

Rake, Peel, Sketch: The Signal Processing Pipeline Revisited

THÈSE N° 7651 (2017)

PRÉSENTÉE LE 5 MAI 2017

À LA FACULTÉ INFORMATIQUE ET COMMUNICATIONS
LABORATOIRE DE COMMUNICATIONS AUDIOVISUELLES
PROGRAMME DOCTORAL EN INFORMATIQUE ET COMMUNICATIONS

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Robin SCHEIBLER

acceptée sur proposition du jury:

Dr O. Lévêque, président du jury
Prof. M. Vetterli, directeur de thèse
Prof. L. Daudet, rapporteur
Prof. N. Ono, rapporteur
Prof. N. Vishnoi, rapporteur



ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2017

To Risa, Rémi, and Louis

To air, without whose vibrations much of this thesis would not be

Acknowledgments

There was a table set out under a tree in front of the house, and the March Hare and the Hatter were having tea at it: a Dormouse was sitting between them, fast asleep, and the other two were using it as a cushion, resting their elbows on it, and talking over its head.

Alice's Adventures in Wonderland
LEWIS CARROLL

Martin, thank you for the amazing journey. I came for the freedom to conduct my own research and was not disappointed! I am grateful for your supervision and the trust you gave me. Even if at times I felt like I was fumbling in the dark, you had new insights and long term visions to put me back on track whenever we met. Thank you for tolerating my short attention span and my multiple side projects. And, finally, thanks for bringing together such great people in the lab. It was an honor and a pleasure to be part of it!

Next I would like to thank my thesis committee, Laurent Daudet, Nobutaka Ono, Nisheeth Vishnoi, and its president Olivier L  v  que, for taking the time to thoroughly read my thesis and provide critical feedback, as well as new ideas.

Serendipity, and sometimes a gentle push from Martin, allowed me to collaborate with a number of brilliant researchers without whom this thesis would be much blander. Thank you Saeid, it was a chance to meet you early on. The quiet strength with which you conduct research is an example for me. Then, I have to thank Ivan for counterbalancing this quietness by bringing his wild and contagious enthusiasm, in research, brewing, and everything else, to our common endeavors. Ren  , thank you for partnering on the crazy microphone array project. There is a lot for me to learn from your meticulous approach to embedded systems and your experience supervising student projects. Thanks Hanjie for an infinite rate of FRI goodness, great breakfasts in Brisbane, and to occasionally indulge my love of hamburgers. Thank you Eric for believing in the demo and bringing so much energy and passion to it.

Next, there are a few mentors I need to thank for their guidance. Amina for introducing me to the LCAV way of doing research, teaching me how to write and make presentations, and generally what it means to care for one's student. Dirk Schr  der for his help and deep wisdom in acoustics and experiments, his strong and honest opinions, and the ski and hiking trips.

One characteristic of life at LCAV is the constant stimulation provided by its collection of singular but very likable characters. Thank you Juri, after a great start in Zürich, it was great fun to meet again and spend some time in the office next door. Thanks Marta for always bringing a fresh perspective and teaching me that Spanish *chiringuitos* are better than Swiss ones (obvious in hindsight)! Mihailo for the late night drinks and dinners (not before 10pm) and all the stories. Paolo for sharing advice on latex and spare bicycle parts. Thanks Frederike for joining us and the robot, and then deciding to stick around. Hesam, my longest running office mate, for improving our shared space with beautiful plants and interesting discussions. And thank you to everyone else who made this journey so enjoyable: Mitra, Niranjan, Xue, Runwei, Dalia, and Miranda, Gilles, Adam, Benjamin, Zichong, Andreas, Golnoosh, and Matthieu, Lionel, Jay, Loïc, Reza, and Julien. Thanks for all the good times, the ICASSP dinners, the parties, and the coffee breaks. While it is sad to part ways, I also know that wherever I will go, there will always be a friend close by!

One of the highlights of my PhD time was to be able to work with many great students on a variety of fun projects. A big thank you to Sidney, Ivan, Thomas, Juan, Basile, and Corentin. Your hard work and dedication made it possible to create fantastic microphone arrays and a great demo! During the realization of the microphone arrays and the demonstration hardware, I was very lucky to benefit from the invaluable help and experience of André Guignard, André Badertscher, and Peter Brühlmeier

Thank you Sachiko for inviting me to collaborate on the Biodesign for the Real World project. It was great fun to step out of my comfort zone to work with these pesky biological organisms. Thanks also to all the students who trusted us enough to spend a semester on this strange looking project. Thanks to the Hackuarium community for welcoming Biodesign and being such a great place to hang out and dream new projects.

At home, I cannot thank Risa enough. For following me here and bearing life without *karaoke* and *izakaya* for four years. For being a hundred percent supportive of anything I do, and always understanding of my antics. Thanks for being here with me. For the last two years, we had the addition of the fantastic Rémi. Thank you for brightening our life and making sure there is not a single boring day. Special thanks to my mother-in-law Tokuko who kindly hosted me in snowy Tohoku while I was writing this thesis.

Last but not least, I would like to thank my family. My fearless older sister Sophie for showing the way one conquers the world. My parents Josiane and André for their unconditional love and support, for nurturing my curiosity and sense of wonder, and teaching me the unknown is not to be feared. Thank you for everything.

Abstract

The prototypical signal processing pipeline can be divided into four blocks. Representation of the signal in a basis suitable for processing. Enhancement of the meaningful part of the signal and noise reduction. Estimation of important statistical properties of the signal. Adaptive processing to track and adapt to changes in the signal statistics. This thesis revisits each of these blocks and proposes new algorithms, borrowing ideas from information theory, theoretical computer science, or communications.

First, we revisit the Walsh-Hadamard transform (WHT) for the case of a signal sparse in the transformed domain, namely that has only $K \leq N$ non-zero coefficients. We show that an efficient algorithm exists that can compute these coefficients in $O(K \log_2(K) \log_2(N/K))$ and using only $O(K \log_2(N/K))$ samples. This algorithm relies on a fast hashing procedure that computes small linear combinations of transformed domain coefficients. A bipartite graph is formed with linear combinations on one side, and non-zero coefficients on the other. A peeling decoder is then used to recover the non-zero coefficients one by one. A detailed analysis of the algorithm based on error correcting codes over the binary erasure channel is given.

The second chapter is about beamforming. Inspired by the rake receiver from wireless communications, we recognize that echoes in a room are an important source of extra signal diversity. We extend several classic beamforming algorithms to take advantage of echoes and also propose new optimal formulations. We explore formulations both in time and frequency domains. We show theoretically and in numerical simulations that the signal-to-interference-and-noise ratio increases proportionally to the number of echoes used. Finally, beyond objective measures, we show that echoes also directly improve speech intelligibility as measured by the perceptual evaluation of speech quality (PESQ) metric.

Next, we attack the problem of direction of arrival of acoustic sources, to which we apply a robust finite rate of innovation reconstruction framework. FRIDA — the resulting algorithm — exploits wideband information coherently, works at very low signal-to-noise ratio, and can resolve very close sources. The algorithm can use either raw microphone signals or their cross-correlations. While the former lets us work with correlated sources, the latter creates a quadratic number of measurements that allows to locate many sources with few microphones. Thorough experiments on simulated and recorded data shows that FRIDA compares favorably with the state-of-the-art.

We continue by revisiting the classic recursive least squares (RLS) adaptive filter with ideas borrowed from recent results on sketching least squares problems. The exact update of RLS is replaced by a few steps of conjugate gradient descent. We propose then two different preconditioners, obtained by sketching the data, to accelerate the convergence of the gradient descent. Experiments on artificial as well as natural signals show that the proposed algorithm has a performance very close to that of RLS at a lower computational burden.

The fifth and final chapter is dedicated to the software and hardware tools developed for this thesis. We describe the *pyroomacoustics* Python package that contains routines for the evaluation

of audio processing algorithms and reference implementations of popular algorithms. We then give an overview of the microphone arrays developed and used for the experimental validation of FRIDA. We use this as an opportunity to start a discussion on the challenges of reproducible research at a global level. We conclude with a modest proposal.

Keywords: Walsh-Hadamard transform, sparsity, sublinear algorithm, peeling decoder, beamforming, rake receiver, echoes, perceptual evaluation of speech quality (PESQ), direction of arrival, finite rate of innovation, wideband, high resolution, adaptive filter, recursive least squares, least squares sketching, conjugate gradient, room impulse response generation, Python, hardware, microphone arrays, reproducible research

Résumé

Un algorithme de traitement des signaux peut typiquement être divisé en quatre blocs. La représentation du signal dans une base qui convient au traitement. La réduction du bruit. L'estimation des propriétés statistiques importantes du signal. Le traitement adaptatif qui traque et s'adapte au changement dans les statistiques du signal. Dans cette thèse nous revisitons chacun de ces blocs et y proposons de nouveaux algorithmes à la lueur d'idées provenant de la théorie de l'information, l'informatique théorique, ou les communications numériques.

En premier, nous revisitons la transformée de Walsh-Hadamard (WHT) dans le cas où le signal est parcimonieux dans le domaine transformé, c'est-à-dire, il ne comporte que peu de coefficients non-zéros. Nous montrons qu'un algorithme efficace existe pour calculer ces coefficients en temps $O(K \log_2 K \log_2(N/K))$ et en n'utilisant que $O(K \log_2(N/K))$ échantillons. Cet algorithme utilise une procédure de hachage rapide qui calcule de petites combinaisons linéaires de coefficients du domaine transformé. Un graphe bipartite est formé avec ces combinaisons linéaires d'un côté, et les coefficients non-zéros de l'autre. Un décodeur-éplucheur est ensuite utilisé pour retrouver les coefficients non-zéros un par un. Une analyse détaillée de l'algorithme est faite, basée sur les codes correcteurs d'erreur sur le canal binaire à effacement.

Le second chapitre porte sur le beamforming. Inspiré par le récepteur-râteau des communications numériques, nous reconnaissons l'importance des échos dans une pièce comme source de diversité additionnelle. Nous étendons plusieurs algorithmes de beamforming classiques pour exploiter de façon constructive les échos, et nous proposons également de nouvelles formulations optimales. Nous explorons aussi bien les formulations en domaine fréquentiel que temporel. Nous montrons théoriquement et numériquement que le rapport signal-sur-interférences-et-bruit (SINR) augmente proportionnellement au nombre d'échos utilisé par le beamformer. Finalement, au-delà des mesures objectives, nous montrons que les échos améliorent directement l'intelligibilité de la voix telle que mesurée par la métrique d'évaluation perceptuelle de la qualité de la voix (PESQ).

Nous attaquons ensuite le problème de la reconstruction de la direction de sources sonores, auquel nous appliquons un algorithme de reconstruction robuste basé sur l'échantillonnage au taux d'innovation fini. FRIDA, l'algorithme résultant, exploite la large bande du signal de façon cohérente, fonctionne à un niveau de rapport signal-sur-bruit très bas, et peut distinguer des sources très rapprochées. L'algorithme peut utiliser soit les mesures des microphones directement, ou alors leurs corrélations croisées. Avec les premières, il est possible de retrouver des sources corrélées, alors que les secondes sont en nombre quadratique et permettent ainsi de localiser beaucoup de sources avec peu de microphones. Des expériences approfondies aussi bien sur données synthétiques qu'enregistrées montrent que l'algorithme se compare favorablement à l'état-de-l'art.

Nous continuons par le traitement adaptatif et revisitons l'algorithme classique des moindres carrés récursif (RLS) avec des idées empruntées aux résultats récents sur les esquisses de moindres carrés. La mise-à-jour exacte de RLS est remplacée par une descente de gradient conjugué.

Nous proposons ensuite deux préconditionneurs formés à partir d'une esquisse des données afin d'accélérer la convergence de la descente de gradient. Des expériences avec des signaux synthétiques autant que naturels montrent que la performance de cet algorithme reste proche de celle de RLS à un coût en calcul moindre.

Le cinquième et dernier chapitre de cette thèse est dédié au logiciel et matériel développé durant celle-ci. Nous y décrivons *pyroomacoustics*, un logiciel Python pour l'évaluation des algorithmes de traitement audio, et qui contient également un bon nombre d'implémentations de références pour des algorithmes populaires de beamforming ou de localisation. Nous donnons ensuite un aperçu des systèmes multi-microphones conçus et utilisés pour la validation expérimentale de l'algorithme FRIDA. Nous utilisons cette opportunité pour commencer une discussion autour des défis de la recherche reproductible. Nous concluons par une proposition modeste à ce sujet.

Mots-Clés : Transformée de Walsh-Hadamard, parcimonie, algorithmes sous-linéaires, décodeur-épelucheur, beamforming, récepteur-râteau, échos, évaluation perceptuelle de la qualité de la voix (PESQ), direction d'arrivée, taux d'innovation fini, large bande, haute résolution, filtre adaptatif, moindres carrés récursif, esquisse des moindres carrés, gradient conjugué, génération de réponse impulsionnelle de pièce, Python, matériel, systèmes multi-microphones, recherche reproductible

Contents

Acknowledgments	iii
Abstract	vii
Résumé	ix
1 Introduction	1
1.1 The Signal Processing Pipeline	2
1.2 Thesis Outline and Main Contributions	4
2 Representation: Sparse Fast Hadamard Transform	9
2.1 Introduction	9
2.1.1 Related Work	10
2.1.2 Main Contribution	11
2.1.3 Notations and Preliminaries	11
2.2 Main Result	12
2.3 The Walsh-Hadamard Transform and its Properties	13
2.3.1 Basic Properties	13
2.4 Hadamard Hashing Algorithm	16
2.4.1 Properties of Hadamard Hashing	17
2.5 Sparse Fast Hadamard Transform	18
2.5.1 Explanation of the Algorithm	19
2.5.2 Complexity Analysis	20
2.6 Performance Analysis of the very Sparse Regime	22
2.6.1 Hash Construction	23
2.6.2 Random Bipartite Graph Construction	23
2.6.3 Performance Analysis of the Peeling Decoder	27
2.7 Performance Analysis of the Less Sparse Regime	31
2.7.1 Hash Construction	31
2.7.2 Bipartite Graph Representation	32
2.7.3 Performance Analysis of the Peeling Decoder	32
2.7.4 Generalized Hash Construction	35
2.8 Experimental Results	36
2.9 Conclusion	37
2.A Proof of the Properties of the WHT	38
2.A.1 Proof of Property 2.1	38

2.A.2	Proof of Property 2.2	38
2.A.3	Proof of Property 2.3	39
2.A.4	Proof of Property 2.4	39
2.A.5	Proof of Property 2.5	40
2.B	Proof of Proposition 2.2	40
2.C	Proof of Proposition 2.3	41
2.D	Proof of Proposition 2.9	42
3	Enhancement: Acoustic Rake Receivers	45
3.1	Introduction	45
3.1.1	Related Work	48
3.1.2	Main Contributions and Limitations	48
3.1.3	Chapter Outline	50
3.2	Signal Model	50
3.3	Simulation of Beamforming Algorithms	52
3.4	Frequency Domain Formulations	54
3.4.1	Classic Beamformers	54
3.4.2	Acoustic Rake Receiver	56
3.4.3	Expected SINR Gain from Raking	59
3.4.4	Numerical Experiments	60
3.5	Time Domain Formulations	64
3.5.1	Notation	65
3.5.2	Time Domain Acoustic Rake Receivers	66
3.5.3	Numerical Experiments	68
3.6	Finding and Tracking the Echoes	69
3.6.1	Known Room Geometry	70
3.6.2	Acoustic Geometry Estimation	70
3.6.3	Without Estimating the Room Geometry	71
3.7	Conclusion	71
3.A	Theorem 3.1	72
4	Estimation: FRI-based Direction of Arrival Finding	75
4.1	Introduction	75
4.1.1	Related Work	76
4.1.2	Main Contribution	77
4.1.3	Chapter Organization	78
4.2	Source Signal and Measurements	78
4.2.1	Sources with Arbitrary Spatial Support	78
4.2.2	Point Sources	79
4.2.3	Uncorrelated Point Sources	80
4.3	Point Source Reconstruction	81
4.3.1	Relation between Measurements and Uniform Samples of Sinusoids	82
4.3.2	Annihilation on the Circle	85
4.3.3	Reconstruction Algorithm	86
4.4	Experiments	88
4.4.1	Influence of Noise	89
4.4.2	Resolving Close Sources	90

4.4.3	Experiments on Recorded Signals	90
4.5	From Direction of Arrival to Blind Sparse Channel Identification	91
4.6	Conclusion	92
4.A	Proof of Lemma 4.1	92
5	Adaptive Processing: The Recursive Hessian Sketch	93
5.1	Introduction	93
5.2	Background	95
5.2.1	Adaptive Filters	96
5.2.2	Least-Squares Sketching	99
5.3	The Recursive Hessian Sketch Algorithm	101
5.3.1	Row Sampling Preconditioner	101
5.3.2	Circulant Preconditioner	103
5.4	Complexity Analysis	105
5.5	Convergence Analysis of Accelerated RHS	106
5.6	Numerical Experiments	108
5.7	Conclusion	111
5.A	Proof of Theorem Theorem 5.2	112
5.B	Fast Matrix-Vector Products for Structured Matrices	114
6	Tools and Methods for Reproducible Research in Computational Acoustics	117
6.1	Introduction	117
6.2	Software: The Pyroomacoustics Package	118
6.2.1	Structure	120
6.2.2	Room Impulse Response Generator	120
6.2.3	STFT Engine	122
6.2.4	Reference Implementations	122
6.2.5	Future Work	124
6.3	Hardware: Flexible Microphone Array Architectures	124
6.3.1	CompactSix Array	126
6.3.2	Pyramic Array	127
6.3.3	Easy-DSP: Browser Based Interface for Embedded Arrays	128
6.4	Thoughts on Open Science and Reproducible Research	129
	Conclusion	132
	Bibliography	135
	Curriculum Vitæ	145

Chapter 1

Introduction

Alice opened the door and found that it led into a small passage, not much larger than a rat-hole: she knelt down and looked along the passage into the loveliest garden you ever saw.

Alice's Adventures in Wonderland

LEWIS CARROLL

The history of signal processing runs parallels to that of computing and algorithmics. In the 17th century, driven by a thirst to understand the natural world, scientists started to collect data in a rigorous way. The experimental data collected was discrete in nature and from the very beginning required algorithms to extract information and predictions from them, for example compute the orbits of the moon and planets. In fact, an early version of the fast Fourier transform (FFT) algorithm was developed just for that problem by Gauss [63]. While this early processing required painstaking manual computations to be carried out, the invention of semiconductor technology and the rediscovery of the FFT by Cooley and Tukey [31] triggered the blossom of digital signal processing (DSP) in the 1970s.

Even while computers widened considerably the possible applications of DSP, the need to crunch ever more numbers kept increasing, pushing computer scientists and engineers alike to devise sophisticated methods to reduce the growing computational burden. These advances often happened independently in separate fields of investigation and their applicability to other disciplines was not immediately noticed. As an example methods developed in the statistical physics community turned out to be applicable to information processing and coding and this connection spurred a very active research area in information theory. Another recent example is that of theoretical computer scientist revisiting the classic FFT algorithm in the context of sparse data, making the first algorithmic progress on the problem in years. Similarly, advances in convex optimization have enabled the practical application of compressed sensing techniques.

In the best engineering tradition, we take a look at the prototypical signal processing pipeline

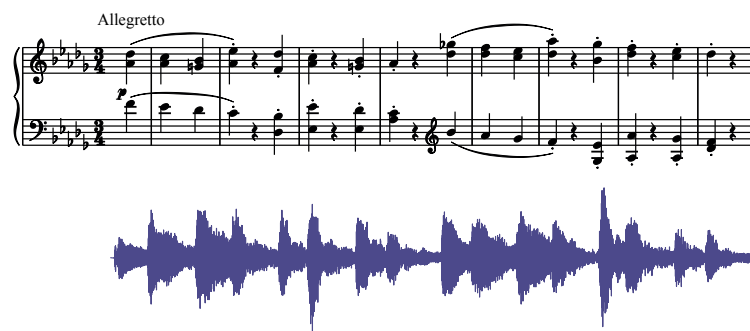


Figure 1.1: The score and the recorded waveform of the opening to the second movement of the *Piano Sonata No. 14*, the Moonlight Sonata, by Beethoven. We observe that while the timing of the notes is still apparent from the waveform, the frequency content cannot be guessed.

as a system composed of several blocks. We proceed through the chapters to revisit these blocks, applied to specific problems, using modern algorithmic tools. We borrow ideas and techniques from neighboring fields, namely theoretical computer science, wireless communications, or information theory, and arrive at a number of new algorithms for old problems.

1.1 The Signal Processing Pipeline

It is traditional in engineering to represent complex systems as a collection of simpler sub-systems, with well-defined tasks, interacting with each other. In signal processing, these sub-systems roughly fall into four categories: representation, enhancement, estimation, and adaptive processing. Many problems can be decomposed into blocks that belong to one of these categories. We begin by describing each of these categories, before giving a couple of examples.

Representation There are always several possible ways of representing (or describing) the same fundamental object. Not all such descriptions turn out to be essential and some will be more suited to some tasks than others. Take the example of a musician who wants to play a piece of Beethoven. He will find it easier to read it from a score than by looking at pictures of vibrating strings, as illustrated in Figure 1.1. The reason is that the score efficiently encodes the important information, the frequency and timing of notes, while the more accurate drawing hides the crucial information, frequency in this example, within irrelevant details.

Representations are often implemented as linear mappings, that is change of basis, linear algebraically speaking. The most famous representation is the Fourier basis — the mathematical equivalent of the musical score of our previous example — it can transform a sound signal captured by a microphone into its frequency content. The constant need for applying such transformation in signal processing has led to the necessity of very fast algorithms like the FFT.

Enhancement A key parameter for signal processing is the signal-to-noise ratio (SNR), defined as the ratio of the expected power of the signal of interest, $x(t)$, to that of the irrelevant

information, or noise, $n(t)$, contained in the measurement $y(t) = x(t) + n(t)$,

$$\text{SNR} = \frac{\mathbb{E} [|x(t)|^2]}{\mathbb{E} [|n(t)|^2]},$$

which is usually expressed in decibels. The purpose of signal enhancement is to output a new signal with a larger SNR than the input. Without being exhaustive, a few enhancement techniques have proved to be popular over the years, let us mention denoising by averaging and by thresholding.

Denoising by averaging relies on having access to multiple copies of the signal of interest corrupted by statistically independent realizations of the noise. If K copies are available, i.e. $y_k(t) = x(t) + n_k(t)$, $k = 1, \dots, K$, then the variance of the corrupting noise in their average $\bar{y}(t) = \sum_k y_k(t) = x(t) + \bar{n}(t)$ is reduced by a factor K . This observation forms the basis for more sophisticated techniques such as beamforming.

Denoising by thresholding assumes that, on the one hand, there exists a sparse representation of the signal of interest, i.e. most of the signal can be represented by a few coefficients. The noise, on the other hand, is assumed not to be sparse in this representation. In particular, we assume that we can find a threshold above which we mostly have signal components, while the noise stays below. Denoising is then achieved simply by zeroing all components smaller than the threshold. Examples of this technique are wavelet thresholding and spectral subtraction in image and speech processing, respectively.

Estimation Often we wish to estimate some key properties of the target signal. For example, it might be of interest to estimate the spectral content of an acoustic signal. Often the estimated quantities are used as inputs to a different algorithm. For example, the value of the threshold in denoising by thresholding can be chosen according to the variance of the noise, which must in turn be estimated from the available signal. Sometimes the estimated quantities are themselves of interest. There are two main families of estimation methods: parametric and non-parametric. Parametric methods assume the target signal follows an underlying model ruled by a few parameters. Non-parametric methods can be applied to any signal.

Adaptive Processing Most conventional statistical signal processing algorithms rely on results in expectation and assume fixed signal statistics. This is in stark contrast with natural signals which are in general strongly non-stationary with changing statistical properties. Adaptive processing is the branch of signal processing that deals specifically with non-stationary signals. Adaptive algorithms learn the statistical properties of the signal on the fly and adapt their processing accordingly. They often implement a kind of online optimization where an objective function is being minimized. When new data is observed, its discrepancy with the current estimate is used to produce a new estimate in a way that reduces the objective. Adaptive algorithms fulfill critical roles in many modern systems such as channel equalization for digital communication systems, echo cancellation in teleconferencing devices, tracking of dynamic sound sources, trajectory smoothing, and data fusion.

A Few Examples Let us now give a few examples of practical systems. First, consider the generic denoising pipeline for a mobile handset pictured in Figure 1.2a. Many handsets are currently equipped with two microphones. One is close to the speaker's mouth, while the other, at the back of the device, captures a reference of the ambient noise. After analog to digital

conversion of the microphones inputs, a short time Fourier transform (STFT) is applied. For acoustic signals, the time-frequency representation of the STFT is not only intuitive, matching the human intuition of acoustic signals, but also provides efficient filtering. Then, an adaptive noise canceller compensates for the difference in propagation between the two microphones before subtracting the noise from the speaker's signal. Because a little bit of residual noise is usually present at the output of the noise canceller, further enhancement is needed. The profile of the residual noise is estimated and subtracted from the signal in the frequency magnitude domain. The inverse STFT is finally applied to synthesize the signal sent to the remote talker.

Our second example is a multi-microphone teleconferencing system illustrated in Figure 1.2b. This is a device sitting at the center of the table of a meeting room that allows conversation with multiple local and remote participants. After the usual STFT step, a sound source localization block tracks the locations of the sound sources. The beamforming filters are then adjusted accordingly to amplify or reduce the different sources as deemed appropriate. Now, because the loudspeaker is in general closer to the microphones than the speakers in the room, it is necessary to filter out the remote speaker from the input to avoid a feedback effect. This is done by an echo cancellation block implemented by an adaptive filter.

1.2 Thesis Outline and Main Contributions

In this thesis we make contributions to all four blocks of the prototypical signal processing pipeline as discussed above: representation, enhancement, estimation, and adaptive processing. They are covered in that order over the next four chapters. The sixth and final chapter describes the tools, software and hardware, that were developed as a by-product of this thesis, and how to share them in a reproducible way.

Chapter 2 — Representation: Sparse Fast Hadamard Transform The first problem we attack is that of computing the Hadamard transform. This is motivated by recent progress made for the computation of the FFT under the assumption of sparsity in the frequency domain. In this case it was shown that algorithms with complexity sublinear in the transform length exist [50]. We design a new iterative low-complexity algorithm for computing the Walsh-Hadamard transform (WHT) of an N dimensional signal with a K -sparse WHT. We suppose that N is a power of two and $K = O(N^\alpha)$, scales sub-linearly in N for some $\alpha \in (0, 1)$. Assuming a random support model for the nonzero transform-domain components, our algorithm reconstructs the WHT of the signal with a sample complexity $O(K \log_2(\frac{N}{K}))$ and a computational complexity $O(K \log_2(K) \log_2(\frac{N}{K}))$. Moreover, the algorithm succeeds with a high probability approaching 1 for large dimension N .

Our approach is mainly based on the subsampling (aliasing) property of the WHT, where, by a carefully designed subsampling of the time-domain signal, a suitable aliasing pattern is induced in the transform-domain. We treat the resulting aliasing patterns as parity-check constraints and represent them by a bipartite graph. We analyze the properties of the resulting bipartite graphs and borrow ideas from codes defined over sparse bipartite graphs to formulate the recovery of the nonzero spectral values as a peeling decoding algorithm for a specific sparse-graph code transmitted over a binary erasure channel (BEC). This enables us to use tools from coding theory (belief-propagation analysis) to characterize the asymptotic performance of our algorithm in the *very sparse* ($\alpha \in (0, \frac{1}{3}]$) and the *less sparse* ($\alpha \in (\frac{1}{3}, 1)$) regime. Comprehensive simulation results are provided to assess the empirical performance of the proposed algorithm.

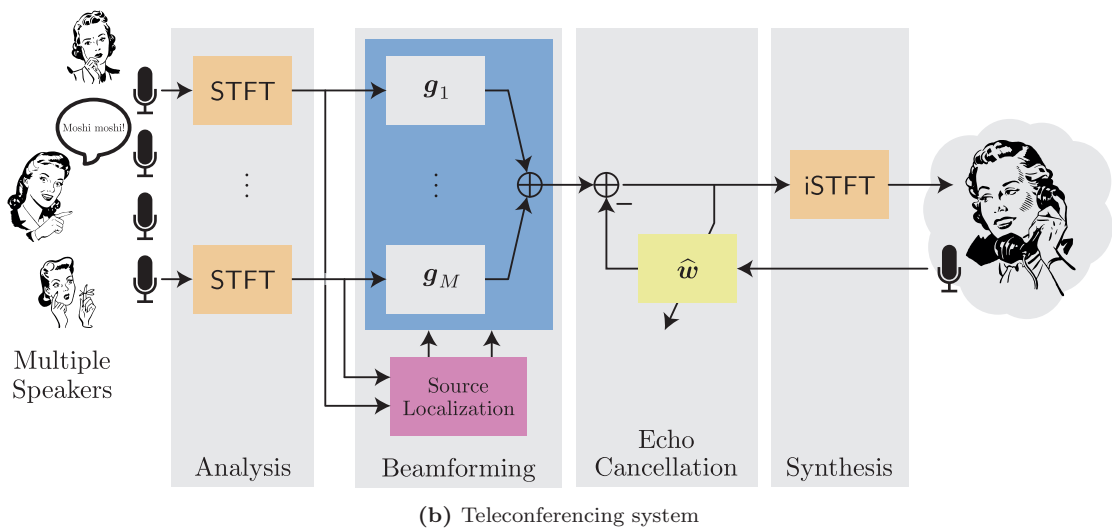
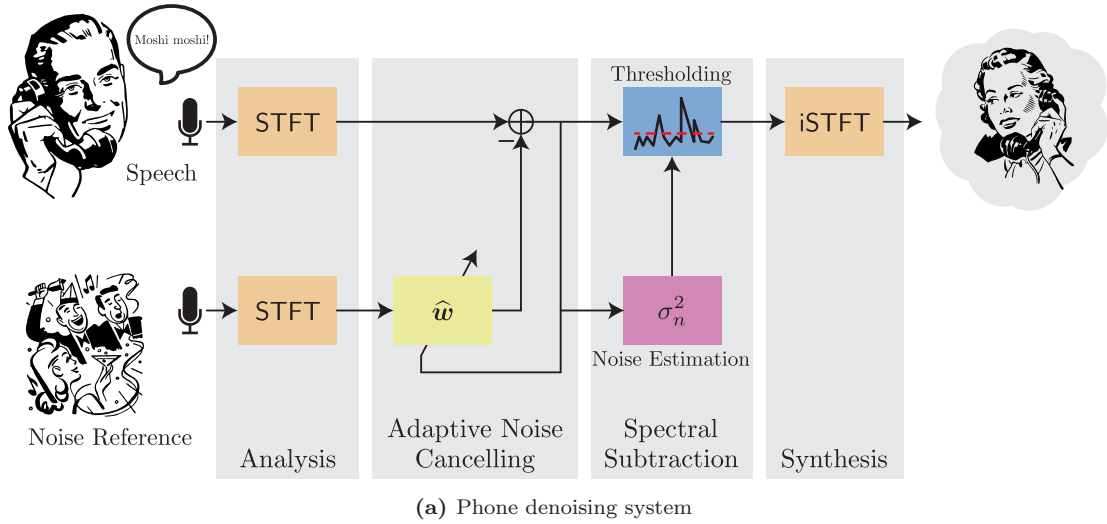


Figure 1.2: Two examples of signal processing applications illustrating the different blocks in the pipeline: representation (orange), enhancement (blue), estimation (pink), and adaptive processing (yellow). (a) A generic denoising system as found in mobile handsets. (b) A teleconferencing system with beamforming and echo cancellation.

Summary of Contributions in Chapter 2

- An algorithm to compute the Hadamard transform of a signal sparse in the transform domain using $O(K \log_2(\frac{N}{K}))$ samples in $O(K \log_2(K) \log_2(\frac{N}{K}))$ time.
- The asymptotic analysis of said algorithm based on sparse graph codes.
- A high performance C-language implementation of the algorithm.

Chapter 3 — Enhancement: Acoustic Rake Receivers The conventional signal processing wisdom that all reverberation is bad and should be combated is not supported by the human sensory system. On the contrary, humans display the ability to integrate early echoes constructively to improve speech intelligibility. We present in this chapter a class of beamformers that mimic this ability — the acoustic rake receivers. The idea to use echoes to improve robustness to noise and interference is well-known in wireless communications where receivers combining constructively the different multipath components have been proposed as early as 1958. Unlike digital radio signals, speech signals have not been carefully designed to be orthogonal to their shifts. Consequently, the design of acoustic rake receivers focuses more on the spatial structure than the temporal. A key aspect is the correspondence of early echoes in time with images sources in space, producing a tractable framework that doesn't require full knowledge of the room impulse response. We present several “intuitive” and optimal formulations of acoustic rake receivers. Frequency domain formulations allow for a simple formulation and efficient computations. In the time domain, tighter control over pre-echoes and the structure of the impulse response is possible. We also experiment with perceptually motivated formulations that relax some psycho-acoustically irrelevant constraints.

Summary of Contributions in Chapter 3

- We propose the concept of an acoustic rake receiver — A beamformer that constructively uses the early echoes in a room. Our work is motivated in part by the previous design of similar receivers in wireless communications, and in part by psychoacoustics results showing the benefits of early echoes in human audition.
- We show theoretically and numerically that the rake formulation of the maximum signal-to-interference-and-noise beamformer offers significant performance boosts in terms of noise and interference suppression.
- Beyond signal-to-noise ratio, we observe gains in terms of the *perceptual evaluation of speech quality* (PESQ) metric.

Chapter 4 — Estimation: FRI-based Direction of Arrival Finding A first step for many spatial audio processing algorithms (e.g. acoustic rake receivers) is to identify the location of sound sources. By observing carefully the phase of signals at the input of a microphone array, it is possible to identify the location of multiple sound sources. When the distance from the array to the sources is much larger than the inter microphone distance, only the direction of the sound can be retrieved — the so-called far-field case.

In this chapter, we present FRIDA—an algorithm for estimating directions of arrival of multiple wideband sound sources. FRIDA combines multi-band information coherently and achieves state-of-the-art resolution at extremely low signal-to-noise ratios. It works for arbitrary array layouts, but unlike the various steered response power and subspace methods, it does not require a grid search. FRIDA leverages recent advances in sampling signals with a finite rate of innovation. A parametric model is applied to the signal and the recovery of the directions is cast as a constrained optimization problem. It is based on the insight that for any array layout, the entries of the spatial covariance matrix can be linearly transformed into a uniformly sampled sum of sinusoids. While the problem is non-convex, an alternating optimization strategy is used to solve it efficiently. This approach can be modified to handle microphone signals directly, rather than the covariance matrix, which enables to treat the correlated sources case too. Both the planar and spherical cases are treated.

Summary of Contributions in Chapter 4

- We propose FRIDA, an algorithm for direction of arrival that is high-resolution, wide-band, works for correlated and uncorrelated signals, using arbitrary array layouts.
- We compare FRIDA to popular DOA finding algorithms through numerical simulations and experiments on recordings. We find that it offers much higher resolution, being able to resolve very close sources.

Chapter 5 — Adaptive Processing: The Recursive Hessian Sketch The fourth and last problem is that of adaptive filtering, a cornerstone of signal processing. The goal of the algorithm is to estimate an unknown filter having access to its input and noisy output only. The popular recursive least squares (RLS) algorithm solves a least squares problem recursively by augmenting the system every time new data arrives.

We introduce the recursive Hessian sketch, a new adaptive filtering algorithm based on sketching the same exponentially weighted least squares problem solved by RLS. The proposed algorithm solves this least squares minimization by conjugate gradient descent. Two preconditioners based on a sketch of the Hessian of the least squares objective are proposed to accelerate the convergence of the algorithm. These preconditioners are efficiently and recursively updated at random time intervals. Using the preconditioners, only a few iterations of conjugate gradient are required at each filter update to obtain fast convergence. The complexity of the proposed algorithm compares favorably to that of RLS. Its convergence properties are studied theoretically and through extensive numerical experiments. With an appropriate choice of parameters, its convergence speed falls between that of least mean squares and RLS adaptive filters, with less computations than the latter.

Summary of Contributions in Chapter 5

- We propose an adaptive filter algorithm based on sketching the Hessian of the recursive least squares system, allowing for a reduced complexity while preserving the convergence speed. The algorithm is based on a conjugate gradient base algorithm with two proposed preconditioners for accelerating the rate of convergence.
- Bounds on the rate of convergence of conjugate gradient are given for one of the preconditioners.

- Extensive numerical simulations for different signal models, filter lengths, and algorithm parameters further confirm the advantage of the method compared to conventional adaptive filters.
- We demonstrate that the algorithm is almost as fast as recursive least squares for a natural, non-stationary signal, and when the unknown filter is a room impulse response.

Chapter 6 — Tools and Methods for Reproducible Research in Computational Acoustics

This thesis condenses the output of about four years of research activity. During these years, it was necessary to develop several tools and methods to support the objective of this research, in particular software and hardware. Albeit essential to carry out experiments and assess algorithms and ideas, these tools are often neglected or relegated to the shadow in the expository part of the work. Considering the time and effort put into the elaboration of such tools, each researcher reinventing the wheel is a waste of resources. In an effort to curb this situation, we share in this chapter the hardware and software contributions made during this thesis. All the software source code as well as the hardware design files are also shared online under open licenses.

We go on to describe how a consistent methodology based on shared standard for documentation and sharing could help promote the adoption of good sharing practices. We also describe how a better system would incentivize and facilitate the sharing of these research artifacts. There is in particular much to learn from the open source software community that shares some characteristics of the research community. Rather than specific tools, we would like to emphasize policy and standardization.

Summary of Contributions in Chapter 6

- Pyroomacoustics — a python package for the development of audio signal processing algorithms geared toward indoor environments. The package includes a small room acoustics simulator based on the image source model, an STFT engine, as well as a host of reference implementations of popular beamforming, source localization, and adaptive filter algorithms. Written in modular object oriented style, the package allows to write clean and intuitive simulation scripts.
- Pyramic — a large microphone array made of 48 MEMS microphones spread on eight printed circuit boards that can be assembled into different shapes. The system sits on an FPGA-ARM combo tasked with reading the microphones and processing their output.
- CompactSix — a six microphone array based on an embedded Linux running on an ARM processor.
- Easy-DSP — a browser-based interface that allows to interact in real time with embedded multi-microphone systems such as Pyramic and CompactSix.

Chapter 2

Representation: Sparse Fast Hadamard Transform*

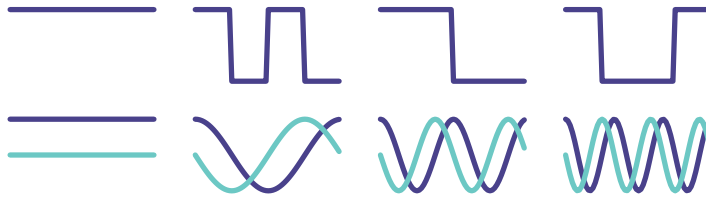
‘Contrariwise,’ continued Tweedledee, ‘if it was so, it might be; and if it were so, it would be: but as it isn’t, it ain’t. That’s logic.’

Through the Looking-Glass
LEWIS CARROLL

2.1 Introduction

The Walsh-Hadamard transform (WHT), the topic of this chapter, is a popular signal representation. Instead of representing signals as a sum of sinusoids, as Fourier analysis does, it uses a family of square functions taking values in $\{\pm 1\}$. It is a close relative of the discrete Fourier transform (DFT) and possesses many strikingly similar properties (see Section 2.3). Its recursive structure enables a fast computation, similar to the FFT algorithm for computing the discrete Fourier transform (DFT) of the signal, with a complexity $O(N \log_2(N))$ in the dimension of the signal N [69, 74]. Illustrated below are a few Hadamard (top) and Fourier functions (bottom, imaginary part in lighter color).

*The material in this chapter is the result of joint work with Saeid Haghghatshoar and Martin Vetterli [113, 114].



In an early application of the WHT to image processing, Pratt and co-authors observed that natural images have a sparse representation in the WHT domain, that is a few WHT coefficients concentrate most of the signal energy [104]. They propose to apply this property to transmit images more efficiently over a communication channel. Interestingly, they also find the method has the side benefit of an added robustness to noise in the channel.

This robustness property of the WHT was in fact first observed by Yates [135] and further investigated by Hotelling [65] in the context of weighing design in statistics. Let x be a vector of n values of interest that we can only measure through a channel that will add a noise vector ϵ . For clarity, let this noise vector be composed of statistically independent components with identical variance σ^2 , e.g. $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. Now, suppose that we can send Hx instead of x through the channel, where H is a Hadamard matrix with entries in $\{\pm 1\}$ and so that $\frac{1}{n}H^\top H = I$. Through the channel, we get the value $y = Hx + \epsilon$ to which we can apply H^\top again so that

$$\frac{1}{n}H^\top y = \frac{1}{n}H^\top (Hx + \epsilon) = x + \frac{1}{n}H^\top \epsilon.$$

This is equivalent to sending x through a channel with noise vector $\epsilon' = \frac{1}{n}H^\top \epsilon$, and thanks to the independence of the noise, $\epsilon' \sim \mathcal{N}(0, \frac{\sigma^2}{n} I)$. We obtain thus a reduction by a factor n in the noise variance by doing multiplexed measurements! This principle has been used in the design of spectroscopic instruments with lower noise [45].

Finally, the WHT also finds applications in multi-user communications in cellular networks via spreading sequences (CDMA) [1], compressed sensing [56], and cryptography [16, 52, 79]. For further details on its nice properties studied in different areas of mathematics, see Horadam [62].

2.1.1 Related Work

A number of recent publications have addressed the problem of computing the DFT of an N dimensional signal when it is sparse in the frequency domain [50, 51, 57, 58, 73]. In particular, it has been shown that the well-known computational complexity $O(N \log_2(N))$ of the FFT algorithm can be strictly improved under the sparsity assumption. Such algorithms are generally known as *sparse* FFT (sFFT) algorithms. In [49], Ghazi et al. developed a very low-complexity algorithm for computing the 2D-DFT of a $\sqrt{N} \times \sqrt{N}$ signal by extending the results of [58]. In a similar line of work, Pawar et al. [99, 100] used the subsampling property of the DFT to develop a low-complexity algorithm for recovering the nonzero frequency-domain components of a signal by using ideas from sparse-graph codes [109].

A closely related work, in spirit, to ours is the work on one-way functions by Goldreich et al. [52, 53] in theoretical computer science, where it was shown that the support recovery of the nonzero transform-domain coefficients is reduced to recovering the value of inner products of boolean vectors. In particular, an efficient algorithm was developed to speed up this computation [53]. Although our signal model and proposed recovery algorithm is completely different, we believe that our results would be of interest in these areas as well.

2.1.2 Main Contribution

We develop a novel algorithm for computing the WHT of an N dimensional signal with a sub-linear sparsity in the Hadamard domain. More precisely, we assume that the number of nonzero Hadamard-domain components $K = O(N^\alpha)$ scales sub-linearly in N for some $\alpha \in (0, 1)$. We first develop some useful properties of the WHT, specially the subsampling and the modulation property, which plays a vital role in the development of the algorithm. In particular, we show that subsampling in time domain allows to induce a well-designed aliasing pattern over the transform-domain components. In other words, it is possible to obtain a linear combination of a controlled collection of transform-domain components (aliasing), which creates interference between the nonzero components if more than one of them are involved in the resulting linear combination. Similar to [100] and by borrowing ideas from sparse-graph codes, we construct a bipartite graph by considering the nonzero values in the transform-domain as variable nodes and, by interpreting every induced aliasing pattern as a parity check constraint over the variables in the graph. We analyze the structure of the resulting graph assuming a random support model for the nonzero coefficients in the transform-domain. Moreover, we give an iterative peeling decoder to recover the nonzero components. In short, our proposed sparse fast Hadamard transform (SparseFHT) consists of a set of deterministic linear hash functions (explicitly constructed) and an iterative peeling decoder that uses the hash outputs to recover the nonzero transform-domain variables. It recovers the K -sparse WHT of the signal with a sample complexity (number of required time-domain samples) $O(K \log_2(\frac{N}{K}))$, total computational complexity $O(K \log_2(K) \log_2(\frac{N}{K}))$ and with a high probability approaching 1 for large dimension N .

2.1.3 Notations and Preliminaries

For an integer m , the set of all integers $\{0, 1, \dots, m-1\}$ is denoted by $[m]$. We use the small letter x for the time domain and the capital letter X for the transform-domain signal. For an N dimensional real-valued vector v , with $N = 2^n$ a power of two, the i -th components of v is equivalently represented by v_i or $v_{i_0, i_1, \dots, i_{n-1}}$, where i_0, i_1, \dots, i_{n-1} denotes the binary expansion of i , with i_0 and i_{n-1} being the least and the most significant bits. Also sometimes the real value assigned to v_i is not important to us and by v_i we simply mean the binary expansion associated to its index i ; however, the distinction must be clear from the context. \mathbb{F}_2 denotes the binary field consisting of $\{0, 1\}$ with summation and multiplication modulo 2. We also denote by \mathbb{F}_2^n the space of all n dimensional vectors with binary components, where the addition of two vectors is done component wise. The inner product of two n dimensional binary vectors u, v is defined by $\langle u, v \rangle = \sum_{t=0}^{n-1} u_t v_t$ with arithmetic over \mathbb{F}_2 , although $\langle \cdot, \cdot \rangle$ is not an inner product in the exact mathematical sense: for example, if $u = [1, 1, 1, 1]^\top$, then $\langle u, u \rangle = 0$, but $u \neq 0$. We often use Σ for a matrix with entries in \mathbb{F}_2 . We denote by Σ^\top the transpose of the matrix Σ . When Σ is non-singular, we use the shorthand notation $\Sigma^{-\top}$ for $(\Sigma^\top)^{-1}$. The null space of a matrix Σ is denoted by $\mathcal{N}(\Sigma) = \{u \mid \Sigma u = 0\}$.

For a signal $X \in \mathbb{R}^N$, the support of X is defined as $\text{supp}(X) = \{i \in [N] : X_i \neq 0\}$. The signal X is called K -sparse if $|\text{supp}(X)| = K$, where for a set $A \subset [N]$, $|A|$ denotes the cardinality or the number of elements of A . For a collection of N dimensional signals $\mathcal{S}_N \subset \mathbb{R}^N$, the sparsity of \mathcal{S}_N is defined as $K_N = \max_{X \in \mathcal{S}_N} |\text{supp}(X)|$.

Definition 2.1

Let \mathcal{S} be a class of signals $\mathcal{S} = \cup_{N=1}^{\infty} \mathcal{S}_N$, where \mathcal{S}_N denotes the subclass of all N -dimensional signals. \mathcal{S} is said to have a sub-linear sparsity if there is some $\alpha \in (0, 1)$ such that $K_N = O(N^\alpha)$, where K_N denotes the sparsity of the collection \mathcal{S}_N . We call α the sparsity index of class \mathcal{S} .

2.2 Main Result

Let us first describe the main result of this work in the following theorem.

Theorem 2.1

Let $\alpha \in (0, 1)$ be a fixed number. Suppose $N = 2^n$ is a power of two and assume $K = N^\alpha$. Let $x \in \mathbb{R}^N$ be a time-domain signal with a WHT $X \in \mathbb{R}^N$. Assume that X is a K -sparse signal in a class of signals with sparsity index α whose support is selected uniformly at random among all possible $\binom{N}{K}$ subsets of $[N]$ of size K . In addition, let the magnitude of the nonzero components of X be independently sampled from some arbitrary continuous distribution (that does not need to be known). Then, there is an algorithm that can compute X and has the following properties:

1. **Sample complexity:** The algorithm uses $CK \log_2(\frac{N}{K})$ time-domain samples of the signal x . C is a function of α and $C \leq (\frac{1}{\alpha} \vee \frac{1}{1-\alpha}) + 1$, where for $a, b \in \mathbb{R}_+$, $a \vee b$ denotes the maximum of a and b .
2. **Computational complexity:** The total number of operations needed in order to successfully decode all the nonzero spectral components or announce a decoding failure is $O(CK \log_2(K) \log_2(\frac{N}{K}))$.
3. **Success probability:** The algorithm correctly computes the K -sparse WHT X with a very high probability asymptotically approaching 1 as the dimension of the signal N tends to infinity, where the probability is taken over all random selections of the support of X . More importantly, the algorithm can find out whether the recovery succeeds or fails.

To prove Theorem 2.1, we distinguish between the very sparse case ($\alpha \in (0, \frac{1}{3}]$) and less sparse one ($\alpha \in (\frac{1}{3}, 1)$), where we implicitly assume that the algorithm knows the value of α , which might not be possible in some applications.

Fortunately, there is some underlying monotonicity in our algorithm that helps to solve this problem. More precisely, the algorithm designed for a specific value of $\alpha = \alpha^*$ can successfully recover all the transform-domain signals with sparsity index less than α^* . Thus, even if the value of α is unknown, we only need to design the algorithm for the largest possible value of α . However, the drawback is that the resulting sample and computational complexity might get much higher than the optimal algorithm that knows the exact value of α .

Fortunately, this problem can also be solved to obtain an optimal algorithm that does not need to know the value of α . The main observation is that in the sub-linear sparsity regime, where the number of nonzero components scales like $K = O(N^\alpha)$, the resulting sample and time complexity are on the order of $O(N^\alpha \log_2(N))$ and $O(N^\alpha \log_2(N)^2)$ respectively. This implies

that for $\alpha_1 < \alpha_2$, and for a sufficiently large signal dimension N , the algorithm designed for α_1 has negligible sample and computational complexity compared with the one designed for α_2 . Hence, we can use an adaptive strategy. We design our algorithm for small values of α . The algorithm will find out if the recovery was successful. If not, we increase the value of α and run a new algorithm for the new value of α , and we continue until the recovery is successful. In this way, we get a sample and computational complexity comparable with the algorithm that knows the exact value of α . The only drawback is that, in the adaptive scheme, the required number of time-domain samples gradually increases as we try larger value of α , thus it might not be useful in applications in which the number of samples should be a priori fixed.

Remark 2.1

In the very sparse regime ($\alpha \in (0, \frac{1}{3})$), we prove that for any value of α the success probability of the optimally designed algorithm is at least $1 - O(1/N^{3\alpha(C/2-1)})$, with $C = \lfloor \frac{1}{\alpha} \rfloor$ where for $u \in \mathbb{R}_+$, $\lfloor u \rfloor = \max\{n \in \mathbb{Z} : n \leq u\}$. It is easy to show that for every value of $\alpha \in (0, \frac{1}{3})$, the success probability can be lower bounded by $1 - O(N^{-\frac{3}{8}})$.

2.3 The Walsh-Hadamard Transform and its Properties

Let x be an $N = 2^n$ dimensional signal indexed with elements $m \in \mathbb{F}_2^n$. The N dimensional WHT of the signal x is defined by

$$X_k = \frac{1}{\sqrt{N}} \sum_{m \in \mathbb{F}_2^n} (-1)^{\langle k, m \rangle} x_m, \quad (2.1)$$

where $k \in \mathbb{F}_2^n$ denotes the corresponding binary index of the transform-domain component. Also, throughout the chapter, borrowing some terminology from the DFT, we call x the time-domain signal and we refer to X as the transform-domain, Hadamard-domain, frequency-domain, or spectral-domain signal. Also, note that the WHT is invertible and its inverse is given by the same expression as in Eq. (2.1) except that the roles of x_m and X_k are interchanged.

Notice that with our notation both the time-domain signal $x : \mathbb{F}_2^n \rightarrow \mathbb{R}$ and the transform-domain signal $X : \mathbb{F}_2^n \rightarrow \mathbb{R}$ are functions from the index set \mathbb{F}_2^n to reals. Therefore, the WHT given by the Eq. (2.1) maps the function x (time-domain signal) onto the function X (transform-domain signal). For simplicity of notation, we will use x_m for the time-domain and X_k for the frequency-domain functions with the convention that both m and k belong to the index set \mathbb{F}_2^n .

2.3.1 Basic Properties

In this section, we review some of the basic properties of the WHT. Some of the properties are not directly used in this work, but we include them for the sake of completeness. They can be of independent interest. The proofs of all the properties are provided in Appendix 2.A.

Property 2.1 (Shift/Modulation)

Let X_k be the WHT of the signal x_m and let $p \in \mathbb{F}_2^n$. Then

$$x_{m+p} \xleftrightarrow{\text{WHT}} X_k (-1)^{\langle p, k \rangle}.$$

The next property is more subtle and enables us to partially permute the Hadamard spectrum in a specific way by applying a corresponding permutation in the time domain. However, the collection of all such possible permutations is limited. We give a full characterization of all those permutations. Technically, this property is equivalent to finding permutations $\pi_1, \pi_2 : [N] \rightarrow [N]$ with corresponding permutation matrices Π_1, Π_2 such that

$$\Pi_2 H_N = H_N \Pi_1, \quad (2.2)$$

where H_N is the Hadamard matrix of order N , and where the permutation matrix corresponding to a permutation π is defined by $(\Pi)_{i,j} = 1$ if and only if $\pi(i) = j$, and zero otherwise. The identity (2.2) is equivalent to finding a row permutation of H_N that can be equivalently obtained by a column permutation of H_N .

Property 2.2

All of the permutations satisfying (2.2) are described by the elements of

$$\text{GL}(n, \mathbb{F}_2) = \{A \in \mathbb{F}_2^{n \times n} \mid A^{-1} \text{ exists}\},$$

the set of $n \times n$ non-singular matrices with entries in \mathbb{F}_2 .

Remark 2.2

The total number of possible permutations in Property 2.2, is $\prod_{i=0}^{n-1} (N - 2^i)$, which is a negligible fraction of all $N!$ permutations over $[N]$.

Property 2.3 (Permutation)

Let $\Sigma \in \text{GL}(n, \mathbb{F}_2)$. Assume that X_k is the WHT of the time-domain signal x_m . Then

$$x_{\Sigma m} \xleftrightarrow{\text{WHT}} X_{\Sigma^{-\top} k}.$$

Notice that $y_m = x_{\Sigma m}$ is the function given by the composition of the function x and the index transformation i_Σ , i.e., $y = x \circ i_\Sigma$, where for $m \in \mathbb{F}_2^n$, $i_\Sigma(m) = \Sigma m$ is the multiplication of the matrix Σ with the index vector m . Moreover, any $\Sigma \in \text{GL}(n, \mathbb{F}_2)$ is a bijection from \mathbb{F}_2^n to \mathbb{F}_2^n , thus $x_{\Sigma m}$ is simply a signal obtained by permuting the components of the signal x_m .

The next property is that of downsampling/aliasing. Notice that for a vector x of dimension $N = 2^n$, we index every component by a binary vector of length n , namely, $x_{m_0, m_1, \dots, m_{n-1}}$. To subsample this vector along dimension i , we freeze the i -th component of the index to either 0 or 1. For example, $x_{0, m_1, \dots, m_{n-1}}$ is a 2^{n-1} dimensional vector obtained by subsampling the vector x_m along the first index.

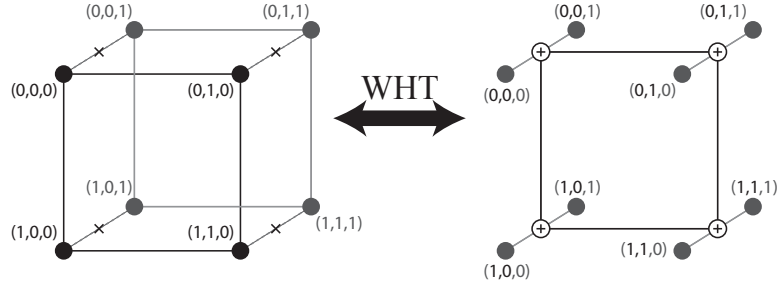


Figure 2.1: Illustration of the downsampling property on a hypercube for $N = 2^3$. The two cubes represent the time-domain (left) and the Hadamard-domain (right) signal. We decide to drop all the nodes whose third coordinate is ‘1’. We illustrate this by adding a ‘x’ on the edges connecting those vertices through the third coordinate. This is equivalent to summing up vertices along the corresponding edges in the Hadamard domain.

Property 2.4 (Downsampling/Aliasing)

Suppose that x is a vector of dimension $N = 2^n$ indexed by the elements of \mathbb{F}_2^n and assume that $B = 2^b$, where $b \in \mathbb{N}$ and $b < n$. Let

$$\Psi_b = [\mathbf{0}_{b \times (n-b)} \ I_b]^\top, \quad (2.3)$$

be the subsampling matrix that freezes the first $n - b$ components in the index to 0. If X_k is the WHT of x , then

$$x_{\Psi_b m} \xleftrightarrow{\text{WHT}} \sqrt{\frac{B}{N}} \sum_{j \in \mathcal{N}(\Psi_b^\top)} X_{\Psi_b k + j},$$

where $m, k \in \mathbb{F}_2^b$ denote the corresponding binary indices of the time and frequency components, and $x_{\Psi_b m}$ is a $B = 2^b$ dimensional signal labelled with $m \in \mathbb{F}_2^b$.

Recall that by our notation, $y_m = x_{\Psi_b m}$ is a function $y : \mathbb{F}_2^b \rightarrow \mathbb{R}$ given by $y = x \circ i_{\Psi_b}$, where $i_{\Psi_b} : \mathbb{F}_2^b \rightarrow \mathbb{F}_2^n$ is the index transformation given by $i_{\Psi_b}(m) = \Psi_b m$. It is not difficult to check that $\Psi_b m$, which is the multiplication of $m \in \mathbb{F}_2^b$ with the matrix Ψ_b of dimension $n \times b$, gives an index in \mathbb{F}_2^n which is the right argument for the function x . Also, index $\Psi_b k + j$ with $j \in \mathcal{N}(\Psi_b^\top)$ gives the suitable index for the function X . Notice that Property 2.4 can be simply applied for any matrix Ψ_b that subsamples any set of indices of length b but not necessarily the b last ones.

To give an intuitive explanation of the downsampling property, notice that the elements of \mathbb{F}_2^n can be visualized as the vertices of an n -dimensional hypercube. This property implies that downsampling along some of the dimensions in the time domain is equivalent to summing up all the spectral components along *the same dimensions* in the spectral domain. This is illustrated in Figure 2.1 for dimension $n = 3$.

In a general downsampling procedure, we can replace the frozen indices by an arbitrary but fixed binary pattern. The only difference is that, instead of summing the aliased spectral

components, we should also take into account the suitable $\{+, -\}$ sign patterns, i.e., we have

$$x_{\Psi_b m+p} \xleftrightarrow{\text{WHT}} \sqrt{\frac{B}{N}} \sum_{j \in \mathcal{N}(\Psi_b^\top)} (-1)^{\langle p, j \rangle} X_{\Psi_b k+j},$$

where p is a binary vector of length n with b zeros at the end. To visualize this property, consider Figure 2.1, where we have a signal over a 3-dimensional cube and we subsample it along the third dimension, i.e., we keep only 4 variables with the third index equal to 0. Notice that these variables lie on a 2-dimensional (square) face of the cube that corresponds to a subsampling with $p = 000$. Instead, we can use $p = 001$ for subsampling and this value of p will select all 4 variables on the other face of the cube corresponding to those variables with the third index equal to 1. This face of the cube is a square parallel to the square corresponding to $p = 000$.

The final property we give is not used in the algorithms but is included here for completeness. This property corresponds to the convolution theorem for the DFT applied to a signal on the hypercube.

Property 2.5 (Convolution)

Let x, y be signals in \mathbb{F}_2^n . Their convolution is given by

$$(x \star y)_m = \sum_{u \in \mathbb{F}_2^n} x_u y_{u+m}.$$

The convolution of x and y is equivalent to multiplication in the transform domain

$$(x \star y)_m \xleftrightarrow{\text{WHT}} X_k Y_k$$

2.4 Hadamard Hashing Algorithm

By applying the basic properties of the WHT, we can design suitable hash functions in the spectral domain. The main idea is that, to compute the output of a hash function, we do not need to have access to the spectral components because it can be computed by low-complexity operations that are directly applied to the time-domain samples of the signal.

Algorithm 2.1 FastHadamardHashing(x, N, Σ, p, B)

Require: Signal x of dimension $N = 2^n$, Σ and p and given number of output bins $B = 2^b$ in a hash.

Ensure: U contains the hashed Hadamard spectrum of x .

$$u_m = x_{\Sigma \Psi_b m+p}, \text{ for } m \in \mathbb{F}_2^b.$$

$$U = \sqrt{\frac{N}{B}} \text{FastHadamard}(u_m, B).$$

Proposition 2.1 (Hashing)

Assume that $\Sigma \in \text{GL}(n, \mathbb{F}_2)$ and $p \in \mathbb{F}_2^n$. Let $N = 2^n$, $b \in \mathbf{N}$, $B = 2^b$ and let $m, k \in \mathbb{F}_2^b$ denote the time and frequency indices of a B dimensional signal and its WHT defined by

$$u_{\Sigma,p}(m) = \sqrt{\frac{N}{B}} x_{\Sigma\Psi_b m+p}.$$

Then, the length B Hadamard transform of $u_{\Sigma,p}$ is given by

$$U_{\Sigma,p}(k) = \sum_{j \in \mathbb{F}_2^b \mid \mathcal{H}j=k} X_j (-1)^{\langle p, j \rangle}, \quad (2.4)$$

where \mathcal{H} is the index hashing operator defined by

$$\mathcal{H} = \Psi_b^\top \Sigma^\top, \quad (2.5)$$

where Ψ_b is as in (2.3). Note that the index of components in the sum (2.4) can be explicitly written as a function of the bin index k as follows:

$$j = \Sigma^{-\top} \Psi_b k + q, \quad q \in \mathcal{N}(\mathcal{H}).$$

The proof simply follows from the properties 2.1, 2.3, and 2.4.

Using Proposition 2.1, we give Algorithm 2.1 for computing the hashed Hadamard spectrum. It chooses B bins for hashing the spectrum, chooses B samples of the time-domain signal according to the subsampling parameters Σ and p , and uses an FFT-like fast Hadamard transform (FHT) to compute the hash output with $O(B \log_2 B)$ operations.

2.4.1 Properties of Hadamard Hashing

In this part, we review some properties of the hashing algorithm that are crucial for developing an iterative peeling decoding algorithm that recovers the nonzero spectral values. We mainly explain how it is possible to identify collisions between the nonzero spectral coefficients that are hashed to the same bin and to estimate the support of non-colliding components.

Let us consider $U_{\Sigma,p}(k)$ for two cases: $p = 0$ and some $p \neq 0$. It is easy to see that in the former $U_{\Sigma,p}(k)$ is obtained by summing all the spectral variables hashed to bin k (those whose index j satisfies $h_\Sigma(j) = \mathcal{H}j = k$); whereas in the latter the same variables are added together weighted by $(-1)^{\langle p, j \rangle}$. Let us define the following ratio test

$$r_{\Sigma,p}(k) = \frac{U_{\Sigma,p}(k)}{U_{\Sigma,0}(k)}.$$

When the sum in $U_{\Sigma,p}(k)$ contains only one nonzero component, it is easy to see that $|r_{\Sigma,p}(k)| = 1$ for ‘any value’ of p . However, if there is more than one nonzero component in the sum, and assuming that those nonzero values are jointly sampled from a continuous distribution, we can show that $|r_{\Sigma,p}(k)| \neq 1$ for at least some values of p . In fact, $n - b$ well-chosen values of p enable us to detect whether there is only one, or more than one nonzero component in the sum.

When there is only one $X_{j'} \neq 0$ hashed to the bin k ($h_\Sigma(j') = k$), the result of the ratio test is precisely 1 or -1 , depending on the value of the inner product between j' and p . In particular,

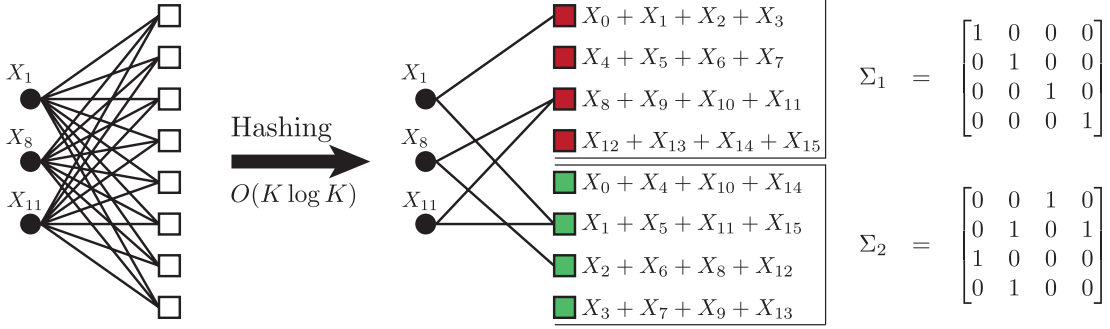


Figure 2.2: On the left, bipartite graph representation of the WHT for $N = 8$ and $K = 3$. On the right, the underlying bipartite graph after applying $C = 2$ different hashing produced by plugging Σ_1, Σ_2 in (2.5) with $B = 4$. The variable nodes (\bullet) are the nonzero spectral values to be recovered. The white check nodes (\square) are the original time-domain samples. The colored squares are new check nodes after applying Algorithm 2.1.

we have

$$\langle p, j' \rangle = \mathbb{1}_{\{r_{\Sigma, p}(k) < 0\}}, \quad (2.6)$$

where $\mathbb{1}_{\{t < 0\}} = 1$ if $t < 0$, and zero otherwise. Hence, if for $n - b$ well-chosen values of p , the ratio test results in 1 or -1 , implying that there is only one nonzero spectral coefficient in the corresponding hash bin, it is even possible to identify the position of this nonzero component. We formalize this result in the following proposition proved in Appendix 2.B.

Proposition 2.2 (Collision Detection / Support Estimation)

Let $\Sigma \in \text{GL}(n, \mathbb{F}_2)$ and let $\sigma_i, i \in [n]$ denote the columns of Σ .

1. If for all $d \in [n - b]$, $|r_{\Sigma, \sigma_d}(k)| = 1$, then almost surely there is only one nonzero spectral value in the bin indexed by k . Moreover, if we define

$$\hat{v}_d = \begin{cases} \mathbb{1}_{\{r_{\Sigma, \sigma_d}(k) < 0\}} & d \in [n - b], \\ 0 & \text{otherwise,} \end{cases} \quad (2.7)$$

the index of the unique nonzero value is given by

$$j = \Sigma^{-\top}(\Psi_b k + \hat{v}). \quad (2.8)$$

2. If there exists a $d \in [n - b]$ such that $|r_{\Sigma, \sigma_d}(k)| \neq 1$, then the bin k contains more than one nonzero coefficient.

2.5 Sparse Fast Hadamard Transform

In this section, we give a brief overview of the main idea of Sparse Fast Hadamard Transform (SparseFHT). In particular, we explain the peeling decoder that recovers the nonzero spectral components, and analyze its computational complexity.

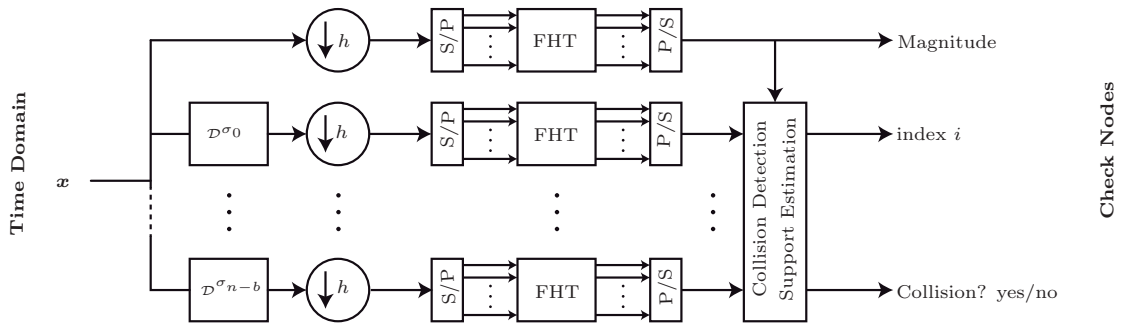


Figure 2.3: A block diagram of the SFHT algorithm in the time domain. The downsampling plus small-size low-complexity FHT blocks compute different hash outputs. Delay blocks denote an index shift by σ_i before hashing. The S/P and P/S are serial-parallel and parallel-serial blocks to emphasize that the FHT operates on the whole signal at once. The collision detection/support estimation block implements Proposition 2.2 to identify if there is a collision and if not to find the index of the only nonzero value. The recovered index i is not valid when there is a collision.

2.5.1 Explanation of the Algorithm

In coding theory, it is common to represent a code over a bipartite graph with variable and check nodes. There is exactly one variable node per code bit and one check node per parity check constraint in the code. A variable node is connected to a check node if and only if it appears in the parity check equation corresponding to that check node. Such a graph is called *sparse* if the number of edges is in the order of the number of nodes.

We can slightly extend the bipartite graph representation to include the WHT. To explain this further, let us consider Eq. (2.1). From the symmetry of WHT, we have

$$x_m = \frac{1}{\sqrt{N}} \sum_{k \in \mathbb{F}_2^n} (-1)^{\langle m, k \rangle} X_k. \quad (2.9)$$

Eq. (2.9) states that knowing a time-domain sample x_m puts a linear constraint on the spectral-domain components X_k . Using the terminology of coding theory, we interpret these linear constraints as parity check constraints over X_k . For example, for $m = 0$, the resulting parity check equation implies that the sum of all the components of X must be equal to the first time-domain sample x_0 multiplied by \sqrt{N} . We associate a bipartite graph representation as follows: we associate variable nodes to the components of X (code bits in coding theory), and corresponding to every known time-domain sample, we add a check node to represent the resulting linear constraint. In particular, if we only keep the nonzero spectral variables, we obtain the induced bipartite graph over these variables. With this picture in mind, we can formulate the recovery of the nonzero spectral values as a decoding problem over this induced bipartite graph.

It is not difficult to see that for the WHT, the induced bipartite graph on the nonzero spectral values is a complete (dense) bipartite graph because any variable node is connected to all the check nodes. This has been depicted in the left part of Figure 2.2, where $\{X_1, X_8, X_{11}\}$ are the only nonzero variables in the spectral domain and each check constraint corresponds to the value of a specific time-domain sample. It is also implicitly assumed that the support (position of nonzero variables) of X is known, e.g., $\{1, 8, 11\}$ in Figure 2.2. At the moment, it is not clear how we can obtain the position of these nonzero variables. In the final version of the algorithm,

this is done by applying Proposition 2.2.

For codes on sparse bipartite graphs, there exists a collection of low-complexity belief propagation algorithms to efficiently recover the code bits given the value of check nodes observed via a noisy channel [109]. Unfortunately, the graph corresponding to WHT is dense, and probably not suitable for any of these belief propagation algorithms.

As explained in Section 2.4, by subsampling the time-domain components of the signal, it is possible to hash the spectral components in different bins as depicted for the same signal X in the right part of Figure 2.2. The advantage of the hashing must be clear from this picture. The idea is that via hashing, we can obtain a new representation with a *sparse* bipartite graph. In some sense, hashing ‘*sparsifies*’ the underlying bipartite graph. It is also important to note that in the former representation, the output value of every parity check equation is an already known time-domain sample of the signal, thus no extra effort is necessary to compute them. For the latter representation obtained via hashing, the value of parity checks (hash outputs) are not a priori known but fortunately, they can be computed by using low-complexity operations on a small subset of time-domain samples as explained in Proposition 2.1.

We propose the following iterative algorithm to recover the nonzero spectral variables over the bipartite graph induced by hashing. The algorithm first tries to find a degree-one check node. Using the terminology of [100], we call such a check node a *singleton*. This singleton check is connected to only one nonzero variable. Using Proposition 2.2, we can find the position and the value of this nonzero variable. This enables us to subtract (peel off) this variable from all the other check nodes that are connected to it, thus we can remove this variable node from the graph along with all the edges that are connected to it. Consequently, the degree of all the check nodes that are connected to this variable node decreases by one, thus there is a chance that another singleton be found. Also, removing the edges, creates an isolated (degree zero) check node that we call a *zeroton*.

The algorithm proceeds to peel off one singleton at a time until all the check nodes are zerotons (decoding succeeds) or all the remaining check nodes have degrees greater than one (we call them *multiton*) and the algorithm fails to completely recover all the nonzero spectral values. A more detailed pseudo-code of the proposed iterative algorithm is given in Algorithm 2.2.

2.5.2 Complexity Analysis

Fig. 2.3 shows a full block diagram of the SparseFHT algorithm. Using this block diagram, we prove part 1 and 2 of Theorem 2.1 regarding the sample and the computational complexity of the SparseFHT algorithm. The proof of the last part of Theorem 2.1, regarding the success probability of the algorithm, is the subject of Sections 2.6 and 2.7.

Computational Complexity: As we further explain in Sections 2.6 and 2.7, depending on the sparsity index of the signal α , we use C different hash blocks, where $C \leq (\frac{1}{\alpha} \vee \frac{1}{1-\alpha}) + 1$, and where each hash has $B = 2^b$ different output bins. We always select B to be equal to the number of nonzero spectral values K . This keeps the average number of nonzero components per bin $\beta = \frac{K}{B}$ equal to 1. As computing the hash outputs via an FHT block of size B needs $O(B \log_2(B))$ operations, selecting $K = B$, the resulting computational complexity is $O(K \log_2(K))$. Moreover, in our algorithm, we need to compute any hash output with $n - b = \log_2(\frac{N}{B})$ different shifts in order to make a collision detection/support estimation. Thus, the computational cost for each hash is $O(K \log_2(K) \log_2(\frac{N}{K}))$. As we need to compute C different hash blocks, the total computational complexity of computing hash outputs for each iteration will be $O(CK \log_2(K) \log_2(\frac{N}{K}))$.

Algorithm 2.2 SparseFHT(x, N, K, C, L, Σ)

Require: Input signal x of length $N = 2^n$. Sparsity K . Hash count C . Number of iterations of decoder L . Array Σ of C matrices in $\text{GL}(n, \mathbb{F}_2)$, $\Sigma_c = [\sigma_{c,1} \mid \cdots \mid \sigma_{c,n}]$, $\sigma_{c,i} \in \mathbb{F}_2^n$.

Ensure: X contains the sparse Hadamard spectrum of x .

```

 $B = O(K)$ 
 $D = n - b + 1$ 
for  $c = 1, \dots, C$  do
   $U_{c,0} = \text{FastHadamardHashing}(x, N, \Sigma_c, 0, B)$ 
  for  $d = 1, \dots, D$  do
     $U_{c,d} = \text{FastHadamardHashing}(x, N, \Sigma_c, \sigma_{c,d}, B)$ 
  end for
end for
for  $l = 1, \dots, L$  do
  for  $c = 1, \dots, C$  do
    for  $k = 0, \dots, B - 1$  do
      if  $U_{c,0,k} = 0$  then
        continue to next  $k$ 
      end if
       $\hat{v} \leftarrow 0$ 
      for  $d = 1, \dots, D$  do
        if  $U_{c,d,k}/U_{c,0,k} = -1$  then
           $\hat{v}_{d-1} \leftarrow 1$ 
        else if  $U_{c,d,k}/U_{c,0,k} \neq 1$  then
          continue to next  $k$ 
        end if
      end for
       $i \leftarrow \Sigma_c^{-\top} (\Psi_b k + \hat{v})$ 
       $X_i \leftarrow U_{c,0,k}$ 
      for  $c' = 1, \dots, C$  do
         $j \leftarrow \Psi_b^\top \Sigma_{c'}^\top i$ 
         $U_{c',0,j} \leftarrow U_{c',0,j} - X_i$ 
        for  $d' = 1, \dots, D$  do
           $U_{c',d',j} \leftarrow U_{c',d',j} - X_i (-1)^{\langle \sigma_{c',d'}, i \rangle}$ 
        end for
      end for
    end for
  end for
end for

```

After computing the hash outputs (output of hash bins), we do a ratio test to find hashes with only one nonzero component. In total, we have $CK \log_2(\frac{N}{K})$ output hash bins, thus this step needs $O(CK \log_2(\frac{N}{K}))$ operations. If the ratio test is successful for a specific bin indexed with k , we compute the binary vector \hat{v} for this bin according to Eq. (2.7). The next step is to find the location of the nonzero component j using Eq. (2.8), i.e., $j = \Sigma^{-\top}(\Psi_b k + \hat{v})$. This can be split in two parts: finding $\Sigma^{-\top} \Psi_b k$ for the bin index k , and computing $\Sigma^{-\top} \hat{v}$ for the binary index \hat{v} . The former can be calculated offline for every hash bin k , thus we focus on the latter. From Eq. (2.7), it is seen that the last b entries of \hat{v} are zero, thus we have $\Sigma^{-\top} \hat{v} = \sum_{i \in [n-b]} \hat{v}_i \bar{\sigma}_i$, where $\bar{\sigma}_i$, $i \in [n]$ are the columns of $\Sigma^{-\top}$. This sum can be computed using at most $(n-b)$ multiplications and bitwise XOR operations. During the whole runtime of the algorithm, this calculation is done only K times corresponding to K nonzero variables to be peeled off, thus the resulting complexity is $O(K \log_2(\frac{N}{K}))$ (recall that $n-b = O(\log_2(\frac{N}{K}))$). Hence, the total computational complexity of every iteration is of the order $O(CK \log_2(K) \log_2(\frac{N}{K}))$. We will explain in Sections 2.6 and 2.7 how the algorithm terminates in a fixed number of iterations independent of the value of α and the dimension of the signal N . Therefore, the total computational complexity of the algorithm is $O(CK \log_2(K) \log_2(\frac{N}{K}))$.

Sample Complexity: Assuming $K = B$, computing each hash with $n-b$ different shifts needs $K \log_2(\frac{N}{K})$ time-domain samples. Therefore, the total sample complexity is $CK \log_2(\frac{N}{K})$.

2.6 Performance Analysis of the very Sparse Regime

In Section 2.5.1, we explained how by applying Proposition 2.1, we can hash the spectral-domain components in a collection of bins and how this can be represented by a bipartite graph. In this section, we consider the very sparse regime, where $\alpha \in (0, \frac{1}{3}]$. We show that by assuming a random support model for nonzero spectral components and for a careful design of hash functions, we obtain a random bipartite graph that behaves similarly to the ensemble of LDPC bipartite graphs. We also show that running the peeling decoder to recover the nonzero spectral components is equivalent to the belief propagation (BP) decoding for a binary erasure channel (BEC). In particular, we prove that the error (decoding failure) probability can be asymptotically characterized by a ‘density evolution’ (DE) equation, thus enabling for a perfect analysis of the peeling decoder. We use the following steps to rigorously analyze the performance of the decoder in the very sparse regime:

1. We explain the construction of suitable hash functions depending on the value of $\alpha \in (0, \frac{1}{3}]$.
2. We rigorously analyze the structure of the induced bipartite graph obtained by treating the nonzero spectral components as variable nodes and the output of hash functions as check nodes. In particular, we prove that the resulting graph is a fully-random left-regular bipartite graph similar to the regular LDPC ensemble. We also obtain variable- and check-degree distribution polynomials for this graph.
3. At every stage, the peeling decoder recovers some of the variable nodes, removing all the edges incident to those variable nodes. We use Wormald’s method, given in [134], to prove the concentration of the number of unpeeled edges around its expected value, which we also characterize. Wormald’s method, as exploited in [80], uses the differential equation approach to track the evolution of the number of edges in the underlying bipartite graph. Specifically, it shows that the number of edges at every step of the algorithm is very well concentrated around the solution of the associated differential equation.

4. Wormald's method gives a concentration bound on the number of remaining edges, as far as their count is a fixed ratio $\gamma \in (0, 1)$ of the initial edges in the graph. Another expander argument, as in [80], is necessary to show that if the peeling decoder peels $1 - \gamma$ fraction of the edges successfully, it can continue to peel off all the remaining edges with a very high probability.

2.6.1 Hash Construction

For the very sparse regime, $\alpha \in (0, \frac{1}{3}]$, consider those values of α that are equal to $\frac{1}{C}$ for some positive integer $C \geq 3$. For $\alpha = \frac{1}{C}$, we build C different hash functions as follows. Let x be an N dimensional time-domain signal with a WHT X , where $N = 2^n$ and let $b = \frac{n}{C}$. We label the components of the vector X by n dimensional binary vector from \mathbb{F}_2^n . We design C different subsampling operators, where the i -th one keeps all the indices ib up to $(i+1)b - 1$ and sets the other indices equal to zero. More precisely, using the terminology of Proposition 2.1, the i -th hashing operator is given by

$$\mathcal{H}_i = [\mathbf{0}_{b \times ib} \ I_b \ \mathbf{0}_{b \times (n - (i+1)b)}],$$

where I_b is the identity matrix of order b . Let $x_0^{n-1} \in \mathbb{F}_2^n$ be the binary labeling of the elements of the signal x . Equivalent to the C different subsampling operators, we can consider functions $h_i, i \in [C]$, where

$$h_i(x_0^{n-1}) = (x_{ib}, x_{ib+1}, \dots, x_{ib+b-1}).$$

Ignoring the multiplicative constants, we can see from Eq. (2.4) that the spectral component labelled with X_0^{n-1} is hashed to the bin labelled with $h_i(X_0^{n-1}) \in \mathbb{F}_2^b$ in the i -th hash.

As we explain in Section 2.6.3 (Proposition 2.9), we need at least $C = 3$ hashes for the peeling algorithm to work successfully and that is the main reason this construction works for $\alpha \leq \frac{1}{3}$. For intermediate values of α , those not equal to $\frac{1}{C}$ for some integer C , we can construct $\lfloor \frac{1}{\alpha} \rfloor$ hashes with $B = 2^{\lfloor n\alpha \rfloor}$ output bins and one hash with a smaller number of output bins. Thus, the required number of hashes is at most $1 + \lfloor \frac{1}{\alpha} \rfloor$.

2.6.2 Random Bipartite Graph Construction

Random Support Models

In the very sparse regime, the number of nonzero spectral components is $K = O(N^\alpha)$ for some $\alpha \in (0, \frac{1}{3}]$. For a given (K, N) , we define RS1(K, N) as the class of Hadamard-domain signals whose support is selected uniformly at random from the set of all $\binom{N}{K}$ possible supports of size K . Model RS1 is equivalent to selecting K out of N objects (locations of nonzero values) at random without replacement. Assuming that the indices for the support are selected independently but with replacement, we obtain another model that we call RS2(K, N). The size of a random support in RS2(K, N) is itself a random variable less than or equal to K . The following proposition, proved in Appendix 2.C, shows that in the sub-linear sparsity regime, RS1 and RS2 are essentially equivalent.

Proposition 2.3

Consider the random support model RS2(K, N), where $K = N^\alpha$ for some fixed $0 < \alpha < 1$ and let H be the random size of the support set. Asymptotically as N tends to infinity $\frac{H}{K}$ converges to 1 in probability.

'Balls and Bins' Model $\mathcal{G}(K, B, C)$

Let us consider C disjoint sets of check nodes S_1, S_2, \dots, S_C of the same size $|S_i| = B$. A graph G in the ensemble $\mathcal{G}(K, B, C)$ is a bipartite graph with K variable nodes on the left and $C \times B$ check nodes $\cup_{i=1}^C S_i$ on the right. Each variable node v in G , independently from other variable nodes, is connected to check nodes $\{s_1, s_2, \dots, s_C\}$, where every $s_i \in S_i$ is selected uniformly at random from S_i , independent of the other s_j . Hence, every edge e in G can be labelled as (v, c) , where $v \in [K]$ is a variable node and c is a check node in one of S_1, S_2, \dots, S_C . If two different variable nodes are connected to exactly the same check nodes, we consider them equivalent and we keep only one of them. By construction, all the resulting bipartite graphs in the ensemble are left regular with the variable degree C but the check node degree is not fixed.

Ensemble of Graphs Generated by Hashing

In Section 2.6.1, we explained the subsampling operator and the hash construction for the very-sparse regime. As we described in Section 2.5.1, we can represent the hashing operation by a bipartite graph. In this section, our aim is to study the resulting bipartite graph for the proposed hash construction.

Recall that, the subsampling operator h_i is given by

$$h_i(x_0^{n-1}) = (x_{ib}, x_{ib+1}, \dots, x_{ib+b-1}),$$

which maps the spectral component labeled with $X_0^{n-1} \in \mathbb{F}_2^n$ into the bin labelled with $h_i(X_0^{n-1}) \in \mathbb{F}_2^b$. Notice that by this hashing scheme there is a one-to-one relation between a spectral element labelled with X_0^{n-1} and its bin indices in different hashes $(h_0(X_0^{n-1}), h_1(X_0^{n-1}), \dots, h_{C-1}(X_0^{n-1}))$.

Now, suppose X_1, X_2, \dots, X_K is a subset of binary indices in \mathbb{F}_2^n that is selected uniformly at random from all the subsets of \mathbb{F}_2^n of size K , and denotes the position of nonzero spectral components. For these K variables and hash functions h_i , we can associate a bipartite graph as follows. We consider K variable nodes corresponding to X_i , $i \in [K]$, and C different set of check nodes S_0, S_1, \dots, S_{C-1} each of size $B = 2^b$. The check nodes in each S_i are labelled by elements of \mathbb{F}_2^b . For each variable X_i we consider C different edges connecting X_i to check nodes labelled with $h_j(X_i) \in S_j$, $j \in [C]$.

As X_i are selected at random *without* replacement (according to RS1), they are not independent and the resulting bipartite graph is not compatible with Balls and Bins model explained in Section 2.6.2. This makes the analysis difficult. We solve this problem in two steps. First, in Proposition 2.4, we prove that we can still obtain a graph compatible with Balls and Bins model $\mathcal{G}(K, B, C)$ if we use RS2 instead of RS1. This is equivalent to sampling the indices randomly and independently but with replacement. Second, in Proposition 2.5, we prove that for a large dimension N , the failure probability of our proposed algorithm over RS1 model is upper bounded by the failure probability over $\mathcal{G}(K(1 + \epsilon), B, C)$, i.e., the Balls and Bins model with a slightly higher number of variables.

Proposition 2.4

Let $h_i : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^b$, $i \in [C]$ be as explained before. Let $\{V_j : j \in [K]\}$ be a random set from the ensemble RS2(K, N) for $N = 2^n$. The bipartite graph associated with variables V_j and hash functions h_i is a graph from ensemble $\mathcal{G}(K, B, C)$, where $B = 2^b$.

Proof.

Recall that in the bipartite graph representation, we assign a variable node to each V_j and

consider a set of check node $\cup_{i \in [C]} S_i$, where S_i is the set of all check nodes corresponding to all hash outputs in hash i . As $\{V_j : j \in [K]\}$ belongs to the ensemble $\text{RS2}(N, K)$, all the variables V_j are independent from each other. Hence, for a fixed hash function h_i , the variables $h_i(V_j), j \in [K]$, are also independent of each other. This implies that in the resulting bipartite graph, different variable nodes select their corresponding check node in S_i independent of each other.

Now consider a specific variable node V_k . This variable node is connected to hash bin $h_i(V_k)$ in the i -th hash. As V_k is a uniformly distributed random variable over \mathbb{F}_2^n , we can represent it by a binary vector B_0^{n-1} whose components B_i are like i.i.d. unbiased bits. For the constructed hash functions, we can see that the corresponding hash indices

$$h_0(B_0^{n-1}), h_1(B_0^{n-1}), \dots, h_{C-1}(B_0^{n-1})$$

are independent from one another because they depend on disjoint subsets of B_0^{n-1} . Moreover, each $h_i(B_0^{n-1})$ is uniformly distributed over \mathbb{F}_2^b . This implies that every variable node V_k selects its neighbor check (hash bin) in each $S_i, i \in [C]$ uniformly and independently of the other $S_{i'}, i' \neq i$. Thus, the resulting graph belongs to $\mathcal{G}(K, B, C)$. \square

In Section 2.5, we explained the peeling decoder for recovering the nonzero spectral components. It is not difficult to see that the performance of the algorithm always improves if we remove some of the variable nodes from the graph because it potentially reduces the number of colliding variables in the graph. This helps the peeling decoder to succeed decoding. With this explanation, we can prove the following proposition.

Proposition 2.5

Let $\alpha, C, K, h_i, i \in [C]$ be as explained before. Let \mathcal{G}_1 be the ensemble of bipartite graphs induced by the random support model $\text{RS1}(K, N)$ and hash functions h_i . For any $\epsilon > 0$ and for large dimension N , the average failure probability of the peeling decoder over \mathcal{G}_1 is upper bounded by its average failure probability over the ensemble $\mathcal{G}(K(1 + \epsilon), B, C)$.

Proof.

Let G_ϵ be a graph from ensemble $\mathcal{G}(K(1 + \epsilon), B, C)$. From Proposition 2.3, for large dimension N , the number of variable nodes in G_ϵ is greater than K with a very high probability. If we drop some of the variable nodes at random from G_ϵ , to keep only K of them, we obtain a graph G_1 from ensemble \mathcal{G}_1 . As the performance of the peeling decoder improves by removing some of the variable nodes, it performs strictly better over G_1 compared with G_ϵ . \square

Proposition 2.4 and Proposition 2.5 enable us to restrict the analysis of the performance of the peeling decoder to graphs from ensemble $\mathcal{G}(K, B, C)$.

Edge Degree Distribution Polynomial

In this section, we restrict ourselves to the graphs from the ensemble $\mathcal{G}(K, B, C)$ for the very sparse regime $\alpha \in (0, \frac{1}{3}]$. We assume that $n\alpha \in \mathbf{N}$ and select $b = n\alpha$. Hence, we have $K = B$. We call $\beta = \frac{K}{B}$ the average number of nonzero components per hash bin. We design the hash functions so that $\beta = 1$. All the graphs from the ensemble $\mathcal{G}(K, B, C)$ are left regular, i.e., all the variable nodes have degree C , whereas the degree of a check node depends on the graph realization.

Proposition 2.6

Let $\mathcal{G}(K, B, C)$ be the random graph ensemble as before with $\beta = \frac{K}{B}$ fixed. Then, as N tends to infinity, the check degree converges to a Poisson random variable with parameter β .

Proof.

The construction of the ensemble \mathcal{G} shows that any variable node has a probability of $\frac{1}{B}$ to be connected to a specific check node c , independent of all other variable nodes. Let $Z_i \in \{0, 1\}$ be a Bernoulli random variable where $Z_i = 1$ if and only if variable i is connected to check node c . It is easy to check that the degree of c will be $Z = \sum_{i=1}^K Z_i$. The Characteristic function of Z can be easily obtained:

$$\begin{aligned}\Phi_Z(\omega) &= \mathbb{E}e^{j\omega Z} = \prod_{i=1}^K \mathbb{E}e^{j\omega Z_i} \\ &= \left(1 + \frac{1}{B}(e^{j\omega} - 1)\right)^{\beta B} \rightarrow e^{\beta(e^{j\omega} - 1)},\end{aligned}$$

showing the convergence of Z to a Poisson distribution with parameter β . \square

For a bipartite graph, the edge degree distribution polynomial is defined by $\rho(\alpha) = \sum_{i=1}^{\infty} \rho_i \alpha^{i-1}$ and $\lambda(\alpha) = \sum_{i=1}^{\infty} \lambda_i \alpha^{i-1}$, where ρ_i (λ_i) is the ratio of all edges that are connected to a check node (variable node) of degree i . Notice that we have $i - 1$ instead of i in the formula. This choice enables us to write analysis in a compact form.

Proposition 2.7

Let G be a random bipartite graph from the ensemble $\mathcal{G}(K, B, C)$ with $\beta = \frac{K}{B}$. Then $\lambda(\alpha) = \alpha^{C-1}$ and $\rho(\alpha)$ converges to $e^{-\beta(1-\alpha)}$ as N tends to infinity.

Proof.

From left regularity of a graph from ensemble \mathcal{G} , it results that all the edges are connected to variable nodes of degree C , thus $\lambda(\alpha) = \alpha^{C-1}$. To find $\rho(\alpha)$, we need to find the fraction of edges that are connected to check nodes of a specific degree. From the symmetry of hash construction, it is sufficient to find the edge degree-distribution polynomial for check nodes of the first hash. The total number of edges that are connected to the check nodes of the first hash is equal to K . Let $i \geq 1$ and let N_i be the number of check nodes in the first hash with degree i . By definition of ρ_i , we obtain that

$$\rho_i = \frac{iN_i}{K} = \frac{iN_i/B}{K/B}.$$

Let Z be the random variable as in the proof of Proposition 2.6, denoting the degree of a specific check node. Then, as N tends to infinity, we can show that

$$\lim_{N \rightarrow \infty} \frac{N_i}{B} = \lim_{N \rightarrow \infty} \mathbb{P}\{Z = i\} = \frac{e^{-\beta} \beta^i}{i!} \text{ a.s.}$$

Thus, ρ_i converges almost surely to $\frac{e^{-\beta} \beta^{i-1}}{(i-1)!}$. As $\rho_i \leq 1$, for any α with $|\alpha| < 1 - \epsilon$, we have $|\rho_i \alpha^{i-1}| \leq (1 - \epsilon)^{i-1}$. Applying the *dominated convergence* theorem, we can prove that $\rho(\alpha)$ converges to $\sum_{i=1}^{\infty} \frac{e^{-\beta} \beta^{i-1}}{(i-1)!} \alpha^{i-1} = e^{-\beta(1-\alpha)}$. \square

As we will explain in Section 2.6.3, the performance of the peeling decoder highly depends on the parameter β ; the lower β , the better the performance of the peeling decoder. The drawback is that decreasing β , via increasing B , increases the time complexity $O(B \log_2(B))$ of computing the hash functions. Generally, we can select B such that $\beta \in [1, 2)$ or at the cost of increasing the computational complexity make β smaller, for example $\beta \in [\frac{1}{2}, 1)$, to obtain a better performance for the peeling decoding.

2.6.3 Performance Analysis of the Peeling Decoder

Consider a random bipartite graph resulting from applying C hashes to the signal spectrum. As we explained in Section 2.5, the iterative peeling algorithm starts by finding a singleton, i.e., a check node of degree 1 that is connected to only one variable node. The decoder peels off this variable node and removes all the edges connected to it from the graph. The algorithm continues by peeling off a singleton at each step until all the check nodes are zero nodes; all the nonzero variable nodes are decoded, or all the remaining unpeeled check nodes are multitons, in which case the algorithm fails to completely decode all the nonzero spectral variables.

Wormald's Method

In order to analyze the behavior of the resulting random graphs under the peeling decoding, the authors in [80] applied Wormald's method to track the ratio of edges in the graph connected to check nodes of degree 1 (singleton). The essence of Wormald's method is to approximate the behavior of a stochastic system (here the random bipartite graph), after applying suitable time normalization, by a deterministic differential equation. The idea is that as the size of the system becomes large (thermodynamic limit), the random state of the system is, uniformly for all times during the run of the algorithm, well concentrated around the solution of the differential equation. In [80], this method is applied to analyze the performance of the peeling decoder for bipartite graph codes over the BEC. We briefly explain the problem setting in [80] and how it can be used in our case.

Assume that we have a bipartite graph G with K variable nodes at the left, $C K$ check nodes at the right and with edge degree polynomials $\lambda(x)$ and $\rho(x)$. We can define a channel code $\mathcal{C}(G)$ over this graph as follows. We assign K independent message bits to K input variable nodes. The output of each check node is the module 2 summation (XOR or summation over \mathbb{F}_2) of all the message bits that are connected to it. Thus, the resulting code will be a systematic code with K message bits, along with $C K$ parity check bits. To communicate a K bit message over the channel, we send K message bits and all the check bits associated with them. While passing through the BEC, some of the message bits or check bits are erased independently. Consider a specific case in which the message bits and check bits are erased independently with probability δ and δ' , respectively. Those message bits that pass perfectly through the channel are successfully transmitted, thus the decoder tries to recover the erased message bits from the redundant information received via check bits. If we consider the induced graph after removing all variable nodes and check nodes corresponding to the erased ones from G , we end up with another bipartite graph G' . It is easy to see that over the new graph G' , we can apply the peeling decoder to recover the erased bits.

In [80], this problem was fully analyzed for the case of $\delta' = 0$, where all the check bits are received perfectly but δ ratio of the message bits are erased independently from one another. In other words, the final graph G' has on average $K\delta$ variable nodes to be decoded. Therefore, the

analysis can be simply applied to our case, by assuming that $\delta \rightarrow 1$, where all the variable nodes are erased (they are all unknown and need to be identified). Notice that from the assumption $\delta' = 0$, no check bit is erased as is the case in our problem. In particular, we use Proposition 2 in [80], which states the following.

Proposition 2 in [80]: Let G be a bipartite graph with edge degrees specified by $\lambda(x)$ and $\rho(x)$ and with K message bits chosen at random. Let δ be fixed so that

$$\rho(1 - \delta\lambda(x)) > 1 - x, \quad \text{for } x \in (0, 1].$$

For any $\eta > 0$, there is some K_0 such that for all $K > K_0$, if the message bits of $\mathcal{C}(G)$ are erased independently with probability δ , then with probability at least $1 - K^{\frac{2}{3}} \exp(-\sqrt[3]{K}/2)$ the recovery algorithm terminates with at most ηK message bits erased.

Replacing $\delta = 1$ in the proposition above, we obtain the following performance guarantee for the peeling decoder in our algorithm.

Proposition 2.8

Let $K = O(N^\alpha)$ for some $\alpha \in (0, \frac{1}{3}]$ and let G be a bipartite graph from the ensemble $\mathcal{G}(K, B, C)$ induced by hashing functions $h_i, i \in [C]$, with $\beta = \frac{K}{B}$, and with edge degree polynomials $\lambda(x) = x^{C-1}$ and $\rho(x) = e^{-\beta(1-x)}$. Suppose that

$$\rho(1 - \lambda(x)) > 1 - x, \quad \text{for } x \in (0, 1].$$

Given any $\epsilon \in (0, 1)$, there is a K_0 such that for any $K > K_0$ with probability at least $1 - K^{\frac{2}{3}} \exp(-\sqrt[3]{K}/2)$ the peeling decoder terminates with at most ϵK unrecovered nonzero spectral components.

Proposition 2.8 does not guarantee the success of the peeling decoder. It only implies that with a very high probability, it can peel off any fraction $\eta \in (0, 1)$ of nonzero components, but not necessarily all of them. Using a combinatorial argument, however, it is possible to prove that with very high probability any graph in the ensemble \mathcal{G} is an expander graph, specifically, every small enough subset of left nodes has many check neighbors. This implies that if the peeling decoder can decode a specific ratio of variable nodes, it can proceed to decode all of them. A slight modification of Lemma 1 in [80] gives the following result proved in Appendix 2.D.

Proposition 2.9

Let $K = O(N^\alpha)$ for some $\alpha \in (0, \frac{1}{3}]$ and let G be a graph from the ensemble $\mathcal{G}(K, B, C)$ with $C \geq 3$. There is some $\eta > 0$ such that with probability at least $1 - O(\frac{1}{N^{3\alpha(C/2-1)}})$, the recovery process restricted to the subgraph induced by any η -fraction of the left nodes terminates successfully.

Proof of Part 3 of Theorem 2.1 for $\alpha \in (0, \frac{1}{3}]$: In the very sparse regime $\alpha \in (0, \frac{1}{3}]$, we construct $C = \lfloor \frac{1}{\alpha} \rfloor \geq 3$ hashes each containing $2^{n\alpha}$ output bins. Combining Proposition 2.8 and Proposition 2.9, we obtain that the success probability of the peeling decoder is lower bounded by $1 - O(\frac{1}{N^{3\alpha(C/2-1)}})$ as mentioned in Remark 2.1.

Analysis Based on Belief Propagation over Sparse Graphs

In this section, we give another method of analysis and provide further intuition about the performance of the peeling decoder and why it works very well in the very sparse regime. This

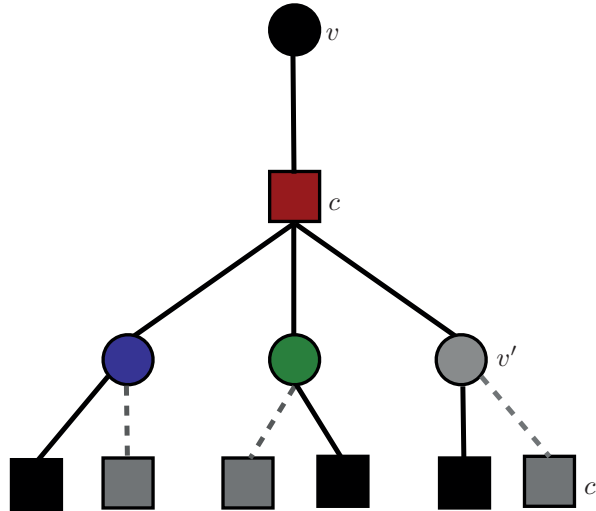


Figure 2.4: Tree-like neighborhood of an edge $e = (v, c)$. Dashed lines show the edges that have been removed before iteration t . The edge e is peeled off at iteration t because all the variable nodes v' connected to c are already decoded, thus c is a singleton check.

method is based on the analysis of belief-propagation (BP) decoder over sparse locally tree-like graphs. The analysis is very similar to the analysis of the peeling decoder for recovering nonzero frequency components in [100]. We first need some terminology from graph theory. A walk of size ℓ in graph G starting from a node $v \in [K]$ is a set of ℓ edges e_1, e_2, \dots, e_ℓ , where v is one of the vertices of the edge e_1 and where consecutive edges are different, $e_i \neq e_{i+1}$, but incident with each other. A directed neighborhood of an edge $e = (v, c)$ of depth ℓ is the induced subgraph in G consisting of all edges and associated check and variable nodes in all walks of size $\ell + 1$ starting from v with the first edge being $e_1 = (v, c)$. A node e is said to have a tree-like neighborhood of depth ℓ if the directed neighborhood of e of depth ℓ is a tree.

Let $e = (v, c)$ be an edge in a graph from ensemble $\mathcal{G}(K, B, C)$ and consider a directed neighborhood of this edge of depth ℓ . At the first stage, it is easy to see that this edge is peeled off from the graph assuming that all the edges (c, v') connected to the check node c are peeled off, because in that case check node c will be a singleton enabling us to decode the variable v . This is shown in Figure 2.4.

One can proceed in this way in the directed neighborhood to find the condition under which the variable v' connected to c can be peeled off, and so on. Assuming that the directed neighborhood is a tree, all the messages that are passed from the leaves up to the head edge e are independent from one another. Let p_ℓ be the probability that edge e is peeled off depending on the information received from the directed neighborhood of depth ℓ assuming a tree up to depth ℓ . A simple analysis similar to [100], gives the following recursion

$$p_{j+1} = \lambda(1 - \rho(1 - p_j)), \quad j \in [\ell], \quad (2.10)$$

where λ and ρ are the edge degree polynomials of the ensemble \mathcal{G} . This iteration shows the progress of the peeling decoder in recovering unknown variable nodes. In [100], it is proved that

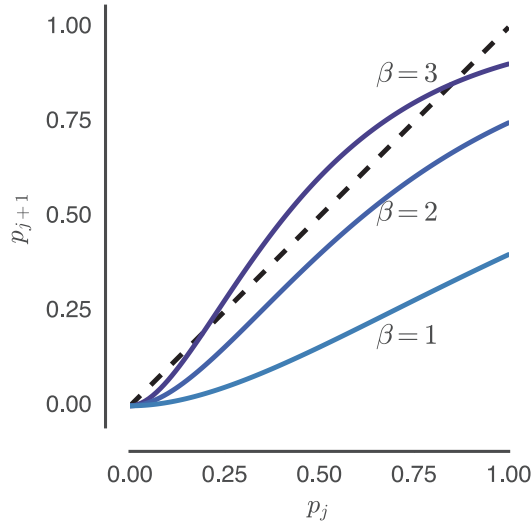


Figure 2.5: Density Evolution equation for $C = 3$ and different values of $\beta = \frac{K}{B}$.

for any specific edge e , asymptotically with a very high probability, the directed neighborhood of e up to any fixed depth ℓ is a tree. Specifically, if we start from a left regular graph \mathcal{G} from $\mathcal{G}(K, B, C)$ with KC edges, after ℓ steps of decoding, the average number of unpeeled edges is concentrated around KCp_ℓ . Moreover, a martingale argument is applied in [100] to show that not only the average number of unpeeled edges is approximately KCp_ℓ , but also with a very high probability the number of those edges is well concentrated around KCp_ℓ .

Eq. (2.10) is generally known as the density evolution equation. Starting from $p_0 = 1$, this equation fully predicts the behavior of the peeling decoding over the ensemble \mathcal{G} . Fig. 2.5 shows a typical behavior of this iterative equation for different values of the parameter $\beta = \frac{K}{B}$.

For very small values of β , this equation has only a fixed point at 0, which implies that for large dimension N , the peeling decoder can recover a fraction of variables very close to 1. However, for large values of β , i.e., $\beta \gtrsim 2.44$ for $C = 3$, this equation has a fixed point greater than 0. The largest fixed point is the place where the peeling decoder stops and cannot proceed to decode the remaining variables. It is easy to see that the only fixed point is 0 provided that for any $p \in (0, 1]$, $p > \lambda(1 - \rho(1 - p))$. As $\lambda : [0, 1] \rightarrow [0, 1]$, $\lambda(x) = x^{C-1}$ is an increasing function of x , by change of variable $x = \lambda^{-1}(p)$, we obtain that $x > 1 - \rho(1 - \lambda(x))$ or equivalently

$$\rho(1 - \lambda(x)) > 1 - x, \quad \text{for } x \in (0, 1].$$

This is exactly the same result that we obtained by applying Wormald's method, as in [80]. In particular, this analysis clarifies the role of x in Wormald's method.

Similar to Wormald's method, this analysis only guarantees that for any $\epsilon \in (0, 1)$, asymptotically as N tends to infinity, $1 - \epsilon$ fraction of the variable nodes can be recovered. An expander argument is again necessary to guarantee the full recovery of all the remaining variables.

2.7 Performance Analysis of the Less Sparse Regime

For the less sparse regime ($\alpha \in (\frac{1}{3}, 1]$), similar to the very sparse case, we will first construct suitable hash functions, which guarantee a low computational complexity of order $O(K \log_2(K) \log_2(\frac{N}{K}))$ for the recovery of nonzero spectral values. Assuming a uniformly random support model in the spectral domain, similar to the very sparse case, we can represent the hashes by a regular bipartite graph. Over this graph, the peeling algorithm proceeds to find singleton checks and peel the associated variables from the graph until no singleton remains. The recovery is successful if all the variables are peeled off, thus all the remaining checks are zerotons otherwise some of the nonzero spectral values are not recovered and the perfect recovery fails.

As we explain in Section 2.7.2, the structure of the induced bipartite graph in this regime is a bit different than the very sparse one. The following steps are used to analyze the performance of the peeling decoder:

1. Constructing suitable hash functions.
2. Representing hashing of nonzero spectral values by an equivalent bipartite graph.
3. Analyzing the performance of the peeling decoder over the resulting bipartite graph.

For simplicity, we consider the case where $\alpha = 1 - \frac{1}{C}$ for some integer $C \geq 3$. We will explain how to deal with arbitrary values of C and α , especially those in the range $(\frac{1}{3}, \frac{2}{3})$, in Section 2.7.4.

2.7.1 Hash Construction

Assume that $\alpha = 1 - \frac{1}{C}$ for some integer $C \geq 3$. Let x be an N dimensional signal with $N = 2^n$ and let X denote its WHT. For simplicity, we label the components of X by a binary vector $X_0^{n-1} \in \mathbb{F}_2^n$. Let $t = \frac{n}{C}$ and let us divide the set of n binary indices X_0^{n-1} into C non-intersecting subsets r_0, r_1, \dots, r_{C-1} , where $r_i = X_{i t}^{(i+1)t-1}$. It is clear that there is a one-to-one relation between each binary vector $X_0^{n-1} \in \mathbb{F}_2^n$ and its representation $(r_0, r_1, \dots, r_{C-1})$. We construct C different hash function $h_i, i \in [C]$ by selecting different subsets of $(r_0, r_1, \dots, r_{C-1})$ of size $C - 1$ and appending them together. For example

$$h_1(X_0^{n-1}) = (r_0, r_1, \dots, r_{C-2}) = X_0^{(C-1)t-1},$$

and the hash output is obtained by appending $C - 1$ first $r_i, i \in [C]$. We can simply check that $h_i, i \in [C]$ are linear surjective functions from \mathbb{F}_2^n to \mathbb{F}_2^b , where $b = (C - 1)t$. In particular, the range of each hash consists of $B = 2^b$ different elements of \mathbb{F}_2^b . Moreover, if we denote the null space of h_i by $\mathcal{N}(h_i)$, it is easy to show that for any $i, j \in [C], i \neq j$, $\mathcal{N}(h_i) \cap \mathcal{N}(h_j) = \mathbf{0} \in \mathbb{F}_2^n$.

Using the subsampling property of the WHT and similar to the hash construction that we had in Section 2.6.1, it is seen that subsampling the time-domain signal and taking WHT of the subsampled signal is equivalent to hashing the spectral components of the signal. In particular, all the spectral components X_0^{n-1} with the same $h_i(X_0^{n-1})$ are mapped into the same bin in hash i , thus different bins of the hash can be labelled with B different elements of \mathbb{F}_2^b .

It is easy to see that, with this construction, the average number of nonzero elements per bin in every hash is kept at $\beta = \frac{K}{B} = 1$ and the complexity of computing all the hashes along with their $n - b$ shifts, which are necessary for collision detection/support estimation, is $CK \log_2(K) \log_2(\frac{N}{K})$. The sample complexity can also be easily checked to be $CK \log_2(\frac{N}{K})$.

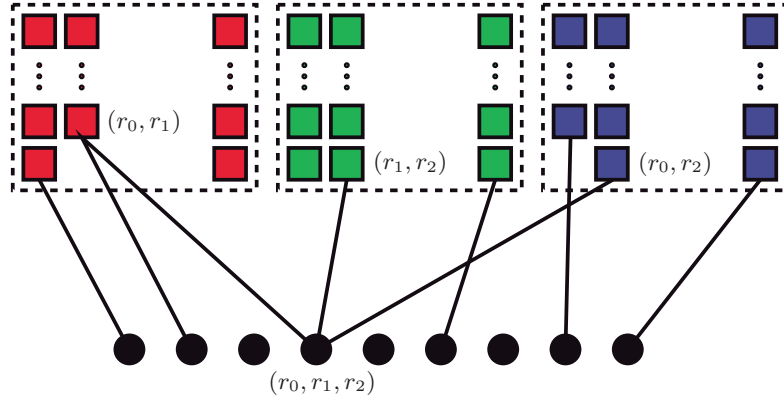


Figure 2.6: Bipartite graph representation for the less sparse case $\alpha = \frac{2}{3}$, $C = 3$

2.7.2 Bipartite Graph Representation

Similarly to the very sparse regime, we can assign a bipartite graph with the K left nodes (variable nodes) corresponding to nonzero spectral components and with CB right nodes corresponding to different bins of all the hashes. In particular, we consider C different set of check nodes S_1, S_2, \dots, S_C each containing B nodes labelled with the elements of \mathbb{F}_2^b , and a specific nonzero spectral component labelled with X_0^{n-1} is connected to nodes $s_i \in S_i$ if and only if the binary label assigned to s_i is $h_i(X_0^{n-1})$. In the very sparse regime, we showed that if the support of the signal is generated according to the RS2(K, N), where K random positions are selected uniformly at random independently from one another from $[N]$, then the resulting graph is a random left regular bipartite graph, where each variable node selects completely independently its C neighbors in S_1, S_2, \dots, S_C . However, in the less sparse regime, the selection of the neighbor checks in different hashes is not completely random. To explain more, suppose $\alpha = \frac{2}{3}$, thus $C = 3$. Also assume that for a nonzero spectral variable labelled with X_0^{n-1} , r_i denotes $X_{it}^{(i+1)t-1}$, where $t = \frac{n}{C}$. In this case, this variable is connected to bins labelled with (r_0, r_1) , (r_1, r_2) and (r_0, r_2) in 3 different hashes. This has been depicted in Figure 2.6.

If we assume that X_0^{n-1} is selected uniformly at random from \mathbb{F}_2^n , then the bin numbers in each hash, i.e. (r_0, r_1) in the first hash, are individually selected uniformly at random among all possible bins. However, it is easily seen that the joint selection of bins is not completely random among different hashes. In other words, the associated bins in different hashes are not independent from one another. However, assuming the random support model, where K variable V_1^K are selected independently as the position of nonzero spectral variables, the bin association for different variables V_i is still made independently.

2.7.3 Performance Analysis of the Peeling Decoder

As the resulting bipartite graph is not a completely random graph, it is not possible to directly apply Wormald's method as we did for the very sparse case as in [80]. However, an analysis based on the DE for the BP algorithm can still be applied. In other words, setting $p_0 = 1$ and

$$p_{j+1} = \lambda(1 - \rho(1 - p_j)), \quad j \in [\ell],$$

as in (2.10) with λ and ρ being the edge degree polynomials of the underlying bipartite graph, it is still possible to show that after ℓ steps of decoding, the average number of unpeeled edges is approximately KCp_ℓ . A martingale argument similar to [100] can be applied to show that the number of remaining edges is also well concentrated around its average. Similar to the very sparse case, this argument asymptotically guarantees the recovery of any ratio of the variables between 0 and 1. Another argument is necessary to show that if the peeling decoder decodes a majority of the variables, it can proceed to decode all of them with very high probability. To formulate this, we use the concept of trapping sets for the peeling decoder.

Definition 2.2

Let $\alpha = 1 - \frac{1}{C}$ for some integer $C \geq 3$ and let $h_i, i \in [C]$ be a set of hash functions as explained before. A subset of variables $T \subset \mathbb{F}_2^n$ is called a trapping set for the peeling decoder if for any $v \in T$ and for any $i \in [C]$, there is another $v_i \in T, v \neq v_i$ such that $h_i(v) = h_i(v_i)$, thus colliding with v in the i -th hash.

Notice that a trapping set cannot be decoded because all its neighbor check nodes are multitons. We first analyze the structure of the trapping set and find the probability that a specific set of variables builds a trapping set. Let X be a spectral variable in the trapping set with the corresponding binary representation X_0^{n-1} and assume that $C = 3$. We can equivalently represent this variable with (r_0, r_1, r_2) , where $r_i = X_{it}^{(i+1)t-1}$ with $t = \frac{n}{C}$. We can consider a three dimensional lattice whose i -th axis is labelled by all possible values of r_i . In this space, there is a simple interpretation for a set T to be a trapping set, namely, for any $(r_0, r_1, r_2) \in T$ there are three other elements $(r'_0, r_1, r_2), (r_0, r'_1, r_2)$ and (r_0, r_1, r'_2) in T that can be reached from (r_0, r_1, r_2) by moving along exactly one axis. Notice that in this case each hash is equivalent to projecting (r_0, r_1, r_2) onto two dimensional planes spanned by different coordinates, for example, $h_1(r_0, r_1, r_2) = (r_0, r_1)$ is a projection on the plane spanned by the first and second coordinate axes of the lattice. A similar argument holds for other values of $C > 3$, thus larger values of α .

For $C \geq 3$, the set of all C -tuples $(r_0, r_1, \dots, r_{C-1})$ is a C -dimensional lattice. We denote this lattice by L . The intersection of this lattice by the hyperplane $R_i = r_i$ is a $(C - 1)$ dimensional lattice defined by

$$L(R_i = r_i) = \{(r_0, \dots, r_{i-1}, r_{i+1}, \dots, r_{C-1}) : (r_0, r_1, \dots, r_{i-1}, r_i, r_{i+1}, \dots, r_{C-1}) \in L\}.$$

Similarly for $S \subset L$, we have the following definition

$$S(R_i = r_i) = \{(r_0, \dots, r_{i-1}, r_{i+1}, \dots, r_{C-1}) : (r_0, r_1, \dots, r_{i-1}, r_i, r_{i+1}, \dots, r_{C-1}) \in S\}.$$

Obviously, $S(R_i = r_i) \subset L(R_i = r_i)$. We have the following proposition whose proof simply follows from the definition of the trapping set.

Proposition 2.10

Assume that T is a trapping set for the C dimensional lattice representation L of the nonzero spectral-domain variables as explained before. Then for any r_i on the i -th axis, $T(R_i = r_i)$ is either empty or a trapping set for the $(C - 1)$ dimensional lattice $L(R_i = r_i)$.

Proposition 2.11

The size of the trapping set for a C dimensional lattice is at least 2^C .

Proof.

We use a simple proof by induction on C . For $C = 1$, we have a one-dimensional lattice along a line labelled with r_0 . In this case, there must be at least two variables on the line to build a trapping set. Consider a trapping set T of dimension C . There are at least two points $(r_0, r_1, \dots, r_{C-1})$ and $(r'_0, r_1, \dots, r_{C-1})$ in T . By Proposition 2.10, $T(R_0 = r_0)$ and $T(R_0 = r'_0)$ are two $(C - 1)$ dimensional trapping sets each consisting of at least 2^{C-1} elements by induction hypothesis. Thus, T has at least 2^C elements. \square

Remark 2.3

The bound $|T| \geq 2^C$ on the size of the trapping set is actually tight. For example, for $i \in [C]$ consider r_i, r'_i where $r_i \neq r'_i$ and let

$$T = \{(a_0, a_1, \dots, a_{C-1}) : a_i \in \{r_i, r'_i\}, i \in [C]\}.$$

It is easy to see that T is a trapping set with 2^C elements corresponding to the vertices of a C dimensional cube.

We now prove the following proposition that implies that if the peeling decoder can decode all the variable nodes except a fixed number of them, with a high probability it can continue to decode all of them.

Proposition 2.12

Let s be a fixed positive integer. Assume that $\alpha = 1 - \frac{1}{C}$ for some integer $C \geq 3$ and consider a hash structure with C different hashes. If the peeling decoder decodes all except a set of variables of size s , it can decode all the variables with very high probability.

Proof.

The proof is very similar to [100]. Let T be a trapping set of size s . By Proposition 2.11, we have $s \geq 2^C$. Let p_i be the number of distinct values taken by elements of T along the R_i axis and let $p_{\max} = \max_{i \in [C]} p_i$. Without loss of generality, let us assume that the R_0 axis is the one having the maximum p_i . Consider $T(R_0 = r_0)$ for those p_{\max} values of r_0 along the R_0 axis. Proposition 2.10 implies that each $T(R_0 = r_0)$ is a trapping set that has at least 2^{C-1} elements according to Proposition 2.11. This implies that $s \geq 2^{C-1} p_{\max}$ or $p_{\max} \leq \frac{s}{2^{C-1}}$. Moreover, T being the trapping set implies that there are subsets T_i consisting of elements from axes R_i and all the elements of T are restricted to take their i -th coordinate values along R_i from the set T_i . Considering the way that we generate the position of nonzero variables X_0^{n-1} with the equivalent representation $(r_0, r_1, \dots, r_{C-1})$, the coordinates of any variable are selected uniformly and completely independently from one another and from the coordinates of the other variables. This implies that

$$\begin{aligned} \mathbb{P}\{F_s\} &\leq \mathbb{P}\{\text{For any variables in } T, r_i \in T_i, i \in [C]\} \\ &\leq \prod_{i=0}^{C-1} \binom{\mathcal{P}_i}{p_i} \left(\frac{p_i}{\mathcal{P}_i}\right)^s \leq \prod_{i=0}^{C-1} \binom{\mathcal{P}_i}{s/2^{C-1}} \left(\frac{s}{2^{C-1}\mathcal{P}_i}\right)^s, \end{aligned}$$

where F_s is the event that the peeling decoder fails to decode a specific subset of variables of size s and where \mathcal{P}_i denotes the number of all possible values for the i -th coordinate of a variable.

By our construction all \mathcal{P}_i are equal to $P = 2^{n/C} = 2^{n(1-\alpha)} = N^{(1-\alpha)}$, thus we obtain that

$$\begin{aligned} \mathbb{P}\{F_s\} &\leq \binom{P}{s/2^{C-1}}^C \left(\frac{s}{2^{C-1}P}\right)^{sC} \\ &\leq \left(\frac{2^{C-1}Pe}{s}\right)^{sC/2^{C-1}} \left(\frac{s}{2^{C-1}P}\right)^{sC} \\ &\leq \left(\frac{se^{1/(2^{C-1}-1)}}{2^{C-1}P}\right)^{sC(1-1/2^{C-1})}. \end{aligned}$$

Taking the union bound over all $\binom{K}{s}$ possible ways of selection of s variables out of K variables, we obtain that

$$\begin{aligned} \mathbb{P}\{F\} &\leq \binom{K}{s} \mathbb{P}\{F_s\} \\ &\leq \left(\frac{eP^{C-1}}{s}\right)^s \left(\frac{se^{1/(2^{C-1}-1)}}{2^{C-1}P}\right)^{sC(1-1/2^{C-1})} \\ &= O\left(1/P^{s(1-\frac{C}{2^{C-1}})}\right) \\ &\leq O\left(1/P^{(2^C-2C)}\right) = O\left(1/N^{\frac{2^C}{C}-2}\right). \end{aligned}$$

For $C \geq 3$, this gives an upper bound of $O(N^{-\frac{2}{3}})$. \square

2.7.4 Generalized Hash Construction

The hash construction for the less sparse regime that we explained in Section 2.7.1 only covers values of $\alpha = 1 - \frac{1}{C}$ for $C \geq 3$, which belongs to the region $\alpha \in (\frac{2}{3}, 1)$. In this section, we explain a hash construction that fills the gap for $\alpha \in (\frac{1}{3}, \frac{2}{3})$, and extends to any value of $\alpha \in (0, 1)$ that is not necessarily of the form $\frac{1}{C}$ (very sparse) or $1 - \frac{1}{C}$ (less sparse).

In the very sparse regime $\alpha = \frac{1}{3}$, we have $C = 3$ different hashes and for a nonzero spectral variable X with the binary index $X_0^{n-1} = (r_0, r_1, r_2)$, the i -th hash output is $h_i(X_0^{n-1}) = r_i$, $i \in \{0, 1, 2\}$, thus the output of different hashes depend on non-overlapping parts of the binary index of X ; whereas for $\alpha = \frac{2}{3}$ the hash outputs are (r_0, r_1) , (r_1, r_2) and (r_0, r_2) , which overlap on a portion of binary indices of length $\frac{n}{3}$. Intuitively, it is clear that in order to construct different hashes for $\alpha \in (\frac{1}{3}, \frac{2}{3})$, we should start increasing the overlapping size of different hashes from 0 for $\alpha = \frac{1}{3}$ to $\frac{n}{3}$ for $\alpha = \frac{2}{3}$. Generally, let C be the desired number of hashes. We give the following construction for the hash functions

$$h_i(X_0^{n-1}) = X_{it}^{it+b-1}, i \in [C],$$

where $b = n\alpha$ and $t = \frac{n}{C}$, and where the values of the indices are computed modulo n , for example $X_n = X_0$. Furthermore, the required number of hashes is given by $C = (\frac{1}{\alpha} \vee \frac{1}{1-\alpha})$.

It is clear that each hash is a surjective map from \mathbb{F}_2^n into \mathbb{F}_2^b . Moreover, for this choice of b ($b = n\alpha$), the number of output bins in each hash is $B = 2^{n\alpha} = N^\alpha = K$, thus the average number of nonzero variables per bin in every hash is equal to $\beta = \frac{K}{B} = 1$. Also, for the intermediate values of $\alpha \in (\frac{1}{3}, \frac{2}{3})$, we expect the performance of the peeling decoder for this regime to be between the very sparse regime $\alpha = \frac{1}{3}$ and the less sparse one $\alpha = \frac{2}{3}$.

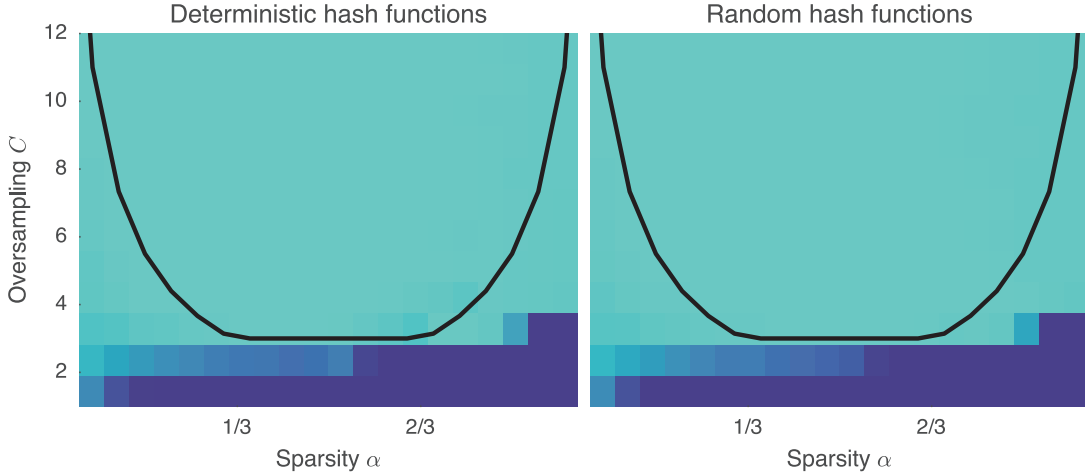


Figure 2.7: Probability of success of the algorithm as a function of α and C for deterministic (left, as described in Section 2.7.4) and random (right) hash constructions. Light shades indicate the algorithm always succeeds, while dark shades indicate it always fails. The dimension of the signal is $N = 2^{22}$. The black line corresponds to $\alpha = \frac{1}{C}$ and $\alpha = 1 - \frac{1}{C}$ in the very sparse and less sparse regimes, respectively. We fix $\beta = 1$.

2.8 Experimental Results

In this section, we empirically evaluate the performance of the SparseFHT algorithm for a variety of design parameters. The simulations are implemented in the C programming language and the success probability of the algorithm is estimated via a sufficient number of trials. We also provide a comparison of the run time of our algorithm and the standard Hadamard transform. In all experiments, the input signal has support uniformly drawn at random without replacement. The nonzero components are drawn from a zero-mean normal distribution with variance $\sigma^2 = 100$. In the spirit of reproducible research, all the material (C, python, and Matlab code) needed to reproduce the results of this chapter is available online at <http://lcav.github.io/SparseFHT/>.

- *Experiment 1:* We fix the signal size to $N = 2^{22}$ and run the algorithm 1000 times to estimate the success probability for $\alpha \in (0, 1)$ and $1 \leq C \leq 12$. The hashing scheme used is as described in Section 2.7.4. Figure 2.7 shows the simulation result. Albeit the asymptotic behavior of the error probability is only guaranteed for $C = (\frac{1}{\alpha} \vee \frac{1}{1-\alpha})$, we observe much better results in practice. Indeed, $C = 4$ already gives a probability of success very close to one over a large range of α , and only up to $C = 6$ seems to be required for the largest values of α .
- *Experiment 2:* We repeat here experiment 1, but instead of deterministic hashing matrices, we now pick Σ_i , $i \in [C]$, uniformly at random from $GL(n, \mathbb{F}_2)$. The result is shown in Figure 2.7. We observe that this scheme performs at least as well as the deterministic one.
- *Experiment 3:* In this experiment, we investigate the sensitivity of the algorithm to the value of the parameter $\beta = K/B$; the average number of nonzero coefficients per bin. In our design for hash function, we always use $\beta \approx 1$ throughout the chapter. However,

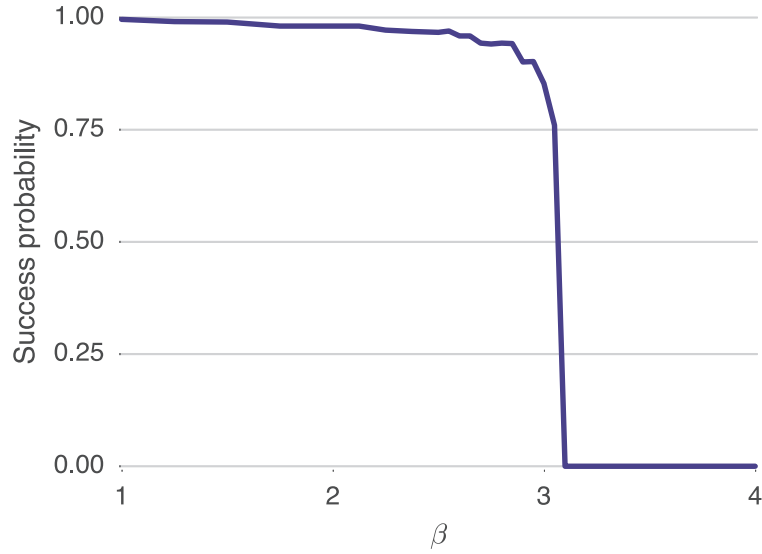


Figure 2.8: Probability of success of the algorithm in the less sparse regime as a function of $\beta = K/B$. We fix $N = 2^{22}$, $B = 2^{17}$, $C = 4$, and vary α in the range 0.7 to 0.9.

using larger values of β is appealing from a computational complexity point of view. For the simulation, we fix $N = 2^{22}$, $B = 2^{17}$, $C = 4$, and vary α between 0.7 and 0.9, thus changing K and as a result β . Figure 2.8 shows the simulation results. For $\beta \leq 2$, the algorithm succeeds with probability very close to one. Moreover, for values of β larger than 3, the success probability sharply goes to 0, as predicted by the theory.

- *Runtime measurement:* We compare the runtime of the SparseFHT algorithm with a straightforward implementation of the fast Hadamard transform. The result is shown in Figure 2.9a for $N = 2^{15}$. SparseFHT performs much faster for $0 < \alpha < 2/3$.

It is also interesting to identify the range of α for which SparseFHT has a lower runtime than the conventional FHT. We define α^* , the largest value of α such that SparseFHT is faster than FHT for any lower value of α . That is

$$\alpha^* = \sup_{\alpha \in (0,1)} \{ \alpha : \forall \alpha' \leq \alpha, T_{FHT}(n) > T_{SFHT}(\alpha', n) \},$$

where T_{FHT} and T_{SFHT} are the runtimes of the conventional FHT and SparseFHT, respectively. We plot α^* as a function of $n = \log_2 N$ in Figure 2.9b.

2.9 Conclusion

We presented a new algorithm for computing the Hadamard transform of a signal of length N that is K -sparse in the Hadamard domain with $K = O(N^\alpha)$ scaling sub-linearly with N for some $\alpha \in (0, 1)$. Our algorithm computes the K -sparse Hadamard transform of the signal with a computational complexity $O(K \log_2 K \log_2 \frac{N}{K})$, and only requires $O(K \log_2 \frac{N}{K})$ time-domain samples

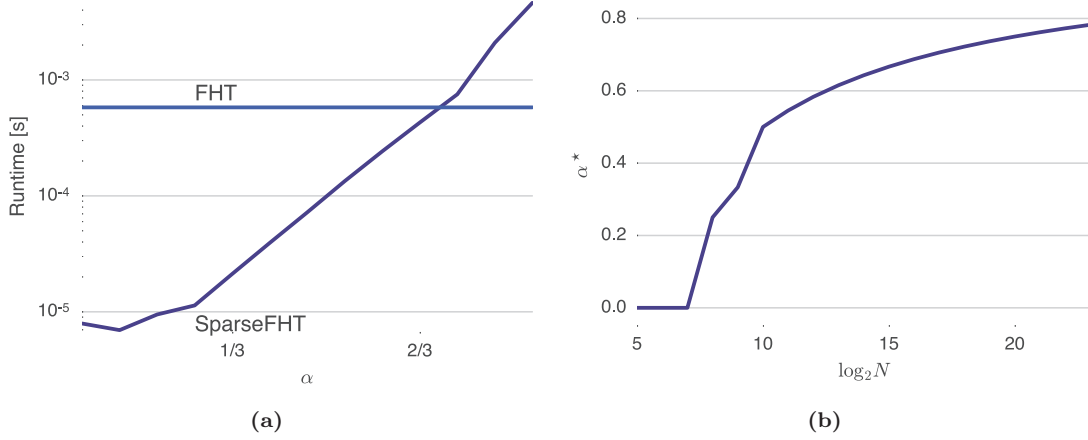


Figure 2.9: (a) Comparison of the median runtime in ms of the SparseFHT and conventional WHT for $N = 2^{15}$ and for different values of α . (b) We change the value of $n = \log_2 N$ and plot α^* , the largest value of α such that SparseFHT runs faster than the conventional WHT. When WHT is always faster, we simply set $\alpha^* = 0$.

of the signal. We have shown that the algorithm correctly reconstructs the Hadamard transform of the signal with a very high probability approaching 1 for a sufficiently large dimension N .

We evaluated empirically the performance of our algorithm through numerical simulations, and compare its speed with that of the conventional fast Hadamard transform. We observe that our algorithm is much faster, even for moderate signal lengths (e.g. $N = 2^{10}$) and reasonable sparsity.

In our algorithm, we considered a noiseless case, where there is no measurement noise in the time-domain samples. This assumption was necessary in Proposition 2.2 in order to make a collision detection/support estimation. Hence, a more robust variant of Proposition 2.2 is necessary for the noisy case. Several publications have since tackled the issue [28, 75, 76].

2.A Proof of the Properties of the WHT

2.A.1 Proof of Property 2.1

$$\sum_{m \in \mathbb{F}_2^n} (-1)^{\langle k, m \rangle} x_{m+p} = \sum_{m \in \mathbb{F}_2^n} (-1)^{\langle k, m+p \rangle} x_m.$$

And the proof follows by taking $(-1)^{\langle k, p \rangle}$ out of the sum and recognizing the Hadamard transform of x_m . ■

2.A.2 Proof of Property 2.2

As we explained, it is possible to assign an $N \times N$ matrix Π to the permutation π as follows

$$(\Pi)_{i,j} = \begin{cases} 1 & \text{if } j = \pi(i) \Leftrightarrow i = \pi^{-1}(j) \\ 0 & \text{otherwise.} \end{cases}$$

Let π_1 and π_2 be the permutations associated with Π_1 and Π_2 . As $(H_N)_{i,j} = (-1)^{\langle i,j \rangle}$, the identity (2.2) implies that

$$(-1)^{\langle \pi_2(i), j \rangle} = (-1)^{\langle i, \pi_1^{-1}(j) \rangle}.$$

Therefore, for any $i, j \in \mathbb{F}_2^n$, π_1, π_2 must satisfy $\langle \pi_2(i), j \rangle = \langle i, \pi_1^{-1}(j) \rangle$. By linearity of the inner product, we obtain

$$\begin{aligned} \langle \pi_2(i+k), j \rangle &= \langle i+k, \pi_1^{-1}(j) \rangle \\ &= \langle i, \pi_1^{-1}(j) \rangle + \langle k, \pi_1^{-1}(j) \rangle \\ &= \langle \pi_2(i), j \rangle + \langle \pi_2(k), j \rangle. \end{aligned}$$

As $i, j \in \mathbb{F}_2^n$ are arbitrary, this implies that π_2 , and by symmetry π_1 , are both linear operators. Hence, all the permutations satisfying (2.2) are in one-to-one correspondence with the elements of $\text{GL}(n, \mathbb{F}_2)$. \blacksquare

2.A.3 Proof of Property 2.3

As Σ is non-singular, Σ^{-1} exists, and from the definition of the WHT, it follows that

$$\begin{aligned} \sum_{m \in \mathbb{F}_2^n} (-1)^{\langle k, m \rangle} x_{\Sigma m} &= \sum_{m \in \mathbb{F}_2^n} (-1)^{\langle k, \Sigma^{-1} m \rangle} x_m \\ &= \sum_{m \in \mathbb{F}_2^n} (-1)^{\langle \Sigma^{-\top} k, m \rangle} x_m. \end{aligned}$$

This completes the proof. \blacksquare

2.A.4 Proof of Property 2.4

Note that $m \in \mathbb{F}_2^b$, and $x_{\Psi_b m}$ is a signal of dimension $B = 2^b$. Let \tilde{X}_k denote its WHT, where $k \in \mathbb{F}_2^b$. From the definition of WHT, we have

$$\begin{aligned} \tilde{X}_k &= \frac{1}{\sqrt{B}} \sum_{m \in \mathbb{F}_2^b} (-1)^{\langle k, m \rangle} x_{\Psi_b m} \\ &\stackrel{(a)}{=} \frac{1}{\sqrt{BN}} \sum_{m \in \mathbb{F}_2^b} (-1)^{\langle k, m \rangle} \sum_{u \in \mathbb{F}_2^n} (-1)^{\langle \Psi_b m, u \rangle} X_u \\ &\stackrel{(b)}{=} \frac{1}{\sqrt{BN}} \sum_{u \in \mathbb{F}_2^n} X_u \sum_{m \in \mathbb{F}_2^b} (-1)^{\langle m, k + \Psi_b^\top u \rangle} \\ &\stackrel{(c)}{=} \frac{B}{\sqrt{BN}} \sum_{u \in \mathbb{F}_2^n} X_u \mathbf{1}_{\{k + \Psi_b^\top u = 0\}}, \end{aligned}$$

where in (a), we used the inverse of the WHT for the N dimensional signal x ($N = 2^n$) and its transform-domain signal X , in (b), we used $\langle \Psi_b m, u \rangle = \langle m, \Psi_b^\top u \rangle$, and in (c), we used the following identity for $s \in \mathbb{F}_2^b$,

$$\sum_{m \in \mathbb{F}_2^b} (-1)^{\langle m, s \rangle} = B \mathbf{1}_{\{s=0\}}.$$

We can check that $k + \Psi_b^\top u = 0$ holds if and only if $u = \Psi_b k + j$ with $j \in \mathcal{N}(\Psi_b^\top)$. Hence, we obtain the desired result $\tilde{X}_k = \sqrt{\frac{B}{N}} \sum_{j \in \mathcal{N}(\Psi_b^\top)} X_{\Psi_b k + j}$. \blacksquare

2.A.5 Proof of Property 2.5

$$\begin{aligned} \sum_{m \in \mathbb{F}_2^n} (x \star y)_m (-1)^{\langle m, k \rangle} &= \sum_{m \in \mathbb{F}_2^n} \sum_{u \in \mathbb{F}_2^n} x_u y_{u+m} (-1)^{\langle m, k \rangle} \stackrel{(a)}{=} \sum_{v \in \mathbb{F}_2^n} \sum_{u \in \mathbb{F}_2^n} x_u y_v (-1)^{\langle v+u, k \rangle} \\ &= \sum_{u \in \mathbb{F}_2^n} x_u (-1)^{\langle u, k \rangle} \sum_{v \in \mathbb{F}_2^n} y_v (-1)^{\langle v, k \rangle} = X_k Y_k, \end{aligned}$$

where in (a) we make the substitution $m = u + v$. ■

2.B Proof of Proposition 2.2

We first show that if multiple coefficients fall in the same bin, it is very unlikely that 1) is fulfilled. Let $\mathcal{I}_k = \{j \mid \mathcal{H}j = k\}$ be the set of variable indices that are hashed to bin k . This set is finite and its elements can be enumerated as $\mathcal{I}_k = \{j_1, \dots, j_{\frac{N}{B}}\}$. In particular, \mathcal{I}_k is an $n - b$ dimensional affine subspace of \mathbb{F}_2^n . We show that a set $\{X_j\}_{j \in \mathcal{I}_k}$ is very unlikely, unless it contains only one nonzero element. Without loss of generality, we consider $\sum_{j \in \mathcal{I}_k} X_j = 1$. Such $\{X_j\}_{j \in \mathcal{I}_k}$ is a solution of

$$\begin{bmatrix} 1 & \cdots & 1 \\ (-1)^{\langle \sigma_1, j_1 \rangle} & \cdots & (-1)^{\langle \sigma_1, j_{\frac{N}{B}} \rangle} \\ \vdots & \ddots & \vdots \\ (-1)^{\langle \sigma_{n-b}, j_1 \rangle} & \cdots & (-1)^{\langle \sigma_{n-b}, j_{\frac{N}{B}} \rangle} \end{bmatrix} \begin{bmatrix} X_{j_1} \\ \vdots \\ X_{j_{\frac{N}{B}}} \end{bmatrix} = \begin{bmatrix} 1 \\ \pm 1 \\ \vdots \\ \pm 1 \end{bmatrix},$$

where $\sigma_i, i \in \{1, \dots, n\}$ denotes the i -th column of the matrix Σ . The left-hand side matrix in the expression above, is $(n - b + 1) \times 2^{n-b}$. As $\sigma_1, \dots, \sigma_{n-b}$ are linearly independent, all the columns are different and are (omitting the top row) the exhaustive list of all 2^{n-b} possible ± 1 vectors. Thus the right-hand side vector is always one of the columns of the matrix and there is a unique solution with only one nonzero component (1-sparse solution) to this system whose support can be uniquely identified. Adding any vector from the null space of the matrix to this solution yields another solution. However, we show that this matrix is full rank (its null space has dimension $2^{n-b} - n + b - 1$), and assuming a continuous distribution for the nonzero components X_j , the probability that $\{X_j\}_{j \in \mathcal{I}_k}$ falls in this null space is zero.

To prove that the matrix is indeed full rank, let us first focus on the sub-matrix obtained by removing the first row. Let us call this matrix A . Also, let $M = -2I + \mathbf{1}\mathbf{1}^\top$, where I is the identity matrix of order $n - b$ and $\mathbf{1}$ is the all-one vector of dimension $(n - b)$. We can simply check that all the components of M are ± 1 . Hence, the columns of M are contained among the columns of the submatrix A . It is not difficult to check that M is a symmetric matrix, thus by spectral decomposition, it has $n - b$ orthogonal eigen-vectors $v_i, i \in [n - b]$. It is also easy to see that the normalized all-one vector $v_0 = \frac{\mathbf{1}}{\sqrt{n-b}}$ of dimension $n - b$ is an eigen-vector of M with eigen-value $\lambda_0 = n - b - 2$. Moreover, as the eigen-vectors are orthogonal to each other, we obtain that $v_i^\top M v_i = \lambda_i = -2$, where we used $v_i^\top \mathbf{1} = v_i^\top v_0 = 0$ for $i \neq 0$. Thus, for $n - b \neq 2$ all the eigen-values are nonzero and M is invertible, which implies that the sub-matrix A is full rank. In the case where $n - b = 2$, one can notice that the Hadamard matrix of size 2 will be contained as a submatrix, and thus the matrix will be full rank.

Now it remains to prove that the initial matrix is also full rank with a rank of $n - b + 1$. Assume that the columns of the matrix are arranged in the lexicographical order such that neglecting the first row, the first and the last column are all 1 and all -1 . If we consider any linear combination of the rows except the first one, it is easy to see that the first and the last element in the resulting row vector have identical magnitudes but opposite signs. This implies that the all-one row cannot be written as a linear combination of the other rows of the matrix. Therefore, the rank of the matrix must be $n - b + 1$.

To prove (2.8), let Σ_L and Σ_R be the matrices containing the first $n - b$ and the last b columns of Σ respectively, such that $\Sigma = [\Sigma_L \Sigma_R]$. If there is only one coefficient in the bin, then (2.6) implies that $\hat{v} = [(j^\top \Sigma_L) \ 0]^\top$. Using definitions (2.3) and (2.5), we obtain that $\Psi_b \mathcal{H}j = [0 \ (j^\top \Sigma_R)]^\top$. We observe that they sum to $\Sigma^\top j$ and the proof follows. ■

2.C Proof of Proposition 2.3

For $t \in [K]$, let H_t denote the size of the random set obtained by picking t objects from $[N]$ independently and uniformly at random with replacement. Let a_t and v_t denote the average and the variance of H_t for $t \in [K]$. It is not difficult to see that $\{H_t\}_{t \in [K]}$ is a Markov process. Moreover,

$$\mathbb{E}[H_{t+1} - H_t | H_t] = (1 - H_t/N), \quad (2.11)$$

because the size of the random set increases by one if and only if we choose an element from $[N]$ that has not been selected until time t , and conditioned on H_t , this happens with probability $1 - \frac{H_t}{N}$. This implies that $a_{t+1} = 1 + \gamma a_t$, where $\gamma = 1 - \frac{1}{N}$. Solving this equation, with initialization $a_0 = 1$, we obtain that

$$a_t = \sum_{r=0}^t \gamma^r = \frac{1 - \gamma^{t+1}}{1 - \gamma} = N(1 - \gamma^{t+1}). \quad (2.12)$$

In particular, $a_K = N(1 - (1 - \frac{1}{N})^K)$, which implies that

$$\begin{aligned} \mathbb{E}\left[\frac{H_K}{K}\right] &= \frac{N}{K} \left(1 - \left(1 - \frac{1}{N}\right)^K\right) \\ &\geq \frac{N}{K} \left(1 - \left(1 - \frac{K}{N} + \frac{K(K-1)}{2N^2}\right)\right) \\ &\geq 1 - \mathcal{O}\left(\frac{K}{N}\right). \end{aligned}$$

We can see that for $K = N^\alpha$, $\alpha \in (0, 1)$, as N tend to infinity, $\mathbb{E}\left[\frac{H_K}{K}\right]$ converges to 1. To find the variance of H_t , we use the formula

$$\text{Var}(H_{t+1}) = \mathbb{E}[\text{Var}(H_{t+1} | H_t)] + \text{Var}(\mathbb{E}[H_{t+1} | H_t]). \quad (2.13)$$

Using Eq. (2.11), we obtain that

$$\text{Var}(\mathbb{E}[H_{t+1} | H_t]) = \text{Var}(1 + \gamma H_t) = \gamma^2 v_t, \quad (2.14)$$

where v_t denotes the variance of H_t . Moreover, for the first term in Eq. (2.13), we have

$$\begin{aligned} \mathbb{E}[\text{Var}(H_{t+1}|H_t)] &= \mathbb{E}_{H_t}[\text{Var}(H_{t+1}|H_t = h_t)] \\ &= \mathbb{E}_{H_t}[\text{Var}(H_{t+1} - H_t|H_t = h_t)] \\ &\stackrel{(a)}{=} \mathbb{E}\left[\frac{H_t}{N}\left(1 - \frac{H_t}{N}\right)\right] \\ &= \frac{a_t}{N} + \frac{a_t^2 + v_t}{N^2}, \end{aligned} \quad (2.15)$$

where in (a), we used the fact that given H_t , $H_{t+1} - H_t$ is a Bernoulli random variable that is zero with probability $\frac{H_t}{N}$, thus its variance is equal to $\frac{H_t}{N}(1 - \frac{H_t}{N})$. Combining (2.14) and (2.15), we obtain

$$v_{t+1} = \left(\gamma^2 + \frac{1}{N^2}\right)v_t + \frac{a_t}{N}\left(1 + \frac{a_t}{N}\right). \quad (2.16)$$

From (2.12), it is easy to see that a_t is an increasing function of t . Moreover, from (2.16) it is seen that v_{t+1} is an increasing function of a_t , thus if we consider the following recursion

$$w_{t+1} = \left(\gamma^2 + \frac{1}{N^2}\right)w_t + \frac{a_K}{N}\left(1 + \frac{a_K}{N}\right),$$

then for any $t \in [K]$, $v_t \leq w_t$. We can also check that w_t , starting with the initialization $w_0 = 0$, is also an increasing sequence of t , thus we have

$$\begin{aligned} v_K \leq w_K \leq w_\infty &= \frac{a_K}{N}\left(1 + \frac{a_K}{N}\right) / \left(1 - \gamma^2 - \frac{1}{N^2}\right) \\ &= \frac{a_K}{2}\left(1 + \frac{a_K}{N}\right) / \left(1 - \frac{1}{N}\right). \end{aligned}$$

Using Chebyshev's inequality, we obtain that for any $\epsilon > 0$

$$\mathbb{P}\left\{\frac{H_K}{K} \leq (1 - \epsilon)\right\} \leq \frac{v_K}{K^2(\epsilon + 1 - \frac{a_K}{K})^2} = O\left(\frac{1}{\epsilon^2 K}\right).$$

Obviously, $\frac{H_K}{K} \leq 1$, thus $\frac{H_K}{K}$ converges to 1 in probability as N , and as a result K , tends to infinity. \blacksquare

2.D Proof of Proposition 2.9

Let S be any subset of unrecovered variable nodes of size at most ηK , where we will choose η later. Let $\mathcal{N}_i(S)$, $i \in [C]$, be the check neighbors of S in hash i . If for at least one of the hashes $i \in [C]$, $|\mathcal{N}_i(S)| > \frac{|S|}{2}$, there must be at least one check node of degree 1 (a singleton) among the check neighbors $\mathcal{N}_i(S)$, thus the peeling decoder can still proceed to decode further variable nodes.

Let \mathcal{E}_s^i denote the event that a specific subset A of size s of variable nodes has at most $\frac{s}{2}$ check neighbors in hash i . Also let $\mathcal{E}_s = \cap_{i=1}^C \mathcal{E}_s^i$. By the construction of the ensemble \mathcal{G} , it is easy to see that $\mathbb{P}\{\mathcal{E}_s\} = \prod_{i=1}^C \mathbb{P}\{\mathcal{E}_s^i\}$. Let T be any subset of check nodes in hash i of size $\frac{s}{2}$. The probability that all the neighbors of A in hash i belong to a specific set T of size $\frac{s}{2}$ is equal

to $(\frac{s}{2B})^s$, where B is the total number of output hash bins. Taking the union bound over $\binom{B}{s/2}$ of all such sets, it is seen that $\mathbb{P}\{\mathcal{E}_s^i\} \leq \binom{B}{s/2} (\frac{s}{2B})^s$, which implies that $\mathbb{P}\{\mathcal{E}_s\} \leq \left(\binom{B}{s/2} (\frac{s}{2B})^s\right)^C$. Taking the union bound over all possible subsets of size s of variables, we obtain that

$$\begin{aligned} \mathbb{P}\{F_s\} &\leq \binom{K}{s} \mathbb{P}\{\mathcal{E}_s\} \leq \binom{K}{s} \left(\binom{B}{s/2} \left(\frac{s}{2B}\right)^s\right)^C \\ &\leq \left(\frac{eK}{s}\right)^s \left(\frac{2eB}{s}\right)^{sC/2} \left(\frac{s}{2B}\right)^{sC} \leq \frac{u^s s^{s(C/2-1)}}{K^{s(C/2-1)}}, \end{aligned}$$

where $u = e^{C/2+1} (\frac{\beta}{2})^{C/2}$ and where F_s denotes the event that the peeling decoder fails to decode a set of variables of size s . We also used the fact that for $n \geq m$, $\binom{n}{m} \leq (\frac{n}{m})^m$. Moreover, $\mathbb{P}\{F_1\} = \mathbb{P}\{F_2\} = 0$ because the number of hashes C is always more than or equal to three. Selecting $\eta = \frac{1}{2u^{2/(C-2)}}$ and applying the union bound, we obtain that

$$\begin{aligned} \mathbb{P}\{F\} &\leq \sum_{s=1}^{\eta K} \mathbb{P}\{F_s\} = \sum_{s=3}^{\eta K} \mathbb{P}\{F_s\} = \sum_{s=3}^{\eta K} \frac{u^s s^{s(C/2-1)}}{K^{s(C/2-1)}} \\ &= O\left(\frac{1}{K^{3(C/2-1)}}\right) = O\left(\frac{1}{N^{3\alpha(C/2-1)}}\right), \end{aligned}$$

where F is the event that the peeling decoder fails to decode all the variables. This completes the proof. \blacksquare

Chapter 3

Enhancement: Acoustic Rake Receivers*

Twinkle, twinkle, little bat!
How I wonder what you're at!
Up above the world you fly!
Like a teatray in the sky

Alice's Adventures in Wonderland
LEWIS CARROLL

3.1 Introduction

Signal enhancement is the task of increasing the signal-to-noise ratio (SNR), the ratio of useful signal to noise power, of a signal of interest. In this chapter we concentrate on signal enhancement by beamforming. Beamforming is a ubiquitous technique in wireless communications where the redundant measurements from multiple antennas are combined to reduce noise and interference. The success of this technique has led to arrays of antennas embedded both in the base stations and within the mobile handsets themselves. Beamforming can be equally applied in acoustics by using multiple microphones. While being an active topic of research for many years, the added processing power needed for these techniques has long been prohibitive for consumer applications. However, due to the falling price of computations and the introduction of cheap MEMS microphones, it has recently gained popularity as a possible enabler of hands free distant speech recognition technology. This is exemplified by the recent introduction of several speech-based human computer interfaces sporting arrays of microphones by companies such as Google

*This chapter is the result of joint work with Ivan Dokmanić and Martin Vetterli [39, 112].

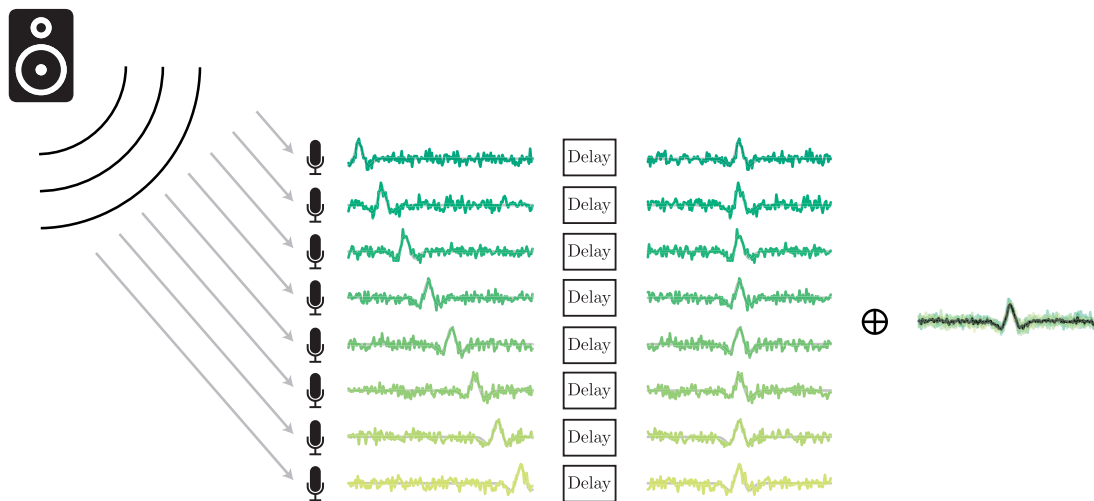


Figure 3.1: Beamforming 101: The simplest example of beamforming. A pulse emitted by a sound source is recorded at multiple spatially distributed microphones on the left. The pulse is degraded at the microphone input by normally distributed parasitic noise. According to sound propagation, the pulse will reach the microphones with different delays according to their location. Knowing this pattern of delays, we can compensate them to align all the pulse copies to the same location as shown in the middle. Finally, by summing up all the signals and thanks to the incoherence of the noise, a gain in SNR proportional to the number of microphones used is achieved. The signal to the right illustrates this by drawing the sum over the scaled individual microphone signals.

and Amazon.

To understand how beamforming works, let us consider the sound from a single source impinging on a microphone array as depicted in Figure 3.1. Due to differences in propagation distance, the wavefront will reach each microphone with a corresponding delay. The simplest beamformer, known as *delay-and-sum* (DS) [121], then manipulates the phase of the microphone signals so that they align in time, before summing them all up. Considering only independent parasitic noise at the microphones, an SNR gain equal to the number of microphones will be achieved in the direction of the source. Figure 3.1 illustrates this principle in more details. The more general architecture of *filter-and-sum* beamforming, where an arbitrary finite impulse response (FIR) filter is applied to each microphone signal before summation, allows for more sophisticated beamforming algorithms. For example, one might want unity gain towards the source of interest while minimizing sounds from all other directions, leading to the well-known minimum variance distortionless response (MVDR) beamformer [24].

In free space, this would be the end of the story as the wavefront would continue to propagate and dissipate to infinity after reaching the microphone. In indoor environment, acquisition of clean speech signals is complicated by reflections of the sound on walls creating the effect known as reverberation. Reverberation notoriously degrades the intelligibility of speech signals and many techniques to reduce or suppress it have been proposed [92]. Due to its ability to focus the sound acquisition towards specific directions, beamforming has been particularly successful.

An important observation, however, is that early reflections carry useful redundant signal information. This did not escape wireless engineers when facing a similar situation with high-frequency radio signals bouncing off the ionosphere and arriving to receivers through several paths. In the case of wideband signals, these different paths can be clearly resolved at the receiver, and by optimal weighting and coherent summing of the different paths an SNR gain directly proportional to the total amount of signal present can be achieved [105]. The name Rake receiver was coined for this technique due to the similarity to the eponymous gardening tool, *raking* energy from the different paths. Originally proposed for single-input-single-output systems, the technique was later extended to arrays of antennas [70, 91] that exploit spatial diversity. Thus multipath components that are not resolvable with a single antenna because they arrive simultaneously, become resolvable because they arrive from different directions.

In spite of the success of the rake receivers in wireless communications, the principle has not received significant attention in room acoustics. Nevertheless, constructive use of echoes in rooms to improve beamforming has been mentioned in the literature [5, 68, 95]. In particular, the term *acoustic rake receiver* (ARR) was used in the SCENIC project proposal [5].

The list of ingredients for ARR in room acoustics is similar as in wireless communications: a wave (acoustic instead of electromagnetic) propagates in space; reflections and scattering cause the wave to arrive at the receiver through multiple paths in addition to the direct path, and these multipath components all contain the source waveform.

The main difference is that in room acoustics we do not get to design the input signal. Spreading sequences used in CDMA are designed to be near-orthogonal to their shifts and orthogonal between different users, which facilitates the multipath channel estimation; such orthogonality is not exhibited by speech. Moreover, speech segments are very long with respect to the time delay between two consecutive echoes, and they are *a priori* unknown at the receiver.

On the contrary, there are no significant differences in terms of the spatial structure. If we know where the echoes are coming from, we can design spatial processing algorithms—for example beamformers—that use multiple copies of the same signal arriving from different directions.

Imagine first that we know the room geometry. Then, if we localize the source, we can predict where its echoes will come from using simple geometric rules [4, 17]. Localizing the direct signal in a reverberant environment is a well-understood problem [129]. What is more, we do not need to know the room shape in detail—locations of the most important reflectors (ceiling, floor, walls) suffice to localize the major echoes. In many cases this knowledge is readily available from the floor plans or measurements. In ad-hoc deployments, the room geometry may be difficult to obtain. If that is the case, we can first perform a calibration step to learn it. An appealing method to infer the room geometry is by using sound, as was demonstrated recently [6, 37, 38, 108].

We may still be able to take advantage of the echoes without estimating the room geometry. Note that we are not after the room geometry itself; rather, we only need to know where the early echoes are coming from. Echoes can be seen as signals emitted by *image sources*—mirror images of the true source across reflecting walls [4]. Knowing where the echoes are coming from is equivalent to knowing where the image sources are.

Image source localization can be solved, for example, by *echo sorting* as described in [38]. Alternatively, O’Donovan, Duraiswami and Zotkin [95] propose to use an *audio camera* with a large number of microphones to find the images. Once the image sources are localized (in a calibration phase or otherwise), we can predict their movement using geometrical rules, as discussed in Section 3.6. Thus, the acoustic raking is a multi-stage process comprising image source localization, image source tracking, and beamforming weight computation. The complete

block diagram of an acoustic rake receiver is shown in Figure 3.2.

3.1.1 Related Work

It is interesting to note the analogy between the ARRs and the human auditory perception. It is well established that the early echoes improve speech intelligibility [19, 77]. In fact, adding energy in the form of early echoes (approximately within the first 50 ms of the room impulse response (RIR)) is equivalent to adding the same energy to the direct sound [19]. This observation suggests new designs for indoor beamformers, with different choices of performance measures and reference signals. A related discussion of this topic is given by Habets and co-authors [54], who examine the tradeoff between dereverberation and denoising in beamforming. In addition to the standard SNR, we propose to use the useful-to-detrimental ratio (UDR), first defined by Lochner and Burger [77], and used by Bradley, Sato and Picard [19]. We generalize UDR to a scenario with interferers, defining it as the ratio of the direct and early reflection energy to the energy of the noise and interference.

ARRs focus on the early part of the RIR, trying to concentrate the energy contained in the early echoes. In that regard, there are similarities between ARRs and channel shortening [122, 137]. Channel shortening produces filters that are much better behaved than complete inversion, *e.g.*, by the multiple-input-output-theorem (MINT) [48, 87]. Nevertheless, it is still tacitly assumed that we know the acoustic impulse responses between the sources and the microphones. In contrast to channel shortening, as well as other methods assuming this knowledge [13, 87], we never attempt the difficult task of estimating the impulse responses. Our task is simpler: we only need to detect the early echoes, and lift them to 3D space as image sources.

In many situations, the shape of the room can be known in advance from blueprints or measurements [38]. Then knowing the location of the real source allows to calculate the positions of the echoes. Localizing the direct sound is a well understood problem [129]. In ad-hoc deployment, recent works propose a calibration step to locate the main reflectors [6, 37, 38, 108]. Note that there is in fact no necessity to know the room geometry exactly, the positions of the image sources being sufficient. The echo sorting algorithm from [38] allows to locate the main echoes from measured RIR. Another approach is the audio camera of [95].

3.1.2 Main Contributions and Limitations

We introduce the acoustic rake receiver (ARR) as the echo-aware microphone beamformer. We present several formulations with different properties, and analyze their behavior theoretically and numerically. The analysis shows that ARRs lead to significantly improved SNR and interference cancellation when compared with standard beamformers that only extract the direct path. ARRs can suppress interference in cases when conventional beamforming is bound to fail, for example when an interferer is occluding the desired source (an illustration is given in Figure 3.3; for a sneak-peak of real beampatterns, fast-forward to Fig. Figure 3.7). We present optimal formulations that outperform the earlier DS approaches [68], especially when interferers are present. Significant gains are observed not only in terms of signal-to-interference-and-noise ratio (SINR) and UDR, but also in terms of perceptual evaluation of speech quality (PESQ) [110]. Inspired by the results in psychoacoustics mentioned earlier [77], we propose relaxed formulations leading to better behaved beamforming filters without sacrificing SINR, while maintaining tight control over pre-echoes.

The raking microphone beamformers are particularly well-suited to extracting the desired

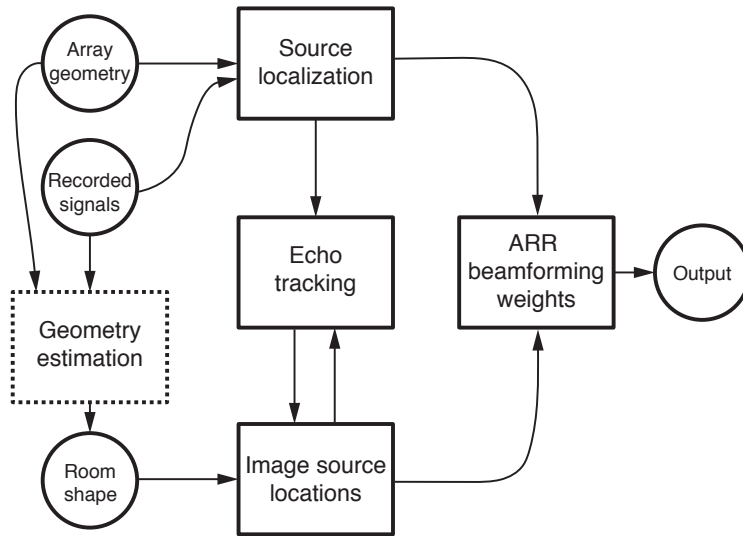


Figure 3.2: A block diagram for acoustic rake receivers. We focus on ARR beamforming weight computation, and we briefly discuss echo tracking and image source localization. The geometry estimation block is optional (room geometry could be known in advance), hence the dashed box.

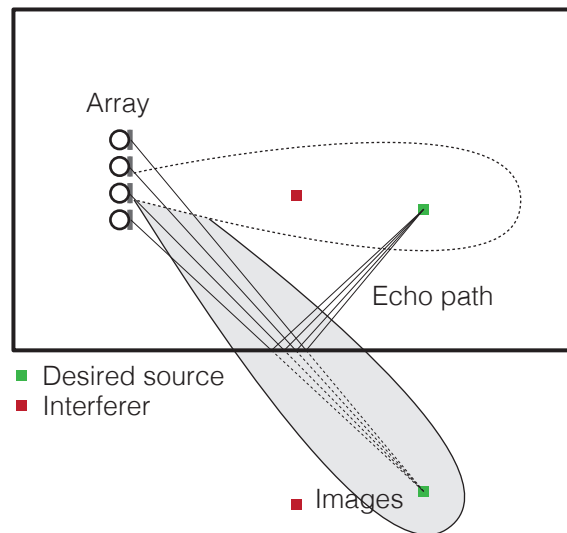


Figure 3.3: Listening behind an interferer by listening to echoes (illustration). A beam directed towards the desired, green source will necessarily *pick up* the red interferer's signal. Acoustic rake receiver avoids it by beamforming towards the echoes of the desired source.

speech signal in the presence of interfering sounds, in part because they can focus on echoes of the desired sound and cancel the echoes of the interfering signals. The analogous human capacity to focus on a particular acoustic stimulus while not perceiving other, unwanted sounds is called the *cocktail party effect* [59]. The title of one of the papers this chapter is based upon — Raking the Cocktail Party — was inspired by this analogy [39].

We propose formulations for ARRrS both in the frequency and time domain, each having their own strengths and weaknesses. Frequency domain formulation is simple and concise; it allows us to focus on objective gains from acoustic raking. Nonetheless, it does not allow precise control over critical parameters of the beamforming filters such as pre-echoes, delay, or length. Time-domain design let us control all these parameters, but comes at the cost of a larger optimization problem size.

Let us also mention some limitations of our results. For clarity, the numerical experiments are presented in a 2D “room”, and as such are directly applicable to planar (*e.g.* linear or circular) arrays. Extension to 3D arrays is straightforward. We do not discuss robust formulations that address uncertainties in the array calibration. Microphones are assumed to be ideally omnidirectional with a flat frequency response. Except for Section 3.6, we assume that the locations of the image sources are known. We explain how to find the image sources when the room geometry is either known or unknown, but we do not provide a deep overview of the geometry estimation techniques. To this end, we suggest a number of references for the interested reader. We consider the walls to be flat-fading; in reality, they are frequency selective. We do not discuss the estimation of various covariance matrices [25].

The results of this chapter are reproducible. Python (NumPy) [96] code for room impulse response generator, the beamforming routines, the STFT processing engine, and to generate the figures and the sound samples is available online at <http://lcav.github.io/AcousticRakeReceiver/> and <http://lcav.github.io/TimeDomainAcousticRakeReceiver/>. The room impulse response generator and the STFT engine were then forked to a dedicated python module that is presented in more details in Section 6.2 and made available at <http://lcav.github.io/pyroomacoustics/>.

3.1.3 Chapter Outline

This chapter is structured as follows. In Section 3.2 we lay out the signal model used to formulate the beamforming algorithms. Section 3.3 summarizes the discretization of impulse responses for the numerical evaluation of beamforming algorithms. Then, Sections 3.4 and 3.5 introduce the frequency and time domain formulations or ARRrS, respectively. Each of these sections is divided in a short introduction, presentation of the ARRrS formulations, and numerical experiments. Section 3.4 has in addition some theoretical guarantees for the Raking DS beamformer in Section 3.4.3. Section 3.6 explains how to locate the image sources, and comments on localizing the direct source.

3.2 Signal Model

We denote all matrices by bold uppercase letters, for example \mathbf{A} , and all vectors by bold lowercase letters, for example \mathbf{x} . The Hermitian transpose of a matrix or a vector is denoted by $(\cdot)^H$, as in \mathbf{A}^H , and the Euclidean norm of a vector by $\|\cdot\|$, that is, $\|\mathbf{x}\| \stackrel{\text{def}}{=} (\mathbf{x}^H \mathbf{x})^{1/2}$.

Suppose that the desired source of sound is at the location \mathbf{s}_0 in a room. Sound from this source arrives at the microphones located at $[\mathbf{r}_m]_{m=1}^M$ via the direct path, but also via the

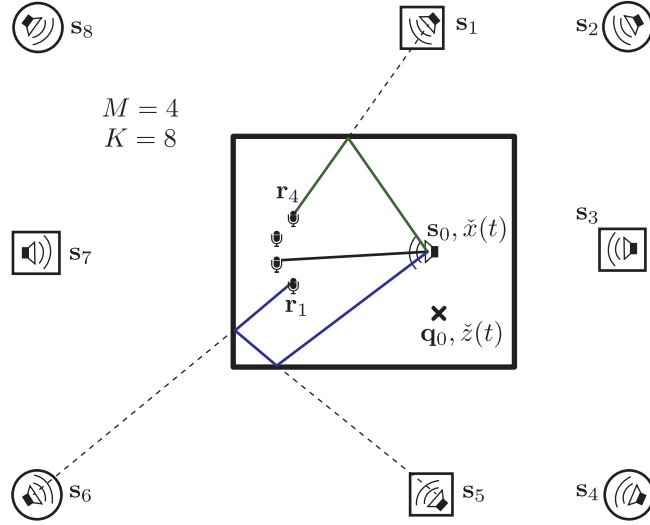


Figure 3.4: Illustration of the notation and concepts. Echoes of the desired signal emitted at \mathbf{s}_0 can be modeled as a direct sound coming from the image sources of \mathbf{s}_0 . Two generations of image sources are illustrated: first ($\mathbf{s}_1, \mathbf{s}_3, \mathbf{s}_5, \mathbf{s}_7$) and second ($\mathbf{s}_2, \mathbf{s}_4, \mathbf{s}_6, \mathbf{s}_8$), as well as the corresponding *sound rays* for \mathbf{s}_5 and \mathbf{s}_6 . The interferer is located at \mathbf{q}_0 (its image sources are not shown), and the microphones are located at $\mathbf{r}_1, \dots, \mathbf{r}_4$.

echoes from the walls. The echoes can be replaced by the image sources—mirror images of the true sources across the corresponding walls—according to the image source model [4, 17]. An important consequence is that instead of modeling the source of the desired or the interfering signal as a single point in a room, we can model it as a collection of points in free space. A more detailed discussion of the image source model is given in Section 3.6.

Denote the signal emitted by the source $\tilde{x}[n]$ (*e.g.* the speech signal). Then all the image sources emit $\tilde{x}[n]$ as well, and the signals from the image sources reach the microphones with the appropriate delays. In our application, the essential fact is that the echoes correspond to image sources. We denote the image source positions by \mathbf{s}_k , $1 \leq k \leq K$, where K denotes the largest number of image sources considered. Note that we do not care about the sequence of walls that generates \mathbf{s}_k , nor do we care about how many walls are in the sequence. For us, all \mathbf{s}_k are simply additional sources of the desired signal. The described setup is illustrated Figure 3.4.

Suppose further that there is an interferer at the location \mathbf{q}_0 (for simplicity, we consider only a single interferer). The interferer emits the signal $\tilde{z}[n]$, and its image sources emit $\tilde{z}[n]$ as well. Similarly as for the desired source, we denote by \mathbf{q}_k , $1 \leq k \leq K'$ the positions of the interfering image sources, with K' being the largest number of interfering image source considered. The model mismatch (*e.g.*, the image sources of high orders and the late reverberation) and the noise are absorbed in the term $\tilde{n}_m[n]$.

The signal received by the m th microphone is then a sum of convolutions

$$y_m[n] = \sum_{k=0}^K (a_m(\mathbf{s}_k) * x)[n] + \sum_{k=0}^{K'} (a_m(\mathbf{q}_k) * z)[n] + \tilde{n}_m[n], \quad (3.1)$$

where $\tilde{a}_m(\mathbf{s}_k)$ denotes the impulse response of the channel between the source located at \mathbf{s}_k and

the m th microphone—in this case a delay and a scaling factor.

We design and analyze all the beamformers in the frequency domain. That is, we will be working with the discrete-time Fourier transform (DTFT) of the discrete-time signal \tilde{x} ,

$$x(e^{j\omega}) \stackrel{\text{def}}{=} \sum_{n \in \mathbb{Z}} \tilde{x}[n] e^{-j\omega n}.$$

In practical implementations, we use the discrete-time short-time Fourier transform (STFT). More implementation details are given in Section 3.3.

Using these notations, we can write the signal picked up by the m th microphone as

$$y_m(e^{j\omega}) = \sum_{k=0}^K a_m(\mathbf{s}_k, \Omega) x(e^{j\omega}) + \sum_{k=0}^{K'} a_m(\mathbf{q}_k, \Omega) z(e^{j\omega}) + n_m(e^{j\omega}),$$

where $n_m(e^{j\omega})$ models the noise and other errors, and $a_m(\mathbf{s}_k, \Omega)$ denotes the m th component of the steering vector for the source \mathbf{s}_k . The steering vector is the Fourier transform of the continuous version of the impulse response $\tilde{a}(\mathbf{s}_k)$, evaluated at the frequency Ω . The discrete-time frequency ω and the continuous-time frequency Ω are related as $\omega = \Omega T_s$, where T_s is the sampling period. The steering vector is then simply $\mathbf{a}(\mathbf{s}_k, \Omega) = [a_m(\mathbf{s}_k, \Omega)]_{m=0}^{M-1}$.

We can write out the entries of the steering vectors explicitly for a point source in free space. They are given as the appropriately scaled free-space Green's functions for the Helmholtz equation [42],

$$a_m(\mathbf{s}_k, \Omega) = \frac{\alpha_k(\mathbf{s}_k)}{4\pi \|\mathbf{r}_m - \mathbf{s}_k\|} e^{-j\kappa \|\mathbf{r}_m - \mathbf{s}_k\|}, \quad (3.2)$$

where we define the wavenumber as $\kappa \stackrel{\text{def}}{=} \Omega/c$, c being the speed of sound, and $\alpha_k(\mathbf{s}_k)$ is the attenuation corresponding to \mathbf{s}_k . In the time domain, instead of steering vectors, we have to consider one impulse response for each source/microphone pair. The impulse response is the equivalent of the previous equation in the time domain and involves a time delay and an attenuation, namely

$$a_m(\mathbf{s}_k, t) = \frac{\alpha_k(\mathbf{s}_k)}{4\pi \|\mathbf{r}_m - \mathbf{s}_k\|} \delta\left(t - \frac{\|\mathbf{r}_m - \mathbf{s}_k\|}{c}\right), \quad (3.3)$$

where δ is the Dirac delta function.

3.3 Simulation of Beamforming Algorithms

The performance of the beamforming algorithms developed in this chapter is done primarily through simulation. The methodology being the same for frequency and time domain beamformers, it is of interest to give it an overview before going into the particulars of beamformer designs. We use a simple room acoustic framework written in Python, which relies on Numpy and Scipy for matrix computations [96]. This framework was written specifically to suit our needs for this project, but we believe however that it could be of interest to a wider audience and describe it in more details in Chapter 6.

We limit ourselves in this chapter to 2D geometry and rectangular rooms. In all experiments, the sampling frequency F_s was set to 8 kHz. An overview of the simulation setup is shown in Figure 3.5.

Starting from the room geometry and the positions of the sources and microphones, we first compute the locations of all images sources up to a fixed number of generations. The reflectivity

Table 3.1: Summary of notation.

Symbol	Meaning
M	Number of microphones
\mathbf{r}_m	Location of the m th microphone
\mathbf{s}_0	Location of the desired source
\mathbf{s}_i	Location of the i th image of the desired source ($i \geq 1$)
\mathbf{q}_0	Location of the interfering source
\mathbf{q}_i	Location of the i th image of the interfering source ($i \geq 1$)
$x[n], x(e^{j\omega})$	Sound from the desired source in the time/DTFT domain, respectively
$z[n], z(e^{j\omega})$	Sound from the interfering source in the time/DTFT domain, respectively
$\mathbf{w}(e^{j\omega})$	Vector of beamformer weights
\mathbf{g}	Vector of stacked time domain beamforming filters
K	Number of considered desired image sources
K'	Number of considered interfering image sources
$a_m(\mathbf{s}, \Omega)$	m th component of the steering vector for a source at \mathbf{s}
$a_m(\mathbf{s}, t)$	Impulse response between the m th microphone and a source at \mathbf{s}
y_m	Signal picked up by the m th microphone
$\ \cdot\ $	Euclidean norm, $\ \mathbf{x}\ = (\sum x_i ^2)^{1/2}$.

of the walls is fixed to a constant, 0.9 in this chapter. Since computer simulation are limited to using audio signals sampled at a finite rate F_s for the sources, the continuous impulse response of (3.3) needs to be discretized. This discretization of the channel response into an FIR filter is done by convolution with an ideal low-pass filter,

$$a_m(\mathbf{s}_k, n) = \int_{-\infty}^{\infty} a_m(\mathbf{s}_k, u) \operatorname{sinc}(n - F_s u) du = \frac{\alpha(\mathbf{s}_k)}{4\pi\|\mathbf{r}_m - \mathbf{s}_k\|} \operatorname{sinc}\left(n - F_s \frac{\|\mathbf{s}_k - \mathbf{r}_m\|}{c}\right). \quad (3.4)$$

The length of the FIR is then limited to length L_h either by truncation or windowing of the cardinal sine function. Finally, the RIR between the real source \mathbf{s}_0 and the microphone \mathbf{r}_m is obtained by summing over the impulse response of \mathbf{s}_0 and its images up to K

$$a_m(\mathbf{s}_0, n) = \sum_{k=0}^K a_m(\mathbf{s}_k, n) = \sum_{k=0}^K \frac{\alpha_k}{4\pi\|\mathbf{r}_m - \mathbf{s}_k\|} \operatorname{sinc}\left(n - F_s \frac{\|\mathbf{r}_m - \mathbf{s}_k\|}{c}\right),$$

where K is the number of image sources considered¹. This discretization will also be used later in the formulation of time domain beamformers.

The general flow of the simulator, illustrated in Figure 3.5, is as follows. The information from the room geometry and the different sources locations in the room is used to generate all the image sources requested. The images are used in turn to create the sampled impulse

¹this K will generally much larger than the number of image sources considered in the design of beamformers in the following sections

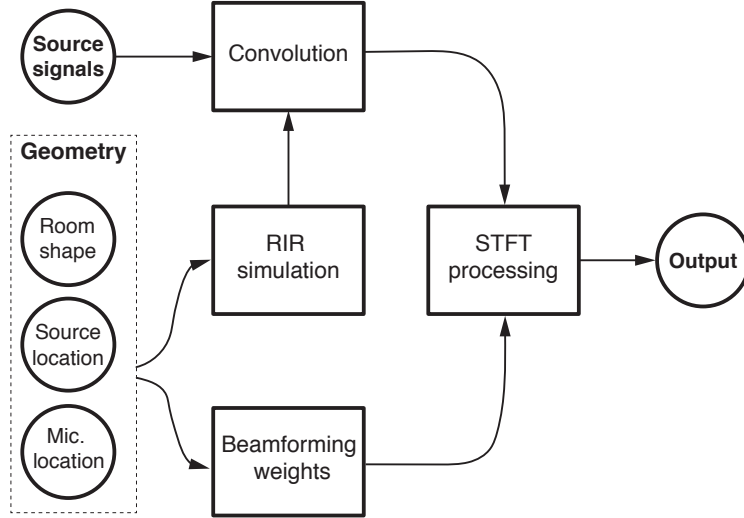


Figure 3.5: Block diagram of the simulation setup used for numerical experiments.

responses between sources and microphones as described above. The impulse responses are used to simulate the propagation of the source signals to the microphones. There, the beamforming filters are applied to the incoming signal either directly in the time domain, or in the frequency domain using an STFT engine. In the case of STFT, we synthesize the output signal using the conventional overlap-add method [118].

3.4 Frequency Domain Formulations

This section treats the frequency domain formulations of ARR. We begin by introducing some notation in Section 3.5.1 and follow by reviewing some of the classic beamformer designs in Section 3.4.1. Then comes the introduction of ARR designs in Section 3.4.2. All the beamformers presented are summarized in Table 3.2. Next, we give some theoretical insights into the expected performance of ARR. We conclude this section by presenting the result of comprehensive numerical experiments in Section 3.4.4.

Before we start, we introduce some notation specific to the frequency domain formulations. Using vector notation, the microphone signals can be written concisely as

$$\mathbf{y}(e^{j\omega}) = \mathbf{A}_s(e^{j\omega})\mathbf{1}x(e^{j\omega}) + \mathbf{A}_q(e^{j\omega})\mathbf{1}z(e^{j\omega}) + \mathbf{n}(e^{j\omega}), \quad (3.5)$$

where $\mathbf{A}_s(e^{j\omega}) \stackrel{\text{def}}{=} [\mathbf{a}(\mathbf{s}_1, \Omega), \dots, \mathbf{a}(\mathbf{s}_K, \Omega)]$, $\mathbf{A}_q(e^{j\omega}) \stackrel{\text{def}}{=} [\mathbf{a}(\mathbf{q}_1, \Omega), \dots, \mathbf{a}(\mathbf{q}_{K'}, \Omega)]$, and $\mathbf{1}$ is the all-ones vector. From here onward, we suppress the frequency dependency of the steering vectors and the beamforming weights to reduce the notational clutter.

3.4.1 Classic Beamformers

Microphone beamformers combine the outputs of multiple microphones in order to achieve spatial selectivity, thereby suppressing noise and interference [125]. In the frequency domain, a

beamformer forms a linear combination of the microphone outputs to yield the output u . That is,

$$u = \mathbf{w}^H \mathbf{y} = \mathbf{w}^H \mathbf{A}_s \mathbf{1}x + \mathbf{w}^H \mathbf{A}_q \mathbf{1}z + \mathbf{w}^H \mathbf{n},$$

where the vector $\mathbf{w} \in \mathbb{C}^M$ contains the beamforming weights.

The weights \mathbf{w} are often selected so that they optimize some design criterion. Common examples of beamformers are the delay-and-sum (DS) beamformer, minimum-variance-distortionless-response (MVDR) beamformer, maximum-signal-to-interference-and-noise (Max-SINR) beamformer, and minimum-mean-squared-error (MMSE) beamformer. In this section we discuss the rake formulation of the DS and the Max-SINR beamformers; for completeness, we first describe the non-raking variants.

Delay-and-Sum Beamformer

DS is the simplest and often quite effective beamformer [125]. Assume that we want to listen to a source at \mathbf{s} . Then we form the DS beamformer by compensating the propagation delays from the source \mathbf{s} to the microphones \mathbf{r}_m ,

$$u_{\text{DS}} = \frac{1}{M} \sum_{m=0}^{M-1} y_m e^{j\kappa \|\mathbf{r}_m - \mathbf{s}\|} \quad (3.6)$$

$$\approx \frac{x}{4\pi \|\bar{\mathbf{r}} - \mathbf{s}\|} + \frac{1}{M} \sum_{m=0}^{M-1} n_m, \quad (3.7)$$

where $\bar{\mathbf{r}} = \frac{1}{M} \sum_{m=0}^{M-1} \mathbf{r}_m$ denotes the center of the array. The beamforming weights can be read out from (3.6) as

$$\mathbf{w}_{\text{DS}} = \mathbf{a}(\mathbf{s}) / \|\mathbf{a}(\mathbf{s})\|, \quad (3.8)$$

where we used the definition of y_m (3.5) and the definition of the steering vector (3.2). We can see from (3.7) that if $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_M)$, then the output noise is distributed according to $\mathcal{N}(0, \sigma^2/M)$, that is, we obtain an M -fold decrease in the noise variance at the output with respect to any reference microphone.

Maximum Signal-to-Interference-and-Noise Ratio Beamformer

The SINR is an important figure of merit used to assess the performance of ARRs, and to compare it with the standard non-raking beamformers. It is computed as the ratio of the power of the desired output signal to the power of the undesired output signal. The desired output signal is the output signal due to the desired source, while the undesired signal is the output signal due to the interferers and noise.

For a desired source at \mathbf{s} and an interfering source at \mathbf{q} we can write

$$\text{SINR} \stackrel{\text{def}}{=} \frac{\mathbb{E} |\mathbf{w}^H \mathbf{a}(\mathbf{s})x|^2}{\mathbb{E} |\mathbf{w}^H (\mathbf{a}(\mathbf{q})z + \mathbf{n})|^2} = \sigma_x^2 \frac{\mathbf{w}^H \mathbf{a}(\mathbf{s}) \mathbf{a}(\mathbf{s})^H \mathbf{w}}{\mathbf{w}^H \mathbf{K}_{nq} \mathbf{w}}, \quad (3.9)$$

where \mathbf{K}_{nq} is the covariance matrix of the noise and the interference.

It is compelling to pick \mathbf{w} that maximizes the SINR (3.9) [125]. The maximization can be solved by noting that the rescaling of the beamformer weights leaves the SINR unchanged. This

means that we can minimize the denominator subject to numerator being an arbitrary constant. The solution is given as

$$\mathbf{w}_{\text{SINR}} = \frac{\mathbf{K}_{nq}^{-1} \mathbf{a}_s}{\mathbf{a}_s^H \mathbf{K}_{nq}^{-1} \mathbf{a}_s}.$$

Using the definition (3.9), we can derive the SINR for the Max-SINR beamformer as

$$\text{SINR} = \sigma_x^2 \mathbf{a}_s^H \mathbf{K}_{nq}^{-1} \mathbf{a}_s.$$

Because \mathbf{K}_{nq}^{-1} is a Hermitian symmetric positive definite matrix, it has an eigenvalue decomposition as $\mathbf{K}_{nq}^{-1} = \mathbf{U}^H \mathbf{\Lambda} \mathbf{U}$, where \mathbf{U} is unitary, and $\mathbf{\Lambda}$ is diagonal with positive entries. We can write $\mathbf{a}^H \mathbf{K}_{nq}^{-1} \mathbf{a} = (\mathbf{U} \mathbf{a})^H \mathbf{\Lambda} (\mathbf{U} \mathbf{a})$. Because $\|\mathbf{U} \mathbf{a}\|^2 = \|\mathbf{a}\|^2$, and because $\mathbf{\Lambda}$ is positive, increasing $\|\mathbf{a}\|^2$ typically leads to an increased SINR, although we can construct counterexamples. This will be important when we discuss the SINR gain of the Rake-Max-SINR beamformer.

3.4.2 Acoustic Rake Receiver

In this section, we present several formulations of the ARR. The Rake-DS beamformer is a straightforward generalization of the conventional DS beamformer. The one-forcing beamformer implements the idea of steering a fixed beam power towards every image source, while trying to minimize the interference and noise. The Rake-Max-SINR and Rake-Max-UDR beamformers optimize the corresponding performance measures; we show in Section 3.4.4 that the Rake-Max-SINR beamforming performs best in terms of all evaluation criteria except, as expected, in terms of UDR.

Delay-and-Sum Raking

If we had access to every echo separately (*i.e.* not summed with all the other echoes), we could align them all to maximize the performance gain. Unfortunately, this is not the case: each microphone picks up the convolution of speech with the impulse response, which is effectively a sum of overlapping echoes of the speech signal. If we only wanted to extract the direct path, we would use the standard DS beamformer (3.8). To build the Rake-DS receiver, we create a DS beamformer for every image source, and average the outputs,

$$\frac{1}{K+1} \sum_{k=0}^K \frac{\alpha'_k}{M} \sum_{m=0}^{M-1} y_m e^{j\kappa \|\mathbf{r}_m - \mathbf{s}_k\|}, \quad (3.10)$$

where $\alpha'_k \stackrel{\text{def}}{=} \alpha_k / (4\pi \|\mathbf{r}_m - \mathbf{s}_k\|)$. We read out the beamforming weights from (3.10) as

$$\mathbf{w}_{\text{R-DS}} = \frac{1}{\|\sum_k \mathbf{a}(\mathbf{s}_k)\|} \sum_{k=0}^K \mathbf{a}(\mathbf{s}_k) = \frac{\mathbf{A}_s \mathbf{1}}{\|\mathbf{A}_s \mathbf{1}\|},$$

where we chose the scaling in analogy with (3.8) (scaling of the weights does not alter the output SINR). It can be seen that this is just a scaled sum of the steering vectors for each image source.

Table 3.2: Summary of frequency-domain beamformers.

Acronym	Description	Beamforming Weights
DS	Align delayed copies of signal at the microphone	$\mathbf{w}_{\text{DS}} = \mathbf{a}(s)/\ \mathbf{a}(s)\ $
Max-SINR	$\max. \mathbf{w}^H \mathbf{a}_s \mathbf{a}_s^H \mathbf{w} / (\mathbf{w}^H \mathbf{K}_{nq} \mathbf{w})$	$\mathbf{w}_{\text{SINR}} = \mathbf{K}_{nq}^{-1} \mathbf{a}_s / (\mathbf{a}_s^H \mathbf{K}_{nq}^{-1} \mathbf{a}_s)$
Rake-DS	Weighted average of DS beamformers over image sources	$\mathbf{w}_{\text{R-DS}} = \mathbf{A}_s \mathbf{1} / \ \mathbf{A}_s \mathbf{1}\ $
Rake-OF	$\min. \mathbb{E} \left \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right ^2$, s.t. $\mathbf{w}^H \mathbf{A}_s = \mathbf{1}^\top$	$\mathbf{w}_{\text{OF}} = \mathbf{K}_{nq}^{-1} \mathbf{A}_s (\mathbf{A}_s^H \mathbf{K}_{nq}^{-1} \mathbf{A}_s)^{-1} \mathbf{1}_M$
Rake-Max-SINR	$\max. \mathbb{E} \left \sum_{k=0}^K \mathbf{w}^H \mathbf{a}(s_k) x \right ^2 / \mathbb{E} \left \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right ^2$	$\mathbf{w}_{\text{R-SINR}} = \mathbf{K}_{nq}^{-1} \mathbf{A}_s \mathbf{1} / (\mathbf{1}^H \mathbf{A}_s^H \mathbf{K}_{nq}^{-1} \mathbf{A}_s \mathbf{1})$
Rake-Max-UDR	$\max. \mathbb{E} \sum_{k=0}^K \mathbf{w}^H \mathbf{a}(s_k) x ^2 / \mathbb{E} \left \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right ^2$	$\mathbf{w}_{\text{R-UDR}} = \mathbf{C}^{-1} \tilde{\mathbf{w}}_{\max} ((\mathbf{C}^{-1})^H \mathbf{A}_s \mathbf{A}_s^H \mathbf{C}^{-1})$

One-Forcing Raking

A different approach, based on intuition, is to design a beamformer that listens to all K image sources with the same power, and at the same time minimizes the noise and interference energy:

$$\begin{aligned} & \text{minimize}_{\mathbf{w} \in \mathbb{C}^M} \mathbb{E} \left| \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right|^2 \\ & \text{subject to } \mathbf{w}^H \mathbf{a}(s_k) = 1, \forall 0 \leq k \leq K. \end{aligned}$$

Alternatively, we may choose to null the interfering source and its image sources. Both cases are an instance of the standard linearly-constrained-minimum-variance (LCMV) beamformer [47]. Collecting all the steering vectors in a matrix, we can write the constraint as $\mathbf{w}^H \mathbf{A}_s = \mathbf{1}^\top$. The solution can be found in closed form as

$$\mathbf{w}_{\text{OF}} = \mathbf{K}_{nq}^{-1} \mathbf{A}_s (\mathbf{A}_s^H \mathbf{K}_{nq}^{-1} \mathbf{A}_s)^{-1} \mathbf{1}_M.$$

A few remarks are in order. First, with M microphones, it does not make sense to increase K beyond M , as this results in more constraints than degrees of freedom. Second, using this beamformer is a bad idea if there is an interferer along the ray through the microphone array and any of image sources.

As with all LCMV beamformers, adding linear constraints uses up degrees of freedom that could otherwise be used for noise and interference suppression. It is better to let the “beamformer decide” or “the beamforming procedure decide” on how to maximize a well-chosen cost function; one such procedure is described in the next subsection.

Max-SINR Raking

The main workhorse of this section is the Rake-Max-SINR. We compute the weights so as to maximize the SINR, taking into account the echoes of the desired signal, and the echoes of the interfering signal,

$$\text{maximize}_{\mathbf{w} \in \mathbb{C}^M} \frac{\mathbb{E} \left| \sum_{k=0}^K \mathbf{w}^H \mathbf{a}(s_k) x \right|^2}{\mathbb{E} \left| \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right|^2}. \quad (3.11)$$

The logic behind this expression can be summarized as follows: we present the beamforming procedure with a set of good sources, whose influence we aim to maximize at the output, and with a set of bad sources, whose power we try to minimize at the output. Interestingly, this leads to the standard Max-SINR beamformer with a structured steering vector and covariance matrix. Define the combined noise and interference covariance matrix as

$$\mathbf{K}_{nq} \stackrel{\text{def}}{=} \mathbf{K}_n + \sigma_z^2 \left(\sum_{k=0}^{K'} \mathbf{a}(q_k) \right) \left(\sum_{k=0}^{K'} \mathbf{a}(q_k) \right)^H,$$

where \mathbf{K}_n is the covariance matrix of the noise term, and σ_z^2 is the power of the interferer at a particular frequency.

Then the solution to (3.11) is given as

$$\mathbf{w}_{\text{R-SINR}} = \frac{\mathbf{K}_{nq}^{-1} \mathbf{A}_s \mathbf{1}}{\mathbf{1}^H \mathbf{A}_s^H \mathbf{K}_{nq}^{-1} \mathbf{A}_s \mathbf{1}}.$$

It is interesting to note that when $\mathbf{K}_{nq} = \sigma^2 \mathbf{I}_M$ (e.g. no interferers and iid noise), the Rake-Max-SINR beamformer reduces to $\mathbf{A}_s \mathbf{1} / \|\mathbf{A}_s \mathbf{1}\|$, which is exactly the Rake-DS beamformer. This is analogous to the non-raking DS beamformer (3.8).

Max-UDR Raking

Finally, it is interesting to investigate what happens if we choose the weights that optimize the perceptually motivated UDR [19, 77]. The UDR expresses the fact that adding early reflections (up to 50 ms in the RIR) is as good as adding the energy to the direct sound in terms of speech intelligibility. The useful signal is a *coherent* sum of the direct and early reflected speech energy, so that

$$\text{UDR} = \frac{\mathbb{E} \sum_{k=0}^K |\mathbf{w}^H \mathbf{a}(s_k) x|^2}{\mathbb{E} \left| \sum_{k=0}^{K'} \mathbf{w}^H \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right|^2}, \quad (3.12)$$

In applications K is rarely large enough to cover all the reflections occurring within 50 ms, simply because it is too optimistic to assume we know all the corresponding image sources. Therefore, (3.12) typically underestimates the UDR.

Alas, because (3.12) is specified in the frequency domain, it is challenging to control whether the reflections in the numerator arrive before or after the direct sound. Nevertheless, it is interesting to analyze it as it provides a basis for future work on time-domain raking formulations, and a meaningful evaluation of the raking algorithms presented here.

To compute the Rake-Max-UDR weights, we solve the following program

$$\text{maximize}_{\mathbf{w} \in \mathbb{C}^M} \frac{\mathbb{E} \sum_{k=0}^K |\mathbf{w}^H \mathbf{a}(s_k) x|^2}{\mathbb{E} \left| \mathbf{w}^H \sum_{k=0}^{K'} \mathbf{a}(q_k) z + \mathbf{w}^H \mathbf{n} \right|^2}.$$

This amounts to maximizing a particular generalized Rayleigh quotient,

$$\frac{\mathbf{w}^H \mathbf{A}_s \mathbf{A}_s^H \mathbf{w}}{\mathbf{w}^H \mathbf{K}_{nq} \mathbf{w}}. \quad (3.13)$$

Assuming that \mathbf{K}_{nq} has a Cholesky decomposition as $\mathbf{K}_{nq} = \mathbf{C}^H \mathbf{C}$ we can rewrite the quotient (3.13) as

$$\frac{\tilde{\mathbf{w}}^H (\mathbf{C}^{-1})^H \mathbf{A}_s \mathbf{A}_s^H \mathbf{C}^{-1} \tilde{\mathbf{w}}}{\tilde{\mathbf{w}}^H \tilde{\mathbf{w}}},$$

where $\tilde{\mathbf{w}} \stackrel{\text{def}}{=} \mathbf{C} \mathbf{w}$. The maximum of this expression is

$$\lambda_{\max}((\mathbf{C}^{-1})^H \mathbf{A}_s \mathbf{A}_s^H \mathbf{C}^{-1}),$$

where $\lambda_{\max}(\cdot)$ denotes the largest eigenvalue of the matrix in the argument. The maximum is achieved by the corresponding eigenvector $\tilde{\mathbf{w}}_{\max}$. Then the optimal weights are given as

$$\mathbf{w}_{\text{R-UDR}} = \mathbf{C}^{-1} \tilde{\mathbf{w}}_{\max}.$$

3.4.3 Expected SINR Gain from Raking

Intuitively, if we have multiple sources of the desired signal scattered in space, and we account for it in the design, we should do at least as well as when we ignore the image sources. Let us see how large the gain can be for the Rake-Max-SINR beamformer. We have that

$$\text{SINR} = \sigma_x^2 (\mathbf{A}_s \mathbf{1})^H \mathbf{K}_{nq}^{-1} (\mathbf{A}_s \mathbf{1}).$$

Intuitively, the larger the norm of $\mathbf{A}_s \mathbf{1}$, the better the SINR (as \mathbf{K}_{nq} is positive). To explicitly see if there is any gain in using the acoustic rake receiver, we should compare the standard Max-SINR beamformer with the Rake-Max-SINR, *e.g.*, we should evaluate

$$\frac{(\sum_k \mathbf{a}(\mathbf{s}_k))^H \mathbf{K}_{nq}^{-1} (\sum_k \mathbf{a}(\mathbf{s}_k))}{\mathbf{a}(\mathbf{s}_0)^H \mathbf{K}_{nq}^{-1} \mathbf{a}(\mathbf{s}_0)}. \quad (3.14)$$

One possible interpretation of (3.14) is that we ask whether the steering vectors $\mathbf{a}(\mathbf{s}_k)$ sum coherently or they cancel out.

To answer this, assume that \mathbf{s}_k , $0 \leq k \leq K$ are the desired sources (true and image), and let $\beta \stackrel{\text{def}}{=} \sum_{k=1}^K (\alpha_k / \alpha_0)^2$, where α_k is the strength of the source \mathbf{s}_k received by the array. Then

$$\mathbb{E} \left(\left\| \sum_{k=0}^K \mathbf{a}(\mathbf{s}_k) \right\|^2 \right) \approx (1 + \beta) \mathbb{E}(\|\mathbf{a}(\mathbf{s}_0)\|^2), \quad (3.15)$$

that is, we can expect an increase in the output SINR approximately by a factor of $(1 + \beta)$ when using the Rake-Max-SINR beamformer. This statement is made precise in Theorem 3.1 in the Appendix. It holds when \mathbf{K}_{nq} has eigenvalues of similar magnitude, which is typically not the case in the presence of interferers. However, we show in Section 3.4.4 that with interferers present, the gains actually increase.

A couple of remarks are in order:

1. This result is in expectation; it says that on average, the SINR will increase by a factor of $(1 + \beta)$. In the worst case, the steering vectors $\mathbf{a}(\mathbf{s}_k)$ can even cancel out so that the SINR decreases. But the numerical experiments suggest that this is very rare in practice, and we can on the other hand observe large gains.

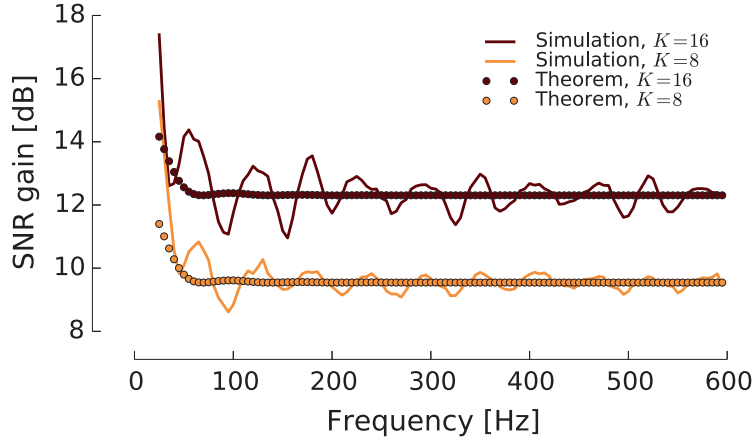


Figure 3.6: Comparison of the simulated SNR gains and the theoretical prediction from Theorem 3.1 for $K = 8$, and $K = 16$. The theoretical prediction of the gain is $10 \log_{10}(8 + 1) \approx 9.54$ for $K = 8$, and $10 \log_{10}(16 + 1) \approx 12.30$ for $K = 16$.

2. We see that summing the phasors in $a_m(\mathbf{s}_k)$ behaves as a two-dimensional random walk. It is known that the root-mean-square distance of a 2D random walk from the origin after n steps is \sqrt{n} [85].
3. Due to the far-field assumption in Theorem 3.1, the attenuations α_k are assumed to be independent of the microphones; in reality they do depend on the source locations. However, they also depend on a number of additional factors, for example wall attenuations and radiation patterns of the sources. Therefore, for simplicity, we consider them to be independent. One can verify that this assumption does not change the described trend.

It is reassuring to observe the behavior suggested by (3.15) in practice. Figure 3.6 shows the comparison of the prediction by Theorem 3.1 with the SNR gains observed in simulated rooms. In this case, we are comparing the pure SNR gain for white noise, without interferers. To generate Figure 3.6, we randomized the location of the source inside the rectangular room. For simplicity we fixed the signal power as received by the microphones to the same value for all the image sources, so that the expected gain is $K + 1$ in the linear scale. The curves agree near-perfectly with the prediction of Theorem 3.1.

3.4.4 Numerical Experiments

In this section, we validate the described theoretical results through numerical experiments. First, we analyze the beam patterns produced by the ARR; second, we evaluate the SINR for various beamformers as a function of the number of image sources used in weight computation; and third, we evaluate the PESQ metric [110]. Finally, we show spectrograms that reveal visually the improved interferer and noise suppression achieved by the ARR.

Numerical experiments are conducted using the framework introduced in Section 3.3. We use discrete-time STFT processing with a frame size of $L = 4096$ samples, 50% overlap and zero padding on both sides of the signal by $L/2$, and compute the beamforming in the frequency domain. A real fast Fourier transform of size $2L$ and a Hann window are used in the analysis. By exploiting the conjugate symmetry of the real FFT we only need to compute $L + 1$ beamforming

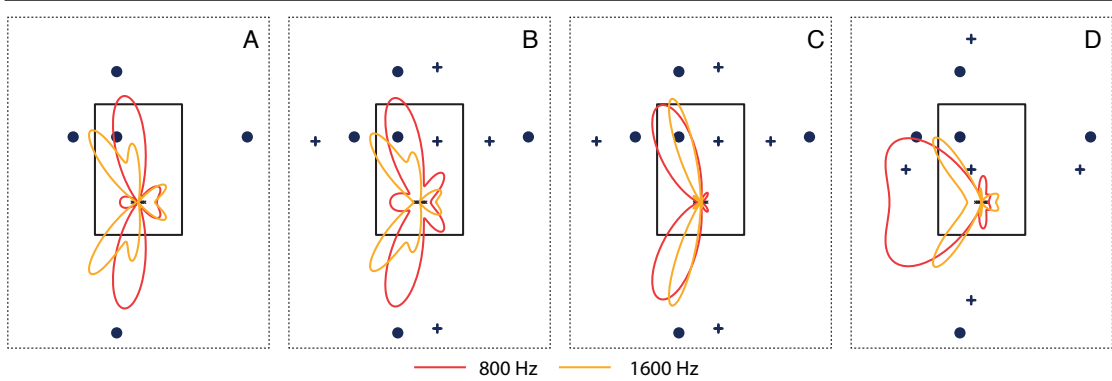


Figure 3.7: Beam patterns in different scenarios. The rectangular room is 4 by 6 metres and contains a source of interest (●) and an interferer (+). The first-order image sources are also displayed. The weight computation of the beamformer includes the direct source and the first order image sources of both desired source and interferer (when applicable). (A) Rake-Max-SINR, no interferer, (B) Rake-Max-SINR, one interferer, (C) Rake-Max-UDR, one interferer, (D) Rake-Max-SINR, interferer is in direct path.

weights, one for every positive frequency bin. The length L is dictated by the length of the beamforming filters in the time-domain and was set empirically so as to avoid any aliasing in the filter responses.

Results

Beampatterns We first inspect the beampatterns produced by the Rake-Max-SINR and Rake-Max-UDR beamformers for different source-interferer placements. We consider a 4 m \times 6 m rectangular room with a source of interest at (1 m, 4.5 m) and a linear microphone array centered at (2 m, 1.5 m), parallel to the x -axis. Spacing between the microphones was set to 8 cm. In Figure 3.7, we show the beampatterns for four different configurations of the source and the interferer. We consider a scenario without an interferer, one with an interferer placed favorably at (2.8 m, 4.3 m), and finally one where the interferer is placed half-way between the desired source and the array, at (1.5 m, 3 m).

The last scenario is the least favorable. Interestingly, we can observe that the Rake-Max-SINR beampattern adjusts by completely ignoring the direct path, and steering the beam towards the echoes of the source of interest. This is validating the intuition that we can “hear behind an interferer by listening for the echoes”. Note that such a pattern cannot be achieved by a beamformer that only takes into account the direct path. We further note that, while the beampatterns only show the magnitude of the beamformer’s response, the phase plays an important role with multiple sources present.

SINR Gains from Raking In the experiments in this subsection, we set the power of the desired source and of the interferer to be equal, $\sigma_x^2 = \sigma_z^2 = 1$. The noise covariance matrix is set to $10^{-3} \cdot \mathbf{I}_M$. We use a circular array of $M = 12$ microphones with a diameter of 30 cm, and randomize the position of the desired source and the interferer inside the room. The resulting curves show median performance out of 20000 runs.

Figure 3.8a shows output SINR for different beamformers. The one-forcing beamformer is left out because it performs poorly in terms of SINR, as predicted in the earlier discussion. Clearly, the Rake-Max-SINR beamformer outperforms all others. The output SINR for beamformers using only the direct path (Max-SINR and DS) remains approximately constant. The UDR is plotted against the number of image sources for various beamformers in Figure 3.8b. Even though the Rake-Max-UDR beamformer performs well in terms of the two measures, its output is perceptually displeasing due to audible pre-echoes; in informal listening tests, the Rake-Max-SINR beamformer did not produce such artifacts. It is interesting to note that the Rake-Max-SINR also performs well in terms of the UDR. Similar SINR gains to those in Figure 3.8a are observed in Figure 3.8c over a range of frequencies. It is therefore justified to extrapolate the results at one frequency in Figure 3.8a to the wideband SINR.

Evaluation of Speech Quality We complement the informal listening tests and the evaluation of SINR and UDR with extensive simulations to assess the improvement in speech quality achieved by acoustic raking. We simulate a room with two sources—a desired source and an interferer—and compare the outputs of the Rake-DS, Rake-Max-SINR, and Rake-Max-UDR as a function of the number of image sources used to design the beamformers.

The same number of image sources is used for the desired source and the interferer ($K = K'$). The performance metric used is the PESQ [110]. In particular, we use the reference implementation described by the ITU P.862 Amendment 2 [67]. PESQ compares the reference signal with the degraded signal and predicts the perceptual quality of the latter as it would be measured by the mean opinion score (MOS) value, on a scale from 1 to 4.5.

We consider the same room and microphone array setting as before (see Figure 3.7A). The desired and the interfering sources are placed uniformly at random in a rectangular area with lower left corner at (1 m, 2.5 m) and upper right corner at (3 m, 5 m). To limit the experimental variation, the speech samples attributed to the sources are fixed throughout the simulation. The two sources start reproducing speech at the same time and approximately overlap for the total duration of the speech samples. The signals are normalized to have the same power at the source, and we add white Gaussian noise to the microphone signals, with power chosen so that the SNR of the direct sound for the desired source is 20 dB at the center of the microphone array. All signals are high-pass filtered with a cut-off frequency of 300 Hz. The reference for all PESQ results is the direct path of the desired source as measured at the center of the array (2 m, 1.5 m).

The median PESQ measure of 10000 Monte Carlo runs, given in raw MOS, is shown in Figure 3.8d. The median PESQ of the degraded signal measured at the center of the array before processing was found to be 1.6 raw MOS. When only the direct sound is used (*i.e.*, $K = 0$), all three beamformers yield the same improvement of about 0.2 raw MOS. We observe that Rake-DS is slightly outperforms the other beamformers. Using any number of echoes in addition to the direct sound results in larger MOS for all beamformers. When more than one image source is used, the Rake-Max-SINR beamformer always yields the largest MOS, with up to 0.5 MOS gain when using 10 images sources.

It is worth mentioning that in the beamformer design, we do not assume that we know the spectrum of the source or the interferer—we design as if it was flat. Thus the interferer acts as a strong source of colored, spatially correlated, non-stationary noise, spectrally mismatched with the designed beamformer. There is another source of model mismatch: while the RIRs were computed using hundreds of image sources, we use only up to ten to design the beamformers.

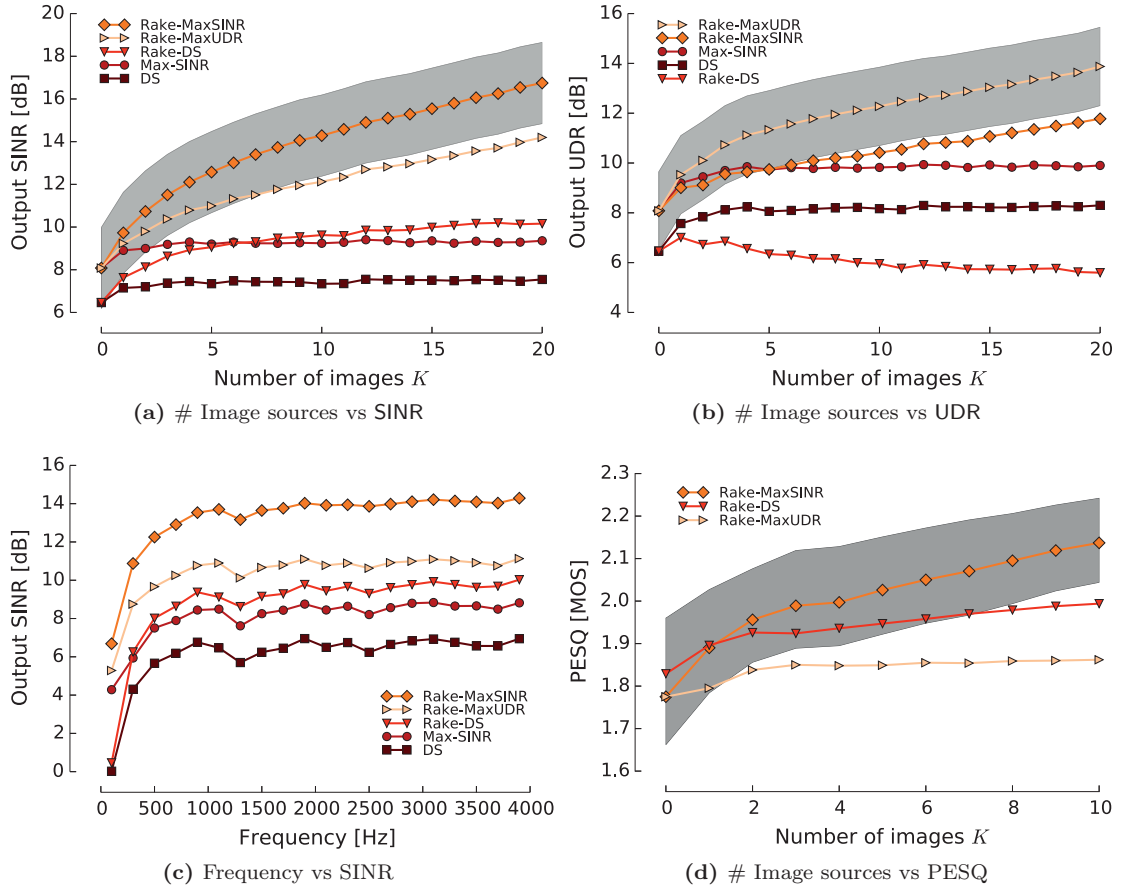


Figure 3.8: Median output (a) SINR and (b) UDR plotted against the number of image sources used in the design for different beamformers, at a frequency $f = 1$ kHz. (c) Output SINR as a function of frequency for different beamformers, $K = K' = 10$, and averaged in the dB domain. (d) Median perceptual quality in MOS, evaluated using PESQ, as a function of the number of image sources used K . The lower limit of the ordinates is set to the median MOS of the degraded signal before processing, as measured at center of the array. The number of Monte Carlo runs is 20000 in (a), (b), and (c), and 10000 in (d). The shaded area contains 50% of the run of Rake-Max-SINR in (a) and (d), and Rake-Max-UDR in (b).

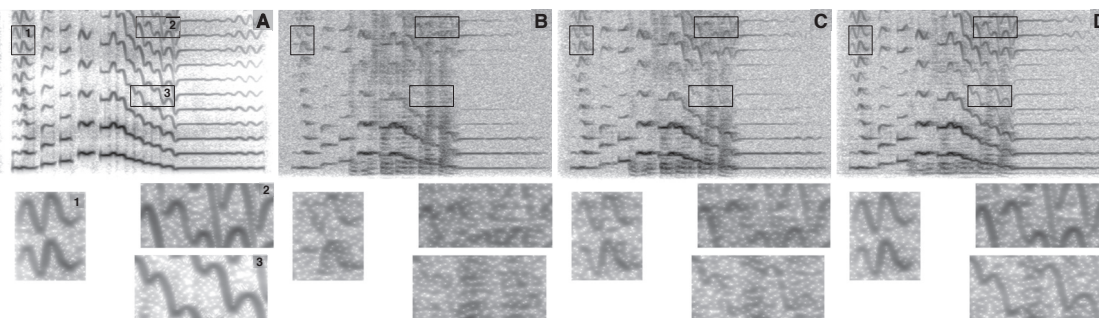


Figure 3.9: Comparison of the conventional Max-SINR and Rake-Max-SINR beamformer on a real speech sample. Spectrograms of (A) clean signal of interest, (B) signal corrupted by an interferer and additive white Gaussian noise at the microphone input, outputs of (C) conventional Max-SINR and (D) Rake-Max-SINR beamformers. Time naturally goes from left to right, and frequency increases from zero at the bottom up to $F_s/2$. To highlight the improvement of Rake-Max-SINR over Max-SINR, we blow-up three parts of the spectrograms in the lower part of the figure. The boxes and the corresponding part of the original spectrogram are numbered in (A). The numbering is the same but omitted in the rest of the figure for clarity.

Spectrograms and Sounds Samples Finally, we present the spectrograms for a scenario where we want to focus on a singer in the presence of interfering speech. We consider the same room, source, interferer, and microphone array geometry as in Figure 3.7B.

The source signal is a passage by a female opera singer (Figure 3.9A), with strongly pronounced harmonics; the interfering signal is a male speech extract. The two signals are normalized to have unit maximum amplitude. We add white Gaussian noise to the microphone signals with power such that the SNR of the direct sound of the desired source is 20 dB at the center of the microphone array. All signals are high-pass filtered with a cut-off frequency of 300 Hz. The Rake-Max-SINR beamformer weights are computed using the direct source and three generations of image sources for both the desired sound source (singing) and the interferer (speech).

The output of the conventional Max-SINR beamformer (Figure 3.9C) is compared to that of the Rake-Max-SINR (Figure 3.9D). We can observe from the spectrogram that the Rake-Max-SINR reduces very effectively the power of the interfering signal at all frequencies, but particularly in the mid to high range. This is true even when the interferer overlaps significantly with the desired signal. Informal listening tests confirm that the Rake-Max-SINR maintains high quality of the desired signal while strongly reducing the interference. The Rake-Max-UDR beamformer provides good interference suppression, but it produces audible pre-echoes that render it unsuitable for speech processing applications. The sound clips can be found online along the code.

3.5 Time Domain Formulations

This section introduces time domain formulations of ARR. Formulating the optimization problem directly in the time domain allows to directly put constraints on the impulse response of the filter. These constraints can be used to limit the delay or length of the filter, or ensure the absence of pre-echoes, among other things. We explore several possible designs. Section 3.5.1

extends the notation introduced in Section 3.2 and allows a cleaner formulation the optimization. Then, Section 3.5.2 presents the different formulations which are summarized in Table 3.3. Finally, Section 3.5.3 shows the result of numerical experiments.

3.5.1 Notation

We begin by introducing some notation specific to time domain formulations. Note that some notation is overloaded in this section. We start from the time domain signal equation (3.1) that we rewrite in matrix form

$$\mathbf{y}_m = \sum_{k=0}^K \mathbf{A}_m(\mathbf{s}_k) \mathbf{x} + \sum_{k=0}^{K'} \mathbf{A}_m(\mathbf{q}_k) \mathbf{z} + \mathbf{b}_m$$

where

$$\begin{aligned} \mathbf{y}_m &= [y_m[n], y_m[n-1], \dots, y_m[n-L_g+1]]^\top, \\ \mathbf{x} &= [x[n], x[n-1], \dots, x[n-L+1]]^\top, \\ \mathbf{z} &= [z[n], z[n-1], \dots, z[n-L+1]]^\top, \\ \mathbf{b}_m &= [b_m[n], b_m[n-1], \dots, b_m[n-L_g+1]]^\top. \end{aligned}$$

and $\mathbf{A}_m(\mathbf{s}_k)$ is an $L_g \times L$ convolution matrix corresponding to the propagation from the k th image source to the m th microphone, with L_g the size of the beamforming filter, $L = L_h + L_g - 1$. It is a Toeplitz matrix whose first row is $a_m(\mathbf{s}_k, n)$, $n = 0, \dots, L_h - 1$, the discretized impulse response from (3.4), padded with $L_g - 1$ zeros, and first column is $a_m(\mathbf{s}_k, 0)$ followed by $L_g - 1$ zeros, namely

$$\mathbf{A}_m(\mathbf{s}_k) = \begin{bmatrix} a_m(\mathbf{s}_k, 0) & \cdots & a_m(\mathbf{s}_k, L_h - 1) & & \mathbf{0} \\ & \ddots & & \ddots & \\ \mathbf{0} & & a_m(\mathbf{s}_k, 0) & \cdots & a_m(\mathbf{s}_k, L_h - 1) \end{bmatrix}.$$

Stacking all the vectors and matrices, indexed by m into a single vector and matrix, and dropping the index, we obtain the following compact form

$$\mathbf{y} = \mathbf{H}_s \mathbf{x} + \mathbf{H}_q \mathbf{z} + \mathbf{b},$$

where $\mathbf{H}_s = \sum_{k=0}^K \mathbf{A}(\mathbf{s}_k)$ and $\mathbf{H}_q = \sum_{k=0}^{K'} \mathbf{A}(\mathbf{q}_k)$. The m th beamforming filter is $\mathbf{g}_m = [g_m[0], \dots, g_m[L_g - 1]]^\top$ and its output at time n can be written as the inner product $\mathbf{g}_m^\top \mathbf{y}_m$. Stacking all M filters in a vector, $\mathbf{g} = [\mathbf{g}_0^\top \cdots \mathbf{g}_{M-1}^\top]^\top$, the sum of all filter outputs is conveniently computed as $\mathbf{g}^\top \mathbf{y}$. The responses of the beamformer towards the desired source and interferer are

$$\mathbf{u}_s = \mathbf{H}_s^\top \mathbf{g}, \quad \mathbf{u}_q = \mathbf{H}_q^\top \mathbf{g},$$

respectively. Finally, the letter τ is used to denote the delay (in samples) of the beamformer.

Table 3.3: Summary of time-domain beamformers.

Acronym	Description	Beamforming Weights
Rake-MVDR	$\min \mathbb{E} \mathbf{g}^\top \mathbf{y} ^2$, s.t. $\mathbf{g}^\top \mathbf{h}_\tau = 1$	$\mathbf{g}_{\text{R-MVDR}} = \mathbf{R}_{yy}^{-1} \mathbf{h}_\tau (\mathbf{h}_\tau^\top \mathbf{R}_{yy} \mathbf{h}_\tau)^{-1}$
Rake-Perceptual	$\min \mathbb{E} \mathbf{g}^\top (\mathbf{H}_q \mathbf{z} + \mathbf{g}) ^2$, s.t. $\mathbf{g}^\top \widehat{\mathbf{H}}_s = \delta_\tau^\top$	$\mathbf{g}_{\text{R-P}} = \mathbf{K}_{nq}^{-1} \widehat{\mathbf{H}}_s (\widehat{\mathbf{H}}_s^\top \mathbf{K}_{nq}^{-1} \widehat{\mathbf{H}}_s)^{-1} \delta_\tau$
Rake-Max-SINR	$\max \mathbb{E} \mathbf{g}^\top \mathbf{H}_s \mathbf{x} ^2 / \mathbb{E} \mathbf{g}^\top (\mathbf{H}_q \mathbf{z} + \mathbf{b}) ^2$	Solve e.v. problem $\mathbf{K}_x \mathbf{g} = \lambda \mathbf{K}_{nq} \mathbf{g}$

3.5.2 Time Domain Acoustic Rake Receivers

Minimum Variance Distortionless Response Rake Beamformer

A time-domain flavour of the classic Capon minimum variance distortionless response (MVDR) beamformer [24] is given by²,

$$\underset{\mathbf{g}}{\text{minimize}} \mathbb{E} |\mathbf{g}^\top \mathbf{y}|^2 \quad \text{subject to } \mathbf{g}^\top \mathbf{h}_\tau = 1,$$

where \mathbf{h}_τ is the τ th column of \mathbf{H}_s . The constraint forces unit response towards the desired source. The value of τ determines the delay of the beamformer and should be larger than the latest arriving echoes that we would like to rake. The objective can be developed into $\mathbb{E} |\mathbf{g}^\top \mathbf{y}|^2 = \mathbf{g}^\top \mathbf{R}_{yy} \mathbf{g}$ where \mathbf{R}_{yy} is the covariance matrix of \mathbf{y} ,

$$\mathbf{R}_{yy} = \mathbf{H}_s \mathbf{R}_{xx} \mathbf{H}_s^\top + \mathbf{H}_q \mathbf{R}_{zz} \mathbf{H}_q^\top + \mathbf{R}_{bb},$$

where in turn \mathbf{R}_{xx} , \mathbf{R}_{zz} , and \mathbf{R}_{bb} are the covariance matrices of \mathbf{x} , \mathbf{z} , and the noise. The optimization problem becomes

$$\underset{\mathbf{g}}{\text{minimize}} \mathbf{g}^\top \mathbf{R}_{yy} \mathbf{g} \quad \text{subject to } \mathbf{g}^\top \mathbf{h}_\tau = 1 \quad (3.16)$$

and is solved for

$$\mathbf{g}_{\text{R-MVDR}} = \mathbf{R}_{yy}^{-1} \mathbf{h}_\tau (\mathbf{h}_\tau^\top \mathbf{R}_{yy}^{-1} \mathbf{h}_\tau)^{-1}.$$

Assuming samples from both sources are independent and identically normally distributed, and that the noise is AWGN, i.e. $\mathbf{R}_{xx} = \sigma_x^2 \mathbf{I}$, $\mathbf{R}_{zz} = \sigma_z^2 \mathbf{I}$, and $\mathbf{R}_{bb} = \sigma_n^2 \mathbf{I}$, (3.16) can be rewritten

$$\begin{aligned} \underset{\mathbf{g}}{\text{minimize}} \quad & \sigma_x^2 \|\mathbf{u}_s\|^2 + \sigma_z^2 \|\mathbf{u}_q\|^2 + \sigma_n^2 \|\mathbf{g}\|^2 \\ \text{subject to} \quad & u_s[\tau] = 1, \quad \mathbf{u}_s = \mathbf{H}_s^\top \mathbf{g}, \quad \mathbf{u}_q = \mathbf{H}_q^\top \mathbf{g}, \end{aligned}$$

where $u_s[\tau]$ is the τ th element of \mathbf{u}_s . From this form, it is clear that the optimal beamformer will balance distortionless response towards desired source, interference cancellation, and noise suppression. For a fixed L_g , adding more image sources will increase L_h and consequently the number of constraints in the optimization problem. Reducing so the feasible set might decrease the noise suppression performance of the beamformer.

Finally, using our geometric interpretation it is possible to know precisely how many echoes can be exploited. Because the response is distortionless, the output of the beamformer should

²Although the response is not truly distortionless, we follow the definition of the time-domain MVDR beamformer of Benesty et al [13].

be the desired source with a delay τ (not considering model inaccuracies). This means that only echoes arriving within the time τ of the direct sound can be used to improve the source power. Knowing the propagation speed of sound translates into a geometrical criterion on which image sources can be included. All image sources within distance $\|\mathbf{s}_0 - \mathbf{r}_m\| + c\tau/F_s$ of the microphone array can be used, c being the speed of sound, and F_s the sampling frequency.

Perceptually Motivated Rake Beamformer

Psychoacoustics studies show that early echoes contribute to perceived power, and speech intelligibility. Lochner and Burger [77] describe precisely how much reverberation is perceptually beneficial. As determined for speech signals, echoes arriving within 30 ms of the direct sound are fully integrated, while those arriving within 95 ms are still partially integrated. Echoes arriving later than 35 ms are noticeable.

In regard of these results, we can partially relax the distortionless requirement. We define the perceptually motivated rake beamformer with the following four criteria.

- Minimize the interference and noise power.
- Zero response before τ (i.e. no pre-echoes).
- Unit response at τ .
- Zero response after $\tau + \kappa$, where $\kappa \sim 35$ ms.

The optimal such beamformer is found by the quadratic program,

$$\underset{\mathbf{g}}{\text{minimize}} \mathbf{g}^T \mathbf{K}_{nq} \mathbf{g} \quad \text{subject to} \quad \mathbf{g}^T \widehat{\mathbf{H}}_s = \delta_\tau^\top,$$

where $\mathbf{K}_{nq} = \mathbf{H}_q \mathbf{R}_{zz} \mathbf{H}_q^\top + \mathbf{R}_{bb}$, the matrix $\widehat{\mathbf{H}}_s$ contains the columns 1 to τ and $\kappa + 1$ to L of \mathbf{H}_s , and δ_τ is the vector with a one at position τ and all other entries zero. Note that an alternative formulation including all zero forcing constraints directly in the objective exists. The solution of this program is

$$\mathbf{g}_{\text{R-P}} = \mathbf{K}_{nq}^{-1} \widehat{\mathbf{H}}_s (\widehat{\mathbf{H}}_s^\top \mathbf{K}_{nq}^{-1} \widehat{\mathbf{H}}_s)^{-1} \delta_\tau.$$

A similar criterion as for Rake MVDR beamformer applies as to which image sources can be used constructively. Thanking to the relaxation, image sources up to distance $\|\mathbf{s}_0 - \mathbf{r}_m\| + c(\tau + \kappa)/F_s$ can be included in the optimization.

Maximum SINR Rake Beamformer

The signal to interference and noise ratio (SINR) is defined as

$$\text{SINR} = \frac{\mathbb{E}|\mathbf{g}^\top \mathbf{H}_s \mathbf{x}|^2}{\mathbb{E}|\mathbf{g}^\top (\mathbf{H}_q \mathbf{z} + \mathbf{b})|^2} = \frac{\mathbf{g}^\top \mathbf{K}_x \mathbf{g}}{\mathbf{g}^\top \mathbf{K}_{nq} \mathbf{g}}, \quad (3.17)$$

where $\mathbf{K}_x = \mathbf{H}_s \mathbf{R}_{xx} \mathbf{H}_s^\top$. This quantity can be optimized directly by solving the generalized eigenvalue problem $\mathbf{K}_x \mathbf{g} = \lambda \mathbf{K}_{nq} \mathbf{g}$, and the maximizer is given by the generalized eigenvector corresponding to the largest generalized eigenvalue. This will however not yield a practical beamformer. Because no constraint is imposed on the response towards the desired source, its signal can be arbitrarily distorted. Nevertheless, this gives an upper bound on achievable SINR.

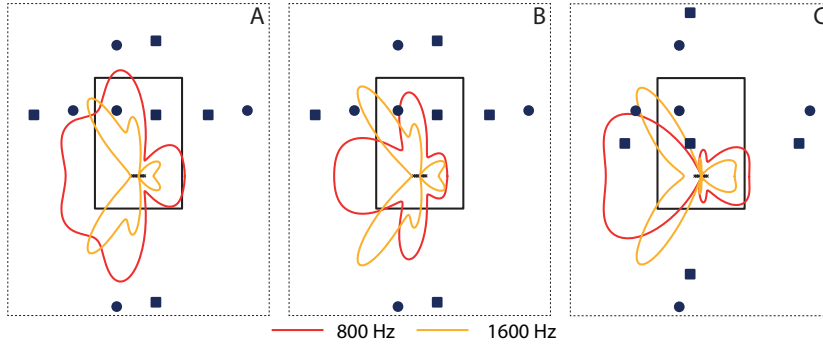


Figure 3.10: Beam patterns of (A) Rake MVDR, and (B), (C) Rake Perceptual, in a 4×6 m room containing the desired source (●) and an interferer (■). In (C), the interferer is in the direct path of the desired source. First order image sources are also displayed. The darker/red and light/yellow lines are for 800 Hz and 1600 Hz, respectively.

3.5.3 Numerical Experiments

In this section, we assess the performance of the three rake beamformers described. First, we inspect the beam patterns obtained. Then, the gain of using additional sources is evaluated in terms of output SINR. We use the same simulation setup as Section 3.4.4. For sound propagation simulation we use up to 10th order reflections (220 image sources). The sampling frequency is 8 kHz. Samples from both sources are assumed to be zero-mean independent and identically distributed and the noise is AWGN so that

$$\mathbf{R}_{xx} = \sigma_x^2 \mathbf{I}, \quad \mathbf{R}_{zz} = \sigma_z^2 \mathbf{I}, \quad \mathbf{R}_{bb} = \sigma_n^2 \mathbf{I},$$

where \mathbf{I} is the identity matrix and $\sigma_x^2 = \sigma_z^2 = 1$.

Results

Beam patterns We consider a 4 by 6 m room with a source of interest at (1, 4.5) and a linear array of eight microphones equally spaced by 8 cm, parallel to the x -axis and centered at (2, 1.5), the origin being the lower left corner of the room. The beamforming filters length is 50 ms ($L_g = 400$ at 8 kHz) with a delay of 20 ms. The noise variance at the microphones is fixed at $\sigma_n^2 = 10^{-7}$. Beam patterns for both Rake MVDR and Rake Perceptual with an interferer placed at (2.8, 4.3) are shown for 800 Hz and 1600 Hz in Figure 3.10. The diagram in the figure shows the beam patterns for Rake Perceptual when the interferer is placed in the direct path of the desired source at (1.5, 3). We observe that in that case, the beamformer completely ignores the direct sound and focuses on the reflections. Such a scenario could not be handled by a beamformer only considering the direct sound.

SINR gain from raking The SINR gain from raking is investigated through Monte-Carlo simulation. We consider the same room and beamforming filters length as in Section 3.5.3, but pick source and interferer positions uniformly at random. At each run, the SINR according to (3.17) is computed for Rake MVDR, Rake Perceptual, and Rake MaxSINR. Even though Rake MaxSINR is not practical, it gives an upper bound on the SINR gain that can be expected. The

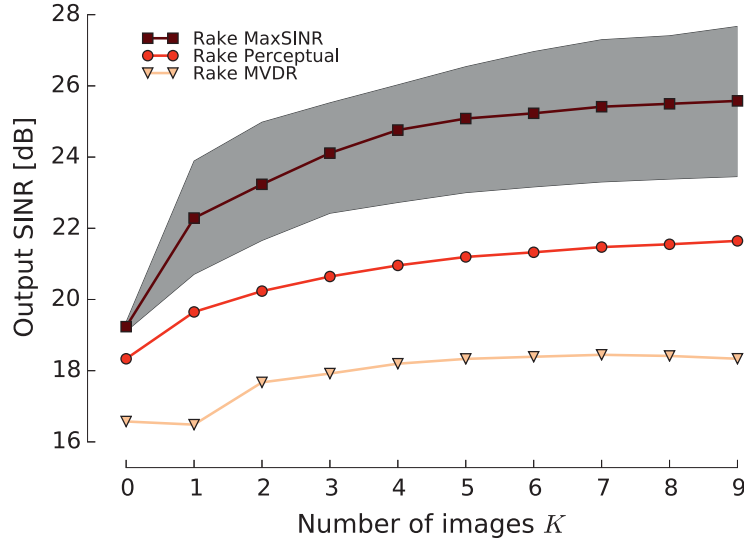


Figure 3.11: Median output SINR computed according to (3.17) against the number of image sources K used in the optimization. The same number of image sources is used for the desired source and the interferer. The ambient noise SNR is fixed to 10 dB with respect to the direct path of the desired source and the center of the microphone array. The grey area contains 50% of the Rake MaxSINR outcomes.

same number of image sources $K = K' = 0, \dots, 9$ is used for the source and the interferer. The noise variance is fixed so that the SNR of the direct path of the desired source is 10 dB at the center of the array or $\sigma_n^2 = 10^{-1}(4\pi \|\mathbf{s}_0 - \bar{\mathbf{r}}\|)^{-2}$ where $\bar{\mathbf{r}} = M^{-1} \sum_{m=0}^{M-1} \mathbf{r}_m$ is the center of the array. The beamforming filters length is fixed to 30 ms (i.e. $L_g = 240$) and the delay is 20 ms.

The outcome of the simulation is depicted in Figure 3.11. Each point is the result of 10000 outcomes. For every beamformer considered, adding more sources results in a net increase in SINR. Adding just the 1st order reflections, or 5 sources, rakes in 1.76 dB and 2.85 dB improvement in SINR for Rake MVDR and Rake Perceptual, respectively. Rake MaxSINR shows that at most 5.85 dB improvement can be expected. We also observe that the extra degrees of freedom of Rake Perceptual are very beneficial as it is consistently 2 to 3 dB above Rake MVDR when image sources are used.

3.6 Finding and Tracking the Echoes

Thus far we assumed that the locations of the image sources are known. In this section we briefly describe some methods to localize them when they are *a priori* unknown. We assume that we can localize the true source, or at least one image source. Combined with the knowledge of the room geometry, this suffices to find the locations of other image sources [94].

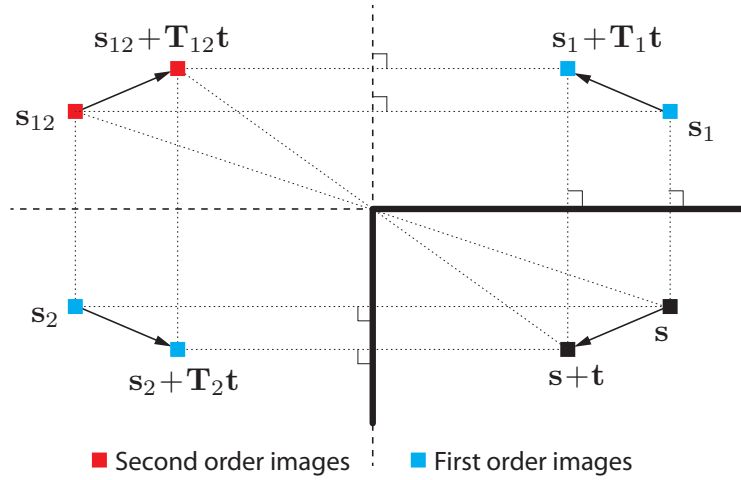


Figure 3.12: Illustration of image source tracking in rectangular geometries.

3.6.1 Known Room Geometry

In many cases, for example for fixed deployments, the room geometry is known. This knowledge could be obtained at the time of the deployment, or from blueprints. In most indoor environments, we encounter a large number of planar reflectors. These reflectors correspond to image sources. With reference to Figure 3.12, we can easily compute the image source locations [4] (we note that the image source model is not limited to right angle geometries [17]).

Suppose that the real source is located at \mathbf{s} . Then the image source with respect to wall i is computed as,

$$im_i(\mathbf{s}) = \mathbf{s} + 2\langle \mathbf{p}_i - \mathbf{s}, \mathbf{n}_i \rangle \mathbf{n}_i,$$

where i indexes the wall, \mathbf{n}_i is the outward normal associated with the i th wall, and \mathbf{p}_i is any point belonging to the i th wall. Analogously, we compute image sources corresponding to higher-order reflections,

$$im_j(im_i(\mathbf{s})) = im_i(\mathbf{s}) + 2\langle \mathbf{p}_j - im_i(\mathbf{s}), \mathbf{n}_j \rangle \mathbf{n}_j.$$

The above expressions are valid regardless of the dimensionality, concretely in 2D and 3D.

3.6.2 Acoustic Geometry Estimation

When the room geometry is not known, it is possible to estimate it using the same array that we use for beamforming. Recently a number of different methods appeared in the literature that propose to use sound to estimate the shape of a room. For example, in [108] the authors use a dictionary of wall impulse responses recorded with a particular array. In [6] the authors use tools from projective geometry together with the Hough transform to estimate the room geometry. In [38] the authors derive an *echo sorting* mechanism that finds the image sources, from which the room geometry is then derived.

3.6.3 Without Estimating the Room Geometry

To design an ARR, we do not really need to know how the room looks like; we only need to know where the major echoes are coming from. The initial estimation of echo locations can be done using traditional localization methods based on time, or time difference, of arrival [11], or alternatively direction of arrival (DOA) finding algorithms. Chapter 4 introduces a DOA algorithm suitable for this task. After the initial localization phase, it is possible to track the movement of image sources by tracking the true source.

We propose a tracking rule that leverages the knowledge of the displacement of the true source. Again with reference to Figure 3.12, we can state the following simple proposition.

Proposition 3.1

Suppose that the room has only right angles so that the walls are parallel with the coordinate axes. Let the source move from \mathbf{s} to $\mathbf{s} + \mathbf{t}$. Then any image source \mathbf{s}_k , moves to a point given by

$$\mathbf{s}_k + \mathbf{T}\mathbf{t},$$

where $\mathbf{T} = \text{diag}(\pm 1, \mp 1)$ for odd generations, and $\mathbf{T} = \pm \mathbf{I}_2$ for even generations.

Proof.

The proof follows directly from the figure. The displacement of the image source is the same as the displacement of the true source, passed through a series of reflections. Reflection matrices are diagonal matrices with ± 1 on the diagonal, and determinant equal to -1 , hence the result. \square

The usefulness of this proposition is that it gives us a tool to track the image sources even when we do not know the room geometry (as long as it has right angles). A possible use scenario is to start with a calibration procedure with a controlled source, and perform the echo sorting to find multiple image sources. Then if possible, we assign to each image source a generation (this is in fact a by-product of echo sorting), or we try different hypotheses using Proposition 3.1, and choose the one that maximizes the output SINR.

3.7 Conclusion

We investigated the concept of acoustic rake receivers—beamformers that use echoes. Unlike earlier related work, we presented optimal formulations that outperform the delay-and-sum style approaches by a large margin. This is especially true in the presence of interferers that might make it impractical to rely solely on the direct sound. We can show how using echoes improves the SINR in expectation. Comprehensive numerical experiments confirm this theoretical finding.

We presented both concise and efficient frequency domain formulations, as well as time-domain formulations that allow a greater control of the impulse response of the beamforming filters. We also introduced notions from psychoacoustics into the design of beamformers leading to relaxed formulations or alternative figure of merits for optimization.

Beyond objective measures such as SINR, we demonstrated that ARRs improve subject quality, as predicted by PESQ, proportionally to the number of image sources included. This improvement was further confirmed by informal listening tests. A particularly striking example is when the interferer sits between the desired source and the receiver. In that case, the ARR simply listens to the echoes.

Perhaps the most important aspect of ongoing work is the design of robust formulations of ARRs. This may involve various heuristics, as well as combinatorial optimization due to the discrete nature of image sources. We expect that the raking beamformers described in this chapter inherit the robustness properties of their classic counterparts. For example, the Rake-DS beamformer is likely to be more robust to array calibration errors than the Rake-Max-SINR beamformer. Furthermore, we expect that taking the image source perspective makes various ARRs more robust to errors in source locations than the schemes that assume the knowledge of the RIR.

3.A Theorem 3.1

We note that the theorem is stated for a linear array, but the described behavior is universal.

Theorem 3.1

Assume that there are $K + 1$ sources located at $\mathbf{s}_k = r_k [\cos \theta_k \ \sin \theta_k]^\top$ where $\theta_k \sim \mathcal{U}(0, 2\pi)$ and $r_k \sim \mathcal{U}(a, b)$ are all independent, for some $0 < a < b$ such that the far-field assumption holds. Let \mathbf{A}_s collect the corresponding steering vectors for a uniform linear microphone array. Then $\mathbb{E} \|\mathbf{A}_s \mathbf{1}\|^2 \geq (1 + \beta) \mathbb{E} \|\mathbf{a}(\mathbf{s}_0)\|^2$, where $\beta = \sum_{k=1}^K (\alpha_k / \alpha_0)^2$, and α_k are attenuations of the steering vectors, assumed independent from the source locations. In fact, $\mathbb{E} (\|\mathbf{A}_s \mathbf{1}\|^2) = (1 + \beta) \mathbb{E} (\|\mathbf{a}(\mathbf{s}_0)\|^2) + O(1/\Omega^3)$.

Proof.

Thanks to the far-field assumption, we can decompose the steering vector into a factor due to the array, and a phase factor due to different distances of different image sources. We have that

$$a_m = (\mathbf{A}_s \mathbf{1})_m = \sum_{k=0}^K \alpha_k e^{-j\kappa m d \sin \theta_k} e^{-j\Omega \delta_k / c},$$

where d is the microphone spacing and $\kappa \stackrel{\text{def}}{=} \Omega/c$. Without loss of generality we assume that $\delta_k \sim \mathcal{U}(a, b)$. We can further write

$$\begin{aligned} \mathbb{E} |a_m|^2 &= \mathbb{E} \left[\left(\sum_{k=0}^K \alpha_k e^{-j\kappa m d \sin \theta_k} e^{-j\kappa \delta_k} \right) \right. \\ &\quad \left. \times \left(\sum_{\ell=0}^K \alpha_\ell e^{j\kappa m d \sin \theta_\ell} e^{j\kappa \delta_\ell} \right) \right] \\ &= \sum_{k=0}^K \alpha_k^2 + \sum_{k \neq \ell=0}^K \alpha_k \alpha_\ell \mathbb{E} \left[e^{j\kappa m d (\sin \theta_\ell - \sin \theta_k)} e^{j\kappa (\delta_\ell - \delta_k)} \right]. \end{aligned} \tag{3.18}$$

Invoking the independence for $k \neq \ell$, we compute the above expectation as

$$\begin{aligned} \mathbb{E} \left[e^{j\kappa m d (\sin \theta_\ell - \sin \theta_k)} e^{j\kappa (\delta_\ell - \delta_k)} \right] \\ = \frac{2J_0^2(m d \kappa) [1 - \cos(\Delta \kappa)]}{(\Delta \kappa)^2}, \end{aligned}$$

where J_0 denotes the Bessel function of the first kind and zeroth order and $\Delta \stackrel{\text{def}}{=} b - a$.

Plugging this back into (3.18), we obtain

$$\mathbb{E} |a_m|^2 = \sum_{k=0}^K \alpha_k^2 \left(1 + C \frac{2J_0^2(md\kappa)[1 - \cos(\Delta\kappa)]}{(\Delta\kappa)^2} \right),$$

where $C = \sum_{k \neq \ell} \alpha_k \alpha_\ell / \sum_k \alpha_k^2$.

Because $|J_0(z)| \leq \sqrt{2/(\pi z)} + O(|z|^{-1})$ ([2], Eq. 9.2.1), we see that the expression in brackets is $1 + O(\Omega^{-3})$. Rewriting

$$\sum_{k=0}^K \alpha_k^2 = \frac{1}{M} \mathbb{E} \|\mathbf{a}(\mathbf{s}_0)\|^2 \left(1 + \sum_{k=1}^K (\alpha_k/\alpha_0)^2 \right)$$

concludes the proof. □

Chapter 4

Estimation: FRI-based Direction of Arrival Finding*

However, this was anything but a regular bee: in fact it was an elephant—as Alice soon found out, though the idea quite took her breath away at first.

Through the Looking Glass
LEWIS CARROLL

4.1 Introduction

Many signal processing algorithms rely on a simple model of the world with a few key parameters. While these algorithms obviously strongly rely on the knowledge of these parameters, they are very rarely available and must be estimated from the data at hand. For example, we showed in Chapter 3 how to design better beamformers if we know where the echoes in a room are coming from. A prerequisite to the proposed acoustic rake receivers is a reliable way of identifying the locations of sound sources and we outlined several possible methods in Section 3.6. This chapter explores in details one of these solutions.

Specifically, we treat the particular problem of estimating the locations of multiple sound sources from measurements taken with multiple microphones. We assume the distance between

*Joint work with Hanjie Pan, Eric Bezzam, Ivan Dokmanić, and Martin Vetterli [98]

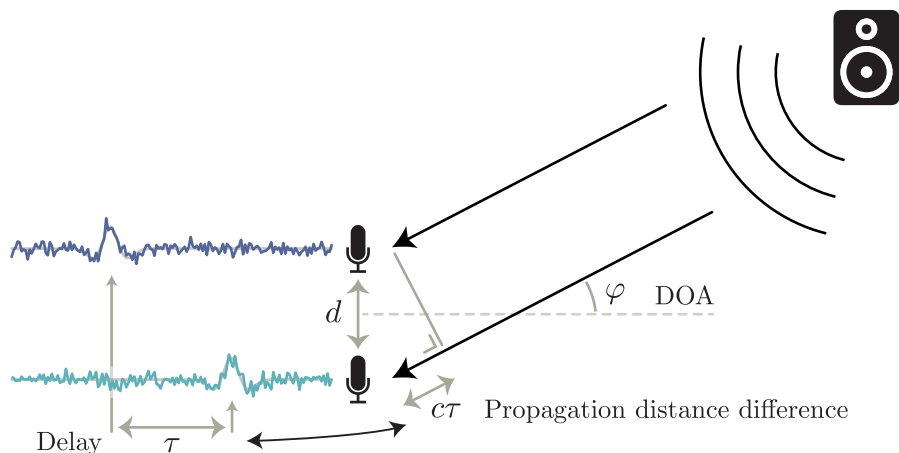


Figure 4.1: DOA 101: Basic inference of DOA with one source and two microphones. The delay between the two microphone signals is directly proportional to the different in propagation distance. Knowing the distance d between the microphones, simple trigonometry yields the DOA φ according to (4.1).

sources and microphones is much larger than between microphones. This is the so-called *far-field* assumption. If it holds, the wave fronts at all microphone locations are near parallel to each other and we can approximate the source signal as a plane wave. A consequence of this approximation is that all notion of distance of the source to the microphones is lost and only *direction of arrival* (DOA) can be reconstructed.

Much like beamforming, DOA finding algorithms exploit the phase of the signal. In beamforming, given the location of a source, we would manipulate the phase of the microphone signals to induce constructive interference in that direction (as illustrated in Figure 3.1). For a single source and two microphones, DOA works much in reverse as shown in Figure 4.1. Both microphones record the same signal with a small delay θ due to propagation time. Knowing the distance d between the microphones, this delay can be geometrically linked to the DOA φ of the source

$$\varphi = \arcsin \frac{c\tau}{d}, \quad (4.1)$$

where c is the speed of sound. While in this simple example we could use some cross-correlation technique [72] to estimate τ , this method would not scale well to more than two microphones. More sophisticated algorithms accommodating more microphones usually do not have this intermediate step.

4.1.1 Related Work

A wishlist for a direction of arrival (DOA) estimator may look something like Table 4.1. It should be high-resolution and give a precise estimate of sources locations. It should work at low signal-to-noise ratios (SNR) and resolve many possibly closely spaced sources. Finally, it should work with few arbitrarily laid out microphones, and do so efficiently, without grid searches.

It is uncommon to have all of these items checked at once. For example, the steered response power (SRP) methods [24] can be made robust, do not require a specific array geometry, and are

	MUSIC	SRP-PHAT	FRIDA
Find multiple sources	✓	✓	✓
Wideband	✓	✓	✓
Arbitrary array layout	✓	✓	✓
High-resolution	✓	✗	✓
Few microphones	✗	✓	✓
No grid search	✗	✗	✓
Correlated sources	✗	✓	✓
Computational complexity	Moderate	Low	High

Table 4.1: A wish-list for DOA algorithms along with a comparison of how MUSIC [116], SRP-PHAT [35], and FRIDA fulfill these wishes (✓) or not (✗).

immune to coherence in signals. Because they are based on beamforming though, they cannot resolve close sources [35].

Close sources can be resolved by the high-resolution DOA finders. Their main representatives are subspace methods such as MUSIC [116], Prony-type methods such as root-MUSIC [46], and methods that attempt to compute the maximum likelihood (ML) estimator such as IQML [20].

Subspace methods exploit the fact that for uncorrelated signal and noise, the eigenspace of the spatial covariance matrix corresponding to largest eigenvalues is spanned by the source steering vectors [116]. These methods are fundamentally narrowband since the signal subspaces vary with frequency; they can be made wideband either by incoherently combining narrowband estimates or, better, by combining them coherently through transforming the array manifold at each frequency to a manifold at a reference frequency (CSSM [127], WAVES [34]). These methods require a search over space unless the array is a uniform linear array (ULA) [9]. Coherent methods also require special “focusing matrices”, essentially initial guesses of the source locations. WAVES can do without focusing but at the cost of performance. In between coherent and incoherent methods is the TOPS algorithm [136], which performs well at mid-SNR, but still requires a search and performs worse than coherent methods at low SNR.

Grid-search free methods usually rely on polynomial rooting and can be applied straightforwardly to uniform linear arrays [9, 61] or circular arrays [64]. They can be extended in some cases to arbitrary array layout by using array interpolation techniques [12, 21, 46, 111].

Finally, in recent years, techniques based on sparsity inducing norm optimization have gained attention [66, 82]. A notable example is the work by Ma et al. that allows to locate more sources than the number of microphones available using Khatri–Rao subspace techniques [81].

4.1.2 Main Contribution

We propose a new finite rate of innovation (FRI) sampling-based algorithm for DOA finding—FRIDA. Among the mentioned algorithms, FRIDA is most reminiscent of IQML [20], especially for narrowband signals and ULAs. Unlike IQML, FRIDA works for arbitrary sensor geometries and for wideband signals. Moreover, it uses multi-band information coherently. Still, it requires

no grid search and no sensitive preprocessing akin to focusing matrices, and it achieves very high resolution at very low SNR, outperforming previous state-of-the-art.

FRIDA can work with fewer microphones than sources as it uses cross-correlations instead of raw microphone streams. The tradeoff is that it is not able to handle completely correlated signals. A straightforward modification of the algorithm which operates on raw signals rather than cross-correlations does not have this issue, but it requires more microphones.

The main ingredient of FRIDA is an FRI sampling algorithm [126]. FRI sampling has recently been extended to non-uniform grids along with a robust reconstruction algorithm [97]. The algorithm is an iterative algorithm similar to IQML, but with an added spectral resampling layer and a modified stopping criterion (Section 4.3.3). The key insight is that the elements of the spatial correlation matrix can be *linearly* transformed into uniformly sampled sums of sinusoids, regardless of the array geometry.

4.1.3 Chapter Organization

In Section 4.2, we lay out a mathematical framework to work with point sources and describe the microphone measurements produced by such sources. The linear mapping between these measurements and a uniformly sampled sum of sinusoids is described in Section 4.3 along with the point reconstruction algorithm. Results of experiments both on synthetic and recorded signals are presented in Section 4.4. Finally, we outline an extension of the method to blind sparse channel identification in Section 4.5.

4.2 Source Signal and Measurements

In this section, we start from far-field sources with an extended spatial support and propose a mathematical derivation of point sources. Given this model for point sources, we describe two different kind of microphone measurements. First, we describe the raw microphone signal. Then, we describe the cross-correlation of microphone signals in the special case of uncorrelated sources. We treat both the 2D and 3D case in a unified manner.

Throughout the chapter, matrices and vectors are denoted by bold upper and lower case letters, respectively. The Euclidean norm of a vector \mathbf{x} is denoted by $\|\mathbf{x}\|_2 = (\mathbf{x}^H \mathbf{x})^{1/2}$, where $(\cdot)^H$ is the Hermitian transpose. In this section, we use \mathbb{S} for both the unit circle (in 2D) and the unit sphere (in 3D). In the rest of the chapter, we'll use the proper notation of \mathbb{S}^1 and \mathbb{S}^2 for the unit circle and sphere, respectively, whenever necessary. Unit propagation vectors will be denoted by \mathbf{p} . In 2D, only the azimuth direction $\varphi \in [0, 2\pi]$ is required so that $\mathbf{p} \in \mathbb{S}^1$ is

$$\mathbf{p} = [\cos \varphi, \sin \varphi]^\top.$$

In 3D, we use azimuth $\varphi \in [0, 2\pi]$ and colatitude $\theta \in [0, \pi]$, thus a unit vector $\mathbf{p} \in \mathbb{S}^2$ is

$$\mathbf{p} = [\cos \varphi \sin \theta, \sin \varphi \sin \theta, \cos \theta]^\top.$$

We will later deal with a collection of propagation vectors $\{\mathbf{p}_k\}_{k=1}^K$ and we will naturally extend the indexing to the corresponding azimuth φ_k and elevation θ_k .

4.2.1 Sources with Arbitrary Spatial Support

We assume a setup with Q microphones located at $\{\mathbf{r}_q \in \mathbb{R}^2\}_{q=1}^Q$, and K monochromatic sources in the far-field whose sound propagates in the direction of the unit vectors $\{\mathbf{p}_k \in \mathbb{S}\}_{k=1}^K$. Within

a narrow band centered at frequency ω , the baseband representation of the signal coming from direction $\mathbf{p} \in \mathbb{S}$ reads

$$x(\mathbf{p}, \omega, t) = \tilde{x}(\mathbf{p}, \omega) e^{j\omega t},$$

where $\tilde{x}(\mathbf{p}, \omega)$ is the emitted sound signal by a source located at \mathbf{p} and frequency ω .

We will consider recovery of the direction of arrival from both the microphone signals and the cross-correlations between microphone pairs. The received signal at the q -th microphone located at \mathbf{r}_q is the integration of all plane waves along the unit circle:

$$y_q(\omega, t) = \int_{\mathbb{S}} x(\mathbf{p}, \omega, t) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} d\mathbf{p}, \quad \text{for } q = 1, \dots, Q, \quad (4.2)$$

where c is the speed of sound. The cross-correlation between a microphone pair (q, q') is

$$V_{q,q'}(\omega) \stackrel{\text{def}}{=} \mathbb{E} [y_q(\omega, t) y_{q'}^*(\omega, t)] = \int_{\mathbb{S}} \int_{\mathbb{S}} \mathbb{E} [\tilde{x}(\mathbf{p}, \omega) \tilde{x}^*(\mathbf{p}', \omega)] e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} e^{j\omega \langle \mathbf{p}', \frac{\mathbf{r}_{q'}}{c} \rangle} d\mathbf{p} d\mathbf{p}'$$

for $q, q' \in [1, Q]$ and $q \neq q'$. We assume frame-based processing, and the expectation is over the randomness of \tilde{x} from frame to frame. As \tilde{x} carries the phase, the assumption $\mathbb{E}\tilde{x} = 0$ holds. In practice, $V_{q,q'}$ is estimated by averaging over frames; for simplicity we use the same symbol for the empirical version.

4.2.2 Point Sources

Assume now that our source distribution is a sum of spatially localized sources. We can write

$$\tilde{x}_\gamma(\mathbf{p}, \omega) = \sum_{k=1}^K \alpha_k(\omega) \phi_\gamma(\mathbf{p} - \mathbf{p}_k) \quad (4.3)$$

where $\phi_\gamma(\mathbf{p}) = \gamma \phi(\gamma \mathbf{p})$, ϕ is a non-negative localized function with $\int \phi d\mathbf{p} = 1$, and $\gamma > 0$ is the spatial scaling factor. The amplitudes $\alpha_k(\omega)$ are complex random variables such that $\mathbb{E}|\alpha_k(\omega)|^2 = \sigma_k^2(\omega)$. From this definition, we formally get that

$$\tilde{x}_\gamma(\mathbf{p}, \omega) \xrightarrow{\gamma \rightarrow \infty} \tilde{x}(\mathbf{p}, \omega) = \sum_{k=1}^K \alpha_k(\omega) \delta(\mathbf{p} - \mathbf{p}_k). \quad (4.4)$$

We are now ready to describe the measurements.

Theorem 4.1

The microphone measurements are

$$y_q(\omega) = \sum_{k=1}^K \alpha_k(\omega) e^{-j\omega \langle \mathbf{p}_k, \frac{\mathbf{r}_q}{c} \rangle}. \quad (4.5)$$

Proof.

We plug (4.4) into (4.2) to obtain

$$y_{q,\gamma}(\omega) = \int_{\mathbb{S}} \tilde{x}_\gamma(\mathbf{p}, \omega) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} d\mathbf{p},$$

and the dominated convergence theorem assures us that

$$y_q(\omega) = \lim_{\gamma \rightarrow \infty} y_{q,\gamma}(\omega) = \int_{\mathbb{S}} \tilde{x}(\mathbf{p}, \omega) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} = \sum_{k=1}^K \alpha_k(\omega) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle}.$$

□

Finally, the following lemma ensures that the signal model used remains bounded in energy.

Lemma 4.1

The source distribution (4.3) has finite energy

$$\int_{\mathbb{S}} \mathbb{E} |\tilde{x}_\gamma(\mathbf{p}, \omega)|^2 d\mathbf{p} \leq K \sum_{k=1}^K \sigma_k^2(\omega).$$

The proof is given in Appendix 4.A.

Note that we have not assumed anything regarding correlation of the sources. This means that it is possible to locate multiple correlated sources using the microphone signal measurements directly.

4.2.3 Uncorrelated Point Sources

In the special situation where all the sources are uncorrelated, it is possible to use the cross-correlations between the microphone signals as measurements. This has the advantage of providing a number of measurements quadratic in the number of microphones available. In turn, this allows to locate more sources than microphones are available.

The cross-correlation measurements for uncorrelated point sources are given by the following theorem.

Theorem 4.2

Let us consider the source model of (4.3) for K uncorrelated sources, that is satisfying

$$\mathbb{E}[\alpha_k(\omega) \alpha_{k'}^*(\omega)] = 0, \quad \forall k \neq k'.$$

Then, the cross-correlation is given by

$$V_{q,q'}(\omega) = \sum_{k=1}^K \sigma_k^2(\omega) e^{-j\omega \langle \mathbf{p}_k, \Delta \mathbf{r}_{q,q'} \rangle},$$

where $q, q' \in [1, Q]$ and $\Delta \mathbf{r}_{q,q'} = \frac{\mathbf{r}_q - \mathbf{r}_{q'}}{c}$.

Proof.

Let us define the cross-correlation of the source distribution

$$\begin{aligned} I_\gamma(\mathbf{p}, \mathbf{p}', \omega) &= \mathbb{E} [\tilde{x}_\gamma(\mathbf{p}, \omega) \tilde{x}_\gamma^*(\mathbf{p}', \omega)] = \sum_{k=1}^K \sum_{k'=1}^K \mathbb{E} [\alpha_k(\omega) \alpha_{k'}^*(\omega)] \phi_\gamma(\mathbf{p} - \mathbf{p}_k) \phi_\gamma(\mathbf{p}' - \mathbf{p}_k) \\ &= \sum_{k=1}^K \sigma_k^2 \phi_\gamma(\mathbf{p} - \mathbf{p}_k) \phi_\gamma(\mathbf{p}' - \mathbf{p}_k), \end{aligned}$$

where we used the uncorrelation of sources assumption in the last equality. Then, the cross-correlation between the microphone signals is given by

$$\begin{aligned} V_{q,q'}(\omega) &= \mathbb{E} [y_{q,\gamma}(\omega)y_{q',\gamma}^*(\omega)] = \int_{\mathbb{S}} \int_{\mathbb{S}} I_{\gamma}(\mathbf{p}, \mathbf{p}', \omega) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} e^{j\omega \langle \mathbf{p}', \frac{\mathbf{r}_{q'}}{c} \rangle} d\mathbf{p} d\mathbf{p}' \\ &= \int_{\mathbb{S}} \int_{\mathbb{S}} \sum_{k=1}^K \sigma_k^2 \phi_{\gamma}(\mathbf{p} - \mathbf{p}_k) \phi_{\gamma}(\mathbf{p}' - \mathbf{p}_k) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} e^{j\omega \langle \mathbf{p}', \frac{\mathbf{r}_{q'}}{c} \rangle} d\mathbf{p} d\mathbf{p}' \\ &= \sum_{k=1}^K \sigma_k^2 \left(\int_{\mathbb{S}} \phi_{\gamma}(\mathbf{p} - \mathbf{p}_k) e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{c} \rangle} d\mathbf{p} \right) \left(\int_{\mathbb{S}} \phi_{\gamma}(\mathbf{p}' - \mathbf{p}_k) e^{j\omega \langle \mathbf{p}', \frac{\mathbf{r}_{q'}}{c} \rangle} d\mathbf{p}' \right). \end{aligned}$$

Finally, by taking $\gamma \rightarrow \infty$ and invoking the dominated convergence theorem, we obtain the desired result

$$V_{q,q'}(\omega) = \lim_{\gamma \rightarrow \infty} V_{q,q',\gamma}(\omega) = \sum_{k=1}^K \sigma_k^2 e^{-j\omega \langle \mathbf{p}, \Delta \mathbf{r}_{q,q'} \rangle}.$$

□

This being settled, we define the *intensity* of the soundfield as

$$I(\mathbf{p}, \omega) = \sum_{k=1}^K \sigma_k^2 \delta(\mathbf{p} - \mathbf{p}_k) \quad (4.6)$$

so that the following relationship holds

$$V_{q,q'}(\omega) = \int_{\mathbb{S}} I(\mathbf{p}, \omega) e^{-j\omega \langle \mathbf{p}, \Delta \mathbf{r}_{q,q'} \rangle} = \sum_{k=1}^K \sigma_k^2 e^{-j\omega \langle \mathbf{p}, \Delta \mathbf{r}_{q,q'} \rangle}. \quad (4.7)$$

4.3 Point Source Reconstruction

We propose to reconstruct the locations of the point sources using FRI techniques. These techniques are concerned with the sampling and reconstructions of signals that can be described by a finite number of parameters, as is the case for DOA estimation. They find their roots in Prony's work on the estimation of sinusoids in time series [106]. Likewise, the archetypal FRI problem is to estimate the frequencies of a sum of uniformly sampled complex exponentials buried in noise. Recently, Pan and collaborators proposed a generalized FRI sampling framework with a robust reconstruction algorithm that extends these methods to non-uniformly sampled data [97]. The three main ingredients of this method are

- a set of uniformly sampled, but unknown, sinusoidal samples,
- a linear mapping between the non-uniformly, but known, measurements and the previously identified uniform samples,
- and an unknown annihilation filter that constrains the location of the uniform samples.

The reconstruction is then formulated as a constrained optimization problem involving a fitting criterion between the reconstructed model and the given non-uniform measurements together with the annihilation constraint.

Following this line of work, we will first identify the set of unknown sinusoidal samples and its relation to the given measurements. The mapping is described for four cases, raw microphone and cross-correlations, in both 2D and 3D. The measurements, mappings, and corresponding sums of sinusoids are summarized in Table 4.2. Then, the DOA estimation is cast as a constrained optimization (see e.g., (4.14)). While the discussion of the optimization and the results or numerical experiments in Section 4.4 are limited to the 2D case to be concise, the method has been since extended to the 3D case.

4.3.1 Relation between Measurements and Uniform Samples of Sinusoids

We develop in this section the mapping for microphone signal measurements first on the circle, then on the sphere, and then do similarly for cross-correlation measurements.

Microphone signal measurements on the Circle Since \mathbf{p} is supported on the circle, we have the following Fourier series representation for the sound field:

$$\tilde{x}(\mathbf{p}, \omega) = \sum_{m \in \mathbb{Z}} \hat{A}_m(\omega) Y_m(\mathbf{p}),$$

where $Y_m(\mathbf{p})$ is the Fourier series basis $Y_m(\mathbf{p}) = Y_m(\varphi) = e^{jm\varphi}$, and $\hat{A}_m(\omega)$ is the associated expansion coefficient for a sub-band centered at frequency ω :

$$\hat{A}_m(\omega) = \frac{1}{2\pi} \int_{\mathbb{S}^1} \tilde{x}(\mathbf{p}, \omega) Y_m^*(\mathbf{p}) d\mathbf{p} = \frac{1}{2\pi} \sum_{k=1}^K \alpha_k(\omega) e^{-jm\varphi_k}. \quad (4.8)$$

Notice that the Fourier series coefficients $\hat{A}_m(\omega)$ for $m \in \mathbb{Z}$ are uniform samples of sinusoids, which are related with the microphone signal (4.5) as:

$$\begin{aligned} y_q(\omega) &= \int_{\mathbb{S}^1} \sum_{m \in \mathbb{Z}} \hat{A}_m(\omega) Y_m(\mathbf{p}) e^{-j\omega \langle \mathbf{p}, \mathbf{r}_q \rangle} d\mathbf{p} \\ &\stackrel{(a)}{=} 2\pi \sum_{m \in \mathbb{Z}} (-j)^m J_m(\|\omega \mathbf{r}_q\|_2) Y_m\left(\frac{\mathbf{r}_q}{\|\mathbf{r}_q\|_2}\right) \hat{A}_m \end{aligned} \quad (4.9)$$

where (a) is from Jacobi-Anger expansion [30] of the complex exponential and $J_m(\cdot)$ is Bessel function of the first kind.

Therefore, we establish a linear mapping from the uniformly sampled sinusoids \hat{A}_m to the given measurements y_q . Concretely, denote a vector of measurements $y_q(\omega)$, by $\mathbf{a}(\omega) \in \mathbb{C}^Q$, and let the vector $\mathbf{b}(\omega)$ be the Fourier series coefficients $\hat{A}_m(\omega)$ for $m \in \mathcal{M}$, where \mathcal{M} is a set of considered Fourier coefficients¹. Define also a $Q \times |\mathcal{M}|$ matrix $\mathbf{G}(\omega)$ as

$$g_{q,m}(\omega) \stackrel{\text{def}}{=} (-j)^m J_m(\|\omega \mathbf{r}_q\|_2) Y_m\left(\frac{\mathbf{r}_q}{\|\mathbf{r}_q\|_2}\right),$$

where rows of \mathbf{G} are indexed by microphone q , and columns of \mathbf{G} are indexed by Fourier bins m . We can then concisely write (4.12) as $\mathbf{a}(\omega) = \mathbf{G}(\omega)\mathbf{b}(\omega)$.

¹Note that these correspond to the spatial Fourier transform of I over the circle, not to sources' temporal spectra.

Table 4.2: Summary of the different mappings.

Measurements — \mathbf{a}	Mapping — \mathbf{G} (2D: $g_{q,m}$, 3D: $g_{(q,q'),(m,n)}$)	Sinusoids — \mathbf{b}
MICROPHONE SIGNALS		
$\sum_{k=1}^K \alpha_k e^{-j\omega \langle \mathbf{p}, \frac{\mathbf{r}_q}{\ \mathbf{r}_q\ _2} \rangle}$	2D $(-j)^m J_m(\ \omega \mathbf{r}_q\ _2) Y_m\left(\frac{\mathbf{r}_q}{\ \mathbf{r}_q\ _2}\right)$	$\sum_{k=1}^K \alpha_k e^{-jm\varphi_k}$
	3D $\sum_{\ell= m }^{\infty} (-j)^\ell \frac{J_{\ell+1/2}(\ \omega \mathbf{r}_q\ _2)}{(\ \omega \mathbf{r}_q\ _2)^{1/2}} Y_{\ell,m}\left(\frac{\omega \mathbf{r}_q}{\ \omega \mathbf{r}_q\ _2}\right) N_{\ell,m} \sum_{n=0}^{\ell- m } \gamma_{n, m }$	$\sum_{k=1}^K \alpha_k (\cos \theta_k)^n (\sin \theta_k)^{ m } e^{-jm\varphi_k}$
CROSS-CORRELATIONS		
$\sum_{k=1}^K \sigma_k^2 e^{-j\omega \langle \mathbf{p}, \Delta \mathbf{r}_{q,q'} \rangle}$	2D $(-j)^m J_m(\ \omega \Delta \mathbf{r}_{q,q'}\ _2) Y_m\left(\frac{\Delta \mathbf{r}_{q,q'}}{\ \Delta \mathbf{r}_{q,q'}\ _2}\right)$	$\sum_{k=1}^K \sigma_k^2 e^{-jm\varphi_k}$
	3D $\sum_{\ell= m }^{\infty} (-j)^\ell \frac{J_{\ell+1/2}(\ \omega \Delta \mathbf{r}_{q,q'}\ _2)}{(\ \omega \Delta \mathbf{r}_{q,q'}\ _2)^{1/2}} Y_{\ell,m}\left(\frac{\omega \Delta \mathbf{r}_{q,q'}}{\ \omega \Delta \mathbf{r}_{q,q'}\ _2}\right) N_{\ell,m} \sum_{n=0}^{\ell- m } \gamma_{n, m }$	$\sum_{k=1}^K \sigma_k^2 (\cos \theta_k)^n (\sin \theta_k)^{ m } e^{-jm\varphi_k}$

Microphone Signal Measurements on the Sphere When sources are located in 3D space, the sound field expansion is done on the sphere \mathbb{S}^2 in terms of the spherical harmonics

$$\tilde{x}(\mathbf{p}, \omega) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \hat{A}_{\ell,m} Y_{\ell,m}(\mathbf{p}). \quad (4.10)$$

Here $Y_{\ell,m}(\cdot)$ is the spherical harmonic of order ℓ and degree m , and $\hat{A}_{\ell,m}$ is the associated expansion coefficient. The spherical harmonics are defined as

$$Y_{\ell,m}(\mathbf{p}) = Y_{\ell,m}(\theta, \varphi) = N_{\ell,m} e^{jm\varphi} P_{\ell}^m(\cos \theta),$$

where P_{ℓ}^m is an associated Legendre polynomial, and N a normalization constant (see [107] for details). On the one hand, the cross-correlation measurements are linearly related with spherical harmonic coefficients from (4.7) and (4.10):

$$\begin{aligned} y_q(\omega) &= \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \hat{A}_{\ell,m} \iint_{\mathbb{S}^2} e^{-j\omega\langle \mathbf{p}, \mathbf{r}_q \rangle} Y_{\ell,m}(\mathbf{p}) d\mathbf{p} \\ &= (2\pi)^{3/2} \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} (-j)^{\ell} \frac{J_{\ell+1/2}(\|\omega \mathbf{r}_q\|_2)}{(\|\omega \mathbf{r}_q\|_2)^{1/2}} Y_{\ell,m} \left(\frac{\omega \mathbf{r}_q}{\|\omega \mathbf{r}_q\|_2} \right) \hat{A}_{\ell,m}, \end{aligned}$$

where $J_{\ell}(\cdot)$ is Bessel function of the first kind. On the other hand, the expansion coefficients of the spherical harmonic decomposition of (4.5) is [36, 97]

$$\hat{A}_{\ell,m} = N_{\ell,m} \sum_{n=0}^{\ell-|m|} \gamma_{n,|m|} b_{n,m}.$$

Here $N_{\ell,m}$ is a normalization factor; $\gamma_{n,|m|}$ are some fixed coefficients for a given ℓ , which can be precomputed; and

$$b_{n,m} = \sum_{k=1}^K \alpha_k(\omega) (\cos \theta_k)^n (\sin \theta_k)^{|m|} e^{-jm\varphi_k}.$$

Therefore, we establish a linear mapping from the uniformly sampled sinusoids to the given cross-correlation measurements:

$$y_q(\omega) = (2\pi)^{3/2} \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} (-j)^{\ell} \frac{J_{\ell+1/2}(\|\omega \mathbf{r}_q\|_2)}{(\|\omega \mathbf{r}_q\|_2)^{1/2}} Y_{\ell,m} \left(\frac{\omega \mathbf{r}_q}{\|\omega \mathbf{r}_q\|_2} \right) N_{\ell,m} \sum_{n=0}^{\ell-|m|} \gamma_{n,|m|} b_{n,m}.$$

Cross-correlation Measurements on the Circle We can find a similar mapping from the cross-correlation measurements (4.7) to uniform samples from sinusoids. Instead of $\tilde{x}(\mathbf{p}, \omega)$, we now consider instead the Fourier series of the intensity (4.6):

$$I(\mathbf{p}, \omega) = \sum_{m \in \mathbb{Z}} \hat{I}_m(\omega) Y_m(\mathbf{p}),$$

where $Y_m(\mathbf{p})$ is the Fourier series basis $Y_m(\mathbf{p}) = Y_m(\varphi) = e^{jm\varphi}$, and $\hat{I}_m(\omega)$ is the associated expansion coefficient for a sub-band centered at frequency ω :

$$\hat{I}_m(\omega) = \frac{1}{2\pi} \int_{\mathbb{S}^1} I(\mathbf{p}, \omega) Y_m^*(\mathbf{p}) d\mathbf{p} = \frac{1}{2\pi} \sum_{k=1}^K \sigma_k^2(\omega) e^{-jm\varphi_k}. \quad (4.11)$$

Table 4.3: Maximum number of sources detectable in the planar case.

Measurements	Max sources
Microphone signals	$K \leq \frac{Q-1}{2}$
Cross-correlation	$K \leq \frac{Q^2-Q-1}{2}$

Notice that the Fourier series coefficients $\hat{I}_m(\omega)$ for $m \in \mathbb{Z}$ are uniform samples of sinusoids, which are related with the cross-correlation (4.7) as:

$$\begin{aligned} V_{q,q'}(\omega) &= \int_{\mathbb{S}^1} \sum_{m \in \mathbb{Z}} \hat{I}_m(\omega) Y_m(\mathbf{p}) e^{-j\omega \langle \mathbf{p}, \Delta \mathbf{r}_{q,q'} \rangle} d\mathbf{p} \\ &\stackrel{(a)}{=} 2\pi \sum_{m \in \mathbb{Z}} (-j)^m J_m(\|\omega \Delta \mathbf{r}_{q,q'}\|_2) Y_m\left(\frac{\Delta \mathbf{r}_{q,q'}}{\|\Delta \mathbf{r}_{q,q'}\|_2}\right) \hat{I}_m \end{aligned} \quad (4.12)$$

where (a) is from Jacobi-Anger expansion [30] of the complex exponential and $J_m(\cdot)$ is the Bessel function of the first kind.

Therefore, we establish a linear mapping from the uniformly sampled sinusoids \hat{I}_m to the given measurements $V_{q,q'}$. Concretely, denote a lexicographically ordered vectorization of the cross-correlations $V_{q,q'}(\omega)$, $q \neq q'$ by $\mathbf{a}(\omega) \in \mathbb{C}^{Q(Q-1)}$, and let the vector $\mathbf{b}(\omega)$ be the Fourier series coefficients $\hat{I}_m(\omega)$ for $m \in \mathcal{M}$, where \mathcal{M} is a set of considered Fourier coefficients¹. Define also a $Q(Q-1) \times |\mathcal{M}|$ matrix $\mathbf{G}(\omega)$ as

$$g_{(q,q'),m}(\omega) \stackrel{\text{def}}{=} (-j)^m J_m(\|\omega \Delta \mathbf{r}_{q,q'}\|_2) Y_m\left(\frac{\Delta \mathbf{r}_{q,q'}}{\|\Delta \mathbf{r}_{q,q'}\|_2}\right),$$

where rows of \mathbf{G} are indexed by microphone pairs (q, q') , and columns of \mathbf{G} are indexed by Fourier bins m . We can then concisely write (4.12) as $\mathbf{a}(\omega) = \mathbf{G}(\omega)\mathbf{b}(\omega)$.

Cross-correlation Measurements on the Sphere Similarly to the case of the microphone signal measurements, we express the signal in terms of spherical harmonics

$$I(\mathbf{p}, \omega) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \hat{I}_{\ell,m} Y_{\ell,m}(\mathbf{p}).$$

After replacing the expression for $\hat{I}_{\ell,m}$ by its spherical harmonic representation and taking the expansion of the complex exponential in the spherical harmonic domain, we can establish a linear mapping from the uniformly sampled sinusoids to the given cross-correlation measurements:

$$V_{q,q'} = (2\pi)^{3/2} \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} (-j)^{\ell} \frac{J_{\ell+1/2}(\|\omega \Delta \mathbf{r}_{q,q'}\|_2)}{(\|\omega \Delta \mathbf{r}_{q,q'}\|_2)^{1/2}} Y_{\ell,m}\left(\frac{\omega \Delta \mathbf{r}_{q,q'}}{\|\omega \Delta \mathbf{r}_{q,q'}\|_2}\right) N_{\ell,m} \sum_{n=0}^{\ell-|m|} \gamma_{n,|m|} b_{n,m}.$$

4.3.2 Annihilation on the Circle

Since \hat{I}_m in (4.11) is a weighted sum of uniformly sampled sinusoids, we know that \hat{I}_m should satisfy a set of annihilation equations [126]:

$$\hat{I}_m * h_m = 0. \quad (4.13)$$

Here h_m is the unknown annihilating filter to be recovered. A polynomial, whose coefficients are specified by the filter h_m , has roots related to sources locations. In 2D, they are located exactly at $e^{-j\varphi_k}$ [126]. The source azimuths φ_k are subsequently reconstructed with polynomial root-finding. When going to 3D, (4.13) is replaced by a two-dimensional convolution. Again, the roots of the associated bi-variate polynomial is linked to the location of the sources, albeit in a slightly less straightforward way [97]. In the remainder of this section, we stick to the 2D case for clarity.

In a multi-band setting, the uniform sinusoidal samples $\hat{I}_m(\omega)$ are different for each sub-band. This is because, the signal power σ_k^2 varies with the mid-band frequency ω in general. However, since we have *the same* source locations φ_k for each sub-band, we only need to find one filter h_m (depending solely on the source locations φ_k) that annihilates $\hat{I}_m(\omega)$ for all ω -s:

$$\hat{I}_m(\omega) *_{m} h_m = 0 \quad \forall \omega.$$

4.3.3 Reconstruction Algorithm

Following the discussion in the previous section, we reconstruct the source locations *jointly* across all sub-bands. More specifically, suppose we consider J sub-bands centered around frequencies $\{\omega_j\}_{j=1}^J$. Then, we formulate the FRIDA estimate as a solution of the following constrained optimization:

$$\min_{\substack{\mathbf{b}_1, \dots, \mathbf{b}_J \\ \mathbf{h} \in \mathcal{H}}} \sum_{i=1}^J \|\mathbf{a}_i - \mathbf{G}_i \mathbf{b}_i\|_2^2 \quad (4.14)$$

$$\text{subject to } \mathbf{b}_i * \mathbf{h} = \mathbf{0} \quad \text{for } i = 1, \dots, J,$$

Here \mathbf{a}_i , \mathbf{b}_i and \mathbf{G}_i are the cross-correlation, uniform sinusoidal samples, and the linear mapping between them for the i -th sub-band as specified in Section 4.3.1; \mathcal{H} is a feasible set that avoids the trivial solution. It was found that a random linear constraint is a good choice for this set, e.g. $\mathcal{H} = \{\mathbf{h} \in \mathbb{C}^{K+1} : \mathbf{h}_0^H \mathbf{h} = 1\}$, with \mathbf{h}_0 a unit norm random vector [97].

Note that (4.14) is a simple quadratic minimization with respect to \mathbf{b}_i -s for a given annihilating filter \mathbf{h} . By substituting the solution of \mathbf{b}_i (in function of \mathbf{h}), we end up with an optimization for \mathbf{h} alone:

$$\min_{\mathbf{h} \in \mathcal{H}} \mathbf{h}^H \mathbf{\Lambda}(\mathbf{h}) \mathbf{h}, \quad (4.15)$$

where

$$\mathbf{\Lambda}(\mathbf{h}) = \sum_{i=1}^J \mathbf{T}^H(\boldsymbol{\beta}_i) \left[\mathbf{R}(\mathbf{h}) (\mathbf{G}_i^H \mathbf{G}_i)^{-1} \mathbf{R}^H(\mathbf{h}) \right]^{-1} \mathbf{T}(\boldsymbol{\beta}_i).$$

Here $\boldsymbol{\beta}_i$ is the least-squares solution to the unconstrained problem

$$\boldsymbol{\beta}_i = (\mathbf{G}_i^H \mathbf{G}_i)^{-1} \mathbf{G}_i^H \mathbf{a}_i.$$

The two operators $\mathbf{T}(\cdot)$ and $\mathbf{R}(\cdot)$ represent the convolution operation and are defined with respect to vectors $\mathbf{b} = [b_{-M}, \dots, b_M]^T$ and $\mathbf{h} = [h_0, \dots, h_K]^T$. In the planar case, the former builds a Toeplitz matrix from the input vector \mathbf{b}

$$\mathbf{T}(\mathbf{b}) = \begin{bmatrix} b_{-M+K} & b_{-M+K-1} & \cdots & b_{-M} \\ b_{-M+K+1} & b_{-M+K} & \cdots & b_{-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ b_M & b_{M-1} & \cdots & b_{M-K} \end{bmatrix},$$

Algorithm 4.1 FRIDA: FRI-based DOA estimation**Require:** cross-correlation of the microphone signals \mathbf{a}_i , linear mapping \mathbf{G}_i , noise level ε^2 **Ensure:** uniform sinusoidal samples \mathbf{b}_i , annihilating filter coefficients \mathbf{h}

```

for loop  $\leftarrow$  1 to max. initializations do
  Initialize  $\mathbf{h}$  with a random vector  $\mathbf{h}^{(0)}$ 
  for  $n \leftarrow$  1 to max. iterations do
    # Build  $\Lambda(\mathbf{h})$  with  $\mathbf{h} = \mathbf{h}^{(n-1)}$  and update  $\mathbf{h}^{(n)}$  by solving (4.15)
     $\mathbf{h}_n \leftarrow \Lambda(\mathbf{h}_{n-1})\mathbf{h}_0 / \mathbf{h}_0^H \Lambda(\mathbf{h}_{n-1})\mathbf{h}_0$ 
    # Re-synthesize  $\mathbf{b}_i^{(n)}$  with the updated annihilating filter  $\mathbf{h} = \mathbf{h}^{(n)}$  as:
     $\mathbf{b}_i(\mathbf{h}) \leftarrow \mathbf{b}_i - (\mathbf{G}_i^H \mathbf{G}_i)^{-1} \mathbf{R}^H(\mathbf{h}) \cdot (\mathbf{R}(\mathbf{h})(\mathbf{G}_i^H \mathbf{G}_i)^{-1} \mathbf{R}^H(\mathbf{h}))^{-1} \mathbf{R}(\mathbf{h})\mathbf{b}_i$ 
    # Check the termination condition
    if  $\sum_{i=1}^J \|\mathbf{a}_i - \mathbf{G}_i \mathbf{b}_i^{(n)}\|_2^2 \leq \varepsilon^2$  then
      Terminate both loops
    end if
  end for
end for
 $\mathbf{b}_i \leftarrow \mathbf{b}_i^{(n)}, \mathbf{h} \leftarrow \mathbf{h}^{(n)}$ 

```

while the latter is the *right-dual* convolution matrix associated with $\mathbf{T}(\cdot)$

$$\mathbf{R}(\mathbf{h}) = \begin{bmatrix} h_K & h_{K-1} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_K & h_{K-1} & \cdots & h_0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & & \ddots & \vdots \\ 0 & \cdots & 0 & h_K & h_{K-1} & \cdots & h_0 \end{bmatrix}$$

such that

$$\mathbf{T}(\mathbf{b})\mathbf{h} = \mathbf{R}(\mathbf{h})\mathbf{b}, \quad \forall \mathbf{b}, \mathbf{h}.$$

This follows from the commutativity of convolution: $\mathbf{b} * \mathbf{h} = \mathbf{h} * \mathbf{b}$. In the spherical case, the corresponding bi-dimensional operators are similarly defined.

In general, it is challenging to solve (4.15) directly. We use an iterative strategy, building $\Lambda(\mathbf{h})$ with the reconstructed \mathbf{h} from the previous iteration. By doing this, (4.14) becomes a linearly constrained quadratic minimization with an analytical solution that can be used to write the following recurrence

$$\mathbf{h}_n = \frac{\Lambda(\mathbf{h}_{n-1})\mathbf{h}_0}{\mathbf{h}_0^H \Lambda(\mathbf{h}_{n-1})\mathbf{h}_0},$$

where \mathbf{h}_0 is the random linear constraint defining the feasible set. It is also used as the initialization of the iterative procedure as was found to work well in practice [97]. Unlike similar approaches (e.g. [20]), we do not aim at obtaining a convergent solution of (4.15) but rather a valid solution such that the reconstructed sinusoidal samples \mathbf{b}_i -s explain the given measurements up to a certain approximation level (ε^2): $\sum_{i=1}^J \|\mathbf{a}_i - \mathbf{G}_i \mathbf{b}_i\|_2^2 \leq \varepsilon^2$. Pseudo-code is given in Algorithm 4.1. Readers are referred to [97] for detailed discussions on the algorithmic details, e.g., choice of ε , implementation details, etc.

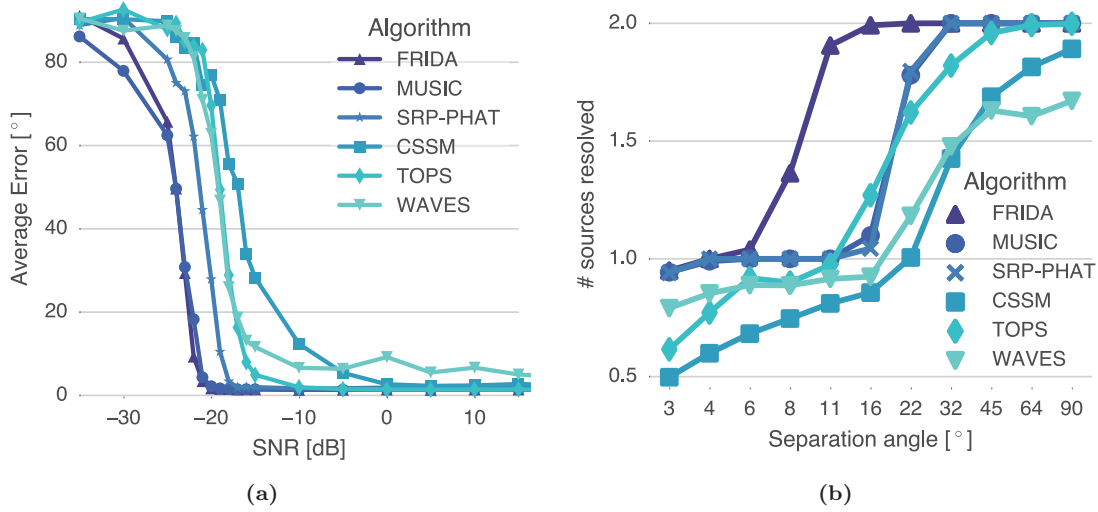


Figure 4.2: (a) Average DOA reconstruction error as a function of SNR. Lower is better. (b) Average number of sources reconstructed for the case of two sources separated by a fixed angle.

4.4 Experiments

In this section, we demonstrate the effectiveness of the proposed algorithm through numerical simulations and practical experiments. We compare the performance of FRIDA to that of other wideband algorithms: incoherent MUSIC [116], SRP-PHAT [35], CSSM [127], WAVES [34], and TOPS [136].

The sampling frequency is fixed at 16 kHz. The narrow-band sub-carriers are extracted by a 256-point short-time Fourier transform (STFT) with a Hanning window and no overlap. We use a triangular array of 24 microphones. Each edge is 30 cm long and carries 8 microphones. The spacing of microphones ranges from 8 mm to 25 cm. This geometry is that of the *Pyramic* compact array designed at EPFL [7] and used to collect the recordings for the practical experiments, see Fig. 4.4A.

The number of frequency bands used (out of the 128 narrow-bands) is a key parameter for performance and was tuned for each algorithm. FRIDA, MUSIC and SRP-PHAT use 20 bands, CSSM and WAVES 10 bands, and TOPS 60 bands. In the synthetic experiments, the source signals are all white noise to simplify the choice of the sub-bands. For speech recordings, the STFT bins with the largest power are chosen. All implementation details are in the supplementary material.

The reconstruction errors are quantified according to the distance on the unit circle defined as

$$d_{\mathbb{S}}(\varphi, \hat{\varphi}) = \min_{s \in \{\pm 1\}} s(\varphi - \hat{\varphi}) \bmod 2\pi. \quad (4.16)$$

For multiple DOA, the originals and their reconstructions are matched to minimize the sum of errors.

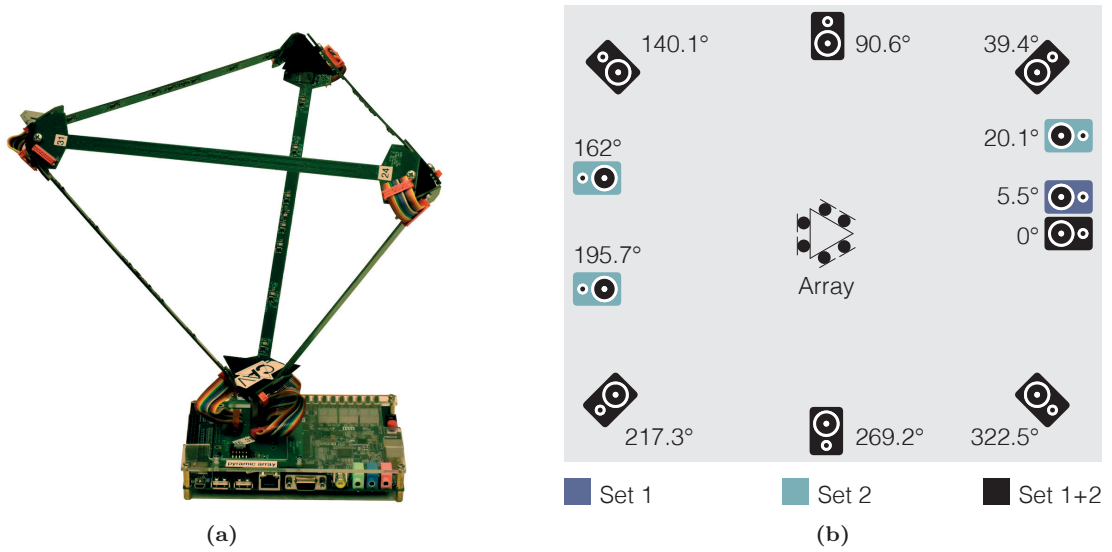


Figure 4.3: (a) Pyramic array, a compact microphone array with 48 MEMS microphones distributed on the edges of a tetrahedron. For the experiments, only the top triangle is used. (b) Locations of the loudspeakers and microphone array in experiments.

DOA	FRIDA	MUSIC	SRP-PHAT
0°	$-0.5 \pm 0.4^\circ$	$1.6 \pm 0.3^\circ$	$1.4 \pm 0.2^\circ$
5.5°	$4.6 \pm 0.2^\circ$	$-93.9 \pm 41.2^\circ$	$-38.1 \pm 8.6^\circ$

Table 4.4: The accuracy of the reconstruction for recordings with sources closely located at 0° and 5.5°. The mean is computed as the logarithm of the average of complex exponentials with argument given by the reconstruction angle. The second number is the average distance (4.16) from the sample to the mean.

4.4.1 Influence of Noise

We study the influence of noise on the algorithms through numerical simulation. One source playing white noise is placed at random on the unit circle. The propagation of sound is simulated by applying fractional delay filters to generate the microphone signals based on the array geometry. Finally, the algorithms are run with additive white Gaussian noise of variance corresponding to a wide range of SNR. The algorithms are fed with 256 snapshots of 256 samples each. It should be noted that 256 snapshots correspond to a processing gain of about 24 dB. We run 500 rounds of Monte-Carlo simulation for each SNR value.

The simulation results in Fig. 4.2A show that FRIDA and MUSIC are the most robust with a breaking points slightly below -20 dB. Next are SRP-PHAT and TOPS, breaking around 2 dB and 4 dB higher, respectively. While WAVES initially seems to perform as well as TOPS, it never reaches zero error. Least resistant to noise is CSSM, breaking down as early as -5 dB. The poor performance of WAVES and CSSM might be attributed to poor initial estimates of the focusing frequencies.

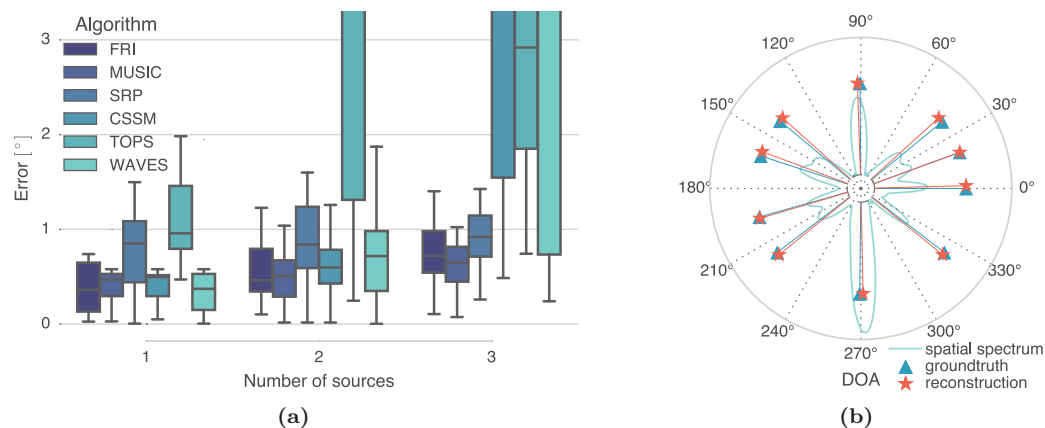


Figure 4.4: (a) Reconstruction error for the different algorithms applied to the recorded speech signals. (b) Reconstruction of 10 sources from only 9 microphones. The average reconstruction error is within 2°.

4.4.2 Resolving Close Sources

Next, we study the minimum angle of separation necessary to resolve distinct sources. We simulate two sources of white noise at angles φ and $\varphi + \delta$ where δ is varied from 90° to 2.8° . The average error is then computed over ten realizations of the noise for 120 values of φ . We mark a DOA as successfully recovered if the reconstruction error is less than $\delta/2$. This criterion seems crude for large δ , but for small δ , where performance is critical, it is very stringent. Here again 256 snapshots are used and the SNR is set to 0 dB.

As seen in Fig. 4.2B, we find that FRIDA largely outperforms the other algorithms. It always separates sources located as close as 11.2° , while the closest contenders, MUSIC and SRP-PHAT, have difficulties for sources closer than 22.5° . The coherent methods perform worse than the incoherent ones; they even suffer from a lack of precision in estimating a single source.

4.4.3 Experiments on Recorded Signals

Finally, we perform two experiments with recorded data to validate the algorithm in non-ideal, real-world conditions. In the first experiment, the Pyramic array is placed at the center of eight loudspeakers (Fig. 4.4B, Set 1). All the loudspeakers are between 1.45 m and 2.45 m away from the array. Recordings are made with all possible combinations of one, two, and three speakers playing simultaneously (distinct) speech segments of 3 to 4 seconds duration. Two of the speakers are located at 5.5° of each other to test the resolving power of the algorithms.

The statistics of the reconstruction errors for the different algorithms are shown in Fig. 4.4C. We find the coherent methods WAVES and CSSM to perform well for one and two sources, but break down for three sources. The TOPS method maintains an acceptable but somewhat imprecise performance for more than one source. FRIDA, MUSIC and SRP-PHAT perform best with a median error within one degree from the ground truth. Where FRIDA distinguishes itself from the conventional methods is for closely spaced sources. This is highlighted in Table 4.4 where the average reconstructed DOA for the sources located at 0° and 5.5° is shown. While all three methods correctly identify the first source, only FRIDA is able to resolve the second.

The second experiment tests the ability of FRIDA to resolve more sources than microphones are used. We place ten loudspeakers (Fig. 4.4B, Set 2) around the Pyramic array and record them simultaneously playing white noise. Then, we discard the signals of all but nine microphones and run FRIDA. The algorithm successfully reconstructs all DOA within 2° of the ground truth, as shown in Fig. 4.4D. Note that none of the subspace methods can achieve this result. While SRP-PHAT is not limited in this way, its resolution is lower (its error is $\sim 4^\circ$ on this recording).

4.5 From Direction of Arrival to Blind Sparse Channel Identification

The problem of direction of arrival is intimately linked to that of blind sparse channel identification. Sparse channels are a simple model for the early echoes in a room impulse response (RIR)

$$h_q(t) = \sum_{k=1}^K \alpha_k \delta(t - \tau_k - \langle \mathbf{r}_q/c, \mathbf{p}_k \rangle) + \text{tail}(t) \quad (4.17)$$

where $\text{tail}(t)$ is the reverberant tail of the RIR. Since the tail is in general not sparse, we do not attempt to estimate it and lump it with the noise instead. For the sake of intelligibility, we limit the following discussion to the planar case.

From a geometrical perspective, and based on the image source model [4], the echoes can be modelled as additional sources, all playing the same signal, but with an attenuation factor and a delay. Assuming the source and its images have DOA $\mathbf{p}_1, \dots, \mathbf{p}_K$, then the narrow-band far-field signal (4.4) is modified

$$\tilde{x}(\mathbf{p}, \omega) = \tilde{x}(\omega) \sum_{k=1}^K \alpha_k e^{-j\omega\tau_k} \delta(\mathbf{p} - \mathbf{p}_k),$$

where $\tilde{x}(\omega)$ is the spectrum of the (unique) source. This signal admits a Fourier series representation similar to that of (4.8). By choosing carefully multiple uniformly sampled frequency bands $\omega_n = \omega_0 n$, this representation is a sum of uniformly sampled sinusoids

$$\hat{A}_{m,n} = \frac{1}{2\pi} \tilde{x}(\omega_n) \sum_{k=1}^K \alpha_k e^{-j\omega_n \tau_k} e^{-jm\varphi_k}.$$

As such, it can be annihilated by a well-chosen 2D filter, i.e. there exists $h_{m,n}$

$$\hat{A}_{m,n} \underset{(m,n)}{*} h_{m,n} = 0.$$

Finally, by a small modification of (4.9), we obtain a linear mapping between the microphone measurements and the uniformly sampled sinusoids signal

$$y_q(\omega_n) = 2\pi \sum_{m \in \mathbb{Z}} (-j)^m J_m(\|\omega_n \mathbf{r}_q/c\|_2) Y_m\left(\frac{\mathbf{r}_q}{\|\mathbf{r}_q\|_2}\right) \hat{A}_{m,n}.$$

Given the linear mapping and the annihilation constraint, it is possible to use the generalized FRI framework [97] to reconstruct the pairs (\mathbf{p}_k, τ_k) and in turn the sparse part of the impulse response (4.17), up to a global timing constant τ_0 due to the unknown random phase of the original source signal.

4.6 Conclusion

We introduced FRIDA, a new algorithm for DOA estimation of sound sources. FRIDA relies on finite rate of innovation sampling to do so efficiently on arbitrary array geometries, avoiding any costly grid search. Its ability to use wideband signal information makes it robust to many types of noise and interference. We demonstrate that FRIDA compares favorably to the state-of-the-art, and clearly outperforms all other algorithms when it comes to resolving close sources. Moreover, FRIDA is notable for resolving more sources than microphones, as demonstrated experimentally on recorded signals.

FRIDA suffers however from a large computational complexity and it is of practical interest to improve this aspect of the algorithm. One possible avenue is to replace the exact solution of linear systems involved in each iteration by an approximation through a few steps of gradient descent, for example. When dealing with moving sources, it is relevant to investigate if warm start from a previous estimate of source locations might help convergence.

4.A Proof of Lemma 4.1

Proof.

Let us first define the cross-correlation between two locations \mathbf{p} and \mathbf{p}' for source k

$$I_k(\mathbf{p}, \mathbf{p}', \omega) = \mathbb{E}[\alpha_k(\omega)\phi_\gamma(\mathbf{p} - \mathbf{p}_k)\alpha_k^*(\omega)\phi_\gamma(\mathbf{p}' - \mathbf{p}_k)] = \sigma_k^2(\omega)\phi_\gamma(\mathbf{p} - \mathbf{p}_k)\phi_\gamma(\mathbf{p}' - \mathbf{p}_k).$$

Now the triangle inequality gives the following upper bound on the energy Using the convexity of the squared norm, we obtain the following bound on the energy

$$\mathbb{E}|\tilde{x}_\gamma(\mathbf{p}, \omega)|^2 \leq K \sum_{k=1}^K \mathbb{E}|\alpha_k(\omega)\phi_\gamma(\mathbf{p} - \mathbf{p}_k)|^2 = K \sum_{k=1}^K I_k(\mathbf{p}, \mathbf{p}, \omega).$$

We observe that $I_k(\mathbf{p}, \mathbf{p}', \omega) \geq 0$, and together with $\int_{\mathbb{S}} \phi_\gamma(\mathbf{p}) d\mathbf{p} = 1$, we obtain

$$\int_{\mathbb{S}} I_k(\mathbf{p}, \mathbf{p}) d\mathbf{p} \leq \int_{\mathbb{S}} \int_{\mathbb{S}} I_k(\mathbf{p}, \mathbf{p}') d\mathbf{p} d\mathbf{p}' = \sigma_k^2(\omega).$$

Together, these two last facts yield the proof. \square

Chapter 5

Adaptive Processing: The Recursive Hessian Sketch^{*}

‘Curiouser and curiouser!’ cried Alice.

Alice’s Adventures in Wonderland
LEWIS CARROLL

5.1 Introduction

In the first three chapters of this thesis we have described classic DSP problems where, loosely speaking, characteristics of the signal do not change over time. For example, the raking beamformers of Chapter 3 are computed assuming a static scene. The real world, however, changes. Sound sources appear, disappear, are moving over time. It is therefore necessary to adapt the processing. Naturally, algorithms that adapt to a changing environment are grouped under the label of adaptive signal processing, a cornerstone of classic statistical signal processing.

Adaptive filters are a prominent example of such algorithms. In this problem, a known signal x_n is fed into an unknown filter \mathbf{w}^* . The output of the filter is then compared to a desired response signal d_n , also available to the algorithm, but corrupted by noise. Given the driving signal and the desired response, the algorithm will maintain an estimate of the unknown filter. At every step, the output of the estimated filter is compared to the desired response and the filter is adjusted to make the mismatch as small as possible. This framework covers a large number of practical applications such as system identification, echo cancellation, channel equalization, and beamforming [60]. These play a critical role in hands-free telephony, teleconferencing, digital communications, and many other practical systems. The example of echo cancellation in telephone systems is described

^{*}This chapter is joint work with Martin Vetterli [115].

in Figure 5.1. Interestingly, the adaptive filter problem finds its roots in the training of early neural networks for classification [131]. For an interesting historical perspective on the invention of adaptive filters, see [130].

Two adaptive filters in particular have proved very popular: the least mean squares (LMS) and the recursive least squares (RLS) algorithms. On the one hand, LMS optimizes the mean squared error (MSE) using a stochastic gradient descent. On the other hand, RLS solves recursively a large least squares (LS) problem. While the former enjoys simplicity of implementation, low-complexity — linear in the filter length — and good stability properties, it sometimes lacks in terms of speed of convergence. The latter can offer a greater speed of convergence at the cost of a computational complexity quadratic in the filter length. This complexity can be alleviated to some extent by iterative methods such as conjugate gradient (CG) with clever exploitation of the data matrix structure [93].

In recent years, random projections have been shown to be an effective tool to reduce the size of large LS problems. The technique is often referred to as sketching and involves solving a smaller problem whose equations are random weighted sums of those in the original problem [18, 41, 103]. Of particular interest is the iterative Hessian sketch (IHS) algorithm proposed by Pilanci and Wainwright [103]. Unlike other methods, IHS does not sketch the whole problem, but only the Hessian of the associated quadratic cost function. The particularity of this algorithm is that the sketching can be iterated to refine the solution produced. With a logarithmic number of iterations, its solution can be made arbitrarily close to that of the original problem. This is in contrast with the conventional sketching algorithms that only offer sharp guarantees for the value of the residual. In a paper building on this initial work, Wang et al. identify the Hessian sketch as a preconditioned gradient descent algorithm where the preconditioner is obtained by sketching the Hessian of the LS cost function [128]. They propose in addition to apply the machinery of conjugate gradient (CG) to obtain an accelerated IHS (AccIHS) algorithm.

The sketching algorithms described so far, including IHS, act globally on the data and are thus not suitable in the streaming model of computation where each data point cannot be processed more than once. This model can apply to real-time data where only very small delays in processing can be tolerated, or to big data where the size of the dataset is simply too large to be read more than once. Adaptive filters were created precisely for streaming data and it is thus of interest to combine them with ideas from sketching. Initial work by Berberidis et al. pioneered the application of conventional sketching to streaming data with RLS-like algorithms [14]. Their algorithm adaptively censors input data based on an information criteria to decrease the complexity of RLS. Nonetheless, by sketching the whole RLS problem, this method suffers from the same limitation as conventional sketching: large censoring leads to poorer solution approximation.

In this chapter, we propose the recursive Hessian sketch (RHS) algorithm. It is a randomized, approximate version of the RLS algorithm using ideas from AccIHS to solve the underlying LS problem recursively. The benefit of this formulation is to allow a reduction in computational complexity at the cost of some convergence speed, thus bridging the gap between LMS and RLS. Rather than compute the exact inverse covariance matrix at every step, RHS keeps a sketch of the inverse Hessian matrix that is updated with probability q at each round. An updated solution is produced in block fashion at fixed intervals. At every update, a fixed number N of conjugate gradient iterations using the sketched inverse Hessian matrix as a preconditioner are run and a new filter is produced. Parameters N and q control the trade-off between complexity and convergence. By using conjugate gradient, our algorithm is guaranteed to converge to the RLS solution as n grows large. In addition, an asymptotic upper bound on the speed of convergence

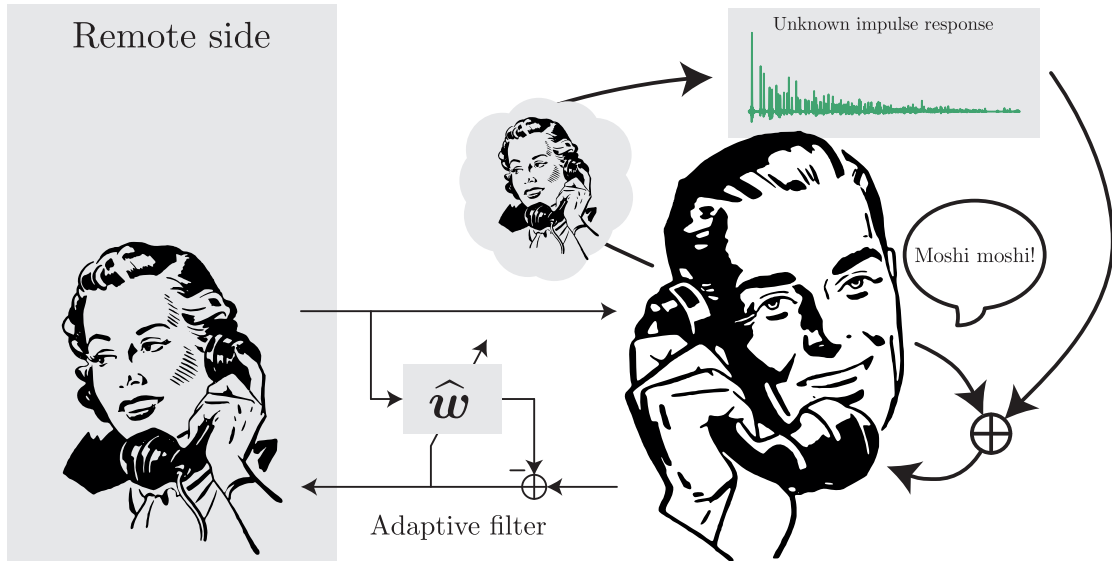


Figure 5.1: Illustration of echo cancellation using an adaptive filter. The distant speech acts as a reference signal and the local speech is the corrupting signal. The unknown impulse response is estimated as \hat{w} and as much as possible of the filtered distant speech is removed from the input before sending to the remote speaker. This residual is also used as a feedback to adapt the current estimate \hat{w} to possibly changing conditions of the environment.

to the RLS solution is derived under some simplifying assumptions. This upper bound is shown through numerical experiments to be accurate even down to n in the order of the filter length p . For small values of n , we observe however that our initial inverse Hessian sketch can lead to poorer convergence of CG than with no preconditioning at all. For this situation, we propose a different sketch with better preconditioning properties. This sketch approximates the covariance matrix of the data as Toeplitz symmetric — a good approximation for natural audio signals. We further apply a circulant approximation to this Toeplitz matrix to efficiently compute its inverse [27]. The performance of the algorithm with the different preconditioners proposed is evaluated through numerical simulations. Various driving signals are investigated: auto-regressive, moving-average, or natural music signals. Its performance is shown to be very close to that of vanilla RLS even with small values of q and N , leading to interesting complexity trade-offs.

The rest of this chapter is organized as follows. Section 5.2 introduces the necessary notation and background material on adaptive filters and sketching. Section 5.3 describes the proposed algorithm while Section 5.4 evaluates its complexity. The convergence of the algorithm is dealt with in Section 5.5. The results of numerical experiments are presented in Section 5.6. We conclude in Section 5.7.

5.2 Background

Throughout this chapter we denote all matrices by bold upper case and vectors by bold lower case letters. The time index is $n \in \mathbb{N}$ and the Euclidean norm operation is $\|\mathbf{x}\| = (\mathbf{x}^\top \mathbf{x})^{1/2}$. A

Table 5.1: Notation

n	The time index
p	The filter length
x_n	The n -th input sample
\mathbf{x}_n	Regression vector $\mathbf{x}_n = [x_n, \dots, x_{n-p+1}]^\top$
\mathbf{X}_n	The data matrix $\mathbf{X}_n = [\mathbf{x}_n \dots, \mathbf{x}_1]^\top$
$\mathbf{X}_{n,L}$	A chunk of the data matrix $\mathbf{X}_{n,L} = [\mathbf{x}_n \dots, \mathbf{x}_{n-L+1}]^\top$
d_n	The n -th noisy output sample
\mathbf{d}_n	Vector of noisy outputs $\mathbf{d}_n = [d_n, \dots, d_1]^\top$
$\mathbf{d}_{n,L}$	$\mathbf{d}_{n,L} = [d_n, \dots, d_{n-L+1}]^\top$
λ	The forgetting factor
$\mathbf{\Lambda}_n$	The exponential weight matrix $\mathbf{\Lambda}_n = \text{diag}(1, \lambda, \dots, \lambda^{n-1})$
$\ \mathbf{x}\ $	Euclidean norm $\ \mathbf{x}\ = (\mathbf{x}^\top \mathbf{x})^{1/2}$
$\ \mathbf{x}\ _{\mathbf{A}}$	\mathbf{A} -norm $\ \mathbf{x}\ _{\mathbf{A}} = \ \mathbf{A}\mathbf{x}\ $
\mathbf{R}_n	The Auto-correlation matrix in RLS $\mathbf{R}_n = \mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{X}_n$
\mathbf{b}_n	The cross-correlation vector in RLS $\mathbf{b}_n = \mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{d}_n$
\mathbf{P}_n	The inverse covariance matrix in RLS $\mathbf{P}_n = \mathbf{R}_n^{-1}$
$\tilde{\mathbf{R}}_n, \tilde{\mathbf{P}}_n$	Sketches of $\mathbf{R}_n, \mathbf{P}_n$
N	The number of iterations in IHS
q	The sketch update probability

summary of the most frequent notation in this chapter is given in Table 5.1.

5.2.1 Adaptive Filters

The adaptive filtering problem aims at finding an estimator of an unknown filter $\mathbf{w}^* \in \mathbb{R}^p$, of length p , from a known reference signal x_n , and its filtered samples corrupted by noise $d_n = (x \star h)_n + v_n$, where $v_n \sim \mathcal{N}(0, \sigma_v^2)$ is additive white Gaussian noise, and \star is the discrete convolution operation. This can be compactly expressed in vector form by defining a vector containing the last p samples

$$\mathbf{x}_n = [x_n, x_{n-1}, \dots, x_{n-p+1}]^\top,$$

and the corrupted output signal becomes

$$d_n = \mathbf{x}_n^\top \mathbf{w}^* + v_n.$$

Then the adaptive filtering problem is mathematically formulated as the repeated optimization of a well-chosen cost-function,

$$\hat{\mathbf{w}}_n = \arg \min_{\mathbf{w}} J(\mathbf{w}; d_n, \dots, d_1, x_n, \dots, x_1),$$

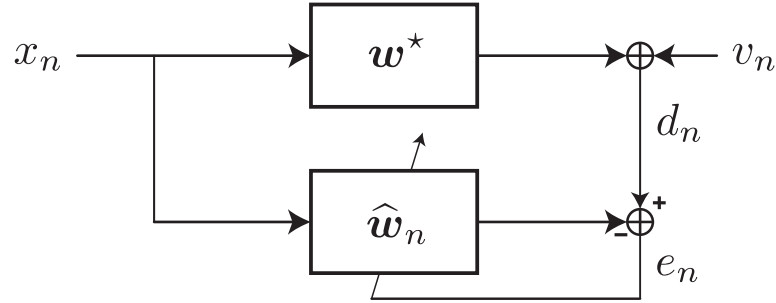


Figure 5.2: Block-diagram of the adaptive filtering problem in the system identification setting. The driving signal x_n is filtered through both the unknown system w^* and its current estimate \hat{w}_n . The output from the estimate is compared to the noisy output of the unknown system d_n , the reference signal. The adaptive filter algorithm then corrects \hat{w}_n to minimize the discrepancy.

where \hat{w}_n is the estimate of w^* at time n . The choice of the cost function along with the method for solving the optimization problem define an adaptive filtering algorithm. The system is illustrated in Figure 5.2.

Two popular instantiations of adaptive filters are the LMS and RLS algorithms [60]. These two algorithms have very distinct philosophies. The LMS filter finds the optimal filter in expectation. It converges to the Wiener filter, but without the need of knowing the statistics of the input signal. Alternatively, RLS follows a data fitting approach by finding the filter that minimizes a weighted mean-squared error between observed inputs and outputs.

Least-Mean Squares

The LMS algorithm takes a stochastic approach to the problem. It tries to find a solution optimal in expectation, much like the classic Wiener filter, but without the need to know the statistics of the signal. The LMS algorithm optimizes the expected squared error

$$J_{\text{LMS}}(\mathbf{w}) = \frac{1}{2} \mathbb{E} |d_n - \mathbf{x}_n^\top \mathbf{w}|^2.$$

One could decide to minimize this quantity by a gradient descent, moving the solution step by step in the direction of the negative gradient. The gradient of the previous quantity can be computed analytically

$$\nabla J_{\text{LMS}}(\mathbf{w}) = -\mathbb{E}[\mathbf{x}_n(d_n - \mathbf{x}_n^\top \mathbf{w})],$$

and this leads to the following update for the filter estimate at step $n + 1$

$$\hat{\mathbf{w}}_{n+1} = \hat{\mathbf{w}}_n - \mu \nabla J_{\text{LMS}}(\hat{\mathbf{w}}_n) = \hat{\mathbf{w}}_n + \mu \mathbb{E}[\mathbf{x}_n(d_n - \mathbf{x}_n^\top \hat{\mathbf{w}}_n)],$$

where μ is the step size. In practice, however, the value of the expectation is not known and must be estimated from the available signal. The simplest method is to use the point estimate

$$\mathbb{E}[\mathbf{x}_n(d_n - \mathbf{x}_n^\top \mathbf{w})] \approx \mathbf{x}_n(d_n - \mathbf{x}_n^\top \mathbf{w}),$$

which leads to the LMS update

$$\hat{\mathbf{w}}_{n+1} = \hat{\mathbf{w}}_n + \mu \mathbf{x}_n (d_n - \mathbf{x}_n^\top \hat{\mathbf{w}}_n).$$

The normalized LMS (NLMS) algorithm solves the problem of choosing the value of μ by using the adaptive step size $\mu = \mu' / (\mathbf{x}^\top \mathbf{x})$. In the absence of noise setting $\mu' = 1$ leads to the optimal learning rate [60].

Recursive Least-Squares

In contrast to LMS, the RLS algorithm takes a data fitting approach. The cost function it minimizes is a weighted sum of squares of the error function with Tikhonov regularization,

$$J_{\text{RLS}}(\mathbf{w}) = \sum_{i=1}^n \lambda^{n-i} |d_i - \mathbf{x}_i^\top \mathbf{w}|^2 + \lambda^n \delta \|\mathbf{w}\|^2,$$

where λ is an exponential forgetting factor allowing the algorithm to adapt to a time-varying filter. The regularization allows the algorithm to produce a solution when $n < p$, that is fewer data points than dimensions have been observed¹. As more and more data points are observed, the regularization is exponentially quickly forgotten. This cost function can be compactly written using matrix formalism

$$J_{\text{RLS}}(\mathbf{w}) = \left\| \mathbf{\Lambda}_n^{1/2} (\mathbf{X}_n \mathbf{w} - \mathbf{d}_n) \right\|^2 + \lambda^n \delta \|\mathbf{w}\|^2, \quad (5.1)$$

where

$$\begin{aligned} \mathbf{X}_n &= [\mathbf{x}_n \cdots \mathbf{x}_1]^\top, \\ \mathbf{d}_n &= [d_n, \dots, d_1]^\top, \\ \mathbf{\Lambda}_n &= \text{diag}(1, \dots, \lambda^{n-1}). \end{aligned}$$

The minimization of (5.1) is a LS problem that admits the analytical solution

$$\hat{\mathbf{w}}_n = \arg \min_{\mathbf{w}} J_{\text{RLS}}(\mathbf{w}) = (\mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{X}_n + \lambda^n \delta \mathbf{I}_p)^{-1} \mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{d}_n,$$

where \mathbf{I}_p is the $p \times p$ identity matrix. For convenience, we define the sample covariance matrix, and cross-covariance vector

$$\begin{aligned} \mathbf{R}_n &= \mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{X}_n + \lambda^n \delta \mathbf{I}_p, \\ \mathbf{b}_n &= \mathbf{X}_n^\top \mathbf{\Lambda}_n \mathbf{d}_n. \end{aligned}$$

These two quantities can be computed recursively from their value at $n-1$, \mathbf{w}_{n-1} , \mathbf{x}_n , and d_n . Since $\mathbf{X}_n = [\mathbf{x}_n \ \mathbf{X}_{n-1}^\top]^\top$ and $\mathbf{d}_n = [d_n \ \mathbf{d}_{n-1}^\top]^\top$, one can check that

$$\begin{aligned} \mathbf{R}_n &= \lambda \mathbf{R}_{n-1} + \mathbf{x}_n^\top \mathbf{x}_n, \\ \mathbf{b}_n &= \lambda \mathbf{b}_{n-1} + d_n \mathbf{x}_n. \end{aligned}$$

¹An alternative solution is to wait until $n \geq p$ samples have been collected to produce the first solution.

The final form of the RLS algorithm is obtained by computing the inverse of \mathbf{R}_n using the recursion above and the matrix inversion lemma [133]

$$\mathbf{P}_n = \mathbf{R}_n^{-1} = \lambda^{-1} \mathbf{P}_{n-1} - \lambda^{-1} \frac{\mathbf{P}_{n-1} \mathbf{x}_n \mathbf{x}_n^\top \mathbf{P}_{n-1}}{\lambda + \mathbf{x}_n^\top \mathbf{P}_{n-1} \mathbf{x}_n}.$$

A little algebra yields the final form of the algorithm as described in Algorithm 5.1².

Algorithm 5.1 BlockRLS

Require: $\lambda, \delta, \mathbf{P}_0 = \delta^{-1} \mathbf{I}_d, \mathbf{w}_0 = 0$

Ensure: $\mathbf{w}_n = \arg \min_{\tilde{\mathbf{w}}} \|\mathbf{\Lambda}_n^{1/2} (\mathbf{X}_n \tilde{\mathbf{w}} - \mathbf{d}_n)\|_2^2$

for every L new samples do:

$\mathbf{Z} \leftarrow \mathbf{P}_n \mathbf{X}_{n,L}^\top$

$\mathbf{G} \leftarrow \lambda^{-L} \mathbf{Z} (\mathbf{\Lambda}_L^{-1} + \lambda^{-L} \mathbf{X}_{n,L} \mathbf{Z})^{-1}$

$\mathbf{w}_{n+L} \leftarrow \mathbf{w}_n + \mathbf{G} (\mathbf{d}_{n,L} - \mathbf{X}_{n,L} \mathbf{w}_n)$

$\mathbf{P}_{n+L} \leftarrow \lambda^{-L} (\mathbf{P}_n - \mathbf{G} \mathbf{Z}^\top)$

5.2.2 Least-Squares Sketching

The other element of this work is a method known as *sketching* used to reduce the size of large-scale LS problems. Consider the following generic constrained LS problem

$$\min_{\mathbf{x} \in \mathcal{C}} \frac{1}{2n} \|\mathbf{A} \mathbf{x} - \mathbf{y}\|^2 \quad (5.9)$$

where the matrix \mathbf{A} is $n \times p$, with $n \gg p$, and \mathcal{C} is a constraint set. Solving this problem involves computing the Gram matrix $\mathbf{A}^\top \mathbf{A}$ which has complexity $O(p^2 n)$. When n is very large, this term dominates the computation cost. The sketching method reduces the number of rows by premultiplying by a matrix $\mathbf{S} \in \mathbb{R}^{m \times n}$, with $m = o(n)$, and solving the new problem

$$\min_{\mathbf{x} \in \mathcal{C}} \frac{1}{2n} \|\mathbf{S} (\mathbf{A} \mathbf{x} - \mathbf{y})\|^2. \quad (5.10)$$

Note that the Gram matrix computation only requires $O(p^2 m)$ now. If m is substantially smaller than n a large computational gain is achieved. Before that can be claimed, two facts should be established. First, how close is the solution (5.10) to that of (5.9). Second, the cost of multiplication by the sketching matrix should not offset the gain in the Gram matrix computation. These questions are answered by Drineas et al. [41] in details. When \mathbf{S} is the fast Johnson-Lindenstrauss transform [3], they provide the following guarantee.

Theorem 5.1 (Classic LS Sketch [41])

Let \mathbf{x}^* and $\tilde{\mathbf{x}}$ be the minimizers of (5.9) and (5.10), respectively, and $\epsilon \in (0, 1)$. Then, with probability at least 0.8

$$\|\mathbf{A} \tilde{\mathbf{x}} - \mathbf{y}\| \leq (1 + \epsilon) \|\mathbf{A} \mathbf{x}^* - \mathbf{y}\|.$$

Furthermore

$$\|\tilde{\mathbf{x}} - \mathbf{x}^*\| \leq \sqrt{\epsilon} \kappa(\mathbf{A}) C \|\mathbf{x}^*\|, \quad (5.11)$$

where $\kappa(\mathbf{A})$ is the condition number of \mathbf{A} and C is a constant that depends on \mathbf{A} and \mathbf{y} .

²Algorithm 5.1 is block RLS with $\mathbf{X}_{n,L} = [\mathbf{x}_{n+L} \cdots \mathbf{x}_{n+1}]^\top$ and $\mathbf{d}_{n,L} = [d_{n+L}, \dots, d_{n+1}]^\top$. Setting $L = 1$ yields standard RLS.

Furthermore, the fast Johnson-Lindenstrauss transform can be applied efficiently with a cost of $O(pn \log n)$ for the sketching step. While this algorithm provides tight guarantees on the optimality of the objective function, the solution approximation bound of (5.11) is plagued by the condition number of \mathbf{A} and other constants. In fact, Pilanci and Wainwright demonstrate that the solution of (5.10) does not converge to that of the original problem, unless the sketching dimension $m \simeq n$, the original problem size [103].

Iterative Hessian Sketch

The iterative Hessian sketch (IHS) is a recently proposed LS sketching algorithm addressing the shortcomings of the classic sketch as described above [103]. Rather than sketching the whole system, as in (5.10), it considers the equivalent problem

$$\min_{\mathbf{x} \in \mathcal{C}} \frac{1}{2} \mathbf{x}^\top \mathbf{A}^\top \mathbf{A} \mathbf{x} - \mathbf{y}^\top \mathbf{A} \mathbf{x}.$$

The Hessian sketch is then obtained by sketching only the quadratic term (the Hessian), rather than the whole objective,

$$\min_{\mathbf{x} \in \mathcal{C}} \frac{1}{2} \mathbf{x}^\top (\mathbf{S}\mathbf{A})^\top (\mathbf{S}\mathbf{A}) \mathbf{x} - \mathbf{y}^\top \mathbf{A} \mathbf{x}.$$

Compared to the classic sketch, the Hessian sketch has the nice property that it can be iterated to obtain ever more precise refinements of the solution. This iterative process is the iterative Hessian sketch (IHS) algorithm. In the rest of this chapter, we only consider the unconstrained case, i.e. $\mathcal{C} = \mathbb{R}^d$, and the IHS algorithm reduces to the following iterative process

$$\hat{\mathbf{x}}_i = \hat{\mathbf{x}}_{i-1} + (\mathbf{A}^\top \mathbf{S}_i^\top \mathbf{S}_i \mathbf{A})^{-1} \mathbf{A}^\top (\mathbf{y} - \mathbf{A} \mathbf{x}_{i-1}), \quad i = 1, \dots, N,$$

where $\hat{\mathbf{x}}_0 = \mathbf{0}$ and $\{\mathbf{S}_i \in \mathbb{R}^{m \times n}\}_{i=1}^N$, $m \leq n$, are sketching matrices drawn independently at random. Sketching matrices can be, for example, matrices with normal iid entries, the fast Johnson-Lindenstrauss transform [3], or a matrix sampling the rows of \mathbf{A} at random. Their key property is to preserve the norms of the columns of \mathbf{A} up to some controlled distortion. They must satisfy $\mathbb{E}[\mathbf{S}^\top \mathbf{S}] = \mathbf{I}_n$. The convergence of the method for unconstrained LS problems is formulated in Corollary 2 of [103].

Proposition 5.1 (Corollary 2 in [103])

Let \mathbf{x}_{LS} be the LS solution to (5.9). For some $\rho \in (0, 1/2)$, suppose we run the IHS for

$$N = 1 + \left\lceil \frac{\log \sqrt{n} \frac{\|\mathbf{x}_{LS}\|_{\mathbf{A}}}{\sigma}}{\log(1/\rho)} \right\rceil$$

iterations using $m = \frac{c_0}{\rho^2} p$ projections per round. Then the output $\hat{\mathbf{x}}_N$ satisfies the bounds

$$\|\hat{\mathbf{x}}_N - \mathbf{x}_{LS}\|_{\mathbf{A}} \leq \sqrt{\frac{\sigma^2 d}{n}}.$$

A recent publication by Wang et al. [128] shed some light on the nature of IHS by recognizing it as a form of preconditioned gradient descent algorithm. In the same work, it is also noted

that the method can in fact diverge when the conditions on the sketching dimension outlined in the proposition above are not met. They remedy to this situation by applying the machinery of Conjugate Gradient (CG) to IHS. The resulting Accelerated IHS (AccIHS) algorithm is faster, requiring less iterations to reach the same precision as IHS, always convergent, and has the same overall complexity.

5.3 The Recursive Hessian Sketch Algorithm

The idea of the recursive Hessian sketch (RHS) is to apply the ideas of Hessian sketching to solve (5.1) recursively as new data streams in. Motivated by insights of Wang et al. [128], we describe a general algorithm that solves the RLS system using CG, then we propose two randomized preconditioners to improve the convergence rate of CG. The preconditioners are obtained by sketching the Hessian of (5.1).

The algorithm proposed is a block update algorithm. A new filter estimate is produced once for every block of L successive samples. At time n , the RLS solution is obtained by solving the system $\mathbf{R}_n \mathbf{w} = \mathbf{b}_n$, as described in Section 5.2.1. In conventional RLS, this system is solved exactly by inversion of \mathbf{R}_n . Instead, we will solve this system approximately by running a fixed number of N iterations of CG. The speed of convergence of CG applied to this minimization can be improved by starting from the previous estimate of the filter $\hat{\mathbf{w}}_{n-1}$ and adding a preconditioner $\tilde{\mathbf{P}}_n$. The modified linear system is

$$\tilde{\mathbf{P}}_n \mathbf{R}_n (\mathbf{w} + \hat{\mathbf{w}}_{n-1}) = \tilde{\mathbf{P}}_n \mathbf{b}_n.$$

The CG algorithm will then take N steps starting from $\hat{\mathbf{w}}_{n-1}$ in directions $\mathbf{p}_1, \dots, \mathbf{p}_N$ satisfying the conjugate condition

$$\mathbf{p}_i^\top \tilde{\mathbf{P}}_n \mathbf{R}_n \mathbf{p}_j = 0, \quad \forall i \neq j.$$

The new filter estimate produced is thus $\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_{n-1} + \sum_{i=1}^N \mathbf{p}_i$. The resulting algorithm is described in Algorithm 5.2. In the simplest case, we can choose $\tilde{\mathbf{P}} = \mathbf{I}_p$, i.e. CG without preconditioning. This will serve as a base line when evaluating the performance of the preconditioners. According to CG convergence results, the convergence speed of this algorithm depends on the condition number $\kappa(\tilde{\mathbf{P}}_n \mathbf{R}_n)$ which we analyze in Section 5.5.

Unlike classic RLS, RHS needs to keep track of the forward matrix \mathbf{R}_n , in addition to the sketches $\tilde{\mathbf{P}}$'s and the right-hand side \mathbf{b}_n . Fortunately, this can be done efficiently for blocks of fixed size $B = O(d)$ using the Toeplitz matrix-vector multiplication algorithm described in Appendix 5.B.

5.3.1 Row Sampling Preconditioner

One can recover conventional RLS from RHS by using \mathbf{R}_n^{-1} as preconditioner. Then, CG converges in exactly one iteration. It is however costly to compute \mathbf{R}_n^{-1} exactly. Adopting a stochastic point of view, \mathbf{R}_n is nothing but an estimate of the covariance matrix of the process x_n . Intuitively, it would then make sense that we would not lose much by using a different estimate of the inverse covariance matrix, one requiring less computations, as a preconditioner. A very simple idea is to use less data points to construct our estimate of the inverse covariance matrix. In fact, we will construct our first preconditioner by including new data in the estimate with probability q .

Algorithm 5.2 Recursive Hessian Sketch (RHS)**Require:** $\lambda, \delta, N, q, \mathbf{R} = \delta \mathbf{I}_d, \hat{\mathbf{w}}, \mathbf{b} = 0, L$ **Ensure:** $\mathbf{w} = \arg \min_{\tilde{\mathbf{w}}} \|\Lambda_n^{1/2} (\mathbf{X}_n \tilde{\mathbf{w}} - \mathbf{d}_n)\|_2^2$

// Update covariance and cross-covariance

if $n \bmod B = 0$ **then**

$$\mathbf{R} \leftarrow \lambda^B \mathbf{R} + \mathbf{X}_{n,B}^\top \Lambda_B \mathbf{X}_{n,B}$$

$$\mathbf{b} \leftarrow \lambda^B \mathbf{b} + \mathbf{X}_{n,B}^\top \Lambda_B \mathbf{d}_{n,B}$$

end if

// Update the preconditioner

$$\tilde{\mathbf{P}} \leftarrow \text{update_preconditioner}(\tilde{\mathbf{P}}, \mathbf{x}_n)$$

// Run CG iterations

if $n \bmod L = 0$ **then**

$$\mathbf{r} \leftarrow -(\mathbf{b} - \mathbf{R}\hat{\mathbf{w}})$$

$$\mathbf{p} \leftarrow \mathbf{0}$$

for $i = 1, \dots, N$ **do**

// Apply preconditioner

$$\mathbf{u} \leftarrow \tilde{\mathbf{P}}_i \mathbf{r}$$

// Compute conjugate direction

$$\beta \leftarrow \frac{\mathbf{r}^\top \mathbf{u}}{\|\mathbf{r}\|^2}$$

$$\mathbf{p} \leftarrow -\mathbf{u} + \beta \mathbf{p}$$

$$\mathbf{v} \leftarrow \mathbf{R} \mathbf{p}$$

$$\alpha \leftarrow \frac{\mathbf{r}^\top \mathbf{u}}{\mathbf{p}^\top \mathbf{v}}$$

$$\mathbf{r} \leftarrow \mathbf{r} + \alpha \mathbf{v}$$

$$\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}} + \alpha \mathbf{p}$$

end for**end if****Algorithm 5.3** update_preconditioner (Row Sampling)**Require:** $\tilde{\mathbf{P}}, \mathbf{x}_n, q, \lambda, k_p = 0$ **Ensure:** Preconditioner $\tilde{\mathbf{P}}$ is updated with new data \mathbf{x}_n

// Random update of inverse sketched matrices

Draw at random $b \stackrel{\text{iid}}{\sim} \text{Bernoulli}(q)$ **if** $b = 1$ **then**

// The time since previous update

$$C \leftarrow n - k_p$$

$$k_p \leftarrow n$$

// Rank-1 update of the inverse

$$\mathbf{z} \leftarrow \tilde{\mathbf{P}} \mathbf{x}_n$$

$$\tilde{\mathbf{P}} \leftarrow \lambda^{-C} \left(\tilde{\mathbf{P}} - \frac{\mathbf{z} \mathbf{z}^\top}{q \lambda^C + \mathbf{x}_n^\top \mathbf{z}} \right)$$

end if

This preconditioner can be built efficiently using the same type of rank-1 updates as RLS. The construction is most easily described by multiplying the data matrix \mathbf{X}_n by a diagonal sketching matrix with Bernoulli random variables on the diagonal

$$\mathbf{S}_n = \begin{bmatrix} b_n/\sqrt{q} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{n-1} \end{bmatrix}, \quad b_n \sim \begin{cases} 1 & \text{w.p. } q \\ 0 & \text{w.p. } 1 - q \end{cases}. \quad (5.12)$$

Note that the random variables are normalized so that $\mathbb{E}[\mathbf{S}_n^\top \mathbf{S}_n] = \mathbf{I}$. Assuming L samples arrived since the previous update, the recursive update for the sketched covariance matrix is

$$\begin{aligned} \tilde{\mathbf{R}}_n &= \mathbf{X}_n^\top \mathbf{\Lambda}_n^{1/2} \mathbf{S}_n^\top \mathbf{S}_n \mathbf{\Lambda}_n^{1/2} \mathbf{X}_n \\ &= q^{-1} \mathbf{x} \mathbf{x}^\top + \lambda^L \tilde{\mathbf{R}}_{n-L}, \end{aligned}$$

and using again the matrix inversion lemma, as in RLS, the inverse matrix update takes the form

$$\tilde{\mathbf{P}}_n = \lambda^{-L} \left(\tilde{\mathbf{P}}_{n-L} - \frac{\tilde{\mathbf{P}}_{n-L} \mathbf{x}_n \mathbf{x}_n^\top \tilde{\mathbf{P}}_{n-L}}{q\lambda^L + \mathbf{x}_n^\top \tilde{\mathbf{P}}_{n-L} \mathbf{x}_n} \right). \quad (5.15)$$

Algorithmically, this can be implemented in the following way. For every new sample received, a Bernoulli random variable $b \sim \text{bern}(q)$ is drawn independently. Then, if $b = 1$, we update the $\tilde{\mathbf{P}}$ according to (5.15), otherwise we do nothing. Pseudocode for the row sampling preconditioner update is given in Algorithm 5.3.

As mentioned earlier, a good estimate of the inverse covariance matrix of the process x_n would make a good preconditioner. We confirm this intuition in Section 5.5 where we analyze the convergence of Algorithm 5.2 with this preconditioner. While behaving very well asymptotically, when very few data is available at the beginning of the algorithm, the estimator built here can be rather poor and degrade convergence. To mitigate this, we develop in the next section a preconditioner that is good right from the start.

5.3.2 Circulant Preconditioner

A weakness of the row sampling preconditioner is that if no regularization is used (i.e. $\delta = 0$), it needs at least p updates to attain full rank. Even when $\delta \neq 0$, the λ^n term means that the influence of regularization decreases as the algorithm progress, and if q is not large enough, this could lead to a large condition number and imperil the fast convergence of CG.

A solution is to constrain the covariance to a special form where a well conditioned estimator is available right from the beginning. When x_n is a real-valued wide sense stationary (WSS) process, its covariance matrix is Toeplitz symmetric,

$$\mathbf{R} = \begin{pmatrix} r_0 & r_1 & \cdots & r_{p-1} \\ r_1 & r_0 & \cdots & r_{p-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & \cdots & r_0 \end{pmatrix},$$

where $r_k = \mathbb{E}[x_n x_{n+k}]$ is the auto-correlation of the process. We know that an effective preconditioner should be close to the inverse covariance matrix of the process. We can thus exploit the

Algorithm 5.4 update_preconditioner (Circulant)

Require: $\tilde{\mathbf{P}}, \mathbf{x}_n, q, \lambda, k_p = 0, \mathbf{r} = [\delta, 0, \dots, 0]^\top$
Ensure: Preconditioner $\tilde{\mathbf{P}}$ is updated with new data \mathbf{x}_n
 // Random update of inverse sketched matrices
 Draw at random $b \stackrel{\text{iid}}{\sim} \text{Bernoulli}(q)$
if $b = 1$ **then**
 // The time since previous update
 $C \leftarrow n - k_p$
 $k_p \leftarrow n$
 // Estimate auto-correlation
for $i = 0, \dots, p - 1$ **do**
 $r_i \leftarrow \lambda^C r_i + \frac{1}{q(p-1)} \sum_{j=n-p+1}^{n-i} x_j x_{j+i}$
end for
 // Circulant approximation
for $i = 0, \dots, p - 1$ **do**
 $c_i \leftarrow \frac{i\tilde{r}_{p-i} + (p-i)r_i}{p}$
end for
 // Compute the inverse
 $\tilde{\mathbf{P}} \leftarrow \text{Circulant}(\text{iFFT}(1/\text{FFT}(\mathbf{c})))$
end if

structure to find a better preconditioner in the beginning of the algorithm. Ideally this preconditioner would be both close to the inverse covariance matrix and computationally efficient.

The auto-correlation can be estimated from any vector \mathbf{x}_n using for example the following estimator

$$r_n[k] = \frac{1}{p-1} \sum_{i=n-p+1}^{n-k} x_i x_{i+k}, \quad k = 0, \dots, p-1,$$

which we write as the vector $\mathbf{r}_n = [r_n[0], \dots, r_n[p-1]]^\top$. Adding the forgetting factor and the random sampling, similar to (5.12), we obtain the following sketched auto-correlation

$$\tilde{\mathbf{r}}_n = \sum_{k=1}^n \frac{b_k}{q} \lambda^{n-k} \mathbf{r}_k,$$

and the sketched covariance matrix $\tilde{\mathbf{R}}_n$ is formed by constructing a Toeplitz matrix from $\tilde{\mathbf{r}}_n$.

The crux here is that the algorithm requires $\tilde{\mathbf{P}}_n = \tilde{\mathbf{R}}_n^{-1}$, and thus an efficient inversion method is needed. Luckily, Toeplitz matrices are well-approximated by circulant matrices. Circulant matrices have the nice property to be diagonalized by the DFT matrix, and thus a fast inversion algorithm using the FFT exists. The optimal circulant approximation of \mathbf{R} in the Frobenius norm can be readily computed [27]. It is the circulant matrix $\tilde{\mathbf{C}}$ with first column $\tilde{\mathbf{c}} = [\tilde{c}_1, \dots, \tilde{c}_{p-1}]^\top$ with

$$\tilde{c}_i = \frac{i\tilde{r}_{p-i} + (p-i)\tilde{r}_i}{p}.$$

The product $\tilde{\mathbf{C}}^{-1}\mathbf{v}$ can now be computed efficiently by taking the FFTs of \mathbf{v} and $\tilde{\mathbf{c}}$, dividing element wise the former by the latter, and computing the iFFT of the result.

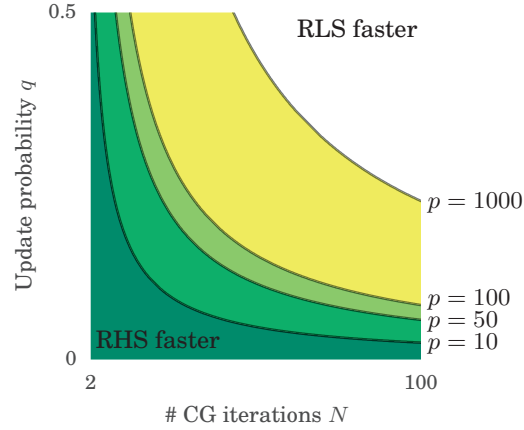


Figure 5.3: The shaded areas indicate where RHS has lower complexity than block RLS for various values of d .

5.4 Complexity Analysis

RHS is a block update algorithm. In general, performing block updates allows some computational gain compared to updating at every sample. For this reason, we compare the complexity of RHS to that of the block RLS described in Algorithm 5.1. Although block RLS only updates the filter every L sample, its output is exactly that of conventional RLS at the corresponding sample. The main advantage of block update is to make use of fast multiplication by $\mathbf{X}_{n,L}$. Since it is an $L \times p$ Hankel matrix, multiplying a vector by $\mathbf{X}_{n,L}$, or its transpose, has complexity $O(L \log p)$. This can be done by flipping upside-down the rows to obtain a Toeplitz matrix, take its circulant extension and use the FFT algorithm to compute the product. See Appendix 5.B for the details. For the comparison, we only establish an approximate count of operations. This is justified for several reasons. First, both algorithms use the same primitives, that is matrix-vector and matrix-matrix products as well as FFT, and thus the big- O constants should be approximately the same for both algorithms. Second, the exact runtime of these primitives is highly dependent on the implementation and the architecture of the machine used. Our goal here is to show that there is an interesting regime for the RHS algorithm.

Let us start with the Block RLS complexity. Counting from Algorithm 5.1 and dividing by the block size L , we obtain

$$C_{\text{RLS}} = O \left(L + p + (L + p + 1) \log p + Lp + L^2 + p^2 + 2 \frac{p^2}{L} \right), \quad (5.16)$$

where p is the dimension of the filter. We assumed a naive algorithm for the multiplication by non-Toeplitz matrices. Notice the quadratic term in p , independent of the block size. This means that the asymptotic complexity of block RLS is identical to that of conventional RLS.

We can do a similar count for Algorithm 5.2 with the row sampling preconditioner described in Algorithm 5.3. Since the algorithm is not deterministic, we will use the average complexity. When the number of samples is large, the actual complexity should be very close to its expectation. Let us first compute the update probability p . At each round, at least one of the sketching matrices

is updated with probability q . By choosing the block size to be $O(q^{-1})$, the inverse sketch matrix will be updated on average once between two filter updates. Adding the N iterations of CG run once per block, we obtain the following average complexity per sample

$$C_{\text{RHS}} = O(q(4p^2 + 2p) + q((2N + 1)p^2 + (11N + 1)p) + (p + 1)\log p + 2p).$$

Figure 5.3 shows regions of the (N, q) space where RHS has a lower computational complexity than block RLS. For RLS, we chose the block size L minimizing (5.16). We observe that even for large filter lengths p there are significant regions where a gain can be obtained. In the following section, numerical experiments will reveal that these regions are compatible with a fast convergence rate.

5.5 Convergence Analysis of Accelerated RHS

In this section, we present an analysis of the convergence of Algorithm 5.2 with row sampling preconditioner. Contrary to the analysis of IHS [103] and AccIHS [128] algorithms, we base the analysis on stochastic properties of the input data. In our case, the simplicity of the sketching procedure cannot ensure success for arbitrary data matrix \mathbf{X} . To see this, imagine that \mathbf{X} is full rank, but has a large subset of linearly dependent rows. In a worst case scenario, the sketched matrix could be rank deficient, and thus not invertible.

The proposed algorithm approximates the RLS procedure by running a few iterations of CG with a preconditioner built by sketching and inverting the data matrix, as described in the previous section. While it is clear that CG will always converge given sufficiently many iterations are run, we would like to have extra guarantees on the rate of convergence. In particular, we would like to ensure that a few iterations is sufficient for fast convergence. We are able to analyze the rate of convergence under some restrictions on the parameters of the algorithm. To simplify the analysis, we make the following assumptions.

A1 The forgetting factor is $\lambda = 1^3$

A2 The rows of \mathbf{X} are independently drawn from a normal distribution, that is $\mathbf{x}_n \stackrel{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma} \in \mathbb{R}^{p \times p}$ is the covariance matrix of the process

The first assumption is made because the exponential weighting matrix makes the spectral structure of the data matrix difficult to analyze. The second assumption will allow us to make use of the Marčenko-Pastur law, a powerful theorem on the eigenstructure of large random matrices. In the algorithm described in the previous section, the data matrix \mathbf{X} has a shift structure and its rows are thus not independent. While the assumption is thus violated by the shift structure of the RLS data matrix, the numerical experiments from Section 5.6 confirm that this is not a problem in practice.

Under these assumptions, we are able to upper bound the convergence rate of Algorithm 5.2 to the RLS solution. Interestingly, the upper bound does not depend on the original correlation in the data, but only on the problem dimensions n and p , and the algorithm parameters q and δ .

³This corresponds to the so-called *growing window* RLS algorithm [60].

Theorem 5.2 (Convergence of RHS with Row Sampling Preconditioner)

Let $\mathbf{x}_i \stackrel{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \Sigma)$, $i = 1, \dots, n$, be independent vectors drawn from a normal distribution and stacked in the data matrix $\mathbf{X} = [\mathbf{x}_n, \dots, \mathbf{x}_1]^\top$. Let m rows of \mathbf{X} be selected for inclusion in the sketch and let $q = m/n$. Then, as n goes to infinity, the solution produced by Algorithm 5.2 run for N iterations of CG with the row sampling preconditioner of Algorithm 5.3, $\lambda = 1$ and regularization parameter δ satisfies

$$\|\mathbf{w}_n^{\text{RLS}} - \mathbf{w}_n^{\text{RHS}}\|_{\mathbf{X}} \leq 2 \left(\frac{\sqrt{\kappa(n, p, q, \delta)} - 1}{\sqrt{\kappa(n, p, q, \delta)} + 1} \right)^N \|\mathbf{w}_n^{\text{RLS}}\|_{\mathbf{X}},$$

where

$$\kappa(n, p, q, \delta) = \frac{q + (1 - q) \left(\frac{\delta}{qn} + \left(1 + \sqrt{\frac{p}{qn}}\right)^2 \right) \left(\frac{\delta}{(1-q)n} + \left(1 + \sqrt{\frac{p}{(1-q)n}}\right)^2 \right)}{q + (1 - q) \left(\frac{\delta}{qn} + \left(1 - \sqrt{\frac{p}{qn}}\right)^2 \right) \left(\frac{\delta}{(1-q)n} + \left(1 - \sqrt{\frac{p}{(1-q)n}}\right)^2 \right)}.$$

The proof of the theorem is given in Appendix 5.A. While the bound holds for asymptotically large values of n , we have run numerical experiments to verify its behavior in practice. Figure 5.4 shows the average value of

$$\rho = \left(\frac{\sqrt{\kappa(\tilde{\mathbf{P}}\mathbf{R})} - 1}{\sqrt{\kappa(\tilde{\mathbf{P}}\mathbf{R})} + 1} \right)$$

as n increases for different signal models and choices of the preconditioner $\tilde{\mathbf{P}}$. We compare the two preconditioners proposed, row sampling and circulant, to using no preconditioner at all, or using the true inverse covariance matrix. We also compare independent regression vectors to generating the matrix \mathbf{X} with the shift structure of the RLS algorithm. Figure 5.4 shows the evolution of the average of ρ for 50 realization of the data matrix according to four different signal models. We make several observations from the result of this experiment.

First, an obvious observation is that the structure of the covariance matrix is important. For a forward matrix close to identity, as is the case for white noise or moving average order 1 (MA(1)) driving signals, a larger number of rows needs to be added to the row sampling preconditioner before it becomes better than running the algorithm with no preconditioning. On the other hand, when the inverse matrix is close to a diagonal one, as is the case for auto-regressive order 1 (AR(1)) process, for example, then the advantage of preconditioning is significant. Secondly, in the experiment, the circulant preconditioner is on the one hand more effective in the beginning of the algorithm than the row sampling preconditioner. But, on the other hand, its effectiveness does not improve as n grows large, likely due to the circulant approximation involved. The row sampling preconditioner does not suffer from this trouble and ρ is guaranteed by the upper bound to asymptotically go to zero with n . Finally, we compare using iid regression vectors with the shift structure. For the ARMA type of signals used in the experiment, we observe that there is little difference in the value of ρ between the two. This is good news and we'll be further reassured when we'll observe the algorithm working well in numerical experiments in the next section.

As a final remark, observe that the theorem above holds for a deterministic number of sampled

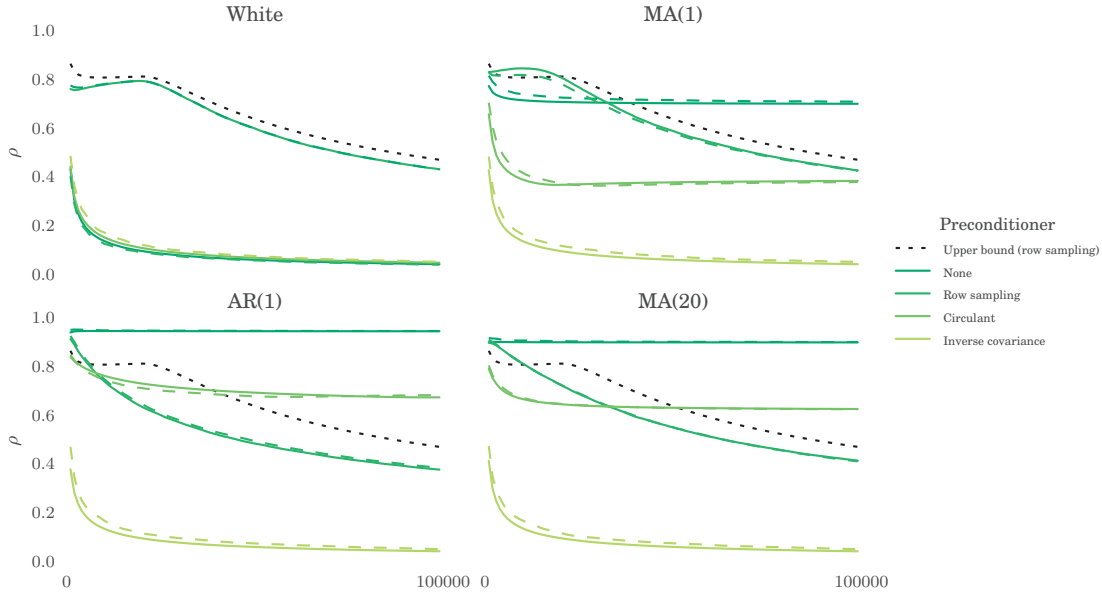


Figure 5.4: Numerical evaluation of $\rho = \left(\frac{\sqrt{\kappa(\bar{\mathbf{P}}\mathbf{R})-1}}{\sqrt{\kappa(\bar{\mathbf{P}}\mathbf{R})+1}} \right)$ for different signal models. The filter length is $p = 200$, the row update probability $q = 0.01$, and the regularization term $\delta = 10$. The lines from darker to lighter correspond to no preconditioner, row-sampling preconditioner, circulant preconditioner, and preconditioning with the true inverse covariance matrix, respectively. For the solid lines, the regression vectors were drawn independently at random from normal distributions with symmetric Toeplitz covariance matrices corresponding to White noise, moving average order 1, auto-regressive order 1, and moving average order 20. The processes coefficients are listed in Table 5.2. The long dashed lines were obtained by generating the above processes and creating the data matrix with shift structure of RLS. The short dashed line is the upper bound of Theorem 5.2.

rows m . Algorithm 5.3 however selects rows with probability q , and thus the number of rows is a sum of Bernoulli random variables $m = \sum b_i$. This number becomes however tightly concentrated around its mean when n is large. By a straightforward application of Hoeffding's inequality, we find that

$$\mathbb{P} \{ |m - \mathbb{E}m| \geq n\epsilon \} \leq 2e^{-2n\epsilon^2}.$$

5.6 Numerical Experiments

In this section we assess the practical performance of the proposed algorithm and compare it to that of the NLMS and RLS algorithms. In the first experiment, four driving signals x_n are used, unit variance white noise, an MA(1), an AR(1), and an MA(19) process. The moving average and autoregressive processes are generated by filtering white noise with rational filters whose coefficients are given in Table 5.2. The unknown filter \mathbf{w}^* is sampled uniformly at random

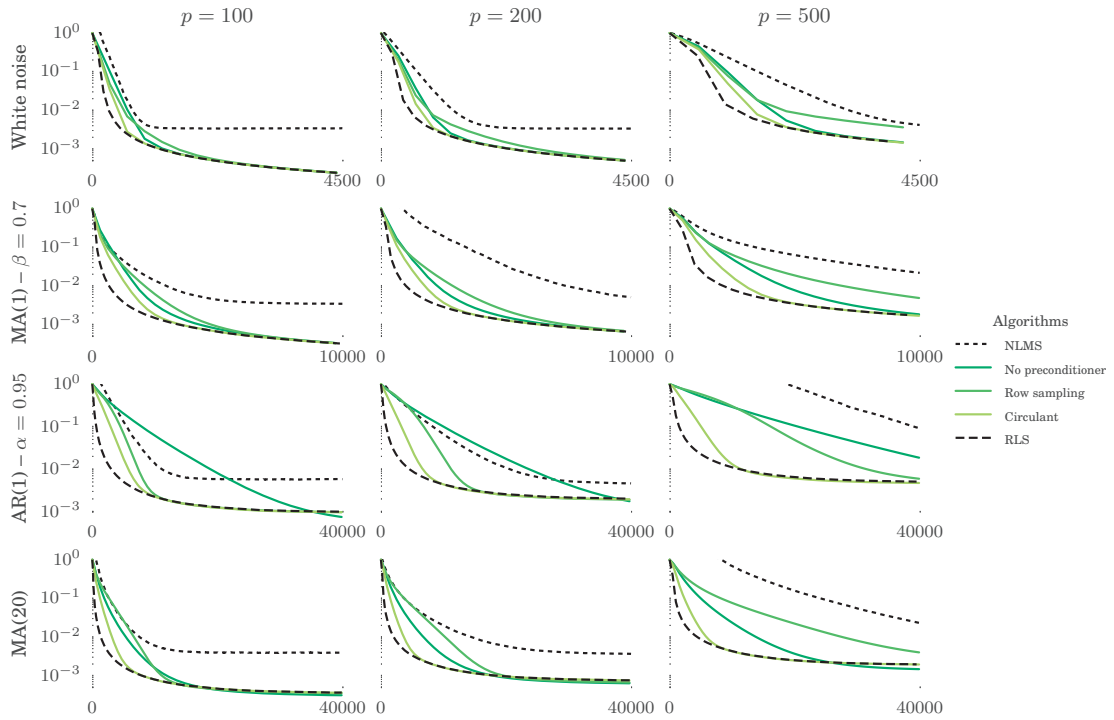


Figure 5.5: Evaluation of the MSE over time averaged over 1000 realizations of an adaptive filter of length $p = 100, 200, 500$, from left to right. The driving signals x_n are, from top to bottom, white noise, MA(1), AR(1), and MA(19) processes. The short and long dashed lines are NLMS and RLS, respectively. The solid lines are for Algorithm 5.2 with no preconditioner, row sampling, and circulant preconditioners (from darker to lighter tone) with $N = 5$ and $q = 0.005$. The SNR is fixed to 20 dB, the NLMS step size is $\mu = 0.5$, the regularization term is $\delta = 10$, and the forgetting factor is fixed to $\lambda = 0.9999$.

AR(1)	1, -0.95
MA(1)	1, 0.7
MA(20)	-1.042, -1.487, -2.631, -1.214, -1.492, -0.564, -1.230, -0.586, -1.018, -0.910, -0.499, -0.578, -0.669, 0.314, -0.062, -0.623, 0.075, 0.665, 0.256, -0.112

Table 5.2: Coefficients of the AR and MA processes used as driving signals in the numerical experiments. These are the coefficients of the denominator (AR) or numerator (MA) of a rational filter.

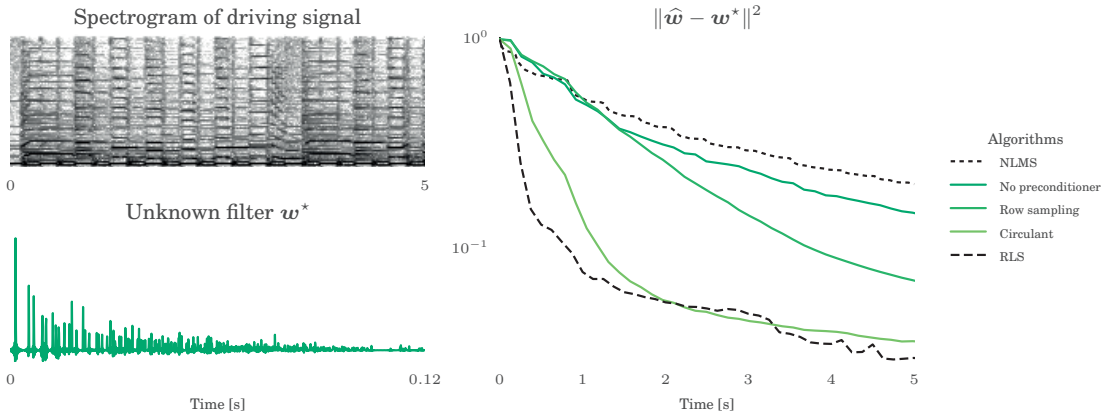


Figure 5.6: Evaluation of the convergence in a practical scenario. The driving signal is a piece of music with very non-stationary statistics sampled at 8 kHz. Its spectrogram is displayed on the top left corner with frequency on the y -axis going from 0 up to 4 kHz. The filter is the simulated impulse response of a 4×5 meters rectangular 2D room (bottom left). It is 1000 samples long. The convergence of Algorithm 5.2 with no preconditioner, row-sampling, and circulant preconditioners is compared to that of NLMS and RLS on the right. The parameters are set to $\lambda = 0.99995$, $\delta = 10$, $N = 10$ and $q = 0.005$. The NLMS step size is set to $\mu = 0.5$.

from the sphere \mathbb{S}^{p-1} with $p = 100, 200, 500$. The reference signal d_n is generated by convolving x_n with \mathbf{w}^* and adding white Gaussian noise such that the signal-to-noise ratio (SNR) is 20 dB. For RLS and RHS, the regularization parameter and forgetting factor are fixed to $\delta = 10$ and $\lambda = 0.9999$. The step size for NLMS is fixed to $\mu = 0.5$. This step size was picked to balance speed of convergence and residual error. Two experiments are done with the parameters of RHS. Finally, we run Algorithm 5.2 with no preconditioning, row sampling preconditioner, and circulant preconditioner. The sketching parameters for the two preconditioners are set to $q = 0.005$ and $N = 5$.

In Figure 5.5, the evolution of the MSE over time averaged over 1000 realizations of the adaptive algorithms is plotted for all configurations just described. As expected, RLS has the fastest convergence and lowest residual error. We observe that Algorithm 5.2 with circulant preconditioning offers the next best performance with the fastest convergence, close to that of RLS, in all cases. The row sampling preconditioning however does not seem to perform so well

unless the inverse covariance matrix is close to a diagonal matrix, as is the case for the AR(1) signal. Without a preconditioner, the algorithm seems to converge reasonably fast when the covariance matrix is close to diagonal, as is the case for White noise and the MA processes. In all cases investigated but $p = 100$ for AR(1), Algorithm 5.2 converges at rate on par, or faster than NLMS. Increasing the number of iterations N would naturally increase the convergence rate, but it is kept low on purpose to demonstrate that even in that case fast convergence is possible.

In the second experiment, we use a piece of music ⁴ sampled at 8 kHz as driving signal. The unknown filter \mathbf{w}^* is the synthetically generated impulse response of a rectangular mid-sized room long of $p = 1000$ samples and normalized to have unit norm. The SNR is set again to 20 dB. The regularization parameter and forgetting factor are $\delta = 10$ and $\lambda = 0.99995$. The NLMS and RLS parameters are as before. The sketching parameters are set to $q = 0.005$ and $N = 10$. The spectrogram of the driving signal and the impulse response of \mathbf{w}^* are displayed along with the convergence curves for the algorithms in Figure 5.6.

In this practical case RLS really demonstrate its superiority in terms of convergence compared to NLMS. While RLS converges to less than 10^{-1} relative error in less than a second, NLMS lingers above that even after five times as much time has passed. Again, Algorithm 5.2 with the circulant preconditioner is almost as fast as RLS itself, fully catching up after 2 seconds. Next, the row sampling preconditioner surprisingly performs better, relative to the other algorithms, than in the first experiment with the synthetic driving signals. While it doesn't catch up to RLS within the 5 seconds investigated, its convergence rate is significantly faster than NLMS. Finally, without any preconditioning, the convergence is only marginally faster than that of NLMS. Indeed, in such a practical scenario, we cannot expect the covariance matrix or its inverse to be close to a diagonal matrix. Hence the need for a good preconditioner to ensure fast convergence.

5.7 Conclusion

Inspired by recent advances in sketching for solving least squares problems, we proposed the recursive Hessian sketch (RHS), a new adaptive filtering algorithm solving the same exponentially weighted least squares problem as the conventional RLS but in an approximate way. The algorithm does so by running a few iterations of conjugate gradient. The sketched Hessian, after inversion, serves as a preconditioner that accelerates the convergence to the RLS solution. We propose two different preconditioners. The first one simply uses a subset of the rows of the data matrix chosen at random to build the sketch. The preconditioner is updated efficiently using the matrix inversion lemma for rank one updates. The second preconditioner adds Toeplitz symmetric constraints, suitable for audio signals, to the covariance matrix estimation and uses a circulant approximation to perform the inversion efficiently.

The convergence of CG using the row sampling preconditioner is analyzed and a bound for the convergence rate is given. Nevertheless, as demonstrated in this analysis and through numerical simulation, this simple row sampling scheme might lead to poor convergence in the beginning of the algorithm. While we do not give a theoretical analysis of the circulant preconditioner, numerical simulations shows it consistently performs well.

Two parameters, the number of CG iterations N and the update probability q , control the computational complexity and the convergence of the algorithm. We found that there are interesting operating points where the computational complexity is lower than that of RLS while

⁴The piece is a five seconds extract from *Vacation* by Katsuhiko Nagasawa, <https://soundcloud.com/katsuhiko-nagasawa/vacation>, CC-BY 3.0.

maintaining a fast convergence and low residual error. This was demonstrated through numerical experiments for a large number of combinations of parameters and driving signals. For natural signals, a piece of music, and an unknown filter of a thousand taps, the circulant preconditioner shows a performance close to that of RLS for very modest value of N and q .

There are several points of interest left to explore. First, it would be necessary to give bounds for the convergence of the algorithm with the circulant preconditioner. Furthermore, the bound developed in this work is a local bound at the current iteration. It does not translate to an explicit bound on the convergence rate to the unknown filter, only to the RLS solution. Having global convergence bounds for both preconditioner would be very valuable. Another aspect that is worth exploring is the behavior of the algorithm in a dynamic setting, i.e. where the unknown filter is slowly changing over time. It is well-known that RLS does not track very well such changing filter and it would be interesting to investigate if sketching might help in this situation. Finally, the algorithm does not consider the data when deciding to update the sketch. Using instead a criterion based on the novelty of the data observed could significantly improve the properties of the algorithm.

5.A Proof of Theorem Theorem 5.2

Unlike the original papers [103] [128], we adopt an analysis based on the stochasticity of the matrix \mathbf{X} . The random sketching will select $m = qn$ rows to include in the sketch. Since the order of the rows do not matter, we can group all m selected rows in matrix \mathbf{X}_s and move them to the top of \mathbf{X} . Let us call \mathbf{X}_{ns} the matrix formed by the non-selected rows, the bottom $n - m$ rows of \mathbf{X} . Thus we can write the auto-correlation matrix and its sketch as

$$\mathbf{R} = \frac{1}{n} (\mathbf{X}^\top \mathbf{X} + \delta \mathbf{I}) = q \underbrace{\frac{\mathbf{X}_s^\top \mathbf{X}_s + \delta \mathbf{I}}{m}}_{=\mathbf{R}_s} + (1 - q) \underbrace{\frac{\mathbf{X}_{ns}^\top \mathbf{X}_{ns} + \delta \mathbf{I}}{n - m}}_{=\mathbf{R}_{ns}}$$

where $q = \frac{m}{n}$. In Algorithm 5.3, \mathbf{R}_s is used as a preconditioner for CG. By classic preconditioned CG analysis [117], after N steps, the residual is bounded above as

$$\|\hat{\mathbf{w}}_{\text{RHS}} - \hat{\mathbf{w}}_{\text{RLS}}\|_{\mathbf{X}} \leq 2 \left(\frac{\sqrt{\kappa(\mathbf{R}_s^{-1} \mathbf{R})} - 1}{\sqrt{\kappa(\mathbf{R}_s^{-1} \mathbf{R})} + 1} \right)^N \|\hat{\mathbf{w}}_{\text{RLS}}\|_{\mathbf{X}},$$

where $\kappa(\cdot)$ is the condition number. We will now proceed to analyze the behavior of the condition number under assumption A2. First, let us study the product $\mathbf{R}_s^{-1} \mathbf{R}$ of the inverse sketch with the sample autocorrelation matrix

$$\mathbf{R}_s^{-1} \mathbf{R} = \mathbf{R}_s^{-1} (q \mathbf{R}_s + (1 - q) \mathbf{R}_{ns}) = q \mathbf{I} + (1 - q) \mathbf{R}_s^{-1} \mathbf{R}_{ns}. \quad (5.17)$$

Now, consider the eigenvalue decomposition of the covariance matrix $\Sigma = \mathbf{V} \mathbf{D} \mathbf{V}^\top$. It is possible to express the rows of the data matrix as $\mathbf{x}_i = \mathbf{V} \mathbf{D}^{1/2} \mathbf{y}_i$ where $\mathbf{y} \sim \mathcal{N}(0, \mathbf{I}_p)$. We can similarly stack the \mathbf{y}_i vectors in matrices \mathbf{Y}_s and \mathbf{Y}_{ns} . Let us define the two matrices

$$\hat{\mathbf{R}}_s = \frac{1}{m} (\mathbf{Y}_s^\top \mathbf{Y}_s + \delta \mathbf{I}) \quad \text{and} \quad \hat{\mathbf{R}}_{ns} = \frac{1}{n - m} (\mathbf{Y}_{ns}^\top \mathbf{Y}_{ns} + \delta \mathbf{I}),$$

and notice that $\mathbf{R}_s = \mathbf{V}\mathbf{D}^{1/2}\widehat{\mathbf{R}}_s\mathbf{D}^{1/2}\mathbf{V}^\top$, and we define \mathbf{R}_{ns} similarly. By expanding \mathbf{R}_s and \mathbf{R}_{ns} using the eigenvalue decomposition in (5.17) we obtain

$$\mathbf{R}_s^{-1}\mathbf{R} = q\mathbf{I} + (1-q)\mathbf{V}\mathbf{D}^{-1/2}\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns}\mathbf{D}^{1/2}\mathbf{V}^\top.$$

Similar matrices share the same eigenvalues, that is $\lambda(\mathbf{P}^{-1}\mathbf{A}\mathbf{P}) = \lambda(\mathbf{A})$ for some invertible \mathbf{P} . Thus, the condition number of the expression can be written

$$\kappa(\mathbf{R}_s^{-1}\mathbf{R}) = \frac{q + (1-q)\lambda_{\max}(\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns})}{q + (1-q)\lambda_{\min}(\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns})}. \quad (5.18)$$

At this point, a few remarks are in order. First, the condition number (5.18) does not depend on the covariance $\boldsymbol{\Sigma}$ of \mathbf{x}_i anymore. Second, the matrices $\widehat{\mathbf{R}}_s$ and $\widehat{\mathbf{R}}_{ns}$ are the (regularized) sample covariance matrices of two disjoint sets of independent unit variance normally distributed random vectors, $\mathbf{y}_i \sim \mathcal{N}(0, \mathbf{I}_p)$. The asymptotic distribution of eigenvalues of such matrices is described by the Marčenko-Pastur law [83]. For a friendly introduction to the topic, see Couillet and Debbah [32].

Theorem 5.3 (Marčenko-Pastur Law [83])

Let $\mathbf{X} \in \mathbb{R}^{n \times p}$ have independent normally distributed entries, i.e. $(\mathbf{X})_{i,j} \stackrel{iid}{\sim} \mathcal{N}(0, \mathbf{I}_p)$. Let the aspect ratio converge to a positive constant, that is $\lim_{n \rightarrow \infty} p/n = c$. Then, the eigenvalues λ of $\frac{1}{n}\mathbf{X}^\top\mathbf{X}$ have distribution

$$f(\lambda) = \max\{0, 1 - c^{-1}\}\delta(x) + \frac{1}{2\pi c\lambda} \sqrt{(\lambda - a)(b - \lambda)},$$

where δ is the Dirac delta function and a and b are defined as

$$a = (1 - \sqrt{c})^2, \quad \text{and} \quad b = (1 + \sqrt{c})^2. \quad (5.19)$$

While we could compute explicitly the distribution of the eigenvalues of $\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns}$ using this result, it is enough in our case to bound the eigenvalues. Since the eigenvalues are contained with high probability in the interval $[a, b]$, defined in (5.19), we have

$$\begin{aligned} \frac{1}{\frac{\delta}{m} + \left(1 + \sqrt{\frac{p}{m}}\right)^2} &\leq \lambda_{\min}(\widehat{\mathbf{R}}_s^{-1}) \leq \lambda_{\max}(\widehat{\mathbf{R}}_s^{-1}) \leq \frac{1}{\frac{\delta}{m} + \left(1 - \sqrt{\frac{p}{m}}\right)^2}, \\ \frac{\delta}{n-m} + \left(1 - \sqrt{\frac{p}{n-m}}\right)^2 &\leq \lambda_{\min}(\widehat{\mathbf{R}}_{ns}) \leq \lambda_{\max}(\widehat{\mathbf{R}}_{ns}) \leq \frac{\delta}{n-m} + \left(1 + \sqrt{\frac{p}{n-m}}\right)^2. \end{aligned}$$

By putting these inequalities together, and using the fact

$$\lambda_{\min}(\widehat{\mathbf{R}}_s^{-1})\lambda_{\min}(\widehat{\mathbf{R}}_{ns}) \leq \lambda_{\min}(\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns}) \leq \lambda_{\max}(\widehat{\mathbf{R}}_s^{-1}\widehat{\mathbf{R}}_{ns}) \leq \lambda_{\max}(\widehat{\mathbf{R}}_s^{-1})\lambda_{\max}(\widehat{\mathbf{R}}_{ns})$$

we obtain the following upper bound on the condition number

$$\kappa(\mathbf{R}_s^{-1}\mathbf{R}) \leq \frac{q + (1-q) \left(\frac{\delta}{qn} + \left(1 + \sqrt{\frac{p}{qn}}\right)^2 \right) \left(\frac{\delta}{(1-q)n} + \left(1 + \sqrt{\frac{p}{(1-q)n}}\right)^2 \right)}{q + (1-q) \left(\frac{\delta}{qn} + \left(1 - \sqrt{\frac{p}{qn}}\right)^2 \right) \left(\frac{\delta}{(1-q)n} + \left(1 - \sqrt{\frac{p}{(1-q)n}}\right)^2 \right)},$$

yielding the proof. \square

5.B Fast Matrix-Vector Products for Structured Matrices

In this appendix, we describe algorithms to quickly compute the product of a Toeplitz (or Hankel) matrix $\mathbf{T} \in \mathbb{R}^{r \times c}$ with a generic vector $\mathbf{v} \in \mathbb{R}^c$. The algorithms described make use of the fact that a Toeplitz matrix can be embedded in a circulant matrix, which is diagonalized by the discrete Fourier transform matrix. This algorithm is also described, albeit in less details, by Chan and Ng [26].

Let \mathbf{T} have left-most column $\mathbf{c} = [t_0, t_1, \dots, t_{r-1}]^\top$, and top-most row $\mathbf{r} = [t_0, t_{r+c-1}, t_{r+c-2}, \dots, t_r]$. It is possible to construct an $(r+c-1) \times (r+c-1)$ circulant matrix \mathbf{C} with left-most column

$$\mathbf{t} = [t_0, t_1, \dots, t_{r+c-1}]^\top$$

and such that \mathbf{T} is embedded in its top left corner,

$$\mathbf{C} = \begin{bmatrix} \mathbf{T} & \cdots \\ \vdots & \ddots \end{bmatrix}.$$

Now since circulant are diagonalized by the DFT matrix $\mathbf{F} \in \mathbb{C}^{N \times N}$, with $(\mathbf{F})_{k,n} = e^{-j2\pi \frac{kn}{N}}$, we can factorize \mathbf{C} as

$$\mathbf{C} = \mathbf{F}^H \mathbf{D} \mathbf{F},$$

where $\mathbf{D} = \text{diag}(\hat{\mathbf{t}})$, and $\hat{\mathbf{t}} = \mathbf{F} \mathbf{t}$ is the DFT of \mathbf{t} .

The procedure to compute the product of the Toeplitz matrix \mathbf{T} with a vector \mathbf{v} is straightforward and can be found from the factorization above and appropriate zero padding

$$\begin{bmatrix} \mathbf{T} \mathbf{v} \\ \vdots \end{bmatrix} = \mathbf{C} \tilde{\mathbf{v}} = \mathbf{F}^H \mathbf{D} \mathbf{F} \tilde{\mathbf{v}},$$

where $\tilde{\mathbf{v}}$ is the vector \mathbf{v} padded with zeros to the correct length. Now the algorithm simply consists in taking the matrix-vector product successively from right to left. This can be done efficiently using the FFT algorithm.

1. Compute $\hat{\mathbf{v}} = \mathbf{F} \tilde{\mathbf{v}}$ using FFT
2. Compute $\hat{\mathbf{t}} = \mathbf{F} \mathbf{t}$ using FFT
3. Compute the element-wise product of $\hat{\mathbf{v}}$ and $\hat{\mathbf{t}}$
4. Apply the iFFT to the resulting vector and discard the $r-1$ trailing elements

The complexity of this procedure is that of three FFTs of size $r+c-1$ and one element-wise vector multiplication of the same size, that is $O((r+c-1) \log(r+c-1))$. It is possible to refine the analysis for the case where one of r or c is much larger than the other. In that case, we can split the \mathbf{T} matrix into B approximately square blocks, for example when $c \gg r$,

$$\mathbf{T} \mathbf{v} = \begin{bmatrix} \mathbf{T}_1 & \cdots & \mathbf{T}_B \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_B \end{bmatrix} = \sum_{i=1}^B \mathbf{T}_i \mathbf{v}_i = \mathbf{F}^H \left(\sum_{i=1}^B \mathbf{D}_i \mathbf{F} \tilde{\mathbf{v}}_i \right),$$

can be efficiently computed in $O(c \log r)$. Note the slight abuse of notation where the last equality only holds after dropping the $r - 1$ trailing elements of the left term. When $r \gg c$, a similar algorithm exists

$$\mathbf{T}\mathbf{v} = \begin{bmatrix} \mathbf{T}_1 \\ \vdots \\ \mathbf{T}_B \end{bmatrix} \mathbf{v} = \begin{bmatrix} \mathbf{F}^H \mathbf{D}_1 \\ \vdots \\ \mathbf{F}^H \mathbf{D}_B \end{bmatrix} \mathbf{F}\tilde{\mathbf{v}},$$

and the complexity is $O(r \log c)$.

If instead we multiply by a Hankel matrix \mathbf{H} , we can still use the same algorithm. By reversing the orders of the rows of \mathbf{H} , we obtain a Toeplitz matrix to which we apply the algorithm just described. Then we just reverse the order of the elements of the vector obtained to get the result of the multiplication by \mathbf{H} .

Chapter 6

Tools and Methods for Reproducible Research in Computational Acoustics*

Alice thought she had never seen such a curious croquet-ground in her life; it was all ridges and furrows; the balls were live hedgehogs, the mallets live flamingoes, and the soldiers had to double themselves up and to stand on their hands and feet, to make the arches.

Alice's Adventures in Wonderland
LEWIS CARROLL

6.1 Introduction

Scholarly literature is the most prominent and visible product of the research activity. Nevertheless, the process leading from idea to scientific discovery generates a number of artifacts just as important. To understand this, let us follow a scientist through his research endeavor. First, he chooses a question he finds intriguing and deems worth investigating. He then embarks on a journey to find the answer by a rigorous application of the scientific method. He makes

*Pyroomacoustics is a collaborative work with Ivan Dokmanić, Sidney Barthe, and Eric Bezzam [10, 39, 112]. The printed circuit board design of the Pyramic array is the work of Francisco Rojo and René Beuchat while the FPGA core was developed by Juan Azcarreta Ortiz and Corentin Ferry [7, 43]. The browser based interface is the work of Basile Bruneau [22].

some hypothesis, and designs one or more experiments to refute or confirm it. Let us assume the experiment leads to fruitful results, yielding an interesting answer to the initial question. At that point, the answer obtained is just a collection of numbers and newly found understanding in our scientist's mind. To bring this new piece of knowledge into the human repository of knowledge, it needs to be turned into scholarly literature — the translation of scientific ideas into narratives. Thus, in addition to the final prose, some or all of the following artifacts were produced in the process:

- experimental protocols,
- datasets,
- instrumentation and hardware,
- simulation software,
- data processing software.

The scientific discovery process just described above is fraught with pitfalls and challenges and the accepted method to root error out of Science is for other researchers to reproduce and, possibly, extend the results obtained. Attempting this without knowledge of the experimental protocol, for example, would be an arduous and frustrating task.

There has been in the past few years growing concern that a large number of scientific results can simply not be reproduced just for this reason. As noted by Munafò et al. in *A manifesto for reproducible science*

Very little of the research process (for example, study protocols, analysis workflows, peer review) is accessible because, historically, there have been few opportunities to make it accessible even if one wanted to do so. [90]

Contrary to this historical reality, we believe that the means to achieve reproducibility might be within our reach.

The work presented in the four previous chapters is characteristic of the plurality of research outputs. The code base created to obtain the results in this thesis is just short of 30000 lines of code, as illustrated in Figure 6.1. It includes the simulation code for each chapter, a standalone package for the generation of room impulse responses (RIR), a number of reference implementations of algorithms. In addition, two microphone arrays were constructed and used for practical experiments. In this chapter, we first present the software package as well as the microphone arrays. Then we outline a few ideas to improve the situation concerning reproducible research.

6.2 Software: The Pyroomacoustics Package

While working on the acoustic rake receivers of Chapter 3, it was necessary to evaluate the performance of the beamformer designs produced. The gold standard for such evaluation is to design and carry out an experiment in a controlled environment with a real microphone array and careful calibration of the locations of all sound sources. The time and effort needed to setup these experiments naturally limit the number of replications of the experiments and the range of parameters that can be explored. In the exploratory phase of research, numerical simulation is an attractive alternative. It allows to quickly test and iterate a large number of ideas. In

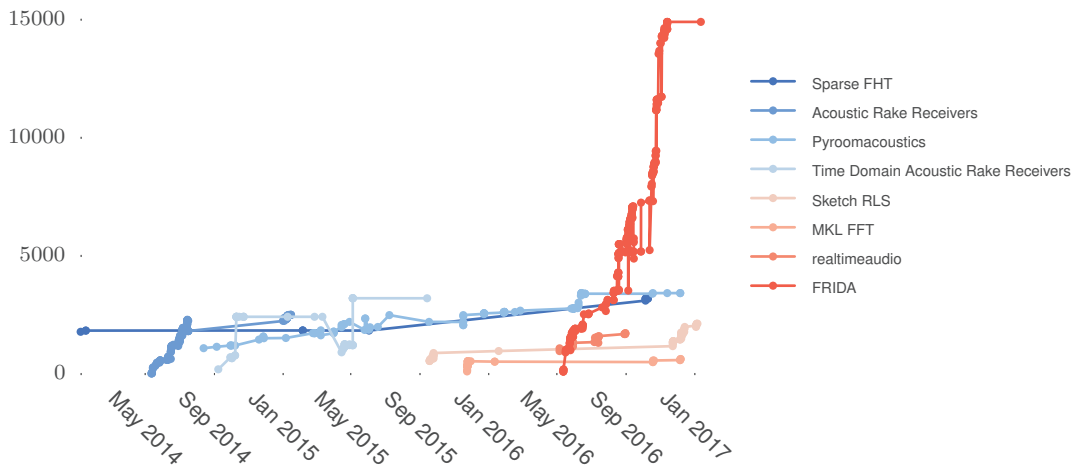


Figure 6.1: The evolution of the number of lines of code for each of the sub-projects making up this thesis.

addition it allows to finely tune parameters for the algorithm before going to experiments on real data.

The prerequisite for a simulation to yield useful information is that it models accurately enough real conditions to provide useful insights. In room acoustics, simulation based on the image source model has been used extensively for this purpose and has well-known strength and weaknesses [4]. This model replaces reflections on walls by virtual sources playing the same sound as the original source and builds RIR from the corresponding delays and attenuations. Its main advantage is its simplicity. The model is accurate only as long as the wavelength of the sound is small relative to the size of the reflectors, that it assumes to be uniformly absorbing across frequencies. The model applies to polyhedral rooms in two and three dimensions, convex and non-convex [17].

Our wishlist for an RIR generator is: affordable, open source, and flexible. A number of generators are available, and most if not all are shared online free of charge. For example the popular generator from Emanuel Habets [55]. Unfortunately, none allow room shapes other than rectangular. Furthermore, most rely on MATLAB, which is widely used in the signal processing community, but whose price means that it might not be available to some students or institutions with limited financial resources. In addition, its restrictive licensing makes it difficult to use in some situations. For example to run large scale simulation on a distributed computing cluster.

Faced by the limitations of available RIR generators, we decided to develop our own generator. We chose to develop the project in the Python language. Python is open source and available for free. It is a modern object oriented language that values readability first. It is suitable for scientific computations and includes interfaces to standard high performance linear algebra libraries such as the linear algebra package (LAPACK), the basic linear algebra subroutine (BLAS), or the Intel math kernel library (MKL) [96]. The popularity of Python being not limited to the science and engineering community means that a variety of package and extensions are available for tasks from database access to web server, making it easy to integrate scientific code

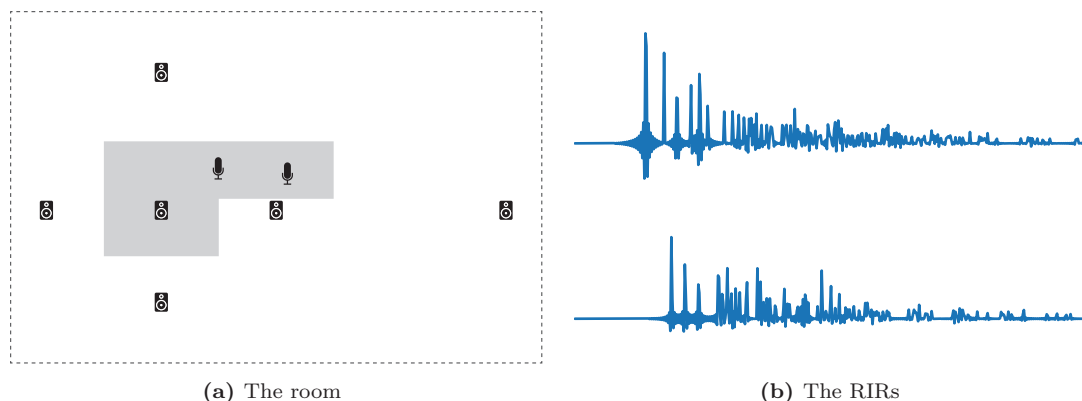


Figure 6.2: (a) An example of a non-convex room containing one source and two microphones with the first order images drawn. (b) The two RIR between the source and the microphones produced by pyroomacoustics. Both figures were produced by the code from Listing 6.1.

in applications. For example creating an online demonstration website for a research project.

6.2.1 Structure

One of the goals of Pyroomacoustics is to exploit the object oriented features of python to create a clean and intuitive interface for room acoustics simulation. This in turn allows to easily write very clear simulation scripts with human readable syntax. A room is defined as an object that has a collection of walls, of sources, and of microphones. As demonstrated in Listing 6.1, generating RIRs is as easy as creating a room by giving a list of corners, adding sources and microphones, and calling the appropriate routines for RIRs generation. The output produced by this script is shown in Figure 6.2.

A Beamformer object, inheriting from MicrophoneArray, can be used instead. In that case, beamforming weights can be computed according to several methods. A sound segment can be assigned to each sources and virtually played, recorded by microphone array, and processed by the beamformer. Listing 6.2 shows an example of a delay-and-sum (DS) beamformer in a rectangular room. The beamforming weights are computed to focus towards the source and the beampatterns produced can be plotted to produce Figure 6.3.

6.2.2 Room Impulse Response Generator

The RIR generator is based on the extension of the image source model to arbitrary polyhedra by Borish [17]. The definition of the room as a collection of walls extends easily from 2D, where the walls are line segments, to 3D where they are now flat polygons defined by a collection of points. For the image source algorithm, walls need to have routines checking on which side of the wall a point is, and to compute intersections of lines with the wall. Specialized routines are provided to create rectangular rooms, or rooms from polygons. Two dimensional polygons can be extruded to create easily 3D rooms with complex floorplans.

When a source is added to a room, all its valid image sources are computed and stored.

```

1  import numpy as np
2  import matplotlib.pyplot as plt
3  import pyroomacoustics as pra
4
5  # Create a 2D room from the corners of a polygon
6  pol = np.array([[0,0], [0,1], [2,1], [2,0.5], [1,0.5], [1,0]]).T
7  room = pra.Room.fromCorners(pol, max_order=9, absorption=0.1)
8
9  # Add a sound source
10 room.addSource([0.5, 0.4])
11
12 # Place two microphones in the room
13 R = np.array([[1., 1.6], [0.75, 0.7]])
14 room.addMicrophoneArray(pra.MicrophoneArray(R, room.fs))
15
16 # Run the image source model
17 room.image_source_model()
18
19 # Show the room and 1st order images
20 room.plot(img_order=1)
21
22 # Display the room impulse responses
23 plt.figure()
24 room.plotRIR()
25 plt.show()
26

```

Listing 6.1: Example of RIR generation

```

1  import numpy as np
2  import matplotlib.pyplot as plt
3  import pyroomacoustics as pra
4
5  # Create a 4 by 6 metres shoe box room
6  room = pra.Room.shoeBox2D([0,0], [4,6])
7
8  # Add a source somewhere in the room
9  room.addSource([2.5, 4.5])
10
11 # Create a linear array beamformer with 4 microphones
12 # with angle 0 degrees and inter mic distance 10 cm
13 R = pra.linear2DArray([2, 1.5], 4, 0, 0.04)
14 room.addMicrophoneArray(pra.Beamformer(R, room.fs))
15
16 # Now compute the delay and sum weights for the beamformer
17 room.micArray.rakeDelayAndSumWeights(room.sources[0][:1])
18
19 # plot the room and resulting beamformer
20 room.plot(freq=[1000, 2000, 4000, 8000], img_order=0)
21 plt.show()
22

```

Listing 6.2: Example of delay-and-sum beamforming

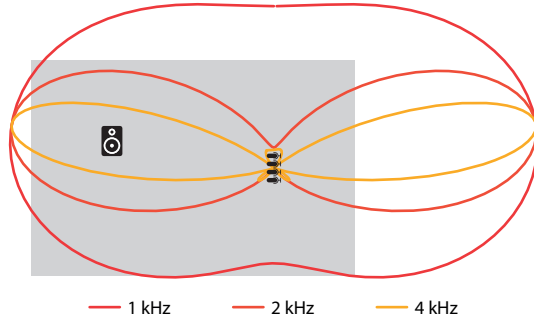


Figure 6.3: Example of the beampatterns of a delay-and-sum beamformer at 1, 2, and 4 kHz as produced by Listing 6.2.

When computing the RIR, the visibility of the microphone from each image source is checked. In non-rectangular rooms the reflections of walls that form an obtuse angle might not be visible from the whole room. In non-convex rooms, the view from the source to the microphone might be obstructed by a wall. For a microphone placed at \mathbf{r} , a real source \mathbf{s}_0 , and a set of its visible images sources $\mathcal{V}_r(\mathbf{s}_0)$, the impulse response between \mathbf{r} and \mathbf{s}_0 , sampled at F_s , is given by

$$a_{\mathbf{r}}(\mathbf{s}_0, n) = \sum_{\mathbf{s} \in \mathcal{V}_r(\mathbf{s}_0)} \frac{\alpha^{\text{gen}(\mathbf{s})}}{4\pi \|\mathbf{r} - \mathbf{s}\|} \delta_{\text{LP}} \left(n - F_s \frac{\|\mathbf{r} - \mathbf{s}\|}{c} \right),$$

where $\text{gen}(\mathbf{s})$ gives the reflection order of source \mathbf{s} , $\alpha \in [0, 1]$ is the reflection factor of the walls, c is the speed of sound, and δ_{LP} is the windowed sinc function

$$\delta_{\text{LP}}(t) = \begin{cases} \frac{1}{2} \left(1 + \cos \left(\frac{2\pi t}{T_w} \right) \right) \text{sinc}(t) & \text{if } -\frac{T_w}{2} \leq t \leq \frac{T_w}{2}, \\ 0 & \text{otherwise.} \end{cases}$$

The parameter T_w controls the width of the window and thus the degree of approximation to a full sinc. Two RIRs produced this way can be seen in Figure 6.2b.

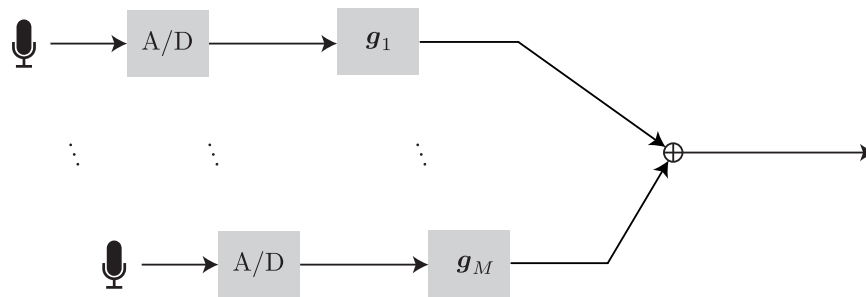
6.2.3 STFT Engine

Once the propagation of signals from sources to microphones has been simulated and the beamforming filters have been computed, the former must be convolved with the latter to produce the output of the beamformer. Two approaches are possible. If the filters are relatively short, it is possible to directly compute the convolutions in the time domain, as in Figure 6.4a. For longer filters, it is computationally interesting to do the computations in the short time Fourier transform (STFT) domain. See Figure 6.4b.

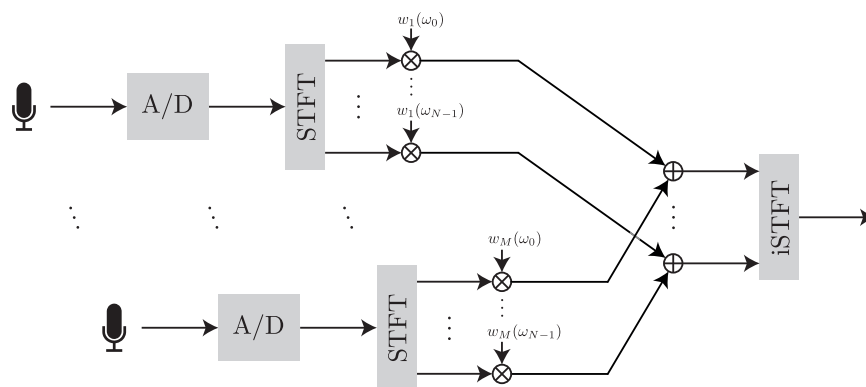
The STFT engine allows any frame size and overlap, zero padding, and different window functions to be used. The inverse STFT uses standard overlap-add with the possibility of using a synthesis window for non-linear processing.

6.2.4 Reference Implementations

When evaluating the performance of new algorithms, a large amount of time is spent re-implementing competing methods to run comparisons and benchmarks. The availability of quality reference



(a) Time domain Processing



(b) STFT Processing

Figure 6.4: Block diagrams of (a) time domain and (b) STFT domain beamforming algorithms.

implementations for popular algorithm has the potential to speed up considerably the time-to-market of new research projects. We provide implementations of several algorithms for beamforming, direction of arrival (DOA) finding, adaptive filtering, and source separation.

Beamforming and Source Separation The classic beamforming algorithms are included as special cases of the acoustic rake receivers of Chapter 3. Namely, by including only the direct source, we recover the DS [121] and MVDR [24] beamformers. Both far and near field formulations can be used. In addition, the blind source separation algorithm TRINICON [23] is included.

DOA Finding Developed for the comparison to FRIDA (Chapter 4), we provide implementations for the popular multiple signal classification (MUSIC) [116] and steered response power phase transform (SRP-PHAT) [35], as well as coherent signal subspace method (CSSM) [127], weighted average of signal subspaces (WAVES) [34], and test of orthogonality of projected subspaces (TOPS) [136].

Adaptive Filtering Implementations of the least mean squares (LMS), normalized LMS (NLMS), and recursive least squares (RLS) were added while working on the recursive Hessian sketch of Chapter 5 [60].

6.2.5 Future Work

We are planning to continue to extend this package in the hope that it can benefit the audio signal processing community. The current version of Pyroomacoustics only supports omnidirectional sources and microphones. The ability to add directivity patterns to loudspeakers and microphones is critical to bridge the gap between simulation and experiments. Ideally, both parametric patterns (e.g. cardioid microphones) and measured ones should be supported.

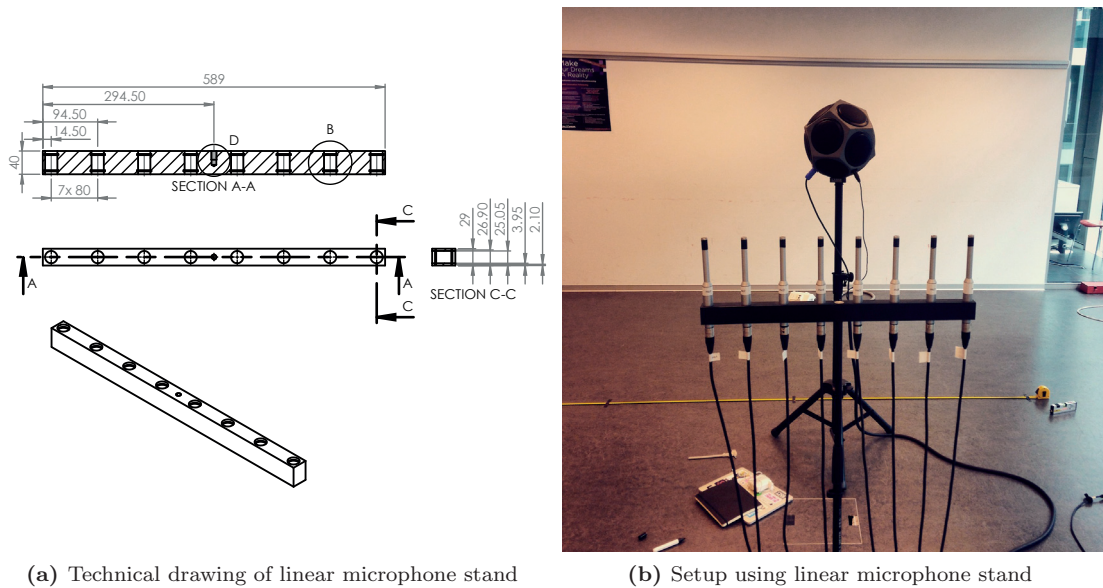
Another weakness of the current simulator is the pure python implementation of the RIR generator lack of speed. Thorough optimization of the critical path of the computation, if possible, or re-implementation as an external C module otherwise, is needed.

Currently the input of room shapes is awkward for more complicated rooms, especially non-convex 3D rooms. One way of simplifying this is to implement set operations of polygons and polyhedra, e.g. union, difference, etc, making it possible to build complex shapes from a set of basic ones such as rectangles and triangles. Another way is to write a parser for files produced by conventional CAD software (e.g. SketchUp, AutoCAD).

Finally, the documentation of the code is a work in progress, and while a large part of the code is systematically documented, there remains a large number of routines without proper comments and explanations. Adding pedagogical examples would also help make the package more user friendly.

6.3 Hardware: Flexible Microphone Array Architectures

While simulations and numerical experiments can be fairly convincing, experiments are the gold standard for the evaluation of new methods. Unfortunately, practical experiments are painstaking to setup, time consuming, and in addition can be difficult to replicate. It is thus essential to use all tools available to reduce the variability in the experiment setup process.



(a) Technical drawing of linear microphone stand

(b) Setup using linear microphone stand

Figure 6.5: A linear microphone array using a custom made rigid stand to simplify the setup of conventional measurement microphones.

A drawback of experimental setups with conventional audio recording equipment are the boom stands used to hold microphones and speakers. Complex geometries are difficult to calibrate in the first place and close to impossible to reproduce for another experiment once the setup has been taken down. In our first attempt to tackle this issue, we custom built a special mount to hold eight ECM8000 Behringer microphones in a linear array configuration. The mount was machined in polyoxymethylene according to the technical drawing shown in Figure 6.5a and uses O-ring to mechanically hold the microphones in place. An example of a setup using the mount is pictured in Figure 6.5b. While easing the process of getting all microphones in the same arrangement over multiple experiments, this mount doesn't alleviate the need for a bulky audio interface and a large number of long wires.

Traditional experimental setups rely on professional grade recording equipment with flat-response microphones and specially calibrated sound sources. While being required for acoustical metrology, audio signal processing algorithms are generally resilient to imperfections in the measurements, and target consumer grade hardware for the final application. An alternative to standard measurement microphones are the newer microelectromechanical systems (MEMS) microphones. Their smaller form factor allows them to be carried on a much lighter structure, for example one made of laser cut acrylic. This open up the door to fixed array geometries that can be consistently reproduced in repeated experiments. New geometries can be easily created simply by laser cutting them, an operation taking but a few minutes. Another advantage is that MEMS microphones often come with an analog to digital converter (ADC) built-in, reducing the complexity of the electrical circuit to design.

Thus motivated, we decided to build our own microphone array platform based on MEMS technology. We present in Sections 6.3.1 and 6.3.2 two different designs. In both cases, the

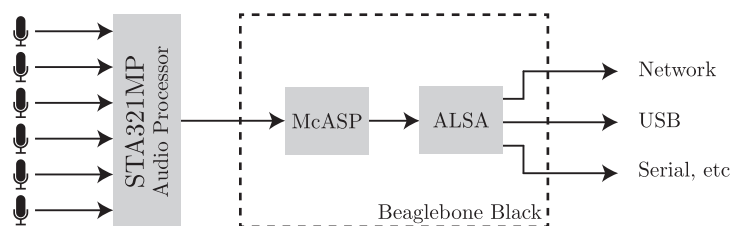


Figure 6.6: Architecture of the CompactSix array.

systems use MEMS microphones with an appropriate interface and an embedded Linux operating system on the host processor. Having a full operating system on the recording device opens up possibilities that were not available before. For example, the device can be accessed through a wired or wireless network connection, or it can be further connected to other devices as part of a demonstration. This greater flexibility, however, comes at the cost of usability as going through the Linux operating system to recover the samples is slightly less convenient. As a solution, a modular browser based interface for the arrays was created and is described in Section 6.3.3.

We should mention that since we started this project, the popularity of speech enabled devices such as Google Voice or Amazon Echo have inspired the development of open source alternatives like the Matrix Creator¹ or the ReSpeaker² platforms.

6.3.1 CompactSix Array

The CompactSix array is a microphone array with six MEMS microphones connected to a Beaglebone Black, an open source single board computer produced by Texas Instrument³. The Beaglebone Black sports an ARM Cortex-8 central unit processor (CPU) and is a good candidate for audio processing thanks to its two programmable real-time units (PRU). It has been identified as suitable for real-time processing of audio and sensor signals [86, 124]. The microphones chosen are manufactured by Knowles and contain an ADC and a sigma-delta modulator producing a pulse density modulation (PDM) signal. PDM signals are digital signals oversampled at 64 times the sampling frequency and quantized to one bit. Noise shaping ensures that the quantization noise is in the high frequency only, so that the spectrum of the sound can be recovered by digital low pass filtering. The STA321MP digital audio processor from ST Microelectronics is used to convert the PDM signal to the more standard pulse coded modulation (PCM) format, then transmitted through an inter-IC sound (I2S) bus to the CPU.

On the software side, the advanced Linux sound architecture (ALSA) is the software framework that handles sound in the Linux kernel. Since no ALSA driver was available for the STA321MP audio processor, we wrote it and are thus able to use any ALSA compatible software to process the samples acquired. The ALSA driver written leverage the multichannel audio serial port (McASP) of the ARM CPU on the Beaglebone Black to decode I2S signals. The architecture of the CompactSix array is shown in Figure 6.6.

Physically, each microphone is soldered onto a small individual printed circuit board (PCB) connected by wires to an extension board hosting the audio processor, itself plugged into the

¹<http://creator.matrix.one>

²<http://www.seeedstudio.com>

³<http://beagleboard.org/>

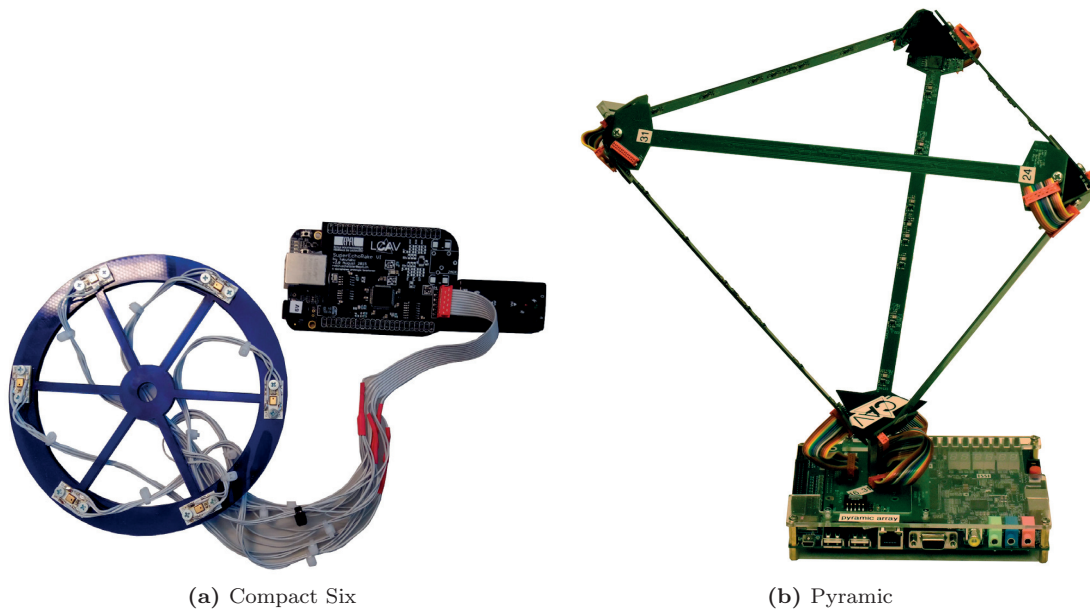


Figure 6.7: The (a) CompactSix and (b) Pyramic arrays.

Beaglebone Black. The microphones PCB can then be screwed onto a laser cut mount to create a microphone array. A picture of a circular array together with the processing platform can be seen in Figure 6.7a.

6.3.2 Pyramic Array

Most conventional audio interfaces in the affordable price range are limited to recording with eight channels. Since we are designing our own hardware, we are not restricted and can explore architectures with a more extreme number of microphones. We designed the Pyramic array with 48 microphones arranged in a semi-reconfigurable configuration. With such a large number of microphones it can be cumbersome to place them individually as we did for CompactSix. Instead, we arranged them in six groups of eight microphones. Each group is placed in a line on a single PCB with an AD7606 ADC from Analog Devices. The PCBs can then be assembled into different shapes with minimal effort. The configuration that we have chosen for experiments is that of a tetrahedron with each edge being a PCB. To reduce the number of wires needed, the PCBs were designed so that two of them can be daisy chained. A picture of the array is shown in Figure 6.7b.

A lot of processing power is needed for all 48 channels and we elected to use a field programmable gate array (FPGA) for the task. We decided to use the DE1-SoC platform from Terasic⁴ as its Altera system-on-chip includes both a Cyclone-V FPGA and an ARM Cortex-A9 CPU, as well as many peripherals, including an audio CODEC that can be used to output audio signals. The ADCs are wired directly to the FPGA where a serial peripheral interface reads the

⁴<http://www.terasic.com.tw/>

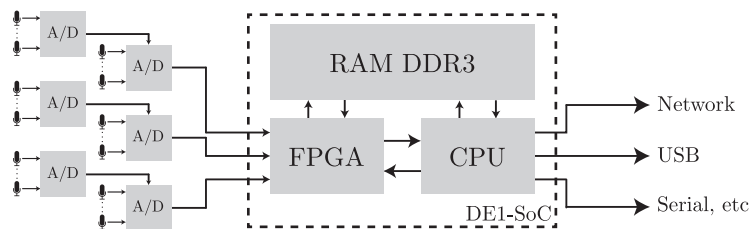


Figure 6.8: Architecture of the Pyramic array acquisition system.

samples of all eight ADCs synchronously, and stores them in the shared DDR3 random access memory (RAM). The ARM CPU running Linux controls when the acquisition system starts and stops, and can access the recorded samples in the RAM. The acquisition system is illustrated in Figure 6.8. Another FPGA module allows to output sound from the CODEC on the DE1-SoC board by writing them to RAM from the CPU. In the future, we are planning to add the possibility to do real-time beamforming with input from the 48 microphones and stereo output.

6.3.3 Easy-DSP: Browser Based Interface for Embedded Arrays

As we created the CompactSix and Pyramic platforms, we realized that the workflow used for data acquisition and processing was both cumbersome and did not scale well. We would first connect to the board via secure shell (SSH), trigger the acquisition command, and finally, after completion of the recording, retrieve the samples using SSH again or some other protocol. This had several disadvantages: the process was very manual, did not let itself be scripted easily, and did not allow for real-time processing of the data. An unfortunate consequence of this is that the platforms did not provide for great demonstrations.

To address these issues, we developed a modular framework providing a way to interact with microphone arrays running on some embedded operating system, Linux for now. This framework relies on three technologies nearly universally available — TCP/IP, Javascript, and Python — drastically reducing the installation requirements to use it. The framework allows to listen and record audio from the microphone array, process it in Python, and display a number of charts updated in real-time. It makes it very easy to quickly prototype and test algorithms in real-time for research, during laboratories with students, or for demonstrations.

On the side of the microphone array, a light server reads samples from the audio interface (ALSA for CompactSix, and directly from the shared memory for Pyramic), and streams them through a WebSocket [44] to connected clients. The WebSocket standard makes it possible to connect to this server from Javascript from within a web browser. This is thus how the interface to the system is implemented. On a basic level, the browser interface allows to configure the hardware — sampling frequency, number of channels, buffer size, and volume — and listen to the audio. By pressing a button, it is possible to record an audio segment that is subsequently downloaded by the browser just as any file would be. The interface provides in addition information on the status of the microphone array.

In addition to the browser based interface, a Python package with the same ability to communicate through WebSockets with the microphone array was developed. This makes it possible to fully script the acquisition process and combine it for example with playing sound samples through a loud speaker during some automatized recording sessions. Beyond simple scripting,

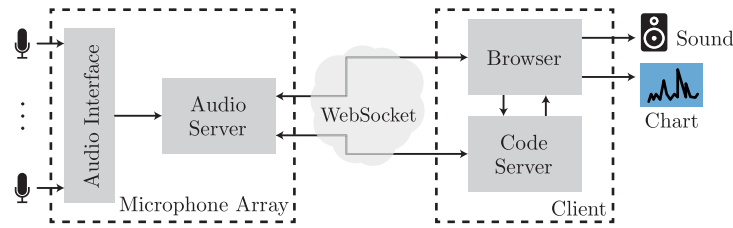


Figure 6.9: System diagram of the modular browser based interface.

the Python code can be written and executed from within the browser through a code server. An editor is included in the interface and, upon pressing the launch button, the code is sent to the code server, connects to the audio server, process the samples and sends them back to the browser. In addition to sound, the code server can also send back data to plot. A diagram of the whole system is shown in Figure 6.9.

6.4 Thoughts on Open Science and Reproducible Research

The challenge we are faced with at this point is how to link the scholarly work presented in the previous four chapters to the artifacts of this one in a way that promotes sharing and reuse while ensuring proper attribution. We can distinguish two kinds of artifacts, text based, e.g. manuscripts, protocols, source code, and non text based, e.g. PCB design files, other types of computer aided design (CAD) files, compiled proprietary code, etc. Interestingly data can be of either kind. The main challenges we need to solve are as follows.

Attribution Research is built on the work done by our peers and proper attribution is required when using prior work. Attribution and citations allow to measure impact, which is the bread and butter of scientists.

Version control The tools and methods evolve together with the research itself to adapt to the needs and the findings along the way. We distinguish two features that are necessary. Versioning, that is being able to point to a specific state of the object in time, and "Diff", the ability to precisely understand what are the, possibly minute, differences between two versions of the object.

Sharing and dissemination Digital storage space is needed for all the research material being produced. It is important that this storage be persistent and continues to be available long after the research is over. In addition, it is important that a way to query systematically and retrieve research artifacts exists.

For the case of source code, the free and open source software (FOSS) community is faced with remarkably similar challenges. It typically involves a large number of collaborators from around the world. They work on the same piece of code so that version tracking is of utmost importance for proper synchronization and issue tracking. Finally, the fame that developers derive from their contributions is often their only form of payment so that proper attribution is crucial. There is thus much to learn from their solutions to these problems from the research community. Version control systems (VCS) are software tools that allow to track atomic contributions to the

source code. Currently the most popular VCS is git⁵ which was initially developed by Linus Torvalds for the development of the Linux kernel. Git marks every individual contribution with a unique SHA-1 hash, keeping a tree that allows to reconstruct any state the source code has ever been. In addition, git works on a distributed model, enabling the concept of *fork* where anyone can branch off the main development tree to create a new project. Apart from version control, software engineering developed a whole set of good practices directly applicable to the development of scientific software such as unit, integration, or regression tests, [132]. Stodden and Miguez give precise guidelines to enable the reproducibility of computational research, from licensing, to documentation, to workflow [120].

For text based artifacts, tools for software such VCS can be leveraged almost without any modification. This makes a very good case for the use of open text formats for manuscripts, protocols, or lab notebooks. The mathematics, and to some extent, engineering communities widespread use of Latex for manuscript typesetting satisfies this requirement. This could make it possible to track changes between different available versions of the same manuscript. The open repository arXiv⁶ already offers some rudimentary ways of updating a manuscript without losing track of the original content. VCS technology offers unprecedented possibilities. Imagine a biologist would like to alter the genome of a bacteria by introducing a new gene and study its effect. To do so, he finds a paper operating on the same bacteria, but with a related, but different gene. The current situation is that our biologist will have to extrapolate from the description available in the paper to a practical protocol, since many routine operations have most likely been omitted to save space. He will then modify the protocol to suit his needs and perform his experiment. If successful, he will publish the results in a new paper citing the reference for the protocol used. Several important information are lost in this process. First, the missing routine details omitted in the original, as well as the new publication, might hide a critical operation that escaped the attention of the authors. Second, if only part of the original protocol was used, it is difficult to understand which. Third, it is not possible to look at both protocols side by side to understand precisely where they are alike and where they are not. By storing protocols, for this example, in a VCS it would become possible to include all the routine and minor steps generally omitted. It would be possible to track the changes to the protocol as it evolves through trial and error towards a successful experiment. It would be possible to reference a precise version of a protocol by its hash value. It would finally be possible to understand precisely minute differences between different versions of the protocol used for different experiments, by different researchers.

For custom hardware and instrumentation developed during research, things are more complicated. First, the design files are generally not text based, and even when they are, they are not human readable. Despite this, VCS designed for natural text information still seems to be the most popular way to share them. A big downside of these formats is that it is very difficult to compare two versions, the best way being currently to open both versions and visually compare them. What would be really needed is a tool that allows to semantically inspect the differences. Imagine an engineer has PCB files for two versions of the same circuit. A useful tool would tell him that the value of resistance R2 changed from 10 kOhms to 27 kOhms. Second, as outlined in the guidelines of the open source hardware association⁷ (OSHWA) copyright law does not apply to hardware as it does to software. Regardless, under some interpretation, the design files might be covered and can thus be licensed. Their recommendation is to license hardware design files under open licenses (such as creative commons) so as to clarify under which condi-

⁵<http://www.git-scm.com>

⁶<http://arxiv.org>

⁷<http://www.oshwa/faq>

tions they can be reused. Third, matters are further complicated by the use of proprietary CAD software covered by far reaching terms of use [88]. Albeit these softwares are in general very expensive, which in itself is a challenge to their use for open science, specially priced licenses for academic use are made available. These academic licenses often come with the extra condition that they cannot be used for commercial purposes. This specific condition has the disastrous consequence of effectively prohibiting the release of the design files under an open license as this would allow anyone to use these files for any use, including commercial, thus violating the terms of the license. As a consequence, open source alternatives to these softwares should be preferred whenever available.

At this point, we have described only the attribution and version control parts of the problem. Let us now concentrate on the problem of sharing and disseminating scholarly work. There are in fact many platforms and repositories, commercial or not, available for the purpose of storing, publishing, or archiving scholarly works. The popular code sharing platform GitHub⁸ has been embraced by the scientific community to share data and code alike [101]. Based on the VCS git, it offers all its advantages described above, in addition to very useful social network features. On the downside, it doesn't allow to use digital object identifiers (DOI), which is the *de facto* way of cross-referencing scholarly objects online. The platform Zenodo⁹ bridges this gap by letting researchers take snapshots of repositories and assigning them DOI. This later lets researchers cite these snapshots. Figshare¹⁰ is another platform offering this service not just for code, but any object (slides, poster, figure, etc). A number of websites like runmycode.org or researchcompedia.org hosts companion pages for code and data supporting published work [119]. Finally, the Dataverse Project¹¹ is the closest to fulfilling the wishlist formulated but is principally geared towards datasets [71]. It is a web application developed at Harvard's Institute for Quantitative Social Science that allows to host datasets and give them unique identifiers that can be linked to specific published work. It lets users upload newer versions of their datasets. The software is open source and can be installed and managed by universities to fulfill their datasets management needs. A community of 19 dataverses currently exists around the world.

This was just a few of the most prominent options available, not including that many universities are creating their own repositories, along with traditional publishers. This profusion of services supporting reproducible research is both encouraging and confusing at the same time. The fact that these projects essentially all act in isolation, and often on grants or research funds leads to worry that they might disappear when the funding runs out. Moreover, the disparity of the platforms and their lack of coordination makes the cross-referencing and search of information close to impossible. The situation is akin to the early days of the world wide web when no effective search engine was available. GitHub, being so large and popular, is the exception here as it has built-in many desirable features like automatic attribution when forking a repository, possibility to raise issues, and pull requests to merge back contributions into projects. Nonetheless, relying on a commercial service that might disappear suddenly when it stops being profitable is not a sustainable solution. Even though many universities have pledged to support reproducibility and open science, the current situation makes enforcement of these commitments difficult.

We would like now to describe our vision of the road ahead toward fulfillment of the ideal of reproducible research. First, most if not all of the frameworks and platforms above are nothing

⁸<http://github.com>

⁹<http://zenodo.org>

¹⁰<http://figshare.com>

¹¹<http://dataverse.org>

more than technical solution to the problem, whereas policy is needed. Just like having nuts and bolts of standard sizes enabled the industrial revolution, a standardized reproducible research framework is necessary for 21st century research.

Our proposition for this is two-fold. First, rather than leaving their data storage needs in the hands of the market, universities should provide all the means for their researchers to host and publish their manuscripts, code, and data through platforms like the infoscience¹² repository of EPFL. Incorporating the features listed above for attribution and version control would allow researchers to measure their impact and facilitate collaboration within the institution. A consequence is that all researchers should be trained in the use of versioning tools. Hopefully, tools more user friendly than git can be developed. On the administration side, it would allow to evaluate compliance of the institution with its own openness commitments. For example a robot could check that publications in the repository include a link to code and data if mentioned in the manuscript, and, if no link exists, flag the paper, and warn the authors.

Second, and learning from the history of the Internet, efforts to form a standardization committee in collaboration with other institutions worldwide should be started. Its goal would be to come up with a standard communication protocol that would let repositories communicate and a common application programming interface (API) that would let them be queried. The communication protocol could be inspired by decentralized peer-to-peer protocols like bittorrent and gnutella. Having such a network would make it possible to query from any of the institutions connected. Having a distributed network rather than a single repository, and considering that universities are some of the longest living institutions in the world, would make the network robust to any single bankruptcy or mismanagement. The different parties could negotiate agreement for the replication of their data across sites to further protect from loss of information. Nevertheless, a particularly important part of any standard is to be sufficiently technology-agnostic leaving implementation details in the hands of the system designers. Such a model would also allow traditional journals and actors from the private sector to build their own services on top of this infrastructure.

On a technical note, an interesting development is taking place thanks to the advance of peer-to-peer technology and cryptocurrencies. Worried about the central role of GitHub, for the reasons mentioned above, a few open source developers have started to create tools for a truly distributed GitHub, in one case, the gittorrent project, by storing repositories in bittorrent [8], and another one, Mango, using Ethereum [15].

To summarize, what we propose is to create a standard architecture and protocol for a distributed repository for manuscripts, code, and data. It would rely on peer-to-peer technology for the storage and query, and use concepts of git for the version control and the cross-referencing of atomic contributions.

¹²<http://infoscience.epfl.ch>

Conclusion

“Begin at the beginning,” the King said,
gravely, “and go on till you come to an end;
then stop.”

Alice’s Adventures in Wonderland
LEWIS CARROLL

We are now reaching the end of this four years journey through the signal processing pipeline. We have proposed a fast algorithm for the Walsh-Hadamard transform (WHT) of signals that can be sparsely represented in this basis. We have made constructive use of echoes by using raking beamformers. We proposed FRIDA, a finite rate of innovation (FRI) direction of arrival finding algorithm with many qualities. The combination of adaptive filters with modern sketching techniques let us decrease complexity without sacrificing much speed of convergence. Finally, we explored the software and hardware by-products of this journey, and talked about reproducible research. Although we are concluding here this thesis, we would like to look towards the future and propose a few avenues to continue further this adventure.

Practical Implementation of Noisy Sparse Fast Hadamard Transforms The learning parity with noise (LPN) problem is attracting attention in the cryptography community as it is thought to remain hard even in the post-quantum world [102]. Interestingly, the best known algorithms for the LPN make use of the WHT. In particular, the largest coefficient of the transform is of interest and fast algorithms to compute it could potentially lead to better strategies [79]. Because of the sheer size of problems in cryptography, this has motivated the implementation of high performance tera-scale fast Hadamard transform for signals of size up to $N = 2^{40}$ [78]. The SparseFHT algorithm has the potential to radically speed up the computation at such signal size. Considering that for such large N reading the data from disk is a major bottleneck, having a very small sampling complexity is a significant advantage. Albeit strategies have been proposed for noisy SparseFHT [28, 75], the trade-off between SNR and computational complexity is not yet very clear. In addition, special strategies for the extreme signal size of the LPN problem, where only the largest coefficient is needed might be devised. Practically, the modification of the algorithm to run on a distributed computing cluster is important.

Sparse and Fast Adaptive Filters As we have seen, adaptive filters are an important tool in signal processing. In many important cases, the unknown channel is sparse, or approximately sparse, for example the early part of room impulse responses or wireless channels. A number of methods have been proposed for sparse channels, some based on sparsity promoting regularization [29], or on FRI techniques [84]. These approaches generally add robustness to the algorithm by further constraining the optimization problem using sparsity, however they do not consider computational complexity. Due to similarity of the problem with the sparse fast Fourier transform [58], it is of interest to develop low-complexity algorithms for sparse recovery in the context of adaptive filters.

Psychoacoustically Aware Beamformers When designing beamformers for speech processing, objective metrics, such as the signal-to-interference-and-noise ratio (SINR), have been favored due to their mathematical tractability. Regardless, the human audition behaves significantly differently from what the SINR might suggest [138]. Acknowledging this, we have proposed in Chapter 3 one design with a psychoacoustically motivated constraint relaxation. In this initial step we only took into account that early echoes improve intelligibility to relax the distortionless constraint. However, echoes have in addition a sparse structure that we ignored. Moreover, we still optimized energy. What would be much better is to find a subjective measure that matches properties of the human audition and can be efficiently optimized.

Passive Room Impulse Response Estimation Room inference from acoustic measurements is possible using as little as a single impulse response [40]. However, the passive measurement of the RIR, and by extension the blind inference of the room shape, is a challenging problem. Early work on this topic focused on very short wireless channels and is thus difficult to apply to long RIRs [89, 123]. More recently Crocco et al. show some success with ℓ_1 norm constraints on examples that could be applicable to room reconstruction [33]. As we outlined in the last section of Chapter 4, direction of arrival estimation is intimately linked to RIR estimation and an approach based on FRI is possible. Furthermore, FRI constraints are very powerful and might be sufficient to overcome the difficulties of long RIRs.

Bibliography

- [1] 3GPP, *Spreading and modulation (FDD)*, 3rd Generation Partnership Project (3GPP), TS 25.213, Sept. 2014.
- [2] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions*. Courier Dover Publications, 1972.
- [3] N. Ailon and B. Chazelle, “The fast Johnson–Lindenstrauss transform and approximate nearest neighbors,” *SIAM Journal on computing*, vol. 39, no. 1, pp. 302–322, Jan. 2009.
- [4] J. B. Allen and D. A. Berkley, “Image method for efficiently simulating small-room acoustics,” *J. Acoust. Soc. Am.*, vol. 65, no. 4, pp. 943–950, 1979.
- [5] P. Annibale, F. Antonacci, P. Bestagini, A. Brutti, A. Canclini, L. Cristoforetti, J. Filos, E. Habets, W. Kellerman, K. Kowalczyk, A. Lombard, E. Mabande, D. Markovic, P. Naylor, and M. Omologo, “The SCENIC project: Space-time audio processing for environment-aware acoustic sensing and rendering,” in Proc. *131st Convention of the Audio Engineering Society*. New York, NY, USA: Audio Engineering Society, 2011.
- [6] F. Antonacci, J. Filos, M. R. P. Thomas, E. A. P. Habets, A. Sarti, P. A. Naylor, and S. Tubaro, “Inference of room geometry from acoustic impulse responses,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 20, no. 10, pp. 2683–2695, 2012.
- [7] J. Azcarreta Ortiz, “Pyramic array: An FPGA based platform for many-channel audio acquisition,” Master’s thesis, EPFL, Lausanne, Switzerland, Aug. 2016.
- [8] C. Ball, “Gittorrent,” <https://github.com/cjb/GitTorrent>, 2015.
- [9] A. Barabell, “Improving the resolution performance of eigenstructure-based direction-finding algorithms,” in Proc. *IEEE ICASSP*, vol. 8, pp. 336–339, Boston, MA, USA, Apr. 1983.
- [10] S. Barthe, “A python package for audio signal processing,” EPFL, Tech. Rep., 2015.
- [11] A. Beck, P. Stoica, and J. Li, “Exact and approximate solutions of source localization problems,” *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1770–1778, 2008.
- [12] F. Belloni, A. Richter, and V. Koivunen, “DoA estimation via manifold separation for arbitrary array structures,” *IEEE Trans. Signal Process.*, vol. 55, no. 10, pp. 4800–4810, Sept. 2007.

-
- [13] J. Benesty, J. Chen, Y. A. Huang, and J. Dmochowski, "On microphone-array beamforming from a MIMO acoustic signal processing perspective," *IEEE Trans., Audio, Speech, Language Process.*, vol. 15, no. 3, pp. 1053–1065, Mar. 2007.
- [14] D. Berberidis, V. Kekatos, and G. B. Giannakis, "Online censoring for large-scale regressions with application to streaming big data," *Signal Processing, IEEE Transactions on*, vol. 64, no. 15, pp. 3854–3867, May 2016.
- [15] A. Beregszaszi, "Mango," <https://github.com/axic/mango>, 2016.
- [16] S. Bogos and S. Vaudenay, "Optimization of LPN solving algorithms," in *Advances in Cryptology – ASIACRYPT 2016*. Springer Berlin Heidelberg, Nov. 2016, pp. 703–728.
- [17] J. Borish, "Extension of the image model to arbitrary polyhedra," *J. Acoust. Soc. Am.*, vol. 75, no. 6, pp. 1827–1836, 1984.
- [18] C. Boutsidis and P. Drineas, "Random projections for the nonnegative least-squares problem," *Linear algebra and its applications*, vol. 431, no. 5-7, pp. 760–771, 2009.
- [19] J. S. Bradley, H. Sato, and M. Picard, "On the importance of early reflections for speech in rooms," *J. Acoust. Soc. Am.*, vol. 113, no. 6, p. 3233, 2003.
- [20] Y. Bresler and A. Macovski, "Exact maximum likelihood parameter estimation of superimposed exponential signals in noise," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 5, pp. 1081–1089, Oct. 1986.
- [21] T. P. Bronez, "Sector interpolation of non-uniform arrays for efficient high resolution bearing estimation," in Proc. *IEEE ICASSP*, pp. 2885–2888, New York, NY, USA, 1988.
- [22] B. Bruneau, "Modular interface for embedded audio acquisition platforms," EPFL, Tech. Rep., 2016.
- [23] H. Buchner, R. Aichner, and W. Kellermann, "TRINICON: a versatile framework for multichannel blind signal processing," in Proc. *IEEE ICASSP*, pp. iii–889–92 vol.3, Montreal, 2004.
- [24] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," in Proc. *IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.
- [25] B. D. Carlson, "Covariance matrix estimation errors and diagonal loading in adaptive arrays," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 24, no. 4, pp. 397–401, July 1988.
- [26] R. H. Chan and M. K. Ng, "Conjugate gradient methods for Toeplitz systems," *SIAM Review*, vol. 38, no. 3, pp. 427–482, Sept. 1996.
- [27] T. F. Chan, "An optimal circulant preconditioner for Toeplitz systems," *SIAM Journal on Scientific and Statistical Computing*, vol. 9, no. 4, pp. 766–771, 1988.
- [28] X. Chen and D. Guo, "Robust sublinear complexity Walsh-Hadamard transform with arbitrary sparse support," in Proc. *IEEE ISIT*, pp. 2573–2577, 2015.
- [29] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in Proc. *IEEE ICASSP*, pp. 3125–3128, Taipei, Taiwan, 2009.

-
- [30] D. Colton and R. Kress, *Inverse acoustic and electromagnetic scattering theory*. Springer Science & Business Media, 2012, vol. 93.
- [31] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of computation*, pp. 297–301, 1965.
- [32] R. Couillet and M. Debbah, "Signal processing in large systems: A new paradigm," *IEEE Signal Processing Magazine*, vol. 30, no. 1, pp. 24–39, 2013.
- [33] M. Crocco and A. Del Bue, "Estimation of TDOA for room reflections by iterative weighted ℓ_1 constraint," in Proc. *IEEE ICASSP*, pp. 3201–3205, Shanghai, China, 2016.
- [34] E. D. di Claudio and R. Parisi, "WAVES: Weighted average of signal subspaces for robust wideband direction finding," *IEEE Trans. Signal Process.*, vol. 49, no. 10, pp. 2179–2191, Oct. 2001.
- [35] J. H. DiBiase, "A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays," Ph.D. dissertation, Brown University, Providence, RI, USA, 2000.
- [36] I. Dokmanić and Y. M. Lu, "Sampling sparse signals on the sphere: Algorithms and applications," *IEEE Trans. Signal Process.*, vol. 64, no. 1, pp. 189–202, January 2016.
- [37] I. Dokmanic, Y. M. Lu, and M. Vetterli, "Can one hear the shape of a room: The 2-D polygonal case," in Proc. *IEEE ICASSP*, pp. 321–324, Prague, 2011.
- [38] I. Dokmanić, R. Parhizkar, A. Walther, Y. M. Lu, and M. Vetterli, "Acoustic echoes reveal room shape," *Proc. Natl. Acad. Sci.*, vol. 110, no. 30, June 2013.
- [39] I. Dokmanić, R. Scheibler, and M. Vetterli, "Raking the cocktail party," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 5, pp. 825–836, 2015.
- [40] I. Dokmanić, "Listening to distances and hearing shapes," Ph.D. dissertation, EPFL, Lausanne, Switzerland, 2015.
- [41] P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlos, "Faster least squares approximation," *Numerische mathematik*, vol. 117, no. 2, pp. 219–249, Feb. 2011.
- [42] D. Duffy, *Green's Functions with Applications*. Chapman and Hall/CRC, 2001.
- [43] C. Ferry, "Extension board for CycloneV : Multi microphone acquisition - signal analysis (extending the Pyramic array)," EPFL, Tech. Rep., 2016.
- [44] I. Fette and A. Melnikov, "The websocket protocol," Internet Requests for Comments, RFC Editor, RFC 6455, December 2011, <http://www.rfc-editor.org/rfc/rfc6455.txt>.
- [45] E. D. Fredman and M. L. Nelson, "Hadamard spectroscopy," *J. Opt. Soc. Am.*, vol. 60, no. 12, pp. 1664–1669, Dec. 1970.
- [46] B. Friedlander, "The root-MUSIC algorithm for direction finding with interpolated arrays," *Signal Processing*, vol. 30, no. 1, pp. 15–29, 1993.
- [47] O. L. I. Frost, "An algorithm for linearly constrained adaptive array processing," in Proc. *IEEE*, pp. 926–935, 1972.

-
- [48] K. Furuya, “Noise reduction and dereverberation using correlation matrix based on the multiple-input/output inverse-filtering theorem (mint),” in Proc. *Intl. Workshop on HSC*, pp. 59–62, Kyoto, Japan, 2001.
- [49] B. Ghazi, H. Hassanieh, P. Indyk, D. Katabi, E. Price, and L. Shi, “Sample-optimal average-case sparse Fourier transform in two dimensions,” in Proc. *Allerton*, pp. 1258–1265, Oct. 2013.
- [50] A. C. Gilbert, M. J. Strauss, and J. A. Tropp, “A tutorial on fast Fourier sampling,” *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 57–66, 2008.
- [51] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss, “Near-optimal sparse Fourier representations via sampling,” in Proc. *STOC’02*, pp. 152–161, 2002.
- [52] O. Goldreich and L. A. Levin, “A hard-core predicate for all one-way functions,” in Proc. *STOC’89*, Feb. 1989.
- [53] O. Goldreich, *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Springer Science & Business Media, 1999.
- [54] E. Habets, J. Benesty, I. Cohen, S. Gannot, and J. Dmochowski, “New Insights Into the MVDR Beamformer in Room Acoustics,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 1, pp. 158–170, Jan. 2010.
- [55] E. A. Habets, “Room impulse response generator,” Technische Universiteit Eindhoven, Tech. Rep. 2.2.4, 01 2010.
- [56] S. Haghshatshoar and E. Abbe, “Polarization of the Rényi information dimension for single and multi terminal analog compression,” in Proc. *IEEE ISIT*, pp. 779–783. IEEE, 2013.
- [57] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, “Simple and practical algorithm for sparse Fourier transform,” in Proc. *SODA’12*, pp. 1183–1194, 2012.
- [58] —, “Nearly optimal sparse Fourier transform,” in Proc. *STOC’12*, pp. 563–578, 2012.
- [59] S. Haykin and Z. Chen, “The cocktail party problem,” *Neural Comput.*, vol. 17, no. 9, pp. 1875–1902, 2005.
- [60] S. Haykin, *Adaptive filter theory*. Prentice Hall, 2014.
- [61] P. J. Hayuningtyas and P. Marziliano, “Finite rate of innovation method for DOA estimation of multiple sinusoidal signals with unknown frequency components,” *Radar Conference (EuRAD)*, pp. 115–118, 2012.
- [62] A. Hedayat and W. Wallis, “Hadamard matrices and their applications,” *Ann. Stat.*, pp. 1184–1238, 1978.
- [63] M. Heideman, D. H. Johnson, and C. S. Burrus, “Gauss and the history of the fast Fourier transform,” *ASSP Magazine, IEEE*, vol. 1, no. 4, pp. 14–21, 1984.
- [64] F.-M. Hoffmann, F. M. Fazi, and P. Nelson, “Plane wave identification with circular arrays by means of a finite rate of innovation approach,” in Proc. *Audio Engineering Society Convention 140*, 2016.

-
- [65] H. Hotelling, "Some improvements in weighing and other experimental techniques," *The Annals of Mathematical Statistics*, 1944.
- [66] M. M. Hyder and K. Mahata, "Direction-of-arrival estimation using a mixed $\ell_{2,0}$ norm approximation," *IEEE Transactions Signal Process.*, vol. 58, no. 9, pp. 4646–4655, Aug. 2010.
- [67] ITU-T P.862 Amendment 2, "Reference implementations and conformance testing for ITU-T Recs P.862, P.862.1 and P.862.2," 11 2005. [Online]. Available: <https://www.itu.int/rec/T-REC-P.862-200511-I!Amd2/en>.
- [68] E.-E. Jan, P. Svaizer, and J. L. Flanagan, "Matched-filter processing of microphone array for spatial volume selectivity," *Proc. IEEE ISCAS*, vol. 2, pp. 1460–1463, 1995.
- [69] J. R. Johnson and M. Püschel, "In search of the optimal Walsh-Hadamard transform," in *Proc. Proc. IEEE ICASSP*, vol. 6, pp. 3347–3350, Istanbul, June 2000.
- [70] B. H. Khalaj, A. Paulraj, and T. Kailath, "2D RAKE receivers for CDMA cellular systems," in *Proc. IEEE GLOBECOM*, pp. 400–404, 1994.
- [71] G. King, "An introduction to the Dataverse Network as an infrastructure for data sharing," 2007.
- [72] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 24, no. 4, 1976.
- [73] D. Lawlor, Y. Wang, and A. Christlieb, "Adaptive sub-linear Fourier algorithms," *Adv. Adapt. Data Anal.*, vol. 05, no. 01, p. 1350003, 2013.
- [74] M. H. Lee and M. Kaveh, "Fast Hadamard transform based on a simple matrix factorization," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 6, pp. 1666–1667, 1986.
- [75] X. Li, J. K. Bradley, S. Pawar, and K. Ramchandran, "The SPRIGHT algorithm for robust sparse Hadamard transforms," in *Proc. IEEE ISIT*, pp. 1857–1861, Honolulu, 2014.
- [76] —, "SPRIGHT: A fast and robust framework for sparse Walsh-Hadamard transform," *arXiv preprint arXiv:1508.06336*, 2015.
- [77] J. Lochner and J. F. Burger, "The influence of reflections on auditorium acoustics," *J. Sound Vib.*, vol. 1, no. 4, pp. 426–454, 1964.
- [78] Y. Lu, "Practical tera-scale Walsh-Hadamard transform," *arXiv preprint arXiv:1607.01039*, 2016.
- [79] Y. Lu and Y. Desmedt, "Walsh transforms and cryptographic applications in bias computing," *Cryptography and Communications*, pp. 1–19, Apr. 2016.
- [80] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, 2001.
- [81] W.-K. Ma, T.-H. Hsieh, and C.-Y. Chi, "DOA estimation of quasi-stationary signals with less sensors than sources and unknown spatial noise covariance: A Khatri–Rao subspace approach," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2168–2180, Mar. 2010.

-
- [82] D. Malioutov, M. Cetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3010–3022, July 2005.
- [83] V. A. Marčenko and L. A. Pastur, "Distribution of eigenvalues for some sets of random matrices," *Mathematics of the USSR-Sbornik*, vol. 1, no. 4, pp. 457–483, 1967.
- [84] M. McCormick, Y. M. Lu, and M. Vetterli, "Learning sparse systems at sub-Nyquist rates: A frequency-domain approach," in Proc. *IEEE ICASSP*, pp. 4018–4021, Dallas, TX, 2010.
- [85] W. McCrea and F. Whipple, "Random paths in two and three dimensions," in Proc. *Proc. Roy. Soc. Edinburgh*, vol. 60, pp. 281–298, 1940.
- [86] A. McPherson and V. Zappi, "An environment for submillisecond-latency audio and sensor processing on BeagleBone Black," in Proc. *AES Convention 138*, pp. 1–7, Feb. 2015.
- [87] M. Miyoshi and Y. Kaneda, "Inverse filtering of room acoustics," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 36, no. 2, pp. 145–152, 1988.
- [88] F. Mondada, "Results held hostage: Hardware design software licenses holding back open science," <https://blogs.openaire.eu/?p=882>, 2016.
- [89] E. Moulines, P. Duhamel, J. F. Cardoso, and S. Mayrargue, "Subspace methods for the blind identification of multichannel FIR filters," *IEEE Trans. Sig. Processing*, vol. 43, no. 2, pp. 516–525, Feb. 1995.
- [90] M. R. Munafò, B. A. Nosek, D. V. M. Bishop, K. S. Button, C. D. Chambers, N. Percie du Sert, U. Simonsohn, E.-J. Wagenmakers, J. J. Ware, and J. P. A. Ioannidis, "A manifesto for reproducible science," *Nature Human Behaviour*, vol. 1, no. 1, p. 0021, Jan. 2017.
- [91] A. F. Naguib, "Space-time receivers for CDMA multipath signals," in Proc. *Proc. IEEE ICC*, pp. 304–308. Montreal: IEEE, 1997.
- [92] P. A. Naylor and N. D. Gaubitch, eds., *Speech Dereverberation*. Springer London, 2010.
- [93] M. K. Ng and R. J. Plemmons, "Fast recursive least squares adaptive filtering by fast Fourier transform-based conjugate gradient iterations," *SIAM Journal on Scientific Computing*, vol. 17, no. 4, pp. 920–941, 1996.
- [94] O. Öçal, I. Dokmanić, and M. Vetterli, "Source localization and tracking in non-convex rooms," in Proc. *IEEE ICASSP*, pp. 1429–1433, Florence, Italy, 2014.
- [95] A. E. O'Donovan, R. Duraiswami, and D. N. Zotkin, "Automatic matched filter recovery via the audio camera," in Proc. *IEEE ICASSP*, pp. 2826–2829, Dallas, 2010.
- [96] T. E. Oliphant, "Python for scientific computing," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 10–20, 2007.
- [97] H. Pan, T. Blu, and M. Vetterli, "Towards generalised FRI sampling with an application to source resolution in radioastronomy," *IEEE Trans. Signal Process.*, vol. 65, no. 4, 2017.
- [98] H. Pan, R. Scheibler, E. Bezzam, I. Dokmanić, and M. Vetterli, "FRIDA: FRI-based DOA estimation for arbitrary array layouts," in Proc. *IEEE ICASSP*, New Orleans, LA, USA, 2017, to appear.

-
- [99] S. Pawar and K. Ramchandran, “A hybrid DFT-LDPC framework for fast, efficient and robust compressive sensing,” in Proc. *Allerton*, pp. 1943–1950, Monticello, IL, USA, 2012.
- [100] —, “Computing a k -sparse n -length discrete Fourier transform using at most $4k$ samples and $o(k \log k)$ complexity,” in Proc. *IEEE ISIT*, pp. 464–468, Istanbul, July 2013.
- [101] J. Perkel, “Democratic databases: science on GitHub,” *Nature*, vol. 538, no. 7623, pp. 127–128, Oct. 2016.
- [102] K. Pietrzak, “Cryptography from learning parity with noise,” in *SOFSEM 2012: Theory and Practice of Computer Science*, M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, and G. Turán, eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 99–114.
- [103] M. Pilanci and M. J. Wainwright, “Iterative hessian sketch: fast and accurate solution approximation for constrained least-squares,” *The Journal of Machine Learning Research*, vol. 17, pp. Paper No. 53–38, 2016.
- [104] W. Pratt, J. Kane, and H. C. Andrews, “Hadamard transform image coding,” in Proc. *IEEE*, pp. 58–68, 1969.
- [105] R. Price and P. E. Green, “A communication technique for multipath channels,” in Proc. *IRE*, pp. 555–570, 1958.
- [106] R. Prony, “Essai expérimental et analytique,” *Journal de l’Ecole Polytechnique*, vol. 1, no. 2, p. 24, 1795.
- [107] B. Rafaely, *Fundamentals of Spherical Array Processing*, Springer Topics in Signal Processing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, vol. 8.
- [108] F. Ribeiro, D. A. Florencio, D. E. Ba, and C. Zhang, “Geometrically constrained room modeling with compact microphone arrays,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 20, no. 5, pp. 1449–1460, 2012.
- [109] T. Richardson and R. L. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.
- [110] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, “Perceptual evaluation of speech quality (PESQ)—A new method for speech quality assessment of telephone networks and codecs,” in Proc. *IEEE ICASSP*, pp. 749–752, Salt Lake City, UT, 2001.
- [111] M. Rubsamen and A. B. Gershman, “Direction-of-arrival estimation for nonuniform sensor arrays: From manifold separation to fourier domain music methods,” *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 588–599, Jan. 2009.
- [112] R. Scheibler, I. Dokmanić, and M. Vetterli, “Raking echoes in the time domain,” in Proc. *IEEE ICASSP*, Brisbane, Australia, 2015.
- [113] R. Scheibler, S. Haghghatshoar, and M. Vetterli, “A fast Hadamard transform for signals with sub-linear sparsity,” in Proc. *Allerton*, Monticello, IL, USA, 2013.
- [114] —, “A fast Hadamard transform for signals with sub-linear sparsity in the transform domain,” *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 2115–2132, 2015.

-
- [115] R. Scheibler and M. Vetterli, "The recursive Hessian sketch for adaptive filtering," in Proc. *IEEE ICASSP*, Shanghai, China, 2016.
- [116] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. 34, no. 3, pp. 276–280, 1986.
- [117] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Carnegie Mellon University, Tech. Rep., 1994.
- [118] J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Process. Mag.*, vol. 9, no. 1, pp. 14–37, 1992.
- [119] V. Stodden, C. Hurlin, and C. Pérignon, "Runmycode.org: a novel dissemination and collaboration platform for executing published computational results," in Proc. *IEEE 8th Int. Conf. E-Science*, pp. 1–8. IEEE, 2012.
- [120] V. Stodden and S. Miguez, "Best practices for computational science: Software infrastructure and environments for reproducible and extensible research," *Journal of Open Research Software*, vol. 2, no. 1, July 2014.
- [121] I. J. Tashev, *Sound Capture and Processing*, Practical Approaches. Chichester, UK: John Wiley & Sons, July 2009.
- [122] M. R. P. Thomas, N. D. Gaubitch, and P. A. Naylor, "Application of channel shortening to acoustic channel equalization in the presence of noise and estimation error," in Proc. *IEEE WASPAA*, pp. 113–116, New Paltz, NY, USA, 2011.
- [123] L. Tong, G. Xu, and T. Kailath, "Blind identification and equalization of multipath channels," in Proc. *SUPERCOMM/ICC*, pp. 1513–1517, 1992.
- [124] J. W. Topliss, V. Zappi, and A. McPherson, "Latency performance for real-time audio on Beaglebone Black," in Proc. *Linux Audio Conference*, Karlsruhe, Germany, 2014.
- [125] B. D. Van Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Mag.*, vol. 5, no. 2, pp. 4–24, 1988.
- [126] M. Vetterli, P. Marziliano, and T. Blu, "Sampling signals with finite rate of innovation," *IEEE Trans. Signal Process.*, vol. 50, no. 6, pp. 1417–1428, 2002.
- [127] H. Wang and M. Kaveh, "Coherent signal-subspace processing for the detection and estimation of angles of arrival of multiple wide-band sources," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 33, no. 4, pp. 823–831, Aug. 1985.
- [128] J. Wang, J. D. Lee, M. Mahdavi, M. Kolar, and N. Srebro, "Sketching meets random projection in the dual: A provable recovery algorithm for big and high-dimensional data," *arXiv.org*, Oct. 2016.
- [129] D. B. Ward, E. A. Lehmann, and R. C. Williamson, "Particle filtering algorithms for tracking an acoustic source in a reverberant environment," *IEEE Trans. Audio, Speech, Language Process.*, vol. 11, no. 6, pp. 826–836, 2003.
- [130] B. Widrow, "Thinking about thinking: the discovery of the LMS algorithm," *IEEE Signal Processing Magazine*, vol. 22, no. 1, pp. 100–106, 2005.

-
- [131] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Convention Record Part IV*, pp. 96–104, 1960.
- [132] G. Wilson, D. A. Aruliah, C. T. Brown, N. P. Chue Hong, M. Davis, R. T. Guy, S. H. D. Haddock, K. D. Huff, I. M. Mitchell, M. D. Plumbley, B. Waugh, E. P. White, and P. Wilson, "Best practices for scientific computing," *PLOS Biology*, vol. 12, no. 1, pp. 1–7, Jan. 2014.
- [133] M. A. Woodbury, "Inverting modified matrices," *Memorandum report*, vol. 42, p. 106, 1950.
- [134] N. C. Wormald, "Differential equations for random processes and random graphs," *Ann. Prob.*, vol. 5, no. 4, pp. 1217–1235, Nov. 1995.
- [135] F. Yates, "Complex experiments," *Supplement to the Journal of the Royal Statistical Society*, vol. 2, no. 2, p. 181, 1935.
- [136] Y.-S. Yoon, L. M. Kaplan, and J. H. McClellan, "TOPS: New DOA estimator for wideband signals," *IEEE Trans. Signal Process.*, vol. 54, no. 6, pp. 1977–1989, May 2006.
- [137] W. Zhang, E. Habets, and P. A. Naylor, "On the use of channel shortening in multichannel acoustic system equalization," in *Proc. Proc. IWAENC*, Tel Aviv, 2010.
- [138] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and models*. Springer Science & Business Media, 2013, vol. 22.

Robin SCHEIBLER

Rue du Clos-de-Bulle 11
1004 Lausanne, Switzerland
Tel: +41 78 634 1675
robin.scheibler@epfl.ch

Swiss citizen
Born August 8th 1984
<http://www.robinscheibler.org>
github: @fakufaku

EDUCATION

- 2012–2017 Ph.D. in Signal Processing**
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
Adviser: Prof. Martin Vetterli
- 2003–2009 B.Sc. / M.Sc. in Communication Systems**
Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland
- 2005–2006 Erasmus exchange**
Royal Institute of Technology, Stockholm (KTH), Sweden

INDUSTRY

- Media Information Processing Laboratory, NEC Corporation, Japan**
January 2011 – August 2012, Research Engineer
Single channel speech denoising with very high perceptual quality for mobile applications.
- IBM Research – Zürich, Switzerland**
February 2009 – September 2010, Research Engineer
Algorithms for the computation of continuous Haar, Fourier and cosine transforms of rectilinear polygons from IC layouts with applications in computational lithography.
- System IP Core Laboratory, NEC Corporation, Japan**
March 2007 – January 2008, Intern
Adaptive channel equalization for 3G communications on a reconfigurable processor architecture.

CITIZEN SCIENCE

- Biodesign for the Real World, India/Indonesia/Switzerland**
September 2012 – Present, <http://biodesign.cc>
An interdisciplinary and collaborative research project addressing real world water problems by defining, building, and field-testing prototypes that integrate wetware, hardware, and software.
- SAFECAST Japan, Tokyo, Japan**
March 2011 – Present, <http://safecast.org>
Development of a novel mobile radiation sensing platform in the weeks following the meltdown at Fukushima Dai-Ichi in March 2011. In the past five years, hundreds of sensors have been deployed and over 50 million measurement points collected worldwide by volunteers using this system.

AWARDS AND HONORS

- 2013** Good Design Award for the SAFECAST radiation detection network (as technical director)
- 2012** EPFL School of Computer and Communication Sciences fellowship

TEACHING AND SUPERVISION

Teaching assistant for digital signal processing, audio signal processing and virtual acoustics, calculus, introduction to computer and communication sciences

Supervising one master thesis (An FPGA based platform for many-channel audio acquisition, Juan Azcarreta-Ortiz, undergoing)

Supervised 20 student semester projects

Instructor at winter school *Biodesign for the Real World*, 20 participants, UNIL, Lausanne, Switzerland, 2016

Instructor of *Reproducible Research using IPython Interactive Publications* workshop, 30 participants, EPFL, Lausanne, Switzerland, 2015

Instructor of SAFecast Geiger counter building workshop, 10 participants, Strasbourg FabLab, France, 2014

RESEARCH INTERESTS

Algorithms for signal processing (sparse transforms, adaptive filters)

Computational and spatial acoustics

Citizen and DIY approaches to science

SKILLS

Technical C/C++, Python, \LaTeX , Matlab, Eagle CAD

Languages French, English, Japanese, German (conversational), Swedish (basic)

SELECTED PRESS FEATURES

In Vivo Magazine, "La science hors des sentiers battus," December 2015.

http://www.invivomagazine.com/fr/mens_sana/en_images/article/225/la-science-hors-des-sentiers-battus

C. Edwards, "Brain science helps computers separate speakers in a crowded room," *Communications of the ACM*, vol. 58, no. 11, November 2015.

Hackaday, "DIY Incubator Cooks Bacteria... Or Yogurt!" December 30, 2013.

<http://hackaday.com/2013/12/30/diy-incubator-cooks-bacteria-or-yogurt/>

Radio Télévision Suisse, "Chasseur de radioactivité à Fukushima," September 2012.

<http://www.nouvo.ch/2012/09/chasseur-de-radioactivit%C3%A9-%C3%A0-fukushima>

PROFESSIONAL ACTIVITIES

Reviewer for *IEEE Transactions on Circuits and Systems II: Express Letters*, *IEEE International Symposium on Information Theory*, Elsevier *Digital Signal Processing*

Member of the IEEE Signal Processing Society

INVITED PRESENTATIONS

Safecast: Crowd-sourced citizen-driven mobile sensing of radiation, *NCCR MICS Final Event*, EPFL, Lausanne, Switzerland, September 5, 2012.

Journal Papers

- [15] I. Dokmanić, R. Scheibler, and M. Vetterli, “Raking the cocktail party”, *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 5, pp. 825–836, 2015.
- [14] R. Scheibler, S. Haghghatshoar, and M. Vetterli, “A fast Hadamard transform for signals with sub-linear sparsity in the transform domain”, *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 2115–2132, 2015.
- [13] R. Scheibler, P. Hurley, and A. Chebira, “Fast continuous Fourier and Haar transforms of rectilinear polygons from very-large-scale integration layouts”, *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 12, no. 4, p. 043 008, 2013.

Conference Papers

- [12] H. Pan, R. Scheibler, E. Bezzam, I. Dokmanić, and M. Vetterli, “FRIDA: FRI-based DOA estimation for arbitrary array layouts”, in *IEEE ICASSP*, to appear, New Orleans, LA, USA, 2017.
- [11] R. Scheibler and M. Vetterli, “The recursive Hessian sketch for adaptive filtering”, in *IEEE ICASSP*, Shanghai, China, 2016.
- [10] R. Scheibler, I. Dokmanić, and M. Vetterli, “Raking echoes in the time domain”, in *IEEE ICASSP*, Brisbane, Australia, 2015.
- [9] R. Scheibler, S. Haghghatshoar, and M. Vetterli, “A fast Hadamard transform for signals with sub-linear sparsity”, in *Allerton*, Monticello, IL, USA, 2013.
- [8] M. Martinez-Camara, I. Dokmanić, J. Ranieri, R. Scheibler, M. Vetterli, and A. Stohl, “The Fukushima inverse problem”, in *IEEE ICASSP*, Vancouver, Canada, 2013.
- [7] R. Scheibler and P. Hurley, “Computing exact Fourier series coefficients of IC rectilinear polygons from low-resolution fast Fourier coefficients”, in *Proc. SPIE 8326, Optical Microlithography XXV*, 2012, p. 83262V.
- [6] R. Scheibler, P. Hurley, and A. Chebira, “Pruned continuous Haar transform of 2D polygonal patterns with application to VLSI layouts”, in *Proc. of the 2010 IRAST Int. Cong. on Comp. App. dn Computational Sci.*, 2010, pp. 984–987.
- [5] R. Scheibler, J. Okello, K Seki, T Kobori, and M Ikekawa, “Interference mitigation for WCDMA using QR decomposition and a CORDIC-based reconfigurable systolic array”, in *IEICE Tech. Rep.*, vol. 107, 2008, pp. 43–48.

Patents

- [4] I. Dokmanić, R. Scheibler, and M. Vetterli, *Optimal acoustic rake receivers*, US Patent, US 20160018510, 2016.
- [3] R. Scheibler, S. Haghghatshoar, and M. Vetterli, *Method for determining the Walsh-Hadamard transform of N samples of a signal and apparatus for performing the same*, US Patent, US 20150098313, 2015.
- [2] R. Scheibler, P. Hurley, K. Kryszczuk, and D. Schipani, *Method and system for computing Fourier series coefficients for mask layouts using FFT*, US Patent, US 8402399 B2, 2013.
- [1] K. Kryszczuk, P. Hurley, R. Scheibler, and J. Ranieri, *Assessing printability of a very-large-scale integration design*, US Patent, US 8327312 B2, 2012.