COMPUTATIONAL ADVANCES IN CONTINUUM TOPOLOGY OPTIMIZATION ALGORITHMS

A THESIS
Submitted by

MOHAMED TAREK MOHAMED

In fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY
(COMPUTER SCIENCE)

SEPTEMBER 2021

SCHOOL OF ENGINEERING AND INFORMATION TECHNOLOGY
UNIVERSITY OF NEW SOUTH WALES AT CANBERRA

UNIVERSITY OF NEW SOUTH WALES

ABSTRACT

Topology optimization is a fascinating area of research with numerous unsolved computational challenges. In this thesis, the author aims to advance the research on improving the computational efficiency of common topology algorithms for practical real life problems. Beside the research contributions in this thesis, the introduction (chapter 1) is written to cover much of the theory behind the algorithms and formulations used in topology optimization including some details that often get ignored in most papers and texts in the field of topology optimization. A lot of the details presented in the introduction is scattered in multiple resources between computational mechanics books, optimization theory books and papers, and topology optimization literature. This makes it difficult for people starting to learn topology optimization to easily cover the theory needed to do advanced research in the field. An attempt is made to give a reasonably comprehensive coverage of the theory of the finite element method with an emphasis on linear elasticity as well as the theory behind common nonlinear programming algorithms used in topology optimization. Additionally, a presentation of all the common paradigms for decision-making under uncertainty is presented. Topology optimization under uncertainty is a field of research with many unsolved computational problems. This presentation will hopefully help more researchers get started in this field of research more easily.

In chapter 2, the first research contribution of this thesis is presented. In particular, a flexible and theoretically sound way to adapt penalties in the continuation

solid isotropic material with penalization (CSIMP) method is proposed which gives significant speedups in the experiments run. Four common test problems from literature, three 2D and one 3D, are used to test the efficacy of the penalty adaptation with different parameter settings. The main factors affecting the efficacy of the penalty adaptation in the CSIMP algorithm in reducing the number of finite element analysis (FEA) simulations needed to converge to the final solution are identified. The experimental results demonstrate a significant reduction in the number of FEA simulations required to reach the optimal solution in the decreasing tolerance CSIMP algorithm, with exponentially decaying tolerance, with little to no detriment in the objective value and the other metrics used. Finally, a mathematical and experimental treatment of the effect of the minimum pseudo-density parameter on the convergence of the CSIMP algorithm is given with some recommendations for choosing a suitable value. These results appear in the Computer Methods in Applied Mechanics and Engineering journal (Tarek and Ray, 2020).

In chapter 3, the problem of handling load uncertainty efficiently in compliance-based topology optimization problems is tackled. A comprehensive review of all the literature on handling uncertainty in compliance-based problems is presented. And a number of exact methods are proposed to handle load uncertainty in compliance-based topology optimization problems where the uncertainty is described in the form of a set of finitely many loading scenarios. This includes mean compliance minimization or constraining the mean compliance, minimizing or constraining a weighted sum of the mean and standard deviation of the load compliances as well as minimizing or constraining the maximum load compliance for all the loading

scenarios. By detecting and exploiting low rank structures in the loading scenarios, significant performance improvements are achieved using some novel methods. The computational complexities of the algorithms proposed are demonstrated and experiments are run to verify the efficacy of the proposed algorithms at reducing the computational cost of these classes of topology optimization problems. The methods presented here are fundamentally data-driven in the sense that no probability distributions or continuous domains are assumed for the loading scenarios. This sets this work apart from most of the literature in the domain of stochastic and robust topology optimization where a distribution or domain is assumed. Additionally, the methods proposed here are shown to be particularly suitable with the augmented Lagrangian algorithm when dealing with maximum compliance constraints. This work appears in the Structural and Multidisciplinary Optimization journal.

In chapter 4, approximate methods for handling many loading scenarios with a high rank loading matrix are developed. In particular, approximation schemes for the mean compliance and a class of scalar-valued functions of the load compliances are developed. The approximation schemes are based on a reformulation of the function approximated as a trace or diagonal estimation problem, opening the door to using many of the available methods for trace or diagonal estimation. The approximation methods are tested on a number of standard 2D and 3D benchmark problems using low and high rank loading scenarios to solve mean compliance minimization as well as minimizing the weighted sum of the mean compliance and its standard deviation. Significant speedups are achieved compared to the exact methods when the rank of the load matrix is high. This work is submitted to the Structural and

Multidisciplinary Optimization journal as of the time of the writing of this thesis.

In chapter 5, a summary of all the findings in this thesis and some potential future work for the author here or for aspiring researchers in topology optimization is presented.

DECLARATION

Signature:

Name:

Date:

Student ID Number

ACKNOWLEDGMENTS

Acknowledgment here

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

Term	Abbreviation
Finite element analysis	FEA
Degree of freedom	dof
Boundary value problem	BVP
Right hand side	RHS
Left hand side	LHS
Level set method	LSM
Solid isotropic material with penalization	SIMP
Evolutionary structural optimization	ESO
Bi-directional evolutionary structural optimization	BESO
Genetic evolutionary structural optimization	GESO
Volume constrained compliance minimization	VCCM
Simultaneous analysis and design	SAND
Nested analysis and design	NAND
Nonlinear program (or programming)	NLP
Integer nonlinear program	INLP
Mixed integer nonlinear program	MINLP
Rational approximation of material properties	RAMP
Continuation solid isotropic material with penalization	CSIMP
Karush-Kuhn-Tucker	KKT
Evolutionary algorithm	EA
Genetic algorithm	GA
Method of moving asymptotes	MMA
Convex linearization method	CONLIN
Interior point optimizer	IPOPT
Limited memory Broyden, Fletcher, Goldfarb, and Shann	l-BFGS
Augmented Lagrangian	AugLag
Linear constraint qualification	LCQ
Linear independence constraint qualification	LICQ
Mangasarian-Fromovitz constraint qualification	MFCQ
Second order correction	SOC
Robust optimization	RO
Stochastic optimization	SO
Risk-averse optimization	RAO
Reliability-based design optimization	RBDO
Value-at-risk	VaR
Conditional value-at-risk	CVaR
First order reliability method	FORM
First order second moment	FOSM

Table 1 continued from previous page

Term	Abbreviation
Most probable point	MPP
Reliability index approach	RIA
Performance measure approach	PMA
Cumulative distribution function	CDF
Probability density function	PDF
Second order reliability method	SORM
Non-probabilistic reliability-based design optimization	NRBDO
Random access memory	RAM
Singular value decomposition	SVD
Messerschmitt-Bolkow-Blohm	MBB
Conjugate gradient	CG
Preconditioned conjugate gradient	PCG
Automatic continuation solid isotropic material with penalization	Auto-CSIMP
Continuous objective	C-Obj
Discrete objective	D-Obj
Decreasing tolerance continuation solid isotropic material with penalization	Dec-Tol CSIMP
Karhunen-Loeve	K-L
Central processing unit	CPU
Graphics processing unit	GPU
Polynomial chaos expansion	PCE
Reliability-based topology optimization	RBTO
Non-probabilistic reliability-based topology optimization	NRBTO
Standard deviation	std

1. INTRODUCTION

1.1 Motivation

In mechanical design, the performance of a component largely depends on its shape and the material it is made of. Each component is expected to handle specific loading and position requirements at the "interfaces" of the component with the external environment. For example, a table is expected to handle a specific load over its top surface area, while having a fixed base, and being contained in a specific bounding box. The requirements therefore do not dictate any specific shape to the legs or their number. The shape is actually a decision the designer has to make.

Designing against failure is the primary goal of mechanical design. Mechanical failure of a structure can occur as a result of a number of factors, e.g. yielding, potentially large deformations due to system bifurcation/instability, fracture, fatigue, creep, etc. It is therefore the job of a designer to identify the best designs that prevent against failure while maximizing or minimizing one or more performance or economic metrics. Automating this task using mathematical models and optimisation algorithms is the purpose of the field of topology and shape optimisation.

There are a number of computational challenges that one faces when trying to computationally optimize the shape of a large or high resolution structure with many degrees of freedom. For example, evaluating the performance metrics may require an expensive physical simulation in the form of a finite element analysis (FEA). The optimization algorithm used should be able to handle a large number of decision variables and constraints. The design produced must be robust to changes

or uncertainty in the load, boundary conditions, shape, material properties or other problem data.

1.2 Aims

In this thesis, a number of techniques to accelerate topology optimization algorithms in different contexts are presented in hope to make computational topology optimization more scalable, practical and therefore feasible.

1.3 Overview of topology optimization

1.3.1 Continuum vs discrete topology optimization

Topology optimization problems can fall into 2 broad categories:

1. Continuum topology optimization

2. Discrete topology optimization

Continuum topology optimization is topology optimization of continuum structures, where continuum structures are structures modelled as one or more volumes of continuum materials, each of which can be meshed to any arbitrary level of fineness. Discrete topology optimization is topology optimization of discrete structures where discrete structures are structures modelled as a finite set of components connected at specific interface points, e.g. truss systems. In this thesis, the focus is on continuum topology optimization however the techniques proposed can be generalized to truss or discrete topology optimization as well.

1.3.2 Sizing, shape and topology optimization

In design optimization literature, there is a distinction between the following 3 terms:

1. Sizing optimization

2. Shape optimization

3. Topology optimization

Topology optimization, also known as generalized shape optimization, is used to refer the case when the algorithm is allowed to choose where to place holes in the shape. Shape optimization on the other hand is used to refer to the case when the algorithm can change the surface of the shape without creating new surfaces or voids. Sizing optimisation is used to refer to the case where a base design is fixed and parameterized and only a few "size" parameters are allowed to change, e.g. the radius of a cylindrical part of the design.

In order to evaluate objectives and constraints of topology optimization problems, solving a finite element analysis (FEA) is necessary. In the next section, the theory of linear elasticity is introduced before presenting particular topology optimization formulations.

1.4 Quasi-static linear elasticity

Topology optimization problems are a sub-class of physics-constrained optimization problems. In this work, the main physics of concern is structural mechanics with the quasi-static, linear elasticity assumption. This section will detail the theory behind quasi-static linear elasticity and how it relates to topology optimization.

1.4.1 Strong form

Let the base design from which material may be removed be a compact volume. Let the open domain of material strictly inside this volume be Ω and its boundary be $\partial\Omega$. Let each point in Ω be positioned at a location $\boldsymbol{x}$ and surrounded by an infinitesimal cube with 2 faces normal to each axis. The following governing differential equation, known as Lamé equation, holds at any point in Ω, which can be derived from Newton's second law applied on any arbitrary volume of material.

$$\sigma_{ij,j}(\boldsymbol{x}) = -\rho f_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega \tag{1.1}$$

where $\boldsymbol{f}$ is the first order tensor resembling the body force per unit mass (typically due to gravity) whose component along the i^{th} axis is f_i. ρ is the point mass density of the material, and $\boldsymbol{\sigma}$ is the symmetric second order stress tensor, where σ_{ij} is the component of the traction along the j^{th} axis acting on the face of the deformed infinitesimall cube whose outward normal was the i^{th} axis before deformation. This equation holds for any point in the open domain Ω but not at the boundary $\partial\Omega$

over which boundary conditions apply. $\sigma_{ij,j}$ is the divergence of the stress tensor along the second index, where ", j" implies the derivative with respect to x_j, and the repeating index j implies the Einstein summation over the span of the index j. So $\sigma_{ij,j}$ is nothing but $\sum_j \frac{\partial \sigma_{ij}}{\partial x_j}$. Throughout the following derivations, tensor notation will be used, where a repeating index in the same term implies summation over that index, unless otherwise stated or an explicit summation is used, and free indices in an equation imply the enumeration of all combinations of the indices' values.

Let $\boldsymbol{u}(\boldsymbol{x})$ be the deformed position of a point originally positioned at $\boldsymbol{x}$. The following is the definition of the symmetric second order strain tensor $\boldsymbol{\epsilon}$, assuming infinitesimal strain. This does not account for the effect of buckling.

$$\epsilon_{ij} = \frac{1}{2}(u_{i,j} + u_{j,i}) \tag{1.2}$$

The constitutive relation between the stress and strain tensors is given as follows.

$$\sigma_{ij} = C_{ijkm}\epsilon_{km} \tag{1.3}$$

where $\boldsymbol{C}$ is the fourth order stiffness tensor of the material and $C_{ijkm} = C_{jikm} = C_{ijmk} = C_{kmij}$. For an isotropic solid, the above equation can be simplified using Hooke's law which defines the relationship between the stress and strain tensors as follows.

$$\sigma_{ij} = \frac{E\nu}{1-\nu^2}\delta_{ij}\epsilon_{kk} + \frac{E\nu}{1+\nu}\epsilon_{ij} \tag{1.4}$$

where E is the Young's modulus, ν is Poisson's ratio and δ_{ij} is the Kronecker delta defined to be 1 when $i = j$ and 0 otherwise.

Let $\partial\Omega_{Di}$ be the portion of the boundary, whose u_i is known to be g_i and therefore enforced by the following so-called Dirichlet boundary condition, e.g. a point with a pin joint for which all 3 degrees of freedom are fixed.

$$u_i(\boldsymbol{x}) = g_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_{Di} \tag{1.5}$$

Let $\partial\Omega_{Ni}$ be the portion of the boundary over which a force, t_i, per unit area along the i^{th} axis is known and therefore enforced by the following so-called Neumann boundary condition.

$$\sigma_{ij}(\boldsymbol{x})n_j(\boldsymbol{x}) = t_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \partial\Omega_{Ni} \tag{1.6}$$

where $\boldsymbol{n}$ is the outward normal at the boundary point. Wherever u_i is not restricted on the boundary by a Dirichlet boundary condition, $t_i = 0$ is used in a Neumann boundary condition. This is often called the natural boundary condition. Note that $\overline{\Omega}$, the closure of Ω, is $\Omega \cup \partial\Omega$, and that $\partial\Omega = \partial\Omega_{Di} \cup \partial\Omega_{Ni}$ for each index i, where $\partial\Omega_{Di} \cap \partial\Omega_{Ni} = \phi$, that is $\partial\Omega_{Di}$ and $\partial\Omega_{Ni}$ are complements of each other in the superset $\partial\Omega$. However, $\partial\Omega_{Di}$ and $\partial\Omega_{Dj}$ are not necessarily disjoint for $i \neq j$, as well as $\partial\Omega_{Ni}$ and $\partial\Omega_{Nj}$.

Let U_i be the set of all functions $\{u_i(\boldsymbol{x}) : u_i(\boldsymbol{x}) = g_i, \forall \boldsymbol{x} \in \partial\Omega_{Di}\}$. In an analysis problem, the shape, material and boundary conditions are known, and the

functions $u_i(\boldsymbol{x}) \in U_i$ over the domain Ω and the surfaces $\partial\Omega_{Ni}$ are to be identified. Post-processing can then be done to identify $\boldsymbol{\sigma}$ over Ω and the reaction forces at the Dirichlet boundaries. Equations 1.1, 1.5 and 1.6 represent what is usually referred to as the strong form of the boundary value problem (BVP). It is called strong form because the function $\boldsymbol{u}(\boldsymbol{x})$ to be found needs to be differentiable twice. This puts a constraint on any approximation scheme that tries to work directly with the strong form. Notice that the BVP is actually made of 2 partial differential equations for a 2D problem and 3 for a 3D problem.

1.4.2 Weak form

Let V_i be the set of once-differentiable functions $\{v_i(\boldsymbol{x}) : v_i(\boldsymbol{x}) = 0, \forall \boldsymbol{x} \in \partial\Omega_{Di}\}$. $v_i(\boldsymbol{x})$ is typically called the weighting function or the variation function because of its interpretation in variational mechanics. Given the strong form, the following equation should hold for any arbitrary weighting function $v_i \in V_i \; \forall i$.

$$\int_{\Omega} v_i \sigma_{ij,j} \, d\boldsymbol{V} = -\int_{\Omega} v_i f_i \rho \, d\boldsymbol{V} \tag{1.7}$$

This is nothing but the weighted sum of all partial differential equations integrated over the domain Ω. Using product rule, we know that $(v_i\sigma_{ij})_{,j} = v_{i,j}\sigma_{ij} + v_i\sigma_{ij,j}$, so $v_i\sigma_{ij,j} = (v_i\sigma_{ij})_{,j} - v_{i,j}\sigma_{ij}$. The above integral can therefore be written as:

$$\int_{\Omega} v_{i,j}\sigma_{ij} \, d\boldsymbol{V} = \int_{\Omega} v_i f_i \rho \, d\boldsymbol{V} + \int_{\Omega} (v_i\sigma_{ij})_{,j} \, d\boldsymbol{V} \tag{1.8}$$

Using the divergence theorem, we know that $\int_{\Omega} (v_i \sigma_{ij})_{,j} \, d\boldsymbol{V} = \int_{\partial\Omega} v_i \sigma_{ij} n_j \, d\boldsymbol{S}$. Unrolling the Einstein summation and decomposing the domain boundary to Dirichlet and Neumann bounadries for each spatial dimension, we can re-write $\int_{\partial\Omega} v_i \sigma_{ij} n_j \, d\boldsymbol{S}$ as:

$$\sum_{i=1}^{nsd} \sum_{j=1}^{nsd} \left(\int_{\partial\Omega_{Di}} v_i \sigma_{ij} n_j \, d\boldsymbol{S} + \int_{\partial\Omega_{Ni}} v_i \sigma_{ij} n_j \, d\boldsymbol{S} \right) \tag{1.9}$$

where nsd is the number of spatial dimensions, 2 for 2D and 3 for 3D. By definition, $v_i = 0$ on $\partial\Omega_{Di}$. Furthermore, substituting in the Neumann boundary condition, $\sigma_{ij} n_j = t_i$, we get the final weak form of the BVP as follows.

$$\int_{\Omega} v_{i,j} \sigma_{ij} \, d\boldsymbol{V} = \int_{\Omega} v_i f_i \rho \, d\boldsymbol{V} + \int_{\partial\Omega_{Ni}} v_i t_i \, d\boldsymbol{S} \quad \forall v_i \in V_i, i \in [1, nsd] \tag{1.10}$$

This equation should hold for any arbitrary variation functions $v_i \in V_i$.

1.4.3 Finite dimensional weak form

The above derivation shows that the strong form implies the weak form. It is also possible to show that the weak form implies the strong form. The 2 forms are therefore both identical and exact. The key idea behind finite element analysis, is that instead of looking for the functions $u_i(\boldsymbol{x})$ in an infinite function space U_i, also known as a Hilbert function space, we limit the search to a finite set of basis functions, such that the approximate function $u_i^h(\boldsymbol{x})$ is assumed to be a linear combination of the finite number of basis functions used.

Let the domain Ω be divided into P nodes connected by E regularly shaped disjoint elements, e.g. quadrilaterals for 2D or hexahedrons for 3D, where $\overline{\Omega}$ is the closure of the open domain Ω and is given by: $\overline{\Omega} = \overline{\bigcup_e \Omega^e}$. The element domains are disjoint such that: $\Omega^e \cap \Omega^{e'} = \phi \, \forall e \neq e'$, where Ω^e is the interior of element e. Let there be P basis functions, one for each node such that $N^i(\boldsymbol{x})$ is the basis function associated with node i. The approximate function $u_i^h(\boldsymbol{x})$ can now be chosen from the finite dimensional function space:

$$U_i^h = \{u_i^h(\boldsymbol{x}) = \sum_{j=1}^{P} d_j^i N^j(\boldsymbol{x}) : \boldsymbol{d}^i \in \mathbb{R}^P, u_i^h(\boldsymbol{x}) = g_i(\boldsymbol{x}) \, \forall \boldsymbol{x} \in \partial\Omega_{Di}\} \tag{1.11}$$

Similarly, let the finite dimensional weighting function $v_i^h(\boldsymbol{x})$ be limited to the function space:

$$V_i^h = \{v_i^h(\boldsymbol{x}) = \sum_{j=1}^{P} c_j^i N^j(\boldsymbol{x}) : \boldsymbol{c}^i \in \mathbb{R}^P, v_i^h(\boldsymbol{x}) = 0 \, \forall \boldsymbol{x} \in \partial\Omega_{Di}\} \tag{1.12}$$

From 1.3 and 1.2:

$$\sigma_{ij}^h = \frac{1}{2} C_{ijkm}(u_{k,m}^h + u_{m,k}^h) = \frac{1}{2} C_{ijkm} u_{k,m}^h + \frac{1}{2} C_{ijkm} u_{m,k}^h = C_{ijkm} u_{k,m}^h \tag{1.13}$$

because $C_{ijkm} = C_{ijmk}$. The weak form can therefore be approximated by:

$$\int_{\Omega} v_{i,j}^h C_{ijkm} u_{k,m}^h \, d\boldsymbol{V} = \int_{\Omega} v_i^h f_i \rho \, d\boldsymbol{V} + \int_{\partial\Omega_{Ni}} v_i^h t_i \, d\boldsymbol{S} \quad \forall v_i^h \in V_i^h, i \in [1, nsd] \tag{1.14}$$

Further approximating the domain by the finite elements, we get:

$$\sum_e \int_{\Omega^e} v^h_{i,j} C_{ijkm} u^h_{k,m} \, d\boldsymbol{V} = \sum_e \int_{\Omega^e} v^h_i f_i \rho \, d\boldsymbol{V} + \sum_{e \in S} \int_{\partial\Omega^e_{Ni}} v^h_i t_i \, d\boldsymbol{S} \quad \forall v^h_i \in V^h_i, i \in [1, nsd] \tag{1.15}$$

where S is the set of elements on the boundary of the discretized domain and $\partial\Omega^e_{Ni}$ resembles the boundary faces of the surface element $e \in S$ over which a Neumann boundary condition applies. In order to identify $v^h_{i,j} = \sum_{k=1}^{P} c^i_k \frac{\partial N^k(\boldsymbol{x})}{\partial x_j}$ and $u^h_{i,j} = \sum_{k=1}^{P} d^i_k \frac{\partial N^k(\boldsymbol{x})}{\partial x_j}$ inside an element e, firstly $N^k(\boldsymbol{x})$ and $\frac{\partial N^k(\boldsymbol{x})}{\partial x_j}$ must be identified for $\boldsymbol{x} \in \Omega_e \; \forall e \in [1, E], k \in [1, P]$.

1.4.4 Basis functions

1.4.4.1 Line elements

Let the linear element e be made of $n = k + 1$ nodes, of local indices $[1, n]$. Let x be the global position coordinate along the 1D element. Moreover, let m be a 1D line master element going from -1 to 1, and let ξ be the local coordinate of any point in m. The following functions are known as the k^{th} order 1D Lagrange polynomial basis functions, defined one for each node of local index A in master element m.

$$M^A(\xi) = \prod_{B \neq A} \frac{\xi - \xi^B}{\xi^A - \xi^B} \tag{1.16}$$

where ξ^A is the value of ξ at the A^{th} node of master element m.

Notice that M^A is 1 at $\xi = \xi^A$ and 0 at $\xi = \xi^B \; \forall B \neq A$. Also, note that the sum

of all basis functions is equal to 1. A one-to-one invertible mapping can now be defined between each point $x \in \Omega_e$, i.e. located in element e, and a corresponding point ξ in the master element m located using the interpolation function:

$$x(\xi) = \sum_{A=1}^{n} x_e^A M^A(\xi) \tag{1.17}$$

where x_e^A is the position x of the node whose local index in element e is A.

In this whole thesis, it will be assumed that the same basis functions are used for the geometric mapping from ξ to x as well as representing the finite dimensional solution to the BVP as follows. This is known as isoparameteric mapping. Let $\boldsymbol{CC}^e$ be the cell connectivity vector of cell e, such that $CC_A^e = i$ where A is the local index and i is the global index of the node. We can now define $N^i(\xi)$ associated with some global node indexed i as follows:

$$N^i(x) = \begin{cases} M^A(\xi(x)) & \text{if} x \in \Omega_e \text{ for } A : CC_A^e = i \\ 0 & \text{otherwise} \end{cases} \tag{1.18}$$

where $\xi(x)$ is the inverse function of the mapping function $x(\xi)$. $N^i(\xi)$ will also be used to refer to $N^i(x(\xi))$, where it is clear from context when x and ξ refer to functions and when they refer to variables. From the above definition, each basis function N^i is only non-zero in the elements including node i, and 0 in all other elements.

1.4.4.2 Quadrilateral elements

Let m be a 2D square master element going from $[-1\ -1]^T$ to $[1\ 1]^T$, and let $\boldsymbol{\xi} = [\xi_1\ \xi_2]^T$ be the local position vector of a point in m. The Lagrange polynomial basis functions for a quadrilateral is formed by taking the tensor product of 2 1D Lagrange polynomials. Note that the number of nodes n required to define a set of k^{th} order Lagrange polynomial basis functions in a quadrilateral is $(k+1)^2$. The following is the definition of the k^{th} order Lagrange polynomial basis functions over the quadrilateral element e:

$$M^A(\xi_1, \xi_2) = M^A(\xi_1) M^A(\xi_2) \tag{1.19}$$

where $n = (k+1)^2$.

A one-to-one invertible mapping can then be defined between each point in element e located at $\boldsymbol{x_i}$ and the corresponding point in the master element m located at $\boldsymbol{\xi}$, using the interpolation function $\boldsymbol{x} = \sum_{A=1}^{n} \boldsymbol{x}_e^A M^A(\boldsymbol{\xi})$, where $\boldsymbol{x}_e^A$ is the position vector of the node whose local index in element e is A. Finally, the global shape functions $N^i(\boldsymbol{x})$ are defined as follows:

$$N^i(\boldsymbol{x}) = \begin{cases} M^A(\boldsymbol{\xi}(\boldsymbol{x})) & \text{if} \boldsymbol{x} \in \Omega_e \text{ for } A : CC_A^e = i \\ 0 & \text{otherwise} \end{cases} \tag{1.20}$$

$N^i(\boldsymbol{\xi})$ will also be used to refer to $N^i(\boldsymbol{x}(\boldsymbol{\xi}))$.

1.4.4.3 Hexahedral elements

Let m be a 3D cubic master element going from $[-1\ -1\ -1]^T$ to $[1\ 1\ 1]^T$, and let $\boldsymbol{\xi} = [\xi_1\, \xi_2\, \xi_3]^T$ be the local position vector of a point in m. The Lagrange polynomial basis functions for a hexahedral is formed by taking the tensor product of 3 1D Lagrange polynomials. The number of nodes n required to define a set of k^{th} order Lagrange polynomial basis functions in a hexahedral element is $(k+1)^3$. The following is the definition of the k^{th} order Lagrange polynomial basis functions over the hexahedral element e:

$$M^A(\xi_1, \xi_2, \xi_3) = M^A(\xi_1) M^A(\xi_2) M^A(\xi_3) \tag{1.21}$$

where $n = (k+1)^3$.

A one-to-one invertible mapping can then be defined between each point in element e located at $\boldsymbol{x}$ and the corresponding point in the master element m located at $\boldsymbol{\xi}$, using the interpolation function $\boldsymbol{x} = \sum_{A=1}^{n} \boldsymbol{x}_e^A M^A(\xi_1, \xi_2, \xi_3)$. Finally, the global shape functions $N^i(\boldsymbol{x})$ are defined as in the 2D case.

1.4.5 Variable substitution

Given the above basis functions:

$$N^i_{,j}(\boldsymbol{\xi}) = \frac{\partial N^i}{\partial x_j}(\boldsymbol{\xi}) = \begin{cases} M^A_{,j}(\boldsymbol{\xi}) & \text{if} \boldsymbol{x}(\boldsymbol{\xi}) \in \Omega_e \text{ for } A : CC^e_A = i \\ 0 & \text{otherwise} \end{cases} \tag{1.22}$$

where

$$M^A_{,j}(\boldsymbol{\xi}) = \sum_{k=1}^{nsd} \frac{\partial M^A}{\partial \xi_k}(\boldsymbol{\xi}) \frac{\partial \xi_k}{\partial x_j}(\boldsymbol{\xi}) \tag{1.23}$$

The matrix $\left[\frac{\partial \xi_k}{\partial x_j}\right]$ is the inverse of the Jacobian $\left[\frac{\partial x_k}{\partial \xi_j}\right]$, written as a matrix-valued function of $\boldsymbol{\xi}$.

Recall that the finite dimensional approximation of the field function $u^h_i(\boldsymbol{x})$ is limited to the following function space:

$$U^h_i = \{u^h_i(\boldsymbol{x}) = \sum_{j=1}^{P} d^i_j N^j(\boldsymbol{x}) : \boldsymbol{d}^i \in \mathbb{R}^P, u^h_i(\boldsymbol{x}) = g_i \, \forall \boldsymbol{x} \in \partial\Omega_{Di}\} \tag{1.24}$$

where P is the number of nodes in the domain. Similarly, the finite dimensional weighting/variation function $v^h_i(\boldsymbol{x})$ is limited to the function space:

$$V^h_i = \{v^h_i(\boldsymbol{x}) = \sum_{j=1}^{P} c^i_j N^j(\boldsymbol{x}) : \boldsymbol{c}^i \in \mathbb{R}^P, v^h_i(\boldsymbol{x}) = 0 \, \forall \boldsymbol{x} \in \partial\Omega_{Di}\} \tag{1.25}$$

Let $\partial\Omega^h_{Di}$ be the set of node indices in the discretized domain bounded by a Dirichlet boundary condition along the i^{th} axis. And let:

$$C^h_i = \{\boldsymbol{c^i} : c^i_j = 0 \, \forall j \in \partial\Omega^h_{Di}, \, c^i_j \in \mathbb{R} \, \text{otherwise}\} \tag{1.26}$$

V_i^h can therefore be more specifically defined as:

$$V_i^h = \{v_i^h(\boldsymbol{x}) = \sum_{j=1}^{P} c_j^i N^j(\boldsymbol{x}) : \boldsymbol{c}^i \in C_i^h\} \tag{1.27}$$

The finite dimensional weak form can now be written as:

$$\sum_e \int_{\Omega^e} c_p^i N_{,j}^p C_{ijkm} d_s^k N_{,m}^s \, d\boldsymbol{V} = \sum_e \int_{\Omega^e} c_p^i N^p f_i \rho \, d\boldsymbol{V} + \sum_{e \in S} \int_{\partial\Omega_{Ni}^e} c_p^i N^p t_i \, d\boldsymbol{S} \quad \forall \boldsymbol{c}^i \in C_i^h, i \in [1, nsd] \tag{1.28}$$

where the p and s indices span $[1, P]$. Notice that the only basis functions N^p which are non-zero in element e are the functions associated with any global node p such that $\exists A : CC_A^e = p$. Similarly, the only non-zero basis functions on a boundary surface element $f \in \partial\Omega_{Ni}^e$ are the basis functions associated with the nodes of that face/edge. So the weak form can also be written using local indices as follows:

$$\sum_e \int_{\Omega^e} \sum_{A=1}^{n_e} \sum_{B=1}^{n_e} c_{CC_A^e}^i M_{,j}^A C_{ijkm} d_{CC_B^e}^k M_{,m}^B \, d\boldsymbol{V} = \sum_e \int_{\Omega^e} \sum_{A=1}^{n_e} c_{CC_A^e}^i M^A f_i \rho \, d\boldsymbol{V} + \sum_{e \in S} \sum_{f \in \partial\Omega_{Ni}^e} \int_f \sum_{A \in f} c_{CC_A^e}^i M^A t_i \, d\boldsymbol{S} \quad \forall \boldsymbol{c}^i \in C_i^h, i \in [1, nsd] \tag{1.29}$$

where n_e is the number of nodes in element e, and $A \in f$ refers to all the local node indices in surface element f. Note that summation is implied over i, j, k and m on the left-hand-side (LHS), and over i on the right-hand-side (RHS). In the rest of the

derivation, global shape functions will be used for neatness sake.

1.4.5.1 Quadrilateral elements

The integrals over Ω^e are area integrals for 2D problems, and the integrals over $\partial\Omega^e_{Ni}$ are line integrals. The area and line integrals over Ω^e and $\partial\Omega^e_{Ni}$ respectively can be transformed to area integrals over the master element, and line integrals over the corresponding edges of the master element respectively. Let the Jacobian of the coordinates $\boldsymbol{x}$ in element e with respect to the master element's reference coordinates $\boldsymbol{\xi}$ be:

$$\boldsymbol{J}^e = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} \tag{1.30}$$

where $|\boldsymbol{J}^e|$ is its determinant. An infinitesimal area in element e is then:

$$d\boldsymbol{V} = |\boldsymbol{J}^e| d\boldsymbol{V}_{\boldsymbol{\xi}} = |\boldsymbol{J}^e| d\xi_1 d\xi_2 \tag{1.31}$$

Let the conditional Jacobian on one of the surface edges $f \in \partial\Omega^e_{Ni}$ be J^{ef}. Let $\bar{i}(f) \in [1, 2]$ be the dimension along which line f is fixed in the master element such that $\xi_{\bar{i}(f)} = -1 \; or \; 1$. For example, if the line f is horizontal in the master element, $\bar{i}(f) = 2$. $\xi_{\bar{i}(f)}$ will be -1 if f corresponds to the bottom edge of the master element and will be 1 if it corresponds to the top edge. Similarly, if the line f is vertical in the master element, $\bar{i}(f) = 1$. $\xi_{\bar{i}(f)}$ will be -1 if f corresponds to the left edge of the master element and will be 1 if it corresponds to the right edge. Let the non-fixed

master element coordinates on line f be $\bar{\boldsymbol{\xi}}^f$. The line/conditional Jacobian is then given by:

$$\boldsymbol{J}^{ef} = \frac{\partial \boldsymbol{x}}{\partial \bar{\boldsymbol{\xi}}^f} \tag{1.32}$$

In the quadrilateral case, the line Jacobian is a 2×1 matrix. If the non-fixed coordinate index on line f is $\bar{j}(f) \neq \bar{i}(f)$, an infinitesimal length on the edge f of element e can be written as:

$$d\boldsymbol{S} = \boldsymbol{J}^{ef} d\boldsymbol{S}_{\boldsymbol{\xi}} = \boldsymbol{J}^{ef} d\xi_{\bar{j}(f)} \tag{1.33}$$

The approximate weak form can therefore be written as:

$$\begin{aligned}
&\sum_e \int_{-1}^{1} \int_{-1}^{1} c_p^i N_{,j}^p C_{ijkm} d_s^k N_{,m}^s(\boldsymbol{\xi}) \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 = \\
&\quad \sum_e \int_{-1}^{1} \int_{-1}^{1} c_p^i N^p f_i \rho \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 + \\
&\quad \sum_{e \in S} \sum_{f \in \partial\Omega_{Ni}^e} \int_{-1}^{1} c_p^i N^p(\xi_{\bar{j}(f)}) t_i \, \boldsymbol{J}^{ef} d\xi_{\bar{j}(f)} \quad \forall \boldsymbol{c}^i \in C_i^h, i \in [1,2]
\end{aligned} \tag{1.34}$$

where $N^p(\xi_{\bar{j}(f)})$ is $N^p(\boldsymbol{\xi})$ with $\xi_{\bar{i}(f)}$ substituted for its value in edge f.

1.4.5.2 Hexahedral elements

The volume integrals over Ω^e can be transformed to volume integrals over the master element, and the surface integrals over $\partial\Omega_{Ni}^e$ can be converted to surface integrals over the corresponding faces of the master element. Let the Jacobian

of the coordinates $\boldsymbol{x}$ in element e with respect to the master elements reference coordinates $\boldsymbol{\xi}$ be:

$$\boldsymbol{J}^e = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} \tag{1.35}$$

where $|\boldsymbol{J}^e|$ is its determinant. An infinitesimal volume in element e is then:

$$d\boldsymbol{V} = |\boldsymbol{J}^e| d\boldsymbol{V}_{\boldsymbol{\xi}} = |\boldsymbol{J}^e| d\xi_1 d\xi_2 d\xi_3 \tag{1.36}$$

Let the conditional Jacobian on one of the surface faces $f \in \partial\Omega^e_{Ni}$ be J^{ef}. Let $\bar{i}(f) \in [1, 3]$ be the dimension along which face f is fixed in the master element such that $\xi_{\bar{i}(f)} = -1 \; or \; 1$. For example, if the face f is horizontal in the master element, $\bar{i}(f) = 3$. $\xi_{\bar{i}(f)}$ will be -1 if f corresponds to the bottom face of the master element and will be 1 if it corresponds to the top face. Let the non-fixed master element coordinates on face f be $\bar{\boldsymbol{\xi}}^f$. The face/conditional Jacobian is then given by:

$$\boldsymbol{J}^{ef} = \frac{\partial \boldsymbol{x}}{\partial \bar{\boldsymbol{\xi}}^f} \tag{1.37}$$

In the hexahedral case, the face Jacobian is a 3×2 matrix. If the non-fixed coordinate indices on face f are $\{\bar{j}(f), \bar{k}(f)\} = [1, 3] \setminus \{\bar{i}(f)\}$, an infinitesimal area on the

face f of element e can be written as:

$$d\boldsymbol{S} = |\boldsymbol{J}^{ef}| d\boldsymbol{S}_{\boldsymbol{\xi}} = |\boldsymbol{J}^{ef}| d\xi_{\bar{j}(f)} d\xi_{\bar{k}(f)} \tag{1.38}$$

The approximate weak form can therefore be written as:

$$\begin{aligned}
&\sum_e \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} c_p^i N_{,j}^p(\boldsymbol{\xi}) C_{ijkm} d_s^k N_{,m}^s(\boldsymbol{\xi}) \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 d\xi_3 = \\
&\quad \sum_e \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} c_p^i N^p(\boldsymbol{\xi}) f_i \rho \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 d\xi_3 + \\
&\quad \sum_{e\in S} \sum_{f\in\partial\Omega_{Ni}^e} \int_{-1}^{1}\int_{-1}^{1} c_p^i N^p(\xi_{\bar{j}(f)}, \xi_{\bar{k}(f)}) t_i \, |\boldsymbol{J}^{ef}| d\xi_{\bar{j}(f)} d\xi_{\bar{k}(f)} \quad \forall \boldsymbol{c}^i \in C_i^h, i \in [1,3]
\end{aligned} \tag{1.39}$$

where $N^p(\xi_{\bar{j}(f)}, \xi_{\bar{k}(f)})$ is $N^p(\boldsymbol{\xi})$ with $\xi_{\bar{i}(f)}$ substituted for its value in face f.

1.4.6 Gaussian quadrature integration

In order to evaluate the above integrals over $[-1, 1]$ 1, 2 or 3 times for line, face and volume integrals, typically the Gauss-Legendre quadrature rule is used to approximate each integral by a weighted summation of the integrand evaluated at n specific points. The points are selected such that a $(2n-1)^{st}$ order polynomial function would be integrated exactly using only n function evaluations. The following table gives the points and weights associated with the Gauss-Legendre quadrature using 1 to 4 points:

n	Points	Weights
1	0	2
2	$\pm\sqrt{\frac{1}{3}}$	1
3	0	$\frac{8}{9}$
	$\pm\sqrt{\frac{1}{3}}$	$\frac{5}{9}$
4	$\pm\sqrt{\frac{3}{7}-\frac{2}{7}\sqrt{\frac{6}{5}}}$	$\frac{18+\sqrt{30}}{36}$
	$\pm\sqrt{\frac{3}{7}+\frac{2}{7}\sqrt{\frac{6}{5}}}$	$\frac{18-\sqrt{30}}{36}$

Table 1.1: Gauss-Legendre quadrature points and weights

1.4.7 Matrix-vector weak form

Let $i'(p,i) = nsd\times(p-1)+i$ be the linearized degree of freedom index corresponding to the p^{th} node and the i^{th} spatial dimension. And let $j'(s,k) = nsd \times (s-1) + k$ be the linearized index corresponding to the s^{th} node and the k^{th} spatial dimension. Each (p,i) index tuple is a degree of freedom in the analysis problem. Moreover, let I_D be

$$I_D = \{(p-1)+i : p \in \partial\Omega^h_{Di}, \forall i \in [1, nsd]\} \tag{1.40}$$

which is the set of Dirichlet bounded degrees of freedom.

Additionally, let $\boldsymbol{K}^e$ be a hyper-sparse square matrix of size $(P\times nsd)\times(P\times nsd)$ associated with element e such that all the entries in the matrix are structural zeroes except the $(i'(p,i), j'(s,k))^{th}$ entries for all $(p,i,s,k) : (\exists A : CC^e_A = p) \wedge (\exists B : CC^e_B = s)$, i.e. nodes p and s are both connected to element e. Similarly, let $\boldsymbol{F}^e$ be a hyper-sparse vector of length $P \times nsd$ such that all the entries in the vector are

structural zeros except the $i'(p,i)^{th}$ entries for all $(p,i) : \exists A : CC_A^e = p$, i.e. node p is connected to element e.

For the quadrilateral case, the $(i'(p,i), j'(s,k))^{th}$ entry of $\boldsymbol{K}^e$ where $(\exists A : CC_A^e = p) \wedge (\exists B : CC_B^e = s)$ is:

$$K^e_{i'(p,i)j'(s,k)} = \int_{-1}^{1}\int_{-1}^{1} N^p_{,j}(\boldsymbol{\xi}) C_{ijkm} N^s_{,m}(\boldsymbol{\xi}) \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 \tag{1.41}$$

And the $i'(p,i)^{th}$ entry of $\boldsymbol{F}^e$ where $\exists A : CC_A^e = p$ is:

$$F^e_{i'(p,i)} = \int_{-1}^{1}\int_{-1}^{1} N^p(\boldsymbol{\xi}) f_i \rho \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 + \sum_{f \in \partial\Omega^e_{Ni}} \int_{-1}^{1} N^p(\xi_{\bar{j}(f)}) t_i \, J^{ef} d\xi_{\bar{j}(f)} \tag{1.42}$$

Similarly for the hexahedral case, , the $(i'(p,i), j'(s,k))^{th}$ entry of $\boldsymbol{K}^e$ where $(\exists A : CC_A^e = p) \wedge (\exists B : CC_B^e = s)$ is:

$$K^e_{i'(p,i)j'(s,k)} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} N^p_{,j}(\boldsymbol{\xi}) C_{ijkm} N^s_{,m}(\boldsymbol{\xi}) \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 d\xi_3 \tag{1.43}$$

And the $i'(p,i)^{th}$ entry of $\boldsymbol{F}^e$ where $\exists A : CC_A^e = p$ is:

$$F^e_{i'(p,i)} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} N^p(\boldsymbol{\xi}) f_i \rho \, |\boldsymbol{J}^e| d\xi_1 d\xi_2 d\xi_3 + \sum_{f \in \partial\Omega^e_{Ni}} \int_{-1}^{1}\int_{-1}^{1} N^p(\xi_{\bar{j}(f)}, \xi_{\bar{k}(f)}) t_i \, |\boldsymbol{J}^{ef}| d\xi_{\bar{j}(f)} d\xi_{\bar{k}(f)} \tag{1.44}$$

The weak form can therefore be written more compactly as:

$$\sum_e c_{i'} K^e_{i'j'} d_{j'} = \sum_e c_{i'} F^e_{i'} \tag{1.45}$$

where $c_{i'}$ and $d_{j'}$ are the linearized tensors c^i_p and d^k_s respectively. This equation should hold for all values of $\boldsymbol{c} \in \{\boldsymbol{c} : (c_{i'} \in \mathbb{R}, \forall i' \notin I_D) \wedge (c_{i'} = 0, \forall i' \in I_D)\}$. $\boldsymbol{K}^e$ is typically called the element stiffness matrix, and $\boldsymbol{F}^e$ is known as the element force vector.

Note also that (p, i) and (s, k) can be respectively swapped in the integral without affecting the function being integrated. This is because $N^p_{,j} C_{ijkm} N^s_{,m} = N^p_{,j} C_{kmij} N^s_{,m} = N^s_{,m} C_{kmij} N^p_{,j} = N^s_{,j} C_{kjim} N^p_{,m}$, so $K^e_{i'j'} = K^e_{j'i'}$. So each matrix $\boldsymbol{K}^e$ is symmetric. Also, note that all the terms multiplied by $c_{i'} = 0$ contribute nothing.

Notice that when a non-boundary element or a boundary element with no external load applied to it is dropped from the analysis, its contribution to the LHS is the term $c_{i'} K^e_{i'j'} d_{j'}$ and its contribution to the RHS is the term $c_{i'} F^e_{i'}$. Notice also that while a valid mesh has been assumed in the derivations, the vertices of such a mesh can be varied freely prior to analysis. So in a shape optimization context, one might consider leaving the mesh vertices as decision variables subject to mesh validity constraints. The vertex positions will not affect the above derivations so long as the cell-vertex connectivity does not change. Most topology optimization approaches however choose to keep vertex positions fixed and assign a binary decision variable to each removable element to either include it in the design or not. The latter approach makes it possible to use existing finite element analysis software to construct element

stiffness matrices and force vectors.

Because the above equation must hold for all $c_{i'}$ where $i' \notin I_D$, the following must hold:

$$\sum_e K^e_{i'j'} d_{j'} = \sum_e F^e_{i'} \quad \forall i' \notin I_D \tag{1.46}$$

Assembling this system of equations in matrix form, we get:

$$\bar{\boldsymbol{K}}\boldsymbol{d} = \bar{\boldsymbol{F}} \tag{1.47}$$

where $\bar{\boldsymbol{K}}$ is a matrix of shape $(P \times nsd - |I_D|, P \times nsd)$, and $\bar{\boldsymbol{F}}$ is a vector of length $P \times nsd - |I_D|$. Finally, because of the Dirichlet boundary condition, $d_{j'}$ is known $\forall j' \in I_D$. Let the column of known $d_{j'}$ values be $\bar{\boldsymbol{d}}$, and the remaining unknowns be $\boldsymbol{y}$. Additionally, let the columns of $\bar{\boldsymbol{K}}$ of indices $j' \in I_D$ be $\bar{\boldsymbol{K}}_D$ and the remaining columns be $\boldsymbol{K}$. The system of equations can therefore be reduced to:

$$\boldsymbol{K}\boldsymbol{y} = \bar{\boldsymbol{F}} - \bar{\boldsymbol{K}}_D\bar{\boldsymbol{d}} \tag{1.48}$$

It is also common to allow the adding of an additional nodal loading vector $\boldsymbol{F}_{conc}$ which represent a concentrated load on specific nodes without a Dirichlet boundary condition. While this is not physical, it is generally allowed computationally and is used in a number of standard topology optimization benchmark problems. Letting $\boldsymbol{F} = \bar{\boldsymbol{F}} - \bar{\boldsymbol{K}}_D\bar{\boldsymbol{d}} + \boldsymbol{F}_{conc}$, we can arrive at the final compact form of the system of

equations:

$$\boldsymbol{K}\boldsymbol{y} = \boldsymbol{F} \tag{1.49}$$

where $\boldsymbol{K}$ is a square matrix of shape $(P \times nsd - |I_D|, P \times nsd - |I_D|)$, and $\boldsymbol{F}$ is a vector of length $P \times nsd - |I_D|$. It is also common to add another term to the load

1.4.8 Analysis to topology optimization

In a topology optimization context, where each element is associated with a decision variable, $\boldsymbol{F}$ will depend on the design in any of the following cases:

1. The elements' weights are not 0.
2. The Neumann boundary conditions, i.e. surface loading, depends on the design.
3. An element with a non-zero Dirichlet boundary condition is not fixed in the design.

A non-zero body weight or a design-dependent Neumann boundary conditions will make $\bar{\boldsymbol{F}}$ depend on the design. If an element with a non-zero Dirichlet boundary condition is not fixed in the design, $\bar{\boldsymbol{K}}_D$ will depend on the design. When any of these conditions apply, the problem is said to have design-dependent loading. For simplicity for the rest of this thesis, the following will be assumed:

1. The body weight is negligible compared to the material strength and surface or node loading and can be ignored.

2. All the Dirichlet boundary conditions either have a zero value or the element on which a non-zero Dirichlet boundary condition applies is fixed to be part of the design and is not allowed to be removed.

3. The Neumann boundary conditions either have 0 value or the elements on which a surface load is applied, are guaranteed to be part of the design and are not allowed to be removed.

4. The concentrated load vector $\boldsymbol{F}_{conc}$ has a fixed value.

In topology optimization literature, the symbol $\boldsymbol{u}$ is usually used in place of $\boldsymbol{y}$ above and the symbol $\boldsymbol{f}$ is used in place of $\boldsymbol{F}$. For the rest of this document, the standard symbols will be used instead. The system of equations to be solved then becomes:

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f} \tag{1.50}$$

Let ρ_e be a pseudo-density associated with element e such that:

$$\boldsymbol{K} = \boldsymbol{K}_0 + \sum_e \rho_e \boldsymbol{K}_e \tag{1.51}$$

where:

1. $\boldsymbol{K}_e$ is the hyper-sparse element stiffness matrix of element e with the zero-valued Dirichlet bounded degrees of freedom eliminated.

2. $\boldsymbol{K}_0$ is the assembled stiffness matrix of all the elements that are fixed to be part of the design, e.g. because they have a non-zero-valued Dirichlet boundary

condition or a non-zero-valued Neumann boundary condition.

1.5 Topology optimization formulations

1.5.1 Design parameterization

Continuum topology optimization is fundamentally an infinitely sized discrete optimization problem of deciding which points have material and which do not. However, the problem cannot be solved in general in this form so the design needs to be parameterized using a finite dimensional parameterization. There are a number of techniques and algorithms that have been developed for continuum topology optimization over the last 3 decades:

1. Homogenization method (Bendsøe and Kikuchi, 1988)

2. Level set method (LSM) (Osher, 1988; Allaire et al., 2002; Wang et al., 2003, 2004; Guirguis and Aly, 2016)

3. Solid isotropic material with penalization (SIMP) (Bendsøe, 1989; Sigmund, 2001; Rojas-Labanda and Stolpe, 2015*a*)

4. Evolutionary and bi-evolutionary structural optimization (ESO / BESO) (Xie and Steven, 1992; Yang et al., 1998; Huang and Xie, 2010)

5. Genetic evolutionary structural optimization (GESO) (Sandgren et al., 1990; Liu et al., 2008).

Different methods use different design parameterizations. The most popular parameterization used by the homogenization method, SIMP, ESO/BESO and GESO

is using a so-called ground mesh of finite elements where each element is assigned decision variable. This reduces the infinitely sized problem to a finitely sized one of deciding which elements of the ground mesh should be assigned material and which should not. LSM uses a different design parameterization. A so-called parameteric level set function is specified over the base design. The level set of the function then defines the boundary between material and void in the design. The level set function has a finite number of parameters so the topology optimization problem reduces to the problem of finding the best parameter values.

The homogenization method was primarily developed to perform shape and topology optimization using composite materials having a layered structure of isotropic materials, and/or using materials with microstructural voids. The homoegenization method therefore relaxes the binary material variable to a continuous one between 0 and 1. While this relaxation can have physical justifications, such as using material with microstructural voids and the relative sheet thickness in 2D structures, in most cases it is desirable to simply create a design out of an existing manufacturable isotropic or anisotropic material, rather than proposing a new material or microstructure altogether. This is particularly important when designing 3D structures because a fractional material cannot have a simple physical interpretation such as sheet thickness in 2D structures. It is for this reason that the most popular topology optimization families of algorithms nowadays are SIMP, BESO and LSM all of which typically deal with isotropic materials and attempt to achieve a black-and-white topology as opposed to having grey areas of fractional material, where black represents material existence and white represents void.

The rest of this thesis will be dedicated to topology optimization methods that use the element-based design parameterization where each element is associated with a decision variable.

1.5.2 Compliance minimization

The linear, elastic, quasi-static, deterministic, volume constrained compliance minimization (VCCM) problem with design-independent loading has been a particularly well studied problem in the field of topology optimization since its inception. The goal of the VCCM problem is to design a statically supported structure subject to external loads such that the structure has minimum compliance subject to a maximum volume constraint, where compliance is defined as:

$$C = \boldsymbol{u}^T \boldsymbol{K} \boldsymbol{u} = \boldsymbol{f}^T \boldsymbol{K}^{-1} \boldsymbol{f} \tag{1.52}$$

i.e. twice the strain energy of the system, where $\boldsymbol{K}$ is the stiffness matrix assembled from element stiffness matrices, $\boldsymbol{u}$ is the displacement vector of all the degrees of freedom of the ground mesh, such that u_i corresponds to the i^{th} degree of freedom of the ground mesh, and $\boldsymbol{f}$ is the load vector assembled from surface and point loads. To evaluate the compliance of a certain mesh, an FEA is performed evaluating the degrees of freedom $\boldsymbol{u} = \boldsymbol{K}^{-1} \boldsymbol{f}$ subject to the loads and boundary conditions of the problem.

When using a ground mesh parameterization, one well documented challenge with the compliance minimization problem is that the optimal solution tends to

exhibit a chequerboard pattern due to an error in the FEA which under-estiamtes the compliance associated with some linear finite elements arranged in the chequerboard pattern (Díaz and Sigmund (1995)), when this arrangement is not even physically feasible. There are a number of approaches to eliminate the chequerboard (Sigmund and Petersson (1998)). One simple way is to use quadratic finite elements which is shown to eliminate this disadvantage of FEA (Rahmatalla and Swan (2004)). However, this approach increases the number of degrees of freedom in every function evaluation. More cheaply, a filter can be used blurring out the chequerboard pattern before proceeding with the FEA thus eliminating the advantage of chequerboard designs. Different types of filters will be discussed in section 1.5.4.

1.5.3 Compliance-constrained optimization

Instead of minimizing the compliance subject to a volume constrained, one may also want to put a constraint on the compliance and minimize the volume instead. This could more closely map a designer's objective of creating light-weight structures that don't significantly deform.

1.5.4 Filter types

Filters are typically used in topology optimization to achieve 1 or more of:

1. Reduce mesh dependence, e.g. chequerboard pattern.
2. Speed up the optimization algorithm's convergence
3. Increase feature thickness in the final design

Filters recompute the density of each element as the weighted mean of the densities of the elements in its neighbourhoods. There are various weighting schemes used in literature but it's unclear if or when any one method is more favourable than the others. The filtering operator is equivalent to multiplying a vector by a matrix $\boldsymbol{A}$ where each row in $\boldsymbol{A}$ sums up to 1. There are 2 ways to apply filters in topology optimization:

1. Density filters
2. Sensitivity filters

Density filters apply a linear filtering operator $\boldsymbol{A}$ to the decision variables $\boldsymbol{x}$, such that the filtered variables become:

$$\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} \tag{1.53}$$

Note that:

$$\frac{d(f(\boldsymbol{y}(\boldsymbol{x})))}{d\boldsymbol{x}} = \boldsymbol{A}' \frac{d(f(\boldsymbol{y}))}{d\boldsymbol{y}} \tag{1.54}$$

Density filters therefore filter the decision variables by $\boldsymbol{A}$ and the gradient vector by $\boldsymbol{A}'$. So if we define the pseudo-density ρ_e as:

$$\rho_e = y_e \tag{1.55}$$

oppositely signed partials in the same neighbourhood will tend to cancel out and

the pseudo-density of element e will be an average of the decision variables in the neighbourhood of element e.

Sensitivity filters on the other hand are mathematically improper filters in that they apply the filtering operator $\boldsymbol{A}$ to the gradient vector only without filtering the decision variables:

$$f(\boldsymbol{y}(\boldsymbol{x})) = f(\boldsymbol{x}) \tag{1.56}$$

$$\frac{d(f(\boldsymbol{y}(\boldsymbol{x})))}{d\boldsymbol{x}} = \boldsymbol{A}\frac{d(f(\boldsymbol{x}))}{d\boldsymbol{x}} \tag{1.57}$$

This is known to empirically achieve a similar effect of eliminating chequerboard patterns but may also cause convergence issues when using mathematical optimization algorithms that are sensitive to the accuracy of the gradient.

1.5.5 Nested vs simultaneous analysis and design

There are 2 broad categories of approaches to topology optimization algorithms:

1. Simultaneous analysis and design (SAND)

2. Nested analysis and design (NAND)

In the SAND formulation, the analysis equations e.g. $\boldsymbol{Ku} = \boldsymbol{f}$ is formulated as a constraint in the optimization problem. The optimization algorithm then attempts to find $\boldsymbol{x}$ and $\boldsymbol{u}$ simultaneously. Taking the VCCM problem as an example, the SAND

formulation is:

$$\underset{\boldsymbol{x},\boldsymbol{u}}{\text{minimize}} \quad C = \boldsymbol{u}^T \boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} \tag{1.58a}$$

subject to

$$\boldsymbol{K}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{f}, \tag{1.58b}$$

$$\boldsymbol{v}^T \boldsymbol{\rho}(\boldsymbol{x}) \leq V \times \mathbf{1}^T \boldsymbol{v}, \tag{1.58c}$$

$$x_e \in \{0, 1\} \quad \forall e \tag{1.58d}$$

where:

1. $\boldsymbol{v}$ is a vector of element volumes.
2. $\boldsymbol{\rho}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$, where $\boldsymbol{A}$ is the filter matrix.
3. $\rho_e(\boldsymbol{x})$ is the e^{th} element of $\boldsymbol{\rho}(\boldsymbol{x})$.
4. $\boldsymbol{K}(\boldsymbol{x}) = \boldsymbol{K}_0 + \sum_e \rho_e(\boldsymbol{x})\boldsymbol{K}_e$.

In the NAND formulation, the field function $\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{K}^{-1}\boldsymbol{f}$ is defined to be a function of the design variables $\boldsymbol{x}$ instead, making $\boldsymbol{x}$ the only decision variable in the problem. This reduces the number of decision variables and constraints in the

optimization problem. The NAND formulation is:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad C = \boldsymbol{f}^T(\boldsymbol{K}(\boldsymbol{x}))^{-1}\boldsymbol{f} \tag{1.59a}$$

subject to

$$\boldsymbol{v}^T\boldsymbol{\rho}(\boldsymbol{x}) \leq V \times \mathbf{1}^T\boldsymbol{v}, \tag{1.59b}$$

$$x_e \in \{0, 1\} \quad \forall e \tag{1.59c}$$

However, since the NAND formulation assumes that the global stiffness matrix $\boldsymbol{K}$ is invertible, the pseudo-densities $\boldsymbol{\rho}$ cannot be allowed to go to 0. This is because if all the elements surrounding a node have a 0 pseudo-density, $\boldsymbol{K}$ can be shown to be singular, which means that this node's degrees of freedoms are free to take any value without violating the physics constraints. It is for this reason that the pseudo-densities are defined differently in the NAND formulation:

$$\boldsymbol{\rho}(\boldsymbol{x}) = (1 - \rho_{min})\boldsymbol{A}\boldsymbol{x} + \rho_{min}\mathbf{1} \tag{1.60}$$

where ρ_{min} is a positive constant $\ll 1$ called the minimum pseudo-density. ρ_{min} approximates the pseudo-density of a void element without letting $\boldsymbol{K}$ be singular. However, ρ_{min} was also shown by the author to play a significant role in the VCCM problem (Tarek and Ray, 2020).

1.5.6 Solid isotropic material with penalization

So far the decision variables $\boldsymbol{x}$ were assumed to be binary, i.e. either 0 or 1. The SIMP algorithm however does a few things differently:

1. The binary constraint on x_e is relaxed to $0 \leq x_e \leq 1$ instead. This turns the optimization problem into a so-called nonlinear program (NLP) instead of an integer nonlinear program (INLP) in the NAND case, i.e a nonlinear program with all integer variables, or a mixed integer nonlinear program (MINLP) in the SAND case, i.e. a nonlinear program with mixed continuous and integer variables.

2. A so-called penalty function P is applied element-wise on the decision variables to indirectly enforce a mostly binary design at the end of the optimization.

3. An optional projection function $Proj$ is also applied element-wise to further enforce the pseudo-densities to be mostly binary.

$P(x)$ and $Proj(x)$ will be used from now on to refer to the (element-wise) application of the penalty and projection functions if x is a scalar (vector).

The term SIMP is often associated with the following choice of penalty function, also known as the power penalty:

$$P(x_e) = x_e^p \tag{1.61}$$

for some penalty value $p \geq 1$. Another popular penalty function choice is the

so-called rational penalty (Stolpe and Svanberg, 2001*a*):

$$P(x_e) = \frac{x_e}{1 + p(1 - x_e)} \tag{1.62}$$

for some $p \geq 0$. When using the rational penalty, the algorithm is often called the *rational approximation of material properties* (RAMP).

The most common projection function used is the regularized Heaviside projection function (Guest et al., 2004):

$$Proj(\rho_e) = 1 - e^{-\beta * \rho_e} + \rho_e * e^{-\beta} \tag{1.63}$$

for some choice of constant $\beta \geq 0$. When β is 0, the projection function is a no-op. Increasing β makes the function approximate a step function at $\rho_e = 0.0$.

The SIMP NAND formulation of the VCCM problem is therefore:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad C = \boldsymbol{f}^T(\boldsymbol{K}(\boldsymbol{x}))^{-1}\boldsymbol{f} \tag{1.64a}$$

subject to

$$\boldsymbol{v}^T\boldsymbol{\rho}(\boldsymbol{x}) \leq V \times \mathbf{1}^T\boldsymbol{v}, \tag{1.64b}$$

$$0 \leq x_e \leq 1 \quad \forall e \tag{1.64c}$$

where $\boldsymbol{\rho}(\boldsymbol{x})$ is the vector of pseudo-densities obtained after sequentially applying to $\boldsymbol{x}$:

1. A chequerboard density filter of the form $f_1(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$,

2. An interpolation of the form $f_2(y) = (1-\rho_{min})y + \rho_{min}$ applied element-wise for some small $\rho_{min} > 0$,

3. A penalty such as the power penalty $f_3(z) = P(z;p)$ applied element-wise for some penalty value p, and

4. A projection function $Proj$ such as the regularized Heaviside projection applied element-wise.

The interpolation and penalization steps may be swapped.

While the original SIMP without projection was first developed as a heuristic with no theoretical results, later Rietz (2001) proved some theoretical properties of SIMP and the VCCM problem. More specifically, Rietz proved that a finite value of p will converge to a 0-1 solution of the VCCM problem under the following assumptions:

(1) All the elements have unit volume $v_e = 1$,

(2) The volume threshold $V \times \sum_e v_e$ is integer, and

(3) The partial derivative $\frac{\partial C}{\partial P(x_e)}$ is upper and lower bounded by finite, strictly negative values for all elements e.

If additionally the original binary VCCM had a unique global solution and each SIMP subproblem is solved globally, then a finite value of p was shown to be sufficient for convergence to the global optimal solution of the binary VCCM problem. A few years later, Martínez (2005) relaxed some of the assumptions made by Rietz and

provided a way to reason about the convergence of SIMP for problems other than the VCCM problem.

Stolpe and Svanberg (2001*a*) proved a similar result for the RAMP scheme for compliance minimization problems showing that there exists a finite penalty value p_{max} at which the compliance function becomes concave, thus admitting some binary solutions as some of the optima of the NLP. A few years later, Martínez (2005) relaxed some of the assumptions made by Rietz and provided a way to reason about the convergence of SIMP for problems other than the VCCM problem.

1.5.7 Continuation SIMP

The traditional SIMP solves a single nonlinear approximation using a single value for p, typically $p = 3$ with the power penalty function. However, it is also common practice to apply a so-called continuation on one or more parameters of the SIMP algorithm. For example, the so-called *continuation* SIMP can solve a sequence of NLP sub-problems with increasing values of penalty p, e.g from $p = 1$ to $p = 5$.

Algorithm 1 Decaying tolerance continuation SIMP

Require: Nonlinear problem callback $Prob$, initial topology $\boldsymbol{x}_0$, penalty function P, initial penalty p_0, penalty step Δp, maximum penalty p_{max}, tolerance tol, tolerance decay factor $\theta = 1$, minimum tolerance $tol_{min} = 0$.

1: $\boldsymbol{x} = \boldsymbol{x}_0$
2: **for** p in $p_0{:}\Delta p{:}p_{max}$ **do**
3: $\quad \boldsymbol{x} = \text{SOLVE}(Prob(P, p), \boldsymbol{x}, tol)$
4: $\quad tol = max(\theta \times tol, tol_{min})$
5: **end for**
6: **return** $\boldsymbol{x}$

The basic continuation SIMP (CSIMP) algorithm in literature is shown in Algorithm 1, where a fixed penalty step Δp is used and the tolerance used to terminate

the NLP solve is optionally decayed by some factor $\theta \leq 1$. The CSIMP algorithm introduces a number of additional hyper-parameters:

1. Initial and final penalties, p_0 and p_{max}.

2. Tolerance decay parameter θ.

3. Minimum tolerance, tol_{min}.

Choosing these hyper-parameters is non-trivial and can be problem-dependent and/or scale-dependent.

CSIMP is commonly associated with its ability to *avoid local minima*. Rojas-Labanda and Stolpe (2015*a*) reviewed a number of CSIMP works in literature showing a consensus that this ability is largely empirical. Empirical evidence was also reestablished in the same paper by comparing CSIMP to a single penalty SIMP on 375 problems showing that the former was more robust at converging to a Karush-Kuhn-Tucker (KKT) point of the desired tolerance for the test problems. On the other hand, Stolpe and Svanberg (2001*b*) demonstrated a few problems where CSIMP fails to produce a 0-1 design even for large penalty values since the fractional solution of the subproblem with $p = 1$ is also a KKT point for the subproblem with higher penalty values.

While CSIMP has no rigorous theoretical grounds, empirically CSIMP has been a good heuristic for obtaining good designs in many cases (Rojas-Labanda and Stolpe, 2015*a*). The intuition behind CSIMP seems to come from the VCCM problem where the original unpenalized nonlinear approximation with $p = 1$ is a proven convex program (Svanberg, 1994), whereas using $p > 1$ makes the problem

non-convex. Consequently, at $p = 1$ the relaxed VCCM problem, if feasible with a solution space of non-empty interior, has a global optimal solution that can be obtained in polynomial time and whose objective value is a lower bound on the binary VCCM's global optimal value (Boyd and Vandenberghe, 2009). It is therefore hoped that the solution using $p = 1.1$ will be close enough to that of $p = 1$ while making the solution slightly less fractional. So assuming the penalty trick is really effective at eliminating fractional values, for high enough values of p, a mostly 0-1 solution would be obtained that is hopefully not too far from the fractional lower bound solution.

The main drawback of CSIMP compared to the single penalty SIMP is that it usually requires a large number of FEA simulations to converge.

1.5.8 Evolutionary structural optimization

ESO and its extension BESO also use a ground mesh parameterization however the binary variable is not relaxed for optimization. BESO can be viewed as a form of binary gradient descent that relies on first order information to compute the *sensitivies* of the elements based on which a certain number of elements is removed and/or added in every iteration until convergence.

Two problem classes where the BESO algorithm was originally developed to solve are: the VCCM problem, and the stress constrained volume minimization problem. The element removal/adding criteria is typically closely related to the gradient of the objective and/or constraints' mathematical form. For instance in

stress constrained problems, mesh elements with low stress values get removed while elements close to high stress areas get added to the design according to a specific rule to ensure convergence. In compliance minimization problems, the cell compliance is the criteria instead where elements with low compliance get removed and elements close to high compliance areas get added to the design.

The BESO algorithm has been improved over the years subject to criticism by Zhou and Rozvany (2001) and Rozvany (2009) for its inability to solve the tie-beam problem shown in Figure 1-1. Huang and Xie (2010) then proposed the improved *soft-kill* BESO. While still a binary gradient descent algorithm, the soft-kill BESO relies on $\rho_{min} > 0$ and a penalty function much like SIMP to compute the sensitivities. These changes enabled the soft-kill BESO to solve the tie-beam problem, however its biggest disadvantage yet is that it is not straightforward to generalize it to problems with arbitrary constraints and objectives often requiring taking the Lagrangian relaxation of the problem and solving for the Lagrangian dual variables while finding the optimal topology (primal variables) (Huang and Xie, 2010). However, in the VCCM class of problems, BESO exhibits a very competitive performance. Another similar binary gradient descent algorithm for topology optimization was also proposed by Browne (2013).

1.5.9 Genetic evolutionary structural optimization

While ESO is an acronym for evolutionary structural optimization, it has no direct relationship with the Darwinian evolutionary algorithms (EAs), in fact ESO/BESO

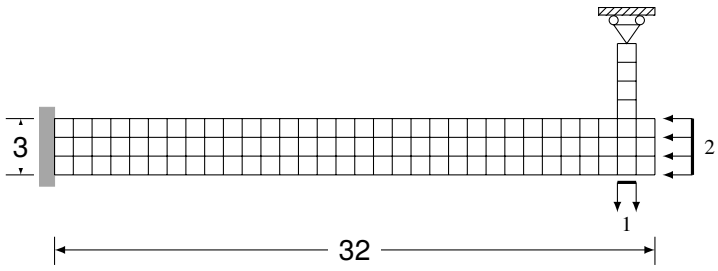


Figure 1-1: Tie beam problem

is a completely deterministic algorithm. Some specializations of the Darwinian EAs to topology optimization was also attempted in papers like Sandgren (1990); Liu et al. (2008). However, the so-called genetic ESO (GESO) method proposed by Liu et al. (2008) can also be viewed as a stochastic version of ESO/BESO since an unconventional definition of the solution population is used, where a single solution is treated as a population and every mesh element is a member of the population. This is different from the conventional meaning of a population in genetic algorithms (GAs), where a population must be made of many solutions each of which has an objective value and can be either feasible or not. The GA-like technique used by Liu et al. (2008) results in a stochastic re-ordering of the removal and addition of elements, compared to the deterministic BESO algorithm, which therefore makes it more like a binary stochastic gradient descent algorithm. Traditional EAs, e.g. GAs (Sandgren, 1990), and more generally zero-order optimization methods, e.g. pattern search (Guirguis and Aly, 2016), tried in literature fail to perform as well as SIMP and BESO since they require a prohibitively large number of function evaluations, more than 10s of thousands, even for toy problems, where each function evaluation can be an expensive FEA. Also the scale of problems required to be solved in practice can involve anywhere from 1000s to 100,000,000s of decision variables

and zero-order methods are known for not scaling well when the number of decision variables is even in the 100s. On the other hand, SIMP and BESO tend to converge to good solutions in 100s of function evaluations at the very most, often in much less. Also huge scale problems with 10s of millions of decision variables were successfully solved in literature, e.g. Aage et al. (2015), using these methods.

1.5.10 Topology optimization problem classes

A number of mechanical structure problems were studied and successfully solved in topology optimization literature where some of the objectives were:

(1) Compliance minimization (Bendsøe, 1989),

(2) Material volume/cost minimization (Payten and Law, 1998; Bruggi and Duysinx, 2012),

(3) Maximum stress minimization (Yang and Chen, 1996; Lian et al., 2017), or

(4) Minimum eigenvalue maximization (Neves et al., 1995; Rahmatalla and Swan, 2003; Munk et al., 2017).

Some of the constraints used were:

(1) Volume constraint (Bendsøe, 1989),

(2) Maximum compliance constraint (Bruggi and Duysinx, 2012; Collet et al., 2017),

(3) Maximum displacement constraint (Huang and Xie, 2010),

(4) Local/global stress constraints (Payten and Law, 1998; Amir, 2017),

(5) Fatigue constraints (Oest and Lund, 2017; Collet et al., 2017), and/or

(6) Global stability/buckling/bifurcation constraints (Kočvara and Stingl, 2004; Browne, 2013; Deng and Suresh, 2017).

Some of the mechanical systems studied were:

(1) Linear, elastic, quasi-static systems (Bendsøe, 1989),

(2) Nonlinear, compliant mechanisms (Sigmund, 1997),

(3) Nonlinear, elasto-plastic systems (Maute and Ramm, 1998; Amir, 2017), or

(4) Linear/nonlinear vibrating systems (Zargham et al., 2016).

And finally the loads handled were:

(1) Single or multiple (Allaire and Jouve, 2004),

(2) Design-independent or design-dependent (Lee et al., 2012),

(3) Static or dynamic (Zhang et al., 2016), and

(4) Deterministic or stochastic (Zhang et al., 2016).

Multiobjective problems combining multiple of the above objectives have also been scarcely considered (Suresh, 2010; Sato et al., 2017). Many of the papers cited above have used either SIMP or BESO variants.

1.6 Nonlinear programming

Part of the SIMP algorithm is solving an NLP using a constrained mathematical optimization algorithm. Zero-order algorithms don't scale too well so they are typically not used for topology optimization. Second order algorithms requiring Hessians are typically too computationally prohibitive because the Hessian with respect to per-element decision variables will typically be a huge dense matrix. Therefore, first order algorithms are by far the most commonly used class of algorithms in topology optimization. There are 2 notable algorithms that were developed specifically for SIMP:

1. The method of moving asymptotes (MMA) (Svanberg, 1987) and its globally convergent extension (Svanberg, 2002).

2. The convex linearization method (CONLIN) (Fleury, 1989)

Other algorithms that have been used in literature (Rojas-Labanda and Stolpe, 2015*a*; Pereira et al., 2004; Tarek and Ray, 2021) include:

1. The primal-dual interior point optimizer (IPOPT) (Wächter and Biegler, 2006), with the limited memory Broyden, Fletcher, Goldfarb, and Shann (l-BFGS) approximation of the inverse Lagrangian Hessian (Nocedal and Wright, 2006)

2. The first order augmented Lagrangian (AugLag) algorithm (Bertsekas, 1996).

In this section, the original MMA, IPOPT and AugLag will be explained further because they are used in the thesis. Notably, MMA cannot handle equality constraints directly, only inequality constraints, while IPOPT and AugLag handle both equality and inequality constraints. MMA, IPOPT and AugLag are all local optimization algorithms that seek to find a so-called KKT point, which under certain assumptions is a local optimum solution.

In this section, the terms *linear* and *affine* will be used inter-changeably in the context of classifying a function/constraint as either a linear/affine or a nonlinear function/constraint. Even though there is a distinction between the 2 terms, optimization literature commonly uses these 2 terms inter-changeably for the same purpose.

1.6.1 Formulations

There are generally a number of ways to formulate the same optimization or decision problem using mathematical functions and decision variables. For instance, consider the following inequality constrained optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} && f_0(\boldsymbol{x}) \\ &\text{subject to} && \\ &f_i(\boldsymbol{x}) \leq 0 && \forall i = 1..I, \\ &l_j \leq x_j \leq u_j && \forall j = 1..V \end{aligned} \tag{1.65}$$

where I is the number of constraints, V is the number of decision variables, f_i is a potentially nonlinear scalar-valued function of $\boldsymbol{x}$ and $\boldsymbol{l}$ and $\boldsymbol{u}$ are finite vectors of lower and upper bounds on the decision vector $\boldsymbol{x}$ respectively. Two formulations are considered equivalent for optimization purposes if every optimal solution of one formulation maps to an optimal solution in the other formulation. In the following sub-sections, 2 relevant re-formulation tricks will be presented. For more re-formulation techniques, the readers are referred to the excellent online book titled: "MOSEK Modeling Cookbook" by MOSEK ApS (ApS, 2018).

1.6.1.1 A linear objective is all you need

For instance, consider the alternative formulation:

$$\begin{aligned} &\underset{\boldsymbol{x},\, c}{\text{minimize}} \quad c \\ &\text{subject to} \\ &f_0(\boldsymbol{x}) \leq c, \\ &f_i(\boldsymbol{x}) \leq 0 \qquad \forall i = 1..I, \\ &l_j \leq x_j \leq u_j \quad \forall j = 1..V \end{aligned} \tag{1.66}$$

where we added a new decision variable c and a new constraint $f_0(\boldsymbol{x}) \leq c$ to the previous formulation and changed the objective to minimizing c which is a linear function of the decision variables.

It is clear that if $\boldsymbol{x}^*$ is an optimal solution of formulation 1.151, then $(\boldsymbol{x}, c) = (\boldsymbol{x}^*, f_0(\boldsymbol{x}^*))$ is optimal in formulation 1.66. This is because $f_0(\boldsymbol{x}^*)$ is the lowest value

$f_0(\boldsymbol{x})$ can take for any feasible $\boldsymbol{x}$ according to the remaining inequality constraints, and $c = f_0(\boldsymbol{x}^*)$ is the lowest value the objective c can take without violating the constraint on c.

Conversely, if $(\boldsymbol{x}^*, c^*)$ is an optimal solution of formulation 1.66, $\boldsymbol{x}^*$ must be optimal in formulation 1.151. Since c shows up in only one constraint:

$$f_0(\boldsymbol{x}) \leq c \tag{1.67}$$

in formulation 1.66, for $(\boldsymbol{x}^*, c^*)$ to be optimal, c^* must be equal to $f_0(\boldsymbol{x}^*)$. And since this is lowest value c can take, it must be the lowest value $f_0(\boldsymbol{x})$ can take without violating any of the other constraints on $\boldsymbol{x}$, which in turn makes $\boldsymbol{x}^*$ an optimal solution to formulation 1.151. This completes the proof.

1.6.1.2 Slack variables

Another common formulation transformation is changing all the inequality constraints to equality constraints except the variable bounds. This can be done by introducing additional slack variables. Let $\boldsymbol{s}$ be a vector of so-called slack variables with length I, where s_i is the i^{th} element of the vector associated with constraint i

for $i \in 1..I$. Formulation 1.151 can be re-written as:

$$\begin{aligned}
&\underset{\boldsymbol{x},\,\boldsymbol{s}}{\text{minimize}} \quad f_0(\boldsymbol{x}) \\
&\text{subject to} \\
&f_i(\boldsymbol{x}) + s_i = 0 \quad \forall i = 1..I, \\
&l_j \leq x_j \leq u_j \quad \forall j = 1..V, \\
&s_i \geq 0 \quad \forall i = 1..I
\end{aligned} \tag{1.68}$$

Since $\boldsymbol{s}$ doesn't show up in the objective, in order to prove that formulations 1.151 and 1.68 are equivalent for optimization purposes, it suffices to show that every feasible solution $\boldsymbol{x}$ to formulation 1.151 can be mapped a feasible solution $(\boldsymbol{x}, \boldsymbol{s})$ in formulation 1.68 and vice versa.

It is simple to show that if $\boldsymbol{x}$ is a feasible solution in formulation 1.151, that $(\boldsymbol{x}, \boldsymbol{s})$ where $\boldsymbol{s} : s_i = -f_i(\boldsymbol{x}) \, \forall i \in 1..I$ is a feasible solution in formulation 1.68. This is because for $\boldsymbol{x}$ to be feasible in formulation 1.151, $f_i(\boldsymbol{x})$ must be non-positive for all i which makes s_i non-negative. Conversely, if $(\boldsymbol{x}, \boldsymbol{s})$ is a feasible solution to formulation 1.68, $\boldsymbol{x}$ is clearly feasible in formulation 1.151 because s_i will be non-negative which implies that $f_i(\boldsymbol{x}) \leq 0$ for all i. This completes the proof.

1.6.1.3 Equality constraints and empty interiors

While the following equality constraint:

$$f(\boldsymbol{x}) = 0 \tag{1.69}$$

is in theory equivalent to the following 2 inequality constraints:

$$0 \leq f(\boldsymbol{x}) \leq 0 \tag{1.70}$$

this transformation does not change the shape or nature of the feasible domain. Not all NLP algorithms that can handle inequality constraints can also handle problems with 2 inequality constraints derived from an equality constraint like this. For instance, the convergence proof of the globally convergent MMA algorithm (Svanberg, 2002) assumes that the feasible domain must have a non-empty interior. Equality constraints introduce an "empty interior". A domain $D \subseteq \mathbb{R}^V$ is said to have a non-empty interior if $\exists(\boldsymbol{x}_c \in D, r > 0)$ such that $\boldsymbol{x}_c + \boldsymbol{u} \in D$ for all $\boldsymbol{u}$ in $\{\boldsymbol{u} : \boldsymbol{u} \in \mathbb{R}^V \wedge ||\boldsymbol{u}||_2 \leq r\}$. Since equality constraints change the feasible domain to a lower dimensional manifold embedded in $\mathbb{R}^V$, there exist no such $(\boldsymbol{x}_c, r)$ in equality constrained problems. Therefore, the MMA algorithm will fail if the interior of the feasible domain is empty.

Linear equality constraints are generally an exception though. While technically still introducing an empty interior, most NLP algorithms can either natively handle linear equality constraints or the linear equality constrained NLP can be re-parameterized such that the interior of the feasible domain is no longer empty. IPOPT and AugLag can handle linear and even nonlinear equality constraints natively. Sequential quadratic programming (SQP) methods (Boyd and Vandenberghe, 2009) can only handle linear equality constraints natively whereas nonlinear ones must get locally approximated by linear ones in an iterative process. MMA however

doesn't handle either linear or nonlinear equality constraints natively. But it can be made to support linear ones with a simple nullspace re-parameterization trick.

Let the following be the linear equality constrained NLP with a nonlinear objective and inequality constraints:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \quad f_0(\boldsymbol{x}) \\ &\text{subject to} \\ &f_i(\boldsymbol{x}) \leq 0 \qquad \forall i = 1..I, \\ &l_j \leq x_j \leq u_j \quad \forall j = 1..V, \\ &\boldsymbol{A}\boldsymbol{x} = b \end{aligned} \tag{1.71}$$

where $\boldsymbol{A}$ is a constant matrix of size $(M \times V)$ and $\boldsymbol{b}$ is a constant vector. Let $\boldsymbol{x}_0$ be an arbitrary point that satisfies the $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ constraint, e.g. $\boldsymbol{x}_0 = \boldsymbol{A}^+\boldsymbol{b}$ where $\boldsymbol{A}^+$ is the Moore-Penrose pseudoinverse of $\boldsymbol{A}$ (Golub and Loan, 1996). Let $\boldsymbol{N}$ be the nullspace matrix of $\boldsymbol{A}$ such that $\boldsymbol{A} \times \boldsymbol{N} = \boldsymbol{0}$, where $\boldsymbol{N}$ is of size $V \times U$. Every feasible solution $\boldsymbol{x}$ to the constraints $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ can now be written as:

$$\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{N}\boldsymbol{y} \tag{1.72}$$

for some $\boldsymbol{y} \in \mathbb{R}^U$, where $U < V$. Substituting for $\boldsymbol{x}$ in formulation 1.71, we get:

$$\begin{aligned} &\underset{\boldsymbol{y}}{\text{minimize}} \quad f_0(\boldsymbol{x}_0 + \boldsymbol{N}\boldsymbol{y}) \\ &\text{subject to} \\ &f_i(\boldsymbol{x}_0 + \boldsymbol{N}\boldsymbol{y}) \leq 0 \quad \forall i = 1..I, \\ &\boldsymbol{l} \leq \boldsymbol{x}_0 + \boldsymbol{N}\boldsymbol{y} \leq \boldsymbol{u} \end{aligned} \tag{1.73}$$

The re-parameterization essentially limits the feasible domain to the nullspace of the linear constraints since the "interior" is non-empty once we limit ourselves to the nullspace. A similar re-parameterization trick can also be used for some nonlinear manifolds, e.g. optimization on the surface of a hyper-sphere.

Beside the iterative linear approximation of nonlinear equality constraints, MMA can also be made to approximately support nonlinear equality constraints by relaxing it as follows:

$$-\epsilon \leq f(\boldsymbol{x}) \leq \epsilon \tag{1.74}$$

where $\epsilon > 0$ which creates a non-empty interior.

1.6.2 Regularity conditions

There are a number of optimization algorithms that can solve a general NLP without assuming convexity. Most NLP algorithms tend to converge to a locally optimal solution without global optimality guarantees and this suffices in many applications.

For simplicity, assume all the inequality constraints have been converted to equality constraints using slack variables. Let the NLP with equality constraints be:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) \tag{1.75a}$$

subject to

$$\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}, \tag{1.75b}$$

$$\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \tag{1.75c}$$

where $\boldsymbol{c}(\boldsymbol{x})$ is of length E. Additionally, let f and $\boldsymbol{c}$ be continuous and twice differentiable functions.

A point $\boldsymbol{x}$ is called a regular point if it satisfies one of the so-called constraint qualification conditions. Three common constraint qualifications for NLPs are:

1. Linear constraint qualification (LCQ): $\boldsymbol{c}(\boldsymbol{x})$ is an affine function.

2. Linear independence constraint qualification (LICQ): the rows of $\nabla \boldsymbol{c}(\boldsymbol{x})$ and the gradients of the active (i.e. satisfied at equality) bound constraints are linearly independent at $\boldsymbol{x}$.

3. Mangasarian-Fromovitz constraint qualification (MFCQ): the rows of $\nabla \boldsymbol{c}(\boldsymbol{x})$ are linearly independent at $\boldsymbol{x}$ and there exists a direction vector $\boldsymbol{d} \in \mathbb{R}^V$ where $d_i > 0$ if $x_i = l_i$, $d_i < 0$ if $x_i = u_i$, and $\nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{d} = \boldsymbol{0}$.

Alternatively, if the problem is convex and the nonlinear inequality constraints are not converted to equality constraints, every feasible point is regular if $\exists \boldsymbol{x}$ such that all the inequality constraints are satisfied but not active (i.e. satisfied at a strict

inequality) and all the linear equality constraints are satisfied. This condition is known as Slater's condition (Boyd and Vandenberghe, 2009).

1.6.3 Sufficient optimality conditions for regular points

Let:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, z_+, z_-) = f(\boldsymbol{x}) + \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} + (\boldsymbol{x} - \boldsymbol{u})^T z_+ - (\boldsymbol{x} - \boldsymbol{l})^T z_- \tag{1.76}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^E$ is the vector of Lagrangian multipliers of the nonlinear constraints, $z_- \in \mathbb{R}^V_+$ is the vector of Lagrangian multipliers of the $\geq$ bound constraints, and $z_+ \in \mathbb{R}^V_+$ is the vector of Lagrangian multipliers of the $\leq$ bound constraints.

If $\boldsymbol{x}$ is regular and is a local optimum of the NLP, then $\exists(\boldsymbol{\lambda}, z_+, z_-)$ such that:

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda}, z_+, z_-) = \mathbf{0} \tag{1.77}$$

$$\boldsymbol{c}(\boldsymbol{x}) = 0 \tag{1.78}$$

$$\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \tag{1.79}$$

$$z_+ \geq \mathbf{0} \tag{1.80}$$

$$z_- \geq \mathbf{0} \tag{1.81}$$

$$(\boldsymbol{x} - \boldsymbol{u})^T z_+ = 0 \tag{1.82}$$

$$(\boldsymbol{x} - \boldsymbol{l})^T z_- = 0 \tag{1.83}$$

$$\nabla \boldsymbol{c}(\boldsymbol{x})^T \nabla^2_{\boldsymbol{xx}} L(\boldsymbol{x}, \boldsymbol{\lambda}, z_+, z_-) \nabla \boldsymbol{c}(\boldsymbol{x}) \succcurlyeq \mathbf{0} \tag{1.84}$$

These conditions are known as the KKT sufficient conditions for optimality and $\boldsymbol{x}$ would be called a KKT point (Boyd and Vandenberghe, 2009). Condition 1.77 is known as the stationarity condition which generalizes the 0 gradient condition for unconstrained NLPs. Conditions 1.78 and 1.79 are known as primal feasibility conditions. Conditions 1.80 and 1.81 are known as dual feasibility conditions. Finally, condition 1.84 is known as the second order KKT optimality condition which generalizes the second order optimality condition for unconstrained NLPs, where $\boldsymbol{A} \succcurlyeq \mathbf{0}$ when $\boldsymbol{A}$ is a matrix means that $\boldsymbol{A}$ must be positive semi-definite.

Note that not every local optimal solution to the NLP must be a regular point or by consequence a KKT point for that matter. Therefore, these conditions are not necessary conditions for optimality for general NLPs. However for convex problems, if Slater's condition is satisfied, these conditions are both necessary and sufficient and the local/global optimum is guaranteed to be a regular and a KKT point (Boyd and Vandenberghe, 2009). All the optimization algorithms presented next seek to find a locally optimal KKT point.

1.6.4 Method of moving asymptotes

Consider the following inequality constrained optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \quad f_0(\boldsymbol{x}) \\ &\text{subject to} \\ &f_i(\boldsymbol{x}) \leq 0 \qquad \forall i = 1..I, \\ &l_j \leq x_j \leq u_j \quad \forall j = 1..V \end{aligned} \tag{1.85}$$

where I is the number of inequality constraints, V is the number of decision variables, f_i is a potentially nonlinear scalar-valued function of $\boldsymbol{x}$ and $\boldsymbol{l}$ and $\boldsymbol{u}$ are finite vectors of lower and upper bounds on the decision vector $\boldsymbol{x}$ respectively.

The first MMA algorithm proposed by Svanberg (1987) relied on a separable convex approximation of the objective and constraint functions. A function $f(\boldsymbol{x})$ is linearly approximated with respect to either $t_{l,j}$ or $t_{u,j}$, where:

$$t_{l,j} = t_l(x_j; L_j) = \frac{1}{x_j - L_j} \tag{1.86}$$

$$t_{u,j} = t_u(x_j; U_j) = \frac{1}{U_j - x_j} \tag{1.87}$$

for all $j \in 1..V$ for some constants L_j and U_j, where $l_j \leq L_j < U_j \leq u_j$, and L_j and U_j are known as the asymptotes of the approximation. The choice of which function, $t_{l,j}$ or $t_{u,j}$, to approximate f with respect to, for each variable x_j, depends on the sign of the partial derivative $\frac{\partial f}{\partial x_j}$ such that the approximation is convex.

More specifically, let the MMA approximation of $f(\boldsymbol{x})$ around $\bar{\boldsymbol{x}}$ be:

$$\bar{f}(\boldsymbol{x};\bar{\boldsymbol{x}}) = f(\bar{\boldsymbol{x}}) + \Sigma_j \bar{f}_j(x_j;\bar{\boldsymbol{x}}) \tag{1.88}$$

Let $\bar{f}_j(x_j;\bar{\boldsymbol{x}})$ be:

$$\bar{f}_j(x_j;\bar{\boldsymbol{x}}) = \begin{cases} \Big(t_l(x_j;L_j) - t_l(\bar{x}_j;L_j)\Big)\frac{\partial f}{\partial t_{l,j}}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) < 0 \\ \Big(t_u(x_j;U_j) - t_u(\bar{x}_j;U_j)\Big)\frac{\partial f}{\partial t_{u,j}}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) \geq 0 \end{cases} \tag{1.89}$$

where $\bar{x}_j$ is the j^{th} element of $\bar{\boldsymbol{x}}$ and $\frac{\partial f}{\partial t_{l,j}}$ and $\frac{\partial f}{\partial t_{u,j}}$ are:

$$\frac{\partial f}{\partial t_{l,j}} = \frac{\partial f}{\partial x_j} \Big/ \frac{dt_{l,j}}{dx_j} = -(x_j - L_j)^2 \frac{\partial f}{\partial x_j} \tag{1.90}$$

$$\frac{\partial f}{\partial t_{u,j}} = \frac{\partial f}{\partial x_j} \Big/ \frac{dt_{u,j}}{dx_j} = (U_j - x_j)^2 \frac{\partial f}{\partial x_j} \tag{1.91}$$

$\bar{f}_j(x_j;\bar{\boldsymbol{x}})$ can therefore be written as:

$$\bar{f}_j(x_j;\bar{\boldsymbol{x}}) = \begin{cases} -\Big(t_l(x_j;L_j) - t_l(\bar{x}_j;L_j)\Big)(\bar{x}_j - L_j)^2 \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) < 0 \\ \Big(t_u(x_j;U_j) - t_u(\bar{x}_j;U_j)\Big)(U_j - \bar{x}_j)^2 \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) \geq 0 \end{cases} \tag{1.92}$$

The signs of the gradient and Hessian of $\bar{f}(\boldsymbol{x};\bar{\boldsymbol{x}})$ therefore depend mostly on $t_l(x_j;L_j)$ and $t_u(x_j;U_j)$. Since the approximation is separable in $\boldsymbol{x}$, the Hessian of

$\bar{f}(\boldsymbol{x};\bar{\boldsymbol{x}})$ wrt $\boldsymbol{x}$ is a diagonal matrix where the j^{th} diagonal element is:

$$\frac{\partial^2 \bar{f}}{\partial x_j^2} = \begin{cases} -\frac{d^2 t_{l,j}}{dx_j^2}(\bar{x}_j - L_j)^2 \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) < 0 \\ \frac{d^2 t_{u,j}}{dx_j^2}(U_j - \bar{x}_j)^2 \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}), & \frac{\partial f}{\partial x_j}(\bar{\boldsymbol{x}}) \geq 0 \end{cases} \tag{1.93}$$

Since $\frac{d^2 t_{l,j}}{dx_j^2}$ and $\frac{d^2 t_{u,j}}{dx_j^2}$ are both positive for all x_j, it is clear that the MMA approximation is always going to be convex.

In each iteration of the MMA algorithm, convex approximations of the objective and all the constraint functions are formed around the current solution and the approximate problem is solved to optimality while restricting the decision variables to be between the asymptotes $\boldsymbol{L}$ and $\boldsymbol{U}$ instead of the original bounds $\boldsymbol{l}$ and $\boldsymbol{u}$. This is similar to the trust region approach. Let the restricted convex approximation of the original nonlinear program be:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \quad \bar{f}_0(\boldsymbol{x};\bar{\boldsymbol{x}}) \\ &\text{subject to} \\ &\bar{f}_i(\boldsymbol{x};\bar{\boldsymbol{x}}) \leq 0 \qquad \forall i = 1..I, \\ &\alpha_j \leq x_j \leq \beta_j \quad \forall j = 1..V \end{aligned}$$

where each decision variable x_j is restricted to be between $\alpha_j = max(l_j, L_j)$ and $\beta_j = min(u_j, U_j)$ and $\bar{f}_i(\boldsymbol{x};\bar{\boldsymbol{x}})$ is the MMA approximation of $f_i(\boldsymbol{x})$ around $\bar{\boldsymbol{x}}$. In order to form the approximation above, the gradient of the objective and the full Jacobian of the constraint functions need to be computed first. This typically

limits the MMA algorithm's scalability in handling many constraints where the full Jacobian can be expensive to form, e.g. in stress-constrained topology optimization. However if only a few constraints exist, the MMA algorithm is usually quite robust when starting from a feasible solution. Most importantly is that once the approximate problem is formed, no more calls to the objective or constraint kernels are required by the primal-dual algorithm to solve the approximate problem to optimality.

Once the approximation is formed, the separable nature of the MMA convex approximation then allows the approximate nonlinear program to be solved to optimality using an efficient primal-dual Lagrangian optimization algorithm (Svanberg, 1987). The dual of the convex approximation problem above is the following lower bound constrained nested optimization problem:

$$\underset{\boldsymbol{\lambda} \geq \mathbf{0}}{\text{maximize}} \quad \min_{\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = r_0 + \sum_{j=1}^{V} \mathcal{L}_j$$

where

$$\mathcal{L}_j(x_j;\boldsymbol{\lambda}) = \frac{p_{0,j} + \sum_i \lambda_i p_{i,j}}{U_j - x_j} + \frac{q_{0,j} + \sum_i \lambda_i q_{i,j}}{x_j - L_j} \tag{1.94}$$

$$r_0 = f_0(\bar{\boldsymbol{x}}) - \sum_{j=1}^{V} \frac{p_{0,j}}{U_j - \bar{x}_j} + \frac{q_{0,j}}{\bar{x}_j - L_j} \tag{1.95}$$

$$p_{i,j} = \begin{cases} (U_j - \bar{x}_j)^2 \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) & \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) > 0 \\ 0 & \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) \leq 0 \end{cases} \tag{1.96}$$

$$q_{i,j} = \begin{cases} 0 & \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) \geq 0 \\ -(\bar{x}_j - L_j)^2 \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) & \frac{\partial f_i}{\partial x_j}(\bar{\boldsymbol{x}}) < 0 \end{cases} \tag{1.97}$$

The primal optimal solution $\boldsymbol{x}^*$ of the convex approximation has an analytic form as a function of the dual solution $\boldsymbol{\lambda}$:

$$x_j^*(\boldsymbol{\lambda}) = \begin{cases} \alpha_j & \frac{\partial \mathcal{L}_j}{\partial x_j}(\alpha_j;\boldsymbol{\lambda}) \geq 0 \\ \beta_j & \frac{\partial \mathcal{L}_j}{\partial x_j}(\beta_j;\boldsymbol{\lambda}) \leq 0 \\ \frac{(p_{0,j}+\sum_i \lambda_i p_{i,j})^{1/2} L_j + (q_{0,j}+\sum_i \lambda_i q_{i,j})^{1/2} U_j}{(p_{0,j}+\sum_i \lambda_i p_{i,j})^{1/2} + (q_{0,j}+\sum_i \lambda_i q_{i,j})^{1/2}} & otherwise \end{cases} \tag{1.98}$$

Since the optimization of the approximate nonlinear program requires no calls to the objective or constraint kernels, its performance is independent of the computational complexity of computing the objective value, its gradient, the constraint values and their Jacobian.

One of the biggest numerical challenges in topology optimization is that the scale of the objective and constraints can often be vastly different. For example in

the VCCM problem, the objective value scales up with the volume of the design and the Young's modulus while the volume fraction constraint does not. Moreover, the ∞-norm of the gradient of the volume fraction constraint scales down as the number of elements increases while that of the gradient of the compliance function does not. This usually results in problems where the objective and constraints span multiple orders of magnitude. This difference in scale usually means that more iterations are needed to converge to sufficiently large or sufficiently small elements of the dual solution $\boldsymbol{\lambda}$ that satisfies the first order KKT optimality conditions. More specifically in the MMA algorithm, this mostly translates to requiring more iterations to solve the approximate problem to optimality and usually not many more subproblems to be solved. However as mentioned earlier, the additional iterations needed to solve the approximate subproblem do not require any additional calls to the objective or constraint kernels. This is an especially attractive feature of the MMA algorithm in topology optimization since scaling issues are common and the computational time is usually dominated by the time it takes to compute the objective, the constraints and their gradients.

Another attractive feature of the MMA algorithm is the low sensitivity of the algorithm's performance to most of the parameters. In this thesis, the lower bound constrained dual of the convex approximation is solved using a log-barrier method with the nonlinear conjugate gradient algorithm (Nocedal and Wright, 2006). The parameters of the subproblem optimizer generally have little effect on the performance of the MMA algorithm for the reason mentioned above. The main parameters which tend to affect the performance of the algorithm by changing the number of

subproblems that need to be solved are: s_{init} used to specify the initial asymptotes, s_{incr} used to widen the asymptotes of each variable and s_{decr} used to tighten the asymptotes for each variable. These parameters are described in details by Svanberg (1987). Finally, a tolerance tol must be picked to terminate the algorithm.

1.6.5 Primal-dual interior point method

In this section, the primal-dual interior point optimizer (IPOPT) as described in Wächter and Biegler (2006) and implemented in the IPOPT software will be described. This algorithm and its implementation are well tested in practice. However, the main disadvantage of the IPOPT algorithm is its complexity and the large number of hyper-parameters as will be seen next.

Consider the following equality constrained optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \\ &\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}, \\ &\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \end{aligned} \tag{1.99}$$

where the length of $\boldsymbol{x}$ is V, E is the number of elements in the output of the vector-valued function $\boldsymbol{c}(\boldsymbol{x})$, f is a potentially nonlinear scalar-valued function of $\boldsymbol{x}$ and $\boldsymbol{l}$ and $\boldsymbol{u}$ are the lower and upper bounds on the variables. If there are inequality constraints, they can be converted to equality constraints by adding slack variables. Additionally, let $\mathcal{I}_{\boldsymbol{l}}$ be the set of variable indices with a finite lower bound and $\mathcal{I}_{\boldsymbol{u}}$

be the set of variable indices with a finite upper bound. Additionally, assume that the lower and upper bounds are not equal for any variable. If a variable's lower and upper bounds are equal, it can be fixed and removed from the optimization.

In the IPOPT algorithm by Wächter and Biegler (2006), a so-called log barrier method (Boyd and Vandenberghe, 2009) is used to guarantee that $\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}$ remains satisfied at every intermediate solution if the initial solution is within the bounds. This is achieved using a so-called barrier function such as $-\mu\Big(\sum_{i \in \mathcal{I}_{\boldsymbol{l}}} \log(x_i - l_i) + \sum_{i \in \mathcal{I}_{\boldsymbol{u}}} \log(u_i - x_i)\Big)$ for some $\mu > 0$ which would go to ∞ if any of the decision variables approaches one of its finite bounds. This creates a barrier stopping the optimizer from every reaching the finite bound. It can be shown that if a decreasing geometric series of values μ are used that the solutions of the sub-problems will follow a so-called critical path converging to a KKT point in convex problems satisfying Slater's condition (Boyd and Vandenberghe, 2009). However, if the optimal solution is exactly on a boundary, the problem can become numerically unstable near the optimal solution. Therefore, the variable bounds are typically relaxed slightly. In particular, the lower bound l_i where $i \in \mathcal{I}_{\boldsymbol{l}}$ is relaxed by $tol \times \max(1, |l_i|)$, and the upper bound u_i where $i \in \mathcal{I}_{\boldsymbol{u}}$ is relaxed by $tol \times \max(1, |u_i|)$. $\boldsymbol{l}$ and $\boldsymbol{u}$ will refer to the relaxed lower and upper bounds from now on.

One final catch in the barrier problem formulation is that if the set of optimal points to the original problem does not consist of isolated points, but contains an unbounded connected subspace, e.g. a line to ∞ or $-\infty$ for one or more variables, the log-barrier term may become $-\infty$. This is only possible if a variable is bounded from one side only where it's allowed to go to $\pm\infty$ from the other side. If this happens, the

minimum objective value of the barrier problem becomes $-\infty$ even when the original problem's objective value is bounded. For this reason, the following additional term is added to the barrier objective:

$$\kappa_d \mu \sum_{i \in \mathcal{I}_{\boldsymbol{l}} \setminus \mathcal{I}_{\boldsymbol{u}}} \log(x_i - l_i) + \kappa_d \mu \sum_{i \in \mathcal{I}_{\boldsymbol{u}} \setminus \mathcal{I}_{\boldsymbol{l}}} \log(u_i - x_i) \tag{1.100}$$

for some small constant $\kappa_d \in (0, 1)$. This way, divergence of variables having only one bound is penalized. The effect of this additional term is reduced as μ decreases and can be shown to not affect the local convergence proof (Wächter and Biegler, 2006). The barrier sub-problem is therefore defined as:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \phi_\mu(\boldsymbol{x}) = f(\boldsymbol{x}) - \mu \sum_{i \in \mathcal{I}_{\boldsymbol{l}}} \log(x_i - l_i) - \mu \sum_{i \in \mathcal{I}_{\boldsymbol{u}}} \log(u_i - x_i) \\ & + \kappa_d \mu \sum_{i \in \mathcal{I}_{\boldsymbol{l}} \setminus \mathcal{I}_{\boldsymbol{u}}} \log(x_i - l_i) + \kappa_d \mu \sum_{i \in \mathcal{I}_{\boldsymbol{u}} \setminus \mathcal{I}_{\boldsymbol{l}}} \log(u_i - x_i) \end{aligned} \tag{1.101}$$

subject to

$$\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}$$

The KKT stationarity condition is therefore:

$$\nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} - \boldsymbol{z}_{\boldsymbol{l}} - \boldsymbol{z}_{\boldsymbol{u}} = \boldsymbol{0} \tag{1.102}$$

where $\boldsymbol{\lambda}$ is the vector Lagrangian multipliers associated with the equality constraint

$\boldsymbol{c}(\boldsymbol{x}) = \mathbf{0}$, $\boldsymbol{z_l}$ is a vector whose i^{th} element is:

$$z_{l_i} = \begin{cases} \frac{\mu}{x_i - l_i} & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \frac{(1-\kappa_d)\mu}{x_i - l_i} & i \in \mathcal{I}_{\boldsymbol{l}} \setminus \mathcal{I}_{\boldsymbol{u}} \\ 0 & otherwise, \end{cases} \tag{1.103}$$

and $\boldsymbol{z_u}$ is a vector whose i^{th} element is:

$$z_{u_i} = \begin{cases} \frac{\mu}{u_i - x_i} & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \frac{(1-\kappa_d)\mu}{u_i - x_i} & i \in \mathcal{I}_{\boldsymbol{u}} \setminus \mathcal{I}_{\boldsymbol{l}} \\ 0 & otherwise, \end{cases} \tag{1.104}$$

Additionally, let $\boldsymbol{Z_l}$ be the diagonal matrix whose diagonal is $\boldsymbol{z_l}$, $\boldsymbol{Z_u}$ be the diagonal matrix whose diagonal is $\boldsymbol{z_u}$, $\boldsymbol{X_l}$ be the diagonal matrix whose diagonal is:

$$\tilde{x}_{l_i} = \begin{cases} x_i - l_i & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \frac{x_i - l_i}{1-\kappa_d} & i \in \mathcal{I}_{\boldsymbol{l}} \setminus \mathcal{I}_{\boldsymbol{u}} \\ 0 & otherwise, \end{cases} \tag{1.105}$$

and $\boldsymbol{X_u}$ be the diagonal matrix whose diagonal is:

$$\tilde{x}_{u_i} = \begin{cases} u_i - x_i & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \frac{u_i - x_i}{1 - \kappa_d} & i \in \mathcal{I}_{\boldsymbol{u}} \setminus \mathcal{I}_{\boldsymbol{l}} \\ 0 & otherwise, \end{cases} \tag{1.106}$$

The first order KKT sufficient conditions for optimality assuming the constraint qualifications are satisfied can therefore be written as:

$$\nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} - \boldsymbol{z_l} - \boldsymbol{z_u} = \mathbf{0} \tag{1.107}$$

$$\boldsymbol{c}(\boldsymbol{x}) = \mathbf{0} \tag{1.108}$$

$$\boldsymbol{X_l Z_l 1} - \mu \mathbf{1} = \mathbf{0} \tag{1.109}$$

$$\boldsymbol{X_u Z_u 1} - \mu \mathbf{1} = \mathbf{0} \tag{1.110}$$

$$\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u} \tag{1.111}$$

$$\boldsymbol{z_l} \geq \mathbf{0} \tag{1.112}$$

$$\boldsymbol{z_u} \geq \mathbf{0} \tag{1.113}$$

where $\mathbf{1}$ is a vector of ones. Conditions 1.109 and 1.110 ensure that the relationship between $\boldsymbol{X_l}$, $\boldsymbol{Z_l}$, $\boldsymbol{X_u}$ and $\boldsymbol{Z_u}$ is maintained according to the definitions of $\boldsymbol{z_l}$ and $\boldsymbol{z_u}$. Assume l_i and u_i are both finite for some index i. Given that μ is positive, condition 1.109 guarantees that $x_i - l_i$ and z_i will either be both positive or both negative. However even if we start from a value for $x_i > l_i$ and $z_i > 0$, during the intermediate line search steps, there is a non-zero chance that both $x_i - l_i$ and z_i may become

negative simultaneously for some i, hence the need for the explicit non-negativity constraints on $\boldsymbol{z_l}$ and $\boldsymbol{z_u}$, and the bounds constraints on $\boldsymbol{x}$.

The primal-dual interior point optimizer presented in Wächter and Biegler (2006) computes an approximate solution to the barrier problem for a fixed value of μ then decreases μ and continues the solution of the next barrier problem from the approximate solution of the previous one. The optimality error is defined as:

$$E_\mu(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{z_l}, \boldsymbol{z_u}) := \max \left\{ \frac{||\nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} - \boldsymbol{z_l} - \boldsymbol{z_u}||}{s_d}, ||\boldsymbol{c}(\boldsymbol{x})||_1, \frac{||\boldsymbol{X_l Z_l 1} - \mu \boldsymbol{1}||_\infty}{s_c}, \frac{||\boldsymbol{X_u Z_u 1} - \mu \boldsymbol{1}||_\infty}{s_c} \right\} \tag{1.114}$$

with scaling parameters $s_d, s_c \geq 1$ defined below. $E_0(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{z_l}, \boldsymbol{z_u})$ is the optimality error for the original problem. Let $(\tilde{\boldsymbol{x}}_*, \tilde{\boldsymbol{\lambda}}_*, \tilde{\boldsymbol{z}}_{\boldsymbol{l},*}, \tilde{\boldsymbol{z}}_{\boldsymbol{u},*})$ be the approximate solution of the barrier problem. The algorithm terminates if $E_0(\tilde{\boldsymbol{x}}_*, \tilde{\boldsymbol{\lambda}}_*, \tilde{\boldsymbol{z}}_{\boldsymbol{l},*}, \tilde{\boldsymbol{z}}_{\boldsymbol{u},*}) \leq tol$ for some tolerance tol.

It is possible that the magnitudes of $\boldsymbol{\lambda}$, $\boldsymbol{z_l}$ and $\boldsymbol{z_u}$ might become very large, e.g. if the gradients of the active constraints are nearly linearly dependent in a solution to the original problem. This is why the scaling factors s_d and s_c are used to make it easier to satisfy the terminating condition around these solutions. s_d and s_c are therefore chosen as:

$$s_d = \max \left\{ s_{max}, \frac{||\lambda||_1 + ||\boldsymbol{z_l}||_1 + ||\boldsymbol{z_u}||_1}{E + V} \right\} / s_{max} \tag{1.115}$$

$$s_c = \max \left\{ s_{max}, \frac{||\boldsymbol{z_l}||_1 + ||\boldsymbol{z_u}||_1}{V} \right\} / s_{max} \tag{1.116}$$

Using these scaling factors, a component of the optimality error will be scaled, whenever the average value of the multipliers becomes larger than a fixed number $s_{max} \geq 1$

Denoting with j the outer iteration counter of barrier sub-problems. Each barrier problem j is terminated when:

$$E_{\mu_{j-1}}(\tilde{\boldsymbol{x}}_{*,j}, \tilde{\boldsymbol{\lambda}}_{*,j}, \tilde{\boldsymbol{z}}_{\boldsymbol{l},*,j}, \tilde{\boldsymbol{z}}_{\boldsymbol{u},*,j}) \leq \kappa_\epsilon \mu_j \tag{1.117}$$

for a constant $\kappa_\epsilon > 0$. The new barrier parameter μ_j is then obtained using:

$$\mu_j = \max\left\{\frac{tol}{10}, \min\left\{\kappa_\mu \mu_{j-1}, \mu_{j-1}^{\theta_\mu}\right\}\right\} \tag{1.118}$$

with constants $\kappa_\mu \in (0, 1)$ and $\theta_\mu \in (1, 2)$. In this way, μ is eventually decreased at a superlinear rate. On the other hand, this update rule stops μ from becoming smaller than necessary given the desired tolerance tol.

In order to solve the barrier problem for a given value $\mu = \mu_j$, damped Newton's method is applied to the primal-dual optimality conditions. Here we use k to denote the iteration counter for the inner iterations when solving the barrier problem. Given an iterate $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{z}_{\boldsymbol{l},k}, \boldsymbol{z}_{\boldsymbol{u},k})$ with $\boldsymbol{l} < \boldsymbol{x}_k < \boldsymbol{u}$ and $\boldsymbol{z}_{\boldsymbol{l},k}, \boldsymbol{z}_{\boldsymbol{u},k} > \boldsymbol{0}$, some search directions $(\boldsymbol{d}_k^{\boldsymbol{x}}, \boldsymbol{d}_k^{\boldsymbol{\lambda}}, \boldsymbol{d}_k^{\boldsymbol{z}_l}, \boldsymbol{d}_k^{\boldsymbol{z}_u})$ are obtained using the regularized linearization of the

optimality conditions (excluding the inequality constraints):

$$\begin{bmatrix} \boldsymbol{W}_k + \delta_w \boldsymbol{I} & \boldsymbol{A}_k & -\boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{A}_k^T & -\delta_c \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{Z}_{\boldsymbol{l},k} & \boldsymbol{0} & \boldsymbol{X}_{\boldsymbol{l},k} & \boldsymbol{0} \\ \boldsymbol{Z}_{\boldsymbol{u},k} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{X}_{\boldsymbol{u},k} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \\ \boldsymbol{d}_k^{\boldsymbol{z_l}} \\ \boldsymbol{d}_k^{\boldsymbol{z_u}} \end{pmatrix} = - \begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \boldsymbol{z}_{\boldsymbol{l},k} - \boldsymbol{z}_{\boldsymbol{u},k} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{X}_{\boldsymbol{l},k} \boldsymbol{Z}_{\boldsymbol{l},k} \boldsymbol{1} - \mu_j \boldsymbol{1} \\ \boldsymbol{X}_{\boldsymbol{u},k} \boldsymbol{Z}_{\boldsymbol{u},k} \boldsymbol{1} - \mu_j \boldsymbol{1} \end{pmatrix} \tag{1.119}$$

where δ_w and δ_c are non-negative constants and $\boldsymbol{A}_k := \nabla \boldsymbol{c}(\boldsymbol{x}_k)^T$ and $\boldsymbol{W}_k := \nabla^2_{\boldsymbol{xx}} \mathcal{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$ which is the Hessian of the Lagrangian function of the original problem, where:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) := f(x) + \boldsymbol{c}(\boldsymbol{x})^T \boldsymbol{\lambda} \tag{1.120}$$

Note that the Lagrangian terms from the bounds constraint $\boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{u}$ in the original problem can be ignored since their contribution to the Hessian is 0. When the Hessian of the Lagrangian is not available, an l-BFGS approximation (Nocedal and Wright, 2006) of the Hessian can be used instead. This changes the IPOPT algorithm from a second order algorithm to a first order one. And the Newton update becomes a quasi-Newton update.

When δ_w and δ_c are 0s, we get the linearization of the primal-dual first order KKT conditions. However, since the Hessian of the Lagrangian function isn't necessarily a positive definite matrix when the NLP is non-convex and in order to guarantee that the search directions obtained are descent directions (Wächter and

Biegler, 2006), a positive value for δ_w is used. Additionally, if the gradients of the active constraints are (nearly) linearly dependent, the matrix on the LHS will be (nearly) singular even if the Hessian of the Lagrangian is positive definite. To stop the matrix from becoming singular in this case, a positive value for δ_c can be used. If the matrix is so ill-conditioned even with large values for δ_w and δ_c, the algorithm gives up on finding a search direction and attempts a feasibility restoration step, hoping that the matrix has better properties close to feasible points. For more on the feasibility restoration phase or on the heuristic used to choose δ_w and δ_c, the readers are referred to section 3.1 in Wächter and Biegler (2006).

Instead of solving the non-symmetric system of equations above, one can instead change the system as such:

$$\begin{bmatrix} \boldsymbol{W}_k + \delta_w \boldsymbol{I} & \boldsymbol{A}_k & -\boldsymbol{I} & -\boldsymbol{I} \\ \boldsymbol{A}_k^T & -\delta_c \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{X}_l^+ \boldsymbol{Z}_{l,k} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{X}_u^+ \boldsymbol{Z}_{u,k} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^x \\ \boldsymbol{d}_k^\lambda \\ \boldsymbol{d}_k^{z_l} \\ \boldsymbol{d}_k^{z_u} \end{pmatrix} = - \begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \boldsymbol{z}_{l,k} - \boldsymbol{z}_{u,k} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{z}_{l,k} - \mu_j \boldsymbol{X}_l^+ \boldsymbol{1} \\ \boldsymbol{z}_{u,k} - \mu_j \boldsymbol{X}_u^+ \boldsymbol{1} \end{pmatrix} \tag{1.121}$$

where $\boldsymbol{X}^+$ is the Moore-Penrose pseudoinverse of $\boldsymbol{X}$ (Golub and Loan, 1996). Adding the third and fourth equations to the first one, the third and fourth blocks of

coefficients of the first equation will be eliminated.

$$\begin{bmatrix} \boldsymbol{W}_k + \delta_w \boldsymbol{I} + \boldsymbol{\Sigma}_k & \boldsymbol{A}_k & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{A}_k^T & -\delta_c \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{Z}_{\boldsymbol{l},k} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{Z}_{\boldsymbol{u},k} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \\ \boldsymbol{d}_k^{\boldsymbol{z_l}} \\ \boldsymbol{d}_k^{\boldsymbol{z_u}} \end{pmatrix} = - \begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{1} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{1} \\ \boldsymbol{c}(\boldsymbol{x}_k) \\ \boldsymbol{z}_{\boldsymbol{l},k} - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{1} \\ \boldsymbol{z}_{\boldsymbol{u},k} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{1} \end{pmatrix} \tag{1.122}$$

where $\boldsymbol{\Sigma}_k = \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{Z}_{\boldsymbol{l},k} + \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{Z}_{\boldsymbol{u},k}$. Therefore, one can now solve for $\boldsymbol{d}_k^{\boldsymbol{x}}$ and $\boldsymbol{d}_k^{\boldsymbol{\lambda}}$ by solving the following symmetric linear system:

$$\begin{bmatrix} \boldsymbol{W}_k + \delta_w \boldsymbol{I} + \boldsymbol{\Sigma}_k & \boldsymbol{A}_k \\ \boldsymbol{A}_k^T & -\delta_c \boldsymbol{I} \end{bmatrix} \begin{pmatrix} \boldsymbol{d}_k^{\boldsymbol{x}} \\ \boldsymbol{d}_k^{\boldsymbol{\lambda}} \end{pmatrix} = - \begin{pmatrix} \nabla f(\boldsymbol{x}_k) + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k - \mu_j \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{1} - \mu_j \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{1} \\ \boldsymbol{c}(\boldsymbol{x}_k) \end{pmatrix} \tag{1.123}$$

then use the value of $\boldsymbol{d}_k^{\boldsymbol{x}}$ to find $\boldsymbol{d}_k^{\boldsymbol{z_l}}$ and $\boldsymbol{d}_k^{\boldsymbol{z_u}}$ using:

$$\boldsymbol{d}_k^{\boldsymbol{z_l}} = -\boldsymbol{z}_{\boldsymbol{l},k} + \mu_j \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{1} - \boldsymbol{X}_{\boldsymbol{l}}^+ \boldsymbol{Z}_{\boldsymbol{l},k} \boldsymbol{d}_k^{\boldsymbol{x}} \tag{1.124}$$

$$\boldsymbol{d}_k^{\boldsymbol{z_u}} == -\boldsymbol{z}_{\boldsymbol{u},k} + \mu_j \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{1} - \boldsymbol{X}_{\boldsymbol{u}}^+ \boldsymbol{Z}_{\boldsymbol{u},k} \boldsymbol{d}_k^{\boldsymbol{x}} \tag{1.125}$$

Having computed the solution to the search directions linear system of equations to obtain $(\boldsymbol{d}_k^{\boldsymbol{x}}, \boldsymbol{d}_k^{\boldsymbol{\lambda}}, \boldsymbol{d}_k^{\boldsymbol{z_l}}, \boldsymbol{d}_k^{\boldsymbol{z_u}})$, now two step sizes $\alpha_k, \alpha_k^z \in (0, 1]$ have to be determined

in order to obtain the next iterate using:

$$\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k^{\boldsymbol{x}} \tag{1.126}$$

$$\boldsymbol{\lambda}_{k+1} := \boldsymbol{\lambda}_k + \alpha_k \boldsymbol{d}_k^{\boldsymbol{\lambda}} \tag{1.127}$$

$$\boldsymbol{z_l}_{,k+1} := \boldsymbol{z_l}_{,k} + \alpha_k^z \boldsymbol{d}_k^{\boldsymbol{z_l}} \tag{1.128}$$

$$\boldsymbol{z_u}_{,k+1} := \boldsymbol{z_u}_{,k} + \alpha_k^z \boldsymbol{d}_k^{\boldsymbol{z_u}} \tag{1.129}$$

When the step sizes are chosen, the bounds on $\boldsymbol{x}$, $\boldsymbol{z_l}$ and $\boldsymbol{z_u}$ are enforced. Let $\tau_j \in (0, 1)$ be the so-called fraction-to-the-boundary parameter given by:

$$\tau_j = \max\{\tau_{min}, 1 - \mu_j\} \tag{1.130}$$

where $\tau_{min} \in (0, 1)$ is its minimum value. The step sizes are then chosen using the following fraction-to-the-boundary rule:

$$\alpha_k^{max} := \max\left\{\alpha \in (0, 1] : \left(\tau \boldsymbol{l} + (1 - \tau_j)\boldsymbol{x}_k\right) \leq \left(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k^{\boldsymbol{x}}\right) \leq \left(\tau \boldsymbol{u} + (1 - \tau_j)\boldsymbol{x}_k\right)\right\} \tag{1.131}$$

$$\alpha_k^z := \max\left\{\alpha \in (0, 1] : \left(\boldsymbol{z_l}_{,k} + \alpha \boldsymbol{d}_k^{\boldsymbol{z_l}} \geq (1 - \tau_j)\boldsymbol{z_l}_{,k}\right) \wedge \left(\boldsymbol{z_u}_{,k} + \alpha \boldsymbol{d}_k^{\boldsymbol{z_u}} \geq (1 - \tau_j)\boldsymbol{z_u}_{,k}\right)\right\} \tag{1.132}$$

where the actual step size $\alpha_k \in (0, \alpha_k^{max}]$ is determined using a backtracking line

search procedure exploring a decreasing sequence of trial step sizes:

$$\alpha_{k,l} = 2^{-l}\alpha_k^{max} \tag{1.133}$$

with $l = 0, 1, 2, \ldots$.

Choosing the step sizes is done using a so-called line search filter method. During line search, a step size is considered acceptable if it leads to an acceptable reduction in the objective value of the barrier problem $\phi_\mu(\boldsymbol{x})$, and/or the constraint violation $\theta(\boldsymbol{x}) = ||\boldsymbol{c}(\boldsymbol{x})||_1$ with a certain emphasis on the latter quantity. Let $\boldsymbol{x}_k(\alpha_{k,l})$ be:

$$\boldsymbol{x}_k(\alpha_{k,l}) := \boldsymbol{x}_k + \alpha_{k,l}\boldsymbol{d}_k^{\boldsymbol{x}} \tag{1.134}$$

There are 2 acceptable criteria used to accept a trial step size. The following conditions are called the "switching conditions":

$$\nabla\phi_{\mu_j}(\boldsymbol{x}_k)^T\boldsymbol{d}_k^{\boldsymbol{x}} < 0 \tag{1.135}$$

$$\alpha_{k,l}[-\nabla\phi_{\mu_j}(\boldsymbol{x}_k)^T\boldsymbol{d}_k^{\boldsymbol{x}}]^{S_\phi} > \delta[\theta(\boldsymbol{x}_k)]^{S_\theta} \tag{1.136}$$

for some constants $\delta > 0$, $S_\phi > 1$ and $S_\theta \geq 1$. If $\theta(\boldsymbol{x}_k) \leq \theta^{min}$, for some constant $\theta^{min} \in (0, \infty)$, and the following so-called "switching conditions" are satisfied, a step size $\alpha_{k,l}$ is considered acceptable if the following so-called Armijo condition

is satisfied:

$$\phi_{\mu_j}(\boldsymbol{x}_k(\alpha_{k,l})) \leq \phi_{\mu_j}(\boldsymbol{x}_k) + \eta_\phi \alpha_{k,l} \nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}} \tag{1.137}$$

for some constant $\eta_\theta \in (0, 0.5)$ and $(\theta(\boldsymbol{x}_k(\alpha_{k,l})), \phi_{\mu_j}(\boldsymbol{x}_k(\alpha_{k,l}))) \notin \mathcal{F}_k$ where $\mathcal{F}_k$ is a filter set (defined below) used to reject a trial solution if its constraint violation is too high or if it has a close barrier objective value and constraint violation value as certain previous iterates $\boldsymbol{x}_k$. This filter method ensures that the algorithm cannot cycle between 2 points that alternately decrease the constraint violation and barrier objective value. If the trial step is rejected, a so-called second order correction (SOC) step is performed. The SOC step is explained below.

If either $\theta(\boldsymbol{x}_k) > \theta^{min}$ or the switching conditions are violated, a step size $\alpha_{k,l}$ is considered acceptable if either:

$$\theta(\boldsymbol{x}_k(\alpha_{k,l})) \leq (1 - \gamma_\theta)\theta(\boldsymbol{x}_k) \tag{1.138}$$

$$(\theta(\boldsymbol{x}_k(\alpha_{k,l})), \phi_{\mu_j}(\boldsymbol{x}_k(\alpha_{k,l}))) \notin \mathcal{F}_k \tag{1.139}$$

or

$$\phi_{\mu_j}(\boldsymbol{x}_k(\alpha_{k,l})) \leq \phi_{\mu_j}(\boldsymbol{x}_k) - \gamma_\phi \theta(\boldsymbol{x}_k) \tag{1.140}$$

$$(\theta(\boldsymbol{x}_k(\alpha_{k,l})), \phi_{\mu_j}(\boldsymbol{x}_k(\alpha_{k,l}))) \notin \mathcal{F}_k \tag{1.141}$$

is satisfied for some constants $\gamma_\theta, \gamma_\phi \in (0, 1)$. If the trial step is rejected, the SOC

step is performed.

The filter set $\mathcal{F}_k$ is initialized as:

$$\mathcal{F}_0 = \{(\theta, \phi) \in \mathbb{R}^2 : \theta \geq \theta^{max}\} \tag{1.142}$$

Then after every iteration k if an accepted trial point $\boldsymbol{x}_k$ doesn't satisfy either the switching conditions or the Armijo rule, the filter set is updated as:

$$\mathcal{F}_{k+1} = \mathcal{F}_k \cup \left\{(\theta, \phi) \in \mathbb{R}^2 : \left(\theta \geq (1 - \gamma_\theta)\theta(\boldsymbol{x}_k)\right) \wedge \left(\phi \geq \phi_{\mu_j}(\boldsymbol{x}_k) - \gamma_\phi \theta(\boldsymbol{x}_k)\right)\right\} \tag{1.143}$$

Every time the barrier parameter μ is decreased in the outer iterations of the algorithm, the filter set is reset to its definition at $k = 0$.

If the backtracking procedure (SOC steps included) fails to find an acceptable trial step $\alpha_{k,l} \geq \alpha_k^{min}$, where:

$$\alpha_k^{min} := \gamma_\alpha \times \begin{cases} \min\left\{\gamma_\theta, \frac{\gamma_\theta \theta(\boldsymbol{x}_k)}{-\nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}}}, \frac{\delta[\theta(\boldsymbol{x}_k)^{S_\theta}]}{[-\nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}}]^{S_\phi}}\right\} & \left(\nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}} < 0\right) \wedge \left(\theta(\boldsymbol{x}_k) \leq \theta^{min}\right) \\ \min\left\{\gamma_\theta, \frac{\gamma_\phi \theta(\boldsymbol{x}_k)}{-\nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}}}\right\} & \left(\nabla \phi_{\mu_j}(\boldsymbol{x}_k)^T \boldsymbol{d}_k^{\boldsymbol{x}} < 0\right) \wedge \left(\theta(\boldsymbol{x}_k) > \theta^{min}\right) \\ \gamma_\theta & otherwise, \end{cases} \tag{1.144}$$

for some safety factor $\gamma_\alpha \in (0, 1]$, the algorithm reverts to a feasibility restoration

phase. For details on the feasibility restoration phase, please refer to section 3.3 in Wächter and Biegler (2006).

If a trial step $\alpha_{k,l}$ is rejected for some iteration l in the backtracking procedure and $\theta(\boldsymbol{x}_k(\alpha_{k,l})) \geq \theta(\boldsymbol{x}_k)$, before moving on to the next trial, an SOC step is performed. The SOC step aims to reduce the infeasibility by applying an additional Newton-type step for the constraints at the point $\boldsymbol{x}_k + \tilde{\boldsymbol{d}}_k^{\boldsymbol{x}}$ where $\tilde{\boldsymbol{d}}_k^{\boldsymbol{x}} = \alpha_{k,l}\boldsymbol{d}_k^{\boldsymbol{x}}$ using the Jacobian $\boldsymbol{A}_k^T$ at $\boldsymbol{x}_k$ to obtain a direction update $\boldsymbol{d}_k^{\boldsymbol{x},soc}$ by solving:

$$\boldsymbol{A}_k^T \boldsymbol{d}_k^{\boldsymbol{x},soc} + \boldsymbol{c}(\boldsymbol{x}_k + \alpha_{k,l}\boldsymbol{d}_k^{\boldsymbol{x}}) = \boldsymbol{0} \tag{1.145}$$

The new corrected search direction is then obtained from:

$$\boldsymbol{d}_k^{\boldsymbol{x},cor} = \alpha_{k,l}\boldsymbol{d}_k^{\boldsymbol{x}} + \boldsymbol{d}_k^{\boldsymbol{x},soc} \tag{1.146}$$

The details of efficiently and carefully calculating $\boldsymbol{d}_k^{\boldsymbol{x},soc}$ can be found in section 2.4 in Wächter and Biegler (2006). Once the corrected search direction $\boldsymbol{d}_k^{\boldsymbol{x},cor}$ has been computed, we again apply the fraction-to-the-boundary rule:

$$\alpha_k^{soc} := \max\left\{\alpha \in (0,1] : \left(\tau\boldsymbol{l} + (1-\tau_j)\boldsymbol{x}_k\right) \leq \left(\boldsymbol{x}_k + \alpha\boldsymbol{d}_k^{\boldsymbol{x},cor}\right) \leq \left(\tau\boldsymbol{u} + (1-\tau_j)\boldsymbol{x}_k\right)\right\} \tag{1.147}$$

and check if the resulting trial point:

$$\boldsymbol{x}_k^{soc} := \boldsymbol{x}_k + \alpha_k^{soc}\boldsymbol{d}_k^{\boldsymbol{x},cor} \tag{1.148}$$

is acceptable to the filter and satisfies the acceptance criteria set earlier replacing $\boldsymbol{x}(\alpha_{k,l})$ with $\boldsymbol{x}_k^{soc}$ while keeping $\boldsymbol{d}_k^{\boldsymbol{x}}$ as-is. If the new iterate is still rejected, the correction step is repeated (replacing $\alpha_{k,l}$ with α_k^{soc}, and $\boldsymbol{d}_k^{\boldsymbol{x}}$ with $\boldsymbol{d}_k^{\boldsymbol{x},cor}$ in the SOC step) unless the correction step has not decreased the constraint violation by a fraction $\kappa_{soc} \in (0, 1)$ or a maximum number p^{max} of SOC steps has been performed. In that case, the original search direction $\boldsymbol{d}_k^{\boldsymbol{x}}$ is reverted to and the regular backtracking line search is resumed with a shorter step size.

When multiple trial steps are rejected because the filter set is too strict or the progress made in the objective or constraint violation is insufficient, 2 accelerating heuristics are further employed to relax the acceptance criteria under some conditions. More on this can be found in section 3.2 in Wächter and Biegler (2006).

Finally after taking a step successfully, each variable z_{l_i} in $\boldsymbol{z_l}$ and each variable z_{u_i} in $\boldsymbol{z_u}$ then get clamped to an interval to avoid their extreme deviation:

$$z_{l_i} \in \begin{cases} \left[\frac{\mu_j}{\kappa_\Sigma(x_i - l_i)}, \frac{\kappa_\Sigma \mu_j}{(x_i - l_i)}\right] & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \left[\frac{(1-\kappa_d)\mu_j}{\kappa_\Sigma(x_i - l_i)}, \frac{\kappa_\Sigma(1-\kappa_d)\mu_j}{x_i - l_i}\right] & i \in \mathcal{I}_{\boldsymbol{l}} \setminus \mathcal{I}_{\boldsymbol{u}} \\ [0, 0] & otherwise \end{cases} \tag{1.149}$$

$$z_{u_i} \in \begin{cases} \left[\frac{\mu_j}{\kappa_\Sigma(u_i - x_i)}, \frac{\kappa_\Sigma \mu_j}{u_i - x_i}\right] & i \in \mathcal{I}_{\boldsymbol{l}} \cap \mathcal{I}_{\boldsymbol{u}} \\ \left[\frac{(1-\kappa_d)\mu_j}{\kappa_\Sigma(u_i - x_i)}, \frac{\kappa_\Sigma(1-\kappa_d)\mu_j}{u_i - x_i}\right] & i \in \mathcal{I}_{\boldsymbol{u}} \setminus \mathcal{I}_{\boldsymbol{l}} \\ [0, 0] & otherwise \end{cases} \tag{1.150}$$

This clamping step creates a safeguard needed for the global convergence proof of

the algorithm (Wächter and Biegler, 2006). A large value for $\kappa_{\Sigma} = 10^{10}$ is used to only minimally interfere with the Newton or quasi-Newton update.

1.6.6 Augmented Lagrangian algorithm

Both the MMA and IPOPT algorithms discussed previously require the full Jacobian of the constraints. This can be computationally prohibitive in some applications. In some cases, computing the Jacobian is much more expensive than calling the operator $\boldsymbol{v} \rightarrow \boldsymbol{J}'\boldsymbol{v}$ where $\boldsymbol{J}$ is the Jacobian of the constraints and $\boldsymbol{v}$ is a vector. The augmented Lagrangian algorithm only requires this operator and not the full Jacobian matrix.

Consider the following inequality constrained optimization problem:

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} && f_0(\boldsymbol{x}) \\ &\text{subject to} && \\ &f_i(\boldsymbol{x}) \leq 0 && \forall i = 1..I, \\ &f_i(\boldsymbol{x}) = 0 && \forall i = I+1..I+E, \\ &l_j \leq x_j \leq u_j && \forall j = 1..V \end{aligned} \tag{1.151}$$

where I is the number of inequality constraints, E is the number of equality constraints and V is the number of decision variables, f_i is a potentially nonlinear scalar-valued function of $\boldsymbol{x}$ and $\boldsymbol{l}$ and $\boldsymbol{u}$ are finite vectors of lower and upper bounds on the decision vector $\boldsymbol{x}$ respectively.

The augmented Lagrangian algorithm (AugLag) is not a single algorithm but a

family of algorithms in which the main idea is to reduce the above minimization problem to the following max-min formulation:

$$\underset{\boldsymbol{\lambda} \in \boldsymbol{Y}_{\boldsymbol{\lambda}}}{\text{maximize}} \quad \min_{\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}} \left\{ \mathcal{L}_c(\boldsymbol{x}, \boldsymbol{\lambda}) = f_0(\boldsymbol{x}) + \sum_{i=1}^{I+E} \lambda_i f_i(\boldsymbol{x}) + c \sum_{i=1}^{I} \max\{f_i(\boldsymbol{x}), 0\}^2 + c \sum_{i=I+1}^{I+E} |f_i(\boldsymbol{x})|^2 \right\} \tag{1.152}$$

for some constant $c \geq 0$, where $\boldsymbol{Y}_{\boldsymbol{\lambda}} = \{\boldsymbol{\lambda} : \lambda_i \geq 0 \forall i \in [1, I], \lambda_i \in \mathbb{R} \forall i \in [I+1, I+E]\}$ and $\mathcal{L}_c$ is known as the augmented Lagrangian function. Assume all the functions used are continuous and twice differentiable.

Note how the inequality and equality constraints were relaxed and added to the objective but the bounds constraints were not. That is the inner minimization still needs to respect the bounds constraints. The choice of which constraints to relax and which to handle directly, as well as which optimizer to use for the inner optimization problem and which one to use for the outer problem can lead to different variants of the augmented Lagrangian family of algorithms. The optimal value of the augmented Lagrangian function subject to some relaxed constraints is known to be always less than or equal to the optimal value of the original optimization problem. It is also well known that the outer maximization problem is always concave (i.e. tractable) even if the objective and constraint functions were non-convex (Bertsekas, 1996).

It is well known (Bertsekas, 1996) that if:

1. The original problem has an isolated set of local minima $\boldsymbol{X}^*$ which is compact,

2. The constant c is increased in a sequence $\{c_k\}$ such that $0 < c_k < c_{k+1}$, and

3. The Lagrangian multipliers $\boldsymbol{\lambda}$ follow an arbitrary bounded sequence $\{\boldsymbol{\lambda}_k\}$ where $\lambda_{k,i}$ is the i^{th} element of $\boldsymbol{\lambda}_k$

then there exists a sub-sequence of points $\{\boldsymbol{x}_k\}_K$ converging to a point $\boldsymbol{x}^* \in \boldsymbol{X}^*$ such that $\boldsymbol{x_k}$ is a local minimum for the following augmented Lagrangian sub-problem:

$$\underset{\boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}}{\text{minimize}} \quad \left\{\mathcal{L}_{c_k}(\boldsymbol{x}, \boldsymbol{\lambda}_k) = f_0(\boldsymbol{x}) + \sum_{i=1}^{I+E} \lambda_{k,i} f_i(\boldsymbol{x}) + c_k \sum_{i=1}^{I} \max\{f_i(\boldsymbol{x}), 0\}^2 + c_k \sum_{i=I+1}^{I+E} f_i(\boldsymbol{x})^2\right\} \tag{1.153}$$

If additionally each $\boldsymbol{x}_k$ is a global minimum of the augmented Lagrangian sub-problem with c_k and $\boldsymbol{\lambda}_k$, and the feasible set is compact, e.g. $\boldsymbol{l}$ and $\boldsymbol{u}$ are finite, then every limit point of the sequence $\{\boldsymbol{x}_k\}$ is a global minimum of the original problem. Note that it is possible for the original problem to have a (unique) global minimum but the augmented Lagrangian sub-problem is unbounded from below. This is a weakness of the AugLag algorithm that needs to be addressed in implementation. However if the feasible domain is compact, this weakness is eliminated.

If additionally the sub-sequence $\{\boldsymbol{x}_k\}_K$ converges to a point $\boldsymbol{x}^* \in \boldsymbol{X}^*$ that satisfies one of the constraint qualification conditions, then the sequence $\{\text{proj}_{Y_\lambda}(\boldsymbol{\lambda}_k + c_k \boldsymbol{G}_k)\}$ converges to a point $\boldsymbol{\lambda}^*$ such that $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ satisfy the first order sufficient KKT optimality conditions, where $\boldsymbol{G}_k$ is a vector of length $I + E$ and whose i^{th} element is $f_i(\boldsymbol{x}_k)$ (Bertsekas, 1996). Another weakness of the AugLag algorithm is that the sub-problem's optimizer may converge to a point that doesn't satisfy any of the constraint qualification conditions for the original problem. One case where this is guaranteed to happen is if the original problem is infeasible so the quadratic term dominates the augmented Lagrangian function as $c \to \infty$ and every (approximate) KKT point of

the sub-problem must (approximately) violate the constraint qualification conditions of the original problem.

Instead of choosing $\boldsymbol{\lambda}_k$ arbitrarily, if $\boldsymbol{\lambda}_{k+1}$ is chosen as:

$$\boldsymbol{\lambda}_{k+1} = \text{proj}_{Y_\lambda}(\boldsymbol{\lambda}_k + \alpha_k \boldsymbol{G}_k) \tag{1.154}$$

where $\alpha_k = c_k$, convergence can often be significantly accelerated and the algorithm typically converges at lower values of c_k. The dual update step size α_k can also be improved to a more "optimal" choice than c_k or a Newton/quasi-Newton direction can be used instead of $\boldsymbol{G}_k$. These can lead to faster convergence but come at an increased computational cost. For more on the augmented Lagrangian algorithm, see Bertsekas (1996).

In the context of topology optimization, one challenge with AugLag is that the Hessian of the augmented Lagrangian function becomes rather ill-conditioned for large values of c where the condition number of the Hessian matrix goes to ∞ as $c \to \infty$. Given that computing the Hessian of the augmented Lagrangian in topology optimization is generally intractable, one must rely on first order methods to optimize the sub-problem. However even quasi-Newton methods may very well encounter difficulty converging if the Hessian approximation is not good enough and/or the starting point is not near a solution. Adding to this the scaling issues that are commonly found in topology optimization and this makes AugLag extremely difficult to fine-tune in practice when using it in topology optimization. The convergence speed and numerical stability of AugLag can be extremely sensitive to

the initial value of c, its increment rate, the initial solution, and the sub-problem's optimizer's terminating conditions.

The main computational advantage of AugLag is that it can handle block constraints more efficiently. Let $\boldsymbol{G}(\boldsymbol{x})$ be the vector-valued function whose components are the constraint functions $f_i(\boldsymbol{x})\ \forall i \in [1, I+E]$ and let $\boldsymbol{M}(\boldsymbol{x}) = \text{proj}_{\boldsymbol{Y}_\lambda}\boldsymbol{G}(\boldsymbol{x})$ be the projection of $\boldsymbol{G}(\boldsymbol{x})$ on $\boldsymbol{Y}_\lambda$. The augmented Lagrangian formulation can therefore be written as:

$$\underset{\boldsymbol{\lambda} \in \boldsymbol{Y}_\lambda}{\text{maximize}} \quad \min_{\boldsymbol{l} \le \boldsymbol{x} \le \boldsymbol{u}} \{\mathcal{L}_c(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + (\boldsymbol{\lambda} + c\boldsymbol{M}(\boldsymbol{x}))^T \boldsymbol{G}(\boldsymbol{x})\} \tag{1.155}$$

Note that the gradient of $\mathcal{L}_c(\boldsymbol{x}, \boldsymbol{\lambda})$ wrt $\boldsymbol{x}$, $\nabla_{\boldsymbol{x}}\mathcal{L}_c(\boldsymbol{x}, \boldsymbol{\lambda})$, can be written in terms of the gradient of $f(\boldsymbol{x})$, $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$, and the Jacobian of $\boldsymbol{G}(\boldsymbol{x})$, $\nabla_{\boldsymbol{x}}\boldsymbol{G}(\boldsymbol{x})$, as:

$$\nabla_{\boldsymbol{x}}\mathcal{L}_c(\boldsymbol{x}, \boldsymbol{\lambda}) = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla_{\boldsymbol{x}}\boldsymbol{G}(\boldsymbol{x})^T(\boldsymbol{\lambda} + 2c\boldsymbol{M}(\boldsymbol{x})) \tag{1.156}$$

To compute the above gradient, the full Jacobian of the constraints need not be built. Only the operator $\boldsymbol{w} \rightarrow \nabla_{\boldsymbol{x}}\boldsymbol{G}(\boldsymbol{x})^T\boldsymbol{w}$ needs to be defined. For the purposes of this paper, the term *block constraint* will be used to refer to constraints on functions $\boldsymbol{G}(\boldsymbol{x})$ where the operator $\boldsymbol{w} \rightarrow \nabla_{\boldsymbol{x}}\boldsymbol{G}(\boldsymbol{x})^T\boldsymbol{w}$ can be defined more efficiently than simply calculating the entire Jacobian, $\nabla_{\boldsymbol{x}}\boldsymbol{G}(\boldsymbol{x})$, and then multiplying its transpose by $\boldsymbol{w}$.

1.7 Topology optimization under uncertainty

Every topology optimization problem has some input data such as:

1. The shape of the base design from which material may be removed,

2. The load applied on the design, or

3. The material properties such as the Young's modulus or Poisson ratio

These data are usually fixed or can be described in terms of other fixed parameters. However, sometimes the optimal solution of the topology optimization problem can be rather sensitive to the values of these data, such that a small change in the data can cause a significant change in the objective or render the optimal solution obtained infeasible. Treating uncertainty in the optimization problem's data is therefore of utmost importance, and indeed it has been studied for a long time in optimization literature in general as well as in design and topology optimization literature.

Robust optimization (RO), stochastic optimization (SO), risk-averse optimization (RAO) and reliability-based design optimization (RBDO) are some of the terms used in literature to describe a plethora of techniques for handling uncertainty in the data of an optimization problem.

1.7.1 Robust optimization

RO defines the data as a set, resembling the uncertainty, such that the objective and/or constraints are defined over the whole set (Bertsimas et al., 2011), e.g. a

coefficient can be allowed to be in the set $[0.9, 1.1]$ instead of exactly 1. The set can be continuous, discrete or a mixed set. In the case of loading uncertainty, this can be an interval set for the load magnitude or angle, a hyper-ellipsoid set for multiple force components, a discrete set of finite loading scenarios, or any other set. The objective is typically either:

1. Minimizing or maximizing a function with no uncertainty in its data, or
2. Minimizing the maximum or maximizing the minimum of a function over the set of different values that the problem's data can take.

The constraint functions are point-based functions that must lie in the feasible domain for every data point in the uncertainty set. Note that when the objective is minimizing the maximum, it can be converted to an equivalent problem with another robust constraint and a single-variable objective function as such:

$$\underset{\boldsymbol{x}, c}{\text{minimize}} \quad c \tag{1.157a}$$

subject to

$$g(\boldsymbol{x}; \boldsymbol{f}) - c \leq 0 \quad \forall \boldsymbol{f} \in U \tag{1.157b}$$

where $\boldsymbol{f}$ is the load vector, $\boldsymbol{x}$ is the topology design variables, c is the objective and U is the set of all possible load scenarios. The same can be done for maximizing the minimum. There is therefore no loss of generality in assuming that the objective is certain, and only the constraints' data is uncertain. For more on RO, the readers are referred to Bertsimas et al. (2011) and Aharon Ben-Tal et al. (2009).

1.7.2 Stochastic and risk-averse optimization

SO and RAO assume that the data follows a known probability distribution (Shapiro et al., 2009; Choi et al., 2007). Let $\boldsymbol{f}$ be a random load and $\boldsymbol{x}$ be the topology design variables. A probabilistic constraint can be defined as:

$$P(g(\boldsymbol{x}; \boldsymbol{f}) \leq 0) \geq \eta \tag{1.158}$$

where $\boldsymbol{f}$ follows a known probability distribution. This constraint is often called a chance constraint or a reliability constraint in RBDO. Note that the minimization of the η-percentile can be modeled using a chance constraint and a deterministic objective as follows:

$$\underset{\boldsymbol{x}, \gamma}{\text{minimize}} \quad \gamma \tag{1.159a}$$

subject to

$$P(g(\boldsymbol{x}; \boldsymbol{f}) - \gamma \leq 0) \geq \eta \tag{1.159b}$$

The objective of an SO problem is typically either deterministic or some probabilistic function such as the mean of a function of the random variable, its variance, standard deviation or a weighted sum of such terms. The biggest challenge of the probabilistic approach is evaluating and differentiating the probabilistic function to use in various optimization algorithms. SO has its roots in optimization theory and operations research where a family of approaches known as chance-constrained optimization

were developed to tractably approximate the probabilistic constraint.

RAO can be considered a sub-field of SO borrowing concepts from risk analysis in mathematical economics to define various risk measures and tractable approximations to be used in objectives and/or constraints in SO. One such risk measure is the conditional value-at-risk (CVaR). Let $z = g(\boldsymbol{x}; \boldsymbol{f})$. The value-at-risk (VaR) is defined as:

$$\mathbf{VaR}(z; \eta) = \inf\{\gamma | \boldsymbol{P}(z \leq \gamma) \geq \eta\} \tag{1.160}$$

The probabilistic/chance/reliability constraint $P(g(\boldsymbol{x}; \boldsymbol{f}) \leq 0) \geq \eta$ is the same as $\mathbf{VaR}(z; \eta) \leq 0$. One common risk measure used in RAO to approximate VaR is the conditional value-at-risk (CVaR) defined as:

$$\mathbf{CVaR}(z; \eta) = \inf_{\beta}(\beta + 1/(1 - \eta)\boldsymbol{E}(z - \beta)_+) \tag{1.161}$$

It is known that $\mathbf{CVaR}(z; \eta) \geq \mathbf{VaR}(z; \eta)$. When $\boldsymbol{f}$ comes from a continuous distribution, an interpretation of $\mathbf{CVaR}(z; \eta)$ is as the expected shortfall, also known as the conditional expectation, $\boldsymbol{E}(z | z \geq \beta^*)$, where $\beta^* = \mathbf{VaR}(z; \eta)$. Other more traditional risk measures include the weighted sum of mean and variance of a function or the weighted sum of the mean and standard deviation. For more on SO and RAO, the reader is referred to Shapiro et al. (2009).

A special case of SO problems is when the objective is defined as the mean of a function, its variance, standard deviation and/or a weighted sum of them over a

deterministic set of data scenarios. While not strictly probabilistic, a probabilistic view would be to assume a uniform distribution over the domain of the set. This can therefore be considered a special-case SO problem. More generally, robust constraints can be viewed as probabilistic constraints with a uniform distribution over the set's domain and $\eta = 1$. This view is not very common in practice though since RO problems typically have more efficient algorithms than those of SO problems with arbitrary distributions.

1.7.3 Reliability-based design optimization

RBDO and its ancestor, reliability analysis, are more common in the sizing optimization literature. Classically, RBDO has been about solving optimization problems with a probabilistic constraint, called the reliability constraint, much like SO. One of the most common RBDO techniques used in topology optimization literature is the first-order reliability method (FORM). In FORM, the random variable $\boldsymbol{f}$ is assumed to be a function of a multivariate unit Gaussian random variable $\boldsymbol{u}$ whose mean is $\mathbf{0}$ and covariance is the identity matrix. Let $G(\boldsymbol{x};\boldsymbol{u}) = g(\boldsymbol{x};\boldsymbol{f}(\boldsymbol{u}))$. The reliability constraint can therefore be written as: $P(G(\boldsymbol{x};\boldsymbol{u}) \leq 0) \geq \eta$ or $P(G(\boldsymbol{x};\boldsymbol{u}) > 0) \leq 1 - \eta$ for some $\eta \geq 0.5$, where $P(G(\boldsymbol{x};\boldsymbol{u}) > 0)$ is known as the probability of failure P_f. The function $G(\boldsymbol{x};\boldsymbol{u})$, also known as the limit state function, is then linearly approximated with respect to $\boldsymbol{u}$ using Taylor series expansion around a point $\boldsymbol{u}_0$:

$$\tilde{G}(\boldsymbol{x};\boldsymbol{u}) = G(\boldsymbol{x};\boldsymbol{u}_0) + \nabla_{\boldsymbol{u}} G(\boldsymbol{x};\boldsymbol{u}_0)^T (\boldsymbol{u} - \boldsymbol{u}_0) \tag{1.162}$$

The function $\tilde{G}$ is then assumed to follow a normal distribution and its first and second moments are computed using the expression above. This approximation approach is known as the first-order second-moment (FOSM) approach. The choice of the linearization point $\boldsymbol{u}_0$ is known to affect the accuracy of FOSM, where the mean $\mathbf{0}$ is typically outperformed by the less obvious alternative known as the most probable point (MPP) $\boldsymbol{u}^*$. There are two ways to define the MPP point: the reliability index approach (RIA) (Yu et al., 1998; Tu et al., 1999) and the performance measure approach (PMA) (Tu et al., 1999).

RIA defines $\boldsymbol{u}^*$ as the optimal solution of the reliability analysis optimization problem:

$$\underset{\boldsymbol{u}}{\text{minimize}} \quad ||\boldsymbol{u}|| \tag{1.163a}$$

subject to

$$G(\boldsymbol{x}; \boldsymbol{u}) = 0 \tag{1.163b}$$

at the current design $\boldsymbol{x}$. The so-called reliability index β is then defined as: $\beta = ||\boldsymbol{u}^*||$. If $\boldsymbol{u}^*$ is a Karush-Kuhn-Tucker (KKT) point of the above problem, given the normality assumption on $\tilde{G}$, the reliability index $\beta = ||\boldsymbol{u}^*|| = \frac{G(\boldsymbol{x};\boldsymbol{u}^*)-\mu_{\tilde{G}}(\boldsymbol{x})}{\sigma_{\tilde{G}}(\boldsymbol{x})} = -\frac{\mu_{\tilde{G}}(\boldsymbol{x})}{\sigma_{\tilde{G}}(\boldsymbol{x})}$, where $\mu_{\tilde{G}}$ is the mean of $\tilde{G}$ and $\sigma_{\tilde{G}}$ is its standard deviation. The probability of failure is therefore given by $P_f = \Phi(-\beta)$, where Φ is the cumulative distribution function (CDF) of the standard normal distribution with mean 0 and unit variance. Since Φ^{-1} is a monotonically increasing function, the reliability constraint can be

re-written as:

$$P(G(\boldsymbol{x};\boldsymbol{u}) > 0) = P_f = \Phi(-\beta) \leq 1 - \eta \tag{1.164}$$

$$\beta \geq -\Phi^{-1}(1-\eta) \tag{1.165}$$

$$\beta \geq \Phi^{-1}(\eta) \tag{1.166}$$

The gradient of β with respect to $\boldsymbol{x}$ can be efficiently computed using the KKT optimality conditions of the reliability analysis optimization problem. $\Phi^{-1}(\eta)$ is sometimes called the target reliability index β_t.

PMA on the other hand uses a different definition for the MPP point $\boldsymbol{u}^*$. Let the function $F_{\tilde{G}}(t;\boldsymbol{x}) = \int_{\tilde{G}(\boldsymbol{x};\boldsymbol{u})\leq t} f_{\boldsymbol{u}} d\boldsymbol{u}$, where $f_{\boldsymbol{u}}$ is the probability density function (PDF) of the random variable $\boldsymbol{u}$. Under the normality assumption of $\tilde{G}$, $F_{\tilde{G}}(t;\boldsymbol{x}) = \Phi(\frac{t-\mu_{\tilde{G}}(\boldsymbol{x})}{\sigma_{\tilde{G}}(\boldsymbol{x})})$ is monotonically increasing in t and invertible. The approximate reliability constraint $P(\tilde{G}(\boldsymbol{x};\boldsymbol{u}) \leq 0) \geq \eta$ can therefore be written as:

$$F_{\tilde{G}}(0;\boldsymbol{x}) \geq \eta \tag{1.167}$$

$$F^{-1}_{\tilde{G}}(\eta;\boldsymbol{x}) \leq 0 \tag{1.168}$$

$$\mu_{\tilde{G}}(\boldsymbol{x}) + \Phi^{-1}(\eta)\sigma_{\tilde{G}}(\boldsymbol{x}) \leq 0 \tag{1.169}$$

Let $G_p(\boldsymbol{x}) = F^{-1}_{\tilde{G}}(\eta;\boldsymbol{x}) = \mu_{\tilde{G}}(\boldsymbol{x}) + \Phi^{-1}(\eta)\sigma_{\tilde{G}}(\boldsymbol{x})$. $G_p(\boldsymbol{x})$ is known as the performance measure. Taking the linearization point $\boldsymbol{u}_0$ to be the optimal KKT solution

$\boldsymbol{u}^*$ of the following so-called inverse reliability optimization problem:

$$\underset{\boldsymbol{u}}{\text{minimize}} \quad G(\boldsymbol{x};\boldsymbol{u}) \tag{1.170a}$$

subject to

$$||\boldsymbol{u}|| = \Phi^{-1}(\eta) \tag{1.170b}$$

at the current design $\boldsymbol{x}$, $||\boldsymbol{u}^*|| = \frac{G(\boldsymbol{x};\boldsymbol{u}^*)-\mu_{\tilde{G}}(\boldsymbol{x})}{\sigma_{\tilde{G}}(\boldsymbol{x})} = \Phi^{-1}(\eta)$. Substituting for $\Phi^{-1}(\eta)$ in $F_{\tilde{G}}^{-1}(\eta;\boldsymbol{x})$, $G_p(\boldsymbol{x}) = G(\boldsymbol{x};\boldsymbol{u}^*)$. The gradient of $G_p(\boldsymbol{x})$ is therefore equal to the partial of $G(\boldsymbol{x};\boldsymbol{u}^*)$ with respect to $\boldsymbol{x}$. One of the advantages of PMA is that efficient algorithms have been developed to solve the inverse reliability optimization problem which is just optimizing an objective over the surface of a sphere.

A second less common reliability method is known as the second order reliability method (SORM) which makes use of a second order Taylor series expansion of the function G with respect to $\boldsymbol{u}$ before assuming the approximation follows a Gaussian distribution. This approximation approach is known as the second order, second moment (SOSM) approach. SORM is less common than FORM though because of the computational cost of computing the second order term in the Taylor series expansion while generally adding little to the accuracy according to Choi et al. (2007).

Both the RIA and PMA approaches essentially turn the stochastic optimization problem with reliability constraint to a deterministic bi-level optimization problem. Some RBDO algorithms solve the optimization problems from the two levels sequentially, i.e. not nested, in order to reduce the computational cost of the algo-

rithms. For more on RBDO and reliability analysis, the reader is referred to Choi et al. (2007) and Youn and Choi (2004). It is interesting to note that while classic RBDO has been about handling probabilistic reliability constraints, more recently the non-probabilistic RBDO (NRBDO) was developed, applying similar techniques as in classic RBDO but for handling set-based, non-probabilistic uncertainty to solve RO problems (Luo et al., 2009; Kang and Luo, 2009; Guo and Lu, 2015; Zheng et al., 2018; Wang, Xia, Zhang and Lv, 2019; Wang, Liu, Yang and Hu, 2019).

1.7.4 Relationship between uncertainty paradigms

There are ways to (approximately) convert between the continuous set, discrete set and probabilistic representations of data uncertainty in constraints. For instance, a constraint generation approach and a separation oracle can be used to represent the continuous set using a finite discrete set of constraint violating scenarios (Aharon Ben-Tal et al., 2009). This approach is also known as the *anti-optimization* method in design optimization literature (Elishakoff et al., 1994; Lombardi and Haftka, 1998). Monte Carlo sampling can be used to conservatively approximate the probabilistic constraint using a robust constraint on a discrete set of scenarios, such that satisfying the robust constraint guarantees satisfying the probabilistic constraint (Tempo et al., 1996). Probabilistic inference methods such as maximum likelihood estimation and maximum a-posteriori estimation can be used to infer the underlying distribution from which a discrete set of data scenarios could have been generated (Bishop, 2006). This can be combined with any unsupervised machine learning model for

dimensionality reduction such as principal component analysis (PCA). The exact convex hull of a discrete set of scenarios can be obtained using computational geometry in 2- and 3- dimensional spaces (de Berg et al., 2008) with approximation algorithms in higher dimensions (Sartipizadeh and Vincent, 2016). Finally, the continuous set robust counterpart of a chance constraint can sometimes be formulated such that satisfying the robust constraint guarantees satisfying the probabilistic constraint (Aharon Ben-Tal et al., 2009).

Note that in topology optimization literature, the term "*robust topology optimization*" is often used to refer to minimizing the weighted sum of the mean, and variance or standard deviation of a function subject to probabilistic uncertainty (Dunning and Kim, 2013; Zhao and Wang, 2014*b*; Cuellar et al., 2018). However, this use of the term "*robust*" is not consistent with the standard definition of RO in optimization theory literature, e.g. Aharon Ben-Tal et al. (2009). The more compliant term is *stochastic topology optimization* or *risk-averse topology optimization*.

1.8 Linear algebra

1.8.1 Solving a linear system

When performing FEA, often a linear system solve of the form $\boldsymbol{u} = \boldsymbol{K} \setminus \boldsymbol{f}$ needs to be performed where $\boldsymbol{K}$ is a sparse positive definite and $\boldsymbol{f}$ is a dense or sparse vector. Let $\boldsymbol{K}$ be a sparse matrix of size $n \times n$. The number of non-zeros in $\boldsymbol{K}$ is typically $O(n)$. The inverse of an arbitrary sparse matrix is generally a dense matrix which requires $O(n^2)$ memory. It is therefore not recommended to attempt to invert $\boldsymbol{K}$ especially that we only need to solve the linear system and don't care about the inverse of $\boldsymbol{K}$. One can solve the linear system above by decomposition $\boldsymbol{K}$ into $\boldsymbol{L}\boldsymbol{L}^T$ where $\boldsymbol{L}$ is a lower triangular matrix. This factorization is known as Cholesky factorization (Golub and Loan, 1996). The linear system solve can therefore be written as:

$$\boldsymbol{u} = (\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{f} \tag{1.171}$$

$$= \boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{f} \tag{1.172}$$

Solving the linear system then boils down to solving the 2 triangular systems below:

$$\boldsymbol{y} = \boldsymbol{L} \setminus \boldsymbol{f} \tag{1.173}$$

$$\boldsymbol{u} = \boldsymbol{L}^T \setminus \boldsymbol{y} \tag{1.174}$$

If $\boldsymbol{K}$ is dense, the Cholesky factorization requires approximately $n^3/3$ floating point operations (flops) (Golub and Loan, 1996). The triangular systems can then be solved efficiently using Gaussian elimination which has $O(n^2)$ time complexity. However, for sparse $\boldsymbol{K}$ the main advantage of the Cholesky factorization comes from the lower fill-in rate in the sparse Cholesky factor $\boldsymbol{L}$ which tends to be more sparse than the inverse. Therefore, the factorization and Gaussian eliminations, although still having a worst case time complexity of $O(n^3)$ and worst case space complexity of $O(n^2)$, tend to be more efficient in practice. In practice also the $O(n^2)$ memory tends to be the bottleneck and not the $O(n^3)$ time. In other words, we usually run out of RAM before the algorithm becomes too slow to use.

If the so-called "direct" solver based on Cholesky factorization becomes computationally prohibitive either in memory or time, one can use iterative solvers instead, e.g. the conjugate gradient (CG) algorithm (Golub and Loan, 1996). CG only requires the multiplication operation $\boldsymbol{v} \rightarrow \boldsymbol{K}\boldsymbol{v}$ to compute the solution to the linear system. The multiplication operator requires only $O(n)$ flops. However the convergence rate of CG tends to be very low if the condition number of $\boldsymbol{K}$ is large. In topology optimization, the global stiffness matrix assembled from the element stiffness matrices and pseudo-densities can be extremely ill-conditioned. Therefore, a good preconditioner is necessary to achieve reasonable convergence speed.

1.8.2 Eigenvalue decomposition

Let $\boldsymbol{A}$ be a square matrix. A normal vector $\boldsymbol{v} : ||\boldsymbol{v}||_2 = 1$ is called an eigenvector of $\boldsymbol{A}$ if:

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v} \tag{1.175}$$

$$(\boldsymbol{A} - \lambda \boldsymbol{I})\boldsymbol{v} = \boldsymbol{0} \tag{1.176}$$

for some number λ and λ is called its eigenvalue. A matrix can have multiple eigenvectors with equal or different eigenvalues.

If $\boldsymbol{A}$ is Hermitian (or real and symmetric), all its eigenvalues will be real numbers. The so-called Rayleigh quotient is then given by:

$$R(\boldsymbol{A}, \boldsymbol{x}) = \frac{\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \tag{1.177}$$

which can be shown to be always bounded between:

$$\lambda_{min} = R(\boldsymbol{A}, \boldsymbol{v}_{min}) \leq R(\boldsymbol{A}, \boldsymbol{x}) \leq R(\boldsymbol{A}, \boldsymbol{v}_{max}) = \lambda_{max} \tag{1.178}$$

$\forall \boldsymbol{x}$ where λ_{min} and λ_{max} are the minimum and maximum eigenvalues of $\boldsymbol{A}$ respectively and $\boldsymbol{v}_{min}$ and $\boldsymbol{v}_{max}$ are the associated eigenvectors.

A square matrix $\boldsymbol{A}$ is called diagonalizable if it is similar to a diagonal matrix

$\boldsymbol{D}$. Two matrices A and $\boldsymbol{D}$ are similar if there exists an invertible matrix $\boldsymbol{P}$ such that:

$$\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^{-1} \tag{1.179}$$

If additionally the L-2 norm of each column of $\boldsymbol{P}$ is 1, this decomposition is called an eigenvalue decomposition, where the diagonal of $\boldsymbol{D}$ will be the vector of $\boldsymbol{A}$'s eigenvalues and the columns of $\boldsymbol{P}$ are the associated eigenvectors (one for each eigenvalue). When $\boldsymbol{A}$ is real and symmetric, $\boldsymbol{P}^{-1} = \boldsymbol{P}^T$, i.e. $\boldsymbol{P}$ is an orthonormal/unitary matrix, and $\boldsymbol{D}$ and $\boldsymbol{P}$ are guaranteed to be matrices of real numbers. A real symmetric matrix is always diagonalizable.

1.8.3 Singular value decomposition

The singular value decomposiiton (SVD) of an $m \times n$ matrix $\boldsymbol{F}$ is a factorization of the form:

$$\boldsymbol{F} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{1.180}$$

$$\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I} \tag{1.181}$$

$$\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I} \tag{1.182}$$

where $\boldsymbol{U}$ is an $m \times n$ unitary matrix, $\boldsymbol{S}$ is an $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal and $\boldsymbol{V}$ is an $n \times n$ unitary matrix. If $\boldsymbol{F}$ is real, $\boldsymbol{U}$ and $\boldsymbol{V}$ will also be real matrices. If $\boldsymbol{F}$ is a tall or wide matrix, often one would be interested in the so-called compact SVD. If $\boldsymbol{F}$ is tall (i.e. $m > n$) in the

compact SVD, $\boldsymbol{S}$ would be $n \times n$ and only the first n columns of $\boldsymbol{V}$ would be kept. If $\boldsymbol{F}$ is wide in the compact SVD, only the first m columns of $\boldsymbol{U}$ would be kept and $\boldsymbol{S}$ would be of size $m \times m$.

1.8.4 Trace estimation

Hutchinson (1990) developed an unbiased estimator for the trace of a square matrix $\boldsymbol{A}$. Hutchinson's trace estimator is given by:

$$tr(\boldsymbol{A}) = E_{\boldsymbol{v}}(\boldsymbol{v}^T \boldsymbol{A} \boldsymbol{v}) \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{v}_i^T \boldsymbol{A} \boldsymbol{v}_i \tag{1.183}$$

where $\boldsymbol{v}$ is a random vector with each element independently distributed with 0 mean and unit variance, $\boldsymbol{v}_i$ are samples of the random vector $\boldsymbol{v}$, also known as probing vectors, and N is the number of such probing vectors. One common distribution used for the elements of $\boldsymbol{v}$ is the Rademacher distribution which is a discrete distribution with support $\{-1, 1\}$ each of which has a probability of 0.5. Hutchinson proved that an estimator with the Rademacher distribution for $\boldsymbol{v}$ will have the least variance among all other distributions.

1.8.5 Diagonal estimation

Much like the trace, the diagonal of a square matrix $\boldsymbol{A}$ can be estimated. One diagonal estimator directly related to Hutchinson's trace estimator was proposed by

Bekas et al. (2007):

$$diag(\boldsymbol{A}) = E(\boldsymbol{D_v A v}) \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{D_{v_i} A v_i} \tag{1.184}$$

where $diag(\boldsymbol{A})$ is the diagonal of $\boldsymbol{A}$ as a vector, $\boldsymbol{D_v}$ is the diagonal matrix with a diagonal $\boldsymbol{v}$, $\boldsymbol{v}$ is a random vector distributed much like in Hutchinson's estimator, $\boldsymbol{v}_i$ are the probing vector instances of $\boldsymbol{v}$ and N is the number of probing vectors. The sum of the elements of the diagonal estimator above gives us Hutchinson's trace estimator. Bekas et. al. also showed that one can use use the deterministic basis of a Hadamard matrix as probing vectors $\boldsymbol{v}_i$ instead of random vectors and that this increases the accuracy of the diagonal estimator.

1.9 Scope and limitations of the study

In this thesis, a number of techniques to accelerate topology optimization algorithms are proposed.

1. In chapter 2, an accelerate technique for continuation SIMP is proposed whereby the penalty is adapted to reduce the number of sub-problems solved in a systematic way. The gains are demonstrated using various compliance minimization problems.

2. In chapter 3, the problem of loading uncertainty is presented and tackled using novel efficient exact methods. The computational time complexities of the proposed methods are shown to be strictly better than the naive straightforward approaches.

3. In chapter 4, approximate methods for stochastic and risk-averse compliance-based topology optimization problems are presented.

1.10 Significance of the study

One of the main obstacles to the wide adoption topology optimization in practice is the lack of computationally efficient, scalable algorithms that can efficiently handle a large number of elements as well as uncertainty in the data, e.g. loading conditions. Reducing the computational time to solve problems of a particular size means that larger problems can be solved for a given time. Practical design problems must also consider uncertainty in the loading conditions which is a further complexity that this thesis helps address.

2. Adaptive continuation SIMP

This work appears in the paper: "Adaptive continuation solid isotropic material with penalization for volume constrained compliance minimization", Computer Methods in Applied Mechanics and Engineering, Volume 363, 2020. (Tarek and Ray, 2020)

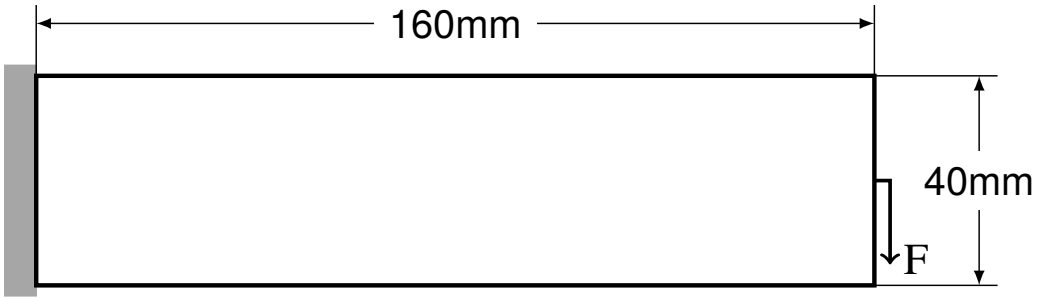


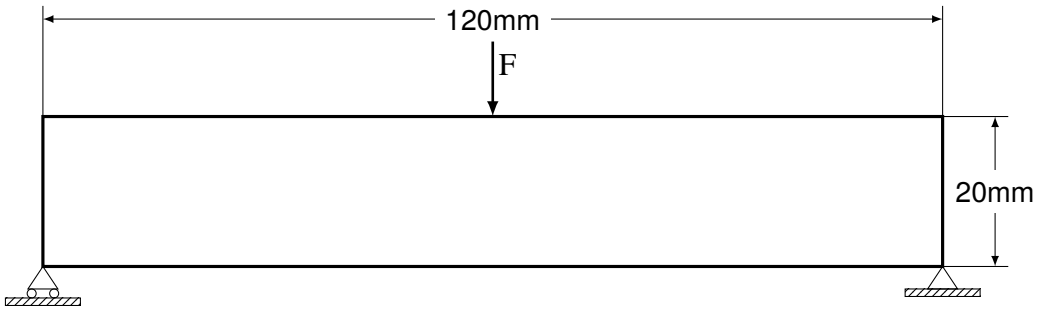
Figure 2-1: Cantilever beam problem before topology optimization.

2.1 Introduction

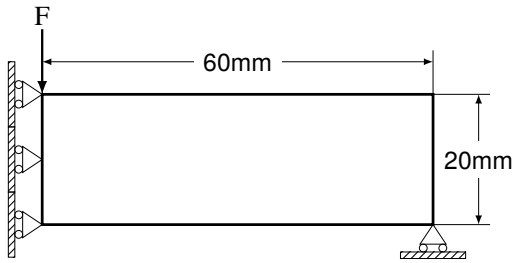
The main drawback of CSIMP compared to the single penalty SIMP is that it usually requires a large number of FEA simulations to converge. This is in contrast to the BESO algorithm in the VCCM class of problems which was shown to converge in much fewer FEA simulations to an equally good solution by Huang and Xie (2010) when solving the cantilever and Messerschmitt-Bolkow-Blohm (MBB) beam VCCM problems, shown in Figures 2-1 and 2-2 respectively. In MMA, each iteration requires the evaluation of the objective's and constraints' zero and first order information, i.e. function values and gradients. Obtaining these requires solving a finite element analysis (FEA) simulation which in the VCCM case solves the system $\boldsymbol{Ku} = \boldsymbol{f}$. The time each FEA simulation takes grows significantly with the size of the ground mesh so reducing the number of such simulations is crucial to improve the scalability of CSIMP.

In the rest of this chapter, the compliance will be defined as:

$$C = \boldsymbol{u}^T \boldsymbol{K} \boldsymbol{u} = \boldsymbol{f}^T \boldsymbol{K}^{-1} \boldsymbol{f} \tag{2.1}$$



(a) Full beam



(b) Half beam

Figure 2-2: Messerschmitt-Bolkow-Blohm (MBB) beam problem

i.e. twice the strain energy of the system, where $\boldsymbol{K}$ is the stiffness matrix assembled from element stiffness matrices, $\boldsymbol{u}$ is the displacement vector of all the degrees of freedom of the ground mesh, such that u_i corresponds to the i^{th} degree of freedom of the ground mesh, and $\boldsymbol{f}$ is the load vector assembled from surface and point loads. The relationship between the global and element stiffness matrices is given by:

$$\boldsymbol{K} = \sum_e \rho_e(P(x_e))\boldsymbol{K}_e \tag{2.2}$$

$$\rho_e(x_e) = (1 - x_{min})x_e + x_{min} \tag{2.3}$$

where $\boldsymbol{K}_e$ is the element stiffness matrix of element e, which describes the relationship between the degrees of freedom inside element e, and such that the (i, j)th entry of $\boldsymbol{K}_e$ is 0 if either degree of freedom i or j is not in element e, P is the penalty

function with a penalty value p, and x_{min} is chosen to be a small value, e.g. 1e-3, to stop $\boldsymbol{K}$ from becoming singular when a node is not part of any solid element.

2.2 Literature review

Some efforts can be found in literature towards decreasing the computational burden of continuation SIMP and topology optimization at large. For instance, approximate reanalysis via reusing the stiffness matrix decomposition in direct linear solvers was proposed by Amir et al. Amir et al. (2009). Later, approximate analysis via reusing the preconditioner in the preconditioned conjugate gradient (PCG) algorithm was also successfully applied to solve compliance problems (Amir, 2015). The decomposition reuse approach was then extended to approximately solve the eigenvalue problem found in topology optimization of oscillating systems (Zheng et al., 2017). Use of multi-resolution meshes and adaptive mesh refinement can also be found in topology optimization literature. For instance, Amir et al. Amir et al. (2014) used a geometric multigrid preconditioner in the PCG algorithm based on a multi-resolution mesh. Adaptive mesh refinement can also be found in numerous old and recent works, e.g. Maute and Ramm (1995, 1998); Stainko (2006); Wang et al. (2010,?); Bruggi and Verani (2011); Wang et al. (2013, 2014); Salazar de Troya and Tortorelli (2018); Lambe and Czekanski (2018).

A few works also studied some penalty adaptation routines. Dadalau et al. (2009) provided a way to adapt the penalty step which was used along with a variant of the optimality criteria method for VCCM problems. More specifically, Dadalau et al.

(2009) set the penalty in each iteration such that it maximizes the ratio of the standard deviation of elements' fractional volumes, $\sigma(v_e x_e)$, and the strain energy, i.e. half the compliance. Maximizing this ratio is supposed to maximize the closeness to a binary solution, without compromising the compliance too much. However, there are two questionable points in this approach, beside not having a proof. Firstly, the standard deviation of elements' fractional volumes is not an accurate measure of how binary a solution is; it is a measure of spread around the mean. This means that the direction in which the standard deviation increases for any one element is always away from the mean. For instance, let the volume of each element be 1.0, and let all elements have $x_e = 0.3 \quad \forall e$, therefore the mean is 0.3 and standard deviation is 0. Making one $x_e = 0.5$, i.e. clearly more fractional, will increase the standard deviation to a strictly positive value, thus falsely claiming that the new solution is closer to a 0-1 design. When the volume constraint is active, the mean value of x_e, weighted by the element volumes, is exactly the volume fraction. The second problem with the approach proposed by Dadalau et al. (2009) is that as shown in their experiments, the penalty value p is allowed to freely increase or decrease with no control over the final penalty, which means that SIMP's (or RAMP's) property of converging to a binary solution for VCCM problems with a high penalty is lost.

Another penalty adaptation approach was proposed by Gao Xingjun et al. (2017) for buckling constrained compliance minimization problems. Gao Xingjun et al. (2017) chose the penalty in each subproblem to promote a certain amount of constraint violation such that the eigenvalue constraint is violated by 0.1 if possible while maintaining an increasing penalty. This was motivated by the observation that for

low penalty values, all the buckling load factors are typically over the threshold, so the stability constraint is not active and does not contribute to the optimization process. However, when the penalty increases, the violated stability constraints require a re-distribution of the material, which when the problem is non-convex, may converge to a bad and/or infeasible solution. The heuristic proposed was shown to improve the stability of the solution, though still infeasible, at the expense of a higher compliance.

However most notably, Rojas-Labanda and Stolpe (2015*a*) tried to reduce the number of FEA simulations needed by CSIMP for generic topology optimization problems using a number of alternative approaches, the most significant of which was the *automatic* continuation SIMP (Auto-CSIMP). In auto-CSIMP the penalty p is treated as a variable and a constraint is defined such that it is feasible only when $p = p_{max}$. A nonlinear equality (or inequality) constraint $g(p;\mu) = 0$ (or ≤ 0) is added, where μ is a hyperparameter, and the whole problem is solved using a first order NLP solver which solves for p and $\boldsymbol{x}$ simultaneously, using linear approximation of the nonlinear constraints, such as the interior point optimizer IPOPT (Wächter and Biegler, 2006). The main purpose of this additional constraint is to stall the convergence of the algorithm achieving a similar effect as the original CSIMP. While this method was shown to successfully decrease the number of simulations needed to converge, there are a few questionable points about the algorithm. Firstly, the choice of $g(p;\mu)$ and its hyperparameter μ seem largely arbitrary with the exception of being a decreasing function in p which was derived from Newton's method applied on the KKT conditions of the barrier problem used in IPOPT. However, the analysis

of the number of iterations needed to reach the final penalty for different functions g and values of μ is not accurate since it assumes a fixed step size of $\alpha = 1$ in Newton's method. So it is not clear that the choice of $g(p;\mu)$ or its hyperparameter thereof will never result in IPOPT satisfying the penalty constraint too quickly thus approaching the behavior of a single penalty SIMP. In the paper by Wächter and Biegler (2006) which describes the algorithm used in IPOPT, the actual step α_k is influenced by the search direction in $[\boldsymbol{x}; p]$, which is a function of:

(1) the Hessian of the Lagrangian of the barrier problem with respect to $[\boldsymbol{x}; p]$,

(2) the Jacobian of the constraints with respect to $[\boldsymbol{x}; p]$, and

(3) the initial primal and dual solutions.

Moreover, α_k is ultimately determined by how the objective and constraints change along the search direction during the backtracking line search for $[\boldsymbol{x}; p]$ combined, as well as the line search's stopping criteria. Additionally, the search direction can itself be influenced by the step size in case the second order correction step is performed. The influence of factors like the number of elements, the volume of each element, the Young's modulus and Poisson's ratio of the material on the step size that IPOPT takes was not studied and is indeed difficult to measure; so the risk of premature convergence of the penalty constraint exists. Secondly, it seems unclear whether using a different NLP solver or improving the existing IPOPT solver will result in a better or worse Auto-CSIMP. While Auto-CSIMP is, in theory, possible to use for any topology optimization problem and was shown to outperform the traditional MMA-based CSIMP with a large penalty step by Rojas-Labanda and

Stolpe (2015*a*), the design space for MMA-based CSIMP algorithms has not been fully explored yet.

2.3 Penalty adaptation

In every CSIMP subproblem, an NLP is solved using a certain penalty value p until convergence before moving on to the next penalty value. When using the KKT stopping criteria, at every penalty step, there is at least 1 FEA simulation to be performed in order to check for convergence. However in certain phases of CSIMP, e.g. near convergence, changing the value of p is not likely to change the solution or the objective much. Therefore, one improvement that can be employed is to use the last solution $\boldsymbol{u}$ from the last FEA simulation of the previous subproblem with the new penalty value to estimate the change in the objective value. If the change is large enough, the subproblem can be solved, otherwise, the penalty can be incremented right away saving unnecessary FEA simulations. The new subproblem's tolerance can be used as this criteria. This provides a simple way to adapt the penalty step not to perform unnecessary FEA simulations.

To develop some mathematical intuition for this approach, let's assume the relative change in the objective stopping criteria is used for the MMA, together with solution feasibility. One can then attempt to estimate the effect of p on the optimal value C^*, through a first order Taylor series expansion of $C^*(p)$, which is the optimal value of C for each penalty value p. Assuming that the local optimal solution $\boldsymbol{x}^*$ is a KKT point of the subproblem at the current p and that the optimal primal-dual

solution, the objective and constraints are continuous and differentiable functions of p, it is possible to prove that:

$$\frac{dC^*(p)}{dp} = \frac{\partial C}{\partial p} - \sum_i \lambda_i \frac{\partial h_i}{\partial p} \tag{2.4}$$

where h_i is the i^{th} constraint, and λ_i is the corresponding Lagrangian multiplier. For the VCCM problem, this is simply $\frac{dC^*(p)}{dp} = \frac{\partial C}{\partial p}$ since the constraints are not a function of p. This also means that the value of p does not change the feasibility status of a solution.

Proof. Let $\boldsymbol{x}(p)$ now denote the optimal solution $\boldsymbol{x}^*$ given a certain value of p. The KKT conditions guarantee that:

$$-\nabla_{\boldsymbol{x}} C + \sum_i \lambda_i \nabla_{\boldsymbol{x}} h_i = \mathbf{0} \tag{2.5}$$

The full derivative of the optimal compliance $C^*(p) = C(\boldsymbol{x}(p), p)$ with respect to p is:

$$\frac{dC^*(p)}{dp} = \frac{\partial C(p)}{\partial p} + \nabla_{\boldsymbol{x}} C \cdot \frac{d\boldsymbol{x}(p)}{dp} \tag{2.6}$$

Taking the dot product of the LHS and RHS of equation 2.5 with $\frac{d\boldsymbol{x}(p)}{dp}$ and adding $-\frac{\partial C}{\partial p} + \sum_i \lambda_i \frac{\partial h_i}{\partial p}$ to both sides, one can show that:

$$-\frac{dC^*}{dp} + \sum_i \lambda_i \frac{dh_i^*}{dp} = -\frac{\partial C}{\partial p} + \sum_i \lambda_i \frac{\partial h_i}{\partial p} \tag{2.7}$$

where $h_i^* = h_i(\boldsymbol{x}(p), p)$. Furthermore, the complementarity slackness property of the KKT solution specifies that:

$$\lambda_i(p) h_i(\boldsymbol{x}(p), p) = 0 \tag{2.8}$$

Using the product rule:

$$\frac{d\lambda_i(p)}{dp} h_i(\boldsymbol{x}(p), p) + \lambda_i(p) \frac{dh_i(\boldsymbol{x}(p), p)}{dp} = 0 \tag{2.9}$$

Multiplying both sides by $\lambda_i(p)$ and using equation 2.8, one can show that:

$$\lambda_i(p)^2 \frac{dh_i(\boldsymbol{x}(p), p)}{dp} = 0 \tag{2.10}$$

so either $\lambda_i(p) = 0$ or $\frac{dh_i(\boldsymbol{x}(p), p)}{dp} = 0$. In other words, equation 2.7 simplifies into equation 2.4. *This completes the proof.* □

Let $\boldsymbol{u}, \boldsymbol{x}$ and p be independent variables, as in Simultaneous Analysis and Design (SAND). The partial derivative of C with respect to p is:

$$\frac{\partial C(\boldsymbol{u}, \boldsymbol{x}, p)}{\partial p} = \boldsymbol{u}^T \frac{\partial \boldsymbol{K}(\boldsymbol{x}, p)}{\partial p} \boldsymbol{u} \tag{2.11}$$

$$\frac{\partial \boldsymbol{K}(\boldsymbol{x}, p)}{\partial p} = \sum_e log(x_e) \times x_e^p \times \boldsymbol{K}_e \tag{2.12}$$

Note that this function is discontinuous at any point where some $x_e = 0$. However, using L'Hopital's rule on $\frac{log(x_e)}{1/x_e^p}$, $\lim_{x_e \to 0} log(x_e) \times x_e^p = -\frac{x_e^p}{p} = 0$. Therefore, $\frac{\partial \boldsymbol{K}}{\partial p}$

can be numerically defined as:

$$\frac{\partial \boldsymbol{K}}{\partial p} = \sum_e \Gamma_e \times \boldsymbol{K}_e \tag{2.13}$$

$$\Gamma_e = \begin{cases} log(x_e) \times x_e^p & x_e \geq \epsilon \\ 0 & x_e < \epsilon \end{cases} \tag{2.14}$$

for some small ϵ.

Incidentally, $\frac{\partial C(\boldsymbol{u},\boldsymbol{x},p)}{\partial p}$, when $\boldsymbol{u}$ is treated as a variable, is equal to $-\frac{\partial C(\boldsymbol{x},p)}{\partial p}$ when treating $\boldsymbol{u}$ as a function of $\boldsymbol{x}$ and p, as in Nested Analysis and Design (NAND). Since $\boldsymbol{u}(\boldsymbol{x}, p)$ is implicitly defined by:

$$\boldsymbol{K}(\boldsymbol{x}, p)\boldsymbol{u}(\boldsymbol{x}, p) = \boldsymbol{f} \tag{2.15}$$

then:

$$\begin{aligned} \frac{\partial \boldsymbol{K}}{\partial p}\boldsymbol{u} + \boldsymbol{K}\frac{\partial \boldsymbol{u}}{\partial p} &= 0 \\ \frac{\partial \boldsymbol{u}}{\partial p} &= -\boldsymbol{K}^{-1}\frac{\partial \boldsymbol{K}}{\partial p}\boldsymbol{u} \end{aligned} \tag{2.16}$$

Hence, the partial derivative of $C(\boldsymbol{x}, p) = \boldsymbol{u}(\boldsymbol{x}, p)^T \boldsymbol{K}(\boldsymbol{x}, p)\boldsymbol{u}(\boldsymbol{x}, p)$ with respect to p is:

$$\begin{aligned} \frac{\partial C(\boldsymbol{x}, p)}{\partial p} &= 2\boldsymbol{u}^T \boldsymbol{K}\frac{\partial \boldsymbol{u}}{\partial p} + \boldsymbol{u}^T \frac{\partial \boldsymbol{K}}{\partial p}\boldsymbol{u} \\ &= -\boldsymbol{u}^T \frac{\partial \boldsymbol{K}}{\partial p}\boldsymbol{u} \end{aligned} \tag{2.17}$$

Since $x_e \leq 1$, and $\boldsymbol{K}_e$ is positive semi-definite for all e, the above term is non-negative. In this chapter, the NAND framework is used.

Assuming the first order approximation, it is also possible to choose the penalty step Δp_k such that the approximate compliance:

$$\tilde{C}_k = C(\boldsymbol{u}_{k-1}, \boldsymbol{x}_{k-1}, p_{k-1} + \Delta p_k) \approx C_{k-1} + \frac{\partial C}{\partial p}(\boldsymbol{u}_{k-1}, \boldsymbol{x}_{k-1}, p_{k-1}) \times \Delta p_k \quad (2.18)$$

satisfies inequality (2.19)

$$|C_{k-1} - C_k|/|C_{k-1}| < tol \quad (2.19)$$

for some tolerance tol. Choosing Δp such that:

$$|C_{k-1} - \tilde{C}_k|/|C_{k-1}| = \beta \times tol \quad (2.20)$$

for some $\beta \geq 1$, we get:

$$\left|\frac{\partial C}{\partial p}\right| \frac{\Delta p}{|C_{k-1}|} = \beta \times tol$$
$$\Delta p = \frac{\beta \times tol \times |C_{k-1}|}{|\partial C/\partial p|} \quad (2.21)$$

On the other hand, using blind penalty adaptation with $\Delta p = \Delta p_0 \times n$ for $n \in \mathbb{N}$, where Δp_0 is the fixed step in CSIMP, the benefit of penalty adaptation can only be observed when $n > 1$, that is some subproblem is skipped because it is deemed

unnecessary. The condition for $n > 1$ is:

$$\Delta p_0 \leq \frac{tol \times |C_{k-1}|}{|\partial C/\partial p|} \tag{2.22}$$

More generally, the condition for i subproblems to be skipped, i.e. $\Delta p \geq \Delta p_0 \times (i+1)$ is:

$$\Delta p_0 \times i \leq \frac{tol \times |C_{k-1}|}{|\partial C/\partial p|} \tag{2.23}$$

Not surprisingly, the penalty adaptation trick can be observed to be more effective in skipping subproblems for small Δp_0 and large tol. While the intuition above was developed using the relative change in objective value, computational experiments show a significant reduction in the number of FEA simulations from skipping subproblems when using the more rigorous KKT stopping criteria.

2.4 Test Problems

There is a wide array of test problems used in literature to evaluate topology optimization algorithms. In the majority of papers, a few test problems are used to demonstrate the effect of algorithms, e.g. Huang and Xie (2010), Stolpe and Svanberg (2001*a*), and Salazar de Troya and Tortorelli (2018). Valdez et al. (2017) compiled a list of linearly elastic 2D structures commonly used as test problems in literature. Another set of problems for compliant mechanisms was also proposed by Deepak et al. (2009). However, perhaps the largest set of benchmark problems ever

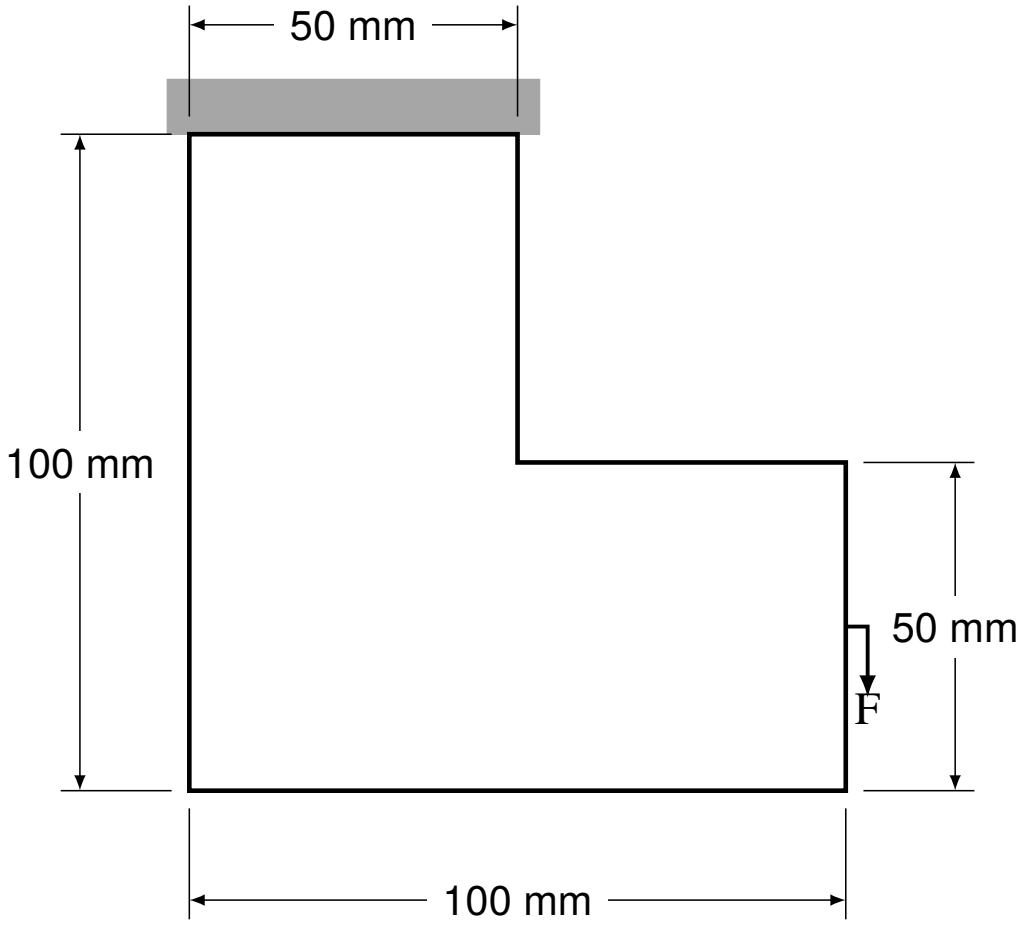


Figure 2-3: L-shaped beam problem

used in literature was by Rojas-Labanda and Stolpe (2015*b*,*a*). Rojas-Labanda et al. used a library of 225 minimum compliance problems, 135 minimum volume and 150 compliant mechanism design problems in Rojas-Labanda and Stolpe (2015*b*). The same authors seem to have used a subset of this library in Rojas-Labanda and Stolpe (2015*a*). In this chapter, three 2D VCCM problems, shown in Figures 2-1-2-3, and 1 3D VCCM problem are used to highlight the effect of penalty adaptation. A possible future work is to perform a more extensive study on a larger library of problems.

The first test problem is the 2D cantilever beam problem, Figure 2-1, which was solved by Huang and Xie (2010) to benchmark soft-kill BESO against CSIMP. The second test problem is the MBB problem, Figure 2-2, was solved in the classical paper by Sigmund (2001). Due to the symmetry of the problem, the problem in Fig. 2-2a is typically reduced to the problem in 2-2b to reduce the area of the topology to be optimized. The third test problem is the L-beam problem, Figure 2-3, which was taken from the book by Bendsøe and Sigmund (2004). These three problems are all part of the 2D collection made by Valdez et al. (2017). Finally, a 3D cantilever beam variant of the first test problem will be tested with a ground mesh of dimensions $60 \times 20 \times 20$.

In the 2D test problems, a ground mesh of plane stress quadrilateral elements is used, where each element is a square of side length 1mm, and a sheet thickness of 1mm. In the 3D cantilever beam problem, hexahedral elements are used, where each element is a cube of side length 1mm. Linear isoparameteric interpolation functions are used in the MBB, L-beam and 3D cantilever beam problems, and quadratic

isoparameteric interpolation functions are used in the 2D cantilever beam problem. The 2D cantilever beam problem with a low volume fraction and linear interpolation functions was found to produce some extremely ill-conditioned geometries with corner contacts, hence the use of quadratic interpolation functions to achieve better accuracy in the FEA simulations. A Young's modulus of 1 MPa and Poisson's ratio of 0.3 are used in all the problems. A chequerboard density filter was used with a radius of 2 mm. The weights were calculated using the method described by Huang and Xie (2010).

2.5 Implementation

2.5.1 Finite element analysis

All the topology optimization algorithms described in this chapter were implemented using the Julia programming language (Bezanson et al., 2014) for handling generic unstructured, isoparameteric meshes. Finite element analysis was done with the help of the finite element tooling package JuAFEM.jl [1]. A direct Cholesky factorization based linear system solver for sparse matrices was used from SuiteSparse [2] wrapped in Julia. The value of x_{min} used was 0.001 for all problems and algorithms since this is the standard value used in literature.

[1] https://github.com/KristofferC/JuAFEM.jl
[2] http://faculty.cse.tamu.edu/davis/suitesparse.html

2.5.2 Optimization

The original MMA algorithm by Svanberg (1987) was implemented and used to solve the SIMP subproblems. MMA parameters of $s_{init} = 0.5$, $s_{incr} = 1.2$ and $s_{decr} = 0.7$ were used as defined in the paper by Svanberg (1987). The dual problem of the convex approximation was solved using a log-barrier box-constrained nonlinear optimization solver, where the barrier problem was solved using the nonlinear CG algorithm for unconstrained nonlinear optimization (Nocedal and Wright, 2006) as implemented in Optim.jl [3] (K Mogensen and N Riseth, 2018). The nonlinear CG itself used the line search algorithm from Hager et al. Hager and Zhang (2006) as implemented in LineSearches.jl [4].

The stopping criteria used was similar to the one adopted by the KKT solver, IPOPT (Wächter and Biegler, 2006). This stopping criteria is less scale sensitive than the KKT residual as it scales down the residual by a value proportional to the mean absolute value of the Lagrangian multipliers. Let the optimization problem be:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) \tag{2.24a}$$

subject to

$$\boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{0}, \tag{2.24b}$$

$$0 \leq x_e \leq 1 \quad \forall e \tag{2.24c}$$

[3] https://github.com/JuliaNLSolvers/Optim.jl
[4] https://github.com/JuliaNLSolvers/LineSearches.jl

Let $\boldsymbol{\lambda}$ be the vector of Lagrangian multipliers of the nonlinear constraints, $\boldsymbol{z}_-$ be the vector of Lagrangian multipliers of the $\geq$ bound constraints, and $\boldsymbol{z}_+$ be the vector of Lagrangian multipliers of the $\leq$ bound constraints. The termination criteria used was:

$$\max\left\{\frac{||\nabla f(\boldsymbol{x}) + \nabla \boldsymbol{c}(\boldsymbol{x})\boldsymbol{\lambda} + (\boldsymbol{z}_+ - \boldsymbol{z}_-)||_\infty}{s_d}, ||\boldsymbol{c}(\boldsymbol{x})_+||_\infty\right\} \leq tol \tag{2.25}$$

$$s_d = \max\left\{s_{max}, \frac{||\boldsymbol{\lambda}||_1 + ||\boldsymbol{z}_+ + \boldsymbol{z}_-||_1}{m+n}\right\}/s_{max} \tag{2.26}$$

where $\boldsymbol{c}_+$ is the element-wise maximum of $\boldsymbol{c}$ and 0, and s_{max} is a constant parameter. A value of $s_{max} = 100$ was used by Wächter and Biegler (2006) and the same is done in this work.

2.6 Evaluating SIMP

There are a number of difficulties that one faces when trying to fairly assess the performance of SIMP variants to solve the VCCM problem:

1. The solution generated by SIMP is a continuous one, where x_e is allowed to take fractional values. Therefore, the objective value is that of the continuous approximation of the problem. Moreover, x_{min} and P can affect the objective value differently for different ground mesh geometries.

2. Obtaining a discrete solution from a continuous one can be done in different ways:

(a) The solution can be rounded. This approach carries the risk of obtaining an infeasible solution after rounding even if the continuous solution was feasible.

(b) A projection function such as the regularized Heaviside projection function $f(x) = 1 - e^{\beta x} + xe^{-\beta}$ (Guest et al., 2004) can be used to make the projected solution closer to a binary one. This method has the disadvantage of increasing the non-convexity of the optimization problem, requiring more iterations to converge, for $\beta > 0$. The regularized Heaviside function is the identity function when $\beta = 0$, and it gradually approximates the Heaviside step function as $\beta \to \infty$. A continuation over β is therefore typically used. From experiments, using a high value of β can cause numerical difficulties for the NLP solver because the slope at $x \approx 0$ becomes very high. Moreover, a very high value of β is often required to get a near-binary solution.

(c) A subset of the elements can be selected based on their sorted x_e values. In VCCM problems, this can be done by selecting the top $100V\%$ of the elements by x_e. However, for other constraints which require solving an FEA simulation to evaluate the feasibility of a solution, this approach is not straight forward to generalize. For constraints which are known to be satisfied with a full grid design, a bisection search can be used, adding or removing elements until a feasible binary solution is obtained.

For the sake of this chapter, the regularized Heaviside projection was used

together with the element selection method, where the top $100V\%$ elements were selected as per the projected filtered densities, then the discrete objective value was computed.

3. After obtaining a discrete solution, the solution can exhibit chequerboard patterns, disconnected or undesired thin features. These are mostly visual features for which no metric in topology optimization literature exists as far as we know.

4. Some SIMP variants can converge quickly to a solution with a good continuous objective value but a bad or undefined discrete objective value, due to disconnectivity or ill-conditioned geometries, e.g. corner contacts. For example in the VCCM problem, using a single low penalty value can generally lead to a lower continuous objective value, closer the lower bound which is the optimal value at $p = 1$, but the discrete design's objective value can be either infeasible or suboptimal.

In SIMP literature, algorithms seem to be compared using their continuous objective values. Therefore, we do the same in this chapter. However, an additional metric will be added to measure how fractional a solution is. A more fractional solution is arguably a less trusted one since the continuous and discrete objectives are likely to be far apart. The measure of how fractional a solution is to be used in this chapter

will be:

$$\%frac = \frac{2\sum_{e} \min(x_e, 1 - x_e) \times v_e}{\sum_{e} v_e} \times 100\% \tag{2.27}$$

which is 0% when a solution is fully binary, 100% when a solution is all 0.5 and it weighs the fractional part of a variable by the volume of the corresponding element. The fractionness of the solution will be evaluated using the projected filtered densities.

In the following section, C-Obj will refer to the objective value of the continuous optimal solution. This includes the effect of x_{min}, the density filter and projection. One additional metric to be used is the discrete design's objective value (D-Obj) after selecting the top $100V\%$ of the elements by their projected and filtered densities, and removing the void elements completely.

2.7 Results and Discussion

2.7.1 Fixed tolerance

Table 2.1: The problem type and parameter settings of the experiments with continuation SIMP with a fixed tolerance.

Exp	Problem	V	tol	Δp_0	Exp	Problem	V	tol	Δp_0
1	CantBeam	0.3	0.01	0.1	17	LBeam	0.3	0.01	0.1
2				0.05	18				0.05
3			0.0001	0.1	19			0.0001	0.1
4				0.05	20				0.05
5		0.5	0.01	0.1	21		0.5	0.01	0.1
6				0.05	22				0.05
7			0.0001	0.1	23			0.0001	0.1
8				0.05	24				0.05
9	HalfMBB	0.3	0.01	0.1	25	3D CantBeam	0.3	0.01	0.1
10				0.05	26				0.05
11			0.0001	0.1	27			0.0001	0.1
12				0.05	28				0.05
13		0.5	0.01	0.1	29		0.5	0.01	0.1
14				0.05	30				0.05
15			0.0001	0.1	31			0.0001	0.1
16				0.05	32				0.05

The CSIMP algorithm with a fixed tolerance was tested on a number of problem instances as shown in Table 2.2. The effect of the penalty adaptation on the continuous objective value (C-Obj), the discrete solution's objective value (D-Obj), the percentage of fractionness (% frac) and the number of FEA simulations is shown below. The experiments' parameters are shown in Table 2.1.

A few observations can be made from the results of the experiments with a fixed tolerance:

1. Penalty adaptation consistently leads to a reduction in the number of FEA simulations in all experiments run.

2. Decreasing tol generally results in a decrease in C-Obj, $\%frac$ and the gap between C-Obj and D-Obj.

3. In some cases where premature convergence of one subproblem due to a high tol leads the algorithm to a different basin of attraction in the next subproblem, a better solution can be obtained for the higher tol variant. This is observed in Exps 9 and 10 compared to Exps 11 and 12 respectively. This is due to the non-convexity of the optimization problem.

4. Increasing tol increases the efficacy of penalty adaptation in reducing the number of FEA simulations needed to converge.

5. Reducing Δp_0 generally increases the efficacy of penalty adaptation. From equation 2.21, one can predict that a smaller Δp_0 leads to a smaller predicted change, which if lower than the tolerance, a subproblem will be skipped.

Table 2.2: This table shows the effect of penalty adaptation on CSIMP. The table shows the results of the experiments studying the effect of solution reuse on: 1) the number of FEA simulations (Sims) required by CSIMP, 2) the final objective of the continuous NLP at $p = 5$ (C-Obj), 3) the objective value of the rounded discrete solution (D-Obj), and 4) the percentage of fractionness (% frac). The parameters of each experiment are described in Table 2.1

Exp	**Fixed $\Delta p = \Delta p_0$**			**% Increase due to Δp adaptation**		
	Sims	**C-Obj / D-Obj**	**% frac**	**Sims**	**C-Obj / D-Obj**	**% frac**
1	149	813.0/555.8	17.3	-20.8	-9.3/1.0	-13.3
2	213	742.2/572.4	14.3	-45.1	1.6/-1.4	5.6
3	291	720.4/549.2	14.1	-18.2	0.5/-0.7	0.7
4	485	712.5/559.9	13.8	-18.1	-0.3/0.5	-0.7
5	103	383.8/343.1	14.0	-32.0	-1.9/0.6	-7.9
6	149	382.7/338.3	12.4	-53.0	-0.1/1.7	6.5
7	251	366.7/338.4	10.0	-21.5	0.1/0.0	1.0
8	376	366.9/338.0	9.9	-24.5	-0.0/0.2	0.0
9	110	584.2/293.5	24.5	-40.0	5.1/-0.4	2.0
10	166	572.7/293.3	24.4	-59.6	9.5/-0.3	2.9
11	206	609.9/291.6	24.5	-18.0	0.1/-0.1	0.0
12	310	607.5/293.5	24.4	-25.2	0.4/0.0	0.4
13	97	229.6/176.7	19.9	-37.1	-5.4/-0.1	-5.0
14	137	229.8/176.8	20.0	-54.7	-4.7/-0.2	-4.0
15	256	219.3/176.5	18.6	-21.9	-0.3/-0.5	-3.2
16	339	217.7/176.9	17.8	-17.4	-0.2/-0.6	0.0
17	104	126.8/103.4	10.2	-33.7	-2.6/-0.2	0.0
18	148	128.4/102.4	10.4	-52.7	-4.4/0.7	-3.8
19	413	117.1/102.0	7.8	-18.4	4.0/1.4	21.8
20	505	121.8/103.5	9.5	-13.7	-0.1/0.1	-1.1
21	91	73.3/66.7	10.1	-39.6	1.0/0.1	-3.0
22	133	72.4/66.5	9.5	-56.4	1.2/0.3	-2.1
23	269	69.9/65.2	7.5	-14.5	-0.1/0.0	-1.3
24	367	69.9/65.3	7.5	-15.5	0.0/-0.2	-2.7
25	63	17.7/13.2	13.0	-61.9	-1.7/0.0	-2.3
26	104	17.3/13.2	12.7	-76.0	-0.6/0.0	0.0
27	232	16.3/13.3	10.0	-20.3	-1.2/-0.8	-1.0
28	299	16.2/13.2	10.0	-23.1	-1.9/-0.8	-3.0
29	65	10.7/9.3	13.3	-60.0	-0.9/0.0	-0.8
30	106	10.6/9.3	13.0	-74.5	0.0/0.0	-0.8
31	214	10.3/9.3	12.3	-22.0	0.0/0.0	0.8
32	293	10.3/9.2	12.2	-37.9	0.0/1.1	-2.5

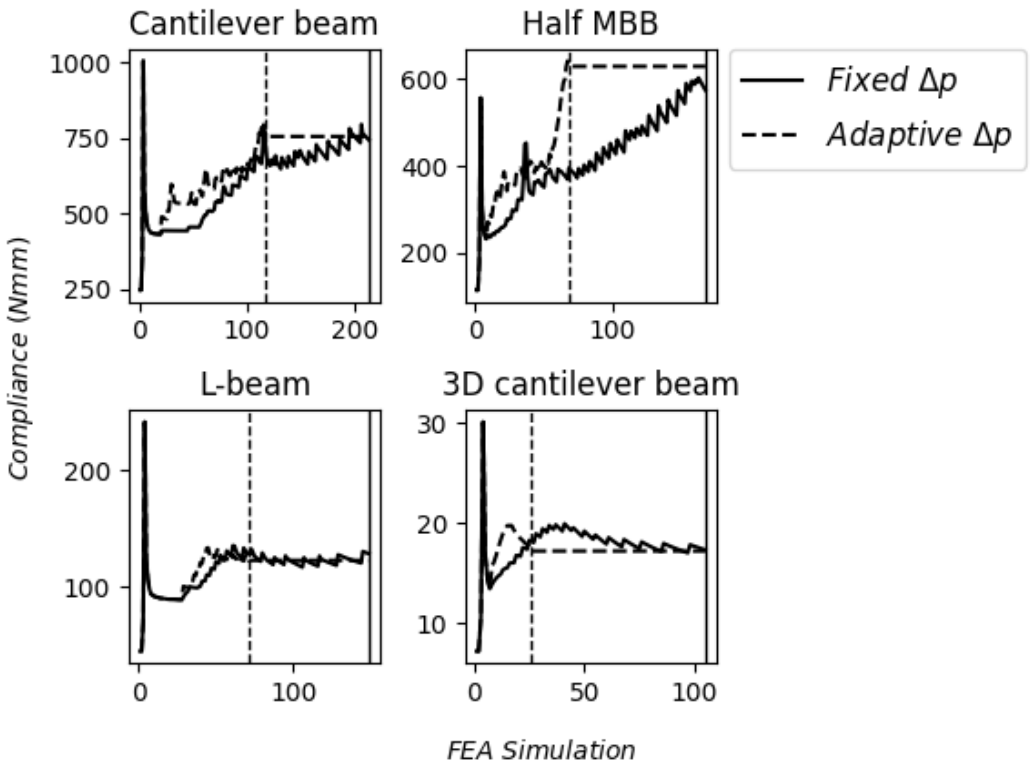


Figure 2-4: Convergence plots for CSIMP with and without penalty adaptation using $V = 0.3$, $tol = 0.01$, $\Delta p = 0.05$.

However, it is not clear from the onset that such skipping of subproblems will lead to a saving in the number of FEA simulations or otherwise. Experiments show that the savings can indeed be significant.

6. A reduction in the volume fraction V causes a larger discrepancy between the continuous and discrete objective values. This makes sense because void elements are not completely removed in the continuous objective therefore their effect will be more obvious as their ratio increases.

Figures 2-4 and 2-5 show typical convergence plots for the test problems using high and low tolerances respectively with and without penalty adaptation. It is clear that the curves follow a similar pattern and converge to similar objective values but

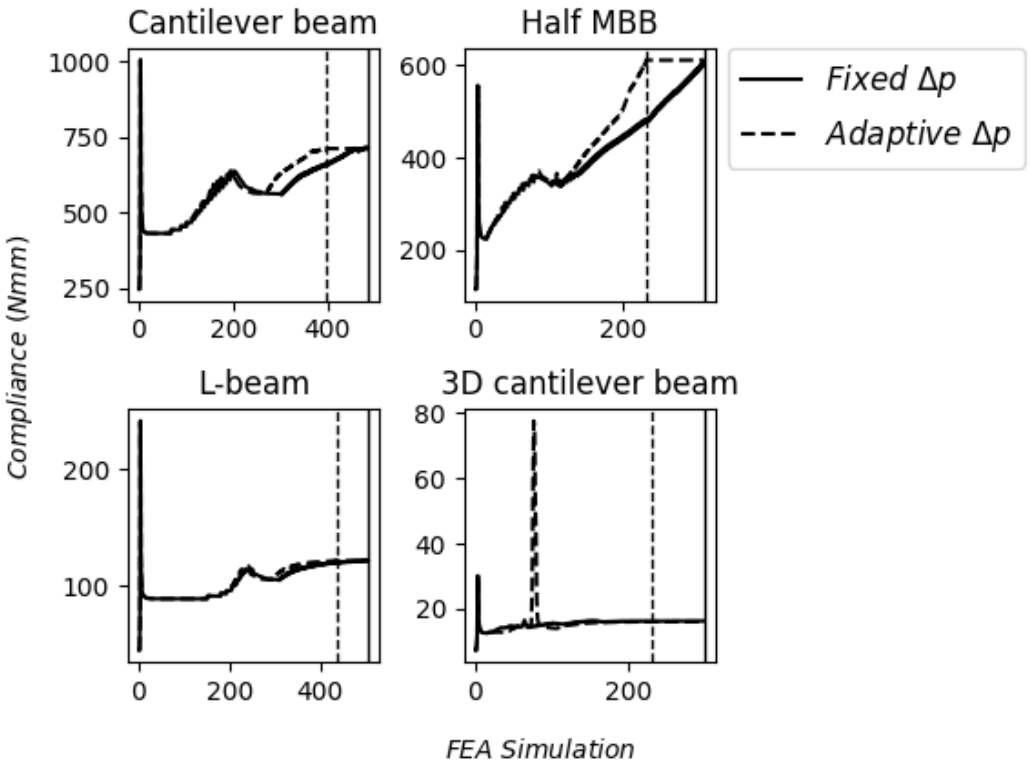


Figure 2-5: Convergence plots for CSIMP with and without penalty adaptation using $V = 0.3$, $tol = 0.0001$, $\Delta p = 0.05$.

the penalty adaptation causes an acceleration to the convergence. In one case, the penalty adaptation curve had a higher peak which can be explained by the extreme changes in the objective of the current solution when skipping many subproblems optimization.

Overall, the results seem to suggest the need for a decreasing tolerance scheme to allow penalty adaptation to be more effective than when using a fixed low tolerance, while reaping the benefits of a low final tolerance in the objective value and reducing the % frac of the continuous solution.

2.7.2 Decreasing tolerance

In the previous section, the effect of the tolerance on $\%frac$, C-Obj and D-Obj was observed. While a lower tolerance seems to have some undeniable benefits in most test problems, it is certainly more computationally expensive. It is therefore customary to use a decreasing tolerance scheme with $\theta < 1$ in Algorithm 1 since for low penalty values p and low β, the solution is still highly fractional and therefore solving it to a low tolerance is thought to be unnecessary. Algorithm 1 uses an exponentially decaying tolerance curve. Other curves can also be used to raise the average tolerance of all the subproblems without changing the minimum and maximum tolerances. In this section, an exponentially decaying tolerance function will be used which starts at tol_0 in the first subproblem and ends at tol_{min} in the last. For $tol_0 = 0.01$ and $tol_{min} = 0.0001$, using a $\Delta p_0 = 0.1$ corresponds to $\theta = 0.8913$, and using $\Delta p_0 = 0.05$ corresponds to $\theta = 0.9441$. Note that the tolerance is

decreased in each subproblem and the new tolerance is used to decide whether the subproblem is worth solving or not as described in section 2.3. Given that penalty adaptation was observed to be more effective when using higher tolerances than lower ones, a decreasing tolerance scheme is likely to enhance the benefits of penalty adaptation. To avoid the skipping of all subproblems, the last subproblem will always be solved, i.e. never skipped. This should guarantee that the proposed algorithm has at least the same theoretical properties of a fixed penalty SIMP where $p = p_{max}$, thus making it theoretically no worse than the status quo.

Table 2.3: The problem type and parameter settings of the experiments with decreasing tolerance continuation SIMP. $tol_0 = 0.01$ and $tol_{min} = 1e-4$ were used in all test cases.

Exp	Problem	V	Δp_0	Exp	Problem	V	Δp_0
1	CantBeam	0.3	0.1	9	LBeam	0.3	0.1
2			0.05	10			0.05
3		0.5	0.1	11		0.5	0.1
4			0.05	12			0.05
5	HalfMBB	0.3	0.1	13	3D CantBeam	0.3	0.1
6			0.05	14			0.05
7		0.5	0.1	15		0.5	0.1
8			0.05	16			0.05

Figure 2-6 shows a typical convergence plot of the decreasing tolerance schemes with and without penalty step adaptation. It is clear that penalty adaptation was successful in many cases in accelerating the convergence of the algorithm for all the

Table 2.4: This table shows the effect of penalty adaptation on CSIMP with decreasing tolerance. The table shows the results of the experiments studying the effect of solution reuse on: 1) the number of FEA simulations (Sims) required by CSIMP, 2) the final objective of the continuous NLP at $p = 5$ (C-Obj), 3) the objective value of the rounded discrete solution (D-Obj), and 4) the percentage of fractionness (% frac). The parameters of each experiment are described in Table 2.3.

Exp	**Fixed $\Delta p = \Delta p_0$**			**% Increase due to Δp adaptation**		
	Sims	**C-Obj / D-Obj**	**% frac**	**Sims**	**C-Obj / D-Obj**	**% frac**
1	185	814.2/581.6	17.3	-14.6	-8.8/-1.3	-18.5
2	316	720.4/553.1	14.0	-16.5	4.1/6.2	0.7
3	139	375.0/342.3	11.5	-10.8	-0.1/-0.0	0.0
4	236	367.3/338.9	9.8	-20.3	0.1/0.1	0.0
5	130	619.6/293.9	24.6	-25.4	0.4/-0.1	0.8
6	260	606.5/292.8	24.3	-39.6	2.6/-0.0	1.6
7	130	219.5/175.3	18.1	-26.2	0.2/0.7	0.6
8	211	218.8/175.1	17.9	-21.8	-0.2/0.5	-1.1
9	153	121.5/103.4	9.0	-20.3	-0.1/0.6	0.0
10	226	123.5/104.1	9.6	-17.7	0.2/0.2	1.0
11	142	69.8/65.6	6.8	-31.7	0.1/0.2	0.0
12	192	69.9/65.7	6.8	-38.0	0.1/0.0	1.5
13	97	16.1/13.1	10.1	-41.2	0.0/0.0	1.0
14	161	16.0/13.1	10.1	-49.1	0.0/0.0	0.0
15	97	10.2/9.3	10.9	-39.2	0.0/0.0	0.0
16	161	10.2/9.3	11.0	-49.1	0.0/0.0	0.0

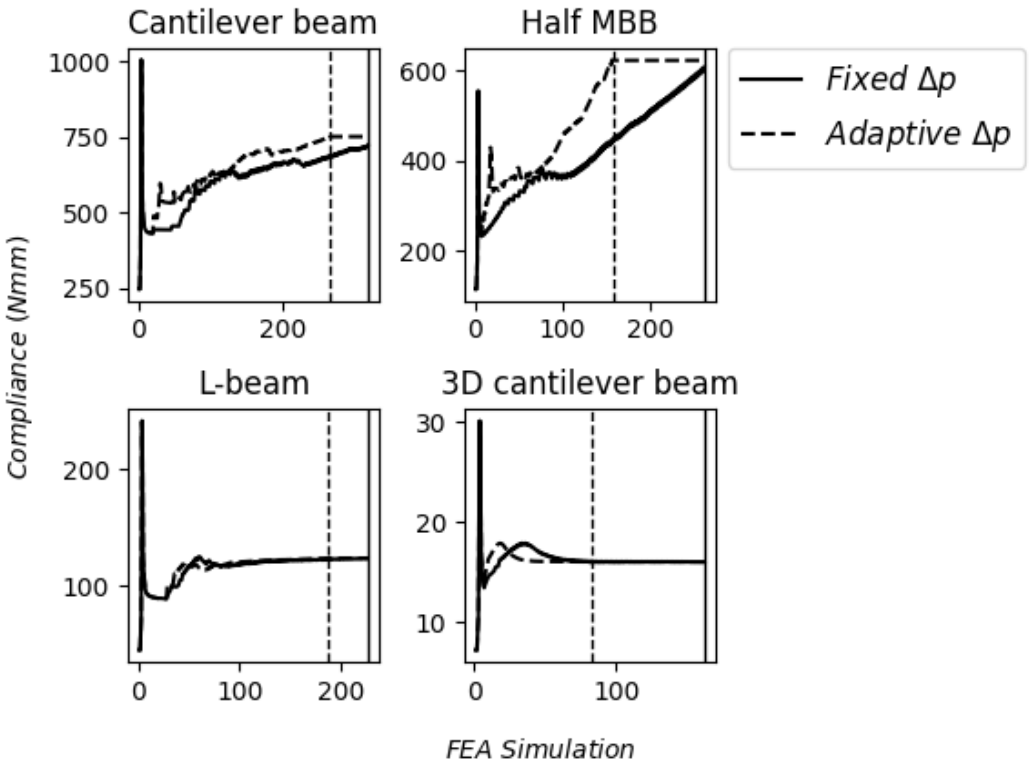


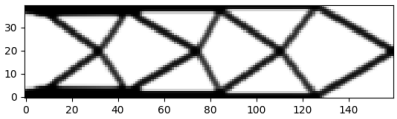
Figure 2-6: Convergence plots of Dec-Tol CSIMP using $tol_0 = 0.01$, $tol_{min} = 0.0001$, $V = 0.3$, and $\Delta p = 0.05$ with and without penalty adaptation for the 4 test problems.

test problems, reducing the number of FEA simulations required by an average of 28.8 %, without significant compromise to C-Obj, 4.1% in the worst case. Finally, Figures 2-7 and 2-8 show the final topologies of some of the test instances with and without penalty step adaptation using the decreasing tolerance scheme. For clarity, the binary solution is shown in the 3D case, while the continuous one is shown for the 2D problems.

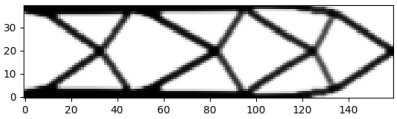
To further highlight that the reduction in the number of FEA simulations due to penalty adaptation is not trivial, Figure 2-9 shows the intermediate solution of Dec-Tol CSIMP without penalty adaptation when solving the L-beam problem in Exp 10, $V = 0.3$, and $\Delta p = 0.05$ after 186 FEA simulations. Dec-Tol CSIMP with penalty adaptation only required 186 FEA simulations to converge for this problem while Dec-Tol CSIMP without penalty adaptation required 226. The intermediate solution, shown in Figure 2-9a is infeasible due to disconnected parts. The compliance of the binary solution is therefore not defined. This is in contrast to Figure 2-10 which shows the solution when using penalty adaptation after the same number of FEA simulations as Figure 2-9.

2.8 Effect of x_{min}

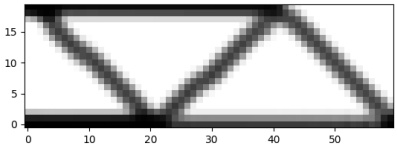
In the vast majority of papers in literature, the value of x_{min} used is 0.001. Some papers seem to penalize the value of $0 \leq x_e \leq 1$ while others seem to penalize the value of $\rho_e = x_e(1 - x_{xmin}) + x_{min}$. In this chapter, the former has been used. The latter has the effect of decreasing the minimum value that ρ_e can take as the penalty



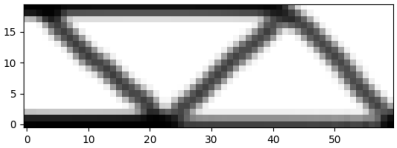
(a) Fixed $\Delta p = 0.05$, $V = 0.3$.



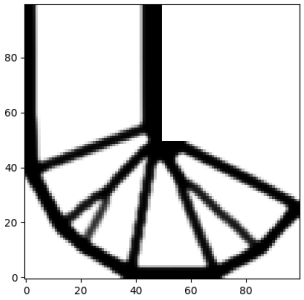
(b) Adaptive Δp, $\Delta p_0 = 0.05$, $V = 0.3$.



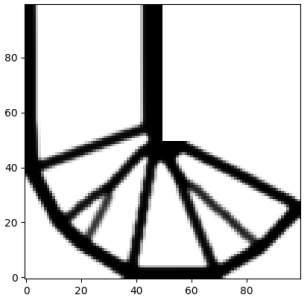
(c) Fixed $\Delta p = 0.05$, $V = 0.3$.



(d) Adaptive Δp, $\Delta p_0 = 0.05$, $V = 0.3$.

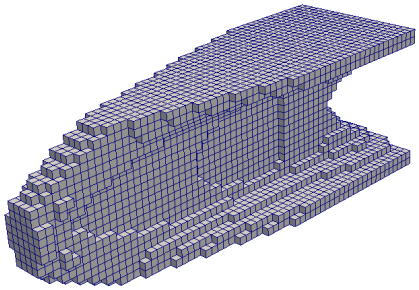


(e) Fixed $\Delta p = 0.05$, $V = 0.3$.

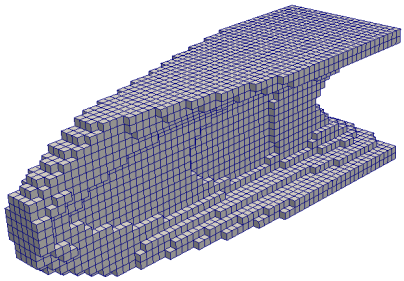


(f) Adaptive Δp, $\Delta p_0 = 0.05$, $V = 0.3$.

Figure 2-7: Continuous solutions of Dec-Tol CSIMP with a maximum tolerance of 0.01 and a minimum tolerance of 0.0001, with and without Δp adaptation.

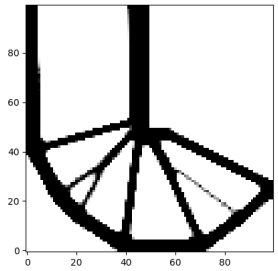


(a) Fixed $\Delta p = 0.05$, $V = 0.3$.

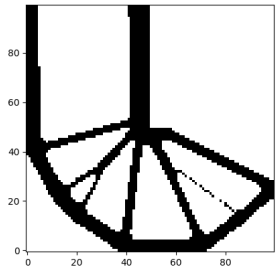


(b) Adaptive Δp, $\Delta p_0 = 0.05$, $V = 0.3$.

Figure 2-8: Binary solutions of the 3D cantilever beam problem using Dec-Tol CSIMP with a maximum tolerance of 0.01 and a minimum tolerance of 0.0001, with and without Δp adaptation.

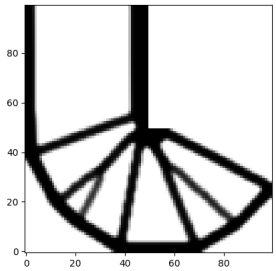


(a) Continuous intermediate solution.

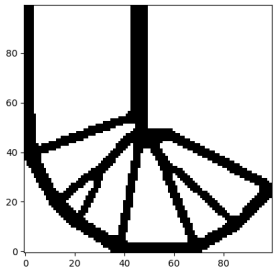


(b) Projected binary intermediate solution.

Figure 2-9: The continuous and projected intermediate solutions of the L-beam problem using Dec-Tol CSIMP without penalty adaptation after 186 FEA simulations using $V = 0.3$, and $\Delta p = 0.05$. CSIMP with penalty adaptation requires only 186 FEA simulations to converge, whereas without penalty adaptation, it takes 226 FEA simulations.



(a) This is the final continuous solution. The objective value is 123.7.



(b) This is the projected final solution where x_e is binary for all e. The objective value is 104.3.

Figure 2-10: The continuous and projected final solutions of the L-beam problem using Dec-Tol CSIMP with penalty adaptation using $V = 0.3$, and $\Delta p = 0.05$. Convergence happened after 186 FEA simulations.

increases. While the mentioned reason for using $x_{min} > 0$ in literature is to stop the stiffness matrix $\boldsymbol{K}$ from becoming singular, this is not free of limitations. In this section, it will be proven that in the VCCM problem, x_{min} plays the role of a penalty for 2 types of infeasible designs: 1) a design with a force on a node with no element to support it, and 2) a design with rigid body modes on which a non-orthogonal force is applied, i.e. the force has a component along one or more of the rigid body modes. The effect of x_{min} on the number of FEA simulations required to solve the problem will then be presented.

Let the nodes that have all their elements removed, i.e. $x_e = 0$, be called *shadow nodes*, and let any degree of freedom (dof) of a shadow node be called a *shadow dof*. In the case where $x_{min} = 0$, shadow dofs do not have any non-zero value in their corresponding rows and columns of the stiffness matrix $\boldsymbol{K}$. A low x_{min} approximates this without letting $\boldsymbol{K}$ become singular. Experiments show that the nodal displacements and compliance of the optimal solutions often reach a finite

limit value as $x_{min} \rightarrow 0$ for the optimal topologies, with non-zero displacements for the shadow nodes. When x_{min} is 0, $\boldsymbol{K}$ becomes rank deficient if a shadow dof exists. Note that:

$$\boldsymbol{u} = \boldsymbol{K}^{-1}\boldsymbol{f} = \sum_j \frac{1}{\lambda_j}\boldsymbol{v}_j\boldsymbol{v}_j^T\boldsymbol{f} \tag{2.28}$$

where $\boldsymbol{v}_j$ is the j^{th} eigenvector of $\boldsymbol{K}$ and λ_j is the corresponding eigenvalue. Additionally, without loss of generality, let the i^{th} dof be a shadow dof that has no Dirichlet boundary condition. It is easy to show that as x_{min} goes to 0, that the span of the eigenspace of $\boldsymbol{K}$ associated with the eigenvalue of 0, admits a new basis vector $\boldsymbol{e}_i$, where $\boldsymbol{e}_i$ is the i^{th} standard basis vector. This is easy to see since the matrix $\boldsymbol{K}$'s i^{th} row and column are all zeros, so $\boldsymbol{K}\boldsymbol{e}_i = 0 \times \boldsymbol{e}_i$. Given the limit eigenvector, $\boldsymbol{e}_i$, one can also obtain the limit eigenvalue using the Rayleigh quotient $\frac{\boldsymbol{e}_i^T\boldsymbol{K}\boldsymbol{e}_i}{\boldsymbol{e}_i^T\boldsymbol{e}_i} = K_{ii} = x_{min} \times \sum_e K_{e,ii}$, where K_{ii} is the i^{th} diagonal element of the matrix $\boldsymbol{K}$, $\boldsymbol{K}_e$ is the extended form of the element stiffness matrix of the element e such that $\boldsymbol{K}_e$ is the same size as $\boldsymbol{K}$ but only has non-zeros where element e's degrees of freedom are, and $K_{e,ii}$ is i^{th} diagonal element of $\boldsymbol{K}_e$. The following relationship therefore

holds:

$$\lim_{x_{min}\to 0} \boldsymbol{u} = \lim_{x_{min}\to 0} \sum_j \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.29}$$

$$= \lim_{x_{min}\to 0} \frac{1}{\lambda_i} \boldsymbol{v}_i \boldsymbol{v}_i^T \boldsymbol{f} + \sum_{j\neq i} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.30}$$

$$= \lim_{x_{min}\to 0} \frac{1}{x_{min} \times \sum_e K_{e,ii}} \boldsymbol{e}_i \boldsymbol{e}_i^T \boldsymbol{f} + \sum_{j\neq i} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.31}$$

$$= \lim_{x_{min}\to 0} \frac{f_i}{x_{min} \times \sum_e K_{e,ii}} \boldsymbol{e}_i + \sum_{j\neq i} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.32}$$

where f_i is the i^{th} element of the vector $\boldsymbol{f}$. Notice that if $f_i = 0$, the limit of the first term will be 0. Whereas, if $f_i \neq 0$, the limit will be $\pm\infty$ depending on the sign of f_i. The above derivations give a natural necessary condition for the finiteness of the displacements $\boldsymbol{u}$ and the compliance C. If the displacements and hence the compliance are to be finite, all shadow degrees of freedom should have no force component corresponding to them. In other words, every force should have at least one element to support it.

The above condition can be enforced with constraints or it can be enforced by making x_{min} small enough. Let x_{min} be small enough, so we can approximate $\boldsymbol{u}$ by its limit term:

$$\boldsymbol{u} \approx \sum_{i\in I} \frac{f_i}{x_{min} \times \sum_e K_{e,ii}} \boldsymbol{e}_i + \sum_{j\notin I} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.33}$$

where I is the set of shadow dofs. The compliance $C = \boldsymbol{f}^T \boldsymbol{u}$ can therefore be

approximated by:

$$C \approx \sum_{i \in I} \frac{f_i^2}{x_{min} \times \sum_e K_{e,ii}} + \sum_{j \notin I} \frac{1}{\lambda_j} (\boldsymbol{v}_j^T \boldsymbol{f})^2 \tag{2.34}$$

Since $\boldsymbol{K}$ is positive semi-definite, $\lambda_j \geq 0 \quad \forall j$. So we can approximately lower bound the compliance C using:

$$C \gtrapprox \sum_{i \in I} \frac{f_i^2}{x_{min} \times \sum_e K_{e,ii}} \tag{2.35}$$

Therefore, one can see that a lower x_{min} *penalizes* the compliance of the designs with shadow degrees of freedom corresponding to a force component. This means that if the binary solution projected from the optimal continuous one introduces shadow dofs, then the value of x_{min} may not have been small enough. In other words, a higher x_{min} increases the chances of getting an *optimal* design with a shadow dof where a force component is applied. This is clearly not a feasible design in the exact sense. The above derivations also suggest the possibility of using a weighted compliance term in the objective in classes of topology optimization problems other than VCCM to trigger the penalization effect of x_{min}.

Similar derivations can be followed to show that x_{min} also penalizes a design with rigid body modes. Let $\boldsymbol{K}(x_{min})$ be the stiffness matrix of a disconnected binary design for some value of x_{min}. Since $\boldsymbol{K}$ is positive definite when $x_{min} > 0$, $\boldsymbol{u}^T \boldsymbol{K} \boldsymbol{u} > 0 \quad \forall \boldsymbol{u}$. Let $\boldsymbol{v}_i$ be the i^{th} rigid body mode of a subset of the elements, e.g. a disconnected piece, when $x_{min} = 0$. Since $\boldsymbol{v}_i$ is a rigid body mode, $\boldsymbol{v}_i^T \boldsymbol{K}(0) \boldsymbol{v}_i = 0$.

Let $\boldsymbol{K} = \boldsymbol{K}_{f_i} + \boldsymbol{K}_{p_i} + x_{min} \times \boldsymbol{K}_{v_i}$ be the decomposition of $\boldsymbol{K}$ into free, pinned and void components of the binary design, respectively, with respect to the rigid body mode $\boldsymbol{v}_i$. $\boldsymbol{K}_{f_i}$ is assembled from the solid elements that the rigid body mode $\boldsymbol{v}_i$ moves, $\boldsymbol{K}_{p_i}$ is assembled from the remaining solid elements, and $\boldsymbol{K}_{v_i}$ is assembled from the void elements. Let $\boldsymbol{K}$ have the boundary conditions properly applied, with the Dirichlet dofs having only a positive diagonal term in $\boldsymbol{K}$. If such a dof is part of 2 matrices of $\boldsymbol{K}_{f_i}$, $\boldsymbol{K}_{p_i}$ and $\boldsymbol{K}_{v_i}$, half the value can be used in each. Let's first consider the case where the component with the free body mode is disconnected from the pinned component. Since $\boldsymbol{v}_i$ is a rigid body mode of the elements making up $\boldsymbol{K}_{f_i}$, and the set of dofs of $\boldsymbol{K}_{p_i}$ is disjoint from the set of dofs making up $\boldsymbol{K}_{f_i}$, $\boldsymbol{v}_i^T \boldsymbol{K}_{f_i} \boldsymbol{v}_i = \boldsymbol{v}_i^T \boldsymbol{K}_{p_i} \boldsymbol{v}_i = 0$. The same holds for the case where the free component is not fully disconnected but hinged on a point, or a line in 3D space, since the dofs of the hinge do not contribute to the rigid body mode. Therefore, $\boldsymbol{v}_i^T \boldsymbol{K} \boldsymbol{v}_i = x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i$. Assuming $\boldsymbol{v}_i$ is a unit vector, $\boldsymbol{v}_i^T \boldsymbol{K} \boldsymbol{v}_i$ is the Rayleigh quotient of the limit eigenvector $\boldsymbol{v}_i$ corresponding to a 0 eigenvalue, as $x_{min} \to 0$. Therefore, the following holds:

$$\lim_{x_{min} \to 0} \boldsymbol{u} = \lim_{x_{min} \to 0} \boldsymbol{K} (x_{min})^{-1} \boldsymbol{f} \tag{2.36}$$

$$= \lim_{x_{min} \to 0} \sum_{i \in J} \frac{\boldsymbol{v}_i^T \boldsymbol{f}}{x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i} \boldsymbol{v}_i + \sum_{j \notin J} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.37}$$

where $\{\boldsymbol{v}_i : i \in J\}$ is the set of rigid body modes at $x_{min} = 0$. Note that the second term above can further be broken down into 2 terms, one for the shadow dofs, and

one for the remaining components:

$$\lim_{x_{min}\to 0} \boldsymbol{u} = \lim_{x_{min}\to 0} \sum_{i\in I} \frac{f_i}{x_{min} \times \sum_e K_{e,ii}} \boldsymbol{e}_i + \sum_{i\in J} \frac{\boldsymbol{v}_i^T \boldsymbol{f}}{x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i} \boldsymbol{v}_i + \sum_{j\notin I, j\notin J} \frac{1}{\lambda_j} \boldsymbol{v}_j \boldsymbol{v}_j^T \boldsymbol{f} \tag{2.38}$$

The limit compliance can then be written as:

$$\lim_{x_{min}\to 0} C = \lim_{x_{min}\to 0} \sum_{i\in I} \frac{f_i^2}{x_{min} \times \sum_e K_{e,ii}} + \sum_{i\in J} \frac{(\boldsymbol{v}_i^T \boldsymbol{f})^2}{x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i} + \sum_{j\notin I} \frac{(\boldsymbol{v}_j^T \boldsymbol{f})^2}{\lambda_j} \tag{2.39}$$

When x_{min} is small enough that one can use the limit expression to approximate the value of C, one can see that x_{min} indeed plays the role of a penalty for any design with a shadow dof i and a force component $f_i \neq 0$, or any design with a rigid body mode on which a non-orthogonal force is applied such that $\boldsymbol{v}_i^T \boldsymbol{f} \neq 0$.

$$C \gtrapprox \sum_{i\in I} \frac{f_i^2}{x_{min} \times \sum_e K_{e,ii}} + \sum_{i\in J} \frac{(\boldsymbol{v}_i^T \boldsymbol{f})^2}{x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_v \boldsymbol{v}_i} \tag{2.40}$$

The experimental results shown in Table 2.5 indicate that a higher value of x_{min} can lead to faster convergence. This is not surprising since MMA uses first order approximation of the objective and constraints. So if the objective function varies significantly in the neighbourhood of local minima, the first order approximation is likely to cause the solution to jump from one basin of attraction to another during an MMA iteration. However, too high a value can lead to a final design that is infeasible in the exact sense. From the above approximate lower bound expression, one can see that the best choice of x_{min} is very much problem-dependent. This is

because the penalization effect of x_{min} is undermined if the norm of $\boldsymbol{K}_e$ is increased for all e. One way to increase the norm of $\boldsymbol{K}_e$ is to use a higher Young's modulus E.

A better way to choose the value of x_{min} is to make sure that $x_{min} = \frac{\epsilon}{\lambda_{max}}$, where ϵ is a small number and λ_{max} is the maximum eigenvalue of the stiffness matrix of the ground mesh. It is simple to show that $\sum_e K_{e,ii}$ and $\boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i$ are both less than or equal to λ_{max}.

Proof. Let λ_{min} and λ_{max} be the minimum and maximum eigenvalues of the positive definite matrix $\boldsymbol{K}$ at the ground mesh, respectively. It is well known that:

$$0 < \lambda_{min} \leq \frac{\boldsymbol{x}^T \boldsymbol{K} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \leq \lambda_{max} \quad \forall \boldsymbol{x} \in \mathbb{R}^n \tag{2.41}$$

where n is the number of degrees of freedom Golub and Loan (1996). This inequality holds for all $\boldsymbol{x} \in \mathbb{R}^n$ including the standard basis vector $\boldsymbol{e}_i$ corresponding to the i^{th} shadow dof as well as any orthonormal rigid body mode $\boldsymbol{v}_i$.

$$0 < \lambda_{min} \leq \boldsymbol{e}_i^T \boldsymbol{K} \boldsymbol{e}_i = \sum_e K_{e,ii} \leq \lambda_{max} \tag{2.42}$$

$$0 < \lambda_{min} \leq \boldsymbol{v}_i^T \boldsymbol{K} \boldsymbol{v}_i \leq \lambda_{max} \tag{2.43}$$

Additionally, since each element stiffness matrix $\boldsymbol{K}_e$ is positive semi-definite, $\boldsymbol{K}_{f_i}$, $\boldsymbol{K}_{p_i}$ and $\boldsymbol{K}_{v_i}$ are all positive semi-definite with a non-negative quadratic form. There-

fore:

$$\boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i \leq \boldsymbol{v}_i^T \boldsymbol{K}_{f_i} \boldsymbol{v}_i + \boldsymbol{v}_i^T \boldsymbol{K}_{p_i} \boldsymbol{v}_i + \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i = \boldsymbol{v}_i^T \boldsymbol{K} \boldsymbol{v}_i \leq \lambda_{max} \tag{2.44}$$

This completes the proof. □

Let $LB = \sum_{i \in I} \frac{f_i^2}{x_{min} \times \sum_e K_{e,ii}} + \sum_{i \in J} \frac{(\boldsymbol{v}_i^T \boldsymbol{f})^2}{x_{min} \times \boldsymbol{v}_i^T \boldsymbol{K}_{v_i} \boldsymbol{v}_i}$. If $x_{min} = \frac{\epsilon}{\lambda_{max}}$:

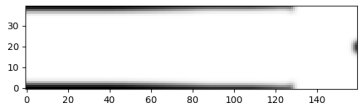
$$LB \geq \sum_{i \in I} \frac{f_i^2}{\epsilon} + \sum_{i \in J} \frac{(\boldsymbol{v}_i^T \boldsymbol{f})^2}{\epsilon} \tag{2.45}$$

$$C \gtrapprox \frac{1}{\epsilon} \left(\sum_{i \in I} f_i^2 + \sum_{i \in J} (\boldsymbol{v}_i^T \boldsymbol{f})^2 \right) \tag{2.46}$$

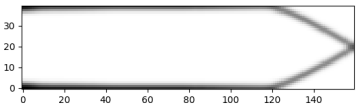
Therefore, penalization will not be undermined by the scale of $\boldsymbol{K}$. Choosing an appropriate value of ϵ is not an easy task. It needs to be small enough so that designs in the neighbourhood with one of the 2 penalized cases have a worse compliance than other feasible designs, in the exact sense of feasibility. A future research direction can be on efficient ways to choose and adapt the value of ϵ. Figure 2-11 shows the final continuous design of Dec-Tol CSIMP without penalty adaptation using $x_{min} = 0.01$ and $x_{min} = 0.001$, for volume fraction $V = 0.1$. It is clear that the second infeasibility case is present. Note that unlike x_{min}, ϵ is not a dimensionless quantity. ϵ has the same unit as the elements of the stiffness matrix, i.e. N/mm.

Table 2.5: This table shows the effect of increasing x_{min} on CSIMP with decreasing tolerance and a fixed penalty step. The table shows the results of the experiments studying the effect of increasing x_{min}: 1) the number of FEA simulations (Sims) required by CSIMP, 2) the final objective of the continuous NLP at $p = 5$ (C-Obj), 3) the objective value of the rounded discrete solution (D-Obj), and 4) the percentage of fractionness (% frac). The parameters of each experiment are described in Table 2.3.

Exp	$x_{min} = 0.001$			**% Increase -** $x_{min} = 0.01$		
	Sims	**C-Obj / D-Obj**	**% frac**	**Sims**	**C-Obj / D-Obj**	**% frac**
1	185	814.2/581.6	17.3	-23.8	-16.4/-4.2	-18.5
2	316	720.4/553.1	14.0	-29.7	-5.9/2.8	0.0
3	139	375.0/342.3	11.5	-15.1	-3.3/-1.3	-12.2
4	236	367.3/338.9	9.8	-23.3	-1.5/-0.1	2.0
5	130	619.6/293.9	24.6	-7.7	-13.9/14.0	-0.8
6	260	606.5/292.8	24.3	-21.2	-12.4/9.6	0.0
7	130	219.5/175.3	18.1	-7.7	-2.9/1.0	-0.6
8	211	218.8/175.1	17.9	-6.6	-2.6/1.1	1.1
9	153	121.5/103.4	9.0	-20.9	-3.0/1.3	1.1
10	226	123.5/104.1	9.6	-14.6	-4.7/0.5	-5.2
11	142	69.8/65.6	6.8	-26.8	-0.9/0.5	-4.4
12	192	69.9/65.7	6.8	-12.0	-1.1/0.2	-2.9
13	97	16.1/13.1	10.1	-1.0	-1.9/0.0	2.0
14	161	16.0/13.1	10.1	0.0	-1.2/0.0	1.0
15	97	10.2/9.3	10.9	0.0	-1.0/0.0	-0.9
16	161	10.2/9.3	11.0	0.0	-1.0/0.0	-0.9



(a) $x_{min} = 0.01$.



(b) $x_{min} = 0.001$.

Figure 2-11: 2D cantilever problem using Dec-Tol CSIMP without penalty adaptation using $V = 0.1$, $\Delta p = 0.05$.

2.9 Conclusion

In this chapter, a penalty step adaptation technique for the continuation SIMP algorithm was proposed and tested. Four common test problems from literature, three 2D and one 3D, were used to test the efficacy of the penalty adaptation with different parameter settings. The main factors affecting the efficacy of the penalty adaptation in the CSIMP algorithm in reducing the number of FEA simulations needed to converge to the final solution were identified. The experimental results demonstrate a significant reduction in the number of FEA simulations required to reach the optimal solution in the decreasing tolerance continuation SIMP algorithm, with exponentially decaying tolerance, with little to no detriment in the objective value and the other metrics used. Finally, a mathematical and experimental treatment of the effect of x_{min} on the convergence of the SIMP algorithm was given with some recommendations for choosing a suitable x_{min}.

3. Exact compliance-based optimization with finitely many loading scenarios

This work is accepted for publication in the Journal of Structural and Multidisciplinary Optimization. The accepted manuscript can be found on arXiv (https://arxiv.org/abs/2103.04594). The paper is titled: "Robust and stochastic compliance-based topology optimization with finitely many loading scenarios" (Tarek and Ray, 2021)

3.1 Introduction

Handling load uncertainty naively in topology optimization can prohibitively increase the computational cost of topology optimization. In this chapter, efficient methods to handle load uncertainty will be reviewed and novel ones will be proposed for compliance-based topology optimization.

3.2 Literature review

3.2.1 Mean compliance minimization

A number of works in literature tackled the problem of load uncertainty in compliance minimization problems. A general scheme for mean compliance minimization subject to uncertain load components with arbitrary covariance was proposed by Guest and Igusa (2008). Guest et al. also considered uncertain load locations by treating the loaded nodes' locations as random and hence the stiffness matrix. Guest et al. then derived an approximately equivalent load distribution that would result in the same mean compliance while deterministically using the mean stiffness matrix. Mean compliance minimization subject to uncertainty of concentrated load magnitude and direction was studied by Dunning et al. (2011). Dunning et al. derived efficient formulations for the mean compliance subject to such uncertainties, where the number of linear systems to be solved scales linearly with the number of independent random variables. This assumes that the distribution of the forces' magnitudes

and rotations are known and independent. A similar formulation was then derived by Zhao and Wang (2014*b*) which requires fewer linear system solves per independent random variable. Zhao et al. also demonstrated how the same approach can handle distributed load uncertainty, modeled as a stochastic field, using Karhunen-Loeve (K-L) expansion. The main limitation of the K-L expansion method in practice is having to define and commit to a covariance kernel function. Additionally, the more local the covariance is, the higher the number of terms required in the K-L expansion to accurately describe the random field. The number of linear system solves required to compute the mean compliance scales with the number of terms in the K-L expansion.

Zhang et al. (2017) proposed an efficient way to compute the mean compliance for finitely many load scenarios with no assumptions about the nature of randomness. This can be used in data-driven design where the loading scenarios are many, and for which data has been collected. Alternatively, loading scenarios may be sampled from the distributions assumed if sufficiently low dimensional. Zhang et al. developed a randomized algorithm inspired by Hutchinson's trace estimator Hutchinson (1990) to minimize the mean compliance: $\frac{1}{N}\sum_i^N \boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i$, leading to significant computational savings compared to the naive approach. The algorithm can be trivially modified to handle weighted mean compliance which can be used in cases where the number of random variables are many following the approach by Zhao and Wang (2014*b*) or when the number of terms of the K-L expansion is high. The main limitation of this approach is that it can only be used to minimize the mean compliance which is not risk-averse since the compliance can be very high for some

probable load scenarios. Finally, Liu and Wen (2018) solved the mean compliance minimization problem under a volume constraint subject to load uncertainty modeled using the fuzzy set theoretic cloud model for describing uncertainty.

3.2.2 Risk-averse compliance minimization

Beside mean compliance minimization, some authors also studied risk-averse compliance minimization by considering the weighted sum of the mean and variance, the weighted sum of the mean and standard deviation, as well as other risk measures. Dunning and Kim (2013) derived an efficient formulation for the variance of the compliance subject to uncertainty in the load magnitude. This was used to minimize a weighted sum of the mean and variance of the compliance. The main limitation of this approach is that it cannot handle random load directions and that the distributions for the load magnitudes are assumed to be known and independent. Zhao and Wang (2014*a*) minimized the weighted sum of the mean and standard deviation of the compliance by deriving an efficient formulation for the calculation of the mean and standard deviation of the compliance and their gradients under uncertain loading modeled as a random field. Zhao et al. used K-L expansion to quantify and describe the randomness using a few random variables, and used Monte Carlo simulation to calculate the fourth moment of those random variables which is required for the efficient computation of the standard deviation of the compliance.

Chen et al. (2010) used the level-set method to minimize the weighted sum of the mean compliance and its variance subject to uncertainty in the load and

material properties. Chen et al. assumed the load and material properties to be random fields and used K-L expansion to reduce their dimensionality. The authors then used Gaussian quadrature sampling to generate a representative set of scenarios to formulate the mean and variance of the compliance. Martínez-Frutos and Herrero-Pérez (2016) used a similar uncertainty quantification approach to develop a multi-GPU density-based topology optimization framework for the large-scale minimization of the weighted sum of the mean compliance and its variance subject to load uncertainty only. However, Martinez-Frutos et al. used sparse grid sampling instead of Gaussian quadrature. Similarly, Cuellar et al. (2018) used K-L expansion for uncertainty quantification and Gaussian quadrature for sampling, and combined them with the non-intrusive polynomial chaos expansion (PCE) method to provide more accurate estimators for the mean and standard deviation of the compliance and their gradients. Martínez-Frutos et al. (2018) also used K-L expansion and the non-intrusive PCE with sparse grid sampling for the quantification and propagation of the uncertainty in the load and material properties. However, Martinez-Frutos et al. minimized a different compliance risk measure called the *excess probability*, which is the probability that the compliance exceeds a certain threshold value. Note that in all the works above which use K-L expansion and sampling-based uncertainty propagation, the number of linear system solves can be made independent from the number of sampling points given the linearity assumption of the displacement as a function of the load exploited by Zhao and Wang (2014*a*) in their derivation, even though in some of the works this property was not exploited. The number of linear system solves can therefore be assumed to be equal to the number of terms in the

K-L expansion only, not the sampling points.

Also in risk-averse topology optimization, Garcia-Lopez et al. (2013) used multi-objective evolutionary optimization to optimize the mean and variance of the compliance and obtain the Pareto front of the two objectives. Garcia-Lopez et al. used a sampling method for uncertainty propagation inspired from Taguchi's method for the design of experiments. In this case, the number of linear system solves is equal to the number of sampling points. That beside the use of an evolutionary algorithm which requires many evaluations of the mean and variance of the compliance make the computational cost of this approach extremely high even for medium-sized problems. Finally, Kriegesmann and Lüdeker (2019) used FOSM instead of sampling to efficiently propagate the uncertainty estimating the mean and standard deviation of the compliance and their gradients from the means and standard deviations of the loads. A weighted sum of the mean and standard deviation of the compliance was then minimized. This approach assumes that the compliance is a linear function of the random load centered at the MPP load, an assumption which leads to a prediction error in the mean and standard deviation of the compliance.

3.2.3 Probabilistic constraints and reliability-based topology optimization

Keshavarzzadeh et al. (2017) solved the problem of volume minimization subject to a probabilistic compliance constraint. In one case, the authors constrained the mean compliance plus a multiple of its standard deviation which is equivalent to a

reliability constraint assuming the compliance is normally distributed. In another, a reliability constraint was used such that the probability that the compliance exceeds a threshold value is constrained. Keshavarzzadeh et al. used the non-intrusive PCE and regularized Heaviside function to approximate the compliance reliability constraint and its gradient. PCE was also used to estimate the mean and standard deviation of the compliance and their gradients.

Beside sampling-based uncertainty propagation, RBDO offers a number of techniques for efficient, approximate uncertainty propagation which can be used for handling probabilistic constraints involving compliance or otherwise. Kharmanda and Olhoff (2002); Kharmanda et al. (2004) proposed the use of RBDO for topology optimization, also known as reliability-based topology optimization (RBTO), to handle probabilistic constraints due to random loads, geometry and material properties. Jung and Cho (2004) used FORM's PMA with SIMP topology optimization method to solve a volume minimization problem with a reliability constraint for geometrically nonlinear structures. The works by Kharmanda et al. and Jung et al. inspired other works such as Kim et al. (2006) who used FORM's RIA and PMA with SIMP to solve volume minimization problems with reliability constraints on the displacement and natural frequency of the structure under loading, material and geometry uncertainties. Another group, Kim et al. (2007, 2008), later used RIA and PMA together with evolutionary structural optimization (ESO) (Xie and Steven, 1992; Yang et al., 1998; Huang and Xie, 2010) to solve volume minimization problems with a reliability constraint subject to a random load and Young's modulus. Ouyang et al. (2008) used FORM's RIA with the level-set method to solve a com-

pliance minimization problem with a reliability constraint subject to uncertainty in the load and geometry of the ground mesh.

Silva et al. (2010) proposed the use of an efficiently obtainable approximate MPP to avoid the need for solving the reliability or inverse reliability problems in every design iteration of RIA or PMA, respectively. Silva et al. and later Nguyen et al. (2011) also considered system reliability-based topology optimization where an aggregated system failure probability is considered instead of component failure probabilities and component limit state functions. Zhao et al. (2016) presented a comparison of a number of RBTO approaches to solve a few topology optimization problems including one with a compliance reliability constraint under stochastic load and Young's modulus. Jalalpour and Tootkaboni (2016) developed a bi-directional ESO (BESO) (Xie and Steven, 1992; Yang et al., 1998; Huang and Xie, 2010) algorithm for handling reliability constraints with displacement limit state functions and a finite number of probable loading scenarios in linearly elastic structures. Finally, Yin et al. (2018) proposed an alternative RBTO approach using fuzzy set theory to describe the uncertainty.

3.2.4 Maximum compliance constraint

Beside the stochastic and risk-averse topology optimization presented above, a number of works also studied maximum compliance minimization and maximum compliance constrained problems under uncertain loading conditions, where the former can be formulated using the latter as shown earlier. In these papers, no probability

distribution is assumed for the uncertain load and therefore they fall under the category of RO. Most of the work on handling non-probabilistic uncertainty in loads assumed that the loads lie in continuous sets. Brittain et al. (2012) for example assumed that the load vector has a fixed norm but arbitrary direction. Brittain et al. used a bi-level min-max optimization approach minimizing the objective with respect to the topology variables in the upper level problem, and maximizing with respect to the load in the lower level problem. However, an efficient algorithm was derived for the lower level maximization problem based on the KKT optimality conditions for the objective and the load's fixed-norm constraint. Holmberg et al. (2015) proposed a nonlinear semidefinite formulation to solve the maximum compliance minimization problem under non-probabilistic load uncertainty, assuming the uncertain load lies in a hyper-ellipsoid uncertainty set:

$$\boldsymbol{f} = \boldsymbol{f}_0 + \boldsymbol{Q}\boldsymbol{r} \tag{3.1}$$

$$||\boldsymbol{r}|| \leq 1 \tag{3.2}$$

where $\boldsymbol{f}_0$ is the fixed component of the load vector and the columns of $\boldsymbol{Q}$ are the axes of the uncertainty hyper-ellipsoid. The same authors, Thore et al. (2017), later generalized their approach from Holmberg et al. (2015) to handle maximum compliance and maximum stress constraints under the same assumption on the load vector. Similarly, Liu and Gea (2018) proposed a bi-level formulation for handling multiple independent loads, each of which lies in a hyper-ellipsoidal uncertainty set. Liu et al. developed an efficient lower level algorithm by solving the Wolfe

dual problem. The Wolfe dual problem of the lower level problem is a maximum generalized eigenvalue minimization problem which was solved using an iterative procedure. The multi-ellipsoidal uncertainty set generalizes the interval as well as the spherical uncertainty sets.

A number of papers were also published on non-probabilistic reliability-based topology optimization (NRBTO) where new reliability indexes and performance measures are defined for various types of continuous uncertainty sets. While some of these works did not solve problems with maximum compliance constraints, but the same techniques can be applied to handle maximum compliance constraints. Luo et al. (2009) proposed a reliability index and performance measure for handling non-probabilistic uncertainty where the uncertain variables lie in a multi-ellipsoid model. Another reliability index for ellipsoidal uncertainty was then proposed by Wang et al. (2018). Wang et al. (2017); Wang, Xia, Zhang and Lv (2019) proposed another reliability index for handling interval uncertainty sets using interval arithmetic. Zheng et al. (2018) proposed a reliability index and performance function for uncertain variables in multidimensional parallelepiped convex sets. Another reliability index for mixed interval and ellipsoidal uncertainty was also developed by Wang, Liu, Yang and Hu (2019).

The rest of this chapter is organized as follows. The proposed approaches for handling load uncertainty in continuum compliance problems in the form of a large, finite number of loading scenarios are detailed in sections 4.2.1, 4.2.2 and 3.5. The experiments used and the implementations are then described in section 4.3. Finally, the results are presented and discussed in section 3.7 before concluding in section

4.7.

3.3 Compliance sample mean and its gradient

3.3.1 Naive approach

The compliance sample mean for a finite number L of loading scenarios is $\mu_C = \frac{1}{L}\sum_{i=1}^{L} \boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i$ where $\boldsymbol{f}_i$ is the i^{th} load scenario, $\boldsymbol{K}$ is the stiffness matrix of the design and $\boldsymbol{F}$ is the matrix whose columns are the individual loading scenarios $\boldsymbol{f}_i$. The direct naive approach is to solve for $\boldsymbol{K}^{-1}\boldsymbol{f}_i$ for all i and calculate the mean compliance using the above formula. This method is not efficient since it requires L linear system solves plus some additional work to compute the mean with a time complexity of $O(L \times n_{dofs})$, where n_{dofs} is the number of degrees of freedom in the design. When $\boldsymbol{F}$ is sparse with only a few n_{loaded} degrees of freedom that are loaded, the complexity of the remaining work to compute the mean compliance $\frac{1}{L}\sum_{i=1}^{L} \boldsymbol{f}_i^T \boldsymbol{u}_i$ becomes $O(L \times n_{loaded})$. Even though the factorization of $\boldsymbol{K}^{-1}$ can be reused to solve for the L linear systems, if L is close to n_{dofs}, the complexity of solving for so many linear systems will be similar to that of the factorization, thus significantly adding to the running time. When using an iterative algorithm to solve for $\boldsymbol{K}^{-1}\boldsymbol{f}_i$, a good, but expensively formed, preconditioner such as the algebraic multi-grid preconditioner can be similarly reused. In general, significantly reducing the number of linear systems to solve is advantageous in practice even if, as theory may show, the running time is dominated by the initial linear system solve.

Let the Jacobian of $\boldsymbol{\rho}(\boldsymbol{x})$ be $\nabla_{\boldsymbol{x}}\boldsymbol{\rho}(\boldsymbol{x})$. Let $\boldsymbol{u}_i$ be the displacement response due to load $\boldsymbol{f}_i$ and C_i be the compliance $\boldsymbol{f}_i^T\boldsymbol{u}_i$. The stiffness matrix $\boldsymbol{K}$ is typically defined as: $\boldsymbol{K} = \sum_e \rho_e \boldsymbol{K}_e$. The partial derivative of the compliance C_i with respect to ρ_e is given by $\frac{\partial C_i}{\partial \rho_e} = -\boldsymbol{u}_i^T \boldsymbol{K}_e \boldsymbol{u}_i$. The gradient of C_i with respect to the decision vector $\boldsymbol{x}$ is therefore given by: $\nabla_{\boldsymbol{x}} C_i(\boldsymbol{x}) = \nabla_{\boldsymbol{x}}\boldsymbol{\rho}(\boldsymbol{x})^T \nabla_{\boldsymbol{\rho}} C_i(\boldsymbol{\rho}(\boldsymbol{x}))$ where $\nabla_{\boldsymbol{\rho}} C_i(\boldsymbol{\rho}(\boldsymbol{x}))$ is the gradient of C_i with respect to $\boldsymbol{\rho}$ at $\boldsymbol{\rho}(\boldsymbol{x})$. The gradient of the mean compliance μ_C is therefore given by $\nabla_{\boldsymbol{x}}\mu_C(\boldsymbol{x}) = \frac{1}{L}\sum_{i=1}^{L} \nabla_{\boldsymbol{x}}\boldsymbol{\rho}(\boldsymbol{x})^T \nabla_{\boldsymbol{\rho}} C_i(\boldsymbol{\rho}(\boldsymbol{x}))$. The additional complexity of computing the mean compliance and its gradient with respect to $\boldsymbol{\rho}$ is $O(n_E \times L)$. Note that the Jacobian of $\boldsymbol{\rho}(\boldsymbol{x})$ does not need to be formed explicitly to compute the gradient above, so long as there is a way to pre-multiply the Jacobian's transpose by a vector. The problem with the naive approach is it requires many linear system solves and so doesn't scale well to many loading scenarios.

3.3.2 Singular value decomposition

Less naively, one can first attempt to find the singular value decomposition (SVD) of $\boldsymbol{F}$. Let the compact SVD of the matrix $\boldsymbol{F}$ be $\boldsymbol{F} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$, where the number of non-zero singular values is n_s, $\boldsymbol{S}$ is the diagonal matrix of singular values, $\boldsymbol{U}$ is a $n_{dofs} \times n_s$ matrix with orthonormal columns, and $\boldsymbol{V}$ is $L \times n_s$ matrix with orthonormal columns. Given the SVD, the mean compliance can be written as:

$\mu_C = \frac{1}{L}\sum_{i=1}^{L} \boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i = \frac{1}{L} tr(\boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F})$. This can be further simplified:

$$\frac{1}{L} tr(\boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F}) = \frac{1}{L} tr(\boldsymbol{V}\boldsymbol{S}\boldsymbol{U}^T \boldsymbol{K}^{-1} \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T) \tag{3.3}$$

$$= \frac{1}{L} tr(\boldsymbol{S}\boldsymbol{U}^T \boldsymbol{K}^{-1} \boldsymbol{U}\boldsymbol{S}) \tag{3.4}$$

$$= \frac{1}{L} \sum_{i=1}^{n_s} \boldsymbol{S}[i,i]^2 \times \boldsymbol{U}[:,i]^T \boldsymbol{K}^{-1} \boldsymbol{U}[:,i] \tag{3.5}$$

This method requires only n_s linear system solves and an SVD. n_s will be small if the loads in $\boldsymbol{F}$ are highly correlated or if only a few degrees of freedom are loaded, i.e. the loads are sparse. Let n_{loaded} be the few loaded degrees of freedom. It is possible to prove in this case that the number of singular values $n_s \leq n_{loaded}$. The computational time complexity of computing the SVD of $\boldsymbol{F}$ in the dense case is $O(min(L, n_{dofs})^2 max(L, n_{dofs}))$, while in the sparse case it is only $O(n_{loaded}^2 L)$. If n_{loaded} is a small constant, finding the SVD will be very efficient. Additionally, when only n_{loaded} degrees of freedom are loaded in $\boldsymbol{F}$, only the same degrees of freedom will be non-zero in $\boldsymbol{U}$, therefore $\boldsymbol{U}$ will also be sparse. Other than the complexity of SVD, the additional work to compute the mean compliance has a computational time complexity of $O(n_s \times n_{dofs})$ when $\boldsymbol{F}$ (and $\boldsymbol{U}$) are dense, and $O(n_s \times n_{loaded})$ when $\boldsymbol{F}$ (and $\boldsymbol{U}$) are sparse.

Given the efficient formula for the mean compliance and using the derivative rule of the inverse quadratic from appendix section .1, the partial $\frac{\partial \mu_C}{\partial \rho_e}$ is given by: $-\frac{1}{L}\sum_{i=1}^{n_s} \boldsymbol{S}[i,i]^2 (\boldsymbol{K}^{-1}\boldsymbol{U})[:,i]^T \boldsymbol{K}_e (\boldsymbol{K}^{-1}\boldsymbol{U})[:,i]$. The time complexity of computing this assuming we already computed $\boldsymbol{K}^{-1}\boldsymbol{U}$ is $O(n_s \times n_E)$.

Table 3.1: Summary of the computational cost of the algorithms discussed to calculate the mean compliance and its gradient. #Lin is the number of linear system solves required.

Method	#Lin	SVD?	Time complexity of additional work	
			Dense	Sparse
Exact-Naive	L	✗	$O(L \times (n_{dofs} + n_E))$	$O(L \times (n_{loaded} + n_E))$
Exact-SVD	n_s	✓	$O(n_s \times (n_{dofs} + n_E))$	$O(n_s \times (n_{loaded} + n_E))$

3.4 Scalar-valued function of load compliances and its gradient

In this section, the above approach for computing the sample mean compliance will be generalized to handle the sample variance and standard deviations. The sample variance of the compliance C is given by $\sigma_C^2 = \frac{1}{L-1}\sum_{i=1}^{L}(C_i - \mu_C)^2$. The sample standard deviation σ_C is the square root of the variance. Let $\boldsymbol{C}$ be the vector of compliances C_i, one for each load scenario. In vector form, $\sigma_C^2 = \frac{1}{L-1}(\boldsymbol{C} - \mu_C\mathbf{1})^T(\boldsymbol{C} - \mu_C\mathbf{1})$. $\boldsymbol{C} = diag(\boldsymbol{A})$ is the diagonal of the matrix $\boldsymbol{A} = \boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}$.

3.4.1 Naive approach

If one can compute the vector of load compliances $\boldsymbol{C}$, computing the variance and standard deviation is trivial. This approach requires L linear system solves which can be computationally prohibitive if L is large. Other than the linear system solves, the remaining work of computing $C_i = \boldsymbol{f}_i^T\boldsymbol{u}_i$ for all i has a complexity of $O(L \times n_{dofs})$ when $\boldsymbol{F}$ is dense and $O(L \times n_{loaded})$ when $\boldsymbol{F}$ is sparse with only

n_{loaded} loaded degrees of freedom. In order to compute the vector $\boldsymbol{C}$ exactly, one needs to compute $\boldsymbol{u}_i = \boldsymbol{K}^{-1}\boldsymbol{f}_i$ for all i. These can further be used to compute the gradients of the load compliances C_i which can be combined to form the Jacobian $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}$. Assuming $\boldsymbol{u}_i$ is cached for all i, the time complexity of computing the Jacobian using $\frac{\partial C_i}{\partial \rho_e} = -\boldsymbol{u}_i^T \boldsymbol{K}_e \boldsymbol{u}_i$ is $O(n_E \times L)$.

However, when interested in the gradient of a scalar-valued function f of $\boldsymbol{C}$, there is no need to form the full Jacobian $\nabla_{\boldsymbol{x}}\boldsymbol{C}(\boldsymbol{x})$. It suffices to define an operator to pre-multiply an arbitrary vector $\boldsymbol{w}$ by $\nabla_{\boldsymbol{x}}\boldsymbol{C}(\boldsymbol{x})^T$. Using the chain rule, the gradient of f with respect to $\boldsymbol{x}$ is given by $\nabla_{\boldsymbol{x}} f(\boldsymbol{C}(\boldsymbol{x})) = \nabla_{\boldsymbol{x}}\boldsymbol{C}(\boldsymbol{x})^T \nabla_{\boldsymbol{C}} f(\boldsymbol{C}(\boldsymbol{x}))$. This operator is equivalent to attempting to find the gradient of the weighted sum of $\boldsymbol{C}$, $\boldsymbol{w}^T\boldsymbol{C}$, where $\boldsymbol{w}$ is the constant vector of weights. In the case of a general scalar-valued function f, $\boldsymbol{w}$ would be $\nabla_{\boldsymbol{C}} f(\boldsymbol{C}(\boldsymbol{x}))$ and is treated as a constant. In the case of the variance, $f(\boldsymbol{C}) = \sigma_C^2 = \frac{1}{L-1}(\boldsymbol{C}-\mu_C\boldsymbol{1})^T(\boldsymbol{C}-\mu_C\boldsymbol{1})$, therefore $\boldsymbol{w} = \nabla_{\boldsymbol{C}} f(\boldsymbol{C}(\boldsymbol{x})) = \frac{2}{L-1}(\boldsymbol{C}-\mu_C\boldsymbol{1})$. And in the case of the standard deviation σ_C, $\boldsymbol{w} = \frac{1}{(L-1)\sigma_C}(\boldsymbol{C} - \mu_C\boldsymbol{1})$. This means that computing $\boldsymbol{C}$ is required to form $\boldsymbol{w}$.

By caching $\boldsymbol{u}_i = \boldsymbol{K}^{-1}\boldsymbol{f}_i$ for all i when computing $\boldsymbol{C}$, one can find the e^{th} element of $\nabla_{\boldsymbol{x}}\boldsymbol{C}(\boldsymbol{x})^T\boldsymbol{w}$ using $\sum_{i=1}^{L} -w_i \boldsymbol{u}_i^T \boldsymbol{K}_e \boldsymbol{u}_i$, where w_i is i^{th} element of $\boldsymbol{w}$. Computing $\boldsymbol{u}_i^T \boldsymbol{K}_e \boldsymbol{u}_i$ requires constant time complexity, therefore the additional time complexity of computing $\nabla_{\boldsymbol{x}}\boldsymbol{C}(\boldsymbol{x})^T\boldsymbol{w}$ after computing $\boldsymbol{C}$ with the direct method is $O(L \times n_E)$. In this case, this is the same complexity as forming the Jacobian first and then multiplying, but in the next algorithms, it will be different.

3.4.2 Singular value decomposition

Much like in the mean compliance calculation, the SVD of $\boldsymbol{F}$ can be computed to find C_i for all i more efficiently from $\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$. The number of linear system solves required to compute $\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$ is n_s, the number of singular values of $\boldsymbol{F}$. The computational cost of computing $C_i = \boldsymbol{f}_i^T\boldsymbol{u}_i = \boldsymbol{f}_i^T(\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S})V^T[:,i]$ for all i using $\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$ and $\boldsymbol{V}$ is $O(L \times n_s \times n_{dofs})$ when $\boldsymbol{F}$ is dense and $O(L \times n_s \times n_{loaded})$ when $\boldsymbol{F}$ is sparse with only n_{loaded} degrees of freedom loaded. The Jacobian $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}$ can be built by first computing $\boldsymbol{K}^{-1}\boldsymbol{F}$ from the cached $\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$ then using it much like in the exact method without SVD. This has a time complexity of $O((n_s \times n_{dofs} + n_E) \times L)$.

However, when interested in $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w}$ instead, a more efficient approach can be used. Let $\boldsymbol{D}_{\boldsymbol{w}}$ be the diagonal matrix with the vector $\boldsymbol{w}$ on the diagonal.

$$\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w} = \nabla_{\boldsymbol{\rho}}(\boldsymbol{C}^T\boldsymbol{w}) = \nabla_{\boldsymbol{\rho}} tr(\boldsymbol{D}_{\boldsymbol{w}}\boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}) \tag{3.6}$$

$$= \nabla_{\boldsymbol{\rho}} tr(\boldsymbol{V}^T\boldsymbol{D}_{\boldsymbol{w}}\boldsymbol{V}\boldsymbol{S}\boldsymbol{U}^T\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}) \tag{3.7}$$

Let $\boldsymbol{X} = \boldsymbol{V}^T\boldsymbol{D}_{\boldsymbol{w}}\boldsymbol{V}$ and $\boldsymbol{Q} = \boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$. The partial derivative of the above with respect to ρ_e is:

$$\frac{\partial}{\partial \rho_e} tr(\boldsymbol{X}\boldsymbol{Q}^T\boldsymbol{S}\boldsymbol{U}^T\boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}) = -tr(\boldsymbol{X}\boldsymbol{Q}^T\boldsymbol{K}_e\boldsymbol{Q}) \tag{3.8}$$

Note that one can cache $\boldsymbol{Q} = \boldsymbol{K}^{-1}\boldsymbol{U}\boldsymbol{S}$ when finding the function value above to be reused in the sensitivity analysis. Let $\boldsymbol{Y}_e = \boldsymbol{Q}^T\boldsymbol{K}_e\boldsymbol{Q}$. The trace above is

therefore $tr(\boldsymbol{X}\boldsymbol{Y}_e) = tr(\boldsymbol{X}^T\boldsymbol{Y}_e) = \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \boldsymbol{X}[i,j] \times \boldsymbol{Y}_e[i,j]$. Computing $\boldsymbol{Y}_e[i,j]$ from the pre-computed $\boldsymbol{Q}$ requires a constant time complexity for each element e, and computing $\boldsymbol{X}$ has a time complexity of $O(L \times n_s^2)$. The additional time complexity of computing $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w}$ using this method is therefore $O((n_E + L) \times n_s^2)$. So if $n_s \ll L$, significant computational savings can be made compared to directly computing the Jacobian then doing the matrix-vector multiplication $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w}$ which has a complexity of $O((n_s \times n_{dofs} + n_E) \times L)$.

Table 3.2: Summary of the computational cost of the algorithms discussed to calculate the load compliances $\boldsymbol{C}$ as well as $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w}$ for any vector $\boldsymbol{w}$. #Lin is the number of linear system solves required. This can be used to compute the variance, standard deviation as well as other scalar-valued functions of $\boldsymbol{C}$. If the full Jacobian is required, the naive method requires the same computational cost as that of computing $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}^T\boldsymbol{w}$, and the SVD-based method has a time complexity of $O((n_s \times n_{dofs} + n_E) \times L)$ for the additional work other than the linear system solves and SVD.

Method	#Lin	SVD?	Time complexity of additional work	
			Dense	Sparse
Exact-Naive	L	✗	$O(L \times (n_{dofs} + n_E))$	$O(L \times (n_{loaded} + n_E))$
Exact-SVD	n_s	✓	$O(L \times n_s \times n_{dofs} + (n_E + L) \times n_s^2)$	$O(L \times n_s \times n_{loaded} + (n_E + L) \times n_s^2)$

3.5 Maximum compliance constraint

The maximum compliance constraint can be efficiently handled using the augmented Lagrangian optimization algorithm (Bertsekas, 1996). Assume the following maximum compliance constrained problem is to be solved for some objective $g(\boldsymbol{x})$ using

the augmented Lagrangian algorithm:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad g(\boldsymbol{x}) \tag{3.9a}$$

subject to

$$C_i = \boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i \leq C_t \quad \forall i \in 1 \ldots L, \tag{3.9b}$$

$$0 \leq x_e \leq 1 \quad \forall e \in 1 \ldots n_E \tag{3.9c}$$

where C_t is the maximum compliance allowed. In the augmented Lagrangian algorithm, the problem is transformed as follows:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad L(\boldsymbol{x}; \boldsymbol{\lambda}, r) \tag{3.10a}$$

subject to

$$0 \leq x_e \leq 1 \quad \forall e \in 1 \ldots n_E \tag{3.10b}$$

$$L(\boldsymbol{x}; \boldsymbol{\lambda}, r) = g(\boldsymbol{x}) + \sum_{i=1}^{L} \left(\lambda_i (C_i - C_t) + r \max(C_i - C_t, 0)^2 \right) \tag{3.11}$$

where $\boldsymbol{\lambda}$ is the vector of Lagrangian multipliers λ_i, one for each compliance constraint, and r is the constant coefficient of the quadratic penalty. Solving the above problem using a first-order box constrained algorithm requires the gradient of $L(\boldsymbol{x})$. Writing $L(\boldsymbol{x})$ in vector form:

$$L(\boldsymbol{x}) = g(\boldsymbol{x}) + \boldsymbol{\lambda}^T (\boldsymbol{C} - C_t \boldsymbol{1}) + r \boldsymbol{M}' \boldsymbol{M} \tag{3.12}$$

where $\boldsymbol{M}$ is the vector whose i^{th} element is $max(C_i - C_t, 0)$. The gradient of $L(\boldsymbol{x})$ is given by:

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} g + \nabla_{\boldsymbol{x}} \boldsymbol{\rho}^T (\nabla_{\boldsymbol{\rho}} (\boldsymbol{\lambda}^T (\boldsymbol{C} - C_t \mathbf{1}) + r \boldsymbol{M}' \boldsymbol{M}) \tag{3.13}$$

$$= \nabla_{\boldsymbol{x}} g + \nabla_{\boldsymbol{x}} \boldsymbol{\rho}^T \nabla_{\boldsymbol{\rho}} \boldsymbol{C}^T (\boldsymbol{\lambda} + 2\boldsymbol{M}) \tag{3.14}$$

As shown in the previous sections, calculating the product $\nabla_{\boldsymbol{\rho}} \boldsymbol{C}^T (\boldsymbol{\lambda} + 2\boldsymbol{M})$ can be done efficiently by finding the gradient $\nabla_{\boldsymbol{\rho}} (\boldsymbol{C}^T \boldsymbol{w})$ using $\boldsymbol{w} = (\boldsymbol{\lambda} + 2\boldsymbol{M})$. Therefore, the results from Table 4.2 apply.

3.6 Setup and Implementation

In this section, the most important implementation details and algorithm settings used in the experiments are presented.

3.6.1 Test problems

3.6.1.1 2D cantilever beam

The 2D cantilever beam problem shown in Figure 4-1 was used to run the experiments. A ground mesh of plane stress quadrilateral elements was used, where each element is a square of side length 1 mm, and a sheet thickness of 1 mm. Linear iso-parametric interpolation functions were used for the field and geometric basis functions. A Young's modulus of 1 MPa and Poisson's ratio of 0.3 were used. Finally, a chequerboard density filter for unstructured meshes was used with a radius

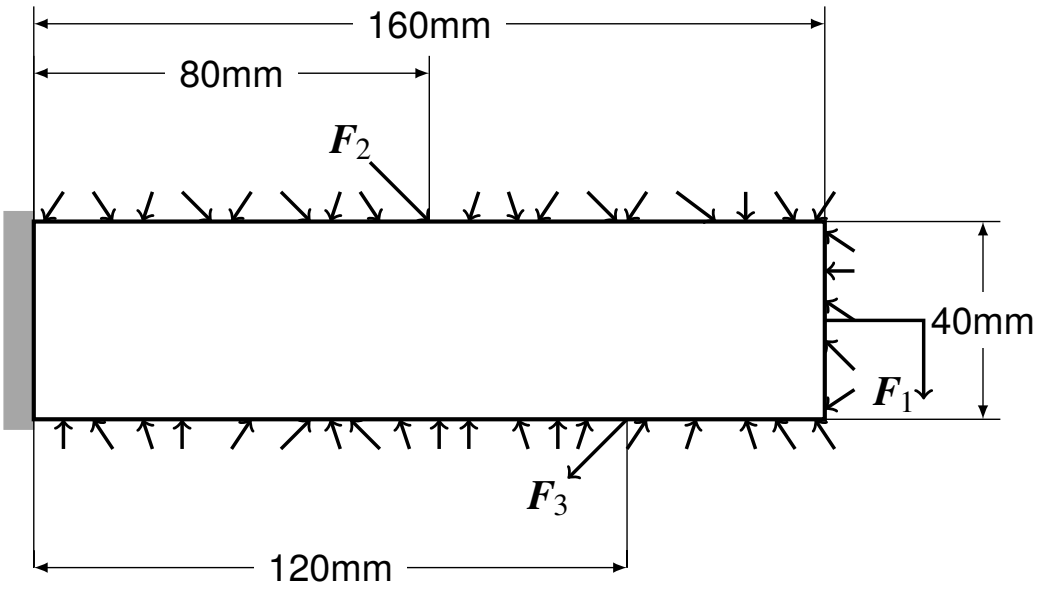


Figure 3-1: Cantilever beam problem. $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are at 45 degree angles.

of 2 mm (Huang and Xie, 2010). A 3D version of the problem above was also solved. Details of the 3D problem and the results are shown in appendix section .1.

Three variants of the cantilever beam problem were solved:

1. Minimization of the mean compliance μ_C subject to a volume constraint with a volume fraction of 0.4,

2. Minimization of a weighted sum of the mean and standard deviation (mean-std) of the compliance $\mu_C+2.0\sigma_C$ subject to a volume constraint with a volume fraction of 0.4, and

3. Volume minimization subject to a maximum compliance constraint with a compliance threshold of 70000 Nmm.

A total of 1000 load scenarios were sampled from:

$$\boldsymbol{f}_i = s_1\boldsymbol{F}_1 + s_2\boldsymbol{F}_2 + s_3\boldsymbol{F}_3 + \frac{1}{7}\sum_{j=4}^{10} s_j\boldsymbol{F}_j \tag{3.15}$$

where $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are unit vectors with directions as shown in Figure 4-1. $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are at 45 degrees. s_1, s_2 and s_3 are identically and independently uniformly distributed random variables between -2 and 2. $\boldsymbol{F}_j$ for j in $4 \ldots 10$ are vectors with non-zeros at all the surface degrees of freedom without a Dirichlet boundary condition. The non-zero values are identically and independently normally distributed random variables with mean 0 and standard deviation 1. s_j for j in $4 \ldots 10$ are also identically and independently normally distributed random variables with mean 0 and standard deviation 1. The same loading scenarios were used for the 3 test problems. Let $\boldsymbol{F}$ be the matrix whose columns are the sampled $\boldsymbol{f}_i$ vectors. The way the loading scenarios are defined, the rank of $\boldsymbol{F}$ can be at most 10 and was actually exactly 10 in our experiments. Given the low rank structure of $\boldsymbol{F}$, the SVD approaches should be expected to be significantly more efficient than their naive counterparts.

3.6.1.2 3D cantilever beam

A 3D version of the 2D cantilever beam test problem above was also solved using the methods proposed. A 60 mm x 20 mm x 20 mm 3D cantilever beam was used with hexahedral elements of cubic shape and side length of 1 mm. The loads $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ were positioned at (60, 10, 10), (30, 20, 10) and (40, 0, 10) where the

coordinates represent the length, height and depth respectively. The remaining loads and multipliers were sampled from the same distributions as the 2D problem. A density filter radius of 3 mm was also used for the 3D problem.

3.6.2 Settings

The value of x_{min} used was 0.001 for all problems and algorithms. Penalization was done prior to interpolation to calculate $\boldsymbol{\rho}$ from $\boldsymbol{x}$. A power penalty function and a regularized Heaviside projection were used. All of the problems were solved using 2 continuation SIMP routines. The first incremented the penalty value from $p = 1$ to $p = 6$ in increments of 0.5. Then the Heaviside projection parameter β was incremented from $\beta = 0$ to $\beta = 20$ in increments of 4 keeping the penalty value fixed at 6. An exponentially decreasing tolerance from $1e-3$ to $1e-4$ was used for both continuations.

The mean and mean-std compliance minimization SIMP subproblems problems were solved using the method of moving asymptotes (MMA) algorithm Svanberg (1987). MMA parameters of $s_{init} = 0.5$, $s_{incr} = 1.1$ and $s_{decr} = 0.7$ were used as defined in the MMA paper with a maximum of 1000 iterations for each subproblem. The dual problem of the convex approximation was solved using a log-barrier box-constrained nonlinear optimization solver, where the barrier problem was solved using the nonlinear CG algorithm for unconstrained nonlinear optimization (Nocedal and Wright, 2006) as implemented in Optim.jl [1] (K Mogensen and N Riseth, 2018). The nonlinear CG itself used the line search algorithm from Hager and Zhang (2006)

[1] https://github.com/JuliaNLSolvers/Optim.jl

as implemented in LineSearches.jl [2]. The stopping criteria used was the one adopted by the KKT solver, IPOPT (Wächter and Biegler, 2006). This stopping criteria is less scale sensitive than the KKT residual as it scales down the residual by a value proportional to the mean absolute value of the Lagrangian multipliers.

The maximum compliance constrained SIMP subproblems were solved using a primal-dual augmented Lagrangian method (Bertsekas, 1996). The inequality constraints were relaxed resulting in a box constrained max-min primal-dual problem. A projected gradient descent algorithm was used for the primal and dual problems with a backtracking line search. The maximum step size of the line search was initialized to 1 and adapted to be 1.5 the step size of the previous line search for both the primal and dual problems. A total of 10 dual iterations were used with a maximum of 50 primal iterations per dual iteration. The IPOPT termination criteria above was also used here. To regularize the scale of the problem, the compliance constraints were divided by the maximum compliance at the full ground mesh design. A trust region of 0.1 was used. An initial quadratic penalty coefficient of 0.1 was used with a growth factor of 3 in every dual iteration. Finally, an initial solution of 1.0 for all the primal variables and 1 for all the Lagrangian multipliers was used.

[2] https://github.com/JuliaNLSolvers/LineSearches.jl

3.7 Results and Discussion

3.7.1 Speed comparison

Tables 4.3 and 4.4 show the values computed for the mean compliance μ_C and its standard deviation σ_C respectively together with the time required to compute their values and gradients using: the exact naive approach (Exact-Naive) and the exact method with SVD (Exact-SVD). As expected, the proposed exact SVD approach computes the exact mean compliance or its standard deviation and their gradient in a small fraction of the time it takes to compute them using the naive approaches.

Table 3.3: The table shows the function values of μ_C computed using the naive exact method (Exact-Naive) and the exact method with SVD (Exact-SVD). The table also shows the time required to compute μ_C and its gradient in each case.

Method	μ_C (Nmm)	Time (s)
Exact-Naive	3328.7	24.2
Exact-SVD	3328.7	0.4

Table 3.4: The table shows the function values of σ_C and its gradients for a full ground mesh computed using the naive exact method (Exact-Naive) and the exact method with SVD (Exact-SVD). The table also shows the time required to compute σ_C and its gradient in each case.

Method	σ_C (Nmm)	Time (s)
Exact-Naive	4172.8	28.0
Exact-SVD	4172.8	1.5

3.7.2 Optimization

In this section, a number of stochastic, risk-averse and robust compliance-based optimization problems are solved using the proposed methods. Figure 3-2 shows

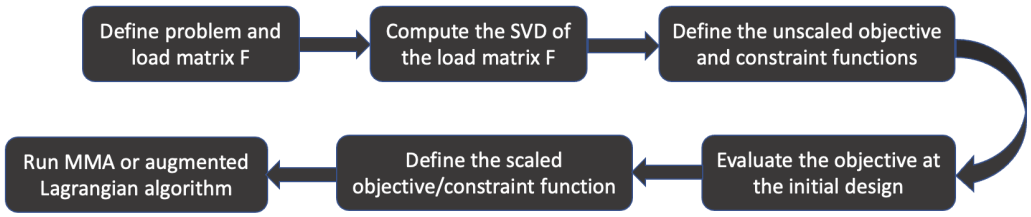


Figure 3-2: Flowchart of the experiments' workflow. Only the mean compliance objective, mean-std compliance objective or maximum compliance constraint are scaled by the inverse of their initial value. The volume function is not scaled.

the experiments' workflow.

3.7.2.1 Mean compliance minimization

To demonstrate the effectiveness of the proposed approaches, the 2D and 3D cantilever beam problems described in section 4.3 were solved using the proposed SVD-based methods. Table 4.5 shows the statistics of the final optimal solutions obtained by minimizing the mean compliance subject to the volume fraction constraint using the SVD-based method to evaluate the mean compliance. The optimal topologies are shown in Figures 4-5 and 4-12.

Table 3.5: Summary statistics of the load compliances of the optimal solutions of the 2D and 3D mean compliance minimization problems using the SVD-based method to evaluate the mean compliance.

Compliance Stat	2D	3D
μ_C (Nmm)	9392.8	22072.1
σ_C (Nmm)	9688.8	16628.7
C_{max} (Nmm)	125455.0	184055.0
C_{min} (Nmm)	467.9	1785.8
V	0.400	0.400
$Time$ (s)	491.5	3849.6

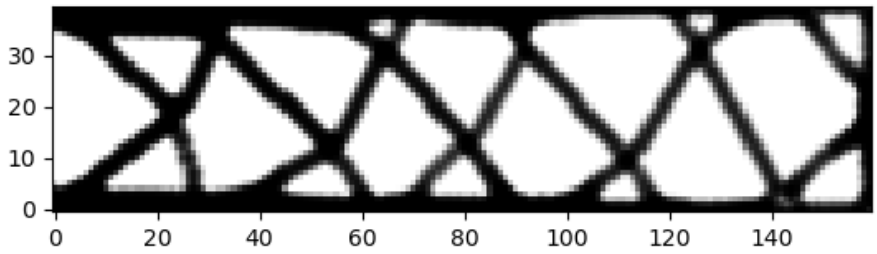
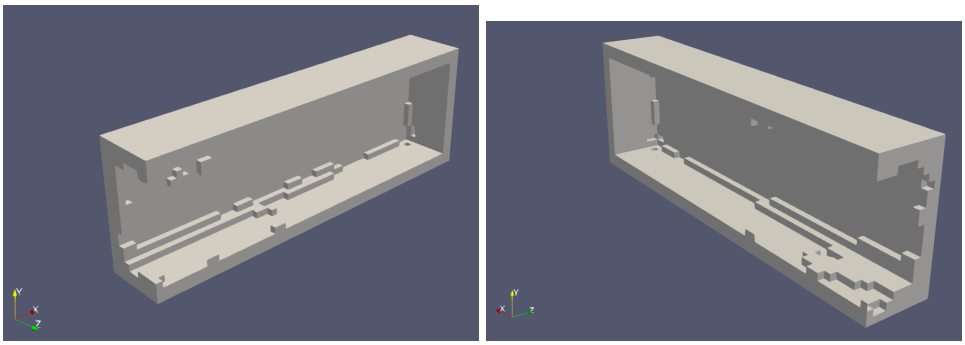


Figure 3-3: Optimal topology of the mean compliance minimization problem using continuation SIMP and the SVD-based method for evaluating the mean compliance.



(a) Left half (b) Right half

Figure 3-4: Cut views of the optimal topologies of the 3D mean compliance minimization problem using exact method with SVD.

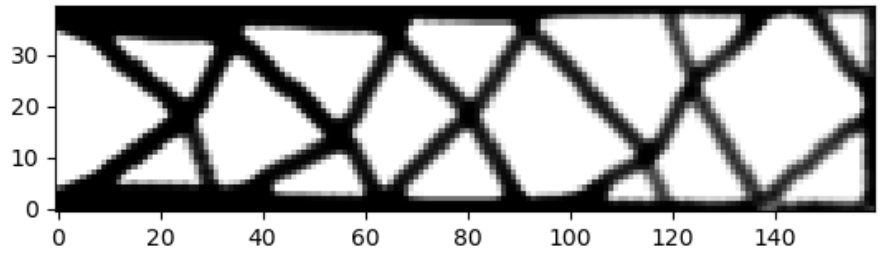
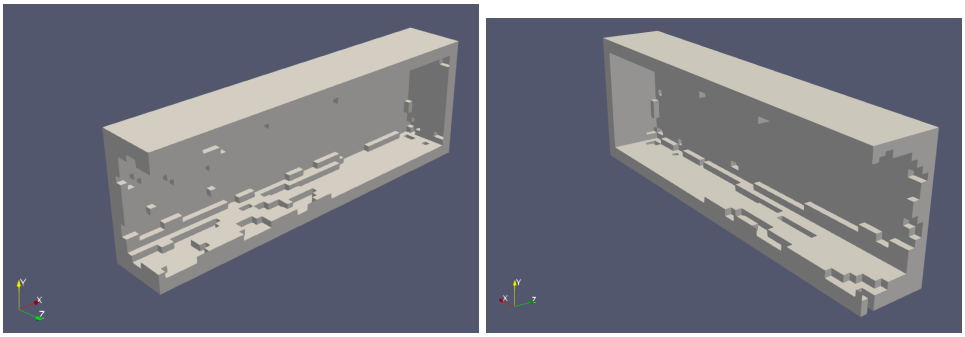


Figure 3-5: Optimal topology of the mean-std compliance minimization problem using continuation SIMP and the SVD-based method to compute the mean-std.

3.7.2.2 Mean-std compliance minimization

Similarly, Table 4.6 shows the statistics of the final solutions of the 2D and 3D mean-std minimization problems solved using the SVD-based method. The optimal topologies are shown in Figures 4-6 and 4-14. The algorithm converged to reasonable, feasible designs. Additionally, as expected the mean-std minimization algorithm converged to solutions with lower compliance standard deviations but higher mean compliances compared to the mean minimization algorithm.

To highlight the effect of the multiple m of the standard deviation in the objective $\mu_C + m \times \sigma_C$, the same problem was solved for different values of m. Figure 3-7 shows the profile of the mean and standard deviation of the compliance. Interestingly due



(a) Left half (b) Right half

Figure 3-6: Cut views of the optimal topologies of the 3D mean-std compliance minimization problem using the exact method with SVD.

Table 3.6: Summary statistics of the load compliances of the optimal solutions of the 2D and 3D mean-std compliance minimization problems using the SVD-based method to evaluate the mean-std compliance.

Compliance Stat	2D	3D
$\mu_C(Nmm)$	9796.9	22216.7
$\sigma_C(Nmm)$	9240.0	16220.2
$\mu_C + 2.0\sigma_C(Nmm)$	28283.7	54848.8
C_{max} (Nmm)	117883.1	176153.2
C_{min} (Nmm)	527.7	1872.0
V	0.400	0.400
Time (s)	229.8	3528.2

to the non-convexity of the problem, increasing the standard deviation's multiple can sometimes lead to a simultaneous increase or reduction in the mean and standard deviation of the compliance. The different optimal topologies are shown in Figure 3-8.

3.7.2.3 Maximum compliance constrained optimization

The 2D and 3D maximum compliance constrained volume minimization problems were solved using the SVD-based approach. The 2D optimal topology, shown in Figure 3-9, had a volume fraction of 0.584 and a maximum compliance of 69847.0

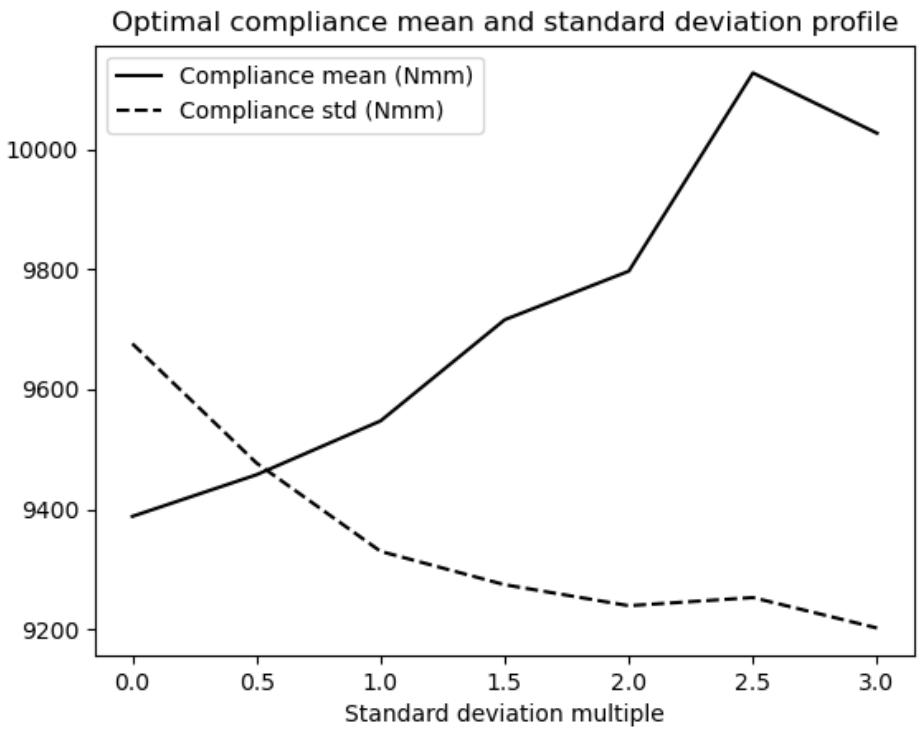
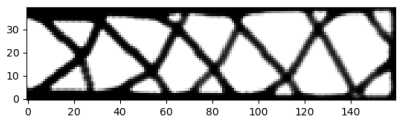
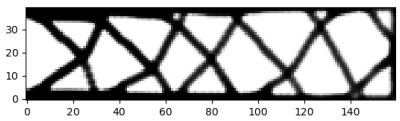


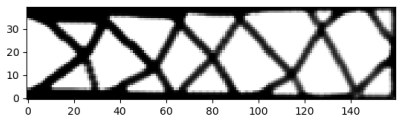
Figure 3-7: Profile of the optimal mean and standard deviation of the compliance for different standard deviation multiples in the objective.



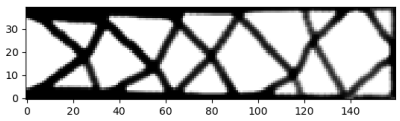
(a) $m = 0$



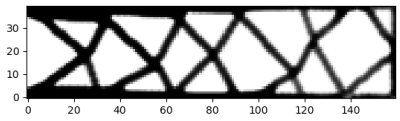
(b) $m = 0.5$



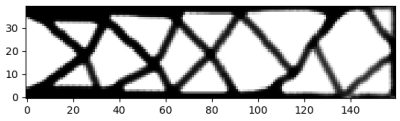
(c) $m = 1.0$



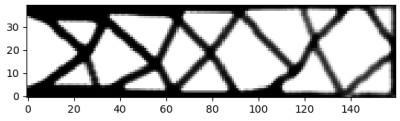
(d) $m = 1.5$



(e) $m = 2.0$



(f) $m = 2.5$



(g) $m = 3.0$

Figure 3-8: Optimal topologies of the 2D mean-std compliance minimization problem using different standard deviation multiples m in the objective $\mu_C + m\sigma_C$.

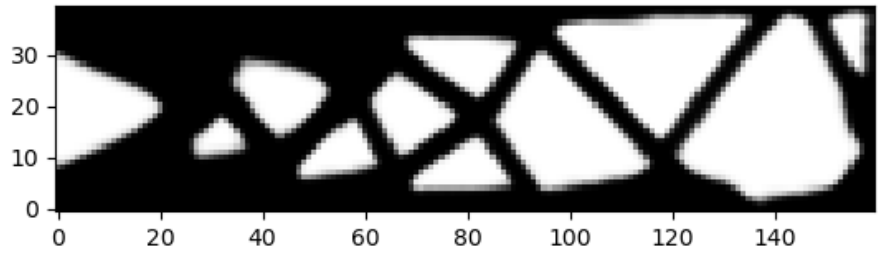
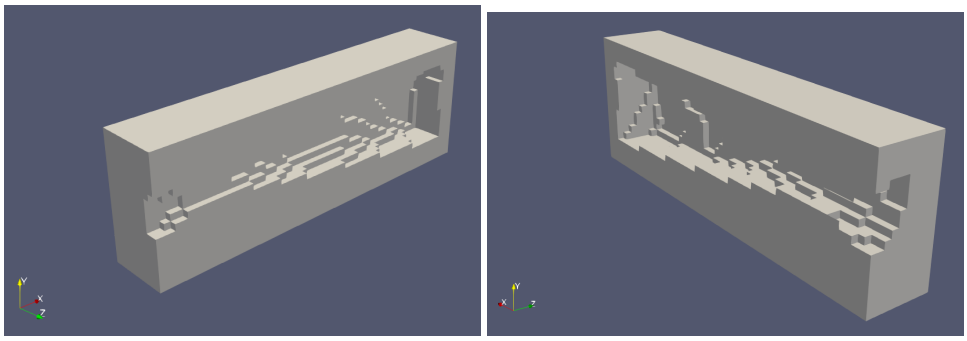


Figure 3-9: Optimal topology of the volume minimization problem subject to a maximum compliance constraint using continuation SIMP and the augmented Lagrangian method with the exact SVD approach. The maximum compliance of the design above is 69847.0 Nmm and the volume fraction is 0.584.

Nmm and was reached in 662.7 s. The 3D optimal topology, shown in Figure 3-10, had a volume fraction of 0.791 and a maximum compliance of 68992.4 Nmm and was reached in 43740.6 s.

3.8 Conclusion

In this chapter, a number of exact methods were proposed to handle load uncertainty in compliance topology optimization problems where the uncertainty is described in the form of a set of finitely many loading scenarios. By exploiting low rank structures in loading scenarios, significant performance improvements were achieved



(a) Left half (b) Right half

Figure 3-10: Cut views of the 3D optimal topology of the volume minimization problem subject to a maximum compliance constraint using continuation SIMP and the augmented Lagrangian method with the exact SVD approach. The maximum compliance of the design above is 68992.4 Nmm and the volume fraction is 0.791.

using novel SVD-based methods. Such improvement was demonstrated via complexity analysis and computational experiments. The methods presented here are fundamentally data-driven in the sense that no probability distributions or domains are assumed for the loading scenarios. This sets this work apart from most of the literature in the domain of stochastic and robust topology optimization where a distribution or domain is assumed. Additionally, the methods proposed here were shown to be particularly suitable with the augmented Lagrangian algorithm when dealing with maximum compliance constraints.

4. Approximate compliance-based optimization with finitely many loading scenarios

This work is submitted for publication in the Journal of Structural and Multidisciplinary Optimization. The submitted manuscript can be found on arXiv (https://arxiv.org/abs/2108.03654). The paper is titled: "Approximation schemes for stochastic compliance-based topology optimization with many loading scenarios".

4.1 Introduction

In the previous chapter, a few exact methods for handling a large number of loading scenarios in compliance-based problems were proposed based on the singular value decomposition (SVD), where the loading matrix $\boldsymbol{F}$ has a low rank and/or a few degrees of freedom are loaded. However when these conditions are not satisfied, the SVD based approach may not be efficient enough. In particular, there are 2 limitations to the SVD-based approaches:

1. The computational time complexity of computing the SVD of $\boldsymbol{F}$ is $O(min(L, n_{dofs})^2 max(L, n_{dofs}))$ if the loads are dense, where L is the number of loading scenarios and n_{dofs} is the number of degrees of freedom, which can be computationally prohibitive for large problems.

2. The load matrix may not be low rank.

In this chapter, a few SVD-free approximate methods will be proposed to estimate the value and gradient of:

1. The mean compliance

2. The standard deviation of the compliance

3. A class of scalar-valued functions of load compliances satisfying a few conditions

In this chapter, computationally efficient approximation schemes are proposed to approximate the values and gradients of the mean compliance, its standard deviation

as well as a class of scalar valued function of individual load compliances subject to a finite number of possible loading scenarios. These approaches can be used in risk-averse compliance minimization.

4.2 Proposed algorithms

4.2.1 Approximating the compliance sample mean and its gradient

The mean compliance can be formulated as a trace function: $\mu_C = \frac{1}{L} tr(\boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F})$. Zhang et al. (2017) showed that Hutchinson's trace estimator Hutchinson (1990) can be used to accurately estimate the compliance for a large number of load scenarios using a relatively small number of linear system solves. Hutchinson's trace estimator is given by:

$$tr(\boldsymbol{A}) = E(\boldsymbol{v}^T \boldsymbol{A} \boldsymbol{v}) \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{v}_i^T \boldsymbol{A} \boldsymbol{v}_i \tag{4.1}$$

where $\boldsymbol{v}$ is a random vector with each element independently distributed with 0 mean and unit variance, $\boldsymbol{v}_i$ are samples of the random vector $\boldsymbol{v}$, also known as probing vectors, and N is the number of such probing vectors. One common distribution used for the elements of $\boldsymbol{v}$ is the Rademacher distribution which is a discrete distribution with support $\{-1, 1\}$ each of which has a probability of 0.5. Hutchinson proved that an estimator with the Rademacher distribution for $\boldsymbol{v}$ will have the least variance

Table 4.1: Summary of the computational cost of the algorithms discussed to calculate the mean compliance and its gradient. #Lin is the number of linear system solves required.

Method	#Lin	Time complexity of additional work	
		Dense	Sparse
Exact	L	$O(L \times (n_{dofs} + n_E))$	$O(L \times (n_{loaded} + n_E))$
Trace estimation	N	$O(N \times (n_{dofs} \times L + n_E))$	$O(N \times (L \times n_{loaded} + n_E))$

among all other distributions. Let $\boldsymbol{A} = \boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F}$. The number of linear system solves required to compute the mean compliance $\frac{1}{L} tr(\boldsymbol{A})$ using the naive approach is L. However, when using Hutchinson's estimator, that number becomes the number of probing vectors N. In general, a good accuracy can be obtained for $N \ll L$. Other than the linear system solves, the time complexity of the remaining work using the trace estimation method is $O(N \times n_{dofs} \times L)$ mostly spent on finding $\boldsymbol{F}\boldsymbol{v}_i$ for all i. If only a small number of degrees of freedom n_{loaded} are loaded, the complexity of the remaining work reduces to $O(N \times n_{loaded} \times L)$.

Let $\boldsymbol{z}_i = \boldsymbol{K}^{-1} \boldsymbol{F} \boldsymbol{v}_i$ be cached from the trace computation. The elements of the gradient of the trace estimate with respect to $\boldsymbol{\rho}$ are given by:

$$\mu_C(\boldsymbol{\rho}) = \frac{1}{L \times N} \sum_{i}^{N} \boldsymbol{z}_i^T \boldsymbol{K} \boldsymbol{z}_i \tag{4.2}$$

$$\frac{\partial \mu_C}{\partial \rho_e} = \frac{1}{L \times N} \sum_{i=1}^{N} -\boldsymbol{z}_i^T \boldsymbol{K}_e \boldsymbol{z}_i \tag{4.3}$$

The additional time complexity of computing the gradient of the trace estimate after computing the trace is therefore $O(N \times n_E)$. For a detailed derivation of the partial above, see the appendix.

4.2.2 Approximating scalar-valued function of load compliances and its gradient

The above scheme for approximating the sample mean compliance can be generalized to handle the sample variance and standard deviations. The sample variance of the compliance C is given by $\sigma_C^2 = \frac{1}{L-1}\sum_{i=1}^{L}(C_i - \mu_C)^2$. The sample standard deviation σ_C is the square root of the variance. Let $\boldsymbol{C}$ be the vector of compliances C_i, one for each load scenario. In vector form, $\sigma_C^2 = \frac{1}{L-1}(\boldsymbol{C} - \mu_C\mathbf{1})^T(\boldsymbol{C} - \mu_C\mathbf{1})$. $\boldsymbol{C} = diag(\boldsymbol{A})$ is the diagonal of the matrix $\boldsymbol{A} = \boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}$.

One can view the load compliances $\boldsymbol{C}$ as the diagonal of the matrix $\boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}$. One way to estimate it is therefore to use a diagonal estimation method. One diagonal estimator directly related to Hutchinson's trace estimator was proposed by Bekas et al. Bekas et al. (2007). The diagonal estimator can be written as follows:

$$diag(\boldsymbol{A}) = E(\boldsymbol{D_v}\boldsymbol{A}\boldsymbol{v}) \approx \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{D}_{\boldsymbol{v}_i}\boldsymbol{A}\boldsymbol{v}_i \tag{4.4}$$

where $diag(\boldsymbol{A})$ is the diagonal of $\boldsymbol{A}$ as a vector, $\boldsymbol{D_v}$ is the diagonal matrix with a diagonal $\boldsymbol{v}$, $\boldsymbol{v}$ is a random vector distributed much like in Hutchinson's estimator, $\boldsymbol{v}_i$ are the probing vector instances of $\boldsymbol{v}$ and N is the number of probing vectors. The sum of the elements of the diagonal estimator above gives us Hutchinson's trace estimator. Let $\boldsymbol{A} = \boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}$:

$$\boldsymbol{C} = diag(\boldsymbol{F^T}\boldsymbol{K^{-1}}\boldsymbol{F}) \approx \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{D}_{\boldsymbol{v}_i}\boldsymbol{F^T}\boldsymbol{K^{-1}}\boldsymbol{F}\boldsymbol{v}_i \tag{4.5}$$

Bekas et al. showed that using the deterministic basis of a Hadamard matrix as probing vectors $\boldsymbol{v}_i$ rather than random vectors increases the accuracy of the diagonal estimator. In this chapter, we do the same and use columns of a Hadamard matrix as probing vectors for the diagonal estimator. Given the diagonal estimate assuming $N \ll L$, one can estimate $\boldsymbol{C}$ using N linear system solves, which can then be used to compute the sample variance and standard deviation. Other than the linear system solves, the additional work required above has a time complexity of $O(N \times n_{dofs} \times L)$. But if only a few n_{loaded} degrees of freedom are loaded, the time complexity of the remaining work goes down to $O(N \times n_{loaded} \times L)$.

The Jacobian of the compliances vector $\boldsymbol{C}$ with respect to $\boldsymbol{\rho}$, $\nabla_{\boldsymbol{\rho}}\boldsymbol{C}$ is simply the stacking of the transposes of the gradients of C_i, $\nabla_{\boldsymbol{\rho}} C_i$, for all i to form a matrix with L rows and n_E columns. The Jacobian of the estimate of $\boldsymbol{C}$ is given by:

$$\nabla_{\boldsymbol{\rho}}\boldsymbol{C} \approx \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_i}\boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}\boldsymbol{v}_i = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_i}\boldsymbol{F}^T\nabla_{\boldsymbol{\rho}}\boldsymbol{t}_i \tag{4.6}$$

where $\boldsymbol{t}_i = \boldsymbol{K}^{-1}\boldsymbol{F}\boldsymbol{v}_i$. The derivative $\frac{\partial \boldsymbol{t}_i}{\partial \rho_e} = -\boldsymbol{K}^{-1}\boldsymbol{K}_e\boldsymbol{t}_i$. Therefore:

$$\frac{\partial \boldsymbol{C}}{\partial \rho_e} \approx \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_i}\boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{K}_e\boldsymbol{K}^{-1}\boldsymbol{F}\boldsymbol{v}_i \tag{4.7}$$

Note that to find the Jacobian in this case, one requires L linear system solves to find $\boldsymbol{K}^{-1}\boldsymbol{F}$. However, with this many linear system solves one can use the exact method so there is no merit to using the diagonal estimation approach. This means that if the full Jacobian is required, the diagonal estimation method here is the wrong choice.

Table 4.2: Summary of the computational cost of the algorithms discussed to calculate the load compliances $\boldsymbol{C}$ as well as $\nabla_{\boldsymbol{\rho}} \boldsymbol{C}^T \boldsymbol{w}$ for any vector $\boldsymbol{w}$. #Lin is the number of linear system solves required. This can be used to compute the variance, standard deviation as well as other scalar-valued functions of $\boldsymbol{C}$.

Method	#Lin	Time complexity of additional work	
		Dense	Sparse
Exact	L	$O(L \times (n_{dofs} + n_E))$	$O(L \times (n_{loaded} + n_E))$
Diagonal estimation	$2N$	$O(N \times (n_{dofs} \times L + n_E))$	$O(N \times (n_{loaded} \times L + n_E))$

However if only interested in $\nabla_{\boldsymbol{\rho}} \boldsymbol{C}(\boldsymbol{\rho})^T \boldsymbol{w}$, a more efficient approach can be used:

$$\nabla_{\boldsymbol{\rho}} \boldsymbol{C}(\boldsymbol{\rho})^T \boldsymbol{w} = \nabla_{\boldsymbol{\rho}} (\boldsymbol{C}(\boldsymbol{\rho})^T \boldsymbol{w}) = \nabla_{\boldsymbol{\rho}} tr(\boldsymbol{D}_{\boldsymbol{w}} \boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{F}) \tag{4.8}$$

Let $\boldsymbol{r}_i = \boldsymbol{K}^{-1} \boldsymbol{F} \boldsymbol{v}_i$ which are cached from the function value calculation, and let $\boldsymbol{t}_i = \boldsymbol{K}^{-1} \boldsymbol{F} \boldsymbol{D}_{\boldsymbol{w}} \boldsymbol{v}_i$.

$$\frac{\partial \boldsymbol{C}(\boldsymbol{\rho})^T \boldsymbol{w}}{\partial \rho_e} = -tr(\boldsymbol{D}_{\boldsymbol{w}} \boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{K}_e \boldsymbol{K}^{-1} \boldsymbol{F}) \tag{4.9}$$

$$\approx -\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{v}_i^T \boldsymbol{D}_{\boldsymbol{w}} \boldsymbol{F}^T \boldsymbol{K}^{-1} \boldsymbol{K}_e \boldsymbol{K}^{-1} \boldsymbol{F} \boldsymbol{v}_i \tag{4.10}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{t}_i^T \boldsymbol{K}_e \boldsymbol{r}_i \tag{4.11}$$

This means that at a cost of an additional N linear system solves, one can compute the vectors $\boldsymbol{t}_i$ and then find the gradient of $\boldsymbol{C}^T \boldsymbol{w}$. Other than the linear system solves, the remaining work has a time complexity of $O(N \times (n_{dofs} \times L + n_E))$, $O(N \times n_{dofs} \times L)$ from the accumulation of $\boldsymbol{F} \boldsymbol{D}_{\boldsymbol{w}} \boldsymbol{v}_i$ and $O(N \times n_E)$ to evaluate the gradient given $\boldsymbol{t}_i$ and $\boldsymbol{r}_i$ for all i. If only a few degrees of freedom n_{loaded} are loaded, then the complexity goes down to $O(N \times (n_{loaded} \times L + n_E))$.

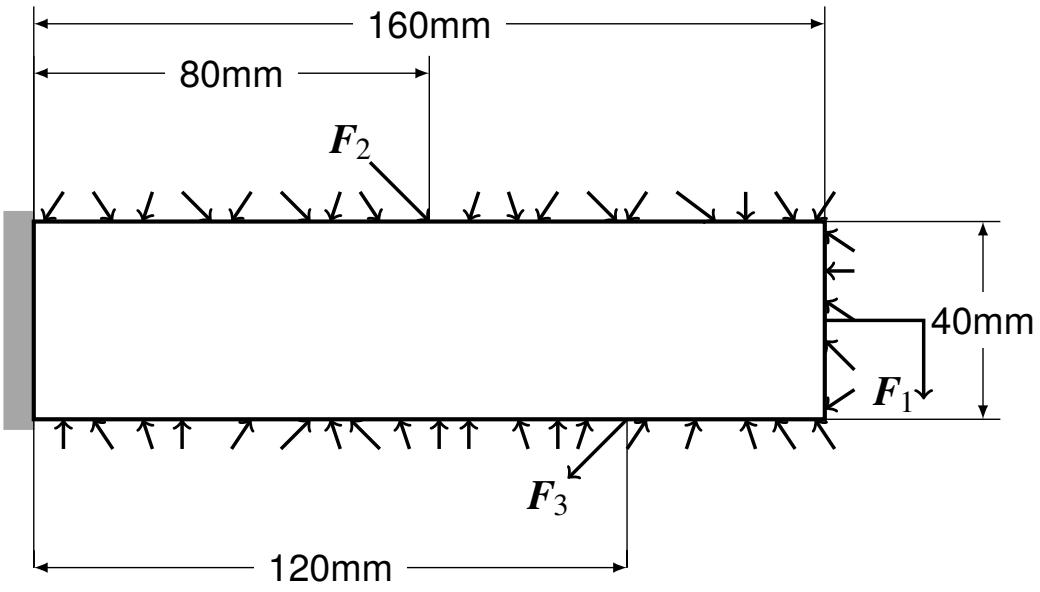


Figure 4-1: Cantilever beam problem. $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are at 45 degree angles.

4.3 Setup and Implementation

In this section, the most important implementation details and algorithm settings used in the experiments are presented.

4.3.1 Test problems

The 2D cantilever beam problem shown in Figure 4-1 was used to run the experiments. A ground mesh of plane stress quadrilateral elements was used, where each element is a square of side length 1 mm, and a sheet thickness of 1 mm. Linear iso-parametric interpolation functions were used for the field and geometric basis functions. A Young's modulus of 1 MPa and Poisson's ratio of 0.3 were used. Finally, a chequerboard density filter for unstructured meshes was used with a radius of 2 mm (Huang and Xie, 2010). A 3D version of the problem above was also

solved.

Three variants of the cantilever beam problem were solved:

1. Minimization of the mean compliance μ_C subject to a volume constraint with a volume fraction of 0.4,

2. Minimization of a weighted sum of the mean and standard deviation (mean-std) of the compliance $\mu_C + 2.0\sigma_C$ subject to a volume constraint with a volume fraction of 0.4, and

3. Volume minimization subject to a maximum compliance constraint with a compliance threshold of 70000 Nmm.

A total of 1000 load scenarios were sampled from:

$$\boldsymbol{f}_i = s_1\boldsymbol{F}_1 + s_2\boldsymbol{F}_2 + s_3\boldsymbol{F}_3 + \frac{1}{R-3}\sum_{j=4}^{R} s_j\boldsymbol{F}_j \tag{4.12}$$

where $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are unit vectors with directions as shown in Figure 4-1 and R is an integer greater than or equal to 4. $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ are at 45 degrees. s_1, s_2 and s_3 are identically and independently uniformly distributed random variables between -2 and 2. $\boldsymbol{F}_j$ for j in $4 \ldots R$ are vectors with non-zeros at all the surface degrees of freedom without a Dirichlet boundary condition. The non-zero values are identically and independently normally distributed random variables with mean 0 and standard deviation 1. s_j for j in $4 \ldots R$ are also identically and independently normally distributed random variables with mean 0 and standard deviation 1. The same loading scenarios were used for the 3 test problems. Let $\boldsymbol{F}$ be the matrix

whose columns are the sampled $\boldsymbol{f}_i$ vectors. Given the way the loading scenarios have been defined the rank of $\boldsymbol{F}$ is almost certainly going to be around R.

4.3.2 Settings

The value of x_{min} used was 0.001 for all problems and algorithms. Penalization was done prior to interpolation to calculate $\boldsymbol{\rho}$ from $\boldsymbol{x}$. A power penalty function and a regularized Heaviside projection were used. All of the problems were solved using 2 continuation SIMP routines. The first incremented the penalty value from $p = 1$ to $p = 6$ in increments of 0.5. Then the Heaviside projection parameter β was incremented from $\beta = 0$ to $\beta = 20$ in increments of 4 keeping the penalty value fixed at 6. An exponentially decreasing tolerance from $1e-3$ to $1e-4$ was used for both continuations.

The mean and mean-std compliance minimization SIMP subproblems problems were solved using the method of moving asymptotes (MMA) algorithm (Svanberg, 1987). MMA parameters of $s_{init} = 0.5$, $s_{incr} = 1.1$ and $s_{decr} = 0.7$ were used as defined in the MMA paper with a maximum of 1000 iterations for each subproblem. The dual problem of the convex approximation was solved using a log-barrier box-constrained nonlinear optimization solver, where the barrier problem was solved using the nonlinear CG algorithm for unconstrained nonlinear optimization (Nocedal and Wright, 2006) as implemented in Optim.jl [1] (K Mogensen and N Riseth, 2018). The nonlinear CG itself used the line search algorithm from Hager and Zhang (2006)

[1] https://github.com/JuliaNLSolvers/Optim.jl

Table 4.3: The table shows the function values of μ_C computed using the exact method and the approximate method of trace estimation with 100 Rademacher-distributed or Hadamard basis probing vectors for a full ground mesh design. The table also shows the time required to compute or approximate μ_C and its gradient in each case. A value of $R = 10$ was used here.

Method	μ_C (Nmm)	Time (s)
Naive Exact	3328.7	24.2
Trace estimation	3596.9 (Rademacher) / 3486.7 (Hadamard)	2.6

as implemented in LineSearches.jl [2]. The stopping criteria used was the one adopted by the KKT solver, IPOPT (Wächter and Biegler, 2006). This stopping criteria is less scale sensitive than the KKT residual as it scales down the residual by a value proportional to the mean absolute value of the Lagrangian multipliers.

4.4 Accuracy and speed comparison

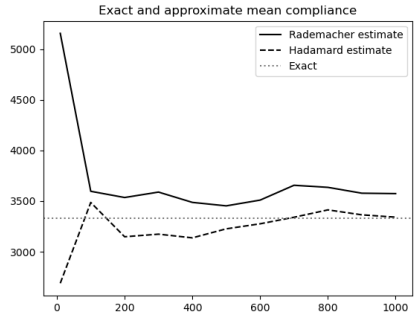
In this section, the accuracy and speed of the approximations proposed are presented and compared to the exact values. A method to boost the accuracy of approximations is also presented and mathematically analyzed. Tables 4.3 and 4.4 show the values computed for the mean compliance μ_C and its standard deviation σ_C respectively together with the time required to compute their values and gradients using: the naive exact approach and the approximate method with trace or diagonal estimation using 100 Rademacher-distributed or Hadamard basis probing vectors. A value of $R = 10$ was used.

As expected, the proposed approximation methods take a fraction of the time it takes to compute the exact mean and mean-std compliances using the approaches.

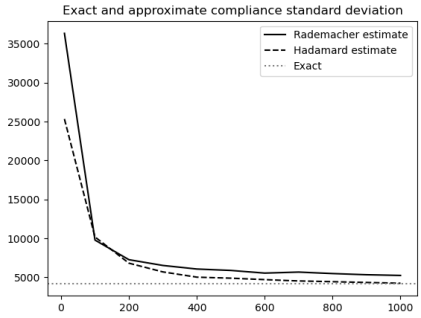
[2] https://github.com/JuliaNLSolvers/LineSearches.jl

Table 4.4: The table shows the function values of σ_C and its gradients for a full ground mesh computed using the exact method and the approximate method of diagonal estimation with 100 Rademacher-distributed or Hadamard basis probing vectors. The table also shows the time required to compute the exact or approximate σ_C and its gradient in each case. A value of $R = 10$ was used here. Note the extreme bias in the estimate so a correction step is necessary.

Method	σ_C (Nmm)	Time (s)
Exact	4172.8	28.0
Diagonal estimation	9774.8 (Rademacher) / 10173.3 (Hadamard)	5.2



(a) Mean compliance estimate using different numbers of probing vectors in the trace estimation method.



(b) Compliance standard deviation estimate using different numbers of probing vectors in the diagonal estimation method.

Figure 4-2: Accuracy profile of the trace and diagonal estimation methods for estimating the mean compliance and its standard deviation using 10, 100, 200, 300, 400, 500, 600, 700, 800, 900 and 1000 probing vectors. A value of $R = 10$ was used here.

Estimates of the mean compliance and its standard deviation for a full ground mesh using different numbers of Rademacher-distributed and Hadamard basis probing vectors are shown in Figure 4-2. In this case, the estimates obtained using the Hadamard basis were always closer to the exact value than that of the Rademacher-distributed one. However, this depends on the order by which the Hadamard basis vectors are used.

4.5 Bias correction

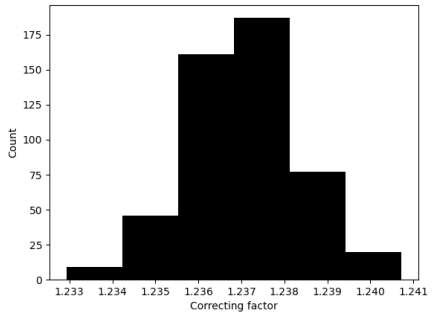
While the Hadamard estimate is converging faster to the exact value compared to the Rademacher one in the above case as the number of probing vectors increases, it is still quite far in the case of the standard deviation unless a large number ofIf we have a constraint over the weighted sum of the mean compliance and its standard deviation, this huge discrepancy from the exact quantity renders the approximate method useless. In this section, it will be shown that usually the estimate can be multiplied by a correcting factor to significantly improve its accuracy. This will be demonstrated experimentally and then mathematically analyzed. When performing topology optimization, the function value and its gradient need to be computed repeatedly. So if we only need to compute the correcting factor a few times, we can still save a lot of computational time when using the approximate method without losing too much accuracy.

4.5.1 Experiments

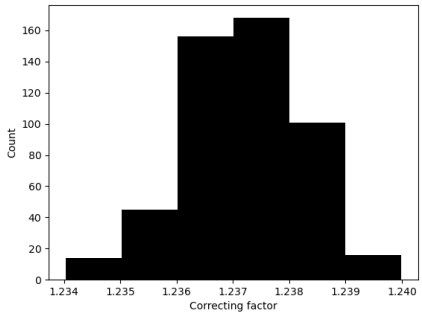
Using a full ground mesh to calculate the correcting factor and only 10 probing vectors and $R = 10$, the ratio between the exact mean compliance and the trace estimate using Hadamard basis probing vectors was 1.238. Similarly, the ratio of the exact compliance standard deviation to the estimated value was 0.165. Figures 4-3 and 4-4 show the distributions of the ratios of the exact value to the estimated one for the mean compliance and its standard deviation respectively. The same 10 Hadamard basis probing vectors were used and each figure was generated using 500 random designs. For each figure, the random designs were sampled from a truncated normal distributions with a different mean and a standard deviation of 0.2, truncated between 0 and 1. One can see that using the same probing vectors, the ratio between the exact and estimated values doesn't change significantly even when changing the mean volume by changing the mean of the truncated normal distribution. One can see that the correcting ratio that can multiply the estimated mean compliance or standard deviation to get the exact one is not very sensitive to the underlying design.

4.5.2 Mathematical analysis

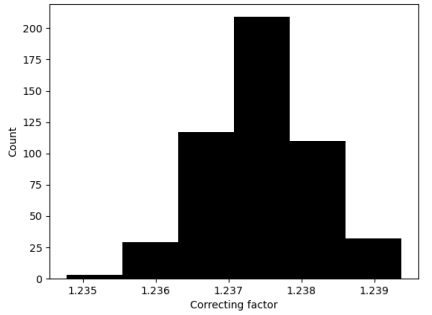
In this section, an attempt will be made to mathematically explain the insensitivity of the estimators' correcting ratios to the design as shown above. While this section doesn't provide a rigorous proof of the phenomena observed, it does provide some mathematical insight into why it is happening and when it can be expected to happen in other problems.



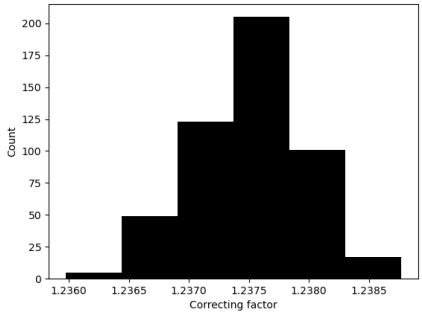
(a) Mean = 0.1



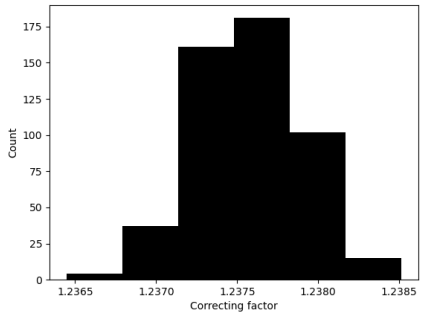
(b) Mean = 0.3



(c) Mean = 0.5

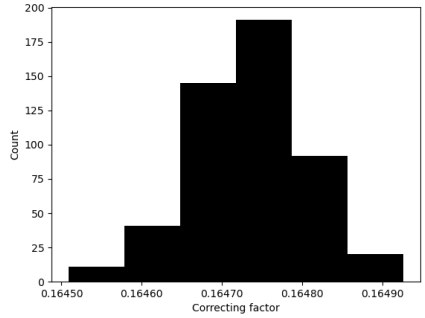


(d) Mean = 0.7

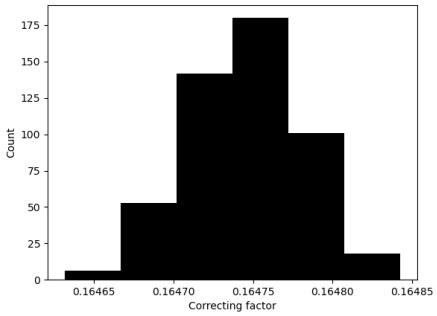


(e) Mean = 0.9

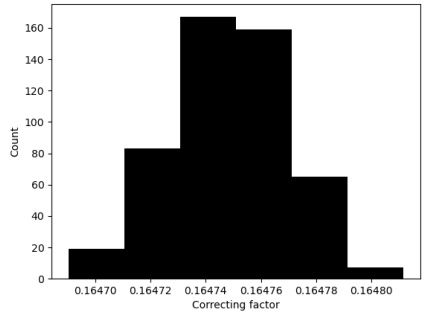
Figure 4-3: Histograms of the ratio between the exact mean compliance and the trace estimate using 10 Hadamard basis probing vectors. In each figure, 500 designs were randomly sampled where each element's pseudo-density is sampled from a truncated normal distribution with the means indicated above and a standard deviation of 0.2, truncated between 0 and 1. A value of $R = 10$ was used here.



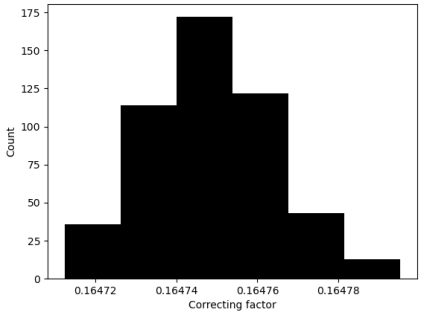
(a) Mean = 0.1



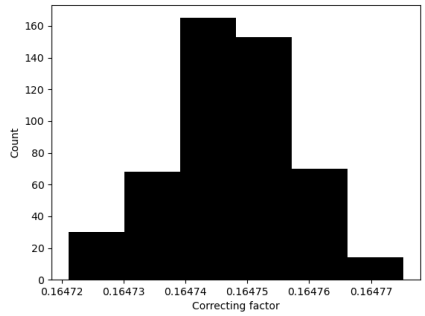
(b) Mean = 0.3



(c) Mean = 0.5



(d) Mean = 0.7



(e) Mean = 0.9

Figure 4-4: Histograms of the ratio between the exact compliance standard deviation and the estimate using 10 Hadamard basis probing vectors. In each figure, 500 designs were randomly sampled where each element's pseudo-density is sampled from a truncated normal distribution with the means indicated above and a standard deviation of 0.2, truncated between 0 and 1. A value of $R = 10$ was used here.

Let the diagonal estimator be:

$$\frac{1}{N}\sum_{k=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_k}\boldsymbol{A}\boldsymbol{v}_k = \frac{1}{N}\Big(\sum_{k=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_k}\boldsymbol{A}\boldsymbol{D}_{\boldsymbol{v}_k}\Big)\mathbf{1} \tag{4.13}$$

where $\boldsymbol{A} = \boldsymbol{F}^T\boldsymbol{K}^{-1}\boldsymbol{F}$. Let a_{ij} be $(i,j)^{th}$ element of $\boldsymbol{A}$. Let v_{ki} be the i^{th} element of $\boldsymbol{v}_k$. The $(i,j)^{th}$ element of $\sum_{k=1}^{N} \boldsymbol{D}_{\boldsymbol{v}_k}\boldsymbol{A}\boldsymbol{D}_{\boldsymbol{v}_k}$ is therefore $a_{ij}\sum_{k=1}^{N} v_{ki}v_{kj}$. Let N_{ij}^{+} be the number of times $v_{ki}v_{kj}$ is 1 and N_{ij}^{-} be the number of times $v_{ki}v_{kj}$ is -1. In the case of Hadamard basis, if N is the smallest power of 2 larger than or equal to the number of loads L, then:

$$N_{ij}^{+} = N_{ij}^{-} = N/2 \quad if \quad i \neq j \tag{4.14}$$

$$N_{ij}^{+} = N, N_{ij}^{-} = 0 \quad if \quad i = j \tag{4.15}$$

This means that the diagonal estimate will be exact in that case. Bekas et al. Bekas et al. (2007) showed that Hadamard basis work well for banded matrices and for matrices where off-diagonal values are decaying rapidly away from the diagonal. However in the case of load compliances, neither of those conditions apply. Therefore, as shown in the experiment above, the accuracy of the estimated diagonal is quite bad as obvious from the standard deviation of the estimate. Let the i^{th} diagonal element (or load compliance) be $C_i = a_{ii}$. The estimator of a_{ii}, $\hat{a}_{ii}$, can

be written as:

$$\hat{a}_{ii} = \frac{1}{N} \sum_{j=1}^{L} a_{ij} \sum_{k=1}^{N} v_{ki} v_{kj} \tag{4.16}$$

$$= \frac{1}{N} \sum_{j=1}^{L} a_{ij} (N_{ij}^{+} - N_{ij}^{-}) \tag{4.17}$$

$$= a_{ii} + \sum_{j \neq i} a_{ij} \frac{N_{ij}^{+} - N_{ij}^{-}}{N} \tag{4.18}$$

The ratio of the estimated diagonal element to the actual diagonal element is:

$$\frac{\hat{a}_{ii}}{a_{ii}} = 1 + \sum_{j \neq i} \frac{a_{ij}}{a_{ii}} \frac{N_{ij}^{+} - N_{ij}^{-}}{N} \tag{4.19}$$

This ratio depends on:

1. $\frac{a_{ij}}{a_{ii}}$ which depends on the design and the load scenarios, and

2. N_{ij}^{+} and N_{ij}^{-} which depend on the Hadamard basis used.

If the same basis are used for all the designs during optimization, then $\frac{a_{ij}}{a_{ii}}$ is the only number that can vary.

$$\frac{a_{ij}}{a_{ii}} = \frac{\boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_j}{\boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i} \tag{4.20}$$

The partial derivative of a_{ij}/a_{ii} with respect to the e^{th} element's density ρ_e is:

$$\frac{\partial(a_{ij}/a_{ii})}{\partial\rho_e} = \frac{\partial(\boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_j/\boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_i)}{\partial\rho_e} \tag{4.21}$$

$$= \frac{\frac{\partial \boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_j}{\partial\rho_e}\boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_i - \frac{\partial \boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_i}{\partial\rho_e}\boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_j}{(\boldsymbol{f}_i^T\boldsymbol{K}^{-1}\boldsymbol{f}_i)^2} \tag{4.22}$$

$$= \frac{-(\boldsymbol{u}_i^T\boldsymbol{K}_e\boldsymbol{u}_j)(\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i) + (\boldsymbol{u}_i^T\boldsymbol{K}_e\boldsymbol{u}_i)(\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_j)}{(\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i)^2} \tag{4.23}$$

$$\tag{4.24}$$

Lemma 4.5.1.

$$|\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_j| \leq \frac{1}{2}\max(|(\boldsymbol{u}_i+\boldsymbol{u}_j)^T\boldsymbol{K}(\boldsymbol{u}_i+\boldsymbol{u}_j)|, |\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i + \boldsymbol{u}_j^T\boldsymbol{K}\boldsymbol{u}_j|) \tag{4.25}$$

if $\boldsymbol{K}$ is positive or negative semi-definite and

$$|\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_j| \leq \frac{1}{2}\Big(|(\boldsymbol{u}_i+\boldsymbol{u}_j)^T\boldsymbol{K}(\boldsymbol{u}_i+\boldsymbol{u}_j)| + |\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i| + |\boldsymbol{u}_j^T\boldsymbol{K}\boldsymbol{u}_j)|\Big) \tag{4.26}$$

otherwise.

Proof.

$$2\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_j = (\boldsymbol{u}_i+\boldsymbol{u}_j)^T\boldsymbol{K}(\boldsymbol{u}_i+\boldsymbol{u}_j) - \boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i - \boldsymbol{u}_j^T\boldsymbol{K}\boldsymbol{u}_j \tag{4.27}$$

If $\boldsymbol{K}$ is indefinite:

$$2|\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_j| \leq |(\boldsymbol{u}_i+\boldsymbol{u}_j)^T\boldsymbol{K}(\boldsymbol{u}_i+\boldsymbol{u}_j)| + |\boldsymbol{u}_i^T\boldsymbol{K}\boldsymbol{u}_i| + |\boldsymbol{u}_j^T\boldsymbol{K}\boldsymbol{u}_j| \tag{4.28}$$

If $\boldsymbol{K}$ is positive semi-definite, then $(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j)$, $\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i$ and $\boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j$ are all non-negative. Therefore:

$$-(\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i + \boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j) \leq 2\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_j \leq (\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j) \tag{4.29}$$

Similarly, if $\boldsymbol{K}$ is negative semi-definite, then $(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j)$, $\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i$ and $\boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j$ are all non-positive. Therefore:

$$(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j) \leq 2\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_j \leq -(\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i + \boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j) \tag{4.30}$$

It follows that:

$$2|\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_j| \leq \max(|(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j)|, |\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i + \boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j|) \tag{4.31}$$

This completes the proof. □

Using the above bound, it follows that if for all combinations of i and j:

$$\frac{\boldsymbol{u}_j^T \boldsymbol{K}_e \boldsymbol{u}_j}{\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i} \leq \alpha_1 \tag{4.32}$$

$$\frac{(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}_e (\boldsymbol{u}_i + \boldsymbol{u}_j)}{2\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i} \leq \beta_1 \tag{4.33}$$

$$\frac{\boldsymbol{u}_j^T \boldsymbol{K} \boldsymbol{u}_j}{\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i} \leq \alpha_2 \tag{4.34}$$

$$\frac{(\boldsymbol{u}_i + \boldsymbol{u}_j)^T \boldsymbol{K}(\boldsymbol{u}_i + \boldsymbol{u}_j)}{2\boldsymbol{u}_i^T \boldsymbol{K} \boldsymbol{u}_i} \leq \beta_2 \tag{4.35}$$

then

$$\left|\frac{\partial(a_{ij}/a_{ii})}{\partial\rho_e}\right| \leq \max(\alpha_1, \beta_1) + \alpha_1 \times \max(\alpha_2, \beta_2) \tag{4.36}$$

It is natural to expect α_1 to be small since the element compliance due to any one load will likely be much smaller than the total compliance due to any other load. If the loading scenarios have widely varying magnitudes, α_1 may be large in that case. However to remedy this, the loading scenarios can be clustered into groups by their norm and a separate estimator can be used for each group. Similarly, β_1 is likely to be small if all the forces have a close enough norm since the element compliance due to the superposition of 2 loads is likely to be much smaller than two times the total compliance due to any other load. If the norms of the loads are somewhat similar, α_2 and β_2 can also be expected to be small constants greater than or equal to 1. This means that absolute value of the individual partial derivatives can be upper bounded by a small positive number. Interestingly, the sum of all the partial derivatives of the correcting ratio with respect to the individual element densities, $\sum_e \frac{\partial(a_{ij}/a_{ii})}{\partial\rho_e}$, is 0. This does not guarantee that the directional derivative in any direction will be small but it increases the chances of term cancellation. This is consistent with the observations.

However, the correcting factor for the estimator $\hat{C}_i = \hat{a}_{ii}$ does not just depend on the individual $\frac{\partial(a_{ij}/a_{ii})}{\partial\rho_e}$ but rather it depends on the sum $\sum_{j\neq i} \frac{a_{ij}}{a_{ii}} \frac{N_{ij}^+ - N_{ij}^-}{N}$. Three factors can make this sum small:

1. A good choice of probing vectors that make the distribution of $N_{ij}^+ - N_{ij}^-$ for

different (i, j) pairs symmetric around 0 promoting term cancellation.

2. Term cancellation due to the alternating signs of a_{ij}. For instance if the mean load vector is the $\mathbf{0}$ vector, the summation $\sum_{j\neq i} \frac{a_{ij}}{a_{ii}}$ is equal to -1 regardless of the number of loading scenarios.

3. A small ratio of the number of loading scenarios to the number of elements. This is detailed below.

For fixed loading scenarios, the values of α_1 and β_1 decrease as the number of elements E increases. This is because the ratio of an individual element's contribution to the total strain energy decreases as the element size decreases. Given that $-1 \leq \frac{N_{ij}^{+}-N_{ij}^{-}}{N} \leq 1$:

$$\left|\sum_{j\neq i} \frac{a_{ij}}{a_{ii}} \frac{N_{ij}^{+} - N_{ij}^{-}}{N}\right| \leq (L-1)(\beta_1 + \alpha_1 + \alpha_1(\beta_2 + \alpha_2)) \tag{4.37}$$

Therefore, if $L \ll E$ and the loads in $\boldsymbol{F}$ have close magnitudes, one can expect the correcting factor to be design insensitive especially near the end of the optimization when the design is not changing much.

The analysis above identified 3 strategies other than using more probing vectors that can help promote the insensitivity of the correcting factors to the design:

1. Clustering the loads by their magnitudes with a maximum number of loads per cluster $\ll E$,

2. Centering the loads around $\mathbf{0}$. Let $\boldsymbol{\mu_f}$ be the sample mean of the loading scenarios and let $\tilde{\boldsymbol{f}}_i = \boldsymbol{f}_i - \boldsymbol{\mu_f}$. The i^{th} load compliance $\boldsymbol{f}_i^T \boldsymbol{K}^{-1} \boldsymbol{f}_i$ would then

be $\tilde{\boldsymbol{f}}_i^T \boldsymbol{K}^{-1} \tilde{\boldsymbol{f}}_i + 2\tilde{\boldsymbol{f}}_i^T \boldsymbol{K}^{-1} \boldsymbol{\mu}_{\boldsymbol{f}} + \boldsymbol{\mu}_{\boldsymbol{f}}^T \boldsymbol{K}^{-1} \boldsymbol{\mu}_{\boldsymbol{f}}$. The terms $\tilde{\boldsymbol{f}}_i^T \boldsymbol{K}^{-1} \tilde{\boldsymbol{f}}_i$ can be obtained from the diagonal estimator of $\tilde{\boldsymbol{F}}^T \boldsymbol{K}^{-1} \tilde{\boldsymbol{F}}$ where the columns of $\tilde{\boldsymbol{F}}$ are the vectors $\tilde{\boldsymbol{f}}_i$. The remaining terms can be computed using a single linear system solve $\boldsymbol{K}^{-1} \boldsymbol{\mu}_{\boldsymbol{f}}$.

3. Using a finer mesh, i.e. increasing E thus decreasing α_1 and β_1.

Note that while the analysis above provides some mathematical insights into why the correcting ratios for the individual compliances may not be sensitive to the design, it is not a complete proof of the phenomena observed because only a single element's ρ_e was assumed to be changing in the analysis. However, from the analysis above one can see that term cancellation is highly likely in practice. For instance, the sum of $\frac{a_{ij}}{a_{ii}}$ for all j is equal to -1 if the mean load is $\mathbf{0}$ regardless of the number of loads, and the sum of $\frac{\partial a_{ij}/a_{ii}}{\partial \rho_e}$ for all e is equal to 0. This term cancellation is the main reason behind the extreme insensitivity of the correcting ratio to the design observed in the experiments above even when all the elements' densities are changing in random directions by large amounts.

Next it will be shown that under some conditions that the above insensitivity of the correcting ratio to any individual ρ_e can be extended to a class of scalar-valued functions of the load compliances. This class of functions includes the mean, variance and standard deviation but not the augmented Lagrangian penalty. Let γ_i be the correcting factor for the compliance C_i. The correcting factor of a scalar

valued function f of the load compliances can therefore be written as:

$$\eta(\boldsymbol{\rho}) = \frac{f(\gamma_1(\boldsymbol{\rho})\hat{C}_1(\boldsymbol{\rho}), \gamma_2(\boldsymbol{\rho})\hat{C}_2(\boldsymbol{\rho}), \ldots, \gamma_L(\boldsymbol{\rho})\hat{C}_L(\boldsymbol{\rho}))}{f(\hat{C}_1(\boldsymbol{\rho}), \hat{C}_2(\boldsymbol{\rho}), \ldots, \hat{C}_L(\boldsymbol{\rho}))} \tag{4.38}$$

Let $f_{\hat{\boldsymbol{C}}} = f(\hat{C}_1, \ldots, \hat{C}_L)$ and $f_{\boldsymbol{C}} = f(\gamma_1\hat{C}_1, \ldots, \gamma_L\hat{C}_L)$. Furthermore, let $f^{(i)}_{\hat{\boldsymbol{C}}}$ be the partial derivative of f with respect to its i^{th} argument evaluated at $(\hat{C}_1, \hat{C}_2, \ldots, \hat{C}_L)$ and let $f^{(i)}_{\boldsymbol{C}}$ be the partial derivative of f with respect to its i^{th} argument evaluated at $(\gamma_1\hat{C}_1, \gamma_2\hat{C}_2, \ldots, \gamma_L\hat{C}_L)$.

$$\frac{\partial \eta}{\partial \rho_e} = \sum_i \left(\frac{\partial \eta}{\partial \gamma_i} * \frac{\partial \gamma_i}{\partial \rho_e} + \frac{\partial \eta}{\partial \hat{C}_i} * \frac{\partial \hat{C}_i}{\partial \rho_e} \right) \tag{4.39}$$

$$= \sum_i \left(\frac{f^{(i)}_{\boldsymbol{C}}\hat{C}_i}{f_{\hat{\boldsymbol{C}}}} \frac{\partial \gamma_i}{\partial \rho_e} + \left(\frac{f^{(i)}_{\boldsymbol{C}}\gamma_i}{f_{\hat{\boldsymbol{C}}}} - \frac{f^{(i)}_{\hat{\boldsymbol{C}}} f_{\boldsymbol{C}}}{f^2_{\hat{\boldsymbol{C}}}} \right) \frac{\partial \hat{C}_i}{\partial \rho_e} \right) \tag{4.40}$$

One can see that if the magnitudes of $f^{(i)}_{\hat{\boldsymbol{C}}}$ and $f^{(i)}_{\boldsymbol{C}}$ scale down as L increases and if

$f_{\boldsymbol{C}}/f_{\hat{\boldsymbol{C}}}^2$ is small that the partial derivative $\frac{\partial \eta}{\partial \rho_e}$ will also likely be small. For all i, let:

$$|f_{\boldsymbol{C}}^{(i)}| \leq \frac{c_1}{L} \tag{4.41}$$

$$|f_{\hat{\boldsymbol{C}}}^{(i)}| \leq \frac{c_1}{L} \tag{4.42}$$

$$\left|\frac{\partial \hat{C}_i}{\partial \rho_e}\right| \leq c_2 \tag{4.43}$$

$$\left|\frac{\partial \gamma_i}{\partial \rho_e}\right| \leq c_3 \tag{4.44}$$

$$\left|\frac{\hat{C}_i}{f_{\hat{\boldsymbol{C}}}}\right| \leq c_4 \tag{4.45}$$

$$\left|\frac{\gamma_i}{f_{\hat{\boldsymbol{C}}}}\right| \leq c_5 \tag{4.46}$$

$$\left|\frac{f_{\boldsymbol{C}}}{f_{\hat{\boldsymbol{C}}}^2}\right| \leq c_6 \tag{4.47}$$

Then one can set the bound:

$$\left|\frac{\partial \eta}{\partial \rho_e}\right| \leq c_1 c_4 c_3 + c_1 c_5 c_2 + c_1 c_6 c_2 \tag{4.48}$$

From the above bound, one can see that c_3, c_5 and c_6 must be small enough to guarantee a low upper bound on the absolute value of $\frac{\partial \eta}{\partial \rho_e}$. This means that:

1. The diagonal's correcting factors must not be sensitive to ρ_e (i.e. c_3 is small). This has been established above under some conditions.

2. The ratio of the diagonal correcting factors to the function estimator $f_{\hat{\boldsymbol{C}}}$ must be small in magnitude, i.e. (c_5 is small). This is true for the experiment above, where the diagonal correcting ratios at the full ground mesh ranged from -19.0 to 21.7 while the estimated compliance mean and standard deviation

were 410.0 and 1329.7 respectively.

3. The ratio $f_{\boldsymbol{C}}/f_{\hat{\boldsymbol{C}}}^2$ must be small in magnitude, (i.e. c_6 is small). This is also true for the experiment above at the full ground mesh where the ratios were $2.7e-3$ and $3.0e-4$ for the mean and standard deviation of the compliance respectively.

To show that the above result applies to the mean, standard deviation and variance functions, it suffices to show that $|f_{\boldsymbol{C}}^{(i)}| \leq \frac{c_1}{L}$ for some constant c_1. If this is true for $f_{\boldsymbol{C}}^{(i)}$ then it is also true for $f_{\hat{\boldsymbol{C}}}^{(i)}$ since this is the same function evaluated at different points. The partial derivatives of the mean, standard deviation and variance of $(C_1, C_2, \ldots, C_L)$ with respect to each C_i are:

$$\frac{\partial \mu_C}{\partial C_i} = \frac{1}{L} \tag{4.49}$$

$$\frac{\partial \sigma_C}{\partial C_i} = \left(1 - \frac{1}{L}\right)\frac{C_i - \mu_C}{(L-1)\times\sigma_C} \leq \frac{2(C_i - \mu_C)}{L\times\sigma_C} \tag{4.50}$$

$$\frac{\partial \sigma_C^2}{\partial C_i} = \left(1 - \frac{1}{L}\right)\frac{2(C_i - \mu_C)}{(L-1)} \leq \frac{4(C_i - \mu_C)}{L} \tag{4.51}$$

because $L - 1 \geq L/2$ for $L > 1$. Let l_{μ_C} and l_{σ_C} be lower bounds on μ_C and σ_C for all the designs. The constant c_1 is therefore 1 for μ_C, $2(C_{max} - l_{\mu_C})/l_{\sigma_C}$ for σ_C and $4(C_{max} - l_{\mu_C})$ for σ_C^2.

Finally for the augmented Lagrangian function, it was not possible to establish

the bound above. Even if the compliance constraints were scaled by $1/L$ allowing a bound of the form c_1/L, c_1 would still scale up with the linear and quadratic penalties of the augmented Lagrangian function. The linear penalty is unbounded from above and the quadratic penalty grows exponentially during the optimization process. This means that no tight bound can be established. The experiments run were also consistent with this result where the diagonal estimation method was found to not work when solving a maximum compliance constrained problem using the augmented Lagrangian algorithm. A meaningless design was produced.

4.6 Optimization

4.6.1 Low rank loads

When minimizing the mean compliance only, the insensitivity of the correcting ratio to the design implies that one can minimize the mean compliance estimate instead of the exact one and get a reasonable design. This will be demonstrated in this section. In this section, a rank $R = 10$ is used and the trace estimation method is compared against the naive exact method where all the loading scenarios are enumerated. When minimizing the weighted sum of the mean and standard deviation of the compliance, a corrected estimator was used by calculating the correcting ratio of the mean and standard deviation estimators separately at the full ground mesh. Let the uncorrected mean compliance estimator be $\hat{\mu}_C$ and the uncorrected standard deviation estimator be $\hat{\sigma}_C$. The corrected estimator, $\hat{W}$, of the weighted sum of the

mean and standard deviation used was:

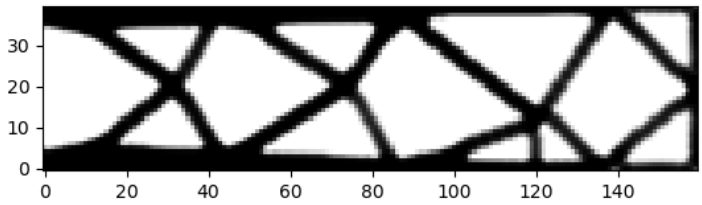
$$\hat{W} = \frac{\mu_C(\boldsymbol{x}_0)}{\hat{\mu}_C(\boldsymbol{x}_0)}\hat{\mu}_C(\boldsymbol{x}) + 2\frac{\sigma_C(\boldsymbol{x}_0)}{\hat{\sigma}_C(\boldsymbol{x}_0)}\hat{\sigma}_C(\boldsymbol{x}) \tag{4.52}$$

Only Hadamard basis probing vectors were used in this section.

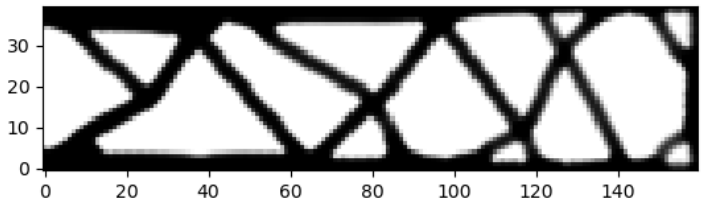
4.6.1.1 Mean compliance minimization

To demonstrate the effectiveness of the proposed approaches, the cantilever beam problem described in section 4.3 was solved using the proposed exact and approximate methods. Table 4.5 shows the statistics of the final optimal solutions obtained by minimizing the mean compliance subject to the volume fraction constraint using exact and trace estimation methods to evaluate the mean compliance. 10 Hadamard basis probing vectors were used in the trace estimator. The optimal topologies are shown in Figure 4-5.

While the designs obtained were different, both algorithms converged to reasonable designs in similar amounts of time. The convergence time shows that the convergence behavior was not affected by the use of an estimator in place of the original objective. However, the design produced by the trace estimation method was significantly worse than the exact method's which is to be expected since an approximate objective was minimized. Finally, note that the correcting ratio of the mean compliance estimator at the final design is 1.276 which is very close to the values shown in Figure 4-3.



(a) Exact method.



(b) Trace estimation method with 10 Hadamard basis probing vectors.

Figure 4-5: Optimal topologies of the mean compliance minimization problem using continuation SIMP.

Table 4.5: Summary statistics of the load compliances of the optimal solution of the mean compliance minimization problem using the exact and trace estimation methods to evaluate the mean compliance. 10 Hadamard basis probing vectors were used in the trace estimator.

Compliance Stat	Value	
	Exact	Trace estimation
μ_C (Nmm)	9397.4	7534.1 (uncorrected approx) / 9563.4 (exact)
σ_C (Nmm)	9689.0	9698.8
C_{max} (Nmm)	125440.7	124021.8
C_{min} (Nmm)	468.7	380.8
V	0.400	0.400
$Time$ (s)	25505.4	375.5

4.6.1.2 Mean-std compliance minimization

Similarly, Table 4.6 shows the statistics of the final solutions of the mean-std minimization problem solved using the exact and the corrected diagonal estimator method with 10 Hadamard basis probing vectors. The optimal topologies are shown in Figure 4-6. Both algorithms converged to reasonable, feasible designs. Additionally, as expected the exact and approximate mean-std minimization algorithms converged to solutions with lower compliance standard deviations but higher means compared to the exact and approximate mean minimization algorithms. It should be noted that the approximation error and non-convexity of the problem can sometimes lead this expectation to be unmet with the approximate approaches. The results indicate that the approximate method is able to converge in a fraction of the time it takes the exact method to converge because evaluating the function and its gradient using diagonal estimation requires $2N = 20$ linear system solves while the naive exact method requires 1000. This problem uses a low rank $\boldsymbol{F}$. The results of using the approximate methods proposed to solve a problem with a load matrix $\boldsymbol{F}$ of rank 100

Table 4.6: Summary statistics of the load compliances of the optimal solution of the mean-std compliance minimization problem using exact and the corrected diagonal estimation method with 10 Hadamard basis probing vectors to evaluate the mean-std compliance.

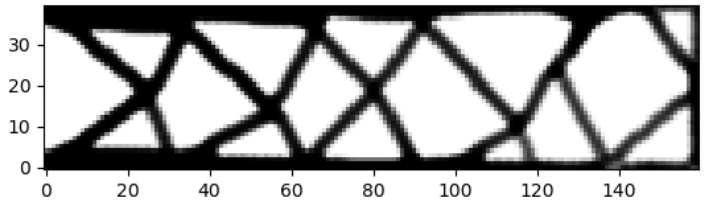
Compliance Stat	Value	
	Exact	Diagonal estimation
$\mu_C(Nmm)$	9871.6	9906.6 (corrected approx) / 9809.8 (exact)
$\sigma_C(Nmm)$	9264.0	9225.4 (corrected approx) / 9348.6 (exact)
$\mu_C + 2.0\sigma_C(Nmm)$	28407.6	28357.4 (corrected approx) / 28513.4 (exact)
C_{max} (Nmm)	117956.4	118853.9
C_{min} (Nmm)	530.2	451.9
V	0.400	0.400
Time (s)	2821.1	540.0

are shown in the next section.

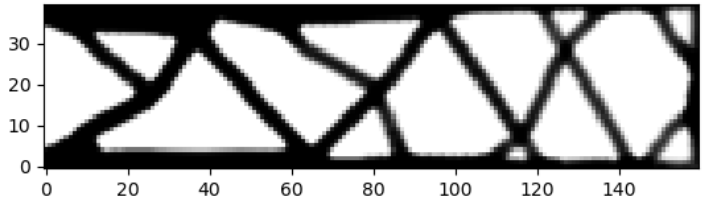
4.6.2 High rank loads

In this section, the 2D problems solved above will be solved using a load scenarios matrix $\boldsymbol{F}$ of rank $R = 100$ instead of 10. Additionally, the SVD-based method proposed by Tarek and Ray (2021) will be used instead of the naive approach used above. This will highlight the disadvantage of the SVD-based method when using a high rank $\boldsymbol{F}$.

The results are shown to be consistent with the low rank $\boldsymbol{F}$ where the corrected estimator's accuracy is significantly improved by a single correction at the beginning of the optimization. The histograms in Figures 4-8 and 4-9 also suggest that the correcting ratio is insensitive to the design. Figure 4-7 shows that the Hadamard probing vectors do not always give a better estimator than the Rademacher-distributed one for the mean but it is consistently better for the standard deviation. Figures 4-10 and 4-11 and tables 4.7 and 4.8 show the optimal topologies and results obtained using

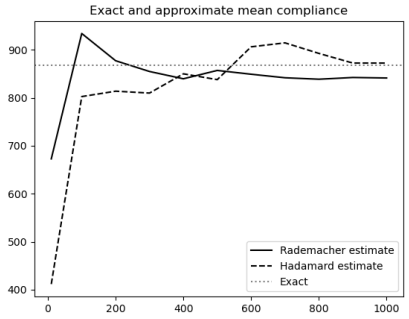


(a) Exact method.

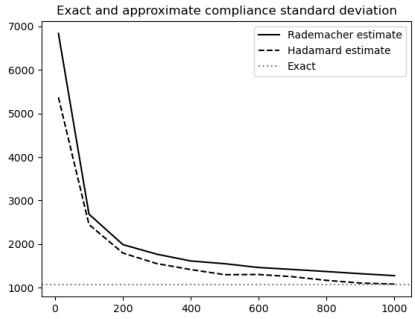


(b) Corrected diagonal estimation method with 10 Hadamard basis probing vectors using the estimator in Eq. 4.52.

Figure 4-6: Optimal topologies of the mean-std compliance minimization problem using continuation SIMP.



(a) Mean compliance estimate using different numbers of probing vectors in the trace estimation method.



(b) Compliance standard deviation estimate using different numbers of probing vectors in the diagonal estimation method.

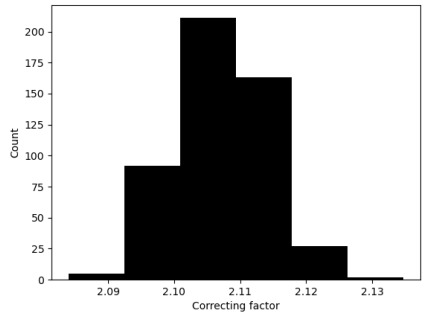
Figure 4-7: Accuracy profile of the trace and diagonal estimation methods for estimating the mean compliance and its standard deviation using 10, 100, 200, 300, 400, 500, 600, 700, 800, 900 and 1000 probing vectors for the high rank $\boldsymbol{F}$ case.

the exact and approximate methods. The results are consistent with the expectations.

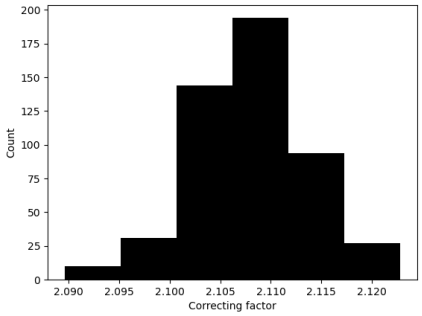
As shown in tables 4.7 and 4.8, the SVD-based methods are slower than the approximation schemes proposed when the rank of the loads is high. This is because the number of non-zero singular values will be 100 which is 10x the number of probing vectors used. In the mean compliance minimization, a 10 speedup is

Table 4.7: Summary statistics of the load compliances of the optimal solution of the mean compliance minimization problem with a high rank $\boldsymbol{F}$ using the exact and trace estimation methods to evaluate the mean compliance. 10 Hadamard basis probing vectors were used in the trace estimator.

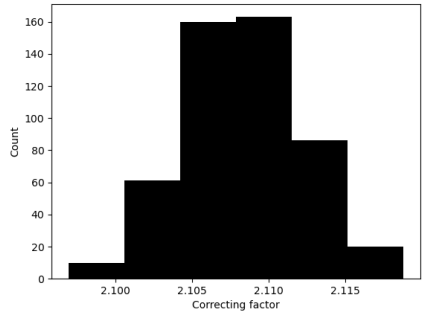
Compliance Stat	Value	
	Exact	Trace estimation
μ_C (Nmm)	2084.3	1094.1 (uncorrected approx) / 2226.4 (exact)
σ_C (Nmm)	2226.6	2412.0
C_{max} (Nmm)	15971.9	16793.3
C_{min} (Nmm)	171.0	152.0
V	0.400	0.400
$Time$ (s)	2123.0	485.3



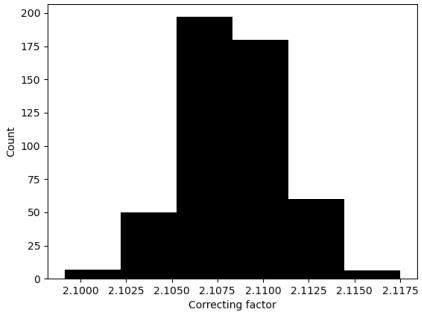
(a) Mean = 0.1



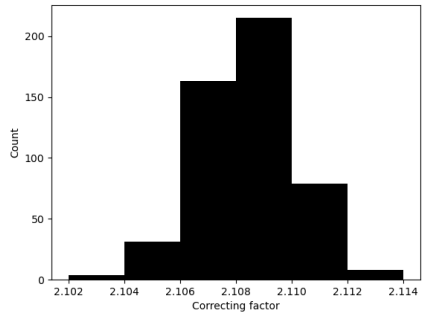
(b) Mean = 0.3



(c) Mean = 0.5

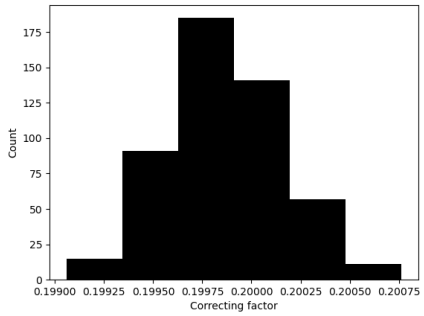


(d) Mean = 0.7

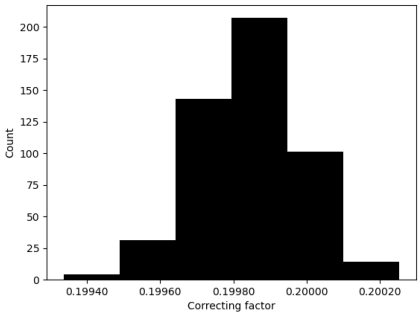


(e) Mean = 0.9

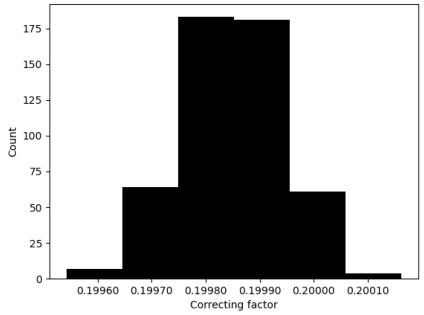
Figure 4-8: Histograms of the ratio between the exact mean compliance and the trace estimate using 10 Hadamard basis probing vectors for the high rank $\boldsymbol{F}$. In each figure, 500 designs were randomly sampled where each element's pseudo-density is sampled from a truncated normal distribution with the means indicated above and a standard deviation of 0.2, truncated between 0 and 1.



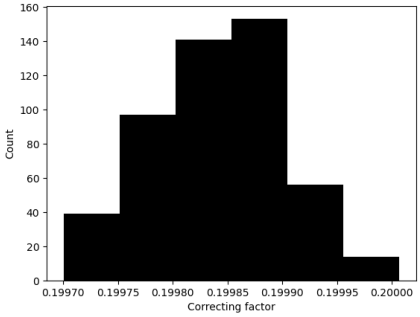
(a) Mean = 0.1



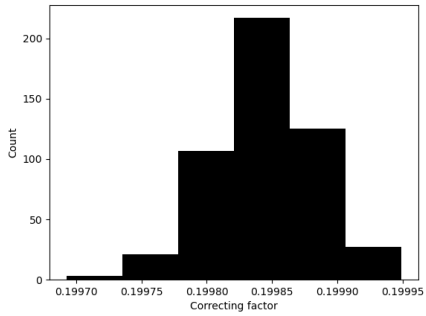
(b) Mean = 0.3



(c) Mean = 0.5

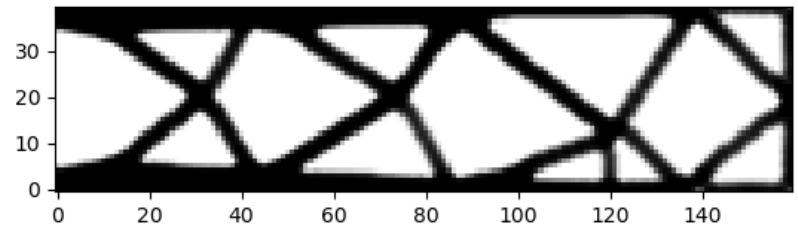


(d) Mean = 0.7

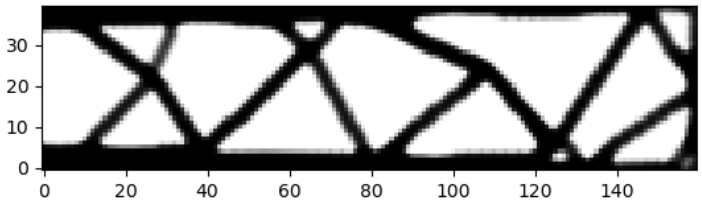


(e) Mean = 0.9

Figure 4-9: Histograms of the ratio between the exact compliance standard deviation and the estimate using 10 Hadamard basis probing vectors for the high rank $\boldsymbol{F}$ case. In each figure, 500 designs were randomly sampled where each element's pseudo-density is sampled from a truncated normal distribution with the means indicated above and a standard deviation of 0.2, truncated between 0 and 1.

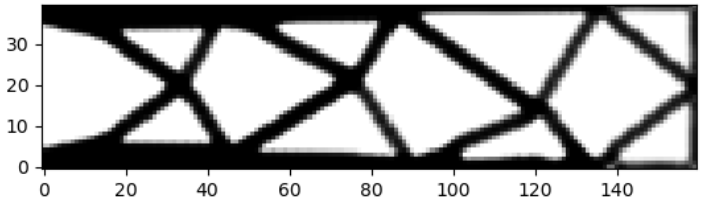


(a) Exact method.

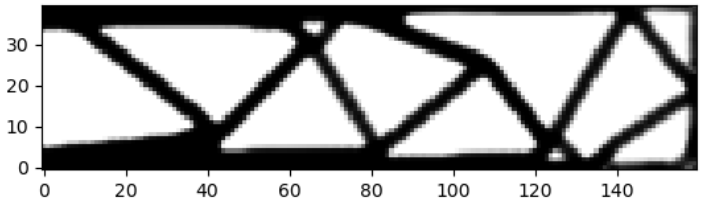


(b) Trace estimation method with 10 Hadamard basis probing vectors.

Figure 4-10: Optimal topologies of the mean compliance minimization problem with a high rank $\boldsymbol{F}$ using continuation SIMP.



(a) Exact method.



(b) Corrected diagonal estimation method with 10 Hadamard basis probing vectors using the estimator in Eq. 4.52.

Figure 4-11: Optimal topologies of the mean-std compliance minimization problem with high rank $\boldsymbol{F}$ using continuation SIMP.

Table 4.8: Summary statistics of the load compliances of the optimal solution of the mean-std compliance minimization problem with a high rank $\boldsymbol{F}$ using exact and the corrected diagonal estimation method with 10 Hadamard basis probing vectors to evaluate the mean-std compliance.

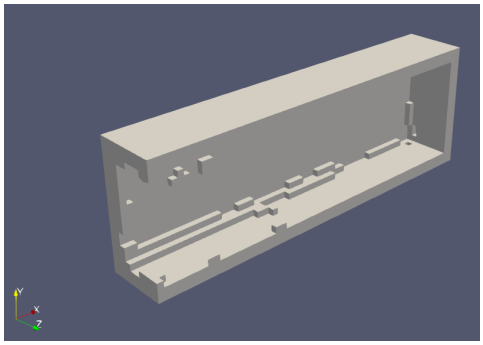
Compliance Stat	Value	
	Exact	Diagonal estimation
μ_C (Nmm)	2151.9	2336.8 (corrected approx) / 2147.5 (exact)
σ_C (Nmm)	2149.0	2195.7 (corrected approx) / 2234.2 (exact)
$\mu_C + 2.0\sigma_C$ (Nmm)	6450.6	6728.3 (corrected approx) / 6616.7 (exact)
C_{max} (Nmm)	15558.2	15559.4
C_{min} (Nmm)	187.4	161.6
V	0.400	0.400
Time (s)	6435.4	650.1

achieved which is consistent with the expectation. In the mean-std compliance minimization, the diagonal estimation method requires 20 linear system solves so only a factor of 5 speedup is achieved with the approximate method compared to the SVD-based method.

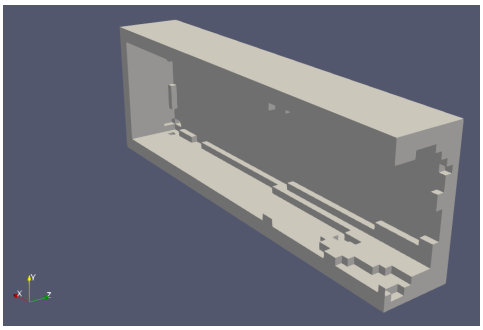
4.6.3 3D cantilever beam problem

A 3D version of the 2D cantilever beam test problem used above was also solved using the methods proposed in this chapter. The problem settings are described and the results are shown below.

A 60 mm x 20 mm x 20 mm 3D cantilever beam was used with hexahedral elements of cubic shape and side length of 1 mm. The loads $\boldsymbol{F}_1$, $\boldsymbol{F}_2$ and $\boldsymbol{F}_3$ were positioned at (60, 10, 10), (30, 20, 10) and (40, 0, 10) where the coordinates represent the length, height and depth respectively. A value of $R = 10$ was used. The remaining loads and multipliers were sampled from the same distributions as the 2D problem. A density filter radius of 3 mm was also used for the 3D problem. The

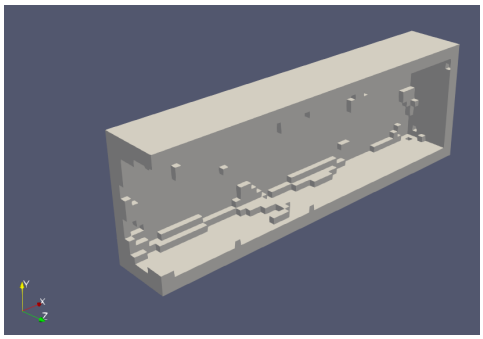


(a) Left half

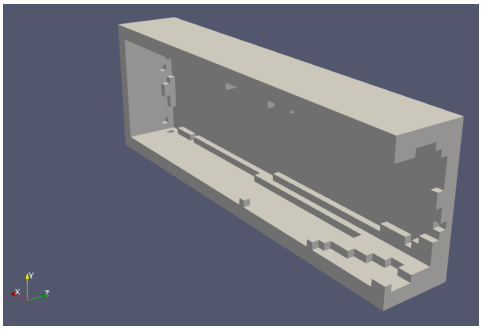


(b) Right half

Figure 4-12: Cut views of the optimal topologies of the 3D mean compliance minimization problem using exact method.



(a) Left half



(b) Right half

Figure 4-13: Cut views of the optimal topologies of the 3D mean compliance minimization problem using the trace estimation method.

same volume constrained mean compliance minimization and volume constrained mean-std compliance minimization problems were solved.

4.6.3.1 Mean compliance minimization

The 3D cantilever beam problem described above was solved using the proposed approximate methods with the objective of minimizing the mean compliance subject to a volume fraction constraint with a limit of 0.4. Table 4.9 shows the statistics of the final optimal solutions obtained by minimizing the mean compliance subject to the volume fraction constraint using the naive exact approach and the trace estimation

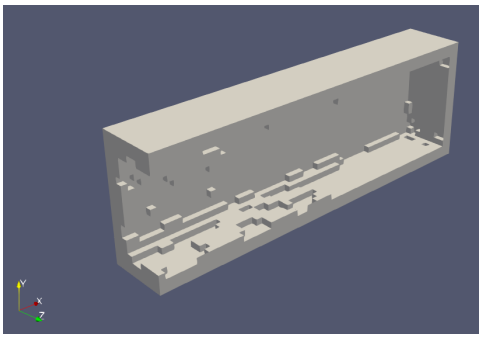
Table 4.9: Summary statistics of the load compliances of the optimal solution of the 3D mean compliance minimization problem using the exact and trace estimation methods to evaluate the mean compliance. 10 Hadamard basis probing vectors were used in the trace estimator.

Compliance Stat	Value	
	Exact	Trace estimation
μ_C (Nmm)	22072.1	24710.5 (uncorrected approx) / 22264.3 (exact)
σ_C (Nmm)	16628.7	17055.2
C_{max} (Nmm)	184055.0	190599.9
C_{min} (Nmm)	1785.8	1790.9
V	0.400	0.400
$Time$ (s)	167321.2	6595.2

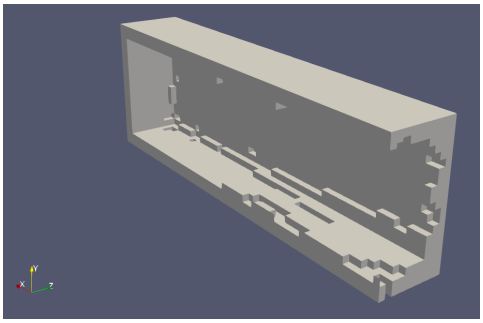
method to evaluate the mean compliance. 10 Hadamard basis probing vectors were used in the trace estimator. The optimal topologies are shown in Figures 4-12 and 4-13. Similar results to the 2D case can be observed where the designs obtained are different but somewhat reasonable. The proposed method converged in a small fraction of the time that the naive method took to converge. However, the design produced by the trace estimation method was worse than the exact method's which is to be expected since an approximate objective was minimized. Finally, note that the corrected estimate is close to the exact value.

4.6.3.2 Mean-std compliance minimization

Similarly, Table 4.10 shows the statistics of the final solutions of the 3D mean-std minimization problem solved using the naive exact approach and the corrected diagonal estimator method with 10 Hadamard basis probing vectors. The optimal topologies are shown in Figures 4-14 and 4-15. Both algorithms converged to reasonable and feasible designs. Additionally, as expected the exact mean-std

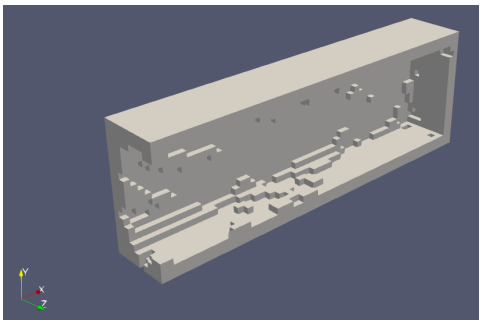


(a) Left half

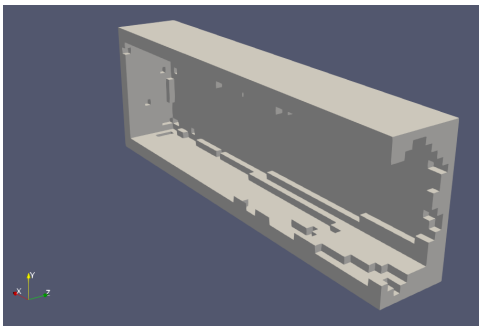


(b) Right half

Figure 4-14: Cut views of the optimal topologies of the 3D mean-std compliance minimization problem using the exact method.



(a) Left half



(b) Right half

Figure 4-15: Cut views of the optimal topologies of the 3D mean-std compliance minimization problem using the corrected diagonal estimation method.

Table 4.10: Summary statistics of the load compliances of the optimal solution of the mean-std compliance minimization problem using exact and the corrected diagonal estimation method with 10 Hadamard basis probing vectors to evaluate the mean-std compliance.

Compliance Stat	Value	
	Exact	Diagonal estimation
μ_C (Nmm)	22216.7	2240.8 (corrected approx) / 22145.9 (exact)
σ_C (Nmm)	16220.2	16501.4 (corrected approx) / 16510.0 (exact)
$\mu_C + 2.0\sigma_C$ (Nmm)	54848.8	55423.7 (corrected approx) / 55366.0 (exact)
C_{max} (Nmm)	176153.2	182209.8
C_{min} (Nmm)	1872.0	1850.4
V	0.400	0.400
Time (s)	39935.4	10949.3

minimization converged to a solution with a lower standard deviation but a higher mean compliance compared to the exact mean minimization. However, due to the approximation error and non-convexity of the problems, the exact and approximate mean-std algorithms converged to solutions with a lower mean and std compared to the approximate mean algorithm. Finally as expected, the exact method took significantly longer to converge than the diagonal estimation method.

4.7 Conclusion

In this chapter, two approximate methods were proposed to handle load uncertainty in compliance topology optimization problems where the uncertainty is described in the form of a set of finitely many loading scenarios. By re-formulating the function as a trace or diagonal estimation problem, significant performance improvements were achieved over the exact methods. Such improvement was demonstrated via complexity analysis and computational experiments. The methods proposed were shown to work well in practice while having a different time complexity profile.

5. Conclusion and future work

In this thesis, a number of things were achieved. Firstly, a reasonably comprehensive introduction to linear elasticity theory for topology optimization was presented. Special care was given to all the details to enable a 1-to-1 mapping from the text to the implementation. This is important when developing topology optimization codes and algorithms on unstructured meshes. This was followed by a presentation of the most common topology optimization algorithms and a detailed presentation of the most common nonlinear programming algorithms used in topology optimization. The author could not find any single reference that compiles detailed explanations of the theory behind the commonly used nonlinear programming algorithms in topology optimization, together with all their assumptions, strengths and weaknesses and tips on when to use each algorithm. Finally to conclude the introduction, a description of all the common paradigms for decision-making under uncertainty was presented and some basic concepts from computational linear algebra were reviewed.

Realizing the importance and prevalence of CSIMP in topology optimization, all the ways to adapt the penalty step in CSIMP found in literature were reviewed. A gap was found for a general penalty adaptation technique for CSIMP. A flexible and theoretically sound way to adapt penalties was proposed which gave significant speedups in the experiments run. Four common test problems from literature, three 2D and one 3D, were used to test the efficacy of the penalty adaptation with different

parameter settings. The main factors affecting the efficacy of the penalty adaptation in the CSIMP algorithm in reducing the number of FEA simulations needed to converge to the final solution were identified. The experimental results demonstrate a significant reduction in the number of FEA simulations required to reach the optimal solution in the decreasing tolerance continuation SIMP algorithm, with exponentially decaying tolerance, with little to no detriment in the objective value and the other metrics used. Finally, a mathematical and experimental treatment of the effect of x_{min} on the convergence of the SIMP algorithm was given with some recommendations for choosing a suitable x_{min}. These results were published in Tarek and Ray (2020). Some potential future work here is to perform more experiments on more problem classes as well as a larger benchmark set involving more problem classes. To this day, there is a lack of a standard benchmark set for topology optimization across algorithms and programming languages. Preparing such data set will be extremely valuable to the topology optimization society.

Handling load uncertainty significantly increases the computational cost of any algorithm. A comprehensive review of all the literature on handling uncertainty in compliance-based problems was therefore conducted and presented. And a number of exact methods were proposed to handle load uncertainty in compliance-based topology optimization problems where the uncertainty is described in the form of a set of finitely many loading scenarios. This includes mean compliance minimization or a constraint on the mean compliance, minimizing or constraining a weighted sum of the mean and standard deviation of the load compliances as well as minimizing or constraining the maximum load compliance for all the loading scenarios. By

detecting and exploiting low rank structures in the loading scenarios, significant performance improvements were achieved using some novel SVD-based methods. The computational complexities of the algorithms proposed were demonstrated and experiments were run to verify the efficacy of the proposed algorithms at reducing the computational cost of these classes of topology optimization problems. The methods presented here are fundamentally data-driven in the sense that no probability distributions or domains are assumed for the loading scenarios. This sets this work apart from most of the literature in the domain of stochastic and robust topology optimization where a distribution or domain is assumed. Additionally, the methods proposed here were shown to be particularly suitable with the augmented Lagrangian algorithm when dealing with maximum compliance constraints. This work was accepted for publication the Structural and Multidisciplinary Optimization journal. Some potential future work here includes developing algorithms for data-driven topology optimization under uncertainty for other classes of topology optimization problems.

Given that the exact methods for handling many loading scenarios require that a low rank exists, more efficient methods were developed when no such low rank exists. In particular, approximation schemes for the mean compliance and a class of scalar-valued functions of the load compliances were developed. The approximation schemes were based on a reformulation of the function approximated as a trace or diagonal estimation problem, opening the door to using many of the available methods for trace or diagonal estimation. The approximation methods were tested on a number of standard 2D and 3D benchmark problems using low and high rank

loading scenarios to solve mean compliance minimization as well as minimizing the weighted sum of the mean compliance and its standard deviation. Significant speedups were achieved compared to the naive method as well as the SVD-based method when the rank of the load matrix is high. This work was submitted to the Structural and Multidisciplinary Optimization journal as of the time of writing this thesis. There are a number of possible extensions to this work including trying or developing other trace and diagonal estimators. More generally, developing approximation schemes for other classes of topology optimization problems where there are finitely many loading scenarios is a largely untouched area of research.

Beside the potential future directions suggested above, there are many other potential future works in topology optimization. In particular, one promising direction to purse is the use of differentiable programming and automatic differentiation for topology optimization to simplify and generalize implementations of topology optimization algorithms in multi-physics applications. Some of these directions are currently pursued by the author.

Bibliography

Aage, N., Andreassen, E. and Lazarov, B. S. (2015), 'Topology optimization using PETSc: An easy-to-use, fully parallel, open source topology optimization framework', *Structural and Multidisciplinary Optimization* **51**(3), 565–572.

Aharon Ben-Tal, Laurent El Ghaoui and Nemirovski, A. (2009), *Robust Optimization*, Princeton University Press.

Allaire, G. and Jouve, F. (2004), 'A level-set method for vibration and multiple loads', *Computer Methods in Applied Mechanics and Engineering* pp. 1–28.

Allaire, G., Jouve, F. and Toader, A.-M. (2002), 'A level-set method for shape optimization', *Comptes Rendus Mathematique* **334**(12), 1125–1130.

Amir, O. (2015), 'Revisiting approximate reanalysis in topology optimization: on the advantages of recycled preconditioning in a minimum weight procedure', *Structural and Multidisciplinary Optimization* **51**(1), 41–57.

Amir, O. (2017), 'Stress-constrained continuum topology optimization: a new approach based on elasto-plasticity', *Structural and Multidisciplinary Optimization* **55**(5), 1797–1818.

Amir, O., Aage, N. and Lazarov, B. S. (2014), 'On multigrid-CG for efficient topology optimization', *Structural and Multidisciplinary Optimization* **49**(5), 815–829.

Amir, O., Bendsøe, M. P. and Sigmund, O. (2009), 'Approximate reanalysis in topol-

ogy optimization', *International Journal for Numerical Methods in Engineering* **78**(12), 1474–1491.

ApS, M. (2018), 'Mosek modeling cookbook'.

URL: *https://docs.mosek.com/MOSEKModelingCookbook-v2.pdf*

Bekas, C., Kokiopoulou, E. and Saad, Y. (2007), 'An estimator for the diagonal of a matrix', *Applied Numerical Mathematics* **57**(11-12), 1214–1229.

Bendsøe, M. P. (1989), 'Optimal shape design as a material distribution problem', *Structural Optimization* **1**(4), 193–202.

Bendsøe, M. P. and Kikuchi, N. (1988), 'Generating optimal topologies in structural design using a homogenization method', *Computer Methods in Applied Mechanics and Engineering* **71**(2), 197–224.

Bendsøe, M. P. and Sigmund, O. (2004), *Topology Optimization: Theory, Methods and Applications.*, 2 edn, Springer-Verlag Berlin Heidelberg.

Bertsekas, D. P. (1996), *Constrained Optimization and Lagrange Multiplier Methods*, Athena Scientific.

Bertsimas, D., Brown, D. B. and Caramanis, C. (2011), 'Theory and applications of robust optimization', *SIAM Review* **53**(3), 464–501.

Bezanson, J., Edelman, A., Karpinski, S. and Shah, V. B. (2014), 'Julia: A Fresh Approach to Numerical Computing'.

Bishop, C. M. (2006), *Pattern recognition and machine learning*, Springer.

Boyd, S. and Vandenberghe, L. (2009), *Convex Optimization*.

Brittain, K., Silva, M. and Tortorelli, D. A. (2012), 'Minmax topology optimization', *Structural and Multidisciplinary Optimization* **45**(5), 657–668.

Browne, P. (2013), Topology Optimization of Linear Elastic Structures, PhD thesis.

Bruggi, M. and Duysinx, P. (2012), 'Topology optimization for minimum weight with compliance and stress constraints', *Structural and Multidisciplinary Optimization* **46**(3), 369–384.

Bruggi, M. and Verani, M. (2011), 'A fully adaptive topology optimization algorithm with goal-oriented error control', *Computers and Structures* **89**(15-16), 1481–1493.

URL: *http://dx.doi.org/10.1016/j.compstruc.2011.05.003*

Chen, S., Lee, S. and Chen, W. (2010), 'Level set based robust shape and topology optimization under random field uncertainties', *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2009, DETC2009* **5**(PART B), 1295–1305.

Choi, S.-K., Grandhi, R. V. and Canfield, R. A. (2007), *Reliability-based Structural Design*, Springer-Verlag.

Collet, M., Bruggi, M. and Duysinx, P. (2017), 'Topology optimization for minimum weight with compliance and simplified nominal stress constraints for fatigue resistance', *Structural and Multidisciplinary Optimization* **55**(3), 839–855.

Cuellar, N., Pereira, A., Menezes, I. F. and Cunha, A. (2018), 'Non-intrusive polynomial chaos expansion for topology optimization using polygonal meshes', *Journal of the Brazilian Society of Mechanical Sciences and Engineering* **40**(12), 1–18. **URL:** *https://doi.org/10.1007/s40430-018-1464-2*

Dadalau, A., Hafla, A. and Verl, A. (2009), 'A new adaptive penalization scheme for topology optimization', *Production Engineering* **3**(4-5), 427–434.

de Berg, M., Cheong, O., van Kreveld, M. and Overmars, M. (2008), *Computational Geometry*, 3 edn, Springer Berlin Heidelberg, Berlin, Heidelberg. **URL:** *http://link.springer.com/10.1007/978-3-540-77974-2*

Deepak, S. R., Dinesh, M., Sahu, D. K. and Ananthasuresh, G. K. (2009), 'A Comparative Study of the Formulations and Benchmark Problems for the Topology Optimization of Compliant Mechanisms', *Journal of Mechanisms and Robotics* **1**(1), 011003.

Deng, S. and Suresh, K. (2017), 'Topology optimization under thermo-elastic buckling', *Structural and Multidisciplinary Optimization* **55**(5), 1759–1772.

Díaz, A. and Sigmund, O. (1995), 'Checkerboard patterns in layout optimization', *Structural Optimization* **10**(1), 40–45.

Dunning, P. D. and Kim, H. A. (2013), 'Robust topology optimization: Minimization of expected and variance of compliance', *AIAA Journal* **51**(11), 2656–2664.

Dunning, P. D., Kim, H. A. and Mullineux, G. (2011), 'Introducing loading uncertainty in topology optimization', *AIAA Journal* **49**(4), 760–768.

Elishakoff, I., Haftka, R. T. and Fang, J. (1994), 'Structural design under bounded uncertainty-Optimization with anti-optimization', *Computers and Structures* **53**(6), 1401–1405.

Fleury, C. (1989), 'CONLIN: An efficient dual optimizer based on convex approximation concepts', *Structural Optimization* **1**(2), 81–89.

Gao Xingjun, Li Lijuan and Ma Haitao (2017), 'An Adaptive Continuation Method for Topology Optimization of Continuum Structures Considering Buckling Constraints', *International Journal of Applied Mechanics* **09**(07), 1750092.

Garcia-Lopez, N. P., Sanchez-Silva, M., Medaglia, A. L. and Chateauneuf, A. (2013), 'An improved robust topology optimization approach using multiobjective evolutionary algorithms', *Computers and Structures* **125**, 1–10.
URL: *http://dx.doi.org/10.1016/j.compstruc.2013.04.025*

Golub, G. H. and Loan, C. F. V. (1996), *Matrix Computations*, Vol. 10.

Guest, J. K. and Igusa, T. (2008), 'Structural optimization under uncertain loads and nodal locations', *Computer Methods in Applied Mechanics and Engineering* **198**(1), 116–124.
URL: *http://dx.doi.org/10.1016/j.cma.2008.04.009*

Guest, J. K., Prévost, J. H. and Belytschko, T. (2004), 'Achieving minimum length scale in topology optimization using nodal design variables and projection functions', *International Journal for Numerical Methods in Engineering* **61**(2), 238–254.

Guirguis, D. and Aly, M. F. (2016), 'A derivative-free level-set method for topology optimization', *Finite Elements in Analysis and Design* **120**, 41–56.

Guo, S. X. and Lu, Z. Z. (2015), 'A non-probabilistic robust reliability method for analysis and design optimization of structures with uncertain-but-bounded parameters', *Applied Mathematical Modelling* **39**(7), 1985–2002.
URL: *http://dx.doi.org/10.1016/j.apm.2014.10.026*

Hager, W. W. and Zhang, H. (2006), 'Algorithm 851: CG_DESCENT, a conjugate gradient method with guaranteed descent', *ACM Transactions on Mathematical Software (TOMS)* **32**(1), 113–137.
URL: *http://portal.acm.org/citation.cfm?id=1132979*

Holmberg, E., Thore, C. J. and Klarbring, A. (2015), 'Worst-case topology optimization of self-weight loaded structures using semi-definite programming', *Structural and Multidisciplinary Optimization* **52**(5), 915–928.

Huang, X. and Xie, Y. M. (2010), 'A further review of ESO type methods for topology optimization', *Structural and Multidisciplinary Optimization* **41**(5), 671–683.

Hutchinson, M. F. (1990), 'A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines', *Communications in Statistics - Simulation and Computation* **19**(2), 433–450.

Jalalpour, M. and Tootkaboni, M. (2016), 'An efficient approach to reliability-based topology optimization for continua under material uncertainty', *Structural and Multidisciplinary Optimization* **53**(4), 759–772.

Jung, H. S. and Cho, S. (2004), 'Reliability-based topology optimization of geometrically nonlinear structures with loading and material uncertainties', *Finite Elements in Analysis and Design* **41**(3), 311–331.

K Mogensen, P. and N Riseth, A. (2018), 'Optim: A mathematical optimization package for Julia', *Journal of Open Source Software* **3**(24), 615.
URL: *http://joss.theoj.org/papers/10.21105/joss.00615*

Kang, Z. and Luo, Y. (2009), 'Non-probabilistic reliability-based topology optimization of geometrically nonlinear structures using convex models', *Computer Methods in Applied Mechanics and Engineering* **198**(41-44), 3228–3238.
URL: *http://dx.doi.org/10.1016/j.cma.2009.06.001*

Keshavarzzadeh, V., Fernandez, F. and Tortorelli, D. A. (2017), 'Topology optimization under uncertainty via non-intrusive polynomial chaos expansion', *Computer Methods in Applied Mechanics and Engineering* **318**, 120–147.
URL: *http://dx.doi.org/10.1016/j.cma.2017.01.019*

Kharmanda, G. and Olhoff, N. (2002), 'Reliability-Based Topology Optimization as a New Strategy to Generate Different Structural Topologies', *15th Nordic Seminar on Computational Mechanics* (January).

Kharmanda, G., Olhoff, N., Mohamed, A. and Lemaire, M. (2004), 'Reliability-based topology optimization', *Structural and Multidisciplinary Optimization* **26**(5), 295–307.

Kim, C., Wang, S., Bae, K. R., Moon, H. and Choi, K. K. (2006), 'Reliability-based

topology optimization with uncertainties', *Journal of Mechanical Science and Technology* **20**(4), 494–504.

Kim, S. R., Lee, W. G., Park, J. Y., Yu, J. S. and Han, S. Y. (2008), 'Reliability-based topology optimization using reliability index approach', *ICEM 2008: International Conference on Experimental Mechanics 2008* **7375**(August 2009), 73752W.

Kim, S.-R., Park, J.-Y., Lee, W.-G., Yu, J.-S. and Han, S.-Y. (2007), 'Reliability-Based Topology Optimization Based on Evolutionary Structural Optimization', *International Journal of Mechanical Systems Science and Engineering* **1**(3), 168–172.

Kočvara, M. and Stingl, M. (2004), 'Solving nonconvex SDP problems of structural optimization with stability control', *Optimization Methods and Software* **19**(5 SPEC. ISS.), 595–609.

Kriegesmann, B. and Lüdeker, J. K. (2019), 'Robust compliance topology optimization using the first-order second-moment method', *Structural and Multidisciplinary Optimization* **60**(1), 269–286.

Lambe, A. B. and Czekanski, A. (2018), 'Topology optimization using a continuous density field and adaptive mesh refinement', *International Journal for Numerical Methods in Engineering* **113**(3), 357–373.

Lee, E., James, K. A. and Martins, J. R. (2012), 'Stress-constrained topology

optimization with design-dependent loading', *Structural and Multidisciplinary Optimization* **46**(5), 647–661.

Lian, H., Christiansen, A. N., Tortorelli, D. A., Sigmund, O. and Aage, N. (2017), 'Combined shape and topology optimization for minimization of maximal von Mises stress', *Structural and Multidisciplinary Optimization* **55**(5), 1541–1557.

Liu, J. T. and Gea, H. C. (2018), 'Robust topology optimization under multiple independent unknown-but-bounded loads', *Computer Methods in Applied Mechanics and Engineering* **329**, 464–479.
URL: *https://doi.org/10.1016/j.cma.2017.09.033*

Liu, J. and Wen, G. (2018), 'Continuum topology optimization considering uncertainties in load locations based on the cloud model', *Engineering Optimization* **50**(6), 1041–1060.
URL: *https://doi.org/10.1080/0305215X.2017.1361417*

Liu, X., Yi, W.-J., Li, Q. and Shen, P.-S. (2008), 'Genetic evolutionary structural optimization', *Journal of Constructional Steel Research* **64**(3), 305–311.

Lombardi, M. and Haftka, R. T. (1998), 'Anti-optimization technique for structural design under load uncertainties', *Computer Methods in Applied Mechanics and Engineering* **157**(1-2), 19–31.

Luo, Y., Kang, Z., Luo, Z. and Li, A. (2009), 'Continuum topology optimization with non-probabilistic reliability constraints based on multi-ellipsoid convex model', *Structural and Multidisciplinary Optimization* **39**(3), 297–310.

Martínez-Frutos, J. and Herrero-Pérez, D. (2016), 'Large-scale robust topology optimization using multi-GPU systems', *Computer Methods in Applied Mechanics and Engineering* **311**, 393–414.

Martínez-Frutos, J., Herrero-Pérez, D., Kessler, M. and Periago, F. (2018), 'Risk-averse structural topology optimization under random fields using stochastic expansion methods', *Computer Methods in Applied Mechanics and Engineering* **330**, 180–206.

URL: *https://doi.org/10.1016/j.cma.2017.10.026*

Martínez, J. M. (2005), 'A note on the theoretical convergence properties of the SIMP method', *Structural and Multidisciplinary Optimization* **29**(4), 319–323.

Maute, K. and Ramm, E. (1995), 'Adaptive topology optimization', *Structural Optimization* **10**(2), 100–112.

Maute, K. and Ramm, E. (1998), 'Adaptive Topology Optimization of Elastoplastic Structures', *Structural Optimization* **15**, 81–91.

Munk, D. J., Vio, G. A. and Steven, G. P. (2017), 'A simple alternative formulation for structural optimisation with dynamic and buckling objectives', *Structural and Multidisciplinary Optimization* **55**(3), 969–986.

Neves, M., Rodrigues, H. and Guedes, J. (1995), 'Generalized topology design of structures with a buckling load criterion', *Structural Optimization* **10**, 71–78.

Nguyen, T. H., Song, J. and Paulino, G. H. (2011), 'Single-loop system reliability-

based topology optimization considering statistical dependence between limit-states', *Structural and Multidisciplinary Optimization* **44**(5), 593–611.

Nocedal, J. and Wright, S. J. (2006), *Numerical Optimization*, Springer Sc.

Oest, J. and Lund, E. (2017), 'Topology optimization with finite-life fatigue constraints', *Structural and Multidisciplinary Optimization* **56**(5), 1045–1059.

Osher, S. J. (1988), 'Fronts Propagating with Curvature Dependent Speed', *Computational Physics* **79**(1), 1–5.

Ouyang, G., Zhang, X. and Kuang, Y. (2008), 'Reliability-based topology optimization of continuous structures', *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)* (50375051), 7021–7025.

Payten, W. M. and Law, M. (1998), 'Generalized shape optimization using stress constraints under multiple load cases', *Structural Optimization* **15**(3-4), 269–274.

Pereira, J. T., Fancello, E. A. and Barcellos, C. S. (2004), 'Topology optimization of continuum structures with material failure constraints', **66**, 50–66.

Rahmatalla, S. and Swan, C. (2004), 'A Q4/Q4 continuum structural topology optimization implementation', *Structural and Multidisciplinary Optimization* **27**(1-2), 130–135.

Rahmatalla, S. and Swan, C. C. (2003), 'Continuum Topology Optimization of Buckling-Sensitive Structures', *AIAA Journal* **41**(6), 1180–1189.

Rietz, A. (2001), 'Sufficiency of a finite exponent in SIMP (power law) methods', *Structural and Multidisciplinary Optimization* **21**(2), 159–163.

Rojas-Labanda, S. and Stolpe, M. (2015*a*), 'Automatic penalty continuation in structural topology optimization', *Structural and Multidisciplinary Optimization* **52**(6), 1205–1221.

Rojas-Labanda, S. and Stolpe, M. (2015*b*), 'Benchmarking optimization solvers for structural topology optimization', *Structural and Multidisciplinary Optimization* **52**(3), 527–547.

Rozvany, G. I. N. (2009), 'A critical review of established methods of structural topology optimization', *Structural and Multidisciplinary Optimization* **37**(3), 217–237.

Salazar de Troya, M. A. and Tortorelli, D. A. (2018), 'Adaptive mesh refinement in stress-constrained topology optimization', *Structural and Multidisciplinary Optimization* **58**(6), 2369–2386.

Sandgren, E. (1990), 'Nonlinear integer and discrete programming in mechanical design optimization', *Journal of Mechanisms, Transmissions, and Automation in Design* **112**(2), 223–229.

Sandgren, E., Jensen, E. and W. Welton, J. (1990), 'Topological design of structural components using genetic optimization methods.', *Sensitivity Analysis and Optimization with Numerical Methods, ASME-AMD* **115**, 31–43.

Sartipizadeh, H. and Vincent, T. L. (2016), 'Computing the Approximate Convex Hull in High Dimensions'.

URL: *http://arxiv.org/abs/1603.04422*

Sato, Y., Izui, K., Yamada, T. and Nishiwaki, S. (2017), 'Pareto frontier exploration in multiobjective topology optimization using adaptive weighting and point selection schemes', *Structural and Multidisciplinary Optimization* **55**(2), 409–422.

Shapiro, A., Dentcheva, D. and Ruszczyński, A. (2009), *Lectures on Stochastic Programming*.

Sigmund, O. (1997), 'On the design of compliant mechanisms using topology optimization', *Mechanics of Structures and Machines* **25**(4), 493–524.

Sigmund, O. (2001), 'A 99 line topology optimization code written in matlab', *Structural and Multidisciplinary Optimization* **21**(2), 120–127.

Sigmund, O. and Petersson, J. (1998), 'Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima', *Structural Optimization* **16**(1), 68–75.

Silva, M., Tortorelli, D. A., Norato, J. A., Ha, C. and Bae, H. R. (2010), 'Component and system reliability-based topology optimization using a single-loop method', *Structural and Multidisciplinary Optimization* **41**(1), 87–106.

Stainko, R. (2006), 'An adaptive multilevel approach to the minimal compliance problem in topology optimization', *Communications in Numerical Methods in Engineering* **22**(2), 109–118.

Stolpe, M. and Svanberg, K. (2001*a*), 'An alternative interpolation scheme for minimum compliance topology optimization', *Structural and Multidisciplinary Optimization* **22**(2), 116–124.

Stolpe, M. and Svanberg, K. (2001*b*), 'On the trajectories of penalization methods for topology optimization', *Structural and Multidisciplinary Optimization* **21**(2), 128–139.

Suresh, K. (2010), 'A 199-line Matlab code for Pareto-optimal tracing in topology optimization', *Structural and Multidisciplinary Optimization* **42**(5), 665–679.

Svanberg, K. (1987), 'The method of moving asymptotes - a new method for structural optimization', *International Journal for Numerical Methods in Engineering* **24**(2), 359–373.

Svanberg, K. (1994), 'On the convexity and concavity of compliances', *Structural Optimization* **7**(1-2), 42–46.

Svanberg, K. (2002), 'A Class of Globally Convergent Optimization Methods Based on Conservative Convex Separable Approximations', *SIAM Journal on Optimization* **12**(2), 555–573.

Tarek, M. and Ray, T. (2020), 'Adaptive continuation solid isotropic material with penalization for volume constrained compliance minimization', *Computer Methods in Applied Mechanics and Engineering* **363**, 112880.
URL: *https://www.sciencedirect.com/science/article/pii/S0045782520300621*

Tarek, M. and Ray, T. (2021), 'Robust and stochastic compliance-based topology optimization with finitely many loading scenarios'.

Tempo, R., Bai, E. W. and Dabbene, F. (1996), 'Probabilistic robustness analysis: explicit bounds for the minimum number of samples', *Proceedings of the IEEE Conference on Decision and Control* **3**(December), 2355–3592.

Thore, C. J., Holmberg, E. and Klarbring, A. (2017), 'A general framework for robust topology optimization under load-uncertainty including stress constraints', *Computer Methods in Applied Mechanics and Engineering* **319**, 1–18.
URL: *http://dx.doi.org/10.1016/j.cma.2017.02.015*

Tu, J., Choi, K. K. and Park, Y. H. (1999), 'A new study on reliability- based design optimization', *Journal of Mechanical Design, Transactions of the ASME* **121**(4), 557–564.

Valdez, S. I., Botello, S., Ochoa, M. A., Marroquín, J. L. and Cardoso, V. (2017), 'Topology Optimization Benchmarks in 2D: Results for Minimum Compliance and Minimum Volume in Planar Stress Problems', *Archives of Computational Methods in Engineering* **24**(4), 803–839.

Wächter, A. and Biegler, L. T. (2006), *On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming*, Vol. 106.

Wang, L., Liang, J. and Wu, D. (2018), 'A non-probabilistic reliability-based topol-

ogy optimization (NRBTO) method of continuum structures with convex uncertainties', *Structural and Multidisciplinary Optimization* **58**(6), 2601–2620.

Wang, L., Liu, D., Yang, Y. and Hu, J. (2019), 'Novel methodology of Non-probabilistic Reliability-based Topology Optimization (NRBTO) for multi-material layout design via interval and convex mixed uncertainties', *Computer Methods in Applied Mechanics and Engineering* **346**, 550–573.
URL: *https://doi.org/10.1016/j.cma.2018.11.035*

Wang, L., Liu, D., Yang, Y., Wang, X. and Qiu, Z. (2017), 'A novel method of non-probabilistic reliability-based topology optimization corresponding to continuum structures with unknown but bounded uncertainties', *Computer Methods in Applied Mechanics and Engineering* **326**, 573–595.
URL: *http://dx.doi.org/10.1016/j.cma.2017.08.023*

Wang, L., Xia, H., Zhang, X. and Lv, Z. (2019), 'Non-probabilistic reliability-based topology optimization of continuum structures considering local stiffness and strength failure', *Computer Methods in Applied Mechanics and Engineering* **346**, 788–809.
URL: *https://doi.org/10.1016/j.cma.2018.09.021*

Wang, M. Y., Wang, X. and Guo, D. (2003), 'A level set method for structural topology optimization', *Computer Methods in Applied Mechanics and Engineering* **192**(1-2), 227–246.

Wang, S., de Sturler, E. and Paulino, G. H. (2010), 'Dynamic Adaptive Mesh

Refinement for Topology Optimization', pp. 1–23.

URL: *http://arxiv.org/abs/1009.4975*

Wang, X., Wang, M. and Guo, D. (2004), 'Structural shape and topology optimization in a level-set-based framework of region representation', *Structural and Multidisciplinary Optimization* **27**(1-2), 1–19.

Wang, Y., Kang, Z. and He, Q. (2014), 'Adaptive topology optimization with independent error control for separated displacement and density fields', *Computers and Structures* **135**, 50–61.

URL: *http://dx.doi.org/10.1016/j.compstruc.2014.01.008*

Wang, Y. Q., He, J. J., Luo, Z. and Kang, Z. (2013), 'An adaptive method for high-resolution topology design', *Acta Mechanica Sinica/Lixue Xuebao* **29**(6), 840–850.

Xie, Y. and Steven, G. (1992), Shape and layout optimization via an evolutionary procedure., *in* 'Proceedings of the International Conference Comput. Eng. (Hong Kong), Hong Kong University', p. 421.

Yang, R. J. and Chen, C. J. (1996), 'Stress-based topology optimization', *Structural Optimization* **12**(2-3), 98–105.

Yang, X., Xie, Y., Steven, G. and Querin, O. (1998), Bi-directional evolutionary structural optimization., *in* 'Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium Multidisc Anal. Optim (St. Louis)', pp. 1449–1457.

Yin, H., Yu, D. and Xia, B. (2018), 'Reliability-based topology optimization for structures using fuzzy set model', *Computer Methods in Applied Mechanics and Engineering* **333**, 197–217.
URL: *https://doi.org/10.1016/j.cma.2018.01.019*

Youn, B. D. and Choi, K. K. (2004), 'Selecting probabilistic approaches for realiability-based design optimization', *AIAA Journal* **42**(1), 124–131.

Yu, X., Chang, K. H. and Choi, K. K. (1998), 'Probabilistic structural durability prediction', *AIAA Journal* **36**(4), 628–637.

Zargham, S., Ward, T. A., Ramli, R. and Badruddin, I. A. (2016), 'Topology optimization: a review for structural designs under vibration problems', *Structural and Multidisciplinary Optimization* **53**(6), 1157–1177.

Zhang, X., Kang, Z. and Zhang, W. (2016), 'Robust topology optimization for dynamic compliance minimization under uncertain harmonic excitations with inhomogeneous eigenvalue analysis', *Structural and Multidisciplinary Optimization* **54**(6), 1469–1484.

Zhang, X. S., de Sturler, E. and Paulino, G. H. (2017), 'Stochastic sampling for deterministic structural topology optimization with many load cases: Density-based and ground structure approaches', *Computer Methods in Applied Mechanics and Engineering* **325**, 463–487.
URL: *http://dx.doi.org/10.1016/j.cma.2017.06.035*

Zhao, J. and Wang, C. (2014*a*), 'Robust structural topology optimization under

random field loading uncertainty', *Structural and Multidisciplinary Optimization* **50**(3), 517–522.

Zhao, J. and Wang, C. (2014*b*), 'Robust topology optimization under loading uncertainty based on linear elastic theory and orthogonal diagonalization of symmetric matrices', *Computer Methods in Applied Mechanics and Engineering* **273**, 204–218.

URL: *http://dx.doi.org/10.1016/j.cma.2014.01.018*

Zhao, Q., Chen, X., Ma, Z. and Lin, Y. (2016), 'A Comparison of Deterministic, Reliability-Based Topology Optimization under Uncertainties', *Acta Mechanica Solida Sinica* **29**(1), 31–45.

Zheng, J., Luo, Z., Jiang, C., Ni, B. and Wu, J. (2018), 'Non-probabilistic reliability-based topology optimization with multidimensional parallelepiped convex model', *Structural and Multidisciplinary Optimization* **57**(6), 2205–2221.

Zheng, S., Zhao, X., Yu, Y. and Sun, Y. (2017), 'The approximate reanalysis method for topology optimization under harmonic force excitations with multiple frequencies', *Structural and Multidisciplinary Optimization* **56**(5), 1185–1196.

Zhou, M. and Rozvany, G. I. N. (2001), 'On the validity of ESO type methods in topology optimization', *Structural and Multidisciplinary Optimization* **21**(1), 80–83.

.1 Partial derivative of the inverse quadratic form

In this section, it will be shown that the i^{th} partial derivative of:

$$f(\boldsymbol{x}) = \boldsymbol{v}^T(\boldsymbol{A}(\boldsymbol{x}))^{-1}\boldsymbol{v} \tag{1}$$

is

$$\frac{\partial f}{\partial x_i} = -\boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y}^T \tag{2}$$

where $\boldsymbol{A}$ is a matrix-valued function of $\boldsymbol{x}$, $\boldsymbol{v}$ is a constant vector and $\boldsymbol{y} = \boldsymbol{A}^{-1}\boldsymbol{v}$ is a an implicit function of $\boldsymbol{x}$ because $\boldsymbol{A}$ is a function of $\boldsymbol{x}$.

$$\boldsymbol{v} = \boldsymbol{A}\boldsymbol{y} \tag{3}$$

$$\boldsymbol{0} = \boldsymbol{A}\frac{\partial \boldsymbol{y}}{\partial x_i} + \frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{4}$$

$$\frac{\partial y}{\partial x_i} = -\boldsymbol{A}^{-1}\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{5}$$

$$f(\boldsymbol{x}) = \boldsymbol{v}^T\boldsymbol{A}^{-1}\boldsymbol{v} \tag{6}$$

$$= \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y} \tag{7}$$

$$\frac{\partial f}{\partial x_i} = 2\boldsymbol{y}^T\boldsymbol{A}\frac{\partial \boldsymbol{y}}{\partial x_i} + \boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{8}$$

$$= -2\boldsymbol{y}^T\boldsymbol{A}\boldsymbol{A}^{-1}\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} + \boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{9}$$

$$= -2\boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} + \boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{10}$$

$$= -\boldsymbol{y}^T\frac{\partial \boldsymbol{A}}{\partial x_i}\boldsymbol{y} \tag{11}$$