\theta}{\omega}\right)\sin(\theta)\sin(\omega)-2=0 \quad (4.3.7)$$

Inspection of the system of equations (4.3.4), shows that the coefficient matrix is never zero for the case of the cantilever beam. It follows that all the eigenvalues are simple eigenvalues. This is mentioned in [VV06].

4.3.1 Calculating the eigenvalues

The solutions of the frequency equations (4.3.5) - (4.3.6) can be calculated using simple numerical methods. Using interval division, the eigenvalues are calculated accurate to at least 4 significant digits.

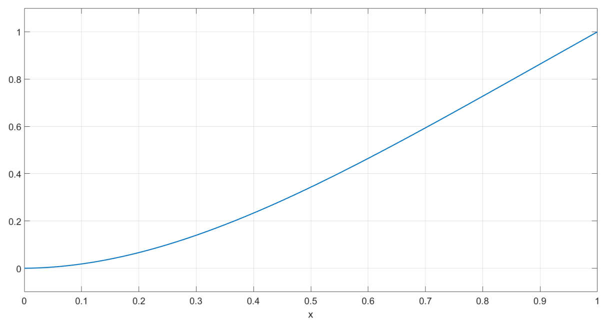
Cantilever Beam Eigenvalues			
i	$\alpha = 1200$	$\alpha = 4800$	$\alpha = 10800$
1	0.04043	0.01025	0.004569
2	1.427	0.3914	0.1772
3	9.657	2.937	1.361
4	31.06	10.62	5.08
5	70.51	27.02	13.4
6	130.8	55.63	28.67
7	213.4	99.54	53.36
8	318.7	161.2	89.85

Table 4.1: First 8 eigenvalues, with $\gamma = 0.25$.

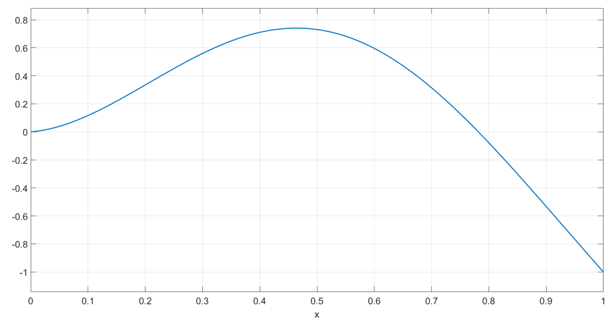
Remark: For interval division, the frequency equations are sketched to determine intervals to isolate the solutions. Another method that requires less user involvement is the bisection method. The advantage of the interval method is that the number of solutions within any given interval can be determined visually before the method is applied.

4.3.2 Example of mode shapes

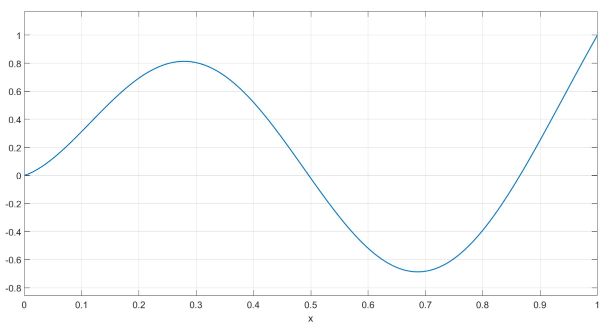
Substituting a calculated eigenvalue λ_k back into (4.3.4), the values for A_k and B_k can be calculated by solving the system. But since $\det(M) = 0$, there are infinity many solutions of $[A_k\ B_k]^T$ for each k. So either one of A_k or B_k can be chosen freely and the other is depended on the choice. The modal shapes are sketched below in Figure 4.1 (modal shapes for u) and Figure 4.2 (modal shapes for ϕ) corresponding to the eigenvalue λ_k with $A_k = 1$, $\alpha = 1200$ and $\gamma = 0.25$.



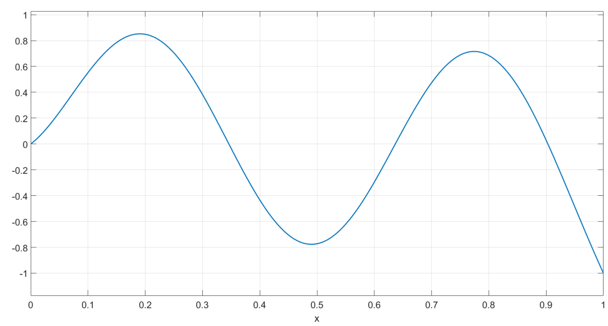
(a) $\lambda_1 = 0.04043$, $B_1 = -1.368$



(b) $\lambda_2 = 1.427$, $B_2 = -0.9768$

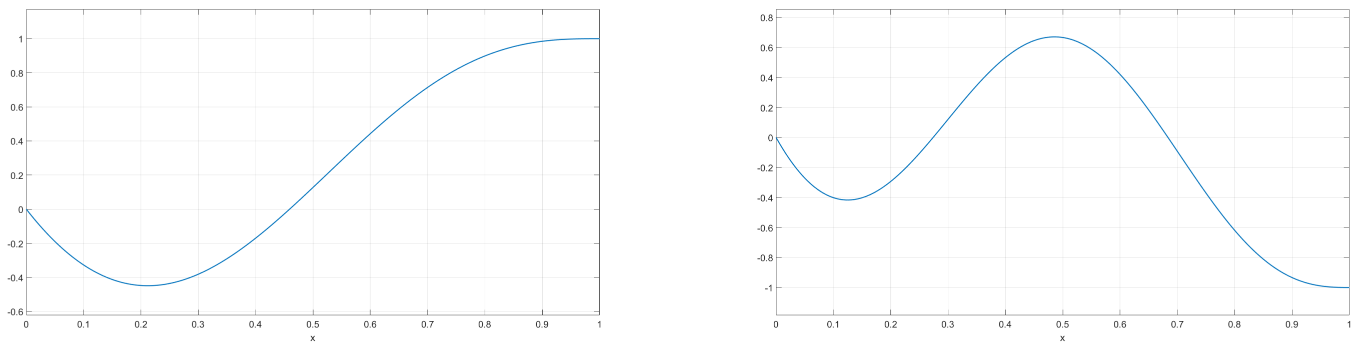


(c) $\lambda_3 = 9.657$, $B_3 = -1.002$



(d) $\lambda_4 = 31.06$, $B_4 = -0.9997$

Figure 4.1: Sketch of w for the first 4 mode shapes of the cantilever beam. $\alpha = 1200$ and $\gamma = 0.25$ with $A_k = 1$.



(a) $\lambda_2 = 1.427$, $B_1 = -0.9768$

(b) $\lambda_3 = 9.657$, $B_2 = -1.002$

Figure 4.2: Sketch of ϕ for mode shapes 2 and 3 of the cantilever beam. $\alpha = 1200$ and $\gamma = 0.25$ with $A_k = 1$.

Remark: The sketch of ϕ for the first mode shape is similar to the sketch of w and is therefore omitted.

The results obtained for the Cantilever beam model are the same as the results in [VV06]. The motivation for this model is Chapter 6. The work of this section will be used for comparisons to a two and three-dimensional cantilever beam model.

4.4 Free-free Timoshenko beam

Consider a free-free Timoshenko beam model (Problem T-4 in Section 1.3.3).The eigenvalue problem of Problem T-4 is denoted by Problem T-4E. Similar to the previous section, this section serves as an example of the application of the theory from [VV02].

Consider the general solutions of the eigenvalue problem for the three cases $\lambda < \alpha$, $\lambda = \alpha$ and $\lambda > \alpha$. Imposing the boundary conditions $x = 0$ results in the following:

$$C = \frac{\omega}{\mu} A \qquad \text{and} \quad D = \frac{\mu^2 + \lambda}{\omega^2 - \lambda} B \quad \text{if } \lambda < \alpha \tag{4.4.1}$$
$$C = \frac{\omega}{\lambda} A \qquad \text{and} \quad D = \frac{\alpha}{\omega^2 - \lambda} B \quad \text{if } \lambda = \alpha \tag{4.4.2}$$
$$C = -\frac{\omega}{\theta} A \qquad \text{and} \quad D = -\frac{\theta^2 - \lambda}{\omega^2 - \lambda} B \quad \text{if } \lambda > \alpha \tag{4.4.3}$$

The boundary conditions at $x = 1$ gives the following homogeneous system of equations for each of the three cases.

$$\begin{bmatrix} M_{11}(\lambda) & M_{12}(\lambda) \\ M_{21}(\lambda) & M_{22}(\lambda) \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.4.4}$$

The entries of the coefficient matrix for each separate case is presented below.

If $\lambda < \alpha$

$$\begin{aligned}
M_{11}(\lambda) &= \sinh(\mu)(\lambda + \mu^2) + \frac{\omega(\lambda - \omega^2)}{\mu} \sin(\omega) \\
M_{12}(\lambda) &= (\lambda + \mu^2)(\cosh(\mu) - \cos(\omega)) \\
M_{21}(\lambda) &= \frac{\lambda}{\mu}(\cos(\omega) - \cosh(\mu)) \\
M_{22}(\lambda) &= \frac{\lambda^2 + \lambda\mu^2}{\omega(\lambda - \omega^2)} \sin(\omega) - \frac{\lambda}{\mu} \sinh(\mu)
\end{aligned}$$

If $\lambda = \alpha$

$$\begin{aligned}
M_{11}(\lambda) &= \frac{\omega\left(\lambda - \omega^2\right)}{\lambda} \sin(\omega) \\
M_{12}(\lambda) &= \alpha - \alpha\cos(\omega) \\
M_{21}(\lambda) &= \frac{\omega^2}{\lambda} \cos(\omega) + \cos(\omega) \frac{(\lambda - \omega^2)}{\lambda} - 1 \\
M_{22}(\lambda) &= -\alpha + \left(\frac{\alpha}{\omega} + \frac{\alpha\omega}{\lambda - \omega^2}\right) \sin(\omega)
\end{aligned}$$

If $\lambda > \alpha$

$$\begin{aligned}
M_{11}(\lambda) &= \left(\lambda - \theta^2\right)\sin(\theta) - \frac{\omega\left(\lambda - \omega^2\right)}{\theta}\sin(\omega) \\
M_{12}(\lambda) &= \left(\lambda - \omega^2\right)(\cos(\omega) - \cos(\theta)) \\
M_{21}(\lambda) &= -\frac{\lambda}{\theta}(\cos(\omega) - \cos(\theta)) \\
M_{22}(\lambda) &= \frac{\lambda^2 - \lambda\theta^2}{\omega(\lambda - \omega^2)}\sin(\omega) - \frac{\lambda}{\theta}\sin(\theta)
\end{aligned}$$

The frequency equations can be determined by simplifying the equation

$$M_{11}(\lambda)M_{22}(\lambda) - M_{12}(\lambda)M_{21}(\lambda) = 0$$

for each case. To retain readability, the frequency equations are not written out.

4.4.1 Calculating the eigenvalues

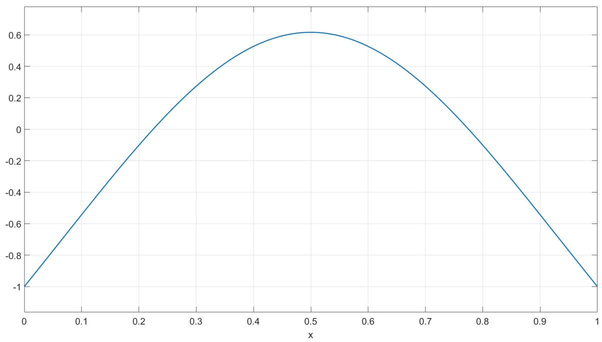
Using interval division, the eigenvalues are calculated accurate to at least 4 significant digits.

Free-Free Beam Eigenvalues			
i	$\alpha = 1200$	$\alpha = 4800$	$\alpha = 10800$
1	1.544	0.4088	0.1837
2	10.26	2.989	1.372
3	33.41	10.88	5.139
4	76.16	27.82	13.59
5	141.3	57.49	29.15
6	229.7	103.2	54.38
7	341.2	167.4	91.75
8	475.0	252.2	143.5

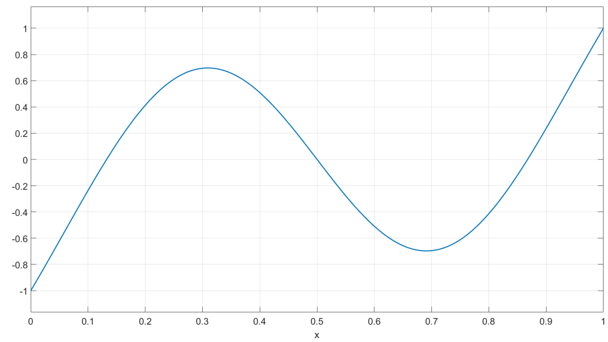
Table 4.2: First 8 eigenvalues, with $\gamma = 0.25$.

4.4.2 Example of mode shapes

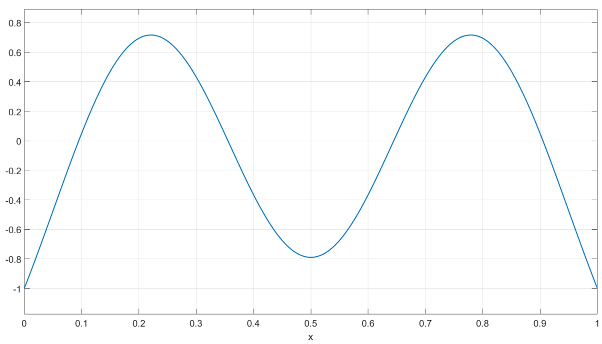
Figure 4.3 (modal shapes for u) and Figure 4.4 (modal shapes for ϕ) show examples of the modal shapes corresponding to the eigenvalue λ_k with $A_k = 1$, $\alpha = 1200$ and $\gamma = 0.25$.



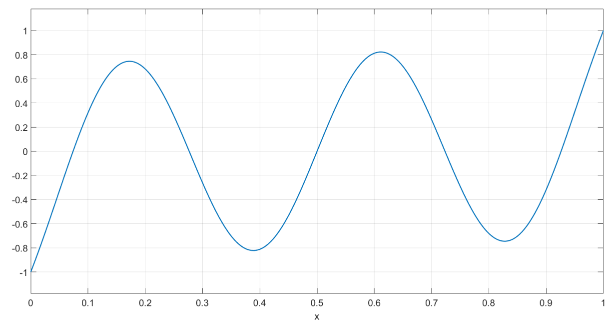
(a) $\lambda_1 = 1.544$, $B_1 = -1.022$



(b) $\lambda_2 = 10.26$, $B_2 = -0.9982$

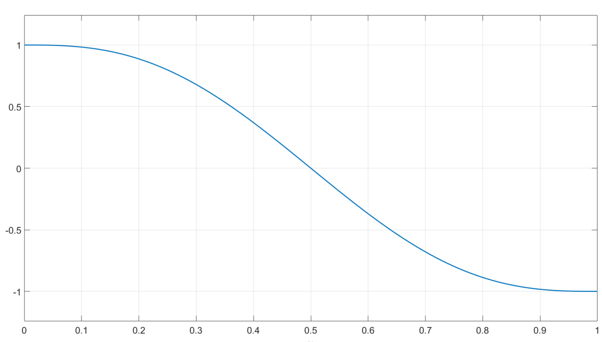


(c) $\lambda_3 = 33.41$, $B_3 = -1.000$

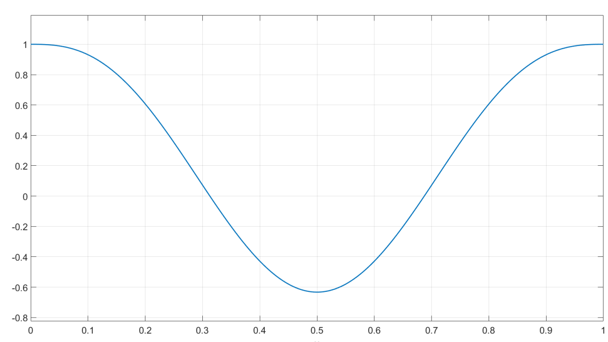


(d) $\lambda_4 = 76.16$, $B_4 = -0.9999$

Figure 4.3: Sketch of w of the first 4 mode shapes of the free-free beam. $\alpha = 1200$ and $\gamma = 0.25$ with $A_k = 1$.



(a) $\lambda_2 = 1.544$, $B_1 = -1.022$



(b) $\lambda_3 = 10.26$, $B_2 = -0.9982$

Figure 4.4: Sketch of ϕ for the first two mode shapes of the free-free beam. $\alpha = 1200$ and $\gamma = 0.25$ with $A_k = 1$.

The motivation for the free-free beam in this dissertation is Section 4.6. In this section, an article is discussed in which the eigenvalues of a free-free Timoshenko beam are compared to empirical results from an experiment conducted in [SP06].

4.5 Validity of the model for a cantilever Timoshenko beam

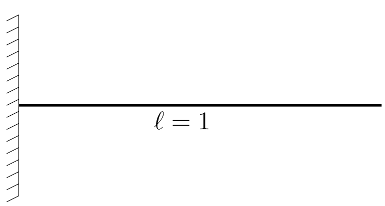
In this section, the validity of the Timoshenko beam model is investigated. Specifically, the validity of a cantilever Timoshenko beam model, using a cantilever two-dimensional beam model as a reference. This follows what is done in the article [LVV09a]. This section discusses the results of the article, focussing on the numerical results. Some of the results are replicated with more significant digits, others are extended to be used in the rest of this chapter.

The article [LVV09a] is titled 'Comparison of linear beam theories'. In this article, the authors compare a cantilever Timoshenko beam to a cantilever two-dimensional beam. The authors start by presenting the models. These are the same as Problem T-2 and Problem 2D-1 that are defined in Chapter 1. The authors also look at the existence and uniqueness of solutions, which is covered in Chapter 2 of this dissertation. The authors then calculate and compare the eigenvalues and eigenfunctions, which will be discussed in this section.

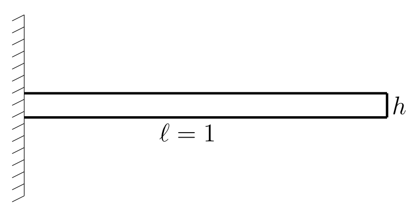
4.5.1 The models

The cantilever Timoshenko beam model is defined in Section 1.3.3, and is referred to as Problem T-2. The cantilever two-dimensional beam model is defined in Section 1.2.3, and is referred to as Problem 2D-1.

Figure 4.5.1 shows the two beams side-by-side.



(a) Timoshenko Cantilever Beam



(b) Two-Dimensional Cantilever Beam

Figure 4.5: Side by side comparison of the beams.

In the derivation of the Timoshenko beam model in Section 1.3.1, the parameter α is introduced. The paramater is given here again for convenience:

$$\alpha = \frac{A\ell^2}{I}.$$

The model is in a dimensionless form, therefore the length of the beam is $\ell = 1$. Since we assumed a square cross-section, the area of the cross-section can be calculated as $A = hb$. The area moment of inertia can also be calculated as $I = \dfrac{h^3 b}{12}$. Substituting these values into the formula for α gives the following relationship between the height of the beam and the parameter α: $\alpha = \dfrac{12}{h^2}$ or equivalently,

$$h = \sqrt{\frac{12}{\alpha}}. \tag{4.5.1}$$

Using this relationship, the height of the beam model can be set by considering the value of α.

4.5.2 Calculating the eigenvalues and eigenvectors

In Section 4.2 a method for calculating the eigenvalues and eigenvectors of the Timoshenko beam is provided. And in Section 4.3, the eigenvalues and eigenvectors of a cantilever Timoshenko beam are calculated as an example.

For the two-dimensional beam, the eigenvalues and eigenvectors are calculated using the Finite Element Method. In Section 5.2, the Finite Element Method for a cantilever two-dimensional beam is derived. Specifically, Section 5.2.4 derives the eigenvalue problem for the two-dimensional beam using the Finite Element Method, called Problem 2D-1E.

Problem 2D-1E

Find a real number λ and a vector $\bar{u} \in R^n$ such that

$$K\bar{u} = M\lambda\bar{u}, \tag{4.5.2}$$

where M and K are the standard Finite Element Method matrices defined in Section 5.2.

In this form, the eigenvalue problem is a system of ordinary differential equations, and can be solved using a numerical method. Computer programs like MATLAB provide functions that are able to solve this.

Accuracy of the eigenvalues

Solving **Problem 2D-1E** with a numerical method, requires a quick investigation into the rate of convergence so that the accuracy of the eigenvalues can be established. Chapter 3 provides the necessary theory on the existence of solutions as well as the proof of convergence of the Finite Element Method.

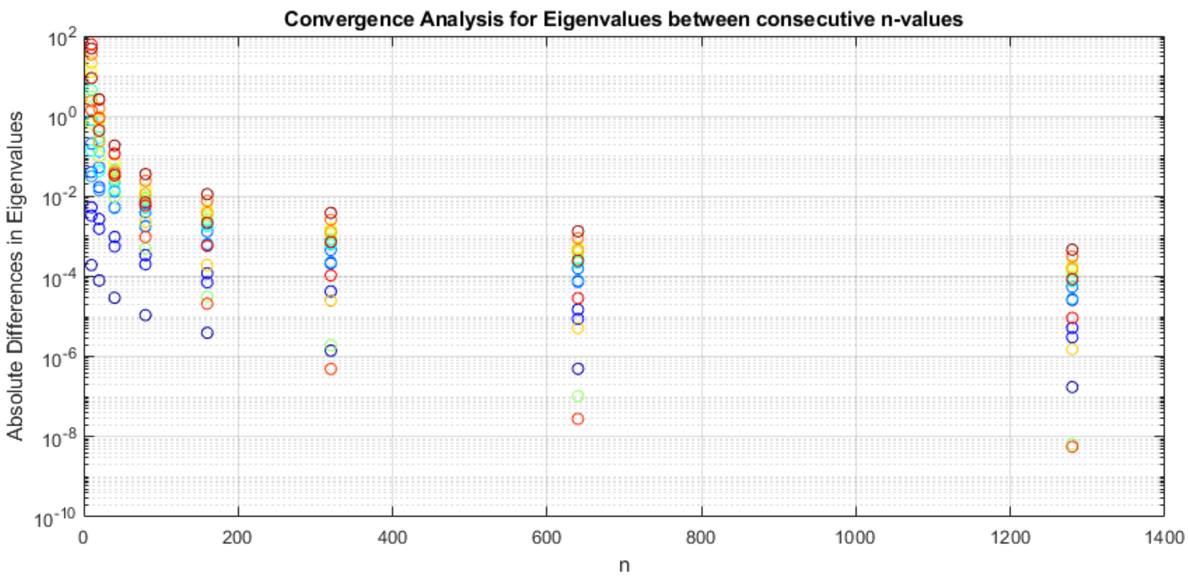


Figure 4.6: Rate of convergence of the first 20 eigenvalues.

Figure 4.6 shows the rate of convergence of the first 20 eigenvalues of the two-dimensional beam. Each color represents a spesific eigenvalue.

The number of elements are chosen so that at least the first 10 eigenvalues are accurate to 5 significant digits.

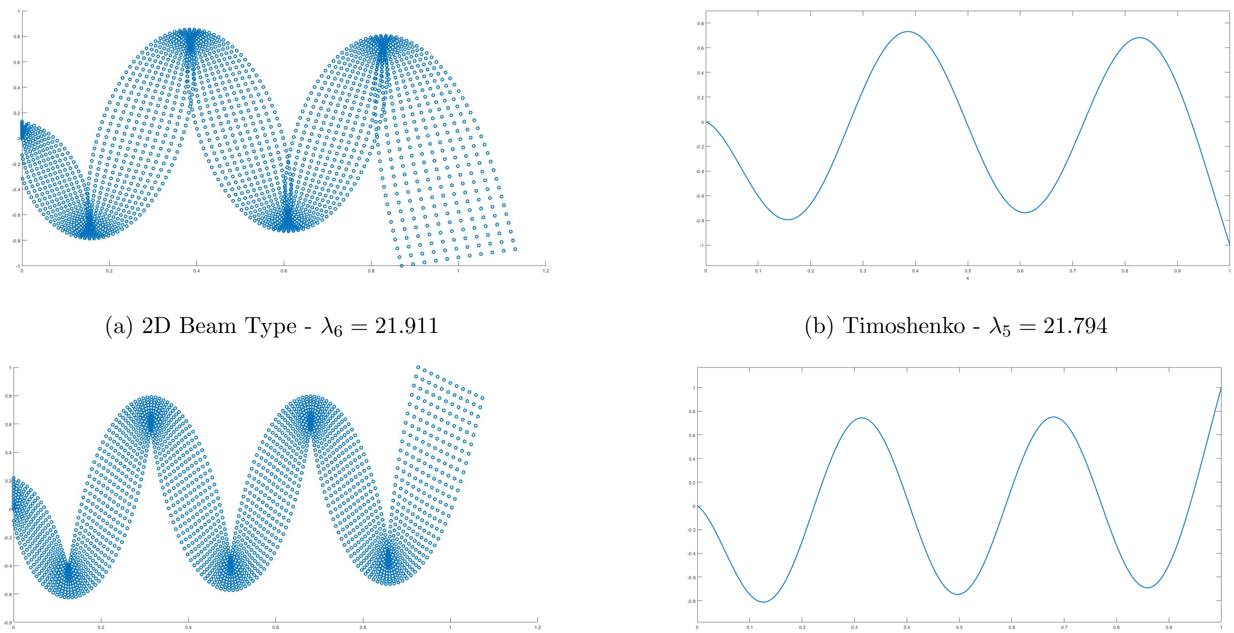
4.5.3 Comparing the mode shapes

Recall from the introduction, that the investigation is focussed on beam-type problems. The two-dimensional model is for an eigenvalue problem and and not specific to beam-type applications, like the Timoshenko beam model. Therefore there are non-beam type modes that can be expected from the two-dimensional model. This is also be reflected in the eigenvalues of the two-dimensional model and therefore eigenvalues irrelevant to beam-type problems exits.

Therefore a method is required to compare and match the eigenvalues of the two models. In [LVV09a], the authors compare the mode shapes of the two models, to match up the eigenvalues. Eigenvalues relating to the mode shapes similar to the mode shapes of the Timoshenko beam, are called beam-type eigenvalues. The other eigenvalues are called non-beam type eigenvalues by the authors.

Shapes relating to beam-type eigenvalues

The following figures are examples of beam-type mode shapes for the displacement w.



(a) 2D Beam Type - $\lambda_6 = 21.911$

(b) Timoshenko - $\lambda_5 = 21.794$

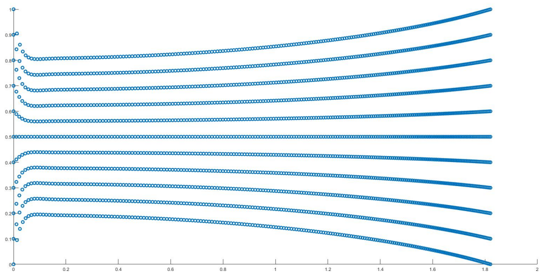
(c) 2D Beam Type - $\lambda_7 = 45.711$

(d) Timoshenko - $\lambda_6 = 45.390$

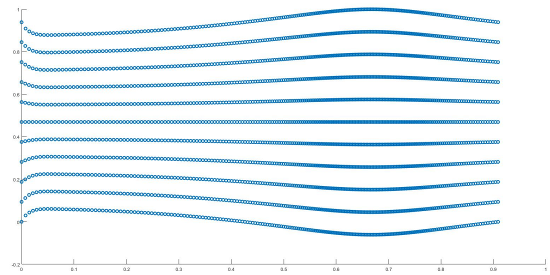
Figure 4.7: Modal shapes of the displacement w for the beam-type 2D body and the Timoshenko beam with $\alpha = 4800$ ($h = 1/20$).

Shapes relating to non-beam type eigenvalues

The following figures are examples of non beam-type mode shapes for the displacement w. These mode shapes are not present in the cantilever Timoshenko beam model and are not beam related.



(a) Non-Beam Type - $\lambda_4 = 7.7077$



(b) Non-Beam Type - $\lambda_8 = 69.344$

Figure 4.8: Modal shapes of the displacement w for the non-beam type 2D body with $h = 1/20$.

Shape of the cross-sections

The Timoshenko beam theory improve some one-dimensional beam theories such as the Euler-Bernoulli beam theory by also including the effect of shear. For Timoshenko beam theory it is assumed that the cross-sections need not remain perpendicular to the neutral axis of the beam. The cross-sections however remain a straight line.

For the two-dimensional beam, the shape of the cross-section can deform into a S-shape. This is explained in [LVV09a] and a similar figure in [LVV09a] is given here.

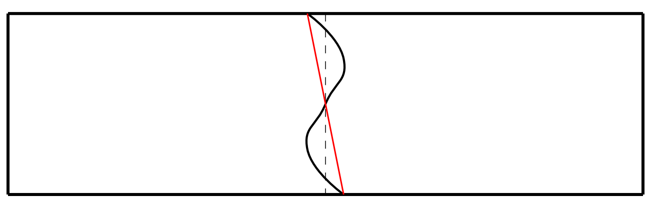


Figure 4.9: S-shape deformation of the cross-section of the two-dimensional beam.

Note that figure 4.9 shows an exaggerated example of the deformation of a cross-section. The red line shows how the authors of [LVV09a] calculated the average rotation of a cross-section of the two-dimensional beam. This average rotation can be used to compare the rotation of the cross-section of the two-dimensional beam to the rotation of the cross-section of the Timoshenko beam given by ϕ.

Direct comparison of mode shapes

Figure 4.10 directly compares a mode shape of the Timoshenko beam to a mode shape of the two-dimensional beam. For the two-dimensional model, the center line of the displacement of the beam is shown. For the Timoshenko beam, the displacement w is shown.

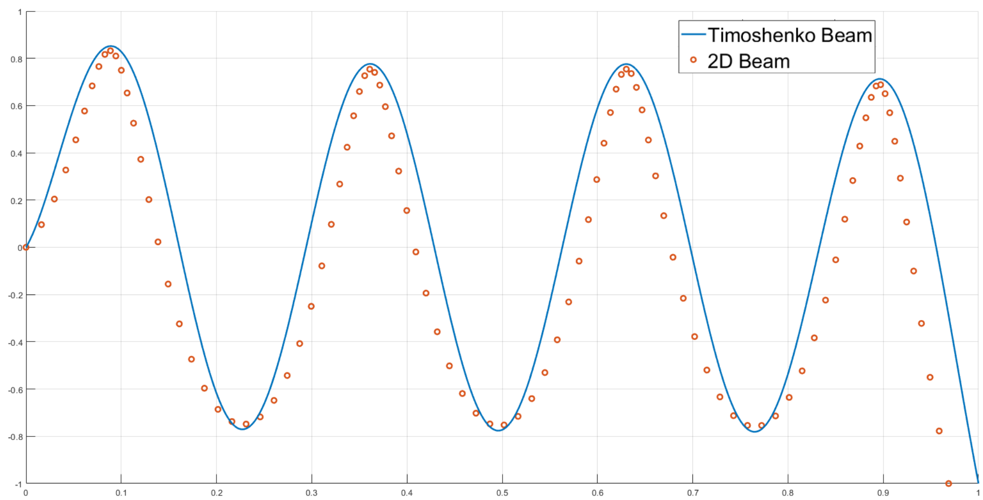


Figure 4.10: Comparison of the displacement w mode shape corresponding to λ_{10} for the 2D beam and λ_8 for the Timoshenko beam with $\alpha = 4800$ ($h = 1/20$)

Similarly, figure 4.11 directly compares the angle of the cross-section of the Timoshenko beam and the two-dimensional beam. The average rotation of the cross-section of the two-dimensional beam's mode shape is calculated as shown in figure 4.9.

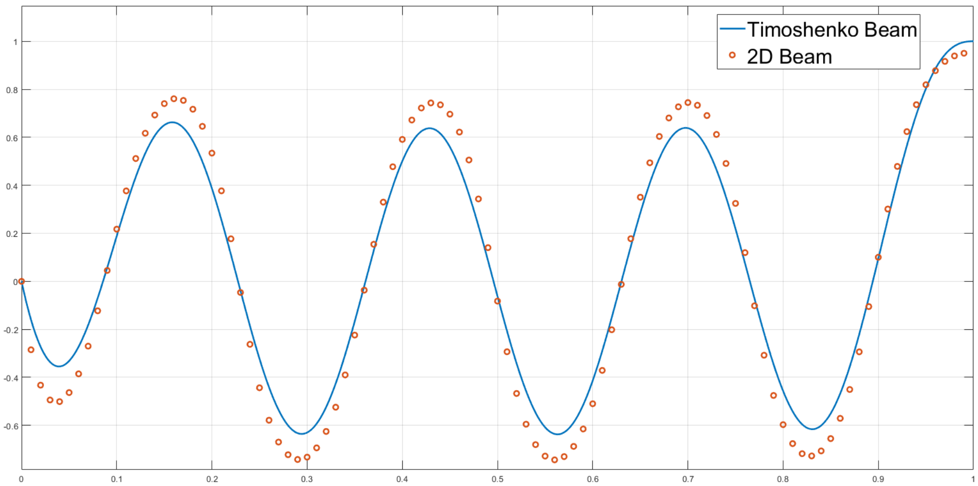


Figure 4.11: Comparison of the angle ϕ (best fit for 2D Beam) mode shape corresponding to λ_{10} for the 2D beam and λ_8 for the Timoshenko beam with $\alpha = 4800$ ($h = 1/20$)

These figures are examples that show how similar the mode shapes of the two models are. This specific example is for $\alpha = 4800$, which represents a typical beam. The authors of [LVV09a] obtained similar results for $\alpha = 300$, which represents a short and thick beam.

Remark: Note that the overall shape is important. This is because any multiple of a eigenvector is still an eigenvector. The mode shapes were specifically scaled to obtain figures 4.10 and 4.11.

4.5.4 Comparing the eigenvalues

Using this method of comparing the mode shapes, the eigenvalues can now be matched up and compared. In the tables, Timo refers to the Timoshenko beam and 2D refers to the two-dimensional beam.

Some results from [LVV09]

The following table contains results from [LVV09a] verbatim to 3 significant digits, as well as the replicated results to 5 significant digits.

This table shows that the results obtained in this dissertation are very similar

Results from [LVV09]			Dissertation		
	2D	Timo		2D	Timo
1	0.0317	0.0316	1	0.031713	0.031639
2	1.14	1.14	2	1.1413	1.1365
3	7.72	-	3	7.7161	-
4	7.92	7.86	4	7.918	7.8617
5	26.2	25.9	5	26.148	25.869
6	60.8	59.9	6	60.816	59.946
7	69.3	-	7	69.344	-
8	115	113	8	115.28	113.23
9	192	188	9	191.57	187.55
10	192	-	10	192.03	-
11	291	284	11	290.76	283.81

Table 4.3: Results from [LVV09] and results obtained in this dissertation. 0^* indicates a 0 as a result of rounding. $\alpha = 1200$.

to the results obtained in [LVV09a]. This is for a specific case where $\alpha = 1200$. In the following table, the eigenvalues for different values of α are compared.

Comparison of Eigenvalues															
$h = 1/5$ or $\alpha = 300$				$h = 1/10$ or $\alpha = 1200$				$h = 1/20$ or $\alpha = 4800$				$h = 1/30$ or $\alpha = 10800$			
i	2D	j	Timo	i	2D	j	Timo	i	2D	j	Timo	i	2D	j	Timo
1	0.12151	1	0.12092	1	0.031713	1	0.031639	1	0.008013	1	0.008004	1	0.003568	1	0.003565
2	3.5460	2	3.5071	2	1.1413	2	1.1365	2	0.30756	2	0.30705	2	0.13869	2	0.13855
3	7.7311		-	3	7.7161		-	3	2.3273	3	2.3213	3	1.0698	3	1.0683
4	20.225	3	19.869	4	7.9180	3	7.8617	4	7.7077		-	4	4.0140	4	4.0058
5	56.109	4	54.766	5	26.148	4	25.869	5	8.5086	4	8.4762	5	7.7047		-
6	69.164		-	6	60.816	5	59.946	6	21.911	5	21.794	6	10.655	5	10.625
7	114.03	5	110.75	7	69.344		-	7	45.711	6	45.390	7	22.975	6	22.890
8	189.17	6	186.50	8	115.28	6	113.23	8	69.344		-	8	43.113	7	42.909
9	192.61			9	191.57	7	187.55	9	82.887	7	82.154	9	69.331		-
10	285.85	7	277.64	10	192.03		-	10	136.03	8	134.58	10	73.230	8	72.803
11	328.40	8	330.29	11	290.76	8	283.81	11	192.48		-	11	115.41	9	114.61
12	357.08		-	12	374.45		-	12	207.29	9	204.69	12	171.61	10	170.20
13	397.33	9	394.02	13	413.20	9	402.27	13	298.38	10	294.10	13	192.52		-
14	442.00	10	439.52	14	558.67	10	542.65	14	376.83		-	14	243.56	11	241.26
15	533.71		-	15	614.11		-	15	410.63	11	404.01	15	332.83	12	329.28
16	538.97	11	541.55	16	726.26	11	704.15	16	545.03	12	535.32	16	377.16		-
17	596.06		-	17	906.28		-	17	621.95		-	17	440.77	13	435.51
18	602.77	12	596.09	18	913.69	12	884.92	18	702.30	13	688.64	18	568.51	14	561.04
19	657.87		-	19	1113.7	13	1080.1	19	882.95	14	864.40	19	623.05		-
20	717.37	13	731.74	20	1218.0		-	20	927.18		-	20	717.04	15	706.74
Max RE:		3.1718%		Max RE:		3.1486%		Max RE:		2.1018%		Max RE:		1.4361%	

Table 4.4: Eigenvalues of a Timoshenko cantilever beam vs the eigenvalues of a cantilever two-dimensional elastic body. *RE is the relative error.

This table shows that the eigenvalues of the Timoshenko model and the two-dimensional model compare very well. The non-beam type eigenvalues are highlighted in grey. For a short thick beam ($\alpha = 300$), the maximum relative error for the first 20 two-dimensional eigenvalues is just over 3%, while for a long thin beam ($\alpha = 10800$), the maximum relative error is just over 1%. This shows that as the beam becomes longer and thinner, the Timoshenko beam is a better approximation of the two-dimensional beam. But overall the Timoshenko beam compares very well.

This table also shows that as the beam gets more narrow, there are less non-beam type eigenvalues within the first few eigenvalues.This would also indicate that the Timoshenko beam would be a better approximation of the two-dimensional beam as the beam gets more narrow since the two-dimensional model behaves ‘more like a beam’.

4.6 Empirical and numerical examination of a Timoshenko beam

Consider a study by Stephen and Puchegger in 2006, article [SP06]. This study investigated the validity of the Timoshenko beam theory by hand of empirical and numerical data. The approach of the study is to compare the natural frequencies of a Timoshenko beam, to that of a three-dimensional elastic beam. The authors conducted comparisons between the models using theoretical methods as well as results from an empirical study on a physical beam, conducted by the authors.

The authors decided on a free-free beam configuration. The natural frequencies of the Timoshenko beam theory was obtained by using frequency equations from an article by Levison and Cooke [**LC81**]. In the theoretical approach of the three-dimensional beam, two methods were considered. A commercial finite elements method (ANSYS), and a resonant ultrasound spectroscopy (RUS) technique.

This section discusses the results of [SP06].

4.6.1 Mathematical models

Let Ω be the reference configuration of a free-free beam with square cross-section. The authors of [SP06] do not formulate the models in the article. Consider Problem T-4 from Section 1.3.3 for the free-free Timoshenko beam and Problem 3D-2 from Section 1.1.3 for the free-free three-dimensional beam. The boundary conditions and reference configurations of the model problems are given below.

Problem T-4

Find a function u, satisfying the equations of motion (1.3.5)- (1.3.6) and constitutive equations (1.3.7)- (1.3.8).

Free-Free Boundary Conditions:

$$V(0,\cdot) = 0, \qquad M(0,\cdot) = 0, \tag{4.6.1}$$
$$V(1,\cdot) = 0, \qquad M(1,\cdot) = 0. \tag{4.6.2}$$

Problem 3D-2

Find a vector valued function u, satisfying the equation of motion (1.1.11) and constitutive equation (1.1.13). Let Ω represent the reference configuration of the beam with a square cross-section.

$$\Omega := \left\{ \bar{x} = \langle x, y, z \rangle \in R \mid 0 \leq x \leq 1, \ -\frac{h}{2} \leq y, z \leq \frac{h}{2} \right\}$$

with h the height and width of the beam. $\partial\Omega$ denotes the boundary of Ω.

Free-Free Boundary Conditions:

$$Tn \ = \ 0 \quad \text{on } \partial\Omega.$$

with n a outward normal vector.

4.6.2 Suspended beam model

In the article [SP06], the beam is suspended at both ends. The beam is then vibrated and the induced motion causes the beam to transition into a free-free beam. This subsection is a short description of the suspended beam model.

The suspended Timoshenko beam model is referred to as Problem T-3 in this 1.3.3. The beam is suspended at both endpoints by linear springs. The boundary conditions of Problem T-3 as given in Section 1.3.3 can be rewritten by substituting the constitutive equations (1.3.7) and (1.3.8).

$$\partial_x w(0,\cdot) - \phi(0,\cdot) = kw(0,\cdot) \quad \text{and} \quad \partial_x \phi(0,\cdot) = 0$$
$$\partial_x w(1,\cdot) - \phi(1,\cdot) = -kw(0,\cdot) \quad \text{and} \quad \partial_x \phi(1,\cdot) = 0$$

The linear springs are allowed to take the weight of the beam and the system settles in a equilibrium state. The elongation of the springs from their natural length is denoted by h. The displacement from this equilibrium state should be considered.

Let $\langle w^*, \ \phi^* \rangle$ represent the solution of the model problem, and $\langle w^e, \ \phi^e \rangle$ the solution to the equilibrium problem. Define $\langle w, \ \phi \rangle = \langle (w^* - w^e), \ (\phi^* - \phi^e) \rangle$

such that it represents the deviation of the beam from the equilibrium solution. Substitution verifies that $\langle w,\ \phi\rangle$ satisfies the partial differential equation and boundary conditions.

If the beam is suspended by cables, in their loaded state they can be considered as linear springs. However if $w > h$ then the cables are no longer supporting the beam. The problem then becomes non-linear. Assume that the motion is small enough so that the problem remains linear.

Variational problem

Find w, ϕ such that for all $t > 0$, $\langle w, \phi\rangle \in C[0,1] \times C[0,1]$ and

$$\begin{aligned}\int_0^1 \partial_t^2 w v + \frac{1}{\alpha}\int_0^1 \partial_t^2 \phi\psi \quad &= \quad \int_0^1 (\partial_x w - \phi)(\psi - v') - \frac{1}{\beta}\int_0^1 \partial_x \phi \psi' \\ &\quad -kv(1)w(1,t) - kv(0)w(0,t)\end{aligned}$$

for all $v, \psi \in C[0,1]$.

This model is required for explanation in Section 4.6.

4.6.3 Experimental setup

Next the experiment conducted by [SP06] is discussed. A short, aluminium alloy beam with near square cross-section is suspended at both ends. The beam is suspended by carbon fibre loops.

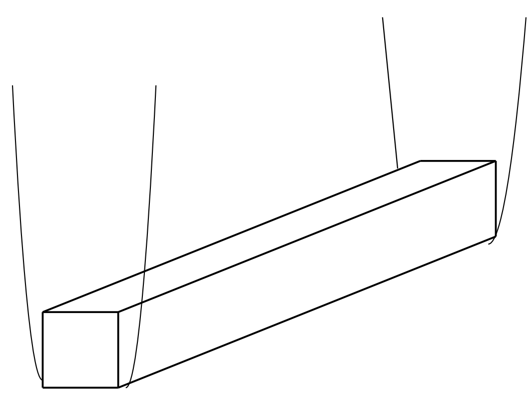


Figure 4.12: Sketch of beam suspended at the end-points by carbon fibre loops.

To vibrate the beam, the authors of [SP06] excited the carbon fibre loops and the frequencies were measured using two piezoelectric transducers. During the vibration, the beam will be momentarily free at both endpoints. The natural frequencies were obtained by sweeping through frequencies until a resonance was found.

The measured parameters of the beam is given in [SP06]. Since the physical beam is only nearly square, the plane has two distinct planes. The plane of the beam with a larger diameter is referred to as the stiff plane and the plane of the beam with smaller diameter is referred to as the flexible plane, by [SP06].

For the flexible plane $\alpha = \pm 190$ and for the stiff plane $\alpha = \pm 189$. The parameter $\gamma = 0.2830$. The natural frequencies f_k is defined as

$$f_k = \sqrt{\frac{\lambda_k}{2\pi}}$$

with λ_k the k'th eigenvalue. These natural frequencies f_k are dimensionless, and can be scaled back by multiplying the natural frequency f_k by $t_0 = \ell\sqrt{\frac{\rho}{G\kappa^2}}$, as defined in Chapter 1.

The cut-off frequency given in [SP06] can be expressed as

$$\omega_{co} = \sqrt{\frac{\kappa^2 AG}{\rho I}} = \frac{\alpha}{t_0}$$

with the dimensionless cut-off frequency α.

4.6.4 Results from SP06

The following table contains relevant results obtained by [SP06].

n	Measured	3D FEM	% Error	Timoshenko Beam	% Error	Side
1	27359.6	27417.6	0,21%	27407.1	0,17%	Flexible
	27423.8	27515.6	0,33%	27505.3	0,30%	Stiff
2	60862	60882.0	0,03%	60851.1	-0,02%	Flexible
	61098.3	61022.0	-0,12%	60992.1	-0,17%	Stiff
3	97609.5	97734.4	0,13%	97796.0	0,19%	Flexible
	97852.4	97881.5	0,03%	97945.4	0,09%	Stiff
4	161494	131658	0,12%	132277	0,60%	Flexible
	131732	131675	-0,04%	132308	0,44%	Stiff
5	161352	161390	0,02%	163547	1,36%	Stiff
	161538	161517	-0,01%	163611	1,31%	Flexible
6	165183	164887	-0,18%	169108	2,38%	Stiff
	165598	165398	-0,12%	169634	2,44%	Flexible
7	194863	194933	0,04%	202352	3,84%	Flexible
	194973	195032	0,03%	202115	3,66%	Stiff
8	195869	195977	0,06%	203319	3,80%	Stiff
	195908	196097	0,10%	203518	3,88%	Flexible
9	213501			241202	12,97%	Flexible
	213635			241067	12,84%	Stiff
10 and 11	220556			247954	12,42%	Flexible
	220702			281542	27,57%	Flexible
	221010			247782	12,11%	Stiff
	221092			281408	27,28%	Stiff

Table 4.5: Results from [SP06] (excluding RUS).

Table 4.3 shows that the Timoshenko model compares well to the measured results from the experiment. The first 8 natural frequencies for the stiff and flexible planes are very close to the natural frequencies of the three-dimensional and physical beam.

5 Finite element method

5.1 Introduction

In this chapter, the Finite Element Method (FEM) is applied to the models of this dissertation, presented in Chapter 1. The goal of this chapter is to obtain an algorithm that can be used to calculate the eigenvalues and eigenvectors for the models. This is in preparation of Chapter 6, where the eigenvalues and eigenvectors of the models are compared.

In Chapter 4, a method was discussed to obtain the eigenvalues and eigenvectors for the Timoshenko beam model. Therefore it is not necessary to apply the Finite Element Method on the Timoshenko model to obtain the eigenvalues and eigenvectors. The Finite Element Method is applied to the two-dimensional and three-dimensional models.

The outline of the chapter is as follows.

> Section 5.2 FEM for a cantilever two-dimensional elastic body. This is problem 2D-1 in Section 1.2.3.
>
> Section 5.3 FEM for a cantilever three-dimensional elastic body. This is problem 3D-1 in Section 1.1.3.
>
> Section 5.4 FEM for a cantilever Reissner-Mindlin plate. This is problem P-1 in Section 1.4.4.

5.2 A cantilever two-dimensional body

Consider a cantilever two-dimensional elastic body, with a rectangular cross-section.

Reference configuration for rectangular cross-section

Let $\{e_1, e_2\}$ be a right-handed orthonormal basis for R^2. Denote the elastic body by $\Omega \in R^2$, with $(0,0)$ as the point of reference. For a rectangular cross-section, the body Ω can be described as,

$$\Omega = \left\{ x \in E_2 \mid 0 \leq x_1 \leq 1,\ -\frac{h}{2} \leq x_2 \leq \frac{h}{2} \right\},$$

where $\partial\Omega$ denotes the boundary of Ω. The boundary $\partial\Omega$ can be divided into the four distinct lines (or edges) as follows:

$$\begin{aligned} \Sigma: \quad & x_1 &= 0 \qquad\qquad & \Gamma_1: \quad & x_2 &= -h/2 \\ \Gamma_3: \quad & x_1 &= 1 \qquad\qquad & \Gamma_2: \quad & x_2 &= h/2 \end{aligned}$$

In this configuration, the body is clamped rigidly to the surface at $x_1 = 0$ denoted as Σ and the body is free-hanging on the other boundaries denoted by Γ.

Cantilever elastic body

Consider a two-dimensional elastic body clamped rigidly to a surface at $x_1 = 0$. The body is free-hanging on the other boundaries. This is Problem 2D-1 in Section 1.2.3.

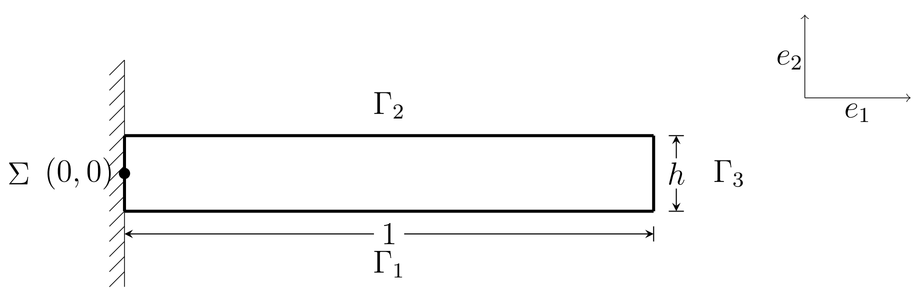


Figure 5.1: A cantilever two-dimensional elastic body.

In Section 1.2.4 the variational problem for the two dimensional cantilever model is derived and refered to as Problem 2D-1V. Some of the results from Section 1.2 are repeated here for convenience. Using the reference configuration, these results are rewritten from a general form to a model specific form.

Problem 2D-1V

Find a function u such that for all $t > 0$, $u \in T(\Omega)$ and

$$c(u, \phi) = -b(u, \phi) + (Q, \phi) \tag{5.2.1}$$

for all $\phi \in T(\Omega)$. With the test function space

$$T(\Omega) \;\; = \;\; \left\{ \phi \in C^1(\bar{\Omega}) \mid \phi = 0 \;\; \text{on} \;\; \Gamma \right\}.$$

The bilinear forms and integral function is defined by

$$b(u, \phi) \;\; = \;\; \int_\Omega c_1 \mathrm{Tr}(\mathcal{E}\Phi) + c_2 \mathrm{Tr}(\mathcal{E})\mathrm{Tr}(\Phi) \; dA, \tag{5.2.2}$$

$$c(\partial_t^2 u, \phi) \;\; = \;\; \int_\Omega (\partial_t^2 u) \cdot \phi \; dA, \tag{5.2.3}$$

$$(f, g) \;\; = \;\; \int_\Omega f \cdot g \; dA, \tag{5.2.4}$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1-\nu^2)}$.

Using the reference configuration, the constitutive equations and the bilinear form b can be rewritten into a model specific form.

Constitutive equations:

$$\sigma_{11} \;\; = \;\; \frac{1}{\gamma(1-\nu^2)}(\partial_1 u_1 + \nu \partial_2 u_2) \tag{5.2.5}$$

$$\sigma_{22} \;\; = \;\; \frac{1}{\gamma(1-\nu^2)}(\partial_2 u_2 + \nu \partial_1 u_1) \tag{5.2.6}$$

$$\sigma_{12} \;\; = \;\; \frac{1}{2\gamma(1+\nu)}(\partial_1 u_2 + \partial_2 u_1) \tag{5.2.7}$$

Bilinear form:

$$\begin{aligned} b(u, \phi) = \;\; & \frac{1}{\gamma(1-\nu^2)} \int_\Omega (\partial_1 u_1 \partial_1 \phi_1 + \partial_2 u_2 \partial_2 \phi_2 + \nu \partial_1 u_1 \partial_2 \phi_2 + \nu \partial_2 u_2 \partial_1 \phi_1) \; dA \\ + \;\; & \frac{1}{2\gamma(1+\nu)} \int_\Omega (\partial_1 u_2 \partial_1 \phi_2 + \partial_1 u_2 \partial_2 \phi_1 + \partial_2 u_1 \partial_1 \phi_2 + \partial_2 u_1 \partial_2 \phi_1) \; dA. \end{aligned} \tag{5.2.8}$$

5.2.1 Weak variational form

Define the inertia space V as the closure of $T(\Omega)$ in $H := H^1(0,1) \times H^1(0,1)$. Denote $X = L^2(0,1) \times L^2(0,1)$. The inertia space is $W = X$ with norm $||\cdot||_W = \sqrt{c(\cdot,\cdot)}$.

Problem 2D-1WV

Find a function u such that $\forall\ t > 0$, $u \in V$ with $\partial_t^2 u \in W$ and

$$c(u,v) + b(u,v) \quad = \quad (Q,v)$$

for all $v \in V$.

See Chapter 2 for the existence theory.

5.2.2 Galerkin approximation

To be able to apply the Finite Element Method to Problem 2D-1V, the body Ω needs to be discretised. This is done by dividing the body Ω into discrete shapes, called elements. There are various types of shapes of elements that can be used. Since the body ω is has a rectangular cross-section, rectangular elements are the simplest elements to use.

Divide the reference configuration Ω into a grid of rectangular elements, such that there are $n = n_1 \times n_2$ nodes.

Define a set of n-dimensional linear independent basis functions. For the two dimensional model, the basis functions can be defined by the set

$$B = \{\langle\phi_1, 0\rangle, \langle\phi_2, 0\rangle, ..., \langle\phi_n, 0\rangle, \langle 0, \phi_1\rangle, \langle 0, \phi_2\rangle, ..., \langle 0, \phi_n\rangle\}.$$

These functions are chosen as piecewise Hermite bi-cubic functions ϕ_i. Simpler bi-linear functions can also be used, however the use of the bi-cubic basis functions results in faster convergence and also the benefit of obtaining the derivatives of the solution at the expense of more complexity.

The subset of basis functions B that satisfies all the conditions of the test function space $T(\Omega)$ are called the admissible basis functions. Denote the admissible basis functions by δ_j, where each δ_j is a unique element of B. The admissible basis functions can be numbered and expressed as the set $A = \{\delta_1, \delta_2, ..., \delta_k\}$ for some $k \leq 2n$.

Define the space

$$S^h = \operatorname{span}\{\delta_i \mid i = 1, 2, ..., k\}.$$

For each function $u^h \in S^h$, u^h can be expressed as

$$u^h = \sum_{i=1}^{k} u_i(t)\delta_i(x).$$

Substitution of u^h into Problem 2D-1V, results in the following Galerkin Approximation, denoted by Problem 2D-1G.

Problem 2D-1G

Find a function u^h such that for all $t > 0$, $u^h \in S^h$ and

$$(u^h, \phi_i) = -b(u^h, \phi_i) + (Q^I, \phi_i)$$

for $i = 1, 2, ..., k$. Q^I is scalar vector obtained after interpolating the function Q over the rectangular grid Ω. i.e. $Q^I_{i,j} = Q(x_i, x_j)$ for $i = 1, 2, ..., n_1$ and $j = 1, 2, ..., n_2$.

5.2.3 System of differential equations

Consider the following standard Finite Element Method matrices

FEM matrices

$$\begin{aligned} \mathbf{M}_{j,i} &= \int_\Omega \phi_i\phi_j \; dA \\ \mathbf{K_{11}}_{j,i} &= \int_\Omega \partial_1\phi_i\partial_1\phi_j \; dA \\ \mathbf{K_{22}}_{j,i} &= \int_\Omega \partial_2\phi_i\partial_2\phi_j \; dA \end{aligned} \qquad \begin{aligned} \mathbf{K_{12}}_{j,i} &= \int_\Omega \partial_2\phi_i\partial_1\phi_j \; dA \\ \mathbf{K_{21}}_{j,i} &= \int_\Omega \partial_1\phi_i\partial_2\phi_j \; dA \end{aligned}$$

for $i, j = 1, 2, ..., k$.

And

$$M_{F\,j,i} = \int_\Omega \phi_i\phi_j \; dA$$

for $i = 1, 2, ..., k$ and for $j = 1, 2, ..., 2n$.

Define the following matrices:

$$\begin{aligned} K_1 &= \frac{1}{\gamma(1-\nu^2)}\mathbf{K_{11}} + \frac{1}{2\gamma(1+\nu)}\mathbf{K_{22}} \\ K_2 &= \frac{\nu}{\gamma(1-\nu^2)}\mathbf{K_{21}} + \frac{1}{2\gamma(1+\nu)}\mathbf{K_{12}} \\ K_3 &= \frac{\nu}{\gamma(1-\nu^2)}\mathbf{K_{12}} + \frac{1}{2\gamma(1+\nu)}\mathbf{K_{21}} \\ K_4 &= \frac{1}{\gamma(1-\nu^2)}\mathbf{K_{22}} + \frac{1}{2\gamma(1+\nu)}\mathbf{K_{11}} \end{aligned}$$

Using the standard FEM matrices and matrices K_1-K_4, the following concatenated matrices are defined.

$$K = \begin{bmatrix} K1 & K2 \\ K3 & K4 \end{bmatrix} \qquad M_f = \begin{bmatrix} M_F & O_F \\ O_F & M_F \end{bmatrix} \tag{5.2.9}$$

$$M = \begin{bmatrix} \mathbf{M} & O \\ O & \mathbf{M} \end{bmatrix} \tag{5.2.10}$$

The matrices O and O_F are the zero matrices of the same size as $\mathbf{M}$ and M_f respectively.

Using (5.2.9) and (5.2.10), Problem 2D-1G is rewritten as a system of ordinary differential equations. This system is referred to as Problem 2D-1ODE.

Problem 2D-1ODE

Find a function $\bar{u} \in S^h$ such that

$$M\ddot{\bar{u}} = K\bar{u} + M_f Q^I. \tag{5.2.11}$$

With $\bar{u}$ in the form $\bar{u} = \langle u, \partial_1 u, \partial_2 u, \partial_{12} u \rangle$.

Remark This form of $\bar{u}$ is determined by the use of the bi-cubic basis functions.

5.2.4 Eigenvalue problem

For the eigenvalue problem, assume that there is no external force, $M_f Q^I = 0$, so that

$$M\ddot{\bar{u}} = K\bar{u}. \tag{5.2.12}$$

It is known that a system of ordinary differential equations has a general solution of the form e^{rt}. Suppose that $\bar{w} = e^{\lambda t}\bar{u}$ is a solution for (5.2.12). In this solution, λ is an eigenvalue and $\bar{u}$ a corresponding eigenfunction. Substitution into (5.2.12) results in

$$M\lambda e^{\lambda t}\bar{u} = Ke^{\lambda t}\bar{u}.$$

Since $e^{\lambda t} > 0$ for all values of λt, we can cancel it from both sides of the equation and formulate the eigenvalue problem Problem 2D-1E.

Problem 2D-1E

Find a real number λ and a function $\bar{u} \in S^h$ such that

$$M\lambda\bar{u} = K\bar{u}. \tag{5.2.13}$$

In this section, a similar approach is applied to the three-dimensional elastic body.

5.3 A cantilever three-dimensional body

Consider a cantilever three-dimensional elastic body, with a rectangular cross-section.

Reference configuration for rectangular cross-section

Let $\{e_1, e_2, e_3\}$ be a right-handed orthonormal basis for R^3. Denote the elastic body as $\Omega \in R^3$ with $(0, 0, 0)$ the point of reference. For a rectangular cross-section, the body Ω can be described as

$$\Omega = \left\{ x \in R^3 \mid 0 \le x_1 \le 1,\ -\frac{h}{2} \le x_2 \le \frac{h}{2},\ -\frac{b}{2} \le x_3 \le \frac{b}{2} \right\}$$

Let $\partial\Omega$ denote the boundary of the body. Divide $\partial\Omega$ into the six distinct flat surfaces as follows:

$$
\begin{array}{llllll}
\Sigma : & x_1 & = 0 & \Gamma_2 : & x_2 & = -h/2 \\
\Gamma_5 : & x_1 & = 1 & \Gamma_3 : & x_3 & = b/2 \\
\Gamma_1 : & x_3 & = -b/2 & \Gamma_4 : & x_2 & = h/2
\end{array}
$$

Using this notation, the boundary conditions are similar to the two-dimensional model in Section 5.2, where the body is clamped rigidly at Σ, and free-hanging on the other sides denoted by Γ.

Cantilever elastic body

Consider a three-dimensional elastic body with rectangular cross-section, rigidly clamped to a surface at attached at the side Σ and free-hanging at all the other sides. This is Problem 3D-1 in Section 1.1.3.

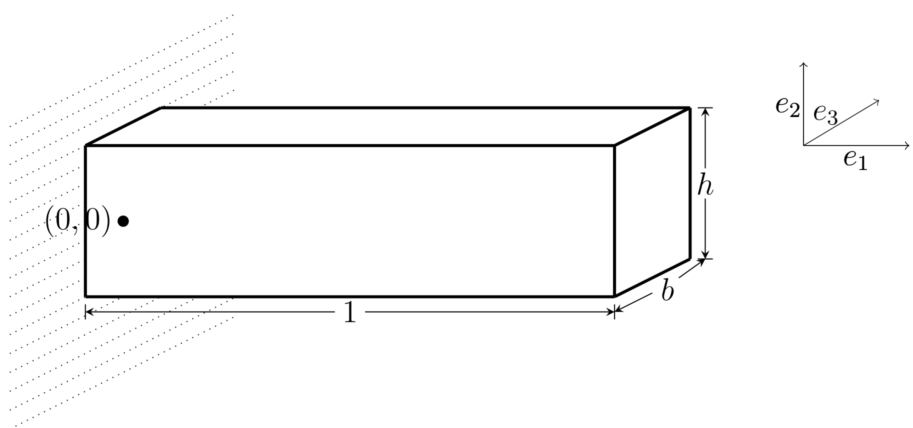


Figure 5.2: Cantilever Three-Dimensional Elastic Body with Rectangular Cross-Section.

In Section 1.1.4, the variational problem for the three-dimensional cantilever model is defined by Problem 3D-1V. This general form can now be rewritten as in a model specific form using the reference configuration. For convenience, some of the results from Section 1.1 are repeated here.

Problem 3D-1V

Find a function u such that for all $t > 0$, $u \in T(\Omega)$ and

$$c(u, \phi) = -b(u, \phi) + (Q, \phi) \tag{5.3.1}$$

for all $\phi \in T(\Omega)$.

With the test function space

$$T(\Omega) \;=\; \{\phi \in C(\Omega) \mid \phi = 0 \;\text{ on }\; \Gamma\}\,.$$

The bilinear forms and integral function are defined by

$$b(u,\phi) \;=\; \int_\Omega c_1 \mathrm{Tr}(\mathcal{E}\Phi) + c_2 \mathrm{Tr}(\mathcal{E})\mathrm{Tr}(\Phi)\; dV, \tag{5.3.2}$$

$$c(u,\phi) \;=\; \int_\Omega (\partial_t^2 u)\cdot\phi \; dV, \tag{5.3.3}$$

$$(f,g) \;=\; \int_\Omega f\cdot g \; dV, \tag{5.3.4}$$

$$\tag{5.3.5}$$

with $c_1 = \dfrac{1}{\gamma(1+\nu)}$ and $c_2 = \dfrac{\nu}{\gamma(1+\nu)(1-2\nu)}$.

Using the definition of the reference configuration, the constitutive equations and the bilinear form b can be rewritten in the following model specific forms:

Constitutive equations

$$\begin{aligned}
\sigma_{11} &= \frac{1}{\gamma(1+\nu)}\partial_1 u_1 + \frac{\nu}{\gamma(1+\nu)(1-2\nu)}(\partial_1 u_1 + \partial_2 u_2 + \partial_3 u_3)\\
\sigma_{22} &= \frac{1}{\gamma(1+\nu)}\partial_2 u_2 + \frac{\nu}{\gamma(1+\nu)(1-2\nu)}(\partial_1 u_1 + \partial_2 u_2 + \partial_3 u_3)\\
\sigma_{33} &= \frac{1}{\gamma(1+\nu)}\partial_3 u_3 + \frac{\nu}{\gamma(1+\nu)(1-2\nu)}(\partial_1 u_1 + \partial_2 u_2 + \partial_3 u_3)\\
\sigma_{23} &= \frac{1}{2\gamma(1+\nu)}(\partial_3 u_2 + \partial_2 u_3)\\
\sigma_{31} &= \frac{1}{2\gamma(1+\nu)}(\partial_3 u_1 + \partial_1 u_3)\\
\sigma_{12} &= \frac{1}{2\gamma(1+\nu)}(\partial_2 u_1 + \partial_1 u_2)
\end{aligned}$$

Bilinear Form

$$\begin{aligned}
b(u,\phi) &= \int_\Omega c_1 \mathrm{Tr}(\mathcal{E}\Phi) + c_2 \mathrm{Tr}(\mathcal{E})\mathrm{Tr}(\Phi)\; dV\\
&= \int_\Omega \sigma_{11}\partial_1\phi_1 + \sigma_{12}\partial_1\phi_2 + \sigma_{13}\partial_1\phi_3 + \sigma_{21}\partial_2\phi_1 + \sigma_{22}\partial_2\phi_2 + \sigma_{23}\partial_2\phi_3 \; dV
\end{aligned}$$

5.3.1 Weak variational form

Bilinear form

Define the inertia space V as the closure of $T(\Omega)$ in $H := H^1(0,1) \times H^1(0,1) \times H^1(0,1)$. Denote $X = L^2(0,1) \times L^2(0,1) \times L^2(0,1)$. The inertia space is $W = X$ with norm $||\cdot||_W = \sqrt{c(\cdot,\cdot)}$.

Problem 3D-1W

Find a function u such that for all $t > 0$, $u(t) \in V$ and $u''(t) \in W$, satisfying the following equation

$$c(u,v) + b(u,v) \quad = \quad (Q,v) \quad \text{ for each } v \in V. \tag{5.3.6}$$

See Chapter 2 for the existence theory.

5.3.2 Galerkin approximation

As mentioned in the previous section, Section 5.2.2, the body Ω needs to be descritised. For this three-dimensional body, three-dimensional elements shapes are required. The elements used in this dissertation are rectangular prismatic elements. These elements are also known as 'brick-shaped' elements and is described in [Wu06]. This shape of element is a natural choice for a three-dimensional elastic body with a rectangular cross-section.

Divide the reference configuration Ω into a grid of rectangular prismatic elements, such that there are $n = n_1 \times n_2 \times n_3$ nodes.

Define a set of n-dimensional linear independent basis functions. For the three-dimensional model, the basis functions can be defined by the set

$$\begin{aligned} B = \{ & \langle \phi_1, 0, 0\rangle, \langle \phi_2, 0, 0\rangle, ..., \langle \phi_n, 0, 0\rangle, \\ & \langle 0, \phi_1, 0\rangle, \langle 0, \phi_2, 0\rangle, ..., \langle 0, \phi_n, 0\rangle, \\ & \langle 0, 0, \phi_1\rangle, \langle 0, 0, \phi_2\rangle, ..., \langle 0, 0, \phi_n\rangle\}. \end{aligned}$$

For this three-dimensional model, piecewise Hermite tri-cubic basis functions are used. Although this is more complex than using piecwise tri-linear basis functions, as mentioned in Section 5.2.2, the tri-cubic basis functions ensure faster convergence and also the derivatives of the solutions.

Recall that the admissible basis functions, are all the basis functions that satisfies all the conditions of the test function space $T(\Omega)$. Denote the admissible basis functions by δ_j, with each δ_j a unique element of B. The set of admissible basis functions can then be expressed as $A = \{\delta_1, \delta_2, ..., \delta_k\}$ with $k \leq 3n$.

Define the space

$$S^h \quad = \quad \text{span}\,\{\delta_i \mid i = 1, 2, ..., k\}$$

For each function $u^h \in S^h$, u^h can be expressed as

$$u^h = \sum_{i=1}^{k} u_i(t)\delta_i(x)$$

Substituting u^h into Problem 3D-1V, results in the following Galerkin approximation, denoted by Problem 3D-1G.

Problem 3D-1G

Find a function u^h such that for all $t > 0$, $u^h \in S^h$ and

$$(u^h, \phi_i) \quad = \quad -b(u^h, \phi_i) + (Q^I \cdot \phi_i)$$

for $i = 1, 2, ..., k$. Q^I is a scalar vector obtained after interpolating the function Q over the rectangular grid Ω, i.e. $Q^I_{i,j,h} = Q(x_i, x_j, x_h)$ for $i = 1, 2, ..., n_1$, $j = 1, 2, ..., n_2$ and $h = 1, 2, ..., n_3$.

5.3.3 System of ordinary differential equations

Consider the following standard Finite Element Method matrices.

FEM matrices

$$\begin{aligned}
\mathbf{M}_{ij} &= \int_\Omega \phi_j \phi_i \; dV \\
K_{11ij} &= \int_\Omega \partial_1\phi_j \partial_1\phi_i \; dV \\
K_{12ij} &= \int_\Omega \partial_1\phi_j \partial_2\phi_i \; dV \\
K_{13ij} &= \int_\Omega \partial_1\phi_j \partial_3\phi_i \; dV
\end{aligned}$$

$$\begin{aligned}
K_{22ij} &= \int_\Omega \partial_2\phi_j \partial_2\phi_i \; dV \\
K_{23ij} &= \int_\Omega \partial_2\phi_j \partial_3\phi_i \; dV \\
K_{33ij} &= \int_\Omega \partial_3\phi_j \partial_3\phi_i \; dV
\end{aligned}$$

for $i, j = 1, 2, ..., k$.

And

$$\mathbf{M_f}_{ij} = \int_\Omega \phi_j \phi_i \, dV$$

for $i = 1, 2, ..., k$ and for $j = 1, 2, ..., 3n$.

The remaining matrices can be defined as

$$\begin{aligned} K_{21} &= {K_{12}}^T, \\ K_{31} &= {K_{13}}^T, \\ K_{32} &= {K_{23}}^T. \end{aligned}$$

Define the following matrices:

$$\begin{aligned} \mathbf{K11} &= (C_1 + C_2) K_{11} + C_3 K_{22} + C_3 K_{33} \\ \mathbf{K12} &= C_3 K_{12} + C_2 K_{21} \\ \mathbf{K13} &= C_3 K_{13} + C_2 K_{31} \\ \mathbf{K21} &= C_3 K_{21} + C_2 K_{12} \\ \mathbf{K22} &= (C_1 + C_2) K_{22} + C_3 K_{11} + C_3 K_{33} \end{aligned}$$

$$\begin{aligned} \mathbf{K23} &= C_3 K_{23} + C_2 K_{32} \\ \mathbf{K31} &= C_3 K_{31} + C_2 K_{13} \\ \mathbf{K32} &= C_3 K_{32} + C_2 K_{23} \\ \mathbf{K33} &= (C_1 + C_3) K_{33} + C_3 K_{11} + C_3 K_{22} \end{aligned}$$

with $C_1 = \frac{1}{\gamma(1+\nu)}$, $C_2 = \frac{\nu}{\gamma(1+\nu)(1-2\nu)}$ and $C_3 = \frac{1}{2\gamma(1-2\nu)}$.

Using the standard FEM matrices and the matrices $K11$ to $K33$, the following concatenated matrices are defined:

$$K = \begin{bmatrix} \mathbf{K11} & \mathbf{K12} & \mathbf{K13} \\ \mathbf{K21} & \mathbf{K22} & \mathbf{K23} \\ \mathbf{K31} & \mathbf{K32} & \mathbf{K33} \end{bmatrix} \qquad M = \begin{bmatrix} \mathbf{M} & O & O \\ O & \mathbf{M} & O \\ O & O & \mathbf{M} \end{bmatrix} \tag{5.3.7}$$

$$M_f = \begin{bmatrix} \mathbf{M_f} & O_f & O_f \\ O_f & \mathbf{M_f} & O_f \\ O_f & O_f & \mathbf{M_f} \end{bmatrix} \tag{5.3.8}$$

The matrices O and O_f are the zero matrices of the same size as $\mathbf{M}$ and $\mathbf{M_f}$ respectively.

Using (5.3.7) and (5.3.8), Problem 3D-1G is rewritten as a system of ordinary differential equations. This system is referred to as Problem 3D-1ODE

Problem 3D-1ODE

Find a function $\bar{u} \in S^h$ such that

$$M\ddot{\bar{u}} = K\bar{u} + M_f Q^I. \tag{5.3.9}$$

With $\bar{u}$ in the form $\bar{u} = \langle \partial_1^i u,\ \partial_2^j u,\ \partial_3^k u\rangle$ for $i, j, k = 0, 1, 2, 3$.

5.3.4 Eigenvalue problem

The derivation and form of the eigenvalue problem for the three-dimensional elastic body is similar to the two-dimensional model given in Section 5.2.4.

Problem 3D-1E

Find a real number λ and a function $\bar{u} \in S^h$ such that

$$M\lambda\bar{u} = K\bar{u}. \tag{5.3.10}$$

5.4 Cantilever plate model

Consider a rectangular cantilever Reissner-Mindlin plate model with a rectangular cross-section.

Reference configuration for a rectangular plate

Let $\{e_1, e_2\}$ be a right-handed orthonormal basis for R^2. Although this basis seems identical to the two-dimensional beam in Section 5.2, it is fact different. It would be more appropriate to use e_1 and e_3 for this plate model. However the use of e_1 and e_2 is kept to reiterate that this is in fact a two-dimensional model.

Denote the elastic body as $\Omega \in R^2$ with the reference point $(0, 0)$. For a rectangular plate,

$$\Omega = \left\{x \in R^2 \mid 0 \leq x_1 \leq 1,\ 0 \leq x_2 \leq b\right\}.$$

Let $\partial\Omega$ denote the boundary of plate. The boundary $\partial\Omega$ can be divided into four distinct lines.

$$\begin{array}{llcllll} \Sigma: & x_1 & = 0 & \qquad\qquad & \Gamma_0: & x_2 & = 0 \\ \Gamma_3: & x_1 & = 1 & & \Gamma_1: & x_2 & = b \end{array}$$

Similar to the two and three-dimensional cantilever models, this notation implies that the plate is clamped at Σ and free hanging at Γ.

Cantilever plate model

Consider a rectangular Reissner-Mindlin plate clamped rigidly to a surface at Σ and free hanging on the remaining edges. This plate model is presented in Section 1.4.4 as Problem P-1.

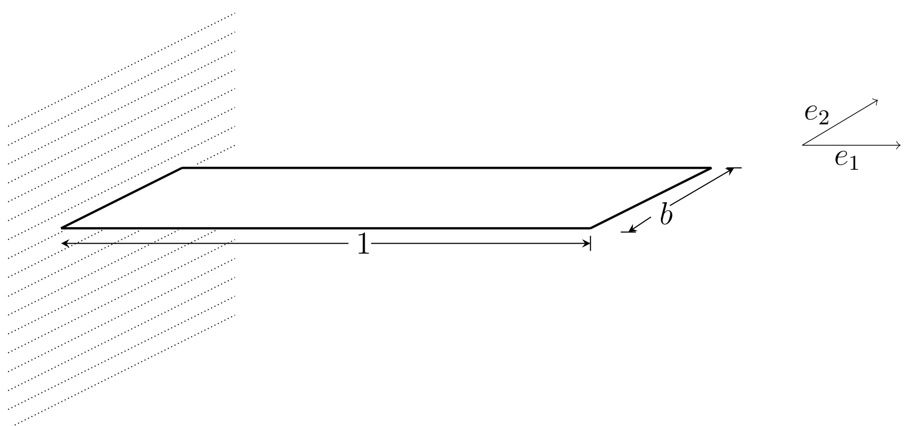


Figure 5.3: Two-dimensional cantilever Reissner-Mindlin plate

In Section 1.4.4 the variational problem for the cantilever Reissner-Mindlin plate is defined by Problem P-1V. For convenience, the relevant results from Section 1.4.4 are repeated.

Problem P-1V

Find a function $u = \langle w, \psi \rangle$, such that for all $t > 0$, $u \in T_1(\bar{\Omega}) \times T_2(\bar{\Omega})$ and the following equations are satisfied

$$c(u, \phi) \quad = \quad -b(u, \phi) + (Q, \phi), \tag{5.4.1}$$

with $\phi = \langle v, \phi \rangle \in T_1(\bar{\Omega}) \times T_2(\bar{\Omega})$ an arbitrary function.

With the test function spaces

$$\begin{aligned} T_1(\bar{\Omega}) &= \left\{ v \in C^1(\bar{\Omega}) \mid v = 0 \text{ on } \Sigma_0 \right\}, \\ T_2(\bar{\Omega}) &= \left\{ \phi = [\phi_1\ \phi_2]^T \mid \phi_1, \phi_2 \in C^1(\bar{\Omega}),\ \phi_1 = \phi_2 = 0 \text{ on } \Sigma_0 \right\}. \end{aligned}$$

The bilinear forms and integral function defined by

$$\begin{aligned} b(u,\phi) &= \int_\Omega \boldsymbol{Q} \cdot \nabla v\ dA + \int_\Omega \mathrm{Tr}(M\Phi)\ dA, \\ c(u,\phi) &= \int_\Omega h(\partial_t^2 w) v\ dA + \int_\Omega I(\partial_t^2 \psi) \cdot \phi\ dA \\ (f,g) &= -\int_\Omega f \cdot g\ dA \end{aligned}$$

Using the definition of the reference configuration, the constitutive equations and the bilinear form b can be rewritten as follows.

Constitutive Equations

$$\boldsymbol{Q} = h(\nabla w + \boldsymbol{\psi}) \tag{5.4.2}$$

$$M_{11} = \frac{1}{2\beta(1-\nu^2)} \left[2(\partial_1 \psi_1 + \nu \partial_2 \psi_2\right] \tag{5.4.3}$$

$$M_{12} = M_{21} = \frac{1}{2\beta(1-\nu^2)} \left[(1-\nu)(\partial_1 \psi_2 + \partial_2 \psi_1)\right] \tag{5.4.4}$$

$$M_{11} = \frac{1}{2\beta(1-\nu^2)} \left[2(\partial_2 \psi_2 + \nu \partial_1 \psi_1)\right] \tag{5.4.5}$$

Bilinear Form

$$\begin{aligned} b(u,\phi) &= \int_\Omega \boldsymbol{Q} \cdot \nabla v\ dA + \int_\Omega \mathrm{Tr}(M\Phi)\ dA, \\ &+ \frac{1}{\beta(1-\nu^2)} \int_\Omega (\partial_1 \psi_1 + \nu \partial_2 \psi_2) \partial_1 \phi_1 + (\partial_2 \psi_2 + \nu \partial_1 \psi_1) \partial_2 \phi_2\ dA, \\ &+ \frac{1}{2\beta(1+\nu)} \int_\Omega (\partial_1 \psi_2 + \partial_2 \psi_1)(\partial_1 \phi_2 + \partial_2 \psi_1)\ dA. \end{aligned}$$

5.4.1 Weak variational form

Similar to Section 2.1, the weak variational form for Problem P-1 can be derived from Problem P-1V.

Bilinear forms

From the bilinear form, we have

$$\begin{aligned} b(f_2, g_2) &= \frac{1}{\beta(1-\nu^2)} \iint_\Omega (\partial_1 f_{2,1} + \nu \partial_2 f_{2,2}) \partial_1 g_{2,1} + (\partial_2 f_{2,2} + \nu \partial_1 f_{2,1}) \partial_2 g_{2,2} \; dA, \\ &+ \frac{1}{2\beta(1+\nu)} \iint_\Omega (\partial_1 f_{2,2} + \partial_2 f_{2,1})(\partial_1 g_{2,2} + \partial_2 g_{2,1}) \; dA \end{aligned}$$

For all $f, g \in T_1(\Omega) \times T_2(\Omega)$, define the bilinear forms

$$\begin{aligned} c(f,g) &= h(f_1, g_1)_\Omega + I(f_2, g_2)_{R^2} \\ b^*(f,g) &= b(f_2, g_2) + h(\nabla f_1 + f_2, \nabla g_1 + g_2)_{R^2} \end{aligned}$$

where the integrals are defined by

$$\begin{aligned} (f_1, g_1)_\Omega &= \iint_\Omega f_1 g_1 \; dA, \\ (f, g)_{R^2} &= \iint_\Omega f \cdot g \; dA. \end{aligned}$$

Define $V_1(0,1)$ as the closure of $T_1(0,1)$ in $H^1(0,1)$ and $V_2(0,1)$ as the closure of $T_2(0,1)$ in $H^1(0,1)^2$.

Denote the space $X = L^2(0,1) \times L^2(0,1)^2$ as a setting for Problem P-1V. A natural inner product for X is $(f,g)_X = (f_1, g_1)_\Omega + (f_2, g_2)_{R^2}$. Define W as the space X with the inner product c and $V = V_1(0,1) \times V_2(0,1)$

Problem Plate-1W

Find a function u such that for all $t > 0$, $u(t) \in V$, $u'(t) \in V$ and $u''(t) \in W$, satisfying the following equation

$$c(u''(t), v) + b^*(u(t), v) = (f(t), v)_\Omega \quad \text{for each } v \in V, \tag{5.4.6}$$

with $u(0) = u_0 = \langle w_0, \boldsymbol{\psi}_0 \rangle$, and $u'(0) = u_1 = \langle w_1, \boldsymbol{\psi}_1 \rangle$.

See Chapter 2 for the existence theory.

5.4.2 Galerkin approximation

To discretise the body Ω, the same shapes as in Section 5.2.2 are used.

Divide the reference configuration Ω into a rectangular grid of elements, such that there are $n = n_1 \times n_2$ nodes.

Define a set of n-dimensional linear independent basis functions. The basis functions can be defined by the set

$$B = \{\langle \phi_1, 0\rangle, \langle \phi_2, 0\rangle, ..., \langle \phi_n, 0\rangle, \langle 0, \phi_1\rangle, \langle 0, \phi_2\rangle, ..., \langle 0, \phi_n\rangle\}.$$

These basis functions are piecewise Hermite bi-cubic basis functions.

Since there are two test function spaces, $T_1(\Omega)$ and $T_2(\Omega)$, two different sets of admissible basis functions are required. Denote the admissible basis functions for $T_1(\Omega)$ by δ_j^1 where each δ_j^1 is a unique element of B. The set of admissible basis functions that satisfies $T_1(\Omega)$ can be defined as $A_1 = \left\{\delta_1^1, \delta_2^1, ..., \delta_{k_1}^1\right\}$ for a $k_1 \leq 2n$. Similarly for $T_2(\Omega)$, the set of admissible basis functions can be defined as $A_2 = \left\{\delta_1^2, \delta_2^2, ..., \delta_{k_2}^2\right\}$ for a $k_2 \leq 2n$.

Define the two spaces

$$\begin{aligned} S_1^h &= \operatorname{span}\left(\left\{\delta_i^1 \mid i = 1, 2, ..., k_1\right\}\right), \\ S_2^h &= \operatorname{span}\left(\left\{\delta_i^2 \mid i = 1, 2, ..., k_2\right\}\right). \end{aligned}$$

For each function $u^h \in S_1^h$ and each function $\psi^h \in S_2^h$, u^h can be expressed as

$$w^h = \sum_{i=1}^{k} w_i(t)\delta_i^1(x)$$

and ψ^h can be expressed as

$$\psi^h = \sum_{i=1}^{k} \psi_i(t)\delta_i^2(x)$$

Substitution of u^h and ψ^h into Problem P-1V, results in the following Galerkin approximation, denoted by Problem P-1G.

Problem P-1G

Find a function $u^h = \langle w^h, \psi^h \rangle$, such that for all $t > 0$, $u^h \in S_1^h \times S_1^h$ and the following equations are satisfied

$$c(u^h, \phi_{i,j}) = -b(u^h, \phi_{i,j}) + (Q^I, \phi_{i,j}), \tag{5.4.7}$$

with $\phi_{i,j} = \langle v_i, \phi_j \rangle$ for $i = 1, 2, ..., k_1$ and $j = 1, 2, ..., k_2$

5.4.3 System of ordinary differential equations

The standard FEM matrices K_{11}, K_{12}, K_{21}, K_{22} and M_F where presented in Section 5.2.3. Even though the definition of R^2 is different in Section 5.2, the definition of the matrices are the same and are not repeated here.

In addition to these matrices, the following standard FEM matrices are also required.

FEM matrices

$$L_{1ij} = \iint_\Omega \phi_j \partial_1 \phi_i \qquad L_{2ij} = \iint_\Omega \phi_j \partial_2 \phi_i$$

for $i, j = 1, 2, ..., p$.

Define the following matrices:

$$\begin{aligned}
\mathbf{K11} &= hK_{11} - hK_{22} \\
\mathbf{K12} &= hL_1 \\
\mathbf{K13} &= hL_2 \\
\mathbf{K21} &= hL_1^T \\
\mathbf{K22} &= AK_{11} + BK_{22} + h\mathbf{M}
\end{aligned} \qquad
\begin{aligned}
\mathbf{K23} &= A\nu K_{12} + BK_{21} \\
\mathbf{K31} &= hL_2^T \\
\mathbf{K32} &= A\nu K_{21} + BK_{12} \\
\mathbf{K33} &= AK_{22} + BK_{11} + h\mathbf{M}
\end{aligned}$$

with $A = \frac{1}{\beta(1-\nu^2)}$ and $B = \frac{1}{2\beta(1+\nu)}$.

Using the standard FEM matrices and the matrices **K11** to **K33**, the following concatenated matrices are defined.

$$K = \begin{bmatrix} \mathbf{K11} & \mathbf{K12} & \mathbf{K13} \\ \mathbf{K21} & \mathbf{K22} & \mathbf{K23} \\ \mathbf{K31} & \mathbf{K32} & \mathbf{K33} \end{bmatrix} \qquad M = \begin{bmatrix} h\mathbf{M} & O & O \\ O & I\mathbf{M} & O \\ O & O & I\mathbf{M} \end{bmatrix} \tag{5.4.8}$$

$$M_f = \begin{bmatrix} M_F & O_F & O_F \\ O_F & M_F & O_F \\ O_F & O_F & M_F \end{bmatrix} \qquad (5.4.9)$$

The matrices O and O_F are the zero matrices of the same size as $\mathbf{M}$ and $\mathbf{M_f}$ respectively.

Using (5.4.8) and (5.4.9), Problem P-1G is rewritten as a system of ordinary differential equations. This system is referred to as Problem P-1ODE

Problem P-1ODE

Find function $\bar{u} \in S_1^h \times S_2^h$ such that

$$M\ddot{\bar{u}} = K\bar{u} + M_f Q^I \qquad (5.4.10)$$

With $\bar{u}$ in the form $\bar{u} = \langle w, \partial_1 w, \partial_2 w, \partial_{12} w, \psi_1, \partial_1\psi_1, \partial_2\psi_1, \partial_{12}\psi_1, \psi_2, \partial_1\psi_2, \partial_2\psi_2, \partial_{12}\psi_2\rangle$.

5.4.4 Eigenvalue problem

The equation (5.4.10) is in the same form as in Section 5.2.4 for the two-dimensional elastic body. Therefore the derivation of the eigenvalue problem is identical. Denote the eigenvalue problem for Problem P-1 by Problem P-1E.

Problem P-1E

Find a vector function $\bar{u}$ and a number λ such that

$$M\lambda\bar{u} = K\bar{u}. \qquad (5.4.11)$$

6 Validity of cantilever beam and plate models

6.1 Introduction

Section 4.5 is a discussion of the article [LVV09a]. In this article, the authors compare a cantilever Timoshenko beam to a cantilever two-dimensional beam. In this chapter, the work of the article is extended to investigate the validity of the two-dimensional cantilever beam and a Reisner-Mindlin cantilever plate.

The following is a summary of the work covered in this chapter.

Section 6.2 extends the work of [LVV09a] further to a comparison of a cantilever two-dimensional beam to a cantilever three-dimensional beam. It is an investigation into the validity of the two-dimensional beam model. This extension is suggested by the authors of [LVV09a]. Although a direct comparison between the Timoshenko beam and the three-dimensional beam is proffered, there are complexities involved that makes this comparison difficult. Some of these complexities are discussed in more detail in the numerical results. The authors of [LVV09a] therefore suggest that the two-dimensional beam model is used as an intermediate step to validate the Timoshenko beam.

Section 6.3 extends the work of [LVV09a] further to plate models. This extension follows the same idea as the previous sections, and originates from non-beam type behaviour observed in the three-dimensional beam model in Section 6.2. In this section, a cantilever two-dimensional Reisner-Mindlin plate model is compared to a cantilever three-dimensional plate model. Same as in the other sections, it is an investigation into the validity of the Reisner-Mindlin plate model.

Global parameters and configuration

All the models in this dissertation are assumed to be made of the same isotropic material and have a square cross-section. The parameters are as follows:

- Elastic modulus (G): This is calculated using the formula $G = \frac{E}{2(1+\nu)}$, where E is the modulus of elasticity and ν is Poisson's ratio.
- Shear correction factor (κ^2): This is set to 5/6, which is common for rectangular cross-sections.
- Poisson's ratio (ν): This is set to 0.3, a typical value for materials like steel used in engineering.

These global parameters are used in all the models, and their consistency helps to ensure that any differences in the results can be attributed to the model structures themselves, rather than variations in the material or geometric properties.

Since our models are all dimensionless, all of the beams and plates have a length of $\ell = 1$. We'll use h to describe the height of the beams and plates, and b to describe the width of the beams and plates.

6.2 Validity of a model for a cantilever two-dimensional beam

In Section 4.5, the article [LVV09a] was discussed. In this article, the authors investigated the validity of a cantilever Timoshenko beam by comparing it to a cantilever two-dimensional beam. In this section, the article is extended and the validity of the cantilever two-dimensional model is investigated.

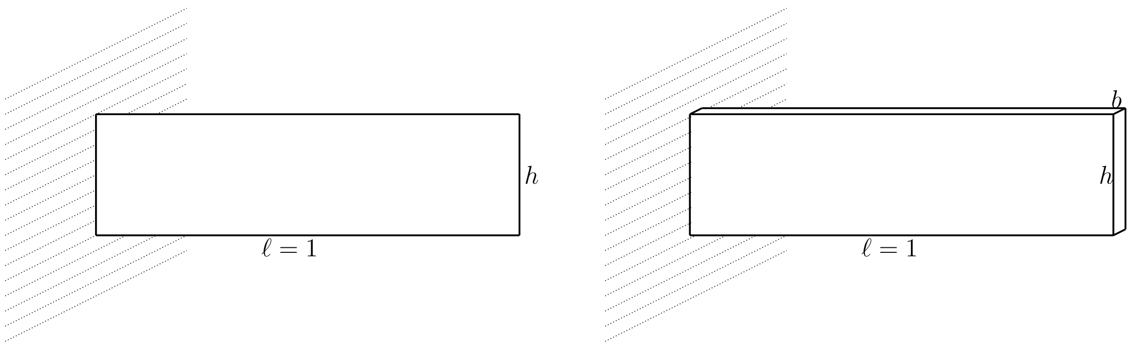
As mentioned in the introduction of the chapter, a beam is a three-dimensional body and therefore a three-dimensional model is more realistic. However the authors of [LVV09a] mention that a direct comparison of the one and three-dimensional models will have complexities and they suggest using the two-dimensional model as an intermediate step.

So in this section, the article [LVV09a] is extended and the validity of a cantilever two-dimensional beam model is investigated, using a cantilever three-dimensional model as a reference.

6.2.1 The models

The two-dimensional model is the same model as used in the previous section, Problem T-2 defined in Section 1.2.3. From Section 1.1.3, the cantilever three-dimensional model, referred to as Problem 3D-1.

Figure 6.1 shows the two beams side-by-side.



(a) Two-Dimensional Elastic Body

(b) Three-Dimensional Elastic Body

Figure 6.1: Side-by-side comparison of the beams.

6.2.2 Calculating the eigenvalues

In Section 5.3, the Finite Element Method for the three-dimensional beam is derived. Similar to the two-dimensional case in 4.5, the Finite Element Method is used to calculate the eigenvalues of the three-dimensional beam. The eigenvalue problem for both models have the same form, but different matrices.

Problem 2D-1E and 3D-1E

Find a real number λ and a function $\bar{u} \in S^h$ such that

$$K\bar{u} \;=\; M\lambda\bar{u}, \tag{6.2.1}$$

where K and M are the standard Finite Element Method matrices defined in Section 5.2.3 for the Problem 2D-1E and Section 5.3.3 for Problem 3D-1E.

Accuracy of the eigenvalues of the three-dimensional model

Figure 6.7 show the rate of convergence of the first 20 eigenvalues of Problem 3D-1E.

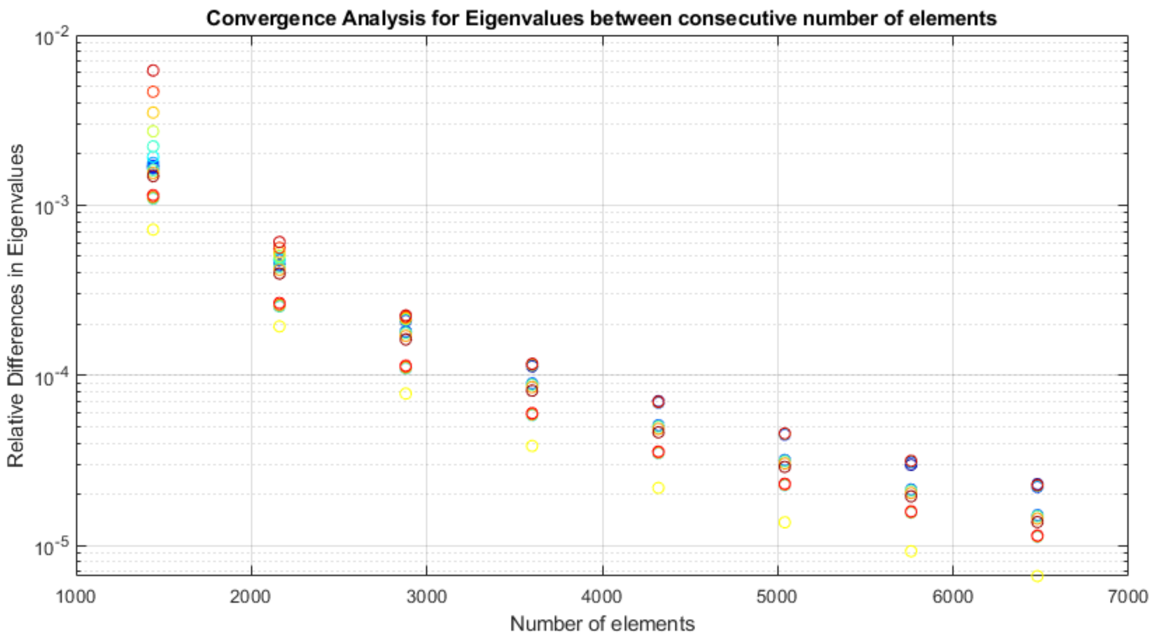


Figure 6.2: Rate of convergence of the first 20 eigenvalues.

Similar to the two-dimensional case, the number of elements can be chosen so that at least the first 20 eigenvalues are accurate to 4 significant digits.

For the three-dimensional model, obtaining this level of accuracy can be difficult and computationally expensive. For the two-dimensional beam in Section 4.5 and the upcoming two-dimensional plate model in Section 6.3 it is easier to get 5 significant digits of accuracy.

6.2.3 Comparing the mode shapes

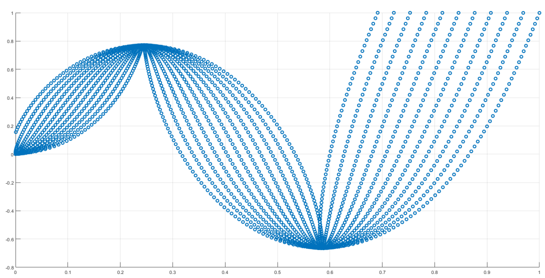
To be able to compare the eigenvalues, the mode shapes of the two models are compared to match up the eigenvalues of the two models. This is the same approach as in Section 4.5.

As seen in Section 4.5, the two-dimensional model as eigenvalues and eigenvectors that are not related to beam type problems. This is also true for the three-dimensional model, and it has even more non-beam type eigenvalues.

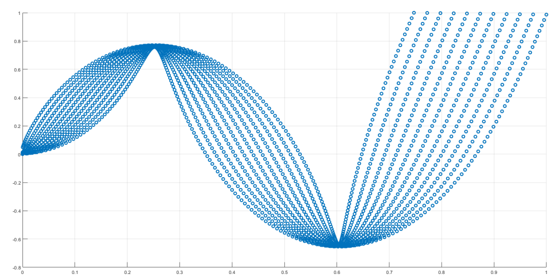
The focus of the investigation remains on beam type problems. Below are some examples of the mode shapes for beam type eigenvalues, mode shapes for non-beam type eigenvalues that are shared between the two and three-dimensional models and also mode shapes for non-beam type eigenvalues that are only present in the three-dimensional model.

Mode shapes relating to beam type eigenvalues.

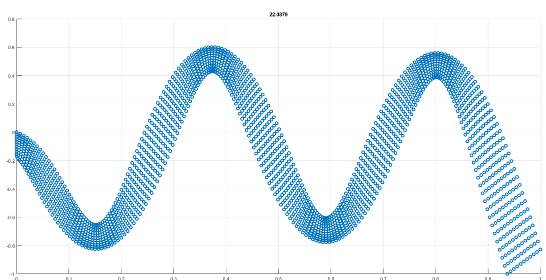
Figure 6.3 show some examples of beam type mode shapes for the displacement u.



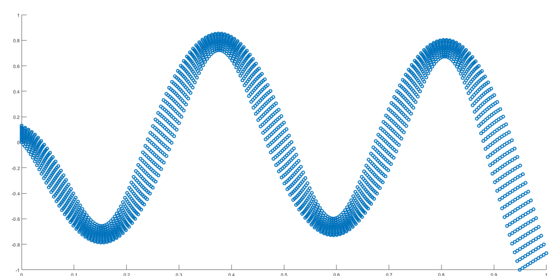
(a) 3D Beam Type - $\lambda_{12} = 2.3293$



(b) 2D Beam Type - $\lambda_3 = 2.3273$



(c) 3D Beam Type - $\lambda_{24} = 21.929$

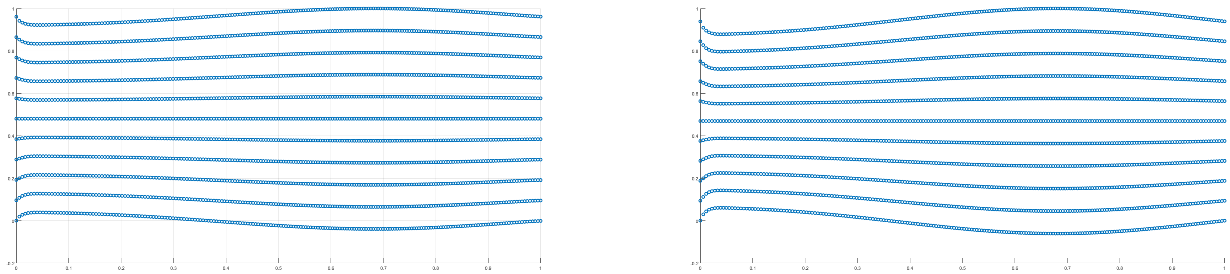


(d) 2D Beam Type - $\lambda_6 = 21.911$

Figure 6.3: Mode shapes of the displacement u with $h = 1/20$.

Mode shapes relating to non-beam type eigenvalues that are present in the two-dimensional model.

Figure 6.4 show examples of mode shapes relating to non-beam type eigenvalues for the displacement u.

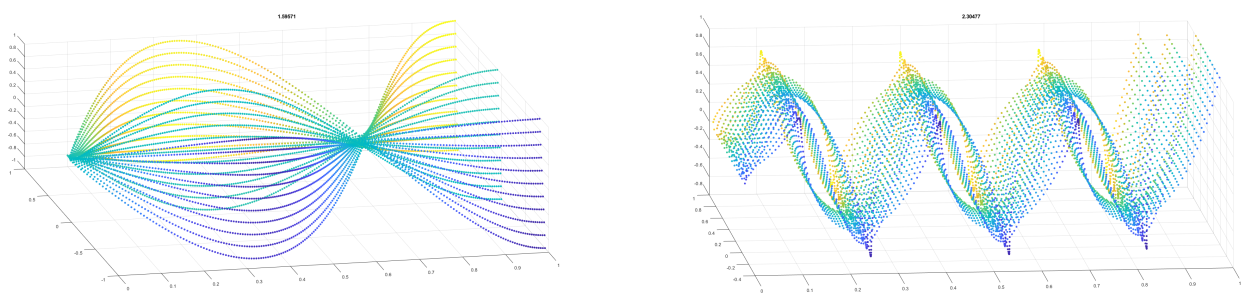


(a) 3D Non-Beam Type - $\lambda_{33} = 69.374$ (b) 2D Non-Beam Type - $\lambda_8 = 69.344$

Figure 6.4: Mode shapes of the displacement u with $h = 1/20$.

Mode shapes relating to non-beam type eigenvalues that are not present in the two-dimensional model.

Figure 6.4 show examples of mode shapes relating to non-beam type eigenvalues for the displacement u which are not present in the two-dimensional model. These mode shapes only appear in the three-dimensional beam.



(a) Non-2D Type - λ_{10} (b) Non-2D Type - λ_{11}

Figure 6.5: Mode shapes of the displacement u with $h = 1/20$.

6.2.4 Comparing the eigenvalues

For a realistic comparison of the models, the parameters need to be chosen carefully. The parameters are h representing the height of the beam and b representing the width of the beam. The two-dimensional model does not have the width parameter.

For h, the values used in Section 4.5 will be used. These values covers a range of beam shapes from a short thick beam, to a long slender beam. two cases are selected that represents realistic cases. For a short and thick beam, consider $h = 1/5$ and for a long and slender beam, consider $h = 1/20$.

For the parameter b, two different cases will be considered. The first case is for $b \leq h$, and the $b > h$. The distinction of these two cases will become apparent in the results. It is important to note that the parameter b will be expressed as a multiple of h.

All of the results will include all the eigenvalues shared between the two-models, including non-beam type eigenvalues. The non-beam type eigenvalues will be highlighted in grey. The non-beam type eigenvalues that are not shared between the two models will be excluded from the results, but the numbering of the eigenvalues will be kept as if they were included.

Case $b \leq h$:

Table 6.1 below compares the eigenvalues of the models for a beam with a small length to height ratio of $h = 1/5$ with decreasing values of b.

Eigenvalues									
i	b = h	i	b = 1/2 h	i	b = 1/4 h	i	b = 1/8 h	j	2D
2	0.12307	2	0.12234	2	0.12198	3	0.12178	1	0.12151
3	3.5773	5	3.5630	6	3.5558	8	3.5519	2	3.5460
5	7.7799	6	7.7596	8	7.7471	11	7.7401	3	7.7311
8	20.334	9	20.283	11	20.26	14	20.247	4	20.225
10	56.247	12	56.173	15	56.156	22	56.142	5	56.109
11	69.197	14	69.319	17	69.281	24	69.238	6	69.164
14	114.03	16	114.01	20	114.05	29	114.06	7	114.03
17	187.01	19	189.14	25	189.37	36	189.34	8	189.17
18	192.21	20	192.41	26	192.58	37	192.63	9	192.61
21	284.76	23	285.43	31	285.74	42	285.84	10	285.85
23	327.57	26	328.24	35	328.37	46	328.40	11	328.40
25	347.77	28	356.44	36	357.30	50	357.33	12	357.08
27	393.69	30	396.84	38	397.28	53	397.37	13	397.33
30	434.46	34	441.05	41	441.81	57	441.99	14	442.00
31	523.65	36	534.04	43	534.17	63	534.03	15	533.71
34	550.51	37	537.82	44	538.86	64	539.06	16	538.97
37	590.86	41	587.43	48	594.17	65	595.58	17	596.06
39	590.86	42	600.52	49	602.25	67	602.69	18	602.77
42	646.21	44	657.22	50	658.04	71	658.06	19	657.87
44	711.07	46	714.62	53	717.10	73	717.51	20	717.37
Max RE: 2.6069%		Max RE: 1.4469%		Max RE: 0.38192%		Max RE: 0.22301%		-	

Table 6.1: Comparsion of Eigenvalues with $h = 1/5$, with decreasing b and $b < h$.

Maximum Relative Error				
	b = h	b = 1/2h	b = 1/4h	b = 1/8h
Beam Type	2.1420 %	0.6804 %	0.38192 %	0.22301 %
Non-Beam Type	2.6069 %	1.4469 %	0.31546 %	0.11680 %

Table 6.2: Maximum relative error for beam type and non-beam type eigenvalues for $h = 1/5$.

Table 6.1 below compares the eigenvalues of the models for a beam with a larger length to height ratio of $h = 1/20$ with decreasing values of b.

Eigenvalues									
i	b = h	i	b = 1/2 h	i	b = 1/4 h	i	b = 1/8 h	j	2D
2	0.008043	2	0.008029	2	0.008023	3	0.00802	1	0.008013
3	0.3087	4	0.30816	5	0.30794	7	0.30785	2	0.30757
5	2.3357	8	2.3316	9	2.3300	12	2.3293	3	2.3273
8	7.7217	10	7.7156	13	7.7124	16	7.7111	4	7.7077
10	8.5387	11	8.5238	14	8.5182	18	8.516	5	8.5086
11	21.986	14	21.948	18	21.934	24	21.929	6	21.911
14	45.863	18	45.781	21	45.756	30	45.746	7	45.712
17	69.444	19	69.408	25	69.385	33	69.374	8	69.344
18	83.149	22	82.999	27	82.960	36	82.944	9	82.887
21	136.44	25	136.19	31	136.14	42	136.12	10	136.03
23	192.62	27	192.62	35	192.58	47	192.56	11	192.48
25	207.87	29	207.5	36	207.43	48	207.41	12	207.29
27	299.14	33	298.63	41	298.56	55	298.53	13	298.38
30	376.68	35	377.01	44	377.01	58	376.98	14	376.83
31	411.58	37	410.89	46	410.83	61	410.81	15	410.63
34	546.15	40	545.27	50	545.24	66	545.23	16	545.03
37	620.77	42	622.02	53	622.19	69	622.18	17	621.95
39	703.55	44	702.49	54	702.29	71	702.53	18	702.31
42	884.27	47	883.02	59	883.14	82	883.19	19	882.96
44	923.68	49	926.88	60	927.43	86	927.50	20	927.18
Max RE: 0.37701%		Max RE: 0.19893%		Max RE: 0.12393%		Max RE: 0.092843%		-	

Table 6.3: Comparsion of Eigenvalues with $h = 1/20$, with decreasing b and $b < h$.

Maximum Relative Error				
	b = h	b = 1/2h	b = 1/4h	b = 1/8h
Beam Type	0.37701 %	0.19893 %	0.12393 %	0.092843 %
Non-Beam Type	0.37682 %	0.10218 %	0.061213 %	0.043601 %

Table 6.4: Maximum relative error for beam type and non-beam type eigenvalues for $h = 1/20$

Tables 6.1 and 6.3 show that the first 20 eigenvalues of the two-dimensional model compares very well to the matching eigenvalues of the three-dimensional beam for $b \leq h$. As the width b decreases, the two-dimensional model becomes a better approximation of the three-dimensional model. Also shown is that the slender beam with $h = 1/20$ compares better than the thick beam where $h = 1/5$, event though the case of $h = 1/5$ is still a very good comparison.

The tables also shows that the three-dimensional model has more non-beam type eigenvalues as the width b decreases, as well as when the width h decreases.This is different from the two-dimensional model as seen in 4.5.

Tables 6.2 and 6.4 breaks up the maximum relative error for the beam type and non-beam type eigenvalues. These tables confirm that the the beam type eigenvalues compare very well.

Case $b > h$:

First, the case is considered where $h = 1/5$.

Eigenvalues							
i	b = 2h	i	b = 4h	i	b = 8h	j	2D
1	0.12474	1	0.12766	1	0.13036	1	0.12151
4	3.6088	4	3.6297	5	3.7766	2	3.5460
6	7.8091	5	7.8389	9	8.9239	3	7.7311
8	20.466	9	20.959	15	21.031	4	20.255
11	56.309	18	58.149	30	60.862	5	56.109
Max RE: 2.6603%		Max RE: 5.0571%		Max RE: 15.428%		-	

Table 6.5: Comparsion of Eigenvalues with $h = 1/5$, with increasing b and $b > h$

Maximum Relative Error			
	b = 2h	b = 4h	b = 8h
Beam Type	2.6603 %	5.0571 %	8.4712 %
Non-Beam Type	1.0096 %	1.3948 %	15.428 %

Table 6.6: Maximum relative error for beam type and non-beam type eigenvalues for $h = 1/5$

In Table 6.5, the height of the beam is set to $h = 1/20$, which was the best case when $b < h$.

Eigenvalues							
i	b = 2h	i	b = 4h	i	b = 8h	j	2D
2	0.008076	1	0.008162	1	0.008324	1	0.008013
3	0.30995	3	0.31298	3	0.31738	2	0.30757
5	2.3462	5	2.3737	6	2.4116	3	2.3273
8	7.7312	8	7.7471	9	7.7711	4	7.7077
10	8.5841	9	8.7082	10	8.7929	5	8.5086
11	22.124	12	22.491	14	23.222	6	21.911
14	46.195	14	47.003	18	47.921	7	45.712
17	69.454	17	69.281	22	72.307	8	69.344
18	83.822	18	85.218	24	80.607	9	82.887
21	137.43	21	138.58	32	140.97	10	136.03
Max RE: 1.1289%		Max RE: 2.8261%		Max RE: 5.9821%		-	

Table 6.7: Comparsion of Eigenvalues with $h = 1/20$, with increasing b and $b > h$

Maximum Relative Error			
	b = 2h	b = 4h	b = 8h
Beam Type	1.1289 %	2.8261 %	5.9821 %
Non-Beam Type	0.30521 %	0.51096 %	4.2734 %

Table 6.8: Maximum relative error for beam type and non-beam type eigenvalues for $h = 1/20$

Tables 6.5 and 6.7 show that the two-dimensional model compares not as well to the three-dimensional model when b is larger than h. Tables 6.6 and 6.8 show that this is also true for the beam type eigenvalues.

For the case of $h = 1/5$, it was also more difficult to obtain reliable eigenvalues using a numerical method.

These results give a detailed overview of the validity of the cantilever two-dimensional beam compared to the cantilever three-dimensional beam. The results show that the two-dimensional beam model compares very well to the three-dimensional beam model for a large range of beam shapes. The results shown were chosen to represent realistic cases, and the results are similar for other cases.

The overall conclusion is that the shape of the beam is very important. If the width b is equal to or less than the height h, the two-dimensional beam model compares very well to the three-dimensional beam model. More so when the beam is slender, and less so when the beam is short and thick.

When the width b is larger than the height h, the two-dimensional beam the comparison degrades very quickly. This is true for both slender and short and thick beams, although with short and thick beams, it is more difficult to obtain reliable eigenvalues using a numerical method.

For practical applications, if $b > h$ the use of a beam model must be brought into question. Other models such as a plate model will be better suited as it is a more realistic model. In the next section, the validity of the Reissner-Mindlin plate mode is investigated using the same method of this section.

6.3 Validity of a model for a cantilever Reissner-Mindlin plate

In the previous section it was shown that for certain applications beam models might not be the appropriate choice. The specific application is for models where the body has a larger width than height. A suggestion would be a plate model.

In this section, the validity of a cantilever Reissner-Mindlin plate model is investigated. This model is a two-dimensional model. So similar to the Timoshenko beam in Section 4.5, it is of value to investigate the validity of the model using a three-dimensional plate model as a reference.

The same method to validate the model will be used as in sections 4.5 and 6.2. A cantilever three-dimensional plate model will be used as a reference, and the eigenvalues and mode shapes of the two models will be compared.

6.3.1 The models

Form Section 1.4.4, the cantilever Reissner-Mindlin plate model is given, and is referred to as Problem P-1. The three-dimensional model is the same as used in the previous section (Section 6.2), and is referred to as Problem 3D-1 defined in Section 1.1.3.

Figure 6.6 shows the two cantilever plates side by side.

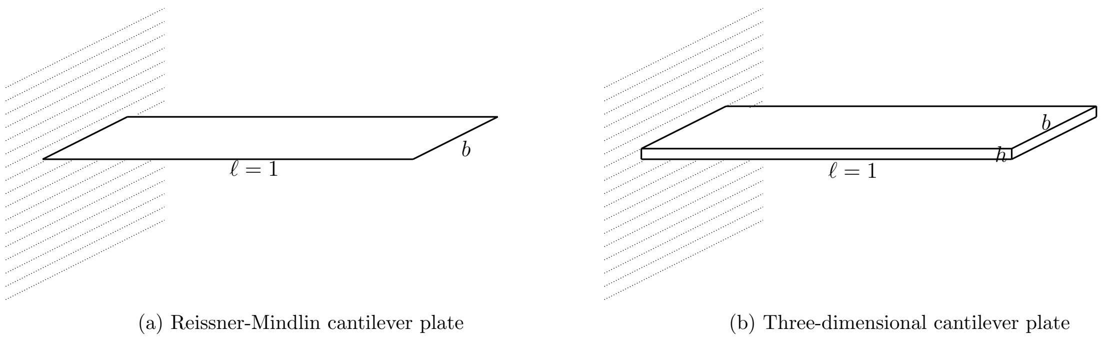


Figure 6.6: Side by side visualization of the cantilever plates.

6.3.2 Calculating the eigenvalues

The eigenvalues for both models are calculated using the Finite Element Method. For the plate model, the Finite Element Method is derived in 5.4 and for the three-dimensional model, the Finite Element Method is derived in 5.3. The eigenvalue problem for both models have the same form, but the matrices are different

Problem 3D-1E and P-1E

Find a vector function u and a number λ such that

$$Ku \quad = \quad M\lambda u, \tag{6.3.1}$$

where K and M are the standard Finite Element Method matrices defined in Section 5.3.3 for Problem 3D-1E and Section 5.4.3 for Problem P-1E.

Accuracy of the eigenvalues

Figure 6.7 show the rate of convergence of the first 20 eigenvalues of Problem 3D-1E.

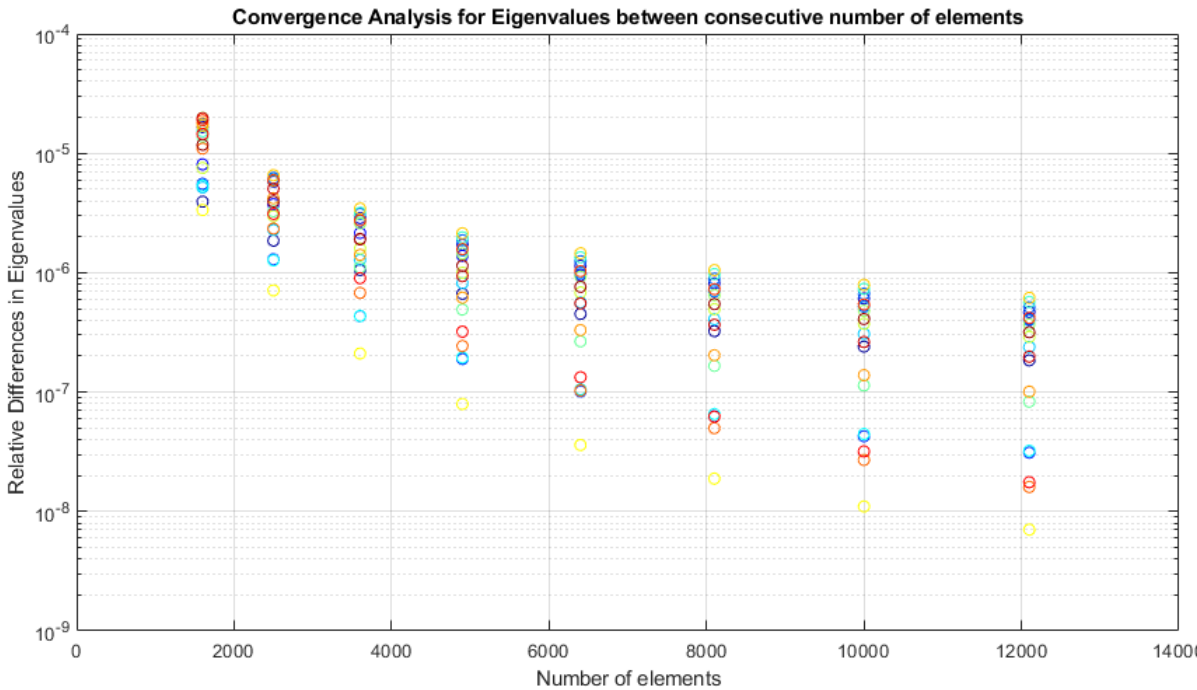


Figure 6.7: Rate of convergence of the first 20 eigenvalues.

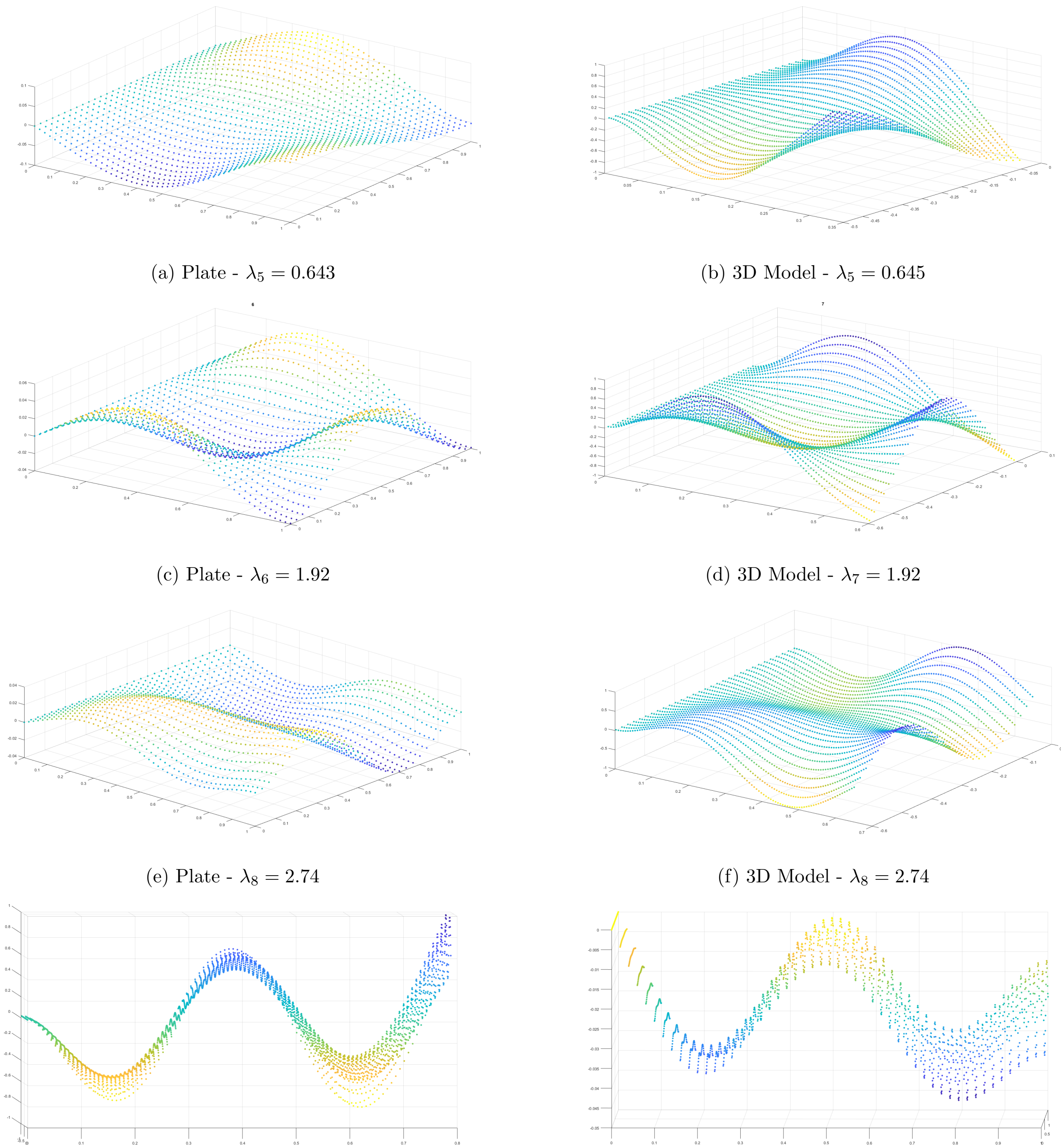
The number of elements can be chosen so that at least the first 20 eigenvalues are accurate to 5 significant digits.

6.3.3 Comparing the mode shapes

Similar to the previous sections (Section 4.5 and Section 6.2), the mode shapes of the two models are compared, to be able to match up the eigenvalues.

Mode shapes relating to plate type eigenvalues

Figure 6.3.3 shows examples of mode shapes relating to plate-type eigenvalues. The plate models has many different shapes.



(a) Plate - $\lambda_5 = 0.643$

(b) 3D Model - $\lambda_5 = 0.645$

(c) Plate - $\lambda_6 = 1.92$

(d) 3D Model - $\lambda_7 = 1.92$

(e) Plate - $\lambda_8 = 2.74$

(f) 3D Model - $\lambda_8 = 2.74$

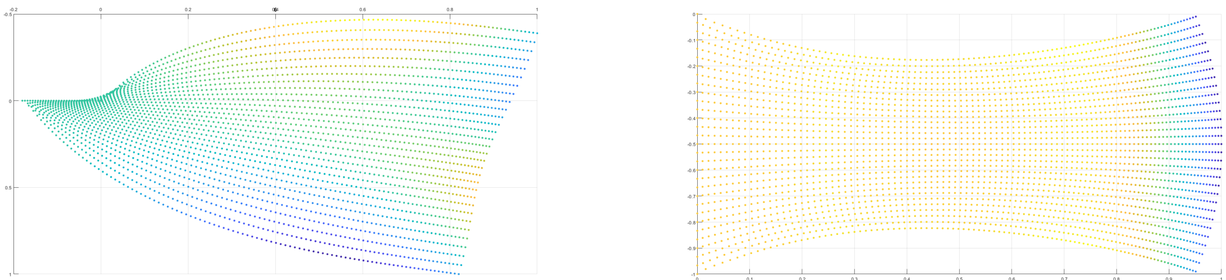
(g) Plate - $\lambda_{12} = 8.96$ (Side view)

(h) 3D Model - $\lambda_{14} = 9.01$ (Side view)

Figure 6.8: Comparison of mode shapes relating to plate-type models. $b = 1/20$, $d = 1$.

Mode shapes relating to non-plate type eigenvalues

Figure 6.3.3 shows examples of mode shapes not relating to plate-type eigenvalues.



(a) 3D Model - $\lambda_6 = 1.35$ (Top view)

(b) 3D Model - $\lambda_{13} = 7.80$ (Top view)

Figure 6.9: Comparison of mode shapes not relating to plate-type models. $b = 1/20$, $d = 1$.

6.3.4 Comparing the eigenvalues

For a realistic comparison of the models, the parameters need to be chosen carefully. Both of the models have a width h and a height b parameter.

Three main cases are considered. These cases are $b = 0.25$ for a narrow plate, $b = 1$ for a plate with equal lenght and width and $b = 1.75$ for a wide plate. For each of the cases, the thickness of the plate is varied.

Remark: Note that the aim of the results is only to investigate the validity of the cantilever Reissner-Mindlin plate model, and not to investigate weather or not a plate model is more suited over beam model when applied to a body that has a larger width than height.

Comparison of Eigenvalues, $b = 0.25$															
$h = 1/5$				$h = 1/10$				$h = 1/20$				$h = 1/30$			
i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate
1	0.12348	1	0.12249	1	0.032327	1	0.032184	1	0.008207	1	0.008186	1	0.003663	1	0.003656
2	0.18638		-	2	0.18531		-	2	0.18476		-	2	0.14204	2	0.14173
3	2.4151	2	2.3954	3	1.161	2	1.1537	3	0.31436	2	0.31341	3	0.18456		-
4	3.5856	3	3.5368	4	1.3155	3	1.31	4	0.44696	3	0.44582	4	0.21514	3	0.21454
5	4.785		-	5	4.7697		-	5	2.3862	4	2.3767	5	1.1001	4	1.0971
6	7.7889		-	6	7.7674	4	7.9816	6	4.1984	5	4.1857	6	2.0374	5	2.0313
7	20.37	4	19.969	7	8.056		-	7	4.7615		-	7	4.1509	6	4.1367
8	21.722	5	21.537	8	12.034	5	11.972	8	7.7543		-	8	4.7585		-
9	25.181		-	9	25.147		-	9	8.7609	6	8.7132	9	6.2078	7	6.1864
10	56.321	6	54.887	10	26.608	6	26.266	10	12.598	7	12.548	10	7.7495		-
11	60.267	7	59.71	11	34.438	7	34.197	11	22.619	8	22.456	11	11.064	8	11.015
12	65.513		-	12	61.829	8	60.801	12	25.128		-	12	13.734	9	13.678
13	69.031		-	13	65.518		-	13	27.297	9	27.155	13	23.854	10	23.724
14	114.09	8	110.7	14	69.133	9	69.548	14	47.123	10	46.695	14	25.121		-
15	117.88	9	116.66	15	70.211		-	15	50.515	11	50.174	15	26.054	11	25.927
16	127.05		-	16	116.78	10	114.43	16	65.523		-	16	36.259	12	36.15
17	184.46	10	185.78	17	121.43	11	119.94	17	69.089		-	17	41.973	13	41.807
18	191.91	11	191.47	18	127.23		-	18	71.563	12	71.166	18	44.95	14	44.685
19	193.03		-	19	178.27	12	175.79	19	81.444	13	80.894	19	45.798	15	45.507
20	193.71		-	20	186.06	13	187.28	20	84.785	14	84.064	20	54.838	16	54.573
Max RE:		2.9764%		Max RE:		2.7585%		Max RE:		0.90845%		Max RE:		0.63525%	

Table 6.9: Comparison of eigenvalues with $b = 0.25$, with decreasing values of h.

Comparison of Eigenvalues, $b = 1$															
$h = 1/5$				$h = 1/10$				$h = 1/20$				$h = 1/30$			
i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate
1	0.12869	1	0.12743	1	0.033784	1	0.033626	1	0.008565	1	0.008543	1	0.003817	1	0.00381
2	0.62189	2	0.61703	2	0.18635	2	0.18562	2	0.049829	2	0.04957	2	0.022511	2	0.022405
3	1.3638		-	3	1.1603	3	1.153	3	0.31428	3	0.31315	3	0.14192	3	0.14156
4	3.5808	3	3.5289	4	1.3579		-	4	0.50842	4	0.50676	4	0.23035	4	0.22969
5	5.8165	4	5.7694	5	1.864	4	1.8577	5	0.64595	5	0.64297	5	0.29502	5	0.29394
6	6.6048	5	6.5216	6	2.2928	5	2.2792	6	1.3555		-	6	0.8928	6	0.88735
7	7.8455		-	7	6.4973	6	6.4546	7	1.9293	6	1.9157	7	1.156	7	1.1527
8	9.8027		-	8	7.8167		-	8	2.5089	7	2.4975	8	1.265	8	1.261
9	17.03	6	16.802	9	8.4533	7	8.3671	9	2.7486	8	2.7369	9	1.3543		-
10	21.225	7	20.743	10	9.3453	8	9.2873	10	3.3087	9	3.2903	10	1.5352	9	1.529
11	23.935	8	23.552	11	9.8009		-	11	5.5365	10	5.4914	11	2.598	10	2.5803
12	24.694		-	12	10.925	9	10.827	12	5.9696	11	5.9246	12	2.8161	11	2.7997
13	27.187	9	26.69	13	17.599	10	17.447	13	7.8024		-	13	4.2654	12	4.2474
14	28.844		-	14	18.596	11	18.407	14	9.0173	12	8.9573	14	4.6674	13	4.6415
15	32.373		-	15	24.729		-	15	9.8006	13	9.838	15	4.9397	14	4.9143
16	41.288	10	40.519	16	27.4	12	27.021	16	9.9172		-	16	5.7243	15	5.6803
17	42.082	11	41.246	17	28.81		-	17	10.365	13	10.286	17	6.6585	16	6.6025
18	51.427		-	18	30.737	13	30.413	18	11.934	14	11.817	18	7.2935	17	7.2474
19	56.963	12	56.1	19	30.83	14	30.413	19	13.915	15	13.767	19	7.7966		-
20	57.741		-	20	32.414			20	15.109	16	14.974	20	9.7996		-
Max RE:		2.2695%		Max RE:		1.3816%		Max RE:		1.0657%		Max RE:		0.84102%	

Table 6.10: Comparison of eigenvalues with $b = 1$, with decreasing values of h.

Comparison of Eigenvalues, $b = 1.75$															
$h = 1/5$				$h = 1/10$				$h = 1/20$				$h = 1/30$			
i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate	i	3D	j	Plate
1	0.13062	1	0.12939	1	0.034245	1	0.034079	1	0.0086697	1	0.0086456	1	0.0038626	1	0.0038541
2	0.32016	2	0.31779	2	0.089872	2	0.089508	2	0.023395	2	0.023323	2	0.010507	2	0.010475
3	1.2496	3	1.2422	3	0.36964	3	0.36841	3	0.098072	3	0.097745	3	0.044239	3	0.044074
4	1.9808		-	4	1.2274	4	1.2188	4	0.33233	4	0.33118	4	0.15003	4	0.14964
5	3.7675	4	3.7086	5	1.352	5	1.3465	5	0.36764	5	0.36652	5	0.16657	5	0.16605
6	4.2021	5	4.1631	6	1.6994	6	1.6894	6	0.4647	6	0.4632	6	0.21059	6	0.21002
7	5.2022	6	5.1381	7	1.9763		-	7	0.79935	7	0.79606	7	0.3656	7	0.36423
8	7.6603	7	7.9399	8	2.8259	7	2.8076	8	1.2445	8	1.2411	8	0.56685	8	0.56529
9	8.0445		-	9	4.4507	8	4.4294	9	1.5708	9	1.563	9	0.72384	9	0.72032
10	9.0791		-	10	5.3923	9	5.3539	10	1.9739		-	10	1.1585	10	1.1545
11	10.135		-	11	7.6371		0	11	2.5137	10	2.5015	11	1.2513	11	1.2466
12	12.888	8	12.736	12	8.4707	10	8.3817	12	2.708	11	2.6947	12	1.4005	12	1.3927
13	14.506	9	14.302	13	9.0612	11	8.9704	13	3.0153	12	2.9984	13	1.4449	13	1.4393
14	14.892		-	14	9.0682		-	14	3.137	13	3.1246	14	1.6806	14	1.6738
15	19.395		-	15	10.018	12	9.9347	15	3.6118	14	3.5928	15	1.973		-
16	21.265	10	20.79	16	10.131		-	16	5.1287	15	5.0977	16	2.4081	15	2.3958
17	22.512	11	22.038	17	10.683	13	10.605	17	5.6368	16	5.6009	17	2.6419	16	2.6253
18	25.193	12	24.775	18	11.806	14	11.682	18	6.5069	17	6.4687	18	3.0386	17	3.0189
19	27.783	13	27.322	19	14.893		-	19	7.6243	18	7.6617	19	3.6563	18	3.6342
20	28.461	14	27.851	20	16.275	15	16.1	20	7.7152		-	20	4.3466	19	4.3267
Max RE:		3.6499%		Max RE:		1.0878%		Max RE:		0.63674%		Max RE:		0.64798%	

Table 6.11: Comparison of eigenvalues with $b = 1.75$, with decreasing values of h.

Overall Tables 6.9, 6.10 and 6.11 show that the cantilever Reissner-Mindlin plate model compares very well to the three-dimensional plate. The maximum relative error is less than 3.65% for all cases considered.

There is less of a dramatic change of the comparison when the width of the plate is changed as with the beam model in Section 6.2. This could be because the Reissner-Mindlin plate model does consider both the width and the height of the plate, while it is only a two-dimensional model. The two-dimensional beam model only considered the height of the beam, and not the width.

Other cases not shown here have also been considered, and the results are similar to the ones shown here. They are therefore not included here, and only worth mentioning.

7 Conclusion

7.1 Overview

Chapter 1

This dissertation is a literature study to investigate the validity of different linear models for beams and plates. The term validity in this dissertation means how well a model compares to a more realistic model for real world applications. The first chapter of the dissertation introduces the models. The simplest model is the Timoshenko model. The other models are a two-dimensional elastic beam model, three-dimensional elastic beam and plate models and a Reissner-Mindlin plate model. The aim in this dissertation is to validate the use of the Timoshenko beam model and the Reissner-Mindlin plate model in applications. These models are simplified one-dimensional beam and two-dimensional plate models. But more realistic models exist, such as a two-dimensional and three-dimensional beam model and a three-dimensional plate model. Using modal analysis (see Chapter 2), it is shown that the solutions of the models can be represented as a linear combination of the modal solutions. Using this idea, it is only required to compare the eigenvalues and eigenfunctions of the models to be able to compare the difference between the solutions. In Chapter 1, the models of the dissertation are given (in dimensionless form). Model problems are then defined by including boundary conditions. These model problems are the models used in the rest of the dissertation. The variational forms are also derived which enables us to see the similarity between the models and the general theory.

Chapter 2

First, a variational form for a model problem of a cantilever Timoshenko beam is given. This model problem in variational form is then extended to complete function spaces, and a weak variational problem is obtained. This weak variational problem is used as an example to explain the main theory of this chapter which is from the article [VV02]. In this article, the authors present a general weak variational form. The weak variational form of all the model problems of this dissertation are special cases of this general weak variational problem. Therefore the assumptions and results can be formulated and applied to all the applications in this dissertation. The article gives four assumptions. Under these assumptions it is shown that the general vibration problem has a unique solution using semi-group theory. The theory is then applied to the example by proving that the assumptions hold.

Finally the concept of modal analysis is introduced, which is fundamental to this dissertation. First the idea of modal analysis is explained by hands of another example, again using a cantilever Timoshenko beam. Then the general case is discussed that follows the article [CVV18]. Given a general vibration problem, it is split into two problems by introducing a trial solution. It can then be verified that these two problems are the eigenvalue problem and an ordinary differential equation. An additional assumption (additional to the assumptions of [VV02]) is introduced by [CVV18]. Under this assumption, the eigenvalue problem has a complete orthonormal sequence of eigen-solutions. The solution of the general vibration problem is an infinite series of modal solutions. This formal series solution is then shown to also be valid for a initial value problem. This general case is also applicable to all the models in this dissertation.

This theory is crucial for this dissertation. It ensures that the solutions of the models will compare well if the eigenvalues and eigenfunctions compare well. Therefore for the comparisons in Chapter 4 and 6, it is only required to compare the eigenvalues and eigenfunctions. The validity of the formal series solution is also important as it ensures that the comparisons remain valid, as long as the models are disturbed in the same way.

Chapter 3

In this chapter, some theory for the Finite Element Method is discussed. This chapter contains two parts, that covers two different theoretical results. The first is on the convergence of the Galerkin approximation of a general vibration problem. This theory is presented in the article [BV13]. The second part concerns the convergence of the eigenvalues and eigenfunctions of a vibration problem when applying the Finite Element Method. This theory is presented in the textbook [SF73].

In the first part, the authors of [BV13] consider the general vibration problem that is studied in Chapter 2. The general vibration problem has a solution as shown in Chapter 2. The Galerkin Approximation is derived from the general vibration problem and is rewritten into a system of ordinary differential using the Finite Element Method. This ordinary differential equation can be proven to have a unique solution. The main results of [BV13] shows that this solution of the Galerkin Approximation converges to the solution of the general vibration problem. The approach of the authors is to calculate the error estimates.

The second part of the chapter considers work done in a textbook [SF73] on eigenvalue problems for elliptic partial differential equations. The specific work discussed, covers the convergence of the eigenvalues and eigenfunctions of a general vibration problem when applying the Finite Element Method. The authors consider a general eigenvalue problem. Then using the Rayleigh-quotient, from the Rayleigh-Ritz method, as well as an approximation theorem from [OR76], the main result is proven. The specific work done in this section is updating the notation of the textbook, as well as expanding some proofs so that the results are easier to understand.

Chapter 4

This chapter is a focus on the main theory of this dissertation, the Timoshenko beam theory. The first section is a discussion of modal analysis applied to the Timoshenko beam theory, and specifically a discussion of the article [VV06]. In this article, the authors present a method to calculate the exact eigenvalues and eigenfunctions of a Timoshenko beam. Starting with a general eigenvalue problem for a Timoshenko beam model, the authors derive a general solution for the ordinary differential equation. The authors then explain the method by hands of an example by applying the method to a cantilever beam model. The next sections of this chapter also then looks at examples of applying the method, first to a cantilever beam model, and then to a pinned-pinned beam model.

Section 4.5 is a discussion of the article [LVV09a]. In this article, the authors investigate the validity of a cantilever Timoshenko beam model, by comparing it to a two-dimensional cantilever Timoshenko beam model. The authors compare the models by comparing the eigenvalues and eigenfunctions. (See Chapter 2 on modal analysis). The two-dimensional model is more complex and there are eigenvalues that are not shared between the models. The authors use the mode shapes to match up the eigenvalues. The eigenvalues matching the eigenvalues of the Timoshenko beam model are referred to by the authors as beam-type eigenvalues. The authors also consider different shapes of beams, from a short thick beam to a long slender beam. The results show that the models compare very well, even for a short and thick beam.

The next section is a discussion on the article [SP06]. In this article, the authors investigate the validity of the Timoshenko beam theory, by comparing the eigenvalues (natural frequencies) for a physical beam, to a Timoshenko beam and a three-dimensional beam using Finite Element Analysis. The authors report on an experiment with forced vibration a free-free beam where the natural frequencies are measured. These empirical results are then compared to the theoretical results. This result, together with the results in chapter 6, give a good picture to the validity of the Timoshenko beam theory.

Chapter 5

In this chapter, the Finite Element Method is applied to cantilever two-dimensional elastic body, cantilever three-dimensional elastic body and a cantilever Reissner-Mindlin plate. The aim of this section is to obtain an algorithm to calculate the eigenvalues and eigenfunctions of the models. The Finite Element Method is not applied to the Timoshenko beam theory, as chapter 4 provides an alternative method. For all the models, the Finite Element Method is applied using bi-cubic or tri-cubic basis functions to improve the rate of convergence and reduce the processing required to obtain accurate results. Each section ends with a eigenvalue problem for the models that can be easily applied to a computer program to calculate the eigenvalues and eigenvectors for the models. This is in preparation of Chapter 6 where the eigenvalues and eigenfunctions are calculated and compared.

Chapter 6

This chapter is an extension of the work of Section 4.5. In Section 4.5, the validity of the Timoshenko beam theory is investigated by comparing a Timoshenko beam model to a two-dimensional beam model.

In real-world applications, a beam is a three-dimensional model. Therefore it should be more realistic to use a three-dimensional model to investigate the validity of the Timoshenko beam theory. This is mentioned by the authors of [LVV09a]. Their suggestion is to use the two-dimensional model as an intermediate step, to avoid complexities. Therefore the validity of the two-dimensional model is investigated, using a three-dimensional beam model as a reference. Again the results show that the comparison relies on the shape of the beams. The two-dimensional model compares well to the three-dimensional beam if the beam is not wide.

If the width of the beam is very large, the use of a beam model can be questioned. A plate model might be more suited. Therefore the last section of this chapter investigates the validity of the Reissner-Mindlin plate model. The Reissner-Mindlin plate model is compared to a three-dimensional plate model. The results show that the Reissner-Mindlin plate model compares well to the three-dimensional plate model.

The same method is used in this chapter as in Section 4.5. The mode shapes are sketched and matched. The corresponding eigenvalues can then be matched up and compared. The eigenvalues relating to the Reissner-Mindlin plate model are referred to as plate-type eigenvalues.

7.2 Contributions

There are four models used in this dissertation. For each of the models, the dimensionless variational form is derived. Also presented are the model problems that are used in this dissertation.

The first result investigates the existence and uniqueness of solutions for general vibration problems. The article that is discussed proves this result for a general vibration problem, using four assumptions. To explain the theory, an example is presented using one of the model problems of the dissertation. The cantilever Timoshenko beam model is chosen for the simple boundary conditions, as well as it's recurring importance in this dissertation. The weak variational form of the model is derived as well as the function spaces defined. This is presented in the same format as the general vibration problem. In fact all the models in this dissertation are special cases of this general vibration problem. The theory is then applied to this example problem, as a demonstration. To apply the theory, the four assumptions are proven to be true.

The next result looks at modal analysis. Before the general case is discussed, again the cantilever Timoshenko beam is used to illustrated the concept of modal analysis.A trial solution to the boundary value problem is suggested. This trial solution is substituted into the partial differential equation and two problems are obtained. The first problem is the eigenvalue problem and the second is an ordinary differential equation. The eigenvalue problem can be solved with theory discussed later in the dissertation and the ordinary differential equation can then be solved. Substitution of these two results into the boundary value problem confirms that the trial solution is correct. The same idea is then discussed for the general case.

We then look at the convergence of the Galerkin Approximation for our general vibration problem. The results of the article discussed are updated with improved notation using a different article. The general case of the Galerkin approximation use a lot of symbols that are not immediately obvious, so again the cantilever Timoshenko beam model problem is used and the Galerkin Approximation is derived in an attempt to explain some of the conventions. The results of the article are then discussed and presented. The results are presented in a concise and practical format to reduce the need to define any unnecessary symbols or notation. The results are also presented in four theorems, summarizing the results of the article that are important but that are not necessarily presented as a theorem in the article.

The next result is on the convergence of the eigenvalues and eigenfunctions of a general vibration problem when using the Finite Element Method. The results are from a textbook. The results are presented with updated notation, coinciding with the notation used in the dissertation. The results are also expanded and extra results are added in an attempt to better explain the theory.

We then look at an important result for the Timoshenko beam theory. This provides a method to calculate the exact eigenvalues and eigenfunctions for the Timoshenko beam theory. Two examples are then used, a cantilever beam and a free-free beam, as an example of the application of the theory. The eigenvalues are calculated and the corresponding mode shapes are plotted. To obtain these results, the equation of motion is plotted, the isolation intervals are determined and the eigenvalues are then calculated using interval division to a desired level of accuracy. The mode shapes can then also be plotted with back substitution. These examples are important preparation for the main comparisons of this dissertation. We also discuss a result comparing the Timoshenko beam theory to results from an empirical study. We add to this article by giving the model problems.

For the rest of the models in this dissertation, the two- and three-dimensional elastic bodies, and the Reissner-Mindlin plate model, a different approach is required to solve the eigenvalue problems. For these models we use the Finite Element Method. For each of the models, their reference configurations are given. All of the models are assumed to have a square cross-section and in a cantilever configuration. This reference configuration is then discretised into a grid of rectangular shaped elements. A set of admissable piecewise Hermite cubic functions are then used and each model is rewritten into a Galerkin Approximation. We the define the standard Finite Element Method Matrices for each case. These are referred to as the mass and stiffness matrices in engineering. Finally our boundary value problems are written into a system of ordinary differential equations in a matrix representation. At this point is it easy to derive the eigenvalue problem for each of the problems in this matrix form. This is in preparation for the main comparisons made in this dissertation.

For our main comparisons, we first look at an article comparing a cantilever Timoshenko beam model to a cantilever two-dimensional model. We discuss the article and replicate the results. The eigenvalues and eigenfunctions of the Timoshenko beam model is obtained using the exact method already described. The eigenvalues and the eigenfunctions for the two-dimensional model are approximated using the Finite Element Method matrix representation of the eigenvalue problem. A MATLAB program is written to approximate the eigenvalues and plot the mode shapes. The accuracy of this approximation is also investigated by looking at the rate of convergence for different grid sizes. Following the article, the eigenvalues are matched up by first comparing the mode shapes of the two models. The eigenvalues can then be matched up and also any eigenvalues not relating to beam-type problems can be filtered out. Based on the results in modal analysis, only a few eigenvalues need to be considered. The relative error between the eigenvalues are then calculated for different shapes of beams. The results improve on the article by showing more significant digits and also including some more results.

We then extend the results of the article to investigate the validity of a cantilever two-dimensional beam model as well as a cantilever Reissner-Mindlin plate model. The same method of the article is followed. To investigate the validity of a cantilever two-dimensional model, a cantilever three-dimensional model is considered. Since both of the models are not beam models, careful consideration needs to be taken to identify the eigenvalues. The same approach is used by comparing the mode shapes. For interest, the non-beam type eigenvalues shared between the two models are also included and the rest that the three-dimensional model does not share with the two-dimensional model are omitted. A clear distinction is made to show the beam type eigenvalues and the beam type results. The shape of the models are also carefully chosen to represent a variety of realistic cases that are interesting. The results show that the shape of the models play an important role in how well the models compare. An interesting result shows that this comparison is not good when the beam gets too wide. We therefore suggest a different model, like a plate model. This lead to the introduction of the Reissner-Mindlin plate model into this dissertation. The validity of a cantilever Reissner-Mindlin plate model is investigated using the same method. The cantilever three-dimensional plate model is again used as the reference model with the restriction that the body is wide.

7.3 Further Research

Future work would include the addition of damping into the models. A lot of the articles do include results for damping, however the results of the modal analysis (the crucial theory of this dissertation) might not be so trivial to prove.

The use of the Hermite cubic basis functions results in the derivatives of the displacement functions to also be available. This brings into question if the stresses of the models can also be compared.

Further improvements to the code can be made. Although a lot of effort was put into optimizing the code, this lead to the code being difficult to understand. Ideally a refactor and simplification of the code, while maintaining its functionality is desired.

Appendix

List of symbols

This list of symbols contain some of the most used symbols in this dissertation. The symbols are grouped into three categories: Vector Spaces and Related Concepts, Mathematical Measures and Operations, and Physical Quantities and Parameters. The first appearance of each symbol is also listed.

Other symbols that are section specific or that are contextually evident might not be in this list.

Vector Spaces and Related Co ncepts

Symbol	Description	First Appearance (Page)
$\mathbb{N}$	set of natural numbers j	57
R^n	n-dimensional space of real numbers	9
Ω	subset of R^n, usually representing a body/reference configuration	9
$\partial\Omega$	the boundary of Ω	13
$\bar{\Omega}$	the boundary of Ω	19
C^n	set of n-times continuously differentiable functions	13
C^∞	space of smooth functions	39
C_0^∞	space of smooth functions with compact support	41
L^n	space of n-power Lebesgue integrable functions	35

H^n	space of functions with weak derivatives up to order n (n'th dimensional Sobolev space)	35
X	global space	36
V	inertia space	36
W	energy space	36
$T(\Omega)$	test function space on Ω	15
S^h	finite dimensional subspace	57
$\mathcal{P}_j$	set of all polynomials of degree at most j	67
E_n	space spanned by the orthonormal basis vectors e_i	95

Mathematical Measures and Operations

Symbol	Description	First Appearance (Page)
$a(\cdot,\cdot), b(\cdot,\cdot), c(\cdot,\cdot)$	bilinear forms	37, 15, 16
$(\cdot,\cdot)_X$	innerproduct of X	36
$\|\| \cdot \|\|_X$	norm in the space X	36
$\partial_x^n f$	n-th partial derivative of f with respect to x	9
$\text{div}X$	divergence of the matrix X	9
$\text{Tr}(X)$	Trace of the matrix X	10
$\det(M)$	determinant of M	69
$\text{span}(\cdot)$	span of a set	52
dV	volume integral measure	18
dS	surface integral measure	18
dA	area integral measure	18
ds	line integral measure	18
$\mathcal{E}(\cdot)$	energy function	48
$R(\cdot)$	Rayleigh quotient	58
Π	interpolation operator	66
$\bar{u}$	another form explicity showing u is a vector	54
RE	abbriviation for the relative error	119

Physical Quantities and Parameters

Symbol	Description	First Appearance (Page)
λ	eigenvalue	46
u or w	displacement vector	9
ϕ	arbitrary vector/rotation of cross-section of Timoshenko beam	13
Q	force per unit volume	9
ρ	density	9
T	stress tensor	9
σ_{ij}	element of the stress tensor T	9
$\mathcal{E}$	infinitesimal strain tensor	10
ε_{ij}	element of the infinitesimal strain tensor $\mathcal{E}$	10
E	Young's modulus	10
ν	Poisson's ratio	10
t	time	11
ℓ	dimension representing length	11
h	dimension representing height	89
b	dimension representing width	100
G	shear modulus of elasticity	11
A	area of a cross-section	20
V	shear force	20
I	area moment of inertia	20
M	moment	20
f^*	dimensionless form of f	11
τ	dimensionless time	11
ξ	dimensionless space	11
α/β	dimensionless constants	21
κ^2	some dimensionless constant/shear correction factor	11
I	identity matrix	11
γ	a dimensionless constant	12
n	a normal vector	13
Σ/Γ	distinct parts of Ω	13
μ	eigenvector	59
e_i	orthonormal basis vector	16

Sobolev spaces

The Space L^2

Consider a measurable space X. The set of square integrable functions is called the L^2 space.

The inner product of L^2 is defined as

$$(f, g) = \int_X fg \qquad \text{for } f, g \in L2.$$

The norm can be defined as $||f|| = (f, f)^{\frac{1}{2}}$ for each $f \in L^2(X)$. For reference, see [Rud53].

The Space L^p

Consider a measurable space X. For a real number $p \geq 1$, the set of p-integrable functions is called the L^p space. A function f belongs to $L^p(X)$ if the p-th power of its absolute value is Lebesgue integrable, that is, if

$$\int_X |f|^p < \infty.$$

The L^p norm (or p-norm) is defined as

$$||f||_p = \left(\int_X |f|^p \right)^{\frac{1}{p}}$$

for each $f \in L^p(X)$.

Continuous function spaces

$C^m(a, b)$ is the space of functions with continuous derivatives up to order m over the open interval (a,b).

$C^m[a, b]$ is the space of functions in $C^m(a, b)$, with existing right derivatives at a and existing left derivatives at b, up to order m.

$C_0^m(a, b)$ contains all functions f in $C^m[a, b]$ with the property that there exists numbers $a < \alpha < \beta < b$ such that f is zero on $[a, \alpha] \cup [\beta, b]$. This property is called compact support.

$C^\infty(a,b)$ contains all functions in $C^m(a,b)$ for all m.

$C^\infty[a,b]$ contains all functions in $C^m[a,b]$ for all m.

$C_0^\infty(a,b)$ contains all functions in $C_0^m(a,b)$ for all m.

First order weak derivative

Suppose $u \in L^2(a,b)$ and there exist a $v \in L^2(a,b)$ such that

$$(u, \phi') = -(v, \phi) \quad \text{for each } \phi \in C_0^\infty(a,b)$$

then v is called the first order weak derivative of u and is denoted by Du.

Higher order weak derivative

Suppose $u \in L^2(a,b)$ and there exist a $v \in L^2(a,b)$ such that

$$(u, \phi^{(m)}) = (-1)^{(m)}(v, \phi) \quad \text{for each } \phi \in C_0^\infty(a,b)$$

then v is called the m'th order weak derivative of u and is denoted by $D^{(m)}u$.

Sobolev spaces

W^n is the space of functions with weak derivatives up to order n. There are also special notation $W^{n,p}$ that indicates that the functions are P-intergrable.

H^n is the space of functions with weak derivatives up to order n and the functions are square integrable. (i.e. $H^n = W^{n,2}$)

MATLAB Code

The following are the main code used in this dissertation to obtain the eigenvalues and eigenvectors of the models. The code and the dissertation is also available on GitHub at `https://github.com/Propagandalf-7/masters`.

The code is optimized for performance, and therefore the presentation of the code is not optimized for readability. The code is also not commented.

Example code for Timoshenko beam model

```
function [u,p,Eig] = TimoshenkoEig(alpha)
syms A;
syms B;
syms C;
syms D;
syms x;
syms m;
syms o;
syms lam;
syms k;
syms a;
syms t;
format long;

%gamma = 0.25;
nu = 0.3;
gamma = 1/(2*(1+nu))*5/6;

delt = 4*gamma/(1+gamma)^2*alpha/lam + (1-gamma)^2/(1+gamma)^2;
omega2 = 1/2*lam*(1+gamma)*(delt^(1/2)+1);
mu2 = 1/2*lam*(1+gamma)*(delt^(1/2)-1);
theta2 = 1/2*lam*(1+gamma)*(1-delt^(1/2));


```

```
u = A*sinh(m*x) + B*cosh(m*x) + C*sin(o*x) + D*cos(o*x);
p = A*((lam+m^2)/m*cosh(m*x)) + B*((lam+m^2)/m*sinh(m*x)) + C
    *(-(lam-o^2)/o*cos(o*x)) + D*((lam-o^2)/o*sin(o*x));

%u = B + C*sin(o*x) + D*cos(o*x);
%p = A + B*a*x + C*(-(lam-o^2)/o)*cos(o*x) + D*((lam-o^2)/o)*
    sin(o*x);

%u = A*sin(t*x) + B*cos(t*x) + C*sin(o*x) + D*cos(o*x);
%p = A*(-(lam-t^2)/t)*cos(t*x) + B*(lam-t^2)/t*sin(t*x) + C*(-(
    lam-o^2)/o)*cos(o*x) + D*(lam-o^2)/o*sin(o*x);


subs((u),x,0);
subs((p),x,0);

u = subs(u,[D,C],[-B,A*(lam+m^2)/m*o/(lam-o^2)]);
p = subs(p,[D,C],[-B,A*(lam+m^2)/m*o/(lam-o^2)]);

%u = subs(u,[B,C],[-D*((lam-o^2)/a),o/lam*(A + k*(-D*((lam-o^2)
    /a) +D))]);
%p = subs(p,[B,C],[-D*((lam-o^2)/a),o/lam*(A + k*(-D*((lam-o^2)
    /a) +D))]);

%u = subs(u,[D,C],[-B,-o/(o^2-lam)*(t^2-lam)/t*A]);
%p = subs(p,[D,C],[-B,-o/(o^2-lam)*(t^2-lam)/t*A]);

M1 = (subs(diff(p),x,1));
M2 = (subs(diff(u) - p,x,1));
M = [subs(M1,[A,B],[1,0]) subs(M1,[A,B],[0,1]);subs(M2,[A,B
    ],[1,0]) subs(M2,[A,B],[0,1])];

%latex(simplify(det(M)))

L = subs(M,k,sqrt(5/6));
L = subs(L,o,(omega2)^(1/2));
L = subs(L,m,(mu2)^(1/2));
L = subs(L,t,(theta2)^(1/2));

Y = det(L);
Y = simplify(subs(Y,lam,x));

%ezplot(Y,[0,300])
%grid on

R = 0;
R = FindRoots(Y,0.001,500,0.1)
%R = FindRoots(Y,100,200,0.1)
```

```
RF = zeros(1,size(R,2));
RF2 = zeros(1,size(R,2));
RF3 = zeros(1,size(R,2));
RF4 = zeros(1,size(R,2));
for i = 1:size(R,2)
    if(R(i)-0.1>0)
        RF(i) = FindRoots(Y,R(i)-0.1,R(i)+0.1,0.0001);
    else
        RF(i) = FindRoots(Y,0.0001,R(i)+0.1,0.0001);
    end
end
for i = 1:size(R,2)
    if(RF(i)-0.0001>0)
        RF2(i) = FindRoots(Y,RF(i)-0.0001,RF(i)+0.0001,0.00001);
    else
        RF2(i) = FindRoots(Y,0.0001,RF(i)+0.0001,0.00001);
    end
end
for i = 1:size(R,2)
    if(RF2(i)-0.00001>0)
        RF3(i) = FindRoots(Y,RF2(i)-0.00001,RF2(i)+0.00001,0.000001);
    else
        RF3(i) = FindRoots(Y,0.00001,RF2(i)+0.00001,0.000001);
    end
end
%for i = 1:size(R,2)
 %    if(RF3(i)-0.00001>0)
 %        RF4(i) = FindRoots(Y,RF3(i)-0.000001,RF3(i)+0.000001,0.0000001);
 %    else
%         RF4(i) = FindRoots(Y,0.000001,RF3(i)+0.000001,0.0000001);
%     end
%end

Eig = RF3';
%ModeNum = 1;
Eig
%{
imageDir = fullfile(cd, 'images');
if ~exist(imageDir, 'dir')
   mkdir(imageDir);
end

for i = 1:size(Eig,1)
    % Get values
```

```
    LS = subs(L,lam,RF(i));
    [a,L1] = gauss(LS,[0;0]);
    B1 = double(-L1(1,1)/L1(1,2))
    us = subs(u,[o,m],[(omega2)^(1/2),(mu2)^(1/2)]);
    ps = subs(p,[o,m],[(omega2)^(1/2),(mu2)^(1/2)]);
    us = simplify(subs(us,[lam,A,B,k],[RF(i),1,B1,sqrt(5/6)]));
    ps = simplify(subs(ps,[lam,A,B,k],[RF(i),1,B1,sqrt(5/6)]));

    xd = 0:0.01:1;
    uss = subs(us,x,xd);
    max = norm(uss,Inf);
    us = us/max;

    pss = subs(ps,x,xd);
    maxp = norm(pss,Inf);
    ps = ps/maxp;

    % Displacement
    f1 = figure('Name', ['Mode ' num2str(i) ' Displacement']);
    clf(f1)
    ezplot(us,[0,1])
    title(['Mode ' num2str(i) ' Displacement'])
    xlabel('x (Position)')
    ylabel('Displacement (Normalized)')
    legend('Displacement', 'Location', 'best')

    % Stress
    f2 = figure('Name', ['Mode ' num2str(i) ' Stress']);
    clf(f2)
    ezplot(ps,[0,1])
    title(['Mode ' num2str(i) ' Stress Distribution'])
    xlabel('x (Position)')
    ylabel('Stress (Normalized)')
    legend('Stress Distribution', 'Location', 'best')

    % Both
    f3 = figure('Name', ['Mode ' num2str(i) ' Displacement and Stress']);
    clf(f3)
    hold on
    ezplot(us,[0,1])
    ezplot(ps,[0,1])
    title(['Mode ' num2str(i) ' Displacement and Stress'])
    xlabel('x (Position)')
    legend('Displacement', 'Stress Distribution', 'Location', 'best')
    hold off

```

```
    saveas(f1, fullfile(imageDir, ['Mode_' num2str(i) '_Displacement.png']));
    saveas(f2, fullfile(imageDir, ['Mode_' num2str(i) '_Stress.png']));
    saveas(f3, fullfile(imageDir, ['Mode_' num2str(i) '_Displacement_and_Stress.png']));
end
writeToExcel(Eig, imageDir);
%}
return;

function writeToExcel(Eig, imageDir)
    % Define the name of the Excel file
    excelFileName = 'TimoshenkoResults.xlsx';

    % Initialize COM server
    Excel = actxserver('Excel.Application');
    Excel.Workbooks.Add;

    % Get active sheet
    WorkSheets = Excel.ActiveWorkBook.Sheets;
    sheet1 = WorkSheets.get('Item', 1);
    sheet1.Activate;

    % Start writing data to Excel
    sheet1.Range('A1').Value = 'Mode Number';
    sheet1.Range('B1').Value = 'Eigen Value';

    for i = 1:size(Eig, 1)
        sheet1.Range(['A' num2str(i + 1)]).Value = i; % Mode Number
        sheet1.Range(['B' num2str(i + 1)]).Value = Eig(i); % Eigen Value

        % Insert images
        pic_path = fullfile(imageDir, ['Mode_' num2str(i) '_Displacement.png']);
        disp(['Image path: ', pic_path]);
        Excel.ActiveSheet.Shapes.AddPicture(pic_path, 0, 1, 100, i*100, 200, 200);
        pic_path = fullfile(imageDir, ['Mode_' num2str(i) '_Stress.png']);
        disp(['Image path: ', pic_path]);
        Excel.ActiveSheet.Shapes.AddPicture(pic_path, 0, 1, 100, i*100, 200, 200);
        pic_path = fullfile(imageDir, ['Mode_' num2str(i) '_Displacement_and_Stress.png']);
        disp(['Image path: ', pic_path]);
```

```
        Excel.ActiveSheet.Shapes.AddPicture(pic_path, 0, 1, 100, i*100, 200, 200);
    end

    % Save and close the Excel file
    Excel.ActiveWorkBook.SaveAs(excelFileName);
    pause(1); % waits for 1 second

    Excel.ActiveWorkbook.Close;
    Excel.Quit;
    Excel.delete;
    clear Excel;

function I = IntervalDivision(a,b,TOL)
if abs(b-a)>=TOL
   m = abs(b-a)/2;
   I = [IntervalDivision(a,a+m,TOL); IntervalDivision(a+m,b,TOL)];
   return;
else
    I = [a b];
end
return

function R = FindRoots(Y,a,b,TOL)
syms x;
I = IntervalDivision(a,b,TOL);
n = size(I,1);
SubsI = zeros(size(I));
for i = 1:n
    SubsI(i,1) = subs(Y,x,I(i,1));
    SubsI(i,2) = subs(Y,x,I(i,2));
end

icount = 1;
for i = 1:n
   if SubsI(i,1) == 0
       R(icount) = I(i,1);
       icount = icount+1;
   elseif SubsI(i,1) == 0
       R(icount) = I(i,1);
       icount = icount+1;
   elseif SubsI(i,1)*SubsI(i,2) < 0
       R(icount) = (I(i,2)+I(i,1))/2;
       icount = icount+1;
   end
end
return
```

Example code for two-dimensional elastic body using bi-cubics

```
function [E, n, m] = TwoDimensionalCantileverCubic(n,alpha,
    graph)
format long g
%gpuDevice(2)
beta = 1;
%alpha = 300;
gamma = 0.3205;
nu = 0.3;
iA = 1/(1-nu^2);
iB = 1/(2*gamma*(1+nu));

ex = sqrt(12/alpha);
h=ex
%ex = h
m = ceil(n*ex);

if(m <= 1)
    m = 2;
end
%if(m >= 15)
%    m = 15;
%end
%n
%m = 2

a = 0;
b = 1;
c = 0;

%d = sqrt(12/alpha);
d =h
deltx = (b-a)/n;
delty = (d-c)/m;

[MM,Kxx,Kxy,Kyy,D0] = CalMatrix(n,m,deltx,delty);
Kyx = Kxy';%CHECKED
All = (n+1)*(m+1);

K1 = Kxx + (1-nu)/2*Kyy;
K2 = nu*Kyx + (1-nu)/2*Kxy;
```

```
K3 = nu*Kxy + (1-nu)/2*Kyx;
K4 = Kyy + (1-nu)/2*Kxx;
O = sparse(size(MM,1),size(MM,2));%CHECKED
MMu = [MM O;%CHECKED
       O MM];%CHECKED
Mf = MMu;
K = 1/(gamma*(1-nu^2))*[K1 K2; K3 K4];%CHECKED
x = [7*All:-1:7*All-(m+1)+1 5*All:-1:5*All-(m+1)+1 3*All:-1:3*
    All-(m+1)+1 1*All:-1:1*All-(m+1)+1];
K(x,:) = [];
K(:,x) = [];
MMu(x,:) = [];
MMu(:,x) = [];
Mf(x,:) = [];
%CHECKED
%eig(Mu,K)
[V,D] = eigs(K,MMu,20,'sm');
E = diag(D);
size(K)

if (graph == 1)
    for i = 1:10
        w = V(:,i);

        f = -1/200;
        F1 = zeros((n+1)*(m+1),1);
        F1(ceil((1+(m+1))/2)) = f;
        F = zeros(8*(n+1)*(m+1),1);
        F(4*(n+1)*(m+1)+1:5*(n+1)*(m+1)) = F1;


        %ueq = K\MMu*(-w);
        tic
        %Kg = gpuArray(K);
        %Mfg = gpuArray(Mf);
        %Fg = gpuArray(F);
        toc

        %b =Mf*(-F);
        %tic
        %ueq = K\Mf*(-F);
        %toc
        b = MMu*(-w);
        tic
        tol = 0.00001;
        maxit = 30000;
        alpha1 = max(sum(abs(K),2)./diag(K))-2;
```

```
        L = ichol(K,struct('type','ict','droptol',1e-3,'diagcomp',alpha1));
        ueq = pcg(K,b,tol,maxit,L,L');
        toc

        Ep = Positions(m,n,deltx,delty);
        %ux = 0;
        %uy = 0;
        ux = [ueq(1:(n+1)*(m+1)-(m+1),1);zeros(m+1,1)] + Ep(:,1);
        dxux = [ueq((n+1)*(m+1)-(m+1)+1:2*(n+1)*(m+1)-(m+1),1)];
        dyux = [ueq(2*(n+1)*(m+1)-(m+1)+1:3*(n+1)*(m+1)-2*(m+1),1);zeros(m+1,1)];
        dxyux = [ueq(3*(n+1)*(m+1)-2*(m+1)+1:4*(n+1)*(m+1)-2*(m+1),1)];
        uy = [ueq(4*(n+1)*(m+1)-2*(m+1)+1:5*(n+1)*(m+1)-3*(m+1),1);zeros(m+1,1)] + Ep(:,2);
        dxuy = [ueq(5*(n+1)*(m+1)-3*(m+1)+1:6*(n+1)*(m+1)-3*(m+1),1)];
        dyuy = [ueq(6*(n+1)*(m+1)-3*(m+1)+1:7*(n+1)*(m+1)-4*(m+1),1);zeros(m+1,1)];
        dxyuy = [ueq(7*(n+1)*(m+1)-4*(m+1)+1:8*(n+1)*(m+1)-4*(m+1),1)];

        ux = flip(ux);
        dxux = flip(dxux);
        dyux = flip(dyux);
        dxyux = flip(dxyux);
        uy = flip(uy);
        dxuy = flip(dxuy);
        dyuy = flip(dyuy);
        dxyuy = flip(dxyuy);

        uxB = ux(D0(ceil((m+1)/2),:));
        uyB = uy(D0(ceil((m+1)/2),:));
        dxuxB = dxux(D0(ceil((m+1)/2),:));
        dxuyB = dxuy(D0(ceil((m+1)/2),:));
        dyuxB = dyux(D0(ceil((m+1)/2),:));
        dyuyB = dyuy(D0(ceil((m+1)/2),:));
        dxyuxB = dxyux(D0(ceil((m+1)/2),:));
        dxyuyB = dxyuy(D0(ceil((m+1)/2),:));

        maxs = norm(uy,Inf);
        uy = uy/maxs;
        stress = ceil((n+1)/2);
        figure(i);

```

```
        scatter(ux,uy, 'b', 'filled'); % blue filled circles
        hold on;

        middleIndex = ceil(size(D0,1)/2);  % Find the middle row of D0
        ux1 = ux(D0(middleIndex,:));
        uy1 = uy(D0(middleIndex,:));
        maxs2 =  norm(uy1,Inf);

        plot(ux1,uy1, 'r', 'LineWidth', 2); % red line with thicker width

        % Add titles, labels, and legends
        title(['Eigenfunction ' num2str(i)]);
        xlabel('Ux');
        ylabel('Uy');
        legend('Ux vs. Uy', 'Transformed Ux vs. Uy', 'Location', 'best');

        grid on; % Add a grid for better readability
        sigma11 = 1/(gamma*(1-nu^2))*(dxuxB + nu*dyuyB);
        sigma22 = 1/(gamma*(1-nu^2))*(dyuyB + nu*dxuxB);
        sigma12 = 1/(2*gamma*(1+nu))*(dyuxB + dxuyB);

        T = [sigma11(stress) sigma12(stress);sigma12(stress) sigma22(stress)]
    end
end
return;

function [Mq,Kxxq,Kxyq,Kyyq] = matrix(deltx,delty)%CHECKED
syms x;
syms y;

Q = [1 x x^2 x^3 y x*y x^2*y x^3*y y^2 x*y^2 x^2*y^2 x^3*y^2 y^3 x*y^3 x^2*y^3 x^3*y^3];

size_num = size(Q,2)/4;
T = MATRIX_T(Q);

Mq = zeros(size(Q,2))*x*y;
Kxxq = zeros(size(Q,2))*x*y;
Kxyq = zeros(size(Q,2))*x*y;
Kyyq = zeros(size(Q,2))*x*y;

for i = 1:size(Q,2)
    for j = 1:size(Q,2)
        Mq(j,i) = Q(j)*Q(i);
```

```
        Kxxq(j,i) = diff(Q(j),x)*diff(Q(i),x);
        Kxyq(j,i) = diff(Q(j),y)*diff(Q(i),x);
        Kyyq(j,i) = diff(Q(j),y)*diff(Q(i),y);
    end
end

Mq = int(int(Mq,x,0,1),y,0,1);
Kxxq = int(int(Kxxq,x,0,1),y,0,1);
Kxyq = int(int(Kxyq,x,0,1),y,0,1);
Kyyq = int(int(Kyyq,x,0,1),y,0,1);

IT = inv(T);

Mq = (IT)'*Mq*IT;
Kxxq = (IT)'*Kxxq*IT;
Kxyq = (IT)'*Kxyq*IT;
Kyyq = (IT)'*Kyyq*IT;

Mq = double(Mq*deltx*delty);
Kxxq = double(Kxxq*delty/deltx);
Kyyq = double(Kyyq*deltx/delty);
Kxyq = double(Kxyq);
return;

function [Adj, Type, D] = Domain(n,m)
D = zeros(m+1,n+1);
icount = 1;
for i = n+1:-1:1
   for j = 1:m+1
       D(j,i) = icount;
       icount = icount + 1;
   end
end
D0 = [zeros(1,n+3);zeros(m+1,1) D zeros(m+1,1);zeros(1,n+3)];
icount = 1;
Adj = zeros((n+1)*(m+1),9);
for i = n+2:-1:2
   for j = 2:m+2
   Adj(icount,1) = D0(j,i); %middel
   Adj(icount,2) = D0(j-1,i); %bo
   Adj(icount,3) = D0(j-1,i+1); %regsbo
   Adj(icount,4) = D0(j,i+1); %regs
   Adj(icount,5) = D0(j+1,i+1); %regs onder
   Adj(icount,6) = D0(j+1,i); %onder
   Adj(icount,7) = D0(j+1,i-1); %links onder
   Adj(icount,8) = D0(j,i-1); %links
   Adj(icount,9) = D0(j-1,i-1); %linksbo
   icount = icount +1;
```

```
    end
end
Type= zeros((n+1)*(m+1),2);
T = [1      0      0      0      0      2      5      4      0;
     2      1      0      0      0      3      6      5      4;
     3      2      0      0      0      0      0      6      5;
     4      0      0      1      2      5      8      7      0;
     5      4      1      2      3      6      9      8      7;
     6      5      2      3      0      0      0      9      8;
     7      0      0      4      5      8      0      0      0;
     8      7      4      5      6      9      0      0      0;
     9      8      5      6      0      0      0      0      0];
 nnz(T);

for i = 1:(n+1)*(m+1)
   Type(i,1) = Adj(i,1);
   for j = 1:9
       bflag = true;
       for k = 1:9
           if(any(T(j,k)) ~= any(Adj(i,k)))
               bflag = false;
           end
       end
       if(bflag == true)
          Type(i,2) = j;
       end
   end
end
return

function [M,Kxx,Kxy,Kyy,D0] = CalMatrix(n,m,deltx,delty)
[Adj, Type, D0] = Domain(n,m);
[Mq,Kxxq,Kxyq,Kyyq] = matrix(deltx,delty);

ns = nnz(Adj);
Ms = zeros(16*ns,1);
Kxxs = zeros(16*ns,1);
Kxys = zeros(16*ns,1);
Kyys = zeros(16*ns,1);

x = [1:(n+1)*(m+1)]';
c = sum(Adj~=0,2);

ix = repelem(x,c);
x = [ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);];
```

```

iy = nonzeros(Adj');
y = [iy;iy;iy;iy;
    iy+(n+1)*(m+1);iy+(n+1)*(m+1);iy+(n+1)*(m+1);iy+(n+1)*(m+1);
    iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);
    iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1)];
B = BMatrix();

Adj(Adj == 0) = nan;
NanAdj = ~isnan(Adj);
NanAdj = NanAdj';

a = 1:9;
b = repelem(a,size(Adj,1),1);
b = b';
iz = b(NanAdj);

for i = 1:ns
  k = 1;
  while(B(Type(ix(i),2),iz(i),k) ~= 0)
    Ms(i)   = Ms(i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kxxs(i) = Kxxs(i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kxys(i) = Kxys(i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kyys(i) = Kyys(i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));

    Ms(ns+i)   = Ms(ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kxxs(ns+i) = Kxxs(ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kxys(ns+i) = Kxys(ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kyys(ns+i) = Kyys(ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));

    Ms(2*ns+i)   = Ms(2*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1));
    Kxxs(2*ns+i) = Kxxs(2*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1));
    Kxys(2*ns+i) = Kxys(2*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1));
```

```
    Kyys(2*ns+i) = Kyys(2*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1));

    Ms(3*ns+i)   = Ms(3*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1));
    Kxxs(3*ns+i) = Kxxs(3*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1));
    Kxys(3*ns+i) = Kxys(3*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1));
    Kyys(3*ns+i) = Kyys(3*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(4*ns+i)   = Ms(4*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(4*ns+i) = Kxxs(4*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(4*ns+i) = Kxys(4*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(4*ns+i) = Kyys(4*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+4);

    Ms(5*ns+i)   = Ms(5*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(5*ns+i) = Kxxs(5*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(5*ns+i) = Kxys(5*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(5*ns+i) = Kyys(5*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+4);

    Ms(6*ns+i)   = Ms(6*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(6*ns+i) = Kxxs(6*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(6*ns+i) = Kxys(6*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(6*ns+i) = Kyys(6*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);

    Ms(7*ns+i)   = Ms(7*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(7*ns+i) = Kxxs(7*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(7*ns+i) = Kxys(7*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(7*ns+i) = Kyys(7*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(8*ns+i)   = Ms(8*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(8*ns+i) = Kxxs(8*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(8*ns+i) = Kxys(8*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(8*ns+i) = Kyys(8*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(9*ns+i)   = Ms(9*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(9*ns+i) = Kxxs(9*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(9*ns+i) = Kxys(9*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(9*ns+i) = Kyys(9*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(10*ns+i)   = Ms(10*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(10*ns+i) = Kxxs(10*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(10*ns+i) = Kxys(10*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(10*ns+i) = Kyys(10*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(11*ns+i)   = Ms(11*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(11*ns+i) = Kxxs(11*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(11*ns+i) = Kxys(11*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(11*ns+i) = Kyys(11*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(12*ns+i)   = Ms(12*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(12*ns+i) = Kxxs(12*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(12*ns+i) = Kxys(12*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(12*ns+i) = Kyys(12*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
```

```
    Ms(13*ns+i)   = Ms(13*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)
   +4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(13*ns+i) = Kxxs(13*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),
   k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(13*ns+i) = Kxys(13*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),
   k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(13*ns+i) = Kyys(13*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),
   k)+4,B(Type(ix(i),2),iz(i),k+1)+12);

    Ms(14*ns+i)   = Ms(14*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)
   +8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(14*ns+i) = Kxxs(14*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),
   k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(14*ns+i) = Kxys(14*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),
   k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(14*ns+i) = Kyys(14*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),
   k)+8,B(Type(ix(i),2),iz(i),k+1)+12);

    Ms(15*ns+i)   = Ms(15*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(15*ns+i) = Kxxs(15*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),
   k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(15*ns+i) = Kxys(15*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),
   k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(15*ns+i) = Kyys(15*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),
   k)+12,B(Type(ix(i),2),iz(i),k+1)+12);

    k = k + 2;
    if(k > 8)
       break;
    end
  end
end
M   = sparse(x,y,Ms,4*(n+1)*(m+1),4*(n+1)*(m+1));
Kxx = sparse(x,y,Kxxs,4*(n+1)*(m+1),4*(n+1)*(m+1));
Kxy = sparse(x,y,Kxys,4*(n+1)*(m+1),4*(n+1)*(m+1));
Kyy = sparse(x,y,Kyys,4*(n+1)*(m+1),4*(n+1)*(m+1));
return;

function B = BMatrix()
B = zeros(9,9,8);
B(3,1,:) = [2 2 0 0 0 0 0 0];
B(3,2,:) = [2 3 0 0 0 0 0 0];
B(3,3,:) = [0 0 0 0 0 0 0 0];
B(3,4,:) = [0 0 0 0 0 0 0 0];
B(3,5,:) = [0 0 0 0 0 0 0 0];
B(3,6,:) = [0 0 0 0 0 0 0 0];
B(3,7,:) = [0 0 0 0 0 0 0 0];
```

```
B(3,8,:) = [2 1 0 0 0 0 0 0];
B(3,9,:) = [2 4 0 0 0 0 0 0];

B(2,1,:) = [2 2 3 3 0 0 0 0];
B(2,2,:) = [2 3 0 0 0 0 0 0];
B(2,3,:) = [0 0 0 0 0 0 0 0];
B(2,4,:) = [0 0 0 0 0 0 0 0];
B(2,5,:) = [0 0 0 0 0 0 0 0];
B(2,6,:) = [3 2 0 0 0 0 0 0];
B(2,7,:) = [3 1 0 0 0 0 0 0];
B(2,8,:) = [2 1 3 4 0 0 0 0];
B(2,9,:) = [2 4 0 0 0 0 0 0];

B(1,1,:) = [3 3 0 0 0 0 0 0];
B(1,2,:) = [0 0 0 0 0 0 0 0];
B(1,3,:) = [0 0 0 0 0 0 0 0];
B(1,4,:) = [0 0 0 0 0 0 0 0];
B(1,5,:) = [0 0 0 0 0 0 0 0];
B(1,6,:) = [3 2 0 0 0 0 0 0];
B(1,7,:) = [3 1 0 0 0 0 0 0];
B(1,8,:) = [3 4 0 0 0 0 0 0];
B(1,9,:) = [0 0 0 0 0 0 0 0];

B(6,1,:) = [1 1 2 2 0 0 0 0];
B(6,2,:) = [1 4 2 3 0 0 0 0];
B(6,3,:) = [1 3 0 0 0 0 0 0];
B(6,4,:) = [1 2 0 0 0 0 0 0];
B(6,5,:) = [0 0 0 0 0 0 0 0];
B(6,6,:) = [0 0 0 0 0 0 0 0];
B(6,7,:) = [0 0 0 0 0 0 0 0];
B(6,8,:) = [2 1 0 0 0 0 0 0];
B(6,9,:) = [2 4 0 0 0 0 0 0];

B(5,1,:) = [1 1 2 2 3 3 4 4];
B(5,2,:) = [1 4 2 3 0 0 0 0];
B(5,3,:) = [1 3 0 0 0 0 0 0];
B(5,4,:) = [1 2 4 3 0 0 0 0];
B(5,5,:) = [4 2 0 0 0 0 0 0];
B(5,6,:) = [4 1 3 2 0 0 0 0];
B(5,7,:) = [3 1 0 0 0 0 0 0];
B(5,8,:) = [3 4 2 1 0 0 0 0];
B(5,9,:) = [2 4 0 0 0 0 0 0];

B(4,1,:) = [3 3 4 4 0 0 0 0];
B(4,2,:) = [0 0 0 0 0 0 0 0];
B(4,3,:) = [0 0 0 0 0 0 0 0];
B(4,4,:) = [4 3 0 0 0 0 0 0];
B(4,5,:) = [4 2 0 0 0 0 0 0];
```

```
B(4,6,:) = [4 1 3 2 0 0 0 0];
B(4,7,:) = [3 1 0 0 0 0 0 0];
B(4,8,:) = [3 4 0 0 0 0 0 0];
B(4,9,:) = [0 0 0 0 0 0 0 0];

B(9,1,:) = [1 1 0 0 0 0 0 0];
B(9,2,:) = [1 4 0 0 0 0 0 0];
B(9,3,:) = [1 3 0 0 0 0 0 0];
B(9,4,:) = [1 2 0 0 0 0 0 0];
B(9,5,:) = [0 0 0 0 0 0 0 0];
B(9,6,:) = [0 0 0 0 0 0 0 0];
B(9,7,:) = [0 0 0 0 0 0 0 0];
B(9,8,:) = [0 0 0 0 0 0 0 0];
B(9,9,:) = [0 0 0 0 0 0 0 0];

B(8,1,:) = [1 1 4 4 0 0 0 0];
B(8,2,:) = [1 4 0 0 0 0 0 0];
B(8,3,:) = [1 3 0 0 0 0 0 0];
B(8,4,:) = [1 2 4 3 0 0 0 0];
B(8,5,:) = [4 2 0 0 0 0 0 0];
B(8,6,:) = [4 1 0 0 0 0 0 0];
B(8,7,:) = [0 0 0 0 0 0 0 0];
B(8,8,:) = [0 0 0 0 0 0 0 0];
B(8,9,:) = [0 0 0 0 0 0 0 0];

B(7,1,:) = [4 4 0 0 0 0 0 0];
B(7,2,:) = [0 0 0 0 0 0 0 0];
B(7,3,:) = [0 0 0 0 0 0 0 0];
B(7,4,:) = [4 3 0 0 0 0 0 0];
B(7,5,:) = [4 2 0 0 0 0 0 0];
B(7,6,:) = [4 1 0 0 0 0 0 0];
B(7,7,:) = [0 0 0 0 0 0 0 0];
B(7,8,:) = [0 0 0 0 0 0 0 0];
B(7,9,:) = [0 0 0 0 0 0 0 0];
return;

function T = MATRIX_T(Q)
syms x;
syms y;

n = size(Q,2);

T = zeros(n);
for j = 1:n
    T(j,1) = subs(Q(j),[x,y],[0,0]);
    T(j,2) = subs(Q(j),[x,y],[1,0]);
    T(j,3) = subs(Q(j),[x,y],[1,1]);
    T(j,4) = subs(Q(j),[x,y],[0,1]);
```

```
   if(n > 4)
   T(j,5) = subs(diff(Q(j),x),[x,y],[0,0]);
   T(j,6) = subs(diff(Q(j),x),[x,y],[1,0]);
   T(j,7) = subs(diff(Q(j),x),[x,y],[1,1]);
   T(j,8) = subs(diff(Q(j),x),[x,y],[0,1]);
   end
   if(n > 8)
   T(j,9) = subs(diff(Q(j),y),[x,y],[0,0]);
   T(j,10) = subs(diff(Q(j),y),[x,y],[1,0]);
   T(j,11) = subs(diff(Q(j),y),[x,y],[1,1]);
   T(j,12) = subs(diff(Q(j),y),[x,y],[0,1]);
   end
   if(n > 12)
   T(j,13) = subs(diff(diff(Q(j),y),x),[x,y],[0,0]);
   T(j,14) = subs(diff(diff(Q(j),y),x),[x,y],[1,0]);
   T(j,15) = subs(diff(diff(Q(j),y),x),[x,y],[1,1]);
   T(j,16) = subs(diff(diff(Q(j),y),x),[x,y],[0,1]);
   end
end
T = T';
return

function E = Positions(m,n,dx,dy)
    E = zeros((n+1)*(m+1),2);
    ix = n+1;
    iy = m+1;
    for i = 1:(n+1)*(m+1)
        E(i,:) = [dx*(ix-1),dy*(iy-1)];

        iy = iy-1;
        if(iy == 0)
            iy = m+1;
            ix = ix -1;
        end
    end
    %[Cubes,CubeNumbers] = CreateCubes(E,N);
    %Plot(E,N,Cubes)
return
```

Example code for Reissner-Mindlin plate model using bi-cubics

```
%function [E,wP,xP,yP,size_c] = PlateCantileverCubic(d,n,h,inum,numEig)
function [E,n,m] = PlateCantileverCubic(d,n,h,inum,numEig)
```

```
format long g
m = ceil(n*d);
a = 0;
b = 1;
c = 0;

%h = sqrt(12/alpha);
%d = 1;
deltx = (b-a)/n;
delty = (d-c)/m;

size_c = (n+1)*(m+1);

nu = 0.3;

kappa_b = (5/6);
kappa_p = 0.9554;%0.29738*nu + 0.763932;

I = (h^3)/12;
beta = kappa_b/((2*(1+nu))*I);%*alpha;%0.3846*kappa_p/I
A = 1/(beta*(1-nu^2));
B = 1/(2*beta*(1+nu));

[MM,Kxx,Kxy,Kyy,Lx,Ly,Edge] = CalMatrix(n,m,deltx,delty);
LxT = Lx';
LyT = Ly';
Kyx = Kxy';%CHECKED
O = sparse(size(MM,1),size(MM,2));
Mu = [MM O O; O I*MM O; O O I*MM];
Ku = [Kxx+Kyy LxT LyT; h*Lx A*Kxx+B*Kyy+h*MM A*nu*Kyx+B*Kxy; h*
    Ly A*nu*Kxy+B*Kyx A*Kyy+B*Kxx+h*MM];%The correct one!
%Ku = [Kxx+Kyy Lx Ly; h*LxT A*Kxx+B*Kyy+h*MM A*nu*Kxy+B*Kyx; h*
    LyT A*nu*Kyx+B*Kxy A*Kyy+B*Kxx+h*MM];%The not correct one!
%Ku = [h*(Kxx+Kyy) h*Lx h*Ly; h*LxT A*Kxx+B*Kyy+h*MM A*nu*Kxy+B
    *Kyx; h*LyT A*nu*Kyx+B*Kxy A*Kyy+B*Kxx+h*MM];
Mq = [MM O O; O O O; O O O];
F = zeros(size(Mu,1),1);
F(1:(m+1),1) = 0.01;
x = [];

for i = [0 2 4 6 8 10]
    x = [x; Edge+(i)*(m+1)*(n+1)];
end
Mu(x,:) = [];
Mu(:,x) = [];
Ku(x,:) = [];
Ku(:,x) = [];
Mq(x,:) = [];
```

```matlab

[R,p,s] = chol(Mu,'vector');
[V,DE,flag] = eigs(Ku,R,numEig,'smallestabs','IsCholesky',true,
    'CholeskyPermutation',s,'Tolerance',1e-4);
E = diag(DE);

%[V,D] = eigs(Ku,Mu,numEig,'sm');
%E = diag(D);
%V = V(:,E>=0);
%E = E(E>=0);

%tic
%Kug = gpuArray(Ku);
%bg = gpuArray(Mq*(F));
%u = gmres(Kug,bg,30,1e-4,30);
%ueq = gather(u);

%toc
%wP = [ueq(1:(m+1)*(n+1)-(m+1),1); zeros(m+1,1)];

%inum = 1
wP = zeros(inum,(m+1)*(n+1));
%wP(1,:) = [ueq(1:(m+1)*(n+1)-(m+1),1); zeros(m+1,1)];
for i = inum:-1:1
w = V(:,i);
tic
Kug = gpuArray(Ku);
bg = gpuArray(Mu*(w));
u = gmres(Kug,bg,30,1e-4,30);
ueq = gather(u);

%bg = Mu*(w);
%ueq = gmres(Ku,bg,30,1e-4,30);

toc
wP(i,:) = [ueq(1:(m+1)*(n+1)-(m+1),1); zeros(m+1,1)];
end

icx = b;
icy = d;
icount = 1;
xP = zeros(1,(n+1)*(m+1));
yP = zeros(1,(n+1)*(m+1));
for i = 1:n+1
    for j = 1:m+1
        xP(1,icount) = icx;
        yP(1,icount) = icy;
        icy = icy - delty;
```

```
        icount = icount + 1;
    end
    icx = icx - deltx;
    icy = d;
end

%for i = inum:-1:1
%figure();
%scatter3(xP(1,:),yP(1,:),wP(i,:));
%end


%}
%w = V(:,8);

%Kyx = Kxy';%CHECKED
%All = (n+1)*(m+1);

%K1 = Kxx + (1-nu)/2*Kyy;
%K2 = nu*Kyx + (1-nu)/2*Kxy;
%K3 = nu*Kxy + (1-nu)/2*Kyx;
%K4 = Kyy + (1-nu)/2*Kxx;
%O = sparse(size(MM,1),size(MM,2));%CHECKED
%MMu = [MM O;%CHECKED
%       O MM];%CHECKED
%Mf = MMu;
%K = 1/(gamma*(1-nu^2))*[K1 K2; K3 K4];%CHECKED
%x = [7*All:-1:7*All-(m+1)+1 5*All:-1:5*All-(m+1)+1 3*All:-1:3*
    All-(m+1)+1 1*All:-1:1*All-(m+1)+1];
%K(x,:) = [];
%K(:,x) = [];
%MMu(x,:) = [];
%MMu(:,x) = [];
%Mf(x,:) = [];
%CHECKED
%eig(Mu,K)
%[V,D] = eigs(K,MMu,10,'sm');
%E = diag(D);
%w = V(:,8);




%ueq = Ku\Mu*(-w)

%b =Mf*(-F);
%tic
%ueq = K\Mf*(-F);
```

```
%toc

%{
tic
tol = 0.0001;
maxit = 300000;
alpha1 = max(sum(abs(K),2)./diag(K))-2;
L = ichol(K,struct('type','ict','droptol',1e-3,'diagcomp',
    alpha1));
%ueq = pcg(K,b,tol,maxit,L,L');
toc

Ep = Positions(m,n,deltx,delty);
%ux = 0;
%uy = 0;
ux = [ueq(1:(n+1)*(m+1)-(m+1),1);zeros(m+1,1)] + Ep(:,1);
dxux = [ueq((n+1)*(m+1)-(m+1)+1:2*(n+1)*(m+1)-(m+1),1)];
dyux = [ueq(2*(n+1)*(m+1)-(m+1)+1:3*(n+1)*(m+1)-2*(m+1),1);
    zeros(m+1,1)];
dxyux = [ueq(3*(n+1)*(m+1)-2*(m+1)+1:4*(n+1)*(m+1)-2*(m+1),1)];
uy = [ueq(4*(n+1)*(m+1)-2*(m+1)+1:5*(n+1)*(m+1)-3*(m+1),1);
    zeros(m+1,1)] + Ep(:,2);
dxuy = [ueq(5*(n+1)*(m+1)-3*(m+1)+1:6*(n+1)*(m+1)-3*(m+1),1)];
dyuy = [ueq(6*(n+1)*(m+1)-3*(m+1)+1:7*(n+1)*(m+1)-4*(m+1),1);
    zeros(m+1,1)];
dxyuy = [ueq(7*(n+1)*(m+1)-4*(m+1)+1:8*(n+1)*(m+1)-4*(m+1),1)];

ux = flip(ux);
dxux = flip(dxux);
dyux = flip(dyux);
dxyux = flip(dxyux);
uy = flip(uy);
dxuy = flip(dxuy);
dyuy = flip(dyuy);
dxyuy = flip(dxyuy);

stress = ceil((n+1)/2);
figure();
scatter(ux,uy);
sigma11 = 1/(gamma*(1-nu^2))*(dxux + nu*dyuy);
sigma22 = 1/(gamma*(1-nu^2))*(dyuy + nu*dxux);
sigma12 = 1/(2*gamma*(1+nu))*(dyux + dyux);

T = [sigma11(stress) sigma12(stress);sigma12(stress) sigma22(
    stress)];
%}
return;

```

```
function [Mq,Kxxq,Kxyq,Kyyq,Lxq,Lyq] = matrix(deltx,delty)%CHECKED
syms x;
syms y;

Q = [1 x x^2 x^3 y x*y x^2*y x^3*y y^2 x*y^2 x^2*y^2 x^3*y^2 y^3 x*y^3 x^2*y^3 x^3*y^3];

%size_num = size(Q,2)/4;
T = MATRIX_T(Q);

Mq   = zeros(size(Q,2))*x*y;
Kxxq = zeros(size(Q,2))*x*y;
Kxyq = zeros(size(Q,2))*x*y;
Kyyq = zeros(size(Q,2))*x*y;
Lxq  = zeros(size(Q,2))*x*y;
Lyq  = zeros(size(Q,2))*x*y;

for i = 1:size(Q,2)
    for j = 1:size(Q,2)
        Mq(j,i)   = Q(j)*Q(i);
        Kxxq(j,i) = diff(Q(j),x)*diff(Q(i),x);
        Kxyq(j,i) = diff(Q(j),y)*diff(Q(i),x);
        Kyyq(j,i) = diff(Q(j),y)*diff(Q(i),y);
        Lxq(j,i)  = Q(j)*diff(Q(i),x);
        Lyq(j,i)  = Q(j)*diff(Q(i),y);
    end
end
Mq = int(int(Mq,x,0,1),y,0,1);
Kxxq = int(int(Kxxq,x,0,1),y,0,1);
Kxyq = int(int(Kxyq,x,0,1),y,0,1);
Kyyq = int(int(Kyyq,x,0,1),y,0,1);
Lxq = int(int(Lxq,x,0,1),y,0,1);
Lyq = int(int(Lyq,x,0,1),y,0,1);

IT = inv(T);

Mq = (IT)'*Mq*IT;
Kxxq = (IT)'*Kxxq*IT;
Kxyq = (IT)'*Kxyq*IT;
Kyyq = (IT)'*Kyyq*IT;
Lxq = (IT)'*Lxq*IT;
Lyq = (IT)'*Lyq*IT;

Mq = double(Mq*deltx*delty);
Kxxq = double(Kxxq*delty/deltx);
Kyyq = double(Kyyq*deltx/delty);
Kxyq = double(Kxyq);
```

```
Lxq = double(Lxq*delty);
Lyq = double(Lyq*deltx);
return;

function [Adj,Type,Edge] = Domain(n,m)
D = zeros(m+1,n+1);
icount = 1;
for i = n+1:-1:1
   for j = 1:m+1
        D(j,i) = icount;
        icount = icount + 1;
   end
end
%D
Edge1 = [];
Edge2 = [];
Edge3 = [];
Edge4 = [];
Edge1 = (D(:,1));
%Edge2 = (D(m+1,:)');
%Edge3 = (D(:,n+1));
%Edge4 = (D(1,:)');

Edge = sort(unique([Edge1; Edge2; Edge3; Edge4]));
%Edge

D0 = [zeros(1,n+3);zeros(m+1,1) D zeros(m+1,1);zeros(1,n+3)];

icount = 1;
Adj = zeros((n+1)*(m+1),9);
for i = n+2:-1:2
   for j = 2:m+2
   Adj(icount,1) = D0(j,i); %middel
   Adj(icount,2) = D0(j-1,i); %bo
   Adj(icount,3) = D0(j-1,i+1); %regsbo
   Adj(icount,4) = D0(j,i+1); %regs
   Adj(icount,5) = D0(j+1,i+1); %regs onder
   Adj(icount,6) = D0(j+1,i); %onder
   Adj(icount,7) = D0(j+1,i-1); %links onder
   Adj(icount,8) = D0(j,i-1); %links
   Adj(icount,9) = D0(j-1,i-1); %linksbo
   icount = icount +1;
   end
end
Type= zeros((n+1)*(m+1),2);
T = [1      0      0      0      0      2      5      4      0;
     2      1      0      0      0      3      6      5      4;
     3      2      0      0      0      0      0      6      5;
```

```
     4      0      0      1      2      5      8      7      0;
     5      4      1      2      3      6      9      8      7;
     6      5      2      3      0      0      0      9      8;
     7      0      0      4      5      8      0      0      0;
     8      7      4      5      6      9      0      0      0;
     9      8      5      6      0      0      0      0      0];
 nnz(T);

for i = 1:(n+1)*(m+1)
   Type(i,1) = Adj(i,1);
   for j = 1:9
       bflag = true;
       for k = 1:9
           if(any(T(j,k)) ~= any(Adj(i,k)))
               bflag = false;
           end
       end
       if(bflag == true)
          Type(i,2) = j;
       end
   end
end
return

function [M,Kxx,Kxy,Kyy,Lx,Ly,Edge] = CalMatrix(n,m,deltx,delty
    )
[Adj, Type, Edge] = Domain(n,m);
[Mq,Kxxq,Kxyq,Kyyq,Lxq,Lyq] = matrix(deltx,delty);

ns = nnz(Adj);
Ms = zeros(16*ns,1);
Kxxs = zeros(16*ns,1);
Kxys = zeros(16*ns,1);
Kyys = zeros(16*ns,1);
Lxs = zeros(16*ns,1);
Lys = zeros(16*ns,1);

x = [1:(n+1)*(m+1)]';
c = sum(Adj~=0,2);

ix = repelem(x,c);
x = [ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);
    ix;ix+(n+1)*(m+1);ix+2*(n+1)*(m+1);ix+3*(n+1)*(m+1);];

iy = nonzeros(Adj');
y = [iy;iy;iy;iy;
```

```
    iy+(n+1)*(m+1);iy+(n+1)*(m+1);iy+(n+1)*(m+1);iy+(n+1)*(m+1);
    iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);iy+2*(n+1)*(m+1);
    iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1);iy+3*(n+1)*(m+1)];
B = BMatrix();

Adj(Adj == 0) = nan;
NanAdj = ~isnan(Adj);
NanAdj = NanAdj';

a = 1:9;
b = repelem(a,size(Adj,1),1);
b = b';
iz = b(NanAdj);

for i = 1:ns
  k = 1;
  while(B(Type(ix(i),2),iz(i),k) ~= 0)
    Ms(i)   = Ms(i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kxxs(i) = Kxxs(i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kxys(i) = Kxys(i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Kyys(i) = Kyys(i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Lxs(i) = Lxs(i) + Lxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));
    Lys(i) = Lys(i) + Lyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1));

    Ms(ns+i)   = Ms(ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kxxs(ns+i) = Kxxs(ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kxys(ns+i) = Kxys(ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Kyys(ns+i) = Kyys(ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Lxs(ns+i) = Lxs(ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));
    Lys(ns+i) = Lys(ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1));

    Ms(2*ns+i)   = Ms(2*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1));
```

```
    Kxxs(2*ns+i) = Kxxs(2*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)
   +8,B(Type(ix(i),2),iz(i),k+1));
    Kxys(2*ns+i) = Kxys(2*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)
   +8,B(Type(ix(i),2),iz(i),k+1));
    Kyys(2*ns+i) = Kyys(2*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)
   +8,B(Type(ix(i),2),iz(i),k+1));
    Lxs(2*ns+i) = Lxs(2*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+8,
   B(Type(ix(i),2),iz(i),k+1));
    Lys(2*ns+i) = Lys(2*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+8,
   B(Type(ix(i),2),iz(i),k+1));

    Ms(3*ns+i)   = Ms(3*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,
   B(Type(ix(i),2),iz(i),k+1));
    Kxxs(3*ns+i) = Kxxs(3*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1));
    Kxys(3*ns+i) = Kxys(3*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1));
    Kyys(3*ns+i) = Kyys(3*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1));
    Lxs(3*ns+i) = Lxs(3*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1));
    Lys(3*ns+i) = Lys(3*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)
   +12,B(Type(ix(i),2),iz(i),k+1));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(4*ns+i)   = Ms(4*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(
   Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(4*ns+i) = Kxxs(4*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)
   ,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(4*ns+i) = Kxys(4*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)
   ,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(4*ns+i) = Kyys(4*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)
   ,B(Type(ix(i),2),iz(i),k+1)+4);
    Lxs(4*ns+i) = Lxs(4*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k),B(
   Type(ix(i),2),iz(i),k+1)+4);
    Lys(4*ns+i) = Lys(4*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k),B(
   Type(ix(i),2),iz(i),k+1)+4);

    Ms(5*ns+i)   = Ms(5*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B
   (Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(5*ns+i) = Kxxs(5*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)
   +4,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(5*ns+i) = Kxys(5*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)
   +4,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(5*ns+i) = Kyys(5*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)
   +4,B(Type(ix(i),2),iz(i),k+1)+4);
    Lxs(5*ns+i) = Lxs(5*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+4,
   B(Type(ix(i),2),iz(i),k+1)+4);
```

```
    Lys(5*ns+i) = Lys(5*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+4);

    Ms(6*ns+i)   = Ms(6*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(6*ns+i) = Kxxs(6*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(6*ns+i) = Kxys(6*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(6*ns+i) = Kyys(6*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Lxs(6*ns+i) = Lxs(6*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);
    Lys(6*ns+i) = Lys(6*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+4);

    Ms(7*ns+i)   = Ms(7*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxxs(7*ns+i) = Kxxs(7*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kxys(7*ns+i) = Kxys(7*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Kyys(7*ns+i) = Kyys(7*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Lxs(7*ns+i) = Lxs(7*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);
    Lys(7*ns+i) = Lys(7*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(8*ns+i)   = Ms(8*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(8*ns+i) = Kxxs(8*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(8*ns+i) = Kxys(8*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(8*ns+i) = Kyys(8*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Lxs(8*ns+i) = Lxs(8*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);
    Lys(8*ns+i) = Lys(8*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(9*ns+i)   = Ms(9*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(9*ns+i) = Kxxs(9*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
```

```
    Kxys(9*ns+i) = Kxys(9*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(9*ns+i) = Kyys(9*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Lxs(9*ns+i) = Lxs(9*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);
    Lys(9*ns+i) = Lys(9*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(10*ns+i)   = Ms(10*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(10*ns+i) = Kxxs(10*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(10*ns+i) = Kxys(10*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(10*ns+i) = Kyys(10*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Lxs(10*ns+i) = Lxs(10*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);
    Lys(10*ns+i) = Lys(10*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+8);

    Ms(11*ns+i)   = Ms(11*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxxs(11*ns+i) = Kxxs(11*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kxys(11*ns+i) = Kxys(11*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Kyys(11*ns+i) = Kyys(11*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Lxs(11*ns+i) = Lxs(11*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
    Lys(11*ns+i) = Lys(11*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+8);
      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    Ms(12*ns+i)   = Ms(12*ns+i) + Mq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(12*ns+i) = Kxxs(12*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(12*ns+i) = Kxys(12*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(12*ns+i) = Kyys(12*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Lxs(12*ns+i) = Lxs(12*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);
    Lys(12*ns+i) = Lys(12*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k),B(Type(ix(i),2),iz(i),k+1)+12);

```

```
    Ms(13*ns+i)   = Ms(13*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(13*ns+i) = Kxxs(13*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(13*ns+i) = Kxys(13*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(13*ns+i) = Kyys(13*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Lxs(13*ns+i) = Lxs(13*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);
    Lys(13*ns+i) = Lys(13*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+4,B(Type(ix(i),2),iz(i),k+1)+12);

    Ms(14*ns+i)   = Ms(14*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(14*ns+i) = Kxxs(14*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(14*ns+i) = Kxys(14*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(14*ns+i) = Kyys(14*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Lxs(14*ns+i) = Lxs(14*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);
    Lys(14*ns+i) = Lys(14*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+8,B(Type(ix(i),2),iz(i),k+1)+12);

    Ms(15*ns+i)   = Ms(15*ns+i) + Mq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxxs(15*ns+i) = Kxxs(15*ns+i) + Kxxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kxys(15*ns+i) = Kxys(15*ns+i) + Kxyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Kyys(15*ns+i) = Kyys(15*ns+i) + Kyyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Lxs(15*ns+i) = Lxs(15*ns+i) + Lxq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);
    Lys(15*ns+i) = Lys(15*ns+i) + Lyq(B(Type(ix(i),2),iz(i),k)+12,B(Type(ix(i),2),iz(i),k+1)+12);

    k = k + 2;
    if(k > 8)
       break;
    end
  end
end
M   = sparse(x,y,Ms,4*(n+1)*(m+1),4*(n+1)*(m+1));
Kxx = sparse(x,y,Kxxs,4*(n+1)*(m+1),4*(n+1)*(m+1));
Kxy = sparse(x,y,Kxys,4*(n+1)*(m+1),4*(n+1)*(m+1));
```

```
Kyy = sparse(x,y,Kyys,4*(n+1)*(m+1),4*(n+1)*(m+1));
Lx = sparse(x,y,Lxs,4*(n+1)*(m+1),4*(n+1)*(m+1));
Ly = sparse(x,y,Lys,4*(n+1)*(m+1),4*(n+1)*(m+1));
return;

function B = BMatrix()
B = zeros(9,9,8);
B(3,1,:) = [2 2 0 0 0 0 0 0];
B(3,2,:) = [2 3 0 0 0 0 0 0];
B(3,3,:) = [0 0 0 0 0 0 0 0];
B(3,4,:) = [0 0 0 0 0 0 0 0];
B(3,5,:) = [0 0 0 0 0 0 0 0];
B(3,6,:) = [0 0 0 0 0 0 0 0];
B(3,7,:) = [0 0 0 0 0 0 0 0];
B(3,8,:) = [2 1 0 0 0 0 0 0];
B(3,9,:) = [2 4 0 0 0 0 0 0];

B(2,1,:) = [2 2 3 3 0 0 0 0];
B(2,2,:) = [2 3 0 0 0 0 0 0];
B(2,3,:) = [0 0 0 0 0 0 0 0];
B(2,4,:) = [0 0 0 0 0 0 0 0];
B(2,5,:) = [0 0 0 0 0 0 0 0];
B(2,6,:) = [3 2 0 0 0 0 0 0];
B(2,7,:) = [3 1 0 0 0 0 0 0];
B(2,8,:) = [2 1 3 4 0 0 0 0];
B(2,9,:) = [2 4 0 0 0 0 0 0];

B(1,1,:) = [3 3 0 0 0 0 0 0];
B(1,2,:) = [0 0 0 0 0 0 0 0];
B(1,3,:) = [0 0 0 0 0 0 0 0];
B(1,4,:) = [0 0 0 0 0 0 0 0];
B(1,5,:) = [0 0 0 0 0 0 0 0];
B(1,6,:) = [3 2 0 0 0 0 0 0];
B(1,7,:) = [3 1 0 0 0 0 0 0];
B(1,8,:) = [3 4 0 0 0 0 0 0];
B(1,9,:) = [0 0 0 0 0 0 0 0];

B(6,1,:) = [1 1 2 2 0 0 0 0];
B(6,2,:) = [1 4 2 3 0 0 0 0];
B(6,3,:) = [1 3 0 0 0 0 0 0];
B(6,4,:) = [1 2 0 0 0 0 0 0];
B(6,5,:) = [0 0 0 0 0 0 0 0];
B(6,6,:) = [0 0 0 0 0 0 0 0];
B(6,7,:) = [0 0 0 0 0 0 0 0];
B(6,8,:) = [2 1 0 0 0 0 0 0];
B(6,9,:) = [2 4 0 0 0 0 0 0];

B(5,1,:) = [1 1 2 2 3 3 4 4];
```

```
B(5,2,:) = [1 4 2 3 0 0 0 0];
B(5,3,:) = [1 3 0 0 0 0 0 0];
B(5,4,:) = [1 2 4 3 0 0 0 0];
B(5,5,:) = [4 2 0 0 0 0 0 0];
B(5,6,:) = [4 1 3 2 0 0 0 0];
B(5,7,:) = [3 1 0 0 0 0 0 0];
B(5,8,:) = [3 4 2 1 0 0 0 0];
B(5,9,:) = [2 4 0 0 0 0 0 0];

B(4,1,:) = [3 3 4 4 0 0 0 0];
B(4,2,:) = [0 0 0 0 0 0 0 0];
B(4,3,:) = [0 0 0 0 0 0 0 0];
B(4,4,:) = [4 3 0 0 0 0 0 0];
B(4,5,:) = [4 2 0 0 0 0 0 0];
B(4,6,:) = [4 1 3 2 0 0 0 0];
B(4,7,:) = [3 1 0 0 0 0 0 0];
B(4,8,:) = [3 4 0 0 0 0 0 0];
B(4,9,:) = [0 0 0 0 0 0 0 0];

B(9,1,:) = [1 1 0 0 0 0 0 0];
B(9,2,:) = [1 4 0 0 0 0 0 0];
B(9,3,:) = [1 3 0 0 0 0 0 0];
B(9,4,:) = [1 2 0 0 0 0 0 0];
B(9,5,:) = [0 0 0 0 0 0 0 0];
B(9,6,:) = [0 0 0 0 0 0 0 0];
B(9,7,:) = [0 0 0 0 0 0 0 0];
B(9,8,:) = [0 0 0 0 0 0 0 0];
B(9,9,:) = [0 0 0 0 0 0 0 0];

B(8,1,:) = [1 1 4 4 0 0 0 0];
B(8,2,:) = [1 4 0 0 0 0 0 0];
B(8,3,:) = [1 3 0 0 0 0 0 0];
B(8,4,:) = [1 2 4 3 0 0 0 0];
B(8,5,:) = [4 2 0 0 0 0 0 0];
B(8,6,:) = [4 1 0 0 0 0 0 0];
B(8,7,:) = [0 0 0 0 0 0 0 0];
B(8,8,:) = [0 0 0 0 0 0 0 0];
B(8,9,:) = [0 0 0 0 0 0 0 0];

B(7,1,:) = [4 4 0 0 0 0 0 0];
B(7,2,:) = [0 0 0 0 0 0 0 0];
B(7,3,:) = [0 0 0 0 0 0 0 0];
B(7,4,:) = [4 3 0 0 0 0 0 0];
B(7,5,:) = [4 2 0 0 0 0 0 0];
B(7,6,:) = [4 1 0 0 0 0 0 0];
B(7,7,:) = [0 0 0 0 0 0 0 0];
B(7,8,:) = [0 0 0 0 0 0 0 0];
B(7,9,:) = [0 0 0 0 0 0 0 0];
```

```
return;

function T = MATRIX_T(Q)
syms x;
syms y;

n = size(Q,2);

T = zeros(n);
for j = 1:n
   T(j,1) = subs(Q(j),[x,y],[0,0]);
   T(j,2) = subs(Q(j),[x,y],[1,0]);
   T(j,3) = subs(Q(j),[x,y],[1,1]);
   T(j,4) = subs(Q(j),[x,y],[0,1]);
   if(n > 4)
   T(j,5) = subs(diff(Q(j),x),[x,y],[0,0]);
   T(j,6) = subs(diff(Q(j),x),[x,y],[1,0]);
   T(j,7) = subs(diff(Q(j),x),[x,y],[1,1]);
   T(j,8) = subs(diff(Q(j),x),[x,y],[0,1]);
   end
   if(n > 8)
   T(j,9) = subs(diff(Q(j),y),[x,y],[0,0]);
   T(j,10) = subs(diff(Q(j),y),[x,y],[1,0]);
   T(j,11) = subs(diff(Q(j),y),[x,y],[1,1]);
   T(j,12) = subs(diff(Q(j),y),[x,y],[0,1]);
   end
   if(n > 12)
   T(j,13) = subs(diff(diff(Q(j),y),x),[x,y],[0,0]);
   T(j,14) = subs(diff(diff(Q(j),y),x),[x,y],[1,0]);
   T(j,15) = subs(diff(diff(Q(j),y),x),[x,y],[1,1]);
   T(j,16) = subs(diff(diff(Q(j),y),x),[x,y],[0,1]);
   end
end
T = T';
return

function E = Positions(m,n,dx,dy)
    E = zeros((n+1)*(m+1),2);
    ix = n+1;
    iy = m+1;
    for i = 1:(n+1)*(m+1)
        E(i,:) = [dx*(ix-1),dy*(iy-1)];

        iy = iy-1;
        if(iy == 0)
            iy = m+1;
            ix = ix -1;
        end
```

```
    end
    %[Cubes,CubeNumbers] = CreateCubes(E,N);
    %Plot(E,N,Cubes)
return
```

Example code for three-dimensional elastic body using tri-cubics

```
%function [Eig,uxB,uyB,uzB,sz] = Copy_of_CubePC(s,n1,h,inum,
    numEig)
function [Eig,n1,n2,n3] = Copy_of_CubePC(s,n1,h,inum,numEig)
format long g
warning off;
method =2;



%mkdir(strcat('\Plots\',sprintf('%.6f',s)));
%gpuDevice(1);
    %mwb = MultiWaitBar(3, 1, '3-Dimensional Beam Eigenvalue
    Calculator', 'g');
    %mwb.Update(1, 1, 0, 'Total Progress - Setting parameters')
    ;
    %mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(0) '%']);
    %mwb.Update(3, 1, 0, 'Plot');
    %alpha = 1200;
    %d2 = sqrt(s/alpha);
    %d1 = sqrt((12*s^2)/(alpha*(1+s^2)));
    %d1 = sqrt(12/(alpha));
    d1 = h;
    d2 = s;
    %d2 = 1;

    n2 = ceil(n1*h);
    if(n2 <= 10)
        n2 = 6;
    end
    n3 = ceil(n2*s);
    %n3 = ceil(n1/s);
    if(n3 <= 10)
        n3 = 6;
    end
    %n2 = 3;

    sz = n1*n2*n3;
```

```matlab

 S = [0 1 0 d1 0 d2]; %Set size of the beam
 N = [n1 n2 n3]; %Number of elements
 Delta = [(S(2)-S(1))/N(1) (S(4)-S(3))/N(2) (S(6)-S(5))/N(3)]; %space step size
 nu = 0.3;
 gamma = 1/(2*(1+nu))*5/6
 %A = 1/(gamma*(1+nu)*(1-2*nu));
 %B = 1/(2*gamma*(1+nu));
 for i = inum:-1:50
     h(i) = figure(i);
     movegui(h(i),'west')
 end
 %A = 1/(1-nu^2);
 %B = 1/(2*gamma*(1+nu));
 %mwb.Update(1, 1, 0.1, 'Total Progress - Creating Matrices');
 [K11,K12,K13,K22,K23,K33,M0,Dom,E] = Matrices(Delta,N,method);
 %mwb.Update(1, 1, 0.3, 'Total Progress - Admissible Basis functions');
 %Om = Omega(N,Dom);
 %F = Initial(N,f);
 Mf = M0;

%K11(1:(N(2)+1)*(N(3)+1),:) = [];
%K11(:,1:(N(2)+1)*(N(3)+1)) = [];
%K12(1:(N(2)+1)*(N(3)+1),:) = [];
%K12(:,1:(N(2)+1)*(N(3)+1)) = [];
%K13(1:(N(2)+1)*(N(3)+1),:) = [];
%K13(:,1:(N(2)+1)*(N(3)+1)) = [];
%K22(1:(N(2)+1)*(N(3)+1),:) = [];
%K22(:,1:(N(2)+1)*(N(3)+1)) = [];
%K23(1:(N(2)+1)*(N(3)+1),:) = [];
%K23(:,1:(N(2)+1)*(N(3)+1)) = [];
%K33(1:(N(2)+1)*(N(3)+1),:) = [];
%K33(:,1:(N(2)+1)*(N(3)+1)) = [];

%M0(1:(N(2)+1)*(N(3)+1),:) = [];
%M0(:,1:(N(2)+1)*(N(3)+1)) = [];
%Mf(1:(N(2)+1)*(N(3)+1),:) = [];

 %mwb.Update(1, 1, 0.4, 'Total Progress - Concatinating matrices');
 Of = sparse(size(Mf,1),size(Mf,2));
 MF = [Mf Of Of; Of Mf Of; Of Of Mf];
 O = sparse(size(M0,1),size(M0,2));
 %M = sparse(3*size(M0,1),3*size(M0,2));
```

```
    M = [M0 0 0; 0 M0 0; 0 0 M0];
    %M([1:size(M0,1)],[1:size(M0,2)]) = M0;
    %M(2*[1:size(M0,1)],2*[1:size(M0,2)]) = M0;
    %M(3*[1:size(M0,1)],3*[1:size(M0,2)]) = M0;
    Mf = M;
    FS = size(M);
    %M = [M0 0; 0 M0];
    K21 = K12';
    K31 = K13';
    K32 = K23';

    a1 = 1/(gamma*(1+nu));
    a2 = nu/(gamma*(1+nu)*(1-2*nu));
    a3 = 1/(2*gamma*(1+nu));

    K1 = a1*K11 + a2*K11 + a3*K22 + a3*K33;
    K2 = a3*K12 + a2*K21;
    K3 = a3*K13 + a2*K31;
    K4 = a2*K12 + a3*K21;
    K5 = a1*K22 + a2*K22 + a3*K11 + a3*K33;
    K6 = a3*K23 + a2*K32;
    K7 = a2*K13 + a3*K31;
    K8 = a2*K23 + a3*K32;
    K9 = a1*K33 + a2*K33 + a3*K11 + a3*K22;

    %K = sparse(size(K1,1)*3,size(K1,2)*3);

    %K([1:size(K1,1)],[1:size(K1,2)]) = K1;
    %K([1:size(K1,1)],2*[1:size(K1,2)]) = K2;
    %K([1:size(K1,1)],3*[1:size(K1,2)]) = K3;
    %K(2*[1:size(K1,1)],[1:size(K1,2)]) = K4;
    %K(2*[1:size(K1,1)],2*[1:size(K1,2)]) = K5;
    %K(2*[1:size(K1,1)],3*[1:size(K1,2)]) = K6;
    %K(3*[1:size(K1,1)],[1:size(K1,2)]) = K7;
    %K(3*[1:size(K1,1)],2*[1:size(K1,2)]) = K8;
    %K(3*[1:size(K1,1)],3*[1:size(K1,2)]) = K9;

    K = [K1 K2 K3; K4 K5 K6; K7 K8 K9];

    All = (N(1)+1)*(N(2)+1)*(N(3)+1);
    x = [22*All+(N(2)+1)*(N(3)+1):-1:22*All+1
         19*All+(N(2)+1)*(N(3)+1):-1:19*All+1
         18*All+(N(2)+1)*(N(3)+1):-1:18*All+1
         16*All+(N(2)+1)*(N(3)+1):-1:16*All+1

         14*All+(N(2)+1)*(N(3)+1):-1:14*All+1
         11*All+(N(2)+1)*(N(3)+1):-1:11*All+1
         10*All+(N(2)+1)*(N(3)+1):-1:10*All+1
```

```
        8*All+(N(2)+1)*(N(3)+1):-1:8*All+1

        6*All+(N(2)+1)*(N(3)+1):-1:6*All+1
        3*All+(N(2)+1)*(N(3)+1):-1:3*All+1
        2*All+(N(2)+1)*(N(3)+1):-1:2*All+1
        0*All+(N(2)+1)*(N(3)+1):-1:0*All+1];

    K(x,:) = [];
    K(:,x) = [];
    M(x,:) = [];
    M(:,x) = [];
    Mf(x,:) = [];

   % K = [1/(gamma*(1+nu))*K11+nu/(gamma*(1+nu)*(1-2*nu))*(K11+K22+K33)  1/(gamma*(1+nu))*K12 1/(gamma*(1+nu))*K13;
  %       1/(gamma*(1+nu))*K12 1/(gamma*(1+nu))*K22+nu/(gamma*(1+nu)*(1-2*nu))*(K11+K22+K33) 1/(gamma*(1+nu))*K23;
   %       1/(gamma*(1+nu))*K13 1/(gamma*(1+nu))*K23 1/(gamma*(1+nu))*K33+nu/(gamma*(1+nu)*(1-2*nu))*(K11+K22+K33)];

   % K33 = (-nu/((1-2*nu)+nu))*(K11+K22);
   % K = [1/(gamma*(1+nu))*K11+nu/(gamma*(1+nu)*(1-2*nu))*(K11+K22+K33)  1/(gamma*(1+nu))*K12;
   %       1/(gamma*(1+nu))*K12 1/(gamma*(1+nu))*K22+nu/(gamma*(1+nu)*(1-2*nu))*(K11+K22+K33)];


    %K = [2*(1-nu)*K11+(1-2*nu)*K22+(1-2*nu)*K33 2*nu*K12+(1-2*nu)*K21 2*nu*K13+(1-2*nu)*K31;
    %    2*nu*K21+(1-2*nu)*K12 (1-2*nu)*K11+2*(1-nu)*K22+(1-2*nu)*K33 2*nu*K23+(1-2*nu)*K32;
    %    2*nu*K31+(1-2*nu)*K13 2*nu*K32+(1-2*nu)*K23 (1-2*nu)*K11+(1-2*nu)*K22+2*(1-nu)*K33];
    %K = 1/(2*gamma*(1+nu)*(1-2*nu))*K;

    %K = [K11+(1-nu)/2*K22+(1-nu)/2*K33 (1-nu)/2*K12+nu*K21 (1-nu)/2*K13+nu*K31;
    %    (1-nu)/2*K21+nu*K12 (1-nu)/2*K11+K22+(1-nu)/2*K33 (1-nu)/2*K23+nu*K32;
    %    (1-nu)/2*K31+nu*K13 (1-nu)/2*K32+nu*K23 (1-nu)/2*K11+(1-nu)/2*K22+K33];
    % K = 1/(gamma*(1-nu^2))*K;
      whos k

%numEig = 100;
%{
alpha = max(sum(abs(K),2)./diag(K))-2;
```

```
L = ichol(K,struct('type','ict','droptol',1e-3,'diagcomp',alpha
    ));
n = size(K,1);
[V,D] = eigs(@(x)pcg(K,x,1e-3,200,L,L'),n,M,numEig,'sm');
%}
 %mwb.Update(1, 1, 0.5, 'Total Progress - Cholsky Decomposition
    ');
[R,p,s] = chol(M,'vector');
p;
 %mwb.Update(1, 1, 0.55, 'Total Progress - Eigs');

%Rand = sprand(K);
%[v, lambda] = lobpcg(Rand, K, M, 1e-5, 20,0)
[V,DE,flag] = eigs(K,R,numEig,'smallestabs','IsCholesky',true,'
    CholeskyPermutation',s,'Tolerance',1e-4);
flag;
%mwb.Update(1, 1, 0.6, 'Total Progress');
Eig = diag(DE);
%%Mg = gpuArray(M);
%%Kg = gpuArray(K);

%{
sV = size(Eig,1);
R = zeros(sV,sV);
for i = 1:sV
   for j = 1:sV
     X = K*V(:,i) - M*V(:,i)*D(j);
     NORMX = norm(X,Inf);
     R(j,i) = NORMX;
   end
end
R = K*V-M*V*D;
xlswrite('CompareEigenValues.xlsx',R)
%}

u1p = 0;
u2p = 0;
u3p = 0;
u1s = 0;
u2s = 0;
u3s = 0;
uplx= 0;
uply = 0;
Psize = 0;
T = 0;
%%{

uxB = zeros(inum,(N(1)+1),(N(2)+1),(N(3)+1));
```

```
uyB = zeros(inum,(N(1)+1),(N(2)+1),(N(3)+1));
uzB = zeros(inum,(N(1)+1),(N(2)+1),(N(3)+1));

f = 0.3;

[D,E] = Domain(N,Delta);
%TD = D(:,ceil((N(2)+1)/2),ceil((N(3)+1)/2));
%TD = D(:,ceil((N(2)+1)/2),:);
TD = D(:,:,ceil((N(3)+1)/2));
TD = TD(:);
TDV = sort(TD(:));
OD = D(:,ceil((N(2)+1)/2),ceil((N(3)+1)/2));

F1 = zeros((N(1)+1)*(N(2)+1)*(N(3)+1),1);

F1(D(N(1)+1,ceil((N(2)+2)/2),ceil((N(3)+2)/2))) = f;
%F1(D(N(1)+1,:,:)) = f;
F = zeros(24*(N(1)+1)*(N(2)+1)*(N(3)+1),1);
F(8*(N(1)+1)*(N(2)+1)*(N(3)+1)+1:9*(N(1)+1)*(N(2)+1)*(N(3)+1))
    = F1;


%Kug = gpuArray(K);
%bg = gpuArray(Mf*(F));
%u = gmres(Kug,bg,30,1e-4,30);
%ueq = gather(u);


for i = inum:-1:1

w = V(:,i);

%Kug = gpuArray(K);
%bg = gpuArray(M*(-w));
%u = gmres(Kug,bg,30,1e-4,30);
%ueq = gather(u);

%ueq = K\(Mf*F);


b = M*(-w);
ueq = gmres(K,b,30,1e-4,30);

ux = [zeros((N(2)+1)*(N(3)+1),1); ueq(1:(N(1)+1)*(N(2)+1)*(N(3)
    +1)-(N(2)+1)*(N(3)+1),1)]+E(:,1);
dxux = [ueq((N(1)+1)*(N(2)+1)*(N(3)+1)-(N(2)+1)*(N(3)+1)+1:2*(N
    (1)+1)*(N(2)+1)*(N(3)+1)-(N(2)+1)*(N(3)+1),1)];
```

```
dyux = [zeros((N(2)+1)*(N(3)+1),1); ueq(2*(N(1)+1)*(N(2)+1)*(N(3)+1)-(N(2)+1)*(N(3)+1)+1:3*(N(1)+1)*(N(2)+1)*(N(3)+1)-2*(N(2)+1)*(N(3)+1),1)];
dzux = [zeros((N(2)+1)*(N(3)+1),1); ueq(3*(N(1)+1)*(N(2)+1)*(N(3)+1)-2*(N(2)+1)*(N(3)+1)+1:4*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1),1)];
dxyux = [ueq(4*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1)+1:5*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1),1)];
dxzux = [ueq(5*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1)+1:6*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1),1)];
dyzux = [zeros((N(2)+1)*(N(3)+1),1); ueq(6*(N(1)+1)*(N(2)+1)*(N(3)+1)-3*(N(2)+1)*(N(3)+1)+1:7*(N(1)+1)*(N(2)+1)*(N(3)+1)-4*(N(2)+1)*(N(3)+1),1)];
dxyzux = [ueq(7*(N(1)+1)*(N(2)+1)*(N(3)+1)-4*(N(2)+1)*(N(3)+1)+1:8*(N(1)+1)*(N(2)+1)*(N(3)+1)-4*(N(2)+1)*(N(3)+1),1)];
uy = [zeros((N(2)+1)*(N(3)+1),1); ueq(8*(N(1)+1)*(N(2)+1)*(N(3)+1)-4*(N(2)+1)*(N(3)+1)+1:9*(N(1)+1)*(N(2)+1)*(N(3)+1)-5*(N(2)+1)*(N(3)+1),1)]+E(:,2);
dxuy = [ueq(9*(N(1)+1)*(N(2)+1)*(N(3)+1)-5*(N(2)+1)*(N(3)+1)+1:10*(N(1)+1)*(N(2)+1)*(N(3)+1)-5*(N(2)+1)*(N(3)+1),1)];
dyuy = [zeros((N(2)+1)*(N(3)+1),1); ueq(10*(N(1)+1)*(N(2)+1)*(N(3)+1)-5*(N(2)+1)*(N(3)+1)+1:11*(N(1)+1)*(N(2)+1)*(N(3)+1)-6*(N(2)+1)*(N(3)+1),1)];
dzuy = [zeros((N(2)+1)*(N(3)+1),1); ueq(11*(N(1)+1)*(N(2)+1)*(N(3)+1)-6*(N(2)+1)*(N(3)+1)+1:12*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1),1)];
dxyuy = [ueq(12*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1)+1:13*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1),1)];
dxzuy = [ueq(13*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1)+1:14*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1),1)];
dyzuy = [zeros((N(2)+1)*(N(3)+1),1); ueq(14*(N(1)+1)*(N(2)+1)*(N(3)+1)-7*(N(2)+1)*(N(3)+1)+1:15*(N(1)+1)*(N(2)+1)*(N(3)+1)-8*(N(2)+1)*(N(3)+1),1)];
dxyzuy = [ueq(15*(N(1)+1)*(N(2)+1)*(N(3)+1)-8*(N(2)+1)*(N(3)+1)+1:16*(N(1)+1)*(N(2)+1)*(N(3)+1)-8*(N(2)+1)*(N(3)+1),1)];
uz = [zeros((N(2)+1)*(N(3)+1),1); ueq(16*(N(1)+1)*(N(2)+1)*(N(3)+1)-8*(N(2)+1)*(N(3)+1)+1:17*(N(1)+1)*(N(2)+1)*(N(3)+1)-9*(N(2)+1)*(N(3)+1),1)]+E(:,3);
dxuz = [ueq(17*(N(1)+1)*(N(2)+1)*(N(3)+1)-9*(N(2)+1)*(N(3)+1)+1:18*(N(1)+1)*(N(2)+1)*(N(3)+1)-9*(N(2)+1)*(N(3)+1),1)];
dyuz = [zeros((N(2)+1)*(N(3)+1),1); ueq(18*(N(1)+1)*(N(2)+1)*(N(3)+1)-9*(N(2)+1)*(N(3)+1)+1:19*(N(1)+1)*(N(2)+1)*(N(3)+1)-10*(N(2)+1)*(N(3)+1),1)];
dzuz = [zeros((N(2)+1)*(N(3)+1),1); ueq(19*(N(1)+1)*(N(2)+1)*(N(3)+1)-10*(N(2)+1)*(N(3)+1)+1:20*(N(1)+1)*(N(2)+1)*(N(3)+1)-11*(N(2)+1)*(N(3)+1),1)];
dxyuz = [ueq(20*(N(1)+1)*(N(2)+1)*(N(3)+1)-11*(N(2)+1)*(N(3)+1)+1:21*(N(1)+1)*(N(2)+1)*(N(3)+1)-11*(N(2)+1)*(N(3)+1),1)];
```

```
dxzuz = [ueq(21*(N(1)+1)*(N(2)+1)*(N(3)+1)-11*(N(2)+1)*(N(3)+1)
    +1:22*(N(1)+1)*(N(2)+1)*(N(3)+1)-11*(N(2)+1)*(N(3)+1),1)];
dyzuz = [zeros((N(2)+1)*(N(3)+1),1); ueq(22*(N(1)+1)*(N(2)+1)*(
    N(3)+1)-11*(N(2)+1)*(N(3)+1)+1:23*(N(1)+1)*(N(2)+1)*(N(3)
    +1)-12*(N(2)+1)*(N(3)+1),1)];
dxyzuz = [ueq(23*(N(1)+1)*(N(2)+1)*(N(3)+1)-12*(N(2)+1)*(N(3)
    +1)+1:24*(N(1)+1)*(N(2)+1)*(N(3)+1)-12*(N(2)+1)*(N(3)+1),1)
    ];


f = figure(i);
movegui(f,'west')
scatter3(ux(TD),uy(TD),uz(TD),5,uz(TD))
title(Eig(i));


%uxB(i,1:N(1)+1,1,1:N(3)+1) = ux(TD);
%uyB(i,1:N(1)+1,1,1:N(3)+1) = uy(TD);
%uzB(i,1:N(1)+1,1,1:N(3)+1) = uz(TD);


%dxuxB(i,1:N(1)+1,1,1) = dxux(TD);
%dxuyB(i,1:N(1)+1,1,1) = dxuy(TD);
%dxuzB(i,1:N(1)+1,1,1) = dxuz(TD);
%dyuxB(i,1:N(1)+1,1,1) = dyux(TD);
%dyuyB(i,1:N(1)+1,1,1) = dyuy(TD);
%dyuzB(i,1:N(1)+1,1,1) = dyuz(TD);
%dzuxB(i,1:N(1)+1,1,1) = dzux(TD);
%dzuyB(i,1:N(1)+1,1,1) = dzuy(TD);
%dzuzB(i,1:N(1)+1,1,1) = dzuz(TD);
%dxyuxB(i,1:N(1)+1,1,1) = dxyux(TD);
%dxyuyB(i,1:N(1)+1,1,1) = dxyuy(TD);
%dxyuzB(i,1:N(1)+1,1,1) = dxyuz(TD);
%dxzuxB(i,1:N(1)+1,1,1) = dxzux(TD);
%dxzuyB(i,1:N(1)+1,1,1) = dxzuy(TD);
%dxzuzB(i,1:N(1)+1,1,1) = dxzuz(TD);
%dyzuxB(i,1:N(1)+1,1,1) = dyzux(TD);
%dyzuyB(i,1:N(1)+1,1,1) = dyzuy(TD);
%dyzuzB(i,1:N(1)+1,1,1) = dyzuz(TD);
%dxyzuxB(i,1:N(1)+1,1,1) = dxyzux(TD);
%dxyzuyB(i,1:N(1)+1,1,1) = dxyzuy(TD);
%dxyzuzB(i,1:N(1)+1,1,1) = dxyzuz(TD);
%set(0,'CurrentFigure',h(i));
%scatter3(ux,uy,uz)
%title(Eig(i));
%scatter3(ux,uy,uz,5,uz)

%scatter3(uxB,uyB,uzB,5,uzB)
```

```
%hold on
%scatter3(ux,uy,zeros(size(uz)));

%hold on
%ux2 = ux(D(:,1,1));
%uy2 = uy(D(:,1,1));
%uz2 = uz(D(:,1,1));

%max2 = norm(uy2,Inf);
%uy2 = uy2/max2*0.8;
%scatter3(ux2,uy2,uz2)

hold off
%axis([0 1.1 -0.025 0.05 -0.025 0.025])
%dxux2 = dxux(TDV);
%dxuy2 = dxuy(TDV);
%dxuz2 = dxuz(TDV);
%dyux2 = dyux(TDV);
%dyuy2 = dyuy(TDV);
%dyuz2 = dyuz(TDV);
%dzux2 = dzux(TDV);
%dzuy2 = dzuy(TDV);
%dzuz2 = dzuz(TDV);


%sigma11 = 1/(gamma*(1+nu))*dxuxB + nu/(gamma*(1+nu)*(1-2*nu))
    *(dxuxB+dyuyB+dzuzB);
%sigma22 = 1/(gamma*(1+nu))*dyuyB + nu/(gamma*(1+nu)*(1-2*nu))
    *(dxuxB+dyuyB+dzuzB);
%sigma33 = 1/(gamma*(1+nu))*dzuzB + nu/(gamma*(1+nu)*(1-2*nu))
    *(dxuxB+dyuyB+dzuzB);
%sigma23 = 1/(2*gamma*(1+nu))*(dzuyB + dyuzB);
%sigma31 = 1/(2*gamma*(1+nu))*(dzuxB + dxuzB);
%sigma12 = 1/(2*gamma*(1+nu))*(dyuxB + dxuyB);

%stress = ceil((N(1)+1)/2);


%Ty = [0.5*(dxux(stress) + dxux(stress)) 0.5*(dxuy(stress) +
    dyux(stress)); 0.5*(dyux(stress) + dxuy(stress)) 0.5*(dyuy(
    stress) + dyuy(stress))]
%Tz = [0.5*(dxux(stress) + dxux(stress)) 0.5*(dxuz(stress) +
    dzux(stress)); 0.5*(dzux(stress) + dxuz(stress)) 0.5*(dzuz(
    stress) + dzuz(stress))]
%T = [sigma11(stress) sigma12(stress) sigma31(stress); sigma12(
    stress) sigma22(stress) sigma23(stress); sigma31(stress)
    sigma23(stress) sigma33(stress)]

```

```
%ux1 = ux(OD);
%uy1 = uy(OD);
%uz1 = uz(OD);


%f = figure();
%movegui(f,pos);
%scatter3(ux,uy,uz);
%hold on
%grid on
%plot3(ux1,uy1,uz1);
%title(strcat(num2str(i),' - ',num2str(Eig(i))))
%view(2)
end
%}
%{
%[KM, KMPat] = sparseinv(K);
%KM = KM*M;
%smallest = 0;
%bestEig = 0;

for i = 20:-1:1
  %smallest = 100;
  plot_fig = figure('NumberTitle', 'off', 'Name', strcat('Eigenvalue: ',int2str(i),' - ',num2str(s)));
  w = V(:,i);

%   for j = 1:numEig
%     X = K*w - M*w*Eig(j);
%     if norm(X) < smallest
%         smallest = norm(X);
%         bestEig = Eig(j);
%     end
%   end
   %plot_fig.suptitle(strcat(int2str(i),' - ',num2str(s)));
%mwb.Update(3, 1, 0.1, 'Plot');
    %wg = gpuArray(w);
    alpha = max(sum(abs(K),2)./diag(K))-2;
    L = ichol(K,struct('type','ict','droptol',1e-3,'diagcomp',alpha));
    u = pcg(K,M*w,1e-1,2000000,L,L');


    %normalize = norm(w,Inf);
    %u = normalize*u;
   % u = (K)\MF*F;

    %alpha = max(sum(abs(K),2)./diag(K))-2;
```

```
    %L = ichol(K,struct('type','ict','droptol',1e-3,'diagcomp',alpha));
    %u = pcg(K,MF*F,1e-3,200000,L,L');
   % mwb.Update(3, 1, 0.3, 'Plot');
    u1 = [zeros((N(2)+1)*(N(3)+1),1); u(1:size(u,1)/3)] + E(:,1);
    u2 = [zeros((N(2)+1)*(N(3)+1),1); u(size(u,1)/3+1:2*size(u,1)/3)] + E(:,2);
    u3 = [zeros((N(2)+1)*(N(3)+1),1); u(2*size(u,1)/3+1:3*size(u,1)/3)]+ E(:,3);


    u1 = 1/norm(u1,Inf)*u1;
    u2 = 1/norm(u2,Inf)*u2;
    u3 = 1/norm(u3,Inf)*u3;

    [D,E] = Domain(N,Delta);
    plane = D(:,:,ceil((N(3)+1)/2))';
    plane = plane(:);
    uplx = u1(plane);
    uply = u2(plane);

   % w1 = w(1:size(w,1)/3);
   % w2 = w(size(w,1)/3+1:2*size(w,1)/3);
   % w3 = w(2*size(w,1)/3+1:3*size(w,1)/3);

  % mwb.Update(3, 1, 0.4, 'Plot');
    ix = [];
    if(N(1)+1 > 200)
       % iy = [1 (N(2)+2)/2 (N(2)+1) (N(2)+1)*(N(3)+1)-(N(2)+1)+1 (N(2)+1)*(N(3)+1)];
        ih = ((N(2)+1)-1)/2;
        iv = 1+((N(3)+1)-1)/2*(N(2)+1);
        iy = iv+ih; %[1 1+ih 1+2*ih iv iv+ih iv+2*ih 2*iv-1 2*iv+ih-1 2*iv+2*ih-1];%[iv+ih];%
        icount = 1;
        div = floor((N(1)+1)/200);
        for k = 1:div:(N(1)+1)
            for j = 1:size(iy,2)
                ix(icount) = iy(j)+ (k-1)*(N(2)+1)*(N(3)+1);
                icount = icount +1;
            end
        end
        u1p = u1(ix);
        u2p = u2(ix);
        u3p = u3(ix);
       % w1p = w1(ix);
       % w2p = w2(ix);
```

```
        % w3p = w3(ix);

        E1p = E(ix,1);
        E2p = E(ix,2);
        E3p = E(ix,3);
        Psize = size(iy,2);
    else
        u1p = u1;
        u2p = u2;
        u3p = u3;

      %  w1p = w1;
      %  w2p = w2;
      %  w3p = w3;

        E1p = E(:,1);
        E2p = E(:,2);
        E3p = E(:,3);

        Psize = (N(2)+1)*(N(3)+1);
    end
    %mwb.Update(3, 1, 0.7, 'Plot');

    u1p = 1/norm(u1p,Inf)*u1p;
    %u2p = 1/norm(u2p,Inf)*u2p;
    %u3p = 1/norm(u3p,Inf)*u3p;

    %umx = u1p(size(u1p,1)/2


    scatter3(u1p,u2p,u3p);
    hold on
    %scatter3(E1p,E2p,E3p,0.1);
    hold on
    %scatter3(w1p,w2p,w3p);
    %mwb.Update(3, 1, 1, 'Plot');
    u1s = size(u1p);
    u2s = size(u2p);
    u3s = size(u3p);
    for k = 1:Psize
        hold on
        plot3(u1p(k:Psize:k+u1s-2*Psize),u2p(k:Psize:k+u2s-2*
   Psize),u3p(k:Psize:k+u3s-2*Psize),'-');
    end
temp_png = strcat('\Plots\',sprintf('%.6f',n1),'\PNG\Plot',
   sprintf('%.6f',i),'.png');
temp_fig = strcat('\Plots\',sprintf('%.6f',n1),'\Fig\Plot',
   sprintf('%.6f',i),'.fig');
```

```
view([0 0 90])
%legend(['Eigenvalue: ' num2str(Eig(i))]);
%saveas(plot_fig,strcat(pwd,temp_png))
%savefig(plot_fig,strcat(pwd,temp_fig))
%close(plot_fig)
end
clear u

%}
%mwb.Update(1, 1, 1, 'Total Progress');
%   mwb.Close();
return;

function [D,E] = Domain(N,Delta)
    D = zeros(N(1)+1,N(2)+1,N(3)+1);
    icount = 1;

    for i = 1:N(1)+1
       for k = 1:N(3)+1
          for j = 1:N(2)+1
             D(i,j,k) = icount;
             icount = icount + 1;
          end
       end
    end
    E = zeros((N(1)+1)*(N(2)+1)*(N(3)+1),3);
    ix = 1;
    iy = 1;
    iz = 1;
    ixt = 0;
    for i = 1:(N(1)+1)*(N(2)+1)*(N(3)+1)
        E(i,:) = [Delta(1)*(ix-1),Delta(2)*(iy-1),Delta(3)*(iz
   -1)];

        iy = iy+1;

        if(ix == N(1)+2)
            ix = 1;
        end
        if(iy == N(2)+2)
            iy = 1;
            ixt = ixt +1;
            iz = iz+1;
        end
        if(ixt == N(3)+1)
            ix = ix+1;
            ixt = 0;
        end
```

```
        if(iz == N(3)+2)
            iz = 1;
        end

    end
    %[Cubes,CubeNumbers] = CreateCubes(E,N);
    %Plot(E,N,Cubes)
return

function Next = Adjacent(N,D)
    Next = zeros((N(1)+1)*(N(2)+1)*(N(3)+1),27);
    %1 - Itself
    %2 - Forward
    %3 - Backward
    %4 - Forward + Left
    %5 - Forward + Right
    %6 - Left
    %7 - Right
    %8 - Backward + Left
    %9 - Backward + Right
    for i = 1:N(1)+1
       for j = 1:N(2)+1
          for k = 1:N(3)+1
             Next(D(i,j,k),1) = D(i,j,k);
             if(i<N(1)+1)
                Next(D(i,j,k),2) = D(i+1,j,k);
             else
                 Next(D(i,j,k),2) = nan;
             end
             if(i>1)
                Next(D(i,j,k),3) = D(i-1,j,k);
             else
                 Next(D(i,j,k),3) = nan;
             end
             if(i<N(1)+1 && j < N(2)+1)
                Next(D(i,j,k),4) = D(i+1,j+1,k);
             else
                 Next(D(i,j,k),4) = nan;
             end
             if(i<N(1)+1 && j > 1)
                Next(D(i,j,k),5) = D(i+1,j-1,k);
             else
                 Next(D(i,j,k),5) = nan;
             end
             if(j < N(2)+1)
                Next(D(i,j,k),6) = D(i,j+1,k);
             else
                 Next(D(i,j,k),6) = nan;
```

```
                end
                if(j>1)
                   Next(D(i,j,k),7) = D(i,j-1,k);
                else
                    Next(D(i,j,k),7) = nan;
                end
                if(i>1 && j < N(2)+1)
                   Next(D(i,j,k),8) = D(i-1,j+1,k);
                else
                    Next(D(i,j,k),8) = nan;
                end
                if(i>1 && j>1)
                   Next(D(i,j,k),9) = D(i-1,j-1,k);
                else
                    Next(D(i,j,k),9) = nan;
                end

                if(k < N(3)+1)
                    Next(D(i,j,k),10) = D(i,j,k+1);
                    if(i<N(1)+1)
                       Next(D(i,j,k),11) = D(i+1,j,k+1);
                    else
                        Next(D(i,j,k),11) = nan;
                    end
                    if(i>1)
                       Next(D(i,j,k),12) = D(i-1,j,k+1);
                    else
                        Next(D(i,j,k),12) = nan;
                    end
                    if(i<N(1)+1 && j < N(2)+1)
                       Next(D(i,j,k),13) = D(i+1,j+1,k+1);
                    else
                        Next(D(i,j,k),13) = nan;
                    end
                    if(i<N(1)+1 && j > 1)
                       Next(D(i,j,k),14) = D(i+1,j-1,k+1);
                    else
                        Next(D(i,j,k),14) = nan;
                    end
                    if(j < N(2)+1)
                       Next(D(i,j,k),15) = D(i,j+1,k+1);
                    else
                        Next(D(i,j,k),15) = nan;
                    end
                    if(j>1)
                       Next(D(i,j,k),16) = D(i,j-1,k+1);
                    else
                        Next(D(i,j,k),16) = nan;
```

```
                end
                if(i>1 && j < N(2)+1)
                   Next(D(i,j,k),17) = D(i-1,j+1,k+1);
                else
                    Next(D(i,j,k),17) = nan;
                end
                if(i>1 && j>1)
                   Next(D(i,j,k),18) = D(i-1,j-1,k+1);
                else
                    Next(D(i,j,k),18) = nan;
                end
            else
                Next(D(i,j,k),10) = nan;
                Next(D(i,j,k),11) = nan;
                Next(D(i,j,k),12) = nan;
                Next(D(i,j,k),13) = nan;
                Next(D(i,j,k),14) = nan;
                Next(D(i,j,k),15) = nan;
                Next(D(i,j,k),16) = nan;
                Next(D(i,j,k),17) = nan;
                Next(D(i,j,k),18) = nan;
            end

            if(k>1)
                Next(D(i,j,k),19) = D(i,j,k-1);
                if(i<N(1)+1)
                   Next(D(i,j,k),20) = D(i+1,j,k-1);
                else
                    Next(D(i,j,k),20) = nan;
                end
                if(i>1)
                   Next(D(i,j,k),21) = D(i-1,j,k-1);
                else
                    Next(D(i,j,k),21) = nan;
                end
                if(i<N(1)+1 && j < N(2)+1)
                   Next(D(i,j,k),22) = D(i+1,j+1,k-1);
                else
                    Next(D(i,j,k),22) = nan;
                end
                if(i<N(1)+1 && j > 1)
                   Next(D(i,j,k),23) = D(i+1,j-1,k-1);
                else
                    Next(D(i,j,k),23) = nan;
                end
                if(j < N(2)+1)
                   Next(D(i,j,k),24) = D(i,j+1,k-1);
                else
```

```
                        Next(D(i,j,k),24) = nan;
                    end
                    if(j>1)
                       Next(D(i,j,k),25) = D(i,j-1,k-1);
                    else
                        Next(D(i,j,k),25) = nan;
                    end
                    if(i>1 && j < N(2)+1)
                       Next(D(i,j,k),26) = D(i-1,j+1,k-1);
                    else
                        Next(D(i,j,k),26) = nan;
                    end
                    if(i>1 && j>1)
                       Next(D(i,j,k),27) = D(i-1,j-1,k-1);
                    else
                        Next(D(i,j,k),27) = nan;
                    end
                else
                    Next(D(i,j,k),19) = nan;
                    Next(D(i,j,k),20) = nan;
                    Next(D(i,j,k),21) = nan;
                    Next(D(i,j,k),22) = nan;
                    Next(D(i,j,k),23) = nan;
                    Next(D(i,j,k),24) = nan;
                    Next(D(i,j,k),25) = nan;
                    Next(D(i,j,k),26) = nan;
                    Next(D(i,j,k),27) = nan;
                end
            end
        end
    end
return

function B = AdjacentType()%CHECKED
    B = zeros(27,27,16);
    %CHECKED
    B(1,1,:) = [2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(1,2,:) = [2 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(1,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(1,4,:) = [2 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(1,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(1,6,:) = [2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(1,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(1,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
 B(1,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
 B(1,10,:) = [2 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(1,11,:) = [2 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(1,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(1,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(1,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(1,15,:) = [2 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(1,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(1,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(1,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down%CHECKED
 B(1,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(1,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(1,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(1,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(1,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(1,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(1,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(1,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(1,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %................................................................................

 %CHECKED
 B(2,1,:) = [1 1 2 2 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
 B(2,2,:) = [2 3 1 4 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(2,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(2,4,:) = [2 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(2,5,:) = [1 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(2,6,:) = [2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(2,7,:) = [1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(2,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(2,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
```

```
    B(2,10,:) = [2 6 1 5 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(2,11,:) = [2 7 1 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(2,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(2,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(2,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(2,15,:) = [2 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(2,16,:) = [1 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(2,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(2,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down%CHECKED
    B(2,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(2,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(2,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(2,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(2,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(2,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(2,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(2,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(2,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %.............................................................................................
    %CHECKED
    B(3,1,:) = [1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(3,2,:) = [1 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(3,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(3,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(3,5,:) = [1 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(3,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(3,7,:) = [1 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(3,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(3,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up%CHECKED
    B(3,10,:) = [1 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(3,11,:) = [1 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(3,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
```

```
    B(3,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(3,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(3,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(3,16,:) = [1 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(3,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(3,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down%CHECKED
    B(3,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(3,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(3,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(3,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(3,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(3,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(3,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(3,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(3,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %............................................................................

    %CHECKED
    B(4,1,:) = [2 2 6 6 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(4,2,:) = [2 3 6 7 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
    B(4,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(4,4,:) = [2 4 6 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(4,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(4,6,:) = [2 1 6 5 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(4,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(4,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(4,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up%CHECKED
    B(4,10,:) = [2 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(4,11,:) = [2 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(4,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(4,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
```

```
    B(4,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(4,15,:) = [2 5 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(4,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(4,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(4,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down%CHECKED
    B(4,19,:) = [6 2 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(4,20,:) = [6 3 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(4,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(4,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(4,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(4,24,:) = [6 1 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(4,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(4,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(4,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %
    ...............................................................................

    %CHECKED
    B(5,1,:) = [1 1 2 2 5 5 6 6 0 0 0 0 0 0 0]; %Itself
    B(5,2,:) = [2 3 1 4 5 8 6 7 0 0 0 0 0 0 0];%Forward
    B(5,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(5,4,:) = [2 4 6 8 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(5,5,:) = [1 3 5 7 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(5,6,:) = [2 1 6 5 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(5,7,:) = [1 2 5 6 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(5,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(5,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up%CHECKED
    B(5,10,:) = [2 6 1 5 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(5,11,:) = [2 7 1 8 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(5,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(5,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(5,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(5,15,:) = [2 5 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
```

```
 B(5,16,:) = [1 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(5,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(5,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down%CHECKED
 B(5,19,:) = [5 1 6 2 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(5,20,:) = [5 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(5,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(5,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(5,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(5,24,:) = [6 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(5,25,:) = [5 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(5,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(5,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %
.........................................................................................................

 %CHECKED
 B(6,1,:) = [1 1 5 5 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
 B(6,2,:) = [1 4 5 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(6,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(6,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(6,5,:) = [1 3 5 7 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(6,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(6,7,:) = [1 2 5 6 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(6,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(6,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
 B(6,10,:) = [1 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(6,11,:) = [1 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(6,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(6,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(6,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(6,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(6,16,:) = [1 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(6,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
B(6,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down%CHECKED
B(6,19,:) = [5 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(6,20,:) = [5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(6,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(6,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(6,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(6,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(6,25,:) = [5 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(6,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(6,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%
..............................................................................
%CHECKED
B(7,1,:) = [6 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(7,2,:) = [6 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
B(7,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(7,4,:) = [6 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(7,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(7,6,:) = [6 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(7,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(7,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(7,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up%CHECKED
B(7,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(7,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(7,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(7,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(7,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(7,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(7,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(7,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(7,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down%CHECKED
```

```
    B(7,19,:) = [6 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(7,20,:) = [6 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(7,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(7,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(7,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(7,24,:) = [6 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(7,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(7,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(7,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %.....................................................................................

    %CHECKED
    B(8,1,:) = [5 5 6 6 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(8,2,:) = [6 7 5 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(8,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(8,4,:) = [6 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(8,5,:) = [5 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(8,6,:) = [6 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(8,7,:) = [5 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(8,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(8,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up%CHECKED
    B(8,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(8,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(8,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(8,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(8,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(8,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(8,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(8,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(8,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down%CHECKED
    B(8,19,:) = [5 1 6 2 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(8,20,:) = [5 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(8,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
```

```
 B(8,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(8,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(8,24,:) = [6 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(8,25,:) = [5 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(8,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(8,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %
.......................................................................................
 %CHECKED
 B(9,1,:) = [5 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
 B(9,2,:) = [5 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(9,3,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(9,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(9,5,:) = [5 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(9,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(9,7,:) = [5 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(9,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(9,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
 B(9,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(9,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(9,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(9,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(9,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(9,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(9,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(9,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(9,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down%CHECKED
 B(9,19,:) = [5 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(9,20,:) = [5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(9,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(9,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
```

```
 B(9,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(9,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(9,25,:) = [5 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(9,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(9,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

 %CHECKED
 B(10,1,:) = [2 2 3 3 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
 B(10,2,:) = [2 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
 B(10,3,:) = [3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(10,4,:) = [2 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(10,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(10,6,:) = [2 1 3 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(10,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(10,8,:) = [3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(10,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
 B(10,10,:) = [2 6 3 7 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(10,11,:) = [2 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(10,12,:) = [3 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(10,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(10,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(10,15,:) = [2 5 3 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(10,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(10,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(10,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down%CHECKED
 B(10,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(10,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(10,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(10,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(10,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(10,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
```

```
 B(10,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(10,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(10,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %.........................................................................................
 %CHECKED
 B(11,1,:) = [1 1 2 2 3 3 4 4 0 0 0 0 0 0 0 0]; %Itself
 B(11,2,:) = [2 3 1 4 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(11,3,:) = [3 2 4 1 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(11,4,:) = [2 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(11,5,:) = [1 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(11,6,:) = [2 1 3 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(11,7,:) = [1 2 4 3 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(11,8,:) = [3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(11,9,:) = [4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up%CHECKED
 B(11,10,:) = [2 6 1 5 3 7 4 8 0 0 0 0 0 0 0 0];%Itself
 B(11,11,:) = [2 7 1 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(11,12,:) = [3 6 4 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(11,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(11,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(11,15,:) = [2 5 3 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(11,16,:) = [1 6 4 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(11,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(11,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down%CHECKED
 B(11,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(11,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(11,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(11,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(11,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(11,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(11,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(11,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
B(11,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%...................................................................................

%CHECKED
B(12,1,:) = [1 1 4 4 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(12,2,:) = [1 4 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(12,3,:) = [4 1 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(12,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(12,5,:) = [1 3 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(12,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(12,7,:) = [1 2 4 3 0 0 0 0 0 0 0 0 0 0 0];%Right
B(12,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(12,9,:) = [4 2 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up%CHECKED
B(12,10,:) = [1 5 4 8 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(12,11,:) = [1 8 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(12,12,:) = [4 5 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(12,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(12,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(12,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(12,16,:) = [1 6 4 7 0 0 0 0 0 0 0 0 0 0 0];%Right
B(12,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(12,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down%CHECKED
B(12,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(12,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(12,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(12,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(12,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(12,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(12,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(12,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(12,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
```

```
%.................................................................
%CHECKED
B(13,1,:) = [2 2 3 3 6 6 7 7 0 0 0 0 0 0 0 0]; %Itself
B(13,2,:) = [2 3 6 7 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
B(13,3,:) = [3 2 7 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(13,4,:) = [2 4 6 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(13,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(13,6,:) = [2 1 6 5 3 4 7 8 0 0 0 0 0 0 0 0];%Left
B(13,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(13,8,:) = [3 1 7 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(13,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up%CHECKED
B(13,10,:) = [2 6 3 7 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(13,11,:) = [2 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(13,12,:) = [3 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(13,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(13,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(13,15,:) = [2 5 3 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(13,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(13,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(13,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down%CHECKED
B(13,19,:) = [6 2 7 3 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(13,20,:) = [6 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(13,21,:) = [7 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(13,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(13,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(13,24,:) = [6 1 7 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(13,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(13,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(13,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%.................................................................
```

```
%CHECKED
B(14,1,:) = [1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8]; %Itself
B(14,2,:) = [1 4 2 3 5 8 6 7 0 0 0 0 0 0 0 0];%Forward
B(14,3,:) = [4 1 3 2 8 5 7 6 0 0 0 0 0 0 0 0];%Backward
B(14,4,:) = [2 4 6 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(14,5,:) = [1 3 5 7 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(14,6,:) = [2 1 6 5 3 4 7 8 0 0 0 0 0 0 0 0];%Left
B(14,7,:) = [1 2 5 6 4 3 8 7 0 0 0 0 0 0 0 0];%Right
B(14,8,:) = [3 1 7 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(14,9,:) = [4 2 8 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up%CHECKED
B(14,10,:) = [1 5 2 6 3 7 4 8 0 0 0 0 0 0 0 0];%Itself
B(14,11,:) = [2 7 1 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(14,12,:) = [3 6 4 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(14,13,:) = [2 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(14,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(14,15,:) = [2 5 3 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(14,16,:) = [1 6 4 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(14,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(14,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down%CHECKED
B(14,19,:) = [5 1 6 2 7 3 8 4 0 0 0 0 0 0 0 0];%Itself
B(14,20,:) = [5 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(14,21,:) = [8 1 7 2 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(14,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(14,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(14,24,:) = [6 1 7 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(14,25,:) = [5 2 8 3 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(14,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(14,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%...................................................................................
B(15,1,:) = [1 1 4 4 5 5 8 8 0 0 0 0 0 0 0 0]; %Itself
B(15,2,:) = [1 4 5 8 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(15,3,:) = [4 1 8 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
```

```
B(15,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(15,5,:) = [1 3 5 7 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(15,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(15,7,:) = [1 2 5 6 4 3 8 7 0 0 0 0 0 0 0 0];%Right
B(15,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(15,9,:) = [4 2 8 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up
B(15,10,:) = [1 5 4 8 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(15,11,:) = [1 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(15,12,:) = [4 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(15,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(15,14,:) = [1 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(15,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(15,16,:) = [1 6 4 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(15,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(15,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down
B(15,19,:) = [5 1 8 4 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(15,20,:) = [5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(15,21,:) = [8 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(15,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(15,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(15,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(15,25,:) = [5 2 8 3 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(15,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(15,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%..............................................................................

B(16,1,:) = [6 6 7 7 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(16,2,:) = [6 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
B(16,3,:) = [7 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(16,4,:) = [6 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(16,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
```

```
    B(16,6,:) = [6 5 7 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(16,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(16,8,:) = [7 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(16,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(16,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(16,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(16,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(16,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(16,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(16,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(16,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(16,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(16,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(16,19,:) = [6 2 7 3 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(16,20,:) = [6 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(16,21,:) = [7 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(16,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(16,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(16,24,:) = [6 1 7 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(16,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(16,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(16,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %.........................................................................

    B(17,1,:) = [5 5 6 6 7 7 8 8 0 0 0 0 0 0 0 0]; %Itself
    B(17,2,:) = [5 8 6 7 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(17,3,:) = [8 5 7 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(17,4,:) = [6 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(17,5,:) = [5 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(17,6,:) = [6 5 7 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(17,7,:) = [5 6 8 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(17,8,:) = [7 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
B(17,9,:) = [8 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up
B(17,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(17,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(17,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(17,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(17,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(17,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(17,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(17,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(17,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down
B(17,19,:) = [5 1 6 2 7 3 8 4 0 0 0 0 0 0 0 0];%Itself
B(17,20,:) = [5 4 6 3 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(17,21,:) = [8 1 7 2 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(17,22,:) = [6 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(17,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(17,24,:) = [6 1 7 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(17,25,:) = [5 2 8 3 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(17,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(17,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%.......................................................................................

B(18,1,:) = [5 5 8 8 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(18,2,:) = [5 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(18,3,:) = [8 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(18,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(18,5,:) = [5 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(18,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(18,7,:) = [5 6 8 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(18,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(18,9,:) = [8 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up
B(18,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
```

```
  B(18,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
  B(18,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
  B(18,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
  B(18,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
  B(18,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
  B(18,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
  B(18,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
  B(18,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
  %Down
  B(18,19,:) = [5 1 8 4 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
  B(18,20,:) = [5 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
  B(18,21,:) = [8 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
  B(18,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
  B(18,23,:) = [5 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
  B(18,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
  B(18,25,:) = [5 2 8 3 0 0 0 0 0 0 0 0 0 0 0 0];%Right
  B(18,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
  B(18,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
  %>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

  B(19,1,:) = [3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
  B(19,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
  B(19,3,:) = [3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
  B(19,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
  B(19,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
  B(19,6,:) = [3 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
  B(19,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
  B(19,8,:) = [3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
  B(19,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
  %Up
  B(19,10,:) = [3 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
  B(19,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
  B(19,12,:) = [3 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
```

```
 B(19,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(19,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(19,15,:) = [3 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(19,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(19,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(19,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Down
 B(19,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(19,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(19,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(19,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(19,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(19,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(19,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(19,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(19,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %.............................................................................

 B(20,1,:) = [3 3 4 4 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
 B(20,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(20,3,:) = [3 2 4 1 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(20,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
 B(20,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
 B(20,6,:) = [3 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
 B(20,7,:) = [4 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
 B(20,8,:) = [3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
 B(20,9,:) = [4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
 %Up
 B(20,10,:) = [4 8 3 7 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
 B(20,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
 B(20,12,:) = [3 6 4 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
 B(20,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
```

```
B(20,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Right
B(20,15,:) = [3 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(20,16,:) = [4 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(20,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Left
B(20,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Right
%Down
B(20,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(20,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(20,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(20,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Left
B(20,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Right
B(20,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(20,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(20,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Left
B(20,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Right
%
...........................................................................................


B(21,1,:) = [4 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(21,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(21,3,:) = [4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(21,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Left
B(21,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Right
B(21,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(21,7,:) = [4 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(21,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Left
B(21,9,:) = [4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward +
Right
%Up
B(21,10,:) = [4 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(21,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(21,12,:) = [4 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(21,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Left
B(21,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward +
Right
B(21,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
```

```
    B(21,16,:) = [4 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(21,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(21,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(21,19,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(21,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(21,21,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(21,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(21,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(21,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(21,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(21,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(21,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %...............................................................................

    B(22,1,:) = [3 3 7 7 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(22,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
    B(22,3,:) = [3 2 7 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(22,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(22,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(22,6,:) = [7 8 3 4 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(22,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(22,8,:) = [3 1 7 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(22,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(22,10,:) = [3 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(22,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(22,12,:) = [3 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(22,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(22,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(22,15,:) = [3 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(22,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(22,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
    B(22,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(22,19,:) = [7 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(22,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(22,21,:) = [7 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(22,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(22,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(22,24,:) = [7 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(22,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(22,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(22,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %............................................................................

    B(23,1,:) = [3 3 4 4 7 7 8 8 0 0 0 0 0 0 0 0]; %Itself
    B(23,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(23,3,:) = [4 1 3 2 8 5 7 6 0 0 0 0 0 0 0 0];%Backward
    B(23,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(23,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(23,6,:) = [3 4 7 8 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(23,7,:) = [4 3 8 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(23,8,:) = [3 1 7 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(23,9,:) = [4 2 8 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(23,10,:) = [4 8 3 7 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(23,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(23,12,:) = [3 6 4 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(23,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(23,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(23,15,:) = [3 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(23,16,:) = [4 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(23,17,:) = [3 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(23,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
```

```
    B(23,19,:) = [8 4 7 3 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(23,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(23,21,:) = [8 1 7 2 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(23,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(23,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(23,24,:) = [7 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(23,25,:) = [8 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(23,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(23,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %....................................................................................

    B(24,1,:) = [4 4 8 8 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(24,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(24,3,:) = [4 1 8 5 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(24,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(24,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(24,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(24,7,:) = [4 3 8 7 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(24,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(24,9,:) = [4 2 8 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(24,10,:) = [4 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(24,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(24,12,:) = [4 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(24,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(24,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(24,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(24,16,:) = [4 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(24,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(24,18,:) = [4 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(24,19,:) = [8 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(24,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(24,21,:) = [8 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
```

```
    B(24,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(24,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(24,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(24,25,:) = [8 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(24,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(24,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %...............................................................................
    B(25,1,:) = [7 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(25,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Forward
    B(25,3,:) = [7 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(25,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(25,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(25,6,:) = [7 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(25,7,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(25,8,:) = [7 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(25,9,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(25,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(25,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(25,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(25,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(25,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(25,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(25,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(25,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(25,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(25,19,:) = [7 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(25,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(25,21,:) = [7 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(25,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(25,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
```

```
B(25,24,:) = [7 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(25,25,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(25,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(25,27,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%............................................................................................
B(26,1,:) = [7 7 8 8 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
B(26,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(26,3,:) = [8 5 7 6 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(26,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(26,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(26,6,:) = [7 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(26,7,:) = [8 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(26,8,:) = [7 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(26,9,:) = [8 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Up
B(26,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(26,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(26,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(26,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(26,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(26,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(26,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(26,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
B(26,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
%Down
B(26,19,:) = [7 3 8 4 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
B(26,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
B(26,21,:) = [8 1 7 2 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
B(26,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
B(26,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
B(26,24,:) = [7 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
B(26,25,:) = [8 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
B(26,26,:) = [7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
```

```
    B(26,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %..........................................................................................
    B(27,1,:) = [8 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0]; %Itself
    B(27,2,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(27,3,:) = [8 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(27,4,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(27,5,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(27,6,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(27,7,:) = [8 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(27,8,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(27,9,:) = [8 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Up
    B(27,10,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(27,11,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(27,12,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(27,13,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(27,14,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(27,15,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(27,16,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(27,17,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(27,18,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
    %Down
    B(27,19,:) = [8 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Itself
    B(27,20,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward
    B(27,21,:) = [8 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward
    B(27,22,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Left
    B(27,23,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Forward + Right
    B(27,24,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Left
    B(27,25,:) = [8 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Right
    B(27,26,:) = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Left
    B(27,27,:) = [8 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0];%Backward + Right
return

```

```
function [K11,K12,K13,K22,K23,K33,M] = SmallMatrix(Delta)
%{

syms x
syms y
syms z
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(10) '%']);
Q = zeros(64,1)*x;
icount = 1;
for i = 0:3
    for j = 0:3
        for k = 0:3
             Q(icount,1) = x^i*y^j*z^k;
             icount = icount + 1;
        end
    end
end
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(20) '%']);
K11 = zeros(size(Q,1))*x*y*z;
K12 = zeros(size(Q,1))*x*y*z;
K13 = zeros(size(Q,1))*x*y*z;
K22 = zeros(size(Q,1))*x*y*z;
K23 = zeros(size(Q,1))*x*y*z;
K33 = zeros(size(Q,1))*x*y*z;
M   = zeros(size(Q,1))*x*y*z;
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(25) '%']);
for i = 1:size(Q,1)
    for j = 1:size(Q,1)
        K11(j,i) = diff(Q(j),x)*diff(Q(i),x);
        K12(j,i) = diff(Q(j),y)*diff(Q(i),x);
        K13(j,i) = diff(Q(j),z)*diff(Q(i),x);
        K22(j,i) = diff(Q(j),y)*diff(Q(i),y);
        K23(j,i) = diff(Q(j),z)*diff(Q(i),y);
        K33(j,i) = diff(Q(j),z)*diff(Q(i),z);
        M(j,i)   = Q(j)*Q(i);
    end
end
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(30) '%']);
K11 = int(int(int(K11,x,[0,1]),y,[0,1]),z,[0,1]);
K12 = int(int(int(K12,x,[0,1]),y,[0,1]),z,[0,1]);
K13 = int(int(int(K13,x,[0,1]),y,[0,1]),z,[0,1]);
K22 = int(int(int(K22,x,[0,1]),y,[0,1]),z,[0,1]);
K23 = int(int(int(K23,x,[0,1]),y,[0,1]),z,[0,1]);
K33 = int(int(int(K33,x,[0,1]),y,[0,1]),z,[0,1]);
M   = int(int(int(M,x,[0,1]),y,[0,1]),z,[0,1]);
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(35) '%']);
T = MATRIX_T(Q);
Tinv = inv(T);
```

```
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(40) '%']);
K11 = (Tinv)'*K11*Tinv;
K12 = (Tinv)'*K12*Tinv;
K13 = (Tinv)'*K13*Tinv;
K22 = (Tinv)'*K22*Tinv;
K23 = (Tinv)'*K23*Tinv;
K33 = (Tinv)'*K33*Tinv;
M   = (Tinv)'*M*Tinv;
mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(45) '%']);

save('matrices.mat','K11','K12','K13','K22','K23','K33','M');
%}
L = load('matrices.mat');
K11 = L.K11;
K12 = L.K12;
K13 = L.K13;
K22 = L.K22;
K23 = L.K23;
K33 = L.K33;
M = L.M;


K11 = double(K11*Delta(2)*Delta(3)/Delta(1));
K12 = double(K12*Delta(3));
K13 = double(K13*Delta(2));
K22 = double(K22*Delta(1)*Delta(3)/Delta(2));
K23 = double(K23*Delta(1));
K33 = double(K33*Delta(1)*Delta(2)/Delta(3));
M   = double(M*Delta(1)*Delta(2)*Delta(3));
%mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(50) '%']);
return

function T = MATRIX_T(Q)
%{
syms x;
syms y;
syms z;

n = size(Q,1);
T = zeros(n);
for j = 1:n
   T(j,1) = subs(Q(j),[x,y,z],[0,1,0]);
   T(j,2) = subs(Q(j),[x,y,z],[0,0,0]);
   T(j,3) = subs(Q(j),[x,y,z],[1,0,0]);
   T(j,4) = subs(Q(j),[x,y,z],[1,1,0]);
   T(j,5) = subs(Q(j),[x,y,z],[0,1,1]);
   T(j,6) = subs(Q(j),[x,y,z],[0,0,1]);
   T(j,7) = subs(Q(j),[x,y,z],[1,0,1]);
```

```
    T(j,8) = subs(Q(j),[x,y,z],[1,1,1]);

    T(j,9) = subs(diff(Q(j),x),[x,y,z],[0,1,0]);
    T(j,10) = subs(diff(Q(j),x),[x,y,z],[0,0,0]);
    T(j,11) = subs(diff(Q(j),x),[x,y,z],[1,0,0]);
    T(j,12) = subs(diff(Q(j),x),[x,y,z],[1,1,0]);
    T(j,13) = subs(diff(Q(j),x),[x,y,z],[0,1,1]);
    T(j,14) = subs(diff(Q(j),x),[x,y,z],[0,0,1]);
    T(j,15) = subs(diff(Q(j),x),[x,y,z],[1,0,1]);
    T(j,16) = subs(diff(Q(j),x),[x,y,z],[1,1,1]);

    T(j,17) = subs(diff(Q(j),y),[x,y,z],[0,1,0]);
    T(j,18) = subs(diff(Q(j),y),[x,y,z],[0,0,0]);
    T(j,19) = subs(diff(Q(j),y),[x,y,z],[1,0,0]);
    T(j,20) = subs(diff(Q(j),y),[x,y,z],[1,1,0]);
    T(j,21) = subs(diff(Q(j),y),[x,y,z],[0,1,1]);
    T(j,22) = subs(diff(Q(j),y),[x,y,z],[0,0,1]);
    T(j,23) = subs(diff(Q(j),y),[x,y,z],[1,0,1]);
    T(j,24) = subs(diff(Q(j),y),[x,y,z],[1,1,1]);

    T(j,25) = subs(diff(Q(j),z),[x,y,z],[0,1,0]);
    T(j,26) = subs(diff(Q(j),z),[x,y,z],[0,0,0]);
    T(j,27) = subs(diff(Q(j),z),[x,y,z],[1,0,0]);
    T(j,28) = subs(diff(Q(j),z),[x,y,z],[1,1,0]);
    T(j,29) = subs(diff(Q(j),z),[x,y,z],[0,1,1]);
    T(j,30) = subs(diff(Q(j),z),[x,y,z],[0,0,1]);
    T(j,31) = subs(diff(Q(j),z),[x,y,z],[1,0,1]);
    T(j,32) = subs(diff(Q(j),z),[x,y,z],[1,1,1]);

    T(j,33) = subs(diff(diff(Q(j),x),y),[x,y,z],[0,1,0]);
    T(j,34) = subs(diff(diff(Q(j),x),y),[x,y,z],[0,0,0]);
    T(j,35) = subs(diff(diff(Q(j),x),y),[x,y,z],[1,0,0]);
    T(j,36) = subs(diff(diff(Q(j),x),y),[x,y,z],[1,1,0]);
    T(j,37) = subs(diff(diff(Q(j),x),y),[x,y,z],[0,1,1]);
    T(j,38) = subs(diff(diff(Q(j),x),y),[x,y,z],[0,0,1]);
    T(j,39) = subs(diff(diff(Q(j),x),y),[x,y,z],[1,0,1]);
    T(j,40) = subs(diff(diff(Q(j),x),y),[x,y,z],[1,1,1]);

    T(j,41) = subs(diff(diff(Q(j),x),z),[x,y,z],[0,1,0]);
    T(j,42) = subs(diff(diff(Q(j),x),z),[x,y,z],[0,0,0]);
    T(j,43) = subs(diff(diff(Q(j),x),z),[x,y,z],[1,0,0]);
    T(j,44) = subs(diff(diff(Q(j),x),z),[x,y,z],[1,1,0]);
    T(j,45) = subs(diff(diff(Q(j),x),z),[x,y,z],[0,1,1]);
    T(j,46) = subs(diff(diff(Q(j),x),z),[x,y,z],[0,0,1]);
    T(j,47) = subs(diff(diff(Q(j),x),z),[x,y,z],[1,0,1]);
    T(j,48) = subs(diff(diff(Q(j),x),z),[x,y,z],[1,1,1]);

    T(j,49) = subs(diff(diff(Q(j),y),z),[x,y,z],[0,1,0]);
```

```
    T(j,50) = subs(diff(diff(Q(j),y),z),[x,y,z],[0,0,0]);
    T(j,51) = subs(diff(diff(Q(j),y),z),[x,y,z],[1,0,0]);
    T(j,52) = subs(diff(diff(Q(j),y),z),[x,y,z],[1,1,0]);
    T(j,53) = subs(diff(diff(Q(j),y),z),[x,y,z],[0,1,1]);
    T(j,54) = subs(diff(diff(Q(j),y),z),[x,y,z],[0,0,1]);
    T(j,55) = subs(diff(diff(Q(j),y),z),[x,y,z],[1,0,1]);
    T(j,56) = subs(diff(diff(Q(j),y),z),[x,y,z],[1,1,1]);

    T(j,57) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[0,1,0])
    ;
    T(j,58) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[0,0,0])
    ;
    T(j,59) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[1,0,0])
    ;
    T(j,60) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[1,1,0])
    ;
    T(j,61) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[0,1,1])
    ;
    T(j,62) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[0,0,1])
    ;
    T(j,63) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[1,0,1])
    ;
    T(j,64) = subs(diff(diff(diff(Q(j),x),z),y),[x,y,z],[1,1,1])
    ;
end
%}
T = [1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
     0 0 0 0 0;
 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1
    1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
     0 0 0;
 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2
    2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
     0 0 0;
 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3
    3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
     0 0 0;
 1 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
     0 0 0;
 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 0 0
    1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0
     0 0 0;
 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 2 0 0
    2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 0 0 0 0 0 0
     0 0 0;
```

```
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 3 0 0
  3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 0 0 0 0 0
   0 0 0;
1 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 2 0 0 2 2 0 0 2 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 2 1 0 0 1 1 0 0
  1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 2 2 0 0 2 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0 0 2 0 0
  2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 4 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 2 0 0 0 0 3 0 0
  3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 0 0 6 0 0 0 0 0
   0 0 0;
1 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 3 0 0 3 3 0 0 3 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 3 1 0 0 1 1 0 0
  1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 3 3 0 0 3 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 3 0 0 0 0 2 0 0
  2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 0 0 6 0 0 0 0 0
   0 0 0;
0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 3 0 0 0 0 3 0 0
  3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9 0 0 9 0 0 0 0 0
   0 0 0;
0 0 1 1 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1
  1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2
  2 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3
  3 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0
  0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 0 0 0
  1 0 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1
   1 1 1;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
  2 0 0 0 0 1 1 1 1 0 0 0 0 2 0 0 2 0 0 0 0 0 0 2 2 0 0 0 0 2
   2 2 2;
```

```
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
  3 0 0 0 0 1 1 1 1 0 0 0 0 3 0 0 3 0 0 0 0 0 0 3 3 0 0 0 0 3
   3 3 3;
0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 1 0 0 0 2 0 0 0 2 0 0 0 0 0 0 0
  0 2 0 0 2 2 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 2 0 0 0 1 0 0 0
  1 0 0 0 0 2 0 0 2 1 0 0 1 1 0 0 1 0 0 0 2 0 0 0 2 2 0 0 2 2
   0 0 2;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
  2 0 0 0 0 2 0 0 2 0 0 0 0 2 0 0 2 0 0 0 0 0 0 0 4 0 0 0 0 4
   0 0 4;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
  3 0 0 0 0 2 0 0 2 0 0 0 0 3 0 0 3 0 0 0 0 0 0 0 6 0 0 0 0 6
   0 0 6;
0 0 0 1 0 0 0 1 1 0 0 1 1 0 0 1 0 0 0 3 0 0 0 3 0 0 0 0 0 0 0
  0 3 0 0 3 3 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0
  1 0 0 0 0 3 0 0 3 1 0 0 1 1 0 0 1 0 0 0 3 0 0 0 3 3 0 0 3 3
   0 0 3;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0
  2 0 0 0 0 3 0 0 3 0 0 0 0 2 0 0 2 0 0 0 0 0 0 0 6 0 0 0 0 6
   0 0 6;
0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0
  3 0 0 0 0 3 0 0 3 0 0 0 0 3 0 0 3 0 0 0 0 0 0 0 9 0 0 0 0 9
   0 0 9;
0 0 1 1 0 0 1 1 0 0 2 2 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1
  1 0 0 0 0 0 0 0 0 0 0 2 2 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2
  2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 4 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3
  3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 6 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 1 0 0 0 1 0 0 0 2 0 0 0 2 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0
  0 0 0 2 2 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 1 1 0 0 0 1 0 0 0
  1 0 0 0 0 0 0 2 2 0 0 0 2 0 0 0 2 0 0 1 1 0 0 1 1 0 0 2 2 0
   0 2 2;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
  2 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 4 0 0 0 0 0 0 2 2 0 0 0 0 0
   0 4 4;
```

```
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
  3 0 0 0 0 0 0 2 2 0 0 0 0 0 0 0 6 0 0 0 0 0 0 3 3 0 0 0 0 0
   0 6 6;
0 0 0 1 0 0 0 1 0 0 0 2 0 0 0 2 0 0 0 2 0 0 0 2 0 0 0 0 0 0 0
  0 0 0 0 4 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 1 0 0 0
  1 0 0 0 0 0 0 0 4 0 0 0 2 0 0 0 2 0 0 0 2 0 0 0 2 0 0 0 4 0
   0 0 4;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
  2 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 4 0 0 0 0 0
   0 0 8;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0
  3 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 6 0 0 0 0 0
   0 0 12;
0 0 0 1 0 0 0 1 0 0 0 2 0 0 0 2 0 0 0 3 0 0 0 3 0 0 0 0 0 0 0
  0 0 0 0 6 0 0 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0
  1 0 0 0 0 0 0 0 6 0 0 0 2 0 0 0 2 0 0 0 3 0 0 0 3 0 0 0 6 0
   0 0 6;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0
  2 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 6 0 0 0 0 0
   0 0 12;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0
  3 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 9 0 0 0 0 0
   0 0 18;
0 0 1 1 0 0 1 1 0 0 3 3 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1
  1 0 0 0 0 0 0 0 0 0 0 3 3 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2
  2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 6 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 1 1 0 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3
  3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9 9 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 1 0 0 0 1 0 0 0 3 0 0 0 3 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0
  0 0 0 3 3 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
   0 0 0;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 1 1 0 0 0 1 0 0 0
  1 0 0 0 0 0 0 3 3 0 0 0 3 0 0 0 3 0 0 1 1 0 0 1 1 0 0 3 3 0
   0 3 3;
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0
  2 0 0 0 0 0 0 3 3 0 0 0 0 0 0 0 6 0 0 0 0 0 0 2 2 0 0 0 0 0
   0 6 6;
```

```
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 3 3 0 0 0 0 0 0 0 9 0 0 0 0 0 0 3 3 0 0 0 0 0 0 9 9;
 0 0 0 1 0 0 0 1 0 0 0 3 0 0 0 3 0 0 0 2 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 6 0 0 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 2 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 6 0 0 0 3 0 0 0 3 0 0 0 2 0 0 0 2 0 0 0 6 0 0 0 6;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 12;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 18;
 0 0 0 1 0 0 0 1 0 0 0 3 0 0 0 3 0 0 0 3 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 9 0 0 0 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 3 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 9 0 0 0 3 0 0 0 3 0 0 0 3 0 0 0 3 0 0 0 9 0 0 0 9;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0 18;
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 9 0 0 0 0 0 0 0 27];
T = T';
return;

function [B11,B12,B13,B22,B23,B33,BM] = AddMatrix(K11,K12,K13,K22,K23,K33,M)
B = AdjacentType();
B11 = zeros(216,216);
B12 = zeros(216,216);
B13 = zeros(216,216);
B22 = zeros(216,216);
B23 = zeros(216,216);
B33 = zeros(216,216);
BM = zeros(216,216);
for g = 1:8
    for f = 1:8
        for i = 1:27
            for j = 1:27
                for k = 1:2:15
                    if B(i,j,k) ~= 0
                        B11(i+(g-1)*27,j+(f-1)*27) = B11(i+(g-1)*27,j+(f-1)*27) + K11(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
```

```
                         B12(i+(g-1)*27,j+(f-1)*27) = B12(i+(g-1)
    *27,j+(f-1)*27) + K12(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                         B13(i+(g-1)*27,j+(f-1)*27) = B13(i+(g-1)
    *27,j+(f-1)*27) + K13(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                         B22(i+(g-1)*27,j+(f-1)*27) = B22(i+(g-1)
    *27,j+(f-1)*27) + K22(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                         B23(i+(g-1)*27,j+(f-1)*27) = B23(i+(g-1)
    *27,j+(f-1)*27) + K23(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                         B33(i+(g-1)*27,j+(f-1)*27) = B33(i+(g-1)
    *27,j+(f-1)*27) + K33(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                         BM(i+(g-1)*27,j+(f-1)*27) = BM(i+(g-1)*27,j
    +(f-1)*27) + M(B(i,j,k)+(g-1)*8,B(i,j,k+1)+(f-1)*8);
                     else
                         break
                     end
                end
            end
        end
    end
end
return;

function [K11,K12,K13,K22,K23,K33,M,D,E] = Matrices(Delta,N,
    method)
%mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(0) '%']);
[K11q,K12q,K13q,K22q,K23q,K33q,Mq] = SmallMatrix(Delta);
[B11,B12,B13,B22,B23,B33,BM] = AddMatrix(K11q,K12q,K13q,K22q,
    K23q,K33q,Mq);
%{
K11 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
K12 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
K13 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
K22 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
K23 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
K33 = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N
    (3)+1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
M = spalloc((N(1)+1)*(N(2)+1)*(N(3)+1),(N(1)+1)*(N(2)+1)*(N(3)
    +1),(N(1)+1)*(N(2)+1)*(N(3)+1)*27);
%}
[D,E] = Domain(N,Delta);
A = Adjacent(N,D);
T = Type(N,A);
%
```

```
%mwb.Update(2, 1, 0, ['Matrix Creation ' num2str(55) '%']);
%{
for i = 1:(N(1)+1)*(N(2)+1)*(N(3)+1)
    mwb.Update(2, 1, i/((N(1)+1)*(N(2)+1)*(N(3)+1)+1), ['Matrix Creation ' num2str(i/((N(1)+1)*(N(2)+1)*(N(3)+1)+1)*100) '%']);
    for j = 1:27
       k = 1;
       while (B(T(i),j,k) ~= 0 && ~isnan(A(i,j)))
            K11(i,A(i,j)) = K11(i,A(i,j)) + K11q(B(T(i),j,k),B(T(i),j,k+1));
            K12(i,A(i,j)) = K12(i,A(i,j)) + K12q(B(T(i),j,k),B(T(i),j,k+1));
            K13(i,A(i,j)) = K13(i,A(i,j)) + K13q(B(T(i),j,k),B(T(i),j,k+1));
            K22(i,A(i,j)) = K22(i,A(i,j)) + K22q(B(T(i),j,k),B(T(i),j,k+1));
            K23(i,A(i,j)) = K23(i,A(i,j)) + K23q(B(T(i),j,k),B(T(i),j,k+1));
            K33(i,A(i,j)) = K33(i,A(i,j)) + K33q(B(T(i),j,k),B(T(i),j,k+1));

            M(i,A(i,j)) = M(i,A(i,j)) + Mq(B(T(i),j,k),B(T(i),j,k+1));
            k = k + 2;
            if(k >= 16)
               break;
            end
       end
    end
end
%}
n = 0;
for i = 1:(N(1)+1)*(N(2)+1)*(N(3)+1)
    for j = 1:27
      if ~isnan(A(i,j))
         n = n +1;
      end
    end
end

%{

%}
%{

NAN_A = ~isnan(A);
```

```
A2 = repmat(A(:,1),1,size(A,2));
A3 = repmat([1:27]',1,size(A,1))';
iy = A(NAN_A);
ix = A2(NAN_A);
iz = A3(NAN_A);
[ix iy iz];

for i = 1:size(iy,1)
    mwb.Update(2, 1, i/(size(iy,1)), ['Matrix Creation ' num2str(i/size(iy,1)*100) '%']);
    k = 1;
    while(k < 16 && B(T(ix(i)),iz(i),k) ~= 0)
        K11s(i) = K11s(i) + K11q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        K12s(i) = K12s(i) + K12q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        K13s(i) = K13s(i) + K13q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        K22s(i) = K22s(i) + K22q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        K23s(i) = K23s(i) + K23q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        K33s(i) = K33s(i) + K33q(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        Ms(i) = Ms(i) + Mq(B(T(ix(i)),iz(i),k),B(T(ix(i)),iz(i),k+1));
        k = k + 2;
    end
end
%}
%%{

if (method == 2)
   % mwb.Update(2, 1, 0.1, ['Matrix Creation ' num2str(55) '%']);
    InvA = A';
    NAN_A = ~isnan(InvA);
    A2 = repmat([1:27]',1,size(InvA,2))';
    A3 = repmat(1:size(A,1),size(A,2),1)';
    InvA3 = A3';
    InvA2 = A2';
    iy1 = InvA(NAN_A);
    ix1 = InvA3(NAN_A);
    iz1 = InvA2(NAN_A);
    Typex1 = T(ix1);

    iy = [];
    ix = [];
```

```
    iz = [];
    Typex = [];


    for i = 1:8
       for j = 1:8
           iy = [iy; iy1+(j-1)*size(A,1)];
           iz = [iz; iz1+(j-1)*27];
           ix = [ix; ix1+(i-1)*size(A,1)];
           Typex = [Typex; Typex1+(i-1)*27];
       end
    end



    %BAdd = B(Typex,Typey,:);
    %mwb.Update(2, 1, 0.2, ['Matrix Creation ' num2str(60) '%']);
    K11s = B11(sub2ind(size(B11),Typex,iz));
    %mwb.Update(2, 1, 0.3, ['Matrix Creation ' num2str(65) '%']);
    K12s = B12(sub2ind(size(B12),Typex,iz));
    %mwb.Update(2, 1, 0.4, ['Matrix Creation ' num2str(70) '%']);
    K13s = B13(sub2ind(size(B13),Typex,iz));
    %mwb.Update(2, 1, 0.5, ['Matrix Creation ' num2str(75) '%']);
    K22s = B22(sub2ind(size(B22),Typex,iz));
    %mwb.Update(2, 1, 0.6, ['Matrix Creation ' num2str(80) '%']);
    K23s = B23(sub2ind(size(B23),Typex,iz));
    %mwb.Update(2, 1, 0.7, ['Matrix Creation ' num2str(85) '%']);
    K33s = B33(sub2ind(size(B33),Typex,iz));
    %mwb.Update(2, 1, 0.8, ['Matrix Creation ' num2str(90) '%']);
    Ms = BM(sub2ind(size(BM),Typex,iz));
    %mwb.Update(2, 1, 0.9, ['Matrix Creation ' num2str(95) '%']);
elseif (method == 1)
    B = AdjacentType();
    K11s = zeros(n,1);
    K12s = zeros(n,1);
    K13s = zeros(n,1);
    K22s = zeros(n,1);
    K23s = zeros(n,1);
    K33s = zeros(n,1);
    Ms = zeros(n,1);
```

```
    ix = zeros(n,1);
    iy = zeros(n,1);
    ii = 1;
    for i = 1:(N(1)+1)*(N(2)+1)*(N(3)+1)
        %mwb.Update(2, 1, i/((N(1)+1)*(N(2)+1)*(N(3)+1)+1), ['
    Matrix Creation ' num2str(i/((N(1)+1)*(N(2)+1)*(N(3)+1)+1)
    *100) '%']);
        for j = 1:27
          if ~isnan(A(i,j))
              ix(ii) = A(i,1);
              iy(ii) = A(i,j);
              k = 1;
              while (B(T(i),j,k) ~= 0 && ~isnan(A(i,j)))

                  K11s(ii) = K11s(ii) + K11q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  K12s(ii) = K12s(ii) + K12q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  K13s(ii) = K13s(ii) + K13q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  K22s(ii) = K22s(ii) + K22q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  K23s(ii) = K23s(ii) + K23q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  K33s(ii) = K33s(ii) + K33q(B(T(i),j,k),B(T(i),j
    ,k+1));
                  Ms(ii) = Ms(ii) + Mq(B(T(i),j,k),B(T(i),j,k+1))
    ;
                  k = k + 2;
                  if(k >= 16)
                      break;
                  end
              end
              Tempi(ii) = T(i);
                  Tempj(ii) = j;
              ii = ii +1;
          end
        end
    end
end

%}
K11 = sparse(ix,iy,K11s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)
    *(N(2)+1)*(N(3)+1));
K12 = sparse(ix,iy,K12s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)
    *(N(2)+1)*(N(3)+1));
K13 = sparse(ix,iy,K13s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)
    *(N(2)+1)*(N(3)+1));
```

```
K22 = sparse(ix,iy,K22s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)*(N(2)+1)*(N(3)+1));
K23 = sparse(ix,iy,K23s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)*(N(2)+1)*(N(3)+1));
K33 = sparse(ix,iy,K33s,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)*(N(2)+1)*(N(3)+1));
M = sparse(ix,iy,Ms,8*(N(1)+1)*(N(2)+1)*(N(3)+1),8*(N(1)+1)*(N(2)+1)*(N(3)+1));
%{
B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,2:2:16)
K11q(B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,1:2:15),B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,2:2:16))

sum(K11q(B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,1:2:15)>0,B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,2:2:16))>0)
K11(1:(N(1)+1)*(N(2)+1)*(N(3)+1),1:(N(1)+1)*(N(2)+1)*(N(3)+1)) = sum(B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,1:2:15))
K11(1,:)
size(K11)
K11(1:(N(1)+1)*(N(2)+1)*(N(3)+1),A(~isnan(A(1:(N(1)+1)*(N(2)+1)*(N(3)+1),1:27)))) =  sum(K11q(B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,1:2:15)>0,B(T(1:(N(1)+1)*(N(2)+1)*(N(3)+1)),1:27,2:2:1))>0);
%}

%mwb.Update(2, 1, 1, ['Matrix Creation ' num2str(100) '%']);
return;

function T = Type(N,A)
T = zeros((N(1)+1)*(N(2)+1)*(N(3)+1),1);
TEST = [1  10  nan  11  nan  2  nan  nan  nan  4  13  nan  14  nan  5  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan;
        2  11  nan  12  10  3  1  nan  nan  5  14  nan  15  13  6  4  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan;
        3  12  nan  nan  11  nan  2  nan  nan  6  15  nan  nan  14  nan  5  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan  nan;
        4  13  nan  14  nan  5  nan  nan  nan  7  16  nan  17  nan  8  nan  nan  nan  1  10  nan  11  nan  2  nan  nan  nan;
        5  14  nan  15  13  6  4  nan  nan  8  17  nan  18  16  9  7  nan  nan  2  11  nan  12  10  3  1  nan  nan;
        6  15  nan  nan  14  nan  5  nan  nan  9  18  nan  nan  17  nan  8  nan  nan  3  12  nan  nan  11  nan  2  nan  nan;
```

```
     7    16   nan    17   nan     8   nan   nan   nan   nan
   nan   nan   nan   nan   nan   nan   nan   nan     4
13   nan    14   nan     5   nan   nan   nan;
     8    17   nan    18    16     9     7   nan   nan   nan
   nan   nan   nan   nan   nan   nan   nan   nan     5
14   nan    15    13     6     4   nan   nan;
     9    18   nan   nan    17   nan     8   nan   nan   nan
   nan   nan   nan   nan   nan   nan   nan   nan     6
15   nan   nan    14   nan     5   nan   nan;
    10    19     1    20   nan    11   nan     2   nan
13    22     4    23   nan    14   nan     5   nan   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    11    20     2    21    19    12    10     3     1
14    23     5    24    22    15    13     6     4   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    12    21     3   nan    20   nan    11   nan     2
15    24     6   nan    23   nan    14   nan     5   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    13    22     4    23   nan    14   nan     5   nan
16    25     7    26   nan    17   nan     8   nan    10
 19     1    20   nan    11   nan     2   nan;
    14    23     5    24    22    15    13     6     4
17    26     8    27    25    18    16     9     7    11
 20     2    21    19    12    10     3     1;
    15    24     6   nan    23   nan    14   nan     5
18    27     9   nan    26   nan    17   nan     8    12
 21     3   nan    20   nan    11   nan     2;
    16    25     7    26   nan    17   nan     8   nan
nan   nan   nan   nan   nan   nan   nan   nan   nan    13
  22     4    23   nan    14   nan     5   nan;
    17    26     8    27    25    18    16     9     7
nan   nan   nan   nan   nan   nan   nan   nan   nan    14
  23     5    24    22    15    13     6     4;
    18    27     9   nan    26   nan    17   nan     8
nan   nan   nan   nan   nan   nan   nan   nan   nan    15
  24     6   nan    23   nan    14   nan     5;
    19   nan    10   nan   nan    20   nan    11   nan
22   nan    13   nan   nan    23   nan    14   nan   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    20   nan    11   nan   nan    21    19    12    10
23   nan    14   nan   nan    24    22    15    13   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    21   nan    12   nan   nan   nan    20   nan    11
24   nan    15   nan   nan   nan    23   nan    14   nan
nan   nan   nan   nan   nan   nan   nan   nan;
    22   nan    13   nan   nan    23   nan    14   nan
25   nan    16   nan   nan    26   nan    17   nan    19
nan    10   nan   nan    20   nan    11   nan;
```

```
          23    nan     14    nan    nan     24     22     15     13
    26    nan     17    nan    nan     27     25     18     16     20
    nan     11    nan    nan     21     19     12     10;
          24    nan     15    nan    nan    nan     23    nan     14
    27    nan     18    nan    nan    nan     26    nan     17     21
    nan     12    nan    nan    nan     20    nan     11;
          25    nan     16    nan    nan     26    nan     17    nan
    nan    nan    nan    nan    nan    nan    nan    nan    nan     22
     nan     13    nan    nan     23    nan     14    nan;
          26    nan     17    nan    nan     27     25     18     16
    nan    nan    nan    nan    nan    nan    nan    nan    nan     23
     nan     14    nan    nan     24     22     15     13;
          27    nan     18    nan    nan    nan     26    nan     17
    nan    nan    nan    nan    nan    nan    nan    nan    nan     24
     nan     15    nan    nan    nan     23    nan     14];
for i = 1:(N(1)+1)*(N(2)+1)*(N(3)+1)
   for j = 1:27
      bflag = true;
      for k = 1:27
         if(isnan(A(i,k))~= isnan(TEST(j,k)))
            bflag = false;
         end
      end
      if(bflag == true)
         T(i) = j;
         break;
      end
   end
end
return
```

Bibliography

[BSV17] M. Basson, B. Stapelberg, and N. F. J. Van Rensburg. "Error Estimates for Semi-Discrete and Fully Discrete Galerkin Finite Element Approximations of the General Linear Second-Order Hyperbolic Equation". In: *Numerical Functional Analysis and Optimization* 38 (Apr. 2017), pp. 466–485. DOI: 10.1080/01630563.2016.1254655. (Visited on 04/09/2022).

[BV13] M. Basson and N. F. J. Van Rensburg. "Galerkin Finite Element Approximation of General Linear Second Order Hyperbolic Equations". In: *Numerical Functional Analysis and Optimization* 34 (Sept. 2013), pp. 976–1000. DOI: 10.1080/01630563.2013.807286. (Visited on 01/13/2023).

[CVV18] D. Civin, N.F.J. Van Rensburg, and A.J. Van Der Merwe. "Using energy methods to compare linear vibration models". In: *Applied Mathematics and Computation* 321 (Mar. 2018), pp. 602–613. DOI: 10.1016/j.amc.2017.11.008. (Visited on 10/09/2022).

[Fun65] Yuan-cheng Fung. *Foundations of Solid Mechanics*. Prentice Hall, 1965.

[LVV09a] A. Labuschagne, N.F.J. Van Rensburg, and A.J. Van Der Merwe. "Comparison of linear beam theories". In: *Mathematical and Computer Modelling* 49 (Jan. 2009), pp. 20–30. DOI: 10.1016/j.mcm.2008.06.006.

[LVV09b] A. Labuschagne, N.F.J. Van Rensburg, and A.J. Van Der Merwe. "Vibration of a Reissner–Mindlin–Timoshenko plate–beam system". In: *Mathematical and Computer Modelling* 50 (Oct. 2009), pp. 1033–1044. DOI: 10.1016/j.mcm.2009.05.029. (Visited on 05/04/2020).

[OR76] John Tinsley Oden and Junuthula Narasimha Reddy. *An Introduction to the Mathematical Theory of Finite Elements*. Wiley, 1976.

[Rud53] Walter Rudin. *Principles of mathematical analysis*. McGraw-Hill Book Company, Inc., 1953.

[Sad05] Martin H. Sadd. *Elasticity, Theory, Applications and Numerics*. 1st. Elsevier Butterworth–Heinemann, 2005.

[SF73] Gilbert Strang and George Fix. *An analysis if the finite element methid*. 2nd. Wellesley-Cambridge Press, 1973.

[SP06] N.G Stephen and S. Puchegger. "On the valid frequency range of Timoshenko beam theory". In: *Journal of Sound and Vibration* 297 (Nov. 2006), pp. 1082–1087. DOI: `10.1016/j.jsv.2006.04.020`. (Visited on 08/28/2023).

[Tim21] S.P. Timoshenko. "LXVI. On the correction for shear of the differential equation for transverse vibrations of prismatic bars". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 41 (May 1921), pp. 744–746. DOI: `10.1080/14786442108636264`. (Visited on 11/24/2020).

[VS18] N F J Van Rensburg and B Stapelberg. "Existence and uniqueness of solutions of a general linear second-order hyperbolic problem". In: *IMA Journal of Applied Mathematics* 84 (Sept. 2018), pp. 1–22. DOI: `10.1093/imamat/hxy048`. (Visited on 09/17/2021).

[VV02] N.F.J Van Rensburg and A.J. Van Der Merwe. "Analysis of the Solvability of Linear Vibration Models". In: *Applicable Analysis* 81 (Jan. 2002), pp. 1143–1159. DOI: `10.1080/0003681021000029837`. (Visited on 11/25/2022).

[VV06] N.F.J. Van Rensburg and A.J. Van Der Merwe. "Natural frequencies and modes of a Timoshenko beam". In: *Wave Motion* 44 (Nov. 2006), pp. 58–69. DOI: `10.1016/j.wavemoti.2006.06.008`. (Visited on 08/03/2019).

[Wu06] Shen R. Wu. "Lumped mass matrix in explicit finite element method for transient dynamics of elasticity". In: *Computer Methods in Applied Mechanics and Engineering* 195 (Sept. 2006), pp. 5983–5994. DOI: `10.1016/j.cma.2005.10.008`. (Visited on 11/26/2020).

[Zie00] L. Zietsman. "Finite Element Analysis of Vibration Models with Interface Conditions". `http://hdl.handle.net/2263/30497`. PhD thesis. Pretoria: University of Pretoria, 2000.