Sparse Representations of Audio Signals with Asymmetric Atoms

Julian Neri

Department of Music Research
Schulich School of Music
McGill University
Montreal, Canada

December 2017

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of the Arts.

Abstract

Sparse representations of audio is a mature field that has evolved from decades of research that established its utility in a variety of applications, for example, audio coding, source-separation, and transformation. However, the performance of sparse approximation algorithms still depend highly on signal length, which is problematic for audio signals with durations of more than a few seconds. Furthermore, there is a need for elementary waveforms, *atoms*, that can effectively adapt to represent temporally asymmetric features. Atoms with temporally asymmetric amplitude evolutions have already shown promise in sparse representation applications, namely the gammatone and formant-wave-function, however, due to their origins from outside the sparse realm, they either cannot adapt to model a wide range of audio features or their mathematical definition reduces the speed of the approximation process.

This thesis addresses these crucial aspects of sparse audio representations. We establish desirable atom properties, for example, mathematical properties that enable efficient parameter estimations and an analytic inner product formula, then compare existing atoms using our criteria to highlight their relative strengths and weaknesses. We establish a new asymmetric atom, the *ramped exponentially damped sinusoid* (REDS), that can model salient audio signal features, especially transients and decaying oscillations, and has all the properties we desire. Results from an experiment show that it can more sparsely represent audio than existing atoms and mathematical proofs show how we can tune the parameters of a REDS such that it approximately equals either existing asymmetric atom.

We introduce a new sparse approximation system, *Partial Trajectory Matching Pursuit* (PTMP), that employs sinusoidal partial tracking to locate long duration atoms and, in parallel, a small-scale sub-dictionary pursuit that locates short duration atoms. PTMP effectively locates arbitrarily long duration atoms, something that previous algorithms have not addressed, and, since these atoms match closely with the audio signal, they increase the sparsity of representation. We establish several estimation methods that work within PTMP to refine the REDS parameter set, which increase representation sparsity even further. Results from a series of experiments that gauge PTMP's performance show that PTMP is a powerful sparse approximation system that manages to avoid pre-echo and produce state-of-the-art sparsity levels by decomposing audio onto REDS atoms at high-speed.

Résumé

La représentation parcimonieuse des signaux sonores est un thème de recherche bien établi ayant bénéficié de plusieurs décennies de travaux qui ont consacrés son usage dans de nombreuses applications, comme par example le codage audio, la séparation de sources, et les transformations sonores. Cependant, la performance des algorithmes d'approximations parcimonieuses dépend encore fortement de la longueur du signal, ce qui est problématique dans le cas des signaux sonores dont la durée est souvent supérieure à quelques secondes. De plus, il y a un besoin de nouvelles formes d'onde élémentaires, ou atomes, qui puissent représenter et s'ajuster aux caractéristiques temporelles asymétriques des signaux sonores. L'utilisation, dans le cadre d'applications de décompositions parcimonieuses, d'atomes à évolution temporelle d'amplitude asymétrique tels que les gammatones ou bien les fonctions d'onde formantiques, a déjà donné lieu à des résultats prometteurs. Cependant, comme ils proviennent de domaines d'étude extérieurs à celui des décompositions parcimonieuses, ils ne peuvent pas s'ajuster pour modéliser une large classe de caractéristiques sonores ou bien leur expression mathématique ne permet pas de mettre en place un processus d'approximation rapide.

Dans cette thèse, nous nous sommes donc intéressés à ces aspects cruciaux des représentations parcimonieuses des signaux audio. Nous avons dressé une liste des propriétés souhaitables des atomes, comme par example les propriétés mathématiques facilitant une estimation efficace des paramètres, et celles menant à une expression analytique du produit scalaire entre atomes; nous avons ensuite comparé les types d'atomes existants selon les critères précédemment établis afin de mettre en lumière leurs avantages et leurs défauts. Nous proposons un nouveau type d'atomes asymétriques, que nous appelons REDS, apte à modéliser les caractéristiques pertinentes des signaux audio, plus spécialement les transitoires et les oscillations amorties, tout en possédant les propriétés de la liste que nous avons établie. Nos résultats expérimentaux montrent que l'utilisation des atomes REDS mène à des modélisations plus parcimonieuses que celles reposant sur les atomes asymétriques pré-existants; de plus nous établissons le lien mathématique montrant comment ajuster les paramètres des atomes REDS afin d'approximer voire même égaler les atomes symétriques pré-existants.

Par ailleurs, nous proposons un nouveau système d'approximation parcimonieuse, que nous appelons *Partial Trajectory Matching Pursuit* (PTMP), reposant sur l'extraction de

trajets de partiels sinusoïdaux afin de localiser les atomes de longue durée, tout en menant simultanément une poursuite sur un sous-dictionnaire d'atomes à petite échelle afin de localiser des atomes de courte durée. PTMP localise effectivement des atomes de durée arbitrairement longue, ce que les algorithmes existants ne détectent pas; et puisque ces longs atomes sont en bonne adéquation avec le signal audio, leur utilisation accroît la parcimonie de la représentation. Nous avons établi et mis en place plusieurs méthodes d'estimation qui affinent le jeux des paramètres REDS au sein de l'algorithme PTMP ce qui accroît encore la parcimonie. Enfin lors d'une série d'expériences destinées à mettre l'algorithme PTMP à l'épreuve, les résultats obtenus montrent que PTMP est un système d'approximation parcimonieuse puissant qui prévient l'apparition de pré-echos et produit une décomposition parcimonieuse comparable à l'état de l'art des autres méthodes tout en assurant une décomposition rapide des signaux audio sur les atomes REDS.

Acknowledgements

I thank my supervisor Professor Philippe Depalle for his knowledge and enthusiasm that inspires my research. I am extremely grateful for the opportunity to study under his expert supervision and support. Several funding sources have allowed me to direct my efforts towards this thesis research: Department of Music Research teaching assistantships, CIRMMT, a research stipend awarded by Philippe Depalle through NSERC, Union Memorial Church, and a research assistantship awarded by Professor Marcelo Wanderley. I am fortunate to have been a student in Gary Scavone, Philippe Depalle, and Marcelo Wanderley's enlightening seminars, and, more generally, of the Music Technology faculty's outstanding teaching. I thank Darryl Cameron for his help during the last two years and for friendships within the Music Technology Area. Finally, I thank my family for their love and encouragement.

Acronyms

Symbol	Description	First Use
REDS	ramped exponentially damped sinusoid	5
STFT	short-time Fourier transform	8
DFT	discrete Fourier transform	8
MP	Matching Pursuit	11
FFT	fast Fourier transform	14
DWT	discrete Wavelet transform	14
NM	Newton's method	17
RRM	recursive reassignment method	18
DS	damped sinusoid	30
FOF	formant-wave-function	35
SRR	signal-to-residual ratio	42
PTMP	partial tracking matching pursuit	45
dB	decibels	46
RIP	recursive inner product	56

Notation

Symbol	Description	First Use
$\mathbf{y}$	audio signal	xv
$\mathbf{\Phi}$	dictionary	7
$\boldsymbol{\phi}$	atom	7
$\mathbf{x}$	solution	7
$\mathbf{r}$	residual audio signal	8
$\boldsymbol{\lambda}$	atom parameter set	15
i	imaginary unit, $\sqrt{-1}$	19
τ	time shift $\in \mathbb{R}$	20
f_c	frequency of oscillation (normalized)	24
n	discrete time	24
α	decay (bandwidth) parameter	24
$u[n]$	unit step function, $u[n] = 1$ for $n \geq 0$, $u[n] = 0$ otherwise	24
$A[n]$	attack envelope	24
n_I	attack envelope influence time	27
Δ_I	time difference between n_I and n_M	27
n_m	time location of the atom's envelope maximum	27
Δt	time spreading	28
Δf	frequency spreading	28
$E[n]$	amplitude envelope	28
p	asymmetric atom order	32
k	iteration count ($\in \mathbb{N}^K$)	45
$\mathbf{P}$	partial trajectory	46
ρ	partial trajectory index	46

Symbol	Description	First Use
κ	frequency bin (index)	46
H	STFT hop size	46
G_h	height amplitude threshold for peak-picking	47
G_g	general amplitude threshold for peak-picking	47
$h[n]$	impulse response	50
ω_δ	peak-to-peak frequency constraint	51
ω_Δ	partial trajectory frequency constraint	51
ν_δ	valley threshold, for splitting partial trajectories	53
γ	sub-system index, either L (large-scale) or S (small-scale)	64
O	best option out of γ; $\lvert x_{\mathrm{O}}\rvert$ is greatest	65

Types	Description
y	Scalar
$y[n]$	Discrete time signal
$y(t)$	Continuous time signal counterpart to $y[n]$
$\mathbf{y}$	Bold lowercase denotes a vector
$\mathbf{Y}$	Bold uppercase denotes a matrix

Operator	Description
$A \gg B$	A is much bigger than B
$A \ll B$	A is much smaller than B
$x^{(k)}$	Value of x after iteration k
$\mathbf{x}^\intercal$	Transpose of $\mathbf{x} \in \mathbb{R}^{\mathrm{N}}$; $(\mathbf{x}^\intercal)_{qj} = \mathbf{x}_{jq}$
$\lvert x\rvert$	Absolute value of x
$\bar{z}$	Complex conjugate of $z \in \mathbb{C}$
$\Re\{z\}$	Real part of $z \in \mathbb{C}$
$\Im\{z\}$	Imaginary part of $z \in \mathbb{C}$
$\boldsymbol{\phi}^{\mathrm{H}}$	Conjugate transpose of $\boldsymbol{\phi} \in \mathbb{C}^{\mathrm{N}}$; $(\boldsymbol{\phi}^{\mathrm{H}})_{qj} = \bar{\boldsymbol{\phi}}_{jq}$
$\phi'(\lambda)$	derivative of ϕ with respect to λ
$\nabla_{\boldsymbol{\lambda}}\phi(\boldsymbol{\lambda})$	gradient of ϕ with respect to $\boldsymbol{\lambda}$
$\mathbf{H}_{\boldsymbol{\lambda}}\phi(\boldsymbol{\lambda})$	Hessian matrix of ϕ

$\hat{x}$	Estimate of x
$\tilde{n}$	Index output from some argument, e.g., $\tilde{n} = \arg\max_n \lvert x[n] \rvert$
$\lVert \mathbf{x} \rVert_p$	ℓ_p norm
$\langle \mathbf{y}, \boldsymbol{\phi} \rangle$	Inner product between discrete signals $y[n]$ and $\phi[n]$: $\langle \mathbf{y}, \boldsymbol{\phi} \rangle = \sum_{n=-\infty}^{+\infty} y[n]\bar{\phi}[n]$
$\mathbf{y} \odot \boldsymbol{\phi}$	Element-wise multiplication
$\mathbf{y} * \boldsymbol{\phi}$	Discrete time convolution
$\mathbf{y} \star \boldsymbol{\phi}$	Discrete time cross-correlation
$\mathcal{F}[\boldsymbol{\phi}](\omega)$	Fourier transform of $\phi(t)$: $\int_{-\infty}^{+\infty} \phi(t) e^{-i\omega t} dt$
$\mathcal{F}[\boldsymbol{\phi}](\kappa)$	discrete Fourier transform of $\phi[n]$: $\sum_{n=0}^{\mathrm{N}-1} \phi[n] e^{-i2\pi\kappa n/\mathrm{N}}$
$\mathcal{Z}[\boldsymbol{\phi}](z)$	Z-transform of $\phi[n]$: $\sum_{n=-\infty}^{+\infty} \phi[n] z^{-n}$

Sets	**Description**
$\mathbb{N}$	Positive integers including 0
$\mathbb{Z}$	Integers
$\mathbb{R}$	Real numbers
$\mathbb{R}_{\geq 0}$	Positive real numbers including 0
$\mathbb{C}$	Complex numbers

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Vocabularies of natural languages contain words with similar meanings that enable the communication of various ideas and allow us to simultaneously differentiate between nearly identical concepts. Alternatively, consider text written with a small vocabulary; the vocabulary might be sufficient to express any idea, but only through full sentence explanations of the unknown words. The same idea holds for representing (describing) sound. An audio representation is broadly any way to describe an audio signal, which can be through the variations of its amplitude over time (its time-domain representation), its frequency content evolution over time (its time-frequency representation), or more generally, any set of data that either directly or indirectly describes the sound that some converter may take as instructions to synthesize it. We extract a representation of an existing signal by comparing it with elementary waveforms (atoms) of a dictionary. A basis expansion, for example through the Fourier basis, sufficiently describes any signal, though the expansions diffuse information makes pattern identification difficult. We can sparsely represent an arbitrary audio signal when our vocabulary of sounds, our dictionary, is highly redundant and includes sounds that match closely with the audio signal [1]. In general terms, the sparse representation problem, as it pertains to audio processing, is as follows; synthesize an audio signal from the combination of as few other sounds as possible.

Sparse representations of audio is a mature field that has evolved from decades of research that established its theoretical foundation and proved its utility in a wide array of applications. Explorations of practical algorithms to extract sparse approximations from arbitrary signals resulted in the establishment of several methods, including algorithms like

Matching Pursuit and Basis Pursuit [1] [2], and theoretical performance evaluations of these methods [3] [4] [5]. To fuel numerical approximation methods, research has also focused on ways to design a dictionary around a signal to ensure it contains atoms that closely match with the signal. This generally involves either concatenating several bases together, learning a dictionary from bulk data via some machine learning method [6], from the sampling of parametric time-frequency atoms [1], or by a hybrid approach using some combination of the three. Finally, a significant portion of papers about sparse audio approximations have covered an extensive list of applications, including audio coding, source-separation, automatic music transcription, visualization, transformation, and de-noising [7] [8] [9].

Although sparse approximation algorithms have been subject to decades of research, the inherent complexity of calculating correlations between the dictionary elements and audio signal, and searching among the dictionary for the highest correlation, has limited their ability to quickly decompose sounds with long durations, which is a major setback for decomposing musical audio that lasts more than a few seconds. Thus, there is a real need for sparse approximation algorithms whose performance depends less on audio signal size. Even the fastest implementation of matching pursuit, Matching Pursuit Toolkit (MPTK) [10], takes hundreds of times longer than the duration of the audio signal when the quality is set high. Besides the fast algorithm methods that MPTK implements, another step towards a faster algorithm came from Daudet [11], who noted that a limiting situation of the algorithm is when comparing long duration atoms with the audio signal, and established that decomposing the signal onto molecules, or strings of multiple similar atoms, increases the decomposition speed. Parallel computing may help to make sparse decompositions of full musical pieces (minutes long) a possibility, as [12] proposed. Besides these alternative techniques, a fundamental way to increase a greedy decomposition's speed is by representing more of the signal with each atom, effectively reducing the number of atoms (iterations) overall. This crucial aspect of creating a sparse representation, whether it be for approximating an existing signal or synthesizing a new one, puts the spotlight back on dictionary design; to determine sounds that model an arbitrary audio signal well.

An oscillation with an amplitude that rises faster than it decays is an appropriate model for sound from mechanical or acoustical vibrations, since these vibrations always decrease over time due to damping, roughly exponentially, after a momentary activation from an external force. Indeed, a faster rise time than decay time means the oscillation's temporal evolution is asymmetric. However, it is common practice to represent sound

with temporally symmetric waveforms, specifically Gabor atoms; to trade coherence with physical reality for desirable mathematical properties [13] [14] [15]. Although certainly justifiable, the representation of asymmetric signal content with symmetric atoms will either be non-sparse or contain energy before the onset of an oscillation that is not present in the original sound, in other words, dark energy and/or pre-echo [16]. Several papers aim to limit pre-echo by constraining the choice of the symmetric atoms location in time such that it is after the onset [16] [17] [18]. While these techniques may help to avoid pre-echo, they still require a non-sparse number of symmetric atoms to represent strong transients and asymmetric features that are inherent to natural and musical sounds, especially at event onsets, because, fundamentally, symmetric atoms do not sufficiently model temporally asymmetric content.

Growing interest in the use of asymmetric atoms to sparsely represent audio started after a proposal by Goodwin, who advocated the use of damped sinusoids to model transient audio behavior better than symmetric Gabor atoms [19]. In light of the dark energy problem, Gribonval proposed to use the formant-wave-function, a waveform that describes the output of a source-filter synthesizer in the time domain, as an atom that sparsely represents audio and typically does not create pre-echo (dark energy) like a symmetric atom does [20]. The gammatone function is a popular model of the cochlear filter, and has thus been the prototype atom of choice for sparse representations that reflect human auditory perception [21] [22] [23]. Recent research has shown that asymmetric atoms can also represent other types of signals. For text-to-speech applications, [24] decomposed the fundamental frequency trajectory of a speech signal with gammatone atoms. For biomedical signal decompositions, [25] designed an asymmetric atom that involves a Gaussian for the attack and a hyperbolic tangent for the decay.

Existing asymmetric atoms have demonstrated their ability to sparsely represent audio in these preliminary studies because they correlate with natural sounds and musical audio, however, due to their origins outside of sparse audio representation realm they do not easily adapt to a wide range of audio signals. The cause of this is either because their mathematical properties make parameter estimation difficult or computationally inefficient, or because their mathematical construction simply does not allow for much adaptation. At the same time, the gammatone and formant-wave-function are almost never discussed within the same context because of their different origins. We argue that a comparative discussion would be beneficial, not only to consolidate knowledge about their performance

as a step towards the design of a better asymmetric atom, but also to expose them to different audio research fields.

In this thesis, our goal is to explore the ability of asymmetric atoms to sparsely represent audio. We define the scope of a sparse representation to not only include sparse approximations of existing sounds (parsimonious analysis) but also the synthesis of new sounds from the smallest possible number of atoms (a sound model). We will determine what factors of the existing asymmetric atoms make them suitable for sparsely representing audio, and if some combination of those parameters in a new atom will outperform the existing ones.

We will develop new ways to locate long duration atoms by searching for horizontal time-frequency components in the signal and employing robust estimation methods and heuristics to transform short-term spectral information into asymmetric atoms. We will create a hybrid algorithm that bridges two separate sub-systems, one that locates long duration atoms and another that searches for short duration atoms, to increase the speed and improve the representation sparsity of sparse approximation algorithms. From the results of the research, we aim to create sparser representations of audio that are useful for applications like audio analysis, audio coding, and music compositional tools.

1.1 Contributions

The three main contributions of this thesis are:

1. A theoretical and practical comparison of existing asymmetric atoms, with the introduction of a new atom that satisfies all the comparison criteria (Chapter 3[1]).

2. A new algorithm that improves the scalability of the greedy sparse approximation algorithm Matching Pursuit by linking two separate search techniques, one that quickly finds long-duration atoms from partial trajectory data and the other that searches for short-duration atoms from the cross-correlations of the signal and a small dictionary (Chapter 4).

[1]Some of Chapter 3 was published in the *Proceedings of the 20th International Conference on Digital Audio Effects (DAFx-17), Edinburgh, UK, September 5-9, 2017.* I was the first author and my advisor was the corresponding author.

3. Results from experiments that test how well the new atoms approximate audio, the performance of the new algorithm, and Newton's method's ability to refine ramped exponentially damped sinusoid (REDS) parameters (Chapter 5).

Other contributions include:

- The adaptation of the Reassignment Method into a recursive structure for refining atom parameters within a sparse approximation algorithm (Section 2.4.2).
- A discussion about the link between audio synthesis models and sparse representations (Section 2.5).
- A recursive inner product algorithm that estimates the onset time and duration of a complex or real-valued damped sinusoid atom (Algorithm 3).
- A derivation of the first and second derivatives of the sparse approximation objective function for Newton's method estimation of complex-valued atom parameters (Appendix A).

1.2 Structure of thesis

Chapter 2 details the theoretical foundations of a sparse approximation and the motivation for achieving a sparse representation in the context of analyzing and synthesizing audio. It provides background information on the topic with links to classic and contemporary research. Chapter 3 details symmetric atoms before delving into a thorough comparative study of existing asymmetric atoms. Then, it introduces a new asymmetric atom, REDS, that satisfies each of the properties we desire, and how it can sparsely represents audio. Chapter 4 reveals a new algorithm that efficiently locates asymmetric atoms with long time durations and adapts REDS parameters to construct a sparse audio approximation. Chapter 5 reports on a series of experiments that test the performance of the new algorithm. Finally, Chapter 6 summarizes the work that the thesis presented and directs the reader into a path of future research.

Chapter 2

Sparsity

Additive sound synthesis generally refers to the process of synthesizing sound by a combination of other sounds, a definition that relates to sound production on different scales, for example: music composition involves combining the sounds of musical instruments to produce a piece that is inherently more complex than the individual parts; a polyphonic instrument creates harmony from the superposition of multiple fundamental frequencies; the combination of a fundamental frequency's harmonics contributes to the sounds timbre [26] [27].

Formally, an additive sound model represents an audio signal as a linear combination of elements,

$$\mathbf{y} = \mathbf{\Phi x} \tag{2.1}$$

where $\mathbf{y} \in \mathbb{R}^{\mathrm{N}}$ is an audio signal, $\mathbf{\Phi} \in \mathbb{R}^{\mathrm{N}\times\mathrm{M}}$ is a matrix (dictionary) whose column vectors $\boldsymbol{\phi}_m$ are elements (atoms), and $\mathbf{x} \in \mathbb{R}^{\mathrm{M}}$ contains the weights of each atom in the dictionary [1] [28]. $\mathbf{x}$ is a sparse representation of $\mathbf{y}$ if many of its values are zero, which means that we can represent $\mathbf{y}$ using only a few columns (atoms) of $\mathbf{\Phi}$. An upcoming section concretely defines the sparse approximation problem.

Additive sound modeling's root is the Fourier series: a theoretical Fourier series synthesizer can create an arbitrary sound by adding together a (possibly infinite) number of infinite duration sinusoids. In reality, a sound's duration is finite and thus has a temporal location. This fact motivated Gabor's investigation of a time-dependent additive signal model that led to his extension of Fourier analysis into the time-frequency plane in 1946 [14] with the Gabor transform. The Gabor transform decomposes a signal onto sinusoids whose

amplitude envelopes are Gaussian functions. The short-time Fourier transform (STFT) is a generalization of the Gabor transform; it models a signal by a linear combination of time-shifted harmonically-related complex sinusoidal atoms that have some amplitude modulation. In the 1980s, research into alternative time-frequency representations to the STFT led to the development of the Wavelet transform [13]. Wavelets window sinusoids equally over time but decrease the scale of the window as frequency increases. The momentum from multi-resolution analysis after the introduction of the Wavelet transform carried into the general idea of atomic modeling, wherein a dictionary may contain waveforms of arbitrary size and construction to sparsely represent audio.

In this chapter we detail the mathematical foundations and practical aspects of sparse audio approximations, then link classical synthesis techniques to sparse representations.

2.1 Sparse approximation

Residual energy quantifies a signal approximation's quality, $\|\mathbf{r}\|_2^2 = \|\mathbf{y} - \mathbf{\Phi x}\|_2^2$. The goal is to achieve a quality of signal approximation such that

$$\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq \epsilon \tag{2.2}$$

where $\epsilon \ll \|\mathbf{y}\|_2^2$ is the residual energy bound.

Approximating an arbitrary signal according to the additive sound model is a matter of solving a linear inverse problem. A system of linear equations is either fully-determined, over-determined, or under-determined [29]. When a system is fully-determined the number of unknowns equals the number of equations (N=M) and its unique solution is $\mathbf{x} = \mathbf{\Phi}^{-1}\mathbf{y}$, when $\mathbf{\Phi}$ is invertible. The discrete Fourier transform (DFT) is an example of a fully-determined linear inverse problem. An over-determined system means the number of equations is greater than the number of unknowns, N>M. Least squares is one method of solving an over-determined linear inverse problem [29]. Finally, when the number of equations is less than the number of unknowns, i.e., $\mathrm{N} < \mathrm{M}$, the system is under-determined. There are either no solutions (not the situation of interest) or infinitely many solutions to an under-determined problem. Given that the number of possible solutions is infinite, we must enforce an additional problem constraint to reduce the number of relevant solutions. A way to constrain the problem to retrieve powerful results is through the assumption that

$\mathbf{x}$ is sparse, which assumes that $\mathbf{y}$ is sparse in its own domain or some transform domain.

Most audio signals are typically sparse in some domain, or at least compressible [8], which implies that the sorted coefficients in $\mathbf{x}$ decay rapidly [30]. For example, a pure tone is non-sparse in time because it requires N points to define its amplitude, however, the frequency domain sparsely represents the pure tone with a single coefficient located at the tone's frequency. We seek a sparse representation, rather than one from the solution of a fully-determined system (e.g., a transform over an orthogonal basis) because it is typically more interpretable, controllable, and efficiently transmittable. The task is to choose a solution constraint such that $\mathbf{x}$ is sparse and still results in a residual energy below some value ϵ (2.2). In other words, the optimization problem's solution constraint must promote sparsity. The next section will explore mathematical constraints that encourage a sparse solution.

2.1.1 Sparsity-promoting norms

Constraining the solution to an ℓ_p norm minimum leads it along some direction that can be sparse depending on the norm p. This section discusses the effect of different minimum norm solution constraints, more precisely, whether or not they promote sparsity, at what cost they have on the problem's mathematical properties, and whether they provide access to a direct solution to a linear inverse problem.

The ℓ_p-norm of $\mathbf{x}$ is

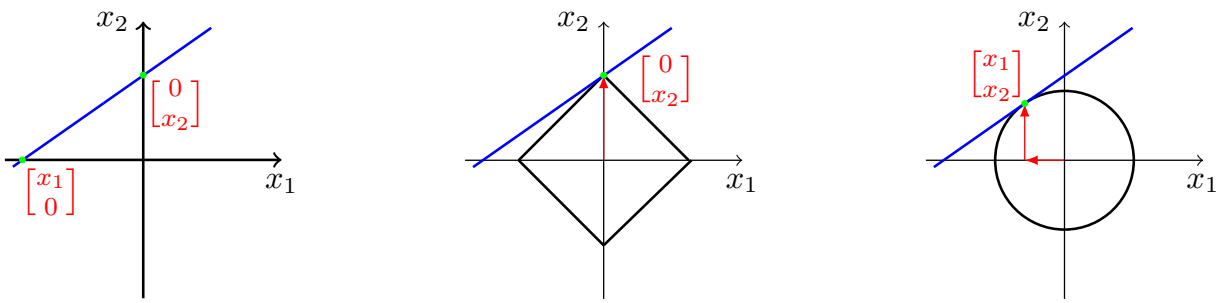
$$\|\mathbf{x}\|_p = (|x_1|^p + |x_2|^p + \ldots + |x_n|^p)^{\frac{1}{p}} \tag{2.3}$$

An ℓ_p-norm's two dimensional shape, also referred to as an ℓ_p-ball, is the solution to the equation

$$\|\mathbf{x}\|_p^p = |x_1|^p + |x_2|^p = c \tag{2.4}$$

where $c \in \mathbb{R}_{\geq 0}$ is a constant (the unit norm ball corresponds to $c = 1$). Figure 2.1 shows two dimensional ℓ_p-norm solution scenarios for each p under consideration. The point of intersection, whose coordinates are x_1 and x_2, is the $\min \|\mathbf{x}\|_p$ solution to the linear inverse problem, $\mathbf{y} = \mathbf{\Phi x}$. Visually, the ℓ_p ball increases from the origin until it intersects with the blue line. That point of intersection is the solution.

First, we discuss the minimum ℓ_2 norm case, $\min \|\mathbf{x}\|_2$, whose ℓ_p-ball is $x_1^2 + x_2^2 = c$,



(a) ℓ_0 (b) ℓ_1 (c) ℓ_2

Figure 2.1 ℓ_p-norm solutions. The blue line is $\mathbf{y} = \mathbf{\Phi x}$.

the equation of a circle, see Figure 2.1c. Since both x_1 and x_2 are non-zero, the solution is non-sparse. Notice how x_1 and x_2 are both smaller in magnitude than the solution values from the other two figures. To summarize, minimizing the solution's ℓ_2 norm provides a unique, small, and non-sparse solution. $\tilde{\mathbf{x}} = \mathbf{\Phi}^{\mathrm{H}}(\mathbf{\Phi}\mathbf{\Phi}^{\mathrm{H}})^{-1}\mathbf{y}$ is a minimum ℓ_2 norm solution.

Next, let us consider minimizing the solution's ℓ_1 norm, whose ℓ_p-ball is $|x_1| + |x_2| = c$, the equation of a square diamond, see Figure 2.1b. While the ℓ_1 norm does not model sparsity directly, it leads to a sparse and unique solution as a constraint for the linear inverse problem. The solution is larger in value and sparser in terms of non-zero values than the ℓ_2 norm case. Basis Pursuit [2] (LASSO) solves the ℓ_1 norm problem.

Finally, we consider minimizing the solution's ℓ_0 "norm", whose ℓ_p-ball is $|x_1|^0 + |x_2|^0 = c$, the unit axis, see Figure 2.1a. Quotation marks around "norm" reflect how ℓ_0 is not technically a norm because $\|l\mathbf{x}\|_0 \neq l\|\mathbf{x}\|_0$ for $l \in \mathbb{R}$. The ℓ_0 "norm" models sparsity directly because it counts the number of elements in $\mathbf{x}$. Minimizing the solution's ℓ_0 "norm" leads to a sparse solution. Since there are two points of intersection, the solution is not unique. This system is non-convex and its direct solution is NP-hard, however, greedy algorithms that approximate a solution exist [3].

2.1.2 Problem statement

As ℓ_0 is a direct measure of sparsity, the sparse approximation problem's canonical form is

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \|\mathbf{x}\|_0 \\ \text{subject to} \quad & \|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq \epsilon \end{aligned} \tag{2.5}$$

$\boldsymbol{\ell}_0$	$\boldsymbol{\ell}_1$	$\boldsymbol{\ell}_2$
Models sparsity directly	Models sparsity indirectly	Non-sparse
Non-convex	Convex	Convex
Non-unique solution	Unique solution	Unique solution
Non-smooth	Non-smooth	Smooth
Greedy algorithms approximate the solution	Solution available via convex optimization	Direct solution available (e.g., least squares)

Table 2.1 ℓ_p norm comparison in the context of promoting a sparse solution to a linear inverse problem.

where, $\|\mathbf{x}\|_0$ is the ℓ_0 "norm" of $\mathbf{x}$ which counts the number of non-zero elements in $\mathbf{x}$ [2]. Although there is no algorithm that can directly solve the sparse approximation problem, greedy algorithms build an approximate solution vector one entry at a time.

2.2 Matching Pursuit (MP)

Matching Pursuit (MP) is an iterative algorithm that approximately solves (2.5) [1]. At each iteration, matching pursuit chooses an atom $\boldsymbol{\phi}_m$ and a coefficient x that minimizes the signal residual energy, formally,

$$\arg\min_{x,m} \|\mathbf{y} - \boldsymbol{\phi}_m x\|_2^2 \tag{2.6}$$

Let $J(x) = \|\mathbf{y} - \boldsymbol{\phi}_m x\|_2^2$. The value of x that minimizes $J(x)$ for any $\boldsymbol{\phi}_m$ is the one that satisfies $\nabla_x J(x) = 0$. The expanded form of $J(x)$ is,

$$\begin{aligned} \|\mathbf{y} - \boldsymbol{\phi}_m x\|_2^2 &= (\mathbf{y} - \boldsymbol{\phi}_m x)^\intercal (\mathbf{y} - \boldsymbol{\phi}_m x) \\ &= \mathbf{y}^\intercal \mathbf{y} - \mathbf{y}^\intercal \boldsymbol{\phi}_m x - x \boldsymbol{\phi}_m^\intercal \mathbf{y} + x \boldsymbol{\phi}_m^\intercal \boldsymbol{\phi}_m x \\ &= \|\mathbf{y}\|_2^2 - 2x \boldsymbol{\phi}_m^\intercal \mathbf{y} + x^2 \|\boldsymbol{\phi}_m\|_2^2 \end{aligned} \tag{2.7}$$

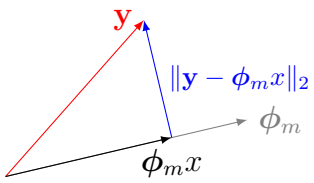


Figure 2.2 Projection of $\mathbf{y}$ onto $\boldsymbol{\phi}_m$.

The partial derivative of $J(x)$ with respect to x, $\nabla_x J(x)$, and the value of x when $\nabla_x J(x) = 0$ is as follows,

$$\begin{aligned} 0 = \nabla_x \|\mathbf{y} - \boldsymbol{\phi}_m x\|_2^2 &= \frac{\partial}{\partial x}\left(\|\mathbf{y}\|_2^2 - 2x\boldsymbol{\phi}_m^\intercal \mathbf{y} + x^2 \|\boldsymbol{\phi}_m\|_2^2\right) \\ &= -2\boldsymbol{\phi}_m^\intercal \mathbf{y} + 2x\|\boldsymbol{\phi}_m\|_2^2 \\ \rightarrow \qquad x &= \frac{\boldsymbol{\phi}_m^\intercal \mathbf{y}}{\|\boldsymbol{\phi}_m\|_2^2} \end{aligned} \tag{2.8}$$

$\boldsymbol{\phi}_m$ is normalized so that $\|\boldsymbol{\phi}_m\|_2^2 = 1$.

More intuitively, the distance $\|\mathbf{y} - \boldsymbol{\phi}_m x\|_2$ is smallest when it is orthogonal to $\boldsymbol{\phi}_m$, see Figure 2.2. $(\mathbf{y} - \boldsymbol{\phi}_m x) \perp \boldsymbol{\phi}_m$ so

$$\begin{aligned} 0 &= (\mathbf{y} - \boldsymbol{\phi}_m x)^\intercal \boldsymbol{\phi}_m \\ \rightarrow \qquad x &= \frac{\boldsymbol{\phi}_m^\intercal \mathbf{y}}{\|\boldsymbol{\phi}_m\|_2^2} \end{aligned} \tag{2.9}$$

The problem simplifies after substituting (2.8) into $J(x)$,

$$\begin{aligned} J(\boldsymbol{\phi}_m^\intercal \mathbf{y}) &= \|\mathbf{y}\|_2^2 - 2\mathbf{y}^\intercal \boldsymbol{\phi}_m \boldsymbol{\phi}_m^\intercal \mathbf{y} + (\boldsymbol{\phi}_m^\intercal \mathbf{y})^2 \|\boldsymbol{\phi}_m\|_2^2 \\ &= \|\mathbf{y}\|_2^2 - 2(\boldsymbol{\phi}_m^\intercal \mathbf{y})^2 + (\boldsymbol{\phi}_m^\intercal \mathbf{y})^2 \\ &= \|\mathbf{y}\|_2^2 - (\boldsymbol{\phi}_m^\intercal \mathbf{y})^2 \end{aligned}$$

Thus, the minimization problem (2.6) reduces to a search over column index m,

$$\arg\min_m \left(\|\mathbf{y}\|_2^2 - (\boldsymbol{\phi}_m^\intercal \mathbf{y})^2\right) \tag{2.10}$$

which is equivalent to maximizing $|\boldsymbol{\phi}_m^{\intercal}\mathbf{y}|$,

$$\arg\max_m |\boldsymbol{\phi}_m^{\intercal}\mathbf{y}| \tag{2.11}$$

Accordingly, matching pursuit chooses the atom that forms the largest inner product with the k^{th} iteration signal residual $\mathbf{r}^{(k)}$, $\boldsymbol{\phi}_{\tilde{m}}$, where $\tilde{m} = \arg\max_m |\boldsymbol{\phi}_m^{\intercal}\mathbf{r}^{(k)}|$, then subtracts it from $\mathbf{r}^{(k)}$ to get $\mathbf{r}^{(k+1)}$,

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - x^{(k)}\boldsymbol{\phi}_{\tilde{m}} \tag{2.12}$$

where $x^{(k)} = \boldsymbol{\phi}_{\tilde{m}}^{\intercal}\mathbf{r}^{(k)}$. For dictionaries of real-valued atoms, MP must estimate phase from a discrete set as it does for the other parameters. Alternatively, when dictionary atoms are complex-valued, the coefficient x contains not only the atom's magnitude information but also its phase, and MP does not need to conduct an explicit search over a set of phases [19]. In this thesis we use dictionaries of complex-valued atoms. For complex-valued atoms $\boldsymbol{\Phi} \in \mathbb{C}^{\mathrm{N}\times\mathrm{M}}$ the update equation is

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - 2\Re\left\{x^{(k)}\boldsymbol{\phi}_{\tilde{m}}\right\} \tag{2.13}$$

where $x^{(k)} = \boldsymbol{\phi}_{\tilde{m}}^{\mathrm{H}}\mathbf{r}^{(k)} \in \mathbb{C}$ and $\tilde{m} = \arg\max_m |\boldsymbol{\phi}_m^{\mathrm{H}}\mathbf{r}^{(k)}|$ (see Algorithm 1) [31].

MP's stopping condition is based on the residual energy and/or iteration number. Stopping MP after k iterations guarantees a solution sparsity of at most k, $\|\mathbf{x}\|_0 \leq k$. The residual energy after k iterations is dependent on $\mathbf{y}$ and $\boldsymbol{\Phi}$. A residual energy stopping condition is applicable when the goal is to achieve some level of representation quality. For this case, MP's iteration count stopping condition acts as a fail-safe; if MP's rate of convergence to the desired residual energy is impractically slow, it will still stop after some

Algorithm 1 Matching Pursuit

init: $k = 0$, $\mathbf{x}^{(k)} = \mathbf{0}$, $\mathbf{r}^{(k)} = \mathbf{y}$
repeat
  $\tilde{m} = \arg\max_m |\boldsymbol{\phi}_m^{\mathrm{H}}\mathbf{r}^{(k)}|$
  $x^{(k)} = \boldsymbol{\phi}_{\tilde{m}}^{\mathrm{H}}\mathbf{r}^{(k)}$
  $\mathrm{x}_{\tilde{m}}^{(k+1)} = \mathrm{x}_{\tilde{m}}^{(k)} + x^{(k)}$
  $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - 2\Re\{x^{(k)}\boldsymbol{\phi}_{\tilde{m}}\}$
  $k = k + 1$
until stopping condition

iteration count k.

Matching pursuit searches for a locally optimal solution. Since it does not consider how atoms from other iterations interact, its solution may be globally sub-optimal. Orthogonal Matching Pursuit (OMP), an MP extension, manages a global view of the problem by projecting the entire solution onto the residual after each iteration [32].

2.3 On parametric dictionaries

A solution to (2.5) is $\mathbf{\Phi} = \mathbf{y}$, so $\mathbf{x} = 1$ and $\|\mathbf{x}\|_0 = 1$. Even though in this case $\mathbf{x}$ is perfectly sparse its representation of $\mathbf{y}$ is meaningless. Given that $\mathbf{x}$'s representation of $\mathbf{y}$ must be *meaningful* (we elaborate on this in the next paragraph), solving (2.5) is a matter of building $\mathbf{\Phi}$ to minimize $\|\mathbf{x}\|_0$ with elements of interpretable, descriptive structure.

Decomposing onto parametrized atoms provides access to the signal's structural information via the parameters of the representative atoms. Parametric atoms have structure and are necessarily elementary. Since audio signals are very high-dimensional (they have complex features like highly non-stationary time and frequency behavior, etc.) and data sets and atoms are low-dimensional (elementary) functions, for a solution to be sparse (to involve only a few atoms), a dictionary must make available an enormous variety of atoms to choose from. This concept is comparable to spoken language's highly redundant vocabularies that enable concise sentence structuring. The problem is that the computational complexity grows, and thus the speed of approximation algorithm slows, proportionally (or worse) to the dimensions of $\mathbf{\Phi}$ [11]. A tractable MP algorithm demands a dictionary whose parametric structuring allow fast algorithms, for example, the fast Fourier transform (FFT) or Mallat's pyramidal discrete Wavelet transform (DWT) [33], to calculate the inner products, and efficient searches for the most correlated atom [15].

Several MP software packages, such as MPTK [10] and LastWave [34], employ fast techniques to decompose 1-D (e.g., audio) signals. Even so, MPTK, which is the fastest implementation of MP currently available, still has an execution time of more than one hundred times the signal duration when either the dictionary contains atoms with large time support or when the number of iterations is high [11]. Given these reasons for an impractically slow run of MP, the opposite points highlight how MP can run quickly to extract a high-quality approximation:

1. Minimize the number of iterations.

2. Minimize the time to run each iteration by:

   i Minimizing the search time for the best correlated atom.

   ii Accelerating and/or minimizing the number of inner products updates.

Interestingly, the sparse approximation optimization problem's objective, minimize $\|\mathbf{x}\|_0$, is also a way to improve the speed of MP. However, the ways to accelerate MP depend on one another and makes the design of a fast MP algorithm a real challenge. For example, increasing the number of atoms in the dictionary tends to decrease the number of iterations. This involves finely sampling the atom's parameter set, which includes duration N, so, by extension, the dictionary will necessarily contain atoms with long durations. However, the time to search through a dictionary for the best atom increases proportionally, or worse, with the number of dictionary columns M and the complexity of the inner product computation grows in proportion, or worse, to atom duration N. The dictionary size's influence on search time and computation complexity is dependent on the search method and inner product computation method (e.g., direct calculation or through an FFT), respectively. MPTK uses an STFT to calculate correlations for most types of atoms and employs an efficient tree search [10]. Conversely, a smaller dictionary enables fewer and faster inner product updates and quick searches for the best correlated atom, however the resulting solution is less sparse and the added iteration count to reach an equivalent approximation quality may result in a slower overall execution time.

Given the aforementioned trade-off, we choose to search for the sparsest approximation of a signal that is also high-quality (has a relatively small ϵ).

2.3.1 Problem reformulation

We release (2.1) from its matrix notation to consider an alternative measure of sparsity.

$$\mathbf{y} = \sum_{k=1}^{K} \phi(\boldsymbol{\lambda}_k) x_k \tag{2.14}$$

where function ϕ creates atom $\boldsymbol{\phi}_k \in \mathbb{R}^{\mathrm{N}}$ from parameter set $\boldsymbol{\lambda}_k \in \mathbb{R}^{Q}$. KQ is the measure of sparsity and $\mathrm{N}(KQ)^{-1}$ is the approximation compression ratio. Like the ℓ_0 "norm", KQ is a direct measure of sparsity. More specifically, KQ is a direct measure for sparse audio coding.

Rather than transmitting the signal $\mathbf{y}$ itself, audio coding involves transmitting instructions for a converter to re-synthesize $\mathbf{y}$. When the sparsity measure is $\|\mathbf{x}\|_0$ and the synthesis model is (2.1), it entails that $\mathbf{x}$ is the instruction set, and so the converter contains a static matrix $\mathbf{\Phi}$. Since we are interested in adaptive dictionaries, we prefer to measure sparsity in terms of 2.14: the instructions are $\boldsymbol{\lambda}$ for the converter $\phi(\boldsymbol{\lambda})$. Thus, the transmitter sends KQ samples of data rather than N. Clearly, the goal is to minimize KQ, more specifically, to design atoms with a minimal number of parameters Q that can also sparsely represent audio and develop ways to approximate $\mathbf{y}$ with a minimal number of those atoms K. The following chapters address this goal.

Next, we look at how to virtually expand dictionary size without increasing the number of inner product computations by searching among an atom's pseudo-continuous parameter space and adapting it to the residual.

2.4 Dictionary adaptation

A facet of dictionary based methods research is directed towards ways to design dictionaries such that they will sparsely represent a signal. There are several ways to create a dictionary: by concatenating bases (e.g., MDCT bases, diracs, wavelets, etc.) [11], learning atoms via some machine learning technique, sampling a parametric atom (e.g., translating and modulating it in time and frequency), and by a mixed method involving two or more of the previous approaches. Dictionary learning involves adapting a dictionary to a signal via some machine learning algorithm [6]. Adaptation may be through the parameters of the atom to design a dictionary around a signal [35], or by a probabilistic method wherein the dictionary is not parametric. Dictionary learning is not aimed at quickly finding a solution as it typically does not scale well, rather, one of its main utilities is in creating a dictionary that fits well with a certain signal type, for example natural sounds and speech [21], then recycling the dictionary to approximate other signals within the same class with a sparse approximation algorithm, like MP or Basis Pursuit.

Another approach of dictionary adaptation is through the refinement of atom parameters within the greedy MP framework [1]. In general, a dictionary's discrete parameter set is coarse so its size does not decrease the algorithm's speed. The best fitting atom of that dictionary points to a parameter value subset. The idea is to search within that subset after locating it to optimize one or more parameters. Next, we explore ways to refine atom

parameters within a greedy algorithm, starting with the flexible and powerful Newton's method that the original paper on MP applied [1], then we contribute a new method that is more robust and efficient than Newton's method at estimating certain parameters like frequency and damping factor.

2.4.1 Newton's method (NM)

In a greedy iterative framework of solving the sparse approximation problem, we seek an atom that minimizes the residual energy. More precisely, let λ be some parameter of atom $\boldsymbol{\phi}_\lambda$, then we seek a value for that parameter that minimizes the residual energy function, $J(\lambda) = \|\mathbf{y} - \boldsymbol{\phi}_\lambda x\|_2^2$. Newton's method (NM) iteratively searches for the minimum of $J(\lambda)$ (i.e., the solution to $\frac{\partial J(\lambda)}{\partial \lambda} = 0$) using the first and second partial derivative of $J(\lambda)$.

Although Newton's method is often mentioned in MP literature as an optional step to refine parameters [1], [15], [20], to the best of our knowledge, the actual equations that enable such an implementation for an arbitrary atom parameter λ are not present in the literature. Therefore, we derive the general-form equations for a complex-valued atom so that one may more readily implement and test atomic Newton method parameter refinement. Note that these equations will also work for a real-valued atom. The first derivative of the residual energy function $J(\lambda)$ is

$$\begin{aligned}\frac{\partial}{\partial \lambda} J(\lambda) = \frac{\partial}{\partial \lambda}\|\mathbf{y} - \boldsymbol{\phi}_\lambda x\|_2^2 &= \frac{\partial}{\partial \lambda}\Big(\mathbf{y}^{\mathrm{H}}\mathbf{y} - 2\Re\{\bar{x}\boldsymbol{\phi}_m^{\mathrm{H}}\mathbf{y}\} + \bar{x}\boldsymbol{\phi}_\lambda^{\mathrm{H}}\boldsymbol{\phi}_\lambda x\Big) \\ &= -2\Re\left\{\bar{x}\frac{\partial \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda}\mathbf{y}\right\} + 2|x|^2\frac{\partial \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda}\boldsymbol{\phi}_\lambda \end{aligned} \tag{2.15}$$

and the second derivative is

$$\begin{aligned}\frac{\partial^2}{\partial \lambda^2} J(\lambda) &= \frac{\partial}{\partial \lambda}\Big(\frac{\partial}{\partial \lambda} J(\lambda)\Big) \\ &= -2\Re\left\{\bar{x}\frac{\partial^2 \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda^2}\mathbf{y}\right\} + 2|x|^2\left(\frac{\partial^2 \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda^2}\boldsymbol{\phi}_\lambda + \frac{\partial \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda}\frac{\partial \boldsymbol{\phi}_\lambda}{\partial \lambda}\right)\end{aligned} \tag{2.16}$$

Newton's method estimate of λ at iteration (k) is

$$\lambda^{(k)} = \lambda^{(k-1)} - \frac{\frac{\partial}{\partial \lambda} J(\lambda^{(k-1)})}{\frac{\partial^2}{\partial \lambda^2} J(\lambda^{(k-1)})} \tag{2.17}$$

One advantage of Newton's method for parametric refinement is its flexibility: in theory, this method can refine any parameter of $\boldsymbol{\phi}$, so long as the function is at least twice-differentiable with respect to that parameter. Even more, we can estimate multiple parameters simultaneously at each iteration with the extension of Newton's method into multiple dimensions. Let $\boldsymbol{\lambda} \in \mathbb{R}^Q$ be a vector that contains multiple parameters of atom $\boldsymbol{\phi}_{\boldsymbol{\lambda}}$, then

$$\boldsymbol{\lambda}^{(k)} = \boldsymbol{\lambda}^{(k-1)} - \Big(\mathbf{H}_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda}^{(k-1)})\Big)^{-1} \nabla_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda}^{(k-1)}). \tag{2.18}$$

where $\nabla_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda})$ is the gradient of $J(\boldsymbol{\lambda})$ and $\mathbf{H}_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda})$ is the Hessian matrix of $J(\boldsymbol{\lambda})$,

$$\big(\mathbf{H}_{\boldsymbol{\lambda}} J(\boldsymbol{\lambda})\big)_{i,j} = \frac{\partial^2 J(\boldsymbol{\lambda})}{\partial \lambda_i \partial \lambda_j} = -2\Re\left\{ \bar{x} \frac{\partial^2 \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda_i \partial \lambda_j} \mathbf{y} \right\} + 2|x|^2 \left(\frac{\partial^2 \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda_i \partial \lambda_j} \boldsymbol{\phi}_\lambda + \frac{\partial \boldsymbol{\phi}_\lambda^{\mathrm{H}}}{\partial \lambda_i} \frac{\partial \boldsymbol{\phi}_\lambda}{\partial \lambda_j} \right) \tag{2.19}$$

The Hessian matrix is square symmetric so the number of unique entries in the matrix is $\frac{1}{2}Q(Q+1)$. A Newton step in multiple dimensions uses $\frac{3}{2}Q(Q+1)$ inner products.

Newton's method is a good choice for refining parameters that are not accessible via a classical sinusoidal parameter estimation method, like the Reassignment Method [36] or ESPRIT [37]. Next, we promote such sinusoidal model estimation techniques for greedy sparse approximation problems.

2.4.2 Recursive Reassignment Method (RRM)

We contribute here a new atom parameter estimation method that requires less computations and tends to converge faster than Newton's method. The recursive reassignment method (RRM) is the reassignment method fit into a recursive structure that iteratively reduces $\|\mathbf{r}\|_2^2$. It involves the atom's envelope $\boldsymbol{w} = |\boldsymbol{\phi}|$ and envelope derivative $\boldsymbol{w}'$, see Algorithm 2, where $\odot$ denotes element-wise multiplication. RRM takes in the initial parameter estimate, in this case the frequency, and refines the estimate until the algorithm reaches some maximum iteration count, or $g^{(k+1)} \not> g^{(k)}$.

A refinement step requires only two inner products, one for the atom derivative and another to update the atom's gain. Moreover, the reassignment method converges quickly, does not rely on a convex objective function, and performs well even when the initial estimate is far from ground truth (this is not the case for the Newton or gradient descent method). RRM is limited to refining an atom's frequency and, if the atom is a damped sinusoid, the damping factor. Note that this technique extends to higher-order parameter

Algorithm 2 RRM frequency refinement

init: $k = 0$, $\boldsymbol{\phi}^{(0)} = e^{i\omega_c^{(0)} n}$, $\boldsymbol{r}_w = \boldsymbol{r} \odot \boldsymbol{w}$, $\boldsymbol{r}_{w'} = \boldsymbol{r} \odot \boldsymbol{w}'$, $g^{(0)} = \langle \boldsymbol{r}_w, \boldsymbol{\phi}^{(0)} \rangle$
repeat
  $\omega_c^{(k+1)} = \omega_c^{(k)} - \Im\left\{ \frac{\langle \boldsymbol{r}_{w'}, \boldsymbol{\phi}^{(k)} \rangle}{g^{(k)}} \right\}$
  $\boldsymbol{\phi}^{(k+1)} = e^{i\omega_c^{(k+1)} n}$
  $g^{(k+1)} = \langle \boldsymbol{r}_w, \boldsymbol{\phi}^{(k+1)} \rangle$
  $k = k + 1$
until stopping condition

estimation, such as the estimation of frequency slope, at the cost of more inner products per iteration.

2.5 Sparsity in synthesis

In this section, we establish how classic audio synthesis techniques link to the more modern idea of a sparse representation. Literature has pointed out that granular analysis-synthesis of audio is an atomic decomposition's synthesis counterpart [31]. We discuss two other audio synthesis techniques and establish how they relate to the sparse synthesis model, more precisely, if the techniques create complex sounds with a sparse number of elementary waveforms (atoms).

2.5.1 Additive synthesis

In additive synthesis from non-stationary sinusoidal modeling, a linear combination of amplitude and frequency modulated sinusoids represent the signal. Rather than sum an entire Fourier basis, as one does with an inverse DFT, the synthesizer chooses prominent sinusoids (peaks) out of the signal's, possibly short-time, frequency spectrum. Often, some estimation technique refines the frequencies of the sinusoids. Due to this reductive selection and refinement, sinusoidal modeling often reconstructs a signal to some degree of quality with less sinusoids than the DFT length. While the DFT is the solution of a fully-determined linear inverse problem that uses a complete dictionary of complex sinusoids, estimating the frequency of a sinusoid responsible for a peak in the DFT spectrum effectively extends the dictionary into a pseudo-continuous frequency parameter space. Extending the set of sinusoids of the DFT makes the dictionary over-complete. The over-complete dictionary

makes a sparse representation realizable.

Representing a signal via sinusoidal modeling involves creating a small sub-set of the STFT dictionary by extracting parameters from the peaks of short-time spectra, then interpolating between those points to create a set of instantaneous parameters, phase $\theta_l[n]$ and amplitude $a_l[n]$:

$$y[n] = \sum_{l}^{L} a_l[n] \cos(\theta_l[n]) \tag{2.20}$$

Although the peaks of each frame are typically sparse, the additive synthesis framework does not guarantee a certain level of reconstruction quality. Regardless, the additive sinusoidal model does not sparsely represent transients because a transient will typically have a dense frequency spectrum. One way to improve the situation is to incorporate into the additive synthesis representation of steady-state content a separate atomic decomposition with damped sinusoids to represent transients [38].

2.5.2 Source-filter

Creating a sustained sound (e.g., a voice) from the source-filter model involves filtering a sparse excitation signal made of a (possibly) periodic sequence of short duration signals. Likewise, synthesizing a percussive sound (e.g., a piano) involves summing the output of several resonant filters with comparably long decay times from a single excitation. In the time-domain method, this means that the model synthesizes a signal $\mathbf{y}$ as a linear combination of time-shifted resonant filter impulse responses (i.e., time-frequency atoms). Formally, we express this as $\mathbf{y} = y[n] = \sum h_\lambda[n-\tau]x_{\lambda,\tau} = \mathbf{\Phi x}$, where $\mathbf{\Phi}$ is a dictionary of atoms $h_{\lambda,\tau}[n] = h_\lambda[n-\tau]$ that are indexed by λ and time shift $\tau \in \mathbb{R}$, and $\mathbf{x}$ contains their amplitude coefficients $x_{\lambda,\tau}$.

In practice, shifting $h[n]$ by τ entails either convolving it with a bandlimited impulse excitation[1] or by placing $h[n]$ at some round integer value of tau, $[\tau]$, then phase-shifting the oscillation of $h[n]$ to fit it to the correct time location. The second method illustrates the connection between source-filter synthesis and the atomic model and furthermore the sparse audio approximation procedure, because, recall that when sparsely approximating

[1]Sinc interpolation is the theoretically ideal method of bandlimiting an impulse function $h[n-\tau]$ for non-integer values of τ. In practice, since the sinc function is non-causal and has an infinite duration, it is common to truncate the sinc function with a finite duration window in exchange for some of its ideal properties.

an existing signal, MP places an atom in some discrete time location and phase shifts it through the complex coefficient x or, if the atom is real, solves for phase via some estimation method. The data that instructs a source-filter synthesizer is a sparse representation of the resulting audio signal. Moreover, the source-filter model matches the framework of atomic modeling and sparse approximations.

2.6 Summary

In this chapter, we described the motivation for enforcing a sparse constraint onto the solution vector of an under-determined linear inverse problem. We explained how MP builds an approximate solution to the sparse approximation problem by choosing one atom at a time, and discussed the difficulty of implementing fast MP algorithms. Then, we reformulated the sparse approximation problem to highlight the importance of parametric atoms and generalized the additive sound model definition by avoiding matrix notation. We established general equations for refining parameters via Newton's method and proposed a recursive estimation technique using sinusoidal model parameter estimators. Finally, we discussed the relation of common audio synthesis techniques to sparse audio representations.

Chapter 3

Ramped Exponentially Damped Sinusoid Atoms

The previous chapter emphasized a crucial step towards a sparser audio representation: design a dictionary with a prototype atom whose definition involves a minimal number of parameters that is capable of representing a wide range of signal content.

Knowledge of salient audio signal features can help guide the design of such an atom: sound commonly has an amplitude envelope that rises faster than it decays (i.e., it is temporally asymmetric) and has time-varying frequency content [27]. Thus, a time-frequency structured signal model that is asymmetric in time is appropriate, for example, a damped sinusoid. However, the damped sinusoid model [19] does not have an amplitude envelope that rises smoothly from zero to a maximum while real signals almost always do. A compromise involves building a heterogeneous dictionary that includes symmetric atoms (e.g., Gabor atoms) and damped sinusoid atoms, although heterogeneous dictionaries typically require more data than homogeneous ones because each prototype atom within the dictionary has a unique parameter set. However, more importantly, decomposing asymmetric signal content with a finite number of symmetric atoms will either lead to a non-sparse solution or pre-echo (*dark energy*) [7].

We prefer to design a homogeneous dictionary (i.e., one that contains a single prototype atom), where the prototype atom is an exponentially damped sinusoid with an attack envelope. Currently, only two functions common in the literature have assumed this atomic role: the formant-wave-function [39] [20] (used in audio synthesis) and the gammatone (used

in perceptual audio coding) [40] [21]. Since neither function originated from the sparse representation area, they either cannot adapt to a wide range of sounds or suffer from a mathematical definition that prohibits parametric estimation and/or fast approximation algorithms.

In this chapter, we present a theoretical and practical comparative discussion of existing prototype atoms to consolidate knowledge and highlight their relative strengths and limitations. Our points of comparison reflect the qualities that we seek in a model: ability to match diverse signal behavior (especially transients), and "good" mathematical properties. Some of the desired mathematical properties include having a concentrated spectrum and an analytic inner product formula. We start with symmetric atoms as they are the common choice then move to an in-depth comparative discussion of asymmetric atoms. Since none of the existing atoms satisfies every criteria, we introduce a new atom that does. Then we establish connections between the new and existing atoms from their mathematical definitions and end with an experiment that shows how the new atom outperforms existing ones.

3.1 Desirable properties

We generalize the form of a prototype atom as

$$\phi[n] = E[n]e^{i\omega_c n}, \tag{3.1}$$

where $E[n]$ is the amplitude envelope (window) function, $\omega_c = 2\pi f_c$ is the normalized angular frequency of oscillation ($0 \leq f_c \leq \frac{1}{2}$). and n is discrete time.

We generalize an asymmetric atom's definition through its envelope,

$$E[n] = A[n]e^{-\alpha n}u[n] \tag{3.2}$$

where $\alpha \in \mathbb{R}_{\geq 0}$ is the damping factor, $u[n]$ is the unit step function and $A[n]$ is an attack envelope that distinguishes each asymmetric atom ($A[n] \in \mathbb{R}_{\geq 0}\ \forall\ n \in \mathbb{N}$).

For organizational purposes, we divide prototype atom properties into three categories: time-frequency properties, algorithmic efficiency, and control & flexibility.

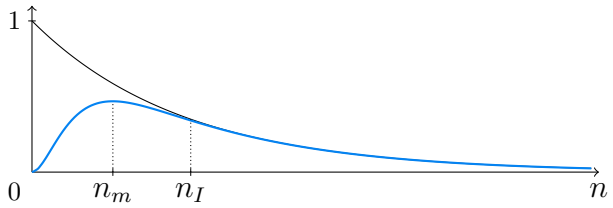


Figure 3.1 An example envelope of the form (3.2) overlaid with an exponential envelope (blue), where n_I is the influence time and n_m is the time location of the envelope maximum.

3.1.1 Time-Frequency Properties

A dictionary of atoms with varying degrees of *time and frequency concentration* is important for creating a sparse representation overall. For example, a sustained piano note begins with a short attack, which is best represented with concentrated time (spread frequency) resolution, followed by a long decay, which requires an atom with long time support and a concentrated spectrum. Multi-resolution analysis involves decomposing a signal onto a set of analyzing functions whose time-frequency tiling is non-uniform [13] [41]. We are going one step further by considering that some sounds require excellent time localization in the transient region and concentrated frequency resolution in the decay region. We aim at representing both regions with atoms whose envelopes are closer to those of natural sounds. We quantify concentration in time and frequency by the time spread, Δt, and frequency bandwidth, Δf, respectively. The Heisenberg-Gabor inequality states $\Delta t \Delta f \geq \frac{1}{2}$ (we use the definition of Δf and Δt from [14]).

Moreover, we prefer an atom that has a *unimodal spectrum*: an atom whose mathematical definition is of the form (3.2) has a spectrum $\mathcal{F}[\phi](\omega)$ that is unimodal if $|\mathcal{F}[\phi](\omega)|$ is monotonically increasing for $\omega \leq \omega_c$ and monotonically decreasing for $\omega \geq \omega_c$. A function that is truncated in time with a rectangular window admits a non-unimodal spectrum because the truncation is equivalent to convolving the spectrum with a sinc function whose oscillations introduce multiple local maxima/minima [42]. Multiple local maxima/minima in the spectrum can complicate spectral parameter estimation. We prefer an infinitely differentiable atom (i.e., of class C^∞, as defined in [43]) because its spectrum is unimodal.

3.1.2 Algorithmic Efficiency

Fast algorithms are one of the focuses of sparse representations research, as they aim to make sparse decomposition processes more tractable. Amid publications dedicated to creating faster algorithms, some reported techniques have become widely adopted [10]. Specifically, certain analytic formulas are known to increase the algorithm speed because they avoid some of the algorithm's most time consuming numerical calculations (e.g., the inner product).

An envelope shape that enables the inner product of two atoms to be expressed as an analytic formula is required for a fast matching pursuit algorithm [1]. Matching pursuit can calculate and store the dictionary's inner products once when the dictionary is static. However, when matching pursuit refines atom parameters within the iterative loop it cannot use pre-computed inner products and, therefore, it must compute them at each iteration. Numerical calculations of many inner products at every iteration prohibit speed. Analytic formulas make the process tractable.

Another way to increase the efficiency of a sparse decomposition program is to use parametric atoms, then refine atom parameters using an estimator. Finding a more adapted atom at every iteration may require less iterations overall. Developing parametric estimation techniques sometimes relies on having analytic discrete Fourier transform (DFT) formula. For example, in derivative methods, two spectra are divided to solve for one or more variables [44].

[19] explains how a recursive property of the complex damped exponential helps calculate the convolution of damped sinusoid atoms with a signal: since the impulse response of a complex one-pole filter is a damped complex exponential sinusoid, a recursive filter can efficiently calculate the correlation. We provide each atom's Z-transform to indicate its *causal filter simplicity* and therefore practicality for calculating the correlation. Besides, the Z-transform is useful for source-filter synthesis and auditory filtering.

3.1.3 Control & Flexibility

We modulate the damped sinusoid with A to enhance the atom's adaptability to natural sounds. A damped sinusoid's damping factor α indirectly controls its Δt and Δf. Smoothing the damped exponential's initial discontinuity with A concentrates its frequency localization in exchange for a more spread time localization. We want a parametrization of

A that enables precise control over its time and frequency characteristics, controllability being an essential aspect of audio synthesis. Furthermore, the attack portion of an audio signal often contains dense spectral content that allows humans to characterize its source [26].

Influence time has a major effect on the atom's overall perceived sound as it controls the degree to which the initial discontinuity is smoothed [26]. We define influence time n_I as the duration that A influences the atom: n_I is the largest value of n for which $e^{-\alpha n}(A[n] - 1) > \delta$ is true (in this chapter $\delta = .001$, see Figure 3.1). The effects of varying influence time are intuitively linked to Δt and Δf. In the frequency domain, influence time mostly controls the spectral envelope far from its center frequency (*skirt width* as defined in [39]). Increasing influence time spreads the atom's time localization and concentrates its spectrum.

An important quantity to compare between the atoms is the time $\Delta_I = n_I - n_m$, where n_m is the time location of E's maximum. n_m is often called a temporal envelope's attack time in sound synthesis [45]. We find n_m by setting E's continuous time derivative equal to zero and solving for n. For a continuous E whose $\alpha > 0$, n_m precedes n_I (i.e., A influences E even after n_m). To compare atoms along this criteria, we equalize their n_m values then compare their Δ_I values. Δ_I indicates the amount of influence that varying the skirt width will have on the bandwidth. We prefer an atom with a small Δ_I value because its 3 dB bandwidth (set through α) is not affected much by the structure of A. An envelope with a small Δ_I also reflects those produced by many acoustic instruments: an exciter increases the system's energy and then releases (at n_I), which results in a freely decaying resonance.

We do not want to complicate the definition of the atom when modulating the damped sinusoid by A either; we encourage *time-domain simplicity*. The damped sinusoid's simple definition enables us to solve for its parameters algebraically. Classic parametric estimation techniques are useful for adapting the damped sinusoid to an arbitrary signal [44]. We want to retain these desirable properties even after introducing A. An atom's time-domain simplicity will depend on how its A marries with the complex damped sinusoid. Finally, after modulating the damped sinusoid with A, we want the atom's envelope to match well with those in actual musical signals.

3.2 Symmetric Atoms

Symmetric atoms are described as such because they are symmetric about some time instant. Symmetric impulse responses are common to FIR filters, and have a linear phase response. Although symmetric atoms are not the focus of this study, we discuss Gabor atom properties to contrast those of the asymmetric atoms. The symmetric Gabor atom is a standard for sparse representation applications and, therefore, its mathematical properties that pertain to implementation efficiency are a reference point for the forthcoming asymmetric atom comparisons.

3.2.1 Gabor Atom

In 1946, Gabor proposed to modulate an infinite duration sinusoid with a Gaussian function whose parameters include time scale and translation, because it concentrates the energy of the waveform into a specific temporal location, i.e., the waveform's time center [14]. The resulting prototype waveform, the Gabor atom, remains a common choice for inclusion in dictionaries for sparse approximations due to its optimal time and frequency concentration (i.e., the Gabor atom's time-frequency localization is such that $\Delta t \Delta f = \frac{1}{2}$).

Since then, the short-time Fourier transform has become a commonplace time-frequency representation [46], for which the Gabor transform is a special case, wherein the modulating amplitude envelope is called a window function. There are many window functions that have since been designed for discrete Fourier analysis which could likewise be used as atoms for sparse representations [42], for example, the Hann window.

Later, [1] constructed a time-frequency Gabor dictionary by scaling, modulating, and translating a Gaussian window,

$$\phi[n] = E[n]e^{i\omega_c n} \tag{3.3}$$

where

$$E[n] = \exp(-\pi(\tfrac{n-\tau}{s})^2) \tag{3.4}$$

where $E[n]$ is the atom's envelope, $s \in \mathbb{R}_{\geq 1}$ is the scale that changes the time support of the atom, $\tau \in \mathbb{R}$ is the time shift, and $\omega_c = 2\pi f_c$. The Gabor atom's ℓ_2-norm is $\|\boldsymbol{\phi}\|_2 = \frac{\sqrt{s}}{2^{1/4}}$.

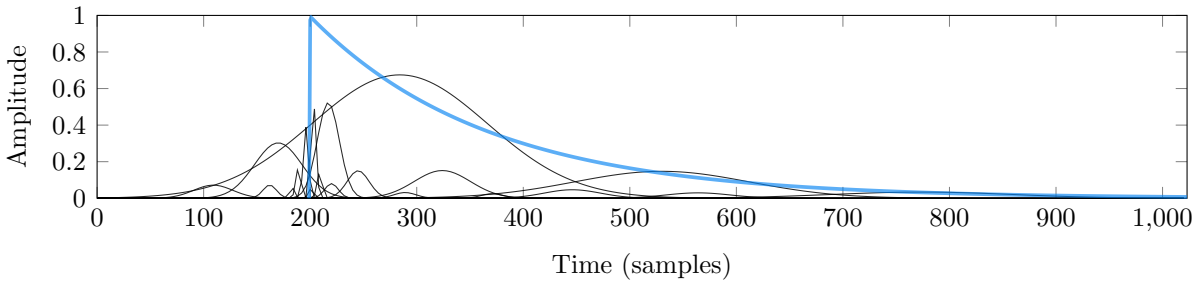


Figure 3.2 Time distribution of Gabor atoms (in black) from decomposition of a damped sinusoid (bold, blue), showing dark energy creation before the onset (i.e., pre-echo).

An analytic formula of the inner product between two Gabor atoms is available [13],

$$\langle \phi_1, \phi_2 \rangle = \frac{\sqrt{2 s_1 s_2}}{\sqrt{s_1^2 + s_2^2}} \exp\left(-\frac{i(s_1^2 \tau_2 + s_2^2 \tau_1)(\omega_{c_2} - \omega_{c_1}) + \pi(\tau_2 - \tau_1)^2}{s_1^2 + s_2^2} - \frac{(\omega_{c_2} - \omega_{c_1})^2}{4\pi(s_1^{-2} + s_2^{-2})} \right) \tag{3.5}$$

This formula's derivation involves an infinite sum in positive and negative directions, and therefore is accurate when neither atom is significantly truncated in time.

Decomposing asymmetric signal content with a finite number of symmetric atoms will either lead to a non-sparse solution or pre-echo (*dark energy*) [16] [7]. Notice the dark energy formation before the signal onset (pre-echo) in Figure 3.2, which shows an MP decomposition example of a damped sinusoid onto a dictionary of Gabor atoms. Several algorithms aim to select symmetric atoms such that it minimizes the audible effect of pre-echo [17] [7] though they slow down MP and generally lead to a non-sparse solution. [20] noted that using asymmetric atoms can help to avoid the creation of pre-echo and dark energy, and [19] showed that damped sinusoids can sparsely represent temporal asymmetries and strong transients.

3.3 Asymmetric Atoms

This section includes an in-depth comparison of existing asymmetric atoms and introduces a new asymmetric atom, the *ramped exponentially damped sinusoid* (REDS). The REDS can adapt to a range of audio signal features and has mathematical properties that enable efficient sparse decompositions and synthesis.

3.3.1 Damped Sinusoid

The damped sinusoid (DS) is essential in audio as it represents a vibrating mode of a resonant structure. The use of a DS model in the context of analysis dates back to Prony's method [47], according to our knowledge, and was the first asymmetric atom used in the context of sparse representations [19].

Properties

Staying with the predefined generic atom expression (3.1):

$$A_{DS}[n] = 1 \tag{3.6}$$

and thus $n_m = n_I = 0$. Its continuous-time Fourier transform is well known,

$$\mathcal{F}[\boldsymbol{\phi}_{DS}](\omega) = \frac{1}{\alpha + i(\omega - \omega_c)} \tag{3.7}$$

as is the DFT,

$$\mathcal{F}[\boldsymbol{\phi}_{DS}](\kappa) = \frac{1 - e^{N(-\alpha + i(\omega_c - 2\pi\kappa/N))}}{1 - e^{-\alpha + i(\omega_c - 2\pi\kappa/N)}} \tag{3.8}$$

and finally, the Z-transform,

$$\mathcal{Z}[\boldsymbol{\phi}_{DS}](z) = \frac{1}{1 - e^{-\alpha + i\omega_c} z^{-1}} \tag{3.9}$$

Figure 3.3 shows the complex damped sinusoid's digital filter block diagram. The DS' spectrum is unimodal but not concentrated. We establish the analytic inner product for two DS atoms:

$$\langle \phi_{DS_1}, \phi_{DS_2} \rangle = \frac{1 - e^{N(-\alpha_1 - \alpha_2 + i(\omega_{c_1} - \omega_{c_2}))}}{1 - e^{-\alpha_1 - \alpha_2 + i(\omega_{c1} - \omega_{c_2})}} \tag{3.10}$$

and the analytic cross-correlation formula:

$$\langle \phi_{DS_1}, \phi_{DS_2} \rangle[m] = \begin{cases} a^{M_2} e^{m(\alpha_1 - i\omega_{c_1})} - e^{(\alpha_2 + i\omega_{c_2})(-m+1) - (\alpha_1 - i\omega_{c_1})} & \text{for } -M_2 \leq m \leq 0, \\ a^{M_1} e^{m(-\alpha_2 - i\omega_{c_2})} - e^{(\alpha_1 - i\omega_{c_1})(m-1) - (\alpha_2 + i\omega_{c_2})} & \text{for } 0 < m \leq M_1, \end{cases} \tag{3.11}$$

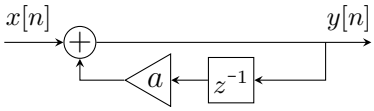


Figure 3.3 Damped sinusoid's digital filter block diagram per (3.9), where $a = e^{-\alpha+i\omega_c}$.

where $a = e^{\alpha_1+\alpha_2+i(\omega_{c_2}-\omega_{c_1})}$, $M_2 = (N_2 - 1)$, $M_1 = (N_1 - 1)$, and m is a discrete lag (shift) variable, $m = \tau_2 - \tau_1$. Although a damped sinusoid theoretically decays for an infinite duration, and thus warrants a cross-correlation derivation with an upper bound of positive infinity, in practice, atoms are of finite duration N, so we must use an upper bound for the cross-correlation derivation based on each atom's respective duration, N_1 and N_2. When the damped sinusoid's amplitude at sample N is less than -60 dB, we could negate the difference between an analytic cross-correlation formula that accounts for truncation and one that does not. However, (3.11) accounts for atoms of finite duration in case either ϕ_{DS_1} or ϕ_{DS_2} has an amplitude greater than -60 dB at sample N_1 or N_2, respectively.

3.3.2 Gammatone

Auditory filter models are designed to emulate cochlea processing and are central to applications like perceptual audio coding, where auditory filters are used to determine which sounds should be coded or not according to auditory masking principles. Auditory filter modeling has a variety of applications in bio-mechanics and psychoacoustic research.

The most popular auditory filter model is the gammatone (GT) filter due to its history and simple time domain expression. Originally described in 1960 as a fitting function for basilar displacement in the human ear [40], the gammatone filter was later found to precisely describe human auditory filters, as proven from psychoacoustic data [48]. [21] shows that atoms learned optimally from speech and natural sounds resemble gammatones. Designing gammatone filters remains a focus in audio signal processing [22].

More recently, filter models closely related to the gammatone filter have emerged, such as the all-pass gammatone filter and the cascade family [49]. Added features of these variants do not overlap with our criteria so they are not included for comparison.

Properties

We assign the gammatone as the prototypical auditory filter model. A single variable polynomial envelope function shapes the gammatone:

$$A_{GT}[n] = n^p \tag{3.12}$$

Literature involving the gammatone typically calls $p+1$ the filter order. A_{GT} is not asymptotic. $n_m = \frac{p}{\alpha}$ and $n_I > 2n_m$. No part of the gammatone is, strictly speaking, a freely decaying sinusoid (excluding when $p = 0$, in which case it is a DS), though it asymptotically approaches a DS as $n \to \infty$.

We demonstrate the filter order's effect by applying the Fourier transform frequency differentiation property to express its spectrum parametrized by p:

$$\mathcal{F}[\boldsymbol{\phi}_{GT}](\omega) = \frac{p!}{(\alpha + i(\omega - \omega_c))^{p+1}} \tag{3.13}$$

From its frequency representation, we see that the filter order determines the denominator polynomial order. Finally, referencing the convolution property of the Fourier transform, the gammatone impulse response is a DS convolved with itself p times.

Frequency spread Δf decreases with respect to the model order, while the time spread Δt increases. A gammatone of order four ($p = 3$) correlates best with auditory models [22]. The gammatone's spectrum is unimodal and concentrated.

The attack envelope is not parametrized, and therefore cannot be controlled independently of α. After setting p, controlling the atom is solely through α and ω_c. Influence time (or skirt width) is not directly controllable, so one cannot tune the atom to have time concentration in exchange for frequency spread. Thus, the adaptability of this model to a range of sound signal behavior is limited.

We establish an analytic formula for the gammatone's Z-transform that supports an arbitrary integer $p > 0$:

$$\mathcal{Z}[\boldsymbol{\phi}_{GT}](z) = \frac{\sum_{r=1}^{p} \left\langle {p \atop r-1} \right\rangle (e^{-\alpha + i\omega_c} z^{-1})^r}{(1 - e^{-\alpha + i\omega_c} z^{-1})^{p+1}} \tag{3.14}$$

where the Eulerian number $\left\langle {p \atop r-1} \right\rangle = \sum_{j=0}^{r} (-1)^j \binom{p+1}{j} (r-j)^p$. Figure 3.4 shows the gam-

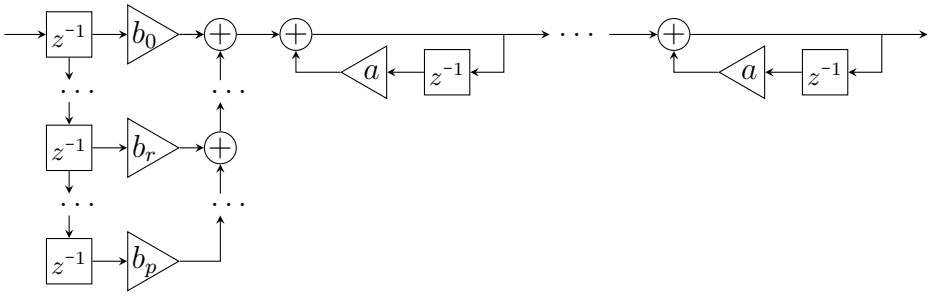


Figure 3.4 Gammatone digital filter block diagram per (3.14), where $a = e^{-\alpha+i\omega_c}$ and $b_r = a^r \left\langle {p \atop r-1} \right\rangle$.

matone's digital filter block diagram. The gammatone's DFT is

$$\mathcal{F}[\boldsymbol{\phi}_{GT}](\kappa) = (-i)^p \frac{d^p}{d\omega_\kappa^p} \left(\frac{1 - e^{N(-\alpha+i(\omega_c-\omega_\kappa))}}{1 - e^{-\alpha+i(\omega_c-\omega_\kappa)}} \right) \tag{3.15}$$

where $\omega_\kappa = 2\pi\kappa/N$.

We establish the gammatone's inner product formula by using the following property: $\frac{de^{-\alpha n}}{d\alpha} = -ne^{-\alpha n}$. We can therefore retrieve the gammatone inner product expression by differentiating the DS's inner product formula (3.10), with respect to either α_1 or α_2, p_1+p_2 times:

$$\langle \phi_{GT_1}, \phi_{GT_2} \rangle = (-1)^{p_1+p_2} \frac{d^{p_1+p_2}}{d\alpha_1^{p_1+p_2}} \left(\frac{1 - e^{N(-\alpha_1-\alpha_2+i(\omega_{c_1}-\omega_{c_2}))}}{1 - e^{-\alpha_1-\alpha_2+i(\omega_{c1}-\omega_{c_2})}} \right) \tag{3.16}$$

The same methodology applies for finding the gammatone's analytic cross-correlation formula from (3.11). These formulas are complicated for $p_1 + p_2 > 3$.

3.3.3 Formant-Wave-Function

In the source-filter model, an output sound signal results from an excitation function sent into a (resonant) filter, called a source-filter pair [39]. Most acoustic instruments involve an exciter, either forced or free, and a resonator [27]. A source-filter model of sound production is an appropriate model for an instrument that has a resonator that is not coupled with the source of excitation. An example is the voice production system, where the vocal tract filters glottal pulses.

Source-filter synthesis involves sending an excitation function through one or more resonant filters in parallel. The filters are typically one or two pole and defined by their

auto-regressive filter coefficients. The excitation function is either an impulse or, more often, an impulse that a window smooths to emulate natural excitation. The window shape effects the transient portion of the time-domain output from the system, and the skirts of the spectral envelope. The filter coefficients control the shape of the spectral envelope near the resonant peak.

Time-domain formant-wave-function synthesis describes the output of the source-filter model by a single function in the time domain. The amplitude envelope of the function generically matches the output envelope of a source-filter pair: a damped exponential (filter) with a smooth or discontinuous onset (excitation). The advantage of this approach is twofold: the formant-wave-function's time-domain definition enables the direct control of its spectrum through its parameters and synthesis by table lookup [39].

Properties

The formant-wave-function (FOF) is ubiquitous with time-domain wave-function synthesis. [39] proposed its use because it has the following desirable properties: its spectral envelope is compact and allows for flexible control over its shape through only two parameters, while its amplitude envelope's temporal evolution matches that of a source-filter synthesized waveform. The FOF's A is:

$$A_{FOF}[n] = \begin{cases} \frac{1}{2}\left(1 - \cos(n\beta)\right) & \text{for } 0 \leq n \leq \frac{\pi}{\beta}, \\ 1 & \text{for } \frac{\pi}{\beta} < n. \end{cases} \tag{3.17}$$

where $\beta \in \mathbb{R}_{>0}$ controls influence time. Decreasing β increases influence time, $n_I \approx \frac{\pi}{\beta}$, and the time location of the maximum,

$$n_m = \frac{1}{\beta}\cos^{-1}\left(\frac{\alpha^2-\beta^2}{\alpha^2+\beta^2}\right) \tag{3.18}$$

Δ_I and $\frac{\alpha}{\beta}$ are positively correlated.

A raised cosine is an excellent attack shape in terms of concentration, however, since it is piecewise (its value must be held at one after half of a period) some other design criteria suffer.

$$\mathcal{F}[\phi_{FOF}](\omega) = \frac{\beta^2}{2}\frac{1 + e^{-\frac{\pi}{\beta}(\alpha + i(\omega-\omega_c))}}{(\alpha + i(\omega-\omega_c))((\alpha + i(\omega-\omega_c))^2 + \beta^2)} \tag{3.19}$$

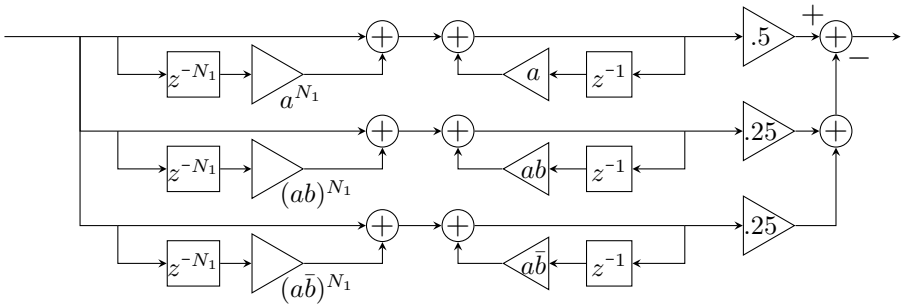


Figure 3.5 Formant-wave-function digital filter block diagram per (3.21), where $a = e^{-\alpha+i\omega_c}$ and $b = e^{i\beta}$.

The FOF's spectrum is not unimodal when the piecewise transition occurs within the window of observation. Moreover, it is difficult to estimate the FOF's parameters and its analytic inner product formula is complicated [20].

We establish the FOF's DFT and Z-transform by converting the cosine function into a sum of complex exponentials and using the linear property. The FOF's DFT is

$$\mathcal{F}[\boldsymbol{\phi}_{FOF}](\kappa) = \frac{1}{2}\frac{1 + a_\kappa^{N_1} - 2a_\kappa^N}{1 - a_\kappa} - \frac{1}{4}\left(\frac{1 - (a_\kappa e^{i\beta})^{N_1}}{1 - a_\kappa e^{i\beta}} + \frac{1 - (a_\kappa e^{-i\beta})^{N_1}}{1 - a_\kappa e^{-i\beta}}\right) \tag{3.20}$$

where $a_\kappa = e^{\alpha+i(\omega_c - 2\pi\kappa/N)}$, and the FOF's Z-transform is

$$\mathcal{Z}[\boldsymbol{\phi}_{FOF}](z) = \frac{1}{2}\frac{1 + a^{N_1}z^{-N_1}}{1 - az^{-1}} - \frac{1}{4}\Big(\frac{1 - (ae^{i\beta})^{N_1}z^{-N_1}}{1 - ae^{i\beta}z^{-1}} + \frac{1 - (ae^{-i\beta})^{N_1}z^{-N_1}}{1 - ae^{-i\beta}z^{-1}}\Big) \tag{3.21}$$

where $a = e^{-\alpha+i\omega_c}$ and $N_1 = [\frac{\pi}{\beta}]$. Therefore the impulse response from the sum of three complex pole-zero filters in (3.21) is a formant-wave-function (FOF), see Figure 3.5. The time-varying input delay complicates controlling attack shape.

3.3.4 Recapitulation

The existing asymmetric atoms have several desired properties missing. While the gammatone's unimodal frequency spectrum and time-domain simplicity are appealing, expressing its DFT and inner product is complicated. Most importantly, without a parameter to control influence time, the gammatone is not flexible enough to sparsely represent a variety of signal features. On the other hand, the FOF's attack function enables precise control over its spectral envelope, however, its piecewise construction is problematic: spectral rip-

ples result from a truncation in time, refining its parameters is difficult, and its frequency, Z-transform, and inner product expressions are complicated.

3.3.5 Towards a New Atom

The starting goal of this study was to design a $C^\infty A$ that is similar to A_{FOF}. While piecewise construction is the reason for the FOF's shortcomings, approximating the raised cosine with a C^∞ function does not necessarily improve the situation because many functions admit complicated frequency-domain and Z-domain formulas when their definitions include a unit step. For example, $(1 - e^{-\beta n^2})$ has a compact bell shape that seems to be, at first inspection, a good candidate to replace the raised cosine. However, after the introduction of a unit step function, it admits a non-algebraic Fourier transform expression (a special function defines the imaginary part). Many bell-shaped functions have the same problem (e.g., $\tanh\,(\beta n)^2$).

On the other hand, there are A options that are simple but have Δ_I that are large compared to the FOF for equal n_m. In fact, any C^∞ function will have a larger Δ_I than the FOF's for equal n_m. Therefore, our goal became more specific: define a C^∞ A that admits simple mathematical expressions when married with a complex damped exponential, and whose Δ_I is close to that of the FOF's for equal n_m. After an exhaustive search, we resolved that designing a function to satisfy all of the design criteria is difficult.

3.3.6 Ramped Exponentially Damped Sinusoid

To reflect generality, we call the new atom the *ramped exponentially damped sinusoid (REDS)*. Identically to existing source-filter and auditory filter models, a complex exponentially damped sinusoid defines the atom's decay section. A binomial with one exponential term shapes the atom's attack envelope. By defining the atom as a sum of exponentials (see (3.24)), we satisfy all the desirable mathematical properties of this study. The main idea is that we can use the linear property of the Fourier transform and Z-transform to segment the derivation into several transforms of complex exponentials because the complex damped exponential's transforms are simple and well known.

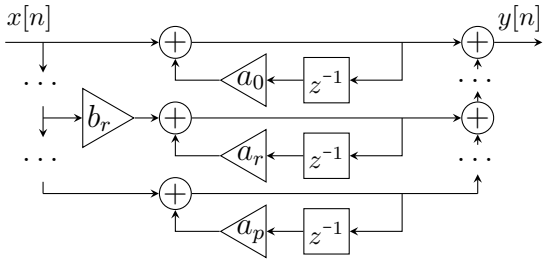


Figure 3.6 REDS digital filter block diagram per (3.27), where $a_r = e^{-\alpha - r\beta + i\omega_c}$ and $b_r = (-1)^r \binom{p}{r}$.

Properties

We define REDS concisely in the time-domain by expressing $A_{REDS}[n]$ polynomially as $(1 - e^{-\beta n})^p$:

$$\phi[n] = \left(1 - e^{-\beta n}\right)^p e^{n(-\alpha + i\omega_c)} u[n] \tag{3.22}$$

where β controls the influence time (or skirt width) and $p + 1$ is the order.

$$n_m = \tfrac{1}{\beta} \log(1 + \tfrac{p\beta}{\alpha}) \tag{3.23}$$

and $n_I \approx -\frac{1}{\beta} \log(1 - (1 - \delta)^{1/p})$, where δ is the same as in Section 3.1.3.

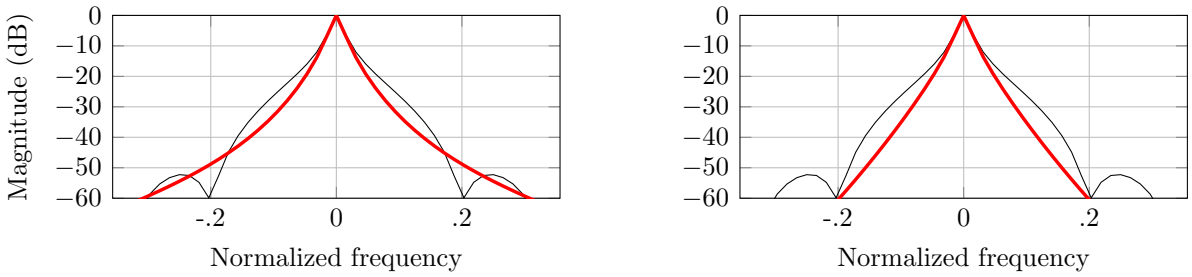
As in the gammatone model, order is often constant within an application: we may choose the order, for example, to match with auditory data or to approximate a frame condition [22]. Given that the order is a constant, the number of control parameters and their effect are the same as the FOF. To summarize, the REDS parameter set is a conflation of the source-filter and auditory filter models.

We express the REDS in binomial form to reveal its sum of exponentials construction:

$$\phi[n] = \sum_{r=0}^{p} (-1)^r \binom{p}{r} e^{n(-\alpha - r\beta + i\omega_c)} u[n] \tag{3.24}$$

where the binomial coefficient $\binom{p}{r} = \frac{p!}{(p-r)!p!}$. Considering the linear property of the Fourier transform, we readily find from (3.24) the Fourier transform of REDS:

$$\mathcal{F}[\boldsymbol{\phi}_{REDS}](\omega) = \sum_{r=0}^{p} (-1)^r \binom{p}{r} \frac{1}{\alpha + r\beta + i(\omega - \omega_c)} \tag{3.25}$$



(a) $p = 2$. (b) $p = 10$.

Figure 3.7 $\mathcal{F}[\boldsymbol{\phi}_{FOF}](\kappa)$ (black) and $\mathcal{F}[\boldsymbol{\phi}_{REDS}](\kappa)$ (red, bold) with constant β and $\alpha = .05$. The spectrum of REDS is unimodal while the FOF's is non-unimodal. REDS is more frequency-selective than the FOF for $p > 2$.

and the DFT:

$$\mathcal{F}[\boldsymbol{\phi}_{REDS}](\kappa) = \sum_{r=0}^{p} (-1)^r \binom{p}{r} \frac{1 - e^{N(-\alpha - r\beta + i(\omega_c - 2\pi\kappa/N))}}{1 - e^{-\alpha - r\beta + i(\omega_c - 2\pi\kappa/N)}} \tag{3.26}$$

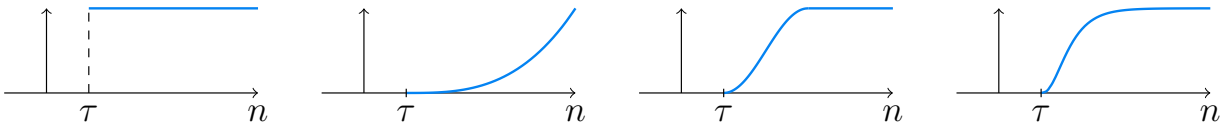
Finally, we apply the linear property to retrieve the Z-transform:

$$\mathcal{Z}[\boldsymbol{\phi}_{REDS}](z) = \sum_{r=0}^{p} (-1)^r \binom{p}{r} \frac{1}{1 - e^{-\alpha - r\beta + i\omega_c} z^{-1}} \tag{3.27}$$

A sum of $p + 1$ complex one-pole filters in parallel will thus output a REDS (see Figure 3.6).

The REDS has a concentrated and unimodal spectrum. Similarly to the FOF, it is possible to precisely control REDS' spectra: by varying β one may exchange concentration in time for frequency, and vice versa. The FOF has greater time concentration than the REDS because the raised cosine attack function has a fast uniform transition from zero to one, while the REDS attack envelope is bell-shaped. Formally, $n_{I_{REDS}} > n_{I_{FOF}}$ when $n_{m_{REDS}} = n_{m_{FOF}}$. REDS' spectral concentration surpasses the FOF's as p increases, see Figure 3.7.

Since the REDS is constructed from a linear combination of $p + 1$ damped sinusoids, the inner product is equal to the sum of $(p_1 + 1)(p_2 + 1)$ damped sinusoid inner products $\langle \phi_{DS_1}, \phi_{DS_2} \rangle$, see (3.10). Likewise, the cross-correlation of two REDS atoms is equal to the sum of $(p_1 + 1)(p_2 + 1)$ damped sinusoid cross-correlations $\langle \phi_{DS_1}, \phi_{DS_2} \rangle[m]$, see (3.11).



(a) DS (b) GT (c) FOF (d) REDS

Figure 3.8 Asymmetric atom attack envelopes, $A[n]$.

Considering that these formulas for Gabor atoms and FOFs provide an efficiency boost in existing MP algorithms [20], and the REDS formulas are simpler than those, it can only be that using the formulas are more efficient than numerical computations.

3.3.7 Relations

There are a few important relations between the attack envelopes, $A[n]$, of the aforementioned asymmetric atoms, see Figure 3.8 for examples of each atom's $A[n]$. By applying the small angle approximation to $A_{FOF}[n]$, we show that a FOF and gammatone of $p = 2$ are approximately equal when the FOF's β is very small:

$$\lim_{\beta \to 0} \tfrac{2}{\beta^2} \left(1 - \cos(\beta n)\right) = n^2 \tag{3.28}$$

We wish to quantify the approximation error in terms of β. To do this, we use the series expansion of $A_{FOF}[n]$ about $\beta = 0$:

$$\left(1 - \cos(\beta n)\right) = \tfrac{1}{2} n^2 \beta^2 - \tfrac{1}{24} n^4 \beta^4 + \tfrac{1}{720} n^6 \beta^6 \ldots \tag{3.29}$$

$$\frac{2}{\beta^2} \left(1 - \cos(\beta n)\right) = n^2 (1 - \tfrac{1}{12} n^2 \beta^2 + \tfrac{1}{360} n^3 \beta^3 \ldots) \tag{3.30}$$

$$\frac{2}{\beta^2} \left(1 - \cos(\beta n)\right) \approx n^2 (1 - \epsilon) \tag{3.31}$$

where $\epsilon = \frac{1}{12} n^2 \beta^2$ is the error between the FOF and gammatone. The error increases with n, however, we know that $A_{GT}[n]$ only influences the atom for a duration of $n_I = 2n_m = \frac{2p}{\alpha} = \frac{4}{\alpha}$. Assuming that $\beta \ll \alpha$ such that the GT and FOF's n_m values are approximately equal, $\epsilon = \frac{1}{12} n_I^2 \beta^2 = \frac{1}{12} (\frac{4}{\alpha})^2 \beta^2$ and

$$\beta = \alpha \sqrt{\tfrac{3}{4} \epsilon} \tag{3.32}$$

Likewise, by applying the small angle approximation to $A_{REDS}[n]$, we show that A_{REDS} and A_{GT} are approximately equal when β is small:

$$\lim_{\beta \to 0} \tfrac{1}{\beta^p} \left(1 - e^{-\beta n}\right)^p = n^p \tag{3.33}$$

We quantify the approximation error between a gammatone and REDS in terms of β by using the series expansion of $A_{REDS}[n]$ about $\beta = 0$:

$$(1 - e^{-\beta n})^p = (\beta n)^p (1 - \tfrac{1}{2} np\beta + \tfrac{1}{24} n^2 \beta^2 p(1 + 3p) \ldots) \tag{3.34}$$

$$\frac{1}{\beta^p}(1 - e^{-\beta n})^p = n^p (1 - \tfrac{1}{2} np\beta + \tfrac{1}{24} n^2 \beta^2 p(1 + 3p) \ldots) \tag{3.35}$$

$$\frac{1}{\beta^p}(1 - e^{-\beta n})^p \approx n^p (1 - \epsilon) \tag{3.36}$$

where $\epsilon = \frac{1}{2} np\beta$ is the error between the REDS and gammatone. We substitute $n_I = 2n_m = \frac{2p}{\alpha}$ for n, assuming that $\beta \ll \alpha$ so their n_m values are approximately equal, and solve for β:

$$\beta = \frac{\alpha \epsilon}{p^2} \tag{3.37}$$

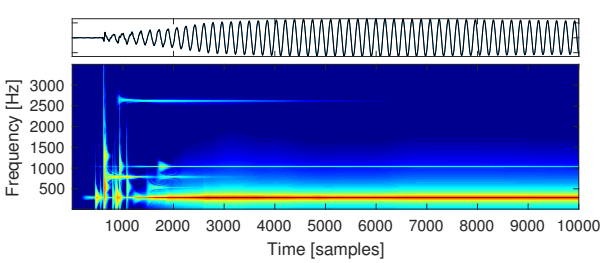
REDS ability to approximate a gammatone, with a quantifiable error, is useful for several reasons. In practice, the perceptual difference between a REDS and gammatone is negligible when $\epsilon < .001$. Regarding dictionary based methods, a homogeneous REDS dictionary can contain approximate gammatone atoms. If one wants to design a gammatone-only dictionary, which is common for research in perceptual auditory coding, a REDS dictionary is a practical alternative because it has a simpler inner product formula. Moreover, a REDS filter requires fewer mathematical operations per sample than a gammatone filter, and can range in envelope shape from a gammatone to a damped exponential and anywhere in-between. Furthermore, by (3.28) and (3.33), $A_{REDS}[n] \approx 2A_{FOF}[n]$ when $p = 2$ and their β values are $\frac{1}{4}\epsilon\alpha$.

3.4 Sparse approximation experiment

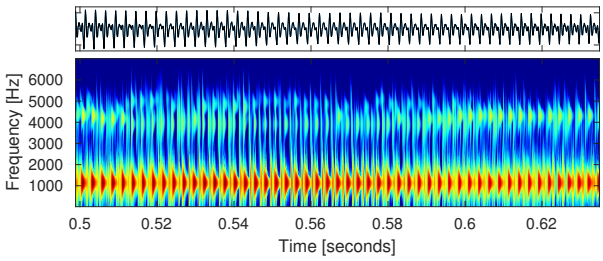
We decomposed a set of real audio signals with MP. We selected the audio signal set to reflect a range of the source-filter model: it includes a vocal sound (sustained, relatively high damping and smooth attack per atom), a vibraphone (not-sustained, made of low

	Dur.	Atoms	N	SRR (dB) DS	GT	FOF	REDS
Vocal	1.0	10^4	2^8	30.7	35.9	37.8	38.8
Violin	1.6	10^4	2^9	20.0	14.6	27.7	28.0
Vibes	5.5	50	2^{17}	17.6	32.1	36.9	37.1

Table 3.1 Asymmetric atom sparse approximation comparison results. The sampling rate of the audio signals is 44100 Hz. The signal's duration, "Dur.", is in seconds.



(a) Vibraphone decomposition onto 50 REDS atoms (transient part shown).



(b) Singing voice. Atom spacing expands/contracts reflecting vibrato.

Figure 3.9 Time-frequency distribution of REDS from sparse approximation experiment.

damping and short attack per atom), and a violin (intermediate situation).

For this comparison study, the MP algorithm did not employ any refinement techniques and thus decomposed each signal onto each static dictionary $\mathbf{\Phi}_{DS}$, $\mathbf{\Phi}_{GT}$, $\mathbf{\Phi}_{FOF}$, and $\mathbf{\Phi}_{REDS}$. The manual dictionary design process involved fitting a dictionary of damped sinusoids to each signal, $\mathbf{\Phi}_{DS}$, then creating the other three dictionaries by modulating $\mathbf{\Phi}_{DS}$ with $\boldsymbol{A}_{GT}$, $\boldsymbol{A}_{FOF}$, and $\boldsymbol{A}_{REDS}$.

We can represent a signal as time-varying partials per the additive model, or as filtered excitation sequences per the source-filter model, by decomposing it onto a dictionary of REDS atoms with constrained damping factors. We chose to demonstrate the ability of the REDS to analyze the signal set from the source-filter viewpoint. For the singing voice, if the dictionary contained atoms with small damping (long time support) then the selected atoms would represent partials of the signal. We set the damping to be high and in doing so, successfully represented spectral formants with a series of short duration atoms with relatively large skirt widths whose temporal spacings, rather than frequencies, oscillated

Criteria	DS	GT	FOF	REDS
Concentrated Spectrum	–	✓	✓	✓
Unimodal Spectrum	✓	✓	–	✓
Influence Time Control	–	–	✓	✓
Time-Domain Simplicity	✓	✓	–	✓
Causal Filter Simplicity	✓	✓	–	✓
Inner Product Simplicity	✓	–	–	✓

Table 3.2 Asymmetric atom comparison results.

to reflect the sound's vibrato, see Figure 3.9b. Regarding the vibraphone, we created a dictionary whose damped sinusoids had large time support with low decay rates.

For each test, the REDS dictionaries provided higher signal-to-residual ratio (SRR) values for the same number of iterations, see Table 3.1, where

$$\text{SRR} = 20 \log_{10} \left(\frac{\|\mathbf{y}\|_2^2}{\|\mathbf{r}\|_2^2} \right) \tag{3.38}$$

For the singing voice, the gammatone and REDS were close in performance because the formant time-domain envelopes had very smooth attacks. REDS matched the vibraphone's envelope tightly, while the gammatone caused pre-echo because it is more symmetric than the signal. The reconstructed signal from the REDS decomposition had an SRR of 38.8 dB, and consisted of 50 atoms (.04% of the signal's length), see Figure 3.9a.

3.5 Summary

In this chapter, we described several types of atoms for sparse representations of audio. We established a set of desirable properties for an asymmetric atom, e.g., the ability to adapt to a range of audio signal features. Then, we compared existing asymmetric atoms along those criteria and introduced a new asymmetric atom, REDS. We established relevant analytical formulas for each atom, for example, analytic inner product formulas, and established under what conditions some of the atoms are approximately equivalent. Table 3.2 summarizes the results of the asymmetric atom comparative study and Figure 3.10 shows each asymmetric atom's envelope $E[n]$ and corresponding frequency spectrum. Given that REDS meets all of our requirements, the next step involves the design of a sparse representation system

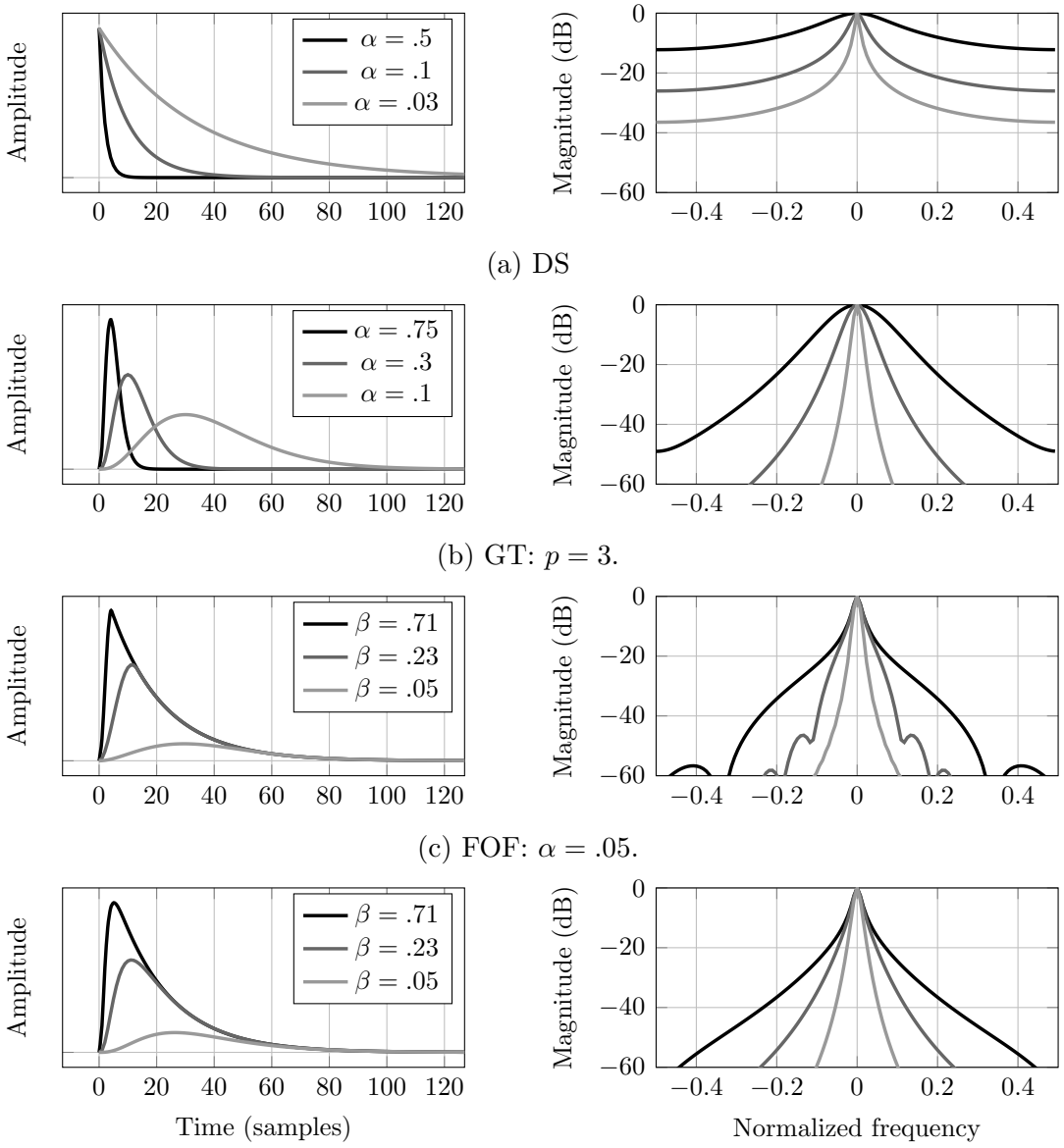


Figure 3.10 Asymmetric atom envelopes $E[n]$ (left) and magnitude normalized spectra (right).

that decomposes an arbitrary audio signal onto REDS atoms.

Chapter 4

Partial Tracking Matching Pursuit

In this chapter, we establish an MP-based system that adapts parametric asymmetric atoms (REDS) to an arbitrary audio signal. It bridges two separate search methods, one that efficiently locates short duration atoms (e.g. on the order of milliseconds) and the other that efficiently extracts long duration atoms (e.g., whose durations last seconds or even minutes). We call the system *partial tracking matching pursuit (PTMP)* because it locates long duration atoms by extracting and reformatting long horizontal partials.

Partial tracking algorithms output partial trajectories that describe the evolution of sinusoidal model parameters over time. Research about additive sound synthesis per the sinusoidal model has led to several publications on partial tracking techniques, the first of its kind being [50] with recent developments in [51] [52]. Daudet emphasized that dictionaries with long duration atoms limit the speed of MP, then addressed the problem by proposing the Molecular Matching Pursuit algorithm that represents a long duration oscillation, not with a long duration atom, but with a "molecule", which is a group of short duration atoms of similar frequencies.

For the first time, we employ the method of partial tracking in a sparse approximation context to efficiently locate atoms of arbitrary duration and overcome part of the sparse approximation's scalability limitations. PTMP outputs REDS parameter set $\boldsymbol{\lambda}^{(k)}$ after iteration $k = (1, 2, \ldots, K)$.

4.1 Partials extraction

The first stage of PTMP's large-scale sub-system extracts partial trajectories, $\mathbf{P}_\rho$ for $\rho = (1, 2, \ldots, R)$, and prepares them for transformation into asymmetric atoms by formatting, splitting, and arranging them to fit with an asymmetric atom model.

4.1.1 Peak picking

The first step involves comparing audio signal $\mathbf{y}$ with time-shifted Fourier atoms and finding prominent atoms (peaks) in the time-frequency plane. A discrete Fourier atom $\phi_\kappa[n]$ is a complex sinusoid modulated by a real and symmetric window $E[n]$:

$$\phi_\kappa[n] = E[n]e^{2\pi i\kappa n/N_{PT}} \tag{4.1}$$

where $\kappa = n = (0, 1, 2, \ldots, N_{PT}-1)$, and $N_{PT} \in \mathbb{N}$ is the atom's discrete length (PT stands for partial tracker). $E[n]$ is normalized such that $\|E[n]\| = 1$, and by extension $\|\phi_\kappa[n]\| = 1$. PTMP uses a symmetric window, more specifically a Blackman-Harris window [42], for the peak picking and the forthcoming spectral analysis step because these tasks demand a compact spectrum with high side-lobe rejection.

We start by calculating the discrete STFT of $\mathbf{y}$, $\boldsymbol{S}_y$,

$$\boldsymbol{S}_y = S_y[m, \kappa] = \langle y[n + mH], \phi_\kappa[n]\rangle = \sum_{n=0}^{N-1} y[n + mH]E[n]e^{-i2\pi\kappa n/N_{PT}} \tag{4.2}$$

where $m \in \mathbb{N}$ is the frame index and $H \in \mathbb{N}$ is the hop size. After calculating the STFT, we detect and determine the peaks of the spectrum for every observation frame m, $\hat{\boldsymbol{\kappa}}^m$.

Bin κ is a peak, $\hat{\kappa}$, when it fulfills several criteria. Besides the definitive criteria for a peak that [50] describes, our peak selection criteria involves a local (inside of one observation frame) relative minimum peak height that [53] establishes (with details in [54]), and another one that we establish to retain a global selectivity over the entire audio signal. We convert the magnitude spectrum to decibels (dB), denoted with a dB subscript: $|S_y[m, \kappa]|_{dB} = 20\log_{10}|S_y[m, \kappa]|$. First of all, per [50], $|S_y[m, \kappa]|_{dB}$ must be a local maximum in the magnitude spectrum, more precisely,

$$|S_y[m, \kappa - 1]|_{dB} < |S_y[m, \kappa]|_{dB} > |S_y[m, \kappa + 1]|_{dB} \tag{4.3}$$

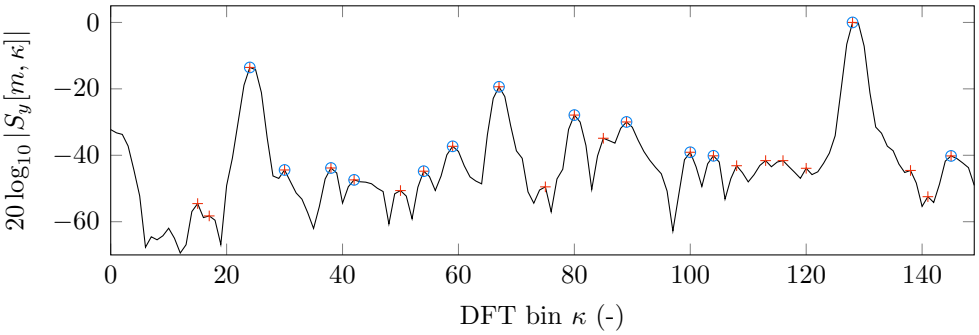


Figure 4.1 Spectral peak picking. The orange marks locate the local maxima that satisfy (4.5) and the blue circles locate the local maxima that satisfy both (4.4) and (4.5), where $G_g = 60\,\text{dB}$ and $G_h = 10\,\text{dB}$.

Second, [54] proposed a relative minimum peak height threshold that adds another level of selectivity to the peak picking decision that helps to avoid the selection of spurious peaks, more precisely, peaks that are likely not the result of an underlying sinusoidal component. We adopt the same technique: $|S_y[m,\kappa]|_{dB}$ must be G_h greater than the averaged magnitude of its neighboring valleys,

$$|S_y[m,\kappa]|_{dB} > \frac{1}{2}(|S_y[m,\kappa-]|_{dB} + |S_y[m,\kappa+]|_{dB}) + G_h \tag{4.4}$$

where G_h is the relative height threshold, and $\kappa-$ and $\kappa+$ are the locations of the valleys (local minima) to the left and right of κ.

Lastly, $|S_y[m,\kappa]|_{dB}$ must be greater than the general amplitude threshold G_g. The value of G_g is relative to the absolute maximum of the signal's STFT, which includes all frames and bins, $|\boldsymbol{S}_y|_{dB}$,

$$|S_y[m,\kappa]|_{dB} > \max|\boldsymbol{S}_y|_{dB} - G_g \tag{4.5}$$

We establish a global selection criteria to reflect that MP selects the best fitting atom over the entire signal, not for every short-time analysis frame. This global amplitude constraint eliminates spectral peaks that have relatively low magnitudes compared to the global maximum. The combination of global selection and local peak height criteria help to eliminate noise-induced spurious peaks and peaks that originate from window $E[n]$'s side lobes rather than the signal itself. Figure 4.1 shows the $|S_y[m,\kappa]|_{dB}$ of a glockenspiel audio sample, superimposed with a marking at each peak, $|S_y[m,\hat{\kappa}]|_{dB}$.

4.1.2 Estimating sinusoidal model parameters

After locating L short-time Fourier transform peaks of frame m, $\hat{\kappa}_l^m$ for $l = (1, 2, \ldots, L)$, we employ the spectral information at the peak locations to extract the parameters of underlying complex damped sinusoids by using phase-based (spectral) non-stationary sinusoidal model estimators: the Reassignment Method [55] and Derivative Method [44]. To the best of our knowledge, these estimators have not intersected with sparse approximation applications.

First, we find an estimate of the underlying sinusoid's time center by employing the Reassignment Method. [55] explains how to calculate reassigned values efficiently using a ratio of Fourier transforms. This estimation requires the computation of another STFT, this time with a time-weighted version of $y[n + mH]$:

$$S_{y_\tau}[m, \kappa] = \langle y[n + mH](n - \tfrac{N_{PT}-1}{2}), \phi_\kappa[n] \rangle \tag{4.6}$$

The time center estimate of the sinusoid corresponding to $S_y[m, \hat{\kappa}]$ is

$$\hat{\tau}_l^m = \tau^m + \underbrace{\Re\left\{ \frac{S_{y_\tau}[m, \hat{\kappa}_l^m]}{S_y[m, \hat{\kappa}_l^m]} \right\}}_{\tau_\delta[\hat{\kappa}_l^m]} \tag{4.7}$$

where $\tau^m = mH + (N_{PT} - 1)/2$ is the discrete time center of frame m.

[51] used the reassignment method to avoid pre-echo creation when re-synthesizing a signal from sinusoidal partials per the additive sound model, and described a technique called "cropping" to better preserve the phase information of the signal ([36] details this idea further). An on-center component is one with a value of $|\tau_\delta[\hat{\kappa}_l^m]|$ that is less than some value, for example, less than the hop size H. An on-center component allows for reliable parameter estimates because it suggests that the sinusoid of interest is relatively stationary and spans the majority, if not all, of the analysis frame. Strong damping also effects τ_δ because it shifts a component's temporal center of energy away from τ^m.

We propose the use of reassignment method to avoid dark energy creation (e.g., in the form of pre-echo) in the context of greedy sparse approximations. Figure 4.2 shows a partial extracted from a test signal before time reassignment, where the time locations are the discrete time frame centers (τ^m) and the same partial after reassigning τ^m to $\hat{\tau}_l^m$

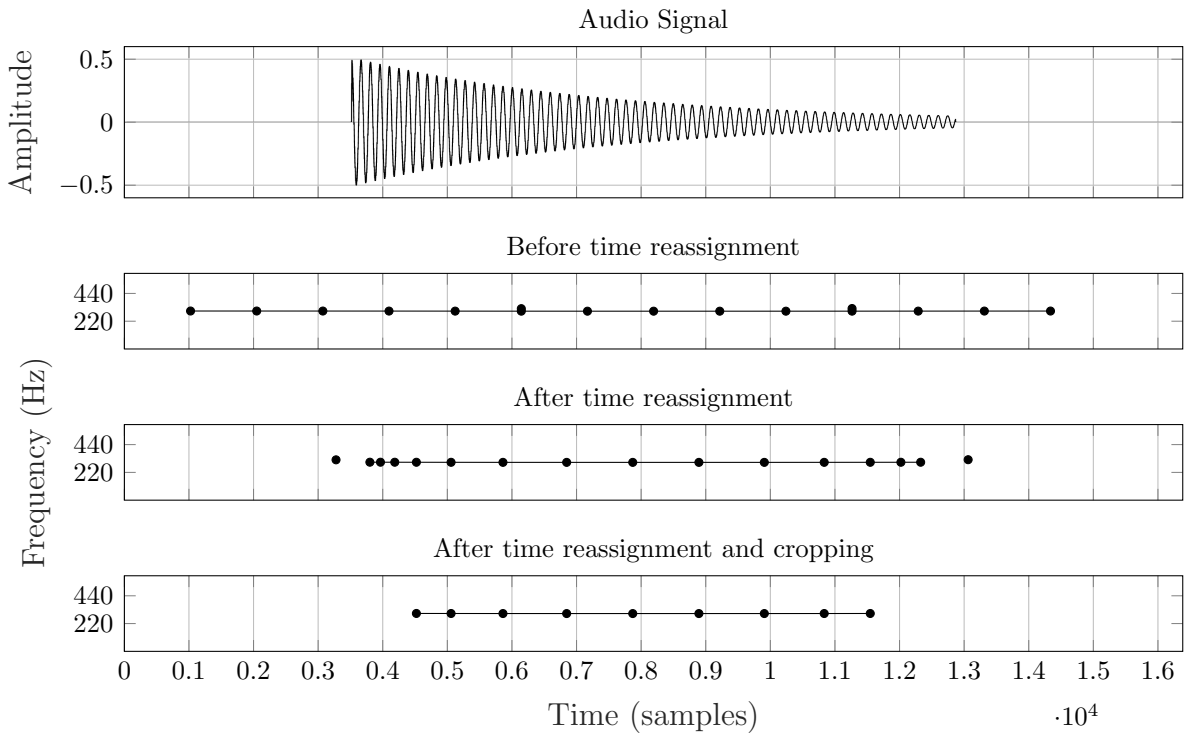


Figure 4.2 Partials (lines) with spectral peaks (dots) from a synthetic audio signal (top panel) before and after time reassignment and cropping. In this case, $N_{PT} = 8192$ and $H = 256$.

per (4.7). Notice how the partial onset is after the onset of the audio signal so there is no pre-echo. We adopt the cropping technique to ensure reliable parameter estimations as well. The bottom panel of the figure illustrates that the remaining reassigned spectral peak locations after cropping are the ones that are more reliably centered inside of the audio signal. PTMP proceeds to estimate the frequency and damping factor for on-center peaks, in other words, the ones that satisfy $|\tau_\delta[\hat{\kappa}_l^m]| \leq H$. We confidently discard peaks whose $|\tau_\delta[\hat{\kappa}_l^m]|$ is greater than the hop size H because we know another analysis frame has a better "view" of that component. At the same time, a component with a damping factor strong enough to make $|\tau_\delta[\hat{\kappa}_l^m]| > H$ suggests that it is relatively transient, so we safely discard the peak since the small-scale sub-system can detect and represent it sufficiently.

Next, we assume a complex damped exponential signal model and use the derivative method to estimate each on-center peak's frequency and damping factor. The derivative method requires the signal's time derivative $\mathbf{y}'$. Since the digital audio signal $\mathbf{y}$ is a discrete-

time vector, we must calculate its numerical derivative. Here, we employ the technique that [44] establishes to compute the signal's numerical derivative, which is to filter the signal with the following differentiator filter impulse response:

$$h[n] = \frac{(-1)^n}{n} \quad \text{for } n \neq 0, \text{ and } h[0] = 0. \tag{4.8}$$

where $n = \{0, 1, \ldots N_h - 1\} - (N_h - 1)/2$. In practice, an impulse response of order $N_h = 1023$ (order must be odd) provides an approximate signal derivative with negligible bias (except at very high frequencies) [44]. The signal's numerical derivative is $\mathbf{y}' = \mathbf{y} * \boldsymbol{h}$.

Alternatively, we could instead use the reassignment method for the estimation of damping and frequency because it provides an equivalent estimation accuracy as the derivative method [44]. Although it requires slightly more computations, the computational difference being the convolution of the signal with $h[n]$, we employ the derivative method in this case because it is the more flexible choice; the derivative method does not rely on having a differentiable window [56].

Another STFT, this time of the signal derivative, provides us with enough information to calculate $\hat{\omega}_l^m$ and $\hat{\alpha}_l^m$.

$$S_{y'}[m, \kappa] = \langle y'[n + mH], \phi_\kappa[n] \rangle \tag{4.9}$$

The frequency estimate for bin $\hat{\kappa}_l^m$ is

$$\hat{\omega}_l^m = \Im \left\{ \frac{S_{y'}[m, \hat{\kappa}_l^m]}{S_y[m, \hat{\kappa}_l^m]} \right\} \tag{4.10}$$

and the damping factor estimate is

$$\hat{\alpha}_l^m = -\Re \left\{ \frac{S_{y'}[m, \hat{\kappa}_l^m]}{S_y[m, \hat{\kappa}_l^m]} \right\} \tag{4.11}$$

Lastly, we store the magnitude of each peak of frame m, $a_l^m = |S_y[m, \hat{\kappa}_l^m]|$. We discard phase information because we are not synthesizing directly from the partial trajectories interpolated phase, which is the usual case for partial tracking applications. A later step determines atom phase.

4.1.3 Frame-to-Frame Peak Matching

PTMP creates sinusoidal trajectories by linking together spectral peaks per the heuristics [50] proposed (i.e., the McAulay-Quatieri method of peak matching). One difference between this procedure and the one that [50] describes is that we enforce an additional constraint onto the peak connection decision, more precisely, in addition to the local frequency deviation ω_δ that [50] describes, we also enforce a general frequency deviation ω_Δ to ensure the partial trajectory's frequency evolution is roughly stationary. Both constraint values are dimensionless.

The McAulay-Quatieri method of peak matching is sufficient in our case because we are interested in extracting long horizontal partials, more precisely, ones with little frequency modulation per the asymmetric atom model. If one wants to extract atoms with frequency modulation, it is beneficial to use a partial tracker that connects peaks in a globally optimal way and uses frequency modulation information to make better decisions, for example, the Hidden Markov Model partial tracker [57].

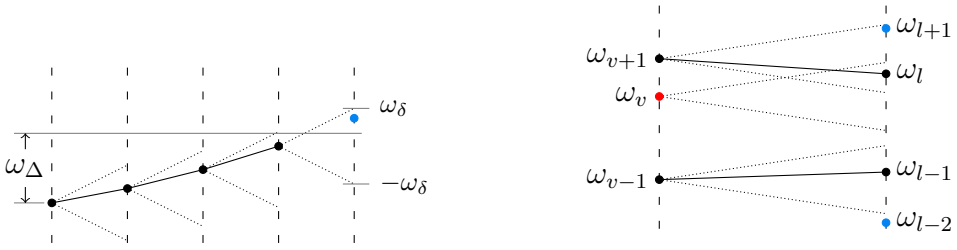
Let $\mathbf{P}_\rho$ be partial number ρ that contains a set of peak parameters, $\mathbf{P}_\rho = \{\omega^m, \alpha^m, \tau^m, a^m\}_\rho$, where $m \in [b_\rho, d_\rho]$ is frame index, b_ρ and d_ρ are the birth and death frames of $\mathbf{P}_\rho$, respectively, and M_ρ is $\mathbf{P}_\rho$'s length in frames, i.e., $M_\rho = d_\rho - b_\rho + 1$.

Suppose that we matched the peaks up to frame $m-1$ and generated a new parameter set for frame m from the aforementioned process. Let $\boldsymbol{\omega}^m \in \mathbb{R}^L$ and $\boldsymbol{\omega}^{m-1} \in \mathbb{R}^V$ be vectors that contain the frequencies of frame m and $m-1$, respectively (in general $L \neq V$). The method of assigning each frequency in frame m to some existing trajectory that contains frequency ω_v^{m-1} from frame $m-1$, or to a new trajectory, is as follows.

1. Calculate the relative difference $\Delta_{l,v} = |\omega_l^m / \omega_v^{m-1} - 1|$ for every combination of l and v.

2. Find the best match for the values that are still in consideration by choosing the combination that results in the minimum difference, $\{\tilde{l}, \tilde{v}\} = \arg\min_{l,v} \Delta_{l,v}$. There is no match for $\omega_{\tilde{v}}^{m-1}$ if $\Delta_{\tilde{l},\tilde{v}} > \omega_\delta$, so declare the trajectory that contains $\omega_{\tilde{v}}^{m-1}$ as dead and take $\omega_{\tilde{v}}^{m-1}$ out of further consideration. If $\Delta_{\tilde{l},\tilde{v}} \leq \omega_\delta$, proceed to check if $\omega_{\tilde{l}}^m$ is within the general frequency deviation range, ω_Δ.

   ω_Δ refers to the maximum amount that a partial trajectory's frequency is allowed to deviate. Let $\boldsymbol{\omega}_{\tilde{v}}$ denote a vector that contains the frequencies of partial trajectory



(a) Types of frequency deviation constraints, ω_δ and ω_Δ.

(b) Peak matching scenario

Figure 4.3 Peak-to-peak matching heuristics. The blue and red dots mark the point of a partial's birth and death, respectively.

$\mathbf{P}_{\tilde{v}}$ that owns $\omega_{\tilde{v}}^{m-1}$. To test this condition, calculate the maximum deviation between the new frequency and the partial trajectory's frequencies, $\max|\omega_l^m/\boldsymbol{\omega}_{\tilde{v}} - 1|$. If $\max|\omega_l^m/\boldsymbol{\omega}_{\tilde{v}} - 1| \leq \omega_\Delta$, assign ω_l^m to the partial trajectory that contains $\omega_{\tilde{v}}^{m-1}$, and take both frequencies out of further consideration. If $\max|\omega_l^m/\boldsymbol{\omega}_{\tilde{v}} - 1| > \omega_\Delta$, declare the trajectory that contains $\omega_{\tilde{v}}^{m-1}$ as dead and take $\omega_{\tilde{v}}^{m-1}$ out of further consideration.

Repeat this step until none of the values in $\boldsymbol{\omega}^{m-1}$ are in consideration (they are either matched or dead).

3. Birth a new partial trajectory for each frequency in $\boldsymbol{\omega}^m$ that was not assigned to an existing partial trajectory.

Existing partial tracking algorithms define the allowable amount of local frequency deviation linearly with respect to the peak frequency (in those cases the frequency deviation variable has a dimension of frequency) [50] [53]. Alternatively, we define ω_δ and ω_Δ as non-linear functions with respect to ω_v for two reasons. One reason is to reflect that the frequency modulation of a harmonic is relative to the modulation of its fundamental. The other reason is to reflect human auditory perception: humans are more sensitive to frequency variations when they occur at low frequencies than at high frequencies [26]. Therefore, we allow less frequency deviation at low frequencies than at high frequencies. Figure 4.3a illustrates how ω_δ and ω_Δ guide peak matching, and Figure 4.4 shows an example of the partial trajectories that the peak-to-peak matcher creates.

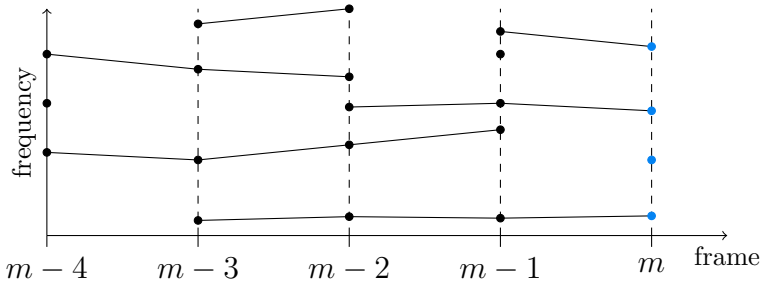


Figure 4.4 Partial trajectory formation example.

4.1.4 Splitting partials

After extracting partial trajectories over the entire signal, we reformat the partial trajectories to generally fit with an asymmetric atom model. At this point, the partial trajectories have frequency evolutions that are consistent with our asymmetric atoms because ω_Δ constrains them. However, we still need to reformat the partial trajectories such that their amplitude modulations are consistent with an asymmetric atom's envelope (i.e., the amplitude envelope has an attack part followed by a damped exponential part).

We split each partial trajectory per the following heuristic procedure. Let $\boldsymbol{a}_\rho$ be a vector containing the magnitudes of partial trajectory $\mathbf{P}_\rho$, where $a_\rho[m]$ is the magnitude of partial trajectory number ρ at location m. Find the valleys of $\boldsymbol{a}_\rho$: m is the location of a valley (local minimum) if

$$a_\rho[m-1] > a_\rho[m] < a_\rho[m+1] \tag{4.12}$$

Next, find the location of the closest local maximum in the positive direction of m, denoted by $m+$. If $a_\rho[m+] - a_\rho[m] \geq \nu_\delta$, where ν_δ is the valley threshold, split the partial at m: one partial ends and another one begins at m. Repeat this process for every valley in $\boldsymbol{a}_\rho$ and for every partial trajectory number ρ.

We split the partials based on amplitude modulation cues to ensure multiple superimposed audio events with close frequencies do not remain hidden under a single partial trajectory. We illustrate the problem with the following example. Suppose an audio signal contains a sequence of damped sinusoids that are close enough in time such that a new one starts while the previous one's amplitude is large enough to constitute a peak in the spectrum (per Section 4.1.1), and close enough in frequency such that their frequency differences are less than ω_δ and ω_Δ (per Section 4.1.3). Due to this frequency and time

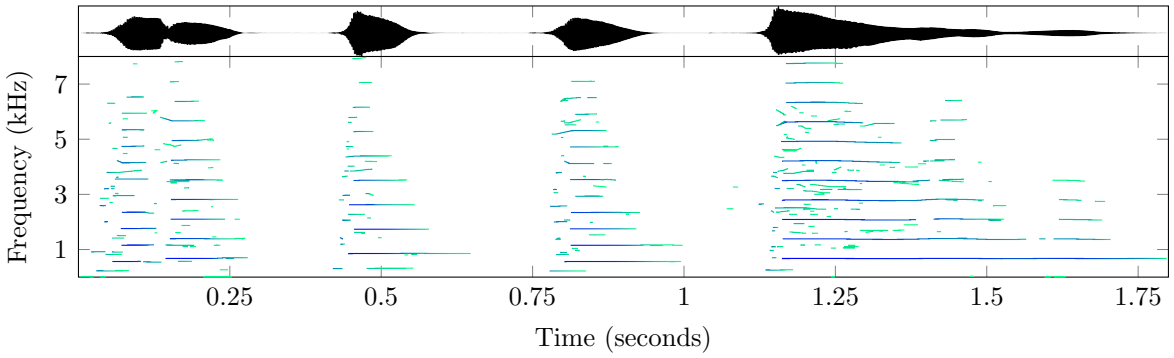


Figure 4.5 Partial trajectories from a clarinet audio signal.

overlap, the partial tracker will capture the sequence with one partial trajectory. We employ the partial trajectory's amplitude data to locate the onset of new asymmetric atoms. This enables a decomposition onto a sequence of multiple atoms, thereby more precisely reflecting the amplitude envelope of the audio signal at that particular frequency.

4.1.5 Arranging partials

In preparation for the next stage, PTMP arranges $\mathbf{P}$ in order of decreasing energy, which we approximate as the sum of the partial's amplitude vector, $\sum_m a_\rho[m]$. After the arrangement, $\mathbf{P}_1$ is the highest energy partial. Figure 4.5 shows trajectories that the partial tracker extracted from a clarinet audio sample.

4.2 Partial to atom

In this section, we describe how to use the data from a partial trajectory to create an asymmetric atom. Given the partial data from the previous step, the following process determines values for the atom's entire parameter set. As a result, this process outputs a fully determined large scale asymmetric atom (REDS), $\phi(\boldsymbol{\lambda}_k)$, where the REDS parameter set $\boldsymbol{\lambda}^{(k)} = \{N, f_c, \alpha, \beta, p, \tau\}^{(k)}$.

Retrieving an atom from a partial trajectory involves one of two approaches: the first approach for the case when the trajectory spans more than some number of frames M_{min}, and the other for when it spans less than M_{min} frames.

4.2.1 Frequency

To start, we calculate the atom's normalized frequency f_c as the weighted average of the partial trajectory frequencies $\omega_\rho[m]$ and amplitudes $a_\rho[m]$:

$$f_c = \frac{1}{2\pi} \frac{\sum_m \omega_\rho[m] a_\rho[m]}{\sum_m a_\rho[m]} \tag{4.13}$$

4.2.2 Damping

If $M_\rho > M_{min}$, we assume that some of $\mathbf{P}_\rho$ reflects a freely decaying sinusoid. We locate the section of $\mathbf{P}_\rho$ that resembles a freely decaying sinusoid and use the damping factors within that section to estimate the atom's damping factor α. The section of $\mathbf{P}_\rho$ where the damping factor is positive corresponds to the decaying part. Let $m^\dagger$ denote the indices of damping factor $\alpha_\rho[m]$ such that $\alpha_\rho[m^\dagger] > 0$ is true. We define the atom's damping factor α as the weighted average of amplitudes $a_\rho[m^\dagger]$ and $\alpha_\rho[m^\dagger]$:

$$\alpha = \frac{\sum_{m^\dagger} \alpha_\rho[m^\dagger] a_\rho[m^\dagger]}{\sum_{m^\dagger} a_\rho[m^\dagger]} \tag{4.14}$$

We avoid skewing the estimate of α by not weighing in the frames that have negative or zero values of $\boldsymbol{\alpha}_\rho$. Alternatively, we could estimate α through least-squares fitting of the partial's amplitude evolution $\boldsymbol{a}_\rho$ to a damped exponential curve. However, we choose to implement the weighted average approach (4.14) because it is simple and provides excellent results.

For $M \leq M_{min}$, the partial has too few data points for a robust estimate of α, so we discard $\boldsymbol{\alpha}_\rho$.

4.2.3 Onset and duration

We seek refined values of the atom's onset τ and duration N for a few reasons. The first reason is that the precision of $\hat{\tau}_\rho[1]$ and $\hat{\tau}_\rho[M_\rho]$ are dependent on the STFT hop size H. Second, we seek to extend the ends and beginnings of the partials that we split (see 4.1.4), because we assume that they are superimposed. Finally, recall from Section 4.1.2 that we reassign the time location of each peak *and* crop the peaks that are off-center. After time reassignment, a partial's start and end times are relatively close to the signal component's

and thus reduces the likelihood of pre and post-echo creation. Then, cropping eliminates peaks relatively close to the signal component's onset and termination, such that $\hat{\tau}_\rho[1]$ and $\hat{\tau}_\rho[M_\rho]$ are both safely inside the time span of the signal component, as shown in Figure 4.2. Therefore, since the oscillation responsible for the partial trajectory $\mathbf{P}_\rho$ starts before $\hat{\tau}_\rho[1]$ and ends after $\hat{\tau}_\rho[M_\rho]$, we must find more accurate estimates of the atom's onset time and duration.

If $M_\rho > M_{min}$, we create a damped sinusoid atom of length $N_\rho = \hat{\tau}_\rho[M_\rho] - \hat{\tau}_\rho[1] + 1$, $\boldsymbol{\phi}_\rho$, with α and f_c, and project it onto the residual to retrieve complex gain $\hat{x}_\rho$

$$\hat{x}_\rho = \langle \mathbf{r}_{\hat{\tau}_\rho[1]}, \boldsymbol{\phi}_\rho \rangle \tag{4.15}$$

where $\mathbf{r}_{\delta_0} \mid r_{\delta_0}[n] = r[n + \delta_0]$.

We use the recursive inner product (RIP) algorithm to refine the atom's onset and duration values, see Algorithm 3. We assume that most of the atom's energy is within the time span $(\hat{\tau}_\rho[1], \hat{\tau}_\rho[M_\rho])$, so we use a real-valued version of RIP to hold the phase of the atom constant with respect to $\angle\hat{x}$. Since we must input the phase of the atom into the real-valued version of RIP, we retrieve the phase of the damped sinusoid atom at the first and last sample, $\theta[1] = \angle\hat{x}_\rho\phi_\rho[1]$ and $\theta[N_\rho] = \angle\hat{x}_\rho\phi_\rho[N_\rho]$, respectively. Then we run the algorithm twice, once starting from $\hat{\tau}_\rho[M_\rho]$ and iterating in the direction of positive time to retrieve a new estimate of the atom's end, and once starting from $\hat{\tau}_\rho[1]$ and iterating in the direction of negative time to retrieve a new estimate of the atom's onset.

If $M_\rho \leq M_{min}$, our procedure is the same except that we extend a complex sinusoid of frequency f_c *without* amplitude modulation since, at this stage, we do not have an estimate of α.

Recursive Inner Product

Goodwin proposed the use of recursion to calculate the cross-correlation between a dictionary of damped sinusoid atoms and $\mathbf{y}$ as an alternative to a direct computation [31]. We establish a recursive algorithm whose purpose is to refine an atom's parameters rather than to calculate dictionary correlations. Our RIP algorithm computes the inner product of the residual $\mathbf{r}$ and a damped exponential atom $\boldsymbol{\phi}_\rho$ on a sample-by-sample basis to extend the length of the atom either forwards or backwards in time and thereby retrieve better estimates of the atom's τ and N.

The algorithm starts at some sample δ_0 of the residual, $\mathrm{r}[\delta_0]$, and extends the atom either forwards or backwards until the inner product between the atom and residual does not increase from one iteration to the next. The output of the algorithm is a refined estimate of the atom's end time, or start time, depending on whether the trace direction is forward or backward, respectively. Recall that $\boldsymbol{\phi}_\rho$ must be normalized such that $\|\boldsymbol{\phi}_\rho\|_2^2 = 1$, see Section 2.2. Since the atom's length changes at each iteration of the algorithm, we incorporate an update mechanism in the algorithm to ensure that the atom's euclidean norm is always one.

Before iterating, the algorithm initializes the atom's euclidean norm, $\eta^{(0)} = \|\boldsymbol{\phi}_\rho\|_2 \in \mathbb{R}^+$, and inner product, $g^{(0)} = \langle \mathbf{r}_{\delta_0}, \boldsymbol{\phi}_\rho \rangle \in \mathbb{C}$. For each iteration, it first calculates the current sample of the complex damped sinusoid $\phi_\rho[\delta_0 + k]$, where k is the iteration number and δ_0 is the start sample, as the output of a complex one-pole filter:

$$\phi_\rho[\delta_0 + k + 1] = \phi_\rho[\delta_0 + k] e^{-\alpha + i2\pi f_c} \tag{4.16}$$

Given the new sample, it updates the atom's euclidean norm,

$$\eta^{(k+1)} = \left(\left(\eta^{(k)}\right)^2 + |\phi_\rho[\delta_0 + k + 1]|^2 \right)^{\frac{1}{2}} \tag{4.17}$$

then updates the inner product of the atom and residual,

$$g^{(k+1)} = \frac{g^{(k)}\eta^{(k)} + \mathrm{r}[\delta_0 + k + 1]\bar{\phi}_\rho[\delta_0 + k + 1]}{\eta^{(k+1)}} \tag{4.18}$$

The algorithm runs until $|g^{(k+1)}| \not> |g^{(k)}|$, so the iteration number that corresponds to the largest inner product is k. The inner product may only decrease momentarily and then continue to increase after some number of samples. This usually happens in the presence of a superimposed transient because it adds rapid fluctuations and distorts the oscillation that the algorithm is tracing. To bridge over the fluctuations, we specify an overshoot value Ξ, which is the number of samples, ξ, to search past k after $|g^{(k+1)}| \not> |g^{(k)}|$ is true. After ξ exceeds Ξ, the algorithm ends. If $|g^{(k+\xi)}| > |g^{(k)}|$, it sets $k = k + \xi$, $\xi = 0$, and continues as normal.

We use real values for the inner product and norm update when we have already committed the atom's phase. In this case, we initialize $\phi_\rho = e^{i\theta[\delta_0]}$, where $\theta[\delta_0]$ is the phase of

Algorithm 3 Recursive Inner Product

init: $\phi = 1$, $\eta = \|\boldsymbol{\phi}_\rho\|_2$, $g = \langle \mathbf{r}_{\delta_0}, \boldsymbol{\phi}_\rho \rangle$
repeat
  $\phi \leftarrow \phi e^{-\alpha + i\omega_c}$
  $g \leftarrow g\eta + \mathrm{r}[\delta_0 + k + 1]\bar{\phi}$
  $\eta \leftarrow \sqrt{\eta^2 + |\phi|^2}$
  $g \leftarrow g\eta^{-1}$
  $k \leftarrow k + 1$
until stopping condition
return k, g, η

atom $\boldsymbol{\phi}_\rho$ at sample δ_0. We convert ϕ_ρ to its real-valued counterpart before updating $g^{(k+1)}$,

$$g^{(k+1)} = \frac{g^{(k)}\eta^{(k)} + 2\Re\{\phi_\rho[\delta_0 + k + 1]\}\mathrm{r}[\delta_0 + k + 1]}{\eta^{(k+1)}} \tag{4.19}$$

and $\|\boldsymbol{\phi}_\rho\|_2$,

$$\eta^{(k+1)} = \left(\left(\eta^{(k)}\right)^2 + \left|2\Re\{\phi_\rho[\delta_0 + k + 1]\}\right|^2\right)^{1/2} \tag{4.20}$$

Real values ensure that the extended section's phase is coherent with the existing atom's phase.

4.2.4 Envelope

For $M_\rho > M_{min}$, the final step retrieves the atom's attack shape and onset time. At this stage, we extracted a damped sinusoid atom whose damping, frequency, and phase estimates are reliable. Moreover we have a good estimate of the end time for the atom. In the previous step, we recursively computed the inner product backwards with a real sinusoid to refine the onset time value, $\hat{\tau}_\rho[1]$. At this point $\hat{\tau}_\rho[1]$ is only accurate if the signal contains a pure damped sinusoid with no attack shape, which is almost never the situation, thus we search for a better estimate of the onset time before committing it as τ within $\boldsymbol{\lambda}$.

In fact, retrieving an accurate estimate of τ is dependent on an accurate estimate of β, and vice-versa. In other words, the quality of their estimations are mutually dependent. Suppose a synthetic signal contains a REDS atom whose attack envelope's maximum is n_m samples after its onset time τ, and that our current estimate of onset time $\hat{\tau}_\rho[1]$ is some

number of samples after the true value of τ, for example $\frac{2}{3}n_m$. Estimating β in this scenario leads to an atom whose envelope maximum occurs $\frac{1}{3}n_m$ after its onset time, to match with the signal. In other words, the estimated attack time is much smaller (skirt width is much wider) to compensate for the error in onset estimation, see Figure 4.6a. Likewise, adapting the onset time with a fixed β results in a sub-optimal approximation that is either before or after the true value of τ, so it might cause pre-echo, see Figure 4.6b. Therefore, to retrieve an accurate attack envelope we estimate β and τ simultaneously.

A Newton step performs the estimation of β and τ. Starting from a real-valued damped sinusoid atom,

$$\boldsymbol{\phi}_{DS}(\tau^{(0)}, \theta) = e^{-\alpha(n-\tau^{(0)})} \cos(\omega_c(n - \tau^{(0)}) + \theta) \tag{4.21}$$

whose phase θ is from the previous step, Newton's method searches for a real-valued REDS atom, $\boldsymbol{\phi}_{REDS}(\beta, \tau, \theta) = A(\beta, \tau)[n]\phi_{DS}(\tau, \theta)[n]$, where

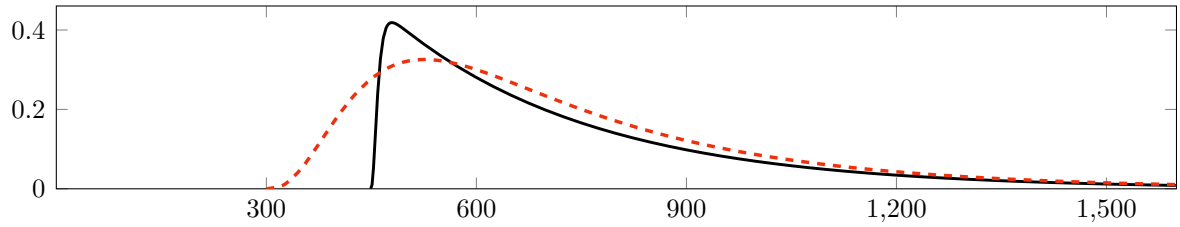
$$A(\beta, \tau)[n] = \left(1 - e^{-\beta(n-\tau)}\right)^p \, u[n - \tau] \tag{4.22}$$

Newton's method's reliability depends on how close the initial estimate $\hat{\beta}^{(0)}$ is to ground-truth. We find a coarse estimate for $\hat{\beta}^{(0)}$ by modulating $\boldsymbol{\phi}_{DS}$ with a set of A_{REDS} to create a sub-dictionary of REDS atoms, calculating the inner products of $\mathbf{r}$ and the sub-dictionary, and setting $\hat{\beta}^{(0)}$ to match the β value of the atom responsible for the largest inner product. In practice, since the sole difference between the atoms in the sub-dictionary are the attack envelopes, the inner product only needs to consider the samples spanning n_I.

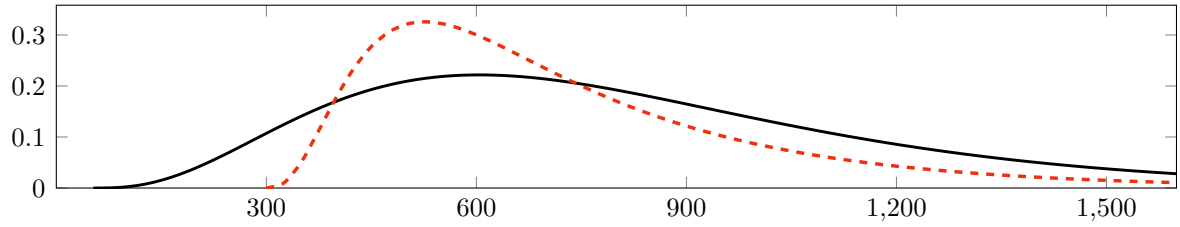
Refining τ involves a multidimensional search over the τ and θ space. Even though we keep the initial θ during the Newton refinement, we must incorporate the phase space into the time space so that the atom can shift in time[1]. A Newton search over only the one-dimensional τ space does not work. Conceptually, the phase blocks the atom's movement through time, like a mountain that the atom cannot pass over. Thus, we calculate the Hessian matrix and gradient vector of $\boldsymbol{\phi}_{REDS}(\beta, \tau, \theta)$ with respect to θ and τ to perform a multidimensional Newton step. See Appendix A for the derivations. Since the dimension of the search involves the phase space as well, the Newton step finds a correct phase shift to match the time shift.

After retrieving a new value of τ, we update $\boldsymbol{\phi}_{REDS}(\beta, \tau, \theta)$ then estimate β via a

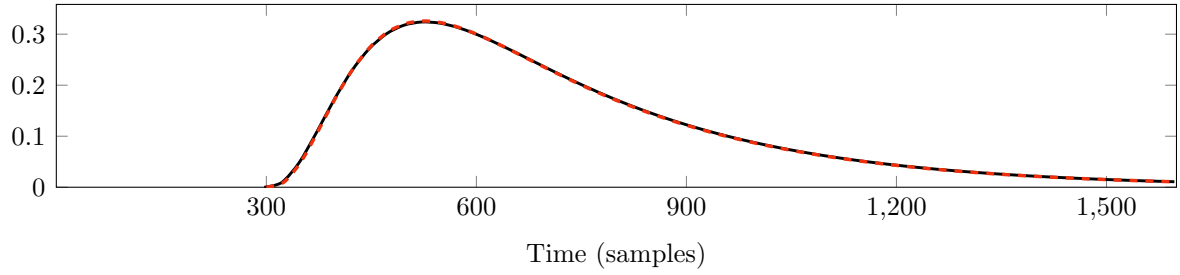
[1]If the atom is complex, the Newton step must refine the argument of the atom's complex gain, $\angle x$, because it describes the atom's phase.



(a) Adapting β with a constant τ.



(b) Adapting τ with a constant β



(c) Simultaneous β and τ adaptation.

Figure 4.6 The envelope of a REDS atom (black) that Newton's method estimated from the ground-truth envelope (dashed, red). The initial value of the REDS atom's α and ω_c were equal to ground-truth, then Newton's method estimated τ and β.

one-dimensional Newton step. Through trials we found that a three-dimensional Newton step to estimate $\beta, \tau, \text{and}\, \theta$ is not stable. We iterate over two Newton steps, for τ then β, until the residual energy from one iteration to the next does not decrease (i.e., when $\|\mathbf{r}^{(k+1)}\|_2^2 \not< \|\mathbf{r}^{(k)}\|_2^2$), see Algorithm 4 and Figure 4.6c.

If $M_\rho \leq M_{min}$, since we could not reliably convert the partial trajectory's damping factors into an approximation of α, the final step retrieves an estimate of β, τ, and α. For this, we incorporate the estimation of α into the first Newton step so that it searches the τ, ϕ, and α multidimensional space. We accommodate the atom's changing envelope

Algorithm 4 REDS attack envelope and onset estimation

init: $k = 0$, $\tau^{(0)} = \hat{\tau}_\rho[1]$, $\beta^{(0)} = \hat{\beta}$
repeat
  $\boldsymbol{\mu} = [\tau^{(k)}, \theta]^\intercal$
  $\boldsymbol{\mu} = \boldsymbol{\mu} - \left(\mathbf{H}_{\boldsymbol{\mu}} \boldsymbol{\phi}(\tau^{(k)}, \beta^{(k)}, \theta)\right)^{-1} \nabla_{\boldsymbol{\mu}} \boldsymbol{\phi}(\tau^{(k)}, \beta^{(k)}, \theta)$
  $\tau^{(k+1)} = \mu[0]$
  $\beta^{(k+1)} = \beta^{(k)} - \frac{\frac{\partial}{\partial \beta} \boldsymbol{\phi}(\tau^{(k+1)}, \beta^{(k)}, \theta)}{\frac{\partial^2}{\partial \beta^2} \boldsymbol{\phi}(\tau^{(k+1)}, \beta^{(k)}, \theta)}$
  $k = k + 1$
until stopping condition

from one iteration to the next by expanding or contracting its length N such that $\phi(N)$ is small enough to negate the discontinuity's perceptual relevance, for example, such that $\frac{\phi(N)}{\phi(n_m)} \approx .001$.

4.2.5 Decay termination

Generally, for atoms with long decay rates, the recursive inner product stops before the damped sinusoid part decays to an inaudible value, which introduces a discontinuity at the atom's end. We consider three possible options to remedy the situation at the atom's end.

As one option, we could assume that the damped sinusoid part must decay freely "forever". Under this assumption, we extend the atom to reach a value of -60 dB, however, in doing so, we create dark energy: arbitrarily extending the atom past its optimal point introduces energy into the residual signal for which later iteration steps must approximate. For example, a piano produces freely decaying oscillations that decay indefinitely if the pianist is stepping down on the sustain pedal, however, if the pianist releases the sustain pedal the vibrations abruptly terminate.

A second option: after tracing the atom forward, if the end of the atom is above some threshold (like -60 dB), then we increase α (increase the decay rate) such that the end, $\phi(N)$ is below the threshold. We avoid post-echo and the need for a taper envelope. The downside is the non-optimal α value: since we are not using the best α, we need multiple similar atoms around its location to make up for the amplitude difference.

The last option involves the application of a taper to the end of the atom so that it smoothly decays to a zero value. The taper is a time-reversed REDS attack envelope, $A_{REDS}[N - 1 - n + \tau]$. The downside of this approach is that the taper exhausts another

parameter. For this thesis, we choose the third option, to taper the end at the cost of an extra parameter, because it allows us to fit the atom to the signal more closely than the other two options and avoids the creation of dark energy.

4.3 Small-scale pursuit

A dictionary of small-scale atoms represents transients and other highly non-stationary signal content sufficiently. Several fast numerical computation techniques keep the small-scale sub-system speedy. Refinement of the fixed dictionary's coarse discretization grid results in a sparser solution per iteration and allows for a compact dictionary size that relieves the sub-system's computational burden.

4.3.1 Discretization and storage

The small-scale dictionary's discretization scheme is as follows. We select a discrete duration set, $N_s = 2^s$ where $s \in [s_{min}, s_{max}]$. We assume that the partial tracking step sufficiently represents atoms greater than or equal to N_{PT}, so we set $N_{s_{max}}$ to be less than N_{PT}. $N_{s_{min}}$ should be small to ensure that some of the atoms can represent highly non-stationary signal content, for example $N_{s_{min}} \leq 64$. For each duration, we construct a block of REDS atoms that share the same gammatone-like amplitude envelope and span a range of frequencies, κ_s,

$$\mathbf{\Phi}_s = \phi_s[n, \kappa_s] = e^{n(-\alpha_s + i2\pi\kappa_s/N_s)}(1 - e^{-\beta_s n})^p u[n] \tag{4.23}$$

where $\kappa_s = (2, 3, \ldots, \frac{N_s}{2} - 2)$, $\alpha_s = \frac{15}{N_s}$ to negate discontinuity artifacts at the atom's end, $\beta = \frac{\alpha_s 10^{-3}}{p^2}$, and $p = 3$.

$\mathbf{\Phi}_s$'s storage is spread throughout multiple sub-dictionaries, one for each unique value of N_s. By compartmentalizing $\mathbf{\Phi}_s$ per atom length, we reduce the computer memory requirements and make the algorithm faster by selecting, per atom length, a fast method of inner product computation. MPTK also uses sub-dictionaries that it calls "blocks": each block is a windowed DFT matrix [10], so it constructs one block for each unique amplitude envelope $E[n]$.

4.3.2 Cross-correlation computation and update

Computing cross-correlations of $\mathbf{y}$ and an overcomplete $\mathbf{\Phi}$ is one of the main computational hurtles of MP. We choose one of two methods to compute the cross-correlation of a sub-dictionary and signal, depending on the difference between the signal's length and the atom's length. The first is direct convolution, more precisely, convolution per the mathematical definition. Direct convolution is the fastest method when the difference between signal length and atom length is greater than some value. For example, if the signal length is 2^{14} and the atom's length is 2^6, direct convolution is the fastest method. When the signal and atom length's are closer to one another, for example if the signal length is 2^{13} and the atom's length is 2^{10}, the fastest method of convolution computation is via the FFT [58], which employs the convolution property of the Fourier transform, $\mathbf{y} * \boldsymbol{\phi} = \mathcal{F}[\mathbf{y}] \odot \mathcal{F}[\boldsymbol{\phi}]$. Thus, we take the zero-padded FFT of the signal and of the time-reverse complex conjugate of each atom $\mathbf{\Phi}_s$, $\bar{\boldsymbol{\phi}}_s[-n]$, multiply their DFT's together, then take the IFFT to retrieve the cross-correlation. Lastly, when the signal length is very large in comparison to the dictionary atom lengths we may use the fast FFT convolution on segments of the signal and overlap-add the results, however, computing one FFT for the entire signal is usually faster.

To initialize the small-scale sub-system, we calculate and store the cross-correlation between the audio signal and dictionary $\mathbf{y} \star \mathbf{\Phi}_s$, and the dictionary's zero-padded DFT. We gain some computational efficiency by avoiding the calculation of the dictionary FFT at every step. If we were to only use atoms inside the dictionary (no refinement or atoms from another sub-system) we could compute and store the dictionary's cross-correlations, although storing these huge multi-dimensional arrays requires a considerable amount of memory allocation and is sometimes not even possible. In fact, most implementations compute the inner products at each iteration [10] [34]. Calculating the cross-correlations via an atom's analytic formulas is another possibility, though the method we adopt is applicable to an arbitrary atom type and is considerably simpler to implement.

The best atom is the one that is responsible for the absolute maximum of all the cross-correlations. Note that the index of the cross-correlation locates the atom in time with respect to the signal, τ.

4.3.3 Parameter refinements

Searching inside of a pseudo-continuous parameter space after locating the best choice from the coarse dictionary typically results in a sparser solution. Starting with the best atom from $\mathbf{\Phi}_s$, we refine its frequency with RRM, see Section 2.4.2. Then we jointly search for refined values of α and β with a multidimensional Newton algorithm that requires the REDS atom's first and second partial derivatives. Appendix A includes these equations. Note that the length of the atom has to increase to accommodate decreases to α.

[31] showed that if atoms have equal angular spacing on circles in the z-plane, we can use the DFT (FFT) to compute the inner products between the atoms and audio signal over ω_c. MPTK [10] uses this approach to calculate dictionary inner products over a coarse time grid with an STFT. As an alternative to computing a cross-correlation between each atom in a dictionary and the signal, we propose the use of an STFT to efficiently compute the dictionary inner products over the coarse time grid, then refine the temporal location of the best-correlated atom with a separate estimation technique, for example, Newton's method. We assume that the speed improvements of this alternative method are negligible because we likely require several iterations of the computation-heavy Newton step to shift the atom to a better temporal location. MPTK [10] employs the STFT approach without refining temporal location. To compute a cross-correlation in MPTK, one sets the STFT's hop size equal to one. Switching to either of the faster cross-correlation methods that we described in Section 4.3.2 is beneficial.

4.4 Algorithm

PTMP retains the MP-based local optimization inner product selection criteria within its hybrid structure. It is hybrid because it involves two sub-systems that employ separate atom search and creation techniques. At each iteration k, PTMP estimates a REDS parameter set $\boldsymbol{\lambda} = \{N, f_c, \alpha, \beta, \tau\}$ that results in the largest inner product with residual, $\boldsymbol{\lambda}^{(k)} = \arg\max_{\lambda} |\boldsymbol{\phi}_{\boldsymbol{\lambda}}^{\mathrm{H}} \mathbf{r}^{(k)}|$ (i.e., that minimizes the residual energy $\|\mathbf{r}\|_2^2$). Either the large-scale or small-scale sub-system, $\gamma = \mathrm{L}$ or S, respectively, locates $\boldsymbol{\lambda}^{(k)}$ depending on which inner product coefficient, x_{S} or x_{L}, is greater in magnitude. PTMP runs as follows (see flowchart in Figure 4.7):

1. Initialization:

   i) Compute the signal and small-scale dictionary cross-correlation, $\mathbf{X}_s = \mathbf{y} \star \mathbf{\Phi}_s$. Find the best small-scale atom and store its parameter set $\boldsymbol{\lambda}_{\mathrm{S}}$ and coefficient x_{S}.

   ii) Extract partial trajectories from $\mathbf{y}$, $\mathbf{P}_\rho$, with the partial tracker and sort them based on energy. Set the partial trajectory index to one, $\rho = 1$. Create an atom from highest energy partial, then store its parameter set $\boldsymbol{\lambda}_{\mathrm{L}}$ and coefficient x_{L}.

2. Sub-system selection:

   i) Locate the potential best option, $\mathrm{O}^{(k)} = \arg\max_\gamma |x_\gamma|$.

   ii) If $\mathrm{O}^{(k)} = \mathrm{O}^{(k-1)}$, go to Step 3.

   iii) If $\mathrm{O}^{(k)} \neq \mathrm{O}^{(k-1)}$, there is potential to switch from sub-system $\mathrm{O}^{(k-1)}$ to $\mathrm{O}^{(k)}$. To be sure, extract and sort new partial trajectories from $\mathbf{r}^{(k-1)}$, $\mathbf{P}_\rho$, reset $\rho = 1$, retrieve a new $\boldsymbol{\lambda}_{\mathrm{L}}$ and coefficient x_{L}, then update $\mathrm{O}^{(k)} = \arg\max_\gamma |x_\gamma|$. Then, if $\mathrm{O}^{(k)} = \mathrm{S}$, update $\mathbf{X}_s$ per $\mathbf{r}^{(k-1)}$, retrieve a new $\boldsymbol{\lambda}_{\mathrm{S}}$ and coefficient x_{S}, then update $\mathrm{O}^{(k)} = \arg\max_\gamma |x_\gamma|$.

3. Residual update

   i) Update the residual $\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - 2\Re\{x_{\mathrm{O}^{(k)}} \phi(\boldsymbol{\lambda}_{\mathrm{O}^{(k)}})\}$ and commit the parameter set and coefficient to memory: $\boldsymbol{\lambda}^{(k)} = \boldsymbol{\lambda}_{\mathrm{O}^{(k)}}$ and $x^{(k)} = x_{\mathrm{O}^{(k)}}$.

   ii) If $\mathrm{O}^{(k)} = \mathrm{S}$, update the part of $\mathbf{X}_s$ that $\phi(\boldsymbol{\lambda}_{\mathrm{O}^{(k)}})$ overlaps with in time. Find the best small-scale atom and store its parameter set $\boldsymbol{\lambda}_{\mathrm{S}}$ and coefficient x_{S}.

   iii) If $\mathrm{O}^{(k)} = \mathrm{L}$, increment the partial trajectory index $\rho = \rho + 1$ to point to the next highest energy partial. If ρ is above the number of partials in $\mathbf{P}$, R, extract and sort new partial trajectories from $\mathbf{r}^{(k)}$, $\mathbf{P}_\rho$, and reset $\rho = 1$. Retrieve a new $\boldsymbol{\lambda}_{\mathrm{L}}$ and coefficient x_{L} from $\mathbf{P}_\rho$.

   iv) Increment the iteration number, $k = k+1$. Terminate if k is above the maximum iterations, K, or if $\|\mathbf{r}\|_2^2$ is below some threshold, ϵ. Otherwise, return to Step 2.

PTMP exploits the fact that residual energy monotonically decreases in an MP-based algorithm, formally $\|\mathbf{r}^{(k)}\|_2^2 < \|\mathbf{r}^{(k-1)}\|_2^2$. Thus, independent of $\boldsymbol{\lambda}$, $|\phi(\boldsymbol{\lambda})^{\mathrm{H}}\mathbf{r}^{(k)}| \leq |\phi(\boldsymbol{\lambda})^{\mathrm{H}}\mathbf{r}^{(k-1)}|$ (they are equal when the residual update is not in the region of $\phi(\boldsymbol{\lambda})$). Notice that in Step

3, PTMP only updates one sub-system's values, $\boldsymbol{\lambda}_{\mathrm{O}^{(k)}}$ and $x_{\mathrm{O}^{(k)}}$. Upon return to Step 2, it compares the updated coefficient to the non-updated coefficient. It confidently does this because the non-updated coefficient is a best-case scenario: had the coefficient updated at the end of Step 3, it would have either decreased or remained the same. Thus, it skips to Step 3 when $\mathrm{O}^{(k)} = \mathrm{O}^{(k-1)}$ (i.e., it remains within the same sub-system while this is true). Finally, when $\mathrm{O}^{(k)} \neq \mathrm{O}^{(k-1)}$, PTMP performs an update to the sub-system in need. The update could decrease the updated coefficient such that it is no longer the maximum, so it re-checks $\arg\max_\gamma |x_\gamma|$. By this logic, PTMP manages a search for the best parameter set with a minimum number of updates per iteration.

4.5 Summary

This chapter established PTMP, an MP-based system that locates REDS atoms via the combination of two separate techniques, from partial trajectories and from the comparison of a static dictionary of small-scale REDS atoms with the audio signal, and bridges the search by comparing each side's results to remain within an MP framework. While the search method is applicable to most types of atoms, whether asymmetric or symmetric, we established methods to transform partial trajectory data into asymmetric atoms, more specifically, REDS, and refine their parameters using a variety of estimation techniques. We detailed how to refine an atom's onset and duration values through a recursive inner product algorithm, then jointly estimate the REDS attack envelope and onset time with Newton's method in multiple dimensions. In the next chapter, we perform experiments that gauge PTMP's performance and report the results.

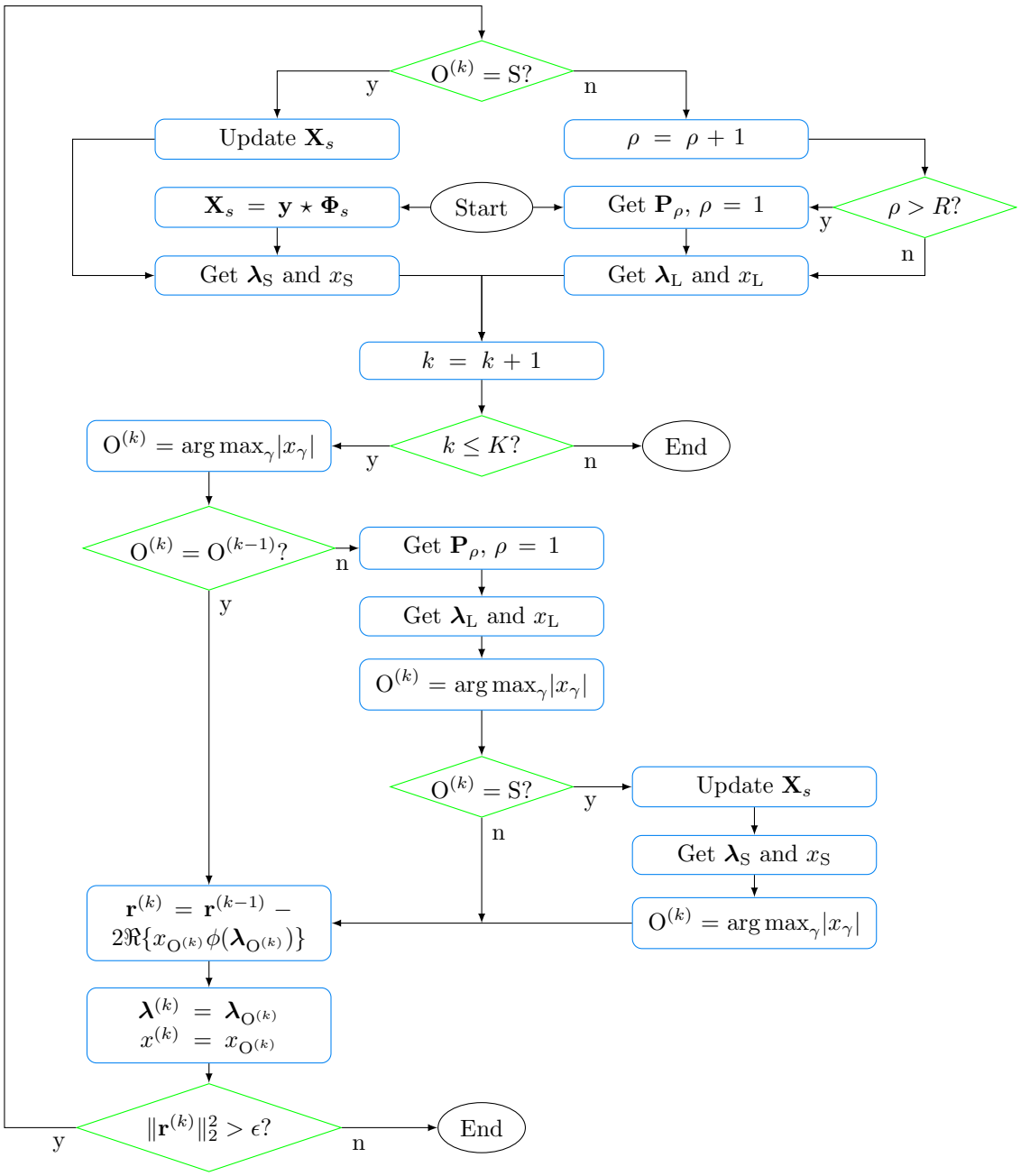


Figure 4.7 Partial tracking matching pursuit (PTMP) flowchart, see Section 4.4 for details.

Chapter 5

Experiments

This chapter describes a series of experiments that test the performance of partial tracking matching pursuit (PTMP) and its estimators. The first group of experiments test how well Newton's method estimates REDS parameters. Then we document PTMP's ability to decompose a range of audio signals, starting with a set of synthetic audio then moving to real musical audio. The synthetic audio set represent a variety of time-frequency behaviors, varying from content that matches well with an asymmetric atom model to content that does not. In the last group of decomposition experiments, we test PTMP's ability to decompose real musical signals, including excerpts of single instrument and multi-instrumental musical pieces. Finally, we experiment with different ways to manipulate the parameter sets of the REDS atoms that PTMP extracted from real audio signals, and report on the resulting sounds. A supplementary website contains audio files of the test signals for this chapter along with sounds that we synthesized after post-processing REDS parameters sets per Section 5.5. The URL for this website is: `http://www.music.mcgill.ca/~julian/thesis`.

5.1 Estimators

This section details experiments that gauge Newton's method's ability to estimate a REDS atom in two situations. The first situation is when we have determined α and f_c and seek values for β and τ (i.e., when $M_\rho > M_{min}$). The second set reflects the situation when we know f_c and need values for β and α and τ (i.e., when $M_\rho \leq M_{min}$).

Let λ be some variable of the synthetic audio signal $\mathbf{y}$ that we want Newton's method to estimate. $\hat{\lambda}^{(0)}$ is an initial estimate of λ that we use to initialize Newton's method and

$\delta_\lambda^{(0)}$ is the difference between the ground-truth value and the initial estimate, $\delta_\lambda^{(0)} = \lambda - \hat{\lambda}^{(0)}$.

We setup the experiment by synthesizing an audio signal $\mathbf{y}$ from one REDS, time shifted by τ with a decay time of α^{-1}, whose attack influences time n_I is such that part of the atom reflects a freely decaying sinusoid. Then, we create a range of differences between the initial estimate and ground-truth for attack influence time $\boldsymbol{\delta}_{n_I}^{(0)} \in [-50, 250]$, time shift $\boldsymbol{\delta}_\tau^{(0)} \in [-250, 250]$, and decay time $\boldsymbol{\delta}_{\alpha^{-1}}^{(0)} \in [-250, 400]$.

One run of an experiment involves the following steps:

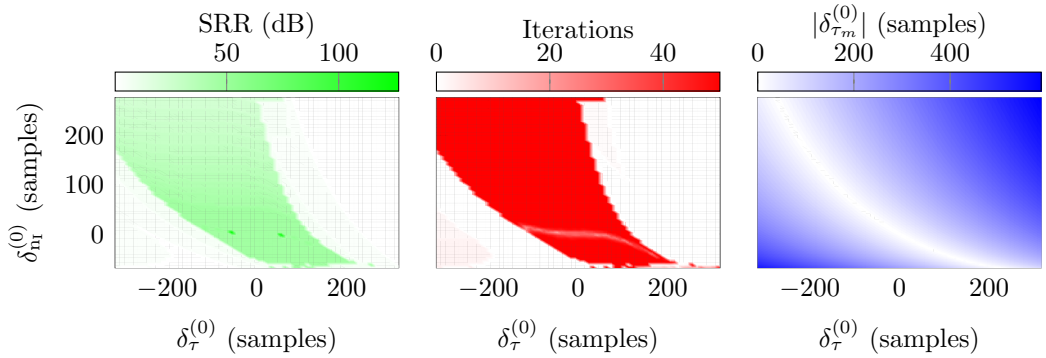
1. Initialize Newton's method with some combination of two initial estimates.
2. Iterate until the number of iterations, k, reaches 50, or until $\text{SRR}^{(k+1)} \not> \text{SRR}^{(k)}$.
3. Record the number of iterations and the SRR at the final iteration. Record the difference, $\delta_{\tau_m}^{(0)}$, between the time location of $\mathbf{y}$'s maximum, τ_m, and the initial time location of the atom's maximum, $\hat{\tau}_m^{(0)}$.

For the experiment where we seek values for the attack shape and onset time (i.e., when $M_\rho > M_{min}$), we complete the aforementioned steps for every combination of $\boldsymbol{\delta}_{n_I}^{(0)}$ and $\boldsymbol{\delta}_\tau^{(0)}$, where $\delta_{\alpha^{-1}}^{(0)} = 0$, see Figure 5.1a. For the second experiment, where we seek values for β and α and τ (i.e., when $M_\rho \leq M_{min}$), we complete the aforementioned steps for every combination of $\boldsymbol{\delta}_{n_I}^{(0)}$ and $\boldsymbol{\delta}_{\alpha^{-1}}^{(0)}$, where $\delta_\tau^{(0)} = 0$, see Figure 5.1b.

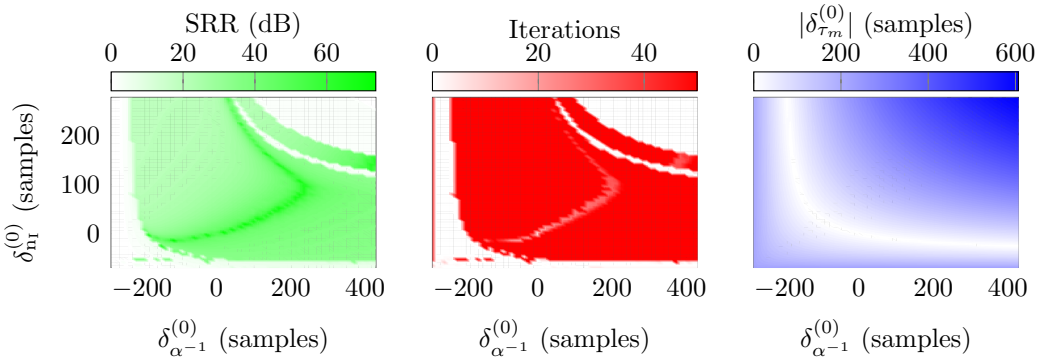
Results from the first experiment verify that the algorithm reaches a high SRR for certain combinations of initial values $\hat{n}_I^{(0)}$ and $\hat{\tau}^{(0)}$, which is the green area that follows a diagonal trend in the SRR plot of Figure 5.1a. Outside of this region, Newton's method does not work (the white sections in either the SRR or Iterations plot). We deduce that Newton's method works (it improves upon the initial parameter estimates) when the combination of initial values result in a relatively small $|\delta_{\tau_m}^{(0)}|$. In other words, the performance of Newton's method for estimating the envelope of a REDS atom depends mainly on the difference between the time location of the signal's amplitude maximum τ_m, and the initial time location of the atom's amplitude maximum, $\hat{\tau}_m^{(0)}$. Thus, it is important to locate the envelope peak in time of the audio signal[1].

Similar results emerge from the second experiment, where we seek values for α and β. Results in Figure 5.1b show that Newton's method iterates when the combination of $\hat{n}_I^{(0)}$

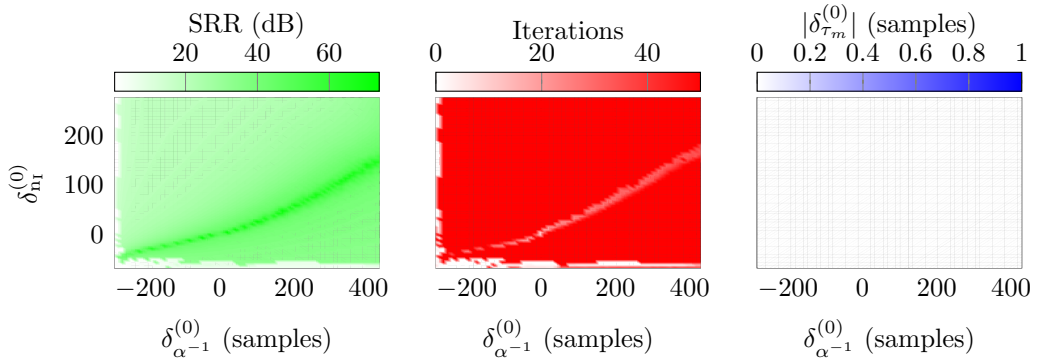
[1]For a real signal $\mathbf{y}$, one method to retrieve the envelope is to bandpass filter $\mathbf{y}$ at f_c then apply the Hilbert transform to retrieve an approximate analytic signal whose absolute value is the approximate amplitude envelope.



(a) Adapt β and τ, where α is equal to ground-truth.



(b) Adapt β and α, where τ is equal to ground-truth.



(c) Adapt β and α, where $|\delta^{(0)}_{\tau_m}| = 0$.

Figure 5.1 Results from the experiments involving Newton's method estimation of REDS parameters.

and $\hat{\alpha}^{-1(0)}$ is such that $|\delta^{(0)}_{\tau_m}|$ is relatively small, see the correlation between the non-white parts of the SRR and Iterations plot, and the parts of the $\delta^{(0)}_{\tau_m}$ plot where $|\delta^{(0)}_{\tau_m}|$ is relatively small. We note that the non-white sections of the SRR and Iterations plot correspond to where $\delta^{(0)}_{\alpha^{-1}}$ and $\delta^{(0)}_{n_I}$ are positive. Therefore, an overestimate of the attack and decay durations is generally better than an underestimate. We think this is the case because large scale atoms have more significant samples to compare with the signal than smaller scale atoms. Overall, a good initial estimate will have its maximum centered around the true maximum and be leaning to the larger scale rather than smaller scale. Overall, it is better to initialize Newton's method with a combination of initial estimates that result in a relatively small $|\delta^{(0)}_{\tau_m}|$ value and an atom whose scale is larger, rather than smaller, than the scale of $\mathbf{y}$.

As reinforcement for the previous statements, the last experiment shows the benefits of initializing Newton's method such that $|\delta^{(0)}_{\tau_m}|$ is relatively small. As in the second experiment, we complete a run for every combination of $\boldsymbol{\delta}^{(0)}_{n_I}$ and $\boldsymbol{\delta}^{(0)}_{\alpha^{-1}}$. However, instead of setting $\delta^{(0)}_{\tau} = 0$, we set a value for $\hat{\tau}^{(0)}$ that is dependent on the combination of $\hat{n}^{(0)}_I$ and $\hat{\alpha}^{-1(0)}$ such that $\delta^{(0)}_{\tau_m} = 0$. More precisely, since $\tau_m = \tau + n_m$, we set $\hat{\tau}^{(0)} = \tau_m - \frac{1}{\hat{\beta}^{(0)}} \log(1 + \frac{p\hat{\beta}^{(0)}}{\hat{\alpha}^{(0)}})$. Figure 5.1c shows that Newton's method iterates for most combinations of initial values, since most of the area of the SRR and Iterations plot is non-white. The only white parts of the two graphs are the bottom row and left column: the bottom row corresponds to an initial estimate of attack time $\hat{n}^{(0)}_I$ that is close to zero, and the left column corresponds to $\hat{\alpha}^{-1(0)}$ values that result in an envelope whose time spread is too small for it to have sufficient bandwidth, more specifically, the envelope's "effective" duration is less than f_c^{-1}.

5.2 Synthetic Audio Tests

5.2.1 One REDS

In this experiment we sample the REDS parameter set randomly to synthesize a signal from one atom, then pass it into PTMP to approximate it with one atom. We repeat this 100 times and record each run's SRR. To test whether noise influences PTMP's estimation accuracy, we repeat the sequence for different SNRs by adding Gaussian white noise to $\mathbf{y}$. The energy levels of the noise relative to the signal in dB are SNR $= (0, -20, \ldots -120)$. The box plots in Figure 5.2 show the reconstruction quality distribution for each SNR.

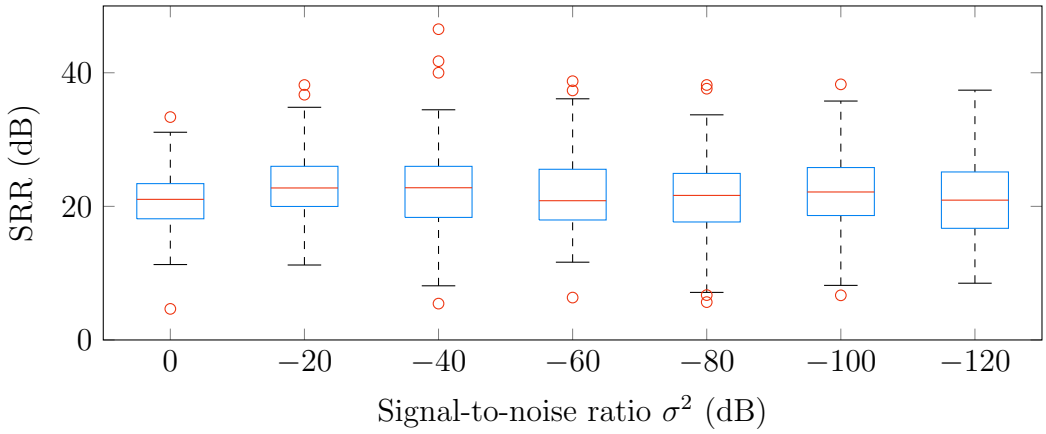


Figure 5.2 PTMP's single iteration approximation of a REDS atom in noise have these SRR values. The data set has 100 samples per SNR.

5.2.2 Multiple REDS

For the second experiment, we test PTMP's ability to decompose multiple superimposed REDS in additive Gaussian white noise. It involves synthesizing a signal from 20 REDS whose parameter sets are random, then decomposing the signal by running 20 iterations of PTMP. We average the residual energy after 20 runs at each level of SNR, and show the reconstruction quality distribution for each SNR in Figure 5.3. Figure 5.4 shows the sonogram of an example test signal, along with the sonogram of PTMP's signal approximation. From this plot, we see that PTMP performs excellently because it is able to locate every atom except one low-energy one (see around 16 kHz, 0.5 seconds).

5.2.3 Symmetric

Given that a benchmark for matching pursuit algorithms is how well they decompose asymmetric content with symmetric atoms [16] [17] [59], we conducted a reverse test to see how well PTMP approximates symmetric content with asymmetric atoms. We performed first a control test by decomposing an asymmetric one, the synthetic damped sinusoid. Then, in one test we decomposed a Gabor atom, and in another test, two superimposed Gabor atoms, with asymmetric (REDS) atoms. Figure 5.5 shows the time-domain distribution of the asymmetric atoms and the time-frequency energy distribution[2] of those atoms.

[2] [7] calls this time-frequency energy distribution a *Wivigram*. We create a Wivigram by superimposing each atom's Wigner-Ville distribution.

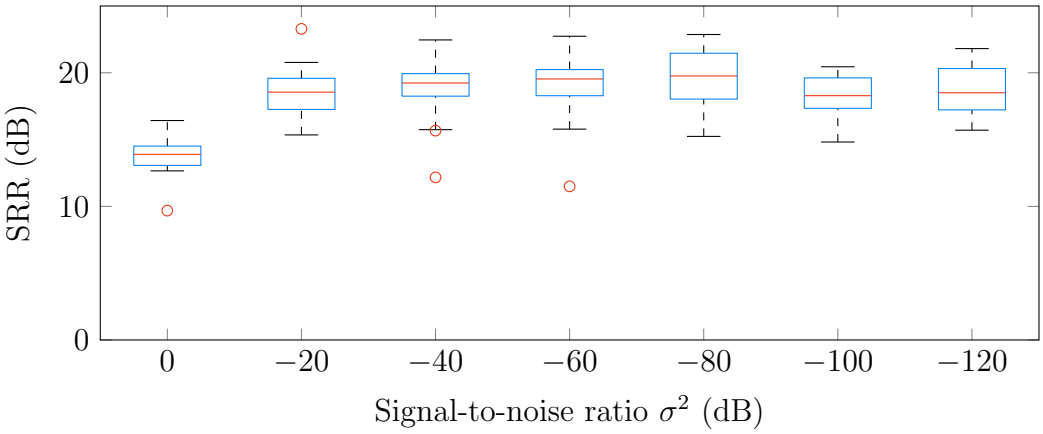


Figure 5.3 PTMP's 20-iteration approximation of 20 REDS atoms in noise have these SRR values. The data set has 20 samples per SNR.

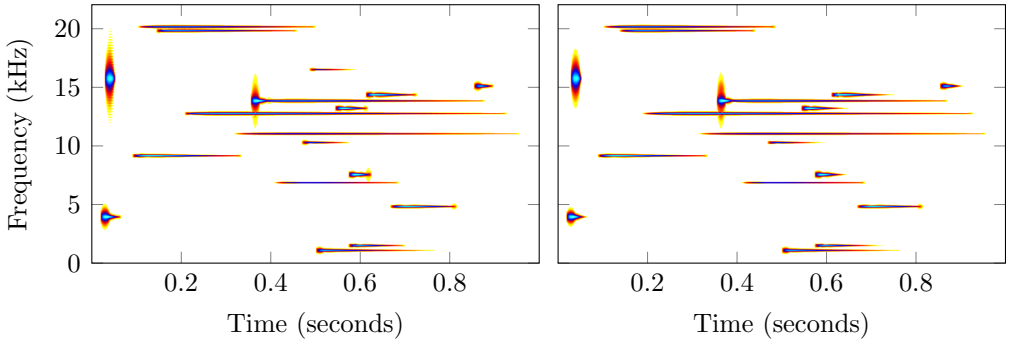
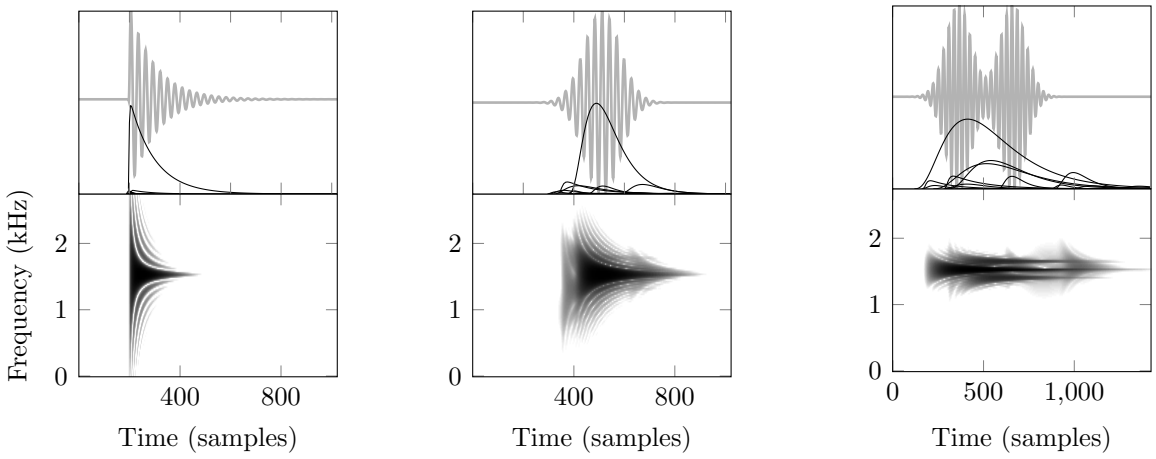


Figure 5.4 Sonogram of the target audio signal $\mathbf{y}$ (left), made of 20 REDS atoms, and the 20 iteration PTMP approximation (right).

Figures 5.5b and 5.5c show how there is no pre-echo before the symmetric signals and there is some dark energy after them. The dual-Gabor test illustrates that the locally optimal choice of first atom spans both Gabor blobs. There are some REDS atoms at slightly higher and lower frequencies than the signal's center frequency that create an interference pattern (i.e., a beat) to match the dual-Gabor's amplitude modulation. A symmetric version of this test in [7] shows that a 60 dB approximation of a damped sinusoid involves 60 to 100 Gabor atoms. Our test shows that a 60 dB approximation of a Gabor atom involves 40 REDS atoms, and the dual-Gabor requires 68 atoms.



(a) Damped sinusoid. (b) Gabor. (c) Dual-Gabors.

Figure 5.5 PTMP approximation of synthetic asymmetric and symmetric audio (SRR = 60 dB). The top graphs are the time domain distribution of REDS atoms, and bottom graphs are the Wivigrams.

5.2.4 Frequency Modulation

In the next experiments, we use synthetic test signals that are more difficult for PTMP to decompose to observe how it behaves when faced with audio features that do not fit with the asymmetric atom model, more precisely, with audio that has frequency modulation. When a dictionary does not include chirp atoms [15], MP approximates chirp signals by stringing together a progression of stationary atoms of increasing or decreasing frequencies. $\mathbf{y}$'s chirp rate determines the length of the stationary atoms; faster chirp rates call for atoms with shorter durations.

An interesting result comes from a test involving the decomposition of a synthetic source-filter model vocal sound. Figure 5.6 displays the partial trajectories of the signal, where the partial tracker did not constrain the frequency deviations to allow for a clear view of the frequency modulations (this is not representative of the partials that PTMP extracts), and the Wivigram from the PTMP approximation. We note that when there is no frequency modulation, in the time span from zero seconds to around 1.3 seconds, PTMP creates long atoms from partial trajectory data. Since PTMP's partial trajectories break at slight frequency changes, partials last only a few frames, or less, where there is vibrato (frequency modulation). Thus, PTMP chooses smaller REDS atoms to represent the vibrato, which starts at around 1.4 seconds and continues onward. In this representation, we see a clear

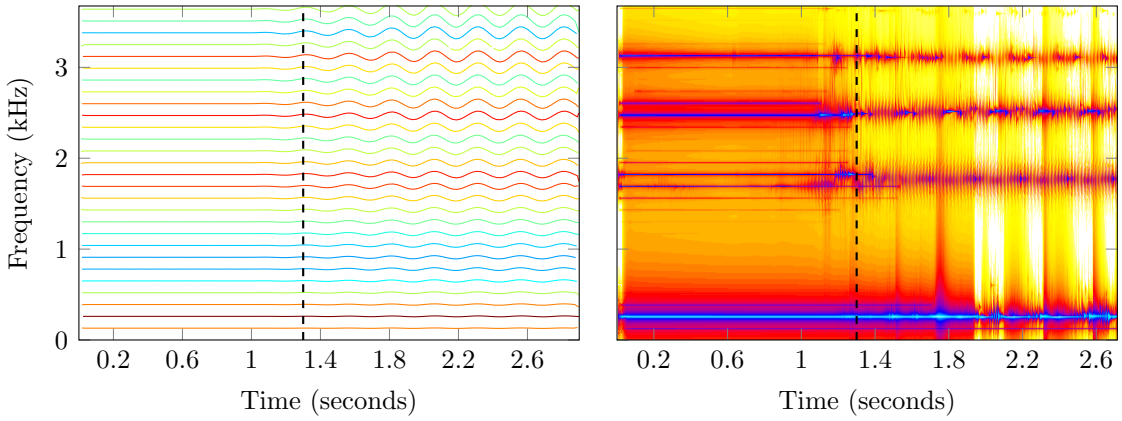


Figure 5.6 Partials (left) and Wivigram (right) of a source-filter synthesized vocal sound that PTMP approximated. Vibrato begins after the vertical dashed line.

transition at around 1.3 seconds. PTMP represents the non-vibrato section with long atoms from partials (reminiscent of the additive synthesis model) and approximates the section with vibrato with short-duration REDS atoms whose temporal spacings fluctuate in response to the vibrato (reminiscent of the source-filter model).

5.3 Real Audio Tests

5.3.1 Instrument Excerpts

For the first real audio experiment, PTMP decomposed thirteen acoustic and electronic musical instrument excerpts. The instrument set spanned the musical instrument families. Visuals of the audio excerpt waveforms along with the Wivigrams are in Figures 5.7, 5.8 and 5.9. For each excerpt, PTMP ran until its approximation reached an SRR of 30 dB. PTMP settings for this experiment are in Table 5.1. We include a summary of the results in Table 5.4.

PTMP sparsely represents percussive instruments, for example, the vibraphone solution has a sparsity of $\|\mathbf{x}\|_0 = 7$. Since PTMP adapts a REDS atom's attack shape to sparsely represent the signal's onsets, there is no pre-echo or dark energy in the final or intermediate stages of the decomposition, see Figures 5.7a, 5.7b, 5.8c, 5.7c, and 5.8d. Partial trajectories also help to minimize $\|\mathbf{x}\|_0$ by successfully locating long asymmetric atoms for the instruments that have freely decaying resonances. Notice in Figures 5.7c and 5.7e how only

	Variable	Experiment 1	Experiment 2
Partial Tracker	N_{PT}	8192	8192
	H	256	
	G_g	30 dB	25 dB
	G_h	10 dB	7 dB
	ν_δ	2 dB	
	ω_Δ	.015	
	ω_δ	.01	
Small Dictionary $\mathbf{\Phi}_s$	s	$(6, 7, 8, 9)$	$(6, 7, \ldots, 11)$
	N_s	2^s	
	p	3	
	α_s	$-\log(.001)/N_s$	
	β_s	α_s	$2\alpha_s$
Refinement Steps	NM Attack	40	20
	NM Envelope	20	30
	RRM Frequency	5	7

Table 5.1 PTMP settings for the real audio signal tests.

a few atoms represent most of $\mathbf{y}$ because their durations and their amplitude modulations follow $\mathbf{y}$'s damped oscillations.

On the other hand, the approximation is not especially sparse for instruments that involve a continuous/sustaining excitation, however, representing these classes of audio is difficult for sparse approximation algorithms in general. More specifically to PTMP, partial trajectories locate some long atoms, however, since the amplitude modulation for the instruments without freely decaying resonances do not follow the asymmetric atom model, $\|\mathbf{r}\|_2^2$ is relatively high even after the large atom decompositions and so $\mathbf{\Phi}_s$-located atoms (short atoms) must compensate. Results from the decomposition experiment in Chapter 3 (see Table 3.1) reinforce this fact because they show that different types of asymmetric atoms approximate the vocal signal to similar SRR values since the vocal sound does not involve freely decaying resonances (long atoms), while the FOF and REDS atoms clearly outperform the others in terms of sparsity for the vibraphone signal because it involves freely decaying resonances (long atoms). A dictionary that contains atoms with frequency modulation and noise may help improve representation sparsity for signals that do not involve long atoms, because it may be able to represent the breath sounds from

Family	Instrument	Comments	Note	Duration	N	$\|\mathbf{x}\|_0$
Percussion	Piano		$A_4^\sharp$	3.63	160,000	2997
	Glockenspiel	Brass Mallet	$A_5^\sharp$	1.32	58,000	1127
	Vibes		$F_3^\sharp$	3.40	150,000	7
	Snare Drum		-	0.27	12,000	1115
	Kick Drum		-	0.36	16,000	58
Brass	Trumpet		$F_5^\sharp$	2.02	89,000	5146
	Tuba		E_1	2.49	110,000	2122
Strings	Violin	w. Vibrato	E_5	3.17	140,000	8897
	Guitar	Electric	$D_3^\sharp$	4.54	2,000,000	586
	Bass	Electric	$D_3^\sharp$	6.80	3,000,000	289
Woodwinds	Flute	w. Flutter	E_4	6.12	270,000	6983
	Clarinet	E♭	$F_4^\sharp$	3.20	141,000	3775
Vocal	Female	w. Vibrato	$F_4^\sharp$	0.95	42,000	1741

Table 5.2 PTMP musical instrument excerpt approximation results. N is audio signal $\mathbf{y}$'s length in samples and $\|\mathbf{x}\|_0$ is the number of atoms in the representation. The sampling rate is 44100 Hz for all signals. SRR = 30 dB for all decompositions.

instruments of the woodwind and brass families, bowing sounds from string instruments, and so on. Interestingly, PTMP represents sounds with relatively large bandwidths, like the noise from a violin bow, with a sequence of REDS atoms that have attack influence times that are short relative to the decay rate (i.e., atoms with wide skirt widths).

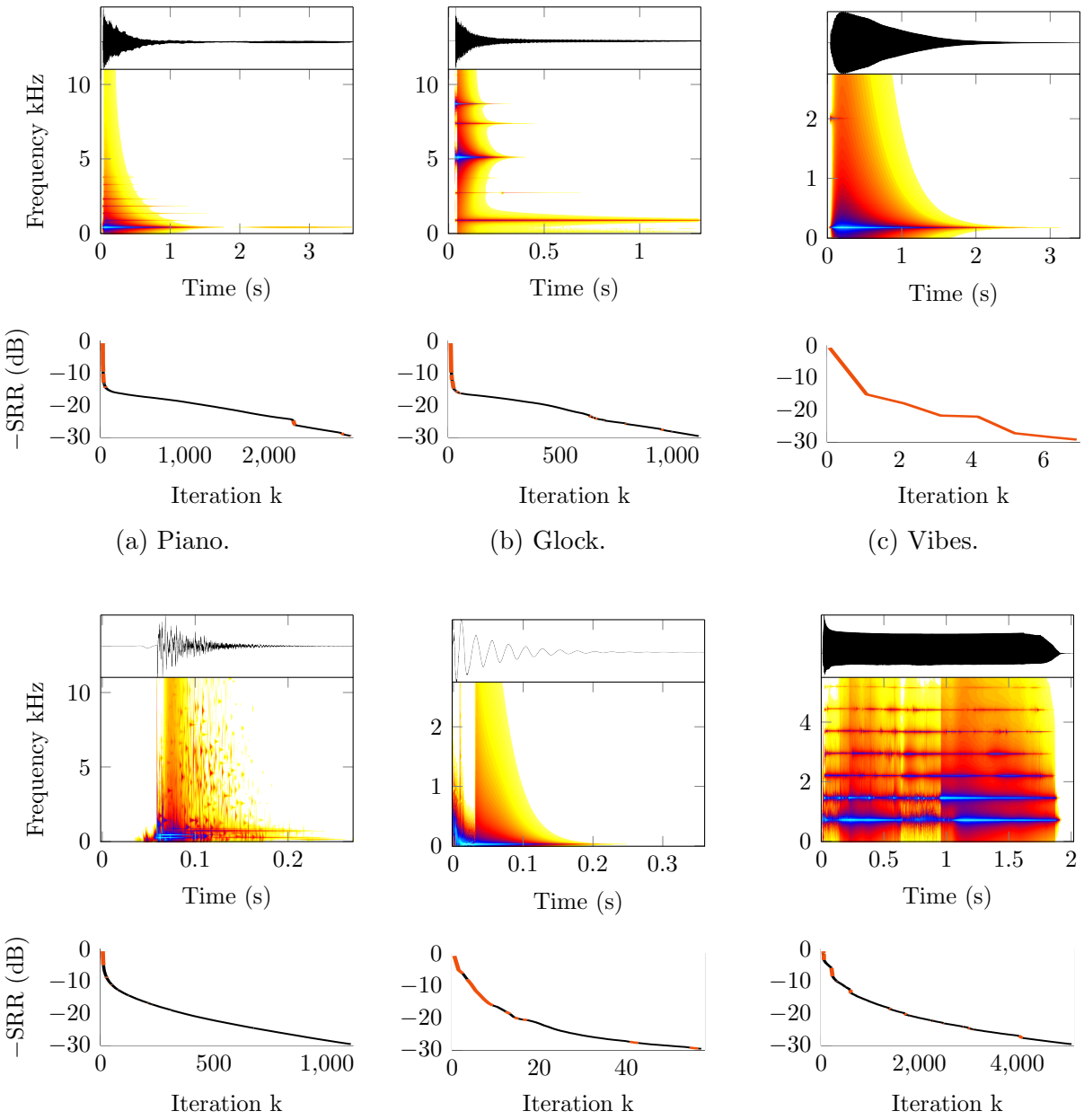
5.3.2 Music

Finally, we test PTMP's ability to sparsely represent excerpts of musical pieces. These are the descriptions of the music test signals:

1. Glock: A glockenspiel playing a melody. The sustain of the notes overlap in time, and some are at the same frequency.

2. Bach: A live solo piano recording of Bach's Chromatic Fantasy that contains audience and venue noise. This segment is the opening fast melodic line and chord.

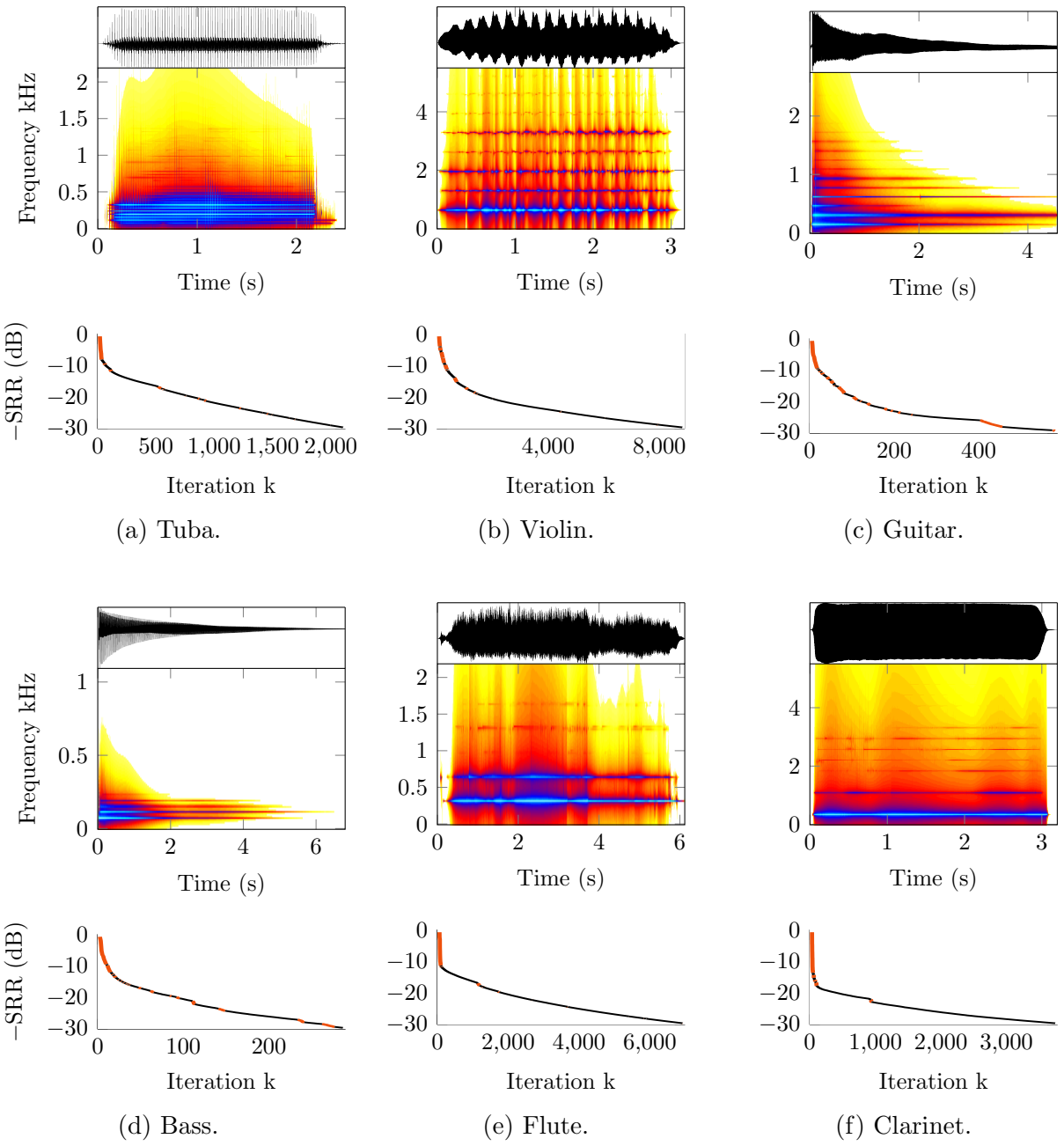
3. Mamavatu: A studio recording of a world music song. The instruments in the song are the tabla, bass, acoustic guitar, and female vocal with vibrato.



(a) Piano. (b) Glock. (c) Vibes.

(d) Snare. (e) Kick. (f) Trumpet.

Figure 5.7 Instrument excerpt results (1/3). $\mathbf{y}$ (top), Wivigram (middle), and $\mathbf{r}$ energy evolution (bottom), where atoms from partials are in red and $\mathbf{\Phi}_s$-located atoms are in black.

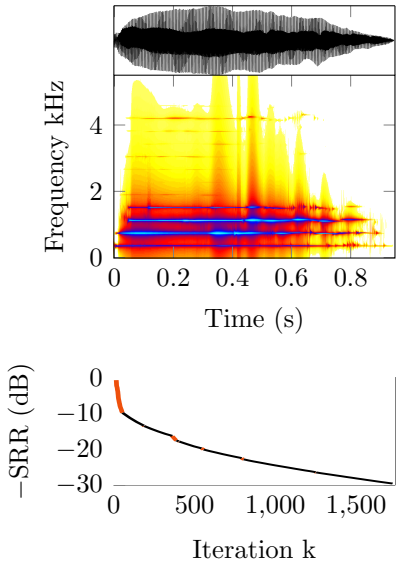


(a) Tuba. (b) Violin. (c) Guitar.

(d) Bass. (e) Flute. (f) Clarinet.

Figure 5.8 Instrument excerpt results (2/3).

4. Sing it Back: A studio recording of a pop song. The instruments in the song are the drum set, electric bass, electric guitar with modulation effects, and female vocals with vibrato.



(a) Female Vocal.

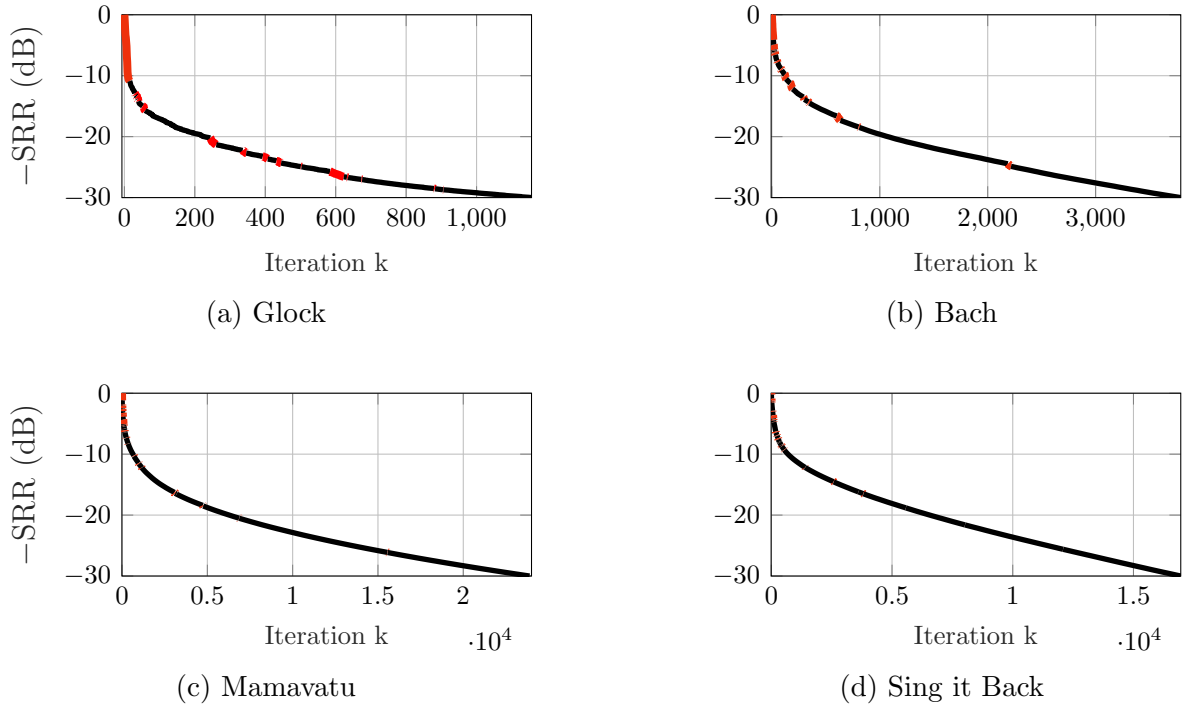
Figure 5.9 Instrument excerpt results (3/3).

Retrieving a sparse representation of the musical excerpts (especially the ones that contain contributions from multiple instruments) is more challenging than the instrument excerpts. The glockenspiel and Bach signals match well with the asymmetric atom model, though the Bach signal is more challenging because the piece involves a fast sequence of notes and the recording includes live venue noise and reverberation. The last two signals are the most challenging for PTMP to create a high-quality sparse approximation because their multi-instrumental composition results in signals that are non-sparse in time and frequency. Table 5.3 includes the results of this test.

The SRR graphs in Figure 5.10 reveal PTMP's sub-system selection throughout the decomposition of the musical signals and approach rate to a -30 dB residual energy. In these graphs, the atoms from partial tracking are in red and the atoms from the small-scale dictionary are in black. The fast residual energy decrease for the glockenspiel signal is due to PTMP representing each of the 15 freely decaying notes with the first 15 atoms, see the steep red line that begins at $k = 0$. The partial tracker does not contribute as much for the other signals because multi-instrument transient contributions segment the partials. For the full-band music signals, almost every atom, even from the first iterations, are from the small-scale dictionary. The intermediate case is the Bach signal: PTMP represents first

	Comments	Duration	N	$\|\mathbf{x}\|_0$
Glock	Melodic line	5.94	262,114	1156
Bach	w. venue noise	3.62	159,677	3789
Mamavatu		5.94	262,114	23,887
Sing it Back		3.86	85,219	16,959

Table 5.3 PTMP decomposition of music. N is audio signal $\mathbf{y}$'s length in samples and $\|\mathbf{x}\|_0$ is the number of atoms in the representation. The sampling rate is 44100 Hz, except "Sing it Back", which is 22050 Hz. SRR = 30 dB for all decompositions.



(a) Glock (b) Bach

(c) Mamavatu (d) Sing it Back

Figure 5.10 Residual energy evolution of music decomposition test. Atoms located by partial tracking are in red and atoms located from $\mathbf{\Phi}_s$ are in black.

the notes of the piano melody, however, the piano is not sustaining throughout the melody, so the atoms durations are smaller than those from the glockenspiel signal.

The full-band music signals (Mamavatu and Sing it Back) have dense spectra from drum-set induced noise bursts, multi-band compression, and frequency modulation. Creating an approximation of 30 dB SRR requires many small atoms to represent their highly non-stationary and stochastic content. For full-band audio, incorporating atoms of filtered

	Number of Atoms		Elapsed Time (min)	
Instrument	MP	PTMP	MP	PTMP
Guitar	4913	1059	1.26	1.54
Vibraphone	363	65	0.12	0.09
Piano	1141	346	0.32	0.31
Trumpet	1232	1168	1.14	1.14
Glockenspiel	4141	2026	1.35	1.49

Table 5.4 Results from 30 dB SRR approximation test using a 2.6 GHz Intel© Core i7 machine.

noise into the dictionary may help to represent that content, though a discussion on sparse representations of noise is out of the scope of this thesis. Gammatone-like REDS atoms produce similar results as Gaussian atoms at the later stages of the pursuit when the algorithm already represented the tonal parts. Thus, depending on whether the signal has content that fits with the REDS model, PTMP with REDS atoms can represent a signal with either more or equal sparsity than a matching pursuit with symmetric atoms. Moreover, PTMP avoids creating pre-echo by adapting REDS atoms to $\mathbf{y}$.

5.4 Comparison with Existing Techniques

In this section, we apply the following algorithms to a real audio decomposition problem and compare their performance: (MP) a fast matching pursuit that uses MPTK techniques; (MP-r) MP with RRM frequency and onset refinement and Newton's method envelope refinement; (PTMP) partial tracking matching pursuit.

First, we decompose real audio excerpts of a guitar, vibraphone, piano, and trumpet, and a melodic line from a glockenspiel, using MP and PTMP. We record the number of atoms and computation time for them to reach a 30 dB SRR approximation and show the results in Table 5.4. When the signal contains slowly decaying resonances, such as the piano, vibraphone, or guitar, PTMP's approximation is significantly more sparse than MP's. When a signal does not contain long temporally asymmetric features, PTMP circumvents partial tracking and effectively equals MP. This desirable behavior is reflected in the trumpet decomposition, as MP and PTMP's approximations have roughly the same sparsity and computation time.

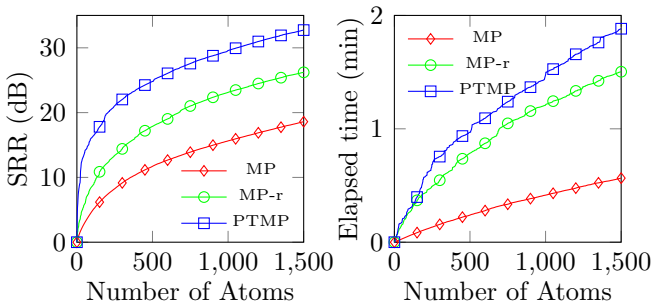


Figure 5.11 Results from equal static dictionary test.

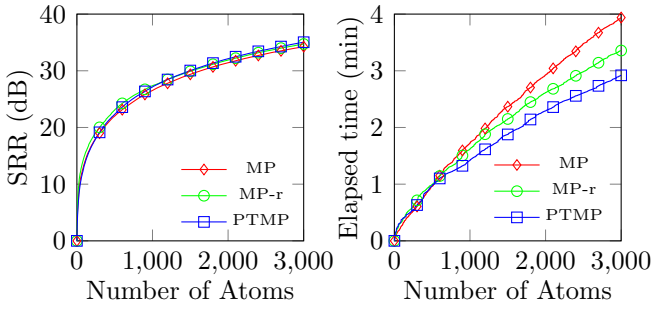


Figure 5.12 Results from equal SRR evolution test.

Next, we compare algorithm convergence rates by using the same dictionary across methods to decompose the guitar signal. Figure 5.11 shows that PTMP converges faster than MP and MP-r, and reaches some SRR value in roughly the same time as MP and MP-r with the fewest number of atoms (most sparsity). To further investigate algorithm speed, we customize the static dictionaries such that the SRR evolutions were approximately equivalent. The results in Figure 5.12 show that PTMP is the fastest algorithm because its static dictionary need not include those long duration atoms that slow MP's inner product updates; it finds them through partial trajectory data. Similarly, MP-r is faster than MP mainly because its refinement steps enable a smaller static dictionary.

These results demonstrate that PTMP can create audio signal approximations that are more sparse than MP in equal time. Bridging a fast MP search for short duration atoms with partial tracking and conversion of long duration atoms ensures that PTMP performance is better than or equal to MP for an arbitrary signal, depending on whether it contains slowly decaying resonances or not.

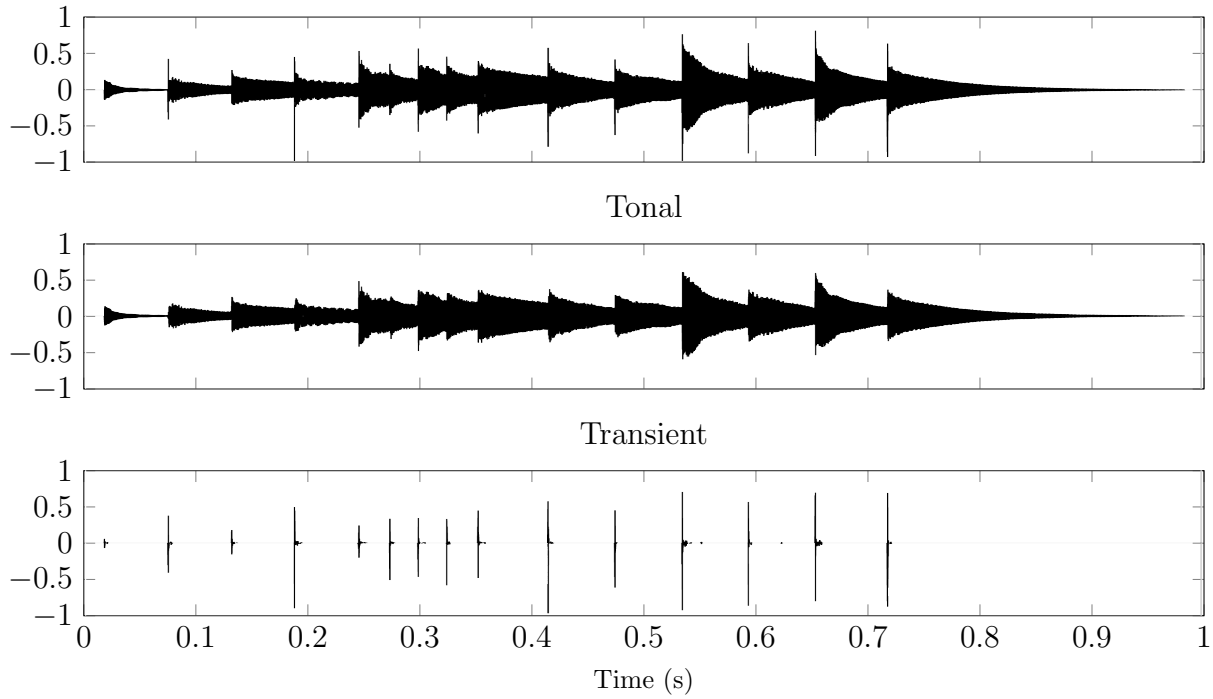
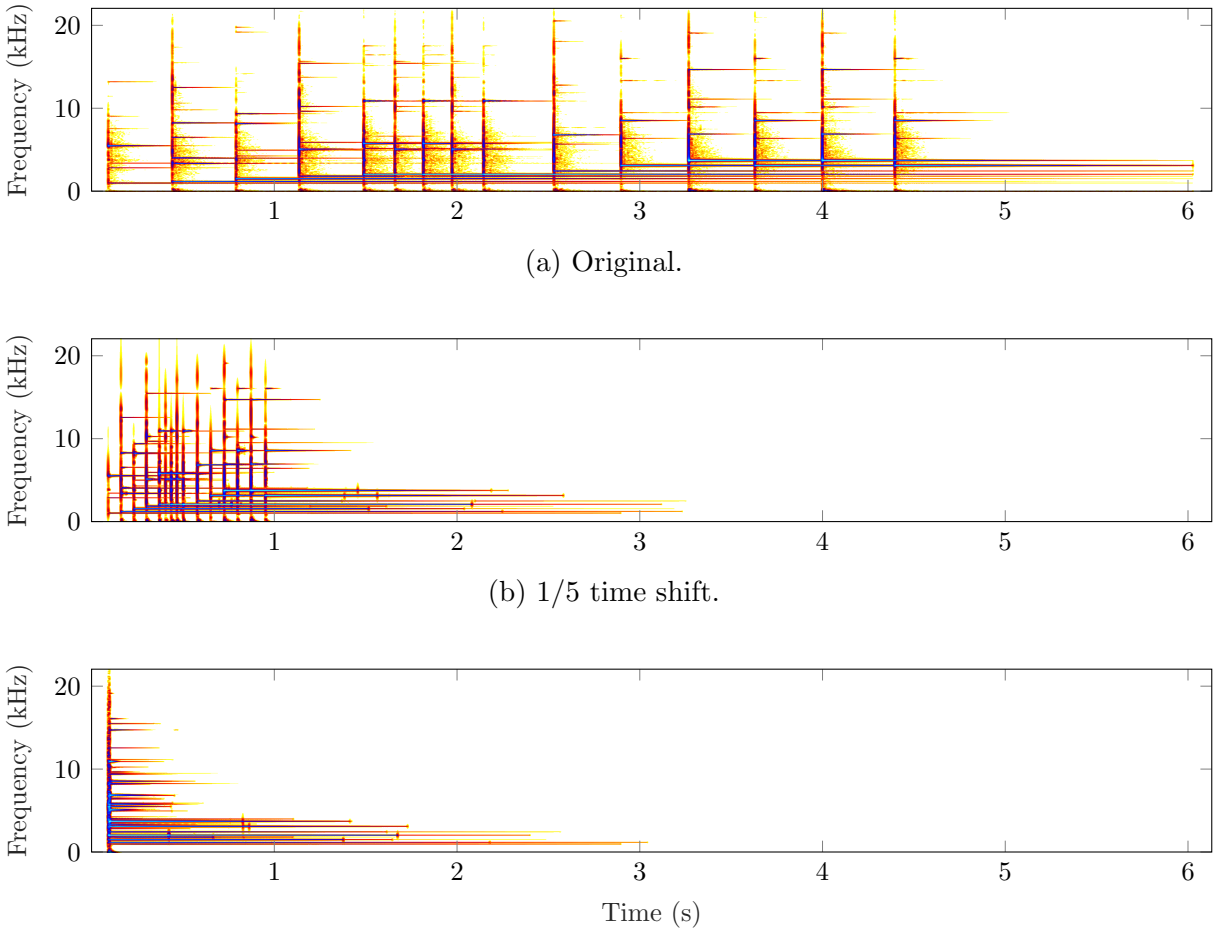


Figure 5.13 Glockenspiel tonal and transient separation after PTMP decomposition.

5.5 Post-processing

We now experiment with some audio transformations of the glockenspiel signal by manipulating the REDS parameter set $\boldsymbol{\lambda}$ that PTMP estimated. In the first experiment, we segment the parameter set in two: one set includes atoms with $N > 512$ and the other has atoms with $N \leq 512$. Synthesizing signals from each of the two parameter sets results in an approximate transient and tonal part separation, see Figure 5.13.

Next, we change the values of τ and hold the other parameters constant. Changing only τ alters the spacing between atoms but does not alter the reconstruction's perceptual tonal character or sound quality. Compacting the spacing of the atoms by multiplying every τ by $\frac{1}{5}$ makes the tempo of the performance five times faster, which sounds like the percussionist is playing a glissando, see Figure 5.14b. Even further, we reduce τ such that all the atoms play at once, so that the melodic sequence becomes one chord, see Figure 5.14c. Since we do not change the decay or attack, each note sustains for the same amount of time as the original approximation. With an algorithm that creates pre-echo and dark



(a) Original.

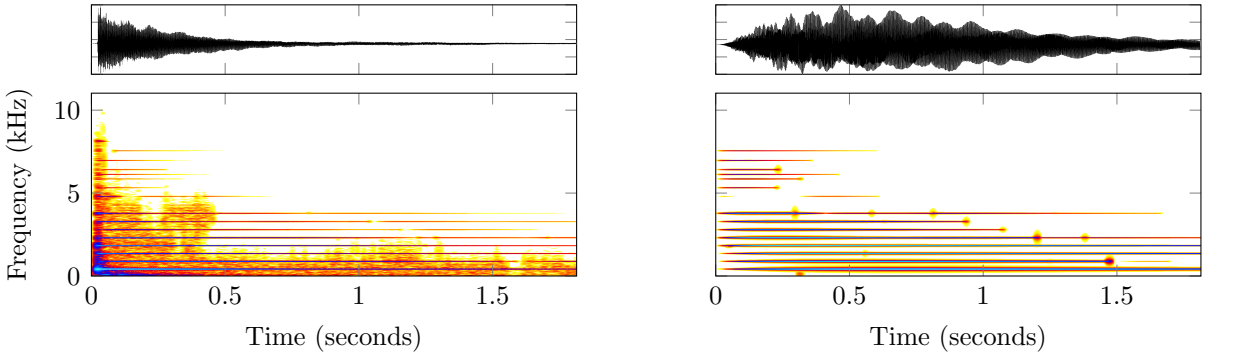
(b) 1/5 time shift.

(c) 1/1000 time shift.

Figure 5.14 Sonogram of the glockenspiel signal before and after time shifting.

energy in the intermediate stages of approximation, shifting the atoms in time will most likely undo the dark energy region phase cancellations and re-surface pre-echo artifacts [7]. Since PTMP's sparse approximation usually has no perceptual pre-echo or dark energy at the onsets, the onsets of the notes remain intact even after time shifting. Thus the signal has a natural sound even after transformation. If we decompose onto only short duration atoms with MP, time shifting ruins implicit co-atom phase relationships and reduces not only the global duration of the excerpt but the duration of each individual note.

In the final experiment, we manipulated atom attack shapes by varying β values. This effect works particularly well for sounds with amplitude envelopes that match closely with



(a) Original.

(b) After multiplying every atom's β value by 1/100.

Figure 5.15 Piano excerpt before and after manipulating the attack shape through β. Time domain waveforms are in the top panels and sonograms are in the bottom panels.

REDS atoms (i.e., with an attack portion followed by a long decay), for example, sounds that originate from a piano, glockenspiel, or guitar. We varied the attack shapes of the set of atoms that PTMP determined from the piano excerpt in Section 5.3.1. We successfully controlled the piano signal approximation's attack shape, from the original signal's fast attack all the way to smooth attack whose influence time was over a second. Figure 5.15 shows the results of multiplying every β value in $\boldsymbol{\lambda}$ by $\frac{1}{100}$. Notice that the original waveform's envelope resembles a damped exponential, while the processed waveform's envelope more closely resembles a gammatone's envelope. After performing this manipulation, the resulting audio signal sounds like a bowed instrument rather than a piano partly because it has a gradual increase in energy followed by a sustaining sound rather than a relatively fast increase followed by a decay, and also because the dense frequency content from the original signal's transient region is not prevalent in the audio signal post-manipulation.

5.6 Summary

This chapter included practical tests of PTMP's ability to decompose audio signals. It showed how REDS can adapt to signal features with the help of Newton's method and the Recursive Reassignment Method. Results from our synthetic and real audio experiments support our hypothesis that the asymmetric atom model fits excellently with audio produced from plucked string and percussive instruments, and that gammatone-like asym-

metric atoms have similar abilities to represent highly non-stationary and sustained parts of audio as symmetric Gaussian atoms. We highlighted that PTMP does not sparsely represent frequency modulations, though it provides a unique signal representation because it jointly models sound through the additive and source-filter models. We showed how the manipulation of the REDS parameters leads to different post-processing effects like time stretching and transposition. Overall, PTMP proves to be a powerful sparse audio approximation tool.

Chapter 6

Conclusion

6.1 Summary

In this thesis, we addressed the problem of creating a sparse audio representation from a practical and theoretical position. The main goal of this thesis was to explore the performance of atoms that are sinusoids with temporally asymmetric (causal) amplitude modulations.

We generalized the form of an asymmetric atom to compare existing asymmetric functions that permeated into the realm of sparse approximations, the result of which led to the creation of a new asymmetric atom, REDS, that not only fits excellently with a range of audio features, but also through its mathematical properties allow estimation methods to determine its parameters. We developed methods to estimate all the parameters of the new asymmetric atom, including the derivations for Newton method refinement and established when Newton's method in multiple dimensions converges to parameter values close to ground-truth.

We improved the greedy sparse approximation algorithm's ability to scale to an audio signal of arbitrary size by bridging two atomic search methods, one that employs sinusoidal partial trajectory data to locate atoms of long duration and another that relies on the classic static dictionary correlation-based search. From this research, we achieved our general goal of extracting sparse representations of audio signals that contain strong transients, for example, audio with percussion and plucked-string instruments.

6.2 Future Work

Now we direct the reader towards future work based upon the research of this thesis. Since our asymmetric atom model assumes a constant frequency, the representation of signals with frequency modulation, like a singing voice with vibrato, are not especially sparse. A natural extension of this research incorporates frequency modulation into REDS through an additional parameter. Then, partial pursuit must adopt frequency modulation estimation methods, for example, by removing the partial frequency deviation constraint. Gribonval researched atom chirp rate estimation [15], though the details of this method are specific to Gabor atoms. The Recursive Reassignment Method may extend to estimate chirp rate rather rapidly. The source-filter model represents frequency modulations as a sequence of constant frequency pulses, so although the signal representation will probably be more sparse after incorporating atoms with frequency modulation support, it will no longer reflect the source-filter model.

During our search for a new asymmetric atom that we describe in Section 3.3.5, we discovered an atom that we call the *betatone (BT)* due to the relation between its amplitude envelope definition and the beta function:

$$\phi_{BT}[n] = n^p(N-n-1)^\alpha e^{i\omega_c n} \tag{6.1}$$

This atom does not fit the general asymmetric atom form (3.2) since it does not involve a damped exponential, and its mathematical construction may not admit simple analytic formulas. Regardless, the betatone has some powerful properties that are worth mentioning as a subject of future research. For one, the BT's envelope is continuously differentiable within the finite span $0 \leq n \leq N-1$, and since it equals zero at both the boundaries ($n = 0$ and $n = N-1$), it does not create any discontinuity artifacts like the other asymmetric atoms. Moreover, depending on the ratio between p and α, the envelope "leans" to either the right or left; its asymmetry may be in either time direction. BT's envelope is temporally symmetric when $p = \alpha$. We see potential for the betatone because of its envelope's malleability, which can adapt to symmetric or asymmetric signal content.

Another future work path is to extend partial pursuit to search for symmetric Gabor atoms. With the exception of Section 4.2, the partial pursuit algorithm also works for symmetric atoms. The main difference is rather than transforming the partial data into

REDS, it will transform into a symmetric atom. Furthermore, a worthwhile effort involves the meshing of REDS with Gabor atoms in the system because it may result in a sparser representation, especially since sustaining sounds, like from the trumpet, do not fit with the asymmetric atom model, but do fit with sound from a superposition of time shifted Gabor atoms. It is worthwhile to explore whether a hybrid dictionary of only two different types of atoms, REDS and Gabor atoms, is sufficient to sparsely represent the sounds from the majority of musical instruments, since research has, to the best of our knowledge, yet to accomplish that goal. Since the incorporation of symmetric atoms into partial pursuit also re-introduces the possibility of pre-echo creation, partial pursuit needs heuristics that select the atom shape, whether symmetric or asymmetric, that creates the least dark energy. Furthermore, the distribution and type of the long duration atoms from the approximation, whether symmetric or asymmetric, may provide clues as to what source the atom is representing, which may be useful for source-separation and classification applications.

Appendix A

REDS partial derivatives

Newton's method searches for a real-valued REDS atom, $\boldsymbol{\phi}_{REDS} = \boldsymbol{A}\boldsymbol{\phi}_{DS}$, where

$$\boldsymbol{A} = \left(1 - e^{-\beta(n-\tau)}\right)^p \, u[n-\tau] \tag{A.1}$$

starting from a damped sinusoid,

$$\boldsymbol{\phi}_{DS} = e^{-\alpha(n-\tau)} \cos(\omega_c(n-\tau) + \theta) \tag{A.2}$$

by estimating values for β, τ, and α. Newton steps in multiple dimensions refine τ and θ then α and β.

The following lists the REDS partial derivatives with respect to each variable. Starting with τ,

$$\frac{\partial \boldsymbol{\phi}}{\partial \tau} = \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \tau}\boldsymbol{A} + \boldsymbol{\phi}_{DS}\frac{\partial \boldsymbol{A}}{\partial \tau} \tag{A.3}$$

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \tau^2} = \frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \tau^2}\boldsymbol{A} + 2\frac{\partial \boldsymbol{\phi}_{DS}}{\partial \tau}\frac{\partial \boldsymbol{A}}{\partial \tau} + \boldsymbol{\phi}_{DS}\frac{\partial^2 \boldsymbol{A}}{\partial \tau^2} \tag{A.4}$$

where

$$\frac{\partial \boldsymbol{\phi}_{DS}}{\partial \tau} = e^{-\alpha(n-\tau)} \left(\alpha \cos(\omega_c(n-\tau)+\theta) + \omega_c \sin(\omega_c(n-\tau)+\theta)\right) \tag{A.5}$$

$$\frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \tau^2} = \alpha \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \tau} + e^{-\alpha(n-\tau)} \left(\alpha\omega_c \sin(\omega_c(n-\tau)+\theta) - \omega_c^2 \cos(\omega_c(n-\tau)+\theta)\right) \tag{A.6}$$

$$\frac{\partial \boldsymbol{A}}{\partial \tau} = -p\beta e^{-\beta(n-\tau)}(1-e^{-\beta(n-\tau)})^{p-1}\, u[n-\tau] \tag{A.7}$$

$$\frac{\partial^2 \boldsymbol{A}}{\partial \tau^2} = \beta \frac{\partial \boldsymbol{A}}{\partial \tau} + p(p-1)\beta^2 e^{-2\beta(n-\tau)}(1-e^{-\beta(n-\tau)})^{p-2} u[n-\tau] \tag{A.8}$$

The partial derivatives with respect to θ are

$$\frac{\partial \boldsymbol{\phi}}{\partial \theta} = \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \theta} \boldsymbol{A} \tag{A.9}$$

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \theta^2} = \frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \theta^2} \boldsymbol{A} \tag{A.10}$$

where

$$\frac{\partial \boldsymbol{\phi}_{DS}}{\partial \theta} = -e^{-\alpha(n-\tau)} \sin(\omega_c(n-\tau)+\theta) \tag{A.11}$$

$$\frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \theta^2} = -e^{-\alpha(n-\tau)} \cos(\omega_c(n-\tau)+\theta) \tag{A.12}$$

The mixed partial derivative with respect to τ and θ is

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \tau \partial \theta} = \frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \tau \partial \theta} \boldsymbol{A} + \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \theta} \frac{\partial \boldsymbol{A}}{\partial \tau} \tag{A.13}$$

where

$$\frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \tau \partial \theta} = e^{-\alpha(n-\tau)} \left(-\alpha \sin(\omega_c(n-\tau)+\theta) + \omega_c \cos(\omega_c(n-\tau)+\theta)\right) \tag{A.14}$$

The partial derivatives with respect to β are

$$\frac{\partial \boldsymbol{\phi}}{\partial \beta} = \boldsymbol{\phi}_{DS} \frac{\partial \boldsymbol{A}}{\partial \beta} \tag{A.15}$$

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \beta^2} = \boldsymbol{\phi}_{DS} \frac{\partial^2 \boldsymbol{A}}{\partial \beta^2} \tag{A.16}$$

where

$$\frac{\partial \boldsymbol{A}}{\partial \beta} = p(n-\tau)e^{-\beta(n-\tau)}(1-e^{-\beta(n-\tau)})^{p-1}\,u[n-\tau] \tag{A.17}$$

$$\frac{\partial^2 \boldsymbol{A}}{\partial \beta^2} = \left(p(p-1)(n-\tau)^2 e^{-2\beta(n-\tau)}(1-e^{-\beta(n-\tau)})^{p-2} - (n-\tau)\frac{\partial \boldsymbol{A}}{\partial \beta}\right) u[n-\tau] \tag{A.18}$$

The partial derivatives with respect to α are

$$\frac{\partial \boldsymbol{\phi}}{\partial \alpha} = \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \alpha}\boldsymbol{A} \tag{A.19}$$

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \alpha^2} = \frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \alpha^2}\boldsymbol{A} \tag{A.20}$$

where

$$\frac{\partial \boldsymbol{\phi}_{DS}}{\partial \alpha} = -(n-\tau)\boldsymbol{\phi}_{DS} \tag{A.21}$$

$$\frac{\partial^2 \boldsymbol{\phi}_{DS}}{\partial \alpha^2} = (n-\tau)^2\boldsymbol{\phi}_{DS} \tag{A.22}$$

The mixed partial derivative with respect to α and β is

$$\frac{\partial^2 \boldsymbol{\phi}}{\partial \alpha \partial \beta} = \frac{\partial \boldsymbol{\phi}_{DS}}{\partial \alpha}\frac{\partial \boldsymbol{A}}{\partial \beta}. \tag{A.23}$$

Bibliography

[1] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, pp. 3397–3415, Dec. 1993.

[2] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comp.*, vol. 20, pp. 33–61, Aug. 1998.

[3] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput.*, vol. 24, pp. 227–234, Apr. 1995.

[4] D. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ_1 minimization," *Proc. Nat. Aca. Sci.*, vol. 100, pp. 2197–2202, Mar. 2003.

[5] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, pp. 2231–2242, Oct. 2004.

[6] M. Yaghoobi, T. Blumensath, and M. Davies, "Dictionary learning for sparse approximations with the majorization method," *IEEE Trans. Signal Process.*, vol. 57, pp. 2178–2191, Dec. 2009.

[7] B. Sturm, C. Roads, A. McLeran, and J. Shynk, "Analysis, visualization, and transformation of audio signals using dictionary-based methods," *Journal of New Music Research*, vol. 38, pp. 325–341, Dec. 2009.

[8] C. Kereliuk and P. Depalle, "Sparse atomic modeling of audio: A review," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-11)*, (Paris, France), pp. 81–92, Sep. 2011.

[9] M. Plumbley, T. Blumensath, and L. Daudet, "Sparse representations in audio and music: from coding to source separation," in *Proc. IEEE*, vol. 98, pp. 995–1005, Jun. 2010.

[10] S. Krstulovic and R. Gribonval, "MPTK: Matching pursuit made tractable," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 3, pp. 496–499, May 2006.

[11] L. Daudet, "Sparse and structured decompositions of signals with the molecular matching pursuit," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 14, pp. 1808–1816, Sep. 2006.

[12] L. Daudet, "Audio sparse decomposition in parallel," *IEEE Signal Process. Magazine*, pp. 90–96, Mar. 2010.

[13] S. Mallat, *A Wavelet Tour of Signal Processing*. Academic Press, third ed., 2009.

[14] D. Gabor, "Theory of communication," *J. IEE*, vol. 93, no. 3, pp. 429–457, 1946.

[15] R. Gribonval, "Fast matching pursuit with a multiscale dictionary of Gaussian chirps," *IEEE Trans. Signal Process.*, vol. 49, pp. 994–1001, May 2001.

[16] B. Sturm, J. Shynk, L. Daudet, and C. Roads, "Dark energy in sparse atomic estimations," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, pp. 671–676, Mar. 2008.

[17] R. Gribonval, P. Depalle, X. Rodet, E. Bacry, and S. Mallat, "Sound signal decomposition using a high resolution matching pursuit," in *Proc. Int. Computer Music Conf. (ICMC)*, (Hong Kong, China), pp. 293–296, Aug. 1996.

[18] S. Jaggi, W. Carl, S. Mallat, and A. Willsky, "High resolution pursuit for feature extraction," *Applied Comput. Harmonic. Anal.*, vol. 5, pp. 428–449, Oct. 1998.

[19] M. Goodwin, "Matching pursuit with damped sinusoids," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 3, (Munich, Germany), pp. 2037–2040, Apr. 1997.

[20] R. Gribonval and E. Bacry, "Harmonic decomposition of audio signals with matching pursuit," *IEEE Trans. Signal Process.*, vol. 51, pp. 101–111, Jan. 2003.

[21] E. C. Smith and M. S. Lewicki, "Efficient auditory coding," *Nature*, vol. 439, pp. 978–982, Feb. 2006.

[22] S. Strahl and A. Mertins, "Analysis and design of gammatone signal models," *J. Acoust. Soc. Am.*, vol. 126, pp. 2379–2389, Nov. 2009.

[23] H. Najaf-Zadeh, R. Pichevar, H. Lahdili, and L. Thibault, "Perceptual matching pursuit for audio coding," in *Audio Eng. Soc. Conv.*, (Amsterdam, The Netherlands), May 2008.

[24] P. Honnet, G. Branislav, and P. Garner, "Atom decomposition-based intonation modelling," in *IEEE Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, (South Brisbane, Australia), pp. 4744–4748, Apr. 2015.

[25] T. Spustek, W. Jedrzejczak, and K. Blinowska, "Matching pursuit with asymmetric functions for signal decomposition and parameterization," *PLOS ONE*, vol. 10, pp. 1–19, Jun. 2015.

[26] A. Bregman, *Auditory Scene Analysis*. Cambridge, MA: MIT Press, 1990.

[27] N. Fletcher and T. Rossing, *The Physics of Musical Instruments*. Springer New York, 2nd ed., 1998.

[28] P. Balazs, M. Doerfler, M. Kowalski, and B. Torresani, "Adapted and adaptive linear time-frequency representations: A synthesis point of view," *IEEE Signal Process. Magazine*, vol. 30, pp. 20–31, Nov. 2013.

[29] M. Hayes, *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Inc., 1996.

[30] D. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, pp. 1289–1306, Apr. 2006.

[31] M. Goodwin, *Adaptive signal models: theory, algorithms and audio applications*. Springer New York, 1998.

[32] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. 27th Asilomar Conf. Signals, Systems and Computers*, vol. 1, (Los Alamitos, CA), pp. 40–44, Nov. 1993.

[33] S. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $L^2(R)$," *Trans. Am. Math. Soc.*, vol. 315, Sep. 1989.

[34] E. Bacry, "Lastwave software (GPL license)." `http://www.cmap.polytechnique.fr/~bacry/LastWave`, November 2012.

[35] M. Yaghoobi, L. Daudet, and M. Davies, "Parametric dictionary design for sparse coding," *IEEE Trans. Signal Process.*, vol. 57, pp. 4800–4810, Dec. 2009.

[36] K. Fitz, *The reassigned bandwidth-enhanced method of additive synthesis*. PhD thesis, University of Illinois at Urbana-Champaign, 1999.

[37] R. Badeau, R. Boyer, and B. David, "EDS parametric modeling and tracking of audio signals," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-02)*, (Hamburg, Germany), pp. 139–144, Sep. 2002.

[38] N. P. Donaldson, "Extending the phase vocoder with damped sinusoid atomic decomposition of transients," Master's thesis, McGill University, Aug. 2011.

[39] X. Rodet, *Time-domain formant-wave-function synthesis*, ch. 4-Speech Synthesis, pp. 429–441. J.C. Simon, Ed. New York, 1980.

[40] J. L. Flanagan, "Models for approximating basilar membrane displacement," *Bell System Technical Journal*, vol. 39, no. 5, pp. 1163–1191, 1960.

[41] P. Balazs, M. Dörfler, F. Jaillet, N. Holighaus, and G. Velasco, "Theory, implementation and applications of nonstationary gabor frames," *J. Comput. Appl. Math*, vol. 236, no. 6, pp. 1481 – 1496, 2011.

[42] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," in *Proc. IEEE*, vol. 66, pp. 51–83, Jan. 1978.

[43] F. W. Warner, *Foundations of Differentiable Manifolds and Lie Groups*. Springer New York, 1983.

[44] S. Marchand and P. Depalle, "Generalization of the derivative analysis method to non-stationary sinusoidal modeling," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-08)*, (Espoo, Finland), Sep. 2008.

[45] M. Mathews and J. Miller, *The technology of computer music*. Cambridge, Mass.: M.I.T. Press, 1969.

[46] J. Allen and L. Rabiner, "A unified approach to short-time Fourier analysis and synthesis," in *Proc. IEEE*, vol. 65, pp. 1558–1564, Nov. 1977.

[47] G. M. R. de Prony, "Essai expérimental et analytique sur les lois de la dilatabilité de fluides élastiques," *Journal de l'École Polytechnique*, vol. 1, no. 22, pp. 24–76, 1795.

[48] R. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, "An efficient auditory filterbank based on the gammatone function," *Paper presented at a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, Dec. 1987.

[49] R. Lyon, A. Katsiamis, and E. Drakakis, "History and future of auditory filter models," in *Proc. IEEE Int. Conf. Circuits and Systems (ISCAS)*, (Paris, France), pp. 3809–3812, Aug. 2010.

[50] R. McAulay and T. Quatieri, "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Trans. Acoustics, Speech, Signal Process.*, vol. 34, pp. 744–754, Aug. 1986.

[51] K. Fitz and L. Haken, "On the use of time–frequency reassignment in additive sound modeling," *J. Audio Eng. Soc.*, vol. 50, pp. 879–893, Sep. 2002.

[52] M. Lagrange, S. Marchand, and J. Rault, "Enhancing the tracking of partials for the sinusoidal modeling of polyphonic sounds," in *Proc. Int. Conf. on Digital Audio Effects (DAFx-03)*, (London, UK), pp. 1–6, Sep. 2003.

[53] X. Serra and J. Smith, "Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition," *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.

[54] X. Serra, *A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition*. PhD thesis, Stanford University, Oct. 1989.

[55] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *IEEE Trans. Signal Process.*, vol. 43, pp. 1068–1089, May 1995.

[56] B. Hamilton, P. Depalle, and S. Marchand, "Theoretical and practical comparisons of the reassignment method and the derivative method for the estimation of the frequency slope," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, (New Paltz, NY), pp. 345–348, Oct. 2009.

[57] P. Depalle, G. Garcia, and X. Rodet, "Tracking of partials for additive sound synthesis using hidden Markov models," in *IEEE Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, vol. 1, (Minneapolis, USA), pp. 225–228, Apr. 1993.

[58] T. Stockham, "High-speed convolution and correlation," in *1966 Spring Joint Comp. Conf., AFIPS Proc.*, vol. 28, (Washington, D.C.), pp. 229–233, Spartan Books, 1966.

[59] E. Ravelli, G. Richard, and L. Daudet, "Extending fine-grain scalable audio coding to very low bitrates using overcomplete dictionaries," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 195–198, Oct. 2007.