Inverse design in Quantum Acoustics

Designing a Phononic Beamsplitter using Inverse Design with Adjoint Simulation

David Hambraeus

[DRAFT]

compiled 2023-05-27, 15:30

Inverse design in Quantum Acoustics:
Designing a Phononic Beamsplitter using Inverse Design with Adjoint Simulation

David Hambraeus
Department of Microtechnology and Nanoscience
Chalmers University of Technology

Abstract

Quantum acoustic devices could enable and improve a broad range of functions in the realm of quantum computing and sensing as well as classical devices. However, such devices are currently often designed by hand, combined with brute force parameter sweeps, which severely limits the designs that can be investigated. This work presents a method for inverse-design of quantum acoustic devices. At the heart of the method lies a fast calculation of the gradient using the adjoint method. I show that this method is theoretically applicable to acoustic devices as well, though implementing it in practice has yielded mixed results. As a test I show that using this method to design a defect for maximum transmission in a simple periodically patterned phononic waveguide yielded a 92 % transmission rate. REVISE PERCENTAGE WITH NEW DATA. As a proof of concept, I also attempted to design an acoustic beamsplitter. The algorithm manages to design performant beamsplitters, but it fails to converge. Further research is required to find why. The two most likely reasons are that the meshing is too coarse, or that the function shape order is too low. In any case, if these problems can be solved, this method looks promising for use in the design of future quantum acoustic devices.

Keywords: lorem, ipsum, dolor, sit, amet, consectetur, adipisicing, elit, sed, do.

Acknowledgements

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

David Hambraeus, Gothenburg, May 2023

Table of contents

List of figures

List of tables

Acronyms

ADAM Adaptive Moment Estimation. vi, 15, 16

GD Gradient Descent. vi, 15, 16

PML Perfectly Matched Layer. vi, 18, 20–23

List of Todos

Todonotes are organized as follows:

General comment / question

Things that could be done now, no further simulations/consultation needed

Things that could be done now but I am not sure if I should, or how to do it

Things that can't be done yet because they depend on other things, e.g. results

Citation needed

1. Introduction

After thesis is done, check that I've used cref and not ref

References before the dot or after? And space or no space?

ctrl+F for all the times when I write I / we and choose one of them...

Write about why I am using *silicon*

Introduction to quantum acoustics...I need to read more literature I think

Restructure a little... I would like to talk about inverse design first and quantum acoustics second. Talk about how inverse design is a concept that has been applied to nanophotonics but not to quantum acoustics yet. Then talk about why we care about quantum acoustics. It feels a little bit forced to do it in that order though, talking about quantum acoustics first might be a better idea, since that would naturally lead one to introduce the problem of design.

In recent years, the research into quantum devices of different kinds has significantly intensified. Much of it is in one way or another connected to the construction or operation of a quantum computer. Though many people are focusing on superconducting circuits, where quanta of microwave-frequency photons are manipulated, there has also been a growing interest in a different medium for quantum information: sound. More precisely, acoustic waves in solid materials. Just like how light, or electromagnetic waves, come in quantized packets of energy called photons, so too does acoustic waves and we call those packets phonons. Just recently, in 2019, researchers coupled an acoustic resonator to a transmon qubit and were able to directly measure the presence or absence of single phonons.

Possible applications of quantum acoustic devices are many. One of them is it's use in quantum memory. Regular computers have both memory and a processing unit that retrieves data from memory, applies operations on it, and then returns it to memory. Keeping all of the data at the same place where the computing happens would be very inefficient. The same goes for quantum computers: storing all of the quantum information in the same place where the computation happens is probably not a scalable plan. Another application of quantum acoustics worth mentioning is coherent transduction between microwave and optical photons. This would enable

communication between physically separated superconducting circuit based quantum computers.

One important problem with such devices is that they can be hard to design. Currently they are often designed by hand, through analytically motivated guesswork combined with brute force parameter sweeps. However, this severely limits the designs that can be investigated. A parameter sweep of just 6 parameters with 10 different values for each requires 1 000 000 simulations. One can of course use smarter optimization algorithms like bayesian optimization or particle swarm optimization[1] to decrease the number of simulations needed, but it will still be of the same order.

cite something, check Ida's thesis maybe

A different approach that has been gaining some popularity is *inverse design with adjoint simulation.*[2] The idea is that if the gradient of the figure of merit with respect to the parameters can be calculated, then we can use gradient based optimization methods, which converge much faster, even if the number of parameters is very large. With these methods, one hopes to be able to search among a much more general class of designs for the optimal one. This has been successfully applied to nanophotonic devices, where a wide variety of components have been designed[3]. In some cases, for example the waveguide bend, the inverse-designed device could be made much more compact than conventionally designed bends. In other cases, for example the vertically-incident wavelength-demultiplexing grating coupler, there are no other known conventional methods of designing the device.

Systematic design of photonic crystal structures using topology optimization: Lowloss waveguide bends,

Quantum acoustic devices have in general been studied much less than photonic devices, and the library of known devices is very small. With this thesis, I explore the possibility of extending the inverse-design paradigm to quantum acoustic devices. Since both acoustics and electromagnetics are wave phenomena, there are many analogies to be drawn, but there are also important differences. I have in this work shown that the adjoint method is applicable to the case of acoustics, as well as derived the form of the equations. As a proof of concept, I attempt to design a phononic 50/50 beamsplitter. The beamsplitter is conceptually one of the simplest devices imaginable, and photonic beamsplitters have been studied in great detail for many years. However, there is still no standard implementation for phononic beamsplitters.

pigott thesis

1.1 Aim and Thesis outline

The aims of this thesis are:

- Rederive the equations for inverse design with adjoint simulation in the case of acoustics and confirm that the methods are theoretically applicable.
- Implement the methods and use them to design a phononic beamsplitter.

Chapter 2 presents some of the theory on solid mechanics and acoustic waves needed to understand the thesis.

2. Acoustic waves and waveguides

In this chapter, the theory of the physics behind my simulations is given. Section 2.1 gives a review of solid mechanics in the frequency domain, culminating in equation (2.13), which is the equation that is solved each simulation. Section 2.2 goes on to show that the solutions to this equation with no external forces for infinite periodic structures can be obtained from simulating a single unit cell of the structure. Lastly, the solutions for the specific periodic structure used for the inputs and outputs in this thesis is shown with a band diagram as well as the shapes of the modes.

2.1 The frequency domain acoustic equation

In order to efficiently model the deformation and stresses in a solid material, a linear elasticity model is often assumed. For small deformations, solid materials obey Hooke's law which in it's full form looks like

$$\sigma = C : \epsilon \tag{2.1}$$

where σ is the stress tensor, C the elasticity tensor which is a rank four tensor that is a property of the material,

$$\epsilon := \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^\intercal) \tag{2.2}$$

is the strain tensor, and : denotes double scalar product[1]. This equation is linear in $\boldsymbol{u}$, hence the name *linear* elasticity. Using this and Newton's equations of motion, the equation governing the dynamics is obtained:

$$\rho \ddot{\boldsymbol{u}} = \nabla \cdot \sigma + \boldsymbol{F}, \tag{2.3}$$

where ρ is the density, $\boldsymbol{u}$ is the displacement and $\boldsymbol{F}$ is the externally applied force. Assuming a time harmonic solution $\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{u}(x)e^{i\omega t}$ with angular frequency ω this becomes

$$-\rho\omega^2 \boldsymbol{u} = \nabla \cdot \sigma + \boldsymbol{F}. \tag{2.4}$$

To combine equations (2.1) to (2.4) into one equation that can be solved for $\boldsymbol{u}$ we

[1] See equation (2.6) for what the double scalar product means in this case. It is in general a product that contracts two indices, as opposed to the regular scalar product that contracts only one.

first rewrite them in index notation to make calculations clearer:

$$\epsilon_{ij} = \frac{1}{2}(\partial_i u_j + \partial_j u_i) \tag{2.5}$$
$$\sigma_{ij} = C_{ijkl}\epsilon_{kl} \tag{2.6}$$
$$= \frac{1}{2}\left(C_{ijkl}\partial_k u_l + C_{ijkl}\partial_l u_k\right) \tag{2.7}$$
$$= C_{ijkl}\partial_k u_l \text{ because of the symmetry } C_{ijkl} = C_{ijlk} \tag{2.8}$$

which gives

$$F_i = -\rho\omega^2 u_i - \partial_j \sigma_{ij} \tag{2.9}$$
$$= -\rho\omega^2 \delta_{ik} u_k - \partial_j \left(C_{ijkl}\partial_l u_k\right) \tag{2.10}$$
$$= -\left(\rho\omega^2 \delta_{ik}\,\cdot + \partial_j \left(C_{ijkl}\partial_l\,\cdot\right)\right) u_k \tag{2.11}$$

where the indices i, j, k, l go over the spatial dimensions x, y, z. Note that throughout this thesis, the Einstein summation convention is used, meaning that repeated indices are summed. All of the tensors in the equation above are really tensor fields, i.e. they are functions of $\boldsymbol{x}$. Defining the operator

$$\hat{A}_{ik} = -\left(\rho\omega^2 \delta_{ik}\,\cdot + \partial_j \left(C_{ijkl}\partial_l\,\cdot\right)\right) \tag{2.12}$$

we can write

$$\hat{A}_{ik} u_k = F_i \tag{2.13}$$

2.2 Bloch States and Band Diagrams

With no external forces, i.e. $F_i = 0$, equation (2.13) can be written as

$$\frac{1}{\rho}\partial_j \left(C_{ijkl}\partial_l u_k\right) = \omega^2 u_i, \tag{2.14}$$

which is an eigenvalue equation for the operator $\hat{O}_{ik} = \frac{1}{\rho}\partial_j \left(C_{ijkl}\partial_l\,\cdot\right)$ where eigenvalues are the angular frequency squared. If furthermore the structure is periodic, then the eigenstates are so called *Bloch states*.

If the structure is periodic in the y direction with some periodicity $\boldsymbol{a} = a\hat{\boldsymbol{y}}$, meaning that $C_{ijkl}(\boldsymbol{x}) = C_{ijkl}(\boldsymbol{x} + n\boldsymbol{a})$ and $\rho(\boldsymbol{x}) = \rho(\boldsymbol{x} + n\boldsymbol{a})$ where n is an integer, then this operator commutes with the translation operator $\hat{T}_{\boldsymbol{a}}[\boldsymbol{u}(\boldsymbol{x})] = \boldsymbol{u}(\boldsymbol{x} + \boldsymbol{a})$. This means that there is a basis of simultaneous eigenstates of both operators. The eigenfunctions of the translation operator are $\boldsymbol{f}(x, z)\exp(ik_y y)$, where $\boldsymbol{f}$ is an arbitrary function, and the eigenvalues are $\exp(ik_y a)$. Defining the reciprocal lattice constant $b = 2\pi/a$, we see that the functions $\boldsymbol{f}(x, z)\exp(i(k_y + mb)y)$ for integer values of m all have the same eigenvalue, which means that they form a degenerate subspace of eigenfunctions. This also means that we can restrict ourselves to the first Brillouin zone: $k_y \in [-b/2, b/2]$,

since any k_y outside this interval can be written as $k'_y + mb$ with $k'_y \in [-b/2, b/2]$. Thus, the simultaneous eigenstate $\boldsymbol{u}_{k_y,\omega^2}$ can be written

$$\boldsymbol{u}_{k_y,\omega^2}(\boldsymbol{x}) = \sum_m \boldsymbol{f}_{m,k_y,\omega^2}(x,z)e^{i(k_y+mb)y} = e^{ik_y y}\tilde{\boldsymbol{f}}_{k_y,\omega^2}(\boldsymbol{x}) \tag{2.15}$$

where $\tilde{\boldsymbol{f}}_{k_y,\omega^2}$ is a periodic function with periodicity $\boldsymbol{a}$ by construction.

Cite chapter 3 of Z:/general interest/literature/books/photonic-crystals-book.pdf

The solutions to these eigenvalue equations are often called modes, and each mode has both a frequency and a wave vector. This gives rise to a band diagram, where the frequency is plotted as a function of the wave vector.

2.2.1 Concrete example of phononic crystal

In this work, a rectangular waveguide patterned with ellipses is used. The structure is clamped at the bottom (meaning that we use a fixed boundary condition there, enforcing $\boldsymbol{u} = 0$) while the other sides are free. A top down schematic of the unit cell can be seen in figure 2.1.

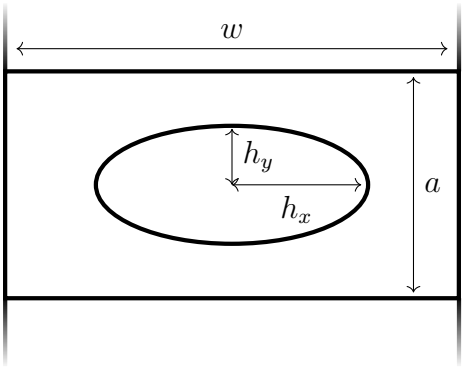


Figure 2.1: *Top down view of unit cell of a phononic crystal.*

An infinite waveguide can be simulated with just one unit cell using periodic boundary conditions at the edge where the unit cell would be attached to the next one. As section 2.2 showed, waves with any $\boldsymbol{k}$ can be investigated with a single unit cell since $\boldsymbol{u}$ is a phase factor $e^{i\boldsymbol{k}\cdot\boldsymbol{x}}$ times some function with the same periodicity as the unit cell. Enforcing a wave vector $\boldsymbol{k} = k\hat{\boldsymbol{y}}$ thus entails using periodic boundary conditions with a specified phase shift over the unit cell. These are called *floquet periodic boundary conditions* in COMSOL, which is the simulation software used throughout this thesis.

Don't I introduce comsol earlier?

Running simulations with different k to find the eigenmodes with their corresponding frequencies for this structure yields the band diagram in figure 2.2. The parameter

values used in the simulations can be found in table 4.1. Figure 2.3 shows the shapes of the eight lowest lying eigenmodes at $k = 0.9\pi/a$.

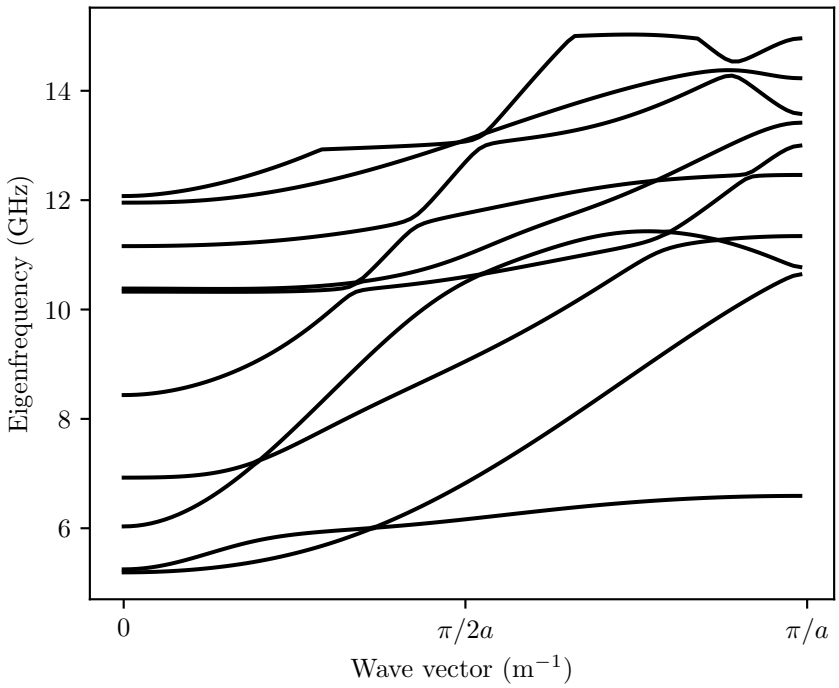


Figure 2.2: *Band diagram of phononic crystal defined in figure 2.1.*

At some point write about phonons? I haven't really had to care about the fact that excitations are discrete so if I talk about it it'd just for applications...

I originally thought to have something about PMLs here, but now it's been put in the method... It is more of a simulations topic, not really a physics one.

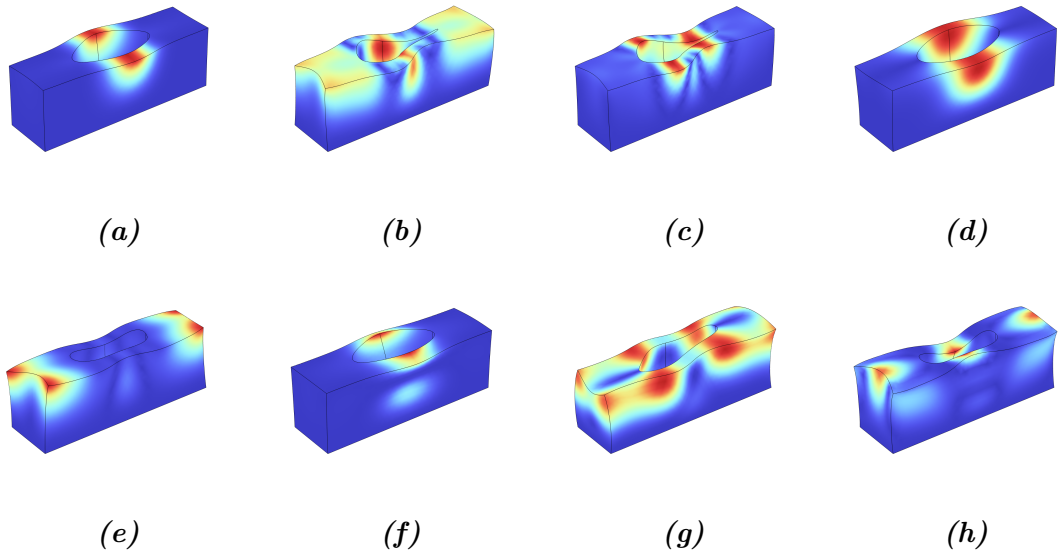


Figure 2.3: *Mode shapes for the lowest eight modes at $k = 0.9\pi/a$. The color denotes the absolute value of the displacement, and the scale is normalized for each figure.*

3. Inverse Design

Inverse design is a design paradigm where the design of a device is guided fully by the desired characteristics. These desired characteristics are quantified through what is called an objective function[2], which I will denote f_{obj}, that should be maximized. When coupled with *adjoint simulation*, which is a clever way to compute gradients, and gradient based optimization algorithms, this is a very powerful methodology.

An overview of the design process is as follows:

1. Initialize a random device design.
2. Calculate the gradient of the design through the adjoint method.
3. Update the device design using the gradient according to the optimization algorithm.
4. If the device performance is good enough, terminate optimization, else return to step 2.

3.1 Adjoint Simulation

Adjoint simulation is a way to compute the gradient of f_{obj} with respect to the design, which in our case means with respect to the material parameters. I will in this section first give a general derivation, following reference [4]. In section 3.1.2 I will then derive the specifics when applying this to acoustics.

3.1.1 General Derivation

Let f_{obj} be a function which depends on some high-dimensional vector v. The vector v can be calculated by solving the linear equation $Av = b$, where b is a fixed vector and A is a matrix that depends on a vector of design parameters p. This could be the acoustics equation, but it might also be the analogous equation for photonics, or something completely different like fluid dynamics. I will refer to the process of solving this equation as a simulation, since for this thesis it is solved through the simulation software COMSOL. The overall goal is to find the parameters p that maximize the objective function f_{obj}. The goal of adjoint simulation is to find $\frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}p}$.

[2]Also called *figure of merit (FoM)*.

This can be expanded through the chain rule as

$$\frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}p} = \frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}v}\frac{\mathrm{d}v}{\mathrm{d}p}. \tag{3.1}$$

To find the latter factor we do

$$\frac{\mathrm{d}}{\mathrm{d}p}[Av = b] \implies \frac{\mathrm{d}A}{\mathrm{d}p}v + A\frac{\mathrm{d}v}{\mathrm{d}p} = 0 \tag{3.2}$$

$$\implies A\frac{\mathrm{d}v}{\mathrm{d}p} = -\frac{\mathrm{d}A}{\mathrm{d}p}v. \tag{3.3}$$

Thus, if we can find a $\tilde{v}$ such that

$$\frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}v} = \tilde{v}A \tag{3.4}$$

then

$$\frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}p} = \tilde{v}A\frac{\mathrm{d}v}{\mathrm{d}p} \tag{3.5}$$

$$= -\tilde{v}\frac{\mathrm{d}A}{\mathrm{d}p}v. \tag{3.6}$$

Finding $\tilde{v}$ from equation (3.4) amounts to solving the so called *adjoint problem*:

$$A^\dagger\tilde{v}^\dagger = \frac{\mathrm{d}f_{\text{obj}}}{\mathrm{d}v}^\dagger \tag{3.7}$$

hence the name adjoint method. As it turns out, A is in many cases symmetric (or self-adjoint) which means that this is simply a normal simulation but with $\mathrm{d}f_{\text{obj}}/\mathrm{d}v^\dagger$ as the source. Thus, to obtain the derivative we just need to run an additional simulation with a different input.

Now you might be wondering: what have we gained by this? Let n be the dimension of v, m the dimension of p and l the dimension of b. This means that A is a matrix with dimension $l \times n$ and $\mathrm{d}A/\mathrm{d}p$ is a rank three tensor with dimension $m \times l \times n$. Thus solving for $\mathrm{d}v/\mathrm{d}p$ from equation equation (3.3) means solving for an $n \times m$ matrix, which much more computationally expensive than solving for just a vector of dimension n.

3.1.2 Specific derivation with acoustics

Now we turn to the specific case of acoustic devices. Here $Av = b$ is replaced by the acoustic field equation (equation (2.13)):

$$\hat{A}_{ik}u_k = F_i. \tag{3.8}$$

Instead of vectors, like we saw in section 3.1.1, these quantities are now functions[3] of $\boldsymbol{x}$. Analogously to the vector of design parameters we now have a *design field* $p(\boldsymbol{x})$, and analogously to f_{obj} being a function of a vector, this f_{obj} is a function of a function, i.e. a *functional*. For a quick overview of functionals and their derivatives, see box 3.1.

[3]Vector-valued funcitons, but that is not the important part here.

Box 3.1: On functionals and their derivatives

Big fat box on functionals and their derivatives. I think this should be included somewhere, since very few of my peers know what a functional derivative is... Not really sure how though. I kinda like the thought of putting it in a box like this. Alternatively, I could put it in an appendix.

Our f_{obj} is no longer a function, but rather a *functional*, and thus we need to use the functional derivative instead of the ordinary derivative. One can think of a functional as a function of a function, i.e. something that maps an element of a function space to a scalar number. There are also functionals which depend on both a function and a real number, or on multiple functions. Below I will give an overview of the notational conventions I use, and then give the definition of the functional derivative as well as some useful properties of it.

Let $\mathcal{Y}$ be a function space of functions $\mathbb{R} \to \mathbb{R}$. A functional $F : \mathcal{Y} \to \mathbb{R}$ evaluated at the function $f \in \mathcal{Y}$ is denoted with the function in square brackets: $F[f]$. Note that in principle, F is the functional while $F[f]$ is just a number, analogously to how f is a function while $f(x)$ is a real number. If the functional additionally depends on a real number, $G : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$, that is put in round brackets: $G[f](x)$.

The functional derivative of F with respect to its function argument is a functional $\mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ denoted $\delta F[f]/\delta f$. In this expression, f is technically a dummy function, writing $\delta F[g]/\delta g$ is exactly the same functional. However, often the argument of F is omitted and the function in the denominator is named in accordance with the names in the definition of the functional. Furthermore, the same notation is also often used to denote the functional derivative evaluated at a certain function. For example, if we define a functional taking two function arguments $F[f_1, f_2] = \int f_1(x) + f_2(x)\,\mathrm{d}x$, one can write

$$\frac{\delta F}{\delta f_2}(x) \qquad \text{meaning} \qquad \frac{\delta F[g_1, g_2]}{\delta g_2}(x) \tag{3.9}$$

$$\frac{\delta F}{\delta f_2}(x) \qquad \text{meaning} \qquad \frac{\delta F[g_1, g_2]}{\delta g_2}[f_1, f_2](x) \tag{3.10}$$

where in the latter case, f_1 and f_2 are specific functions defined previously.

The functional derivative is defined by

$$\int \frac{\delta F}{\delta f}(x)\varphi(x)\,\mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}\varepsilon} F[f + \varepsilon\varphi] \tag{3.11}$$

where F is a functional of f and φ is an arbitrary test function.

I find the definition somewhat difficult to comprehend and really didn't understand it until I sat down and did some examples for myself. Maybe talk about analogies to vector functions...

I will use two properties of the functional derivative:

- If F is the functional $F[f](y) = f(y)$, then $\delta F(y)/\delta f(x) = \delta(y - x)$.
- The chain rule: if F is a functional with one function argument, G is a functional with one function and one real argument, and H is the functional defined as $H[f] = F[G[f](y)]$, then

$$\frac{\delta H}{\delta f}(x) = \int \frac{\delta F}{\delta G[f]}(y) \frac{\mathrm{d}G(y)}{\mathrm{d}f}(x)\,\mathrm{d}y \tag{3.12}$$

For simplicity I will limit myself to the case where the objective function is an overlap integral of the displacement field $u_k(\boldsymbol{x})$ with some function $\varphi_k^*(\boldsymbol{x})$:

$$f_{\text{obj}}[\boldsymbol{u}] = \int_\Omega u_i(\boldsymbol{x})\varphi_i^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{3.13}$$

where Ω is the domain of $\boldsymbol{u}$. Such an integral is an inner product in the space of functions on Ω.

Analogously to the general derivation in section 3.1.1, the chain rule is used to expand $\delta f_{\text{obj}}/\delta p(\boldsymbol{x})$, see box 3.1 for the form of the chain rule for the functional derivative. However, because $\boldsymbol{u}$ is in general complex, I will split it into its real and imaginary components: $\boldsymbol{u} = \boldsymbol{v} + i\boldsymbol{w}$.

$$\frac{\delta f_{\text{obj}}}{\delta p}(\boldsymbol{x}) = \int_\Omega \mathrm{d}\boldsymbol{y} \frac{\delta f_{\text{obj}}}{\delta v_i}(\boldsymbol{y}) \frac{\delta v_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) + \frac{\delta f_{\text{obj}}}{\delta w_i}(\boldsymbol{y}) \frac{\delta w_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) \tag{3.14}$$

The first factor of each of the two terms is easy enough to calculate:

$$\begin{aligned}
\frac{\delta f_{\text{obj}}}{\delta v_i}(\boldsymbol{y}) &= \frac{\delta}{\delta v_i(\boldsymbol{y})} \int_\Omega u_j(\boldsymbol{x})\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} && (3.15)\\
&= \int_\Omega \frac{\delta}{\delta v_i(\boldsymbol{y})} u_j(\boldsymbol{x})\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} && (3.16)\\
&= \int_\Omega \delta(\boldsymbol{x} - \boldsymbol{y})\delta_{ij}\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} && (3.17)\\
&= \varphi_i^*(\boldsymbol{y}) && (3.18)
\end{aligned}$$

and

$$\frac{\delta f_{\text{obj}}}{\delta w_i}(\boldsymbol{y}) = \frac{\delta}{\delta w_i(\boldsymbol{y})} \int_\Omega u_j(\boldsymbol{x})\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{3.19}$$

$$= \int_\Omega \frac{\delta}{\delta w_i(\boldsymbol{y})} u_j(\boldsymbol{x})\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{3.20}$$

$$= \int_\Omega i\delta(\boldsymbol{x}-\boldsymbol{y})\delta_{ij}\varphi_j^*(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{3.21}$$

$$= i\varphi_i^*(\boldsymbol{y}) \tag{3.22}$$

which gives us

$$\frac{\delta f_{\text{obj}}}{\delta p}(\boldsymbol{x}) = \int_\Omega \mathrm{d}\boldsymbol{y}\varphi_i^*(\boldsymbol{y})\frac{\delta v_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) + i\varphi_i^*(\boldsymbol{y})\frac{\delta w_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) \tag{3.23}$$

$$= \int_\Omega \mathrm{d}\boldsymbol{y}\varphi_i^*(\boldsymbol{y})\mathrm{Re}\left(\frac{\delta u_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x})\right) + i\varphi_i^*(\boldsymbol{y})\mathrm{Im}\left(\frac{\delta u_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x})\right) \tag{3.24}$$

$$= \int_\Omega \mathrm{d}\boldsymbol{y}\varphi_i^*(\boldsymbol{y})\frac{\delta u_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) \tag{3.25}$$

To find $\delta u_i(\boldsymbol{y})/\delta p(\boldsymbol{x})$ we apply $\delta/\delta p(\boldsymbol{x})$ to equation (3.8), which gives us

$$0 = \frac{\delta \hat{A}_{ik}}{\delta p}(\boldsymbol{x})u_k(\boldsymbol{y}) + \hat{A}_{ik}\frac{\delta u_k(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) \tag{3.26}$$

rewrite this

The point of inverse design is that we now want to find an adjoint field $\tilde{u}_i(\boldsymbol{y})$ such that the integral in equation (3.25) is

$$\int_\Omega \mathrm{d}\boldsymbol{y}\,\varphi_i^*(\boldsymbol{y})\frac{\delta u_i(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) = \int_\Omega \mathrm{d}\boldsymbol{y}\,\tilde{u}_i(\boldsymbol{y})\hat{A}_{ik}\frac{\delta u_k(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) \tag{3.27}$$

which by equation equation (3.26) is equal to

$$-\int_\Omega \mathrm{d}\boldsymbol{y}\,\tilde{u}_i(\boldsymbol{y})\frac{\delta \hat{A}_{ik}}{\delta p}(\boldsymbol{x})u_k(\boldsymbol{y}). \tag{3.28}$$

To further simplify this expression, the dependence of $\hat{A}$ on p must be specified. A linear dependence is proposed here, though extending the formula to more complicated dependences is rather easy. Taking $\rho(\boldsymbol{y}) = \rho^0 p(\boldsymbol{y})$ and $C_{ijkl}(\boldsymbol{y}) = C^0_{ijkl}p(\boldsymbol{y})$ in the definition of $\hat{A}_{ik}$ from equation (2.12) implies

$$\frac{\delta \hat{A}_{ik}(\boldsymbol{y})}{\delta p}(\boldsymbol{x}) = -\rho^0\omega^2\delta_{ik}\delta(\boldsymbol{x}-\boldsymbol{y}) - \partial_j\left(C^0_{ijkl}\delta(\boldsymbol{x}-\boldsymbol{y})\partial_l\,\cdot\right) \tag{3.29}$$

$$= -\rho^0\omega^2\delta_{ik}\delta(\boldsymbol{x}-\boldsymbol{y}) - 2C^0_{ijkl}\delta(\boldsymbol{x}-\boldsymbol{y})\partial_j\partial_l. \tag{3.30}$$

Plugging this back into the integral in equation (3.28) gives

What? Minus sign???

$$\omega^2 \rho^0 \tilde{u}_i(\boldsymbol{x}) u_i(\boldsymbol{x}) + 2C^0_{ijkl} \tilde{u}_i(\boldsymbol{x}) \partial_j \partial_l u_k(\boldsymbol{x}) \tag{3.31}$$

which is comparatively easily evaluated. The only thing remaining is to calculate $\tilde{\boldsymbol{u}}$. The way to do this is through an *adjoint simulation*.

When solving these equations in practice, space is discretized and the fields are represented by vectors and the operators by matrices. Thus equation (3.27) becomes

$$\varphi^*_a \eta_{ab} = \tilde{u}_c A_{ca} \eta_{ab} \tag{3.32}$$

where $\delta \boldsymbol{u}/\delta p$ has been labeled η for notational simplicity, and the indices go over the discretized spatial meshpoints. Meaning that to find $\tilde{u}$ we solve

$$\varphi^*_a = \tilde{u}_c A_{ca} \tag{3.33}$$

or, taking the transpose of both sides

$$A^\intercal_{ac} \tilde{u}_c = \varphi^*_a. \tag{3.34}$$

Comment from PB: it might be nice to clarify at which point one can see that the adjoint simulation would result in the solution u tilde you could consider writing a short summary at the end of the chapter where you explain step by step how inverse design is practically done. E.g.: 1. We are solving the system with the regular acoustic field equation for a certain p and receive u 2. we are interested in df/dp so that we know how to change p in order to maximize f 3. df/dp was calculated in eqn X. I think this would be instructive for the reader :)

I think that the last part of this, on the adjoint simulation, is in need of a rewrite. For one thing, before I talk about the discretization, there is no adjoint in the formula for the adjoint field. Or there is no dagger at least...

3.2 Optimization Algorithms

rephrase first sentence

In the last section I painstakingly derived how one can obtain the gradient, and in this section I will attempt to justify that by describing how one can use the gradient. I will begin by describing the advantages of gradient based optimization algorithms over those that don't use the gradient. Following that I describe the algorithm that I used, as well as some of it's predecessors.

An optimization algorithm is an algorithm for finding the optimum of a function. The function is often called the *objective function* or the *cost function*. A very naive

optimization method would be to simply try some number of inputs and then choose the one with the highest function value. This requires a large number of points before a good value is found, meaning that it takes a long time. An improvement to this method is to use the information gained from the points already tried to decide which points to try next. If some point has a bad value, then try somewhere else; if some point has a good value, try another close by. Examples of algorithms that do this are bayesian optimization, particle swarm optimization, However, if the domain of the objective function is very high-dimensional, "close by" is a very large space. For such functions, it is essential to know in which direction the function increases. That is why gradient based optimization algorithms are so powerful; they enable us to quickly find the right direction to go in for best improvement.

more examples

3.2.1 Gradient Descent

The simplest gradient based optimization algorithm is called *gradient descent*. Like all of the algorithms I will describe it is an iterative algorithm, meaning that it generates a sequence of points that converges to an optimum, and the next point in the sequence is derived from the previous ones. In the case of gradient descent, the next point is gotten by

$$p_n = p_{n-1} + \eta g_{n-1} \tag{3.35}$$

where η is the so called *learning rate* and g_{n-1} is the gradient of the objective function at p_{n-1}. For ordinary gradient descent the learning rate would be fixed, and choosing an appropriate value for this parameter is one of the problems of this method. If a too high value is chosen, then the steps taken will be too large and the optimium might be missed entirely. A too low value results in too small steps which will yield a slow convergence. Almost all gradient descent implementations use a so called learning schedule, which means that η is not constant during the optimization. This introduces the additional problem of how to choose how fast and between which values it should change.

3.2.2 Adaptive Moment Estimation (ADAM)

The ADAM algorithm is an improved version of gradient descent. It has three main differences:

1. Each dimension has a separate learning rate.

2. The learning rate is automatically set from the previously seen gradients.

3. The evolution carries some momentum.

Figure 3.1 shows the difference in performance between ADAM and GD. With a slightly too large learning rate, the GD algorithm gets stuck in an oscillation and makes very little progress towards the minimum. If the learning rate is decreased, the oscillations disappear but the stepsize is now too small to make it all the way to the true minimum. Since the ADAM algorithm has some momentum, the motion

along the valley gets compounded while the motion perpendicular gets dampened which means that the oscillations are not as much of a problem. The adaptive learning rate also means that the algorithm doesn't get stuck prematurely due to the small gradient at the bottom of the valley. Admittedly, this objective function is specifically chosen to showcase the advantages of ADAM, but it has been shown to outperform GD in almost all cases.

citations

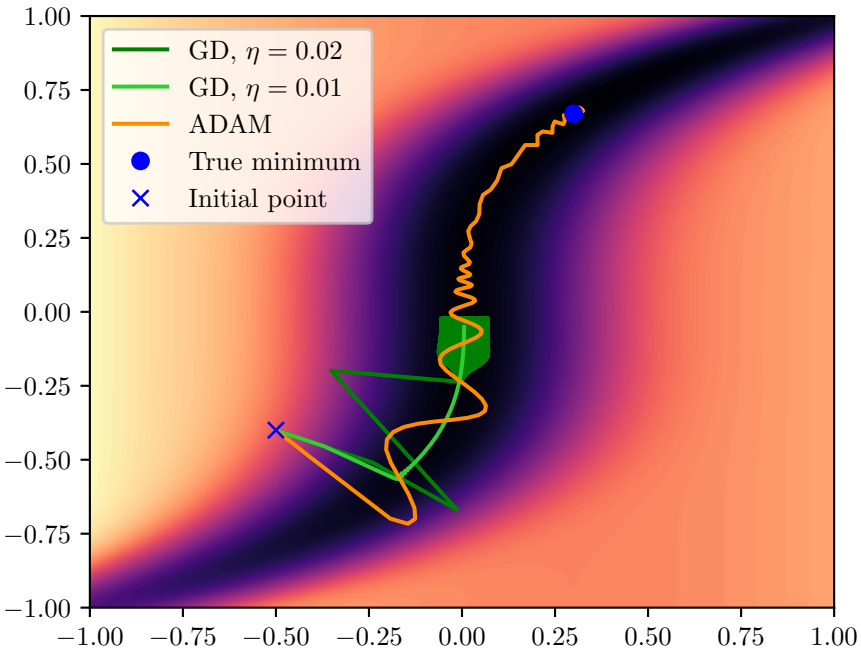


Figure 3.1: *A comparison between ADAM and GD in an optimization landscape with a narrow canyon. The two different GD algorithms are shown with 1000 steps, while 200 steps with the ADAM algorithm are shown.*

Pseudocode for the algorithm

Somewhere I should write about epsilon, and alpha and how I set them, but that probably needs to come later, in the method

4. Methods

The aim of this thesis is to use inverse design to find a phononic beamsplitter, a task that can be divided into three parts: First, some definitions of what should be designed and what constitutes a "good" design needs to be made. Second, we need a way to calculate the gradient of the "goodness" with respect to the design. And lastly, we need a gradient based optimization algorithm to find the optimal design. All of this will be described in this chapter.

Comment from PB: at some point you should explain what a beam splitter is, explain why it is so popular in photonics and how it could be interesting in phononics. also you should refer to previous work on phononic beam splitters, their plattform and basic differences. Doesnt have to be too detailed but some context is required.

4.1 Design

The device design to be optimized can be seen in figure 4.1. The input and output waveguides consists of unit cells like the one in figure 2.1. The values for the parameters in the sketch are given in table 4.1. The reason for using this mode in this waveguide is that it has been shown to be interesting for avoiding mechanical leakage into the substrate on which it is clamped, as well as retaining a high optomechanical coupling.[5]

Inside the design area, there can be one of two kinds of designs. The first is a *continuous design*, meaning that the material parameters ρ and C_{ijkl} are continuously varying. The range of values that they can take are between the density and elasticity of pure silicon and that of air. Any in-between values are obviously not something that can be physically realized, but it is useful as a first step in the optimization. This is parametrized through the *design field*, p, which takes values between 0, which means pure air, and 1, which means pure silicon. The second kind is a binary design, where each point either has silicon or not and there are no in-between values. This is accomplished using level-set methods, which will be explained in section 4.1.4.

Because the device is completely symmetric, only one half of it needs to be modeled, and the other half is extrapolated with a symmetry boundary condition.

Are the dx and dy in figure 4.1 clear? I thought of having arrows but it feels like the image gets quite messy then

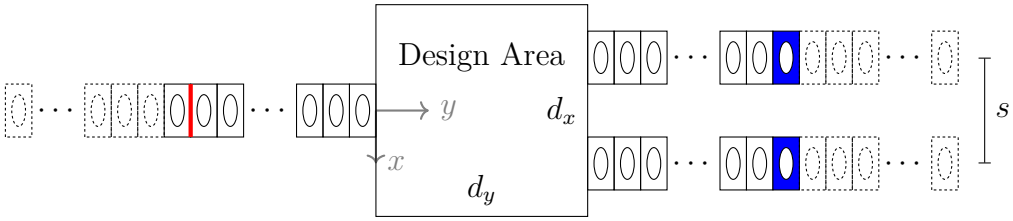


Figure 4.1: *Device design to be optimized. At the red line, a wave traveling right is excited. The output is measured over the blue unit cells. The dashed unit cells are PMLs. The large, rectangular design area has dimensions* $d_x \times d_y \times h$.

Table 4.1: *Values for the geometric parameters of the device. Reference figures 2.1 and 4.1 for what the quantities mean.*

Parameter	value
a	187 nm
w	187 nm
h_x	153.5 nm
h_y	49.5 nm
h	220 nm
d_x	$6w$
d_y	$4w$
s	$3w$

Write about why we use the mode we use, and why I clamp the bottom. Can reference Johan and Pauls paper

4.1.1 Objective function

The figure of merit of the device is how much of the input excitation gets transmitted into the output beams. Furthermore, all of the excitation of the output waveguide should be in the same mode that was excited at the input. Therefore, a mode overlap integral is used:

$$I = \int_{\Omega_1} \boldsymbol{m}^*(\boldsymbol{x})\boldsymbol{u}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \tag{4.1}$$

where $\boldsymbol{m}$ is the shape of the mode (figure 2.3a). Because we are not interested in the phase of the output waves, the absolute value squared of the overlap integral is taken as the objective function,

$$f_{\text{obj}} = |I|^2 = II^*. \tag{4.2}$$

This will be maximized when the excitation of the mode m in the output waveguide is maximized, regardless of which phase it has.

The functional derivative of f_{obj} with respect to p then becomes

$$\frac{\delta f_{\text{obj}}}{\delta p}(x) = \frac{\delta I}{\delta p}(x)I^* + I\frac{\delta I^*}{\delta p}(x) \tag{4.3}$$

$$= \frac{\delta I}{\delta p}(x)I^* + \left(I^*\frac{\delta I}{\delta p}(x)\right)^* \tag{4.4}$$

$$= 2\mathrm{Re}\left(\frac{\delta I}{\delta p}(x)I^*\right). \tag{4.5}$$

The derivative of I was derived in section 3.1.2.

Paragraph about pure part of objective function, enforcing the minimum feature size

4.1.2 Excitation

In order to excite the input waveguide in the desired mode, the stress on the boundary of a unit cell was exported from a unit cell eigenmode simulation with $k = 0.9\pi/a$ and applied to the boundary marked in red in figure 4.1. Since the frequency is perfectly controlled, this should excite only the desired mode, since that is the only permitted mode close by as seen in the band diagram in figure 2.2.

In order to confirm that the excitation was indeed fully in the desired mode, a separate model with only a waveguide with 200 unitcells was created. After applying the excitation and running the simulation, the proportion of the excitation that ended up in the desired mode was calculated. This was done by first calculating the mode

overlap integral $\int uu_m^* \, \mathrm{d}\boldsymbol{x}$ and comparing that to the norm of the displacement field $\int uu^* \, \mathrm{d}\boldsymbol{x}$. If $u = au_m + bu_r$ for some scalars a and b, then $\int uu^* \, \mathrm{d}\boldsymbol{x} = a \int uu_m^* \, \mathrm{d}\boldsymbol{x} + b \int uu_r^* \, \mathrm{d}\boldsymbol{x}$. And assuming u_r is orthogonal to u_m, $a = \int uu_m^* \, \mathrm{d}\boldsymbol{x} / \int u_m u_m^* \, \mathrm{d}\boldsymbol{x}$. The result was near perfect ($b < 0.01a$) excitation of only the desired mode. To obtain such high fidelity, it was important that the mesh used for the unitcells in the wave guide was the same as the mesh in the unit cell simulation. High fidelity was also achieved if both meshes were made very fine, but such fine meshes carries a prohibitively large computational cost.

4.1.3 Perfectly Matched Layers (PMLs)

Humour?

Ideally, the input and outputs are infinite waveguides. Unfortunately, it has been discovered that infinity is big; simulating infinite waveguides would take infinite time, and the author would like to be done by June. Instead, PMLs are placed at the caps of the input and output waveguides. The purpose of the PML is to absorb any incoming waves without reflection, which would make it act as if there was an infinite waveguide on the other side into which the waves propagate indefinitely. The way to accomplish this is to add an imaginary component to the density of the material. This needs to be done smoothly, otherwise the abrupt change in material parameters would induce reflections anyway. Therefore, the imaginary part is taken to be exponentially increasing. Furthermore, the curve is shifted vertically such that it is 0 at $y = y_0 - n$, and rescaled so that it is $\rho_{\text{si}} s$ at $y = y_0$.

$$\rho_{\text{im}} = \rho_{\text{si}} \cdot s \cdot \frac{e^{-|y-y_0|/d} - e^{-n/d}}{1 - e^{-n/d}} \tag{4.6}$$

Figure 4.2 shows the effect of changing these parameters on the shape of the profile of the imaginary component.

Explain why adding an imag part absorbs waves

There are three possible sources of reflections. Firstly, if the transition from no imaginary component to some imaginary component is too abrupt, that causes reflections. Secondly, if the imaginary component is too small, the waves will not be dampened completely when they reach the end of the PML and thus reflect off of that. And lastly, if d is small then there can be reflections from the steep increase that happens some distance away from the beginning of the PML. See figure 2.2 for an illustration of where the different types of reflections occur.

It is desirable to make n as small as possible while still eliminating all reflections. In order to do so, a long waveguide with the same parameters as used for the input and output waveguides in the beamsplitter design was created. To discern where there was some component of the wave reflected, a fourier transform of the displacement field was made. Figure 4.4 shows the amplitude of the reflection, quantified as the

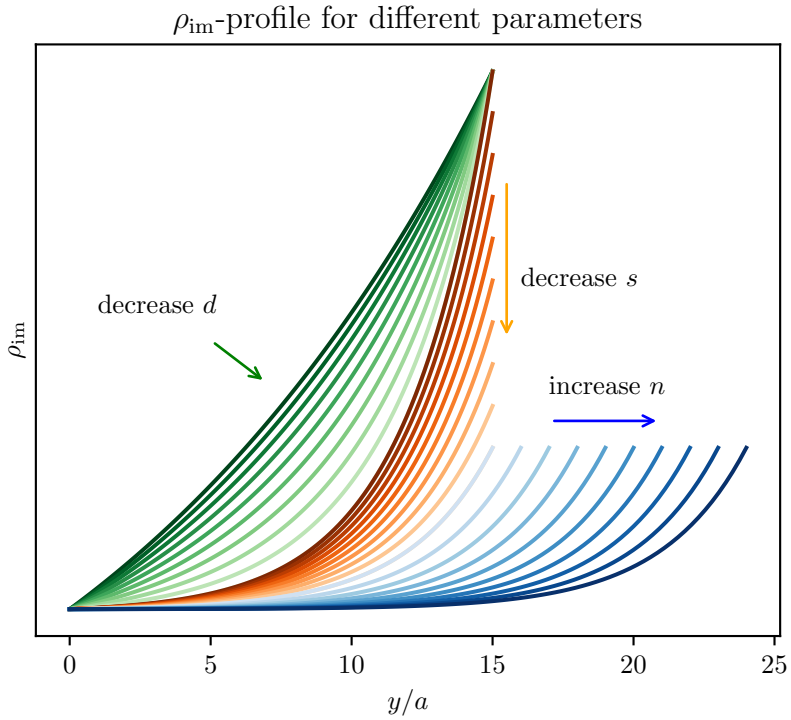


Figure 4.2: *This figure shows the effect of changing different parameters. The green curves shows changing d while keeping the other parameters fixed, and the orange and blue show s and n respectively. Darker colour means higher value, and the last green curve coincides with the first orange, and the last orange with the first blue.*

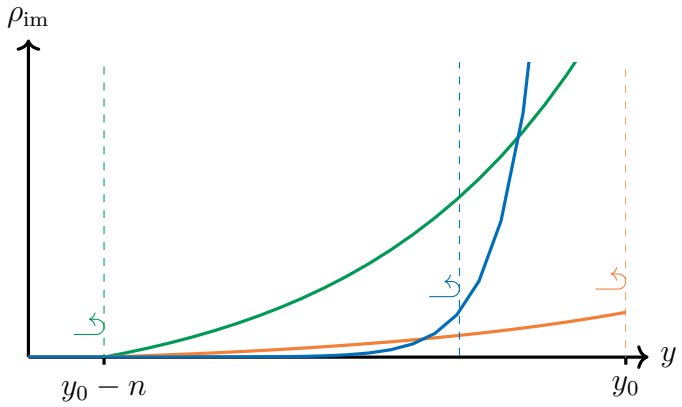


Figure 4.3: *For the green curve, the initial sudden increase of the imaginary component of the density at the beginning of the PML causes reflections. For the blue curve, the beginning of the PML is smooth but there is an increase partway through sharp enough to cause reflections. For the orange curve, the PML never becomes strong enough to completely dampen the waves, and they get reflected at the end.*

peak height relative to the forward propagating wave, for different profiles. These figures fit well with the three types of reflection mentioned previously. To achieve a short yet functional PML, $d = 5$, $s = 0.03$ and $n = 20$ was chosen.

4.1.4 Level-set

Ultimately, we want our device to consist of regions of material and regions of no material. There are basically two ways of doing this. The first, and perhaps most intuitive, is to simply store the coordinates of the boundary between the filled and empty regions. In addition to storing the coordinates, one must also store which points neighbour which. The second method, which is the one used in this report, is called the *level-set method*. In this method, the boundary is not directly stored, but rather is stored via an *implicit function*, $\phi(x)$, defined such that the boundary is the 0-isocontour of ϕ, i.e. the points x where $\phi(x) = 0$.

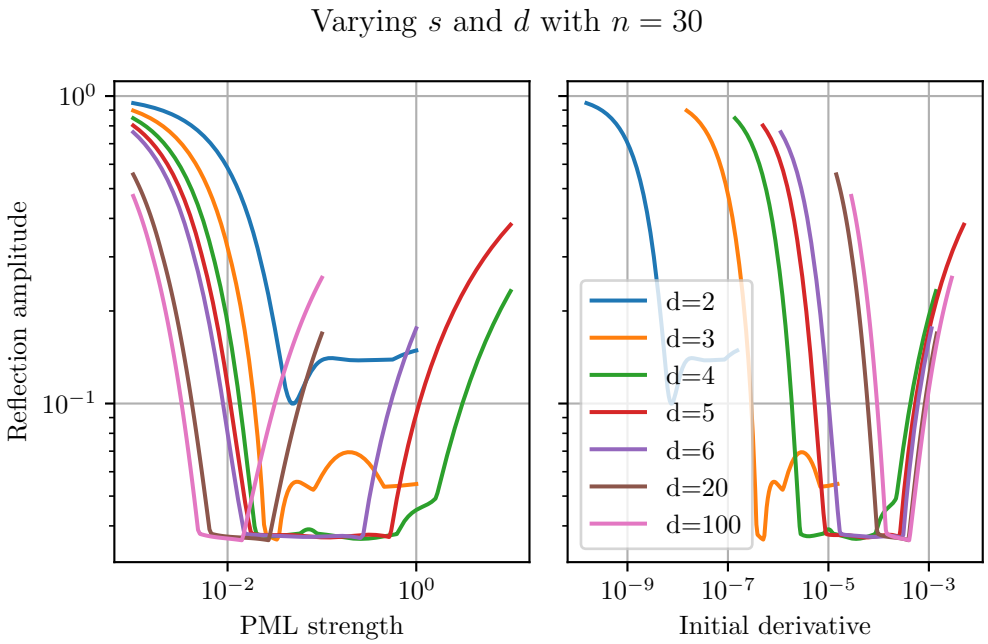
There are two main advantages of using the level-set method rather than directly storing the boundary points. Firstly, when moving the boundary we would like to do so in the normal direction, as moving it along itself has no effect. Computing the normal direction of a directly stored boundary is slightly cumbersome, though certainly achievable. With level-set, moving the boundary in the normal direction is as easy as adding a constant to the implicit function. Secondly, while the boundary is changing, the resolution in one part might need to be increased while the resolution in another needs to be decreased. Deciding where and when to add new points is non-trivial when directly storing the boundary. Furthermore, if two boundaries merge, or if one splits in two, points need to be removed and the connectivities changed, which is quite complex. Figure 4.5 illustrates these problems with direct storage concretely. Both of these issues are automatically handled with the level-set method.

How? (Isn't it obvious?)

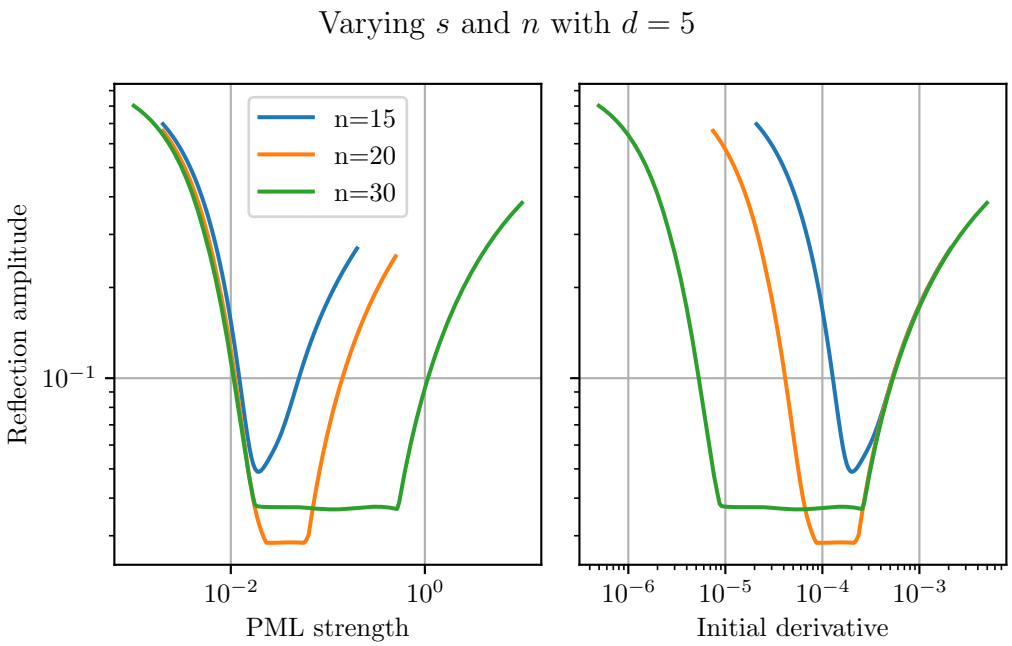
There are of course a lot of possible functions $\phi(x)$ that have a given boundary as it's 0-isocontour. There is one choice that simplifies a lot of calculations though: the signed distance function. This function is defined as the distance from the closest point on the boundary, with a plus sign if it is inside and a minus sign if it is outside the boundary. See figure 4.6 for an example. It has the advantage that if one wishes to locally shift the boundary some length s in the normal direction, then simply add s to the function there. Figure 4.7 shows this effect in one dimension.

Create another figure that shows it in two dimensions. I'm thinking a circular boundary, and adding s in the left half and subtracting s in the right half. Alternatively adding $s \cdot x$ (unit circle centered on 0) so that it will be smooth

Using a signed distance function means that a gradient descent step can be taken by simply adding the gradient field to the signed distance field. However, there are



(a) *On the left is the reflection amplitude plotted as a function of the PML strength s for different d. For $d > 4$ there are two sources of reflection: for small s reflection at the end of the PML occurs, and for large s, there is reflection at the beginning. The right figure makes it clear that it is the slope at the beginning of the PML that matters, since the point at which it becomes significant is the same for all d.*



(b) *This is the same as figure 4.4a but with varying n. For $n = 15$, the reflections from the beginning do not subside before the reflections from the end become significant, so at least $n = 20$ is necessary.*

Figure 4.4

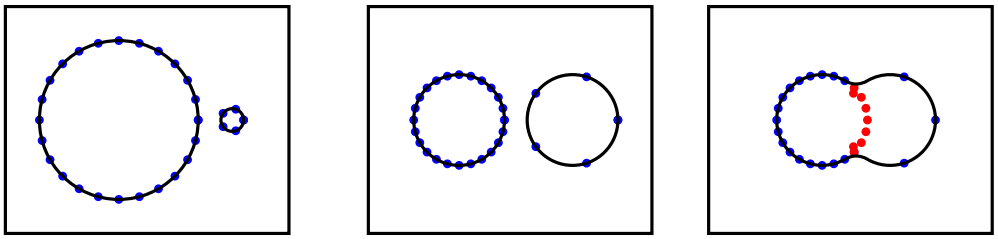


Figure 4.5: *Possible evolution of boundary. In the leftmost figure, the boundary is defined by pretty much evenly spaced points. In the center figure the boundaries have moved and the spacing is no longer even, and the right circle is very poorly resolved. The rightmost figure shows the boundary after the two circles moved closer together. Now there are multiple points that need to be removed, marked in red, and the connectivity of the points that remain must be changed such that the two boundaries are merged.*

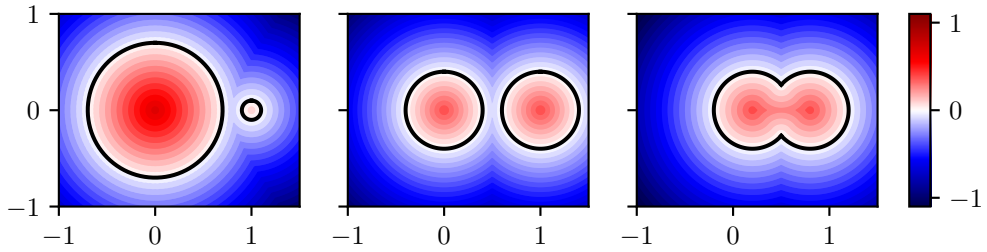


Figure 4.6: *Example of three signed distance functions for three different boundaries.*

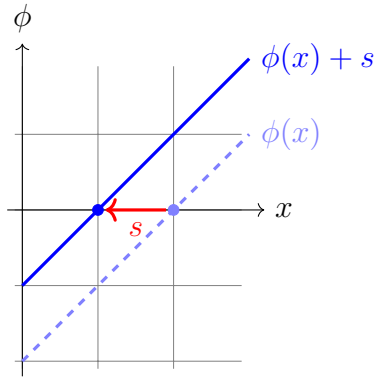


Figure 4.7: *Adding s to the signed distance function shifts boundary by s.*

some pitfalls that must be avoided. Firstly, the gradient needs to be rescaled so that the boundary moves an appropriate distance. This has been done such that the boundary moves maximally 5 nm. Secondly, since the gradient is occasionally sharply peaked somewhere which may not lie near the boundary, only the gradient near the boundary is actually added to the signed distance field. After performing this addition, what was previously a signed distance field will now no longer be that, and thus the signed distance field is recalculated from the new boundary. This recalculation comes with a performance penalty, but since the COMSOL simulations are orders of magnitude slower than all other parts of the optimization, this is of little concern.

check this number before finalizing, I change it every now and then

Meshing of the level-set designs?

4.2 Simulations

How much specifics should I have? Should I write about exporting the gradient to a file and how I calculate the 2D gradient in my matlab scripts from that

Mesh export / import and why that is done should probably be mentioned since it's quite important to get the excitation in the right mode.

4.3 Optimization

Describe what optimization algorithm was used, as well as how this changed during the simulation. E.g. first 200 iterations ADAM; next ADAM but with sigmoid function application; sigmoid + feature size; and finally level-set.

5. Results

The continuous optimization yielded devices with near perfect performance. However, the optimization never converged. After reaching a high value, the algorithm invariably descended to very low levels. This failure to converge prevented a gradual introduction of penalty terms for in-between values, which would have allowed for a smooth transition to the level-set design paradigm. Nevertheless, the level-set methods were also tried, though since the best devices of the continuous optimization were very far from binary, manual designs were used as a starting point. The level-set optimization fared worse in some regards, better in others. The algorithm did converge after a couple hundred iterations. However, the final value to which it converged was only 25% of a perfect performance.

Paragraph summarizing the results: best figure of merit, no of iterations and maybe time for simulations.

More specifics... what should I even put here, and how should I structure it?

A word on errors and why they happen. E.g. the density -> 0 numerically unstable thing

5.1 Continuous optimization

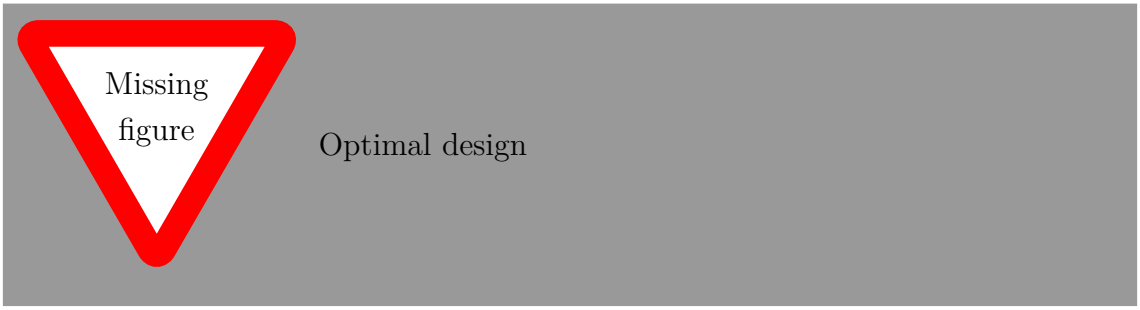


Figure 5.1

5.2 Level-set optimization

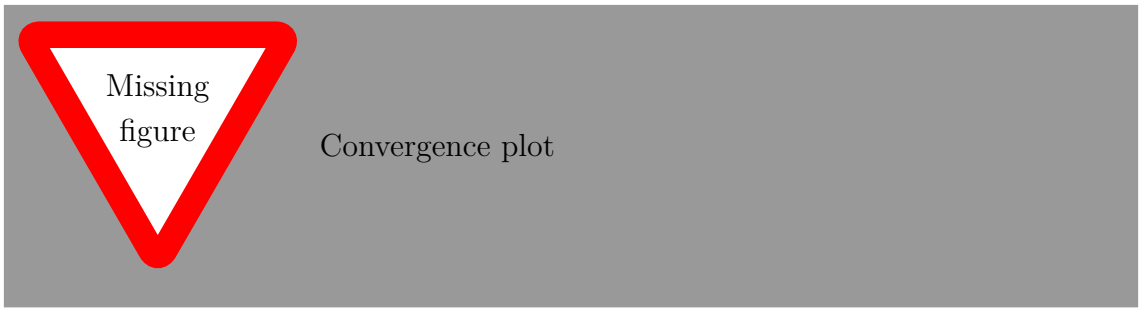


Figure 5.2

Figure 5.3

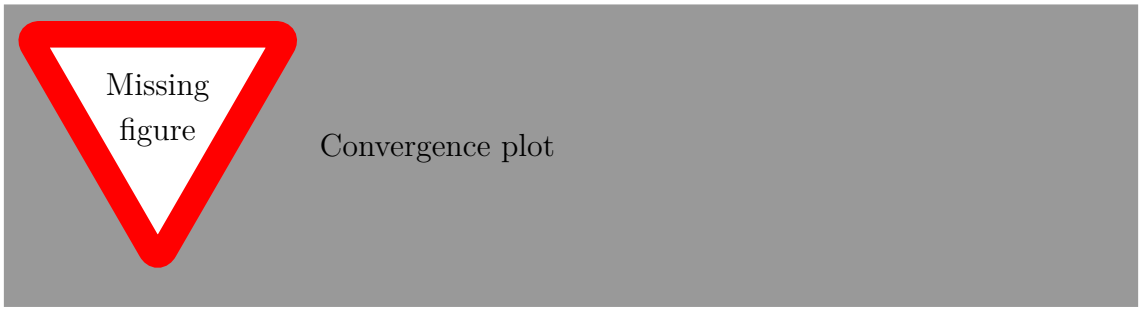


Figure 5.4

6. Conclusion

References

[1] Y. Zhang, S. Yang, A. E.-J. Lim, *et al.*, "A compact and low loss y-junction for submicron silicon waveguide," *Optics Express*, vol. 21, no. 1, pp. 1310–1316, Jan. 14, 2013, Publisher: Optica Publishing Group. DOI: 10.1364/OE.21.001310. [Online]. Available: https://opg.optica.org/oe/abstract.cfm?uri=oe-21-1-1310 (visited on 2023-04-19).

[2] S. Molesky, Z. Lin, A. Y. Piggott, W. Jin, J. Vucković, and A. W. Rodriguez, "Inverse design in nanophotonics," *Nature Photonics*, vol. 12, no. 11, pp. 659–670, Nov. 2018, Number: 11 Publisher: Nature Publishing Group. DOI: 10.1038/s41566-018-0246-9. [Online]. Available: https://www.nature.com/articles/s41566-018-0246-9 (visited on 2023-04-14).

[3] L. Su, D. Vercruysse, J. Skarda, N. V. Sapra, J. A. Petykiewicz, and J. Vuckovic, "Nanophotonic inverse design with spins: Software architecture and practical considerations," 2019. arXiv: 1910.04829 [physics.app-ph].

[4] M. B. Giles and N. A. Pierce, "An introduction to the adjoint approach to design," *Flow, Turbulence and Combustion*, vol. 65, no. 3, pp. 393–415, Dec. 1, 2000. DOI: 10.1023/A:1011430410075. [Online]. Available: https://doi.org/10.1023/A:1011430410075 (visited on 2023-05-27).

[5] J. Kolvik, P. Burger, J. Frey, and R. Van Laer, *Clamped and sideband-resolved silicon optomechanical crystals*, Mar. 31, 2023. DOI: 10.48550/arXiv.2303.18091. arXiv: 2303.18091[physics,physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2303.18091 (visited on 2023-04-14).

A. First appendix

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

B. Second appendix

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.