EFFICIENT WAVE-BASED SOUND PROPAGATION AND OPTIMIZATION FOR COMPUTER-AIDED DESIGN

Nicolas M. Morales

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2018

Approved by:

Dinesh Manocha

Ming Lin

Gary Bishop

Yun Jing

Jan Prins

© 2018
Nicolas M. Morales
ALL RIGHTS RESERVED

ABSTRACT

Nicolas M. Morales: Efficient Wave-based Sound Propagation and Optimization for Computer-Aided Design
(Under the direction of Dinesh Manocha)

Acoustic phenomena have a large impact on our everyday lives, from influencing our enjoyment of music in a concert hall, to affecting our concentration at school or work, to potentially negatively impacting our health through deafening noises. The sound that reaches our ears is absorbed, reflected, and filtered by the shape, topology, and materials present in the environment. However, many computer simulation techniques for solving these sound propagation problems are either computationally expensive or inaccurate. Additionally, the costs of some methods are dramatically increased in design optimization processes in which several iterations of sound propagation evaluation are necessary.

The primary goal of this dissertation is to present techniques for efficiently solving the sound propagation problem and related optimization problems for computer-aided design. First, we propose a parallel method for solving large acoustic propagation problems, scalable to tens of thousands of cores. Second, we present two novel techniques for optimizing certain acoustic characteristics such as reverberation time or sound clarity using wave-based sound propagation. Finally, we show how hybrid sound propagation algorithms can be used to improve the performance of acoustic optimization problems and present two algorithms for noise minimization and speech intelligibility improvement that use this hybrid approach.

All the algorithms we present are evaluated on various benchmarks that are computer models of architectural scenes. These benchmarks include challenging environments for existing sound propagation algorithms, such as large indoor or outdoor scenes, structural complex scenes, or the prevalence of difficult-to-model sound propagation phenomena. Using the techniques put forth in this dissertation, we can solve many challenging sound propagation and optimization problems on the scenes in an efficient manner. We are able to accurately model sound propagation using wave-based approaches up to 10 kHz (the full range of human speech) and for the full range of human hearing (22 kHz) using our hybrid approach. Our noise minimization methods show improvements of up to 13dB in noise reduction on some scenes, and we show a 71% improvement in speech intelligibility using our algorithm.

ACKNOWLEDGMENTS

The research in this dissertation could not have been done without the help and support of others. First, I would like to thank my coauthors on the various papers I present. These include Ravish Mehra, Vivek Chavda, Zhenyu Tang, and of course my advisor, Dinesh Manocha. On many of these papers, I benefited from the knowledge and experience of Abhinav Golas, Alok Meshram, Atul Rungta, Allan Porterfield, Carl Schissler, and Auston Sterling, all who had important suggestions and critiques.

Aside from these technical contributions, I never would have made it through grad school without my friends and family. My parents have always encouraged me to do my best, and they were always supportive during the highs and lows of my graduate career. I would also like to thank all my friends. Their support during my work on this dissertation was essential.

Finally, I would like to thank my committee: Profs. Dinesh Manocha, Ming Lin, Gary Bishop, Yun Jing, and Jan Prins. They provided many helpful critiques and suggestions to improve my dissertation. My advisor, Dinesh, always pushed me to publish and submit to quality journals and conferences. Without him, I would not have had the success that I had with my publications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Sound is ubiquitous in our lives and has a widely-varying effect on audibility, health, comfort, learning, and enjoyment. Therefore, the architectural design of theaters, hospitals, workplaces, schools, and concert halls take into account sound propagation, or the manner in which sound travels throughout an environment. For example, hospitals must be designed in a way to minimize the impact of environmental noise, since this noise can have a negative effect on health and recovery (Basner et al., 2014). The World Health Organization publishes guidelines for acceptable environmental noise levels (Birgitta et al., 1999) for hospitals.

Similarly, the quality of concert halls can be described by various acoustic metrics. These metrics measure the impact of the propagation environment on the sound that the audience hears. Auditoriums and lecture halls have similar requirements — it is important that audience members are able to hear the speaker clearly and understand them.

Furthermore, these design considerations are often incorporated early in the architectural design process. Changing the way in which sound propagates in an environment after it is built is difficult, and can sometimes amplify fundamental acoustic problems in the design (Egan, 1988). As such, many architects use mathematical or computer models to analyze the effects of sound propagation in their designs.

1.1 Models for Sound Propagation

Sound travels as a wave through an elastic medium, such as air, water, and solid objects such as materials commonly used in construction. Sound waves are longitudinal, and driven by the compression and rarefaction of the medium. In air, these waves change atmospheric pressure as they propagate: areas of compression contain the highest pressure, while areas of rarefaction reduce the pressure below normal atmospheric conditions. As such, sound from a simple pure tone travels through air in a sinusoidal manner, with the period being inversely proportional to the frequency of the sound source.

The speed at which sound travels through a fluid medium depends on compressibility and density. In dry air, at 20 °C, sound travels at 343 $\mathrm{m\,s^{-1}}$. In many cases, particularly in outdoor acoustics, the speed of sound changes depending on varying temperature and humidity in air. However, in this discussion, we assume a homogeneous environment with a constant speed of sound, denoted as $c = 343\,\mathrm{m\,s^{-1}}$.

Sound from a source travels directly in a path outwards from the source. When encountering obstacles, the sound maybe be reflected, where it continues at an equivalent angle off of a surface; diffused, where sound is scattered in a random manner from the surface; diffracted, or bent around an obstacles smaller than the wavelength of the sound; transmitted through an obstacle; or absorbed. Wave phenomena such as diffraction and scattering can have a significant effect on auralized sound (Torres et al., 2001a,b).

These propagation effects are governed by the time-domain wave equation in a homogeneous domain:

$$\frac{\partial^2}{\partial t^2}p(\vec{x},t) - c^2\nabla^2 p(\vec{x},t) = f(\vec{x},t), \tag{1.1}$$

where the pressure at time t and spatial position $\vec{x}$ is $p(\vec{x},t)$, the constant speed of sound is c, ∇^2 is the Laplacian, and $f(\vec{x},t)$ is a term for forcing sound pressure, used to represent a sound source or external boundary conditions on the domain.

An important concept in architectural acoustics is that of the room impulse response (IR). The impulse response is the output of an acoustic system when excited by an impulse source signal. Such a source signal can be defined by the Dirac function $\delta\left(t - t_0\right)$, which has an infinite value at t_0 and a value of zero everywhere else. The spectrum of the Dirac function is thus given by a constant value for all frequencies. Given a Dirac source signal at $\vec{x}_1$, where t_0 is the starting time delay of the sound source, solving Equation 1.1 at a specific listener location location $\vec{x}_2$ over a sufficient period of time will yield the impulse response $r\left(t\right)$ at that location.

The output $s'\left(t\right)$ of an acoustic system with respect to a source signal $s\left(t\right)$ can then be obtained by the convolution of the source signal with the impulse response originating from the same location (Kuttruff, 2016):

$$s'\left(t\right) = s\left(t\right) * r\left(t\right) \tag{1.2}$$

Additionally, the impulse response is useful for the evaluation of various acoustic metrics used in architectural acoustics. The impulse response function varies depending on the source and listener positions $\vec{x}_1$ and $\vec{x}_2$ and the acoustic properties of the environment, such as the topology or reflectivity of various surfaces in the environment.

Therefore, a fundamental aspect of evaluating a room or structure for the purposes of acoustic design is measuring or computing the impulse response. Computer models for evaluating the impulse response are particularly useful early in the design process when the structure is not actually built and the impulse response cannot be measured. However, Equation 1.1 is a partial differential equation (PDE) that has no closed form solution on arbitrary complex domains. As a result, many techniques have been developed for numerical approaches to solving these problems.

Typical techniques for solving sound propagation problems fall under two primary categories: geometric methods and wave-based methods. Geometric methods use the underlying assumption of sound waves being able to be represented by geometric primitives such as rays, beams, frustums, etc. (Funkhouser et al., 2003). Such techniques are usually compute performance-dependent logarithmically on the geometric complexity of the environment and the number of source and listener locations, but are efficient enough for real time applications. The drawback to geometric approaches is the limited ability of geometric representations of sound waves to capture all wave phenomena, including diffraction and scattering. There are geometric approaches for approximating scattering effects, but these techniques are often accurate only for low-order diffraction (Tsingos et al., 2001). Given the importance of scattering at lower frequencies, where obstacles are of similar size to the wavelength of propagated sound, geometric techniques are not sufficiently accurate to represent lower frequencies of sound propagation for the purpose of acoustic evaluation and design.

Wave-based techniques, on the other hand, numerically solve Equation 1.1. These include finite element methods (FEM) (Thompson, 2006), boundary element methods (Ciskowski and Brebbia, 1991), and finite difference time domain (FDTD) along with finite volume methods (Sakamoto et al., 2006; Bilbao, 2013; Kowalczyk and Van Walstijn, 2011). These techniques have well-defined convergence properties and accurately solve the wave equation. However, these techniques are computationally expensive, their asymptotic complexity for compute scaling with frequency to the fourth power and linearly with volume. Therefore, wave-based methods are often intractable for architectural acoustics at higher frequencies. For example, human beings primarily hear sound energy between 20 Hz and 20 000 Hz frequencies. Human speech generally is limited to between 125 Hz and 8000 Hz. However, most wave-based methods are limited to 1 kHz to 2 kHz due to their asymptotic compute cost scaling with frequency to the fourth power (Raghuvanshi et al., 2009a).

1.2 Acoustic Optimization

Multidisciplinary design optimization (MDO) is a wide ranging area of research in engineering and design. The primary focus of MDO techniques are to integrate various aspects of the design process, including structure, vibration, acoustics, and so on. It is commonly used in many areas including aerodynamic optimization of wings and entire aircraft, architectural features such as bridges and buildings, railway cares, microscopes, automobiles, turbines, and ships (Martins and Lambe, 2013). MDO applications intrinsically imply some sort of interface or constraint in a design process. For example, the structural design of a building may enforce constraints on what kind of acoustic materials may be used. This has an impact on the acoustic design of the building. We are particularly interested in how these techniques can be applied to acoustics and be used for improving the acoustics of a design while conforming to constraints in the design process.

The main purpose of acoustic design optimization is to change the shape, materials, topology, or source/listener positions in order to improve specific qualities of propagated sound. For example, shape optimization could include changing the dimensions of balconies in a concert hall to improve the distribution of acoustic energy in a scene, such as in (Robinson et al., 2014). Material optimization techniques attempt to change what materials are used in a design or to place acoustic absorbers to improve sound (Saksela et al., 2015). Topology optimization, such as the method proposed in Dühring et al. Dühring et al. (2008), involve generalized changes in the structure of a design, including placement of sound barriers, columns, walls, doorways, etc. The degree of freedom involved in topology optimization processes, however, makes this class of problems more challenging to solve. Finally source/listener position optimization problems typically involve the placement of speakers for sound field reproduction (SFR) applications or the design of home theater systems. These approaches typically select optimal placements for either sound sources or receivers (like a microphone or human listener).

The main advantage of these kind of design optimization techniques is that they allow engineers to solve complex engineering problems in an automatic way. While there exist many rule of thumb techniques or approximations in acoustic engineering (such as Sabine's law for estimating reverberation time in an enclosed area), computer optimization techniques can utilize both fundamental aspects of acoustic simulation models, such as complex wave phenomena, and a wider range of goals, such as multiobjective optimization where many acoustic metrics are optimized for at once. These methods often incorporate geometric methods for

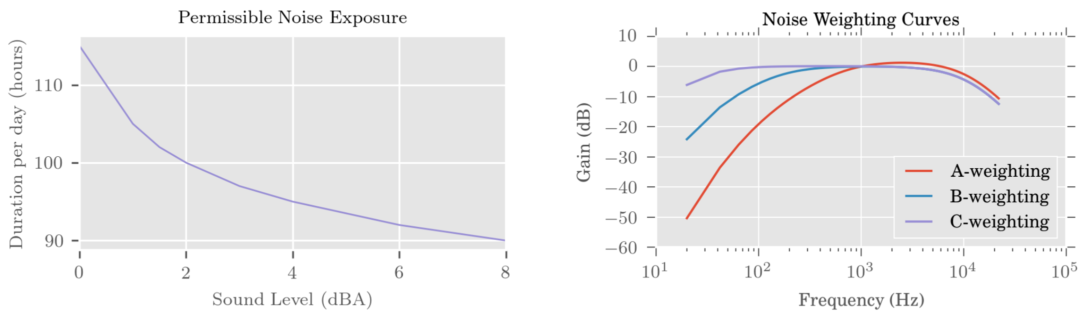


Figure 1.1: Plots showing permissible noise exposure and the noise weighting curves commonly used in regulatory agencies such as OSHA (OSHA, 2008). Human beings can suffer health problems or hearing damage from instantaneous exposure to noise over 115 dB or noise over 90 dB over the course of an 8-hour workday. Noise levels are weighted according to various weighting curves, including the most commonly used A-weighting curve that is adjusted according to the human frequency response to sound.

sound propagation (Monks et al., 2000; Saksela et al., 2015; Bassuet et al., 2014) which are efficient, but lack accuracy at lower frequencies.

Some of the optimization applications where acoustic design is important include concert hall and theater design, noise minimization, and sound environment reproduction.

The problem of designing theaters and concert halls for appealing audience experience is as old as the ancient Roman architects (Pollio, 1914). Everything, from the material used in construction, the size of architectural features, the topology, and the placement of the performers or audience can affect the acoustics and audience experience. Modern work in this area includes work by Bassuet et al. (2014), Robinson et al. (2014), and Monks et al. (2000) in the area of concert hall design.

Further optimization applications include the design of workplaces or sensitive areas such as schools and hospitals to limit noise that workers, patients, or students are subjected to. Excessive noise can have a large impact on human health (Birgitta et al., 1999; Basner et al., 2014; Mazer, 2005) in addition to environmental ecosystem health (Kight and Swaddle, 2011; Brumm, 2004). Noise has also been shown to have a negative affect on human learning processes (Bronzaft, 2012). In the United States, workplace sound safety is enforced by regulatory agencies such as the Occupational Safety and Health Administration (OSHA). This agency limits the allowable amount of noise that workers can be exposed to both at a single period of time and during the course of the entire work day (OSHA, 2008). Using optimization techniques to limit noise can ensure compliance with these regulations and ensure that workers and students are comfortable in their environment.

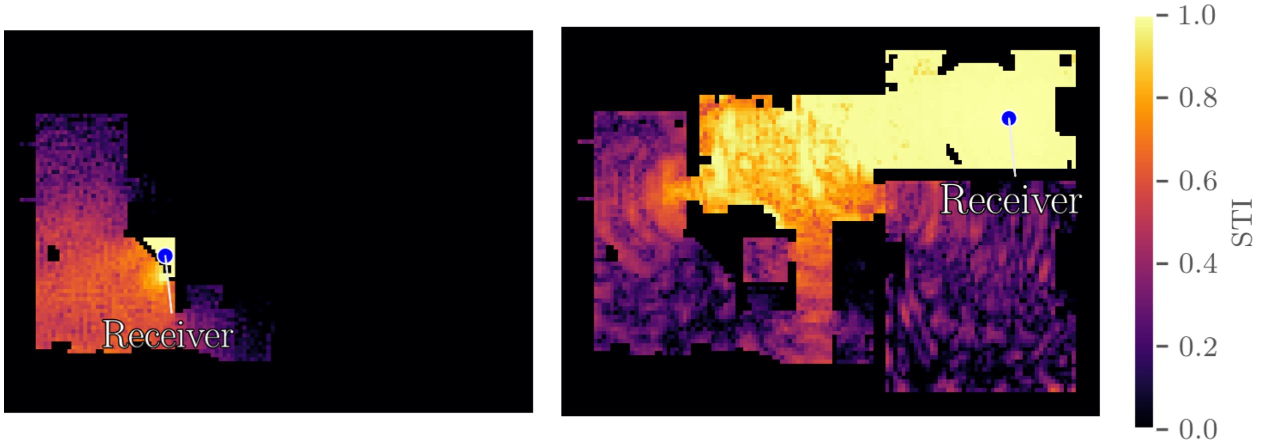


Figure 1.2: Side-by-side comparison of how speech intelligibility at the receiver location of an ASR device can change according to its location. Each datapoint on this plot corresponds to a user standing at that position and interacting with the ASR device. The STI metric used here is explored in more detail in Chapter 5.

Another recent field where noise has a detrimental effect is that of automatic speech recognition (ASR) algorithms (Rabiner and Juang, 1993). These algorithms translate spoken audio clips to text for a variety of applications including home automation systems, smart devices, electronic personal assistants, and so on. These techniques generally suffer under conditions of excessive noise and reverberation (Gillespie and Atlas, 2002). An open area of research is how these problems can be minimized through acoustic changes in the environment or placement of ASR devices. Figure 1.2 shows how simply relocating the receiver for an ASR device in a home can affect the quality of received sound.

Optimization procedures can also be used for the purpose of reproduction of sound environments. The placement of speakers for the purpose of sound field reproduction (SFR) in acoustic signal processing can be accomplished using acoustic optimization technique (Khalilian et al., 2015). It is also possible to use optimization techniques to match simulated and measured impulse responses for the purpose of discovering the acoustic material properties of various surfaces in the environment (Schissler et al., 2017).

1.3 Challenges

Currently, our ability to solve these acoustic problems are limited by the compute and memory cost of acoustic simulation (in the case of wave-based solvers) or the accuracy of the solvers (in the case of geometric techniques). These limitations of geometric and numerical methods beg the question on how one can efficiently and accurately compute sound propagation in an architectural environment. It is certainly the case that many architectural environments exhibit complex wave phenomena, such as diffraction and

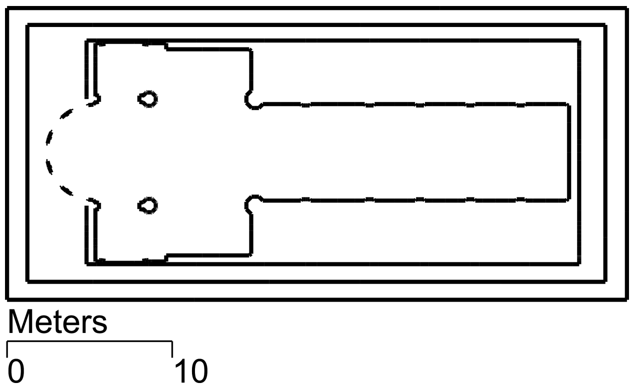


Figure 1.3: A top-down slice of the simulation domain of the Šibenik Cathedral. The overall size of the scene is $38 \times 18 \times 30$ m, which comes out to around $3.6x10^12$ cells in an FDTD discretization. Using FDTD the compute costs of sound propagation on this scene for the full range of human hearing are around 150 exaflops.

scattering. Diffraction occurs when the size of an obstacle is similar to the wavelength (Egan, 1988); given $c = 343\,\mathrm{m\,s^{-1}}$, obstacles of a size d meters will produce diffraction effects at $\frac{343}{d}$ Hz. Thus, a simple doorway a little over a meter wide is sufficient to diffract sound at around 350 Hz, which is well within the range of human speech and is similar in pitch to the mid-range of many orchestral instruments. Since geometric techniques do not accurately represent such phenomena, they can be inaccurate for architectural acoustic problems.

However, the aforementioned computational complexity of wave-based techniques make the problem of sound propagation using numerical methods a challenging prospect in many architectural scenes. Consider a large architectural scene, such as the Cathedral of St. James in Šibenik, Croatia (Figure 1.3). The simulation domain of this scene is almost $20\,000\,\mathrm{m}^3$. Spatially discretizing this environment for a standard FDTD method yields a grid composed of around 3 trillion cells. The total compute cost for evaluating an impulse response for the cathedral at 20 kHz (the full range of human hearing) is around 150 exaflops. Assuming best-case peak compute, an NVIDIA Tesla K80 GPU would take 4 days to compute this sound propagation problem. Similarly, an Intel Xeon Phi would take 12 days to finish. This is in addition to a total memory use of 28 terabytes, which is too large for the memory capacity of existing general purpose computing accelerators.

However, while it is possible to use geometric solvers, even at real time (Schissler and Manocha, 2011; Chandak et al., 2008; Taylor et al., 2012), on these size scenes, the inaccuracy of geometric solvers when representing diffraction and scattering effects mean that on architectural models at lower frequencies, some wave phenomena are not being accurately captured. Particularly when diffraction effects can have a significant impact on the sound that human beings hear, geometric solvers are not sufficiently accurate for architectural approaches (Savioja, 2010a). However, despite this and because of the performance advantages to using geometric approaches, they are prevalent in real-time auralization and analysis.

The performance disparity between geometric and wave-based approaches is even more apparent in acoustic optimization applications. In most optimization applications, the acoustic field or impulse response must be computed in the evaluation of the objective function (for example, in Monks et al. (2000)). Thus, optimization schemes with a large number of iterations (say, in the hundreds) is not tractable for most wave-based approaches on architectural environments and are often limited to smaller domains. Robinson et al. (2014), for example, use only a 2D approach in their concert hall shape optimization algorithm. However, in some optimization applications, it is possible to reduce the number of iterations and the number of solves required in evaluation of the objective functions. This approach can dramatically improve the performance of design optimization processes that use wave-based techniques.

A second element of difficulty that has implications on performance and accuracy is that many architectural acoustic problems have complex domains — it is straightforward to evaluate the propagation of sound in a simple scene such as a rectangular room (in fact, analytic solutions of the wave equation are possible (Kuttruff, 2016)), but more complex domains do not have closed form solutions to the wave equation.

Additionally, it is often desirable to enforce constraints on any kind of acoustic optimization process. For example, to avoid some of the problems mentioned earlier with materials that may be acoustically desirable, but not structurally practical, it is important to be able to constrain materials so that only certain materials are selected by the optimization process. In other acoustic problems, such as microphone or speaker placement, it is often the case that microphones and speakers cannot arbitrarily be placed in the scene. For example, a speaker array may obstruct a particular audience view. This can limit the scope of some optimization approaches.

Finally, the scope of some optimization applications include multiobjective optimization. For example, in the context of concert hall acoustics, there are several metrics that are important to audience experience (Beranek, 2011; Miśkiewicz et al., 2012). It may not be sufficient to simply optimize for one of the metrics. In other optimization problems, it is necessary to optimize for multiple listener locations. Rather than having a single audience member with a great experience at a concert (according to the aforementioned metrics), it is more important for all audience members to have that experience. This can have some implications for performance. Multiobjective optimization techniques can have difficulty converging, and thus can impact the number of iterations and number of evaluations of the objective function.

1.4 Thesis Statement

Large-scale distributed parallel wave-solvers and hybrid solvers can be used to accurately and efficiently solve acoustic propagation and optimization problems for computer-aided design.

The primary goal of this dissertation is to develop novel techniques for accurate and efficient sound propagation and optimization in architectural acoustics applications. Our focus is on computer-aided design, where simulation and optimization methods can be used to model how sound propagates in architectural environments and how the acoustic characteristics of a structure can be improved before the structure is constructed. First, we explore how large distributed parallel techniques can be used in conjunction with low-dispersion wave solvers in order to efficiently compute sound pressure fields on large indoor and outdoor domains. Second, we analyze how wave-based techniques can be used in acoustic material optimization problems. Finally, we discuss the limitations of solely wave-based approaches and how geometric techniques can be used to augment wave-based solutions for higher frequency sounds in the context of optimization problems.

1.5 Main Contributions

These goals are achieved through novel algorithms in acoustic simulation and optimization. We present Massively Parallel Adaptive Rectangular Decomposition (MPARD), a technique based on the Adaptive Rectangular Decomposition (ARD) approach (Raghuvanshi et al., 2009b), a low dispersion technique for solving sound propagation problems. We then present two new techniques for optimizing the acoustic materials of a CAD model using wave-based acoustics, in order to achieve desired acoustic characteristics. Finally, we present two hybrid optimization algorithms for the minimization of environmental noise in workplace environments and for the improvement of speech intelligibility for home and office environments that are using automated speech recognition (ASR) hardware.

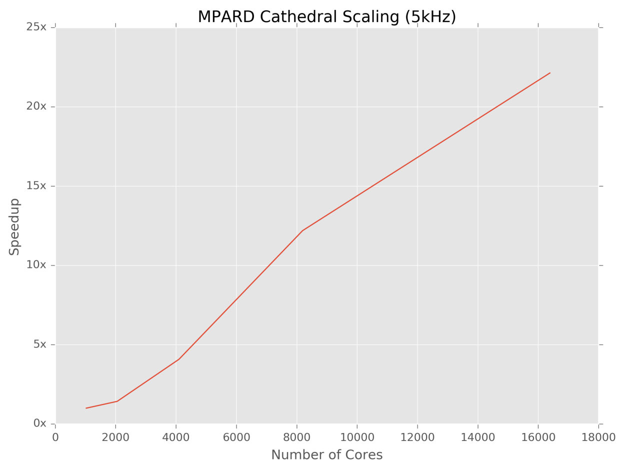


Figure 1.4: Scalability of the MPARD technique from 1024 to 16384 cores on the 5 kHz cathedral scene. We obtain close-to-linear scaling in this result. The base speedup is 1024 on 1024 cores (assuming 1024 cores is 1024 times faster than one core) since the scene will not fit in memory on a lower number of cores.

1.5.1 Parallel Wave-Based Sound Propagation

The computational and memory cost of accurate wave-based techniques for high frequencies is often intractable for architectural acoustic applications. Recent advances in low dispersion techniques have improved numerical dispersion errors of finite difference and finite element type solvers (Guddati and Thirunavukkarasu, 2016; Davis, 1991; Hu et al., 1996; Savioja and Valimaki, 2000a). As such, minimized dispersion error can reduce the number of samples required per wavelength of the discretization, allowing greater computational efficiency. One such technique, Adaptive Rectangular Decomposition (ARD) (Raghuvanshi et al., 2009b) minimizes dispersion error through the use of analytic subdomains, thus requiring 2-3 samples per wavelength as opposed to 10 samples per wavelength that is common in many FDTD methods. However, the computational cost is still well within the exaflop range for large acoustic problems at high frequencies.

Therefore, we propose a novel parallel ARD algorithm, called MPARD. This technique is scalable on large supercomputing clusters, tested up to 16000 cores (see Figure 1.4). Using this technique, we are able to simulate acoustic sound propagation on large indoor and outdoor architectural scenes up to 10 kHz frequencies. This is far beyond the limitations of traditional methods, which generally remain in the 1 kHz to 2 kHz range. The main contributions of this algorithm are as follows:

- An asynchronous scheme for communication and evaluation ARD subdomains and interfaces in parallel
- A domain splitting technique for ensuring load balancing while minimizing numerical error

- Hypergraph partitioning for load balancing and minimizing communication cost among cores
- A preprocessing step for allocating work efficiently among cores

These components of the algorithm ensure a linearly scalable technique that has been tested up to 16000 cores. Additionally, we have evaluated the technique on various indoor and outdoor CAD models. Using this technique, we can efficiently and accurately solve acoustic propagation problems for high frequencies for the purpose of evaluating an acoustic design.

1.5.2 Wave-Based Techniques for Acoustic Material Optimization

A standing problem in acoustic design is that of selecting the correct materials for construction in order to yield the desired sound. Buildings such as concert halls and auditoriums often have certain desirable acoustic characteristics that can be measured using the impulse response produced by sound propagating.

We present two novel techniques for optimizing the acoustic materials used in the design of structures such as concert halls and auditoriums. In the first method, we use a continuous gradient-based optimization scheme for minimizing the distance of various acoustic metrics from the goal metrics. In the second method, we use a Pareto-optimal simulated annealing minimization technique for optimizing materials under discrete constraints. Overall, our contributions can be summarized as follows:

- Multiobjective optimization methods for determining acoustic materials that yield the desired acoustic properties in a CAD model
- Efficient evaluation of acoustic metrics using impulse responses computed by the ARD wave-based technique
- Automatic differentiation techniques for computing the analytical gradient, enabling faster convergence in continuous optimization
- A Pareto-optimal simulated annealing technique for discrete multiobjective optimization
- Constraint-based optimization using acoustic material libraries

Using these techniques, we can obtain material configurations of a scene such that the desired acoustic characteristics are closely matched. We test our method on various benchmarks, including concert halls, auditoriums, and indoor-outdoor architectural features.

1.5.3 Hybrid Techniques for Acoustic Optimization

Previous wave-based optimization techniques are limited to lower frequencies, less than 500 Hz. With explicit time domain methods, the time step of the method is governed by the Courant-Friedrichs-Lewy condition (CFL). At higher acoustic frequencies, the time step becomes smaller, but the total time range of the simulation must remain the same for a scene. Therefore a large part of the computational cost of explicit time domain methods is due to the increase in the total number of simulation steps. This is not as straightforward to parallelize as the spatial domain and thus becomes a much larger factor in the performance of iterative optimization methods where the objective function may need be evaluated hundreds of times of the course of optimization.

To address these problems, we introduce an optimization technique using hybrid acoustic simulation. In hybrid propagation, we use expensive wave-based techniques for the lower simulation frequencies, where accurate solutions to the wave equation are necessary for representing effects such as diffraction and scattering that are more noticeable. Then, higher frequencies are simulated using geometric methods. At these frequencies, wave effects are less prevalent and wave-based methods are more expensive. We present two applications for this technique. The first application technique uses this hybrid approach in order to place noise emitters in an environment such that noise reaching specific listener regions is minimized. Our second application aims to improve the effectiveness of automatic speech recognition (ASR) hardware by placing the speech receiver in an environment where speech intelligibility is maximized. Our contributions include:

- A hybrid acoustic solver for improved efficiency while retaining the accuracy of solely wave-based approaches
- Sound source clustering for reducing the optimization search space
- Impulse response caching for reducing the number of solves required for evaluating the objective function
- Adaptive start and end criteria for simulated annealing, reducing the number of optimization iterations

These approaches are evaluated in a large variety of scenes, from office spaces, industrial parks, and warehouses where environmental noise can cause health problems in workers to houses and apartments where consumers may use automatic speech recognition devices. We show an improvement in the achievable frequency range of our optimization techniques by using hybrid acoustic simulation.

1.6 Organization

The rest of this dissertation is organized as follows:

Chapter 2 provides some of the mathematical background of the techniques used in the dissertation in addition to providing a survey of prior work in the topics covered.

Chapter 3 introduces parallel sound propagation with ARD and how it can be extended to compute clusters with tens of thousands of available cores.

Chapter 4 presents acoustic material optimization techniques using wave-based sound simulation for evaluating a variety of acoustic metrics.

Chapter 5 describes a frequency-partitioned hybrid sound propagation technique that is used for various optimization applications, including noise minimization and the maximization of speech intelligibility.

Chapter 6 Concludes the dissertation, along with thoughts about limitations and future challenges in the area of wave-based acoustics in computer aided design.

CHAPTER 2: BACKGROUND

This chapter provides an overview of some of the previous work in sound propagation research, along with work done in optimization and noise reduction. Background on the Adaptive Rectangular Decomposition method (ARD) that is central to many of the algorithms presented in this dissertation is also included.

2.1 Sound Propagation

Methods for solving the acoustic wave equation (Equation 1.1) are generally separated into two main categories: the geometric methods and the wave based methods.

2.1.1 Geometric Methods

Geometric methods generally model of sound propagating in the form of a ray or other geometric primitive, and work on the assumption that the wavelength of modeled sound is much smaller than the size of obstacles in the environment (Funkhouser et al., 2003).

Geometric methods tend to fall under the category of image source methods, ray tracing techniques, and beam tracing or frustum tracing approaches. Image source methods mainly consider specular paths of sound, i.e. sound that travels in a mirror-like manner when reflecting off an obstacle. They model sound by mirroring audio sources over polygonal surfaces in the environment. Image source methods were introduced by Allen and Berkley Allen and Berkley (1979). Additional approaches include a generalization to arbitrary-sided polyhedra (Borish, 1984). Schissler et al. Schissler and Manocha (2011) introduce a listener image method, where this technique is used to cache propagation paths in conjunction with backwards ray-tracing.

Ray tracing techniques primarily encompass techniques where reverberation paths are found between the source and listener, either by emanating rays from the source or from the listener. These approaches often include stochastic methods, such as Monte Carlo integration, for diffuse reflections. Ray tracing approaches for sound propagation were introduced by Krokstad Krokstad et al. (1968). Further approaches include techniques for simulated diffraction and scattering effects include (Tsingos et al., 2001; Embrechts et al., 2001; Tsingos et al., 2007; Schissler et al., 2014a).

Beam tracing methods include intersections between geometry and pyramidal sets of rays. These approaches include beam tracing for interactive environments (Funkhouser et al., 1998), hybrid beam tracing and image source methods (Monks et al., 1996), and frustum tracing (Chandak et al., 2008).

2.1.2 Wave-Based Sound Propagation

Wave-based approaches, on the other hand, numerically solve the wave equation (1.1). These approaches encompass implicit and explicit time-domain algorithms. Wave-based approaches are generally limited to lower frequencies, as higher frequencies require dramatically more compute power and memory (compute costs asymptotically increase as $O\left(f^4\right)$).

Finite-element techniques use a subdivision into linear elements for solving the Helmholtz equation, and ensure convergence with either the collocation method or Galerkin method for global convergence (Funkhouser et al., 2003). FEM techniques include works by Nefske et al. Nefske et al. (1982) and Craggs Craggs (1972). An overview of these approaches can be found in Thompson Thompson (2006). Boundary-element methods work in a similar manner, and include approaches for solving radiation and scattering problems (Chen et al., 2010). An overview of boundary element methods can be found in (Ciskowski and Brebbia, 1991).

Explicit time-domain methods include finite-difference time domain approaches (FDTD). These approaches include work in FDTD for room acoustics (Sakamoto et al., 2006) and work in both finite differences and finite volume approaches (Bilbao, 2013). FDTD approaches have been adapted for real time applications (Savioja, 2010a). Low-dispersion techniques for finite-difference methods include the interpolated wideband scheme (Savioja and Valimaki, 2003; Kowalczyk and Van Walstijn, 2011) and modified wave-guide approaches (Savioja and Valimaki, 2000a). Other low-dispersion approaches in the family of FDTD methods include the Adaptive Rectangular Decomposition approach (Raghuvanshi et al., 2009b).

2.2 Adaptive Rectangular Decomposition

For the techniques put forth in this dissertation, we primarily use the Adaptive Rectangular Decomposition (ARD) solver (Raghuvanshi et al., 2009b; Mehra et al., 2012; Morales et al., 2015a), including mathematical details that are relevant to this thesis.

2.2.1 Pressure Field Computation

Consider a cuboidal domain in three dimensions of size (l_x, l_y, l_z) with perfectly reflecting walls. The acoustic wave equation on this domain has an analytical solution represented as:

$$p(x,y,z,t) = \sum_{i=(i_x,i_y,i_z)} m_i(t)\Phi_i(x,y,z), \tag{2.1}$$

where m_i are the time-varying mode coefficients and Φ_i are the eigen functions of the Laplacian for the cuboidal domain, given by:

$$\Phi_i(x,y,z) = cos\left(\frac{\pi i_x}{l_x}x\right) cos\left(\frac{\pi i_y}{l_y}y\right) cos\left(\frac{\pi i_z}{l_z}z\right). \tag{2.2}$$

We limit the modes computed to the Nyquist rate of the maximum simulation frequency.

In order to compute the pressure, we need to compute the mode coefficients. Reinterpreting equation (2.1) in the discrete setting, the discrete pressure $P(X,t)$ corresponds to an inverse Discrete Cosine Transform (iDCT) of the mode coefficients $M_i(t)$:

$$P(X,t) = iDCT(M_i(t)). \tag{2.3}$$

Substituting the above equation in Equation 1.1 and applying a DCT operation on both sides, we get

$$\frac{\partial^2}{\partial t^2}M_i + c^2{k_i}^2 M_i = DCT(F(X,t)), \tag{2.4}$$

where ${k_i}^2 = \pi^2\left(\frac{{i_x}^2}{{l_x}^2} + \frac{{i_y}^2}{{l_y}^2} + \frac{{i_z}^2}{{l_z}^2}\right)$ and $F(X,t)$ is the force in the discrete domain. Assuming the forcing term $F(X,t)$ to be constant over a time-step Δt of simulation, the following update rule can be derived for the mode coefficients:

$$M_i^{n+1} = 2M_i^n cos(\omega_i \Delta t) - M_i^{n-1} + \frac{2\widetilde{F}^n}{\omega_i^2}(1 - cos(\omega_i \Delta t)), \tag{2.5}$$

where $\widetilde{F} = DCT(F(X,t))$. This update rule can be used to generate the mode-coefficients (and thereby pressure) for the next time-step. This gives us a method to compute the analytical solution of the wave equation for a cuboidal domain. Next, we describe how these solutions are used to solve the wave equation over the entire scene.

2.2.2 Interface Handling

In order to ensure correct sound propagation across the boundaries of these subdomains, we use a (2,6) finite difference stencil to patch two subdomains together. A 6th order scheme was chosen because has been experimentally determined (Mehra et al., 2012) to produce reflection errors at 40 dB below the incident sound field. We derive the stencil as follows.

First, we examine the projection along an axis of two neighboring axis-aligned cuboids. Our local solution inside the cuboids assumes a reflective boundary condition, $\left.\frac{\partial p}{\partial x}\right|_{x=0} = 0$. Looking at the rightmost cuboidal partition, the solution can be represented by the discrete differential operator, ∇^2_{local}, that satisfies the boundary solutions. Referring back to the wave equation, we have the global solution:

$$\frac{\partial^2 p}{\partial t^2} - c^2 \nabla^2_{global} p = f(X,t).$$

Using the local solution of the wave equation inside the cuboid (∇^2_{local}), we derive:

$$\frac{\partial^2 p}{\partial t^2} - c^2 \nabla^2_{local} p = f(X,t) + f_I(X,t),$$

where $f_I(X,t)$ is the forcing term that needs to be contributed by the interface to derive the global solution. Therefore, we need to solve for this term, using our two previous identities:

$$f_I(X,t) = c^2 \left(\nabla^2_{global} - \nabla^2_{local}\right) p. \tag{2.6}$$

While the exact solution to this is computationally expensive, we can approximate the solution using a 6th order finite difference stencil, with spatial step size h:

$$f_I(x_j) = \sum_{i=j-3}^{-1} p(x_i)s[j-i] - \sum_{i=0}^{2-j} p(x_i)s[i+j+1], \tag{2.7}$$

where $j \in [0,1,2]$, $f_I(x_j) = 0$ for $j > 2$, and $s[-3 \ldots 3] = \frac{1}{180h^2}\{2, -27, 270, -490, 270, -27, 2\}$.

Additionally, interfaces between subdomains that cover homogeneous air regions and subdomains that are solid are modeled by this method. The forcing terms for an air-wall interface are computed as follows:

$$f_I(x_j) = \sum_{i=j-3}^{-1} p(x_i)s[j-i]\alpha - \sum_{i=0}^{2-j} p(x_i)s[i+j+1]\alpha, \tag{2.8}$$

where α is a material absorption value.

The absorption value α is used to model absorption at air-wall boundaries. If this value is zero, then the FDTD stencil is not taken into effect and the wall is regarded as perfectly reflective. On the other hand, if the value is one, the FDTD is taken into full effect and the wave is propagated into the PML partition where it is absorbed. Similarly, values used between zero and one result in partial reflectance.

A limitation of this approach is that only a single absorption value is used that does not take into account the dependence on the frequency. For more realistic materials, it is necessary to run the simulator for multiple frequency bands, i.e. multiple bands in the human hearing range 20 Hz to 20 000 Hz, though in practice current solvers are limited to a few thousand kilohertz.

2.2.3 ARD computational pipeline

The ARD technique has two main stages: *Preprocessing* and *Simulation*. During the preprocessing stage, the input scene is voxelized into grid cells. The spatial discretization of the grid h is determined by the maximum simulation frequency ν_{max} determined by the relation $h = c/(\nu_{max}s)$, where s is the number of samples per wavelength (=2.66 for ARD) and c is the speed of sound ($343m/sec$ at standard temperature and pressure). The next step is the computation of adaptive rectangular decomposition, which groups the grid cells into rectangular (cuboidal) domains. These generate the partitions, also known as *air partitions*. Pressure-absorbing layers are created at the boundary by generating Perfectly-Matched-Layer (PML) partitions. These partitions are needed to model partial or full sound absorption by different surfaces in the scene. Artificial interfaces are created between the air-air and the air-PML partitions to propagate pressure between partitions and combine the local pressure fields of the partitions into the global pressure field.

During the simulation stage, the global acoustic field is computed in a time-marching scheme as follows (see Figure 3.2 (top row)):

1) Local update

For all air partitions

(a) Compute the DCT to transform force F to $\widetilde{F}$.

(b) Update mode coefficients M_i using update rule.

(c) Transform M_i to pressure P using iDCT.

For all PML partitions

Update the pressure field of the PML absorbing layer.

2) Interface handling

For all interfaces

(a) Compute the forcing term F within each partition.

3) Global update

Update the global pressure field.

The first stage solves the wave equation in each rectangular region by taking the Discrete Cosine Transform (DCT) of the pressure field, updating the mode coefficients using the update rule, and then transforming the mode coefficients back into the pressure field using inverse Discrete Cosine Transform (iDCT). Both the DCT and iDCT are implemented using fast Fast Fourier Transforms (FFT) libraries. Overall, this step involves two FFT evaluations and a stencil evaluation corresponding to the update rule. The pressure fields in the PML-absorbing layer partitions are also updated in this step.

The second stage uses a finite difference stencil to propagate the pressure field between partitions. This involves a time-domain stencil evaluation for each grid cell on the boundary.

The last stage uses the forcing terms computed in the interface handling stage to update the global pressure field.

CHAPTER 3: A Massively Parallel Solver for Large Compute Clusters

The main challenge is that the computational cost of computing the sound propagation in the environment scales with the 4th power of frequency and linearly with the volume of the scene, while memory use scales with the 3rd power of frequency and linearly with the scene volume. Since the human aural range scales from 20 Hz to 20 kHz, large scenes such as a cathedral (10 000 m^3 to 15 000 m^3) would require tens of Exaflops of computation and tens of terabytes of memory to compute using prior wave-based solvers. This makes high frequency acoustic wave simulation one of the more challenging problems in scientific computation (Engquist and Runborg, 2003).

Recently, there has been a lot of emphasis on reducing the computational cost of wave-based methods. These techniques, called *low dispersion techniques* use coarser computational meshes or decompositions in order to evaluate the wave equation. One example of a low dispersion algorithm is the Adaptive Rectangular Decomposition method (ARD) (Raghuvanshi et al., 2009b; Mehra et al., 2012), a solver for the 3D acoustic wave equation in homogeneous media. ARD is a domain decomposition method that subdivides the computational domain into rectangular regions. One of the main advantages of ARD is the greatly reduced computational and memory requirements over more traditional methods like FDTD (Mehra et al., 2012).

Previously, ARD has been used to perform acoustic simulations on small indoor and outdoor scenes for a maximum frequency of 1 kHz to 2 kHz using only a few gigabytes of memory on a high-end desktop machine. However, performing accurate acoustic simulation for large acoustic spaces up to the full auditory range of human hearing still requires terabytes of memory. Therefore, there is a need to develop efficient parallel algorithms, scalable on distributed memory clusters, to perform these large-scale acoustic simulations. In this chapter, we discuss our algorithms for parallelizing ARD. This work was originally published in (Morales et al., 2015a) and (Morales et al., 2017).

Main results: We present MPARD, a Massively Parallel Adaptive Rectangular Decomposition method capable of computing 3D sound propagation through large architectural and outdoor scenes for pressure field computations at large frequencies (at least 10 kHz) and is designed to utilize tens of thousands of CPU cores.

Scaling to this number of CPU cores with such a large domain introduces many challenges, including communication cost, numerical stability, and reducing redundant computations. MPARD addresses all of these in a novel method that is comprised of:

- An efficient asynchronous communication technique for overlapping communication between cores with computation
- A hypergraph partitioning approach for load balance and communication reduction among multiple nodes
- A subdomain splitting algorithm to maintain the numerical accuracy of our wave-based simulator while ensuring subdomain computations are load balanced

MPARD is capable of computing the acoustic propagation in large indoor scenes (20 000 m^3) up to at least 10 kHz at around 80 s per time step on 1024 cores. The algorithm has been tested to and scales up to at least 16000 cores, where each time step of a 5 kHz scene takes around 0.2 s. We have analyzed many aspects of our parallel algorithm including the scalability over tens of thousands of cores, the time spent in different computation stages, communication overhead, and numerical errors. To the best of our knowledge, this is one of the first wave-based solver that can handle such large domains and high frequencies.

3.1 Previous work

There has been considerable research in the area of wave-based acoustic solvers, including the field of parallel solvers, domain decomposition approaches, and low-dispersion acoustic solvers. In this section, we provide a brief overview of some of these approaches.

3.1.1 Parallel wave-based solvers

Parallel wave-based solvers are used in a multitude of scientific domains, including the studying of seismic, electromagnetic, and acoustic waves. A large category of these solvers are parallel FDTD solvers either for large clusters (Guiffaut and Mahdjoubi, 2001; Vaccari et al., 2011; Yu et al., 2005, 2008) or for GPUs (Saarelma and Savioja, 2014; Savioja, 2010b; Sheaffer and Fazenda, 2014; Sypek et al., 2009; Webb and Bilbao, 2012; Salomons et al., 2016; Hornikx et al., 2016). A category of parallel methods are also based on finite-element schemes (Diekmann et al., 1996; Bhandarkar and Kalé, 2000).

Additionally, there are several parallel methods developed for specific applications of the wave equation. PetClaw (Alghamdi et al., 2011) is a scalable distributed solver for time-dependent non-linear wave propagation. Other methods include distributed finite difference frequency-domain solvers for visco-acoustic wave propagation (Operto et al., 2007), discontinuous Galerkin solvers for heterogeneous electromagnetic and aeroacoustic wave propagation (Bernacki et al., 2006), scalable simulation of elastic wave propagation in heterogeneous media (Bao et al., 1998), etc.

3.1.2 Domain decomposition

MPARD, like ARD, is a domain decomposition method. Domain decomposition approaches subdivide the computational domain into smaller domains that can be solved locally. The solver uses these local solutions to compute a global solution of the scientific problem. Many of these approaches are designed for coarse grain parallelization where each subdomain or a set of subdomains is computed locally on a core or a node on a large cluster.

Domain decomposition approaches tend to fall into two categories: *overlapping subdomain* and *non-overlapping subdomain* methods (Chan and Widlund, 1994).

Existing domain decomposition approaches in computational acoustics include parallel multigrid solvers for 3D underwater acoustics (Saied and Holst, 1991), pseudospectral acoustic solvers (Hornikx et al., 2016; Zeng et al., 2004), and the non-overlapping subdomain ARD approach (Raghuvanshi et al., 2009b). Additional non-overlapping approaches include Trefftz methods that can be decomposed for parallelization (Pluymers et al., 2007) and line source decomposition methods for ESM (equivalent source method) techniques (Hornikx and Forssén, 2007).

3.1.3 Low dispersion acoustic solvers

The goal of low dispersion methods is to reduce computation cost by using coarser computational meshes. A wide variety of approaches exist including an interpolated wideband scheme (Kowalczyk and van Walstijn, 2011), extended pseudospectral methods for 3D domains (Hornikx et al., 2010), and a variety of low dispersion BEM approaches (Sakuma et al., 2014). Waveguide approaches similarly reduce computation costs by reducing the computational grid size (Van Duyne and Smith, 1993; Savioja et al., 1995; Savioja and Valimaki, 2000b).

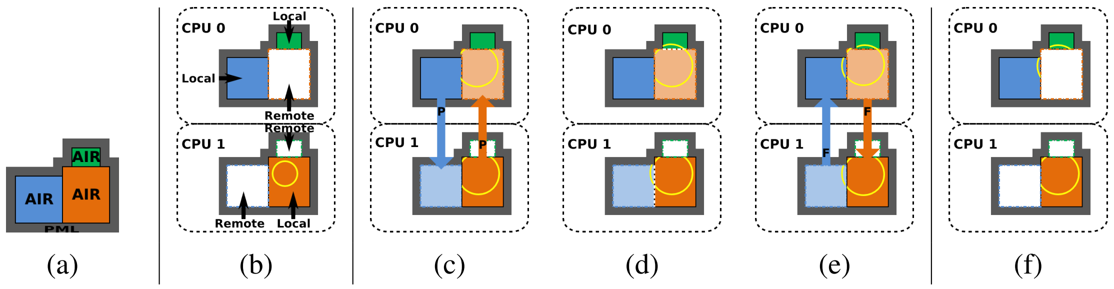


Figure 3.1: The parallel ARD pipeline. The initial partitioning and voxelization is the input into the simulation (a). Next, the local partition update occurs, evaluating the exact solution for an individual partition rectangle (b). In order to calculate interfaces, we need to transfer the pressure terms from the spanned partitions to the core that is evaluating the interface (c). The interface is then evaluated (d) and the forcing terms transferred back (e). Finally, global update occurs as the forcing terms are used to compute the global pressure field (f).

Our approach is primarily based on the ARD method (Raghuvanshi et al., 2009b) which partitions the computational domain into rectangular regions. It utilizes the property that the wave equation has a closed form solution in a homogeneous rectangular domain. Therefore, the only numerical error originates from the interfaces between these rectangular regions, allowing a much coarser grid size.

3.2 Parallel ARD

Table 3.1 summarizes the problem size limitations of previous ARD algorithms (see Chapter 2), including parallel ARD. Previous algorithms have been limited to problem sizes under 1 kHz to 2 kHz on large architectural scenes. The MPARD algorithm, on the other hand, is capable of operating on frequencies an order of magnitude greater than previous ARD methods. However, all algorithms are based on the same fundamental principles.

Solver	Num. Cores	Maximum Frequency on Cathedral
ARD (Raghuvanshi et al., 2009b)	1	1000 Hz to 2000 Hz
GPUARD (Mehra et al., 2012)	480 CUDA cores	1650 Hz
MPARD	16384	10 000 Hz

Table 3.1: Comparison between different ARD techniques include MPARD, the method we introduce. We use the Cathedral scene as a benchmark, which is 19 177 m^3 in volume. We show the highest frequency used to obtain results using these techniques. In the case of GPUARD memory becomes a limiting factor, while in ARD computation time is. In this chapter, we introduce the MPARD technique, which can scale to much higher frequencies (10 kHz) and a much higher number of cores.

3.2.1 Partition and Interface Ownership

This first step of parallelization involves distributing the problem domain onto the cores of the cluster. In the case of ARD, the different domains include air partitions, PML partitions, and the interfaces.

Air and PML partitions: The ARD solver can be parallelized because the partition updates for both the air and the PML partitions are independent at each time step: each partition update is a localized computation that does not depend on the data associated with other partitions. As a result, partitions can be distributed onto separate cores of the cluster and the partition update step is evaluated in parallel at each time step without needing any communication or synchronization. In other words, each core exclusively handles a set of partitions. These *local partitions* compute the full pressure field in memory. The rest of the partitions are marked as *remote partitions* for this core and are evaluated by other cores of the cluster. Only metadata (size, location, etc.) for remote partitions needs to be stored on the current core, using only a small amount of memory (see Figure 3.1 (b)).

Interfaces: Interfaces, like partitions, retain the concept of ownership. One of the two cores that owns the partition of an interface, takes ownership of that interface, and is responsible for performing the computations with respect to that interface. Unlike the partition update, the interface handling step has a data dependency with respect to other cores. Before the interface handling computation is performed, pressure data needs to be transferred from the dependent cores to the owner (Figure 3.1 (c)). In the next step, the pressure data is used, along with the source position, to compute the forcing terms (Figure 3.1 (d)). Once the interface handling step is completed, the interface-owning core must send the results of the force computation back to the dependent cores (Figure 3.1 (e)). The global pressure field is updated (Figure 3.1 (f)) and used at the next time step.

3.2.2 Parallel Algorithm

The overall technique proceeds in discrete time steps (see Figure 3.1). Each time step in our parallel ARD algorithm is evaluated through three main stages described in Chapter 2; these evaluations are followed by a barrier synchronization at the end of the time step. Each core starts the time step at the same time, then proceeds along the computation without synchronization until the end of the time step.

Local Update: This step updates the pressure field in the air and PML partitions for each core independently, similarly to the basic ARD algorithm (see Chapter 2 for details). However, only partitions owned on a core are evaluated. Therefore, each core computes the local step in parallel with other cores.

Pressure field transfer: After an air partition is updated, the resulting pressure data is sent to all interfaces that are dependent upon this data. The pressure data is transferred to the destination core asynchronously as soon as it becomes available. Partitions are evaluated in order from partitions with the most neighbors to those with the least neighbors; this ensures that dependent interfaces are not starved of pressure data.

Interface handling: This stage uses the pressure transferred in the previous stage to compute forcing terms for the partitions. Before the interface can be evaluated, it needs data from all of its dependent partitions. Since partitions are evaluated in order of the number of dependent interfaces, data starvation is minimized.

Force transfer: After an interface is computed, the owner needs to transfer the forcing terms asynchronously back to the dependent cores. A core receiving forcing terms can then use them as soon as the message is received.

Global update: Each core updates the pressure field using the forcing terms received from the interface operators.

Barrier synchronization: A barrier is needed at the end of each time step to ensure that the correct pressure and forcing values have been computed. This is necessary before local update is performed for the next time step.

3.2.3 Efficient Load Balancing

For each time step, the computation time is proportional to the time required to compute all of the partitions. This implies that a core with larger partitions or more partitions than another core would take longer to finish its computations; this would lead to load imbalance in the system, causing the other cores of the cluster to wait at the synchronization barrier instead of doing useful work. Load imbalance of this kind results in suboptimal performance.

Imbalanced partition sizes in ARD are rather common as ARD's rectangular decomposition step uses a greedy scheme based on fitting the biggest possible rectangle at each step. This can generate a few big partitions and large number of small partitions. In a typical scene (e.g. the Cathedral benchmark), the load imbalance results in poor performance.

A naive load balancing scheme would reduce the size of each partition to exactly one voxel, but this would negate the advantage of the rectangular decomposition scheme's use of analytical solutions; furthermore, additional interfaces introduced during the process at partition boundaries would reduce the overall accuracy of the simulator (Raghuvanshi et al., 2009b). The problem of finding the optimal decomposition scheme to generate perfect load balanced partitions while minimizing the total interface area is non-trivial. This problem is known in computational geometry as *ink minimization* (Keil, 2000). While the problem can be solved for a rectangular decomposition in two dimensions in polynomial time, the three-dimensional case is NP-complete (Dielissen and Kaldewaij, 1991). As a result, we approach the problem using a top-down approximate technique that bounds the sizes of partitions and subdivides large ones yet avoids increasing the interface area significantly. This is different from a bottom-up approach that would coalesce smaller partitions into larger ones.

Each rectangular region that has a volume greater than some volume threshold Q is subdivided by the new algorithm. Q is determined by the equation

$$Q = \frac{V}{pf}, \tag{3.1}$$

where V is the total air volume of the scene in spatial discretization units, p is the number of cores the solver is to be run on, and f is the balance factor. It is usually the case that $f = 1$, although this can be changed to a higher value if smaller rectangles are desired. However, in general, smaller values of Q increase the overall interface error of the solver since it increases the total interface area.

3.3 Scaling to Large Clusters

However, when scaling to large clusters, care must be taken to efficiently handle communication. Additionally when splitting subdomains for the purpose of load balancing, it is important to consider the numerical error that can be introduced by poorly-formed partitions. Finally, we show how work allocation for cores can be done as a preprocessing step.

3.3.1 Communication efficiency

Each interface in the scene covers two or more subdomains (for example, if the subdomains on either side are especially thin, the 6th order stencil may cover more than two). When subdomains on an interface are owned by different cores, the pressure terms required by the interface and the forcing terms generated by the interface need to be communicated.

MPARD uses asynchronous communication calls to avoid blocking while sending messages. When a core completes an operation (either a subdomain update or an interface update), it can send off a message to the cores that require the computed results. The sending core does not have to wait for the message to be received and can continue working on the next computation. When receiving data, a core can place the message on an internal queue until it needs the data for that operation. As a particular core finishes working on a subdomain, it sends off an asynchronous communication message to cores that require the pressure field of that subdomain for interface computation. As the message is being sent, that particular core begins to process the next subdomain, sending off a message upon completion. At the same time the receiving core is working on its own subdomains. The incoming message is stored on an internal queue for use when needed.

Another important thing to note is that in general ARD's communication cost of evaluating all the interfaces is proportional to the surface area of the rectangular region while the cost of evaluating each subdomain is proportional to the volume of the subdomain. This implies a rough $O(n^2)$ running time for interface evaluation (where n is the length of one side of the scene) while local update has an $O(n^3)$ running time. This means that as the size of the scene or the simulation frequency increases, the computation cost of local update dominates over the communication cost of transferring pressure and forcing values for interface evaluation. On the other hand the finer grid size of larger complex scenes can introduce many more interfaces which causes an increase in the cost of communication.

In order to evaluate these interfaces, communication is only required when resolving the interface between two or more subdomains that lie on different cores. As a result, minimizing the number of these interactions can greatly reduce the total amount of communicated data.

This can be performed by ensuring that neighboring subdomains that are part of the same interface are located on the same core. Due to the complexity of the scene and the interactions between different rectangular regions, this is not feasible. However, we can use a heuristic algorithm to minimize the number of interfaces that span across multiple cores.

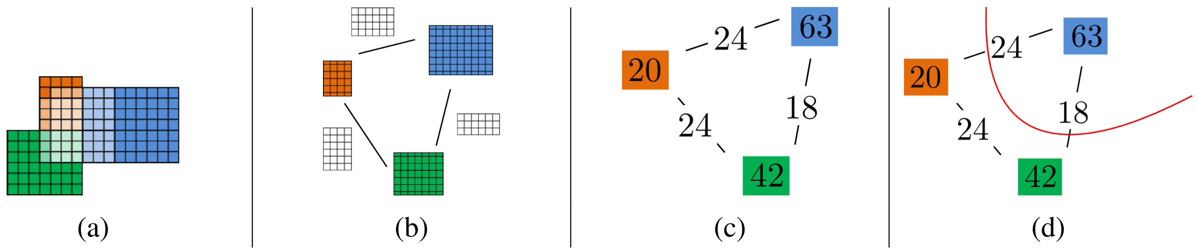


Figure 3.2: The relationship between computational elements of MPARD (rectangular subdomains and interfaces) and the hypergraph structure. (a) shows an example scene of three subdomains with three interfaces (shown in grey). The organization of the resulting hypergraph is shown in (b), while (c) shows the hypergraph with the node weights determined by the sizes of the rectangles and the hyperedge weights determined by the size of the interfaces. Subfigure (d) shows an example partitioning of the simple graph, taking into account both the computation cost of each node and the cost of each interface.

This problem can be reworded as a hypergraph partitioning problem. Our hypergraph can be represented as the pair $H = (X, E)$ where X, the nodes of the hypergraph are the rectangular regions of our decomposition, while the hyperedges E represent the interfaces between the rectangular regions. Hyperedges are used rather than regular edges because an interface can actually cover more than two subdomains.

The goal of the hypergraph partitioning algorithm is to divide the hypergraph into k regions such that the cost function of the hyperedges spanning the regions is minimized (Çatalyürek and Aykanat, 2011). In ARD, the partitioning algorithm can be run to divide our computational elements (interfaces and rectangles) into k regions, where k is the number of processors used in the simulation. As a result, the interface cost between cores is minimized.

Additionally, because the hypergraph partitioning algorithm attempts to generate k regions of equal cost, the heuristic serves as a way of load balancing the assignment of work to cores. The cost of evaluating a rectangular region is linearly related to the volume of the region. Therefore, we can input the volume of each rectangular subdomain as the weight parameter for a node in the hypergraph.

We implemented a hypergraph partitioning scheme through the PaToH (Partitioning Tool for Hypergraphs) library in order to minimize core-to-core communication and equally distribute computation load across all cores (Çatalyürek and Aykanat, 2011). We pass the rectangular regions into the PaToH partitioner as vertices with weights equal to the respective subdomain volumes. The hyperedges are defined by the interfaces connecting separate subdomains. By partitioning the resulting hypergraph, the resulting core assignments will reduce overall computation cost, since neighboring vertices will tend to be assigned to the same core and they will not require communication. This also allows for a balanced load distribution by assigning each core roughly equal volume, which is linearly related to computation time. We use the following connectivity metric to determine the cost of communication across boundaries:

$$C(\Pi) = \sum_{n \in N_E} w_n(\lambda_n - 1), \tag{3.2}$$

where $C(\Pi)$ is the cost of a particular partitioning Π, N_E is the set of cut hyperedges, w_n is the weight of hyperedge n, and λ_n is the connectivity of the hyperedge. This metric is particularly useful because it prioritizes hyperedges that connect many vertices and is such directly proportional to the computation cost.

3.3.2 Load balancing and numerical stability

The splitting algorithm for load balancing introduced in earlier minimizes the number of extra interfaces created by splitting in some cases. The larger of the resulting subdomains is exactly below the maximum volume threshold of the splitting algorithm.

However, on a cluster with a higher number of cores and where the volume threshold can be relatively small, this splitting algorithm can introduce a series of very small and thin rectangles. Small and thin rectangles can create numerical instability during interface resolution. These errors are caused by the interaction between multiple overlapping interfaces.

Care must be taken to address these numerical problems. In this particular case, it is more advantageous to have wider rectangular subdomains that may introduce more interface area rather than degenerate rectangles which can result in numerical issues. A revised approach favors well-formed subdomains that are more cuboidal in shape rather than long and thin rectangles by choosing a subdivision on the minimal axis.

3.3.3 Interface and PML computation

An important step of the ARD and parallel ARD methods is the initialization of interfaces and PML regions. This generally involves determining subdomain adjacency and which grid cells are in an interface. This computation is linear with respect to the number of grid cells in the scene – that is, it scales linearly with the volume of the scene and with the 4th power of frequency. At higher frequencies, the interface and PML setup can consume several hours of time before any actual simulation steps are run.

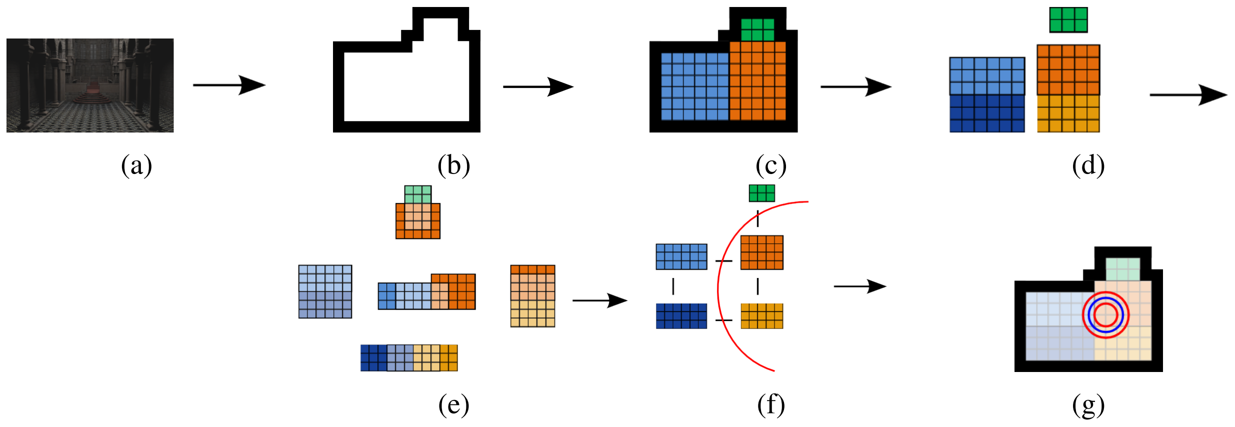


Figure 3.3: The MPARD pipeline. The input geometry (a) is voxelized in the first step (b). The rectangular decomposition step divides the domain into multiple non-overlapping subdomains (c). The splitting step then splits these subdomains when they are greater than the volume threshold Q (d). These partitions are then processed by the interface initialization stage which computes interface metadata (e). The final preprocessing stage allocates subdomains to nodes using the hypergraph partitioning (f). Finally, the simulation is run (g).

In order to reduce the running time of the simulation, MPARD introduces a new preprocessing step in which the interfaces can be initialized offline. This preprocessing step occurs after any splitting and load balancing, after the decomposition for the scene is final.

Additionally, this extra preprocessing step allows for further memory optimization in MPARD. With the kind of global metadata computed in the preprocessing, each core only needs to load the exact interfaces and PML regions it needs for its computations.

3.3.4 MPARD Pipeline

MPARD introduces new preprocessing stages and a modified simulation stage. The pipeline overview can be found in Figure 3.3. The scene input (Figure 3.3(a)) is first voxelized (Figure 3.3(b)). Next, the rectangular decomposition fills the available air space with rectangular subdomains using a greedy algorithm (Figure 3.3(c)). Next, our new splitting algorithm that avoids degenerate subdomains splits rectangular regions that are larger than the volume threshold (Figure 3.3(d)). After interface regions are calculated (Figure 3.3(e)), we assign subdomains to cores using hypergraph partitioning (f). Finally, our solver reads the preprocessing data and runs the simulation for a set number of time steps.

3.3.4.1 Voxelization

The voxelization stage takes in a triangle mesh representing the environment in which we want to compute the sound propagation. Because MPARD targets large and high frequency scenes that may consume a large amount of memory, we use a CPU method for voxelization. We implement an *accurate* and *minimal* method (meaning voxels should fully cover the geometry but not more than necessary) as introduced by Huang et al. (Huang et al., 1998).

The spatial discretization for the voxelization is determined by the minimum simulated wavelength and the required number of spatial samples per wavelength, which is typically between 2 and 4 (Raghuvanshi et al., 2009b). Therefore, the voxelization only needs to be run once per desired maximum frequency.

3.3.4.2 Decomposition

The decomposition stage then reads the voxel field and determines the location of the different cuboidal subdomains. The process is a greedy approach, attempting to expand each rectangular subdomain into as large a volume as possible under the constraints of the wall voxels.

At very high frequencies, such as 10 kHz, this process can take several days to complete but only needs to run once for a given voxel input.

3.3.4.3 Core allocation and subdomain splitting

The next stage of the preprocessing is the core allocation and subdomain splitting stage. In addition to a decomposition computed in the previous stage, this step also requires the number of cores the solver will run on. This stage uses the input values to compute a hypergraph partitioning for the decomposition in addition to splitting any rectangular regions that have volumes greater than the volume threshold Q.

The core allocation stage then determines the assignment of subdomains to cores by using the hypergraph partitioning assignment or alternatively a simple bin-packing algorithm. This load balancing step ensures that each core has a roughly equal amount of work to complete during the acoustic simulation.

The allocation and splitting stage must be run for each decomposition for each desired core configuration.

Scene Name	Volume	Frequency	Number of Triangles
Cathedral	$19\,177\,\mathrm{m}^3$	5 kHz, 10 kHz	55415
Village	$362\,987\,\mathrm{m}^3$	1.5 kHz	358

Table 3.2: Dimensions and complexity of the scenes used in our experiments. The input triangle mesh is voxelized according to the simulation frequency.

3.3.4.4 Interface and PML preprocessing

The final stage of preprocessing computes interfaces and creates PML regions from wall voxels. This stage takes as input a modified decomposition from the core allocation and splitting stage in addition to a refinement parameter r that can be used to subdivide voxels in the final acoustic simulation. This allows us to run at r times the frequency the decomposition was run at. However, this is at the expense of some accuracy where high frequency geometric features of the scene that may affect sound propagation cannot be accurately represented.

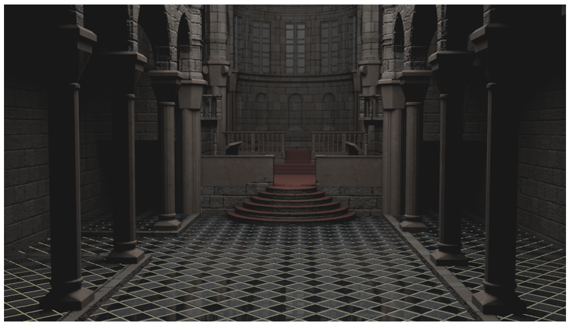
One additional caveat of the interface and PML preprocessing file is file read performance in the simulator. The interface file can be several GBs in size, and thousands of CPU cores reading the file can cause a bottleneck. As a solution, we use the file striping feature of the Lustre file system (Schwan, 2003) to increase file read performance over all cores.

The interface and PML initialization stage only needs to be run once for each core configuration.

3.4 Results and analysis

Our method was tested on two computing clusters: the large-scale Blue Waters supercomputer (Bode et al., 2012) at the University of Illinois and the UNC KillDevil cluster. Blue Waters is one of the world's leading compute clusters, with 362240 XE Bulldoze CPU cores and 1.382 PB of memory. The KillDevil cluster has 9600 CPU cores and 41 TB of memory.

Our primary experiments were performed on the Šibenik Cathedral scene and the Village scene (Figure 3.2). Both scenes provide a challenge for the underlying ARD solver. Cathedral has many curved surfaces, creating very small rectangular regions in the rectangular decomposition. Additionally, the large areas in the center of the cathedral creates very large rectangular subdomains. Furthermore, the size of the scene is around $20\,000\,\mathrm{m}^3$, making communicating the sound propagation of the scene at high frequencies with wave-based methods very challenging. For example, a 10 kHz voxelization of the cathedral has almost 4 billion voxels.


(a) Sibenik Cathedral Scene

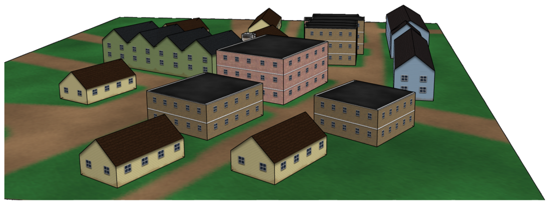
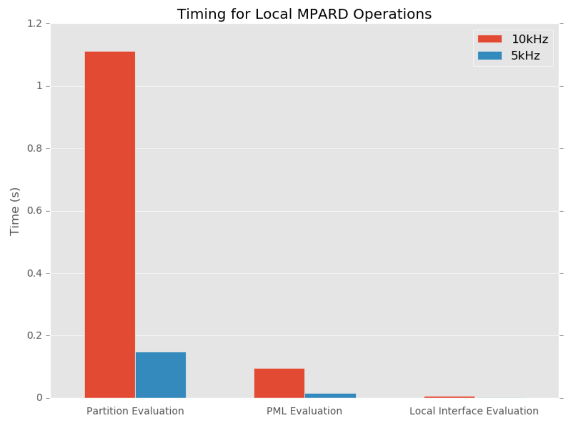
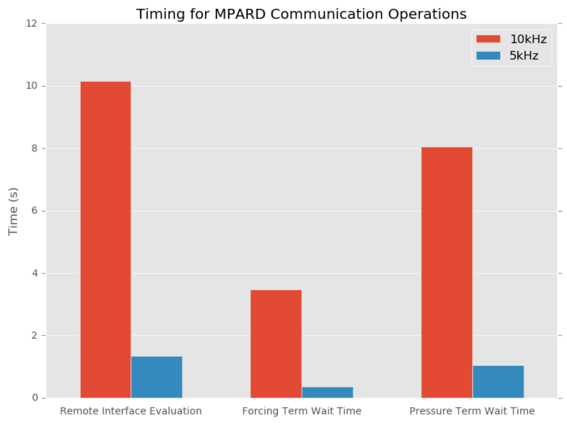

(b) Village Scene

Figure 3.4: The scenes used in our experiments.



(a) Local Timings



(b) Communication Timings

Figure 3.5: The average running time of each stage of our solver on a 10 kHz scene compared to the 5 kHz scene. These results were obtained on 1024 cores. The communication timings show the various stages of interface evaluation and communication. Remote Interface evaluation is the compute cost of the interface, while the Forcing Term Wait time is the communication cost of transferring forcing terms from the interface to the spanned subdomains, and the Pressure Term Wait time is the communication cost of transferring pressure terms to the interface from the spanned subdomains.

On the other hand, Village is mostly a large open area with a few buildings (Figure 3.4). Village is also much larger than cathedral. The size of the scene (362 000 m^3) presents many computational challenges, particularly with computing interfaces. The scene contains over 100GB of interface data compared to 40GB in the 10 kHz Cathedral scene. On the other hand, the cathedral scene contains many more partitions (176k compared to 31k), making communication more of a challenge.

We were able to run the MPARD solver on the cathedral scene up to 10 kHz. Figure 3.5 shows the average running time of each stage of our algorithm on this scene in comparison to the 5 kHz. The wait time for interface terms shows the necessity for optimizing communication at higher frequencies.

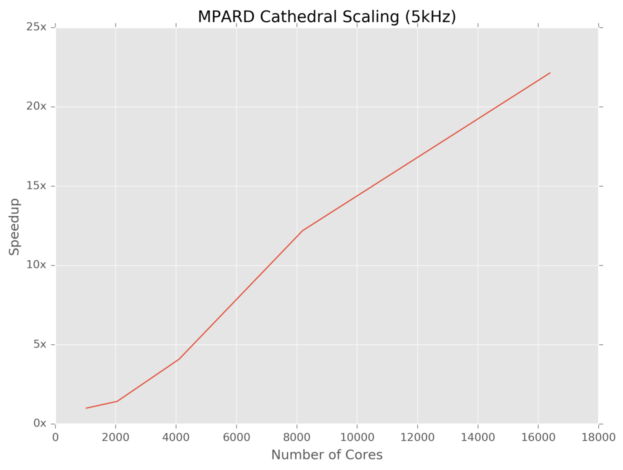


Figure 3.6: Scalability results from 1024 to 16384 cores on the 5 kHz cathedral scene. We obtain close-to-linear scaling in this result. The base speedup is 1024 on 1024 cores (assuming 1024 cores is 1024 times faster than one core) since the scene will not fit in memory on a lower number of cores.

3.4.1 Scalability results

The primary scalability experiment was done on the cathedral scene at 5 kHz. The experiment was executed on the Blue Waters supercomputer up to 16384 cores. The main purpose of this experiment is to understand the performance of MPARD at very high number of cores. Figure 3.6 shows the performance of MPARD on the cathedral scene for 1024 cores all the way up to 16384 cores. With this kind of compute power, we are able to compute each time step on Cathedral in 0.193 75 s for a 5 kHz scene.

The Village scene also shows scalability up to 8192 cores. The dominating cost in the village scene is the computation time of the interfaces (Figure 3.8). We show sublinear scaling in this case, with compute times as fast as 0.6761 s per time step for a 1.5 kHz scene.

Figure 3.8 shows a plot of how the interface area generated by our splitting scheme increases as the number of cores used in the splitting algorithm increases. We show a comparison between the Cathedral and Village scenes.

3.4.2 Load Balancing

We analyze the load balancing of our experiment on the Cathedral scene all the way up to 16384 cores. Figure 3.9 summarizes these results on varying numbers of cores using the bin-packing algorithm. Our bin-packing algorithm is very effective at reducing load imbalance in partition volume, but at a lower number

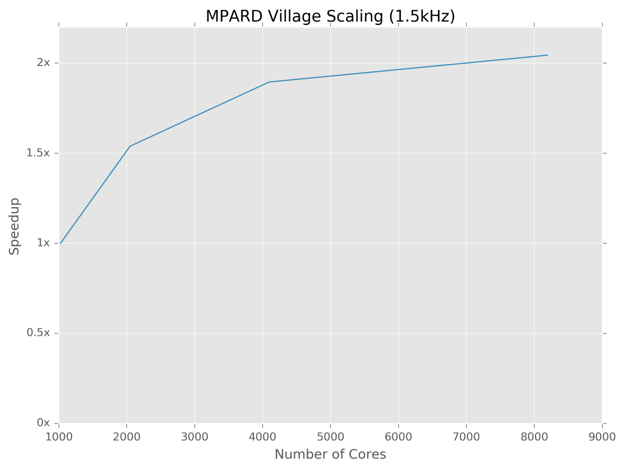
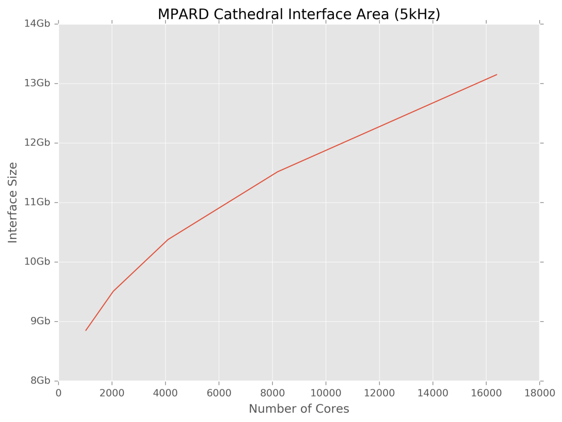


Figure 3.7: Scalability results from 1024 to 8192 cores on the 1.5 kHz village scene. We obtain sublinear scaling in this result. The base speedup is 1024 on 1024 cores since the scene will not fit in memory on a lower number of cores.



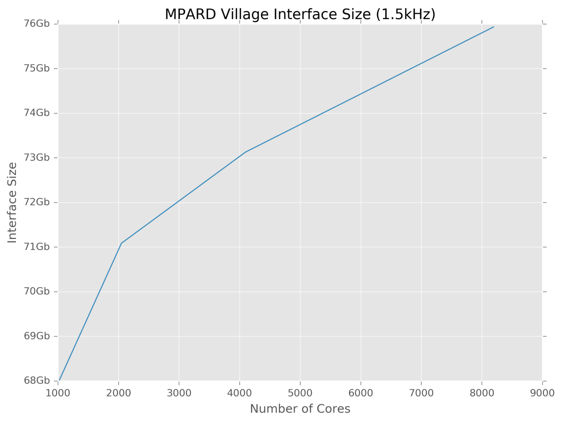


Figure 3.8: Interface area generated by our splitting scheme for the Village and Cathedral scenes. The interface area tapers off as the number of cores increases.

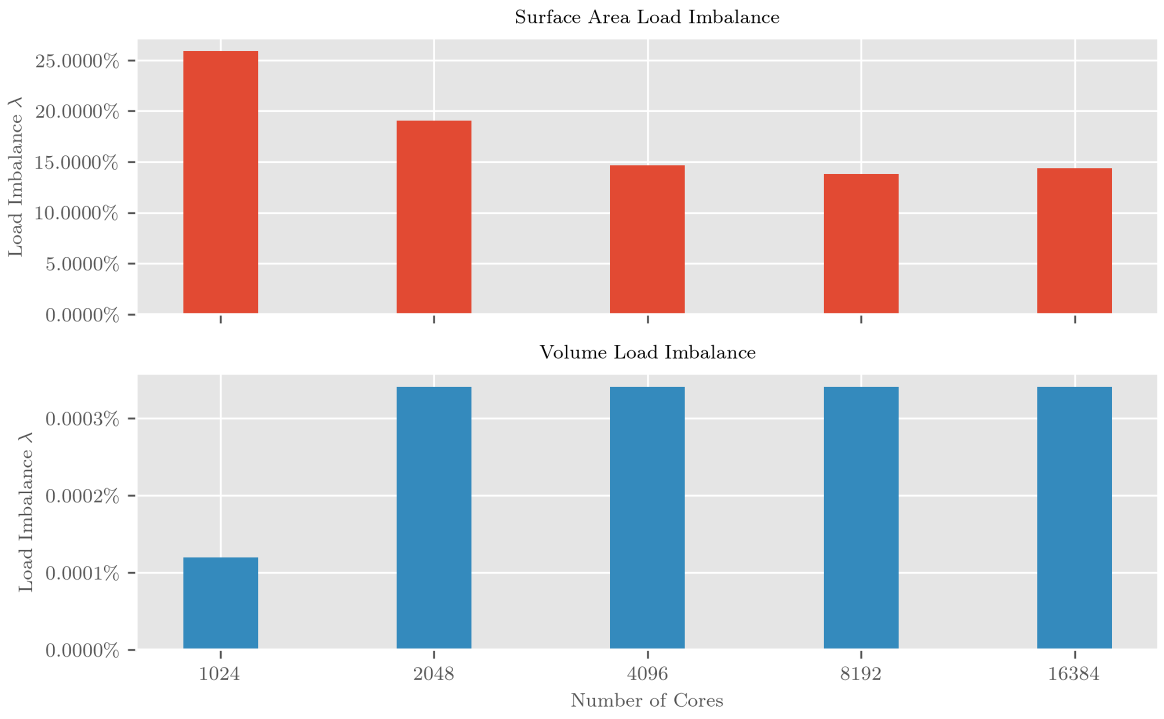


Figure 3.9: Results from our load balancing experiments. We measured the load imbalance in terms of partition work (volume) and interface work and communication (surface area) with the bin-packing algorithm for node assignment. We show less than one hundredth of a percent load imbalance in volume, but up to a 25% load imbalance in the interface area at lower numbers of cores

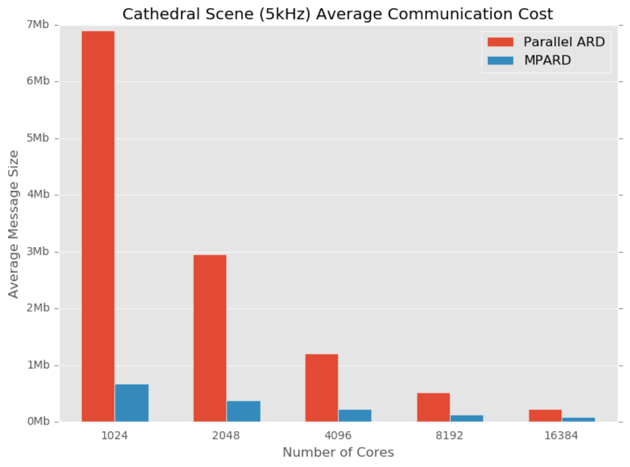


Figure 3.10: Communication cost comparison between MPARD and parallel ARD on a 5 kHz cathedral scene. In this case, the large number of interfaces in the cathedral scene means that the reduction in communication cost using MPARD is much larger.

of cores suffers from up to 25% imbalance in the interface area imbalance. The load imbalance λ was computed as follows (Pearce et al., 2012):

$$\lambda = \left(\frac{L_{max}}{\bar{L}} - 1 \right). \tag{3.3}$$

From Figure 3.6 it may seem that the algorithm performs superlinear scaling. However, this is an artifact of the less-than-optimal performance of the experiment at a lower number of cores. The reduced performance is a direct result of load imbalance at the interface area that affects both computation and communication.

3.4.3 Communication Costs

In order to examine the benefits of hypergraph partitioning, the total size of all messages averaged per core sent during a single simulation time step of both scenes was computed. The scenes had cores assigned through the bin-packing approach and the other had cores assigned through the hypergraph partitioning approach. Figure 3.10 shows the results of this experiment on a 5 kHz scene where the hypergraph partitioning approach has a 10x reduction in communication costs per core for 1024 cores and a 3x reduction per core for 16384 cores. The village scene, half as large in memory also had a reduction in communication costs (Figure 3.11).

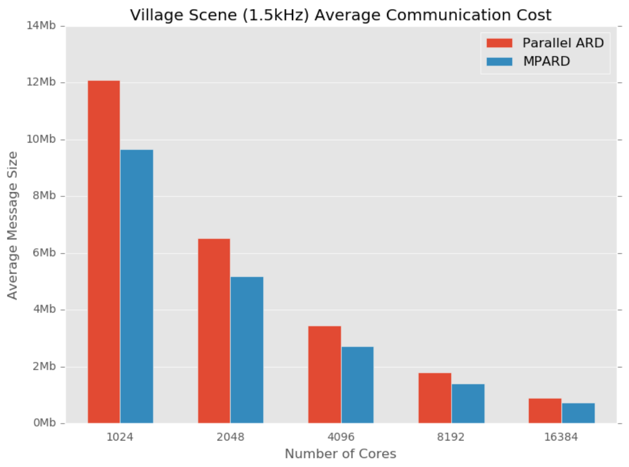
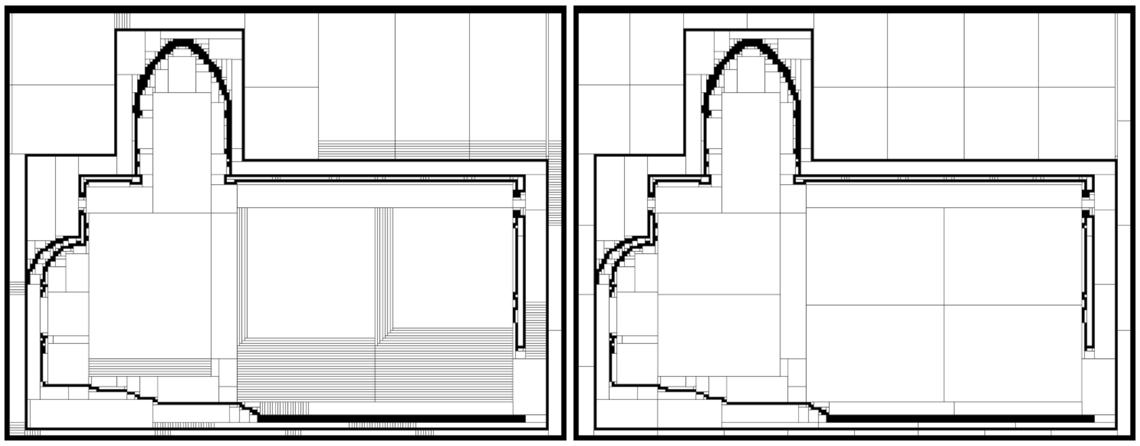


Figure 3.11: Communication cost comparison between MPARD and parallel ARD on a 1.5 kHz village scene. The village scene has fewer interfaces so there is less reduction in communication cost, but MPARD still is more efficient in this case.



(a) Thin Partitions (b) Even-volume Partitions

Figure 3.12: A comparison of the domain decomposition between a greedy splitting algorithm and the even-volume partitioning algorithm. This result was computed on a decomposition for 1024 cores on a 500 Hz scene.

3.4.4 Error and Validation

Figure 3.12 shows the difference in decomposition between two approaches to splitting on the Cathedral scene. Both scene decompositions were run on 1024 cores to show the difference between the two approaches at a high number of cores. A naïve approach creates a series of very thin rectangles along the edge of the interface while the reduced error splitting scheme creates more cuboidal subdomains.

The thin rectangular decomposition can cause numerical instability. This numerical instability is not present in MPARD simulations. Even as the number of cores increase to 16000, the error of sampled impulse responses does not exceed 5% (see Figure 3.13). The error was computed using the following metric:

$$\frac{\sum\limits_{i=1}^{n} \left(p_i^{'}(\vec{x}) - p_i(\vec{x})\right)^2}{\sum\limits_{i=1}^{n} p_i^2}, \tag{3.4}$$

where p_i is the reference pressure computed in Equation 1.1 at listener position $\vec{x}$ without partition splitting, $p_i^{'}$ is the pressure computed with partition splitting, and n is the total number of time steps.

A comparison in Figure 3.14 between a reference single-threaded impulse response and the impulse response calculated on a 16384 core run shows how various features of the impulse responses match. Figure 3.15 shows how the error of the full pressure field varies over 2000 time steps, until most of the sound dissipates. For the full-field error, we used the mean-squared error metric for each time-step:

$$\frac{1}{m} \sum_{j=1}^{m} \left(p_j^{'} - p_j\right)^2 \tag{3.5}$$

where m is number of grid points in the volume, p_j is the reference pressure at grid index j, and $p_j^{'}$ is the computed pressure at grid index j.

This shows MPARD's stability and accuracy over the whole course of a full simulation. For these experiments, we followed a similar experimental setup as previous work that have validated the serial ARD technique (Mehra et al., 2014). This work focused on the village scene, a digital reproduction of a village in which the propagation of various sound sources was measured. The measured impulse responses at various listener positions were compared to impulse responses computed by the ARD method. The comparison was done by comparing average dB and comparing spectrograms with dynamic time warping to account for slight

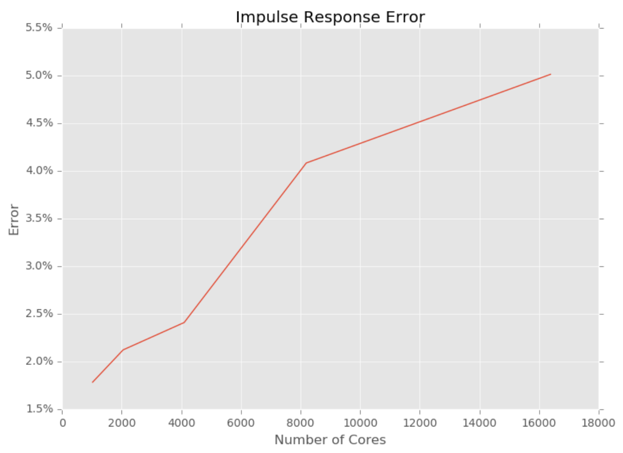


Figure 3.13: Error of an impulse response taken around 10 m away from the source on the Village scene using the MPARD method for a single band impulse at 225 Hz. The impulse response is calculated for 1024 cores all the way to 16384 cores. Subdomain splitting increases as the number of cores increases, so the error also increases. Even for an extreme number of cores (16384), the error is limited to around 5%.

differences in the arrival times of different wave forms. We compared our results directly to the serial ARD implementation of this scene, and so did not have to use dynamic time warping techniques.

3.4.5 Comparisons and benefits

In comparison to standard methods like FDTD, ARD does not require as fine of a computational grid. Traditional FDTD methods generally require a spatial discretization that is $1/10$ the minimum wavelength (although the low-dispersion methods listed in Section 3.1.3 aim to lower this requirement). In comparison ARD can use a much coarser grid size, around $1/2.6$ times the minimum wavelength (Raghuvanshi et al., 2009b; Mehra et al., 2012). This means that ARD can inherently be $24-50$ times more memory efficient than FDTD and up to $75-100$ times faster.

Comparison with GPU ARD: MPARD provides advantages over previous GPU parallel ARD approaches. Large scenes at high frequencies require terabytes of memory that are easily available on large compute clusters but are not available on GPUs (Mehra et al., 2012). Secondly, MPARD scales over a much larger number of cores while GPU ARD is limited to a single machine and a shared memory architecture.

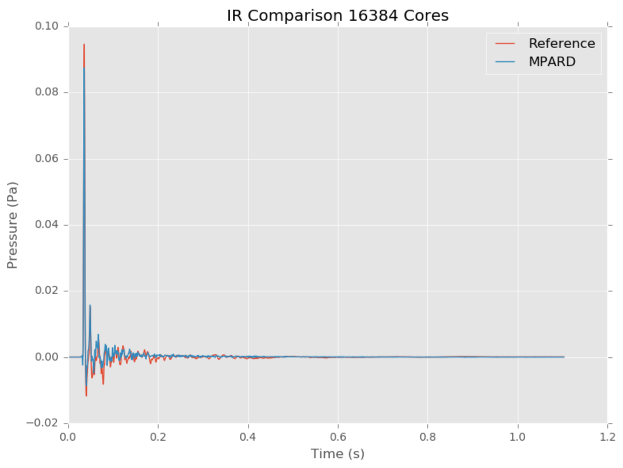


Figure 3.14: Comparison of impulse responses 10 m away from a 225 Hz source. The reference impulse response uses the serial implementation of ARD (Mehra et al., 2014), while the simulated impulse response uses a subdomain partitioning for 16384 cores. We show that the impulse response matches closely with the serial implementation validated in Mehra et al.

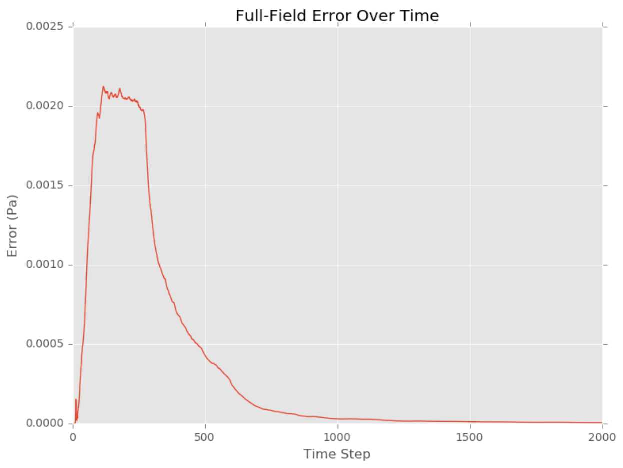


Figure 3.15: Error of the full field MPARD simulation over time on a 16384 core simulation excited by a single source at 225 Hz. The error is introduced by spurious reflections and dispersion, but does not accumulate as time increases. The error was computed using the mean-squared error for each time step.

3.5 Conclusion and future work

MPARD is a massively parallel approach showing scalability up to 16000 cores and is capable of calculating pressure fields for large scenes with frequencies up to 10 kHz. MPARD introduces several improvements over parallel ARD, reducing communication cost by assigning cores with a hypergraph partitioning scheme, providing better numerical stability for higher numbers of cores, and implementing an extended preprocessing pipeline that reduces the usage of valuable cluster resources for redundant calculations.

In the future, although we have shown the ability to compute very high frequency scenes, we would like to test the method on more scenes including even larger architectural and outdoor scenes at high frequencies. There has already been some work on hybrid sound propagation, using geometric methods for higher frequencies and ARD for lower frequencies (Yeh et al., 2013), so we would like to expand on this for large outdoor scenes.

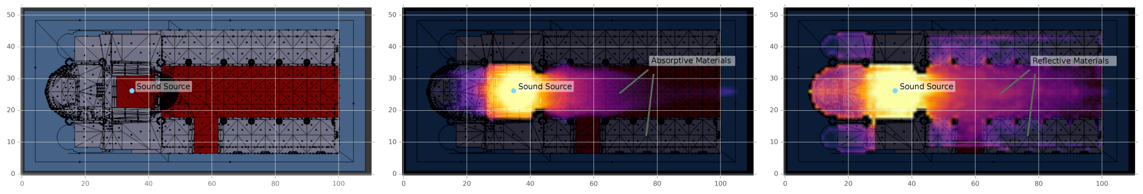
Scaling to large outdoor scenes such as cities or towns poses many technical challenges. Although we reduce communication costs between cores with a hypergraph partitioning technique, a hybrid shared/distributed memory approach may reduce communication costs even further. Additionally, we could take advantage of the computational power of GPUs for a fully heterogenous computing algorithm.

CHAPTER 4: Efficient Wave-Based Acoustic Material Design Optimization

Architectural and engineering design of structures often requires the incorporation of various design goals, such as functionality, reliability, operation, and aesthetics. Moreover, the design of these structures is often governed by specific constraints, such as performance, cost, maintainability, testability, and so on. Of particular importance is the interaction between sound waves and the structure. These sound waves are typically produced by human speech and noise, machines, musical performances, etc. The acoustic characteristics of a space can have an effect on the perception of that space, human communication, and behavior. These characteristics are usually determined by the shape, topology, structure and surface materials, and objects inside the acoustic space. Figure 4.1 shows an example of how the sound Strength, or sense of fullness of the sound, is affected by the materials inside the space.

The acoustic characteristics of architectural models are measured in terms of sound clarity, strength, delay, reverberation, etc. Different architectural models impose varying requirements on these acoustic characteristics. For example, the premium seats in concert halls often require a sound clarity measure (C80) of between -2dB and 4dB. Other constraints or standards are imposed due to health or environmental factors. The WHO recommends that the equivalent continuous noise level from the environment in hospitals during the night should not exceed 30dB (Birgitta et al., 1999). The implementation of noise minimization procedures in hospitals can result in a significant drop in medical errors (Mazer, 2005). Studies have also shown that poor acoustics can have a negative effect on classrooms (Bronzaft, 2012).

Recent trends in computer-aided design for acoustic design have focused on simulation technologies for prototyping architectural and engineering structures. For example, acoustic models are used in the design of airplanes to predict the noise caused by engine vibration and the propagation of acoustic waves throughout the aircraft cabin (Johnescu, 2003). Additionally, manufacturers use large noise engineering laboratories for measuring airframe and aircraft noise (Turnage, 2002). Urban habitation designers (Tsou et al., 2003) and automobile manufacturers (Genuit and Bray, 2001) have also used acoustic simulation for prototyping designs. However, current acoustic simulation tools are limited in their accuracy and domain capabilities. Often they do not provide reliable solutions. A direct consequence of this lack of reliability is that acoustic



(a) Acoustic Scene and Materials (b) High Absorption (c) Low Absorption

Figure 4.1: An example of how the materials in a building can affect the acoustic properties of that building. In this figure the materials are each assigned a different color in (a). Parts (b) and (c) show the acoustic Strength (G), or feeling of loudness and fullness of sound, at each point in the scene for different material configurations. The brighter color corresponds to a higher Strength. Part (b) shows the scene with almost fully absorptive materials, where sound waves are mostly absorbed by the walls and floor. Note how overall, high acoustic Strength values are limited to those areas with a direct line of sight to the sound source. Part (c) shows a low absorption scene, where sound waves are reflected and keep most of their energy. In this case, listeners can get a better feeling of fullness and loudness without being directly in front of the sound source.

design is frequently performed by human designers with a limited set of acoustic simulation tools. It is also common for acoustic engineers to build physical prototypes to validate the acoustic characteristics of their design. This can cause long design cycles or even non-optimal acoustic designs (Bassuet et al., 2014; Monks et al., 2000; Mendez, 2014).

This chapter will discuss our approaches to acoustic material optimization, using wave-based solvers such as the ARD method. The work in this chapter was originally published in (Morales and Manocha, 2016).

4.1 Prior Work

Computational acoustics is an area of active research in engineering design and scientific computing, and is also studied in seismology, geophysics, and meteorology. In this section, we limit ourselves to computational acoustic methods for large architectural models.

4.1.1 Simulation and Computer Aided Design

Extensive research and software development has focused on vibration analysis, interior and exterior acoustic radiation computation, vibro-acoustics, and aero-acoustic modeling (Keane and Nair, 2005; Doicu, 2000). Much work has focused on modelling of Noise, Vibration, and Harshness (NVH) measurements for car interiors (Genuit, 2004). Additionally, many commercial tools are available for acoustic analysis of objects, structures, or small spaces. Often, these tools are not sufficiently accurate or applicable to large acoustic spaces prevalent in architectural design such as auditoriums, concert halls, or outdoor environments where the volume may exceed 10 000 m^3 to 100 000 m^3.

Analysis of the acoustic characteristics of architectural spaces is often studied in the context of room acoustics (Kuttruff, 2009). Work in this field has been done by Sabine dating back to the early 1900s. This work was conducted through ray-based acoustics and Sabine and Eyring's reverberation time formula for rectangular rooms. More recently, geometric acoustic techniques based on ray-tracing (Krokstad et al., 1968; Vorländer, 1989; Schissler et al., 2014b) have become prevalent in the evaluation of indoor acoustic designs such as concert halls, theatres, and auditoriums. Geometric acoustic methods are used in several commercial packages but suffer accuracy issues because of the underlying assumption that sound propagates as rays rather than as waves. Therefore, wave effects of sound that are prevalent at lower frequencies, such as diffraction and scattering, are often neglected.

Wave-based methods, on the other hand, directly solve the acoustic wave equation and do not suffer from these accuracy issues. Numerical solvers for wave-based methods include finite difference methods (Taflove and Hagness, 2005), finite element methods (Thompson, 2006), and boundary element methods (Gumerov and Duraiswami, 2009). However, the computational and memory requirements for these methods are much higher and, as a result, wave-based techniques are usually limited in practice to small spaces (less than 1000 m^3, for example) and low frequencies (less than 2 kHz). Recent advances in wave-based methods have reduced the computational and memory complexity of these algorithms. These works include low dispersion methods such as the Adaptive Rectangular Decomposition (ARD) method (Raghuvanshi et al., 2009b; Mehra et al., 2012; Morales et al., 2015b) and the equivalent source method (Mehra et al., 2013).

4.1.2 Acoustic Optimization

A common problem in mechanical and architectural design is the automatic optimization of a set of parameters on the computational domain. This field, part of Multidisciplinary Design Optimization (MDO), is widely used in many areas including the aerodynamic optimization of wings and entire aircraft, architectural features such as bridges and buildings, railway cars, microscopes, automobiles, turbines, and ships (Martins and Lambe, 2013). Many commercial design optimization tools are available for building information modeling (such as Autodesk Revit building design software tools) and are frequently used for lighting analysis, structure analysis, energy analysis, and so on. Often these tools can take advantage of the computational capabilities of large distributed clusters or large-scale cloud computing. However, the state of the art in acoustic design for large architectural spaces is still at its infancy.

The acoustic optimization problem is a subset of MDO, and is useful for designing acoustic spaces or engineering structures according to certain target acoustic metrics. These metrics can range in complexity from sound intensity minimization to sound clarity or reverb time to binaural acoustic evaluations (ISO, 2009). Much of the prior work in acoustic optimization has focused on Noise, Vibration, and Harshness measurements (NVH). A more limited set of methods has focused on space optimization, such as methods targeting architectural acoustics. We introduce two terms to describe these models. The first term, *object models*, refers to sound and vibration traveling through the objects. *Spatial models*, on the other hand, refers to sound waves propagating through the atmosphere in the space within a structure.

4.1.2.1 Object Models

Object models of acoustic optimization deal with the analysis of vibrations or sound propagation through individual objects rather than through acoustic spaces. Most of these models have been used for the design of automobiles, engines, and architectural support structures such as beams. Maressa et al. (Maressa et al., 2010) introduce a method for NVH measurements for a car interior that is represented by a finite element mesh. This method explicitly studies the relationship between the structure of the object and the acoustics in the object. Other techniques that study the acoustic-structure interaction are based on a unified approach (Yoon et al., 2007), which uses a mixed formulation to represent both the acoustic propagation and the elastic displacement of the structure. Nandy et al. (Nandy and Jog, 2012) bypass acoustic simulation entirely in the optimization process. Du, Song, and Olhoff (Du et al., 2011) present a method of acoustic-structure interaction at a finite boundary around the vibrating object. Shu et al. (Shu et al., 2014) use a level set based topology optimization to minimize sound resulting from vibrations in an outer structure. The topology optimization allows certain structures, such as beams, to be generated in order to reduce the sound.

4.1.2.2 Spatial Models

Our goal is to optimize the acoustic material properties of architectural models to satisfy some constraints on the acoustic characteristics. This is an example of *spatial acoustic optimization* and the driving application is acoustic design and optimization of large architectural models. In this context, there are three main subproblems. The first, material optimization, involves the modification of the materials of the scene as parameters of the optimization problem. The second, shape optimization, deals with the modification of dimensions of certain aspects of the scene, but not the actual topology. The third and last one deals with topology optimization.

Material optimization problems limit the number of parameters being optimized to a discrete set of materials for each surface or to an arrangement of material placements. Work on this has been done by Saksela et al. (Saksela et al., 2015) and Monks (Monks et al., 2000) using geometric acoustic solvers. Shape optimization, on the other hand, reduces the optimization parameters to a smaller set of dimensions or measurements. For example, Robinson et al. (Robinson et al., 2014) parameterize the shape of balconies in a concert hall in order to determine the best shape for sound clarity and strength in all areas in the concert hall. Other shape optimization approaches (Fuchi and Gea, 2013) minimize the acoustic pressure in a specific section of a room by modifying a series of columns at the top of the room. Floody and Venegas (Floodya and Venegasb, 2007) modify the dimensions of a rectangular room in order to reduce resonance frequencies. In the area of topology optimization, Dühring et al. (Dühring et al., 2008) minimize the noise in a target area in an output domain by modifying the topology of the design domain. The modification is performed by discretizing the design domain and assigning each element a value between 0 and 1, where 0 corresponds to air and 1 represents an aluminum material. They use an adjoint sensitivity approach and use the Method of Moving Asymptotes to solve the optimization problem.

Many of these techniques are classified as either continuous or discrete. Continuous optimization approaches often use gradient-based approaches but in general encompass any technique that yields continuous values. These include Saksela et al (Saksela et al., 2015), who use a linear least-squares approach. Other continuous approaches include Dühring's technique, which uses the adjoint method for computing the continuous gradient. Discrete optimization techniques, on the other hand, select from a set of non-continuous configurations. The challenge in these problems often lies in their large search space, which makes the problems combinatorial in nature. The primary example of this approach is Audioptimization (Monks et al., 2000), which uses Simulated Annealing (SA) in addition to a discretized steepest-descent algorithm to search through a discretized set of geometric transformations and materials.

4.2 Acoustic Metrics

Several useful measures can be derived from the impulse response $p(\vec{x})$. These give an indication to the acoustic engineer of certain aspects of the sound that is produced in the concert hall or structure that is being designed. Many of the metrics are specified in the ISO standard 3882 (ISO, 2009). Our algorithm focuses on acoustic Strength (G), Clarity (C80), and Reverb Time (RT60), although it can be extended with more metrics including binaural metrics. The function $H_k(\Omega)$ represents an arbitrary acoustic metric as computed by ARD, where k is the index of the metric and Ω is the simulation domain.

4.2.1 Strength and Clarity

The strength and clarity metrics are computed using the energy of various portions of the impulse response. Clarity measures the ratio of the energy of the first t_e ms of the impulse response (we use $t_e = 80$) to the energy of the remainder of the impulse response. For clarity,

$$H_k(\Omega) = C_{t_e} = 10 \log \frac{\sum_{i=0}^{n-1} p_i^2}{\sum_{i=n}^{m-1} p_i^2} \text{ dB}, \tag{4.1}$$

where $t_e = n\Delta t$, p_i is the pressure at time step i, and m is the maximum number of simulation time steps determined by the total energy decay and maximum simulation frequency.

Strength, on the other hand, measures the loudness and sound distribution of a concert hall:

$$H_k(\Omega) = G = 10 \log \frac{\sum_{i=0}^{m-1} p_i^2}{\sum_{i=0}^{m-1} p_i'^2} \text{ dB}, \tag{4.2}$$

where in this case p_i' is the pressure at the reference IR in a free-field condition. The free-field environment can be evaluated by simulating an enclosed space with perfectly absorbing boundary conditions. This result is computed then cached as a reference value in determining acoustic strength.

4.2.2 Reverberation Time

This section explains in some detail the implementation of reverb time calculation for the ARD wave solver. ARD suffers from very low amplitude spurious reflections and numerical dispersion at the interfaces of rectangular subdomains. These inaccuracies take the characteristics of a noise floor that is proportional to the amplitude of the acoustic signal.

Primarily, the reverse-time integration posited by Schroeder (Schroeder, 1965) is used to smooth the impulse response and ensure that it is monotonically decreasing. The range from −5 dB to −35 dB is used to compute a least-squared regression. The intercept of this line with the −60 dB line yields the reverb time in seconds.

Given m time samples in the discretization of the impulse response, define the energy decay curve of the impulse response R_i with $i = 0, 1, 2, \ldots, m - i$:

$$R_i = 10 \log \frac{\sum\limits_{j=i}^{m-1} p_j^2}{\sum\limits_{j=0}^{m-1} p_j^2} \text{ dB}, \tag{4.3}$$

where p_j is the pressure at time $j\Delta t$. A linear least-squares regression on R yields a fit $y = \alpha + \beta t$. Simulated impulse responses with ARD have a nominal noise floor at 21 dB as a result of absorption errors (Raghuvanshi et al., 2009b). The decay range used for linear regression should be 10 dB above the noise floor. The resulting metric is then:

$$H_k(\Omega) = RT60 = -\frac{\alpha}{\beta} \tag{4.4}$$

4.3 Gradient-Based Acoustic Optimization and Sensitivity Analysis

We present a novel approach for optimizing the acoustic material properties of CAD models for the purpose of simulation-based acoustic engineering design. Our approach is designed for large architectural models and is based on accurately computing the acoustic pressure field as a function of the material properties.

Our formulation is based on using the ARD wave-based solver (see Chapter 2). We present a novel and efficient sensitivity analysis of the ARD wave solver with respect to the input acoustic materials. The resulting sensitivities are used to drive a gradient-based iterative algorithm that can automatically optimize the material properties to satisfy the acoustic design criteria of the space. Overall, the three novel components of our work include:

- Wave-based sensitivity analysis of the acoustic pressure field for design optimization.
- A fast acoustic design optimization algorithm based on the ARD acoustic wave solver.
- A multi-objective acoustic material optimization that can simultaneously optimize for various acoustic properties including strength and clarity.

We show the results of optimizations using our algorithm on large scale 3D scenes, including CAD models of well-known architectural models. We automatically compute the material properties to give tight bounds on the acoustic characteristics of the resulting models. The overall approach is general purpose and takes many tens of minutes on large architectural models of about 20 000 m^3 volume on a desktop computer. To the best of our knowledge, this is the first method able to perform accurate sensitivity analysis for acoustic material design optimization of large CAD models.

4.3.1 Acoustic Optimization

Our goal is to design an optimizer that can target multiple acoustic design parameters. For example, a designer might want a specific Strength and a specific Clarity value for a concert hall. We use a linear weighted sum optimization function, in which different target characteristics are weighted according to their importance. This allows a designer to put more emphasis on some acoustic characteristics over others. Therefore, our formulation of the optimization problem is expressed as:

$$\min \sum_{i=1}^{n} w_i \parallel f_i(\Omega) - Z_i \parallel , \tag{4.5}$$

where n is the number of acoustic properties, $w_i \geq 0$ is the weighting for the acoustic property i, $f_i()$ is the calculation of the acoustic property on the domain Ω, and Z_i is the target value for the acoustic property i.

The function $f(\Omega)$ is calculated by using an acoustic solver. This solver could be wave-based or geometric, but in our case we use a wave-based simulator. The input to the function is the geometric representation and layout of the scene and the set of acoustic materials in the scene (Ω). The output is a vector of the characteristic acoustic properties mentioned earlier. We use this linear combination in preference to a weightless norm because we wanted designer control over the relative importance of different acoustic characteristics. For example, in a concert hall, the parameter C80 is probably the most important along with

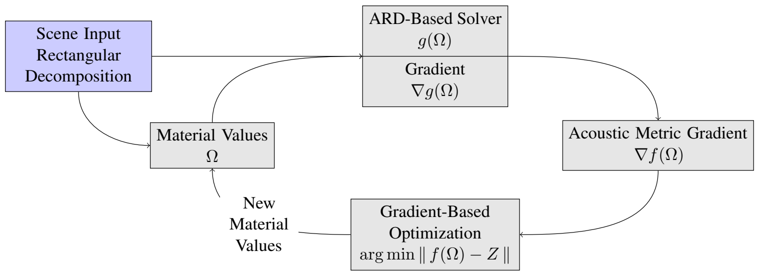


Figure 4.2: Our optimization pipeline, which computes the full derivative at the same time as it computes the pressure field. The scene input determines the initial material values and geometry. At each optimization step, a new set of material absorption values is computed.

RT60 (Kuttruff, 2009). Moreover, a linear-weighted objective function has been shown to be sufficient for combining various acoustic metrics (Monks et al., 2000).

Our optimization pipeline (Figure 4.2) computes the full derivative of the optimization problem. This is used to guide a gradient-based method such as gradient descent, conjugate gradient methods, or quasi-Newtonian methods. As detailed in Section 4.3.4, the full derivative of the acoustic pressure field is actually computed along with the ARD based acoustic pressure solver.

For the optimization method, we chose to use a standard BFGS algorithm. Using this method is advantageous because of the non-smooth nature of the optimization function. BFGS is a gradient-based optimization method that can deal well with non-smooth functions (Lewis and Overton, 2009). Additionally, BFGS is a subset of quasi-Newtonian methods that are designed to solve multidimensional optimization problems (Dennis and Moré, 1977). Because we are interested in architectural models that have multiple materials that we would like to optimize, this is particularly useful.

The implementation we use is the Dlib library which provides both BFGS and LBFGS and algorithms. The optimizer is driven by the acoustic solver ARD (see Chapter 2) and derivatives produced by Automatic Differentiation on the solver code (see Section 4.3.2). We use a delta stop condition to detect minima — when there is little change in value the optimizer finishes.

Part of the input of the solver is a set of *material segments*. These are regions of the architectural model that are assigned the same material. For example, if an acoustic engineer wanted to determine the optimal absorption of the carpet for obtaining a certain acoustic strength value, she could mark the floor as a material segment. These segments can be specified in any standard modelling program.

For optimization, material values are directly driven by the optimization algorithm. Some materials must remain constant, however. These materials are either specified by the user to not be optimized (as would be in the case where parts of the architecture must be made of a certain material) or are part of the free-field boundary condition of the scene. ARD implements a perfectly absorbing boundary condition at the edges of the scene to simulate open areas. Therefore, the solver and optimizer work well with both indoor and outdoor scenes, including hybrid scenes with aspects of both.

The vector of these material segments is initialized with a specified or random starting value per material segment and then modified by the optimizer until convergence. The random starting value allows multiple processes of the optimizer to run in parallel in order to avoid local minima.

4.3.2 Sensitivity Analysis

As part of our optimization algorithm, we need to design a method to efficiently compute the sensitivity of the ARD solver with respect to the input material properties. Our goal in terms of sensitivity analysis is to determine how each acoustic characteristic of the scene changes when these inputs are changed. For example, if we have a scene with the set of material properties Ω, the sensitivity is the full derivative of the function f (Equation 4.3.1). We assume that the input parameters to this equation (i.e. the acoustic materials) do not occur in any other parameter, so the full derivative is simply the gradient of f. Therefore we have:

$$f^{(k)'} = \nabla_{\Omega_k} f, \tag{4.6}$$

where $f^{(k)'}$ is the full derivative of f when one of the materials is modified. The calculation of this derivative can be performed using finite differences, but this approach has inaccuracies that can lead to issues in the optimization problem (Sobieszczanski-Sobieski, 1990) or may take a greater number of steps to converge to the target value.

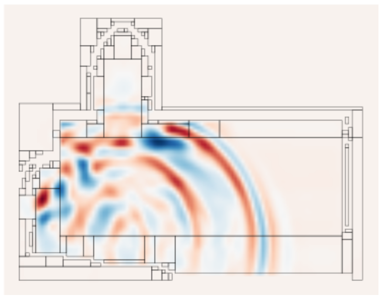


Figure 4.3: Domain decomposition of a cathedral using the ARD method.

The computation of this full derivative can be non-trivial. ARD is a domain decomposition method in which each subdomain Ω_i is tightly coupled with a neighboring subdomain Ω_j over the interface Γ_{ij}. In ARD, the analytic solution to the wave equation over each subdomain is used for the pressure computation, but the interfaces use a 6th order FDTD stencil for the computation. Figure 4.3 shows an example of this decomposition and the spatial complexity of a scene with many subdomains and interfaces.

4.3.3 Computing the Pressure Field Gradient

ARD decomposes the scene into rectangular subdomains which can solve the time-domain acoustic wave equation analytically. The analytical solution inside these domains can be described by the discretization in the modal space of Equation 1.1 (Raghuvanshi et al., 2009b):

$$\frac{\partial^2 M}{\partial t^2} + \omega^2 M = \text{iDCT}(F(t)), \tag{4.7}$$

where M is the mode coefficient, ω is the characteristic frequency, t is time, and $F(t)$ is the forcing term. This leads to the following update rule for the rectangular subdomains:

$$M^{n+1} = 2M^n \cos(\omega \Delta t) - M^{n-1} + \frac{2\widetilde{F^n}}{\omega^2}(1 - \cos(\omega \Delta t)), \tag{4.8}$$

where Δt is the time step and $\widetilde{F^n}$ is the forcing term in mode-space (Raghuvanshi et al., 2009b).

Our goal is to compute the gradient of the pressure field as the acoustic material characteristics in the scene change. Therefore, we propose a modification of the update rule that take the derivatives into account. For a single material, Ω_k, we have:

$$\begin{aligned} \frac{\partial M^{n+1}}{\partial \Omega_k} = {} & 2\frac{\partial M_i^n}{\partial \Omega_k}\cos(\omega_i \Delta t) - \frac{\partial M_i^{n-1}}{\partial \Omega_k} \\ & + \frac{2}{\omega_i^2}\frac{\partial \widetilde{F^n}}{\partial \Omega_k}(1 - \cos(\omega_i \Delta t)), \end{aligned} \tag{4.9}$$

We then apply the chain rule, and can therefore expand the three terms in this equation. $\frac{\partial M_i^n}{\partial \Omega_k}$ and $\frac{\partial M_i^{n-1}}{\partial \Omega_k}$ are similarly updated at different time steps. Of particular interest is $\frac{\partial \widetilde{F^n}}{\partial \Omega_k}$, however. $\widetilde{F^n}$ is the mode-space component of the forcing term F:

$$\widetilde{F^n} = \mathrm{DCT}(F^n). \tag{4.10}$$

The DCT is the sum of coefficients of cosines, so the derivative is simply expressed as:

$$\frac{\partial \widetilde{F^n}}{\partial \Omega_k} = \frac{\partial\, \mathrm{DCT}(F^n)}{\partial \Omega} \tag{4.11}$$

$$= \mathrm{DCT}(\frac{\partial F^n}{\partial \Omega}). \tag{4.12}$$

However, F^n may or may not be dependent on certain materials. If the forcing term is not dependent on any material, such as the forcing term originating from a sound source, then the derivative is zero. However, forcing terms originating from interfaces may be dependent on the material parameter. Finally, forcing terms originating from wall interfaces are directly and possibly indirectly dependent on the wall material. Consider the case where F^n originates from a wall interface. This is governed by the following equation:

$$F^n = \Omega_k c^2 S^n, \tag{4.13}$$

where S^{n-1} is a (2,6) FDTD stencil applied at the interface. This yields the following derivative:

$$\frac{\partial F^n}{\partial \Omega} = c^2(\Omega_k \frac{\partial S^n}{\partial \Omega_k} + S^n). \tag{4.14}$$

This formulation isolates the material term. However, $\frac{\partial S^n}{\partial \Omega_k}$ may still be dependent on the material, since the stencil covers pressure field locations that may have sound dependent on other walls or even previous steps of the same interface.

These dependencies present problems for computing the derivative. We aim to present a *general purpose* optimization method that does not depend on a specific scene configuration. However, the derivations presented in this section show several cases in which the dependency of various forcing terms or pressure values is unknown. These dependencies are intrinsically related to the geometry and material parameters of the scene, or the pressure field at previous time steps. ARD is composed of many coupled systems including an analytical solver inside the cuboid subdomains, an FDTD solver at the interface, and a PML implementation at the boundary and wall partitions. The way in which these different systems are coupled

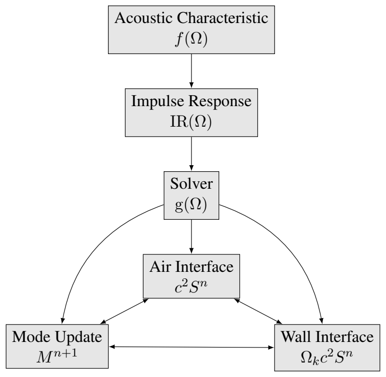


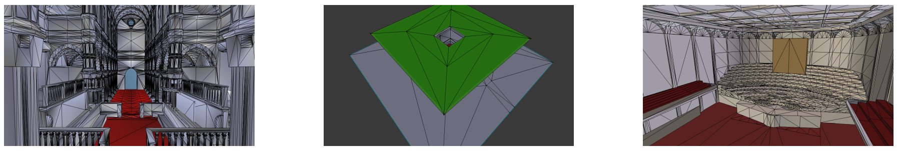
Figure 4.4: Dependencies of different subsystems of the ARD solver. These dependencies add to the complexity of the derivative calculation. Automatic Differentiation inherently deals with the problems of subsystem dependencies. The impulse response is an acoustic measurement of the room used to calculate various acoustic metrics.

together depends heavily on the scene configuration. Figure 4.4 shows the relationship and dependencies among the equations of different subsystems of ARD.

4.3.4 Automatic Differentiation (AD)

Recently a class of methods called Automatic Differentiation methods (AD) has been developed in order to find the sensitivities of various simulation and engineering codes (Bartholomew-Biggs et al., 2000). The advantage of AD methods is that they compute the analytical derivative of the code (as written). As such, they produce more accurate gradients than numerical methods such as finite differences. Automatic Differentiation works through repeated applications of the chain rule. In this way, a tree of operations and their derivatives can be combined automatically. Two methods of derivative accumulation can be used: forward accumulation and reverse accumulation that determine the order in which the chain rule is applied. Reverse accumulation can allow an AD technique to easily compute a large number of derivatives.

These properties make AD an ideal candidate for a general purpose solver like ARD. AD techniques can deal with the inherent complexity of the system, as dependency information is retained with each operation and application of the chain rule. For example, Equation 4.11 can either use a forcing term derivative from a wall interface (Equation 4.14) or an air interface. In Automatic Differentiation, the computation of these derivatives is performed at the same time that the pressure field is computed. If the scene configuration or the rectangular decomposition is changed, the application of AD will yield the correct derivative for that configuration.



(a) Cathedral (b) Twilight (c) Concert Hall

Figure 4.5: Geometry and material segmentation of the various benchmark scenes. These are specified by the designer. Our algorithm is general purpose and allows arbitrary flexibility in the assignment of these materials. In this figure, different acoustic materials are assigned different colors.

Additionally, our AD-based algorithm takes advantage of some of the efficiencies of ARD. Since ARD is a low-dispersion method, we can use a coarser discretization of the spatial domain as compared to other methods such as FDTD. This means that the update equation (Equation 4.8) needs to be evaluated at fewer points. As a result, the derivative update (Equation 4.9) can be computed with similar efficiency. This is especially important considering that computing the derivative essentially multiplies the memory requirements of the system by the number of components in the gradient. In our case, the number of components is the number of materials in the scene. The number of components in the gradient has a similar effect on computation, where the time cost of evaluating the gradient is linearly proportional to the number of components in the gradient.

For computing the sensitivity of ARD, we use an implementation of the Sacado AD library developed by Sandia National Laboratories as part of the open source Trilinos scientific computing package (Phipps and Pawlowski, 2012). Sacado is used in many scientific computing applications for calculating the sensitivity of various simulation codes. We chose it because of the inherent efficiency of the library – it uses expression templates and operator overloading in C++ to simplify expressions at compile time.

Because Sacado uses operator overloading, we can compute the derivative of the impulse response at the same time we compute the impulse response itself. As a result of the chain rule, further post-processing to derive the acoustic characteristics of the scene (strength, clarity, etc) will also contain the correct derivatives. Furthermore, we only have to evaluate this once per optimization step. This favorably compares to finite difference techniques, where we would need two evaluations of the simulation and acoustic characteristics to compute the derivative.

Scene	Volume	Num. Triangles	Num. Materials	Iteration Time
Cathedral	20 686.03 m^3	55665	2	62.0645 s
Twilight	26 759.28 m^3	270	3	85.4107 s
Concert Hall	34 510.03 m^3	5532	5	138.113 s

Table 4.1: Summary of the various benchmark scenes. The running time of the underlying ARD solver does not depend on the geometry of the scene, but rather the volume of the scene and the number of materials.

4.4 Results and Analysis of Gradient-Based Acoustic Optimization

In this section we examine the results of our design optimization process for acoustic materials. We show convergence on various scenes and the effect of the optimization, including impulse responses and the value of various acoustic characteristics before and after optimization.

4.4.1 Benchmarks

A variety of scenes and acoustic materials were used in our experimental setup. All simulations were done with a low frequency (187 Hz) Gaussian derivative pulse as a sound source. Source and listener positions were placed in reasonable locations. We used three benchmark scenes in total: Cathedral, Twilight, and Concert Hall. These provide three locations in which the acoustic properties of the scene are important. Cathedral provides a large, open area that can be particularly reverberant even at higher absorption values. Twilight has a unique architecture including parts of the structure that are open-air. Finally, the Concert Hall model allows us to explore the acoustic characteristics of music hall design. The geometry and material segmentation of each scene is shown in Figure 4.5. These scenes range in volume from the smallest (Cathedral), which is 20 000 m^3 to the largest (Concert Hall), which is 35 000 m^3. Table 4.1 shows a summary of the scenes, their respective volumes, their geometric complexity, the number of material segments used, and the iteration time for a single optimization step.

4.4.1.1 Acoustic Materials

The acoustic materials in our scene are determined as a result of our optimization process (see Figure 4.2). Acoustic materials determine the absorption coefficients of different walls or other geometric elements in our scene. These coefficients were constrained to a range of 0.1 to 0.7 (where 0 is fully reflective and 1 is fully absorptive), which limits absorption to a more realistic range.

Some typical absorption values for our range of materials can vary from painted concrete at a coefficient of about 0.1 to materials representing an audience in upholstered seating with a coefficient of about 0.6.

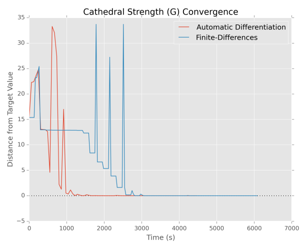
4.4.1.2 Acoustic Metrics and the Impulse Response

We determine the values of various acoustic metrics by measuring the impulse response at a particular listener location. The impulse response is a measure of sound pressure over time when the room is excited by a sound impulse at a specific source location. We used a Gaussian derivative impulse, which has a zero DC-component (which insures the correctness and numerical stability of the solver). The length of the IR is dependent on how much sound energy remains in the scene.

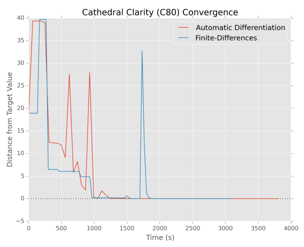
We measured two different acoustic measurements: Strength (G) and Clarity (C80). These acoustic properties were specifically chosen for their importance in concert hall design (Beranek, 2011; Miśkiewicz et al., 2012). All of these acoustic properties were derived from the impulse response. Acoustic Strength is computed from the IR by comparing the logarithmic ratio of energy in the impulse response with the energy of an impulse response computed with an equivalent sound source 10 m away in a free-field condition. Generally, Strength indicates the fullness of the sound. Clarity, on the other hand, gives an idea of how clearly the listener can hear the original sound source. Computing clarity is done by comparing the logarithmic ratio of the first 80 ms of sound to that of the remaining portion of the impulse response. This essentially measures direct sound and early reflections and compares it to the less clear reverberant tail of the impulse response. In all cases, we chose target values that were realistic and represented common practice for the design of concert halls. We chose a target value for acoustic Strength between 3 dB and 5 dB (Hyde and Möller, 2006) and a target Clarity value between −3 dB and 4 dB (Ballou, 2013).

4.4.2 Convergence

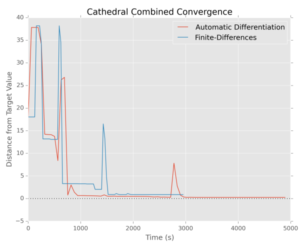
Our approach using Automatic Differentiation compares favorably to computing the derivative via a finite differences technique. The finite differences technique is a simple approach that measures the slope of a function using an epsilon x-component rather than an infinitesimal. In our experiments we picked an epsilon of 10^{-5}. Figure 4.6 shows the number of optimization steps required when computing sound Strength (G), Clarity (C80), and a combined acoustic measurement of both Strength and Clarity on the cathedral scene. We compare the AD approach with the finite difference approach. The AD approach converges faster, and in some cases the finite difference approach does not converge. In this case, it is stuck in a local minimum



(a) Strength (G)



(b) Clarity (C80)



(c) Combined Optimization

Figure 4.6: Comparison between different derivation methods: Automatic Differentiation and finite differences at 10^{-5}. We show much faster convergence using Automatic Differentiation for the Strength and Clarity measure. In the combined optimization, the finite differences result finishes early without converging on the target metric. The combined optimization attempts to optimize for both strength and clarity.

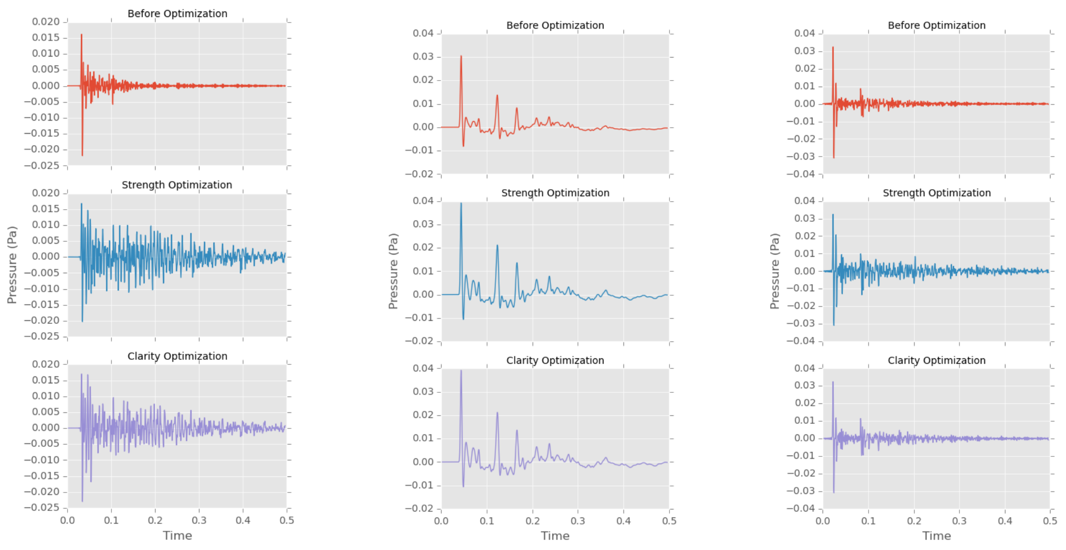
Scene	Metric	Target (weight)	Initial Materials	Final Materials	Final Metric Values
Cathedral	G	$4\,\text{dB}\ (1)$	$0.3, 0.3$	$0.0455, 0.2333$	$4\,\text{dB}$
	C80	$1\,\text{dB}\ (1)$	$0.3, 0.3$	$0.0472, 0.6814$	$1\,\text{dB}$
	G, C80	$4\,\text{dB}\ (0.25)$,	$0.3, 0.3$	$0.0448, 1$	$2.9242\,\text{dB}$
		$1\,\text{dB}\ (0.75)$			$1.0001\,\text{dB}$
Twilight	G	$4\,\text{dB}\ (1)$	$0.5, 0.5, 0.5$	$0.3884, 0.5, 0.4852$	$3.9998\,\text{dB}$
	C80	$1\,\text{dB}\ (1)$	$0.5, 0.5, 0.5$	$0.2725, 0.4998, 0.516$	$0.9995\,\text{dB}$
	G, C80	$4\,\text{dB}\ (0.25)$,	$0.5, 0.5, 0.5$	$0.3288, 0.4988, 0.9756$	$4.0003\,\text{dB}$
		$1\,\text{dB}\ (0.75)$			$0.999\,\text{dB}$
Concert Hall	G	$4\,\text{dB}\ (1)$	$0.5, 0.5, 0.5, 0.36$	$0.1, 0.1, 0.7, 0.1$	$2.9149\,\text{dB}$
	C80	$1\,\text{dB}\ (1)$	$0.5, 0.5, 0.5$	$0.1468, 0.4177, 0.3593$	$1\,\text{dB}$
	G, C80	$4\,\text{dB}\ (0.25)$,	$0.5, 0.5, 0.5, 0.36$	$0.1, 0.7, 0.7, 0.1$	$2.4050\,\text{dB}$,
		$1\,\text{dB}\ (0.75)$			$6.0772\,\text{dB}$

Table 4.2: The various starting and ending material values for optimization. The value w represents the assigned weight for each value. We show close convergence for all scenes in this example. The material range was limited to absorptions of between 0.1 and 0.7.

that is not the optimal result that the AD approach computes. Overall, using AD is advantageous since it converges to the correct result in addition to converging faster.

4.4.3 Optimization Results

In addition to examining the convergence of our optimization method, we show the effect our optimization has on both the entire pressure field and individual impulse responses at specific listener positions. We ran various experiments one each scene, including one for each acoustic metric and a third experiment for a



(a) Cathedral (b) Twilight (c) Concert Hall

Figure 4.7: Impulse responses before and after optimization on each scene. Optimizing for Strength increases the overall energy in the response, while optimizing for Clarity increases the ratio energy in the first 80 ms of the impulse response to the remaining energy. These changes can be seen in the figures after optimization is performed.

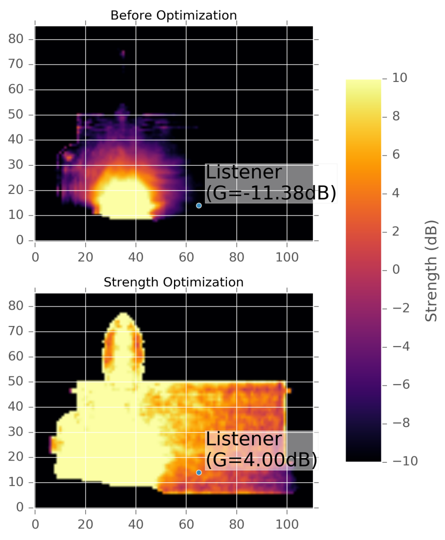
combined metric that targeted both Strength and Clarity for optimization. Table 4.2 shows various material values before and after optimization for the various acoustic characteristic metrics and combinations.

Figure 4.7 shows the impulse responses at each scene before optimization and after optimizing for the different acoustic metrics. Materials with lower absorption values will generally tend to yield impulse responses with more energy after the direct sound impulse (the first maxima in the impulse response). Because acoustic Strength uses the energy of the full impulse response, Strength optimization tended to increase overall energy. On the other hand, Clarity optimization tended to decrease the proportion of the second part of the impulse response compared to the first part, despite overall increasing the reflectivity of the scene materials.

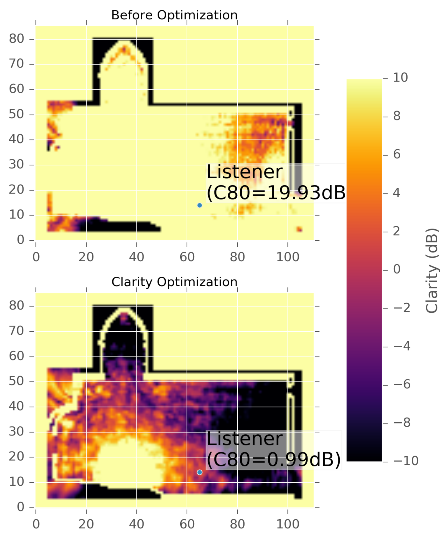
Figure 4.8, Figure 4.9, and Figure 4.10 show the acoustic Strength and Clarity values before and after optimizing for acoustic Strength. These images show how the results for a single source and listener position can be used to drive the acoustic design of a concert hall or other acoustic space.

4.4.4 Analysis

Our analysis shows that our method is effective in determining the material parameters that effectively yield the desired acoustic characteristics. We are able, in most cases, to complete the optimization in less

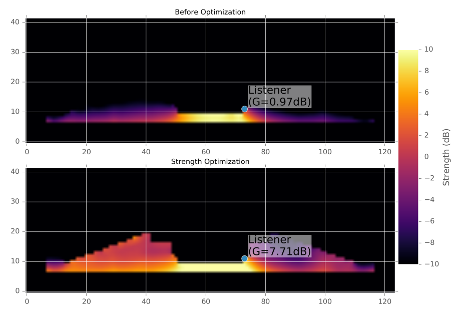


(a) Strength (G) Optimization

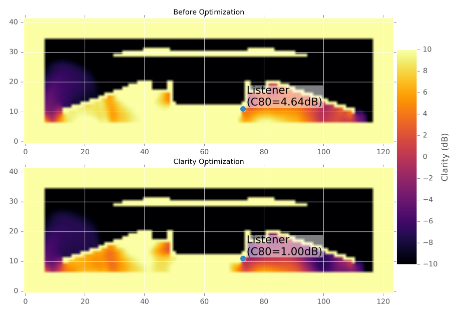


(b) Clarity (C80) Optimization

Figure 4.8: Cathedral field slices before and after optimization. Optimizing the acoustic materials causes a change in the full acoustic field. At the sample listener position we use for the impulse responses in Figure 4.7, we show the desired target acoustic value after optimization.

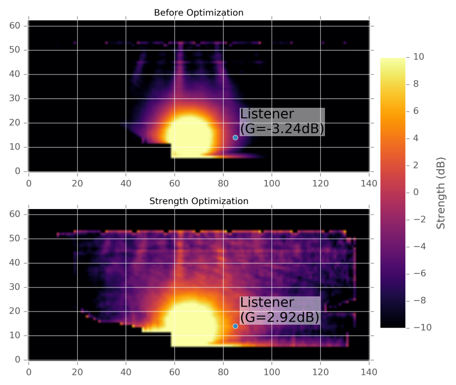


(a) Strength (G) Optimization

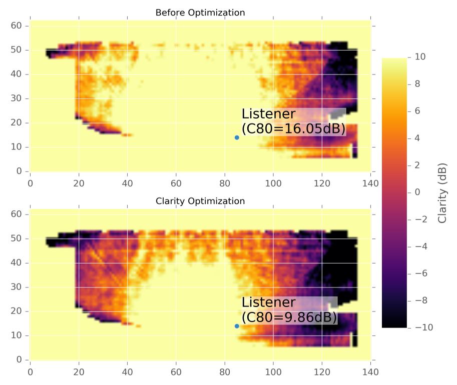


(b) Clarity (C80) Optimization

Figure 4.9: Twilight field slices before and after optimization. In this scene partly open to air, we can see how the strength and clarity values outside of the main structure are affected by the optimization of acoustic materials.



(a) Strength (G) Optimization



(b) Clarity (C80) Optimization

Figure 4.10: Concert Hall field slices before and after optimization. The listener position in this case is further away from the source and mostly affected by direct sound, yielding a high clarity but lower strength.

time than an equivalent finite difference technique for determining the gradient. In the cases in which we do not, our method yields results that are more accurate and closer to the desired acoustic characteristics.

Additionally, our method is general purpose and is capable of working on a multitude of scenes. The only input is the scene with the desired material segment assignments. This is particularly useful for engineers and architects as arbitrarily complex CAD models can be used for optimization.

Finally, our method uses a fast underlying acoustic wave propagation simulation that can give accurate results with much lower computational and memory requirements compared to other standard methods such as FDTD. For example, the clarity optimization on the Cathedral scene took approximately 1 hour to compute. Using FDTD would take around 75 hours to compute (see Chapter 3). This can make an important difference in turnaround time for architectural acoustic design and development.

4.5 Discrete material optimization for wave-based acoustic design

One drawback of the continuous material optimization approach is the difficulty of adding useful constraints on acoustic materials. While the minimum and maximum absorption of a material can be specified by box constraints, more complex constraints such as the quality or type of material cannot be easily incorporated. Additionally, continuous material absorption values are not necessarily feasible to construct in the real-world — rather than having arbitrary absorptions, most acoustic materials are constructed for specific absorption levels.

We propose a practical, accurate, and efficient method for the automated determination of acoustic materials in large architectural structures using real-world material databases. Our algorithm is based on discrete optimization techniques, which allows the use of constraints and realistic materials in the design process. Furthermore, our algorithm uses the wave-based Adaptive Rectangular Decomposition (ARD) solver for evaluating the acoustic field in an efficient and accurate manner. The solver and optimization algorithm are general purpose: the only input is a CAD model along with the desired acoustic properties of the scene. These properties can be any metric determined by the acoustic impulse response, although we focus on Strength (G), Clarity (C80), and Reverb Time (RT60).

4.5.1 The Objective Function

Generally, acoustic engineers have multiple requirements for the design of a concert hall in order to obtain the desired sound (Egan, 1988; Lokki et al., 2011a; Beranek, 2008). For each metric H_k, with $k = 1, \dots n$ we compute the minimization function for that metric:

$$O_k(\Omega) = \parallel H_k(\Omega) - Z_k \parallel , \tag{4.15}$$

where H_k is the function for computing the acoustic metric, Ω is the domain of the system including the set of various material parameters, and Z_k is the target acoustic metric specified by the user.

However, we are interested in the multiobjective formulation, taking into account multiple acoustic metrics. This, the objective function is a vector:

$$\Omega' = \arg\min \{O_1, O_2, \dots, O_n\} , \tag{4.16}$$

The result of minimizing the objective function is a set of new materials that yields an optimal material configuration for the desired acoustic metrics, Ω'.

We can define a Pareto-optimal solution for this objective function as one that is not dominated by any other solution; that is, in our formulation, one where no other configuration has all metric values closer to the target metric value. The goal of the optimization procedure is to find configurations on the Pareto-front.

More formally, Ω' is said to be a Pareto optimal decision vector if $O_k(\Omega') \leq O_k(\Omega)\, \forall k$ and $O_k(\Omega') < O_k(\Omega)$ for some k (Nam and Park, 2000).

The Pareto-optimal formulation avoids the need for user-defined weights that do not always have a predictable effect on the optimization and in some cases cannot find solutions in concave regions (Nam and Park, 2000). Additionally, with this objective function, arbitrary acoustic metrics can be used in addition to the ones used in our experiments. We believe that depending on the underlying application, different metrics can be used. In this paper, we focus on Strength, Clarity, and Reverberation Time, but in general our approach can be applicable to any acoustic metrics that are computed using the acoustic pressure field or impulse response. However, it is important to consider that some metrics cannot be achieved in certain room configurations; in a larger room, for example, sound strength will decay past a certain distance from the source position because of the comparison with a source 10 m away in a free field condition (ISO, 2009). Thus optimizing for high strength in an inaccessible position in the scene would not be possible.

4.5.2 Optimization Approach

The primary challenge in discrete material optimization for acoustic scenes is the dimensionality of the optimization problem. In our technique, scenes are divided into user-controlled material segments, where every surface that is part of a segment shares the same material. Examples of this include segments encompassing the floor or the ceiling where every part of the surface has the same absorption value. Therefore, the total size of the optimization search space is $\mid L \mid^n$ where n is the number of material segments and L is the set of materials available in the material library. Even a relatively small search space such as one with 5 material segments and 30 available materials has several million total combinations. Larger ones such as those provided by Egan (Egan, 1988) or by an acoustic consultant make a brute-force approach intractable. Additionally, the non-convex shape of the objective function can limit gradient methods where optimization can result in a local minima that is not the global minima. Furthermore, this local minima can be far from the optimal solution. Ideally, local minima are acceptable as long as there is no perceptible difference in the optimization quantities or the distance from the optimal solution is within the specified range. For example, it is common for concert halls to be designed for a range of sound clarity between −2 dB and 4 dB.

There are further challenges in optimization due to the computational cost in evaluating the objective function (Equation 4.16). As evaluating the objective function requires evaluating the wave equation, this is an $O(f^4)$ asymptotic cost per iteration of the optimization function, where f is the maximum simulation frequency. In discrete optimization techniques like Genetic Algorithms (GA) or Particle Swarm methods, multiple evaluations of the objective function are needed each iteration, which makes the per-iteration cost even more expensive.

4.5.2.1 Simulated Annealing

These challenges are overcome in our optimization algorithm based on a simulated annealing approach. The discrete nature of simulated annealing means that materials and constraints can be specified by the acoustic engineers and then used by the algorithm. Furthermore, the problem statement allows for acceptable local minima: the algorithm is not constrained to only the most optimal material configuration, but rather one that satisfies the design requirements. This is especially relevant since it may not be possible for the optimization problem to converge to the desired target solution. Simulated annealing also requires only one evaluation of the objective function per iteration, which means that the number of evaluations can be limited if the number of iterations is kept low.

Algorithm 1 describes the simulated annealing approach used for Pareto-optimal discrete material optimization. Neighbor states are computed by selecting a random material segment to modify. The modification is either accepted or rejected depending on whether a given state vector dominates the previous state. Additionally, the algorithm has a chance of accepting a solution not on the Pareto front if the temperature is high. Over time, the temperature decreases so that the algorithm is more likely to accept less optimal values near the start. This probabilistic approach allows the optimization technique to search through the problem space in an efficient manner. All elements of this material segment are then assigned the same random material. This allows for easy determintation of the neighbor states. Besides the initial state computed by the underlying acoustic simulator, each iteration of the simulated annealing algorithm computes the impulse response only once, no matter how many acoustic parameters are specified in the objective function.

We use a modified version of the typical simulated annealing algorithm that stores the current best solution for every iteration. The strength of simulated annealing is that it can explore less-optimal solutions early on in the process, thus avoiding local minima. However, in some situations, the algorithm may converge on a local minima when the temperature is low. In these situations, we can use the current best solution. This is generally only a problem when the optimization does not converge and it is desirable to have a solution with properties as close as possible to the desired characteristics.

```
Input  : Scene definition $\Omega$
         Initial temperature $T_0$
         Cooling rate $\alpha$
Output : Optimal materials $q$
initialize simulator;
/* Evaluate initial simulation state                                    */
$q \leftarrow$O($\Omega$);
$T \leftarrow T_0$;
while $T > 1$ do
    $k \leftarrow$Rand(numMaterials);
    RandomizeMaterial($\Omega_k$) /* Compute new state                   */
    $q' \leftarrow$O($\Omega$);
    if AcceptEnergy($q$,$q'$,$T$) then
        $q \leftarrow q'$;
    end
    $q \leftarrow$Min($q$,bestState);
    $T \leftarrow \alpha T$;
end
Procedure AcceptEnergy($q$,$q'$,$T$)
    if EnergyDominates($q'$, $q$) then
        return true;
    else if EnergyDominates($q$, $q'$) then
        $p \leftarrow \exp \frac{\sum q - \sum q'}{T}$;
        return $p <$ Rand(0,1);
    else
        return true;
Procedure EnergyDominates($q'$, $q$)
    foreach metric index $k$ do
        if $q'_k \geq q_k$ then
            return false;
        end
    end
    return true;
```

Algorithm 1: The simulated annealing algorithm for discrete acoustic material optimization. Evaluating the objective function $O(\Omega)$ requires evaluation of the ARD wave-based solver $H(\Omega)$. We use a Pareto-optimal variation of the simulated annealing algorithm, and accept a solution that dominates a previous solution. In the case in which the previous solution dominates the new one, we accept it with a probability metric depending on the sum of the energy in the solution.

4.5.2.2 Reducing Iteration Cost

The primary cost of wave-based optimization is the valuation of the pressure field and impulse response. This needs to be done for each evaluation of $O(\Omega)$. Therefore, reducing the number of iterations is beneficial to the performance of the algorithm. Our algorithm use two primary methods for this: adaptive parameterization and early stopping criteria.

The simulated annealing process requires two parameters aside from the objective function: the initial temperature T_0 and the cooling factor $0 < \alpha < 1$. These define the cooling function, which directly influences whether or not a given solution is accepted. Because simulated annealing is a heuristic approach, its performance directly depends on these parameters. A high initial temperature in conjunction with a very small cooling factor will give very good optimal solutions at the cost of many evaluations of the objective function. Since this entails the computation of the entire impulse response using the ARD wave-based solver, it is expensive to use many iterations to converge to the optimal solution. Typical recommended values of α are between 0.8 and 0.99 (Eglese, 1990).

Our algorithm uses an adaptive technique for determining T. In practice, values of T that yield an 80% acceptance rate for initial energies that are greater than the previous step's energy yield good convergence rates (Eglese, 1990). Since an initial state is used that is either randomly-generated or user-provided, the algorithm can determine T dynamically using the initial state and the next state.

The discrete optimization algorithm uses a number of stopping criteria for the simulated annealing optimization process. These are the generally minimum temperature and maximum number of iterations but can also include stopping at solutions that are within a desired range. This allows for ending the process early in these situations because often specifications for concert halls and other similar structures are specified in ranges with tolerance. Additionally, the Just-Noticeable Difference (JND) of human perception of these metrics implies an inherent tolerance. These values can be as low as 5% for RT60 and 1 dB (ISO, 2009) for C80 but can vary according to the application and design of the concert hall (Blevins et al., 2013). Using range as a stopping criteria is particularly useful because it helps to avoid redundant calculations at the end of the annealing process: with a very low probability of acceptance most evaluations of the objective function will involve rejections. Our implementation uses a tolerance ϵ as threshold for which the annealing process can be terminated early.

Scene	Interior Volume	Computational Volume	No. materials ($\vert\Omega\vert$)
Cathedral	10 699 m^3	20 686 m^3	2
Concert Hall	17 623 m^3	34 510 m^3	5

Table 4.3: Volumes and number of material segments for each scene. Our algorithm is able accurately compute wave propagation on very large architectural scenes using ARD. Additionally, our system can handle very complex scenes with a large range of materials. In this table, the interior volume refers to the volume of the acoustic space, while the computational volume refers to the volume used in the acoustic solver. For example, the cathedral scene has holes for windows and we add an absorbent wall on the exterior to emulate a free-field condition. The computational volume includes this area.

4.5.3 Acoustic Materials

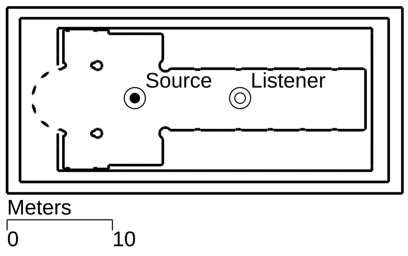
ARD uses per-frequency absorption values. Each frequency band is then simulated separately. Absorption values are taken from the material database M (see (Egan, 1988) for a list of materials that we have used in our experiments) and a set of constraints K so that

$$\{\Omega_j \in \Omega \mid \Omega_j \in M \wedge \Omega_j \notin K\}\,. \tag{4.17}$$

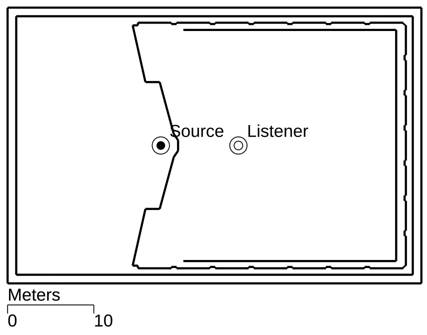
Using constraints reflects the nature of real world materials where some materials could not be realistically used for some parts of a structure or architects want to avoid using some materials for aesthetic purposes. Additionally, factors such as material costs could be incorporated to integrate our algorithm with a more general architectural design algorithm.

4.6 Results and Validation of Discrete Material Optimization

Experiments using our algorithm show optimal results on different varieties of scenes. The primary scenes used for the experiments were the Šibenik Cathedral in Croatia and the Royal Concertgebouw in Amsterdam. These scenes were selected because they are CAD models of real-world architectural scenes in which the acoustics of the building are important to the design. The scenes, their volumes, and the number of material segments used in the optimization process are summarized in Table 4.3. Each scene has two separate volumes: the interior volume, which represents the acoustic space, and the computation volume, which represents the total volume of the bounding box that is used to compute the propagation solution in the volume. The bounding box is typically larger than the scene and helps emulate a free-field or outdoor condition in the case of open windows or other gaps in the scene mesh.



(a) Cathedral



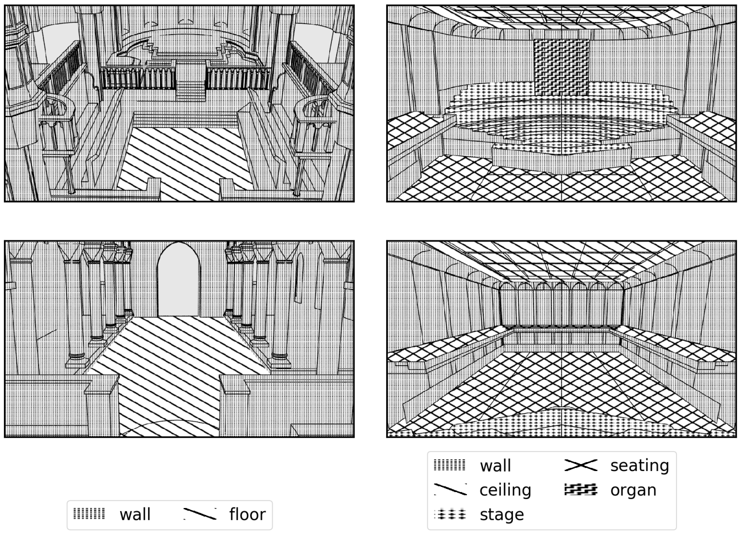
(b) Concert

Figure 4.11: Overview of the scene geometry, source, and listener positions. These figures are a top-down slice of each scene, showing the source and listener relative to different regions of the scene. Note that the source and listener positions may be at different heights.

4.6.1 Experimental Setup

Experiments were run in single-threaded code on a 3.40 GHz Intel i7-3770 64-bit processor. The experiments used an initial temperature $T_0 = 4.48$ and cooling factor $\alpha = 0.97$. We used a 500Hz sound source for our experiments. Figure 4.11 shows a 2D slice of the scenes from the top down, including source and listener positions used for the optimization process. In our experiments, results used a starting point selected with a uniformly random selection of materials.

Target values Z_i and their tolerances for the single-objective optimization are specified in Table 4.4. These values were used for all experiments and the values selected according to common requirements for concert halls in addition to psychoacoustic findings (Egan, 1988; Lokki et al., 2011a; Beranek, 2008). In scenes such as the concert hall where measurements are available, we used measured values as the targets.



(a) Cathedral (b) Concert

Figure 4.12: Experimental setup with material segments of each scene. Initial materials were assigned randomly.

Scene	$G \pm \epsilon$	*C80* $\pm \epsilon$	*RT60* $\pm \epsilon$
Cathedral	$2(1)\,\mathrm{dB}$	$-5(1)\,\mathrm{dB}$	$4.0(5)\,\mathrm{s}$
Concert Hall	$2(1)\,\mathrm{dB}$	$-3.3(10)\,\mathrm{dB}$	$2.2(5)\,\mathrm{s}$

Table 4.4: Target values and weights for the single-metric optimization optimization. The $\pm\epsilon$ refers to the threshold range for which we end optimization early. The optimization experiments were performed for each acoustic metric both individually and with multiobjective optimization.

Scene	Material Labels	Absorption (500 Hz)		Metrics		
		Initial	Opt.	Target	Initial	Opt.
Cathedral	wall	0.83	0.2	G=2 dB	−1.11 dB	4.34 dB
	floor	0.14	0.17	C80=−5 dB	6.79 dB	−2.05 dB
				RT60=4 s	1.44 s	1.45 s
Concert	wall	0.78	0.78	G=2 dB	1.678 dB	2.93 dB
	ceiling	0.1	0.08	C80=−3.3 dB	−2.26 dB	−3.33 dB
	stage	0.05	0.07	RT60=2.2 s	1.5 s	1.68 s
	seating	0.49	0.06			
	organ	0.8	0.63			

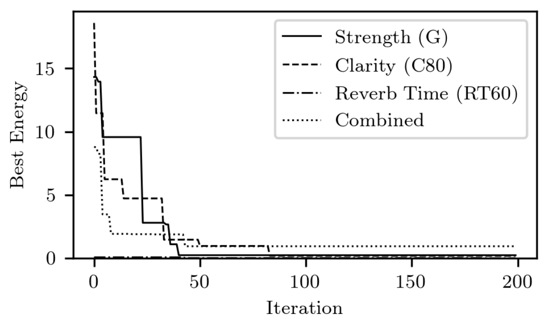
Table 4.5: Summary of the multiobjective optimization experiments. For these results we used Strength (G), Clarity (C80), and Reverb Time (RT60). The initial material absorption values in these experiments were randomized, so the initial Strength or Reverb Time metrics are the result of the randomized materials.

4.6.2 Multiobjective Optimization

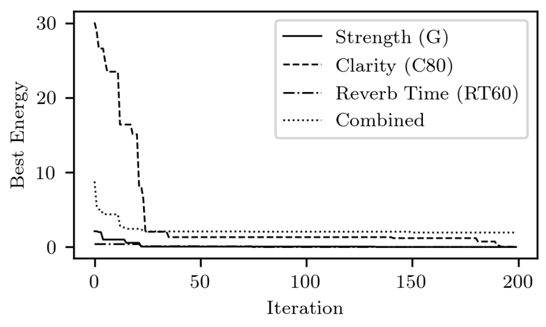
In addition to single-objective optimization, the multi-objective optimization function was tested. For these experiments, we used a multi-objective function that included all three metrics. The results are summarized in Table 4.5. On all three scenes, our algorithm optimizes Strength within 1 dB, the just-noticeable difference of human hearing. The results of the experiment also depended on the initial random state; for example, the experiments on the concert hall yielded almost no improvement in reverberation time, but that is partly because the reverberation time for the initial state was already close. However, the other objectives were less optimal.

4.6.3 Convergence

We performed several convergence experiments on the three scenes. The first experiment uses a fixed number of iteration steps, and for performance reasons were limited to 125 Hz as the number of iterations was large. In this case, there is no tolerance specified so the simulated annealing process does not halt early. The results show energy over the number of iterations in Figure 4.13. The closer this value is to zero, the closer the objective function is to the target values. This experiment used a fixed initialization of material parameters.

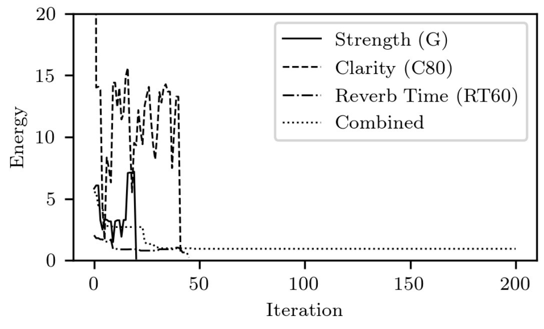


(a) Cathedral

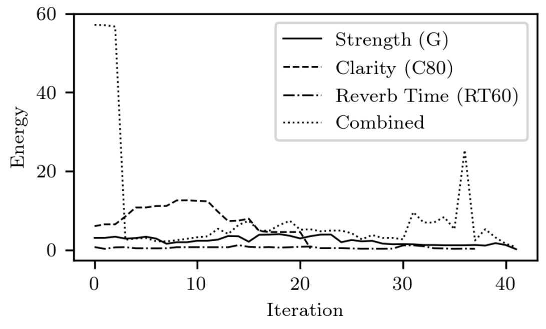


(b) Concert Hall

Figure 4.13: Convergence up to 200 iterations of the simulated annealing process in single-objective optimization. The plots show the best energy at each iteration. Our method shows fast convergence in a fixed number of steps, for complex real-world scenes like the Concert Hall.



(a) Cathedral



(b) Concert Hall

Figure 4.14: Convergence of simulated annealing for adaptive initial temperature and early-cutoff. The adaptive schedule shows faster convergence compared to fixed-step and non-adaptive initialization.

4.6.3.1 Early-Cutoff and Adaptive Initial Temperature

Using early cutoff and an adaptive initial temperature improves the convergence of our simulated annealing algorithm. Figure 4.14 shows the convergence rate resulting from the use of early-cutoff and adaptive-cooling. Adaptive initialization is particularly useful in these cases because a fixed initial temperature can greatly vary the iteration time of the optimization algorithm when the scene parameters and objective function change. This also minimizes the amount of parameter tuning required in order to achieve optimal convergence.

Figure 4.15 shows the results of running multiple experiments with a uniformly random starting material configuration. Using adaptive cooling schedules, our simulated annealing algorithm converges quickly in spite of very bad initial conditions or local minima.

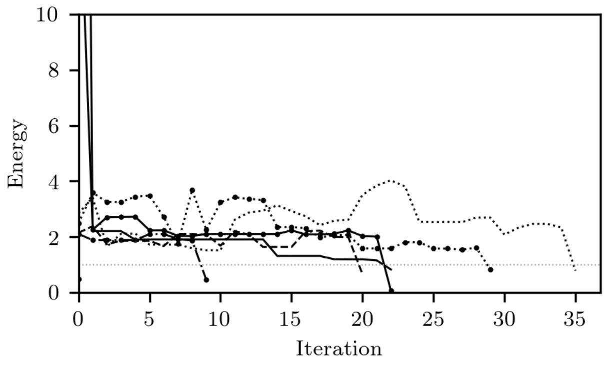


Figure 4.15: The current energy per iteration of six different experiments optimizing for acoustic Strength (G) on the Concert Hall scene. Each experiment was initialized with a random starting material configuration uniformly distributed among the possible initial states. These results include the early-cutoff; the threshold for this is specified by the dashed line. Our algorithm works well even from a very poor initial state and can avoid local minima. These results used a 125 Hz frequency in order to make multiple experiments feasible.

4.6.4 Performance

We summarize the performance results in Table 4.6 at the 500 Hz that we used for our experiments. Because of the complexity of using a wave-based solver, the running time is heavily dependent on the convergence of the algorithm. In some cases, the target goal was difficult or impossible to achieve. In those situations, convergence was slow, but results still show an improvement over the initial state. The Concert Hall scene converged particularly quickly because the use of real world measurements as the target values meant that the target values were achievable.

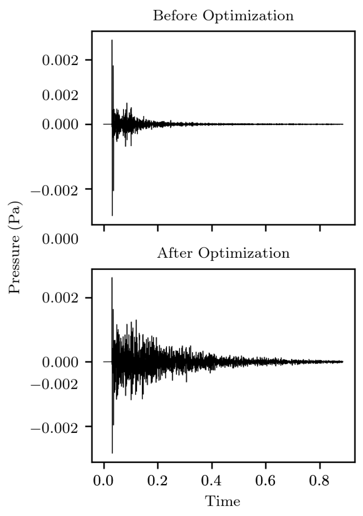
Metrics such as reverberation time (RT60) cause difficulties for the performance of our algorithm. This is because although Strength (G) and Clarity (C80) can be estimated using shorter impulse responses, reverberation time requires a large number of time steps (directly proportional to the RT60) in order to be computed accurately.

4.6.5 Impulse Responses

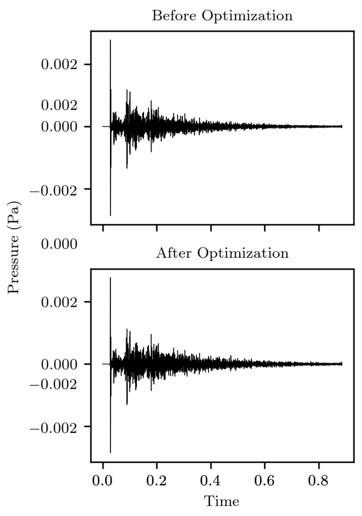
We show the effect of our combined optimization process on the impulse responses in Figure 4.16. These results used the single optimization experiments, where the goals are specified in Table 4.4.

Scene	Running Time			
	Single G	Single C80	Single RT60	Combined
Cathedral	4.74 h	19.94 h	56.92 h	136.49 h
Concert	8.15 h	11.70 h	11.86 h	105.62 h
	Iterations			
	Single G	Single C80	Single RT60	Combined
Cathedral	23	99	41	99
Concert	6	9	4	45

Table 4.6: Running time and number of iterations for our experiments including single-target G, C80, and RT60 experiments and one experiment using a multi-objective target for multiple metrics. These experiments were run at 500 Hz simulations and used the adaptive cooling schedule with a maximum number of iterations at 99.

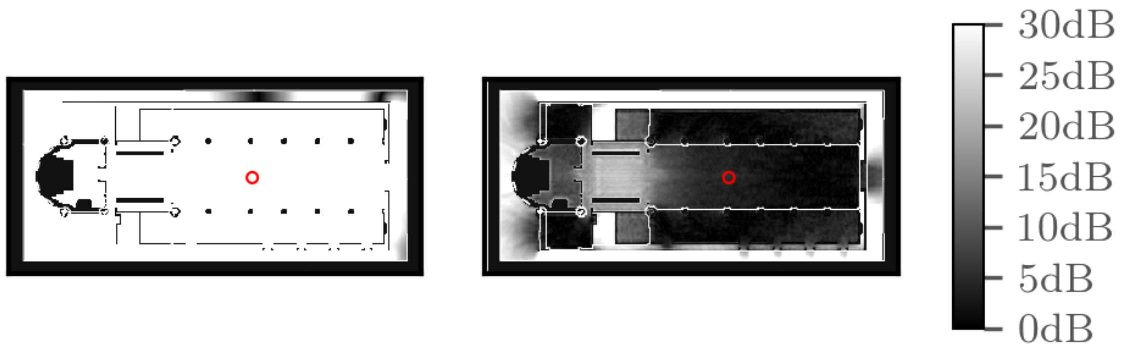


(a) Cathedral

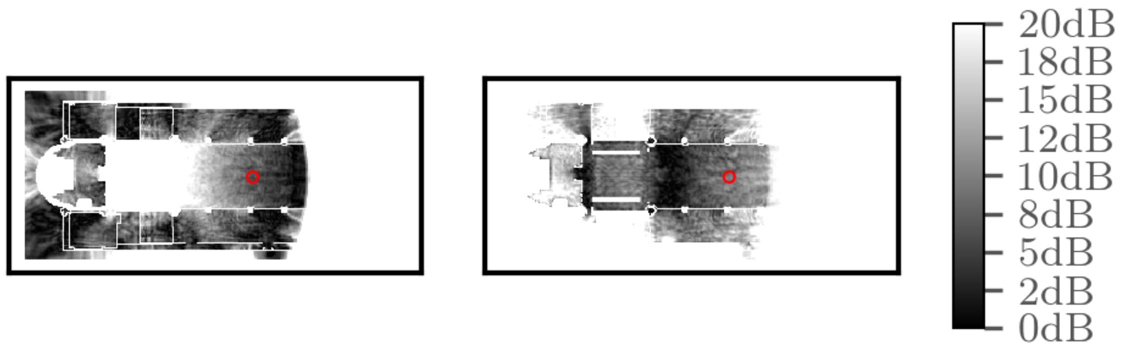


(b) Concert Hall

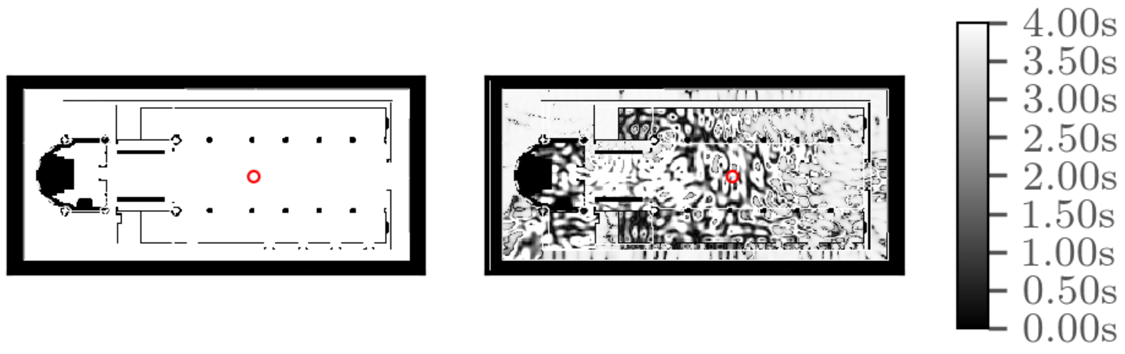
Figure 4.16: Impulse responses before and after individual optimization optimization. The initial responses are shown on the top row, while responses after combined optimizations are shown below. These impulse responses show how the acoustic characteristics of the scene change when an individual metric is optimized for.



(a) Strength (G) difference before (left) and after (right) optimization

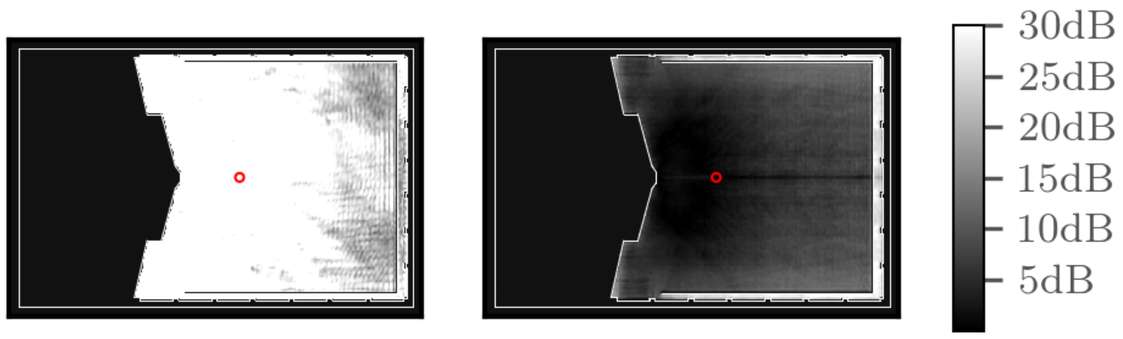


(b) Clarity (C80) difference before (left) and after (right) optimization

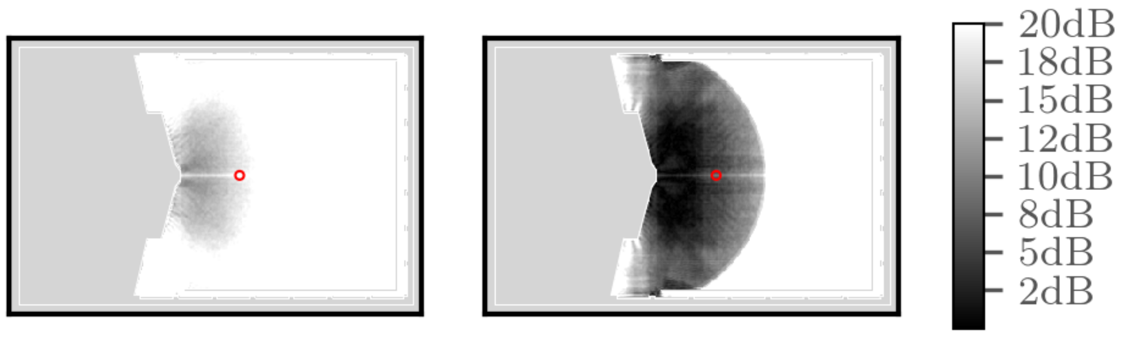


(c) Reverb time (RT60) difference before (left) and after (right) optimization

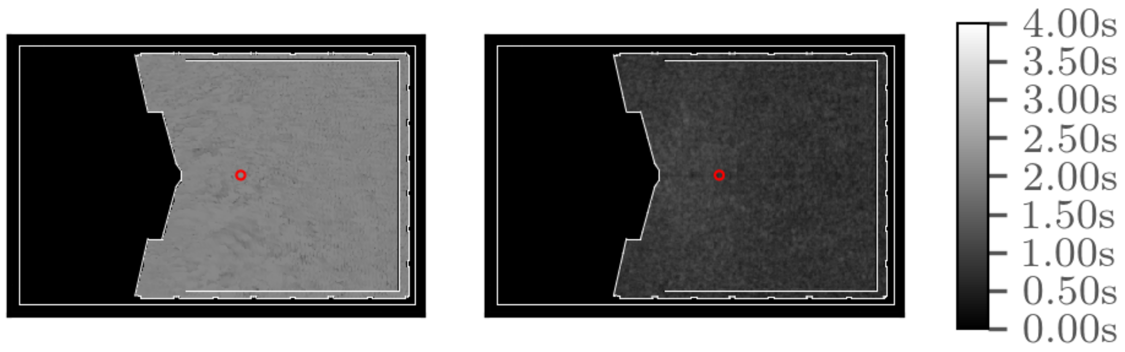
Figure 4.17: Full-field top-down slices for the Cathedral scene showing the difference from the target metric before and after optimizing for various acoustic parameters. The dot indicates the listener position used for the optimization.



(a) Strength (G) difference before (left) and after (right) optimization



(b) Clarity (C80) difference before (left) and after (right) optimization



(c) Reverb time (RT60) difference before (left) and after (right) optimization

Figure 4.18: Full-field top-down slices for the Concert Hall scene showing the difference from the target metric before and after optimizing for various acoustic parameters. The dot indicates the listener position used for the optimization.

4.6.6 Full-field Results

Other experiments were done on the full-field response of the model. These experiments show the impact of the optimization process on the entire acoustic space. The ARD simulation discretizes the acoustic space, the impulse response at each discretized location can be measured. This gives a sense of how the impulse response changes in the hall as a result of the optimization process. Figures 4.17 and 4.18 shows images of the full-field results for the various scenes using single-objective optimization.

Figures 4.19 and 4.20 show the Strength and Reverberation Time metrics over the full field as a result of the multiobjective optimization. The change in materials affects the entire acoustic space although only one listener position is used.

4.6.7 Validation

We conducted additional experiments to show a comparison between discrete optimization using our method and a similar optimization process using both Sabine's and Eyring's equation for reverb time.

ARD uses a multiplier for absorption values that applies to the interface stencil between the air region and the wall, but Eyring and Sabine use physically-based absorption coefficients. This yields a mismatch between reverb time calculations for lower absorption values. Additionally, Eyring and Sabine often suffer from reverb time overestimation compared to some computer simulation techniques (Schroeder, 1970).

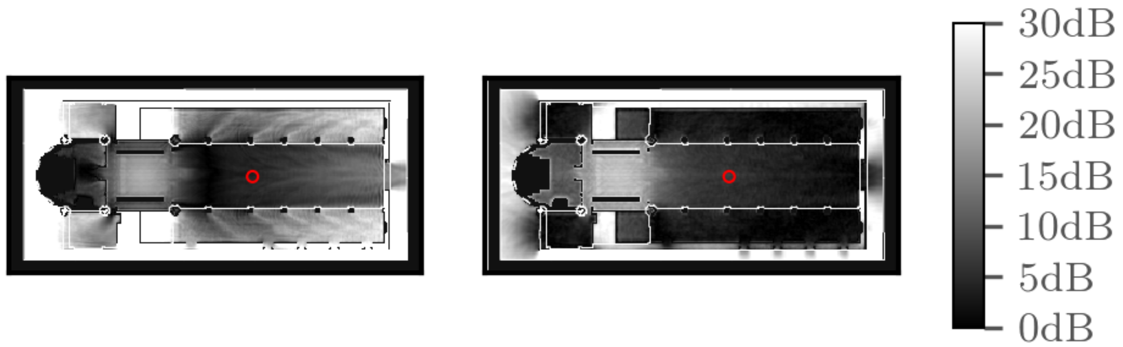
Using an empirical parameter mapping matching ARD and Sabine equations on a cube, ARD can approximate a physical material absorption value by using the internal ARD interface multiplier

$$x' = x10^{\alpha}, \tag{4.18}$$

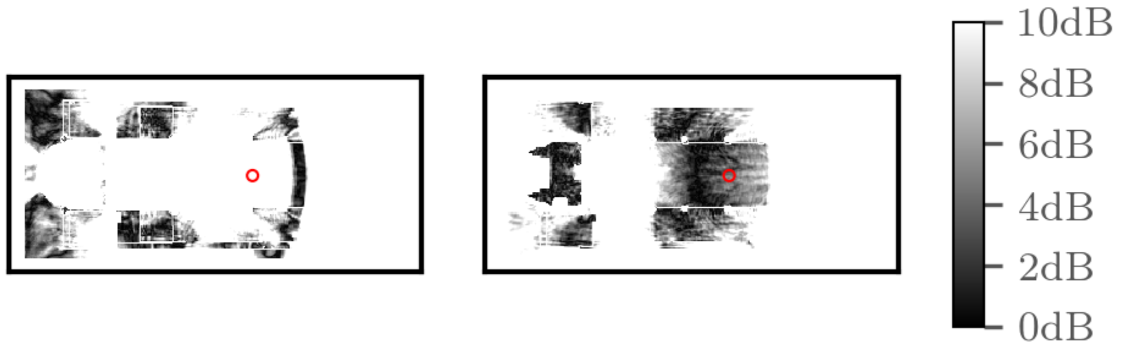
where

$$\alpha = \frac{R - (1 + c_0)X - c_1}{c_0}, \tag{4.19}$$

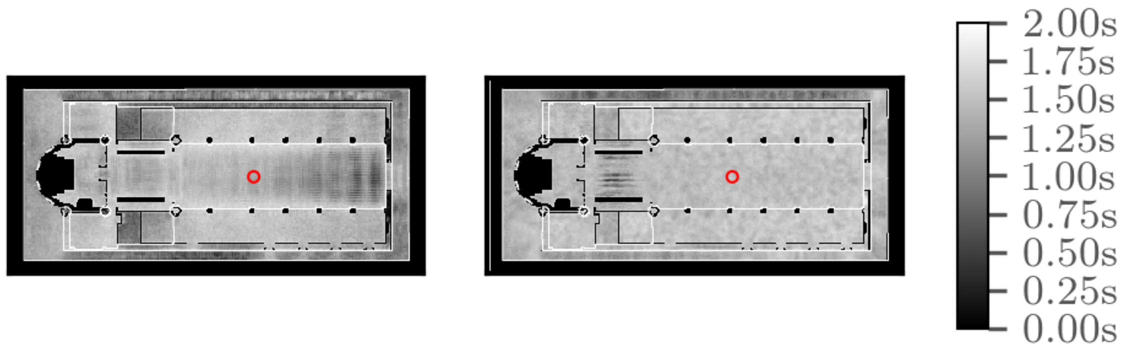
with R being the Sabine ratio $\frac{0.1611V}{S}$ with volume V and surface area S in log space, X being the log-space absorption, and c_0 and c_1 being experimentally-determined constants.



(a) Strength (G) difference before (left) and after (right) optimization

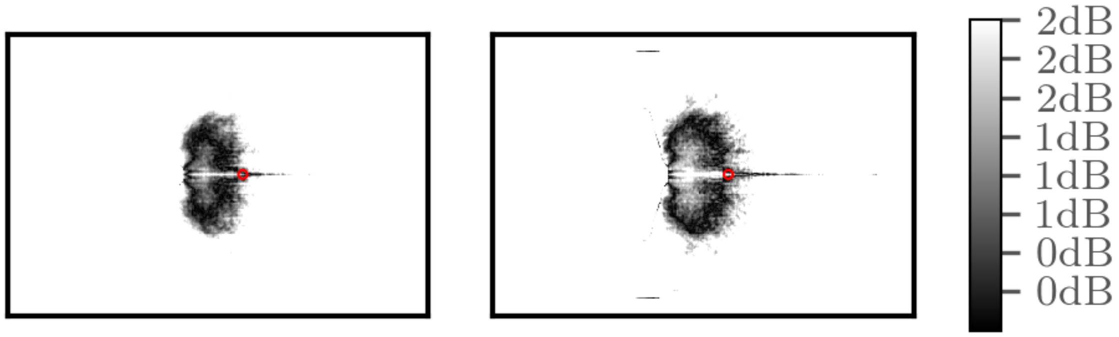


(b) Clarity (C80) difference before (left) and after (right) optimization

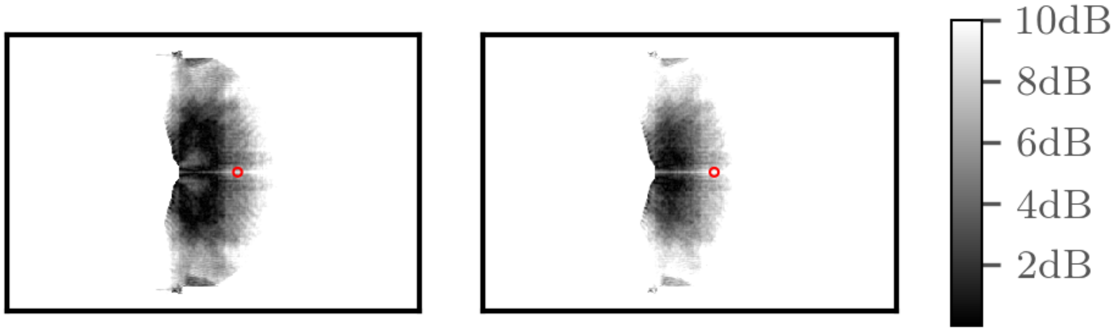


(c) Reverb time (RT60) difference before (left) and after (right) optimization

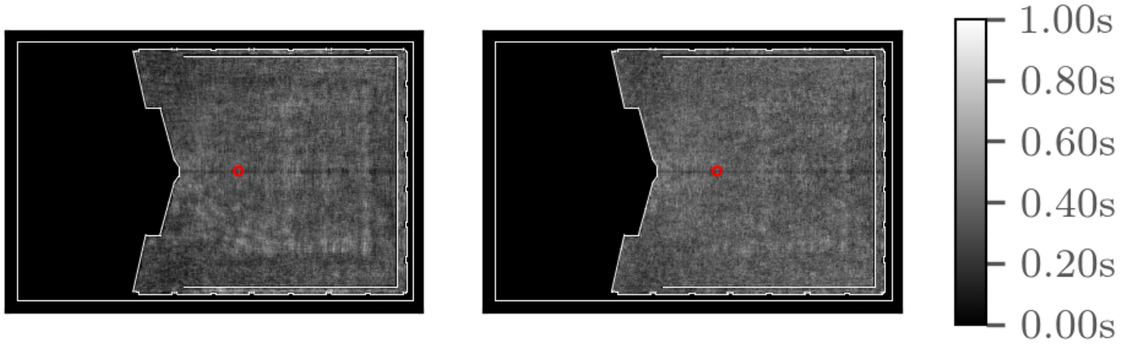
Figure 4.19: Full-field top-down slices for the Cathedral scene showing the difference from the target metric before and after multiobjective optimization. The dot indicates the listener position used for the optimization.



(a) Strength (G) difference before (left) and after (right) optimization



(b) Clarity (C80) difference before (left) and after (right) optimization



(c) Reverb time (RT60) difference before (left) and after (right) optimization

Figure 4.20: Full-field top-down slices for the Concert Hall scene showing the difference from the target metric before and after multiobjective optimization. The dot indicates the listener position used for the optimization.

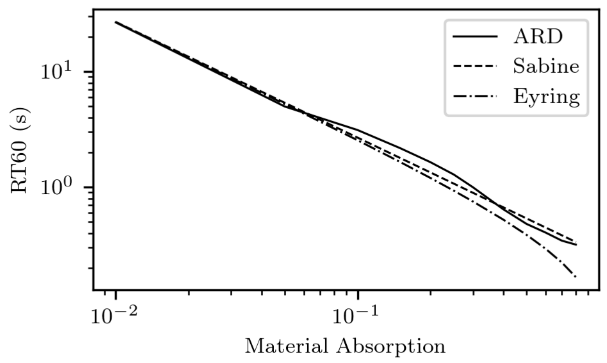


Figure 4.21: Logarithmic plot of reverberation time for ARD compared to Sabine and Eyring.

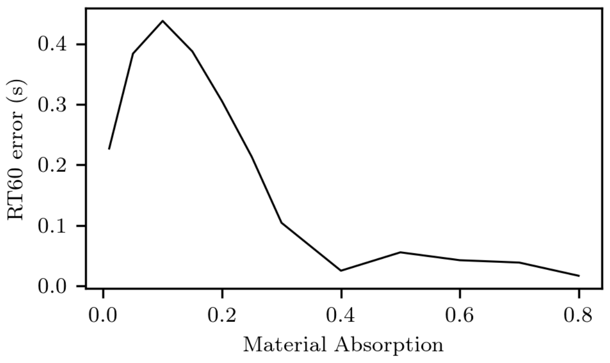


Figure 4.22: Absolute error of the RT60 in seconds of ARD absorption values compared to Sabine values.

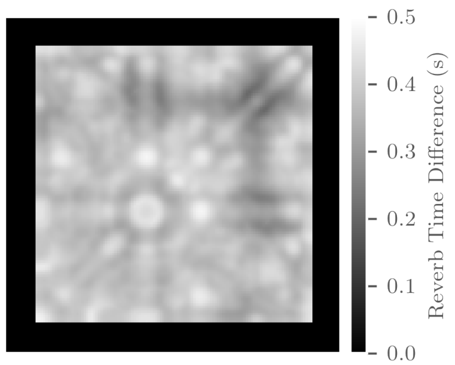


Figure 4.23: Full-field plot showing the difference between the reverberation time at each grid point using our discrete optimization technique and using Sabine's equation.

Method	Material Absorption	RT60	Error (s)
Sabine	0.2235	1.79015	+0.29015
ARD	0.18	1.36391	−0.13609

Table 4.7: Comparison of discrete material optimization using ARD and using Sabine. The scene was a cube scene, where Sabine is accurate. The target RT60 value was 1.5 s.

The mapping has some error, but it is limited. Figure 4.21 shows a comparison of the reverberation time for various absorption values comparing ARD, Sabine, and Eyring on a cube scene. Figure 4.22 shows the characteristics of the matching error over various absorption values.

This formulation was used for an optimization comparison on a simple cube scene, of dimension $10 \times 10 \times 10\,\mathrm{m}$ where Sabine's equation would be accurate. The comparison was done by using the same starting point for each optimization method. Then, the resulting material definitions were tested by computing the reverberation time with the ARD method for both optimization results. Figure 4.23 shows a full-field comparison between the resulting reverberation times and Table 4.7 shows the error when comparing optimization using ARD and Sabine.

4.7 Conclusion and Future Work

We introduce two algorithms for efficient wave-based acoustic material design optimizer that are capable of handling multiple material segments and a multiobjective optimization target.

In the first approach, We show that using the exact derivatives from Automatic Differentiation helps us converge faster on the target optimization result. Additionally, we take advantage of the performance and memory efficiency of the ARD solver compared to other standard acoustic wave solvers. Finally, we show how our system can be used in the application of designing concert halls or other acoustic spaces.

Our second approach addresses some of the limitations of continuous optimization approaches for acoustic materials. Our discrete multiobejctive optimization method allows for the use of real-world material libraries and databases for the optimization process. Additionally, experiments demonstrate fast convergence using a modified Simulated Annealing method with adaptive parameterization.

There are some limitations to our method. Currently, some starting states in the simulated annealing algorithm cause problems with local minima or long convergence. Developing a method for determining a good starting state for the optimization process could improve convergence.

Most importantly, however, the large compute cost required for multiple iterations of simulations makes a solely wave-based method impractical for high frequencies. At 500 Hz long numbers of iterations can cause convergence only after over a hundred hours. Moreover, metrics such as reverberation time tend to require a larger number of time steps. The large number of time steps additionally makes the technique difficult to parallelize.

In the future, optimization techniques for large outdoor scenes, such as a city block, could be an interesting area of research. This area has many challenges and applications. For example, it would be useful to use acoustic optimization for noise reduction in urban areas. However, large-scale scenes are difficult for wave-based solvers to compute and become computationally intractable at higher frequencies. There are some possible techniques using hybrid geometric and wave-based approaches that have some promise for this sort of application.

Finally, it would be beneficial extend the discrete optimization technique to use a more detailed material model. The ARD method is currently limited to homogeneous environments with a constant speed of sound and does not handle vibration or transmission. Heterogeneous phenomena have an influence on propagated sound, especially on larger scales such as a city block where temperature differences are more apparent. Additionally, the ARD material model uses an approximate mapping to real-world material absorption values. With a more advanced material models we could more accurately represent acoustics on large or complex scenes.

CHAPTER 5: Hybrid Approaches for Noise Minimization and Improving Speech Intelligibility

The drawback of previous ARD-based optimization approaches is the performance of those methods at higher simulation frequencies. At these frequencies, wave-based techniques scale with $O\left(f^4\right)$. In essence, the cost of a single time step increases with the cube of the frequency, but the time step must additionally be proportionally smaller. Although efficient methods exist to parallelize ARD across the spatial domain (see Chapter 3), time domain parallelization methods are difficult.

However, while wave-based methods are computationally expensive for higher frequencies, geometric methods are very efficient. Additionally, geometric methods are more accurate for higher frequencies, where diffraction and scattering techniques are not as prevalent, but less accurate for lower frequencies. These wave effects are prevalent on many architectural scenes (see Figure 5.1). Therefore, in this chapter, we introduce hybrid ARD and geometric simulation methods for the purpose of acoustic optimization. Our hybrid technique uses spectral decomposition, with ARD used for the lower frequencies and a geometric method (Schissler et al., 2014a) for the geometric method.

Previous hybrid algorithms include techniques based on spectral decomposition that use wave-solvers for lower frequencies and geometric methods for higher frequencies (Lokki et al., 2011b; Southern et al., 2011). The computational complexity of these hybrid approaches is dominated by the wave-based methods that are performed over the entire acoustic domain, and current techniques are limited to small acoustic spaces. Different techniques have also been proposed for appropriate coupling at the interfaces between the geometric and numeric methods interfaces (Yeh et al., 2013; Hampel et al., 2008; Wang et al., 2000). Our approach for handling interfaces is also based on these methods.

In the following chapter, we discuss two hybrid acoustic optimization methods: a method for noise control and reduction and a technique for improving speech intelligibility for the purpose of automated speech recognition. Our work on noise control was previously published in (Morales and Manocha, 2017).

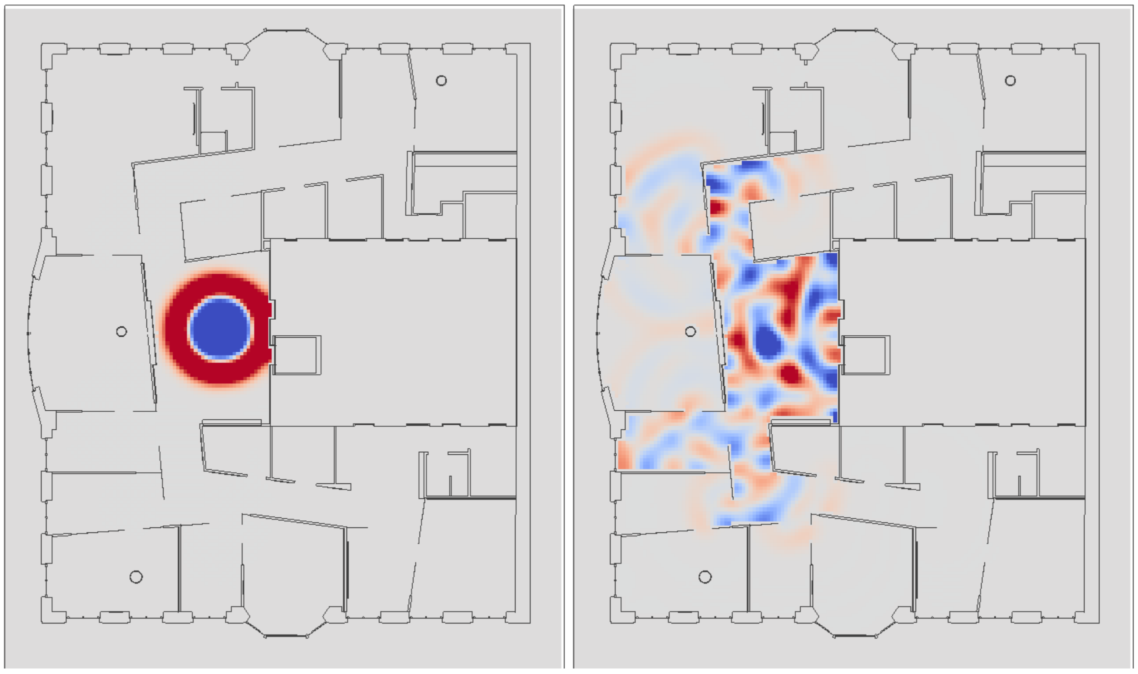

(a) $t = 50$ (b) $t = 150$

Figure 5.1: This figure shows the importance of diffraction (low frequency) effects in sound propagation. It shows the pressure field at two different time steps $t = 50$ ((a)) and $t = 150$ ((b)) of are, red corresponds to positive pressure while blue is negative pressure. In this slice of an office environment, a sound source placed near the elevator hallway propagates through the various office and work areas. Diffraction effects such as this are challenging to compute using geometric solvers but are inherent to wave-based solvers. Prior methods for noise modeling do not take such effects into account.

5.1 Hybrid Propagation

In order to compute the impulse response and various noise or speech intelligibility metrics for a specific source location, we use a novel hybrid sound propagation algorithm. The propagated sound is computed directly from the source sound convolved with the set of impulse responses corresponding to the combinations of available listener locations and source locations.

Using a geometric solver for this computation is fast, but it does not easily capture low frequency diffraction effects. On the other hand, wave-based solvers are expensive for higher frequencies. Moreover, we need to compute the field response for every single source in the environment. This means that during every iteration of an optimization algorithm, we would need to compute the pressure field multiple times. Since wave-based methods scale with $O\left(f^4\right)$, where f is the highest simulation frequency, this process becomes even more expensive for high-frequency sounds.

On the other hand, we can take advantage of geometric sound propagation for higher frequencies where diffraction effects are less prevalent. In large models, ARD is used for all frequencies up to 500 Hz (or 250 Hz for large models where ARD is more expensive), where its computational overhead is reasonable. After computing the results using wave-based and geometric methods for separate frequency ranges, we need to calibrate them.

Our algorithm uses a frequency-domain separation as opposed to a spatial-domain separation where a more accurate wave solver is used for detailed or complex objects (Yeh et al., 2013; Hampel et al., 2008; Wang et al., 2000). Similarly to other frequency-domain separation techniques (Lokki et al., 2011b; Southern et al., 2011), we use a wave-based solver for lower frequencies and geometric for higher frequencies. However, for lower frequencies we use the ARD solver (see Chapter 2 rather than an FDTD solver. This allows us to obtain some performance improvements because the cell size for ARD is much coarser than that of the FDTD solver.

To combine the methods, we low-pass the wave-based impulse response and high-pass the geometric response using appropriate filters. Given a frequency response $R_w(f)$ computed by the wave-based solver up to 500 Hz and a frequency response $R_g(f)$ for the geometric technique, we can determine the hybrid impulse response with the application of a Linkwitz-Riley crossover filter (Linkwitz, 1976):

$$r(t) = \mathscr{F}\left\{B^2_{low}(R_w(f)) + B^2_{high}(R_g(f))\right\}, \tag{5.1}$$

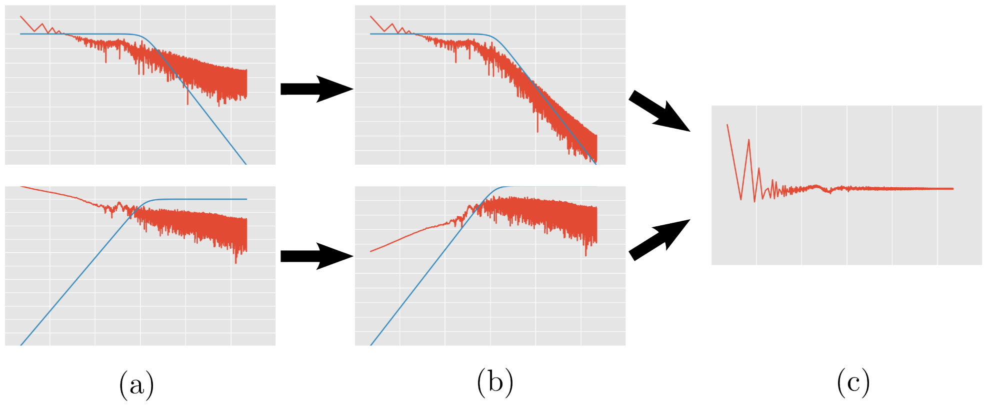


Figure 5.2: **Combining impulse responses using hybrid methods:** (a) The top impulse responses is computed using ARD and the bottom impulse response is computed using a geometric methods. These impulse responses are then filtered using the Linkwitz-Riley crossover filter (Linkwitz, 1976) shown in (b). Finally the filtered IRs are added together to produce (c), which represents the accurate IR over the entire band of the sound source.

where B^2_{low} is the composition of two Butterworth lowpass filters and B^2_{high} is the composition of two Butterworth highpass filters. We use cascading Butterworth filters to help avoid ringing artifacts at the crossover locations. Figure 5.2 shows the details of this computation.

The calibration of the two approaches, based on wave-based and geometric methods, depends on amplitude normalization of the two IRs. We use the direct sound for both these methods for normalization; this corresponds to the part of the sound that reaches a listener directly without reflecting, scattering, or diffraction. Under these conditions, the sound pressure level at a distance d from a sound source is proportional to $\frac{1}{d}$. Given a sound source of power w in watts, our sound pressure at distance d is computed as:

$$p = \sqrt{\frac{w}{4\pi d^2} z_0}, \tag{5.2}$$

where z_0 is the atmospheric impedance value (generally 413 N s m^{-3}). If we pick $w = 1$ for our sound source power in the impulse response, we can compute an impulse response independent of the sound source clip S. While convolving the IR with S later, as long as S has an appropriate scaling for its sound power, we get the propagated sound at the correct loudness values.

5.2 Noise Control and Optimization

A leading problem in health and workplace safety is the effect of environmental noise on workers or building occupants. Environmental noise can have significant impacts on human health (Birgitta et al., 1999; Basner et al., 2014), in addition to having an effect on workplace productivity and child learning ability. The industrial noise associated with manufacturing or production processes is regarded as a major occupational problem (Sequeira and Cortínez, 2016). Additionally, noise level can have a negative impact on the animals (Kight and Swaddle, 2011). In order to deal with these challenges, numerous regulations have been recommended to limit the environmental noise levels.

A key challenge is terms of computer-aided design of indoor structures like architectural models, factories, hospitals, or schools, is to ensure that they satisfy the noise and standards and regulations. Many engineering tasks revolve around optimizing an existing design so that it satisfies different criteria, including acoustic characteristics. In this paper, we address the problem of reducing the interior noise that is generated by different sources (washing machines, HVAC, fans, refrigerators, flush toilets, drains, etc.) in architectural models. The frequencies of these sound sources have a very large range: lower frequencies around $30-50$Hz and high frequencies that are more than $10KHz$. The noise levels are measured in terms of the sound pressure levels (SPL) generated from these sources at different points in the environment. Our goal is to automatically compute the location of these sources so that the resulting SPL is minimized throughout the environment or satisfies the noise standards.

The simplest methods that used for noise prediction enumerate different values of the characteristics of the absorbers or the enclosures in the scene or the source locations, and compute the maximum noise levels for each value. However, such trial and error methods can be very expensive and time consuming and may not be able to capture low frequency noise propagation effects, such as diffraction. Recently, various algorithms have been developed to automatically optimize acoustic characteristics of large models, including modifying shape (Bassuet et al., 2014; Monks et al., 2000), material (Sequeira and Cortínez, 2016; Monks et al., 2000), and topology (Dühring et al., 2008). Furthermore, these techniques can also be utilized to evaluate the noise characteristics. However, current methods either use simplified models based on acoustic diffusion for noise or do not provide sufficient accuracy in terms of modeling the low and high frequency components of noise. As a result, they do not provide sufficient accuracy for large architectural models.

A recent trend in designing large architectural or acoustic spaces is to use acoustic simulation or prediction methods that are sufficiently accurate as well as computationally fast. These simulation methods are frequently used to estimate the noise levels for different source locations or placements as part of overall optimization.There are generally two categories of acoustic propagation simulation: geometric and wave-based. Geometric techniques are based on ray-tracing, beam-tracing, and image source methods (Allen and Berkley, 1979; Funkhouser et al., 1998; Vorländer, 1989; Taylor et al., 2012). These techniques work for higher frequencies and can handle a large number of sources in real time (Schissler and Manocha, 2016). On the other hand, geometric techniques cannot accurately model low-frequency effects such as diffraction or scattering. These effects are often prevalent and noticeable at lower frequencies in terms of noise modeling. While some geometric techniques can represent lower-order scattering effects (Embrechts et al., 2001; Tsingos et al., 2007; Deines, 2008), psycho-acoustic studies have shown that geometric diffraction techniques sound noticeably different from diffraction simulated using wave-based techniques (Rungta et al., 2016). Finally, state of the art geometric diffraction techniques can suffer in geometrically complex scenes (Tsingos et al., 2001). In the second category of acoustic propagation simulation, the wave-based methods, that directly solve the acoustic wave equation using numeric methods. These include finite difference time domain (FDTD) techniques (Bilbao, 2013; Sakamoto et al., 2006), finite element methods (Thompson, 2006), boundary element methods (Ciskowski and Brebbia, 1991; Chen et al., 2010; Sakuma et al., 2014), pseudo-spectral techniques (Hornikx et al., 2010), and domain decomposition techniques (Raghuvanshi et al., 2009b). In general, such wave-solvers can accurately compute the acoustic pressure field. However, their computational cost increases as a fourth power the simulation frequency and current wave-based methods are limited to lower frequencies, approximately less than 1 kHz or 2 kHz.

Main Results: We present a novel acoustic discrete optimization algorithm for source placement in large architectural or CAD models. Our primary contributions for this algorithm are as follows:

- An accurate and efficient hybrid sound propagation algorithm using a Linkwitz-Riley crossover filter for merging low and high frequency bands that can capture low-frequency wave effects such as diffraction while avoiding the cost of more expensive wave-based simulations.
- Source clustering of nearby sources to reduce the optimization search space between 2.5 and 8 times, which is necessary for efficient computations on scenes with a large degree of freedom for placement of sound sources.

- An efficient discrete optimization method for optimizing source placement that uses impulse response caching to improve convergence, reducing the effective algorithmic complexity of the heuristic optimization algorithm from worst-case $O(m!)$ to $O(m)$.

Our formulation performs discrete optimization to select from a finite number of possible positions for each position. The source positions are initially clustered to reduce the discrete search space. Then, in order to accurately compute the pressure field, we use a hybrid acoustic simulator that combines geometric and wave-based methods to take into account all source frequencies, including low and high frequencies. Our optimization models the environmental noise levels using an A-weighted curve model and computes the noise based on the impulse responses. Our hybrid acoustic optimization algorithm uses both geometric and wave-based techniques: geometric for the middle and high-frequencies of the simulation, and wave-based for the low-frequency bands. Thus, we gain the advantage of wave-based simulation at frequencies where diffraction effects are noticeable, but maintain computational efficiency of the algorithm overall by using geometric techniques for the higher frequencies. Additionally, we used a modified simulated annealing algorithm that caches the impulse responses of source-listener configurations for efficient optimization convergence.

We highlight the performance of our method on different CAD benchmarks that mimic a variety of real-world workplaces like offices, warehouses, and industrial zones. These include industrial locations, where machinery can cause loud noise levels, and commercial office locations, where HVAC and light machinery noise can affect workplace productivity. To the best of our knowledge, this is the first algorithm that can optimize the source location in large CAD models to minimize the noise levels in selected regions.

5.2.1 Prior Work

The problem of environmental noise reduction is a cross-disciplinary field with literature spanning across various fields from ecology (Kight and Swaddle, 2011) to urban planning (Kang, 2006) to computer-aided design and acoustics. Additionally, various regulations and recommendations are in place to control environmental noise including World Health Organization (WHO) recommendations (Birgitta et al., 1999) and other recommendations from regulatory agencies in various countries, including the Occupational Safety and Health Administration (OSHA) in the United States (OSHA, 2008).

There is considerable literature on the measurement of environmental noise. This literature includes various works on environmental noise in certain geographic regions (Jamrah et al., 2006; Piccolo et al., 2005; Zannin et al., 2002) and the assessment of the impact of noise reduction on production lines (Xue et al., 2011). Ondet and Barbry predict noise in a room using ray-tracing simulation techniques (Ondet and Barbry, 1989). Keränen et al (Keränen et al., 2003) discuss the accuracy of geometric method for noise computation. Sequeira and Cortínez developed a method for optimizing the acoustic treatment of a structure using a simplified acoustic diffusion model (Sequeira and Cortínez, 2016). Further literature includes discussions on environmental noise in urban environments (Kang, 2006; Bucur, 2007).

Various work has been done in the field of geometric sound propagation, including work that includes some scattering and diffraction effects. Some of these works include Tsingos et al. (Tsingos et al., 2007), Embrechts et al. (Embrechts et al., 2001), and the dissertation of Deines (Deines, 2008). Various geometric room acoustic techniques are discussed in a survey by Savioja (Savioja and Svensson, 2015). However, geometric techniques for diffraction and scattering cannot easily represent higher orders of diffraction which can be prevalent at higher frequencies. Rungta et al. (Rungta et al., 2016) show that participants in their study could observe a noticeable difference in propagated sound in up to three orders of diffraction when comparing a UTD-based geometric approach and a wave-based approach.

The problem of speaker and microphone placement is closely related to our optimization goal. Speakers in this case are analogous to the location of noise-emitting machinery, while microphones correspond to the location of workers or patients in a hospital. As such, there have been various works in the problem of optimal speaker and microphone placement. Khalilian et al. (Khalilian et al., 2015) use methods for optimal speaker placement for sound field reproduction (SFR) problems. D'Antonio et al. developed an algorithm for optimizing speaker placement with constraints in addition to acoustic treatment and room dimensions for home theater systems (D'Antonio and Cox, 1997). Other techniques include using genetic optimization techniques for optimal loudspeaker and microphone placement (Montazeri et al., 2003).

5.2.2 Computing Noise Exposure

There are a variety of methods for computing environmental noise, but most regulatory agencies use weighted decibel levels (known as A-weighting, B-weighting, or C-weighting) and time-weighted average (TWA) metrics (OSHA, 2008).

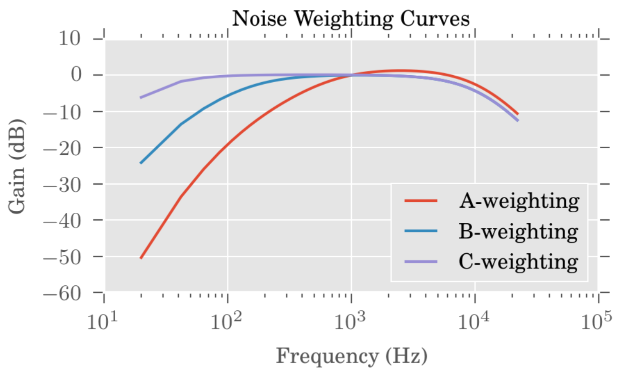


Figure 5.3: The A, B, and C-weighting curves. These are used to account for the loudness of a sound perceived by a human being. A-weighting is commonly used for noise measurements although C-weighting is often desirable because of its flat curve. In our noise modeling results, we used the A-weighting, though it is possible to use other curves as well.

Weighted sound level curves are designed to account for the loudness perceived by human hearing. There are usually three curves: A, B, and C. A-weighting is the most commonly used curve by equipment manufacturers and regulatory agencies, while C-weighting is often used when a flat frequency response is desired (Berger, 2003). B-weighting is mostly useful for higher sound pressure levels, but is rarely used nowadays (OSHA, 2008). Figure 5.3 shows the gain of each of these curves across the range of human hearing. In our work, we focus on the A-weighting curve. It is the most popular and is also standard in many regulations.

To determine the averaged A-weighted sound level from recorded or simulated sound, we compute the average sound level in each frequency band f. The formula for the average A-weighted sound is as follows:

$$L = \log_{10} \sum_{f} 10^{\frac{P(f)+A(f)}{10}} \text{ dB}, \tag{5.3}$$

where $P(f)$ is the sound pressure level in decibels of the frequency band corresponding to f and $A(f)$ is the A-weighting adjustment at frequency f. The A-weighting adjustment is given by (IEC, 2013):

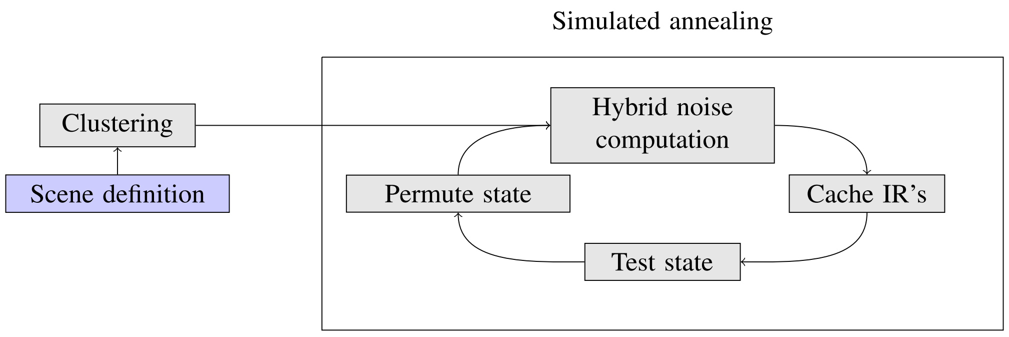


Figure 5.4: **Our optimization algorithm:** The input to the algorithm is the scene representation (i.e. the triangulated CAD model), which includes possible source locations and the listener regions. The state corresponds to the current set of source positions. We permute between them to move to the next set of positions. We perform source cluster before simulated annealing algorithm. The various components of simulated annealing including noise computation, IR caching, testing states for acceptance based on metrics, and permuting the states to get the neighbor states. Section 5.2.2 covers the details of how noise is computed.

$$\begin{aligned} U(f) &= (f^2 + 20.6^2)(f^2 + 12194^2), \\ V(f) &= (f^2 + 107.7^2)(f^2 + 737.9^2), \\ R(f) &= \frac{12194^2 f^4}{U(f)\sqrt{V(f)}}, \\ A(f) &= 20 \log_{10} R(f) + 2.0 \end{aligned} \tag{5.4}$$

In frequency space, the final adjusted environmental noise level per frequency band is:

$$L(f) = rS + A\,\mathrm{dB}, \tag{5.5}$$

where A is an adjustment on the final convolved sound pressure level.

5.3 Optimizing Source Placement for Noise Minimization Using Hybrid Acoustic Simulation

In this section, we describe the overall source placement algorithm for noise optimization. In terms of optimization, we are interested in the minimization of noise at a set of n listener positions $\ell_1 \dots \ell_n$. These listener positions are located in the areas where noise can be a problem, such as a work location or a hospital bed. The noise in the listener positions are induced by a set of m sound sources $s_1 \dots s_m$. In our optimization formulation, we are interested in the placement of the sources according to constraints such that the noise reaching the listener positions is minimized. Given this goal, we can derive the following objective function for our optimization process:

$$\arg\min \left(\max_i \sum_j L\left(\ell_i, s_j\right) \right), \tag{5.6}$$

where $L(\ell_i, s_j)$ is the averaged A-weighted level across all frequency bands induced by a sound source s_j on the listener position ℓ_i. We sum each of these A-weighted levels to get the total noise for all frequency bands at listener ℓ_i, generated by all the sound sources. We are particularly interested in the listener position with the maximum noise level. While the maximum operation could be replaced by an average or other form of statistical analysis, we are interested in this formulation of the problem that ensures that that the noise at any location in the environment is not too high, and below the guideline. This is useful in a situation in which an acoustic designer must conform to a set of regulations in which no part of the work environment can exceed safe noise levels.

One major issue is that the evaluation of L is non-trivial and there is no easy closed-form solution. This is because it involves evaluating both the geometric and ARD solver one or more times. The geometric solver is used to compute the response at the higher frequencies and the wave-solver is used to compute the response at the lower frequencies. Therefore, in our algorithm, we explore ways to efficiently compute the result. Typically, the complexity of the propagation algorithms is a linear function in the number of sources. In order to perform efficient computations, we use three techniques:

- We use sound source clustering to reduce the optimization search space.
- We use efficient simulated annealing approach for optimization that basically performs faster search for a solution.
- We introduce impulse response caching to accelerate iteration time for the optimization.

The combination of these three techniques can considerably improve the runtime performance and makes it possible to optimize the source positions in large CAD scenes for noise minimization. Figure 5.4 summarizes these elements in the context of our overall algorithm. Next, we describe each of these techniques in detail.

5.3.1 Source configuration

The possible locations of sound sources in our algorithm can be defined by the configuration space C. This is a subset of $\mathbb{R}^3$ where user-defined constraints and collision constraints are enforced on the sound source locations (e.g. the sound source cannot be in a wall). The goal of the optimization algorithm is to find the source positions s_i within the configuration space where the objective function (equation 5.6) is minimized. However, the user and collision constraints must be adhered to during the optimization process.

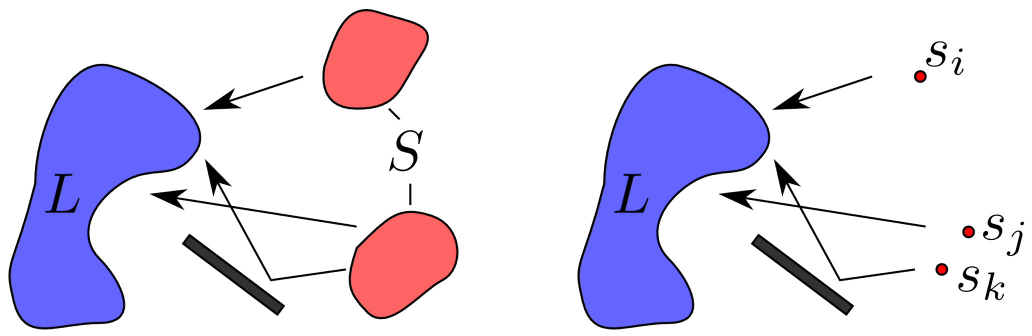
Initially, we only enforce the user constraints. This reflects the natural constraints of placing the machinery or the equipment, such as fixtures or other structural or utilitarian constraints. For example, in a home, a washing machine can only be placed on a specific washing machine fixture where it is connected to the water supply. Similarly, there are fixed wiring locations for the fans or HVAC. In an industrial context, a generator may need to be placed near machinery operated by a worker. The user specifies these constraints using a plugin to the popular open source modelling tool, Blender.

The space defined by these initial constraints is discretized into a set of points S. These points $s_i \ldots s_m$ define the possible locations where a source can be placed and are determined by a stratified sampling of the configuration space C. The sampling density ρ is empirical in our algorithm, but should be chosen densely enough such that features of the configuration space are well sampled.

Collision constraints are then enforced next. We use the low-frequency spatial discretization of the ARD method to remove possible source positions that are within wall cells. This ensures consistency between the wave-based and geometric methods. Wall cells are determined by a flood-fill algorithm once the CAD model geometry is voxelized in the ARD pre-processing. An ”air” voxel is marked by the user and any voxel not reachable from that point is considered to be a wall.

5.3.2 Sound Source Clustering

One consequence of this is that the size of the search space for the optimization problem can very large. This can yield many iterations of the optimizer, and each iteration requires a full evaluation of both geometric and ARD per source-listener pair. This is very slow as a result of using the more accurate wave-based solver, which can take a few minutes to solve a single instance of the source-listener problem. Therefore, we find that the running time of our algorithm is heavily dependent on the total volume of constraint regions.



(a) Original configuration

(b) After clustering

Figure 5.5: Clustering allows us to represent large areas of possible source locations by a single (virtual) source location. Our clustering approach is guided by the fact that the noise difference between the representative source and the other sources in the cluster is less than a threshold. In our experiments, we used a threshold of 1 dB, or approximately the just-noticeable difference (JND) for the human auditory system. JND is governed by psycho-acoustic studies.

One way of ameliorating this problem is by exploiting the property that some neighboring samples inside these constraint regions have essentially the same A-weighted level in terms of the noise characteristics. This can be evaluated using a free-field region of space, i.e. one with no obstacles. Two sound sources of equivalent distance and acoustic characteristics, barring interference, will result in the same sound pressure level. This criteria is explained in Figure 5.5, where source regions are replaced by representative sources according to the clustering algorithm.

Geometrically, it is difficult to determine which portions of the model actually correspond closely to this free-field condition from a given CAD model. This is a consequence of the complexity of interactions between the sound waves, surfaces, and the obstacles. However, we can use sound propagation algorithms to determine which sample locations have similar sound levels. Using geometric sound propagation for this computation is appealing for two reasons. First, that geometric sound propagation is orders of magnitude faster than wave-based techniques and the current methods can handle large scenes in tens of millisecond. Secondly, it provides an accurate sound pressure level for higher frequencies. Although low-frequency wave phenomena are important, metrics such as A-weighting weight lower frequencies less.

Therefore, we evaluate the sound pressure level from each sample point and implement hierarchical clustering for the sample points. The threshold for hierarchical clustering is set to be a loudness value that is an acceptable error in the optimization process. In other words, for two sources s_j and s_k, they can be clustered together if they are near each other and

$$\left\| \max_i L\left(\ell_i, s_j\right) - \max_i L\left(\ell_i, s_k\right) \right\| < \tau, \tag{5.7}$$

where the threshold τ is generally the just-noticeable difference (JND) for human hearing. This clustering allows us to select representative samples for each constraint region. In the case where the sound pressure level for multiple sound sources in the constraint region changes greatly, we use a separate representative sound source for each cluster.

As a result of this clustering, we significantly reduce the search space for our optimization algorithm. The amount depends on the location of sources, but we experienced a reduction on some of our benchmarks by about 5x to 7x for a clustering threshold of 1 dB (approximately the just-noticeable difference of human hearing). This reduction improves the convergence rate and performance of our algorithm.

5.3.3 Simulated Annealing

```
Input  : Listener regions $\Omega$
         Initial temperature $T_0$
         Cooling rate $\alpha$
Output : Optimal source locations $q$
initialize simulator;
$\vec{s} \leftarrow$ ComputeInitialState()
$q \leftarrow \sum L_{max}(\Omega, s_i))$;
$T \leftarrow T_0$;
while $T > 1$ do
    $\vec{s} \leftarrow$PermuteState($\vec{s}$) /* Compute new state                    */
    $q' \leftarrow \max_i \sum_j L\left(\ell_i, s_j\right)$;
    if TestState($q$,$q'$,$T$) then
        $q \leftarrow q'$;
    end
    $q \leftarrow$Min($q$,bestState);
    $T \leftarrow \alpha T$;
end
Procedure TestState($q$,$q'$,$T$)
    if $q' < q$ then
        return true;
    end
    $p \leftarrow \exp \frac{q-q'}{T}$;
    return $p <$ Rand(0,1);
```

Algorithm 2: The simulated annealing algorithm for noise minimization. Each iteration of the algorithm involves computing the acoustic pressure field using both the geometric acoustic solver and ARD.

After sound source clustering, we end up with a set of discrete locations that a sound source s_j can be placed. This is fundamentally a combinatorial problem and implies the need for a discrete optimization approach. The objective function in Equation 5.6 can therefore be minimized by using a simulated annealing approach. This technique is efficient for large search spaces — even after clustering, there can often be on the order of millions of total combinations of source locations. Brute force solutions for search spaces of this size are completely intractable. Even at an optimistic minute per iteration for a naive implementation an exhaustive search would take years.

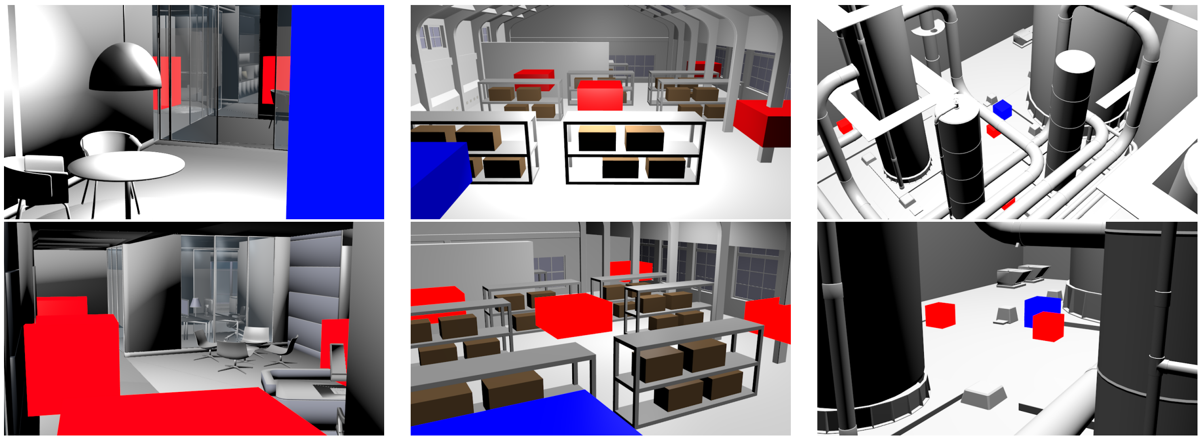
The advantage of simulated annealing is that it can avoid local minimum in the search for a global minimum. It minimizes an objective function by randomly permuting the state, accepting new states that have lower energy. Additionally, in order to avoid local minimum, the annealing algorithm will sometimes also accept less optimal states. The probability of this happening depends on the system temperature, T. Each iteration, this temperature decreases by a factor of the cooling rate, α. As the system cools, the algorithm is less likely to select less-optimal states.

5.3.3.1 Simulated Annealing State

The state variable $\vec{s}$ represents a list of all the sound sources in the scene. The end goal of our optimization algorithm is to determine a state $\vec{s}$ such that the maximum noise at all listener positions is minimized. Algorithm 2 shows how the state is iterated on to yield a global minimum. First, the state is assigned through a random shuffle (in `ComputeInitialState`). The state variable gets permuting every iteration by `PermuteState`. The energy q' of this new state is then used to determine whether the new state is accepted or rejected. In our algorithm q is simply the maximum noise value across all listener positions, or $\max_i \sum_j L\left(\ell_i, s_j\right)$. In simulated annealing a state q' that is better than the old state q is always accepted. If the converse is true, the new state may still be accepted depending on the temperature T.

5.3.4 Impulse Response Caching

Simulated annealing provides an efficient heuristic for minimizing in a discrete search space, even if the search space is combinatorial in nature. However, even for a reduced number of iterations computing the impulse responses using the ARD wave-based solver is expensive.



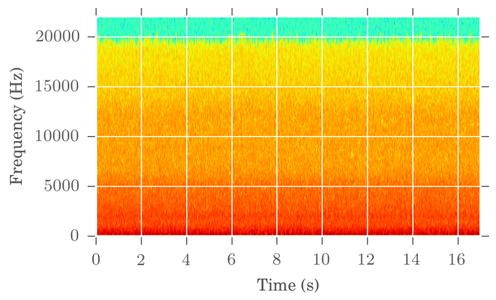
(a) The office scene (b) The warehouse scene (c) The industrial scene

Figure 5.6: We have evaluated our algorithm on three complex CAD benchmarks: the office scene, the warehouse scene, and the industrial scene. All three benchmarks represent different environments in which one would want to minimize noise levels. In this figure, the blue rectangular regions represent our listener regions; we sample from this region in order to get source-listener pairs. Additionally, the red regions represent possible sound source locations. For example, in the office scene some of the source locations represent areas in which an air conditioning duct could be placed. We also sample from these locations, clustering the samples in order to reduce the total number of possible source locations to a representative set.

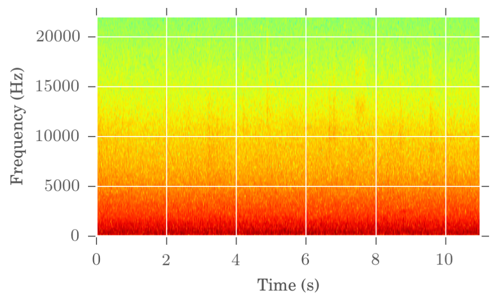
For each simulated annealing iteration, our algorithm must compute the impulse response between each source and listener pair in the state. It's worth considering that for n listeners and m sources, the total number of these pairs is $O(nm)$, much less than the size of the total search space which is $O(m!)$.

As a direct consequence, we can cache source-listener pairs that already have a computed IR. Instead of redundantly computing an already computed IR, we retrieve it from the cache. This significantly reduces the cost of later iterations of the simulated annealing algorithm, particularly since the wave-based solver would otherwise have to be executed once for each non-cached pair that is in the current iteration state.

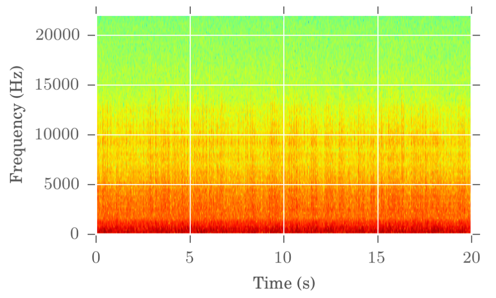
Even though the search space is still $O(m!)$, exploring all source-listener pairs means that no more impulse responses need to be computed. Thus, our algorithm can be stated as essentially having $O(nm)$ asymptotic time given that the iteration time for cached impulse responses approaches 0. However, in practice, since the ARD method is significantly more expensive than the geometric technique and dominates our computation time, the asymptotic time can be further reduced to $O(m)$. This is because ARD computes a global solution to the wave equation, so the running time is not dependent on the number of listener positions.



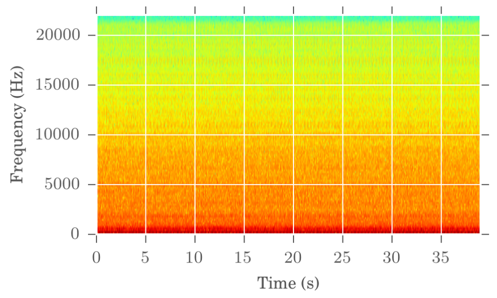
(a) HVAC



(b) Machine



(c) Heavy machine



(d) Generator

Figure 5.7: The spectrograms of various sound sources that we used in our benchmarks and evaluated their noise effects. The red color represents higher amplitude in those frequencies at that point in time. Many of these sound sources have a wide range of frequencies; for example the HVAC sound source has contains very high and very low frequencies. We are able to optimize across this large frequency range because our hybrid sound propagation algorithm can easily deal with both low-frequency and high-frequency effects.

5.4 Results of Noise Minimization

We have evaluated our algorithm on three different and complex CAD environments: an office, a warehouse, and an industrial zone. These scenes were obtained from existing model databases such as TurboSquid that include some architectural models and were adjusted so that the scaling and size were appropriate and would be similar to CAD models used in designs for noise analysis. In addition, for each model, we determine a set of listener locations and the possible source locations.

Along with the CAD models, we used a variety of sound clips to represent the variety of sound or noise sources that might occur in an industrial or workplace environment. The spectrograms of these sources are shown in Figure 5.7. Table 5.1 shows a summary of the scenes and the sound sources used in these models. Additionally, the table shows the advantage of our clustering algorithm: the clustering ratio for each scene represents how much the number of source locations was reduced by. For example, the Warehouse scene initially had 25 possible source locations, with a sampling distance of 0.7 meters, but after clustering had

Scene	Sound Sources	Noise level	Avg. source region size	Sampling density	Num. source regions	Clustering ratio
Office	1x machine (70 dB), 5x HVAC (50 dB)	38.48 dB	$1 \times 1 \times 1$ m	0.4	11	7.55x
Warehouse	3x heavy machine (85 dB)	59.79 dB	$2 \times 2 \times 1$ m	0.7	25	2.5x
Industrial	5x heavy machine (85 dB), 2x generator (90 dB)	50.14 dB	$1.5 \times 1.5 \times 1.5$ m	0.7	104	8.0x

Table 5.1: We highlight the benchmark scenes, their sound sources, the minimized noise level, and the clustering ratio. In some cases we used multiple types of the same source. For example, in the Industrial scene we optimized the placement of two generators. The clustering ratio is used to evaluate the performance of our clustering algorithm. The higher value of the clustering ratio indicates greater reduction in possible source positions. For example, a clustering ratio of 5.36X implies that number of sources was reduced to approximately $\frac{1}{5}$ of their original number.

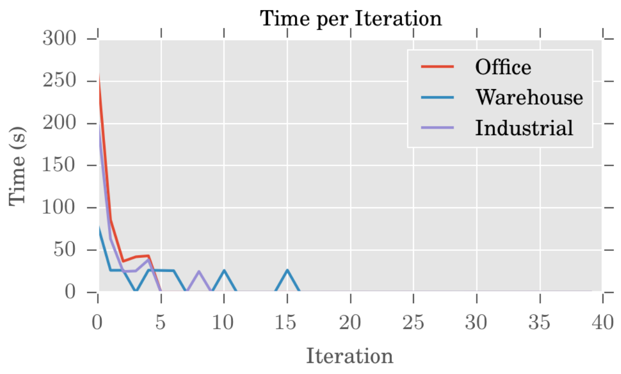


Figure 5.8: The time spent during each iteration for the three benchmarks. Importantly, as the number of iterations increase, it is more more likely that all of the listener-source pairs corresponding to the current state are already cached. This helps improve the running time of the simulated annealing algorithm even if the number of iterations is large. In this figure, iterations after step 10 do not need to calculate new impulse responses and just use the cached values.

a 2.5 times reduction to yield 10 source locations. This considerably improves the performance of sound simulation and sound propagation algorithms.

5.4.1 Performance

One key benefit of our algorithm is the efficiency in optimization using clustering and IR caching. This dramatically improves the convergence and runtime perform of our acoustic optimization algorithm. In Figure 5.8, we demonstrate how the IR caching algorithm improves our iteration time. Without caching, the iteration time would be roughly constant. Instead, we observe that as the number of iterations increase, we are more likely to have covered all of the $O(mn)$ combinations of source and listener pairs.

Scene	Crossover Frequency	Number Tris	Number cells	Total time
Office	500 Hz	973373	2.5m	1398.02 s
Warehouse	500 Hz	21188	3.7m	1254.84 s
Industrial	250 Hz	51802	14.3m	4232.0 s

Table 5.2: The total running time of each scene along with the number of cells used for the ARD algorithm and the number of triangles in the CAD models corresponding to our benchmarks. Additionally, the crossover frequency for the hybrid simulation is listed. The most important factor int he running time is the number of cells, since the ARD wave-solver is the main bottleneck in the algorithm. The geometric sound propagation based on ray tracing only takes a few tens of milliseconds performance. Due to the use of IR caching and source clustering, we considerably reduce the runtime performance.

Additionally, Table 5.2 shows how our algorithm behaves on different types of benchmarks. We compute the total running time of the algorithm over all iterations, e.g. we use 100 iterations in our simulated annealing process. The primary factor in performance is the number of grid cells. This corresponds to the number of voxels that ARD uses for its regular grid when solving the wave equation. On some scenes, such as office, we had to artificially use a higher number of grid cells than required by the stability conditions of ARD (Raghuvanshi et al., 2009b) to generate highly accurate results.

On the other hand, the number of triangles in the scene has little effect on the wave-based algorithm since operates on a grid, not a triangle mesh and the grid size is governed by the maximum simulation frequency. The complexity of the mesh does affect the performance of the geometric solver, but this is negligible compared to the time required for the ARD solver. The geometric solver is based on ray tracing, that uses a bounding volume hierarchy to accelerate the computation. Its complexity is logarithmic in the number of triangles.

5.4.2 Noise Minimization

Using our algorithm, we can pick the best source locations that will minimize the noise for a set of listener locations. Figures 5.9, 5.10, and 5.11 show the full-field noise levels for the entire domain as a result of our optimization process. In the figures, we show both the regions that contain different listener positions in addition to the source location regions. Our algorithm gets quite close to optimal noise levels in these regions.

Additionally, we show result on how the noise field has changed. Figure 5.12, 5.13 and 5.14 show how optimization changed the noise distribution in the scene.

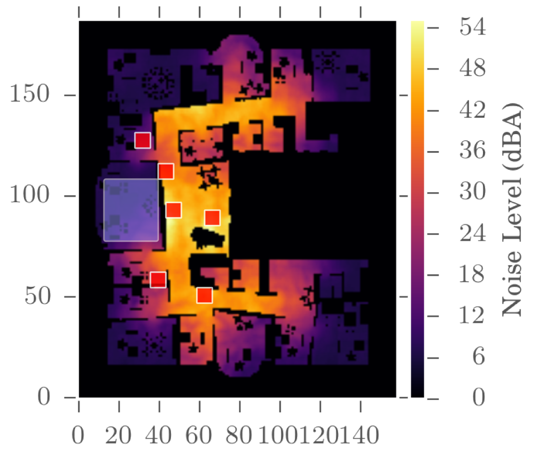


Figure 5.9: Noise field slice for the office scene calculated with ARD using the impulse response at every location. The noise field is minimized in our listener region (blue). The red regions are areas where sources can be placed. Each red area can have multiple source locations depending on the clustering.

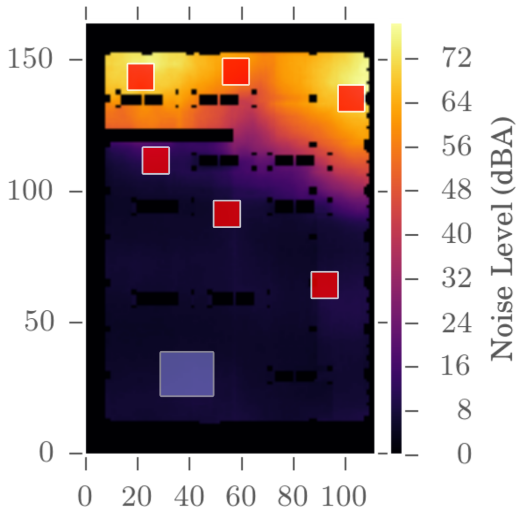


Figure 5.10: Noise field slice for the warehouse scene calculated with ARD using the impulse response at every location. The noise field is an output of our algorithm, where the goal is to minimize the listener region (blue). The maximum noise level in the blue region is used for computing the energy for the simulated annealing process.

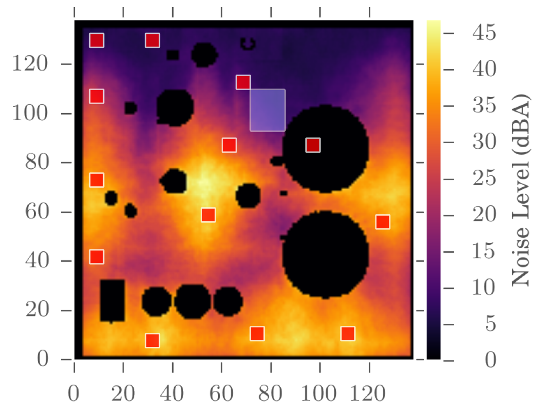


Figure 5.11: Noise field slice for the industrial scene calculated with ARD using the impulse response at every location. The minimization region, where the listener positions are, is shown in blue, while the source constraints are shown with red markers.

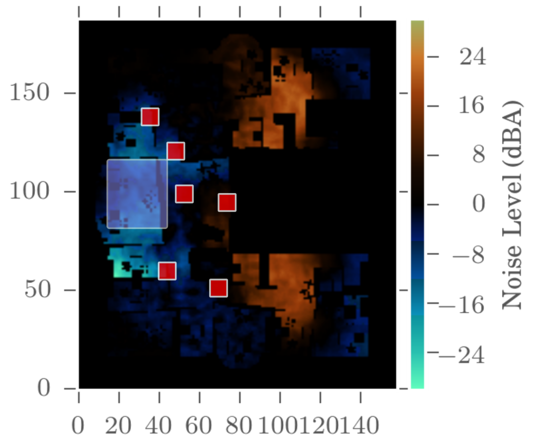


Figure 5.12: Change in the noise field for the office scene. Blue areas are regions in which the noise was reduced and orange areas are regions in which the noise was increased as a result of the changes. Note how the listener positions (the blue box) are located in the blue regions.

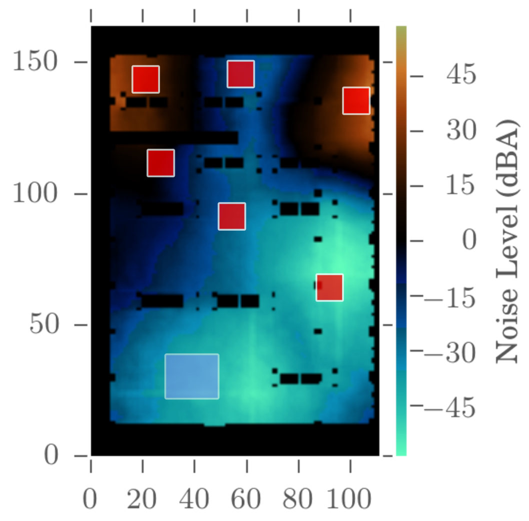


Figure 5.13: Change in the noise field for the warehouse scene. After optimizing, the noise in the blue regions is minimized. Note that the listener locations are in the blue regions.

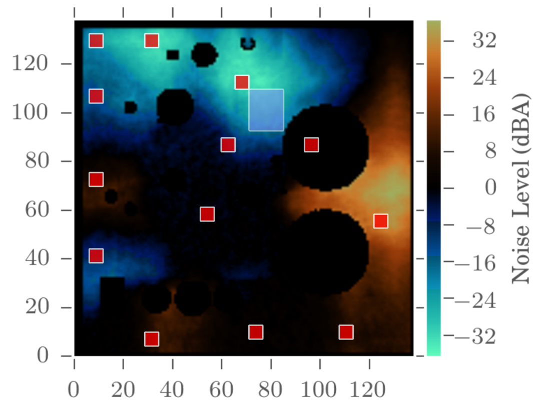
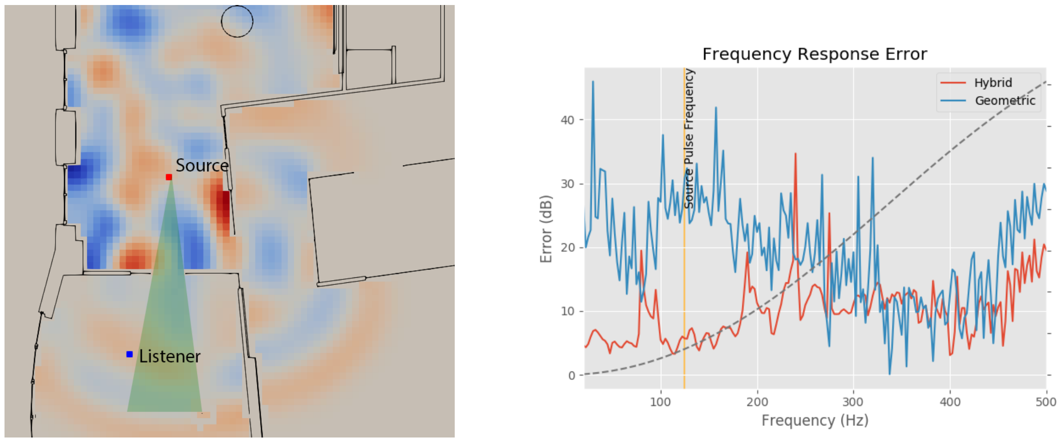


Figure 5.14: Change in the noise field for the industrial scene. Blue areas are regions in which the noise was reduced and orange areas are regions in which the noise was increased as a result of the changes. As a result of our algorithm, the noise level in the listener regions (the blue box) was reduced.



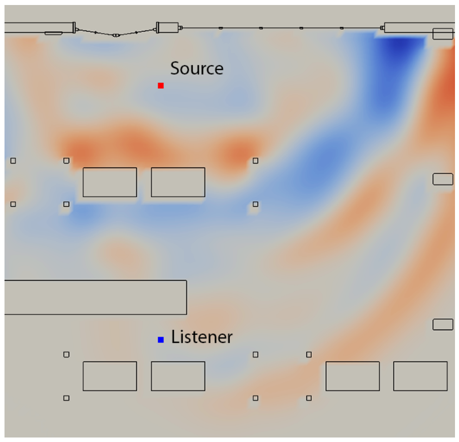
(a) Experimental Setup (b) Frequency Response Error

Figure 5.15: Error comparison with a high resolution numerical simulation on the office scene. (a) shows the experimental setup, with the green region denoting the direct sound outside of which the geometric approach cannot accurately compute. In (b) we compared an impulse response generated by a Gaussian derivative source with a primary frequency of 125 Hz. The yellow line indicates the source frequency while the dotted gray line indicates the transition between the numerical and geometric components of the hybrid approach. The critical frequency of this filter is at 500Hz. In the frequency band centered around the source frequency (the 88Hz to 177Hz band), we noted an average 6 dB error for the hybrid approach, which, while noticeable, is small. On the other hand the geometric approach yielded an average 27 dB error, which is significant.

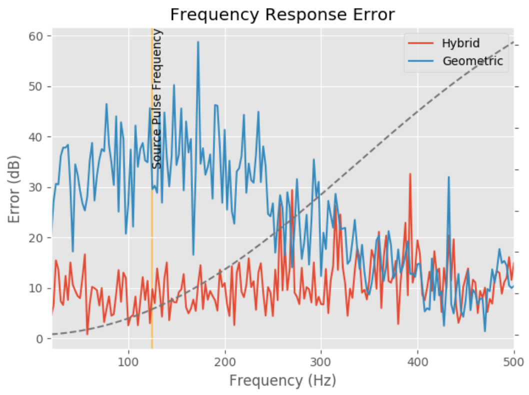
5.4.3 Error Analysis

In addition to performance and noise minimization results, we analyzed some of the error properties of our hybrid simulation approach. We chose the office scene for the first analysis because of its complex geometry and the prevalence of diffraction effects where geometric approaches struggle. Figure 5.15(a) shows our experimental setup. Additionally, we conducted an experiment on the warehouse scene, where the presence of many shelves makes diffraction effects important. The setup for this scene is shown in Figure 5.16(a).

Figure 5.15(b) and Figure 5.16(b) show the error of the frequency responses of the hybrid and geometric approaches induced by a Gaussian derivative source with a primary frequency of 125 Hz. The hybrid approach in this case used a critical frequency of 500 Hz to transition between wave-based and geometric components. In the office scene, in the octave frequency band centered around the source, the hybrid approach had an average error of only 6 Hz, while the geometric approach had a much more significant error of 27 dB where it did not compute the diffractive properties of the propagated sound. Similarly, in the warehouse scene, the hybrid approach had an average error of 8 dB on the frequency band, while the geometric approach had a higher error of 36 dB error.



(a) Experimental Setup



(b) Frequency Response Error

Figure 5.16: Error comparison with a high resolution numerical simulation on the warehouse scene. (a) shows the experimental setup. In (b) we compared an impulse response generated by a Gaussian derivative source with a primary frequency of 125 Hz. The yellow line indicates the source frequency while the dotted gray line indicates the transition between the numerical and geometric components of the hybrid approach. The critical frequency of this filter is at 500Hz. In the frequency band centered around the source frequency (the 88Hz to 177Hz band), we noted an average 8 dB error for the hybrid approach, which, while noticeable, is small. On the other hand the geometric approach yielded an average 36 dB error, which is significant.

5.5 Minimizing Noise in Speech Recognition Applications

One of the primary challenges in Automated Speech Recognition (ASR) applications is recovery of the spoken source signal. In indoor environments, noise can be introduced by secondary sound sources (such as an air conditioning unit, outdoor traffic, or other voices and devices like televisions), by environmental sound propagation effects such as diffraction and reverberation, and the characteristics of the receiver microphone. In particular, reverberation can have a detrimental effect on speech recognition algorithms. Its impact on ASR algorithms has been studied extensively (Gillespie and Atlas, 2002).

The Speech Transmission Index (STI) is a common metric for evaluating the intelligibility of spoken audio (EN, 2011), and regarded as an accurate subjective measure for human recognition of speech (Galster, 2007). STI is negatively impacted by reverberation and noise, and thus serves a useful metric in the evaluation of the intelligibility of a propagation environment and source/receiver configuration. In this paper, we address the problem of computing an optimal placement of the receiver that minimizes the noise due to sound propagation, environmental effects, and secondary sources in order to maximize the STI.

Prior work on ASR techniques has focused on denoising and dereverberation filters on the incoming audio on the receiver in order to reduce extraneous noise (Tashev and Allred, 2005). Many of these approaches are based on machine learning techniques for noise minimization (Feng et al., 2014), and may need a considerable amount of training data. However, these methods have some limitations. While they can reduce the effects of propagated noise, denoising filters are limited in their applicability to sound propagation paths where the signal to noise ratio (SNR) of the incoming audio is not sufficient to recover the original signal. For example, speech from an adjacent room may only be propagated to the receiver by indirect paths through transmission, diffraction, or reflection. Although previous works have incorporated dereverberation techniques to reduce some of these problems, they often use approximation of reverberation time or decay rate that may not capture the acoustic characteristics of complex environments such as multi-room apartments or offices.

Main Results: We present a novel algorithm for receiver placement using sound optimization. Our optimization algorithm maximizes the STI at a receiver by relocating it based on sound propagation characteristics of an indoor environment. We use a hybrid sound simulation technique for accurate sound propagation, computing the Room Impulse Response (RIR) with wave-based sound simulation techniques at lower frequencies for accuracy and geometric propagation techniques at higher frequencies for performance. We present our optimization algorithm for computing the ideal location for maximizing the STI in Section 5.6. Additionally, using our algorithm, we are able to significantly improve the speech intelligibility at the receiver and minimize the impact of noise and reverberation. We highlight the performance of our algorithm on indoor office and residential scenes (see Section 4.6), where we show an improvement of the STI up to 71% on some scenes.

5.5.1 Prior Work

The impact of reverberation and noise remains one of the primary challenges in terms of designing ASR algorithms. There has been extensive work both on identifying the impact of various noise sources and ways of reducing the effect of noise and reverberation on speech recognition algorithms. Various benchmarks exist to study the effect of noise on speech recognition (Hirsch and Pearce, 2000). Gillespie et al. (Gillespie and Atlas, 2002) study the effect of reverberation time (RT60) on ASR algorithms. They found that even moderate reverberation causes a catastrophic decrease in the performance of speech-based systems.

Different techniques have been proposed to reduce the impact of reverberation effects on speech recognition. Tashev and Allred (Tashev and Allred, 2005) use a multi-band decay model for real-time reverberation, but are unable to accurately model all the sound propagation paths. Ko et al. (Ko et al., 2017) use image-

t	time	f	frequency
r	room impulse response	R	room freq. response
f_m	modulation freq.	SNR_k	SNR in band k

Table 5.3: Notation and symbols used in our acoustic solver and optimization algorithm.

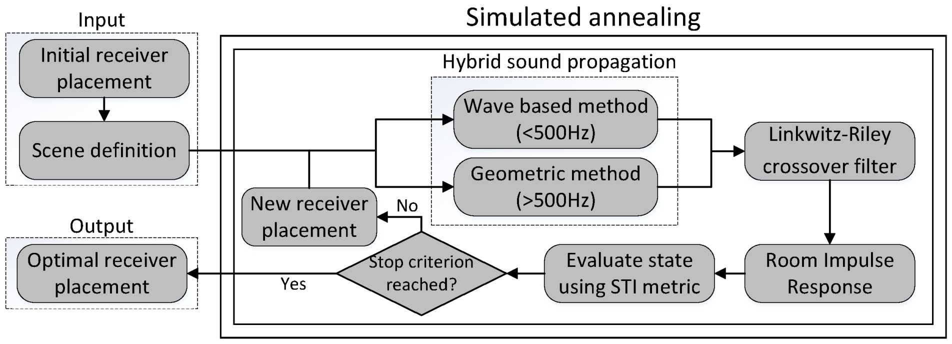

Input
Initial receiver placement
Scene definition
Output
Optimal receiver placement
Simulated annealing
Hybrid sound propagation
Wave based method (<500Hz)
Geometric method (>500Hz)
Linkwitz-Riley crossover filter
New receiver placement
No
Stop criterion reached?
Yes
Evaluate state using STI metric
Room Impulse Response

Figure 5.17: We highlight various components of our approach. The Room Impulse Response is computed using a hybrid sound propagation approach, with a wave based technique under 500 Hz and a geometric technique above 500 Hz. The STI is then computed from the RIR and averaged in order to evaluate the objective function. Our simulated annealing approach then computes a new receiver location for the next iteration until the stopping criteria are encountered.

source methods for computing an RIR that more accurately captures reverberation, but is limited to specular reflections. Feng et al. (Feng et al., 2014) use machine learning techniques to filter out noise and reverberation effects on incoming signals. Our technique is complementary to these methods and we use an accurate impulse response computation method instead. Palomaki et al. (Palomäki et al., 2004) attempt to improve the performance of ASR algorithms by mimicking the binaural properties of human hearing. Chen et al. (Chen et al., 2016) use machine learning techniques to isolate speech features in order to improve the effectiveness of hearing aids.

5.6 Receiver Placement Optimization

Our algorithm is composed of three major components: sound propagation given a specific receiver placement, evaluation of the optimality of the placement, and the selection of a new placement to evaluate. Notation throughout this section is referenced in Table 5.3.

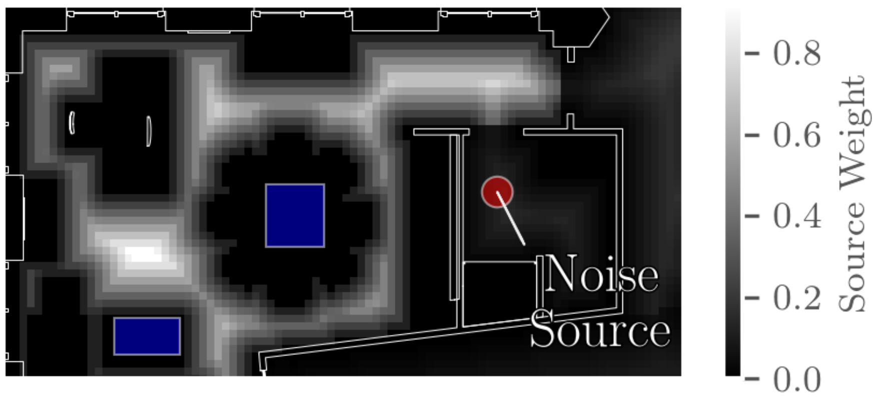


Figure 5.18: Diagram of the placements constraints used in the discretization of a portion of the Office scene. Blue regions correspond to allowable areas within which the listener can be placed. The gradient corresponds to possible source positions corresponding to where a human speaker may be located and the weighting of that speaker location. The red dot refers to a noise source that can interfere with STI.

5.6.1 Speech Transmission Index

In particular, we are interested in maximizing the STI. It is an objective measure to predict speech intelligibility of a transmission channel that has been widely applied since 1970s (EN, 2011).

The basis of STI is the computation of the Modulation Transfer Function (MTF) (Houtgast and Steeneken, 1985). We use the simulated RIR to derive MTF of the transmission channel, using Equation 5.8:

$$m_k(f_m) = \frac{|\int_0^\infty r_k(t)^2 e^{-j2\pi f_m t} dt|}{\int_0^\infty r_k(t)^2 dt} \times (1 + 10^{-\mathrm{SNR}_k/10})^{-1}, \tag{5.8}$$

where $r_k(t)$ is our impulse response filtered to octave band k. Note that the noise in SNR$_k$ is a combination of physical noise, threshold and masking effects, which can be simulated with our propagation model. The result $m_k(f_m)$ is the modulation transfer ratio at f_m. For a full evaluation of the STI, we use 14 modulation frequencies (0.63Hz to 12.5Hz, 1/3 octave spaced) per band, yielding 98 samples of $m_k(f_m)$ in total. For robust implementation of STI from RIR, we refer to (Cabrera et al., 2014).

5.6.2 Our Optimization Algorithm

In a typical environment where speech recognition is used, having a high STI for a single source location is not sufficiently optimal for the entire environment. For example, in a household, the user of a speech recognition device could use the device from many different locations. As the user's location changes, so does the source position and the propagated sound. Therefore, we consider the set of all possible source locations S in our optimization formulation, based on user-defined constraints. This set is sampled discretely according to a uniform distribution yielding the source locations $s_1 \ldots s_n$ where n is the number of sampled locations. Given the optimization variable for the receiver location ℓ, we use the following objective function:

$$\arg\max_{\ell} \sum_{i=1}^{n} w_i \, \mathrm{STI}\left(r\left(s_i, \ell\right)\right), \tag{5.9}$$

where w_i is a weighting for the source location s_i defined by the user. Using this weight, certain regions of the environment can be prominent or dominant in our optimization algorithm. For example, if a speech recognition device is primarily for use in the living room, source locations in that room should be weighted higher. An overview of how the objective function is used to drive the overall optimization of the receiver location is shown in Figure 5.17.

In order to maximize Equation 5.9, we select from a set of discrete receiver locations. A discrete formulation allows the easy introduction of constraints. The primary constraint of the receiver location is an allowable set of surfaces the receiver can be placed on. A device would commonly be placed on a table or counter top, but the floor would not be a desirable location. Receiver locations are sampled from the areas allowed by the constraints. Figure 5.18 shows these constraints on a typical scene, such as an office workplace.

We use a simulated annealing (SA) approach for maximizing the STI. In typical SA approaches, an initial temperature T is chosen. Then, using a specified cooling schedule, the temperature is reduced on each iteration as the optimization variables are perturbed. The temperature is used to determine the probability of moving to a less-optimal state. Whenever a new state is chosen for the optimization variables, better states are allowed but worse states may still be permitted depending on the temperature. The probability for accepting a less optimal state is $e^{\frac{E'-E}{T}}$, where E' is the new energy given by the summation in Equation 5.9 and E is the energy of the previous iteration. Accepting less optimal states allows the simulated annealing algorithm to avoid local maxima.

In order to compute the objective function for a specific listener position, we compute the RIR and STI for every source. We can take advantage of the principle of acoustic reciprocity, evaluating the receiver position as a source, for greater efficiency.

STI	0–0.3	0.3–0.45	0.45–0.6	0.6–0.75	0.75–1
Quality	bad	poor	fair	good	excellent

Table 5.4: STI scale and quality (Steeneken and Houtgast, 2002).

Scene	Avg. STI Before	Avg. STI After	% Imprv.	Iterations	Time (s)
Office	0.5518	0.6757	22%	60	2357
Berlin	0.4377	0.5601	28%	76	2707
Suburban	0.3958	0.6571	71%	37	1686

Table 5.5: Our optimization process is able to improve the STI of scenes of differing complexity. We were able to improve the receiver STI in the Berlin scene from an intelligibility rating of “poor” to an intelligibility rating of “fair”. Our optimization process improved intelligibility of the suburban scene from “poor” to “good” (see Table 5.4).

5.7 Results of Speech Intelligibility Optimization

We tested our receiver placement optimization on three workplace and residential scenes. Our method is able to accurately compute the STI on complex indoor scenes. The optimization was computed on a desktop machine, using 8 threads for the hybrid sound propagation. Our benchmark scenes included Office, a multi-room workplace with conference rooms; Suburban, the ground floor of a multi-story house; and Berlin, a small apartment with two connected rooms.

We evaluate the performance of our algorithm on different benchmarks. On each scene we were able to obtain a result significantly greater than the just noticeable difference (JND) of STI, which is 0.03 (Bradley et al., 1999). A reference for the intelligibility of different STI values is included in Table 5.4. Our optimization algorithm is able to converge in a few iterations to the maximum STI. These results are summarized in Table 5.5.

Figure 5.19 shows the impact the choice of listener position has on the various speaking positions within the Suburban scene. Our optimization process takes into account multiple speaking locations throughout the environment rather than a single location.

A summary of the convergence of our algorithm is in Figure 5.20. The starting points for these experiments were selected randomly, and show how the simulated annealing process avoids local maxima over the course of optimization.

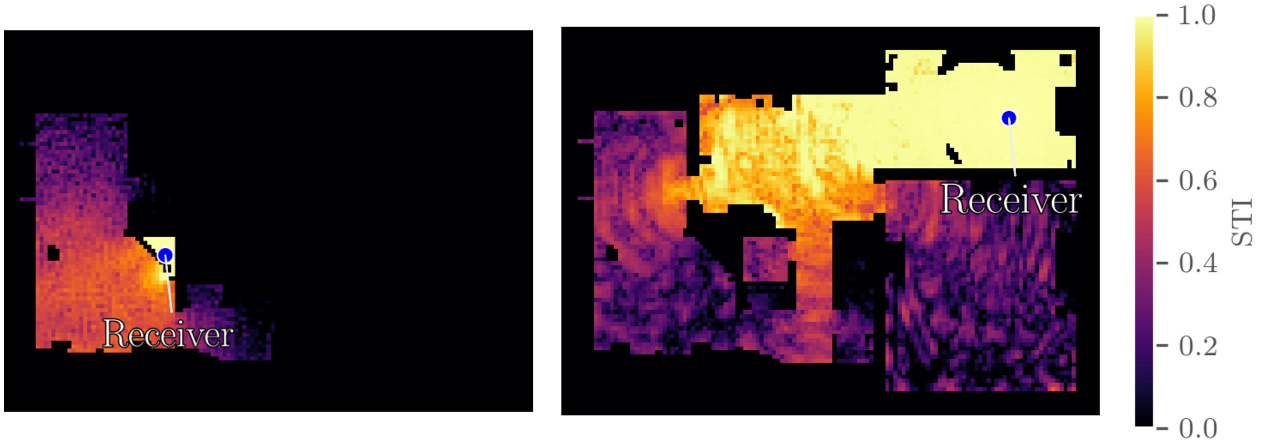


Figure 5.19: Side-by-side comparison of the minimal and maximal locations for the receiver according to our optimization process. The figure on the left shows a receiver placement next to the noise-emitting television, yielding a very poor STI rating. The figure on the right shows placement as a result of our optimization process. The receiver is placed away from the noise source, but also avoiding the reflective floor of the central kitchen area.

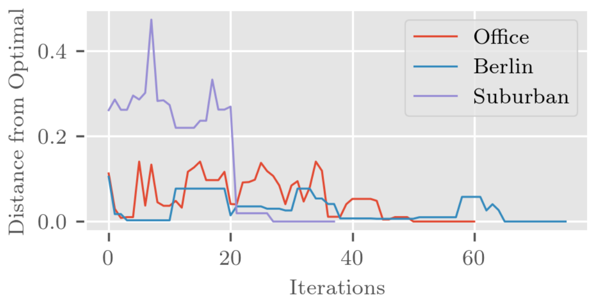


Figure 5.20: Convergence plot of our optimization process in different environments. These show the overall energy difference at each iteration in comparison to the final converged energy. The starting point was selected randomly.

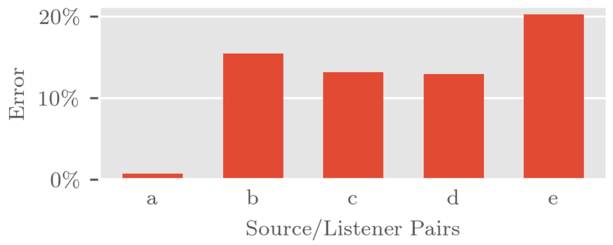


Figure 5.21: Comparison to using a Sabine decay approximation to generate impulse responses. Source/listener pairs *a* through *d* were selected from the Berlin scene. The error displayed is the relative error of a the STI from an RIR generated by using the Sabine reverb approximation compared to a full acoustic simulation. It is not sufficient to use Sabine even for simpler scenes such as the Berlin scene.

Comparison with Reverberation Models: Additionally, we compare our results to using an approximate reverberation model, such as Sabine's equation. We show significant errors using Sabine's equation, up to 20%, well outside the JND of STI. Figure 5.21 shows a comparison of several listener positions inside of a room in the Berlin scene.

5.8 Conclusion and Future Work

In this chapter, we described two efficient discrete optimization techniques using hybrid acoustic simulation that minimize acoustic noise at specific listener areas and maximize the STI of a sound receiver.

5.8.1 Noise Minimization

Our goal in hybrid noise minimization is to optimize the source locations to reduce noise. Our approach takes advantage of source clustering and impulse response caching in order to reduce the total search space of the algorithm in addition to reducing the iteration cost of the algorithm. We highlight the performance of our optimization technique on a variety of CAD models and we have evaluated on complex models of office and warehouses. Additionally, our approach is general for use by designers and engineers since it only requires a CAD mesh definition as input.

Our approach has some limitations. We assume that acoustic material characteristics of the environment are available and fixed. The accuracy of the propagation results is a function of these material characteristics. We also assume that the optimization function and the solver can compute the most optimal noise based on equation 5.6 or acoustic characteristics of the environment. Additionally, our wave-based technique assumes a heterogeneous environment and thus cannot propagate transmission effects. We would like to explore a hybrid scheme that can utilize these effects for noise propagation. In the future, we would like to extend to other types of noise models. We only utilized A-weighting curves. However other measures such as time-weighted average metrics are often used for safety and health purposes. Additionally, some metrics are based on noise rating curves; these test whether or not a frequency spectrum is fully above or below a specific noise curve rating. We would like to evaluate on other complex scenarios, including industrial models corresponding to the factories with heavy machinery. We would also like to explore alternatives to just source placement and also optimize the geometry or obstacle placement within a scene. Finally, we would like to extend to outdoor noise control applications, where it is important to know the occluder or wall locations (e.g. along a highway).

5.8.2 Speech Intelligibility Improvement

We present a novel optimization-based receiver placement algorithm to improve the intelligibility of spoken phrases. Using hybrid sound propagation, we are able to keep our performance costs low while maintaining accuracy. We have applied our approach to complex indoor scenes and obtained considerable improvement in STI. To the best of our knowledge, this is the first algorithm that performs accurate sound propagation optimization for this application.

In the future, we would like to couple our method with existing denoising and dereverberation algorithms. Since our optimization approach relocates the receiver, it works complementary to, rather than as a replacement to, these kinds of algorithms. Additionally, it is possible to use propagation information from our optimization process in order to improve denoising techniques.

CHAPTER 6: SUMMARY AND CONCLUSIONS

The primary purpose of this dissertation was to show how distributed parallel wave-based solvers and hybrid approaches can be used for solving computer-aided design problems in acoustics. Along with presenting a variety of techniques for solving sound propagation and optimization problems, we have shown how these algorithms can improve on the performance of existing approaches while maintaining accuracy, specifically at lower simulation frequencies. Furthermore, our methods have been tested on a wide variety of benchmark scenes, including complex indoor and outdoor environments. Our optimization problems show a dramatic improvement in the acoustics of the scenes, matching our target optimization values in the material optimization approaches and including a a 71% improvement in speech intelligibility and a 13 dB reduction in noise using our two hybrid optimization approaches.

6.1 Summary of Results

In this dissertation, we proposed a variety of algorithms for solving the problem of efficient and accurate acoustic simulation in sound propagation and optimization problems. We presented an approach for efficiently parallelizing the low-dispersion solver ARD and how it can be used for solving challenging acoustic problems at high frequencies. Additionally, we introduced two algorithms for automated acoustic material design and optimization using the ARD solver. Finally, we introduce the hybrid acoustic method that uses the ARD technique for lower frequencies while using a geometric approach for higher frequencies.

Using our distributed time-domain parallelization approach, called MPARD, we efficiently solve acoustic propagation problems efficiently at frequencies 10 kHz, far beyond the capabilities of most existing wave-based approaches that are limited to 1 kHz to 2 kHz. The MPARD method is able to effectively scale up to tens of thousands of cores. Taking advantage of the inherent efficiencies of the ARD method in our parallelization, we show practical high-frequency acoustics using wave-based methods.

Next, we show how efficient wave-based propagation can be used to solve 3D acoustic material design optimization problems. Given a CAD model of a scene, we are able to efficiently compute the optimal materials for specific desired acoustic characteristics such as strength, clarity, and reverberation time. Our two approaches include a continuous approach that uses analytic derivatives computed automatically in order to improve the convergence of the algorithm. Our derivative computation is analogous to the pressure field computation and therefore takes advantage of the same inherent efficiencies as ARD. Our second approach uses a discrete optimization formulation in order to apply constraints on the optimized materials. This approach selects the best acoustic material from a library in order to minimize the objective function. Early termination conditions for the optimization reduce the number of iterations required for convergence.

In order to overcome the low-frequency limitations of our optimization approaches, we introduce a hybrid optimization scheme that takes advantage of both the accuracy of wave-based simulations and the efficiency of geometric approaches. We propose two acoustic optimization schemes using this hybrid method: one for environmental noise reduction and the other for improving speech intelligibility. In our noise minimization scheme, we are able to take advantage of source clustering and impulse response caching in order to reduce the number of solves required for evaluating the objective function, thus further improving the efficiency of our algorithm. Using this technique, we are able to show a significant reduction in the noise field in sensitive areas. Our speech intelligibility improvement algorithm also takes advantage of the hybrid simulation technique, but using the acoustic reciprocity principle to simplify the problem of receiver placement. We similarly show a significant improvement of the STI on various scenes, including on some scenes from a rating of "poor" to "good".

6.2 Limitations

There are a few limitations to the methods to have been proposed in the dissertation. These typically deal with assumptions of the ARD solver that also motivate some future work on the solver.

The material model of ARD is simplistic, and only incorporates a non-physical absorption coefficient for all frequencies. Therefore, in our material optimization approaches, we use an empirical mapping to to real-world absorption values in sabins. Additionally, for multiple frequency bands, multiple simulations of the ARD solver are required, further increasing the computational cost of the method at higher frequencies. It is worthwhile future work to incorporate physically-based absorption and impedance properties in ARD materials.

A second limitation is the presence of numerical dispersion error and spurious reflections at the interface boundary between partitions in the ARD method. Overall, these errors are not significant (Raghuvanshi et al., 2009b), but in certain corner cases and partition configurations (particularly with small partitions), these can become problematic. A thorough error analysis of ARD is required to characterize the nature of these dispersion and reflection problems, but this is nontrivial to generalize to arbitrary scene configurations.

Additionally, ARD assumes a homogeneous media. For most purposes of room acoustics, this is a reasonable assumption to make. However, in some cases transmission effects may be important, such as in the domain of noise control and reduction. Furthermore, in the area of large-scale acoustics such as outdoor acoustics and aeronautical acoustics, heterogeneous changes in temperature and humidity may have a larger effect on the propagated sound. Similarly, the problem of underwater acoustic requires heterogeneous domains, so extension of the proposed methods to underwater acoustics would require the homogeneous assumption to be addressed.

Some further limitations of our methods include the time-domain cost of the ARD method. On particularly reverberant scenes, computing the full impulse response requires tens of thousands of time steps. Although this is not always necessary, metrics such as reverberation require a longer impulse response to accurately compute. Since ARD is parallelized on the spatial domain and not the time domain, efficient parallelization does not help as well for these problems. However, time-domain parallelization of an explicit time-stepping method like ARD is difficult.

Finally, a limiting factor of the accuracy of our simulations with respect to real-world environments is the lack of availability of accurate material absorption values and noise levels. Although we use well-known absorption values of various acoustic materials, in reality many of the materials in an environment may have differing properties. Furthermore, we found that particularly in our speech intelligibility improvement techniques, we may not model all of the factors in microphone reception, including ambient noise and microphone quality and directivity.

6.3 Future Work

In the future, we would like to address some of the aforementioned limitations of ARD and our methods. Some of the limitations, such as the assumption of a homogeneous domain and problems with numerical dispersion error could be ameliorated by using a non-uniform Fast-Fourier Transform for the ARD update step — a non-regular grid could be used to represent various speeds of sound and use a finer grid at partition boundaries while maintaining a course interior. This approach is not straightforward, however, as matching the grid resolution at the boundaries would be important and could limit some of the flexibility of the solver. Additionally, the ARD method would have to incorporate variable impedance characteristics.

Additionally, we are interested in the problem of large-domain or outdoor optimization. Combining the hybrid simulation approach with MPARD may yield a practical method of outdoor acoustic optimization, such as for the purposes of noise reduction on city streets or modeling sonic booms from aircraft flyovers. However, a limiting factor of this is still the number of time steps required for the ARD approach and potentially the heterogeneous nature of media at these scales.

We would also like to explore various other ways in which acoustic optimization techniques can be applied. Potential applications include recommendations for seating in concert halls or theaters, taking into account various acoustic, visual, and psychological metrics. Furthermore, we would like to explore other metrics in the area of noise control. Metrics such as the Time-Weighted Average (TWA) are used in many regulatory agencies to measure how much noise workers are allowed to be exposed to over longer periods of times. Other potential applications would be using information from acoustic simulation to drive filters or machine learning processes for automatic speech recognition algorithms.

Finally, we are interested in the direct application and validation of our models to real-world scenes. It would be interesting to see how accurately our models can predict acoustics in architectural structures. Additionally, we would like to measure how accurate recommendations from noise reduction are, and how they affect the environmental noise in an area. Additionally, we would like to test our speech intelligibility work with real speech recognition algorithms and judge how their effectiveness is improved.

BIBLIOGRAPHY

Alghamdi, A., Ahmadia, A., Ketcheson, D. I., Knepley, M. G., Mandli, K. T., and Dalcin, L. (2011). Petclaw: a scalable parallel nonlinear wave propagation solver for python. In *Proceedings of the 19th High Performance Computing Symposia*, HPC '11, pages 96–103, San Diego, CA, USA. Society for Computer Simulation International.

Allen, J. B. and Berkley, D. A. (1979). Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950.

Ballou, G. (2013). *Handbook for sound engineers*. Taylor & Francis.

Bao, H., Bielak, J., Ghattas, O., Kallivokas, L. F., O'Hallaron, D. R., Shewchuk, J. R., and Xu, J. (1998). Large-scale simulation of elastic wave propagation in heterogeneous media on parallel computers. *Computer methods in applied mechanics and engineering*, 152(1):85–102.

Bartholomew-Biggs, M., Brown, S., Christianson, B., and Dixon, L. (2000). Automatic differentiation of algorithms. *Journal of Computational and Applied Mathematics*, 124(1):171–190.

Basner, M., Babisch, W., Davis, A., Brink, M., Clark, C., Janssen, S., and Stansfeld, S. (2014). Auditory and non-auditory effects of noise on health. *The Lancet*, 383(9925):1325–1332.

Bassuet, A., Rife, D., and Dellatorre, L. (2014). Computational and optimization design in geometric acoustics. *Building Acoustics*, 21(1):75–86.

Beranek, L. (2011). The sound strength parameter g and its importance in evaluating and planning the acoustics of halls for musica). *The Journal of the Acoustical Society of America*, 129(5):3020–3026.

Beranek, L. L. (2008). Concert hall acoustics2008. *Journal of the Audio Engineering Society*, 56(7/8):532–544.

Berger, E. H. (2003). *The noise manual*. Aiha.

Bernacki, M., Fezoui, L., Lanteri, S., and Piperno, S. (2006). Parallel discontinuous galerkin unstructured mesh solvers for the calculation of three-dimensional wave propagation problems. *Applied mathematical modelling*, 30(8):744–763.

Bhandarkar, M. A. and Kalé, L. V. (2000). A parallel framework for explicit fem. In *High Performance ComputingHiPC 2000*, pages 385–394. Springer.

Bilbao, S. (2013). Modeling of complex geometries and boundary conditions in finite difference/finite volume time domain room acoustics simulation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(7):1524–1533.

Birgitta, B., Thomas, L., and Dietrich, H. (1999). Guidelines for community noise. *World Health Organization, Geneva*, page 21.

Blevins, M. G., Buck, A. T., Peng, Z., and Wang, L. M. (2013). Quantifying the just noticeable difference of reverberation time with band-limited noise centered around 1000 hz using a transformed up-down adaptive method. In *International Symposium on Room Acoustics*. Citeseer.

Bode, B., Butler, M., Dunning, T., Gropp, W., Hoe-fler, T., Hwu, W.-m., and Kramer, W. (2012). The blue waters super-system for super-science. In Jeffrey S . Vetter, C. and 2013, H., editors, *Contemporary HPC Architectures*. Sitka Publications.

Borish, J. (1984). Extension of the image model to arbitrary polyhedra. *The Journal of the Acoustical Society of America*, 75(6):1827–1836.

Bradley, J., Reich, R., and Norcross, S. (1999). A just noticeable difference in c 50 for speech. *Applied Acoustics*, 58(2):99–108.

Bronzaft, A. L. (2012). A quieter school: An enriched learning environment. *Quiet Classrooms. Retrieved on*, 15.

Brumm, H. (2004). The impact of environmental noise on song amplitude in a territorial bird. *Journal of Animal Ecology*, 73(3):434–440.

Bucur, V. (2007). *Urban forest acoustics*. Springer Science & Business Media.

Cabrera, D., Lee, D., Leembruggen, G., and Jimenez, D. (2014). Increasing robustness in the calculation of the speech transmission index from impulse responses. *Building Acoustics*, 21(3):181–198.

Çatalyürek, Ü. and Aykanat, C. (2011). Patoh (partitioning tool for hypergraphs). In *Encyclopedia of Parallel Computing*, pages 1479–1487. Springer.

Chan, T. F. and Widlund, O. B. (1994). Domain decomposition algorithms. In *Acta Numerica*, pages 61–143.

Chandak, A., Lauterbach, C., Taylor, M., Ren, Z., and Manocha, D. (2008). Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1707–1722.

Chen, J., Wang, Y., Yoho, S. E., Wang, D., and Healy, E. W. (2016). Large-scale training to increase speech intelligibility for hearing-impaired listeners in novel noises. *The Journal of the Acoustical Society of America*, 139(5):2604–2612.

Chen, J.-T., Lee, Y.-T., and Lin, Y.-J. (2010). Analysis of mutiple-shepers radiation and scattering problems by using a null-field integral equation approach. *Applied Acoustics*, 71(8):690–700.

Ciskowski, R. D. and Brebbia, C. A. (1991). *Boundary element methods in acoustics*. Springer.

Craggs, A. (1972). The use of simple three-dimensional acoustic finite elements for determining the natural modes and frequencies of complex shaped enclosures. *Journal of Sound and Vibration*, 23(3):331–339.

D'Antonio, P. and Cox, T. J. (1997). Room optimizer: A computer program to optimize the placement of listener, loudspeakers, acoustical surface treatment and room dimensions in critical listening rooms. In *Audio Engineering Society Convention 103*. Audio Engineering Society.

Davis, S. (1991). Low-dispersion finite difference methods for acoustic waves in a pipe. *The Journal of the Acoustical Society of America*, 90(5):2775–2781.

Deines, E. (2008). *Acoustic simulation and visualization algorithms*. PhD thesis, Techn. Univ., Fachbereich Informatik.

Dennis, Jr, J. E. and Moré, J. J. (1977). Quasi-newton methods, motivation and theory. *SIAM review*, 19(1):46–89.

Diekmann, R., Dralle, U., Neugebauer, F., and Römke, T. (1996). Padfem: a portable parallel fem-tool. In *High-Performance Computing and Networking*, pages 580–585. Springer.

Dielissen, V. J. and Kaldewaij, A. (1991). Rectangular partition is polynomial in two dimensions but np-complete in three. *Information Processing Letters*, 38(1):1 – 6.

Doicu, A. (2000). Acoustic and electromagnetic scattering analysis using discrete sources. *Acoustic and Electromagnetic Scattering Analysis Using Discrete Sources, ISBN: 978-0-12-219740-6, p. ix-xi.*, 1.

Du, J., Song, X., and Olhoff, N. (2011). Topological design of acoustic structure based on the bem-fem format and the mixed formulation. In *Proceeding of the 9th world congress on structural and multidisciplinary optimization, Shizuoka, Japan*.

Dühring, M. B., Jensen, J. S., and Sigmund, O. (2008). Acoustic design by topology optimization. *Journal of sound and vibration*, 317(3):557–575.

Egan, M. D. (1988). *Architectural acoustics*. McGraw-Hill Custom Publishing.

Eglese, R. (1990). Simulated annealing: a tool for operational research. *European journal of operational research*, 46(3):271–281.

Embrechts, J.-J., Archambeau, D., and Stan, G.-B. (2001). Determination of the scattering coefficient of random rough diffusing surfaces for room acoustics applications. *Acta Acustica united with Acustica*, 87(4):482–494.

EN, B. (2011). 60268-16: 2011.". *Sound system equipment–Part 16: Objective rating of speech intelligibility by speech transmission index*.

Engquist, B. and Runborg, O. (2003). Computational high frequency wave propagation. *Acta numerica*, 12:181–266.

Feng, X., Zhang, Y., and Glass, J. (2014). Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 1759–1763. IEEE.

Floodya, S. E. and Venegasb, R. B. (2007). Shape optimization in rectangular rooms for a correct modal distribution at low frequencies based on psychoacustical model. *Mecánica Computacional*, 26:81–95.

Fuchi, K. and Gea, H. C. (2013). Room acoustic optimization with variable thickness columns. *World Congress on Structural and Multidisciplinary Optimization*.

Funkhouser, T., Carlbom, I., Elko, G., Pingali, G., Sondhi, M., and West, J. (1998). A beam tracing approach to acoustic modeling for interactive virtual environments. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 21–32. ACM.

Funkhouser, T., Tsingos, N., and Jot, J.-M. (2003). Survey of methods for modeling sound propagation in interactive virtual environment systems. *Presence and Teleoperation*.

Galster, J. A. (2007). *The Effect of Room Volume on Speech Recognition in Enclosures with Similar Mean Reverberation Time*. PhD thesis.

Genuit, K. (2004). The sound quality of vehicle interior noise: a challenge for the nvh-engineers. *International Journal of Vehicle Noise and Vibration*, 1(1):158–168.

Genuit, K. and Bray, W. R. (2001). A virtual car: Prediction of sound and vibration in an interactive simulation environment. Technical report, SAE Technical Paper.

Gillespie, B. W. and Atlas, L. E. (2002). Acoustic diversity for improved speech recognition in reverberant environments. In *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, volume 1, pages I–557. IEEE.

Guddati, M. and Thirunavukkarasu, S. (2016). Reducing numerical dispersion and reflections in finite element and finite difference simulation of acoustic wave propagation and scattering. *The Journal of the Acoustical Society of America*, 139(4):2199–2199.

Guiffaut, C. and Mahdjoubi, K. (2001). A parallel fdtd algorithm using the mpi library. *Antennas and Propagation Magazine, IEEE*, 43(2):94–103.

Gumerov, N. A. and Duraiswami, R. (2009). A broadband fast multipole accelerated boundary element method for the three dimensional helmholtz equation. *The Journal of the Acoustical Society of America*, 125(1):191–205.

Hampel, S., Langer, S., and Cisilino, A. (2008). Coupling boundary elements to a raytracing procedure. *International journal for numerical methods in engineering*, 73(3):427–445.

Hirsch, H.-G. and Pearce, D. (2000). The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*.

Hornikx, M. and Forssén, J. (2007). The 2.5-dimensional equivalent sources method for directly exposed and shielded urban canyons. *The Journal of the Acoustical Society of America*, 122(5):2532–2541.

Hornikx, M., Krijnen, T., and van Harten, L. (2016). openpstd: The open source pseudospectral time-domain method for acoustic propagation. *Computer Physics Communications*, 203:298–308.

Hornikx, M., Waxler, R., and Forssén, J. (2010). The extended fourier pseudospectral time-domain method for atmospheric sound propagation. *The Journal of the Acoustical Society of America*, 128(4):1632–1646.

Houtgast, T. and Steeneken, H. J. (1985). A review of the mtf concept in room acoustics and its use for estimating speech intelligibility in auditoria. *The Journal of the Acoustical Society of America*, 77(3):1069–1077.

Hu, F., Hussaini, M., and Manthey, J. (1996). Low-dissipation and low-dispersion runge–kutta schemes for computational acoustics. *Journal of computational physics*, 124(1):177–191.

Huang, J., Yagel, R., Filippov, V., and Kurzion, Y. (1998). An accurate method for voxelizing polygon meshes. In *Volume Visualization, 1998. IEEE Symposium on*, pages 119–126. IEEE.

Hyde, J. R. and Möller, H. (2006). Sound strength in small halls. *Proceedings of the Institute of Acoustics*, 28(PT 2).

IEC (2013). Electroacoustics - sound level meters. IEC 61672-1, International Electrotechnical Commission, Geneva, Switzerland.

ISO (2009). Acoustics – measurement of room acoustic parameters. ISO 3882, International Organization for Standardization, Geneva, Switzerland.

Jamrah, A., Al-Omari, A., and Sharabi, R. (2006). Evaluation of traffic noise pollution in amman, jordan. *Environmental Monitoring and Assessment*, 120(1):499–525.

Johnescu, J. (2003). Modeling aircraft noise. *Occupational Health and Safety*, 72(7):106–108.

Kang, J. (2006). *Urban sound environment*. CRC Press.

Keane, A. and Nair, P. (2005). *Computational approaches for aerospace design: the pursuit of excellence*. John Wiley & Sons.

Keil, J. M. (2000). Polygon decomposition. *Handbook of Computational Geometry*, 2:491–518.

Keränen, J., Airo, E., Olkinuora, P., and Hongisto, V. (2003). Validity of ray-tracing method for the application of noise control in workplaces. *Acta Acustica united with Acustica*, 89(5):863–874.

Khalilian, H., Bajić, I. V., and Vaughan, R. G. (2015). Joint optimization of loudspeaker placement and radiation patterns for sound field reproduction. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 519–523. IEEE.

Kight, C. R. and Swaddle, J. P. (2011). How and why environmental noise impacts animals: an integrative, mechanistic review. *Ecology letters*, 14(10):1052–1061.

Ko, T., Peddinti, V., Povey, D., Seltzer, M. L., and Khudanpur, S. (2017). A study on data augmentation of reverberant speech for robust speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 5220–5224. IEEE.

Kowalczyk, K. and Van Walstijn, M. (2011). Room acoustics simulation using 3-d compact explicit fdtd schemes. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):34–46.

Kowalczyk, K. and van Walstijn, M. (2011). Room acoustics simulation using 3-d compact explicit fdtd schemes. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(1):34–46.

Krokstad, A., Strom, S., and Sørsdal, S. (1968). Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration*, 8(1):118–125.

Kuttruff, H. (2009). *Room acoustics*. CRC Press.

Kuttruff, H. (2016). *Room acoustics*. Crc Press.

Lewis, A. S. and Overton, M. L. (2009). Nonsmooth optimization via bfgs. *Submitted to SIAM J. Optimiz.*

Linkwitz, S. H. (1976). Active crossover networks for noncoincident drivers. *Journal of the Audio Engineering Society*, 24(1):2–8.

Lokki, T., Pätynen, J., Kuusinen, A., Vertanen, H., and Tervo, S. (2011a). Concert hall acoustics assessment with individually elicited attributes. *The Journal of the Acoustical Society of America*, 130(2):835–849.

Lokki, T., Southern, A., Siltanen, S., and Savioja, L. (2011b). Studies of epidaurus with a hybrid room acoustics modelling method. *Acoustics of Ancient Theaters Patras, Greece*.

Maressa, A., Pluymers, B., Donders, S., and Desmet, W. (2010). Nvh optimization methodologies based on bead modification analysis in vehicle body design. In *Proceedings of the international conference on noise and vibration engineering ISMA*, pages 4319–36. Citeseer.

Martins, J. R. and Lambe, A. B. (2013). Multidisciplinary design optimization: a survey of architectures. *AIAA journal*, 51(9):2049–2075.

Mazer, S. (2005). Curing the noise epidemic. *The Journal of the Acoustical Society of America*, 118(3):1956–1956.

Mehra, R., Raghuvanshi, N., Antani, L., Chandak, A., Curtis, S., and Manocha, D. (2013). Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Transactions on Graphics (TOG)*, 32(2):19.

Mehra, R., Raghuvanshi, N., Chandak, A., Albert, D. G., Wilson, D. K., and Manocha, D. (2014). Acoustic pulse propagation in an urban environment using a three-dimensional numerical simulation. *The Journal of the Acoustical Society of America*, 135(6):3231–3242.

Mehra, R., Raghuvanshi, N., Savioja, L., Lin, M. C., and Manocha, D. (2012). An efficient gpu-based time domain solver for the acoustic wave equation. *Applied Acoustics*, 73(2):83–94.

Mendez, T. (2014). *Computational Search in Architectural Design*. PhD thesis, Politecnico di Torino.

Miśkiewicz, A., Rogala, T., Rościszewska, T., Rudzki, T., and Fidecki, T. (2012). Concert hall sound clarity: A comparison of auditory judgments and objective measures. *Archives of Acoustics*, 37(1):41–46.

Monks, M., Mok Oh, B., and Dorsey, J. (1996). Acoustic simulatin and visualization using a new unified beam tracing and image source approach. In *Audio Engineering Society Convention 101*. Audio Engineering Society.

Monks, M., Oh, B. M., and Dorsey, J. (2000). Audioptimization: Goal-based acoustic design. *Computer Graphics and Applications, IEEE*, 20(3):76–90.

Montazeri, A., Poshtan, J., and Kahaei, M. (2003). Optimal placement of loudspeakers and microphones in an enclosure using genetic algorithm. In *Control Applications, 2003. CCA 2003. Proceedings of 2003 IEEE Conference on*, volume 1, pages 135–139. IEEE.

Morales, N., Chavda, V., Mehra, R., and Manocha, D. (2017). Mpard: A high-frequency wave-based acoustic solver for very large compute clusters. *Applied Acoustics*, 121:82–94.

Morales, N. and Manocha, D. (2016). Efficient wave-based acoustic material design optimization. *Computer-Aided Design*, 78:83–92.

Morales, N. and Manocha, D. (2017). Optimizing source placement for noise minimization using hybrid acoustic simulation. *Computer-Aided Design*.

Morales, N., Mehra, R., and Manocha, D. (2015a). A parallel time-domain wave simulator based on rectangular decomposition for distributed memory architectures. *Applied Acoustics*, 97:104–114.

Morales, N., Mehra, R., and Manocha, D. (2015b). A parallel time-domain wave simulator based on rectangular decomposition for distributed memory architectures. *Applied Acoustics*, 97:104–114.

Nam, D. and Park, C. H. (2000). Multiobjective simulated annealing: A comparative study to evolutionary algorithms. *International Journal of Fuzzy Systems*, 2(2):87–97.

Nandy, A. K. and Jog, C. (2012). Optimization of vibrating structures to reduce radiated noise. *Structural and Multidisciplinary Optimization*, 45(5):717–728.

Nefske, D., Wolf, J., and Howell, L. (1982). Structural-acoustic finite element analysis of the automobile passenger compartment: a review of current practice. *Journal of sound and vibration*, 80(2):247–266.

Ondet, A. and Barbry, J. (1989). Modeling of sound propagation in fitted workshops using ray tracing. *The Journal of the Acoustical Society of America*, 85(2):787–796.

Operto, S., Virieux, J., Amestoy, P., LExcellent, J.-Y., Giraud, L., and Ali, H. B. H. (2007). 3d finite-difference frequency-domain modeling of visco-acoustic wave propagation using a massively parallel direct solver: A feasibility study. *Geophysics*, 72(5):SM195–SM211.

OSHA (2008). Occupational noise exposure. OSHA 1910.95, Occupational Safety and Health Administration.

Palomäki, K. J., Brown, G. J., and Wang, D. (2004). A binaural processor for missing data speech recognition in the presence of noise and small-room reverberation. *Speech Communication*, 43(4):361–378.

Pearce, O., Gamblin, T., De Supinski, B. R., Schulz, M., and Amato, N. M. (2012). Quantifying the effectiveness of load balance algorithms. In *Proceedings of the 26th ACM international conference on Supercomputing*, pages 185–194. ACM.

Phipps, E. and Pawlowski, R. (2012). Efficient expression templates for operator overloading-based automatic differentiation. In *Recent Advances in Algorithmic Differentiation*, pages 309–319. Springer.

Piccolo, A., Plutino, D., and Cannistraro, G. (2005). Evaluation and analysis of the environmental noise of messina, italy. *Applied Acoustics*, 66(4):447–465.

Pluymers, B., Van Hal, B., Vandepitte, D., and Desmet, W. (2007). Trefftz-based methods for time-harmonic acoustics. *Archives of Computational Methods in Engineering*, 14(4):343–381.

Pollio, V. (1914). *Vitruvius: The ten books on architecture*. Harvard university press.

Rabiner, L. R. and Juang, B.-H. (1993). Fundamentals of speech recognition.

Raghuvanshi, N., Lloyd, B., Govindaraju, N. K., and Lin, M. C. (2009a). Efficient numerical acoustic simulation on graphics processors using adaptive rectangular decomposition. In *EAA Symposium on Auralization*, volume 33.

Raghuvanshi, N., Narain, R., and Lin, M. C. (2009b). Efficient and accurate sound propagation using adaptive rectangular decomposition. *Visualization and Computer Graphics, IEEE Transactions on*, 15(5):789–801.

Robinson, P. W., Siltanen, S., Lokki, T., and Savioja, L. (2014). Concert hall geometry optimization with parametric modeling tools and wave-based acoustic simulations. *Building Acoustics*, 21(1):55–64.

Rungta, A., Rust, S., Morales, N., Klatzky, R., Lin, M., and Manocha, D. (2016). Psychoacoustic characterization of propagation effects in virtual environments. *ACM Transactions on Applied Perception (TAP)*, 13(4):21.

Saarelma, J. and Savioja, L. (2014). An open source finite-difference time-domain solver for room acoustics using graphics processing units. *Acta Acustica united with Acustica*.

Saied, F. and Holst, M. J. (1991). Multigrid methods for computational acoustics on vector and parallel computers. *Urbana*, 51:61801.

Sakamoto, S., Ushiyama, A., and Nagatomo, H. (2006). Numerical analysis of sound propagation in rooms using the finite difference time domain method. *The Journal of the Acoustical Society of America*, 120(5):3008–3008.

Saksela, K., Botts, J., and Savioja, L. (2015). Optimization of absorption placement using geometrical acoustic models and least squares. *The Journal of the Acoustical Society of America*, 137(4):EL274–EL280.

Sakuma, T., Sakamoto, S., and Otsuru, T. (2014). *Computational simulation in architectural and environmental acoustics*. Springer.

Salomons, E. M., Lohman, W. J., and Zhou, H. (2016). Simulation of sound waves using the lattice boltzmann method for fluid flow: Benchmark cases for outdoor sound propagation. *PloS one*, 11(1):e0147206.

Savioja, L. (2010a). Real-time 3d finite-difference time-domain simulation of low-and mid-frequency room acoustics. In *13th Int. Conf on Digital Audio Effects*, volume 1, page 75.

Savioja, L. (2010b). Real-time 3d finite-difference time-domain simulation of low-and mid-frequency room acoustics. In *13th Int. Conf on Digital Audio Effects*, volume 1, page 75.

Savioja, L., Backman, J., Järvinen, A., and Takala, T. (1995). Waveguide mesh method for low-frequency simulation of room acoustics.

Savioja, L. and Svensson, U. P. (2015). Overview of geometrical room acoustic modeling techniques. *The Journal of the Acoustical Society of America*, 138(2):708–730.

Savioja, L. and Valimaki, V. (2000a). Reducing the dispersion error in the digital waveguide mesh using interpolation and frequency-warping techniques. *IEEE Transactions on Speech and Audio Processing*, 8(2):184–194.

Savioja, L. and Valimaki, V. (2000b). Reducing the dispersion error in the digital waveguide mesh using interpolation and frequency-warping techniques. *Speech and Audio Processing, IEEE Transactions on*, 8(2):184–194.

Savioja, L. and Valimaki, V. (2003). Interpolated rectangular 3-d digital waveguide mesh algorithms with frequency warping. *IEEE transactions on speech and audio processing*, 11(6):783–790.

Schissler, C., Loftin, C., and Manocha, D. (2017). Acoustic classification and optimization for multi-modal rendering of real-world scenes. *IEEE Transactions on Visualization and Computer Graphics*.

Schissler, C. and Manocha, D. (2011). Gsound: Interactive sound propagation for games. In *Audio Engineering Society Conference: 41st International Conference: Audio for Games*. Audio Engineering Society.

Schissler, C. and Manocha, D. (2016). Interactive sound propagation and rendering for large multi-source scenes. *ACM Transactions on Graphics (TOG)*, 36(1):2.

Schissler, C., Mehra, R., and Manocha, D. (2014a). High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Transactions on Graphics (TOG)*, 33(4):39.

Schissler, C., Mehra, R., and Manocha, D. (2014b). High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Transactions on Graphics (SIGGRAPH 2014)*, 33(4):39.

Schroeder, M. R. (1965). New method of measuring reverberation time. *The Journal of the Acoustical Society of America*, 37(3):409–412.

Schroeder, M. R. (1970). Digital simulation of sound transmission in reverberant spaces. *The Journal of the Acoustical Society of America*, 47(2A):424–431.

Schwan, P. (2003). Lustre: Building a file system for 1000-node clusters. In *Proceedings of the 2003 Linux Symposium*, volume 2003.

Sequeira, M. E. and Cortínez, V. H. (2016). Optimal acoustic design of multi-source industrial buildings by means of a simplified acoustic diffusion model. *Applied Acoustics*, 103:71–81.

Sheaffer, J. and Fazenda, B. (2014). Wavecloud: an open source room acoustics simulator using the finite difference time domain method. *Acta Acustica united with Acustica*.

Shu, L., Wang, M. Y., and Ma, Z. (2014). Level set based topology optimization of vibrating structures for coupled acoustic–structural dynamics. *Computers & Structures*, 132:34–42.

Sobieszczanski-Sobieski, J. (1990). Sensitivity of complex, internally coupled systems. *AIAA journal*, 28(1):153–160.

Southern, A., Siltanen, S., and Savioja, L. (2011). Spatial room impulse responses with a hybrid modeling method. In *Audio Engineering Society Convention 130*. Audio Engineering Society.

Steeneken, H. J. and Houtgast, T. (2002). Validation of the revised sti r method. *Speech Communication*, 38(3):413–425.

Sypek, P., Dziekonski, A., and Mrozowski, M. (2009). How to render fdtd computations more effective using a graphics accelerator. *Magnetics, IEEE Transactions on*, 45(3):1324–1327.

Taflove, A. and Hagness, S. C. (2005). *Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House Publishers, 3 edition.

Tashev, I. and Allred, D. (2005). Reverberation reduction for improved speech recognition. *Proc. Hands-Free Communication and Microphone Arrays*.

Taylor, M., Chandak, A., Mo, Q., Lauterbach, C., Schissler, C., and Manocha, D. (2012). Guided multiview ray tracing for fast auralization. *IEEE Transactions on Visualization and Computer Graphics*, 18(11):1797–1810.

Thompson, L. L. (2006). A review of finite-element methods for time-harmonic acoustics. *The Journal of the Acoustical Society of America*, 119(3):1315–1330.

Torres, R., Kleiner, M., Svensson, U., and Dalenbäck, B. (2001a). Edge diffraction and surface scattering in auralization. *Proceedings of SIGGRAPH Campfire*.

Torres, R. R., Svensson, U. P., and Kleiner, M. (2001b). Computation of edge diffraction for more accurate room acoustics auralization. *The Journal of the Acoustical Society of America*, 109(2):600–610.

Tsingos, N., Dachsbacher, C., Lefebvre, S., and Dellepiane, M. (2007). Instant sound scattering. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 111–120. Eurographics Association.

Tsingos, N., Funkhouser, T., Ngan, A., and Carlbom, I. (2001). Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 545–552. ACM.

Tsou, J.-Y., Chow, B., and Lam, S. (2003). Performance-based simulation for the planning and design of hyper-dense urban habitation. *Automation in construction*, 12(5):521–526.

Turnage, A. (2002). Reducing aircraft noise with computer graphics. *Computer Graphics and Applications, IEEE*, 22(3):16–21.

Vaccari, A., Cala'Lesina, A., Cristoforetti, L., and Pontalti, R. (2011). Parallel implementation of a 3d subgridding fdtd algorithm for large simulations. *Progress In Electromagnetics Research*, 120:263–292.

Van Duyne, S. and Smith, J. O. (1993). The 2-d digital waveguide mesh. In *Applications of Signal Processing to Audio and Acoustics, 1993. Final Program and Paper Summaries., 1993 IEEE Workshop on*, pages 177–180. IEEE.

Vorländer, M. (1989). Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm. *The Journal of the Acoustical Society of America*, 86(1):172–178.

Wang, Y., Safavi-Naeini, S., and Chaudhuri, S. K. (2000). A hybrid technique based on combining ray tracing and fdtd methods for site-specific modeling of indoor radio wave propagation. *IEEE Transactions on antennas and propagation*, 48(5):743–754.

Webb, C. J. and Bilbao, S. (2012). Binaural simulations using audio rate fdtd schemes and cuda. In *Proc. of the 15th Int. Conference on Digital Audio Effects (DAFx-12), York, United Kingdom*.

Xue, M., Yang, Y., Ruan, J., and Xu, Z. (2011). Assessment of noise and heavy metals (cr, cu, cd, pb) in the ambience of the production line for recycling waste printed circuit boards. *Environmental science & technology*, 46(1):494–499.

Yeh, H., Mehra, R., Ren, Z., Antani, L., Manocha, D., and Lin, M. (2013). Wave-ray coupling for interactive sound propagation in large complex scenes. *ACM Transactions on Graphics (TOG)*, 32(6):165.

Yoon, G. H., Jensen, J. S., and Sigmund, O. (2007). Topology optimization of acoustic–structure interaction problems using a mixed finite element formulation. *International journal for numerical methods in engineering*, 70(9):1049–1075.

Yu, W., Liu, Y., Su, T., Hunag, N.-T., and Mittra, R. (2005). A robust parallel conformal finite-difference time-domain processing package using the mpi library. *Antennas and Propagation Magazine, IEEE*, 47(3):39–59.

Yu, W., Yang, X., Liu, Y., Ma, L.-C., Sul, T., Huang, N.-T., Mittra, R., Maaskant, R., Lu, Y., Che, Q., et al. (2008). A new direction in computational electromagnetics: Solving large problems using the parallel fdtd on the bluegene/l supercomputer providing teraflop-level performance. *Antennas and Propagation Magazine, IEEE*, 50(2):26–44.

Zannin, P. H. T., Diniz, F. B., and Barbosa, W. A. (2002). Environmental noise pollution in the city of curitiba, brazil. *Applied Acoustics*, 63(4):351–358.

Zeng, Y. Q., Liu, Q. H., and Zhao, G. (2004). Multidomain pseudospectral time-domain (pstd) method for acoustic waves in lossy media. *Journal of Computational Acoustics*, 12(03):277–299.