Department of Construction Sciences
Solid Mechanics

ISRN LUTFD2/TFHF-22/5250-SE(1-56)

Computationally Efficient Methods in Topology Optimization

Master's Dissertation by

Vilmer Dahlberg

Supervisor:
Anna Dalklint

Examiner:
Mathias Wallin

Printed by Media-Tryck AB, Lund, Sweden
For information, address:
Division of Solid Mechanics, Lund University, Box 118, SE-221 00 Lund, Sweden
Webpage: www.solid.lth.se

Acknowledgments

Firstly, I would like to thank my supervisor Anna Dalklint for providing fruitful discussions throughout my thesis, and for being a great source of inspiration. I would also like to thank Oded Amir for guiding me through my thesis, helping me keep my focus straight. A special thanks is given to Krister Svanberg for providing an implementation of MMA in MATLAB.

As this thesis marks the end of my masters degree, I would like to thank my friends Johan and Rebecka for putting up with me for these last five years. Last, and certainly not least, I would like to thank Tuva for always supporting me.

Abstract

In topology optimization, iterative, gradient-based methods are used to find the material distribution of structures which maximizes some objective function, typically the structures' stiffness, or in some cases the fundamental frequency. Finite element analysis is used to compute the structural response in each iteration, leading to large systems of equations. Several hundred iterations may be needed for convergence of the optimization problem, however the design changes may be very small, particularly towards the end of the optimization process. This raises the question if the systems need to be solved exactly, or if information from previous iterations can be used to reduce the computational effort. This is the fundamental idea of reanalysis, which Kirsch used to develop effective basis generation methods for reduced order models, known as combined approximation (CA) [1].

Kirsch's combined approximation has seen some use for static problems in topology optimization, and methods which take the approximations inaccuracies into account for a consistent sensitivity analysis have been developed [2]. Kirsh's CA has also been used for eigenvalue problems, and consistent sensitivity analysis for optimization of a single eigenfrequency have been developed.

We found that some of the basis generation methods Kirsch proposes may be ill suited when multiple eigenfrequencies are used to approximate the fundamental frequency, and we propose a simple remedy to these problems. The sensitivities of the eigenfrequencies and the objective function are derived using the adjoint method, and are compared to finite difference approximations. The simulations show that the basis generation methods which Kirsch proposes are inconsistent, but that the novel method is consistent with a full model. Although, all reduced order methods produced indiscernible results and had similar performance in regard to computational effort saved.

Contents

Chapter 1

Introduction

Many modern Finite Element Analysis software packages, such as COMSOL and ABAQUS, now include implementations of structural optimization which are capable of handling three-dimensional structures and advanced techniques such as stress constraints. Structural optimization is a term which refers to three different types of optimization, *Sizing*, *Shape* and *Topology optimization*. In topology optimization the goal is to find the material distribution in a given design domain, which maximizes some objective function under some constraints. The idea was first presented by Bendsøe and Kikuchi [3] in their seminal paper from 1989 where they used two different material models to describe void and mass. Since then the technique has been applied to various problems, for example designing materials with exceptional properties such as negative Poisson's ratio [4], bi-stable structures [5], and compliant mechanisms [6].

Diaz and Kikuchi [7] used topology optimization to reinforce an existing structure, altering its structural response to free vibrations. This can for example be used to increase a structures fundamental frequency, or more recently it has been used to design crystal materials with tunable phononic band-gaps [8].

In topology optimization, the given design domain is discretized into finite elements so that the structural response (for example deformations, stresses and eigenfrequencies) can be evaluated. The user then determines the objective and constraints, resulting in an optimization problem. Gradient based iterative methods are used to solve the problems, meaning at each design cycle the optimization problem is linearized and solved, resulting in a new design. This process is repeated until convergence, which can require several hundred design cycles, depending on the problem. Consequently a large number of (linear or nonlinear) systems need to be solved, which is computationally expensive, particularly for large and plastic deformations where non-linear finite element analysis is required.

As the end user requires a certain quality, the finite element analysis results in large, sparse, systems of equations. In modern applications this can result in several millions of unknowns, for which exact analysis is not feasible due to limitations on computational power, memory and storage. This problem is prominent in topology optimization where several hundred design cycles may be needed, each of which involves solving at least one (linear or non-linear) system. For this reason consistent numerical methods which reduce the computational effort are sought. Krylov subspace methods and multigrid methods are popular choices in classical analysis, and have seen use in high-level implementations of topology optimization, see for example [9]. Kirsch [1] introduced a technique which reuses information from previous design cycles (known as *reanalysis*) to produce a *reduced order model* of the structural response.

The key component in a reduced order model is the basis generation method. In topology optimization there is a natural way of generating the basis vectors, namely by reusing information from a previous iteration. For the static equilibrium equations, basis vectors are used to approximate the deformations. Hence one set of basis vectors used to approximate the deformations need to

be generated. For the generalized eigenvalue problem, basis vectors are used to approximate the eigenvectors. Hence, for each sought eigenpair one set of basis vectors need to be generated. The basis generation method is particularly important for generalized eigenvalue problems, if they are not generated carefully one might accidentally approximate the incorrect eigenmode.

1.1 Aim

The aim of this work is to investigate reduced order models (ROM) for generalized eigenfrequency problems in topology optimization. Previous implementations only consider a single frequency, whereas in this work a number of the lowest modulus eigenfrequencies are used to approximate the fundamental frequency. This has implications on the sensitivity analysis, in particular consistent sensitivity analysis for different basis generation methods is of interest. In addition, the amount of work saved is investigated.

1.2 Scope

This work investigates Kirsch's Combined Approximation for solving generalized eigenproblems in topology optimization. In order to evaluate the numerical methods a thin beam is studied, meaning plane stress conditions are assumed. The beam's structural response to small, harmonic oscillations is computed using finite element analysis. The objective is to maximize the fundamental frequency which is approximated using the smooth, differentiable p-norm. Svanberg's method of moving asymptotes (MMA) is used to solve the optimization problem.

Chapter 2

Background

In this chapter, essential theory for understanding this thesis is presented. Firstly, a brief overview of finite element analysis (FEA) for statics and dynamics is given, followed by some important concepts from linear algebra, and lastly structural optimization and some numerical considerations are presented. In the next chapter, reduced order models and basis generation methods are discussed, and in the final chapter sensitivity analysis is performed.

In the first two sections the finite element method, used to discretize the balance principle of forces in continuum structures, is presented. FEA is the basis of topology optimization, and is therefore important for understanding this work. Linear elasticity and small deformations are considered, as such the resulting systems of equations are linear. The finite element method for small deformations is well known, so only a brief overview of the concepts is given, a more detailed derivation is given in [10].

2.1 The Finite Element Method - Statics

2.1.1 Equilibrium equations

Consider a continuous body at rest, meaning it is in static equilibrium, subjected to body forces and surface forces. Let the vector of body forces be denoted by $\boldsymbol{b}$, the traction vector be denoted by $\boldsymbol{t}$. It can then be shown that the equilibrium equations become [10]

$$\int_S \boldsymbol{t} dS + \int_V \boldsymbol{b} dV = \boldsymbol{0}, \tag{2.1}$$

where V is the body's domain and S its boundary. For plane stress conditions, which accurately describes thin beams, this can be reduced to

$$\oint_{\mathcal{L}} \boldsymbol{t} d\mathcal{L} + \int_A \boldsymbol{b} dA = \boldsymbol{0}, \tag{2.2}$$

where A is a cross section of the body and $\mathcal{L}$ its boundary. The traction vector can be written in terms of the symmetric stress tensor $\boldsymbol{S}$ and the normal vector $\boldsymbol{n}$ as $\boldsymbol{t} = \boldsymbol{S}\boldsymbol{n}$. This fact can be used in conjunction with Green-Gauss divergence theorem to rewrite the equilibrium equations in the compact vectorized form

$$\tilde{\boldsymbol{\nabla}}^T \boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0}, \tag{2.3}$$

where the matrix differential operator $\tilde{\boldsymbol{\nabla}}$ and the stress vector $\boldsymbol{\sigma}$ have been introduced. For plane stress conditions they may be written as

$$\tilde{\boldsymbol{\nabla}}^T = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & \frac{\partial}{\partial y} \\ 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \text{ and } \boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{bmatrix}.$$

Now that the equilibrium of forces have been established consider the body's deformation, described by the normal and shear strains. For small deformations, the strains $\boldsymbol{\epsilon}$ can be expressed in terms of the vector of displacements $\boldsymbol{u} = \begin{bmatrix} u_x & u_y \end{bmatrix}$ through the *kinematic relation* below

$$\boldsymbol{\epsilon} = \tilde{\boldsymbol{\nabla}}\boldsymbol{u} \text{ where } \boldsymbol{\epsilon}^T = \begin{bmatrix} \epsilon_{xx} & \epsilon_{yy} & \gamma_{xy} \end{bmatrix}. \tag{2.4}$$

Finally, the stresses and strains may be related through a *constitutive relation*. In this work linear elasticity is assumed, which gives the constitutive relation

$$\boldsymbol{\sigma} = \boldsymbol{D}\boldsymbol{\epsilon}, \tag{2.5}$$

where $\boldsymbol{D}$ is the material tangent stiffness matrix, which for isotropy can be described by two parameters: Young's elasticity modulus E and Poissons ratio ν.

$$\boldsymbol{D} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1}{2}(1-\nu) \end{bmatrix}$$

Equations 2.3 are known as the *strong form* of the equilibrium equations, and hold globally.

By multiplying the strong form (2.3) by an arbitrary weight function $\boldsymbol{v}$ and integrating over the body, the equations may be transformed into the equivalent *weak form* [10]. Inserting the constitutive relation (2.5) and the kinematic relation (2.4) and making use of Green-Gauss theorem to move derivatives from $\boldsymbol{\sigma}$ to $\boldsymbol{v}$ gives

$$\int_A (\tilde{\boldsymbol{\nabla}}\boldsymbol{v})^T \boldsymbol{D}\tilde{\boldsymbol{\nabla}}\boldsymbol{u} dA = \oint_{\mathcal{L}} \boldsymbol{v}^T \boldsymbol{t} d\mathcal{L} + \int_A \boldsymbol{v}^T \boldsymbol{b} dA. \tag{2.6}$$

The advantage of transferring derivatives from $\boldsymbol{\sigma}$ to $\boldsymbol{v}$ is that the number of derivatives that is required on $\boldsymbol{u}$ is reduced. In fact, the number of derivatives required on $\boldsymbol{u}$ is now only one, which allows the use of a piecewise continuous representation.

2.1.2 Finite Element Formulation

Lastly, an approximation of the displacement field $\boldsymbol{u}$ is introduced by dividing the body into elements (also known as *discretizing* the body), and approximating the displacement field inside each element using form functions. This may be expressed very generally as

$$\boldsymbol{u} = \boldsymbol{N}\boldsymbol{a}, \quad \tilde{\boldsymbol{\nabla}}\boldsymbol{u} = \boldsymbol{B}\boldsymbol{a},$$

where $\boldsymbol{a}$ is a constant vector describing the displacements in the element's nodes, $\boldsymbol{N}$ are the global form functions and $\boldsymbol{B} = \tilde{\boldsymbol{\nabla}}\boldsymbol{N}$. In this work four-node isoparametric quadrilaterals will be used, see [10] for details, although the derivation holds for any element. The arbitrary vector $\boldsymbol{v}$ is also approximated using the form functions according to the Galerkin condition, that is

$$\boldsymbol{v} = \boldsymbol{N}\boldsymbol{c}, \quad \tilde{\boldsymbol{\nabla}}\boldsymbol{v} = \boldsymbol{B}\boldsymbol{c},$$

where $\boldsymbol{c}$ is an arbitrary, constant vector. Inserting the approximation into the weak form (2.6) gives

$$\boldsymbol{c}^T \left(\int_A \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B}\boldsymbol{a} dA - \oint_{\mathcal{L}} \boldsymbol{N}^T \boldsymbol{t} d\mathcal{L} - \int_A \boldsymbol{N}^T \boldsymbol{b} dA \right) = \boldsymbol{0}.$$

The equation above must hold for all weight functions $\boldsymbol{v}$ and vectors $\boldsymbol{c}$, meaning the term inside the parenthesis must be zero. By introducing the stiffness matrix $\boldsymbol{K}$ and the force vector $\boldsymbol{f}$ as

$$\boldsymbol{K} = \int_A \boldsymbol{B}^T \boldsymbol{D}\boldsymbol{B} dA \tag{2.7}$$

$$\boldsymbol{f} = \oint_{\mathcal{L}} \boldsymbol{N}^T \boldsymbol{t} d\mathcal{L} + \int_A \boldsymbol{N}^T \boldsymbol{b} dA \tag{2.8}$$

the equilibrium equations may be expressed as the linear system of equations

$$\boldsymbol{K}\boldsymbol{a} = \boldsymbol{f}. \tag{2.9}$$

Since the form functions of any element only has support on said element, the integration in equations 2.7 and 2.8 may be carried out on each element separately, after which the system stiffness matrix and system force vector are *assembled*. Let the symbol $\sum$ denote this operation, the quantities can then be written as $\boldsymbol{K} = \sum \boldsymbol{K}^e$ and $\boldsymbol{f} = \sum \boldsymbol{f}^e$ where

$$\boldsymbol{K}^e = \int_{A^e} \boldsymbol{B}^{eT} \boldsymbol{D}^e \boldsymbol{B}^e dA^e \tag{2.10}$$

$$\boldsymbol{f}^e = \oint_{\mathcal{L}^e} \boldsymbol{N}^{eT} \boldsymbol{t}^e d\mathcal{L}^e + \int_{A^e} \boldsymbol{N}^{eT} \boldsymbol{b}^e dA^e \tag{2.11}$$

It can be shown [10] that the stiffness matrix is symmetric positive *semidefinite*, meaning the equilibrium equations (2.9) do not have a unique solution. The stiffness matrix can however always be made positive definite by imposing sufficient boundary conditions. Typically, the deformation is prescribed on a subset of the nodes and the remaining nodes are *free*. Consequently, the forces acting on the nodes with prescribed displacements is unknown, and the forces on the free nodes must be known. The equilibrium equations (2.9) can then be split in the following way

$$\begin{bmatrix} \boldsymbol{K}_{ff} & \boldsymbol{K}_{fp} \\ \boldsymbol{K}_{pf} & \boldsymbol{K}_{pp} \end{bmatrix} \begin{bmatrix} \boldsymbol{a}_f \\ \boldsymbol{a}_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_f \\ \boldsymbol{f}_p \end{bmatrix},$$

where $\boldsymbol{a}_p$ is the deformation in the prescribed nodes and $\boldsymbol{a}_f$ is the deformation in the free nodes.

$$\boldsymbol{K}_{ff}\boldsymbol{a}_f = \boldsymbol{f}_f - \boldsymbol{K}_{fp}\boldsymbol{a}_p, \tag{2.12}$$

where $\boldsymbol{K}_{ff}$ is symmetric positive definite (SPD), guaranteeing that equation 2.12 has a unique solution. For the coming chapters, we will always assume that sufficient boundary conditions have been imposed, meaning the equilibrium equations (2.9) can always be reduced to the form above. In order to simplify notation, we will denote $\boldsymbol{K}_{ff}$ by $\boldsymbol{K}$, $\boldsymbol{a}_f$ by $\boldsymbol{a}$ and so on. It is important to remember that $\boldsymbol{K}$ now refers to $\boldsymbol{K}_{ff}$, which is SPD.

2.2 Dynamics

The starting point of statics is that the acceleration of a body at rest is zero, while for a dynamic description the acceleration must be included. According to Newton's second law, the sum of forces acting on a body must equal the body's mass times its acceleration. The strong form of equilibrium becomes

$$\tilde{\boldsymbol{\nabla}}^T \boldsymbol{\sigma} + \boldsymbol{b} = \rho_m \ddot{\boldsymbol{u}},$$

where ρ_m is the mass density. The weak form can then be derived in the same way as for statics, resulting in

$$\int_A (\tilde{\boldsymbol{\nabla}}\boldsymbol{v})^T \boldsymbol{D} \tilde{\boldsymbol{\nabla}} \boldsymbol{u} dA + \int_A \rho_m \boldsymbol{v}^T \ddot{\boldsymbol{u}} dA = \oint_{\mathcal{L}} \boldsymbol{v}^T \boldsymbol{t} d\mathcal{L} + \int_A \boldsymbol{v}^T \boldsymbol{b} dA.$$

The next step is to discretize the body and to introduce the approximation of the displacement field. In dynamics the form functions are assumed to be constant in time, but the nodal displacements are allowed to vary, meaning

$$\boldsymbol{u}(t) = \boldsymbol{N}\boldsymbol{a}(t) \implies \ddot{\boldsymbol{u}}(t) = \boldsymbol{N}\ddot{\boldsymbol{a}}(t).$$

Choosing the arbitrary vector $\boldsymbol{v}$ according to Galerkins condition gives

$$\boldsymbol{c}^T \left(\int_A \boldsymbol{B}^T \boldsymbol{D} \boldsymbol{B} \boldsymbol{a} dA + \int_A \rho_m \boldsymbol{N}^T \boldsymbol{N} \ddot{\boldsymbol{a}} dA - \oint_{\mathcal{L}} \boldsymbol{N}^T \boldsymbol{t} dl - \int_A \boldsymbol{N}^T \boldsymbol{b} dA \right) = \boldsymbol{0}.$$

Again, noting that $\boldsymbol{c}$ is an arbitrary vector, the term inside the parenthesis must be zero, which results in the equations of motion

$$\boldsymbol{M}\ddot{\boldsymbol{a}} + \boldsymbol{K}\boldsymbol{a} = \boldsymbol{f}, \tag{2.13}$$

where the SPD mass matrix $\boldsymbol{M}$ has been introduced. Similar to the stiffness matrix, the mass matrix may be written as $\boldsymbol{M} = \sum \boldsymbol{M}^e$ where the element mass matrix is given by

$$\boldsymbol{M}^e = \int_{A^e} \rho_m^e \boldsymbol{N}^{eT} \boldsymbol{N}^e dA^e. \tag{2.14}$$

The equations of motion may be split similarly to the static equations, which gives the equation of motion in the free nodes

$$\boldsymbol{M}_{ff}\ddot{\boldsymbol{a}}_f + \boldsymbol{K}_{ff}\boldsymbol{a}_f = \boldsymbol{f}_f - \boldsymbol{K}_{fp}\boldsymbol{a}_p - \boldsymbol{M}_{fp}\ddot{\boldsymbol{a}}_p,$$

where analogous to statics $\ddot{\boldsymbol{a}}_f$ and $\boldsymbol{a}_f$ are the accelerations and deformations in the free nodes, and $\ddot{\boldsymbol{a}}_p$ and $\boldsymbol{a}_p$ are the corresponding quantities in the prescribed nodes. Note that both $\boldsymbol{M}_{ff}$ and $\boldsymbol{K}_{ff}$ are SPD, so to simplify notation we will denote $\boldsymbol{M}_{ff}$ by $\boldsymbol{M}$ and so on.

2.2.1 Free vibrations

The dynamic analysis in this work is limited to free vibrations, meaning all external forces are set to zero and the nodal displacements are described as harmonic vibrations. Hence, an ansatz is made comprised of two parts: one time independent vector describing the vibration shape $\boldsymbol{\psi}$, and a time dependent part describing the vibrations amplitude. A convenient way of introducing harmonic motion is through the exponential function

$$\boldsymbol{a}(t) = \boldsymbol{\psi} e^{j\omega t} \implies \ddot{\boldsymbol{a}}(t) = -\omega^2 \boldsymbol{\psi} e^{j\omega t},$$

where the constant vector $\boldsymbol{\psi}$ is known as the mode shape (or eigenmode), ω is the angular frequency and j is the complex square root of -1. It is assumed that the frequencies are positive, as a negative frequency only has the effect of shifting the phase. The ansatz is inserted into the equation of motion (2.13), according to the proverb *ein ansatz setz mann in*, giving

$$e^{j\omega t} \left(-\omega^2 \boldsymbol{M}\boldsymbol{\psi} + \boldsymbol{K}\boldsymbol{\psi} \right) = \boldsymbol{0}.$$

Since the exponential function is never zero, the expression inside the parenthesis must be zero. Rearranging the terms gives the *generalized eigenproblem*

$$\boldsymbol{K}\boldsymbol{\psi} = \omega^2 \boldsymbol{M}\boldsymbol{\psi}, \tag{2.15}$$

where the numbers ω^2 and vectors $\boldsymbol{\psi}$ are commonly referred to as the *eigenvalues* and *eigenvectors* respectively. In this case the eigenvalues are the squares of the eigenfrequencies and the eigenvectors are the eigenmodes. Sometimes the eigenvalues will be denoted by $\lambda = \omega^2$ for convenience.

2.3 Linear Algebra

In this section, the solution of linear systems of equations and the associated eigenproblems are discussed. These concepts are the building blocks of the reduced order methods discussed in later parts, and are therefore very important for understanding this work. The following sections are restricted to symmetric, positive definite (SPD) matrices as this is generally the case for finite element analysis (assuming sufficient boundary conditions are imposed).

2.3.1 Matrix Factorization - solving the equilibrium equations

Consider the SPD $n \times n$ matrix $\boldsymbol{A}$. It has been proven in several texts, see for example [11], that $\boldsymbol{A}$ always has a unique Cholesky factorization, meaning it can be written as the product

$$\boldsymbol{A} = \boldsymbol{R}^T \boldsymbol{R},$$

where $\boldsymbol{R}$ is an upper triangular $n \times n$ matrix with nonzero diagonal elements. This fact can be used to solve systems of equations with real, symmetric positive definite matrices. Take for example the system below, where $\boldsymbol{b}$ is an arbitrary vector with length n.

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{2.16}$$

By computing the Cholesky factorization $\boldsymbol{A} = \boldsymbol{R}^T \boldsymbol{R}$ and inserting it into the equation above, the problem can be split into two sub problems, namely

$$\boldsymbol{R}^T \boldsymbol{t} = \boldsymbol{b} \tag{2.17}$$

$$\boldsymbol{R}\boldsymbol{x} = \boldsymbol{t} \tag{2.18}$$

Since $\boldsymbol{R}$ is upper triangular, $\boldsymbol{t}$ can be found by solving Eq. 2.17 using forward substitution, after which $\boldsymbol{x}$ is found by inserting $\boldsymbol{t}$ into Eq. 2.18, and using backward substitution. This can be written as

$$\boldsymbol{x} = \boldsymbol{R}^{-1}\boldsymbol{R}^{-T}\boldsymbol{b}, \tag{2.19}$$

where the symbol $\boldsymbol{R}^{-T}$ is unambiguous since $\left(\boldsymbol{R}^{-1}\right)^T = \left(\boldsymbol{R}^T\right)^{-1} = \boldsymbol{R}^{-T}$. The notation $\boldsymbol{x} = \boldsymbol{R}^{-1}\boldsymbol{t}$ means to solve for $\boldsymbol{x}$ using backward substitution, that is $\boldsymbol{R}^{-1}$ is not actually computed. Similarly, $\boldsymbol{t} = \boldsymbol{R}^{-T}\boldsymbol{b}$ means to solve for $\boldsymbol{t}$ using forward substitution, without actually computing $\boldsymbol{R}^{-T}$. The notation in Eq. 2.19 is quite cumbersome, so when the linear system of equations (2.16) is solved using the Cholesky factorization we will instead write

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b},$$

where the symbol $\boldsymbol{A}^{-1}$ refers to the action of computing a forward and a backward substitution on $\boldsymbol{b}$, which is equivalent to multiplying by $\boldsymbol{A}^{-1}$. The work for computing the Cholesky factorization is approximately $\frac{1}{3}n^3$ floating point operations (flops), and the work for backward- and forward substitution is approximately n^2 flops each [11]. The work of solving the linear system of equations is hence dominated by the computation of the Cholesky factorization, and totals to about $\frac{1}{3}n^3$ flops for large n.

2.3.2 Solving eigenproblems

Let us begin this section by considering non-trivial solutions, meaning vectors $\boldsymbol{x}$ which are not the zero vector, to the standard eigenproblem

$$\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}, \tag{2.20}$$

where λ is an eigenvalue and $\boldsymbol{x}$ is an eigenvector to the matrix $\boldsymbol{A}$. Classically, this is approached by considering the *characteristic polynomial*, which can be found by moving all terms to the left hand side.

$$(\boldsymbol{A} - \lambda\boldsymbol{I})\boldsymbol{x} = \boldsymbol{0},$$

where $\boldsymbol{I}$ is the identity matrix. he eigenproblem can now be interpreted in the following way: the eigenvalues are the scalars λ for which the matrix $\boldsymbol{A}_\lambda = \boldsymbol{A} - \lambda\boldsymbol{I}$ is singular, and the eigenvectors $\boldsymbol{x}$ are the vectors which span the nullspace of $\boldsymbol{A}_\lambda$. Defining the characteristic polynomial as the

determinant $\boldsymbol{A}_\lambda$, that is $p_{\boldsymbol{A}}(\lambda) = \det(\boldsymbol{A} - \lambda \boldsymbol{I})$. The matrix $\boldsymbol{A}_\lambda$ is singular if and only if its determinant is zero, hence the eigenvalues can be found by solving for the zeros of the characteristic polynomial. However, finding the eigenvalues by solving for the zeros of the characteristic polynomial is ill conditioned [11], meaning it is very susceptible to round-off errors. Therefore, iterative methods such as Arnoldi iteration are used in practice, for more details see [11].

Next, consider non-trivial solutions to the generalized eigenproblem

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{B}\boldsymbol{x} \tag{2.21}$$

where again λ is an eigenvalue and $\boldsymbol{x}$ is the corresponding eigenvector. All generalized eigenproblems with SPD left or right hand side can be transformed by the use of similarity transformations to standard eigenproblems. By assumption $\boldsymbol{B}$ is SPD so it has a Cholesky factorization $\boldsymbol{B} = \boldsymbol{R}^T\boldsymbol{R}$, which inserted into the generalized eigenproblem gives

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{R}^T\boldsymbol{R}\boldsymbol{x} \iff \left(\boldsymbol{R}^{-T}\boldsymbol{A}\boldsymbol{R}^{-1}\right)\boldsymbol{R}\boldsymbol{x} = \lambda \boldsymbol{R}\boldsymbol{x} \iff \boldsymbol{C}\boldsymbol{y} = \mu \boldsymbol{y}.$$

where $\boldsymbol{C} = \boldsymbol{R}^{-T}\boldsymbol{A}\boldsymbol{R}^{-1}$ and $\boldsymbol{y} = \boldsymbol{R}\boldsymbol{x}$. That is, the generalized eigenproblem has been transformed into a standard eigenproblem which can be solved with iterative methods. The eigenvectors of the generalized eigenproblem $\boldsymbol{x}$ can be found by forward substitution.

2.3.3 Properties of the Generalized Eigenproblem

In this section some properties of the generalized eigenproblem (2.15) are discussed. Firstly, if $\boldsymbol{\psi}$ solves the generalized eigenproblem (2.15), then so does any scaled vector $\alpha\boldsymbol{\psi}$. In other words, the eigenmodes are unique up to a scaling. Typically, the scaling is chosen such that the eigenmodes are $\boldsymbol{M}$-normal, meaning $\boldsymbol{\psi}^T\boldsymbol{M}\boldsymbol{\psi} = 1$.

Furthermore, it can be shown that the eigenfrequencies ω^2 must be positive real numbers by multiplying both sides of the generalized eigenproblem (2.15) by $\boldsymbol{\psi}^T$

$$\boldsymbol{\psi}^T\boldsymbol{K}\boldsymbol{\psi} = \omega^2\boldsymbol{\psi}^T\boldsymbol{M}\boldsymbol{\psi} \iff \omega^2 = \frac{\boldsymbol{\psi}^T\boldsymbol{K}\boldsymbol{\psi}}{\boldsymbol{\psi}^T\boldsymbol{M}\boldsymbol{\psi}} \in \mathbb{R}^+,$$

where it was used that $\boldsymbol{M}$ and $\boldsymbol{K}$ are SPD, that is $\boldsymbol{x}^T\boldsymbol{K}\boldsymbol{x},\ \boldsymbol{x}^T\boldsymbol{M}\boldsymbol{x} \in \mathbb{R}^+$ for any nonzero vector $\boldsymbol{x}$. It can also be shown that for $n \times n$ matrices $\boldsymbol{M}$ and $\boldsymbol{K}$ there are n eigenvalues ω^2 that solve the equation 2.15. Firstly, since $\boldsymbol{M}$ is SPD it has a unique Cholesky factorization $\boldsymbol{M} = \boldsymbol{L}^T\boldsymbol{L}$. The generalized eigenproblem can then be transformed into a standard eigenproblem with the same eigenvalues by inserting the factorization and premultiplying by $\boldsymbol{L}^{-T}$.

$$\boldsymbol{L}^{-T}\boldsymbol{K}\boldsymbol{L}^{-1}\boldsymbol{L}\boldsymbol{\psi} = \omega^2\boldsymbol{L}\boldsymbol{\psi} \iff \boldsymbol{A}\boldsymbol{y} = \omega^2\boldsymbol{y} \tag{2.22}$$

where $\boldsymbol{A} = \boldsymbol{L}^{-T}\boldsymbol{K}\boldsymbol{L}^{-1}$ and $\boldsymbol{y} = \boldsymbol{L}\boldsymbol{\psi}$. It has been shown in many texts, see for example Trefethen and Bau in [11], that the standard eigenproblem (2.22) has n eigenvalues counting multiplicity. It then follows that the generalized eigenproblem (2.15) also has n eigenvalues.

Finally, let $\omega_1,\ \omega_2, \ldots\ \omega_n$ denote the positive roots of the n eigenvalues of the generalized eigenproblem (2.15). Since they are positive numbers they may be labeled in order of ascending magnitude, hence the eigenpair with index (or order) k denoted by $(\omega_k, \boldsymbol{\psi}_k)$ is well defined up to a scaling of $\boldsymbol{\psi}_k$. As noted previously, this is chosen such that the eigenmodes are $\boldsymbol{M}$-normal.

An important property of the generalized eigenproblem is that two eigenmodes with distinct eigenfrequencies are $\boldsymbol{M}$-orthogonal. To prove this, let $(\omega_i, \boldsymbol{\psi}_i)$ and $(\omega_j, \boldsymbol{\psi}_j)$ be two eigenpairs with distinct eigenfrequencies, meaning $\omega_i \neq \omega_j$. They both satisfy the generalized eigenproblem (2.15), meaning¨

$$\boldsymbol{K}\boldsymbol{\psi}_i = \omega_i^2\boldsymbol{M}\boldsymbol{\psi}_i$$
$$\boldsymbol{K}\boldsymbol{\psi}_j = \omega_j^2\boldsymbol{M}\boldsymbol{\psi}_j$$

Premultiplying the first row by $\boldsymbol{\psi}_j$, the second by $\boldsymbol{\psi}_i$ and taking their difference gives

$$\boldsymbol{\psi}_j^T \boldsymbol{K} \boldsymbol{\psi}_i - \boldsymbol{\psi}_i^T \boldsymbol{K} \boldsymbol{\psi}_j = \omega_i^2 \boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{\psi}_i - \omega_j^2 \boldsymbol{\psi}_i^T \boldsymbol{M} \boldsymbol{\psi}_j \iff \mathbf{0} = \left(\omega_i^2 - \omega_j^2\right) \boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{\psi}_i$$

where the symmetry of $\boldsymbol{K}$ and $\boldsymbol{M}$ was used, meaning $\boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{\psi}_i = \boldsymbol{\psi}_i^T \boldsymbol{M} \boldsymbol{\psi}_j$. Since the eigenfrequencies are distinct the term $\omega_i^2 - \omega_j^2$ is nonzero, hence the eigenvectors must be $\boldsymbol{M}$-orthogonal. This implies that the eigenmodes are also $\boldsymbol{K}$-orthogonal, since

$$\boldsymbol{\psi}_j^T \boldsymbol{K} \boldsymbol{\psi}_i = \omega_i^2 \boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{\psi}_i = \omega_i^2 \boldsymbol{\psi}_i^T \boldsymbol{M} \boldsymbol{\psi}_i \delta_{ij}.$$

2.4 Topology Optimization of Continuum Structures

In this section, topology optimization of continuum structures is presented. Numerical considerations such as checkerboard patterns, mesh dependence, and the non-convex nature of the problem are discussed. Lastly, the method of moving asymptotes (MMA) which is used to solve the optimization problem is presented, see Svanberg [12].

Structural optimization consists of two main parts: structural analysis and optimization. The former is formulated as an objective function, often tied to the structure in question, and some constraints on said structure. Typically the objective is to maximize the stiffness of a structure, or perhaps the fundamental frequency, and the available material usually serves as a constraint (also known as the volume constraint). The latter divides the field into three separate groups of problems:

1. *sizing optimization* in which structures comprised of bars are studied and the design variables are the bars' thicknesses

2. *topology optimization* which considers continuum structures which are discretized into finite elements and the design variables are the elements' densities

3. *shape optimization* which also handles continuum structures, but where the mesh's boundary is subject to change and its parameterization is the design variable

The common denominator for all groups is that they can be formulated as mathematical optimization problems of the form

$$(\mathbb{SO}) \begin{cases} \min\limits_{\boldsymbol{z}} \; g_0(\boldsymbol{z}) \\ \text{subject to} \\ \quad g_j(\boldsymbol{z}) \leq 0, \;\; j = 1, 2, ..., m \\ \quad \boldsymbol{z} \in \mathcal{Z} = \left\{ \boldsymbol{z} \in \mathbb{R}^n \;\middle|\; z_{\min} \leq z_l \leq z_{\max}, \;\; l \in [1, n] \right\} \end{cases} \tag{2.23}$$

where g_0 is the objective function, $\boldsymbol{z}$ are the design variables constrained to the domain $\mathcal{Z}$ and g_i are functions describing constraints put on the design variables. Typically, equilibrium equations are not used as constraints in the optimization formulation, but are instead enforced implicitly through the connection between the design variables and the objective function.

The focus of this work is on topology optimization, which was first presented by Bendsøe and Kikuchi [3] as a material distribution problem. In their formulation, a continuum is discretized into finite elements, each of which is described by one of two different microscopic material constituents: substance or void. Consequently, the design variables are binary which is problematic since binary functions are not differentiable. Later, Bendsøe [13] proposed that the design variables should instead be allowed to vary continuously from 0 to 1. This makes the approximation differentiable, at the cost of the physical interpretation of the design variables and the resulting material distribution.

Bendsøe suggested what is now known as the SIMP scheme (Solid Isotropic Material with Penalization) where an elements elasticity modulus is given as a continuous function of the design variables

$$E(z_e) = z_e^q E_{\max},$$

where $E_{\max}$ is the elasticity modulus of the given material, z_e is the design variable of element e, and q is a penalty exponent (usually set to 3) which is introduced to penalize intermediate values of z_e. Since the volume constraint is linear in the design variables z, intermediate values offer low stiffness at an unreasonably high cost driving the optimal solution to a nearly binary result. The scheme has then been refined by Bendsøe and Sigmund [14] to include measures preventing the stiffness matrix from becoming singular due to the elasticity modulus nearing void,

$$E(z_e) = E_{\min} + (E_{\max} - E_{\min}) z_e^q. \tag{2.24}$$

2.4.1 Numerical considerations

It is important to remember that the resulting material distribution produced by the optimization algorithm must be a structure which is possible to construct. Unfortunately, the scheme suggested by Bendsøe and Sigmund is ill posed and produces results which depend on mesh's resolution and contain checkerboard patterns. Moreover, the optimization problem is non-convex meaning that if the program converges there is no guarantee that it does so to the global minima of the problem (that is the truly "optimal" solution). In the next subsections methods which aim to mitigate the problems above are discussed.

2.4.2 Filtering

One idea which solves both the issue of checkerboard patterns and mesh dependence is to introduce a restriction on the length scale, which can be done through a filter. Many different filtering techniques have been proposed, for example the density filter [15], the sensitivity filter [16] and most recently the PDE-based filter [17] [18]. However, direct filtering of the design variables through a density filter or a PDE-filter is the among the most widely used today. In the density filter approach, the design variables z are mapped to the filtered densities ρ by the relation

$$\rho_e = \frac{\sum_{s=1}^{n_{\text{elm}}} w(x_e - x_s) z_s v_s}{\sum_{s=1}^{n_{\text{elm}}} w(x_e - x_s) v_s},$$

where $v_s = \int_{\Omega_s} dv$ is the volume of element s with support on Ω_s and center of mass at x_s. The kernel, w, is often chosen as the linearly decaying hat function $w(x_e - x_s) = \max(0,\ r - ||x_e - x_s||)$ where r is the filter radius and determines the length scale. Ultimately the filter can be expressed in the matrix-vector relation

$$\boldsymbol{\rho} = \mathbf{M}^{\text{f}} \boldsymbol{z}. \tag{2.25}$$

2.4.3 Heaviside thresholding

While the density filter does solve the issue of checkerboard patterns and serves as an effective way of introducing a minimum length scale to the problem, it does so at the cost of smearing the boundary, see for example figure 2.2. Guest [19] suggests to project the densities using a smooth approximation of the Heaviside step-function as a final measure to achieve a binary design. In

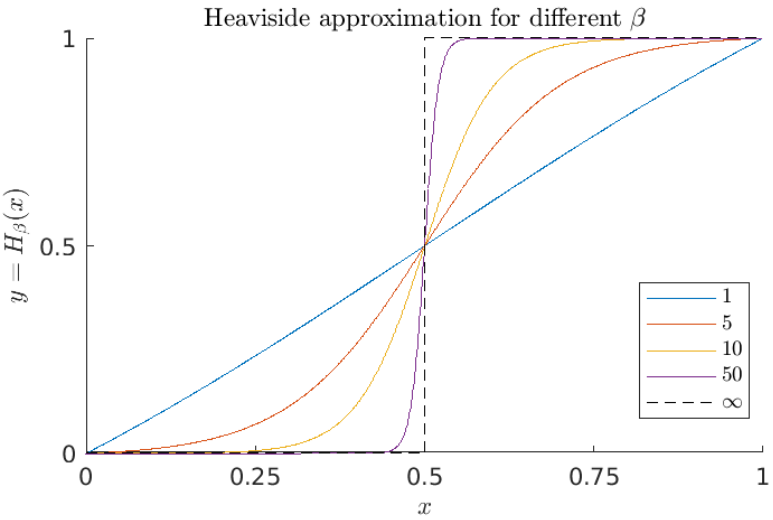


Figure 2.1: The figure shows the Heaviside approximation in equation 2.26 for $\eta = 0.5$ and different choices of β in the interval $x \in [0, 1]$. Note that for $\beta = 1$ the approximation is almost the identity function $y = x$, and for $\beta \to \infty$ the approximation approaches the Heaviside step-function, shown in a dashed black line.

this work the approximation based on the hyperbolic tangent, introduced in [20], is used. The thresholded densities $\bar{\rho}_e$ are in this formulation given by

$$\bar{\rho}_e = H(\rho_e; \beta, \eta) = \frac{\tanh(\beta\eta) + \tanh(\beta(\rho_e - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1 - \eta))}, \tag{2.26}$$

where η determines the point of the transition and β determines its steepness, see figure 2.1. As β approaches infinity H converges to the Heaviside step-function, and for $\beta = 1$ H approximates the identity $\bar{\rho}_e \approx \rho_e$. Typically, the parameter is allowed to change throughout the optimization procedure, starting at a low value such that the optimization is not affected and increasing incrementally in order to force a binary layout. Figure 2.2 shows, from left to right, as first a density filter and then Heaviside projection is applied to a checkerboard pattern.

2.4.4 Sequential Programming

The problems arising from structural optimization can be solved using any optimization algorithm, such as CG or Newton-Raphson schemes, see for example [21]. What is used instead are methods that solve a sequence of convex approximations to the optimization problem. The two most general such methods are *Sequential Linear Programming* (SLP) which is simply a linearization of the problem, and *Sequential Quadratic Programming* (SQP) which extends SLP by adding a second order term [22]. For structural optimization there are methods that more effectively exploit the specific structure of the problem, namely *Convex Linearization* (CONLIN) developed by Fleury and Braibant in [23] and later the *Method of Moving Asymptotes* (MMA) developed by Svanberg in [24] and [12].

The inspiration for CONLIN comes from studying trusses, where it is found that the deformations are given by the reciprocals of the bars' elasticity moduli, see Christensen and Klarbring [22] for details. In sizing optimization the bars' elasticity moduli are the design variables, hence it is natural to linearize the objective function in the reciprocals of the design variables. MMA extends this further by introducing move limits, or *asymptotes*, which control how conservative the approximation is. That is, at iteration k the objective function (or constraint) g_i is approximated

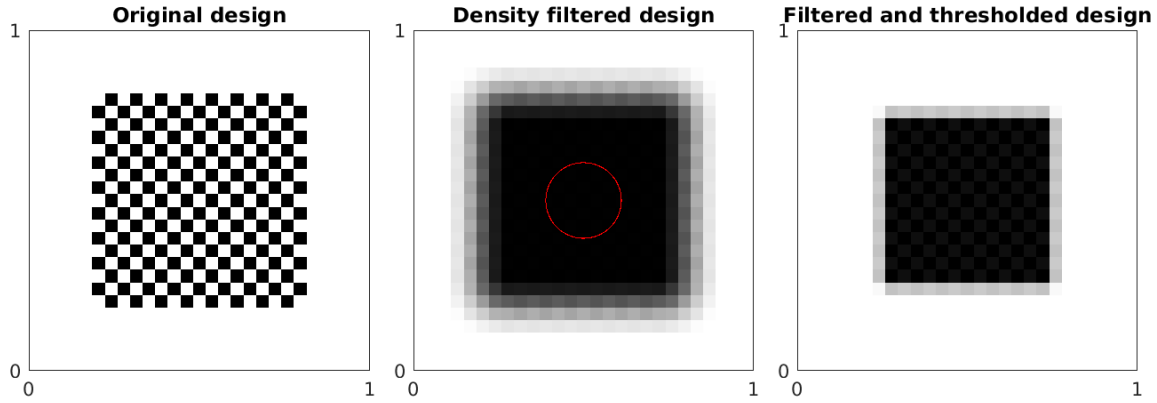


Figure 2.2: From left to right, the first plot shows a design exhibiting a checkerboard pattern. In the next plot a density filter is applied, smoothing out the checkerboard pattern. The filter kernel is shown as the red circle. Finally, Heaviside thresholding is applied to force a (nearly) binary design.

as

$$g_i^{[M],k}(\boldsymbol{z}) = r_i^k + \sum_{l=1}^{n} \left(\frac{p_{i,l}^k}{U_l^k - z_l} - \frac{q_{i,l}^k}{z_l - L_l^k} \right),$$

where U_l^k and L_l^k are the upper and lower asymptotes for design variable z_l at iteration k, which determine how conservative the approximation is. The variables $p_{i,l}^k$, $q_{i,l}^k$ and r_i^k are given by

$$p_{i,l}^k = \begin{cases} \left(U_l^k - z_l^k\right)^2 \frac{\partial g_i}{\partial z_l}\big|_{\boldsymbol{z}^k} & \text{if } \frac{\partial g_i}{\partial z_l}\big|_{\boldsymbol{z}^k} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$q_{i,l}^k = \begin{cases} 0 & \text{if } \frac{\partial g_i}{\partial z_l}\big|_{\boldsymbol{z}^k} \geq 0 \\ \left(z_l^k - L_l^k\right)^2 \frac{\partial g_i}{\partial z_l}\big|_{\boldsymbol{z}^k} & \text{otherwise} \end{cases}$$

$$r_i^k = g_i(\boldsymbol{z}^k) - \sum_{l=1}^{n} \left(\frac{p_{i,l}^k}{U_l^k - z_l^k} + \frac{q_{i,l}^k}{z_l^k - L_l^k} \right)$$

The choice of r_i^k guarantees that the approximation $g_i^{[M],k}$ is exact at the point of linearization, meaning $g_i^{[M],k}(\boldsymbol{z}^k) = g_i(\boldsymbol{z}^k)$, while $p_{i,l}^k$ and $q_{i,l}^k$ guarantee that the gradient is exact at the point of linearization, meaning $\frac{\partial g_i^{[M],k}}{\partial z_l}\big|_{\boldsymbol{z}^k} = \frac{\partial g_i}{\partial z_l}\big|_{\boldsymbol{z}^k}$. By introducing the move limits α_l^k and β_l^k such that $L_l^k < \alpha_l^k \leq z_l^j \leq \beta_l^k < U_l^k$ the approximation is prevented from becoming singular. Putting everything together gives the MMA-approximation of the optimization problem at step k.

$$(\mathbb{SO}^{[M],k}) \begin{cases} \min\limits_{\boldsymbol{z}} \; g_0^{[M],k}(\boldsymbol{z}) \\ \text{subject to} \\ \quad g_j^{[M],k}(\boldsymbol{z}) \leq 0, \;\; j = 1, 2, ..., m \\ \quad \boldsymbol{z} \in \mathcal{Z}^{[M],k} = \left\{ \boldsymbol{z} \in \mathbb{R}^n \;\middle|\; \alpha_l \leq z_l \leq \beta_l, \;\; l \in [1, n] \right\} \end{cases}$$

It can be shown that this approximation is convex and separable, meaning each design variable z_l can be solved for individually using Lagrangian duality [22]. In this work the *Globally Convergent Method of Moving Asymptotes* (GCMMA) is used, which modifies the MMA slightly to guarantee some convergance properties. Since this work is ultimately about numerical methods the reader is referred to [12] for details regarding the optimization scheme.

Chapter 3

Reduced Order Models

This chapter deals with Reduced Order Models and methods for generating basis vectors. In particular *Combined Approximation* (CA) developed by Kirsch in [1] will be presented for static problems and for the generalized eigenproblem. In the context of Topology Optimization these methods need special care, as the choice of basis vectors affect not only the accuracy of the solution, but also the sensitivity analysis. Therefore, a modified version of Kirsch's CA which preserves consistency in sensitivity analysis is presented. The sensitivity analysis itself is discussed in later chapters.

In an earlier chapter, a method for solving linear systems of equations using matrix factorization was presented, and it was shown that the number of flops grows cubically with the number of unknowns. Moreover, the solution requires storing the Cholesky factorization, which can require large amounts of memory. Both these problems put a restriction on how large systems of equations can be solved using matrix factorization. Hence, for large systems of equations other methods should be considered.

One way of reducing the number of flops is to approximate the solution in a subspace of the solution space, which is known as *Reduced Order Models* (ROM). The idea is to first generate a set of basis vectors, based on prior knowledge of the problem, and then approximate the solution as a linear combination of such basis vectors. In this work Kirsch's CA is used to generate the basis vectors.

There are other methods which can reduce the computational effort when solving linear systems of equations. Krylov subspace methods such as GMRES, CG with or without preconditioners and multigrid methods are examples of *iterative methods* which have also been proven to be efficient in topology optimization, see for example [2], [9]. In fact, Kirsch has shown that CA is mathematically equivalent to a preconditioned CG procedure with zero initial guess [25].

3.1 Reduced Static Problem

To begin, consider the equilibrium equations (2.9). Classical methods such as matrix factorization solve the system exactly, using about n^3 operations for a system of size $n \times n$. The computational effort can be reduced, at the cost of decreasing the accuracy, by the use of an approximation. Therefore, let us express the solution to the equilibrium equations (2.9) as a linear combination of the linearly independent *basis vectors* $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_s$, or

$$\boldsymbol{u}^* = \boldsymbol{u}_1 y_1 + \boldsymbol{u}_2 y_2 + ... \boldsymbol{u}_s y_s = \sum_{i=1}^{s} \boldsymbol{u}_i y_i, \tag{3.1}$$

where y_i are the scalar unknowns to be determined, and the star $*$ is used to separate the exact solution $\boldsymbol{u}$ from the approximation $\boldsymbol{u}^*$. If $s = n$ linearly independent basis vectors are used, the

whole solution space is spanned meaning $\boldsymbol{u}^* = \boldsymbol{u}$. However, using a large amount of many basis vectors is expensive, and the point of a reduced order method is to find good basis vectors such that only a few $s << n$ are needed (typically five to ten is enough for a good approximation).

Expressing $\boldsymbol{u}^*$ as a linear combination of basis vectors is equivalent to looking for a solution in the space spanned by the basis vectors, which can be written as $\boldsymbol{u}^* \in \mathcal{U}_s = \text{span}(\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_s)$. Expressing the approximation in this way can be convenient when the basis vectors $\boldsymbol{u}_i$ are manipulated. For example if $\boldsymbol{u}_1$ is replaced with $\alpha\boldsymbol{u}_1$ the space spanned by the basis vectors does not change, so this does not affect the solution $\boldsymbol{u}^*$. For convenience, the linear combination (3.1) is commonly written in matrix-vector format. Therefore, the matrix of basis vectors $\boldsymbol{U}$ and vector of unknowns $\boldsymbol{y}$ are introduced,

$$\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_s] \quad \boldsymbol{y} = [y_1, y_2, ...y_s]^T.$$

Equation 3.1 may now be written as

$$\boldsymbol{u}^* = \sum_{i=1}^{s} \boldsymbol{u}_i y_i = \boldsymbol{U}\boldsymbol{y}. \tag{3.2}$$

Provided the basis vectors $\boldsymbol{u}_i$ are given, it remains to find the vector $\boldsymbol{y}$, after which the approximation is given by equation 3.2. To find $\boldsymbol{y}$, let us insert the linear combination (3.2) directly into the equilibrium equations (2.9) and premultiply by $\boldsymbol{U}$

$$\boldsymbol{U}^T\boldsymbol{K}\boldsymbol{U}\boldsymbol{y} = \boldsymbol{U}^T\boldsymbol{f}. \tag{3.3}$$

Introducing the reduced model stiffness matrix $\boldsymbol{K}_R = \boldsymbol{U}^T\boldsymbol{K}\boldsymbol{U}$, and the reduced model force $\boldsymbol{f}_R = \boldsymbol{U}^T\boldsymbol{f}$ the system may be written as

$$\boldsymbol{K}_R\boldsymbol{y} = \boldsymbol{f}_R. \tag{3.4}$$

The reduced system (3.4) only has s unknowns, and hence has the complexity of about s^3. If an accurate approximation $\boldsymbol{u}^*$ can be found by a small number s basis vectors, the complexity of solving the system can be reduced greatly. The amount of computations that can be saved clearly depends on how the basis vectors are generated.

3.1.1 Reanalysis

In this section Kirsch's method for generating basis vectors, introduced in [1], is derived in the context of topology optimization. In other words, given the vector of design variables $\boldsymbol{z}$ the task is to solve the equilibrium equations

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}, \tag{3.5}$$

where the stiffness matrix is a function of the design variables $\boldsymbol{K} = \boldsymbol{K}(\boldsymbol{z})$ and, without loss of generality, the load $\boldsymbol{f}$ is a constant vector. Assume that this is not the first design iteration, meaning there is some initial design $\boldsymbol{z}_0$ in which the equilibrium equations have been solved, that is

$$\boldsymbol{K}_0\boldsymbol{u}_0 = \boldsymbol{f}, \tag{3.6}$$

where the zeros indicate that the quantities correspond to a previous design cycle, and are hence known constants with respect to the current design $\boldsymbol{z}$. Moreover, let us assume that the equilibrium equations at the initial design were solved by first computing the Cholesky factorization of the old stiffness matrix $\boldsymbol{K}_0$

$$\boldsymbol{K}_0 = \boldsymbol{R}^T\boldsymbol{R}.$$

The goal is to use the factorization above to generate a set of basis vectors that can be used to solve the equilibrium equations for the updated design $\boldsymbol{z}$. The stiffness matrix $\boldsymbol{K}$ at the current design may be related to the old stiffness matrix by introducing the change $\Delta\boldsymbol{K}$,

$$\boldsymbol{K} = \boldsymbol{K}_0 + \Delta\boldsymbol{K}. \tag{3.7}$$

The stiffness matrix in equation 3.5 may then be replaced by equation 3.7

$$(\boldsymbol{K}_0 + \Delta\boldsymbol{K})\boldsymbol{u} = \boldsymbol{f}.$$

As mentioned previously it is assumed that the Cholesky factorization of $\boldsymbol{K}_0$ is known and can be utilized by premultiplying the equation above by $\boldsymbol{K}_0^{-1}$, giving

$$(\boldsymbol{I} + \boldsymbol{K}_0^{-1}\Delta\boldsymbol{K})\boldsymbol{u} = \boldsymbol{K}_0^{-1}\boldsymbol{f}.$$

The inverse of $(\boldsymbol{I} + \boldsymbol{K}_0^{-1}\Delta\boldsymbol{K})$ exists provided the 2-norm of $\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}$ is bounded from above by 1, that is $||\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}||_2 < 1$, see for example Renardy and Rogers in [26]. The inverse is given by the well known power series

$$(\boldsymbol{I} + \boldsymbol{B})^{-1} = \sum_{k=0}^{\infty}(-\boldsymbol{B})^k \quad \text{where} \quad \boldsymbol{B} = \boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}.$$

This power series has an analog for scalars, recall from elementary analysis that $\sum_{k=0}^{\infty} a^k = (1-a)^{-1}$ where $|a| < 1$. A series representation of $\boldsymbol{u}$ can then be found by premultiplying by the inverse above, giving

$$\boldsymbol{u} = (\boldsymbol{I} + \boldsymbol{B})^{-1}\boldsymbol{K}_0^{-1}\boldsymbol{f} = (\boldsymbol{I} - \boldsymbol{B} + \boldsymbol{B}^2 - ...)\boldsymbol{K}_0^{-1}\boldsymbol{f}.$$

It turns out that using only the first few terms, say $s << n < \infty$, as a basis for a reduced order model may produce an accurate approximation of $\boldsymbol{u}$, even if $||\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}||_2 \geq 1$ [1]. The basis vectors can be generated recursively by

$$\begin{aligned} &\boldsymbol{K}_0\boldsymbol{u}_1 = \boldsymbol{f} \\ &\boldsymbol{K}_0\boldsymbol{u}_i = -\Delta\boldsymbol{K}\boldsymbol{u}_{i-1} \quad i = 2..s \end{aligned} \tag{3.8}$$

The computational effort is dominated by the forward- and backward substitution, hence to generate s basis vectors the computational effort is on the magnitude sn^2.

It can be shown that the approximation's error depends on the norm of $\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}$ (to be precise it is proportional to $||\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}||^s$) [9]. Therefore, the larger the changes are between the current and old design, the larger the error becomes. In other words, the more frequently the stiffness matrix is factorization, the more accurate the approximation becomes.

3.2 Reduced Eigenproblem

Consider approximating the eigenvector of index k in the generalized eigenproblem (2.15) by the basis vectors $\boldsymbol{u}_{k,1}, \boldsymbol{u}_{k,2}, ..., \boldsymbol{u}_{k,s}$. Note that in general a unique set of basis vectors is needed for each sought eigenvector, which is emphasized by the index k in the basis vectors. Expressing the approximated eigenvector $\boldsymbol{\psi}_k^*$ as a linear combination of the basis vectors gives

$$\boldsymbol{\psi}_k^* = \boldsymbol{u}_{k,1}y_{k,1} + \boldsymbol{u}_{k,2}y_{k,2} + ..\boldsymbol{u}_{k,s}y_{k,s} = \sum_{i=1}^{s}\boldsymbol{u}_{k,i}y_{k,i},$$

where the basis vectors are linearly independent. Expressing the eigenvectors as a linear combination of basis vectors is equivalent to searching for a vector $\boldsymbol{\psi}_k^*$ in the span of the basis vectors,

or $\boldsymbol{\psi}_k^* \in \mathcal{U}_{k,s} = \text{span}(\boldsymbol{u}_{k,1}, \boldsymbol{u}_{k,2}, ...\boldsymbol{u}_{k,s})$. It can be convenient to express the eigenvector using a matrix-vector expression instead of the sum above. Therefore, the matrix of basis vectors $\boldsymbol{U}_k$ and vector of unknowns $\boldsymbol{y}_k$ are introduced,

$$\boldsymbol{U}_k = [\boldsymbol{u}_{k,1}, \boldsymbol{u}_{k,2}, ..., \boldsymbol{u}_{k,s}], \quad \boldsymbol{y}_k = [y_{k,1}, y_{k,2}, ..., y_{k,s}].$$

The linear combination may now be written as

$$\boldsymbol{\psi}_k^* = \sum_{i=1}^{s} \boldsymbol{u}_{k,i} y_{k,i} = \boldsymbol{U}_k \boldsymbol{y}_k. \tag{3.9}$$

Inserting equation 3.9, into the generalized eigenproblem (2.15) and premultiplying by $\boldsymbol{U}_k$ gives

$$\boldsymbol{U}_k^T \boldsymbol{K} \boldsymbol{U}_k \boldsymbol{y}_k = (\omega_k^*)^2 \boldsymbol{U}_k^T \boldsymbol{M} \boldsymbol{U}_k \boldsymbol{y}_k. \tag{3.10}$$

Introducing the reduced model stiffness matrix $\boldsymbol{K}_k^R = \boldsymbol{U}_k^T \boldsymbol{K} \boldsymbol{U}_k$ and reduced model mass matrix $\boldsymbol{M}_k^R = \boldsymbol{U}_k^T \boldsymbol{M} \boldsymbol{U}_k$ gives the reduced generalized eigenproblem.

$$\boldsymbol{K}_k^R \boldsymbol{y}_k = (\omega_k^*)^2 \boldsymbol{M}_k^R \boldsymbol{y}_k \tag{3.11}$$

The first eigenpair of the reduced generalized eigenproblem can then be used to approximate the kth eigenpair of the full generalized eigenproblem. In other words, $\lambda_k^* = (\omega_k^*)^2$ approximates the kth eigenvalue $\lambda_k = \omega_k^2$, meaning $\lambda_k^* \approx \lambda_k$. An important note is that the vector $\boldsymbol{\psi}_k^*$ may not solve the full generalized eigenproblem, meaning

$$\boldsymbol{K} \boldsymbol{\psi}_k^* - \lambda_k^* \boldsymbol{M} \boldsymbol{\psi}_k^* = \Delta \boldsymbol{f} \neq \boldsymbol{0}$$

Furthermore, there is no guarantee that the first eigenpair of the reduced generalized eigenproblem (3.11) approximates the kth eigenpair of the full generalized eigenproblem. In fact, if the basis vectors are chosen poorly, the first eigenpair may approximate a higher or lower order eigenpair. The method of generating basis vectors is therefore critical for the accuracy of the method.

3.2.1 Kirsch's Combined Approximation

In this section Kirsch's Combined Approximation of generating basis vectors for the generalized eigenproblem is presented in the context of topology optimization. In other words, given the vector of design variables at the current design cycle $\boldsymbol{z}$, the task is to solve the generalized eigenproblem

$$\boldsymbol{K} \boldsymbol{\psi}_k = \lambda_k \boldsymbol{M} \boldsymbol{\psi}_k, \tag{3.12}$$

where the stiffness- and mass-matrix are functions of the design variables $\boldsymbol{K} = \boldsymbol{K}(\boldsymbol{z})$ and $\boldsymbol{M} = \boldsymbol{M}(\boldsymbol{z})$. In the context of topology optimization, it is natural to assume that the generalized eigenproblem has been solved in a previous design iteration, meaning the eigenpair $(\lambda_{0,k}, \boldsymbol{\psi}_{0,k})$ which satisfies

$$\boldsymbol{K}_0 \boldsymbol{\psi}_{0,k} = \lambda_{0,k} \boldsymbol{M}_0 \boldsymbol{\psi}_{0,k} \tag{3.13}$$

has been found. Assume that, in the process of solving the generalized eigenproblem (3.13), the Cholesky factorization of $\boldsymbol{K}_0$ has been computed. The goal is to use the factorization of $\boldsymbol{K}_0$ at the current design cycle to build a set of basis vectors which can be used to and approximate the eigenmode with a reduced order model. The stiffness matrix can be expressed in terms of the old, plus some change due to changes in the design,

$$\boldsymbol{K} = \boldsymbol{K}_0 + \Delta \boldsymbol{K}. \tag{3.14}$$

Inserting equation 3.14 into the generalized eigenproblem (3.12), gives

$$(\boldsymbol{K}_0 + \Delta\boldsymbol{K})\,\boldsymbol{\psi}_k = \lambda_k \boldsymbol{M}\boldsymbol{\psi}_k.$$

As mentioned previously it is assumed that the Cholesky factorization of $\boldsymbol{K}_0$ is known, therefore it is justified to premultiply by $\boldsymbol{K}_0^{-1}$,

$$(\boldsymbol{I} + \boldsymbol{K}_0^{-1}\Delta\boldsymbol{K})\boldsymbol{\psi}_k = \lambda_k \boldsymbol{K}_0^{-1}\boldsymbol{M}\boldsymbol{\psi}_k. \tag{3.15}$$

Provided the norm of $\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}$ is bounded from above by one, that is $||\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}||_2 < 1$, the inverse of the left hand side exists [26] and can be expressed in the power series

$$(\boldsymbol{I} + \boldsymbol{B})^{-1} = \sum_{k=1}^{\infty}(-\boldsymbol{B})^k \quad \text{where} \quad \boldsymbol{B} = \boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}.$$

Inserting the power series into equation 3.15 gives

$$\boldsymbol{\psi}_k = \left(\mathbf{I} - \boldsymbol{B} + \boldsymbol{B}^2 ...\right)\lambda_k \boldsymbol{K}_0^{-1}\boldsymbol{M}\boldsymbol{\psi}_k.$$

It turns out that using only the first few terms, say $s << n < \infty$, as a basis for a reduced order model may produce an accurate approximation of $\boldsymbol{\psi}$, even if $||\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}||_2 \geq 1$ [1]. From practical experience it has been found that the basis vectors produced by the equation above may be poorly scaled relative to each other resulting in ill conditioned reduced order stiffness and mass matrices. Therefore, the basis vectors are normalized with respect to the mass matrix.

Finally, the basis vectors may be generated recursively through the equations below, see algorithm 1 in appendix A for the pseudocode. This method will be referred to as Kirsch's combined approximation for eigenproblems, or CAE for short.

$$\begin{aligned} \boldsymbol{K}_0\boldsymbol{u}_1 &= \boldsymbol{M}\boldsymbol{\psi}_{0,k} \\ \boldsymbol{K}_0\boldsymbol{u}_i &= -\Delta\boldsymbol{K}\boldsymbol{t}_{i-1} \qquad i = 2..s \\ \boldsymbol{t}_i &= \boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2} \quad i = 1..s \end{aligned} \tag{3.16}$$

Note that the eigenvalue λ_k has been ignored since it is an arbitrary scaling factor and will only affect the generation of basis vectors, the space they span is unchanged.

3.2.2 Basis Deflation

Kirsch noted that the accuracy of approximation can be very poor for higher order eigenpairs, since the approximation may converge to lower order eigenmodes (also known as *relaxing*) instead of the sought eigenmode [1]. This can be prevented by *deflating* the eigenvectors, meaning components of lower order eigenmodes are removed from the basis vectors through orthogonalization, preventing the approximation from converging to them.

Assume that the first l eigenmodes have been approximated, that is $\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, ...\boldsymbol{\psi}_l$ are known, and we wish to deflate the basis vector $\boldsymbol{t}_i$ used to approximate the $l+1$ eigenmode $\boldsymbol{\psi}_{l+1}$. The resulting deflated basis vector $\boldsymbol{v}_i$ can then be expressed as a linear combination of $\boldsymbol{t}_i$ and the eigenmodes $\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, ...\boldsymbol{\psi}_l$.

$$\boldsymbol{v}_i = \boldsymbol{t}_i - \sum_{j=1}^{l}\alpha_j\boldsymbol{\psi}_j, \tag{3.17}$$

where α_j are scalars that should be chosen such that $\boldsymbol{v}_i$ is orthogonal to the old eigenmodes, in other words

$$\boldsymbol{\psi}_k^T\boldsymbol{M}\boldsymbol{v}_i = 0 \quad k = 1...l.$$

The scalars can be found by premultiplying equation 3.17 by $\boldsymbol{\psi}_k^T \boldsymbol{M}$ and using the orthonormality of the eigenmodes $\boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{\psi}_j = \delta_{kj}$,

$$0 = \boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{v}_i = \boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{t}_i - \sum_{j=1}^{l} \alpha_j \boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{\psi}_j = \boldsymbol{\psi}_k^t \boldsymbol{M} \boldsymbol{t}_i - \alpha_k \implies \alpha_k = \boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{t}_i.$$

Inserting the expression found above into equation 3.17 gives

$$\boldsymbol{v}_i = \boldsymbol{t}_i - \sum_{j=1}^{l} \left(\boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{t}_i \right) \boldsymbol{\psi}_j.$$

Combining this with Kirsch's CA (3.16) gives the eigenmode orthogonalized CA, or CAEEON for short. The scheme to generate the basis vectors is shown below, see algorithm 2 for the pseudocode.

$$\begin{aligned} \boldsymbol{K}_0 \boldsymbol{u}_1 &= \boldsymbol{M} \boldsymbol{\psi}_{0,k} & & \\ \boldsymbol{K}_0 \boldsymbol{u}_i &= -\Delta \boldsymbol{K} \boldsymbol{t}_{i-1} & & i = 2..s \\ \boldsymbol{t}_i &= \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-1/2} & & i = 1..s \\ \boldsymbol{v}_i &= \boldsymbol{t}_i - \sum_{j=1}^{k-1} \left(\boldsymbol{\psi}_j^T \boldsymbol{M} \boldsymbol{t}_i \right) \boldsymbol{\psi}_j & & i = 1..s \end{aligned} \tag{3.18}$$

3.2.3 Modified Basis Deflation

Note that by orthogonalizing the basis vectors with respect to the current eigenvectors as done above, the basis vectors corresponding to eigenvalue l depend on the eigenmodes of first $l-1$ eigenmodes. Consequently, the n_e reduced eigenvalue problems are coupled, meaning they can not be solved independently. In coming chapters sensitivity analysis of the eigenvalues is discussed, in which it is seen that the coupled nature of the orthogonalization is very problematic. Since the eigenmodes are dependent on the current design, their derivative appears in the sensitivity analysis.

We present a novel approach to the orthogonalization which remedies the problems mentioned above by orthogonalizing with respect to a set of old eigenmodes. Firstly, the orthogonalization is done with respect to a given set of known vectors, meaning the problems are no longer dependent on each other, hence the problems are no longer coupled. Secondly, the old eigenmodes are constants with respect to the current design hence their derivative is zero and the problems in sensitivity analysis is avoided.

The question is which eigenmodes should be chosen for orthogonalization. Remember, the reason the basis vectors are orthogonalized to the eigenvectors is to deflate them such that the reduced eigenproblem converges to the desired eigenvalue. Hence, one could argue that the best option would be the most recent eigenvectors. We argue that the natural choice would be the eigenmodes corresponding to $\boldsymbol{K}_0$, that is $\boldsymbol{\psi}_{0,1}, \boldsymbol{\psi}_{0,2} ... \boldsymbol{\psi}_{0,l-1}$, although it is up to the user which eigenmodes are chosen. The scheme to generate the basis vectors is shown below, where $\hat{\boldsymbol{\psi}}_j$ denotes some old eigenvector of index j, see algorithm 3 for the pseudocode. This method will be referred to as the modified eigenmode orthogonalized CA, or CAEEONmod for short.

$$\begin{aligned} \boldsymbol{K}_0 \boldsymbol{u}_1 &= \boldsymbol{M} \boldsymbol{\psi}_{0,k} & & \\ \boldsymbol{K}_0 \boldsymbol{u}_i &= -\Delta \boldsymbol{K} \boldsymbol{t}_{i-1} & & i = 2..s \\ \boldsymbol{t}_i &= \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-1/2} & & i = 1..s \\ \boldsymbol{v}_i &= \boldsymbol{t}_i - \sum_{j=1}^{k-1} \left(\hat{\boldsymbol{\psi}}_j^T \boldsymbol{M} \boldsymbol{t}_i \right) \hat{\boldsymbol{\psi}}_j & & i = 1..s \end{aligned} \tag{3.19}$$

3.3 Numerical considerations

As noted previously, the accuracy of the reduced order model depends on how often a factorization of the stiffness matrix is performed. At the same time, the fewer times this is done the more work is saved overall. Therefore, it is important to consider how frequently the stiffness matrix should be factorized. As mentioned previously the error of the approximation depends to the norm of $\boldsymbol{K}_0\Delta\boldsymbol{K}$, meaning a measure of the change in the design could be used to determine when a new factorization needs to be performed.

Amir [2] suggests to use the cosine of the angle between the current design $\boldsymbol{z}$, and the design corresponding to the stiffness matrix $\boldsymbol{K}_0$, meaning $\boldsymbol{z}_0$, to determine if a new factorization is necessary. The cosine between two vectors is given by the dot-product

$$\cos(\theta) = \frac{\boldsymbol{z}^T\boldsymbol{z}_0}{||\boldsymbol{z}||_2||\boldsymbol{z}_0||_2}. \tag{3.20}$$

If the design changes are small, the vectors $\boldsymbol{z}$ and $\boldsymbol{z}_0$ should be roughly parallel meaning the cosine of the angle is 1. If, on the other hand, the designs are very different the vectors should be almost perpendicular, meaning the cosine of the angle is 0. Equivalently, one can consider the sine of the angle, given by the trigonometric identity

$$\sin(\theta) = \sqrt{1-\cos^2(\theta)} = \sqrt{1 - \frac{\left(\boldsymbol{z}^T\boldsymbol{z}_0\right)^2}{||\boldsymbol{z}||_2^2||\boldsymbol{z}_0||_2^2}}. \tag{3.21}$$

The advantage is that the sine of the angle is small when the change between the vectors is small, and the sine of the angle is almost 1 when the designs are perpendicular, which corresponds nicely to the norm of two vectors.

During the optimization, if the sine of the angle between the current and old design supersedes the tolerance α_{tol} a factorization is performed. In addition, Amir suggests to force a factorization at set intervals. Bogomolny found that often times CA would produce accurate results, but that sometimes an additional exact analysis is needed [27]. He suggests that a criterion on the approximation's accuracy should be introduced, which he defines as the magnitude of the vector $\Delta\boldsymbol{f}_k$ relative $\boldsymbol{K}\boldsymbol{U}_k\boldsymbol{y}_k$, or

$$\delta_k = \frac{||\Delta\boldsymbol{f}_k||_2}{||\boldsymbol{K}\boldsymbol{U}_k\boldsymbol{y}_k||_2} = \frac{||\boldsymbol{K}\boldsymbol{U}_k\boldsymbol{y}_k - \lambda_k\boldsymbol{M}\boldsymbol{U}_k\boldsymbol{y}_k||_2}{||\boldsymbol{K}\boldsymbol{U}_k\boldsymbol{y}||_2} \leq \delta_{\text{tol}}. \tag{3.22}$$

Meaning, after the reduced order model is solved the condition above is checked, and if it fails an additional factorization is performed and the full problem is solved.

Chapter 4

Consistent Sensitivity Analysis

In this chapter expressions for the eigenfrequencies sensitivity to design changes (gradients) will be derived. In [2] Amir uses reduced order methods to approximate the equilibrium equations and presents a consistent approach to sensitivity analysis which takes the inaccuracies of the reduced order method into account. Consistent sensitivity analysis has then been extended to the reduced eigenproblem by Bogomolny [27], which is used as a base for this chapter. The sensitivity analysis will depend on the choice of basis vectors, hence each basis generation method described in the previous chapter needs its own sensitivity analysis.

In the following sections the numerator convention for computing derivatives between scalars, vectors and matrices is used. Hence, vector valued functions retain their orientation when differentiated with respect to scalars, whereas the derivative of a scalar valued function with respect to a column vector is a row vector, and vice versa.

4.1 Full eigenproblem

First, consider the full generalized eigenproblem (2.15). To find the sensitivity of the kth simple eigenvalue $\lambda_k = \omega_k^2$ differentiate the equation implicitly with respect to the filtered design variables $\bar{\rho}_e$,

$$\left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\right) \boldsymbol{\psi}_k + \left(\boldsymbol{K} - \lambda_k \boldsymbol{M}\right) \frac{\partial \boldsymbol{\psi}_k}{\partial \bar{\rho}_e} - \frac{\partial \lambda_k}{\partial \bar{\rho}_e} \boldsymbol{M} \boldsymbol{\psi}_k = \boldsymbol{0}.$$

Premultiplying by the eigenvector implies

$$\boldsymbol{\psi}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\right) \boldsymbol{\psi}_k + \boldsymbol{\psi}_k^T \left(\boldsymbol{K} - \lambda_k \boldsymbol{M}\right) \frac{\partial \boldsymbol{\psi}_k}{\partial \bar{\rho}_e} = \frac{\partial \lambda_k}{\partial \bar{\rho}_e} \boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{\psi}_k.$$

Note that the middle term is zero, since $\boldsymbol{\psi}_k$ solves the generalized eigenproblem (2.15). Assuming that the eigenvectors are $\boldsymbol{M}$-normal, meaning $\boldsymbol{\psi}_k^T \boldsymbol{M} \boldsymbol{\psi}_k = 1$ the sensitivity may be expressed as

$$\frac{\partial \lambda_k}{\partial \bar{\rho}_e} = \boldsymbol{\psi}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\right) \boldsymbol{\psi}_k. \tag{4.1}$$

4.2 Reduced eigenproblem

The sensitivity of the eigenfrequencies produced by the reduced order model may be found in a similar way as above, by differentiating the reduced generalized eigenproblem (3.11) implicitly with

respect to the design variables.

$$\left(\boldsymbol{V}_k^T\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e}\boldsymbol{V}_k-\lambda_k^*\boldsymbol{V}_k^T\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\boldsymbol{V}_k\right)\boldsymbol{y}_k+\left(\boldsymbol{V}_k^T\boldsymbol{K}\boldsymbol{V}_k-\lambda_k^*\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\right)\frac{\partial \boldsymbol{y}_k}{\partial \bar{\rho}_e}-\frac{\partial \lambda_k^*}{\partial \bar{\rho}_e}\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\boldsymbol{y}_k+$$
$$2\left(\boldsymbol{V}_k^T\boldsymbol{K}-\lambda_k^*\boldsymbol{V}_k^T\boldsymbol{M}\right)\frac{\partial \boldsymbol{V}_k}{\partial \bar{\rho}_e}\boldsymbol{y}_k=\boldsymbol{0}.$$

By multiplying the equation above by the reduced model eigenmode $\boldsymbol{y}_k$ from the left and moving some terms around the expression may be simplified

$$\frac{\partial \lambda_k^*}{\partial \bar{\rho}_e}\boldsymbol{y}_k^T\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\boldsymbol{y}_k=\boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e}-\lambda_k^*\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k+\boldsymbol{y}_k^T\left(\boldsymbol{V}_k^T\boldsymbol{K}\boldsymbol{V}_k-\lambda_k^*\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\right)\frac{\partial \boldsymbol{y}_k}{\partial \bar{\rho}_e}+$$
$$2\boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\boldsymbol{K}-\lambda_k^*\boldsymbol{M}\right)\frac{\partial \boldsymbol{V}_k}{\partial \bar{\rho}_e}\boldsymbol{y}_k$$

The expression can be simplified further by assuming that the approximate eigenmodes are normalized with respect to the reduced mass matrix, meaning $\boldsymbol{y}_k^T\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\boldsymbol{y}_k=1$. Furthermore, since $\boldsymbol{y}_k$ solves the reduced generalized eigenproblem (3.11), the term multiplying the derivative of $\boldsymbol{y}_k$ is zero. Lastly, the quantity $\Delta\boldsymbol{f}_k=\left(\boldsymbol{K}-\lambda_k^*\boldsymbol{M}\right)\boldsymbol{V}_k\boldsymbol{y}_k$ may be introduced to condense the expression slightly.

$$\frac{\partial \lambda_k^*}{\partial \bar{\rho}_e}=\boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e}-\lambda_k^*\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k+\sum_{i=1}^{s}2\Delta\boldsymbol{f}_k^T\frac{\partial \boldsymbol{v}_i}{\partial \bar{\rho}_e}y_i$$

Two things should be noted here. Firstly, as the approximate eigenpair $(\lambda_k^*,\boldsymbol{V}_k\boldsymbol{y}_k)$ approaches the exact eigenpair, $\Delta\boldsymbol{f}_k$ approaches zero. Thus the quantity $\Delta\boldsymbol{f}_k$ can be interpreted as the approximated eigenpair's 'error'. Consequently, if the approximation is good (meaning in this context that the 'error' is small), the last term vanishes and the sensitivity approaches the exact value in equation 4.1. Hence, if the reduced order model accurately approximates the eigenfrequencies, the sensitivity analysis is also accurate. However, if $\Delta\boldsymbol{f}_k$ is not zero, the second term is needed for a *consistent* sensitivity analysis.

Secondly, the derivative of the basis vectors appeared in the sensitivity. Consider for the sake of argument, differentiating the first basis vector $\boldsymbol{u}_1$ given by the first row in equation 3.16,

$$\boldsymbol{K}_0\frac{\partial \boldsymbol{u}_1}{\partial \bar{\rho}_e}=\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\boldsymbol{\psi}_{0,k}\implies\frac{\partial \boldsymbol{u}_1}{\partial \bar{\rho}_e}=\boldsymbol{K}_0^{-1}\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e}\boldsymbol{\psi}_{0,k}.$$

Thus, if the basis vectors' derivatives were to be used directly, one pair of backward and forward substitutions would be needed for each basis vector and design variable. This is not feasible for anything more than a few design variables. Hence the sensitivity needs to be computed in some other way, by for example the use of adjoint equations similar to [2] and [27]. This process will of course depend on how the basis vectors are generated, and will be discussed in upcoming sections.

4.2.1 Standard basis vectors

As noted above and in [2] the basis vectors needs to be taken into account in sensitivity analysis. This was then developed by Bogomolny in [27] for consistent sensitivity analysis for eigenproblems, which will be the basis of this section.

Therefore, consider first the optimization problem when the standard basis vectors from section

3.2.1 is used to reduce the generalized eigenvalue problem

$$\begin{aligned}
&\min\ g_0(\lambda_k^*)\\
&\text{subject to}\\
&\boldsymbol{V}^T\boldsymbol{K}\boldsymbol{V}\boldsymbol{y}_k = \lambda_k^*\boldsymbol{V}^T\boldsymbol{M}\boldsymbol{V}\boldsymbol{y}_k\\
&\boldsymbol{K}_0\boldsymbol{u}_1 = \boldsymbol{M}\boldsymbol{\psi}_{0,k} && (4.2)\\
&\boldsymbol{K}_0\boldsymbol{u}_i = -\Delta\boldsymbol{K}\boldsymbol{v}_{i-1} \qquad i = 2,...s && (4.3)\\
&\boldsymbol{v}_i = \boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2} \qquad i = 1,...s && (4.4)
\end{aligned}$$

As suggested by Bogomolny adjoints corresponding to terms 4.2 and 4.3 are added to the eigenvalue. However, in contrast to the work by Bogomolny terms corresponding to basis normalization (4.4), are added in our work. The resulting augmented eigenvalue is given below

$$\tilde{\lambda}_k^* = \lambda_k^* + \boldsymbol{z}_1^T\left(\boldsymbol{K}_0\boldsymbol{u}_1 - \boldsymbol{M}\boldsymbol{\psi}_{0,k}\right) + \sum_{i=2}^{s}\boldsymbol{z}_i^T\left(\boldsymbol{K}_0\boldsymbol{u}_i + \Delta\boldsymbol{K}\boldsymbol{v}_{i-1}\right) + \sum_{i=1}^{s}\boldsymbol{q}_i^T\left(\boldsymbol{v}_i - \boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2}\right)$$

The constant vectors $\boldsymbol{z}_j$ and $\boldsymbol{q}_j$ are known as *adjoints* and since they multiply zeros λ_k^*, $\tilde{\lambda}_k^*$ and their sensitivities are equal. The advantage of introducing the adjoints is that they are free to be chosen, and so the sensitivity of $\tilde{\lambda}_k^*$ may be easier (and even cheaper) to evaluate if the adjoints are chosen in a smart way. Differentiating the augmented eigenvalue $\tilde{\lambda}_k^*$ with respect to the design variables gives

$$\begin{aligned}
\frac{\partial\tilde{\lambda}_k^*}{\partial\bar{\rho}_e} &= \boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e} - \lambda_k^*\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k + \sum_{i=1}^{s}2\Delta\boldsymbol{f}_k^T y_i\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e}\\
&+ \boldsymbol{z}_1^T\left(\frac{\partial\boldsymbol{K}_0}{\partial\bar{\rho}_e}\boldsymbol{u}_1 + \boldsymbol{K}_0\frac{\partial\boldsymbol{u}_1}{\partial\bar{\rho}_e} - \frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_{0,k} - \boldsymbol{M}\frac{\partial\boldsymbol{\psi}_{0,k}}{\partial\bar{\rho}_e}\right)\\
&+ \sum_{i=2}^{s}\boldsymbol{z}_i^T\left(\frac{\partial\boldsymbol{K}_0}{\partial\bar{\rho}_e}\boldsymbol{u}_i + \boldsymbol{K}_0\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e} + \frac{\partial\Delta\boldsymbol{K}}{\partial\bar{\rho}_e}\boldsymbol{v}_i + \Delta\boldsymbol{K}\frac{\partial\boldsymbol{v}_{i-1}}{\partial\bar{\rho}_e}\right)\\
&+ \sum_{i=1}^{s}\boldsymbol{q}_i^T\left(\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e} - \frac{\partial}{\partial\bar{\rho}_e}\left[\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2}\right]\right)
\end{aligned} \tag{4.5}$$

Using the definition of $\Delta\boldsymbol{K}$ in equation 3.14 and that $\boldsymbol{K}_0$ and $\boldsymbol{\psi}_{0,k}$ are constants with regard to the current design, gives

$$\frac{\partial\boldsymbol{\psi}_{0,k}}{\partial\bar{\rho}_e} = \boldsymbol{0}, \quad \frac{\partial\boldsymbol{K}_0}{\partial\bar{\rho}_e} = \boldsymbol{0}, \quad \frac{\partial\Delta\boldsymbol{K}}{\partial\bar{\rho}_e} = \frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e} \tag{4.6}$$

Lastly, rewriting the normalization term $\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2}$ using index notation (disregarding the index i for a moment) its sensitivity can be computed

$$\begin{aligned}
&\frac{\partial}{\partial x}\left[u^l\left(u^m M^{mn}u^n\right)^{-1/2}\right] =\\
&= \frac{\partial u^l}{\partial x}\left(u^m M^{mn}u^n\right)^{-1/2} - \frac{1}{2}u^l\left(u^o M^{op}u^p\right)^{-3/2}\left(\frac{\partial u^q}{\partial x}M^{qr}u^r + u^s\frac{\partial M^{st}}{\partial x}u^t + u^v M^{vw}\frac{\partial u^w}{\partial x}\right)\\
&= \left(\left(u^m M^{mn}u^n\right)^{-1/2}\frac{\partial u^l}{\partial x} - u^l\left(u^o M^{op}u^p\right)^{-3/2}\frac{\partial u^q}{\partial x}M^{rq}u^r\right) - \frac{1}{2}u^l\left(u^o M^{op}u^p\right)^{-3/2}u^s\frac{\partial M^{st}}{\partial x}u^t\\
&= \left(\delta_{ql}u^m M^{mn}u^n - u^l u^r M^{rq}\right)\left(u^o M^{op}u^p\right)^{-3/2}\frac{\partial u^q}{\partial x} - \frac{1}{2}u^l\left(u^o M^{op}u^p\right)^{-3/2}u^s\frac{\partial M^{st}}{\partial x}u^t,
\end{aligned}$$

where it was used that the mass matrix is symmetric, meaning $u^v M^{vw}\frac{\partial u^w}{\partial x} = u^v M^{wv}\frac{\partial u^w}{\partial x} = \frac{\partial u^v}{\partial x}M^{vw}u^w$. The expression may now be written in matrix-vector format

$$\frac{\partial}{\partial\bar{\rho}_e}\left[\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2}\right] = \left[\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i - \boldsymbol{u}_i\boldsymbol{u}_i^T\boldsymbol{M}\right)\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e} - \frac{1}{2}\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{u}_i\right)\right]\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-3/2} \tag{4.7}$$

Inserting equations 4.6 and 4.7 into equation 4.5 gives

$$\begin{aligned}\frac{\partial\tilde{\lambda}_k^*}{\partial\bar{\rho}_e} &= \boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}-\lambda_k^*\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k+\sum_{i=1}^{s}2\Delta\boldsymbol{f}_k^T y_i\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e}\\ &+\boldsymbol{z}_1^T\left(\boldsymbol{K}_0\frac{\partial\boldsymbol{u}_1}{\partial\bar{\rho}_e}-\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_{0,k}\right)+\sum_{i=2}^{s}\boldsymbol{z}_i^T\left(\boldsymbol{K}_0\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e}+\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}\boldsymbol{v}_{i-1}+\Delta\boldsymbol{K}\frac{\partial\boldsymbol{v}_{i-1}}{\partial\bar{\rho}_e}\right)\\ &+\sum_{i=1}^{s}\boldsymbol{q}_i^T\left(\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e}-\left[\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i-\boldsymbol{u}_i\boldsymbol{u}_i^T\boldsymbol{M}\right)\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e}-\frac{1}{2}\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{u}_i\right)\right]\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-3/2}\right)\end{aligned}$$

The trick is now to find expressions for the adjoints such that the basis vector's sensitivities do not contribute. Therefore, collect the terms that multiply them.

$$\begin{aligned}\frac{\partial\tilde{\lambda}_k^*}{\partial\bar{\rho}_e} &= \boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}-\lambda_k^*\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k-\boldsymbol{z}_1^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_{0,k}\\ &+\sum_{i=2}^{s}\boldsymbol{z}_i^T\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}\boldsymbol{v}_{i-1}+\frac{1}{2}\sum_{i=1}^{s}\boldsymbol{q}_i^T\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{u}_i\right)\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-3/2}\\ &+\sum_{i=1}^{s}\left(\boldsymbol{z}_i^T\boldsymbol{K}_0-\boldsymbol{q}_i^T\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i-\boldsymbol{u}_i\boldsymbol{u}_i^T\boldsymbol{M}\right)\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-3/2}\right)\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e}\\ &+\sum_{i=1}^{s-1}\left(2y_i\Delta\boldsymbol{f}_k^T+\boldsymbol{q}_i^T+\boldsymbol{z}_{i+1}^T\Delta\boldsymbol{K}\right)\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e}\\ &+\left(2y_s\Delta\boldsymbol{f}_k^T+\boldsymbol{q}_s^T\right)\frac{\partial\boldsymbol{v}_s}{\partial\bar{\rho}_e}\end{aligned}\tag{4.8}$$

The first two rows of equation 4.8 are simple to evaluate, and do not contain derivatives of basis vectors, although the last rows do. It is at this point the augmented eigenvalue is advantageous, as the adjoints are free to choose. If they are chosen according to the equations below, the last three rows may be eliminated.

$$\begin{aligned}&\boldsymbol{q}_s=-2y_s\Delta\boldsymbol{f}_k\\ &\boldsymbol{K}_0\boldsymbol{z}_i=(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i-\boldsymbol{M}\boldsymbol{u}_i\boldsymbol{u}_i^T)(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i)^{-3/2}\boldsymbol{q}_i \qquad i=s,...1\\ &\boldsymbol{q}_i=-2y_i\Delta\boldsymbol{f}_k-\Delta\boldsymbol{K}\boldsymbol{z}_{i+1} \qquad i=s-1,...1\end{aligned}\tag{4.9}$$

These equations must be solved in reverse order, meaning first $\boldsymbol{q}_s$ can be solved, followed by $\boldsymbol{z}_s$ after which $\boldsymbol{q}_{s-1}$ and $\boldsymbol{z}_{s-1}$ can be solved, and so on. The vectors $\boldsymbol{q}_j$ are given by simple matrix-vector multiplication and do not require a lot of work, but solving for the s vectors $\boldsymbol{z}_j$ does require an additional s backward and forward substitutions, which has the complexity of about sn^2, but this is still an inexpensive step compared to finding the eigenvalues of the full system which has the complexity of n^3.

After the adjoint equations are solved, the sensitivities become

$$\begin{aligned}\frac{\partial\tilde{\lambda}_k^*}{\partial\bar{\rho}_e} &= \boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}-\lambda_k^*\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k-\boldsymbol{z}_1^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_{0,k}\\ &+\sum_{i=2}^{s}\boldsymbol{z}_i^T\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}\boldsymbol{v}_{i-1}+\frac{1}{2}\sum_{i=1}^{s}\boldsymbol{q}_i^T\boldsymbol{v}_i\left(\boldsymbol{v}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{v}_i\right)\end{aligned}\tag{4.10}$$

4.2.2 Eigenmode Orthogonalized Combined Approximation

The same procedure as above may be performed for eigenmode orthogonalized CA, where the optimization problem may be stated as

$$
\begin{aligned}
&\max\ g_0(\lambda_k^*)\\
&\text{subject to}\\
&\quad \boldsymbol{V}_k^T\boldsymbol{K}\boldsymbol{V}_k\boldsymbol{y}_k = \lambda_k^*\boldsymbol{V}_k^T\boldsymbol{M}\boldsymbol{V}_k\boldsymbol{y}_k\\
&\quad \boldsymbol{K}_0\boldsymbol{u}_1 = \boldsymbol{M}\boldsymbol{\psi}_{0,k}\\
&\quad \boldsymbol{K}_0\boldsymbol{u}_i = -\Delta\boldsymbol{K}\boldsymbol{t}_{i-1} && i = 2,...s\\
&\quad \boldsymbol{t}_i = \boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2} && i = 1,...s\\
&\quad \boldsymbol{v}_i = \boldsymbol{t}_i - \sum_{j=1}^{k-1}\left(\boldsymbol{t}_i^T\boldsymbol{M}\boldsymbol{\psi}_j\right)\boldsymbol{\psi}_j && i = 1,...s
\end{aligned}
$$

As in section 4.2.1, one term per constraint (in total 3s terms) is added to the eigenvalue in order to remove derivatives of basis vectors

$$
\begin{aligned}
\tilde{\lambda}_k^* = \lambda_k^* &+ \boldsymbol{z}_1^T\left(\boldsymbol{K}_0\boldsymbol{u}_1 - \boldsymbol{M}\boldsymbol{\psi}_{0,k}\right) + \sum_{i=2}^{s}\boldsymbol{z}_i^T\left(\boldsymbol{K}_0\boldsymbol{u}_i + \Delta\boldsymbol{K}\boldsymbol{t}_{i-1}\right)\\
&+ \sum_{i=1}^{s}\boldsymbol{q}_i^T\left(\boldsymbol{t}_i - \boldsymbol{u}_i\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-1/2}\right) + \sum_{i=1}^{s}\boldsymbol{w}_i^T\left(\boldsymbol{v}_i - \boldsymbol{t}_i + \sum_{j=1}^{k-1}\left(\boldsymbol{t}_i^T\boldsymbol{M}\boldsymbol{\psi}_j\right)\boldsymbol{\psi}_j\right),
\end{aligned}
\tag{4.11}
$$

where, due to the extra set of equations, an extra set of adjoints $\boldsymbol{w}_j$ have been added. Differentiating with respect to the design variables gives

$$
\begin{aligned}
\frac{\partial\tilde{\lambda}_k^*}{\partial\bar{\rho}_e} &= \boldsymbol{y}_k^T\boldsymbol{V}_k^T\left(\frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e} - \lambda_k^*\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\right)\boldsymbol{V}_k\boldsymbol{y}_k + 2\Delta\boldsymbol{f}_k^T\sum_{i=1}^{s}\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e}y_i\\
&+ \boldsymbol{z}_1^T\left(\boldsymbol{K}_0\frac{\partial\boldsymbol{u}_1}{\partial\bar{\rho}_e} - \frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_{0,k}\right) + \sum_{i=2}^{s}\boldsymbol{z}_i^T\left(\boldsymbol{K}_0\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e} + \frac{\partial\boldsymbol{K}}{\partial\bar{\rho}_e}\boldsymbol{t}_{i-1} + \Delta\boldsymbol{K}\frac{\partial\boldsymbol{t}_{i-1}}{\partial\bar{\rho}_e}\right)\\
&+ \sum_{i=1}^{s}\boldsymbol{q}_i^T\left(\frac{\partial\boldsymbol{t}_i}{\partial\bar{\rho}_e} - \left[\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i - \boldsymbol{u}_i\boldsymbol{u}_i^T\boldsymbol{M}\right)\frac{\partial\boldsymbol{u}_i}{\partial\bar{\rho}_e} - \frac{1}{2}\boldsymbol{u}_i\left(\boldsymbol{u}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{u}_i\right)\right]\left(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i\right)^{-3/2}\right)\\
&+ \sum_{i=1}^{s}\boldsymbol{w}_i^T\left(\frac{\partial\boldsymbol{v}_i}{\partial\bar{\rho}_e} - \frac{\partial\boldsymbol{t}_i}{\partial\bar{\rho}_e} + \sum_{j=1}^{k-1}\left(\frac{\partial\boldsymbol{t}_i^T}{\partial\bar{\rho}_e}\boldsymbol{M}\boldsymbol{\psi}_j\right)\boldsymbol{\psi}_j + \left(\boldsymbol{t}_i^T\frac{\partial\boldsymbol{M}}{\partial\bar{\rho}_e}\boldsymbol{\psi}_j\right)\boldsymbol{\psi}_j + \left(\boldsymbol{t}_i^T\boldsymbol{M}\boldsymbol{\psi}_j + \boldsymbol{\psi}_j\boldsymbol{t}_i^T\boldsymbol{M}\right)\frac{\partial\boldsymbol{\psi}_j}{\partial\bar{\rho}_e}\right)
\end{aligned}
$$

Collecting terms gives

$$\begin{aligned}
\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} &= \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k^* \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k - \boldsymbol{z}_1^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} + \sum_{i=2}^{s} \boldsymbol{z}_i^T \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} \\
&+ \frac{1}{2} \sum_{i=1}^{s} \boldsymbol{q}_i^T \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{u}_i \right) \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-3/2} + \sum_{i=1}^{s} \sum_{j=1}^{k-1} \boldsymbol{w}_i^T \boldsymbol{\psi}_j \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_j \right) \\
&+ \sum_{i=1}^{s} \left[\boldsymbol{z}_i^T \boldsymbol{K}_0 - \boldsymbol{q}_i^T (\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i - \boldsymbol{u}_i \boldsymbol{u}_i^T \boldsymbol{M}) \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-3/2} \right] \frac{\partial \boldsymbol{u}_i}{\partial \bar{\rho}_e} \\
&+ \sum_{i=1}^{s-1} \left[\boldsymbol{z}_{i+1}^T \Delta \boldsymbol{K} + \boldsymbol{q}_i^T - \boldsymbol{w}_i^T + \sum_{j=1}^{k-1} (\boldsymbol{w}_i^T \boldsymbol{\psi}_j) \boldsymbol{\psi}_j^T \boldsymbol{M} \right] \frac{\partial \boldsymbol{t}_i}{\partial \bar{\rho}_e} \\
&+ \left[\boldsymbol{q}_s^T - \boldsymbol{w}_s^T + \sum_{j=1}^{k-1} (\boldsymbol{w}_s^T \boldsymbol{\psi}_j) \boldsymbol{\psi}_j^T \boldsymbol{M} \right] \frac{\partial \boldsymbol{t}_s}{\partial \bar{\rho}_e} \\
&+ \sum_{i=1}^{s} \left[2 \Delta \boldsymbol{f}_k^T y_i + \boldsymbol{w}_i^T \right] \frac{\partial \boldsymbol{v}_i}{\partial \bar{\rho}_e} \\
&+ \sum_{i=1}^{s} \boldsymbol{w}_i^T \sum_{j=1}^{k-1} \left(\boldsymbol{t}_i^T \boldsymbol{M} \boldsymbol{\psi}_j + \boldsymbol{\psi}_j \boldsymbol{t}_i^T \boldsymbol{M} \right) \frac{\partial \boldsymbol{\psi}_j}{\partial \bar{\rho}_e}
\end{aligned} \tag{4.12}$$

Choosing the adjoint vectors according to the equations below eliminates derivatives of basis vectors, but there are not enough adjoints to eliminate derivatives of the eigenmodes (the last row of equation 4.12). Again, another s back- and forward substitutions are needed.

$$\begin{aligned}
\boldsymbol{w}_i &= -2 \Delta \boldsymbol{f}_k y_i && i = 1, ...s \\
\boldsymbol{q}_s &= \boldsymbol{w}_s - \sum_{j=1}^{k-1} (\boldsymbol{w}_s^T \boldsymbol{\psi}_j)(\boldsymbol{M} \boldsymbol{\psi}_j) \\
\boldsymbol{K}_0 \boldsymbol{z}_i &= (\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i - \boldsymbol{M} \boldsymbol{u}_i \boldsymbol{u}_i^T)(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i)^{-3/2} \boldsymbol{q}_i && i = s, ...1 \\
\boldsymbol{q}_i &= \boldsymbol{w}_i - \sum_{j=1}^{k-1} (\boldsymbol{w}_i^T \boldsymbol{\psi}_j)(\boldsymbol{M} \boldsymbol{\psi}_j) - \Delta \boldsymbol{K} \boldsymbol{z}_{i+1} && i = s-1, ...1
\end{aligned} \tag{4.13}$$

Inserting the adjoints (4.13) into equation 4.12 gives

$$\begin{aligned}
\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} &= \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k^* \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k - \boldsymbol{z}_1^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} + \sum_{i=2}^{s} \boldsymbol{z}_i^T \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} \\
&+ \frac{1}{2} \sum_{i=1}^{s} \boldsymbol{q}_i^T \boldsymbol{t}_i \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{t}_i \right) + \sum_{i=1}^{s} \sum_{j=1}^{k-1} \boldsymbol{w}_i^T \boldsymbol{\psi}_j \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_j \right) \\
&+ \sum_{i=1}^{s} \boldsymbol{w}_i^T \sum_{j=1}^{k-1} \left(\boldsymbol{t}_i^T \boldsymbol{M} \boldsymbol{\psi}_j + \boldsymbol{\psi}_j \boldsymbol{t}_i^T \boldsymbol{M} \right) \frac{\partial \boldsymbol{\psi}_j}{\partial \bar{\rho}_e}.
\end{aligned} \tag{4.14}$$

However, the sensitivities of the eigenmodes $\frac{\partial \boldsymbol{\psi}_j}{\partial \bar{\rho}_e}$ are not known so they are discarded, and the eigenvalues sensitivites become approximately

$$\begin{aligned}\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} \approx \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k^* \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k - \boldsymbol{z}_1^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} + \sum_{i=2}^{s} \boldsymbol{z}_i^T \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} \\ + \frac{1}{2} \sum_{i=1}^{s} \boldsymbol{q}_i^T \boldsymbol{t}_i \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{t}_i \right) + \sum_{i=1}^{s} \sum_{j=1}^{k-1} \boldsymbol{w}_i^T \boldsymbol{\psi}_j \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_j \right).\end{aligned} \tag{4.15}$$

4.2.3 Modified Eigenmode Orthogonalized Combined Approximation

Finally, in the following section the sensitivities of the eigenvalues when modified eigenmode orthogonalized CA is used, is derived. The eigenmodes with hats correspond to an old design iteration, and are constants in regard to the current design iteration. Hence their sensitivities to the current design is indeed zero. The optimization problem is given below

$$\begin{aligned} &\max\ g_0(\lambda_k^*) \\ &\text{subject to} \\ &\quad \boldsymbol{V}^T \boldsymbol{K} \boldsymbol{V} \boldsymbol{y}_k = \lambda_k^* \boldsymbol{V}^T \boldsymbol{M} \boldsymbol{V} \boldsymbol{y}_k \\ &\quad \boldsymbol{K}_0 \boldsymbol{u}_1 = \boldsymbol{M} \boldsymbol{\psi}_{0,k} \\ &\quad \boldsymbol{K}_0 \boldsymbol{u}_i = -\Delta \boldsymbol{K} \boldsymbol{t}_{i-1} && i = 2, ...s \\ &\quad \boldsymbol{t}_i = \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-1/2} && i = 1, ...s \\ &\quad \boldsymbol{v}_i = \boldsymbol{t}_i - \sum_{j=1}^{k-1} \left(\boldsymbol{t}_i^T \boldsymbol{M} \hat{\boldsymbol{\psi}}_j \right) \hat{\boldsymbol{\psi}}_j && i = 1, ...s \end{aligned}$$

As in section 4.2.1, one term per constraint (in total 3s terms) is added to the objective function in order to remove derivatives of basis vectors

$$\begin{aligned}\tilde{\lambda}_k^* = \lambda_k^* + \boldsymbol{z}_1^T \left(\boldsymbol{K}_0 \boldsymbol{u}_1 - \boldsymbol{M} \boldsymbol{\psi}_{0,k} \right) + \sum_{i=2}^{s} \boldsymbol{z}_i^T \left(\boldsymbol{K}_0 \boldsymbol{u}_i + \Delta \boldsymbol{K} \boldsymbol{t}_{i-1} \right) + \sum_{i=1}^{s} \boldsymbol{q}_i^T \left(\boldsymbol{t}_i - \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-1/2} \right) + \\ + \sum_{i=1}^{s} \boldsymbol{w}_i^T \left(\boldsymbol{v}_i - \boldsymbol{t}_i + \sum_{j=1}^{k-1} \left(\boldsymbol{t}_i^T \boldsymbol{M} \hat{\boldsymbol{\psi}}_j \right) \hat{\boldsymbol{\psi}}_j \right)\end{aligned}$$

Differentiating with respect to the design variables gives the expression

$$\begin{aligned}\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} = \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda^k \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k + 2 \Delta \boldsymbol{f}_k^T \sum_{i=1}^{s} \frac{\partial \boldsymbol{v}_i}{\partial \bar{\rho}_e} y_i \\ + \boldsymbol{z}_1^T \left(\boldsymbol{K}_0 \frac{\partial \boldsymbol{u}_1}{\partial \bar{\rho}_e} - \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} \right) + \sum_{i=2}^{s} \boldsymbol{z}_i^T \left[\boldsymbol{K}_0 \frac{\partial \boldsymbol{u}_i}{\partial \bar{\rho}_e} + \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} + \Delta \boldsymbol{K} \frac{\partial \boldsymbol{t}_{i-1}}{\partial \bar{\rho}_e} \right] \\ + \sum_{i=1}^{s} \boldsymbol{q}_i^T \left(\frac{\partial \boldsymbol{t}_i}{\partial \bar{\rho}_e} - \left[\left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i - \boldsymbol{u}_i \boldsymbol{u}_i^T \boldsymbol{M} \right) \frac{\partial \boldsymbol{u}_i}{\partial \bar{\rho}_e} - \frac{1}{2} \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{u}_i \right) \right] \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-3/2} \right) \\ + \sum_{i=1}^{s} \boldsymbol{w}_i^T \left[\frac{\partial \boldsymbol{v}_i}{\partial \bar{\rho}_e} - \frac{\partial \boldsymbol{t}_i}{\partial \bar{\rho}_e} + \sum_{j=1}^{k-1} \left(\frac{\partial \boldsymbol{t}_i^T}{\partial \bar{\rho}_e} \boldsymbol{M} \hat{\boldsymbol{\psi}}_j \right) \hat{\boldsymbol{\psi}}_j + \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \hat{\boldsymbol{\psi}}_j \right) \hat{\boldsymbol{\psi}}_j \right]\end{aligned}$$

Collecting terms gives

$$\begin{aligned}
\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} &= \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda^k \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k - \boldsymbol{z}_1^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} + \sum_{i=2}^{s} \boldsymbol{z}_i^T \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} \\
&+ \frac{1}{2} \sum_{i=1}^{s} \boldsymbol{q}_i^T \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{u}_i \right) \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-3/2} + \sum_{i=1}^{s} \sum_{j=1}^{k-1} \boldsymbol{w}_i^T \hat{\boldsymbol{\psi}}_j \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \hat{\boldsymbol{\psi}}_j \right) \\
&+ \sum_{i=1}^{s} \left[\boldsymbol{z}_i^T \boldsymbol{K}_0 - \boldsymbol{q}_i^T \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i - \boldsymbol{u}_i \boldsymbol{u}_i^T \boldsymbol{M} \right) \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-3/2} \right] \frac{\partial \boldsymbol{u}_i}{\partial \bar{\rho}_e} \\
&+ \sum_{i=1}^{s-1} \left[\boldsymbol{z}_{i+1}^T \Delta \boldsymbol{K} + \boldsymbol{q}_i^T - \boldsymbol{w}_i^T + \sum_{j=1}^{k-1} (\boldsymbol{w}_i^T \hat{\boldsymbol{\psi}}_j) \hat{\boldsymbol{\psi}}_j^T \boldsymbol{M} \right] \frac{\partial \boldsymbol{t}_i}{\partial \bar{\rho}_e} \\
&+ \left[\boldsymbol{q}_s^T - \boldsymbol{w}_s^T + \sum_{j=1}^{k-1} (\boldsymbol{w}_s^T \hat{\boldsymbol{\psi}}_j) \hat{\boldsymbol{\psi}}_j^T \boldsymbol{M} \right] \frac{\partial \boldsymbol{t}_s}{\partial \bar{\rho}_e} \\
&+ \sum_{i=1}^{s} \left[2 \Delta \boldsymbol{f}_k^T y_i + \boldsymbol{w}_i^T \right] \frac{\partial \boldsymbol{v}_i}{\partial \bar{\rho}_e}
\end{aligned}$$

Choosing the adjoint vectors according to the equations below eliminates derivatives of basis vectors, and in contrast to the sensitivities of CAEEON no terms including derivatives of the eigenmodes are present. The sensitivities are hence consistent with the reduced order model.

$$\begin{aligned}
\boldsymbol{w}_i &= -2 \Delta \boldsymbol{f}_k y_i && i = 1, ...s \\
\boldsymbol{q}_s &= \boldsymbol{w}_s - \sum_{j=1}^{k-1} (\boldsymbol{w}_s^T \hat{\boldsymbol{\psi}}_j)(\boldsymbol{M} \hat{\boldsymbol{\psi}}_j) \\
\boldsymbol{K}_0 \boldsymbol{z}_i &= (\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i - \boldsymbol{M} \boldsymbol{u}_i \boldsymbol{u}_i^T)(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i)^{-3/2} \boldsymbol{q}_i && i = s, ...1 \\
\boldsymbol{q}_i &= \boldsymbol{w}_i - \sum_{j=1}^{k-1} (\boldsymbol{w}_i^T \hat{\boldsymbol{\psi}}_j)(\boldsymbol{M} \hat{\boldsymbol{\psi}}_j) - \Delta \boldsymbol{K} \boldsymbol{z}_{i+1} && i = s-1, ...1
\end{aligned}$$

The sensitivities become

$$\begin{aligned}
\frac{\partial \tilde{\lambda}_k^*}{\partial \bar{\rho}_e} &= \boldsymbol{y}_k^T \boldsymbol{V}_k^T \left(\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} - \lambda_k^* \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \right) \boldsymbol{V}_k \boldsymbol{y}_k - \boldsymbol{z}_1^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{\psi}_{0,k} + \sum_{i=2}^{s} \boldsymbol{z}_i^T \frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} \boldsymbol{t}_{i-1} \\
&+ \frac{1}{2} \sum_{i=1}^{s} \boldsymbol{q}_i^T \boldsymbol{t}_i \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \boldsymbol{t}_i \right) + \sum_{i=1}^{s} \sum_{j=1}^{k-1} \boldsymbol{w}_i^T \hat{\boldsymbol{\psi}}_j \left(\boldsymbol{t}_i^T \frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} \hat{\boldsymbol{\psi}}_j \right)
\end{aligned} \tag{4.16}$$

Chapter 5

Problem formulation

In this chapter the optimization problem used to test the reduced order methods discussed in chapter 3 is presented. In particular, the aim is to maximize the fundamental frequency under a volume constraint. In addition to presenting the optimization problem, the sensitivities of the objective function and the constraints are derived. Finally, the geometry of the design domain is presented. It should be noted that the reduced order methods are not limited to this specific problem, and can be applied to any generalized eigenproblem in topology optimization.

5.1 Objective and constraints

The idea of manipulating a structure's eigenfrequencies using topology optimization was first presented by Días and Kikuchi in [7], where the homogenization method from [3] was used to maximize the fundamental frequency of an existing structure. Manipulating the eigenfrequencies has been applied to other problems, for example by Dalklint in [8] to create crystals with phononic band gaps. In this work maximizing the fundamental frequency will be considered as a test problem. The motivation is that by maximizing the fundamental frequency resonance can be avoided.

As suggested by Torii and Faria [28] the p-norm is used as a smooth approximation of the fundamental frequency, which they showed is always differentiable. In other words, the fundamental frequency $\omega_{\min}$ is approximated as

$$\omega_{\min} \approx ||\boldsymbol{\omega}||_{-p} = \left(\sum_{i=1}^{n_e} \omega_i^{-p} \right)^{-\frac{1}{p}}, \tag{5.1}$$

where p is the penalization exponent (also known as the p-value), ω_i are the eigenfrequencies, and n_e is the number of eigenvalues included in the approximation. Although the form above is very compact, it may be hard to interpret. The p-norm may be rewritten in the form below, allowing for a richer interpretation

$$||\boldsymbol{\omega}||_{-p} = \frac{1}{\left(\left(\frac{1}{\omega_1}\right)^p + \left(\frac{1}{\omega_2}\right)^p + ... \left(\frac{1}{\omega_{n_\mathrm{e}}}\right)^p \right)^{\frac{1}{p}}} = \frac{\omega_1}{\left(1 + \left(\frac{\omega_1}{\omega_2}\right)^p + ... \left(\frac{\omega_1}{\omega_{n_\mathrm{e}}}\right)^p \right)^{\frac{1}{p}}}$$

A few remarks can be made about the approximation. Firstly, due to the eigenfrequencies being positive numbers the denominator is always larger than one, meaning the approximation is conservative, or $||\boldsymbol{\omega}||_{-p} < \omega_1$.

Secondly, the larger an eigenfrequency is compared to the fundamental frequency ω_1, the less it influences the approximation. If for example an eigenfrequency is two times larger than the fundamental frequency, then a p-value of eight will make its influence $0.5^8 \approx 0.004$, compared to

the fundamental frequency which has the weight 1. In other words, only eigenfrequencies which are very close to the fundamental frequency will influence the approximation.

Thirdly, the higher the p-value, the less other eigenvalues influence the approximation, and the more accurately it approximates the fundamental frequency. As p approaches infinity this approximation becomes increasingly accurate. However, using a too large p can cause problems due to numerical round-off errors. Thus, a moderate value of about eight is usually chosen. Lastly, the number of eigenvalues considered needs to be larger than one. Torri and Faria found that using a small amount of eigenvalues, in the range of five to ten, is usually sufficient.

The objective is to maximize the fundamental frequency, which is equivalent to minimizing minus one times the fundamental frequency. Using the p-norm this gives the objective function

$$g_0 = -||\boldsymbol{\omega}||_{-p}$$

The total volume of the design can be expressed as the scalar product between the thresholded densities and the vector of element volumes $\boldsymbol{v}$ who's ith component is the volume of element i. The volume of the structure for a certain design $\bar{\boldsymbol{\rho}}$, V, and total available volume, V_0, can then be expressed as

$$V = \sum_{i=1}^{n_{\text{elm}}} v_i \bar{\rho}_i = \boldsymbol{v}^T \bar{\boldsymbol{\rho}} \quad \text{and} \quad V_0 = \sum_{i=1}^{n_{\text{elm}}} v_i = \boldsymbol{v}^T \mathbf{1},$$

where $\mathbf{1}$ is a vector of ones. A constraint is added on the total volume of the structure, which should be no more than a portion of the total available volume. Introducing the volume fraction $\alpha \in (0, 1]$, this can be written as

$$g_1 = V - \alpha V_0 \leq 0. \tag{5.2}$$

The design variables are normalized such that they are contained in the interval from zero to one, which is often referred to as box constraints. Thus, and element with density $\bar{\rho}_i = 1$ is interpreted as filled, whereas an element with density $\bar{\rho}_i << 1$ is interpreted as void. The optimization problem may now be stated as

$$(\mathbb{TO}) : \begin{cases} \min\limits_{\boldsymbol{z}} \; g_0 \\ \text{subject to} \\ \quad g_1 \leq 0 \\ \quad 0 \leq z_i \leq 1 \quad i = 1...n_{\text{elm}} \end{cases} . \tag{5.3}$$

The SIMP scheme is used to penalize intermediate density values, which means that the elasticity modulus for the element with index e is given by the SIMP-scheme (2.24) where the penalty exponent q is chosen as 3. The element stiffness matrix may then be expressed as

$$\boldsymbol{K}^e = \left(E_{\min} + (E_{\max} - E_{\min})\bar{\rho}_e^3\right) \boldsymbol{K}_0^e \tag{5.4}$$

where $\boldsymbol{K}_0^e$ is the element stiffness matrix defined in equation 2.10 with unit elasticity modulus. A well known consequence of using SIMP in eigenfrequency optimization is the appearance of localized low energy modes which appear in low density regions where the stiffness is very small compared to the mass. A remedy to this is suggested by Du and Olhoff in [29], where the mass in low density regions is scaled according to the scheme below,

$$\boldsymbol{M}^e = \begin{cases} \bar{\rho}_e \boldsymbol{M}_0^e & \bar{\rho}_e \geq 0.1 \\ \left(c_1 \bar{\rho}_e^6 + c_2 \bar{\rho}_e^7\right) \boldsymbol{M}_0^e & \bar{\rho}_e < 0.1 \end{cases}, \tag{5.5}$$

where $\boldsymbol{M}_0^e$ is the element mass matrix defined in equation 2.14 and the coefficients c_1 and c_2 are chosen such that the interpolation is $\mathcal{C}^1$ (meaning it is continuous and has a piecewise continuous derivative). Choosing $c_1 = 6 \cdot 10^5$ and $c_2 = -5 \cdot 10^6$ fulfills this requirement.

5.2 Sensitivity Analysis

As discussed in section 2.4 gradient based algorithms are used to solve the optimization problem, as such the sensitivities of the objective function and the constraint(s) are needed. Using the chain rule for the sensitivity of the objective function gives

$$\frac{\partial g_0}{\partial z_e} = \sum_{i=1}^{n_\mathrm{e}} \frac{\partial g_0}{\partial \omega_i} \frac{\partial \omega_i}{\partial z_e}, \tag{5.6}$$

where the first term in the sum can be found by considering the derivative of $||\boldsymbol{\omega}||_{-p}$, see equation 5.1

$$\frac{\partial ||\boldsymbol{\omega}||_{-p}}{\partial \omega_i} = \frac{\partial}{\partial \omega_i} \left(\sum_{i=1}^{n_\mathrm{e}} \omega_i^{-p} \right)^{-\frac{1}{p}} = \frac{-1}{p} \left(\sum_{i=1}^{n_\mathrm{e}} \omega_i^{-p} \right)^{-\frac{1}{p}(p+1)} \left(-p\omega_i^{-p-1} \right) = \left(\frac{\omega_i}{||\boldsymbol{\omega}||_{-p}} \right)^{-p-1}$$

Expanding the second term in the sum using the chain rule and $\frac{\partial \lambda_i}{\partial z_e} = \frac{\partial \omega_i^2}{\partial z_e} = 2\omega_i \frac{\partial \omega_i}{\partial z_e}$ gives

$$\frac{\partial \omega_i}{\partial z_e} = \frac{1}{2\omega_i} \sum_{j=1}^{n_\mathrm{elm}} \frac{\partial \lambda_i}{\partial \bar{\rho}_j} \frac{\partial \bar{\rho}_j}{\partial \rho_j} \frac{\partial \rho_j}{\partial z_e}.$$

The density filter (2.25) gives $\frac{\partial \rho_j}{\partial z_e} = M_{je}^f$, and the smooth Heaviside approximation (2.26) may be differentiated using hyperbolic trigonometric identities

$$\frac{\partial \bar{\rho}_j}{\partial \rho_j} = \frac{\partial H(\rho_j)}{\partial \rho_j} = \frac{\partial}{\partial \rho_j} \frac{\tanh(\beta\eta) + \tanh(\beta(\rho_j - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))} = \frac{\beta \cosh^{-2}(\beta(\rho_j - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))},$$

where it was used that $\frac{\partial}{\partial x} \tanh(x) = \cosh^{-2}(x)$. The second term in equation 5.6 can then be expanded to

$$\frac{\partial \omega_i}{\partial z_e} = \frac{1}{2\omega_i} \sum_{j=1}^{n_\mathrm{elm}} \frac{\partial \lambda_i}{\partial \bar{\rho}_j} \frac{\beta M_{je} \cosh^{-2}(\beta(\rho_j - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))}.$$

The sensitivity of the objective function is finally given by

$$\frac{\partial g_0}{\partial z_e} = -\left(\frac{\omega_i}{||\boldsymbol{\omega}||_{-p}} \right)^{-p-1} \frac{1}{2\omega_i} \sum_{j=1}^{n_\mathrm{elm}} \frac{\partial \lambda_i}{\partial \bar{\rho}_j} \frac{\beta M_{je} \cosh^{-2}(\beta(\rho_j - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))},$$

where the eigenvalues sensitivity, $\frac{\partial \lambda_i}{\partial \bar{\rho}_j}$ depend on what method is used to compute them, see chapter 4. Lastly, the stiffness and mass matrices sensitivities and the volume constraints sensitivity are given below

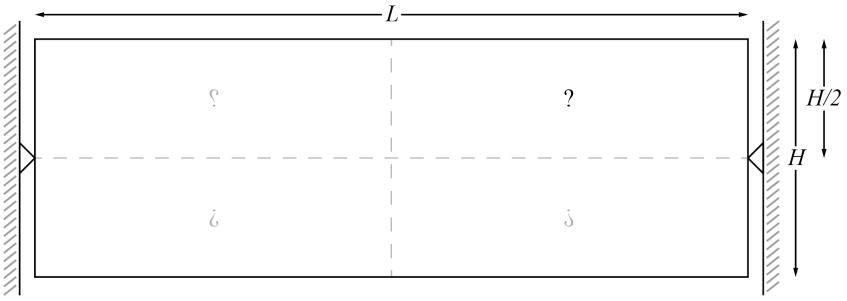
$$\frac{\partial \boldsymbol{K}}{\partial \bar{\rho}_e} = 3(E_\mathrm{max} - E_\mathrm{min})\bar{\rho}_e^2 \boldsymbol{K}_0^e$$

$$\frac{\partial \boldsymbol{M}}{\partial \bar{\rho}_e} = \begin{cases} \boldsymbol{M}_0^e & \bar{\rho}_e \geq 0.1 \\ \left(6c_1\bar{\rho}_e^5 + 7c_2\bar{\rho}_e^6\right) \boldsymbol{M}_0^e & \bar{\rho}_e < 0.1 \end{cases}$$

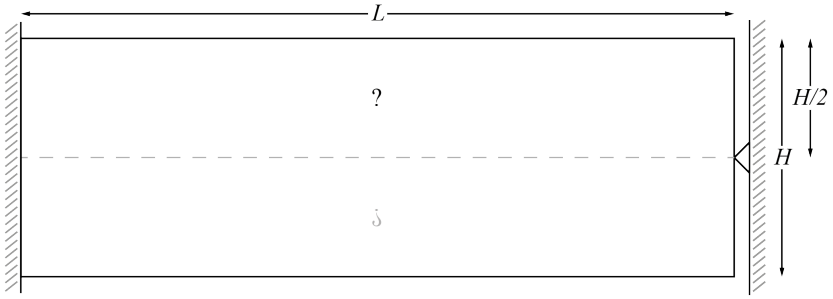
$$\frac{\partial g_1}{\partial z_e} = \sum_{j=1}^{n_\mathrm{elm}} \frac{\partial g_1}{\partial \bar{\rho}_j} \frac{\partial \bar{\rho}_j}{\partial z_e} = \sum_{j=1}^{n_\mathrm{elm}} \frac{v_j \beta M_{je} \cosh^{-2}(\beta(\rho_j - \eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))}$$

5.3 Geometry and design domain

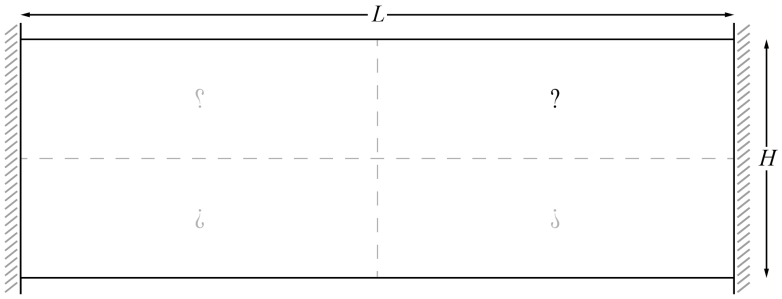
The design domains used to test the methods are shown in figures 5.1a-5.1c. Since the designs are expected to have some symmetry properties, they are subject to symmetry constraints. This also reduces the computational effort of solving the resulting optimization problem slightly, although the majority of the time is spent on the FEA. Three different sets of boundary conditions are used in order to investigate the numerical methods, hence three different design domains are shown.



(a) The design domain is the upper right quadrant with the black question mark. The densities in the remaining quadrants are given by reflecting the densities in the upper right quadrant about the dashed lines. The reflection is indicated by the reflected, gray question marks. The beam is referred to as the symmetric A-beam.



(b) The design domain is the upper half with the black question mark. The densities in lower half are given by reflecting the densities in the upper half about the dashed line. The reflection is indicated by the reflected, gray question mark. The beam is referred to as the symmetric B-beam.



(c) The design domain is the upper right quadrant with the black question mark. The densities in the remaining quadrants are given by reflecting the densities in the upper right quadrant about the dashed lines. The reflection is indicated by the reflected, gray question marks. The beam is referred to as the symmetric C-beam.

Figure 5.1: The figures depict three beams with dimensions $L \times H$ and unknown density. For the nodes connected to walls or triangles the boundary conditions are zero displacement in both the horizontal and vertical direction.

5.4 Finite difference approximation

In order to verify that the analytical expressions for sensitivities are correct, or to evaluate the error, a finite difference approximation is used. The approximation is based on well known explicit Euler scheme, which approximates the scalar function f of one variable at x by

$$\left.\frac{\partial f}{\partial x}\right|_x \approx \frac{f(x+h)-f(x)}{h}.$$

This approximation closely resembles the derivatives definition, except the infinitesimal small h is replaced by a finite difference h, since finite precision computers are used. In order to avoid numerical round off errors h is often chosen to be around the square root of machine tolerance, which is about 10^{-8}. For a function f of multiple variables the gradient at $\boldsymbol{x}$ may be approximated as

$$\left.\frac{\partial f}{\partial x_i}\right|_{\boldsymbol{x}} \approx \frac{f(\boldsymbol{x}+\mathbf{e}_i h)-f(\boldsymbol{x})}{h},$$

where $\mathbf{e}_i$ is the vector of unit length with a 1 at index i. Evaluating the gradient at every x_i can be very expensive if the cost of evaluating the function is, thus only a few x_i are chosen to verify that the sensitivities in those particular directions are correct.

Chapter 6

Numerical tests

The goal of this chapter is to investigate the methods performance. The optimization problem is given by equation 5.3, and is tested on a set of reference geometries consisting of thin beams, see figures 5.1a-5.1c. The length and height of the beams are set to 8000 and 1000 mm respectively, and are discretized using isoparametric quadrilateral elements. Symmetry conditions are imposed on the designs in order to reduce the computational effort of solving the optimization problem, and to guarantee symmetry in the designs. The material parameters used in all tests are the same, and are given by table 6.1. The volume fraction α is set to 50%.

The implementation is written in MATLAB. Krister Svanberg's GCMMA code is used to solve the optimization problem, and MATLAB's `eigs`-function is used to solve the generalized eigenproblem and the reduced order generalized eigenproblem. When the full model is used to solve the generalized eigenproblem, a Cholesky factorization of the stiffness matrix using MATLAB's `chol` is first computed and then used in `eig`. This factorization is then stored to build a reduced order model.

6.1 Verifying sensitivities

Firstly, the analytical expressions for the sensitivities derived in previous chapters are verified. This can be done by comparing the sensitivities computed using the expressions derived in Chapter 4 to finite-difference approximations. These approximations can be computed by first choosing a set of n_{check} design variables, with indices $I = \{i_1, i_2, ..., i_{n_{\text{check}}}\}$, and then computing the finite difference approximations of the sensitivities

$$\frac{\partial \omega_k^*}{\partial z_{i_j}} \approx \left(\frac{\partial \omega_k^*}{\partial z_{i_j}}\right)^{\text{fda}} = \frac{\omega_k^*(\boldsymbol{z} + \mathbf{e}_{i_j} h) - \omega_k^*(\boldsymbol{z})}{h}$$
$$\frac{\partial g_0}{\partial z_{i_j}} \approx \left(\frac{\partial g_0}{\partial z_{i_j}}\right)^{\text{fda}} = \frac{g_0(\boldsymbol{z} + \mathbf{e}_{i_j} h) - g_0(\boldsymbol{z})}{h}$$

For example, consider verifying the analytical sensitivities for the kth eigenfrequency for a reduced order model using CAE. The first step is to compute the basis vectors according to equation 3.16, after which the approximated eigenfrequency is found by solving the reduced order generalized eigenproblem (3.11). The design variable with index i_j is then perturbed by h, after which the basis vectors are computed again according to equations 3.16 and the eigenfrequency is found by solving equation 3.11. Now a finite difference approximation of the kth sensitivity may be computed according to the equation above, where $\omega_k^*(\boldsymbol{z} + \mathbf{e}_{i_j} h)$ and $\omega_k^*(\boldsymbol{z})$ are the approximated eigenfrequencies at the perturbed and unperturbed design, respectively. The result may be compared to the analytical sensitivity of the kth eigenfrequency, which is found by first solving for the adjoints according to equations 4.9 and then inserting them into equation 4.10.

Table 6.1: Material parameters used in the optimization.

Material parameters	
$E_{\max}$	210 GPa
ν	0.3
t	10 mm
ρ_m	7800 kgm^{-3}

Each sensitivity test consists of performing 1000 design cycles (n_{its}) using the method of moving asymptotes. The A-beam is used as the design domain, given by figure 5.1a, which is discretized into a 160×20 grid of quadrilateral isoparametric elements. The filter radius is set to $5L/160$, the p-value is kept at 8 and the Heaviside projection is removed by keeping β at 1. A factorization is performed every 10 design cycles or if the sine of the angle between the old and current design supersedes the tolerance $\alpha_{\text{tol}} = 10 \cdot 10^{-3}$. The tolerance on the error in the design is set to $\delta_{\text{tol}} = 0.1$. The methods were tested using 4 and 8 eigenvalues and 2, 4 and 8 basis vectors.

A finite difference approximation of the eigenfrequencies and objective function's sensitivities are computed at each design cycle. Since each approximation requires solving the eigenproblem an additional time, only a few design variables are chosen. In this case the elements with index 10, 20, 50 and 150. The step-size is set to $h = 10^{-5}$.

The root-mean-square-norm (RMS-norm) of the difference between the eigenfrequencies sensitivity and the finite difference approximation is computed, giving an average over all iterations and the subset of design variables for which the sensitivities are tested. The RMS-norm of the difference between the objective functions sensitivity and the finite difference approximation is also computed, giving an average over the chosen design variables, with one point for each design cycle. The expressions are given below.

$$||\Delta\omega_k^*||_{RMS} = \sqrt{\frac{1}{n_{\text{its}}} \sum_{l=1}^{n_{\text{its}}} \frac{1}{n_{\text{check}}} \sum_{j=1}^{n_{\text{check}}} \left| \left(\frac{\partial \omega_k^*}{\partial z_{i_j}} \right)_l - \left(\frac{\partial \omega_k^*}{\partial z_{i_j}} \right)_l^{\text{fda}} \right|}$$

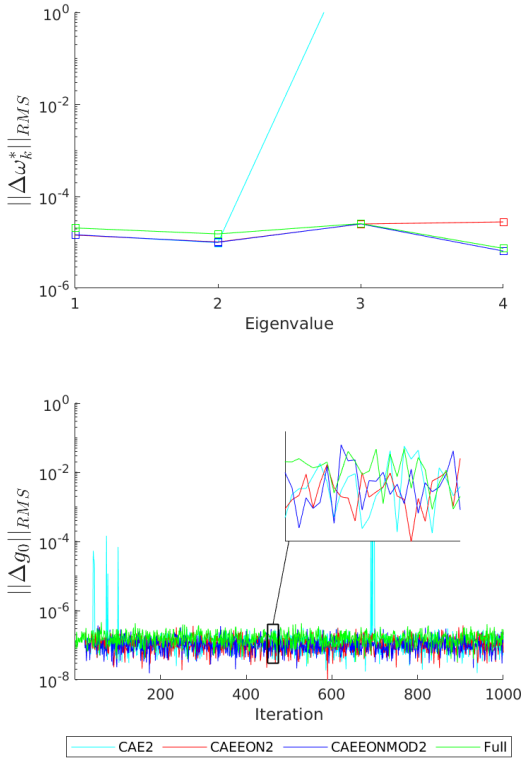
$$||\Delta g_0||_{RMS} = \sqrt{\frac{1}{n_{\text{check}}} \sum_{j=1}^{n_{\text{check}}} \left| \left(\frac{\partial g_0}{\partial z_{i_j}} \right) - \left(\frac{\partial g_0}{\partial z_{i_j}} \right)^{\text{fda}} \right|}$$

6.1.1 Results

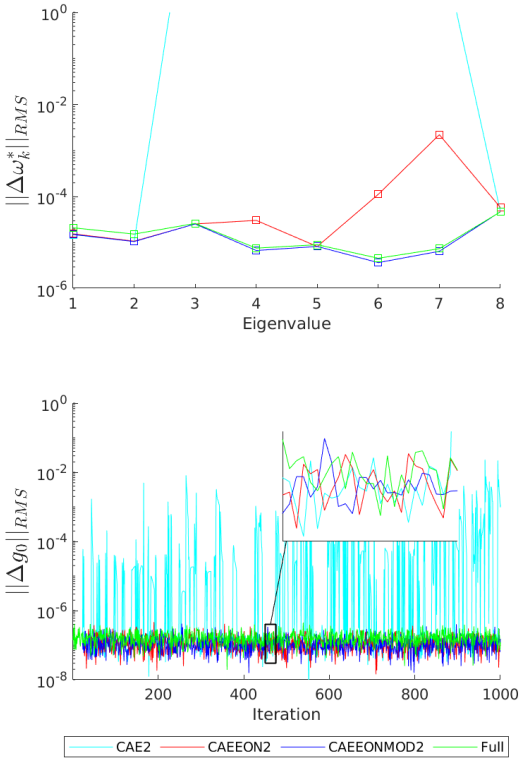
Figures 6.1a to 6.3b show plots of the RMS-norms of the errors for different choices of number of eigenvalues and number of basis vectors. For two basis vectors and four eigenvalues, see figure 6.1a, CAE seems to follow the full models error, except for a few iterations. CAEEON seems to follow the full models errors with a slight deviation for the fourth eigenvalue, while the error in the objective function seems to follow the full model. Lastly, CAEEONmod seems to follow the full models errors exactly.

The deviation is increased when eight eigenvalues are considered, see figure 6.1b. For CAE, the first two eigenvalues are still accurate, while the higher order eigenmodes are not. This is also seen in the objective function's error. For CAEEON the errors in the higher order eigenvalues also deviate, whereas the objective function's error seems to follow the full models.

Moving on to the next test case where four basis vectors are used, see figure 6.2a - 6.2b. The only notable difference from the previous case is that CAE seems to struggle more, even for four eigenvalues. There is also a slight increase in the error of the sixth eigenvalue for CAEEON. The same seems to happen when the number of basis vectors is increased to eight, see figures 6.3a - 6.3b.

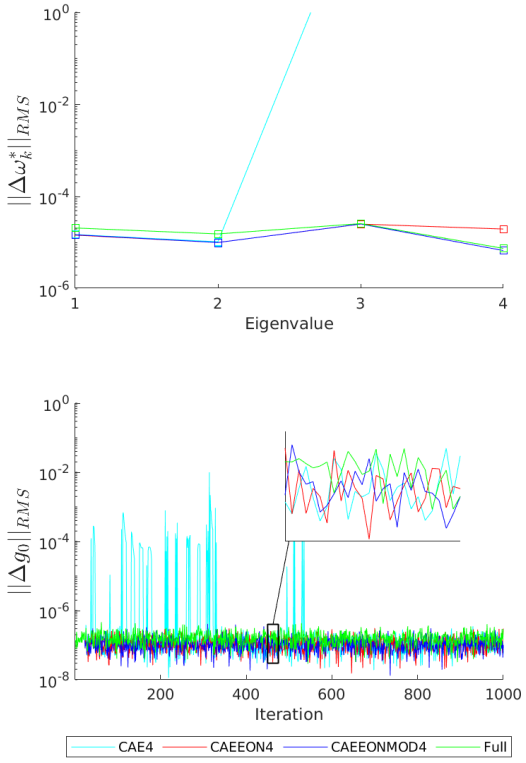


(a) The figure shows the errors when 4 eigenvalues are used in the p-norm.

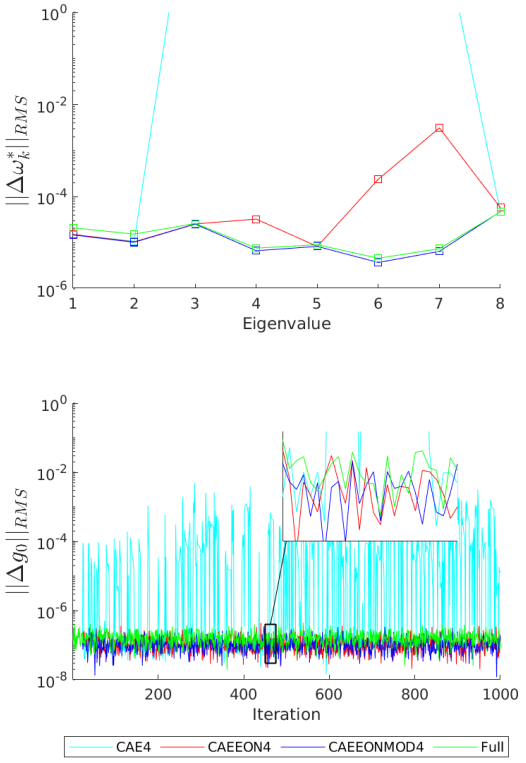


(b) The figure shows the errors when 8 eigenvalues are used in the p-norm.

Figure 6.1: The figures show the difference between the analytical sensitivities and the finite difference approximation when 2 basis vectors are used. On the top of respective figure is the error in the eigenvalues, and on the bottom is the error in the sensitivities. As a reference the error when the full formulation is used is plotted in a green line.

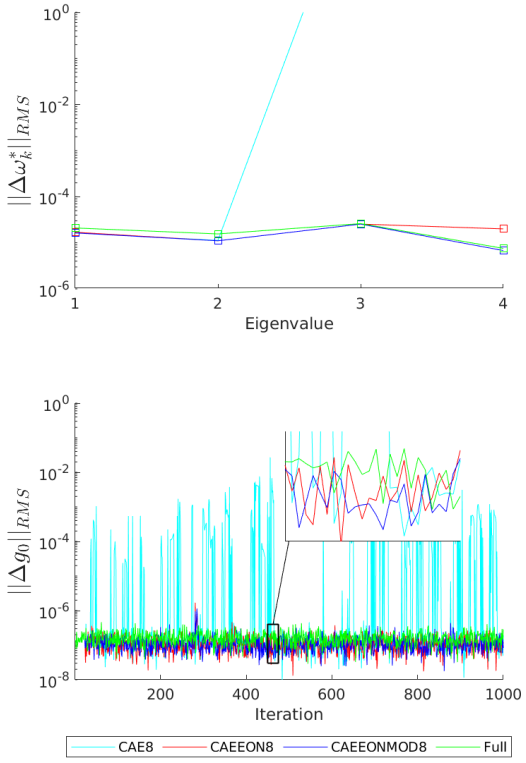


(a) The figure shows the errors when 4 eigenvalues are used in the p-norm.

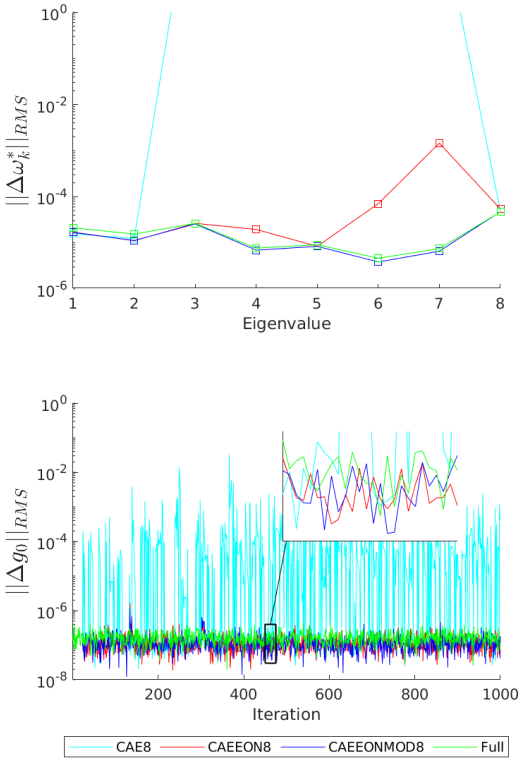


(b) The figure shows the errors when 8 eigenvalues are used in the p-norm.

Figure 6.2: The figures show the difference between the analytical sensitivities and the finite difference approximation when 4 basis vectors are used. On the top of respective figure is the error in the eigenvalues, and on the bottom is the error in the sensitivities. As a reference the error when the full formulation is used is plotted in a green line.



(a) The figure shows the errors when 4 eigenvalues are used in the p-norm.



(b) The figure shows the errors when 8 eigenvalues are used in the p-norm.

Figure 6.3: The figures show the difference between the analytical sensitivities and the finite difference approximation when 8 basis vectors are used. On the top of respective figure is the error in the eigenvalues, and on the bottom is the error in the sensitivities. As a reference the error when the full formulation is used is plotted in a green line.

6.2 Performance

In this section the overall performance of the reduced order methods is investigated, and is compared to the full model. The methods are tested using a grid of 400×50 elements, using eight eigenfrequencies and two, four and six basis vectors. The number of factorizations is counted throughout the optimization, and their proportion is plotted in figures 6.4a - 6.4c. The angle between consecutive iterations is plotted in figures 6.5a - 6.5c. Lastly, the proportion of factorizations and the objective function at the final iteration are given in table 6.2 and 6.3. The geometries used are the symmetric beams, see figures 5.1a - 5.1c. The filter radius is set to $3L/400$, and α_{tol} and δ_{tol} is the same as before.

6.2.1 Results

Figures 6.4a-6.4c show what proportion of iterations uses a factorization throughout the optimization. For the first 25-50 iterations a factorization is almost always performed, after the first 100 iterations however the design changes are small enough to never require a factorization. The design changes throughout the optimization are shown in figures 6.5a-6.5c. The number of factorizations after all 1000 iterations is shown in table 6.2. Note that generally, increasing the number of basis vectors reduces the number of factorizations. The change is most significant going from two basis vectors to four, and very subtle going from four to six.

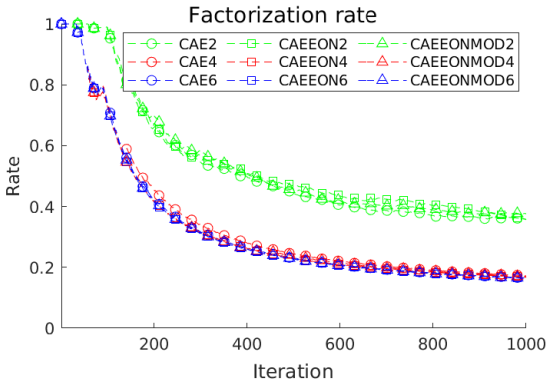
Table 6.3 shows the objective function's value at the final iteration. The difference is on the magnitude of a few parts per thousand. The results suggest that while increasing the number of basis vectors can reduce the amount of work, it does not increase the quality of the results, at least not noticeably. Lastly, figures 6.6a-6.8d show the final designs. The differences in design are very subtle.

Table 6.2: The table shows the proportions of the total amount of iterations where a factorization was performed (in percent), for the three beams.

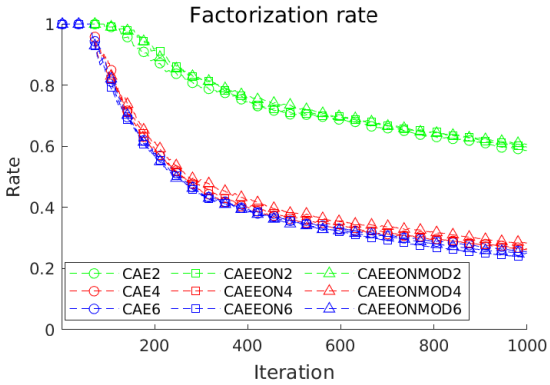
	A			B			C		
# Basis vectors	2	4	6	2	4	6	2	4	6
CAE	35.9	17.3	16.5	58.7	26.5	25.5	21.0	15.3	15.3
CAEEON	35.7	16.8	16.4	59.9	26.2	23.6	21.7	15.5	15.4
CAEEONMOD	37.6	16.9	16.4	60.7	28.3	25.0	22.4	15.4	15.5

Table 6.3: The table shows the value of the objective function (in Hz) at the final design cycle for the three beams. The top row shows the value when the full model is used, and the bottom three rows show the difference between the full model and the reduced model.

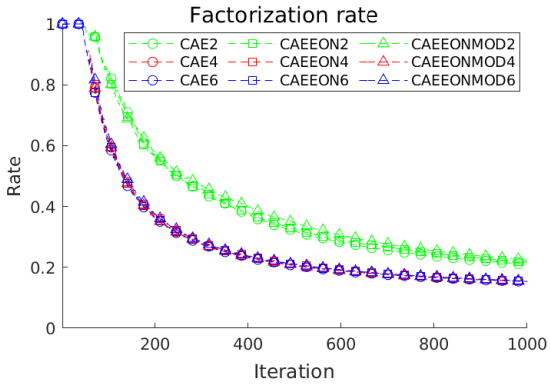
	A			B			C		
# Basis vectors	2	4	6	2	4	6	2	4	6
Full	267.08			434.30			644.63		
CAE	-0.01	-0.00	-0.35	+0.06	+0.01	-0.10	+0.14	-0.03	-0.03
CAEEON	-0.06	-0.02	-0.35	+0.13	+0.35	+0.31	+0.20	+0.03	-0.06
CAEEONMOD	+0.05	-0.11	-0.06	+0.11	+0.24	+0.22	+0.02	+0.09	+0.10



(a) The figure shows the proportions of design cycles where a factorization was performed for the symmetric A-beam, see figure 5.1a.

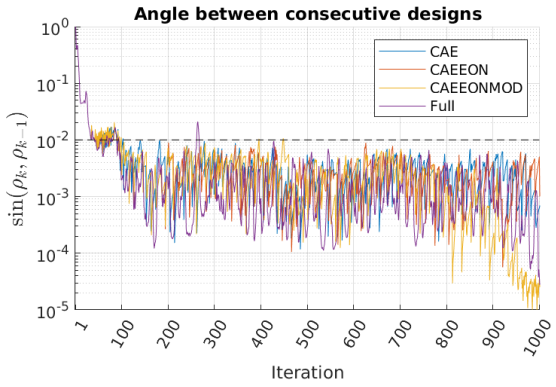


(b) The figure shows the proportions of design cycles where a factorization was performed for the symmetric B-beam, see figure 5.1b.

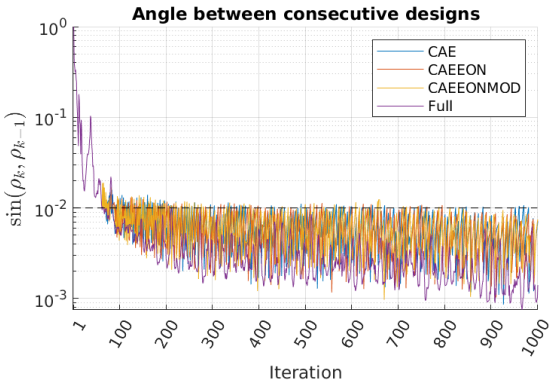


(c) The figure shows the proportions of design cycles where a factorization was performed for the symmetric C-beam, see figure 5.1c.

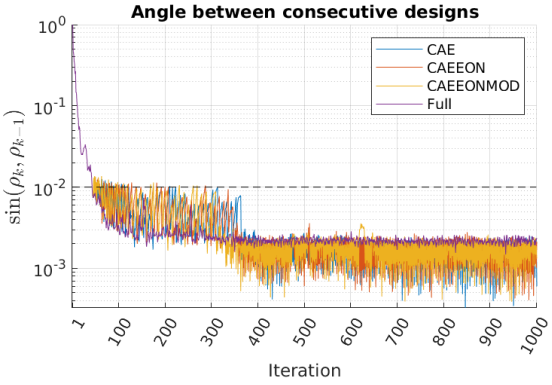
Figure 6.4: The figures show the proportions of design cycles where a factorization was performed throughout the optimization. The methods were tested on the three different beams, using two, four and six basis vectors and eight eigenvalues.



(a) The figure shows the sine of the angle between consecutive designs for the A-beam. The black dashed line shows the tolerance α_{tol} used to determine when a factorization should be performed.

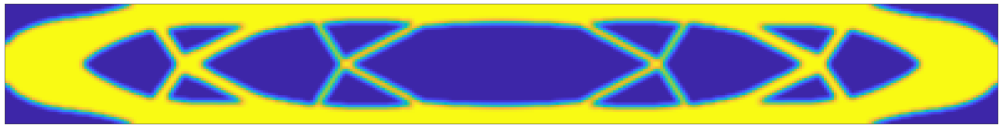


(b) The figure shows the sine of the angle between consecutive designs for the B-beam. The black dashed line shows the tolerance α_{tol} used to determine when a factorization should be performed.

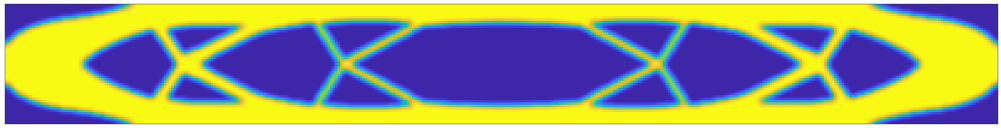


(c) The figure shows the sine of the angle between consecutive designs for the C-beam. The black dashed line shows the tolerance α_{tol} used to determine when a factorization should be performed.

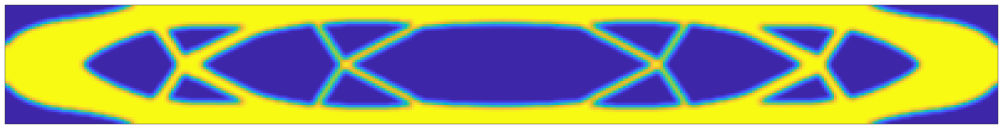
Figure 6.5: The figures show the sine of the angle between consecutive designs throughout the optimization. Four basis vectors and eight eigenvalues was used.



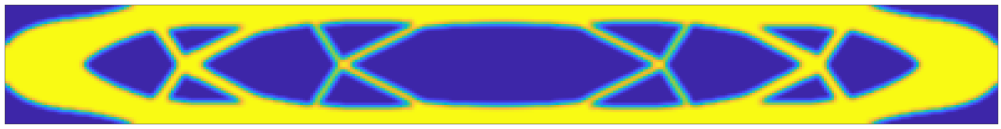
(a) Final design using full model.



(b) Final design using reduced order model with CAE.

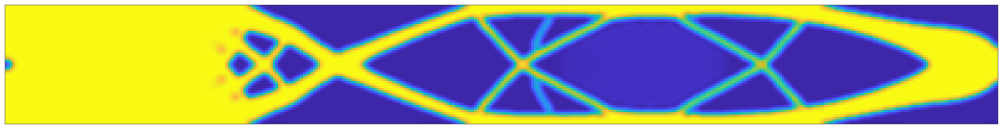


(c) Final design using reduced order model with CAEEON.

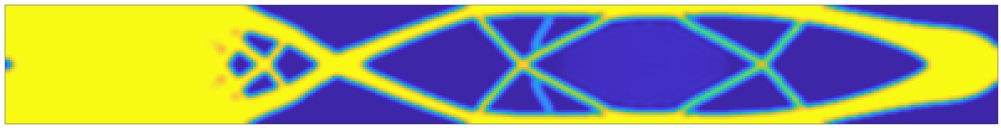


(d) Final design using reduced order model with CAEEONmod.

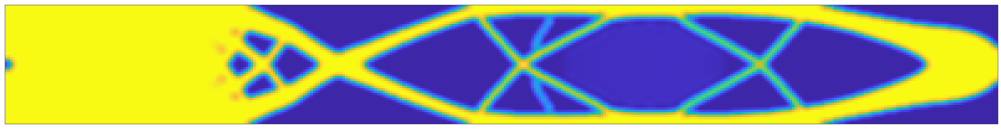
Figure 6.6: The figures show the final design for the full model and the different reduced order methods without Heaviside thresholding for the A-beam. Yellow indicates a filled element and blue void. Since ramping is not used some regions contain intermediate density values.



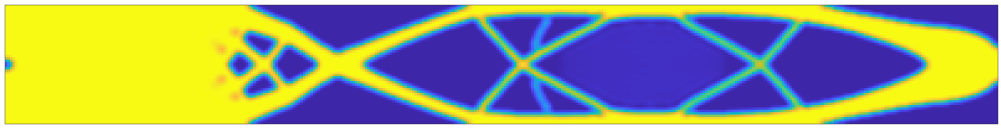
(a) Final design using full model.



(b) Final design using reduced order model with CAE.

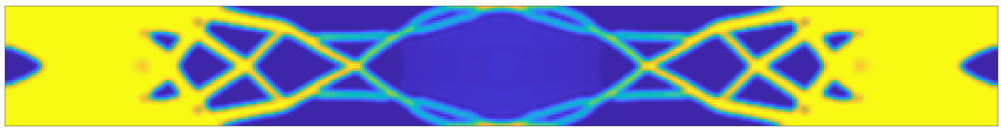


(c) Final design using reduced order model with CAEEON.

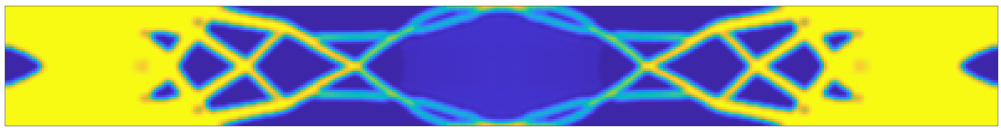


(d) Final design using reduced order model with CAEEONmod.

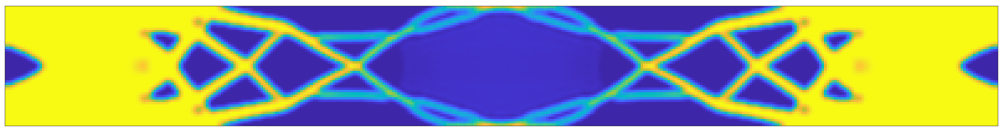
Figure 6.7: The figures show the final design for the full model and the different reduced order methods without Heaviside thresholding for the B-beam. Yellow indicates a filled element and blue void. Since ramping is not used some regions contain intermediate density values.



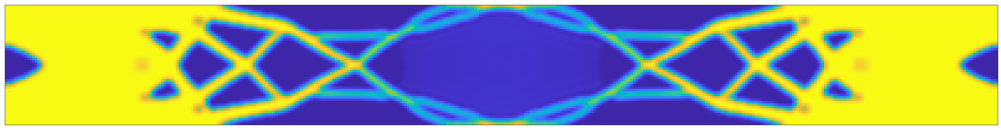
(a) Final design using full model.



(b) Final design using reduced order model with CAE.



(c) Final design using reduced order model with CAEEON.



(d) Final design using reduced order model with CAEEONmod.

Figure 6.8: The figures show the final design for the full model and the different reduced order methods without Heaviside thresholding for the C-beam. Yellow indicates a filled element and blue void. Since ramping is not used some regions contain intermediate density values.

Chapter 7

Discussion

7.1 Consistent sensitivity analysis

As seen in figure 6.1a the analytical sensitivities for all the reduced order methods seem to agree with the full models sensitivities, for two basis vectors and four eigenfrequencies. This is expected for CAE and the modified CAEEON, and some error is expected for CAEEON due to the term ignored in the sensitivity analysis. We expect the error in eigenvalues sensitivities to grow with the number of basis vectors and eigenvectors.

Moving on to the case when eight eigenfrequencies are considered, see figure 6.1b. For CAEEON, the eigenfrequencies error deviates from the other methods, and it seems to grow with the eigenvalue number. This is expected, which is again due to the term in the sensitivity analysis which is ignored. Although, this does not appear to have any effect on the objective functions sensitivity. This can be explained by the fact that the p-norm is used, meaning that the effect from the higher order eigenfrequencies is many times smaller than that of lower order eigenfrequencies.

The same cannot be said for CAE, whose errors in the higher order eigenfrequencies are many orders of magnitude larger than the errors from the other methods. This can also be seen in the objective functions sensitivities, which deviates from the other methods. The reason CAE struggles more could be explained by the basis vectors relaxing to lower order eigenmodes since they are not deflated. In other words, it seems that deflation is necessary to achieve accurate sensitivity analysis.

When the number of basis vectors is doubled from two to four, see figures 6.2a-6.2b CAE struggles even more while CAEEON does not seem to be affected as much. CAE struggling more is expected, again due to the fact that deflation is not used and now more basis vectors are used, resulting in larger errors. This may be the reason CAEEON is not affected by the increase in basis vectors. It should also be pointed out that the modified CAEEON is still looking very similar to the full model.

When the number of basis vectors is doubled again, the errors for CAE increase further. Meanwhile, the modified CAEEON seems to follow the full model, and while there is a slight error in the higher eigenvalues for CAEEON the objective function's sensitivity is unchanged.

Seeing that the resulting designs of the methods are very similar, see table 6.3 and figures 6.6-6.8, the question is if the errors in the sensitivity analysis matter.

7.2 Performance

The reduced order models performance is measured in the number of factorizations, since it requires on the order of n^3 operations, where the reduced order models need about sn^2 operations. Thus, for moderately large $n >> s$ this should be a good measure. The wall-time is *not* used as a metric to compare the methods, since it greatly depends on the implementation. The reduced order models are implemented in a high-level language and do not take advantage of effective memory usage or

parallellization. This also limits studying the methods for a small number of unknowns, here a grid of 400×50 elements is used which amounts to about 41,000 degrees of freedom, whereas the method is intended for up to 1,000,000-10,000,000. Beyond that point memory requirements and computational effort becomes too large, so iterative methods are used instead.

Evaluating the performance of a numerical method can be difficult, particularly if the goal is to make general claims about how it performs versus other methods. One reason is that there are many different parameters which, if tuned correctly can bring out the full potential in a method, and if not can bring out all its flaws.

Let us begin with considering the number of basis vectors, see figures 6.4a - 6.4c which show the proportions of design cycles where a factorization was performed. The gain of going from two basis vectors to four is substantial for all the cases, around half the number of factorizations is needed. The gain of going from four to six is however very slight, in fact studying table 6.2 shows only a few percentage points is saved at most. Thus it seems that using a moderate amount of basis vectors, say about four, is best. For the B-beam the difference between using two and four basis vectors is the most significant, while for the C-beam the difference is only around 25%. Although using circa four basis vectors seems like the best choice, it is clear that the results depend on what problem is considered.

Furthermore, from table 6.3, we see that the objective's value is very similar among the methods. In fact, the difference between the methods is so small that the table shows the difference of the objective function's value between the full model and the reduced model. At most a few tenth's of a Hz is the difference, plus or minus, which is completely insignificant. In particular for topology optimization which is used as a first step in most design processes. Figures 6.6 - 6.8 show that even the designs are indistinguishable. Perhaps a less conservative value of α_{tol} and δ_{tol} could be used, since the results are so close to the full model.

Moving on to comparing the methods, see table 6.2, we see there is not a huge difference among the methods with regard to the amount of factorizations computed, at best differing by a percentage point. It is also seen that although the proportion of design cycles where a factorization is performed changes with the iteration number, it is a steady decrease. This is expected since the changes in design decreases with the iteration number, see figures 6.5a - 6.5c, meaning that the reduced order model, which is based on a previous design, should be accurate when the design changes are small.

The question is, given a certain stopping critera if one of the three reduced order methods would converge quicker than the other. This is very difficult to investigate since there can be many stopping criteria and the best method could depend on what measure is chosen. The modulus of the gradient, the change in objective function value, or the norm of the design changes between consecutive iterations can be used as convergence measures. Take for example the sine of the angle between consecutive designs, see figures 6.5a - 6.5c. Apart from the C-beam, it is hard to choose a value at which the design changes reach a minimum, that is where the optimization would converge. Moreover we see that the sine of the angle has a large variance, meaning the optimization could reach the threshold by pure chance.

This brings us to the final question, which method should be chosen? From the sensitivity analysis we see that CAEEONmod is more accurate, although they result in similar designs and save about the same amount of work in terms of factorizations. Perhaps there are cases where the error in sensitivity analysis for CAE and CAEEON are large enough to impact the end result, and if so CAEEONmod is the more reliable choice. However no such cases could be found. There could also be cases where CAEEON outperforms CAEEONmod.

Lastly, it should be noted the parameters δ_{tol} and α_{tol} are here kept constant, but they will also influence the performance of the methods. If a lower α_{tol} is used the number of factorizations is expected to increase, while a higher value might lead to less accurate results from the reduced order model, but save a few factorizations. However, there is nothing that says one method could benefit

more from tuning these parameters than the other. Furthermore, the results from varying the amount of basis vectors indicate that the optimal value depends on which problem is considered. Hence no truly 'best' values may exist, and they should be determined heuristically from trial and error.

From table 6.2 we see that we can save about 85-75% of the work after 1000 iterations. This will again depend on the choice of α_{tol}, δ_{tol}, the number of basis vectors s, the size of the problem n, and the implementation. The amount of work saved will also depend on the convergance criteria, since the majority of factorizations occur in the beginning of the optimization process.

Chapter 8

Conclusions

From sensitivity analysis we saw that, for a small number of basis vectors and eigenfrequencies, the sensitivity analysis for all reduced order methods were consistent with the full models sensitivities. However, when the number of eigenfrequencies and basis vectors was increased, only the novel method which used the old eigenmodes to deflate the basis vectors was consistent. With that said, this did not seem to have a significant impact on the methods performance, the resulting designs and fundamental frequencies were identical regardless of method. No definitive answer could be given as to which method is the best, although the novel method was shown to be more consistent. Using a moderate amount of basis vectors was seen to have the most success for all reduced order methods, reducing the work by around 80% for four basis vectors.

Bibliography

[1] U. Kirsch, *Reanalysis of Structures*. Springer Science, 2008.

[2] O. Amir, M. P. Bendsøe, and O. Sigmund, "Approximate reanalysis in topology optmization," *International Journal for numerical methods in engineering*, vol. 78, no. 12, pp. 1474–1491, 2009. DOI: 0.1002/nme.2536.

[3] M. P. Bendsøe and N. Kikuchi, "Generating optimal topologies in structural design using a homogenization method," *Computer Methods in Applied Mechanics and Engineering*, vol. 71, no. 2, pp. 197–224, 1988. DOI: 10.1016/0045-7825(88)90086-2.

[4] E. Andreassen, B. S. Lazarov, and O. Sigmund, "Design of manufacturable 3d extremal elastic microstructure," *Mechanics of Materials*, vol. 69, no. 1, pp. 1–10, 2014. DOI: 10.1016/j.mechmat.2013.09.018.

[5] A. Dalklint, M. Wallin, and D. Tortorelli, "Eigenfrequency constrained topology optimization of finite strain hyperelastic structures," *Structural and Multidisciplinary Optimization*, vol. 61, no. 6, pp. 2577–2594, 2020. DOI: 10.1007/s00158-020-02557-9.

[6] L. Yin and G. K. Ananthasuresh, "Design of distributed compliant mechanisms," *Mechanics Based Design of Structures and Machines*, vol. 31, no. 2, pp. 151–179, 2003. DOI: 10.1081/SME-120020289.

[7] A. R. Diaz and N. Kikuchi, "Solutions to shape and topology eigenvalue optimization problems using a homogenization method," *International journal for numerical methods in engineering*, vol. 35, no. 7, pp. 1487–1502, 1992. DOI: 10.1002/nme.1620350707.

[8] A. Dalklint, M. Wallin, K. Bertoldi, and D. Tortorelli, "Tunable phononic bandgap material designed via topology optimization," *Journal of the Mechanics and Physics of Solids*, vol. 163, p. 104 849, 2022. DOI: 10.1016/j.jmps.2022.104849.

[9] O. Amir, N. Aage, and B. S. Lazarov, "On multigrid-cg for efficient topology optimization," *Structual Multidisciplinary Optimization*, vol. 49, no. 5, pp. 815–829, 2014. DOI: 10.1007/s00158-013-1015-5.

[10] N. S. Ottosen and H. Petersson, *Introduction to the Finite Element Method*. Prentice Hall Europe, 1992.

[11] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, 1997.

[12] K. Svanberg, "A class of globally convergent optimization methods based on conservative convex separable approximations," *Society for Industrial and Applied Mathematics*, vol. 12, no. 2, pp. 555–573, 2002. DOI: 10.1137/S1052623499362822.

[13] M. P. Bendsøe, "Optimal shape design as a material distribution problem," *Structural Optimization*, vol. 1, no. 4, pp. 193–202, 1989. DOI: 10.1007/BF01650949.

[14] M. P. Bendsøe and O. Sigmund, "Material interpolation schemes in topology optimization," *Archive of Applied Mechanics*, vol. 69, no. 9, pp. 635–654, 1999. DOI: 10.1007/s004190050248.

[15] T. E. Bruns and D. A. Tortorelli, "Topology optimization of non-linear elastic structures and compliant mechanisms," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 26, pp. 3443–3459, 2001. DOI: 10.1016/S0045-7825(00)00278-4.

[16] O. Sigmund, "Morphology-based black and white filters for topology optimization," *Structural Multidisciplinary Optimization*, vol. 33, no. 4, pp. 401–424, 2007. DOI: 10.1007/s00158-006-0087-x.

[17] B. S. Lazarov and O. Sigmund, "Filters in topology optimization based on helmholtz-type differential equations," *International Journal for Numerical Methods in Engineering*, vol. 86, no. 6, pp. 765–781, 2011. DOI: 10.1002/nme.3072.

[18] M. Wallin, N. Ivarsson, and O. Amir, "Consistent boundary conditions for pde filter regularization in topology optimization," *Structural and Multidisciplinary Optimization*, vol. 62, no. 3, pp. 1299–1311, 2020. DOI: 10.1007/s00158-020-02556-w.

[19] J. K. Guest, J. H. Prévost, and T. Belytschko, "Achieving minimum length scale in topology optimization using nodal desgin variables and projection functions," *International journal for numerical methods in engineering*, vol. 61, no. 2, pp. 238–254, 2004. DOI: 10.1002/nme.1064.

[20] F. Wang, B. S. Lazarov, O. Sigmund, and J. S. Jensen, "On projection methods, convergence and robust formulations in topology optimization," *Structural Multidisciplinary Optimization*, vol. 43, no. 6, pp. 767–784, 2011. DOI: 10.1007/s00158-010-0602-y.

[21] L.-C. Böiers, *Mathematical Methods of Optimization*. Studentlitteratur, 2010.

[22] P. W. Christensen and A. Klarbring, *An introduction to structural optimization*. Springer Science, 2009.

[23] C. Fleury and V. Braibant, "Structural optimization: A new dual method using mixed variables," *International journal for numerical methods in engineering*, vol. 23, no. 3, pp. 409–428, 1986. DOI: 10.1002/nme.1620230307.

[24] K. Svanberg, "The method of moving asympotets - a new method for structural optimization," *International journal for numerical methods in engineering*, vol. 24, no. 2, pp. 359–373, 1987. DOI: 10.1002/nme.1620240207.

[25] U. Kirsch, M. Kocvara, and J. Zowe, "Accurate reanalysis of structures by a preconditioned conjugate gradient method," *International journal for numerical methods in engineering*, vol. 55, no. 2, pp. 233–251, 2002. DOI: https://doi.org/10.1002/nme.496.

[26] M. Renardy and R. C. Rogers, *An Introduction to partial differential equations*. Springer-Verlag, 2004.

[27] M. Bogomolny, "Topology optimization for free vibrations using combined approximations," *International Journal for Numerical Methods in Engineering*, vol. 82, no. 5, pp. 617–636, 2010. DOI: 10.1002/nme.2778.

[28] A. J. Torii and J. R. Faria, "Structural optimization considering smallest magnitude eigenvalues: A smooth approximation," *The Braizilian Society of Mechanical Sciences and Engineering*, vol. 39, no. 5, pp. 1745–1754, 2017. DOI: 10.1007/s40430-016-0583-x.

[29] J. Du and N. Olhoff, "Topological design of freely vibrating continuum structures for maximum values of simple and multiple eigenfrequencies and frequency gaps," *Structural Multidisciplinary Optimization*, vol. 34, no. 2, pp. 91–110, 2007. DOI: 10.1007/s00158-007-0101-y.

Appendices

Appendix A

Pseudocode

Algorithm 1: CA

Input: $\Delta \boldsymbol{K}, \boldsymbol{K}_0^{-1}, \boldsymbol{M}, \boldsymbol{\psi}_0$
Output: T
/* Compute first basis vector */
$\boldsymbol{B} \leftarrow \boldsymbol{K}_0^{-1} \Delta \boldsymbol{K}$;
$\boldsymbol{u}_1 \leftarrow \boldsymbol{K}_0^{-1} \boldsymbol{M} \boldsymbol{\psi}_0$;
$\boldsymbol{t}_1 \leftarrow \boldsymbol{u}_1 \left(\boldsymbol{u}_1^T \boldsymbol{M} \boldsymbol{u}_1 \right)^{-1/2}$;
/* Compute remaining basis vectors */
for $i = 2 \rightarrow s$ **do**
  $\boldsymbol{u}_i \leftarrow -\boldsymbol{K}_0^{-1} \Delta \boldsymbol{K} \boldsymbol{t}_{i-1}$;
  $\boldsymbol{t}_i \leftarrow \boldsymbol{u}_i \left(\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i \right)^{-1/2}$;
end

Algorithm 2: CA with Eigenmode Orthogonalization (CAEEON)

Input: $\Delta \boldsymbol{K}, \boldsymbol{K}_0^{-1}, \boldsymbol{M}, \boldsymbol{\psi}_{0,k}, \{\boldsymbol{\psi}_i\}_{i=1}^{k-1}$
Output: V
/* Compute first basis vectors */
$\boldsymbol{u}_1 = \boldsymbol{K}_0^{-1} \boldsymbol{M} \boldsymbol{\psi}_{0,k}$;
$\boldsymbol{t}_1 = \boldsymbol{u}_1 (\boldsymbol{u}_1^T \boldsymbol{M} \boldsymbol{u}_1)^{-1/2}$;
$\boldsymbol{v}_1 \leftarrow \boldsymbol{t}_1$;
for $j = 1 \rightarrow k-1$ **do**
  $\boldsymbol{v}_1 -= (\boldsymbol{t}_1^T \boldsymbol{M} \boldsymbol{\psi}_j) \boldsymbol{\psi}_j$;
end
/* Compute remaining basis vectors */
for $i = 2 \rightarrow s$ **do**
  $\boldsymbol{u}_i = -\boldsymbol{K}_0^{-1} \Delta \boldsymbol{K} \boldsymbol{t}_{i-1}$;
  $\boldsymbol{t}_i = \boldsymbol{u}_i (\boldsymbol{u}_i^T \boldsymbol{M} \boldsymbol{u}_i)^{-1/2}$;
  $\boldsymbol{v}_i \leftarrow \boldsymbol{t}_i$;
  for $j = 1 \rightarrow k-1$ **do**
    $\boldsymbol{v}_i -= \left(\boldsymbol{t}_i^T \boldsymbol{M} \boldsymbol{\psi}_j \right) \boldsymbol{\psi}_j$
  end
end

Algorithm 3: Modified CAEEON (CAEEONmod)

Input: $\Delta \boldsymbol{K}, \boldsymbol{K}_0^{-1}, \boldsymbol{M}, \boldsymbol{\psi}_{0,k}, \{\hat{\boldsymbol{\psi}}_{0,i}\}_{i=1}^{k-1}$
Output: $\mathbf{V}$
`/* Compute first basis vector */`
$\boldsymbol{u}_1 = \boldsymbol{K}_0^{-1}\boldsymbol{M}\boldsymbol{\psi}_{0,k}$;
$\boldsymbol{t}_1 = \boldsymbol{u}_1(\boldsymbol{u}_1^T\boldsymbol{M}\boldsymbol{u}_1)^{-1/2}$;
$\boldsymbol{v}_1 \leftarrow \boldsymbol{t}_1$;
for $j = 1 \rightarrow k-1$ **do**
  $\boldsymbol{v}_1 -= (\boldsymbol{t}_1^T\boldsymbol{M}\hat{\boldsymbol{\psi}}_j)\hat{\boldsymbol{\psi}}_j$;
end
`/* Compute remaining basis vectors */`
for $i = 2 \rightarrow s$ **do**
  $\boldsymbol{u}_i = -\boldsymbol{K}_0^{-1}\Delta\boldsymbol{K}\boldsymbol{t}_{i-1}$;
  $\boldsymbol{t}_i = \boldsymbol{u}_i(\boldsymbol{u}_i^T\boldsymbol{M}\boldsymbol{u}_i)^{-1/2}$;
  $\boldsymbol{v}_i \leftarrow \boldsymbol{t}_i$;
  for $j = 1 \rightarrow k-1$ **do**
    $\boldsymbol{v}_i -= \left(\boldsymbol{t}_i^T\boldsymbol{M}\hat{\boldsymbol{\psi}}_j\right)\hat{\boldsymbol{\psi}}_j$
  end
end